Options Headers (Hop-by-Hop Options and Destination Options)

Bit Number

```
                    1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 3 3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
```

Next Header	Hdr Ext Len		4
Options			8

Next Header
8-bit identifier for the header immediately following this one. Uses the same codes as the main IPv6 header.

Hdr Ext Len
8-bit length of the Hop-by-Hop Options header in 8-octet units not including the first 8 octets, i.e. (length ***in*** octets-8)/8.

Options
Variable-length field, containing the options.
NOTE: length ***must*** be a multiple of 8 octets long.

Option Encoding:

```
                    1 1 1 1 1 1
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 ...
```

Option Type	Opt Data Len	Option Data
W W C T T T T T		

Option Type	8-bit Identifier
WW	indicate what to do if this option is not recognized:
00	skip this option and continue processing the header.
01	discard packet.
10	discard packet and send an ICMP Parameter Problem code 2 back to the source address pointing to the unrecognized Option Type.
11	discard packet and, if destination is not a multicast address, behave like type 10.
C	indicates whether the option data for this option can change en-route to the destination. Relevant if, in particular, an AH is present.
0	no change
1	can change
TTTTT	rest of the option type code

Opt Data Len
8-bit length of the Option Data field of this option, in octets.

Option Data
Variable-length field.

Options which must be implemented:

i) Pad1 option, special case:

```
0 1 2 3 4 5 6 7
```

0

NOTE: no length or field values!

ii) PadN option:

```
                    1 1 1 1 1 1
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5
```

1	Opt Data Len	Option Data

Routing Header (similar to IPv4 LSRR and RR options)

Bit Number

```
                    1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 3 3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
```

Next Header	Hdr Ext Len	Routing Type	Segments Left	4
type-specific data				

Next Header
8-bit identifier for the header immediately following this one. Uses the same codes as the main IPv6 header.

Hdr Ext Len
8-bit length of the Hop-by-Hop Options header in 8-octet units not including the first 8 octets, i.e. (length ***in*** octets-8)/8.

Routing Type
8-bit identifier

Segments Left
8-bit integer giving the number of listed intermediate nodes which still need to be visited.

type-specific data
Variable-length field which depends on the routing type. Must be a multiple of 8 octets.

Only one routing header type has been defined, type 0:

Type 0:

```
                    1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 3 3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
```

Next Header	Hdr Ext Len	Routing Type = 0	Segments Left	4
Reserved (MBZ)				8
				12
Address[1]				16
				20
				24
				28
Address[2]				32
				36
				40
Address[n]				

INTERNET STORM CENTER

IPv6 TCP/IP and tcpdump

POCKET REFERENCE GUIDE

SANS INSTITUTE

incidents@sans.org • +1 317.580.9756 • http://www.sans.org • http://www.incidents.org

tcpdump Usage

```
tcpdump  [-aenStvx]  [-F file]
[-i int]  [-r file]  [-s snaplen]
[-w file]  ['filter_expression']

-e  Display data link header.
-F  Filter expression in file.
-i  Listen on int interface.
-n  Don't resolve IP addresses.
-r  Read packets from file.
-s  Get snaplen bytes from each packet.
-S  Use absolute TCP sequence numbers.
-t  Don't print timestamp.
-v  Verbose mode.
-w  Write packets to file.
-x  Display in hex.
-X  Display in hex and ASCII.
```

Acronyms

AH	Authentication Header (RFC 2402)	ISAKMP	Internet Security Association & Key Management Protocol (RFC 2408)
ARP	Address Resolution Protocol (RFC 826)		
BGP	Border Gateway Protocol (RFC 1771)	L2TP	Layer 2 Tunneling Protocol (RFC 2661)
CWR	Congestion Window Reduced (RFC 2481)	NNTP	Network News Transfer Protocol (RFC 977)
DF	Don't Fragment bit (IP)	OSPF	Open Shortest Path First (RFC 1583)
DHCP	Dynamic Host Configuration Protocol (RFC 2131)	POP3	Post Office Protocol v3 (RFC 1460)
DNS	Domain Name System (RFC 1035)	RFC	Request for Comments
ECN	Explicit Congestion Notification (RFC 3168)	RIP	Routing Information Protocol (RFC 2453)
EIGRP	Extended IGRP (Cisco)	LDAP	Lightweight Directory Access Protocol (RFC 2251)
ESP	Encapsulating Security Payload (RFC 2406)	SKIP	Simple Key-Management for Internet Protocols
FTP	File Transfer Protocol (RFC 959)	SMTP	Simple Mail Transfer Protocol (RFC 821)
GRE	Generic Routing Encapsulation (RFC 2784)	SNMP	Simple Network Management Protocol (RFC 1157)
HTTP	Hypertext Transfer Protocol (RFC 1945)	SSH	Secure Shell
ICMP	Internet Control Message Protocol (RFC 792)	SSL	Secure Sockets Layer (Netscape)
IGMP	Internet Group Management Protocol (RFC 2236)	TCP	Transmission Control Protocol (RFC 793)
IGRP	Interior Gateway Routing Protocol (Cisco)	TFTP	Trivial File Transfer Protocol (RFC 1350)
IMAP	Internet Message Access Protocol (RFC 2060)	TOS	Type of Service field (IP)
IP	Internet Protocol (RFC 791)	UDP	User Datagram Protocol (RFC 768)

