CHAPTER **3**

IPv6 Headers

This third chapter provides a more detailed overview of
IPv6 packet headers. In the first part, the basic IPv6
header will be described, and in the following sections, ex-
tension headers will be presented. (Extension headers are
additional headers that can be present in the IPv6
packet.) Some of these optional headers will be described
in more detail in the following chapters, and the promi-
nent problems related to IPv6 addresses will be discussed
in Chapter 4.
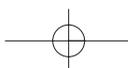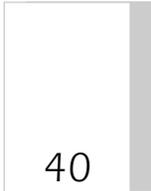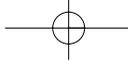
3.1   The IPv6 Header

The IPv6 header was introduced in Chapter 1, but it is shown again in Figure 3-1 for convenience.

We can begin to understand IPv6 better by inspecting its header's fields.

3.1.1   Version

The 4-bit *Version* field contains the number 6. This field is the same size as the IPv4 version field that contains the number 4. Nevertheless, the use of this field is limited because IPv4 and IPv6 packets are not distinguished on the basis of the value contained in it, but as a function of a different protocol type present in the layer 2 envelope (for example, Ethernet or PPP). See, for example, Section 2.9, which describes the encapsulation of IPv6 into LANs and differences with the analogous IPv4 encapsulation.

3.1.2   Priority

The 4-bit *Priority* field in the IPv6 header can assume 16 different values. It enables the source node to differentiate packets it generates by associating different delivery priorities to them. These 16 possible values are further divided into two groups: from 0 through 7 and from 8 through 15.

Values 0 through 7 are used to specify the priority of traffic for which the source is providing traffic control. A typical example is the traffic of

Figure 3-1
The IPv6 header
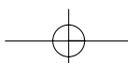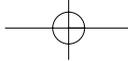
Version	Priorit.	Flow Label		
Payload Length			Next Header	Hop Limit
Source Address				
Destination Address				

applications that use TCP and its congestion control mechanisms based on variable sizes of windows.

RFC 1883[1] proposes the association between priorities and applications shown in Table 3-1.

Values 8 through 15 are used to specify the priority of traffic that does not back off in response to congestion. A typical example is represented by real-time packets like those associated with the transmission of films or sound. Priority 8 is associated with those packets that the network will discard first under conditions of congestion (for example, high-fidelity video traffic), and priority 15 is associated with those packets that the sender will discard at the end, only if absolutely necessary (for example, low-quality telephone audio traffic).

3.1.3   Flow Label

The 24-bit *Flow Label* field in the IPv6 header can be used by a source to label a set of packets belonging to the same flow. A flow is uniquely identified by the combination of the source address and of a nonzero Flow Label. Multiple active flows may exist from a source to a destination (with the same source address but with nonzero and different Flow Labels) as well as traffic that is not associated with any flow (carrying a Flow Label of zero).

In Section 1.3.8, we learned that a flow is a sequence of packets in some way correlated (for example, generated by the same application) and sharing parameters such as the source and destination address, the QoS, the

Table 3-1

Associations between priorities and applications

Priorities	Applications
0	Uncharacterized traffic
1	"Filler" traffic (for example, netnews)
2	Unattended data transfer (for example, e-mail)
3	Reserved for future purposes
4	Attended bulk transfer (for example, FTP, NFS)
5	Reserved for future purposes
6	Interactive traffic (for example, Telnet, X-Windows)
7	Internet control traffic (for example, routing protocols, SNMP)
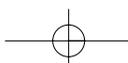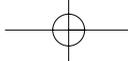
accounting, authorizations, the authentication, and the security. Flows can be unicast (from a node toward another node) or multicast (from a node toward a set of nodes).

Packets belonging to the same flow must be coherently handled by IPv6 routers. The way to handle packets belonging to a given flow can be specified by information within the packets themselves or conveyed by a control protocol such as RSVP (Resource reSerVation Protocol)[2].

RFC 1883[1] specifies that problems related to flows, at the time the RFC itself was published, are still experimental and subject to change when requirements for the Internet flow handling will become clearer. In the meanwhile, nodes that cannot support the function of the Flow Label field are required to set the field to zero when originating a packet, pass the field on unchanged when forwarding a packet, and ignore the field when receiving a packet.

