Header Offset Shortcuts

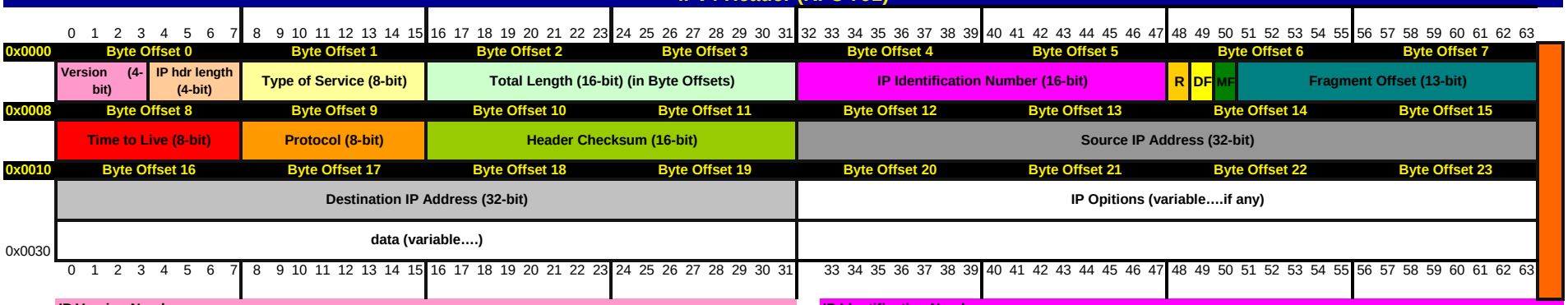
Field	Length (bits)	TCPDUMP Filter	Notes
IP Header Length	4	ip[0] &0x0F	Remember to use a 4 byte multiplier to find header length in bytes
IP Packet Length	16	ip[2:2]	The is no multiple for this length field
IP TTL	8	ip[8]	
IP Protocol	8	ip[9]	

	D	Hex	Proto	D	Hex	Proto	D	Hex	Proto
	1	0x01	ICMP	9	0x09	IGRP	47	0x2F	GRE
	2	0x02	IGMP	17	0x11	UDP	50	0x32	ESP
	6	0x06	TCP	47	0x2F	GRE	51	0x33	AH

IP Address - Src	32	ip[12:4]	
IP Address - Dst	32	ip[16:4]	
IP Fragmentation	flag=3	ip[6] &0x20 = 0x20	More Fragment bit is set.
	offset=13	ip[6:2] &0x1fff != 0x000	fragment offset in not 0
ICMP Type	8	icmp[0]	
ICMP Code	8	icmp[1]	
TCP Src Port	16	tcp[0:2]	
TCP Dst Port	16	tcp[2:2]	
TCP Header Length	4	tcp[12] &0xF0	Remember to use a 4 byte multiplier to find header length in bytes
TCP Flags	8	tcp[13]	
TCP Windows Size	16	tcp[14:2]	
UDP Src Port	16	udp[0:2]	
UDP Dst Port	16	udp[2:2]	
UDP Header Length	16	upd[4:2]	The is no multiplier for this length field

Hdr Offset

IPv4 Header (RFC 791)



IP Version Number
Valid values are: 4 for IP version 4 6 for IP version 6

IP Header Length (4 byte multiplier)
Number of 32-bit words in IP header minimum value 5 (5 x 4 = 20 bytes) maximum value 15 (15 x 4 = 60 bytes)

Type of Service (Used by gateways as a QoS type field) (Most OS's default to 0) (Over)

Total Length (No multiplier)
Number of bytes in packet maximum length = 65,535

Time To Live

IP Protocol

D	Hex		D	Hex		D	Hex		D	Hex	
1	0x01	ICMP	9	0x09	IGRP	47	0x2F	GRE	88	0x58	EIGRP
2	0x02	IGMP	17	0x11	UDP	50	0x32	ESP	89	0x59	OSPF
6	0x06	TCP	47	0x2F	GRE	51	0x33	AH			

Header Checksum
Covers IP header only Validated along the path from source to destination

Type of Service (Used by gateways as a QoS type field) (Most OS's default to 0)

Bit 0 - 2 Precedence
Bit 3 0 = Normal Delay 1 = Low Delay
Bit 4 0 = Normal Throughput 1 = High Throughput
Bit 5 0 = Normal Relibility 1 = High Relibility
Bit 6 & 7 Reserved for future use (Always set to 0)

IP Identification Number
Uniquely identifies every datagram sent by host, value typically incremented by 1 (AKA Fragment ID)

Flags
R is reserved and must be set to 0
D is Don't Fragment Flag 1=Don't Fragment 0=Can Fragment
MF is More Fragments 1=More Fragments 0=No Fragment or no more Fragments

(frag x:y@z where x is the fragment ID, y is # of bytes (must be divisible by 8) and z is the fragment offset)
(In Ethernet the MTU 1500 should see middle fragments of size 1480 (1480 data + 20 ip header = 1500)

Fragment Offset (8 byte multiplier) (Max fragment offest 65528)
Position of this fragment in the original datagram value is multipled by 8 to get bytes

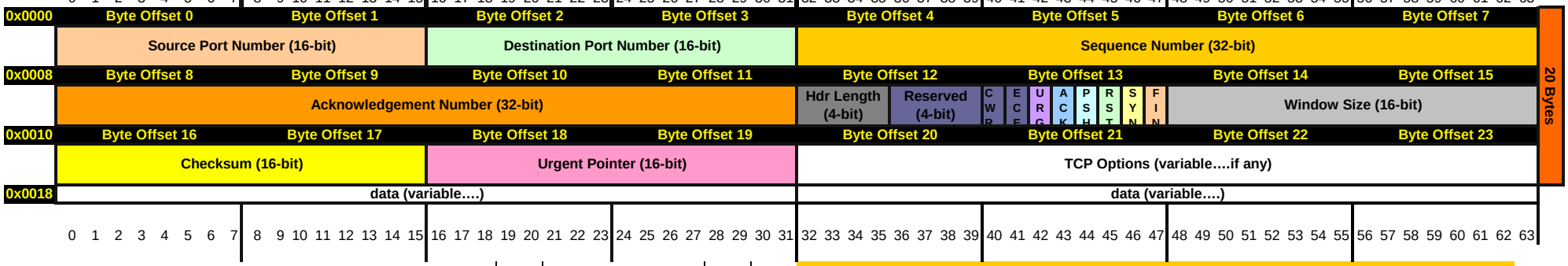
Options (0-40 bytes; 1st @ 20th byte offset; padded 4-byte boundary) (Processed by each router as packet passes)

D	Hex		D	Hex	
0	0x00	End of Option list	68	0x44	Timestamp
1	0x01	No operation (pad)	131	0x83	Loose source route (security risk)
7	0x07	Record Route (security risk)	137	0x89	Strict source route (security risk)

Precedence

1	1	1	Network Control
1	1	0	Internetwork Control
1	0	1	CRITIC / ECP
1	0	0	Flash Override
0	1	1	Flash Override
0	1	0	Immediate
0	0	1	Priority
0	0	0	Routine

TCP Header (RFC 793)



Common Port Numbers

D	Hex		D	Hex		D	Hex		D	Hex	
7	0x07	echo	22	0x16	ssh	80	0x50	http	143	0x8F	imap
19	0x13	chargen	25	0x19	smtp	110	0x6E	pop3	179	0xB3	bgp
20	0x14	ftp-data	53	0x35	domain	119	0x77	nntp	389	0x185	ldap
21	0x15	ftp-control	79	0x4F	finger	139	0x8B	netbios-ssn	445	0x1BD	ms-ds

Acknowledgement Number
Represents next byte of data receiving host expects: (last received sequence number + 1)

Checksum
Covers psedoheader and entire TCP segment

Urgent Pointer
Points to the sequence number of the byte following urgent data

Sequence Number
32-bit number uniquely identifies initial byte of segment data.

Header Length (4 byte multiplier)
Number of 32-bit words in TCP header minimum value 5 (5x4=20bytes) maximum value 15 (5x15=60bytes)

Reserved 4 bits set to 0

Congestion Window Reduced (CWR)
Set to 0 unless ECN is used. (1 = sender has cut congestion window in half)

Explicit Congestion Notification Echo (ECE)
Set to 0 unless ECN is used. (1 = receiver cuts congestion window in half)

Flags
URG = Urgent ACK = Acknowledgment PSH = Push RST = Reset SYN = Syncronize
FIN = Finish

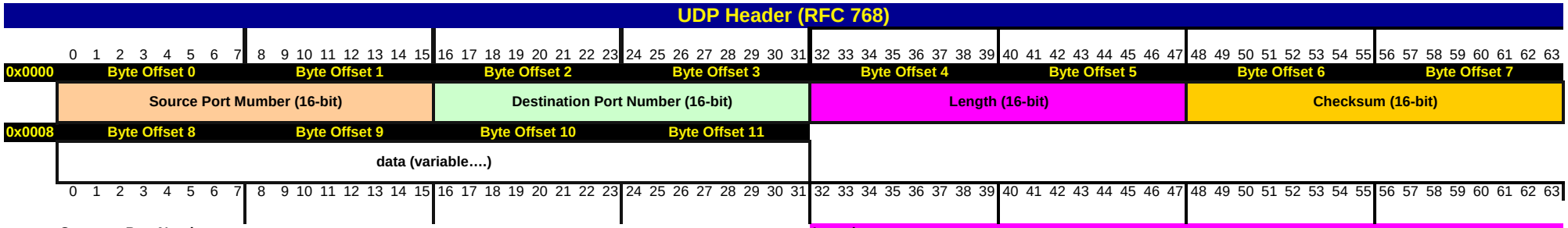
Window Size
Acts as flow control. Window size dynamically changes as data is received. A 0 window size tells src host to wait.

Options

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

TCP Hdr

UDP Header (RFC 768)



Common Port Numbers

D	Hex		D	Hex		D	Hex	
7	0x07	echo	69	0x45	tftp	514	0x202	syslog
19	0x13	chargen	137	0x89	netbios-ns	520	0x208	rip
37	0x25	time	138	0x8A	netbios-dgm	33434	829A	traceroute
53	0x35	domain	161	0xA1	snmp			
67	0x43	bootps	162	0xA2	snmp-trap			
68	0x44	bootpc	500	0x1F4	isakmp			

Length

Number of bytes in the entire datagram including header

minimum value 8

(Which is the length of just the header with no data)

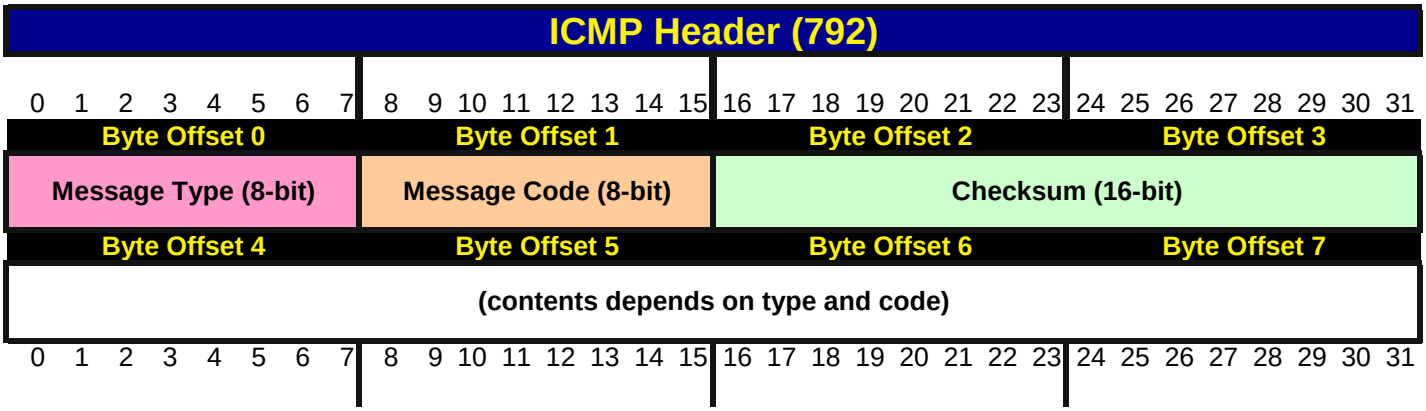
maximum value 65515 (or 65507 bytes of UDP data)

