COMPUTER SCIENCE AND ENGINEERING

COMPUTER NETWROK

(CODE: - 18CS504)

MATERIAL

CSE-3B

REGD.NO:-.....

NAME:-

.....

BAPATLA ENGINEERING COLLEGE::BAPATLA

(AUTONOMOUS)

DEPARTMENT OF CSE

3/4-BTECH CSE

TEACHER:-K.Arun Babu.

BAPATLA ENGINEERING COLLEGE

(AUTONOMOUS)

Department of CSE

Proposed Syllabus for COMPUTER NETWORKS

III B.Tech (Theory)

Lectures: 4 Periods/Week | Continuous Assessment: 50M

Sem End Exam Duration: 3 hours | Sem End Exam : 50M

Credits: 3

UNIT-I

Data Communications & Networking Overview: A Communications Model, Data Communications, Data Communication Networking.

Protocol Architecture: The Need for a Protocol Architecture, A Simple Protocol Architecture, OSI, The TCP/IP Protocol Architecture.

Digital Data Communication Techniques: Asynchronous & Synchronous Transmission, Types of Errors, Error Detection, Error Correction.

UNIT-2(16 Periods)

Data Link Control: Flow Control, Error Control.

Network Layer: Network Layer Design Issues: Store-and-Forward Packet Switching, Services Provided to the Transport Layer, Implementation of Connectionless Service, Implementation of Connection Oriented Service, Comparison of Virtual-Circuit & Datagram Subnets.

Routing Algorithms: The Optimality Principle, Shortest Path Routing, Flooding, Distance Vector Routing, Link State Routing, Hierarchical Routing.

Congestion Control Algorithms: General Principles of Congestion Control, Congestion Prevention Policies, Congestion Control in Virtual-Circuit Subnets, Congestion Control in Datagram Subnets, Load Shedding, Jitter Control.

UNIT-3(15 Periods)

Quality of Service: Requirements, Techniques for Achieving Good Quality of Service the Network Layer in the Internet: The IP Protocol, IP Addresses, Internet Control Protocols.

The Transport Layer:

The Transport Service: Services Provided to the Upper Layers, Transport Service Primitives, Berkeley sockets

Elements of Transport Protocols: Addressing, Connection Establishment, Connection Release, Flow Control and Buffering, Multiplexing, Crash Recovery.

UNIT-4 (14 Periods)

The Internet Transport Protocol (UDP): Introduction to UDP, Remote Procedure Call, The Real-TimeTransport Protocol.

The Internet Transport Protocols (TCP): Introduction to TCP, The TCP Service Model, The TCP Protocol, The TCP Segment Header, TCP Connection Establishment, TCP Connection Release, Modeling TCP Connection Management, TCP Transmission Policy, TCP Congestion Control, TCP Timer Management.

Application Layer: The Domain Name System(DNS): The DNS Name Space, Resource Records, Name Servers.

UNIT-I

CHAPTER-1:-*Data Communications & Networking Overview*

- When we communicate , we share information
- Information can be LOCAL or REMOTE
- Between Individuals LOCAL communication occurs face to face
- REMOTE communication occurs over a long distance
- When we refer to COMPUTER SYSTEMS, Data is represented in the form of Binary Units (Bits) in the form of Zeros (0's) and One's (1's)
- Also the entities can most of the times be considered to be COMPUTERS

- **DEFINITION OF DATA COMMUNICATION: -**

"Data Communication is the exchange of data (in the form of 0's and 1's) between two devices (computers) via some form of the transmission medium."

- **LOCAL and REMOTE Data Communication: -**

LOCAL: -

Data communication is considered to be local if the communicating devices are present in the same building or a similarly restricted geographical area.

REMOTE: -Data Communication is considered remote, if the devices are farther apart.

- **Data Communication System:-**

For Data Communication to occur, the communicating devices must be a part of a communication system made up of some specific kind of hardware and software
This type of a system is known as a **"DATA COMMUNICATION SYSTEM".**

Note:The main aim of every communication system is to transmit data with maximum rate of transmission, with maximum accuracy and with least possible transmission impairments(noise,attenuation,delay distortion..etc).

- **Effectiveness of Data Comm. System characteristics: -**

Effectiveness depends upon four fundamental characteristics:

- Delivery
- Accuracy
- Timeliness (Better NEVER than LATE)
- Jitter

I. Delivery- The system must deliver data to the correct destination. Data must be received by the intended device or user and only by that device or user.

2. Accuracy- The system must deliver the data accurately. Data that have been altered in transmission and left uncorrected are unusable.

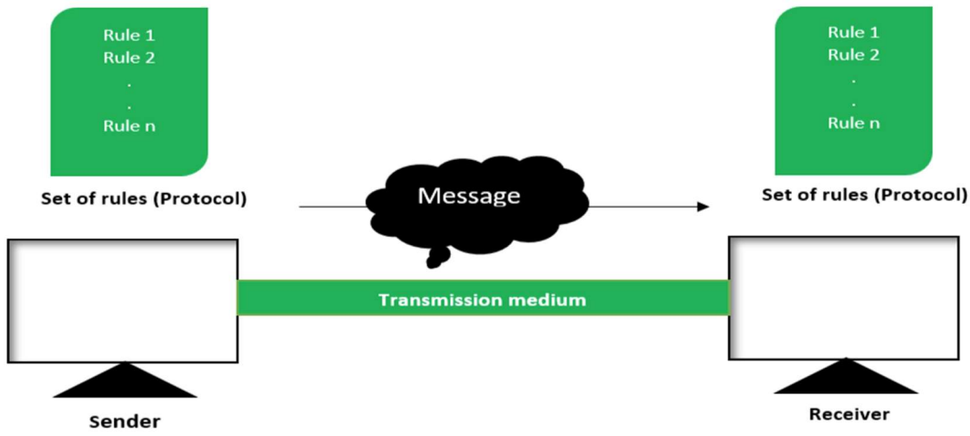
3. Timeliness- The system must deliver data in a timely manner. Data delivered late are useless. In the case of video and audio, timely delivery means delivering data as they are produced, in the same order that they are produced, and without significant delay. This kind of delivery is called real-time transmission.

4. Jitter- Jitter refers to the variation in the packet arrival time. It is the uneven delay in the delivery of audio or video packets. For example, let us assume that video packets are sent every 3D ms. If some of the packets arrive with 3D-ms delay and others with 4D-ms delay, an uneven quality in the video is the result.

❖ Components of Data Com Systems:

Any system is made up of more than one component. Similarly, a data communication system is made up of 5 components as shown in the fig:

- Message
- Sender
- Receiver
- Transmission Medium
- Protocol (Set of rules)



1. Message - It is the information to be communicated. Popular forms of information include text, pictures, audio, video etc. Text is converted to binary, number does not converted, image is converted to pixels, etc.

2. Sender - It is the device which sends the data messages. It can be a computer, workstation, telephone handset etc.

3. Receiver - It is the device which receives the data messages. It can be a computer, workstation, telephone handset etc.

4. Transmission Medium - It is the physical path by which a message travels from sender to receiver. Some examples include twisted-pair wire, coaxial cable, radio waves etc.

Transmission Media

Medium	Speed	Cost
Twisted Wire	300bps-10Mbps	Low
Microwave	256Kbps-100Mbps	Low
Coaxial Cable	56Kbps-200Mbps	Low
Fiber Optic Cable	500Kbps-10Gbps	High

5. Protocol - It is a set of rules that governs the data communications. It represents an agreement between the communicating devices. Without a protocol, two devices may be connected but not communicating.

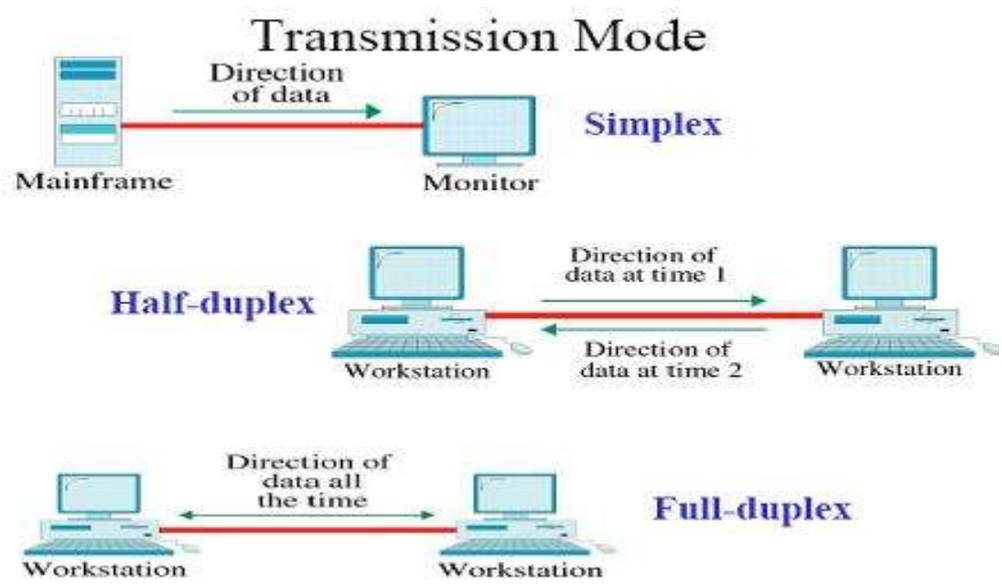
❖ **Communication between two devices can be simplex, half-duplex, or full-duplex:-**

→Simplex

In simplex mode, the communication is unidirectional, as on a one-way street. Only one of the two devices on a link can transmit; the other can only receive. Keyboards and traditional monitors are examples of simplex devices. The keyboard can only introduce input; the monitor can only accept output. The simplex mode can use the entire capacity of the channel to send data in one direction.

→Half-Duplex

In half-duplex mode, each station can both transmit and receive, but not at the same time. When one device is sending, the other can only receive, and vice versa. The half-duplex mode is like a one-lane road with traffic allowed in both directions. When cars are traveling in one direction, cars going the other way must wait. In a half-duplex transmission, the entire capacity of a channel is taken over by whichever of the two devices is transmitting at the time. Walkie-talkies and CB(citizens band) radios are both half-duplex systems.



Full Duplex: -

In full-duplex mode, both stations can transmit and receive simultaneously. The full-duplex mode is like a two-way street with traffic flowing in both directions at the same time. One common example of full-duplex communication is the telephone network. When two people are communicating by a telephone line, both can talk and listen at the same time.

❖ A COMPONENTS OF A DATA COMMUNICATION SYSTEM: -

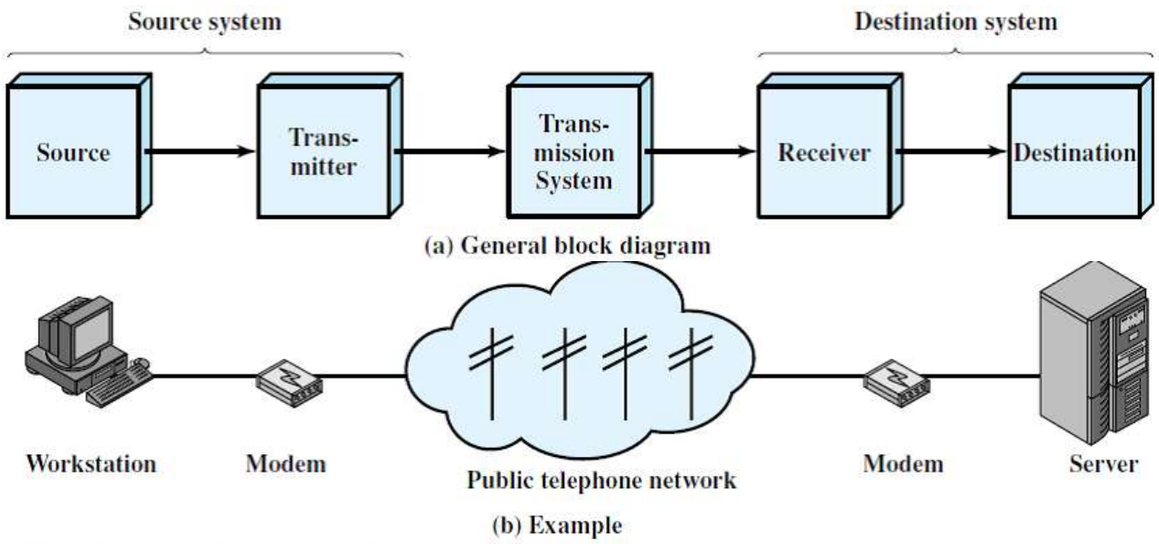


Figure 1.2 Simplified Communications Model

The fundamental purpose of a communications system is the exchange of data between two parties. Figure 1.2b presents one particular example, which is communication between a workstation and a server over a public telephone network. Another example is the exchange of voice signals between two telephones over the same network.The key elements of the model are as follows:

- **Source**. This device generates the data to be transmitted; examples are telephones and personal computers.

- **Transmitter**: Usually, the data generated by a source system are not transmitted directly in the form in which they were generated. Rather, a transmitter transforms and encodes the information in such a way as to produce electromagnetic signals that can be transmitted across some sort of transmission system. For example, a modem takes a digital bit stream from an attached device such as a personal computer and transforms that bit stream into an analog signal that can be handled by the telephone network.
- **Transmission system**: This can be a single transmission line or a complex network connecting source and destination.
- **Receiver**: The receiver accepts the signal from the transmission system and converts it into a form that can be handled by the destination device.
  For example, a modem will accept an analog signal coming from a network or transmission line and convert it into a digital bit stream.

- **Destination**: Takes the incoming data from reciver.

A SIMPLIFIED DATA COMMUNICATION MODAL: -

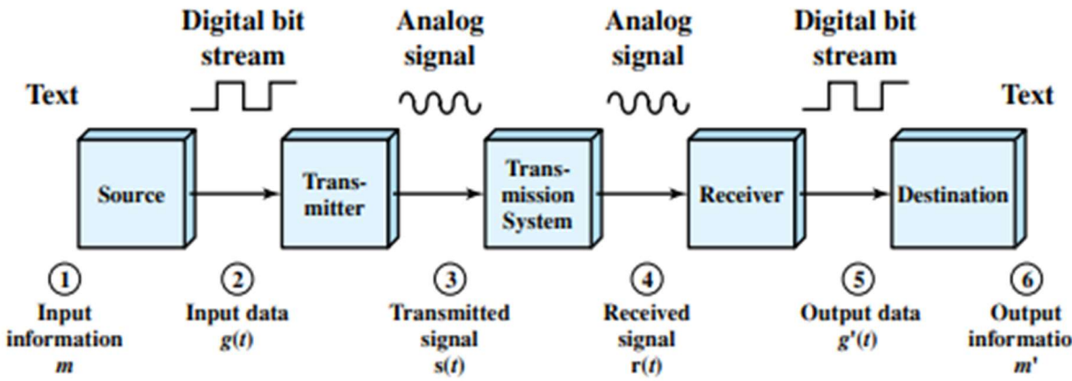


Figure 1.3 Simplified Data Communications Model

→Suppose that the input device and transmitter are components of a personal computer. The user of the PC wishes to send a message m to another user. The user activates the electronic mail package on the PC and enters the message via the keyboard (input device). The character string is briefly buffered in main memory. We can view it as a sequence of bits (g) in memory. The personal computer is connected to some transmission medium, such as a local network or a telephone line, by an I/O device (transmitter), such as a local network transceiver or a modem. The input data are transferred to the transmitter as a sequence of voltage shifts [g(t)] representing bits on some communications bus or cable. The transmitter is connected directly to the medium and converts the incoming stream [g(t)] into a signal [s(t)] suitable for transmission.

The transmitted signal s(t) presented to the medium is subject to a number of impairments,before it reaches the receiver. Thus, the received signal r(t) may differ from s(t). The receiver will attempt to estimate the original s(t), based on r(t) and its knowledge of the medium, producing a sequence of bits These bits are sent to the output personal computer, where they are briefly buffered in memory as a block of bits In many cases, the destination system will attempt to determine if an error has occurred and, if so, cooperate with the source system to eventually obtain a complete, error-free block of data. These data are then presented to the user via an output device, such as a printer or screen. The message as viewed by the user will usually be an exact copy of the original message (m).

EXAMPLE:- Now consider a telephone conversation. In this case the input to the telephone is a message (m) in the form of sound waves. The sound waves are converted by the telephone into electrical signals of the same frequency.These signals are transmitted without modification over the telephone line. Hence the input signal g(t) and the transmitted signal s(t) are identical. The signals (t) will suffer some distortion over the medium, so that r(t) will not be identical to s(t). Nevertheless, the signal r(t) is converted back into a sound wave with no attempt at correction or improvement of signal quality. Thus, is not an exact replica of m. However, the received sound message is generally comprehensible to the listener.

❖ Communication Tasks:-

There are some key tasks that must be performed in a data communication system
Elements can be added, deleted, or merged together.

Communications Tasks

Transmission system utilization	Addressing
Interfacing	Routing
Signal generation	Recovery
Synchronization	Message formatting
Exchange management	Security
Error detection and correction	Network management
Flow control	

Transmission System Utilization :-Need to make efficient use of Transmission facilities that are shared among a no. of communicating devices

For Example:

→Techniques like **Multiplexing** to allow multiple users to share total capacity of a Transmission Medium

→**Congestion Control**: TX. System should not be overwhelmed by traffic.

Interfacing: Communication between the devices and transmission channel takes place with the help of Interface. Interface is the way; a computer presents information to users or receives information from users.

Signal Generation Electromagnetic Signals travel over Transmission Medium. Once an interface is established, Signal generation is required

→**Properties of Signals**

→Capable of being propagated over TX. Medium

→Interpretable as data at the Receiver

(Or)

After establishing an interface, the other job of communication system is to generate the signals. These signals can either be in analog or digital format and in the format that is easily understood by destination.

Synchronization :-The transmission and the reception should be properly synchronized. Synchronization means that the receiver must be able to determine, when to expect a new transmission and when to send acknowledgements. In other words transmitter and receiver should have an agreement on the nature as well as timing of the signals.

Exchange Management: - If the data needs to be exchanged in both directions over a period of time, both parties must cooperate as follows

→Whether both devices must transmit simultaneously or take turns

→Amount of Data to be sent at one time

→Format of the Data
→What to do when an Error Arises

Error Detection and Correction: -In all comm. Systems, there is a potential risk for errors and impairments.
→Signals are distorted to some extent before reaching their destination. Error Detection & Correction needs to be employed in Data Processing Systems where a change in say the contents of a file cannot be tolerated

Flow Control: - To make sure that source does not overwhelm destination by sending data faster than it can be handled and processed

Addressing & Routing: -If TX facility is shared by two or more devices, source must specify the identity or the address of the destination system and if Tx. System is itself a system, a proper route must be allocated that the data will take in order to reach the desired destination.

(Or)

Routing is a mechanism of choosing optimal path or shortest path through the network when different paths are available.

Recovery: -If a data transmission is interrupted due to a fault somewhere in the system, recovery techniques are needed. The objective is either to resume activity at the point of interruption and to restore the state of the system to what it was prior to the interruption

Message formatting: - sender and receiver must first agree on the format of data to be exchanged which can either be i the form of signals or binary format.

Security:- is very important issue in a Data Communication System. The sender needs to be assured that
→Only the Intended receiver receives the data
→Data is delivered unaltered.

Network management: -Data communication facility is a complex system that cannot create or run itself. Network management capabilities are needed to configure the system, monitor its status, react to failures and overloads and plan intelligently.

❖ Data Communication Networking: -

Communication networks are generally used to solve the problem, when it is difficult for two devices to be connected using point-to-point link.

Classification of Communication Networks:

1. Wide Area Network (WAN)
2. Local Area Network (LAN).

1.Wide Area Network:

WAN is geographically dispersed communication network that covers a wide range of areas which even include national and international boundaries. It consists of numerous smaller network such as LANs and MANs. In WAN, communication is made possible with the help of switches or routers which act as the intermediate nodes between source and destination. These nodes are not concerned about the data content, their job is to just transmit the data to the specified destination. An example of WAN is the internet.

The technologies that are used in WAN are as follows:

(i) Circuit switching
(ii) Packet switching
(iii) Frame relay
(iv) Asynchronous Transfer Mode (ATM).

(i) Circuit Switching:

It is a type of network that is used for establishing a dedicated circuit between nodes and terminals that is, a physical path is obtained between two end points in the network for a duration of connection. In circuit switching, the sender transfers the data over the physical path at high speed that reaches the intermediate nodes. The data is then routed or switched to the specified destination without any delay in transmission.

Telephone exchanges is a good example of circuit switching. It is generally used when real-time data is to be transmitted.

(ii) Packet Switching:

Packet switching is different from circuit switching. In packet switching data is transmitted in the form of packets which are created before sending the data. There is no dedicated physical path between source and destination. Each packet follows its own route to reach the destination over the network. At the destination, all packets are reassembled together to form the original message. Packet switching is

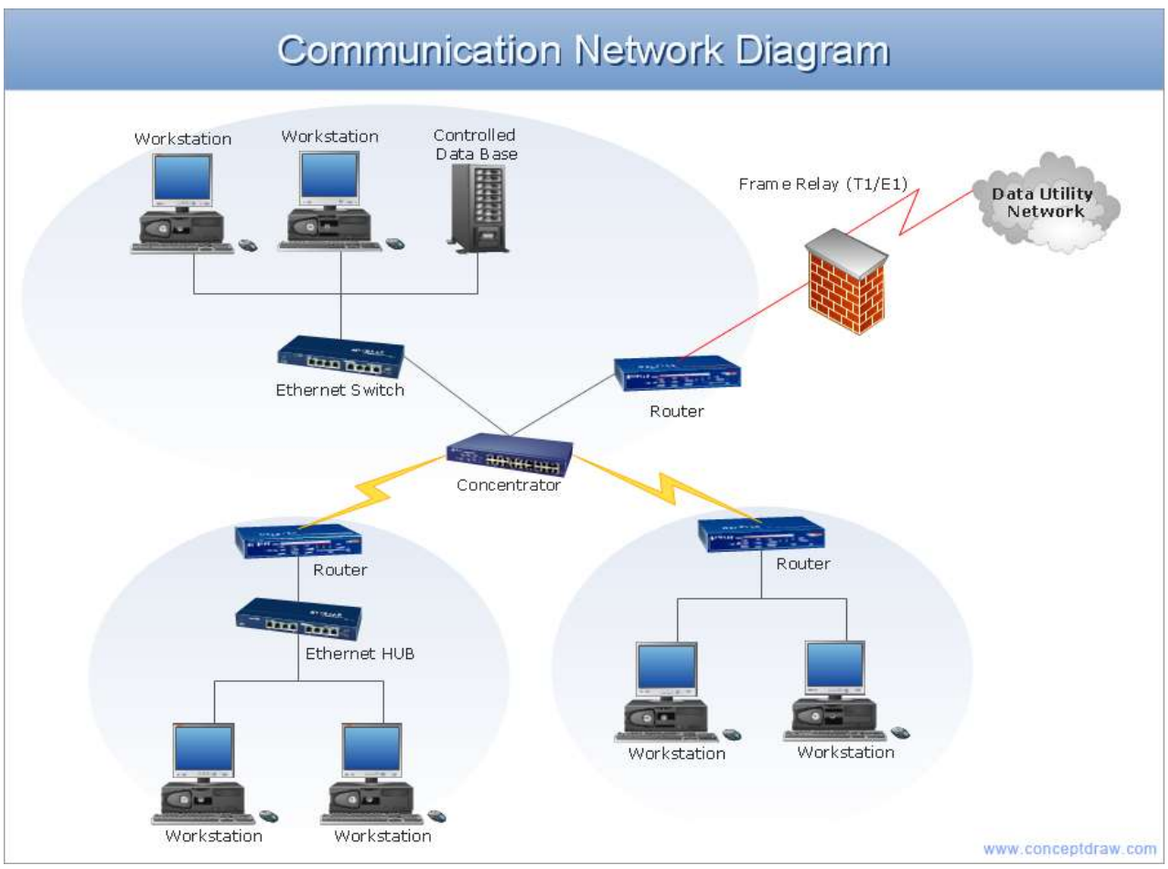
more efficient and robust for data that have a strong resistance over delays using transmission such as e-mails.

(iii) Frame Relay:

Frame relay is a WAN protocol that have a very high performance and which operates at the physical and datalink layer of ISO-OSI reference model. Frame relay consists of how many data transmission techniques that are used to transfer digital data quickly in the form of frames to and from source/destination. Frame relay network handles the transmission over a frequently changing path that is transparent to all end users.

(iv) Asynchronous Transfer Mode (ATM):-

ATM is an improvement over packet switching, circuit switching and frame relay. In frame relay data is transmitted as variable-size frames but in ATM, data is transmitted as fixed- sized cells. ATM is also known as cell relay which encodes data into small fixed-sized cells. ATM is a switching technology that have a dedicated logical connection that transmits data over physical medium. ATM transfers data at the rate of 10-100 Mbps. ATM reduces the overhead of error control and flow control when compared to other switching technologies.



EXAMPLE OF WAN NETWORK

2.Local Area Network (LAN):-

LAN is a smaller version of WAN.LAN is a computer network that covers small geographical areas such as single building or group of buildings. The advantages of van LAN and WAN, is that the data is transmitted with high speed and there is no need to have leased telecommunication lines

***Different types of LANS: -**

1.Switched LANS

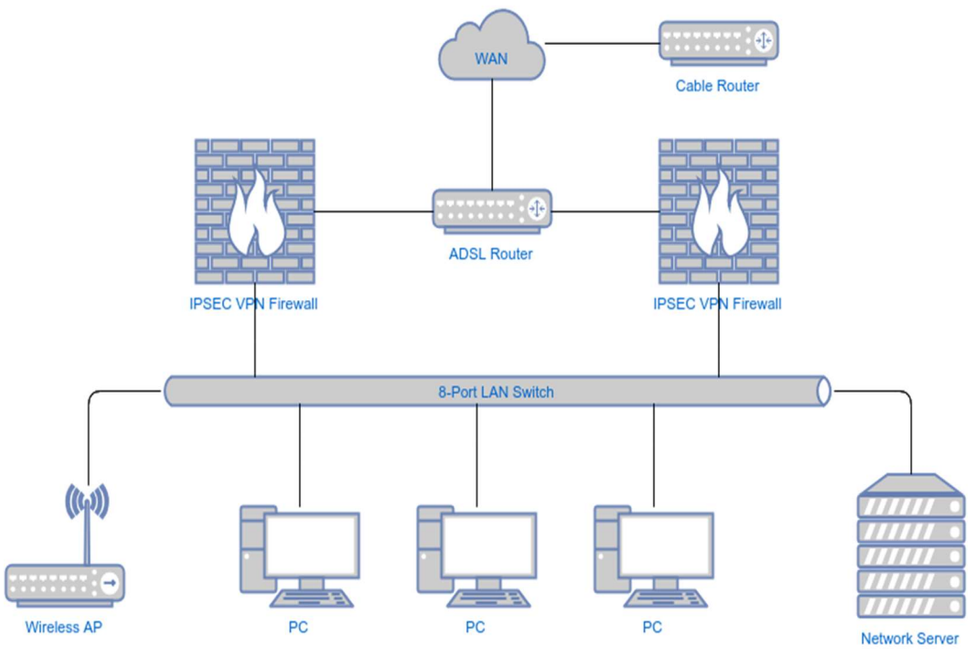
2.wireless LANS

1.Switched LANS: -

Ethernet is the most commonly used switched local area network with a single or multiple interconnected switches' other types of switched LANS include. Token ring, Fiber Distributed data interface (FDDI).

2.Wireless LANS: -

Wireless LANS is a way by which two or more computers ae connected without wires. These computers communicate with each other by transmitting information remotely. Using wireless LANs both voice and data can be transmitted. Wireless LANs have become popular because of easy installation and configuration.



EXAMPLE DIAGRAM FOR LAN

The distinguishing features of LAN are

- Network size is limited to a small geographical area, presently to a few kilometers.
- Data transfer rate is generally high. They range from 100 Mbps to 1000 Mbps.
- In general, a LAN uses only one type of transmission medium, commonly category 5 coaxial cables.
- A LAN is distinguished from other networks by their topologies. The common topologies are bus, ring, mesh, and star.
- The number of computers connected to a LAN is usually restricted. In other words, LANs are limitedly scalable.
- IEEE 802.3 or Ethernet is the most common LAN. They use a wired medium in conjuncture with a switch or a hub. Originally, coaxial cables were used for communications. But now twisted pair cables and fiber optic cables are also used. Ethernet's speed has increased from 2.9 Mbps to 400 Gbps.

❖ Metropolitan Area Network (MAN)

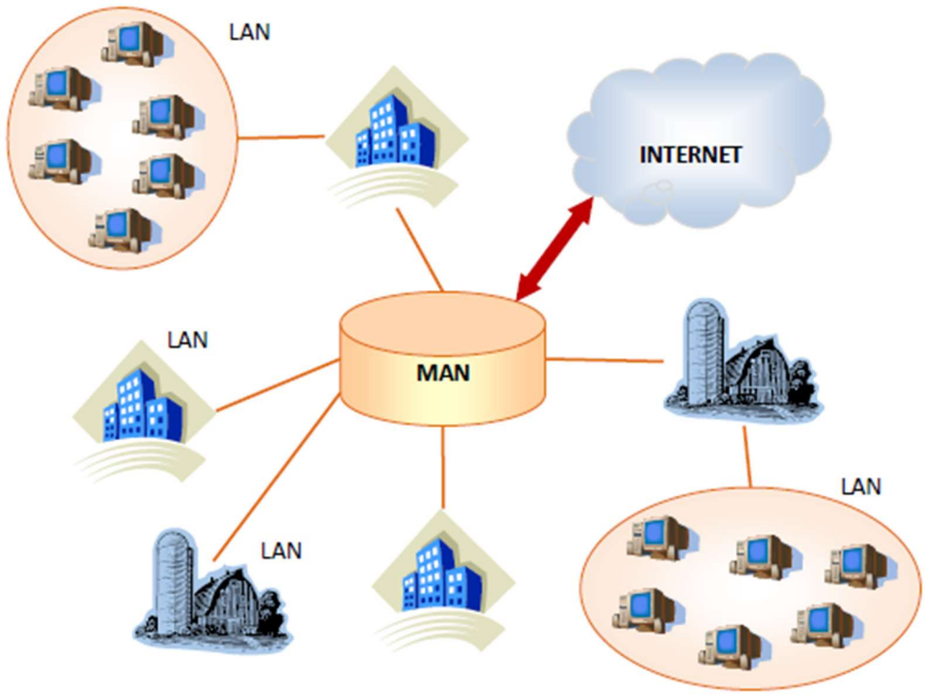
MAN is a network which is larger than LANs and smaller than WANs. MAN covers a geographic area or region that covers several Blocks of buildings or entire city. The advantage of MAN over WAN is it transfers the data at low cost but with high efficiency.

The distinguishing features of MAN are

- Network size generally ranges from 5 to 50 km. It may be as small as a group of buildings in a campus to as large as covering the whole city.
- Data rates are moderate to high.
- In general, a MAN is either owned by a user group or by a network provider who sells service to users, rather than a single organization as in LAN.
- It facilitates sharing of regional resources.
- They provide uplinks for connecting LANs to WANs and Internet.

Example of MAN

- Cable TV network
- Telephone networks providing high-speed DSL lines
- IEEE 802.16 or WiMAX, that provides high-speed broadband access with Internet connectivity to customer premises.



MAN NETWROK

CHAPTER-2

: Protocol Architecture:

Protocol: -

Protocol may be defined as set of conventions or rules that are useful in controlling the connection, communication and transfer of data between two devices. It controls the way data is sent from one computer to another. These rules are generally used for governing three important elements which forms the key elements of protocols. They are as follows,

1.Syntax.
2.Semantics.
3.Synchronization.

