CHAPTER 5

Implementing QoS

The rapid adoption of IP communications and the deployment of more complex and bandwidth-demanding applications led to a tremendous increase in network use. In a network with limitless resources, this would not be an issue. Although the technology is available to throw bandwidth and more powerful switching nodes at the problem, that is not always the best solution. A TCP session, for example, tries to use as much bandwidth as is available, to the detriment of other IP flows using the same links. Regardless of bandwidth, links can become congested due to various factors such as backing up failed paths in the network or handling the unpredictable traffic resulting from a security attack. At the same time, upgrading links and network nodes to handle higher bandwidths usually is an expensive proposition. Companies will try to make the most of the existent infrastructure and increase their return on investments (ROI). Concerted proactive measures can limit the probability that congestion will occur, but ultimately, traffic congestion is a fact of life in any network. Under such circumstances, the network operator must decide how to manage it and how to allocate the network resources based on traffic types. Congestion-management mechanisms should be considered regardless of the bandwidth available.

Networks provide a service to applications by transporting their data. Different application types have different expectations from this service; they demand a certain quality of service (QoS). These expectations are for the entire path of the traffic, making QoS an end-to-end concept. Applications such as interactive voice communications, audio, and video are sensitive to delay and delay variations (jitter), but they can afford to lose randomly a small percentage of the traffic. At the other end of the spectrum are the mission-critical applications that need reliable, no-loss data transfers.

The service level needs of any type of traffic can generally be quantified through a set of parameters such as the following:

- Service availability, which represents the percentage of uptime for the service
- Packet delivery ratio
- Round-trip time
- Jitter, which measures delay variations

For any given service, the target values for such parameters are listed in a service level agreement (SLA). The SLA can represent the internal network performance goals of an enterprise. It can also represent a contractual agreement between a service provider and its customers.

To meet the requirements of SLAs, the network has to be able to identify different types of traffic, to reserve bandwidth, to improve the loss characteristics, to avoid and manage congestion, and to prioritize the traffic. These functions are performed by routers and switches across the entire network in the context of an end-to-end QoS deployment model. Two such models are identified in the case of IPv4:

- **Integrated services (IntServ) (RFC1633)**—Rely on a signaling mechanism to reserve the necessary resource prior to forwarding the traffic across the network. This approach simulates the operational concepts of a circuit switched environment.
- **Differentiated services (DiffServ) (RFC 2475)**—Rely on policies that are defined on the network nodes and on packets being matched against and switched based on these policies.

Network elements leverage several mechanisms that enable them to support the implementation of these IP QoS models, as follows:

- **Classification and marking**—Classification is used to separate packets based on certain characteristics such as Source or Destination Address, predefined patterns of the 8 bits in the Type of Service (ToS) IP header field (IP precedence or differentiated services code point, DSCP) and higher-layer protocol information. The ToS field of a packet header can be overwritten by routers with a value relevant to the QoS policies defined in that network. This action on a packet is called *marking*.
- **Traffic conditioning (policing and shaping)—**The policing function enables the router to force inbound and outbound traffic to stay within a defined profile. Any traffic that does not observe the constraints of the profile is dropped. Through shaping, the router avoids downstream congestion by buffering traffic that does not fit a defined profile.
- **Congestion avoidance**—A mechanism that routers can implement to detect the possible buildup of congestion by monitoring the use of output buffers. If the buffers are getting full, the low-priority packets are dropped to save resources for the high-priority ones.
- **Congestion management**—The way the router handles an overflow of traffic. This functionality is implemented through various queuing algorithms.

Figure 5-1 schematically captures these mechanisms as they operate within a router.

These mechanisms are implemented the same way in both versions of IP. A number of books focus specifically on this topic, such as *IP Quality of Service,* by Srinivas Vegesna. Table 5-1 lists the QoS features supported by Cisco platforms both IPv4 and IPv6 along with layer 2 QoS features.

Table 5-1 shows that in Cisco products only a limited number of IPv4 QoS features are not available for IPv6 at the time of this writing.

Figure 5-1 *QoS Mechanisms in a Router*

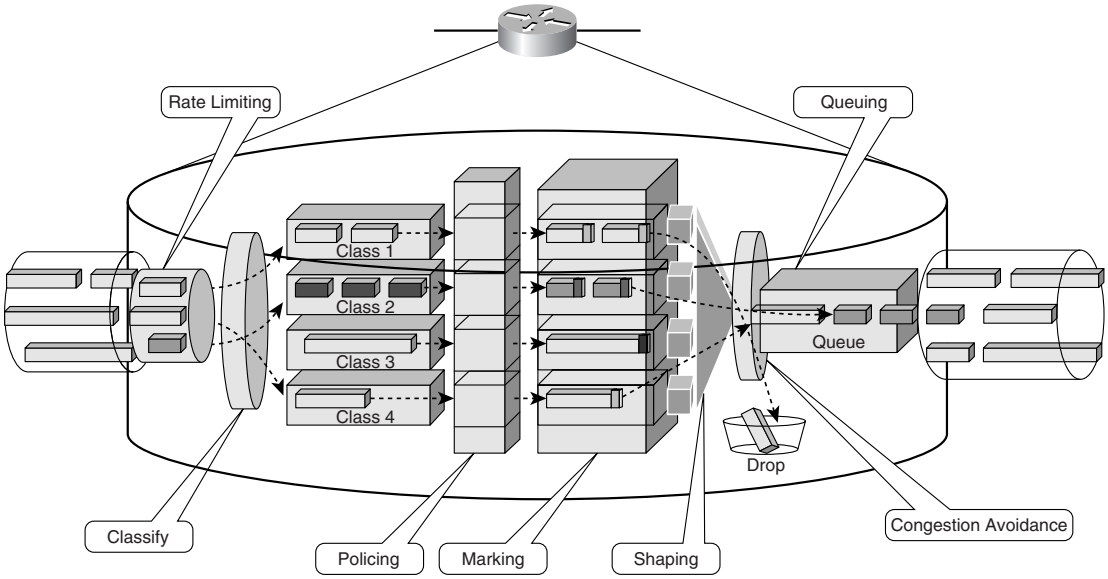


Table 5-1 *QoS Mechanisms and Their Implementation in Cisco Devices*

QoS Mechanism	Implementation Notes	IPv4	IPv6
Classification	Precedence	X	X
	DSCP	X	X
	Network-based application recognition (NBAR)	X	N/A
Marking	Class-based marking (CBM)	X	X
	Committed access rate (CAR)	X	
	Policy-based routing (PBR)	X	X
Policing and shaping	Rate limiting	X	X
	Class-based policing (CBP)	X	X
	Generic traffic shaping (GTS)	X	N/A
	Frame Relay traffic shaping (FRTS)	X	X
Congestion avoidance	Weighted random early detection (WRED)	X	X
Congestion management (queuing)	First in, first out (FIFO)	X	X

continues

Table 5-1 *QoS Mechanisms and Their Implementation in Cisco Devices (Continued)*

QoS Mechanism	Implementation Notes	IPv4	IPv6
	Priority queuing (PQ)	X	N/A
	Custom queuing (CQ)	X	N/A
	Flow-based weighted fair queue (FB-WFQ)	X	X
	Class-based weighted fair queue (CB-WFQ)	X	X
	Low-latency queue (LLQ)	X	X
	Modified Deficit Round Robin (MDRR)	X	X
Layer 2 QoS	ATM	X	X
	Frame Relay	X	X
	Ethernet 802.1p (CoS)	X	X
	Cable (DOCSIS)	X	N/A
Link-efficiency mechanisms	Compressed Real Time Protocol (cRTP)	X	N/A
	Link fragmentation and interleaving (LFI)	X	X

Because of the many similarities between IPv4 and IPv6 QoS, this chapter focuses on the few things that differentiate them today. This chapter covers the following topics:

- A review of differences between IPv6 and IPv4 QoS
- A discussion on the implementation of QoS for IPv6 over MPLS deployments
- Examples of configuring IPv6 QoS in a native environment and in an MPLS-based environment
- IPv6 QoS deployment considerations

It is assumed that the reader is familiar with fundamental concepts of IPv4 QoS. You can apply this knowledge directly toward deploying IPv6 QoS.

