ABSTRACT

The current version of the Internet Protocol (IP) is becoming obsolete because of its limited address space, lack of needed functionality, and inadequate security features. The next generation of IP, called IPv6, has now been standardized and will carry TCP/IP networks and applications well into the next century.

IPv6: The New Internet Protocol

William Stallings

With the changing nature of the Internet and business networks, the current Internet Protocol (IP), which is the backbone of Transmission Control Protocol (TCP)/IP networking, is rapidly becoming obsolete.

Until recently, the Internet and most other TCP/IP networks have primarily provided support for rather simple distributed applications, such as file transfer, electronic mail, and remote access using TELNET; but today, the Internet is increasingly becoming a multimedia, application-rich environment, led by the huge popularity of the World Wide Web. At the same time, corporate networks have branched out from simple e-mail and file transfer applications to complex client/server environments and, most recently, intranets that mimic the applications available on the Internet.

All of these developments have outstripped the capability of IP-based networks to supply needed functions and services. An internetworked environment needs to support real-time traffic, flexible congestion control schemes, and security features. None of these requirements is easily met with the existing IP.

However, the driving force behind the development of a new IP is the stark fact that the world is running out of IP addresses for networked devices. The fixed 32-bit address length of IP is inadequate for the explosive growth of networks.

With the shortcomings of the existing IP becoming increasingly evident, a new protocol, known as IPv6 (IP version 6), has been defined to ultimately replace IP.[1] This article begins with an overview of the role of an internet protocol, looks at the motivation for developing a new version of IP, and then examines some of its details.

THE ROLE OF AN INTERNET PROTOCOL[2]

IP provides the functionality for interconnecting end systems across multiple networks. For this purpose, IP is implemented in each end system and in routers, which are devices that provide connection between networks. Higher-level data at a source end system are encapsulated in an IP protocol data unit (PDU) for transmission. This PDU is then passed through one or more networks and connecting routers to reach the destination end system.

The router must be able to cope with a variety of differences among networks, including:

Addressing Schemes — The networks may use different schemes for assigning addresses to devices. For example, an IEEE 802 local area network (LAN) uses either 16-bit or 48-bit binary addresses for each attached device; an X.25 public packet-switching network uses 12-digit decimal addresses (encoded as 4 b/digit for a 48-bit address). Some form of global network addressing must be provided, as well as a directory service.

Maximum Packet Sizes — Packets from one network may have to be broken into smaller pieces to be transmitted on another network, a process known as *fragmentation*. For example, Ethernet imposes a maximum packet size of 1500 bytes; a maximum packet size of 1000 bytes is common on X.25 networks. A packet that is transmitted on an Ethernet system and picked up by a router for retransmission on an X.25 network may have to segment the incoming packet into two smaller ones.

Interfaces — The hardware and software interfaces to various networks differ. The concept of a router must be independent of these differences.

Reliability — Various network services may provide anything from a reliable end-to-end virtual circuit to an unreliable service. The operation of the routers should not depend on an assumption of network reliability.

The operation of the router, as Fig. 1 indicates, depends on an internet protocol. In this example. the IP of the TCP/IP protocol suite performs that function. IP must be implemented in all stations on all networks as well as on the routers. In addition, each station must have compatible protocols above IP to communicate successfully. The intermediate routers need only have up through IP.

Consider the transfer of a block of data from station *X* to station *Y* in Fig. 1. The IP layer at X receives blocks of data to be sent to Y from TCP in X. The IP layer attaches a header that specifies the global internet address of Y. That address is in two parts: network identifier and station identifier. Let us refer to this block as the IP datagram. Next, IP recognizes that the destination (Y) is on another subnetwork. So the first step is to send the datagram to a router, in this case router 1. To accomplish this, IP hands its data unit down to logical link

[1] *You may think I have skipped a few versions in this narrative. The currently deployed version of IP is actually IP version 4; previous versions of IP (1 through 3) were successively defined and replaced to reach IPv4. Version 5 was assigned to the Stream Protocol (ST) which runs parallel to IPv4 in some routers; hence, the use of the label "version 6."*

[2] *This section is based on material in [1].*

0163-6804/96/$05.00 © 1996 IEEE

control (LLC) with the appropriate addressing information. LLC creates an LLC PDU, which is handed down to the medium access control (MAC) layer. The MAC layer constructs a MAC packet whose header contains the address of router 1.

Next, the packet travels through the LAN to router 1. The router removes the packet and LLC headers and trailers and analyzes the IP header to determine the ultimate destination of the data, in this case Y. The router must now make a routing decision. There are two possibilities:

- The destination station Y is connected directly to one of the subnetworks to which the router is attached.
- To reach the destination, one or more additional routers must be traversed.

In this example, the datagram must be routed through router 2 before reaching the destination. So router 1 passes the IP datagram to router 2 via the intermediate network. For this purpose, the protocols of that network are used. For example, if the intermediate network is an X.25 network, the IP data unit is wrapped in an X.25 packet with appropriate addressing information to reach router 2. When this packet arrives at router 2, the packet header is stripped off. The router determines that this IP datagram is destined for Y, which is connected directly to a subnetwork to which the router is attached. The router therefore creates a packet with a destination address of Y and sends it out onto the LAN. The data finally arrive at Y, where the packet, LLC, and internet headers and trailers can be stripped off.

This IP service is unreliable; that is, IP does not guarantee that all data will be delivered, or that the delivered data will arrive in the proper order. It is the responsibility of the next higher layer, in this case TCP, to recover from any errors that occur. This approach provides for a great deal of flexibility. The IP approach means that each datagram is passed from router to router in an attempt to get from source to destination. Since delivery is not guaranteed, there is no particular reliability requirement on any of the subnetworks; thus, the protocol will work with any combination of subnetwork types. Because the sequence of delivery is not guaranteed, successive datagrams can follow different paths through the internet. This allows the protocol to react to congestion and failure in the internet by changing routes.

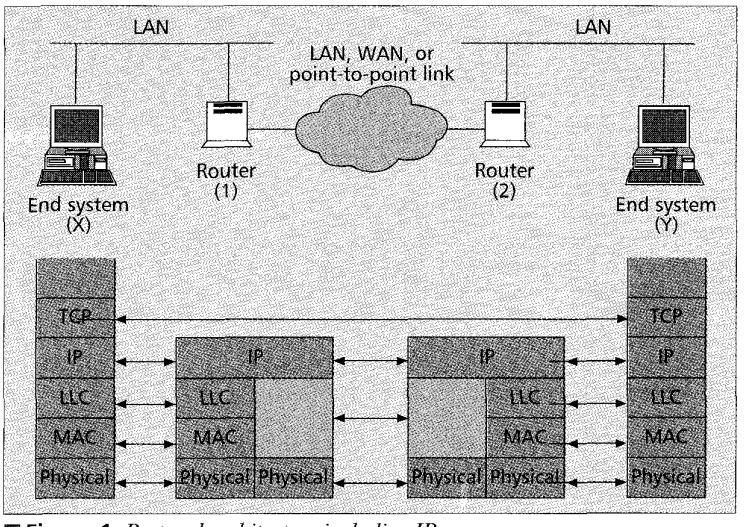


Figure 1. *Protocol architecture including IP.*

IP Next Generation

As the Internet has grown, it became apparent to many observers that the existing version of IP was inadequate to meet the performance and functional requirements for the Internet.

In response to these needs, the Internet Engineering Task Force (IETF) issued a call for proposals for a next-generation IP (IPng) in July of 1992. A number of proposals were received, and by 1994 the final design for IPng emerged. A major milestone was reached with the publication of RFC 1752, "The Recommendation for the IP Next Generation Protocol," issued in January 1995. RFC 1752 outlines the requirements for IPng, specifies the PDU formats, and highlights the IPng approach in the areas of addressing, routing, and security. A number of other Internet documents define details of the protocol, now officially called IPv6 (see Table 1a). In addition, a number of new security features have been designed for use with IPv6 but can also be used with the existing IPv4; these also have been documented in requests for comments (RFCs) (Table 1b). The driving motivation for the adoption of a new version of IP was the limitation imposed by the 32-bit address field in IPv4. But other considerations as well drove the design of IPv6. We look at these areas in this section.