(Max IP is 65535 bytes - 20 byte header = 65515 bytes for UDP packet - 8 bytes UDP header = 65507)

Checksum

Covers psedoheader and entire UDP datagram

(Note: By RFC, the crc is not required)

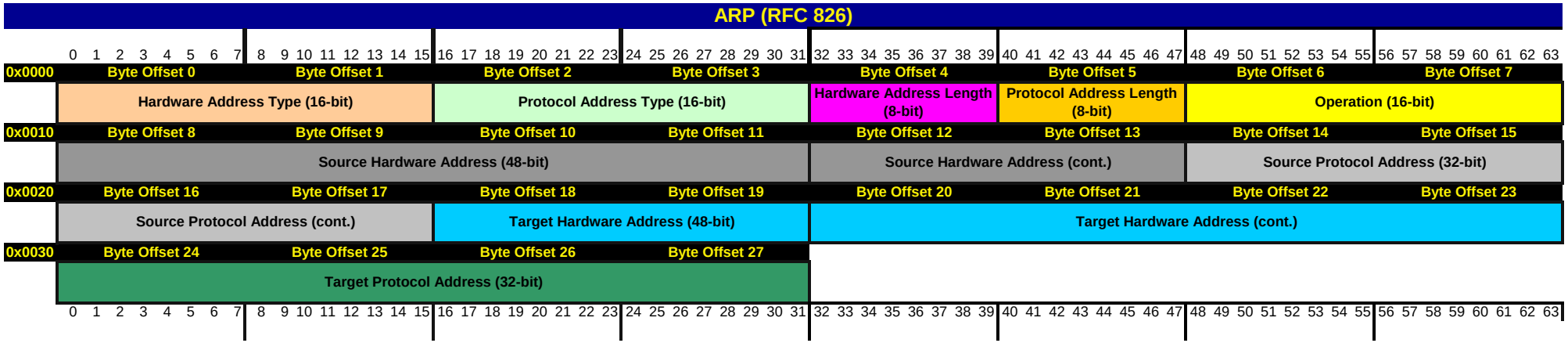


Common Types & Codes

- 0 Echo reply
- 3 Destination Unreachable
  - 0 Net Unreachable
  - 1 Host Unreacheable
  - 2 Protocol Unreachable
  - 3 Port Unreachable
  - 4 Fragmentation Needed & Don't Fragment Flag Set
  - 5 Source Route Failed
  - 6 Destionation Network Unknown
  - 7 Destination Host Unknown
  - 8 Source Route Isolated
  - 9 Network Administratively Prohibited
  - 10 Host Administratively Prohibited
  - 11 Network Unreachable for TOS
  - 12 Host Unreachable for TOS
  - 13 Communication Administratively Prohibited
- 4 Source Quench
- 5 Redirect
  - 0
  - 1
  - 2
  - 3
- 8 Echo
- 9 Router Advertisement
- 10 Router Selection
- 11 Time Exceeded
  - 0 Time to Live exceeded in transit
  - 1 Fragment Reassembly Time Exceeded
- 12 Parameter Problem
  - 0 Pointer indicates the error
  - 1 Missing a Required Option
  - 2 Bad Length
- 13 Timestamp Request
- 14 Timestamp Reply
- 15 Information Request
- 16 Information Reply
- 17 Address Mask Request
- 18 Address Mask Reply
- 30 Traceroute

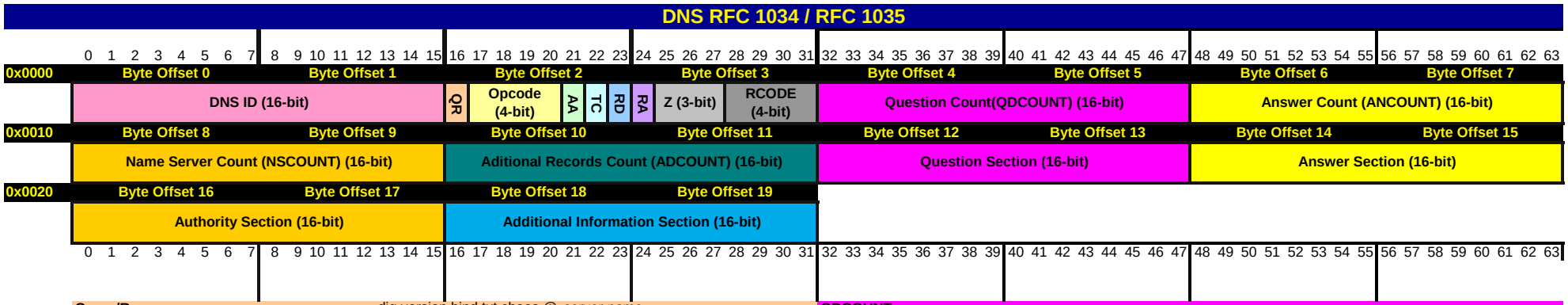
(Note: Byte offset 4-5: identification #)
(Note: Byte offset 6-7: sequence #)

ICMP Hdr



ARP maps the logical address (IP) to the physical address (MAC)

Hardware Address Type		Protocol Address Type		Hardware Address Length		Protocol Address Length		Operation	
1	Ethernet	2048	IPv4 (0x0800)	6	for Ethernet/IEEE 802	4	for IPv4	1	Request
6	IEEE 802 Lan							2	Reply



Query/Response
0 Query
1 Response

dig version.bind txt chaos @ *server name*
dig @ *server name* txt chaos version.bind

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

AA
1 Athoritative Answer

TC
1 Truncation

RD
1 Recursion Desired

RA
1 Recursion Available

Z
Reserved; set to 0

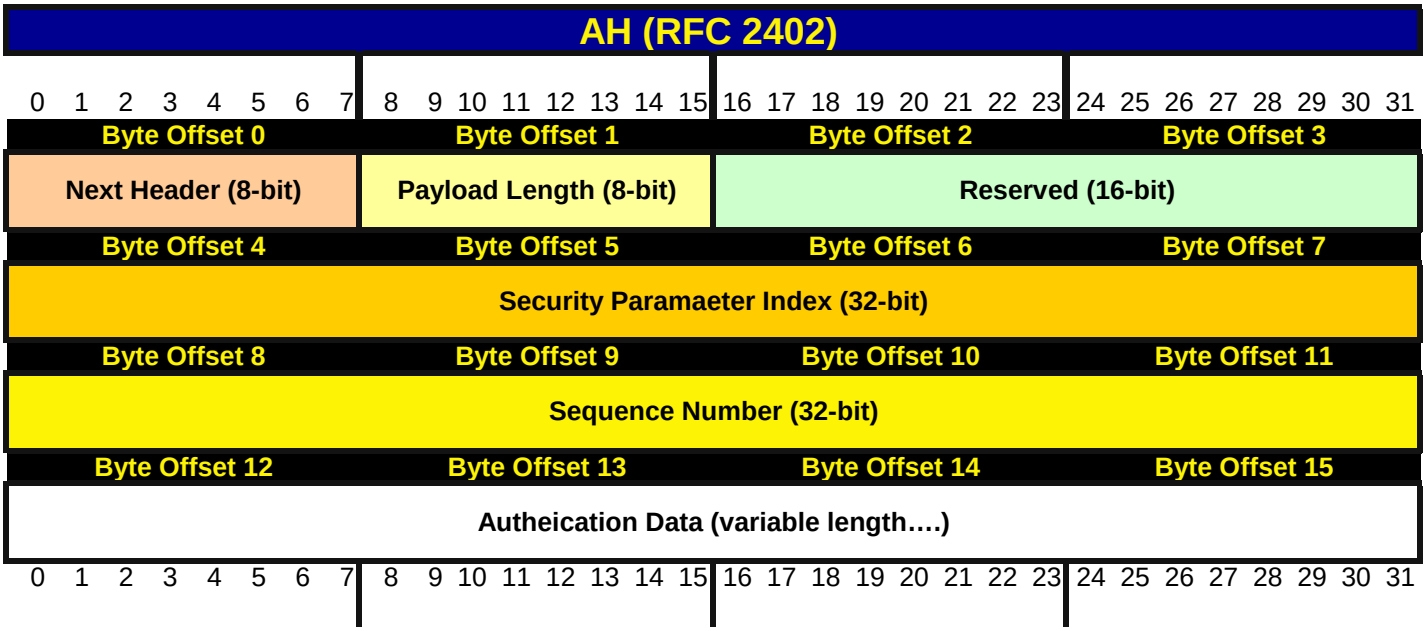
Response Code
0 No Error
1 Format Error
2 Server Failure
3 Non-existant Domain (NXDOMAIN)
4 Query Type Not Implemented
5 Query Refused

NSCOUNT
(Number of name server resource records in Authority section)

ARCOUNT
(Number of resource records in Additional Information section)

QDCOUNT
(Number of entries in Question section)

ANCOUNT
(Number of resource records in Answer section)



Next Header

Equivalent to the IP Protocol Identifier field in IPv4

D	Hex		D	Hex		D	Hex		D	Hex	
1	0x01	ICMP	9	0x09	IGRP	47	0x2F	GRE	88	0x58	EIGRP
2	0x02	IGMP	17	0x11	UDP	50	0x32	ESP	89	0x59	OSPF
6	0x06	TCP	47	0x2F	GRE	51	0x33	AH			

Payload Length

Specifies the length of the Authentication Header (number of 32-bit words - 2 for IPv6 compatibility)

Reserved

Zero filled field

Security Parameter Index (SPI)

Random 32-bit value used with dst IP address and IP Sec protocol to uniquely identify the SA.

The SPI is generally selected by the dst IP Sec node.

Sequence Number

A 32-bit sequence number starting at zero and incremented by one for each packet.

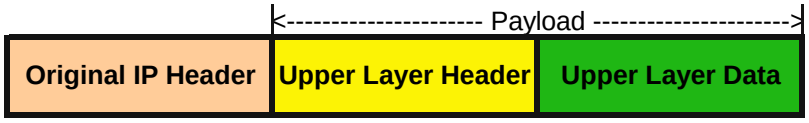
This monotonically increasesing sequence number is the AH anti-replay mechanism.

Authentication Data

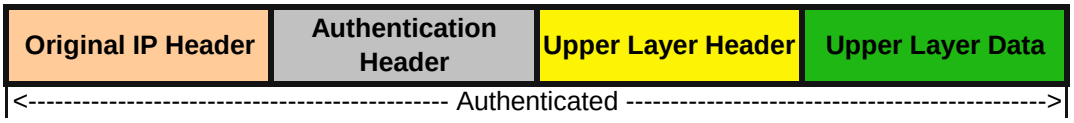
A variable-length field that contains the Integrity Check Value (ICV) for the packet.

The length of the IVC must be an integral multiple of 32 bits; will ne padded or truncated to meet the requirement.

