Configuring Modular QoS Service Packet Classification and Marking on Cisco ASR 9000 Series Routers

Packet classification identifies the traffic flow and marking identifies traffic flows that require congestion management or congestion avoidance on a data path. The Modular Quality of Service (QoS) command-line interface (MQC) is used to define the traffic flows that should be classified, where each traffic flow is called a *class of service*, or *class*. Subsequently, a traffic policy is created and applied to a class. All traffic not identified by defined classes falls into the category of a default class.

This module provides the conceptual and configuration information for QoS packet classification.

Line Card, SIP, and SPA Support

Feature	ASR 9000 Ethernet Line Cards	SIP 700 for the ASR 9000
Classification Based on DEI	yes	no
Class-Based Unconditional Packet Marking	yes	yes
In-Place Policy Modification	yes	yes
IPv6 QoS	yes	yes
Packet Classification and Marking	yes	yes
Policy Inheritance	yes	yes
Port Shape Policies	yes	no
Shared Policy Instance	yes	no

Feature History for Configuring Modular QoS Packet Classification and Marking on Cisco ASR 9000 Series Routers

Release	Modification
Release 3.7.2	The Class-Based Unconditional Packet Marking feature was introduced on ASR 9000 Ethernet Line Cards. The IPv6 QoS feature was introduced on ASR 9000 Ethernet Line Cards. (QoS matching on IPv6 ACLs is not supported.) The Packet Classification and Marking feature was introduced on ASR 9000 Ethernet Line Cards.

Release 3.9.0	The Class-Based Unconditional Packet Marking feature was supported on the SIP 700 for the ASR 9000. The Packet Classification and Marking feature was supported on the SIP 700 for the ASR 9000. The Policy Inheritance feature was introduced on ASR 9000 Ethernet Line Cards and on the SIP 700 for the ASR 9000. The Shared Policy Instance feature was introduced on ASR 9000 Ethernet Line Cards.
Release 4.0	The Classification Based on DEI feature was introduced on ASR 9000 Ethernet Line Cards. The In-Place Policy Modification feature was introduced on ASR 9000 Ethernet Line Cards and on the SIP 700 for the ASR 9000. The IPv6 QoS feature was supported on the SIP 700 for the ASR 9000. Support for three stand-alone marking actions and three marking actions as part of a policer action in the same class was added on the SIP 700 for the ASR 9000. (ASR 9000 Ethernet Line Cards support two stand-alone marking actions and two marking actions as part of a policer action in the same class.)
Release 4.0.1	Support for the Port Shape Policies feature was introduced on ASR 9000 Ethernet Line Cards.

Contents

Prerequisites for Configuring Modular QoS Packet Classification and Marking on Cisco ASR 9000 Series Routers

The following prerequisites are required for configuring modular QoS packet classification and marking on your network:

- You must be in a user group associated with a task group that includes the proper task IDs.The command reference guides include the task IDs required for each command. If you suspect user group assignment is preventing you from using a command, contact your AAA administrator for assistance.
- You must be familiar with Cisco IOS XR QoS configuration tasks and concepts.

Information About Configuring Modular QoS Packet Classification and Marking on Cisco ASR 9000 Series Routers

To implement QoS packet classification features in this document, you must understand the following concepts:

Packet Classification Overview

Packet classification involves categorizing a packet within a specific group (or class) and assigning it a traffic descriptor to make it accessible for QoS handling on the network. The traffic descriptor contains information about the forwarding treatment (quality of service) that the packet should receive. Using packet classification, you can partition network traffic into multiple priority levels or classes of service. The source agrees to adhere to the contracted terms and the network promises a quality of service. Traffic policers and traffic shapers use the traffic descriptor of a packet to ensure adherence to the contract.

Traffic policers and traffic shapers rely on packet classification features, such as IP precedence, to select packets (or traffic flows) traversing a router or interface for different types of QoS service. For example, by using the three precedence bits in the type of service (ToS) field of the IP packet header, you can categorize packets into a limited set of up to eight traffic classes. After you classify packets, you can use other QoS features to assign the appropriate traffic handling policies including congestion management, bandwidth allocation, and delay bounds for each traffic class.

Note IPv6-based classification is supported only on Layer 3 interfaces.

Traffic Class Elements

The purpose of a traffic class is to classify traffic on your router. Use the **class-map** command to define a traffic class.

A traffic class contains three major elements: a name, a series of **match** commands, and, if more than one **match** command exists in the traffic class, an instruction on how to evaluate these **match** commands. The traffic class is named in the **class-map** command. For example, if you use the word *cisco* with the **class-map** command, the traffic class would be named *cisco*.

The **match** commands are used to specify various criteria for classifying packets. Packets are checked to determine whether they match the criteria specified in the **match** commands. If a packet matches the specified criteria, that packet is considered a member of the class and is forwarded according to the QoS specifications set in the traffic policy. Packets that fail to meet any of the matching criteria are classified as members of the default traffic class. See the "Default Traffic Class" section on page 12.

The instruction on how to evaluate these **match** commands needs to be specified if more than one match criterion exists in the traffic class. The evaluation instruction is specified with the **class-map match-any** command. If the **match-any** option is specified as the evaluation instruction, the traffic being evaluated by the traffic class must match at least one of the specified criteria. If the **match-all** option is specified, the traffic must match all of the match criteria.

The function of these commands is described more thoroughly in the *Cisco ASR 9000 Series Aggregation Services Routers Modular Quality of Service Command Reference*. The traffic class configuration task is described in the "Creating a Traffic Class" section on page 19.

Traffic Policy Elements

The purpose of a traffic policy is to configure the QoS features that should be associated with the traffic that has been classified in a user-specified traffic class or classes. The **policy-map** command is used to create a traffic policy. A traffic policy contains three elements: a name, a traffic class (specified with the **class** command), and the QoS policies. The name of a traffic policy is specified in the policy map Modular Quality of Service (MQC) (for example, the **policy-map** *policy1* command creates a traffic policy named *policy1*). The traffic class that is used to classify traffic to the specified traffic policy is defined in class map configuration mode. After choosing the traffic class that is used to classify traffic to the traffic policy, the user can enter the QoS features to apply to the classified traffic.

The MQC does not necessarily require that users associate only one traffic class to one traffic policy. When packets match to more than one match criterion, as many as 1024 traffic classes can be associated to a single traffic policy. The 1024 class maps include the default class and the classes of the child policies, if any.