Addressing

With a 32-bit address field, it is in principle possible to assign 2^{32} different addresses, which is over 4 billion possible addresses. One might think that this number of addresses was more than adequate to meet addressing needs on the Internet. However, in the late 1980s it was perceived that there would be a problem, and this problem began to manifest itself in the early 1990s. Some of the reasons for the inadequacy of 32-bit addresses include:

- The two-level structure of the IP address (network number, host number) is convenient but wasteful of address space. Once a network number is assigned to a network, all of the host-number addresses for that network number are assigned to that network. The address space for that network may be sparsely used, but as far as the effective IP address space is concerned, if a network number is used, all addresses within the network are.
- It is general practice to assign a unique network number to an IP network whether or not it is actually connected to the Internet. It is possible on a private internet to reuse numbers that are in use either in the public Internet or in other private internetworks, but this is a risky and cumbersome policy.
- Networks are proliferating rapidly. Most organizations boast multiple LANs, not just a single LAN system. Wireless networks are gradually assuming a major role. The Internet itself has been growing explosively for years.
- Growth of TCP/IP usage in new areas will result in a rapid growth in the demand for unique IP addresses. Examples are using TCP/IP to interconnect electronic point-of-sale terminals and for cable television receivers.
- Typically, a single IP address is assigned to each host. A more flexible arrangement is to allow multiple IP addresses per host. This, of course, increases the demand for IP addresses.

To meet these addressing needs, IPv6 uses 128-bit addresses instead of the 32-bit addresses of IPv4. This is an increase of address space by a factor of 2^{96}! Even if addresses are very inefficiently allocated, this address space seems secure.

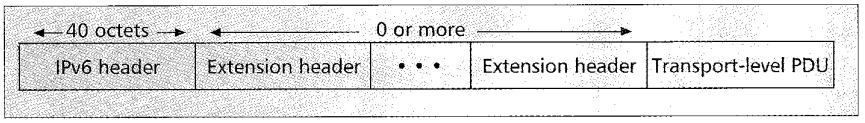


Figure 2. *IPv6 PDU general form.*

PERFORMANCE

Both LANs and wide area networks (WANs) have progressed to ever-greater data rates, pushing into hundreds of megabits per second, with plans for gigabit LANs and WANs. In addition, as more services, especially graphics-based services, become available over the Internet, we can expect that the ratio of external traffic (traffic that leaves the local network) to internal traffic will rise. With these immense speeds and the increased load, it is critical that routers perform their functions as rapidly as possible. The router should be able to process and forward IP datagrams fast enough to fully utilize its high-speed links and to keep up with the traffic flow. A major factor is the hardware platform itself, but the design of the IP also plays a critical role.

Three aspects of IPv6 design contribute to meeting performance requirements:

- The number of fields in the IPv6 packet header are reduced from IPv4. A number of IPv6 options are placed in separate optional headers located between the IPv6 header and the transport-layer header. Most of these optional headers are not examined or processed by any router on the packet's path. This simplifies and speeds up router processing of IPv6 packets compared to IPv4 datagrams.[3] It also makes it easier to add additional options.
- The IPv6 packet header is fixed-length whereas the IPv4 header is variable-length. Again, the IPv6 design simplifies processing.
- Packet fragmentation is not permitted by IPv6 routers, although it is in IPv4. In IPv6, fragmentation may only be performed by the source.

NETWORK SERVICE

It should be possible to associate packets with particular service classes, perform the routing function on the basis of those classes, and allow the networks along the route to make use of this class information. In particular, it is important to be able to support real-time services and to specify priority levels to determine discard strategy in the event of congestion. IPv4 provides minimal assistance in this area. IPv6 enables the labeling of packets belonging to a particular traffic *flow* for which the sender requests special handling. This aids in the support of specialized traffic such as real-time video.

ADDRESSING FLEXIBILITY

IPv4 is best employed for unicast addressing: a single address bit pattern corresponds to a single host. Other forms of addressing are poorly supported, partly because the address size is limited to 32 bits and partly because no provision is made for certain addressing modes. IPv6 includes the concept of an *anycast address*, for which a packet is delivered to just one of a set of nodes. The scalability of multicast routing is improved by adding a scope field to multicast addresses.

SECURITY CAPABILITIES

IPv4 provides no security capabilities other than an optional security label field. Although end-to-end security can be provided at the application level, there is now support for a standardized IP-level security service which any application can use without providing security features in that application. For example, an IP-level security service can be used to create virtual secure networks across the Internet or any public internetwork. IPv6 provides a range of security capabilities; in particular, IPv6 includes features that support authentication and privacy. As was mentioned, these features can also be incorporated into IPv4.

RFC Number	Title	Date
	(a) IPv6	
1752	The Recommendation for the IP Next Generation Protocol	January 1995
1809	Using the Flow Label in IPv6	June 1995
1881	IPv6 Address Allocation Management	December 1995
1883	Internet Protocol, Version 6 Specification	December 1995
1884	IP Version 6 Addressing Architecture	December 1995
1885	Internet Control Message Protocol (ICMPv6) for the Internet Protocol Version 6 (IPv6) Specification	December 1995
1886	DNS Extensions to Support IP Version 6	December 1995
1887	An Architecture for IPv6 Unicast Address Allocation	December 1995
1897	IPv6 Testing Address Allocation	January 1996
1924	A Compact Representation of IPv6 Addresses	April 1996
1933	Transition Mechanisms for IPv6 Hosts and Routers	April 1996
	(b) IPv4 and IPv6 Security	
1825	Security Architecture for the Internet Protocol	August 1995
1826	IP Authentication Header	August 1995
1827	IP Encapsulating Security Payload (ESP)	August 1995
1828	IP Authentication Using Keyed MD5	August 1995
1829	The ESP DES-CBC Transform	August 1995
1851	The ESP Triple DES Transform	September 1995
1852	IP Authentication Using Keyed SHA	September 1995

Table 1. *New IP RFCs.*

IPv6 FORMATS AND FUNCTIONS

An IPv6 protocol data unit (known as a packet) has the general form shown in Fig. 2.

The only header that is required

[3] *The protocol data unit for IPv6 is referred to as a packet rather than a datagram, which is the term used for IPv4 PDUs.*

is referred to simply as the *IPv6 header*. This is of fixed size with a length of 40 octets, compared to 20 octets for the mandatory portion of the IPv4 header. The following extension headers have been defined:

- Hop-by-hop options header: Defines special options that require hop-by-hop processing
- Routing header: Provides extended routing, similar to IPv4 source routing
- Fragment header: Contains fragmentation and reassembly information
- Authentication header: Provides packet integrity and authentication
- Encapsulating security payload header: Provides privacy
- Destination options header: Contains optional information to be examined by the destination node

The IPv6 standard recommends that, when multiple extension headers are used, the IPv6 headers appear in the following order:

- IPv6 header: Mandatory, must always appear first.
- Hop-by-hop options header
- Destination options header: For options to be processed by the first destination that appears in the IPv6 destination address field plus subsequent destinations listed in the routing header.
- Routing header
- Fragment header
- Authentication header
- Encapsulating security payload header
- Destination options header: For options to be processed only by the final destination of the packet

Figure 3 shows an example of an IPv6 packet that includes an instance of each header. Note that the IPv6 header and each extension header include a next header field. This field identifies the type of the header immediately following. If the next header is an extension header, then this field contains the type identifier of that header; otherwise, this field contains the protocol identifier of the upper-layer protocol using IPv6 (typically a transport-level protocol), using the same values as the IPv4 protocol field. In the figure, the upper-layer protocol is TCP, so the upper-layer data carried by the IPv6 packet consists of a TCP header followed by a block of application data.

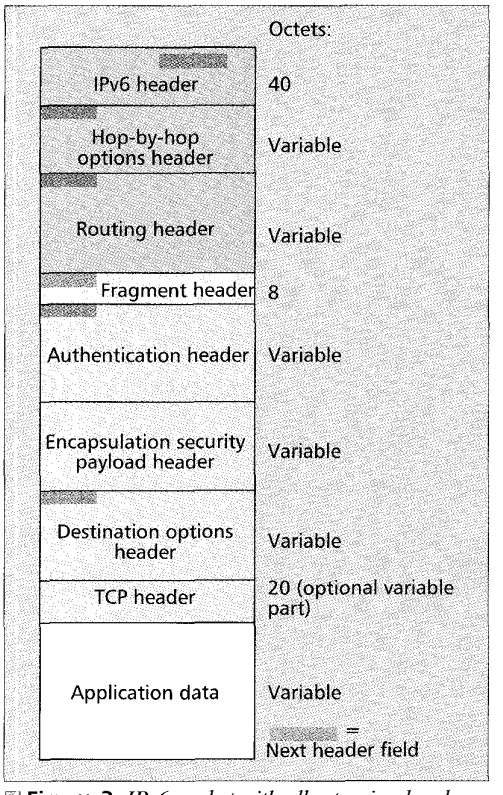


Figure 3. *IPv6 packet with all extension headers.*

We first look at the main IPv6 header, and then examine each of the extensions in turn.

