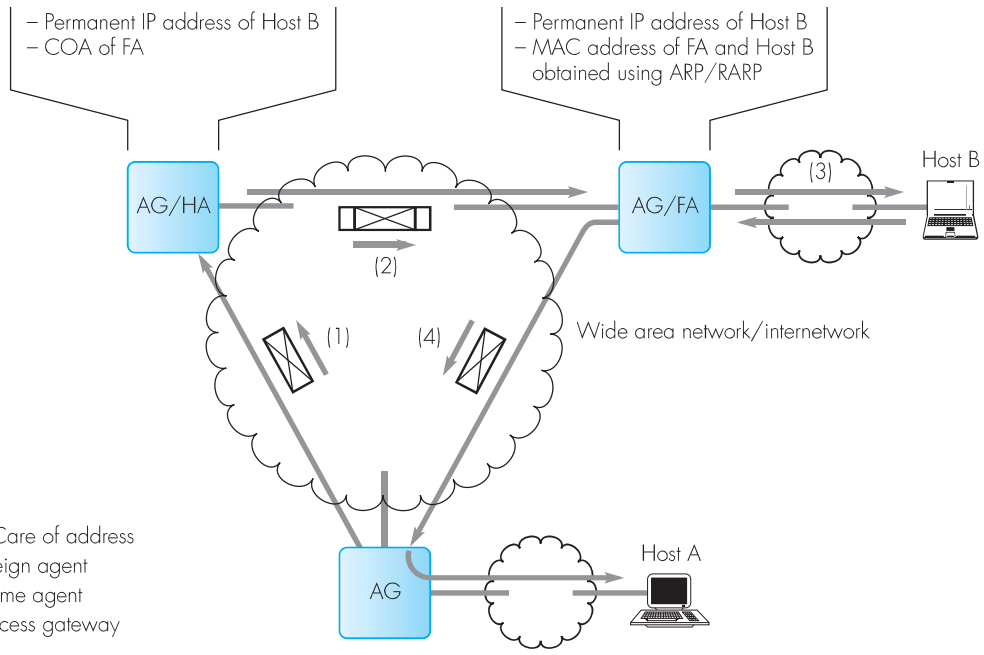


– IP packet sent from Host A to Host B using the permanent IP address of Host B in the destination address and the IP address of Host A in the source address. (1)
– HA reads the permanent IP address of Host B and, from its routing table, determines the packet must be rerouted to the COA of FA.
– HA encapsulates the packet inside a new packet with COA of FA in the destination address field. (2)
– FA receives the encapsulated packet and extracts the original packet. It then determines from its routing table and the destination address field in the packet header that the packet is for the visiting Host B.
– The FA then proceeds to broadcast the packet over the LAN in a frame with the destination MAC address of Host B. (3)
– Host B receives the packet and generates a response packet with the permanent IP address of Host B in the source address field and the IP address of Host A in the destination address field. It then sends the response packet to the FA using the MAC address of the FA.
– The FA relays the packet unchanged to the AG of Host A and from there to Host A. (4)

Figure 6.34 Mobile IP – indirect routing of packets after Host B has registered with the FA and HA.

6.7 QoS support

Congestion arises within a network when the demand for a network resource exceeds the level that is provided. For example, if a burst of packets arrive at a router (within the global internetwork) on a number of different input lines that all require the same output line, then the output line will become congested if the rate of arrival of packets is greater than the rate they can be output. To allow for this possibility, each output line has a first-in first-out

(FIFO) queue associated with it which is used to hold a defined number of packets that are awaiting output on that line. Hence, providing the burst is of a relatively short duration and the number of packets to be queued is less than the number of packet buffers available, the congestion will be transient and the only effect should be a small increase in the end-to-end transfer delay experienced by each packet using that line. In the event of a longer burst, however, then all the packet buffers may become full and, as a result, some packets will have to be discarded.

Similarly, at a network-wide level, since the Internet is a best-effort connectionless network, the global internetwork will become congested if, over a sustained period, the aggregate rate at which packets are entering the internetwork exceeds its total capacity in terms of transmission bandwidth and packet buffers. As we saw in Section 1.2.4, associated with each call is a defined set of parameters that form what is called the minimum quality of service (QoS) requirements for the call. For example, with a packet-switched network like the Internet, these include a defined minimum mean packet throughput rate and a maximum end-to-end packet transfer delay. Hence, if as a result of congestion these requirements are not met, then the quality of the call may no longer be acceptable to the user. This is the case with applications involving real-time media streams, for example, such as Internet telephony.

As we can conclude from the above, two levels of congestion control are required, one that operates at the global internetwork level and the other that operates at the router level. The aim of the first is to limit the aggregate rate at which packets are entering the global internetwork to below its maximum rate, and the aim of the second is to maximize the flow of packets through each router. In the following subsections we discuss aspects of two of the schemes that are used to perform these functions. In addition, we describe the essential features of a technique called MultiProtocol Label Switching (MPLS). This is a scheme that is used to improve IP routing performance in high-throughput core IP networks.

6.7.1 Integrated services

Most early applications of the Internet were text based and hence relatively insensitive to delay and jitter. Examples include FTP and e-mail, both of which can tolerate the added delays incurred by the use of a host-to-host retransmission control mechanism to overcome the effect of lost packets resulting from the best-effort service provided by IP. Other text-based applications, however, cannot tolerate the delay caused by retransmissions but nevertheless require minimal packet losses. Examples of this type of application are those relating to network control.

More recently, a number of interpersonal applications involving packetized speech and video were introduced. These require the packets that are generated at the source to be transferred over the Internet and played out at the destination in real time. This means that the retransmission of lost

packets is not possible and that the packet flow is particularly sensitive to lost packets and jitter. Such applications also require a guaranteed minimum bandwidth. To meet this more varied set of QoS requirements, two schemes have been researched and standardized, one called integrated services (IntServ) and the other differentiated services (DiffServ). In this section we describe aspects of the IntServ scheme while aspects of DiffServ are described in Section 6.7.2.

In both schemes, the packets relating to the different types of call/session are each allocated a different value in the precedence bits of the *type of service* field of the IP packet header. This is used by the routers within the Internet to differentiate between the packet flows relating to the different types of call. The IntServ solution defines three different classes of service:

- **guaranteed**: in this class, a specified maximum delay and jitter and an assured level of bandwidth are guaranteed. It is intended for applications involving the playout of real-time streams;
- **controlled load** (also known as predictive): in this class, no firm guarantees are provided but the flow obtains a constant level of service equivalent to that obtained with the best-effort service at light loads. Examples of applications in this class are those involving real-time streams that have the capability of adjusting the amount of real-time data that is generated to the level that is offered;
- **best-effort**: this is intended for text-based applications.

To cater for the three different types of packet flows, within each router, three separate output queues are used for each line, one for each class. In addition, appropriate control mechanisms are used to ensure the QoS requirements of each class are met. We shall first discuss a number of these as these are used in both the IntServ and the DiffServ schemes.

Token bucket filter

This is used with each of the packet flows in both the guaranteed and predictive service classes. A portion of the bandwidth of the outgoing line and an amount of buffer/queue space is reserved for the packet flow relating to each call. A control mechanism called the token bucket filter is used to enforce these allocations so that the guaranteed QoS requirements in terms of bandwidth, delay, and jitter are met.

Associated with each flow is a container called a *bucket* into which tokens are entered at a rate determined by the bandwidth requirement of the flow. The size of the bucket is the same as the maximum amount of buffer/queue space the flow may consume. A packet relating to a flow can only be transferred to the output queue if there are sufficient tokens currently in the bucket. The number of tokens required is determined by the packet length and, if sufficient tokens are currently in the bucket, the packet is queued and the corresponding number of tokens taken from the bucket. As we can deduce

from this, therefore, providing the arrival rate of packets is less than or equal to the rate of entry of tokens into the bucket, then both the agreed bandwidth and delay/jitter will be met. If the arrival rate of packets exceeds the allocated bandwidth, normally these are relegated to the best-effort queue.

Weighted fair queuing

Since the packet rate – and hence bandwidth – associated with each flow may be different, when the packets relating to each flow are queued for transmission, in order to ensure the guaranteed delay bounds are met, it is necessary to ensure that the packets relating to each flow are not delayed by the packets of other flows. Hence a queue management scheme is also required to schedule the order that queued packets are transmitted. The **weighted fair queuing** (**WFQ**) scheme performs this function.

In order to ensure the delay bounds for each flow are met, the order of transmission of packets from the queue is changed each time a new packet arrives for queuing. When a packet arrives at an incoming line of the router it is given a time-stamp. This is determined from the arrival time of the packet and its scheduled departure time, the latter computed from the bandwidth associated with the flow and the packet length. The time-stamp of the packet is then compared with the time-stamps of the packets that are currently queued and the packet with the smallest time-stamp is transmitted first. In this way the delay bounds of each flow are met.

Random early detection

The requirements of the queue management scheme used with the best-effort queue are different from those of the other two queues. As we indicated earlier, normally, a router simply discards/drops a packet if the required output queue is already full. However, as we shall describe later in Section 7.3.2, with TCP, each time a packet relating to a call/session is lost, the TCP in the source host detects this and halves its current rate of entry of new packets for the call. Since this is done by all the hosts that lose a packet, the utilization of the link bandwidth falls dramatically. This is also detected by the affected TCPs which then quickly ramp-up the rate of entry of new packets. This, in turn, often results in full queues and dropped packets occurring, again with the effect that the utilization of the available transmission bandwidth is poor. To stop this occurring, the **random early detection** (**RED**) queue management scheme is often used.

With RED, when a packet arrives for an output queue and the queue is full, instead of discarding the packet, a packet that is already in the queue is randomly selected for discarding. This has the affect that a reduced number of different applications are affected and hence the bandwidth utilization of the link is much improved.

To implement the scheme, two thresholds relating to the queue are defined: a minimum threshold (MinTH) and a maximum threshold

(MaxTH). Also, the average length (AvrLEN) of the queue is continuously monitored and used as a measure of the current level of traffic using the line. The action taken by the scheduler is determined by the current AvrLEN relative to the two thresholds as follows:

AvrLEN < MinTH:	the new packet is entered into the queue;
AvrLEN < MaxTH:	the new packet is dropped;
MaxTH < AvrLEN < MinTH:	a randomly-selected packet from the queue is dropped and the new packet is queued.

As we can deduce from the last condition, the probability of packets that are already in the queue being dropped increases as the AvrLEN increases. This has been found to give high levels of bandwidth utilization during periods of congestion.

Resource reservation protocol

With the IntServ scheme, in order to ensure the aggregate bandwidth of real-time traffic flows does not exceed that which is allocated for both the guaranteed and controlled-load traffic, the resources required for each flow (in terms of transmission bandwidth and buffer capacity) are reserved in advance of each packet flow starting. The protocol used to do this is called the **resource reservation protocol** (**RSVP**).

Because many of the new real-time applications involve multiple participants, RSVP is used to reserve resources in each router along either a unicast or a multicast path. The actual routing of the packets associated with both types of call, however, are not part of RSVP and these are carried out in the normal way using one of the routing algorithms we described earlier in Section 6.6. A selection of the messages associated with RSVP are shown in Figure 6.35(a).

Each traffic flow is identified uniquely by the combined source and destination addresses in the IP header and the port number in the UDP header. To perform a reservation, the AP in the host that wishes to set up the call/session sends a *path* message in a UDP datagram with either the unicast or multicast address of the other host(s) in the destination address field of the IP header and the port number in the UDP datagram header. The purpose of the *path* message is, firstly, to enable each router along the path/route followed by the packet to create an entry in a table known as the **path-state table** and, secondly, to gather information about the resources that are currently available in each router along the path. The entry in the path-state table includes the flow identifier, a specification of the required parameters associated with the call/session – known as the traffic specification or Tspec – and the (IP) address of the router from which it received the *path* message.

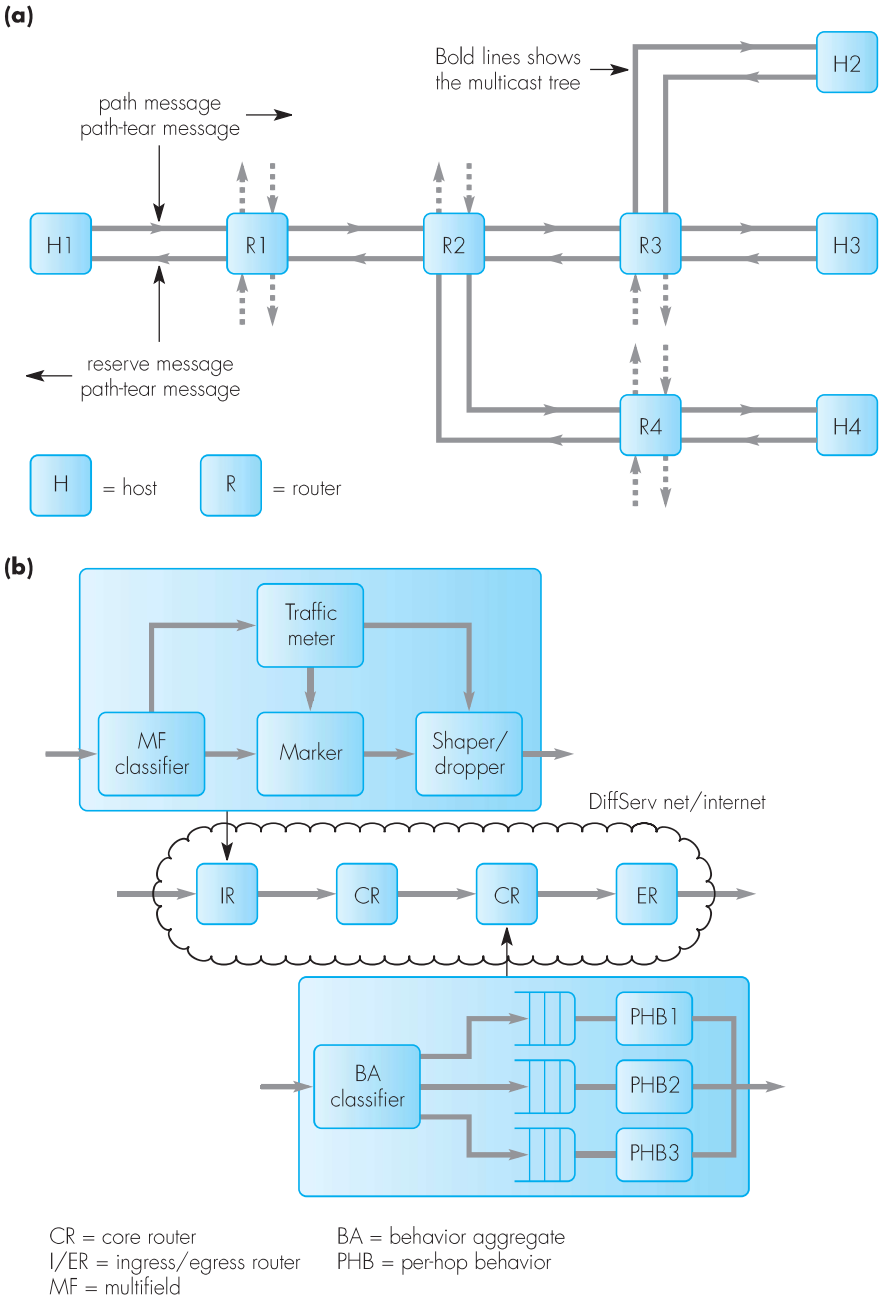


Figure 6.35 QoS support mechanisms: (a) RSVP principles; (b) DiffServ architecture.