The order in which classes are configured in a policy map is important. The match rules of the classes are programmed into the TCAM in the order in which the classes are specified in a policy map. Therefore, if a packet can possibly match multiple classes, only the first matching class is returned and the corresponding policy is applied.

The function of these commands is described more thoroughly in the *Cisco ASR 9000 Series Aggregation Services Router Modular Quality of Service Command Reference*.

The traffic policy configuration task is described in the "Creating a Traffic Policy" section on page 22.

Default Traffic Class

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

If the user does not configure a default class, packets are still treated as members of the default class. However, by default, the default class has no enabled features. Therefore, packets belonging to a default class with no configured features have no QoS functionality. These packets are then placed into a first

in, first out (FIFO) queue and forwarded at a rate determined by the available underlying link bandwidth. This FIFO queue is managed by a congestion avoidance technique called tail drop. For further information about congestion avoidance techniques, such as tail drop, see the "Configuring Modular QoS Congestion Avoidance on Cisco ASR 9000 Series Routers" module in this guide.

Bundle Traffic Policies

When a policy is bound to a bundle, the same policy is programmed on every bundle member (port). For example, if there is a policer or shaper rate, the same rate is configured on every port. Traffic is scheduled to bundle members based on the load balancing algorithm.

A policy can be bound to:

- Bundles
- Bundle Layer 3 subinterfaces
- Bundle Layer 2 subinterfaces (Layer 2 transport)

Both ingress and egress traffic is supported. Percentage-based policies and absolute rate-based policies are supported. However, for ease of use, it is recommended to use percentage-based policies.

Shared Policy Instance

After the traffic class and traffic policy have been created, Shared Policy Instance (SPI) can optionally be used to allow allocation of a single set of QoS resources and share them across a group of subinterfaces, multiple Ethernet flow points (EFPs), or bundle interfaces.

Using SPI, a single instance of qos policy can be shared across multiple subinterfaces, allowing for aggregate shaping of the subinterfaces to one rate. All of the subinterfaces that share the instance of a QoS policy must belong to the same physical interface. The number of subinterfaces sharing the QoS policy instance can range from 2 to the maximum number of subinterfaces on the port.

For bundle interfaces, hardware resources are replicated per bundle member. All subinterfaces that use a common shared policy instance and are configured on a Link Aggregation Control Protocol (LAG) bundle must be load-balanced to the same member link.

When a policy is configured on a bundle EFP, one instance of the policy is configured on each of the bundle member links. When using SPI across multiple bundle EFPs of the same bundle, one shared instance of the policy is configured on each of the bundle member links. By default, the bundle load balancing algorithm uses hashing to distribute the traffic (that needs to be sent out of the bundle EFPs) among its bundle members. The traffic for single or multiple EFPs can get distributed among multiple bundle members. If multiple EFPs have traffic that needs to be shaped or policed together usingSPI, the bundle load balancing has to be configured to select the same bundle member (hash-select) for traffic to all the EFPs that belong the same shared instance of the policy. This ensures that traffic going out on all the EFPs with same shared instance of the policy use the same policer/shaper Instance.

This is normally used when the same subscriber has many EFPs, for example, one EFP for each service type, and the provider requires shaping and queuing to be implemented together for all the subscriber EFPs.

Configuration tasks for SPI are described in the "Configuring Shared Policy Instance" section on page 26.

Policy Inheritance

When a policy map is applied on a physical port, the policy is enforced for all Layer 2 and Layer 3 subinterfaces under that physical port.

Port Shape Policies

When a port shaping policy is applied to a main interface, individual regular service policies can also be applied on its subinterfaces. Port shaping policy maps have the following restrictions:

- class-default is the only allowed class map.
- The shape class action is the only allowed class action.
- They can only be configured in the egress direction.
- They can only be applied to main interfaces, not to subinterfaces.
- Two- and three- level policies are not supported. Only one level or flat policies are supported.

If any of the above restrictions are violated, the configured policy map isapplied as a regular policy, not a port shaping policy.

For more information on configuring class maps, configuring policy maps, applying class actions, and applying service policies, see How to Configure Modular QoS Packet Classification and Marking on Cisco ASR 9000 Series Routers, page 19.

Class-based Unconditional Packet Marking Feature and Benefits

The Class-based, Unconditional Packet Marking feature provides users with a means for efficient packet marking by which the users can differentiate packets based on the designated markings.

The Class-based, Unconditional Packet Marking feature allows users to perform the following tasks:

- Mark packets by setting the IP precedence bits or the IP differentiated services code point (DSCP) in the IP ToS byte.
- Mark Multiprotocol Label Switching (MPLS) packets by setting the EXP bits within the imposed or topmost label.
- Mark packets by setting the Layer 2 class-of-service (CoS) value.
- Mark packets by setting inner and outer CoS tags for an IEEE 802.1Q tunneling (QinQ) configuration.
- Mark packets by setting the value of the *qos-group* argument.
- Mark packets by setting the value of the *discard-class* argument.

Note *qos-group* and *discard-class* are variables internal to the router, and are not transmitted.

Unconditional packet marking allows you to partition your network into multiple priority levels or classes of service, as follows:

- Use QoS unconditional packet marking to set the IP precedence or IP DSCP values for packets entering the network. Routers within your network can then use the newly marked IP precedence values to determine how the traffic should be treated.

For example, weighted random early detection (WRED), a congestion avoidance technique, can be used to determine the probability that a packet is dropped. In addition, low-latency queueing (LLQ) can then be configured to put all packets of that mark into the priority queue.

- Use QoS unconditional packet marking to assign packets to a QoS group. To set the QoS group identifier on MPLS packets, use the **set qos-group** command in policy map class configuration mode.

Note Setting the QoS group identifier does not automatically prioritize the packets for transmission. You must first configure an egress policy that uses the QoS group.

- Use CoS unconditional packet marking to assign packets to set the priority value of IEEE 802.1p/ Inter-Switch Link (ISL) packets. The router uses the CoS value to determine how to prioritize packets for transmission and can use this marking to perform Layer 2-to-Layer 3 mapping. To set the Layer 2 CoS value of an outgoing packet, use the **set cos** command in policy map configuration mode.

