This chapter covers the following subjects:

- Introduction to QoS
- Identifying and Comparing QoS Models
- QoS Implementation Methods

CHAPTER 2

IP Quality of Service

This chapter provides the essential background, definitions, and concepts for you to start learning IP quality of service (QoS). The following two chapters complement this one and provide more coverage of this topic. It is probably safe to expect about 20 percent of the ONT exam questions from this chapter.

"Do I Know This Already?" Quiz

The purpose of the "Do I Know This Already?" quiz is to help you decide whether you really need to read the entire chapter. The 20-question quiz, derived from the major sections of this chapter, helps you determine how to spend your limited study time.

Table 2-1 outlines the major topics discussed in this chapter and the "Do I Know This Already?" quiz questions that correspond to those topics. You can keep track of your score here, too.

Table 2-1 *"Do I Know This Already?" Foundation Topics Section-to-Question Mapping*

Foundation Topics Section Covering These Questions	Questions	Score
"Introduction to QoS"	1–7	
"Identifying and Comparing QoS Models"	8–13	
"QoS Implementation Methods"	14–20	
Total Score	**(20 possible)**	

CAUTION The goal of self-assessment is to gauge your mastery of the topics in this chapter. If you do not know the answer to a question or are only partially sure of the answer, mark this question wrong for purposes of the self-assessment. Giving yourself credit for an answer you correctly guess skews your self-assessment results and might provide you with a false sense of security.

You can find the answers to the "Do I Know This Already?" quiz in Appendix A, "Answers to the 'Do I Know This Already?' Quizzes and Q&A Sections." The suggested choices for your next step are as follows:

- **15 or less overall score**—Read the entire chapter. This includes the "Foundation Topics," "Foundation Summary," and "Q&A" sections.
- **16–17 overall score**—Begin with the "Foundation Summary" section and then follow up with the "Q&A" section at the end of the chapter.
- **18 or more overall score**—If you want more review on this topic, skip to the "Foundation Summary" section and then go to the "Q&A" section. Otherwise, proceed to the next chapter.

1. Which of the following items is *not* considered one of four major issues and challenges facing converged enterprise networks?
   a. Available bandwidth
   b. End-to-end delay
   c. Delay variation (jitter)
   d. Packet size
2. Which of the following is defined as the maximum bandwidth of a path?
   a. The bandwidth of the link within the path that has the largest bandwidth
   b. The bandwidth of the link within the path that has the smallest bandwidth
   c. The total of all link bandwidths within the path
   d. The average of all the link bandwidths within the path
3. Which of the following is *not* considered one of the main methods to tackle the bandwidth availability problem?
   a. Increase (upgrade) the link bandwidth.
   b. Classify and mark traffic and deploy proper queuing mechanisms.
   c. Forward large packets first.
   d. Use compression techniques.
4. Which of the following is *not* considered a major delay type?
   a. Queuing delay
   b. CEF (Cisco Express Forwarding) delay
   c. Serialization delay
   d. Propagation delay

5. Which of the following does *not* reduce delay for delay-sensitive application traffic?
   a. Increasing (upgrade) the link bandwidth
   b. Prioritizing delay-sensitive packets and forwarding important packets first
   c. Layer 2 payload encryption
   d. Header compression
6. Which of the following approaches does *not* tackle packet loss?
   a. Increase (upgrade) the link bandwidth.
   b. Increase the buffer space.
   c. Provide guaranteed bandwidth.
   d. Eliminate congestion avoidance.
7. Which of the following is *not* a major step in implementing QoS?
   a. Apply access lists to all interfaces that process sensitive traffic
   b. Identify traffic types and their requirements
   c. Classify traffic based on the requirements identified
   d. Define policies for each traffic class
8. Which of following is *not* one of the three main QoS models?
   a. MPLS QoS
   b. Differentiated services
   c. Best effort
   d. Integrated services
9. Which two of the following items are considered drawbacks of the best-effort model?
   a. Inability to scale
   b. Lack of service guarantee
   c. Lack of service differentiation
   d. Difficulty in implementing (complexity)
10. Which of the following is *not* a function that IntServ requires to be implemented on the routers along the traffic path?
    a. Admission control and policing
    b. Classification
    c. Queuing and scheduling
    d. Fast switching

11. Which of the following is the role of RSVP within the IntServ model?
    a. Routing
    b. Switching
    c. Signaling/Bandwidth Reservation
    d. Caching
12. Which of the following is *not* considered a benefit of the IntServ model?
    a. Explicit end-to-end resource admission control
    b. Continuous signaling per active flow
    c. Per-request policy admission control
    d. Signaling of dynamic port numbers
13. Which of the following is *not* true about the DiffServ model?
    a. Within the DiffServ model, QoS policies (are deployed to) enforce differentiated treatment of the defined traffic classes.
    b. Within the DiffServ model, classes of traffic and the policies are defined based on business requirements; you choose the service level for each traffic class.
    c. Pure DiffServ makes extensive use of signaling; therefore, it is called *hard QoS.*
    d. DiffServ is a scalable model.
14. Which of the following is *not* a QoS implementation method?
    a. Cisco IOS CLI
    b. MQC
    c. Cisco AVVID (VoIP and Enterprise)
    d. Cisco SDM QoS Wizard
15. Which of the following is *not* a major step in implementing QoS with MQC?
    a. Define traffic classes using the class map.
    b. Define QoS policies for the defined traffic classes using the policy map.
    c. Apply the defined policies to each intended interface using the **service-policy** command.
    d. Enable AutoQoS.

16. Which of the following is the simplest QoS implementation method with an option specifically for VoIP?

    a. AutoQoS (VoIP)

    b. CLI

    c. MQC

    d. Cisco SDM QoS Wizard

17. Select the most time-consuming and the least time-consuming QoS implementation methods.

    a. CLI

    b. MQC

    c. AutoQoS

    d. Cisco SDM QoS Wizard

18. What is the most significant advantage of MQC over CLI?

    a. It requires little time to implement.

    b. It requires little expertise to implement.

    c. It has a GUI and interactive wizard.

    d. It separates traffic classification from policy definition.

19. Before you enable AutoQoS on an interface, which two of the following must you ensure have been configured on that interface?

    a. Cisco modular QoS is configured.

    b. CEF is enabled.

    c. The SDM has been enabled.

    d. The correct bandwidth on the interface is configured.

20. Select the item that is *not* a main service obtained from SDM QoS.

    a. It enables you to implement QoS on the network.

    b. It enables you to fine-tune QoS on the network.

    c. It enables you to monitor QoS on the network.

    d. It enables you to troubleshoot QoS on the network.

Foundation Topics

Introduction to QoS

This section introduces the concept of QoS and discusses the four main issues in a converged network that have QoS implications, as well as the Cisco IP QoS mechanisms and best practices to deal with those issues. This section also introduces the three steps in implementing a QoS policy on a network.

Converged Network Issues Related to QoS

A converged network supports different types of applications, such as voice, video, and data, simultaneously over a common infrastructure. Accommodating these applications that have different sensitivities and requirements is a challenging task on the hands of network engineers.

The acceptable end-to-end delay for the Voice over IP (VoIP) packets is 150 to 200 milliseconds (ms). Also, the delay variation or jitter among the VoIP packets must be limited so that the buffers at the receiving end do not become exhausted, causing breakup in the audio flow. In contrast, a data application such as a file download from an FTP site does not have such a stringent delay requirement, and jitter does not impose a problem for this type of application either. When numerous active VoIP and data applications exist, mechanisms must be put in place so that while critical applications function properly, a reasonable number of voice applications can remain active and function with good quality (with low delay and jitter) as well.

Many data applications are TCP-based. If a TCP segment is dropped, the source retransmits it after a timeout period is passed and no acknowledgement for that segment is received. Therefore, TCP-based applications have some tolerance to packet drops. The tolerance of video and voice applications toward data loss is minimal. As a result, the network must have mechanisms in place so that at times of congestion, packets encapsulating video and voice receive priority treatment and are not dropped.

Network outages affect all applications and render them disabled. However, well-designed networks have redundancy built in, so that when a failure occurs, the network can reroute packets through alternate (redundant) paths until the failed components are repaired. The total time it takes to notice the failure, compute alternate paths, and start rerouting the packets must be short enough for the voice and video applications not to suffer and not to annoy the users. Again, data applications usually do not expect the network recovery to be as fast as video and voice applications expect it to be. Without redundancy and fast recovery, network outage is unacceptable, and mechanisms must be put in place to avoid it.

Based on the preceding information, you can conclude that four major issues and challenges face converged enterprise networks:

- **Available bandwidth**—Many simultaneous data, voice, and video applications compete over the limited bandwidth of the links within enterprise networks.
- **End-to-end delay**—Many actions and factors contribute to the total time it takes for data or voice packets to reach their destination. For example, compression, packetization, queuing, serialization, propagation, processing (switching), and decompression all contribute to the total delay in VoIP transmission.
- **Delay variation (jitter)**—Based on the amount of concurrent traffic and activity, plus the condition of the network, packets from the same flow might experience a different amount of delay as they travel through the network.
- **Packet loss**—If volume of traffic exhausts the capacity of an interface, link, or device, packets might be dropped. Sudden bursts or failures are usually responsible for this situation.

