GATE CSE NOTES

by

Joyoshish Saha

ISO/OSI Model.

* An internetwork is a collection of individual networks, connected by intermediate networking devices, that functions as a single large network.

* Functions for internetwork -

a) Mandatory : Error control, flow control, access control, multiplexing & demultiplexing, addressing, etc.

b) Optional : Encryption & decryption, check-pointing, routing & so on.

* To implement all the above functionalities there are various reference models which classify all the above functionalities & define what functions are carried out at a particular layer.

✓ Different models -

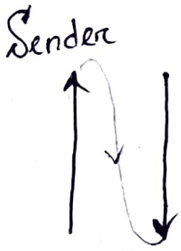
1. ISO-OSI
2. TCP/IP
3. ATM (Asynchronous transfer mode)
4. IEEE (Deals with LAN technologies)
5. X.25

* Open System Interconnection (OSI) Reference Model

Developed by ISO (International organisation of Standardization) in 1984.

It describes how information from a software application in one computer moves through a network medium to a S/W application in another computer. It's a conceptual model composed of 7 layers, each specifying particular N/W functions. It is now considered the primary architectural model for internetworking. The OSI model divides the tasks involved with moving information between networked computers into seven smaller, more manageable task groups. A task or group of tasks is then assigned to each of the seven OSI layer.

→ Layers as defined by the standard in the increasing order of functional complexity:



1. Physical layer (H/W) } H/W layers
2. Data link layer (H/W & S/W) }
3. Network layer (complex) }
4. Transport layer (Heart of OSI) Thick layer.
5. Session layer } S/W layers.
6. Presentation layer } User interaction.
7. Application layer }

TCP/IP layers
Application
Transport
Internet
Link

→ Advantages of layering:

i) divide & conquer
ii) Encapsulation is possible
iii) Abstraction
iv) Testing made easy

refer RFC (request for comments)

→ Characteristics :

Seven layers can be divided into 2 categories : upper & lower layers.

The upper layers of the model deal with application issues & generally are implemented only in software. The highest layer, the application layer, is closest to the end user. Both users & application layer processes interact with software applications that contain a communications component.

The lower layers handle data transport issues. The physical layer & the data link layer are implemented in H/W & S/W. The physical layer is closest to the physical network medium & is responsible for actually placing information on the medium.

Upper layers	Lower layers
Ap., Pr., Ses.	Tr., Ne., Da., Ph.

→ Protocols

The OSI model provides a conceptual framework for communication between computers, but the model itself is not a method of communication. Actual communication is made possible by using communication protocols. In the context of data networking, a protocol is a formal set of rules & conventions that governs

how computers exchange information over a N/W medium. A protocol implements the functions of one or more of the OSI layers.

Some communication protocols -

a) LAN protocols (physical & DL layer)

b) WAN protocols (lowest 3 layers)

c) Routing protocols (operate at N/W layer)

→ OSI Model & Communication b/w systems.

A given layer in the model generally communicates with 3 other OSI layers - the layer directly above it, below it & its peer layer in other networked computer systems. If system A has information to send to system B, for example, the DL layer of system A communicates with the N/W layer of A, physical layer of A and DL layer of B.

One OSI layer communicates with another layer to make use of the services provided by the 2nd layer. The services provided by adjacent layers help a given OSI layer communicate with its peer layer in other communication systems. Three basic elements are involved in layer services - the service user, the service provider & the service access point (SAP).

The service user is the layer that requests services from an adjacent layer. The service provider is the layer that provides services. OSI layers can provide services to multiple service users. The SAP is a conceptual location at which one OSI layer can request the services of another OSI layer.

Information exchange: Seven OSI layers use various forms of control information to communicate with their peer layers in other computer systems. This control information consists of specific requests & instructions that are exchanged between peer OSI layers. Control information typically takes one of two forms: headers and trailers. Headers are prepended to data that has been passed down from upper layers. Trailers are appended to data that has been passed down from upper layers.

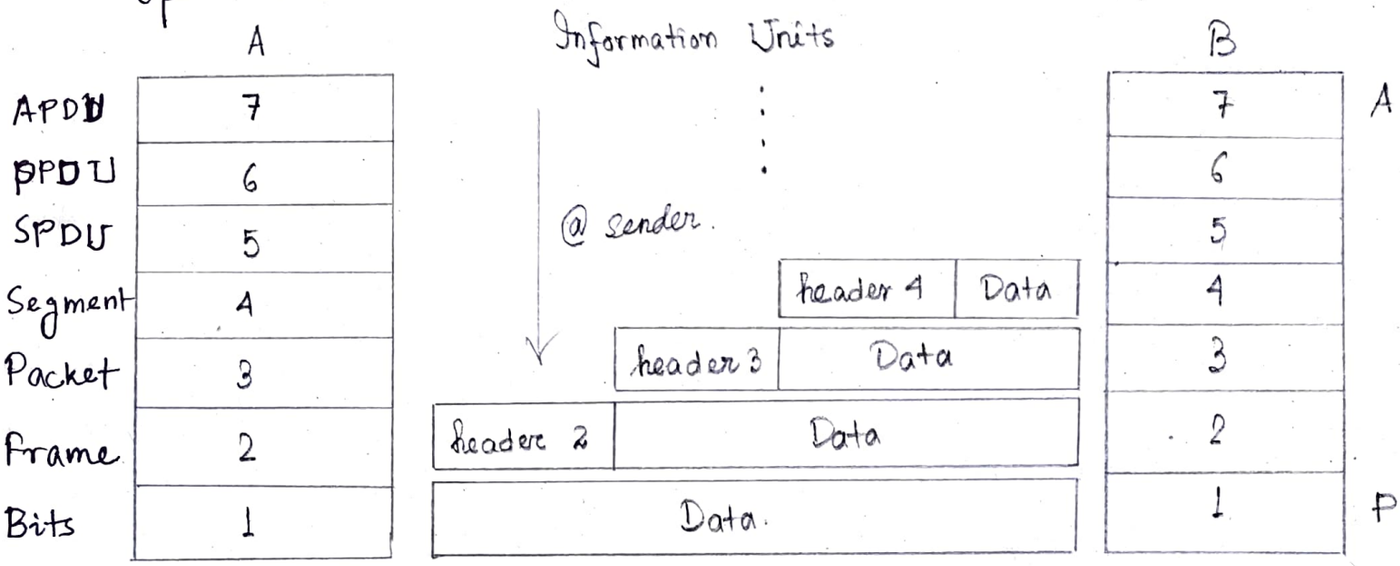
For ~~An~~ an OSI layer it is not mandatory ~~required~~ to attach a header or a trailer to data from upper layers.

Headers, trailers & data are relative concepts, depending on the layer that analyzes the information unit. At the N/W layer, for example, an information unit consists of a layer 3 header & data. At the data link layer, however, all the information passed down by the N/W layer (layer 3 header & data) is treated as

data.

The data portion of an information unit at a given OSI layer potentially can contain headers, trailers & data from all the higher layers. This is known as encapsulation.

Figure shows how the header & data from one layer are encapsulated into the header of the next lowest layer.



Information exchange: The information exchange process occurs between peer OSI layers. Each layer in the system A (source system) adds control information to data, & each layer in the system B (destination) analyzes & removes the control information from that data.

If system A has data from a S/W application to send to system B, the data is passed to the application layer. The Ap layer in A then communicates any control information required by the Ap layer in system B by prepending a Header to the data. The resulting information unit is passed to the presentation layer, which prepends its

own header containing control information intended for the presentation layer in system B. The information unit grows in size as each layer prepends its own header (in some cases a trailer) that contains control information to be used by its peer layer in system B. At the ph. layer, the entire information unit is placed onto the N/W medium.

The ph. layer in B receives the information unit & passes it to the data link layer. The data link layer in B then reads the control information contained in the header prepended by the data link layer in A. The header is then removed, & the remainder is passed to the N/W layer. Each layer performs the same actions: the layer reads the header from its peer layer, strips it off & passes the remaining information unit to the next highest layer. After the app. layer performs these actions, the data is passed to the recipient software application in B, in exactly the form in which it ~~was~~ was transmitted by the application in A.

- Encapsulation: A packet (header & data) at level 7 is encapsulated in a packet at level 6. The whole packet at level 6 is encapsulated in a packet at level 5 & so on.

The data portion of a packet at level N-1 carries the whole packet (data & header) from level N. This is called encapsulation. Level N-1 is not aware of which part of the encapsulated packet is data & which part is the header or trailer. For level N-1, the whole packet coming from level N is treated as one integrat unit.

- Physical layer.

→ The physical layer defines the electrical, mechanical, procedural & functional specifications for activating, maintaining & deactivating the physical link between communicating network systems. Physical layer specifications define characteristics such as voltage levels, timing of voltage changes, physical data rates, maximum transmission distances & physical connectors. Physical layer implementations can be categorised as either LAN or WAN specifications.

→ Responsible for the actual physical connection between the devices. The physical layer contains information in the form of bits. It is responsible for transmitting individual bits from one node to other.

While receiving data, this layer will get the signal received & convert it into 0s & 1s & send them to the data link layer, which will put the frame back together.
To be transmitted bits must be encoded into electrical or optical signals.

→ Physical layer is concerned with:

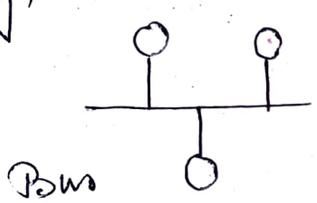
i) Hardware specification: Details of the physical cables, network interface cards, wireless radios etc.

ii) Encoding & Signalling: How are the bits encoded in the medium is decided by this layer. For example, on the copper wire medium, we can use different voltage levels for a certain time interval to represent '0' & '1'. We may use +5mV for 1 nsec to represent '1' & -5 mV for 1 nsec to represent '0'. All the issues with modulation is dealt here, eg. we may use BPSK for representations of 1 & 0 rather than using different voltage levels if we have to transfere in RF waves.

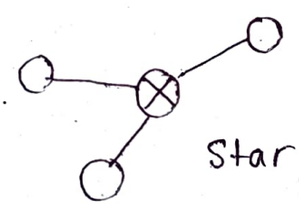
iii) Data transmission & reception: Transfer of each bit of data is the responsibility of this layer. This layer assures the transmission of each bit with a high poob-ability. Transmission of the bits is not completely

reliable as ~~their~~ there is no error correction on this layer.

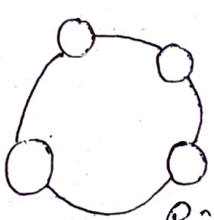
iv) Topology of Network design : Which part of the N/W is the router going to be placed, where the switches will be used, where we will put the hubs, how many machines is each switch going to handle, what server is going to be placed where, & many such concerns are to be taken care of by the layer. Various kinds of network topologies that we use are ring, bus, star, hybrid.



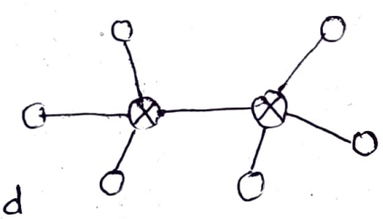
Bus



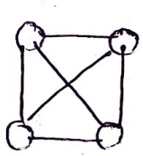
Star



Ring



Hybrid



Mesh

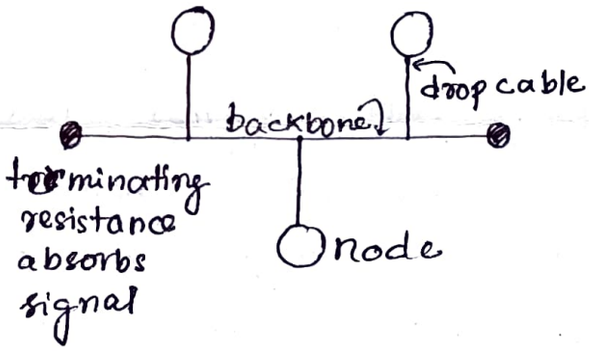
v) Bit synchronisation : By providing a clock. The clock controls both sender & receiver thus providing synchronisation at bit level.

vi) Bit rate control : Defines the transmission rate (# of bits sent per second).

vii) Transmission mode : Simplex, half-duplex or full-duplex.

* Network Topologies.

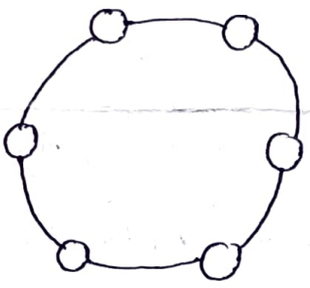
1. Bus topology: All stations are connected through a single cable known as backbone cable. Each node is either connected to the backbone by drop cable or directly connected to the backbone. When a node wants to send a message over the N/W it puts a message over the N/W. All the stations available in the N/W will receive the message whether it has been addressed or not. Mainly used in 802.3 (Ethernet) & 802.4 standard N/Ws. Through backbone msg is broadcasted to all stations. Most common access method for bus topology is CSMA (CSMA/CD & CSMA/CA).



Adv.: i) Low cost cable (as no hubs), ii) moderate data speed (upto 10 Mbps with coaxial or twisted pair), iii) familiar technology, iv) broadcasting, multicasting much simpler, v) N/W is redundant in the sense that failure of one node does not effect the whole N/W. vi) good for smaller N/Ws not requiring very high speed.

Disadv.: i) Extensive cabling, ii) difficult troubleshooting - if any fault occurs in the cable, it would disrupt the comm^n for all nodes, iii) signal interference, iv) adding new devices to the N/W would slow down the N/W. v) attenuation - loss of signal in long distance

2. Ring topology: All nodes are connected in a closed circuit of cable (circular). Messages that are transmitted travel around the ring until they reach the computer that they are addressed to, the signal being refreshed by each node. Transmission is mainly unidirectional, but it can be made bidirectional by having 2 connections between every N/W nodes pair (or by having another ring with the oppositely directed transmission (Dual ring topology). Data transferred bit by bit in a slow orderly fashion. Every node gets a chance to send a packet & it is guaranteed that every node gets to send a packet in a finite amount of time.

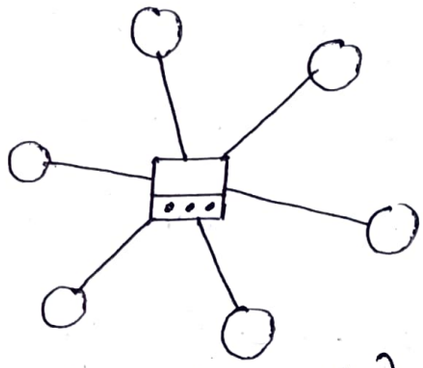


Most common access method for ring topology is token passing.

Adv.: i) N/W mgmt. - faulty devices removed without bringing N/W down. ii) product availaibility - many h/w s/w tools available for operation & monitoring, iii) low installation cost., iv) broadcasting - multicasting is simple, v) very orderly N/W where every device gets chance to transmit & it performs better than a star N/W under heavy load.

Disadv.: i) Breakdown in one station leads to the failure of overall N/W. ii) adding new devices increases the communication delay. iii) Changes of devices can effect N/W. iv) Slower than star topology under normal load.

3. Star topology: Every node connected to a central hub, switch or central computer. Coaxial cable or RJ-45 cables are used to connect these computers. Signals are transmitted & received through the hub. It is the simplest & oldest & all telephone switches are based on this. There exists P2P connection between hosts & hub (central hub). Hub acts as a single point of failure.

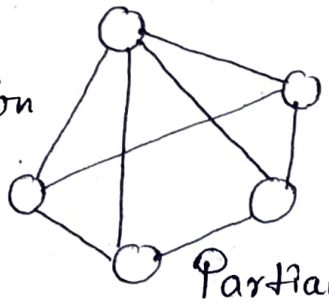


Adv.: i) Efficient troubleshooting (as all are connected to hub, one has to go to the single station to troubleshoot unlike bus topology where we have to check kms of cable), ii) Complex N/W control features can be easily implemented iii) Limited failure - failure in one node does not affect whole N/W., iv) familiar technology, v) easily expandable, vi) cost-effective, vii) high data speed - Ethernet 100BaseT is one of the most popular star N/Ws.

Disadv.: i) Central point of failure, ii) broadcasting & multicasting not easy as extra functionality needed in central hub, iii) installation cost high - each node needs to be connected to central hub.

4. Mesh topology: Computers are connected with each other through various redundant connections. This topology has hosts in P2P with every other host or may also have hosts which are in P2P to few hosts only. → Full mesh - for every

new host $\frac{n(n-1)}{2}$ connections are reqd. Most reliable N/W structure. → Partially mesh - Where we need to provide reliability to selected nodes only. Mesh is used for WAN implementations where failures are a critical concern.

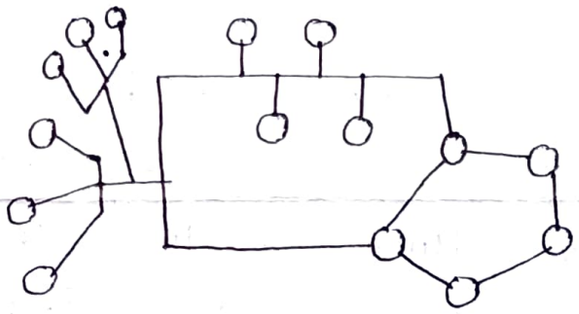


Partially mesh

Adv.: i) reliability ii) fast communication iii) easier reconfiguration - adding new devices is easy.

Disadv.: i) Cost ii) mgmt, iii) efficiency is low as redundant connections are more.

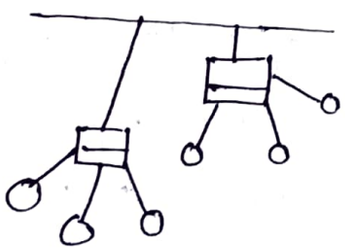
5. Hybrid topology: Combination of diff. topologies.



Adv.: reliable, scalable, flexible, effective.

Disadv.: Complex design, costly hubs - hybrid topology hubs are expensive.

6. Tree topology: Combination of bus & star. Parent-child hierarchy. Only one path b/w 2 nodes



adv.: i) support for broadband transmission, ii) easily expandable, iii) easily manageable, iv) error detn easy, v) limited failure.

disadv.: i) difficult troubleshooting, ii) high cost, iii) failure in main bus cable, iv) difficult to reconfigure.

7. Daisy chain topology: Connects all hosts in linear fashion. All connected to 2 hosts except end ones. Each link works as a single point of failure. Every intermediate host works as relay for its immediate hosts.

→ Hub, repeater, modem, cables are physical layer medium devices.

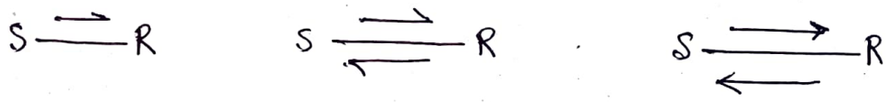
→ Types of medium:

a) Guided media: Signal is guided by the presence of physical media i.e. signal is under control & remains in the physical wire. eg. copper wire.

b) Unguided media: No physical path for the signal to propagate (essentially electromagnetic waves). No control on flow of signal. eg. radio waves.

→ Communication links:

· Communication throug Links classified as — Simplex, half-duplex, full-duplex.



· Links can be classified as —

i) Point to point: Only 2 nodes are connected to each other.

ii) Multipoint: Sharing communication, in which signal can be received by all nodes. (Broadcast)

2 kinds of problem arise in transmission —

✓ i) Attenuation: When a signal transmits in a N/W then the quality of signal degrades as the signal travels longer distances in the wire. To improve quality of amplifiers are used in regular distances.

ii) Noise : In a communication channel many signals are transmitted simultaneously. Certain random signals are also present in the medium. Due to the interference, signals get disrupted.

→ Bandwidth : # of bits that can be transmitted per second in the communication channel.

→ In guided transmission media generally two kinds of materials are used —

1. Copper → Coaxial cable
   → Twisted pair → Unshielded
   → Shielded
2. Optical fiber
   (Light used to send data). - Total internal reflection

Coaxial cable - cable TV

Twisted pair - telephone system

→ Wireless transmission. —

i) Radio transmission (cordless keyboad, wireless LANs, wireless ethernet)

ii) Terrestial microwave (Focused beam between 2 antennas)

iii) Satellite communication (Satellite acts as a switch in the sky. On earth VSAT - very small aperture terminal is used to transmit & receive data from satellite.)

→ Data Encoding :

A) Digital to Analog (Modem → modulator - demodulator used) — ASK, FSK, PSK

B) Digital to Digital C) Analog to digital (PCM, DM)

D) Analog to Analog (AM, FM, PM)

→ Digital to Analog :

i) Amplitude shift keying (Represents digital data as variations in the amplitude of a carriere wave.)

ii) Frequency shift keying (Change of frequency defines different digits)

iii) Phase shift keying (Phase of the carrier is discretely varied in relation either to a reference phase or to the phase of the immediately preceding signal element, in accordance with data being transmitted. Phase of carrier is shifted to represent '0', '1'.

→ Encoding techniques. : Digital to Digital

i) Non return to zero (NRZ)

* Problem with NRZ - Peterson Davie 112

Voltage level is constant during a bit interval. Problem arises when there's a long sequence of 1's or 0's & the voltage level is maintained at the same value for a long time. This creates a problem on the receiving side because now, the clock synchronisation is lost due to lack of any transitions & hence, it is difficult to determine the exact number of 0's & 1's in this sequence.

* Baseline wander, clock recovery problem (clock drift)

Two variations are —

a) NRZ-level : NRZ-L codes ~~shown~~ have the property that the

polarity of the signal changes only when the incoming signal changes from a '1' to a '0' or vice-versa.

b) NRZ-inverted: Transition at the beginning of bit interval = bit 1 and no transition at beginning of bit interval = bit 0 or vice-versa. (Differential encoding)

* NRZ-I has an advantage over NRZ-L. Consider the situation where 2 data wires are wrongly connected in each other's place. In NRZ-L all bit sequences will get reversed (as voltage levels got swapped). Whereas in NRZ-I since bits are recognised by transition, the bits will be correctly interpreted.

A disadvantage in NRZ is that a string of 0's or 1's will prevent synchronisation of transmitter clock with receiver clock & a separate clock line need to be provided.

ii) Biphase encoding: Modulation rate twice that of NRZ & BW correspondingly greater. Since there can be transition at the beginning as well as in the middle of the bit interval the clock operates at twice the data transfer rate.

a) Biphase Manchester / Manchester encoding: Synchronous clock encoding technique.

Baud rate = 2 x Bit rate (Biphase Manchester & Diff. Manchester)

* Clock rate = 2x Data transfer rate in biphase encoding

If we use 4B/5B or other schemes forcing enough transitions reliable clock recovery is possible.