On receipt of the *path* message, the AP (identified by the port number) in the destination host(s) uses the resource levels reported by each router to determine which type of call – guaranteed, controlled-load, or neither – the path can support. Assuming the call can be accepted, the AP in each destination host then returns a *reserve* message (containing the Tspec for the call) to the router from which it received the *path* message. If the router still has the necessary resources, it reserves these for the flow (the resources may be different in each direction), makes an entry in the path-state table of the address of the router which sent it the message, and then sends the (*reserve*) message to the next router along the path. If the resources are not available, then a *path-tear* message is returned along the forward and return paths in order to release any resources that have been reserved and delete the entry in the path-state table. This procedure is repeated by each router along the path back to the source which, on receipt of the *reserve* message, proceeds with the call.

The path associated with each call/session may change during the lifetime of a call, for example due to a router going down. To allow for this occurring, associated with each entry in the path-state table kept by each router is a timer called the *cleanup timer* and, should this expire, the entry is deleted. The timer is restarted each time a *path* message relating to the call is received. At periodic intervals – less than the cleanup timeout period – the source host sends a new *path* message, which is acted upon by each router in the same way as the first *path* message. Hence should the new path not include a particular router, its cleanup timer will expire and the related path-state information be deleted. This mode of operation is known therefore as *soft-state* since it may change during a call.

Finally, on completion of the call/session, the AP that set up the call sends out a *path-tear* message which results in the entry in the path-table held by each router along the path(s) being deleted. The RFCs associated with RSVP and the other IntServ procedures are in RFC 2205–2216.

The major disadvantage of RSVP is that state information is retained by each router for each call/flow. Although this may be acceptable in a company intranet, for example, in a backbone router of the Internet the tables used for this purpose can be very large. Hence with the very high bit rates that are used, the heavy overheads per call can be unacceptably high. It is for this reason that the alternative DiffServ scheme was developed.

6.7.2 Differentiated services

Using the DiffServ approach, individual flows are not identified and instead the individual flows in each service class are aggregated together. Flows are then treated on a per-class basis rather than a per-flow basis. The general architecture used with a DiffServ network is shown in Figure 6.35(b).

The incoming packet flows relating to individual calls – referred to as microflows – are classified by the router/gateway at the edge of the DiffServ-

compliant net/internet into one of the defined service/traffic classes by examining selected fields in the various headers in the packet. Within the DiffServ network the *type of service* (*TOS*) field in the IP packet header is replaced by a new field called the *differentiated services* (*DS*) *field*. As we saw earlier in Section 6.2, this is an 8-bit field, although currently only six bits are used for the DS field. The six bits form what is called the (**DS**) **packet codepoint** (**DSCP**), which is used to enable each router to determine the traffic class to which the packet data relates and the output queue into which the packet should be put. The queue management/scheduling procedure relating to each queue is called a **per-hop behavior** (**PHB**) and, since this applies to an aggregated set of packet flows, a PHB is said to be applied to a **behavior aggregate** (**BA**).

Within the DiffServ network, a defined level of resources in terms of the buffer space within each router and the bandwidth of each output line is allocated to each traffic class using, for example, a token bucket filter. As each packet arrives at the ingress router it is first classified as belonging to a particular traffic class. In some instances, however, as we shall see later, the actual classification is also a function of how well the microflow (to which the packet relates) is conforming to the agreed traffic profile for the flow. This is determined by the **traffic meter module** and, based on the level of conformance, the traffic meter informs the **marker module** whether the packet should be marked with a low, medium, or high drop precedence. In addition, the traffic meter also informs the **shaper/dropper** module of this and the latter decides whether the packet should be dropped or allowed into the network.

Normally, real-time microflows with hard QoS guarantees are placed in the highest-priority traffic class. Typically, the traffic meter for this type of stream is a token bucket filter with a defined rate and bucket depth. Hence if an arriving packet relating to a flow in this class is deemed to be out-of-profile, the traffic meter informs the shaper/dropper module of this. The latter then either drops the packet or relegates it to the best-effort class by setting all six DSCP bits to zero.

Once within the network, as each packet arrives at a core router, the **BA classifier** first determines to which traffic class the packet belongs and hence to which output queue the packet should be transferred. Each queue is then serviced using an appropriate PHB. Currently, two PHBs have been defined: **expedited forwarding** (**EF**) and **assured forwarding** (**AF**). The EF PHB is similar to the guaranteed service class associated with IntServ and hence has the highest priority. The PHB used with this is based on a queue scheduling procedure such as weighted fair queuing.

The AF PHB has four ordered traffic classes associated with it, each of which has three drop procedure levels: low, medium and high. Should congestion arise, this is used together with the traffic class to determine which packet(s) should be dropped. The PHB used in this case, therefore, is based on a queue scheduling procedure such as random early discard. The RFCs associated with DiffServ are in RFC 2474–5.

6.7.3 MPLS

Before describing how a packet is forwarded in an MPLS network it is helpful to first review how a packet is forwarded in a standard IP-based network. A schematic diagram illustrating the forwarding of packets by a router in a conventional IP-based network is shown in Figure 6.36.

On receipt of a packet from one of the input queue interfaces, the router first reads the IP destination address in the packet header and proceeds to determine the netid by using the related address mask. It then uses the netid to determine from its forwarding table the outgoing port/line that is on the shortest path route to this netid. In addition, the ToS field in the packet header is passed to the **packet classifier**, which uses this to determine the queuing and scheduling rules to be applied to the packet. This information is then passed to the output queue interface unit and this determines the output queue relating to the output port/line that should be used to forward the packet to the next-hop router.

As we have just seen, with the introduction of integrated services three different classes of service are supported, each of which has a separate output queue associated with it. A queuing and scheduling rule is then used for each class and this is then passed to the output interface unit to ensure the QoS requirements of each class are met.

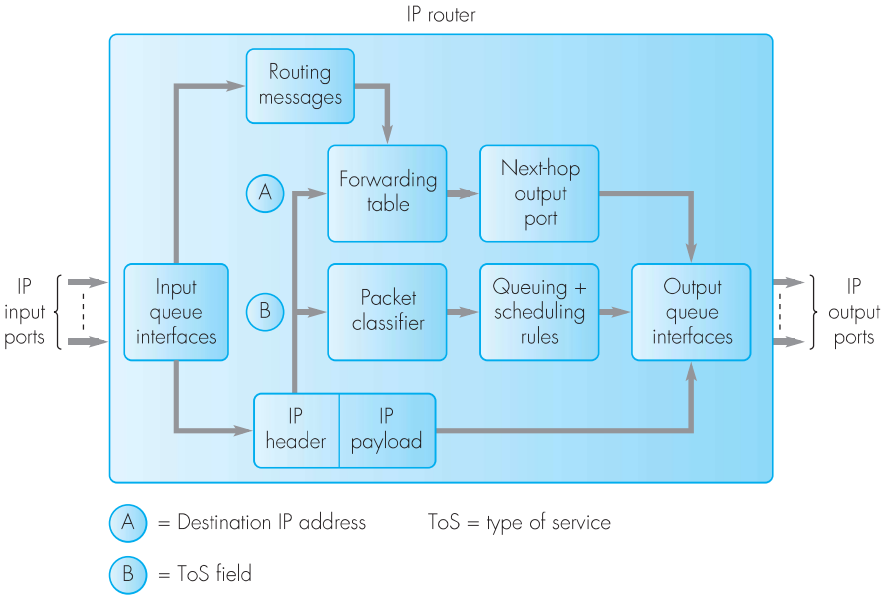


Figure 6.36 Packet forwarding in a conventional IP router.

Using differentiated services – DiffServ – individual flows are not identified and instead the individual flows in each service class are aggregated together. Flows are then treated on a per-class basis rather than a per-flow basis.

As we can conclude from this summary, the efficiency of the forwarding mechanism used within each router becomes all important and especially so in the routers that make up the high-throughput core networks within the Internet. Hence, as we indicated in Section 6.7, **MultiProtocol Label Switching** (**MPLS**) is a technique that is now widely used in the routers within these core networks to obtain higher forwarding rates of packets. MPLS is defined in RFC 3031.

As we saw earlier in Section 6.6.1, the Internet is composed of a large number of autonomous systems (ASs). Associated with each AS is a backbone area network and it is within these that MPLS is most widely deployed. In order to see where the various QoS mechanisms are used, we shall use the example network architecture shown in Figure 6.37. Typically, DiffServ is used for traffic aggregation and the differentiation of the packet flows with different classes of service. MPLS is then used for traffic aggregation and load balancing which, collectively, are referred to as **traffic engineering** (**TE**).

The access gateway/router classifies all packet flows entering the backbone area network. This is based on the IP source and destination addresses and the TCP/UDP source and destination port numbers. Normally, the

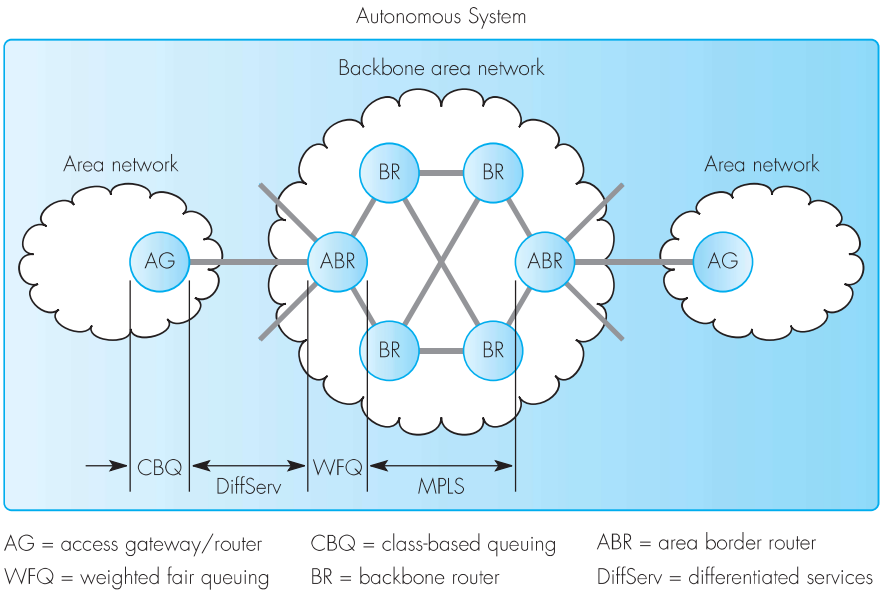


Figure 6.37 MPLS example network architecture with related QoS mechanisms.

access gateway (AG) uses **class-based queuing** (**CBQ**) for this function since it must distinguish between a large number of applications and users. Typically, these are implemented as a set of software queues.

Once the traffic flows have been classified and queued, the AG schedules and separates – known as **shaping** – each flow. Shaping is necessary because there may be many different classes of flows but, once they are within the backbone area network, the area border router segregates the flows – known as **grooming** – into a relatively small number of queues, for example, real time, non-real time and best effort. The AG also controls the allocation of DiffServ values for the different packet flows. The main aim of DiffServ is to achieve the differentiation of packet flows using a short classification key. This, in turn, simplifies the design of very high throughput routers. Normally, the DiffServ values are marked by hosts – in the DS/ToS field in the IP packet header and then used by the AG. In addition, the AG may assign DiffServ values as a means of classifying network management and signaling messages which, as we shall see, are used by all the routers in the AS backbone.

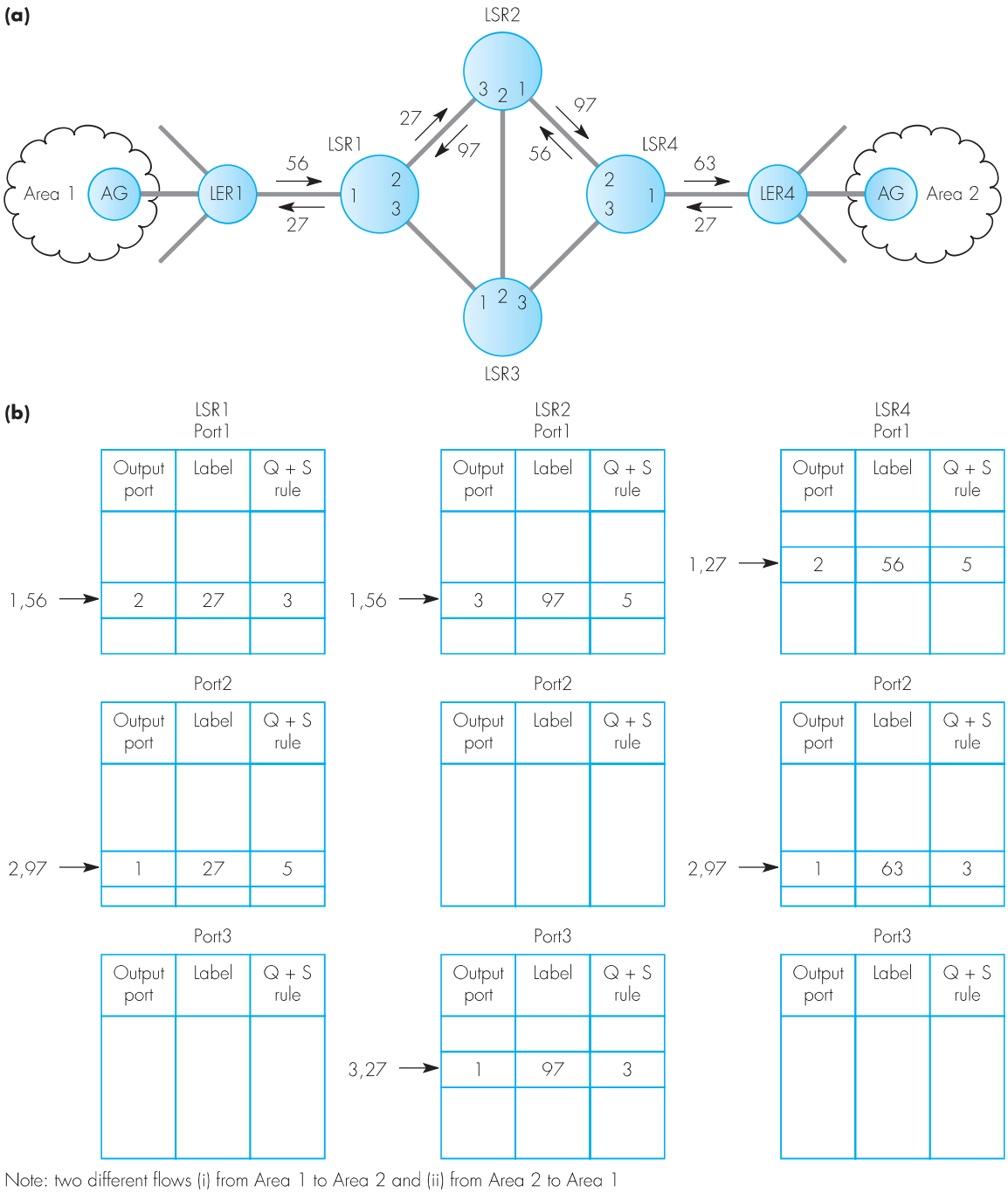
Area border routers

Each area border router (ABR) receives packet flows from a set of access gateways. The ABR also serves as a **label edge router** (**LER**) since they append and remove MPLS headers to/from the head of IP packets. In contrast to an IP destination address that is used to forward a packet over the entire shortest-path-route through the network, an MPLS label has only per-hop significance. This is done so that an MPLS packet can be forced to follow a specific path through the network by using a sequence of labels – and hence keys into the related forwarding tables – to the intended destination ABR/LER.