The Flow Label assigned to a flow is a numeric value randomly chosen by the source node from the range 1 to FFFFFF (hexadecimal). This value must be different from Flow Labels in use on the source node or used in the recent past. All packets belonging to the same flow must be sent with the same source address, destination address, priority, and Flow Label. Moreover, if any Hop-by-Hop or Routing extension headers are present (see Section 3.2), they must be the same in all packets belonging to the same flow.

When routers receive the first packet of a new flow, they can process the information carried by the IPv6 header and by Hop-by-Hop and Routing extension headers, "remember" the result (for example, on which interface packets must be retransmitted) in a cache memory, and then apply the result to all other packets belonging to the same flow (with the same source address and the same Flow Label), by reading it directly from the cache memory.

RFC 1883[1] specifies that the cache memory lifetime is limited to 6 seconds, independent from the presence of traffic. For example, let's suppose that a router has in its cache a rule for the flow identified by the source address A and by the Flow Label 37. After 6 seconds, the rule expires; the first packet reaching the router with source address A and Flow Label 37 will be entirely processed and will reestablish the rule, in most cases equal to the previous one.

3.1.4 Payload Length

The 16-bit *Payload Length* field contains the payload length—that is, the length of the data field following the IPv6 header, in octets. Because it is
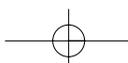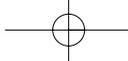
a 16-bit field, the maximum length of an IPv6 packet payload is 64 Kbytes. If a wider data field is needed, a Jumbo Payload extension header can be used (see Section 3.2.4). The presence of a Jumbo Payload is indicated by the value zero in the Payload Length field.

3.1.5 Next Header

The 8-bit *Next Header* field identifies the type of header immediately following the IPv6 header and located at the beginning of the data field (payload) of the IPv6 packet.

   The two most common kinds of Next Headers are clearly TCP (6) and UDP (17), but many other headers are possible. The format adopted for this field is the one proposed for IPv4 by RFC 1700[3]; it is summarized in Table 3-2 where appropriate integration for IPv6 has been inserted. The Next Header field is generally the same as the IPv4 Protocol field.

3.1.6 Hop Limit

The 8-bit *Hop Limit* field is decremented by one by each node (typically a router) that forwards a packet. If the Hop Limit field is decremented to zero, the packet is discarded. The main function of this field is to identify and to discard packets that are looping because of erroneous routing information. Clearly, between two IPv6 nodes, we cannot have more than 255 hops (links), which means no more than 254 routers.

3.1.7 Source Address

The 128-bit *Source Address* field contains the IPv6 address of the node originating the packet. The IPv6 address format, which is specified by RFC 1884[4], will be discussed in Chapter 4 of this book.

3.1.8 Destination Address

The 128-bit *Destination Address* field contains the IPv6 address of the node recipient of the packet. If a Routing header is present, this address is not that of the ultimate recipient (see Section 3.3.5).
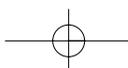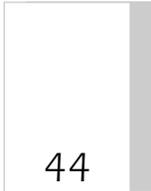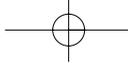
Table 3-2

Possible values for the Next Header field

Decimal Value	Keyword	Protocol
0		Reserved (IPv4)
0	HBH	Hop-by-Hop option (IPv6)
1	ICMP	Internet Control Message (IPv4)
2	IGMP	Internet Group Management (IPv4)
3	GGP	Gateway-to-Gateway Protocol
4	IP	IP in IP (IPv4 encapsulation)
5	ST	Stream
6	TCP	Transmission Control
8	EGP	Exterior Gateway Protocol
9	IGP	Any private interior gateway
16	CHAOS	Chaos
17	UDP	User Datagram
29	ISO-TP4	ISO Transport Protocol Class 4
36	XTP	XTP
43	RH	Routing header (IPv6)
44	FH	Fragmentation header (IPv6)
45	IDRP	Inter-Domain Routing Protocol
46	RSVP	Reservation Protocol
50	ESP	Encapsulating Security Payload
51	AH	Authentication header (IPv6)
54	NHRP	NBMA Next Hop Resolution Protocol
58	ICMP	Internet Control Message (IPv6)
59	Null	No next header (IPv6)
60	DOH	Destination Options header (IPv6)
80	ISO-IP	ISO Internet Protocol (CLNP)
83	VINES	VINES
88	IGRP	IGRP
89	OSPF	OSPF (Open Shortest Path First)
93	AX.25	AX.25 Frames
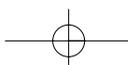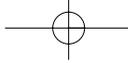
3.1.9 Examples of IPv6 Packets