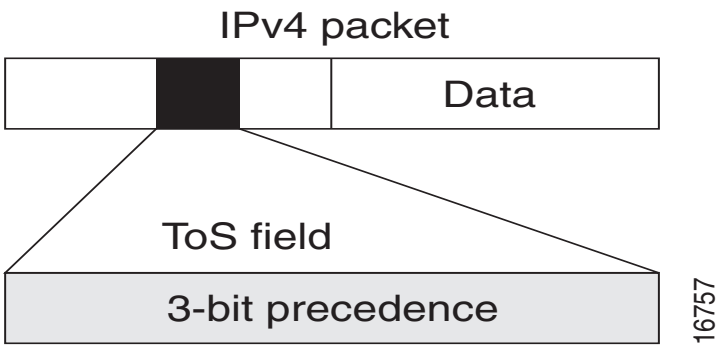
The configuration task is described in the "Configuring Class-based Unconditional Packet Marking" section on page 29.

Note

Specification of the CoS for a Packet with IP Precedence

Use of IP precedence allows you to specify the CoS for a packet. You use the three precedence bits in the ToS field of the IP version 4 (IPv4) header for this purpose. Figure 1 shows the ToS field.

Figure 1 IPv4 Packet Type of Service Field



Using the ToS bits, you can define up to eight classes of service. Other features configured throughout the network can then use these bits to determine how to treat the packet in regard to the ToS to grant it. These other QoS features can assign appropriate traffic-handling policies, including congestion management strategy and bandwidth allocation. For example, queueing features such as LLQ can use the IP precedence setting of the packet to prioritize traffic.

By setting precedence levels on incoming traffic and using them in combination with the Cisco IOS XR QoS queueing features, you can create differentiated service.

So that each subsequent network element can provide service based on the determined policy, IP precedence is usually deployed as close to the edge of the network or administrative domain as possible. This allows the rest of the core or backbone to implement QoS based on precedence.

The configuration task is described in the "Configuring Class-based Unconditional Packet Marking" section on page 29.

IP Precedence Bits Used to Classify Packets

Use the three IP precedence bits in the ToS field of the IP header to specify the CoS assignment for each packet. As mentioned earlier, you can partition traffic into a maximum of eight classes and then use policy maps to define network policies in terms of congestion handling and bandwidth allocation for each class.

For historical reasons, each precedence corresponds to a name. These names are defined in RFC 791. Table 1 lists the numbers and their corresponding names, from least to most important.

Table 1 IP Precedence Values

Number	Name
0	routine
1	priority
2	immediate
3	flash
4	flash-override
5	critical
6	internet
7	network

Note IP precedence bit settings 6 and 7 are reserved for network control information, such as routing updates.

IP Precedence Value Settings

By default, Cisco IOS XR software leaves the IP precedence value untouched. This preserves the precedence value set in the header and allows all internal network devices to provide service based on the IP precedence setting. This policy follows the standard approach stipulating that network traffic should be sorted into various types of service at the edge of the network and that those types of service should be implemented in the core of the network. Routers in the core of the network can then use the precedence bits to determine the order of transmission, the likelihood of packet drop, and so on.

Because traffic coming into your network can have the precedence set by outside devices, we recommend that you reset the precedence for all traffic entering your network. By controlling IP precedence settings, you prohibit users that have already set the IP precedence from acquiring better service for their traffic simply by setting a high precedence for all of their packets.

The class-based unconditional packet marking, LLQ, and WRED features can use the IP precedence bits.

Classification Based on DEI

You can classify traffic based on the Drop Eligible Indicator (DEI) bit that is present in 802.1ad frames and in 802.1ah frames. Default DEI marking is supported. The set DEI action in policy maps is supported on 802.1ad packets for:

- Ingress and egress
- Layer 2 subinterfaces
- Layer 2 main interfaces
- Layer 3 main interfaces

Note The set DEI action is ignored for traffic on interfaces that are not configured for 802.1ad encapsulation.

Default DEI Marking

Incoming Packet	Ingress Set Action	Default DEI on Imposed 802.1ad Headers
802.1q packet	None	0
802.1ad packet	None	DEI of top-most tag of the incoming packet
802.1q packet translated to 802.1ad packet or 802.1ad packet	set dei {0 | 1}	0 or 1 Based on DEI value in the set action

IP Precedence Compared to IP DSCP Marking

The IP DSCP value is the first six bits in the ToS byte, and the IP precedence value is the first three bits in the ToS byte. The IP precedence value is actually part of the IP DSCP value. Therefore, both values cannot be set simultaneously.

If you need to mark packets in your network and all your devices support IP DSCP marking, use the IP DSCP marking to mark your packets because the IP DSCP markings provide more unconditional packet marking options. If marking by IP DSCP is undesirable, however, or if you are unsure if the devices in your network support IP DSCP values, use the IP precedence value to mark your packets. The IP precedence value is likely to be supported by all devices in the network.

You can set up to 8 different IP precedence markings and 64 different IP DSCP markings.

In-Place Policy Modification

The In-Place Policy Modification feature allows you to modify a QoS policy even when the QoS policy is attached to one or more interfaces. When you modify the QoS policy attached to one or more interfaces, the QoS policy is automatically modified on all the interfaces to which the QoS policy is attached. A modified policy is subject to the same checks that a new policy is subject to when it is bound to an interface.

If the policy-modification is successful, the modified policy takes effect on all the interfaces to which the policy is attached. The configuration session is blocked until the policy modification is complete.

However, if the policy modification fails on any one of the interfaces, an automatic rollback is initiated to ensure that the pre-modification policy is in effect on all the interfaces. The configuration session is blocked until the rollback is complete on all affected interfaces.

If unrecoverable errors occur during in-place policy modification, the policy is put into an inconsistent state on target interfaces. Use the **show qos inconsistency** command to view inconsistency in each location. (This command is supported only on ASR 9000 Ethernet Line Cards). The configuration session is blocked until the modified policy is effective on all interfaces that are using the policy. No new configuration is possible until the configuration session is unblocked.

When a QoS policy attached to an interface is modified, there might not be any policy in effect on the interfaces in which the modified policy is used for a short period of time.

Note The QoS statistics for the policy that is attached to an interface are lost (reset to 0) when the policy is modified.

