

Modular QoS Deployment Scenarios

This module provides deployment scenarios use cases for specific QoS features or for QoS implementations of features that are described in other technology guides such as L2VPN or MPLS.

Line Card, SIP, and SPA Support

Feature	ASR 9000 Ethernet Line Cards	SIP 700 for the ASR 9000
802.1ad DEI	yes	no
Frame Relay QoS	no	yes
IPHC QoS	no	2-Port Channelized OC-12c/DS0 SPA only
L2VPN QoS	yes	yes
MLPPP/MLFR QoS	no	2-Port Channelized OC-12c/DS0 SPA only
MPLS QoS	yes	yes
QoS on Multicast VPN	yes	yes
QoS on NxDS0 Interfaces	no	2-Port Channelized OC-12c/DS0 SPA only

Feature History for QoS Deployment Scenarios on Cisco ASR 9000 Series Routers

Release	Modification
Release 3.7.2	The L2VPN QoS feature was introduced on ASR 9000 Ethernet Line Cards. The MPLS QoS feature was introduced on ASR 9000 Ethernet Line Cards.
Release 3.9.0	The MLPPP QoS feature was introduced on the SIP 700 for the ASR 9000.
Release 3.9.1	The QoS on Multicast VPN feature was introduced on ASR 9000 Ethernet Line Cards.

Release	Modification
Release 4.0.0	The 802.1ad DEI feature was introduced on the SIP 700 for the ASR 9000. The Frame Relay QoS feature was introduced on the SIP 700 for the ASR 9000. The IP Header Compression QoS feature was introduced on the SIP 700 for the ASR 9000. The L2VPN QoS feature was supported on the SIP 700 for the ASR 9000. The MLFR QoS feature was introduced on the SIP 700 for the ASR 9000. The suspend/resume approach was added for MLPPP and MLFR interfaces. The MPLS QoS feature was supported on the SIP 700 for the ASR 9000. The QoS on NxDS0 Interfaces feature was introduced on the SIP 700 for the ASR 9000.
Release 4.1.0	The VPLS and VPWS QoS feature was introduced.

- 802.1ad DEI, on page 2
- Frame Relay QoS, on page 3
- IP Header Compression QoS, on page 7
- L2VPN QoS, on page 8
- MLPPP QoS/MLFR QoS, on page 11
- MPLS QoS, on page 13
- QoS on Multicast VPN, on page 18
- QoS on NxDS0 Interfaces, on page 19
- VPLS and VPWS QoS, on page 20
- Related Information, on page 23

802.1ad DEI

You can classify traffic based on the Drop Eligible Indicator (DEI) bit that is present in 802.1ad frames and in 802.1ah frames. DEI support includes the ability to:

- Police to a certain rate and, based on whether the traffic is conforming or exceeding, mark the DEI as 0 or 1.
- On ingress, police and set up the discard class (even on an interface that is not configured for 802.1ad encapsulation).
- On egress, mark the DEI based on the discard class value (802.1ad interfaces only).

You can manage congestion based on the Drop Eligible Indicator (DEI) bit that is present in 802.1ad frames and 802.1ah frames. DEI support includes the ability to:

- Do weighted random early detection (WRED) based on the value of the DEI bit.
- Do active queue management during traffic congestion on an interface by giving preferential treatment to traffic (bigger thresholds) or set up smaller thresholds for out-of-profile traffic based on a DEI value.

Mark DEI Based on a Policing Action: Example

In this example, the police rate is set to 5 Mbps. Conforming traffic is marked with a DEI value of 0; traffic that exceeds the police rate is marked with a DEI value of 1.

```
policy-map 1ad-mark-dei
class c1
police rate 5 mbps
      conform-action set dei 0
      exceed-action set dei 1
end-policy-map
```

Mark DEI Based on Incoming Fields: Example

In this example, 802.1ad CoS plus DEI is derived from the incoming 802.1q CoS. Packets with a CoS value of 0 are remarked with a DEI value of 1.

```
class-map match-any remark-cos
  match cos 0
end-class-map

policy-map p1
  class remark-cos
  set dei 1
end-policy-map

interface GigabitEthernet0/4/0/39.1 l2transport
 encapsulation dot1q 1
 rewrite ingress tag push dot1ad 5 symmetric
 service-policy input p1
!
```

Congestion Management Using DEI: Example

In this example, congestion is managed by dropping packets with a DEI value of 1 before dropping packets with a DEI value of 0.

```
policy-map dei-sample
  class class-default
    random-detect dei 1 1000 6000
    random-detect dei 0 5000 10000
end-policy-map
```

Frame Relay QoS

The main difference between Frame Relay QoS and other interface types is that you can perform:

- Frame Relay DLCI classification
- Frame Relay DE classification
- Frame Relay DE marking

Note A QoS policy can be applied only to a PVC under a Frame Relay subinterface; it cannot be applied directly to a Frame Relay subinterface.

Frame Relay DLCI Classification

This configuration allows users to match on the Frame Relay DLCI value of packets encapsulated in Frame Relay. Packets that are not Frame Relay encapsulated do not match this configuration.

```
class-map foo
            match frame-relay list of dlci-values
```

The list of DLCI values can contain ranges as well as individual values, as in this example:

```
class-map foo
            match frame-relay dlci 1-100 150 200-300
```

Note DLCI matching is supported only on main interfaces.

