Classify Packets to Identify Specific Traffic

Classify Packets to Identify Specific Traffic

Read this section to get an overview of packet classification and the different packet classification types for your router.

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.

When traffic descriptors are used to classify traffic, the source agrees to adhere to the contracted terms and the network promises a quality of service. This is where traffic policers and traffic shapers come into the picture. Traffic policers and traffic shapers use the traffic descriptor of a packet—that is, its classification—to ensure adherence to the contract.

The Modular Quality of Service (QoS) command-line interface (MQC) is used to define the traffic flows that must be classified, where each traffic flow is called a class of service, or class. Later, a traffic policy is created and applied to a class. All traffic not identified by defined classes fall into the category of a default class.

QoS Support for VPWS

Table 1: Feature History Table

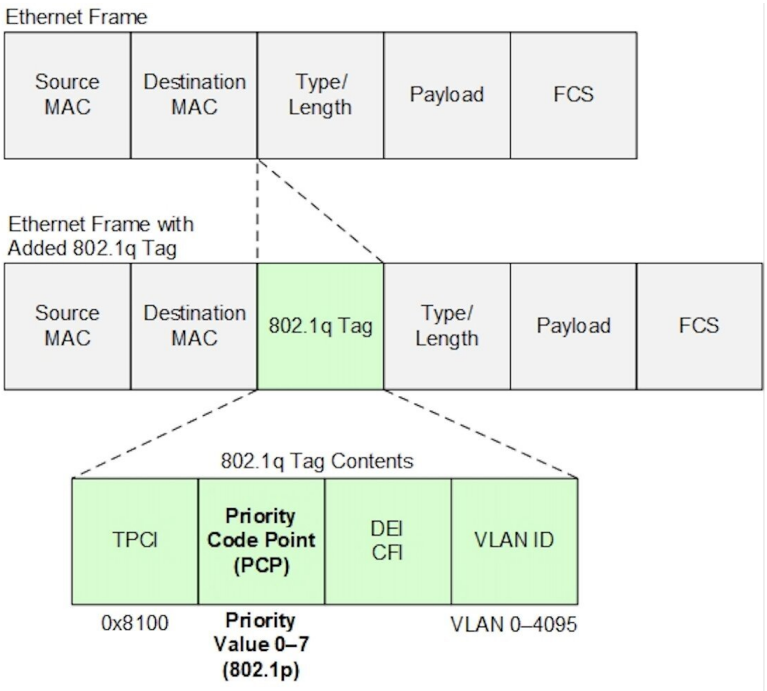
Feature Name	Release Information	Feature Description
QoS Support for VPWS	Release 7.7.1	Now, you can apply QoS packet classification on Layer 2 subinterfaces, and for VPWS traffic. This feature provides ingress classification support for L2 attachment circuit (AC) traffic based on the 802.1p field priority value. With this feature, you can use QoS policies on VPWS traffic in your network on Cisco 8000 Series routers.

EVPN-VPWS is a BGP control plane solution for point-to-point services. It implements the signaling and encapsulation techniques for establishing an EVPN instance between a pair of PEs. It has the ability to forward traffic from one network to another without a MAC lookup.

By configuring QoS, you can provide preferential treatment to specific types of traffic at the expense of other traffic types. Without QoS, the device offers best-effort service for each packet, regardless of the packet contents or size. The device sends the packets without any assurance of reliability, delay bounds, or throughput.

The 802.1Q standard (and 802.1ad, a subset of 802.1Q) defines a system of VLAN tagging for Ethernet frames and also contains a provision for a QoS prioritization scheme known as 802.1p, which indicates the priority level of the frame, as shown in the figure. For more information on IEEE standards, browse the IEEE website.

Figure 1: 802.1p



VLAN Tag Priority Field Based Classification

The QoS term class of service (CoS) is a 3-bit field called Priority Code Point (PCP) which specifies a priority value between 0 and 7 that is used by QoS to differentiate traffic. Drop Eligible Indicator (DEI) is a 1-bit field that is used to indicate frames eligible to be dropped during traffic congestion.

MQC allows configuration of class-map match condition based on the PCP and DEI fields. The classification is supported on 802.1Q and 802.1ad interfaces.

Note The ingress classification supports AC-to-AC traffic flow and AC-to-PWE traffic flow.

Configuration Example for QoS Support for VPWS

Follow these steps to enable this feature:

- Enable VPWS configuration. Refer the L2VPN Configuration Guide for details.
- Configure ingress traffic classification, based on the PCP and DEI fields in the VLAN header.
  - Create class maps for different traffic classes.
  - Associate them with a policy map.
- Configure ingress traffic remarking.

