

IPv4, IPv6, UDP, DNS

- 1) Assume you have a dial-up connection and want to send a UDP datagram of 5000 bytes to a server on the Internet. The connection between the two nodes that includes a PPP link with an MTU of 512 bytes, two Ethernet links with an MTU of 1500 bytes and an FDDI ring with an MTU of 4096 bytes. Draw a diagram of the connections, describe the fragmentation of the datagram as it is transferred to its destination and show the effect of the loss of a fragment. Contrast the behaviour of the 512-MTU-bytes dial-up link with 1500-MTU-bytes ADSL connection.

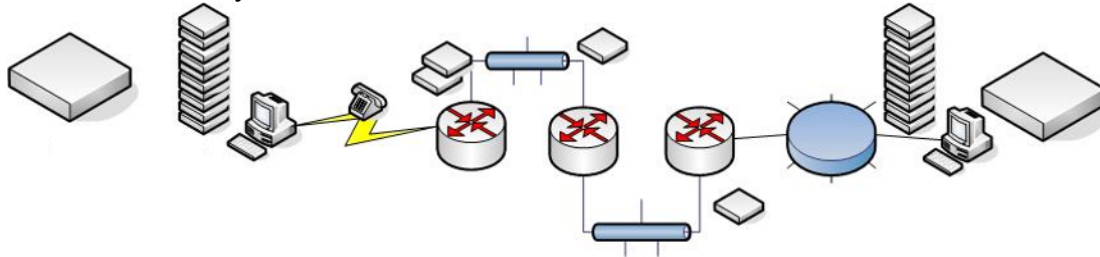


Figure 1: Fragmentation of a packet of 5000 bytes into 10 492-byte fragments and one 80-byte fragment

The original IPv4 packet with 5000 bytes payload and 20 bytes IPv4 header would be split into 10 packets each with its own header: 512 bytes in total or 20 bytes IPv4 header + 492 bytes payload; and one packet with the remaining 80 bytes payload plus 20 bytes header. These packets will be forwarded at each hop; because the MTUs of the subsequent networks are all larger than the first connection, the packets will not be split up any further. The destination node will attempt to assemble the fragments, once all fragments have been received. If one of the fragments is dropped by a router along the path, the destination will discard the received fragments once a timer has expired.

In this example, the small MTU of the first connection leads to a large number of small fragments and these fragments are then individually forwarded towards the destination, requiring individual routing decisions etc at every hop. If the MTU of the initial connection would be larger, the number of fragments would be smaller; so, fewer routing decisions etc would be required along the path. In the past, this scenario used to be common with dial-up connections where the MTUs were small; at the moment, some protocols for IoT devices exhibit similarly small MTUs and may lead to a high number of fragments at a traffic source.

- 2) Protocol Encapsulation

Draw a diagram of the individual headers i.e. UDP, IP, Ethernet header, of an Ethernet packet that includes an UDP packet addressed to an application on host 156.202.34.43 port 21 from the local application on address 134.226.34.85 port 10567. Assume values for fields of the individual headers if these values are not given above. For each value give a short explanation why you chose this particularly value.

0		8		16		24		31		0		8		16		24		31			
Preamble										10101010101010101010101010101010											
Preamble (contd)								SFD		1010101010101010								10101011			
Destination Address										00:11:93:85:E0:C3											
Dest. Address (contd)						Source Address				Dest. Address (contd)						00:11:93:85:BC:05					
Source Address (contd)										Source Address (contd)											
Length or type										0x0800											
vers.		IHL		Type		Total Length				4		20		0		36					
Identification						Flag		Fragm. Offset				0x1234						0		0	
Time-to-live				Protocol		Header Checksum				254		17		1110101110101001							
Source Address										134.226.34.85											
Destination Address										156.202.34.43											
Source Port						Destination Port				10567						21					
Length						Checksum				8						0					
Payload										2B5A YAWN											
CRC										10101011111011101010101011111010											

- Ethernet header
- IP header
- UDP header
- Payload

Notes to the values:

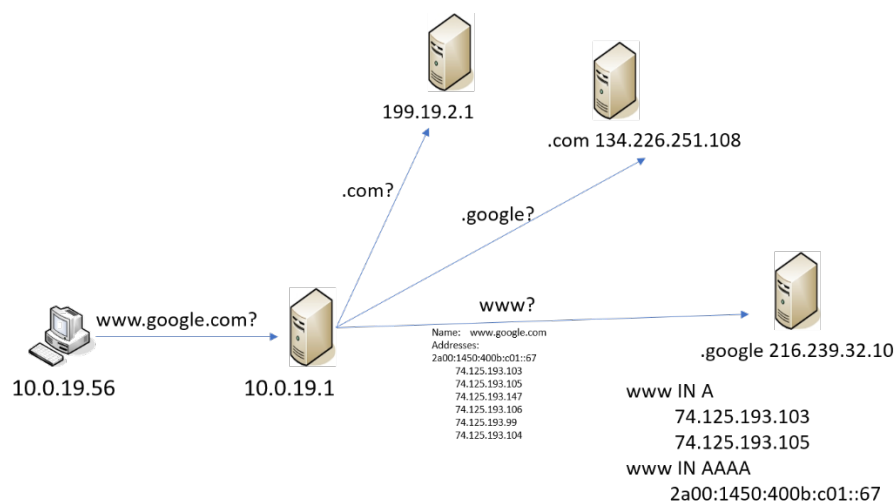
- Preamble + Start Frame Delimiter (SFD) are predefined in the Ethernet standard
 - Ethernet Source and Destination address are random 48bit addresses
 - Type field contains the value for an IP packet in the Ethernet payload
 - The CRC is a random value in this example – generally, it would be calculated over the whole Ethernet frame
-
- The version field in IPv4 is set to 4
 - The header length is 20 bytes
 - The type of the packet is not set
 - The total length of the IP packet is the sum of the IP header length (20 bytes), the UDP header (8 bytes) and the payload (8 bytes)
 - The identification of the packet is a random value in this example

- None of the flags are set
- The fragmentation offset is 0 because this packet is not a fragment of some larger packet
- The time-to-live is 254
- The protocol field contains the value for a UDP packet as payload
- The header checksum is a random value in this example – generally, it would be calculated over the IP header
- The source and destination address are taken from the question above
- The source and destination port are taken from the question above
- The length of the payload is 8 byte in this example
- The checksum is not used in this example

3) Assume that your computer located in the residences in TCD has a private IP address. Describe the communication that is involved to resolve the name of the server when you type the URL “www.google.com” into a web browser on your machine?

The output of a name resolution program like nslookup may look like the following:

```
> nslookup www.google.com
Server: UnKnown
Address: 10.0.19.1
Non-authoritative answer:
Name:   www.google.com
Addresses: 2a00:1450:400b:c01::67
          74.125.193.103
          74.125.193.99
```



4) Discuss how a dual stack implementation handles an Ethernet frame which carries either an IPv4 or an IPv6 packet and write out the details of the Ethernet and IP header in as much detail as possible. The Ethernet type IPv4 payload is 0x0800 and for IPv6 payload 0x86dd; assume all other details as you see fit.