Frame Relay DE Classification

This configuration allows the user to match Frame Relay packets that have the discard eligible (DE) bit set in the Frame Relay header:

```
class-map fr_class
  match fr-de 1
```

To match Frame Relay DE bit 0, use this configuration:

```
class-map match-not-fr-de
  match not fr-de 1
```

Note DE bit classification is not supported on Layer 3 interfaces.

Frame Relay DE Marking

In this example, the fr-de bit is set when traffic exceeds the policing committed information rate, so the downward system (when experiencing congestion) discards traffic with the fr-de bit set to 1.

```
policy-map fr_de_marking
 class class-default
  police rate percent 50
   conform-action transmit
```

```
   exceed-action set fr-de 1
  !
 !
 end-policy-map
```

Note DE bit marking is not supported on Layer 3 interfaces.

Frame Relay QoS: Example

In this example, parent_policy is applied to the Multilink Frame Relay main interface. There are two classes in parent_policy, which match on Frame Relay DLCIs. The Multilink Frame Relay main interface has two Frame Relay PVCs configured (DLCI 16, DLCI 17).

```
show run int multi 0/2/1/0/1
Mon Aug  2 11:34:31.019 UTC
interface Multilink0/2/1/0/1
 service-policy output parent_policy
 encapsulation frame-relay
 frame-relay intf-type dce
!

show run policy-map parent_policy
Mon Aug  2 11:34:36.118 UTC
policy-map parent_policy
 class parentQ_1
  service-policy child_queuing_policy
  shape average 64 kbps
 !
 class parentQ_2
  service-policy child_queuing_policy
  shape average 1 mbps
 !
 class class-default
 !
 end-policy-map
!

show run class-map  parentQ_1 <----- class map parent class dlci=16
Mon Aug  2 11:34:43.363 UTC
class-map match-any parentQ_1
 match frame-relay dlci 16
 end-class-map
!

show run class-map  parentQ_2 <----- class map parent class dlci=17
Mon Aug  2 11:34:45.647 UTC
class-map match-any parentQ_2
 match frame-relay dlci 17
 end-class-map
!

show run int multi 0/2/1/0/1.16 <------ dlci 16 pvc config
Mon Aug  2 11:34:53.988 UTC
interface Multilink0/2/1/0/1.16 point-to-point
 ipv4 address 192.1.1.1 255.255.255.0
 pvc 16
  encap cisco
 !
```

```
!
show run int multi 0/2/1/0/1.17  <------ dlci 17 pvc config
Mon Aug  2 11:34:56.862 UTC
interface Multilink0/2/1/0/1.17 point-to-point
 ipv4 address 192.1.2.1 255.255.255.0
 pvc 17
  encap cisco
 !
!
show run policy-map child_queuing_policy <--------- child policy-map
Mon Aug  2 11:35:05.821 UTC
policy-map child_queuing_policy
 class voice-ip
  priority level 1
  police rate percent 20
  !
 !
 class video
  bandwidth percent 40
 !
 class premium
  service-policy gchild_policy
  bandwidth percent 10
  random-detect discard-class 2 10 ms 100 ms
  random-detect discard-class 3 20 ms 200 ms
  queue-limit 200 ms
 !
 class best-effort
  bandwidth percent 20
  queue-limit 200 ms
 !
 class class-default
 !
 end-policy-map
!

show run policy-map gchild_policy  <-------- grandchild policy map
Mon Aug  2 11:35:15.428 UTC
policy-map gchild_policy
 class premium_g1
  police rate percent 10
  !
  set discard-class 2
 !
 class premium_g2
  police rate percent 50
  !
  set discard-class 3
 !
 class class-default
 !
 end-policy-map
!

show run class-map  <----------- shows all class map configs
Mon Aug  2 11:35:19.479 UTC
class-map match-any video
 match precedence 1
 end-class-map
!
class-map match-any premium
 match precedence 2 3
 end-class-map
!
```

```
class-map match-any voice-ip
 match precedence 0
 end-class-map
!
class-map match-any parentQ_1
 match frame-relay dlci 16
 end-class-map
!
class-map match-any parentQ_2
 match frame-relay dlci 17
 end-class-map
!
class-map match-any premium_g1
 match precedence 2
 end-class-map
!
class-map match-any premium_g2
 match precedence 3
 end-class-map
!
class-map match-any best-effort
 match precedence 4
 end-class-map
!
```

IP Header Compression QoS

An IP Header Compression (IPHC) profile can be enabled on an interface so that the IPHC profile applies only to packets that match a QoS service policy. In this case, the QoS service-policy class attributes determine which packets are compressed. This allows users to fine tune IPHC with greater granularity.

Policy maps are attached to an interface using the **service-policy** command. IPHC action applies only to output service policies. IPHC is not supported on input service policies. (IPHC is supported in the input direction but there is no use case to configure IPHC in an input policy.)

You can configure IPHC using QoS as follows:

- Create a QoS **policy** with the **compress header ip** action.
- Attach the IPHC profile to the interface using the **ipv4 iphc profile** *profile_name* **mode service-policy** command.
- Attach the QoS **policy** with **compress header ip** action using the **service-policy output** command.