The sections that follow explore these challenges in detail.

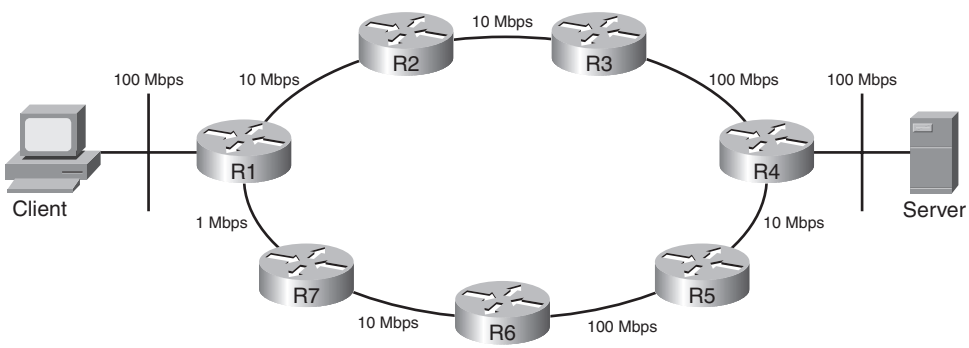
Available Bandwidth

Packets usually flow through the best path from source to destination. The maximum bandwidth of that path is equal to the bandwidth of the link with the smallest bandwidth. Figure 2-1 shows that R1-R2-R3-R4 is the best path between the client and the server. On this path, the maximum bandwidth is 10 Mbps because that is the bandwidth of the link with the smallest bandwidth on that path. The average available bandwidth is the maximum bandwidth divided by the number of flows.

Figure 2-1 *Maximum Bandwidth and Average Available Bandwidth Along the Best Path (R1-R2-R3-R4) Between the Client and Server*

$Bandwidth_{(Max)}$ = Min(10 Mbps, 10 Mbps, 100 Mbps) = 10 Mbps

$Bandwidth_{(Avail)}$ = $Bandwidth_{(Max)}$/Flows



Lack of sufficient bandwidth causes delay, packet loss, and poor performance for applications. The users of real-time applications (voice and video) detect this right away. You can tackle the bandwidth availability problem in numerous ways:

- **Increase (upgrade) link bandwidth**—This is effective, but it is costly.
- **Classify and mark traffic and deploy proper queuing mechanisms**—Forward important packets first.
- **Use compression techniques**—Layer 2 payload compression, TCP header compression, and cRTP are some examples.

Increasing link bandwidth is undoubtedly beneficial, but it cannot always be done quickly, and it has cost implications. Those who just increase bandwidth when necessary notice that their solution is not very effective at times of heavy traffic bursts. However, in certain scenarios, increasing link bandwidth might be the first action necessary (but not the last).

Classification and marking of the traffic, combined with congestion management, is an effective approach to providing adequate bandwidth for enterprise applications.

Link compression, TCP header compression, and RTP header compression are all different compression techniques that can reduce the bandwidth consumed on certain links, and therefore increase throughput. Cisco IOS supports the Stacker and Predictor Layer 2 compression algorithms that compress the payload of the packet. Usage of hardware compression is always preferred over software-based compression. Because compression is CPU intensive and imposes yet another delay, it is usually recommended only on slow links.

NOTE Most compression mechanisms must be configured on a link-by-link basis—in other words, on both ends of each link. Classification, marking, compression, and advanced queuing mechanisms are discussed in Chapters 3, 4, and 5 in detail.

End-to-End Delay

There are different types of delay from source to destination. End-to-end delay is the sum of those different delay types that affect the packets of a certain flow or application. Four of the important types of delay that make up end-to-end delay are as follows:

- Processing delay
- Queuing delay
- Serialization delay
- Propagation delay

Processing delay is the time it takes for a device such as a router or Layer 3 switch to perform all the tasks necessary to move a packet from the input (ingress) interface to the output (egress) interface. The CPU type, CPU utilization, switching mode, router architecture, and configured features on the device affect the processing delay. For example, packets that are distributed-CEF switched on a versatile interface processor (VIP) card cause no CPU interrupts.

Queuing delay is the amount of time that a packet spends in the output queue of a router interface. The busyness of the router, the number of packets waiting in the queue, the queuing discipline, and the interface bandwidth all affect the queuing delay.

Serialization delay is the time it takes to send all the bits of a frame to the physical medium for transmission across the physical layer. The time it takes for the bits of that frame to cross the physical link is called the *propagation delay.* Naturally, the propagation delay across different media can be significantly different. For instance, the propagation delay on a high-speed optical connection such as OC-192 is significantly lower than the propagation delay on a satellite-based link.

NOTE In best-effort networks, while serialization and propagation delays are fixed, the processing and queuing delays are variable and unpredictable.

Other types of delay exist, such as WAN delay, compression and decompression delay, and de-jitter delay.

Delay Variation

The variation in delays experienced by the packets of the same flow is called *delay variation* or *jitter.* Packets of the same flow might not arrive at the destination at the same rate that they were released. These packets, individually and independent from each other, are processed, queued, de-queued, and so on. Therefore, they might arrive out of sequence, and their end-to-end delays might vary. For voice and video packets, it is essential that at the destination point, the packets are released to the application in the correct order and at the same rate that they were released at the source. The de-jitter buffer serves that purpose. As long as the delay variation is not too much, at the destination point, the de-jitter buffer holds packets, sorts them, and releases them to the application based on the Real-Time Transport Protocol (RTP) time stamp on the packets. Because the buffer compensates the jitter introduced by the network, it is called the *de-jitter buffer.*

Average queue length, packet size, and link bandwidth contribute to serialization and propagation delay. You can reduce delay by doing some or all of the following:

- **Increase (upgrade) link bandwidth**—This is effective as the queue sizes drop and queuing delays soar. However, upgrading link capacity (bandwidth) takes time and has cost implications, rendering this approach unrealistic at times.

- **Prioritize delay-sensitive packets and forward important packets first**—This might require packet classification or marking, but it certainly requires deployment of a queuing mechanism such as weighted fair queuing (WFQ), class-based weighted fair queuing (CBWFQ), or low-latency queuing (LLQ). This approach is not as costly as the previous approach, which is a bandwidth upgrade.
- **Reprioritize packets**—In certain cases, the packet priority (marking) has to change as the packet enters or leaves a device. When packets leave one domain and enter another, this priority change might have to happen. For instance, the packets that leave an enterprise network with critical marking and enter a provider network might have to be reprioritized (remarked) to best effort if the enterprise is only paying for best effort service.
- **Layer 2 payload compression**—Layer 2 compression reduces the size of the IP packet (or any other packet type that is the frame's payload), and it frees up available bandwidth on that link. Because complexity and delay are associated with performing the compression, you must ensure that the delay reduced because of compression is more than the delay introduced by the compression complexity. Note that payload compression leaves the frame header in tact; this is required in cases such as frame relay connections.
- **Use header compression**—RTP header compression (cRTP) is effective for VoIP packets, because it greatly improves the overhead-to-payload ratio. cRTP is recommended on slow (less than 2 Mbps) links. Header compression is less CPU-intensive than Layer 2 payload compression.

Packet Loss

Packet loss occurs when a network device such as a router has no more buffer space on an interface (output queue) to hold the new incoming packets and it ends up dropping them. A router may drop some packets to make room for higher priority ones. Sometimes an interface reset causes packets to be flushed and dropped. Packets are dropped for other reasons, too, including interface overrun.

TCP resends the dropped packets; meanwhile, it reduces the size of the send window and slows down at times of congestion and high network traffic volume. If a packet belonging to a UDP-based file transfer (such as TFTP) is dropped, the whole file might have to be resent. This creates even more traffic on the network, and it might annoy the user. Application flows that do not use TCP, and therefore are more drop-sensitive, are called *fragile flows*.

During a VoIP call, packet loss results in audio breakup. A video conference will have jerky pictures and its audio will be out of synch with the video if packet drops or extended delays occur. When network traffic volume and congestion are heavy, applications experience packet drops, extended delays, and jitter. Only with proper QoS configuration can you avoid these problems or at least limit them to low-priority packets.

On a Cisco router, at times of congestion and packet drops, you can enter the **show interface** command and observe that on some or all interfaces, certain counters such as those in the following list have incremented more than usual (baseline):

- **Output drop**—This counter shows the number of packets dropped, because the output queue of the interface was full at the time of their arrival. This is also called *tail drop*.
- **Input queue drop**—If the CPU is overutilized and cannot process incoming packets, the input queue of an interface might become full, and the number of packets dropped in this scenario will be reported as input queue drops.
- **Ignore**—This is the number of frames ignored due to lack of buffer space.
- **Overrun**—The CPU must allocate buffer space so that incoming packets can be stored and processed in turn. If the CPU becomes too busy, it might not allocate buffer space quickly enough and end up dropping packets. The number of packets dropped for this reason is called *overruns*.
- **Frame error**—Frames with cyclic redundancy check (CRC) error, runt frames (smaller than minimum standard), and giant frames (larger than the maximum standard) are usually dropped, and their total is reported as frame errors.