1.Syntax: -

It is defined as a rule, that state how the data and signal must be used. Syntax is used for specifying the structure of the data.

2.Semantics: -

Semantics give the meaning of each bit in the bit stream. It is also used for coordinating and for handling errors.

3.Synchronization: -

The two main functions that are provided by this element are as follows.

1.Matching the transfer speed.
2.Synchronizing the communication.

Network Models

- A network is a combination of **hardware** and **software** that sends data from one location to another.

  Hardware consists of the **physical equipment** that carries signals from one point of the network to another.

  Software consists of **instruction sets** that make possible the services that we expect from a network.

- Computer networks are created by different entities.

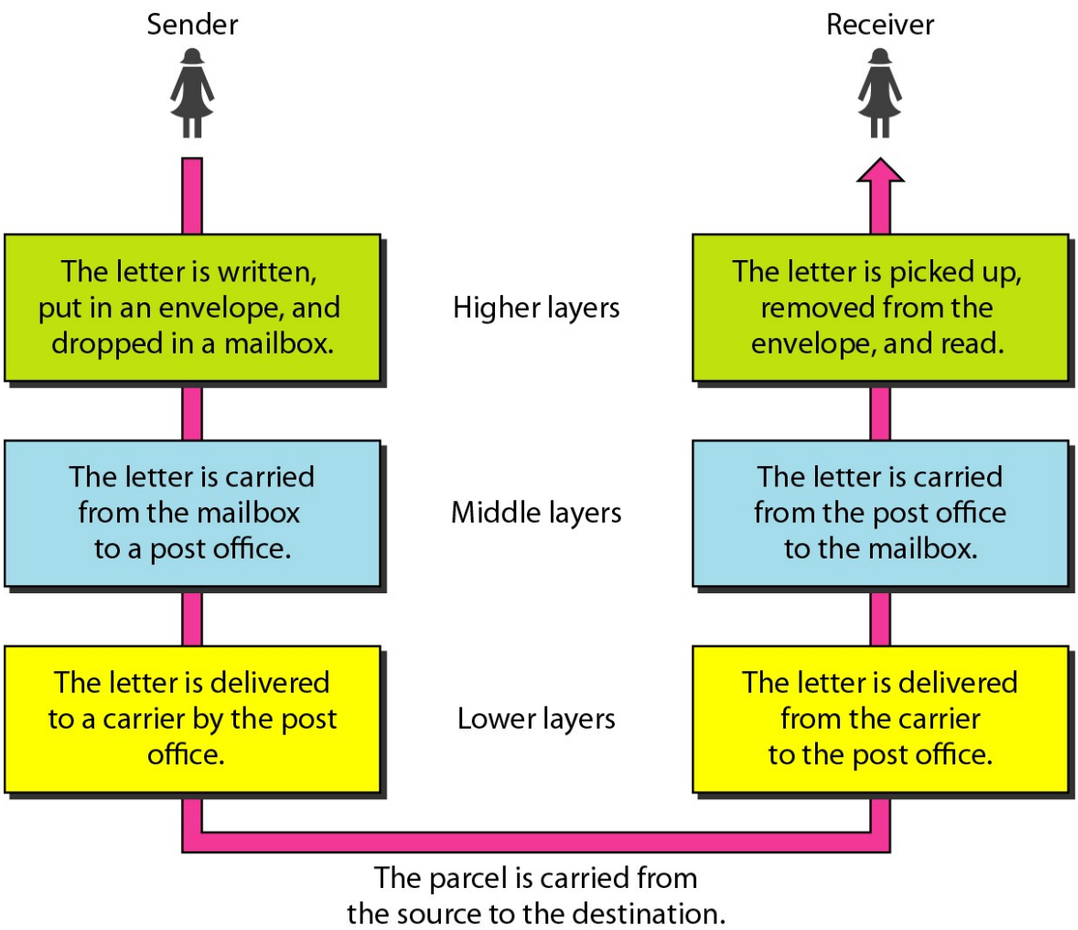
- Standards are needed so that these heterogeneous networks can communicate with one another.

Layered Tasks:-

Protocol architecture is basically used to understand the way how communication and exchange of data takes place between two end points.

We use the concept of layers in our daily life. As an example, let us consider two friends who communicate through postal mail The process of sending a letter to a friend would be complex if there were no services available from the post office. Figure shows the steps in this task.

Tasks involved in sending a letter



Sender, Receiver, and Carrier in Figure we have a sender, a receiver, and a carrier that transports the letter. There is a hierarchy of tasks.

At the Sender Site

Let us first describe, in order, the activities that take place at the sender site.

→ Higher layer. The sender writes the letter, inserts the letter in an envelope, writes the sender and receiver addresses, and drops the letter in a mailbox.

→Middle layer. The letter is picked up by a letter carrier and delivered to the post office.

→Lower layer. The letter is sorted at the post office; a carrier transports the letter.

011 the Way

The letter is then on its way to the recipient. On the way to the recipient's local post office, the letter may actually go through a central office. In addition, it may be transported by truck, train, airplane, boat, or a combination of these.

At the Receiver Site

→Lower layer. The carrier transports the letter to the post office.

→Middle layer. The letter is sorted and delivered to the recipient's mailbox.

→Higher layer. The receiver picks up the letter, opens the envelope, and reads it.

Three Layer Simplified Protocol Architecture Model: -

There are three different elements that are essential for initiating any communication. They are as follows.

1.Application.
2.Computer.
3.Network.

1.Application: -

It is a program that is designed to fulfil particular purpose which execute on the computer.

2. Computer :

The application that is to be processed is send to computer. Data exchange takes place from one computer to another that are connected through network.

3. Network :

A network consists of systems that are interconnected to each other.

Using these three elements, communication can be arranged in an orderly way between three independent layers.

❖ **Layers of Three-Layer Simplified Protocol Architecture Model:**

(i) Network access layer

(ii) Transport layer

(iii) Application layer

(i) **Network Access Layer:**

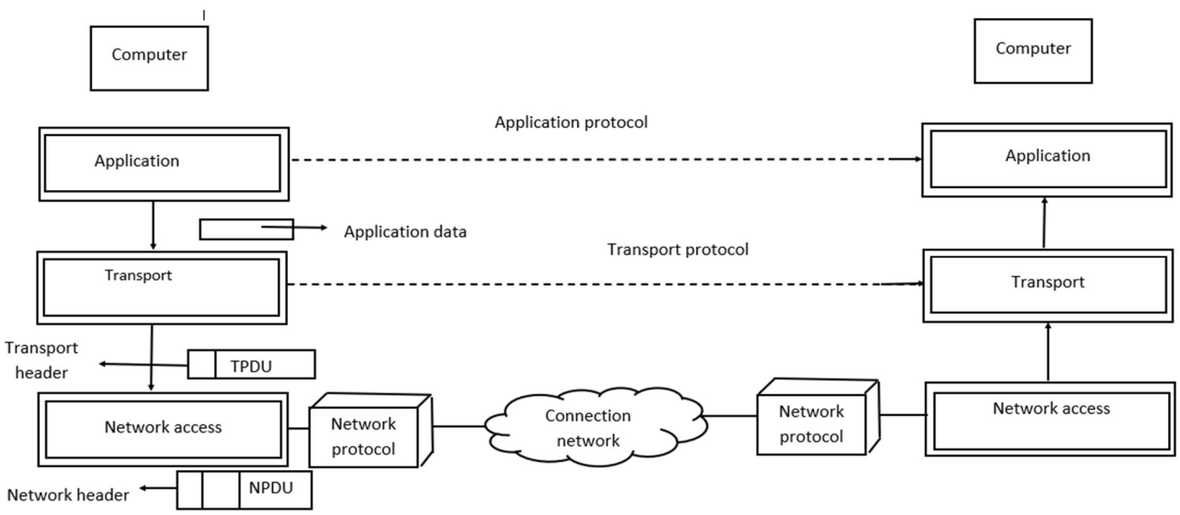
The main responsibility of network access layer is to transmit data between computer and network. The source must specify the destination address to network before transmitting the application. Using the address, the application is routed to its correct destination by the network. Thus, network access layer is concerned with the delivery of data from source to appropriate destination.

(ii) **Transport Layer:**

The main function of transport layer is to synchronize the exchange of data, so that data does not arrive out of order. This layer is also concerned with the reliability which is independent of nature of application while the data is being exchanged.

(iii)Application Layer: -

This Layer contains a set of principles that are used in preparing a computer to support different types of user application. A separate unit of program that perform a specific function is created for each type of application.



Each entity that is being transmitted must contain two addresses.

a) Service Access Point
b) Network Address.

Service Access Point: -

Using this unique address, transport layer supports multiple application that are present within each computer so that these applications can access the services available in transport layer.

Network Address: -

The main purpose of network address is to allow the network to transmit the application to appropriate destination. It is also unique address which is available with each computer.

→Steps for Exchanging Data

Control information as well as user data must be transmitted with each application to control the exchange operation.

Step 1: -

The application layer sends the block of data to transport layer.

Step 2: -

Transport layer receives these blocks and fragments them into smaller pieces and attaches a header that contain data that is received from application layer as well as control information, together which forms the transport protocol data unit. The main purpose for providing control information is to allow the destination host to identify the host that is sending the data. The various fields that are present in the header are destination address, sequence number, code for error detection.

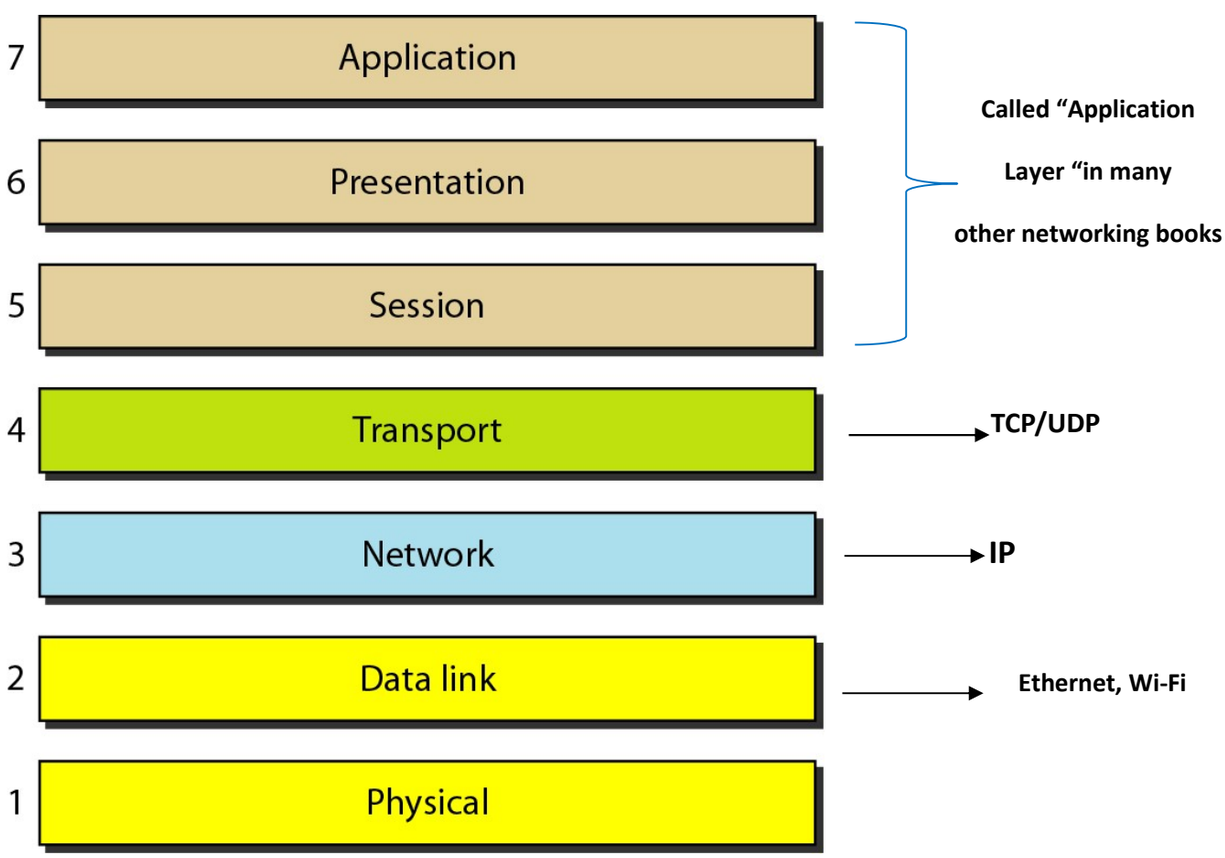
Step 3: -

Each protocol data unit is then transmitted to the network access layer which intern appends a network header that is used to transmit the data unit to the specified destination. The various field in network header are, destination network address, facilities request.

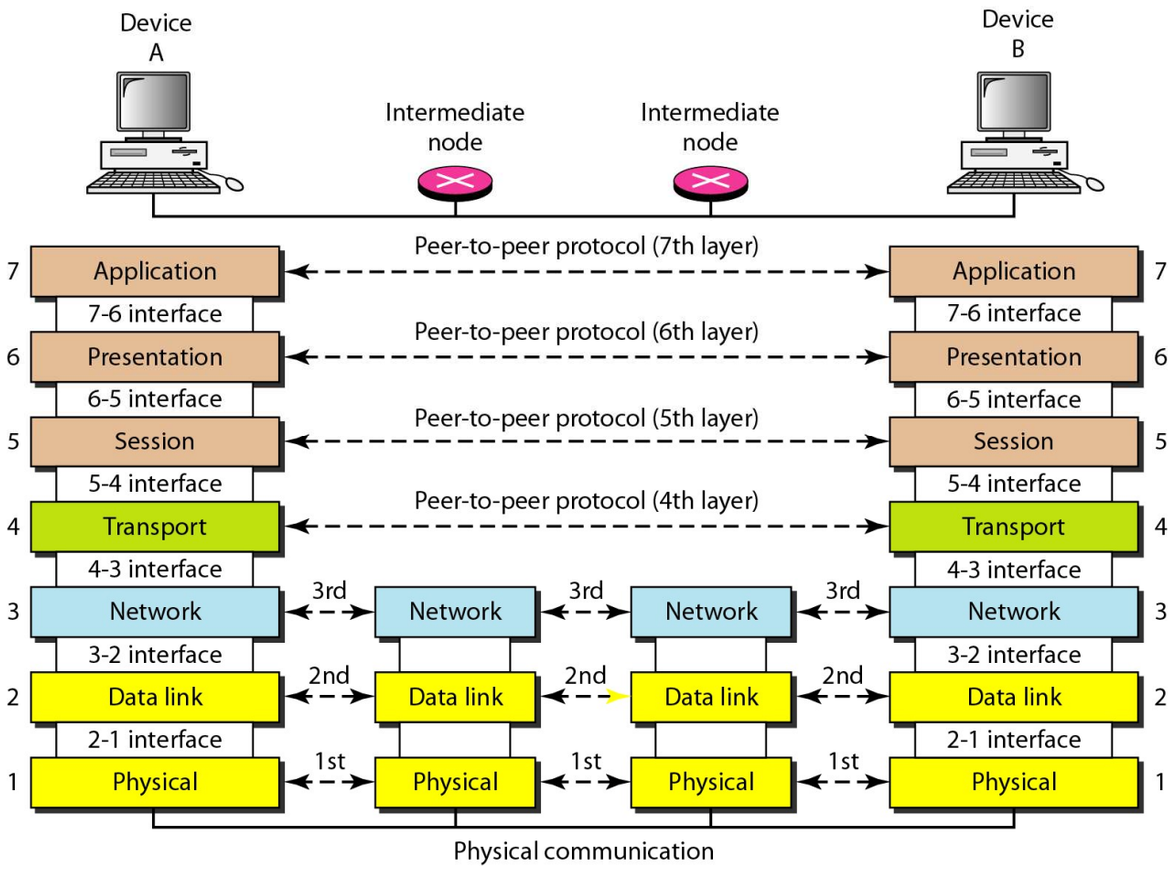
THE OSI MODEL: -

Established in 1947, the International Standards Organization (ISO) is a multinational body dedicated to worldwide agreement on international standards. An ISO standard that covers all aspects of network communications is the Open Systems Interconnection (OSI) model. It was first introduced in the late 1970s.

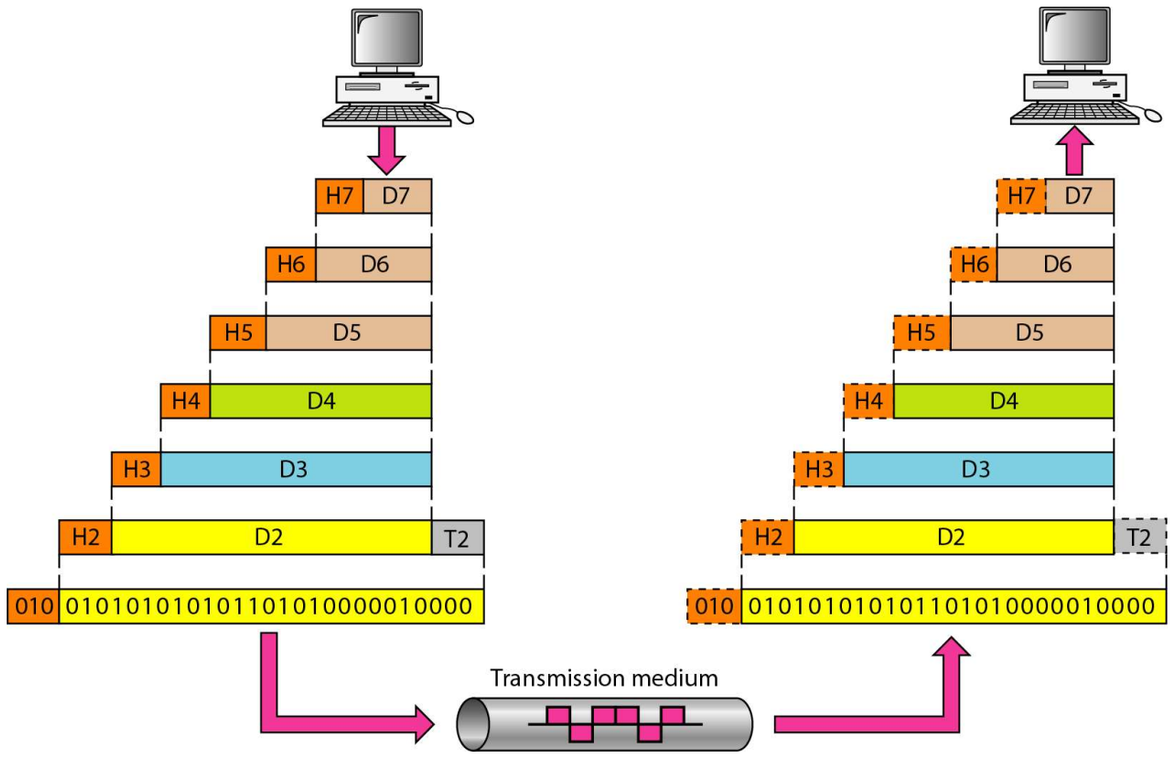
Seven *layers of the OSI model*



The interaction between layers in the OSI model



An exchange using the OSI model (encapsulation)



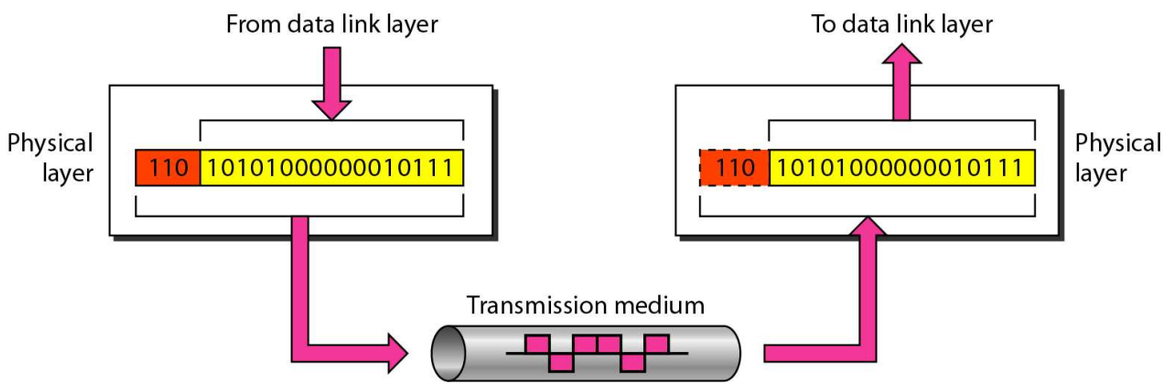
LAYERS IN THE OSI MODEL

→Physical Layer
→Data Link Layer
→Network Layer
→Transport Layer
→Session Layer
→Presentation Layer
→Application Layer

Function of the layers

Physical layer:-

The physical layer is responsible for movements of *individual bits* from one hop (node) to the next.



Physical medium:

- direct digital signals (e.g., Ethernet, optical fiber)
- modulated signals (e.g., WiFi, 3G)

Duties of physical layer

→Physical characteristics of interfaces and medium:
It defines the characteristic of the interface between devices and media. It also defines the type of transmission media
→Representation of bits:
The bit stream must be encoded into signals. It defines the type of representation (how 0, 1 are changed to signal).
→Data rate:
It defines the number of bits sent per second and also the duration of bits.
→Synchronization of bits:
The sender and receiver must be use the same bit rate also the receiver clock must be synchronized

→Line configuration:

Physical layer is concerned with the connection of devices to the media

(point-to point or multipoint)

→Physical topology:

How devices connected to make a network

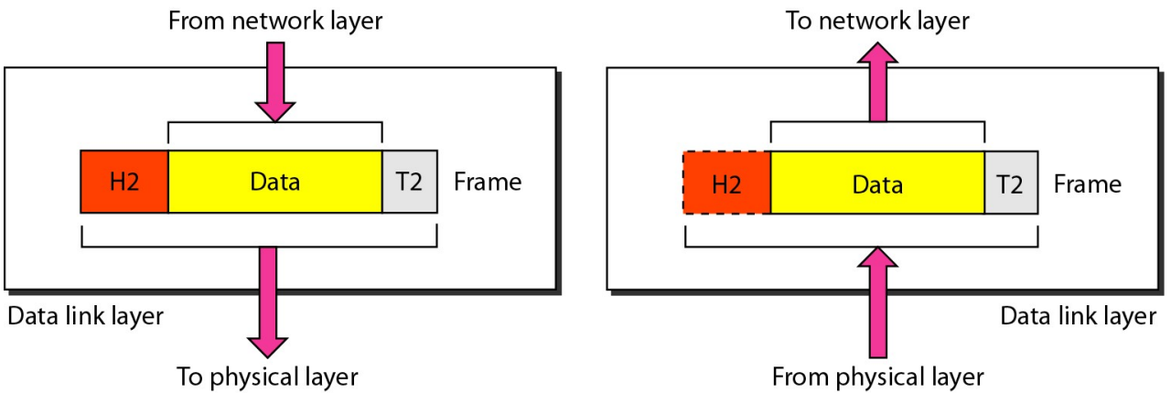
Devices can connected by using **Star, mesh , bus, ring or hybrid topology**

→Transmission mode:

It defines the direction of transmission between two devices (**simplex, half-duplex, or full duplex)**

Data link layer:-

The data link layer is responsible for moving frames from one hop (node) to the next.



Duties of data link layer: -

→Framing:

Divide the stream of bits received from network layer into data units called frames

→Physical addressing:

It adds a header to the frame to define the sender and receiver of the frame.

If the frame for a system outside the sender's network the receiver address: is the address of the connecting device that connects the network to next one (Router/switch).

→Flow control

It imposes a flow control mechanism, if the data rate at the receiver is less than produced by sender the data link layer imposes a flow control to avoid overwhelming the receiver.

→Error control

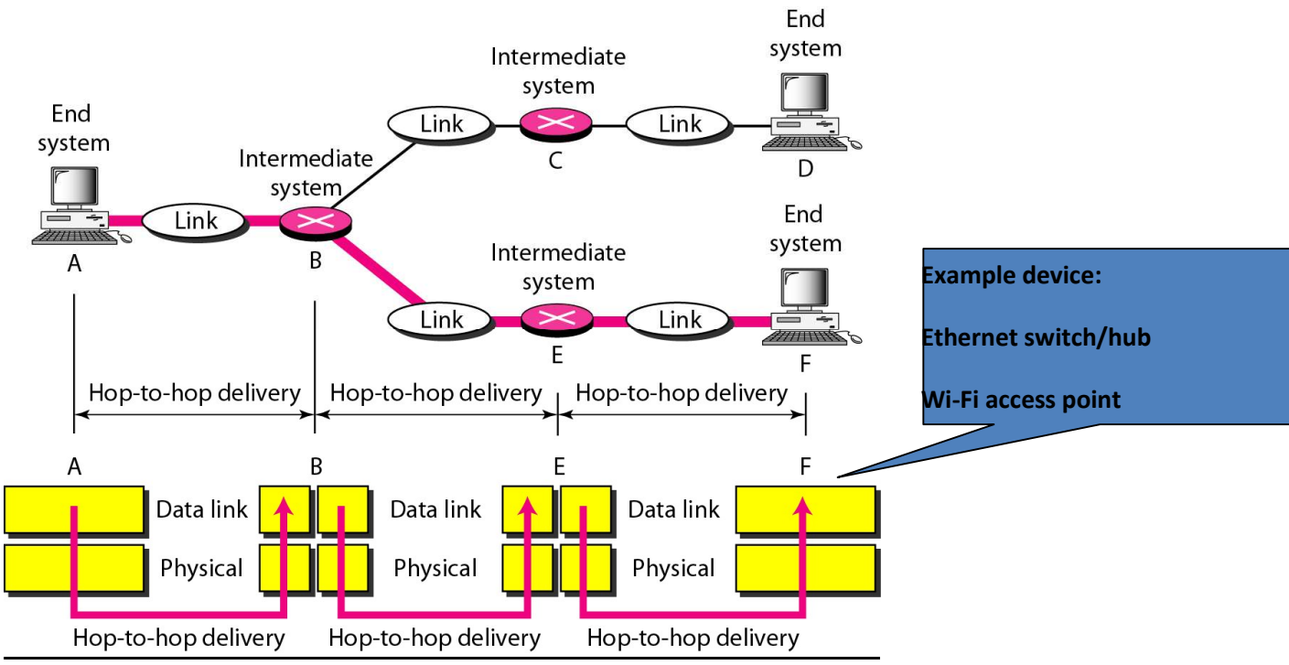
- Add mechanisms to detect and retransmit damaged or lost frames.
- Prevent also duplication of frames.

- Error control is normally achieved through a trailer added to the end of frame.

→**Access control**

- When two or more devices than one device are connected to the same link, data link layer protocols are necessary to determine which device has control over the link at given time.

Hop-to-hop delivery for data link layer



Physical address: -

→Known also as the MAC or link address

→Is the address of a node as defined by its LAN or WAN

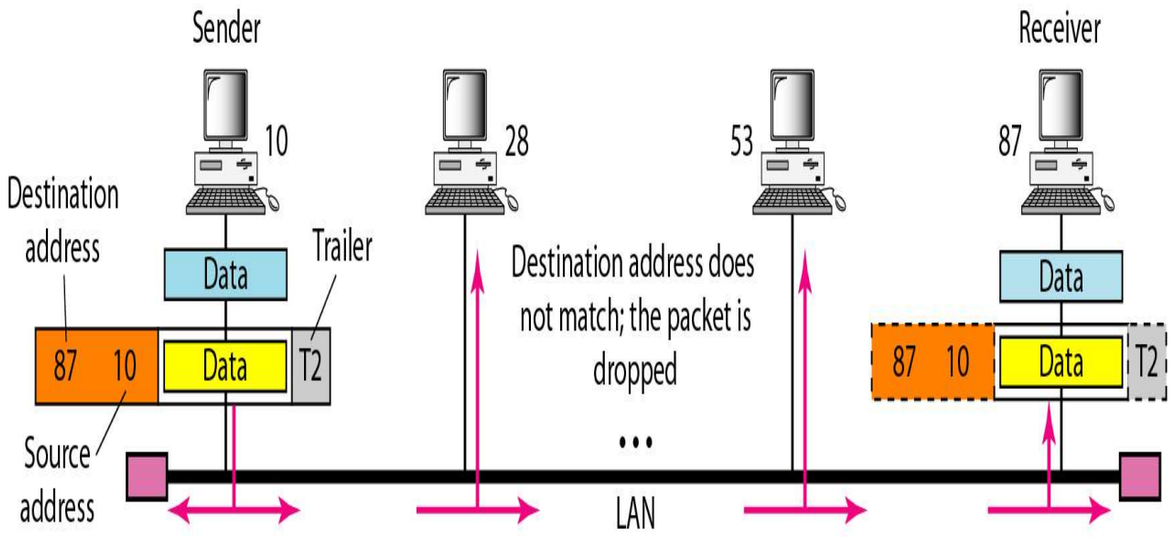
→It is included in the frame used by data link layer(Header)

→Ethernet uses 6-bytes (48bits) physical address that imprinted on the NIC

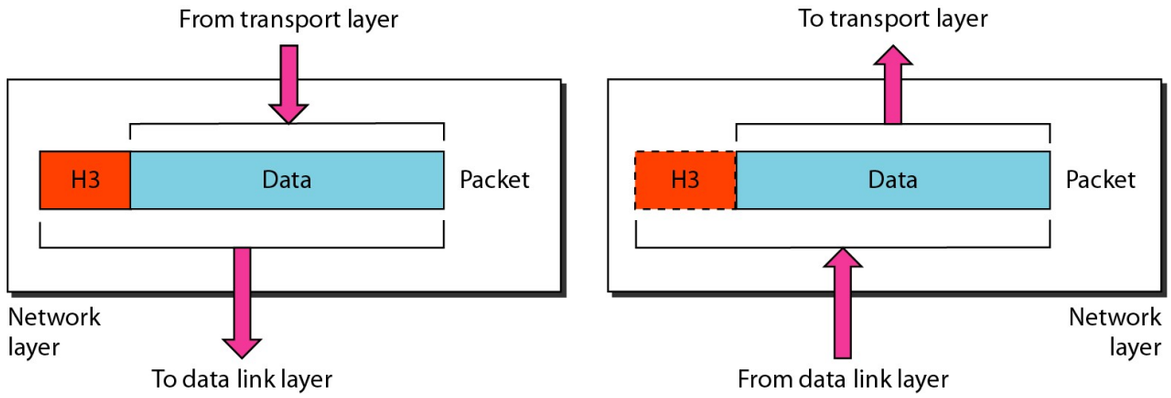
Example 1:-

A node with physical address 10 sends a frame to a node with physical address 87. The two nodes are connected by a link. At the data link level this frame contains physical addresses in the header. These are the only addresses needed. The rest of the header contains other information needed at this level. The trailer usually contains extra bits needed for error detection?

Example1 :Physical addresses



Network layer:-



Note

The network layer is responsible for the delivery of individual packets from the source host to the destination host.

The two hosts can be many hops away

The data link layer is responsible for moving *frames* from one hop (node) to the next.