IPv6 Header

The IPv6 header has a fixed length of 40 octets, consisting of the following fields (Fig. 4):

- Version (4 bits): IP version number; the value is 6.
- Priority (4 bits): Priority value, discussed below.
- Flow label (24 bits): May be used by a host to label those packets for which it is requesting special handling by routers within a network; discussed below.
- Payload length (16 bits): Length of the remainder of the IPv6 packet following the header, in octets. In other words, this is the total length of all the extension headers plus the transport-level PDU.
- Next header (8 bits): Identifies the type of header immediately following the IPv6 header.
- Hop limit (8 bits): The remaining number of allowable hops for this packet. The hop limit is set to some desired maximum value by the source, and decremented by 1 by each node that forwards the packet. The packet is discarded if hop limit is decremented to zero. This is a simplification over the processing implied for the time-to-live field of IPv4. The consensus was that the extra effort in accounting for time intervals in IPv4 added no significant value to the protocol. In fact, IPv4 routers, as a general rule, treat the time-to-live field as a hop limit field.
- Source address (128 bits): The address of the originator of the packet.
- Destination address (128 bits): The address of the intended recipient of the packet. This may not in fact be the intended ultimate destination if a routing header is present, as explained below.

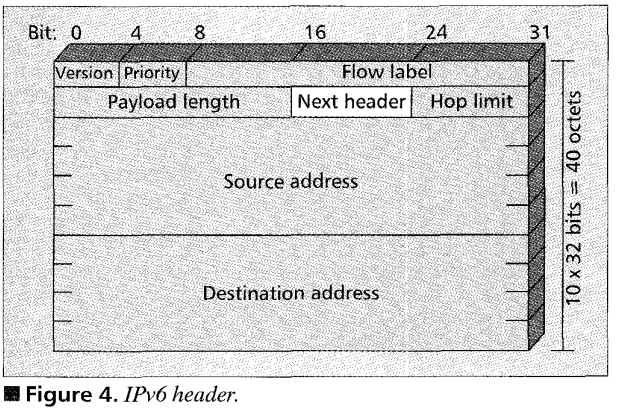


Figure 4. *IPv6 header.*

Although the IPv6 header is longer than the mandatory portion of the IPv4 header (40 octets versus 20 octets), it contains fewer fields (8 versus 12). Thus, routers have less processing to do per header, which should speed up routing.

Priority Field

The 4-bit priority field enables a source to identify the desired transmit and delivery priority of each packet relative to other packets from the same source. In fact, the field enables the source to identify two separate priority-related characteristics of each packet. First, packets are classified as being part of traffic for which the source is either providing congestion control or not; and second, packets are assigned one of eight levels of relative priority within each classification.

Congestion-Controlled-Traffic — refers to traffic for which the source "backs off" in response to congestion; an

example is TCP. Let us consider what this means. If there is congestion in the network, TCP segments will take longer to arrive at the destination, and hence acknowledgments from the destination back to the source will take longer. As congestion increases, it becomes necessary for segments to be discarded en route. The discarding could be done by a router when that router experiences buffer overflow, or it could be done by an individual network along the route when a switching node within the network becomes congested. Whether it is a data segment or an acknowledgment segment, the effect is that TCP does not receive an acknowledgment to its transmitted data segment. TCP responds by retransmitting the segment, and also by reducing the flow of segments it generates.

The IPv6 standard defines a flow as a sequence of packets sent from a particular source to a particular (unicast or multicast) destination for which the source desires special handling by the intervening routers.

The nature of congestion-controlled traffic is that a variable amount of delay in the delivery of packets, and even for packets to arrive out of order, is acceptable. IPv6 defines the following categories of congestion-controlled traffic, in decreasing priority:

- Internet control traffic: This is the most important traffic to deliver, especially during times of high congestion. For example, routing protocols such as Open Shortest Path First (OSPF) and BGP need to receive updates concerning traffic conditions so that they can adjust routes to try to relieve congestion. Management protocols such as Simple Network Management Protocol (SNMP) need to be able to report congestion to network management applications, perform dynamic reconfiguration, and alter performance-related parameters to cope with congestion.
- Interactive traffic: After Internet control traffic, the most important traffic to support is interactive traffic, such as on-line user-to-host connections. User efficiency is critically dependent on rapid response time during interactive sessions, so delay must be minimized.
- Attended bulk transfer: These are applications that may involve the transfer of a large amount of data; during these, the user is generally waiting for the transfer to complete. This category differs from interactive traffic in the sense that the user is prepared to accept a larger amount of delay for the bulk transfer than experienced during an interactive dialogue. A good example of this is File Transfer Protocol (FTP.) Another example is Hypertext Transfer Protocol (HTTP), which supports Web browser-server interaction.[4]
- Unattended data transfer: These are applications that are initiated by a user but not expected to be delivered instantly. Generally, the user does not wait for the transfer to be complete, but goes on to other tasks. The best example of this category is electronic mail.
- Filler traffic: This is traffic expected to be handled in the background, when other forms of traffic have been delivered. USENET messages are good examples.
- Uncharacterized traffic: If the upper-layer application gives IPv6 no guidance about priority, then traffic is assigned this lowest-priority value.

[4] *In the case of HTTP, the duration of the underlying connection may be too brief for the sender to receive feedback; therefore, HTTP may not do congestion control.*

Non-Congestion-Controlled Traffic — This is traffic for which a constant data rate and a constant delivery delay, or at least relatively smooth data rate and delivery delay, are desirable. Examples are real-time video and audio, for which it makes no sense to retransmit discarded packets; further, it is important to maintain smooth delivery flow. Eight levels of priority are allocated for this type of traffic from the lowest priority, 8 (most willing to discard), to the highest priority, 15 (least willing to discard). In general, the criterion is how much the quality of the received traffic will deteriorate in the face of some dropped packets. For example, low-fidelity audio, such as a telephone voice conversation, would typically be assigned a high priority. The reason is that the loss of a few packets of audio is readily apparent as clicks and buzzes on the line. On the other hand, a high-fidelity video signal contains a fair amount of redundancy, and the loss of a few packets will probably not be noticeable; therefore, this traffic is assigned a relatively low priority.

Please note that there is no priority relationship implied between the congestion-controlled priorities on one hand and the non-congestion-controlled priorities on the other. Priorities are relative only within each category.

Flow Label

The IPv6 standard defines a flow as a sequence of packets sent from a particular source to a particular (unicast or multicast) destination for which the source desires special handling by the intervening routers. A flow is uniquely identified by the combination of a source address and a nonzero 24-bit flow label. Thus, all packets that are to be part of the same flow are assigned the same flow label by the source.

From the source's point of view, a flow typically will be a sequence of packets that are generated from a single application instance at the source and have the same transfer service requirements. A flow may comprise a single TCP connection or even multiple TCP connections; an example of the use of multiple TCP connections is a file transfer application, which could have one control connection and multiple data connections. A single application may generate a single flow or multiple flows. An example of the use of multiple flows is multimedia conferencing, which might have one flow for audio and one for graphic windows, each with different transfer requirements in terms of data rate, delay, and delay variation.

From the router's point of view, a flow is a sequence of packets that share attributes which affect how they are handled by the router. These include path, resource allocation, discard requirements, accounting, and security attributes. The router may treat packets from different flows differently in a number of ways, including allocating different buffer sizes, giving different precedence in terms of forwarding, and requesting different qualities of service from subnetworks.

There is no special significance to any particular flow label; instead, the special handling to be provided for a packet flow must be declared in some other way. For example, a source might negotiate or request special handling ahead of time from routers by means of a control protocol, or at transmission time by information in one of the extension headers in the packet, such as the hop-by-hop options header. Examples of special handling that might be requested include some sort of nonde-

fault quality of service and some form of real-time service.

In principle, all of a user's requirements for a particular flow could be defined in an extension header and included with each packet. If we wish to leave the concept of flow open to include a wide variety of requirements, this design approach could result in very large packet headers. The alternative, adopted for IPv6, is the flow label, in which the flow requirements are defined prior to flow commencement and a unique flow label is assigned to the flow. In this case, the router must save flow requirement information about each flow.