You can use many methods, all components of QoS, to tackle packet loss. Some methods protect packet loss from all applications, whereas others protect specific classes of packets from packet loss only. The following are examples of approaches that packet loss can merit from:

- **Increase (upgrade) link bandwidth**—Higher bandwidth results in faster packet departures from interface queues. If full queue scenarios are prevented, so are tail drops and random drops (discussed later).
- **Increase buffer space**—Network engineers must examine the buffer settings on the interfaces of network devices such as routers to see if their sizes and settings are appropriate. When dealing with packet drop issues, it is worth considering an increase of interface buffer space (size). A larger buffer space allows better handling of traffic bursts.
- **Provide guaranteed bandwidth**—Certain tools and features such as CBWFQ and LLQ allow the network engineers to reserve certain amounts of bandwidth for a specific class of traffic. As long as enough bandwidth is reserved for a class of traffic, packets of such a class will not become victims of packet drop.
- **Perform congestion avoidance**—To prevent a queue from becoming full and starting tail drop, you can deploy random early detection (RED) or weighted random early detection (WRED) to drop packets from the queue before it becomes full. You might wonder what the merit of that deployment would be. When packets are dropped before a queue becomes full, the packets can be dropped from certain flows only; tail drop loses packets from all flows.

With WRED, the flows that lose packets first are the lowest priority ones. It is hoped that the highest priority packet flows will not have drops. Drops due to deployment of RED/WRED slow TCP-based flows, but they have no effect on UDP-based flows.

Most companies that connect remote sites over a WAN connection transfer both TCP- and UDP-based application data between those sites. Figure 2-2 displays a company that sends VoIP traffic as well as file transfer and other application data over a WAN connection between its remote branch and central main branch. Note that, at times, the collection of traffic flows from the remote branch intending to cross R2 and the WAN connection (to go to the main central branch) can reach high volumes.

Figure 2-2 *Solutions for Packet Loss and Extended Delay*

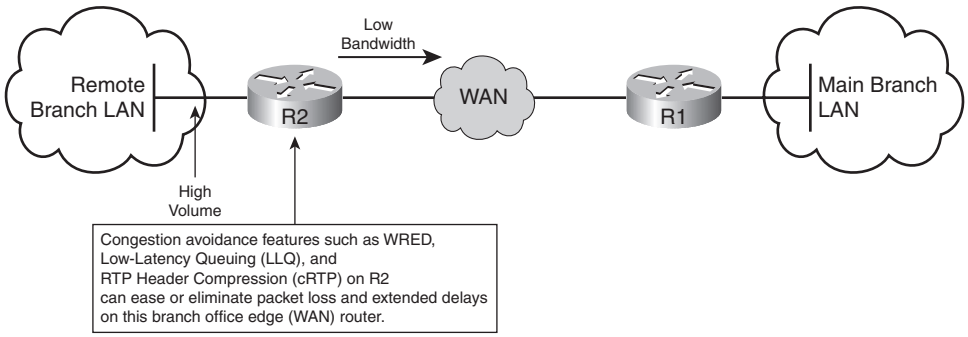


Figure 2-2 displays the stated scenario that leads to extended delay and packet loss. Congestion avoidance tools trigger TCP-based applications to throttle back before queues and buffers become full and tail drops start. Because congestion avoidance features such as WRED do not trigger UDP-based applications (such as VoIP) to slow down, for those types of applications, you must deploy other features, including compression techniques such as cRTP and advanced queuing such as LLQ.

Definition of QoS and the Three Steps to Implementing It

Following is the most recent definition that Cisco educational material provides for QoS:

QoS is the ability of the network to provide better or special service to a set of users or applications or both to the detriment of other users or applications or both.

The earliest versions of QoS tools protected data against data. For instance, priority queuing made sure packets that matched an access list always had the right of way on an egress interface. Another example is WFQ, which prevents small packets from waiting too long behind large packets on an egress interface outbound queue. When VoIP started to become a serious technology, QoS tools were created to protect voice from data. An example of such a tool is RTP priority queue.

RTP priority queue is reserved for RTP (encapsulating voice payload). RTP priority queuing ensures that voice packets receive right of way. If there are too many voice streams, data applications begin experiencing too much delay and too many drops. Strict priority queue (incorporated in LLQ) was invented to limit the bandwidth of the priority queue, which is essentially dedicated to voice packets. This technique protects data from voice; too many voice streams do not downgrade the quality of service for data applications. However, what if there are too many voice streams? All the voice calls and streams must share the bandwidth dedicated to the strict priority queue that is reserved for voice packets. If the number of voice calls exceeds the allocated resources, the quality of those calls will drop. The solution to this problem is call admission control (CAC). CAC prevents the number of concurrent voice calls from going beyond a specified limit and hurting the quality of the active calls. CAC protects voice from voice. Almost all the voice requirements apply to video applications, too; however, the video applications are more bandwidth hungry.

Enterprise networks must support a variety of applications with diverse bandwidth, drop, delay, and jitter expectations. Network engineers, by using proper devices, Cisco IOS features, and configurations, can control the behavior of the network and make it provide predictable service to those applications. The existence of voice, video, and multimedia applications in general not only adds to the bandwidth requirements in networks but also adds to the challenges involved in having to provide granular and strictly controlled delay, jitter, and loss guarantees.

Implementing QoS

Implementing QoS involves three major steps:

Step 1 Identifying traffic types and their requirements

Step 2 Classifying traffic based on the requirements identified

Step 3 Defining policies for each traffic class

Even though many common applications and protocols exist among enterprise networks, within each network, the volumes and percentages of those traffic types vary. Furthermore, each enterprise might have its own unique application types in addition to the common ones. Therefore, the first step in implementing QoS in an enterprise is to study and discover the traffic types and define the requirements of each identified traffic type. If two, three, or more traffic types have identical importance and requirements, it is unnecessary to define that many traffic classes. Traffic classification, which is the second step in implementing QoS, will define a few traffic classes, not hundreds. The applications that end up in different traffic classes have different requirements; therefore, the network must provide them with different service types. The definition of how each traffic class is serviced is called the *network policy.* Defining and deploying the network QoS policy for each class is Step 3 of implementing QoS. The three steps of implementing QoS on a network are explained next.

Step 1: Identifying Traffic Types and Their Requirements

Identifying traffic types and their requirements, the first step in implementing QoS, is composed of the following elements or substeps:

- **Perform a network audit**—It is often recommended that you perform the audit during the busy hour (BH) or congestion period, but it is also important that you run the audit at other times. Certain applications are run during slow business hours on purpose. There are scientific methods for identifying the busy network moments, for example, through statistical sampling and analysis, but the simplest method is to observe CPU and link utilizations and conduct the audit during the general peak periods.
- **Perform a business audit and determine the importance of each application**—The business model and goals dictate the business requirements. From that, you can derive the definition of traffic classes and the requirements for each class. This step considers whether delaying or dropping packets of each application is acceptable. You must determine the relative importance of different applications.
- **Define the appropriate service levels for each traffic class**—For each traffic class, within the framework of business objectives, a specific service level can define tangible resource availability or reservations. Guaranteed minimum bandwidth, maximum bandwidth, guaranteed end-to-end maximum delay, guaranteed end-to-end maximum jitter, and comparative drop preference are among the characteristics that you can define for each service level. The final service level definitions must meet business objectives and satisfy the comfort expectations of the users.

Step 2: Classifying Traffic Based on the Requirements Identified

The definition of traffic classes does not need to be general; it must include the traffic (application) types that were observed during the network audit step. You can classify tens or even hundreds of traffic variations into very few classes. The defined traffic classes must be in line with business objectives. The traffic or application types within the same class must have common requirements and business requirements. The exceptions to this rule are the applications that have not been identified or scavenger-class traffic.

Voice traffic has specific requirements, and it is almost always in its own class. With Cisco LLQ, VoIP is assigned to a single class, and that class uses a strict priority queue (a priority queue with strict maximum bandwidth) on the egress interface of each router. Many case studies have shown the merits of using some or all of the following traffic classes within an enterprise network:

- **Voice (VoIP) class**—Voice traffic has specific bandwidth requirements, and its delay and drops must be eliminated or at least minimized. Therefore, this class is the highest priority class but has limited bandwidth. VoIP packet loss should remain below 1% and the goal for its end-to-end delay must be 150 ms.