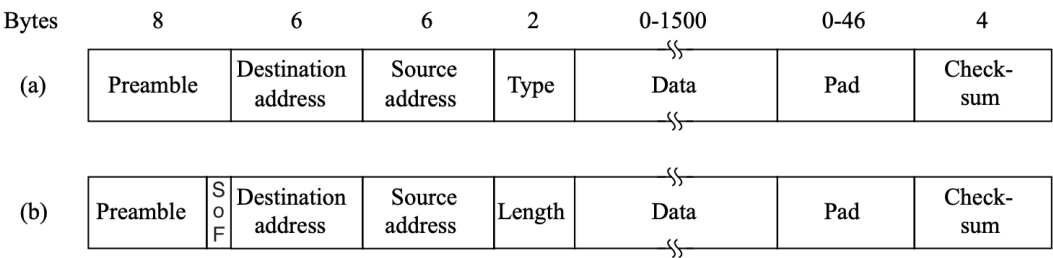


Figure 4-14. Frame formats. (a) Etherne

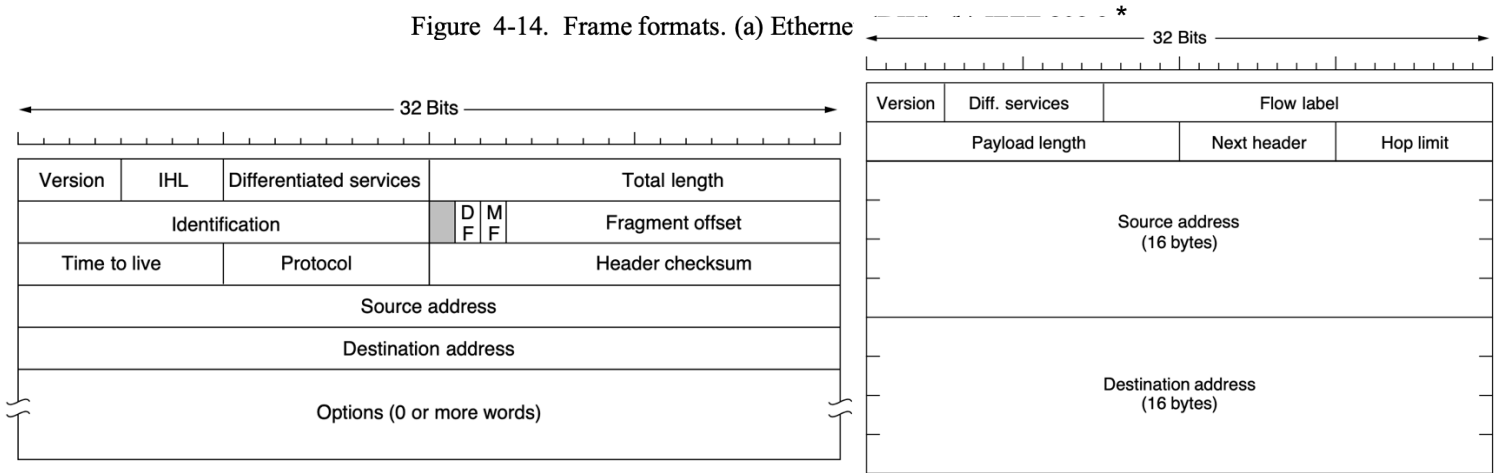
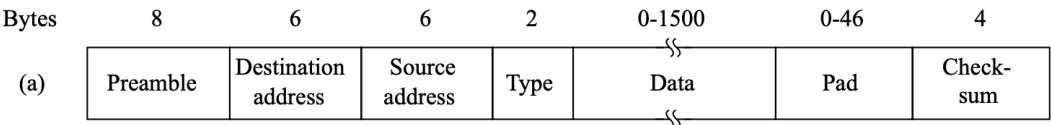


Figure 5-47. The IPv4 (Internet Protocol version 4) header.

Figure 5-57. The IPv6 fixed header (required).

*Note from Computer Networks: "... Unfortunately, by the time 802.3 was published, so much hardware and software for DIX Ethernet was already in use that few manufacturers and users were enthusiastic about repackaging the Type and Length fields. In 1997, IEEE threw in the towel and said that both ways were fine with it. Fortunately, all the Type fields in use before 1997 had values greater than 1500, then well established as the maximum data size. Now the rule is that any number there less than or equal to 0x600 (1536) can be interpreted as Length, and any number greater than 0x600 can be interpreted as Type."