Modifications That Can Trigger In-Place Policy Modifications

Modifications to QoS Policies

- Add new actions, such as bandwidth or police
- Add new service policies (increasing the hierarchy level)
- Remove existing actions
- Modify existing actions
- Remove service-policies (decreasing the hierarchy level)
- Add new classes along with new actions
- Add or remove multiple classes in the policy
- Modify a child policy

Modifications to Class Maps

- Add new match statements
- Remove existing match statements
- Change the match type (from match-all to match-any, and vice versa)
- Modify existing match statements

Modifications to Access Lists Used in Class Maps

- Add new access control entries (ACEs)
- Remove ACEs
- Modify ACEs

Recommendations for Using In-Place Policy Modification

For a short period of time while a QoS policy is being modified, there might not be any policy in effect on the interfaces in which the modified policy is used. For this reason, modify QoS policies that affect the fewest number of interfaces at a time. Use the **show policy-map targets** command to identify the number of interfaces that will be affected during policy map modification.

How to Configure Modular QoS Packet Classification and Marking on Cisco ASR 9000 Series Routers

This section contains instructions for the following tasks:

- Creating a Traffic Class, page 19 (required)
- Creating a Traffic Policy, page 22 (required)
- Attaching a Traffic Policy to an Interface, page 24 (required)
- Shared Policy Instance, page 13 (optional)
- Configuring Class-based Unconditional Packet Marking, page 29 (required)

Creating a Traffic Class

To create a traffic class containing match criteria, use the **class-map** command to specify the traffic class name, and then use the following **match** commands in class-map configuration mode, as needed.

For conceptual information, see the "Traffic Class Elements" section on page 11.

Restrictions

All **match** commands specified in this configuration task are considered optional, but you must configure at least one match criterion for a class.

SUMMARY STEPS

1. **configure**
2. **class-map** [**type qos**] [**match-any**] [**match-all**] *class-map-name*
3. **match access-group** [**ipv4** | **ipv6**] *access-group-name*
4. **match** [**not**] **cos** [*cos-value*] [*cos-value1 ... cos-value7*]
5. **match** [**not**] **cos inner** [*inner-cos-value*] [*inner-cos-value1...inner-cos-value7*]
6. **match destination-address mac** *destination-mac-address*
7. **match source-address mac** *source-mac-address*
8. **match** [**not**] **discard-class** *discard-class-value* [*discard-class-value1 ... discard-class-value6*]
9. **match** [**not**] **dscp** [**ipv4** | **ipv6**] *dscp-value* [*dscp-value1 ... dscp-value7*]
10. **match** [**not**] **mpls experimental topmost** *exp-value* [*exp-value1 ... exp-value7*]
11. **match** [**not**] **precedence** [**ipv4** | **ipv6**] *precedence-value* [*precedence-value1 ... precedence-value7*]

12. **match** [**not**] **protocol** *protocol value* [*protocol-value1 ... protocol-value7*]
13. **match** [**not**] **qos-group** [*qos-group-value1 ... qos-group-value8*]
14. **match vlan** [**inner**] *vlanid* [*vlanid1 ... vlanid7*]
15. **end**
    or
    commit

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RSP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**class-map** [**type qos**] [**match-any**] [**match-all**] *class-map-name* **Example:** `RP/0/RSP0/CPU0:router(config)# class-map class201`	Enters class map configuration mode. • Creates a class map to be used for matching packets to the class whose name you specify. • If you specific **match-any**, one of the match criteria must be met for traffic entering the traffic class to be classified as part of the traffic class. This is the default. If you specify **match-all**, the traffic must match all the match criteria.
Step 3	**match access-group** [**ipv4** | **ipv6**] *access-group-name* **Example:** `RP/0/RSP0/CPU0:router(config-cmap)# match access-group ipv4 map1`	(Optional) Configures the match criteria for a class map based on the specified access control list (ACL) name.
Step 4	**match** [**not**] **cos** [*cos-value*] [*cos-value0 ... cos-value7*] **Example:** `RP/0/RSP0/CPU0:router(config-cmap)# match cos 5`	(Optional) Specifies a *cos-value* in a class map to match packets. • *cos-value* arguments are specified as an integer from 0 to 7.
Step 5	**match** [**not**] **cos inner** [*inner-cos-value*] [*inner-cos-value0...inner-cos-value7*] **Example:** `RP/0/RSP0/CPU0:router match cos inner 7`	(Optional) Specifies an *inner-cos-value* in a class map to match packets. • *inner-cos-value* arguments are specified as an integer from 0 to 7.
Step 6	**match destination-address mac** *destination-mac-address* **Example:** `RP/0/RSP0/CPU0:router(config-cmap)# match destination-address mac 00.00.00`	(Optional) Configures the match criteria for a class map based on the specified destination MAC address.

	Command or Action	Purpose
Step 7	**match source-address mac** *source-mac-address* **Example:** `RP/0/RSP0/CPU0:router(config-cmap)# match source-address mac 00.00.00`	(Optional) Configures the match criteria for a class map based on the specified source MAC address.
Step 8	**match** [**not**] **discard-class** *discard-class-value* [*discard-class-value1 ... discard-class-value6*] **Example:** `RP/0/RSP0/CPU0:router(config-cmap)# match discard-class 5`	(Optional) Specifies a *discard-class-value* in a class map to match packets. • *discard-class-value* argument is specified as an integer from 0 to 7. The **match discard-class** command is supported only for an egress policy.
Step 9	**match** [**not**] **dscp** [**ipv4** | **ipv6**] *dscp-value* [*dscp-value ... dscp-value*] **Example:** `RP/0/RSP0/CPU0:router(config-cmap)# match dscp ipv4 15`	(Optional) Identifies a specific DSCP value as a match criterion. • Value range is from 0 to 63. • Reserved keywords can be specified instead of numeric values. • Up to eight values or ranges con be used per match statement.
Step 10	**match** [**not**] **mpls experimental topmost** *exp-value* [*exp-value1 ... exp-value7*] **Example:** `RP/0/RSP0/CPU0:router(config-cmap)# match mpls experimental topmost 3`	(Optional) Configure a class map so that the three-bit experimental field in the topmost Multiprotocol Label Switching (MPLS) labels are examined for experimental (EXP) field values. The value range is from 0 to 7.
Step 11	**match** [**not**] **precedence** [**ipv4** | **ipv6**] *precedence-value* [*precedence-value1 ... precedence-value6*] **Example:** `RP/0/RSP0/CPU0:router(config-cmap)# match precedence ipv4 5`	(Optional) Identifies IP precedence values as match criteria. • Value range is from 0 to 7. • Reserved keywords can be specified instead of numeric values.
Step 12	**match** [**not**] **protocol** *protocol-value* [*protocol-value1 ... protocol-value7*] **Example:** `RP/0/RSP0/CPU0:router(config-cmap)# match protocol igmp`	(Optional) Configures the match criteria for a class map on the basis of the specified protocol.
Step 13	**match** [**not**] **qos-group** [*qos-group-value1 ... qos-group-value8*] **Example:** `RP/0/RSP0/CPU0:router(config-cmap)# match qos-group 1 2 3 4 5 6 7 8`	(Optional) Specifies service (QoS) group values in a class map to match packets. • *qos-group-value* identifier argument is specified as the exact value or range of values from 0 to 63. • Up to eight values (separated by spaces) can be entered in one match statement. • **match qos-group** command is supported only for an egress policy.