Appendix B shows some examples of IPv6 packets captured with a Rad-com RC100 protocol analyzer. In particular, Section B.3 shows a TCP packet, and Section B.4 shows a UDP packet.

3.2 Extension Header

The IPv4 header has space for some optional fields requiring a particular processing of packets. These optional fields are not used often, and they can deteriorate router performance remarkably because their presence must be checked for each packet. IPv6 replaces optional fields with *extension headers*. Extension headers are based on the principle that most of the packets need a very simple processing, and therefore basic fields of the IPv6 header are sufficient (see Figure 3-1). Packets requiring additional information at the network layer can encode this information in additional headers that can be placed in a header between the IPv6 header and the upper layer header. Headers are connected by the Next Header field (see Section 3.2.5) to form a chain similar to the one shown in Figure 3-2.

An IPv6 packet can have no extension headers, one extension header, or several extension headers. The few existing extension headers will be described later in this chapter.

Each extension header has a length equal to a multiple of 8 octets (64 bits).

A full implementation of IPv6 must include support for the following extension headers:

■ Hop-by-Hop Options
■ Routing (Type 0)

Figure 3-2
Extension headers
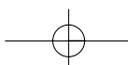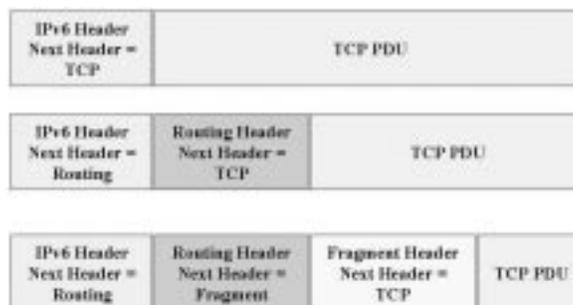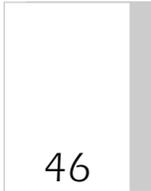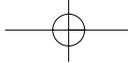
- Fragment
- Destination Options
- Authentication
- Encapsulating Security Payload

3.2.1 Extension Headers Order

Extension headers must be processed in the order they appear in the packet. Most extension headers will be processed only by the destination node (for example, those related to security); therefore, they do not affect the router's performance.

The only type of extension header to be processed by all nodes along its delivery path is the Hop-by-Hop Options header (see Section 3.2.3), which, if present, must immediately follow the IPv6 header. Its presence is indicated by the value zero in the Next Header field of the IPv6 header (see Table 3-2).

When more than one extension header is used in the same packet, RFC 1883[1] recommends that those headers appear in the following order:

- IPv6 header
- Hop-by-Hop Options header
- Destination Options header (see note 1)
- Routing header
- Fragment header
- Authentication header
- Encapsulating Security Payload header
- Destination Options header (see note 2)
- Upper layer header

Each extension header should occur at most once, with the only exception being the Destination Options header, which can occur twice in different positions (see notes 1 and 2).

NOTE 1: *For options to be processed by all nodes whose address appears in the IPv6 Destination Address field and in the Routing header.*
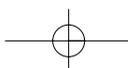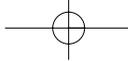
NOTE 2: *For options that must be processed only by the final destination of the packet.*

3.2.2 Options

Among extension headers previously listed, the Hop-by-Hop Options header and the Destination Options header carry variable numbers of options. These options, which are encoded in a format called *TLV (Type-Length-Value),* are shown in Figure 3-3.

