

MPLS QoS

This document provides detailed understanding about MPLS QoS and its configuration.

- Feature history for MPLS QoS, on page 1
- MPLS QoS, on page 1
- MPLS experimental field, on page 2
- Benefits of MPLS EXP classification and marking, on page 3
- Prerequisites for MPLS QoS, on page 3
- Restrictions for MPLS QoS, on page 3
- Configure MPLS QoS, on page 4
- Configuration examples, on page 8

Feature history for MPLS QoS

This table provides release and platform support information for the features explained in this module.

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

Release	Feature name and description	Supported platform
Cisco IOS XE 26.2.1ea	MPLS QoS: MPLS QoS feature support has been introduced.	Cisco C9550 Series Smart Switches
Cisco IOS XE 17.18.2	MPLS QoS: MPLS QoS feature allows you to classify, mark, and queue network traffic by modifying the MPLS experimental bits field.	Cisco C9350 Series Smart Switches Cisco C9610 Series Smart Switches

MPLS QoS

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 class of service (CoS) applicable to each transmitted IP packet. Different classes of service are established for IP packets by setting the IP precedence bit in the packet header.

Classifications, remarkings, and queuings on MPLS networks are 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 an 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.
- 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.
- Queueing: Queueing prevents congestion with priority level queueing, WTD, scheduling, shaping, and WRED features.

The system supports HQoS on MPLS QoS, as same as standard QoS.

MPLS QoS and standard QoS share the same set of configuration restrictions.

MPLS experimental field

The MPLS experimental bits (EXP) field is a 3-bit field in the MPLS header used to define the QoS treatment (per-hop behavior) given to a packet by a node. 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.

Cisco IOS Software defaults to copying 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.



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.

Service providers may choose not to modify the value of the IP precedence field in packets transported through their network. Instead, they can use the MPLS EXP field value to classify and mark IP packets. By selecting different values for the MPLS EXP field, critical packets can be marked to ensure priority during network congestion.

WRED monitors network traffic to anticipate congestion and prevent it 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.

Packet transport through an MPLS network

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, mapping the IP precedence bits 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 enables the classification, marking, and queuing of network traffic by modifying the Multiprotocol Label Switching (MPLS) experimental bits (EXP) field. If you do not want to modify the value of the IP precedence field in packets transported through the network, 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.

Prerequisites for MPLS QoS

- Configure the switch as an MPLS provider edge (PE) or provider (P) router, including 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 at egress by IP ToS, CoS, or Quality of Service (QoS) group (disposition case).
- Use QoS-group to apply QoS on traffic across protocol boundaries. You can classify and assign ingress traffic to the QoS-group. Then, use the QoS-group at egress to classify and apply QoS.
- When a packet is encapsulated in MPLS, it cannot be checked for classification or marking of other protocols such as IP. 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. Transport packets using either uniform mode or pipe mode.

- MPLS classification and marking are not supported on SVI. Apply the policy on either physical or port-channel interfaces.

Configure MPLS QoS

This section provides information about configuring MPLS QoS.

Classify MPLS encapsulated packets

Use the **match mpls experimental topmost** command to define traffic classes based on the packet EXP values, inside the MPLS domain. Use these classes to define service 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.

Mark MPLS EXP on the outermost label

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.

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

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:**	Sets the value of the MPLS EXP field on top label.

	Command or Action	Purpose
	`Device(config-pmap-c)# set mpls experimental imposition 2`	
Step 6	**end** **Example:** `Device(config-pmap-c)# end`	(Optional) Returns to privileged EXEC mode.

Mark MPLS EXP on label switched packets

Note Use the **set mpls experimental topmost** command to mark the EXP value for the outermost label of MPLS traffic. Since this marking occurs at ingress policy, ensure the egress policy includes classification based on 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:**	Sets the MPLS EXP field value in the topmost label on the output interface.

	Command or Action	Purpose
	`Device(config-pmap-c)# set mpls experimental topmost 2`	
Step 6	**end** **Example:** `Device(config-pmap-c)# end`	(Optional) Returns to privileged EXEC mode.

Configure conditional marking

Before you begin

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

Perform this task to conditionally set the MPLS EXP field value for all imposed labels.

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.

	Command or Action	Purpose
Step 5	**police cir** *bps* **bc pir** *bps* **be** **Example:** `Device(config-pmap-c)# police cir 100000000 pir 200000000`	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`	Defines the action to take on packets that conform to the values specified by the policer.
Step 7	**exceed-action set-mpls-exp-topmost-transmit exp table** *table-map-name* **Example:** `Device(config-pmap-c-police)# exceed-action set-mpls-exp-topmost-transmit exp table dscp2exp`	Defines the action to take on packets that exceed the values specified by the policer.
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. • Ensure the exceed action precedes 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.

Configuration examples

This section provides configuration examples for MPLS QoS.

Note MPLS traffic can be queued by classifying packets and assigning them to different queues based on their class of service (CoS) or MPLS experimental (EXP) bits.

Example: Classify MPLS encapsulated packets

Define a color of traffic for QoS enforcement

This example shows how to define a class map for the MPLS EXP Marking based on color of traffic for QoS enforcement:

```
Device(config)#class
Device(config)#class-map match-any c1
Device(config-cmap)#match mpls experimental topmost 1
Device(config-cmap)#exit
Device(config)#policy-map p1
Device(config-pmap)#class c1
Device(config-pmap-c)#police cir 2570000 ?
Device(config-pmap-c)#police cir 2570000 conform-action transmit ?
Device(config-pmap-c-police)#
Device(config-pmap-c-police)#$nform-action transmit exceed-action drop
Device(config-pmap-c-police)#exit
Device(config-pmap-c)#
```

Define an MPLS EXP class map

This 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
```

Define a policy map and apply the policy map to an ingress interface

This example shows how to use the class map created in the example above to define a policy map. It also shows how to apply 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 FiftyGigE1/0/2
Device(config-if)# service-policy input change-exp-3-to-2
Device(config-if)# exit
```

Example: Mark MPLS EXP on outermost label

Define an MPLS EXP imposition policy map

This 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
```

Apply the MPLS EXP imposition policy map to a main interface

This example applies a policy map to Fifty Gigabit Ethernet interface 1/0/2:

```
Device# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Device(config)# interface FiftyGigE1/0/2
Device(config-if)# service-policy input mark-up-exp-2
Device(config-if)# exit
```

Example: Mark MPLS EXP on label-switched packets

Define an MPLS EXP label switched packets policy map

This 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
```

Apply the MPLS EXP on label-switched packets policy map to a main interface

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

```
Device# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Device(config)# interface FiftyGigE1/0/2
Device(config-if)# service-policy input mark-up-exp-2
Device(config-if)# exit
```

Example: Configure conditional marking

This 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 Fifty Gigabit Ethernet interface interface:

```
Device(config)# policy-map ip2tag
Device(config-pmap)# class iptcp
Device(config-pmap-c)# police cir 100000000 pir 200000000
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 FiftyGigE1/0/2
Device(config-if)# service-policy input ip2tag
```

This is a sample output of the **sh policy-map interface** command.

```
Device(config-pmap)#interface HundredGigE1/0/1
Device(config-if)#service-policy input ip2tag
Device(config-if)#do sh policy-map interface HundredGigE1/0/1
 HundredGigE1/0/1

  Service-policy input: ip2tag

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