	Command or Action	Purpose
Step 14	**match vlan** [**inner**] *vlanid* [vlanid1 ... vlanid7] **Example:** `RP/0/RSP0/CPU0:router(config-cmap)# match vlan vlanid vlanid1`	(Optional) Specifies a VLAN ID or range of VLAN IDs in a class map to match packets. • *vlanid* is specified as an exact value or range of values from 1 to 4094. • Total number of supported VLAN values or ranges is 8.

Step 15	**end**
	or
	commit
	
	Example:
	`RP/0/RSP0/CPU0:router(config-cmap)# end`
	or
	`RP/0/RSP0/CPU0:router(config-cmap)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Creating a Traffic Policy

To create a traffic policy, use the **policy-map** global configuration command to specify the traffic policy name.

The traffic class is associated with the traffic policy when the **class** command is used. The **class** command must be issued after you enter the policy map configuration mode. After entering the **class** command, the router is automatically in policy map class configuration mode, which is where the QoS policies for the traffic policy are defined.

The following class-actions are supported:

- bandwidth—Configures the bandwidth for the class. See the “Configuring Modular Quality of Service Congestion Management on Cisco ASR 9000 Series Routers” module in this guide.
- police—Police traffic. See the “Configuring Modular Quality of Service Congestion Management on Cisco ASR 9000 Series Routers” module in this guide.
- priority—Assigns priority to the class. See the “Configuring Modular Quality of Service Congestion Management on Cisco ASR 9000 Series Routers” module in this guide.
- queue-limit—Configures queue-limit (tail drop threshold) for the class. See the “Configuring Modular QoS Congestion Avoidance on Cisco ASR 9000 Series Routers” module in this guide.
- random-detect—Enables Random Early Detection. See the “Configuring Modular QoS Congestion Avoidance on Cisco ASR 9000 Series Routers” module in this guide.

- service-policy—Configures a child service policy.
- set—Configures marking for this class. See the "Class-based Unconditional Packet Marking Feature and Benefits" section on page 14.
- shape—Configures shaping for the class. See the "Configuring Modular Quality of Service Congestion Management on Cisco ASR 9000 Series Routers" module in this guide.

For additional commands that can be entered as match criteria, see the *Cisco ASR 9000 Series Aggregation Services Router Modular Quality of Service Command Reference*.

For conceptual information, see "Traffic Policy Elements" section on page 12.

SUMMARY STEPS

1. **configure**
2. **policy-map** [**type qos**] *policy-name*
3. **class** *class-name*
4. **set precedence** *precedence-value*
5. **end**
   or
   commit

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RSP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**policy-map** [**type qos**] *policy-name* **Example:** `RP/0/RSP0/CPU0:router(config)# policy-map policy1`	Enters policy map configuration mode. • Creates or modifies a policy map that can be attached to one or more interfaces to specify a service policy.
Step 3	**class** *class-name* **Example:** `RP/0/RSP0/CPU0:router(config-pmap)# class class1`	Specifies the name of the class whose policy you want to create or change.
Step 4	**set precedence** *precedence-value* **Example:** `RP/0/RSP0/CPU0:router(config-pmap-c)# set precedence 3`	Sets the precedence value in the IP header. **Note**

	Command or Action	Purpose

Step 5	**end**
	or
	commit
	
	Example:
	`RP/0/RSP0/CPU0:router(config-pmap-c)# end`
	or
	`RP/0/RSP0/CPU0:router(config-pmap-c)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Attaching a Traffic Policy to an Interface

After the traffic class and traffic policy are created, you must use the **service-policy** interface configuration command to attach a traffic policy to an interface, and to specify the direction in which the policy should be applied (either on packets coming into the interface or packets leaving the interface).

For additional commands that can be entered in policy map class configuration mode, see the *Cisco ASR 9000 Series Aggregation Services RoutersModular Quality of Service Command Reference.*.

Prerequisites

A traffic class and traffic policy must be created before attaching a traffic policy to an interface.

Restrictions

None

SUMMARY STEPS

1. **configure**
2. **interface** *type interface-path-id*
3. **service-policy {input | output}** *policy-map*
4. **end**
   or
   commit
5. **show policy-map interface** *type interface-path-id* [**input** | **output**]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RSP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**interface** *type interface-path-id* **Example:** `RP/0/RSP0/CPU0:router(config)# interface gigabitethernet 0/1/0/9`	Enters interface configuration mode and configures an interface.
Step 3	**service-policy** {**input** | **output**} *policy-map* **Example:** `RP/0/RSP0/CPU0:router(config-if)# service-policy output policy1`	Attaches a policy map to an input or output interface to be used as the service policy for that interface. • In this example, the traffic policy evaluates all traffic leaving that interface.

Step 4	**end**
	or
	commit
	
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# end`
	or
	`RP/0/RSP0/CPU0:router(config-if)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Step 5	**show policy-map interface** *type interface-path-id* [**input** | **output**] **Example:** `RP/0/RSP0/CPU0:router# show policy-map interface gigabitethernet 0/1/0/9`	(Optional) Displays statistics for the policy on the specified interface.

Configuring Shared Policy Instance

Attaching a Shared Policy Instance to Multiple Subinterfaces

After the traffic class and traffic policy are created, you can optionally use the **service-policy (interface)** configuration command to attach a shared policy instance to multiple subinterfaces, and to specify the direction in which the policy should be applied (either on packets coming into or leaving the subinterface).

Note A shared policy can include a combination of Layer 2 and Layer 3 subinterfaces.

For additional commands that can be entered in policy map class configuration mode, see the *Cisco ASR 9000 Series Aggregation Services Routers Modular Quality of Service Command Reference*.

Prerequisites

A traffic class and traffic policy must be created before attaching a shared policy instance to a subinterface.