Configure Ingress Traffic Classification Based on PCP and DEI Fields in the VLAN Header

```
/* Create class maps for different traffic classes */

Router# configure terminal
Router(config)# class-map match-all CONTROL_PLANE
Router(config-cmap)# match cos 7
Router(config-cmap)# end-class-map

Router(config)# class-map match-all VOIP
Router(config-cmap)# match cos 6
Router(config-cmap)# end-class-map

Router(config)# class-map match-all VIDEO_STREAM
Router(config-cmap)# match cos 5
Router(config-cmap)# end-class-map

Router(config)# class-map match-all TRANSACTIONAL_DATA
Router(config-cmap)# match cos 4
Router(config-cmap)# end-class-map

Router(config)# class-map match-all DB_SYNC
Router(config-cmap)# match cos 3
Router(config-cmap)# match dei 1
Router(config-cmap)# end-class-map

Router(config)# class-map match-all BULK_DATA
Router(config-cmap)# match cos 2
Router(config-cmap)# match dei 1
Router(config-cmap)# end-class-map

Router(config)# class-map match-all SCAVENGER
Router(config-cmap)# match cos 1
Router(config-cmap)# match dei 1
```

```
Router(config-cmap)# end-class-map
Router(config)# commit
```

```
/* Create a policy map and associate the class maps to it */

Router# configure terminal
Router(config)# policy-map INGRESS_L2_AC
Router(config-pmap)# class CONTROL_PLANE
Router(config-pmap-c)# set traffic-class 7
Router(config-pmap-c)# exit

Router(config-pmap)# class VOIP
Router(config-pmap-c)# set traffic-class 6
Router(config-pmap-c)# exit

Router(config-pmap)# class VIDEO_STREAM
Router(config-pmap-c)# set traffic-class 5
Router(config-pmap-c)# exit

Router(config-pmap)# class TRANSACTIONAL_DATA
Router(config-pmap-c)# set traffic-class 4
Router(config-pmap-c)# exit

Router(config-pmap)# class DB_SYNC
Router(config-pmap-c)# set traffic-class 3
Router(config-pmap-c)# exit

Router(config-pmap)# class BULK_DATA
Router(config-pmap-c)# set traffic-class 2
Router(config-pmap-c)# exit

Router(config-pmap)# class SCAVENGER
Router(config-pmap-c)# set traffic-class 1
Router(config-pmap-c)# exit

Router(config-pmap)# class class-default
Router(config-pmap-c)# exit
Router(config-pmap)# end-policy-map
Router(config)# commit
```

Configure Ingress Traffic Remarking

```
/* Set CoS and DEI values for traffic classes, as needed */

Router# configure terminal
Router(config)# policy-map INGRESS_L2_AC
Router(config-pmap)# class CONTROL_PLANE
Router(config-pmap-c)# set traffic-class 7
Router(config-pmap-c)# exit

Router(config-pmap)# class VOIP
Router(config-pmap-c)# set traffic-class 6
Router(config-pmap-c)# set cos 7
Router(config-pmap-c)# exit

Router(config-pmap)# class VIDEO_STREAM
Router(config-pmap-c)# set traffic-class 5
Router(config-pmap-c)# set cos 5
Router(config-pmap-c)# exit

Router(config-pmap)# class TRANSACTIONAL_DATA
Router(config-pmap-c)# set traffic-class 4
Router(config-pmap-c)# set cos 5
Router(config-pmap-c)# exit
```

```
Router(config-pmap)# class DB_SYNC
Router(config-pmap-c)# set traffic-class 3
Router(config-pmap-c)# set dei 0
Router(config-pmap-c)# exit

Router(config-pmap)# class BULK_DATA
Router(config-pmap-c)# set traffic-class 2
Router(config-pmap-c)# set cos 3
Router(config-pmap-c)# set dei 0
Router(config-pmap-c)# exit

Router(config-pmap)# class SCAVENGER
Router(config-pmap-c)# set traffic-class 1
Router(config-pmap-c)# exit

Router(config-pmap)# class class-default
Router(config-pmap-c)# set dei 1
Router(config-pmap-c)# end-policy-map
Router(config)# commit
```

/* Associate Policy-Map INGRESS_L2_AC With the Designated Subinterface */

Before you enable the subinterface, ensure that the parent interface state is up.

```
Router(config)# interface hundredGigE 0/11/0/31.102
Router(config-subif)# service-policy input INGRESS_L2_AC
Router(config-subif)# commit
```

Verification

Verify ingress QoS policy configuration. In the output, you can see that traffic is classified and transmitted for some categories.