The three fields that appear in the described option have the following meanings:

- *Option Type* is an 8-bit unsigned integer; it identifies the type of option.
- *Option Data Length* is an 8-bit unsigned integer that contains the length of the Option Data field in octets.
- *Option Data* is a variable length field that contains the option-type specific data.

The Option Type field assigns a particular meaning to the three highest order bits, as shown in Figure 3-4.

In particular, the two highest order bits (*Action*) specify the action that must be taken if the processing IPv6 node doesn't recognize the option type. These bits can have the following values:

00   Skip over this option and continue to process eventual subsequent options.

01   Discard the packet.

10   Discard the packet, regardless of whether the packet destination address is multicast; the source node is notified by an ICMP packet.

Figure 3-3
TLV Options (Type-Length-Value)

Figure 3-4
Detail of the Option Type field

48
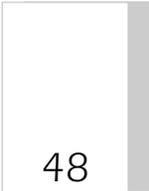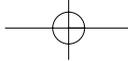
Chapter Three

11   Discard the packet, and only if its destination address is not
        multicast, the source node is notified by an ICMP packet.

The third bit *C* (Change) specifies whether the Option data can change
en route to the packet's final destination:

0   Option data cannot change en route.
1   Option data can change en route.

Two *padding* options are used, when necessary, to align subsequent op-
tions. They are shown in Figure 3-5.

The (a) option, which is called *Pad1,* is used to insert one octet of padding
into the Options area of a header, and the (b) option, which is called *PadN,* is
used to insert two or more octets of padding in the Options area of a header.

3.2.3   Hop-by-Hop Options Header

The Hop-by-Hop Options header is used to carry optional information
that must be examined by every node along a packet delivery path. This
type of header must immediately follow the IPv6 header, and its presence
is identified by a value zero in the Next Header field of the IPv6 header.
Its format is shown in Figure 3-6.

The 8-bit *Next Header* field has the same meaning as the field with the
same name in the IPv6 header (see Section 3.1.5).

The 8-bit *Hdr Ext Len* (Header Extension Length) field contains the
length of the Hop-by-Hop Options header in 8-octet units (64 bits), not in-

Figure 3-5
Padding options
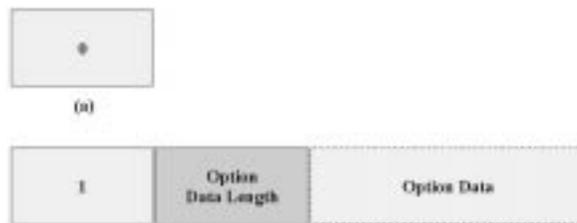
Figure 3-6
Hop-by-Hop Options
header
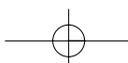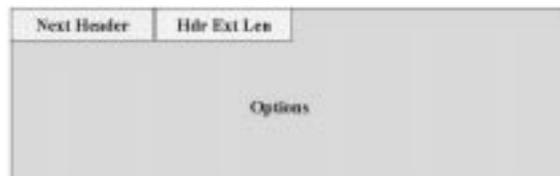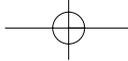
cluding the first 8 octets. Expressing the length this way can appear extravagant, but it speeds up IPv6 implementations.

In fact, remember that extension headers have lengths in multiples of 8 octets (64 bits), and they obviously cannot be empty; therefore, they are almost always 64 bits long. (This condition is indicated by the value zero in the Hdr Ext Len field.) For example, if the Hop-by-Hop Options header is 256 octets long, the Hdr Ext Len value is 3. This coding simplifies the implementation because it avoids continued testing on the Hdr Ext Len value's validity. If the header length were measured in octets, the values 0, 1, 2, 3, 4, 5, 6, 7, 9, and so on would not be valid!

The *Options* field has a variable length, and it contains one or more TLV options (see Section 3.2.2).

In addition to the Pad1 and PadN options, we will define the Jumbo Payload option.

3.2.4   The Jumbo Payload Option

As we've already seen in Section 3.1.4, the choice to have a 16-bit Payload Length field limits the IPv6 payload length to 65,535 octets. The Jumbo Payload option can be used to exceed this limit.