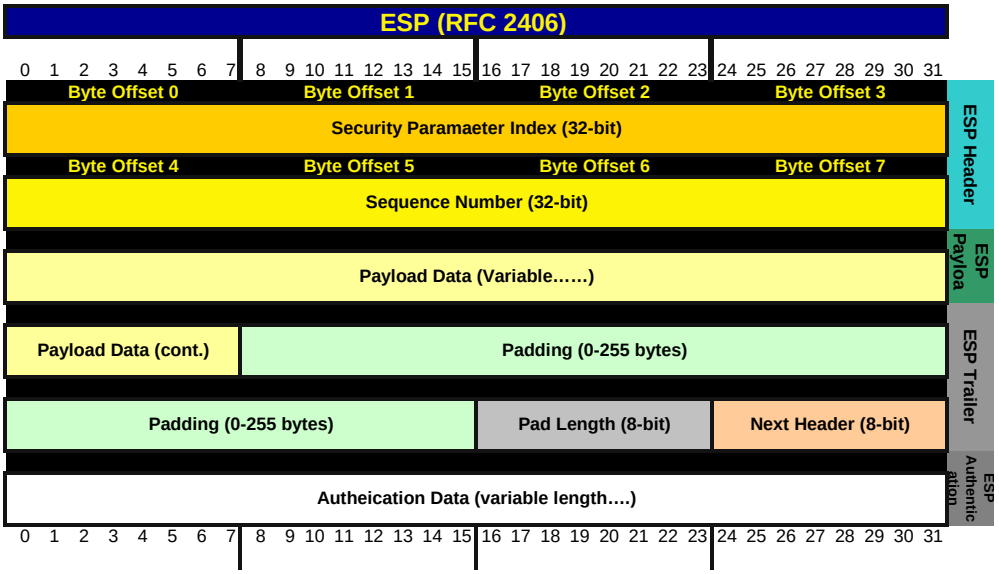
Original Packet



AH Tranport Mode Packet



AH Tunnel Mode Packet



ESP Header

Security Parameter Index (SPI)

Random 32-bit value used with dst IP address and IP Sec protocol to uniquely identify the SA.
The SPI is generally selected by the dst IP Sec node.

Sequence Number

A 32-bit sequence number starting at zero and incremented by one for each packet.
This monotonically increasesing sequence number is the AH anti-replay mechanism.

ESP Payload

Payload Data

A variable-length field containing the data to be protected by the ESP protocol; i.e., the original IP packet

ESP Trailer

Padding

A 0-255 byte field used for varity of purposes. It is primarily used to ensure that the Payload, Pad Length, & Next Header align on a 32-bit boundary. It can also be used if the ESP encryption algorithm requires a certain minimum number of bytes. Finally, it may be used to hide the real size of the payload (protect againts traffic flow analysis)

Pad Length

8-bit value indicating the number of Pad bytes that were inserted.

Next Header

Equivalent to the IP Protocol Identifier field in IPv4

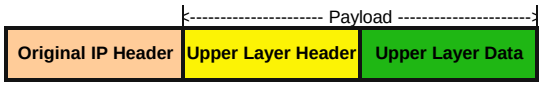
D	Hex		D	Hex		D	Hex		D	Hex	
1	0x01	ICMP	9	0x09	IGRP	47	0x2F	GRE	88	0x58	EIGRP
2	0x02	IGMP	17	0x11	UDP	50	0x32	ESP	89	0x59	OSPF
6	0x06	TCP	47	0x2F	GRE	51	0x33	AH			

ESP Authentication

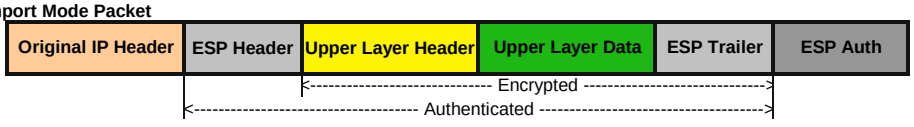
Authentication Data

A variable-length field that contains the Integrity Check Value (ICV) for ESP the packet. The length of the this field is dependent upon the authentication function used. This field is peresent only if an authenication service is being employed in the SA.

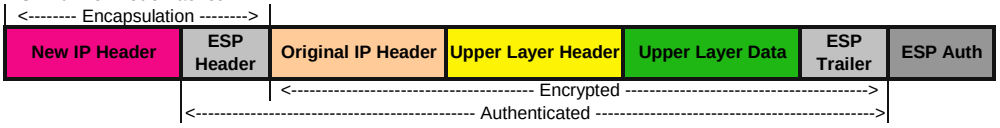
Original Packet



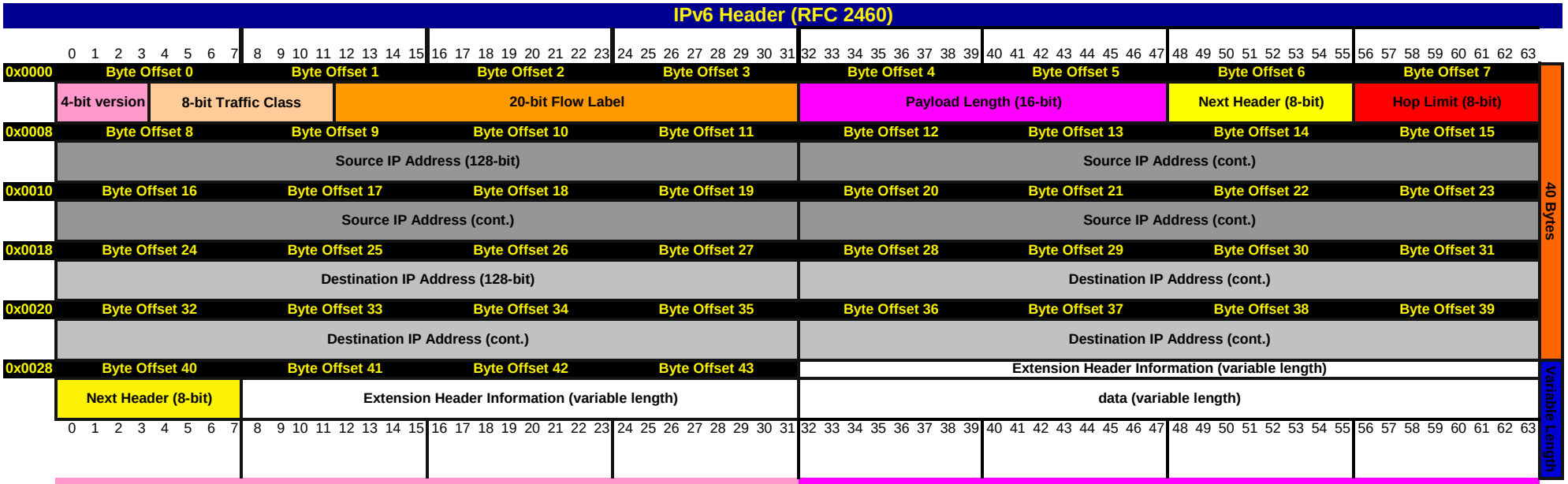
ESP Tranport Mode Packet



ESP Tunnel Mode Packet

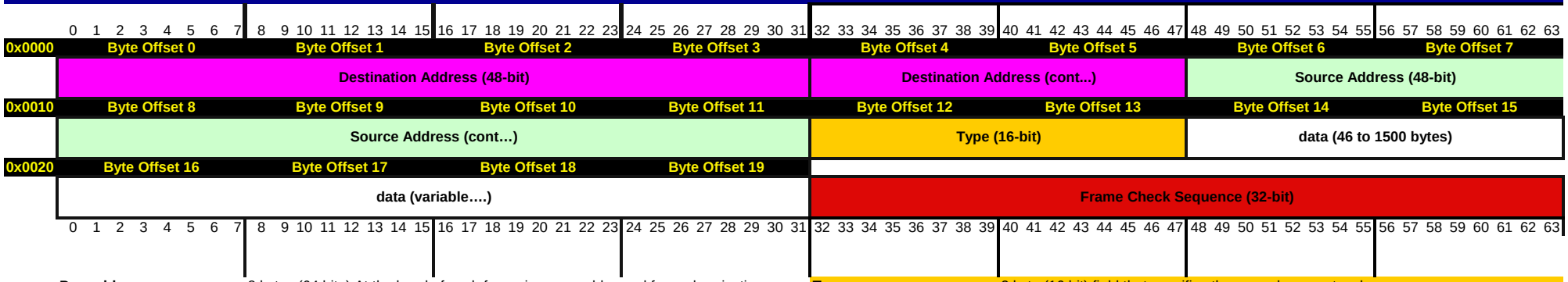


ESP



IPv6 Hdr

Ethernet Version 2 Frame Format



Preamble: 8 bytes (64 bite) At the head of each frame is a preamble used for sychronization 1010…10101011

Destinnation Address: 6 byte (48 bit) desination media access control (MAC) address

Soutce Address: 6 byte (48 bit) source media access control (MAC) address

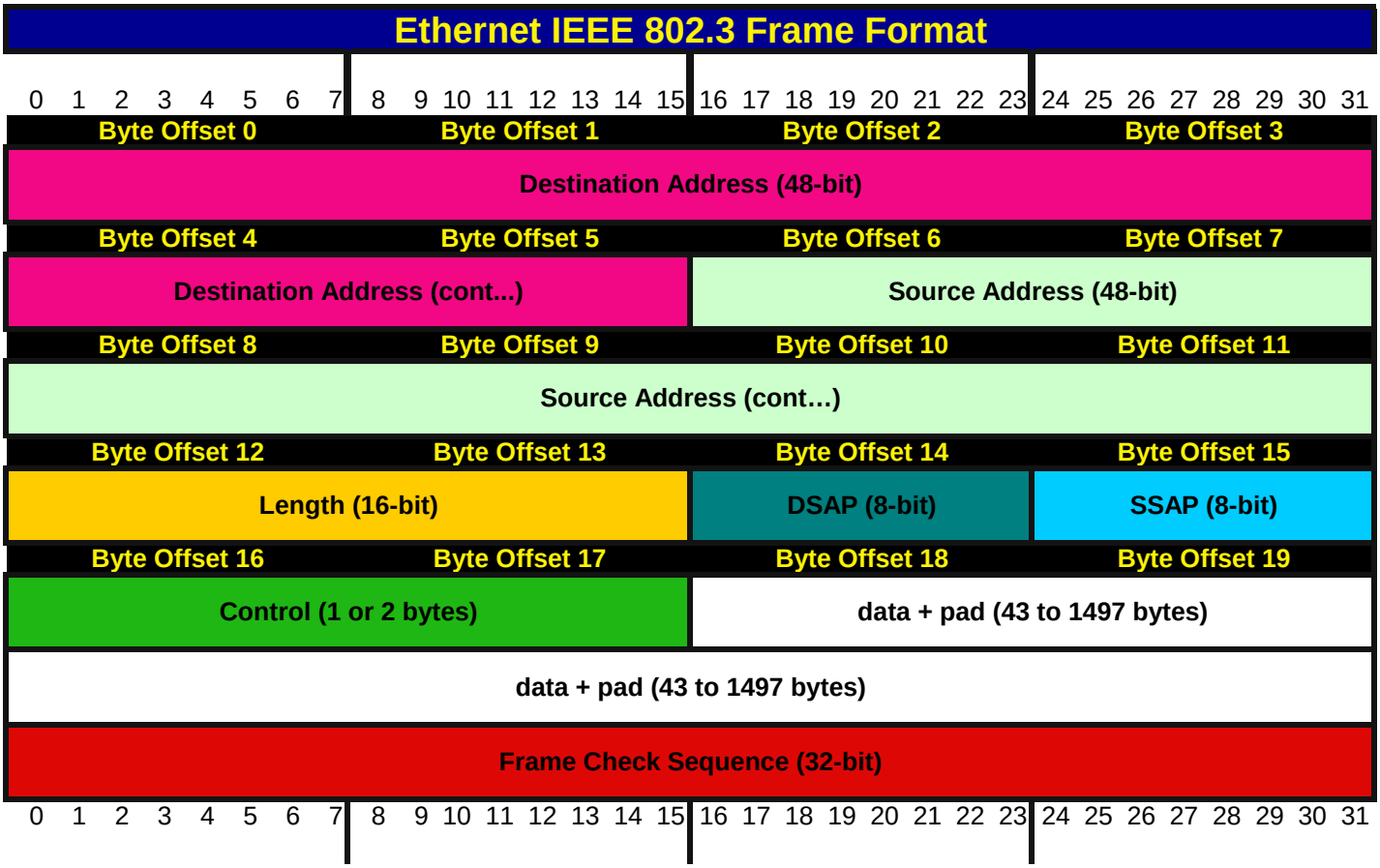
Data: 46 to 1500 bytes of upper-layer protocol information

Frame Check Sequence: The cyclic redundancy check (CRC) or checksum for the Ethernet Frame

Type: 2 byte (16 bit) field that specifies the upper-layer protocol

Type	Value
NetWare	8137
XNS	0600, 0807
IP	800
IP (VINES)	0BAD, 80C4
ARP	806
RARP	8035
DRP	6003
LAT	6004
LAVC	6007
ARP (Atalk)	80F3

Ether v2 Hdr



Preamble: 8 bytes (64 bite) At the head of each frame is a preamble used for sychronization 1010…10101011

Destinnation Address: 6 byte (48 bit) desination media access control (MAC) address

Soutce Address: 6 byte (48 bit) source media access control (MAC) address

Length: 2 byte (16 bit) field that specifies the number of bytes (3-1500) in the LLC and data fields

Logical Link control The logical link control (LLC) is made up of the DSAP, SSAP and Control fields. This is a mothed for telling the 802.3 IEEE and Netware (RAW) formats. The IEEE 802.3 format has the LLS and the NetWare 802.3 "Raw" for mate does not.