QoS for IPv6

It is difficult to understate the value of QoS in today's networks. Its importance is demonstrated by the fact that IPv4 QoS is deployed in more and more networks. Some hoped for QoS improvements in the next generation of the IP protocol, and some still believe that IPv6 is better than IPv4 in this respect. The reality is that neither evolutionary nor revolutionary changes were introduced in IPv6 QoS. QoS improvements in IPv6 are but a myth at this point. The same

concepts and same architectures apply to the new protocol, with a few small differences and implementation considerations that are worth mentioning.

Differences Between IPv6 and IPv4 QoS

QoS is implemented at both layer 2 and layer 3 of the protocol stack. This section discusses feature implementation and feature support differences between IPv4 QoS and IPv6 QoS at both layer 2 and layer 3. These differences revolve mostly around the traffic-classification process where packets or flows are differentiated through the use of various parameters such as IP source address, IP destination address, DSCP, or IP precedence values, and higher-level protocol types. Once classified, the packets can be processed according to a policy that reflects their service level. Differences between the two versions of the IP protocol could lead to different classifiers.

Layer 3 QoS

The QoS dedicated resource in the IPv4 packet header, the 8-bit ToS field, is mapped identically to the Traffic Class field in IPv6 and it is used in the same fashion. With IPv6, however, you must consider several additional classifiers, all related to the IPv6 packet header format:

- **Protocol type or version**—Because of the anticipated coexistence of the two protocols, it is worth considering the case where different service levels are applied to IPv4 and IPv6 traffic. The Protocol Type field can be used to distinguish the two protocols. Subsequently, more discrete classifications can be done for the traffic belonging to each protocol type.
- **Flow label**—The flow label is unique to IPv6 and was originally intended for use with resource-reservation–based QoS architectures. It was meant to allow routers to easily recognize a flow for which the resources were reserved. RFC3697 documents the flow label specifications. This field has the advantage of being located before the Source Address and the Destination Address fields, and that placement helps reduce lookup delays. Moreover, the flow label has an advantage over upper-layer classifiers: It is not lost when the packet load is encrypted. Some hosts' implementations (FreeBSD) do have the option of setting the Flow Label field for TCP connections, which can then be used on Cisco routers for classifying the flow.
- **Extension headers**—The extension headers substitute the functionalities of the IPv4 header options while keeping the main IPv6 header at a fixed size of 40 bytes. These additional elements could also be viewed as possible traffic classifiers, but this can increase uncontrollably the number of options. At this point, there does not seem to be a justification for implementing classification based on extension headers.

For the time being, among these IPv6-specific classifiers, the Protocol Type field is most used in deployments to differentiate between IPv4 and IPv6 traffic. The use of the other options is still being studied.

Other then these few elements that build on the packet format differences between the two protocols, IPv6 and IPv4 are similar as far as QoS is concerned. Table 5-1 shows some features missing from the Cisco implementation of IPv6 QoS at the time of this writing. These gaps are just a matter of implementation prioritization and schedule.

Layer 2 QoS

QoS is implemented at layer 2, as well. Moreover, layer 2 devices such as switches often integrate the QoS functionalities in hardware, allowing them to support a high-performance service. Therefore, layer 2 QoS represents an important element in the overall network deployment of QoS.

Various layer 2 technologies implement QoS in a specific way. ATM uses many of the features listed in Table 5-1 for individual virtual circuits (VC) or for bundles of VCs. Frame Relay relies on the Discard Eligible (DE) bit to deal with congestion. Ethernet with its expansion from enterprise networks to metropolitan-area and service provider networks has seen an increased interest in implementing differentiated services. Ethernet is leveraging its own QoS, as described by 802.1p. All these technologies implement QoS functions on a hop-by-hop basis, relying on concepts similar to those used by DiffServ at the IP layer.

A detailed discussion of layer 2 QoS is beyond the scope of this book. Moreover, the actual operation of these QoS mechanisms depends little on the IP protocol type. The dependency relates to the fact that layer 3 criteria and parameters are used to differentiate traffic in classes and mark it for the layer 2 QoS. This means that network elements performing classification and marking for layer 2 might have to be able to differentiate between the two versions of IP.

An example of such layer 2 QoS dependency on layer 3 information is that of cable-technologies QoS. Cable implements its own form of layer 2 QoS standardized through DOCSIS. The current version of Data Over Cable Service Interface Specifications (DOCSIS) 2.0 does not mention ways of classifying IPv6 packets. Therefore, IPv6 packets can be handled only in a Best Effort manner by cable networks. In congestion situations, IPv4 voice traffic would be properly prioritized, whereas IPv6 voice traffic would be impacted. At the time of this writing, proposals have been made to update the DOCSIS standard in its 3.0 revision to support IPv6 QoS. Cisco is actively participating in the CableLabs consortium defining the DOCSIS specifications.

Link-Efficiency Mechanisms

Time-sensitive applications have stringent requirements on end-to-end packet delivery. In the case of voice transport, for example, the end-to-end delay should not exceed 200 milliseconds. It is imperative to make sure the packets spend as little time as possible in the routers (processing delay). Moreover, time delay variations (jitter) can be particularly annoying in interactive voice communications. This leads to a second constraint in that the packets have to be delivered consistently. Assuming that the time-sensitive traffic is

appropriately prioritized for forwarding through other QoS mechanisms, one thing left to do in helping with the timely and consistent delivery of traffic is conditioning the traffic to better use the links. This conditioning is particularly important when traffic is switched from a fast interface such as 100-Mbps Ethernet to a slow interface such as 56-kbps Frame Relay. Two mechanisms are often used to optimize the link usage for time-sensitive traffic: Compressed Real Time Protocol (cRTP) and link fragmentation and interleaving (LFI).

The Real Time Protocol (RTP) supports audio and video applications over unicast or multicast. Its header, together with the IP and UDP header, represents a significant portion of the total packet (which leads to inefficient use of bandwidth, with serious consequences under congestion conditions). cRTP compresses the IP, the UDP, and the RTP headers in one single header with significant improvement in link utilization. The process is done by routers and it is done for each individual link, for each hop. The resources expended by the router on this process are sometimes justified by the service needs. In the case of IPv6, cRTP has to be aware of the IPv6 header format before performing the compression. As shown in Table 5-1, cRTP is not currently supported for IPv6 in IOS.

Large data packets can hold the smaller voice packets in the queue, leading to delays or delay variations that impact the voice quality. To deal with this problem, a router can fragment the large frames into smaller sizes and interleave them with the smaller ones carrying voice traffic. This feature is called *link fragmentation and interleaving*, or LFI. The optimization of the link utilization, although coming at a processing cost, can prove useful. This feature is independent of the IP protocol type being transported.

NOTE Note that with LFI, frames (layer 2) and not packets (layer 3) are being fragmented on lower-speed interfaces such as ATM and Frame Relay. The feature is transparent to the layer 3 protocols, and that is particularly useful in the case of IPv6 (where IP fragmentation is not permitted).

Differentiated Services

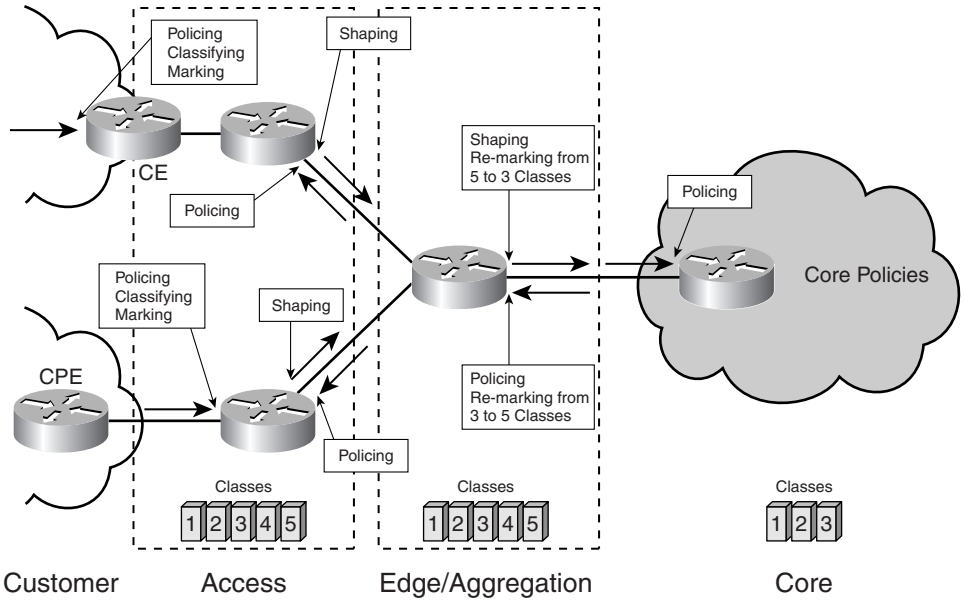
The premise of the QoS architecture presented in RFC 2475 is that as long as the traffic is categorized and marked, network nodes can assign resources to the various categories based on pre-established policies that reflect SLA requirements. When traversing a QoS DiffServ-enabled network, a packet goes through the following stages:

- The edge or access network elements categorize the packet based on layer 2 through 7 parameters. The packet is placed in one of few classes predetermined by the QoS design of the network.
- Based on the class it belongs to, the packet is marked by setting a certain value for the DSCP field. This marking makes it easy for the downstream nodes to handle the packet based on the class it belongs to. Reclassification and remarking is possible along the way.

