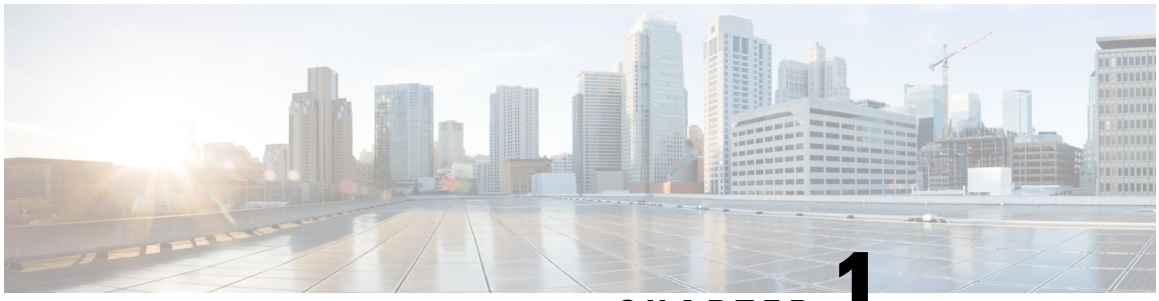
QoS Modular QoS Command-Line Interface Configuration Guide, Cisco IOS XE Everest 16.5

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

Read Me First

Important Information about Cisco IOS XE 16

Effective Cisco IOS XE Release 3.7.0E for Catalyst Switching and Cisco IOS XE Release 3.17S (for Access and Edge Routing) the two releases evolve (merge) into a single version of converged release—the Cisco IOS XE 16—providing one release covering the extensive range of access and edge products in the Switching and Routing portfolio.

Feature Information

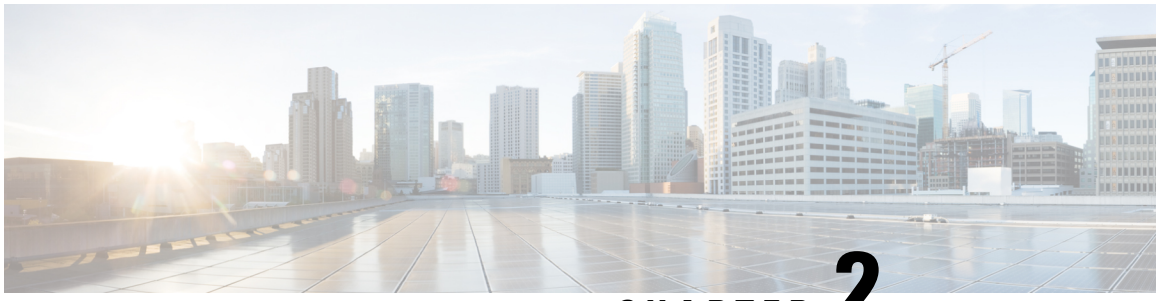
Use Cisco Feature Navigator to find information about feature support, platform support, and Cisco software image support. An account on Cisco.com is not required.

Related References

- Cisco IOS Command References, All Releases

Obtaining Documentation and Submitting a Service Request

- To receive timely, relevant information from Cisco, sign up at Cisco Profile Manager.
- To get the business impact you're looking for with the technologies that matter, visit Cisco Services.
- To submit a service request, visit Cisco Support.
- To discover and browse secure, validated enterprise-class apps, products, solutions and services, visit Cisco Marketplace.
- To obtain general networking, training, and certification titles, visit Cisco Press.
- To find warranty information for a specific product or product family, access Cisco Warranty Finder.



CHAPTER 2

Applying QoS Features Using the MQC

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.

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-based QoS does not support classification of legacy Layer 2 protocol packets such as Internetwork Packet Exchange (IPX), DECnet, or AppleTalk. When these types of packets are being forwarded through a generic Layer 2 tunneling mechanism, the packets can be handled by MQC but without protocol classification. As a result, legacy protocol traffic in a Layer 2 tunnel is matched only by a "match any" class or class-default.

The number of QoS policy maps and class maps supported varies by platform and release.

Note The policy map limitations do not refer to the number of applied policy map instances, only to the definition of the policy maps.

The following restrictions apply to Cisco IOS XE Release 3.5S for the Cisco ASR 903 router:

- QoS policy maps are not supported in sessions.

- Nested traffic maps are not supported.

For more information on restrictions for Cisco ASR 903 router, refer the Quality of Service Configuration Guidelines for Cisco ASR 903 Router.

Table 1: Cisco ASR 903 Policy and Class Map Support

	Cisco IOS XE 3.5S, 3.6S, 3.7S, 3.8S	Cisco IOS XE 3.9S and higher
Number of unique policy-maps	1024	1024
Number of unique class-maps	4096	4096
Number of classes per policy-map	512	4096
Number of filters per class-map	16	16

Table 2: Cisco ASR 1000 Series Policy and Class Map Support

	Cisco IOS XE 2.0S, 2.1S, 2.2S	Cisco IOS XE 2.3S	Cisco IOS XE 3.5S, 3.6S, 3.7S, 3.8S, 3.9S	Cisco IOS XE 3.10S and higher
Number of unique policy-maps	1024	4096	4096	4096 16000 (RP2, ESP40, ESP100, ESP200 models only)
Number of unique class-maps	4096	4096	4096	4096
Number of classes per policy-map	8	256	1000	1000
Number of filters per class-map	16	16	32	32

Table 3: Cisco CSR 1000V Policy and Class Map Support

	Cisco IOS XE 3.10S, 3.11S, 3.12S	Cisco IOS XE 3.13S and higher
Number of unique policy-maps	30	256
Number of unique class-maps	256	512
Number of classes per policy-map	32	512
Number of filters per class-map	8	16

Table 4: Cisco ISR 4000 Series Integrated Services Routers Policy and Class Map Support

	Cisco IOS XE 3.9.1S, 3.9.2	Cisco IOS XE 3.10S (ISR 4451 only)
Number of unique policy-maps	4096	4000
Number of unique class-maps	4096	4096
Number of classes per policy-map	1000	256
Number of filters per class-map	32	32

About

The MQC Structure

The MQC (Modular Quality of Service (QoS) Command-Line Interface (CLI)) enables you to set packet classification and marking based on a QoS group value. MQC CLI allows you to create traffic classes and policies, enable a QoS feature (such as packet classification), and attach these policies to interfaces.

The MQC structure necessitates developing the following entities: traffic class, policy map, and service policy.

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 XE release. For more information about the commands and command syntax, see the *Cisco IOS Quality of Service Solutions* Command Reference.

Table 5: 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 cos	Matches a packet based on a Layer 2 class of service (CoS) marking.

Command	Purpose
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 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. **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.
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. **Note** A separate **match protocol** (NBAR) command is used to configure network-based application recognition (NBAR) to match traffic by a protocol type known to NBAR.
match protocol fasttrack	Configures NBAR to match FastTrack peer-to-peer traffic.
match protocol gnutella	Configures NBAR to match Gnutella peer-to-peer traffic.

Command	Purpose
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 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.

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).

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.

Commands Used to Enable QoS Features

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 QoS Command Reference*.

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

Table 6: 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.
fair-queue	Enables the flow-based queueing feature within a traffic class.
fair-queue pre-classify	Configures and checks whether the **qos pre-classify** command can be used for fair queue. When the **qos pre-classify** command is enabled on the tunnel inteface, and then the **fair-queue pre-classify** command is enabled for the policy-map, the policy-map is attached to either the tunnel interface or the physical interface. The inner IP header of the tunnel will be used for the hash algorithm of the fair queue.
drop	Discards the packets in the specified traffic class.
police	Configures traffic policing.
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 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 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.

Command	Purpose
set [**ip**] **dscp**	Marks a packet by setting the differentiated services code point (DSCP) value in the type of service (ToS) byte.
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.

Nested Traffic Classes

The MQC does not necessarily require that you associate only one traffic class to one traffic policy.

In a scenario where packets satisfy more than one match criterion, the MQC enables you to associate multiple traffic classes with a single traffic policy (also termed nested traffic classes) using the **match class-map** command. (We term these *nested class maps* or *MQC Hierarchical class maps*.) This command provides the only method of combining match-any and match-all characteristics within a single traffic class. By doing so, you can create a traffic class using one match criterion evaluation instruction (either match-any or match-all) and then use that traffic class as a match criterion in a traffic class that uses a different match criterion type. For example, a traffic class created with the match-any instruction must use a class configured with the match-all instruction as a match criterion, or vice versa.

Consider this likely scenario: Suppose A, B, C, and D were all separate match criterion, and you wanted traffic matching A, B, or C and D (i.e., A or B or [C and D]) to be classified as belonging to a 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; you would 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 is equivalent to A or B or [C and D]).

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

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 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.

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

Creating a Traffic Class

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. For more information about the **match-all** and **match-any** keywords of the class-map comand, see the "match-all and match-any Keywords of the class-map Command" section.

Note The **match cos** command is shown in Step 4. 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 the "match-all and match-any Keywords of the class-map Command" section.

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 step 6.
6. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**class-map** [**match-all** | **match-any**] *class-map-name* **Example:** `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. • Enter the class name. **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* **Example:** `Router(config-cmap)# match cos 2`	Matches a packet on the basis of a Layer 2 class of service (CoS) number. • Enter the CoS number. **Note** The **match cos** command is an example of the **match** commands you can use. For information about the other **match** commands that are available, see the "match-all and match-any Keywords of the class-map Command" section.
Step 5	Enter additional match commands, if applicable; otherwise, continue with step 6.	--
Step 6	**end** **Example:** `Router(config-cmap)# end`	(Optional) Exits QoS class-map configuration mode and returns to privileged EXEC mode.

Creating a Traffic Policy

Note The **bandwidth** command is shown in Step 5. The **bandwidth** command is an example of the commands that you can use in a policy map to enable a QoS feature (in this case, Class-based Weighted Fair Queuing (CBWFQ). For information about other available commands, see the "Elements of a Traffic Policy" section.

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 Step 7.
7. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**policy-map** *policy-map-name* **Example:** `Router(config)# policy-map policy1`	Creates or specifies the name of the traffic policy and enters QoS policy-map configuration mode. • Enter the policy map name.
Step 4	**class** {*class-name*	**class-default**} **Example:** `Router(config-pmap)# class class1`	Specifies the name of a traffic class and enters QoS policy-map class configuration mode. **Note** This step associates the traffic class with the traffic policy.
Step 5	**bandwidth** {*bandwidth-kbps*	**percent** *percent*} **Example:** `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 kb/s or by a percentage of the overall available bandwidth.

	Command or Action	Purpose
		Note The **bandwidth** command enables CBWFQ. The **bandwidth** command is an example of the commands that you can use in a policy map to enable a QoS feature. For information about the other commands available, see the "Elements of a Traffic Policy" section.
Step 6	Enter the commands for any additional QoS feature that you want to enable, if applicable; otherwise, continue with Step 7.	--
Step 7	**end** **Example:** `Router(config-pmap-c)# end`	(Optional) Exits QoS policy-map class configuration mode and returns to privileged EXEC mode.

Attaching a Traffic Policy to an Interface Using the MQC

Note Cisco IOS XE Release 2.3.0 and later releases do not support the attachment of policies for ATM interfaces that have unspecified bit rate (UBR) configured as the default mode on their VC or virtual path (VP). An attempt to use this configuration results in an error message: CBWFQ: Not supported on ATM interfaces with UBR configuration. You can also specify UBR with a rate in the UBR configuration, if you do not want to use the default UBR value.

SUMMARY STEPS

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

DETAILED STEPS

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

	Command or Action	Purpose
Step 3	**interface** *type number* **Example:** `Router(config)# interface serial 0/0/1`	Configures an interface type and enters interface configuration mode. • Enter the interface type and interface number.
Step 4	**service-policy** {**input** | **output**} *policy-map-name* **Example:** `Router(config-if)# service-policy input policy1`	Attaches a policy map to an interface. • Enter either the **input** or **output** keyword and the policy map name.
Step 5	**end** **Example:** `Router(config-if)# end`	(Optional) Exits interface configuration mode and returns to privileged EXEC mode.

Verifying the Traffic Class and Traffic Policy Information

The show commands described in this section are optional and can be entered in any order.

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** *type number*
6. **exit**

DETAILED STEPS

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

	Command or Action	Purpose
	`show policy-map policy1 class class1`	
Step 4	**show policy-map** **Example:** `Router# show policy-map`	(Optional) Displays the configuration of all classes for all existing policy maps.
Step 5	**show policy-map interface** *type number* **Example:** `Router# show policy-map interface serial 0/0/1`	(Optional) Displays the statistics and the configurations of the input and output policies that are attached to an interface. • Enter the interface type and number.
Step 6	**exit** **Example:** `Router# exit`	(Optional) Exits privileged EXEC mode.

Configuration Examples for Applying QoS Features Using the MQC

Creating a Traffic Class

In the following example, we create traffic classes and define their match criteria. For the first traffic class (class1), we use access control list (ACL) 101 as match criteria; for the second traffic class (class2), ACL 102. We check the packets against the contents of these ACLs to determine if they belong to the class.

```
class-map class1
  match access-group 101
  exit
class-map class2
  match access-group 102
  end
```

Creating a Policy Map

In the following example, we define a traffic policy (policy1) containing the QoS features that we will apply to two classes: class1 and class2. The match criteria for these classes were previously defined in Creating a Traffic Class, on page 15).

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

```
policy-map policy1
  class class1
    bandwidth 3000
```

```
  queue-limit 30
  exit
class class2
  bandwidth 2000
  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 fastethernet 1/1/1
Router(config-if)# service-policy output policy1
Router(config-if)# exit
Router(config)# interface fastethernet 1/0/0
Router(config-if)# service-policy output policy1
Router(config-if)# end
```

Using the match not Command

Use the **match not** command to specify a QoS policy value that is not used as a match criterion. All other values of that QoS policy become successful match criteria. For instance, if you issue the **match not qos-group 4** command 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:

```
class-map noip
  match not protocol ip
  end
```

Configuring a Default Traffic Class

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

If you do not configure a default class, packets are still treated as members of that class. The default class has no QoS features enabled so packets belonging to this class have no QoS functionality. Such packets are placed into a first-in, first-out (FIFO) queue managed by tail drop, which 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 active, packets are dropped until the congestion is eliminated and the queue is no longer full.

The following example configures a policy map (policy1) for the default class (always called class-default) with these characteristics: 10 queues for traffic that does not meet the match criteria of other classes whose policy is defined by class policy1, and a maximum of 20 packets per queue before tail drop is enacted to handle additional queued packets.

In the following example, we configure a policy map (policy1) for the default class (always termed class-default) with these characteristics: 10 queues for traffic that does not meet the match criterion of other classes whose policy is defined by the traffic policy policy1.

```
policy-map policy1
  class class-default
    shape average 100m
```

How "fair-queue" Supports "pre-classify" Command

Prior to the Cisco IOS 16.4 release, when you configure fair-queue on the tunnel interface, the outer IP header of the tunnel was used for the hash algorithm of fair queue. Therefore, the packets of all flows on the tunnel were put into the same flow queue. This is the behavior seen even when the **qos pre-classify** command is configured on the tunnel interface

From the Cisco IOS 16.4 release onwards, **fair-queue** supports the **pre-classify** command. This command is added so that **qos pre-classify** can be used with the **fair-queue** command.

The following example configures **fair-queue pre-classify** command for policy-map under class configuration:

```
interface tunnel 0
    qos pre-classify
policy-map po1
    class c1
        shape average percentage 10
        fair-queue pre-classify
```

When **qos pre-classify** is enabled on the tunnel interface, and the **fair-queue pre-classify** is enabled for policy-map, then the policy-map is attached to either the tunnel interface or the physical interface. The inner IP header of the tunnel is used for the hash algorithm of the fair queue.

To disable this feature, use the **fair-queue** command without the **pre-classify** keyword.

The default behavior of fair queue remains unchanged.

How Commands "class-map match-any" and "class-map match-all" Differ

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

The following examples show a traffic class configured with the **class-map match-all** command:

```
class-map match-all cisco1
  match qos-group 4
  match access-group 101
```

If a packet arrives on a router with traffic class cisco1 configured on the interface, we assess whether it matches the IP protocol, QoS group 4, and access group 101. If all of these match criteria are met, the packet is classified as a member of the traffic class cisco1 (a logical AND operator; Protocol IP AND QoS group 4 AND access group 101).

```
class-map match-all vlan
  match vlan 1
```

```
  match vlan inner 1
```

The following example illustrates use of the **class-map match-any** command. Only one match criterion must be met for us to classify the packet as a member of the traffic class (i.e., a logical OR operator; protocol IP OR QoS group 4 OR access group 101):

```
class-map match-any cisco2
  match protocol ip
  match qos-group 4
  match access-group 101
```

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

Establishing 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 second and more common reason is to mix match-all and match-any characteristics in one traffic policy. This enables you to create a traffic class using one match criterion evaluation instruction (either match-any or match-all) and then use that traffic class as a match criterion in a traffic class that uses a different match criterion type.

Consider this likely scenario: Suppose A, B, C, and D were all separate match criterion, and you wanted traffic matching A, B, or C and D (i.e., A or B or [C and D]) to be classified as belonging to a 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; you would 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 is equivalent to A or B or [C and D]).

Example: Nested Traffic Class for Maintenance

In the following example, the traffic class class1 has the same characteristics as the traffic class class2, with the exception that the former 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 you to include all of the characteristics in the traffic class called class2 in the traffic class 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, use the match-any instruction to create a traffic class that uses a class configured with the match-all instruction as a match criterion (through the **match class-map** command).

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 QoS 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 permanent virtual connection (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 ?
384-100000000 Kilo Bits per second
```

```
level Multi-Level Priority Queue
percent % of total bandwidth
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
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
Frame Relay Fragmentation (FRF) PVCs	"FRF .20 Support" module
Selective Packet Discard	"IPv6 Selective Packet Discard" module
Scaling and performance information	"Broadband Scalability and Performance" module of the Cisco ASR 1000 Series Aggregation Services Routers Software Configuration Guide.

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

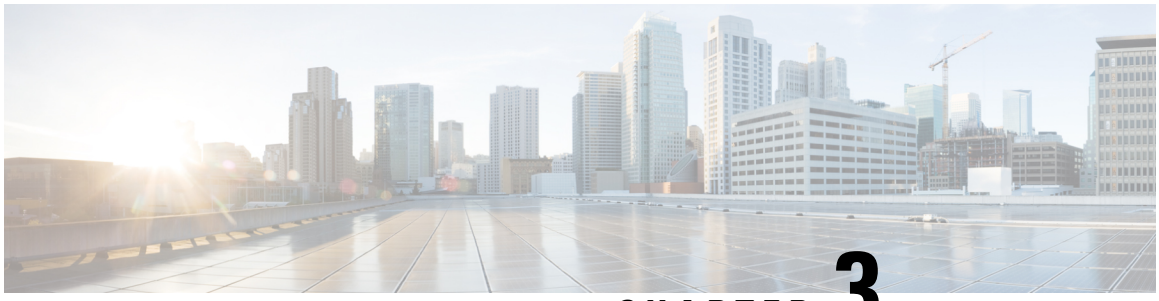
Feature Information for Applying QoS Features Using the MQC

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 7: Feature Information for Applying QoS Features Using the MQC

Feature Name	Releases	Feature Information
Class-Based Weighted Fair Queueing (CBWFQ)	Cisco IOS XE Release 2.1 Cisco IOS XE Release 3.5S	This feature was introduced on Cisco ASR 1000 Series Aggregation Services Routers. In Cisco IOS XE Release 3.5S, support was added for the Cisco ASR 903 Router.
Modular QoS CLI (MQC)	Cisco IOS XE Release 2.1 Cisco IOS XE Release 3.5S	This module describes how to apply and configure quality of service (QoS) features using the modular QoS CLI (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. This feature was introduced on Cisco ASR 1000 Series Aggregation Services Routers. This feature was enhanced to provide infrastructure support for additional features included with Cisco IOS XE Release 2.3. In Cisco IOS XE Release 3.5S, support was added for the Cisco ASR 903 router.
MQC Hierarchical Class Map	Cisco IOS XE Release 3.2	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 feature was introduced on Cisco ASR 1000 Series Aggregation Services Routers.
Priority Queueing	Cisco IOS XE Release 2.1	This feature was introduced on Cisco ASR 1000 Series Aggregation Services Routers.
Weighted Random Early Detection	Cisco IOS XE Release 2.1	This feature was introduced on Cisco ASR 1000 Series Aggregation Services Routers.



CHAPTER 3

3-Level User-Defined Queuing Policy Support

3-level user-defined queuing policy support feature allows 3 level policy with topmost layer user defined classes to support and enhance the flexibility of the traffic class in the hierarchy.

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.

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 3-Level User-Defined Queuing Policy Support

- User-defined class in top layer of a 3-level hierarchical queuing policy is not supported on port-channel main interface.

  User-defined class at the topmost layer is not supported on any logical target. Logical targets include service-group, tunnel, session, dealer interface, etc.

Information About 3-Level User-Defined Queuing Policy Support

Three-Parameter Scheduler in Hierarchical QoS

Classic IOS uses max value (shape) and min value (bandwidth) to define each scheduler node behavior when traffic congestion happens, or 2 parameter scheduler.

ASR 1000 utilize a different 3-parameter scheduler: max value (shape), min value (bandwidth) and excess value (bandwidth remaining) which is to adjust sharing of excess bandwidth. In a 2-parameter scheduler, the excess bandwidth are shared by the classes proportionally (same as the bandwidth ratio for each class); But in a 3-parameter scheduler, the excess bandwidth are shared equally by default after satisfying minimum bandwidth requirements, but it can be tuned when using 'bandwidth remaining' command. ISR 4000 platforms share the same design.

In Classic IOS, it is permitted to configure bandwidth at the leaf and intermediate nodes of a hierarchy. In IOS XE, bandwidth (bandwidth rate , or bandwidth percent) is only allowed at the leaf node of a hierarchy. In other words, bandwidth (bandwidth rate , or bandwidth percent) class cannot be attached with a child policy-map containing queuing features. This is a restriction in software and may be lifted in the future.

For current deployments where a Classic IOS QoS policy-map is being moved to a IOS XE platform, the best option is to convert the intermediate node bandwidth commands to bandwidth remaining commands. bandwidth remaining percent or bandwidth remaining ratio commands could be used to achieve very similar behavior.

Guidelines for Hierarchical Policies

In general, three levels of hierarchy are supported on ASR 1000. Hierarchical policy can be applied on most of the physical and logical targets that supports QoS.

If you mix queuing and non-queuing policies together in a hierarchy, the non-queuing policy-maps must be at the leaf level of the policy-map (for example, child policy beneath grandparent and parent queuing policies).

If the policy-map is applied to a virtual interface (such as a tunnel or session), there may be additional restrictions limiting the hierarchy to two levels of queuing, depending on the configuration.

- Queuing features: shape, bandwidth, bandwidth remaining, random-detect, fair-queue, queue limit, and priority.
- Non-queuing features: police, mark, and account.

Hierarchy Feature Combination	Ingress Policy Support	Egress Policy Support
One-level Non-queuing Policy	Yes	Yes
Two-level Non-queuing Policy (including color-aware police)	Yes	Yes
Three-level Non-queuing Policy (including hierarchical color-aware police)	Yes	Yes
One-level Queuing Policy	-	Yes

Hierarchy Feature Combination	Ingress Policy Support	Egress Policy Support
Two-level Queuing Policy	-	Yes
Three-level Queuing Policy	-	Yes
Two-level Mixed Policy, Queuing feature at parent level	-	Yes
Three-level Mixed Policy, Queuing feature at grandparent level, or at grandparent + parent level	-	Yes

User-defined Traffic Class in Top-level Policy of HQoS

Any traffic class configured explicitly by 'class-map' is called 'user-defined class'. Class-default classes need not be configured, and can be used in any policy to match all the traffic that does not belong to user-defined classes.

In a three-level queuing policy-map, only class-default class can be configured in the highest level before Release Polaris 16.3. From Polaris 16.3, user-defined class in top layer of a 3-level hierarchical policy is supported.

How to Configure 3-Level User-Defined Queuing Policy Support

Configuring 3-level Hierarchical QoS Policy

```
enable
configure terminal
class-map vlan10
 match vlan10
class-map vlan20
 match vlan 20
class-map ef
 match dscp ef
policy-map child
 class ef
  priority
  police 1000000
 class class-default
 police 3000000
policy-map parent
 class vlan10
  shape average 4000000
  service-policy child
 class vlan20
  shape average 8000000
  service-policy child
policy-map grand-parent
 class class-default
  shape average 10000000
  service-policy parent
end
```

Configuring User-Defined Traffic Class in Top Level Policy

```
ip access-list extended PEER
permit ip host 200.0.0.2 any

class-map match-all ef
 match dscp ef
class-map match-all vlan100
 match vlan 100
class-map match-all vlan101
 match vlan 101
class-map match-all PEER
 match access-group name PEER

policy-map child
 class ef
  bandwidth remaining percent 15
 class class-default
  fair-queue
  queue-limit 512 packets
  bandwidth remaining percent 85

policy-map parent
 class PEER
  shape average 8000000
  bandwidth remaining percent 10
  service-policy child
class class-default
 shape average 8000000

policy-map grandparent
 class vlan100
  shape average 8000000
  bandwidth remaining ratio 1000
  service-policy parent
 class vlan101
  shape average 8000000
  bandwidth remaining ratio 1000
  service-policy parent
 class class-default
  bandwidth remaining ratio 1
  shape average 10000000
end
```

Additional References for 3-Level User-Defined Queuing Policy Support

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Commands List, All Releases

Technical Assistance

Description	Link
The Cisco Support website provides extensive online resources, including documentation and tools for troubleshooting and resolving technical issues with Cisco products and technologies. To receive security and technical information about your products, you can subscribe to various services, such as the Product Alert Tool (accessed from Field Notices), the Cisco Technical Services Newsletter, and Really Simple Syndication (RSS) Feeds. Access to most tools on the Cisco Support website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

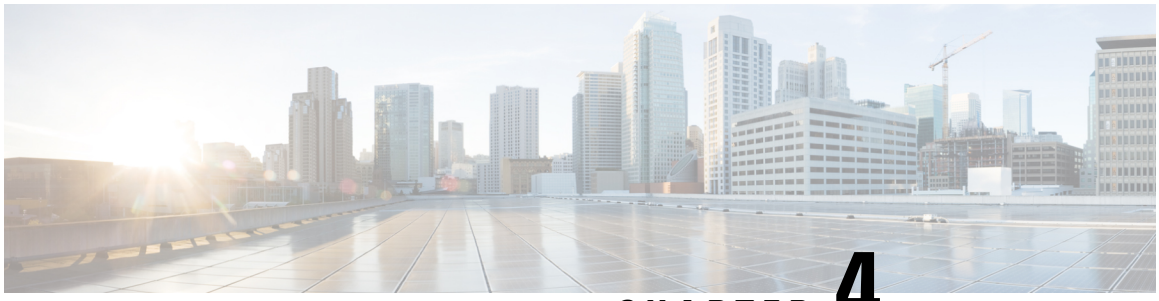
Feature Information for 3-Level User-Defined Queuing Policy Support

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 3-Level User-Defined Queuing Policy Support

Feature Name	Releases	Feature Information
3-Level User-Defined Queuing Policy Support	Cisco IOS XE Denali 16.3.1.	This feature is introduced on Cisco ASR 1000, ISR4000, CSR1000v platforms. User-defined class can be configured in top layer of a 3-level hierarchical policy.



CHAPTER 4

Configuring IP to ATM Class of Service

This module describes the tasks for configuring the IP to ATM Class of Service (CoS), a feature suite that maps QoS characteristics between IP and ATM.

Use Cisco Feature Navigator to find information about platform support and software image support. Cisco Feature Navigator enables you to determine which Cisco IOS and Catalyst OS software images support a specific software release, feature set, or platform. To access Cisco Feature Navigator, go to http://www.cisco.com/go/cfn . An account on Cisco.com is not required.

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.

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.

IP to ATM CoS on a Single ATM VC Configuration Task List

To configure IP to ATM CoS for a single ATM virtual circuit (VC), perform the tasks described in the following sections. The tasks in the first two sections are required; the tasks in the remaining sections are optional.

The IP to ATM CoS feature requires ATM permanent virtual circuit (PVC) management.

Defining the WRED Parameter Group

Command	Purpose
`Router(config)# random-detect-group group-name`	Defines the WRED or VIP-distributed WRED (DWRED) parameter group.

Configuring the WRED Parameter Group

SUMMARY STEPS

1. Device(config)# **random-detect-group** *group-name*
2. Device(config)# **exponential-weighting-constant** *exponent*
3. Device(config)# **precedence** *precedence min-threshold max-threshold mark-probability-denominator*

DETAILED STEPS

	Command or Action	Purpose
Step 1	Device(config)# **random-detect-group** *group-name*	Specifies the WRED or DWRED parameter group.
Step 2	Device(config)# **exponential-weighting-constant** *exponent*	Configures the exponential weight factor for the average queue size calculation for the specified WRED or DWRED parameter group. or
Step 3	Device(config)# **precedence** *precedence min-threshold max-threshold mark-probability-denominator*	Configures the specified WRED or DWRED parameter group for a particular IP precedence.

Displaying the WRED Parameters

Command	Purpose
`Router# show queueing random-detect [interface atm_subinterface [vc [[vpi/] vci]]]`	Displays the parameters of every VC with WRED or DWRED enabled on the specified ATM subinterface.

Displaying the Queueing Statistics

Command	Purpose
`Router# show queueing interface interface-number [vc [[vpi/] vci]]`	Displays the queueing statistics of a specific VC on an interface.

IP to ATM CoS on an ATM Bundle Configuration Task List

To configure IP to ATM CoS on an ATM bundle, perform the tasks in the following sections.

The IP to ATM CoS feature requires ATM PVC management.

Creating a VC Bundle

Command	Purpose
`Router(config-subif)# bundle bundle-name`	Creates the specified bundle and enters bundle configuration mode.

Applying Bundle-Level Parameters

Configuring Bundle-Level Parameters

Command	Purpose
`Device(config-atm-bundle)# protocol protocol {protocol-address | inarp} [[no] broadcast]`	Configures a static map or enables Inverse Address Resolution Protocol (Inverse ARP) or Inverse ARP broadcasts for the bundle. **Note** Bundle-level parameters can be applied either by assigning VC classes or by directly applying them to the bundle. Parameters applied through a VC class assigned to the bundle are superseded by those applied at the bundle level. Bundle-level parameters are superseded by parameters applied to an individual VC.
`Device(config-atm-bundle)# encapsulation aal-encap`	Configures the ATM adaptation layer (AAL) and encapsulation type for the bundle.
`Device(config-atm-bundle)# inarp minutes`	Configures the Inverse ARP time period for all VC bundle members.
`Device(config-atm-bundle)# broadcast`	Enables broadcast forwarding for all VC bundle members.
`Device(config-atm-bundle)# oam retry up-count down-count retry frequency`	Configures the VC bundle parameters related to operation, administration, and maintenance (OAM) management.
`Device(config-atm-bundle)# oam-bundle [manage] [frequency]`	Enables end-to-end F5 OAM loopback cell generation and OAM management for all VCs in the bundle.

Configuring VC Class Parameters to Apply to a Bundle

Command	Purpose
`Router(config-vc-class)# oam-bundle [manage] [frequency]`	Enables end-to-end F5 OAM loopback cell generation and OAM management for all VCs in the bundle. **Note** Use of a VC class allows you to configure a bundle applying multiple attributes to it at once because you apply the class itself to the bundle. Use of a class allows you to generalize a parameter across all VCs, after which (for some parameters) you can modify that parameter for individual VCs. (See the section "Applying Parameters to Individual VCs" for more information.)

Attaching a Class to a Bundle

Command	Purpose
`(config-atm-bundle)# class-bundle vc-class-name`	Configures a bundle with the bundle-level commands contained in the specified VC class. **Note** Parameters set through bundle-level commands contained in the VC class are applied to the bundle and all of its VC members. Bundle-level parameters applied through commands configured directly on the bundle supersede those applied through a VC class. Some bundle-level parameters applied through a VC class or directly to the bundle can be superseded by commands that you directly apply to individual VCs in bundle-vc configuration mode.

Committing a VC to a Bundle

Command	Purpose
`Device(config-atm-bundle)# pvc-bundle pvc-name [vpi/] [vci]`	Adds the specified VC to the bundle and enters bundle-vc configuration mode in order to configure the specified VC bundle member.

Applying Parameters to Individual VCs

Configuring a VC Bundle Member Directly

Command	Purpose
`Device(config-if-atm-member)# ubr output-pcr [input-pcr]`	Configures the VC for unspecified bit rate (UBR) QoS and specifies the output peak cell rate (PCR) for it.

Command	Purpose
`Device(config-if-atm-member)# ubr+ output-pcr output-mcr [input-pcr] [input-mcr]`	Configures the VC for UBR QoS and specifies the output PCR and output minimum guaranteed cell rate for it.
`Device(config-if-atm-member)# vbr-nrt output-pcr output-scr output-mbs [input-pcr] [input-scr] [input-mbs]`	Configures the VC for variable bit rate nonreal-time (VBR-nrt) QoS and specifies the output PCR, output sustainable cell rate, and output maximum burst cell size for it.
`Device(config-if-atm-member)# precedence [other | range]`	Configures the precedence levels for the VC.
`Device(config-if-atm-member)# bump {implicit | explicit precedence-level | traffic}`	Configures the bumping rules for the VC.
`Device(config-if-atm-member)# protect {group | vc}`	Configures the VC to belong to the protected group of the bundle or to be an individually protected VC bundle member.

Configuring VC Class Parameters to Apply to a VC Bundle Member

Command	Purpose
`Device(config-vc-class)# bump {implicit | explicit precedence-level | traffic}`	Specifies the bumping rules for the VC member to which the class is applied. These rules determine to which VC in the bundle traffic is directed when the carrier VC bundle member goes down. **Note** You can also add the following commands to a VC class to be used to configure a VC bundle member: **ubr**, **ubr+**, and **vbr-nrt**. When a VC is a member of a VC bundle, the following commands cannot be used in vc-class mode to configure the VC: **encapsulation**, **protocol**, **inarp**, and **broadcast**. These commands are useful only at the bundle level, not the bundle member level. Configuration for an individual VC overrides the collective configuration applied to all VC bundle members through application of a VC class to the bundle.
`Device(config-vc-class)# precedence precedence min-threshold max-threshold mark-probability-denominator`	Defines precedence levels for the VC member to which the class is applied.
`Device(config-vc-class)# protect {group | vc}`	Configures the VC as a member of the protected group of the bundle or as an individually protected VC.

Applying a VC Class to a Discrete VC Bundle Member

Command	Purpose
`Device(config-if-atm-member)# class-vc vc-class-name`	Assigns a VC class to a VC bundle member.

Configuring a VC Not to Accept Bumped Traffic

Command	Purpose
`Device(config-if-atm-member)# no bump traffic`	Configures the VC not to accept any bumped traffic that would otherwise be redirected to it.

Monitoring and Maintaining VC Bundles and Their VC Members

Command	Purpose
`Device# show atm bundle bundle-name`	Displays the bundle attributes assigned to each bundle VC member and the current working status of the VC members.
`Device# show atm bundle bundle-name statistics [detail]`	Displays statistics or detailed statistics on the specified bundle.
`Device# show atm map`	Displays a list of all configured ATM static maps to remote hosts on an ATM network and on ATM bundle maps.
`Device# debug atm bundle errors`	Displays information on bundle errors.
`Device# debug atm bundle events`	Displays a record of bundle events.