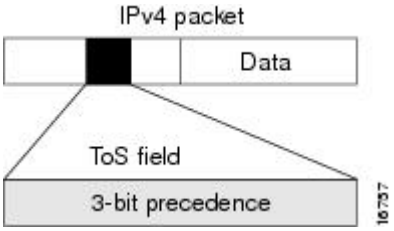
```
Router# show policy-map interface Hu0/11/0/31.102 input

HundredGigE0/11/0/31.102 input: INGRESS_L2_AC

Class CONTROL_PLANE
  Classification statistics          (packets/bytes)     (rate - kbps)
 Matched             :             9813350/13738690000          936847
    Transmitted         :              9813350/13738690000          936847
    Total Dropped       :                    0/0                    0
Class VOIP
  Classification statistics          (packets/bytes)     (rate - kbps)
 Matched             :           117760245/164864343000         11242157
    Transmitted         :            117760245/164864343000         11242157
    Total Dropped       :                    0/0                    0
Class VIDEO_STREAM
  Classification statistics          (packets/bytes)     (rate - kbps)
 Matched             :            49066792/68693508800          4684233
    Transmitted         :             49066792/68693508800          4684233
    Total Dropped       :                    0/0                    0
Class TRANSACTIONAL_DATA
  Classification statistics          (packets/bytes)     (rate - kbps)
 Matched             :           225707344/315990281600         21547467
    Transmitted         :            225707344/315990281600         21547467
    Total Dropped       :                    0/0                    0
Class DB_SYNC
  Classification statistics          (packets/bytes)     (rate - kbps)
    Matched             :                    0/0                    0
    Transmitted         :                    0/0                    0
    Total Dropped       :                    0/0                    0
```

```
Class BULK_DATA
  Classification statistics          (packets/bytes)     (rate - kbps)
    Matched             :                  0/0                   0
    Transmitted         :                  0/0                   0
    Total Dropped       :                  0/0                   0
Class SCAVENGER
  Classification statistics          (packets/bytes)     (rate - kbps)
    Matched             :                  0/0                   0
    Transmitted         :                  0/0                   0
    Total Dropped       :                  0/0                   0
Class class-default
  Classification statistics          (packets/bytes)     (rate - kbps)
 Matched             :           500482413/700675378200          47779164
    Transmitted         :           500482413/700675378200          47779164
    Total Dropped       :                  0/0                   0
Policy Bag Stats time: 1657528790084  [Local Time: 07/11/22 08:39:50.084]
```

Specification of the CoS for a Packet with IP Precedence

Use of IP precedence allows you to specify the CoS for a packet. You can create differentiated service by setting precedence levels on incoming traffic and using them in combination with the QoS queuing features. 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.

Figure 2: IPv4 Packet Type of Service Field



You can use the three precedence bits in the type-of-service (ToS) field of the IPv4 header for this purpose. 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, queuing features such as LLQ can use the IP precedence setting of the packet to prioritize traffic.

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. 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.

Each precedence corresponds to a name. IP precedence bit settings 6 and 7 are reserved for network control information, such as routing updates. These names are defined in RFC 791.

IP Precedence Value Settings

By default, the routers leave 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 and LLQ features can use the IP precedence bits.

IP Precedence Compared to IP DSCP Marking

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.

Packet Classification on Your Router

On your router, there are two types of packet classification systems:

- In the ingress direction, QoS map and Ternary Content Addressable Memory (TCAM).

Note TCAM is not supported on fixed-configuration routers (where router interfaces are built in). It is supported only on modular routers (that have multiple slots that allow you to change the interfaces on the router).

- In the egress direction, QoS map.

When a policy is matching only on Differentiated Services Code Point (DSCP) or precedence value (also called DSCP or Precedence-based classification), the system selects map-based classification system; else, it selects TCAM.

The TCAM is an extension of the Content Addressable Memory (CAM) table concept. A CAM table takes in an index or key value (usually a MAC address) and looks up the resulting value (usually a switch port or VLAN ID). Table lookup is fast and always based on an exact key match consisting of two input values: 0 and 1 bits.

The QoS map is a table-based classification system for traffic packets.

Classify and Remark Layer 3 Header on Layer 2 Interfaces

When you need to mark packets for Layer 2 interface traffic that flows across bridge domains and bridge virtual interfaces (BVIs), you can create a mixed QoS policy. This policy has both map-based and TCAM-based classification class-maps. The mixed policy ensures that both bridged (Layer 2) and Bridge Virtual Interface (BVI, or Layer 3) traffic flows are classified and remarked.

Guidelines

- A class-map with TCAM classification may not match bridged traffic. TCAM entries match only routed traffic while map entries match both bridged and BVI traffic.
- A class-map with map-based classification matches both bridged and BVI traffic.