Single hop delivery

Duties of network layer

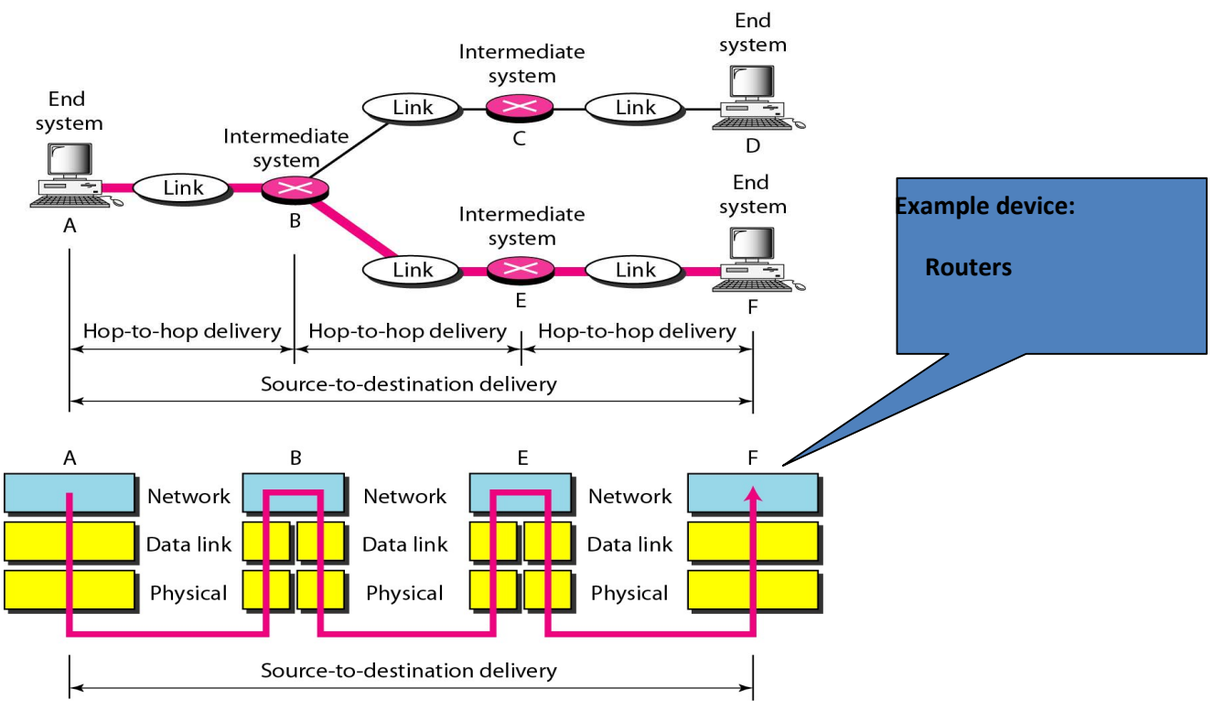
→**Logical addressing: IP addresses:-**

- In contrast to physical addressing implemented by data link layer handling the addressing problem locally. Network layer adds unique identifier (IP or logical address) to the packet.
- These unique identifier (as tel. no, each tel. has unique number) enable special devices called router to make sure the packet get to correct system.

→**Routing:-**

- Provide the routing mechanism for the router which route the packet to their final destination.
- Routers: devices used when independent networks are connected to create an internetworking (network of networks)

Source-to-destination delivery : -



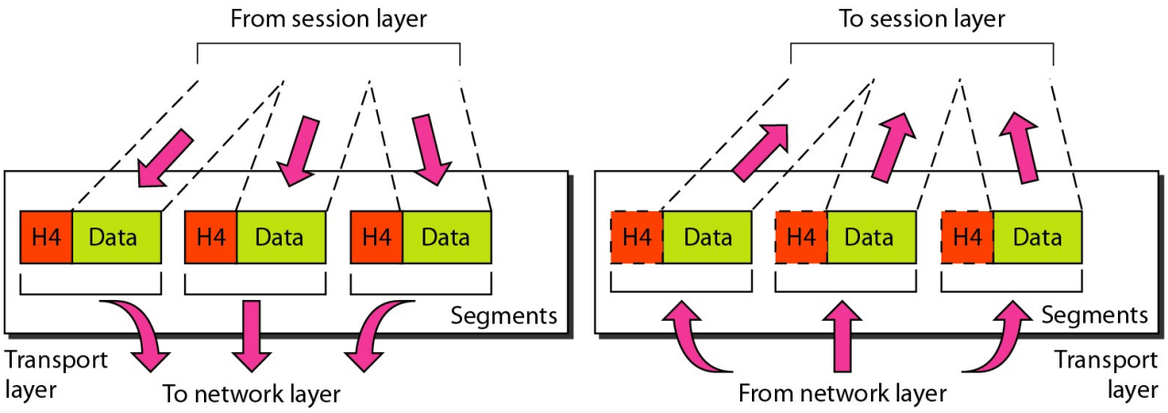
Logical address (IP):-

- Communications that are independent of physical network.
- No two-host address on the internet can have the same IP address
- IPaddresses32 bit address that uniquely define a host connected to the Internet

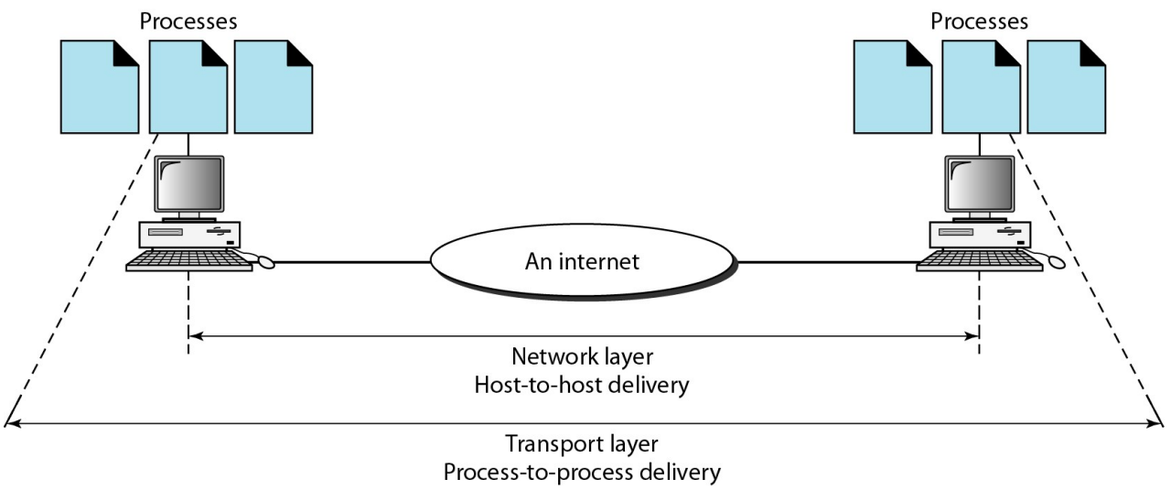
The physical addresses will change from hop to hop, but the logical addresses usually remain the same.

Transport layer

The transport layer is responsible for the delivery of a message from one *process* to another.



Reliable process-to-process delivery of a message



Duties of transport layer:-

1. Port addressing (Service-point addressing)

- Computer often run several process (running programs) at the same time, so the process to process delivery means delivery from a specific process on a computer to specific process to the other.
- The transport layer header must include Port address(16-bit addresses represented by decimal number range from 0-65535) to choose among multiple processes on the destination host
- Destination port No is needed for delivery
- Source port no is needed for replay.

2. Segmentation and reassembly:

- A message is divided into small pieces (Segment), each segment containing sequence number. These number enable the transport layer to reassemble the message correctly at destination and to identify and replace segment that were lost in transmission.

3.Flow control:

- Like the data link layer, transport layer responsible for flow control. Flow control at this layer is performed end to end rather than across a signal link.

4. Error control:

- Error control at this layer is performed process-to-process rather than across a single link
- The sending transport layer makes sure that the entire message arrives at the receiving transport layer without error (damage, loss or duplicated).
- Error correction is usually achieved through retransmission.

5.Connection control

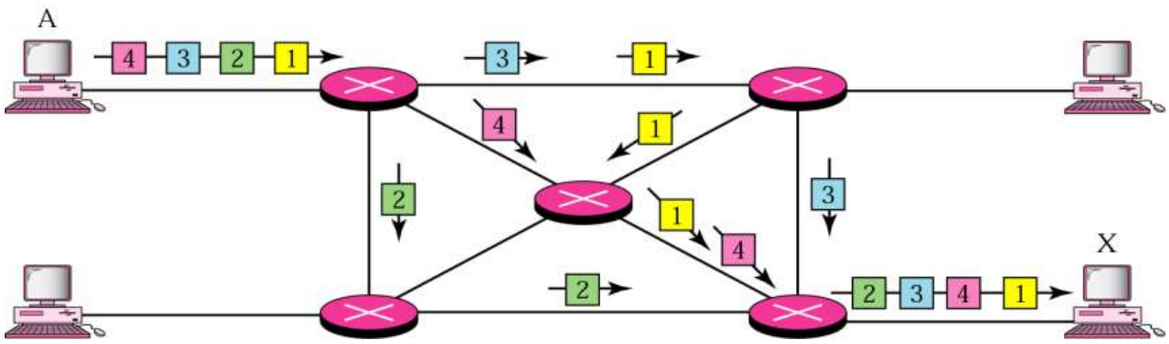
The transport layer can be either connection less or connection oriented

Connection oriented

- Makes a connection with the transport layer at the destination machine first before delivering the packers.
- When the connection established a sequence of packets from source to the destination can be sent one after another on the same path and in sequential order.
- When all packets of message have been delivered, the connection is terminated
- This makes the sending transport **layer ensure that the message arrives at the receiving transport layer without error (damage, loss or duplication)**

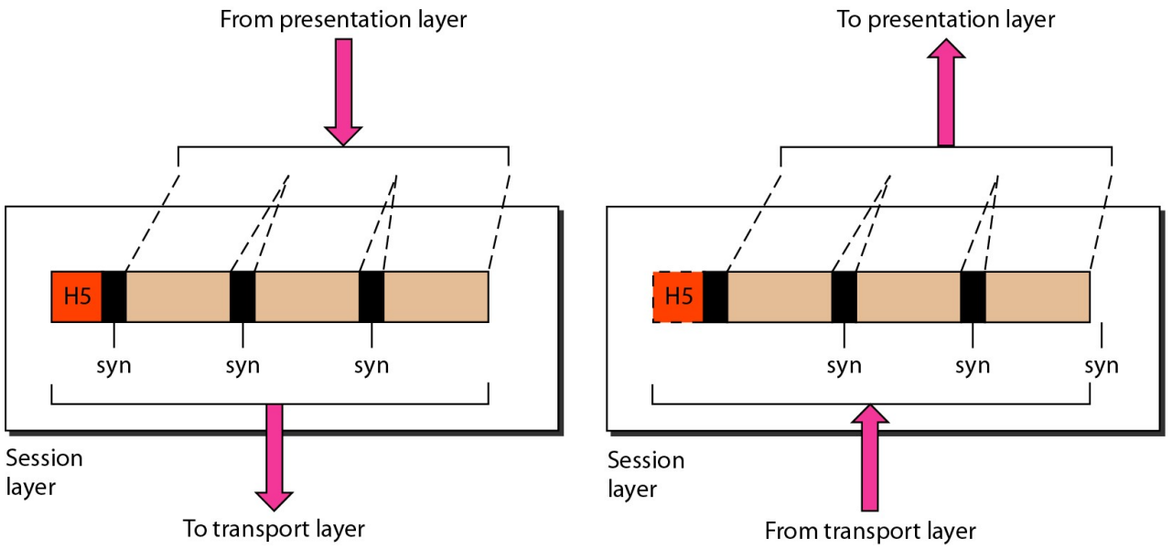
Connection Less (as Internet):-

- It sends the data, but does not establish and verify a connection between hosts before sending data.
- Treats each packet independently, the packets in a message may or may not travel the same path to their destination.



Session layer:-

The session layer is responsible for dialog control and synchronization.



Duties of Session layer

1. Dialog control:

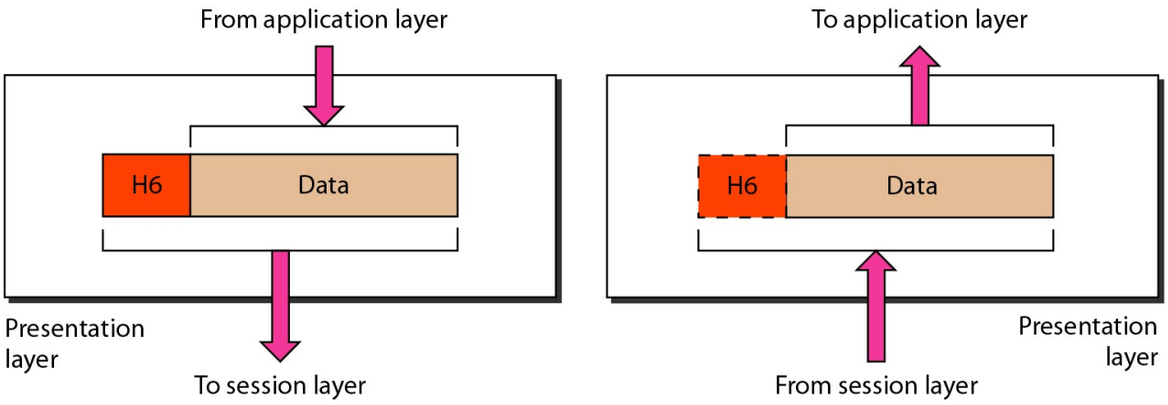
Allows two systems to enter into dialog. It allows communication between two processes in either half or full duplex.

2. Synchronization (Recovery)

Allow a process to add **check points (Synchronization point)** into a stream of data . So that if a failure of some sort occurs between checkpoints, the layer can retransmit all data since the last checkpoint.

Presentation layer:-

The presentation layer is responsible for translation, compression, and encryption.



Duties of presentation layer:-

1.Translation

At the sender it changes the information from its sender –dependent format into common format. At receiving, changes the common format into its receiver-dependent format

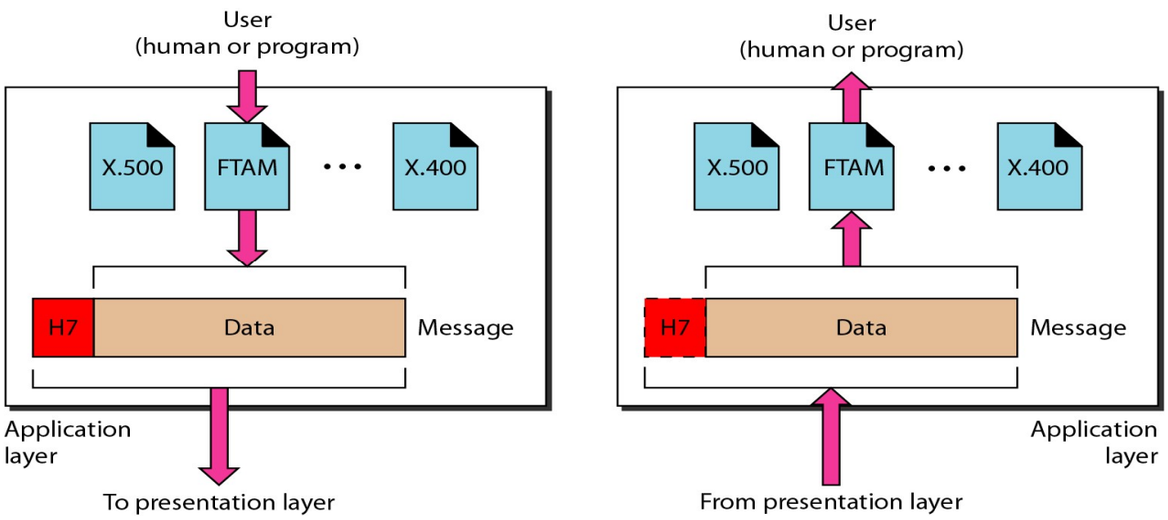
2. Encryption-Decryption

To ensure privacy and security

3.Compression

Data compression reduces the number of bits contained in the information. It is important in the transmission of multimedia such as audio or video

Application layer:-



→The application layer is responsible for providing services to the user.

Specific services provided by the application layer include the following:

→**Network virtual terminal**. A network virtual terminal is a software version of a physical terminal, and it allows a user to log on to a remote host. To do so, the application creates a software emulation of a terminal at the remote host. The user's computer talks to the software terminal which, in turn, talks to the host, and vice versa. The remote host believes it is communicating with one of its own terminals and allows the user to log on.

→**File transfer, access, and management**. This application allows a user to access files in a remote host (to make changes or read data), to retrieve files from a remote computer for use in the local computer, and to manage or control files in a remote computer locally.

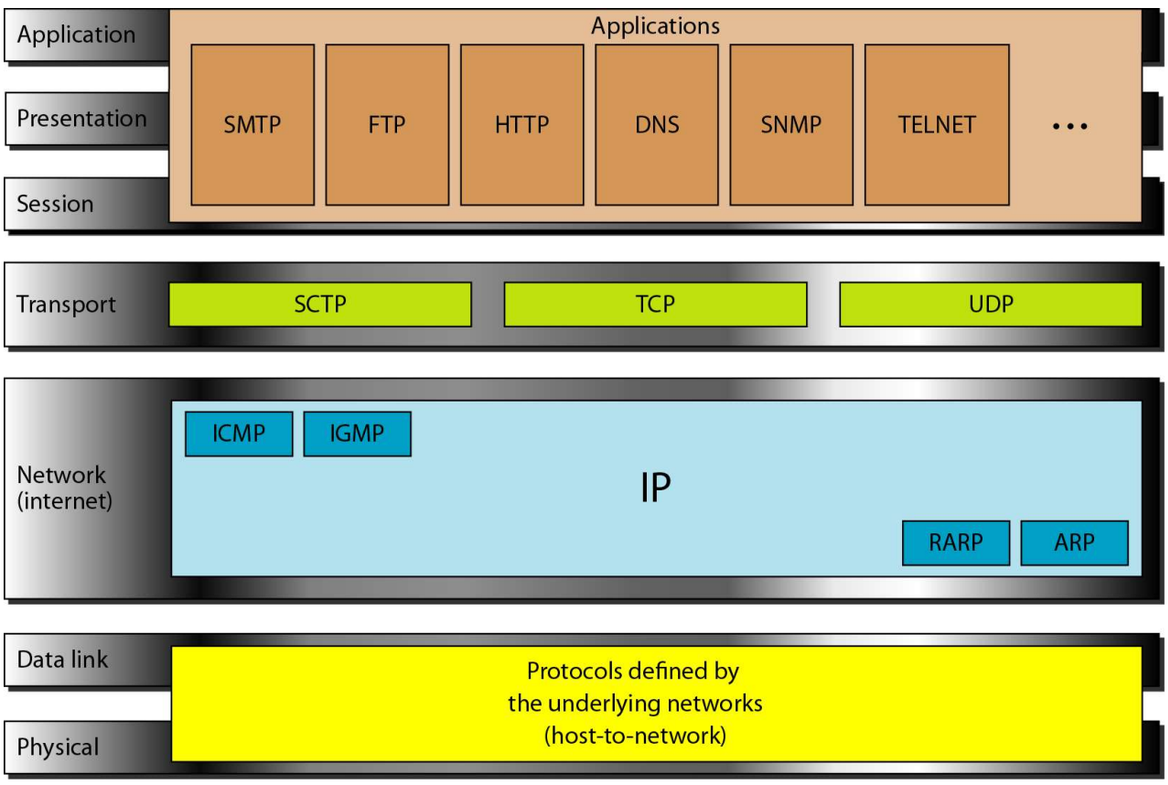
→**Mail services.** This application provides the basis for e-mail forwarding and storage.

→**Directory services.** This application provides distributed database sources and access for global information about various objects and services.

TCP/IP Protocol Suite:-

TCP/IP model is sometimes called as internet reference model. This model was developed before the OSI reference model's model does not reflect the real-world protocol architecture.

Basically TCP/IP consists of four layers, but it is now viewed as five-layer model.



Layers of TCP/IP Model

1. Host to network layer.
2. Internet network layer.
3. Transport layer.
4. Application layer.

1. Host to network layer:-

The physical layer and datalink layer of OSI references model are combined to form host-to-network layer of TCP/IP protocol suite. It performs the function similar to that of physical and datalink layer. Ethernet, token ring, repeaters, hubs, cables are the standard devices that are used in this layer. It will perform the function of datalink layer

such as appending a packet header to make it ready for transmission over the physical medium.

2. Internet network layer: -

The protocol that is used in network layer is the internet protocol. Internet protocol is responsible for delivering data packet from source host to destination host. Internet protocol is connectionless protocol therefore each packet travels independently of the other. IP protocol is used to provide a function for routing the packets. The other protocols that are supported by network layer are as follows,

→ Internet Control Message Protocol (ICMP)
→ Internet Group Message Protocol (IGMP)
→ Address Resolution Protocol (ARP)
→ Reverse Address Resolution Protocol (RARP)
→ Internet Control Message Protocol (ICMP)

It is a protocol that is used to control the flow of message over transmission medium by sending notification messages.

Internet Group Message Protocol (IGMP)

It is a protocol that is used to message multicast packets.

Address Resolution Protocol (ARP)

It is used to determine the physical address when logical address is known.

Reverse Address Resolution Protocol (RARP)

It is used to determine the logical address when physical address is known.

3.Transport Layer

Transport layer performs the function similar to that of ISO/OSI transport layer function. Its function includes end-to-end delivery of entire message without any errors. The protocols that are used in transport layer are,

- TCP (Transmission Control Protocol)
- UDP (User Datagram Protocol)

TCP

TCP is a connection oriented reliable protocol. Using TCP data arrives in sequence order with minimal error. If there is any duplicate or error in data packets, then the whole stream of the error packets is discarded and is retransmitted from the packet that contained an error. TCP include congestion control parameters for controlling the congestion.

UDP

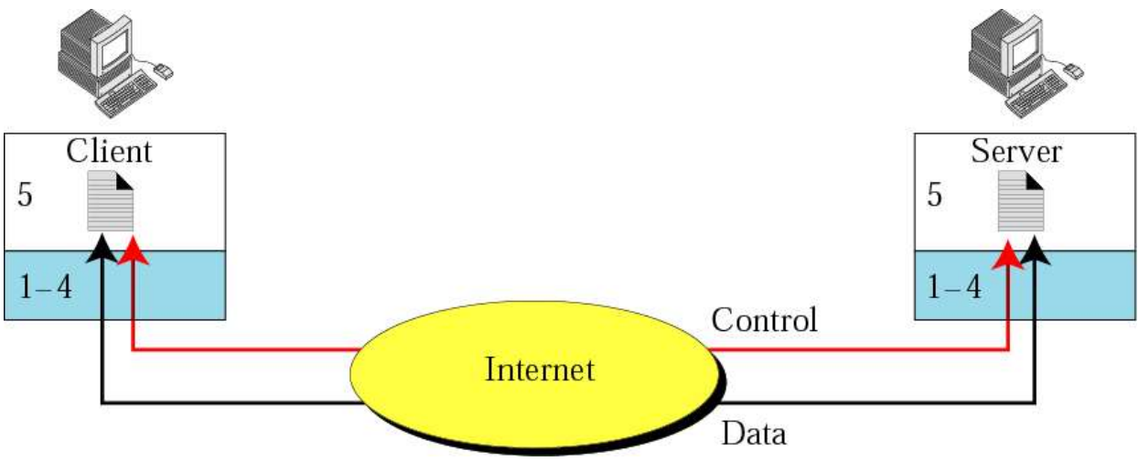
UDP is a connection less unreliable protocol. Data is transmitted in the form of datagram which are independent of routes. UDP is less reliable because it have weak error correction that is checksum and error detection. Algorithm UDP is used for transmitting real-time data such as audio, video where time is important than reliability.

4.Application Layer

Data is converted into format that is specific to application which is then encapsulated with a header to pass it down to the lower layer which is transport layer. The protocol that are present in this layer are FTP, Telnet, HTTP, NNTP.

File Transfer Protocol (FTP):-

The standard protocol on the Internet for transferring a file from one machine to another is the File Transfer Protocol (FTP). FTP was designed to respond to traditional problems related to file transfer, one problem is the different coding systems in use; one machine may use ASCII, and the other may use Unicode. Another problem is the different file formats in use. FTP was designed to resolve these problems.



Simple Mail Transfer Protocol (SMTP)

By far the most popular application on the Internet today is electronic mail (email). The protocol that supports email on the Internet is Simple Mail Transfer Protocol (SMTP). To send and receive email, the user must install both client and server SMTP software on a computer. SMTP is often used with another protocol such as Post Office Protocol (POP).

Hypertext Transfer Protocol (HTTP)

Hypertext Transfer Protocol (HTTP) is a client-server program that is used to access and transfer documents in the World Wide Web.

Summary:-

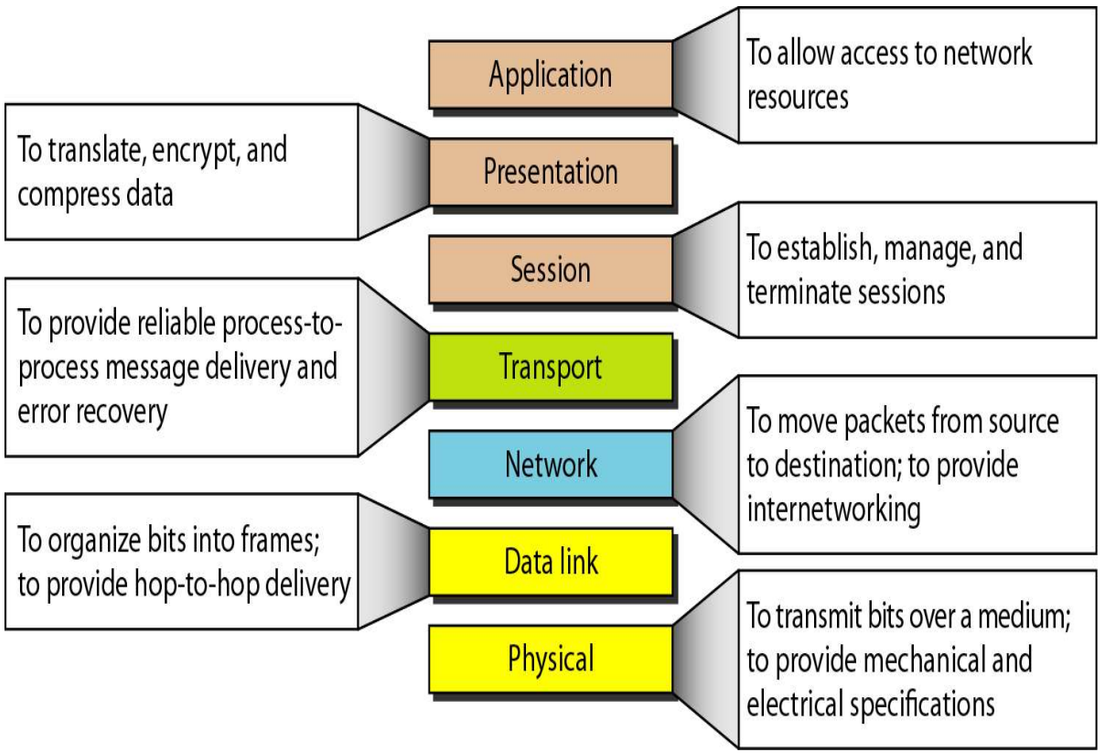
- The Internet Protocol (IP) is TCP/IP's unreliable protocol at the internet layer.
- An IP address identifies each computer connected to the Internet.
- User Datagram Protocol (UDP) and Transmission Control Protocol (TCP) are protocols at the transport layer.
- File Transfer Protocol (FTP) is a TCP/IP client-server application for coping files from one host to another.
- The protocol that supports electronic mail (email) on the Internet is Simple Mail Transfer Protocol (SMTP).
- TELNET is a client-server application that allows a user to log on to a remote machine, giving the user access to the remote system.
- Hypertext Transfer Protocol (HTTP) is a client-server program for accessing and transferring documents on the World Wide Web (WWW), a collection of multimedia documents.

OSI Model			
	Layer	Data Unit	Function
User support layer	Application	DATA	**Access to Network Resources**
	Presentation		**Data representation: translate compress and encryption**
	Session		**Controls the dialogues , Establishes, manages and terminates the connections between the local and remote application**
Link	Transport	Segment	**End-to-end connections and reliability**
Network support layer	Network	Datagram/Packet	**Path determination (Routing) and logical Addressing**
	Data link	Frame	**Hop to Hop delivery and Physical addressing**
	Physical	Bit	**Media, signal and binary transmission,…**

❖ DIFFERENCES BETWEEN OSI AND TCP/IP

S.no	OSI	TCP/IP
1	Strict layering	Loosely layered
2	Protocol independent standard	protocol dependent standard
3	Less credible	more credible (believable)
4	All packets are reliably delivered	TCP reliably delivers packets, IP does not reliably deliver packets
5	It is a reference model	it is a protocol suite
6	Host on OSI implementations do not participate network operations	TCP/IP hosts participate in most network protocols.
7	OSI makes the distinction between services, interfaces, and protocol.	TCP/IP does not originally clearly distinguish between services, interface, and protocol.
8	The OSI model supports both connectionless and connection-oriented communication in the network layer, but only connection-oriented communication in the transport layer	The TCP/IP model has only one mode in the network layer (connectionless) but supports both modes in the transport layer, giving the user choice.

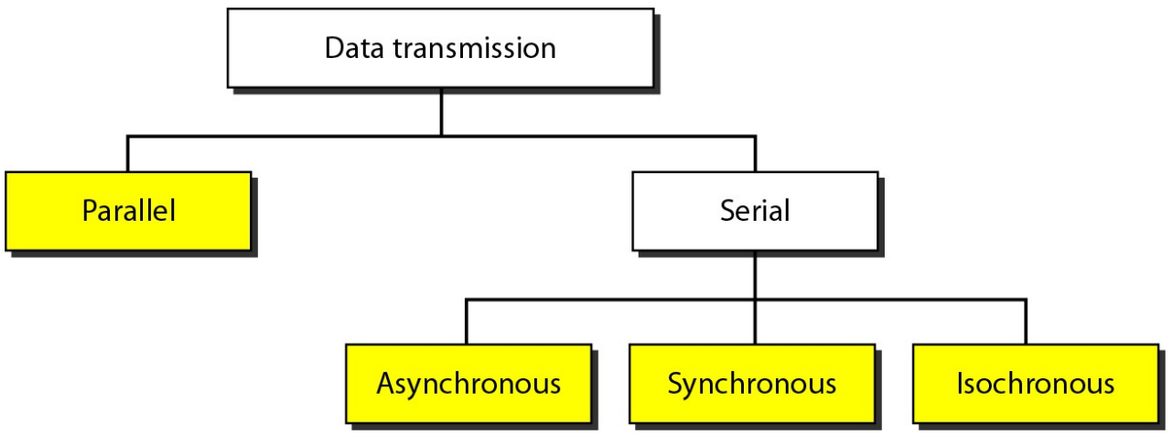
Summary of layers:-



CHAPTER-3
Digital Data Communication Techniques

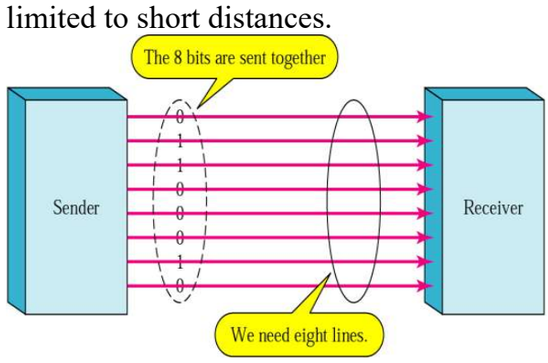
*Transmission Mode: -

The transmission of binary data across a link can be accomplished in either parallel or serial mode. In parallel mode, multiple bits are sent with each clock tick. In serial mode, 1 bit is sent with each clock tick. While there is only one way to send parallel data, there are three subclasses of serial transmission: asynchronous, synchronous, and isochronous.