Characteristics ~

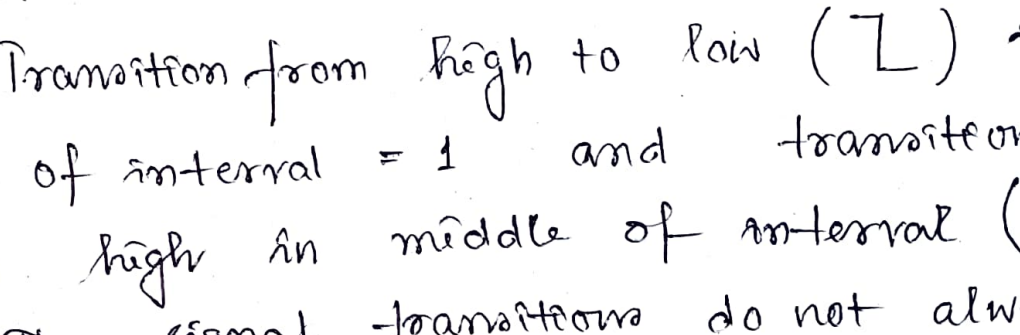
Transition from high to low (⊥) in middle of interval = 1 and transition from low to high in middle of interval (⌐) = 0.

The signal transitions do not always occur at the bit boundary but there is always a transition at the centre of each bit. It is biphase as each bit is encoded by a positive 90 degrees phase transition or by -ve. 90° phase transition.

The manchester encoding consumes twice the bandwidth of the original signal.

Advantages of the encoding is that the DC component of the signal carries no information.

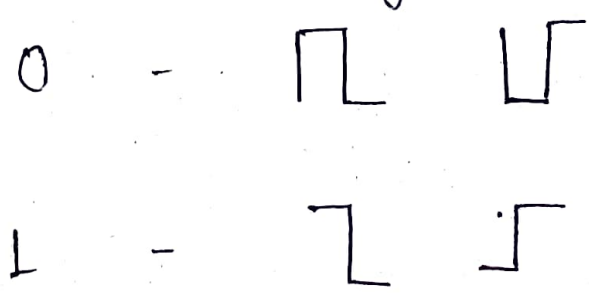
eg.

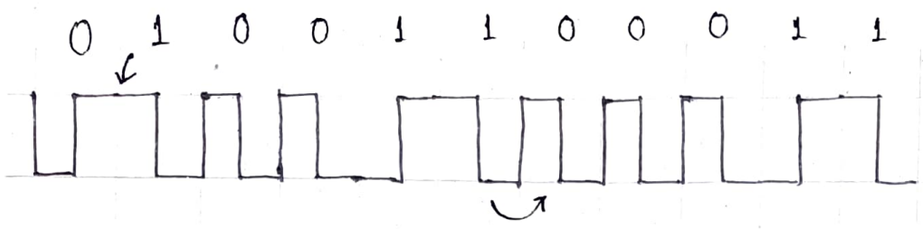


b) Differential Manchester Encoding :

Always a transition in middle of interval.

No transition at beginning of interval = 1 &
transition at beginning of interval = 0.





no transition at beginning as 1

transition at beginning as 0.

iii) 4B5B Encoding: In Manchester encoding, there's a transition after every bit. It means that we must have clocks with double the speed to send same amount of data as in NRZ encoding. We may say that only 50% data is sent. This performance factor can be significantly improved if we use a better encoding scheme. This scheme may have a transition after fixed number of bits instead of every other bit. Like if we have a transition after every 4 bits, then we will be sending 80% data of actual capacity. This is a significant improvement.

Block coding.

In 4B/5B, we convert 4 bits to 5 bits, ensuring at least one transition in them. Basic idea here is that 5 bit code must have at most one leading 0 & no more than two trailing zeros. Thus, it's ensured that we can't have more than 3 consecutive 0s. Now, these 5 bit codes are transmitted using NRZ-I then problem of consecutive 1's is solved.

Should have one leading 0 & no more than 2 trailing 0 } Words chosen such that 2 transactions take place per block of bits.

→ Analog to digital :

Digitization

$f_s \geq 2 f_{highest}$

i) Pulse code modulation (PCM)

ii) Delta Modulation.

- Data Link Layer (DLL) – Layer 2

→ DLL responsible for the node to node delivery of the message. Main function of this layer is to make sure data transfer is error-free from one node to another, over the physical layer. When a packet arrives in a N/W, it's the responsibility of DLL to transmit it to the host using its MAC address.

→ Packet in DLL is referred as frame. DLL is handled by the NIC & device drivers of host machines. Switch & bridge are DLL devices.

→ The packet received from N/W layer is further divided into frames depending on the frame size of NIC. DLL also encapsulates S's & R's MAC address in the header. The R's MAC address is obtained by placing an ARP (Address Resolution Protocol) request onto the wire asking "Who has that IP address" & the destination host will reply with its MAC address.

→ Responsibilities :

i) Framing : DLL divides the stream of bits received from the N/W layer into manageable data units called frames.

ii) Physical addressing : If frames are to be distributed to ~~distributed~~ different systems on the network, the DLL adds a header to the frame to define the sender and/or receiver of the frame. If the frame is intended for a system outside the sender's network, the receiver address is the address of the device that connects the N/W to the next one.

iii) Flow control : If the rate at which the data are absorbed by the receiver is less than the rate at which data are produced in the sender, the DLL imposes a flow control mechanism to avoid overwhelming the receiver.

iv) Error control : DLL adds reliability to the physical layer by adding mechanisms to detect & retransmit damaged or lost frames. It also uses a mechanism to recognise duplicate frames. Error control is normally achieved through a trailer added to the end of the frame.

v) Access control : When 2 or more devices are connected to the same link, DLL protocols are necessary to determine which device has control over the link at any given time.

→ Data link layer is divided into 2 sublayers -

i) Logical link control (LLC) — EC, FC

ii) Media access control (MAC) - framing, FC, EC, addressing

Framing The LLC manages communications between devices over a single link of a N/W. LLC is defined in the IEEE 802.2 & supports both connectionless and connection-oriented services used by higher layer protocols.

The MAC manages protocol access to the physical N/W medium.

Framing

a) Fixed size framing - No need for defining the boundaries of the frames; the size can be used as a delimeter.

✓ b) Variable size framing - Need a way to define the end of the frame & the beginning of the next.

2 approaches - i) Character oriented & ii) bit oriented.

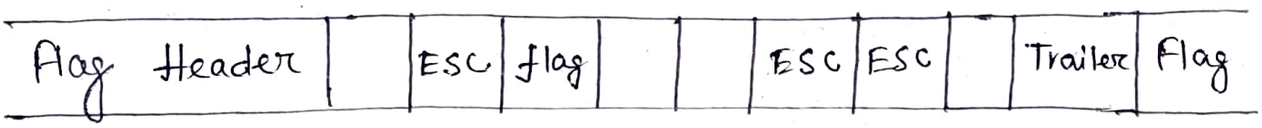
Character oriented protocols. (Byte stuffing)

Data to be carried are 8 bit characters. To separate one frame from the next, an 8-bit flag is added at the beginning & at the end of the frame to signal the start & end of frame.

Byte stuffing is the process of adding 1 extra byte if there is a flag or escape character

in the text data itself.

Frame sent



→ In variable size framing, we can use length field or end delimeter to define start & end of frame.

→ Length field: We can introduce a length field in the frame to indicate the length of the frame. Used in Ethernet 802.3. Problem is that the length field might get corrupted.

→ End delimeter (ED): We introduce an ED (pattern) to indicate end of the frame. Used in Token ring. Problem with this is that ED can occur in the data. Can be solved by —

A) Character stuffing / Byte stuffing

Used when frames consist of character. If data contains ED then, byte is stuffed into data to differentiate it from ED.

Let ED = '$'. If data contains '$' anywhere, it can be escaped using '\0' character. If data contains '\0' then use \0\0 \0$.

If it is used, the disadvantage is it is very costly & obsolete method.

Each frame starts with DLE STX (Data link escape - start of text) & ends with DLE ETX.

✓ B) Bit stuffing

Let ED = 01111 and if data = 01111, Sender stuffs a bit to break the pattern i.e. appends a 0 in the data 01110 1. Receiver receives the frame. If data contains 011101, receiver removes the 0 & reads it.

eg. If data = 0111000111110 & ED = 01111

then after bit stuffing we have to add a 0 after we see three 1's in the data.

0111 0 000 1110 10

eg. data = 01111, ED = 01111,

we're gonna add 0 after 3 consecutive 1's.

011101

eg. ED = 011111 Add 0 after 4 1's

data 011110 → 0111100

011111 → 0111101

Q. G'14. A bit stuffing based framing protocol uses an 8 bit delimeter pattern of 01111110. If the o/p bit string after stuffing is 011111001 01, then the i/p string is —

01111101 01.

→ In fixed size framing the drawback is it suffers from internal fragmentation if data size is less than the frame size.

Solution to it - use padding (adding dummy bits to data that is less than the frame size).

◆ Physical addressing :

MAC - Local identification
IP - Global identification

2 types of addresses -

a) Physical addresses (static, constant)

b) Logical addresses.

Physical address should be unique within the N/W. Logical address should be unique in the entire world.

IP address - 32 bit no., software no.

(Physical address) MAC address - 48 bit no., hardware no. printed on our NIC → ROM. MAC address is divided into 3 parts -

i) Manufacturer / Vendor ID
ii) Date of manufacture
iii) Serial no. of the device

} Unique globally

** What directs the packet from S to D is the IP address. But, what gets the packet from the S to R_A & then from R_A to R_B & then from R_B to D is the MAC address. MAC address handles the physical connection from computer to computer while IP addresses handle the logical routeable connection from both computer to computer & N/W to N/W.

No 2 devices in a N/W should have same MAC.

→ AppleTalk does not use MAC. It artificially generates a random number & assigns to users.

- **Network Layer:**

Works for the transmission of data from one host to the other located in different N/Ws. It also takes care of packet routing i.e. selection of the shortest path to transmit the packet, from the number of routes available.

→ If 2 systems are connected to the same link, there's no need for a N/W layer.

→ Main responsibilities:

i) Host to host connectivity

ii) Switching

iii) Routing: Determining how packets will be routed from source to destⁿ. It can be of 3 types – a) static (routes are based on static tables that are wired into the network & are rarely changed), b) dynamic (all packets of one application can follow different routes depending upon the topology of the N/W, the shortest path & the current N/W load), c) semi-dynamic (a route is chosen at the start of each conversation & then all the packets of the application follow the same route).

iv) Congestion control: If all the N/Ws send packets at the same time with maximum rate possible then the router may not be able to handle all the packets & may drop some packets. In this context, the dropping of packets should be minimised & the source whose packet was dropped should be informed. The control of such congestion is also a function of the N/W layer.

Other issues in this layer - transmitting time, delays, jittering.

v) Logical addressing: In order to identify each device on internetwork uniquely, N/W layer defines an addressing scheme. The sender's & receiver's IP address are placed in the header by network layer.

vi) Fragmentation.

→ Services provided by N/W -

i) Connection-less ii) Connection-oriented.

→ N/W layer does not guarantee that the packet will reach its intended destn.

→ Segment in N/W layer is called a packet.

→ Router is a networking device.
(Router has only PL, DL, NL).

- Transport layer :

Responsible for process to process delivery of the entire message. A process is an application program running on a host. Whereas the N/W layer oversees source-to-dest^n delivery of individual packets, it does not recognise any relationship between those packets. It treats each one independently. The transport layer, ensures that the whole message arrives intact & in order. We refer to transport layer packet as a segment.

Main functions –

a) Service point addressing : Transport layer header must include a type of address called service point address (SPA' or port address).

b) Segmentation & reassembly : Data accepted by transport layer from the session layer is split up into smaller units (~~fragmentation~~) if needed & then passed to the N/W layer. The data provided by the N/W layer to the transport layer on the receiving side is reassembled.

c) Connection control : A connectionless transport layer treats each segment as an independent packet & delivers it to the transport layer at the dest^n. A connection oriented transport layer makes

a connection with the transport layer at the dest^n machine first before delivering packets. After all data is transferred, connection is terminated.

d) Flow control (End to end rather than across a single link)

e) Error control (Error correction through retransmission)

f) Multiplexing & demultiplexing

- Session layer: Responsible for dialogue control & synchronisation.

a) Dialogue control: Session layer allows 2 systems to enter into a dialogue. It allows the communication between 2 processes to take place in either half-duplex or full-duplex mode.

b) Synchronization: Allows a process to add checkpoints or sync. points to a stream of data.

- Presentation layer: Concerned with the syntax & semantics of the information transmitted.

a) Translation: Processes in 2 systems are usually exchanging info in the form of character strings, numbers etc. Info must be changed to bit streams before being transmitted.

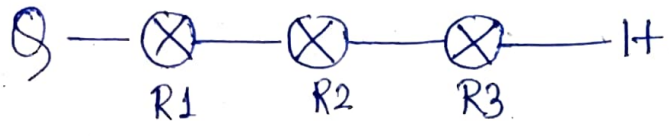
b) Encryption: To carry sensitive data, a system must be able to ensure privacy.

c) Compression: In the transmission of text, audio, video.

- Application Layer:

Enables the user, whether human or software, to access the N/W. Uses protocols like HTTP, FTP, SMTP, DNS etc. Packet of information in this layer is message.

Q. G'14.



S connects to H via HTTP & downloads file. Session layer encryption is used, with DES as the shared key encryption protocol. Consider these information —

1. URL of the file downloaded by S
2. TCP port no. at S & H
3. IP addresses of S & H
4. Link layer addresses of S & H.

Which of these can an intruder learn through sniffing at R2 alone?

→ Can't learn (1); as URLs & download are functioned at application layer.

✓ Can learn (2) as port no. is encapsulated in the payload field of IP datagram.

Can learn (3), as IP addresses & routers are functioned at N/W layer of OSI model.

Can't learn (4) as it is related to DLL. (Only have MAC addresses of R1 & R3, not B or H)

* A packet is an information unit whose source & destination are network layer entities.

A packet is composed of network layer header (& possibly a trailer) & upper-layer data. The header & trailer contain control information intended for the N/W layer entity in the destination system.

Datagram usually refers to an information unit whose source & destn are N/W layer entities that use connectionless network service.

Segment usually refers to an information unit whose source & destination are transport layer entities.

Message is an information unit whose source & destn entities exist above the N/W layer.

Cell is an information unit of a fixed size whose source & destination are data link layer entities. (Used in switched environments, such as Asynchronous Transfer Mode - ATM & Switched Multimegabit Data Service - SMDS). A cell is composed of the header & payload.

Data unit is a generic term that refers to a variety of information units.

PL: Moves bits b/w devices, specified voltage, rate, pin out cables.

Bit

Protocols – EIA/TIA-232, 100BaseTX, ISDN, 802.11.

DLL: Combines data bytes into frames, perform error detection (not correction) & provides access to media using MAC address, physical addressing, framing

MACL
LLCL
frame

P - RAPAE, PPP, Frame relay, ATM, Fiber cable.

NLC: Provisions logical addressing, using which routers route data.

Packet

P- IP, IPX, ICMP, IPSEC, ARP, MPLS.

TL: Provisions con. oriented & -less end-end delivery of segments, error correction

Segment

P- TCP, UDP

SL: Keep different app. data separate & synchronization, dialogue control

P- NETBIOS, SAP

PL: Presents data & handle encryption., translation, compression.

P- MPEG, ASCH, SSL, TLS,

AL: Provides user interface using FTP, HTTP.

P- SMTP, FTP, HTTP, POP3, SNMP.

PL: ISDN (Integrated service digital network), DSL (Digital subscriber line), Ethernet physical layer (10 BASES, 10 BASE-T, 100 BASE-T)

DLL: ~~ARP (Addr. resol^n protocol)~~, X
FDDI (Fibre Distributed Data Interface)
HDLC (High level data link control)
VLAN (Virtual LAN)

NLP: ATM (Async. transfer mode), ARP,
SPB (Shortest Path bridging)
IP, ICMP

NL+TL: AppleTalk, IPX (Internetwork packet exchange) / SPX (Sequenced packet exchange), IP suite

TL: TCP, UDP, DCCP (Datagram congestion control protocol),
FCP (Fibre Channel Protocol)

SL: RPC (remote procedure call), H.245, NetBIOS.

PL: TLS (Transport layer security), SSH, Telnet

AL: DHCP (dynamic host config protocol)
DNS, HTTP, HTTPS, POP3, SMTP, TFTP.

Devices.

1. Hubs, repeaters, cables, fibres
2. bridge, modem, network card, 2 layer switch
3. router, brouter (bridge+..), 3-layer switch
4. gateway, firewall
5. gateway, " , PCs
6. "
7. " phones, servers → User apps

LAN Technologies

* Local area networks (LAN).

Network of computers confined to a small area which may be a room, building or a group of buildings. LAN may be wired, wireless or a combination of both.

* Standard technologies used to build a wired LAN are — ethernet, token ring.

* Ethernet (DLL)

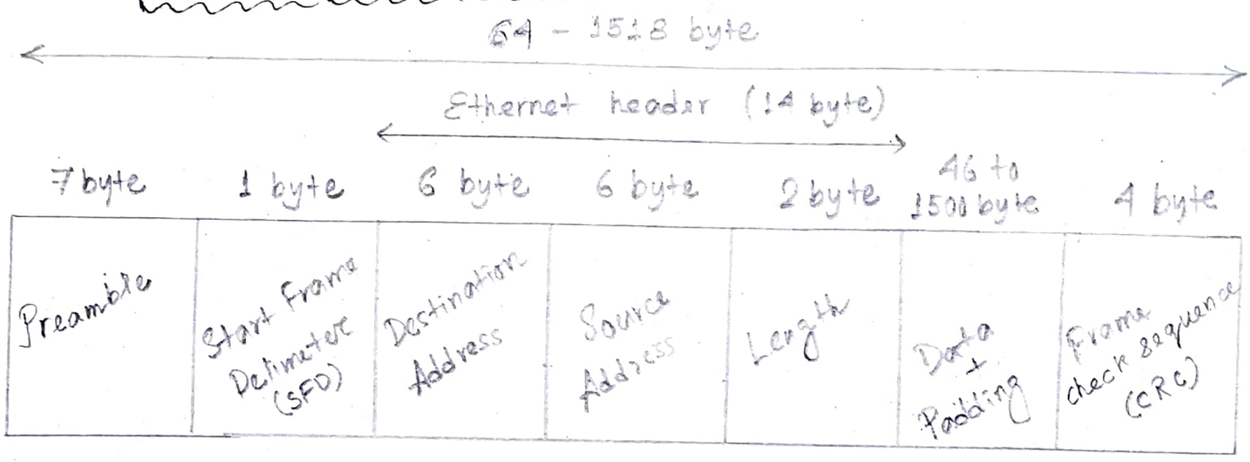
Defined under IEEE 802.3.

→ Characteristics

1. Ethernet uses bus topology.
2. All stations are connected to a single half duplex link.
3. Ethernet uses CSMA/CD as access control method to deal with the collisions.
4. Ethernet uses manchester encoding for converting data bits into signals.
5. Ethernet evolution has four generations -

Standard ethernet	10 Mbps
Fast ethernet	100 Mbps
Gigabit ethernet	1 Gbps
Ten Gigabit ethernet	10 Gbps

→ Ethernet frame format.



1. Preamble : It alerts the stations that a frame is going to start. Also enables the sender & receiver to establish bit synchronisation. It is actually added at the physical layer & is not part of the frame.

2. SFD : Signals the beginning of the frame.

Preamble & SFD are added by the physical layer & represents the physical layer header. Sometimes, SFD is considered to be a part of preamble.

3. DA : MAC address of the dest^n for which data is destined.

4. SA : MAC address of the source that is sending the data.

5. Length : Length of the data field. As ethernet uses variable sized frames, this field is required.

Max value that can be accomodated in this field is $2^{(8+8)} - 1 = 65535$. But, it does not mean max. data that can be sent in one frame is 65535 bytes. Max amount that can be sent is 1500 bytes in a ethernet frame. This is to avoid

the monopoly of any single station.

6. Data: Also called payload field. Length of the field lies in the range [46 bytes, 1500 bytes]. Thus, in an Ethernet frame, min data has to be 46 bytes & max data can be 1500 bytes.

- Minimum length of data field ~

In CSMA/CD (as Ethernet uses it),

min length of data packet $= 2 \times T_p \times B$

$T_t \geq 2T_p$

$L \geq 2T_pB$.

Substituting standard values of Ethernet, it is found the min length of ethernet frame has to be 64 bytes, starting from the destination address field to the CRC field & (72 bytes including preamble & SFD.)

Therefore min length of data field has to be = $64 - (6+6+2+4) = 46$ bytes.

- Maximum length of data field ~ (as per 802.3

max. amount of data that can be sent in a Ethernet frame is 1500 bytes. If Ethernet allows the frames of big sizes, then other stations may not get the fair chance to send their data.

7. Frame check sequence: Contains CRC code for error detection.

→ Advantages of using Ethernet.

i) Simple to understand & implement.

ii) Maintenance easy.

iii) Cheap.

→ Limitations.

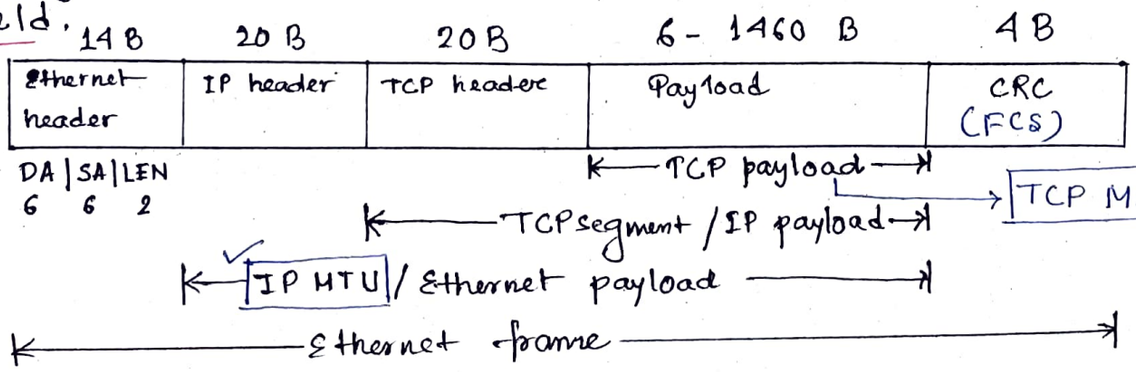
i) It can't be used for real time applications. Real time applications require the delivery of data within some time limit. Ethernet is not reliable for high probability of collisions. High no. of collisions may cause a delay in delivering the data to its destn.

ii) It can't be used for interactive applications. They require the delivery of even very small amount of data (Ethernet min 46 bytes).

iii) Can't be used for client-server applications. They require that server must be given higher priority than clients. Ethernet has no facility to set priorities. (In token ring → prioritization of master node by increased THT)

Token ring overcomes these limitations.