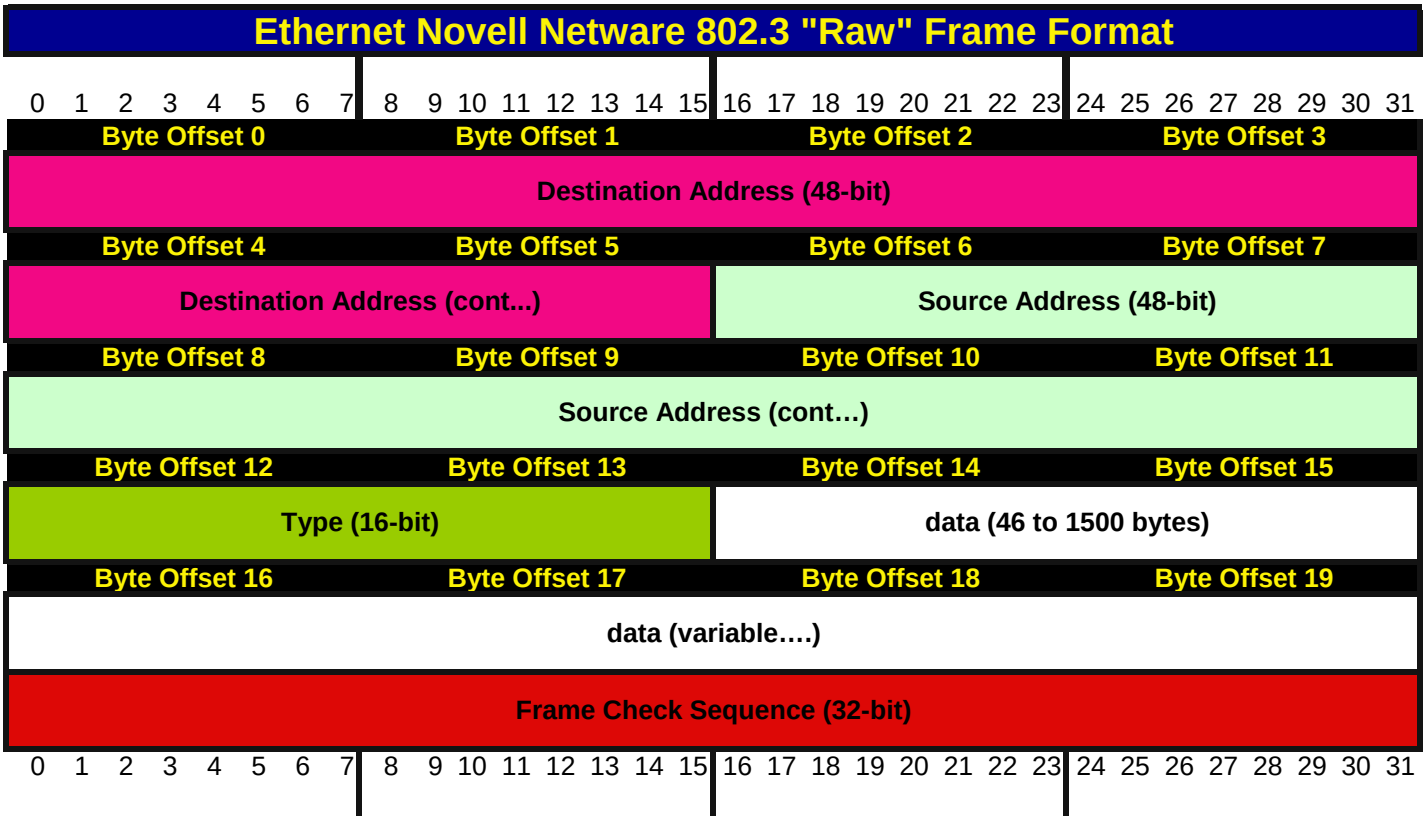
- **DSAP:** 1 byte destination service access point; receiving process at destination
- **SSAP:** 1 byte source service access point; sending process at source
- **Control:** 1 byte is various control information (Connection less)
  2 bytes are for connection-oriented LLC

Pad: Pads the frame to minimum of 46 bytes of data and LLC (so collisions can be detected)

Data: 46 to 1500 bytes of upper-layer protocol information

Frame Check Sequence: The cyclic redundancy check (CRC) or checksum for the Ethernet Frame

Ether 802.3 Hdr



IP Version Number

Preamble: 8 bytes (64 bite) At the head of each frame is a preamble used for sychronization 1010…10101011

Destinnation Address: 6 byte (48 bit) desination media access control (MAC) address

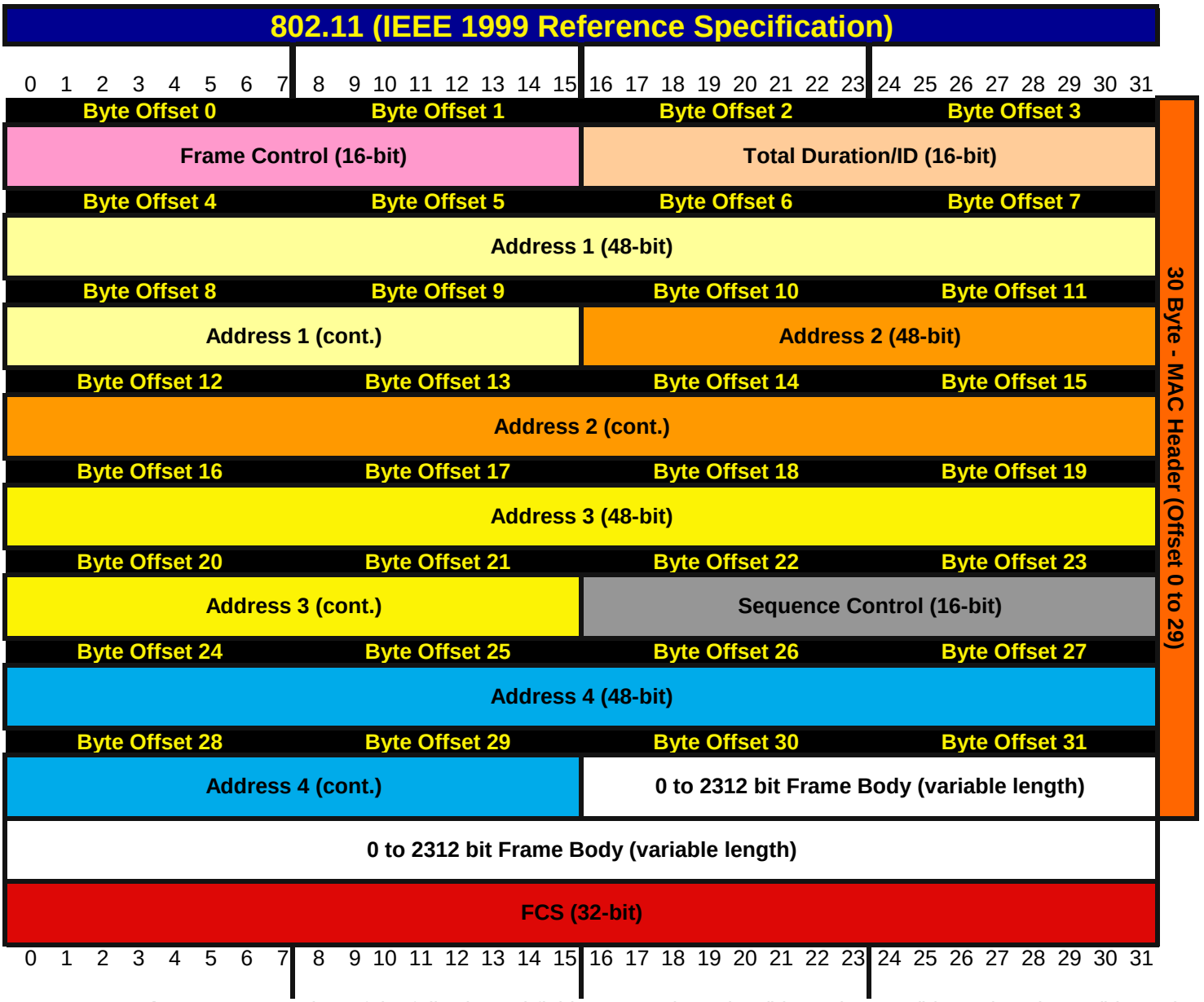
Soutce Address: 6 byte (48 bit) source media access control (MAC) address

Length: 2 byte (16 bit) field that specifies the number of bytes (46-1500) in the LLC and data fields Note the lack of the LLC fields, this is who you tell Netware 802.3 from IEEE 802.3

Data: 46 to 1500 bytes of upper-layer protocol information. IPX header starting with 2 byte checksum (usually FFF) followed by NetWare higher layers ('data')

Frame Check Sequence: The cyclic redundancy check (CRC) or checksum for the Ethernet Frame

Ether 802.3 RAW Hdr



Frame Control Consists of the following subfields: Protocol Version (bits 0-1), Type (bits 2-3), Subtyoe (bits 4-7), To DS (bit 8), From DS (bit 9), More Fragment (bit 10), Retry (bit 11), Power management (bit 12), More Data (bit 13), WEP (bit 14) and Order (bit 15)

Duration / ID

Duration/ID field encoding

15	14	bit 13 - 0	Usage
0	0 - 32767		Duration
1	0	0	Fixed value within frames transmitted during the CFP
1	0	1-16383	Reserved
1	1	0	Reserved
1	1	1-2007	AID in PS-Poll frames
1	1	2008 - 16383	Reserved

Address Fields There are 4 address fields in the MAC frame format. These fields are used to indicate the BSSID, source address (SA), destination address (DA), transmitting station address (TA), and the receiving station address (RA).

Sequence Control Consists of the following subfields: Fragment Number (bits 0-3) and Sequence Number (bits 4-15).

Frame Body Variable length field that contains information specific to individual frame types and subtypes.