Per-VC WFQ and CBWFQ Configuration Task List

To configure IP to ATM CoS for per-VC WFQ and CBWFQ, perform the tasks described in the following sections.

The IP to ATM CoS feature requires ATM PVC management.

Configuring Class-Based Weighted Fair Queueing

Before configuring CBWFQ for a VC, you must perform the following tasks using standard CBWFQ commands:

- Create one or more classes to be used to classify traffic sent across the VC

- Define a policy-map containing the classes to be used as the service policy

Note You can configure class policies for as many classes as are defined on the router, up to the maximum of 64. However, the total amount of bandwidth allocated for all classes included in a policy-map to be attached to a VC must not exceed 75 percent of the available bandwidth of the VC. The remaining 25 percent of available bandwidth is used for encapsulation, such as the ATM cell overhead (also referred to as ATM cell tax), routing and best-effort traffic, and other functions that assume overhead. For more information on bandwidth allocation, see the "Congestion Management Overview" module.

Because CBWFQ gives you minimum bandwidth guarantee, you can only apply CBWFQ to VCs having these classes of service: available bit rate (ABR) and variable bit rate (VBR). You cannot apply per-VC WFQ and CBWFQ to UBR and unspecified bit rate plus (UBR+) VCs because both of these service classes are best-effort classes that do not guarantee minimum bandwidth. When CBWFQ is enabled for a VC, all classes configured as part of the service policy are installed in the fair queueing system.

In addition to configuring CBWFQ at the VC level, the IP to ATM CoS feature allows you to configure flow-based WFQ at the VC level. Because flow-based WFQ gives you best-effort class of service--that is, it does not guarantee minimum bandwidth--you can configure per-VC WFQ for all types of CoS VCs: ABR, VBR, UBR, and UBR+.

Per-VC WFQ uses the class-default class. Therefore, to configure per-VC WFQ, you must first create a policy-map and configure the class-default class. (You need not create the class-default class, which is predefined, but you must configure it.) For per-VC WFQ, the class-default class must be configured with the **fair-queue** policy-map class configuration command.

In addition to configuring the **fair-queue** policy-map class configuration command, you can configure the default class with either the **queue-limit** command or the **random-detect**command, but not both. Moreover, if you want the default class to use flow-based WFQ, you cannot configure the default class with the **bandwidth** policy-map class configuration command--to do so would disqualify the default class as flow-based WFQ, and therefore limit application of the service policy containing the class to ABR and VBR VCs.

Attaching a Service Policy and Enabling CBWFQ for a VC

Attaching a Policy-Map to a Standalone VC and Enabling CBWFQ

Command	Purpose
`Router(config-if-atm-vc)#` **`service-policy output`** *`policy-map`*	Enables CBWFQ and attaches the specified service policy-map to the VC being created or modified.

Attaching a Policy-Map to an Individual VC and Enabling CBWFQ

Command	Purpose
`Router(config-if-atm-member)#` **`service-policy output`** *`policy-map`*	Enables CBWFQ and attaches the specified service policy-map to the VC being created or modified.

Note The **service-policy output** and **random-detect-group** commands are mutually exclusive; you cannot apply a WRED group to a VC for which you have enabled CBWFQ through application of a service policy. Moreover, before you can configure one command, you must disable the other if it is configured.

Configuring a VC to Use Flow-Based WFQ

SUMMARY STEPS

1. Device(config)# **policy-map** *policy-map*
2. Device(config-pmap)# **class class-default** *default-class-name*
3. Device(config-pmap-c)# **fair-queue** *number-of-dynamic-queues*
4. Do one of the following:
   - Device(config-pmap-c)# **queue-limit** *number-of-packets*
   - Device(config-pmap-c)# **random-detect**

DETAILED STEPS

	Command or Action	Purpose
Step 1	Device(config)# **policy-map** *policy-map*	Specifies the name of the policy-map to be created or modified.
Step 2	Device(config-pmap)# **class class-default** *default-class-name*	Specifies the default class so that you can configure or modify its policy. **Note** You can include other classes in the same policy-map as the one that contains the flow-based WFQ class. Packets not otherwise matched are selected by the default class-default class match criteria.
Step 3	Device(config-pmap-c)# **fair-queue** *number-of-dynamic-queues*	Specifies the number of dynamic queues to be reserved for use by flow-based WFQ running on the default class. **Note** By default--that is, even if you do not configure the class-default class with the **fair-queue** policy-map class configuration command and you do not configure it with the **bandwidth** policy-map class configuration command--the default class is defined as flow-based WFQ.
Step 4	Do one of the following: • Device(config-pmap-c)# **queue-limit** *number-of-packets* • Device(config-pmap-c)# **random-detect**	Specifies the maximum number of packets that can be queued for the class. Enables WRED. The class policy will drop packets using WRED instead of tail drop.

Attaching a Policy-Map to a Standalone VC and Enabling WFQ

Command	Purpose
`Device(config-if-atm-vc)# service-policy output policy-map`	Enables WFQ for the VC by attaching the specified policy-map containing the class-default class to the VC being created or modified.

Attaching a Policy-Map to an Individual VC and Enabling WFQ

Command	Purpose
`Device(config-if-atm-member)# service-policy output policy-map`	Enables WFQ for the VC bundle member by attaching the specified policy-map containing the class-default class to the VC bundle member.

Monitoring per-VC WFQ and CBWFQ

Command	Purpose
`Device# show policy-map interface interface-number [vc [vpi/] vci]]`	Displays the contents of packets inside a queue for a particular interface or VC.

Enabling Logging of Error Messages to the Console

Command	Purpose
`Router(config)# logging console level`	Limits messages logged to the console based on severity.

IP to ATM CoS Configuration Examples

Example Single ATM VC with WRED Group and IP Precedence

The following example creates a PVC on an ATM interface and applies the WRED parameter group called sanjose to that PVC. Next, the IP Precedence values are configured for the WRED parameter group sanjose.

```
interface ATM1/1/0.46 multipoint
 ip address 200.126.186.2 255.255.255.0
 no ip mroute-cache
 shutdown
pvc 46
 encapsulation aal5nlpid
 random-detect attach sanjose
!
random-detect-group sanjose
```

```
 precedence 0  200   1000 10
 precedence 1  300   1000 10
 precedence 2  400   1000 10
 precedence 3  500   1000 10
 precedence 4  600   1000 10
 precedence 5  700   1000 10
 precedence 6  800   1000 10
 precedence 7  900   1000 10
```

Example VC Bundle Configuration Using a VC Class

This example configures VC bundle management on a router that uses Intermediate System-to-Intermediate System (IS-IS) as its IP routing protocol.

Bundle-Class Class

At the outset, this configuration defines a VC class called bundle-class that includes commands that set VC parameters. When the class bundle-class is applied at the bundle level, these parameters are applied to all VCs that belong to the bundle. Note that any commands applied directly to an individual VC of a bundle in bundle-vc mode take precedence over commands applied globally at the bundle level. Taking into account hierarchy precedence rules, VCs belonging to any bundle to which the class bundle-class is applied will be characterized by these parameters: aal5snap encapsulation, broadcast on, use of Inverse Address Resolution Protocol (ARP) to resolve IP addresses, and operation, administration, and maintenance (OAM) enabled.

```
router isis
 net 49.0000.0000.0000.1111.00
vc-class atm bundle-class
 encapsulation aal5snap
 broadcast
 protocol ip inarp
 oam-bundle manage 3
 oam retry 4 3 10
```

The following sections of the configuration define VC classes that contain commands specifying parameters that can be applied to individual VCs in a bundle by assigning the class to that VC.

Control-Class Class

When the class called control-class is applied to a VC, the VC carries traffic whose IP Precedence level is 7. When the VC to which this class is assigned goes down, it takes the bundle down with it because this class makes the VC a protected one. The QoS type of a VC using this class is vbr-nrt.

```
vc-class atm control-class
 precedence 7
 protect vc
 vbr-nrt 10000 5000 32
```

Premium-Class Class

When the class called premium-class is applied to a VC, the VC carries traffic whose IP Precedence levels are 6 and 5. The VC does not allow other traffic to be bumped onto it. When the VC to which this class is applied goes down, its bumped traffic will be redirected to a VC whose IP Precedence level is 7. This class makes a VC a member of the protected group of the bundle. When all members of a protected group go down, the bundle goes down. The QoS type of a VC using this class is vbr-nrt.

```
vc-class atm premium-class
 precedence 6-5
 no bump traffic
 protect group
 bump explicitly 7
 vbr-nrt 20000 10000 32
```

Priority-Class Class

When the class called priority-class is applied to a VC, the VC is configured to carry traffic with IP Precedence in the 4-2 range. The VC uses the implicit bumping rule, it allows traffic to be bumped, and it belongs to the protected group of the bundle. The QoS type of a VC using this class is ubr+.

```
vc-class atm priority-class
 precedence 4-2
 protect group
 ubr+ 10000 3000
```

Basic-Class Class

When the class called basic-class is applied to a VC, the VC is configured through the **precedence other** command to carry traffic with IP Precedence levels not specified in the profile. The VC using this class belongs to the protected group of the bundle. The QoS type of a VC using this class is ubr.

```
vc-class atm basic-class
 precedence other
 protect group
 ubr 10000
```

The following sets of commands configure three bundles that the router subinterface uses to connect to three of its neighbors. These bundles are called new-york, san-francisco, and los-angeles. Bundle new-york has four VC members, bundle san-francisco has four VC members, and bundle los-angeles has three VC members.

new-york Bundle

The first part of this example specifies the IP address of the subinterface, the router protocol--the router uses IS-IS as an IP routing protocol--and it creates the first bundle called new-york and enters bundle configuration mode:

```
interface atm 1/0.1 multipoint
 ip address 10.0.0.1 255.255.255.0
 ip router isis
 bundle new-york
```

From within bundle configuration mode, the next portion of the configuration uses two protocol commands to enable IP and Open Systems Interconnect (OSI) traffic flows in the bundle. The OSI routing packets will use the highest precedence VC in the bundle. The OSI data packets, if any, will use the lowest precedence VC in the bundle. If configured, other protocols, such as IPX or AppleTalk, will always use the lowest precedence VC in the bundle.

As the indentation levels of the preceding and following commands suggest, subordinate to bundle new-york is a command that configures its protocol and a command that applies the class called bundle-class to it.

```
  protocol ip 1.1.1.2 broadcast
```

```
 protocol clns 49.0000.0000.2222.00 broadcast
 class-bundle bundle-class
```

The class called bundle-class, which is applied to the bundle new-york, includes a **protocol ip inarp** command. According to inheritance rules, **protocol ip**, configured at the bundle level, takes precedence over **protocol ip inarp** specified in the class bundle-class.

The next set of commands beginning with **pvc-bundle ny-control 207**, which are further subordinate, add four VCs (called ny-control, ny-premium, ny-priority, and ny-basic) to the bundle new-york. A particular class--that is, one of the classes predefined in this configuration example--is applied to each VC to configure it with parameters specified by commands included in the class.

As is the case for this configuration, to configure individual VCs belonging to a bundle, the router must be in bundle mode for the mother bundle. For each VC belonging to the bundle, the subordinate mode is pvc-mode for the specific VC.

The following commands configure the individual VCs for the bundle new-york:

```
pvc-bundle ny-control 207
 class-vc control-class
pvc-bundle ny-premium 206
 class-vc premium-class
pvc-bundle ny-priority 204
 class-vc priority-class
pvc-bundle ny-basic 201
 class-vc basic-class
```

san-francisco Bundle

The following set of commands create and configure a bundle called san-francisco. At the bundle configuration level, the configuration commands included in the class bundle-class are ascribed to the bundle san-francisco and to the individual VCs that belong to the bundle. Then, the **pvc-bundle** command is executed for each individual VC to add it to the bundle. After a VC is added and bundle-vc configuration mode is entered, a particular, preconfigured class is assigned to the VC. The configuration commands comprising that class are used to configure the VC. Rules of hierarchy apply at this point. Command parameters contained in the applied class are superseded by the same parameters applied at the bundle configuration level, which are superseded by the same parameters applied directly to a VC.

```
bundle san-francisco
 protocol clns 49.0000.0000.0000.333.00 broadcast
 inarp 1
 class-bundle bundle-class
 pvc-bundle sf-control 307
  class-vc control-class
 pvc-bundle sf-premium 306
  class-vc premium-class
 pvc-bundle sf-priority 304
  class-vc priority-class
 pvc-bundle sf-basic 301
  class-vc basic-class
```

los-angeles Bundle

The following set of commands create and configure a bundle called los-angeles. At the bundle configuration level, the configuration commands included in the class bundle-class are ascribed to the bundle los-angeles and to the individual VCs that belong to the bundle. Then, the **pvc-bundle** command is executed for each individual VC to add it to the bundle. After a VC is added and bundle-vc configuration mode is entered,

precedence is set for the VC and the VC is either configured as a member of a protected group (protect group) or as an individually protected VC. A particular class is then assigned to each VC to further characterize it. Rules of hierarchy apply. Parameters of commands applied directly and discretely to a VC take precedence over the same parameters applied within a class to the VC at the bundle-vc configuration level, which take precedence over the same parameters applied to the entire bundle at the bundle configuration level.

```
bundle los-angeles
 protocol ip 1.1.1.4 broadcast
 protocol clns 49.0000.0000.4444.00 broadcast
 inarp 1
 class-bundle bundle-class
 pvc-bundle la-high 407
  precedence 7-5
  protect vc
  class-vc premium-class
 pvc-bundle la-mid 404
  precedence 4-2
  protect group
  class-vc priority-class
 pvc-bundle la-low 401
  precedence other
  protect group
  class-vc basic-class
```

Example Per-VC WFQ and CBWFQ on a Standalone VC

The following example creates two class maps and defines their match criteria. For the first map class, called class1, the numbered access control list (ACL) 101 is used as the match criterion. For the second map class called class2, the numbered ACL 102 is used as the match criterion.

Next, the example includes these classes in a policy-map called policy1. For class1, the policy includes a minimum bandwidth allocation request of 500 kbps and maximum packet count limit of 30 for the queue reserved for the class. For class2, the policy specifies only the minimum bandwidth allocation request of 1000 kbps, so the default queue limit of 64 packets is assumed. Note that the sum of the bandwidth requests for the two classes comprising policy1 is 75 percent of the total amount of bandwidth (2000 kbps) for the PVC called cisco to which the policy-map is attached.

The example attaches the policy-map called policy1 to a PVC. Once the policy-map policy1 is attached to the PVC, its classes constitute the CBWFQ service policy for that PVC. Packets sent on this PVC will be checked for matching criteria against ACLs 101 and 102 and classified accordingly.

Because the **class-default** command is not explicitly configured for this policy-map, all traffic that does not meet the match criteria of the two classes comprising the service policy is handled by the predefined class-default class, which provides best-effort flow-based WFQ.

```
class-map class1
 match access-group 101
class-map class2
 match access-group 102
policy-map policy1
 class class1
  bandwidth 500
  queue-limit 30
 class class2
  bandwidth 1000
interface ATM1/1/0.46 multipoint
 ip address 200.126.186.2 255.255.255.0
 pvc 46
```

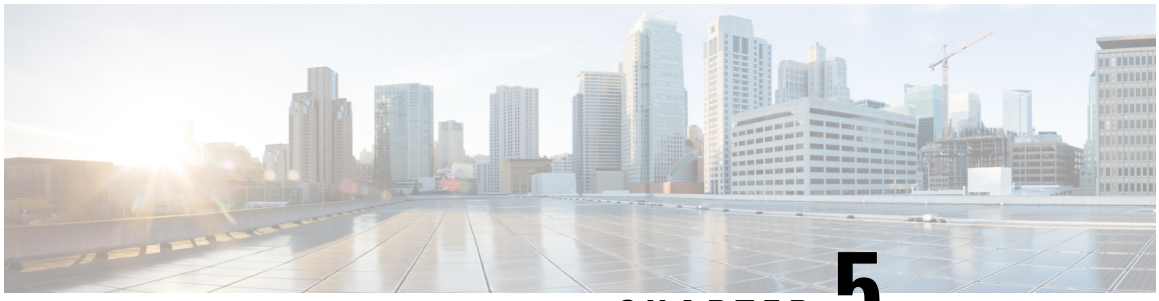
```
 vbr-nrt 2000 2000
 encap aal5snap
 service policy output policy1
```

Example Per-VC WFQ and CBWFQ on Bundle-Member VCs

The following example shows a PVC bundle called san-francisco with members for which per-VC WFQ and CBWFQ are enabled and service policies configured. The example assumes that the classes included in the following policy-maps have been defined and that the policy-maps have been created: policy1, policy2, and policy4. For each PVC, the IP to ATM CoS **pvc-bundle** command is used to specify the PVC to which the specified policy-map is to be attached.

Note that PVC 0/34 and 0/31 have the same policy-map attached to them, policy2. Although you can assign the same policy-map to multiple VCs, each VC can have only one policy-map attached at an output PVC.

```
bundle san-francisco
 protocol ip 1.0.2.20 broadcast
 encapsulation aal5snap
 pvc-bundle 0/35
  service policy output policy1
  vbr-nrt 5000 3000 500
  precedence 4-7
 pvc-bundle 0/34
  service policy output policy2
  vbr-nrt 5000 3000 500
  precedence 2-3
 pvc-bundle 0/33
  vbr-nrt 4000 3000 500
  precedence 2-3
  service policy output policy4
 pvc-bundle 0/31
  service policy output policy2
```



CHAPTER 5

Complex Hierarchical Scheduling: Fragmented Policies (i.e, Policies Aggregation)

The QoS: Policies Aggregation feature supports Modular QoS CLI (MQC) configuration of default traffic classes in policy maps on different subinterfaces to be queued as a single, user-defined traffic class at the main-interface policy map. It is most useful in quality of service (QoS) configurations where you have several subinterface policy maps on the same physical interface and you want identical treatment of the default traffic classes on those subinterfaces.

Beginning in Cisco IOS XE Release 2.6, the QoS: Policies Aggregation feature is enhanced to support queueing aggregation at the primary interface for other traffic classes, including Differentiated Services Code Point (DSCP) traffic classes such as the expedited forwarding (EF), Assured Forwarding 1 (AF1), and AF4 traffic classes. With this enhancement, any traffic classes from VLAN subinterfaces can share a common queue for that traffic class at the main-interface policy map. Other enhancements include the ability to configure and show drop statistics that occur at the aggregate level for these classes.

Prerequisites for QoS: Policies Aggregation

- This feature is configured using the MQC.
- All traffic over the main interface should come through one or more subinterfaces.

Restrictions for QoS: Policies Aggregation

- Applies only when multiple subinterfaces with policy maps are attached to the same physical interface. This feature cannot be used to collectively classify default traffic classes or other traffic classes of policy maps on different physical interfaces.

- Certain traffic class configuration prior to Cisco IOS XE Release 2.6 at the subinterface policy map and main-interface policy map will have different behavior and queueing results. See the "Understanding the QoS Policies Aggregation MQC" section on page 3 and the "Differences Between the Original Feature and the MQC Support for Multiple Queue Aggregation" section on page 4.
- The **service-fragment** keyword is only supported on the Gigabit Ethernet interfaces and not on Fast Ethernet interfaces.

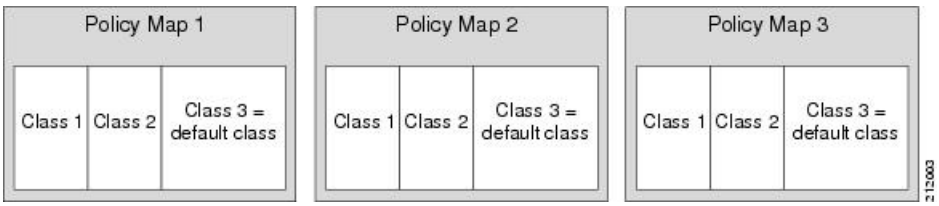
About QoS: Policies Aggregation

Fragments in Class Definition Statements

QoS: Policies Aggregation introduces the idea of fragments in class definition statements. A default traffic class definition statement can be marked as a fragment within a policy map. Other policy maps on the same interface can also define their default traffic class statements as fragments, if desired. A separate policy map can then be created with a service-fragment class definition statement that will be used to apply QoS to all of the fragments as a single group.

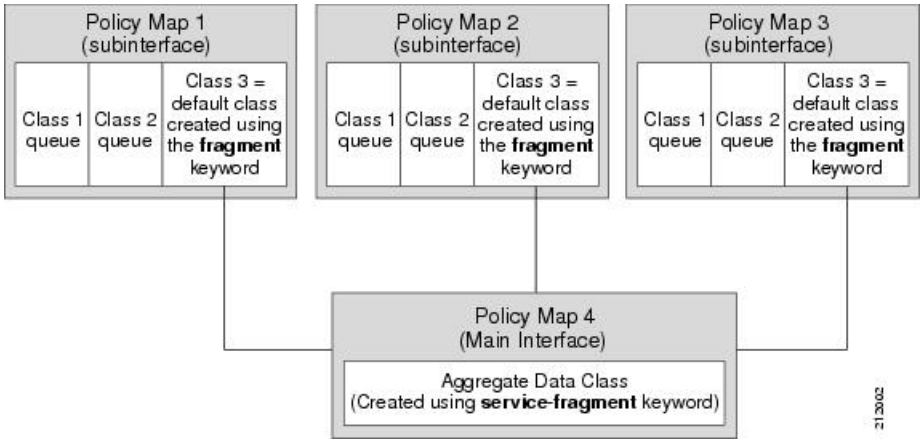
The figure below provides an example of one physical interface with three attached policy maps that is not using fragments. Note that each policy map has a default traffic class that can only classify traffic for the default traffic within its own policy map.

Figure 1: Three Policy Maps Configured Without Fragments



The figure below shows the same configuration configured with fragments and adds a fourth policy map with a class definition statement that classifies the fragments collectively. The default traffic classes are now classified as one service-fragment group rather than three separate default traffic classes within the individual policy maps.

Figure 2: Three Policy Maps Configured Using Fragments



Fragments for Gigabit Etherchannel Bundles

When fragments are configured for Gigabit Etherchannel bundles, the policy-maps that have a default traffic class configured using the **fragment** keyword are attached to the member subinterface links, and the policy-maps that have a traffic class configured with the **service-fragment** keyword to collectively classify the fragments is attached to the physical interface.

All port-channel subinterfaces configured with fragments that are currently active on a given port-channel member link will use the aggregate service fragment class on that member link. If a member link goes down, the port-channel subinterfaces that must switch to the secondary member link will then use the aggregate service fragment on the new interface.

Fragment Traffic Class in a Policy Map

Only the default class statement in a policy map can be configured as a fragment.

Fragments work only when multiple policy maps are attached to the same physical interface. This process cannot be used to classify default traffic classes as fragments on policy maps on different physical interfaces.

Only queuing features are allowed in classes where the **fragment** keyword is entered, and at least one queuing feature must be entered in classes where the **fragment** keyword is used.

A policy map with a class using the **fragment** keyword can only be applied to traffic leaving the interface (policy maps attached to interfaces using the **service-policy output** command).

The **fragment** keyword cannot be entered in a child policy map.

Understanding Service Fragment Traffic Classes

A service fragment can be used to collectively classify fragments only from the same physical interface. Fragments from different interfaces cannot be classified using the same service fragment.

Only queueing features are allowed in classes where the **service-fragment** keyword is entered, and at least one queueing feature must be entered in classes when the **service-fragment** keyword is used.

A policy map with a class using the **service-fragment** keyword can be applied only to traffic leaving the interface (policy maps attached to interfaces using the **service-policy output** command).

A class configured using the **service-fragment** keyword cannot be removed when it is being used to collectively apply QoS to fragments that are still configured on the interface. If you wish to remove a class configured using the **service-fragment** keyword, remove the fragment traffic classes before removing the service fragment.

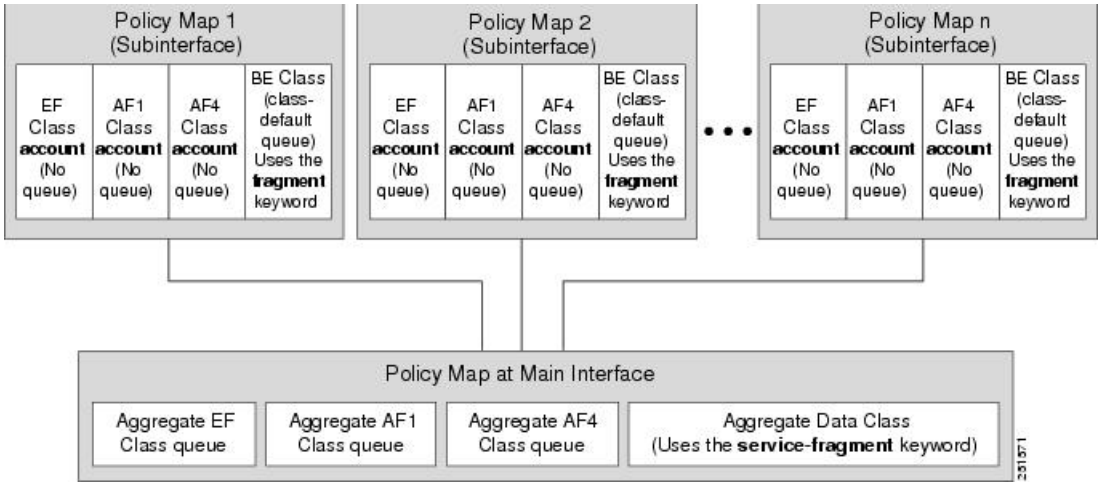
The **service-fragment** keyword cannot be entered in a child policy map.

QoS: Policies Aggregation MQC

The QoS: Policies Aggregation MQC Support for Multiple Queue Aggregation at Main Interface feature extends the previous support of aggregation of class-default traffic using the **fragment** and **service-fragment** configurations, to other user-defined traffic classes in a subinterface policy-map, such as DSCP-based traffic classes, that are aggregated at the main-interface policy-map as shown in the figure below.

When no queueing is configured on a traffic class in the subinterface policy-map, the **account** command can be used to track queueing drops that occur at the aggregate level for these classes, and can be displayed using the **show policy-map interface** command.

Figure 3: Policy-Map Overview for the MQC Support for Multiple Queue Aggregation at Main Interface Feature



Differences Between the Original Feature and the MQC Support for Multiple Queue Aggregation

Although some of the configuration between the original QoS policies aggregation feature and enhancements in the MQC Support for Multiple Queue Aggregation at Main Interface feature appears similar, there are some important differences in the queueing behavior and the internal data handling.

For example, both configurations share and require the use of the **fragment** keyword for the **class class-default** command in the subscriber policy map, as well as configuration of the **service-fragment** keyword for a user-defined class in the main-interface policy map to achieve common policy treatment for aggregate traffic. However, the use of this configuration results in different behavior between the original and enhanced QoS policies aggregation implementation:

- In the original implementation (prior to Cisco IOS XE Release 2.6) using the fragment and service-fragment architecture, all default class traffic and any traffic for classes without defined queueing features at the subinterface goes to the class-default queue and is aggregated into a common user-defined queue and policy defined at the main policy map. Subinterface traffic aggregation (for example, from

multiple subscribers on the same physical interface) ultimately occurs only for a single class, which is the default class.

Here are the feature characteristics:

- All subinterface traffic classes have queues. However, when a traffic class in the subinterface policy-map is not configured with any queueing feature (commands such as **priority**, **shape**, **bandwidth**, **queue-limit**, **fair-queue**, **random-detect**, and so on, are not configured), the traffic is assigned to the class-default queue.
- Default class traffic from multiple subinterfaces can be aggregated into a common policy map at the main interface when you use the **fragment** keyword at the subinterface **class class-default** configuration, and **service-fragment** configuration at the main-interface class.
- No classification occurs or is supported at the main-interface policy map for any subinterface traffic classes that do not use the **fragment** and **service-fragment** configuration.
- Queueing occurs at the subinterface for other traffic classes defined with queueing features in the subinterface policy map.

- In the enhanced implementation (beginning with Cisco IOS XE Release 2.6) of the MQC Support for Multiple Queue Aggregation at Main Interface feature also using the fragment and service-fragment architecture, all default class traffic also goes to the class-default queue and is aggregated into a common user-defined queue and policy defined at the main policy map. However, other classes, such as DSCP-based subscriber traffic classes, are also supported for an aggregate policy. These traffic classes do not support any queues or queueing features other than **account** at the subscriber policy map. The use of the fragment and service-fragment architecture enables these other subscriber traffic classes (from multiple subscribers on the same physical interface) to achieve common policy treatment for aggregate traffic that is defined for those same classes at the main policy map.

  Here are the feature characteristics:

  - Subinterface traffic classes without configured queueing features do not have queues at the subscriber level.
  - Default class traffic from multiple subinterfaces can be aggregated into a common policy map at the main interface when you use the **fragment** keyword at the subinterface **class class-default** configuration, and **service-fragment** configuration at the main-interface class. This configuration additionally enables support for other subinterface traffic classes (such as DSCP-based classes) to be aggregated into a common policy-map at the main interface.
  - Other class traffic from multiple subinterfaces can be aggregated into a common policy map at the main interface, according to the following configuration requirements:
  - You enable this behavior by using the **fragment** keyword at the subinterface **class class-default** configuration, and **service-fragment** configuration at the main-interface class (this also enables aggregation of the default class).
  - You do not configure any queueing features at the subinterface policy-map for the other traffic classes.
  - Queueing occurs at the main-interface policy map for other subinterface traffic classes as an aggregate.
  - Optional tracking of statistics is supported using the **account** command for other traffic classes in the subinterface policy map.

Changes in Queue Limit and WRED Thresholds

In Cisco IOS XE Release 2.6 the Cisco ASR 1000 Series Routers support the addition of bytes as a unit of configuration for both queue limits and WRED thresholds. Therefore, as of this release, packet-based and byte-based limits are configurable, with some restrictions.

Configuration Examples for QoS: Policies Aggregation

Examples 1: Configuring QoS: Policies Aggregation for an Interface

Configuring a Fragment Traffic Class in a Policy-Map

Before you begin

This procedure shows only how to configure the default traffic class as a fragment within a policy-map. It does not include steps on configuring other classes within the policy-map, or other policy-maps on the device.

Example

Note This example shows a sample configuration that is supported in releases prior to Cisco IOS XE Release 2.6.

In the following example, a fragment named BestEffort is created in policy-map subscriber1 and policy-map subscriber 2. In this example, queuing features for other traffic classes are supported at the subinterface policy-map.

```
policy-map subscriber1
 class voice
  set cos 5
  priority level 1
 class video
  set cos 4
  priority level 2
 class class-default fragment BestEffort
  shape average 200000000
  bandwidth remaining ratio 10
policy-map subscriber 2
 class voice
  set cos 5
  priority level 1
 class video
  set cos 4
  priority level 2
 class class-default fragment BestEffort
  shape average 200000000
  bandwidth remaining ratio 10
```

Note This example shows a sample configuration that is supported in Cisco IOS XE Release 2.6 and later releases.

The following example also shows how to configure a fragment named BestEffort for the default class in a policy-map on a subinterface using the QoS Policies Aggregation MQC Support for Multiple Queue Aggregation at Main Interface implementation. In this example, notice that queuing features are not supported for the other classes in the policy-map:

```
policy-map subscriber1
 class voice
  set cos 5
  account
 class video
  set cos 4
  account
 class AF1
  account
 class class-default fragment BestEffort
  shape average 200000000
  bandwidth remaining ratio 10
```

After configuring default class statements as fragments in multiple subinterface policy-maps, a separate policy-map with a class statement using the service-fragment keyword must be configured to apply QoS to the class stratements configured as fragments.

What to Do Next

After configuring default class statements as fragments in multiple subinterface policy maps, a separate policy map with a class statement using the **service-fragment** keyword must be configured to apply QoS to the class statements configured as fragments.

This task is documented in the "Configuring a Service Fragment Traffic Class" section on page 8.

Configuring a Service Fragment Traffic Class

Before you begin

This task describes how to configure a service fragment traffic class statement within a policy-map. A service fragment traffic class is used to apply QoS to a collection of default class statements that have been configured previously in other policy-maps as fragments.

This procedure assumes that fragment default traffic classes were already created. The procedure for creating fragment default traffic classes is documented in the “Configuring a Fragment Traffic Class in a Policy-Map” section.

Like any policy-map, the configuration does not manage network traffic until it has been attached to an interface. This procedure does not cover the process of attaching a policy-map to an interface.

Note A service fragment can be used to collectively classify fragments only from the same physical interface. Fragments from different interfaces cannot be classified using the same service fragment.

Only queueing features are allowed in classes where the **service-fragment** keyword is entered, and at least one queueing feature must be entered in classes when the **service-fragment** keyword is used.

A policy-map with a class using the **service-fragment** keyword can be applied only to traffic leaving the interface (policy-maps attached to interfaces using the **service-policy output** command).

A class configured using the **service-fragment** keyword cannot be removed when it is being used to collectively apply QoS to fragments that are still configured on the interface. If you wish to remove a class configured using the **service-fragment** keyword, remove the fragment traffic classes before removing the service fragment.

The **service-fragment** keyword cannot be entered in a child policy-map.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **policy-map** *policy-map-name*
4. **class** *class-name* **service-fragment** *fragment-class-name*
5. **shape average percent** *percent*
6. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**policy-map** *policy-map-name* **Example:** `Device(config)# policy-map BestEffortFragments`	Specifies the name of the traffic policy to configure and enters policy-map configuration mode.
Step 4	**class** *class-name* **service-fragment** *fragment-class-name* **Example:** `Device(config-pmap)# class data service-fragment BestEffort`	Specifies a class of traffic that is the composite of all fragments matching the *fragment-class-name*. The *fragment-class-name* when defining the fragments in other policy-maps must match the *fragment-class-name* in this command line to properly configure the service fragment class.

	Command or Action	Purpose
Step 5	**shape average percent** *percent* **Example:** `Device(config-pmap-c)# shape average percent 50`	Enters a QoS configuration command. Only queueing features are supported in default traffic classes configured as fragments. The queueing features that are supported are **bandwidth**, **shape**, and **random-detect exponential-weighting-constant**. Multiple QoS queueing commands can be entered.
Step 6	**end** **Example:** `Device(config-pmap-c)# end`	Exits policy-map class configuration mode and returns to privileged EXEC mode.

Examples

Note This example shows a sample configuration that is supported in releases prior to Cisco IOS XE Release 2.6.

In the following example, a policy-map is created to apply QoS to all fragments named BestEffort.

```
policy-map main-interface
 class data service-fragment BestEffort
  shape average 400000000
```

In the following example, two fragments are created and then classified collectively using a service fragment.

```
policy-map subscriber1
 class voice
  set cos 5
  priority level 1
 class video
  set cos 4
  priority level 2
 class class-default fragment BestEffort
  shape average 200000000
  bandwidth remaining ratio 10
policy-map subscriber 2
 class voice
  set cos 5
  priority level 1
 class video
  set cos 4
  priority level 2
 class class-default fragment BestEffort
  shape average 200000000
  bandwidth remaining ratio 10
```

Note This example shows a sample configuration that is supported in Cisco IOS XE Release 2.6 and later releases.

