Configuring MPLS QoS

Prerequisites for MPLS QoS

- The switch must be configured as a Multiprotocol Label Switching (MPLS) provider edge (PE) or provider (P) router, which includes the configuration of a valid label protocol and underlying IP routing protocols.

Restrictions for MPLS QoS

- MPLS classification and marking can only occur in an operational MPLS network.
- If a packet is classified by IP type of service (ToS) or class of service (CoS) at ingress, it cannot be reclassified by MPLS EXP at egress (imposition case). However, if a packet is classified by MPLS at ingress, it can be reclassified by IP ToS, CoS, or Quality of Service (QoS) group at egress (disposition case).
- To apply QoS on traffic across protocol boundaries, use QoS-group. You can classify and assign ingress traffic to the QoS-group. Thereafter, you can use the QoS-group at egress to classify and apply QoS.
- If a packet is encapsulated in MPLS, the MPLS payload cannot be checked for other protocols, such as IP, for classification or marking. Only MPLS EXP marking affects packets that are encapsulated by MPLS.
- The short pipe mode is not supported to transport packets through the MPLS network. You can transport packets using any one of the following modes—uniform mode or pipe mode.

Information About MPLS QoS

This section provides detailed information about MPLS QoS.

MPLS QoS Overview

The MPLS QoS functionality enables network administrators to provide differentiated services across an MPLS network. Network administrators can satisfy a wide range of networking requirements by specifying the CoS applicable to each transmitted IP packet. Different classes of service can be established for IP packets by setting the IP precedence bit in the header of each packet. Classification, remarking, and queuing on an MPLS network is performed over MPLS EXP bits. In the MPLS network, the packets are differentiated by the MPLS EXP field marking, and are treated accordingly, depending on the weighted early random detection (WRED) configuration.

The MPLS EXP field in MPLS packet allows you to:

- Classify traffic

  The classification process selects the traffic to be marked. Classification accomplishes this by partitioning traffic into multiple priority levels, or classes of service. Traffic classification is the primary component of class-based QoS provisioning. For more information, see “Classifying Network Traffic”.

- Police and mark traffic

  Policing causes traffic that exceeds the configured rate to be discarded or marked to a different drop level. Marking traffic is a way to identify packet flows to differentiate them. Packet marking allows you to partition your network into multiple priority levels or classes of service. For more information, see “Marking Network Traffic”.

- Queueing

  Queueing helps prevent traffic congestion. This includes priority level queueing, weighted tail drop (WTD), scheduling, shaping and weighted random early detection (WRED) features.

MPLS Experimental Field

The MPLS experimental bits (EXP) field is a 3-bit field in the MPLS header that you can use to define the QoS treatment (per-hop behavior) that a node should give to a packet. In an IP network, the DiffServ Code Point (DSCP) (a 6-bit field) defines a class and drop precedence. The EXP bits can be used to carry some of the information encoded in the IP DSCP and can also be used to encode the dropping precedence.

By default, Cisco IOS Software copies the three most significant bits of the DSCP or the IP precedence of the IP packet to the EXP field in the MPLS header. This action happens when the MPLS header is initially imposed on the IP packet. However, you can also set the EXP field by defining a mapping between the DSCP or IP precedence and the EXP bits. This mapping is configured using the **set mpls experimental** or **police** commands. For more information, see the “How to Classify and Mark MPLS EXP” section.

Note A policy map configured with **set ip dscp** is not supported on the provider edge device because the policy action for MPLS label imposition node should be based on **set mpls experimental imposition** value. However, a policy map with action **set ip dscp** is supported when both the ingress and egress interfaces are Layer 3 ports.

You can perform MPLS EXP marking operations using table-maps. It is recommended to assign QoS-group to a different class of traffic in ingress policy and translate QoS-group to DSCP and EXP markings in egress policy using table-map.

If a service provider does not want to modify the value of the IP precedence field in packets transported through the network, they can use the MPLS EXP field value to classify and mark IP packets.

By choosing different values for the MPLS EXP field, you can mark critical packets so that those packets have priority if network congestion occurs.

WRED monitors network traffic to anticipate and prevent congestion at common network and internetwork bottlenecks. WRED can selectively discard lower priority traffic when an interface becomes congested. This feature can also provide differentiated performance characteristics for different classes of service.