You can also display IPHC statistics using the **show policy-map interface** command, as shown in the following example:

```
show policy-map interface Serial0/0/3/0/3:0 output

show policy-map int  Serial0/0/3/0/3:0  output
Mon May 18 22:06:14.698 UTC
Serial0/0/3/0/3:0 output: p1
Class class-default
  Classification statistics          (packets/bytes)     (rate - kbps)
    Matched             :                  0/0                   0
    Transmitted         :                  0/0                   0
    Total Dropped       :                  0/0                   0
  Queueing statistics
    Queue ID                             : 0
```

```
    High watermark  (Unknown)            : 0
    Inst-queue-len  (packets)            : 0
    Avg-queue-len   (packets)            : 0
    Taildropped(packets/bytes)           : 0/0
  Compression Statistics
    Header ip rtp
    Sent Total       (packets)           : 880
    Sent Compressed  (packets)           : 877
    Sent full header (packets)           : 342
    Saved            (bytes)             : 31570
    Sent             (bytes)             : 24750
    Efficiency improvement factor        : 2.27
```

IP Header Compression QoS: Example

In this example, IPHC is configured through QoS as an action under the class map using the **compress header ip** command.

The packets are classified according to the criteria in the class maps. The policy map specifies which behavior to apply to which classes. IPHC is enabled using the **compress header ip** action for the class. An IPHC profile with a QoS service policy is attached to a serial interface.

```
class-map match-all voice1
  match precedence 2
class-map match-all voice2
  match access-group acl_iphc

access-list acl_iphc permit udp any range lower-bound src udp port 5000 upper-bound src udp
 port15000 any lower-bound udp dst port 5000 upper-bound dst udp port 15000

ipv4 access-list acl_iphc permit udp any range 5000 15000 any range 5000 15000

policy-map iphc_policy
 class iphc_class_1
  compress header ip
 class iphc_class_2
  compress header ip

interface serial 0/1/0/1:1
 ipv4 iphc profile Profile_3 mode service-policy
  service-policy output iphc_policy

interface Serial 0/2/0/0/1/1/1:1
  ipv4 address 10.0.0.1 255.255.255.252
  ipv4 iphc profile Profile_3 mode service-policy
  service-policy output iphc_policy
  encapsulation ppp
```

L2VPN QoS

This section describes the following Frame Relay L2VPN deployment scenarios:

- Frame Relay <-> Frame Relay over pseudowire
- Frame Relay <-> Ethernet over pseudowire

Note There are local-connect variants of these scenarios that do not go over a pseudowire. This discussion focuses on the pseudowire scenarios.

Frame Relay - Frame Relay Over Pseudowire: Example

This example shows that you can match based on the Frame Relay DLCI on the ingress Frame Relay interface on router PE1 and set the fr-de value. This configuration is carried over the L2VPN pseudowire. When the Frame Relay packet exits router PE2 through the Frame Relay l2transport interface, the fr-de value is intact.

This configuration allows you to manipulate and carry over the Frame Relay QoS values across L2VPN. This figure shows the network topology.

Figure 1: Frame Relay Over MPLS



CE1

```
interface pos0/2/0/0.26
 pvc 26
 ipv4 add 10.0.0.1 255.0.0.0
```

PE1

```
interface pos0/2/0/0.26 l2transport
 pvc 26

l2vpn
 xconnect group frfr
 p2p p1
interface pos0/2/0/0.26
 neighbor y.y.y.y pw-id 1001

!QoS Policy
class-map matchdlci
 match frame-relay dlci 26

policy-map setde1
 class matchdlci
  set fr-de 1

interface pos0/2/0/0
 service-policy input setde1
```

PE2

```
interface pos0/3/0/0.26 l2transport
 pvc 26
```

```
l2vpn
 xconnect group frfr
 p2p p1
interface pos0/3/0/0.26
 neighbor x.x.x.x pw-id 1001
```

CE2

```
interface pos0/3/0/0.26
 pvc 26
 ipv4 add 10.0.0.2 255.0.0.0
```

Frame Relay - Ethernet Over Pseudowire: Example

This example shows that you can match based on the fr-de value on the ingress Frame Relay l2transport interface on router PE1 and set a specific MPLS EXP value. When the MPLS packet exits the PE1 core interface, this EXP value is set. When the packet exits router PE2 through the Ethernet l2transport interface, this value is part of the Ethernet packet CoS field.

This configuration allows you to carry over or map the QoS field from the Frame Relay network to the Ethernet network. This figure shows the network topology.

Figure 2: IP Interworking Over MPLS



CE1

```
interface pos0/2/0/0.26
 pvc 26
 ipv4 add 10.0.0.1 255.0.0.0
```

PE1

```
interface pos0/2/0/0.26 l2transport
 pvc 26

l2vpn
 xconnect group freth
 p2p p1
interface pos0/2/0/0.26
 neighbor y.y.y.y pw-id 1001
 interworking ipv4

!QoS Policy
class-map matchfrde
 match fr-de 1

policy-map setexp
 class matchfrde
  set mpls exp imposition 5
```

```
interface pos0/2/0/0.26 l2transport
 pvc 26
 service-policy input setexp
```

PE2

```
interface gig0/4/0/0.26 l2transport
 encapsulation dot1q 100

l2vpn
 xconnect group freth
 p2p p1
interface gig0/4/0/0.26
 neighbor x.x.x.x pw-id 1001
 interworking ipv4
```

CE2

```
interface gig0/4/0/0.26
 encapsulation dot1q 100
 ipv4 add 10.0.0.2 255.0.0.0
```

MLPPP QoS/MLFR QoS

Multilink provides a mechanism for aggregating multiple serial links into a bundle. Bundles support more bandwidth, load balancing between links, and improved service availability by protecting against single points of failure. The service allows users to increase bandwidth by aggregating multiple low speed links, which can be more cost-effective than upgrading to a single higher speed link. This provides a cost-effective solution for users requiring leased line service with bandwidth greater than T1 rates but below T3 rates.