This is an important aspect of traffic engineering since, when all shortest-path routes are being used, some routers can become overloaded and form what are called hotspots. Hence to overcome this, specific routes can be selected that bypass potential **hotspots**. This is called **load balancing**. The backbone routers within the AS then simply read the MPLS label and forward the packet on the new output port/line with a new label at very high throughput rates. This mode of operation is known as **label switching** and a simple example illustrating this is shown in Figure 6.38.

Normally, all the router functions in backbone routers are implemented in hardware. For example, the set of queues associated with each output port/line are implemented in hardware and, typically, use a weighted fair queuing (WFQ) scheduling procedure. The queues are based on a relatively small number of different network classes of service. These include real time packet flows, non-real time, best effort, signaling, and network management. As a result, the packet flows relating to the different user applications are combined and assigned to a particular network class of service.

As we indicated earlier, the primary role of MPLS is for traffic engineering and, more generally, optimizing the use of network resources. To achieve



Note: two different flows (i) from Area 1 to Area 2 and (ii) from Area 2 to Area 1

Figure 6.38 Label switching schematic.

this, all packet flows are constrained to follow a specific **label switched path** (**LSP**) through the backbone network. All the flows that are grouped together with the same label are said to belong to the same **forward equivalent class** (**FEC**) since all the packets are then forwarded in the same way. At the destination the ABR/LER terminates the LSP by removing the MPLS label. The destination IP address is then used to forward the packet to the addressed access gateway/network. Once in the access gateway, the original classification is restored based on the IP source and destination addresses and the TCP/UDP source and destination port numbers. The IP destination address is then used to forward the packet to the intended recipient host in the normal way. A schematic diagram illustrating the functionality of an ABR/LER is shown in Figure 6.39(a).

On receipt of an IP packet, the IP destination address in the packet header is used first to obtain the output port/line from the forwarding table. This is then passed, together with the DiffServ value in the DS field, to the LSP table module. This then uses these to select a new next-hop label for this flow and the queuing and scheduling rules to be used with it. The MPLS header is then created and appended to the IP packet. This is then passed to the output queue interface module ready for forwarding using the selected output port and its associated queuing and scheduling rules.

As we can see in Figure 6.39(b), the length of the *label* field in the MPLS header is 20 bits which is sufficient to identify a significant number of unique next-hop labels and output port/line identifiers. The 3-bit *CoS* field is then used to select a specific queuing and scheduling rule for this flow. The 8-bit *time to live* field is used to detect and discard packets that are looping. It is set to an initial value by the ingress ABR/LER and decremented by each BR/LSR it visits and, should it become zero, the packet is discarded.

The *S bit* is used primarily to support multiple virtual private networks. A service provider can separate out the packet flows relating to each virtual private network by stacking a number of 32-bit MPLS labels at the head each original IP packet and setting the S bit to 0. Each BR/LSR along a path then pops the top entry from the stack and uses the 32-bit value to forward the packet. Then, when there is only a single entry in the stack left, the S bit is set to 1. Hence when only a single MPLS network is present, the S bit is set to 1.

Normally, the lines linking the BRs/LSRs within the backbone network are derived from SDH/SONET lines which we described in Section 2.2.5. Each MPLS packet is then transmitted using the PPP protocol we described in Section 2.6.4. Also, since all routing within the core backbone network is carried out using the MPLS label, the payload of the MPLS packet can relate to any protocol – IP, IPX, etc. This is the origin of the term MultiProtocol in the title.

Signaling

As we indicated in Section 6.6.4, within the area backbone network of an AS the routing protocol is OSPF. This is used to determine both the topology of

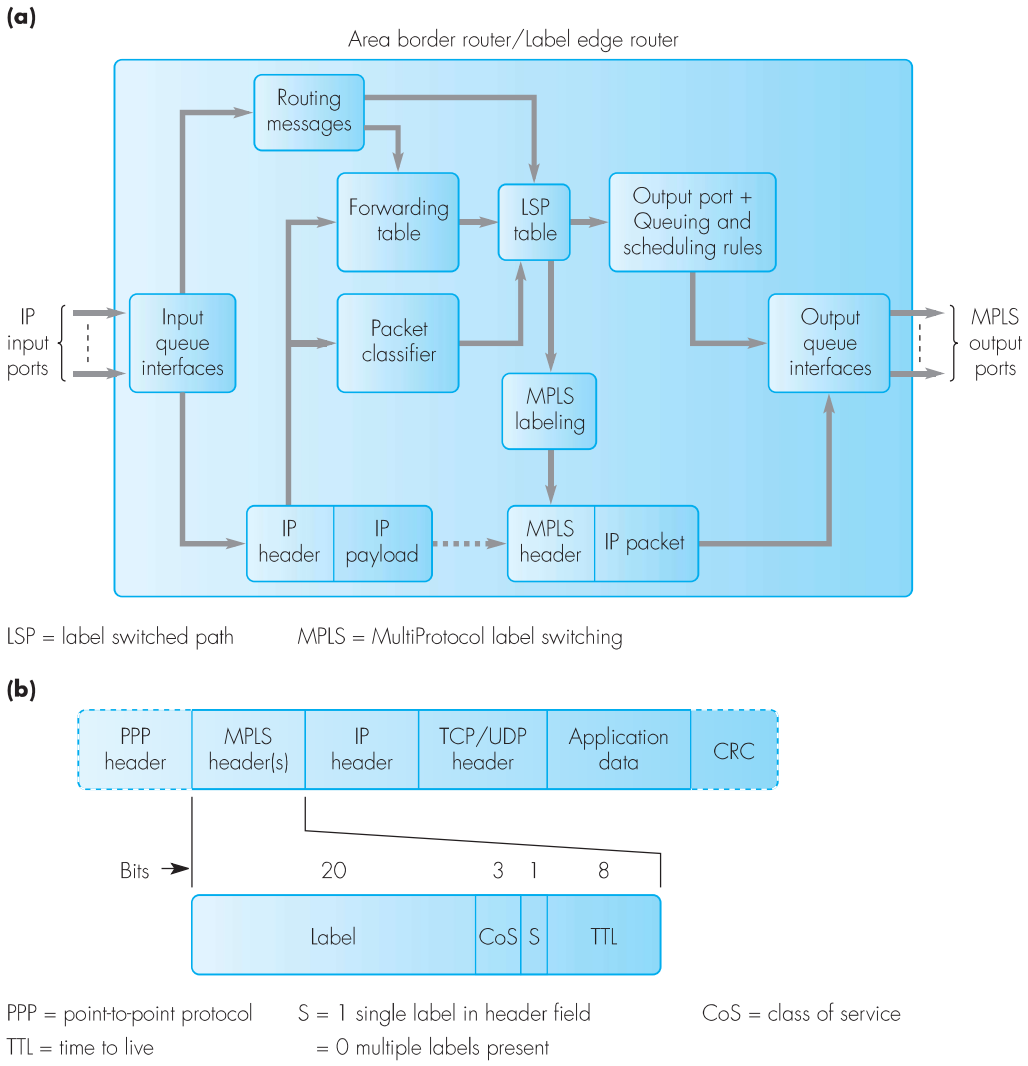


Figure 6.39 MPLS: (a) area border router/label edge router schematic; (b) MPLS header details.

the network and the shortest-path route assignments for each router. As we indicated earlier, however, with MPLS the route assignments are simply per-hop identifiers. Hence with MPLS it is necessary to have what is called a signaling protocol to determine a set of label-switched paths through the network and to assign a set of MPLS labels that identify uniquely each path. In practice, there are a number of ways of doing this so we shall study just one approach as an example.

In what is called basic MPLS, every label switched router (LSR) within the backbone network is assigned an IP address and runs the OSPF protocol to establish the topology of the backbone. However, the shortest-path routes are not used directly and instead an additional protocol called the **label distribution protocol** (**LDP**) is used to create a complete set of label-switched paths (LSPs) between each ingress label edge router (LER) and egress LER. Unique labels are then assigned for each hop in each of the label-switched paths. The use of OSPF in this way has the advantage that routing changes can be readily incorporated. The disadvantage is that the set of shortest path routes computed may contain hotspots as we indicated earlier. Hence the actual routes used are sometimes different from the computed shortest-path routes. To illustrate this, consider the simple backbone topology shown in Figure 6.40.

As we can see, this involves two access networks – areas A and B – that both want to communicate with area C. Clearly, the shortest path route from A to C is the same as that from B to C. Hence if these are used, the link from LSR3 to LER5 may become overloaded even though there is an alternative route through LSR4. Hence to utilize the available bandwidth in a more efficient way, the path from B to C could be LER2 – LSR3 – LSR4 – LER5 since although this route is longer, the transmission bandwidth is more distributed. This approach is called **constraint routed LSPs** (**CR-LSPs**) and the modified protocol is called the **constraint routed label distribution protocol** (**CR-LDP**).

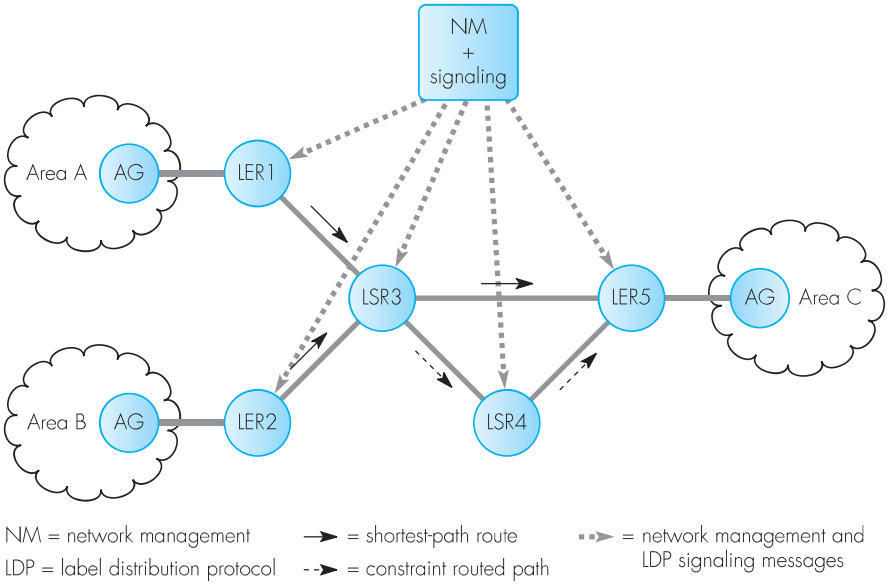


Figure 6.40 Constraint routed LSPs as used with CR-LDP.

Finally, as we show in the figure, the signaling component of the network management station computes the label switched paths between each pair of LERs using CR analysis. Each router within the backbone runs LDP and the network management station then uses LDP to download the contents of the LSP table held within each router. All the signaling and network management messages are downloaded using the same set of lines/links that are used to carry MPLS packets. As we can deduce from this, all the paths allocated are under the control of the network operator.

6.8 IPv6

Until the mid-1990s the Internet was used primarily by universities, government agencies, research establishments and some sectors of industry. Since that time, however, it has gone through unprecedented growth owing to a large extent to the rapid rise in interest in the use of the World Wide Web. As a result, most schools, colleges, and many homes now have PCs with connections to the Internet. These are now being used, in addition to Web access, to exploit the range of other applications that are supported by the Internet. Moreover, this growth is predicted to increase even faster as new applications emerge: for example, the widespread use of hybrid mobile phones/computers with Internet interfaces, television set-top boxes with integral Web browsers, plus a potentially vast number of consumer products – such as meters, household appliances, office equipment, and so on – many of which may have an Internet interface.

Although the introduction of CIDR has extended considerably the usable address space of IPv4, the IETF has already defined and is using a new version of IP which has a number of features that have been introduced to meet the predicted growth levels. This is known as **IP version 6** (**IPv6**) or sometimes **IP next generation** (**IPng**). It is defined in RFCs 1883–7 and a number of supporting RFCs. In addition to providing a large increase in the number of IP addresses, the IETF has taken the opportunity to correct some of the deficiencies associated with IPv4 and to provide a number of other features. The main new features of IPv6 are:

- a much increased address space from 32 bits to 128 bits;
- hierarchical addresses to reduce the size of the routing tables associated with the routers in the core backbone network;
- a simplified header to enable routers and gateways to process and route packets faster;
- the introduction of improved security and data integrity features including authentication and encryption;
- an autoconfiguration facility that enables a host to obtain an IP address via the network without human intervention;

- harder quality-of-service guarantees by means of the preferential treatment by routers of the packets associated with interactive and multimedia applications relative to those relating to traditional applications such as e-mail and file transfers;
- support for mobile computing by the use of autoconfiguration to obtain an IP address dynamically via the network for the duration of a call/session.

