EXAMPLE MSS and Bandwidth Efficiency

The value of the maximum segment size (MSS) affects the efficiency with which a given transmisson link is used. To discuss the effect of MSS on bandwidth efficiency consider Figure 8.34. Each TCP segment consists of a 20-byte TCP header followed by the block of data. The segment in turn is encapsulated in an IP packet that includes 20 additional bytes of header. The MSS for a connection is the largest block of data a segment is allowed to contain. TCP provides 16 bits to specify the MSS option, so the maximum block of data that can be carried in a segment is 65,495 bytes; that is, 65,535 minus 20 bytes for the IP header and 20 bytes for the TCP header.

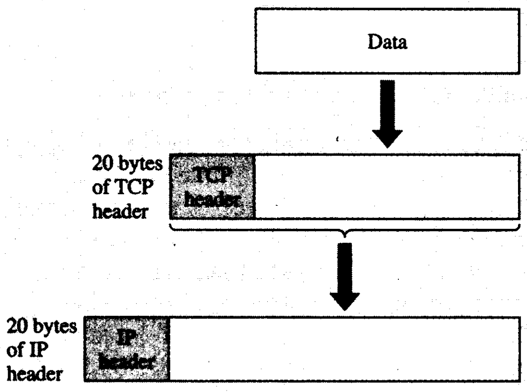


FIGURE 8.34 Header overhead.

In TCP the default MSS is 536 bytes, and the corresponding IP packet is then 576 bytes long. In this case 40 bytes out of every 576 bytes, that is, 7 percent are overhead. If instead, the MSS were 65,495 bytes, then the overhead would only be 0.06 percent. In practice, however, various networks impose limits on the size of the blocks they can handle. For example, Ethernet limits the MSS to 1460.

EXAMPLE Sequence Number Wraparound and Timestamps

The original TCP specification assumed a maximum segment lifetime (MSL) of 2 minutes. Let's see how long it takes to wrap around the sequence number space using current high-speed transmission lines. The 32-bit sequence number wraps around when $2^{32} = 4{,}294{,}967{,}296$ bytes have been sent. Because TCP uses Selective Repeat ARQ, the maximum allowable window size is 2^{31} bytes. In a T-1 line the time required is $(2^{32} \times 8)/(1.544 \times 10^6) = 6$ hours. In a T-3 line (45 Mbps), the wraparound will occur in 12 minutes. In an OC-48 line (2.4 Gbps), the wraparound will occur in 14 seconds! Clearly, sequence number wraparound becomes an issue for TCP operating over very high-speed links.

When the timestamp option is in use, the sending TCP inserts a 32-bit timestamp into a 4-byte field in the header of each segment it transmits. The receiving TCP echoes the 32-bit timestamp it receives by inserting the timestamp into a 4-byte field in the header of each ACK segment. By combining the 32-bit timestamp with the 32-bit sequence number we obtain what amounts to a 64-bit sequence number to deal with the wraparound problem. To be effective, the timestamp clock should tick forward at least once every 2^{31} bytes sent. This requirement places a lower bound on the clock frequency. The clock should also not be too fast; it should not complete a cycle in less than one MSL period. Clock frequencies in the range of one tick every 1 ms to 1 second meet these bound requirements. For example, a 1 ms clock period works for transmission rates up to 8 terabits/second, and the timestamp clock wraps its sign bit in 25 days [RFC 1323].

EXAMPLE Delay-Bandwidth Product and Advertised Window Size

Consider a link that has a round-trip time of 100 ms. The number of bytes in such a link at a T-1 speed is $1.544 \times 10^6 \times .100/8 = 19{,}300$ bytes. For a T-3 line (45 Mbps), the number of bytes is 562,500 bytes, and for an OC-48 line it increases to 3 megabytes. Suppose that a *single* TCP process needs to keep these links fully occupied transmitting its information. Its advertised window must then be at least as large as the RTT × bandwidth product. We saw at the beginning of this section that the normal maximum advertised window size is only 65,535, which is not enough for the T-3 and OC-48 lines. The window scale option, however, allows a window size of up to $65{,}535 \times 2^{14} =$ 1 gigabyte, which is enough to handle these cases.

TCP CONNECTION TERMINATION

TCP provides for a **graceful close** that involves the independent termination of each direction of the connection. A termination is initiated when an application tells TCP that it has no more data to send. The TCP entity completes transmission of its data and, upon receiving acknowledgment from the receiver, issues a segment with the FIN bit set. Upon receiving a FIN segment, a TCP entity informs its application that the other entity has terminated its transmission of data. For example, in Figure 8.35 the TCP entity in host A terminates its transmission first by issuing a FIN segment. Host B sends an ACK segment to acknowledge receipt of the FIN segment from A. Note that the FIN segment uses one byte, so the ACK is 5087 in the example.

After B receives the FIN segment, the direction of the flow from B to A is still open. In the figure host B sends 150 bytes in one segment, followed later by a FIN segment. Host A sends an acknowledgment. The TCP in host A then enters the **TIME_WAIT state** and starts the TIME_WAIT timer with an initial value set to twice the *maximum segment lifetime* (2MSL). The only valid segment that can arrive while host A is in the TIME_WAIT state is a retransmission of the FIN segment from host B (for example, if host A's ACK was lost, and host B's retransmission time-out has expired). If such a FIN segment arrives while host A is the TIME_WAIT state, then the ACK segment is retransmitted and the TIME_WAIT timer is restarted at 2MSL. When the TIME_WAIT timer expires, host A closes the connection and then deletes the record of the connection.

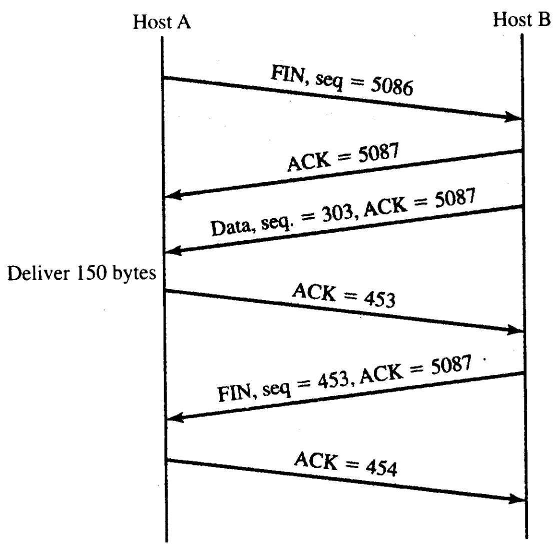


FIGURE 8.35 TCP graceful close: Host A sends a segment with FIN bit set to close a session and host B acknowledges receipt of the FIN segment.

The TIME_WAIT state serves a second purpose. The MSL is the maximum time that a segment can live inside the network before it is discarded. The TIME_WAIT state protects future incarnations of the connection from delayed segments. The TIME_WAIT forces TCP to wait at least two MSLs before setting up an incarnation of the old connection. The first MSL accounts for the maximum time a segment in one direction can remain in the network, and the second MSL allows for the maximum time a reply in the other direction can be in the network. Thus all segments from the old connection will be cleared from the network at the end of the TIME_WAIT state.

Figure 8.31 shows a packet capture of a graceful close in a Telnet example. Frames 6 and 7 close the connection from the client to the server. Frames 8 and 9 close the connection from the server to the client.

SAYING GOODBYE IS HARD TO DO

Under normal conditions the TCP closing procedure consists of one or more transmissions of a final ACK from host A until a final ACK successfully reaches host B. Host B then closes its TCP connection. Subsequently, host A closes its TCP connection after the expiry of the TIME_WAIT state. It is possible, however, that all retransmitted FIN segments from host B can get lost in the network so that host A closes its connection before host B receives the final ACK. Let's see why this situation is unavoidable. Suppose we insisted that host A not close its connection until it knew that host B had received the final ACK. How is host A supposed to know this? By getting an ACK from B acknowledging A's final ACK! But this scenario is the same problem we started with, all over again, with hosts A and B exchanging roles. Thus the sensible thing to do is for host B to retransmit its FIN some number of times and then to quit without receiving the ACK from A.

EXAMPLE Client/Server Application Revisited

Let us revisit the connection termination process depicted in Figure 8.35 in the context of a client/server application. Once more let the client reside in host A and the server in host B. The client initiates an *active close* by issuing a `close` call informing the server that the TCP module in host A has no more data to send. The TCP module in host A must still be prepared to receive data from server B. When the server is done transmitting data, it issues the `close` call. This action causes the host B TCP module to issue its FIN segment.

TCP provides for an abrupt connection termination through reset (RST) segments. An RST segment is a segment with the RST bit set. If an application decides to terminate the connection abruptly, it issues an ABORT command, which causes TCP to discard any data that is queued for transmission and to send an RST segment. A TCP module that receives the RST segment then notifies its application process that the connection has been terminated.

RST segments are also sent when a TCP module receives a segment that is not appropriate. For example, an RST segment is sent when a connection request arrives for an application process that is not listening on the given port. Another example involves half-open connections that arise from error conditions. For example, an application in host A may crash. The TCP module in host B, unaware of the crash, may then send a segment. Upon receiving this segment, the TCP module in host A sends an RST segment, thus informing host B that the connection has been terminated.

TCP STATE TRANSITION DIAGRAM

A TCP connection goes through a series of states during its lifetime. Figure 8.36 shows the state transition diagram. Each arrow indicates a state transition, and the associated label indicates associated events and actions. Connection establishment begins in the CLOSED state and proceeds to the ESTABLISHED state. Connection termination goes from the ESTABLISHED state to the CLOSED state. The normal transitions for a client are indicated by thick solid lines, and the normal transitions for a server are denoted by dashed lines. Thus when a client does an active open, it goes from the CLOSED state, to SYN_SENT, and then to ESTABLISHED. The server carrying out a passive open goes from the CLOSED state, to LISTEN, SYN_RCVD, and then to ESTABLISHED.

The client normally initiates the termination of the connection by sending a FIN. The associated state trajectory goes from the ESTABLISHED state, to FIN_WAIT_1 while it waits for an ACK, to FIN_WAIT_2 while it waits for the other side's FIN, and then to TIME_WAIT after it sends the final ACK. When the TIME_WAIT 2MSL period expires, the connection is closed and the transmission control block that stores all the TCP connection variables is deleted. Note that the state transition diagram does not show all error conditions that may arise, especially in relation to the TIME_WAIT state. The server normally goes from the ESTABLISHED state to the CLOSE_WAIT state after it receives a FIN, to the LAST_ACK when it sends its FIN, and finally to CLOSE when it receives the final ACK. We explore other possible state trajectories in the problem section at the end of the chapter.

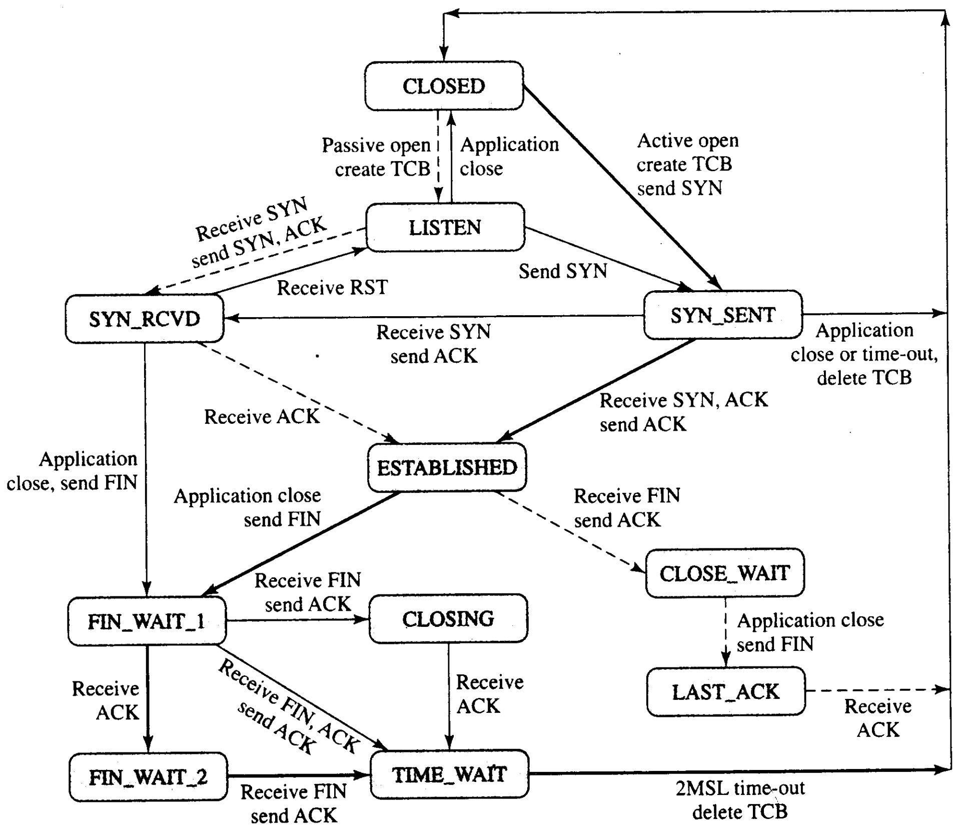


FIGURE 8.36 TCP state transition diagram. Note: The thick solid line is the normal state trajectory for a client; the dashed line is the normal state trajectory for a server.

8.5.3 TCP Congestion Control

Recall from Chapter 5 that TCP uses a sliding window protocol for end-to-end flow control. This protocol is implemented by having the receiver specify in its acknowledgment (ACK) the amount of bytes it is willing to receive in the future, called the *advertised window*. The advertised window ensures that the receiver's buffer will never overflow, since the sender cannot transmit data that exceeds the amount that is specified in the advertised window. Unfortunately, the advertised window does not prevent the buffers in the intermediate routers from overflowing, hence the congestion situation. Because the IP layer does not implement a mechanism to control congestion, it is up to the higher layer to detect congestion and take proper actions. It turns out that TCP window mechanism can also be used to control congestion in the network.

If TCP senders are too aggressive by sending too many packets, the network will eventually experience congestion. On the other hand, if TCP senders are too conservative, the network will be underutilized. The objective of TCP congestion control is to have each sender transmit just the right amount of data to keep the network resources utilized but not overloaded. To achieve this objective, the TCP protocol defines another window, called the **congestion window**, which specifies the maximum number of bytes that a TCP sender is allowed to transmit with the assumption that congestion will not be triggered with the given amount of data. The TCP congestion control algorithm at the sender dynamically adjusts the congestion window according to the current condition of the network. To avoid both network congestion and receiver buffer overflow, the maximum amount of data that a TCP sender is actually allowed to transmit at any time is the minimum of the advertised window and the congestion window. We will see that TCP regulates the rate at which segments are transmitted into the network. The segments in turn generate the IP packets that traverse the network.

The operation of the TCP congestion control algorithm may be divided into three phases (the following discussion assumes that the source always has data to transmit). The first phase is run when the algorithm begins a data transfer or when the algorithm restarts after recovery from segment loss. The technique is called **slow start** and is accomplished by first setting the congestion window to a small value (usually one MSS). Each time the sender receives an ACK from the receiver, the sender increases the congestion window by one segment (1 MSS). Therefore, after sending the first segment, if the sender receives an ACK before a time-out, the sender increases the congestion window to two segments. Later, if these two segments are acknowledged, the congestion window increases to four segments, and so on. As shown in Figure 8.37 the congestion window size grows exponentially during the slow-start phase. The reason for the exponential increase is that slow start needs to fill the pipe as quickly as possible to utilize network resources maximally. The name "slow start" is perhaps a misnomer, since the algorithm ramps up very quickly.

After a certain point, it may be unwise to keep increasing the congestion window exponentially, since overshoot could be significant. Specifically, the slow start phase ends when the congestion window reaches or exceeds a certain value specified as the

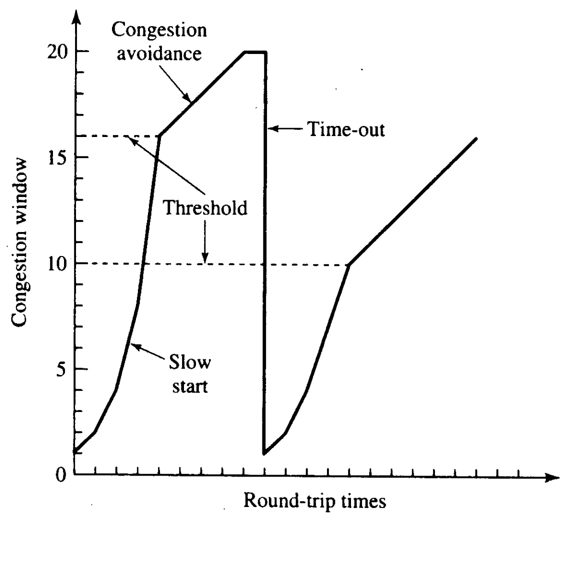


FIGURE 8.37 Dynamics of TCP congestion window.

congestion threshold (which may be set initially to an arbitrarily high value). At this point the **congestion avoidance** phase takes over. This phase assumes that the pipe is approaching full utilization. It is thus wise for the algorithm to reduce the rate of increase so that it will not overshoot the pipe capacity. Specifically, the algorithm increases the congestion window linearly rather than exponentially during congestion avoidance. This is realized by increasing the congestion window by one segment for each round-trip time.

Obviously, the congestion window also cannot be increased indefinitely, since the capacity of the network is finite. The congestion window stops increasing when TCP detects that the network is congested. (We momentarily defer the discussion of how congestion is detected.) The algorithm enters the third phase. Assuming that the current congestion window correlates to the point where the pipe is full, TCP sets the **congestion threshold** to one-half the current congestion window for probing the pipe next time. Then the congestion window is set to one maximum-sized segment, and the algorithm restarts, using the slow start technique.

Figure 8.37 illustrates the dynamics of the congestion window as time progresses. Initially slow start kicks in and the sender transmits one segment. After the first round-trip time, the sender transmits two additional segments. At the second round-trip time, the sender transmits four additional segments, and so on until the congestion threshold (set at 16 segments) is reached. Then congestion avoidance increases the window linearly so that the sender can only increment one segment at each round-trip time. When the congestion window is 20, a time-out occurs indicating that the network is congested, and the congestion threshold is then set to 10 and the congestion window is reset. The algorithm then resumes with slow start again.

Congestion in the network typically causes many segments to be dropped, resulting in missing ACKs in the reverse direction. The TCP sender uses this fact and assumes that congestion occurs in the network when an ACK does not arrive before the retransmission timer expires. However, when only one segment is dropped, the TCP sender can recover more quickly. The reason is that subsequent segments trigger the TCP receiver to transmit duplicate ACKs, and these triggered ACKs usually arrive at the TCP sender before the retransmission timer expires. When the TCP sender receives three duplicate ACKs, it performs **fast retransmit** by immediately retransmitting the lost segment. The sender then performs **fast recovery** by first setting the congestion threshold as explained earlier. Then the congestion window is set to the congestion threshold plus three MSSs to account for the additional three segments that have caused the duplicate ACKs. The TCP sender then continues in the congestion avoidance phase.[7]

One basic assumption underlying TCP congestion control is that segment loss is due to congestion rather than errors. This assumption is quite valid in a wired network where the percentage of segment losses due to transmission errors is generally very low (less than 10^{-6}). However, it should be noted that this assumption might not be valid in a wireless network where transmission errors can be relatively high. Interaction between TCP congestion control and wireless link data link control is a topic of current interest.

[7]See RFC 2581 for more details.

In the estimation of RTT, an ambiguity exists when a segment is retransmitted because the time-out has expired. When the acknowledgment eventually comes back, do we associate it with the first segment or with the retransmitted segment? Karn proposed ignoring the RTT when a segment is retransmitted because using the RTT might corrupt the estimate. This idea is generally called *Karn's algorithm.*

8.6 INTERNET ROUTING PROTOCOLS

The global Internet topology can be viewed as a collection of autonomous systems. An **autonomous system (AS)** is loosely defined as a set of routers or networks that are technically administered by a single organization such as a corporate network, a campus network, or an ISP network. There are no restrictions that an AS should run a single routing protocol within the AS. The only important requirement is that to the outside world, an AS should present a consistent picture of which ASs are reachable through it.

There are three categories of ASs:

1. *Stub AS* has only a single connection to the outside world. Stub AS is also called single-homed AS.
2. *Multihomed AS* has multiple connections to the outside world but refuses to carry **transit traffic** (traffic that originates and terminates in the outside world). Multihomed AS carries only **local traffic** (traffic that originates or terminates in that AS).
3. *Transit AS* has multiple connections to the outside world and can carry transit and local traffic.

SMT A

For the purpose of AS identification, an AS needs to be assigned a globally unique AS number (ASN) that is represented by a 16-bit integer and thus is limited to about 65,000 numbers. We show later how ASNs are used in exterior (inter-AS) routing. Care must be taken not to exhaust the AS space. As of this writing, the number of registered ASs has exceeded ten thousand. Fortunately, a stub AS, which is the most common type, does not need an ASN, since the associated prefixes are placed at the provider's routing table. On the other hand, a transit AS needs an ASN. At present, an organization may request an ASN from ARIN in North America, RIPE in Europe, or APNIC in Asia.

Routing protocols in the Internet are arranged in a hierarchy that involves two types of protocols: **Interior Gateway Protocol (IGP)** and **Exterior Gateway Protocol (EGP)**. IGP is used for routers to communicate within an AS and relies on IP addresses to construct paths. EGP is used for routers to communicate among different ASs and relies on AS numbers to construct AS paths. In this section we cover two popular IGPs: Routing Information Protocol and Open Shortest Path First. We will discuss the current de facto standard for EGP, which is Border Gateway Protocol (BGP) version 4 (BGP-4). As an analogy, IGP can be thought of as providing a map of a county detailing how to reach each building (host/router), while EGP provides a map of a country, connecting each county (AS).

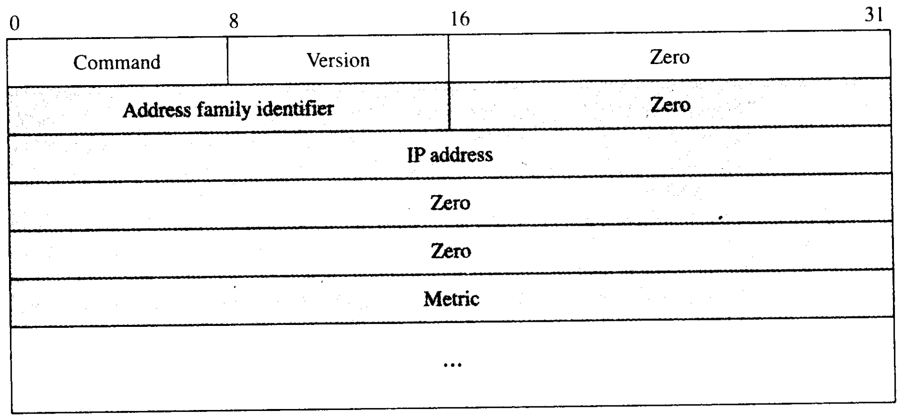


FIGURE 8.38 RIP message format.

8.6.1 Routing Information Protocol

Routing Information Protocol (RIP) is based on a program distributed in BSD[8] UNIX called `routed`[9] and uses the distance-vector algorithm discussed in Chapter 7. RIP runs on top of UDP via a well-known UDP port number 520. The metric used in the computation of shortest paths is typically configured to be the number of hops. The maximum number of hops is limited to 15 because RIP is intended for use in local area environments where the diameter of the network is usually quite small. The cost value of 16 is reserved to represent infinity. This small number helps to combat the count-to-infinity problem.

A router implementing RIP sends an update message to its neighbors every 30 seconds nominally. To deal with changes in topology such as a link failure, a router expects to receive an update message from each of its neighbors within 180 seconds in the worst case. The reason for choosing a value greater than 30 seconds is that RIP uses UDP, which is an unreliable protocol. Thus some update messages may get lost and never reach the neighbors. If the router does not receive the update message from neighbor X within this limit, it assumes that the direct link to X has failed and sets the corresponding minimum cost to 16 (infinity). If the router later receives a valid minimum cost to X from another neighbor, the router will replace infinity with the new minimum cost.

RIP uses split horizon with poisoned reverse to reduce routing loops.[10] Convergence is speeded up by requiring a router to implement triggered updates. However, a router may want to randomly delay the triggered update messages to avoid loading the network excessively.

The RIP message format is shown in Figure 8.38. The message consists of a command field, a version field, 16 bits that must be set to zero, and a variable number

[8]BSD stands for Berkeley Software Distribution.

[9]Pronounced "route dee" for route daemon.

[10]Split horizon with poisoned reverse was discussed in Section 7.5.1.

TABLE 8.4 RIP fields.

Field	**Description**
Command	The command field specifies the purpose of this message. Two values are currently defined: a value of 1 requests the other system to send its routing information, and a value of 2 indicates a response containing the routing information in the sender's routing table.
Version	This field contains the protocol version. RIP-1 set this field to 1, and RIP-2 sets this to 2.
Address family identifier	This field is used to identify the type of address. Currently, only IP address is defined, and the value is 2 for IP.
IP address	This field indicates the address of the destination, which can be a network or host address.
Metric	This field specifies the cost (number of hops) to the destination, which can range from 1 to 15. A value of 16 indicates that the destination is unreachable.

of routing information messages called RIP entries (up to 25 such entries). Fields that are set to zero are not used and reserved as placeholders for future extensions. Each RIP entry is 20 bytes long and consists of an address family identifier, an IP address, a metric, and some fields that are set to zero. Table 8.4 lists the purpose of each field.

Although its simplicity is clearly an advantage, RIP also has some limitations, including limited metric use and slow convergence. With the use of hop counts and a small range of value (1 to 15) specified in the metric, the protocol cannot take account of network load conditions. Furthermore, the protocol cannot differentiate between high-bandwidth links and low ones. Although the split horizon helps speed up convergence, the protocol may perform poorly under certain types of failures.

RIP-2 allows the RIP packet to carry more information (e.g., subnet mask, next hop, and routing domain). RIP-2 also provides a simple authentication procedure. Unlike RIP-1, RIP-2 can be used with CIDR. The complete specification is documented in RFC 2453.

8.6.2 Open Shortest Path First

The **Open Shortest Path First (OSPF)** Protocol is an IGP protocol that was developed to fix some of the deficiencies in RIP. Unlike RIP where each router learns from its neighbors only the distance to each destination, OSPF enables each router to learn the complete network topology. RFC 2328 describes the most recent version of OSPF.

Each OSPF router monitors the cost (called the **link state**) of the link to each of its neighbors and then floods the link-state information to other routers in the network. For this reason OSPF is often called a link-state protocol. The flooding of the link-state information allows each router to build an identical **link-state database** (or topological database) that describes the complete network topology.

At steady state the routers will have the same link-state database, and so they will know how many routers are in the network, the interfaces and links between them, and the cost associated with each link. The information in the link-state database allows a router to build the shortest-path tree with the router as the root. The computation of the shortest paths is usually performed by Dijkstra's algorithm, although other routing

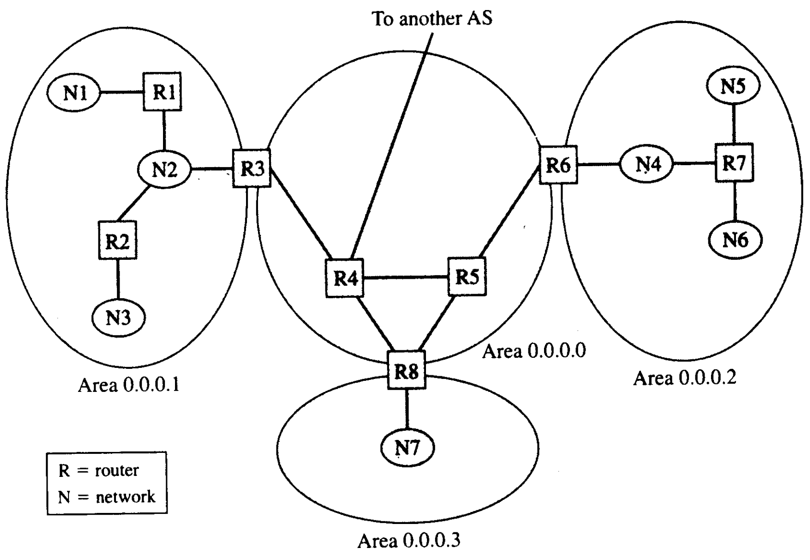


FIGURE 8.39 OSPF areas.

algorithms can be equally applied. Because the link-state information provides richer information than does distance-vector information, OSPF typically converges faster than RIP when a failure occurs in the network.

Some of the features of OSPF include

- Calculation of multiple routes to a given destination, one for each IP type of service.[11] This functionality provides an added flexibility that is not available in RIP.
- Support for variable-length subnetting by including the subnet mask in the routing message.
- A more flexible link cost that can range from 1 to 65,535. The cost can be based on any criteria.
- Distribution of traffic (load balancing) over multiple paths that have equal cost to the destination. *Equal-cost multipath (ECMP)* is a simple form of traffic engineering.
- Authentication schemes to ensure that routers are exchanging information with trusted neighbors.
- Multicast rather than broadcast of its messages to reduce the load on systems that do not understand OSPF.
- Use of a **designated router** (and a backup designated router) on multiaccess networks (where routers can talk directly to each other) to reduce the number of OSPF messages that are exchanged. The designated router is elected by a Hello protocol.

To improve scalability, OSPF introduces a two-level hierarchy that allows an AS to be partitioned into several groups called **areas**, that are interconnected by a central **backbone area** as shown in Figure 8.39. An area is identified by a 32-bit number known

[11]The type of service field was discussed in Section 8.2.1.

as the area ID. Continuing our previous analogy, an area can be thought of as a city or town within a county (AS). The backbone area is identified with area ID 0.0.0.0. The topology of an area is hidden from the rest of the AS in the sense that each router in an area only knows the complete topology inside the area. This approach limits the flooding traffic to an area, which makes the protocol more scalable. The information from other areas is summarized by **area border routers (ABRs)** that have connections to multiple areas. The concept of areas allows OSPF to provide a two-level hierarchy where different areas can exchange packets through the backbone area.

Four types of routers are defined in OSPF. An *internal router* is a router with all its links connected to the networks within the same area. An *area border router* is a router that has its links connected to more than one area. A *backbone router* is a router that has its links connected to the backbone. Finally, an *autonomous system boundary router (ASBR)* is a router that has its links connected to another autonomous system. ASBRs learn about routes outside the AS through an exterior gateway protocol such as BGP. In Figure 8.39 routers 1, 2, and 7 are internal routers. Routers 3, 6, and 8 are area border routers. Routers 3, 4, 5, 6, and 8 are backbone routers. Router 4 is an ASBR.

Two routers are said to be *neighbors* if they have an interface to a common network. A **Hello protocol** allows neighbors to be discovered automatically. Neighbor routers are said to become *adjacent* when they synchronize their topology databases through the exchange of link-state information. Neighbors on point-to-point links become adjacent. Neighbors on multiaccess networks become adjacent to designated routers as explained below. The use of designated routers reduces the size of the topological database and the network traffic generated by OSPF.

A multiaccess network is simply a set of routers that can communicate directly with each other. (Think of a multiaccess network as a clique of friends.) In a broadcast multiaccess network, the routers communicate with each other by using a broadcast network such as a LAN. On the other hand, in nonbroadcast multiaccess (NBMA) networks, the routers communicate through a nonbroadcast network, for example, a packet-switched network such as ATM and frame relay. OSPF uses a *designated router* (and a backup designated router) on multiaccess networks. (Think of the designated router as the most popular member of the clique.) The number of OSPF messages that are exchanged is reduced by having the designated router participate in the routing algorithm on behalf of the entire multiaccess network; that is, the designated router generates the link advertisements that list the routers that are attached to its multiaccess network. (The most popular member will promptly give the rest of the clique the lowdown on the network state.) The designated router is elected by the Hello protocol.

OSPF OPERATION

The OSPF protocol runs directly over IP, using IP protocol 89. The OSPF header format is shown in Figure 8.40. Its fields are identified in Table 8.5. Each OSPF packet consists of an OSPF common header followed by the body of a particular packet type. There are five types of OSPF packets: hello, database description, link-state request, link-state update, and link-state ACK. OSPF packets are sent to the multicast address 224.0.0.5, which is recognized as the AllSPFRouters address on point-to-point links

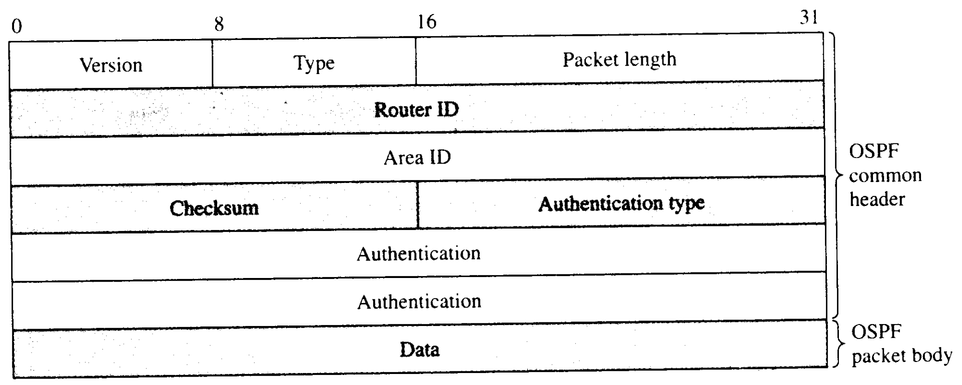


FIGURE 8.40 OSPF common header precedes each OSPF packet.

and on broadcast multiaccess networks. OSPF packets need to be sent to specific IP addresses in nonbroadcast multiaccess networks.

The OSPF operation consists of the following stages.

1. Neighbors are discovered through transmissions of Hello messages and designated routers are elected in multiaccess networks.
2. Adjacencies are established and link-state databases are synchronized.
3. Link-state advertisements (LSAs) are exchanged by adjacent routers to allow topological databases to be maintained and to advertise interarea and interAS routes. The routers use the information in the database to generate routing tables.