Example

```
ipv4 access-list acl_v4
10 permit ipv4 host 100.1.1.2 any
20 permit ipv4 host 100.1.100.2 any
ipv6 access-list acl_v6
10 permit tcp host 50:1:1::2 any
20 permit tcp any host 50:1:200::2
class-map match-any c_match_acl
match access-group ipv4 acl_v4 ! This entry does not match bridged traffic
match access-group ipv6 acl_v6 ! This entry does not match bridged traffic
match dscp af11 This entry matches bridged and BVI traffic
class-map match-all c_match_all
match protocol udp ! This entry does not match bridged traffic
match prec 7
class-map match-any c_match_protocol
match protocol tcp ! This entry, and hence this class does not match bridged traffic
class-map match-any c_match_ef
match dscp ef ! This entry/class matches bridged and BVI traffic
class-map match-any c_qosgroup_1 This class matches bridged and BVI traffic
!
match qos-group 1
policy-map p_ingress
class c_match_acl
set traffic-class 1
set qos-group 1
!
class c_match_all
set traffic-class 2
set qos-group 2
!
class c_match_ef
set traffic-class 3
set qos-group 3
!
class c_match_protocol
set traffic-class 4
set qos-group 4
policy-map p_egress
class c_qosgroup_1
set dscp af23
interface FourHundredGigE0/0/0/0
l2transport
service-policy input p_ingress
service-policy output p_egress
!
```

```
!
interface FourHundredGigE0/0/0/1
ipv4 address 200.1.2.1 255.255.255.0
ipv6 address 2001:2:2::1/64
service-policy input p_ingress
service-policy output p_egress
```

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 - to specify various criteria for classifying packets.
- An instruction on how to evaluate these **match** commands (if more than one **match** command exists in the traffic class)

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 match types supported on the router.

Match Type Supported	Min, Max	Max Entries	Support for Match NOT	Support for Ranges	Direction Supported on Interfaces
IPv4 DSCP IPv6 DSCP	(0,63)	64	Yes	Yes	Ingress
DSCP					Egress
IPv4 Precedence IPv6 Precedence	(0,7)	8	Yes	No	Ingress
Precedence					Egress
MPLS Experimental Topmost	(0,7)	8	Yes	No	Ingress Egress
Access-group	Not applicable	8	No	Not applicable	Ingress
Match qos-group	(1-31)	7 + class-default	No	No	Egress
Protocol	(0, 255)	1	Yes	Not applicable	Ingress

Match Type Supported	Min, Max	Max Entries	Support for Match NOT	Support for Ranges	Direction Supported on Interfaces
CoS	(0,7)	8	Yes	No	Ingress and Egress
DEI	(0,1)	2	Yes	No	Ingress and Egress

Default Traffic Class

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

If the user doesn't 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.

For egress classification, match on **qos-group** for seven groups with range (1-31) is supported. Match **qos-group 0** can't be configured. The class-default in the egress policy maps to **qos-group 0**.

This example shows how to configure a traffic policy for the default class:

```
configure
 policy-map ingress_policy1
 class class-default
  police rate percent 30
 !
```

Create 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 **match** commands in class-map configuration mode, as needed.

Guidelines

- Users can provide multiple values for a match type in a single line of configuration; that is, if the first value does not meet the match criteria, then the next value indicated in the match statement is considered for classification.
- Use the **not** keyword with the **match** command to perform a match based on the values of a field that are not specified.
- All **match** commands specified in this configuration task are considered optional, but you must configure at least one match criterion for a class.
- If you specify **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.
- For the **match access-group** command, QoS classification based on the packet length or TTL (time to live) field in the IPv4 and IPv6 headers is not supported.

- For the **match access-group** command, when an ACL list is used within a class-map, the deny action of the ACL is ignored and the traffic is classified based on the specified ACL match parameters.
- The **match qos-group**, **traffic-class**, **DSCP/Prec**, and **MPLS EXP** are supported only in egress direction, and these are the only match criteria supported in egress direction
- The egress default class implicitly matches **qos-group** 0.
- Multicast takes a system path that is different than unicast on router, and they meet later on the egress in a multicast-to-unicast ratio of 20:80 on a per interface basis. This ratio is maintained on the same priority level as that of the traffic.
- Egress QoS for multicast traffic treats traffic classes 0-5 as low-priority and traffic classes 6-7 as high priority. Currently, this is not user-configurable.
- Egress shaping does not take effect for multicast traffic in the high priority (HP) traffic classes. It only applies to unicast traffic.
- If you set a traffic class at the ingress policy and do not have a matching class at egress for the corresponding traffic class value, then the traffic at ingress with this class will not be accounted for in the default class at the egress policy map.
- Only traffic class 0 falls in the default class. A non-zero traffic class assigned on ingress but with no assigned egress queue, falls neither in the default class nor any other class.