There are two ways to transport packets through the MPLS network:

Uniform mode: Uniform mode of transferring packets operates on one layer of QoS. The Provider Edge at ingress copies the DSCP information from the incoming IP packet into the MPLS EXP bits of the imposed labels and the IP precedence bits are mapped to the MPLS EXP field . As the EXP bits travel through the core, they may or may not be modified by the intermediate devices on the network. The Provider Edge at egress copies the EXP bits to the DSCP bits of the newly exposed IP packet.

Pipe mode: Pipe mode of transferring packets operates on two layers of QoS. An underlying QoS for the data that remains unchanged when traversing the core. A per-core QoS, which is separate from that of the underlying IP packets. The DSCP information is saved and stored as the packet travels through the MPLS network. The MPLS EXP label is applied by the PE at ingress but the IP precedence bits are not stored. At egress, the original IP precedence value is preserved.

Benefits of MPLS EXP Classification and Marking

The QoS EXP Matching feature allows you to classify, mark and queue network traffic by modifying the Multiprotocol Label Switching (MPLS) experimental bits (EXP) field. If a service provider does not want to modify the value of the IP precedence field in packets transported through the network, they can use the MPLS EXP field value to classify and mark IP packets.

By choosing different values for the MPLS EXP field, you can mark critical packets so that those packets have priority if network congestion occurs.

How to Configure MPLS QoS

This section provides information about how to configure MPLS QoS:

Classifying MPLS Encapsulated Packets

You can use the **match mpls experimental topmost** command to define traffic classes based on the packet EXP values, inside the MPLS domain. You can use these classes to define services policies to mark the EXP traffic using the **police** command.

Procedure

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

Step 3	**class-map** [**match-all** | **match-any**] *class-map-name*
	Example:
	`Device(config)# class-map exp3`
		Creates a class map to be used for matching traffic to a specified class, and enters class-map configuration mode.
		Enter the class map name.

Step 4	**match mpls experimental topmost** *mpls-exp-value* **Example:** `Device(config-cmap)# match mpls experimental topmost 3`	Specifies the match criteria. **Note** The **match mpls experimental topmost** command classifies traffic on the basis of the EXP value in the topmost label header.

Step 5	**end**
	Example:
	`Device(config-cmap)# end`
		(Optional) Returns to privileged EXEC mode.

Marking MPLS EXP on the Outermost Label

Perform this task to set the value of the MPLS EXP field on imposed label entries.

Before you begin

Marking MPLS packets at imposition is used with ingress classification on IP ToS or CoS fields.

Note

- For IP imposition marking, the IP precedence value is copied to the MPLS EXP value by default.
- The egress policy on provider edge works with MPLS EXP class match, only if there is a remarking policy at ingress. The provider edge at ingress is an IP interface and only DSCP value is trusted by default. If you do not configure remarking policy at ingress the label for queueing is generated based on DSCP value and not MPLS EXP value. However, a transit provider router works without configuring remarking policy at ingress as the router works on MPLS interfaces.
- The **set mpls experimental imposition** command works only on packets that have new or additional MPLS labels added to them.

Procedure

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

Step 4	**class** *class-map-name*
	Example:
	`Device(config-pmap)# class prec012`
		Creates a class map to be used for matching traffic to a specified class, and enters class-map configuration mode.
		Enter the class map name.

Step 5	**set mpls experimental imposition** *mpls-exp-value* **Example:** `Device(config-pmap-c)# set mpls experimental imposition 2`	Sets the value of the MPLS EXP field on top label.

Step 6	**end**
	Example:
	`Device(config-pmap-c)# end`
		(Optional) Returns to privileged EXEC mode.

Marking MPLS EXP on Label Switched Packets

Note The **set mpls experimental topmost** command marks EXP for the outermost label of MPLS traffic. Due to this marking at ingress policy, the egress policy must include classification based on the MPLS EXP values.

Perform this task to set the MPLS EXP field on label switched packets.

Procedure

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

Step 4	**class** *class-map-name*
	Example:
	`Device(config-pmap)# class-map exp012`
		Creates a class map to be used for matching traffic to a specified class, and enters class-map configuration mode.
		Enter the class map name.