FCS 32-bit check sum field calculated over all the fields of the MAC header and Frame body

TCPDUMP / WINDUMP

windump -i <interface> -nx	capture from interface (-i <interface>) do not convert names(-n) and print on hex and ascii (-x)
windump -i <interface> -nx -s0	capture from interface (-i <interface>) do not convert names(-n), print on hex and ascii (-x) and capture all the packet
windump -r <file> -nxp	capture from file (-r <file>), do not convert names (-n), print out hex and ascii (-x), not in permiscous mode (-p)

Keywords

host (host)	**ip**	**vrrp**	**ether multicast**
src host (host)	**ip6**	**ip broadcast**	**vlan** (vlan_id)
dst host (host)	**arp**	**ip proto** (protocol)	**atalk**
gateway (host)	**icmp**	**ip protochan** (protocol)	**decnet**
net (net/len)	**icmp6**	**ip6 proto** (protocol)	**decnet src**
src net (net)	**tcp**	**ip6 protochain** (protocol)	**decnet host**
dst net (net)	**udp**	**ip multicast**	**iso**
port (port)	**ah**	**ip6 multicast**	**stp**
src port (port)	**esp**	**ether host** (MAC)	**ipx**
dst port (port)	**igmp**	**ether src** (MAC)	**netbeui**
less (length)	**igrp**	**ether dst** (MAC)	
greater (length)	**rarp**	**ether proto** (protocol)	

Bit Masking	tcpflags	icmptype			
And unwanted bits with 0	tcp-fin		icmp-echoreply	icmp-echo	icmp-paramprob
And wanted bits with 1	tcp-syn		icmp-unreachable	icmp-ireq	icmp-tstamp
0 AND 0 = 0	tcp-rst		icmp-sourcequench		icmp-tstampreply
0 AND 1 = 0	tcp-push		icmp-redirect		icmp-ireq
1 AND 0 = 0	tcp-ack		icmp-routeradvert		icmp-ireqreply
1 AND 1 = 1	tcp-urg		icmp-routersolicit		icmp-maskreq
			icmp-timxceed		icmp-maskreply

Expressions: >, <, >=, <=, =, !=, +, -, *, /, &, | **!** or **not** **&&** or **and** **||** or **or**

filter format <protocol header>[offset:length]<relation><value>
tcpdump [command line options] ['filter']
windump [command line options] ["filter"]

Examples

host A and B	Connections between host and and host B				
ip[9] = 1	icmp	ip[9] = 6	tcp	ip[9] = 17	udp
tcp[2:2] < 20	The TCP dst port is greater than 20			udp[6:2] != 0	Non-zero UDP checksum
tcp[tcpflags]=tcp-syn	Only Syn	tcp[13] &0x02 != 0	At minimum the SYN bit set		
tcp[tcpflags]=tcp-ack	Only Ack	tcp[13] &0x10 != 0	At minimum the ACK bit set		
tcp[tcpflags]=tcp-fin	or	tcp[13] &0xff = 0x1	Only the FIN bit is set		tcp[13] &0xff = 1
tcp[13] &0xff =16	or	tcp[13] &0xff = 0x10	Only the ACK bit is set		
icmp[0]=3 and icmp[1]=2	icmp type 3 is destination unreachable category and a code of 2 specifies that this is an ICMP protocol unreachable (**Good filter for detecting protocol scans**)				
(tcp and (tcp[13] &0x0f != 0) and not port 25 and not port 20)	A tcp packet where any combination of PSH, RST, SYN, FIN are set and the packet is not port 25 or 20				
udp[21:4]=0x56455253	Looks for "VERS" in udp payload for VERSION.BIND				
tcp[20:4] = 0x5353482d	Looks for "SSH-" in TCP payload				
ip[6:2] & 0x3fff != 0	Look for ALL fragmented ip packets				
ip[6] &0x20 = 0x20 or ip[6:2] &0x1fff != 0	Look for more fragment bit set **or** fragment offset greater than 0 (**Look for ALL fragmented ip packets**)				
ip[6] &0x20 = 0 and ip[6:2] &0x1fff != 0	Look for more fragment bit **not** set **and** fragment offset greater than 0 (**Last fragment packets**)				

Command Line Options

Options	Description
-a	Attempt to convert network and broadcast addresses to names
-A	
-B <size>	Set driver's buffer size to size in KiloBytes. The default buffer size is 1 megabyte (i.e 1000).
-c <count>	Exit after receiving <count> of packets
-C <file size>	Before writing a raw packet to a savefile, check whether the file is currently larger than file_size
-d	Dump the compiled packet-matching code in a human readable form to standard output and stop
-dd	Dump packet-matching code as a C program fragment
-ddd	ddd Dump packet-matching code as decimal numbers (preceded with a count)
-D	Print the list of the interface cards available on the system. WINDUMP ONLY
-e	Print the link-level header on each dump line
-E <algo:secret>	Use algo:secret for decrypting IPsec ESP packets where algorithms may be des-cbc, 3des-cbc,
-f	Print 'foreign' internet addresses numerically rather than symbolically
-F <file>	Use file as input for the filter expression
-i <interface>	Listen on interface (defaults to lowest numbered interface)
-l	Make stdout line buffered. ``tcpdump -l | tee dat'' or ``tcpdump -l > dat & tail -f dat''
-L	
-m <module>	Load SMI MIB module definitions from file module
-n	Don't convert addresses to names
-N	Don't print domain name qualification of host names
-O	Do not run the packet-matching code optimizer
-p	Don't put the interface into promiscuous mode
-q	Quick output – print less protocol information
-r <file>	Read packets from file (created with the –w option)
-R	Assume ESP/AH packets to be based on old specs
-s <snaplen>	Snarf snaplen bytes of data from each packet (default is 68) 1518 Max Ethernet Frame (14 byte Ethernet header + 1500 byte IP + 4 byte Ethernet trailer) 64 Min Ethernet Frame (14 byte Ethernet header + 64 byte IP + 4 byte Ethernet trailer) **Note: -s0 mean full ethernet packet**
-S	Print absolute, rather than relative TCP sequence numbers
-t	Don't print a timestamp on each dump line
-T <type>	Force packets selected by "expressions" to be interpreted the specified type (cnfp, rpc, rtp, snmp,
-tt	Print an unformatted timestamp on each dump line
-ttt	Print a delta (in micro-seconds) between current and previous line on each dump line
-tttt	Print a timestamp in default format proceeded by date on each dump line
-u	Print undecoded NFS handles
-U	
-v	Verbose output (TOS, TTL, IP ID, Fragment Offset, IP Flags, length)
V	
-w <file>	Write the raw packet to file rather than parsing and printing to stdout
-x	Print each packet (minus link level header) in hex
-X	Print each packet in hex and ascii
-y <datalinktype>	

http://www.tcpdump.org/tcpdump_man.html

http://windump.polito.it/docs/manual.htm#Wdump

TCPDUMP

NGREP

ngrep

<-hXViwqpevxlDtT> <-IO pcap_dump> <-n num> <-d dev> <-A num> <-s snaplen> <-S limitlen>

<match expression>

<bpf filter>

Command Line Options	
-A (num)	is dump num packets after a match
-D	is replay pcap_dumps with their recorded time intervals
-d (device)	is use a device different from the default (pcap)
-e	is show empty packets
-h	is help/usage
-i	is ignore case
-I (file)	is read packet stream from pcap format file pcap_dump (**Capitol i**)
-l	is make stdout line buffered
-n (num)	is look at only num packets
-O (file)	is dump matched packets in pcap format to pcap_dump
-p	is don't go into promiscuous mode
-q	is be quiet
-S (limitlen)	is set the limitlen on matched packets
-s (snaplen)	is set the bpf caplen
-t	is print timestamp every time a packet is matched
-T	is print delta timestamp every time a packet is matched
-V	is version information
-v	is invert match
-w	is word-regex (expression must match as a word)
-X	is interpret match expression as hexadecimal
-x	is print in alternate hexdump format

<match expression>	is either an extended regular expression or a hexadecimal string. see the man page for more information.
<bpf filter>	is any bpf filter statement.

Examples:

ngrep " icmp	print all UDP packets
ngrep " tcp	print all TCP packets
ngrep " udp	print all UDP packets
ngrep " port 53	print all packets to or from TCP or TDP port 53
ngrep " tcp port 53	print all packets to or from only TCP port 53
ngrep - v " tcp port 53	print all packets but those to or from TCP port 53
ngrep 'USER|PASS' tcp port 21	print all packets to or from TCP port 21 where USER or PASS
ngrep 'SSH-' port tcp 22	print all packets to or from TCP port 22 where SSH-
ngrep 'LILWORD' port 138	print Microsoft browsing traffic for NT domain LILWORLD
ngrep -iq 'rcpt to|mail from' tcp port 25	monitor current delivery and print sender and recipients
ngrep 'user' port 110	monitor POP3
ngrep -q 'abcd' icmp	"pinging" host running a Microsoft operating system?
ngep -i -I <input file> "Yahoo"	read from input file and search for case insensitive "Yahoo"

OS Fingerprinting

OS	Version	Platform	TTL	Window	DF	TOS	TCP Options
DC-Osx	1.1-95	Pyramid/NILE	30	8192	n	0	
Windows	9x/NT	Intel	32	5000-9000	y	0	
NetApp	OnTap	5.1.2-5.2.2	54	8760	y	0	
HPJetDirect	?	HP_Printer	59	2100-2150	n	0	
AIX	4.3.X	IBM/RS6000	60	16000-16100	y	0	MSS
AIX	4.2.X	IBM/RS6000	60	16000-16100	n	0	
Cisco	11.2	7507	60		y	0	
DigitalUnix	4	Alpha	60		y	16	
IRIX	6.x	SGI	60		y	16	
OS390	2.6	IBM/S390	60		n	0	
Reliant	5.43	Pyramid/RM1000	60		n	0	
FreeBSD	3.x	Intel	64		y	16	
JetDirect	G.07.x	J311A	64		n	0	
Linux	2.2.x	Intel	64	32120	y	0	MSS, SackOK, wscale, Timestamp, one NOP
Linux	2.4	Intel	64	5840			MSS, SackOK, wscale, Timestamp, one NOP
OpenBSD	2.x	Intel	64		n	16	MSS, Timestamp, wscale, sacks OK, 5 nops
0s/400	r4.4	AS/400	64		y	0	
SCO	R5	Compaq	64		n	0	
Solaris	8	Intel/Sparc	64		y	0	
FTX(Unix)	3.3	STRATUS	64	32678	n	0	
Unisys	x	Mainframe	64	32768	n	0	
Netware	4.11	Intel	126	32000-32768	y	0	
Windows	9x/NT	Intel	128	5000-9000	y	0	
Windows	2000	Intel	128	17000-18000	y	0	MSS, SackOK, 2 NOPs
Cisco	12	2514	255	3800-5000	n	192	
Solaris	2.x	Intel/Sparc	255	8760	y	0	

ADDITIONAL NOTES
#
Cisco IOS 12.0 normally starts all IP sessions with IP ID of 0
Solaris 8 uses a smaller TTL (64) then Solaris 7 and below (255).
Windows 2000 uses a much larger Window Size then NT.

Decimal to hexadecimal to ASCII Chart

Dec	Hex	ASCII
0	0	NUL
1	1	SOH
2	2	STX
3	3	ETX
4	4	EOT
5	5	ENQ
6	6	ACK
7	7	BEL
8	8	BS
9	9	HT
10	A	LF
11	B	VT
12	C	FF
13	D	CR
14	E	SO
15	F	SI
16	10	DLE
17	11	DC1
18	12	DC2
19	13	DC3
20	14	DC4
21	15	NAK
22	16	SYN
23	17	ETB
24	18	CAN
25	19	EM
26	1A	SUB
27	1B	ESC
28	1C	FS
29	1D	GS
30	1E	RS
31	1F	US

Dec	Hex	ASCII
32	20	SP
33	21	!
34	22	"
35	23	#
36	24	$
37	25	%
38	26	&
39	27	'
40	28	(
41	29	)
42	2A	*
43	2B	+
44	2C	,
45	2D	-
46	2E	.
47	2F	/
48	30	0
49	31	1
50	32	2
51	33	3
52	34	4
53	35	5
54	36	6
55	37	7
56	38	8
57	39	9
58	3A	:
59	3B	;
60	3C	<
61	3D	=
62	3E	>
63	3F	?