The following example shows the creation of two fragments called BestEffort in the subinterface policy-maps, followed by a sample configuration for the **service-fragment** called BestEffort to aggregate the queues at the main interface policy-map:

```
policy-map subscriber1
 class voice
  set cos 5
  account
 class video
  set cos 4
  account
 class AF1
  account
 class class-default fragment BestEffort
  shape average 200000000
  bandwidth remaining ratio 10
policy-map subscriber2
 class voice
  set cos 5
  account
 class video
  set cos 4
  account
 class AF1
  account
 class class-default fragment BestEffort
  shape average 200000000
  bandwidth remaining ratio 10
policy-map main-interface
 class voice
  priority level 1
 class video
  priority level 2
 class AF1
  bandwidth remaining ratio 90
 class data service-fragment BestEffort
  shape average 400000000
  bandwidth remaining ratio 1
```

Troubleshooting Tips

Ensure that all class statements that are supposed to be part of the same service fragment share the same *fragment-class-name*.

What to Do Next

The policy map (traffic policy) must be attached to an interface. This task is documented in the "Attaching a Traffic Policy to an Interface Using the MQC" section in chapter "Applying QoS Features Using the MQC."

Configuring QoS: Policies Aggregation on Gigabit Etherchannels

To properly configure QoS: Policies Aggregation on a Gigabit Etherchannel bundle, the following actions must be completed:

- Service-fragment traffic classes must be configured and attached to the main physical interfaces.
- Fragment traffic classes must be configured and attached to the member link subinterfaces.

Configuring Service Fragments on a Physical Interface Supporting a Gigabit Etherchannel Bundle

Before you begin

This procedure assumes that a service fragment traffic class has already been created. A service fragment traffic class cannot be configured without configuring a fragment class. The procedure for creating a fragment class is documented in the "Configuring a Fragment Traffic Class in a Policy-Map" section. The procedure for creating a service fragment traffic classes is documented in the "Configuring a Service Fragment Traffic Class" section.

These instructions do not provide any details about the options that can be configured for Gigabit Etherchannel member link subinterfaces. These instructions document only the procedure for attaching a policy-map that already has a fragment traffic class to a member link subinterface.

Note For proper behavior, when a port-channel member link goes down, all member links should have the same policy-map applied.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface GigabitEthernet** *card/bay/port*
4. **service-policy output** *service-fragment-class-name*
5. **end**

DETAILED STEPS

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

	Command or Action	Purpose
Step 3	**interface GigabitEthernet** *card/bay/port* **Example:** `Device(config)# interface GigabitEthernet 0/1/0`	Specifies the member link physical interface that receives the service-policy configuration.
Step 4	**service-policy output** *service-fragment-class-name* **Example:** `Device(config-if)# service-policy output aggregate-member-link`	Attaches a service policy that contains a service fragment default traffic class to the physical Gigabit Ethernet interface.
Step 5	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

Examples

In the following example, the policy-map aggregate-member-link is attached to the physical interface.

```
interface GigabitEthernet1/1/1
 service-policy output aggregate-member-link
!
interface GigabitEthernet1/1/2
 service-policy output aggregate-member-link
```

What to do next

Ensure that the fragment class name is consistent across service-fragment and fragment class definitions. Continue to the “Configuring Fragments on Gigabit Etherchannel Member Link Subinterfaces” section.

Troubleshooting Tips

Ensure that the *fragment-class-name* is consistent across service-fragment and fragment-class definitions.

What to Do Next

Attach the fragment service policy on the Gigabit Etherchannel member link subinterfaces. This task is documented in the "Configuring Fragments on Gigabit Etherchannel Member Link Subinterfaces" section on page 14.

Configuring Fragments on Gigabit Etherchannel Member Link Subinterfaces

Before you begin

This task assumes that a service-fragment traffic class has already been created. A service-fragment traffic class cannot be configured without configuring a fragment class. The procedure for creating a fragment class is documented in the "Configuring a Fragment Traffic Class in a Policy Map" section on page 6. The procedure

for creating a service-fragment traffic classes is documented in the "Configuring a Service Fragment Traffic Class" section on page 8.

These instructions do not provide any details about the options that can be configured for Gigabit Etherchannel member link subinterfaces. These instructions only document the procedure for attaching a policy map that already has a fragment traffic class to a member link subinterface.

Note Fragments cannot be used for traffic on two or more physical interfaces. The GEC must all be on the same physical interface for this configuration to work properly.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface port-channel** *port-channel-interface-number.port-channel-subinterface-number*
4. **service-policy output** *fragment-class-name*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface port-channel** *port-channel-interface-number.port-channel-subinterface-number* **Example:** `Router(config)# interface port-channel 1.100`	Enters subinterface configuration mode to configure a Etherchannel member link subinterface.
Step 4	**service-policy output** *fragment-class-name* **Example:** `Router(config-subif)# service-policy output subscriber`	Attaches a service policy that contains a fragment default traffic class to the Etherchannel member link subinterface.

Example

Note This example shows a sample configuration that is supported for the original QoS: Policies Aggregation feature in releases prior to Cisco IOS XE Release 2.6. By following the newer policy-map configuration guidelines for the updates in Cisco IOS XE Release 2.6, it can be adapted to the QoS: Policies Aggregation MQC Support for Multiple Queue Aggregation at Main Interface feature.

In the following example, the service policy named subscriber has a fragment default traffic class and is attached to the member link subinterface of a Gigabit Etherchannel bundle.

Note This example only shows how to attach a fragment default traffic class to the member link subinterface of a Gigabit Etherchannel bundle. This configuration is incomplete and would not classify default traffic appropriately until the physical interface was configured to support a service-fragment traffic class.

```
policy-map subscriber
 class voice
  priority level 1
 class video
  priority level 2
 class class-default fragment BE
  shape average 100000000
  bandwidth remaining ratios 80
policy-map aggregate-member-link
 class BestEffort service-fragment BE
  shape average 100000000
!
interface Port-channel1
 ip address 172.16.2.3 255.255.0.0
!
interface Port-channel1.100
 encapsulation dot1Q 100
 ip address 192.168.2.100 255.255.255.0
 service-policy output subscriber
!
```

Troubleshooting Tips

This configuration will not work until a service-fragment default traffic class is created to classify the default traffic classes marked as fragments. This service-fragment traffic class must be configured for this configuration to have any affect on network traffic.

How to Configure QoS: Policies Aggregation MQC

Some backward-compatibility exists between support of policies aggregation feature configuration in Cisco IOS XE Release 2.6 and prior Cisco IOS XE software releases. However, we recommend that you follow these upgrade guidelines for any physical interface where you want to move to the QoS: Policies Aggregation MQC Support for Multiple Queue Aggregation at Main Interface feature configuration.

For best results, you should upgrade any service policies configuration that you implemented prior to Cisco IOS XE Release 2.6, to the latest supported configuration.

The original and enhanced QoS: Policies Aggregation feature configuration can only reside on the same Cisco ASR 1000 Series Router if the mixed configuration does not reside on the same physical interface. In other words, you can support the original configuration for one physical interface, and the enhanced configuration on a different physical interface.

The QoS: Policies Aggregation MQC Support for Multiple Queue Aggregation at Main Interface feature requires the same configuration of a fragment traffic class as the original feature, using the **class class-default fragment** command to enable and then define all subinterface policies aggregation, both for the default traffic class and the other traffic classes.

In the QoS: Policies Aggregation MQC Support for Multiple Queue Aggregation at Main Interface feature, the queueing features for the aggregate class queues (with traffic from the corresponding classes identified at the subinterfaces), are configured at the main-interface policy map.

Upgrading Your Service Policies for QoS: Policies Aggregation MQC

Before You Begin

Upgrading your service policies to support the QoS: Policies Aggregation MQC Support for Multiple Queue Aggregation at Main Interface feature assumes the following network conditions:

- The corresponding class-map statements appropriate for your network traffic are already configured.
- QoS service policies aggregation has been previously configured and applied for the main-interface policy map for a given physical interface and its corresponding subinterfaces, or subscriber interfaces, prior to Cisco IOS XE Release 2.6 for the default traffic class.
- A port on the same physical interface where you have previously configured the service policies aggregation feature prior to Cisco IOS XE Release 2.6 needs to support the configuration for the QoS: Policies Aggregation MQC Support for Multiple Queue Aggregation at Main Interface.

Upgrade Tasks

SUMMARY STEPS

1. Configure the service policies for the QoS: Policies Aggregation MQC Support for Multiple Queue Aggregation at Main Interface feature.
2. Remove any service policies configured prior to Cisco IOS XE Release 2.6 for any prior configured policies aggregation features using the **no service-policy** and **no policy-map** commands as follows:
3. Apply the new service policies for the QoS: Policies Aggregation MQC Support for Multiple Queue Aggregation at Main Interface feature at the appropriate interfaces using the **service-policy output** command as follows:

DETAILED STEPS

Step 1 Configure the service policies for the QoS: Policies Aggregation MQC Support for Multiple Queue Aggregation at Main Interface feature.

See the tasks described in the "Configuring QoS Policies Aggregation MQC Traffic Classes" section on page 18.

Step 2 Remove any service policies configured prior to Cisco IOS XE Release 2.6 for any prior configured policies aggregation features using the **no service-policy** and **no policy-map** commands as follows:

a) At each of the subinterfaces, configure the **no service-policy** command. Be sure to remove the policies at the subinterfaces first.

b) At the physical interface, configure the **no service-policy**command.

Step 3 Apply the new service policies for the QoS: Policies Aggregation MQC Support for Multiple Queue Aggregation at Main Interface feature at the appropriate interfaces using the **service-policy output** command as follows:

a) At the physical interface, configure the **service-policy output** command.

b) At each of the subinterfaces, configure the **service-policy output** command.

Configuring QoS: Policies Aggregation MQC Traffic Classes

Configuring Traffic Classes on the Subscriber Interface

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **policy-map** *policy-map-name*
4. **class** *class-name*
5. **account** [**drop**]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**policy-map** *policy-map-name* **Example:** `Router(config)# policy-map subscriber1`	Specifies the name of the traffic policy to configure and enters policy map configuration mode.

	Command or Action	Purpose
Step 4	**class** *class-name* **Example:** `Router(config-pmap)# class EF`	Specifies the name of the traffic class to be aggregated at the main-interface policy map, and enters policy-map class configuration mode. **Note** Do not configure any queueing features for this class. Queueing is configured and aggregated at the main-interface policy map for all subinterfaces associated with this class and physical interface.
Step 5	**account** [**drop**] **Example:** `Router(config-pmap-c)# account`	(Optional) Enables collection of statistics for packets matching the traffic class where this command is configured, where the **drop** keyword collects all packet drop statistics. Collection of drop statistics is the default.

Example

The following example configures the EF traffic class for policies aggregation at the subscriber subinterface with collection of drop statistics:

```
policy-map subscriber1
 class EF
 account
```

What to Do Next

Perform this task for all traffic classes that you want to aggregate, then perform the task in the "Configuring the Fragment Traffic Class on a Subinterface" section on page 19.

Configuring the Fragment Traffic Class on a Subinterface

What to Do Next

If you are upgrading your subinterface policy-map configuration from an earlier implementation of the QoS: Policies Aggregation feature, then remove the current service-policy from the subinterface using the **no service-policy** command.

Apply the new policy map to outbound traffic on the subinterface using the **service-policy output** command.

Configuring Traffic Classes at the Main Interface

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **policy-map** *policy-map-name*
4. **class** *class-name*
5. **priority level** *level*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**policy-map** *policy-map-name* **Example:** `Router(config)# policy-map main-interface`	Specifies the name of the traffic policy to configure and enters policy map configuration mode.
Step 4	**class** *class-name* **Example:** `Router(config-pmap)# class EF`	Specifies the name of the traffic class to be aggregated at the main-interface policy map, and enters policy-map class configuration mode.
Step 5	**priority level** *level* **Example:** `Router(config-pmap-c)# priority level 1`	Enters a QoS configuration command. The queueing features that are currently supported are **bandwidth**, **priority**, **shape**, and **random-detect exponential-weighting-constant**. Multiple QoS queueing commands can be entered.

Example

The following example configures three traffic classes at the main-interface policy map, along with the aggregate service-fragment data class:

```
policy-map main-interface
 class voice
  priority level 1
 class video
  priority level 2
 class AF1
  bandwidth remaining ratio 90
 class data service-fragment BestEffort
  shape average 400000000
  bandwidth remaining ratio 1
```

What to Do Next

Perform this task to define queueing features for all traffic classes that you want to aggregate, then perform the task in the "Configuring the Service Fragment Traffic Class at the Main Interface" section on page 21.

Configuring the Service Fragment Traffic Class at the Main Interface

What to Do Next

After configuring multiple default class statements as fragments in a policy-map, a separate policy-map with a class statement using the **service-fragment** keyword must be configured to apply QoS to the class statements configured as fragments.

This process is documented in the "Configuring a Service Fragment Traffic Class" section.

Configuring QoS: Policies Aggregation MQC Support

The QoS: Policies Aggregation MQC Support for Multiple Queue Aggregation at Main Interface feature also supports configuration of the enhanced service policies on Gigabit Etherchannels according to the subscriber and main-interface configuration guidelines described for this enhancement.

For more information, see the following sections:

Verifying the Traffic Policy Class Policy Information and Drop Statistics

To display information about policy-map configuration and subscriber drop statistics enabled using the account command, use the **show policy-map interface** command:

```
Router# show policy-map interface port-channel 1.1
Port-channel1.1
   Service-policy input: input_policy
     Class-map: class-default (match-any)
       0 packets, 0 bytes
       5 minute offered rate 0000 bps, drop rate 0000 bps
       Match: any
       QoS Set
       dscp default
       No packet marking statistics available
   Service-policy output: Port-channel_1_subscriber
     Class-map: EF (match-any)
       105233 packets, 6734912 bytes
       5 minute offered rate 134000 bps, drop rate 0000 bps
       Match: dscp ef (46)
       Match: access-group name VLAN_REMARK_EF
       Match: qos-group 3
       Account QoS statistics
         Queueing
           Packets dropped 0 packets/0 bytes
       QoS Set
       cos 5
       No packet marking statistics available
       dscp ef
       No packet marking statistics available
     Class-map: AF4 (match-all)
       105234 packets, 6734976 bytes
       5 minute offered rate 134000 bps, drop rate 0000 bps
       Match: dscp cs4 (32)
       Account QoS statistics
         Queueing
           Packets dropped 0 packets/0 bytes
       QoS Set
       cos 4
       No packet marking statistics available
```

```
    Class-map: AF1 (match-any)
      315690 packets, 20204160 bytes
      5 minute offered rate 402000 bps, drop rate 0000 bps
      Match: dscp cs1 (8)
      Match: dscp af11 (10)
      Match: dscp af12 (12)
      Account QoS statistics
        Queueing
          Packets dropped 0 packets/0 bytes
      QoS Set
      cos 1
      No packet marking statistics available
    Class-map: class-default (match-any) fragment Port-channel_BE
      315677 packets, 20203328 bytes
      5 minute offered rate 402000 bps, drop rate 0000 bps
      Match: any
      Queueing
        queue limit 31250 bytes
        (queue depth/total drops/no-buffer drops) 0/0/0
        (pkts output/bytes output) 315679/20203482
        bandwidth remaining ratio 1
```

Configuration Examples for QoS: Policies Aggregation

Example: QoS: Policies Aggregation

Note This example shows a sample configuration that is supported in the original QoS: Policies Aggregation feature prior to Cisco IOS XE Release 2.6.

In the following example, QoS: Policies Aggregation is used to define a fragment class of traffic to classify default traffic using the default traffic class named BestEffort. All default traffic from the policy maps named subscriber1 and subscriber2 is part of the fragment default traffic class named BestEffort. This default traffic is then shaped collectively by creating a class called data that uses the **service-fragment** keyword and the **shape** command.

Note the following about this example:

- The *class-name* for each fragment default traffic class is "BestEffort."
- The *class-name* of "BestEffort" is also used to define the class where the **service-fragment** keyword is entered. This class applies a shaping policy to all traffic forwarded using the fragment default traffic classes named "BestEffort."

```
policy-map subscriber1
 class voice
  set cos 5
  priority level 1
 class video
  set cos 4
  priority level 2
 class class-default fragment BestEffort
  shape average 200000000
  bandwidth remaining ratio 10
```

```
policy-map subscriber 2
 class voice
  set cos 5
  priority level 1
 class video
  set cos 4
  priority level 2
 class class-default fragment BestEffort
  shape average 200000000
  bandwidth remaining ratio 10
policy-map input_policy
 class class-default
  set dscp default
policy-map main-interface
 class data service-fragment BestEffort
  shape average 400000000
interface portchannel1.1001
 encapsulation dot1q 1001
 service-policy output subscriber1
 service-policy input input_policy
interface portchannel1.1002
 encapsulation dot1q 1002
 service-policy output subscriber2
 service-policy input input_policy
interface gigabitethernet 0/1
 description member-link1
 port channel 1
 service-policy output main-interface
interface gigabitethernet 0/2
 description member-link2
 port channel 1
service-policy output main-interface
```

Example: Gigabit Etherchannel QoS Policies Aggregation

Note This example shows a sample configuration that is supported in the original QoS: Policies Aggregation feature prior to Cisco IOS XE Release 2.6.

In the following example, policy map subscriber is configured with a fragment class named BE. The fragment is then configured as part of a policy map named aggregate-member-link. Policy map subscriber is then attached to the bundle subinterfaces while policy map aggregate-member-link is attached to the physical interface.

```
port-channel load-balancing vlan-manual
class-map match-all BestEffort
!
class-map match-all video
!
class-map match-all voice
!
policy-map subscriber
 class voice
  priority level 1
 class video
  priority level 2
 class class-default fragment BE
  shape average 100000000
```

```
  bandwidth remaining ratios 80
policy-map aggregate-member-link
 class BestEffort service-fragment BE
  shape average 100000000
!
interface Port-channel1
 ip address 10.1.1.3 255.255.0.0
!
interface Port-channel1.100
 encapsulation dot1Q 100
 ip address 10.1.2.1 255.255.255.0
 service-policy output subscriber
!
interface Port-channel1.200
 encapsulation dot1Q 200
 ip address 10.1.2.2 255.255.255.0
 service-policy output subscriber
!
interface Port-channel1.300
 encapsulation dot1Q 300
 ip address 10.1.2.3 255.255.255.0
 service-policy output subscriber
!
interface GigabitEthernet1/1/1
 no ip address
 channel-group 1 mode on
 service-policy output aggregate-member-link
!
interface GigabitEthernet1/1/2
 no ip address
 channel-group 1 mode on
 service-policy output aggregate-member-link
```

Example: QoS: Policies Aggregation MQC Support at Main Interface

Note This example shows a sample configuration that is supported beginning in Cisco IOS XE Release 2.6.

At the main-interface policy map called Port-channel_1_main_policy, the queueing features for the DSCP-based subscriber traffic classes are configured. You can also see the use of byte-based queue limits and random-detect thresholds implemented at the main-interface queues.

The service fragment called Port-channel_BE is also configured to aggregate the traffic from the subscriber class-default fragment class.

```
policy-map Port-channel_1_main_policy
 class EF
  priority level 1
  queue-limit 547500 bytes
 class AF4
  priority level 2
  queue-limit 4037500 bytes
 class AF1
  bandwidth remaining ratio 90
  queue-limit 750000 bytes
  random-detect dscp-based
  random-detect dscp 8 750000 bytes 750000 bytes
  random-detect dscp 10 750000 bytes 750000 bytes
  random-detect dscp 12 600000 bytes 675000 bytes
```

```
 class data service-fragment Port-channel_BE
  shape average 250000000
  bandwidth remaining ratio 1
!
```

In this example, the policy map Port-channel_1_subscriber is configured with a fragment class named Port-channel_BE. (For simplicity, only a single subinterface policy is shown.) This enable queueing and policies aggregation for the subscriber traffic classes at the main-interface policy map.

The Port-channel_1_subscriber policy map identifies the DSCP-based traffic classes of EF, AF4, and AF1 and enables collection of drop statistics for those classes.

```
policy-map Port-channel_1_subscriber
 class EF
  account
  set cos 5
  set dscp ef
 class AF4
  account
  set cos 4
 class AF1
  account
  set cos 1
 class class-default fragment Port-channel_BE
  bandwidth remaining ratio 1
  queue-limit 31250 bytes
!
port-channel load-balancing vlan-manual
!
interface Port-channel1
 no ip address
 no negotiation auto
!
```

The service policies are applied first to the physical interface, and then to the subinterfaces as shown:

```
interface GigabitEthernet1/2/0
 no ip address
 negotiation auto
 no cdp enable
 service-policy output Port-channel_1_main_policy
 channel-group 1
!
interface GigabitEthernet2/2/0
 no ip address
 negotiation auto
 service-policy output Port-channel_1_main_policy
 channel-group 1
!
interface Port-channel1.1
 encapsulation dot1Q 2 primary GigabitEthernet1/2/0 secondary GigabitEthernet2/2/0
 ip address 10.0.0.2 255.255.255.0
 service-policy output Port-channel_1_subscriber
```

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*
Modular Quality of Service Command-Line Interface	"Applying QoS Features Using the MQC" module
Distribution of Remaining Bandwidth Using Ratio	"Distribution of Remaining Bandwidth Using Ratio" module
Class-Based Shaping	"Regulating Packet Flow--Using Class-Based Traffic Shaping" module

Standards

Standard	Title
No new or modified standards are supported, and support for existing standards has not been modified by this feature.	

MIBs

MIB	MIBs Link
CISCO-CLASS-BASED-QOS-MIB	To locate and download MIBs for selected platforms, Cisco IOS XE software releases, and feature sets, use Cisco MIB Locator found at the following URL: 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

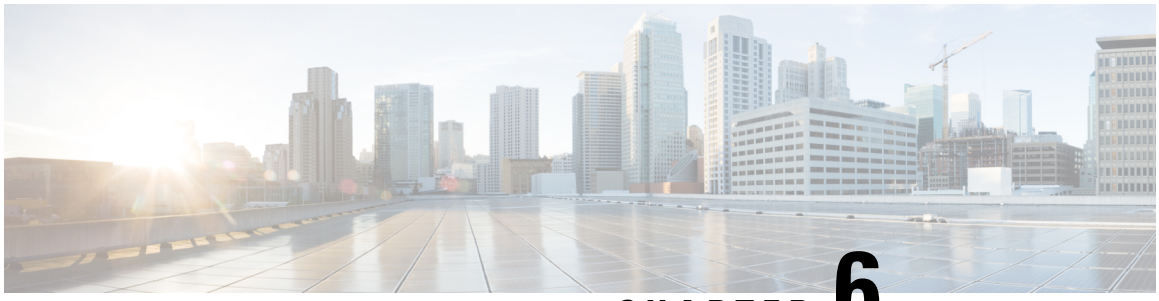
Feature Information for QoS: Policies Aggregation

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 9: Feature Information for QoS: Policies Aggregation

Feature Name	Releases	Feature Information
QoS: Policies Aggregation	Cisco IOS XE Release 2.1	This feature was introduced on Cisco ASR 1000 Series Routers. The following command was modified: **class (policy-map)**.
QoS: Policies Aggregation MQC Support for Multiple Queue Aggregation at Main Interface	Cisco IOS XE Release 2.6	This feature was enhanced to support queueing aggregation at the primary interface for other traffic classes, including DSCP-based classes such as EF, AF1, and AF4 traffic classes. With this enhancement, other traffic classes from different subinterfaces share a common queue for that traffic class. Other enhancements include the ability to configure and show per-subscriber drop statistics on the aggregate queues and byte-based queue limits and WRED thresholds. In Cisco IOS XE Release 2.6, support for the CISCO-CLASS-BASED-QOS-MIB was added. The following commands are new or modified: **account**, **show policy-map interface**.



CHAPTER 6

Legacy QoS Command Deprecation

The functionality provided by these hidden commands has been replaced by similar functionality provided via the modular QoS CLI (MQC). The MQC is a set of a platform-independent commands for configuring QoS on Cisco platforms. This means that you must now provision QoS by defining traffic classes, creating traffic policies containing those classes, and attaching those policies to the desired interfaces. This document lists the hidden commands and their replacement MQC commands.

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.

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 Legacy QoS Command Deprecation

QoS Features Applied Using the MQC

The MQC structure lets you define a traffic class (also called a class map), create a traffic policy (also called a policy-map), and attach the traffic policy to an interface. This comprises 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. A traffic policy 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 to the interface by using the **service-policy** command.

Steps 1 and 3 do not involve legacy QoS hidden commands, which means that they are not within the scope of this document. For more information about these two steps, see the " Applying QoS Features Using the MQC " module in the *Quality of Service Solutions Configuration Guide* .

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 10: 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
• random-detect-group
• random-detect (per VC)
Note This command is not supported in Cisco IOS Release 15.0(1)S.
Command Usage
`Router(config)# random-detect-group group-name [dscp-based|prec-based]`
`Router(config)# interface atm type number`
`Router(config-if)# pvc [name] vpi/vci`
`Router(config-if-atm-vc)# random-detect [attach group-name ]`
	Command Usage
	None (this functionality no longer exists).

Configuring Weighted Random Early Detection	

Hidden, Removed or Unsupported Commands	Replacement MQC Command Sequence

Commands
• random-detect
• random-detect dscp
• random-detect (dscp-based keyword)
• random-detect flow
• random-detect exponential-weighting-constant
• random-detect (prec-based keyword)
• random-detect precedence
Command Usage
`Router(config)# interface type number`
`Router(config-if)# random-detect [number]`
`Router(config-if)# random-detect exponential-weighting-constant exponent`
`Router(config-if)# random-detect flow`
`Router(config-if)# random-detect precedence {precedence|rsvp} min-threshold max-threshold max-probability-denominator`
`Router(config-if)# random-detect prec-based`
`Router(config-if)# random-detect dscp-based`
`Router(config-if)# random-detect dscp dscp-value min-threshold max-threshold[max-probability-denominator]`
	Command Usage
	`Router(config)# policy-map`
	`policy-map-name`
	`Router(config-pmap)# class class-default`
	`Router(config-pmap-c)# random-detect dscp dscp-value min-threshold max-threshold[ mark-probability-denominator]`
	`Router(config-pmap-c)# random-detect clp clp-value min-threshold max-threshold[mark-probability-denominator]`
	`Router(config-pmap-c)# random-detect cos cos-value min-threshold max-threshold[mark-probability-denominator]`
	`Router(config-pmap-c)# random-detect discard-class discard-class-value min-threshold max-threshold[ mark-probability-denominator]`
	`Router(config-pmap-c)# random-detectprecedence ip-precedence min-threshold max-threshold[mark-probability-denominator]`
	`Router(config-pmap-c)# random-detect precedence-based`
	`Router(config-pmap-c)# random-detect ecn`
	`Router(config-pmap-c)# random-detect exponential-weighting-constant exponent`
	`Router(config-pmap-c)# random-detect cos-based`
	`Router(config-pmap-c)# random-detect dscp-based`

Commands
• random-detect flow
• random-detect flow average-depth-factor
• random-detect flow count
Command Usage
`Router(config)# interface type number`
`Router(config-if)# random-detect [number]`
`Router(config-if)# random-detect flow`
`Router(config-if)# random-detect flow count number`
`Router(config-if)# random-detect flow average-depth-factor scaling-factor`
	Command Usage
	None (this functionality no longer exists).

Configuring Bandwidth Allocation	

Hidden, Removed or Unsupported Commands	Replacement MQC Command Sequence

Commands
• max-reserved-bandwidth
Command Usage
`Router(config)# interface type number`
`Router(config-if)# max-reserved-bandwidth percentage`
	Command Usage
	`Router(config)# policy-map policy-map-name`
	`Router(config-pmap)# class class-default`
	`Router(config-pmap-c)# bandwidth{bandwidth-in-kbps |remaining percent percentage | percent percentage}`

Configuring Custom Queueing	

Commands
• custom-queue-list
Note This command is not supported in Cisco IOS Release 15.0(1)S.
Command Usage
`Router(config)# interface type number`
`Router(config-if)#`
`custom-queue-list[list-number]`
	Command Usage
	`Router(config)# policy-map policy-map-name`
	`Router(config-pmap)# class class-default`
	`Router(config-pmap-c)# bandwidth{ bandwidth-in-kbps |remaining percent percentage |percent percentage}`

Configuring Priority Queueing	

Commands
• ip rtp priority
• ip rtp reserve
Command Usage
`Router(config)# interface type number`
`Router(config-if)# ip rtp priority starting-port-number port-range bandwidth`
`Router(config)# interface type number`
`Router(config-if)# ip rtp reserve lowest-udp-port range-of-ports [maximum-bandwidth] 1000`
	Command Usage
	`Router(config)# policy-map policy-map-name`
	`Router(config-pmap)# class class-name`
	`Router(config-pmap-c)# priority`

Configuring Weighted Fair Queueing	

Hidden, Removed or Unsupported Commands	Replacement MQC Command Sequence

Commands
• fair-queue (WFQ)
Command Usage (Cisco IOS Release 15.0(1)S)
`Router(config)# interface type number`
`Router(config-if)# fair-queue`
Command Usage (Cisco IOS Release 15.1(3)T)
`Router(config)# interfacetype number`
`Router(config-if)# fair-queue [congestive-discard-threshold [dynamic-queue-count [reserved-queue-count]]]`
	Command Usage (Cisco IOS Release 15.0(1)S)
	`Router(config)# policy-map policy-map-name`
	`Router(config-pmap)# class class-default`
	`Router(config-pmap-c)# fair-queue`
	Command Usage (Cisco IOS Release 15.1(3)T)
	`Router(config)# policy-map policy-map-name`
	`Router(config-pmap)# class class-default`
	`Router(config-pmap-c)# fair-queue[dynamic-queues ]`

Assigning a Priority Group to an Interface	

Commands
• priority-group
Note This command is not supported in Cisco IOS Release 15.0(1)S.
Command Usage
`Router(config)# interface type number`
`Router(config-if)# priority-group list-number`
	Command Usage
	`Router(config)# policy-map policy-map-name`
	`Router(config-pmap)# class class-default`
	`Router(config-pmap-c)# priority`
	`Router(config-pmap-c)# priority bandwidth-in-kbps [burst-in-bytes]`
	`Router(config-pmap-c)# priority percent percent [burst-in-bytes]`
	`Router(config-pmap-c)# priority level level`
	`Router(config-pmap-c)# priority level level [bandwidth-in-kbps [burst-in-bytes]]`
	`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
• frame-relay congestion threshold de
Command Usage
`Router(config)# map-class frame-relay map-class-name`
`Router(config-map-class)# frame-relay congestion threshold de percentage`
	Command Usage
	`Router(config)# policy-map policy-map-name1`
	`Router(config-pmap)# class class-default`
	`Router(config-pmap-c)# random-detect discard-class-based`
	`Router(config-pmap-c)# random-detect discard-class discard-class min-threshold max-threshold`
	`Router(config-pmap-c)# exit`
	`Router(config-pmap)# exit`
	`Router(config)# policy-map shape`
	`Router(config-pmap)# class class-default`
	`Router(config-pmap-c)# shape average rate`
	`Router(config-pmap-c)# service-policy policy-map-name1`
	`Router(config-pmap-c)# exit`
	`Router(config-pmap)# exit`
	`Router(config)# policy-map policy-map-name2`
	`Router(config-pmap)# class class-name`
	`Router(config-pmap-c)# set discard-classdiscard-class`

Hidden, Removed or Unsupported Commands	Replacement MQC Command Sequence
Configuring Frame Relay Custom Queueing for Virtual Circuits	

Commands
• frame-relay custom-queue-list
Command Usage
`Router(config)# map-class frame-relay map-class-name`
`Router(config-map-class)# frame-relay custom-queue-list list-number`
	Command Usage
	`Router(config)# policy-map policy-map-name`
	`Router(config-pmap)# class class-default`
	`Router(config-pmap-c)# bandwidth{`*bandwidth-in-kbps* `| remaining percent` *percentage* `| percent`*percentage*`}`

Configuring Frame Relay ECN Bits Threshold	

Commands
• frame-relay congestion threshold ecn
Command Usage
`Router(config)# map-class frame-relay map-class-name`
`Router(config-map-class)# frame-relay congestion threshold ecn percentage`
	Command Usage
	None (this functionality no longer exists).
	The closest equivalent is MQC traffic shaping (not based on ECN).
	`Router(config)# policy-map policy-map-name`
	`Router(config-pmap)# class class-default`
	`Router(config-pmap-c)# shape average rate`

Configuring Frame Relay Weighted Fair Queueing	

Commands
• frame-relay fair-queue
Command Usage
`Router(config)# map-class frame-relay map-class-name`
`Router(config-map-class)# frame-relay fair-queue [discard-threshold [dynamic-queue-count[reserved-queue-count [buffer-limit]]]]`
	Command Usage
	`Router(config)# policy-map policy-map-name`
	`Router(config-pmap)# class class-default`
	`Router(config-pmap-c)# fair-queue`
	`Router(config-pmap-c)# fair-queue queue-limit packets`
	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
• frame-relay ip rtp priority
Command Usage
`Router(config)# map-class frame-relay map-class-name`
`Router(config-map-class)# frame-relay ip rtp priority starting-port-number port-range bandwidth`
	Command Usage
	`Router(config)# policy-map policy-map-name`
	`Router(config-pmap)# class class-name`
	`Router(config-pmap-c)# priority bandwidth-in-kbps [burst-in-bytes]`

Hidden, Removed or Unsupported Commands	Replacement MQC Command Sequence
Assigning a Priority Queue to Virtual Circuits Associated with a Map Class	

Commands
• frame-relay priority-group
Command Usage
`Router(config)# map-class frame-relay map-class-name`
`Router(config-map-class)# frame-relay priority-group group-number`
	Command Usage
	`Router(config)# policy-map policy-map-name`
	`Router(config-pmap)# class class-default`
	`Router(config-pmap-c)# priority`
	`Router(config-pmap-c)# priority bandwidth-in-kbps [burst-in-bytes]`
	`Router(config-pmap-c)# priority percent percentage [burst-in-bytes]`
	`Router(config-pmap-c)# priority level level [percent percentage [burst-in-bytes]]`
	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
• frame-relay adaptive-shaping (becn keyword)
Command Usage
`Router(config)# map-class frame-relay map-class-name`
`Router(config-map-class)# frame-relay adaptive-shaping becn`
	Command Usage
	None (this functionality no longer exists). The closest equivalent is MQC traffic shaping (not based on BECN).
	`Router(config)# policy-map policy-map-name`
	`Router(config-pmap)# class class-default`
	`Router(config-pmap-c)# shape adaptive rate`

Configuring the Frame Relay Rate Adjustment to ForeSight Messages	

Commands
• frame-relay adaptive-shaping (foresight keyword)
Command Usage
`Router(config)# map-class frame-relay map-class-name`
`Router(config)# frame-relay adaptive-shaping foresight`
	Command Usage
	None (this functionality no longer exists).