- **Mission-critical traffic class**—Critical business applications are put in one or two classes. You must identify the bandwidth requirements for them.
- **Signaling traffic class**—Signaling traffic, voice call setup and teardown for example, is often put in a separate class. This class has limited bandwidth expectations.
- **Transactional applications traffic class**—These applications, if present, include interactive, database, and similar services that need special attention. You must also identify the bandwidth requirements for them. Enterprise Resource Planning (ERP) applications such as Peoplesoft and SAP are examples of these types of applications.
- **Best-effort traffic class**—All the undefined traffic types are considered best effort and receive the remainder of bandwidth on an interface.
- **Scavenger traffic class**—This class of applications will be assigned into one class and be given limited bandwidth. This class is considered inferior to the best-effort traffic class. Peer-to-peer file sharing applications are put in this class.

Step 3: Defining Policies for Each Traffic Class

After the traffic classes have been formed based on the network audit and business objectives, the final step of implementing QoS in an enterprise is to provide a network-wide definition for the QoS service level that must be assigned to each traffic class. This is called *defining a QoS policy*, and it might include having to complete the following tasks:

- Setting a maximum bandwidth limit for a class
- Setting a minimum bandwidth guarantee for a class
- Assigning a relative priority level to a class
- Applying congestion management, congestion avoidance, and many other advanced QoS technologies to a class.

To provide an example, based on the traffic classes listed in the previous section, Table 2-2 defines a practical QoS policy.

Table 2-2 *Defining QoS Policy for Set Traffic Classes*

Class	Priority	Queue Type	Min/Max Bandwidth	Special QoS Technology
Voice	5	Priority	1 Mbps Min 1 Mbps Max	Priority queue
Business mission critical	4	CBWFQ	1 Mbps Min	CBWFQ

continues

Table 2-2 *Defining QoS Policy for Set Traffic Classes (Continued)*

Class	Priority	Queue Type	Min/Max Bandwidth	Special QoS Technology
Signaling	3	CBWFQ	400 Kbps Min	CBWFQ
Transactional	2	CBWFQ	1 Mbps Min	CBWFQ
Best-effort	1	CBWFQ	500 Kbps Max	CBWFQ CB-Policing
Scavenger	0	CBWFQ	Max 100 Kbps	CBWFQ +CB-Policing WRED

Identifying and Comparing QoS Models

This section discusses the three main QoS models, namely best-effort, Integrated Services, and Differentiated Services. The key features, and the benefits and drawbacks of each of these QoS models, are explained in turn.

Best-Effort Model

The best-effort model means that no QoS policy is implemented. It is natural to wonder why this model was not called no-effort. Within this model, packets belonging to voice calls, e-mails, file transfers, and so on are treated as equally important; indeed, these packets are not even differentiated. The basic mail delivery by the post office is often used as an example for the best-effort model, because the post office treats all letters as equally important.

The best-effort model has some benefits as well as some drawbacks. Following are the main benefits of this model:

- **Scalability**—The Internet is a best-effort network. The best-effort model has no scalability limit. The bandwidth of router interfaces dictates throughput efficiencies.
- **Ease**—The best-effort model requires no special QoS configuration, making it the easiest and quickest model to implement.

The drawbacks of the best-effort model are as follows:

- **Lack of service guarantee**—The best-effort model makes no guarantees about packet delivery/loss, delay, or available bandwidth.
- **Lack of service differentiation**—The best-effort model does not differentiate packets that belong to applications that have different levels of importance from the business perspective.

Integrated Services Model

The Integrated Services (IntServ) model, developed in the mid-1990s, was the first serious attempt to provide end-to-end QoS, which was demanded by real-time applications. IntServ is based on explicit signaling and managing/reserving network resources for the applications that need it and demand it. IntServ is often referred to as Hard-QoS, because Hard-QoS guarantees characteristics such as bandwidth, delay, and packet loss, thereby providing a predictable service level. Resource Reservation Protocol (RSVP) is the signaling protocol that IntServ uses. An application that has a specific bandwidth requirement must wait for RSVP to run along the path from source to destination, hop by hop, and request bandwidth reservation for the application flow. If the RSVP attempt to reserve bandwidth along the path succeeds, the application can begin operating. While the application is active, along its path, the routers provide the bandwidth that they have reserved for the application. If RSVP fails to successfully reserve bandwidth hop by hop all the way from source to destination, the application cannot begin operating.

IntServ mimics the PSTN model, where every call entails end-to-end signaling and securing resources along the path from source to destination. Because each application can make a unique request, IntServ is a model that can provide multiple service levels. Within the Cisco QoS framework, RSVP can act both as a signaling mechanism and as a CAC mechanism. If an RSVP attempt to secure and reserve resources for a voice call fails, the call does not get through. Controlled volume services within the Cisco IOS QoS feature set are provided by RSVP and advanced queuing mechanisms such as LLQ. The Guaranteed Rate service type is offered by deploying RSVP and LLQ. Controlled Load service is provided by RSVP and WRED.

For a successful implementation of IntServ, in addition to support for RSVP, enable the following features and functions on the routers or switches within the network:

Admission control—Admission control responds to application requests for end-to-end resources. If the resources cannot be provided without affecting the existing applications, the request is turned down.

Classification—The traffic belonging to an application that has made resource reservations must be classified and recognized by the transit routers so that they can furnish appropriate service to those packets.

Policing—It is important to measure and monitor that applications do not exceed resource utilization beyond their set profiles. Rate and burst parameters are used to measure the behavior of an application. Depending on whether an application conforms to or exceeds its agreed-upon resource utilizations, appropriate action is taken.

Queuing—It is important for network devices to be able to hold packets while processing and forwarding others. Different queuing mechanisms store and forward packets in unique ways.

Scheduling—Scheduling works in conjunction with queuing. If there are multiple queues on an interface, the amount of data that is dequeued and forwarded from each queue at each cycle, hence the relative attention that each queue gets, is called the *scheduling algorithm*. Scheduling is enforced based on the queuing mechanism configured on the router interface.

When IntServ is deployed, new application flows are admitted until requested resources can no longer be furnished. Any new application will fail to start because the RSVP request for resources will be rejected. In this model, RSVP makes the QoS request for each flow. This request includes identification for the requestor, also called the *authorized user* or *authorization object*, and the needed traffic policy, also called the *policy object*. To allow all intermediate routers between source and destination to identify each flow, RSVP provides the flow parameters such as IP addresses and port numbers. The benefits of the IntServ model can be summarized as follows:

- Explicit end-to-end resource admission control
- Per-request policy admission control
- Signaling of dynamic port numbers

Some drawbacks to using IntServ exist, the most important of which are these:

- Each active flow has a continuous signaling. This overhead can become substantially large as the number of flows grows. This is because of the stateful architecture of RSVP.
- Because each flow is tracked and maintained, IntServ as a flow-based model is not considered scalable for large implementations such as the Internet.

Differentiated Services Model

Differentiated Services (DiffServ) is the newest of the three QoS models, and its development has aimed to overcome the limitations of its predecessors. DiffServ is not a guaranteed QoS model, but it is a highly scalable one. The Internet Engineering Task Force (IETF) description and discussion on DiffServ are included in RFCs 2474 and 2475. Whereas IntServ has been called the "Hard QoS" model, DiffServ has been called the "Soft QoS" model. IntServ, through usage of signaling and admission control, is able to either deny application of requested resources or admit it and guarantee the requested resources.

Pure DiffServ does not use signaling; it is based on per-hop behavior (PHB). PHB means that each hop in a network must be preprogrammed to provide a specific level of service for each class of traffic. PHB then does not require signaling as long as the traffic is marked to be identified as one of the expected traffic classes. This model is more scalable because signaling and status monitoring (overhead) for each flow are not necessary. Each node (hop) is prepared to deal with a limited variety of traffic classes. This means that even if thousands of flows become active, they

are still categorized as one of the predefined classes, and each flow will receive the service level that is appropriate for its class. The number of classes and the service level that each traffic class should receive are decided based on business requirements.

Within the DiffServ model, traffic is first classified and marked. As the marked traffic flows through the network nodes, the type of service it receives depends on its marking. DiffServ can protect the network from oversubscription by using policing and admission control techniques as well. For example, in a typical DiffServ network, voice traffic is assigned to a priority queue that has reserved bandwidth (through LLQ) on each node. To prohibit too many voice calls from becoming active concurrently, you can deploy CAC. Note that all the voice packets that belong to the admitted calls are treated as one class.

The DiffServ model is covered in detail in Chapters 3, 4, and 5. Remember the following three points about the DiffServ model:

- Network traffic is classified.
- QoS policies enforce differentiated treatment of the defined traffic classes.
- Classes of traffic and the policies are defined based on business requirements; you choose the service level for each traffic class.

The main benefit of the DiffServ model is its scalability. The second benefit of the DiffServ model is that it provides a flexible framework for you to define as many service levels as your business requirements demand. The main drawback of the DiffServ model is that it does not provide an absolute guarantee of service. That is why it is associated with the term Soft QoS. The other drawback of this model is that several complex mechanisms must be set up consistently on all the elements of the network for the model to yield the desired results.

