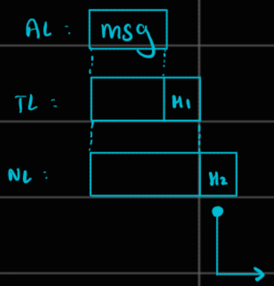
IPv4

- IPv4 header :



Version (4 bits)	header length (4 bits)	Services (8 bits)	Total length (16 bits)
Identification no (16 bits)	Flags (3 bits)	Fragment offset (13 bits)	
Time to live (8 bits)	Protocol (8 bits)	Header checksum (16 bits)	
Source IP address (32 bits)			
Destination IP address (32 bits)			
☆ Option (0 to 40 Bytes)			

← 32 bits or 4 Bytes →

☆ 2 bytes are reserved in options. So, only 38 bytes are available

- header length (HL) : (4 bits)
  - Size of header ÷ 4 always. header size is ALWAYS a multiple of 4.
  - For eg. if header size = 20B ⇒ HL = 20/4 = 5 = 0101

    if header size = 32B ⇒ HL = 32/4 = 8 = 1000

    if header size = 60B ⇒ HL = 60/4 = 15 = 1111

    if header size = 30B ⇒ Pad it by 2B

    ⇒ header size = 32B ⇒ HL = 32/4 = 8 = 1000

  [padding = 2B is represented as 0010 in the padding field. NOTE that is the no. of Bytes that we are encoding using bits]

- Services : (8 bits)
  - In this interpretation, first 3 bits are called PRIORITY (or precedence) bits, the next 4 bits are called type of service (TOS) bit and the last bit is unused

P	P	P	D	T	R	C

← Priority bits → ← TOS bits →

D → Min. delay ; T → max. throughput

R → max. reliability ; C → Min. cost

- **Priority bits** (3 bits) :
  - •1 Is a 3-bit subfield ranging from 0 to 7
  - •2 Is needed if a router is CONGESTED and it needs to discard some datagram. Datagrams with lower priority are discarded

- **Type of service bits** (4 bits) :
  - •1 Is a 4-bit subfield where EACH bit has a special meaning
  - •2 One and only one of the bits can have the value 1 in each datagram. So, only one out of D, T, R and C can be chosen at a time.

– **Total length** (16 bits) :
- Total length = Size of data + Size of header
- max total length = $2^{16}-1$ = 65,535 BYTES. So, max. data length = max. total length – min. header size = 65,535 – 20 = 65,515 BYTES

– **Identification no.** (16 bits) :
- It is used to identify all the fragments of a datagram
- All the fragment of same datagram will have the same identification no.
- Also known as datagram no.

– **Flag** (3 bits) :
- It is a 3-bit field :

X	DF	MF

MF → More fragment

X ↓ Not used

DF → Don't fragment

DF :
- If DF = 1, datagram CAN NOT be fragmented else it can be fragmented. If the packet size > maximum transmission unit (MTU) and if DF = 1, router DISCARDS the packet and sends a ICMP error saying packet size is greater than MTU. Sender then adjusts the packet size and re-sends it. Same thing happens when there are multiple routers. So, sender needs to ensure the packet size ≤ min (MTU) for all links in the path if it sends DF = 1

[NOTE : While fragmenting a packet, we send the header to all the fragments because it is not guaranteed that all of them would take the same path]

- Why would sender even set DF = 1 ? Answer : Efficiency. For eg. say A wants to send 300 bytes of data to B via a router (MTU = 100 bytes). If we set DF = 1, then

$$\eta_{DF=1} = \frac{\text{Useful bytes}}{\text{Total bytes}} = \frac{300}{320} = 93.75\%$$ (20 bytes header)

but if DF = 0, then (assuming the packet is divided into 3 fragments)

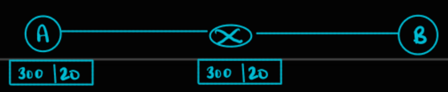
$$\eta_{DF=0} = \frac{\text{Useful bytes}}{\text{Total bytes}} = \frac{300}{3*(100+20)} = \frac{300}{360} = 83.3\%$$

(header is ALWAYS an overhead)

MF :
- If MF = 1, it means " this is not the last fragment, more fragments are yet to come "
- If MF = 0, it means it is the last fragment

- Fragment Offset : (13 bits)
  - Indicate no. of data byte AHEAD of this fragment in that particular packet
  - Just does INDEXING since the packets can be received out of order.

Eg.