All RFCs can be found at http://www.rfc-editor.org

DNS

Bit Number

```
                              1   1   1   1   1   1
0   1   2   3   4   5   6   7   8   9   0   1   2   3   4   5
```

ID.							
QR	Opcode	AA	TC	RD	RA	Z	RCODE
QDCOUNT							
ANCOUNT							
NSCOUNT							
ARCOUNT							
Question Section							
Answer Section							
Authority Section							
Additional Information Section							

Query/Response
0 Query
1 Response

Opcode
0 Standard query (QUERY)
1 Inverse query (IQUERY)
2 Server status request (STATUS)

AA
(1 = Authoritative Answer)

TC
(1 = TrunCation)

RD
(1 = Recursion Desired)

RA
(1 = Recursion Available)

Z
(Reserved; set to 0)

Response code
0 No error
1 Format error
2 Server failure
3 Non-existant domain (NXDOMAIN)
4 Query type not implemented
5 Query refused

QDCOUNT
(No. of entries in Question section)

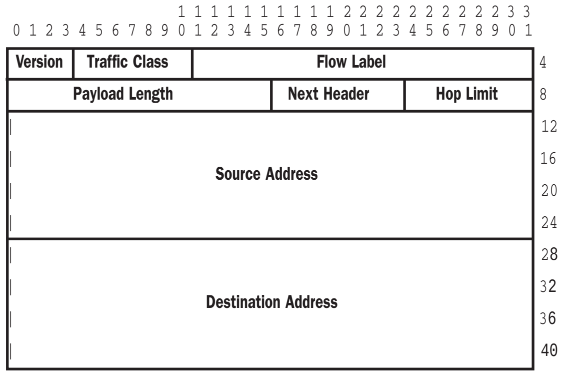
ANCOUNT
(No. of resource records in Answer section)

NSCOUNT
(No. of name server resource records in Authority section)

ARCOUNT
(No. of resource records in Additional Information section.

IPv6 Header

Bit Number



Version

4-bit Internet Protocol version number = 6.

Traffic Class

8-bit traffic class field (Experimental)
Default = 0
To be used for QoS and traffic prioritisation

Flow Label

20-bit flow label (Experimental)
Default = 0
Used in association with "traffic class" to
label packets for QoS.

Payload Length

16-bit integer.
Payload length in octets (packet - header)
NOTE: extension headers are considered part of the payload!

Next Header

8-bit "selector". Identifies the type of header immediately following the IPv6 header.

Some examples:

0	Hop-by-Hop Options (NOTE: special processing)
43	Routing (Type 0)
44	Fragment
50	Encapsulating Security Payload
51	Authentication
58	ICMPv6
59	No next header
60	Destination Options

Standard headers inherited from IPv4:

6	TCP
17	UDP

Hop Limit

8-bit unsigned integer. Decremented by 1 by each node that forwards the packet. The packet is discarded if Hop Limit is decremented to zero.

Source Address

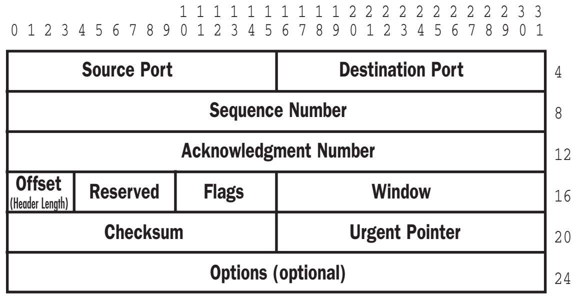
128-bit source address

Destination Address

128-bit destination address
NOTE: not necessarily the final destination if a Routing header is present!