Enabling Frame Relay Traffic-Shaping FECNs as BECNs	

Hidden, Removed or Unsupported Commands	Replacement MQC Command Sequence

Commands
• frame-relay fecn-adapt
Command Usage
`Router(config)# map-class frame-relay map-class-name`
`Router(config-map-class)#frame-relay fecn-adapt`
	Command Usage
	None (this functionality no longer exists). The closest equivalent is MQC traffic shaping (not based on FECN/BECN).
	`Router(config)# policy-map policy-map-name`
	`Router(config-pmap)# class class-default`
	`Router(config-pmap-c)# shape average rate`

Configuring the Frame Relay Enhanced Local Management Interface	

Commands
• frame-relay qos-autosense
Note This command has not been hidden in Cisco IOS Release 15.0(1)S.
Command Usage
`Router(config)# interface type number`
`Router(config-if)#no ip address`
`Router(config-if)# encapsulation frame-relay`
`Router(config-if)# frame-relay lmi-typeansi`
`Router(config-if)# frame-relay traffic-shaping`
`Router(config-if)# frame-relay qos-autosense`
	Command Usage
	None (this functionality no longer exists).

Configuring Frame Relay Minimum Committed Information Rate (MINCIR)	
Commands • frame-relay mincir **Command Usage** `Router(config)# frame-relay mincir {in | out} bps`	**Command Usage** None (this functionality no longer exists).
Configuring Frame Relay Priority to a permanent virtual circuit (PVC)	

Hidden, Removed or Unsupported Commands	Replacement MQC Command Sequence

Commands
• frame-relay interface-queue
Command Usage
`Router(config)# interface type number`
`Router(config-if)#no ip address`
`Router(config-if)# frame-relay interface-queue priority 10 20 30 40`
	Command Usage
	`Router(config)# policy-map policy-map-name`
	`Router(config-pmap)# class class-default`
	`Router(config-pmap-c)# priority`
	`Router(config-pmap)# class class-default`
	`Router(config-pmap-c)# priority`

Configuring Frame Relay Traffic Shaping	

Commands
• frame-relay bc
• frame-relay be
• frame-relay cir
Note In Cisco IOS Release 15.1(3)T, these commands are not hidden, but they are valid only for SVCs (not PVCs).
Command Usage
`Router(config)# map-class frame-relay map-class-name`
`Router(config-map-class)# frame-relay bc {in | out} committed-burst-size-in-bits`
`Router(config-map-class)# frame-relay be {in | out} excess-burst-size-in-bits`
`Router(config-map-class)# frame-relay cir {in | out} bits-per-second`
	Command Usage
	`Router(config)# policy-map policy-map-name`
	`Router(config-pmap)# class class-default`
	`Router(config-pmap-c)# shape average rate`

Configuring Frame Relay Traffic Shaping on a VC	

Commands
• frame-relay traffic-rate
Command Usage
`Router(config)# map-class frame-relay map-class-name`
`Router(config-map-class)# traffic-rate average [peak]`
	Command Usage
	`Router(config)# policy-map policy-map-name`
	`Router(config-pmap)# class class-default`
	`Router(config-pmap-c)# shape average rate`
	`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	

Hidden, Removed or Unsupported Commands	Replacement MQC Command Sequence
Commands • show queue **Command Usage** `Router# show queue interface`	**Command Usage** `Router# show policy-map interface`
Displaying Queueing Strategies	
Commands • show queueing **Command Usage** `Router# show queueing`	**Command Usage** `Router# show policy-map interface`
Displaying Weighted Random Early Detection (WRED) Information	
Commands • show interfaces random-detect **Command Usage** `Router# show interfaces [type number] random-detect`	**Command Usage** `Router# show policy-map interface`
Displaying WRED Parameter Groups	
Commands • show random-detect-group **Command Usage** `Router# show random-detect-group`	**Command Usage** `Router# show policy-map interface`
Displaying the Traffic-Shaping Configuration, Queueing, and Statistics	

Hidden, Removed or Unsupported Commands	Replacement MQC Command Sequence

Commands
• show traffic-shape
• show traffic-shape queue
• show traffic-shape statistics
Command Usage
`Router# show traffic-shape [interface-type interface-number]`
`Router# show traffic-shape queue [interface-number [dlci dlci-number]]`
`Router# show traffic-shape statistics [interface-type interface-number]`
	Command Usage
	`Router# show policy-map interface`

Displaying Weighted Fair Queueing Information	
Commands • show interfaces fair-queue **Command Usage** `Router# show interfaces [interface-type interface-number] fair-queue`	**Command Usage** `Router# show policy-map interface`

Additional References

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Commands List, All Releases
Defining traffic classes; attaching traffic policies to interfaces	" Applying QoS Features Using the MQC " module in the *Quality of Service Solutions Configuration Guide*
Reference pages for QoS commands	*Cisco IOS Quality of Service Solutions Command Reference*
Reference pages for wide-area networking commands	*Cisco IOS Wide-Area Networking Command Reference*

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 Legacy QoS Command Deprecation

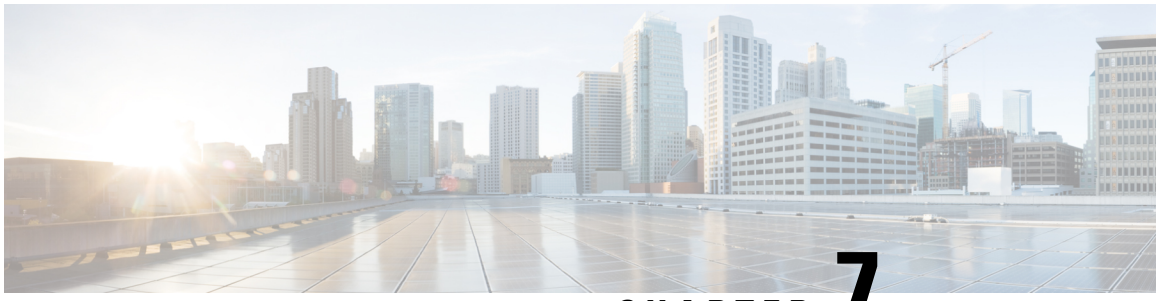
The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train 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 11: Feature Information for Legacy QoS Command Deprecation

Feature Name	Releases	Feature Information
Legacy QoS Command Deprecation: Hidden Commands	15.0(1)S 15.1(3)T	To streamline Cisco IOS QoS, certain commands have been hidden, which means that if you try to view a hidden command by entering a question mark (**?**) at the command line, the command does not appear. However, if you know the command syntax, you can enter it. These commands will be removed in a future release. The functionality provided by these hidden commands is replaced by similar functionality from the modular QoS CLI (MQC), which is a set of a platform-independent commands for configuring QoS. The following commands were modified: **custom-queue-list**, **fair-queue (WFQ)**, **frame-relay adaptive-shaping** (**becn** keyword), **frame-relay adaptive-shaping** (**foresight** keyword), **frame-relay bc**, **frame-relay be**, **frame-relay cir**, **frame-relay congestion threshold de**, **frame-relay congestion threshold ecn**, **frame-relay custom-queue-list**, **frame-relay fair-queue**, **frame-relay fecn-adapt**, **frame-relay ip rtp priority**, **frame-relay priority-group**, **frame-relay qos-autosense**, **ip rtp priority**, **max-reserved-bandwidth**, **priority-group**, **random-detect**, **random-detect dscp**, **random-detect**(**dscp-based** keyword), **random-detect exponential-weighting-constant**, **random-detect flow**, **random-detect flow average-depth-factor**, **random-detect flow count**, **random-detect**(**prec-based** keyword), **random-detect precedence**, **random-detect-group**, **show interfaces fair-queue**, **show interfaces random-detect**, **show queue**, **show queueing**, **show random-detect-group**, **show traffic-shape**, **show traffic-shape queue**, **show traffic-shape statistics**.

Feature Name	Releases	Feature Information
Legacy QoS Command Deprecation: Hidden Commands	Cisco IOS XE Release 2.6	To streamline Cisco IOS XE QoS, certain commands have been hidden, which means that if you try to view a hidden command by entering a question mark (**?**) at the command line, the command does not appear. However, if you know the command syntax, you can enter it. These commands will be removed in a future release. The functionality provided by these hidden commands is replaced by similar functionality from the modular QoS CLI (MQC), which is a set of a platform-independent commands for configuring QoS. The following commands were modified: **custom-queue-list**, **fair-queue (WFQ)**, **frame-relay adaptive-shaping** (**becn** keyword), **frame-relay adaptive-shaping** (**foresight** keyword), **frame-relay bc**, **frame-relay be**, **frame-relay cir**, **frame-relay congestion threshold de**, **frame-relay congestion threshold ecn**, **frame-relay custom-queue-list**, **frame-relay fair-queue**, **frame-relay fecn-adapt**, **frame-relay ip rtp priority**, **frame-relay priority-group**, **frame-relay qos-autosense**, **ip rtp priority**, **max-reserved-bandwidth**, **show interfaces fair-queue**, **show interfaces random-detect**, **show queue**, **show queueing**, **show traffic-shape**, **show traffic-shape queue**, **show traffic-shape statistics**.
Legacy QoS Command Deprecation: Removed Commands	Cisco IOS XE Release 3.2S	The legacy QoS commands were removed. This means that you must use the appropriate replacement MQC commands. The following commands were removed: **custom-queue-list**, **fair-queue (WFQ)**, **frame-relay adaptive-shaping** (**becn** keyword), **frame-relay adaptive-shaping** (**foresight** keyword), **frame-relay bc**, **frame-relay be**, **frame-relay cir**, **frame-relay congestion threshold de**, **frame-relay congestion threshold ecn**, **frame-relay custom-queue-list**, **frame-relay fair-queue**, **frame-relay fecn-adapt**, **frame-relay ip rtp priority**, **frame-relay priority-group**, **frame-relay qos-autosense**, **ip rtp priority**, **max-reserved-bandwidth**, **show interfaces fair-queue**, **show interfaces random-detect**, **show queue**, **show queueing**, **show traffic-shape**, **show traffic-shape queue**, **show traffic-shape statistics**.



CHAPTER 7

QoS Packet Marking

QoS Packet Marking refers to changing a field within a packet either at Layer 2 (802.1Q/p CoS, MPLS EXP) or Layer 3 (IP Precedence, DSCP and/or IP ECN). It also refers to preserving any classification decision that was reached previously.

About

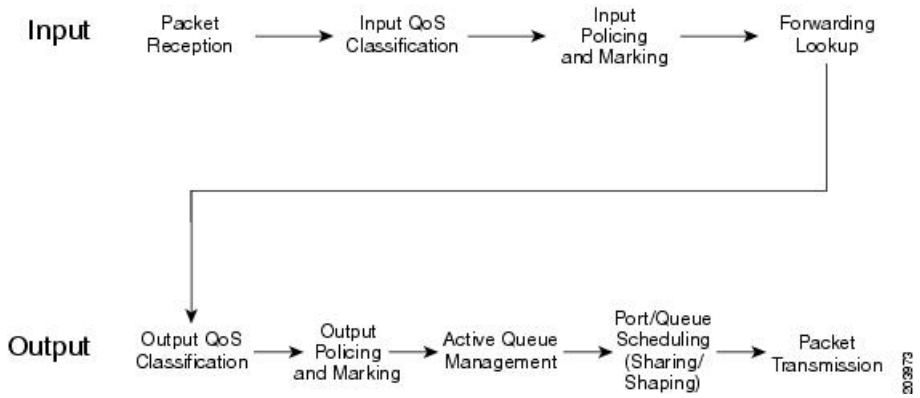
Marking Definition

Marking is similar conceptually to "service class" designation on an airplane ticket: first, business, or economy. This value reflects the level (quality) of service you should receive. Similarly, we mark a value in a packet to indicate the service class (henceforth termed service-class) for that packet as it traverses the network. By examining the marked value, network elements can decide how to treat your packet.

People in business-class may have used a variety of means to achieve that designation. They may have paid extra, used airmiles, or been lucky and booked at the normal rate when no other seat was available. Elsewhere, someone performed the complex task of classification - determining eligibility for a particular service-class then marked the ticket with a mere designation: first-class, business-class, or economy-class. Flight-attendants are unconcerned with how eligibility was determined; they simply look at the class marked on the ticket and provide that level of service.

This dynamic plays out in the networking world. One device may perform complex classification on the data in a flow, determining an appropriate service-class. Other network elements "trust" the value marked in packets they receive and provide service appropriate for that designation.

Figure 4: QoS Packet Processing



Within the context of QoS packet processing, marking occurs after classification and before queuing and is applicable on ingress or egress.

Typically, you would create a *trust boundary* at the edge of the network, then classify and mark packets on the edge device. Then, you would use that marked field for classification and determination of per-hop treatment throughout the network.

Note A trust boundary enables you to apply network-controlled marking on all packets as they enter the network and to remove or modify any non-default markings you did not apply.

Imagine that your system recognizes router ports with attached VoIP devices. You could mark the differentiated services code point (DSCP) value of voice packets as EF (at the edge of the network) and employ DSCP-based classification throughout the network to determine those packets that warrant low latency treatment.

Why Mark Packets

Reasons for marking packets include the following:

- Indicate the treatment you would like a packet to receive as it traverses the network.
- Perform complex classification once. By marking the service class, you can use simpler, less cpu-intensive classification elsewhere in the network.
- Perform classification at a point in the network where you have greater visibility into the flow. For example, if data is encrypted, you cannot perform complex classification such as determining the application carried within that flow. Instead, you could classify prior to encryption and mark a value in the unencrypted header that is visible to network elements along the path.

  As a packet traverses networks managed by different autonomous entities (e.g., the service provider network between two enterprise offices), you may need to re-mark if the markings to service-level designations are inconsistent across those networks.

  As a packet traverses different networking technologies the fields available to indicate service-class may differ. For example, you might carry service-class designation in the DSCP field of an IP packet but if this packet traverses an the multiprotocol label switching (MPLS) network only the MPLS experimental (EXP) field may be usable by network elements to determine service-class. As you enter that portion of the network, you may need to determine the appropriate marking of the MPLS EXP bits.

As a network operator you may contract to accept data from a user at a certain rate. Rather than dropping packets that exceed that rate, you can mark them as a lesser service-class.

Approaches to Marking Packets

You have two main approaches to marking packets: the **set** command and a policer marking action.

Note We only briefly touch upon "policing" actions within this chapter.

set Command

The simplest approach to marking packets on a router is to use the **set** command in a *policy-map* definition. (A policy-map is where you specify a QoS action for each class of traffic that you have defined).

You may decide to classify all RTP ports into a traffic class and mark each packet with AF41. If so, the policy-map may look something like this:

```
policy-map mark-rtp
  class rtp-traffic
    set dscp af41
```

Policer Marking Action

Recall that you can use a policer to drop packets within a traffic class above a defined rate. Alternatively, you could mark packets above that rate and allow them to receive a different per-hop treatment than packets below that rate.

For example, let's say that video traffic arrives at your router marked AF41. You may decide to consider user traffic up to 2 Mbps top *assured forwarding behavior* and to demote any traffic exceeding 2Mbps to AF42 (and considered *out of contract* - non-conforming).

The policy-map might appear as follows:

```
class-map video-traffic
  match dscp af41
!
policy-map enforce-contract
  class video-traffic
    police cir 2m conform-action transmit exceed-action set-dscp-transmit AF42
```

Scope of Marking Action

Similar to classification, marking cannot access every field within a data packet. For example, if an IP packet is encapsulated in multiprotocol label switching (MPLS), it cannot mark the DSCP within the IP header as that would require first de-capsulating from MPLS. However, you could mark the MPLS experimental (EXP) bits.

Note Only Layer 2 and outer Layer 3 headers are available for marking.

Multiple Set Statements

You can configure multiple marking rules within a single class (or policer action). This allows you to mark both Layer 2 and Layer 3 fields within the same packet, or if multiple traffic types are present in the same class, define marking values for each type.

For example consider the following egress policy attached to an Ethernet subinterface:

```
policy-map mark-rtp
  class rtp-traffic
    set cos 4
    set mpls exp topmost 4
    set dscp af41
```

If an MPLS packet were forwarded through this subinterface, the Layer 2 COS field and the EXP bits in the MPLS header would be marked. If an IP datagram were encapsulated in that packet, its DSCP value would remain unchanged. However, if an IP packet were forwarded through the subinterface, its Layer 2 COS value and Layer 3 DSCP values would be marked.

For details, refer to the command pages for set cos, on page 102, set mpls experimental topmost, on page 106, and set dscp, on page 103.

Marking Internal Designators

Cisco routers allow you to mark two internal values (qos-group and discard-class) that travel with the packet within the router but do not modify the packet's contents.

Typically, you mark these in an ingress policy and use them to classify to a traffic-class or WRED drop profile in an egress policy. For example, you may want to base your egress classification on a user's IP address but realize that encryption is configured and the user's IP address is invisible on an egress interface. You could classify their traffic on ingress (before encryption) and set an appropriate qos-group value. On egress, you could now classify based on the qos-group and choose the action accordingly.

Ingress vs. Egress Marking Actions

Certain marking values are only relevant to ingress or egress policies. For example, marking the ATM CLP bit or Frame Relay DE bit in an ingress policy is meaningless as they are discarded when the packet is decapsulated. Similarly, marking qos-group or discard-class in an egress policy is unproductive as these leave the packet unchanged and are discarded when we enqueue the packet for forwarding to the next hop.

Imposition Marking

Under special circumstances, you can mark a header field that has not yet been added to a packet (we term this behavior *imposition marking*).

The most common example of imposition marking is the application ofthe **set mpls experimental imposition** command - you can use it on an ingress interface where a packet may arrive containing an IP datagram and no multiprotocol label switching (MPLS) header. When and if the router encapsulates the datagram with a MPLS header, the EXP bits will be marked accordingly as specified by this command.

Application of the **set dscp tunnel** and **set precedence tunnel** commands (for IPv4 only) represent another example of imposition marking. If an egress policy is applied on a tunnel interface, no tunnel header exists when the policy executes. This means that any marking would apply to the original (eventually inner) IP

header. Using either command, you can mark the tunnel (outer) IP header and leave the original header unchanged.

The following table lists the tunnel types and encapsulation variants that support these commands:

Table 12: Supported DSCP and Precedence Tunnel Marking Configurations

Name	Outer Header (encapsulating)	Inner Header (payload)	Comments
GRE (4 over 4)	IPv4/GRE	IPv4	Supported
GRE (6 over 4)	IPv4/GRE	IPv6	Encapsulation not supported
GREv6 (4 over 6)	IPv6/GRE	IPv4	Encapsulation not supported
GREv6 (6 over 6)	IPv6/GRE	IPv6	Encapsulation not supported
IP-IP	IPv4	IPv4	Supported
IPv6-IP	IPv4	IPv6	Supported
IPv6 (4 over 6)	IPv6	IPv4	Encapsulation not supported
IPv6 (6 over 6)	IPv6	IPv6	Not supported
IPSEC (4 over 4)	IPv4/IPSEC	IPv4	Not supported
IPSEC (6 over 4)	IPv4/IPSEC	IPv6	Not supported
IPSECv6 (4 over 6)	IPv6/IPSEC	IPv4	Encapsulation not supported
IPSECv6 (6 over 6)	IPv6/IPSEC	IPv6	Not supported
mVPN(Multicast VPN)	IPv4/GRE	IPv4	Supported
DMVPN(dynamic multipoint VPN)			Supported
Multipoint GRE			Supported
MPLSoGREv4	IPv4/GRE	MPLS	Not supported
MPLSoGREv6	IPv6/GRE	MPLS	Not supported
L2TP	IPv4/L2TP	PPPoX	Not supported

When a new header is added (encapsulated), any QoS marking in the inner header is copied to the outer header. For example, when an IP datagram is encapsulated with an MPLS header, the default behavior is to copy the IP Precedence bits from the IP header to the MPLS EXP bits in the newly-imposed header.

Regarding header disposition, we typically do not copy any outer marking(s) to the inner header. For example, at the endpoint for a GRE tunnel, let's say that we receive a packet with different DSCP values in the outer and inner IP headers. When we remove the outer header we do not copy its DSCP value to the inner header.

For examples of configuring Imposition Marking, see Example 4: Configuring Tunnel Imposition Marking, on page 89 and Example 5: Using Tunnel Imposition Marking to Remark for an SP Network, on page 100.

For command details, please refer to set mpls experimental imposition, on page 106, set dscp tunnel, on page 104, and set precedence tunnel, on page 107.

Configuration Examples

Example 1: Configuring Ingress Marking

You can set up a trust boundary at the edge of a network (where marking is used) to indicate service-class for some traffic and to bleach all other traffic (see *** below). Enforcing a trust boundary at all ingress ports to the network allows you to maintain control of which applications are mapped to each service-class within the network:

```
policy-map ingress-marking
  class voice
    set dscp ef
  class video
    set dscp af41
  class scavenger
    set dscp cs1
  class class-default                                    ***
    set dscp 0                                           ***
!
interface gigabitethernet1/0/0
  Service-policy in ingress-marking
```

For details, refer to the page set dscp, on page 103.

Example 2: Configuring Egress Marking

If a different administrator controls a portion of a network path and uses a different DSCP to service-class mapping, egress marking may be necessary (e.g., within your enterprise, you classify 12 distinct classes of traffic as described in RFC4594). However, your service provider only provides a three-class model.

You may also need egress marking to indicate treatment for certain classes in a Layer 2 network (like Ethernet, frame-relay, or ATM switched networks):

```
policy-map egress-marking
  class scavenger
    set atm-clp
```

For command details, refer to the page set atm-clp, on page 102.

Example 3: Configuring MPLS EXP Imposition

With MPLS, a provider edge (PE) router encapsulates datagrams or frames with MPLS headers. Switching decisions within the core are based on the MPLS headers without visibility into the encapsulated data.

Consider a Layer 3 MPLS network where IPv4 datagrams are encapsulated in MPLS headers. On the customer edge (CE) facing interface we have visibility into the IPv4 header of the packet. On the core-facing interface, we have encapsulated datagrams with MPLS headers and we cannot see beyond those headers.

By default, we copy the IP precedence to the MPLS EXP bits. What if we want to override this behavior? We can't parse the IPv4 type of service byte on the core-facing interface. We can, however, parse the IP header on ingress and store the EXP value we plan to set when MPLS headers are added. Although MPLS headers

are absent when we execute the command, the router retrieves the instruction and marks the EXP bits on the egress interface:

```
policy-map mpls-exp-remark
  class voice
    set mpls experimental imposition 5
  class video
    set mpls experimental imposition 4
  class scavenger
    set mpls experimental imposition 0
!
interface gigabitethernet1/0/0
  policy-map input mpls-exp-remark
```

For command details, refer to the page set mpls experimental imposition, on page 106.

Example 4: Configuring Tunnel Imposition Marking

Conceptually, tunnel and MPLS EXP imposition marking are similar. We want to mark a value in a header that has not yet been added to the packet and with a Layer 3 tunneling technology like GRE or IPinIP, a Layer 3 datagram may be encapsulated with an outer IP header. (Refer to Imposition Marking, on page 86.)

Let's say that we have a DMVPN network where a branch location encrypts data and encapsulates it with a GRE header before sending it over a public IP network. An administrator may attach a policy-map to the tunnel interface to prioritize applications within that tunnel and may also need to mark the DSCP of the outer IP header to indicate service-class within the provider's network. When the policy is executed, the outer header has not yet been added and commands like **set dscp** or **set precedence** would mark the inner IP header.

To solve the problem, we use the **set dscp tunnel** and **set precedence tunnel** commands, as they allow you to set the value in an outer header that has not yet been added.

In the following example, voice and video traffic are classified and queued separately within the enterprise network. The service provider has a smaller number of service-classes and we have decided to put both voice and video into the priority class within the provider's network.

By marking the DSCP in the outer tunnel header we achieve this yet preserve original markings in the inner header:

```
policy-map mark-outer-gre-header
  class voice
    priority level1 percent 20
    set dscp tunnel ef
  class video
    priority level 2 percent 20
    set dscp tunnel ef
!
interface tunnel100
  service-policy out mark-outer-gre-header
```

For command details, refer to the page set dscp tunnel, on page 104.

Example 5: Configuring QoS-Group Marking

Occasionally, you may want to base egress queuing on ingress classification. For example, let's say you want more than 8 egress queues on a MPLS-enabled interface. Using egress classification, you are limited to MPLS EXP bits and therefore 8 classes. As a solution, you could perform classification on the ingress interface and

set a QoS group for packets that match that classification. QoS group has relevance only within the current router; it doesn't alter anything in the packet header. Instead, it's a value associated with the packet as it passes through the router.

In the following example we use Network Based Application Recognition (NBAR) classification on ingress and mark both telepresence and jabber video with qos-group 4. In the egress policy we classify based on the qos-group we marked on ingress (see "***"):

```
class-map telepresence-video
  match protocol telepresence-media
class-map jabber-video
  match protocol cisco-jabber-video
class-map egress-video-traffic                          ***
  match qos-group 4                                     ***
!
policy-map mark-qos-group
  class telepresence-video
    set qos-group 4
  class jabber-video
    set qos-group 4
!
policy-map egress-queuing
  class egress-video-traffic
    bandwidth remaining percent 50
!
interface gig 1/0/0
  service-policy in mark-qos-group
!
interface serial1/1/0
  service-policy out egress-queuing
```

For command details, refer to the page set qos-group, on page 107.

Example 6: Configuring Discard-Class Marking

In Example 5: Configuring QoS-Group Marking, on page 89, we marked both telepresence video and jabber video with qos-group 4 and placed both of these applications into the same egress queue.

What if we want to run Weighted Random Early Detection (WRED) on the egress queue and drop the jabber video first during congestion. Typically, WRED examines the precedence or DSCP value to determine drop thresholds for a flow. However, as indicated in Example 3: Configuring MPLS EXP Imposition, on page 88, we do not have visibility into the IP header. A solution is to mark a second internal value named discard-class. Then, we could use the qos-group to select the egress class (and queue) and the discard-class to select the WRED drop profile within that class.

```
class-map telepresence-video
  match protocol telepresence-media
class-map jabber-video
  match protocol cisco-jabber-video
class-map egress-video-traffic
  match qos-group 4
!
policy-map mark-qos-group
  class telepresence-video
    set qos-group 4
    set discard-class 1
  class jabber-video
    set qos-group 4
    set discard-class 2
```

```
!
policy-map egress-queuing
  class egress-video-traffic
    bandwidth remaining percent 50
    random-detect discard-class-based
    random-detect discard-class 1 24 40
    random-detect discard-class 2 22 30
!
interface gig 1/0/0
  service-policy in mark-qos-group
!
interface serial1/1/0
  service-policy out egress-queuing
```

For command details, refer to the page set discard-class, on page 103.

Verifying QoS Packet Marking

The **show policy-map interface** command is the primary means of verifying any QoS behavior on IOS XE platforms. Although the packet forwarding path (dataplane) is separated from the IOS instance (control plane), statistics are still reported through this well-known IOS command. This functionality is enabled by default.

This table describes the fields we employ in the following sections.

Table 13: show policy-map interface Field Descriptions (those useful for verifying marking)

Field	Description
Service-policy input	Denotes the name of the input service policy applied to the specified interface or VC
Class-map	Specifies the class of traffic being displayed. Output is displayed for each configured class in the policy. The choice for implementing class matches (e.g., match-all or match-any) can also appear adjacent to the traffic class
packets, bytes	Specifies the number of packets (shown in bytes) identified as belonging to the class of traffic being displayed
offered rate	Specifies the rate in bits per second of the packets entering the class
Match	Specifies the match criteria for the traffic class
QoS Set	Details the QoS marking actions configured for the particular class
Packets marked	If enabled, denotes the total number of packets marked for the particular class. If not enabled, you see "Marker statistics: Disabled."

Verifying with the show policy-map interface Command

The **show policy-map interface** command is the primary means of verifying any QoS behavior on IOS XE platforms. Ordinarily, knowing how many packets match a particular class ("class match statistics, " which is enabled by default) and what (if any) marking action is configured suffices to know how many packets were marked by that action.

Note You should understand how *class match statistics* (enabled by default) and *marking statistics* (disabled by default) differ. Typically, the former is sufficient. When a packet "hits" a class, you can assume it is marked. However, if you configure multiple, mutually exclusive marking values, and need to know how many packets were marked with each **set** command, you can enable marking statistics with all its caveats.

Here is an example of ingress marking with a policy attached to a physical interface. In this example, let's say that jabber-video is configured on ports 2000-3000:

```
class-map match-all jabber-video
  match ip rtp 2000 3000
!
policy-map mark-traffic
  class jabber-video
    set dscp af41

show policy-map int g1/0/0
GigabitEthernet1/0/0

  Service-policy input: mark-traffic

    Class-map: jabber-video (match-all)
      850 packets, 51000 bytes                                    note 1
      5 minute offered rate 2000 bps, drop rate 0000 bps
      Match: ip rtp 2000 3000
      QoS Set                                                     note 2
        dscp af41
          Marker statistics: Disabled

    Class-map: class-default (match-any)                          note 3
      0 packets, 0 bytes
      5 minute offered rate 0000 bps, drop rate 0000 bps
      Match: any
```

Footnotes

note 1	Statistics for the class match
note 2	Packet matching section
note 3	Class-Default statistics section

Observe "`Marker statistics: Disabled`" in the output of ingress marking. If you are invoking multiple statistics and find the information provided in the previous output insufficient, you can enable "Packet Marker Statistics."

Verifying with QoS Packet Marking Statistics

Before you begin

Either

- Remove all policy-maps, issue the command, and re-attach all policy-maps.
- Issue the command, save the configuration, and reload the router.

Note Enabling QoS: Packet Marking Statistics may increase CPU utilization on a scaled configuration. Weigh the benefits of displaying statistics information against the increased CPU utilization for your system.

Enabling QoS Packet Marking Statistics

To enable Packet Marking Statistics, issue the **platform qos marker-statistics** command in configuration mode.

Displaying QoS Packet Marking Statistics

To display the packet statistics of all classes that are configured for all service policies either on the specified interface (or subinterface) or on a specific Permanent Virtual Circuit (PVC), use the **show policy-map interface** command.

When we singularly-configure marking in a policy-map, the output from an ASR 1000 Series Aggregation Services Router would appear as follows:

```
policy-map remark-af41
  class af41-traffic
    set dscp tunnel ef
```

Let's place this map on a tunnel interface with traffic marked af41 in the user's IP header and DSCP marked EF in the GRE IP header. The output of the **show policy-map interface** will appear as follows:

```
show policy-map interface tunnel1

  Service-policy output: remark-af41

    Class-map: af41-traffic (match-all)
      978 packets, 68460 bytes                                  note 1
      5 minute offered rate 2000 bps, drop rate 0000 bps
      Match:  dscp af41 (34)
      QoS Set                                                   note 2
        dscp tunnel ef
          Marker statistics: Disabled                           note 3

    Class-map: class-default (match-any)
      365 packets, 25550 bytes
      5 minute offered rate 0000 bps, drop rate 0000 bps
      Match: any
```

Footnotes

note 1	Displays the class match statistics (assume all "observed" packets are marked AF41).
note 2	Marking is the only action configured.

note 3	Per-set action statistics are disabled by default.

Now, if we enable marking statistics, output from the **show policy-map interface** command would appears as follows:

```
show policy-map interface tunnel1

  Service-policy output: remark-af41

    Class-map: af41-traffic (match-all)
      575 packets, 40250 bytes
      5 minute offered rate 1000 bps, drop rate 0000 bps
      Match:  dscp af41 (34)
      QoS Set
        dscp tunnel ef
          Packets marked 575                    note

    Class-map: class-default (match-any)
      0 packets, 0 bytes
      5 minute offered rate 0000 bps, drop rate 0000 bps
      Match: any
```

Footnote

note	We have now enabled marking statistics but in this example the information is redundant.

For command details, refer to the page set dscp tunnel, on page 104.

Validating the Dataplane Configuration

To verify that the dataplane configuration reflects the IOS control plane configuration, use the **show platform hardware qfp active feature qos interface [input|output]** command, which engages only if issued before you attach any policy-map to an interface. So, you must do one of the following:

- Remove all policy-maps, issue the command and re-attach all policy-maps.
- Issue the command, save the configuration and reload the router.

In the following output, notice that we have configured the actions and set the values on the dataplane:

```
show platform hardware qfp active feature qos interface g1/0/0 input

Interface: GigabitEthernet1/0/0, QFP interface: 12
  Direction: Input
    Hierarchy level: 0
    Policy name: mark-traffic
      Class name: jabber-video, Policy name: mark-traffic
        QOS Set:
            dscp 34                                                    note
      Class name: class-default, Policy name: mark-traffic
```

Footnote

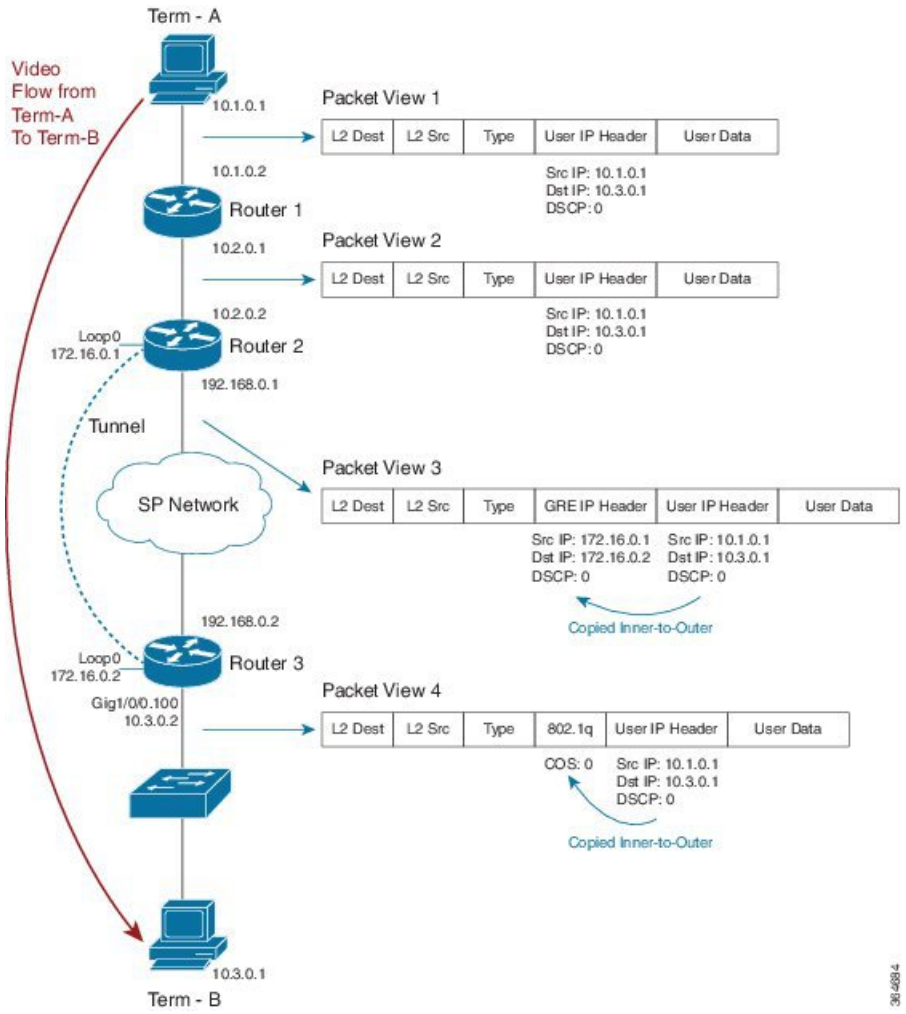
note	Dataplane is programmed to mark.

Network-Level Configuration Examples

In the scenarios that follow, a video-flow transits from Terminal-A to Terminal-B.