(MTU = 100 bytes)

100 | 20	100 | 20	100 | 20	
100	100	100	ID NO.
0	1	1	MF
200	100	0	Fragment offset

NOTE :

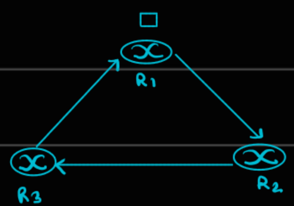
- IP is a packet stream protocol i.e. every packet is associated with a seq. no.
- TCP is a byte stream protocol i.e. every byte is associated with a seq. no.

- **Time to live (TTL)** : (8 bits)
  - Is used to avoid infinite looping. It is used to control the max. no. of hops visited by datagram
  - When a source host sends a datagram, it stores a no. in this field. Each router that processes the datagram decrements this no. by one. If TTL reaches 0 before the datagram arrives at its destination, then the datagram is discarded and an ICMP message is sent back to the sender

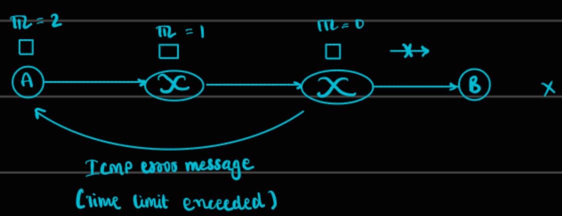
TTL can NEVER be 0. So, range of TTL is [1, 255]

- Since it is a 8-bit field, only 256 routers can be present b/w 2 hosts A and B. Today, even inter-continental traffic requires only 10 to 30 routers so this hard limit is not an issue. And even if it becomes one, we can use "tunneling protocols" like GRE (Generic Routing Encapsulation) or VPNs to encapsulate packets in such a way that it "resets" the TTL field

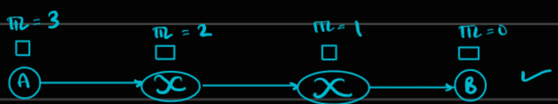
- **Infinite looping case :** When the destination IP's NID (network ID) does not match any interfaces in the router's routing table, the router forwards the packet to the default interface (a default router). Now, say default router of R_1 is R_2, that of R_2 is R_3 and R_3 is R_1 and the above mentioned case occurs on all 3 routers then we can get an infinite loop

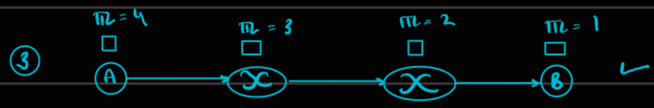


Eg. ①



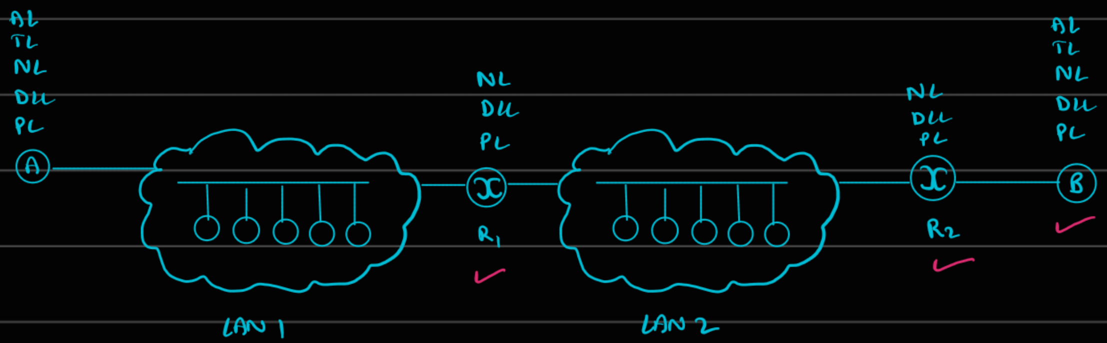
②





NOTE :
- Any device that has a network layer CAN DECREMENT TTL.
- Destination host CAN DECREMENT TTL

Also, note in this example :



Only routers R_1, R_2 and destination host B CAN DECREMENT TTL. NONE of the hosts INSIDE the LAN can decrement TTL.