Following are the benefits of DiffServ:

- Scalability
- Ability to support many different service levels

The drawbacks of DiffServ are as follows:

- It cannot provide an absolute service guarantee.
- It requires implementation of complex mechanisms through the network.

QoS Implementation Methods

This section explores the four main QoS implementation methods, namely CLI, MQC, Cisco AutoQoS, and SDM QoS Wizard. A high-level explanation of each QoS implementation method and the advantages and disadvantages of each are provided in turn.

Legacy Command-Line Interface (CLI)

Legacy CLI was the method used up to about six years ago to implement QoS on network devices. Legacy CLI requires configuration of few to many lines of code that for the most part would have to be applied directly at the interface level. Configuration of many interfaces required a lot of typing or cutting and pasting. Maintaining consistency, minimizing errors, and keeping the configuration neat and understandable were difficult to do using legacy CLI.

Legacy CLI configuration required the user to log into the router via console using a terminal (or a terminal emulator) or via a virtual terminal line using a Telnet application. Because it was a nonmodular method, legacy CLI did not allow users to completely separate traffic classification from policy definition and how the policy is applied. Legacy CLI was also more error prone and time consuming. Today, people still use CLI, but mostly to fine-tune the code generated by AutoQoS, which will be discussed later.

You began legacy CLI configuration by identifying, classifying, and prioritizing the traffic. Next, you had to select one of the available and appropriate QoS tools such as link compression or an available queuing mechanism such as custom or priority queuing. Finally, you had to enter from a few to several lines of code applying the selected QoS mechanisms for one or many interfaces.

Modular QoS Command-Line Interface (MQC)

Cisco introduced MQC to address the shortcomings of the legacy CLI and to allow utilization of the newer QoS tools and features available in the modern Cisco IOS. With the MQC, traffic classification and policy definition are done separately. Traffic policies are defined after traffic classes. Different policies might reference the same traffic classes, thereby taking advantage of the modular and reusable code. When one or more policies are defined, you can apply them to many interfaces, promoting code consistency and reuse.

MQC is modular, more efficient, and less time consuming than legacy CLI. Most importantly, MQC separates traffic classification from policy definition, and it is uniform across major Cisco IOS platforms. With MQC, defined policies are applied to interfaces rather than a series of raw CLI commands being applied to interfaces.

Implementing QoS with MQC involves three major steps:

Step 1 Define traffic classes using the **class-map** command. This step divides the identified network traffic into a number of named classes.

Step 2 Define QoS policies for the defined traffic classes using the **policy-map** command. This step involves QoS features being linked to traffic classes. It defines the treatment of the defined classes of traffic.

Step 3 Apply the defined policies in the inbound or outbound direction to each intended interface, subinterface, or circuit, using the **service-policy** command. This step defines where the defined policies are applied.

Each class map, which has a case-sensitive name, is composed of one or more **match** statements. One or all of the **match** statements must be matched, depending on whether class map contains the **match-any** or the **match-all** command. When neither **match-any** nor **match-all** is specified on the **class-map** statement, **match-all** applies by default.

Example 2-1 shows two class maps. The first class map is called VOIP. This class map specifies that traffic matching access list 100 is classified as VOIP. The second class map is called Business-Application. It specifies that traffic matching **access-list 101** is classified as Business-Application.

Example 2-1 *Class Maps*

```
class-map VOIP
 match access-group 100
!
class-map Business-Application
 match access-group 101
!
```

In Example 2-1, note that both of the class maps have only one **match** statement, and neither **match-all** nor **match-any** is specified, which defaults to **match-all**. When only one **match** statement exists, **match-all** and **match-any** yield the same result. However, when more than one **match** statement exists, using **match-any** or **match-all** makes a big difference. **match-any** means only one of the match statements needs to be met, and **match-all** means all the **match** statements must be met to bind the packet to the class.

NOTE The opposite of the **match** condition is the **match not** condition.

You create traffic policies by associating required QoS features to traffic classes defined by class maps; you use the **policy-map** command to do that. A policy map has a case-sensitive name and can associate QoS policies for up to 256 traffic classes (each defined by a class map). Example 2-2 exhibits a policy map called Enterprise-Policy. This policy map specifies that traffic classified as

VOIP is assigned to a priority queue that has a bandwidth guarantee of 256 Kbps. Enterprise-Policy also states that the traffic classified as Business-Application is assigned to a WFQ with a bandwidth guarantee of 256 Kbps. According to this policy map, all other traffic, classified as **class-default**, will be assigned to a queue that gets the rest of the available bandwidth, and a WFQ policy will be applied to it.

Example 2-2 *Policy Map*

```
policy-map Enterprise-Policy
 class VOIP
  priority 256
 class Business-Application
  bandwidth 256
 class class-default
  fair-queue
!
```

If you configure a policy map that includes a **class** statement followed by the name of a nonexistent class map, as long as the statement includes a condition, a class map is created and inserted into the configuration with that name automatically. If, within a policy map, you do not refer to the **class-default** (and do not configure it), any traffic that the defined classes do not match will still be treated as **class-default**. The **class-default** gets no QoS guarantees and can use a FIFO or a WFQ.

A policy map is applied on an interface (or subinterface, virtual template, or circuit) in the outbound or inbound direction using the **service-policy** command (and the direction specified using the **input** or **output** keywords). You can apply a defined and configured policy map to more than one interface. Reusing class maps and policy maps is highly encouraged because it promotes standardization and reduces the chance of errors. Example 2-3 shows that the policy map Enterprise-Policy is applied to the serial 1/0 interface of a router on the outbound direction.

Example 2-3 *Service-Policy*

```
interface serial 1/0
 service-policy output Enterprise-Policy
!
```

The following commands allow you to display and verify QoS classes and policies you have configured using the MQC:

show class-map—This command displays all the configured class maps.

show policy-map—This command displays all the configured policy maps.

show policy-map interface *interface*—This command displays the policy map that is applied to a particular interface using the **service-policy** command. This command also displays QoS interface statistics.

AutoQoS

AutoQoS is a value-added feature of Cisco IOS. After it is enabled on a device, AutoQoS automatically generates QoS configuration commands for the device. The initial release of AutoQoS (Auto QoS VoIP) focused on generating commands that made the device ready for VoIP and IP Telephony. Later, the AutoQoS Discovery feature was introduced. The next generation of AutoQoS that takes advantage of AutoQoS discovery is called AutoQoS for the Enterprise. AutoQoS Discovery, as its name implies, analyzes live network traffic for as long as you let it run and generates traffic classes based on the traffic it has processed. Next, you enable the AutoQoS feature. AutoQoS uses the traffic classes (class maps) formed by AutoQoS Discovery to generate network QoS policy (policy map), and it applies the policy. Based on the interface type, AutoQoS might also add features such as fragmentation and interleaving, multilink, and traffic shaping to the interface configuration.

The main advantage of AutoQoS is that it simplifies the task of QoS configuration. Network administrators who lack in-depth knowledge of QoS commands and features can use AutoQoS to implement those features consistently and accurately. AutoQoS participates in all the main aspects of QoS deployment:

- **Classification**—AutoQoS for the Enterprise, through AutoQoS Discovery, automatically discovers applications and protocols (using Network Based Application Recognition, or NBAR). It uses Cisco Discovery Protocol (CDP) to check whether an IP phone is attached to a switch port.
- **Policy generation**—It provides appropriate treatment of traffic by the QoS policies that it auto-generates. AutoQoS checks interface encapsulations, and accordingly, it considers usage of features such as fragmentation, compression, and traffic shaping. Access lists, class maps, and policy maps, which normally have to be entered manually, are automatically generated by AutoQoS.
- **Configuration**—It is enabled by entering only one command, **auto qos**, at the interface. In a matter of seconds, proper commands to classify, mark, prioritize, preempt packets, and so on are added to the configuration appropriately.
- **Monitoring and reporting**—It generates system logging messages, SNMP traps, and summary reports.
- **Consistency**—The commands generated on different routers, using AutoQoS, are consistent and interoperable.

AutoQoS was introduced in Cisco IOS Software Release 12.2(15)T and provides a quick and consistent way to enter the bulk of QoS commands. Network administrators can then modify those commands and policies or optimize them using CLI. Cisco SDM QoS Wizard is a newer GUI tool that generates QoS commands and policies; that tool will be discussed in the next section.

AutoQoS performs a series of functions on WAN devices and interfaces. It creates a traffic class for voice payload (RTP), and it builds another class for voice signaling (Skinny, H.323, SIP, and MGCP). Service policies for voice bearer and voice signaling are created and deployed using LLQ with bandwidth guarantees. Voice traffic is assigned to the priority queue. On Frame Relay connections, AutoQoS turns on Frame Relay traffic shaping (FRTS) and link fragmentation and interleaving (LFI); on other types of links, such as PPP links, AutoQoS might turn on multilink PPP (MLP) and compressed RTP (cRTP). AutoQoS also provides SNMP and syslog alerts for VoIP packet drops.