Example 1: Propagating Service-Class Information Throughout the Network

Figure 5: Propagating Service-Class Information Throughout the Network

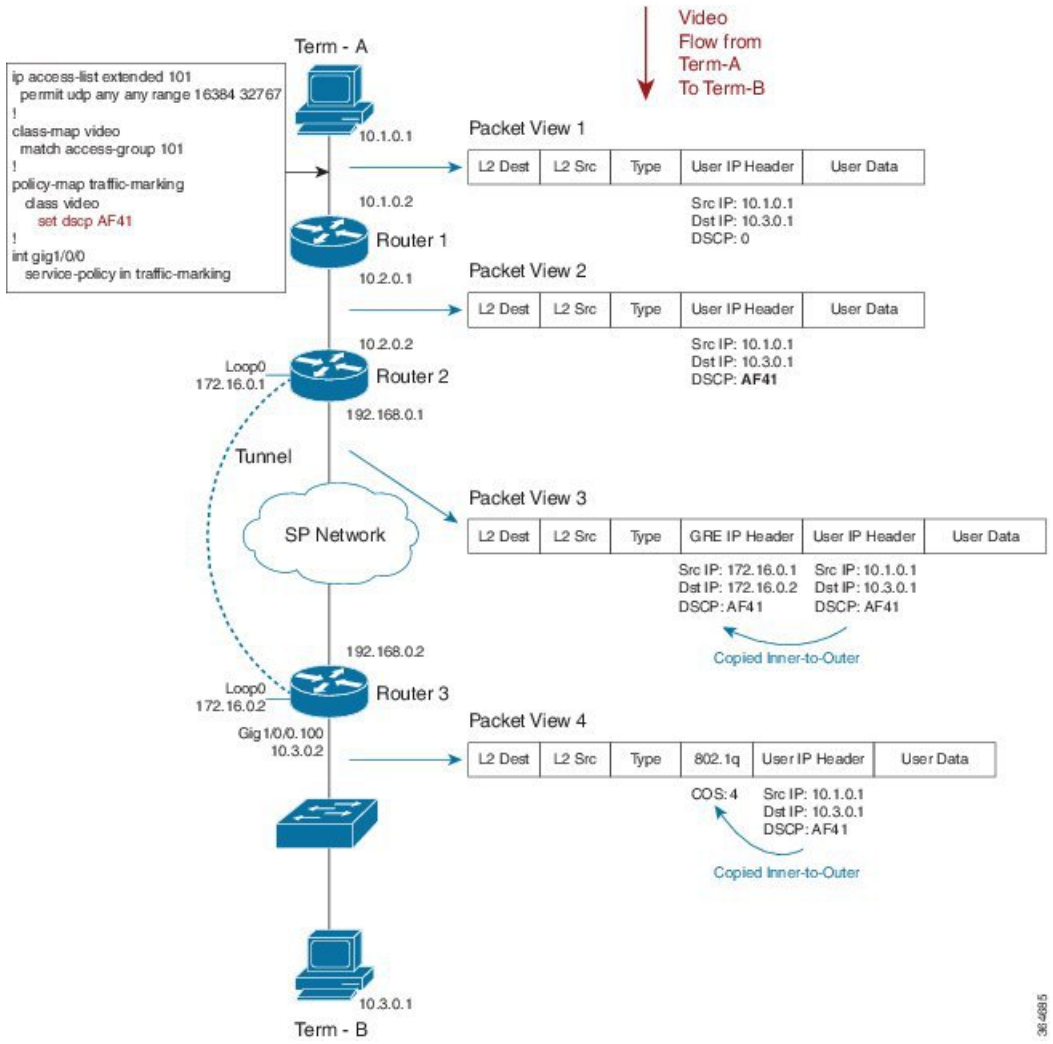


Imagine that an application marks the video stream with DSCP codepoint 0 (see Packet View 1). To cross the provider's network, we send the stream through a GRE tunnel (possibly encrypted). Packet View 3 shows that we have encapsulated the users' IP datagram in a GRE packet. Notice how the DSCP codepoint is copied by default to the imposed GRE header.

With the last hop at the final destination, Router 3 sends a VLAN tagged packet to a switch (see Packet View 4). Observe that the GRE header was stripped and a Dot1Q header was added due to the VLAN configuration. The precedence portion of the user's DSCP 0 (000 000) is copied by default to the COS bits of the VLAN header. The COS value set is 0 (000).

Example 2: Indicating Service-Class by Marking at the Network's Edge

Figure 6: Indicating Service-Class by Marking at the Network's Edge



In this example, we modify the default behavior by remarking the DSCP of users' traffic in an ingress policy as it enters Router 1. The following code shows how we do this:

```
ip access-list extended 101
 permit udp any any range 16384 32767
!
class-map video
 match access-group 101
!
policy-map traffic-marking
  class video
    set dscp AF41
!
int gig1/0/0
```

```
service-policy in traffic-marking
```

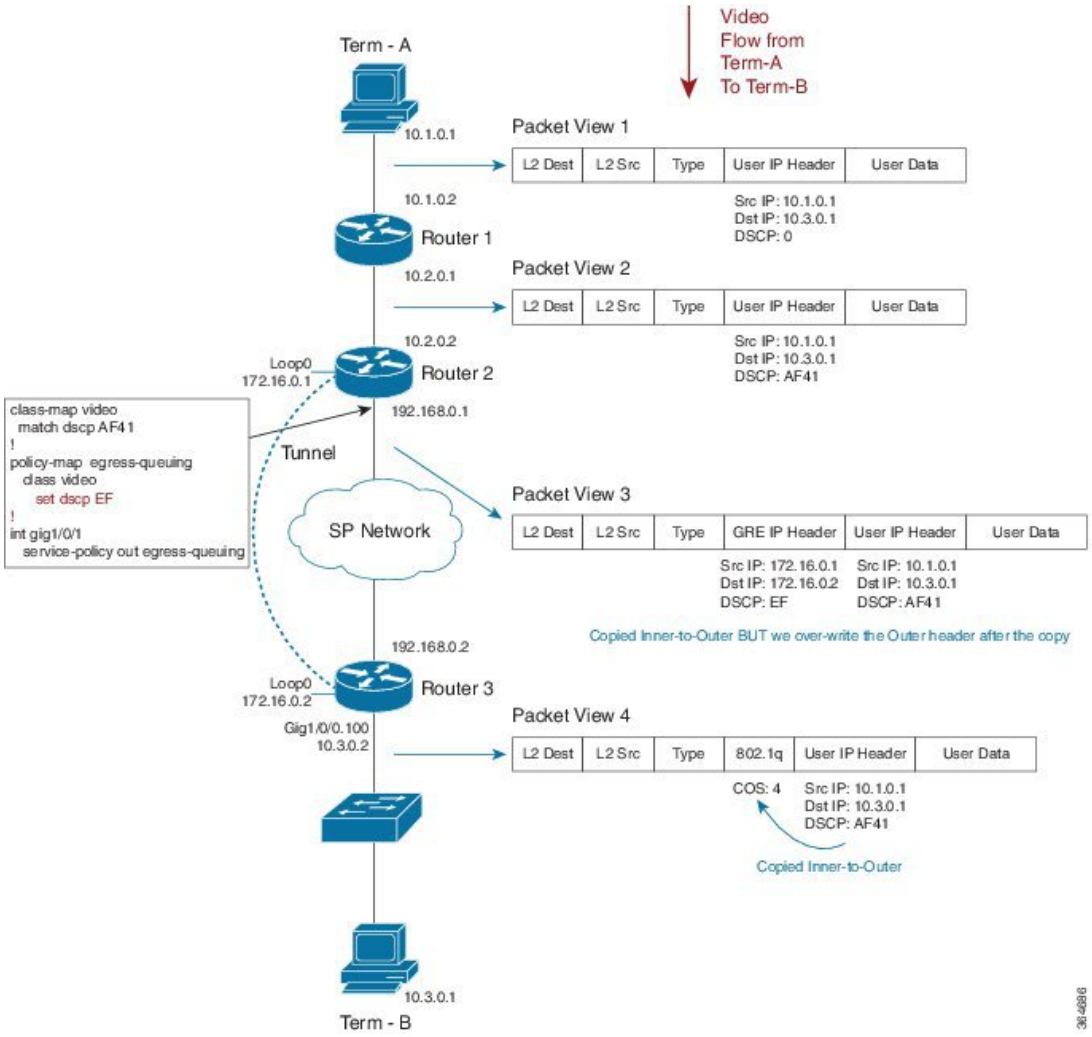
Let's say that we designate video traffic as DSCP AF41 throughout the network. When the packet reaches the GRE interface on egress, its DSCP value has already been changed to AF41 and its behavior matches that in Example 1. We send the stream through a GRE tunnel (possibly encrypted) as it traverses the providers network. Notice how the newly-marked DSCP codepoint (AF41) is copied by default to the imposed GRE header.

When we arrive at our destination, the router sends a VLAN-tagged packet to the last hop (a switch). The precedence portion of the users' DSCP value is copied by default into the COS bits of the VLAN header. As our DSCP is now AF41 (100 010), the COS value will be 4 (100).

For command details, refer to the command page set dscp, on page 103.

Example 3: Remarking Traffic to Match Service Provider Requirements

Figure 7: Remarking Traffic to Match Service Provider Requirements



In this example, we mark the DCSP value within the network while the service provider anticipates a different marking. The following code shows how we handle this:

```
class-map video
  match dscp AF41
!
policy-map  egress-queuing
  class video
    set dscp EF
!
int gig1/0/1
  service-policy out egress-queuing
```

We mark DCSP as AF41 for video within out network while the service provider expects video packets to be marked EF. On the egress Gig interface of Router 2, we add a policy that contains queuing commands (recall that we are only focusing on the marking portion of the configuration in this example).

When the packet reaches the egress physical interface it already has the GRE header imposed and we copy the DSCP value of AF41 from the inner encapsulated datagram. The policy on the physical interface changes the DSCP value in the outer GRE header only.

Note Notice how the inner-user datagram IP header is unchanged.

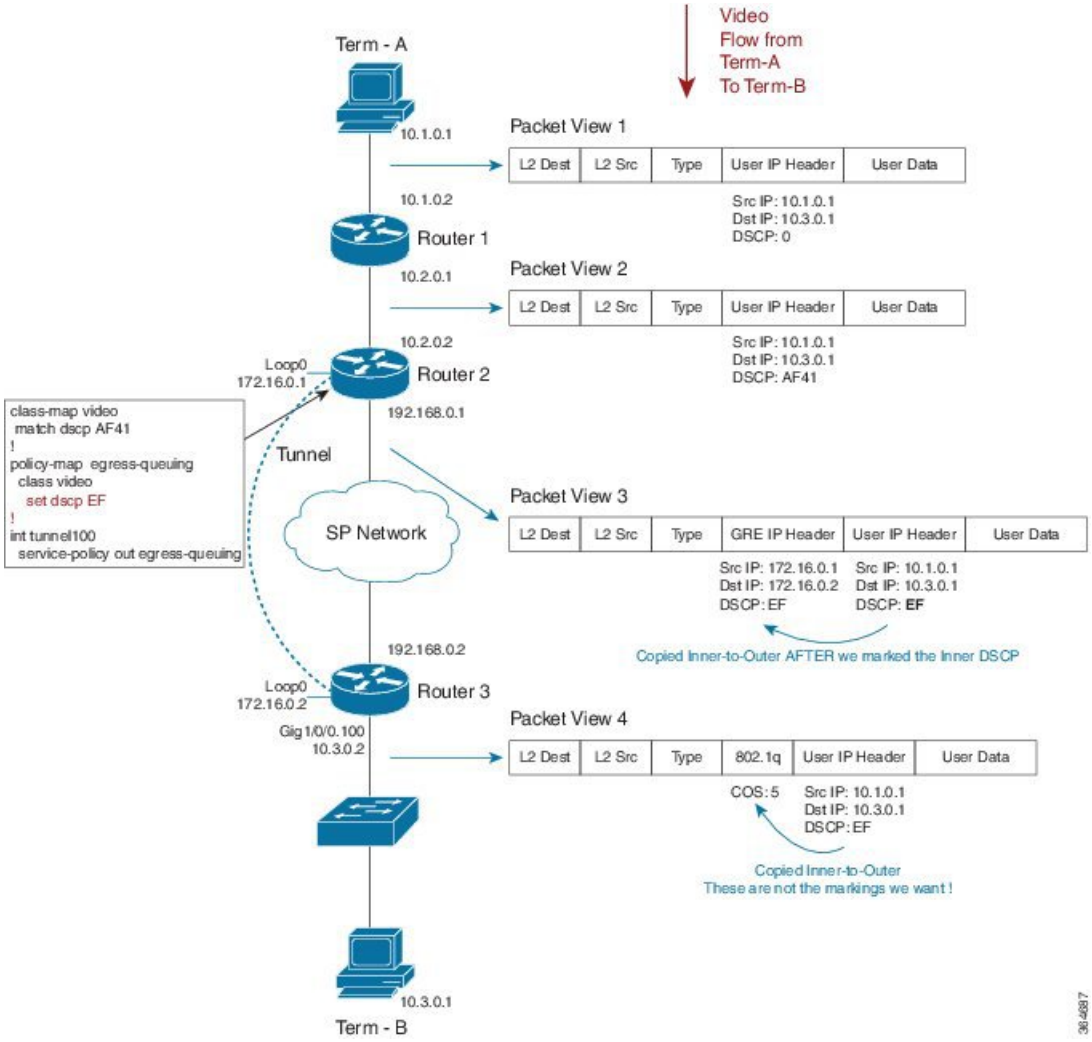
When we reach Router 3 and exit the tunnel, the tunnel GRE header is stripped. Henceforth, only the user datagram IP header is visible, still preserving the AF41 value we marked on ingress to the network.

As in previous examples, the router sends a VLAN-tagged packet to the last hop (a switch). By default, the precedence portion of the User IP Header's DSCP value is copied into the COS bits of the VLAN header (802.1q). As the DSCP value is currently af41 (100 010), the COS value will be 4 (100).

For command details, refer to the page set dscp, on page 103.

Example 4: Remarking on a Tunnel Interface for an SP Network - Potential Gotcha

Figure 8: Remarking on a Tunnel Interface for an SP Network - Potential Gotcha



In this example, we place the QoS policy on the tunnel interface of Router 1 rather than on the physical interface. (There are many advantages to configuring queuing per tunnel rather than as an aggregate policy on the physical interface.) The following code shows how we do this:

```
class-map video
  match dscp AF41
!
policy-map  egress-queuing
  class video
    set dscp EF
!
int tunnel100
```

```
service-policy out egress-queuing
```

We focus solely on the marking portion of the policy. The key point is that marking on the tunnel interface is performed before the tunnel headers are added.

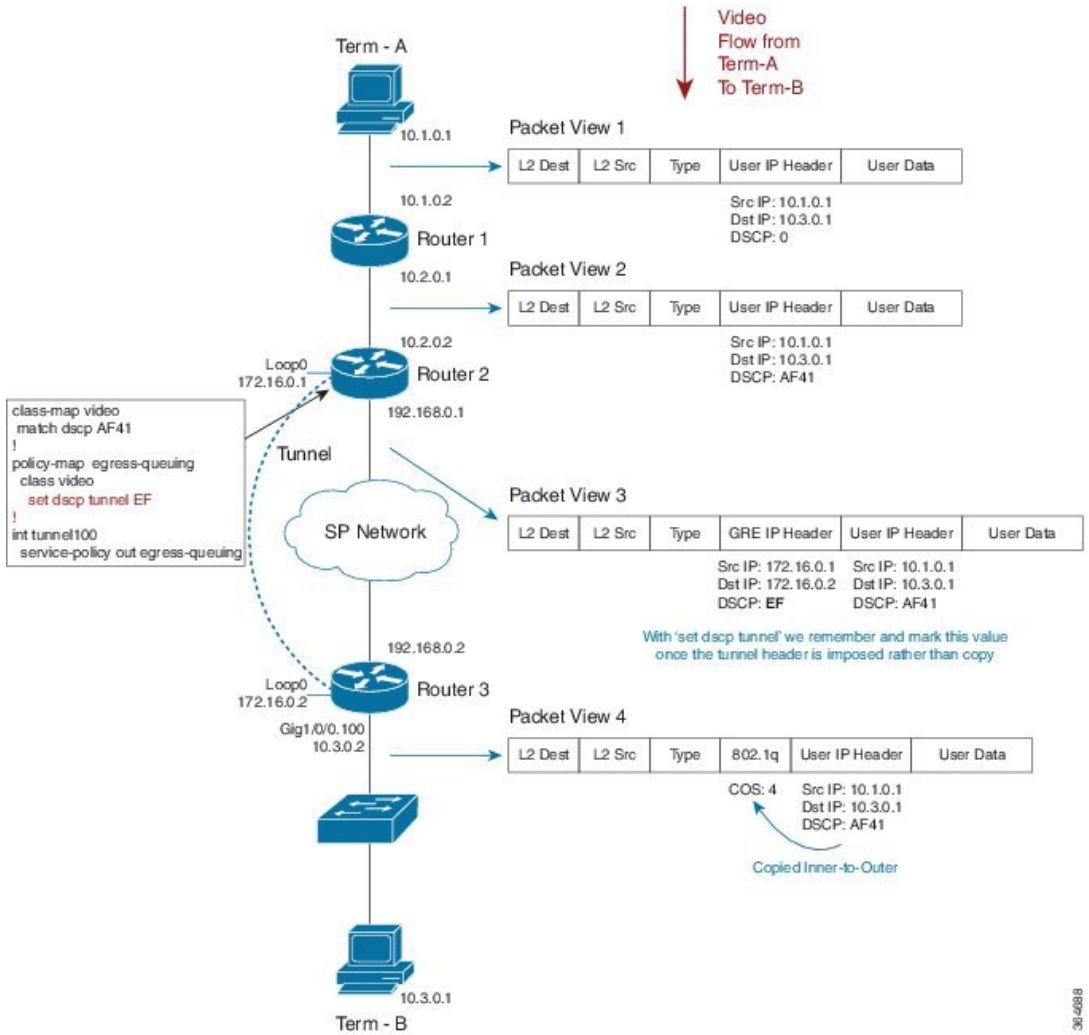
Notice how our policy has over-written the DSCP in the user datagram IP header. Because this happened before GRE encapsulation, we copy the newly-marked value to the outer header.

When we reach Router 3 and exit the tunnel the tunnel GRE header is stripped. Because we marked the user datagram header, the new value propagates through the rest of the network. This is not the behavior we wanted.

For command details, refer to the page for set dscp, on page 103.

Example 5: Using Tunnel Imposition Marking to Remark for an SP Network

Figure 9: Using Tunnel Imposition Marking to Remark for an SP Network



In this example, we use the **set dscp tunnel** *dscp-value* command to alter only the tunnel IP Header:

```
class-map video
  match dscp AF41
!
policy-map  egress-queuing
  class video
    set dscp tunnel EF
!
int tunnel100
  service-policy out egress-queuing
```

We have a QoS policy on the tunnel interface of Router 2 and we have used the **set dscp tunnel** command rather than **set dscp** command.

We have yet to impose the GRE header. The **set dscp tunnel** command dictates that we remember the DSCP value; during encapsulation we use this value instead of copying "inner to outer." Observe that the DSCP value in the users IP datagram header is unchanged. The **set dscp tunnel** command will alter only the tunnel IP header.

For command details, refer to the page for set dscp tunnel, on page 104.

Command Reference

platform qos marker-statistics

To enable individual statistics collection for each marking action in every policy configured on the router, use the **platform qos marker-statistics** command in global configuration mode. To disable packet marking statistics, use the **no** form of this command.

[no] platform qos marker-statistics

Syntax Description

This command has no arguments or keywords.

Command Default

Disabled (no packet marking statistics are displayed). The network operator relies on class match statistics.

Command Modes

policy-map (config-pmap)

Usage Guidelines

This command executes only if issued before any policy-map is attached to an interface. So, you must do one of the following:

- Remove all policy-maps, issue the command and re-attach all policy-maps.
- Issue the command, save the configuration and reload the router.

Note Enabling packet marking statistics may increase CPU utilization on a scaled configuration. So, weigh the benefits of the statistics information against the increased CPU utilization for your system.

set atm-clp

To set the ATM cell loss priority (CLP) bit, use the **set atm-clp** command in policy-map class configuration mode. To disable this setting, use the **no** form of this command.

[no] set atm-clp

Syntax Description

This command has no arguments or keywords.

Command Default

The ATM CLP bit is not set.

Command Modes

policy-map (config-pmap)

Usage Guidelines

On ATM interfaces, you can use the **set atm-clp** command in an outbound policy to set the ATM-CLP bit in ATM cell headers to 1.

This command is supported for ATM, PPPoA, PPPoEoA and L2TPv3 encapsulations. It is not supported if the policy is attached to a tunnel rather than directly to the VC.

You cannot attach a policy-map containing ATM set cell loss priority (CLP) bit QoS to PPP over X (PPPoX) sessions. The map is accepted only if you do not specify the **set atm-clp** command.

For an example using the **set atm-clp** command to configure egress marking, please refer to Example 2: Configuring Egress Marking, on page 88.

set cos

To set the Layer 2 class of service (CoS) value of an outgoing packet, use the **set cos** command in policy-map class configuration mode. To disable this setting, use the **no** form of this command.

[no] set cos *cos-value*

Syntax Description

cos-value	Specifies the IEEE 802.1Q CoS value of an outgoing packet ranging from 0 to 7

Command Default

Either IP Precedence or MPLS EXP bits are copied from the encapsulated datagram.

Command Modes

policy-map (config-pmap)

Usage Guidelines

You can use the **set cos** command to propagate service-class information to a Layer 2 switched network. Although a Layer 2 switch may not be able to parse embedded Layer 3 information (such as DSCP), it might be able to provide differentiated service based on CoS value. Switches can leverage Layer 2 header information, including the marking of a CoS value.

Traditionally the **set cos** command had meaning only in service policies that are attached in the egress direction of an interface because routers discard Layer 2 information from received frames. With the introduction of features like EoMPLS and EVC, the setting of CoS on ingress has meaning, such that you can preserve Layer 2 information throughout the routed network.

set cos-inner

To set the Layer 2 CoS value in the inner VLAN tag of a QinQ packet, use the **set cos-inner** command in policy-map class configuration mode. To disable this setting, use the **no** form of this command.

[no] set cos-inner *cos-value*

Syntax Description

cos-value	Specifies a IEEE 802.1q CoS value ranging from 0-7

Command Default

Either IP Precedence or MPLS EXP bits are copied from the encapsulated datagram.

Command Modes

policy-map (config-pmap)

Usage Guidelines

Traditionally, because routers discard Layer 2 information from received frames, the **set cos-inner** command had meaning only in service policies that are attached in the egress direction of an interface. With the introduction of features like EoMPLS and EVC, the setting of CoS on ingress has significance as you can preserve Layer 2 information throughout the routed network.

set discard-class

To set the QoS discard class for a packet, use the **set discard-class** command in policy-map configuration mode. To disable this setting, use the **no** form of this command.

[no] set discard-class *discard-class-value*

Syntax Description

discard-class-value	Specifies a Discard Class value ranging from 0 to 7

Command Default

The discard-class value associated with a packet is set to 0.

Command Modes

policy-map (config-pmap)

Usage Guidelines

The **set discard-class** command allows you to associate a discard class value with a packet while processed by the router. Setting this value leaves the packet unchanged.

You can use the discard class and discard-class based WRED in egress policies to control which packets are dropped during congestion.

set dscp

To set the DSCP value in the IP header, use the **set dscp** command in policy-map class configuration mode. To disable this setting, use the **no** form of this command.

[no] set dscp *dscp-value*

Syntax Description

dscp-value	Sets the DSCP value in an IP header ranging from 0 to 63. You can specify the value numerically or by using it's well known DiffServe name (e.g., EF)

Command Default

Retain the existing DSCP value in the received packet.

Command Modes

policy-map (config-pmap)

Usage Guidelines

The command may be used in ingress or egress policies.

You can use the DSCP value to indicate the QoS treatment a packet should receive as it traverses a network.

Note The differentiated services architecture using DSCP supersedes use of precedence.

This command marks packets where the outermost Layer 3 header is either IPv4 or IPv6.

If issued in an egress policy-map, this command will not alter the class or queue selection but might influence the WRED drop profile selection.

The **set dscp** and **set ip dscp** commands behave identically, marking both IPv4 and IPv6 packets.

Note This differs from the process of classification wherein the **match ip dscp** command classifies only IPv4 packets while the **match dscp** command classifies both IPv4 and IPv6 packets.

set dscp tunnel

To set the DSCP value in a tunnel header that has not yet been added to a packet, use the **set dscp tunnel** command in policy-map class configuration mode. To disable this setting, use the **no** form of this command.

[no] set dscp tunnel *dscp-value*

Syntax Description

dscp-value	Specifies the DSCP value in a tunnel header ranging from 0 to 63. You can either specify the value numerically or use its well known DiffServe name (e.g. EF).

Command Default

DSCP value from an encapsulated datagram is copied to the newly-imposed tunnel header.

Command Modes

policy-map (config-pmap)

Usage Guidelines

This command only makes sense before a tunnel header is added.

Note You can use this command in either an ingress or egress policy that is attached to a tunnel interface. However, if the latter is attached, the command has no meaning because all headers would be added when the policy is evaluated.

On the Cisco ASR Series Aggregation Services Router, the **set dscp tunnel** command is supported for IPv4 only. See Imposition Marking, on page 86 for a table that lists the supported DSCP tunnel marking configurations.

For an example using this command to encapsulate a Layer 3 datagram with an outer IP header, please refer to Example 4: Configuring Tunnel Imposition Marking, on page 89.

set fr-de

To set the frame-relay (FR) discard eligible (DE) bit, use the **set fr-de** command in policy-map class configuration mode. To disable the setting, use the **no** form of this command.

[no] set fr-de

Syntax Description This command has no arguments or keywords.

Command Default The DE bit is not set when datagrams are encapsulated with frame relay.

Usage Guidelines On serial interfaces configured with Frame Relay encapsulation, you can use the **set fr-de** command in an outbound policy to set the Discard Eligible bit in the Frame Relay header to 1.

set ip dscp

To preserve backwards compatibility, we support two command variants that perform identical functions: **set ip dscp** and **set dscp**. You can use either to mark the DSCP value in the IP header. Please refer to the **set dscp** command page (set dscp, on page 103) for more information.

set ip dscp tunnel

To preserve backwards compatibility, we support two command variants that perform identical functions: **set ip dscp tunnel** and **set dscp tunnel**. Please refer to the **set dscp tunnel** command page (set dscp tunnel, on page 104) for details.

set ip precedence

To preserve backwards compatibility, we support two command variants that perform identical functions: **set ip precedence** and **set precedence**. You can use either to mark the precedence value in the IP header. Please refer to the **set precedence** command page (set precedence, on page 106) for more information.

set ip precedence tunnel

To preserve backwards compatibility, we support two command variants that perform identical functions: **set ip precedence tunnel** and **set precedence tunnel**. Please refer to the **set precedence tunnel** command page (set precedence tunnel, on page 107) for more information.

set mpls experimental imposition

To set the value of the MPLS EXP field on all imposed label entries, use the **set mpls experimental imposition** command in policy-map class configuration mode. To disable this setting, use the **no** form of this command.

[no] set mpls experimental imposition *mpls-exp-value*

Syntax Description

mpls-exp-value	Specifies the MPLS EXP value, which ranges from 0 to 7

Command Default

MPLS value is copied from the appropriate field (usually precedence) in the encapsulated packet.

Command Modes

policy-map (config-pmap)

Usage Guidelines

The **set mpls experimental imposition** command is supported only on input interfaces. Use this command during label imposition to set the MPLS EXP field on all imposed label entries.

For an example of using this command to set the EXP bits in an MPLS header that we use to encapsulate the datagram or frame, please refer to Example 3: Configuring MPLS EXP Imposition, on page 88.

set mpls experimental topmost

To set the MPLS EXP field value in the topmost label, use the **set mpls experimental topmost** command in policy-map class configuration mode. To disable this setting, use the **no** form of this command.

[no]set mpls experimental topmost *mpls-exp-value*

Syntax Description

mpls-exp-value	Specifies the MPLS EXP value ranging from 0 to 7

Command Default

The MPLS EXP value is either copied from the innermost header on encapsulation or remains unchanged.

Command Modes

policy-map (config-pmap)

Usage Guidelines

This command marks packets provided the outermost Layer 3 header is an MPLS label when the command is evaluated.

This command sets the MPLS EXP value in the topmost label only. If multiple labels exist in a stack, the MPLS EXP value in labels other than the topmost remain unchanged.

set precedence

To set the IP Precedence value in the packet header, use the **set precedence** command in policy-map class configuration mode. To disable this setting, use the **no** form of this command.

[no] set precedence *precedence-value*

Syntax Description

precedence-value	Sets the precedence bit in the packet header, which ranges from 0 to 7

Command Default

Retain the precedence value in the received packet.

Command Modes

policy-map (config-pmap)

Usage Guidelines

The command may be used in ingress or egress policies. However, if you issue the command in an egress policy-map, it will not alter the class or queue selection but it may influence the WRED drop profile selection.

By setting a precedence value, you indicate the QoS treatment a packet should receive as it traverses a network.

Note The differentiated services architecture using DSCP largely supersedes the use of precedence.

The **set precedence** and **set ip precedence** commands behave identically, marking packets where the outermost Layer 3 header is IPv4 or IPv6. In contrast, the **match ip precedence** command classifies only IPv4 packets while the **match precedence** command classifies both IPv4 and IPv6.

set precedence tunnel

To set the IP precedence value in a tunnel header that has not yet been added to a packet, use the **set precedence tunnel** command in policy-map class configuration mode. To disable this setting, use the **no** form of this command.

[no] set precedence tunnel *precedence-value*

Syntax Description

precedence-value	Sets the precedence bit in the tunnel header ranging from 0 to 7

Command Default

DSCP (and the precedence portion) are copied from the encapsulated to the newly-imposed header.

Command Modes

policy-map (config-pmap)

Usage Guidelines

On the Cisco ASR Series Aggregation Services Router, the **set precedence tunnel** command is supported for IPv4 only. See Imposition Marking, on page 86 for a table that lists the supported DSCP tunnel marking configurations.

set qos-group

To set the QoS group identifier (ID) for a packet, use the **set qos-group** command in policy-map class configuration mode. To disable this setting, use the **no** form of this command.

[no] set qos-group *group-id*

Syntax Description

group-id	Specifies a QoS group ID ranging from 0 to 99

Command Default

QoS group-id defaults to 0.

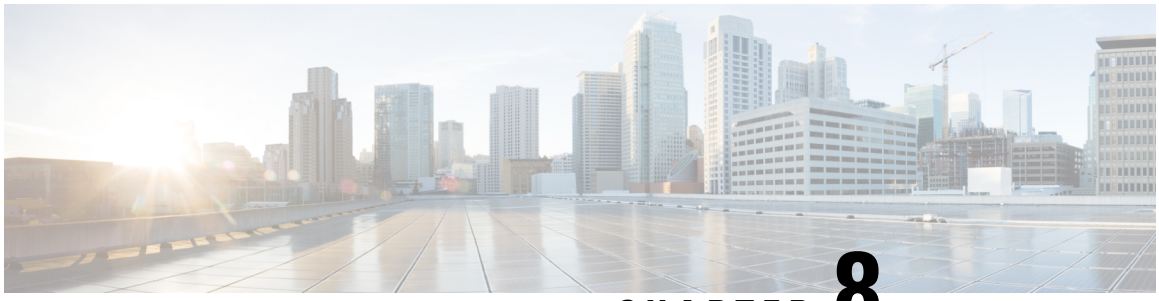
Command Modes

policy-map (config-pmap)

Usage Guidelines

The **set qos-group** command allows you to associate a group ID with a packet as it is processed by the router.

You can use the group ID in egress policies to classify packets to service-classes. Historically, this action had no meaning because we chose the service-class before egress marking occured. With color-aware policing, however, setting the QoS group ID in an egress policy can have meaning.



CHAPTER 8

QoS Packet-Matching Statistics Configuration

The QoS Packet-Matching Statistics feature comprises the following subfeatures:

- The QoS Packet-Matching Statistics: Per Filter feature allows users to count and display the number of packets and bytes matching individual filters (match statements) within a QoS class-map.
- The QoS Packet-Matching Statistics: Per ACE feature allows users to count and display the number of packets and bytes matching the individual access control entries (ACEs) in the filter.

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.

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.

Prerequisites for QoS Packet-Matching Statistics Feature

You cannot enable or disable the QoS Packet-Matching Statistics: Per Filter feature if a policy-map is associated with any interface on the system.

The QoS Packet-Matching Statistics: Per ACE feature is dependent on the QoS Packet Matching Statistics feature. Therefore, the following prerequisites apply:

- If the QoS Packet-Matching Statistics: Per Filter is not enabled and a user tries to enable the QoS Packet-Matching Statistics: Per ACE feature, the command to enable this feature will be rejected by the CLI. An informational message will be displayed to let the user know why the command was rejected.

- If the QoS Packet-Matching Statistics: Per ACE feature is enabled and a user tries to disable this feature, the command to disable this feature will be rejected by the CLI. An informational message will be displayed to let the user know why the command was rejected.

Restrictions for QoS Packet-Matching Statistics Feature

Enabling the Qos: Packet Matching Statistics feature may increase CPU utilization on a scaled configuration. Before enabling the Qos: Packet Matching Statistics feature, weigh the benefits of the statistics information against the increased CPU utilization for your system.

This section provides information about the restrictions pertaining to the QoS Packet-Matching Statistics: Per Filter feature and the QoS Packet-Matching Statistics: Per ACE feature.

The followings are the restrictions for the QoS Packet Matching Statistics feature:

- Enabling the QoS Packet-Matching Statistics: Per Filter feature may increase CPU utilization on a scaled configuration. Before enabling the QoS Packet-Matching Statistics: Per Filter feature, weigh the benefits of the statistics information against the increase in CPU utilization for your system.

- QoS Packet-Matching Statistics: Per Filter is not supported for the match-all class-maps. However, QoS Packet-Matching Statistics: Per ACE is supported for the match-all class-maps.

The following table provides information about the QoS Packet-Matching Statistics: Per ACE scaling limitations:

Table 14: QoS Packet-Matching Statistics: Per ACE Scaling Limitations

Platform	ACEs (IPv4 or IPv6)
ASR1000-ESP5, ASR1001, ASR1002-F, ASR1002-X	25,000
ASR1000-ESP10	30,000
ASR1000-ESP20/ESP40/ESP100	30,000
ISR4400	20,000
CSR1000V	1,000

Information About QoS Packet-Matching Statistics

This section provides an overview of the QoS Packet-Matching Statistics: Per Filter feature and the QoS Packet-Matching Statistics: Per ACE feature.

QoS Packet-Matching Statistics: Per Filter Feature Overview

The QoS Packet-Matching Statistics: Per Filter feature allows you to count and display the number of packets and bytes matching a filter.

To define a filter, use the **class-map** command with the **match-any** keyword, for example:

```
class-map match-any my_class
   match ip precedence 4 <-------- User-defined filter
   match qos-group 10 <-------- User-defined filter
```

Using this information, you can perform the following tasks:

- Compare the amount of voice traffic with the amount of data traffic on a segment of your network
- Adjust bandwidth availability
- Accurately determine billing
- Troubleshoot service problems

The system collects packet matching statistics in 10-second cycles. If there are many interfaces or sessions, the system collects statistics for about 8000 of them during each cycle. In a scaled configuration, several 10-second cycles may be required to gather all the statistics.