Dec	Hex	ASCII
64	40	@
65	41	A
66	42	B
67	43	C
68	44	D
69	45	E
70	46	F
71	47	G
72	48	H
73	49	I
74	4A	J
75	4B	K
76	4C	L
77	4D	M
78	4E	N
79	4F	O
80	50	P
81	51	Q
82	52	R
83	53	S
84	54	T
85	55	U
86	56	V
87	57	W
88	58	X
89	59	Y
90	5A	Z
91	5B	[
92	5C	\
93	5D	]
94	5E	^
95	5F	

Dec	Hex	ASCII
96	60	'
97	61	a
98	62	b
99	63	c
100	64	DEL
101	65	e
102	66	f
103	67	g
104	68	h
105	69	i
106	6A	j
107	6B	k
108	6C	l
109	6D	m
110	6E	n
111	6F	o
112	70	p
113	71	q
114	72	r
115	73	s
116	74	t
117	75	u
118	76	v
119	77	w
120	78	x
121	79	y
122	7A	z
123	7B	{
124	7C	|
125	7D	}
126	7E	~
127	7F	DEL

Dec	Hex	ASCII
128	80	Ç
129	81	ü
130	82	é
131	83	â
132	84	ä
133	85	à
134	86	å
135	87	ç
136	88	ê
137	89	ë
138	8A	è
139	8B	ï
140	8C	î
141	8D	ì
142	8E	Ä
143	8F	Å
144	90	É
145	91	æ
146	92	Æ
147	93	ô
148	94	ö
149	95	ò
150	96	û
151	97	ù
152	98	ÿ
153	99	Ö
154	9A	Ü
155	9B	¢
156	9C	£
157	9D	¥
158	9E	₧
159	9F	ƒ

Dec	Hex	ASCII
160	A0	á
161	A1	í
162	A2	ó
163	A3	ú
164	A4	ñ
165	A5	Ñ
166	A6	ª
167	A7	º
168	A8	¿
169	A9	⌐
170	AA	¬
171	AB	½
172	AC	¼
173	AD	¡
174	AE	«
175	AF	»
176	B0	▒
177	B1	▒
178	B2	▓
179	B3	│
180	B4	┤
181	B5	╡
182	B6	╢
183	B7	╖
184	B8	╕
185	B9	╣
186	BA	║
187	BB	╗
188	BC	╝
189	BD	╜
190	BE	╛
191	BF	┐

Dec	Hex	ASCII
192	C0	└
193	C1	┴
194	C2	┬
195	C3	├
196	C4	─
197	C5	┼
198	C6	╞
199	C7	╟
200	C8	╚
201	C9	╔
202	CA	╩
203	CB	╦
204	CC	╠
205	CD	═
206	CE	╬
207	CF	╧
208	D0	╨
209	D1	╤
210	D2	╥
211	D3	╙
212	D4	╘
213	D5	╒
214	D6	╓
215	D7	╫
216	D8	╪
217	D9	┘
218	DA	┌
219	DB	▉
220	DC	▊
221	DD	▋
222	DE	▌
223	DF	▍

Dec	Hex	ASCII
224	E0	α
225	E1	ß
226	E2	Γ
227	E3	π
228	E4	Σ
229	E5	σ
230	E6	µ
231	E7	τ
232	E8	Φ
233	E9	Θ
234	EA	Ω
235	EB	δ
236	EC	∞
237	ED	φ
238	EE	ε
239	EF	∩
240	F0	≡
241	F1	±
242	F2	≥
243	F3	≤
244	F4	⌠
245	F5	⌡
246	F6	÷
247	F7	≈
248	F8	°
249	F9	∙
250	FA	·
251	FB	√
252	FC	ⁿ
253	FD	²
254	FE	■
255	FF	Hardspace

ASCII

TCPDUMP

packetlife.net

Command Line Options

-A	Print frame payload in ASCII	**-q**	Quick output
-c <count>	Exit after capturing **count** packets	**-r <file>**	Read packets from **file**
-D	List available interfaces	**-s <len>**	Capture up to **len** bytes per packet
-e	Print link-level headers	**-S**	Print absolute TCP sequence numbers
-F <file>	Use **file** as the filter expression	**-t**	Don't print timestamps
-G <n>	Rotate the dump file every n seconds	**-v[v[v]]**	Print more verbose output
-i <iface>	Specifies the capture interface	**-w <file>**	Write captured packets to **file**
-K	Don't verify TCP checksums	**-x**	Print frame payload in hex
-L	List data link types for the interface	**-X**	Print frame payload in hex and ASCII
-n	Don't convert addresses to names	**-y <type>**	Specify the data link type
-p	Don't capture in promiscuous mode	**-Z <user>**	Drop privileges from root to **user**

Capture Filter Primitives

[src|dst] host <host>	Matches a host as the IP source, destination, or either
ether [src|dst] host <ehost>	Matches a host as the Ethernet source, destination, or either
gateway host <host>	Matches packets which used **host** as a gateway
[src|dst] net <network>/<len>	Matches packets to or from an endpoint residing in **network**
[tcp|udp] [src|dst] port <port>	Matches TCP or UDP packets sent to/from **port**
[tcp|udp] [src|dst] portrange <p1>-<p2>	Matches TCP or UDP packets to/from a port in the given range
less <length>	Matches packets less than or equal to **length**
greater <length>	Matches packets greater than or equal to **length**
(ether|ip|ip6) proto <protocol>	Matches an Ethernet, IPv4, or IPv6 protocol
(ether|ip) broadcast	Matches Ethernet or IPv4 broadcasts
(ether|ip|ip6) multicast	Matches Ethernet, IPv4, or IPv6 multicasts
type (mgt|ctl|data) [subtype <subtype>]	Matches 802.11 frames based on type and optional subtype
vlan [<vlan>]	Matches 802.1Q frames, optionally with a VLAN ID of **vlan**
mpls [<label>]	Matches MPLS packets, optionally with a label of **label**
<expr> <relop> <expr>	Matches packets by an arbitrary expression

Protocols

arp	**ip6**	**slip**
ether	**link**	**tcp**
fddi	**ppp**	**tr**
icmp	**radio**	**udp**
ip	**rarp**	**wlan**

TCP Flags

tcp-urg	**tcp-rst**
tcp-ack	**tcp-syn**
tcp-psh	**tcp-fin**

Modifiers

! or **not**
&& or **and**
|| or **or**

Examples

udp dst port not 53	UDP not bound for port 53
host 10.0.0.1 && host 10.0.0.2	Traffic between these hosts
tcp dst port 80 or 8080	Packets to either TCP port

ICMP Types

icmp-echoreply	**icmp-routeradvert**	**icmp-tstampreply**
icmp-unreach	**icmp-routersolicit**	**icmp-ireq**
icmp-sourcequench	**icmp-timxceed**	**icmp-ireqreply**
icmp-redirect	**icmp-paramprob**	**icmp-maskreq**
icmp-echo	**icmp-tstamp**	**icmp-maskreply**

Wireshark Display Filters · Part 1

packetlife.net

Ethernet

eth.addr	eth.len	eth.src
eth.dst	eth.lg	eth.trailer
eth.ig	eth.multicast	eth.type

IEEE 802.1Q

vlan.cfi	vlan.id	vlan.priority
vlan.etype	vlan.len	vlan.trailer

IPv4

ip.addr	ip.fragment.overlap.conflict
ip.checksum	ip.fragment.toolongfragment
ip.checksum_bad	ip.fragments
ip.checksum_good	ip.hdr_len
ip.dsfield	ip.host
ip.dsfield.ce	ip.id
ip.dsfield.dscp	ip.len
ip.dsfield.ect	ip.proto
ip.dst	ip.reassembled_in
ip.dst_host	ip.src
ip.flags	ip.src_host
ip.flags.df	ip.tos
ip.flags.mf	ip.tos.cost
ip.flags.rb	ip.tos.delay
ip.frag_offset	ip.tos.precedence
ip.fragment	ip.tos.reliability
ip.fragment.error	ip.tos.throughput
ip.fragment.multipletails	ip.ttl
ip.fragment.overlap	ip.version

IPv6

ipv6.addr	ipv6.hop_opt
ipv6.class	ipv6.host
ipv6.dst	ipv6.mipv6_home_address
ipv6.dst_host	ipv6.mipv6_length
ipv6.dst_opt	ipv6.mipv6_type
ipv6.flow	ipv6.nxt
ipv6.fragment	ipv6.opt.pad1
ipv6.fragment.error	ipv6.opt.padn
ipv6.fragment.more	ipv6.plen
ipv6.fragment.multipletails	ipv6.reassembled_in
ipv6.fragment.offset	ipv6.routing_hdr
ipv6.fragment.overlap	ipv6.routing_hdr.addr
ipv6.fragment.overlap.conflict	ipv6.routing_hdr.left
ipv6.fragment.toolongfragment	ipv6.routing_hdr.type
ipv6.fragments	ipv6.src
ipv6.fragment.id	ipv6.src_host
ipv6.hlim	ipv6.version

ARP

arp.dst.hw_mac	arp.proto.size
arp.dst.proto_ipv4	arp.proto.type
arp.hw.size	arp.src.hw_mac
arp.hw.type	arp.src.proto_ipv4
arp.opcode	

TCP

tcp.ack	tcp.options.qs
tcp.checksum	tcp.options.sack
tcp.checksum_bad	tcp.options.sack_le
tcp.checksum_good	tcp.options.sack_perm
tcp.continuation_to	tcp.options.sack_re
tcp.dstport	tcp.options.time_stamp
tcp.flags	tcp.options.wscale
tcp.flags.ack	tcp.options.wscale_val
tcp.flags.cwr	tcp.pdu.last_frame
tcp.flags.ecn	tcp.pdu.size
tcp.flags.fin	tcp.pdu.time
tcp.flags.push	tcp.port
tcp.flags.reset	tcp.reassembled_in
tcp.flags.syn	tcp.segment
tcp.flags.urg	tcp.segment.error
tcp.hdr_len	tcp.segment.multipletails
tcp.len	tcp.segment.overlap
tcp.nxtseq	tcp.segment.overlap.conflict
tcp.options	tcp.segment.toolongfragment
tcp.options.cc	tcp.segments
tcp.options.ccecho	tcp.seq
tcp.options.ccnew	tcp.srcport
tcp.options.echo	tcp.time_delta
tcp.options.echo_reply	tcp.time_relative
tcp.options.md5	tcp.urgent_pointer
tcp.options.mss	tcp.window_size
tcp.options.mss_val	

UDP

udp.checksum	udp.dstport	udp.srcport
udp.checksum_bad	udp.length	
udp.checksum_good	udp.port	

Operators

eq or ==	
ne or !=	
gt or >	
lt or <	
ge or >=	
le or <=	

Logic

and or &&	Logical AND
or or ||	Logical OR
xor or ^^	Logical XOR
not or !	Logical NOT
[n] […]	Substring operator

Wireshark Display Filters · Part 2

packetlife.net

Frame Relay

fr.becn	fr.de
fr.chdlctype	fr.dlci
fr.control	fr.dlcore_control
fr.control.f	fr.ea
fr.control.ftype	fr.fecn
fr.control.n_r	fr.lower_dlci
fr.control.n_s	fr.nlpid
fr.control.p	fr.second_dlci
fr.control.s_ftype	fr.snap.oui
fr.control.u_modifier_cmd	fr.snap.pid
fr.control.u_modifier_resp	fr.snaptype
fr.cr	fr.third_dlci
fr.dc	fr.upper_dlci

PPP

ppp.address	ppp.direction
ppp.control	ppp.protocol

MPLS

mpls.bottom	mpls.oam.defect_location
mpls.cw.control	mpls.oam.defect_type
mpls.cw.res	mpls.oam.frequency
mpls.exp	mpls.oam.function_type
mpls.label	mpls.oam.ttsi
mpls.oam.bip16	mpls.ttl

ICMP

icmp.checksum	icmp.ident	icmp.seq
icmp.checksum_bad	icmp.mtu	icmp.type
icmp.code	icmp.redir_gw	

DTP

dtp.neighbor	dtp.tlv_type	vtp.neighbor
dtp.tlv_len	dtp.version	

VTP

vtp.code	vtp.vlan_info.802_10_index
vtp.conf_rev_num	vtp.vlan_info.isl_vlan_id
vtp.followers	vtp.vlan_info.len
vtp.md	vtp.vlan_info.mtu_size
vtp.md5_digest	vtp.vlan_info.status.vlan_susp
vtp.md_len	vtp.vlan_info.tlv_len
vtp.seq_num	vtp.vlan_info.tlv_type
vtp.start_value	vtp.vlan_info.vlan_name
vtp.upd_id	vtp.vlan_info.vlan_name_len
vtp.upd_ts	vtp.vlan_info.vlan_type
vtp.version	