In the following discussion we indicate how the various OSPF packet types are used during these stages.

Stage 1: Hello packets

OSPF sends Hello packets (type 1) to its neighbors periodically to discover, establish, and maintain neighbor relationships. Hello packets are transmitted to each interface

TABLE 8.5 OSPF header fields.

Field	Description
Version	This field specifies the protocol version. The most current version is 2.
Type	The type field specifies the type of OSPF packet. The following types are defined: hello, database description, link-state request, link-state update, link-state acknowledgments.
Packet Length	This field specifies the length of OSPF packet in bytes, including the OSPF header.
Router ID	This field identifies the sending router. This field is typically set to the IP address of one of its interfaces.
Area ID	This field identifies the area this packet belongs to. The area ID of 0.0.0.0 is reserved for backbone.
Checksum	The checksum field is used to detect errors in the packet.
Authentication type and authentication	The combination of these fields can be used to authenticate OSPF packets.

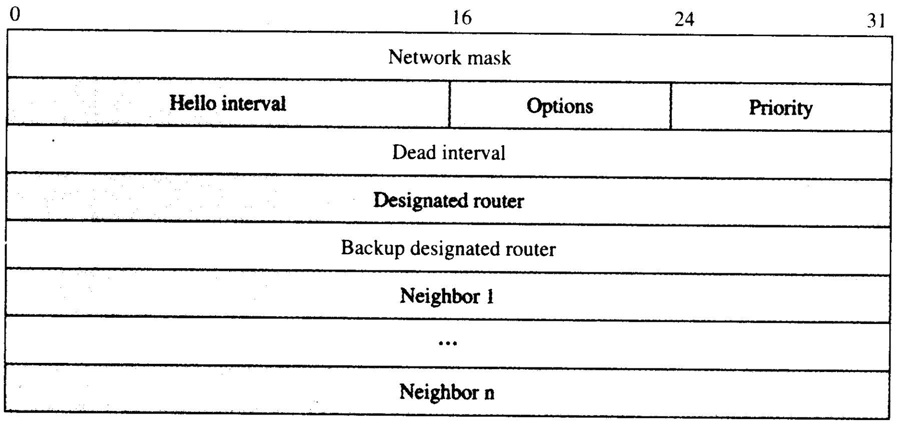


FIGURE 8.41 OSPF Hello packet format.

periodically, typically every 10 seconds. The format of the body of the Hello packet is shown in Figure 8.41, its fields are identified in Table 8.6. Each router broadcasts a Hello packet periodically onto its network. When a router receives a Hello packet, it replies with a Hello packet containing the router ID of each neighbor it has seen. When a router receives a Hello packet containing its router ID in one of the neighbor fields, the router is assured that communication to the sender is bidirectional. Designated routers are elected in each multiaccess network after neighbor discovery. The election is based on the highest value of the priority and ID fields.

Stage 2: Establishing adjacencies and synchronizing databases

OSPF involves establishing "adjacencies" between a subset of the routers in AS. Only routers that establish adjacencies participate in the operation of OSPF. Once two neighboring routers establish the connectivity between them, they exchange database

TABLE 8.6 Hello packet fields.

Field	Description
Network mask	The network mask is associated with the interface the packet is sent on.
Hello interval	This field specifies the number of seconds between Hello packets.
Options	Optional capabilities that are supported by the router.
Priority	This field specifies the priority of the router. It is used to elect the (backup) designated router. If set to 0, this router is ineligible to be a (backup) designated router.
Dead interval	This field specifies the number of seconds before declaring a nonresponding neighbor down.
Designated router	This field identifies the designated router for this network. This field is set to 0.0.0.0 if there is no designated router.
Backup designated router	Same as designated router.
Neighbor	This field gives the router ID of each neighbor from whom Hello packets have recently been received.

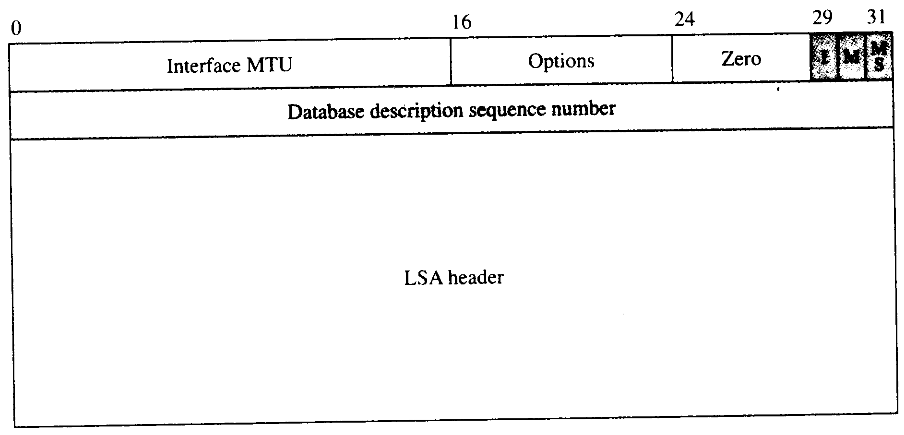


FIGURE 8.42 OSPF database description packet.

description packets (type 2) to synchronize their link-state databases. One router acts as a master and the other as a slave. Multiple packets may be used to describe the link-state database. The format of the database description packet is shown in Figure 8.42. Table 8.7 lists the fields. Note that the database description packet may contain multiple LSA headers. The routers send only their LSA headers instead of sending their entire database. The neighbor can then request the LSAs that it does not have. The synchronization of the link-state databases of all OSPF routers in an area is essential to synthesizing routers that are correct and free of loops.

The format of the LSA header is shown in Figure 8.43. Its corresponding fields are listed in Table 8.8. There are several link-state types. The *router link advertisement* is generated by all OSPF routers, and it gives the state of router links within the area. This information is flooded within the area only. The *network link advertisement* is generated by the designated router, lists the routers connected to the broadcast or NBMA network, and is flooded within the area only. The *summary link advertisement* is generated by

TABLE 8.7 OSPF database description packet fields.

Field	Description
Interface MTU	This field specifies the maximum transmission unit of the associated interface.
Options	Optional capabilities that are supported by the router.
I bit	The Init bit is set to 1 if this packet is the first packet in the sequence of database description packets.
M bit	The More bit is set to 1 if more database description packets follow.
MS bit	The Master/Slave bit indicates that the router is the master. Otherwise, it is the slave. The router with the highest router ID is the master.
Database description sequence number	This field identifies the packet number sequentially so that the receiver can detect a missing packet. The master sets the sequence number.
LSA header	The link state advertisement (LSA) header describes the state of the router or network. Each LSA header contains enough information to uniquely identify an entry in the LSA (type, ID, and advertising router).

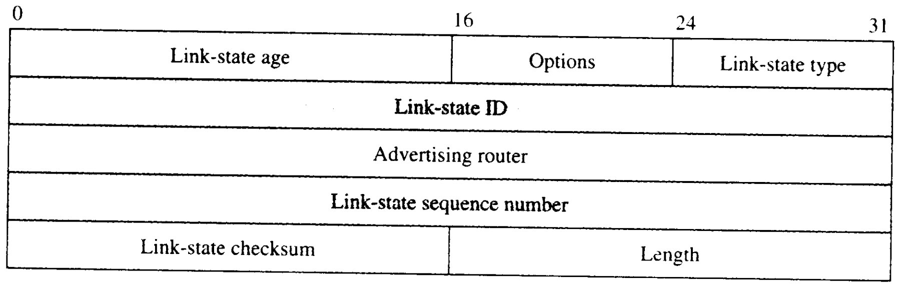


FIGURE 8.43 LSA header.

area border routers; it gives routes to destinations in other areas and routes to ASBRs. Finally, the AS *external link advertisement* is generated by ASBRs, describes routes to destinations outside the OSPF network, and is flooded in all areas in the OSPF network.

Stage 3: Propagation of link-state information and building of routing tables

When a router wants to update parts of its link-state database, it sends a *link-state request packet* (type 3) to its neighbor listing the LSAs it needs. The format of this message is shown in Figure 8.44. Each LSA request is specified by the link-state type, link-state ID, and the advertising router. These three fields are repeated for each link.

In response to a link-state request or when a router finds that its link state has changed, the router will send the new link-state information, using the *link-state update message* (type 4). The contents of the link-state update message are composed of LSAs, as shown in Figure 8.45.

OSPF uses reliable flooding to ensure that LSAs are updated correctly. Suppose that a router's local state has changed, so the router wishes to update its LSA. The router then issues a link-state update packet that invokes the reliable flooding procedure.

TABLE 8.8 LSA header fields.

Field	Description
Link-state age	This field describes how long ago the LSA was originated.
Options	Optional capabilities that are supported by the router.
Link-state type	This field specifies the type of the LSA. The possible types are: router, network, summary (for IP networks), summary (for ASB routers), and AS-external. In a network of routers connected by point-to-point links, OSPF uses only router LSAs.
Link-state ID	This field identifies the piece of the routing domain that is being described by the LSA. The contents of this field depend on the link-state type.
Advertising router	This field identifies the router ID of the router that originated the LSA.
Link-state sequence number	This field numbers the LSAs sequentially and can be used to detect old or duplicate LSAs.
Link-state checksum	The checksum includes the entire contents of the LSA except the link-state age.
Length	This field specifies the length in bytes of the LSA including this header.

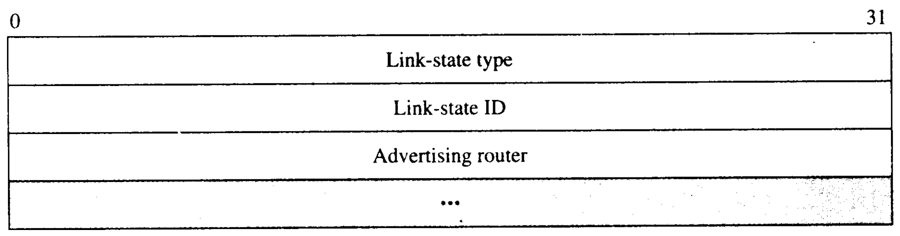


FIGURE 8.44 OSPF link-state request packet.

Upon receiving such a packet, a neighbor router examines the LSAs in the update. The neighbor installs each LSA in the packet update that is more recent than the corresponding LSA already in its database and then sends an LSA acknowledgment packet back to the router. These ACK packets consist of a list of LSA headers. The neighbor also prepares a new link-state update packet that contains the LSA and floods the packet on all interfaces other than the one in which the LSA arrived. All routers eventually receive the update LSA. When the link-state database is updated, the router needs to recompute the shortest path algorithm and modify the routing table according to the updated information. A router retransmits an LSA that has been sent to a neighbor periodically until receiving corresponding acknowledgment from the neighbor.

OSPF requires that all originators of LSAs refresh their LSAs every 30 minutes. This practice protects against accidental corruption of the router database.

Figure 8.46 shows a sequence of OSPF packet exchanges: Hello Packets, followed by link-state update packets, and then link-state acknowledgment packets. The middle pane shows the details of frame 11, which contains a link-state update packet. It can be seen that the packet carries five LSAs: the first two are summary link advertisements giving routes to other areas, and the latter three are AS external link advertisements describing routes to destinations outside the OSPF network. The middle pane also shows all the fields of the first summary LSA, which, in particular, describes a route to network 172.16.16.0 with network mask 255.255.255.0 and metric 64.

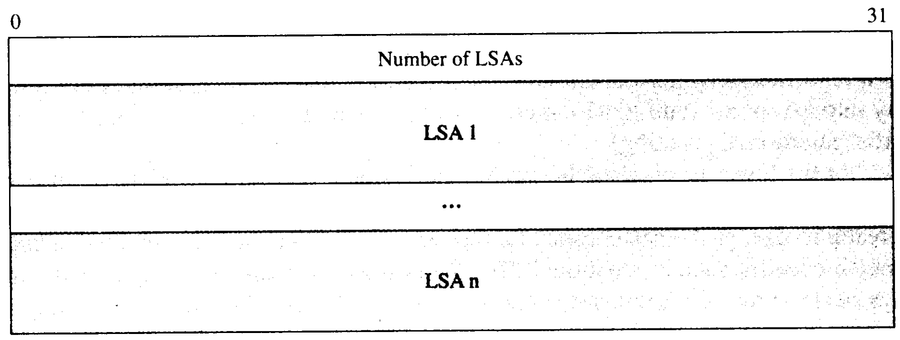
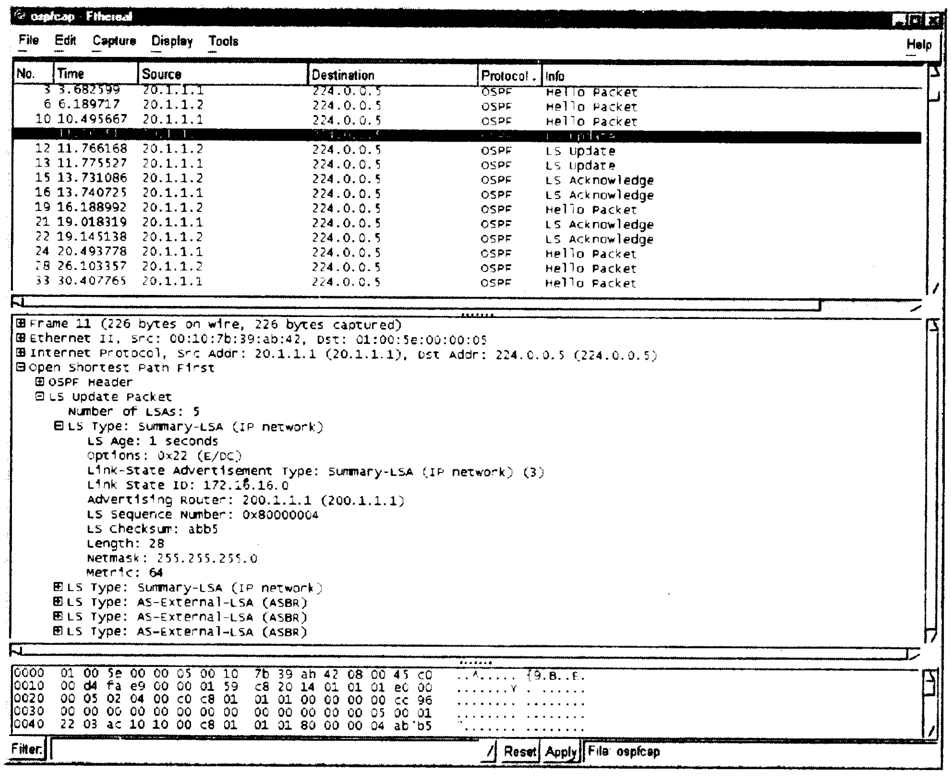


FIGURE 8.45 OSPF link-state update packet.



```
No.  Time       Source      Destination   Protocol  Info
   3 3.682599   20.1.1.1    224.0.0.5     OSPF      Hello Packet
   6 6.189717   20.1.1.2    224.0.0.5     OSPF      Hello Packet
  10 10.495667  20.1.1.1    224.0.0.5     OSPF      Hello Packet
  11 [illegible] 20.1.1.1   224.0.0.5     OSPF      LS Update
  12 11.766168  20.1.1.2    224.0.0.5     OSPF      LS Update
  13 11.775527  20.1.1.1    224.0.0.5     OSPF      LS Update
  15 13.731086  20.1.1.2    224.0.0.5     OSPF      LS Acknowledge
  16 13.740725  20.1.1.1    224.0.0.5     OSPF      LS Acknowledge
  19 16.188992  20.1.1.2    224.0.0.5     OSPF      Hello Packet
  21 19.018319  20.1.1.1    224.0.0.5     OSPF      LS Acknowledge
  22 19.145138  20.1.1.2    224.0.0.5     OSPF      LS Acknowledge
  24 20.493778  20.1.1.1    224.0.0.5     OSPF      Hello Packet
  28 26.103357  20.1.1.2    224.0.0.5     OSPF      Hello Packet
  33 30.407765  20.1.1.1    224.0.0.5     OSPF      Hello Packet
```

```
⊞ Frame 11 (226 bytes on wire, 226 bytes captured)
⊞ Ethernet II, Src: 00:10:7b:39:ab:42, Dst: 01:00:5e:00:00:05
⊞ Internet Protocol, Src Addr: 20.1.1.1 (20.1.1.1), Dst Addr: 224.0.0.5 (224.0.0.5)
⊟ Open Shortest Path First
    ⊞ OSPF Header
    ⊟ LS Update Packet
        Number of LSAs: 5
      ⊟ LS Type: Summary-LSA (IP network)
          LS Age: 1 seconds
          Options: 0x22 (E/DC)
          Link-State Advertisement Type: Summary-LSA (IP network) (3)
          Link State ID: 172.16.16.0
          Advertising Router: 200.1.1.1 (200.1.1.1)
          LS Sequence Number: 0x80000004
          LS Checksum: abb5
          Length: 28
          Netmask: 255.255.255.0
          Metric: 64
      ⊞ LS Type: Summary-LSA (IP network)
      ⊞ LS Type: AS-External-LSA (ASBR)
      ⊞ LS Type: AS-External-LSA (ASBR)
      ⊞ LS Type: AS-External-LSA (ASBR)
```

```
0000  01 00 5e 00 00 05 00 10  7b 39 ab 42 08 00 45 c0   ..^..... {9.B..E.
0010  00 d4 fa e9 00 00 01 59  c8 20 14 01 01 01 e0 00   .......Y . ......
0020  00 05 02 04 00 c0 c8 01  01 01 00 00 00 00 cc 96   ........ ........
0030  00 00 00 00 00 00 00 00  00 00 00 00 00 05 00 01   ........ ........
0040  22 03 ac 10 10 00 c8 01  01 01 80 00 00 04 ab b5   "....... ........
```

FIGURE 8.46 Example of an OSPF link state update packet.

ROUTING HIERARCHY AND INTERNET SCALABILITY

The Internet is a vast collection of networks that are logically linked by a globally unique address space and that provide communications using the TCP/IP protocol suite. Routing protocols are responsible for determining connectivity in the Internet and for generating the routing tables that direct packets to their destinations. In principle, the routing protocols must provide connectivity between every pair of routers in the Internet. This requirement poses a huge scalability challenge with the explosive growth of the Internet. The Autonomous System structure introduces a two-level hierarchy that decomposes the problem of determining Internet connectivity into two parts: routing within an AS (intradomain routing) and routing between ASs (interdomain routing).

At the lower level in the hierarchy, intradomain routing is handled by interior gateway protocols that identify optimal paths within an AS. However, ASs may vary greatly in size; an AS can consist of a campus network such as a university or a large transit network such as a national ISP. The problem of scale arises again in a large AS because the IGP must deal with all routers in the AS. To deal with routing in a large AS. OSPF introduces another level of hierarchy within the AS through the notion of areas. The IGP must then deal only with the routers inside an area.

At the higher level, the introduction of hierarchy also results in the problem of determining connectivity between ASs. Exterior gateway protocols such as BGP address this problem. BGP allows interdomain routers to advertise information about how to reach various networks in the Internet. Furthermore we will see that CIDR addressing allows BGP routers to advertise aggregated paths that reduce the amount of global routing information that needs to be exchanged.

8.6.3 Border Gateway Protocol

The purpose of an exterior gateway protocol is to enable two different ASs to exchange routing information so that IP traffic can flow across the AS border. Because each AS has its own administrative control, the focus of EGPs is more on policy issues (regarding the type of information that can cross a border) than on path optimality. The **Border Gateway Protocol (BGP)** is an interAS (or interdomain) routing protocol that is used to exchange network reachability information among BGP routers (also called BGP speakers).

First consider the example with the three autonomous systems shown in Figure 8.47. Suppose that AS3 wants to advertise to other ASs that network N1 can be reached through it. A BGP border router (R4) in AS3 would advertise this information to a neighboring BGP border router (R3). R3 examines this information and applies its policies to decide whether it wants to forward its packets to N1 via R4, and if so the routing table in R3 is updated to indicate R4 as the next hop for destination N1.

Now suppose that AS2 is also willing to carry transit packets from other ASs to network N1 in AS3. Consider how AS2 can disseminate the reachability of N1 to other ASs. In particular suppose that AS2 wishes to advertise network reachability information to AS1 using BGP border router R2. (Assume that AS2 has a means to convey to R2 BGP route information received at its other BGP border routers.) R2 sends a message to R1 that states that network N1 is reachable through AS2 and includes a set of attributes of the advertised route to N1. These attributes may include: the sequence of ASs that are traversed to reach N1, that is (AS2, AS3); the next hop router address; a list of metrics that indicate degrees of preference for the route; and origin information

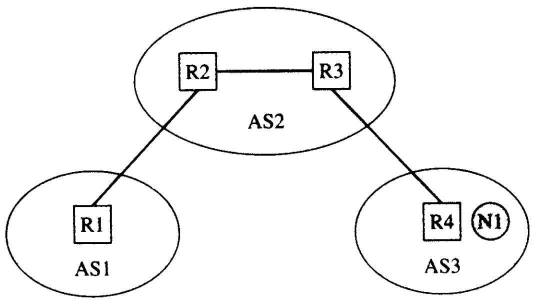


FIGURE 8.47 Interdomain routing with BGP.

indicating how the route was created at the source AS. The route attribute information provided by BGP allows policies to be enforced. Examples of policies are: "never use AS X"; "never use AS X to get to a destination in Y"; and "never use AS X and AS Y in the same path." We are now ready to discuss the operation of BGP.

Each BGP speaker establishes a TCP connection with one or more BGP speakers. In each TCP connection, a pair of BGP speakers exchange BGP messages to set up a BGP session, to exchange information about new routes that have become active or about old routes that have become inactive, and to report error conditions that lead to a termination of the TCP connection. A BGP speaker will advertise a route only if it is actively using the route to reach a particular CIDR prefix. The BGP advertisement also provides **attributes** about the path associated with the particular prefix.

The network reachability information that is exchanged by BGP speakers contains a sequence of ASs that packets must traverse to reach a destination network, or group of networks reachable through a certain prefix. The reachability information allows a BGP router to construct a graph of AS connectivity from its AS to other ASs with routing loops pruned (e.g., see Figure 8.48).

BGP is a path vector protocol in the sense that BGP advertises the sequence of AS numbers to the destination. An example of a path vector is route 10.10.1.0/24 is reachable via AS1, AS2, AS6, and AS7. Path vector information can easily be used to prevent a routing loop. When a router receives a path advertisement, the router ensures that its AS number does not appear in the path. If it does appear, the router can simply not use that path.

BGP can enforce policy by influencing the selection of different paths to a destination and by controlling the redistribution of routing information. The BGP routing process begins with the application of import policies to accept, deny or set preferences on the route advertisements received from neighbors. Next the best routes are selected and the routing tables are set. Finally export policies are applied to determine which routes should be advertised to neighbors. Import policies can be used to filter out route information from sources that are considered unreliable. For example, an ISP may not consider all the routing information that it receives from a customer to be reliable. By preventing routes from being advertised, export policies can be used to reject traffic from certain ASs. For example, an ISP may have agreements to carry transit traffic from certain ISPs and not from others.

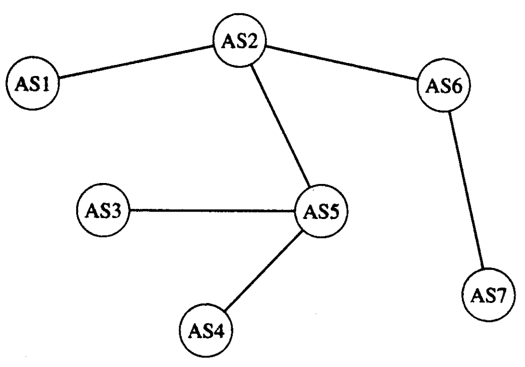


FIGURE 8.48 A graph of AS connectivity from AS1 to other ASs.

Two BGP speakers exchanging information on a connection are called peers or neighbors. BGP peers run over TCP on port 179 to exchange messages so that reliability is ensured. Using TCP to provide reliable service simplifies the protocol significantly, since complicated timer management is avoided. Using TCP also allows the protocol to update its routing table incrementally with high confidence.

Initially, BGP peers exchange the entire BGP routing table. Only incremental updates are sent subsequently, instead of periodic refreshes such as in OSPF or RIP. The incremental update approach is advantageous in terms of bandwidth usage and processing overhead. Keepalive messages are sent periodically to ensure that the connection between the BGP peers is alive and to detect a bad connection. These messages are typically sent every 30 seconds and, being only 19 bytes long, should consume minimal bandwidth. Notification messages are sent in response to errors or some exceptions.

Although BGP peers are primarily intended for communications between different ASs, they can also be used within an AS. BGP connections inside an AS are called **internal BGP (iBPG)**, and BGP connections between different ASs are called **external BGP (eBGP)**, as shown in Figure 8.49. Although eBGP peers are directly connected at the link layer, iBGP peers are typically not. The purpose of iBGP is to ensure that network reachability information is consistent among the BGP routers in the same AS. For example, in Figure 8.47 if the next-hop external route from R2 to AS1 is via R1, then the next-hop external route from R3 to AS1 is also via R1. A router can easily determine whether to run iBGP or eBGP by comparing the ASN of the other router with its ASN. All iBGP peers must be fully meshed logically so that all eBGP routes are consistent within an AS. As shown in Figure 8.49, an iBGP mesh may create too many TCP sessions. The connectivity required by iBGP mesh can be reduced through methods called *confederation* and *route reflection*.

BGP4 and CIDR have played a key role in enabling the Internet to scale to its current size. Large ISPs use path aggregation in their BGP advertisements to other ASs. An ISP can use CIDR to aggregate the addresses of many customers into a single advertisement, and thus reduce the amount of information required to provide routing to the customers. The ISP obtains the aggregate by filtering the set of paths so that only aggregates are advertised and more specific paths are not advertised.

BGP MESSAGES

All BGP messages begin with a fixed-size header that identifies the message type. Figure 8.50 shows the BGP message header format.

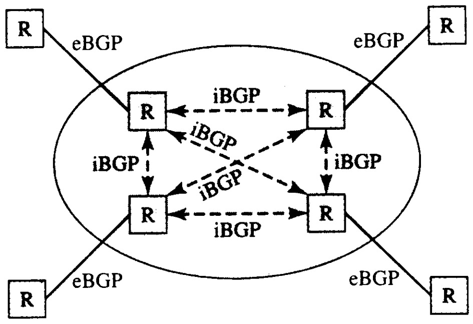


FIGURE 8.49 eBGP and iBGP.

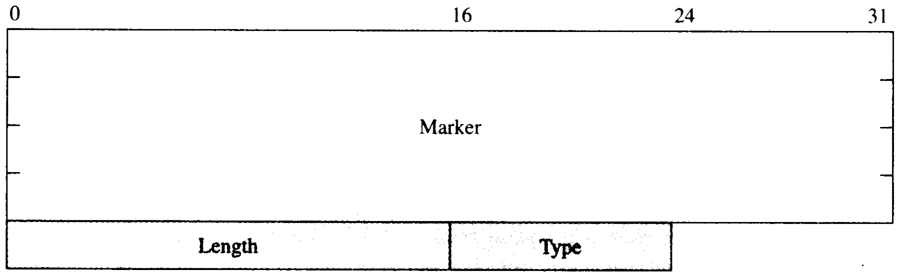


FIGURE 8.50 BGP header format.

A description of each field follows.

Marker: The marker field authenticates incoming BGP messages or detects loss of synchronization between a pair of BGP peers. If the message is an OPEN message, the marker field can be predicted based on some authentication mechanism used, which is likely to be Message-Digest Algorithm version 5 (MD-5). If the OPEN message carries no authentication, the marker field must be set to all 1s.

Length: This field indicates the total length of the message in octets, including the BGP header. The value of the length field must be between 19 and 4096.

Type: This field indicates the type of the message. BGP defines four message types:

1. OPEN
2. UPDATE
3. NOTIFICATION
4. KEEPALIVE

Each message type is discussed in the following four sections.

OPEN message

The first message sent by a BGP router after the TCP connection is established is the OPEN message. Figure 8.51 shows the format of the OPEN message.

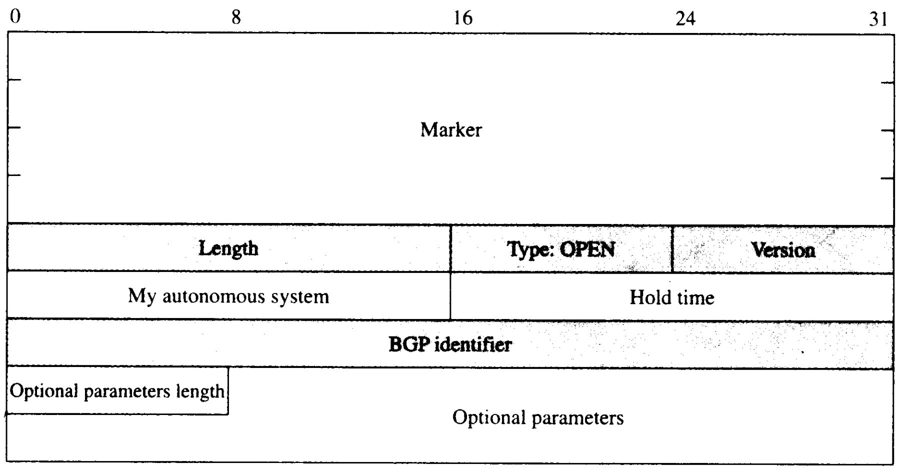


FIGURE 8.51 BGP OPEN message.

A description of each field follows.

Version: This field indicates the protocol version number of the message. The current BGP version number is 4.

My autonomous system: This field indicates the AS number of the sending router.

Hold time: This field is used by the sender to propose the number of seconds between the transmission of successive KEEPALIVE messages. The receiving router calculates the value of the Hold Time by using the smaller of its configured hold time and the hold time received in the OPEN message. A hold time value of zero indicates that KEEPALIVE messages will not be exchanged at all.

BGP identifier: This field identifies the sending BGP router. The value is determined by one of the IP local interface addresses of the BGP router and is used for all BGP peers regardless of the interface used to transmit the BPG messages. Some implementations use the highest IP address in the router or the address of the "virtual software interface."

Optional parameters length: This field indicates the total length of the optional parameters field in octets. This field is set to zero if there is no optional parameter.

Optional parameters: This field contains a list of optional parameters, encoded in TLV (that is, parameter type, parameter length, parameter value) structure. Currently, only authentication information (defined as parameter type 1) is defined. The parameter value contains an authentication code field (one octet) followed by an authentication data field, which is variable length. The authentication code specifies the type of authentication mechanism used, the form and meaning of the authentication data, and the algorithm used for computing the value of the marker.

KEEPALIVE message

BGP speakers continuously monitor the reachability of the peers by exchanging the KEEPALIVE messages periodically. The KEEPALIVE message is just the BGP header with the type field set to 4 (see Figure 8.52). The KEEPALIVE messages are exchanged often enough to prevent the hold timer from expiring. A recommended time between successive KEEPALIVE messages is one-third of the hold time interval. This value ensures that KEEPALIVE messages arrive at the receiving router almost always before the hold timer expires even if the transmission delay of a TCP is variable. If the hold time is zero, then KEEPALIVE messages will not be sent.

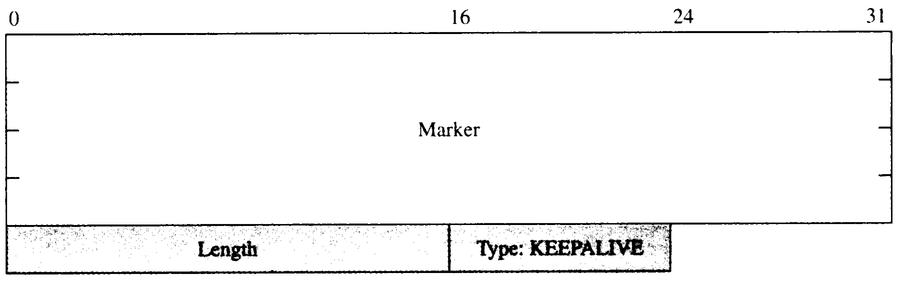


FIGURE 8.52 KEEPALIVE message format.

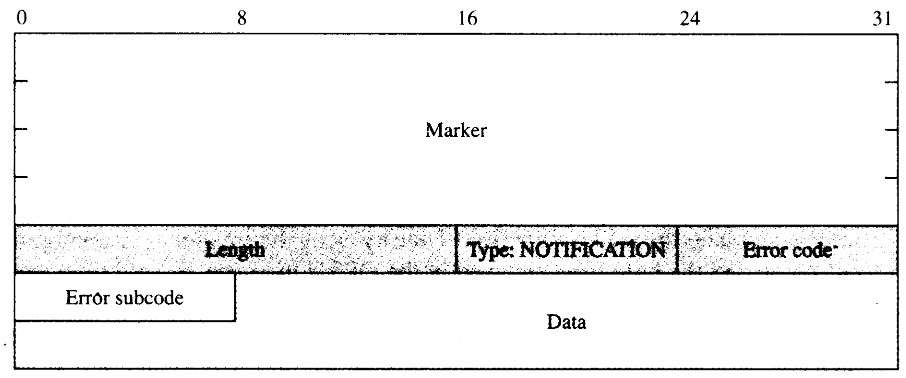


FIGURE 8.53 NOTIFICATION message format.

NOTIFICATION message

When a BGP speaker detects an error or an exception, the speaker sends a NOTIFICATION message and then closes the TCP connection. Figure 8.53 shows the NOTIFICATION message format. The error code indicates the type of error condition, while the error subcode provides more specific information about the nature of the error. The data field describes the reason for the notification, whose length can be deduced from the message length. The error codes and error subcodes are defined in Table 8.9.