- Each network element processes the packet based on the policies or a per-hop behavior (PHB) assigned for its class. This policy is implemented through mechanisms such as traffic conditioning and congestion management.

This hop-by-hop operation model makes DiffServ a scalable and easily interoperable approach to implementing QoS in both wide- and local-area networks. A number of classes are defined for each layer of the network. More classes are defined at the access layer, where the bandwidth resources are smaller, for a more granular traffic differentiation. In the network core, traffic can be aggregated in fewer classes. QoS mechanisms are leveraged by each network element to implement defined PHBs. Figure 5-2 presents a network-perspective example of a DiffServ-based QoS deployment. Each node implements appropriate queuing mechanisms and congestion-avoidance mechanisms.

Figure 5-2 *A Network Perspective of DiffServ Operation*



Support for IPv6

The IPv6 implementation of DiffServ is identical to IPv4. As mentioned in Table 5-1, some QoS features might not be ported to IPv6, but the available ones suffice to implement QoS in an IPv6 network.

The same classifiers can be used to differentiate both IPv6 and IPv4 packets, as follows:

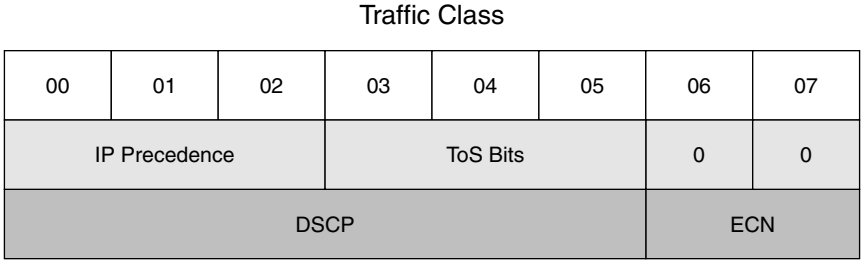
- Source IP address, destination IP address, IP Protocol field, source port number, and destination port number
- IP precedence or DSCP values

- TCP/IP header parameters, such as packet length
- Source and destination MAC addresses

The IPv6-specific classifiers that were discussed earlier in this chapter are not currently used.

The ToS field in the IPv4 packet header was more appropriately named Traffic Class in IPv6, but it is used in the same way: for packet marking and packet classification. The guidelines for using these 8 bits, also called the Differentiated Service (DS) field in the context of the DiffServ architecture, are standardized in RFC 2474. The formatting of the Traffic Class field in relation to the DSCP bits is shown in Figure 5-3. The figure also shows the original (RFC 791) segmentation of this field into IP precedence and ToS bits. The use of the last 2 bits of the Traffic Class, called explicit congestion notification (ECN), is defined in RFC 2481.

Figure 5-3 *DSCP and the IPv6 Traffic Class Field*



The DSCP marking of a packet is used by the router to classify it and implement the corresponding PHB. Several PHBs are standardized:

- Best Effort (BE).
- Expedited Forwarding (EF) is defined in RFC 2598 as a low-loss, low-latency, low-jitter, assured-bandwidth, end-to-end service.
- Assured Forwarding (AF) is defined in RFC 2597 and further detailed in RFC 3260. Four classes are defined, with three levels of drop precedence.

DSCP code points are matched to these PHBs, as shown in Table 5-2.

Table 5-2 *DSCP Code Points for Standardized PHBs*

PHB	Low Drop Precedence	Medium Drop Precedence	High Drop Precedence
BE	000000 (No Constraints)		
EF	101110	Low latency, low jitter, assured bandwidth	
AF1	001010 (AF11)	001100 (AF12)	001110 (AF13)
AF2	010010 (AF21)	010100 (AF22)	010110 (AF23)
AF3	011010 (AF31)	011100 (AF32)	011110 (AF33)
AF4	100010 (AF41)	100100 (AF42)	100110 (AF43)

Both IPv4 and IPv6 use the features listed in Table 5-1 to implement the PHBs. For example, the EF PHB would use the LLQ for its packets, whereas the AF PHBs could use CB-WFQ combined with a congestion-avoidance mechanism such as WRED that allows the router to drop inbound traffic based on precedence.

Configuration Example

The implementation similarities between IPv4 and IPv6 Diffserv QoS translate into configuration similarities. This section captures some of the IPv6 DiffServ concepts in configuration examples. The most relevant aspect of these configurations is the classification; everything else in terms of QoS configuration is IP version independent. The Modular QoS Command Line Interface (MQC) used for IPv4 is available for IPv6, too, with slight syntax changes. With MQC, three configuration steps are necessary to define and implement QoS on a router:

1. **Class map configuration**—Define the QoS classes that will be used with this deployment and the matching criteria for each of them. The number and type of classes used is driven by the SLAs that will be honored by the network. Their design can be different in various layers of the network (fewer in the core and more at the access and edge, for example).
2. **Policy map configuration**—Define the actions that the router should apply on the packets of each class.
3. **Apply the service policy**—Attach the policies defined through policy maps to the input or output traffic of an interface.

These steps are implemented in the following edge router QoS configuration example. The relevant aspects of each configuration line are highlighted. The highlights will help you connect the explanations in text with the configurations. First, four classes are defined on the router through the following class maps:

- EF-IPv6 is an EF class for the IPv6 traffic. The packets that belong to this class are identified by the protocol type and DSCP EF value.

```
class-map match-all EF-IPv6
 match protocol ipv6
 match  dscp ef
```

- EF-IPv4 is an EF class for the IPv4 traffic. The packets that belong to this class are identified by their DSCP EF value.

```
class-map match-all EF-IPv4
 match ip dscp ef
```

  The **ip** qualifier in the **match** command indicates that it is to be applied only to IPv4.

- AF1 is an AF class for both IPv6 and IPv4 traffic. The packets that belong to this class are identified by their DSCP AF11, AF12, AF13 values.

```
class-map match-all AF1
 match  dscp af11  af12  af13
```

The classification applies to both IP protocols because there is no matching done on the protocol type.

- Voice-Control is the class of packets that are of SIP type and that are intended for the server 2001:ABCD:EF:1::1.

```
class-map match-all Voice-Control
 match protocol sip
 match access-group name Control
```

One **match** statement applies to the SIP protocol, and the other is matching packets based on the access control list (ACL) named Control.

```
ipv6 access-list Control
 permit ipv6 any host 2001:ABCD:EF:1::1
```

The ACL identifies traffic destined to 2001:ABCD:EF:1::1.

NOTE Numbered ACLs are not supported with the **match access-group** command for IPv6.

The Cisco IOS implementation of QoS reserves a class called class-default for all traffic that does not meet the matching criteria of any of the other defined classes.

NOTE This example was designed to specifically highlight the fact that common classes can be used for IPv4 and IPv6 if the same policies are to be applied to both. On the other hand, when the QoS design is different for the two protocols, Cisco IOS software enables the user to distinguish between IPv4 and IPv6 by using distinct classes.

The next step is to instruct the router in this example on how it should handle the traffic in each of the classes identified in the previous step. These instructions represent the PHB for this network element. In this example, the router is required to police the inbound traffic for each class, based on the information provided in the ingress policy shown in Example 5-1.