Multilink interfaces can be configured with PPP encapsulation (MLPPP) or with Frame Relay encapsulation (MLFR). When a multilink interface is configured with Frame Relay encapsulation, subinterfaces can be configured below it.

The total bandwidth available for the multilink interface can change dynamically when links are added or removed to or from a multilink interface. The total bandwidth available can also change if the member links change state operationally to up or down, or by modifying the suspended condition of the policy. QoS policies applied on such interfaces need to be updated based on the bandwidth changes. In this case, one of the following actions is taken:

- Suspend the policy—Policy is suspended if the bandwidth requirements of the attached policy are more than the available bandwidth (which is reduced due to a member link going operationally down). Once the policy is suspended, any incoming or outgoing packets on that interface are not subject to QoS.

  A policy is suspended on ingress under these conditions:

  - In Enhanced Hierarchical Ingress Policing, when the sum of child police rates is greater than the parent police conform rate
  - Police peak rate is less than the police conform rate

    A policy is suspended on egress under these conditions:

  - Minimum bandwidth rate + priority class police rate is greater than the interface rate
  - Shape rate is less than the minimum bandwidth rate

- Priority class police conform rate is greater than the interface rate
- Priority class police peak rate is greater than the interface rate
- Police peak rate is less than the police conform rate

- Resume the policy—Policy is resumed if the bandwidth requirements of the attached policy are less than or equal to the available bandwidth, which increased due to a member link going operationally up. A suspended policy can also be resumed by modifying the suspended condition of the policy map without any change in the member link status.
- Update the policy—Active policy rates are updated to reflect the new available bandwidth. The available bandwidth could have increased or decreased, but the applied policy's bandwidth requirements can still be satisfied.

QoS statistics are not retained for the policy that transitions from an active state to a suspended state. If the policy is reactivated, all the previously collected statistics are lost and only the packets that pass through the interface after the reactivation are counted. The suspended policy can be modified to reduce its bandwidth requirements, so that it can be reactivated. A suspended policy can be modified while still attached to the interface.

Multiclass MLPPP with QoS

Multiclass Multilink Point-to-Point Protocol (MLPPP) can be used with QoS and configured using the **encap-sequence** command under a class in a policy map. The **encap-sequence** command specifies the MLPPP MCMP class ID for the packets in an MQC defined class.

The valid values for the **encap-sequence** ID number are **none**, 1, 2, or 3. The **none** value is applicable only when the **priority level** is 1 and indicates that there is no MLPPP encapsulation. The values 1, 2, or 3 can be used with priority 1 or 2 classes or other classes with queuing actions. An **encap-sequence** ID number of zero (0) is used by the system and is reserved for the default class; it cannot be specified in any other classes.



Note The **encap-sequence** ID numbers must be configured in numeric order. For example, you cannot assign an ID number of 3 unless you have already assigned 1 and 2.

The number of **encap-sequence** ID numbers must be less than the number of MLPPP classes that are negotiated between the peers via the multilink header. The user must ensure that the configuration is consistent as the system does not verify this.

The **ppp multilink multiclass remote apply** command provides a way to ensure this. You can ensure that the number of classes using an **encap-sequence** ID number (including the default of 0) is less than the min-numbe*r* value in the **ppp multilink multiclass remote apply** command. For example, if the min-number value is 4, you can only have three or fewer classes with encap-sequence ID numbers.

The QoS policy validates the following conditions. If these conditions are not met, the policy is rejected:

- The **encap-sequence** ID number is within the allowed values of 1 to 3.
- When **encap-sequence** is configured for any class in a policy map, all classes in that policy map with **priority level 1** must also contain an **encap-sequence** ID number.
- The **encap-sequence none** configuration is restricted to classes with **priority level** 1.

- The class-default does not contain an **encap-sequence** configuration.
- Only classes containing a queuing action have the **encap-sequence** configuration.

Note Classes that share the same **encap-sequence** ID number must have the same priority.

A QoS policy map is configured as follows:

```
config
 policy-map type qos policy-name  class class-name   action   action   action
. . .
```

The following example shows how to configure a policy map for MLPPP:

```
config
 policy-map foo
  class ip-prec-1
   encap-sequence none
   police rate percent 10
   priority level 1
 !
  class ip-prec-2
   encap-sequence 1
   shape average percent 80
 !
  class ip-prec-3
   encap-sequence 1
   bandwidth percent 10
 !
  class class-default
 !
 end-policy-map
 !
```

MLPPP QoS/MLFR QoS: Example

Because a bundle interface dynamically changes its bandwidth as the member links go up or down, QoS policies applied on such interfaces need to be updated based on the bandwidth changes.

MPLS QoS

Note The introductory text and topology diagrams are taken from “MPLS Fundamentals,” Luc De Ghein, Copyright 2007, Cisco Systems, Inc.