This option requires a $4n + 2$ alignment, where n can be any natural number. This means that the option can begin with octets 2, 6, 10, 14, and so on. The format of the Jumbo Payload option is shown in Figure 3-7.

The 8-bit *Option Type* field contains the value 194, which indicates the Jumbo Payload option. The 8-bit *Option Data Length* field contains the value 4, which indicates that 4 octets of data will follow—that is, the *Jumbo Payload Length* field. The last one indicates the packet length in octets, excluding the IPv6 header but including the Hop-by-Hop Options header. This length must be more than 65,535 octets.

The Payload Length field of the IPv6 header must be set to zero in every packet that carries the Jumbo Payload option. The Jumbo Payload option is not consistent with the Fragment header (see Section 3.3.6); therefore, they cannot both be present in the same IPv6 packet.

Figure 3-7
The Jumbo Payload option
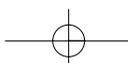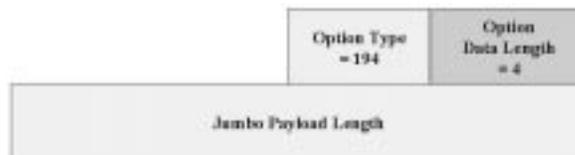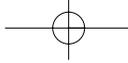
If an IPv6 implementation doesn't support the Jumbo Payload option, it cannot have an interface to a link whose link MTU is greater than 65,575 octets (40 octets of the IPv6 header plus 65,535 octets of Payload).

3.2.5 Routing Header

The Routing header is used by an IPv6 source to specify a list of intermediate nodes that a packet has to traverse on the path to its destination. The specification can be for each node on the path binding (*Strict*) or not (*Loose*). This header supports a function very similar to the IPv4 packet Source Route option.

The Routing header is identified by a Next Header value of 43 (see Table 3-2). The Routing header can be of different types, but currently RFC 1883[1] specifies only the type zero, whose organization is shown in Figure 3-8.

The 8-bit *Next Header* field uses the same values as the field with the same name in the IPv6 header (see Section 3.1.5).

The 8-bit *Hdr Ext Len* (Header Extension Length) field contains the length of the Routing header in 8-octet (64-bit) units, not including the first 8 octets. In the case of a Type 0 Routing header, the Hdr Ext Len value must be less than or equal to 46, equal to twice the number of addresses in the header itself, and therefore even. In fact, the first 64 bits contain the fixed part of the Type 0 Routing header (Next Header, Hdr Ext Len, Routing Type, Segment Left, Reserved, and Strict/Loose Bit Map), and each address has 128 bits—that is, two times 64 bits.

Figure 3-8
Type 0 Routing header
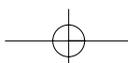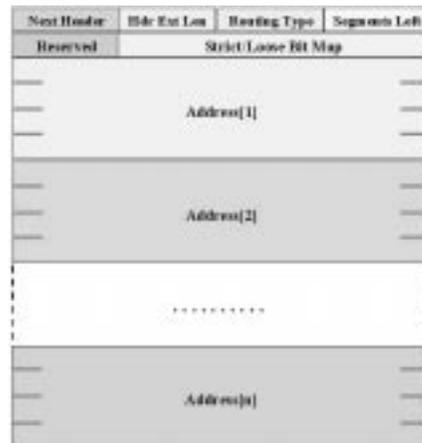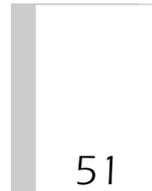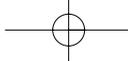
The 8-bit *Routing Type* field always contains, in this case, the zero value. Different values can be used in the future to support new types of Routing headers.

The 8-bit *Segments Left* field contains the number of explicitly listed intermediate nodes still to be visited on the path to the destination—that is, the number of addresses not yet used. The maximum legal value for this field is 23.

The 8-bit *Reserved* field is reserved for future uses. It must be set to zero for transmission and ignored on reception.

The 24-bit *Strict/Loose Bit Map* field is a mask containing a Strict/Loose bit for each address. If the *Strict/Loose* bit associated with an address is zero, then the address must be treated as *Loose;* if equal to 1, the address must be treated as *Strict*.