On the sending side, the firmware of the Ethernet adapter implanting the link layer would create an Ethernet frame that indicates that it is carrying either an IPv4 or IPv6 packet by filling the type field with 0x0800 or x86dd hexadecimal and then transmitting the frame to the next hop indicated in the destination address. The receiving node will inspect the destination address and CRC and then hand the payload of the Ethernet frame to the corresponding IP implementation, depending on the type field. If the node is intermediate hop e.g. a router, it will inspect the destination address of the IP packet and send the IP packet in new Ethernet frame to the next hop. The size of the IPv4 header without options will be 20 bytes and the size of the IPv6 header will be 40 bytes.

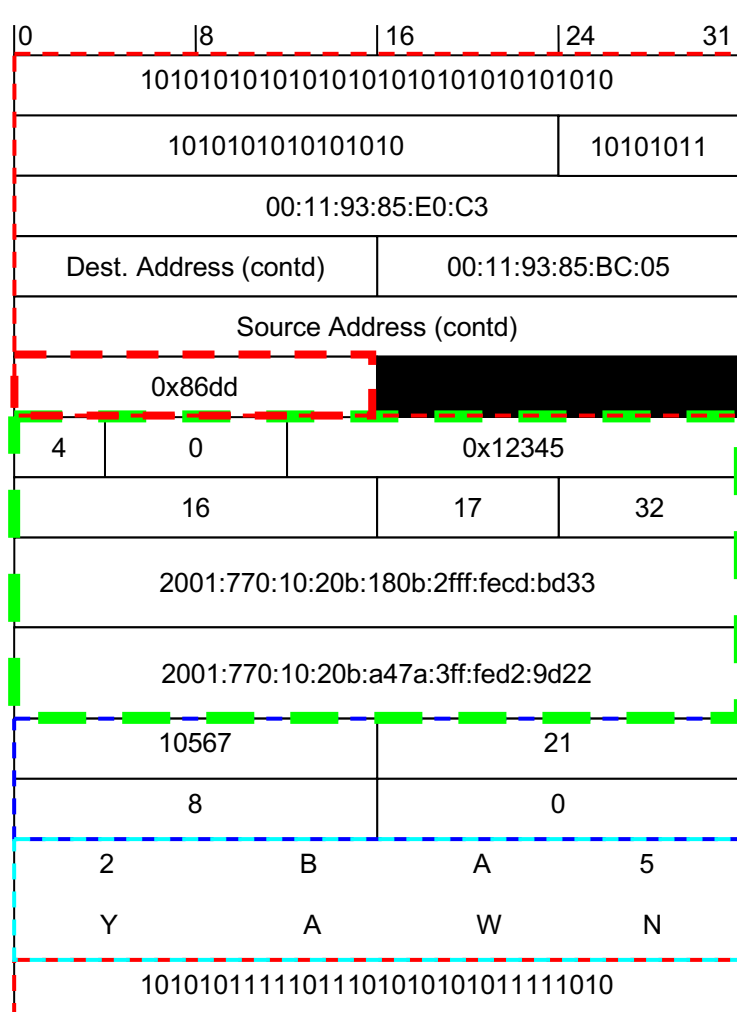
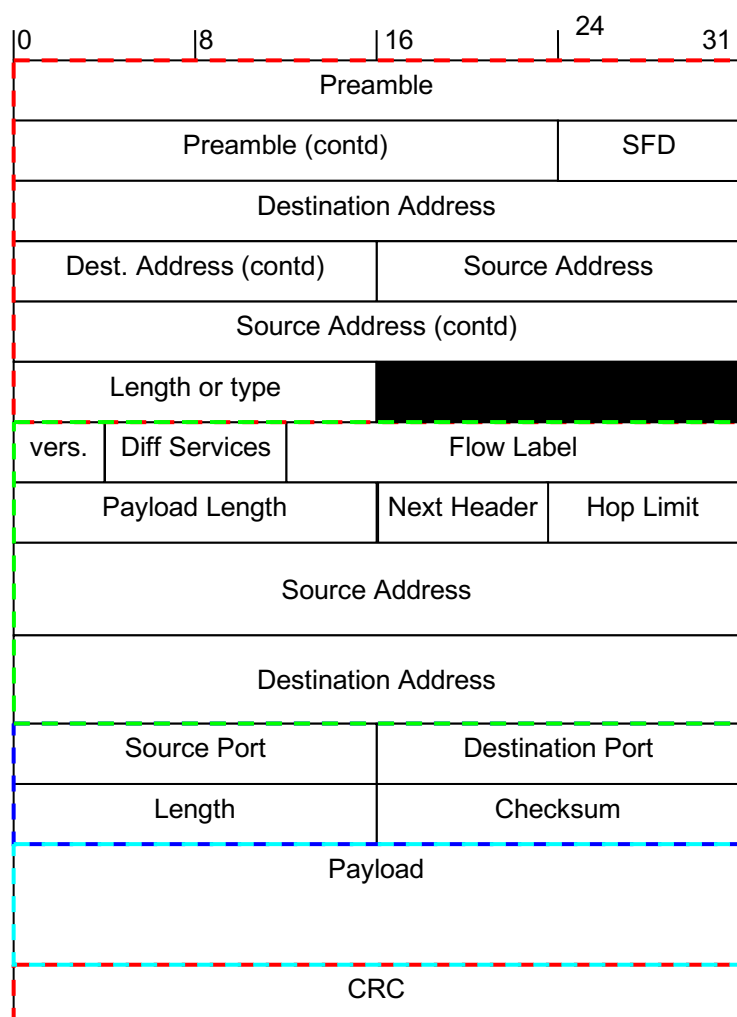