The following rules apply to the flow label:

- Hosts or routers that do not support the flow label field must set the field to zero when originating a packet, pass the field unchanged when forwarding a packet, and ignore the field when receiving a packet.
- All packets originating from a given source with the same nonzero flow label must have the same destination address, source address, priority, hop-by-hop options header contents (if this header is present), and routing header contents (if this header is present). The intent is that a router can decide how to route and process the packet by simply looking up the flow label in a table, without examining the rest of the header.
- The source assigns a flow label to a flow. New flow labels must be chosen (pseudo-) randomly and uniformly in the range 1 to $2^{24} - 1$, subject to the restriction that a source must not reuse a flow label for a new flow within the lifetime of the existing flow.

This last point requires some elaboration. The router must maintain information about the characteristics of each active flow that may pass through it, presumably in some sort of table. In order to be able to forward packets efficiently and rapidly, table lookup must be efficient. One alternative is to have a table with 2^{24} (about 16 million) entries, one for each possible flow label; this imposes an unnecessary memory burden on the router. Another alternative is to have one entry in the table per active flow, include the flow label with each entry, and require the router to search the entire table each time a packet is encountered. This imposes an unnecessary processing burden on the router. Instead, most router designs are likely to use some sort of hash table approach. With this approach a moderate-sized table is used, and each flow entry is mapped into the table using a hashing function on the flow label. The hashing function might simply be the low-order few bits (say 10 or 12) of the flow label or some simple calculation on the 24 bits of the flow label. In any case, the efficiency of the hash approach typically depends on the flow labels being uniformly distributed over their possible range; hence the third requirement above.

IPv6 Addresses

IPv6 addresses are 128 bits in length. Addresses are assigned to individual interfaces on nodes, not to the nodes themselves.[5] A single interface may have multiple unique unicast addresses. Any of the unicast addresses associated with a node's interface may be used to uniquely identify that node.

The combination of long addresses and multiple addresses per interface enables improved routing efficiency over IPv4. In IPv4, addresses generally do not have a structure that assists routing, and therefore a router may need to maintain huge table of routing paths. Longer internet addresses allow for aggregating addresses by hierarchies of network, access provider, geography, corporation, and so on. Such aggregation should make for smaller routing tables and faster table lookups. The allowance for multiple addresses per interface would allow a subscriber using multiple access providers across the same interface to have separate addresses aggregated under each provider's address space.

The first field of any IPv6 address is the variable-length format prefix, which identifies various categories of addresses. Table 2 indicates the current allocation of addresses based on the format prefix, and Fig. 5 shows some IPv6 address formats.

IPv6 allows three types of addresses:

- Unicast: An identifier for a single interface. A packet sent to a unicast address is delivered to the interface identified by that address.
- Anycast: An identifier for a set of interfaces (typically belonging to different nodes). A packet sent to an anycast address is delivered to one of the interfaces identified by that address (the "nearest" according to the routing protocols' measure of distance).
- Multicast: An identifier for a set of interfaces (typically belonging to different nodes). A packet sent to a multicast address is delivered to all interfaces identified by that address.

Unicast Addresses

Unicast addresses may be structured in a number of ways. The following have been identified: provider-based global, link-local, site-local, IPv4-compatible IPv6, and loopback.

A *provider-based global unicast address* provides for global addressing across the entire universe of connected hosts. The address has five fields after the format prefix (Fig. 5a):

- Registry ID: Identifies the registration authority which assigns the provider portion of the address

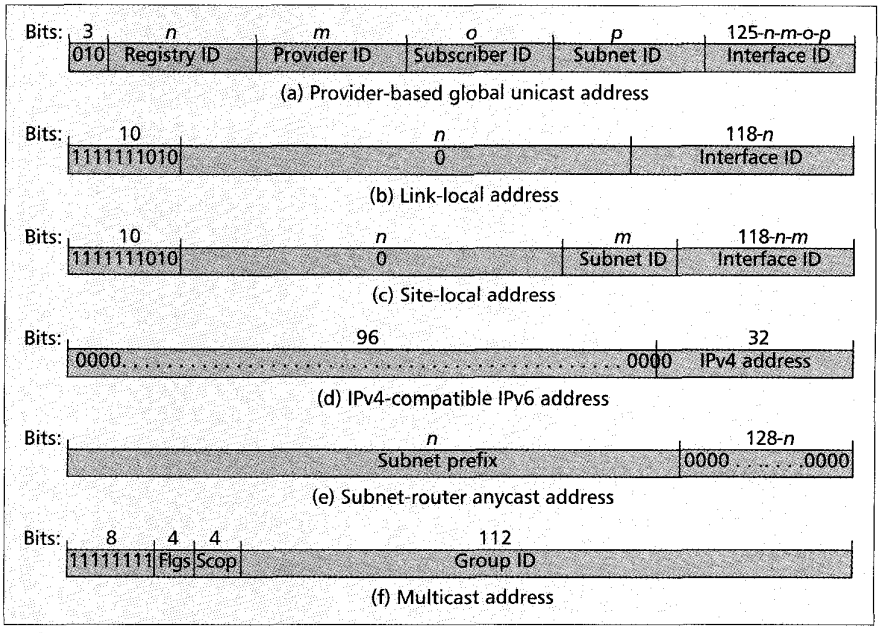


Figure 5. *IPv6 address formats.*

[5] *In IPv6, a node is any device that implements IPv6; this includes hosts and routers.*

Allocation Space	Prefix (binary)	Fraction of Address Space
Reserved	0000 0000	1/256
Unassigned	0000 0001	1/256
Reserved for NSAP Allocation	0000 001	1/128
Reserved for IPX Allocation	0000 010	1/128
Unassigned	0000 011	1/128
Unassigned	0000 1	1/32
Unassigned	0001	1/16
Unassigned	001	1/8
Provider-Based Unicast Address	010	1/8
Unassigned	011	1/8
Reserved for Geographic-Based Unicast Addresses	100	1/8
Unassigned	101	1/8
Unassigned	110	1/8
Unassigned	1110	1/16
Unassigned	1111 0	1/32
Unassigned	1111 10	1/64
Unassigned	1111 110	1/128
Unassigned	1111 1110 0	1/512
Link Local Use Addresses	1111 1110 10	1/1024
Site Local Use Addresses	1111 1110 11	1/1024
Multicast Addresses	1111 1111	1/256

Table 2. *Address allocation.*

- Provider ID: A specific internet service provider which assigns the subscriber portion of the address
- Subscriber ID: Distinguishes among multiple subscribers attached to the provider portion of the address
- Subnet ID: A topologically connected group of nodes within the subscriber network
- Interface ID: Identifies a single node interface among the group of interfaces identified by the subnet prefix.

At present, there is no fixed length assigned to any of these fields. However, from the point of view of a network manager or network designer responsible for a subscriber's installation, the subnet ID and interface ID are the principal concerns. The subscriber can deal with these fields in a number of ways. One possibility for a LAN-based installation is to use a 48-bit interface field and use the IEEE 802 medium access control (MAC) address for that field. The remainder of the available bits would then be the subnet ID field, identifying a particular LAN at that subscriber site.

IPv6 accommodates local-use unicast addresses. Packets with such addresses can only be routed locally, that is, within a subnetwork or set of subnetworks of a given subscriber.

Two types of local-use addresses have been defined: link-local and site-local.

Link-local addresses are to be used for addressing on a single link or subnetwork (Fig. 5b). They cannot be integrated into the global addressing scheme. Examples of their use include auto-address configuration, discussed presently, and neighbor discovery.

Site-local addresses are designed for local use but formatted in such a way that they can later be integrated into the global address scheme. The advantage of such addresses is that they can be used immediately by an organization that expects to transition to the use of global addresses. The structure of these addresses (Fig. 5c) is such that the meaningful portion of the address is confined to the low-order bits not used for global addresses. The remaining bits consist of a local-use format prefix (1111 1110 10 or 1111 1110 11) followed by a zero field. When an organization is ready for global connection, these remaining bits can be replaced with a global prefix (e.g., 010 + registry ID + provider ID + subscriber ID).

A key issue in deploying IPv6 is the transition from IPv4 to IPv6. It is not practical to simply replace all IPv4 routers in the Internet or a private internet with IPv6 routers and replace all IPv4 addresses with IPv6 addresses. Instead, there will be a lengthy transition period when IPv6 and IPv4 must coexist, and therefore IPv6 addresses and IPv4 addresses must coexist [2]. *IPv4-compatible IPv6 addresses* accommodate this coexistence period. This form of address consists of a 32-bit IPv4 address in the lower-order 32 bits prefixed by 96 zeroes (Fig. 5d). The IPv4-compatible IPv6 address supports a technique known as *automatic tunneling*. Tunneling may be employed to forward such traffic through IPv4 routing topologies. In essence, an IPv6 packet is encapsulated in an IPv4 datagram. With the use of an IPv4-compatible IPv6 address for the destination node, the tunneling may be automated in the sense that the destination IPv4 address may be derived from the IPv6 address, avoiding the need to explicitly configure the mapping from an IPv6 address to an IPv4 address.