Although many of the above features require some radical changes to IPv4 – for example a different address structure and datagram/packet format – in terms of the protocols that are used within the expanded global internetwork, these operate in much the same way as the current IPv4 protocols. For example, the LS-SPF (OSPF) routing algorithm we described in Section 6.5.4 is used as the standard interior gateway protocol (IGP) for IPv6 except, of course, that 128-bit addresses are used in the link-state and routing tables. There is also an updated version of the RIP – based on the distance vector routing algorithm we described in Section 6.5.3 – called **RIPng**. Similarly, at the backbone level, the border gateway protocol (BGP) we described in Section 6.6.5 (including the CIDR we described in Section 6.4.3) is used but with extensions to allow reachability information based on IPv6 hierarchical addresses to be exchanged. Hence in the remainder of this section we limit our discussion to a selection of the new features that are used.

6.8.1 Datagram format

In relation to the IPv4 datagram/packet header, a number of fields have been dropped and others have been made optional. The result is a basic/main header of 40 bytes, the contents and format of which are shown in Figure 6.41(a). The use of each field is as follows:

Version

This is set to 6 to enable routers to discriminate IPv6 packets from IPv4 packets. It is envisaged that both will need to coexist for many years.

Traffic class

This field plays a similar role to the *ToS* byte in the IPv4 header. It allows the source IP to allocate a different priority to packets relating to, say, multimedia applications involving real-time streams from those relating to traditional applications. It contains a 4-bit priority field and hence 16 priorities are possible; the higher the priority value, the higher the packet priority. Values in the range 0–7 are for packets relating to applications for which best-effort delivery is acceptable; for example network news (1) and FTP(4). Both these applications are less sensitive to delay and delay variation (jitter) than applications involving real-time media but FTP is more sensitive to packet loss than network news. Values in the range 8–15 are for packets containing real-time streams such as audio and video. Typically, such streams/packets are

more sensitive to delay and jitter than the packets in the first category and hence should be transmitted before them during periods of congestion. Also, within the second category, packets containing compressed video are more sensitive to packet loss than packets containing just audio and hence are given a higher priority.

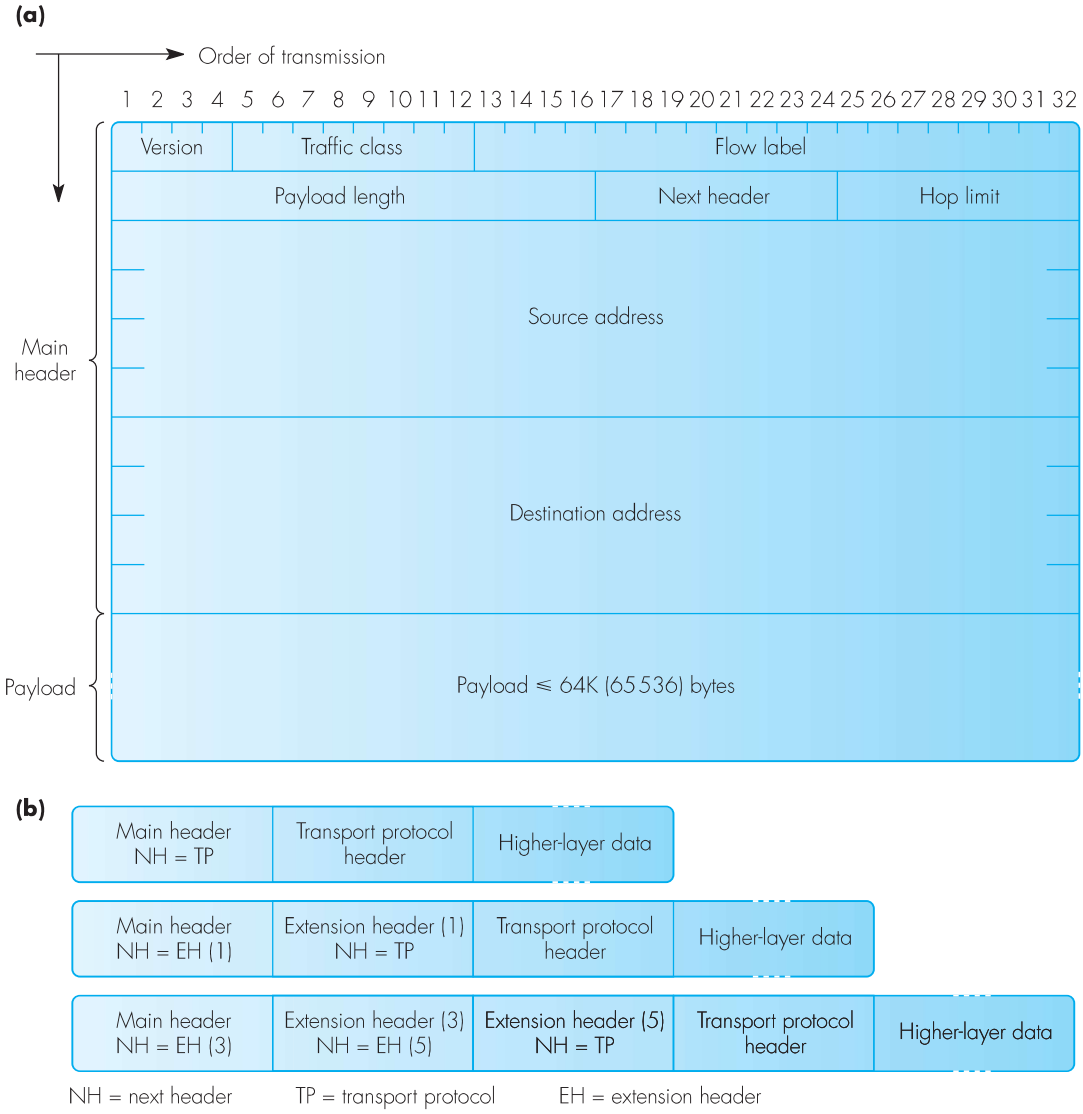


Figure 6.41 IPv6: (a) main header fields and format; (b) position and order of extension headers.

Flow label

This is a new field and is closely linked to the *traffic class* field. It is set to zero in best-effort packets and, in packets in the second category, it is used to enable a router to identify the individual packets relating to the same call/session. As we saw earlier in Section 6.7.1, one approach to handling packets containing real-time streams is to reserve resources – for example transmission bandwidth – for such calls in advance of sending the packets relating to the call. During the reservation procedure, the call is allocated a *flow label* by the source. Also, each router along the reserved path keeps a record of this, together with the source and destination IP addresses, in a table. Routers then use the combined *flow label* and source IP address present in each packet header to relate the packet to a specific call/flow. The related routing information is then retrieved from the routing table and this is used, together with the *traffic class* value, to ensure the QoS requirements of the call/flow are met during the forwarding procedure.

Payload length

This indicates the number of bytes that follow the basic 40-byte header in the datagram. The minimum length of a basic datagram is 536 bytes and the maximum length is 64K bytes. The *payload length* is slightly different from the *total length* field used in the header of an IPv4 datagram since, as we explained in Section 6.2, the *total length* includes the number of bytes in the datagram header.

Next header

As we show in Figure 6.41(b), a basic IPv6 datagram comprises a main header followed by the header of the peer transport layer protocol (TCP/UDP) and, where appropriate, the data relating to the higher layers. With a basic datagram, therefore, the *next header* field indicates the type of transport layer protocol (header) that follows the basic header. If required, however, a number of what are called ***extension headers*** can be inserted between the main header and the transport protocol header. Currently, there are six types of extension header defined and, when present, each extension header starts with a new *next header* field which indicates the type of header that follows. The *next header* field in the last extension header always indicates the type of transport protocol header that follows. Thus, the *next header* field in either the main header or the last extension header plays the same role as the *protocol* field in an IPv4 datagram header.

Hop limit

This is similar to the *time-to-live* parameter in an IPv4 header except the value is a hop count instead of a time. In practice, as we explained in Section 6.2, most IPv4 routers also use this field as a hop count so the change in the field's name simply reflects this. The initial value in the *hop limit* field is set by the source and is decremented by 1 each time the packet/datagram is forwarded. The packet is discarded if the value is decremented to zero.

Source address and destination address

As we indicated earlier, these are 128-bit addresses that are used to identify the source of the datagram and the intended recipient. In most cases this will be the destination host but, as we shall explain later, it might be the next router along a path if source routing is being used. Unlike IPv4 addresses, an IPv6 address is assigned to the (physical) interface, not to the host or router. Hence in the case of routers (which have multiple interfaces) these are identified using any of the assigned interface addresses.

6.8.2 Address structure

As we showed in Figure 6.19, the various networks and internetworks that make up the global Internet are interconnected in a hierarchical way with the access networks at the lowest level in the hierarchy and the global backbone network at the highest level. However, the lack of structure in the netid part of IPv4 addresses means that the number of entries in the routing tables held by each gateway/router increases with increasing height in the hierarchy. At the lowest level, most access gateways associated with a single site LAN have a single netid, while at the highest level, most backbone routers/gateways have a routing table containing many thousands of netids.

In contrast, the addresses associated with telephone networks are hierarchical with, for example, a country, region, and exchange code preceding the local number. This has a significant impact on the size of the routing tables held by the switches since, at a particular level, all calls with the same preceding code are routed in the same way. This is known as **address aggregation**.

As we explained earlier in Section 6.4.3, classless inter-domain routing is a way of introducing a similar structure with IPv4 addresses and reduces considerably the size of the routing tables held by the routers/gateways in the global backbone. From the outset, therefore, IPv6 addresses are hierarchical. Unlike telephone numbers, however, the hierarchy is not constrained just to a geographical breakdown. The large address space available means that a number of alternative formats can be used. For example, to help interworking with existing IPv4 hosts and routers, there is a format that allows IPv4 addresses to be embedded into an IPv6 address. Also, since the majority of access networks are now Internet service provider (ISP) networks, there is a format that allows large blocks of addresses to be allocated to individual providers. The particular format being used is determined by the first set of bits in the address. This is known as the **prefix format** (**PF**) and a list of the prefixes that have been assigned – together with their usage – is given in Figure 6.42(a).

Unicast addresses

As we can see, addresses starting with a prefix of 0000 0000 are used to carry existing IPv4 addresses. There are two types, the formats of which are shown

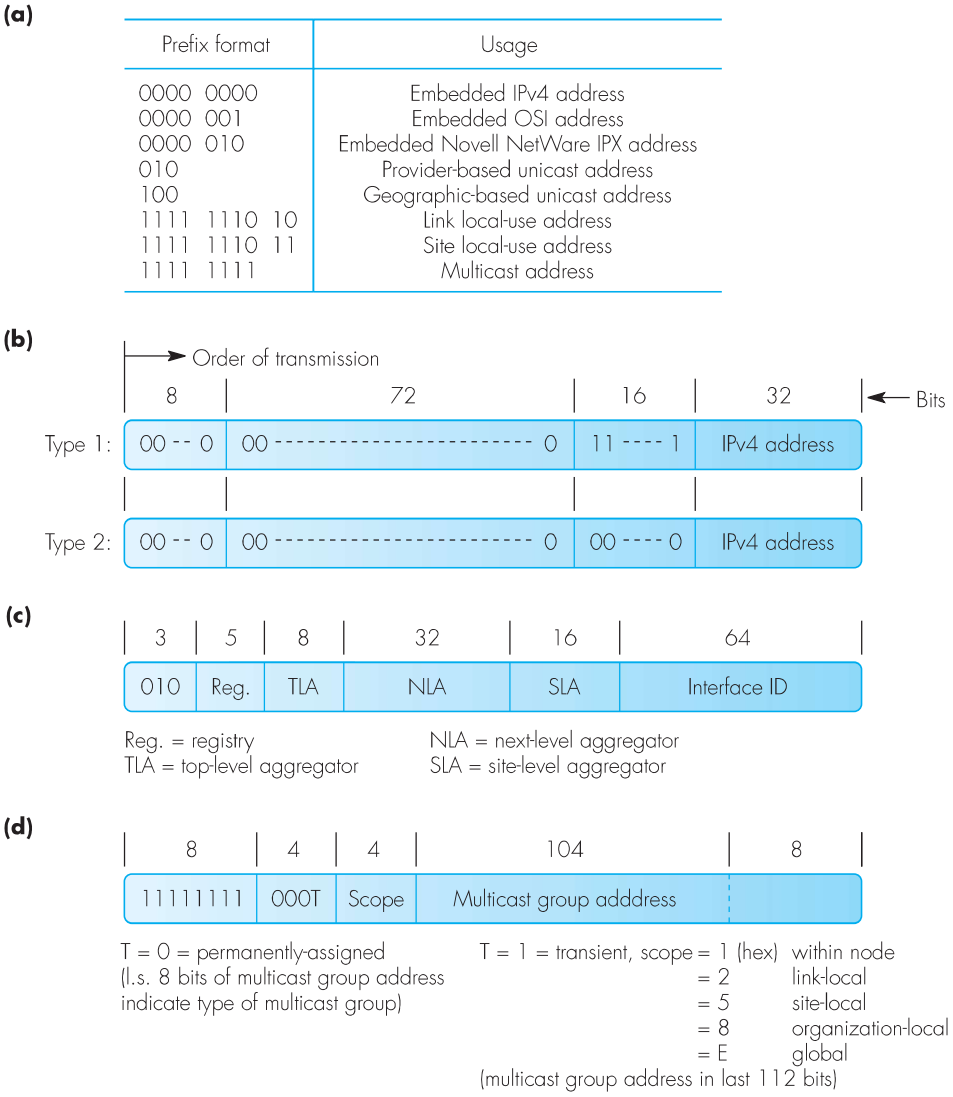


Figure 6.42 IPv6 addresses: (a) prefix formats and their use; (b) IPv4 address types; (c) provider-based unicast address format; (d) multicast address format.

in Figure 6.42(b). As we shall expand upon in Section 6.9, a common requirement during the transition from IPv4 to IPv6 is to tunnel the IPv6 packets being generated by the two communicating IPv6 hosts – often written **V6 hosts** – over an existing IPv4 network/internetwork. Hence to simplify the routing of the IPv4 packet – containing the IPv6 packet within it – the IPv6

address contains the IPv4 address of the destination gateway embedded within it. The second type is to enable a V4 host to communicate with a V6 host. The IPv4 address of the V4 host is then preceded by 96 zeros. In addition, two addresses in this category are reserved for other uses. An address comprising all zeros indicates there is no address present. As we shall expand upon in Section 6.8.4, an example of its use is for the source address in a packet relating to the autoconfiguration procedure. An address with a single binary 1 in the least significant bit position is reserved for the loopback address used during the test procedure of a host protocol stack.