Example 5-1 *Ingress Policy Configuration Example*

```
policy-map Ingress-Policy
 class EF-IPv6
  police cir 500000
    exceed-action drop
 class EF-IPv4
  police cir 1000000
    exceed-action drop
 class AF1
  police cir 100000
    exceed-action set-dscp-transmit default
```

For each class, a committed information rate (CIR) is defined in bits per second. The router drops the traffic that exceeds this CIR for each class except AF1. In the latter case, the router remarks the DSCP value of the offending packets to Best Effort (all bits set to 0). The policy is applied as inbound on the network-access-facing interface, as shown here:

```
interface FastEthernet6/2
 service-policy input Ingress-Policy
```

On the egress, each traffic class is reserved a certain percentage of the bandwidth and a certain amount of queue resources. The voice service control traffic to the server identified by the Control ACL is marked for expedited forwarding. All these instructions are captured in the Egress-Policy-Child shown in Example 5-2.

Example 5-2 *Child Egress Policy Configuration Example*

```
policy-map Egress-Policy-Child
 class EF-IPv6
  bandwidth percent 15
  queue-limit 512
 class EF-IPv4
  bandwidth percent 20
  queue-limit 512
 class AF1
  bandwidth percent 10
  queue-limit 1024
 class Voice-Control
  bandwidth percent 1
  queue-limit 100
  set dscp ef
```

MQC enables users to nest QoS policies and create complex PHBs. For example, Egress-Policy-Child is made part of an envelope policy called Egress-Policy-Parent that also shapes the traffic, as shown in Example 5-3.

Example 5-3 *Parent Egress Policy Configuration Example Integrating the Child Policy from Example 5-2*

```
policy-map Egress-Policy-Parent
 class class-default
  shape average percent 5 100 ms 100 ms
  service-policy Egress-Policy-Child
```

This parent policy is applied to the network-core-facing interface:

```
interface FastEthernet6/1
 service-policy output Egress-Policy-Parent
```

You can use the **show policy-map** command to review the policy maps. The same command directed to a specific interface provides a lot more useful detail on the policies applied to it. For the router in this example, the output for the network-core-facing interface is shown in Example 5-4.

Example 5-4 *Review of QoS Policies Applied to a Router Interface*

```
Router#show policy-map interface FastEthernet6/1
 FastEthernet6/1
  Service-policy output: Egress-Policy-Parent
    Class-map: class-default (match-any)
      521 packets, 59402 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: any
      Traffic Shaping
           Target/Average   Byte   Sustain   Excess    Interval  Increment
             Rate           Limit  bits/int  bits/int  (ms)      (bytes)
                5 (%)             100 (ms)    100 (ms)
          5000000/5000000   125000 500000    500000    100       62500
        Adapt  Queue     Packets   Bytes     Packets   Bytes     Shaping
        Active Depth                         Delayed   Delayed   Active
        -      0         521       59402     0         0         no
      Service-policy : Egress-Policy-Child
        Class-map: EF-IPv6 (match-all)
          0 packets, 0 bytes
          5 minute offered rate 0 bps, drop rate 0 bps
          Match: protocol ipv6
          Match:  dscp ef
          Queueing
            Output Queue: Conversation 137
            Bandwidth 15 (%)
            Bandwidth 750 (kbps) Max Threshold 512 (packets)
            (pkts matched/bytes matched) 0/0
        (depth/total drops/no-buffer drops) 0/0/0
        Class-map: EF-IPv4 (match-all)
          0 packets, 0 bytes
          5 minute offered rate 0 bps, drop rate 0 bps
          Match: ip dscp ef
          Queueing
            Output Queue: Conversation 138
            Bandwidth 20 (%)
            Bandwidth 1000 (kbps) Max Threshold 512 (packets)
            (pkts matched/bytes matched) 0/0
        (depth/total drops/no-buffer drops) 0/0/0
        Class-map: AF1 (match-all)
          0 packets, 0 bytes
          5 minute offered rate 0 bps, drop rate 0 bps
          Match:  dscp af11  af12  af13
          Queueing
            Output Queue: Conversation 139
            Bandwidth 10 (%)
            Bandwidth 500 (kbps) Max Threshold 1024 (packets)
            (pkts matched/bytes matched) 0/0
        (depth/total drops/no-buffer drops) 0/0/0
        Class-map: Voice-Control (match-all)
          0 packets, 0 bytes
          5 minute offered rate 0 bps, drop rate 0 bps
          Match: protocol sip
```

continues

Example 5-4 *Review of QoS Policies Applied to a Router Interface (Continued)*

```
          Match: access-group name Control
          Queueing
            Output Queue: Conversation 140
            Bandwidth 1 (%)
            Bandwidth 50 (kbps) Max Threshold 100 (packets)
            (pkts matched/bytes matched) 0/0
        (depth/total drops/no-buffer drops) 0/0/0
          QoS Set
            dscp ef
              Packets marked 0
        Class-map: class-default (match-any)
          521 packets, 59402 bytes
          5 minute offered rate 0 bps, drop rate 0 bps
          Match: any
```

Another useful command in troubleshooting IPv6 QoS is **show cef interface detail**. Cisco Express Forwarding (CEF) switching has to be enabled for the QoS features to operate on an interface.

The other important aspect of a complete QoS deployment is the implementation of congestion-avoidance and congestion-management mechanisms. The queuing mechanisms supported for IPv6 (FIFO, FB-WFQ, CB-WFQ, LLQ, MDRR) and the congestion-avoidance mechanisms (WRED) mentioned in Table 5-1 are configured in the same way as they are for IPv4.

As mentioned previously, layer 2 technologies employ various hop-by-hop QoS mechanisms, too. The layer 3 QoS discussed so far has the capability to modify parameters that are used in implementing certain PHBs by devices that operate at lower layers. The example presented in this section had the Egress-Policy-Child set an EF value for the DSCP field of packets in class Voice-Control. The **set** command provides options that enable the router to modify layer 2–relevant QoS parameters, too, as shown in Example 5-5.

Example 5-5 *QoS Options Available for the set Command*

```
Router(config-pmap-c)#set ?
  atm-clp         Set ATM CLP bit to 1
  cos             Set IEEE 802.1Q/ISL class of service/user priority
  discard-class   Discard behavior identifier
  dscp            Set DSCP in IP(v4) and IPv6 packets
  fr-de           Set FR DE bit to 1
  ip              Set IP specific values
  mpls            Set MPLS specific values
  precedence      Set precedence in IP(v4) and IPv6 packets
  qos-group       Set QoS Group
```

With the options highlighted in Example 5-5 enabled, a router modifies the marking of layer 2 frames regardless of the version of the transported IP packet.

As you can understand from the example in this section (if you are familiar with IPv4 QoS), other than classification-specific differences, DiffServ QoS is configured the same way for both IP protocols.

Integrated Services

The IntServ model operates similarly to circuit-switched networks. Prior to sending the traffic, resources are reserved across the entire path based on the service level required. This explains the natural mapping of IntServ to circuit-based network types such as ATM and Frame Relay. In this architecture, there are two sides to implementing QoS. One is a cross-network control plane that manages the reservation of resources, and the second is the traffic handling by each node based on the reservations made for it.

The control aspect of the IntServ is handled by the Resource ReSerVation Protocol (RSVP; RFC 2205). It is a control protocol similar to Internet Control Message Protocol (ICMP). In a nutshell, the operation of RSVP relies on two steps. First, the source of traffic sends a Path message to the destination, and on the way it collects resource information from the traversed nodes. Second, the receiver sends a response. The message is called Reservation, and it requests the resources needed by the application. This is a unidirectional process, so for a bidirectional flow it has to be started by each end. Through the reservation process, RSVP initiates and maintains soft state for each flow on all network elements that are traversed by it. This is, of course, a source of scalability concerns because routers will have to maintain state for a significant number of flows in large networks. Despite improvements made to the RSVP implementation, these concerns have slowed the adoption of the IntServ versus the DiffServ model.

After resources have been reserved for a given flow, routers have to recognize the traffic and assign to it the reserved resources. In performing these functions, network elements use some of the mechanisms listed in Table 5-1.

Support for IPv6

Differentiated handling of IPv6 traffic at the network element level is supported through the implementation of the various mechanisms listed in Table 5-1. Their availability in Cisco IOS software was discussed in the DiffServ section of this chapter. This leaves RSVP as the only missing piece necessary to support the IntServ model for IPv6 QoS.