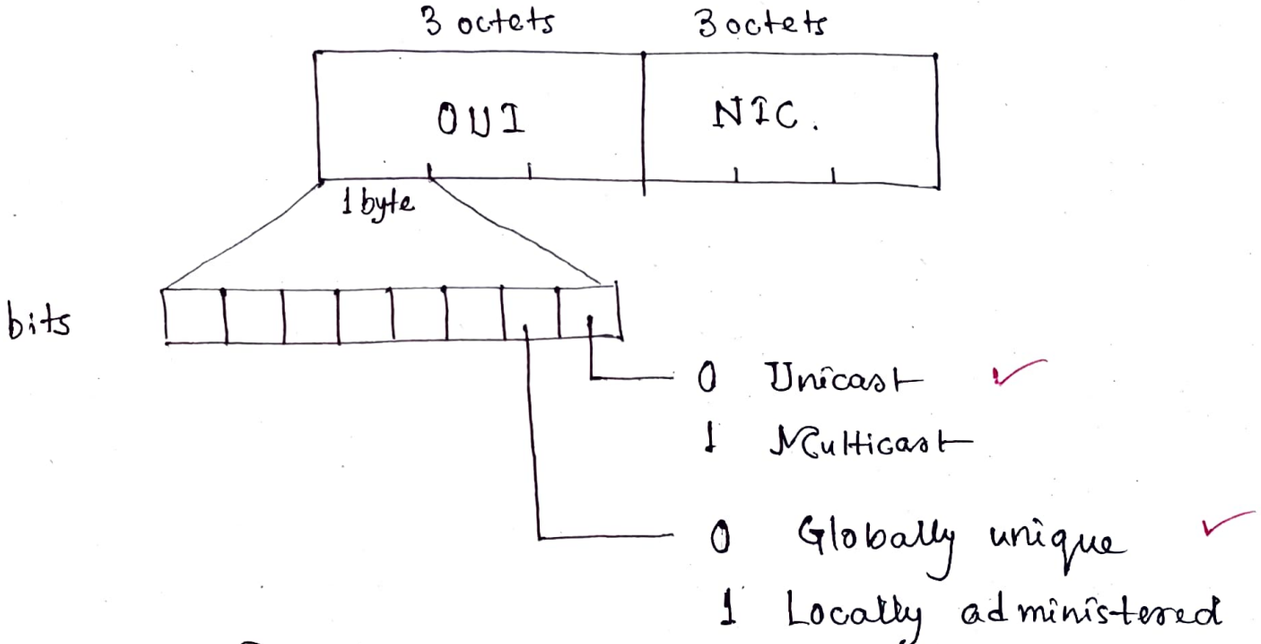
* For data transmission, TCP segment sits inside the IP datagram payload field. IP datagram sits inside the Ethernet payload field.



→ MAC Addressing

Each station on an Ethernet network has its own NIC. The NIC provides the station with a 6 byte physical address.

MAC address is a 12 digit hexadecimal number, represented by colon-Hex notation. First 6 digits identify the manufacturer (Organisational unique identifier). The rightmost 6 digits represent Network interface controller, which is assigned by manufacturer.



Types:

1. Unicast: A unicast addressed frame is only sent out to the interface leading to specific NIC. If the LSB of first octet of an address is set to zero, the frame is meant to reach only one receiving NIC. MAC address of source machine is always unicast.

2. Multicast: Allows the source to send a frame to group of devices. In layer-2 multicast address, LSB of first octet is set to one. IEEE has allocated the address block 01-80-C2-xx-xx-xx for group

addresses for use by standard protocols.

3. Broadcast : Similar to N/W layer, broadcast is also possible on underlying layer (data link layer). Ethernet frames with ones in all bits of the dest^{n} address, referred as broadcast address. Frames which are destined with MAC address FF-FF-FF-FF-FF-FF will reach to every computer belonging to that LAN segment.

eg. AA:30:10:21:10:1A

A = 1010 Hence unicast

eg. 47:20:1B:2E:08:EE

7 = 0111 Hence, multicast

eg. FF:FF:FF:FF:FF:FF

All 1's Hence, broadcast.

Multicast is superset of broadcast.

48b bin.
↓
12 Hex

8b : .... : 8b
bin

2dig : .... : 2dig
Hex

Switching

* Switching : Process of moving the data packets towards their destination by forwarding them from one port to the other port.

Switching techniques –

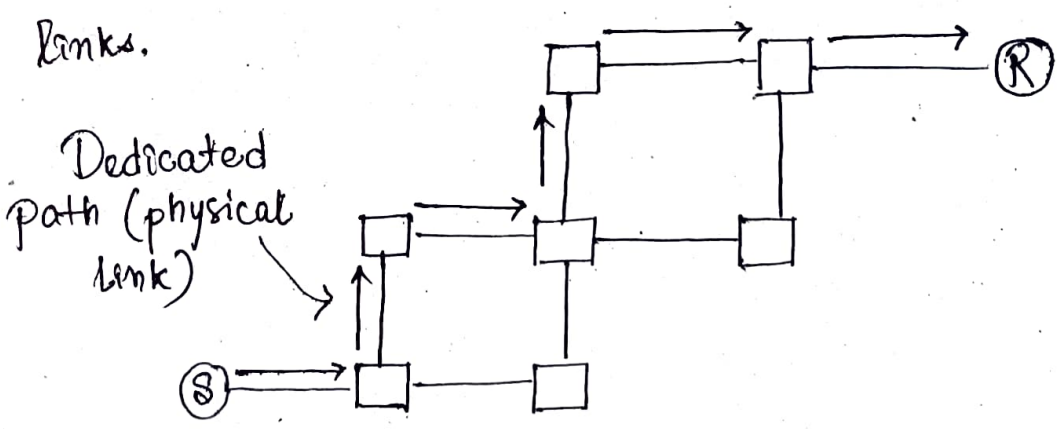
1. Circuit switching
2. Message switching
3. Packet switching
   - Datagram switching
   - Virtual circuit switching

* Circuit switching (Implemented at physical layer. Now outdated.)

Operates in 3 phases –

1. Establishing a circuit : A circuit is established between the two ends.

Circuit provides a dedicated path for data to travel from one to the other end. Resources are reserved at intermediate switches which are used during the transmission. The intermediate switches are connected by the physical links.



2. Transferring the data. : After the circuit is established, the entire data travels over the dedicated path from one end to the other.

3. Disconnecting the circuit : After the data transfer is completed, the circuit is torn down.

→ Total time :

Time taken to transmit a message in circuit switched N/W =

Connection setup time + T_t + T_p + Tear down time.

→ T_t is independent of the # of links.

Where $T_t = L/B$

$T_p = \frac{\text{\# hops on way} \times \text{distance}}{\text{propagation speed}}$

→ Advantages

i) A well defined & dedicated path exists for the data to travel.

ii) No header overhead.

iii) No waiting time at any switch & the data is transmitted without any delay.

iv) Data always reaches the other end in order.

v) No reordering is required.

→ Disadvantages.

i) Channel is blocked for two ends only.

ii) Inefficient in terms of utilization of system resources.

iii) Time required for establishing the circuit is too long.

iv) Dedicated channels require more bandwidth.

v) More expensive.

vi) Routing decisions cannot be changed once the circuit is established.

Q. Consider all links in the N/W use TDM with 24 slots & have a data rate of 1.536 Mbps. Assume that host A takes 500 msec to establish an end to end circuit with host B before begin to transmit the file. If the file is 512 Kilobytes, then how much time it will take to send the file from host A to B?

→ Bandwidth per user = $\frac{1.536 \text{ Mbps}}{24}$ = 64 Kbps

$T_t = \frac{512 \text{ KB}}{64 \text{ Kbps}}$ = 65536 msec

Time taken to send file = 500 msec + 65536 msec

= 66036 msec.

* Message Switching

No dedicated path to transfer data. The entire message is treated as a single data unit. The message is then forwarded from hop to hop. Store & forward is an important characteristic. The message carries a header that contains the full information about destination. When any intermediate switch receives the message, it stores the entire message. The message is stored until sufficient resources become available to transfer it to the next switch. When resources become available, the switch forwards the message to the next switch.

→ Advantages:

i) It improves the channel efficiency over circuit switched network. In circuit switching, channel is blocked for 2 ends only. But here, more devices can share the channel.

ii) Reduces traffic congestion. The message may be temporarily stored in the route & then forwarded whenever required.

iii) Helpful in setting the message priorities due to store & forward technique.

→ Disadvantages:

i) It requires enough storage at every switch to accomodate the entire message during the transmission.

ii) Extremely slow due to store & forward technique. Also, the message has to wait until sufficient resources become available to transfer it to the next switch.

→ Message switching is replaced by packet switching.

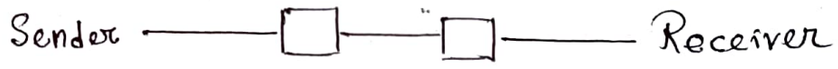
* Packet Switching

→ The entire message to be sent is divided into multiple smaller size packets. This process of dividing a single message into smaller size packets, is called packetization. These smaller packets are sent one after the other. It gives the advantage of pipelining & reduces the total time taken to transmit the message.

→ Optimal packet size :

If the packet size is not chosen wisely, then it may result in adverse effects. It might increase the time taken to transmit the message.

eg. N/W having B = 1 MBps, message size is 1000 B, each packet contains a header of 100 B.



Now, question is, how many packets the message must be divided into to minimise the total time taken to send the message.

We ignore T_p (propagation delay). The reason is in packet switching, T_t dominates over T_p. This is because each packet is transmitted over the link at each hop.

✗ case 1 : Sending in 1 packet only

$$T_t = \frac{(1000 + 100)\ B}{1\ MBps} = 1.1\ msec$$

✓ Time taken = 3 X 1.1 msec = 3.3 msec.
↑ because 3 hops

✗ case 2 : Sending in 5 packets.

Data sent in one packet = $\frac{1000}{5} = 200\ B$

✓ Size of one packet = (200 + 100) = 300 B

✓ $T_t = \frac{300\ B}{1\ MBps} = 0.3\ msec$

Time taken by first packet = 3×0.3 msec
$= 0.9$ msec

Time taken by remaining packets due to pipelining = $4 \times 0.3 = 1.2$ msec.

Total time taken = $0.9 + 1.2 = \underline{2.1 \text{ msec}}$.

After 0.9 ms is over, 2 nos 0.3 ms the receiver will get a packet. 0.3×4 ms

* case-3: Sending in 10 packets

$L = \frac{1000}{10} B + 100 B = 200 B$

$T_t = \frac{200 B}{1 MBps} = 0.2$ msec.

In the beforesaid way, time taken in total =

$(3 \times 0.2 + 9 \times 0.2) = \underline{2.4 \text{ msec}}$.
first other 9

case-4: Sending in 20 packets

$L = \left(\frac{1000}{20} + 100\right) B = 150 B$

$T_t = \frac{150 B}{1 MBps} = 0.15$ msec.

Total time = $(3 \times 0.15 + 19 \times 0.15) = 3.3$ msec

generally,
$t = T_t (\#hops + (n-1))$
↓ for 1 packet ↓ #packets

So, we can conclude –

✓ total time decreas... is reduced but only up to ... after that total time increas...

In the example, ... would be the best choice

Optimal packet size:

$n = \sqrt{\frac{M(\#-1)}{h}}$

packet size = $\left(\frac{M}{n} + h\right)$

M - msg size
\# - #hops
h - hdr size
n - #packets divided into

In circuit sw^n g, message units don't have to wait at switches, unlike packet switching (where store & forward is used).

In CS → $T_p > T_t$ (have doubt)
In PS → $T_t > T_p$

→ Sending one packet from source to destⁿ over a path consisting of N links each of rate R (thus, there are N-1 routers),

$$\text{total delay, } d_{\text{end-to-end}} = N\frac{L}{R}.$$ for one packet

For P packets it will be (as pipelining is used),

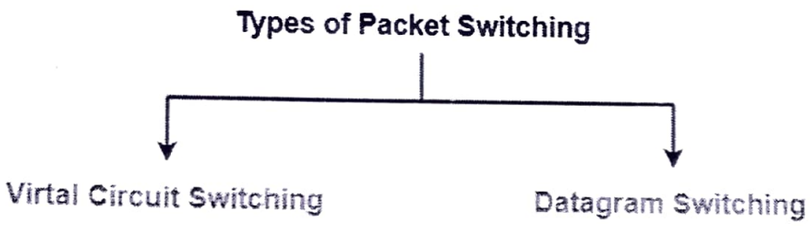
$$d = N\frac{L}{R} + (P-1)\frac{L}{R}.$$

→ Queuing delays & packet loss.

Each packet switch has multiple links attached to it. For each link, the switch has an output buffer / queue, which stores packets that the router is about to send into that link. If an arriving packet needs to be transmitted onto a link but finds the link busy, the arriving packet must wait in the o/p buffer. Thus, in addition to store-and-forward delays, packets suffer o/p buffer queuing delays. These delays are variable & depend on the level of congestion in the N/W. Since, the amount of buffer space is finite, an arriving packet may find that the buffer is full. In this case, packet loss will occur. Either the arriving packet or one of the already queued packets will be dropped (packet dropping).

Types of Packet Switching-

Packet switching may be carried out in the following 2 ways-



1. Virtual Circuit Switching
2. Datagram Switching

Virtual Circuit Switching-

Virtual circuit switching operates in the following three phases-

1. Establishing a circuit
2. Transferring the data
3. Disconnecting the circuit

3. Disconnecting The Circuit-

After the data transfer is completed,

- The connection is disconnected.

Datagram Switching-

In datagram switching,

- There exists no dedicated path for data to travel.
- The header of each packet contains the destination address.
- When any intermediate switch receives the packet, it examines its destination address.
- It then consults the routing table.
- Routing table finds the corresponding port through which the packet should be forwarded.

Virtual Circuit Switching Vs Datagram Switching-

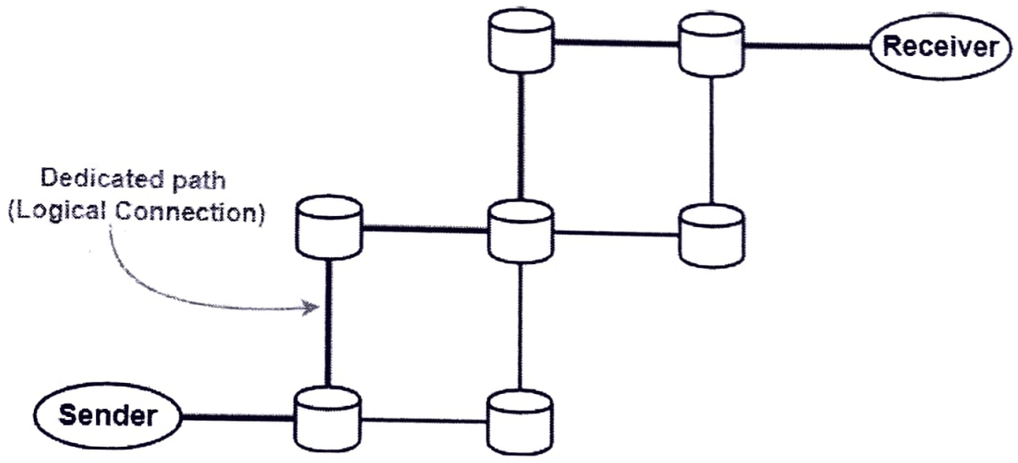
The following table shows a comparison between virtual circuit switching and datagram switching-

Virtual Circuit Switching	Datagram Switching

1. Establishing A Circuit-

In this phase,

- A logical connection is established between the two ends.
- It provides a dedicated path for data to travel from one to the other end.
- Resources are reserved at intermediate switches which are used during the transmission.



2. Transferring The Data-

After the connection is established,

- The entire data travels over the dedicated path from one end to the other end.

3. Disconnecting The Circuit-

After the data transfer is completed,

- The connection is disconnected.

Datagram Switching-

In datagram switching,

- There exists no dedicated path for data to travel.
- The header of each packet contains the destination address.
- When any intermediate switch receives the packet, it examines its destination address.
- It then consults the routing table.
- Routing table finds the corresponding port through which the packet should be forwarded.

Virtual Circuit Switching Vs Datagram Switching-

The following table shows a comparison between virtual circuit switching and datagram switching-

Virtual Circuit Switching	Datagram Switching

The first packet during its transmission- 1) Informs the intermediate switches that more packets are following. 2) Reserve resources (CPU, bandwidth and buffer) for the following packets at all the switches on the way.	The first packet does not perform any such task during its transmission.
The packets are never discarded at intermediate switches and immediately forwarded since resources are reserved for them.	The packets may be discarded at intermediate switches if sufficient resources are not available to process the packets.
It is a connection oriented service since resources are reserved for the packets at intermediate switches.	It is a connection less service since no resources are reserved for the packets.
All the packets follow the same dedicated path.	All the packets take path independently.
Data appears in order at the destination since all the packets take the same dedicated path.	Data may appear out of order at the destination since the packets take path independently.
It is highly reliable since no packets are discarded.	It is not reliable since packets may be discarded.
It is costly.	It is cost effective.
Only first packet requires a global header which identifies the path from one end to other end. All the following packets require a local header which identifies the path from hop to hop.	All the packets require a global header which contains full information about the destination.
ATM (Asynchronous Transfer Mode) uses virtual circuit switching.	IP Networks use datagram switching.
Virtual circuit switching is normally implemented at data link layer.	Datagram switching is normally implemented at network layer.

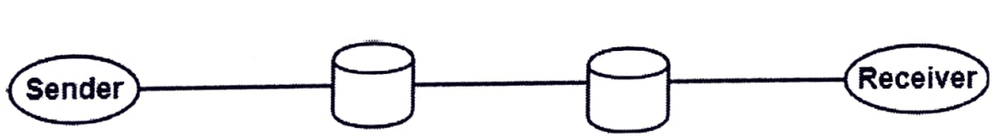
PRACTICE PROBLEM BASED ON PACKET SWITCHING TECHNIQUE-

Problem-

In a packet switching network, packets are routed from source to destination along a single path having two intermediate nodes. If the message size is 24 bytes and each packet contains a header of 3 bytes, then the optimum packet size is-

1. 4 bytes
2. 6 bytes
3. 7 bytes
4. 9 bytes

Solution-



Let bandwidth of the network = X Bps and 1 / X = a

Option-A: Packet Size = 4 Bytes

In this case,

- The entire message is divided into packets of size 4 bytes.
- These packets are then sent one after the other.

Data Sent in One Packet-

Data size

= Packet size – Header size

= 4 bytes – 3 bytes

= 1 byte

Thus, only 1 byte of data can be sent in each packet.

Number Of Packets-

Number of packets required

= Total data to be sent / Data contained in one packet

= 24 bytes / 1 byte

= 24 packets

Transmission Delay-

Transmission delay

= Packet size / Bandwidth

= 4 bytes / X Bps

= 4a sec

Time Taken By First Packet-

Time taken by the first packet to reach from sender to receiver

= 3 x Transmission delay

= 3 x 4a sec

= 12a sec

Time Taken By Remaining Packets-

Time taken by the remaining packets to reach from sender to receiver

= Number of remaining packets x Transmission delay

= 23 x 4a sec

= 92a sec

Total Time Taken-

Total time taken to send the complete message from sender to receiver

= 12a sec + 92a sec

= 104a sec

Option-B: Packet Size = 6 bytes

In this case,

- The entire message is divided into packets of size 6 bytes.
- These packets are then sent one after the other.

Data Sent in One Packet-

Data size

= Packet size – Header size

= 6 bytes – 3 bytes

= 3 bytes

Thus, only 3 bytes of data can be sent in each packet.

Number Of Packets-

Number of packets required

= Total data to be sent / Data contained in one packet

= 24 bytes / 3 bytes

= 8 packets

Transmission Delay-

Transmission delay

= Packet size / Bandwidth

= 6 bytes / X Bps

= 6a sec

Time Taken By First Packet-

Time taken by the first packet to reach from sender to receiver

= 3 x Transmission delay

= 3 x 6a sec

= 18a sec

Time Taken By Remaining Packets-

Time taken by the remaining packets to reach from sender to receiver

= Number of remaining packets x Transmission delay

= 7 x 6a sec

= 42a sec

Total Time Taken-

Total time taken to send the complete message from sender to receiver

= 18a sec + 42a sec

= 60a sec

Option-C: Packet Size = 7 bytes

In this case,

- The entire message is divided into packets of size 7 bytes.
- These packets are then sent one after the other.

Data Sent in One Packet-

Data size

= Packet size – Header size

= 7 bytes – 3 bytes

= 4 bytes

Thus, only 4 bytes of data can be sent in each packet.

Number Of Packets-

Number of packets required

= Total data to be sent / Data contained in one packet

= 24 bytes / 4 bytes

= 6 packets

Transmission Delay-

Transmission delay

= Packet size / Bandwidth

= 7 bytes / X Bps

= 7a sec

Time Taken By First Packet-

Time taken by the first packet to reach from sender to receiver

= 3 x Transmission delay

= 3 x 7a sec

= 21a sec

Time Taken By Remaining Packets-

Time taken by the remaining packets to reach from sender to receiver

= Number of remaining packets x Transmission delay

= 5 x 7a sec

= 35a sec

Total Time Taken-

Total time taken to send the complete message from sender to receiver

= 21a sec + 35a sec

= 56a sec

Option-D: Packet size = 9 Bytes

In this case,

- The entire message is divided into packets of size 9 bytes.
- These packets are then sent one after the other.

Data Sent in One Packet-

Data size

= Packet size – Header size

= 9 bytes – 3 bytes

= 6 bytes

Thus, only 6 bytes of data can be sent in each packet.

Number Of Packets-

Number of packets required

= Total data to be sent / Data contained in one packet

= 24 bytes / 6 bytes

= 4 packets

Transmission Delay-

Transmission delay

= Packet size / Bandwidth

= 9 bytes / X Bps

= 9a sec

Time Taken By First Packet-

Time taken by the first packet to reach from sender to receiver

= 3 x Transmission delay

= 3 x 9a sec

= 27a sec

Time Taken By Remaining Packets-

Time taken by the remaining packets to reach from sender to receiver

= Number of remaining packets x Transmission delay

= 3 x 9a sec

= 27a sec

Total Time Taken-

Total time taken to send the complete message from sender to receiver

= 27a sec + 27a sec

= 54a sec

Observations-

- Total time taken when packet size is 4 bytes = 104a sec
- Total time taken when packet size is 6 bytes = 60a sec
- Total time taken when packet size is 7 bytes = 56a sec
- Total time taken when packet size is 9 bytes = 54a sec

Result-

Time taken is minimum when packet size is 9 bytes.