The OSI and NetWare address prefixes have been defined to enable a host connected to one of these networks to communicate directly with a V6 host. The most widely used is the provider-based prefix as this format reflects the current structure of the Internet. A typical format of this type of address is shown in Figure 6.42(c). As we saw in Figure 6.19, the core backbone of the Internet consists of a number of very high bandwidth lines that interconnect the various continental backbones together. The routers that perform this function are owned by companies known as **top-level aggregators** (**TLAs**). Each TLA is allocated a large block of addresses by what is called a **registry**, the identity of which is in the field immediately following the 010 prefix. Examples include the North American registry, the European registry, the Asia and Pacific registry, and so on.

From their allocation, the TLAs allocate blocks of addresses to large Internet service providers and global enterprises. These are known as **next-level aggregators** (**NLAs**) and, in the context of Figure 6.19, operate at the continental backbone and national and regional levels. The various NLAs allocate both single addresses to individual subscribers and blocks of addresses to large business customers. The latter are known as **site-level aggregators** (**SLAs**) and include ISPs that operate at the regional and national levels. The 64-bit **interface ID** is divided locally into a fixed subnetid part and a hostid part. Typically, the latter is the 48-bit MAC address of the host and hence 16 bits are available for subnetting.

As we can deduce from Figure 6.42(c), the use of hierarchical addresses means that each router in the hierarchy can quickly determine whether a packet should be routed to a higher-level router or to another router at the same level simply by examining the appropriate prefix. Also, the routers at each level can route packets directly using the related prefix. The same overall description applies to the processing of geographic-based addresses.

As their names imply, the two types of **local-use addresses** are for local use only and have no meaning in the context of the global Internet. As we shall expand upon in Section 6.8.4, *link local-use addresses* are used in the autoconfiguration procedure followed by hosts to obtain an IPv6 address from a local router. The router only replies to the host on the same link the request was received and hence this type of packet is not forwarded beyond the router.

The *site local-use addresses* are used, for example, by organizations that are not currently connected to the Internet but wish to utilize the technology

associated with it. Normally, the 64-bit interface ID part is subdivided and used for routing purposes within the organization. In this way, should the organization wish to be connected to the Internet at a later date, it is only necessary to change the site local-use prefix with the allocated subscriber prefix.

Multicast addresses

The format of an IPv6 multicast address is shown in Figure 6.42(d). As we can see, following the multicast prefix are two additional fields that have been introduced to limit the geographic scope of the related multicast operation. The first is known as the *flags* field and is used to indicate whether the multicast is a permanently-assigned (reserved) address (0000) or a temporary (transient) address (0001). In the case of a permanently-assigned address, the least significant 8 bits of the *multicast group address* field identify the type of the multicast operation, while for a transient address, the full 112 bits identify the multicast group address.

In both cases, the 4-bit *scope* field defines the geographic scope of the multicast packet. The various alternatives are identified in the figure and mrouters use this field to determine whether the (multicast) packet should be forwarded further or discarded.

Anycast addresses

In addition to unicast and multicast addresses, a new address type known as an ***anycast group address*** has been defined. These are allocated from the unicast address space and are indistinguishable from a unicast address. With an anycast address, however, a group of hosts or routers can all have the same (anycast) address. A common requirement in a number of applications is for a host or router to send a packet to any one of a group of hosts or routers, all of which provide the same service. For example, a group of servers distributed around a network may all contain the same database. Hence in order to avoid all clients needing to know the unique address of its nearest server, all the servers can be members of the same anycast group and hence have the same address. In this way, when a client makes a request, it uses the assigned anycast address of the group and the request will automatically be received by its nearest server. Similarly, if a single network/internetwork has a number of gateway routers associated with it, they can all be allocated the same anycast address. As a result, the shortest-path routes from all other networks/internetworks will automatically use the gateway nearest to them.

In order to perform the routing function, although an anycast address has the same format as a unicast address, when an anycast address is assigned to a group of hosts or routers, it is necessary for each host/router to be explicitly informed – by network management for example – that it is a member of an anycast group. In addition, each is informed of the common part of the address prefixes which collectively identify the topological region in which all the hosts/routers reside. Within this region, all the routers then maintain a separate entry for each member of the group in its routing table.

Address representation

A different form of representation of IPv6 addresses has been defined. Instead of each 8-bit group being represented as a decimal number (with a dot/period between them), groups of 16 bits are used. Each 16-bit group is then represented in its hexadecimal form with a colon between each group. An example of an IPv6 address is:

FEDC:BA98:7654:3210:0000:0000:0000:0089

In addition, a number of abbreviations have been defined:

- One or more consecutive groups of all zeros can be replaced by a pair of colons.
- Leading zeros in a group can be omitted.

Hence the preceding address can also be written as:

FEDC:BA98:7654:3210::89

Also, for the two IPv4 embedded address types, the actual IPv4 address can remain in its dotted decimal form. Hence assuming a dotted decimal address of 15.10.0.6, the two embedded forms are:

:: 150.10.0.6 (IPv4 host address)
:: FFFF:150.10.0.6 (IPv4 tunnel address)

Example 6.6

Derive the hexadecimal form of representation of the following link-local multicast addresses:

(i) a permanently-assigned multicast group address of 67,

(ii) a transient multicast group address of 317.

Answer:

The formats of the two types of multicast addresses were shown in Figure 6.42(d). Hence:

The most significant 16 bits are FF02 = permanently-assigned, link-local
and FF12 = transient, link-local

(i) A permanently-assigned multicast group address of 67 = 0043 (hex)
Hence IPv6 address = FF02 :: 67

(ii) A transient multicast group address of 317 = 013D (hex)
Hence IPv6 address = FF12 :: 13D

6.8.3 Extension headers

As we have just indicated, if required, a number of extension headers can be added to the main header to convey additional information, either to the routers visited along the path followed or to the destination host. The six types of extension header currently defined are:

- **hop-by-hop options**: information for the routers visited along a path;
- **routing**: list of routers relating to source routing information;
- **fragment**: information to enable the destination to reassemble a fragmented message;
- **authentication**: information to enable the destination to verify the identity of the source;
- **encapsulating security payload**: information to enable the destination to decrypt the payload contents;
- **destination options**: optional information for use by the destination.

The two options headers can contain a variable number of option fields, each possibly of a variable length. For these, each option field is encoded using a **type-length-value** (**TLV**) format. The *type* and *length* are both single bytes and indicate the option type and its length (in bytes) respectively. The *value* is then found in the following number of bytes indicated by the *length*. The option type identifiers have been chosen so that the most significant two bits specify the action that must be taken if the type is not recognized. These are:

00 ignore this option and continue processing the other option fields in the header;

01 discard the complete packet;

10 discard the packet and return an ICMP error report to the source indicating a parameter problem and the option type not recognized;

11 same as for 10 except the ICMP report is only returned if the destination address is not a multicast address.

Note also that since the type of extension header is indicated in the preceding *next header* field, the related decoder that has been written to decode the contents of the header is invoked automatically as each header is processed. Some examples of each header type now follow.

Hop-by-hop options

This type of header contains information that must be examined by all the gateways and routers the packet visits along its route. The *next header* value for this is 0 and, if this header is present, it must follow the main header. Hence the *next header* field in the main header is set to 0. An example of its use is for a host to send a datagram that contains a payload of more than 64K bytes. This is

particularly useful, for example, when a host is transferring many very large files over a path that supports a maximum transmission unit (MTU) significantly greater than 64 kbytes. Datagrams that contain this header are known as **jumbograms** and the format of the header is shown in Figure 6.43(a).

As we can see, this type of header is of fixed length and comprises two 32-bit words (8 bytes). The *header extension length* field indicates the length of the header in multiples of 8 bytes, excluding the first 8 bytes. Hence in this case the field is 0. This option contains only one option field, the *jumbo payload length*. As we indicated earlier, this is encoded in the TLV format. The *type* for this option is 194 (11000010) and the *length* is 4 (bytes). The *value* in the *jumbo payload length* is the length of the packet in bytes, excluding the main header but including the 8 bytes in the extension header. This makes the *payload length* in the main header redundant and hence this is set to zero.

Routing

This plays a similar role to the (strict) *source routing* and *loose source routing* optional headers used in IPv4 datagrams. The *next header* value for this type of

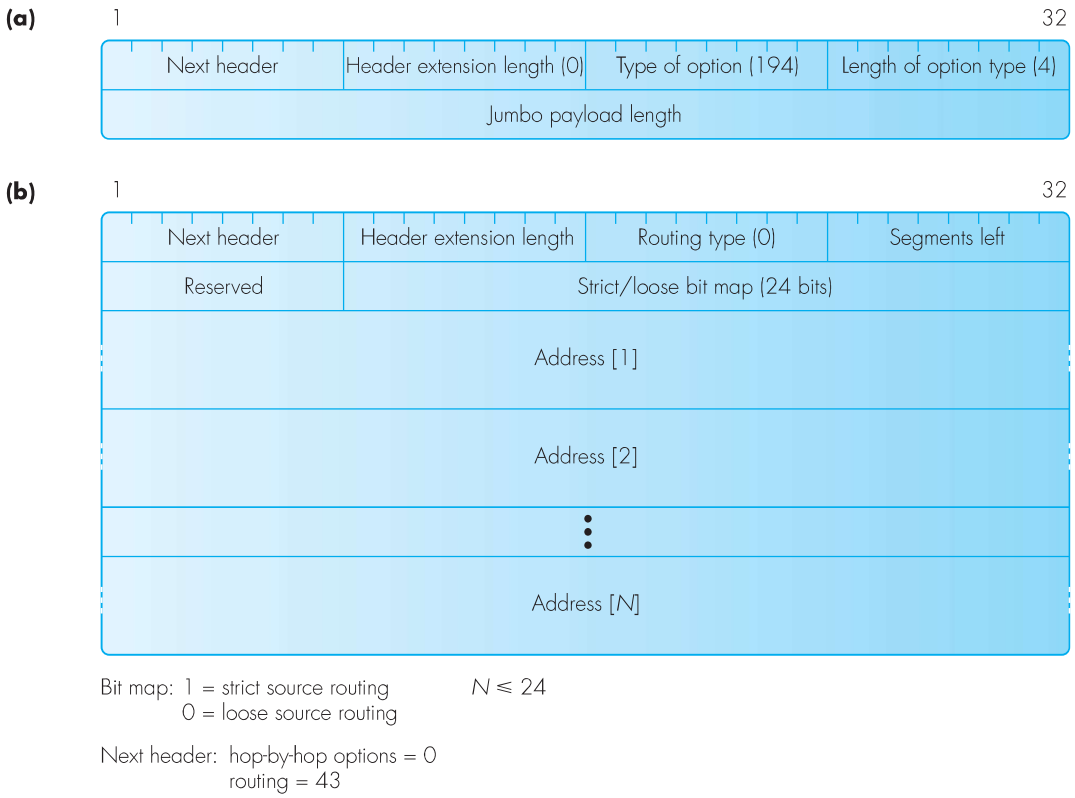


Figure 6.43 Extension header formats: (a) hop-by-hop options; (b) routing.

header is 43. Currently only one type of routing header has been defined, the format of which is shown in Figure 6.43(b).

The *header extension length* is the length of the header in multiples of 8 bytes, excluding the first 8 bytes. Hence, as we can deduce from the figure, this is equal to two times the number of (16-byte) addresses present in the header. This must be an even number less than or equal to 46. The *segments left* field is an index to the list of addresses that are present. It is initialized to 0 and is then incremented by the next router in the list as it is visited. The maximum therefore is 23.

The second 4-byte word contains a *reserved* byte followed by a 24-bit field called the *strict/loose bit map*. This contains one bit for each potential address present starting at the leftmost bit. If the related bit is a 1, then the address must be that of a directly attached (neighbor) router – that is, strict source routing. If the bit is a 0, then the address is that of a router with possibly several other routers in between – loose source routing. The latter is used, for example, when tunneling is required to forward the datagram. In the case of strict source routing, at each router the destination address in the main header is changed to that obtained from the list of addresses before the index is incremented. In the case of loose source routing, the destination address will be that of an attached neighbor which is on the shortest-path route to the specified address.

Example 6.7

A datagram is to be sent from a source host with an IPv6 address of A to a destination host with an IPv6 address of B via a path comprising three IPv6 routers. Assuming the addresses of the three routers are R1, R2, and R3 and strict source routing is to be used, (i) state what the contents of the initial values in the various fields in the extension header will be and (ii) list the contents of the source and destination address fields in the main header and the *segments left* field in the extension header as the datagram travels along the defined path.

Answer:

(i) Extension header initial contents:

Next header = Transport layer protocol
Header extension length = $2 \times 3 = 6$
Routing type = 0
Segments left = 0
Strict/loose bit map = 11100000 00000000 00000000
List of addresses = R1, R2, R3 (each of 16 bytes)

(ii) Contents of main header fields:

At source	SA = A	DA = R1	Segments left = 0
At R1	SA = A	DA = R2	Segments left = 1
At R2	SA = A	DA = R3	Segments left = 2

Fragment

This header is present if the original message submitted by the transport layer protocol exceeds the MTU of the path/route to be used. The *next header* value for a *fragment* extension header is 44. The fragmentation and reassembly procedures are similar to those used with IPv4 but, in the case of IPv6, the fragmentation procedure is carried out only in the source host and not by the routers/gateways along the path followed by the packet(s). As we saw in Figure 6.41(a), there is no don't fragment (D) bit in the IPv6 main header since, in order to speed up the processing/routing of packets, IPv6 routers do not support fragmentation. Hence, as we explained in Section 6.6.9, either the minimum MTU size of 576 bytes must be used or the *path MTU discovery* procedure is used to determine if the actual MTU size is greater than this. In either case, if the submitted message (including the transport protocol header) exceeds the chosen MTU (minus the 40 bytes for the IPv6 main header), then the message must be sent in multiple packets, each with a main header and a *fragment* extension header. The various fields and the format of each *fragment* extension header are shown in Figure 6.44(a). An example is shown in Figure 6.44(b).

Each packet contains a main header – plus, if required, a *hop-by-hop options* header and a *routing* header – followed by a *fragment* extension header and the fragment of the message being transmitted. Thus the maximum size of the payload (and hence message fragment) in each packet will be the MTU size being used minus the number of 8-byte fields required for the main header and any extension headers that are present. The *payload length* field in the main header of the first packet indicates the total number of bytes in the message being transmitted – including the IP header – plus the number of bytes that are required for the other extension headers that are being used. The *payload length* in the main header of the remaining packets indicates the number of bytes in the packet following the main header.

The various fields in each *fragment* header have similar functions to those used with IPv4. The *fragment offset* indicates the position of the first byte of the fragment contained within the packet relative to the start of the complete message being transmitted. Its value is in units of 8-bytes. The *M-bit* is the *more fragments bit*; it is a 1 if more fragments follow and a 0 if the packet contains the last fragment. Similarly, the value in the *identification* field is used by the destination host, together with the source address, to relate the data fragments contained within each packet to the same original message. Normally, the source uses a 32-bit counter (that is incremented by 1 for each new message transmitted) to keep track of the next value to use.