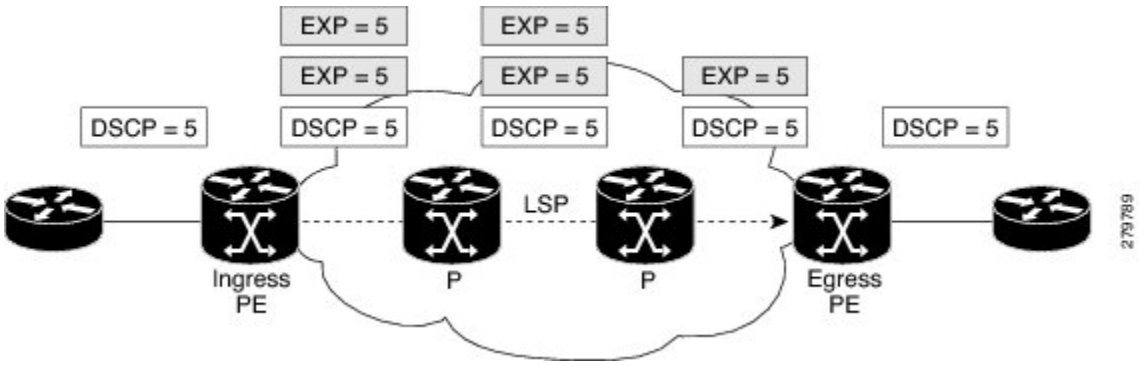
For MPLS QoS, there are three deployment scenarios based on tunneling model: uniform mode, pipe mode, and short pipe mode. Table 2 shows an overview of the tunneling models.

Tunneling Mode	IP-to-Label	Label-to-Label	Label-to-IP
Uniform	Copy IP precedence /DiffServ to MPLS EXP	MPLS EXP copied	Copy MPLS EXP to IP precedence/DiffServ
Pipe	MPLS EXP set according to service provider policy	MPLS EXP copied	Preserve IP precedence /DiffServ Forwarding treatment based on MPLS EXP
Short Pipe	MPLS EXP set according to service provider policy	MPLS EXP copied	Preserve IP precedence /DiffServ Forwarding treatment based on IP precedence/DiffServ

MPLS Uniform Mode

In uniform mode (as shown in following figure), there is only one DiffServ marking that is relevant for a packet when traversing the MPLS network. If the DiffServ marking of the packet is modified within the MPLS network, the updates information is the one considered meaningful at the egress of the LSP. Any changes to the packet marking within the MPLS network are permanent and get propagated when the packet leaves the MPLS network.

Figure 3: Uniform Mode

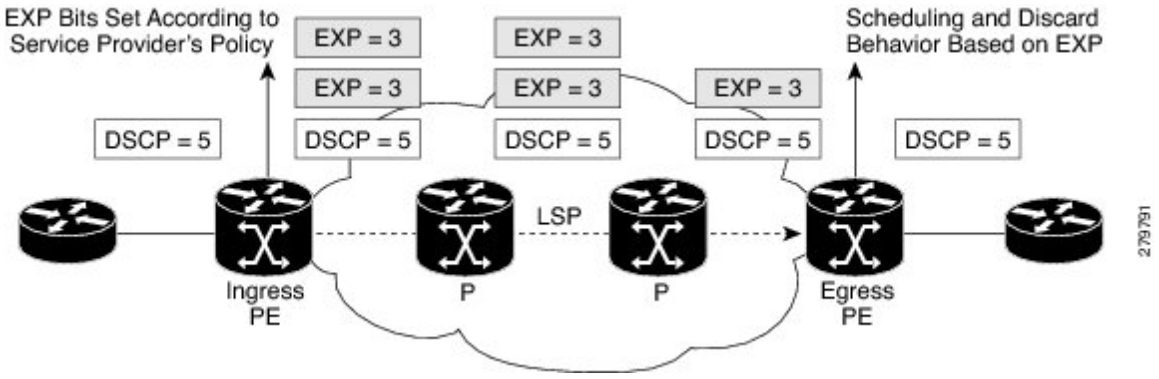


MPLS Pipe Mode

In pipe mode (as shown in the following figure), two markings are relevant for a packet when traversing the MPLS network. First, the marking used by intermediate nodes along the LSP span including the egress LSR. Second, the original marking carried by the packet before entering the MPLS network that will continue to be used once the packet leaves the MPLS network. Any changes to the packet marking within the MPLS network are not permanent and do not get propagated when the packet leaves the MPLS network.

Note that the egress LSR still uses the marking that was used by intermediate LSRs. However, the egress LSR has to remove all labels imposed on the original packet. In order to preserve this marking carried in the labels, the edge LSR keeps an internal copy of the marking before removing the labels. This internal copy is used to classify the packet on the outbound interface (facing the CE) once the labels are removed. This is usually achieved using the **set qos-group** and **match qos-group** commands.