Configuration Example

You have to accomplish the following to complete the traffic class configuration:

1. Creating a class map
2. Specifying the match criteria for classifying the packet as a member of that particular class

   (For a list of supported match types, see Traffic Class Elements, on page 9.)

```
Router# configure
Router(config)# class-map match-any qos-1
Router(config-cmap)# match qos-group 1
Router(config-cmap)# end-class-map
Router(config-cmap)# commit
```

Use this command to verify the class-map configuration:

```
Router#show class-map qos-1
1) ClassMap: qos-1    Type: qos
    Referenced by 2 Policymaps
```

Also see, Attach a Traffic Policy to an Interface, on page 13.

Related Topics

- Traffic Class Elements, on page 9
- Traffic Policy Elements, on page 12

Traffic Policy Elements

A traffic policy contains three elements:

- Name
- Traffic class
- QoS policies

After choosing the traffic class that is used to classify traffic to the traffic policy, the user can enter the QoS features to be applied to the classified traffic.

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

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 router supports 8 classes per policy-map in the ingress direction and 8 classes per policy-map in the egress direction.

This table shows the supported class-actions on the router.

Supported Action Types	Direction supported on Interfaces
bandwidth-remaining	egress
mark	See Packet Marking
police	ingress
priority	egress (level 1 to level 7)
queue-limit	egress
shape	egress
red	egress

RED supports the **discard-class** option; the only values to be passed to the discard-class being 0 and 1.

Create a Traffic Policy

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.

To configure a traffic class, see Create a Traffic Class, on page 10.

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. With dual policy support, you can have two traffic policies, one marking and one queuing attached at the output. See, Attach a Traffic Policy to an Interface, on page 13.

Configuration Example

You have to accomplish the following to complete the traffic policy configuration:

1. Creating a policy map that can be attached to one or more interfaces to specify a service policy
2. Associating the traffic class with the traffic policy
3. Specifying the class-action(s) (see Traffic Policy Elements, on page 12)

```
Router# configure
Router(config)# policy-map  test-shape-1
Router(config-pmap)# class qos-1

/* Configure class-action ('shape' in this example).
Repeat as required, to specify other class-actions */
Router(config-pmap-c)# shape average percent 40
Router(config-pmap-c)# exit

/* Repeat class configuration as required, to specify other classes */

Router(config-pmap)# end-policy-map
Router(config)# commit
```

Related Topics

- Traffic Policy Elements, on page 12
- Traffic Class Elements, on page 9

Attach a Traffic Policy to an Interface

After the traffic class and the traffic policy are created, you must attach the traffic policy to interface, and specify the direction in which the policy should be applied.

Note Hierarchical policies are not supported.

When a policy-map is applied to an interface, the transmission rate counter of each class is not accurate. This is because the transmission rate counter is calculated based on the exponential decay filter.

Configuration Example

You have to accomplish the following to attach a traffic policy to an interface:

1. Creating a traffic class and the associated rules that match packets to the class (see Create a Traffic Class, on page 10)
2. Creating a traffic policy that can be attached to one or more interfaces to specify a service policy (see Create a Traffic Policy, on page 12)
3. Associating the traffic class with the traffic policy
4. Attaching the traffic policy to an interface, in the ingress or egress direction

```
Router# configure
Router(config)#  interface fourHundredGigE 0/0/0/2
Router(config-int)# service-policy output strict-priority
Router(config-int)# commit
```

Running Configuration

```
/* Class-map configuration */

class-map match-any traffic-class-7
 match traffic-class 7
 end-class-map

!class-map match-any traffic-class-6
 match traffic-class 6
 end-class-map

class-map match-any traffic-class-5
 match traffic-class 5
 end-class-map

class-map match-any traffic-class-4
 match traffic-class 4
 end-class-map

class-map match-any traffic-class-3
 match traffic-class 3

class-map match-any traffic-class-2
 match traffic-class 2
 end-class-map

class-map match-any traffic-class-1
 match traffic-class 1
 end-class-map


/* Traffic policy configuration */

policy-map test-shape-1
 class traffic-class-1
  shape average percent 40
 !

policy-map strict-priority
 class tc7
  priority level 1
  queue-limit 75 mbytes
 !
 class tc6
  priority level 2
  queue-limit 75 mbytes
 !
 class tc5
  priority level 3
  queue-limit 75 mbytes
 !
 class tc4
  priority level 4
  queue-limit 75 mbytes
```

```
 !
 class tc3
  priority level 5
  queue-limit 75 mbytes
 !
 class tc2
  priority level 6
  queue-limit 75 mbytes
 !
 class tc1
  priority level 7
  queue-limit 75 mbytes
 !
 class class-default
  queue-limit 75 mbytes
 !
 end-policy-map

- - -
- - -

/* Attaching traffic policy to an interface in egress direction */
interface fourHundredGigE 0/0/0/2
 service-policy output strict-priority
 !
```