RFC 2205 makes all the provisions necessary to support RSVP on top of IPv6, and they are similar to IPv4. The RFC also points out that the IntServ model can capitalize on the flow label that is characteristic to IPv6. The flow label could be used to efficiently mark the packets of a flow for the entire path reserved for it. RFC 2205 provisions support for the exchange of flow label information in IPv6. The flow label use with RSVP was envisioned as early as RFC 1809, and operational guidelines for its use were provided in RFC 3697; however, at the time of this writing, no implementations leverage it.

QoS services implemented based on the IntServ model are not common. Today's networks are more likely to leverage the high-bandwidth infrastructures along with DiffServ implementations. For these reasons, at the time of this writing, there is no implementation of RSVP for IPv6 in Cisco IOS software or other vendor products. Nevertheless, RSVP could be implemented in the future, justified by user demand and to address specific applications such as videoconferencing. It is also important to note that demand for its implementation might not be driven just by QoS. RSVP found its use in implementing other services such as label exchange in Multiprotocol Label Switching (MPLS) and in MPLS traffic engineering. The future IPv6-based MPLS implementations might drive the implementation of RSVP, too.

NOTE Note that IPv6 might leverage IPv4 RSVP during the deployment of 6PE and 6VPE, as described in the following section.

QoS for IPv6 over MPLS

MPLS does not define any new QoS architecture. DiffServ architectures as defined in RFC 2474 and RFC 2475 still apply in the MPLS environment. However, MPLS has a number of characteristics and specific mechanisms that enable unique capabilities. For these reasons, RFC 3270 was written to describe MPLS support of DiffServ. This specification allows support of DiffServ for both IPv4 and IPv6 traffic transported over an MPLS network. Using DiffServ in an MPLS environment for IPv6 traffic is not different regardless of whether the LSP is set up using IPv4 protocols (this is the case for 6PE and for 6VPE) or set up using IPv6 protocols. In fact, there are no interactions at all between MPLS DiffServ, which deals with the MPLS shim layer, and the LSP setup mechanisms itself; this enables DiffServ to be used indistinctly on traffic using label paths set up by LDP IPv4, LDP IPv6, RSVP, or even manually set up.

MPLS-TE offers the means to set up paths with explicitly reserved bandwidth across an MPLS core. The RSVP signaling used by MPLS-TE can be IPv4 or IPv6. At the time of this writing, IPv6 RSVP is not yet supported on Cisco routers. However, because 6PE and 6VPE use IPv4-signaled LSP, they benefit seamlessly from IPv4 MPLS-TE setup mechanisms, including IPv4 RSVP. However, the mechanisms to achieve tunnel selection for certain IPv6 traffic may be IPv6 specific. This is the case, for instance, when the operational choice is to have tunnels dedicated on a per–network layer basis. Those mechanisms are detailed in the RSVP-TE section.

The next sections cover DiffServ and MPLS-TE in the 6PE/6VPE context and provide configuration examples for both.

Using DiffServ in a 6PE or 6VPE Environment

MPLS DiffServ can extend the IP DiffServ mechanisms to allow service providers to deliver QoS-based service without interfering with the customer's traffic marking. In the MPLS network, QoS information is carried in the MPLS shim header as described in RFC 3270 and shown in the Table 5-3.

Table 5-3 *MPLS Shim Header*

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
Label																				Exp			S	TTL							

Label: Label Value, 20 bits

Exp: Experimental Use, 3 bits

S: Bottom of Stack, 1 bit

TTL: Time To Live, 8 bits

The MPLS header is 4 bytes, out of which 3 bits are used for DiffServ and referred to as the Exp field. These bits help define the QoS treatment (PHB) that a node should give to a packet. You can think of this as a sort of hierarchical PHB, where the Exp field is used for QoS treatment inside the MPLS core, and at the egress boundary, whereas DSCP is used outside the core, and at the ingress boundary.

Using DiffServ on the MPLS ingress boundary for differentiating IPv6 traffic is almost exactly the same as using it outside an MPLS environment. Packets need to be classified, which may take place prior to reaching the MPLS edge router or at the MPLS edge router itself. In both cases, the MPLS Exp field must be filled, either by simply copying it from the DSCP field or by applying any sort of classification rule. Because the Exp field is only 3 bits (compared to 6 bits for DSCP), you can just copy the class-selector bits into the Exp field, or decide to "map" the DSCP into the Exp field based on a predefined scheme. The former is the default behavior and good enough in environments where the CEs are managed by the service provider or simply trusted to deliver DSCP classes directly valid in the core network. The latter is used when the CE is untrusted and can also provide some flexibility to distinguish core classes based on dropping precedence. However, it requires additional packet classification and conditioning at the PE (6PE or 6VPE). This is sometimes referred to as the *pipe model*. In some cases, the Exp bits can even be used exclusively to encode the dropping precedence.

NOTE Note that 3 bits allow up to 8 classes in the core, which so far has been plentiful in known core QoS deployments.

Configuration Example

The following example shows how you can implement DiffServ in a 6PE or 6VPE network. Let's assume that the CE is unmanaged and does not always set the proper values for the DSCP field. Therefore, the SP wants to classify and mark explicitly the MPLS Exp field for traffic coming from this CE.

The first step is to classify incoming traffic from the CE. This is done using the following ALCs that identify the following:

- RTP traffic under the UDP ports 16383 or 16384
- Traffic to 2001:300::/64 or with flow label set to 1111
- BGP traffic

Example 5-6 shows the configuration of the ACLs identifying these three traffic types.

Example 5-6 *Access Lists Identifying Three Traffic Types*

```
ipv6 access-list RTP
 permit udp any any eq 16383
 permit udp any any eq 16384
!
ipv6 access-list BUSINESS
 permit ipv6 any 2001:300::/64
 permit ipv6 any any flow-label 1111
!
ipv6 access-list RP
 sequence 20 permit tcp any eq bgp any
```

On the 6PE, the following four access classes are defined:

1. VoIP
2. Business Data
3. Management Traffic
4. Internet

The Internet class uses the default class, which is always present on Cisco routers. The traffic for classes 1 to 3 is classified based on the DSCP values and access lists above, using the class maps shown in Example 5-7.

Example 5-7 *Configuration of the Four Access Classes Defined for the 6PE QoS Example*

```
class-map match-any voip
 match  dscp cs5
 match access-group name RTP
!
class-map match-any business
 match access-group name BUSINESS
 match  dscp af31
 match  dscp af32
 match  dscp af33
!
```

Example 5-7 *Configuration of the Four Access Classes Defined for the 6PE QoS Example (Continued)*

```
class-map match-any management
 match  dscp 6
 match access-group name RP
```

In the MPLS core network, another four classes (could be fewer) are defined, and specific Exp fields are associated to each:

1. Real-Time (RT) for Voice and Video (EXP=5)
2. Business Class data (BU) for golden customers or golden applications (EXP=3)
3. Control Traffic (CTRL) for BGP and SNMP (EXP=6)
4. Internet data (STD) for anything else (EXP=0)

Note that the MPLS core classes apply to both IPv4 and IPv6 traffic, and in most cases preexist to the enabling of 6PE or 6VPE. A policy map is configured referring the set of classes defined on the access, and setting the Exp field in labels imposed at the 6PE/6VPE as shown in Example 5-8.

Example 5-8 *Policy Identifying the Exp Bits to be Assigned to Tagged IPv6 Packets Belonging to the Four Access QoS Classes*

```
policy-map CE1
 class voip
  set mpls experimental imposition 5
 class management
  set mpls experimental imposition 6
 class business
  set mpls experimental imposition 3
 class class-default
  set mpls experimental imposition 0
```

The policy map is then applied inbound to the 6PE interface facing the CE:

```
interface Serial0/0
 ip address 50.1.1.2 255.255.255.0
 service-policy input CE1
 ipv6 address 2001:10::72B/64
```

Core policies are also defined on the 6PE router, consistent with the ones on each P-router in the core, as shown in Example 5-9.