Restrictions

Shared policy instance across multiple physical interfaces is not supported.

SUMMARY STEPS

1. **configure**
2. **interface** *type interface-path-id*
3. **service-policy {input | output}** *policy-map* **[shared-policy-instance** *instance-name*]
4. **end**
   or
   commit
5. **show policy-map shared-policy-instance** *instance name* **[input | output] location** *rack/slot/module*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RSP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**interface** *type interface-path-id* **Example:** `RP/0/RSP0/CPU0:router(config)# interface gigabitethernet 0/1/0/0.1`	Enters interface configuration mode and configures a subinterface.

	Command or Action	Purpose
Step 3	**service-policy** {**input** | **output**} *policy-map* [**shared-policy-instance** *instance-name*] **Example:** `RP/0/RSP0/CPU0:router(config-if)# service-policy output policy1 shared-policy-instance Customer1`	Attaches a policy map to an input or output subinterface to be used as the service policy for that subinterface. • In this example, the traffic policy evaluates all traffic leaving that interface.

Step 4	**end**
	or
	commit
	
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# end`
	or
	`RP/0/RSP0/CPU0:router(config-if)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting (yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Step 5	**show policy-map shared-policy-instance** *instance-name* [**input** | **output**] **location** *rack/slot/module* **Example:** `RP/0/RSP0/CPU0:router# show policy-map shared-policy-instance Customer1 location 0/1/0/7.1`	(Optional) Displays statistics for the policy on the specified shared policy instance subinterface.

Attaching a Shared Policy Instance to Bundle Interfaces or EFP Bundles

After the traffic class and traffic policy are created, you can optionally use the **service-policy (interface)** configuration command to attach a shared policy instance to bundle interfaces and to bundle EFPs, and to specify the direction in which the policy should be applied (either on packets coming into or leaving the subinterface).

For additional commands that can be entered in policy map class configuration mode, see the *Cisco ASR 9000 Series Aggregation Services Router Modular Quality of Service Command Reference*.

Prerequisites

A traffic class and traffic policy must be created before attaching a shared policy instance to bundle interfaces or EFP bundles.

Restrictions

Shared policy instance across multiple physical interfaces is not supported.

SUMMARY STEPS

1. **configure**
2. **interface Bundle-Ether** *bundle-id*
3. **service-policy** {**input** | **output**} *policy-map* [**shared-policy-instance** *instance-name*]
4. **end**
   or
   commit
5. **show policy-map shared-policy-instance** *instance name* [**input** | **output**] **location** *location-id*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RSP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**interface Bundle-Ether** *bundle-id* **Example:** `RP/0/RP1/CPU0:router(config)# interface Bundle-Ether 100.1 l2transport`	Enters interface configuration mode and configures a bundle interface.
Step 3	**service-policy** {**input** | **output**} *policy-map* [**shared-policy-instance** *instance-name*] **Example:** `RP/0/RSP0/CPU0:router(config-if)# service-policy output policy1 shared-policy-instance Customer1`	Attaches a policy map to an input or output bundle interface to be used as the service policy for that subinterface. • In this example, the traffic policy evaluates all traffic leaving that interface.

	Command or Action	Purpose

Step 4	**end**
	or
	commit
	
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# end`
	or
	`RP/0/RSP0/CPU0:router(config-if)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Step 5	**show policy-map shared-policy-instance** *instance-name* [**input** | **output**] **location** *location-id* Example: `RP/0/RSP0/CPU0:router# show policy-map shared-policy-instance Customer1 location 0/rsp0/cpu0`	(Optional) Displays statistics for the policy at the specified shared policy instance location.

Configuring Class-based Unconditional Packet Marking

This configuration task explains how to configure the following class-based, unconditional packet marking features on your router:

- IP precedence value
- IP DSCP value
- QoS group value (ingress only)
- CoS value (egress only on Layer 3 subinterfaces)
- MPLS experimental value
- Discard class

Note IPv4 and IPv6 QoS actions applied to MPLS tagged packets are not supported. The configuration is accepted, but no action is taken.

Note Choose only two **set** commands per class.

SUMMARY STEPS

1. **configure**
2. **policy-map** *policy-name*
3. **class** *class-name*
4. **set precedence** *number*
5. **set dscp** *dscp-value*
6. **set qos-group** *qos-group-value*
7. **set cos** [**inner**] *cos-value*
8. **set mpls experimental** {**imposition** | **topmost**} *exp-value*
9. **set discard-class** *discard-class-value*
10. **exit**
11. **exit**
12. **interface** *type interface-path-id*
13. **service-policy** {**input** | **output**} *policy-map*
14. **end**
    or
    commit
15. **show policy-map interface** *type instance* [**input** | **output**]

Note Although this task illustrates all of the possible **set** commands, only two **set** commands can be configured per class.

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RSP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**policy-map** *policy-name* **Example:** `RP/0/RSP0/CPU0:router(config)# policy-map policy1`	Enters policy map configuration mode. • Creates or modifies a policy map that can be attached to one or more interfaces to specify a service policy.
Step 3	**class** *class-name* **Example:** `RP/0/RSP0/CPU0:router(config-pmap)# class class1`	Enters policy class map configuration mode. • Specifies the name of the class whose policy you want to create or change.

	Command or Action	Purpose
Step 4	**set precedence** *number* **Example:** `RP/0/RSP0/CPU0:router(config-pmap-c)# set precedence 1`	• Sets the precedence value in the IP header.
Step 5	**set dscp** *dscp-value* **Example:** `RP/0/RSP0/CPU0:router(config-pmap-c)# set dscp 5`	• Marks a packet by setting the DSCP in the ToS byte.
Step 6	**set qos-group** *qos-group-value* **Example:** `RP/0/RSP0/CPU0:router(config-pmap-c)# set qos-group 31`	Sets the QoS group identifiers on IPv4 or MPLS packets. The **set qos-group** command is supported only on an ingress policy.
Step 7	**set cos** [**inner**] *cos-value* **Example:** `RP/0/RSP0/CPU0:router(config-pmap-c)# set cos 7`	Sets the specific IEEE 802.1Q Layer 2 CoS value of an outgoing packet. Values are from 0 to7. Sets the Layer 2 CoS value of an outgoing packet. • This command should be used by a router if a user wants to mark a packet that is being sent to a switch. Switches can leverage Layer 2 header information, including a CoS value marking. • For Layer 2 interfaces, the **set cos** command: – Is rejected on ingress or egress policies on a main interface. – Is accepted but ignored on ingress policies on a subinterface. – Is supported on egress policies on a subinterface. • For Layer 3 interfaces, the **set cos** command: – Is ignored on ingress policies on a main interface. – Is rejected on ingress policies on a subinterface. – Is supported on egress policies on main interfaces and subinterfaces.
Step 8	**set mpls experimental** {**imposition** | **topmost**} *exp-value* **Example:** `RP/0/RSP0/CPU0:router(config-pmap-c)# set mpls experimental imposition 3`	Sets the experimental value of the MPLS packet top-most or imposition labels. • **imposition** can be used only in service policies that are attached in the ingress policy.
Step 9	**set discard-class** *discard-class-value* **Example:** `RP/0//CPU0:router(config-pmap-c)# set discard-class 3`	Sets the discard class on IP Version 4 (IPv4) or Multiprotocol Label Switching (MPLS) packets. • This command can be used only in service policies that are attached in the ingress policy.