Full coexistence can be maintained as long as all IPv6 nodes employ an IPv4-compatible address. Once more general IPv6 addresses come into use, coexistence will be more difficult to maintain.

The unicast address 0:0:0:0:0:0:0:1 is called the *loopback address*.[6] It may be used by a node to send an IPv6 packet to itself; such packets are not to be sent outside a single node.

Anycast Addresses

An anycast address enables a source to specify that it wants to contact any one node from a group of nodes via a single address. A packet with such an address will be routed to the nearest interface in the group, according to the router's measure of distance. An example of the use of an anycast address is within a routing header to specify an intermediate address along a route. The anycast address could refer to the group of routers associated with a particular provider or particular subnet, thus dictating that the packet be routed through that provider or internet in the most efficient manner.

Anycast addresses are allocated from the same address space as unicast addresses. Thus, members of an anycast group must be configured to recognize that address, and routers must be configured to be able to map an anycast

[6] *The preferred form for representing IPv6 addresses as text strings is x:x:x:x:x:x:x:x, where each x is the hexadecimal value of a 16-bit portion of the address; it is not necessary to show leading zeros but each position must have at least one digit.*

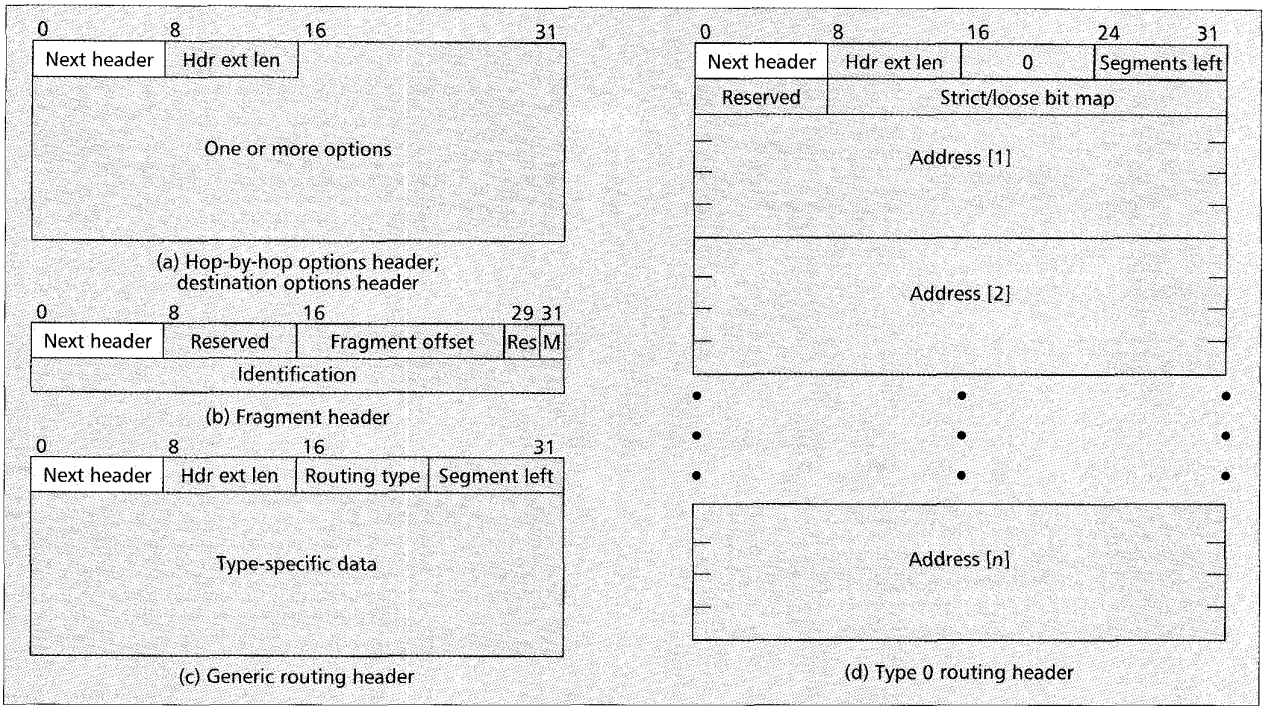


Figure 6. *IPv6 extension headers.*

address to a group of unicast interface addresses.

One particular form of anycast address, the *subnet-router anycast address*, is predefined (Fig. 5e). The subnet prefix field identifies a specific subnetwork. For example, in a provider-based global address space, the subnet prefix is of the form (010 + registry ID + provider ID + subscriber ID + subnet ID). Thus, the anycast address is identical to a unicast address for an interface on this subnetwork, with the interface ID portion set to zero. Any packet sent to this address will be delivered to one router on the subnetwork; all that is required is to insert the correct interface ID into the anycast address to form the unicast destination address.

Multicast Addresses

IPv6 includes the capability to address a predefined group of interfaces with a single multicast address. A packet with a multicast address is to be delivered to all members of the group. A multicast address consists of an 8-bit format prefix of all ones, a 4-bit flags field, a 4-bit scope field, and a 112-bit group ID.

At present, the flags field consists of 3 zero bits followed by a T bit with:

- $T = 0$: Indicates a permanently assigned or well-known multicast address, assigned by the global internet numbering authority
- $T = 1$: Indicates a nonpermanently-assigned, or transient, multicast address

The scope value is used to limit the scope of the multicast group. The values are:

0: reserved
1: node -local
2: link-local
3: unassigned
4: unassigned
5: site-local
6: unassigned
7: unassigned
8: organization-local
9: unassigned
10: unassigned
11: unassigned
12: unassigned
13: unassigned
14: global
15: reserved

The group ID identifies a multicast group, either permanent or transient, within the given scope. In the case of a permanent multicast address, the address itself is independent of the scope field, but that field limits the scope of addressing for a particular packet. For example, if the "NTP servers group" is assigned a permanent multicast address with a group ID of 43 (hex), then:

FF05:0:0:0:0:0:0:43 means all NTP servers at the same site as the sender.
FF0E:0:0:0:0:0:0:43 means all NTP servers in the internet.

Non-permanently-assigned multicast addresses are only meaningful within a given scope, enabling the same group ID to be reused, with different interpretations, at different sites.

Multicasting is a useful capability in a number of contexts. For example, it allows hosts and routers to send neighbor discovery messages only to those machines that are registered to receive them, removing the necessity for all other machines to examine and discard irrelevant packets. As another example, most LANs provide a natural broadcast capability. A multicast address can be assigned that has a scope of link-local with a group ID configured on all nodes on the LAN to be a subnet broadcast address.

Hop-by-Hop Options Header

The hop-by-hop options header carries optional information that, if present, must be examined by every router along the path. This header consists of (Fig. 6a):

- Next header (8 bits): Identifies the type of header immediately following this header
- Header extension length (8 bits): Length of this header in 64-bit units, not including the first 64 bits
- Options: A variable-length field consisting of one or

more option definitions. Each definition is in the form of three subfields: option type (8 bits), which identifies the option; length (8 bits), which specifies the length of the option data field in octets; and option data, which is a variable-length specification of the option.

It is actually the lowest-order five bits of the option type field that are used to specify a particular option. The high-order two bits indicate the action to be taken by a node that does not recognize this option type, as follows:

00 – Skip over this option and continue processing the header.
01 – Discard the packet.
10 – Discard the packet and send an ICMP Parameter Problem, Code 2, message to the packet's source address, pointing to the unrecognized option type.
11 – Discard the packet and, only if the packet's destination address is not a multicast address, send an ICMP Parameter Problem, Code 2, message to the packet's source address, pointing to the unrecognized option type.

The third highest-order bit specifies whether the option data field does not change (0) or may change (1) en route from source to destination. Data that may change must be excluded from authentication calculations, as discussed in later.

These conventions for the option type field also apply to the destination options header.

In the IPv6 standard, only one option is so far specified: the jumbo payload option, used to send IPv6 packets with payloads longer than $2^{16} = 65{,}536$ octets. The option data field of this option is 32 bits long and gives the length of the packet in octets, excluding the IPv6 header. For such packets, the payload length field in the IPv6 header must be set to zero, and there must be no fragment header. With this option, IPv6 supports packet sizes up to more than 4 billion octets. This facilitates the transmission of large video packets and enables IPv6 to make the best use of available capacity over any transmission medium.