UPDATE message

After the connection is established, BGP peers exchange routing information by using the UPDATE messages. The UPDATE messages are used to construct a graph of AS connectivity. As shown in Figure 8.54, an UPDATE message may contain three pieces of information: unfeasible routes, path attributes, and network layer reachability

TABLE 8.9 BGP error codes and subcodes.

Code	Description (with subcodes where present)			
1	Message header error			
	1	Connection Not Synchronized	3	Bad Message Type
	2	Bad Message Length		
2	OPEN message error			
	1	Unsupported Version Number	4	Unsupported Optional Parameter
	2	Bad Peer AS	5	Authentication Failure
	3	Bad BGP Identifier	6	Unacceptable Hold Time
3	UPDATE message error			
	1	Malformed Attribute List	7	AS Routing Loop
	2	Unrecognized Well-known Attribute	8	Invalid NEXT_HOP Attribute
	3	Missing Well-known Attribute	9	Optional Attribute Error
	4	Attribute Flags Error	10	Invalid Attribute Error
	5	Attribute Length Error	11	Malformed AS_PATH
	6	Invalid Origin Attribute		
4	Hold timer expired			
5	Finite state machine error			
6	Cease			

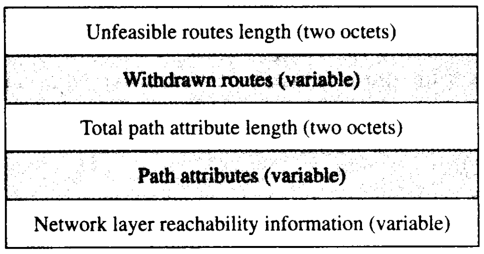

Unfeasible routes length (two octets)
Withdrawn routes (variable)
Total path attribute length (two octets)
Path attributes (variable)
Network layer reachability information (variable)

FIGURE 8.54 UPDATE message.

information (NLRI). The figure shows the big picture of the body of the UPDATE message. An UPDATE message can advertise a single route and/or withdraw a list of routes.

The withdrawn routes field provides a list of IP address prefixes for the routes that need to be withdrawn from BGP routing tables. Each IP address prefix is encoded by a 2-tuple of the form ⟨length, prefix⟩. The length field (one octet long) indicates the length in bits of the IP address prefix. A length of zero indicates a prefix that matches all the IP address. The prefix field contains the IP address prefix possibly followed by padding bits so that the length is a multiple of 8 bits.

The unfeasible routes length field (two octets) indicates the total length of the withdrawn routes field in octets. An UPDATE message can withdraw multiple unfeasible routes from service. If a BGP router does not intend to withdraw any routes, then the unfeasible routes length field is set to zero and the withdrawn routes field is not present.

A BGP router uses the NLRI along with the total path attributes length and the path attributes to advertise a route as shown in Figure 8.54. The NLRI field contains a list of IP address prefixes that can be reached by the route. The NLRI is encoded as one or more 2-tuples of the form ⟨length, prefix⟩. The length of the NLRI field is not encoded explicitly, but is deduced from the message length, total path attribute length, and unfeasible routes length. Path attributes describe the characteristics of the route and are used to affect routing behavior. If the NLRI field is present, the total path attribute length field indicates the total length of the path attributes field in octets. Otherwise, this field is set to zero, meaning that no route is being advertised.

An UPDATE message has a variable-length sequence of path attributes. Each path attribute is a triple ⟨attribute type, attribute length, attribute value⟩, as shown in Figure 8.55. Figure 8.56 displays the format of the attribute type, consisting of attribute flags (one octet) and attribute type code (one octet). The O bit (higher-order bit) indicates whether the attribute is optional ($O = 1$) or well-known (required). The T bit indicates whether the attribute is transitive or nontransitive (local). Well-known attributes are always transitive. The P bit indicates whether the information in the optional transitive attribute is partial ($P = 1$) or complete. The E bit indicates whether the attribute length is one octet ($E = 0$) or two octets.

Some explanation on path attributes is in order. There are four categories of path attributes: well-known mandatory, well-known discretionary, optional transitive, and optional nontransitive. Well-known attributes must be recognized by all BGP speakers, and well-known mandatory attributes must be sent in any UPDATE message. On the

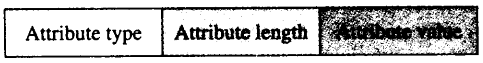


FIGURE 8.55 Format of each attribute.

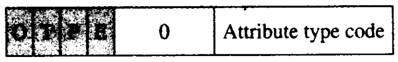


FIGURE 8.56 Attribute type.

other hand, optional attributes may not be recognized by a BGP speaker. Paths with unrecognized transitive optional attributes should be accepted and passed to other BGP peers with the P bit set to 1. Paths with unrecognized nontransitive optional attributes must be silently rejected.

The attribute type code field contains the attribute code. The attribute codes that are defined in the standards follow.

ORIGIN (type code 1): The ORIGIN attribute is a well-known mandatory attribute that defines the origin of the NLRI. The attribute value (one octet) can have one of three values. A value of 0 (for IGP) indicates that the NLRI is interior to the originating AS, for example, when IGP routes are redistributed to EGP. A value of 1 (for EGP) indicates that NLRI is learned via BGP. A value of 2 (for INCOMPLETE) indicates that NLRI is learned by some other means. This usually occurs when a static route is distributed to BGP and the origin of the route will be incomplete.

AS_PATH (type code 2): The AS_PATH attribute is a well-known mandatory attribute that lists the sequence of ASs that the route has traversed to reach the destination. When a BGP speaker originates the route, the speaker adds its own AS number when sending the route to its external peers. When a BGP speaker propagates a route that it has learned from another BGP speaker, the propagating speaker prepends its own AS number to the AS_PATH list. BGP uses the AS_PATH attribute to detect a potential loop. If the AS number of the BGP speaker is already contained in the list of AS numbers that the route has already been through, then the route should be rejected. When a customer uses a private AS number (64512-65535), the provider must make sure to strip the private AS number from the AS_PATH list, since these numbers are not globally unique.

NEXT_HOP (type code 3): The NEXT_HOP attribute is a well-known mandatory attribute that defines the IP address of the border router that should be used as the next hop to the destinations listed in the NLRI. Figure 8.57 clarifies the meaning of next hop. For eBGP the next hop is the IP address of the BGP peer. For example, R4 advertises 10.1.2.0/24 to R3 with a next hop of 10.10.1.2. For iBGP the next hop advertised by eBGP should be carried into iBGP. For example, R3 should also advertise 10.1.2.0/24 to R2 with a next hop of 10.10.1.2. To work properly, R2 should be able to reach 10.10.1.2 via IGP. This is done through *recursive route lookup*. For example, the IP routing table at R2 would indicate that the next hop for destination 10.1.2.0/24 is 10.10.1.2. Another lookup to 10.10.1.2 indicates that the next hop is 10.10.4.1. R2 can then send packets destined to 10.1.2.0/24 directly through R1, which will forward to R3, and so on.

MULTI_EXIT_DISC (type code 4): The MULTI_EXIT_DISC (MED) attribute is an optional nontransitive **attribute** that is used to discriminate among multiple entry/exit points to a neighboring AS and give a *hint to the neighboring AS about the preferred path*. The MED field is encoded in a four-octet unsigned integer. An exit or entry point with lower MED value should be preferred. The value can be derived from the IGP metric. The MED attribute received is never propagated to other ASs (i.e., it is nontransitive). Figure 8.58 illustrates the use of the MED attribute. In this example, R2 and R3 advertise the same route (10.1.1.0/24) to

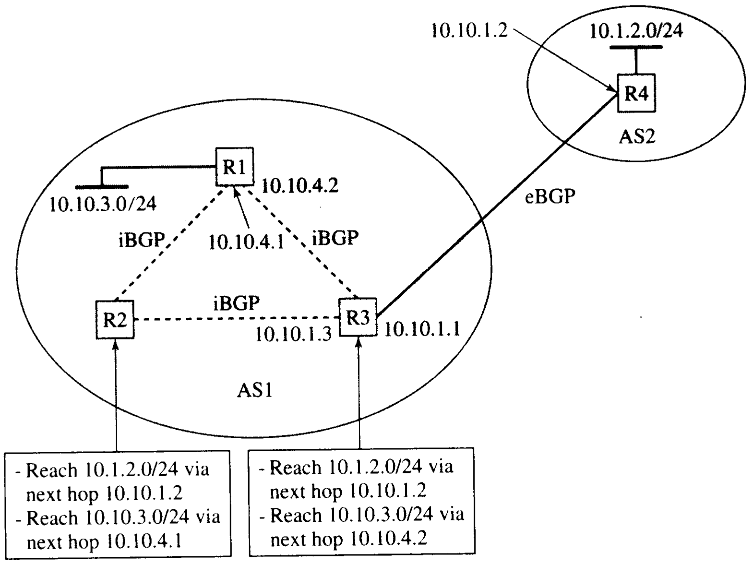

10.10.1.2
10.1.2.0/24
R4
AS2
eBGP
R1
10.10.3.0/24
10.10.4.2
iBGP
10.10.4.1
iBGP
iBGP
R2
R3
10.10.1.3
10.10.1.1
AS1
- Reach 10.1.2.0/24 via next hop 10.10.1.2
- Reach 10.10.3.0/24 via next hop 10.10.4.1
- Reach 10.1.2.0/24 via next hop 10.10.1.2
- Reach 10.10.3.0/24 via next hop 10.10.4.2

FIGURE 8.57 BGP NEXT_HOP.

R1. R2 uses a MED value of 100, while R3 uses a MED value of 200. It is a hint that AS2 prefers AS1 to use the route via R2 for this case. The usage of the MED attribute becomes complicated when another AS advertises the same route, since the IGP metrics used by different ASs can be different. In such a case comparing a MED value used by one AS with another MED value used by another AS makes no sense.

LOCAL_PREF (type code 5): The LOCAL_PREF attribute is a well-known discretionary attribute that informs other BGP speakers *within the same AS* of its degree of preference for an advertised route. The LOCAL_PREF attribute is exchanged only among the iBGP peers and should not be exchanged by the eBGP peers. The attribute that indicates the degree of preference for the exit points is encoded in a four-octet unsigned integer. A higher value indicates a higher preference.

ATOMIC_AGGREGATE (type code 6): The ATOMIC_AGGREGATE attribute is a well-known discretionary attribute that informs other BGP speakers that it selected a less specific route without selecting a more specific one that is included

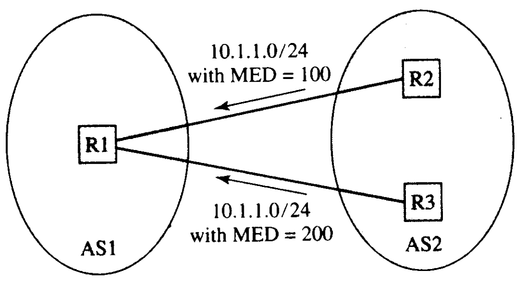
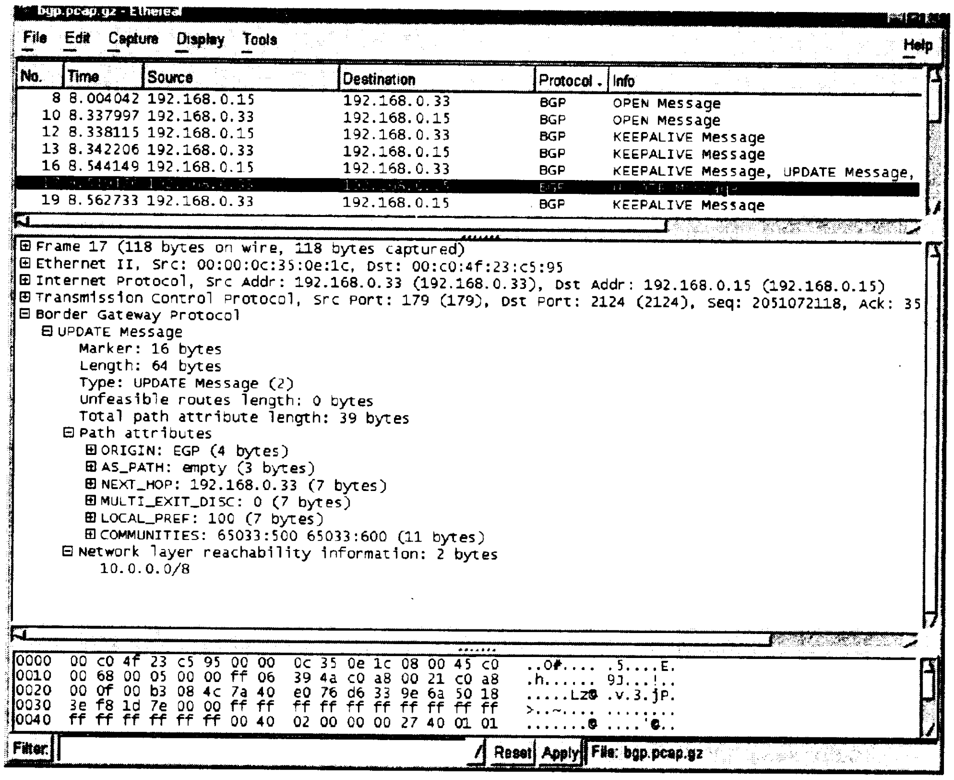


FIGURE 8.58 Example of the MED attribute.


bgp.pcap.gz - Ethereal
File Edit Capture Display Tools Help
Filter: Reset Apply File: bgp.pcap.gz

```
No.  Time      Source          Destination      Protocol  Info
 8 8.004042 192.168.0.15     192.168.0.33     BGP       OPEN Message
10 8.337997 192.168.0.33     192.168.0.15     BGP       OPEN Message
12 8.338115 192.168.0.15     192.168.0.33     BGP       KEEPALIVE Message
13 8.342206 192.168.0.33     192.168.0.15     BGP       KEEPALIVE Message
16 8.544149 192.168.0.15     192.168.0.33     BGP       KEEPALIVE Message, UPDATE Message,
17 [illegible] 192.168.0.33  192.168.0.15     BGP       UPDATE Message
19 8.562733 192.168.0.33     192.168.0.15     BGP       KEEPALIVE Message
```

```
⊞ Frame 17 (118 bytes on wire, 118 bytes captured)
⊞ Ethernet II, Src: 00:00:0c:35:0e:1c, Dst: 00:c0:4f:23:c5:95
⊞ Internet Protocol, Src Addr: 192.168.0.33 (192.168.0.33), Dst Addr: 192.168.0.15 (192.168.0.15)
⊞ Transmission Control Protocol, Src Port: 179 (179), Dst Port: 2124 (2124), Seq: 2051072118, Ack: 35
⊟ Border Gateway Protocol
  ⊟ UPDATE Message
      Marker: 16 bytes
      Length: 64 bytes
      Type: UPDATE Message (2)
      Unfeasible routes length: 0 bytes
      Total path attribute length: 39 bytes
    ⊟ Path attributes
      ⊞ ORIGIN: EGP (4 bytes)
      ⊞ AS_PATH: empty (3 bytes)
      ⊞ NEXT_HOP: 192.168.0.33 (7 bytes)
      ⊞ MULTI_EXIT_DISC: 0 (7 bytes)
      ⊞ LOCAL_PREF: 100 (7 bytes)
      ⊞ COMMUNITIES: 65033:500 65033:600 (11 bytes)
    ⊟ Network layer reachability information: 2 bytes
        10.0.0.0/8
```

```
0000  00 c0 4f 23 c5 95 00 00  0c 35 0e 1c 08 00 45 c0   ..O#.... .5....E.
0010  00 68 00 05 00 00 ff 06  39 4a c0 a8 00 21 c0 a8   .h...... 9J...!..
0020  00 0f 00 b3 08 4c 7a 40  e0 76 d6 33 9e 6a 50 18   .....Lz@ .v.3.jP.
0030  3e f8 1d 7e 00 00 ff ff  ff ff ff ff ff ff ff ff   >..~.... ........
0040  ff ff ff ff ff ff 00 40  02 00 00 00 27 40 01 01   .......@ ....'@..
```

FIGURE 8.59 Example of BGP packet exchange and update message.

in it. The attribute length for ATOMIC_AGGREGATE is zero (no attribute value). A BGP speaker that receives a route with the ATOMIC_AGGREGATE attribute should not remove the attribute when propagating it to other speakers. Also, the BGP speaker should not make the route more specific when advertising this route to other BGP speakers.

AGGREGATOR (type code 7): The AGGREGATOR attribute is an optional transitive attribute that specifies the last AS number that formed the aggregate route (encoded in two octets) followed by the IP address of the BGP speaker that formed the aggregate route (encoded in four octets).

Figure 8.59 contains a series of BGP packet exchanges between two BGP speakers. The middle pane describes the fields in a BGP update message for the path to prefix 10.0.0.0/8. The message indicates that the next hop router for this destination is 192.168.0.33.

8.7 MULTICAST ROUTING

The routing mechanisms discussed so far assume that a given source transmits its packets to a single destination. For some applications such as teleconferencing, a source may want to send packets to multiple destinations simultaneously. This requirement

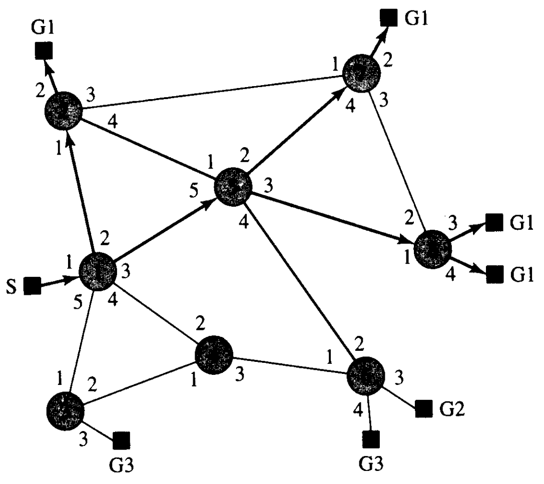


FIGURE 8.60 Multicast tree rooted at source S.

calls for another type of routing called **multicast routing** as illustrated in Figure 8.60. In the figure, source S wants to transmit to destinations with multicast group G1. Although the source can send each copy of the packet separately to each destination by using conventional unicast routing, a more efficient method would be to minimize the number of copies. For example, when router 1 receives a packet from the source, router 1 copies the packet to routers 2 and 5 simultaneously. Upon receipt of these packets, router 2 forwards the packet to its local network, and router 5 copies the packet to routers 7 and 8. The packet will eventually be received by each intended destination.

With multicast routing, each packet is transmitted once per link. If unicast routing is used instead, the link between the source and router 1 will carry four copies for each packet transmitted. In general, the bandwidth saving with multicast routing becomes more substantial as the number of destinations increases.

There are many ways to generate a multicast tree. One approach that is used in **multicast backbone (MBONE)** is called *reverse-path multicasting.* MBONE is basically an overlay packet network on the Internet supporting routing of IP multicast packets (with Class D addresses).

8.7.1 Reverse-Path Broadcasting

The easiest way to understand reverse-path multicasting is by first considering a simpler approach called **reverse-path broadcasting (RPB)**. RPB uses the fact that the set of shortest paths to a node forms a tree that spans the network. Assuming that each router already knows the current shortest path to a given destination, the operation of RPB is very simple:

- Upon receipt of a multicast packet, a router first records the source address of the packet and the port the packet arrives on.
- If the shortest path from the router back to the source is through the port the packet arrived on, the router forwards the packet to all ports except the one the packet arrived on; otherwise, the router drops the packet. The port over which the router expects to receive multicast packets from a given source is referred to as the **parent port**.

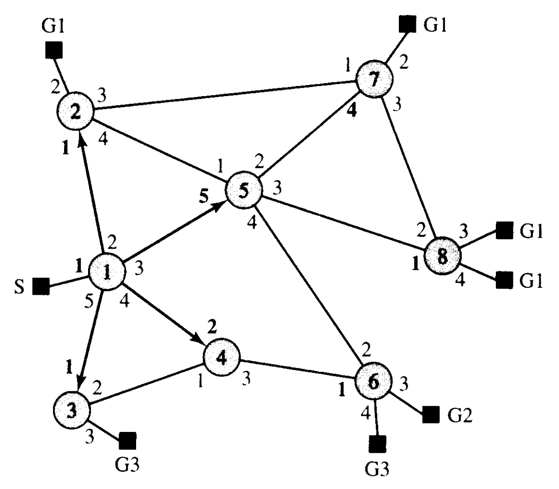


FIGURE 8.61 Router 1 copies the multicast packet (parent ports are in bold numbers).

The advantages of RPB are that routing loops are automatically suppressed and each packet is forwarded by a router exactly once. The basic assumption underlying the RPB algorithm is that the shortest path from the source to a given router should be the same as the shortest path from the router to the source. This assumption requires each link to be symmetric (each direction has the same cost). If links are not symmetric, then the router must compute the shortest path from the source to the router. This step is possible only if link-state protocols are used.

To see how RPB works for the network shown in Figure 8.61, assume that source S transmits a multicast packet with group G1. The parent ports for this multicast group are identifed in bold numbers, as shown in Figure 8.61. First router 1 receives a packet on port 1 from source S. Because the packet comes from the parent port, the router copies the packet to all other ports.

Assume that these packets reach routers 2, 3, 4, and 5 some time later. Router 2 computes the shortest path from itself to S and finds that the parent port is port 1. It then copies the packet through all other ports. Similarly, routers 3, 4, and 5 find that the packet arrives on the parent ports. As a result, each router copies the packet to its other ports, as shown in Figure 8.62. Note that the bidirectional link indicates that each router forwards the packet to the other.

Next assume that the packets arrive at routers 2 through 8. Because router 2 receives the packet from port 4, which is not the parent port, router 2 drops the packet. Routers 3, 4, and 5 also drop the packet for the same reason. Router 6, however, finds that the parent port is port 1, so router 6 copies the packet coming from router 4 to its other ports. The packet that came from router 5 to router 6 is dropped. Routers 7 and 8 copy the packet that came from router 5. Figure 8.63 illustrates this process. The reader should verify that the packet will no longer be propagated after this point.

Note that although the hosts connected to routers 3 and 6 belong to different multicast groups, the packets for group G1 are still forwarded by these routers. Typically, these hosts are attached to the router via a shared medium LAN. Therefore, unnecessary bandwidth is wasted regardless of the multicast group. This problem can be solved by

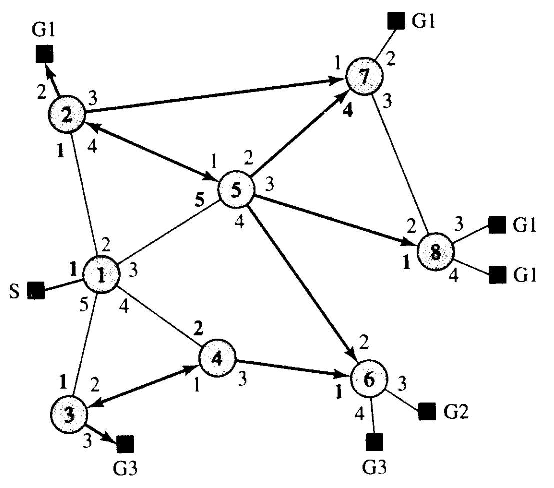


FIGURE 8.62 Routers 2, 3, 4, and 5 copy the packets.

having the router truncate its transmission to the local network if none of the hosts attached to the network belong to the multicast group. This refinement is called **truncated reverse-path broadcasting (TRPB)**. Note that TRPB performs truncation only at the "leaf" routers (routers at the edge of the network). The next section describes a protocol that allows a router to determine which hosts belong to which multicast groups.

8.7.2 Internet Group Management Protocol

The **Internet Group Management Protocol (IGMP)** allows a host to signal its multicast group membership to its attached router. IGMP runs directly on IP using protocol

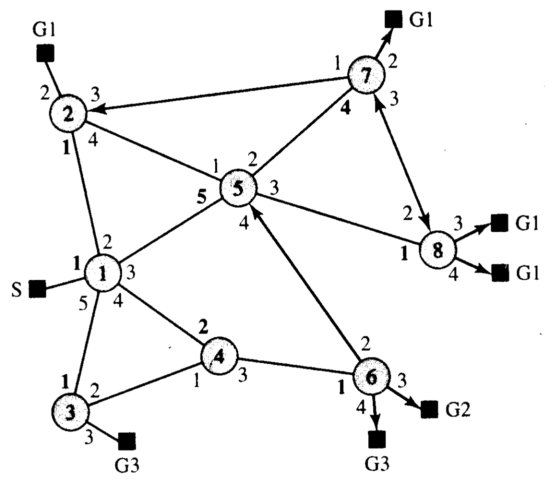


FIGURE 8.63 Routers 6, 7, and 8 copy the packets.

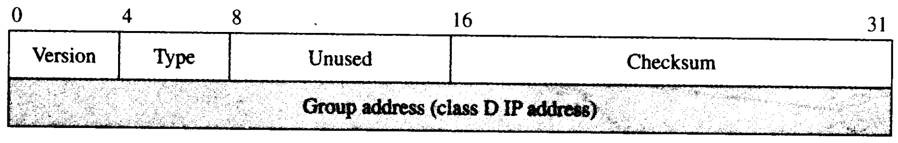


FIGURE 8.64 IGMP message format.

type 2. However, IGMP is usually considered as part of IP. The IGMP message format is very simple, as can be seen from Figure 8.64. A description of each field follows.

Version: This field identifies the version number.

Type: This field identifies the message type. There are two message types: Type 1 indicates a query message sent by a router, and type 2 indicates a report sent by a host.

Unused: This field must be set to zero.

Checksum: This field contains a checksum for all eight bytes of the IGMP message.

Group Address: This address is the class D IPv4 address. This field is set to zero in a query message, and is set to a valid group address in the response.

When a host wants to join a new multicast group, the host sends an IGMP report specifying the group address to join. The host needs to issue only one IGMP report, even though multiple applications are joining the same group. The host does not issue a report if it wants to leave the group.

A multicast router periodically issues an IGMP query message to check whether there are hosts belonging to multicast groups. The IGMP message is sent with the IP destination address set to the *all-hosts multicast address* (i.e., 224.0.0.1). When a host receives a query, it must respond with a report for each group it is joining. By exchanging queries and reports, a router would know which multicast groups are associated with a given port so that the router would forward an incoming multicast packet only to the ports that have hosts belonging to the group. Note that the router does not have to know how many hosts belong to a particular group. It only has to know that there is at least one host for a particular group.

To make sure that multiple hosts do not send a report at the same time when receiving a query that would lead to a collision, the scheduled transmission time should be randomized. Because the router only has to know that at least one host belongs to a particular group, efficiency can be improved by having a host monitor if another host sends the same report earlier than its scheduled transmission. If another host has already sent the same report, the first host cancels its report.

8.7.3 Reverse-Path Multicasting

As Section 8.7.2 just explained IGMP allows hosts to indicate their group members. We now continue with selective multicast routing called **reverse-path multicasting (RPM)**, which is an enhancement of TRPB. Unlike TRPB, RPM forwards a multicast packet only to a router that will lead to a leaf router with group members.

Each router forwards the first packet for a given (source, group) according to TRPB. All leaf routers receive at least the first multicast packet. If a leaf router does not find a member for this group on any of its ports, the leaf router will send a **prune message** to its upstream router instructing the upstream router not to forward subsequent packets belonging to this (source, group). If the upstream router receives the prune messages from all of its downstream neighbors, it will in turn generate a prune message to its further upstream router.

Referring to an earlier example in Figure 8.60, router 3 would send a prune message to routers 1 and 4. When router 1 receives the prune message, router 1 stops forwarding the subsequent multicast packets to port 5. Similarly, router 6 would send a prune message to routers 4 and 5. When router 4 realizes that it has received the prune messages from all of its downstream neighbors, router 4 sends a prune message to router 1, which subsequently stops forwarding the packets to router 4. The reader should verify that the multicast packets eventually follow the paths shown in Figure 8.60.

Each prune information is recorded in a router for a certain lifetime. When the prune timer expires, the router purges the information and starts forwarding the multicast packets to its downstream neighbors.

A host may later decide to join a multicast group after a prune message has been sent by its leaf router. In this case the leaf router would send a **graft message** to its upstream router to cancel its earlier prune message. Upon receiving the graft message, the first upstream router will forward subsequent multicast packets to this leaf router. If the first upstream router has also sent a prune message earlier, the first upstream router will send a graft message to the second upstream router. Eventually, a router in the multicast tree will reactivate its affected port so that the multicast packets will travel downstream toward the host. Figure 8.65 shows the grafting message flow when a host attached to router 6 wants to join the group. Subsequently, router 1 will forward the multicast packets to router 4, which will forward the multicast packets to router 6. Eventually, the multicast packets arrive at the host.

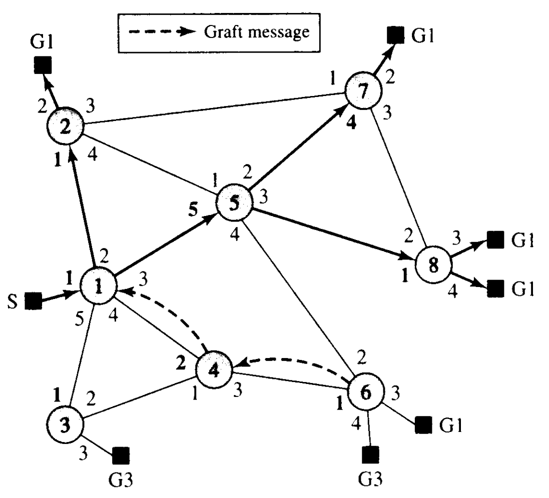


FIGURE 8.65 Grafting to cancel pruning: Router 6 sends graft message to rejoin a multicast group.

8.7.4 Distance-Vector Multicast Routing Protocol

Distance-Vector Multicast Routing Protocol (DVMRP) is the multicast routing algorithm that is used in the MBONE routers. DVMRP is based on a combination of RIP and RPM. DVMRP uses a variation of RIP to determine the previous hop toward the source. RPM and pruning are then used on the resulting source-level tree. DVMRP uses a tunneling mechanism to traverse networks that do not support multicasting. Tunnels are manually configured between two multicast router (mrouters).

One of the severe shortcomings of DVMRP is its inability to operate in a large internetwork. The first reason for this shortcoming is that the maximum number of hops is limited to 15. Second, it is not desirable to flood the entire Internet's leaf routers with the first multicast packet for a particular (source, group). Periodic refloodings due to the soft state mechanism associated with the prune information compound the problem. Also, the flat nature of the routing domain make it unsuitable for a large network.

8.8 DHCP, NAT, AND MOBILE IP

A host requires three elements to connect to the Internet: an IP address, a subnet mask, and the address of a nearby router. Each time a user moves or relocates, these elements must be reconfigured. In this section we discuss protocols that have been developed to automate this configuration process.

8.8.1 Dynamic Host Configuration Protocol

The **Dynamic Host Configuration Protocol (DHCP)** automatically configures hosts that connect to a TCP/IP network. An earlier protocol, Bootstrap Protocol (BOOTP), allowed diskless workstations to be remotely booted up in a network. DHCP builds on the capability of BOOTP to deliver configuration information to a host and uses BOOTP's well-known UDP ports: 67 for the server port and 68 for the client port. DHCP is in wide use because it provides a mechanism for assigning temporary IP network addresses to hosts. This capability is used extensively by Internet service providers to maximize the usage of their limited IP address space.

When a host wishes to obtain an IP address, the host broadcasts a DHCP Discover message in its physical network. The server in the network may respond with a DHCP Offer message that provides an IP address and other configuration information. Several servers may reply to the host, so the host selects one of the offers and broadcasts a DHCP Request message that includes the ID of the server. The selected server then allocates the given IP address to the host and sends a DHCP ACK message assigning the IP address to the host for some period of lease time T. The message also includes two time thresholds T1 (usually = .5T) and T2 (usually .875T). When time T1 expires the host attempts to extend the lease period by sending a DHCP Request to the original server. If the host gets the corresponding ACK message, it will also receive new values for T, T1, and T2. If the host does not receive an ACK by time T2, then it broadcasts a DHCP Request to any server on the network. If the host does not receive an ACK by time T, then it must relinquish the IP address and begin the DHCP process from scratch.

8.8.2 Network Address Translation

We have seen that three ranges of *unregistered* IP addresses have been set aside for use in private internets. For example, the class C range of addresses from 192.168.0.0 to 192.168.255.255 is used in most home networks. Packets with unregistered addresses are discarded by routers in the global Internet, so a computer must have a registered IP address in order to communicate over the Internet. **Network address translation (NAT)** refers to a method for mapping packets generated by machines in a private network into packets that can traverse the global Internet. NAT also transfers packets arriving from the global Internet to the appropriate destination machine in the private network. In particular, NAT allows a number of machines in a private network to share a limited number of registered IP addresses.

We will limit our discussion to the case where a single registered IP address is shared by the machines in a private network. NAT works as follows. When a machine in the private network generates a packet that has a destination outside the private network, the packet is transferred to a NAT router. This router maintains a table for mapping packets from the private network into the Internet and back. Each time a machine generates a packet destined for the Internet, a new entry is created in a table in the NAT router. The entry contains the private IP address of the machine as well as the TCP or UDP port number of the packet, say x. Another port number that is not already in use is selected and assigned to the given packet, say y. The NAT router then sends the packet into the Internet with the registered address and the port number y. When the response packet arrives, the port number y is used to retrieve the original private IP address and port number x. The packet can then be delivered to the appropriate machine. We have only considered a single, but important case of network address translation. RFC 1631 addresses various forms of NAT.

8.8.3 Mobile IP

Mobile networking is a subject that is becoming increasingly important as portable devices such as personal digital assistants (PDAs) and notebook computers are becoming more powerful and less expensive, coupled with people's need to be connected whenever and wherever they are. The link between the portable device and the fixed communication network can be wireless or wired. If a wireless link is used, the device can utilize a radio or infrared channel. Of these two alternatives, radio channels can traverse longer distance without the line-of-sight requirement, but introduce electromagnetic interference and are often subject to federal regulations (e.g., Federal Communication Commission, or FCC). Infrared channels are often used in shorter distances. A wireless connection enables a user to maintain its communication session as it roams from one area to another, providing a very powerful communication paradigm. In this section we look at a simple IP solution for mobile computers.