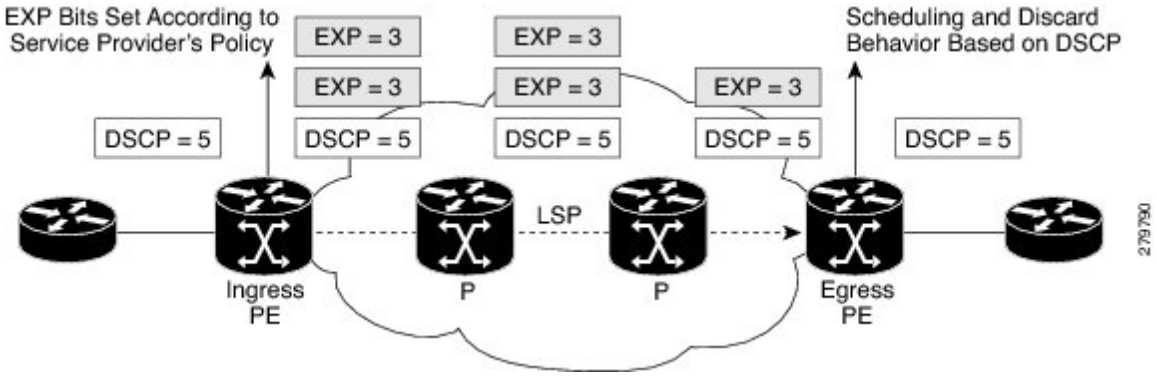
Figure 4: Pipe Mode



MPLS Short Pipe Mode

The short pipe mode, is a slight variation of the pipe mode. The only difference is that the egress LSR uses the original packet marking instead of using the marking used by the intermediate LSRs.

Figure 5: Short Pipe Mode



Uniform, Pipe, Short Pipe Modes: Ingress PE Example

This example shows how to implement the MPLS DiffServ and demonstrates the configuration needed on the ingress PE. Only precedence 4 is being matched. Precedence 4 is mapped to EXP bits value 4 by the policer, unless the bandwidth is exceeded, in which case the EXP bits are recolored to the value 2. The egress interface configuration is not needed for the MPLS DiffServ uniform model, but it is added to show how to perform QoS on the EXP bits.

```
!Ingress interface:
class-map prec4
match precedence 4
!
policy-map set-MPLS-PHB
class prec4
police rate 8000 kbps
conform-action set mpls experimental imposition 4
exceed-action set mpls experimental imposition 2
!
```

```
interface GigabitEthernet0/0/0/1
service-policy input set-MPLS-PHB

!Egress interface:
class-map exp2and4
match mpls experimental topmost 2 4
!
policy-map output-qos
class exp2and4
bandwidth percent 40
random-detect default
!
interface GigabitEthernet0/0/0/2
service-policy output output-qos
```

Uniform Mode: Egress PE Example

On the egress PE, the EXP bits are copied to the precedence bits using the **set qos-group** and **match qos-group** commands.

```
!Ingress interface:
class-map exp2
match mpls experimental topmost 2
!
class-map exp4
match mpls experimental topmost 4
!
policy-map policy2
class exp2
set qos-group 2
class exp4
set qos-group 4
!
interface GigabitEthernet0/0/0/2
service-policy input policy2

!Egress interface:
class-map qos2
match qos-group 2
class-map qos4
match qos-group 4
!
policy-map policy3
class qos2
set precedence 2
bandwidth percent 20
random-detect default
class qos4
set precedence 4
bandwidth percent 20
random-detect default
!
interface GigabitEthernet0/0/0/1
service-policy output policy3
```

Pipe Mode: Egress PE Example

This example shows the configuration of the egress PE for the MPLS DiffServ pipe mode. The egress LSR does not copy the EXP bits to the precedence bits of the outgoing IP packet. The scheduling of the packets

on the egress interface is done indirectly on the EXP bits using the **set qos-group** and **match qos-group** commands.

```
!Ingress interface:
class-map exp2
match mpls experimental topmost 2
!
class-map exp4
match mpls experimental topmost 4
!
policy-map policy2
class exp2
set qos-group 2
class exp4
set qos-group 4
!
interface GigabitEthernet0/0/0/2
service-policy input policy2

!Egress interface:
class-map qos2
match qos-group 2
class-map qos4
match qos-group 4
!
policy-map policy3
class qos2
bandwidth percent 20
random-detect default
class qos4
bandwidth percent 20
random-detect default
!
interface GigabitEthernet0/0/0/1
service-policy output policy3
```

Short Pipe Mode: Egress PE Example

This example shows the configuration of the egress PE for the MPLS DiffServ short pipe mode. The egress LSR forwards the packet based on the precedence or differentiated services code point (DSCP) bits of the IP packet after removing the labels. The egress LSR does not copy the EXP bits to the precedence bits of the outgoing IP packet.

```
! Configuration is not needed for ingress interface

!Egress interface:
class-map prec4
match precedence 4
!
policy-map policy3
class prec4
bandwidth percent 40
random-detect precedence 4 100 ms 200 ms
!
interface GigabitEthernet0/0/0/1
service-policy output policy3
```

QoS on Multicast VPN

QoS on Multicast VPN: Example

Supporting QoS in an mVPN-enabled network requires conditional and unconditional marking of the DSCP or precedence bits onto the tunnel header. Unconditional marking marks the DSCP or precedence tunnel as a policy action. Conditional marking marks the DSCP or precedence values on the tunnel header as a policer action (conform, exceed, or violate).

Unconditional Marking

```
class-map c1
  match vlan 1-10

policy-map p1
 class c1
  set precedence tunnel 3
```

Conditional Marking

```
policy-map p2
 class c1

  police rate percent 50
  conform action set dscp tunnel af11
  exceed action set dscp tunnel af12
```

SIP 700 for the ASR 9000

The **set precendence tunnel** and **set dscp tunnel** commands are not supported but general Multicast VPN is supported, as shown in the following example.