By implementing security at the IP level, an organization can ensure secure networking not only for applications that have security mechanisms but for the many security-ignorant applications.

Fragment Header

In IPv6, fragmentation may only be performed by source nodes, not by routers along a packet's delivery path. To take full advantage of the internetworking environment, a node must perform a path discovery algorithm that enables it to learn the smallest maximum transmission unit (MTU) supported by any subnetwork on the path. In other words, the path discovery algorithm enables a node to learn the MTU of the "bottleneck" subnetwork on the path. With this knowledge, the source node will fragment, as required, for each given destination address. Otherwise, the source must limit all packets to 576 octets, which is the minimum MTU that must be supported by each subnetwork.

The fragment header consists of (Fig. 6b):

- Next header (8 bits): Identifies the type of header immediately following this header.
- Reserved (8 bits): For future use.
- Fragment Offset (13 bits): indicates where in the original packet the payload of this fragment belongs. It is measured in 64-bit units. This implies that fragments (other than the last fragment) must contain a data field that is a multiple of 64 bits long.
- Res (2 bits): Reserved for future use.
- M Flag (1 bit): 1 = more fragments; 0 = last fragment.
- Identification (32 bits): Intended to uniquely identify the original packet. The identifier must be unique for the packet's source address and destination address for the time during which the packet will remain in the internet.

The fragmentation algorithm is the same as that used in IPv4.

Routing Header

The routing header contains a list of one or more intermediate nodes to be visited on the way to a packet's destination. All routing headers start with a 32-bit block consisting of four 8-bit fields, followed by routing data specific to a given routing type (Fig. 6c). The four 8-bit fields are:

- Next header: Identifies the type of header immediately following this header.
- Header extension length: Length of this header in 64-bit units, not including the first 64 bits.
- Routing type: Identifies a particular routing header variant. If a router does not recognize the routing type value, it must discard the packet.
- Segments left: Number of explicitly listed intermediate nodes still to be visited before reaching the final destination.

In addition to this general header definition, RFC 1883 defines the Type 0 routing header (Fig. 6d). In addition to the leading four 8-bit fields, the Type 0 header includes a 24-bit strict/loose bit map. The bits of the field are considered to be numbered from left to right (bit 0 through bit 23), with each bit corresponding to one of the hop. Each bit indicates whether the corresponding next destination address must be a neighbor of the preceding address (1 = strict, must be a neighbor; 0 = loose, need not be a neighbor).

When using the Type 0 routing header, the source node does not place the ultimate destination address in the IPv6 header. Instead, that address is the last address listed in the routing header (address [n] in Fig. 6d), and the IPv6 header contains the destination address of the first desired router on the path. The routing header will not be examined until the packet reaches the node identified in the IPv6 header. At that point, the packet IPv6 and routing header contents are updated, and the packet is forwarded. The update consists of placing the next address to be visited in the IPv6 header and decrementing the segments left field in the routing header.

IPv6 requires an IPv6 node to reverse routes in a packet it receives containing a routing header in order to return a packet to the sender. With this in mind, let us consider a set of examples from [3], which illustrates the power and flexibility of the routing header feature. Figure 7 shows a configuration in which two hosts are connected by two providers, and the two providers are in turn connected by a wireless network. In the scenarios to follow, address sequences are shown in the following format:

SRC, I_1, I_2, ..., I_n, DST

where SRC is the source address listed in the IPv6 header, I_i is the ith intermediate address, and DST in the ultimate destination. If there is no routing header, DST appears in the IPv6 header; otherwise, it is the last address in the routing header. Now consider three scenarios in which H1 sends a packet to H2.

- No routing header is used. H1 sends a packet to H2 containing the sequence (H1, H2). H2 replies with a packet containing (H2, H1). In this case, either provider could be used on each transfer.
- H1 wants to enforce a policy that all traffic between itself and H2 can only use P1. It constructs a packet with the sequence (H1, P1, H2) and dictates strict routing. This ensures that when H2 replies to H1, it will use the sequence (H2, P1, H1) with strict routing, thus enforcing H1's policy of using P1.
- H1 becomes mobile and moves to provider PR. It could maintain communication (not breaking any TCP connections) with H2 by sending packets with (H1, PR, P1, H2). This would ensure that H2 would enforce the policy of using P1 by replying with (H2, P1, PR, H1).

These examples show the ability in IPv6 to select a particular provider, to maintain connections while mobile, and to route packets to new addresses dynamically.

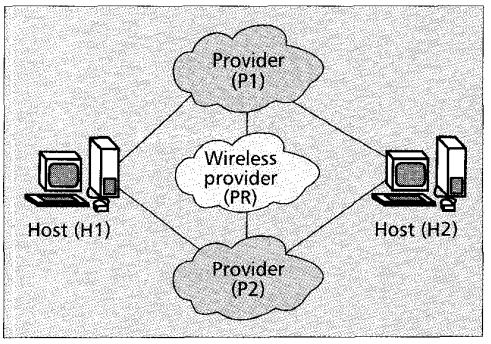


Figure 7. *Example routing configuration.*

Destination Options Header

The destination options header carries optional information that, if present, is examined only by the packet's destination node. The format of this header is the same as that of the hop-by-hop options header (Fig. 6a).

IPv6 Security

The Internet community has developed application-specific security mechanisms in a number of application areas, including electronic mail (Privacy Enhanced Mail, PGP), network management (SNMPv2 security), web access (Secure HTTP, Secure Sockets Layer), and others. However, users have some security concerns that cut across protocol layers. For example, an enterprise can run a secure, private TCP/IP network by disallowing links to untrusted sites, encrypting packets that leave the premises, and authenticating packets that enter the premises. By implementing security at the IP level, an organization can ensure secure networking not only for applications that have security mechanisms but for the many security-ignorant applications.

IP-level security encompasses two functional areas: authentication and privacy. The authentication mechanism ensures that a received packet was in fact transmitted by the party identified as the source in the packet header. In addition, this mechanism ensures that the packet has not been altered in transit. The privacy facility enables communicating nodes to encrypt messages to prevent eavesdropping by third parties.

In August 1995, the IETF published five security-related proposed standards that define a security capability at the internet level. The documents provide:

- RFC 1825: An overview of a security architecture
- RFC 1826: Description of a packet authentication extension to IP
- RFC 1828: A specific authentication mechanism
- RFC 1827: Description of a packet encryption extension to IP
- RFC 1829: A specific encryption mechanism

Support for these features is mandatory for IPv6 and optional for IPv4. In both cases, the security features are implemented as extension headers that follow the main IP header. The extension header for authentication is known as the *authentication header*; that for privacy, the *encapsulating security payload* (ESP) header.

Security Associations

A key concept that appears in both the authentication and privacy mechanisms for IP is the security association. An association is a one-way relationship between a sender and a receiver. If a peer relationship is needed for two-way secure exchange, then two security associations are required.

A security association is uniquely identified by an internet destination address and a security parameter index (SPI). Hence, in any IP packet,[7] the security association is uniquely identified by the destination address in the IPv4 or IPv6 header and the SPI in the enclosed extension header (authentication header, AH, or ESP header).

A security association is normally defined by the following parameters:

- Authentication algorithm and algorithm mode being used with the IP AH (required for AH implementations)
- Key(s) used with the authentication algorithm in use with the AH (required for AH implementations)
- Encryption algorithm, algorithm mode, and transform being used with the IP ESP (required for ESP implementations)
- Key(s) used with the encryption algorithm in use with the ESP (required for ESP implementations)
- Presence/absence and size of a cryptographic synchronization or initialization vector field for the encryption algorithm (required for ESP implementations)
- Authentication algorithm and mode used with the ESP transform, if any is in use (recommended for ESP implementations)
- Authentication key(s) used with the authentication algorithm that is part of the ESP transform, if any (recommended for ESP implementations)
- Lifetime of the key or time when key change should occur (recommended for all implementations)
- Lifetime of this security association (recommended for all implementations)
- Source address(es) of the security association; might be a wildcard address if more than one sending system shares the same security association with the destination (recommended for all implementations)
- Sensitivity level (e.g., secret or unclassified) of the protected data (required for all systems claiming to provide multilevel security, recommended for all other systems)

The key management mechanism used to distribute keys is coupled to the authentication and privacy mechanisms only by way of the SPI. Hence, authentication and privacy have been specified independent of any specific key management mechanism.