Thus, Option (D) is correct.

Circuit Switching	Packet Switching	
	Virtual Circuit Switching	Datagram Switching
Connection oriented service	Connection oriented service	Connection less service
Ensures in order delivery	Ensures in order delivery	Packets may be delivered out of order
No reordering is required	No reordering is required	Reordering is required
A dedicated path exists for data transfer	A dedicated path exists for data transfer	No dedicated path exists for data transfer
All the packets take the same path	All the packets take the same path	All the packets may not take the same path
Resources are allocated before data transfer	Resources are allocated on demand using 1st packet	No resources are allocated
Stream oriented	Packet oriented	Packet oriented
Fixed bandwidth	Dynamic Bandwidth	Dynamic bandwidth
Reliable	Reliable	Unreliable
No header overheads	Only label overheads	Higher overheads
Implemented at physical layer	Implemented at data link layer	Implemented at network layer
Inefficient in terms of resource utilization	Provides better efficiency than circuit switched systems	Provides better efficiency than message switched systems
Example- Telephone systems	Examples- X.25, Frame relay	Example- Internet

→ Forwarding tables, Routing protocols.

Each router has a forwarding table that maps destination addresses (or portions of dest^n addresses) to that router's outbound links. When a packet arrives at a router, the router examines the address & searches its forwarding table, using this dest^n address, to find the outbound link.

- Routing table & forwarding table.

Routing table is a layer 3 table that states that for x.x.x.x/y IP dest^n, go through z.z.z.z router.

Forwarding table is a layer 2 table that states for communicating with z.z.z.z router send packets to MAC address aa:bb:cc:dd:ee:ff.

For example, for. table might say that a packet bound to a dest^n in 192.168.1.0/24 should be sent out of physical port ethernet1.

In our local network, we use the forwarding table to get the other hosts' MAC addr. & send them the packets. Your network device will broadcast an arp (whohas ip z.z.z.z) packet at layer 2 to get the relevant MAC addr.

To communicate with a host in a diff. subnet, we should route it through a router within our local network. Routing table tell the IP of that router.

Routing table contains all the routes a node is willing to keep & the information there is being used by routing protocols. Forwarding table is used by the h/w to physically move the packets in & out of interfaces.

See a pdf named - "routing-forwarding" in CN folder

Forwarding : effective transfer of a packet, frame etc.
(Direct & Indirect forwarding)

Routing : decision to take route b/w a S & D

Internet Protocol.

IPv4 | IPv4 Header | IPv4 Header Format

Computer Networks

6

Internet Protocol Version 4-

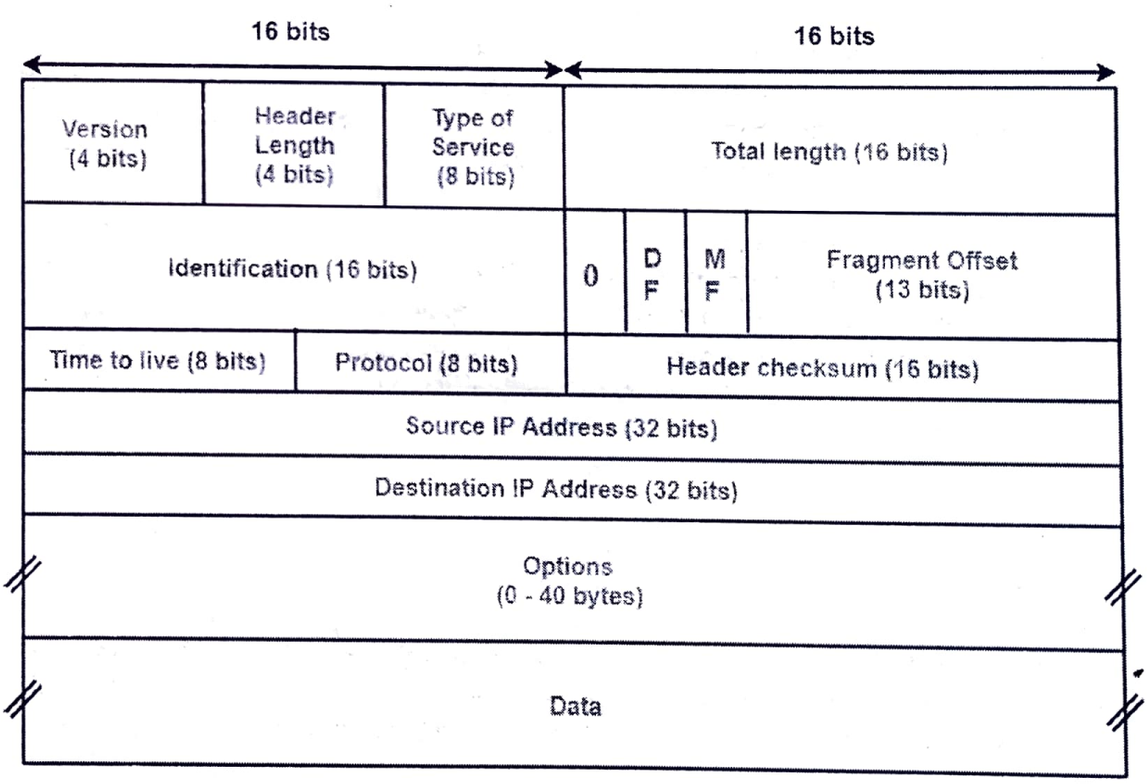
IP – N/W layer

- IPv4 short for Internet Protocol Version 4 is the fourth version of the **Internet Protocol** (IP).
- IP is responsible to deliver data packets from the source host to the destination host.
- This delivery is solely based on the **IP Addresses** in the packet headers.
- IPv4 is the first major version of IP.
- IPv4 is a connectionless protocol for use on **packet-switched networks**.

In this article, we will discuss about IPv4 Header.

IPv4 Header-

The following diagram represents the IPv4 header-



IPv4 Header

Let us discuss each field of IPv4 header one by one.

1. Version-

- Version is a 4 bit field that indicates the IP version used.
- The most popularly used IP versions are version-4 (IPv4) and version-6 (IPv6).
- Only IPv4 uses the above header.
- So, this field always contains the decimal value 4.

NOTES

It is important to note-

- Datagrams belonging to different versions have different structures.
- So, they are parsed differently.
- IPv4 datagrams are parsed by version-4 parsers.
- IPv6 datagrams are parsed by version-6 parsers.

2. Header Length-

- Header length is a 4 bit field that contains the length of the IP header.
- It helps in knowing from where the actual data begins.

Minimum And Maximum Header Length-

The length of IP header always lies in the range-

[20 bytes , 60 bytes]

- The initial 5 rows of the IP header are always used.
- So, minimum length of IP header = 5 x 4 bytes = 20 bytes.
- The size of the 6th row representing the Options field vary.
- The size of Options field can go up to 40 bytes.
- So, maximum length of IP header = 20 bytes + 40 bytes = 60 bytes.

Concept of Scaling Factor-

- Header length is a 4 bit field.
- So, the range of decimal values that can be represented is [0, 15].
- But the range of header length is [20, 60].
- So, to represent the header length, we use a scaling factor of 4.

In general,

Header length = Header length field value x 4 bytes

Examples-

- If header length field contains decimal value 5 (represented as 0101), then-

  Header length = 5 x 4 = 20 bytes

- If header length field contains decimal value 10 (represented as 1010), then-

  Header length = 10 x 4 = 40 bytes

- If header length field contains decimal value 15 (represented as 1111), then-

Header length = 15 x 4 = 60 bytes

NOTES

It is important to note-

- Header length and Header length field value are two different things.
- The range of header length field value is always [5, 15].
- The range of header length is always [20, 60].

While solving questions-

- If the given value lies in the range [5, 15] then it must be the header length field value.
- This is because the range of header length is always [20, 60].

Otherwise actual length

3. Type Of Service-

- Type of service is a 8 bit field that is used for Quality of Service (QoS).
- The datagram is marked for giving a certain treatment using this field.

4. Total Length-

- Total length is a 16 bit field that contains the total length of the datagram (in bytes).

Total length = Header length + Payload length

- Minimum total length of datagram = 20 bytes (20 bytes header + 0 bytes data)
- Maximum total length of datagram = Maximum value of 16 bit word = 65535 bytes

5. Identification-

- Identification is a 16 bit field.
- It is used for the identification of the fragments of an original IP datagram.

When an IP datagram is fragmented,

- Each fragmented datagram is assigned the same identification number.
- This number is useful during the re assembly of fragmented datagrams.
- It helps to identify to which IP datagram, the fragmented datagram belongs to.

6. DF Bit-

- DF bit stands for Do Not Fragment bit.
- Its value may be 0 or 1.

When DF bit is set to 0,

- It grants the permission to the intermediate devices to fragment the datagram if required.

When DF bit is set to 1,

- It indicates the intermediate devices not to fragment the IP datagram at any cost.
- If network requires the datagram to be fragmented to travel further but settings does not allow its fragmentation, then it is discard
- An error message is sent to the sender saying that the datagram has been discarded due to its settings.

Then sender may send again with changed DF bit.

7. MF Bit-

- MF bit stands for More Fragments bit.
- Its value may be 0 or 1.

When MF bit is set to 0,

- It indicates to the receiver that the current datagram is either the last fragment in the set or that it is the only fragment.

When MF bit is set to 1,

- It indicates to the receiver that the current datagram is a fragment of some larger datagram.
- More fragments are following.
- MF bit is set to 1 on all the fragments except the last one.

8. Fragment Offset-

- Fragment Offset is a 13 bit field.
- It indicates the position of a fragmented datagram in the original unfragmented IP datagram.
- The first fragmented datagram has a fragment offset of zero.

Fragment offset for a given fragmented datagram

= Number of data bytes ahead of it in the original unfragmented datagram

data bytes only (not the header)
header size not included

Concept Of Scaling Factor-

- We use a scaling factor of 8 for the fragment offset.
- Fragment offset field value = Fragment Offset / 8

Need Of Scaling Factor For Fragment Offset

- In IPv4 header, the total length field comprises of 16 bits.
- Total length = Header length + Payload length.
- Minimum header length = 20 bytes.
- So, maximum amount of data that can be sent in the payload field = $2^{16} - 20$ bytes.
- In worst case, a datagram containing $2^{16} - 20$ bytes of data might be fragmented in such a way that the last fragmented datagram contains only 1 byte of data.
- Then, fragment offset for the last fragmented datagram will be $(2^{16} - 20) - 1 = 2^{16} - 21 \cong 2^{16}$

(if no scaling factor is used)

- Now, this fragment offset value of 2^{16} can not be represented.
- This is because the fragment offset field consists of only 13 bits.
- Using 13 bits, a maximum number of 2^{13} can be represented.
- So, to represent 2^{16} we use the concept of scaling factor.
- Scaling factor = $2^{16} / 2^{13} = 2^3 = 8$.

9. Time To Live-

- Time to live (TTL) is a 8 bit field.
- It indicates the maximum number of hops a datagram can take to reach the destination.
- The main purpose of TTL is to prevent the IP datagrams from looping around forever in a routing loop.

The value of TTL is decremented by 1 when-

- Datagram takes a hop to any intermediate device having network layer.
- Datagram takes a hop to the destination.

If the value of TTL becomes zero before reaching the destination, then datagram is discarded.

NOTES

It is important to note-

- Both intermediate devices having network layer and destination decrements the TTL value by 1.
- If the value of TTL is found to be zero at any intermediate device, then the datagram is discarded.
- So, at any intermediate device, the value of TTL must be greater than zero to proceed further.
- If the value of TTL becomes zero at the destination, then the datagram is accepted.
- So, at the destination, the value of TTL may be greater than or equal to zero.

10. Protocol-

- Protocol is a 8 bit field.
- It tells the network layer at the destination host to which protocol the IP datagram belongs to.

- In other words, it tells the next level protocol to the network layer at the destination side.
- Protocol number of ICMP is 1, IGMP is 2, TCP is 6 and UDP is 17.

Why Protocol Number Is A Part Of IP Header?

Consider-

- An IP datagram is sent by the sender to the receiver.
- When datagram reaches at the router, it's buffer is already full.

In such a case,

- Router does not discard the datagram directly.
- Before discarding, router checks the next level protocol number mentioned in its IP header.
- If the datagram belongs to TCP, then it tries to make room for the datagram in its buffer.
- It creates a room by eliminating one of the datagrams having lower priority.
- This is because it knows that TCP is a reliable protocol and if it discards the datagram, then it will be sent again by the sender.
- The order in which router eliminate the datagrams from its buffer is-

ICMP > IGMP > UDP > TCP

If protocol number would have been inside the datagram, then-

- Router could not look into it.
- This is because router has only three layers- physical layer, data link layer and network layer.

That is why, protocol number is made a part of IP header.

11. Header Checksum-

- Header checksum is a 16 bit field.
- It contains the checksum value of the entire header.
- The checksum value is used for error checking of the header. (header only)

At each hop,

- The header checksum is compared with the value contained in this field.
- If header checksum is found to be mismatched, then the datagram is discarded.
- Router updates the checksum field whenever it modifies the datagram header.

The fields that may be modified are-

1. TTL
2. Options
3. Datagram Length
4. Header Length
5. Fragment Offset

NOTE

It is important to note-

- Computation of header checksum includes IP header only.
- Errors in the data field are handled by the encapsulated protocol.

Also Read- Checksum

12. Source IP Address-

- Source IP Address is a 32 bit field.
- It contains the logical address of the sender of the datagram.

13. Destination IP Address-

- Destination IP Address is a 32 bit field.
- It contains the logical address of the receiver of the datagram.

14. Options-

- Options is a field whose size vary from 0 bytes to 40 bytes.
- This field is used for several purposes such as-

1. Record route
2. Source routing
3. Padding

1. Record Route-

- A record route option is used to record the IP Address of the routers through which the datagram passes on its way.
- When record route option is set in the options field, IP Address of the router gets recorded in the Options field.

NOTE

The maximum number of IPv4 router addresses that can be recorded in the Record Route option field of an IPv4 header is 9.

9

Explanation-

- In IPv4, size of IP Addresses = 32 bits = 4 bytes.
- Maximum size of Options field = 40 bytes.
- So, it seems maximum number of IP Addresses that can be recorded = 40 / 4 = 10.
- But some space is required to indicate the type of option being used.
- Also, some space is to be left between the IP Addresses.
- So, the space of 4 bytes is left for this purpose.
- Therefore, the maximum number of IP addresses that can be recorded = 9.

2. Source Routing-

- A source routing option is used to specify the route that the datagram must take to reach the destination.
- This option is generally used to check whether a certain path is working fine or not.
- Source routing may be loose or strict.

3. Padding-

- Addition of dummy data to fill up unused space in the transmission unit and make it conform to the standard size is called as padding.
- Options field is used for padding.

Example-

- When header length is not a multiple of 4, extra zeroes are padded in the Options field.
- By doing so, header length becomes a multiple of 4.
- If header length = 30 bytes, 2 bytes of dummy data is added to the header.
- This makes header length = 32 bytes.
- Then, the value 32 / 4 = 8 is put in the header length field.
- In worst case, 3 bytes of dummy data might have to be padded to make the header length a multiple of 4.

Also Read- TCP Header | UDP Header

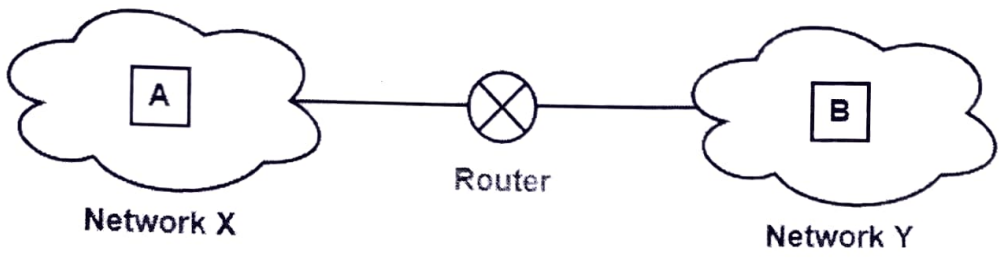
Fragmentation

IP Fragmentation | Fragmentation in Networking

Computer Networks

IP Fragmentation-

- IP Fragmentation is a process of dividing the datagram into fragments during its transmission.
- It is done by intermediary devices such as routers at the destination host at network layer.



Need-

- Each network has its maximum transmission unit (MTU).
- It dictates the maximum size of the packet that can be transmitted through it.
- Data packets of size greater than MTU can not be transmitted through the network.
- So, datagrams are divided into fragments of size less than or equal to MTU.

See (MTU, MSS, Segmn, Fragmn)
IP MTU
TCP MSS

Datagram Fragmentation-

When router receives a datagram to transmit further, it examines the following-

- Size of the datagram
- MTU of the destination network
- DF bit value in the IP header

Then, following cases are possible-

Case-01:

- Size of the datagram is found to be smaller than or equal to MTU.
- In this case, router transmits the datagram without any fragmentation.

Case-02:

- Size of the datagram is found to be greater than MTU and DF bit set to 1.
- In this case, router discards the datagram.

Case-03:

- Size of the datagram is found to be greater than MTU and DF bit set to 0.

- In this case, router divides the datagram into fragments of size less than or equal to MTU.
- Router attaches an IP header with each fragment making the following changes in it.
- Then, router transmits all the fragments of the datagram.

Changes Made By Router-

Router makes the following changes in IP header of each fragment-

- It changes the value of total length field to the size of fragment.
- It sets the MF bit to 1 for all the fragments except the last one.
- For the last fragment, it sets the MF bit to 0.
- It sets the fragment offset field value.
- It recalculates the header checksum.

Also Read- IPv4 Header

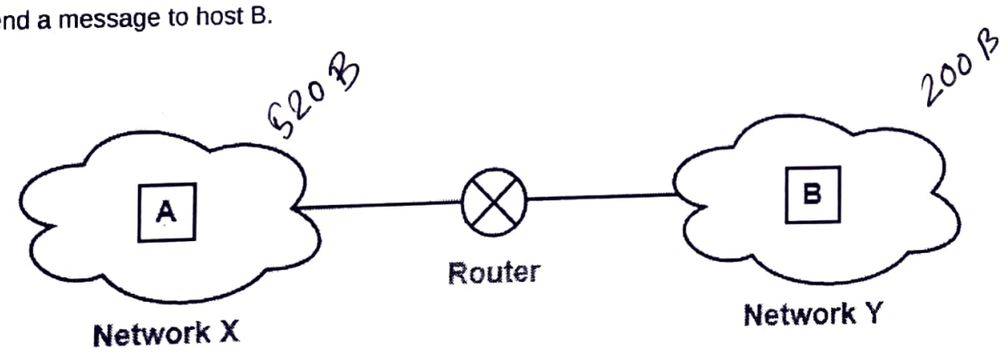
IP Fragmentation Examples-

Now, lets us discuss some examples of IP fragmentation to understand how the fragmentation is actually carried out.

Example-01:

Consider-

- There is a host A present in network X having MTU = 520 bytes.
- There is a host B present in network Y having MTU = 200 bytes.
- Host A wants to send a message to host B.



Consider router receives a datagram from host A having-

- Header length = 20 bytes
- Payload length = 500 bytes
- Total length = 520 bytes
- DF bit set to 0

Now, router works in the following steps-

Step-01:

Router examines the datagram and finds-

- Size of the datagram = 520 bytes
- Destination is network Y having MTU = 200 bytes
- DF bit is set to 0

Router concludes-

- Size of the datagram is greater than MTU.
- So, it will have to divide the datagram into fragments.
- DF bit is set to 0.
- So, it is allowed to create fragments of the datagram.

Step-02:

Router decides the amount of data that it should transmit in each fragment.

Router knows-

- MTU of the destination network = 200 bytes.
- So, maximum total length of any fragment can be only 200 bytes.
- Out of 200 bytes, 20 bytes will be taken by the header.
- So, maximum amount of data that can be sent in any fragment = 180 bytes.

Router uses the following rule to choose the amount of data that will be transmitted in one fragment-

RULE

The amount of data sent in one fragment is chosen such that-

- It is as large as possible but less than or equal to MTU.
- It is a multiple of 8 so that pure decimal value can be obtained for the fragment offset field.

NOTE

- It is not compulsory for the last fragment to contain the amount of data that is a multiple of 8.
- This is because it does not have to decide the fragment offset value for any other fragment.

Following the above rule,

- Router decides to send maximum 176 bytes of data in one fragment.
- This is because it is the greatest value that is a multiple of 8 and less than MTU.

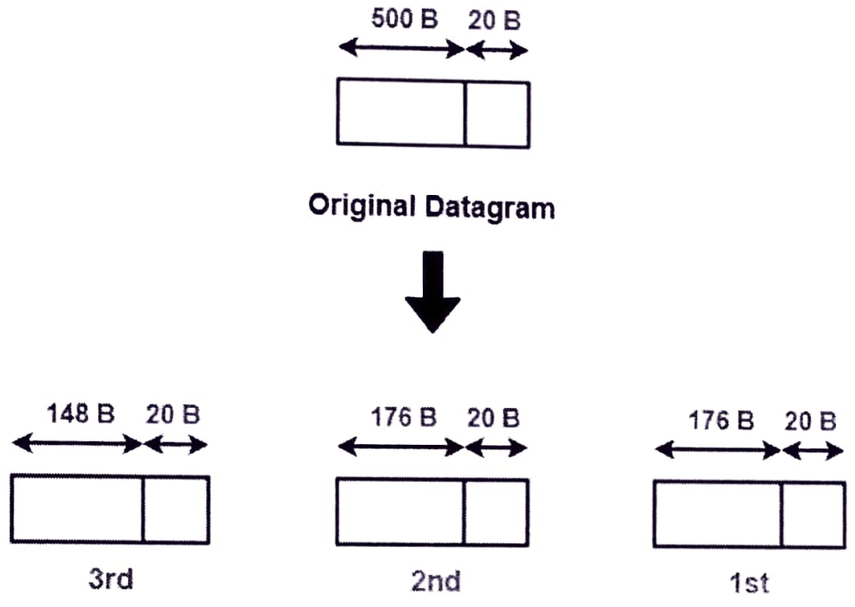
180 B % 8 ≠ 0
✓ 176 % 8 = 0

(Max data sent in a fragment of size = MTU)

Step-03:

Router creates three fragments of the original datagram where-

- First fragment contains the data = 176 bytes
- Second fragment contains the data = 176 byes
- Third fragment contains the data = 148 bytes



Fragments of Original Datagram

The information contained in the IP header of each fragment is-

Header Information Of 1st Fragment-

- Header length field value = 20 / 4 = 5
- Total length field value = 176 + 20 = 196
- MF bit = 1
- Fragment offset field value = 0
- Header checksum is recalculated.
- Identification number is same as that of original datagram.