Step 5	**set mpls experimental topmost** *mpls-exp-value* **Example:** `Device(config-pmap-c)# set mpls experimental topmost 2`	Sets the MPLS EXP field value in the topmost label on the output interface.

Step 6	**end**
	Example:
	`Device(config-pmap-c)# end`
		(Optional) Returns to privileged EXEC mode.

Configuring Conditional Marking

To conditionally set the value of the MPLS EXP field on all imposed label, perform the following task:

Before you begin

Note The **set-mpls-exp-topmost-transmit** action affects MPLS encapsulated packets only. The **set-mpls-exp-imposition-transmit** action affects any new labels that are added to the packet.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**policy-map** *policy-map-name* **Example:** `Device(config)# policy-map ip2tag`	Specifies the name of the policy map to be created and enters policy-map configuration mode. • Enter the policy map name.

Step 4	**class** *class-map-name*
	Example:
	`Device(config-pmap)# class iptcp`
		Creates a class map to be used for matching traffic to a specified class, and enters policy-map class configuration mode.
		• Enter the class map name.

Step 5	**police cir** *bps* **bc pir** *bps* **be** **Example:** `Device(config-pmap-c)# police cir 1000000 pir 2000000`	Defines a policer for classified traffic and enters policy-map class police configuration mode.
Step 6	**conform-action transmit** **Example:** `Device(config-pmap-c-police)# conform-action transmit 3`	Defines the action to take on packets that conform to the values specified by the policer. • In this example, if the packet conforms to the committed information rate (cir) or is within the conform burst (bc) size, the MPLS EXP field is set to 3.
Step 7	**exceed-action set-mpls-exp-topmost-transmit exp table** *table-map-name* **Example:**	Defines the action to take on packets that exceed the values specified by the policer.

	Command or Action	Purpose
	`Device(config-pmap-c-police)# exceed-action set-mpls-exp-topmost-transmit exp table dscp2exp`	
Step 8	**violate-action drop** **Example:** `Device(config-pmap-c-police)# violate-action drop`	Defines the action to take on packets whose rate exceeds the peak information rate (pir) and is outside the bc and be ranges. • You must specify the exceed action before you specify the violate action. • In this example, if the packet rate exceeds the pir rate and is outside the bc and be ranges, the packet is dropped.

Step 9	**end**
	Example:
	`Device(config-pmap-c-police)# end`
		(Optional) Returns to privileged EXEC mode.

Configuring WRED for MPLS EXP

Perform this task to enable WRED for MPLS EXP.

Procedure

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

Step 4	**class** *class-map-name*
	Example:
	`Device(config-pmap)# class exp`
		Creates a class map to be used for matching traffic to a specified class, and enters class-map configuration mode.
		• Enter the class map name.

	Command or Action	Purpose

Step 5	**bandwidth** {*kbps* | **remaining***percentage* | **percent***percentage* }
	Example:
	`Device(config-pmap-c)# bandwidth percent 30`
		Specify either the bandwidth allocated for a class belonging to a policy map or the traffic shaping.

Step 6	[illegible] **Example:** `Device(config-pmap-c)# random-detect mpls-exp- based`	Configures WRED to use the MPLS EXP value when it calculates the drop probability for the packet.

Step 7	**random-detectexp***exp-value***percent***min-threshold max-threshold*
	Example:
	`Device(config-pmap-c)# random-detect exp 1 10 20`
	`Device(config-pmap-c)# random-detect exp 2 30 40`
	`Device(config-pmap-c)# random-detect exp 2 40 80`
		Specifies the MPLS EXP value, minimum and maximum thresholds, in percentage.

Step 8	**end**
	Example:
	`Device(config-pmap-c-police)# end`
		(Optional) Returns to privileged EXEC mode.

Configuration Examples for MPLS QoS

This section provides configuration examples for MPLS QoS.