Authentication and encapsulating security payload

As we shall expand upon in Section 10.7.1, authentication and the related subject of encryption are both mechanisms that are used to enhance the security of a message during its transfer across a network. In the case of authentication, this enables the recipient of a message to validate that the

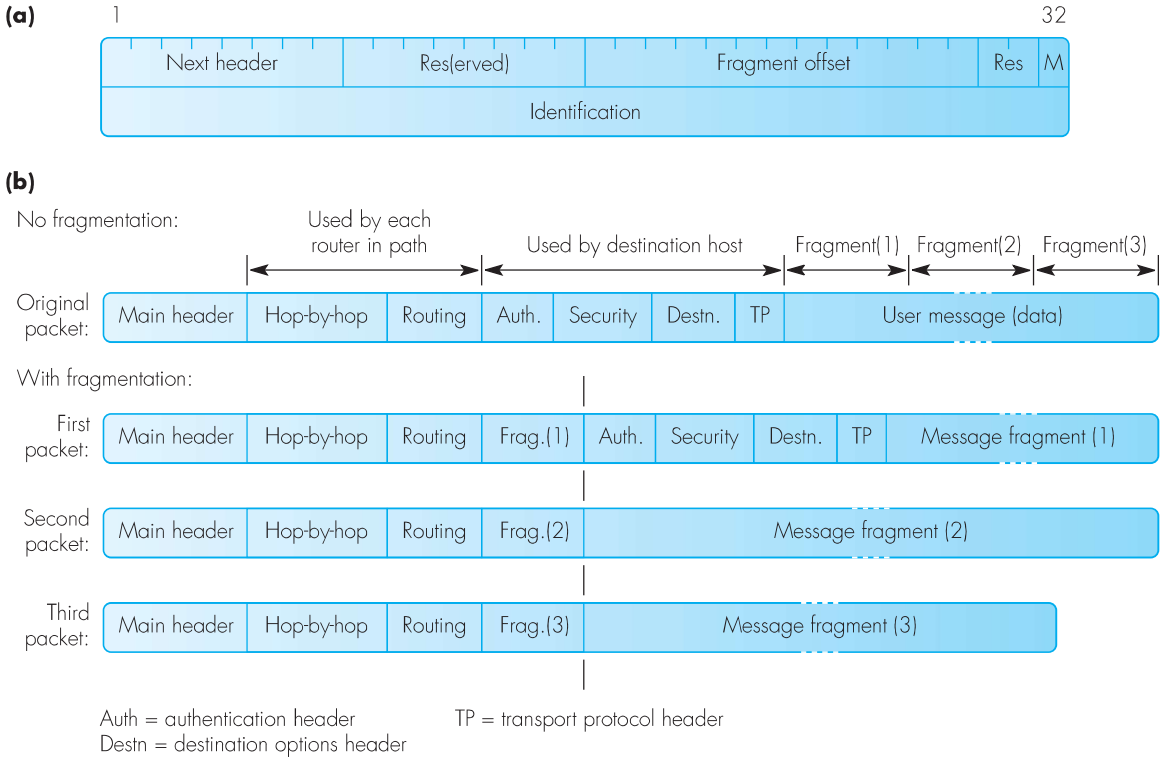


Figure 6.44 IPv6 fragmentation: (a) fragment header fields and format; (b) example.

message was indeed sent by the source address present in the packet/datagram header and not by an impostor. Encryption is concerned with ensuring that the contents of a message can only be read by – and hence have meaning to – the intended recipient. The *authentication* and *encapsulating* security payload (ESP) extension headers are present when both these features are being used at the network layer.

When using IPv6 authentication, prior to any information (packets) being exchanged, the two communicating hosts first use a secure algorithm to exchange secret keys. An example is the MD5 algorithm we describe later in Section 10.3. Then, for each direction of flow, the appropriate key is used to compute a checksum on the contents of the entire datagram/packet. The computed checksum is then carried in the authentication header of the packet. The same computation is repeated at the destination host, and only if the computed checksum is the same as that carried in the authentication header is it acknowledged that the packet originated from the source host address indicated in the main header and also that the packet contents have

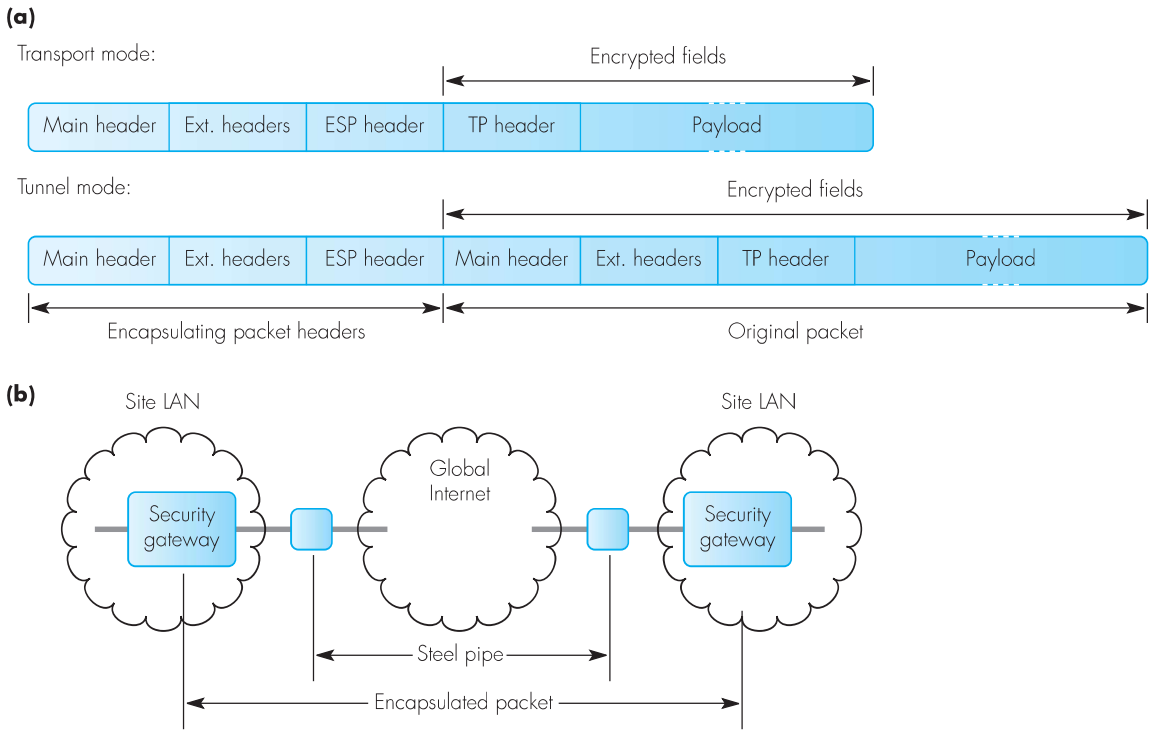


Figure 6.45 Encapsulating security payload: (a) transport and tunnel modes; (b) example application of tunnel mode.

not been modified during its transfer over the network. We discuss the subject of authentication further in Section 10.4.

The encryption algorithm used with the ESP is also based on the use of an agreed secret key. An example is the DES algorithm we describe later in Section 10.2.3. The agreed key is used to encrypt either the transport protocol header and payload parts of each packet or, in some instances, the entire packet including the main header and any extension headers present. In the case of the latter, the encrypted packet is then carried in a second packet containing a completely different main and, if necessary, extension headers. The first is known as **transport mode encryption** and the second **tunnel mode encryption**. Figure 6.45(a) illustrates the principle of operation of both modes.

As we can deduce from the figure, the overheads associated with the tunnel mode are significantly higher than those of the transport mode. The extra security obtained with the tunnel mode is that the information in the main and extension headers of the original packet cannot be interpreted by a person passively monitoring the transmissions on a line. An example use of this mode is in

multisite enterprise networks that use the (public) Internet to transfer packets from one site to another. The general scheme is shown in Figure 6.45(b).

As we will show in Figure 9.4(a) and explain in the accompanying text, associated with each site is a security gateway through which all packet transfers to and from the site take place. Hence to ensure the header information (especially the routing header) of packets is not visible during the transfer of the packet across the Internet, the total packet is encrypted and inserted into a second packet by the IP in the security gateway with the IPv6 address of the two communicating gateways in the source and destination address fields of the main header. The path through the Internet connecting the two gateways is referred to as a **steel pipe**.

Destination options

These are used to convey information that is examined only by the destination host. As we indicated earlier, one of the ways of encoding options is to use the type-length-value format. Hence in order to ensure a header that uses this format comprises a multiple of 8 bytes, two *destination options* have been defined. These are known as Pad1 and Pad*N*, the first to insert one byte of padding and the second two or more bytes of padding. Currently these are the only two options defined.

6.8.4 Autoconfiguration

As we described earlier in Section 6.1, the allocation of the IP addresses for a new network involves a central authority to allocate a new netid – the ICANN – and a local network administrator to manage the allocation of hostids to each attached host/end system. Thus, the allocation, installation and administration of IPv4 addresses can entail considerable effort and expenditure. To alleviate this, IPv6 supports an autoconfiguration facility that enables a host to obtain an IP address dynamically via the network and, in the case of mobile hosts, use it just for the duration of the call/session.

Two types of autoconfiguration are supported. The first involves the host communicating with a local (site) router using a simple (stateless) request-response protocol. The second involves the host communicating with a site (or enterprise) address server using an application protocol known as the **dynamic host configuration protocol** (**DHCP**). The first is suitable for small networks that operate in the broadcast mode (such as an Ethernet LAN) and the second for larger networks in which the allocation of IP addresses needs to be managed.

With the first method, a simple protocol known as **neighbor discovery** (**ND**) is used. As we show in Figure 6.46, this involves the host broadcasting a *router solicitation* packet/message on the subnet/network and the router responding with a *router advertisement* message. Both messages are ICMPv6 messages and hence are carried in an IPv6 packet. The latter is then broadcast over the LAN in a standard frame.

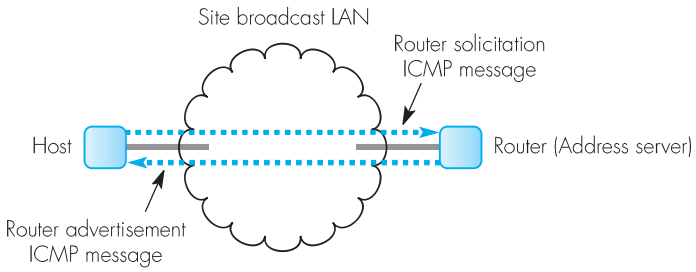


Figure 6.46 Neighbor discovery protocol messages.

The main header of the IPv6 packet containing the router solicitation message has an IPv6 source address created by the host. This is made up of the (standard) link-local address prefix and the 48-bit MAC address of the host's LAN interface. As we indicated earlier, with IPv6 a number of permanent multicast group addresses have been defined including an all-routers group address. Hence the destination address in the packet main header is set to this and, since the packet is broadcast over the LAN, it is received by the ICMP in all the routers that are attached to the LAN.

A single router is selected to process this type of ICMP message and this responds to a router solicitation message with a route advertisement message containing the Internet-wide address prefix for the subnet/network. On receipt of this, the ICMP in the host proceeds to create its own IPv6 address by adding its 48-bit MAC address to the prefix. As we can deduce from this, in relation to IPv4, this is equivalent to the router providing the netid of the site and the host using its own MAC address as the hostid. Also, the same procedure can be used by mobile hosts.

With the second method, the host requests an IPv6 address from the site address server using the DHCP. The **DHCP address server** first validates the request and then allocates an address from the managed list of addresses the server contains. Alternatively if a site does not have its own address server – for example if the site LAN is part of a larger enterprise network — then a designated router acts as a **DHCP relay agent** to forward the request to the DHCP address server.

6.9 IPv6/IPv4 interoperability

The widespread deployment of IPv4 equipment means that the introduction of IPv6 is being carried out in an incremental way. Hence a substantial amount of the ongoing standardization effort associated with IPv6 is concerned with the interoperability of the newer IPv6 equipment with existing IPv4 equipment. Normally, when a new network/internetwork is created, it is based on the IPv6 protocol and, in the context of the existing

(IPv4) Internet, it is referred to as an **IPv6 island**. It is necessary to provide a means of interworking between the two types of network at both the address level and the protocol level. In this section, we identify a number of situations where interoperability is required and describe a selection of the techniques that are used to achieve this.

6.9.1 Dual protocols

Dual stacks are already widely used in networks that use dissimilar protocol stacks, for example a site server that supports IPX on one port and IP on a second port. In a similar way, dual protocols can be used to support both IPv4 and IPv6 concurrently. An example is shown in Figure 6.47.

In this example, the site has a mix of hosts, some that use IPv4 and others IPv6. In order to be able to respond to requests originating from both types of host, the server machine has both an IPv4 and an IPv6 protocol at the network layer. The value in the version field of the datagram header is then used by the link layer protocol to pass a datagram to the appropriate IP. In this way the upper layer protocols are unaware of the type of IP being used at the network layer.

6.9.2 Dual stacks and tunneling

A common requirement is to interconnect two IPv6 islands (networks/internetworks) through an intermediate IPv4 network/internetwork. To achieve this, the gateway/router that connects each IPv6 island to the IPv4 network

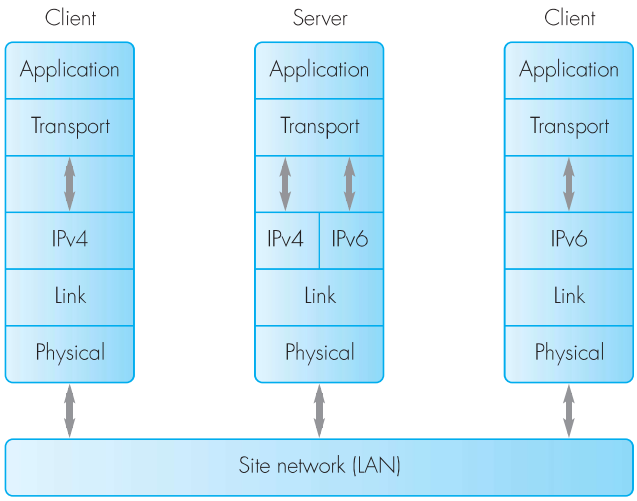


Figure 6.47 IPv6/IPv4 interoperability using dual (IPv6/IPv4) protocols.

must have dual stacks, one that supports IPv6 and the other IPv4. The IPv6 packets are then transferred over the IPv4 network using tunneling. The general approach is illustrated in Figure 6.48(a) and the protocols involved in Figure 6.48(b).