Header Information Of 2nd Fragment-

- Header length field value = 20 / 4 = 5
- Total length field value = 176 + 20 = 196
- MF bit = 1
- Fragment offset field value = 176 / 8 = 22 (Remember to ÷ by 8)
- Header checksum is recalculated.

- Identification number is same as that of original datagram.

Header Information Of 3rd Fragment-

- Header length field value = 20 / 4 = 5
- Total length field value = 148 + 20 = 168
- MF bit = 0
- Fragment offset field value = (176 + 176) / 8 = 44
- Header checksum is recalculated.
- Identification number is same as that of original datagram.

Router transmits all the fragments.

Step-04:

At destination side,

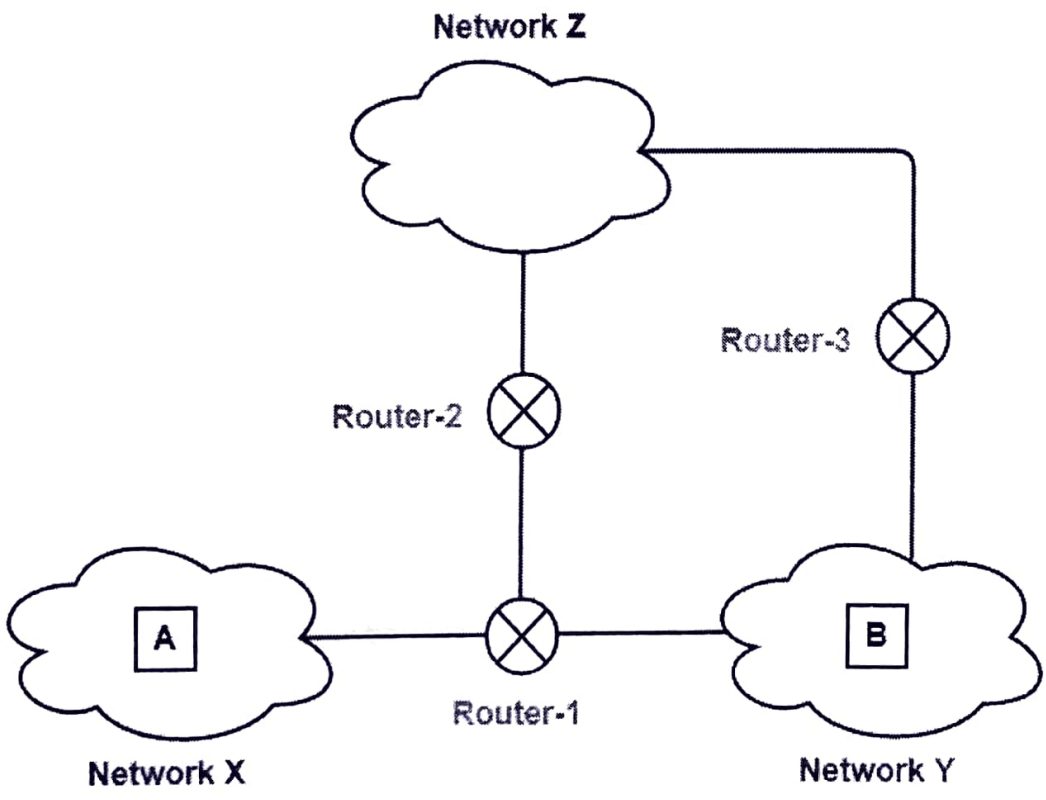
- Receiver receives 3 fragments of the datagram.
- Reassembly algorithm is applied to combine all the fragments to obtain the original datagram.

(using identification number)

Example-02:

Consider-

- There is a host A present in network X having MTU = 520 bytes.
- There is a host B present in network Y having MTU = 200 bytes.
- There exists a network Z having MTU = 110 bytes.
- Host A wants to send a message to host B.



Consider Router-1 receives a datagram from host A having-

- Header length = 20 bytes
- Payload length = 500 bytes
- Total length = 520 bytes
- DF bit set to 0

Consider Router-1 divides the datagram into 3 fragments as discussed in Example-01.

Then,

- First fragment contains the data = 176 bytes
- Second fragment contains the data = 176 byes
- Third fragment contains the data = 148 bytes

Now, consider-

- First and third fragment reaches the destination directly.
- However, second fragment takes its way through network Z and reach the destination through Router-3.

Journey Of Second Fragment-

Now, let us discuss the journey of fragment-2 and how it finally reaches the destination.

Router-2 receives a datagram (second fragment of original datagram) where-

- Header length = 20 bytes
- Payload length = 176 bytes
- Total length = 196 bytes
- DF bit set to 0

Now, Router-2 works in the following steps-

Step-01:

Router-2 examines the datagram and finds-

- Size of the datagram = 196 bytes
- Destination is network Z having MTU = 110 bytes
- DF bit is set to 0

Router-2 concludes-

- Size of the datagram is greater than MTU.
- So, it will have to divide the datagram into fragments.
- DF bit is set to 0.
- So, it is allowed to create fragments of the datagram.

Step-02:

Router-2 decides the amount of data that it should transmit in each fragment.

Router-2 knows-

- MTU of the destination network = 110 bytes.
- So, maximum total length of any fragment can be only 110 bytes.
- Out of 110 bytes, 20 bytes will be taken by the header.
- So, maximum amount of data that can be sent in any fragment = 90 bytes.

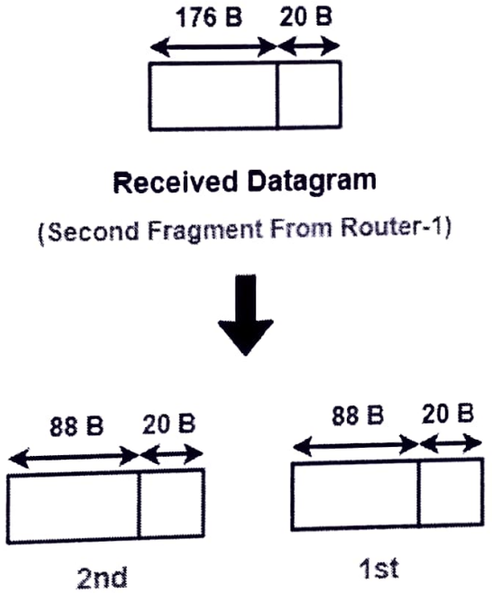
Following the rule,

- Router-2 decides to send maximum 88 bytes of data in one fragment.
- This is because it is the greatest value that is a multiple of 8 and less than MTU.

Step-03:

Router-2 creates two fragments of the received datagram where-

- First fragment contains the data = 88 bytes
- Second fragment contains the data = 88 byes



Fragments of Received Datagram

The information contained in the IP header of each fragment is-

Header Information Of 1st Fragment-

- Header length field value = 20 / 4 = 5
- Total length field value = 88 + 20 = 108
- MF bit = 1
- Fragment offset field value = 176 / 8 = 22
- Header checksum is recalculated.
- Identification number is same as that of original datagram.

NOTE-

- This fragment is **NOT** the first fragment of the original datagram.
- It is the first fragment of the datagram received by Router-2.
- The datagram received by Router-2 is the second fragment of the original datagram.
- This datagram will serve as the second fragment of the original datagram.
- Therefore, fragment offset field is set according to the first fragment of the original datagram.

2nd

Header Information Of 2nd Fragment-

- Header length field value = 20 / 4 = 5
- Total length field value = 88 + 20 = 108
- MF bit = 1
- Fragment offset field value = (176 + 88) / 8 = 33
- Header checksum is recalculated.
- Identification number is same as that of original datagram.

NOTE-

- This fragment is **NOT** the last fragment of the original datagram.
- It is the last fragment of the datagram received by Router-2.
- The datagram received by Router-2 is the second fragment of the original datagram.
- This datagram will serve as the third fragment of the original datagram.
- There is another fragment of the original datagram that follows it.
- That is why, here MF bit is not set to 0.

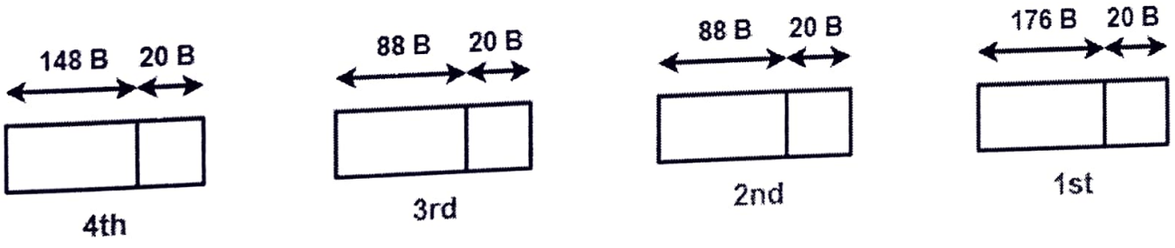
Router-2 transmits both the fragments which reaches the destination through Router-3.

Router-3 performs no fragmentation.

Step-04:

At destination side,

- Receiver receives 4 fragments of the datagram.
- Reassembly algorithm is applied to combine all the fragments to obtain the original datagram.



Fragments Received By Receiver

Reassembly Algorithm-

Receiver applies the following steps for reassembly of all the fragments-

1. It identifies whether datagram is fragmented or not using MF bit and Fragment offset field.
2. It identifies all the fragments belonging to the same datagram using identification field.
3. It identifies the first fragment. Fragment with offset field value = 0 is the first fragment.
4. It identifies the subsequent fragments using total length, header length and fragment offset.
5. It repeats step-04 until MF bit = 0.

Fragment Offset field value for the next subsequent fragment

= (Payload length of the current fragment / 8) + Offset field value of the current fragment

= (Total length – Header length / 8) + Offset field value of the current fragment

Fragmentation Overhead-

- Fragmentation of datagram increases the overhead.
- This is because after fragmentation, IP header has to be attached with each fragment.

Total Overhead

= (Total number of fragmented datagrams – 1) x size of IP header

Efficiency = Useful bytes transferred / Total bytes transferred

OR

Efficiency = Data without header / Data with header

Bandwidth Utilization or Throughput = Efficiency x Bandwidth

Important Notes-

Note-01:

- Source side does not require fragmentation due to wise segmentation by transport layer.
- The transport layer looks at the datagram data limit and frame data limit.
- Then, it performs segmentation in such a way that the resulting data can easily fit in a frame.
- Thus, there is no need of fragmentation at the source side.

Note-02:

- Datagrams from the same source to the same destination may take different routes in the network.

IP

In

I

T

Fragmentation (packets) → N/W layer

Segmentation (datagram) → Transport layer

Note-03:

- Fragment offset field value is set to 0 for the first fragmented datagram.
- MF bit is set to 1 for all the fragmented datagrams except the last one.

Note-04:

- Unique combinations of MF bit value and fragment offset value.

MF bit	Offset value	Represents
1	0	1st Fragment
1	!=0	Intermediate Fragment
0	!=0	Last Fragment
0	0	No Fragmentation

Note-05:

- Identification number for all the fragments is same as that of the original datagram.
- This is to identify all the fragments of the same datagram while re-assembling them.

Note-06:

- Consider datagram goes through a path where different intermediaries having different bandwidths.
- Then, while calculating the throughput, consider the minimum bandwidth since it act as a bottleneck.

Note-07:

- Fragmentation is done by intermediary devices such as routers.
- The reassembly of fragmented datagrams is done only after reaching the destination.

Note-08:

Reassembly is not done at the routers because-

- All the fragments may not meet at the router.
- Fragmented datagrams may reach the destination through independent paths.
- There may be a need for further fragmentation.

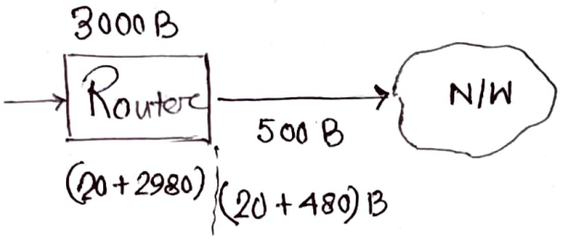
Note-09:

If a fragment (say parent) is re fragmented into multiple datagrams then-
1. The fragment offset value for the first re-fragment is always same as its parent.
2. The MF bit bit value for the last re-fragment is always same as its parent.

(see example 2)

Q. A datagram of 3000B (20B of IP header + 2980 B of IP payload) reached at router & must be forwarded to link with MTU of 500 B. How many fragments will be generated & also find MF, offset, total length value for all.

→



$\left\lceil \frac{2980}{480} \right\rceil = 7$ fragments.

$2980 - 480 \times 6 = 100$

P_7	P_6	P_5	P_4	P_3	P_2	P_1	
100 + 20 = 300	480 + 20 = 500	480 + 20 = 500	480 + 20 = 500	480 + 20 = 500	480 + 20 = 500	480+20 = 500	TL
0	1	1	1	1	1	1	MF
360	300	240	$\frac{3\times480}{8} = 180$	$\frac{2\times480}{8} = 120$	$\frac{480}{8} = 60$	0	Offset

MTU, MSS, Fragmentation, Segmentation

The MTU is the Maximum IP packet size for a given link. Packets bigger than the MTU are fragmented at the point where the lower MTU is found and reassembled further down the chain.
If no fragmentation is wanted, either you have to check the MTU at each hop or use a helper protocol for that (Path MTU Discovery).
Note that IPv6 does NOT support packet fragmentation by routers, hence PMTUD with ICMPv6 is mandatory if you don't want to lose a packet somewhere because of small MTU. Endpoints can fragment, but not routers Also, IPv6 has a much higher MINIMUM MTU.
MSS is Maximum TCP segment Size. Unlike MTU, packet exceeding MSS aren't fragmented, they're simply discarded. MSS is normally decided in the TCP three-way handshake, but some setup might yield path where the decided upon MSS is still too big, leading to dropped packets. The MSS isn't negotiated packet per packet, but for a complete TCP session, nor does it take into account TCP/IP headers

When using PPPoE, all the overhead means you needs to reduce the MSS on the way, normally by specifying it on the router where the chokepoint is found, which will then replace the MSS of passing threeway handshake by the correct lower value if it's higher. PPPoE is simply adding 8 bytes (6 bytes PPPoE + 2 bytes PPP) on top of everything (IP+TCP) and is meant to be run over Ethernet at 1500 bytes MTU, hence the 1492 MSS normally configured to make it go through.
Your IP stack will chop off data to be sent up to the MSS, put it in a TCP segment, then put it in one or more IP packets (depending if it's bigger than local MTU settings) before sending it. Intermediate router could chop it down further if they have lower MTU, but they're only affecting the IP Packet itself, not playing into the TCP segment/header.

The MTU is the Maximum IP packet size for a given link . Packets greater in size than the MTU is fragmented at the point just where the lower MTU is found and reassembled further down the chain .
MSS is Maximum TCP segment Size . Unlike MTU , packet greater than MSS aren't fragmented , they're simply just discarded . MSS is usually made a decision in the TCP three-way handshake , however some setup might yield path where the decided upon MSS is still too big , leading to dropped packets . The MSS isn't negotiated packet per packet , but for a complete TCP session , nor does it take into account TCP/IP headers
The IP stack will chop off data to be sent up to the MSS , put it in a TCP segment , then put it in one or more IP packets (based on if it's bigger than local MTU settings) before sending it . Intermediate router could chop it down further if they have lower MTU , however they're only affecting the IP Packet itself , not playing into the TCP segment/header .
e .g . When you use PPPoE , all the overhead will mean you needs to reduce the MSS on the way , normally by specifying it on the router where the chokepoint is found , which will then replace the MSS of passing threeway handshake by the correct lower value if it's higher . PPPoE is just adding 8 bytes (6 bytes PPPoE + 2 bytes PPP) on the top of everything (IP+TCP) and is intended to be run over Ethernet at 1500 bytes MTU , therefore the 1492 MSS normally configured to make it go through .
If no fragmentation is wanted , either you will have to check the MTU at each hop or use a helper protocol for that (PMTUD)
Note that IPv6 does NOT support packet fragmentation by routers , therefore PMTUD with ICMPv6 is mandatory in the event that don't want to loose packet somewhere because of the small MTU . Endpoints can fragment , but not routers Also , IPv6 has a higher MINIMUM MTU .

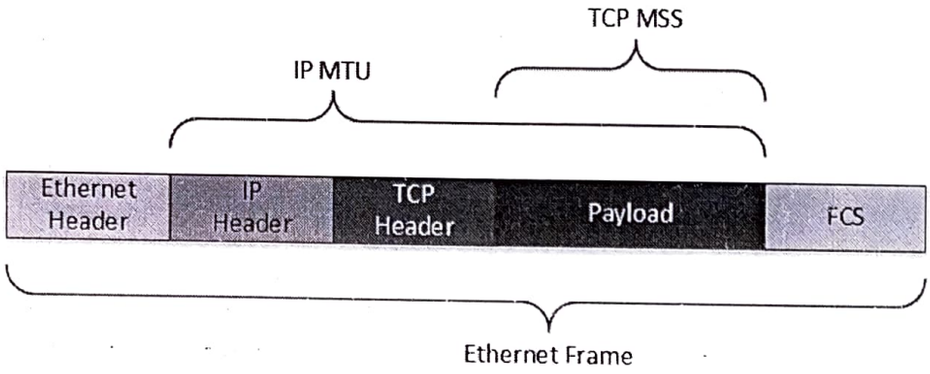
What is MTU?

The MTU, or 'Maximum Transmission Unit', is the largest block of data that can be handled at layer-3 of the OSI model. This is usually IP, so the MTU usually refers to the maximum size a packet can be.

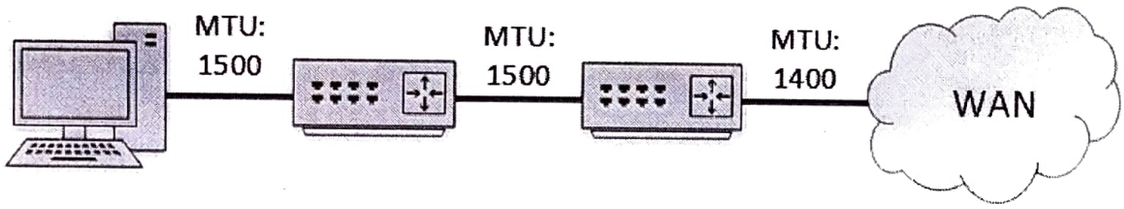
Where Does This Limit Come From?

The limit at layer-3 comes directly from layer-2. Layer-2 uses frames, and each frame has a maximum size limit.

The Ethernet standard, for example, sets a maximum frame size at 1518 bytes. Ethernet headers are 18 bytes long, leaving 1500 bytes for the packet to use. Therefore, the packet's MTU is 1500 bytes.
We don't always use Ethernet as our layer-2 protocol. Many WAN standards, like PPPoE, use different frame sizes. This results in a different MTU for the packet.
In this article, we'll assume Ethernet unless specified otherwise.
Some devices support Jumbo Frames. Newer Ethernet standards support frames up to around 9000 bytes. This is more common in campuses and the data centre, but still rare in the WAN.
For the rest of this article, we'll talk about the non-Jumbo frame limit of 1500 bytes.
The entire packet needs to fit into the MTU limit. This includes the IP headers as well as the payload.
TCP has a limit called Maximum Segment Size, or MSS. This is the size of the layer-4 payload (without the IP and TCP headers).
IP and TCP headers usually add up to 40 bytes in total. So, if we started with an MTU of 1500 bytes, we now have an MSS of 1460 bytes.



Fragmentation
A host will definitely know the MTU of its own connection to the network. But, it has no way of knowing the MTU of a link further up the path.
For example, a computer's NIC may have an MTU of 1500. However, the connection to the WAN has an MTU of 1400.



This could leave us in a sticky situation. The workstation may send a 1500 byte packet. The router connected to the WAN would be unable to send the packet, as the packet is larger than the 1400 byte MTU.
The solution is fragmentation. When a packet is larger than the MTU, a device (often a router) will break the packet into smaller fragments.
Each of these fragments is still a packet, just smaller than the original. The packets move along the path to the destination just like normal. The destination device reassembles the fragments into the original packet and processes it as normal.
This might sound like a wonderful solution, but it comes with its drawbacks. For one, each fragment has duplicated IP headers, which results in more traffic begin sent.
Also, this adds processing overhead on the router that fragments the packets, and on the destination that reassembles them.
And what if one fragment goes missing or becomes corrupted? We need to resend the entire packet, not just the fragment.
Firewalls may also have problems with fragments. Some of them will drop fragments if they arrive out of order, seeing them as unsolicited traffic.

TIP: Where possible, avoid fragmentation

What Can Cause Fragmentation?

As we've already mentioned, some WAN links use a smaller MTU. But there are other causes as well.
Tunnels, such as a GRE tunnel, will add additional headers to a packet. This may push the packet size over the frame limit and cause fragmentation.
Encryption, IPSec for example, may also add additional headers, with the same effect.
So what do we do about this? In the case of a tunnel (with or without encryption), we would manually lower the MTU on the tunnel's interface.
We're effectively restricting the size of the payload, so the payload plus the additional headers do not go over the frame limit.

But that still leaves us with fragmentation. A workstation doesn't know about the smaller MTU size of the tunnel, sends large packets, and fragmentation occurs. This is not ideal.
We now need to think about how to handle this. One option might be to manually set the MTU on each workstation. That's a very big job.
Another option is to use Path MTU Discovery (PMTUD).

PMTUD
PMTUD is a method of dynamically discovering low MTU's along a path. This works by setting the 'Don't Fragment' (DF) bit in the IP header of each packet.
Setting this bit tells devices that fragmentation is not allowed for this packet. If it's too large, and this bit is set, delivery is not possible and the device drops the packet.
The device that dropped the packet will send an ICMP 'Destination Unreachable (Fragmentation was Needed and DF was set)' message back to the sender.
When the sender gets this message, it knows that the packets it is sending are too large for the path.

TIP: Don't block ICMP type 3 code 4 messages! These are needed for PMTUD to work!

The ICMP message does not tell the sender what the MTU should be. The sender needs to lower it's MTU for this connection and try again. It will repeat the process until the packet is small enough that these messages are not generated.
PMTUD is only supported by TCP and UDP
PMTUD is not just done when a connection is initially set up. It happens continually. This is because network paths may change, so MTU's may change. This allows devices to adapt to changes in the network.

It also works independently in both directions. Imagine a case where a client is requesting HTTP from a server. The client will usually send small request packets to the server, not triggering PMTUD. The server would likely send larger packets back, which may trigger PMTUD.
Another case where we see this is with asymmetric routing. Different paths are used in different directions, which may have different MTU's.