Each *Address* field is 128 bits long, and up to 23 Address fields can be used. They contain IPv6 addresses of nodes to be traversed along the path to the destination. Nodes are visited in the order Address[1] . . . Address[*n*].

When a Routing header is processed by a node, the node checks whether the Segment Left field is different from 0, and if so, it extracts the following address and the Strict/Loose bit associated with the address. If the bit indicates that the address must be treated in the Strict way, the node checks that the address belongs to an adjacent node (a neighbor on one of the links), and it delivers the packet on the interface associated with that adjacent node; if the node is not adjacent, the packet is discarded. If the bit indicates that the address must be treated in the Loose way, the node examines its routing tables and routes the packet to the address.

The complete flowchart of the Routing header management procedure can be found in Section A.1 of Appendix A, and an example of routing is shown in Section A.2.

3.2.6   Fragment Header

The management of fragmentation in IPv4 is rather different from the management in IPv6. In fact, one of the IPv4 router's tasks is to fragment a packet if its size is too large to fit the MTU of the link on which it must retransmit the packet. For example, if an IPv4 router receives a 4000-octet packet from an FDDI ring and must retransmit it on an Ethernet network with an MTU link of 1500 octets, the router fragments the packet into three packets with sizes less than or equal to 1500 octets. This IPv4 function can be deactivated by setting the *don't fragment* bit to 1 in the IPv4 header.
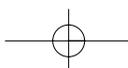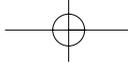
Figure 3-9
Fragment header

Next Header	Reserved	Fragment Offset	Res	M
Identification				
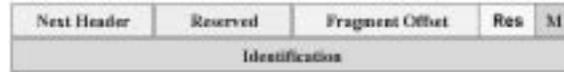
IPv6 routers always operate as if the don't fragment bit were equal to 1. When they receive a packet whose size exceeds the MTU link, they discard the packet and signal this fact to the source (through an ICMP packet), by indicating what is the maximum length that can be accepted for the packet. The source node will appropriately fragment the packet before retransmitting it.

The fragmentation can be implemented in IPv6 only by the source node through an extension header and in particular by a Fragment header. The Fragment header is processed only by the destination node and ignored by nodes along the path. The Fragment header (see Figure 3-9) is identified by the value 44 in the Next Header field.

The 8-bit *Next Header* field uses the same values as the field with the same name in the IPv4 header (see Section 3.1.5).

The 8-bit *Reserved* field is reserved for future uses. It is initialized to zero for transmission and is ignored on reception.

The 13-bit *Fragment Offset* field contains the offset of the data following this header relative to the start of the fragmentable part of the original packet (before fragmentation), in 8-octet (64-bit) units.

The 2-bit *Res* field is reserved for future uses. It must be set to zero for transmission and ignored on reception.

The 1-bit *M* (More fragments) field indicates whether a fragment is the last in a packet (M = 0) or not (M = 1).

The 32-bit *Identification* field contains a unique identification of the packet generated by the node that executes the fragmentation. Its aim is to simply identify all fragments belonging to the same packet.

3.2.7 The Fragmentation Process

To better understand the use of the Fragment header, let's see how an IPv6 host manages a fragmentation process. The first point to understand is that an IPv6 packet consists of a fragmentable part and an unfragmentable part (see Figure 3-10).

The unfragmentable part consists of the IPv6 header plus other headers that must be processed by all nodes along the path—that is, extension headers up to and including the Routing header.
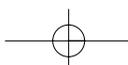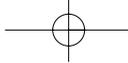
Figure 3-10
*An IPv6 header be-
fore fragmentation*

Unfragmentable part	Fragmentable part

Figure 3-11
*Packets resulting
from a fragmentation*

Unfragmentable part	Fragment Header	Fragment 1

Unfragmentable part	Fragment Header	Fragment 2

....

Unfragmentable part	Fragment Header	Last fragment

The fragmentable part consists of the rest of the packet—that is, ex-
tension headers that must be processed only by the destination node and
the IPv6 payload. This part is divided into fragments that are multiples
of 8 octets (with the possible exception of the last one). A Fragment header
is put before each fragment, and fragments are transmitted as separate
IPv6 packets, as illustrated in Figure 3-11.