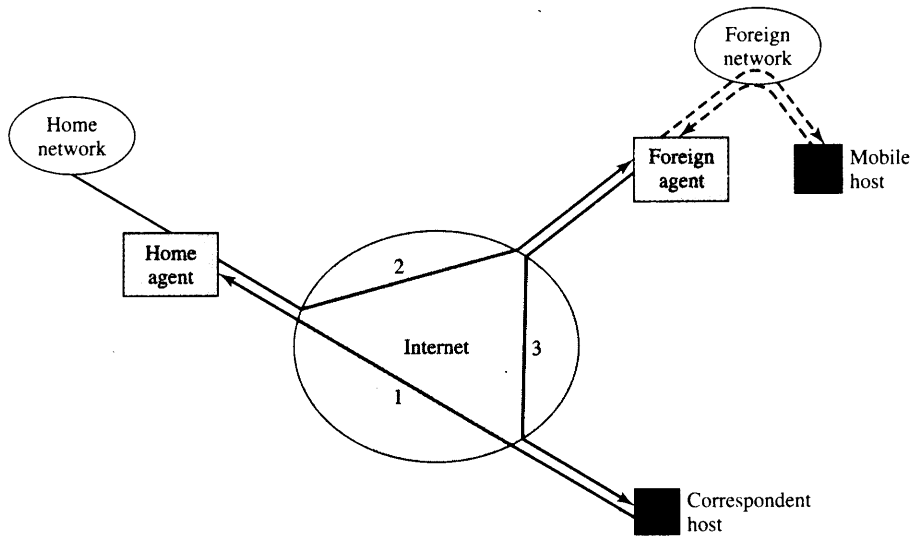


FIGURE 8.66 Routing for mobile hosts.

Mobile IP allows portable devices called **mobile hosts (MHs)** to roam from one area to another while maintaining the communication sessions. One requirement in mobile IP is that a legacy host communicating with an MH and the intermediate routers should not be modified. This requirement implies that an MH must continuously use its permanent IP address even as it roams to another area. Otherwise, existing sessions will stop working and new sessions should be restarted when an MH moves to another area. The basic mobile IP solution is sketched in Figure 8.66.

The mobile IP routing operates as follows:

- When a **correspondent host (CH)** wants to send a packet to an MH, the CH transmits the standard IP packet with its address as the source IP address and the MH's address as the destination IP address. This packet will be intercepted by the mobile host's router called the **home agent (HA)**, which keeps track of the current location of the MH. The HA manages all MHs in its **home network** that use the same address prefix. If the MH is located in the home network, the HA simply forwards the packet to its home network.
- When an MH moves to a **foreign network**, the MH obtains a **care-of address** from the **foreign agent (FA)** and registers the new address with its HA. The care-of address reflects the MH's current location and is typically the address of the FA. Once the HA knows the care-of address of the MH, the HA can forward the registration packet to the MH via the FA.

Unfortunately, the HA cannot directly send packets to the MH in a foreign network in a conventional way (i.e., by using the care-of address as the destination address of the IP packet, the packet final destination will be the FA rather than the MH). The problem is solved by providing a tunnel between the HA and the FA, and implemented by encapsulating each IP packet at the HA with an outer IP header (see Figure 8.67) containing the HA's address as the source IP address and the care-of address as the

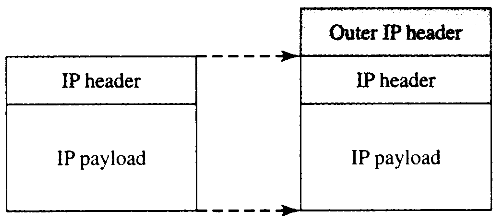


FIGURE 8.67 IP to IP encapsulation.

destination IP address. When the FA receives the packet, the FA decapsulates the packet that produces the original IP packet with the CH's address as the source IP address and the MH's address as the destination IP address. The FA can then deliver the packet to the MH.

Packets transmitted by the MH to the CH typically use a normal IP packet format with the MH's address as the source IP address and the CH's address as the destination IP address. These packets follow the default route.

Observe that the route traveled by the packet from the CH to the MH is typically longer than that from the MH to the CH. For example, it may happen that a CH in New York sends a packet to the HA in Seattle that is tunneled back to New York, since the MH is visiting New York! (This phenomenon is called triangle routing.) Several proposals exist to improve the routing mechanism so that the CH may send packets directly to the care-of address endpoint in subsequent exchanges. One solution is illustrated in Figure 8.68.

When the HA receives a packet from a CH destined to an MH (1), it tunnels the packet to the current care-of address (2a), as before. However, it also sends a **binding message** back to the CH containing the current care-of address (2b). The CH can save this message in its **binding cache** so that future packets to the MH can be directly tunneled to the care-of address (4).

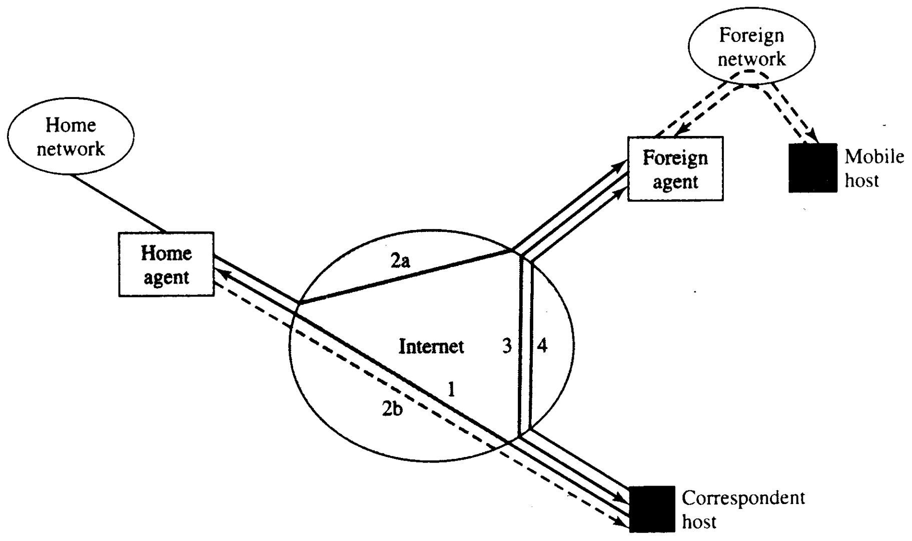


FIGURE 8.68 Route optimization for mobile IP.

SUMMARY

We began this chapter with an end-to-end view of the flow of application layer messages through the IP layer in the end systems and across an IP internetwork. We then examined the details of how IP enables communications across a collection of networks. We paid particular attention to the hierarchical structure of IP addresses and explained their role in ensuring scalability of the Internet. The role of address prefixes and the use of masks was explained. We also examined the interplay between IP addresses, routing tables, and router operation. We discussed how the IP layer must perform segmentation and reassembly in order to provide the higher layers with independence from the underlying network technologies. We also introduced IP version 6 and identified its advantages relative to IPv4 in terms of address space and streamlined header design.

The Internet offers two basic communication services that operate on top of IP: TCP reliable stream service and UDP datagram service. We examined the simple operation of UDP, and we then reviewed how TCP provides reliable stream service and flow control. We also discussed the various issues that underlie the design of TCP: the rationale for using the three-way handshake; the separation of acknowledgments and window size, the difficulties in closing a connection, and the consequences of sequence number space limitations as well as of round-trip time variability. Finally, we discussed the mechanism that TCP uses to perform congestion control.

We then explained the control-plane protocols employed in IP networks. We introduced the autonomous system structure of the Internet and explained how the routing problem is then partitioned into intradomain and interdomain components. We introduced RIP and OSPF for intradomain routing. We identified the advantages of OSPF and explained its operation in detail. We discussed how intradomain and interdomain routing involve different concerns (e.g., optimal path routing versus policy routing). We also explained how BGP4 provides interdomain routing. Finally, we showed how the hierarchy of the autonomous system structure and the hierarchy inherent in CIDR addressing work together to provide an Internet that can scale to enormous size.

In Section 8.8 we provided an introduction to multicast routing and explained a simple way to build a multicast tree using a combination of graft and prune messages. Finally, we introduced DHCP, NAT, and mobile IP and explained their role in extending IP service to mobile and transitory users.

CHECKLIST OF IMPORTANT TERMS

Address Resolution Protocol (ARP)
American Registry for Internet
  Numbers (ARIN)
area
area border router (ABR)
attribute
autonomous system (AS)
backbone area
binding cache
binding message
Border Gateway Protocol (BGP)
care-of address
classless interdomain routing (CIDR)
congestion avoidance
congestion threshold
congestion window

correspondent host (CH)
datagram
designated router
Dijkstra's algorithm
Distance-Vector Multicast Routing Protocol (DVMRP)
dotted-decimal notation
Dynamic Host Configuration Protocol (DHCP)
Exterior Gateway Protocol (EGP)
external BGP (eBGP)
fast retransmit
fast recovery
foreign agent (FA)
foreign network
fragment
graceful close
graft message
Hello protocol
home agent (HA)
home network
host ID
initial sequence number (ISN)
Interior Gateway Protocol (IGP)
internal BGP (iBGP)
Internet Control Message Protocol (ICMP)
Internet Group Management Protocol (IGMP)
IPv6
link state
link-state database
local traffic
longest prefix match
lost/wandering duplicate
maximum segment lifetime (MSL)
maximum segment size (MSS)
maximum transmission unit (MTU)
mobile host (MH)
multicast backbone (MBONE)
multicast routing
Nagle algorithm
network address translation (NAT)
network ID
Open Shortest Path First (OSPF)
packet
parent port
piggyback
prune message
pseudoheader
Reverse Address Resolution Protocol (RARP)
reverse-path broadcasting (RPB)
reverse-path multicasting (RPM)
Routing Information Protocol (RIP)
segment
silly window syndrome
slow start
subnet mask
subnetting
supernetting
three-way handshake
timestamp
TIME_WAIT state
time-to-live (TTL) field
transit traffic
transmission control block (TCB)
Transmission Control Protocol (TCP)
truncated reverse-path broadcasting (TRPB)
tunnel
User Datagram Protocol (UDP)
window scale

FURTHER READING

Comer, D. E., *Internetworking with TCP/IP, Volume 1*, 3rd ed., Prentice Hall, Englewood Cliffs, New Jersey, 1995.

Degermark, M., A. Brodnik, S. Carlsson, and S. Pink, "Small Forwarding Tables for Fast Routing Lookups," in *Designing and Building Gigabit and Terabit Internet Routers*, ACM SigComm 98 tutorial proceedings, September 1998.

Huitema, C., *IPv6, The New Internet Protocol*, Prentice Hall PTR, Upper Saddle River, New Jersey, 1996.

Huitema, C., *Routing in the Internet*, Prentice Hall, Englewood Cliffs, New Jersey, 1995.

Jacobson, V., "Congestion Avoidance and Control," *Computer Communications Review*, Vol. 18, No. 4, pp. 314–329, August 1988.

Moy, J. T., *OPSF: Anatomy of an Internet Routing Protocol*, Addison-Wesley, Reading, Massachusetts, 1998.

Murhammer, M. W., O. Atakan, S. Bretz, L. R. Pugh, K. Suzuki, and D. H. Wood, *TCP/IP Tutorial and Technical Overview*, Prentice Hall PTR, Upper Saddle River, New Jersey, 1998.

Stallings, W., "IPv6: The New Internet Protocol," *IEEE Communications*, Vol. 34, No. 7, pp. 96–108, July 1996.

Stevens, W. R., *TCP/IP Illustrated, Volume 1*, Addison-Wesley, Reading, Massachusetts, 1994.

Stewart, J., *BGP4: Inter-Domain Routing in the Internet*, Addison-Wesley, Reading, Massachusetts, 1999.

Waldvogel, M., G. Varghese, J. Turner, and B. Plattner, "Scalable High Speed IP Routing Lookups," in *Designing and Building Gigabit and Terabit Internet Routers*, ACM SigComm 98 tutorial proceedings, September 1998.

RFC 791, J. Postel (ed.), "Internet Protocol: DARPA Internet Program Protocol Specification," September 1981.

RFC 792, J. Postel (ed.), "Internet Control Message Protocol: DARPA Internet Program Protocol Specification," September 1981.

RFC 793, J. Postel (ed.), "Transmission Control Protocol: DARPA Internet Program Protocol Specification," September 1981.

RFC 1058, C. Hedrick, "Routing Information Protocol," June 1988.

RFC 1122, R. Braden (ed.), "Requirements for Internet Hosts—Communication Layers," October 1989.

RFC 1518, Y. Rekhter and T. Li, "An Architecture for IP Address Allocation with CIDR," September 1993.

RFC 1519, V. Fuller et al., "Classless Inter-Domain Routing (CIDR): An Address Assignment and Aggregation Strategy," September 1993.

RFC 1631, R. Egevang and P. Francis, "The IP Network Address Translator (NAT)," May 1994.

RFC 1771, Y. Rekhter and T. Li, "A Border Gateway Protocol 4 (BGP-4)," May 1995.

RFC 2002, C. Perkins, "IP Mobility Support," October 1996.

RFC 2113, D. Katz, "IP Router Alert Option," February 1997.

RFC 2328, J. Moy, "OSPF Version 2," April 1998.

RFC 2453, G. Malkin, "RIP Version 2," November 1998.

RFC 2460, S. Deering and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification," December 1998.

See our website for additional references available through the Internet.

PROBLEMS

8.1. The IP header checksum only verifies the integrity of IP header. Discuss the pros and cons of doing the checksum on the header part versus on the entire packet.

8.2. Identify the address class of the following IP addresses: 200.58.20.165; 128.167.23.20; 16.196.128.50; 150.156.10.10; 250.10.24.96.

8.3. Convert the IP addresses in Problem 8.2 to their binary representation.

8.4. Identify the range of IPv4 addresses spanned by Class A, Class B, and Class C.

8.5. What are all the possible subnet masks for the Class C address space? List all the subnet masks in dotted-decimal notation, and determine the number of hosts per subnet supported for each subnet mask.

8.6. A host in an organization has an IP address 150.32.64.34 and a subnet mask 255.255.240.0. What is the address of this subnet? What is the range of IP addresses that a host can have on this subnet?

8.7. A university has 150 LANs with 100 hosts in each LAN.
(a) Suppose the university has one Class B address. Design an appropriate subnet addressing scheme.
(b) Design an appropriate CIDR addressing scheme.

8.8. A small organization has a Class C address for seven networks each with 24 hosts. What is an appropriate subnet mask?

8.9. A packet with IP address 150.100.12.55 arrives at router R1 in Figure 8.8. Explain how the packet is delivered to the appropriate host.

8.10. In Figure 8.8 assign a physical layer address 1, 2, ... to each physical interface starting from the top row, moving right to left, and then moving down. Suppose H4 sends an IP packet to H1. Show the sequence of IP packets and Ethernet frames exchanged to accomplish this transfer.

8.11. ARP is used to find the MAC address that corresponds to an IP address; RARP is used to find the IP address that corresponds to a MAC address. True or false?

8.12. Perform CIDR aggregation on the following /24 IP addresses: 128.56.24.0/24; 128.56.25.0/24; 128.56.26.0/24; 128.56.27.0/24.

8.13. Perform CIDR aggregation on the following /24 IP addresses: 200.96.86.0/24; 200.96.87.0/24; 200.96.88.0/24; 200.96.89.0/24.

8.14. The following are estimates of the population of major regions of the world: Africa 900 million; South America 500 million; North America 400 million; East Asia 1500 million; South and Central Asia 2200 million; Russia 200 million; Europe 500 million.
(a) Suppose each region is to be assigned 100 IPv4 addresses per person. Is this possible? If not, how many addresses can be assigned per person. Repeat for IPv6.
(b) Design an appropriate CIDR scheme to provide the addressing in part (a).

8.15. Suppose four major ISPs were to emerge with points of presence in every major region of the world. How should a CIDR scheme treat these ISPs in relation to addressing for each major region?

8.16. Discuss the difficulties with using actual time in the TTL field.

8.17. Look up the `netstat` command in the manual for your system. Use the command to display the routing table in your host. Try the different command options.

8.18. Suppose a router receives an IP packet containing 600 data bytes and has to forward the packet to a network with maximum transmission unit of 200 bytes. Assume that the IP header is 20 bytes long. Show the fragments that the router creates and specify the relevant values in each fragment header (i.e., total length, fragment offset, and more bit).

8.19. Design an algorithm for reassembling fragments of an IP packet at the destination IP.

8.20. Does it make sense to do reassembly at intermediate routers? Explain.

8.21. Describe the implementation issues of IPv4 packet processing.

8.22. Use Ethereal to capture ARP packets to find the MAC addresses in a LAN.

8.23. Abbreviate the following IPv6 addresses:
(a) 0000:0000:0F53:6382:AB00:67DB:BB27:7332
(b) 0000:0000:0000:0000:0000:0000:004D:ABCD
(c) 0000:0000:0000:AF36:7328:0000:87AA:0398
(d) 2819:00AF:0000:0000:0000:0035:0CB2:B271

8.24. What is the efficiency of IPv6 packets that carry 10 ms of 64 kbps voice? Repeat if an IPv6 packets carries 1 frame of 4 Mbps MPEG2 video, assuming 30 frames/second.

8.25. Why does IPv6 allow fragmentation at the source only?

8.26. Assuming the population estimates in Problem 8.14, how many IP addresses does IPv6 provide per capita?

8.27. Suppose that IPv6 is used over a noisy wireless link. What is the effect of not having header error checking?

8.28. Explain how the use of hierarchy enhances scalability in the following aspects of Internet:
(a) Domain name system
(b) IP addressing
(c) OSPF routing
(d) Interdomain routing

8.29. The TCP in station A sends a SYN segment with ISN = 1000 and MSS = 1000 to station B. Station B replies with a SYN segment with ISN = 5000 and MSS = 500. Suppose station A has 10,000 bytes to transfer to B. Assume the link between stations A and B is 8 Mbps and the distance between them is 200 m. Neglect the header overheads to keep the arithmetic simple. Station B has 3000 bytes of buffer available to receive data from A. Sketch the sequence of segment exchanges, including the parameter values in the segment headers, and the state as a function of time at the two stations under the following situations:
(a) Station A sends its first data segment at $t = 0$. Station B has no data to send and sends an ACK segment every other frame.

(b) Station A sends its first data segment at $t = 0$. Station B has 6000 bytes to send and sends its first data segment at $t = 2$ ms.

8.30. Suppose that the TCP in station A sends information to the TCP in station B over a two-hop path. The data link in the first hop operates at a speed of 8 Mbps, and the data link in the second hop operates at a speed of 400 kbps. Station B has a 3-kilobyte buffer to receive information from A, and the application at station B reads information from the receive buffer at a rate of 800 kbps. The TCP in station A sends a SYN segment with ISN = 1000 and MSS = 1000 to station B. Station B replies with a SYN segment with ISN = 5000 and MSS = 500. Suppose station A has 10,000 bytes to transfer to B. Neglect the header overheads to keep the arithmetic simple. Sketch the sequence of segment exchanges, including the parameter values in the segment headers, and the state as a function of time at the two stations. Show the contents of the buffers in the intermediate switch as well as at the source and destination stations.

8.31. Suppose that the delays experienced by TCP segments traversing the network are equally likely to be any value in the interval [50 ms, 75 ms]. (See Equations 5.17 to 5.20.)
(a) Find the mean and standard deviation of the delay.
(b) Most computer languages have a function for generating uniformly distributed random variables. Use this function in a short program to generate random times in the above interval. Also, calculate t_{RTT} and d_{RTT} and compare to part (a).

8.32. Suppose that the advertised window is 1 Mbyte long. If a sequence number is selected at random from the entire sequence number space, what is the probability that the sequence number falls inside the advertised window?

8.33. Explain the relationship between advertised window size, RTT, delay-bandwidth product, and the maximum achievable throughput in TCP.
(a) Plot the maximum achievable throughput versus delay-bandwidth product for an advertised window size of 65,535 bytes.
(b) In the preceding plot include the maximum achievable throughput when the window size is scaled up by a factor of 2^K, where $K = 4, 8, 12$.
(c) Place the following scenarios in the plot obtained in part (b): Ethernet with 1 Gbps and distance 100 meters; 2.4 Gbps and distance of 6000 km; satellite link with speed of 45 Mbps and RTT of 500 ms; 40 Gbps link with distance of 6000 km.

8.34. Consider the three-way handshake in TCP connection setup.
(a) Suppose that an old SYN segment from station A arrives at station B, requesting a TCP connection. Explain how the three-way handshake procedure ensures that the connection is rejected.
(b) Now suppose that an old SYN segment from station A arrives at station B, followed a bit later by an old ACK segment from A to a SYN segment from B. Is this connection request also rejected?

8.35. Suppose that the initial sequence number (ISN) for a TCP connection is selected by taking the 32 low-order bits from a local clock.
(a) Plot the ISN versus time assuming that the clock ticks forward once every $1/R_c$ seconds. Extend the plot so that the sequence numbers wrap around.
(b) To prevent old segments from disrupting a new connection, we forbid sequence numbers that fall in the range corresponding to 2MSL seconds prior to their use as an

ISN. Show the range of forbidden sequence numbers versus time in the plot from part (a).

(c) Suppose that the transmitter sends bytes at an average rate $R > R_c$. Use the plot from part (b) to show what goes wrong.

(d) Now suppose that the connection is long-lived and that bytes are transmitted at a rate R that is much lower than R_c. Use the plot from part (b) to show what goes wrong. What can the transmitter do when it sees that this problem is about to happen?

8.36. Suppose that during the TCP connection closing procedure, a machine that is in the TIME_WAIT state crashes, reboots within MSL seconds and immediately attempts to reestablish the connection, using the same port numbers. Give an example that shows that delayed segments from the previous connections can cause problems. For this reason RFC 793 requires that for MSL seconds after rebooting TCP is not allowed to establish new connections.

8.37. Are there any problems if the server in a TCP connection initiates an active close?

8.38. Use a network analyzer to capture the sequence of packets in a TCP connection. Analyze the contents of the segments that open and close the TCP connection. Estimate the rate at which information is transferred by examining the frame times and the TCP sequence numbers. Do the advertised windows change during the course of the connection?

8.39. Devise an experiment to use a network analyzer to observe the congestion control behavior of TCP. How would you obtain Figure 8.37 empirically? Run the experiment and plot the results.

8.40. A fast typist can do 100 words a minute, and each word has an average of six characters. Demonstrate Nagle's algorithm by showing the sequence of TCP segment exchanges between a client, with input from our fast typist, and a server. Indicate how many characters are contained in each segment sent from the client. Consider the following two cases:

(a) The client and server are in the same LAN and the RTT is 20 ms.

(b) The client and server are connected across a WAN and the RTT is 100 ms.

8.41. *Simultaneous Open.* The TCP state transition diagram allows for the case where the two stations issue a SYN segment at nearly the same time. Draw the sequence of segment exchanges and use Figure 8.36 to show the sequence of states that are followed by the two stations in this case.

8.42. *Simultaneous Close.* The TCP state transition diagram allows for the case where the two stations issue a FIN segment at nearly the same time. Draw the sequence of segment exchanges and use Figure 8.36 to show the sequence of states that are followed by the two stations in this case.

8.43. Suppose that a TCP source (with unlimited amount of information to transmit) begins transmitting onto a link that has 1 Mbps in available bandwidth. Sketch congestion window versus time trajectory. Now suppose that another TCP source (also with unlimited amount of information to transmit) begins transmitting over the same link. Sketch the congestion window versus the time for the initial source.

8.44. What is the maximum width of an RIP network?

8.45. Let's consider the bandwidth consumption of the RIP protocol.
(a) Estimate the number of messages exchanged per unit time by RIP.
(b) Estimate the size of the messages exchanged as a function of the size of the RIP network.
(c) Estimate the bandwidth consumption of an RIP network.

8.46. RIP runs over UDP, OSPF runs over IP, and BGP runs over TCP. Compare the merits of operating a routing protocol over TCP, UDP, IP.

8.47. Compare RIP and OSPF with respect to convergence time and the number of messages exchanged under several trigger conditions, that is, link failure, node failure, and link coming up.

8.48. Consider the OSPF protocol.
(a) Explain how OSPF operates in an autonomous system that has no defined areas.
(b) Explain how the notion of area reduces the amount of routing traffic exchanged.
(c) Is the notion of area related to subnetting? Explain. What happens if all addresses in an area have the same prefix?

8.49. Assume that there are N routers in the network and that every router has m neighbors.
(a) Estimate the amount of memory required to store the information used by the distance-vector routing.
(b) Estimate the amount of memory required to store the information by the link-state algorithm.

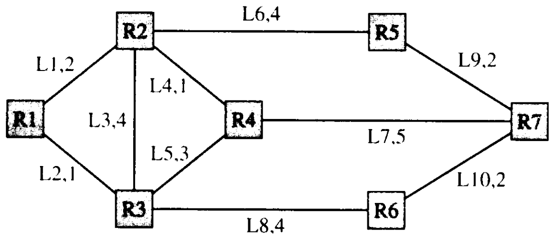
8.50. Suppose a network uses distance-vector routing. What happens if the router sends a distance vector with all 0s?

8.51. Suppose a network uses link-state routing. Explain what happens if:
(a) The router fails to claim a link that is attached to it.
(b) The router claims to have a link that does not exist.

8.52. Consider a broadcast network that has n OSPF routers.
(a) Estimate the number of database exchanges required to synchronize routing databases.
(b) What is the number of database exchanges after a designated router is introduced into the network?
(c) Why is the backup designated router introduced? What is the resulting number of database exchanges?

8.53. Suppose n OSPF routers are connected to a nonbroadcast multiaccess network, for example, ATM.
(a) How many virtual circuits are necessary to provide the required full connectivity?
(b) Does OSPF function correctly if a virtual circuit fails?
(c) Is the number of required virtual circuits reduced if point-to-multipoint virtual circuits are available?

8.54. The figure below shows seven routers connected with links that have the indicated costs. Use the Hello protocol to show how the routers develop the same topology database for the network.



8.55. Consider the exchange of Hello messages in OSPF.

(a) Estimate the number of Hello messages exchanged per unit time.
(b) Estimate the size of the Hello messages.
(c) Estimate the bandwidth consumed by Hello messages.

8.56. Consider the notion of adjacency in OSPF.

(a) Explain why it is essential that all adjacent routers be synchronized.
(b) Explain why it is sufficient that all adjacent routers be synchronized; that is, it is not necessary that all pairs of routers be synchronized.

8.57. Consider the robustness of OSPF.

(a) Explain how the LSA checksum provides robustness in the OSPF protocol.
(b) An OSPF router increments the LS Age each time the router inserts the LSA into a link-state update packet. Explain how this step protects against an LSA that is caught in a loop.
(c) OSPF defines a minimum LS update interval of 5 seconds. Explain why.

8.58. Assume that for OSPF updates occur every 30 minutes, an update packet can carry three LSAs, and each LSA is 36 bytes long. Estimate the bandwidth used in advertising one LSA.

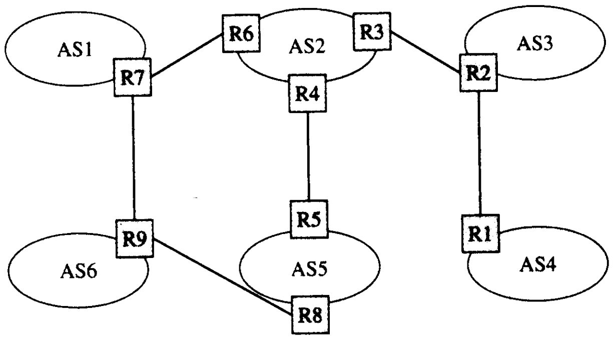
8.59. Identify elements where OSPF and BGP are similar and elements where they differ. Explain the reasons for similarity and difference.

8.60. Discuss the OSPF alternate routing capability for the following cases:

(a) Traffic engineering, that is, the control of traffic flows in the network.
(b) QoS routing, that is, the identification of paths that meet certain QoS requirements.
(c) Cost-sensitive routing, that is, the identification of paths that meet certain price constraints.
(d) Differential security routing, that is, the identification of paths that provide different levels of security.

8.61. Consider the autonomous systems and BGP routers in the following figure.

(a) Suppose that a certain network prefix belongs to AS4. Over which router pairs will the route to the given network be advertised?
(b) Now suppose the link between R1 and R2 fails. Explain how a loop among AS1, AS2, AS5, and AS6 is avoided.



(c) Suppose that R9 is configured to prefer AS1 as transit and AS6 is configured to prefer AS1 as transit. Explain how BGP handles this situation.

8.62. Why does BGP not exchange routing information periodically as RIP does?

8.63. Consider the network shown in Figure 8.60. Suppose that a source connected to router 7 wishes to send information to multicast group G3.
(a) Find the set of paths that are obtained from reverse-path broadcasting.
(b) Repeat for truncated reverse-path broadcasting.
(c) Repeat for reverse-path multicasting.

8.64. Discuss the operation of the reverse-path multicasting in the following two cases:
(a) The membership in the multicast group in the network is dense.
(b) The membership in the multicast group in the network is sparse.

8.65. Suppose an ISP has 1000 customers and that at any time during the busiest hour of the day the probability that a particular user requires service is 20 percent. The ISP uses DHCP. Is a Class C address enough to make the probability less than 1 percent that no IP address is available when a customer places a request?

8.66. Compare mobile IP with the procedures used by cellular telephone networks (Chapter 4) to handle roaming users.
(a) Which cellular network components provide the functions of the home and foreign agent?
(b) Is the handling of mobility affected by whether the transfer service is connectionless or connection oriented?

8.67. Consider a user that can be in several places (home networks) at different times. Suppose that the home networks of a user contain registration servers where users send updates of their location at a given time.
(a) Explain how a client process in a given end system can find the location of a given user to establish a connection, for example, Internet telephone, at a given point in time.
(b) Suppose that proxy servers are available, and their function is to redirect location requests to another server that has more precise location information about the callee. For example, a university might have such a server, which redirects requests for prof@university.edu to departmental servers. Explain how a location request for engineer@home.com might be redirected to a.prof@ece.university.edu.

CHAPTER 9

ATM Networks

In Chapter 7 we saw that asynchronous transfer mode (ATM) was developed to combine the attributes of time-division circuit-switched networks and packet-switched networks. We also saw how ATM provides the capability of providing Quality-of-Service support in a connection-oriented packet network. In this chapter we present the details of ATM network architecture.

The chapter is organized as follows:

1. *Why ATM?* We first provide a historical context and explain the motivation for the development of ATM networks.
2. *BISDN reference model.* We examine the BISDN reference model that forms the basis for ATM, and we explain the role of the user and control planes.
3. *ATM layer.* We examine the "network layer" of the ATM architecture, and we explain the operation of the ATM protocol. Quality of service (QoS) and the ATM network service categories and the associated traffic management mechanisms are introduced.
4. *ATM adaptation layer.* We introduce the various types of ATM adaptation layer protocols that have been developed to support applications over ATM connections.
5. *ATM signaling.* We provide an introduction to ATM addressing and to ATM signaling standards.
6. *PNNI routing.* We briefly describe a dynamic routing protocol for ATM networks, called PNNI.
7. *Classical IP over ATM.* We describe an overlay model used to run IP over ATM networks.

In the next chapter we show how IP and ATM networks can be made to work together. We also show how IP networks are evolving to provide QoS support.

9.1 WHY ATM?

The concept of ATM networks emerged from standardization activities directed at the development of *Integrated Services Digital Networks (ISDNs)*. In the 1970s the trend toward an all-digital (circuit switched) telephone network was clearly established and the need to (eventually) extend digital connectivity to the end user was recognized. It was also apparent that data applications (e.g., computer communications and facsimile) and other nonvoice applications (e.g., videoconferencing) would need to be accommodated by future networks. It was also clear that circuit switching would not be suitable for bursty data traffic and that packet switching would have to be provided. The ISDN standards were the first effort at addressing these needs.

The recommendations adopted by the CCITT (now the telecommunications branch of the International Telecommunications Union) in 1984 defined an *ISDN* as a network that provides *end-to-end digital connectivity* to support a *wide range of services* to users through a limited set of *standard user-network interfaces*. The basic rate interface consisted of two constant bit rate 64 kbps B channels and a 16 kbps D channel. The primary rate interface provided for either 23 B channels and a 64 kbps channel, primarily in North America, or for 30 B channels and a 64 kbps channel elsewhere. The ISDN recommendations provided for the establishment of voice and data connections. The recommendations focused exclusively on the interface between the user and the network and did not address the internal organization of the network. Indeed once inside the network, voice and data traffic would typically be directed to separate circuit-switched and packet-switched networks. Thus the network operator was still burdened with the complex task of operating multiple dissimilar networks.

It soon became clear that higher rate interfaces would be required to handle applications such as the interconnection of high-speed local area networks (LANs) as well as to transfer high-quality digital television. The initial discussions on *broadband ISDN (BISDN)* focused on defining additional interfaces along the lines of established rates in the telephone digital multiplexing hierarchy. However, eventually the discussions led to a radically different approach, known as ATM, that attempts to handle steady stream traffic, bursty data traffic, and everything in between. ATM involves converting all traffic that flows in the network into 53-byte blocks called *cells*. As shown in Figure 9.1, each cell has 48 bytes of *payload* and a 5-byte *header* that allows the network to forward each cell to its destination.

The connection-oriented cell-switching and multiplexing principles underlying ATM were already discussed in Chapter 7. Here we reiterate the anticipated advantages of ATM networks.

1. The network infrastructure and its management is simplified by using a single transfer mode for the network; indeed, extensive bandwidth management capabilities have been built into the ATM architecture.

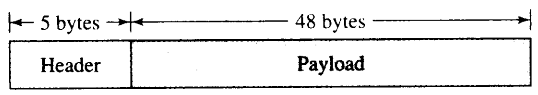


FIGURE 9.1 The ATM cell.