ICMPv6

icmpv6.all_comp	icmpv6.option.name_type.fqdn
icmpv6.checksum	icmpv6.option.name_x501
icmpv6.checksum_bad	icmpv6.option.rsa.key_hash
icmpv6.code	icmpv6.option.type
icmpv6.comp	icmpv6.ra.cur_hop_limit
icmpv6.haad.ha_addrs	icmpv6.ra.reachable_time
icmpv6.identifier	icmpv6.ra.retrans_timer
icmpv6.option	icmpv6.ra.router_lifetime
icmpv6.option.cga	icmpv6.recursive_dns_serv
icmpv6.option.length	icmpv6.type
icmpv6.option.name_type	

RIP

rip.auth.passwd	rip.ip	rip.route_tag
rip.auth.type	rip.metric	rip.routing_domain
rip.command	rip.netmask	rip.version
rip.family	rip.next_hop	

BGP

bgp.aggregator_as	bgp.mp_reach_nlri_ipv4_prefix
bgp.aggregator_origin	bgp.mp_unreach_nlri_ipv4_prefix
bgp.as_path	bgp.multi_exit_disc
bgp.cluster_identifier	bgp.next_hop
bgp.cluster_list	bgp.nlri_prefix
bgp.community_as	bgp.origin
bgp.community_value	bgp.originator_id
bgp.local_pref	bgp.type
bgp.mp_nlri_tnl_id	bgp.withdrawn_prefix

HTTP

http.accept	http.proxy_authorization
http.accept_encoding	http.proxy_connect_host
http.accept_language	http.proxy_connect_port
http.authbasic	http.referer
http.authorization	http.request
http.cache_control	http.request.method
http.connection	http.request.uri
http.content_encoding	http.request.version
http.content_length	http.response
http.content_type	http.response.code
http.cookie	http.server
http.date	http.set_cookie
http.host	http.transfer_encoding
http.last_modified	http.user_agent
http.location	http.www_authenticate
http.notification	http.x_forwarded_for
http.proxy_authenticate	

IP Packet Fields

IP Protocol ID Numbers

This is a list of IP numbers used in the Protocol field of the IPv4 header and the Next Header field of IPv6 header.

Decimal	Hex	Keyword	Protocol	References
0	0x00	HOPOPT	IPv6 Hop-by-Hop Option	RFC 2460
1	0x01	ICMP	Internet Control Message Protocol	RFC 792
2	0x02	IGMP	Internet Group Management Protocol	RFC 1112
3	0x03	GGP	Gateway-to-Gateway Protocol	RFC 823
4	0x04	IP-in-IP	IP in IP (encapsulation)	RFC 2003
5	0x05	ST	Internet Stream Protocol	RFC 1190, RFC 1819
6	0x06	TCP	Transmission Control Protocol	RFC 793
7	0x07	CBT	Core-based trees	RFC 2189
8	0x08	EGP	Exterior Gateway Protocol	RFC 888
9	0x09	IGP	Interior Gateway Protocol (any private interior gateway (used by Cisco for their IGRP))	
10	0x0A	BBN-RCC-MON	BBN RCC Monitoring	
11	0x0B	NVP-II	Network Voice Protocol	RFC 741
12	0x0C	PUP	Xerox PUP	
13	0x0D	ARGUS	ARGUS	
14	0x0E	EMCON	EMCON	
15	0x0F	XNET	Cross Net Debugger	IEN 158
16	0x10	CHAOS	Chaos	
17	0x11	UDP	User Datagram Protocol	RFC 768
18	0x12	MUX	Multiplexing	IEN 90
19	0x13	DCN-MEAS	DCN Measurement Subsystems	
20	0x14	HMP	Host Monitoring Protocol	RFC 869
21	0x15	PRM	Packet Radio Measurement	
22	0x16	XNS-IDP	XEROX NS IDP	
23	0x17	TRUNK-1	Trunk-1	
24	0x18	TRUNK-2	Trunk-2	
25	0x19	LEAF-1	Leaf-1	
26	0x1A	LEAF-2	Leaf-2	
27	0x1B	RDP	Reliable Datagram Protocol	RFC 908
28	0x1C	IRTP	Internet Reliable Transaction Protocol	RFC 938
29	0x1D	ISO-TP4	ISO Transport Protocol Class 4	RFC 905
30	0x1E	NETBLT	Bulk Data Transfer Protocol	RFC 998
31	0x1F	MFE-NSP	MFE Network Services Protocol	
32	0x20	MERIT-INP	MERIT Internodal Protocol	
33	0x21	DCCP	Datagram Congestion Control Protocol	RFC 4340
34	0x22	3PC	Third Party Connect Protocol	
35	0x23	IDPR	Inter-Domain Policy Routing Protocol	RFC 1479
36	0x24	XTP	Xpress Transport Protocol	
37	0x25	DDP	Datagram Delivery Protocol	
38	0x26	IDPR-CMTP	IDPR Control Message Transport Protocol	
39	0x27	TP++	TP++ Transport Protocol	
40	0x28	IL	IL Transport Protocol	
41	0x29	IPv6	IPv6 Encapsulation	RFC 2473
42	0x2A	SDRP	Source Demand Routing Protocol	RFC 1940
43	0x2B	IPv6-Route	Routing Header for IPv6	RFC 2460
44	0x2C	IPv6-Frag	Fragment Header for IPv6	RFC 2460
45	0x2D	IDRP	Inter-Domain Routing Protocol	
46	0x2E	RSVP	Resource Reservation Protocol	RFC 2205
47	0x2F	GRE	Generic Routing Encapsulation	RFC 2784, RFC 2890
48	0x30	MHRP	Mobile Host Routing Protocol	
49	0x31	BNA	BNA	
50	0x32	ESP	Encapsulating Security Payload	RFC 4303
51	0x33	AH	Authentication Header	RFC 4302
52	0x34	I-NLSP	Integrated Net Layer Security Protocol	TUBA
53	0x35	SWIPE	SwIPe	IP with Encryption
54	0x36	NARP	NBMA Address Resolution Protocol	RFC 1735
55	0x37	MOBILE	IP Mobility (Min Encap)	RFC 2004

IP Packet Fields

Decimal	Hex	Keyword	Protocol	References
56	0x38	TLSP	Transport Layer Security Protocol (using Kryptonet key management)	
57	0x39	SKIP	Simple Key-Management for Internet Protocol	RFC 2356
58	0x3A	IPv6-ICMP	ICMP for IPv6	RFC 4443, RFC 4884
59	0x3B	IPv6-NoNxt	No Next Header for IPv6	RFC 2460
60	0x3C	IPv6-Opts	Destination Options for IPv6	RFC 2460
61	0x3D		Any host internal protocol	
62	0x3E	CFTP	CFTP	
63	0x3F		Any local network	
64	0x40	SAT-EXPAK	SATNET and Backroom EXPAK	
65	0x41	KRYPTOLAN	Kryptolan	
66	0x42	RVD	MIT Remote Virtual Disk Protocol	
67	0x43	IPPC	Internet Pluribus Packet Core	
68	0x44		Any distributed file system	
69	0x45	SAT-MON	SATNET Monitoring	
70	0x46	VISA	VISA Protocol	
71	0x47	IPCU	Internet Packet Core Utility	
72	0x48	CPNX	Computer Protocol Network Executive	
73	0x49	CPHB	Computer Protocol Heart Beat	
74	0x4A	WSN	Wang Span Network	
75	0x4B	PVP	Packet Video Protocol	
76	0x4C	BR-SAT-MON	Backroom SATNET Monitoring	
77	0x4D	SUN-ND	SUN ND PROTOCOL-Temporary	
78	0x4E	WB-MON	WIDEBAND Monitoring	
79	0x4F	WB-EXPAK	WIDEBAND EXPAK	
80	0x50	ISO-IP	International Organization for Standardization Internet Protocol	
81	0x51	VMTP	Versatile Message Transaction Protocol	RFC 1045
82	0x52	SECURE-VMTP	Secure Versatile Message Transaction Protocol	RFC 1045
83	0x53	VINES	VINES	
84	0x54	TTP	TTP	
84	0x54	IPTM	Internet Protocol Traffic Manager	
85	0x55	NSFNET-IGP	NSFNET-IGP	
86	0x56	DGP	Dissimilar Gateway Protocol	
87	0x57	TCF	TCF	
88	0x58	EIGRP	EIGRP	
89	0x59	OSPF	Open Shortest Path First	RFC 1583
90	0x5A	Sprite-RPC	Sprite RPC Protocol	
91	0x5B	LARP	Locus Address Resolution Protocol	
92	0x5C	MTP	Multicast Transport Protocol	
93	0x5D	AX.25	AX.25	
94	0x5E	IPIP	IP-within-IP Encapsulation Protocol	RFC 2003
95	0x5F	MICP	Mobile Internetworking Control Protocol	
96	0x60	SCC-SP	Semaphore Communications Sec. Pro	
97	0x61	ETHERIP	Ethernet-within-IP Encapsulation	RFC 3378
98	0x62	ENCAP	Encapsulation Header	RFC 1241
99	0x63		Any private encryption scheme	
100	0x64	GMTP	GMTP	
101	0x65	IFMP	Ipsilon Flow Management Protocol	
102	0x66	PNNI	PNNI over IP	
103	0x67	PIM	Protocol Independent Multicast	
104	0x68	ARIS	IBM's ARIS (Aggregate Route IP Switching) Protocol	
105	0x69	SCPS	SCPS (Space Communications Protocol Standards)	SCPS-TP[1]
106	0x6A	QNX	QNX	
107	0x6B	A/N	Active Networks	
108	0x6C	IPComp	IP Payload Compression Protocol	RFC 3173
109	0x6D	SNP	Sitara Networks Protocol	
110	0x6E	Compaq-Peer	Compaq Peer Protocol	
111	0x6F	IPX-in-IP	IPX in IP	
112	0x70	VRRP	Virtual Router Redundancy Protocol, Common Address Redundancy Protocol (not IANA assigned)	VRRP:RFC 3768

IP Packet Fields

Decimal	Hex	Keyword	Protocol	References
113	0x71	PGM	PGM Reliable Transport Protocol	RFC 3208
114	0x72		Any 0-hop protocol	
115	0x73	L2TP	Layer Two Tunneling Protocol Version 3	RFC 3931
116	0x74	DDX	D-II Data Exchange (DDX)	
117	0x75	IATP	Interactive Agent Transfer Protocol	
118	0x76	STP	Schedule Transfer Protocol	
119	0x77	SRP	SpectraLink Radio Protocol	
120	0x78	UTI	Universal Transport Interface Protocol	
121	0x79	SMP	Simple Message Protocol	
122	0x7A	SM	Simple Multicast Protocol	draft-perlman-simple-multicast-03
123	0x7B	PTP	Performance Transparency Protocol	
124	0x7C	IS-IS over IPv4	Intermediate System to Intermediate System (IS-IS) Protocol over IPv4	RFC 1142 and RFC 1195
125	0x7D	FIRE	Flexible Intra-AS Routing Environment	
126	0x7E	CRTP	Combat Radio Transport Protocol	
127	0x7F	CRUDP	Combat Radio User Datagram	
128	0x80	SSCOPMCE	Service-Specific Connection-Oriented Protocol in a Multilink and Connectionless Environment	ITU-T Q.2111 (1999)
129	0x81	IPLT		
130	0x82	SPS	Secure Packet Shield	
131	0x83	PIPE	Private IP Encapsulation within IP	Expired I-D draft-petri-mobileip-pipe-00.txt
132	0x84	SCTP	Stream Control Transmission Protocol	
133	0x85	FC	Fibre Channel	
134	0x86	RSVP-E2E-IGNORE	Reservation Protocol (RSVP) End-to-End Ignore	RFC 3175
135	0x87	Mobility Header	Mobility Extension Header for IPv6	RFC 6275
136	0x88	UDPLite	Lightweight User Datagram Protocol	RFC 3828
137	0x89	MPLS-in-IP	Multiprotocol Label Switching Encapsulated in IP	RFC 4023
138	0x8A	manet	MANET Protocols	RFC 5498
139	0x8B	HIP	Host Identity Protocol	RFC 5201
140	0x8C	Shim6	Site Multihoming by IPv6 Intermediation	RFC 5533
141	0x8D	WESP	Wrapped Encapsulating Security Payload	RFC 5840
142	0x8E	ROHC	Robust Header Compression	RFC 5856
143-252	0x8F-0xFC	UNASSIGNED		
253-254	0xFD-0xFE	Use for experimentation and testing		RFC 3692
255	0xFF	Reserved.		