Authentication

The authentication header provides support for data integrity and authentication of IP packets. The authentication header consists of the following fields (Fig. 8):

- Next header (8 bits): Identifies the type of header immediately following this header
- Length (8 bits): Length of authentication data field in 32-bit words.

[7] *In the remainder of this section, the term "IP packet" refers to either an IPv4 datagram or an IPv6 packet.*

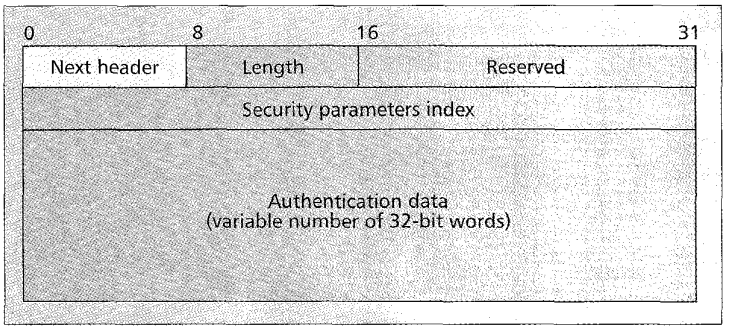


Figure 8. *Authentication header.*

- Reserved (16 bits): For future use
- Security parameters index (32 bits): Identifies a security association
- Authentication data (variable): An integral number of 32-bit words

The authentication data field contents will depend on the authentication algorithm specified. In any case, the authentication data is calculated over the entire IP packet, excluding any fields that may change in transit. Such fields are set to zero for purposes of calculation at both source and destination. The authentication calculation is performed prior to fragmentation at the source and after reassembly at the destination; hence, fragmentation-related fields can be included in the calculation.

For IPv4, the time to live and header checksum fields are subject to change and therefore set to zero for the authentication calculation. IPv4 options must be handled in accordance with the rule that any option whose value might change during transit must not be included in the calculation.

For IPv6, the hop limit field is the only field in the base IPv6 header subject to change; it is therefore set to zero for the calculation. For the hop-by-hop options and destination options headers, the option type field for each option contains a bit that indicates whether the option data field for this option may change during transit; if so, this option is excluded from the authentication calculation.

Authentication Using Keyed MD5 — RFC 1828 specifies the use of MD5 for authentication. The MD5 algorithm is performed over the IP packet plus a secret key by the source and then inserted into the IP packet. At the destination, the same calculation is performed on the IP packet plus the secret key and compared to the received value. This procedure provides both authentication and data integrity.

Specifically, the MD5 calculation is performed over the following sequence:

key, keyfill, IP packet, key, MD5fill

where

key: the secret key for this security association
keyfill: padding so that key + keyfill is an integer multiple of 512 bits
IP packet: with the appropriate fields set to zero
MD5fill: padding supplied by MD5 so that the entire block is a integer multiple of 512 bits

The IP authentication service can be used in several ways. Authentication can be provided directly between server and client workstations; the workstation can be on either the same network as the server or an external network. So long as the workstation and the server share a protected secret key, the authentication process is secure. Alternatively, a remote workstation authenticates itself to the corporate firewall, either for access to the entire internal network or because the requested server does not support the authentication feature.

Encapsulating Security Payload

The use of the ESP provides support for privacy and data integrity for IP packets. Depending on the user's requirements, this mechanism can be used to encrypt either a transport-layer segment (e.g., TCP, user data gram protocol — UDP, or ICMP), known as *transport-mode ESP*, or an entire IP packet, known as *tunnel-mode ESP*.

The ESP header begins with a 32-bit SPI, which identifies a security association. The remainder of the header, if any, may contain parameters dependent on the encryption algorithm being used. In general, the first part of the header, including the SPI and possibly some parameters, is transmitted in unencrypted (plaintext) form, while the remainder of the header, if any, is transmitted in encrypted (ciphertext) form.

Transport-Mode ESP — Transport-mode ESP is used to encrypt the data carried by IP. Typically, this data is a transport-layer segment, such as a TCP or UDP segment, which in turn contains application-level data. For this mode, the ESP header is inserted into the IP packet immediately prior to the transport-layer header (e.g., TCP, UDP, ICMP). In the case of IPv6, if a destination options header is present, the ESP header is inserted immediately prior to that header (see Fig. 3).

Transport-mode operation may be summarized as follows:

- At the source, the block of data consisting of a trailing portion of the ESP header plus the entire transport-layer segment is encrypted, and the plaintext of this block is replaced with its ciphertext to form the IP packet for transmission (Fig. 9a).
- This packet is then routed to the destination. Each intermediate router needs to examine and process the IP header plus any plaintext IP extension headers, but does not need to examine the ciphertext.
- The destination node examines and processes the IP header plus any plaintext IP extension headers. Then, on the basis of the SPI in the ESP header, the destination node decrypts the remainder of the packet to recover the plaintext transport-layer segment.

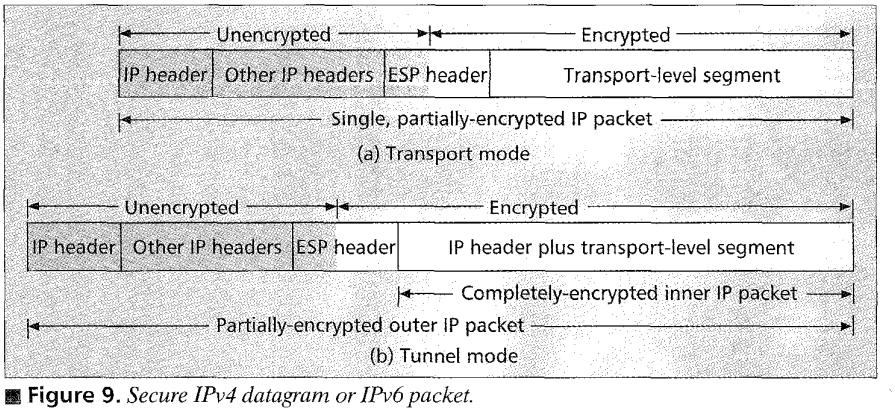
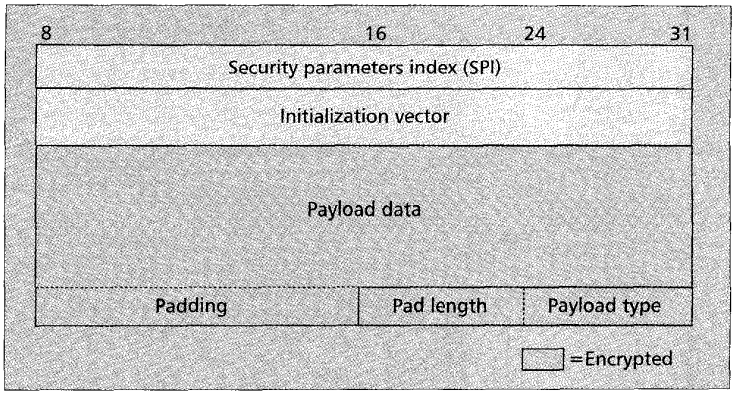


Figure 9. *Secure IPv4 datagram or IPv6 packet.*

Transport-mode operation provides privacy for any application that uses it, thus avoiding the need to implement privacy in every individual application. This mode of operation is also reasonably efficient, adding little to the total length of the IP packet. One drawback to this mode is that it is possible to do traffic analysis on the transmitted packets.

Tunnel-Mode ESP — Tunnel-mode ESP is used to encrypt an entire IP packet. For this mode, the ESP is prefixed to the packet and then the packet plus a trailing portion of the ESP header is encrypted. This method can be used to counter traffic analysis.



■ **Figure 10.** *Encapsulating security payload format.*

Because the IP header contains the destination address and possibly source routing directives and hop-by-hop option information, it is not possible to simply transmit the encrypted IP packet prefixed by the ESP header. Intermediate routes would be unable to process such a packet. Therefore, it is necessary to encapsulate the entire block (ESP header plus encrypted IP packet) with a new IP header that will contain sufficient information for routing but not for traffic analysis.

Whereas the transport mode is suitable for protecting connections between hosts that support the ESP feature, the tunnel mode is useful in a configuration that includes a firewall or other sort of security gateway which protects a trusted network from external networks. In this latter case, encryption occurs only between an external host and the security gateway or between two security gateways. This relieves hosts on the internal network of the processing burden of encryption and also simplifies the key distribution task by reducing the number of needed keys. Furthermore, it thwarts traffic analysis based on ultimate destination.