QoS on Multicast VPN: Example

In this example, there are three services offered across the network: mobile, enterprise, and other services. Mobile traffic is classified as broadband 2G mobile traffic and 3G mobile traffic.

Control traffic needs the highest priority and has priority level 1. Broadband 2G mobile traffic has priority level 2. A priority queue is associated with each of these traffic classes. Traffic in these classes is policed at a rate of 100 percent, which means that full line rate bandwidth is dedicated to these traffic classes.

Remaining bandwidth is distributed across the Mcast_BBTV_Traffic class, Enterprise_Traffic class, and Enterprise_Low_Traffic class.

```
policy-map CompanyA-Profile
 class Control_Traffic
  priority level 1
  police rate percent 100
  !
 !
 class BB_2GMobile_Traffic
  priority level 2
```

```
 police rate percent 100
 !
!
class Mcast_BBTV_Traffic
 bandwidth remaining ratio 1000
!
class 3GMobile_Traffic
 bandwidth remaining ratio 100
!
class Enterprise_Traffic
 bandwidth remaining ratio 10
!
class Enterprise_Low_Traffic
 bandwidth remaining ratio 1
!
class class-default
!
end-policy-map
```

QoS on NxDS0 Interfaces

For QoS on NxDS0 interfaces, the shape, police, and queuing minimum rate is 8 kbps and granularity is 1 kbps. When QoS is applied to a low speed NxDS0 link, frame relay fragmentation (frf12) configuration is also recommended in order to provide low delay for real-time priority traffic. The common configurations on NxDS0 interfaces are:

- One-level policy applied to a main interface without Frame Relay configured
- Two-level policy applied to a subinterface with Frame Relay configured

One-Level Policy Applied to Main Interface: Example

```
show run int Serial0/2/1/0/1/1:0

Mon Aug  9 11:29:50.721 UTC
interface Serial0/2/1/0/1/1:0
 service-policy output fractional_T1_E1_policy □--------policy applied to serial interface
 encapsulation frame-relay
!

RP/0/RSP1/CPU0:viking-1#show run policy-map
policy-map fractional_T1_E1_policy
 class Conversational
  priority level 1
  police rate 64 kbps
  !
 !
 class Streaming-Interactive
  bandwidth remaining percent 35
 !
 class Background
  bandwidth remaining percent 15
 !
 class TCP-traffic
  bandwidth remaining percent 10
 !
 class class-default
 bandwidth remaining percent 40
```

```
 !
 end-policy-map
```

Two-Level Policy Applied to a Subinterface: Example

```
show run int Serial0/2/1/0/1/1:0

Mon Aug  9 11:29:50.721 UTC
interface Serial0/2/1/0/1/1:0
 encapsulation frame-relay
 frame-relay intf-type dce
!

Mon Aug  9 11:29:37.150 UTC
interface Serial0/2/1/0/1/1:0.16 point-to-point
 ipv4 address 192.1.1.1 255.255.255.0
 pvc 16
 service-policy output parent_policy □--------policy applied to serial subinterface
  encap cisco
  fragment end-to-end 350  □-------------------frf12 enabled
  !
 !
!

show run policy-map

policy-map parent_policy
 class class-default
  shape average rate 768 kbps

show run policy-map

policy-map fractional_T1_E1_policy
 class Conversational
  priority level 1
  police rate 64 kbps
  !
 !
 class Streaming-Interactive
  bandwidth remaining percent 35
 !
 class Background
  bandwidth remaining percent 15
 !
 class TCP-traffic
  bandwidth remaining percent 10
 !
 class class-default
 bandwidth remaining percent 40
 !
 end-policy-map
```

VPLS and VPWS QoS

To support QoS on virtual private LAN service (VPLS)-enabled and virtual private wire service (VPWS)-enabled networks, packets can be classified based on these match criteria:

- Match on vpls broadcast (applicable to VPLS)

- Match on vpls multicast (applicable to VPLS)
- Match on vpls control (applicable to VPLS)
- Match on ethertype arp (applicable to both VPLS and VPWS)

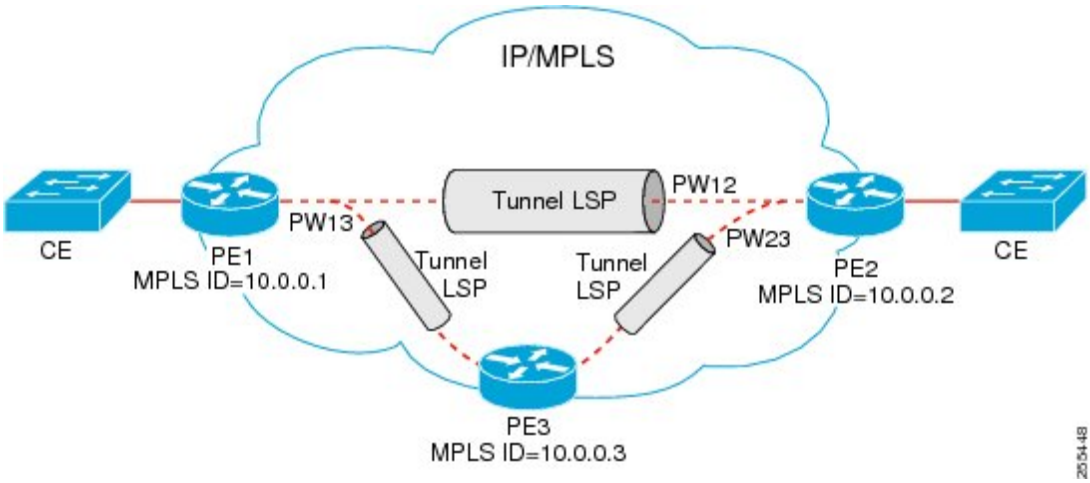
Note VPLS-specific and VPWS-specific classification are performed only in the ingress direction.