2. Unlike shared-media networks, ATM is not limited by speed or distance; the switched nature of ATM allows it to operate over LANs as well as global backbone networks at speeds ranging from a few Mbps to several Gbps.
3. The QoS attributes of ATM allow it to carry voice, data, and video, thus making ATM suitable for an integrated services network.

The ATM standardization process has taken place under the auspices of the ITU-T in concert with national and regional bodies such as ANSI in the United States and ETSI in Europe. The development of industry implementation agreements has been mainly driven by the ATM Forum.

9.2 BISDN REFERENCE MODEL

The BISDN reference model is shown in Figure 9.2. The model contains three planes: the user plane, the control plane, and the management plane. The *user plane* is concerned with the transfer of user data including flow control and error recovery. The *control plane* deals with the signaling required to set up, manage, and release connections. The *management plane* is split into a layer management plane that is concerned with the management of network resources and a plane management plane that deals with the coordination of the other planes. We focus on the user and control planes.

The user plane has three basic layers that together provide support for user applications: the ATM adaptation layer, the ATM layer, and the physical layer. The **ATM adaptation layer (AAL)** is responsible for providing different applications with the appropriate support, much as the transport layer does in the OSI reference model. Several AAL types have been defined for different classes of user traffic. The AAL is also responsible for the conversion of the higher-layer service data units (SDUs) into

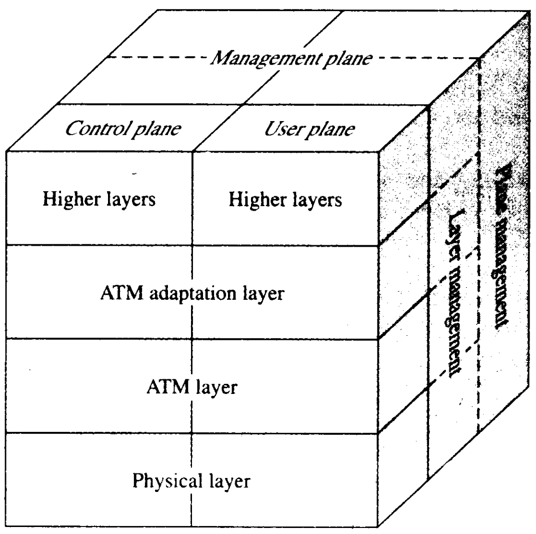


FIGURE 9.2 The broadband ISDN reference model.

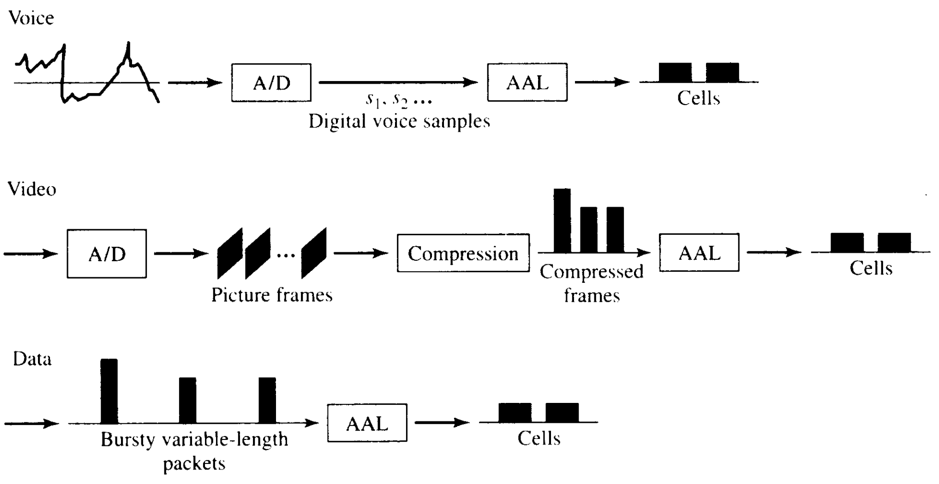
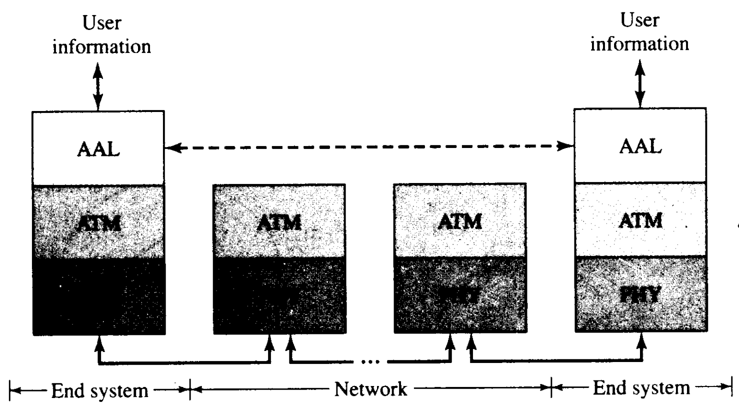


FIGURE 9.3 The AAL converts user information into cells.

48-byte blocks that can be carried inside ATM cells. Figure 9.3 shows how the information generated by voice, data, and video applications are taken by AALs and converted into sequences of cells that can be transported by the ATM network. The AAL entity on the receiver side is responsible for reassembling and delivering the information in a manner that is consistent with the requirements of the given application. Note that the AAL entities reside in the terminal equipment, and hence they communicate on an end-to-end basis across the ATM network as shown in Figure 9.4.

The **ATM layer** is concerned solely with the sequenced transfer of ATM cells in connections set up across the network. The ATM layer accepts 48-byte blocks of information from the AAL and adds a 5-byte header to form the ATM cell. The header contains a label that identifies the connection and that is used by a switch to determine the next hop in the path as well as the type of priority/scheduling that the cell is to receive.



46

FIGURE 9.4 User plane layers.

ATM can provide different QoS to different connections. This requires that a *service contract* be negotiated between the user and the network when the connection is set up. The user is required to describe its traffic and the required QoS when it requests a connection. If the network accepts the request, a contract is established that guarantees the QoS as long as the user complies with its traffic description. Queue priority and scheduling mechanisms implemented in ATM switches provide the capability of delivering QoS. To deliver on its QoS commitments, the ATM network uses policing mechanisms to monitor user compliance with the connection contract and may discard cells not found in compliance.

In terms of number of users involved, ATM supports two types of connections: point to point and point to multipoint. Point-to-point connections can be unidirectional or bidirectional. In the latter case different QoS requirements can be negotiated for each direction. Point-to-multipoint connections are always unidirectional.

In terms of duration, ATM provides permanent virtual connections (PVCs) and switched virtual connections (SVCs). PVCs act as "permanent" leased lines between user sites. PVCs are typically provisioned "manually" by an operator. SVCs are set up and released on demand by the end user.

SVCs are set up through signaling procedures. Initially the source user must interact with the network through a **user-network interface (UNI)** (see Figure 9.5). The connection request must propagate across the network and eventually involve an interaction at the destination UNI. Within a network, switches must interact across the **network-network interface (NNI)** to exchange information. Switches that belong to different public networks communicate across a **broadband intercarrier interface (B-ICI)**. The source and destination end systems as well as all switches along the path

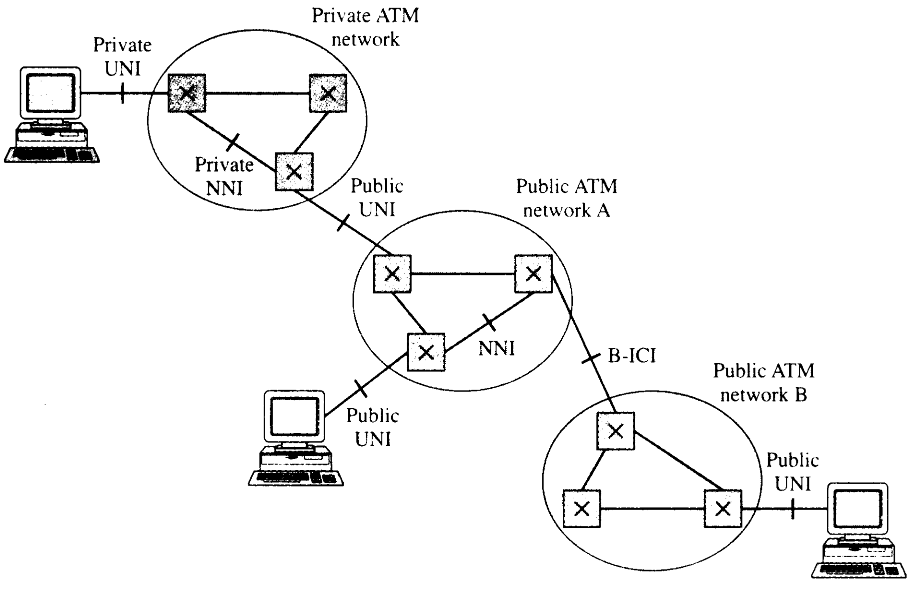


FIGURE 9.5 ATM network interfaces.

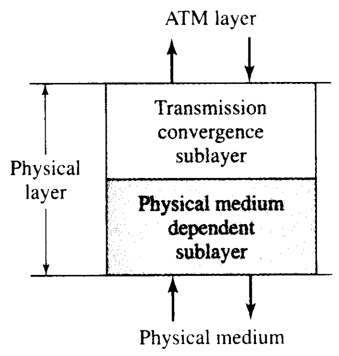

ATM layer
Transmission convergence sublayer
Physical medium dependent sublayer
Physical layer
Physical medium

FIGURE 9.6 ATM physical layer.

across the network are eventually involved in the allocation of resources to meet the QoS requirements of a connection.

Signaling can be viewed as an application in which end systems and switches exchange higher-level messages that establish connections across the network. The role of the **control plane** is to support signaling and network control applications. The control plane has the same three basic layers as the user plane. A *signaling AAL* has been defined for the control plane to provide for the reliable exchange of messages between ATM systems. Higher-layer protocols have been defined for use over the UNI, for the NNI, and for the B-ICI.

The *physical layer* is divided into two sublayers as shown in Figure 9.6. The *physical medium dependent sublayer* is the lower of the two layers and is concerned with details of the transmission of bits over the specific medium, such as line coding, timing recovery, pulse shape, as well as connectors. The *transmission convergence sublayer* establishes and maintains the boundaries of the ATM cells in the bit stream; generates and verifies header checksums; inserts and removes "idle" ATM cells when cells are not available for transmission; and, of course, converts ATM cells into a format appropriate for transmission in the given physical medium.

A large number of physical layers have been defined to provide for support for ATM in a wide range of network scenarios, for example, LAN/WAN and private/public scenarios. The approach in the ATM Forum has been to use and adapt existing physical layer standards as much as possible. In addition to SONET/SDH, physical layers have been defined for 1.5 Mbps (DS-1), 2.0 Mbps (E-1), 45 Mbps (DS-3), 34.4 Mbps (E3), 139 Mbps (E-4), 100 Mbps (FDDI), and 155 Mbps (Fiber Channel). Other physical layers are defined as needed.

9.3 ATM LAYER

The ATM layer is concerned with the sequenced transfer of cells of information across connections established through the network. In this section we examine the operation of the ATM layer in detail. We begin with a description of the ATM cell header. We then discuss how fields in the ATM header are used to identify network connections. This section is followed by a description of the types of ATM network service categories and the traffic management mechanisms required to provide these services. A later section deals with ATM addressing and ATM signaling.

9.3.1 ATM Cell Header

Different ATM cell headers have been defined for use in the UNI and in the NNI. The UNI is the interface point between ATM end users and a private or public ATM switch, or between a private ATM switch and a public carrier ATM network, as shown in Figure 9.5. The NNI is the interface between two nodes (switches) in the same ATM network.

Figure 9.7 shows the 5-byte cell header for the UNI. We first briefly describe the functions of the various fields. We then elaborate on their role in ATM networks.

Generic flow control: The GFC field is 4 bits long and was intended to provide flow control and shared medium access to several terminals at the UNI. It is currently undefined and is set to zero. The GFC field has significance only at the UNI and is not carried end to end across the network. The UNI and NNI cell headers differ in that the GFC field does not appear in the NNI cell header; instead the VPI field is augmented to 12 bits.

Virtual path identifier: The VPI field is 8 bits long, so it allows the definition of up to $2^8 = 256$ virtual paths in a given UNI link. Recall from Figure 7.38 that each virtual path consists of a bundle of virtual channels that are switched as a unit over the sequence of network nodes that correspond to the path.

Virtual channel identifier: The VCI field is 16 bits long, so it allows the definition of up to $2^{16} = 65{,}536$ virtual channels per virtual path. The VIP/VCI field is the

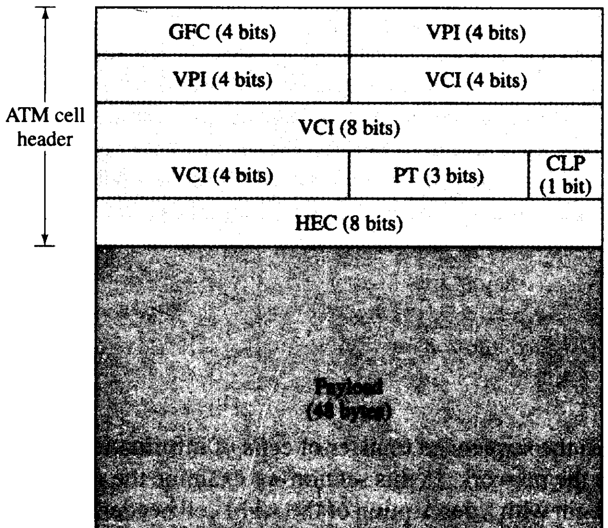


FIGURE 9.7 ATM cell header format.

TABLE 9.1 ATM payload types.

Payload type identifier	Meaning
000	User data cell, congestion not experienced, SDU type = 0 (that is, beginning or continuation of SAR-SDU in AAL5)
001	User data cell, congestion not experienced, SDU type = 1 (that is, end of SAR-SDU in AAL5)
010	User data cell, congestion experienced, SDU type = 0
011	User data cell, congestion experienced, SDU type = 1
100	OAM F5 segment associated cell
101	OAM F5 end-to-end associated cell
110	Resource management (RM) cell (used in traffic management)
111	Reserved for future use

local identifier for a given connection in a given link, and the value of the field changes at every switch.

Payload type: The 3-bit payload type field allows eight types of ATM payloads as shown in Table 9.1. The most significant bit is used to distinguish between data cells ($b_3 = 0$) and operations, administration, and maintenance (OAM) cells ($b_3 = 1$).

For data cells ($b_3 = 0$), the second bit serves as the explicit forward congestion indication (EFCI), which is set by switches to indicate congestion and is used by the congestion control mechanism for the available bit rate (ABR) service defined below.

For data cells ($b_3 = 0$), the least significant bit (b_1) is carried transparently across the network. We show below that $b_1 = 1$ is used by AAL type 5 (AAL5) to signal that a cell carries the end of a SDU.

The payload field (110) defines resource management cells that are used in traffic management.

Cell loss priority: The CLP bit establishes two levels of priorities for ATM cells. A cell that has CLP = 0 is to be treated with higher priority than a cell with CLP = 1 during periods of congestion. In particular, CLP = 1 cells should be discarded before CLP = 0 cells. The CLP bit can be set by terminals to indicate less important traffic or may be set by the network to indicate lower-priority QoS flows or cells that have violated their traffic contract.

Header error control: An 8-bit CRC checksum, using the generator polynomial described in Table 3.7, is calculated over the first four bytes of the header. This code can correct all single errors and detect all double errors in the header. The checksum provides protection against misdelivery of cells from errors that may occur in transit. Two modes are defined. In *detection* mode cells with inconsistent checksums are discarded. In *correction* mode single bit errors are corrected. Correction mode is suitable only in media where single errors predominate over multibit errors. The HEC needs to be recomputed at every switch, since the VPI/VCI value changes at every hop.[1]

[1]The HEC may also be used for cell delineation.

9.3.2 Virtual Connections

In Chapter 7 we described how ATM uses virtual path and virtual channel identifiers in the cell headers to identify a connection across a network. These locally defined identifiers are used to forward cells that arrive at a switch to the appropriate output port. At each switch the VPI/VCI identifier are used to access tables that specify the output port and the VPI/VCI identifier that is to be used in the next hop. In this manner the chain of identifiers define a connection across the network.

The VPI/VCI format allows ATM to switch traffic at two levels. In VP switching, entire bundles of VCs arriving at a given input port and identified by a given VPI are transferred to the same output port. The switch does not look at the VCI value. Prior to transfer along the next hop, the VPI value is mapped into the value that is defined for the next hop. In VP switching, however, the VCI value is not changed. The ability to handle bundles of VCs at a time is very useful to the network operator in facilitating the management of network resources and in simplifying routing topologies.

As indicated above, ATM networks provide two basic types of connections. *Permanent virtual connection (PVCs)* are long-term connections that are typically used by network operators to provision bandwidth between endpoints in an ATM network. *Switched virtual connections (SVCs)* are shorter-term connections that are established in response to customer requests. In SVCs the table entries are established during the call setup procedure that precedes the transfer of ATM cells in a connection.

9.3.3 QoS Parameters

A central objective of ATM is to provide QoS guarantees in the transfer of cell streams across the network. In ATM the QoS provided by the network is specified in terms of the values of several end-to-end, cell-level parameters. A total of six QoS performance parameters have been specified.

The following three QoS network performance parameters are defined in ATM standards. These parameters are not negotiated at the time of connection setup and are indicators of the intrinsic performance of a given network.

Cell error ratio: The *CER* of a connection is the ratio of the number of cells that are delivered with one or more bit errors during a transmission to the total number of transmitted cells. The CER is dependent on the underlying physical medium. The CER calculation excludes blocks of cells that are severely errored (defined below).

Cell misinsertion rate: The *CMR* is the average number of cells/second that are delivered mistakenly to a given connection destination (that is, that originated from the wrong source). The CMR depends primarily on the rate at which undetected header errors result in misdelivered cells. The CMR calculation excludes blocks of cells that are severely errored (defined below). Note that CMR is a rate in cells/second rather than a ratio, since the mechanism that produces misinsertion is independent of the number of cells produced in a connection.

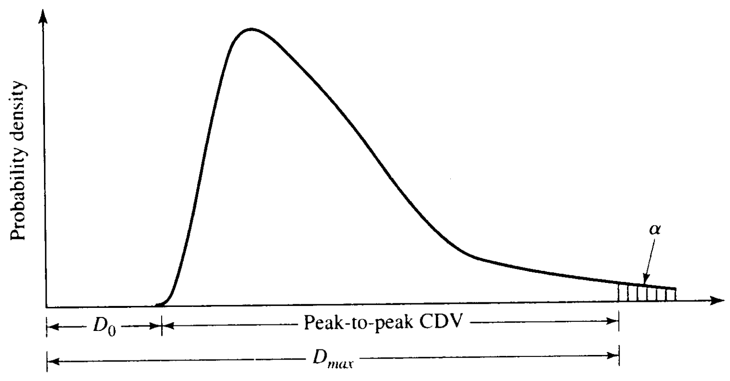


FIGURE 9.8 Probability density function of cell transfer delay.

Severely errored cell block ratio: A severely errored cell block event occurs when more than M cells are lost, in error, or misdelivered in a given received block of N cells, where M and N are defined by the network provider. The severely errored cell block ratio (SECBR) is the ratio of severely errored cell blocks to total number of transmitted cell blocks in a connection. The SECBR is determined by the properties of the error mechanisms of the transmission medium, by buffer overflows, and by operational effects of the underlying transmission system such as losses of information that occur when active transmission links are replaced by backup transmission links in response to faults.

The following three QoS parameters may be negotiated between the user and the network during connection setup.

Cell loss ratio: The *CLR* for a connection is the ratio of the number of lost cells to total number of transmitted cells. The CLR value is negotiated between the user and the network during call setup and specifies the CLR objective for the given connection. It is specified as an order of magnitude in the range of 10^{-1} to 10^{-15}. It can also be left unspecified. The CLR objective can apply to either the CLP = 0 (conforming) cell flow or to the CLP = 0 + 1 (all cells) in the cell flow. The degree to which CLR can be negotiated depends on the sophistication of the buffer-allocation strategies that are available in a given network.

Cell transfer delay: The *CTD* is the time that elapses from the instant when a cell enters the network at the source UNI to the instant when it exists at the destination UNI. The CTD includes propagation delay, processing delays, and queueing delays at multiplexers and switches. In general, different cells in a connection experience different values of delays, so the CTD is specified by a probability density function as shown in Figure 9.8.[2] The standards provide for the negotiation of the "maximum" CTD. As shown in Figure 9.8, the *maximum*

[2]The probability that the delay value falls in an interval is the area under the probability density function for that interval. See [Leon-Garcia 1994].

CTD is defined as the value D_{max} for which the fraction $1 - \alpha$ of all cells have CTD less than D_{max}, where α is some appropriately small value. For example, the requested value of CLR places an upper bound on the value of α. Queue-scheduling algorithms in the ATM switches can be used to control the CTD experienced by cells in a given connection.

Cell delay variation: The *CDV* measures the variablity of the total delay encountered by cells in a connection. The CDV excludes the fixed component D_0 of the CTD that is experienced by all cells in a connection, for example, the propagation delay and fixed processing delays (see Figure 9.8). Current standards provide for the negotiation of the peak-to-peak CDV, which is simply the difference between the maximum CTD D_{max} and the fixed delay component D_0. Note that network switches have only limited control over the spread (variance) of CTD values, and consequently the range of CDV values that can be negotiated for a connection is also limited.

9.3.4 Traffic Descriptors

The capability of a network to provide given levels of QoS to a connection depends on the manner in which the connection produces cells for transmission, that is, at a constant smooth rate or in a highly bursty fashion. This capability also depends on the amount of network resources, that is, bandwidth and buffers, that the network allocates to the connection. Therefore, the connection contract between the user and the network must specify the manner in which the source will produce cells. For this purpose standards have specified a number of *source traffic descriptor* parameters. To be enforceable, policing algorithms must be available to monitor the traffic produced by a source to determine whether it conforms to the connection contract.

The following source traffic parameters have been defined to specify this pattern of demand for transmission.

Peak cell rate: The *PCR* specifies the rate in cells/second that a source is never allowed to exceed. The minimum allowable interval between cells is given by $T = 1/\text{PCR}$.

Sustainable cell rate: The *SCR* is the average cell rate, in cells/second, produced by the source over a long time interval.

Maximum burst size: The *MBS*, in number of cells, specifies the maximum number of consecutive cells that may be transmitted by a source at the peak cell rate (PCR).

Minimum cell rate: The *MCR* is the minimum average cell rate, in cells/second, that the source is always allowed to send.

Even if a source produces cells at exactly the PCR rate, subsequent ATM cell multiplexing and physical layer processing (e.g., insertion of a cell into a bit stream) can produce certain variability about the PCR rate. The policing mechanism must take into account this unavoidable variability. The following interface performance parameter has been defined for this purpose.

Cell delay variation tolerance: The *CDVT* specifies the level of cell delay variation that must be tolerated in a given connection.

9.3.5 ATM Service Categories

ATM connections with arbitrary traffic flow properties and arbitrary QoS are possible by selecting values for the traffic descriptor and the negotiable QoS parameters. In practice there are several clearly identifiable classes of traffic in terms of traffic properties and network QoS requirements. The ATM Forum has defined five *ATM service categories* as shown in Table 9.2. The first two categories apply to connections that are real time in the sense of having stringent delay and timing requirements.

Constant bit rate: The *CBR* ATM service category is intended for traffic with rigorous timing requirements, such as voice, circuit emulation, and certain types of video, that require a constant cell transmission rate for the entire duration of a connection. The traffic rate is specified by the PCR. The QoS is specified by the CTD and CDV, as well as by the CLR.

Real-time variable bit rate: The *rt-VBR* ATM service category is intended for variable-bit-traffic, such as certain types of video, with rigorous timing requirements. The traffic is described by the PCR, SCR, and MBS. The QoS is specified by the CLR, CTD, and CDV.

Three categories of nonreal-time connections have been defined.

TABLE 9.2 ATM service category attributes [ATM April 1996].

	ATM layer service category				
Attribute	**CBR**	**rt-VBR**	**nrt-VBR**	**UBR**	**ABR**
Traffic parameters					
PCR and CDVT[4,5]	Specified	Specified	Specified	Specified[2]	Specified[3]
SCR, MBS, CDVT[4,5]	n/a	Specified	Specified	n/a	n/a
MCR[4]	n/a	n/a	n/a	n/a	Specified
QoS parameters					
peak-to-peak CDVT	Specified	Specified	Unspecified	Unspecified	Unspecified
maxCTD	Specified	Specified	Unspecified	Unspecified	Unspecified
CLR[4]	Specified	Specified	Specified	Unspecified	See note 1
Other attributes					
Feedback	Unspecified	Unspecified	Unspecified	Unspecified	Specified[6]

Notes:
[1]CLR is low for sources that adjust cell flow in response to control information. Whether a quantitative value for CLR is specified is network specific.
[2]May not be subject to connection admission control and usage parameter control procedures.
[3]Represents the maximum rate at which the ABR source may ever send data. The actual rate is subject to the control information.
[4]These parameters are either explicitly or implicitly specified for PVCs or SVCs.
[5]CDVT refers to the cell delay variation tolerance. CDTV is not signaled. In general, CDVT need not have a unique value for a connection. Different values may apply at each interface along the path of a connection.
[6]See discussion on congestion control in Section 7.8.2.

Nonreal-time variable bit rate: The *nrt-VBR* ATM service category addresses bursty sources, such as data transfer, that do not have rigorous timing requirements. The traffic is described by the PCR, SCR, and MBS. The QoS is specified by the CLR, and no delay requirements are specified.

Available bit rate: The *ABR* ATM service category is intended for sources that can dynamically adapt the rate at which they transmit cells in response to feedback from the network. This service allows the sources to exploit the bandwidth that is *available* in the network at a given point in time. The traffic is specified by a PCR and MCR, which can be zero. The sources adjust the rate at which they transmit into the network by implementing a congestion control algorithm that dictates their response to resource management cells that explicitly provide rate flow information. Connections that adapt their traffic in response to the network feedback can expect a low CLR as well as a "fair" share of the available bandwidth.

Unspecified bit rate: The *UBR* ATM service category does not provide *any* QoS guarantees. The PCR may or may not be specified. This service is appropriate for noncritical applications that can tolerate or readily adjust to the loss of cells.

In ATM the QoS guarantees are provided on a per connection basis; that is, every connection can expect that its QoS requirements will be met. Since ATM involves the handling of flows of cells from many connections that necessarily interact at multiplexing points, it is worth considering the nature of the QoS guarantees and the corresponding resource allocation strategies for the different ATM service categories.

For CBR connections the source is free to transmit at the negotiated PCR at any time for any duration. It follows then that the network must allocate sufficient bandwidth to allow the source to transmit continuously at the PCR. The scheduling/priority discipline in the multiplexer must ensure that such bandwidth is regularly available to the connection so that the CDV requirement is also met. The steady flow assumed for CBR sources implies that only limited interaction occurs between different CBR flows. In essence, the individual CBR flows act as if they were in separate, isolated transmission links.

For rt-VBR connections the source transmission rate is expected to vary dynamically around the SCR and below the PCR. Therefore, it is to the benefit of the network operator to statistically multiplex these flows to improve the actual utilization of the bandwidth. However, the mixing of rt-VBR flows must be done in a way that maintains some degree of isolation between flows. In particular it is essential to meet the delay and CLR requirements of the connections.

The situation for nrt-VBR connections is similar to that of rt-VBR connections. Again it is in the interest of the network operator to statistically multiplex nrt-VBR sources. In this case the degree of multiplexing is limited only by the commitment to provide conforming flows with the negotiated CLR.

UBR connections, with their lack of any QoS guarantees, provide an interesting contrast to the preceding service categories. When the traffic levels in the network are low, UBR connections may experience performance that is as good as that of the service categories with QoS guarantees. Only as network traffic levels increase, do the QoS guarantees become noticeable. From the point of view of the network operator, a low tariff for UBR service can be used to stimulate demand for bandwidth when the utilization of network sources is low. This approach is useful to a broad range of users when traffic levels are low; however, it is increasingly less useful as traffic levels increase and the network performance experienced becomes less consistent. The ABR service category fills a niche in this context. UBR connections receive no guarantees as network traffic levels vary. ABR connections, on the other hand, are assured of a low level of CLR as long as they conform by responding to the network feedback information. The option to negotiate a nonzero MCR also provides ABR connections with additional assurance of service.

9.3.6 Traffic Contracts, Connection Admission Control, and Traffic Management

Traffic management refers to the set of control functions that together ensure that connections receive the appropriate level of service. To provide QoS guarantees to each connection, the network must allocate an appropriate set of resources to each new connection. In particular the network must ensure that new VCs are assigned to links that have sufficient available bandwidth and to ports that have sufficient buffers to handle the new as well existing connections at the committed levels of QoS. The queue scheduling algorithms presented in Section 7.7 can be used to ensure that the cells in specific connections receive appropriate levels of performance in terms of delay and loss.

Connection admission control (CAC) is a network function that determines whether a request for a new connection should be accepted or rejected. If accepted, the user and network are said to enter into a *traffic contract*. The contract for a connection includes the ATM service category, the traffic descriptors, and the QoS requirements that were introduced in Section 9.3.5 and shown in Table 9.2. The CAC procedure takes the proposed traffic descriptors and QoS requirements and determines whether sufficient resources are available along a route from the source to the destination to support the new connection as well as already established connections. Specific CAC algorithms are not specified by the standards bodies and are selected by each network operator.

The QoS guarantees are valid only if the user traffic conforms to the connection contract. *Usage parameter control (UPC)* is the process of enforcing the traffic agreement at the UNI. Each connection contract includes a cell conformance definition that specifies how a cell can be policed, that is, determined to be either conforming or nonconforming to the connection contract. The generic cell rate algorithm (GCRA) is equivalent to the leaky-bucket algorithm described in Section 7.8 and can be used to determine whether a cell conforms to the specified PCR and CDVT. Cells that are found to be not conforming are "tagged" by having their cell loss priority (CLP) bit in the header set to 1 at the access to the network. When congestion occurs in the network, cells with CLP $= 1$ are discarded first. Thus nonconforming cells are more likely to be discarded. The GCRA algorithm can be modified so that cell conformance to the PCR, CDVT, as well as the SCR and MBS can be checked.

A connection is said to be *compliant* with its contract if the proportion of nonconforming cells does not exceed a threshold specified by the network operator. As long as a connection remains compliant, the network will provide the QoS specified in the contract. However, if a connection becomes noncompliant, the network may then cease providing the contracted QoS.

Traffic shaping is a mechanism that allows sources to ensure that their traffic conforms to the connection contract. In traffic shaping, a leaky bucket is used to identify nonconforming cells that are then buffered and delayed so that all cells entering the network are conforming. The token bucket mechanism discussed in Chapter 7 is a method for doing traffic shaping.

Congestion can still occur inside the network even if all cells that enter the network conform to their connection contract. Such congestion will occur when many cells from different connections temporarily coincide. The purpose of *congestion control* is to detect the onset of congestion and to activate mechanisms to minimize the impact and duration of congestion. ATM networks employ two types of congestion control. *Selective cell discarding* involves discarding CLP = 1 cells during periods of congestion. The onset of congestion may be defined by a threshold on the contents of buffers in switches and multiplexers. Discarding low-priority cells helps meet the QoS commitments that have been made to the high-priority cells.

A second class of congestion control involves sending *explicit congestion feedback* from the network to the sources. This type of control is appropriate for ABR connections, which by definition involve sources that can adapt their input rate to network feedback. The rate-based flow control algorithm described in Chapter 7 has been recommended for this type of congestion control.

9.4 ATM ADAPTATION LAYER

An application that operates across an ATM network has a choice of the five ATM connection service categories shown in Table 9.2. Every application involves the transfer of one or more blocks or of a stream of information across the network. The ATM service categories provide for the sequenced transfer of cells across the network with a certain delay or loss performance. At the very least a conversion is required from the application data blocks to ATM cells at the source and a conversion back to the application blocks at the destination. One purpose of the ATM adaptation layer is to provide for the mapping between application data blocks to cells.

Applications naturally specify their QoS requirements in terms of their data blocks, not in terms of ATM cells. It is also possible that the service provided by the ATM layer does not meet the requirements of the application. For example, the ATM layer does not provide reliable stream service by itself, since some cell losses can occur. Another purpose of the ATM adaptation layer, then, is to enhance the service provided by the ATM layer to the level required by the application. It should be noted that multiple higher layers may operate over the AAL, for example, HTTP over TCP over AAL. To be more precise, we emphasize that the function of the AAL is to provide support for the layer directly above it. Thus if the layer above the AAL is TCP, then the AAL need not be concerned with providing reliable stream service. On the other hand, the AAL may be called on to provide reliable stream service when such service is not available in the higher layers, as, for example, in signaling applications.

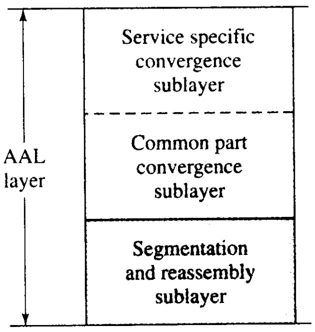

Service specific convergence sublayer
Common part convergence sublayer
Segmentation and reassembly sublayer
AAL layer

FIGURE 9.9 AAL sublayers.

Different applications require a different combination of functions in an AAL. For example, circuit emulation applications require that information be transferred as if the underlying ATM connection were a dedicated digital transmission line. Real-time voice and certain video applications present similar requirements. On the other hand, frame relay applications require the nonreal-time, connection-oriented transfer of a sequence of frames between two end systems. In yet another example, IP routers require the connectionless transfer of a packet to another router, using the ATM network as a "data link." In some cases the IP packets carry payloads that are controlled by TCP entities at the end systems. Each of these examples impose different requirements on the AAL.