- **Protocol** (8 bits) :
  - Tells us which protocol is encapsulated in the IP packet
  - At time of traffic, some packets must be discarded. In this case it will be helpful to know which protocol data it contains. The order in which the router will eliminate the datagram from buffer is :

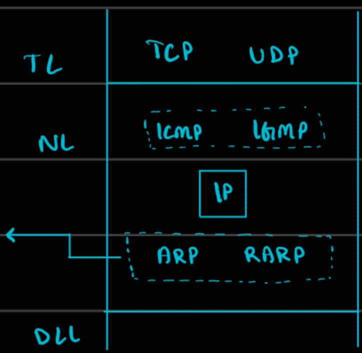
Protocol no. 01 02 17 06

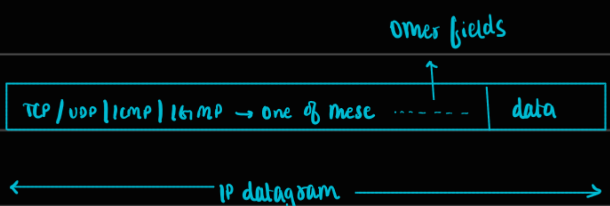
(First to be eliminated) $\underbrace{\text{ICMP} > \text{IGMP} > \text{UDP}}_{\text{Connectionless. So, it is okay to discard}} > \underbrace{\text{TCP}}_{\text{Requires connection. So, it is not okay to discard}}$ (Last to be eliminated)

NOTE :

Above IP : ICMP and IGMP used by TL protocols

Below IP : These directly use NL protocols without the services of TL. Usually used for routing packets





NOTE : • OSFP : Open shortest path first : does not require TCP/UDP. Uses only the NL since it is a routing protocol. Has a protocol no. of 89

- **Header Checksum** (16 bits) :
  - It is calculated ONLY for the header part since rest of the component is already covered by TCP Checksum
  - Is calculated at each and every router because few fields in header might change (eg. TTL, fragment offset, MF, total length, etc)

- **Source Address** (32 bits) :
  - IP of source. Remains unchanged

- **Destination Address** (32 bits) :
  - IP of destination. Remains unchanged

- **Options** (max. 40 Bytes) :
  - First 16 bits (2 bytes) are reserved for option type (8 bits) and length (8 bits). So, only 38 bytes are available
  - There are 5 options :

  - **Strict Source Routing** :
    - Route is decided by the source. It is used by the source to pre-determine a route for a datagram as it travels through the internet

    Eg. $R_4 \to R_3 \to R_2 \to R_1$

    ✓ $R_4 \to R_3 \to R_2 \to R_1$

    X $R_4 \cdot \ \ \to R_5 \to R_2 \to R_1$

  - **Loose Source Routing** :
    - Similar to strict source routing but here the datagram can visit the other routers as well
    - It does need to go through every router in the list though

    Eg. $R_4 \to R_3 \to R_2 \to R_1$

    ✓ $R_4 \to R_3 \to R_5 \to R_2 \to R_1$

    ✓ $R_4 \to R \quad R_3 \to R_2 \to R_1$

- Record Routing :
  - Route is set by the router
  - It is an option used to record internet routers that handle the datagram. It can list upto 9 router addresses NOT 10
  - All the routers are supposed to record their IP address on their IP packets

Since IP is 32 bits (4 bytes) and space available is 38 bytes ⟵

- Timestamp :
  - It is used to find out delays at each router.
  - Every router should record incoming and outgoing time

- Padding :
  - Used to make the header length a multiple of 4
  - Examples discussed already (Check Pg. 1)

Eg. ① IPv4 packet ; HLEN = $(1100)_2$; Bytes of options = ?

HLEN = $(12)_{10}$ ⟹ Total length = $12 * 4 = 48$ = Fixed header length + Options

Options = $48 - 20 = 28$ bytes

② IPv4 packet : $(45000052000030000590 6 \ldots)_{16}$. How many hops can this packet take before being dropped ?

HL TL

$(4500005 2000 30000 \; 59 \; 06 \ldots)_{16}$

Version, Services, TL, ID. NO, Flags + Frag. offset, Time to live

$= (59)_{16} = 16^1 \times 5 + 16^0 \times 9$

$= 89$

⟹ No. of hops = 89

③ IPv4 packet : $(45000052000030000590 6 \ldots)_{16}$. How many hops can this packet take before being dropped ?

1st row 2nd row 3rd row

(45000056 | 00030000 | 5906 ....)$_{16}$

TTL Protocol = $(06)_{16}$ = 16 = TCP

Fragmentation in IPv4

- Points to note:

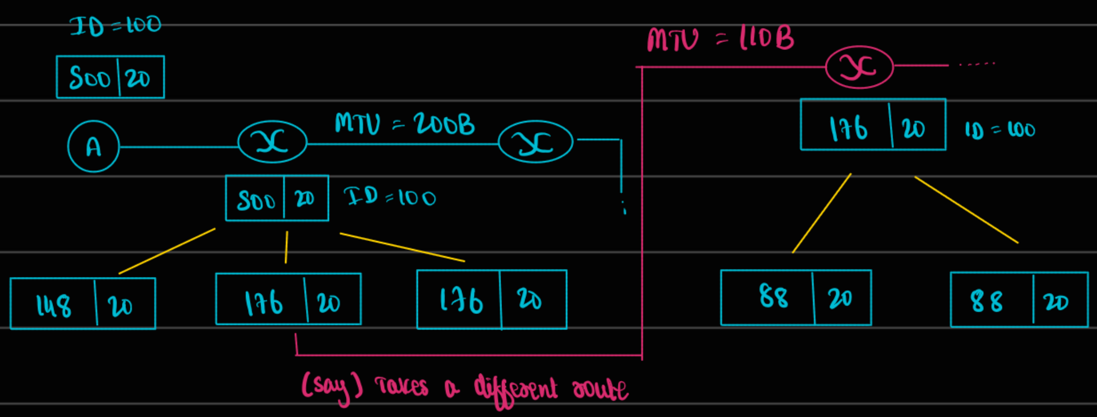
  - While splitting the packets, only data is split not the header
  - Size of data in each split packet must be divisible by 8. Reason:

Max. no. of bytes of data that can be transmitted in 1 packet $= 65,535 - 20 = 65,515$ bytes $\sim 2^{16}$

Max. no. of fragmentation $= 2^{13} - 1 = 8191 \sim 2^{13}$

So, we should use a scaling factor of $2^{16}/2^{13} = 8$ (like we did for HLEN)

Eg.



ID	100	100	100	100	100
MF	0	1	1	1	1
HLEN	5	5	5	5	5
Total length	168	196	196	108	108

Not 20 ← (HLEN) due to ÷4

offset $176+176/8 = 44$ $176/8 = 22$ 0 [1] $176+88/8 = 33$ $176/8 = 22$ [2]

•1 → This is the 1st fragment

•2 Packet $\boxed{176 \mid 20}$ is still in front of it (although it is taking a different route)

The final CAN HAVE OUT of order packets (makes sense since the splitted packets may take different routes). Issue is, how will we identify them?

Idea 1: Since offset sort of does indexing, why not use that?

This FAILS because we can identify whether the first (offset = 0) or the last (MF = 0) packet is lost BUT not if any other packet is lost

The solution is an improved version of this idea called the Reassemble Algorithm.

- **Reassamble Algorithm:**

•1 Identify the 1st fragment by using the fact : offset = 0

•2 Identify the last fragment by using the fact : MF = 0

•3 Divide the DATA LENGTH (so, no headers) by 8. The 2nd fragment has an offset of that value.

•4 Divide the DATA LENGTH of 1st and 2nd by 8 and sum it. The 3rd fragment has an offset of that value

•3 Repeat the above process until you get all the middle fragments

$$\boxed{\text{offset}_n = \sum_{j=1}^{n-1} (\text{data length of fragment}_j) / 8}$$

Eg. ① Host A sends a UDP datagram containing 8880 bytes of data to host B over an Ethernet LAN (MTU = 1500 bytes). Size of UDP header = 8 bytes and Size of IP header = 20 bytes. There is no option field in the IP header. How many total no. of IP fragments will be transmitted and what will be the content of the offset field in the last fragment ?

A) 6 and 925 C) 7 and 1110

B) 6 and 7400 D) 7 and 8880

$\Rightarrow$ Data length = 8888

$\lceil 8888/1480 \rceil = 6$. But we also need a fragment to put the remainder (8) as well. So, total no. of fragments = 6+1 = 7

MTU = 1500 and header size = 20

last fragment (7th): | 8 | 20 | .... 2nd fragment: | 1480 | 20 | 1st fragment: | 1480 | 20 |

So, offset of last fragment = $\frac{1480 \times 6}{8} = 1110$

∴ Answer is C)

② In an IPv4 datagram, HLEN = 5 and total length = 200. What is the position of datagram ?

Header size = HLEN × 4 = 20 ⇒ Data length = 200 − 20 = 180 ⟶ NOT divisible by 8

⇒ This datagram is the LAST fragment

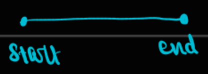
③ IPv4 datagram ; M bit is 0 ; HLEN = 10 ; Total length = 400 ; fragment offset value = 300 ; Sequence nos. of 1^{st} and last bytes of the payload are ?

Header size = $4 \times HLEN = 40 \Rightarrow$ Data length = $400 - 40 = 360$

Offset value = $300 \Rightarrow$ No. of bytes ahead of this fragment $= 300 \times 8 = 2400$

We sorta don't need to do that since the datagram is the last fragment and to find the 1^{st} datagram, all we need to do is add the offset (and -1)

So, first byte is 2400 and last byte is $2400 + 360 - 1 = 2759$



: no. of bytes b/w start and end = (end − start + 1)

Same logic for the (−1) term above