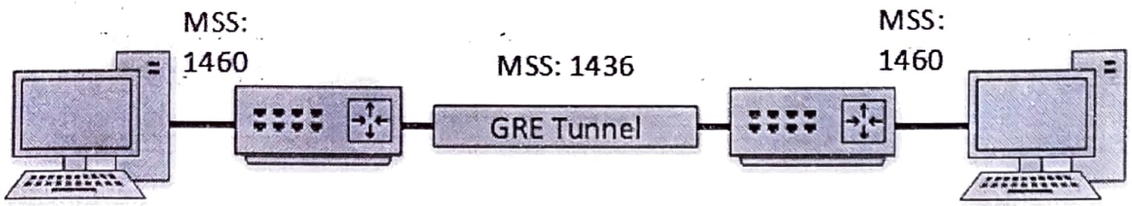
TCP MSS
Another option is to tune the MSS. As mentioned earlier, the MSS is like the MTU, but used with TCP at layer 4. Put simply, the MSS is the maximum size that the payload can be, after subtracting space for the IP, TCP, and other headers. So, if the MTU is 1500 bytes, and the IP and TCP headers are 20 bytes each, the MSS is 1460 bytes.

While establishing a new TCP connection, a three-way handshake is performed. Each device inserts its MSS into TCP headers, so in this sense, it's announcing its MSS to the remote device.
The remote device will see the MSS, and if necessary it will adjust the payload size that it uses when it uses this connection.

It's important to note here that hosts are simply announcing their MSS. There is no negotiation of mutually acceptable values. They simply say 'This is the largest TCP payload I can handle'. It's up to the remote end to honour this.

Interim Devices

The MSS is based on the MTU. We face a similar problem as before. A device will only know the MTU, and therefore the MSS, of its local link. They will not know about a lower MSS on a link somewhere along the path. Think of a network with a GRE tunnel for example. Each host will have an MSS of 1460. However, the tunnel has extra headers, lowering its MSS to 1436 bytes.



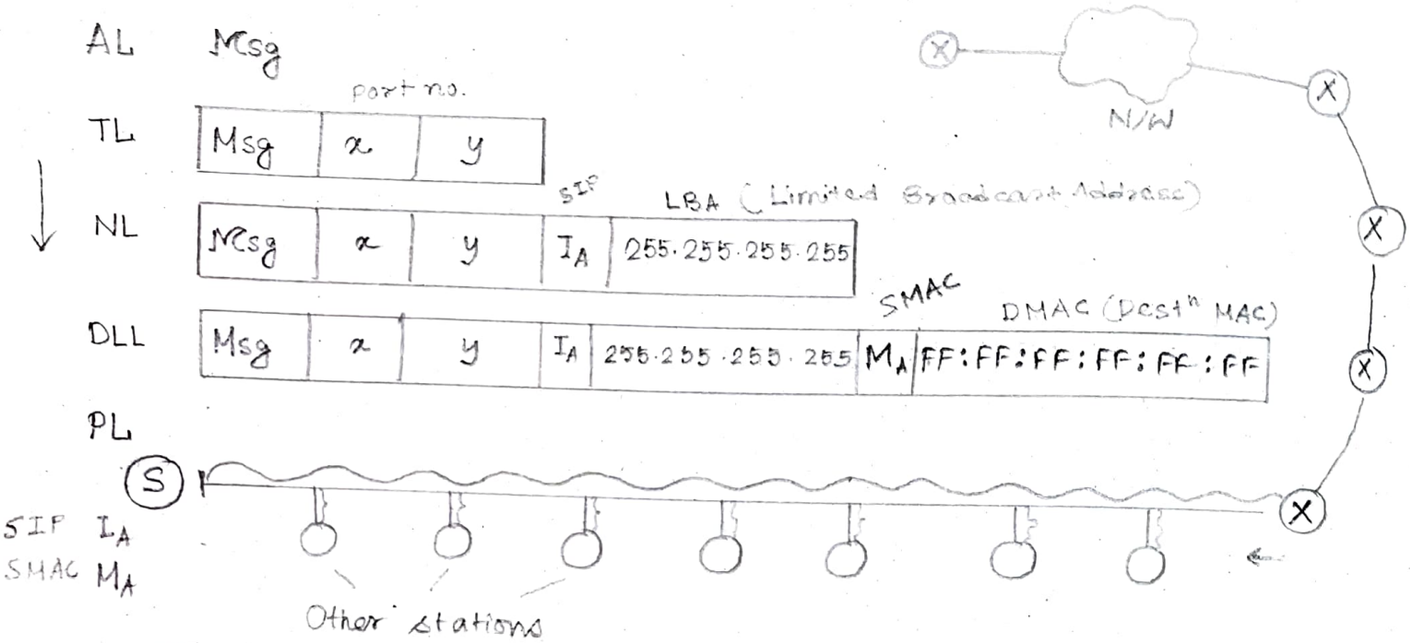
How can we work with cases like this? We can configure the routers to rewrite the MSS.
A Cisco router, for example, will have this command configured on the tunnel:
ip tcp adjust mss 1436

When TCP traffic, such as the three-way handshake, passes across the tunnel, the router will see the MSS is set to 1460 in the TCP header.
Knowing that the MSS of the tunnel is lower than this, it will rewrite this to 1436.
Now, the host at the other end will see the MSS for the path, and adjust the maximum payload for this connection accordingly.
It's important to realise that this is a feature of TCP. This won't work with UDP or other traffic types.
UDP needs to fall back on other methods like PMTUD, manual MTU adjustments, or fragmentation.

https://www.cisco.com/c/en/us/support/docs/ip/generic-routing-encapsulation-gre/25885-pmtud-ipfrag.html

Protocols and Concepts @ Network Layer

* Implementation of Broadcasting



- There exists broadcasting of I/P addresses at N/W layer and broadcasting of MAC addresses at data link layer.

- Limited broadcasting -

  255.255.255.255

  FF:FF:FF:FF:FF:FF

  Sending message to everyone in the same N/W.

- Directed broadcasting -

  11.0.0.0 N/W ID

  ↓

  11.255.255.255.

  Sending message to everyone in different N/W

- In broadcasting we don't send message to any process, we send the message to hosts.

- According to the rules of networking, all the stations before a router have to accept the message, when broadcasted message

is sent by anyone.

- Every N/W is bounded by a router. In limited broadcasting, the router will read the broadcasted message, but never forward it.

- Broadcasting done at datalink layer will never cross the network boundaries. Therefore, there is no broadcasting at N/W layer. Even ~~thro~~ though, there is broadcasting at N/W layer, it has to take support from data link layer.

→ In case of directed broadcasting, in N/W layer we put the DBA. Depending on the N/W ID the packet will be given to the router first. So, in DLL, there will be router's MAC address as the destination MAC address. The router will route the packet as a normal unicast packet.

The packet goes through all the routers and when the router of the directed broadcast address is found, it is changed to limited broadcast address.

NAT (N/W Address Translation)

→ Issues with IPv4 addressing : i) address space limited (while # devices increasing)

ii) a large no. of addresses are wasted or remain unutilized (class D or E).

⇒ Solution : Make the address reusable, leveraging on the fact that all users will never connect to the internet at the same time.

→ In NAT, we divide addresses into reusable (private) & non-reusable (public) blocks.

Translate internal (private) addresses to external (public) addresses.

Hide internal machines from external devices.

Allow internet access to large no. of users via few public addresses.

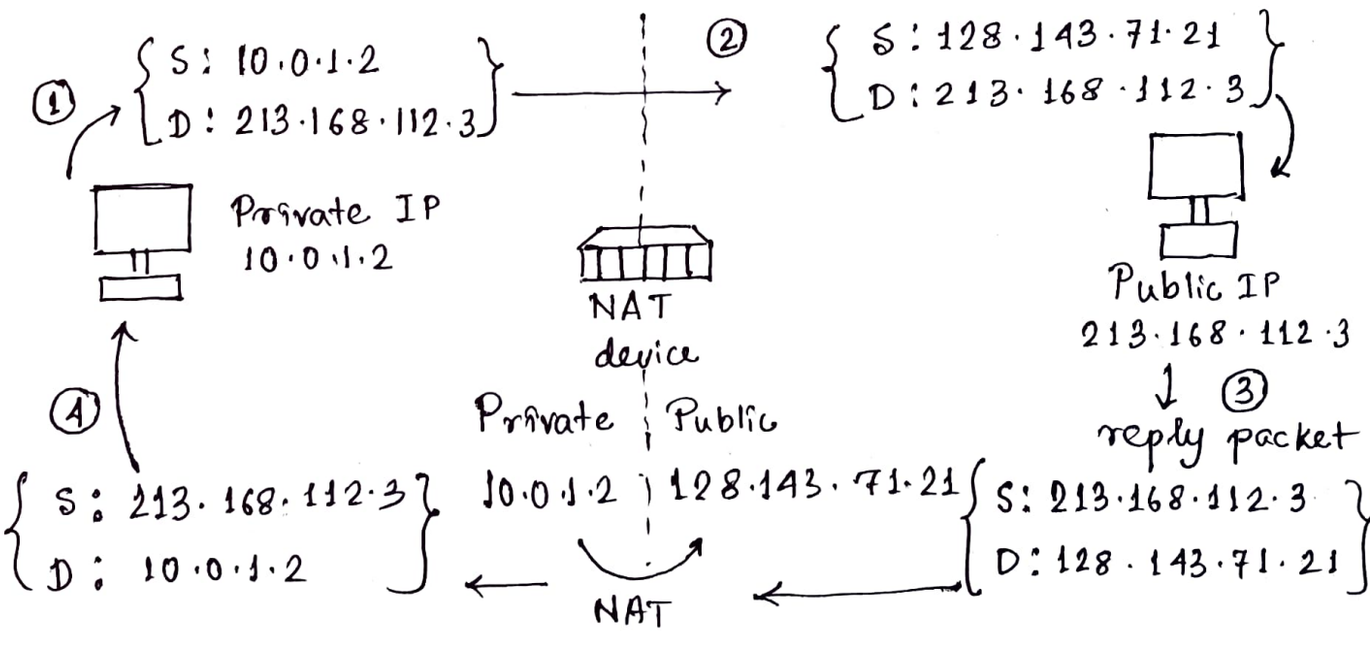
- IPv4 private IP addresses :-

10.0.0.0 – 10.255.255.255

172.16.0.0 – 172.32.255.255

192.168.0.0 – 192.168.255.255.

→



→ Organizations manage internal private N/W.
NAT boxes manage a pool of public IP addresses.

→ An org. can connect to multiple ISPs for better reliability. NAT box can be configured to use alternative ISPs in case of a failure. (Migration b/w ISPs)

→ IP masquerading: Single public IP address is mapped to multiple hosts.
(using modification of port addresses)

→ In case of broadcasting, IP address is not going to help. Internet does not have the concept of broadcasting, only LANs have. or support

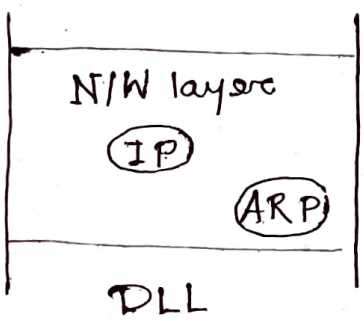
Support from the MAC addresses at DLL is needed.

* ARP (Address Resolution Protocol). → Know other host's MAC Important.

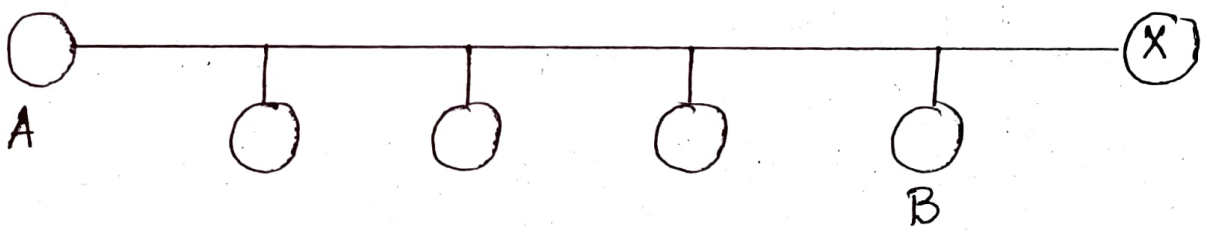
IP → MAC

Purpose of ARP is - given an IP address for a station, we get the MAC address for it.

Finding out MAC address of destination.

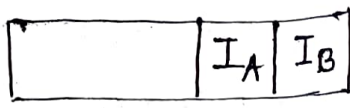


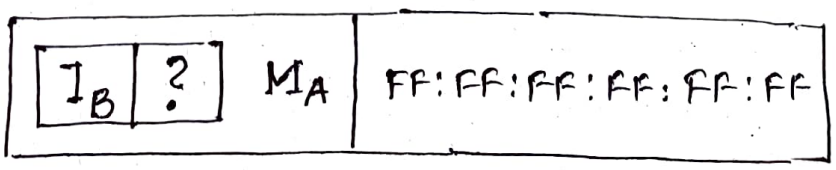
IP will be given to ARP. From ARP it will goto DLL.



Assume, A wants to send data to B. A comes to know that B is in the same N/W by looking at the subnet mask.

ARP takes the IP of the destination & creates ARP request packet —



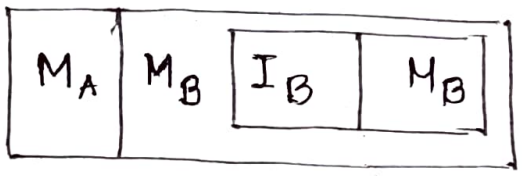


ARP req. packet (broadcast msg)

ARP request packet is sent to every one in

the network, but only B accepts it. Now B sends its MAC address M_B to A through unicast.

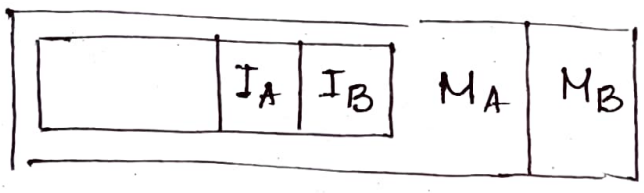
ARP reply



sent to A.

Now, A knows M_B

Now, main packet -



→ When destination is not in the same N/W, ARP finds out that the station (destination) is not in the same N/W using subnet mask and sends to router. ARP asks for the MAC address of the router.

→ ARP request is broadcasting.

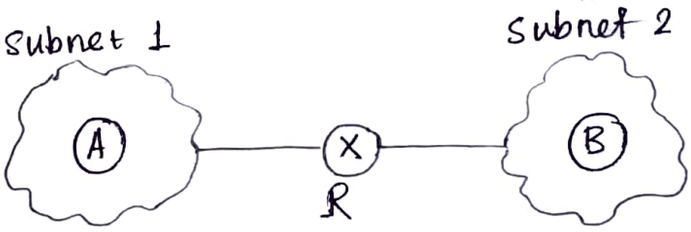
FF:FF:FF:FF:FF:FF

ARP reply is unicast.

→ Cases where ARP is applied -

i) Host wants to find out MAC address of host.
ii) Host " router.
iii) Router " host.
iv) Router " router.

* ARP working for different subnets:



1. Using B's IP & the routing table, A comes to know B is on different N/W.
2. A broadcasts an ARP request to all devices on subnet 1, composed by a query with IP of R router.
3. Having the matching IP, R sends ARP reply (unicast) to A, which includes MAC of R.
4. A transmits IP packet to R using MAC of R.
5. R forwards the packet to B (R might send an ARP request to identify MAC of B).

Relay agent

RARP + Centralised ARP table = BOOTP

BOOTP + Dynamic table = DHCP.

We provide IPs to the hosts when they need to go online or when they don't have memory to save IP, then we provide IP to it.

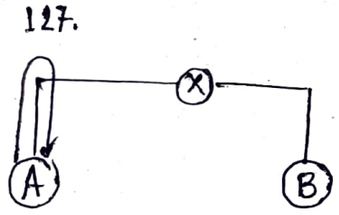
→ If source & destination are not in the same network, they donot know about each other's MAC address. (MAC dropped at N/W, default router)

In order to trace, we can have help from the router (default gateway) which saves mappings from IP addresses to MAC addresses. Then we can track the device. This, is why it's illegal to change the MAC address.

* Special Address 127 (Localhost / 127.0.0.1)

→ Testing self connectivity : If a station A sends a packet to itself & receives it properly, that means the NIC of A is working properly.

→ If we have to check whether a station is connected to itself or not, we use loop-back address 127._._._ (except 0.0.0 & 255.255.255). It makes the station to send packet to itself (packet comes down to DLL and gets back to application layer). It also tests whether all the layers are working or not.

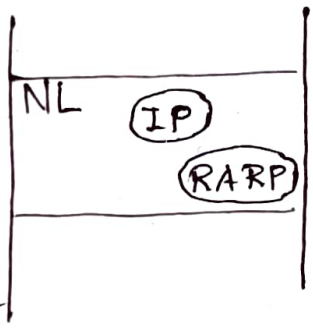


ping 127.0.0.1

* RARP (Reverse ARP) — MAC → IP

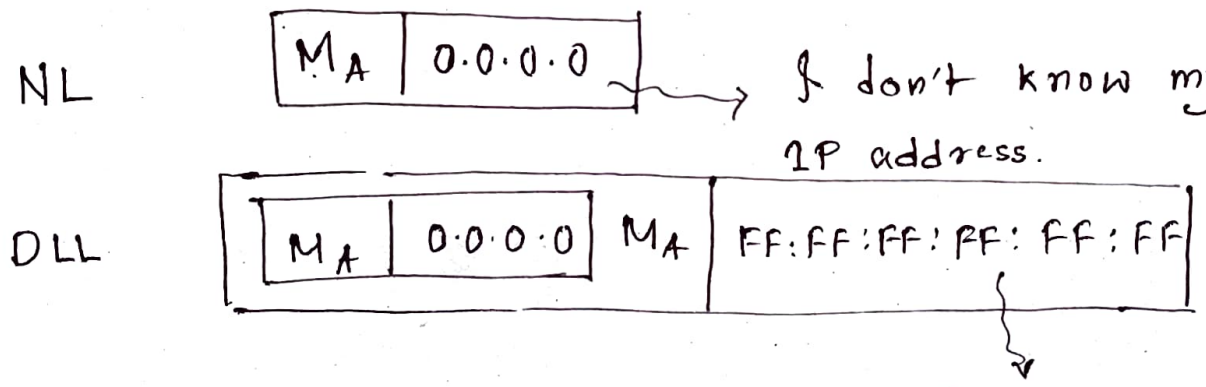
Many hosts do not have hard disk. All the files are saved on the central file system (NFS – N/W file server).

The hosts do not remember anything except the MAC on NIC. When a new host is added on the N/W, RARP is used to know its ip address.

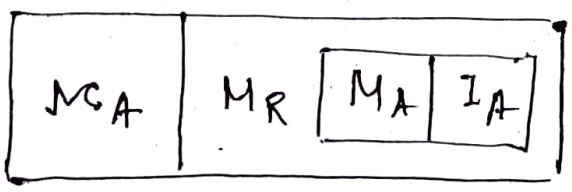


RARP server remembers what IP address is given to which MAC using a table (mapping table).

To know the position of RARP server, a RARP request packet is produced at NL. In DLL, we broadcast as we don't know dest^{h} MAC.



Now, only RARP server will reply using the table. It is unicast as RARP server already knows who's asking.



✓ Disadvantage of RARP is if we have more than 1 networks for a router, we need an RARP server for every N/W.

RARP is used by a client machine in a LAN to request its IP from the gateway - router's ARP table (mapping of MAC & IP address).

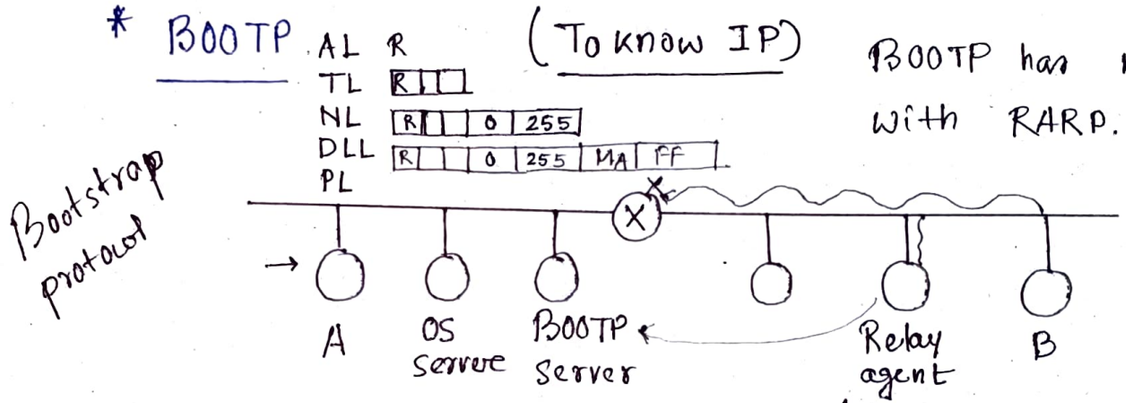
This will lead to distributed information. This causes various discreapancies. (IP address conflict.)

Another demerit of RARP is the static table. (#IP addresses ≥ active #hosts)

→ Acc. to RARP, every N/W should have an RARP server.

→ RARP is obsolete.

* BOOTP (To know IP)



BOOTP has minor differe with RARP.

RARP runs at N/W layer, but BOOTP runs at Application layer.

If any client wants to know the ip address, it is going to send the broadcast request as usu

- Bootp has fixed port no. In the transport layer, the port no. is attached to the request packet. The operating system provides with the port number.

- In the N/W layer, we add the information that A does not know its IP address & the BOOTP server's IP address. So, in the source address we put 0.0.0.0 & destination as 255.255.255.255 (broadcast).

- In the DLL, the source address is put as M_A (A's MAC address) and destination MAC address as FF:FF:FF:FF:FF:FF (broadcasted).

- The BOOTP server will reply to the request and it acknowledges the request packet.

BOOTP server will reply to A with the IP address. BOOTP server uses a static mapping table to give the IP address. (Certain disadvantages of static mapping table ?).

- There has to be a BOOTP server in every network as the router with bound the N/W. So, to request the BOOTPS, every N/W needs a BOOTP server.

A solution to having a BOOTPS in every N/W is to have a relay agent in the N/W where there is no BOOTP server. The relay agent knows the address of the BOOTPS in some other network. If B on the network having no BOOTPS requests, then the RA reads the BOOTP request packet & unicast it to BOOTP server on other N/W (unicast packet). Then the BOOTP server looks up the table & sends reply to the relay agent through intermediate router using unicast. Then relay agent will unicast the packet to B.

— In a N/W we have a BOOTP server & in the other N/Ws, we have relay agents.

So, now the mapping table is centralised.

This is an advantage of BOOTP, over RARP.