There have been several attempts to categorize applications into a small set of classes and to design AALs to meet the requirements of each class. These efforts have not met with success, and no clear correspondence exists between application classes, AALs, and ATM connection service categories. Our approach here is to discuss the formats and services of the AALs that have been developed to date. We then discuss what combinations of applications, AALs, and ATM service categories make sense. It should be noted that users are also free to use proprietary (nonstandard) AALs.

The AAL is divided into two sublayers as shown in Figure 9.9. The purpose of the **segmentation and reassembly (SAR)** sublayer is to segment the PDUs of the higher layer into blocks that are suitable for insertion into the ATM cell payloads at the source and to reassemble the higher-layer PDUs from the sequence of received ATM cell payloads at the destination. The **convergence sublayer (CS)** is divided into a **common part (CPCS)** and a **service-specific part (SSCS)**. The CPCS deals with packet framing and error-detection functions that all AAL users require. The SSCS provides functions that depend on the requirements of specific classes of AAL users. Consequently, each AAL usually has a specific SAR and CPCS sublayer and several optional SSCS sublayers.

9.4.1 AAL1

The ATM adaptation layer type 1 (AAL1) supports services that require the transfer of information at a constant rate. Examples of this type of service are a single 64 kbps PCM voice call, sub-T-1 connections that consist of n $\times$ 64 kbps streams, T-1/E-1 and other digital circuits from the telephone hierarchy, and constant bit-rate digital

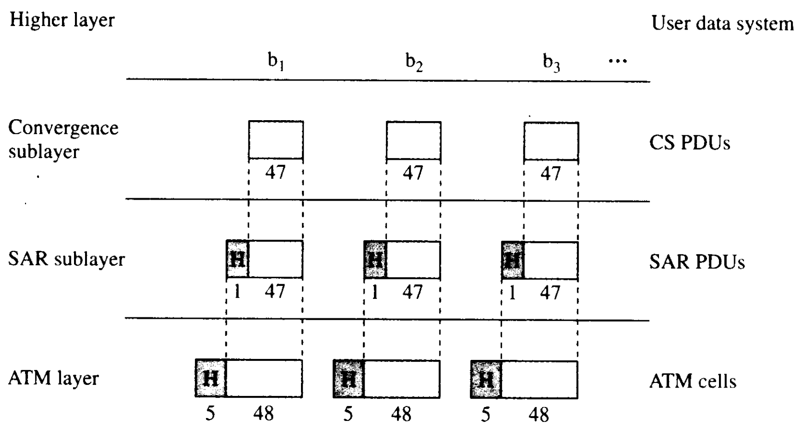


FIGURE 9.10 AAL1 process.

video. The AAL PDU structure contains fields that enable clock recovery and sequence numbering. It also contains an option for the transfer of the internal (frame) structure within a continuous bit stream.

The generic AAL1 process is shown in Figure 9.10. The convergence sublayer function takes the user data stream, optionally inserts a 1-byte pointer to provide structure information, and produces 47-byte CS PDUs, which it then passes with three-bit sequence numbering to the segmentation and reassembly sublayer. Thus the CS PDU normally contains either 47 bytes or 46 bytes of user information, depending on whether a pointer is inserted. Note that the CS PDU need not be completely filled with 47 bytes of user information. In low-bit-rate applications with low-delay requirements, for example, a single 64 kbps voice call, the CS PDU may be required to pass 47-byte blocks that are only partially filled with user information.

Figure 9.11 shows the AAL1 PDUs. The SAR sublayer attaches a 1-byte header to each CS PDU as shown in Figure 9.11a. The first four bits constitute the *sequence number (SN)* field. The first bit in the SN field is the *convergence sublayer indicator (CSI)* and is followed by the three-bit sequence number that can be used for the detection

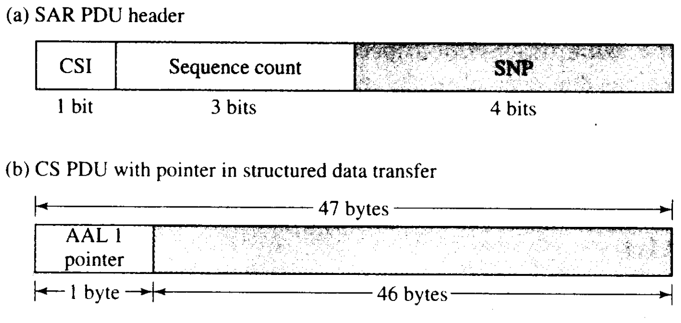


FIGURE 9.11 AAL1 PDUs.

and monitoring of loss and misinsertion of SAR payloads, and hence cells. In even-numbered cells the CSI bit may be used to indicate the existence of a CS sublayer pointer to the destination, as shown in Figure 9.11b. In odd-numbered cells the CSI bit may also optionally be used to convey timing information from the source to the destination. The last four bits of the header contain *sequence number protection (SNP)* check bits that provide error-detection and correction capability for the SAR header. The four check bits are computed by using a Hamming (7,4) code with an additional overall parity check bit. This makes it possible to correct ail single-bit and to detect all double-bit error patterns.

As an example of the use of AAL1 for unstructured information transfer, consider the transfer of a T-1 connection. Recall that the T-1 stream consists of a frame consisting of one framing bit followed by 24 bytes at a repetition rate of 8 kHz. In unstructured AAL1 transfer the bits from the T-1 stream are grouped into blocks of 47 bytes and passed to the SAR. Note that the bytes in the CS PDU will not be aligned to the bytes in the T-1 stream, since each frame has an odd number of bits. Now suppose that the T-1 connection uses *structured data transfer*. The T-1 stream is now viewed as a sequence of 24-byte frames (the framing bit is ignored). The bytes in the T-1 stream are mapped directly into the bytes in the CS PDUs. By prior agreement between the source and destination AAL entities, every so many cells, eight, for example, the first byte of the CS PDU contains a *pointer* that can be used to determine the beginning of the next frame. Because pointers can be inserted only in even-numbered cells, the beginning of the frame can be anywhere in the payloads of the current or the next cell. The periodic insertion of pointers provides protection against loss of synchronization that could result from cell losses.

The convergence sublayer at the destination provides a number of services that are useful to applications requiring a constant transfer rate. The odd-numbered CSI bits can be used to convey a residual timestamp that provides the relative timing between the local clock and a common reference clock. The destination uses these residual timestamps to reconstruct the source clock and replay the received data stream at the correct frequency. This synchronous residual timestamp method is described in Section 5.3.2. It should be noted that other methods for clock recovery, such as adaptive buffer, do not require the use of CSI bits. These methods are also discussed in Chapter 5.

To deliver information to the user at a fixed rate, the convergence sublayer at the destination can carry out timing recovery and then use a playout technique to absorb the cell delay variation in the received sequence. The convergence sublayer can also use the sequence numbers to detect lost or misinserted cells. The capability to request retransmissions is not provided, so the CS can provide only an indication to the user that a loss or misinsertion has occurred.

The convergence sublayer can also implement either of two forward-error correction (FEC) techniques to correct for cell errors and losses. For applications that have a low-delay requirement, the first FEC technique operates on groups of 15 cells and adds sufficient check bits to form 16 cells. The technique can correct one lost cell per group of 16 and can also correct certain other error patterns. The additional FEC delay incurred is then 15 cells. The second FEC technique uses a variation of the interleaving techniques discussed in Chapter 3 to arrange the CS-PDUs of 124 cells as columns in an array. Four additional columns of check bits are added to provide the capability to correct up to four cell losses as well as certain other error patterns. The technique involves incurring a delay of 124 cells at the source and at the destination, so it is generally appropriate only for non-CTD-sensitive traffic, for example, video streaming.

9.4.2 AAL2

AAL type 2 (AAL2) was originally intended to provide support for applications that generate information at a bit rate that varies dynamically with time and that also has end-to-end timing requirements. The prime example of such an application is video that when compressed produces a bit stream that varies widely depending on the degree of detail and the degree of motion in a scene. The development of an AAL for this type of traffic was never completed and the ITU subsequently began work on the development of an AAL, also designated type 2, for a different class of applications. We will discuss this latter AAL in this section.

The new AAL2 is intended for the bandwidth-efficient transfer of low-bit-rate, short-packet traffic that has a low-delay requirement. In effect the AAL2 adds a third level of multiplexing to the VP/VC hierarchy of ATM so that two or more low-bit-rate users can share the same ATM connection. An example where this functionality is required arises in the transfer of compressed voice information from a base station in a digital cellular system to its mobile telephone switching office as shown in Figure 9.12. The low-bit-rate digital streams for individual voice calls need to be transferred in a timely fashion to the switching office where the actual telephone switching takes place. The low delay and low bit rate imply that cell payloads would be only partially filled if each call had its own VC. AAL2 multiplexes the streams from multiple calls to provide both low delay and high utilization of cell payloads.

Figure 9.13 shows the operation of the AAL2. The AAL2 layer is divided into the common part sublayer (CPCS) and the service-specific convergence sublayer (SSCS). In this discussion we focus on the CPCS that transfers CPCS SDUs from a CPCS user at the source to a CPCS user at the destination. The CPCS provides nonassured operation;

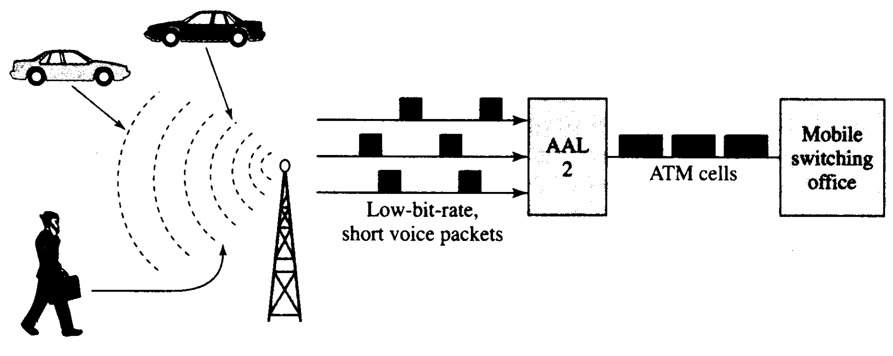


FIGURE 9.12 Application scenario for AAL2.

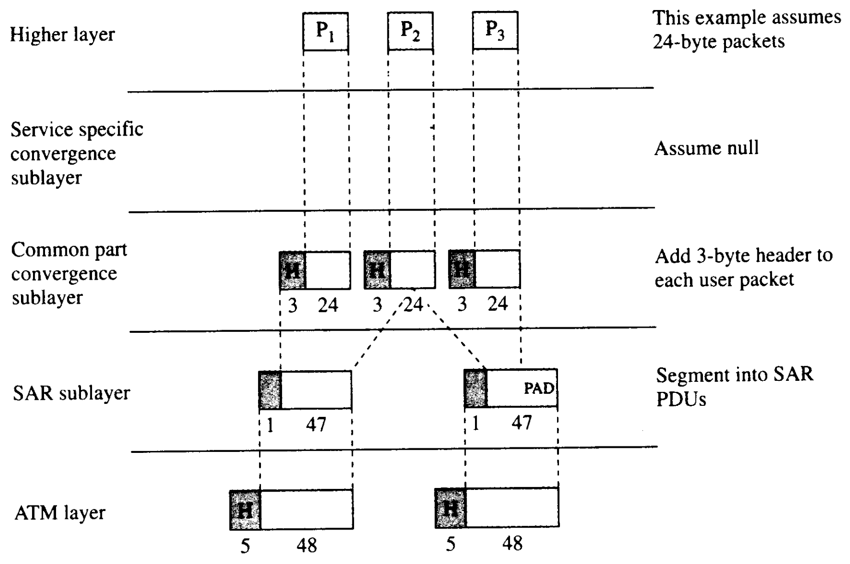

Higher layer
P1
P2
P3
This example assumes 24-byte packets
Service specific convergence sublayer
Assume null
Common part convergence sublayer
H
3
24
Add 3-byte header to each user packet
SAR sublayer
1
47
PAD
Segment into SAR PDUs
ATM layer
5
48

FIGURE 9.13 AAL2 process.

that is, SDUs may be delivered incorrectly or not at all, as a result of cell losses. The CPCS can multiplex SDUs from multiple SSCS users. These users can be of different type and involve different SSCS functions.

User layer packets are transferred to the AAL2 layer as shown in Figure 9.13. These packets can vary in size, since users can be of different type. The maximum allowable packet size is 64 bytes. Assuming that the SSCS is not present, a three-byte header is added to each packet to form a CPCS packet. As shown in Figure 9.14a, the first byte is the channel identifier (CID) that identifies each user. These AAL channels are bidirectional, and the same CID is used for both directions. The six higher-order bits of the next byte are a length indicator that specifies one less than the number of bytes in the CPCS-packet payload. The remaining two bits of the second byte in the header specify the packet payload type (PPT). A value of 3 indicates that the CPCS packet is serving an OAM function. When the value is not 3, the packet is serving the application-specific functions, for example, the transfer of voice. The higher-order three bits of the third header byte are the user-to-user indication (UUI). When the PPT is not 3 the UUI is carried transparently between the SSCS protocol entities. When the PPT is 3, the UUI is carried transparently between the AAL layer management entities. The final five bits of the header are check bits that are used to detect errors. The encoding uses the generator polynomial $g(x) = x^5 + x^2 + 1$.

As shown in Figure 9.13, the CPCS packets are concatenated one after another prior to segmentation into 48-byte ATM SDUs. Each ATM SDU consists of a 1-byte start field and 47 bytes of CPCS packet bytes or padding (see Figure 9.14b). The start field provides the offset from the end of the STF to the start of the first CPCS packet in the SDU or, in the absence of such, the start of the PAD field. The maximum CPCS packet size is 64 bytes, so a CPCS packet may overlap one or two ATM cell boundaries.

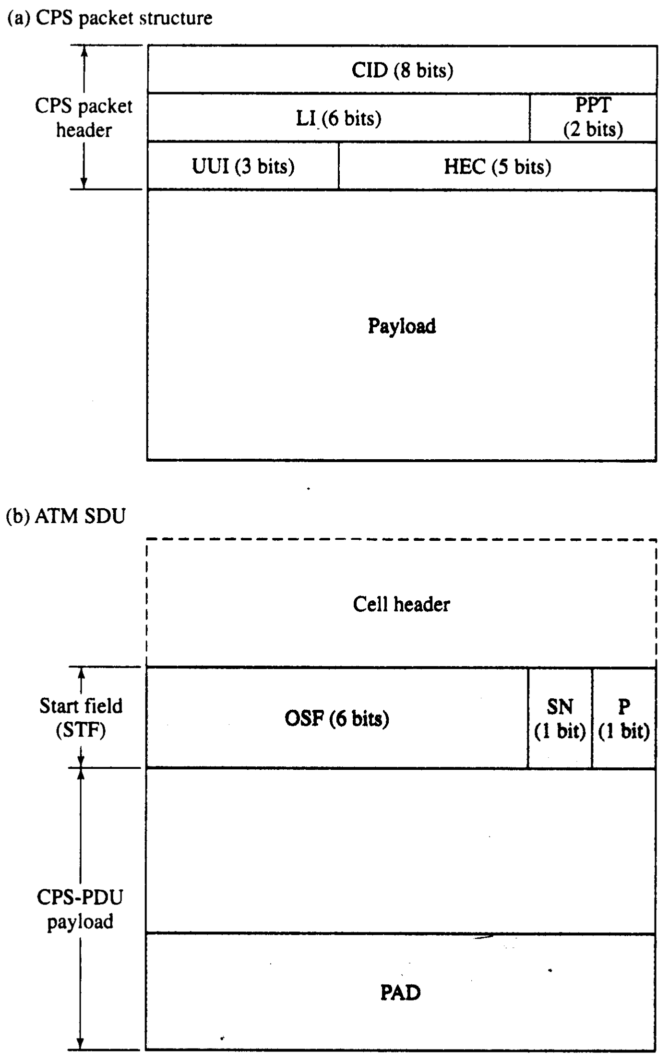


FIGURE 9.14 CPS packets.

To meet the low-delay requirements, it is possible for the payload in the last cell to be only partially filled as shown, for example, in the second cell in Figure 9.13.

9.4.3 AAL3/4

In the early ATM standardization efforts, AAL type 3 (AAL3) was intended for applications that generate bursts of data that need to be transferred in connection-oriented fashion with low loss, but with no delay requirement. AAL type 4 (AAL4) was similarly intended for connectionless transfer of such data. By convention *all* connectionless packets at the UNI use the *same* VPI/VCI number, so a multiplexing ID was introduced

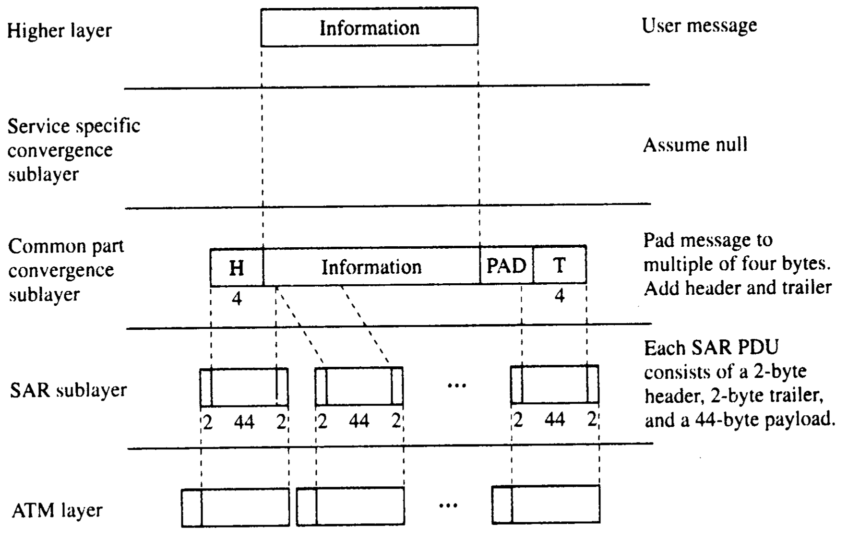

Higher layer
Information
User message
Service specific convergence sublayer
Assume null
Common part convergence sublayer
H
Information
PAD
T
4
4
Pad message to multiple of four bytes. Add header and trailer
SAR sublayer
2 44 2
2 44 2
...
2 44 2
Each SAR PDU consists of a 2-byte header, 2-byte trailer, and a 44-byte payload.
ATM layer
...

FIGURE 9.15 AAL3/4 process.

in AAL4 to distinguish different packets. The efforts to develop AAL3 and AAL4 were later combined to produce AAL3/4, which can be used for either connection-oriented or connectionless transfer. The distinguishing feature of AAL3/4 is that it allows long messages from multiple users to be simultaneously multiplexed and interleaved in the same ATM VC.[3]

AAL3/4 operates in two modes: message mode and stream mode. In message mode the AAL accepts a single user message for segmentation into ATM payloads, and the destination delivers the message. In stream mode one or more user PDUs, each as small as a single byte, are accepted one at a time by the AAL and are subsequently delivered to the destination without an indication of the boundaries between the original PDUs. Both modes allow for assured or nonassured operation. In assured operation an end-to-end protocol is implemented in the SSCS to allow for the error-free delivery of messages. In nonassured operation, messages may be delivered in error or not at all.

The AAL3/4 process in message mode is shown in Figure 9.15. The user information is first passed to the SSCS and then to the CPCS, which adds fill bytes to make the CPCS payload a multiple of four bytes (32 bits). The CPCS PDU is formed by adding four bytes of header and four bytes of trailer. The CPCS PDU is passed to the SAR sublayer, which produces SAR PDUs with 4 bytes of overhead and 44 bytes of payload. If necessary, the last SAR PDU is padded to fill the payload. Finally the 48-byte SAR PDUs are passed to the ATM layer.

The CPCS PDU has a header, followed by the CPCS-PDU payload, possibly with padding, and finally a trailer. The CPCS-PDU payload can have a length of 1 to 65,535 bytes. As shown in Figure 9.16a, the header begins with a one-byte common part indicator (CPI) field that specifies how subsequent fields are to be interpreted. Only CPI = 0 has been defined to indicate that the BAsize and Length fields are to be

[3]See Goralski for a discussion on the merging of AAL3 and AAL4.

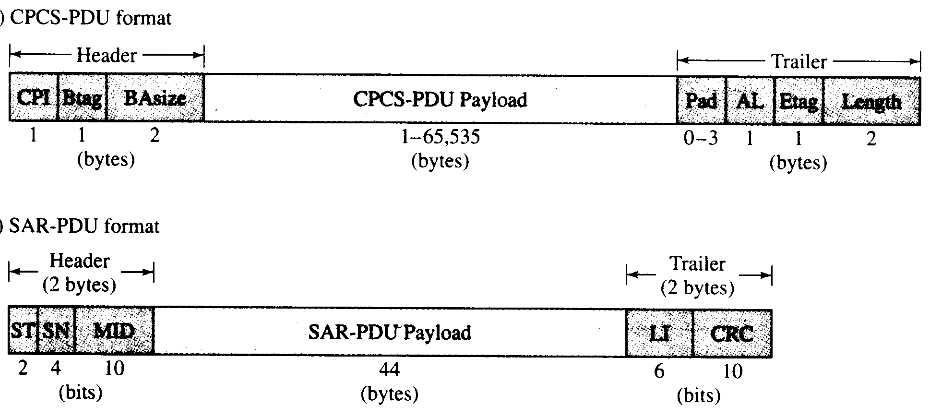


FIGURE 9.16 AAL3/4 CPCS and SAR formats.

interpreted in units of bytes. The one-byte beginning tag (Btag) field and the end tag (Etag) field are set to the same value at the source and changed for each successive CPCS PDU. This practice allows the destination to detect when the incorrect header and trailer have been associated. The two-byte buffer allocation size indication (BAsize) field informs the destination of the maximum buffer size required to receive the current CPCS PDU. The padding field contains zero to three pad bytes to make the CPCS PDU a multiple of four bytes. This approach ensures that the trailer part will be aligned to a 32-bit boundary, making it easier to process the trailer. The alignment field consists of a byte of zeros to make the trailer four bytes long. The two-byte length field indicates the length of the payload.

Figure 9.16b shows that the SAR PDU contains a 2-byte header, 44 bytes of payload, and a 2-byte trailer. The first two bits in the header are the segment type: the value 10 indicates that the PDU contains the beginning of a message (BOM), 00 denotes continuation of message (COM), 01 indicates end of message (EOM), and 11 indicates a single-segment message (SSM). The next four bits provide the sequence number (SN) of the SAR PDU within the same CPCS PDU. The sequence numbers are used to ensure correct sequencing of cells when the CPCS PDU is rebuilt at the destination. The remaining 10 bits in the header are for the **multiplexing identifier**, also called **message identifier, (MID)**. The MID allows the SAR sublayer to multiplex the messages of up to 2^{10} AAL users on a single ATM VC. All SAR PDUs of the same CPCS PDU have the same MID. The six-bit length indicator (LI) in the trailer specifies the size of the payload. Except for the last cell, all cells for a given CPCS PDU are full, so LI = 44. The last cell can have LI from 4 to 44. The 10-bit CRC provides for the detection of errors that may occur anywhere in the PDU.

Figure 9.17 elaborates on how multiplexing is done in AAL3/4. When multiple users share the same VC, the messages from each user are sent to a *different* instance of the AAL. Each such AAL will produce one or more SAR PDUs that are then intermixed and interleaved prior to being passed to the ATM layer. Thus the SAR PDUs from different users arrive intermixed at the destination. The MID allows the messages from each user to be reassembled. Note that the MID can be viewed as setting up a very

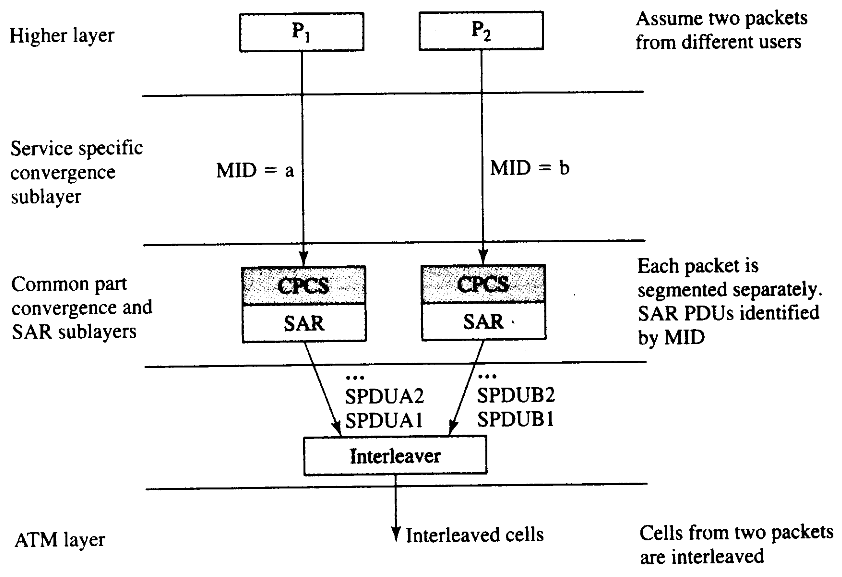

Higher layer
P1
P2
Assume two packets from different users
Service specific convergence sublayer
MID = a
MID = b
Common part convergence and SAR sublayers
CPCS
SAR
CPCS
SAR
Each packet is segmented separately. SAR PDUs identified by MID
SPDUA2
SPDUA1
SPDUB2
SPDUB1
Interleaver
ATM layer
Interleaved cells
Cells from two packets are interleaved

FIGURE 9.17 Multiplexing in AAL3/4.

short term connection within the VC. Each such connection is for the duration of a single packet transfer and is delimited by the BOM and EOM cells. The MID feature of AAL3/4 was developed to provide compatibility with the IEEE 802.6 Metropolitan Area Standard, which provides connectionless LAN interconnection service.

A problem with AAL3/4 is that it is heavy in terms of overhead. Each message has at least eight bytes added at the CSCP sublayer, and subsequently each ATM cell payload includes four additional bytes of overhead. The 10-bit CRC and the 4-bit sequence numbering also may not provide enough protection. These factors led to the development of AAL5.

9.4.4 AAL5

AAL type 5 (AAL5) provides an efficient alternative to AAL3/4. AAL5 forgoes the multiplexing capability of AAL3/4 but does support message and stream modes, as well as assured and nonassured delivery.

Figure 9.18 shows the operation of AAL5. A user PDU is accepted by the AAL layer and is processed by the SSCS if necessary. The SSCS then passes a block of data to the CPCS, which attaches 0 to 47 bytes of padding and an 8-byte trailer to produce a CPCS PDU that is a multiple of 48 bytes. The maximum CPCS-PDU payload is 65,535 bytes.

As shown in Figure 9.19, the trailer contains one byte of UU that is passed transparently between the end-system entities, one byte of CPI that aligns the trailer to eight bytes, a two-byte LI that specifies the number of bytes of user data in the CPCS-PDU payload, and a four-byte CRC check to detect errors in the PDU. The SAR sublayer

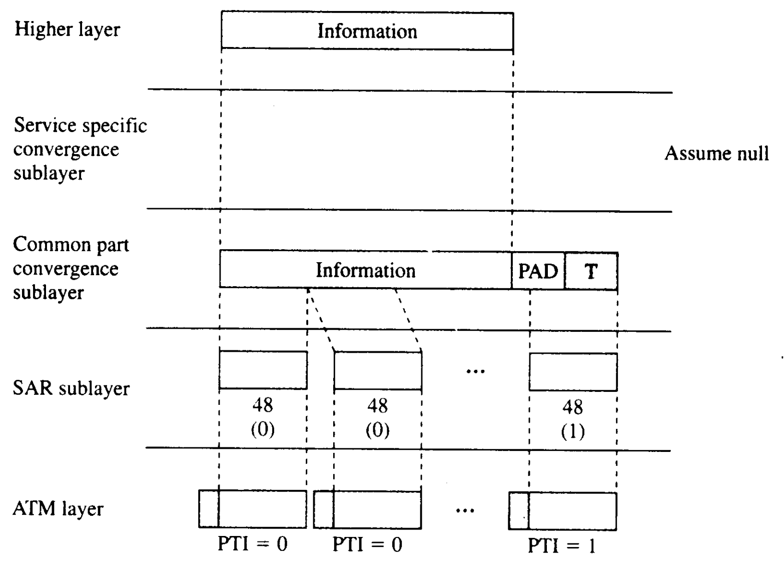


FIGURE 9.18 AAL5 process.

segments the CPCS PDU into 48-byte payloads that are passed to the ATM layer. The SAR also directs the ATM layer to set the PTI field in the header of the last cell of a CPCS PDU. This step allows the boundary between groups of cells corresponding to different messages to be distinguished at the destination. Note that unlike AAL3/4, AAL5 can have only one packet at a time in a VC because the cells from different packets cannot be intermixed.

AAL5 is much more efficient than AAL3/4, as AAL5 does not add any overhead in the SAR sublayer. AAL5 does not include sequence numbers for the SAR PDUs and instead relies on its more powerful CRC checksum to detect lost, misinserted, or out-of-sequence cells. AAL5 is by far the most widely implemented AAL.

9.4.5 Signaling AAL

The signaling AAL (SAAL) has been standardized as the AAL in the control plane. The SAAL provides reliable transport for the signaling messages that are exchanged among end systems and switches to set up ATM VCs.

The SAAL is divided into a common part and a service-specific part as shown in Figure 9.20. The service-specific part in turn is divided into a service-specific connection-oriented protocol (SSCOP) and a service-specific coordination function (SSCF). The SSCF supports the signaling applications above it by mapping the services they require

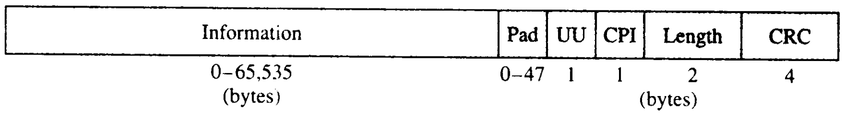


FIGURE 9.19 AAL5 PDU.

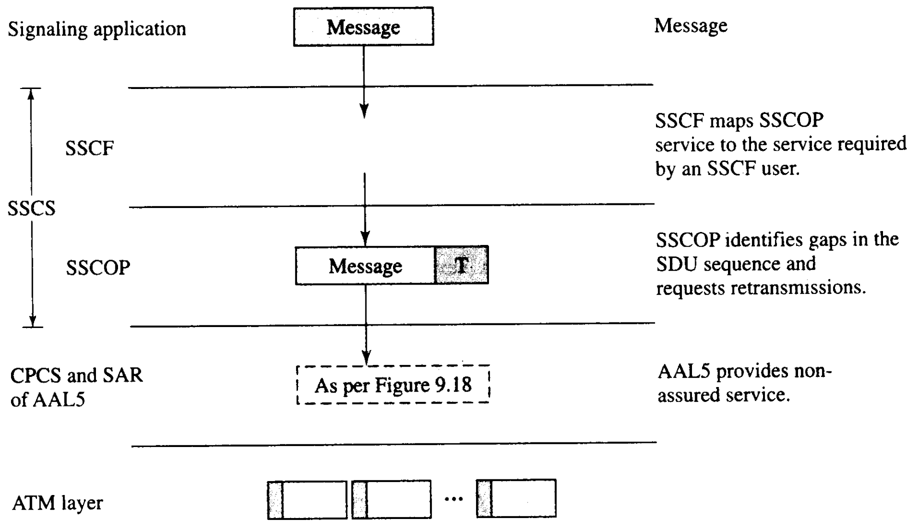

Signaling application
Message
Message
SSCF
SSCF maps SSCOP service to the service required by an SSCF user.
SSCS
SSCOP
Message
T
SSCOP identifies gaps in the SDU sequence and requests retransmissions.
CPCS and SAR of AAL5
As per Figure 9.18
AAL5 provides non-assured service.
ATM layer

FIGURE 9.20 SAAL process.

into the services provided by the SSCOP. SSCF sublayers have been developed for UNI and NNI.

The SSCOP is a peer-to-peer protocol that provides for the reliable transfer of messages. It provides for the ordered delivery of messages in either assured or unassured mode. In assured mode SSCOP uses a form of Selective Repeat ARQ for error recovery. The ARQ protocol is suitable for use in situations that have large delay-bandwidth products, for example, satellite channels and ATM links. Thus the protocol uses large window sizes to provide sequence numbers for the transmitted packets. To achieve bandwidth efficiency, selective retransmission is used. SSCOP makes use of the service provided by the convergence sublayer and SAR sublayers of AAL5 as shown in Figure 9.20. The destination AAL5 layer passes SDUs up to the SSCOP layer only if their checksum is correct. The SSCOP buffers all such SDUs and looks for gaps in the SDU sequence. Because ATM is connection-oriented, the SSCOP knows that the SDUs corresponding to these gaps have errors or have been lost. The sender periodically polls the receiver to find the status of the receive window. Based on the response from the receiver, the transmitter then selectively retransmits the appropriate SDUs. This approach allows SSCOP to ensure that SDUs are retransmitted and that messages are delivered in the correct order and without errors.

Figure 9.21 shows the structure of the SSCOP PDU. A trailer is added, consisting of zero to three bytes of padding, a two-bit pad length indicator, a two-bit reserved (unassigned) field, and a four-bit PDU type field to specify the type of message in the payload. The payload types include sequenced data messages as well as the poll and

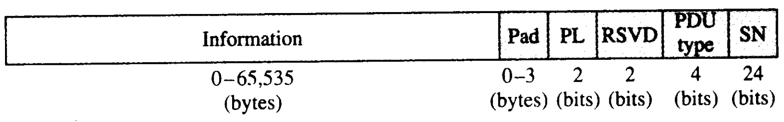


FIGURE 9.21 SSCOP PDU.

TABLE 9.3 Features that characterize application requirements.