These guidelines apply to the VPLS and VPWS QoS feature:

- Supported on ingress Layer 2 bundle and nonbundle subinterfaces.
- Not supported on Layer 3 subinterfaces, but supported on ports with port inheritance policy. The system ignores VPLS classification on Layer 3 subinterfaces associated with the port.
- Match VPLS <control | multicast | broadcast> and match ethertype arp can be applied on a Layer 2 interface regardless of the Layer 2 service type, however VPLS <control | multicast | broadcast> classification is ignored on a non-VPLS Layer 2 interface type.

The following figure illustrates a typical VPLS topology. The VPLS network is a mesh of pseudowires (PWs) interconnected to bridge domains in the routers. Each of the provider edge (PE) routers has a bridge domain. Each PW is a bridge port into the bridge domain. The customer edge (CE) connection into each PE router is an attachment circuit (AC) bridge port into the same bridge domain. QoS configuration commands are applied to the AC that connects to the CE router on the one end and the bridge domain of the PE router on the other.

Figure 6: Typical VPLS Network Topology



VPLS and VPWS QoS: Example

This section contains a configuration example based on the components shown in VPLS and VPWS QoS, on page 20, and explains how the network matches packets based on the configured values.

The policy-map and PE-to-CE connection are configured as follows on the PE1 router:

```
class c1
  match vpls multicast
!
class c2
  match vpls broadcast
!
class c3
  match vpls control
!
class c4
  match ethertype arp
!
policy-map p1
  class c1
   set qos-group 3
   set mpls experimental imposition 4
   shape average percent 40
!
  class c2
   bandwidth remaining percent 10
   set mpls experimental imposition 5
!
  class c3
   police rate percent 10
   set mpls experimental imposition 6
!
  class c4
   bandwidth remaining percent 10
   set mpls experimental imposition 7
!
class class-default
!
end policy-map

interface GigabitEthernet0/2/0/0 l2transport
  description PE to CE connection
  service-policy input p1
!

l2vpn
 bridge group examples
  bridge-domain vpls-bridge
   interface GigabitEthernet0/2/0/0
   !
   vfi pe12link
    neighbor 10.0.0.2 pw-id 12
    !
   !
   vfi pe13link
    neighbor 10.0.0.3 pw-id 13
    !
   !
  !
 !
!
```

In the network designed and configured according to this example, and with VPLS and VPWS enabled, the packets that meet the match criteria receive QoS treatment according to the policy actions defined in the policy:

- If a VPLS multicast packet arrives on the ingress interface of the PE router, it matches class c1.

- If a VPLS broadcast packet arrives on the ingress interface of the PE router, it matches class c2.
- If a VPLS control packet arrives on the ingress interface of the PE router with MAC address ranging from 01-80-C2-00-00-00 to 01-80-C2-00-00-3F, it matches class c3.
- If an ARP packet arrives on the ingress interface of the PE router, it matches class c4.

Related Information

The information in this module focuses on the QoS implementation of features that are described in other technology guides. This table indicates the guides where you can find more information about these features.

Feature	Guide
802.1ad DEI	"Configuring Modular QoS Packet Classification and Marking" and "Configuring Modular QoS Congestion Management" in this guide
Frame Relay	*Cisco ASR 9000 Series Aggregation Services Router Interface and Hardware Component Configuration Guide* *Cisco ASR 9000 Series Aggregation Services Router Interface and Hardware Component Command Reference*
IP Header Compression	*Cisco ASR 9000 Series Aggregation Services Router Interface and Hardware Component Configuration Guide* *Cisco ASR 9000 Series Aggregation Services Router Interface and Hardware Component Command Reference*
L2VPN	*Cisco ASR 9000 Series Aggregation Services Router L2VPN and Ethernet Services Configuration Guide* *Cisco ASR 9000 Series Aggregation Services Router L2VPN and Ethernet Services Command Reference*
MLPPP/MLFR	*Cisco ASR 9000 Series Aggregation Services Router Interface and Hardware Component Configuration Guide* *Cisco ASR 9000 Series Aggregation Services Router Interface and Hardware Component Command Reference*
MPLS	*Cisco ASR 9000 Series Aggregation Services Router MPLS Configuration Guide* *Cisco ASR 9000 Series Aggregation Services Router MPLS Command Reference*
QoS on Multicast VPN	*Cisco ASR 9000 Series Aggregation Services Router Multicast Configuration Guide* *Cisco ASR 9000 Series Aggregation Services Router Multicast Command Reference*
QoS on NxDS0 Interfaces	*Cisco ASR 9000 Series Aggregation Services Router Interface and Hardware Component Configuration Guide* *Cisco ASR 9000 Series Aggregation Services Router Interface and Hardware Component Command Reference*

Related Information