TCP Header

Bit Number



Common TCP Well-Known Server Ports

7 echo	110 pop3
19 chargen	111 sunrpc
20 ftp-data	119 nntp
21 ftp-control	139 netbios-ssn
22 ssh	143 imap
23 telnet	179 bgp
25 smtp	389 ldap
53 domain	443 https (ssl)
79 finger	445 microsoft-ds
80 http	1080 socks

Offset

Number of 32-bit words in TCP header; minimum value = 5

Reserved

4 bits; set to 0

ECN bits (used when ECN employed; else 00)

CWR (1 = sender has cut congestion window in half)

ECN-Echo (1 = receiver cuts congestion window in half)

Flags (UAPRSF)

U (1 = Urgent pointer valid)
A (1 = Acknowledgement field value valid)
P (1 = Push data)
R (1 = Reset connection)
S (1 = Synchronize sequence numbers)
F (1 = no more data; Finish connection)

Checksum

Covers pseudoheader and entire TCP segment

Urgent Pointer

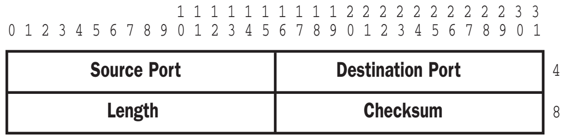
Points to the sequence number of the byte
following urgent data.

Options

0 End of Options list	3 Window scale
1 No operation (pad)	4 Selective ACK ok
2 Maximum segment size	8 Timestamp

UDP Header

Bit Number



Common UDP Well-Known Server Ports

7	echo	138	netbios-dgm
19	chargen	161	snmp
37	time	162	snmp-trap
53	domain	500	isakmp
67	bootps (DHCP)	514	syslog
68	bootpc (DHCP)	520	rip
69	tftp	33434	traceroute
137	netbios-ns		

Length

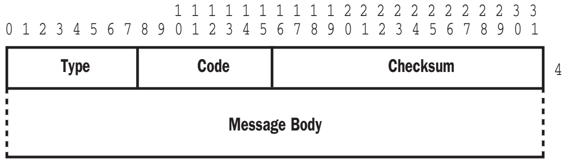
(Number of bytes in entire datagram including header;
minimum value = 8)

Checksum

(Covers pseudo-header and entire UDP datagram)

ICMPv6 (header type 58)

Bit Number

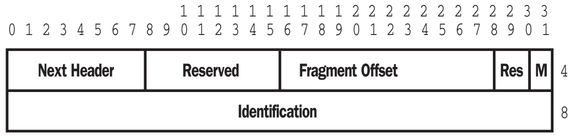


Type	Code	
1	0	no route to destination
	1	communication administratively prohibited
	2	(not assigned)
	3	address unreachable
	4	port unreachable
2	0	packet too big message, message body contains MTU of next hop link.
3	0	hop limit exceeded in transit
	1	fragment reassembly time exceeded
4	0	erroneous header field encountered
	1	unrecognized "Next Header" type encountered
	2	unrecognized IPv6 option encountered
128	0	echo request
129	0	echo reply

Fragment Header

Note: fragmentation can only be performed by the source nodes, not routers!

Bit Number



Next Header

8-bit identifier for the header immediately following this one. Uses the same codes as the main IPv6 header.

Reserved

8-bit reserved field. Initialized to zero for transmission; ignored on reception.

Fragment Offset

13-bit unsigned integer. The offset, in 8-octet units, of the data following this header, relative to the start of the data which can be fragmented of the original packet. Note that the IPv6 header and extensions headers which need to be processed at every hop ***cannot*** be fragmented! [This is known as the "Unfragmentable Part" in IPv6 jargon].

Res

2-bit reserved field. Initialized to zero for transmission; ignored on reception.

M flag

1 = more fragments; 0 = last fragment.

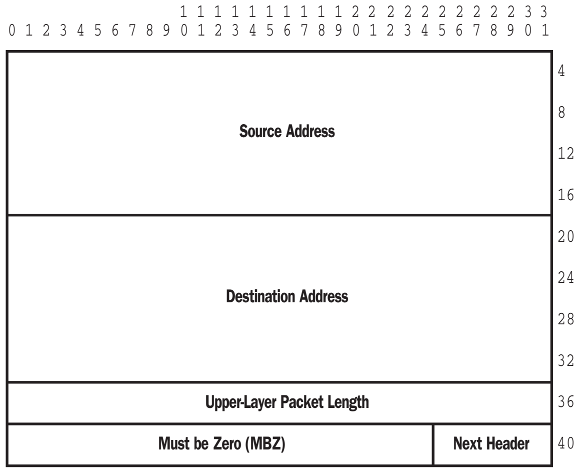
Identification

32 bits identifier for reassembly.

Checksums

*The IPv6 header does **not** include checksums on the assumption that if checksumming is required then it will be done via an AH header which provides cryptographically strong authentication (and hence a checksum) of the whole packet.*
There remains an issue with upper-layer protocols, for exmaple TCP and UDP which include a checksum calculation. In particular the "pseudo-header" to be used in IPv6 TCP/UDP checksum calculations is:

Bit Number



Note: unlike IPv4 the UDP checksum is compulsory when carried over IPv6!