Flags field

A three-bit field follows and is used to control or identify fragments. They are (in order, from high order to low order):

Value	Description
bit 0	Reserved; must be zero.
bit 1	Don't Fragment (DF)
bit 2	More Fragments (MF)

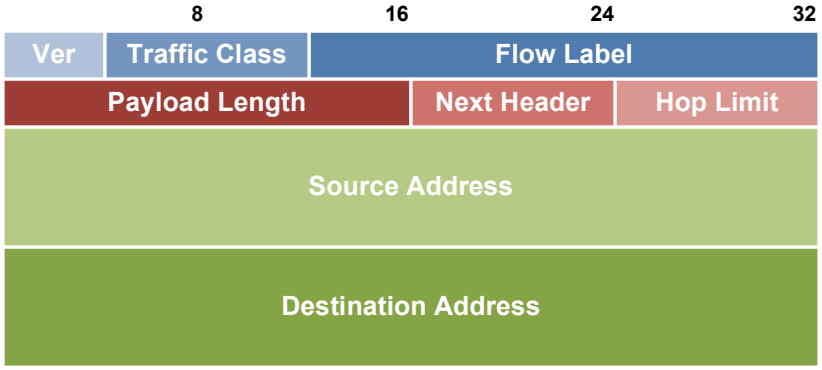
Options field

Field	Size (bits)	Description
Copied	1	Set to 1 if the options need to be copied into all fragments of a fragmented packet.
Option Class	2	A general options category. 0 is for "control" options, and 2 is for "debugging and measurement". 1, and 3 are reserved.
Option Number	5	Specifies an option.
Option Length	8	Indicates the size of the entire option (including this field). This field may not exist for simple options.
Option Data	Variable	Option-specific data. This field may not exist for simple options.

IPv6

packetlife.net

Protocol Header



Version (4 bits) · Always set to 6

Traffic Class (8 bits) · A DSCP value for QoS

Flow Label (20 bits) · Identifies unique flows (optional)

Payload Length (16 bits) · Length of the payload in bytes

Next Header (8 bits) · Header or protocol which follows

Hop Limit (8 bits) · Similar to IPv4's time to live field

Source Address (128 bits) · Source IP address

Destination Address (128 bits) · Destination IP address

Address Types

Unicast · One-to-one communication

Multicast · One-to-many communication

Anycast · An address configured in multiple locations

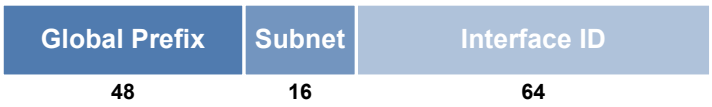
Address Notation

· Eliminate leading zeros from all two-byte sets

· Replace up to one string of consecutive zeros with a double-colon (::)

Address Formats

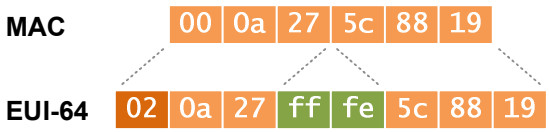
Global unicast



Link-local unicast

Multicast

EUI-64 Formation



· Insert 0xfffe between the two halves of the MAC

· Flip the seventh bit (universal/local flag) to 1

Multicast Scopes

1 Interface-local	**5** Site-local
2 Link-local	**8** Org-local
4 Admin-local	**E** Global

Special-Use Ranges

::/0	Default route
::/128	Unspecified
::1/128	Loopback
::/96	IPv4-compatible*
::FFFF:0:0/96	IPv4-mapped
2001::/32	Teredo
2001:DB8::/32	Documentation
2002::/16	6to4
FC00::/7	Unique local
FE80::/10	Link-local unicast
FEC0::/10	Site-local unicast*
FF00::/8	Multicast

* Deprecated

Extension Headers

Hop-by-hop Options (0)
Carries additional information which must be examined by every router in the path

Routing (43)
Provides source routing functionality

Fragment (44)
Included when a packet has been fragmented by its source

Encapsulating Security Payload (50)
Provides payload encryption (IPsec)

Authentication Header (51)
Provides packet authentication (IPsec)

Destination Options (60)
Carries additional information which pertains only to the recipient

Transition Mechanisms

Dual Stack
Transporting IPv4 and IPv6 across an infrastructure simultaneously

Tunneling
IPv6 traffic is encapsulated into IPv4 using IPv6-in-IP, UDP (Teredo), or Intra-Site Automatic Tunnel Addressing Protocol (ISATAP)

Translation
Stateless IP/ICMP Translation (SIIT) translates IP header fields, NAT Protocol Translation (NAT-PT) maps between IPv6 and IPv4 addresses

Decimal-Binary-Hexadecimal Conversion Chart

This chart shows all of the combinations of decimal, binary and hexadecimal from 0 to 255 decimal. When making a change in a CV this chart will show the conversion for different numbering systems. Some decoders split the CV into two parts. When you modify a CV you need to write back all 8 bits. This chart will help determine the correct bit value a CV.

Decimal	Binary	Hex	Decimal	Binary	Hex	Decimal	Binary	Hex	Decimal	Binary	Hex
Bit No.>	76543210			76543210			76543210			76543210	
0	00000000	0	64	01000000	40	128	10000000	80	192	11000000	C0
1	00000001	1	65	01000001	41	129	10000001	81	193	11000001	C1
2	00000010	2	66	01000010	42	130	10000010	82	194	11000010	C2
3	00000011	3	67	01000011	43	131	10000011	83	195	11000011	C3
4	00000100	4	68	01000100	44	132	10000100	84	196	11000100	C4
5	00000101	5	69	01000101	45	133	10000101	85	197	11000101	C5
6	00000110	6	70	01000110	46	134	10000110	86	198	11000110	C6
7	00000111	7	71	01000111	47	135	10000111	87	199	11000111	C7
8	00001000	8	72	01001000	48	136	10001000	88	200	11001000	C8
9	00001001	9	73	01001001	49	137	10001001	89	201	11001001	C9
10	00001010	A	74	01001010	4A	138	10001010	8A	202	11001010	CA
11	00001011	B	75	01001011	4B	139	10001011	8B	203	11001011	CB
12	00001100	C	76	01001100	4C	140	10001100	8C	204	11001100	CC
13	00001101	D	77	01001101	4D	141	10001101	8D	205	11001101	CD
14	00001110	E	78	01001110	4E	142	10001110	8E	206	11001110	CE
15	00001111	F	79	01001111	4F	143	10001111	8F	207	11001111	CF
16	00010000	10	80	01010000	50	144	10010000	90	208	11010000	D0
17	00010001	11	81	01010001	51	145	10010001	91	209	11010001	D1
18	00010010	12	82	01010010	52	146	10010010	92	210	11010010	D2
19	00010011	13	83	01010011	53	147	10010011	93	211	11010011	D3
20	00010100	14	84	01010100	54	148	10010100	94	212	11010100	D4
21	00010101	15	85	01010101	55	149	10010101	95	213	11010101	D5
22	00010110	16	86	01010110	56	150	10010110	96	214	11010110	D6
23	00010111	17	87	01010111	57	151	10010111	97	215	11010111	D7
24	00011000	18	88	01011000	58	152	10011000	98	216	11011000	D8
25	00011001	19	89	01011001	59	153	10011001	99	217	11011001	D9
26	00011010	1A	90	01011010	5A	154	10011010	9A	218	11011010	DA
27	00011011	1B	91	01011011	5B	155	10011011	9B	219	11011011	DB
28	00011100	1C	92	01011100	5C	156	10011100	9C	220	11011100	DC
29	00011101	1D	93	01011101	5D	157	10011101	9D	221	11011101	DD
30	00011110	1E	94	01011110	5E	158	10011110	9E	222	11011110	DE
31	00011111	1F	95	01011111	5F	159	10011111	9F	223	11011111	DF
32	00100000	20	96	01100000	60	160	10100000	A0	224	11100000	E0
33	00100001	21	97	01100001	61	161	10100001	A1	225	11100001	E1
34	00100010	22	98	01100010	62	162	10100010	A2	226	11100010	E2
35	00100011	23	99	01100011	63	163	10100011	A3	227	11100011	E3
36	00100100	24	100	01100100	64	164	10100100	A4	228	11100100	E4
37	00100101	25	101	01100101	65	165	10100101	A5	229	11100101	E5
38	00100110	26	102	01100110	66	166	10100110	A6	230	11100110	E6
39	00100111	27	103	01100111	67	167	10100111	A7	231	11100111	E7
40	00101000	28	104	01101000	68	168	10101000	A8	232	11101000	E8
41	00101001	29	105	01101001	69	169	10101001	A9	233	11101001	E9
42	00101010	2A	106	01101010	6A	170	10101010	AA	234	11101010	EA
43	00101011	2B	107	01101011	6B	171	10101011	AB	235	11101011	EB
44	00101100	2C	108	01101100	6C	172	10101100	AC	236	11101100	EC
45	00101101	2D	109	01101101	6D	173	10101101	AD	237	11101101	ED
46	00101110	2E	110	01101110	6E	174	10101110	AE	238	11101110	EE
47	00101111	2F	111	01101111	6F	175	10101111	AF	239	11101111	EF
48	00110000	30	112	01110000	70	176	10110000	B0	240	11110000	F0
49	00110001	31	113	01110001	71	177	10110001	B1	241	11110001	F1
50	00110010	32	114	01110010	72	178	10110010	B2	242	11110010	F2
51	00110011	33	115	01110011	73	179	10110011	B3	243	11110011	F3
52	00110100	34	116	01110100	74	180	10110100	B4	244	11110100	F4
53	00110101	35	117	01110101	75	181	10110101	B5	245	11110101	F5
54	00110110	36	118	01110110	76	182	10110110	B6	246	11110110	F6
55	00110111	37	119	01110111	77	183	10110111	B7	247	11110111	F7
56	00111000	38	120	01111000	78	184	10111000	B8	248	11111000	F8
57	00111001	39	121	01111001	79	185	10111001	B9	249	11111001	F9
58	00111010	3A	122	01111010	7A	186	10111010	BA	250	11111010	FA
59	00111011	3B	123	01111011	7B	187	10111011	BB	251	11111011	FB
60	00111100	3C	124	01111100	7C	188	10111100	BC	252	11111100	FC
61	00111101	3D	125	01111101	7D	189	10111101	BD	253	11111101	FD
62	00111110	3E	126	01111110	7E	190	10111110	BE	254	11111110	FE
63	00111111	3F	127	01111111	7F	191	10111111	BF	255	11111111	FF

Binary Number System for one byte

Bit Number| 7| 6| 5| 4|3|2|1|0|
Bit Weight|128|64|32|16|8|4|2|1|

Some Commonly used CVs

CV-1 Short Address
CV-2 Start Voltage
CV-3 Acceleration Rate
CV-4 Deceleration Rate
CV-5 Maximum Voltage
CV-6 Mid Point Voltage
CV-7 Ver Number
CV-8 Maker ID
CV-17/18 Long Address
CV-19 Consist Address

CV-21 Advance Consist function control
CV-22 Advance Consist headlight control
CV-23 Advance Consist acceleration rate
CV-24 Advance Consist deceleration rate

SEE YOUR DECODER MANUAL FOR ALL OF THE CVs IT USES AND THE RANGE OF VALUES.

CV-29 Configuration Register
Bit 0=Direction of travel
Bit 1=Speed step 28
Bit 2=d.c. enable
Bit 3= Advance acknowledgment
Bit 4= Alternate speed table
Bit 5= Long address.

CV-66 Forward Trim
CV-67 to 94 Speed Table
CV-95 Reverse Trim

DEF 24April02