QoS Packet-Matching Statistics: Per ACE Feature Overview

The QoS Packet-Matching Statistics: Per ACE feature allows you to track and display the number of packets and bytes matching individual ACEs that are used in QoS policies (access groups used in class maps).

This feature provides hit counters for ACEs used in QoS policies. When this feature is enabled, it will add QoS hit counters for the ACEs used in a QoS policy to the existing security access list counters for that particular ACE. The access list counters can be seen in the following command output:

```
Router# show ip access-lists

Extended IP access list A1
    10 permit ip 32.1.1.0 0.0.0.255 any (129580275 matches)
Extended IP access list A6and7
    10 permit ip 32.1.6.0 0.0.0.255 any (341426749 matches)
    20 permit ip 32.1.7.0 0.0.0.255 any (398245767 matches)
Extended IP access list source
    10 permit ip any host 16.1.1.5 (16147976 matches)
```

The QoS hit counters (for the ACEs used in QoS policies) will be added to the access list counters. We recommend that you pay attention to the following points when you enable this feature:

- Access list counts are not interface specific, as can be seen in the output of the **show ip access-lists** command (there is no mention of interface). They are aggregate counters of all the hits, for all the features that use the ACEs and support the counts, across all interfaces and directions.
- Interface-specific counts are provided in the existing QoS command(**show policy-map interface**) if the QoS Packet-Matching Statistics: Per Filter feature is enabled. However, the command specified previously shows only the counts per filter (ACL or access group), not per ACE, as can be seen in the following sample output:

```
Router# show access-lists

Extended IP access list A1
    10 permit ip 32.1.1.0 0.0.0.255 any (2000 matches)

Router# show policy-map interface GigabitEthernet0/0/2

  Service-policy input: simple

    Class-map: A1-class (match-all)
      1000 packets, 124000 bytes
      5 minute offered rate 4000 bps
      Match: access-group name A1

    Class-map: class-default (match-any)
      0 packets, 0 bytes
      5 minute offered rate 261000 bps, drop rate 0 bps
      Match: any
```

- If an ACE is present in a QoS filter (match statement within a class map), but the packet does *not* match the statement, the ACE counter will *not* be incremented for that packet. This can happen if:
  - The ACE is used in a deny statement.
  - Other matching criteria in a match-all class map definition (such as match ip prec 1) prevent the packet from matching the class.
  - Other matching criteria in a match-any class map definition (such as match ip prec 1) match the packet and keep it from matching the ACE match criteria. (This filter precedes the ACE filter and the packet matches both the statements).
- Access list counts are an aggregate (for a particular ACE) of the hit counts for all the features using that ACE, and support the per ACE counts. (In Cisco IOS XE3.10, only Security and QoS ACLs support per ACE counts, but that may change in future releases). Therefore, it is possible that a single packet will hit (and be counted by) multiple features using the same ACE and hence result in multiple counts for the same packet (as it traverses each feature). The following is an example of this:

```
ip access-list extended A1
  permit ip 32.1.1.0 0.0.0.255 any
class-map match-all A1-class
  match access-group name A1

interface GigabitEthernet0/0/2
 ip address 32.0.0.1 240.0.0.0
 ip access-group A1 in
 duplex auto
 speed auto
 media-type rj45
 no negotiation auto
 service-policy input simple


Router# show access-lists

Extended IP access list A1
    10 permit ip 32.1.1.0 0.0.0.255 any (2000 matches)

Router# show policy-map interface GigabitEthernet0/0/2

  Service-policy input: simple
```

```
Class-map: A1-class (match-all)
  1000 packets, 124000 bytes
  5 minute offered rate 4000 bps
  Match: access-group name A1

Class-map: class-default (match-any)
  0 packets, 0 bytes
  5 minute offered rate 261000 bps, drop rate 0 bps
  Match: any
```

How to Configure QoS Packet-Matching Statistics

This section provides information about how to configure QoS Packet-Matching Statistics.

Configuring QoS Packet-Matching Statistics: Per Filter

Before you begin

- Before enabling the QoS Packet-Matching Statistics: Per Filter feature, ensure that no policy-maps are associated with the interfaces on the system. If they are, the system returns the following message:

```
Either a) A system RELOAD or
       b) Remove all service-policies, re-apply the change
            to the statistics, re-apply all service-policies
   is required before this command will be activated.
```

- Before enabling the QoS Packet-Matching Statistics: Per Filter feature, ensure that you have defined a filter that is using the **class-map** command with the **match-any** keyword.

Note Enabling the QoS Packet-Matching Statistics: Per Filter feature may increase CPU utilization on a scaled configuration. Before enabling the QoS Packet-Matching Statistics: Per Filter feature, weigh the benefits of the statistics information against an increase in CPU utilization for your system.

To configure the QoS Packet-Matching Statistics: Per Filter feature, perform the following procedure:

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **platform qos match-statistics per-filter**
4. **interface** *interface -name*
5. **service-policy** {**input** | **output**} *policy-map-name*
6. **end**
7. **show policy-map interface** *interface-name*
8. **configure terminal**
9. **interface** *interface-name*

10. **no service-policy** {**input** | **output**} *policy-map-name*
11. **exit**
12. **no platform qos match-statistics per-filter**
13. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables the privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters the global configuration mode.
Step 3	**platform qos match-statistics per-filter** **Example:** `Router(config)# platform qos match-statistics per-filter`	Enables the QoS Packet-Matching Statistics: Per Filter feature.
Step 4	**interface** *interface -name* **Example:** `Router(config)# interface GigabitEthernet0/0/0`	Specifies the interface for attaching the policy-map.
Step 5	**service-policy** {**input** | **output**} *policy-map-name* **Example:** `Router(config-if)# service-policy input pol1`	Attaches a QoS policy-map to the interface. The QoS Packet Matching Statistics feature should be enabled before attaching any QoS policies.
Step 6	**end** **Example:** `Router# end`	Exits the configuration mode.
Step 7	**show policy-map interface** *interface-name* **Example:** `Router# show policy-map interface serial4/0/0`	Displays the packet statistics of all the classes that are configured for all the service policies that are present on the specified interface, subinterface, or a specific permanent virtual circuit (PVC) on the interface.
Step 8	**configure terminal** **Example:** `Router# configure terminal`	Enters the global configuration mode.

	Command or Action	Purpose
Step 9	**interface** *interface-name* **Example:** `Router(config)# interface GigabitEthernet0/0/0`	Specifies the interface for removing the policy-map.
Step 10	**no service-policy** {**input** | **output**} *policy-map-name* **Example:** `Router(config-if)# no service-policy input pol1`	Removes a QoS policy-map from an interface. All the QoS policies should be removed from the interfaces before the QoS Packet Matching Statistics feature can be disabled.
Step 11	**exit** **Example:** `Router(config-if)# exit`	Exits the interface configuration mode.
Step 12	**no platform qos match-statistics per-filter** **Example:** `Router(config)# no platform qos match-statistics per-filter`	Disables the QoS Packet-Matching Statistics: Per Filter feature.
Step 13	**end** **Example:** `Router# end`	Exits the configuration mode.

Examples

Use the **show policy-map interface** command to display the packet statistics of all the classes that are configured for all the service policies that are present on the specified interface, subinterface, or a specific PVC on the interface:

```
Router# show policy-map interface gig1/1/0

 GigabitEthernet1/1/0
  Service-policy input: pol1     ! target = gig1/1/0,input
    Class-map: class1 (match-any)
      1000 packets, 40000 bytes
      5 minute offered rate 0000 bps, drop rate 0000 bps
      Match: ip precedence 1 <-------- User-defined filter
        800 packets, 32000 bytes <-------- Filter matching results
      Match: ip precedence 2 <-------- User-defined filter
        200 packets, 8000 bytes <-------- Filter matching results
      QoS Set
        ip precedence 7
          No packet marking statistics available
    Class-map: class-default (match-any)
      500 packets, 20000 bytes
      5 minute offered rate 0000 bps, drop rate 0000 bps
      Match: any <-------- User-defined filter
        500 packets, 20000 bytes <-------- Filter matching results
```

Configuring QoS Packet-Matching Statistics: Per ACE

Before you begin

Before enabling the QoS Packet-Matching Statistics: Per ACE feature, ensure that the QoS Packet-Matching Statistics: Per Filter feature has been enabled.

The following example shows how to check the feature status by using the **show platform hardware qfp active feature qos configuration global** command:

```
Router# show platform hardware qfp active feature qos configuration global
Marker statistics are: disabled
Match per-filter statistics are: enabled <<<<<<<
Match per-ace statistics are: enabled <<<<<<
Performance-Monitor statistics are: disabled
```

To configure the QoS Packet-Matching Statistics: Per ACE feature, perform the following procedure:

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **platform qos match-statistics per-filter**
4. **platform qos match-statistics per-ace**
5. **interface** *interface-name*
6. **service-policy**{**input**|**output**}*policy-map-name*
7. **end**
8. **show policy-map interface** *interface-name*
9. **show access-lists**
10. **configure terminal**
11. **interface** *interface-name*
12. **no service-policy**{**input**|**output**}*policy-map-name*
13. **exit**
14. **no platform qos match-stat per-ace**
15. **no platform qos match-statistics per-filter**
16. **end**

DETAILED STEPS

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

	Command or Action	Purpose
Step 3	**platform qos match-statistics per-filter** **Example:** `Router(config)# platform qos match-statistics per-filter`	Enables the QoS Packet-Matching Statistics: Per Filter feature.
Step 4	**platform qos match-statistics per-ace** **Example:** `Router(config)# platform qos match-statistics per-ace`	Enables the QoS Packet-Matching Statistics: Per ACE feature.
Step 5	**interface** *interface-name* **Example:** `Router(config)# interface GigabitEthernet0/0/0`	Specifies the interface for attaching the policy-map.
Step 6	**service-policy**{**input**|**output**}*policy-map-name* **Example:** `Router(config-if)# service-policy input pol1`	Attaches a QoS policy-map to an interface. The QoS Matching Statistics feature should be enabled before attaching QoS policies.
Step 7	**end** **Example:** `Router# end`	Exits the configuration mode.
Step 8	**show policy-map interface** *interface-name* **Example:** `Router# show policy-map interface serial4/0/0`	Displays the packet statistics pertaining to all the classes that are configured for all the service policies either on the specified interface, subinterface, or on a specific PVC on the interface.
Step 9	**show access-lists** **Example:** `Router# show access-lists`	Displays the contents of current access lists, including the QoS Packet-Matching Statistics: Per ACE.
Step 10	**configure terminal** **Example:** `Router# configure terminal`	Enters the global configuration mode.
Step 11	**interface** *interface-name* **Example:** `Router(config)# interface GigabitEthernet0/0/0`	Specifies the interface for removing the policy-map.
Step 12	**no service-policy**{**input**|**output**}*policy-map-name* **Example:** `Router(config-if)# no service-policy input pol1`	Removes a QoS policy-map from an interface. All the QoS policies should be removed from the interfaces before the QoS Matching Statistics feature can be disabled.

	Command or Action	Purpose
Step 13	**exit** **Example:** `Router(config-if)# exit`	Exits the interface configuration mode.
Step 14	**no platform qos match-stat per-ace** **Example:** `Router(config)# no platform qos match-stat per-ace`	Disables the QoS Packet-Matching Statistics: Per ACE feature.
Step 15	**no platform qos match-statistics per-filter** **Example:** `Router(config)# no platform qos match-statistics per-filter`	Disables the QoS Packet-Matching Statistics: Per Filter feature.
Step 16	**end** **Example:** `Router# end`	Exits the configuration mode.

Example

Use the **show policy-map interface** command to display the per-filter statistics of all the classes that are configured for all the service policies on the specified interface, subinterface, or on a specific PVC on the interface:

```
Router# show policy-map interface GigabitEthernet0/0/2


  Service-policy input: test-match-types

    Class-map: A1orA2-class (match-any)
      482103366 packets, 59780817384 bytes
      5 minute offered rate 6702000 bps
      Match: access-group name A1
        62125633 packets, 7703578368 bytes
        5 minute rate 837000 bps
      Match: access-group name A2
        419977732 packets, 52077238892 bytes
        5 minute rate 5865000 bps

    Class-map: A3andprec1-class (match-all)
      5673520 packets, 703516480 bytes
      5 minute offered rate 837000 bps
      Match: access-group name A3
      Match: ip precedence 1

    Class-map: A5-class (match-all)
      227101820 packets, 28160625680 bytes
      5 minute offered rate 3351000 bps
      Match: access-group name A5
```

```
    Class-map: A6and7-class (match-all)
      627615840 packets, 77824340228 bytes
      5 minute offered rate 9215000 bps
      Match: access-group name A6and7

    Class-map: A3-class (match-all)
      111548288 packets, 13831987712 bytes
      5 minute offered rate 1675000 bps
      Match: access-group name A3

    Class-map: A4andsource (match-all)
      16115590 packets, 1998333160 bytes
      5 minute offered rate 2513000 bps
      Match: access-group name A4
      Match: access-group name source

    Class-map: class-default (match-any)
      164881212 packets, 20445270288 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: any
```

Use the **show ip access-lists** command to display the contents of current access lists (which includes the QoS Packet-Matching Statistics: Per ACE):

```
Router# show ip access-lists

Extended IP access list A1
    10 permit ip 32.1.1.0 0.0.0.255 any (129580275 matches)
Extended IP access list A2
    10 permit ip 32.1.2.0 0.0.0.255 any (486342300 matches)
Extended IP access list A3
    10 permit ip 32.1.3.0 0.0.0.255 any (306738457 matches)
Extended IP access list A4
    10 permit ip 32.1.4.0 0.0.0.255 any (16147975 matches)
Extended IP access list A5
    10 permit ip 32.1.5.0 0.0.0.255 any (294357455 matches)
Extended IP access list A6and7
    10 permit ip 32.1.6.0 0.0.0.255 any (341426749 matches)
    20 permit ip 32.1.7.0 0.0.0.255 any (398245767 matches)
Extended IP access list source
    10 permit ip any host 16.1.1.5 (16147976 matches)
```

Troubleshooting Tips

To confirm that the QoS: Packet Matching Statistics feature is enabled, use the **show platform hardware qfp active feature qos config global** command. If the feature is disabled, you should see a message similar to the following:

Router# **show platform hardware qfp active feature qos config global**

```
Marker statistics are: enabled
Match per filter statistics are: enabled
```

Example: Configuring a QoS Packet-Matching Statistics: Per Filter

The following example shows how to configure a QoS Packet-Matching Statistics: Per Filter, perform the following tasks:

- Define a QoS packet matching filter
- Display the **show policy-map interface** command output

```
Router# show policy-map interface Tunnel1

  Service-policy output: DATA-OUT-PARENT
    Class-map: class-default (match-any)
      4469 packets, 4495814 bytes
      5 minute offered rate 0000 bps, drop rate 0000 bps
      Match: any <-------- User-defined filter
      Queueing
      queue limit 416 packets
      (queue depth/total drops/no-buffer drops) 0/0/0
      (pkts output/bytes output) 4469/4558380
      shape (average) cir 100000000, bc 400000, be 400000
      target shape rate 100000000
      Service-policy : DATA-OUT
        queue stats for all priority classes:
          Queueing
          queue limit 200 packets
          (queue depth/total drops/no-buffer drops) 0/0/0
          (pkts output/bytes output) 4469/4558380
        Class-map: ATM-VTI-RIP-SPK1-DATA (match-any)
          4469 packets, 4495814 bytes <-------- Filter matching results
          5 minute offered rate 0000 bps, drop rate 0000 bps
          Match: access-group 121 <-------- User-defined filter
            4469 packets, 4495814 bytes <-------- Filter matching results
            5 minute rate 0 bps
          QoS Set
            ip precedence 3
              Packets marked 4469
          Priority: 100 kbps, burst bytes 2500, b/w exceed drops: 0
```

Additional References

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Commands List, All Releases
Quality of service commands	*Cisco IOS Quality of Service Command Reference*

Standards

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

MIBs

MIB	MIBs Link
CISCO-CLASS-BASED-QOS-MIB	To locate and download MIBs for selected platforms, Cisco software releases, and feature sets, use Cisco MIB Locator found at the following URL: 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

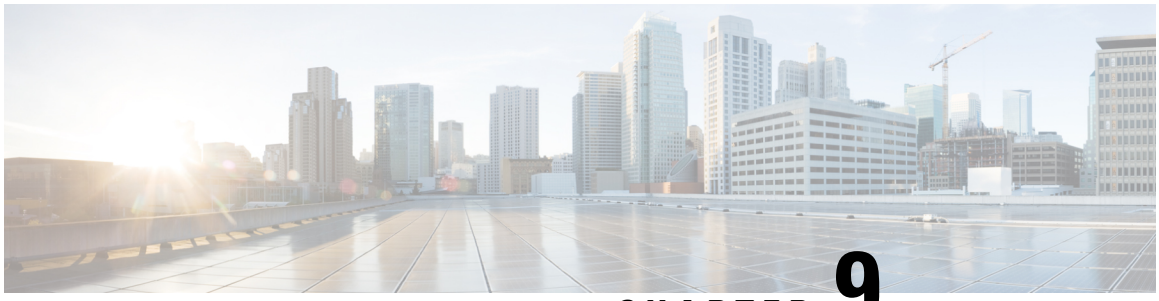
Feature Information for QoS Packet-Matching Statistics

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 15: Feature Information for QoS Packet-Matching Statistics

Feature Name	Releases	Feature Information
QoS Packet-Matching Statistics: Per Filter	Cisco IOS XE Release 3.3S	The QoS Packet-Matching Statistics: Per Filter feature allows you to count and display the number of packets matching individual filters (match statements) used in class-maps within QoS service policies that have. The following commands were introduced or modified: • **platform qos match-statistics per-filter** • **no platform qos match-statistics per-filter** • **show platform hardware qfp active feature qos config global**
QoS Packet-Matching Statistics: Per ACE	Cisco IOS XE Release 3.10S	The QoS Packet-Matching Statistics: Per ACE feature allows you to track and display the number of packets and bytes matching individual ACEs that are used in QoS policies (access groups used in class maps). The following command was introduced: **platform qos match-statistics per-ace**



CHAPTER 9

Set ATM CLP Bit Using Policer

The Set ATM CLP Bit Using Policer feature allows you to police and then mark outbound PPP over ATM (PPPoA) traffic. You can set the ATM cell loss priority (CLP) bit using either of the following methods:

- A policed threshold
- Matching a 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.

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.

Prerequisites for Set ATM CLP Bit Using Policer

If you are setting the ATM CLP bit by a policed threshold, ensure that a policy-map includes the **set-clp-transmit**action. The new policer action conditionally marks PPPoA traffic in the matched class for a higher drop probability in the ATM network when traffic exceeds a given rate.

If you are setting the ATM CLP bit strictly by matching a class, ensure that a policy-map includes the **set atm-clp** action. The set directive marks all traffic in the matched class for higher drop probability in the ATM network.

You can attach policy-maps with the **set-clp-transmit**or **set atm-clp**actions to a virtual template. This template is cloned when PPPoA sessions are created or by dynamic assignment.

Information About Set ATM CLP Bit Using Policer

ATM CLP Bit

The ATM CLP bit shows the drop priority of the ATM cell. During ATM network congestion, the router discards ATM cells with the CLP bit set to 1 before discarding cells with a CLP bit setting of 0.

Using the Set ATM CLP Bit Using Policer feature, you can configure the **police** command to enable the ATM CLP bit in cell headers. The ATM CLP bit can be explicitly marked by a set directive.

The Set ATM CLP Bit Using Policer feature supports the **set-clp-transmit** policing action in the following types of policies:

- Single-rate policing
- Dual-rate policing
- Hierarchical

How to Set the ATM CLP Bit Using Policer

Configuring PPPoA Broadband Traffic Policing

Before you begin

Before configuring the policy-map, ensure that you have defined any class maps used to classify traffic.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **policy-map** *policy-map-name*
4. **class** {*class-name*| **class-default**]
5. **police** [**cir** *cir*] [**conform-action** *action*] [**exceed-action** *action*]
6. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**policy-map** *policy-map-name* **Example:** `Device(config)# policy-map parent-policy`	Enters policy-map configuration mode and creates a policy-map.
Step 4	**class** {*class-name*| **class-default**] **Example:** `Device(config-pmap)# class class-default`	Enters policy-map class configuration mode. Specifies the name of the class whose policy you want to create or change or specifies the default class (commonly known as the class-default class) before you configure its policy. Repeat this command as many times as necessary to specify the child or parent classes that you are creating or modifying: • *class name* --Name of the class to be configured or whose policy is to be modified. The class name is used for both the class map and to configure a policy for the class in the policy-map. • **class-default** --Specifies the default class so that you can configure or modify its policy.

Step 5	**police** [**cir** *cir*] [**conform-action** *action*] [**exceed-action** *action*]
	Example:
	`Device(config-pmap-c)# police 1000000`
	Example:
	`Router(config-pmap-c-police)# conform-action`
	Example:
	`transmit`
	Example:
	`Device(config-pmap-c-police)# exceed-action`
	Example:
	`set-clp-transmit`
		Configures traffic policing and specifies multiple actions applied to packets marked as conforming to, exceeding, or violating a specific rate.
		• Enters policy-map class police configuration mode. Use one line per action that you want to specify:
		  • **cir**--(Optional) Committed information rate. Indicates that the CIR will be used for policing traffic.
		  • **conform-action**--(Optional) Action to take on packets when the rate is less than the conform burst.
		  • **exceed-action**--(Optional) Action to take on packets whose rate is within the conform and conform plus exceed burst.

Step 6	**end** **Example:**	(Optional) Returns to privileged EXEC mode.

	Command or Action	Purpose
	`Device(config-pmap-c)# end`	

Example

The following example shows you how to set the ATM CLP using a policer:

```
policy-map egress_atm_clp_policer
 class prec0
  police cir 5000000
 class prec1
  police cir 3000000 conform-action transmit exceed-action set-clp-transmit
 class class-default
  police cir 1000000 conform-action transmit exceed-action set-clp-transmit
```

Marking the ATM CLP Bit

Before you begin

Before configuring the policy-map, ensure that you have defined any class maps used to classify traffic.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **policy-map policy-map-name**
4. **class** {*class-name*| **class-default**]
5. **set atm-clp**
6. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**policy-map policy-map-name** **Example:** `Router(config)# policy-map parent-policy`	Enters policy-map configuration mode and creates a policy-map.

	Command or Action	Purpose
Step 4	**class** {*class-name*| **class-default**] **Example:** `Router(config-pmap)# class class-default`	Enters policy-map class configuration mode. Specifies the name of the class whose policy you want to create or change or specifies the default class (commonly known as the class-default class) before you configure its policy. Repeat this command as many times as necessary to specify the child or parent classes that you are creating or modifying: • *class name* --Name of the class to be configured or whose policy is to be modified. The class name is used for both the class map and to configure a policy for the class in the policy-map. • **class-default** --Specifies the default class so that you can configure or modify its policy.
Step 5	**set atm-clp** **Example:** `Router(config-pmap-c)# set atm-clp`	Configures marking of the ATM CLP bit for all traffic matching this class.
Step 6	**end** **Example:** `Router(config-pmap-c)# end`	(Optional) Returns to privileged EXEC mode.

Example

The following example shows you how to set the ATM CLP using explicit marking:

```
policy-map egress_atm_clp_policer
 class prec0
  police cir 5000000
 class class-default
  set atm-clp
```

Configuration Examples for Set ATM CLP Bit Using Policer

Example Marking the ATM CLP by Policer Action Matching a Class

This example shows how to do the following:

- Define traffic classes.
- Configure a two-layer policy-map.
- Apply the policy-map to PPPoA sessions.

This policy conditionally marks the ATM CLP bit on the traffic in the matching low_interest class once traffic on the class exceeds a given rate.

```
class-map voice
  match precedence 4
!
class-map web
  match precedence 3
!
class low_interest
  match precedence 1 0
!
policy-map child
 child class voice
  police cir 256000
  priority level 1
 class web
  bandwidth remaining ratio 10
 class low_interest
  police cir 1000000 conform-action transmit exceed-action set-clp-transmit
 class class-default
  bandwidth remaining ratio 1
!
policy-map parent
 class class-default
  shape average 15000000
  service-policy child
```

Policy-maps attached to virtual templates are cloned and used to create a virtual access interface for each PPPoA session:

```
interface Virtual-Template1
 ip unnumbered Loopback1
 load-interval 30
 peer default ip address pool POOL1
 ppp authentication chap ppp
 ipcp address required
 service-policy output parent
```

Example Marking the ATM CLP by Policer Action Policed Threshold

This example shows how to do the following:

- Define traffic classes.
- Configure a two-layer policy-map.
- Apply the policy-map to PPPoA sessions.

This policy marks all non-essential traffic with the ATM CLP bit so that it is eligible for dropping if the ATM network becomes congested.

```
class-map video
  match precedence 5
!
class-map voice
  match precedence 4
!
class-map web
```

```
  match precedence 3
!
policy-map child
 child class voice
  police cir 256000
  priority level 1
 class video
  police cir 4000000
  priority level 2
 class web
  set atm-clp
  bandwidth remaining ratio 10
 class class-default
  bandwidth remaining ratio 1
  set atm-clp
!
interface Virtual-Template1
 ip unnumbered Loopback1
 load-interval 30
 peer default ip address pool POOL1
 ppp authentication chap ppp
 ipcp address required
 service-policy output parent
```

Additional References

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Commands List, All Releases
Quality of Service commands	*Cisco IOS Quality of Service Command Reference*

Standards

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

MIBs

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

RFCs

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

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

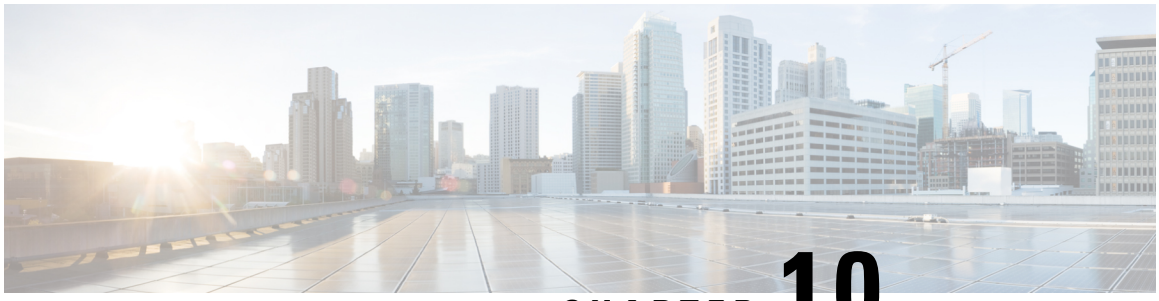
Feature Information for Set ATM CLP Bit Using Policer

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 16: Feature Information for Set ATM CLP Bit Using Policer

Feature Name	Releases	Feature Information
Set ATM CLP Bit Using Policer	Cisco IOS Release XE 3.3S Cisco IOS Release XE 3.14S	The Set ATM CLP Bit Using Policer feature allows you to police and then mark outbound PPPoA traffic. In Cisco IOS Release XE 3.14S, support for this feature was added on the Cisco 4451-X Integrated Services Router. The following commands were introduced or modified: **set atm-clp**and **police**.



CHAPTER 10

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.

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 131, 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 17: 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. **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. **Note** Classifying traffic using the **match source-address mac**command is supported in the input direction only.
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 18: 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.
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.

Command	Purpose
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.
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 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** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**class-map** [**match-all** | **match-any**] *class-name* **Example:** `Router(config)# class-map match-any class1`	Creates a class map and enters class-map configuration mode. • The class map is used for matching packets to the specified class. **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* **Example:**	Matches a packet on the basis of a Layer 2 CoS number. **Note** The **match cos** command is an example of a **match** command you can use.

	Command or Action	Purpose
	`Router(config-cmap)# match cos 2`	
Step 5	Enter additional **match** commands, if applicable; otherwise, proceed with the next step.	--
Step 6	**end** **Example:** `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** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.

Step 3	**policy-map** *policy-map-name*
	Example:
	`Router(config)#`
	`policy-map policy1`
		Creates or specifies the name of the traffic policy and enters QoS policy-map configuration mode.

	Command or Action	Purpose

Step 4	**class** {*class-name*| **class-default**}
	Example:
	`Router(config-pmap)#`
	`class class1`
		Specifies the name of a class and enters QoS policy-map class configuration mode.
		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*] **Example:** `Router(config-pmap-c)# police 3000`	(Optional) Configures traffic policing. **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** **Example:** `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 10. 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** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-type interface-number* **Example:** `Router(config)# interface gigabitethernet 0/0/1`	Configures an interface type and enters interface configuration mode. • Enter the interface type and interface number.
Step 4	**service instance** *id* **ethernet** [*evc-name*] **Example:** `Router(config-if)# service instance 333 ethernet evc1`	Configures an Ethernet service instance on an interface and enters Ethernet service configuration mode. • Enter the service instance identification number and, if applicable, the EVC name (optional).
Step 5	**encapsulation dot1q** *vlan-id* [,*vlan-id*[-*vlan-id*]] [**native**] **Example:** `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** **Example:** `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* **Example:** `Router(config-if-srv)# bridge domain 1`	Configures a bridge domain. • Enter the bridge domain number.
Step 8	**service-policy** {**input**	**output**} *policy-map-name* **Example:** `Router(config-if-srv)#`	Attaches a policy-map to an interface. • Enter either the **input** or **output** keyword and the policy-map name.

	Command or Action	Purpose
	`service-policy input policy1`	
Step 9	**end** **Example:** `Router(config-if-srv)# end`	(Optional) Returns to privileged EXEC mode.
Step 10	**show policy-map interface** *type number* **service instance** *service-instance-number* **Example:** `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. • 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: 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

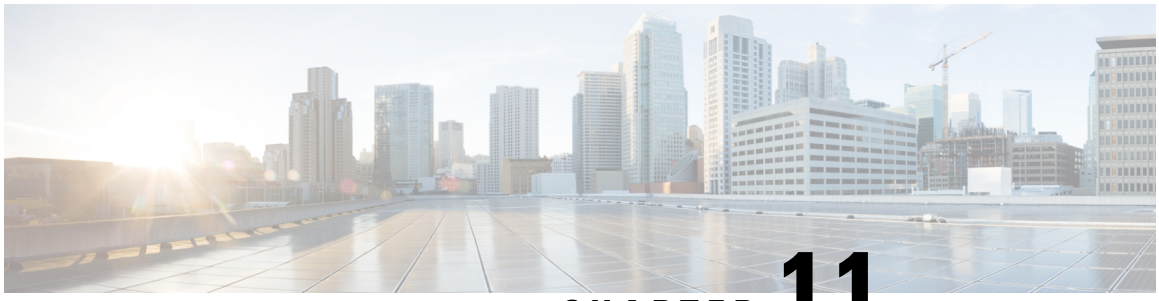
Feature Information for Configuring EVC Quality of Service

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 19: Feature Information for EVC Quality of Service

Feature Name	Releases	Feature Information
EVC Quality of Service	Cisco IOS XE Release 3.3 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). The EVC Quality of Service feature was introduced on the Cisco ASR 1000 Series Aggregation Services Router. The following commands were introduced or modified: **service-policy**, **show policy-map interface service instance**.



CHAPTER 11

Quality of Service for Etherchannel Interfaces

Quality of Service (QoS) is supported on Ethernet Channel (Etherchannel) interfaces on Cisco ASR 1000 Series Routers. The QoS functionality has evolved over several Cisco IOS XE releases and has different capabilities based on software level, Etherchannel configuration, and configured Modular QoS CLI (MQC) features.

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.

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 QoS for Etherchannels

Etherchannel with QoS Feature Evolution

An Etherchannel is a port-channel architecture that allows grouping of several physical links to create one logical Ethernet link for the purpose of providing fault tolerance, and high-speed links between switches, routers, and servers. An Etherchannel can be created from between two and eight active Fast, Gigabit, or 10-Gigabit Ethernet ports, with an additional one to eight inactive (failover) ports, which become active as the other active ports fail.

QoS for Etherchannel interfaces has evolved over several Cisco IOS XE releases. It is important to understand what level of support is allowed for your current level of Cisco IOS XE software and underlying Etherchannel

configuration. Various combinations of QoS are supported based on how Etherchannel is configured. There are three different modes in which Etherchannel can be configured:

- Etherchannel VLAN-based load balancing via port-channel subinterface encapsulation CLI
- Etherchannel Active/Standby with LACP (no Etherchannel load balancing)
- Etherchannel with LACP with load balancing

Each of these models has specific restrictions regarding which levels of Cisco IOS XE software include support and the possible QoS configurations with each.

The following summarizes the various Etherchannel and QoS configuration combinations that are supported. Example configurations will be provided later in this document. Unless specifically mentioned together, the combination of service policies in different logical and physical interfaces for a given Etherchannel configuration is not supported.

Etherchannel VLAN-Based Load Balancing via Port-Channel Subinterface Encapsulation CLI

Supported in Cisco IOS XE Release 2.1 or later:

- Egress MQC Queuing Configuration on Port-Channel Subinterface
- Egress MQC Queuing Configuration on Port-Channel Member Link
- QoS Policies Aggregation—Egress MQC Queuing at Subinterface
- Ingress Policing and Marking on Port-Channel Subinterface
- Egress Policing and Marking on Port-Channel Member Link

Supported in Cisco IOS XE Release 2.6 or later:

- QoS Policies Aggregation—MQC Support for Multiple Queue Aggregation at Main Interface - Egress MQC Queuing at Main Interface

Etherchannel Active/Standby with LACP (No Etherchannel Load Balancing)

Supported in Cisco IOS XE 2.4 or later:

- Egress MQC Queuing on Port-Channel Member Link—No Etherchannel Load Balancing

Etherchannel with LACP and Load Balancing

Supported in Cisco IOS XE 2.5 or later:

- Egress MQC Queuing Configuration on Port-Channel Member Link—Etherchannel Load Balancing

Supported in Cisco IOS XE 3.12 or later:

- General MQC QoS support on Port-channel main-interface

  We recommend that as a best practice for QoS, that you use port-channel aggregation—see the "Aggregate EtherChannel Quality of Service" chapter.

Supported in Cisco IOS XE 3.16.3 or later and in Cisco IOS XE Fuji 16.3 or later:

- General MQC QoS support on Port-channel sub-interface

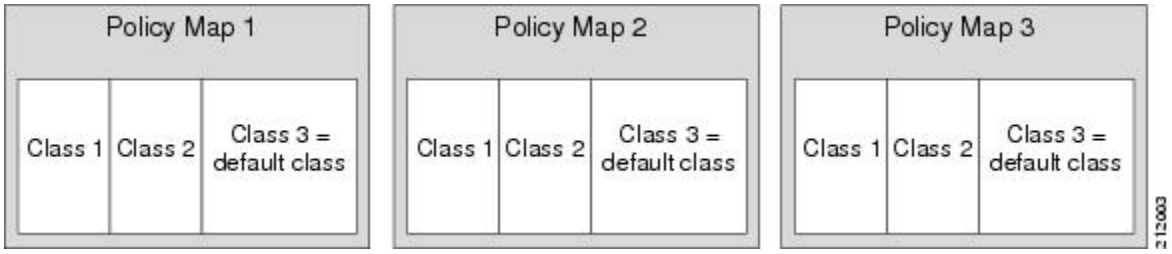
  We recommend that as a best practice for QoS, that you use port-channel aggregation—see the "Aggregate EtherChannel Quality of Service" chapter.