Example 5-9 *Configuration of the Four Core Classes and the Corresponding Policies Defined for the 6PE QoS Example*

```
class-map match-any RT
 match mpls experimental topmost 5
!
class-map match-any MNGT
 match mpls experimental topmost 6
class-map match-any BUS
```

continues

Example 5-9 *Configuration of the Four Core Classes and the Corresponding Policies Defined for the 6PE QoS Example (Continued)*

```
 match mpls experimental topmost 3
!
policy-map TO-CORE
class RT
   police 64000
  priority 64
!
class MNGT
  bandwidth 10
  police cir 10000
    exceed-action drop
!
class BUS
  bandwidth 154
  police cir 154000
    conform-action transmit
    exceed-action drop
```

The policy is applied outbound on the core-facing interface, as shown in Example 5-10.

Example 5-10 *Applying QoS Policies on the Core-Facing Interface of the 6PE Router*

```
interface Serial2/0
 ip address 40.1.1.2 255.255.255.0
 ip router isis
 service-policy output TO-CORE
 tag-switching ip
```

At the egress 6PE or 6VPE, the DSCP field can be overwritten based on specific policies, including the value of the Exp field, or simply left to the value they had at the ingress PE, before entering the MPLS network.

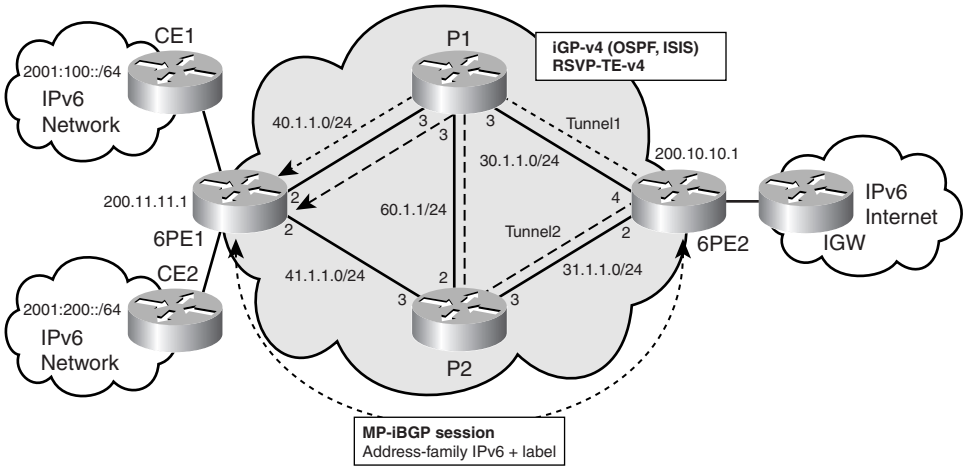
Note that in the preceding example, the PE is explicitly setting the Exp field. In fact, in most cases, the SP will just carry the DSCP field, with the advantage that no classification needs to take place at the PE. So the ingress input policies and corresponding policy map (CE1) are not necessary.

Using RSVP-TE in a 6PE or 6VPE Environment

IP routing protocols are not good at optimizing network utilization and performance. Although they can recover from network failures, they typically select the shortest path to a destination, ignoring other paths. Amazingly, loop-avoidance mechanisms rely significantly on the assumption that all nodes within a routing area have a consistent view of the topology, hence will be using the same path to the destination. Shortest-path routing often leads to unbalanced traffic distribution across the network, creating congestion hot spots in some areas, while some links are underutilized.

One well-known issue related to the topic is the so-called *fish problem*, named after the shape of the typical topology that illustrates it. In Figure 5-4, traffic flowing from 6PE2 to 6PE1 and beyond will tend to use only one path (for instance, via P1).

Figure 5-4 *Using RSVP-TE with 6PE*



Load balancing may sometimes help improve the traffic distribution, but is certainly not the panacea, particularly when unequal cost paths are available. In the most common case, label binding is based on routing information, and MPLS performs destination-based forwarding. However, by enabling source-routing-like mechanisms via the label switch path (LSP), MPLS offers good opportunities to resolve the issue of bandwidth optimization. When more flexible (or rather more controlled) forwarding policies are required, RSVP-TE provides alternative for label binding, other than exclusively based on routing information. With RSVP-TE, you can define forwarding policies at a granularity of a flow or group of flows. This technology is referred to as MPLS *traffic engineering.* One or many tunnels are set up between edge routers (PEs) or between core routers (Ps), explicitly or automatically, based on available and required bandwidth. The traffic between tunnels' endpoints can then select a particular tunnel based on a variety of criteria. Those criteria encompass different strategies:

- Applications driven (for instance, by looking at UDP or TCP port numbers)
- Customers driven (for instance, by looking at source/destination, or at the incoming interface [at PE-CE boundary])
- Network layer driven (for instance, IPv4 or IPv6)
- QoS driven (using the DSCP or Exp fields, respectively, in the IP [IPv4 or IPv6] or in the MPLS headers)

One common MPLS-based approach for traffic engineering is the overlay model. Service providers build a virtual network that includes a full mesh of logically connected PEs.

These logical connections can be MPLS explicit routes enforced via bandwidth reservation. The MPLS LSPs explicitly set up using RSVP-TE can be used to replace the LSP routes provided by the combination of an interior gateway protocol (IGP) and LDP to enable more efficient resource utilization.

Both Chapter 3, "Delivering IPv6 Unicast Services" and Chapter 7, "VPN IPv6 Architecture and Services," review in detail the way IPv6 traffic can use IPv4-signaled LSPs. A direct consequence is that whichever mechanism is available at the IPv4 LSP level can be beneficial to 6PE and 6VPE. As far as MPLS-TE is concerned, RSVP-TE can be used in the IPv4-based MPLS core to optimize the bandwidth and for fast-reroute purposes. The tunnels are set up exactly as they would be for providing the service to IPv4 traffic. In fact, it is recommended that the tunnels be set up for both IPv4 and IPv6 at the same time because the criteria to select these tunnels have less to do with the network layer than with applications, type of traffic, or specific customer.

There are (at least) two methods for selecting the RSVP-TE tunnels for IPv6 (6PE or 6VPE) traffic:

- The egress 6PE (or 6VPE) could advertise several different BGP next hops for different sets of reachable destinations. These next hops (IPv4-mapped IPv6 addresses) can then be used as the tail end of RSVP tunnels by the ingress 6PE.
- The ingress 6PE could set up multiple TE tunnels to the egress 6PE and select one or the other based on MPLS QoS after label imposition.

Examples of each are provided in the next sections.

Using Multiple BGP Next Hops

The egress 6PE can explicitly set up the next hop announced in BGP updates, based, for instance, on prefixes being advertised. The following configuration provides an example of this technique.

A route map is defined that sets different next hops for different match criteria as shown in Example 5-11.

Example 5-11 *Route Maps and Corresponding Access Lists Identifying the TE Tunnel-Selection Criteria*

```
PE1#
ipv6 prefix-list list-te1 seq 5 permit 2001:100::/64
!
ipv6 prefix-list list-te2 seq 5 permit 2001:200::/64
route-map NH-TE-SELECT permit 10
 match ipv6 address prefix-list list-te1
 set ipv6 next-hop ::FFFF:200.8.8.1
!
route-map NH-TE-SELECT permit 20
 match ipv6 address prefix-list list-te2
 set ipv6 next-hop ::FFFF:200.9.9.1
```

The route map is applied on prefixes announced by the 6PE, as shown in Example 5-12.

Example 5-12 *Applying the Route-Maps Defined in Example 5-11 to the Prefixes Advertised by the 6PE Router*

```
PE1#
router bgp 100
 neighbor 200.10.10.1 remote-as 100
 neighbor 200.10.10.1 update-source Loopback0
 !
address-family ipv6
 neighbor 200.10.10.1 activate
 neighbor 200.10.10.1 route-map NH-TE-SELECT out
 neighbor 200.10.10.1 send-label
```

At the ingress PE, two RSVP-TE tunnels are set up, as shown in Example 5-13.

Example 5-13 *Two TE Tunnels Configured on the PE Router*

```
interface Tunnel1
 ip unnumbered Loopback0
 tunnel destination 200.11.11.1
 tunnel mode mpls traffic-eng
 tunnel mpls traffic-eng priority 7 7
 tunnel mpls traffic-eng bandwidth  10
 tunnel mpls traffic-eng path-option 1 explicit name te1
 tunnel mpls traffic-eng record-route
!
interface Tunnel2
 ip unnumbered Loopback0
 tunnel destination 200.11.11.1
 tunnel mode mpls traffic-eng
 tunnel mpls traffic-eng priority 7 7
 tunnel mpls traffic-eng bandwidth  10
 tunnel mpls traffic-eng path-option 1 explicit name te2
```

In this example, the tunnel paths are explicitly configured as shown in Example 5-14.