Example: Classifying MPLS Encapsulated Packets

Defining an MPLS EXP Class Map

The following example shows how to define a class map named exp3 that matches packets that contains MPLS experimental value 3:

```
Device(config)# class-map exp3
Device(config-cmap)# match mpls experimental topmost 3
Device(config-cmap)# exit
```

Defining a Policy Map and Applying the Policy Map to an Ingress Interface

The following example shows how to use the class map created in the example above to define a policy map. This example also shows how to applys the policy map to a physical interface for ingress traffic.

```
Device(config)# policy-map change-exp-3-to-2
Device(config-pmap)# class exp3
Device(config-pmap-c)# set mpls experimental topmost 2
Device(config-pmap)# exit
Device(config)# interface GigabitEthernet 0/0/0
Device(config-if)# service-policy input change-exp-3-to-2
Device(config-if)# exit
```

Defining a Policy Map and Applying the Policy Map to an Egress Interface

The following example uses the class map created in the example above to define a policy map. This example also applies the policy map to a physical interface for egress traffic.

```
Device(config)# policy-map WAN-out
Device(config-pmap)# class exp3
Device(config-pmap-c)# shape average 10000000
Device(config-pmap-c)# exit
Device(config-pmap)# exit
Device(config)# interface GigabitEthernet 0/0/0
Device(config-if)# service-policy output WAN-out
Device(config-if)# exit
```

Example: Marking MPLS EXP on Outermost Label

Defining an MPLS EXP Imposition Policy Map

The following example defines a policy map that sets the MPLS EXP imposition value to 2 based on the IP precedence value of the forwarded packet:

```
Device# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Device(config)# class-map prec012
Device(config-cmap)# match ip prec 0 1 2
Device(config-cmap)# exit
Device(config)# policy-map mark-up-exp-2
Device(config-pmap)# class prec012
Device(config-pmap-c)# set mpls experimental imposition 2
Device(config-pmap-c)# exit
Device(config-pmap)# exit
```

Applying the MPLS EXP Imposition Policy Map to a Main Interface

The following example applies a policy map to Gigabit Ethernet interface 0/0/0:

```
Device# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Device(config)# interface GigabitEthernet 0/0/0
Device(config-if)# service-policy input mark-up-exp-2
Device(config-if)# exit
```

Example: Marking MPLS EXP on Label-Switched Packets

Defining an MPLS EXP Label Switched Packets Policy Map

The following example shows how to define a policy map that sets the MPLS EXP top-most value to 2 according to the MPLS EXP value of the forwarded packet:

```
Device# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Device(config)# class-map exp012
Device(config-cmap)# match mpls experimental topmost 0 1 2
Device(config-cmap)# exit
Device(config-cmap)# policy-map mark-up-exp-2
Device(config-pmap)# class exp012
Device(config-pmap-c)# set mpls experimental topmost 2
Device(config-pmap-c)# exit
Device(config-pmap)# exit
```

Applying the MPLS EXP on Label-Switched Packets Policy Map to a Main Interface

The following example shows how to apply the policy map to a main interface:

```
Switch# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Device(config)# interface GigabitEthernet 0/0/0
Device(config-if)# service-policy input mark-up-exp-2
Device(config-if)# exit
```

Example: Configuring Conditional Marking

The following example shows how to create a policer for the **iptcp** class, which is part of the **ip2tag** policy map, and attaches the policy map to the Gigabit Ethernet interface:

```
Device(config)# policy-map ip2tag
Device(config-pmap)# class iptcp
Device(config-pmap-c)# police cir 1000000 pir 2000000
Device(config-pmap-c-police)# conform-action transmit
Device(config-pmap-c-police)# exceed-action set-mpls-exp-imposition-transmit 2
Device(config-pmap-c-police)# violate-action drop
Device(config-pmap-c-police)# exit
Device(config-pmap-c)# exit
Device(config-pmap)# exit
Device(config)# interface GigabitEthernet 0/0/1
Device(config-if)# service-policy input ip2tag
```

Example: Configuring WRED for MPLS EXP

The following example shows how to enable WRED for MPLS EXP:

```
Device# configure terminal
Device(config)# policy-map wred_exp
Device(config-pmap-c)# bandwidth percent 30
Device(config-pmap-c)# random-detect mpls-exp-based
Device(config-pmap-c)# random-detect exp 1 10 20
Device(config-pmap-c)# random-detect exp 2 30 40
Device(config-pmap-c)# random-detect exp 2 40 80
```

Displaying WRED Threshold Labels

The **show policy-map** *policy-map-name* command verifies the WRED configuration for MPLS EXP.