As we showed earlier in Figure 6.16 and explained in the accompanying text, tunneling is used to transfer a packet relating to one type of network layer protocol across a network that uses a different type of network layer protocol. Hence in the example shown in Figure 6.48, each IPv6 packet is transferred from one (IPv6/IPv4) edge gateway to the other edge gateway within an IPv4 packet. As we show in the figure, in order to do this, the two

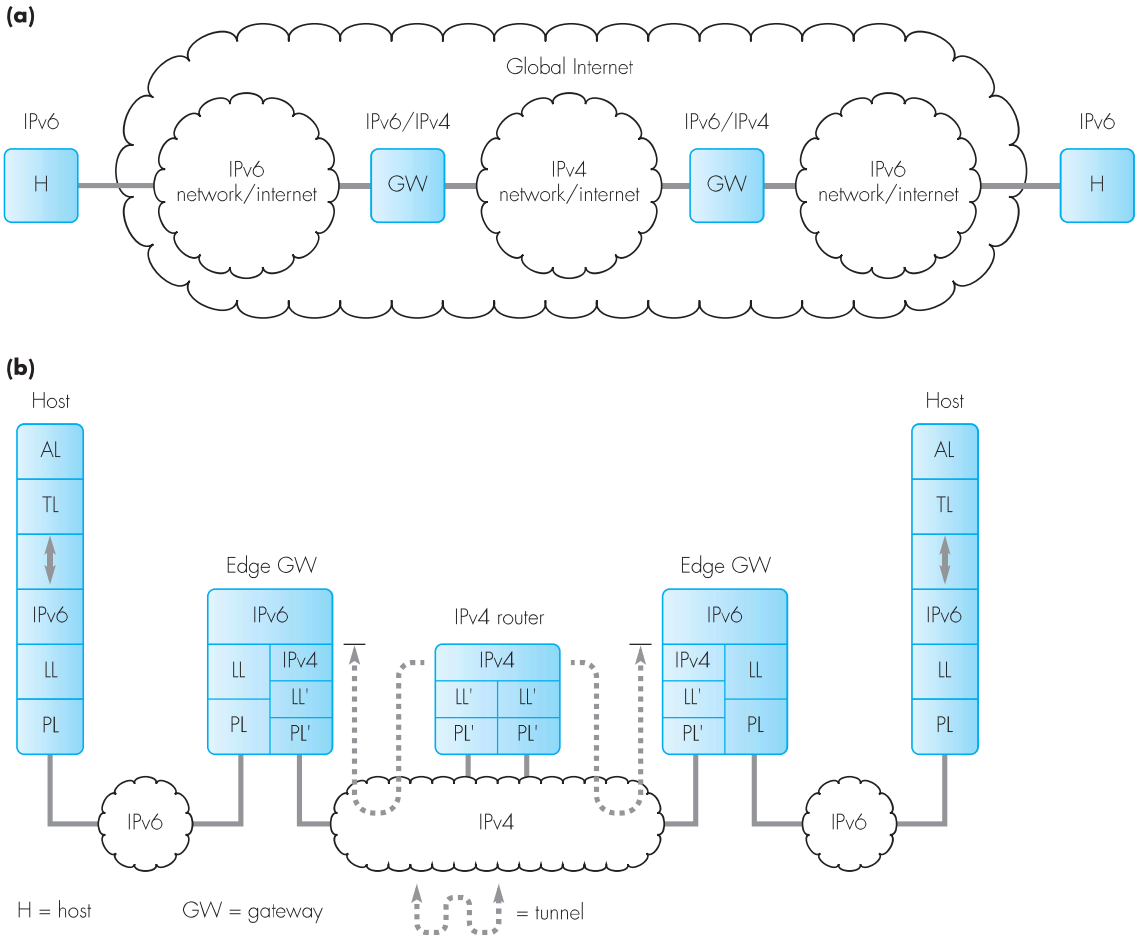


Figure 6.48 IPv6/IPv4 interoperability using dual stacks and tunneling: (a) schematic; (b) protocols.

edge gateways contain dual stacks each of which has a related IPv6/IPv4 address associated with it. Normally, entered by network management, the routing table entry for the remote destination IPv6 host is the IPv4 address of the remote edge gateway.

The IPv6 in each gateway, on determining from its routing table that an (IPv6) packet should be forwarded to a remote IPv6 network via an IPv4 tunnel, passes the packet to the IPv4 protocol together with the IPv4 address of the remote gateway. The IPv4 protocol first encapsulates the IPv6 packet in an IPv4 datagram/packet with the IPv4 address of the remote gateway in the destination address field. It then uses this address to obtain the (IPv4) address of the next-hop router from its own (IPv4) routing table and proceeds to forward the packet over the IPv4 network/internetwork. On receipt of the packet, the IPv4 in the remote gateway, on detecting from its own routing table that it is a tunneled packet, strips off the IPv4 header and passes the payload – containing the original IPv6 packet – to the IPv6 layer. The latter then forwards the packet to the destination host identified in the packet header in the normal way.

6.9.3 Translators

A third type of interoperability requirement is for a host attached to an IPv6 network – and hence having an IPv6 address and using the IPv6 protocol – to communicate with a host that is attached to an IPv4 network – hence having an IPv4 address and using the IPv4 protocol. In this case, both the addresses and the packet formats are different and so a translation operation must be carried out by any intermediate routers/gateways. As we show in Figure 6.49, the translations can be performed at either the network layer – part (a) – or the application layer – part (b).

Using the first approach, on receipt of an IPv6/IPv4 packet, this is converted into a semantically equivalent IPv4 /IPv6 packet. This involves a **network address translator** (**NAT**) and a **protocol translator** (**PT**). The intermediate gateway is then known as a **NAT-PT gateway**. As we explained earlier in Section 6.8.2, an IPv4 address can be embedded in an IPv6 address. Hence to send a datagram/packet from a host with an IPv6 address to a host with an IPv4 address, the NAT in the gateway can readily obtain the destination IPv4 address from the destination address in the main header of the IPv6 packet. The issue is what the source address in the IPv4 packet header should be.

A proposed solution is for the NAT to be allocated a block of hostids for the destination IPv4 network. These, together with the netid of the destination network, then form a block of unique IPv4 addresses. For each new call/session, the NAT allocates an unused IPv4 address from this block for the duration of the call/session. It then makes an entry in a table containing the IPv6 address of the V6 host and its equivalent (temporary) IPv4 address. The NAT translates between one address and the other as the

packet is relayed. A timeout is applied to the use of such addresses and, if no packets are received within the timeout interval, the address is transferred back to the free address pool.

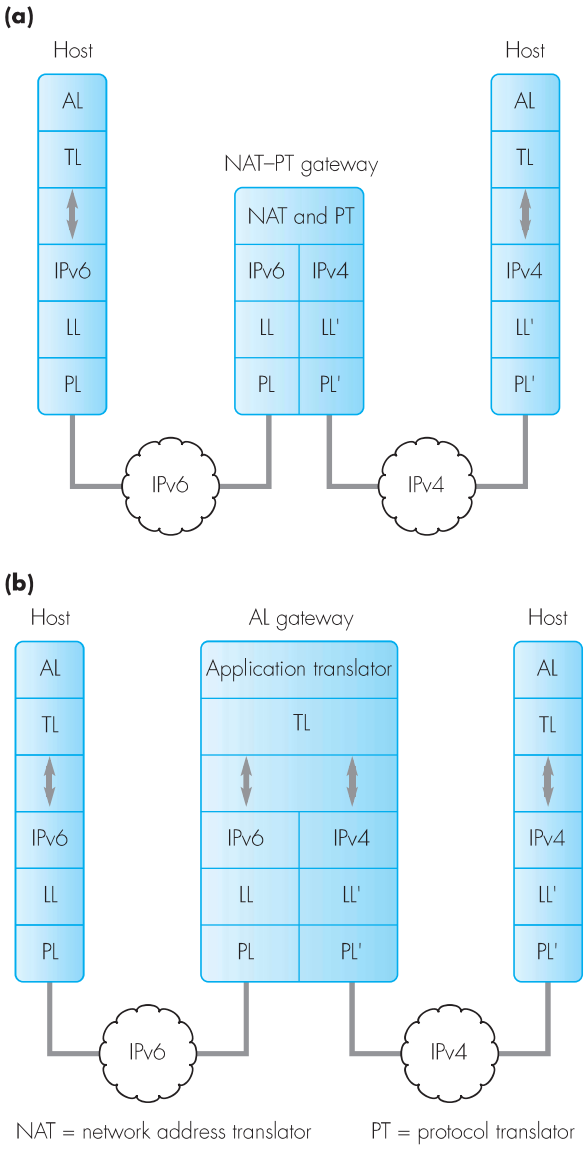


Figure 6.49 IPv6/IPv4 interoperability using translators: (a) network level; (b) application level.

The protocol translation operation is concerned with translating the remaining fields in the packet header and, in the case of ICMP messages, converting ICMPv4 messages into and from ICMPv6 messages. As we indicated in Section 6.8.1, most of the fields in the IPv6 main header have the same meaning as those in the IPv4 header and hence their translation is relatively straightforward. In general, however, there is no attempt to translate the fields in the options part. Similarly, since ICMPv6 messages have different *type* fields, the main translation performed is limited to changing this field. For example, the two ICMPv4 query messages have *type* values of 8 and 0 and the corresponding ICMPv6 messages are 128 and 129.

The use of a NAT-PT gateway works providing the packet payload does not contain any network addresses. Although this is the case for most application protocols, a small number do. The FTP application protocol, for example, often has IP addresses embedded within its protocol messages. In such cases, therefore, the translation operation must be carried out at the application layer. The associated gateway is then known as an **application level gateway** (**ALG**). This requires a separate translation program for each application protocol. Normally, therefore, most translations are performed at the network layer – using a NAT and a PT – and only the translations relating to application protocols such as FTP are carried out in the application layer.

Summary

A summary of the topics discussed in this chapter is given in Figure 6.50.

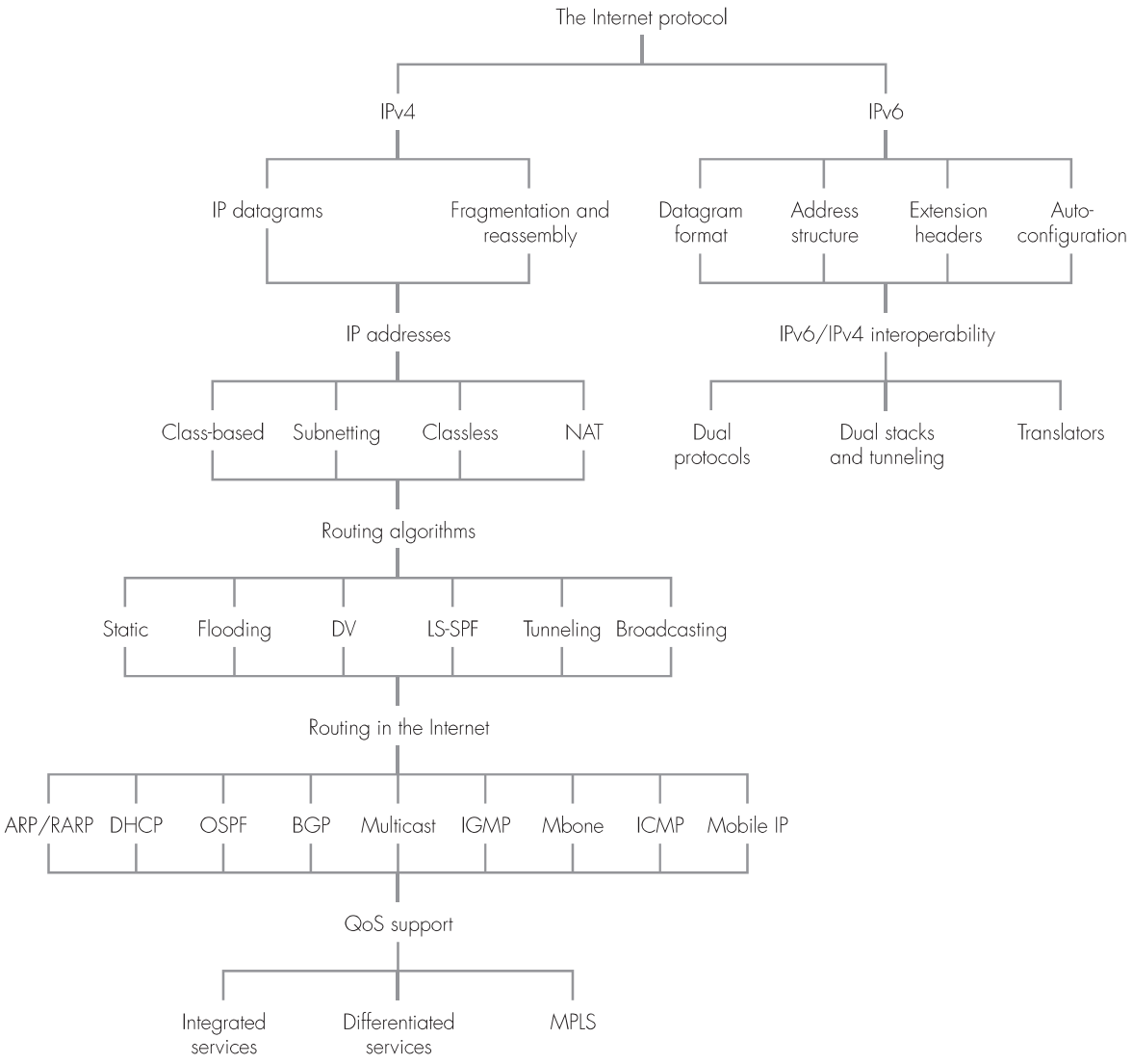


Figure 6.50 Internet protocol summary.

Exercises

Section 6.1

6.1 With the aid of the network schematic shown in Figure 6.1, explain briefly the role of the following network components and protocols:
(i) access network gateway,
(ii) routing gateway,
(iii) internet protocol (IP),
(iv) datagram/packet.

6.2 With the aid of the protocol stack shown in Figure 6.2, explain briefly the meaning of the term "adjunct protocol" and how the IP in the destination host determines to which protocol the contents/payload of a received IP datagram should be passed.

Section 6.2

6.3 In relation to the IP datagram/packet format shown in Figure 6.3, explain the role of the following header fields:
(i) IHL,
(ii) TOS,
(iii) Total length and Identification,
(iv) flag bits,
(v) Fragment offset,
(vi) Time-to-live,
(vii) Protocol,
(viii) Header checksum,
(ix) Options.