	Command or Action	Purpose
Step 10	**exit** **Example:** `RP/0/RSP0/CPU0:router(config-pmap-c)# exit`	Returns the router to policy map configuration mode.
Step 11	**exit** **Example:** `RP/0/RSP0/CPU0:router(config-pmap)# exit`	Returns the router to global configuration mode.
Step 12	**interface** *type interface-path-id* **Example:** `RP/0/RSP0/CPU0:router(config)# interface pos 0/2/0/0`	Enters interface configuration mode and configures an interface.
Step 13	**service-policy** {**input** | **output**} *policy-map* **Example:** `RP/0/RSP0/CPU0:router(config-if)# service-policy output policy1`	Attaches a policy map to an input or output interface to be used as the service policy for that interface. • In this example, the traffic policy evaluates all traffic leaving that interface.

Step 14	**end**
	or
	commit
	
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# end`
	or
	`RP/0/RSP0/CPU0:router(config-if)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Step 15	**show policy-map interface** *type interface-path-id* [**input** | **output**] **Example:** `RP/0/RSP0/CPU0:router# show policy-map interface pos 0/2/0/0`	(Optional) Displays policy configuration information for all classes configured for all service policies on the specified interface.

16. **ipv4 bgp policy propagation input {ip-precedence | qos-group} {destination [ip-precedence {destination | source}] | source [ip-precedence {destination | source}] }**

Configuration Examples for Configuring Modular QoS Packet Classification and Marking on Cisco ASR 9000 Series Routers

This section contains the following examples:

Traffic Classes Defined: Example

In the following example, two traffic classes are created and their match criteria are defined. For the first traffic class called class1, ACL 101 is used as the match criterion. For the second traffic class called class2, ACL 102 is used as the match criterion. Packets are checked against the contents of these ACLs to determine if they belong to the class.

```
class-map class1
  match access-group ipv4 101
  exit
!
class-map class2
  match access-group ipv4 102
  exit
```

Use the **not** keyword with the **match** command to perform a match based on the values of a field that are not specified. The following example includes all packets in the class qos_example with a DSCP value other than 4, 8, or 10.

```
class-map match-any qos_example
  match not dscp 4 8 10
!
end
```

Traffic Policy Created: Example

In the following example, a traffic policy called policy1 is defined to contain policy specifications for the two classes—class1 and class2. The match criteria for these classes were defined in the traffic classes created in the "Traffic Classes Defined: Example" section on page 33.

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

```
policy-map policy1
```

```
  class class1
    bandwidth 3000
    queue-limit bytes 1000000000
    exit
!
  class class2
    bandwidth 2000
    exit
```

Traffic Policy Attached to an Interface: Example

The following example shows how to attach an existing traffic policy to an interface (see the "Traffic Classes Defined: Example" section on page 33). After you define a traffic policy with the **policy-map** command, you can attach it to one or more interfaces to specify the traffic policy for those 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 at the input and only one traffic policy attached at the output.

```
interface gigabitethernet 0/1/0/9
  service-policy output policy1
  exit
!
interface TenGigE 0/5/0/1
  service-policy output policy1
  exit
```

Traffic Policy Attached to Multiple Subinterfaces: Example

The following example shows how to attach an existing traffic policy to multiple subinterfaces. After you define a traffic policy with the **policy-map** command, you can attach it to one or more subinterfaces using the service policy command in subinterface configuration mode.

```
interface gigabitethernet 0/1/0/0.1
  service-policy input policy1 shared-policy-instance ethernet101
  exit
!
interface gigabitethernet 0/1/0/0.2
  service-policy input policy1 shared-policy-instance ethernet101
  exit
```

Traffic Policy Attached to a Bundle Interface: Example

The following example shows how to attach an existing traffic policy to a bundle interface. After you define a traffic policy with the **policy-map** command, you can attach it to one or more bundle subinterfaces using the service policy command in subinterface configuration mode.

```
interface Bundle-Ether 100.1
  service-policy tripleplaypolicy shared-policy-instance subscriber1
  exit
!
interface Bundle-Ether 100.2
  service-policy output tripleplaypolicy shared-policy instance subscriber1
  exit
```

EFP Load Balancing with Shared Policy Instance: Example

The following examples show how to configure load balancing of an EFP when SPI is implemented. For additional information on EFP load balancing on link bundles, see the *Cisco ASR 9000 Series Aggregation Services Router Interface and Hardware Component Configuration Guide*.

Configuring a Bundle Interface: Example

```
interface Bundle-Ether 50
interface gigabitethernet 0/1/0/5
  bundle id 50 mode active
interface gigabitethernet 0/1/0/8
  bundle id 50 mode active
```

Configuring Two Bundle EFPs with the Load Balance Options: Example

This example configures the traffic for two bundle EFPs go over the same physical member link.

```
interface Bundle-Ether 50.25 l2transport
  encapsulation dot1q 25
  bundle load-balance hash-select 2
!
interface Bundle-Ether 50.36 l2transport
  encapsulation dot1q 36
  bundle load-balance hash-select 2
```

Default Traffic Class Configuration: Example

The following example shows how to configure a traffic policy for the default class of the traffic policy called policy1. The default class is named class-default, consists of all other traffic, and is being shaped at 60 percent of the interface bandwidth.

```
policy-map policy1
  class class-default
    shape average percent 60
```

class-map match-any Command Configuration: Example

The following example illustrates how packets are evaluated when multiple match criteria exist. Only one match criterion must be met for the packet in the **class-map match-any** command to be classified as a member of the traffic class (a logical OR operator). In the example, protocol IP OR QoS group 4 OR access group 101 have to be successful match criteria:

```
class-map match-any class1
  match protocol ipv4
  match qos-group 4
  match access-group ipv4 101
```

In the traffic class called class1, the match criteria are evaluated consecutively until a successful match criterion is located. Each matching criterion is evaluated to see if the packet matches that criterion. If the packet matches at least one of the specified criteria, the packet is classified as a member of the traffic class.