In LAN environments, AutoQoS trust boundaries are set and enforced on the different types of switch ports, such as access ports and uplinks. Expedited queuing (strict priority) and weighted round-robin (WRR) are also enforced where required. Traffic is assigned to the proper queue based on its marking or application recognition based on NBAR.

Using AutoQoS has some prerequisites. Before you enable AutoQoS on an interface, you must ensure that the following tasks have been completed:

- Cisco Express Forwarding (CEF) is enabled. CEF is the prerequisite for NBAR.
- NBAR is enabled. AutoQoS for the Enterprise (not Auto QoS VoIP) uses NBAR for traffic discovery and classification.
- The correct bandwidth on the interface is configured. AutoQoS configures LLQ, cRTP, and LFI based on the interface type and the interface bandwidth. On certain interfaces, such as Ethernet, the bandwidth is auto-sensed; however, on other interfaces, such as synchronous serial interface, if the bandwidth is not specified, the IOS assumes a bandwidth of 1544 Kbps.

After these tasks have been completed, AutoQoS can be configured (enabled) on the desired interface. Example 2-4 shows a serial interface that has been configured with bandwidth, IP address, CEF, and AutoQoS.

Example 2-4 *Configuring AutoQoS on an Interface*

```
ip cef
interface serial 1/0
bandwidth 256
ip address 10.1.1.1 255.255.255.252
auto qos voip
!
```

Note that in Example 2-4, the command **auto qos voip** is applied to interface serial 1/0. This command represents the first generation of AutoQoS. The focus of **auto qos voip** was to automate generation of QoS commands to get the device ready for VoIP traffic. In the second generation *AutoQoS for the Enterprise*, you must first enter the **auto discovery qos** so that the router discovers and analyzes network traffic entering the interface using NBAR. Next, you enter the **auto qos** command. When you enter the **auto qos** command on an interface, the router builds class maps (based on the results of discovery) and then creates and applies a policy map on the interface. AutoQoS will be discussed in detail in Chapter 7, "Implementing AutoQoS."

Router and Security Device Manager (SDM) QoS Wizard

Cisco SDM is a web-based device-management tool for Cisco routers. With SDM, router deployment and troubleshooting of network and VPN connectivity issues becomes simpler. Proactive management through performance monitoring is also accomplished using SDM.

Cisco SDM supports a range of Cisco IOS Software releases and is available on many Cisco router models (from Cisco 830 Series to Cisco 7301); on several router models, SDM is preinstalled. Cisco SDM offers smart wizards that provide step-by-step assistance for configuration of LAN and WAN interfaces, Network Address Translation (NAT), firewall policy, IPS, IPsec VPN, and QoS. Inexperienced users find the SDM GUI easier to use than the CLI and enjoy the comprehensive online help and tutorials for SDM.

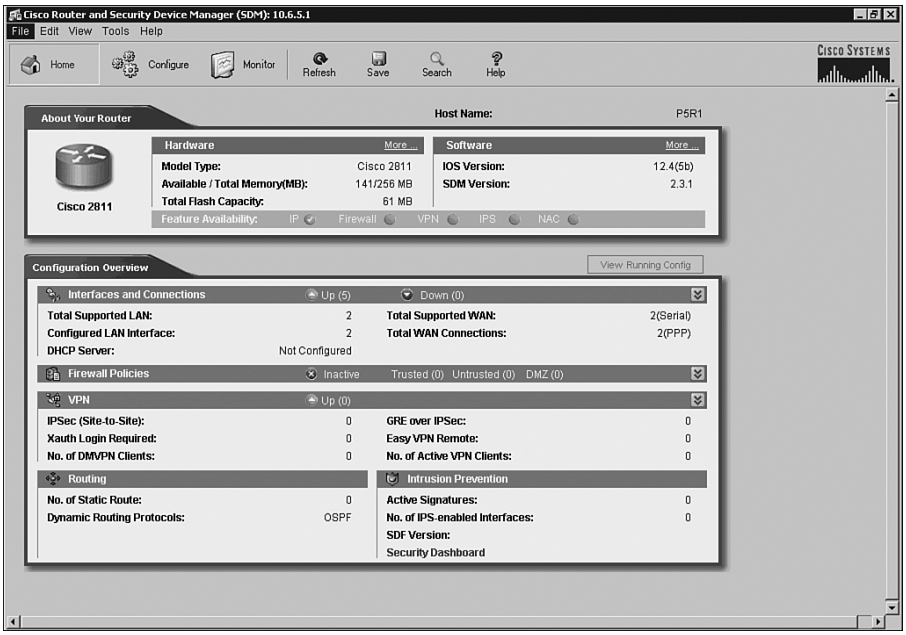
The QoS Wizard of SDM provides you with an easy-to-use user interface to define traffic classes and configure QoS policies for your network. The SDM predefines three different application categories: real-time, business-critical, and best-effort. SDM supports and uses NBAR to validate the bandwidth consumed by different application categories. Additional features offered by the SDM QoS Wizard include QoS policing and traffic monitoring. The SDM QoS Wizard enables you to do three things:

- Implement QoS
- Monitor QoS
- Troubleshoot QoS on your network

Figure 2-3 displays the main page of Cisco SDM. This page is comprised of two sections:

- About Your Router
- Configuration Overview

Figure 2-3 *Main Page of Cisco SDM*



In the About Your Router section of the SDM main page you can find information about your router's hardware, software, and the available features. For example, you can see the router's total and available memory, flash capacity, IOS version, SDM version, and whether features such as IP, firewall, VPN, IPS, and NAC are available. Further information can be seen through the **More...** options in the hardware and software sections. The Configuration Overview section of the SDM main page provides information about your router's LAN and WAN interfaces, firewall policies, VPN, routing, and IPS configurations. You can also see the router's running configuration through the **View Running Config** option. You can navigate to the main page by pressing the **Home** button on the main tool bar of the Cisco SDM. The other two important buttons on the Cisco SDM main tool bar are the **Configure** and **Monitor** buttons. The tasks available on the left side of the Configure page are:

- **Interfaces and Connections**
- **Firewall and ACL**
- **VPN**
- **Security Audit**
- **Routing**

- **NAT**
- **Intrusion Prevention**
- **Quality of Service**
- **NAC**
- **Additional Tasks**

The tasks available on the left side of the Monitor page are:

- **Overview**
- **Interface Status**
- **Firewall Status**
- **VPN Status**
- **Traffic Status**
- **NAC Status**
- **Logging**
- **IPS Status**

If you select the **Traffic Status** task, you will have the option to view graphs about QoS or application/protocol traffic.

The remainder of this section takes you through the steps necessary to create a QoS policy, apply it to an interface, and monitor the QoS status using the Cisco SDM (GUI) Wizard. For each step one or more figures are provided so that you are well prepared for the exam questions that might be asked about creating QoS policy using the SDM Wizard.

To begin to create a QoS policy you must complete the following steps:

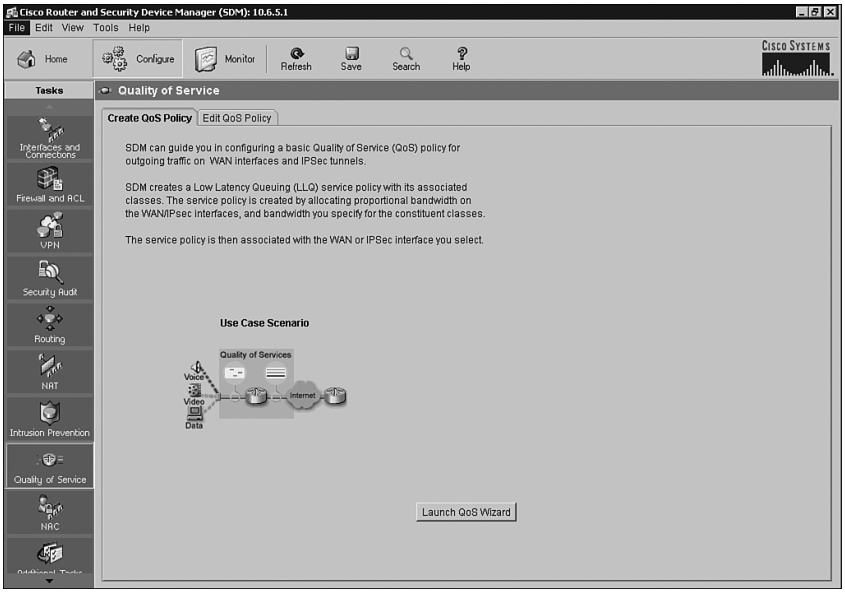
Step 1 Click the **Configure** button on the main toolbar of SDM.

Step 2 Click the **Quality of Service** button on the tasks toolbar on the left side of the SDM window (in Configuration mode; see Figure 2-4).

Step 3 Click the **Create QoS Policy** tab in the middle section of the SDM window (see Figure 2-4).