Feature	Application Requirements	
Transfer granularity	Stream	Message
Bit rate	Constant	Variable
Reliability	Nonassured	Assured
Accuracy	Error tolerant	Error intolerant
Delay sensitivity	Delay/jitter sensitive	Delay/jitter insensitive
Multiplexing	Single user	Multiple users
Payload efficiency	Bandwidth inexpensive	Bandwidth expensive

other messages used by the SSCOP protocol. A 24-bit sequence number is provided for each CPCS PDU. Note that the cell error and loss detection capabilities are provided by the 32-bit CRC that is used in the AAL5 layers, as per Figure 9.19.

9.4.6 Applications, AALs, and ATM Service Categories

We saw in the previous section that AALs can be called upon to support a wide range of applications, from emulating a digital transmission line to transferring packet streams of various types. Table 9.3 lists features that characterize the requirements of various types of applications. Table 9.4 summarizes the capabilities of the various AALs.

The application/higher layer that operates over the AAL may require that information be transferred in the form of a stream or as discrete messages. The transfer may involve a constant rate of transfer or may vary with time. The application may be satisfied with a nonassured delivery of information, where there may be occasional losses or misdeliveries, or it may require high levels of assurance on the correct delivery

TABLE 9.4 Capabilities of the AAL types.

Sublayer	Feature	AAL1	AAL2	AAL3/4	AAL5	SAAL
SSCS	*Forward error control*	Optional	Optional	Optional	Optional	No
	Error detection and retransmission	No	No	Optional	Optional	SSCOP
	Timing recovery	Optional	Optional	No	Optional	No
CPCS	*Multiplexing*	No	8-bit CID	10-bit MID	No	No
	Framing structure	Yes	No	No	No	No
	Message delimiting	No	Yes	Yes	PTI	PTI
	Advance buffer allocation	No	No	Yes	No	No
	User-to-user indication	No	3 bits	No	1 byte	No
	Overhead	0	3 bytes	8 bytes	8 bytes	4 bytes
	padding	0	0	4 bytes	0–44 bytes	0–47 bytes
	Checksum	No	No	No	32 bit	32 bit
	Sequence numbers	No	No	No	No	24 bit
SAR	*Payload/overhead*	46–47 bytes	47 bytes	44 bytes	48 bytes	48 bytes
	Overhead	1–2 bytes	1 byte	4 bytes	0	0
	Checksum	No	No	10 bits	No	No
	Timing information	Optional	No	No	No	No
	Sequence numbers	3 bit	1 bit	4 bit	No	No

of all information. In the case of nonassured delivery, there may be various degrees of tolerance to errors in the delivered information. Some applications may be tolerant to relatively high delays and significant levels of delay variation, whereas other applications may require tight tolerance in the delay and the jitter. The ability to multiplex streams/packets from different users is an important requirement in certain settings. Finally, the demand for efficiency in the use of the cell payload will depend on the cost of bandwidth in a given setting.

Various combinations of the features in Table 9.3 appear in different applications. A simple example such as voice can require different combinations of features in different contexts. Voice-based applications are almost always stream oriented. In many situations these applications are error tolerant. However, the degree of error tolerance is different when the stream is carrying voice than when it is carrying modem signals that carry data or fax. Furthermore, as the bit rate of voice signals decreases with compression, the degree of error tolerance also decreases. If silence suppression is used, the stream becomes variable bit rate rather than constant bit rate. Telephone conversations between humans requires real-time transfer with low delay and jitter. But voice mail and voice response applications are much more tolerant of delay. Finally, multiplexing may be important in voice applications where bandwidth costs are significant, but not relevant where bandwidth is cheap as in LANs.

Given the diversity of requirements in different voice applications, it is not surprising that many of the AALs have been adapted for use with different voice applications. A simple AAL1 has been adapted for the transfer of individual 64 kbps voice calls. Other versions of AAL1 with structured data transfer (SDT) are intended for handling multiples of 64 kbps calls. In addition, AAL2 provides the capability to multiplex several low-bit-rate voice calls. AAL5 has also been adapted to carry voice traffic. The rationale here was that AAL5 was required for signaling, so the cost of adding AAL1 could be avoided by operating voice over AAL5.

The choice of which ATM service category to use below the AAL depends on the performance requirements the AAL is committed to deliver to the service above it. It also depends on the manner in which the user level passes SDUs to the AAL because this feature influences the manner in which the cell traffic is generated. Thus the CBR and the rt-VBR service categories are suitable for applications with a real-time requirement, for example, voice over AAL5 over CBR. However, under certain network loading conditions, the other service categories may provide adequate performance. For example, under low network loads voice over AAL5 over UBR may give adequate performance. Policing may also be required to ensure that the cell stream produced by the AAL conforms to the connection contract.

Data transfer applications require different combinations of features in the AAL. These applications tend to be message oriented and variable in bit rate. Best-effort (nonassured) service is usually sufficient, but certain applications can require assured service. Most data transfer applications are relatively insensitive to delays, but it is possible to conceive of monitoring/control applications that require very low delay. As indicated above, AAL5 with nonassured service is the most widely deployed AAL. AAL5 with SSCOP has also been used to provide assured service in the user plane.

Video applications represent a third broad class of applications that include relatively low-bit-rate videoconferencing, for example, n $\times$ 64 kbps, to high-quality video distribution and video on demand. Delay plays an important part in applications that involve real-time interactivity. Cell-delay variation plays an important role in almost all video applications because the receiver must compensate for the CDV and present the data to the video decoder at very nearly constant rate. This stringent CDV requirement arises from the manner in which traditional analog television signals deal with the three color components. Very small errors in synchronization among the color components have a dramatic impact on the picture quality.

As a concrete example, consider the transport of constant-bit-rate MPEG2 video in a video-on-demand application. An issue in the design of such a system is whether the timing recovery should be done by the AAL, by the MPEG2 systems layer,[4] or possibly by a layer in between. Another issue involves the degree of error detection and correction required and the use of error concealment techniques. Efficiency in the use of cell payload is also a concern. The recommendation developed by the ATM Forum addressed the issues as follows. AAL1 was not selected for a number of reasons. The SRTS capability of AAL1 could not be used because many end systems do not have access to the common network clock required by the method. The FEC interleaving technique was deemed to introduce too much delay and to be too costly. On the other hand, AAL5 is less costly because of its much wider deployment. It was also found that AAL5 over CBR ATM service could carry MPEG2 transport packets (of 188 bytes each) with sufficiently low jitter that timing recovery could be done in the MPEG2 systems layer. The recommendation does not require that the CBR ATM stream coming out of the video server be shaped to conform to the negotiated CBR parameters.

9.5 ATM SIGNALING

The utility of a network is directly related to the capability to connect dynamically to any number of destinations. Signaling provides the means for dynamically setting up and releasing switched virtual connections in an ATM network. The establishment of an end-to-end connection involves the exchange of signaling messages across a number of interfaces, for example, user-network interface (UNI), network-network interface (NNI), and broadband intercarrier interface (B-ICI). Signaling standards are required for each of these interfaces. In this section we focus on the signaling standards for UNI and NNI.

The establishment of dynamic connections requires the ability to identify endpoints that are attached in the network. This function is provided by network addresses, the topic of the next section.

9.5.1 ATM Addressing

ATM uses two basic types of addresses: telephony-oriented E-164 addresses intended for use in public networks and ATM end system addresses (AESAs) intended for use in private networks. E-164 telephone numbers can be variable in length and have a maximum length of 15 digits. For example in the United States and Canada, 11-digit

[4]MPEG video coding is discussed in Chapter 12.

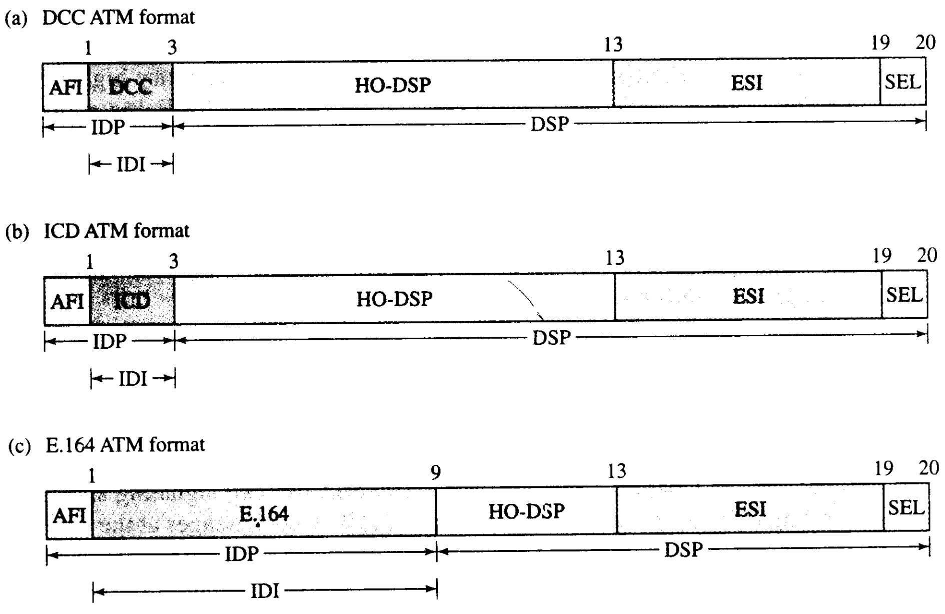


FIGURE 9.22 ATM formats.

numbers are used: 1-NPA-NXX-ABCD, for example, 1-416-555-1212. The first 1 is the ITU assigned country code; the next three digits, NPA, are the area code; the following three digits, NXX, are the office code; and the final four digits, ABCD, are the subscriber number. Telephone numbers for other countries begin with a different country code and then follow a different format.

AESAs are based on the ISO Network Service Access Point (NSAP) format that consists of 20-byte addresses. The NSAP format has a hierarchical structure as shown in Figure 9.22. Each address has two parts.

- The *initial domain part (IDP)* identifies the administrative authority that is responsible for allocating the addresses in the domain-specific part.
- The *domain-specific part (DSP)* contains the address allocated by the given authority.

The IDP itself consists of two parts: a one-byte authority and format identifier (AFI) identifies which structure is to follow and the initial domain identifier (IDI) specifies the authority that allocates the DSP that follows.

Figure 9.22 shows the format of the three initial types of AESAs: data country code (DCC) with AFI = 39_{HEX}; international code designator (ICD) with AFI = 47_{HEX}; and E.164 (contained within the AESA format) with AFI = 45_{HEX}. From Figure 9.22a, it can be seen that the IDI in the DCC format is 2 bytes long, followed by a 10 byte "higher-order domain specific part (HO-DSP)." For example, IDI = 840_{HEX} identifies the United States, for which ANSI administers the DCC addresses. The format used by ANSI is as follows: the first three bytes identify the organization that uses these addresses, the next byte is used for other purposes, and the final six bytes are for the organization to assign. The ICD addresses are administered by the British Standards Institute. The ICD format, shown in Figure 9.22b, places a four-digit code in the IDI bytes to identify an organization, which is then allowed to administer the next 10 bytes. The six-byte end system identifier (ESI) identifies an end system and usually consists of a MAC address. The one-byte selector (SEL) can be used by the end system to identify higher-layer protocol entities that are to receive the traffic. The AESA format for E-164 addresses is shown in Figure 9.22c. The IDI consists of eight bytes that can hold the 15 digits of the E-164 address.

E-164 "native" addresses are supported in the public UNI and in the B-ICI. AESAs are supported in the private UNI and NNI, as well as in the public UNI. The ATM Forum and other bodies are working on the interworking of these public and private addresses to provide end-to-end connectivity.

9.5.2 UNI Signaling

The ATM signaling standards are based on the standards developed for telephone networks. We saw in Chapter 4 that telephone networks use two signaling standards: ISDN signaling (Q.931) is used in the exchange of call setup messages at the UNI; the ISUP protocol of Signaling System #7 is used to establish a connection from a source switch to a destination switch within the network. ATM signaling has developed along similar lines with signaling procedures developed for the UNI, the NNI, and the B-ICI. In this section we consider UNI signaling.

ITU-T recommendation Q.2931, derived from Q.931, specifies B-ISDN signaling at the *ATM UNI.* ATM Forum UNI signaling 4.0 is based on Q.2931. A number of messages have been defined for use in the setup and release of connections. ATM connections involve many more parameters than narrowband ISDN involves, so the signaling messages carry special fields, called *information elements (IEs)*, that describe the user requests. These signaling messages are transferred across the UNI using the services of the SAAL layer in the control plane. Recall that the SAAL provide reliable message transfer using the SSCOP protocol that operates over AAL5. ITU-T recommendation Q.2130 specifies the SSCF that is used between the Q.2931 signaling application and SSCOP. The signaling cells that are produced by AAL5 use the default virtual channel identifier by VPI = 0 and VCI = 5.

Table 9.5 shows the capabilities provided by UNI 4.0. These capabilities are categorized as being applicable to end-system or switch equipment and as being mandatory (M) or optional (O). Point-to-point as well as point-to-multipoint calls are supported. Point-to-point ABR connections are supported. Signaling of individual QoS parameters and negotiation of traffic parameters are also supported. The leaf-initiated join capability allows an end system to join a point-to-multipoint connection with or without the intervention of the root. Group addressing allows a group of end systems to be identified. Anycast capability allows the setting up of a connection to an end system that is part of an ATM group. The reader is referred to ATM Forum UNI 4.0 signaling specification for details on these capabilities.

Table 9.6 shows a few of the messages used by UNI 4.0 and Q.2931 and their significance when sent by the host or the network. Each signaling message contains a *call reference* that serves as a local identifier for the connection at the UNI. Each message also contains a number of information elements. These include obvious parameters such

TABLE 9.5 Capabilities of UNI 4.0.

Number	Capability	Terminal equipment	Switching system
1	Point-to-point calls	M	M
2	Point-to-multipoint calls	O	M
3	Signaling of individual QoS parameters	O	M
4	Leaf-initiated join	M	M
5	ATM anycast	O	O
6	ABR signaling for point-to-point calls	O	(1)
7	Generic identifier transport	O	O
8	Virtual UNIs	O	O
9	Switched virtual path (VP) service	O	O
10	Proxy signaling	O	O
11	Frame discard	O	O (2)
12	Traffic parameter negotiation	O	O
13	Supplementary services	–	–
13.1	Direct dialing in (DDI)	O	O
13.2	Multiple subscriber number (MSN)	O	O
13.3	Calling line identification presentation (CLIP)	O	O
13.4	Calling line identification restriction (CLIR)	O	O
13.5	Connected identification presentation (COLP)	O	O
13.6	Connected line identification restriction (COLR)	O	O
13.7	Subaddressing (SUB)	O	(3)
13.8	User-user signaling (UUS)	O	O

Notes:
[1]This capability is optional for public networks/switching systems and is mandatory for private networks/switching systems.
[2]Transport of the frame discard indication is mandatory.
[3]This capability is mandatory for network/switching systems (public and private) that support only native E.164 address formats.

as calling and called party numbers, AAL parameters, ATM traffic descriptor and QoS parameters, and connection identifier. Many other mandatory and optional parameters are defined.

The signaling procedures specify the sequence of message exchanges to establish and release connections. They also address the handling of many error conditions that

TABLE 9.6 Signaling messages involved in connection setup.

Message	Meaning (when sent by host)	Meaning (when sent by network)
SETUP	Requests that a call be established	Indicates an incoming call
CALL PROCEEDING	Acknowledges the incoming call	Indicates the call request will be attempted
CONNECT	Indicates acceptance of the call	Indicates the call was accepted
CONNECT ACK	Acknowledges acceptance of the call	Acknowledges making the call
RELEASE	Requests that the call be terminated	Terminates the call
RELEASE COMPLETE	Acknowledges releasing the call	Acknowledges releasing the call

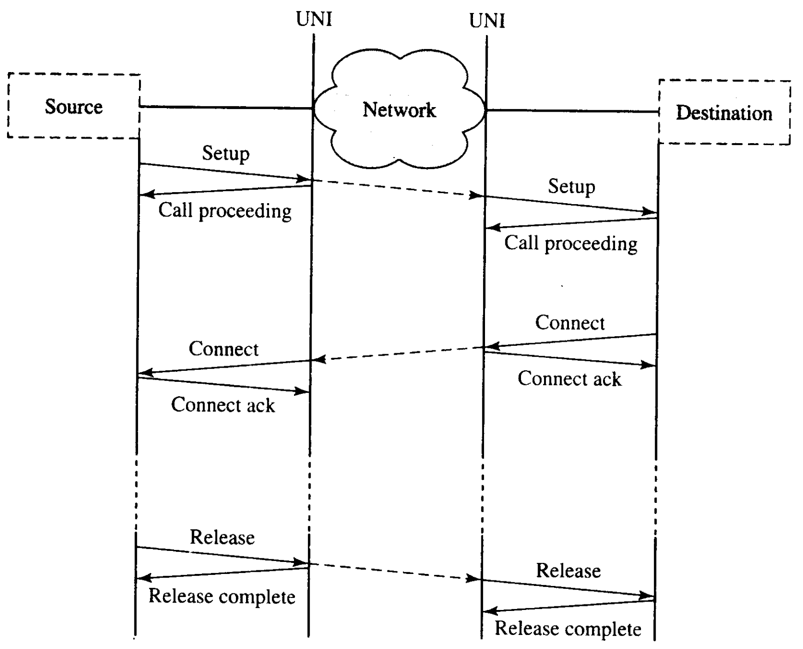


FIGURE 9.23 UNI signaling example.

can arise. In Figure 9.23, we consider the simple case of the establishment of a point-to-point virtual connection using Q.2931.

1. Host A sends a SETUP message on VPI/VCI = 0/5 identifying the destination (host B) and other parameters specifying details of the requested connection.
2. The first switch analyzes the contents of the SETUP message to see whether it can handle the requested connection. If the switch can handle the request, the network returns a CALL PROCEEDING message to the host containing the VPI/VCI for the first link. It also forwards the SETUP message across the network to the destination.
3. Upon arrival of the SETUP message, the destination sends a CALL PROCEEDING message.
4. If the destination accepts the call, it sends a CONNECT message that is forwarded across the network back to host A. The CONNECT messages trigger CONNECT ACKNOWLEDGE messages from the network and eventually from the source.
5. The connection is now established, and the source and destination can exchange cells in the bidirectional VC that has been established.
6. Either party can subsequently initiate the termination of the call by issuing a RELEASE message. This step will trigger RELEASE COMPLETE messages from the network and from the other party.

A point-to-multipoint connection is established as follows. The root of the connection begins by establishing a connection to the first destination (leaf) by using the above procedure. It then issues ADD PARTY messages that attach additional destinations (leaves) to the connection. Note that point-to-multipoint connections are unidirectional, so cells can flow only from the root to the leaves.

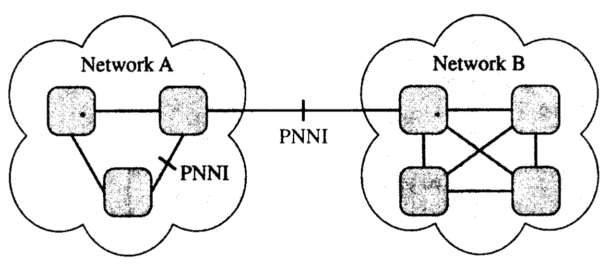


FIGURE 9.24 PNNI contexts.

9.5.3 PNNI Signaling

The ATM Forum has developed the PNNI specification for use between private ATM switches (private network node interface) and between groups of private ATM switches (private network-to-network interface) as shown in Figure 9.24. The PNNI specification includes two types of protocols.

1. A routing protocol that provides for the selection of routes that can meet QoS requirements (this routing protocol is discussed in the next section).
2. A complementary signaling protocol for the exchange of messages between switches and between private networks. In this section we consider the signaling protocol.

The PNNI signaling protocol provides for the establishment and release of point-to-point as well as point-to-multipoint connections. The protocol is based on UNI 4.0 with extensions to provide support for source routing, for crankback (a feature of the routing protocol), and for alternate routing of connection requests in the case of connection setup failure. UNI signaling is asymmetric in that it involves a user and a network. PNNI modifies UNI signaling to make it symmetric. It also includes modifications in the information elements to carry routing information.

PNNI uses source routing where the first switch selects the route to the destination. In Figure 9.25 the source host requests a connection to host B by sending a SETUP message, using UNI signaling. The first switch carries out the connection admission control (CAC) function and returns a CALL PROCEEDING message if it can handle the connection request. The first switch maintains and uses a topology database to calculate a route to the destination that can meet the requirements of the connection contract.[5] The route consists of a vector of switches that are to be traversed. The SETUP message propagates across the network, using the source route. Each switch along the path performs CAC and forwards the SETUP message along the next hop if it can handle the connection request. It also issues a CALL PROCEEDING message to the preceding switch along the route. If the destination accepts the call, a connect message is returned across the network to the source. Connection release proceeds in similar fashion as shown in the figure.

The PNNI routing protocol introduces hierarchy in the ATM network that provides a switch with detailed routing information in its immediate vicinity and only

[5]The PNNI routing protocol includes procedures for distributing the link-state information required to determine the routes that can meet specific QoS requirements.

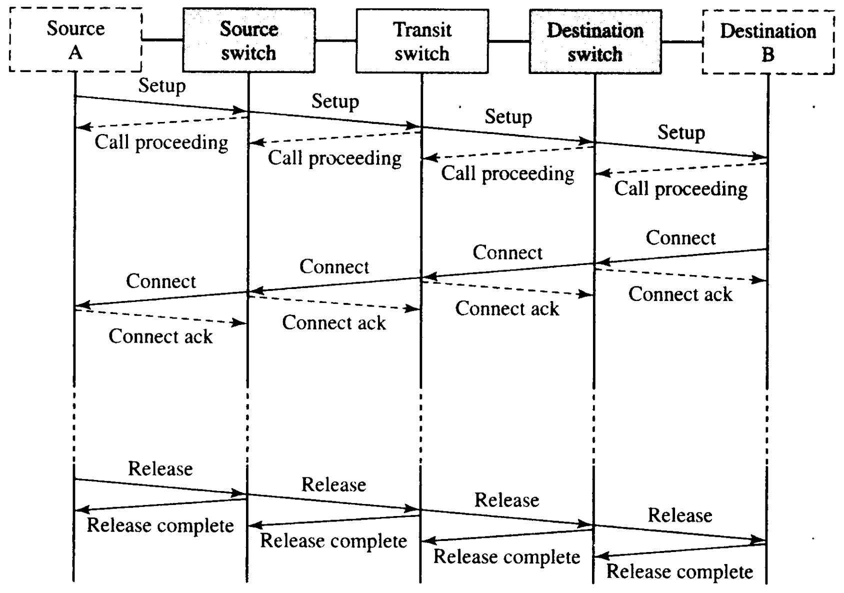


FIGURE 9.25 PNNI signaling example.

summary information about distant destinations. The signaling protocol is somewhat more complicated than the preceding example because of this hierarchical feature.

9.6 PNNI ROUTING

Routing in ATM networks is a subject that has not been investigated as much as other ATM areas such as congestion control and switching. The most visible result on ATM routing comes from the ATM Forum standards work on the **private network-to-network interface (PNNI)**.[6] In this section, we briefly look at the main concepts behind PNNI routing techniques. Unlike Internet routing, which is divided into intradomain and interdomain routing protocols, PNNI works for both cases.

PNNI adopts the link-state philosophy in that each node would know the topology of the network. However, PNNI adds a new twist to make the routing protocol scalable so that it can work well for small networks as well for large networks with thousands of nodes. This goal is achieved by constructing a **routing hierarchy**, as illustrated in Figure 9.26.

As shown in the figure, a *peer group (PG)* is a collection of nodes (physical or logical) where each maintain an identical view of the group. For example, peer group A.1 consists of three nodes A.1.1, A.1.2, and A.1.3. A PG is abstractly represented in a higher-level routing hierarchy as a *logical group node (LGN)*. For example, LGN B represents PG B in the next hierarchy. Note that the definition of PG and LGN is recursive. At the lowest level a PG is a collection of physical switches connected by physical links. At higher levels a PG is a collection of LGNs connected by logical links.

[6]PNNI also stands for private network node interface.

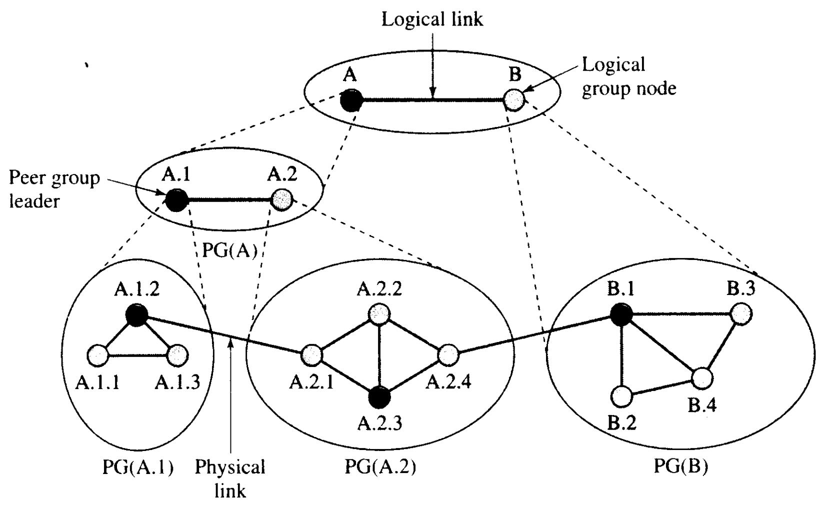


FIGURE 9.26 Example of PNNI hierarchy.

Each PG contains a *peer group leader (PGL)* that actually executes the functions of the logical group nodes for that PG. A PGL summarizes the topological information within the PG and injects this information into the higher-order group. A PGL also passes down summarized topological information to its PG. The advantage of using this hierarchical structure is that each switch maintains only a partial view of the entire network topology, thereby reducing the amount of routing information kept at each switch. For example, from the point of view of switch A.1.1, the topology of the network is shown in Figure 9.27, which is much simpler than that of the entire topology.

PNNI uses source routing to set up connections. First, the source node specifies the entire path across its peer group, which is described by a **designated transit list (DTL)**. Suppose that a station attached to switch A.1.1 requests a connection setup to another station attached to switch B.3. After the source station requests a connection setup, switch A.1.1 chooses the path to be (A.1.1, A.1.2, A.2, B). In this case three DTLs organized in a stack will be built by A.1.1 in the call setup:

DTL: [A.1.1, A.1.2] pointer-2
DTL: [A.1, A.2] pointer-1
DTL: [A, B] pointer-1

The current transit pointer specifies which node in the list is currently being visited at that level, except at the top TDL where the transit pointer specifies the node to be visited next. Thus the top pointer points to A.1.2 and the other pointers point to A.1 and A respectively.

When A.1.2 receives the call setup message, the switch realizes that the entry in the top of the stack is exhausted. After removing the top DTL, A.1.2 finds that the next entry is A.2, which is also its immediate neighbor. The DTLs become

DTL: [A.1, A.2] pointer-2
DTL: [A, B] pointer-1

When A.2.1 receives the call setup message, the switch finds that the target has been reached, since A.2.1 is in A.2. So A.2.1 builds a route to B, say, through A.2.3

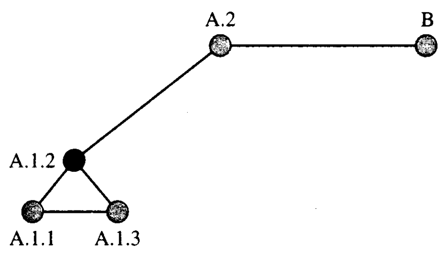


FIGURE 9.27 Topology seen by switch A.1.1.

and A.2.4, and pushes a new DTL onto the stack:

DTL: [A.2.1, A.2.3, A.2.4] pointer-2
DTL: [A.1, A.2] pointer-2
DTL: [A, B] pointer-1

When A.2.3 receives the call setup message, the switch advances the current transit pointer and forwards the message to A.2.4. When A.2.4 receives the call setup message, it finds that the targets at the top two DTLs have been reached. So A.2.4 removes the top two DTLs and forwards the message with the following DTL to its neighbor:

DTL: [A, B] pointer-2

When B.1 receives the call setup message, B.1 finds that the curernt DTL has been reached. B.1 builds a new DTL, giving

DTL: [B.1, B.3] pointer-2
DTL: [A, B] pointer-2

When the message reaches its destination, B.3 determines that it is a DTL terminator, since all DTLs are at the end and B.3 is the lowest-level node.

It is possible that a call setup will be blocked when the requested resources at a switch are not available. PNNI provides **crankback** and **alternate routing**. When a call is blocked at a particular DTL, it is cranked back to the creator of the DTL.

A switch determines the path based on the connection's traffic descriptor, QoS requirements, and the resources stored in its database. Because CAC is not standardized, PNNI uses a **generic connection admission control (GCAC)** to select a path that is likely to satisfy the connection's end-to-end traffic and QoS requirements. GCAC, however, only predicts the most likely path that can satisfy the connection traffic, since the information known by the switch may be outdated when the actual call request is made at other switches along the path.

GCAC requires the following parameters:

- Available cell rate (ACR): A measure of available bandwidth on the link.
- Cell rate margin (CRM): A measure of the difference between the aggregate allocated bandwidth and sustained rate of the existing connections.
- Variance factor (VF): A relative measure of the CRM normalized by the variance of the aggregate rate.

When a new connection with peak cell rate PCR and sustained cell rate SCR requests a connection setup, the GCAC algorithm examines each link along the path and performs the following decision:

```
if $PCR \le ACR$
        include the link
else if $SCR > ACR$
        exclude the link
else if $[ACR - SCR][ACR - SCR + 2 * CRM] \ge VF * SCR\ (PCR - SCR)$
        include the link
else
        exclude the link
```

The VF is CRM^2/VAR, and the variance is given by

$$VAR = \sum_i SCR(i)[PCR(i) - SCR(i)]$$

After the path has been selected, each switch along the path eventually has to perform its own CAC to decide the acceptance/rejection of the connection request. Note that GCAC uses VF only in the case SCR < ACR < PCR.

9.7 CLASSICAL IP OVER ATM

The **classical IP over ATM (CLIP)** model [RFC 2255] is an IETF specification whereby IP treats ATM as another subnetwork to which IP hosts and routers are attached. In the CLIP model multiple IP subnetworks are typically overlaid on top of an ATM network. The part of an ATM network that belongs to the same IP subnetwork is called a **logical IP subnetwork (LIS)**, as shown in Figure 9.28. All members (IP end systems) in the same LIS must use the same IP address prefix (e.g., the same network number and subnet number). Two members in the same LIS communicate directly via an ATM virtual channel connection (VCC).

Each LIS operates and communicates independently of other LISs on the same ATM network. Communications to hosts outside the LIS must be provided via an IP router that is connected to the LIS. Therefore, members that belong to different LISs must communicate through router(s).

Suppose a host (host S) wants to use CLIP to send packets to another host (host D). When host S sends the first packet to host D in the same LIS, host S knows only the IP address of host D. To set up a VCC, host S needs to know the ATM address of host D. Host S resolves the ATM address of host D from the IP address by contacting an **ATM Address Resolution Protocol** (ATM ARP) server in the LIS. The ATM address of the ATM ARP server is configured at each host. When a host boots up, it registers its IP and ATM addresses to the ATM ARP server on the same LIS. When a host wants to resolve the ATM address of another host from the IP address, the first host asks the ATM ARP server for the corresponding ATM address. After the host receives an ATM ARP reply from the ATM ARP server, the host can establish a VCC to the destination host and send packets over the VCC. This process involves fragmenting the IP packet into ATM cells at the source host and the reassembly of the packet at the destination host.

What happens if the destination host belongs to another LIS? In this situation the source host simply establishes a VCC to the router connected to the same LIS. The router

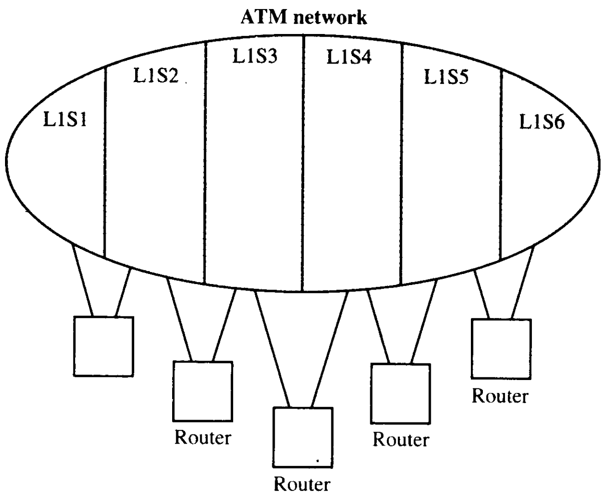


FIGURE 9.28 Classical IP over ATM model.

examines the IP packet, determines the next-hop router, establishes a VCC, and forwards the packet along to the next router. The process is continued until the router of the LIS of the destination is reached, and the packet is then delivered to the destination host.

In CLIP, IP packets sent from the source host to the destination host in a different LIS must undergo routing through the LIS router, even if it is possible to establish a direct VCC between the two IP members over the ATM network. This requirement precludes the establishment of a VCC with a specific QoS between end systems.

ATM IS DEAD, LONG LIVE ATM!

The development of ATM standards was a massive effort that attempted to develop an entire future network architecture from the ground up. BISDN was viewed as a future multiservice network that could encompass LANs and WANs in a common framework. Alas, this was not to be.

The ATM connection-oriented networking paradigm emerged at the same time as the explosion of the World Wide Web. The Web is built on HTTP and the TCP/IP protocol suite which is decidedly connectionless in its network orientation. ATM to the user is not what was needed and so the vision of an end-to-end universal ATM network perished.

ATM technology does have advantages such as facilitating high-speed switching and enabling the management of traffic flows in the network. For this reason ATM has found wide application in carrier backbone networks and, to a lesser extent, in campus networks that require prioritization of traffic flows.