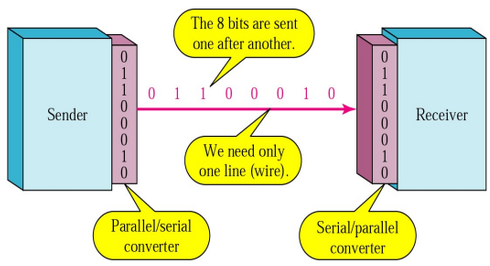
- **Parallel: -**

→Bits in a group are sent simultaneously, each using a separate link
→n wires are used to send n bits at one time
→Advantage: speed
→Disadvantage: cost; limited to short distances.



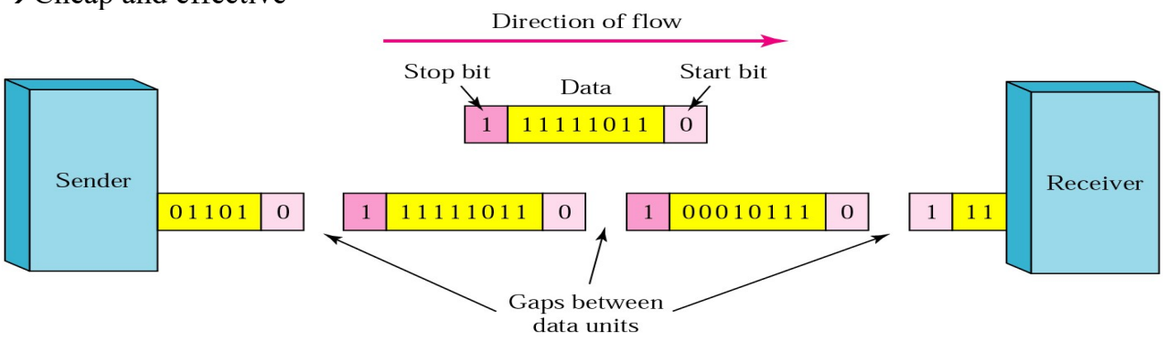
- **Serial: -**

→Transmission of data one bit at a time using only one single link
→**Advantage**: reduced cost
→**Disadvantage**: requires conversion devices
→Methods:
→Asynchronous
→Synchronous



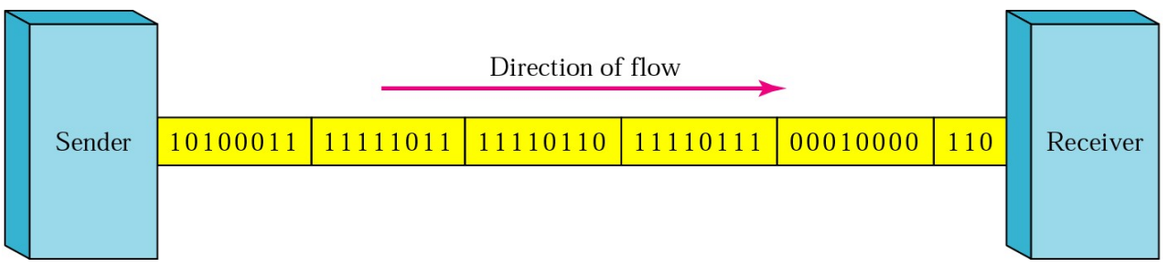
***Asynchronous Transmission: -**

→Slower, ideal for low-speed communication when gaps may occur during transmission (ex: keyboard)
→Transfer of data with start and stop bits and a variable time interval between data units
→Timing is unimportant
→Start bit alerts receiver that new group of data is arriving
→Stop bit alerts receiver that byte is finished
→Synchronization achieved through start/stop bits with each byte received ➔ Requires additional overhead (start/stop bits)
→Cheap and effective

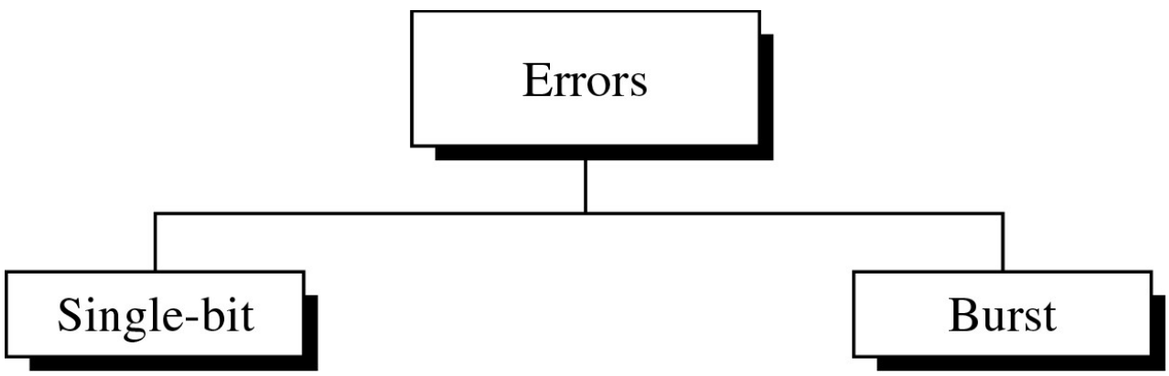


***Synchronous Transmission: -**

→Bit stream is combined into longer frames, possibly containing multiple bytes
→Requires constant timing relationship
→Any gaps between bursts are filled in with a special sequence of 0s and 1s indicating idle
→Advantage: speed, no gaps or extra bits
→Byte synchronization accomplished by data link layer.



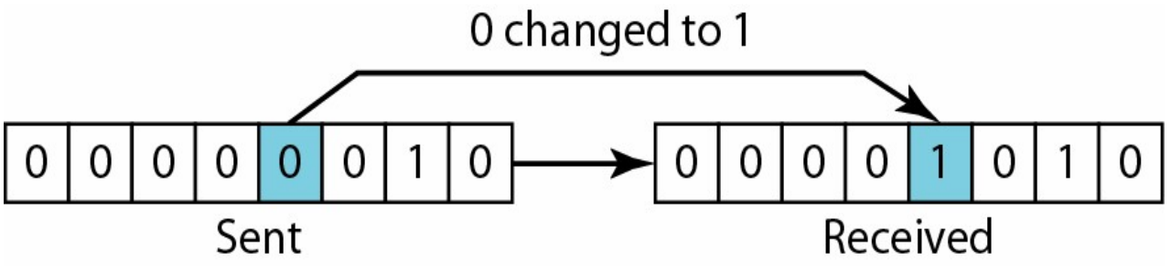
Types of Errors: -



❖ **Single-Bit Error**

→The term single-bit error means that only 1 bit of a given data unit (such as a byte,

→character, or packet) is changed from 1 to 0 or from 0 to 1.

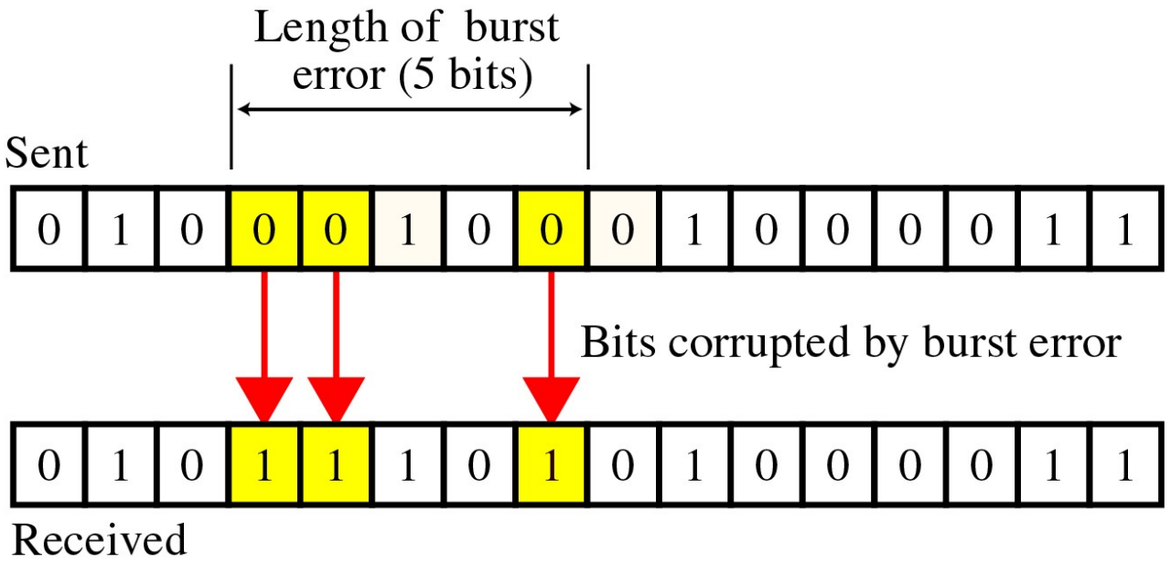


❖ **Burst Error:-**

→The term burst error means that 2 or more bits in the data unit have changed from 1 to 0 or from 0 to 1.

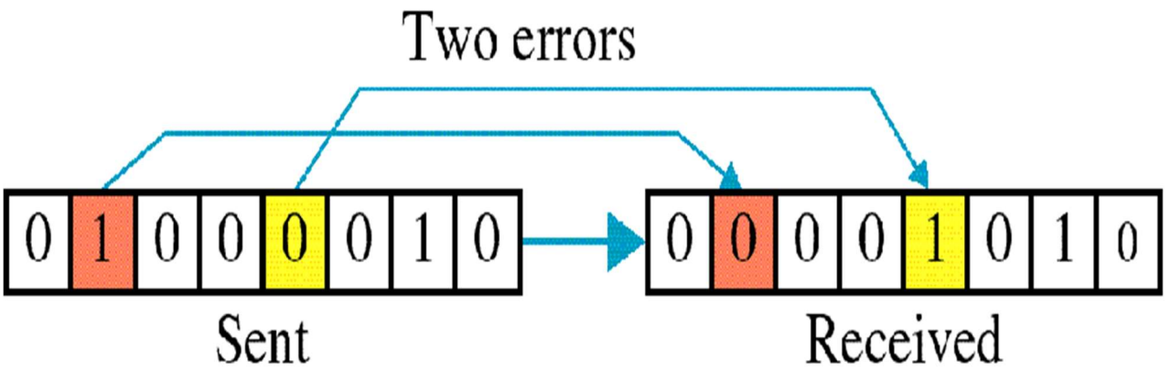
→a burst error does not necessarily mean that the errors occur in consecutive bits.

→The length of the burst is measured from the first corrupted bit to the last corrupted bit.



❖ **Multiple-Bit Error:-**

→Is when two or more nonconsecutive bits in the data unit have changed.

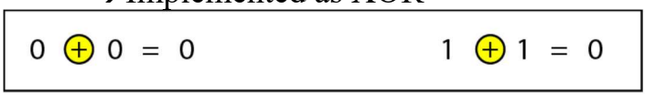


::Error Detection::

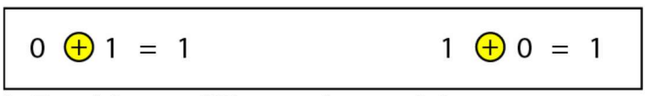
Modular Arithmetic: -

→In modulo-N arithmetic, we use only the integers in the range 0 to N −1, inclusive.

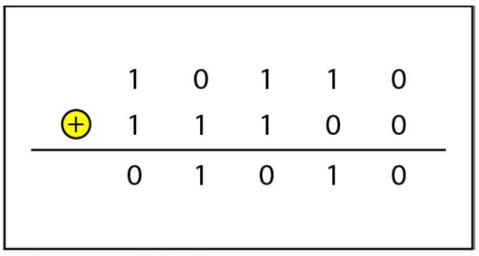
- Modulo-2 Arithmetic
  - →N is 2
  - →Use 0 and 1
  - →Implemented as XOR



a. Two bits are the same, the result is 0.



b. Two bits are different, the result is 1.



c. Result of XORing two patterns

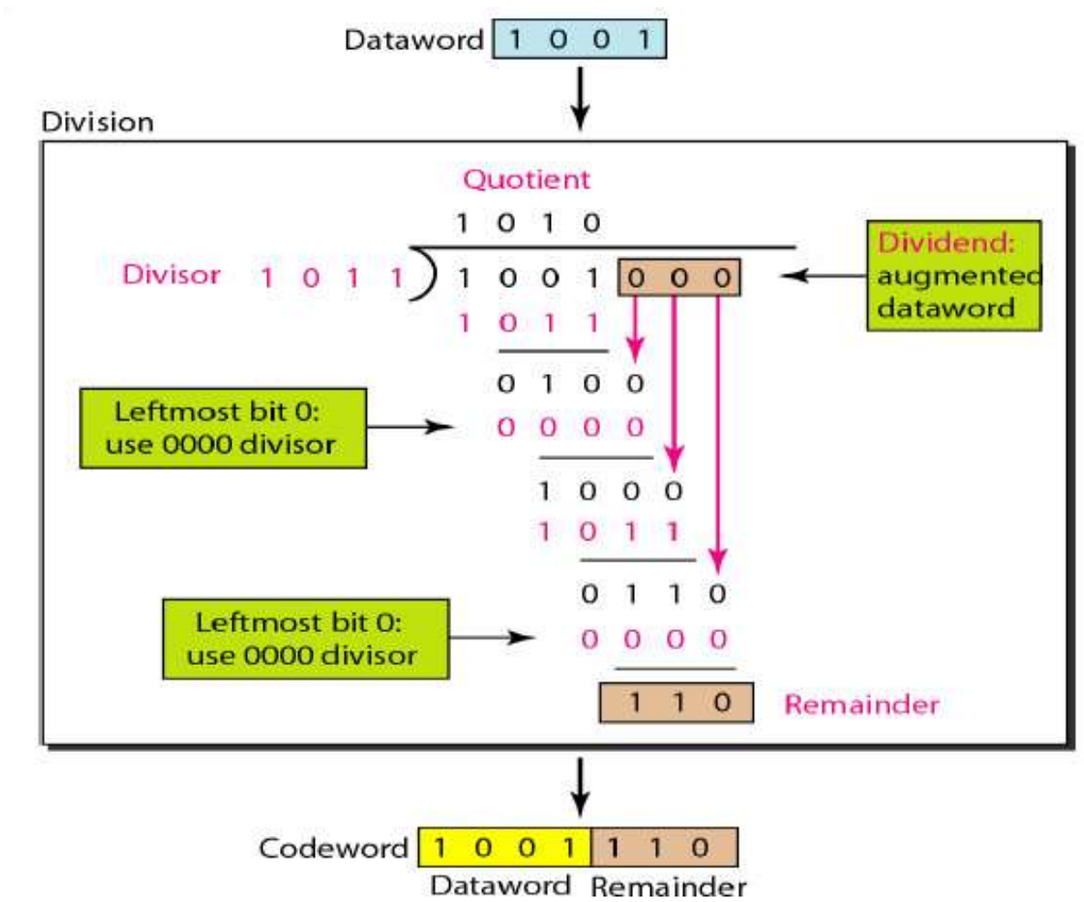
Cyclic Redundancy Check:-

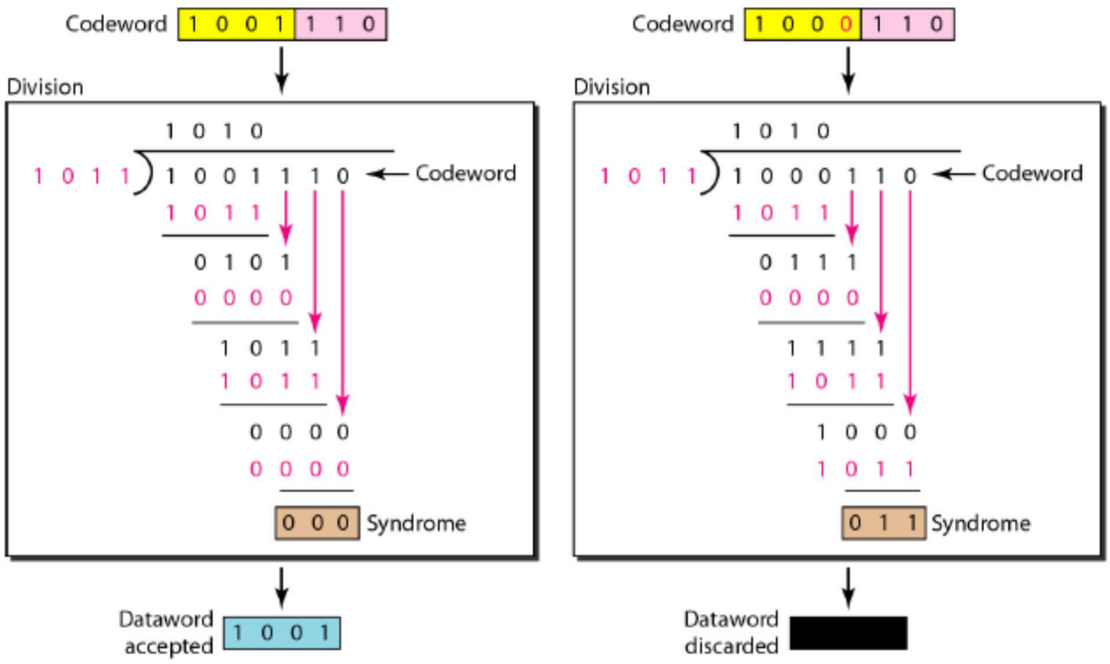
→We can create cyclic codes to correct errors.

→we simply discuss a category of cyclic codes called the cyclic redundancy check (CRC) that is used in networks such as LANs and WANs.

Division in CRC Encoder: -

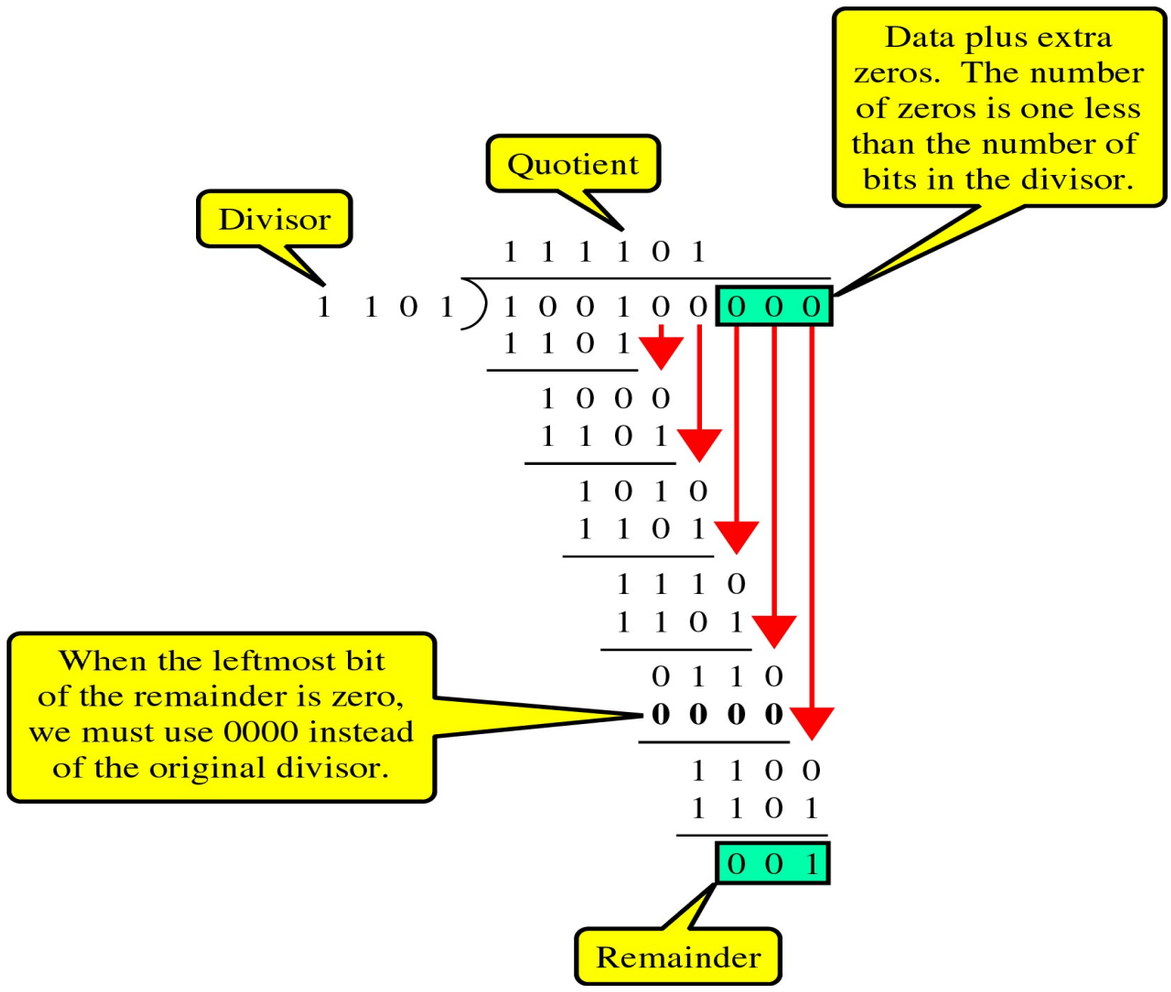
Example-1: -



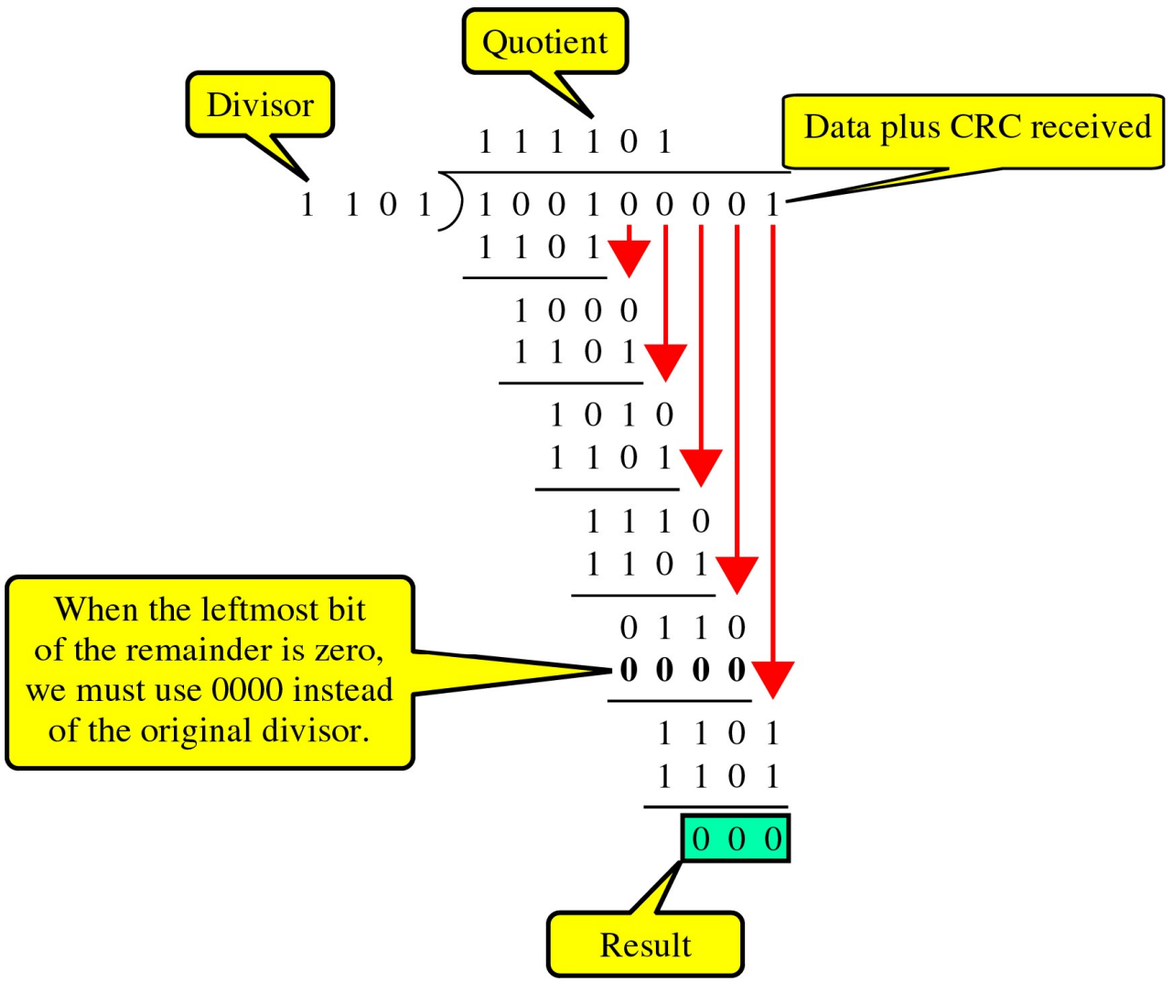


Example-2:-

→Binary Division in a CRC Generator



→Binary Division in a CRC Checker

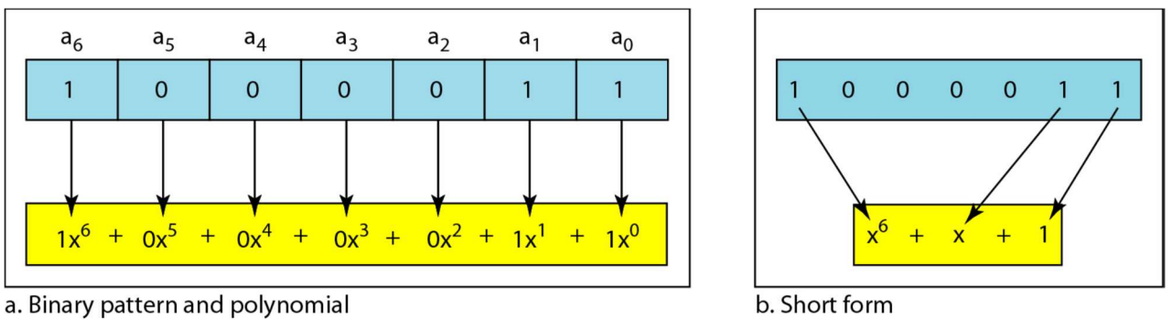


❖ **Polynomials**

→A better way to understand cyclic codes and how they can be analyzed is to represent them as polynomials.

→CRC generator(divisor) is most often represented not as a string of 1s and 0s, but as an algebraic polynomial.

Example-1:-



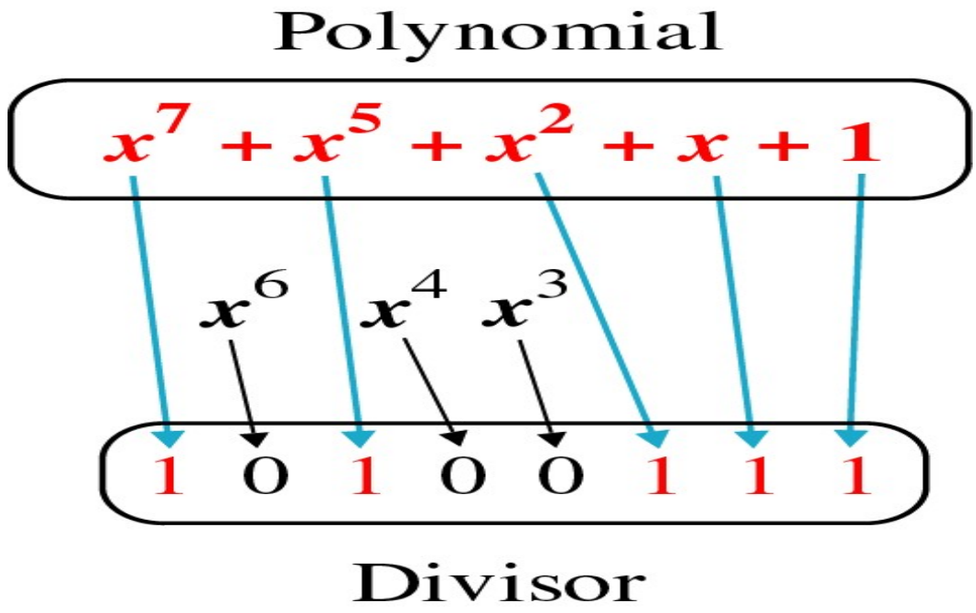
a. Binary pattern and polynomial

b. Short form

Example-2:-

$$x^7 + x^5 + x^2 + x + 1$$

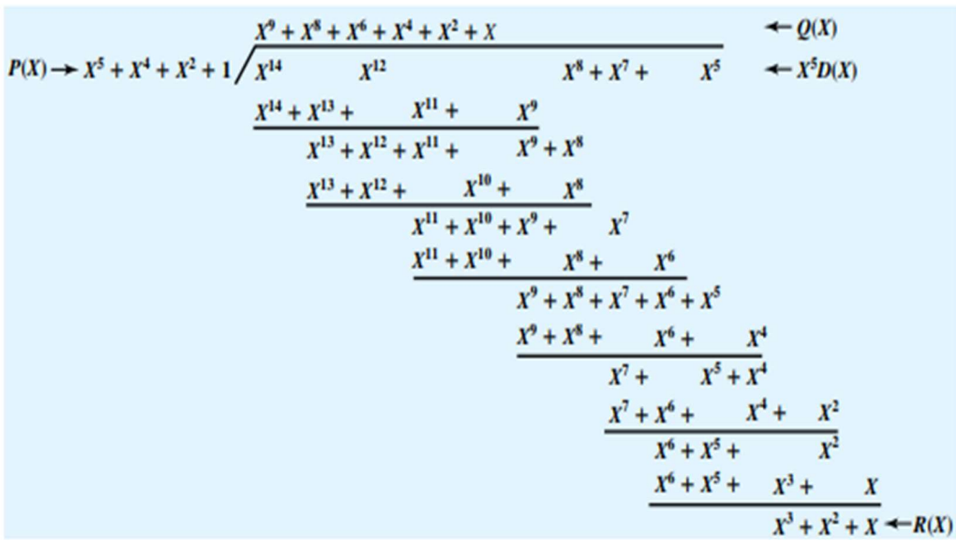
Ans:-



POLYNOMIAL DIVISION:-

EXAMPLE 6.7 Using the preceding example, for $D = 1010001101$, we have $D(X) = X^9 + X^7 + X^3 + X^2 + 1$, and for $P = 110101$, we have $P(X) = X^5 + X^4 + X^2 + 1$. We should end up with $R = 01110$, which corresponds to $R(X) = X^3 + X^2 + X$. Figure 6.4 shows the polynomial division that corresponds to the binary division in the preceding example.

ANS:-



::Error Correction::

~ can be handled in two ways

→when an error is discovered, the receiver can have the sender retransmit the entire data unit.

→a receiver can use an error-correcting code, which automatically corrects certain errors.

Single-Bit Error Correction

→parity bit

→The secret of error correction is to locate the invalid bit or bits

→For ASCII code, it needs a three-bit redundancy code(000-111)

Redundancy Bits

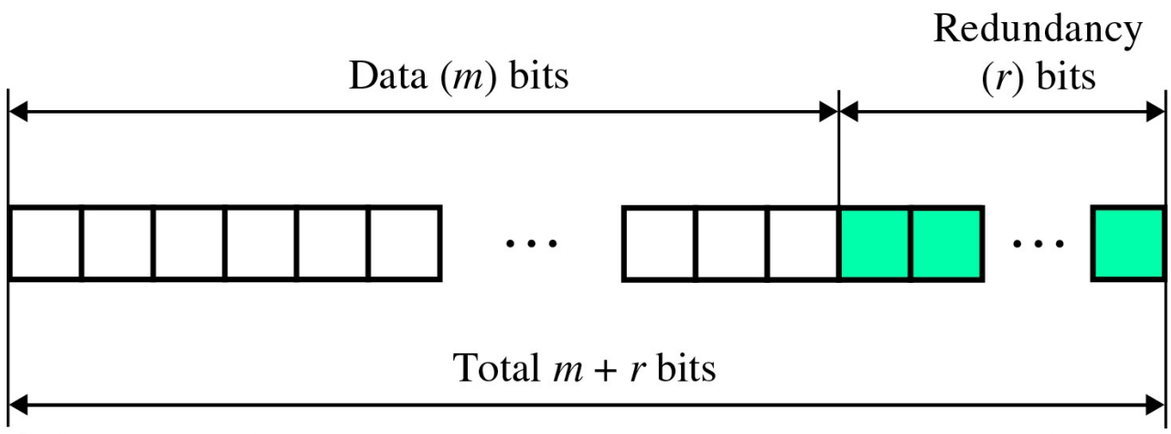
→to calculate the number of redundancy bits (R) required to correct a given number of data bit (M)

→The central concept in detecting or correcting errors is redundancy.