The following sample output displays WRED threshold labels:

```
Device# show policy-map wred_exp
Policy Map wred_exp
Class exp
bandwidth 30 (%)
percent-based wred, exponential weight 9
exp    min-threshold    max-threshold
--------------------------------------
 0          -                -
 1          10               20
 2          30               40
 3          40               80
 4          -                -
 5          -                -
 6          -                -
 7          -                -
```

Example: Configuring WRED for MPLS EXP

The following example shows how to enable WRED for MPLS EXP on the Cisco Catalyst 9600 Series Supervisor 2 Module.

```
Device(config)#table-map exp2tc
Device(config-tablemap)#map from 0 to 0
Device(config-tablemap)#map from 1 to 1
Device(config-tablemap)#map from 2 to 2
Device(config-tablemap)#map from 3 to 3
Device(config-tablemap)#map from 4 to 4
Device(config-tablemap)#map from 5 to 5
Device(config-tablemap)#map from 6 to 6
Device(config-tablemap)#map from 7 to 7
Device(config-tablemap)#end
Device(config)#table-map exp2discard
Device(config-tablemap)#map from 0 to 0
Device(config-tablemap)#map from 1 to 0
Device(config-tablemap)#map from 2 to 0
Device(config-tablemap)#map from 3 to 0
Device(config-tablemap)#map from 4 to 0
Device(config-tablemap)#map from 5 to 1
Device(config-tablemap)#map from 6 to 1
Device(config-tablemap)#map from 7 to 1
Device(config)#exit

Device(config)#policy-map p1
Device(config-pmap)#cla
Device(config-pmap)#class class-default
Device(config-pmap-c)# set traffic-class mpls experimental imposition table exp2tc
Device(config-pmap-c)# set discard-class mpls experimental imposition table exp2discard
Device(config-pmap-c)#end

Device(config)#class-map tc7
Device(config-cmap)#match traffic-class 7
Device(config-cmap)#exit
Device(config)#policy-map type queueing p2
Device(config-pmap)#class tc7
Device(config-pmap-c)#shape average 1000000000
Device(config-pmap-c)#  queue-limit 300000000 bytes
Device(config-pmap-c)#  random-detect discard-class-based
Device(config-pmap-c)#  random-detect discard-class 0 percent 40 90 1
Device(config-pmap-c)#  random-detect discard-class 1 percent 40 90 1
Device(config-pmap-c)#end
```

The following example displays the running policy configuration of WRED for MPLS EXP on the Cisco Catalyst 9600 Series Supervisor 2 Module.

```
Device#show running-config policy-map WRED
Building configuration...

Current configuration : 1723 bytes
!
policy-map type queueing WRED
 class tc7
  shape average 1000000000
  queue-limit 300000000 bytes
  random-detect discard-class-based
  random-detect discard-class 0 percent 40 90 1
  random-detect discard-class 1 percent 40 90 1
 class tc6
  shape average 1000000000
  random-detect discard-class-based
  random-detect discard-class 0 percent 50 90 1
  random-detect discard-class 1 percent 20 50 1
  queue-limit 300000000 bytes
 class tc5
  shape average 1000000000
  queue-limit 300000000 bytes
  random-detect discard-class-based
  random-detect discard-class 0 percent 40 90 1
  random-detect discard-class 1 percent 30 90 1
 class tc4
  shape average 1000000000
  queue-limit 300000000 bytes
  random-detect discard-class-based
  random-detect discard-class 0 percent 30 90 1
  random-detect discard-class 1 percent 20 80 1
 class tc3
  shape average 1000000000
  queue-limit 300000000 bytes
  random-detect discard-class-based
  random-detect discard-class 0 percent 60 90 1
  random-detect discard-class 1 percent 20 50 1
 class tc2
  shape average 1000000000
  queue-limit 300000000 bytes
  random-detect discard-class-based
  random-detect discard-class 0 percent 40 90 1
  random-detect discard-class 1 percent 30 90 1
 class tc1
  shape average 1000000000
  queue-limit 300000000 bytes
  random-detect discard-class-based
  random-detect discard-class 0 percent 40 90 1
  random-detect discard-class 1 percent 30 90 1
 class class-default
  shape average 1000000000
  queue-limit 300000000 bytes
  random-detect discard-class-based
  random-detect exponential-weighting-constant 3
  random-detect discard-class 0 percent 20 90 1
  random-detect discard-class 1 percent 20 70 1
!
end
```