The unfragmentable part, which is repeated in each packet, is equal to
the original unfragmentable part with two exceptions:

■ The Payload Length field of the IPv6 header reflects the new
length.
■ The Next Header field of the last header of the unfragmentable
part is set to 44 to indicate the Fragment header presence.

3.2.8 Destination Options Header

The Destination Options Header is used to carry additional information that
must be processed only by the destination node or nodes, not by each node
on the routing path of the packet. This kind of header is identified by a Next
Header field value of 60 (see Table 3-2). It has been decided to create only
one type of header for all destination options because the Next Header field
is limited to 8 bits, and therefore only 256 values are totally available. The
Destination Options header structure is illustrated in Figure 3-12.
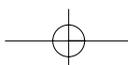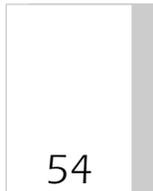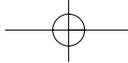
Figure 3-12
*Destination Options
header*

The 8-bit *Next Header* field uses the same values as the field with the
same name in the IPv4 header (see Section 3.1.5).

The 8-bit *Hdr Ext Len* (Header Extension Length) field contains the
length of the Destination Options header in 8-octet (64-bit) units, not in-
cluding the first 8 octets.

The variable-length *Options* field contains one or more TLV-encoded op-
tions (see Section 3.1.2).

The only options specified by RFC 1883[1] that can be part of a Desti-
nation Options header are the Pad1 and the PadN (see Figure 3-5). RFC
1888[6] specifies an option called NSAP (see Section 4.7.9) that can be part
of a Destination Options header.

3.2.9   No Next Header

The value 59 in the Next Header field of the IPv6 header or any exten-
sion header indicates that nothing follows that header. If the Payload
Length field of the IPv6 header indicates the presence of other octets af-
ter the header, they must be ignored and retransmitted unchanged.

3.2.10   Security Header

Two extension headers, like the Destination Options header, are used to
carry additional information that must be processed only by the destina-
tion node or nodes, not by each node on the routing path of the packet.
These two headers, which are dedicated to security problems, are called the
Authentication header and the Encapsulating Security Payload header.

The Encapsulating Security Payload header is indicated by the value
50 in the Next Header field (see Table 3-2).

These headers will be described in Chapter 8, which is dedicated to se-
curity problems.
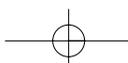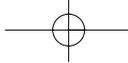
3.3   Size of IPv6 Packets

RFC 1883[1] explicitly recommends that, to allow IPv6 to operate correctly, each link must have an MTU greater than or equal to 576 octets. If any link cannot meet this constraint, it must provide fragmentation mechanisms at the link layer, typically at layer 2 of the ISO/OSI reference model.

Links that have a configurable MTU must be configured to have an MTU greater than or equal to 576 octets.

It is recommended that IPv6 nodes implement Path MTU discovery[5] to use MTUs greater than 576 octets. However, very simple nodes, with restricted IPv6 implementations, can simply send packets not longer than 576 octets and be certain that, fitting the MTU path, they will be delivered.

REFERENCES

[1] S. Deering, R. Hinden, *RFC 1883: Internet Protocol, Version 6 (IPv6) Specification*, December 1995.

[2] B. Braden, L. Zhang, D. Estrin, S. Herzog, S. Jamin, *RSVP: Resource ReSerVation Protocol (RSVP)—Version 1 Functional Specification*, Work in progress, January 1996.

[3] J. Reynolds, J. Postel, *RFC 1700: Assigned Numbers*, October 1994.

[4] R. Hinden, S. Deering, *RFC 1884: IP Version 6 Addressing Architecture*, December 1995.

[5] J.C. Mogul, S.E. Deering, *RFC 1191: Path MTU discovery*, November 1990.

[6] J. Bound, B. Carpenter, D. Harrington, J. Houldsworth, A. Lloyd, *RFC 1888: OSI NSAPs and IPv6*, August 1996.