Example 5-14 *Explicit Configuration of the Tunnel Paths*

```
ip explicit-path name te1 enable
 next-address 30.1.1.3
 next-address 40.1.1.2
!
ip explicit-path name te2 enable
 next-address 31.1.1.3
 next-address 41.1.1.2
```

And static routes are configured to select the tunnel based on the next hop received from the egress PE (PE1):

```
ip route 200.8.8.1 255.255.255.255 Tunnel1
ip route 200.9.9.1 255.255.255.255 Tunnel2
```

At the ingress PE2, you can display the detailed LSP used by the forwarding plane (CEF) for destination 2001:100::/64 and 2001:200::/64. The output of the **show ipv6 cef** command provides the following information:

Example 5-15 *IPv6 CEF Information for Prefix 2001:100::/64*

```
PE2#show ipv6 cef 2001:100::/64 internal
  recursive via 200.8.8.1[IPv4:Default] label 17, fib 024AC3E8, 1 terminal fib
    attached to Tunnel1, adjacency IP midchain out of Tunnel1 027F4918
  output chain: label 17 TAG midchain out of Tunnel1 027F47A8 label 16 TAG adj out
   of Ethernet0/0, addr 30.1.1.3 027F4D68
PE2#show ipv6 cef 2001:200::/64 internal
  recursive via 200.9.9.1[IPv4:Default] label 16, fib 024AC360, 1 terminal fib
    attached to Tunnel2, adjacency IP midchain out of Tunnel2 027F44C8
  output chain: label 16 TAG midchain out of Tunnel2 027F4358 label 16 TAG adj out
of Ethernet2/0, addr 31.1.1.3 027F4A88
```

Note that the two paths are distinct from the start, one via interface Ethernet0/0 and the other via interface Ethernet2/0.

COS-Based TE Tunnel Selection (CBTS)

The idea of CBTS is to allow different parallel tunnels between the same head end and the same tail end to each carry a different subset of the class of service (CoS).

At ingress PE, the CBTS configuration involves the following:

- Create multiple TE tunnels with the same head end and same tail end. Existing TE attributes are configured completely independently for each tunnel (bandwidth, bandwidth pools, preemption priorities, path options, and so on).
- Indicate on each of these tunnels which DiffServ code points are to be transported.
- Make these tunnels visible to routing. This can be achieved using Autoroute (every prefix via the tail end of the tunnel is routed via the tunnel) or using static routes pointing to the tunnels.

At the ingress PE, one tunnel is then selected over the other based on the Exp field value found in the topmost label of each packet. As previously discussed, packets arriving unlabeled (IPv6 packets) at ingress 6PE can be classified and marked with a particular Exp value while the label stack is being imposed, so that CBTS *at the same node* can then use this value to further select the tunnel matching the corresponding code points.

In the following example, tunnels (tunnel1 and tunnel2) are configured the same way as in Example 5-13, and again, CEF has parallel paths to get to a particular destination prefix, via tunnel1 or tunnel2. But in CBTS case, each tunnel gets the additional command highlighted in Example 5-16.

Example 5-16 *Tunnel Configuration Example in the CBTS Scenario*

```
interface Tunnel1
 ip unnumbered Loopback0
 tunnel destination 200.11.11.1
 tunnel mode mpls traffic-eng
 tunnel mpls traffic-eng priority 7 7
 tunnel mpls traffic-eng bandwidth  10
 tunnel mpls traffic-eng path-option 1 explicit name te1
 tunnel mpls traffic-eng record-route
 tunnel mpls traffic-eng exp 3
!
interface Tunnel2
 ip unnumbered Loopback0
 tunnel destination 200.11.11.1
 tunnel mode mpls traffic-eng
 tunnel mpls traffic-eng priority 7 7
 tunnel mpls traffic-eng bandwidth  10
 tunnel mpls traffic-eng path-option 1 explicit name te2
 tunnel mpls traffic-eng exp 5
```

Then, as in the DiffServ example, traffic is classified and the Exp field is set, so that some of this traffic is put into tunnel1 and some into tunnel2 (see Example 5-17).

Example 5-17 *Configuration of Policies Setting the Exp Bits Used for Tunnel Selection in the CBTS Scenario*

```
policy-map CE1
 class tunnel1
  set mpls experimental imposition 3
 class tunnel2
  set mpls experimental imposition 5
!
interface Serial0/0
 ip address 50.1.1.2 255.255.255.0
 service-policy input CE1
 ipv6 address 2001:10::72B/64
```

In this example, classification (class map for class tunnel1 and tunnel2) is not shown, but would be similar to the examples detailed in DiffServ sections. The CE1 policy is applied inbound on the CE-facing interface.

NOTE At the time of this writing, CBTS, when used in conjunction with 6PE or 6VPE, is not yet available on Cisco routers. The feature will become available in the near future.

Deploying QoS for IPv6

QoS is always enabled in an environment that already provides unicast connectivity. The tools and features used to enable this service depend on the characteristics of this infrastructure. IPv6 most likely is the newcomer in a preexistent IPv4 network. Chapter 3 discussed the IPv6 unicast deployment options and how the QoS design has to be tailored to them. This section looks at design recommendations for native and MPLS-based IPv6 deployments. Inevitably, the coexistence of the two protocols has to be addressed, too.

QoS in a Native IPv6 Deployment

The similarity between the implementations of QoS for the two versions of the IP protocol implies that the IPv4 QoS deployment rules are applicable to IPv6, too. You can find a detailed analysis of these design rules in the book *End-to-End QoS Network Design: Quality of Service in LANs, WANs, and VPNs,* by Tim Szigeti and Christina Hattingh. The basic steps to follow in planning for and designing a QoS deployment are as follows:

1. The most important first step that must be taken before designing or even considering the deployment of QoS is that of defining its business objectives network. A DiffServ option is discussed here for the QoS deployment.

   Note IntServ might be considered for applications such as videoconferencing that benefit from reserving resources prior to sending the traffic. The IntServ implementation most likely would be an overlay on top of DiffServ used for most traffic types.

2. A thorough evaluation of the network capacity is required. The identified constraints imposed by the existent infrastructure provide the framework for prioritizing the business objectives and the resources to be allocated to them.
3. The outcome of the first two steps is the set of classes that are used throughout the network to differentiate the traffic types. Three to five classes typically suffice for most deployments; too many classes make the service difficult to manage.
4. Classify the traffic as close as possible to the source and mark it using DSCP.
5. Define the PHBs for the various devices in the network. The devices that can perform the QoS functions in hardware should be more heavily used in the design. Policing should be performed as close to the source as possible.

Note QoS classification and marking can and should be done prior to it being encrypted and forwarded through tunnels.

6. Test and verify the QoS design before deployment.

The network layers usually support the end-to-end QoS in different ways:

- **Access**—This layer provides the best opportunity to enforce customer traffic profiles through policing and rate limiting. This layer is also the closest to the sources of traffic and it represents the best place to classify and mark packets. The access layer might use more classes than other layers of the network to sort more finely customer traffic. The QoS function can be done in both layer 2 and layer 3 devices.
- **Edge/aggregation**—The traffic that reaches the edge/aggregation layer is typically marked and conditioned by the access layer. At this point, the traffic might be policed again, but it will be shaped before entering the core. The number of classes is reduced to the recommended three to five, and hence the packets might be remarked.
- **Core**—In the past, a common approach was to rely on a high-bandwidth core that would be engineered so that it would not see congestion (thus rendering the use of QoS unnecessary). Even today, some argue that in SP environments that it is cheaper to oversize the core network than to design it with QoS. When network bandwidth starts to be scarce, it is more convenient to increase link capacity than to deploy any sort of mechanism that brings complexity and maintenance overhead. Making the core aware of application, protocols layers (for instance, IPv4 and IPv6), customers, and so on may not be considered as a scalable thus viable solution. In such environments, only edge policies will be configured. However, this general philosophy may turn to be expensive for a number of reasons, including the following:
  - The network must accommodate peaks, sometimes an order of magnitude more than the average traffic. Turning on QoS in the network to differentiate traffic may prove to be more economic than oversizing the network.
  - The network must be able to react well to link failures, where traffic is rerouted on alternate paths. When the failure occurs on a large trunk, and/or when the failure affects several trunks, some congestion can build on smaller links, which were not sized for absorbing the rerouted traffic.
  - Denial-of-service attacks can always saturate links. QoS allows some time-sensitive traffic such as voice to work perfectly even at the peak of a denial-of-service incident.
  - The implementation of QoS in the core is probably the simplest. The traffic is already conditioned by the edge/aggregation and only a few (equal to the number used in the edge/aggregation layer) classes are used.