Note The **match qos-group** command is supported only on egress policies.

Class-based, Unconditional Packet Marking Examples

The following are typical class-based, unconditional packet marking examples:

IP Precedence Marking Configuration: Example

In the following example, a service policy called *policy1* is created. This service policy is associated to a previously defined class map called *class1* through the use of the **class** command, and then the service policy is attached to the output POS interface 0/1/0/0. The IP precedence bit in the ToS byte is set to 1:

```
policy-map policy1
  class class1
    set precedence 1
!
interface pos 0/1/0/0
  service-policy output policy1
```

IP DSCP Marking Configuration: Example

In the following example, a service policy called policy1 is created. This service policy is associated to a previously defined class map through the use of the **class** command. In this example, it is assumed that a class map called class1 was previously configured.

In the following example, the IP DSCP value in the ToS byte is set to 5:

```
policy-map policy1
  class class1
    set dscp 5

  class class2
    set dscp ef
```

After you configure the settings shown for voice packets at the edge, all intermediate routers are configured to provide low-latency treatment to the voice packets, as follows:

```
class-map voice
  match dscp ef
policy-map qos-policy
  class voice
    priority level 1
    police rate percent 10
```

QoS Group Marking Configuration: Example

In the following example, a service policy called *policy1* is created. This service policy is associated to a class map called *class1* through the use of the **class** command, and then the service policy is attached in the input direction on a GigabitEthernet interface 0/1/0/9. The qos-group value is set to 1.

```
class-map match-any class1
  match protocol ipv4
  match access-group ipv4 101

policy-map policy1
  class class1
    set qos-group 1
  !
interface gigabitethernet 0/1/0/9
  service-policy input policy1
```

Note The **set qos-group** command is supported only on an ingress policy.

CoS Marking Configuration: Example

In the following example, a service policy called *policy1* is created. This service policy is associated to a class map called *class1* through the use of the **class** command, and then the service policy is attached in the output direction on a 10-Gigabit Ethernet interface, TenGigE0/1/0/0. The IEEE 802.1p (CoS) bits in the Layer 2 header are set to 1.

```
class-map match-any class1
  match protocol ipv4
  match access-group ipv4 101

policy-map policy1
  class class1
    set cos 1
  !
interface TenGigE0/1/0/0
  service-policy output policy1
```

MPLS Experimental Bit Imposition Marking Configuration: Example

In the following example, a service policy called *policy1* is created. This service policy is associated to a class map called *class1* through the use of the **class** command, and then the service policy is attached in the input direction on a 10-Gigabit Ethernet interface, TenGigE0/1/0/0. The MPLS EXP bits of all imposed labels are set to 1.

```
class-map match-any class1
  match protocol ipv4
  match access-group ipv4 101

policy-map policy1
  class class1
    set mpls exp imposition 1
 !
interface TenGigE0/1/0/0
  service-policy input policy1
```

Note The **set mpls exp imposition** command is supported only on an ingress policy.

MPLS Experimental Topmost Marking Configuration: Example

In the following example, a service policy called *policy1* is created. This service policy is associated to a class map called *class1* through the use of the **class** command, and then the service policy is attached in the output direction on a 10-Gigabit Ethernet interface, TenGigE0/1/0/0. The MPLS EXP bits on the TOPMOST label are set to 1:

```
class-map match-any class1
  match mpls exp topmost 2

policy-map policy1
  class class1
    set mpls exp topmost 1
  !
interface TenGigE0/1/0/0
  service-policy output policy1
```

In-Place Policy Modification: Example

In this example, the precedence is changed from 3 to 5 after the policy is defined and attached to an interface:

Define a class:

```
class-map match-any class1
 match cos 7
 end-class-map
```

Define a policy map that uses the class:

```
policy-map policy1
 class class1
 set precedence 3
```

Attach the policy map to an interface:

```
interface gigabitethernet 0/6/0/1
 service-policy output policy1
 commit
```

Modify the precedence value of the policy map:

```
policy-map policy1
 class class1
 set precedence 5
 commit
```

Note The modified policy *policy1* takes effect on all the interfaces to which the policy is attached. Also, you can modify any class map used in the policy map. The changes made to the class map take effect on all the interfaces to which the policy is attached.

Output from the **show policy-map targets** command indicates that the Gigabit Ethernet interface 0/1/0/0 has one policy map attached as a main policy (as opposed to being attached to a child policy in a hierarchical QoS configuration). Outgoing traffic on this interface is affected if the policy is modified:

```
show policy-map targets

Fri Jul 16 16:38:24.789 DST
1) Policymap: policy1    Type: qos
     Targets (applied as main policy):
       GigabitEthernet0/1/0/0 output
     Total targets: 1

     Targets (applied as child policy):
     Total targets: 0
```

Additional References

The following sections provide references related to implementing packet classification.

Related Documents

Related Topic	Document Title
Initial system bootup and configuration	*Cisco ASR 9000 Series Aggregation Services Router Getting Started Guide*
Master command reference	*Cisco ASR 9000 Series Aggregation Services Router Master Command Listing*
QoS commands	*Cisco ASR 9000 Series Aggregation Services Router Modular Quality of Service Command Reference*
User groups and task IDs	"Configuring AAA Services on Cisco ASR 9000 Series Router" module of *Cisco Cisco ASR 9000 Series Aggregation Services Router System Security Configuration Guide*

Standards

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

MIBs

MIBs	MIBs Link
—	To locate and download MIBs using Cisco IOS XR software, use the Cisco MIB Locator found at the following URL and choose a platform under the Cisco Access Products menu: http://cisco.com/public/sw-center/netmgmt/cmtk/mibs.shtml

RFCs

RFCs	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 Technical Support website contains thousands of pages of searchable technical content, including links to products, technologies, solutions, technical tips, and tools. Registered Cisco.com users can log in from this page to access even more content.	http://www.cisco.com/techsupport