Section 6.3

6.4 Assume a message block of 7000 bytes is to be transferred from one host to another as shown in the example in Figure 6.4. In this instance, however, assume the token ring LAN has an MTU of 3000 bytes. Compute the header fields in each IP packet shown in the figure as it flows
(i) over the token ring LAN,
(ii) over the Ethernet LAN.

6.5 State why fragmentation is avoided whenever possible and the steps followed within each host IP to achieve this.

Section 6.4

6.6 List the four types of IP address schemes that are in use and give a brief summary of the use of each scheme.

6.7 Explain the meaning of the term "IP address class" and why these classes were created. Hence, with the aid of the three (unicast) address classes identified in Figure 6.5, identify a particular application for each class.

6.8 State the meaning of the following addresses:
(i) an address with a netid of all 0s,
(ii) an address with a netid of all 1s,
(iii) an address with a hostid of all 0s,
(iv) an address of all 1s.

6.9 Explain the meaning of the term "dotted decimal". Hence derive the netid and hostid for the following IP addresses expressed in dotted decimal notation:
(i) 13.0.0.15,
(ii) 132.128.0.148,
(iii) 220.0.0.0,
(iv) 128.0.0.0,
(v) 224.0.0.1.

6.10 With the aid of the example in Figure 6.6, explain why subnetting was introduced. Hence state the meaning of a subnet router and an address mask.

6.11 A site with a netid of 127.0 uses 20 subnet routers. Suggest a suitable address mask for the site which allows for a degree of expansion in the future. Give an example of a host IP address at this site.

6.12 With the aid of Example 6.3, explain the classless inter-domain routing (CIDR) scheme.

6.13 With the aid of Example 6.4, explain how, with CIDR, multiple address matches are possible with a network that has been allocated a large block of addresses. State which of these matches is chosen and why.

6.14 Explain why the network address translation (NAT) scheme is now used with most new access networks.

6.15 In relation to the NAT scheme shown in Figure 6.7(a), determine the number of host addresses/interfaces that have been declared for private use.

6.16 In relation to the outline operation of NAT shown in Figure 6.7(b), explain:
(i) how the four IP addresses are used and the role of the NAT table,
(ii) how the source and destination addresses in the related IP and TCP fields are derived.

Section 6.5

6.17 State the meaning of the following terms relating to the routing of packets over an internet:
(i) line cost,
(ii) path cost,
(iii) hopcount,
(iv) routing metric,
(v) shortest path.

6.18 With the aid of the routing table entries shown in Figure 6.9, explain the meaning of the terms:
(i) static routing tables,
(ii) next-hop routing,
(iii) optionality principle,
(iv) alternative paths.

6.19 With the aid of the broadcast diagram shown in Figure 6.10, explain:
(i) why the broadcast following the route via R2 is assumed to arrive first
(ii) how duplicate copies of a packet are determined by R3
(iii) how the number of copies of the packet produced is limited
(iv) why flooding is an example of an adaptive/dynamic routing algorithm.

6.20 In relation to the distance vector algorithm, with the aid of the example shown in Figure 6.11, explain:
(i) the meaning of the term "connectivity/adjacency table" and how the table's contents are obtained,
(ii) how the final routing table entries for R3 are built up,
(iii) how a packet from a host attached to netid3 is routed to a host attached to netid1,
(iv) the limitations of the algorithm including how looping may arise.

6.21 Assuming the connectivity/adjacency tables given in Figure 6.12(a), show how the overall network topology is built up by router R3 using the link state algorithm. Hence derive the contents of the netid location table for R3.

6.22 Assuming the initial network topology shown in Figure 6.13(a), use the Dijkstra algorithm to derive the shortest paths from R3 to each other router. State the meaning of the terms "tentative" and "permanent" relating to the algorithm and the implications of alternative paths/routers.

6.23 Using the set of link state, routing, and connectivity tables for R1, R2 and R3 in Figure 6.15, explain how a packet received by R3 from G3 with a destination netid of 1 is routed using hop-by-hop routing.

6.24 Explain how a packet received by R3 from G3 with a destination netid of 1 is routed using source routing. Include how the routing tables you use are derived.

6.25 In relation to the link-state algorithm, explain why each link-state message contains a sequence number and a timeout value. How are these used?

6.26 Explain the term "tunneling" and when it is used. Hence with the aid of the schematic diagram shown in Figure 6.16, explain how the host on the left of the diagram sends an IP datagram/packet to a host attached to the Internet. Include in your explanation the role of the two multiprotocol routers.

State an application of tunneling IP packets over an IP network.

6.27 What are the aims of both the reverse path forwarding algorithm and the spanning tree broadcast algorithm?

6.28 Use the final routing tables and broadcast sequence relating to the reverse path forwarding algorithm shown in Figure 6.17 to explain why only the (broadcast) packet received by SR6 from SR3 is broadcast at the fourth stage. What is the number of duplicate broadcasts that occur?

6.29 Assuming the network topology shown in Figure 6.18(a) and that SR3 is an access gateway, determine the spanning tree derived by each subnet router. Use this to derive the broadcast sequence.

Section 6.6

6.30 With the aid of the generalized Internet architecture shown in Figure 6.19, give an example of the type of network that is used at each tier in the hierarchy and the routing method that is used with it.

6.31 Define the terms "IP address", "MAC address", and "hardware/physical address". Also explain the terms "address-pair" and "ARP cache".

6.32 In relation to the simple network topology shown in Figure 6.20, explain why:
(i) on receipt of an ARP request message, each host retains a copy of the IP/MAC address-pair of the source host in its ARP cache,
(ii) on receipt of an ARP reply message, the ARP in the source host makes an entry of the IP/MAC address-pair in its own cache,
(iii) the Lan port of the gateway keeps a copy of the IP/MAC address-pair from each ARP request and reply message that it receives.

6.33 Explain the role of a proxy ARP. Hence explain how an IP packet sent by a host at one site is routed to a host at a different site. Also explain how the reply packet is returned to the host that sent the first packet.

6.34 Explain how the reverse ARP is used to enable a diskless host to determine its own IP address from its local server.

6.35 With the aid of the two frame formats shown in Figure 6.21, explain:
(i) how the MAC sublayer in the receiver determines whether a received frame is in the Ethernet format or IEEE802.3,
(ii) the number of pad bytes required with each frame type.

6.36 State the role of the dynamic host configuration protocol (DHCP).

6.37 Use the example network topology shown in Figure 6.22(a) to describe the DHCP message exchange sequence to obtain an IP address.

6.38 In relation to the simplified Internet structure shown in Figure 6.23, explain the function of the four types of router.

6.39 With the aid of the simplified autonomous system (AS) topology shown in Figure 6.24(a), describe the operation of the OSPF algorithm. First describe how the directed graph is derived and then the derivation of the SPF tree for R7.

6.40 List the message types used with OSPF and explain their function when routing within an AS.

6.41 List the four message types that are used in the border gateway protocol (BGP) and explain their function.

6.42 Given the example backbone topology shown in Figure 6.25, give an example of an update message that changes the route followed between a pair of boundary routers.

6.43 In relation to multicasting over a LAN, describe how ICANN controls the allocation of

multicast addresses. Also explain how the 48-bit MAC address and 28-bit IP address of a host are derived from the allocated address. Hence with the aid of the schematic diagram shown in Figure 6.27(b), describe how a host joins a multicast session that is taking place over the LAN. Include the role of the multicast address table and group address table held by each member of the group.

6.44 What is the meaning of the term "multicast router"? Outline the sequence of steps that are followed to route an IP packet with a multicast address over the Internet.

6.45 Assume the same topology, multicast address table contents, routing table contents, and routing table entries as shown in Figure 6.28. Assuming the DVMRP, explain how a packet arriving from one of its local networks with a multicast address of C is routed by MR3 to all the other MRs that have an interest in this packet.

6.46 Repeat Exercise 6.45 but this time using the MOSPF routing protocol and the spanning tree shown in Figure 6.29(b).

6.47 What is the role of the IGMP protocol?

With the aid of the example shown in Figure 6.30, explain how a host that is attached to a local network/subnet of an MR joins a multicast session. Include in your explanation the table entries retained by both the host and the MR and how multicast packets relating to the session are then routed to the host.

6.48 With the aid of the example shown in Figure 6.30, explain the procedure followed when a host that is attached to a local network/subnet of an MR leaves a multicast session.

6.49 The multicast backbone (M-bone) network shown in Figures 6.28 and 6.29 comprised a set of multicast routers interconnected by single links. In practice, these are logical links since each may comprise multiple interconnected routers that do not take part in multicast routing. Explain how a multicast packet is sent from one mrouter to another using IP tunneling.

6.50 Explain briefly the role of the ICMP protocol and the different procedures associated with it. Hence explain how the path MTU discovery procedure is used to determine the MTU of a path/route prior to sending any datagrams.

6.51 In the simplified network architecture shown in Figure 6.32, explain the use of the following terms:

(i) home agent,
(ii) foreign agent.

Discuss the issues to be resolved when Host A wants to communicate with Host B.

6.52 In relation to the example shown in Figure 6.33, use a sequence diagram to illustrate the exchange of mobile IP messages in order to register a mobile host (Host B) with its HA and FA. List the main addresses held by both the HA and the FA after the registration procedure is complete.

6.53 Using the example shown in Figure 6.34, describe the indirect routing method used to route packets between Host A and Host B once Host B has been registered with the HA and FA.

Section 6.7

6.54 Discuss the reasons why improved levels of QoS support are now being used within the Internet.

6.55 Describe the role and principle of operation of the following control mechanisms used within Internet routers:

(i) token bucket filter,
(ii) weighted fair queuing,
(iii) random early detection.

6.56 Define the three different classes of service used with the IntServ scheme. With the aid of the network topology shown in Figure 6.35(a) describe the operation of the resource reservation protocol (RSVP). Include in your

description the meaning/role of the following:
(i) path, reserve, and path-tear messages,
(ii) path-state table,
(iii) cleanup timer,
(iv) soft-state.

6.57 Define the usage of the type of service (ToS) field in each packet header with the DiffServ scheme including the meaning of the term "DS packet codepoint".

6.58 With the aid of the general architecture shown in Figure 6.35(b), describe the operation of the DiffServ scheme. Include in your description the meaning/role of the following components of an ingress router:
(i) behavior aggregate,
(ii) traffic meter module,
(iii) MF classifier,
(iv) marker module,
(v) shaper/dropper.

6.59 Use the schematic diagram of a router in Figure 6.36 to explain the packet forwarding procedure in a conventional router.

6.60 Explain the terms traffic engineering, class-based queuing, shaping and grooming in an MPLS network.

6.61 In relation to the MPLS network architecture shown in Figure 6.37, explain where and why the various QoS mechanisms are located.

6.62 Using the example topology shown in Figure 6.38, insert two further sets of table entries to illustrate the label switching procedure.

6.63 Use the schematic diagram of the area border router/label edge router shown in Figure 6.39 to explain the MPLS forwarding procedure. Include in your explanation the role of the packet classifier, the LSP table, the output queue interfaces and the associated scheduling rules.

6.64 Explain how alternative routes to those computed using OSPF are used with the constraint-routed label distribution protocol (CR-LDP). Describe how the alternative routing tables are downloaded after the CR analysis.

Also explain the meaning/role of the following components of a core router:
(i) BA classifier,
(ii) per-hop behavior,
(iii) expedited forwarding,
(iv) assured forwarding.

Section 6.8

6.65 Discuss the reasons behind the definition of IP version 6, IPv6/IPng, including the main new features associated with it.

6.66 With the aid of the frame format shown in Figure 6.41(a), explain the role of the following fields in the IPv6 packet header:
(i) traffic class,
(ii) flow label,
(iii) payload length (and how this differs from the total length in an IPv4 packet header),
(iv) next header,
(v) hop limit,
(vi) source and destination addresses.

6.67 In relation to IPv6 addresses, with the aid of the prefix formats shown in Figures 6.42(a) and (b), explain the meaning/use of:
(i) address aggregation,
(ii) prefix formats,
(iii) embedded IPv4 addresses.

6.68 With the aid of the frame format shown in Figure 6.42(c), explain the meaning/use of the following IPv6 fields:
(i) registry,
(ii) top-level aggregators,
(iii) next-level agrregators,
(iv) site-level aggregators,
(v) interface ID.
Comment on the implications of adopting a hierarchical address structure.

6.69 With the aid of examples, explain the use of a link local-use address and a site local-use address.

6.70 Explain the format and use of
(i) a multicast address,
(ii) an anycast address.

6.71 With the aid of examples, show how an IPv6 address can be represented:
(i) in hexadecimal form,
(ii) with leading zeros removed,
(iii) when it contains an IPv4 embedded address.

6.72 Explain the role of the extension headers that may be present in an IPv6 packet. List the six types of extension header and state their use. Also, with the aid of examples, state the position and order of the extension headers in relation to the main header.

6.73 The fields in an options extension header are encoded using a type-length-value format. Use the hop-by-hop options header as an example to explain this format.

6.74 In relation to the routing extension header, explain:
(i) the difference between strict and loose source routing, and the associated bit map,
(ii) the use of the segments left field.

6.75 In relation to the packet formats shown in Figure 6.44, explain:
(i) the meaning and use of the identification field and the M-bits in each extension header,
(ii) why the hop-by-hop and routing headers are present in each fragment packet.

6.76 In relation to the encapsulating security payload header, with the aid of diagrams, explain:
(i) the difference between transport mode and tunnel mode encryption,
(ii) the meaning and use of the term "steel pipe".

6.77 State the aim of the autoconfiguration procedure used with IPv6 and the application domain of:
(i) the neighbor discovery (ND) protocol,
(ii) the dynamic host configuration protocol (DHCP).

6.78 With the aid of Figure 6.46, explain the operation of the ND protocol. Include the role of the router solicitation and router advertisement messages and how a host creates its own IP address.

6.79 Explain how an IPv6 address is obtained:
(i) using a DHCP address server,
(ii) using a DHCP relay agent.

Section 6.9

6.80 With the aid of Figure 6.47, explain how a LAN server can respond to requests from both an IPv4 and an IPv6 client using dual protocols.

6.81 With the aid of Figure 6.48, explain how two hosts, each of which is attached to a different IPv6 network, communicate with each other if the two IPv6 networks are interconnected using an IPv4 network. Include the addresses that are used in each message transfer.

6.82 State the meaning of the terms "network address translation" (NAT) and "protocol translation" (PT). Hence, with the aid of the schematic diagram shown in Figure 6.49(a), explain the role and operation of a NAT-PT gateway. Include what the source address in each IPv6 packet should be.

6.83 Identify when the use of a NAT-PT gateway is not practical. Hence, with the aid of the schematic diagram shown in Figure 6.49(b), explain the role of an application level gateway.