→To be able to detect or correct errors, we need to send some extra bits with our data.

→These redundant bits are added by the sender and removed by the receiver.

→Their presence allows the receiver to detect or correct corrupted bits.



→If the total number of bits in a transmittable unit is m+r, then r must be able to indicate at least m+r+1 different states

$$2^r \geq m + r + 1$$

ex) For value of m is 7(ASCII), the smallest r value that can satisfy this equation is 4

$$2^4 \geq 7 + 4 + 1$$

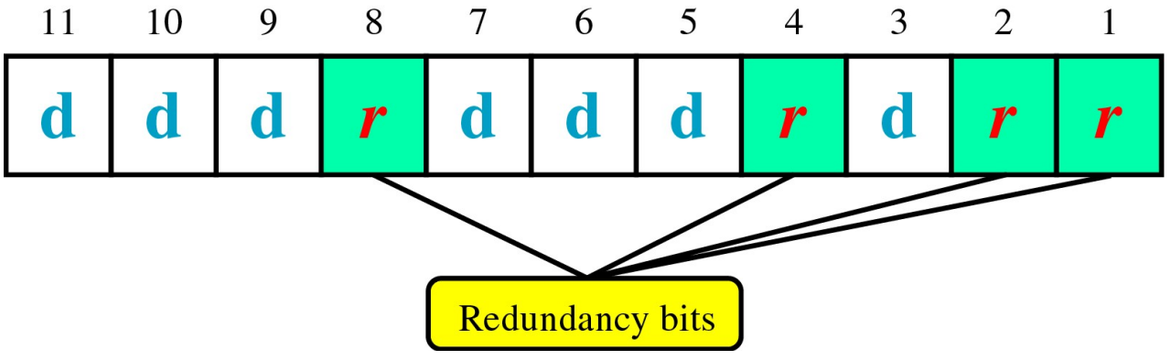
❑ **Relationship between data and redundancy bits**

Number of Data Bits (m)	Number of Redundancy Bits (r)	Total Bits (m+r)
1	2	3
2	3	5
3	3	6
4	3	7
5	4	9
6	4	10
7	4	11

❖ **Hamming Code**

→developed by R.W.Hamming

→positions of redundancy bits in Hamming code.



→each r bit is the VRC bit for one combination of data bits

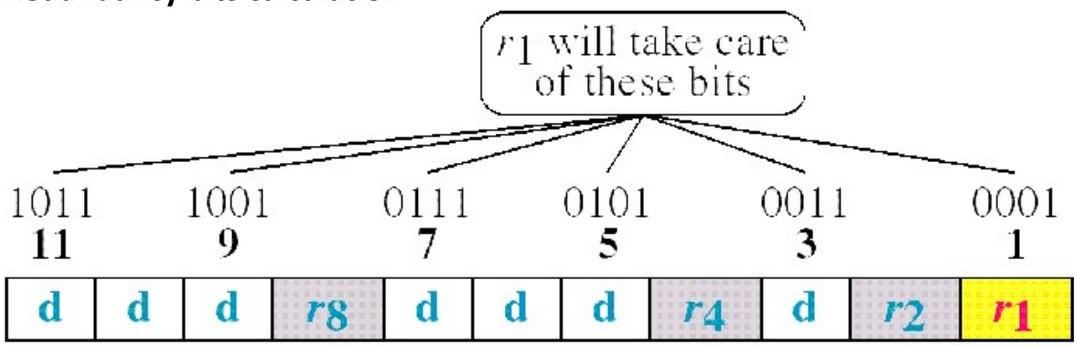
r_1 = bits 1, 3, 5, 7, 9, 11

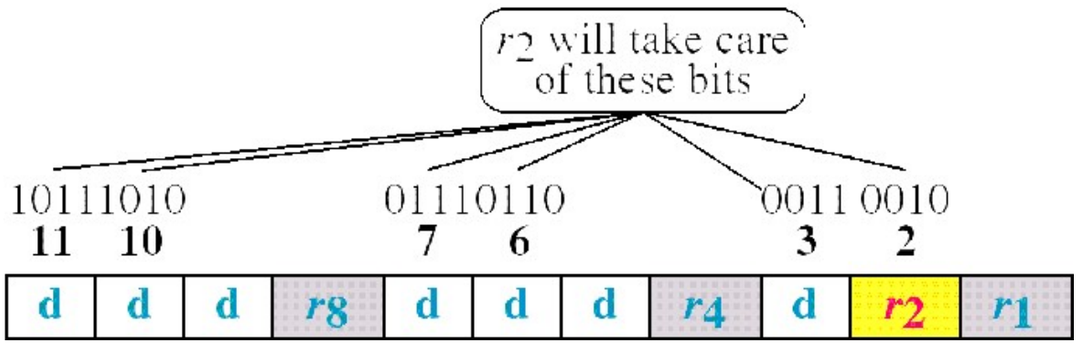
r_2 = bits 2, 3, 6, 7, 10, 11

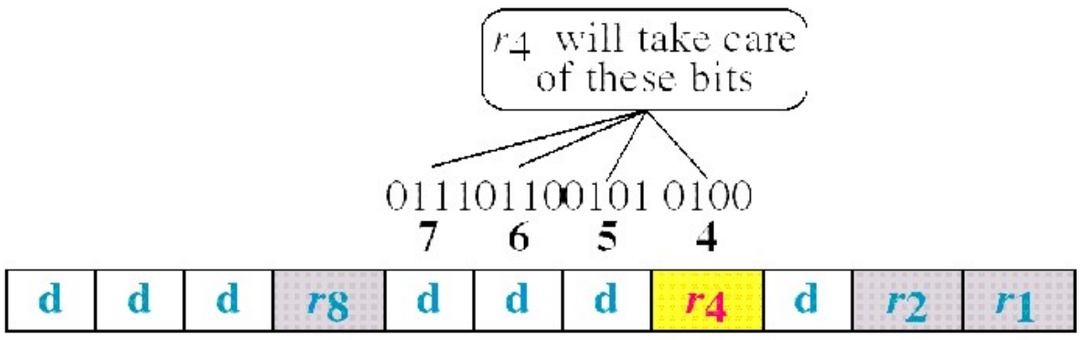
r_4 = bits 4, 5, 6, 7

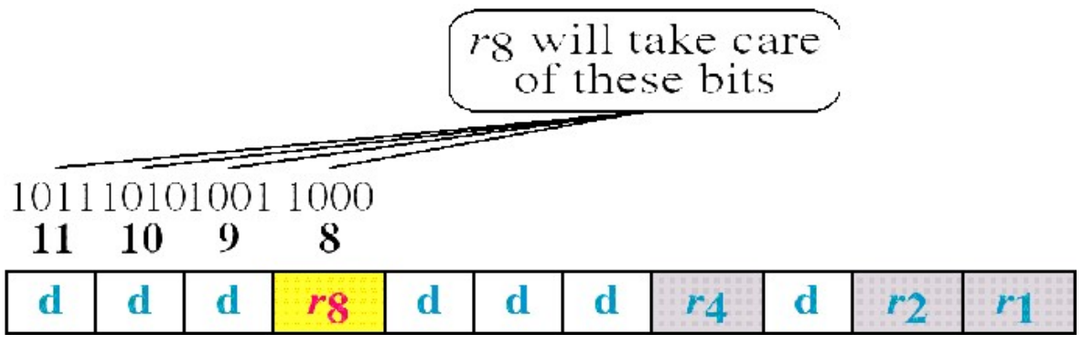
r_8 = bits 8, 9, 10, 11

- **Redundancy bits calculation**



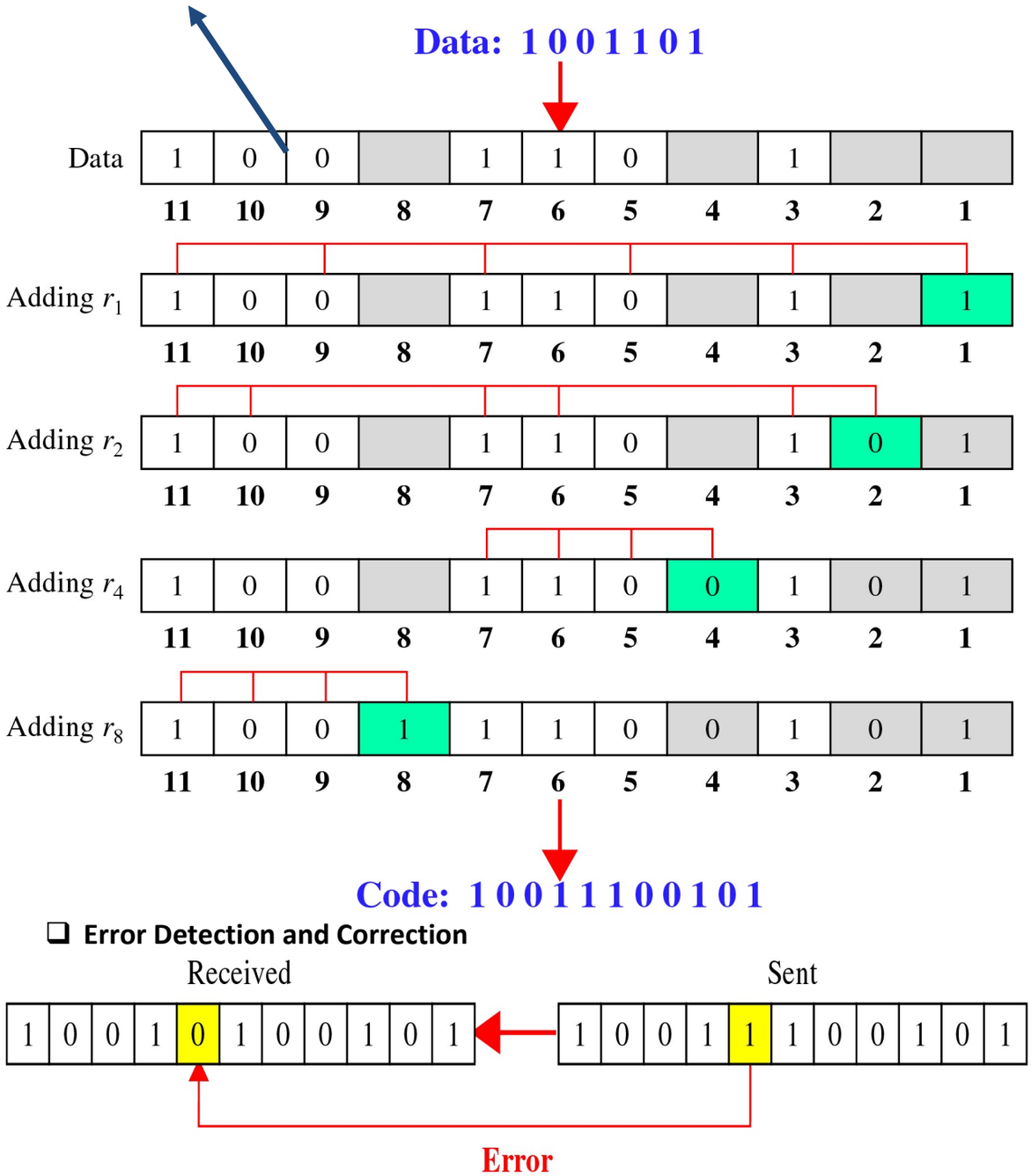






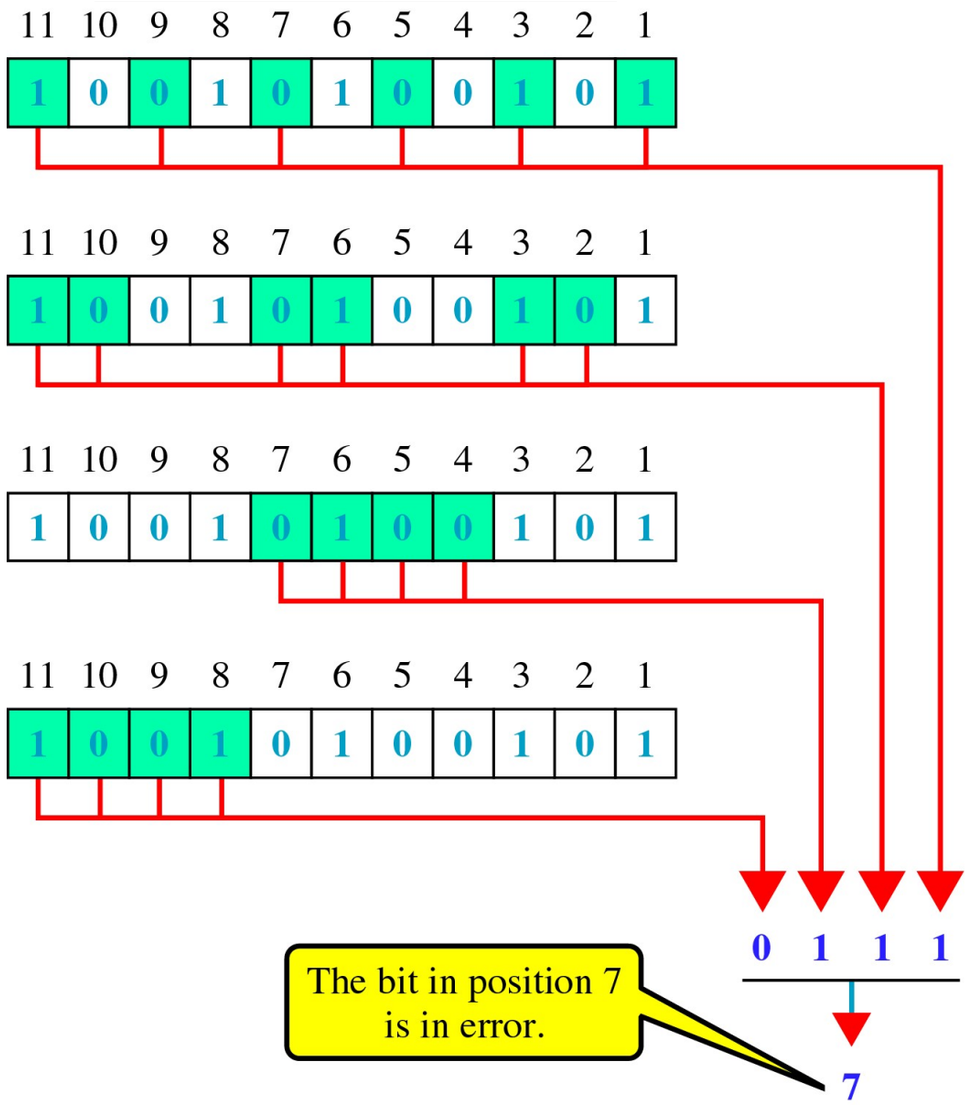
❖ **Calculating the r values**

Calculating Even Parity



❑ **Error Detection and Correction**

❑ **Error detection using Hamming Code**



→Detection Versus Correction

- In error detection, we are looking only to see if any error has occurred. The answer is a simple yes or no.
- In error correction, we need to know the exact number of bits that are corrupted and more importantly, their location in the message.
- The number of the errors and the size of the message are important factors.

→Forward Error Correction Vs Retransmission

- There are two main methods of error correction.
- Forward error correction is the process in which the receiver tries to guess the message by using redundant bits.
- Correction by retransmission is a technique in which the receiver detects the occurrence of an error and asks the sender to resend the message.