— The disadvantage remains as it uses static mapping table. If there are n stations we have n IP address entries, irrespective of the fact that we're using a station or not.

* DHCP (Dynamic host configuration protocol).

(To know IP)

Only difference between DHCP & BOOTP is the table is not static in DHCP.

DHCP table has 2 parts — one static & other dynamic part. Which N/Ws need to be online all the time, have their entries in the static part (like web server, email server), file server, OS server). For the dynamic part, only when a host asks for an IP to go online, we provide it with an IP from a pool of IP addresses dynamically.

for each dynamically allocated IP, if the host does not use the IP for a certain amount of time, its IP will be pulled back

Used to give away IP to hosts in the N/W

DORA (Discover - offer - request - acknowledgement)

Again, when that hosts requests an IP, there is no guarantee it will get the previously allocated IP.

Static

MAC	IP
M_1	I_1
M_2	I_2
⋮	⋮

Dynamic

MAC	IP	Lease time
M_A	I_A	t_A
M_B	I_B	t_B

DHCP Mapping table

✓ — Only one DHCP server is enough (as we use relay agents).

✓ — Number of IP entries is equal to no. of hosts online as we use static as well as dynamic table. → (Last page of this section).

DHCP Provides the host with an IP address, N/W mask, host's address, DNS server address, lease time etc. default gateway

* ICMP. (Internet Control Message Protocol)

✓ ICMP
- Error handling or feedback messaging
  - TTL exceed
  - Parameter problem
  - Source quench
  - Source redirect
  - Destination unreachable
- Request and reply
  - Echo request & reply
  - Time stamp request & reply
  - N/W mask request & reply
  - Router solicitation & advertisement

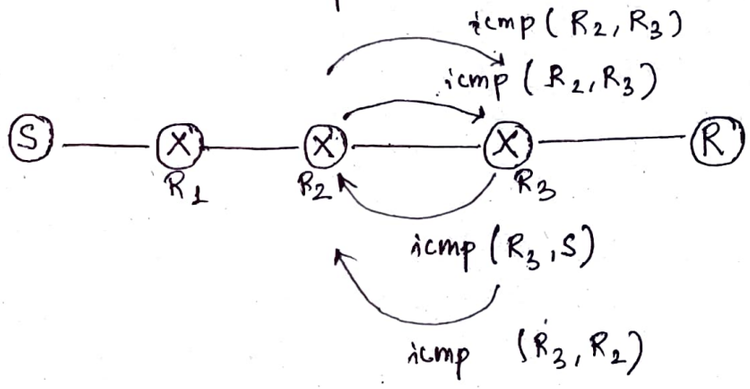
✓ ICMP works at N/W layer.

- ICMP will take the support of IP in order to send the data packets.

- In case of error handling, only if some packet is lost or some error occurs an ICMP packet is sent.

- In case of request & reply, the sender ICMP sends an request to other & it gets a reply ICMP.

- When a host (or router) discards a packet, it sho send an ICMP packet to the source.



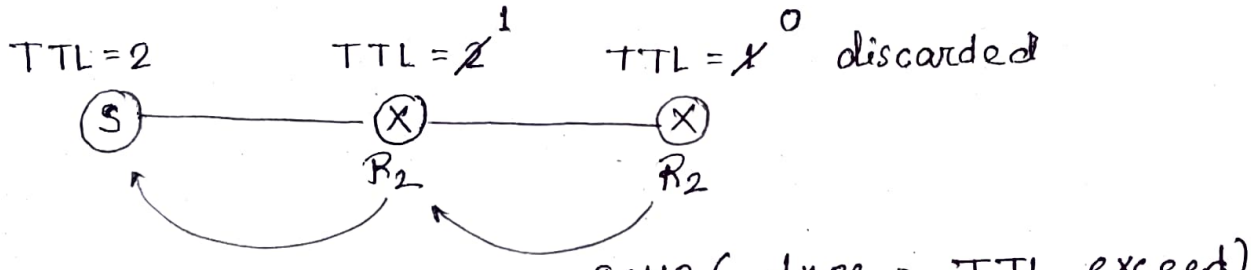
When R_3 discards a packet due to heavy congestion, it sends an icmp packet directed to S. Now, due to heavy congestion R_2 discard this icmp packet & sends an icmp packet to R_3. This loop goes on & creates congestion. (ICMP transmission loop)

Solution to this is not to generate an ICMP packet when an ICMP packet is discarded ICMP packet should be genrated only when IP packet is discarded.

Hence, there is no guarantee that the ICMP packet will reach the concerned destination. So, still the IP + ICMP protocol is ~~unreliable~~ unreliable.

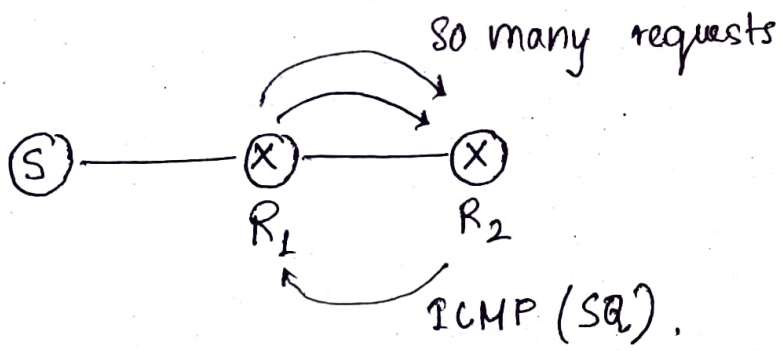
→ Feedback messaging :

- TTL Exceed



When S receives the ICMP packet, it may increment the TTL or it might understand that the packet is involved in a circular loop.

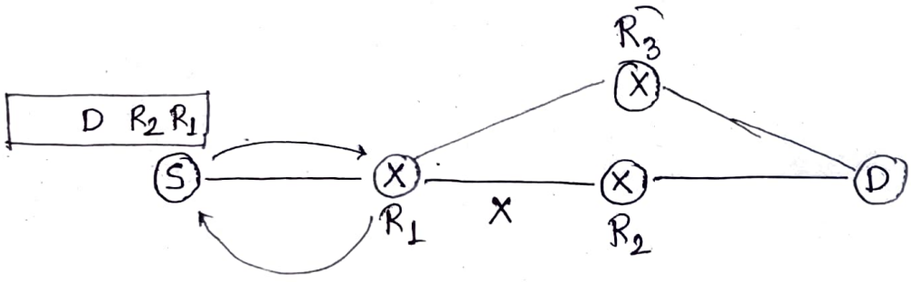
- Source quench.



R_1 is sending so many requests to R_2. R_2 then sends ICMP packet (SQ) to R_1 to stop sending requests.

Parameter problem.

Suppose S is using strict source routing (on the packet the route is put, through which the packet will traverse.)



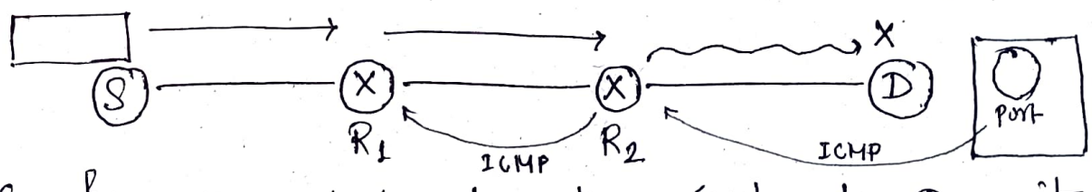
Due to some problem between R_1 & R_2 the packet can't be sent. Then R_1 sends ICMP packet saying the parameter problem arised, as the parameters of the packet did not let the packet to travel through R_3.

Also, in some case the checksum value may get corrupted in the packet. Then, also ICMP packet quoting parameter problem will be be generate

Destination unreachable.

(Destⁿ host or Destⁿ port unreachab

host, port, mtu not compatible

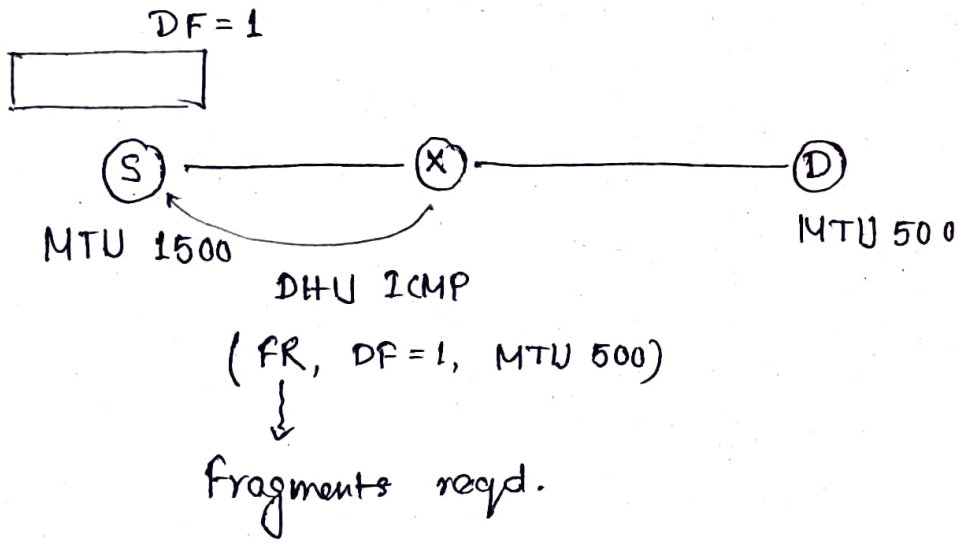


If S has a packet to be sent to D, it sends it through R_1, R_2. Now, if D is unreachable or down i.e. D does not reply on R_2's ARP request (IP → MAC), then the packet can't be sent to D. Now, R_2 sends an ICMP packet quoting 'destination host unreachabl

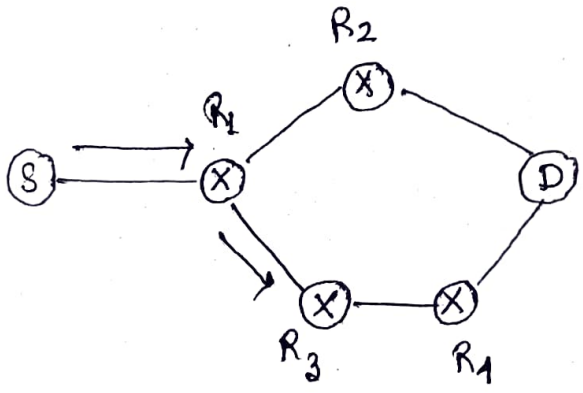
Another problem arises when the destination host is available but the exact port where the packet has to be sent has no process associated with it. Now, the host is going to discard the packet and will send an ICMP packet quoting destination port unreachable.

The concerned router sends the 'Destⁿ host unreachable' ICMP packet; the host sends the 'Destⁿ port unreachable' ICMP packet.

In some other scenario, when the destination MTU (max. transmission unit) is less than the source MTU and in the packet 'Do not fragment' bit is set to true, there will be an ICMP packet stating destination host unreachable.



Source redirect

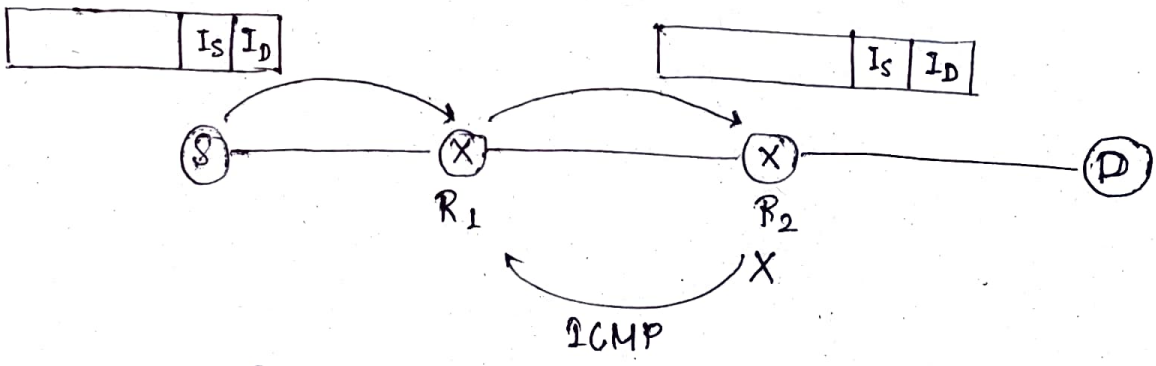


The best route from S to D is $R_1 R_2$. But, for some reason, in the routing table entries get corrupted and it says to go to R_3. Now,

In case the router R_3 knows that there's a better path ~~from~~ from S to D, it sends an ICMP packet quoting 'source redirect', that is there's better route and we need to redirect the packet.

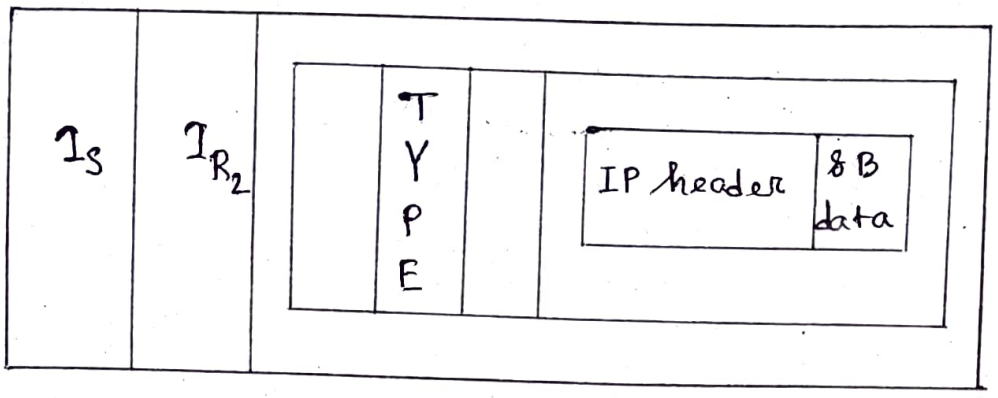
In the 'source redirect', although ICMP packet is generated, the packet coming from source is not discarded at the router who is sending ICMP packet. So, source redirect ICMP works as a warning. When R_1 gets the ICMP packet from R_3, then from the next time R_1 sends packets through R_2 only.

Things the sender should know when a packet is discarded



When R_2 discards a packet from S, it needs to say that it discarded the packet, why it did so, & which packet it has discarded. So, in the ICMP packet, it puts the IP of R_2 which tells who discarded the packet. In the ICMP datagram, there's a field of 'type' that have the concerned ICMP type. Also, to state which packet

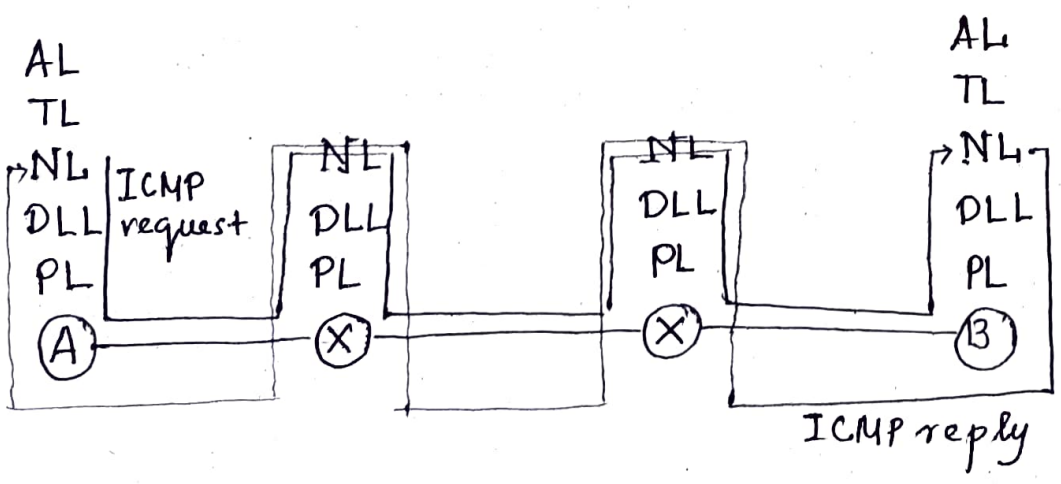
discarded, it puts the discarded packet's IP header along with its 8 byte data part (this contains the TCP part - sequence no., source port & dest^n port).



- Both for TCP & UDP, ICMP packet will be generated. For ICMP packets, ICMP packets are not generated. If there are some fragments of some data packet only the first packet's ICMP packet is generated.

If the first fragment is not discarded, then also ICMP packet will not be generated. Only if the first fragment is lost, the sender will be getting the ICMP packet.

→ Request & reply



ICMP works at network layer. So, from N/W layer of A, ICMP packet is sent directed to B. Then B replies to A.

It helps to know of the N/W layer of the destination of the layers in the intermediate routers are working properly or not.

While we ping (packet internet groper), we send ICMP packets as trial of connection. All the echo requests go till the network layer.

* — The first request a host will send as soon as it is online, is the DHCP request to know the IP of itself. After that it needs to know about its default gateway. For this, the host sends an ICMP packet (Router solicitation). The routers present in the N/W reply saying they are available to be used as default gateway.

Whenever a new router is added to the N/W it will advertise about itself saying that it's available (router advertisement).

Next, the host will need to know its subnet mask. The router is requested to send the N/W mask & it replies: (N/W mask request & reply.)

1. IP address
2. Default getaway
3. N/W mask

- When there are different hosts at different time zones, there is problem of synchronisation, So, a new kind of ICMP packet is used - timestamp request & reply, using which the hosts can know each other's time & synchronise (highly unreliable).

- Traceroute. (Application of ICMP)

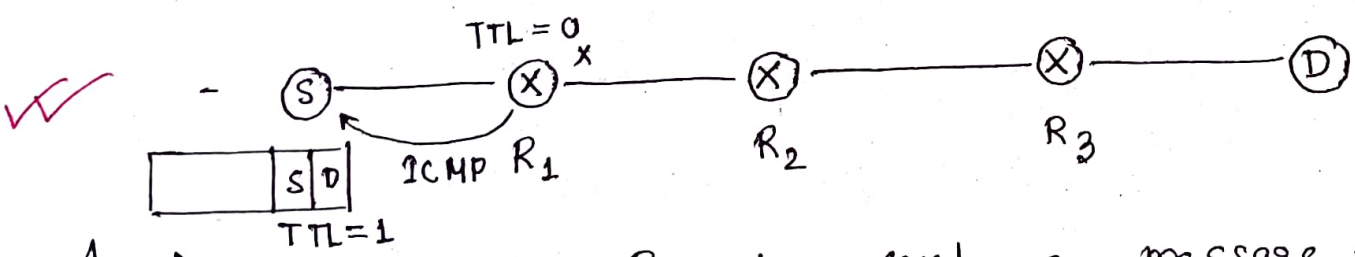
eg. traceroute www.google.com [in linux]

Says what are all the routers on the way between the host & Google.

Implemented using ICMP.

- Recordroute (in ICMP) : All routers are going to write their IP addresses on the packet & destination will find out what are the routers IP addresses.

In contrast, traceroute will let the the sender know what are the routers en route.



At first we make R_1 to send a message to S to know R_1's address. Then R_2, R_3 consecutively.

* We come to know about Router's address by making the R generate ICMP (TTL) packet.

Here, we send the packet from S with TTL = 1.

At R_1, packet is discarded & R_1 sends ICMP packet. to S.

Next, we send the packet from S with TTL=2. Now, we get R_2's ICMP (TTL exceed) and come to know its address.

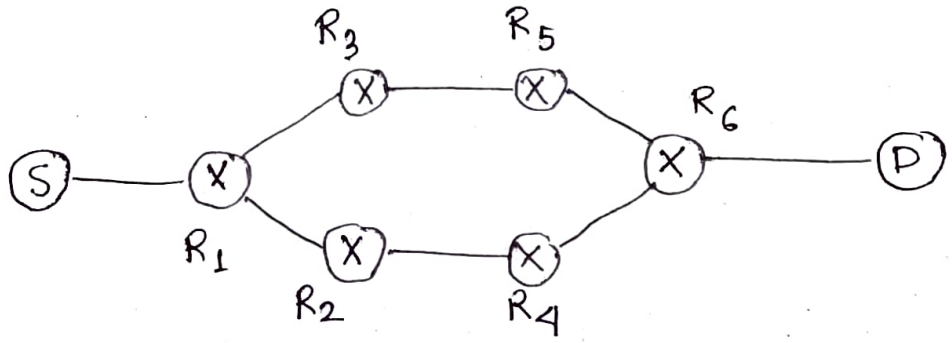
This way, we know the routers' addresses.

- If at some point, we don't get any ICMP packet at S, we should not assume that the packet has reached D. It may be the case that ICMP packet is lost.

- Most of the time ICMP packets are discarded at routers.

- To distinguish between the ICMP lost & destination reached, we can make the destination also send an ICMP packet. And we have to intentionally create an error at D so that it sends an ICMP packet. To do this we put UDP packet with a dummy port no. on the source's message, so that when the packet reaches D, it is discarded & D sends ICMP packet of type 'destination port unreachable'. So, whenever we get an ICMP ~~pkt~~ quoting 'DPU', we can say dest^n is reached.

- Sometimes, we may not get a feasible path from S to D.



TTL 1 R_1

TTL 2 R_1 R_3

TTL 3 R_1 R_2 R_4

TTL 4 $R_1 R_3 R_5 R_6$

TTL 5 D

S R_1 R_3 R_4 G

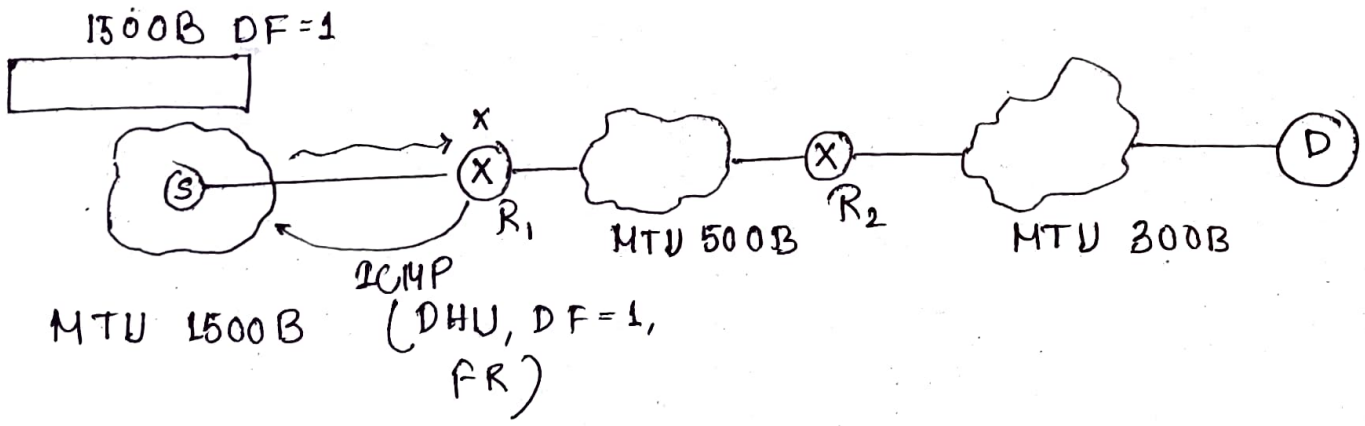
Not a possible path.