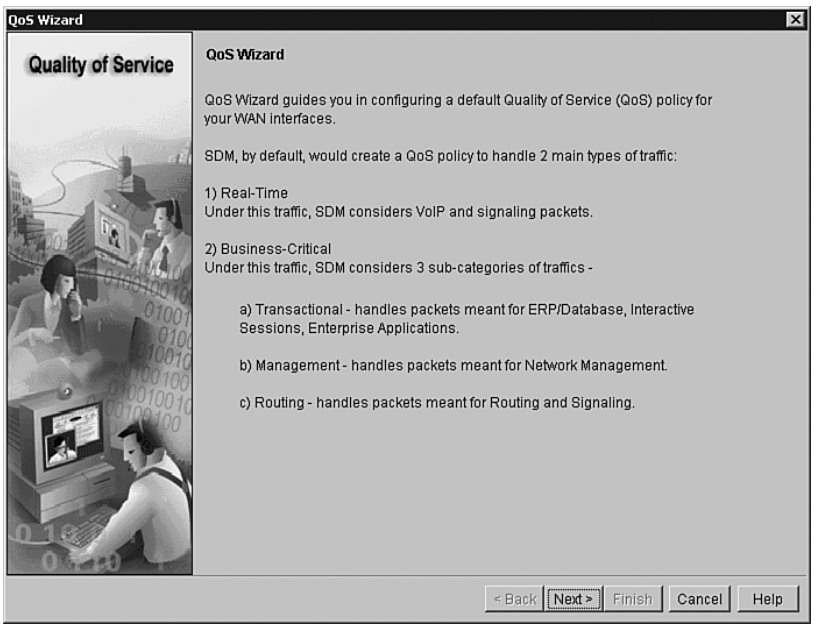
Step 4 Click the **Launch QoS Wizard** button on the bottom right side of the SDM window (see Figure 2-4).

Figure 2-4 *Four Steps to Start Creating a QoS Policy with SDM*



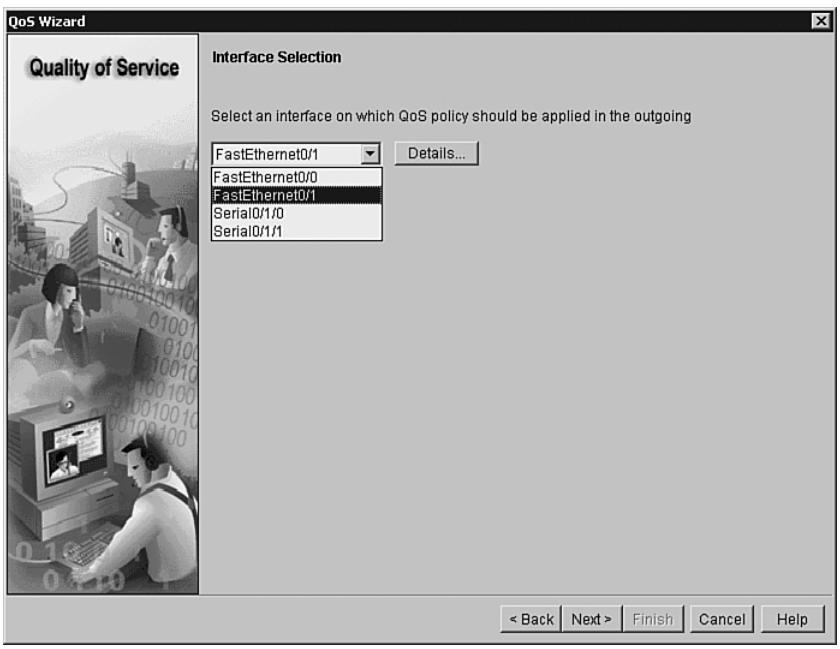
Now the SDM QoS Wizard page pops up on your computer screen (see Figure 2-5) and it informs you that SDM by default creates QoS policy to handle two main types of traffic, namely Real-Time and Business-Critical. To proceed press the **Next** button.

Figure 2-5 *SDM QoS Wizard Initial Page*



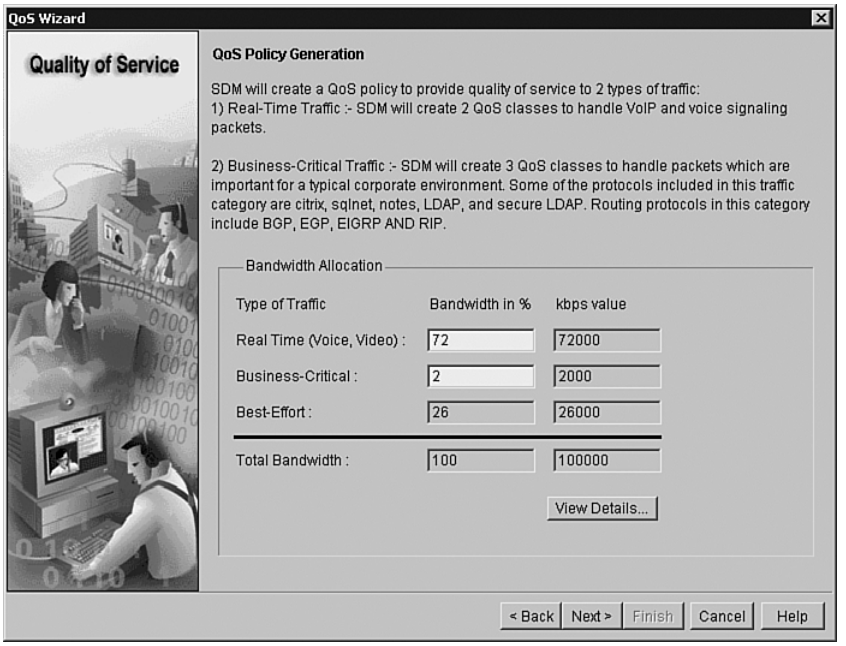
The QoS Wizard asks you to select an interface on which you want the QoS policy to be applied. Figure 2-6 shows you this screen. After making your selection press the **Next** button on that screen to proceed.

Figure 2-6 *Interface Selection Page of SDM QoS Wizard*



The SDM QoS Wizard asks you to enter the bandwidth percent for Real Time and Business-Critical traffic (see Figure 2-7). SDM will then automatically compute the bandwidth percent for the Best-Effort traffic and the actual bandwidth (kbps) for all three traffic classes.

Figure 2-7 *QoS Policy Generation Page of SDM QoS Wizard*



After you press **Next** the new page shows a summary of the configuration applied to the interface you have previously selected for the policy (see Figure 2-8). On this page you can scroll down and up to see the policy generated (and to be applied) in its entirety. Once you press the **Finish** button.

After you press the **Finish** button on the SDM QoS summary of the configuration screen, a Commands Delivery Status window appears (see Figure 2-9). This screen first informs you that commands are being prepared, then it tells you that the commands are being submitted, and finally it tells you that the commands have been delivered to the router. At this time, you can press the **OK** button and the job is complete.

Figure 2-8 *QoS Policy: Summary of the Configuration*

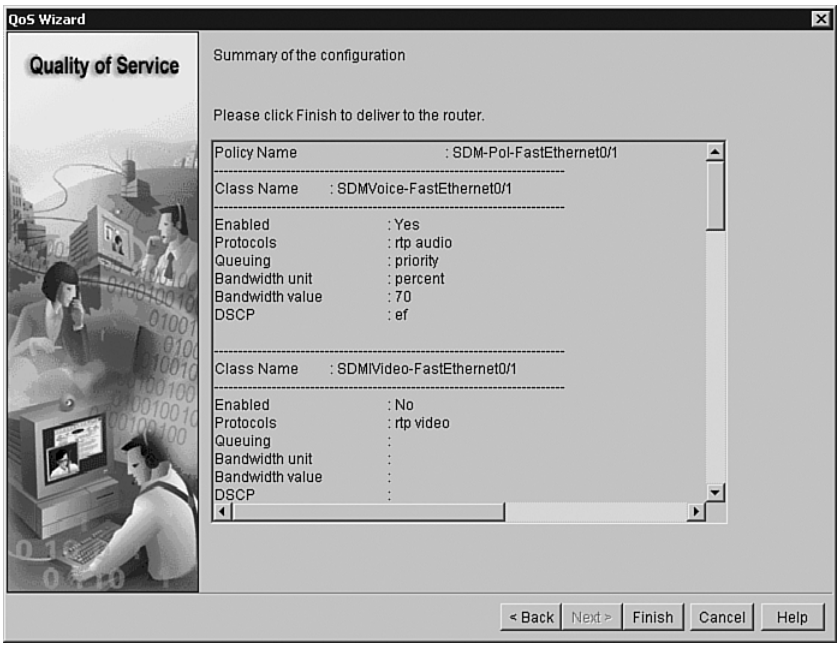
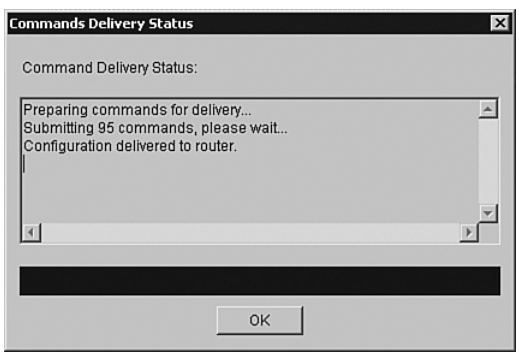