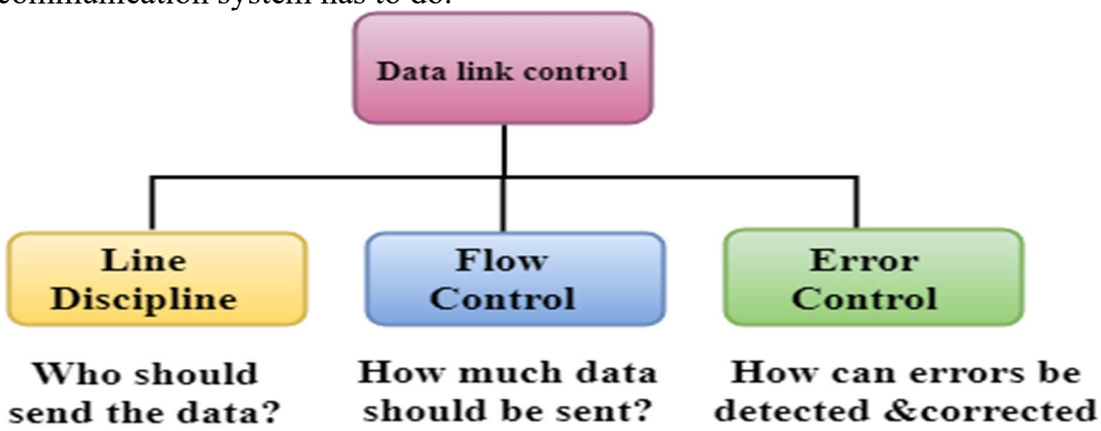
~~~~~~~~~~~~~~~~~~~~~~**END OF UNIT-1**~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~~~

UNIT-2

CHAPTER-1 :-DATA LINK CONTROL

DATA LINK CONTROL(DLC):-

DLC is the service provided by the Data Link layer of function defined in the Open Systems Interconnection OSI model for network communication. The Data Link layer is responsible for providing reliable data transfer across one physical link (or telecommunications path) within the network. Technical communication systems are developed to transmit messages from a sender to a receiver beside the task of sending and receiving information over a channel, there are many other tasks a communication system has to do.



Line Discipline:

→Coordinates the link systems, which device can send and when it can send?

Flow Control:

→The amount of data that can be sent before the receiving acknowledgment

→It also provides the receiver's acknowledgment for frames received intact and so is linked to error control

Error control:

→Means Error detection and correction

→It allows the receiver to inform the sender of any frames lost or damaged in TX and coordinates Retransmission of those frames by the sender.

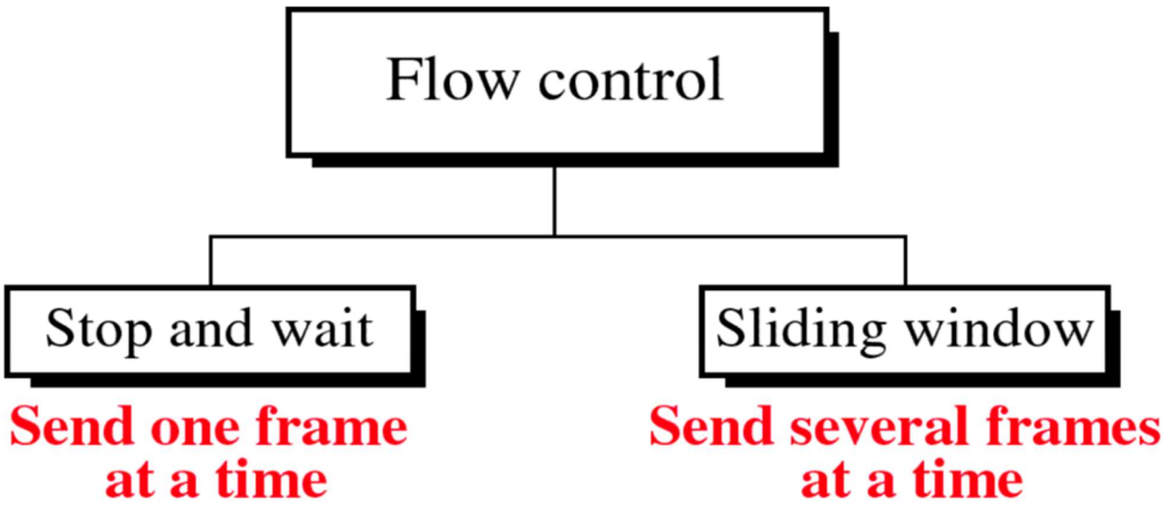
Flow control -Definition:-

→2nd aspect of data link control is Flow control.

→In most protocols,flow control is a set of procedures that tells the sender how much data it can transmit before it must wait for an ACK from the receiver

→The flow of data must not be allowed to overwhelm the receiver.

Different types of flow Controls:-



Flow Control- Explanation:-

→Any receiving device has a limited speed at which it can process incoming data and a limited amount of memory in which to store incoming data

→The receiving device must be able to inform the sending device before those limits are reached and to request that the TX device send fewer frames or stop temporarily

- **Flow Control-Buffer**

→Incoming data must be processed and checked before it can be used

→The rate of such processing is often slower than the rate of TX

→So, each receiving device has a block of memory called BUFFER, reserved for storing incoming data until it is processed

→If the buffer begins to fill up, the receiver must be able to tell the sender to halt the TX until it is once again able to receive

- **Methods for Flow Control**

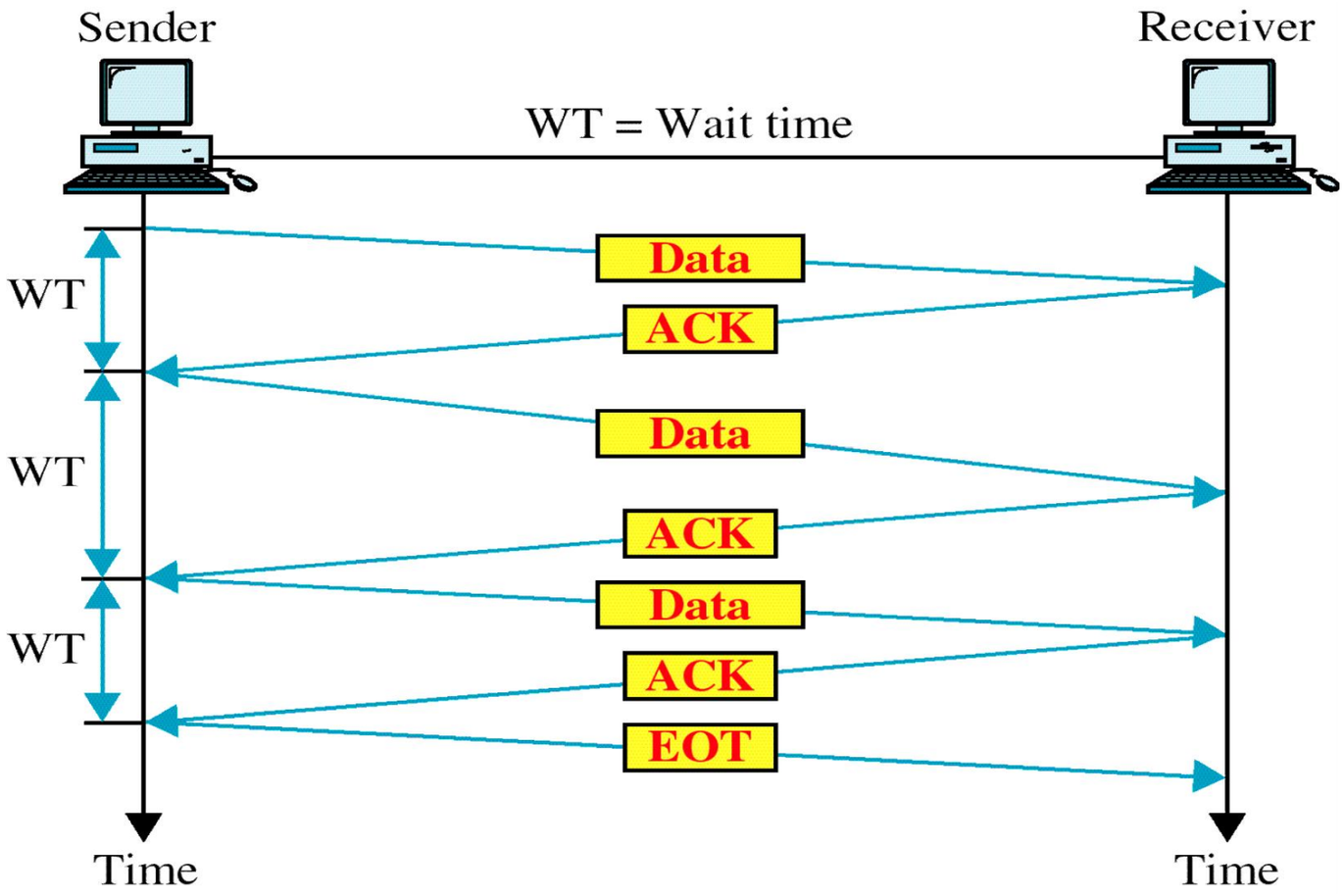
→Two methods have been developed to control the flow of data across communication links:

  - Stop and wait
  - Sliding Window

- **Stop and Wait**

In this method , the sender waits for an ACK after every frame it sends

Flow Control : Stop and Wait:-



→Only when an ACK has been received, is the next frame sent

→This process of alternately sending and waiting repeats until the sender transmits an EOT frame

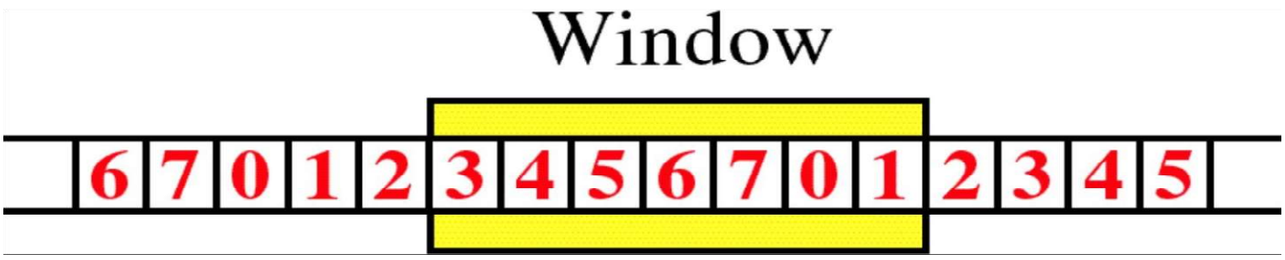
Example: Officer giving dictation to the typist, He says a word, typist says OK, he says the next word, typist says OK and so on

- **Advantages of Stop and Wait**
  →**SIMPLICITY**
  - Each frame is checked and acknowledged before the next frame is sent
- **Disadvantages of Stop and Wait**
  →**INEFFICIENT (Slow)**
  - Each frame must travel all the way to the receiver and an ACK must travel all the way back before the next frame can be sent
  - If the distance between devices is long, the time spent waiting for ACKs between each frame can be significantly long

❖ Sliding Window

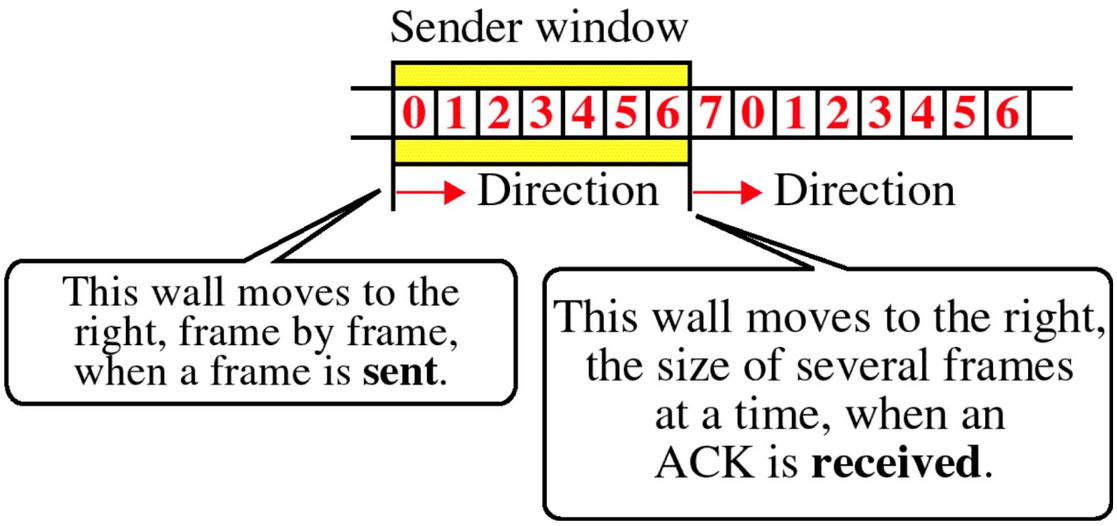
- In this method, sender can transmit several frames before needing an ACK
- Frames can be sent one right after another meaning link can carry several frames at once and its capacity can be used efficiently
- The receiver uses a single ACK to confirm the receipt of multiple data frames
- Sliding Window refers to imaginary boxes at both the sender and the receiver
- This window can hold frames at either end and provides the upper limit on the number of frames that can be sent before requiring an ACK
- Frames may be ACK at any point w/o waiting for the window to fill up and may be TX as long as the window is not yet Full
- To keep track of which frames have been transmitted and which received, sliding window introduces an identification scheme based on the size of the window
- The frames are numbered modulo-n means from 0 to n-1
- If n=8, frames are numbered 0,1,2,3,4,5,6,7,0,1,2,3,4,5,6,7,0,….

- When the receiver sends the ACK, it includes the number of the next frame it expects to receiver

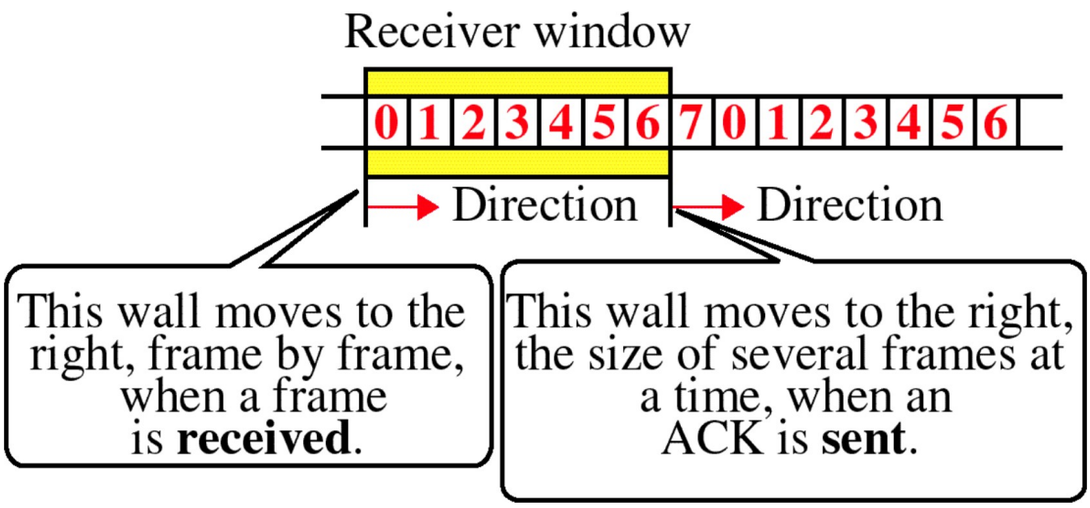


For example: to ACK the receipt of a string of frames ending in frame 4,the receiver sends an ACK with number 5

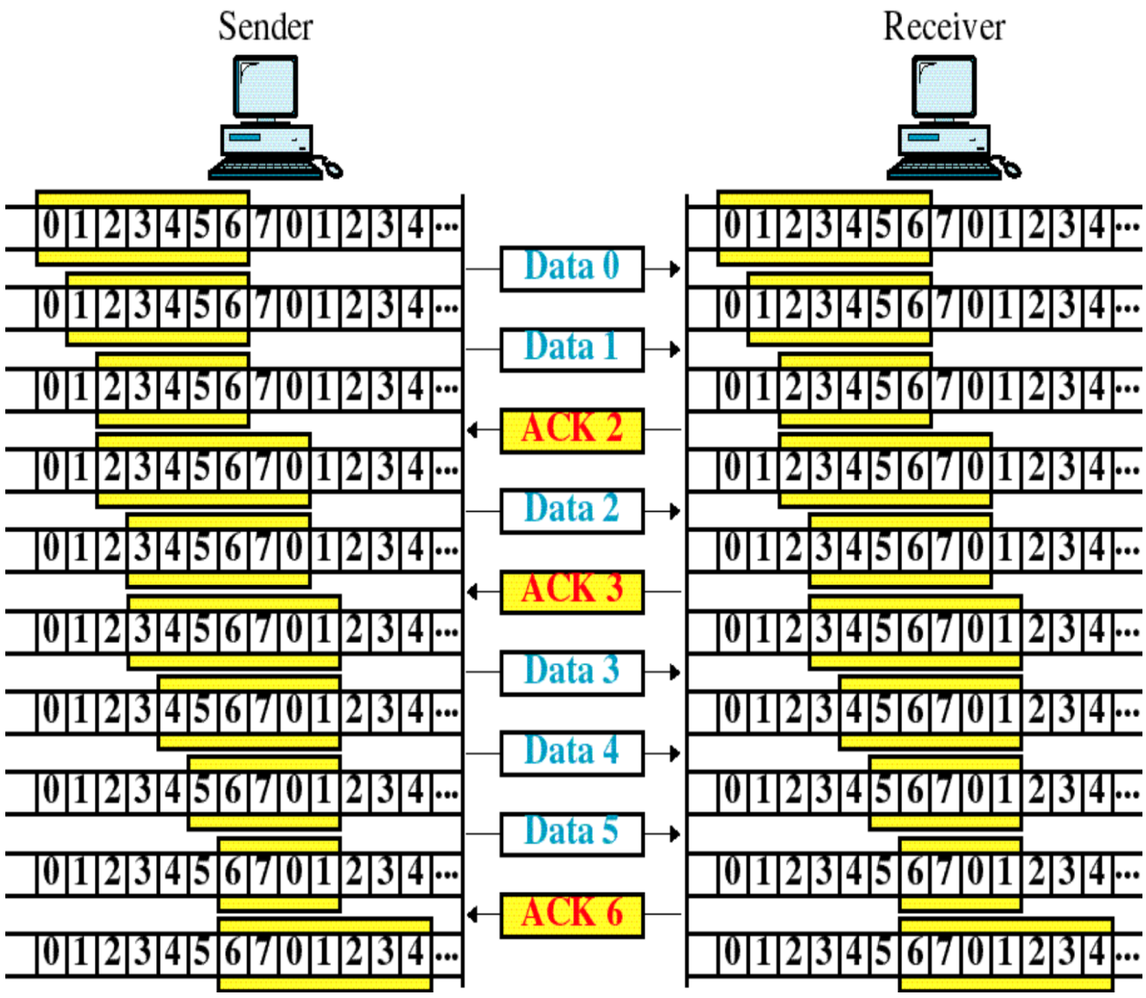
- The window can hold n-1 frames at either end, therefore a max of n-1 frames may be sent before an ACK is required.



- At the beginning of a TX, sender's window contains n-1 frames
- As frames are sent out, the left boundary of window moves inward, shrinking the size of the window
- When an ACK is received, the window expands to allow in a number of new frames equal to the number of frames acknowledged by that ACK.



- At the beginning of TX ,the receiver window contains n-1 spaces for frames
- As new frames come in the size of the receiver window shrinks
- The Reciever window therefore does not show the frames that are received but the frames that may still be received before an ACK is sent.

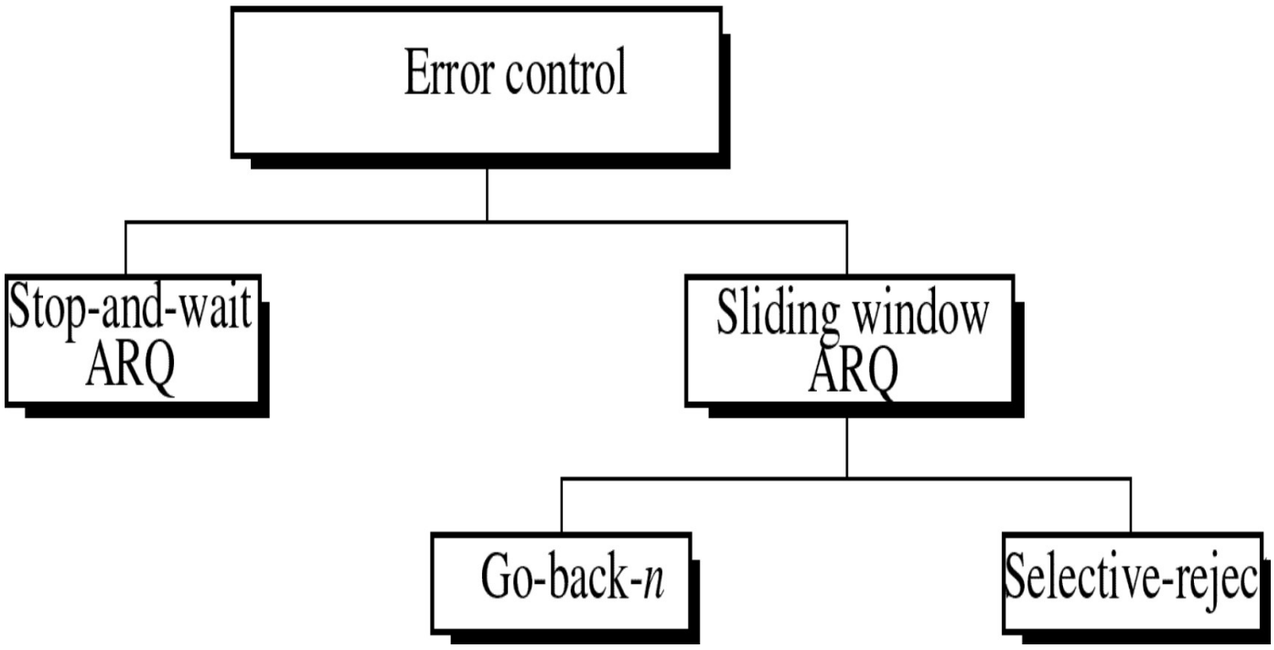


Error Control

→In the data link layer, the term error control refers primarily to methods of error detection and retransmission.

- **Automatic Repeat Request (ARQ):** Error correction in the data link layer is implemented simply:
  →Anytime an error detected in an exchange, a negative acknowledgement (NAK) is returned and the specified frames are retransmitted.
  →This process is called automatic repeat request (ARQ).
  →ARQ error control is implemented in the data link layer as an adjunct to flow control. in fact, stop-and wait flow control is usually implemented as stop-and-wait ARQ and sliding window is usually implemented as one of two variants of sliding window ARQ, called go-back-n or selective-reject.

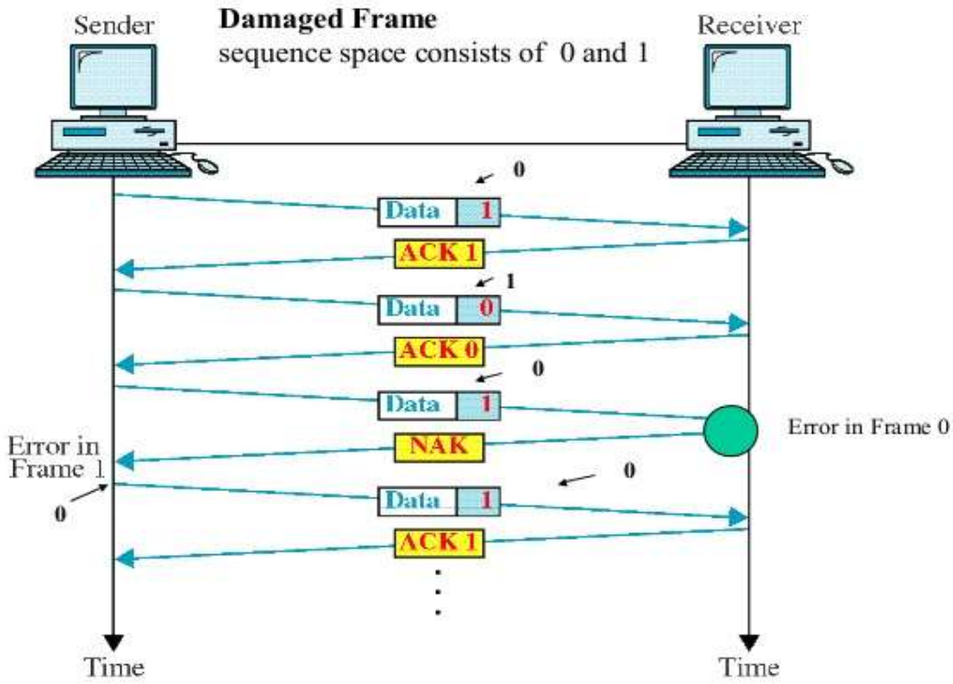
CATEGORIES OF ERROR CONTROL



Stop and Wait ARQ

- It is a form of stop-and-wait flow control extended to include retransmission of data in case of lost or Damaged frames
- For retransmission to work, four features are added to the basic flow control mechanism
- Sending device keeps a copy of the last frame transmitted until it receives the ACK for the frame
- Both data and ACK frames are numbered 0 and 1 alternately
- A data 0 frame is acknowledged by a ACK 1 frame indicating that the receiver has received data 0 and is now expecting data 1
- If an error is discovered in a data frame an NAK frame is returned NAK frames which are not sent tell the sender to retransmit the last frame
- The sending device is equipped with Timer. If an expected ACK is not received within an allotted time period, the sender assumes that the last frame sent is lost and resends the frame

2. Error Control – Stop-and-Wait ARQ



LOST FRAME:-

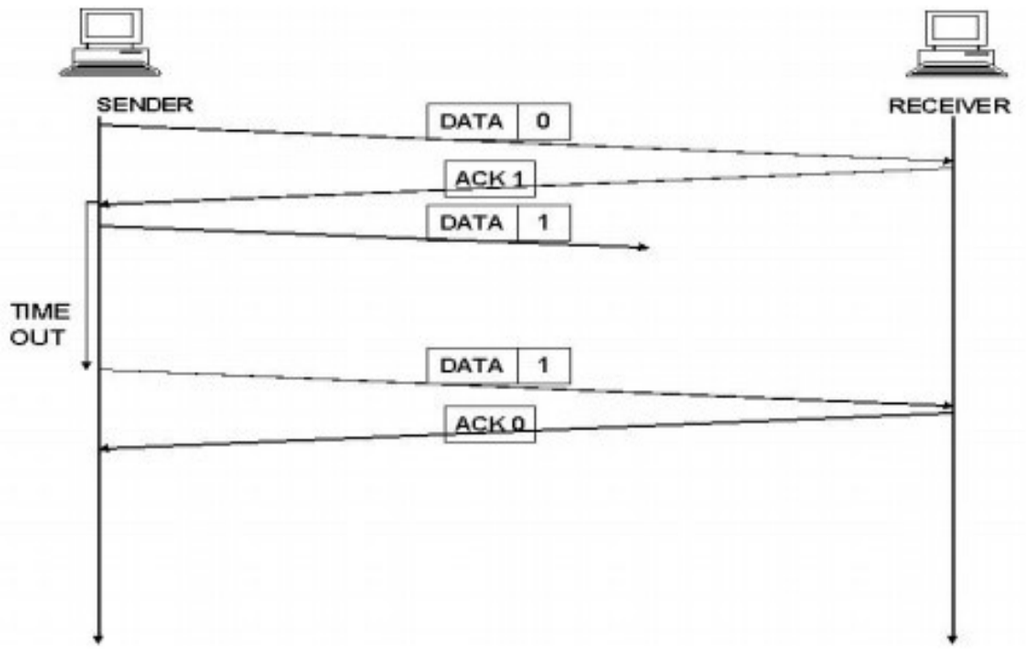
Any of 3 frame types can be lost in transfit:

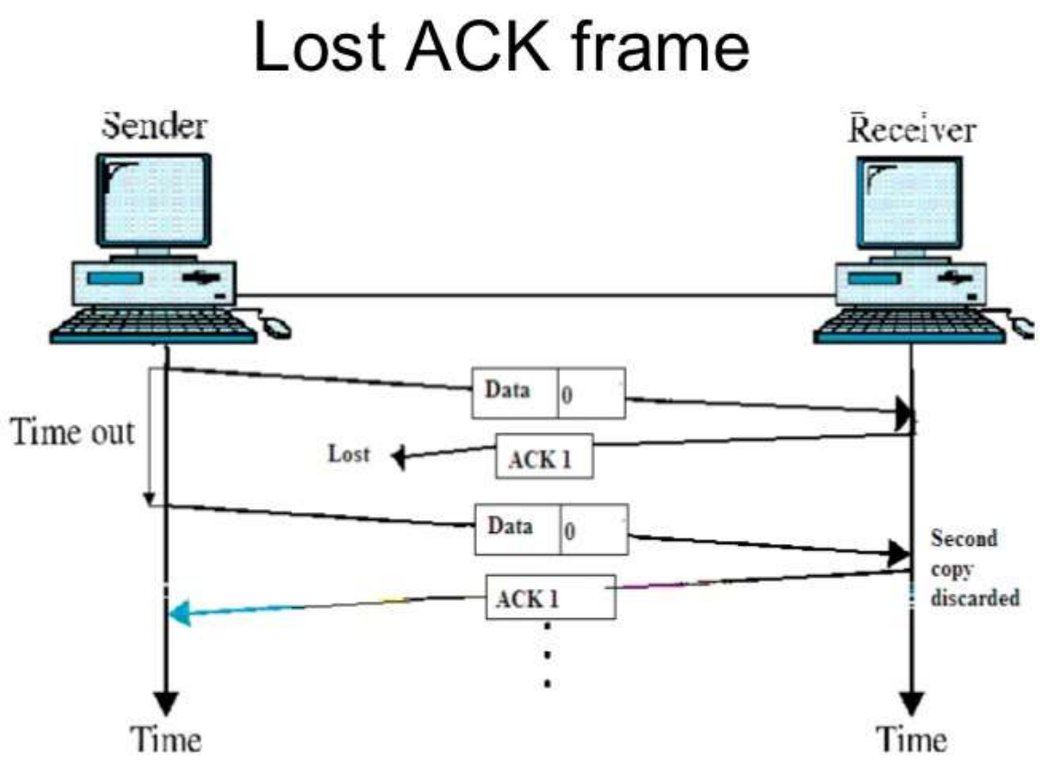
→Lost Data Frame

→Lost ACK Frame

→Lost NAK Frame

LOST NAK FRAME





SLIDING WINDOW ARQ:-

Among several popular mechanism for error control two protocols are important:

- Go-back-n ARQ
- Selective Reject ARQ

Three features are added to sliding window flow control to allow for the retransmission of the lost or the damaged frames:

→The sending device keeps copies of the transmitted frames until all of them have been acknowledged

→In addition to ACK frames, receiver also has the option of NAK frames, if data has been received damaged

→Because sliding window is a continuous TX mechanism, both ACK and NAK frames must be numbered for identification

- ACK frames carry the number of the next frame expected
- ACK 5 tells sender that all frames up to frame 5 are received
- NAK frames carry the number of the damaged frame itself
- If data frames 4 and 5 are damaged. NAK 4 and NAK 5 must be sent

→Like sender in stop-and wait ARQ, the sliding window ARQ is also equipped with a timer in the sender to deal with lost ACKs.

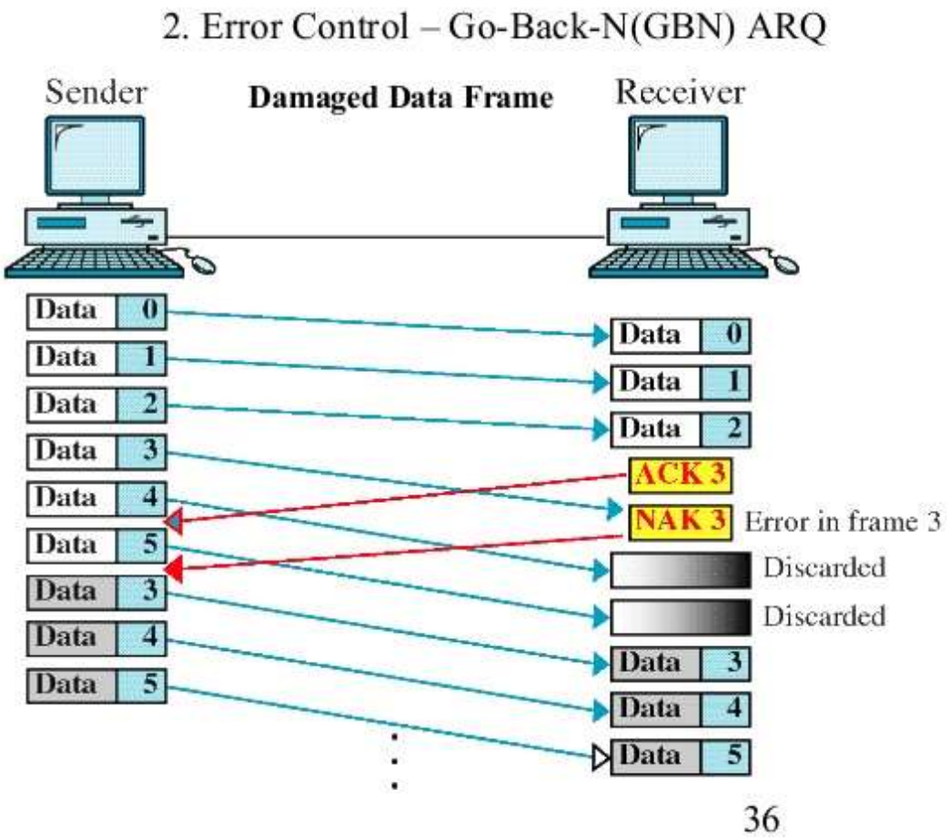
GO Back n ARQ: -

In Go Back n ARQ, if one frame is lost or damaged, all frames sent since last frame acknowledged are retransmitted.

→In this protocol we can send several frames before receiving acknowledgements; we keep a copy of these frames until the acknowledgements arrive.

→ **Sequence number filed :**

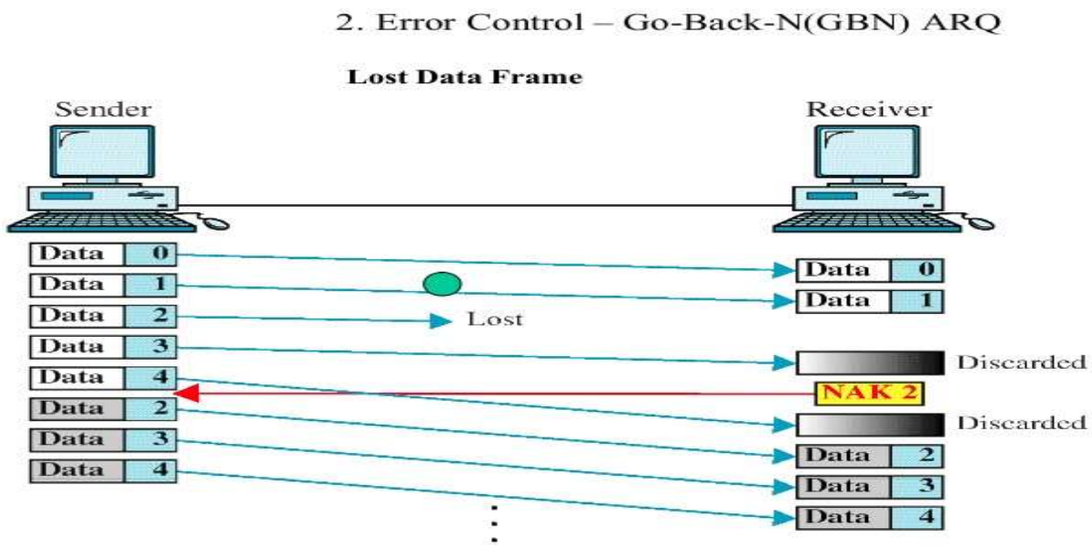
→ In Go-Back-N protocol, the sequence numbers are **modulo 2^m** , where m is the size of the sequence number field in bits. (header of the frame).



Go Back n- Lost Data Frame: -

→Sliding window requires that data frames be transmitted sequentially.

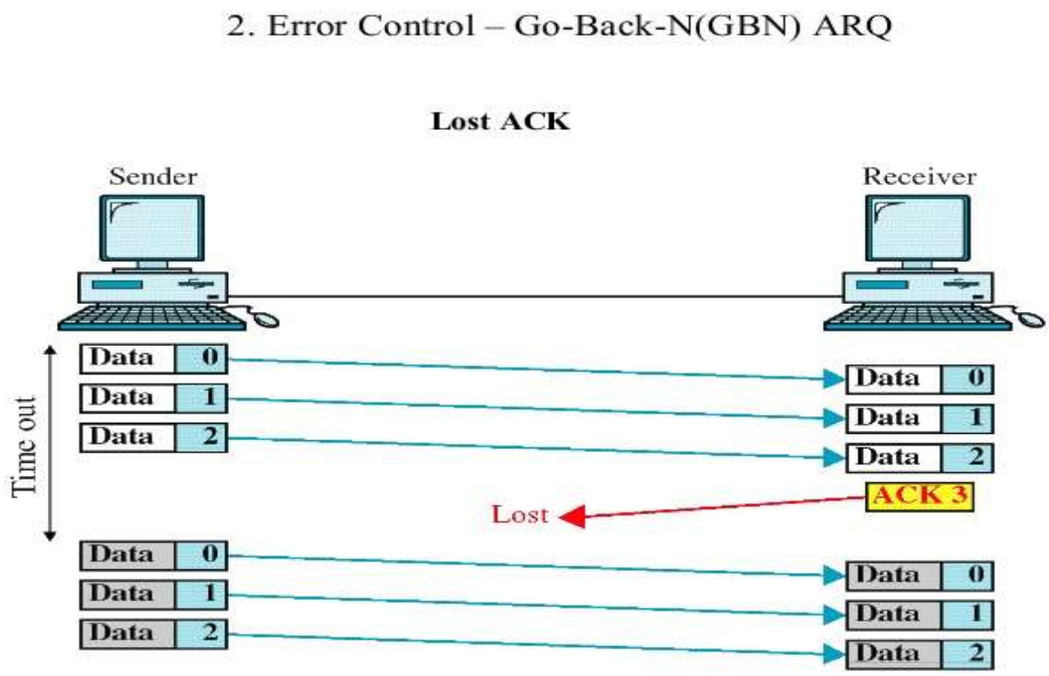
→If one or more frames are so noise corrupted that they become lost in transit, the next frame to arrive at the receiver will be out of sequence.



Go Back n- Lost ACK: -

→When the window capacity is reached and all frames allowed have been sent, the sender starts a timer

→If an ACK is not received before that Timer expires, sender retransmits all the frames since the last ACK.



Selective Reject ARQ: -

→In Selective -Reject ARQ, only the specific damaged or lost frame is retransmitted.

→If a frame is corrupted in transit, a NAK is returned and the frame is reset out of sequence.

→The Rx device must be able to sort frames and insert the retransmitted frame into the proper place

The selective reject ARQ differs from Go Back n in the following ways:

→The Rx device must contain sorting logic to enable it to reorder frames received out of sequence

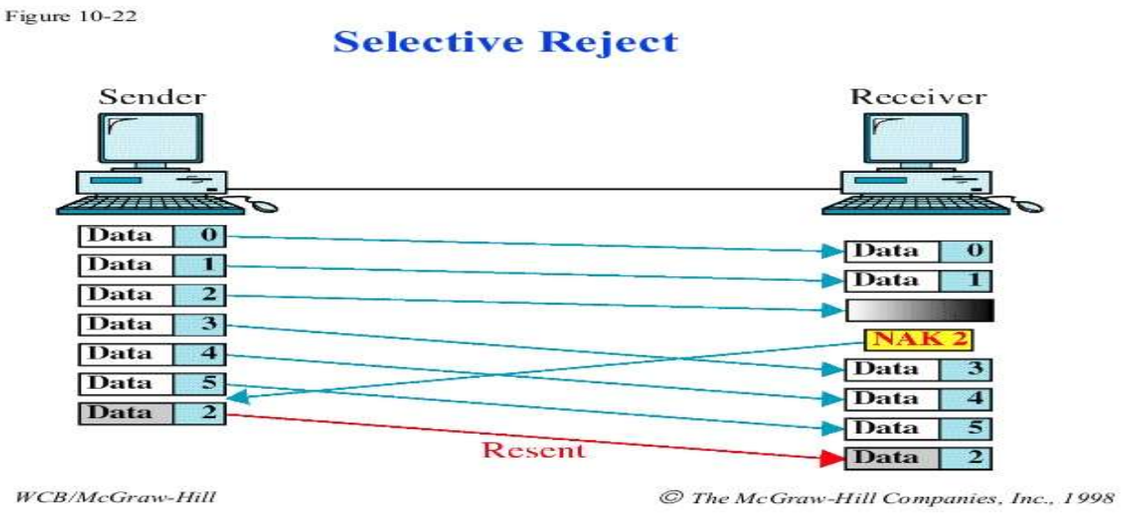
→Sending device must contain a searching mechanism that allows it to find and select only the requested frame for retransmission

→Selective Reject ARQ

→A buffer in the receiver must keep all previously received frames on hold until all retransmissions have been stored

→To avoid Selectively, ACK number, like NAK numbers must refer to frame received instead of next expected frame

→A smaller window size is required because of this added complexity



- Selective Reject ARQ-Lost Frame

Lost ACK/NAK are treated exactly in the same way as by Go Back n

Selective Reject ARQ vs Go Back n

- Although retransmitted only specific damaged or lost frames may seem more efficient than resending all the frames
- Because of the complexity of sorting and storage required by the receiver and extra logic needed by sender to select specific frames for retransmission, selective reject ARQ is EXPENSIVE and not often used
- Selective reject gives better performance but in practice it is usually it is discarded in favour of go-back-n for simplicity of implementation.

CHAPTER-2
Network Layer

The network layer is concerned with getting packets from the source all the way to the destination. Getting to the destination may require making many hops at intermediate routers along the way. This function clearly contrasts with that of the data link layer, which has the more modest goal of just moving frames from one end of a wire to the other. Thus, the network layer is the lowest layer that deals with end-to-end transmission.

Network Layer Design Issues:-

→These issues include the service provided to the transport layer and the internal design of the network.

❖ **Store-and-Forward Packet Switching: -**

→The major components of the network are the ISP's equipment (routers connected by transmission lines), shown inside the shaded oval, and the customers' equipment, shown outside the oval.

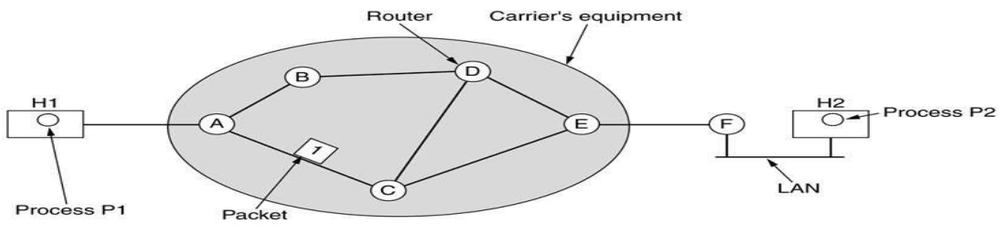
→ Host H1 is directly connected to one of the ISP's routers, A, perhaps as a home computer that is plugged into a DSL modem.

→ In contrast, H2 is on a LAN, which might be an office Ethernet, with a router, F, owned and operated by the customer.

→ This router has a leased line to the ISP's equipment. We have shown F as being outside the oval because it does not belong to the ISP.

→ For the purposes of this chapter, however, routers on customer premises are considered part of the ISP network because they run the same algorithms as the ISP's routers (and our main concern here is algorithms).

Store-and-Forward Packet Switching



The environment of the network layer protocols.

This equipment is used as follows. A host with a packet to send transmits it to the nearest router, either on its own LAN or over a point-to-point link to the ISP. The packet is stored there until it has fully arrived and the link has finished its processing by verifying the checksum. Then it is forwarded to the next router along the path until it reaches the destination host, where it is delivered. This mechanism is store-and-forward packet switching.

❖ Services Provided to the Transport Layer:-

The network layer provides services to the transport layer at the network layer/transport layer interface. An important question is precisely what kind of services the network layer provides to the transport layer. The services need to be carefully designed with the following goals in mind:

1. The services should be independent of the router technology.
2. The transport layer should be shielded from the number, type, and topology of the routers present.
3. The network addresses made available to the transport layer should use a uniform numbering plan, even across LANs and WANs.

→Given these goals, the designers of the network layer have a lot of freedom in writing detailed specifications of the services to be offered to the transport layer.

→ This freedom often degenerates into a raging battle between two warring factions. The discussion centres on whether the network layer should provide connection oriented service or connectionless service.

→One camp (represented by the Internet community) argues that the routers' job is moving packets around and nothing else.

→In this view (based on 40 years of experience with a real computer network), the network is inherently unreliable, no matter how it is designed.

→Therefore, the hosts should accept this fact and do error control (i.e., error detection and correction) and flow control themselves.

→This viewpoint leads to the conclusion that the network service should be connectionless, with primitives SEND PACKET and RECEIVE PACKET and little else.

→In particular, no packet ordering and flow control should be done, because the hosts are going to do that anyway and there is usually little to be gained by doing it twice.

→This reasoning is an example of the end-to-end argument, a design principle that has been very influential in shaping the Internet (Seltzer et al., 1984).

→ Furthermore, each packet must carry the full destination address, because each packet sent is carried independently of its predecessors, if any.

→The other camp (represented by the telephone companies) argues that the network should provide a reliable, connection-oriented service.

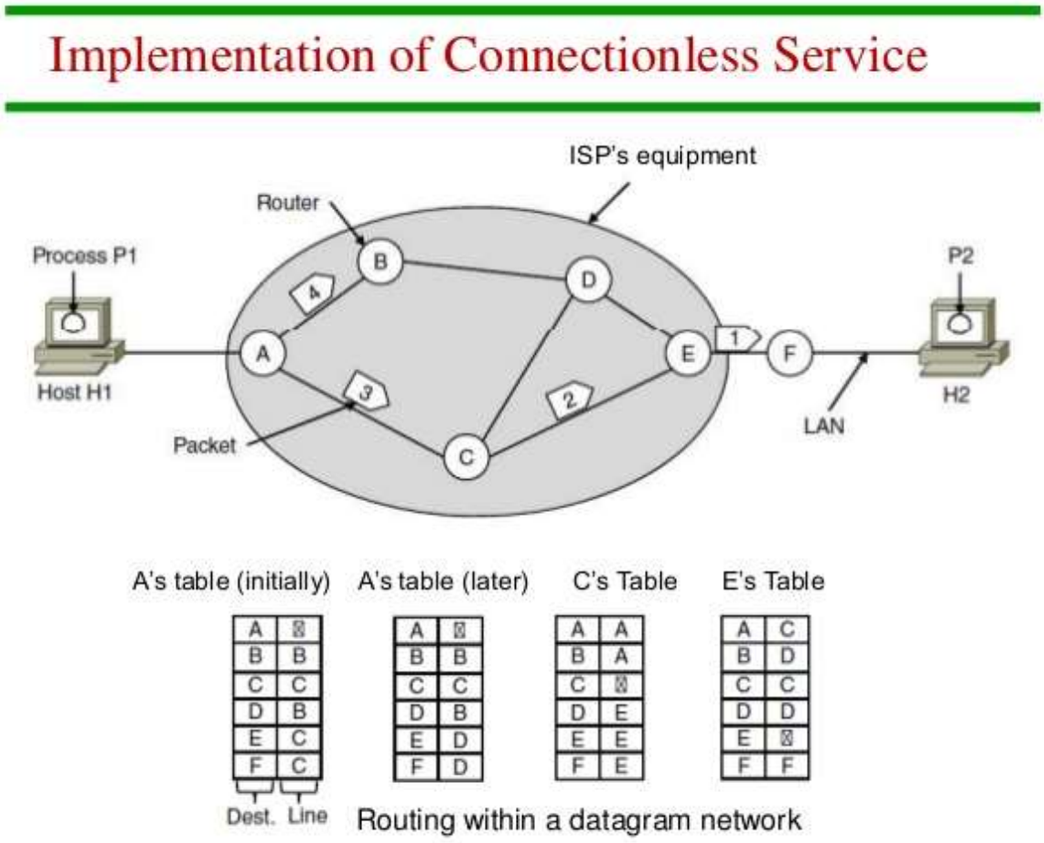
→They claim that 100 years of successful experience with the worldwide telephone system is an excellent guide.

→In this view, quality of service is the dominant factor, and without connections in the network, quality of service is very difficult to achieve, especially for real-time traffic such as voice and video.

❖ Implementation of Connectionless Service: -

- Having looked at the two classes of service the network layer can provide to its users, it is time to see how this layer works inside.
- Two different organizations are possible, depending on the type of service offered. If connectionless service is offered, packets are injected into the network individually and routed independently of each other.
- No advance setup is needed. In this context, the packets are frequently called **datagrams** (in analogy with telegrams) and the network is called a **datagram network**.
- If connection-oriented service is used, a path from the source router all the way to the destination router must be established before any data packets can be sent.
- This connection is called a **VC (virtual circuit)**, in analogy with the physical circuits set up by the telephone system, and the network is called a **virtual-circuit network**.
- In this section, we will examine datagram networks; in the next one, we will examine virtual-circuit networks.

Let us now see how a datagram network works. Suppose that the process P1 in Fig. 5-2 has a long message for P2. It hands the message to the transport layer, with instructions to deliver it to process P2 on host H2. The transport layer code runs on H1, typically within the operating system. It prepends a transport header to the front of the message and hands the result to the network layer, probably just another procedure within the operating system.



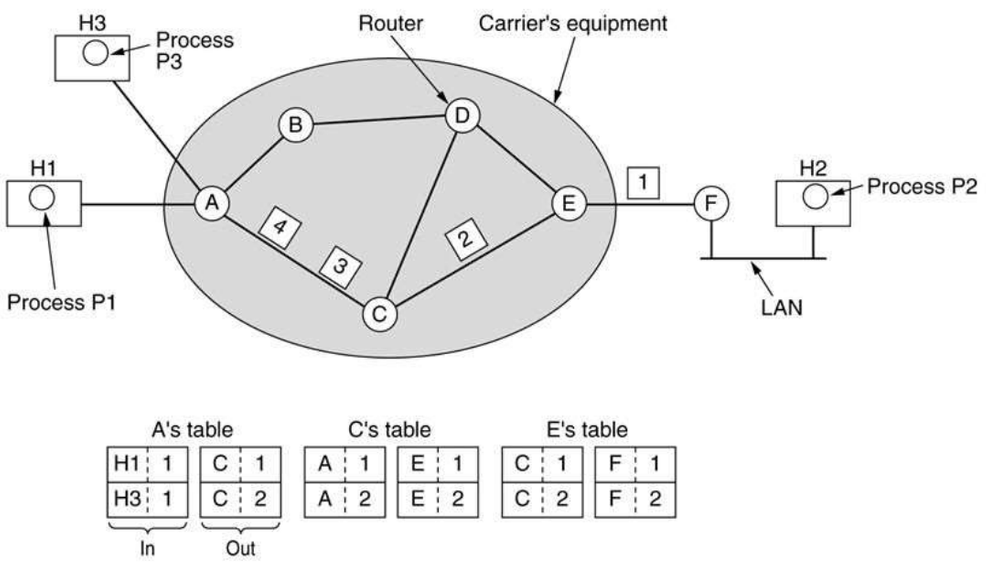
- Let us assume for this example that the message is four times longer than the maximum packet size, so the network layer has to break it into four packets, 1, 2, 3, and 4, and send each of them in turn to router A using some point-to-point protocol, for example, PPP.
- At this point the ISP takes over. Every router has an internal table telling it where to send packets for each of the possible destinations.
- Each table entry is a pair consisting of a destination and the outgoing line to use for that destination.
- Only directly connected lines can be used. For example, in Fig. 5-2, A has only two outgoing lines—to B and to C—so every incoming packet must be sent to one of these routers, even if the ultimate destination is to some other router. A's initial routing table is shown in the figure under the label ''initially.''
- At A, packets 1, 2, and 3 are stored briefly, having arrived on the incoming link and had their checksums verified.
- Then each packet is forwarded according to A's table, onto the outgoing link to C within a new frame. Packet 1 is then forwarded to E and then to F.
- When it gets to F, it is sent within a frame over the LAN to H2. Packets 2 and 3 follow the same route.
- However, something different happens to packet 4. When it gets to A it is sent to router B, even though it is also destined for F. For some reason, A decided to send packet 4 via a different route than that of the first three packets.
- Perhaps it has learned of a traffic jam somewhere along the ACE path and updated its routing table, as shown under the label ''later.'' The algorithm that manages the tables and makes the routing decisions is called the **routing algorithm**.

❖ **Implementation of Connection Oriented Service**:-

- For connection-oriented service, we need a virtual-circuit network. Let us see how that works.
- The idea behind virtual circuits is to avoid having to choose a new route for every packet sent, as in Fig. 5-2. Instead, when a connection is established, a route from the source machine to the destination machine is chosen as part of the connection setup and stored in tables inside the routers.
- That route is used for all traffic flowing over the connection, exactly the same way that the telephone system works.
- When the connection is released, the virtual circuit is also terminated. With connection-oriented service, each packet carries an identifier telling which virtual circuit it belongs to.

As an example, consider the situation shown in Fig. 5-3. Here, host H1 has established connection 1 with host H2. This connection is remembered as the first entry in each of the routing tables. The first line of A's table says that if a packet bearing connection identifier 1 comes in from H1, it is to be sent to router C and given connection identifier 1. Similarly, the first entry at C routes the packet to E, also with connection identifier 1.

Implementation of Connection-Oriented Service



Routing within a virtual-circuit subnet.

Now let us consider what happens if H3 also wants to establish a connection to H2. It chooses connection identifier 1 (because it is initiating the connection and this is its only connection) and tells the network to establish the virtual circuit. This leads to the second row in the tables. Note that we have a conflict here because although A can easily distinguish connection 1 packets from H1 from connection 1 packets from H3, C cannot do this. For this reason, A assigns a different connection identifier to the outgoing traffic for the second connection. Avoiding conflicts of this kind is why routers need the ability to replace connection identifiers in outgoing packets.

- In some contexts, this process is called **label switching**. An example of a connection-oriented network service is **MPLS (Multiprotocol Label Switching).**
- It is used within ISP networks in the Internet, with IP packets wrapped in an MPLS header having a 20-bit connection identifier or label.
- MPLS is often hidden from customers, with the ISP establishing long-term connections for large amounts of traffic, but it is increasingly being used to help when quality of service is important but also with other ISP traffic management tasks.

❖ **Comparison of Virtual-Circuit & Datagram Subnets**:-

Both virtual circuits and datagrams have their supporters and their detractors. We will now attempt to summarize both sets of arguments. The major issues are listed in table, although purists could probably find a counterexample for everything in the table.

Comparison of Virtual-Circuit and Datagram Networks

Issue	Datagram network	Virtual-circuit network
Circuit setup	Not needed	Required
Addressing	Each packet contains the full source and destination address	Each packet contains a short VC number
State information	Routers do not hold state information about connections	Each VC requires router table space per connection
Routing	Each packet is routed independently	Route chosen when VC is set up; all packets follow it
Effect of router failures	None, except for packets lost during the crash	All VCs that passed through the failed router are terminated
Quality of service	Difficult	Easy if enough resources can be allocated in advance for each VC
Congestion control	Difficult	Easy if enough resources can be allocated in advance for each VC