Example: Configuring Priority Queue for MPLS EXP

The following example shows how to enable Priority Queue for MPLS EXP on the Cisco Catalyst 9600 Series Supervisor 2 Module.

```
Device(config)#table-map exp2tc
Device(config-tablemap)#map from 0 to 0
Device(config-tablemap)#map from 1 to 1
Device(config-tablemap)#map from 2 to 2
Device(config-tablemap)#map from 3 to 3
Device(config-tablemap)#map from 4 to 4
Device(config-tablemap)#map from 5 to 5
Device(config-tablemap)#map from 6 to 6
Device(config-tablemap)#map from 7 to 7
Device(config-tablemap)#end
Device(config)#policy-map p1
Device(config-pmap)#class class-default
Device(config-pmap-c)# set traffic-class mpls experimental imposition table exp2tc
Device(config)#policy-map type queueing p2
Device(config-pmap)#class tc7
Device(config-pmap-c)#priority level 1
```

The following example displays the Priority Queue for MPLS EXP on the Cisco Catalyst 9600 Series Supervisor 2 Module.

```
Device#sh policy-map type queueing p2
  Policy Map type queueing p2
    Class tc7
      priority level 1
    Class tc6
      priority level 2
    Class tc5
      Average Rate Traffic Shaping
      cir 20%

interface Port-channel30
no switchport
ip address 30.30.30.1 255.255.255.0
mpls ip
bfd interval 750 min_rx 750 multiplier 3
service-policy input p1
end

interface FiftyGigE6/0/47
no switchport
ip address 31.31.31.1 255.255.255.0
mpls ip
service-policy type queueing output p2
end
```

```
Device#sh policy-map int po30 in
Port-channel30

  Service-policy input: p1

    Class-map: class-default (match-any)
      1203656 packets
      Match: any
      QoS Set
         traffic-class mpls experimental imposition table exp2tc

Device#sh policy-map type queueing interface fif6/0/47 out
FiftyGigE6/0/47

  Service-policy queueing output: p2

    queue stats for all priority classes:
      Queueing
      priority level 1
      queue limit 96000 bytes
      (total drops) 0
      (bytes output) 145201

    queue stats for all priority classes:
      Queueing
      priority level 2
      queue limit 96000 bytes
      (total drops) 0
      (bytes output) 870

    Class-map: tc7 (match-all)
      1074 packets
      Match: traffic-class 7
      Priority: Strict,

      Priority Level: 1

    Class-map: tc6 (match-all)
      12 packets
      Match: traffic-class 6
      Priority: Strict,

      Priority Level: 2

    Class-map: tc5 (match-all)
      2183538 packets
      Match: traffic-class 5
      Queueing
      queue limit 7500000 bytes
      (total drops) 0
      (bytes output) 2192272152
      shape (average) cir 2000000000, bc 20000000, be 20000000
      target shape rate 2000000000

    Class-map: class-default (match-any)
      17584359 packets
      Match: any

      queue limit 7500000 bytes
      (total drops) 0
      (bytes output) 17654601516
```

Additional References

Related Documents

Related Topic	Document Title
QoS commands	*Cisco IOS Quality of Service Solutions Command Reference*

Feature History for QoS MPLS EXP

This table provides release and related information for features explained in this module.

These features are available on all releases subsequent to the one they were introduced in, unless noted otherwise.

Release	Feature	Feature Information
Cisco IOS XE Gibraltar 16.11.1	QoS MPLS EXP	The QoS EXP Matching feature allows you to classify, mark and queue network traffic by modifying the Multiprotocol Label Switching (MPLS) experimental bits (EXP) field.
Cisco IOS XE Amsterdam 17.3.1	MPLS QoS - WRED	Introduces support for weighted random early detection (WRED) in MPLS Quality of Service (QoS). This feature configures WRED to use the MPLS experimental bits (EXP) to calculate the drop probability of a packet.
Cisco IOS XE Cupertino 17.7.1	QoS MPLS EXP	Support for this feature was introduced on the Cisco Catalyst 9600 Series Supervisor 2 Module (C9600X-SUP-2).

Use Cisco Feature Navigator to find information about platform and software image support. To access Cisco Feature Navigator, go to http://www.cisco.com/go/cfn.