The long-term future of ATM in an all-IP world is uncertain. In effect, it is possible that ATM may disappear over the longer run, but that many of its innovations in high-speed switching, traffic management, and QoS will survive in an IP-networking framework.

SUMMARY

In this chapter we examined the architecture of ATM networks. We discussed the BISDN reference model and the role of the control plane in setting up ATM connections. We then examined the structure of the ATM header and its relationship to virtual paths and virtual connections. The connection setup involves the establishment of a traffic contract between the user and the network that commits the user to a certain pattern of cell transmission requests and the network to a certain level of QoS support. We examined the traffic management mechanisms that allow ATM to provide QoS guarantees across a network. The various types of ATM service categories to meet various information transfer requirements were introduced.

We discussed the role of the ATM adaptation layer in providing support for a wide range of user applications. This support included segmentation and reassembly, reliable transmission, timing recovery and synchronization, and multiplexing.

We introduced PNNI signaling and its associated routing protocol which are used to set up connections with QoS guarantees across ATM networks. Finally, we introduced classical IP over ATM.

CHECKLIST OF IMPORTANT TERMS

alternate routing
ATM adaptation layer (AAL)
ATM Address Resolution Protocol(ATM ARP)
ATM layer
broadband intercarrier interface (B-ICI)
classical IP over ATM (CLIP)
common part (CSCP)
control plane
convergence sublayer (CS)
crankback
designated transit list (DTL)
generic connection admission control (GCAC)
logical IP subnetwork (LIS)
message identifier (MID)
multiplexing identifier
network-network interface (NNI)
private network-to-network interface (PNNI)
routing hierarchy
segmentation and reassembly (SAR)
service specific part (SSCS)
user-network interface (UNI)

FURTHER READING

ATM Forum Technical Committee, "ATM User-Network Interface (UNI) Signaling Specification, Version 4.0," af-sig-0061.000, July 1996.

ATM Forum Technical Committee, "Private Network-Network Interface Specification, Version 1.0," af-pnni-0055.000, March 1996.

ATM Forum Technical Committee, "Traffic Management Specification, Version 4.0," af-tm-0056.000, April 1996.

Black, U., *ATM: Foundation for Broadband Networks*, Prentice Hall, Englewood Cliffs, New Jersey, 1995.

Dutton, H. J. R. and P. Lenhard, *Asynchronous Transfer Mode (ATM)*, Prentice Hall PTR, Upper Saddle River, New Jersey, 1995.

Helgert, H. J., *Integrated Services Digital Networks: Architectures, Protocols, Standards*, Addison-Wesley, Reading, Massachusetts, 1991.

Ibe, O. C., *Essentials of ATM Networks and Services*, Addison-Wesley, Reading, Massachusetts, 1997. Excellent concise introduction to ATM.

ITU draft, I.363.2, "B-ISDN ATM Adaptation Layer Type 2 Specification," November 1996.

Leon-Garcia, A., *Probability and Random Processes for Electrical Engineering*, Addison-Wesley, Reading, Massachusetts, 1994.

McDysan, D. E. and D. L. Spohn, *ATM: Theory and Application*, McGraw-Hill, New York, 1995.

Prycker, M. de, *Asynchronous Transfer Mode: Solution for Broadband ISDN*, Prentice Hall, Englewood Cliffs, New Jersey, 1995.

See our website for additional references available through the Internet.

PROBLEMS

9.1. Suppose that instead of ATM, BISDN had adopted a transfer mode that would provide constant-bit-rate connections with bit rates given by integer multiples of 64 kbps connections. Comment on the multiplexing and switching procedures that would be required to provide this transfer mode. Can you give some reasons why BISDN did not adopt this transfer mode?

9.2. (a) Compare the bandwidth management capabilities provided by ATM virtual paths to the capabilities provided by SONET networks.
(b) Can ATM virtual paths be adapted to provide the fault tolerance capabilities of SONET rings? Explain your answer.

9.3. In Chapter 6 we saw that the performance of MACs for LANs can depend strongly on the delay-bandwidth product of the LAN. Consider the use of ATM in a LAN. Does the performance depend on the delay-bandwidth product? If yes, explain how; if no explain why not? Does the same conclusion apply to any LAN that involves the switching packets, rather than the use of a broadcast medium?

9.4. Does the performance of ATM depend strongly on the delay-bandwidth product of a wide-area network? Specifically, consider the performance of ATM congestion control in a network with large delay-bandwidth product. Does the size of the buffers in the ATM switches influence the severity of the problem? Give an order of magnitude for the amount of buffering required in the switches.

9.5. Compare a conventional TDM leased line with an ATM PVC from the user's point of view and from the network operator's point of view. Which features of PVCs make them attractive from both points of view?

9.6. An inverse multiplexer is a device that takes as input a high-speed digital stream and divides it into several lower-speed streams that are then transmitted over parallel transmission lines to the same destination. Suppose that a very high speed ATM stream is to be sent over an inverse multiplexer. Explain which requirements must be met so that the inverse demultiplexer produces the original ATM stream.

9.7. Suppose an ATM switch has 32 input ports and 32 output ports.
(a) Theoretically, how many connections can the switch support?
(b) What are the table lookup requirements for supporting a large number of connections? Do they limit the practical size on the number of connections that can actually be supported?
(c) (Optional) Do a web search on content addressable memories (CAMs) and explain how these can help in addressing the table-lookup problem.

9.8. Explain how the header error checksum can be used to synchronize to the boundary of a sequence of contiguous ATM cells. What is the probability that an arbitrary five-octet block will satisfy the header error checksum? Assume bits are equally likely to be 0 or 1. What is the probability that two random five-octet blocks that correspond to two consecutive headers pass the error check?

9.9. We saw that ATM provides a GFC field in the UNI ATM header. Suppose that several terminals share a medium to access an ATM network.
(a) Explain why flow control may be required to regulate the access of traffic from the terminals into the network. Explain how the GFC field can be used to do this task?
(b) Explain how the GFC field can be used as a subaddress to provide point-to-multipoint access to an ATM network. Does this usâge conflict with the flow control requirement?

9.10. The purpose of the header error control (HEC) field is to protect against errors in the header that may result in the misaddressing and misdelivery of cells. The CRC used in the ATM header can correct all single errors and can detect (but not correct) all double errors that occur in the header. Some, but not all, multiple errors in excess of two can also be detected.
(a) Suppose that bit errors occur at random and that the bit error rate is p. Find the probability that the header contains no errors, a single error, a double error, more than two errors. Evaluate these probabilities for $p = 10^{-3}, 10^{-6}, 10^{-9}$.
(b) Relate the calculations in part (a) to the CMR experienced in such an ATM network.
(c) In practice the bit errors may not always be random, and so to protect against bursts of errors the following adaptive procedure may be used.

Normally the receiver is in the "correction" mode. If a header is error free, the receiver stays in this mode; if the receiver detects a single error, the receiver corrects the error and changes to "detection" mode; if the receiver detects a multiple error, the receiver changes to HEC error detection mode. If the receiver is in "detection" mode and one or more errors are detected in the header, then the cell is discarded and the receiver stays in the detection mode; otherwise, the receiver changes to the correction mode.

Explain why this procedure protects against bursts of errors. If the bit errors are independent, what is the probability that two consecutive headers contain single errors and hence that a cell is discarded unnecessarily?

9.11. What is the difference between CER and CLR? Why is one negotiated during connection setup and the other is not?

9.12. Why does the calculation of CER and CMR exclude blocks that are counted in the SECBR?

9.13. Explain how weighted fair queueing scheduling can be used to affect the CLR and CTD experienced by cells in an ATM connection.

9.14. Explain the effect of a single cell loss from a long packet in a situation that uses an end-to-end ARQ retransmission protocol. Can you think of strategy to deal with such losses? Can you think of a way to ameliorate the impact of packet retransmission?

9.15. Consider a sequence of ATM cells carrying PCM voice from a single speaker.
(a) What are the appropriate traffic descriptors for this sequence of cells, and what is an appropriate leaky bucket for policing this stream?
(b) Suppose that an ATM connection is to carry the cell streams for M speakers. What are appropriate traffic descriptors for the resulting aggregate stream, and how can it be policed?

9.16. Consider a sequence of ATM cells carrying PCM voice from a single speaker, but suppose that silence suppression is used.
(a) What are the appropriate traffic descriptors for this sequence of cells, and what is an appropriate leaky bucket(s) arrangement for policing this stream? Which situation leads to nonconforming cells?
(b) Suppose that an ATM connection is to carry the cell streams for M speakers. What are appropriate traffic descriptors for the resulting aggregate stream, and how can it be policed?

9.17. Suppose that constant-length packets (of size equal to M cells) arrive at a source to be carried by an ATM connection and that such packets are separated by exponential random times T. What are the appropriate traffic descriptors for this sequence of cells, and what is an appropriate leaky bucket(s) arrangement for policing this stream? Which situation leads to nonconforming cells?

9.18. Explain why each specific set of traffic descriptors and QoS parameters were selected for each of the ATM service categories.

9.19. Suppose that IP packets use AAL5 prior to transmission over an ATM connection. Explain the transfer-delay properties of the IP packets if the ATM connection is of the following type: CBR, rt-VBR, nrt-VBR, ABR, or UBR.

9.20. Proponents of ATM argue that VBR connections provide a means of attaining multiplexing gains while providing QoS. Proponents of IP argue that connectionless IP routing can provide much higher multiplexing gains. Can you think of arguments to support each claim. Are these claims conflicting, or can they both be correct?

9.21. (a) Consider a link that carries connections of individual voice calls using PCM. What information is required to perform call admission control on the link?
(b) Now suppose that the link carries connections of individual voice calls using PCM but with silence suppression. What information is required to do call admission control?

9.22. Suppose that an ATM traffic stream contains cells of two priorities, that is, high-priority cells with CLP $= 0$ in the headers and low-priority cells with CLP $= 1$.
(a) Suppose we wish to police the peak cell rate of the CLP $= 0$ traffic to p_0 as well as the peak rate of the combined CLP $= 0$ and CLP $= 1$ traffic to p_{0+1}. Give an arrangement of two leaky buckets to do this policing. Nonconforming cells are dropped.
(b) Compare the following policing schemes: (1) police CLP $= 0$ traffic to peak rate p_0 and police CLP $= 1$ traffic to peak rate p_1, (2) police the combined CLP $= 0$ and CLP $= 1$ traffic to peak rate $p_0 + p_1$. Which approach is more flexible?
(c) Repeat part (a) if CLP $= 0$ cells that do not conform to the p_0 are tagged by changing the CLP bit to 1. Cells that do not conform to p_{0+1} are dropped.

9.23. Suppose that an ATM traffic stream contains cells of two priorities, that is, high-priority cells with CLP $= 0$ in the headers and low-priority cells with CLP $= 1$.

(a) Suppose we wish to police the sustainable cell rate of the CLP $= 0$ traffic to SCR_0 and BT as well as the peak rate of the combined CLP $= 0$ and CLP $= 1$ traffic to p_{0+1}. Give an arrangement of two leaky buckets to do this policing. Nonconforming cells are dropped.

(b) Repeat part (a) if cells that do not conform to SCR_0 and BT are tagged by changing the CLP bit to 1. Cells that do not conform to p_{0+1} are dropped.

9.24. Suppose that an ATM traffic stream contains cells of two priorities, that is, high-priority cells with CLP $= 0$ in the headers and low-priority cells with CLP $= 1$. Suppose we wish to police the peak cell rate and the sustainable cell rate of the combined CLP $= 0$ and CLP $= 1$ traffic. Give an arrangement of two leaky buckets to do this policing. Nonconforming cells are dropped.

9.25. Explain how weighted fair queueing might be used to combine the five ATM service categories onto a single ATM transmission link. How are the different service categories affected as congestion on the link increases?

9.26. (a) Discuss what is involved in calculating the end-to-end CLR, CTD, and CDV.

(b) Compare the following two approaches to allocating the end-to-end QoS to per link QoS: equal allocation to each link; unequal allocation to various links. Which is more flexible? Which is more complex?

9.27. Suppose that an application uses the reliable stream service of TCP that in turns uses IP over ATM over AAL5.

(a) Compare the performance seen by the application if the AAL5 uses CBR connection; nrt-VBR connection; ABR connection; UBR connection.

(b) Discuss the effect on the peformance seen by the application if the ATM connection encounters congestion somewhere in the network.

9.28. Suppose that an ATM connection carries voice over AAL1. Suppose that the packetization delay is to be kept below 10 ms.

(a) Calculate the percentage of overhead if the voice is encoded using PCM.

(b) Calculate the percentage of overhead if the voice is encoded using a 12 kbps speech-encoding scheme from cellular telephony.

9.29. Explain how the three-bit sequence number in the AAL1 header can be used to deal with lost cells and with misinserted cells.

9.30. How much delay is introduced by the two interleaving techniques used in AAL1?

9.31. (a) How many low-bit-rate calls can be supported by AAL2 in a single ATM connection?

(b) Estimate the bit rate of the connection if an AAL2 carries the maximum number of calls carrying voice at 12 kbps.

(c) What is the percentage of overhead in part (b)?

9.32. Compare the overhead of AAL3/4 with that of AAL5 for a 64K byte packet.

9.33. Discuss the purpose of the error checking that is carried out at the end systems and in the network for an ATM connection that carries cells produced by AAL3/4. Repeat for AAL5.

9.34. Suppose that in Figure 9.17 packets from A and B arrive simultaneously and each produces 10 cells. Show the sequence of SPDUs produced including segment type, sequence number, and multiplexing ID.

9.35. Consider SSCOP, the AAL protocol for signaling. Discuss the operation of the Selective Repeat ARQ procedure to recover from cell losses. In particular, discuss how the protocol differs from the Selective Repeat ARQ protocol introduced in Chapter 5.

9.36. Can the SSCOP AAL protocol be modified to provide the same reliable stream service that is provided by TCP? If yes, explain how; if no, explain why not.

9.37. The "cells in frames" proposal attempts to implement ATM on workstations attached to a switched Ethernet LAN. The workstation implements the ATM protocol stack to produce cells, but the cells are transmitted using Ethernet frames as follows. The payload of the Ethernet frame consists of a four-byte CIF header, followed by a single ATM header, and up to 31 ATM cell payloads.
(a) Find the percentage of overhead of this approach and compare it to standard ATM.
(b) The workstation implements ATM signaling, and the NIC driver is modified to handle several queues to provide QoS. Discuss the changes required in the Ethernet switch so that it can connect directly to an ATM switch.

9.38. Compare the size of the address spaces provided by E-164 addressing, AESA addressing, IPv4 addressing, IPV6 addressing, and IEEE 802 MAC addressing.

9.39. Can IP addresses be used in ATM? Explain why or why not?

9.40. Identify the components that contribute to the end-to-end delay experienced in setting up an ATM connection using PNNI.

9.41. Describe the sequence of DTLs that are used in setting up a connection from A.1.3 to A.2.2 in Figure 9.26. Repeat for a connection from B.4 to A.1.2.

9.42. (a) Discuss the differences and similarities between PNNI and OSPF.
(b) Can PNNI be modified to provide QoS routing in the Internet? Explain.

9.43. Compare the hierarchical features of the combination of BGP4 and OSPF with the features of PNNI.

9.44. Which aspects of the ATM network architecture depend on the fixed-length nature of ATM cells? What happens if ATM cells are allowed to be variable in length?

9.45. Which aspects of ATM traffic management change if ATM connections must belong to one of a limited number of classes and if QoS is guaranteed not to individual connections but the class as a whole? Can VPs play a role in providing QoS to these classes?

9.46. Explain how the ATM architecture facilitates the creation of multiple virtual networks that coexist over the same physical ATM network infrastructure but that can be operated as if they were separate independent networks. Explain how such virtual networks can be created and terminated on demand.

CHAPTER 10

Advanced Network Architectures

In the previous two chapters we examined two key network architectures: (1) IPv4 and its next version IPv6 and (2) ATM. IP has proven to be extremely powerful because it was designed to provide internetworking capabilities over a variety of underlying network technologies. However, traditional IP networks can only provide best-effort service and so cannot readily support applications that require real-time response. ATM, on the other hand, was designed at the outset to provide end-to-end Quality of Service (QoS) to support a wide range of services such as voice, data, and video. Nevertheless, ATM does not have the scalability and cannot achieve the ubiquity of IP. Recent interest in supporting QoS, scalability, reliability, and other enhancements in IP networks have spurred proposals for enabling new capabilities in the next-generation IP infrastructures. In this chapter we consider some proposals that are intended to support QoS, provide traffic engineering, and support the deployment of new services.

The contents of the present chapter are technology-oriented and evolving in nature. Moreover, due to the rapid advances in technology, today's preferred solutions may easily become out-of-date tomorrow as new technology makes alternatives more desirable. The present chapter introduces a sampling of advanced networking architectures that are being adopted by the industry and believed to have a good chance of being incorporated into emerging networks. The first part of the chapter discusses QoS for the Internet. The second part presents promising networking architectures for the packet and transport networks. The third part describes the protocols that are available for the transport of multimedia information over packet networks. The chapter is organized as follows:

1. *Integrated services for the Internet.* We introduce the integrated services model for the Internet and describe two services that have been standardized: guaranteed service for providing real-time delivery guarantees and controlled-load service for providing low-load performance.

2. *RSVP.* We present the Resource Reservation Protocol (RSVP), which is used to set up resource reservations in routers along the intended path of a packet flow.
3. *Differentiated services.* We describe the differentiated services model that has been introduced as a scalable approach to provide levels of QoS support in the Internet.
4. *Network interconnection models.* We consider network interconnection models that arise when client networks are interconnected through a service network. In particular, we introduce two interconnection models called the overlay model and the peer model.
5. *MPLS and GMPLS.* We present multiprotocol label switching (MPLS), which adopts the peer model and constitutes a promising architecture for controlling the flow of packet traffic in a core packet network. We then discuss an extension of MPLS for nonpacket networks called GMPLS.
6. *Real-Time Transport Protocol.* We introduce RTP, which operates over UDP and provides the framework for the transport of multimedia over Internet.
7. *Session control protocols.* We introduce the Session Initiation Protocol, which has been designed to work with RTP. We also introduce the ITU-T H.323 recommendation for packet-based multimedia communication systems. We also briefly discuss the interworking of traditional telephony and corresponding systems in IP networks.

10.1 INTEGRATED SERVICES IN THE INTERNET

Traditionally, the Internet has provided best-effort service to every user regardless of its requirements. Because every user receives the same level of service, congestion in the network often results in serious degradation for applications that require some minimum amount of bandwidth to function properly. As the Internet becomes universally available, there is also interest in providing real-time service delivery to some applications such as IP telephony. Thus an interest has developed in having the Internet provide some degree of QoS.

To provide different QoS commitments, the IETF developed the **integrated services (intserv) model**, which requires resources such as bandwidth and buffers to be explicitly reserved for a given data flow to ensure that the application receives its requested QoS. The model requires the use of *packet classifiers* to identify flows that are to receive a certain level of service as shown in Figure 10.1. It also requires the use of *packet schedulers* to handle the forwarding of different packet flows in a manner that ensures that QoS commitments are met. *Admission control* is also required to determine whether a router has the necessary resources to accept a new flow. Thus the intserv model is analogous to the ATM model where admission control coupled with policing are used to provide QoS to individual applications.

The Resource Reservation Protocol (RSVP), which is discussed in Section 10.2, is used by the intserv model to reserve appropriate resources along the path traversed by a new flow requesting a QoS service. RSVP essentially informs each router of the requested QoS, and if the flow is found admissible, each router in turn adjusts its packet classifier and scheduler to handle the given packet flow.

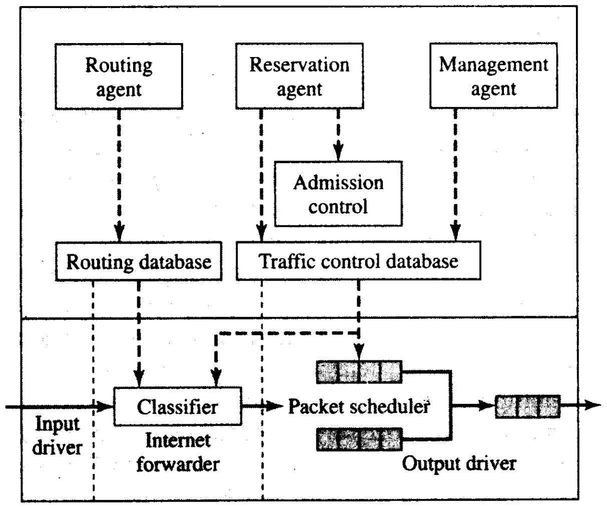


FIGURE 10.1 Router model in integrated services IP.

- A *flow descriptor* is used to describe the traffic and QoS requirements of a flow. The flow descriptor consists of two parts: a *filter specification (filterspec)*; and a *flow specification (flowspec)*. The filterspec provides the information required by the packet classifier to identify the packets that belong to the flow. The flowspec consists of a *traffic specification (Tspec)* and a *service request specification (Rspec)*. The Tspec specifies the traffic behavior of the flow in terms of a token bucket as discussed in Chapter 7. The Rspec specifies the requested QoS in terms of bandwidth, packet delay, or packet loss.

The intserv model introduces two new services: guaranteed service and controlled-load service.

10.1.1 Guaranteed Service

The **guaranteed service** in the Internet can be used for applications that require real-time service delivery. For this type of application, data that is delivered to the application after a certain time limit is generally considered worthless. Thus guaranteed service has been designed to provide a firm bound on the end-to-end packet delay for a flow. How does guaranteed service provide a firm delay bound?

Let b be the token-bucket size and r be the token rate. Recall that from Section 7.8.1, if a flow is shaped by a (b, r) token bucket and is guaranteed to receive at least R bits/second, then the delay experienced by the flow will be bounded by b/R with a fluid flow model, assuming that $R > r$. This delay bound has to be adjusted by error terms accounting for the deviation from the fluid flow model in the actual router or switch, as indicated by Equation (7.24).

To support guaranteed service, each router must know the traffic characteristics of the flow and the desired service. Based on this information, the router uses admission control to determine whether a new flow should be accepted. Once a new flow is accepted, the router should police the flow to ensure compliance with the promised traffic characteristics.

10.1.2 Controlled-Load Service

The **controlled-load service** is intended for adaptive applications that can tolerate some delay but that are sensitive to traffic overload conditions. These applications typically perform satisfactorily when the network is lightly loaded but degrade significantly when the network is heavily loaded. Thus the controlled-load service was designed to provide approximately the same service as the best-effort service in a lightly loaded network regardless of the actual network condition. The above interpretation is deliberately imprecise for a reason. Unlike the guaranteed service that specifies a quantitative guarantee, the controlled-load service is qualitative in the sense that no target values on delay or loss are specified. However, an application requesting a controlled-load service can expect low queueing delay and low packet loss, which is a typical behavior of a statistical multiplexer that is not congested. Because of these loose definitions of delay and loss, the controlled-load service requires less implementation complexity than the guaranteed service requires. For example, the controlled-load service does not require the router to implement a sophisticated per-flow scheduling algorithm.

As in the guaranteed service, an application requesting a controlled-load service has to provide the network with the token bucket specification of its flow. The network uses admission control and policing to ensure that enough resources are available for the flow. Flows that conform to the token bucket specification should be served with low delay and low loss. Flows that are nonconforming should be treated as best-effort service.

10.2 RSVP

The resource **ReSerVation Protocol (RSVP)** was designed as an IP signaling protocol for the intserv model. RSVP allows a host to request a specific QoS resource for a particular flow and enables routers to provide the requested QoS along the path(s) by setting up appropriate states.[1]

Because IP traditionally did not have any signaling protocol, the RSVP designers had the liberty of constructing the protocol from scratch. RSVP has the following features:

- Performs resource reservations for multicast (multipoint-to-multipoint) applications, adapting dynamically to changing group membership and changing routes. Unicast applications are handled as a special case.
- Requests resources in one direction from a sender to a receiver (i.e., a simplex resource reservation). A bidirectional resource reservation requires both end systems to initiate separate reservations.
- Requires the receiver to initiate and maintain the resource reservation.

[1]RSVP can be extended for use in other situations. For example, RSVP has been proposed to set up connections and install state related to forwarding in MPLS [RFC 3209] and in GMPLS.

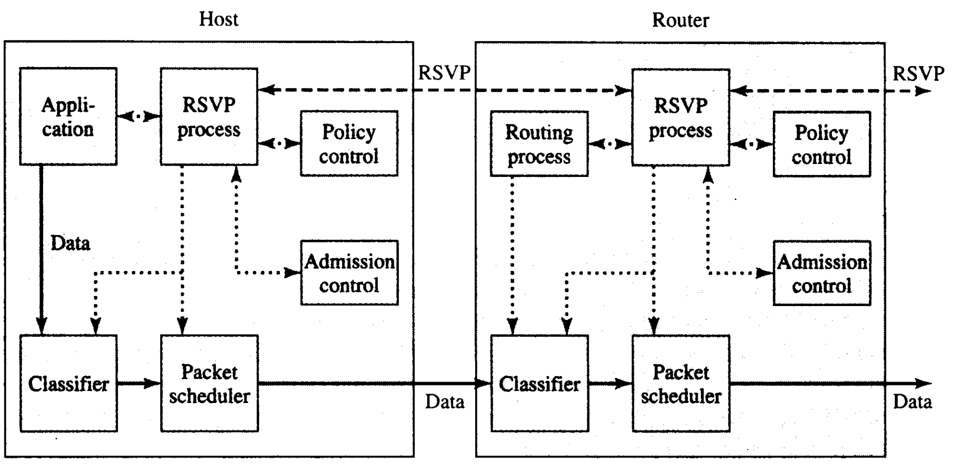


FIGURE 10.2 RSVP architecture.

- Maintains soft state at each intermediate router: A resource reservation at a router is maintained for a limited time only, and so the sender must periodically refresh its reservation. If a certain reservation is not refreshed within a certain time-out period, the corresponding state is deleted and the resource is released.
- Does not require each router to be RSVP capable. Non-RSVP-capable routers use a best-effort delivery technique.
- Provides different reservation styles so that requests may be merged in several ways according to the applications.
- Supports both IPv4 and IPv6.

To enable resource reservations an RSVP process (or daemon) in each node has to interact with other modules, as shown in Figure 10.2. If the node is a host, then the application requiring a QoS delivery service first has to make a request to an RSVP process that in turn passes RSVP messages from one node to another. Each RSVP process passes control to its two local control modules: policy control and admission control. The policy control determines if the application is allowed to make the reservation. Relevant issues to be determined include authentication, accounting, and access control. The admission control determines whether the node has sufficient resources to satisfy the requested QoS. If both tests succeed, parameters are set in the classifier and packet scheduler to exercise the reservation. If one of the tests fails at any node, an error notification is returned to the originating application.

In the RSVP parlance a **session** is defined to be a data flow identified by a destination and a transport-layer protocol. Specifically, an RSVP session is defined by its destination IP address, IP protocol number, and optionally, destination port. The destination IP address can be unicast or multicast. The optional destination port may be specified by a TCP or UDP port number. Within a session there can be more than one sender, and so in the general multicast case a single RSVP session involves packet flows from *multiple senders to multiple receivers* as indicated in Figure 10.3 where the cloud represents the mesh of multicast delivery paths determined by the multicast routing protocol. In

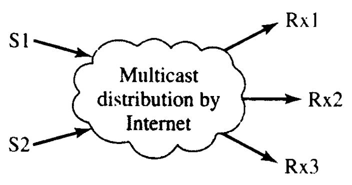


FIGURE 10.3 Packet flow in an RSVP session.

the figure every packet produced by sender S*i* is delivered to every receiver Rx*j* in the session. In the case of a unicast distribution session there is only one receiver.

An RSVP reservation request consists of a **flowspec** and a **filterspec**. The flowspec is used to set parameters in the node's packet scheduler. Generally, the parameters include a service class, an Rspec (R for reserve) that defines the requested QoS, and a Tspec (T for traffic) that describes the sender's traffic characteristics. The filterspec specifies the set of packets that can use the reservation and is used to set parameters in the packet classifier. The set of packets is typically defined in terms of one or more sender IP addresses and sender ports that are allowed to share the reserved resources.

10.2.1 Receiver-Initiated Reservation

RSVP adopts the receiver-initiated reservation principle, meaning that the receiver rather than the sender initiates the resource reservation. This principle is similar in spirit to many multicast routing algorithms where each receiver joins and leaves the multicast group independently without affecting other receivers in the group. The main motivation for adopting this principle is because RSVP is primarily designed to support multiparty conferencing with heterogeneous receivers. In this environment the receiver actually knows how much bandwidth it needs. If the sender were to make the reservation request, then the sender must obtain the bandwidth requirement from each receiver. This may cause an implosion problem for large multicast groups.

One problem with the receiver-initiated reservation is that the receiver does not directly know the path taken by data packets. RSVP solves this by introducing **Path** messages that originate from the sender and travel along the unicast/multicast routes towards the receiver(s). The main purposes of the Path message are to store the "path state" in each node along the path and to carry information regarding the sender's traffic characteristics and the end-to-end path properties. The path state includes the unicast IP address of the previous RSVP-capable node. The Path message contains the following information:

- **Phop:** The address of the previous-hop RSVP-capable node that forwards the Path message.
- **Sender template:** The sender IP address and optionally the sender port.
- **Sender Tspec:** The sender's traffic characteristics.
- **Adspec:** Information used to advertise the end-to-end path to receivers. The contents of the Adspec may be updated by each router along the path.

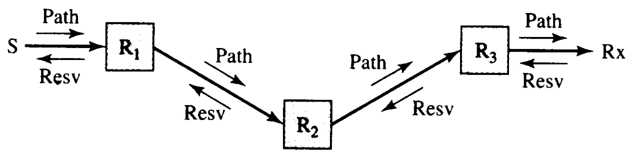


FIGURE 10.4 RSVP Path and Resv messages.

Upon receiving the Path message, the receiver sends a **Resv** message in a unicast fashion towards the sender along the reverse path that the data packets use. A Resv message carries reservation requests to the routers along the path.

Figure 10.4 illustrates the traces of Path and Resv messages. When sender S has data to send to receiver Rx, S sends Path messages periodically toward Rx along the path determined by the routing protocol. The nodes that receive the Path message record the state of the path. When Rx receives a Path message, Rx can begin installing the reservation by sending a Resv message along the reverse path. The IP destination address of a Resv message is the unicast address of a previous-hop node, obtained from the path state. RSVP messages are sent as "raw" IP datagrams with protocol number 46. However, it is also possible to encapsulate RSVP messages as UDP messages for hosts that do not support the raw I/O capability.

10.2.2 Reservation Merging

When there are multiple receivers, the resource is not reserved for each receiver but is shared up to the point where the paths to different receivers diverge. From the receiver's point of view, RSVP merges the reservation requests from different receivers at the point where multiple requests converge. When a reservation request propagates upstream toward the sender, it stops at the point where there is already an existing reservation that is equal to or greater than that being requested. The new reservation request is merged with the existing reservation and is not forwarded further. This may reduce the amount of RSVP traffic appreciably when there are many receivers in the multicast tree. Figure 10.5 illustrates the reservation merging example. Here the reservation requests from Rx1 and Rx2 are merged at R_3, which are in turn merged at R_2 with the request coming from Rx3.

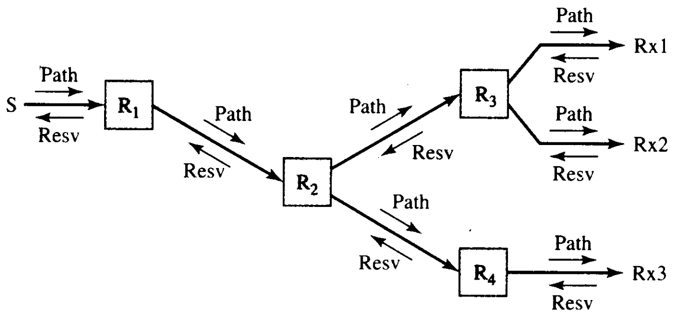


FIGURE 10.5 Merging reservations.

10.2.3 Reservation Styles

A reservation request specifies a reservation style that indicates whether senders in the session have distinct or shared resource reservations and whether senders are selected according to an explicit list. The following three examples show how the need for different reservation styles comes about.

EXAMPLE Audioconferencing

An audioconference meeting in which only one participant is allowed to speak by an automated moderator. In this case, only enough bandwidth to support one voice signal is required at every link that connects senders to receivers; that is, the reserved bandwidth is shared by all the senders.

EXAMPLE Videoconferencing

A videoconference meeting in which each of the N participants receives a video image of the other $N - 1$ participants in the meeting. In this case, each sender must have reserved bandwidth on every link that connects it to every receiver, and so no bandwidth is shared.

EXAMPLE Videostreaming

A broadcast of a layered-encoded video stream in which different receivers can receive and display video images at different qualities. For example, layered-encoded video may be organized so that the lowest quality level requires 128 kbps, the next higher quality adds an additional 256 kbps, and the premium quality adds an additional 1.152 Mbps. The amount of bandwidth that needs to be reserved in a given link is the amount required by the highest quality level that traverses the link.

Three reservation styles are defined in RSVP: wildcard filter, fixed-filter, and shared-explicit. Figure 10.6 shows a router configuration somewhere in a distribution tree for the purpose of distinguishing different reservation styles. Three senders and three receivers are attached to the router. It is assumed that packets from S1, S2, and S3 are forwarded to both output interfaces.

The *wildcard-filter (WF) style* creates a single reservation that is shared by all senders in the session. This style can be thought of as a shared pipe whose resource

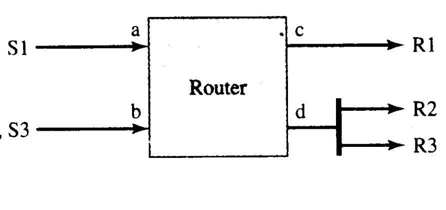


FIGURE 10.6 Example for different reservation styles.