Understanding Fragments in Class Definition Statements

The QoS Policies Aggregation feature introduces the idea of fragments in class definition statements. A default traffic class definition statement can be marked as a fragment within a policy-map. Other policy-maps on the same interface can also define their default traffic class statements as fragments, if desired. A separate policy-map can then be created with a service fragment class definition statement that will be used to apply QoS to all of the fragments as a single group.

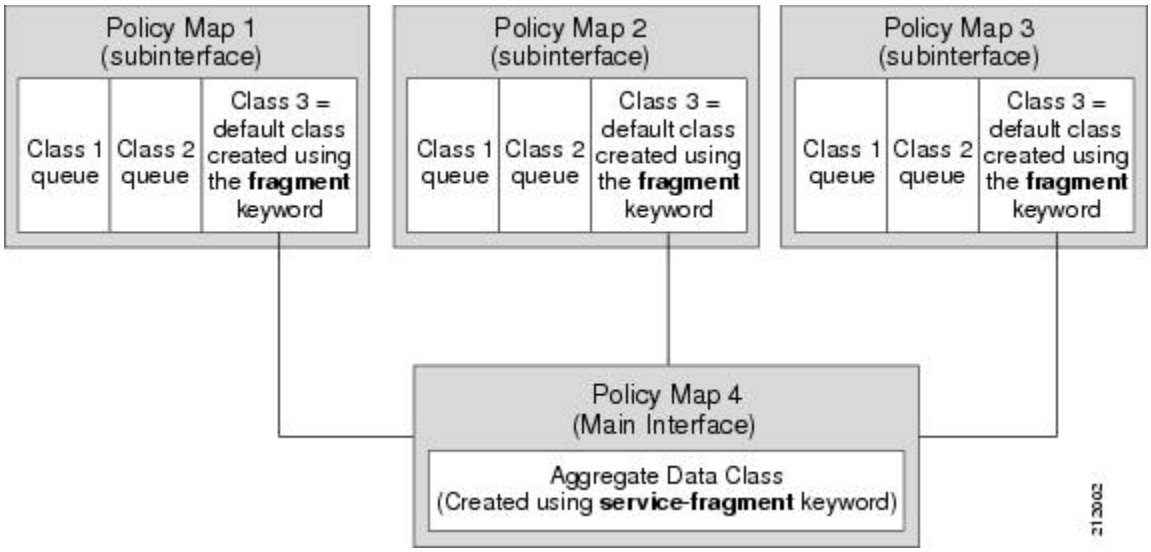
The figure below provides an example of one physical interface with three attached policy-maps that is not using fragments. Note that each policy-map has a default traffic class that can classify traffic only for the default traffic within its own policy-map.

Figure 10: Physical Interface with Policy-Maps—Not Using Fragments



The figure below shows the same configuration configured with fragments, and adds a fourth policy-map with a class definition statement that classifies the fragments collectively. The default traffic classes are now classified as one service fragment group rather than three separate default traffic classes within the individual policy-maps.

Figure 11: Physical Interface with Policy-Maps—Using Fragments



Fragments for Gigabit Etherchannel Bundles

When fragments are configured for Gigabit Etherchannel bundles, the policy-maps that have a default traffic class configured using the **fragment** keyword are attached to the member subinterface links, and the policy-maps

that have a traffic class configured with the **service-fragment** keyword to collectively classify the fragments is attached to the physical interface.

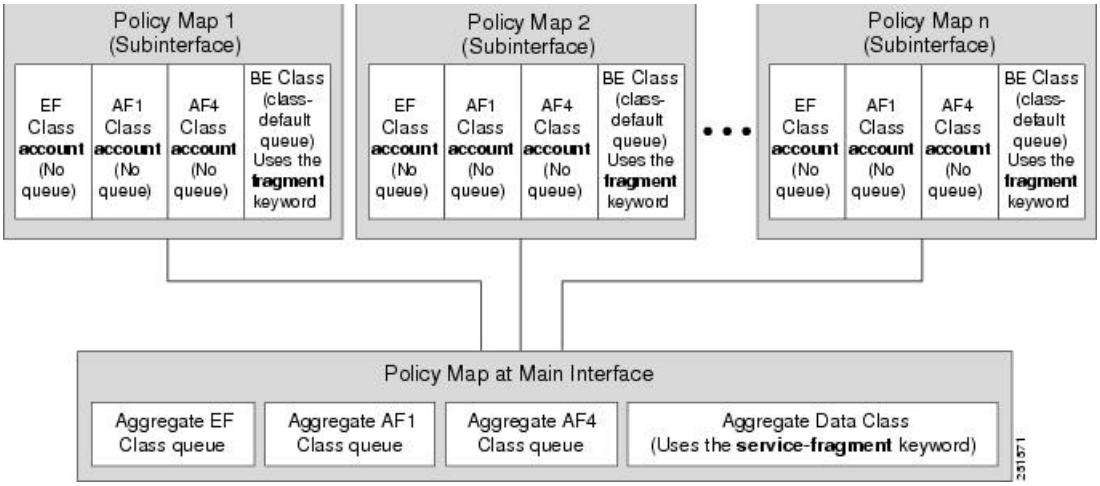
All port-channel subinterfaces configured with fragments that are currently active on a given port-channel member link will use the aggregate service fragment class on that member link. If a member link goes down, the port-channel subinterfaces that must switch to the secondary member link will then use the aggregate service fragment on the new interface.

QoS: Policies Aggregation MQC

The QoS: Policies Aggregation MQC Support for Multiple Queue Aggregation at Main Interface feature extends the previous support of aggregation of class-default traffic using the **fragment** and **service-fragment** configurations, to other user-defined traffic classes in a subinterface policy-map, such as DSCP-based traffic classes, that are aggregated at the main-interface policy-map as shown in the figure below.

When no queueing is configured on a traffic class in the subinterface policy-map, the **account** command can be used to track queueing drops that occur at the aggregate level for these classes, and can be displayed using the **show policy-map interface** command.

Figure 12: Policy-Map Overview for the MQC Support for Multiple Queue Aggregation at Main Interface Feature



Differences Between the Original Feature and the MQC Support for Multiple Queue AggregationDifferences Between Policy Aggregation—Egress MQC Queuing at Subinterface and the MQC Support for Multiple Queue Aggregation at Main Interface

Although some of the configuration between the “Policy Aggregation – Egress MQC Queuing at Subinterface” scenario and the “MQC Support for Multiple Queue Aggregation at Main Interface - Egress MQC Queuing at Main Interface” scenario appear similar, there are some important differences in the queuing behavior and the internal data handling. See the figure in the “Understanding the QoS: Policies Aggregation MQC” section.

For example, both configurations share and require the use of the **fragment** keyword for the **class class-default** command in the subscriber policy-map, as well as configuration of the **service-fragment** keyword for a user-defined class in the main-interface policy-map to achieve common policy treatment for aggregate traffic.

However, the use of this configuration results in different behavior between the original and enhanced QoS policies aggregation implementation:

- In the original implementation using the fragment and service-fragment architecture, all default class traffic and any traffic for classes without defined queueing features at the subinterface goes to the class-default queue and is aggregated into a common user-defined queue and policy defined at the main policy-map. Subinterface traffic aggregation (for example, from multiple subscribers on the same physical interface) ultimately occurs only for a single class, which is the default class.
- In the enhanced implementation of the MQC Support for Multiple Queue Aggregation at Main Interface feature also using the fragment and service-fragment architecture, all default class traffic also goes to the class-default queue and is aggregated into a common user-defined queue and policy defined at the main policy-map. However, other classes, such as DSCP-based subscriber traffic classes, are also supported for an aggregate policy. These traffic classes do not support any queues or queueing features other than **account** at the subscriber policy-map. The use of the fragment and service-fragment architecture enables these other subscriber traffic classes (from multiple subscribers on the same physical interface) to achieve common policy treatment for aggregate traffic that is defined for those same classes at the main policy-map.

How to Configure QoS for Etherchannels

Configuring Egress MQC Queuing on Port-Channel Subinterface

Before you begin

Traffic classes must be configured using the **class-map** command. A one- or two-level hierarchical policy-map should be configured using previously defined class maps. The port-channel subinterface should have been previously configured with the appropriate encapsulation subcommand to match the select primary and secondary physical interfaces on the Etherchannel. Cisco IOS XE Release 2.1 or later software is required. The global configuration must contain the **port-channel load-balancing vlan-manual** command, or the port-channel main-interface configuration must contain the **load-balancing vlan** command. It is assumed that these commands have already been executed.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface port-channel** *port-channel-number* **.** *subinterface-number*
4. **service-policy output** *policy-map-name*
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface port-channel** *port-channel-number . subinterface-number* **Example:** `Device(config)# interface port-channel 1.200`	Specifies the port-channel subinterface that receives the service policy configuration.
Step 4	**service-policy output** *policy-map-name* **Example:** `Device(config-subif)# service-policy output WAN-GEC-sub-Out`	Specifies the name of the service policy that is applied to output traffic.
Step 5	**end** **Example:** `Device(config-subif)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

Configuring Egress MQC queuing on Port-Channel Member Links

Before you begin

Traffic classes must be configured using the **class-map** command. A one- or two-level hierarchical policy-map that uses queuing features should be configured using previously defined class maps. The Etherchannel member link interface should already be configured to be part of the channel group (Etherchannel group). No policy-maps that contain queuing commands should be configured on any port-channel subinterfaces. Cisco IOS XE Release 2.1 or later software is required. The global configuration must contain the **port-channel load-balancing vlan-manual** command, or the port-channel main-interface configuration must contain the **load-balancing vlan** command. It is assumed that these commands have already been executed.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface GigabitEthernet** *card/bay/port*
4. **service-policy output** *policy-map-name*
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable**	Enables privileged EXEC mode.

	Command or Action	Purpose
	Example: `Device> enable`	• Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface GigabitEthernet** *card/bay/port* **Example:** `Device(config)# interface GigabitEthernet 0/1/0`	Specifies the member link physical interface that receives the service policy configuration.

Step 4	**service-policy output** *policy-map-name*
	Example:
	`Device(config-if)#`
	`service-policy output WAN-GEC-sub-Out`
		Specifies the name of the service policy that is applied to output traffic for this physical interface that is part of the Etherchannel.

Step 5	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

Configuring QoS Policies Aggregation—Egress MQC Queuing at Subinterface

Before you begin

Default class traffic from multiple Port-channel subinterfaces can be aggregated into a common policy-map at the main interface when you use the **fragment** keyword at the subinterface **class class-default** configuration, and the **service-fragment** configuration at the main interface class. Queuing occurs at the subinterface for other traffic classes that are defined with queuing features in the subinterface policy-map.

This feature is configured using Modular QoS CLI (MQC). It is most useful in QoS configurations where several policy-maps attached to the same physical interface want aggregated treatment of multiple default traffic classes from multiple port-channel sub-interfaces. Cisco IOS XE Release 2.1 or later software is required. The global configuration must contain the **port-channel load-balancing vlan-manual** command, or the port-channel main-interface must have the **load-balancing vlan** command. It is assumed that these commands have already been executed.

Note This feature is supported when policy-maps are attached to multiple port-channel subinterfaces and the port-channel member link interfaces. This feature cannot be used to collectively classify default traffic classes of policy-maps on different physical interfaces. It can collectively classify all traffic directed toward a given port-channel member link when designated by the **primary** or **secondary** directives on the subinterface **encapsulation** command. All subinterface traffic classes should have queues. However, when a traffic class in the subinterface policy-map is not configured with any queuing feature (commands such as **priority**, **shape**, **bandwidth**, **queue-limit**, **fair-queue**, or **random-detect**), the traffic is assigned to the class-default queue. No classification occurs or is supported at the main interface policy-map for any subinterface traffic classes that do not use the **fragment** and **service-fragment** configuration.

A multistep process is involved with the complete configuration of the QoS Policies Aggregation feature. The following sections detail those steps.

Note the following about attaching and removing a policy-map:

- To configure QoS Policies Aggregation, you must attach the policy-map that contains the **service-fragment** keyword to the main interface first, and then you must attach the policy-map that contains the **fragment** keyword to the subinterface.
- To disable QoS Policies Aggregation, you must remove the policy-map that contains the fragment keyword from the subinterface first, and then you must remove the policy-map that contains the service-fragment keyword from the main interface.

Configuring a Fragment Traffic Class in a Policy-Map

Before you begin

This procedure shows only how to configure the default traffic class as a fragment within a policy-map. It does not include steps on configuring other classes within the policy-map, or other policy-maps on the device.

Example

Note This example shows a sample configuration that is supported in releases prior to Cisco IOS XE Release 2.6.

In the following example, a fragment named BestEffort is created in policy-map subscriber1 and policy-map subscriber 2. In this example, queuing features for other traffic classes are supported at the subinterface policy-map.

```
policy-map subscriber1
 class voice
  set cos 5
  priority level 1
 class video
  set cos 4
  priority level 2
 class class-default fragment BestEffort
  shape average 200000000
  bandwidth remaining ratio 10
```

```
policy-map subscriber 2
 class voice
  set cos 5
  priority level 1
 class video
  set cos 4
  priority level 2
 class class-default fragment BestEffort
  shape average 200000000
  bandwidth remaining ratio 10
```

Note This example shows a sample configuration that is supported in Cisco IOS XE Release 2.6 and later releases.

The following example also shows how to configure a fragment named BestEffort for the default class in a policy-map on a subinterface using the QoS Policies Aggregation MQC Support for Multiple Queue Aggregation at Main Interface implementation. In this example, notice that queuing features are not supported for the other classes in the policy-map:

```
policy-map subscriber1
 class voice
  set cos 5
  account
 class video
  set cos 4
  account
 class AF1
  account
 class class-default fragment BestEffort
  shape average 200000000
  bandwidth remaining ratio 10
```

After configuring default class statements as fragments in multiple subinterface policy-maps, a separate policy-map with a class statement using the service-fragment keyword must be configured to apply QoS to the class stratements configured as fragments.

What to Do Next

After configuring multiple default class statements as fragments in a policy-map, a separate policy-map with a class statement using the **service-fragment** keyword must be configured to apply QoS to the class statements configured as fragments.

This process is documented in the “Configuring a Service Fragment Traffic Class” section.

Configuring a Service Fragment Traffic Class

Before you begin

This task describes how to configure a service fragment traffic class statement within a policy-map. A service fragment traffic class is used to apply QoS to a collection of default class statements that have been configured previously in other policy-maps as fragments.

This procedure assumes that fragment default traffic classes were already created. The procedure for creating fragment default traffic classes is documented in the “Configuring a Fragment Traffic Class in a Policy-Map” section.

Like any policy-map, the configuration does not manage network traffic until it has been attached to an interface. This procedure does not cover the process of attaching a policy-map to an interface.

Note A service fragment can be used to collectively classify fragments only from the same physical interface. Fragments from different interfaces cannot be classified using the same service fragment.

Only queueing features are allowed in classes where the **service-fragment** keyword is entered, and at least one queueing feature must be entered in classes when the **service-fragment** keyword is used.

A policy-map with a class using the **service-fragment** keyword can be applied only to traffic leaving the interface (policy-maps attached to interfaces using the **service-policy output** command).

A class configured using the **service-fragment** keyword cannot be removed when it is being used to collectively apply QoS to fragments that are still configured on the interface. If you wish to remove a class configured using the **service-fragment** keyword, remove the fragment traffic classes before removing the service fragment.

The **service-fragment** keyword cannot be entered in a child policy-map.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **policy-map** *policy-map-name*
4. **class** *class-name* **service-fragment** *fragment-class-name*
5. **shape average percent** *percent*
6. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**policy-map** *policy-map-name* **Example:** `Device(config)# policy-map BestEffortFragments`	Specifies the name of the traffic policy to configure and enters policy-map configuration mode.
Step 4	**class** *class-name* **service-fragment** *fragment-class-name* **Example:**	Specifies a class of traffic that is the composite of all fragments matching the *fragment-class-name*. The *fragment-class-name* when defining the fragments in other policy-maps must match the *fragment-class-name* in this

	Command or Action	Purpose
	`Device(config-pmap)# class data service-fragment BestEffort`	command line to properly configure the service fragment class.
Step 5	**shape average percent** *percent* **Example:** `Device(config-pmap-c)# shape average percent 50`	Enters a QoS configuration command. Only queueing features are supported in default traffic classes configured as fragments. The queueing features that are supported are **bandwidth**, **shape**, and **random-detect exponential-weighting-constant**. Multiple QoS queueing commands can be entered.
Step 6	**end** **Example:** `Device(config-pmap-c)# end`	Exits policy-map class configuration mode and returns to privileged EXEC mode.

Examples

Note This example shows a sample configuration that is supported in releases prior to Cisco IOS XE Release 2.6.

In the following example, a policy-map is created to apply QoS to all fragments named BestEffort.

```
policy-map main-interface
 class data service-fragment BestEffort
  shape average 400000000
```

In the following example, two fragments are created and then classified collectively using a service fragment.

```
policy-map subscriber1
 class voice
  set cos 5
  priority level 1
 class video
  set cos 4
  priority level 2
 class class-default fragment BestEffort
  shape average 200000000
  bandwidth remaining ratio 10
policy-map subscriber 2
 class voice
  set cos 5
  priority level 1
 class video
  set cos 4
  priority level 2
 class class-default fragment BestEffort
```

```
  shape average 200000000
  bandwidth remaining ratio 10
```

Note This example shows a sample configuration that is supported in Cisco IOS XE Release 2.6 and later releases.

The following example shows the creation of two fragments called BestEffort in the subinterface policy-maps, followed by a sample configuration for the **service-fragment** called BestEffort to aggregate the queues at the main interface policy-map:

```
policy-map subscriber1
 class voice
  set cos 5
  account
 class video
  set cos 4
  account
 class AF1
  account
 class class-default fragment BestEffort
  shape average 200000000
  bandwidth remaining ratio 10
policy-map subscriber2
 class voice
  set cos 5
  account
 class video
  set cos 4
  account
 class AF1
  account
 class class-default fragment BestEffort
  shape average 200000000
  bandwidth remaining ratio 10
policy-map main-interface
 class voice
  priority level 1
 class video
  priority level 2
 class AF1
  bandwidth remaining ratio 90
 class data service-fragment BestEffort
  shape average 400000000
  bandwidth remaining ratio 1
```

Troubleshooting Tips

Ensure that all class statements that should be part of the same service fragment share the same *fragment-class-name*.

What to Do Next

Attach the service fragment traffic classes to the main physical interfaces.

Attach the fragment traffic classes to the member-link subinterfaces.

Configuring Service Fragments on a Physical Interface Supporting a Gigabit Etherchannel Bundle

Before you begin

This procedure assumes that a service fragment traffic class has already been created. A service fragment traffic class cannot be configured without configuring a fragment class. The procedure for creating a fragment class is documented in the "Configuring a Fragment Traffic Class in a Policy-Map" section. The procedure for creating a service fragment traffic classes is documented in the "Configuring a Service Fragment Traffic Class" section.

These instructions do not provide any details about the options that can be configured for Gigabit Etherchannel member link subinterfaces. These instructions document only the procedure for attaching a policy-map that already has a fragment traffic class to a member link subinterface.

Note For proper behavior, when a port-channel member link goes down, all member links should have the same policy-map applied.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface GigabitEthernet** *card/bay/port*
4. **service-policy output** *service-fragment-class-name*
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface GigabitEthernet** *card/bay/port* **Example:** `Device(config)# interface GigabitEthernet 0/1/0`	Specifies the member link physical interface that receives the service-policy configuration.

	Command or Action	Purpose
Step 4	**service-policy output** *service-fragment-class-name* **Example:** `Device(config-if)# service-policy output aggregate-member-link`	Attaches a service policy that contains a service fragment default traffic class to the physical Gigabit Ethernet interface.
Step 5	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

Examples

In the following example, the policy-map aggregate-member-link is attached to the physical interface.

```
interface GigabitEthernet1/1/1
 service-policy output aggregate-member-link
!
interface GigabitEthernet1/1/2
 service-policy output aggregate-member-link
```

What to do next

Ensure that the fragment class name is consistent across service-fragment and fragment class definitions. Continue to the "Configuring Fragments on Gigabit Etherchannel Member Link Subinterfaces" section.

Configuring Fragments on Gigabit Etherchannel Member Link Subinterfaces

Before you begin

This procedure assumes that a service fragment traffic class has already been created. A service fragment traffic class cannot be configured without configuring a fragment class. The procedure for creating a fragment class is documented in the "Configuring a Fragment Traffic Class in a Policy-Map" section. The procedure for creating a service fragment traffic class is documented in the "Configuring a Service Fragment Traffic Class" section.

These instructions do not provide any details about the options that can be configured for Gigabit Etherchannel member link subinterfaces. These instructions only document the procedure for attaching a policy-map that already has a fragment traffic class to a member link subinterface.

Fragments cannot be used for traffic on two or more physical interfaces.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface port-channel** *port-channel-interface-number* **.** *port-channel-subinterface-number*
4. **service-policy output** *fragment-class-name*

5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface port-channel** *port-channel-interface-number* **.** *port-channel-subinterface-number* **Example:** `Device(config)# interface port-channel 1.100`	Enters subinterface configuration mode to configure an Etherchannel member link subinterface.
Step 4	**service-policy output** *fragment-class-name* **Example:** `Device(config-subif)# service-policy output subscriber`	Attaches a service policy that contains a fragment default traffic class to the Etherchannel member link subinterface
Step 5	**end** **Example:** `Device(config-subif)# end`	Exits subinterface configuration mode and returns to privileged EXEC mode.

Example

In the following example, the service policy named subscriber has a fragment default traffic class and is attached to the port-channel subinterface of an Etherchannel bundle.

```
interface port-channel 1.100
 service-policy output subscriber
```

Configuring Ingress Policing and Marking on Port-Channel Subinterface

Before you begin

Traffic classes must be configured using the **class-map** command. A one- or two-level hierarchical policy-map should be configured using previously defined class maps. The Etherchannel member link interface should

already be configured to be part of the channel group (Etherchannel group). Cisco IOS XE Release 2.1 or later software is required. The global configuration must contain the **port-channel load-balancing vlan-manual** command or the port-channel main-interface configuration must contain the **load-balancing vlan** command. It is assumed that these commands have already been executed.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface port-channel** *port-channel-number* **.** *port-channel-interface-number* **.** *sub-interface-number*
4. **service-policy input** *policy-map-name*
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface port-channel** *port-channel-number* **.** *port-channel-interface-number* **.** *sub-interface-number* **Example:** `Device(config)# interface port-channel 1.100.100`	Enters subinterface configuration mode to configure an Etherchannel member link subinterface.
Step 4	**service-policy input** *policy-map-name* **Example:** `Device(config-subif)# service-policy input sub-intf-input`	Specifies the name of the service policy that is applied to input traffic for the port-channel subinterface previously specified.
Step 5	**end** **Example:** `Device(config-subif)# end`	Exits subinterface configuration mode and returns to privileged EXEC mode.

Example

In the following example, the service policy named sub-intf-input is defined and attached to the port-channel subinterface in the input direction.

```
policy-map sub-intf-input
  class voice
    set precedence 5
  class video
    set precedence 6
  class class-default
    set precedence 3
!
interface Port-channel 1.100
 service-policy input sub-intf-input
```

Configuring Egress Policing and Marking on Port-Channel Member Links

Before you begin

Traffic classes must be configured using the **class-map** command. A one- or two-level hierarchical policy-map should be configured using previously defined class maps. The Etherchannel member link interface should already be configured to be part of the channel group (Etherchannel group). Cisco IOS XE Release 2.1 or later software is required. The global configuration must contain the **port-channel load-balancing vlan-manual** command or the port-channel main-interface configuration must contain the **load-balancing vlan** command. It is assumed that these commands have already been executed.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface port-channel** *port-channel-number . port-channel-interface-number . sub-interface-number*
4. **service-policy output** *policy-map-name*
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface port-channel** *port-channel-number . port-channel-interface-number . sub-interface-number* **Example:** `Device(config)# interface port-channel 1.100.100`	Enters subinterface configuration mode to configure an Etherchannel member link subinterface.

	Command or Action	Purpose
Step 4	**service-policy output** *policy-map-name* **Example:** `Device(config-subif)# service-policy output WAN-GEC-member-Out-police`	Specifies the name of the service policy that is applied to output traffic for the Etherchannel member link subinterface specified in the previous step.
Step 5	**end** **Example:** `Device(config-subif)# end`	Exits subinterface configuration mode and returns to privileged EXEC mode.

Example

In the following example, the service policy named WAN-GEC-member-Out-police is defined and attached to the port-channel subinterface in the output direction.

```
policy-map WAN-GEC-member-Out-police
  class voice
    set precedence 5
  class video
    set precedence 6
  class class-default
    set precedence 3
!
interface port-channel 1.100
 service-policy output WAN-GEC-member-Out-police
```

Configuring Policies Aggregation—MQC Support for Multiple Queue Aggregation at Main Interface

Before you begin

This feature is configured using the MQC. It is most useful in QoS configurations where several policy-maps attached to the same physical interface want aggregated treatment of multiple user-defined traffic classes from multiple port-channel subinterfaces. Cisco IOS XE Release 2.6 or later software is required. The global configuration must contain the following command: **port-channel load-balancing vlan-manual** or the main interface of the port-channel being configured must have the following command: **port-channel load-balancing vlan**. It is assumed that these commands have already been executed.

This feature is supported when policy-maps are attached to multiple port-channel subinterfaces and the port-channel member link interfaces. This feature cannot be used to collectively classify default traffic classes of policy-maps on different physical interfaces. It can collectively classify all traffic directed towards a given Port-channel member-link when designated by the **primary** or **secondary** directives on the sub-interface **encapsulation** command. The following items describe the behavior and restrictions on configuring this type of QoS Policy Aggregation with Etherchannel:

- Subinterface traffic classes without configured queuing features do not have queues at the subscriber level

- Default class traffic from multiple subinterfaces can be aggregated into a common policy-map at the main interface when you use the **fragment** keyword at the subinterface **class class-default** configuration, and **service-fragment** configuration at the main interface class
- This configuration additionally enables support for other subinterface traffic classes (such as DSCP-based classes) to be aggregated into a common policy-map at the main interface.
- This feature is enabled by using the **fragment** keyword in the subinterface **class-default** class, and **service-fragment** configuration in the main interface class (this also enables aggregation of the default class.
- Queuing features are not configured at the subinterface policy-map for the other traffic classes.
- Queuing occurs at the main interface policy-map for other subinterface traffic classes as an aggregate.
- Optional tracking of statistics is supported using the **account** command for other traffic classes in the subinterface policy-map.

A multistep process is involved with the complete configuration of QoS multiple queue aggregation at a main interface feature, as follows:

1. Configure default class statements as fragments in multiple subinterface policy-maps as described in the "Configuring a Fragment Traffic Class in a Policy-Map" section.
2. Configure a separate policy-map with a class statement using the **service-fragment** keyword in order to apply QoS to the class statements configured as fragments as described in the "Configuring a Service Fragment Traffic Class" section.
3. Configure service fragment traffic classes and attach them to the main physical interfaces as described in the "Configuring Service Fragments on a Physical Interface Supporting a Gigabit Etherchannel Bundle" section.
4. Configure fragment traffic classes and attach them to the member link subinterfaces as described in the "Configuring Fragments on Gigabit Etherchannel Member Link Subinterfaces" section.

Configuring MQC Queuing on Port-Channel Member Link—No Etherchannel Load Balancing

Before you begin

Traffic classes must be configured using the **class-map** command. A one or two level hierarchical policy-map should be configured using previously defined class maps.

Cisco IOS XE Release 2.4 or later software is required.

The port-channel main interface should also contain the following commands that create an active/standby scenario. Such a configuration will allow only a single interface to be active and forwarding traffic at any time.

- **interface Port-channel1**
- **lacp fast-switchover**
- **lacp max-bundle 1**

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface GigabitEthernet** *card/bay/port*
4. **service-policy output** *policy-map-name*
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface GigabitEthernet** *card/bay/port* **Example:** `Device(config)# interface GigabitEthernet 0/1/0`	Specifies the member link physical interface that receives the service policy configuration.

Step 4	**service-policy output** *policy-map-name*
	Example:
	`Device(config-if)#`
	`service-policy output WAN-GEC-member-Out`
		Specifies the name of the service policy that is applied to output traffic.

Step 5	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

Example

In the following example, the service policy named main-intf is defined and attached to the port-channel member links in the output direction.

```
interface Port-channel 1
 lcap fast-switchover
 lacp max-bundle 1
!
policy-map main-intf
 class voice
  priority
  police cir 10000000
```

```
 class video
  bandwidth remaining ratio 10
 class class-default
  bandwidth remaining ratio 3
!
interface GigabitEthernet0/0/0
 channel-group 1 mode active
 service-policy output main-intf
!
interface GigabitEthernet0/0/1
 channel-group 1 mode active
 service-policy output main-intf
```

Configuring MQC Queuing Configuration on Port-Channel Member Link—Etherchannel Load Balancing

Before you begin

Traffic classes must be configured using the **class-map** command. A one- or two-level hierarchical policy-map should be configured using previously defined class maps. The port-channel subinterface should have been previously configured with the appropriate encapsulation subcommand to match the select primary and secondary physical interfaces on the Etherchannel. Cisco IOS XE Release 2.5 or later software is required.

The Etherchannel setup may have multiple active interfaces with flow-based load balancing enabled.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface GigabitEthernet** *card/bay/port*
4. **service-policy output** *policy-map-name*
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface GigabitEthernet** *card/bay/port* **Example:** `Device(config)# interface GigabitEthernet 0/1/0`	Specifies the member link physical interface that receives the service policy configuration.

	Command or Action	Purpose
Step 4	**service-policy output** *policy-map-name* **Example:** `Device(config-if)# service-policy output WAN-GEC-member-Out`	Specifies the name of the service policy that is applied to output traffic.
Step 5	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

Example

In the following example, the service policy named main-intf is defined and attached to the port-channel member links in the output direction.

```
  class voice
   priority
   police cir 10000000
  class video
   bandwidth remaining ratio 10
  class class-default
   bandwidth remaining ratio 3
!
interface GigabitEthernet0/0/0
 channel-group 1 mode active
 service-policy output main-intf
!
interface GigabitEthernet0/0/1
 channel-group 1 mode active
 service-policy output main-intf
```

Configuration Examples for QoS for Etherchannels

Example: Configuring QoS Policies Aggregation—Egress MQC Queuing at Subinterface

```
port-channel load-balancing vlan-manual
!
class-map match-all BestEffort
!
class-map match-all video
 match precedence 4
!
class-map match-all voice
 match precedence 5
!
```

```
policy-map subscriber
 class voice
  priority level 1
 class video
  priority level 2
 class class-default fragment BE
  shape average 100000000
  bandwidth remaining ratios 80

policy-map aggregate-member-link
 class BestEffort service-fragment BE
 shape average 100000000
!
interface Port-channel1
 ip address 209.165.200.225 255.255.0.0
!
interface Port-channel1.100
 encapsulation dot1Q 100
 ip address 209.165.200.226 255.255.255.0
 service-policy output subscriber
!
interface Port-channel1.200
 encapsulation dot1Q 200
 ip address 209.165.200.227 255.255.255.0
 service-policy output subscriber
!
interface Port-channel1.300
 encapsulation dot1Q 300
 ip address 209.165.200.228 255.255.255.0
 service-policy output subscriber
!
interface GigabitEthernet1/1/1
 no ip address
 channel-group 1 mode on
 service-policy output aggregate-member-link
!
interface GigabitEthernet1/1/2
 no ip address
 channel-group 1 mode on
 service-policy output aggregate-member-link
```

Example: Configuring QoS Policies Aggregation—MQC Support for Multiple Queue Aggregation at Main Interface

```
port-channel load-balancing vlan-manual
!
policy-map subscriber1
 class voice
  set cos 5
  account
 class video
  set cos 4
  account
 class AF1
  account
 class class-default fragment BestEffort
  shape average 200000000
  bandwidth remaining ratio 10
!
policy-map subscriber2
```

```
 class voice
  set cos 2
  account
 class video
  set cos 3
  account
 class AF1
  account
 class class-default fragment BestEffort
  shape average 200000000
  bandwidth remaining ratio 10
!
policy-map main-interface-out
 class voice
  priority level 1
 class video
  priority level 2
 class AF1
  bandwidth remaining ratio 90
 class data service-fragment BestEffort
  shape average 400000000
  bandwidth remaining ratio 1
!
interface GigabitEthernet1/1/1
 no ip address
 channel-group 1 mode on
 service-policy output main-interface-out
!
interface GigabitEthernet1/1/2
 no ip address
 channel-group 1 mode on
 service-policy output main-interface-out
!
interface Port-channel1.100
 encapsulation dot1Q 100
 ip address 10.0.0.1 255.255.255.0
 service-policy output subscriber1
!
interface Port-channel1.200
 encapsulation dot1Q 200
 ip address 10.0.0.2 255.255.255.0
 service-policy output subscriber2
!
interface Port-channel1.300
 encapsulation dot1Q 300
 ip address 10.0.0.4 255.255.255.0
 service-policy output subscriber2
```

Additional References

Related Documents

Related Topic	Document Title
Cisco IOS commands	*Cisco IOS Master Command List, All Releases*

Related Topic	Document Title
QoS commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples	*Cisco IOS Quality of Service Solutions Command Reference*
Modular Quality of Service Command-Line Interface	"Applying QoS Features Using the MQC" module
Configuring RADIUS-based policing	*Intelligent Services Gateway Configuration Guide*
CISCO ASR 1000 Series software configuration	*Cisco ASR 1000 Series Aggregation Services Routers Software Configuration Guide*

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 Quality of Service for Etherchannel Interfaces

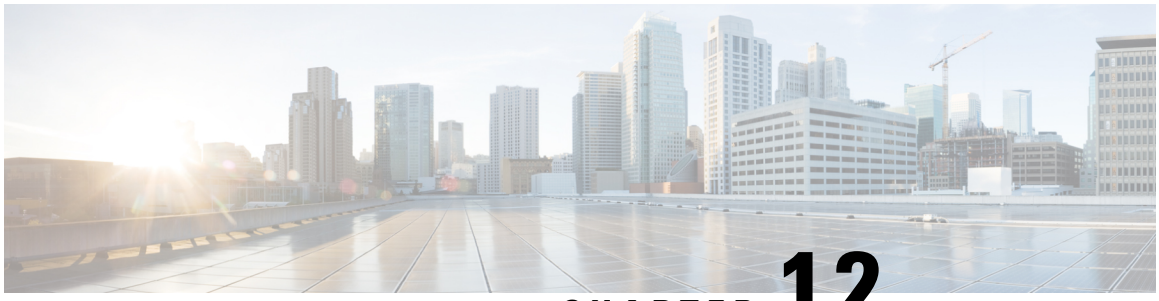
The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train 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 20: Feature Information for Quality of Service for Etherchannel Interfaces

Feature Name	Releases	Feature Information
Egress MQC Queuing Configuration on Port-Channel Subinterface	Cisco IOS XE Release 2.1	This feature supports the configuration of Egress MQC queuing on port-channel subinterface. This feature was introduced on Cisco ASR 1000 Series Routers.