Consider a case in which an external host wishes to communicate with a host on an internal network protected by a firewall, and in which ESP is implemented in the external host and the firewalls. The following steps occur for transfer of a transport-layer segment from the external host to the internal host:

- The source prepares an inner IP packet with a destination address of the target internal host. This packet is prefixed by an ESP header; then the packet and a portion of the ESP header are encrypted. The resulting block is encapsulated with a new IP header (base header plus optional extensions such as routing and hop-by-hop options) whose destination address is the firewall; this forms the outer IP packet (Fig. 9b).
- The outer packet is routed to the destination firewall. Each intermediate router needs to examine and process the outer IP header plus any outer IP extension headers, but does not need to examine the ciphertext.
- The destination firewall examines and processes the outer IP header plus any outer IP extension headers. Then, on the basis of the SPI in the ESP header, the destination node decrypts the remainder of the packet to recover the plaintext inner IP packet. This packet is then transmitted in the internal network.
- The inner packet is routed through zero or more routers in the internal network to the destination host.

The ESP DES-CBC Transform — All implementations that claim conformance with the ESP specification must implement the DES-CBC (data encryption standard-cipher block chaining) method of encryption. In the CBC method, the data to be encrypted (plaintext) is processed as a sequence of 64-bit blocks. The input to the encryption algorithm is the XOR of the current plaintext block and the preceding ciphertext block; the same key is used for each block. In effect, this chains together the processing of the sequence of plaintext blocks. The input to the encryption function for each plaintext block bears no fixed relationship to the plaintext block. Therefore, repeating patterns of 64 bits are not exposed. To produce the first block of ciphertext, an initialization vector (IV) is XORed with the first block of plaintext. On decryption, the IV is XORed with the output of the decryption algorithm to recover the first block of plaintext.

Figure 10 shows the format of the ESP header plus payload using DES-CBC, and indicates which portions are encrypted. The fields are:

- Security parameters index (32 bits): Identifies a security association
- Initialization vector (variable): Input to the CBC algorithm and a multiple of 32 bits
- Payload data (variable): Prior to encryption, this field contains the block of data to be encrypted, which may be a transport-layer segment (transport mode) or an IP packet (tunnel mode)
- Padding: Prior to encryption, filled with unspecified data to align pad length and payload type fields at a 64-bit boundary
- Pad length (8 bits): The size of the unencrypted padding field
- Payload type (8 bits): Indicates the protocol type of the payload data field (e.g., IP, TCP)

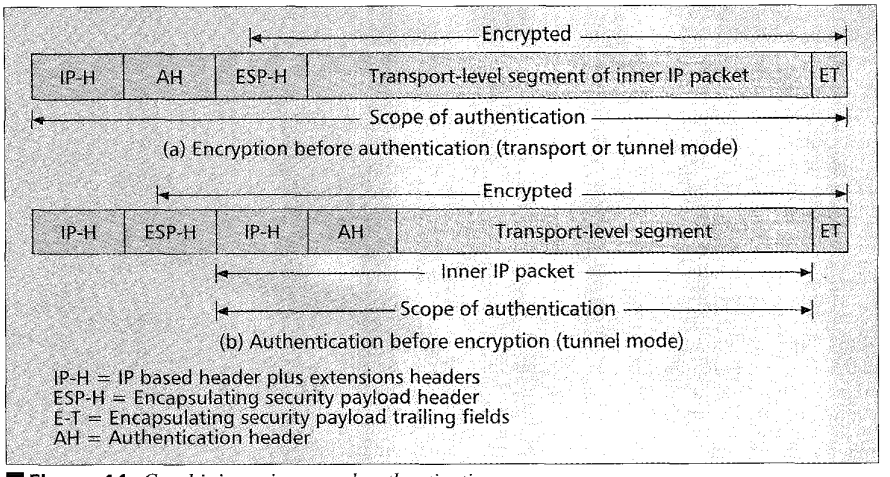
Note that the IV is transmitted in plaintext. This is not the most secure approach; however, for the purposes of this application it should provide acceptable security.

AUTHENTICATION PLUS PRIVACY

The two IP security mechanisms can be combined in order to transmit an IP packet that has both privacy and authentication. There are two approaches that can be used, based on the order in which the two services are applied.

Encryption Before Authentication — Figure 11a illustrates the case of encryption applied before authentication. In this case, the entire transmitted IP packet is authenticated, including both encrypted and unencrypted parts. In this approach, the user first applies ESP to the data to be protected, then prepends the authentication header and the plaintext IP header(s). There are actually two subcases:

- Transport-mode ESP: Authentication applies to the entire IP packet delivered to the ultimate destination, but only the transport-layer segment is protected by the privacy mechanism (encrypted).
- Tunnel-mode ESP: Authentication applies to the entire IP packet delivered to the outer IP destination address (e.g., a



■ **Figure 11.** *Combining privacy and authentication.*

firewall), and authentication is performed at that destination. The entire inner IP packet is protected by the privacy mechanism, for delivery to the inner IP destination.

Authentication Before Encryption — Figure 11b illustrates the case of authentication applied before encryption. This approach is only appropriate for tunnel-mode ESP. In this case, the authentication header is placed inside the inner IP packet. This inner packet is both authenticated and protected by the privacy mechanism.

As we have just seen, the functions of authentication and encryption can be applied in either order for tunnel-mode ESP. The use of authentication prior to encryption might preferable for several reasons. First, since the AH is protected by ESP, it is impossible for anyone to intercept the message and alter the AH without detection. Second, it may be desirable to store the authentication information with the message and the destination for later reference. It is more convenient to do this if the authentication information applies to the unencrypted message; otherwise, the message would have to be re-encrypted to verify the authentication information.

SUMMARY

The Internet Protocol (IP) has been the foundation of the Internet and virtually all multivendor private internetworks. This protocol is reaching the end of its useful life and a new protocol, known as IPv6 (IP version 6), has been defined to ultimately replace IP.

The driving motivation for the adoption of a new version of IP was the limitation imposed by the 32-bit address field in IPv4. In addition, IP is a very old protocol, and new requirements in the areas of security, routing flexibility, and traffic support have developed. To meet these needs, IPv6 has been defined, and includes functional and formatting enhancements over IPv4. In addition, a set of security specifications have been issued that can be used with both IPv4 and IPv6. With most of the technical details of these enhancements frozen, vendors may begin to move this capability into their product lines. As IPv6 is gradually deployed, the Internet and corporate networks will be rejuvenated, able to support the applications of the 21st century.

FURTHER READING

Two books devoted to IPv6 provide additional insight into this new protocol and to the IP security enhancements. *IPng: Internet Protocol Next Generation* [4] is a collection of chapters edited by the two co-chairs of the IETF/IPng process. This book was finished somewhat ahead of the standards process and so is not heavy on technical details of the actual specifications. Instead, the aim of the book is to explain the rationale behind the format and numerous features in IPv6, and to discuss the various applications and networking technologies that IPv6 can potentially support. *IPv6: The New Internet Protocol* [5], by a former chair of the Internet Activities Board, is a straightforward technical description of the various RFCs that together make up the IPv6 specification. The book provides a discussion of the purposes of various features and the operation of the protocol.

Two good discussions of the rationale for IPv6 are in [2] and [6].

To keep up to date on the latest developments, visit the IPng working group at http://playground.sun.com/pub/ipng/html/ipng-main.html.

REFERENCES

[1] W. Stallings, *Data and Computer Communications*, 5th Edition, Upper Saddle River, NJ: Prentice Hall, 1996.
[2] R. Gilligan and R. Callon, "IPv6 Transition Mechanisms Overview," *Connexions*, Oct. 1995.
[3] R. Hinden, "IP Next Generation Overview," *Connexions*, Mar. 1995.
[4] S. Bradner and A. Mankin, *IPng: Internet Protocol Next Generation*, Reading, MA: Addison-Wesley, 1996.
[5] C. Huitema, *IPv6: The New Internet Protocol,* Upper Saddle River, NJ: Prentice Hall, 1996.
[6] E. Britton, J. Tavs, and R. Bournas, "TCP/IP: The Next Generation," *IBM Sys. J.*, no. 3, 1995.

BIOGRAPHY

WILLIAM STALLINGS is a consultant and lecturer, and the author of over a dozen professional reference books and textbooks on data communications and computer networking. He has designed and implemented both TCP/IP and OSI-based protocol suites on a variety of computers and operating systems, ranging from microcomputers to mainframes. He has a Ph.D. from Massachusetts Institute of Technology in computer science. His home in cyberspace is http://www.shore.net/~ws/welcome.html, and he can be reached at ws@shore.net.