# QoS Overview

Quality of Service (QoS) is the technique of prioritizing traffic flows and providing preferential forwarding for higher-priority packets. The fundamental reason for implementing QoS in your network is to provide better service for certain traffic flows. A traffic flow can be defined as a combination of source and destination addresses, source and destination socket numbers, and the session identifier. A traffic flow can more broadly be described as a packet moving from an incoming interface that is destined for transmission to an outgoing interface. The traffic flow must be classified, and prioritized on all routers and passed along the data forwarding path throughout the network to achieve end-to-end QoS delivery. The terms *traffic flow* and *packet* are used interchangeably throughout this module.

This module contains overview information about modular QoS features within a service provider network.



# Information About Modular Quality of Service Overview

Before configuring modular QoS on your network, you must understand these concepts:

# QoS Techniques

QoS on Cisco IOS XR relies on these techniques to provide end-to-end QoS delivery across a heterogeneous network:

- QoS classification - classification techniques identify the traffic flow, and provide the capability to partition network traffic into multiple priority levels or classes of service. Identification of a traffic flow can be performed by using several methods within a single router, such as IP precedence, IP differentiated service code point (DSCP), MPLS EXP bit, or Class of Service (CoS).

- Qos Marking - after classification, the traffic is marked to indicate the required level of QoS for that traffic. Packets marked as priority are met with preferential treatment.
- QoS Policing - allows the user to control the maximum rate of traffic sent or received on an interface and to partition a network into multiple priority levels or class of service (CoS).
- Queueing - is a congestion management technique. Cisco NCS 4000 supports default queue selection for layer2 and layer3. Use the set traffic-class command (ingress side) to explicitly choose a queue and override the default queue selection.
- H-QoS - allows the user to specify QoS behavior at multiple policy levels, which provides a high degree of granularity in traffic management.
- Dual Policy - enables support for two output policies.
- Congestion avoidance - includes techniques that monitor network traffic flows, in an effort to anticipate and avoid congestion at common network and internetwork bottlenecks before problems occur.

Before implementing the QoS features for these techniques, you should identify and evaluate the traffic characteristics of your network because not all techniques are appropriate for your network environment.

# General QoS Terminology

This section provides a summary of the frequently used QoS terminologies.

- Dropping technologies (Tail Drop and WRED) - Tail drop is a congestion avoidance technique that drops packets when an output queue is full until congestion is eliminated. WRED drops packets selectively based on IP precedence.
- Shapers and policers - are needed to ensure that a packet adheres to a contract and service. A policer typically drops traffic flow, when the traffic flow exceeds the policer rate. A shaper delays excess traffic flow using a buffer, or queuing mechanism, to hold the traffic for transmission at a later time.
- DEI - incase of congestion, a packet marked with DEI (Drop Eligible Indicator) is dropped.
- DSCP - the DSCP (Differentiated Services Code Point) bit in the IP header is used for packet classification.

# Modular QoS Command-Line Interface

QoS features are enabled through the Modular QoS command-line interface (MQC) feature. The *MQC* is a command-line interface (CLI) structure that allows you to create policies and attach these policies to interfaces. A traffic policy contains a traffic class and one or more QoS features. A traffic class is used to classify traffic, whereas the QoS features in the traffic policy determine how to treat the classified traffic. One of the main goals of MQC is to provide a platform-independent interface for configuring QoS across Cisco platforms.

# Traffic Class Elements

The purpose of a traffic class is for queuing and classification of traffic on the 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*.

**Note** The **class-map** command supports the **match-any** keyword only. The **match-all** keyword is not supported.

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. .

This table shows the details of the match types supported on the Cisco 4000 Series router.

| Supported Match Type | Min, Max | Max Entries | Support for Ranges | Direction supported for interfaces |
|---|---|---|---|---|
| IPv4 DSCP<br>DSCP | (0, 63) | 64 | yes | Ingress |
| IPv4 Precedence<br>Precedence | (0,7) | 8 | no | Ingress |
| MPLS Experimental Topmost | (0,7) | 8 | no | Ingress |
| QoS-group | (1,7) | 7 | no | Egress |

# 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 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 **class-map** command is used to define a traffic class and the associated rules that match packets to the class.

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 8 traffic classes can be associated to a single traffic policy. The 8 class maps include the default class and the classes of the child policies, if any.

# 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.

# Creating a Traffic Policy

To create a traffic policy (supported on egress), use the **policy-map** 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.

These class-actions are supported:

- bandwidth—Configures the bandwidth for the class.
- bandwidth remaining ratio—Configures the bandwidth remaining ratio for the class.
- police—Police traffic.
- priority—Assigns priority to the class.
- queue-limit—Configures queue-limit (tail drop threshold) for the class.
- random-detect—Enables Random Early Detection.
- service-policy—Configures a child service policy.
- set—Configures marking for this class.
- shape—Configures shaping for the class.

### Restrictions

A maximum of 8 classes for Level 1 and 1 for Level 2.

### Procedure

**Step 1** **configure**

**Step 2** **policy-map** [ **type qos** ] *policy-name*

**Example:**

```
RP/0/(config)# policy-map policy1
```

Creates or modifies a policy map that can be attached to one or more interfaces to specify a service policy and enters the policy map configuration mode.

**Step 3** **class** class-name

**Example:**

```
RP/0/(config-pmap)# class class1
```

Specifies the name of the class whose policy you want to create or change.

**Step 4** **commit**

---

#### Running configuration example for Creating a traffic policy

```
policy-map ingress_POLICER_POLICY
class CLASS_1_POLICERIPV4PREC
  set traffic-class 7
!
```

# 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).

#### Prerequisites

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

#### Procedure

---

**Step 1** **configure**

**Step 2** **interface** *type interface-path-id*

**Example:**

```
RP/0/(config)# interface HundredGigE 0/7/0/1
```

Configures an interface and enters the interface configuration mode.

**Step 3** **service-policy** {**input** | **output**} *policy-map*

**Example:**

```
RP/0/(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** **commit**

**Step 5** **show policy-map interface** *type interface-path-id* [**input** | **output**]

**Example:**

```
RP/0/# show policy-map interface HundredGigE 0/7/0/1
```

(Optional) Displays statistics for the policy on the specified interface.

**Running configuration example for attaching a traffic policy to an interfac e**

```
interface TenGigE0/5/0/1/3.100
service-policy input ingress_POLICER_POLICY
ipv4 address 10.0.0.1 255.255.255.0
encapsulation dot1q 100
!
```

# 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. A modified policy is subjected 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. 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.

You can also 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.

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

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.

**Verification**

If unrecoverable errors occur during in-place policy modification, the policy is put into an inconsistent state on target interfaces. No new configuration is possible until the configuration session is unblocked. It is recommended to remove the policy from the interface, check the modified policy and then re-apply accordingly.

# Recommendations for Using In-Place Policy Modification

For a short period of time while a QoS policy is being modified, no QoS policy is active on the interface. In the unlikely event that the QoS policy modification and rollback both fail, the interface is left without a QoS policy.

For these reasons, it is best to 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.

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.