Verification

```
Router# #show qos int fourHundredGigE 0/0/0/2 output

NOTE:- Configured values are displayed within parentheses Interface FourHundredGigE0/0/0/2
 ifh 0xf0001c0  -- output policy

NPU Id:                        0
Total number of classes:       8
Interface Bandwidth:           400000000 kbps
Policy Name:                   strict-priority
VOQ Base:                      2400
Accounting Type:               Layer1 (Include Layer 1 encapsulation and above)
------------------------------------------------------------------------------
Level1 Class (HP1)                       =   tc7
Egressq Queue ID                         =   2407 (HP1 queue)
Queue Max. BW.                           =   no max (default)
TailDrop Threshold                       =   74999808 bytes / 2 ms (75 megabytes)
WRED not configured for this class

Level1 Class (HP2)                       =   tc6
Egressq Queue ID                         =   2406 (HP2 queue)
Queue Max. BW.                           =   no max (default)
TailDrop Threshold                       =   74999808 bytes / 2 ms (75 megabytes)
WRED not configured for this class

Level1 Class (HP3)                       =   tc5
Egressq Queue ID                         =   2405 (HP3 queue)
Queue Max. BW.                           =   no max (default)
TailDrop Threshold                       =   74999808 bytes / 2 ms (75 megabytes)
WRED not configured for this class

Level1 Class (HP4)                       =   tc4
Egressq Queue ID                         =   2404 (HP4 queue)
Queue Max. BW.                           =   no max (default)
TailDrop Threshold                       =   74999808 bytes / 2 ms (75 megabytes)
```

```
WRED not configured for this class

Level1 Class (HP5)                      =   tc3
Egressq Queue ID                        =   2403 (HP5 queue)
Queue Max. BW.                          =   no max (default)
TailDrop Threshold                      =   74999808 bytes / 2 ms (75 megabytes)
WRED not configured for this class

Level1 Class (HP6)                      =   tc2
Egressq Queue ID                        =   2402 (HP6 queue)
Queue Max. BW.                          =   no max (default)
TailDrop Threshold                      =   74999808 bytes / 2 ms (75 megabytes)
WRED not configured for this class

Level1 Class (HP7)                      =   tc1
Egressq Queue ID                        =   2401 (HP7 queue)
Queue Max. BW.                          =   no max (default)
TailDrop Threshold                      =   74999808 bytes / 2 ms (75 megabytes)
WRED not configured for this class

Level1 Class                            =   class-default
Egressq Queue ID                        =   2400 (Default LP queue)
Queue Max. BW.                          =   no max (default)
Inverse Weight / Weight                 =   1 / (BWR not configured)
TailDrop Threshold                      =   74999808 bytes / 150 ms (75 megabytes)
WRED not configured for this class

!
```

Related Topics

- Traffic Policy Elements, on page 12
- Traffic Class Elements, on page 9

QoS Policy Propagation via BGP (QPPB)

Table 2: Feature History Table

Feature Name	Release Information	Feature Description
QoS Policy Propagation via BGP	Release 7.5.2	You now have the ability to install a BGP route in the routing table with a QoS Group so that IP packets that match the route receive the QoS policies associated with the QoS group. This functionality enables convenient classification and marking when BGP is deployed, overcoming the administrative challenges of classifying based on ACLs.

QoS Policy Propagation via Border Gateway Protocol (QPPB via BGP) is a mechanism that allows propagation of quality of service (QoS) policy and classification by the sending party that is based on the following:

- Access lists
- Community lists
- Autonomous system paths in the BGP

QPPB thus helps in classification that is based on the destination address instead of the source address.

QoS policies that differentiate between different types of traffic are defined for a single enterprise network. For instance, one enterprise may want to treat important web traffic, not-important web traffic, and all other data traffic as three different classes. And thereafter, use the different classes for the voice and video traffic.

Hence, QPPB overcomes:

- the administrative challenges of classifying that is based on ACLs.
- the administrative problems of just listing the networks that need premium services.

QPPB allows marking of packets that are based on QoS group value that is associated with a Border Gateway Protocol (BGP) route.