0				8				16				24				31				0				8				16				24				31			
Preamble																10101010101010101010101010101010																							
Preamble (contd)												SFD				1010101010101010												10101011											
Destination Address																00:11:93:85:E0:C3																							
Dest. Address (contd)								Source Address								Dest. Address (contd)								00:11:93:85:BC:05															
Source Address (contd)																Source Address (contd)																							
Length or type																0x0800																							
vers.		IHL		Type		Total Length						4		20		0		36																					
Identification						Flag		Fragm. Offset						0x1234						0		0																	
Time-to-live				Protocol				Header Checksum								254				17				1110101110101001															
Source Address																134.226.34.85																							
Destination Address																156.202.34.43																							
Source Port								Destination Port								10567								21															
Length								Checksum								8								0															
Payload																2BAY																							
CRC																10101011111011101010101011111010																							

- Ethernet header
- IPv4 header
- UDP header
- Payload

Notes to the values of the IPv6 example:

- Type field of the Ethernet is set to 0x86dd (IPv6 Payload)
- The version field of the IP packet is set to 6
- The differential services is set to 0
- The flow label – a sequence number maintained by the sender – is a random value in this example
- The payload length is 16 bytes UDP header (8 bytes) and the payload (8 bytes)
- The next header field is set to 17 to indicate a UDP datagram as payload
- The hop limit is set to 32
- The 128-bit source and destination IP address are from two servers in the School of Computer Science and Statistics



- Ethernet header
- IPv6 header
- UDP header
- Payload

Past Question: Discuss the two proposed formats of addresses for IPv6 and contrast them with the adapted format of addresses for IPv6:

During the development of IPv6, two particular formats were proposed: One with fixed 64-bit addresses and one with flexible addresses with up to 160 bits. The advantage of the fixed addresses of 64 bits is that hardware could be developed to analyse addresses for routing. The disadvantage of fixed 64-bit addresses would be that the address space may become overpopulated again and lead to an address shortage in future. Flexible addresses would have the advantage that they could be adapted to future requirements and situations with a variety of requirements; however, the interpretation of flexible addresses for routing may be more time intensive and slower. The 128-bit address format of IPv6 has the advantage that it is a fixed address size, avoiding some of the effort that is necessary to interpret flexible addresses and it offers 64 bit more address space than the proposed 64-bit address space. The disadvantage of this approach may be that a 128-bit address is too large to be exploited efficiently and leads to avoidable overhead. One could argue that a 64-bit address space for a network ID would have been sufficient and that a flexible-size host ID would offer the option to adapt host IDs to various requirements.