Similar to IPv4, congestion control and management are implemented depending on the characteristics of the traffic identified through the various classes.

Layer 2 QoS should be leveraged as much as possible throughout the network because switches often implement it in hardware. Usually a mapping of the top three DSCP bits or the TOS bits to COS is sufficient in terms of marking. However, the option is available to have independent marking policies.

QoS in an MPLS-Based IPv6 Deployment

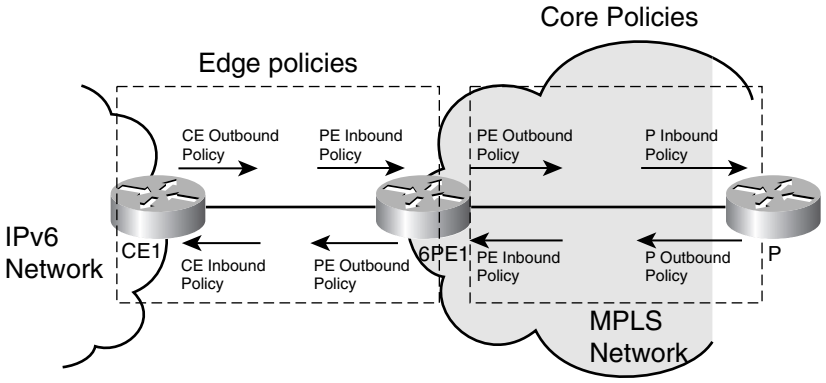
The SLA and its associated network-deployed enforcing mechanism (QoS) are end-to-end concepts. Therefore, it is logical that it is implemented primarily by service providers, who are the ones with the end-to-end perspective. Because MPLS is one of the dominant technologies deployed in SP networks, it is also one of the primary areas for QoS implementation.

Similar to native IP QoS, MPLS QoS technologies can be split into IntServ and DiffServ. The latter is the most widely deployed mechanism, but one starts to see some deployments with MPLS traffic-engineered tunnels, to provide guaranteed bandwidth across the MPLS core. MPLS-TE uses RSVP-TE to set up a labeled path across the MPLS domain. MPLS DiffServ and MPLS-TE can be combined together into an even more powerful tool for delivering QoS on packet networks. Most of the MPLS QoS deployments focus on the network edge, although some service providers are starting to deploy QoS in the MPLS core, too.

Both 6PE (IPv6 MPLS over v4-signalled Label Switch Path, described in Chapter 3) and 6 VPE (BGP MPLS IPv6 VPN over v4-signalled LSP, described in Chapter 7) involve dual-stack edge routers communicating over a v4-based MPLS backbone. To enable QoS for 6PE and 6VPE, IPv6 traffic must be classified and conditioned (policed, shaped, and marked) pretty much the same way as IPv4, before entering the MPLS backbone. This typically must take place at the customer edge (CE) routers in the case of a managed service or at the provider edge (PE) router otherwise. This implies identifying IPv6 classes, then performing policing, shaping, and marking. In Figure 5-5, those tasks are referred to as *edge policies*. In practice, either the CE outbound policy or the PE inbound policy is set up: CE inbound policy is never used. In addition, in case QoS is also implemented in the MPLS core, a PE outbound policy is configured together with outbound policies at each hop (P-routers) in the MPLS provider core. This is the PHB, which, in this case, is essentially queuing and dropping.

MPLS is the foundation of a multiservice network, which is intended to transport a large variety of network layer protocols (IPv4 unicast and multicast, IPv6 unicast and multicast, IPv4 VPN, IPv6 VPN, as well as ATM, FR, PPP, Ethernet, and so on) and a variety of application data (Internet content, VoIP, video, and so on). When deploying QoS in the MPLS core, one does not expect to make it aware of the network layer protocol or of the application being carried. In other words, it is unlikely and even not recommended to differentiate IPv6 traffic from IPv4, but rather to treat equally IPv4 and IPv6 real-time traffic, and differentiate them from IPv4 and IPv6 data traffic, for instance.

Figure 5-5 *Deploying IPv6 QoS over MPLS*



In that context, enforced by the 6PE (and 6VPE) approach where the same LSP used to transport IPv4 and VPNv4 traffic is also used to transport IPv6 traffic, the setting of core QoS for the latter is straightforward. It is just a matter of classifying the 6PE/IPv6 traffic into existing MPLS classes, and marking MPLS-encapsulated IPv6 traffic accordingly.

NOTE There is no difference regarding DiffServ in an MPLS-based IPv6 deployment, whether the LSP was set up using IPv4 (6PE and 6VPE) or IPv6 or whether it was setup using LDP, RSVP, or even BGP.

Instead of, or in addition to DiffServ, there may be some interest in deploying an IntServ strategy across the core (for instance, to reserve some paths for specific customers, or to optimize a better use of the network bandwidth). This can be done using MPLS-TE.

MPLS, combined with RSVP, provides a mechanism to set up explicit paths across the core, and to associate them with bandwidth reservation or even DiffServ strategies. Despite the management complexity, some service providers with an MPLS infrastructure have meshed their core with TE tunnels and manually set up paths with reserved bandwidth. In that context, DiffServ can also be used to manage congestion conditions inside the TE tunnels themselves.

When using MPLS-TE in networks where IPv4 and IPv6 traffic coexist, two approaches are possible: Either TE tunnels are shared for IPv4 and IPv6 traffic (recommended) or separate tunnels are signaled and used for the two protocol versions. TE tunnels can be combined with DiffServ, where the selection of tunnels is performed based on DSCP bits. This is referred to as CBTS. DiffServ-aware traffic engineering enables service providers to perform separate admission control and separate route computation for discrete subsets of traffic (for example, voice and data traffic, regardless of the network layer protocol).

NOTE When IPv6-RSVP is available, it might still make sense to use a single MPLS-TE setup mechanism (IPv4 or IPv6) to set up a single tunnel used by both IPv4 and IPv6 for certain traffic classes.

IPv4 and IPv6 Coexistence

One question arises with the coexistence of IPv4 and IPv6: Should IPv6 traffic be differentiated from IPv4 traffic, or rather should traffic of a given class (for instance, real time) be differentiated from traffic of other classes (for instance, data) regardless of IP protocol type? Not only are both approaches possible, but different strategies can also apply to different parts of the network.

The PHBs for the two protocols might be different under the following considerations:

- IPv4 traffic is revenue generating, and it most likely is more important for the business than the IPv6 traffic, at least in the beginning. In that case one might choose to prioritize the IPv6 traffic lower than IPv4 and provide fewer resources for it.
- IPv4 and IPv6 traffic might have to observe different PHBs depending on traffic patterns used by various applications.

In these cases, you should define different classes and policies for each traffic type.

NOTE With transition mechanisms, the IPv6 traffic can leverage the deployed QoS of the traversed IPv4 infrastructure. In some circumstances, the IPv6 traffic might also lose its markings after crossing the IPv4 network.

Differentiating based on applications rather than network layer protocol makes the most sense: After all, end users running a particular application (for instance, IP telephony) do not care about the network layer being used and expect the same level of network quality in all cases. This approach also reduces the management overhead for the QoS deployment and the use of network element resources.

It is recommended that no differentiation should be made between the two protocol types in implementing QoS at layer 2. Similar to the recommendation made for MPLS, the infrastructure should be kept unaware of the transported protocol type. Moreover, different PHBs for IPv4 and IPv6 would lead to an unmanageable number of policies.

The aim of IPv6 to reestablish a peer-to-peer model for IP transport will most definitely impact in a positive way the deployment of QoS. The boundary between private and public domains will no longer even out the characteristics of individual streams coming from different internal sources. In an IPv6 world, true end-to-end QoS policy implementations are closer to reality. Despite lacking a consolidated architecture at this time, the features available for IPv6 do provide the means to deploy QoS at least at the level of current IPv4 deployments.