Benefits of QPPB

- QPPB provides an IP prefix-based QoS capability.
- Traffic to IP addresses that have specific IP prefixes can be prioritized above other IP addresses.
- IP prefixes of interest are tagged through the control plane that uses common BGP route-map techniques, including the community attribute.
- Traffic to the tagged BGP prefixes is then classified and prioritized via the data forwarding plane by using the IOS-XR MQC (Modular QoS CLI) mechanisms, such as remarking.
- QPPB provides the glue between the BGP control plane and the IP data forwarding plane in support of IP prefix-based QoS.
- BGP configuration within QPPB uses a table map to match specific prefixes learned through BGP neighbors, and then sets the router's local QoS Group variable that is maintained within the Forwarding Information Base (FIB) for those specific prefixes.

Guidelines and Limitations

- While configuring QPPB in the route policy for BGP, the range of the QoS group value that you can set using the **set qos-group** command is between 0 through 7.
- IP precedence-based QPPB isn't supported; only QoS group-based QPPB is supported.
- The order of precedence among ACL features (in ascending order):

  With QoS-based group policy:

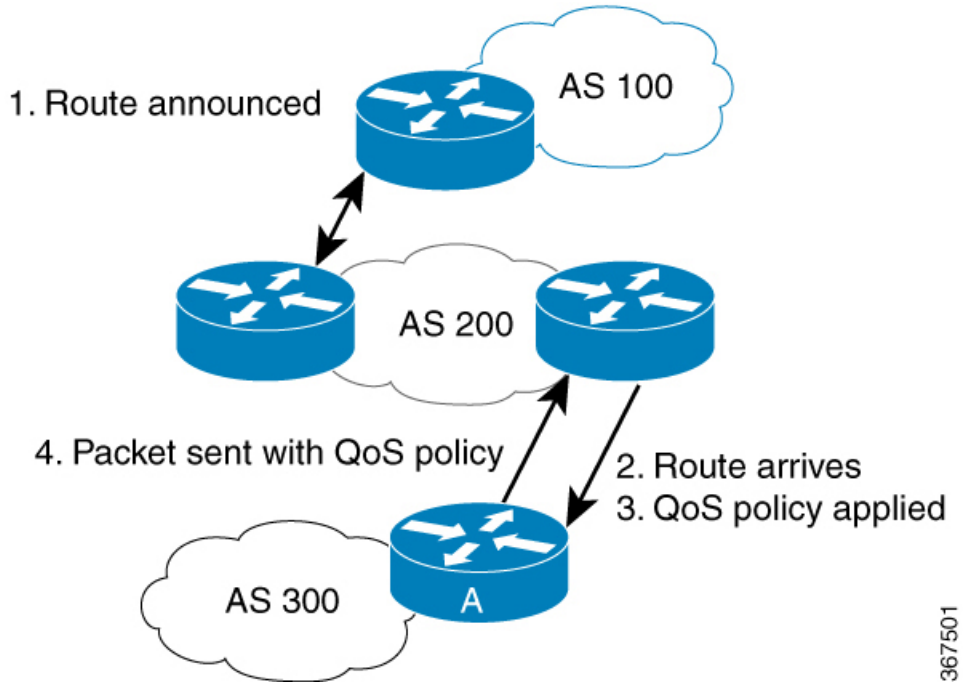
  - Security ACL (0), BGP-FS (1), QPPB (2), Peering QoS (3)

- **With ACL-based policy:**

  - Security ACL (0), ACL based QoS (1), BGP-FS (2), QPPB (3)

- You can configure QPPB on the following interface types:
  - Physical
  - Bundle
  - Physical subinterface
  - Bundle subinterface
- QPPB is not supported on Q100-based routers or line cards.
- QPPB overwrites the QoS Group value set by Security ACL.
- If you configure ACL-based Switched Port Analyzer (SPAN) on an interface, you can't enable QPPB on that interface. To enable QPPB, remove the ACL-based SPAN from the interface.
- If you configure QPPB on an interface, you can't enable ACL-based SPAN on that interface. To enable ACL-based SPAN, remove QPPB from the interface.
- Peering QoS overwrites the QoS Group value set by QPPB.
- Peering QoS and QPPB overwrite BGP FlowSpec actions except setting Drop and Policer. Currently, BGP FlowSpec sets DSCP and Meter.
- Remarking support is the same that QoS currently supports.
- Ingress policing support is the same that QoS currently supports.
- Egress policing is not supported.
- The router supports a subset of full QPPB options—only IP destination prefix mode on input policy is supported.