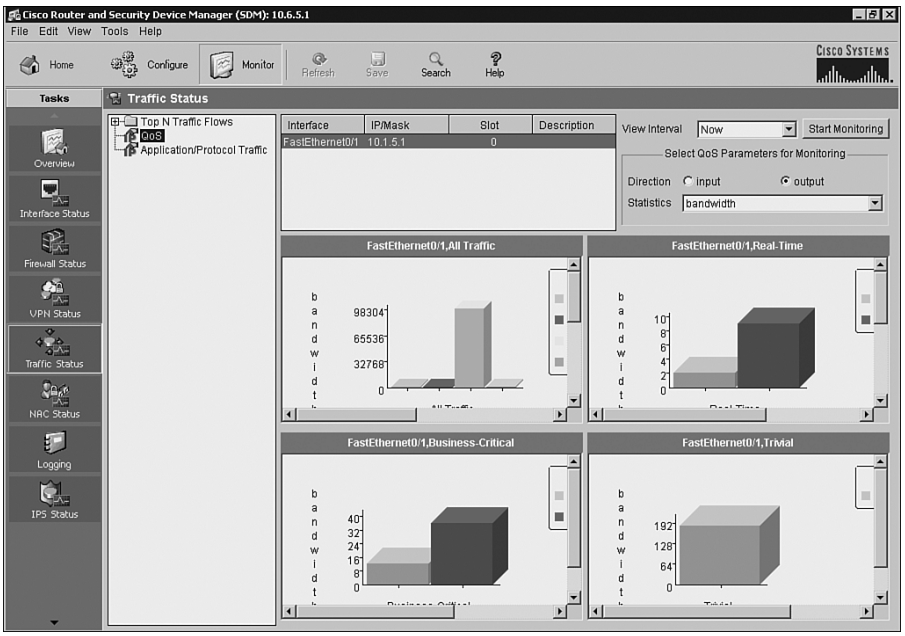
Figure 2-9 *QoS Policy: Commands Delivery Status*



Upon completion of your QoS configuration tasks, SDM allows you to monitor the QoS status. You must first click the **Monitor** button of the SDM main tool bar. Next, from the list of available tasks you must select **Traffic Status** (see Figure 2-10). Note that in the ONT courseware, this option is shown as **QoS Status**, probably due to SDM version differences. In the middle of the Traffic Status screen, you will then notice a folder called **Top N Traffic Flows** with **QoS** and **Application/Protocol Traffic** as two options displayed below it. If you click **QoS** (effectively

requesting to see the QoS status), you can then choose any of the interfaces displayed in the Traffic Status screen and see informative QoS-related graphs about the chosen interface.

Figure 2-10 *SDM Monitor Traffic/QoS Status*



When you select the **QoS** option of the Traffic Status, notice that on the top right corner of the screen you can select the **View Interval** (**Now**, **Every 1 Minute**, **Every 5 Minutes**, **Every 1 Hour**). Furthermore, there is a small area with the "Select QoS Parameters for Monitoring" title that allows you to select the **Direction** (**input** or **output**) of the traffic, and the **Statistics** (**bandwidth**, **byte**, and **packets dropped**) for which you want to see graphs.

Foundation Summary

The "Foundation Summary" is a collection of information that provides a convenient review of many key concepts in this chapter. If you are already comfortable with the topics in this chapter, this summary can help you recall a few details. If you just read this chapter, this review should help solidify some key facts. If you are doing your final preparation before the exam, the information in this section is a convenient way to review the day before the exam.

In a converged enterprise network, four major issues affect the performance and perceived quality of applications:

- Available bandwidth
- End-to-end delay
- Variation of delay (jitter)
- Packet loss

Lack of sufficient bandwidth, high end-to-end delay, high variation in delay, and excessive packet loss lower the quality of applications.

QoS is the ability of the network to provide better or "special" service to a set of users or applications or both to the detriment of other users or applications or both. You can use several QoS features, tools, and technologies to accomplish the QoS goals. Classification, marking, congestion avoidance, congestion management, compression, shaping, and policing are examples of QoS tools available in Cisco IOS. The three steps of implementing QoS in an enterprise network are as follows:

Step 1 Identify the network traffic and its requirements

Step 2 Define traffic classes

Step 3 Define a QoS policy for each traffic class

The main QoS models of today are as follows:

- **Best-effort**—The best-effort model requires no QoS configuration and mechanisms; therefore, it is easy and scalable, but it provides no Differentiated Service to different application types.
- **IntServ**—IntServ provides guaranteed service (Hard QoS). It uses signaling to reserve and guarantee resources for each traffic flow below it. RSVP is the common signaling protocol for resource reservation signaling on IP networks. Per-flow signaling and monitoring escalate the overhead of the IntServ model and make it nonscalable.

- **DiffServ**—DiffServ is the most modern of the three models. It requires traffic classification and marking and providing differentiated service to each traffic class based on its marking. DiffServ is scalable, but its drawback is that it requires implementation of complex QoS features on network devices throughout the network.

Network administrators have four methods at their disposal to implement QoS on their network's Cisco devices:

- **Cisco IOS CLI**—Configuring QoS using Cisco IOS CLI is the most complex and time-consuming method. It requires that you learn different syntax for each QoS mechanism.
- **MQC**—MQC is a modular command-line interface that is common across different Cisco platforms, and it separates the task of defining different traffic classes from the task of defining QoS policies.
- **Cisco AutoQoS**—Because AutoQoS automatically generates QoS commands on your router or switch, it is the simplest and fastest method among the four QoS implementation methods. However, should you need to fine-tune the AutoQoS configuration results, you must use MQC (or CLI) to do so. Fine-tuning of the commands that AutoQoS generates is seldom necessary.
- **Cisco Router and Security Device Manager (SDM) QoS Wizard**—Cisco SDM offers several wizards for implementing services, such as IPsec, VPN, and proactive management through performance monitoring, in addition to the QoS Wizard. Cisco SDM QoS Wizard allows you to remotely configure and monitor your Cisco routers without using the CLI. The SDM GUI makes it simple for you to implement QoS services, features, and policies.

Table 2-3 compares Cisco IOS CLI, MQC, AutoQoS, and SDM with respect to how easy they are to use, whether they allow you to fine-tune their results, how time consuming they are, and how modular they are.

Table 2-3 *Comparing QoS Implementation Methods*

Method	CLI	MQC	AutoQoS	SDM
Ease of use	Most difficult	Easier than legacy CLI	Simple	Simple
Ability to fine-tune	Yes (OK)	Very well	Limited	Limited
Time consuming to implement	Most time consuming (longest)	Moderate time consumed (average)	Least time consuming	Very little time consumed (short)
Modularity	Weakest (poor)	Very modular (excellent)	Very modular (excellent)	Good

MQC is the recommended and the most powerful method for implementing QoS. It is modular, it promotes re-use of written code, and it facilitates consistency of QoS configurations among your Cisco devices. MQC also reduces the chances for errors and conflicts, while allowing you to take advantage of the latest features and mechanisms offered by your version of Cisco IOS.

Q&A

Some of the questions that follow challenge you more than the exam by using an open-ended question format. By reviewing now with this more difficult question format, you can exercise your memory better and prove your conceptual and factual knowledge of this chapter. The answers to these questions appear in Appendix A.

1. List the four key quality issues with converged networks.
2. Provide a definition for maximum available bandwidth and average available bandwidth per flow.
3. List at least three types of delay.
4. Provide at least three ways to reduce delay.
5. Provide at least two ways to reduce or prevent loss of important packets.
6. Provide a definition for QoS.
7. List the three key steps in implementing QoS on a network.
8. List the three main QoS models.
9. Provide a short description of the best-effort model.
10. What are the benefits and drawbacks of the best-effort model?
11. Provide a short description for the IntServ model.
12. Name the functions that the IntServ model requires on the network routers and switches.
13. What are the benefits and drawbacks of the IntServ model?
14. What are the main features of the DiffServ model?
15. What are the benefits and drawbacks of the DiffServ model?
16. What are the four QoS implementation methods?
17. Which of the four QoS implementation methods is nonmodular and the most time consuming?
18. What are the main benefits of MQC?
19. What is the most important advantage of AutoQoS?
20. What are the prerequisites for Auto QoS VoIP?
21. What are the prerequisites for Auto QoS for the enterprise?
22. Which of the four QoS implementation methods is the fastest?
23. What are the three main tasks that you can accomplish using the SDM QoS Wizard?