So, it is not guaranteed, the routers traceroute is giving will form a real route.

But for recordroute, there's always a route from S to D as the routers send their address en route.

- PMTUD. (Application of ICMP)

Path MTU Discovery.



S wants to know what is the least MTU in the route, so that it can know what size data it has to send to avoid fragmentation

S sends packet of size 1500B & R_1 discards it & sends an ICMP packet (DHU, do not fragment = 1 & fragmentation reqd.). This ICMP packet

will also let S know that the MTU of next N/W is 500 B (say).

Now, S sends packet of size 500 B, with DF = 1. R_1 allows this packet, but R_2 does not. R_2 sends an ICMP packet (DHU, FR, DF = 1, MTU = 300B).

Then, S sends packet of size 300 B, Packets that S send have a UDP packet with dummy port. So, when D is reached, D will send 'dest^n port unreachable' ICMP packet. It says the dest^n is reached.

- DHCP > 4 Steps (DORA)

i) Discover : New host that wants to connect to the internet sends a discover request (broadcast) – can someboc give an IP address to me.

Discover Packet

S: 0.0.0.0	D: 255.255.255.255
Sender UDP port 68 (client)	Receiver UDP port 67 (server)

ii) Offer : DHCP server offers IP address by broadcasting (because server doesn't know which host needs it) – I can offer 192.168.1.3 for next 3600 sec.

Offer packet

S: DHCP server IP	D: 255
UDP 67	UDP 68

iii) Request : Host chooses offer & requests address. to all (broadcast) – DHCP request. – I want to accep the offer to use 192.168.1.3 for next 3600s.

Request packet

S: 0.0.0.0	D: 255
68	67.

iv) Acknowledgement : DHCP server acknowledges that host can use the IP address provided (broadcast). – as of now you can use 192.168.1.3 for next 3600 s.

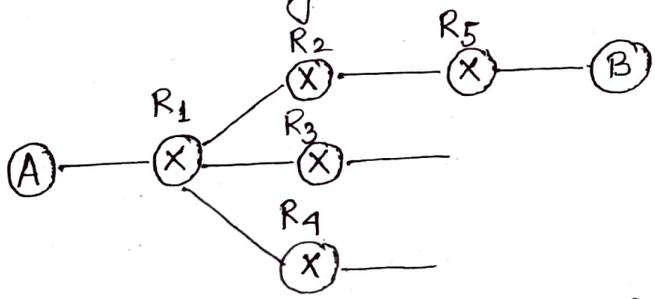
DHCP ACK Packet

S: DHCP server IP	D: 255
67	68

– There can be multiple DHCP servers & they all can respond with an offer. Then, host chooses one offer.

Routing

* Process of preparing the routing table so that we can do switching better.



If routing is not used, we do flooding — sending packets to all available routers.

* Advantages of flooding:
  i) No routing table required.
  ii) Shortest path is guaranteed (each path is taken in flooding).
  iii) Highly reliable.

These are disadvantages of routing.

Disadvantages of flooding:
  i) Duplicate packets.
  ii) High traffic.

These are advantages of routing.

* In military networks we use flooding.

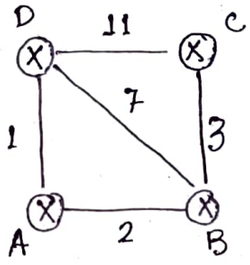
* Types of routing algorithms —
  a) Static (Manually creating the routing table offline) Depending on traffic & topology they don't change. This is not used.
  b) Dynamic (Depending on T & T routing table changes.)
     - DVR (Distance vector routing)
     - LSR (Link state routing)

* Distance Vector Routing (DVR).

Distributed Bellman-Ford algorithm

We discuss about the shortest paths between 2 routers, as networks are connected to the routers by only one hop.

eg.



Every router will generate a routing table (dest^n, weight, next hop) with its local knowledge, & share to its neighbours.

1st round: RT for B.

D^n	Dist.	NH
A	2	A
B	0	B
C	3	C
D	7	D

RT for A.

D^n	Distance	NH
A	0	A
B	2	B
C	∞	-
D	1	D

RT for C

D^n	Dist	NH
A	∞	-
B	3	B
C	0	C
D	11	D

RT for D

D^n	Dist	NH
A	1	A
B	7	B
C	11	C
D	0	D

After this, every router will send its distance vector to all its neighbours.

2nd round:

@ A: DV from B, D

@ B: DV from A, C, D.

@ C: DV from B, D

@ D: DV from A, B, C

Now, every router will update its distance vector based only on neighbours' information.

For instance, @ A,

From B

2
0
3
7

From D.

1
7
11
0

⇒

D^n	Dist	NH
A	0	A
B	2	B
C	5	B
D	1	D

$A \rightsquigarrow B = \min \begin{cases} A \xrightarrow{1} D + D \xrightarrow{7} B \\ A \xrightarrow{2} B + B \rightsquigarrow B \checkmark 2 \end{cases}$

next hop.

$A \rightsquigarrow C = \min \begin{cases} A \xrightarrow{2} B + B \xrightarrow{3} C \checkmark 5 \\ A \xrightarrow{1} D + D \xrightarrow{11} C. \end{cases}$

$A \rightsquigarrow D = \min \begin{cases} A \xrightarrow{1} D + D \xrightarrow{0} D \checkmark 1 \\ A \xrightarrow{2} B + B \xrightarrow{7} D. \end{cases}$

Faster processing -

B	D.
2	1
0	7
3	11
7	0

→ take each row & add with AB, AD, then take the minimum (except for A → A case).

AB = 2 AD = 1.

⇒

	AB	AD
B	2+0	7+1
C	3+2	11+1
D	7+2	0+1

A

A	0	A
B	2	B
C	5	B
D	1	D

→ previously ∞

* Always use distance vectors from previous round.

Now, @ B.

A	C	D.
0	∞	1
2	3	7
∞	0	11
1	11	0.

BA = 2 BC = 3 BD = 7

B

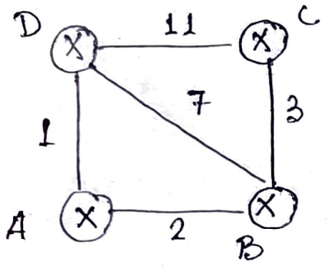
A	2	A
B	0	B
C	3	C
D	3	A

→ previously 7.

We do this for C & D.

Run this for 3 rounds.

✓ - We can come up with the final routing table with our intuition.



@ A

A	0	A
B	2	B
C	5	B
D	1	D

@ B

A	2	A
B	0	B
C	3	C
D	3	A

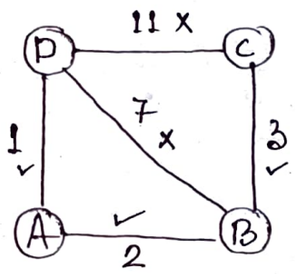
@ C

A	5	B
B	3	B
C	0	C
D	6	B

@ D

A	1	A
B	3	A
C	6	A
D	0	D.

✓ - Edges that are unused after DVR converges -



We take each edge & try to find if there is an alternative better path available. If available, this edge will not be use

For example, AB edge has weight 2.

Other alternatives. $A \rightarrow D \rightarrow C \rightarrow B$ 15

$A \rightarrow D \rightarrow B$ 8

No better path, so AB is used

CD edge has weight 11.

Better path $C \rightarrow B \rightarrow A \rightarrow D$ 6 weight.

So, CD unused.

✓ - Routing tables are prepared $n-1$ times if there are n routers. (Shortest path between 2 nodes contains at most $n-1$ edges if there are n nodes)

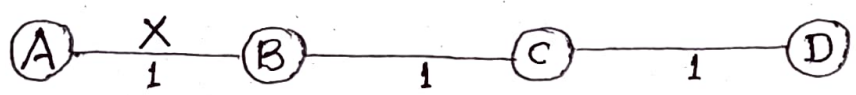
→ Count to infinity. - disadvantage of DVR.

Bad news spreads slow.
Good news spreads fast.

Routing loop.

Routing loops occur when an interface goes down, or when 2 routers send updates to each other at the same time.

eg.



Only one link between A & the other parts of the network.

	A	B	C	D
A	0, A	1, B	2, B	3, C
B	1, B	0, B	1, C	2, C
C	2, B	1, B	0, C	1, D
D	3, C	2, C	1, C	0, D.

Imagine link between A & B is cut. At this time, B corrects its table. After a specific amount of time, routers exchange their tables & B receives C's RT. Since, C doesn't know what has happened to the link between A & B, it says that it has a link to A with the weight of 2 ($C \xrightarrow{1} B$, $B \xrightarrow{1} A$). B receives this table & thinks there is a separate link between C & A, so it corrects its table & changes infinity to 3 ($B \xrightarrow{1} C$, $C \xrightarrow{2} A$).

Once again, routers exchange their tables. When C receives B's RT, it sees that B has changed the weight of its link to A from 1 to 3, so C updates its table & changes the weight of the link to A to 4.

This process loops until all nodes find out that the weight of link to A is infinity. In this way, DV algorithms have a slow convergence rate.

	B	C	D.
Sum of weight to A after cut	∞, A	2, B	3, C.
" " " after 1st update	3, C	2, B	3, C
" " " after 2nd "	3, C	4, B	3, C
" " " " 3rd "	5, C	4, B	5, C
" " " " 4th "	5, C	6, B	5, C
" " " " 5th "	7, C	6, B	7, C
" " " " nth "	...	....	....
∞	∞	∞	∞

2+1 B-C, C

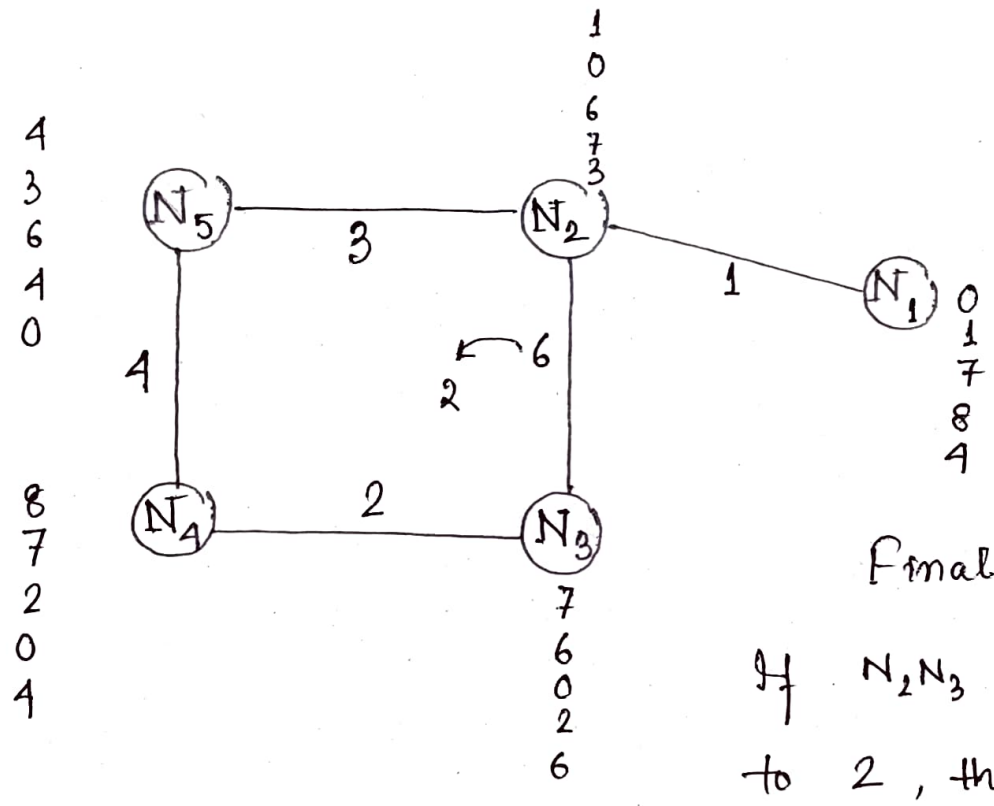
3+1 C-B, B

- One way to solve this problem is for routers to send information only to the neighbours that are not exclusive links to the destn.

C shouldn't send any information to B about A, because B is the only way to A.

- The DVR algorithm keeps on repeating periodically & never stops. This is to update the shortest path in case any link goes down or topology changes.

Q



Final DVs given

If N_2N_3 changes from 6 to 2, then what will be new DV at N_3?

→ @ N_3, after update,

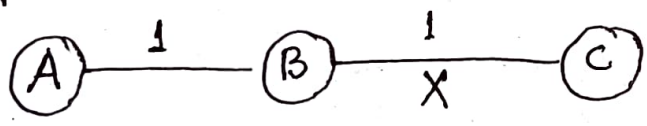
from N_2	from N_4
1	8
0	7
6	2
7	0
3	4.
N_2N_3 2	N_3N_4 2

N_1	3	N_2
N_2	2	N_2
N_3	0	N_3
N_4	2	N_4
N_5	5	N_2

↳ New distance vector.

- Solution to count to infinity problem.

Preventing routing loops in DVR by prohibiting a router from advertising a route back onto the interface from which it was learned.



- Split horizon route advertisement.

If B & C link goes down & B had received a route from A, B could end up using that route via A. A would send the packet right back to B, creating a loop. But, according to split horizon rule,

node A does not advertise its route for C (namely A to B to C) back to B. On the surface, this seems redundant since B will never route via node A because the route costs more than the direct route from B to C.

So, if a neighbouring router sends a route to a router, the receiving router will not propagate this route back to the advertising router on the same interface.

- Route poisoning

When a router fails, distance vector protocols spread the bad news about a route failure by poisoning the route. Route poisoning refers to the practice of advertising a route, but with a special metric value called infinity. Routers consider routes advertised with an infinite metric to have failed.

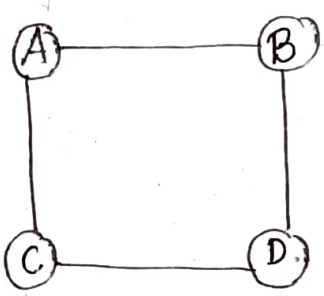
Main disadvantage of poison reverse is that it can significantly increase the size of routing announcements in certain fairly common N/W topologies.

- Holddown timers.

Whenever a router learns about an unreachable route, it starts a timer. Until the time is up, the route discards any routing update that tells the unreachable route has become reachable. Ensures the router waits until the N/W is stable

to modify its routing table.

eg. Route poisoning



B's RT

A	1	A
B	0	B
C	3	A
D	4	D.

What B shares to A.

→

∞
0
∞
4

As B learns from A to go to A & C.

What B shares to D

→

1
0
3
∞

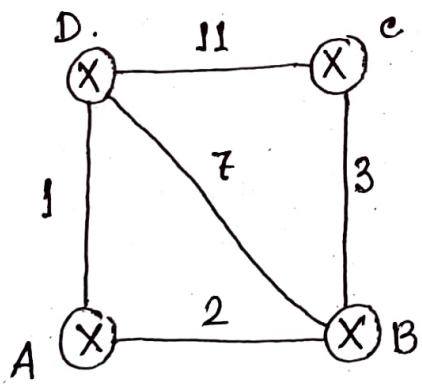
As B learns from D to go to D.

- Disadvantages of DVR.

i) Count to infinity.
ii) Converges slowly.
iii) Routing loops.

* Link state routing (LSR). (uses Dijkstra's algo)

eg.



@ B.

A	2
D	7
C	3

@ C

D	11
B	3

@ A

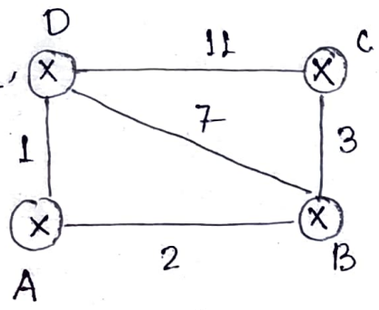
B	2
D	1

Here, each node will flood the information it has to everyone.

@ D

C	11
B	7
A	1

LSR is based on global knowledge. (Whereas DVR is based on local knowledge).



Now, @ A, looking all info. (Global database)

A will apply single source shortest path algorithm (Dijkstra's).

⇓ Local routing table

Desⁿ	Dist.	NH
A	0	A
B	2	B
C	5	B
D	1	D.

LSR converges fast.

- As we are using flooding, one disadvantage is heavy traffic.

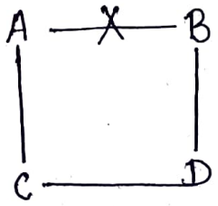
- To differentiate between latest & old packets coming from same source, we use sequence no.

Routers	Latest seq. no. seen.	
B	~~10~~ 15	(B, 8) Discarded
C	20	(B, 15) Accepted as latest
D	30	

- To solve loop problem, we have TTL entry.

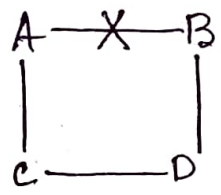
- We have lifetime/validity of the distance entries so prevent rejection of updated values due to error in some stage.

- Transient problem.



AB link goes down. Before some timeout, A will send packets to B having no knowledge of the down link. This creates black hole problem

✓ Transient looping in LSR.



C will not have the immediate knowledge of the link being down. C sends packets to A ↑ A to C.

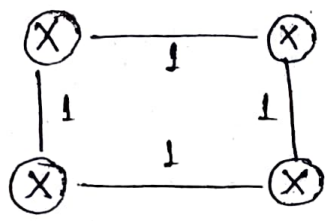
* DVR vs. LSR.

DVR	LSR.
1. BW required is less due to local sharing, small packets & no flooding.	1. BW reqd. is more due to flooding & sending of large link state packets.
2. Based on local knowledge, since it updates table based on information from neighbours.	2. Based on global knowledge, i.e. it has knowledge about entire N/W.
3. Uses Bellman-Ford algo.	3. Uses Dijkstra's algo.
4. Traffic is less.	4. Traffic is more.
5. Converges slowly.	5. Converges faster.
6. Count to infinity problem.	6. No count to infinity.
7. Persistent looping problem.	7. Transient looping.
8. Practical implementation is RIP & IGRP.	8. OSPF & ISIS.
9. CPU & memory low utilization.	9. Intensive.
10. ~~Period~~ Periodic updates.	10. Triggered updates.

→ RIP (Routing Information Protocol).

Metric - hop count.

∞ - 16



→ OSPF (Open shortest path first).

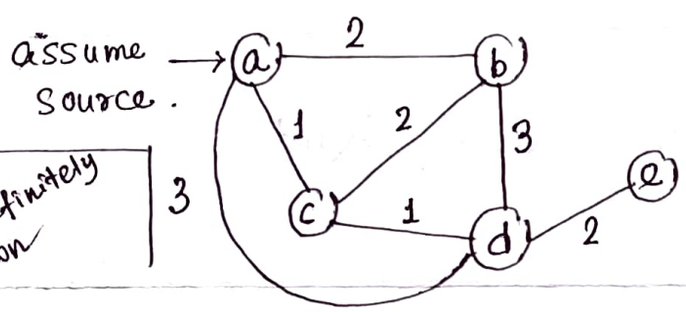
Divide routers into areas. Flooding done in one region. Border router keeps summary of entire area & sends to area 0 / zone 0 / backbone zone. Area 0 sends to other N/W areas. # of flooded packets is less.

Link State Routing

Each node has the knowledge of entire topology of network (global knowledge).

2 steps – building routing ~~table~~ topology from global knowledge,
finding shortest path using Dijkstra's & creating routing table.

At first link state packets are sent through flooding to build global knowledge about the topology.



$D(v)$ – distance of v node from source

$P(v)$ – previous node of v in present least cost path

NH

set of nodes whose least cost path definitely known

↑ subgroup.

N'	D(b) P(b)	D(c) P(c)	D(d) P(d)	D(e) P(e)
a	2, a	(1, a) shortest	3, a	∞
ac (through c to other)	(2, a)	(2, c) a→c→d shortest		∞
acd	(2, b) shortest		4, d a→c→d→	
acdb			(4, d) shortest	
acdbe				

→ only direct links @ first

pick anyone as shortest.

$$D(v) = \min_{v=b,\, w=c} \{D(v), D(w) + C(w,v)\}$$

$$2 \quad , \quad 1 + 2$$

LSR Steps 1. Identify the neighbouring nodes. 2. Measure the cost to each of its neighbours. 3. Form a packet containing all the info. 4. Send the packet to all other nodes (flooding). 5. Compute the shortest path to every other node using Dijkstra's algo.

$$D(v) = \min \{ D(v), D(w) + c(w,v) \}.$$

Info is shared at regular interval.

- Link state packet :

Advertiser ID	N/W ID	Cost	Next hop

- LSR requires more CPU, memory but it converges faster than DVR.

Distance Vector Routing.

Bellman - Ford Algo -

$d_x(y) \rightarrow$ least cost path from x to y

$$= \min_v \{ c(x,v) + d_v(y) \}.$$

A

B	2
D	1

B

A	2
C	3
D	7

C

B	3
D	11

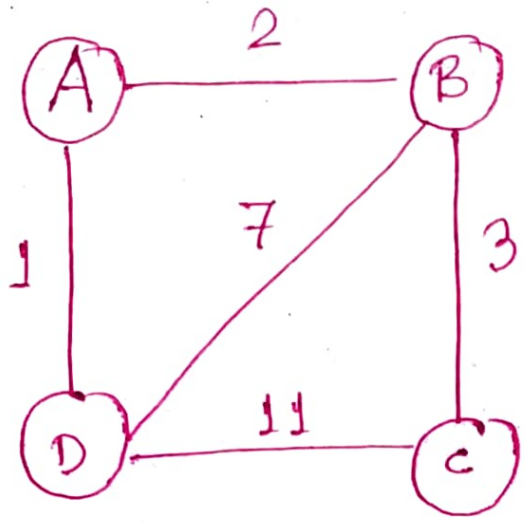
D

A	1
B	7
C	11

@ A,

global knowledge

Source →



AD = 1 ✓

AB = 2

AC = 5, B

N'

	B	C	D
A	2, B	∞	1, D
AD	2, B	12, D	
ADB		5, B	