Figure 3: Sample Scenario



Router A learns routes from AS 200 and AS 100. QoS policy is applied to any ingress interface of Router A to match the defined route maps with destination prefixes of incoming packets. Matching packets on Router A to AS 200 or AS 100 are sent with the appropriate QoS policy from Router A.

BGP maintains a scalable database of destination prefixes, QPPB, by using BGP table maps. BGP adds the ability to map a qos-group value to desired IP destinations. These qos-group values are used in QoS policies applied locally on ingress interfaces. Whenever a packet that is bound for such destinations is encountered, the qos-group value matching that destination route looks up with work inside the policy classmap, and marks that packet for any configured policy actions.

Configuration Workflow

Use the following configuration workflow for QPPB:

- Define routing policy.
- Put routing policy at table-policy attach point under BGP.
- Define classmaps and ingress policy to use the qos-groups that are used in table-policy.
- Enable IPv4 or IPv6 QPPB configuration under the desired interfaces.

Define Routing Policy

A routing policy instructs the router to inspect routes, filter them, and potentially modify their attributes as they are accepted from a peer, advertised to a peer, or redistributed from one routing protocol to another.

The routing policy language (RPL) provides a language to express routing policy. You must set up destination prefixes either to match inline values or one of a set of values in a prefix set.

Example:

```
prefix-set prefix-list-v4
    70.1.1.1,
    70.2.1.0/24,
    70.2.2.0/24 ge 28,
    70.2.3.0/24 le 28
end-set
prefix-set prefix-list-v6
    2001:300::2,
    2003:200::3
end-set

route-policy qppb1
    if destination in (60.60.0.2/24) then
        set qos-group 5
    elseif destination in prefix-list-v4 then
        set qos-group 4
    else
        set qos-group 1
     pass
endif
end-policy
route-policy qppb2
    if destination in prefix-list-v6 then
        set qos-group 5
    elseif destination in (2001:300::2) then
        set qos-group 4
    else
        set qos-group 1
    pass
endif
end-policy
```

Put Routing Policy at table-policy Attach Point Under BGP

The table-policy attach point permits the routing policy to perform actions on each route as they are installed into the RIB routing table. QPPB uses this attachment point to intercept all routes as they are received from peers. Ultimately the RIB will update the FIB in the hardware forwarding plane to store destination prefix routing entries, and in cases where table policy matches a destination prefix, the qos-group value is also stored with the destination prefix entry for use in the forwarding plane.

Example:

```
router bgp 900
    [vrf <name>]
    bgp router-id 22.22.22.22
    address-family ipv4 unicast
        table-policy qppb1
    address-family ipv6 unicast
        table-policy qppb2
    neighbor 30.2.2.1
        remote-as 500
        address-family ipv4 unicast
            route-policy pass in
            route-policy pass out
        address-family ipv6 unicast
            route-policy pass in
            route-policy pass out
```

Ingress QoS and IPv4 or IPv6 BGP Configuration

QPPB is enabled per interface and individually for IPv4 and IPv6. An ingress policy matches on the QoS groups marked by QPPB and takes the required action.

If a packet is destined for a destination prefix on which BGP route policy has stored a **qos-group**, but it ingresses on an interface on which QPPB is not enabled, it would not be remarked with **qos-group**.

Example:

```
class-map match-any qos-group5
    match qos-group 5
    end-class-map

class-map match-any qos-group4
    match qos-group 4
    end-class-map

policy-map ingress-marker-po1
    class qos-group5
        set precedence 0
        set discard-class 0
        set traffic-class 1

    class qos-group4
        set precedence 1
        set discard-class 1
        set traffic-class 2
    class class-default

end-policy-map

Interface hun 0/0/0/0
    [vrf vrfA]
    ipv4 address 25.1.1.1/24
    ipv6 address 2001:db8:a0b:12f0::1/64
    ipv4 bgp policy propagation input qos-group destination
    ipv6 bgp policy propagation input qos-group destination
    service-policy input ingress-marker-pol
```

Egress Interface Configuration

The traffic-class set on ingress has no existence outside the device. Also, traffic-class is not a part of any packet header but is associated internal context data on relevant packets. It can be used as a match criteria in an egress policy to set up various fields on the outgoing packet or shape flows.

```
class-map match-any level1
     match traffic-class 1
end-class-map

class-map match-any level2
    match traffic-class 2
end-class-map

policy-map output-po1
    class level1
        bandwidth percent 50
    class level2
        bandwidth percent 20
        queue-limit 50 ms
end-policy-map
```

```
interface hun 0/5/0/0
     ipv4 address 30.1.1.1/24
     ipv6 address 2001:da8:b0a:12f0::1/64
     service-policy output output-po1
```