Feature Name	Releases	Feature Information
Egress MQC Queuing Configuration on Port-Channel Member Link	Cisco IOS XE Release 2.1	This feature supports the configuration of Egress MQC queuing on port-channel member link. This feature was introduced on Cisco ASR 1000 Series Routers.
QoS Policies Aggregation—Egress MQC Queuing at Subinterface	Cisco IOS XE Release 2.1	This feature supports the configuration of QoS Policies Aggregation - Egress MQC queuing at subinterface. This feature was introduced on Cisco ASR 1000 Series Routers.
Ingress Policing and Marking on Port-Channel Subinterface	Cisco IOS XE Release 2.1	This feature supports the configuration of Ingress Policing and Marking on port-channel subinterface. This feature was introduced on Cisco ASR 1000 Series Routers.
Egress Policing and Marking on Port-Channel Member Link	Cisco IOS XE Release 2.1	This feature supports the configuration of Egress policing and marking on port-channel member link. This feature was introduced on Cisco ASR 1000 Series Routers.
Egress MQC Queuing Configuration on Port-Channel Member Link - No Etherchannel Load Balancing	Cisco IOS XE Release 2.4	This feature supports the configuration of Egress MQC Queuing on Port-Channel Member Link - no Etherchannel Load Balancing. This feature was introduced on Cisco ASR 1000 Series Routers.
Egress MQC Queuing Configuration Supported on Port-Channel Member Link - Etherchannel Load Balancing	Cisco IOS XE Release 2.5	This feature supports the configuration of Egress MQC Queuing on Port-Channel Member Link - Etherchannel Load Balancing. This feature was introduced on Cisco ASR 1000 Series Routers.
QoS Policies Aggregation - MQC Support for Multiple Queue Aggregation at Main Interface - Egress MQC Queuing at Main Interface	Cisco IOS XE Release 2.6	This feature supports the configuration of QoS Policies Aggregation - MQC Support for Multiple Queue Aggregation at Main Interface - Egress MQC Queuing at Main Interface. This feature was introduced on Cisco ASR 1000 Series Routers.



CHAPTER 12

Aggregate EtherChannel Quality of Service

The Aggregate EtherChannel Quality of Service (QoS) feature allows you to apply an aggregate egress-queuing policy-map on a port-channel main interface or subinterface. This feature enables QoS support on the aggregate port-channel main interface for the Cisco ASR 1000 Series Aggregation Services Routers.

Restrictions for Aggregate EtherChannel Quality of Service

- The configuration of QoS on Ethernet Virtual Circuit (EVC) with an aggregate port-channel interface is not supported.
- Point-to-Point Protocol over Ethernet (PPPoE) and IP over Ethernet (IPoE) sessions in the context of the Intelligent Services Gateway (ISG) and Intelligent Wireless Access Gateway (iWAG) (with or without QoS) across an aggregate port-channel interface is not supported.
- Virtual Private LAN Services (VPLS) with QoS on an aggregate port-channel interface is not supported.
- Xconnect with QoS on an aggregate port-channel interface is not supported.
- The use of fragment and service-fragment Modular QoS CLI (MQC) keywords in conjunction with the aggregate port-channel interface type is not supported.
- The aggregate-type port-channel interfaces have the following limitations:
  - All the member links of a port channel must be of the same speed. This prevents a potential packet reordering issue. It is not supported to combine Gigabit Ethernet, Fast Ethernet, or Ethernet interfaces into the same port channel.

- 10-Gigabit Ethernet is supported in Cisco IOS XE 3.16.3 or later (it is not supported in Cisco IOS XE 3.17). 10-Gigabit Ethernet is also supported in Cisco IOS XE Denali 16.3 and later.

- MPOL policy applied on both aggregate port-channel main interface and port-channel sub-interface is not supported by any Cisco IOS XE 3S release and is not supported on Cisco IOS XE Everest 16.5.x or earlier.
- QoS on an aggregate port-channel subinterface is not supported for Cisco IOS XE 3.16.2 or earlier (and it is also not supported in Cisco IOS XE 3.17).

Information About Aggregate EtherChannel Quality of Service

Supported Features for Aggregate EtherChannel Quality of Service

The Aggregate EtherChannel Quality of Service feature supports:

- Flow-based load balancing
- Up to three levels of hierarchy
- Configuration of shaping, absolute bandwidth, and relative bandwidth
- A minimum amount of bandwidth for subclasses (VLANs)
- Input QoS (policing and marking) and output QoS (all queuing features) that are enabled simultaneously on an aggregate port-channel main interface and subinterface

Unsupported Feature Combinations for Aggregate EtherChannel Quality of Service

The following combinations of tunnel-type interfaces with QoS are not supported:

- Generic Routing Encapsulation (GRE) tunnels with queuing policy-maps applied, which egress via a port channel with aggregate queuing
- Static virtual tunnel interface (SVTI) and dynamic virtual tunnel interface (DVTI) with queuing QoS applied, which egress via a port channel with aggregate queuing
- Sub-interface belongs to service group and sub-interface applied with service-policy cannot be configured on the same aggregate port-channel simultaneously
- MPOL - policy applied on both aggregate port-channel main interface and port-channel sub-interface

Note Tunnels without queuing QoS (described above) are supported, but are not recommended because hashing algorithms may overload a given physical interface without adequate diversity in IP addresses.

Scalability for Aggregate EtherChannel Quality of Service

The QoS policy can be applied to an aggregate port-channel interface subject to the following scalability limits:

- Up to 8 port channels
- Up to 4 member links in a port channel
- Member links can be split across multiple shared port adapters (SPAs) and SPA interface processor (SIP) cards

How to Configure Aggregate EtherChannel Quality of Service

This procedure describes how to configure Aggregate EtherChannel QoS on the Cisco ASR 1000 Series Aggregation Services Routers.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **platform qos port-channel-aggregate** *port-channel-number*
4. **interface port-channel** *port-channel-number*
5. **service-policy** {**output**} *policy-map*
6. **service-policy** {**input**} *policy-map*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**platform qos port-channel-aggregate** *port-channel-number* **Example:** `router(config)# platform qos port-channel-aggregate 1`	Enables the aggregate port-channel interface.
Step 4	**interface port-channel** *port-channel-number* **Example:** `router(config)# interface port-channel 1`	Enters interface configuration mode to configure a specific port channel.

	Command or Action	Purpose
Step 5	**service-policy** {**output**} *policy-map* **Example:** `router(config-if)# service-policy output egress_policy`	Attaches a policy-map to an output interface to be used as the service policy for that interface.
Step 6	**service-policy** {**input**} *policy-map* **Example:** `router(config-if)# service-policy input ingress_policy`	Attaches a policy-map to an input interface to be used as the service policy for that interface.

How to Unconfigure Aggregate EtherChannel Quality of Service

This procedure describes how to unconfigure Aggregate EtherChannel QoS on the Cisco ASR 1000 Series Aggregation Services Routers.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **no interface port-channel** *port-channel-number*
4. **no platform qos port-channel-aggregate** *port-channel-number*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**no interface port-channel** *port-channel-number* **Example:** `router(config)# no interface port-channel 1`	Unconfigures a specific port channel.
Step 4	**no platform qos port-channel-aggregate** *port-channel-number* **Example:** `router(config)# no platform qos port-channel-aggregate 1`	Disables the aggregate port-channel interface and removes the required QoS policies on it.

Configuration Examples for Aggregate EtherChannel Quality of Service

Example: Configuring Aggregate Port-Channel Interface

```
Router# configure terminal
Router(config)# platform qos port-channel-aggregate 1
Router(config)# interface port-channel 1
Router(config-if)# interface GigabitEthernet1/0/1
Router(config-if)# channel-group 1
Router(config-if)# interface GigabitEthernet1/0/0
Router(config-if)# channel-group 1
Router(config-if)# interface port-channel 1.1
Router(config-subif)# encap
Router(config-subif)# encapsulation dot
Router(config-subif)# encapsulation dot1Q 2
Router(config-subif)# ip addr 14.0.1.2 255.255.255.0
Router(config-subif)# interface port-channel 1.2
Router(config-subif)# encapsulation dot1Q 3
Router(config-subif)# ip addr 14.0.2.2 255.255.255.0
Router(config-subif)# interface port-channel 1.3
Router(config-subif)# encapsulation dot1Q 4
Router(config-subif)# ip addr 14.0.3.2 255.255.255.0
Router(config-subif)# end
```

Example: Configuring a Class Map for QoS

```
Router# configure terminal
Router(config)# class-map vlan_2
Router(config-cmap)# match vlan  2
Router(config-cmap)# class-map vlan_3
Router(config-cmap)# match vlan  3
Router(config-cmap)# class-map vlan_4
Router(config-cmap)# match vlan  4
Router(config-cmap)# class-map prec1
Router(config-cmap)# match precedence 1
Router(config-cmap)# class-map prec2
Router(config-cmap)# match precedence 2
Router(config-cmap)# class-map prec3
Router(config-cmap)# match precedence 3
Router(config-cmap)# class-map prec4
Router(config-cmap)# match precedence 4
Router(config-cmap)# end
```

Example: Configuring a Policy-Map for QoS

```
Router# configure terminal
Router(config)# policy-map child-vlan
Router(config-pmap)# class prec1
Router(config-pmap-c)# police cir percent 20
Router(config-pmap-c-police)# exit
Router(config-pmap-c)# priority level 1
```

```
Router(config-pmap-c)# class prec2
Router(config-pmap-c)# police cir percent 40
Router(config-pmap-c-police)# exit
Router(config-pmap-c)# priority level 2
Router(config-pmap-c)# class prec3
Router(config-pmap-c)# bandwidth remaining ratio 3
Router(config-pmap-c)# class class-default
Router(config-pmap-c)# bandwidth remaining ratio 1
Router(config-pmap-c)# random-detect
Router(config-pmap-c)#!
Router(config-pmap-c)# policy-map egress_policy
Router(config-pmap)# class vlan_2
Router(config-pmap-c)# shape average 100000000
Router(config-pmap-c)# service-policy child-vlan
Router(config-pmap-c)# class vlan_3
Router(config-pmap-c)# shape average 200000000
Router(config-pmap-c)# service-policy child-vlan
Router(config-pmap-c)# class vlan_4
Router(config-pmap-c)# shape average 300000000
Router(config-pmap-c)# service-policy child-vlan
Router(config-pmap-c)#!
Router(config-pmap-c)# policy-map ingress_policy
Router(config-pmap)# class vlan_2
Router(config-pmap-c)# police cir 80000000
Router(config-pmap-c-police)# conform-action set-prec-transmit 1
Router(config-pmap-c-police)# class vlan_2
Router(config-pmap-c)# set dscp AF21
Router(config-pmap-c)# class class-default
Router(config-pmap-c)# set dscp 0
Router(config-pmap-c)# end
```

Example: Applying QoS to Port Channel Interface

```
Router# configure terminal
Router(config)# interface port-channel 1
Router(config-if)# service-policy output egress_policy
Router(config-if)# service-policy input ingress_policy
Router(config-if)# end
```

How to Configure Aggregate EtherChannel Subinterface Quality of Service

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **platform qos port-channel-aggregate** *port-channel-number*
4. **interface port-channel** *port-channel-number*
5. **interface port-channel** *port-channel-number.subinterface-number*
6. **service-policy** {**output**} *policy-map*
7. **service-policy** {**input**} *policy-map*
8. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**platform qos port-channel-aggregate** *port-channel-number* **Example:** `Device(config)# platform qos port-channel-aggregate 1`	Enables the aggregate port-channel interface.
Step 4	**interface port-channel** *port-channel-number* **Example:** `Device(config)# interface port-channel 1`	Enters interface configuration mode to configure a specific port channel.
Step 5	**interface port-channel** *port-channel-number.subinterface-number* **Example:** `Device(config)# interface port-channel 1.2`	Enters interface configuration mode to configure a specific port channel subinterface.
Step 6	**service-policy** {**output**} *policy-map* **Example:** `Device(config-if)# service-policy output egress_policy`	Attaches a policy-map to an output interface to be used as the service policy for that interface.
Step 7	**service-policy** {**input**} *policy-map* **Example:** `Device(config-if)# service-policy input ingress_policy`	Attaches a policy-map to an input interface to be used as the service policy for that interface.
Step 8	**end** **Example:** `Device(config)# end`	Exits global configuration mode.

How to Unconfigure Aggregate EtherChannel Subinterface Quality of Service

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **no interface port-channel** *port-channel-number.subinterface*
4. **no platform qos port-channel-aggregate** *port-channel-number*
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**no interface port-channel** *port-channel-number.subinterface* **Example:** `Device(config)# no interface port-channel 1.2`	Unconfigures a specific port channel subinterface.
Step 4	**no platform qos port-channel-aggregate** *port-channel-number* **Example:** `Device(config)# no platform qos port-channel-aggregate 1`	Disables the aggregate port-channel interface and removes the required QoS policies on it.
Step 5	**end** **Example:** `Device(config)# end`	Exits global configuration mode.

Configuration Examples for Aggregate EtherChannel Subinterface Quality of Service

Example: Configuring Aggregate Port-Channel Interface and Subinterface

```
Device# configure terminal
Device(config)# platform qos port-channel-aggregate 2
Device(config)# interface port-channel 2
Device(config-if)# interface GigabitEthernet1/1/1
Device(config-if)# channel-group 2
Device(config-if)# interface GigabitEthernet1/1/0
Device(config-if)# channel-group 2
Device(config-if)# interface port-channel 2.200
Device(config-subif)# encapsulation dot1Q 200
Device(config-subif)# ip addr 15.0.1.2 255.255.255.0
Device(config-subif)# interface port-channel 2.300
Device(config-subif)# encapsulation dot1Q 300
Device(config-subif)# ip addr 15.0.2.2 255.255.255.0
Device(config-subif)# end
```

Example: Configuring a Class Map for QoS

```
Device# configure terminal
Device(config)# class-map vlan_2
Device(config-cmap)# match vlan  2
Device(config-cmap)# class-map vlan_3
Device(config-cmap)# match vlan  3
Device(config-cmap)# class-map vlan_4
Device(config-cmap)# match vlan  4
Device(config-cmap)# class-map prec1
Device(config-cmap)# match precedence 1
Device(config-cmap)# class-map prec2
Device(config-cmap)# match precedence 2
Device(config-cmap)# class-map prec3
Device(config-cmap)# match precedence 3
Device(config-cmap)# class-map prec4
Device(config-cmap)# match precedence 4
Device(config-cmap)# end
```

Example: Configuring a Policy-Map for QoS

```
Device# configure terminal
Device(config)# policy-map subinterface_child
Device(config-pmap)# class prec1
Device(config-pmap-c)# police cir percent 30
Device(config-pmap-c-police)# exit
Device(config-pmap-c)# priority level 1
Device(config-pmap-c)# class prec2
Device(config-pmap-c)# police cir percent 30
Device(config-pmap-c-police)# exit
Device(config-pmap-c)# priority level 2
Device(config-pmap-c)# class prec3
```

```
Device(config-pmap-c)# bandwidth remaining ratio 3
Device(config-pmap-c)# class class-default
Device(config-pmap-c)# bandwidth remaining ratio 1
Device(config-pmap-c)#!
Device(config-pmap-c)# policy-map sub_egress_policy
Device(config-pmap-c)# class class-default
Device(config-pmap-c)# shape average 300000000
Device(config-pmap-c)# service-policy subinterface_child
Device(config-pmap-c)#!
Device(config-pmap-c)# policy-map sub_ingress_policy
Device(config-pmap)# class class-default
Device(config-pmap-c)# police cir 80000000
Device(config-pmap-c)# end
```

Example: Applying QoS to Port Channel Subinterface

```
Device# configure terminal
Device(config)# interface port-channel 2.200
Device(config-if)# service-policy output egress_policy
Device(config-if)# service-policy input ingress_policy
Device(config)# interface port-channel 2.300
Device(config-if)# service-policy output egress_policy
Device(config-if)# service-policy input ingress_policy
Device(config-if)# end
```

Additional References

Related Documents

Related Topic	Document Title
Cisco IOS commands	*Cisco IOS Master Commands List, All Releases*
QoS commands	Cisco IOS Quality of Service Solutions Command Reference

MIBs

MIB	MIBs Link
No new or modified MIBs are supported by this feature.	To locate and download MIBs for selected platforms, Cisco software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

Technical Assistance

Description	Link
The Cisco Support website provides extensive online resources, including documentation and tools for troubleshooting and resolving technical issues with Cisco products and technologies. To receive security and technical information about your products, you can subscribe to various services, such as the Product Alert Tool (accessed from Field Notices), the Cisco Technical Services Newsletter, and Really Simple Syndication (RSS) Feeds. Access to most tools on the Cisco Support website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for Aggregate EtherChannel Quality of Service

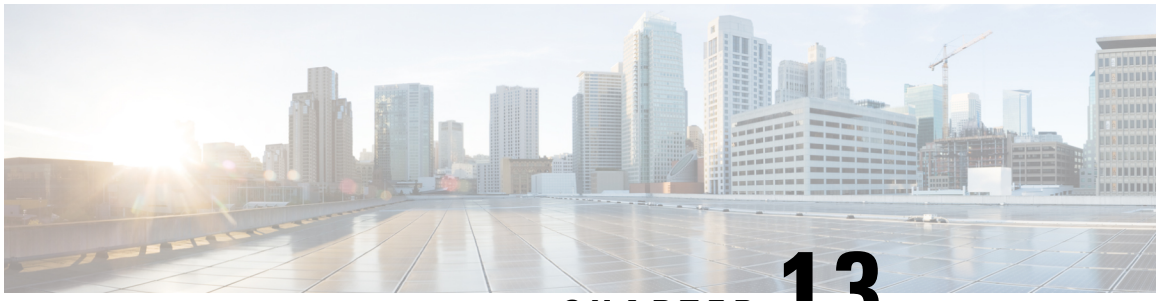
The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train 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 21: Feature Information for Aggregate EtherChannel Quality of Service

Feature Name	Releases	Feature Information
Aggregate EtherChannel Quality of Service	Cisco IOS XE Release 3.12S	The Aggregate EtherChannel Quality of Service (QoS) feature allows you to apply an aggregate egress-queuing policy-map on a port-channel main interface or subinterface. This feature enables QoS support on the aggregate port-channel main interface for the Cisco ASR 1000 Series Aggregation Services Routers. In Cisco IOS XE Release 3.12S, this feature was implemented on the Cisco ASR 1000 Series Aggregation Services Routers.
Aggregate GEC QoS 10G support	Cisco IOS XE Release 3.16.3S Cisco IOS XE Denali 16.3.1	In Cisco IOS XE Release 3.16.3S, this feature was implemented on the Cisco ASR 1000 Series Aggregation Services Routers.

Feature Name	Releases	Feature Information
QoS on GEC portchannel subinterface on ASR1K	Cisco IOS XE Release 3.16.3S Cisco IOS XE Denali 16.3.1	In Cisco IOS XE Release 3.16.3S, this feature was implemented on the Cisco ASR 1000 Series Aggregation Services Routers.
QoS on GEC portchannel subinterface on ISR 4000	Cisco IOS XE Everest 16.6.1	In Cisco IOS XE Everest 16.6.1 release, this feature was implemented on the Cisco ISR 4000 Series Integrated Services Routers.



CHAPTER 13

PPPoGEC Per Session QoS

The PPPoGEC Per Session QoS feature supports the configuration of specific QoS policies on PPPoE sessions on the PPP Termination and Aggregation (PTA), L2TP Access Concentrator (LAC), or L2TP Network Server (LNS) devices in a PPPoE /L2TP environment (broadband deployments). PPPoE sessions with Etherchannel Active/Standby functionality is also supported on Cisco ASR 1000 Series Routers acting as PTA, LAC, or LNS devices in a PPPoE/L2TP environment.

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.

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 PPPoGEC Per Session QoS

Restrictions for PPPoGEC Per Session QoS

- QoS policy-maps cannot be configured on member links, a port-channel main interface, or a port-channel subinterface that is associated with the transmit path for PPPoE sessions with QoS.

PPPoGEC Sessions with Active/Standby Etherchannel

PPPoE sessions with active/standby Etherchannel support one-level or two-level hierarchical output policy-maps (with queueing settings) also support flat input policy-maps (without queueing settings). The policy-maps are configured using previously defined class maps. The traffic classes must be configured using the **class-map** command.

The output hierarchical policy-map and the input policy-map can be associated with the PPPoE sessions in one of the following ways:

- Configuration settings on a virtual template interface
- Dynamic configuration settings via external tools configured in the authentication, authorization, and accounting (AAA) model (for example, a radius server). For more information, see the *Intelligent Services Gateway Configuration Guide* and the *Cisco ASR 1000 Series Aggregation Services Routers Software Configuration Guide*.

The port-channel main interface must contain the following commands that create an active/standby scenario. Such a configuration will allow only a single interface to be active and forwarding traffic at any time.

- **interface port-channel1**
- **lacp fast-switchover**
- **lacp max-bundle 1**

How to Configure PPPoGEC Per Session QoS

Configuring QoS on PPPoE Sessions with Etherchannel Active/Standby

To configure QoS on PPPoE sessions, you must specify the virtual template to use for PPP sessions on the Etherchannel interface, specify the name of the service policy that is applied to input traffic, and specify the output traffic. This configuration shows how to associate the output hierarchical policy-map and the input policy-map with the PPPoE sessions by defining a virtual template interface.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface virtual-template** *number*
4. **service-policy output** *policy-map-name*
5. **service-policy input** *policy-map-name*
6. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:**	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
	`Device> enable`	
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface virtual-template** *number* **Example:** `Device(config)# interface virtual-template 99`	Creates a virtual template interface that can be configured and applied dynamically in creating virtual access interfaces, and enters interface configuration mode. • Specify the virtual template to use for PPP sessions on the Etherchannel interface.
Step 4	**service-policy output** *policy-map-name* **Example:** `Device(config-if)# service-policy output session_parent`	Specifies the name of the service policy that is applied to output traffic.
Step 5	**service-policy input** *policy-map-name* **Example:** `Device(config-if)# service-policy input session_ingress`	Specifies the name of the service policy that is applied to input traffic.
Step 6	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

Configuration Examples for PPPoGEC Per Session QoS

Example: QoS on PPPoE Sessions with Etherchannel Active/Standby

The following example shows the session_parent hierarchical policy-map and the session_ingress policy-map. These policy-maps are attached to a virtual template interface using the **service-policy** command.

```
policy-map session_child
  class voice
    priority level 1
    police cir 256000
    set precedence 5
  class web
    bandwidth remaining ratio 10
  class p2p
```

```
    bandwidth remaining ratio 1
    set precedence 1
  class class-default
    set precedence 2
    bandwidth remaining ratio 5
!
policy-map session_parent
  class class-default
    bandwidth remaining ratio 1
    shape average 25000000
    service-policy session_child
!
policy-map session_ingress
  class voip
    police cir 256000
  class p2p
    police cir 256000 pir 512000
      conform-action set-prec-transmit 1
      exceed  set-prec-transmit 0
      violate drop
  class class-default
    police cir 5000000
      conform-action set-prec-transmit 2
      exceed  drop
!
interface Virtual-template 99
 service-policy output session_parent
 service-policy input session_ingress
```

Additional References for PPPoGEC Per Session QoS

Related Documents

Related Topic	Document Title
Cisco IOS commands	*Cisco IOS Master Command 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*
Modular Quality of Service Command-Line Interface	"Applying QoS Features Using the MQC" module
Configuring RADIUS-based policing	*Intelligent Services Gateway Configuration Guide*
CISCO ASR 1000 Series software configuration	*Cisco ASR 1000 Series Aggregation Services Routers Software Configuration Guide*

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

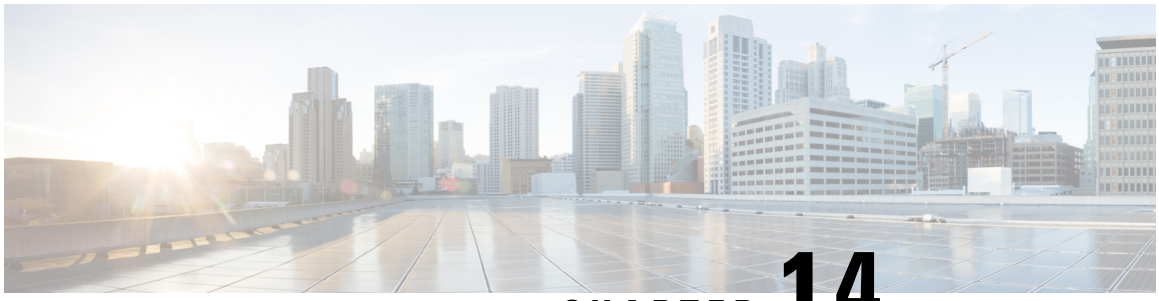
Feature Information for PPPoGEC Per Session QoS

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 22: Feature Information for PPPoGEC Per Session QoS

Feature Name	Releases	Feature Information
PPPoGEC: Per Session QoS	Cisco IOS XE Release 3.7S	This feature supports the configuration of specific QoS policies on PPPoE sessions on the PTA, LAC, and LNS for broadband deployments. This feature was introduced on Cisco ASR 1000 Series Routers. In Cisco IOS XE Release 3.8S, support was added for per-session QoS in 1:1 mode for PPPoGEC. Also, support for Point-to-Point Protocol (PPP) and IP over PPPoE was also added for PPPoGEC. In Cisco IOS XE Release 3.9S, support was added for IP session over GEC in 1:1 mode.



CHAPTER 14

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.

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 process input queue is less than the SPD minimum threshold.
- Random drop: The process input queue is between the SPD minimum and maximum thresholds.
- Max: The process input queue is equal to the SPD maximum threshold.

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 Mode

Three IPv6 SPD modes are supported: none (which is the default), aggressive drop, and OSPF mode. The aggressive drop mode discards incorrectly formatted packets when the IPv6 is in the random drop state. OSPF mode provides a mechanism whereby OSPF packets are handled with SPD priority.

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 6, then the packet is given priority.

SPD prioritizes IPv6 packets with a precedence of 6 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).

Because Interior Gateway Protocols (IGPs) 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).

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 are treated as normal priority as a result of being Layer 2 instead of Layer 3. In addition, 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. Therefore, during BGP convergence or during times of very high BGP activity, IGP hellos and keepalives often are dropped, causing IGP adjacencies to fail.

How to Configure IPv6 Selective Packet Discard

Configuring the 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** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 spd queue max-threshold** *value* **Example:** `Router(config)# ipv6 spd queue max-threshold 60000`	Configures the maximum number of packets in the SPD process input queue.
Step 4	**ipv6 spd queue min-threshold** *value* **Example:** `Router(config)# ipv6 spd queue max-threshold 4094`	Configures the minimum number of packets in the IPv6 SPD process input queue.
Step 5	exit **Example:** `Router(config)# exit`	Returns the router to privileged EXEC mode.
Step 6	**show ipv6 spd** **Example:** `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** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 spd mode** {**aggressive** | **tos protocol ospf**} **Example:** `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** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**spd headroom** *size* **Example:** `Router(config)# spd headroom 200`	Configures SPD headroom.

	Command or Action	Purpose
Step 4	**spd extended-headroom** *size* **Example:** `Router(config)# spd extended-headroom 11`	Configures extended SPD headroom.
Step 5	exit **Example:** `Router(config)# exit`	Returns the router to privileged EXEC mode.
Step 6	**show ipv6 spd** **Example:** `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

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

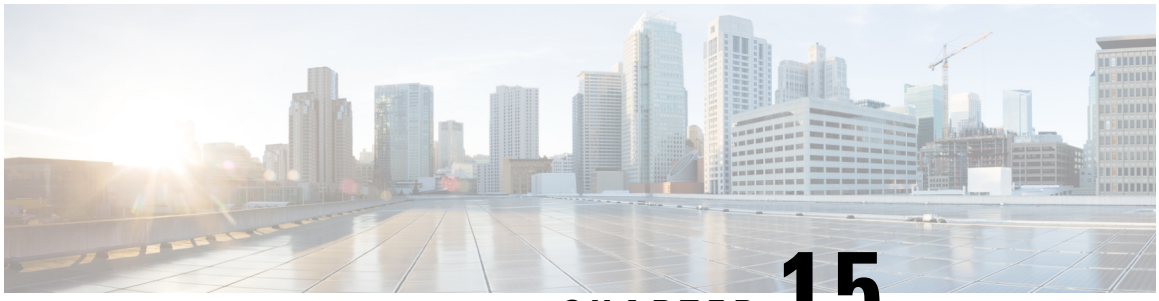
Feature Information for IPv6 Selective Packet Discard

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 23: Feature Information for IPv6 Selective Packet Discard

Feature Name	Releases	Feature Information
IPv6: Full Selective Packet Discard Support	Cisco IOS XE Release 2.6	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. 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 15

Per ACE QoS Statistics

The Per ACE QoS Statistics feature extends the QoS Packet Matching Statistics feature to allow you to track the number of packets and bytes matching individual access control elements (ACEs) used in a filter. The filter is part of the class-map definition of a quality of service (QoS) policy-map.

You can use the **show access-lists** command to display per-ACE statistics.

See the "QoS Packet Matching Statistics" module for information on defining a QoS packet filter and displaying the number of packets and bytes matching that filter.

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.

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.

Prerequisites for Per ACE QoS Statistics

Before you configure the **platform qos match-statistics per-ace** command to enable QoS per-ACE packet-matching statistics, you must configure the **platform qos match-statistics per-filter** command to enable QoS per-filter packet-matching statistics. If you do not, the CLI rejects the command and displays an error message.

Restrictions for Per ACE QoS Statistics

If a QoS policy-map is attached to the device when you configure the **platform qos match-statistics per-ace** command, the command does not take effect until you do one of the following:

- Reload the device.
- Detach all QoS policies and configure the command again.

Enabling the Per ACE QoS Statistics feature may increase CPU utilization on a scaled configuration. Before you enable it, you should weigh the benefits of the statistics information against the increased CPU utilization on the system.

Note You must configure the **platform qos match-statistics per-filter** command before you configure the **platform qos match-statistics per-ace** command.

Information About Per ACE QoS Statistics

Per ACE QoS Statistics Overview

The Per ACE QoS Statistics feature provides hit counters for ACEs used in QoS policies. When enabled, the feature adds QoS hit counters for any ACEs used in a QoS policy to the existing security access-list counters for that ACE. You can use the **show ip access-lists** command to display the access-list counters, as shown in this example:

```
Device# show ip access-lists

Extended IP access list A1
10 permit ip 10.1.1.0 0.0.0.255 any (129580275 matches)
Extended IP access list A6and7
10 permit ip 10.1.6.0 0.0.0.255 any (341426749 matches)
20 permit ip 10.1.7.0 0.0.0.255 any (398245767 matches)
Extended IP access list source
10 permit ip any host 10.1.1.5 (16147976 matches)
```

The QoS hit counters (for ACEs used in QoS policies) will be added to the counters shown in the sample output.

Note the following conditions when you enable the Per ACE QoS Statistics feature:

- The **show ip access-lists** command does not display interface information. This means that access-list counts are not interface-specific; they are aggregate counters of all hits for all features that use the ACEs and support the counts across all interfaces and directions.
- You can use the **show policy-map interface** command to display interface-specific counts if QoS per-filter packet matching statistics is enabled. However, this command displays only counts per-filter [access-control list (ACL) or access-group], not counts per-ACE, as shown in this example:

  ```
  Device# show policy-map interface GigabitEthernet0/0/2
  ```

```
GigabitEthernet0/0/2

 Service-policy input: test-match-types

   Class-map: A1orA2-class (match-any)
     482103366 packets, 59780817384 bytes
     5 minute offered rate 6702000 bps
     Match: access-group name A1
       62125633 packets, 7703578368 bytes
       5 minute rate 837000 bps
     Match: access-group name A2
       419977732 packets, 52077238892 bytes
       5 minute rate 5865000 bps
```

- If an ACE is present in a QoS filter (that is, a match statement within a class map) but the packet does not match the ACE, the ACE counter is not incremented for that packet. This can happen in the following circumstances:
  - The ACE is used in a “deny” statement.
  - Other matching criteria in a “match-all” class-map definition (for example, “match ip prec 1”) prevent the packet from matching the class.
  - Other matching criteria in a “match-any” class-map definition (for example, “match ip prec 1”) match the packet and prevent it from matching the ACE match criteria (that filter precedes the ACE filter and the packet matches both statements).

- Access-list counts are an aggregate, for a particular ACE, of the hit counts for all features that use that ACE and support per-ACE counts. This means that a single packet might hit, and be counted by, multiple features using the same ACE, and, therefore, result in multiple counts for the same packet as it traverses each feature.

  The following example shows these multiple counts. Only 1,000 packets were received on the interface but the access-list counts show 2,000 hits, 1,000 for the security access list and 1,000 for the QoS service policy.

```
Device(config)# ip access-list extended A1
permit ip 32.1.1.0 0.0.0.255 any
class-map match-all A1-class
match access-group name A1
interface GigabitEthernet0/0/2
ip address 10.0.0.1 240.0.0.0
ip access-group A1 in
duplex auto
speed auto
media-type rj45
no negotiation auto
service-policy input simple
end

Device# show access-lists

Extended IP access list A1
10 permit ip 10.1.1.0 0.0.0.255 any (2000 matches)

Device# show policy-map interface GigabitEthernet0/0/2

Service-policy input: simple
Class-map: A1-class (match-all)
1000 packets, 124000 bytes
5 minute offered rate 4000 bps
```

```
Match: access-group name A1
Class-map: class-default (match-any)
0 packets, 0 bytes
5 minute offered rate 261000 bps, drop rate 0 bps
Match: any
```

How to Configure Per ACE QoS Statistics

Configuring Per ACE QoS Statistics

Before you begin

The **platform qos match-statistics per-filter** command must be configured to enable QoS per-filter packet-matching statistics. You can use the **show platform hardware qfp active feature qos config global** command to verify the status of packet-matching statistics.

```
Device# show platform hardware qfp active feature qos config global

Marker statistics are: disabled
Match per-filter statistics are: enabled <<<<<<<
Match per-ace statistics are: disabled <<<<<<
Performance-Monitor statistics are: disabled
```

SUMMARY STEPS

1. **platform qos match-statistics per-filter**
2. **platform qos match-statistics per-ace**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**platform qos match-statistics per-filter** **Example:** `Device(config)# platform qos match-statistics per-filter`	Enables QoS packet-matching statistics for individual filters in a class map.
Step 2	**platform qos match-statistics per-ace** **Example:** `Device(config)# platform qos match-statistics per-ace`	Enables QoS packet-matching statistics for ACEs used in QoS filters.

Additional References for Per ACE QoS Statistics

Related Documents

Related Topic	Document Title
Cisco IOS commands	*Cisco IOS Master Command List, All Releases*
QoS commands	*Cisco IOS Quality of Service Solutions Command Reference*
Defining a QoS packet filter and displaying the number of packets and bytes matching it	"QoS Packet Matching Statistics"

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 Per ACE QoS Statistics

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 24: Feature Information for Per ACE QoS Statistics

Feature Name	Releases	Feature Information
Per ACE QoS Statistics	Cisco IOS XE Release 3.10S	Allows you to configure per ACE QoS statistics to track the number of packets and bytes matching individual ACEs used in a filter within a QoS service policy. The following command was introduced or modified: **platform qos match-statistics per-ace**.