Inside the network, several trade-offs exist between virtual circuits and datagrams. One trade-off is setup time versus address parsing time. Using virtual circuits requires a setup phase, which takes time and consumes resources. However, once this price is paid, figuring out what to do with a data packet in a virtual-circuit network is easy: the router just uses the circuit number to index into a table to find out where the packet goes. In a datagram network, no setup is needed but a more complicated lookup procedure is required to locate the entry for the destination.

A related issue is that the destination addresses used in datagram networks are longer than circuit numbers used in virtual-circuit networks because they have a global meaning. If the packets tend to be fairly short, including a full destination address in every packet may represent a significant amount of overhead, and hence a waste of bandwidth.

CHAPTER-3
ROUTING ALGORITHMS

The main function of the network layer is routing packets from the source machine to the destination machine. In most networks, packets will require multiple hops to make the journey.

- The only notable exception is for broadcast networks, but even here routing is an issue if the source and destination are not on the same network segment.
- The algorithms that choose the routes and the data structures that they use are a major area of network layer design.
- The routing algorithm is that part of the network layer software responsible for deciding which output line an incoming packet should be transmitted on.
- If the network uses datagrams internally, this decision must be made anew for every arriving data packet since the best route may have changed since last time.
- If the network uses virtual circuits internally, routing decisions are made only when a new virtual circuit is being set up.
- Thereafter, data packets just follow the already established route. The latter case is sometimes called **session routing** because a route remains in force for an entire session (e.g., while logged in over a VPN).
- It is sometimes useful to make a distinction between routing, which is making the decision which routes to use, and forwarding, which is what happens when a packet arrives.
- One can think of a router as having two processes inside it. One of them handles each packet as it arrives, looking up the outgoing line to use for it in the routing tables. This process is forwarding.
- The other process is responsible for filling in and updating the routing tables. That is where the routing algorithm comes into play.
- Regardless of whether routes are chosen independently for each packet sent or only when new connections are established, certain properties are desirable in a routing algorithm: **correctness, simplicity, robustness, stability, fairness, and efficiency.**
- Correctness and simplicity hardly require comment, but the need for robustness may be less obvious at first.
- Once a major network comes on the air, it may be expected to run continuously for years without system-wide failures. During that period there will be hardware and software failures of all kinds.
- Hosts, routers, and lines will fail repeatedly, and the topology will change many times. The routing algorithm should be able to cope with changes in the topology and traffic without requiring all jobs in all hosts to be aborted.
- Imagine the havoc if the network needed to be rebooted every time some router crashed! Stability is also an important goal for the routing algorithm.
- There exist routing algorithms that never converge to a fixed set of paths, no matter how long they run.

- A stable algorithm reaches equilibrium and stays there. It should converge quickly too, since communication may be disrupted until the routing algorithm has reached equilibrium.
- Fairness and efficiency may sound obvious—surely no reasonable person would oppose them—but as it turns out, they are often contradictory goals.

As a simple example of this conflict, look at Fig. 5-5. Suppose that there is enough traffic between A and A′, between B and B′, and between C and C′ to saturate the horizontal links. To maximize the total flow, the X to X′ traffic should be shut off altogether. Unfortunately, X and X′ may not see it that way. Evidently, some compromise between global efficiency and fairness to individual connections is needed.

Before we can even attempt to find trade-offs between fairness and efficiency, we must decide what it is we seek to optimize. Minimizing the mean packet delay is an obvious candidate to send traffic through the network effectively, but so is maximizing total network throughput. Furthermore, these two goals are also in conflict, since operating any queueing system near capacity implies a long queueing delay.

As a compromise, many networks attempt to minimize the distance a packet must travel, or simply reduce the number of hops a packet must make. Either choice tends to improve the delay and also reduce the amount of bandwidth consumed per packet, which tends to improve the overall network throughput as well. Routing algorithms can be grouped into two major classes: nonadaptive and adaptive. Nonadaptive algorithms do not base their routing decisions on any measurements or estimates of the current topology and traffic.

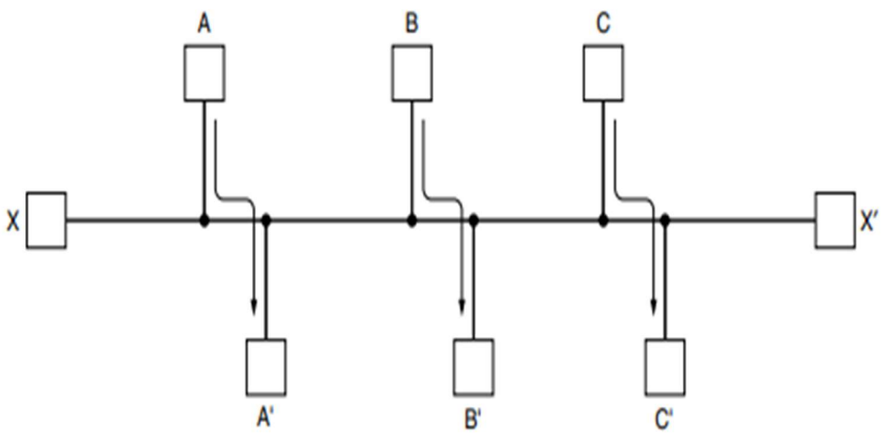


Figure 5-5. Network with a conflict between fairness and efficiency.

Instead, the choice of the route to use to get from I to J (for all I and J) is computed in advance, offline, and downloaded to the routers when the network is booted. This procedure is sometimes called **static routing**. Because it does not respond to failures, static routing is mostly useful for situations in which the routing choice is clear.

Adaptive algorithms, in contrast, change their routing decisions to reflect changes in the topology, and sometimes changes in the traffic as well. These **dynamic routing algorithms** differ in where they get their information (e.g., locally, from adjacent routers, or from all routers), when they change the routes (e.g., when the topology changes, or every ΔT seconds as the load changes), and what metric is used for optimization (e.g., distance, number of hops, or estimated transit time).

→The network layer is concerned with getting packets from the source all the way to the destination.

→The packets may require to make many hops at the intermediate routers while reaching the destination.

→This is the lowest layer that deals with end to end transmission.

→In order to achieve its goals, the network later must know about the topology of the communication network.

→It must also take care to choose routes to avoid overloading of some of the communication lines while leaving others idle.

The main functions performed by the network layer are as follows:

Routing:

Routing is the process of forwarding of a packet in a network so that it reaches its intended destination.

The main goals of routing are:

Correctness: The routing should be done properly and correctly so that the packets may reach their proper destination.

Simplicity: The routing should be done in a simple manner so that the overhead is as low as possible.

With increasing complexity of the routing algorithms the overhead also increases.

Robustness: Once a major network becomes operative, it may be expected to run continuously for years without any failures.

The algorithms designed for routing should be robust enough to handle hardware and software failures and should be able to cope with changes in the topology and traffic without requiring all jobs in all hosts to be aborted and the network rebooted every time some router goes down.

Stability: The routing algorithms should be stable under all possible circumstances.

Fairness: Every node connected to the network should get a fair chance of transmitting their packets.

This is generally done on a first come first serve basis.

Optimality: The routing algorithms should be optimal in terms of throughput and minimizing mean packet delays.

Here there is a trade-off and one has to choose depending on his suitability.

Throughput:

Throughput refers to how much data can be transferred from source to destination within a given timeframe. Throughput measures how many packets arrive at their destinations successfully.
For the most part, throughput capacity is measured in bits per second, but it can also be measured in data per second.

❖ The Optimality Principle:-

- Before we get into specific algorithms, it may be helpful to note that one can make a general statement about optimal routes without regard to network topology or traffic.
- This statement is known as the optimality principle (Bellman, 1957). It states that if router J is on the optimal path from router I to router K, then the optimal path from J to K also falls along the same route.
- To see this, call the part of the route from I to J r 1 and the rest of the route r 2. If a route better than r 2 existed from J to K, it could be concatenated with r 1 to improve the route from I to K, contradicting our statement that r 1r 2 is optimal.
- As a direct consequence of the optimality principle, we can see that the set of optimal routes from all sources to a given destination form a tree rooted at the destination.
- Such a tree is called a sink tree and is illustrated in Figure where the distance metric is the number of hops. The goal of all routing algorithms is to discover and use the sink trees for all routers.

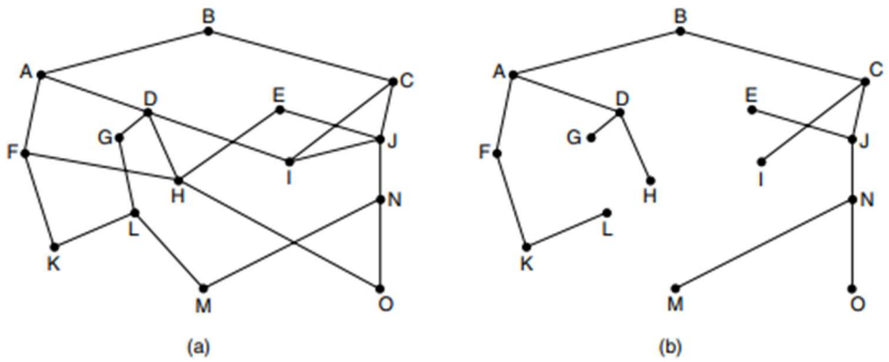


Figure 5-6. (a) A network. (b) A sink tree for router *B*.

Note that a sink tree is not necessarily unique; other trees with the same path lengths may exist. If we allow all of the possible paths to be chosen, the tree becomes a more general structure called a **DAG (Directed Acyclic Graph).** DAGs have no loops. We will use sink trees as a convenient shorthand for both cases. Both cases also depend on the technical assumption that the paths do not interfere with each other so, for example, a traffic jam on one path will not cause another path to divert.

❖ Shortest Path Routing: -

- These paths are the ones that we want a distributed routing algorithm to find, even though not all routers may know all of the details of the network.
- The idea is to build a graph of the network, with each node of the graph representing a router and each edge of the graph representing a communication line, or link.
- To choose a route between a given pair of routers, the algorithm just finds the shortest path between them on the graph.
- The concept of a **shortest path** deserves some explanation. One way of measuring path length is the number of hops.
- Using this metric, the paths ABC and ABE in Fig. 5-7 are equally long. Another metric is the geographic distance in kilometres, in which case ABC is clearly much longer than ABE (assuming the figure is drawn to scale).

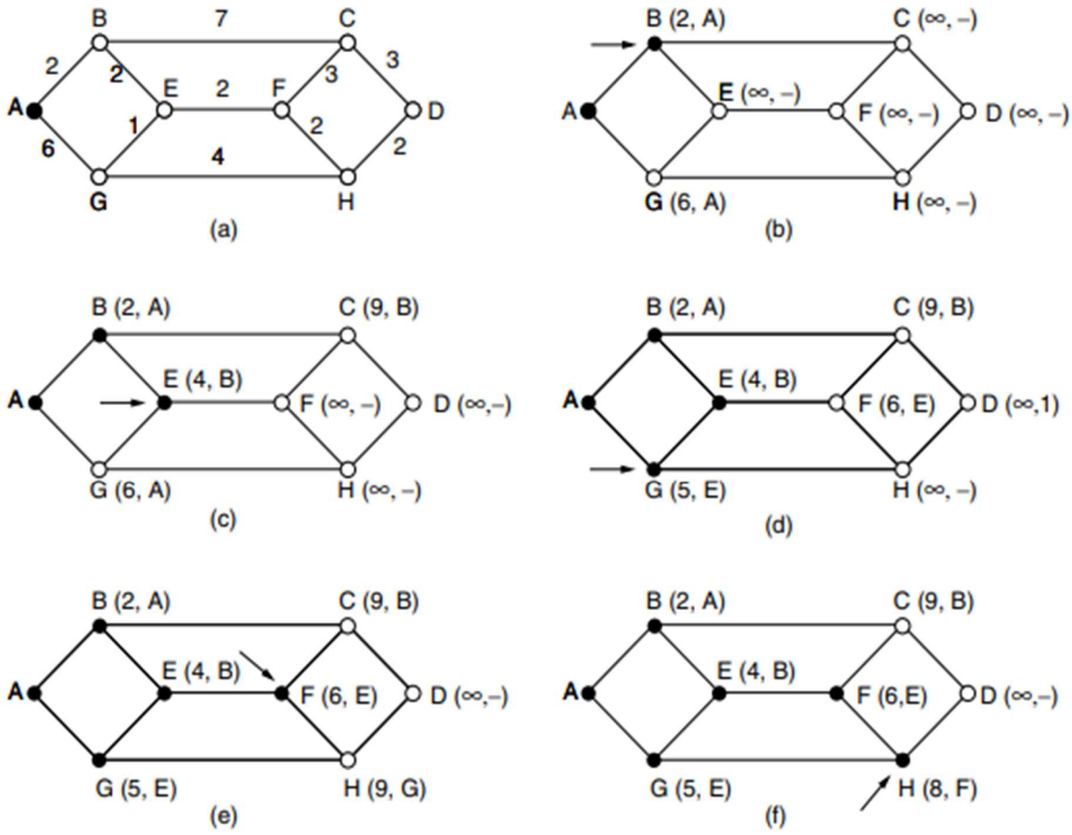


Figure 5-7. The first six steps used in computing the shortest path from *A* to *D*. The arrows indicate the working node.

- However, many other metrics besides hops and physical distance are also possible. For example, each edge could be labelled with the mean delay of a standard test packet, as measured by hourly runs.
- With this graph labelling, the shortest path is the fastest path rather than the path with the fewest edges or kilometres.
- In the general case, the labels on the edges could be computed as a function of the distance, bandwidth, average traffic, communication cost, measured delay, and other factors.
- By changing the weighting function, the algorithm would then compute the ‘‘shortest’’ path measured according to any one of a number of criteria or to a combination of criteria.
- Several algorithms for computing the shortest path between two nodes of a graph are known.
- This one is due to Dijkstra (1959) and finds the shortest paths between a source and all destinations in the network.
- Each node is labelled (in parentheses) with its distance from the source node along the best-known path.
- The distances must be non-negative, as they will be if they are based on real quantities like bandwidth and delay.

- Initially, no paths are known, so all nodes are labelled with infinity. As the algorithm proceeds and paths are found, the labels may change, reflecting better paths.
- A label may be either tentative or permanent. Initially, all labels are tentative.
- When it is discovered that a label represents the shortest possible path from the source to that node, it is made permanent and never changed thereafter.

❖ Flooding: -

- When a routing algorithm is implemented, each router must make decisions based on local knowledge, not the complete picture of the network.
- A simple local technique is **flooding**, in which every incoming packet is sent out on every outgoing line except the one it arrived on.
- Flooding obviously generates vast numbers of duplicate packets, in fact, an infinite number unless some measures are taken to damp the process.
- One such measure is to have a hop counter contained in the header of each packet that is decremented at each hop, with the packet being discarded when the counter reaches zero.
- Ideally, the hop counter should be initialized to the length of the path from source to destination.
- If the sender does not know how long the path is, it can initialize the counter to the worst case, namely, the full diameter of the network.
- Flooding with a hop count can produce an exponential number of duplicate packets as the hop count grows and routers duplicate packets they have seen before.
- A better technique for damming the flood is to have routers keep track of which packets have been flooded, to avoid sending them out a second time.
- One way to achieve this goal is to have the source router put a sequence number in each packet it receives from its hosts.
- Each router then needs a list per source router telling which sequence numbers originating at that source have already been seen.
- If an incoming packet is on the list, it is not flooded.

❖ *DISTANCE VECTOR ROUTING:-*

- Computer networks generally use dynamic routing algorithms that are more complex than flooding, but more efficient because they find shortest paths for the current topology.
- Two dynamic algorithms in particular, distance vector routing and link state routing, are the most popular. In this section, we will look at the former algorithm. In the following section, we will study the latter algorithm.
- A **distance vector routing algorithm** operates by having each router maintain a table (i.e., a vector) giving the best-known distance to each destination and which link to use to get there.
- These tables are updated by exchanging information with the neighbors. Eventually, every router knows the best link to reach each destination.

- The distance vector routing algorithm is sometimes called by other names, most commonly the distributed **Bellman-Ford routing algorithm**, after the researchers who developed it (Bellman, 1957; and Ford and Fulkerson, 1962).
- It was the original ARPANET routing algorithm and was also used in the Internet under the name RIP. In distance vector routing, each router maintains a routing table indexed by, and containing one entry for each router in the network.
- This entry has two parts: the preferred outgoing line to use for that destination and an estimate of the distance to that destination. The distance might be measured as the number of hops or using another metric, as we discussed for computing shortest paths.
- The router is assumed to know the ‘‘distance’’ to each of its neighbors. If the metric is hops, the distance is just one hop. If the metric is propagation delay, the router can measure it directly with special ECHO packets that the receiver just timestamps and sends back as fast as it can.
- As an example, assume that delay is used as a metric and that the router knows the delay to each of its neighbors.
- Once every T msec, each router sends to each neighbor a list of its estimated delays to each destination. It also receives a similar list from each neighbor.
- Imagine that one of these tables has just come in from neighbor X, with Xi being X’s estimate of how long it takes to get to router i.
- If the router knows that the delay to X is m msec, it also knows that it can reach router i via X in Xi + m msec.
- By performing this calculation for each neighbor, a router can find out which estimate seems the best and use that estimate and the corresponding link in its new routing table.
- Note that the old routing table is not used in the calculation. This updating process is illustrated in Fig. 5-9. Part (a) shows a network.
- The first four columns of part (b) show the delay vectors received from the neighbors of router J. A claim to have a 12-msec delay to B, a 25-msec delay to C, a 40- msec delay to D, etc. Suppose that J has measured or estimated its delay to its neighbors, A, I, H, and K, as 8, 10, 12, and 6 msec, respectively.

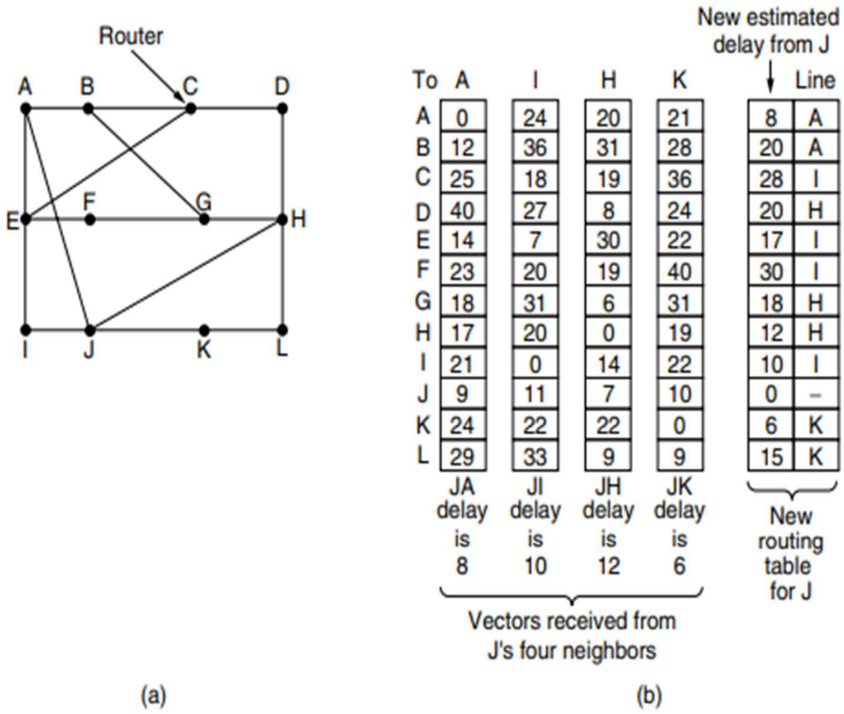


Figure 5-9. (a) A network. (b) Input from *A*, *I*, *H*, *K*, and the new routing table for *J*.

- ❖ Consider how J computes its new route to router G. It knows that it can get to A in 8 msec, and furthermore A claims to be able to get to G in 18 msec, so J knows it can count on a delay of 26 msec to G if it forwards packets bound for G to A.
- ❖ Similarly, it computes the delay to G via I, H, and K as 41 (31 + 10), 18 (6 + 12), and 37 (31 + 6) msec, respectively. The best of these values is 18, so it makes an entry in its routing table that the delay to G is 18 msec and that the route to use is via H. The same calculation is performed for all the other destinations, with the new routing table shown in the last column of the figure.

❖ Link State Routing:-

- Distance vector routing was used in the ARPANET until 1979, when it was replaced by link state routing.
- The primary problem that caused its demise was that the algorithm often took too long to converge after the network topology changed (due to the count-to-infinity problem).
- Consequently, it was replaced by an entirely new algorithm, now called link state routing.

- Variants of link state routing called IS-IS and OSPF are the routing algorithms that are most widely used inside large networks and the Internet today.
- The idea behind link state routing is fairly simple and can be stated as five parts. Each router must do the following things to make it work:
  1.Discover its neighbours and learn their network addresses.
  2. Set the distance or cost metric to each of its neighbours.
  3. Construct a packet telling all it has just learned.
  4. Send this packet to and receive packets from all other routers.
  5. Compute the shortest path to every other router.
- In effect, the complete topology is distributed to every router.
- Then Dijkstra's algorithm can be run at each router to find the shortest path to every other router. Below we will consider each of these five steps in more detail.

Learning about the Neighbours: -When a router is booted, its first task is to learn who its neighbors are. It accomplishes this goal by sending a special HELLO packet on each point-to-point line. The router on the other end is expected to send back a reply giving its name. These names must be globally unique because when a distant router later hears that three routers are all connected to F, it is essential that it can determine whether all three mean the same F.

When two or more routers are connected by a broadcast link (e.g., a switch, ring, or classic Ethernet), the situation is slightly more complicated. Fig. 5-11(a) illustrates a broadcast LAN to which three routers, A, C, and F, are directly connected. Each of these routers is connected to one or more additional routers, as shown.

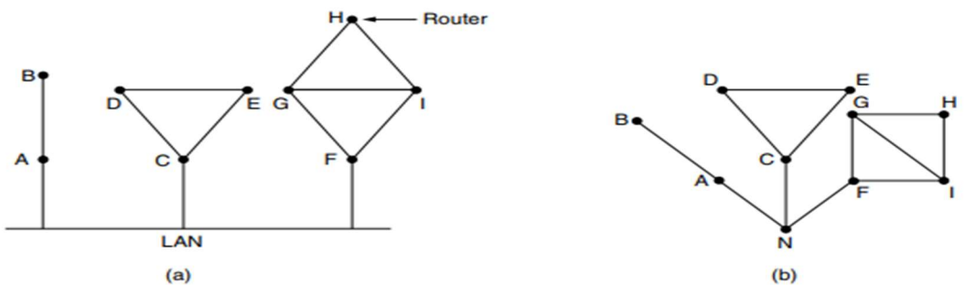


Figure 5-11. (a) Nine routers and a broadcast LAN. (b) A graph model of (a).

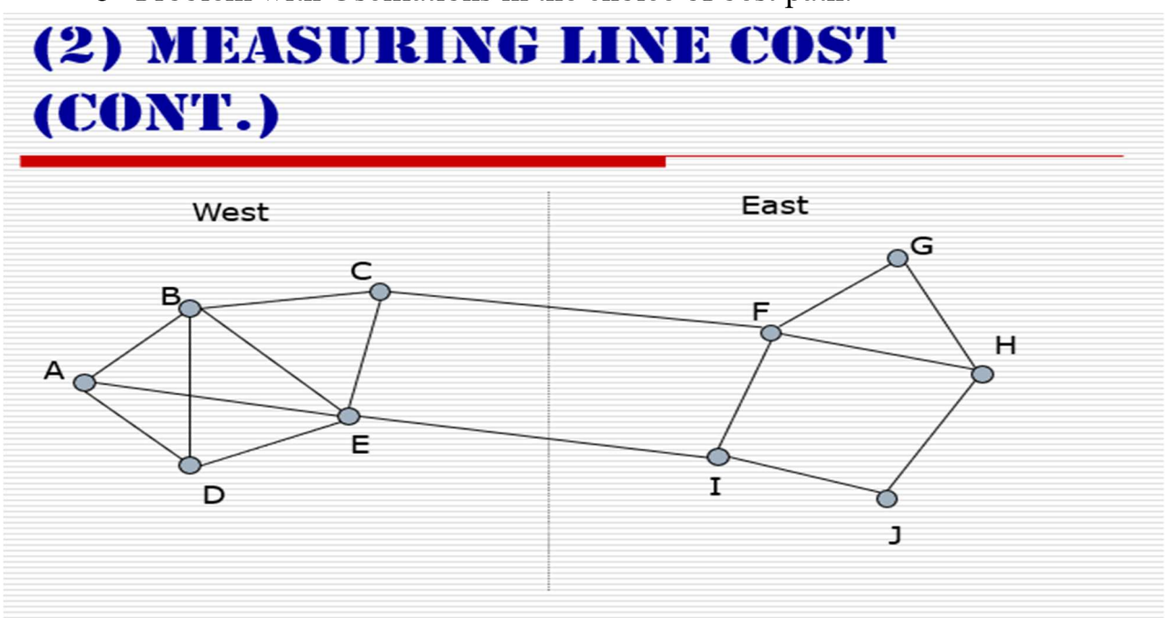
The broadcast LAN provides connectivity between each pair of attached routers. However, modelling the LAN as many point-to-point links increases the size of the topology and leads to wasteful messages. A better way to model the LAN is to consider it as a node itself, as shown in Fig. 5-11(b). Here, we have introduced a new, artificial node, N, to which A, C, and F are connected. One designated router on the LAN is selected to play the role of N in the routing protocol. The fact that it is possible to go from A to C on the LAN is represented by the path ANC here.

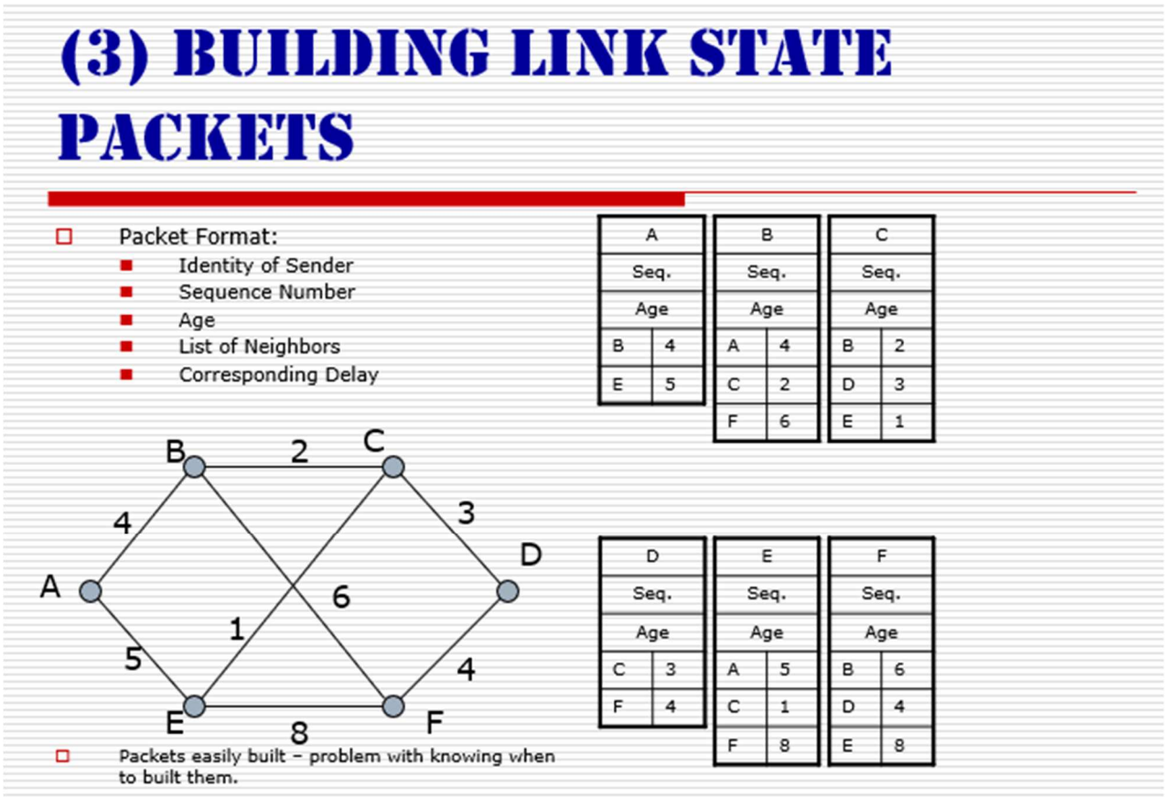
➢ **Measuring Line Cost:-**

❖ It is required by the Link State Routing algorithm that each router not have a reasonable estimate of the delay/cost to each of its neighbors.
  - Send "ECHO" packet (ping) that the other side is required to send back immediately.
    - Measure Round Trip time; Divide by 2 to get an estimate.
    - More accurate estimate by repeating the process several times and by averaging estimates.
    - Assumes symmetric delay.

❖ Channel Load Issue when Measuring Delay
  - To factor the load in: round trip timer must be started when the ECHO packed is queued.
  - To ignore the load: round trip timer must be started when ECHO packed reaches front of the queue.

❖ Including Traffic-induced Delays:
  - If a router has a choice from 2 lines with the same bandwidth, one of which is heavily loaded all the time and one of which is not, the router will regard the route over the unloaded line as shorter path. This choice in general will result in better performance
  - Problem with Oscillations in the choice of best path.





→Distributing the Link State Packets:-

- Distributing Link State Packets ***Reliably*** is tricky:
  - As the packets are distributed and installed, the routers getting the first ones will change their routes before other routers in the network update their routing tables.
  - Different Routers may be using different versions of the topology (inconsistencies, loops, unreachable machines, etc.)
- Basic Algorithm**:** ***Flooding***
  - Sequence Number (incremented for each new packet sent) is used to keep the flood in check.
  - Routers keep track of all the source router packets they have been sent to.
  - New link state packets is checked against the track list:
    - If new/unseen (based on the sequence number) then it is broadcasted to all neighboring routers with exception of the sender.
    - If duplicate, it is disregarded.
    - If sequence number is lower than the highest one in the track list, it is rejected.

- Problems with basic algorithm:
    1. Sequence Number wrap around.
        - Make a long precision number (e.g., 32-bit)
    2. Crash of a router: losing track of sequence number.
    3. Corruption of sequence number.
- Solution: Include Age of each packet.
    - Decrement this value once per second.
    - When zero, this state information is disregarded.
    - Normally a new packed is send every 10 sec.
        - Router information times out when:
            - Router is down, or

A Number of (e.g., 6) consecutive packets have been lost

Refinements of Distribution Algorithm:

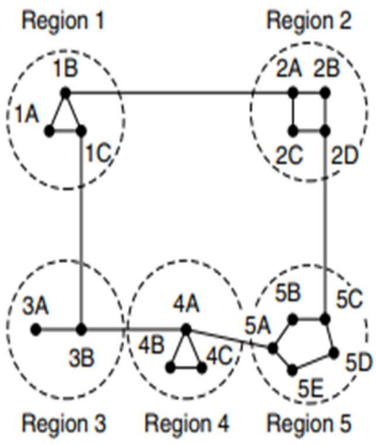
- Holding the packet:
- Example of packed buffer for router B of subnet in previous figure (Fig. 5-13 (a))

			Sent Flags			Acknowledged Flags			
Source	Seq.	Age	A	C	F	A	C	F	Data
A	21	60	0	1	1	1	0	0	
F	21	60	1	1	0	0	0	1	
E	21	59	0	1	0	1	0	1	
C	20	60	1	0	1	0	1	0	
D	21	59	1	0	0	0	1	1	

❖ **Hierarchical Routing**: -

- As networks grow in size, the router routing tables grow proportionally. Not only is router memory consumed by ever-increasing tables, but more CPU time is needed to scan them and more bandwidth is needed to send status reports about them.
- At a certain point, the network may grow to the point where it is no longer feasible for every router to have an entry for every other router, so the routing will have to be done hierarchically, as it is in the telephone network.
- When hierarchical routing is used, the routers are divided into what we will call regions.
- Each router knows all the details about how to route packets to destinations within its own region but knows nothing about the internal structure of other regions.
- When different networks are interconnected, it is natural to regard each one as a separate region to free the routers in one network from having to know the topological structure of the other ones.
- For huge networks, a two-level hierarchy may be insufficient; it may be necessary to group the regions into clusters, the clusters into zones, the zones into groups, and so on, until we run out of names for aggregations.
- As an example of a multilevel hierarchy, consider how a packet might be routed from Berkeley, California, to Malindi, Kenya.
- The Berkeley router would know the detailed topology within California but would send all out-of-state traffic to the Los Angeles router.
- The Los Angeles router would be able to route traffic directly to other domestic routers but would send all foreign traffic to New York.
- The New York router would be programmed to direct all traffic to the router in the destination country responsible for handling foreign traffic, say, in Nairobi. Finally, the packet would work its way down the tree in Kenya until it got to Malindi. Figure 5-14 gives a quantitative example of routing in a two-level hierarchy with five regions.
- The full routing table for router 1A has 17 entries, as shown in Fig. 5-14(b). When routing is done hierarchically, as in Fig. 5-14(c), there are entries for all the local routers, as before, but all other regions are condensed into a single router, so all traffic for region 2 goes via the 1B-2A line, but the rest of the remote traffic goes via the 1C-3B line.
- Hierarchical routing has reduced the table from 17 to 7 entries. As the ratio of the number of regions to the number of routers per region grows, the savings in table space increase. Unfortunately, these gains in space are not free.
- There is a penalty to be paid: increased path length. For example, the best route from 1A to 5C is via region 2, but with hierarchical routing all traffic to region 5 goes via region 3, because that is better for most destinations in region 5.
- When a single network becomes very large, an interesting question is ‘‘how many levels should the hierarchy have?’’

For example, consider a network with 720 routers. If there is no hierarchy, each router needs 720 routing table entries. If the network is partitioned into 24 regions of 30 routers each, each router needs 30 local entries plus 23 remote entries for a total of 53 entries. If a three-level hierarchy is chosen, with 8 clusters each containing 9 regions of 10 routers, each router needs 10 entries for local routers, 8 entries for routing to other regions within its own cluster, and 7 entries for distant clusters, for a total of 25 entries. Kamoun and Kleinrock (1979) discovered that the optimal number of levels for an N router network is ln N, requiring a total of e ln N entries per router. They have also shown that the increase in effective mean path length caused by hierarchical routing is sufficiently small that it is usually acceptable.



(a)

Full table for 1A

Dest.	Line	Hops
1A	–	–
1B	1B	1
1C	1C	1
2A	1B	2
2B	1B	3
2C	1B	3
2D	1B	4
3A	1C	3
3B	1C	2
4A	1C	3
4B	1C	4
4C	1C	4
5A	1C	4
5B	1C	5
5C	1B	5
5D	1C	6
5E	1C	5

(b)

Hierarchical table for 1A

Dest.	Line	Hops
1A	–	–
1B	1B	1
1C	1C	1
2	1B	2
3	1C	2
4	1C	3
5	1C	4

(c)

Figure 5-14. Hierarchical routing.

Chapter 4 in UNIT II

Congestion Control Algorithms

1.1 What is congestion?

In general, **congestion** is the state of network that the **load on a subnet** (or in part of it i.e. some routers) is greater than the capacity of resources on a subnet can handle. This **decreases the performance** of the network. And it may also result in packet loss.

(Or)

When too many packets are sent to a subnet more than its capacity, the Situation that arises is called **congestion**.

The following diagram explains about **subnet performance during congestion**

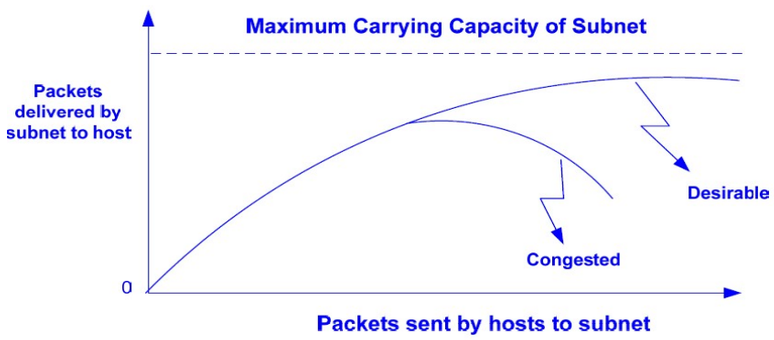
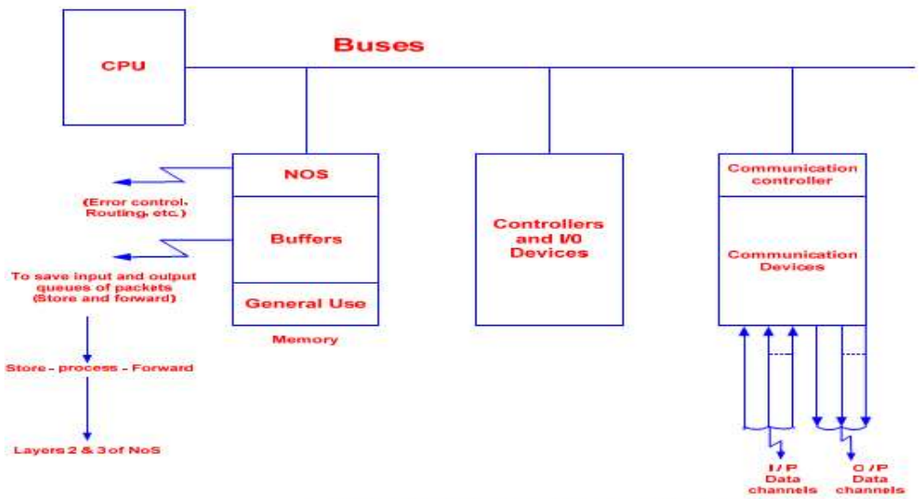


Fig Subnet performance During Congestion

- In any subnet system the maximum utilization of resources will be up to 80%.
- It can deliver the packets perfectly up to the capacity of subnet, after that if the traffic increases in subnet it can handle for a **desirable** period.
- And finally at very high traffic, performance collapses completely and almost no packets are delivered this situation was called congestion.

Why congestion occurs?

There are several reasons for congestion some of them are as follows; Lets define the congestion on router level first. The **router** is generally a computer connected to input and output data channels.



Simple Block Diagram of Router

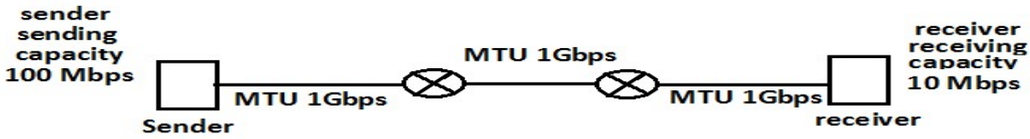
- **Insufficient router buffer:** if the number of input packets to the router is greater than the number of output packets then buffered packets will grow rapidly which eventually causes saturation of buffers and later loss of newly received packets.

  Increase the buffer size get congestion worse i.e. if the buffer size increases the time take to gets packet front of queue is also increase mean while the packet life time may reach zero.

- **Slow performance:** If the routers' CPUs are slow at performing the tasks queuing buffers, updating tables and etc… may cause congestion.
- **Low bandwidth:** low-bandwidth lines can also cause congestion.

Flow control Vs **congestion control:**

- **Flow control:** Flow control is a mechanism used to control the flow of data between a sender and a receiver, such that a slow receiver will not be outran by a fast sender.



- **Congestion control:** Congestion control is a mechanism that controls data flow when congestion actually occurs. It controls data entering in to a network such that the network can handle the traffic within the network.
- **Example:** consider a network with 1-Mbps lines and 1000 large computers, half of which are trying to transfer files at 100 kbps to the other half. Here the problem is the total traffic exceeds what the network can handle.
- **Difference between flow and congestion control:**
    - Flow control is an **end to end** mechanism that controls the traffic between a sender and a receiver.
    - Congestion control is a mechanism that is used by a network to control congestion in the network **globally**.
    - Congestion control is a mechanism that makes sure that an entire network can handle the traffic that is coming to the network.
    - But, flow control refers to mechanisms used to handle the transmission between a particular sender and a receiver.

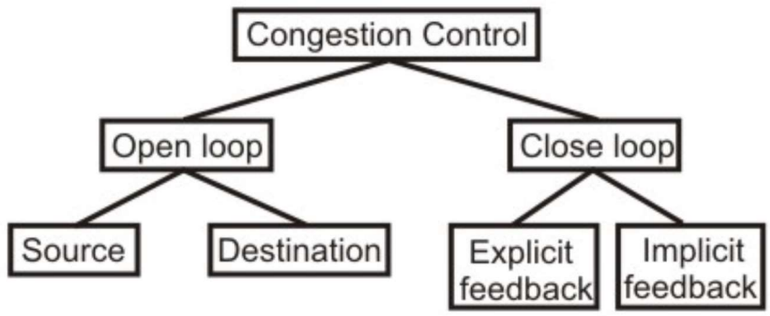
1.2 General Principles of Congestion Control:

We already know that the performance of the subnet collapse when the congestion occurs. The solution is control the congestion.

Solutions to congestion problems i.e. the **congestion control algorithm** can divide into two groups

- Open loop
- Closed loop

Diagram:



Open loop solutions:

- Open loop solutions attempt to **prevent or avoid** congestion, ensuring that the network or part of it never enters into a **Congested State.**
- Open loop solutions attempt to solve the problems by good design to make sure it does not occur in the first place.
- Once the system is up and running, mid course corrections are not made. Don't consider the state of the network.
- Open solutions are somewhat **static** in nature.
- The open loop algorithms are further divided on the basis of whether these **acts on source or destination**.

Closed loop solutions:

Closed loop solutions allow the system to enter congested state, detect it, and remove it.

- Closed loop solutions are based on the concept of a **feedback loop.**
- Feedback loops have **three parts** when applied to congestion control

- **Step1: [Monitor/detect]** monitors the system to detect when and where congestion occurs.
- **Step 2: [Feedback]** Pass this information to places where action can be taken.
- **Step 3: [Correct]** and then finally adjust the system operation to correct the problem.

Explanation:

- For **detecting/monitoring** the congestion in network various **metrics** can be used. Some of them are:
  - average queue length
  - Number of packets that are timed-out
  - Average packet delay
  - Number of packets discarded due to lack of buffer space, and etc.
- There are several mechanisms are there to send the **feedback**
  - **Mechanism 1:** Router which detects the congestion send special **feedback** packets to the source (responsible for the congestion) announcing the problem. These extra packets increase the load at that moment of time, but these are necessary to bring down the congestion at a later time.
  - **Mechanism 2:** hosts or routers send out **probe packets** at regular intervals about the congestion. And source itself regulate its transmission rate, if congestion is detected in the network. This is a **pro-active approach**, as source tries to get knowledge about congestion in the network and act accordingly**.**
  - **Mechanism 3:** in every packet a field was reserved to mention the state of congestion. Whenever congestion gets above some threshold level. Or if routers detect this congested state, then fills that field and send to all outgoing packets, to warn the neighbors.

    - And finally congestion has been detected and this information has been passed to a place where the action needed to be done.
- Then finally correct the problem, there are two basic approaches that can overcome the problem.
    - **Increase the resources:** Separate dial-up lines or alternate links can be used to increase the bandwidth between two points where congestion occurs.
    - **Decrease the load:** decrease the rate at which a particular sender in transmitting packets out into the network.
- These closed loop algorithms are further divided into two categories:
    - **Implicit feedback:** The source reduces the congestion existence by making local observations.
    - **Explicit feedback:** Packets are sent back from the point of congestion to warn source.

We have two basic mechanisms to controlling congestion in a network those are:

- **Preventive (open loop):** These policies are designed to minimize congestion in the first place, rather than letting it happen and reacting after the effect.
- **Recovery (closed loop):** in this type of policies, allow the congestion into the network and find out congestion and remove it.

1.3 Congestion Prevention Policies:

The network should be designed to minimize the congestion, rather than letting it happen and reacting later. The congestion minimization can be achieved by **adopting appropriate policies** at various **layers** those are

- Data link layer
- Network layer
- Transport layer

Diagram:

Layer	Policies
Transport	• Retransmission policy • Out-of-order caching policy • Acknowledgement policy • Flow control policy • Timeout determination
Network	• Virtual circuits vs datagram in subnet • Packet queueing and service policy • Packet discard policy • Routing algorithm • Packet lifetime management
Data link	• Retransmission policy • Out-of-order caching policy • Acknowledgement policy • Flow control policy

Policies at Transport layer and Data link layer:

- **Retransmission policy:** whenever packets are discard because of timeout or error in header and etc... The discarded packets must retransmit again by the sender. If the sender uses the **GO back N** mechanism to retransmit the packets it put heavy load on network and it causes the congestion. Use **selective repeat** instead of Go-Back-N to minimize the congestion.

  Who retransmits the discarded packets?

  - In **datagram networks** the sending host retransmits discarded packets.
  - In **virtual circuit networks** the previous-hop router retransmits the packet when it fails to receive an acknowledgment.
- **NOTE 1:** flow control mechanisms are **stop and wait, GO back N** and **Selective repeat.**
- **Out of order caching policy:** at the network layer, IP protocol assigns a unique identification number to each and every packet to maintain the same order at sender and receiver. Receivers discard all out-of-order packets. These packets will have to be transmitted again later and creating extra load on network cause congestion. Here also **selective repeat** is clearly better than **GO back N.**

- **Acknowledgement policy:** if sender uses the GO back N to transmit the packet, then each packet is acknowledged immediately, acknowledgement packets generate extra traffic leads to congestion. Here also **selective repeat** is clearly better than **GO back N.**
- **Flow Control Policy:** Flow control is a scheme for the control of the data flow between sender and receiver. In other words, its **limit an amount of data** transmitted by the sending transport entity to a certain rate, that the receiver can manage. If the size of the receiver window is small may cause congestion.
- **NOTE 1**: flow control mechanisms
    - stop and wait
    - Sliding window protocols.
- **Timeout determination:** If the timeout interval is too short, it discarded with in less time and extra packets will be sent unnecessarily. If it is too long, congestion will be reduced but the response time will suffer whenever a packet is lost.

Policies at Network layer:

- **Virtual circuits Vs datagram in subnet:** the type of subnet we will use i.e. **virtual circuits** or **data grams** is also affects congestion since many congestion control algorithms work only with virtual-circuit subnets.
- **Packet queuing and service:** the congestion is also depends on how packets are stored/queued and in which order the packets are processed (round robin or priority based and etc...)
- **Discard policy:** it is the rule telling which packet is dropped when there is no space in buffer. A good policy can help minimize the congestion and a bad one can make it more badly.
- **Routing algorithms:** congestion can be minimizing by choosing that best routing algorithm (like OSPF) that spread the output packets to all output

channels instead of send too much traffic over already congested lines like DV, link state and etc…

- **Packet Life Time Management:** It deals with how long a packet may live before being discarded. if it is too long, lost packet waste the network bandwidth. If it is too short, packet may be discarded before reaching their destination.

1.4 Congestion Control in Virtual-Circuit Subnets:

Until now we were discussed about how to avoid the congestion from the occurring at first time. From now onwards we will discuss about how to handle the congestion **after occurrence**?

- Here we will describe some approaches to dynamically controlling congestion in virtual-circuit subnets.
    - Admission control
    - Careful establishment
    - Resource reservation

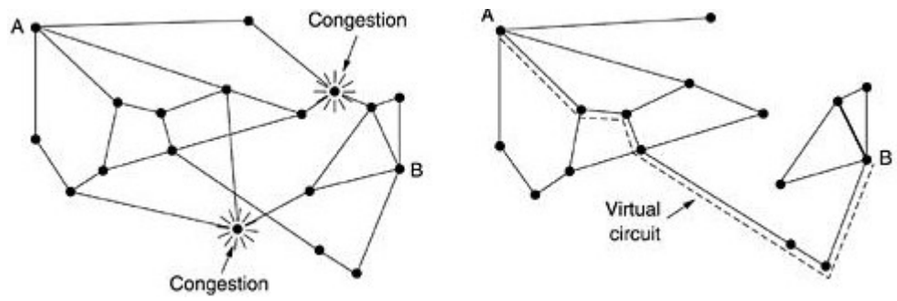
Admission control

- It is the one of the technique that is used when the congestion occurs to keep the **congestion not getting worse**.
- The idea behind this is “once the congestion was identified in a network (by one of the close loop solution mechanisms) no new virtual circuits are set up until the problem was gone away”. I.e. by allowing more things in getting matter more badly.
- It is easy to implement
- **Example:** In the telephone system when a switch gets overloaded, it also practices admission control by not giving dial tones.

Careful establishment

- An alternative approach is to allow new virtual circuits but carefully route/establish all new virtual circuits around problem areas.
- The working of this approach like this
    - **Step 1:** find out the congested routers in the network by using open loop mechanisms.
    - **Step 2:** construct a new subnet by eliminate those congested routers from the network.
    - **Step 3:** establish a new virtual connection.

Example:



Resources reservation:

- Congestion can be controlled by allocating maximum resources at the time of establishing the virtual circuit.
- The details of this virtual circuit are stored in the buffer of routers at the time of establishing virtual circuit.
- In this way, congestion is unlikely to occur on the new virtual circuits because all the necessary resources are guaranteed to be available.
- **Disadvantage:** waste of resources

1.5 Congestion Control in Datagram Subnets:

- In the above section we were discussed about how to achieve congestion in virtual circuits. Now have a look at the how to control the congestion in **datagram subnets**.
- In datagram subnet congestion was finding out by using the characteristics routers like **average queue length**, Number **of packets that are timed-out and Average packet delay** and etc… and **performance** or **utilization** of outgoing lines of router.
- A router can identify utilization of its outgoing lines by using below formula

$$U_{new} = a * U_{old} + (1-a) * f$$

Where

- 'f' is 0 (line is in use) or 1(not in use)
- 'a' is constant that defines how fast a router forgets its history.

- Whenever the value of 'U' move above the threshold value then the router thinks that that link in warning state. Then the router chooses any one of the following mechanism to control congestion.
  - Warning Bit
  - Choke Packets
  - Hop-by-Hop Choke Packets

1.5.1 The Warning Bit:

- In this mechanism a special **field** was used in header to indicate the warning state of resource.
- If the packet pass through the congested router, this field was filled with **1** else the value is **0**.
- When the packets reach its destination the value of this field was copied into the **ACK** and send to the source.

- As long as the router was in the warning state, it continued to set the warning bit, which meant that the source continued to get acknowledgements.
- If the source receives a warning state **ACK**, it reduces its transmission speed by ½ or ¼ or 1/8 and …… times.

1.5.2 Choke packets:

- In the previous mechanism/algorithm, it will take so much time to tell the source. To overcome that problem this mechanism introduces a new technique that is **just directly told to source.**
- **What is Choke packet?** A choke packet is a control packet produced at a congested node and transmitted back to a source node to reduce traffic flow.

Procedure:

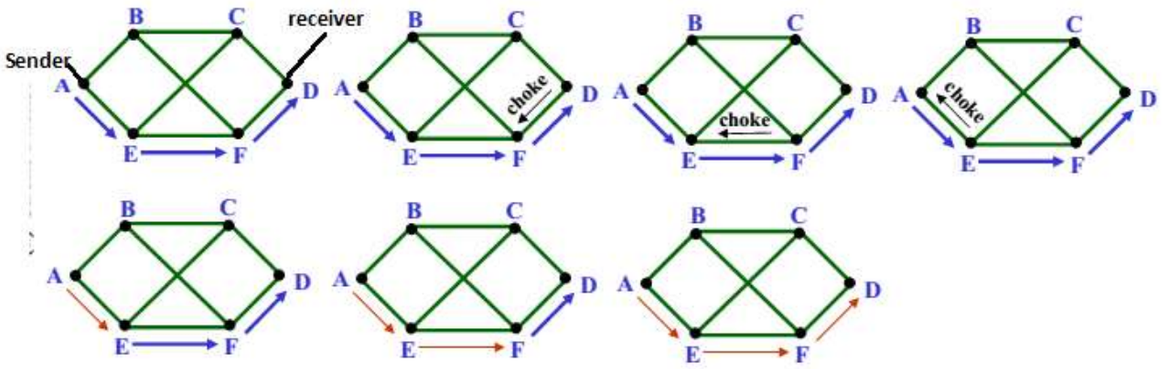
Step 1: router monitors the level of congestion around them.

Step 2: When congestion is present, they can send **choke packets** to the sender that say **`slow down'.**

Step 3: When the source host gets the choke packet, it is required to reduce flow sent to the specified destination by some percentage.

- In this mechanism sender listens the choke packets **periodically**. Because other packets aimed at the same destination are probably already under way and will generate yet more choke packets.

Example:



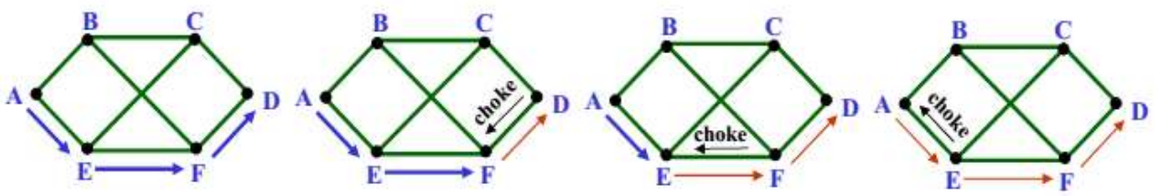
Advantages: Host sends as much data as it wants, the network informs it when it is sending too much.

Disadvantages: Defining time that the sender slows down his speed is critical.

1.5.3 Hop by hop choke packets:

- If there is long distance exist between source and destination, sending choke packets to sender takes so much time. To overcome this problem the new technique was introduced in hop by **hop choke packet.**
- With **hop-by-hop choke packets** each **intermediate router also reacts on a choke packet** by reducing its sending rate. For that it needs sufficient buffers to store the packets which still come in at a too high rate.

Example:



1.6 Load Shedding:

- When none of the above methods minimize the congestion routers can bring out the **load shedding** into action.

- The dictionary meaning of **shedding** is “accidentally discarding the carrying things”.
- **Load shedding** is a fancy way of saying that when router is busy by packets those they cannot handle, they just dropping them.
- The main goal of load shedding is **choose the right packet to discord**, so that the number of retransmitting packets are minimizes.
- Discarding packets does not do randomly. Discarding will be done by using one of the following mechanisms
    - Total packet discarding.
    - Age based discarding.
    - Priority based discarding.
    - Random Early Detection.

1.6.1 Total packet discarding:

- When the buffer fills and a packet segment is dropped, drop all the rest of the segments from that packet, since they will be useless.
- **Disadvantage:** Only works with routers that support segmentation and reassembling packets.

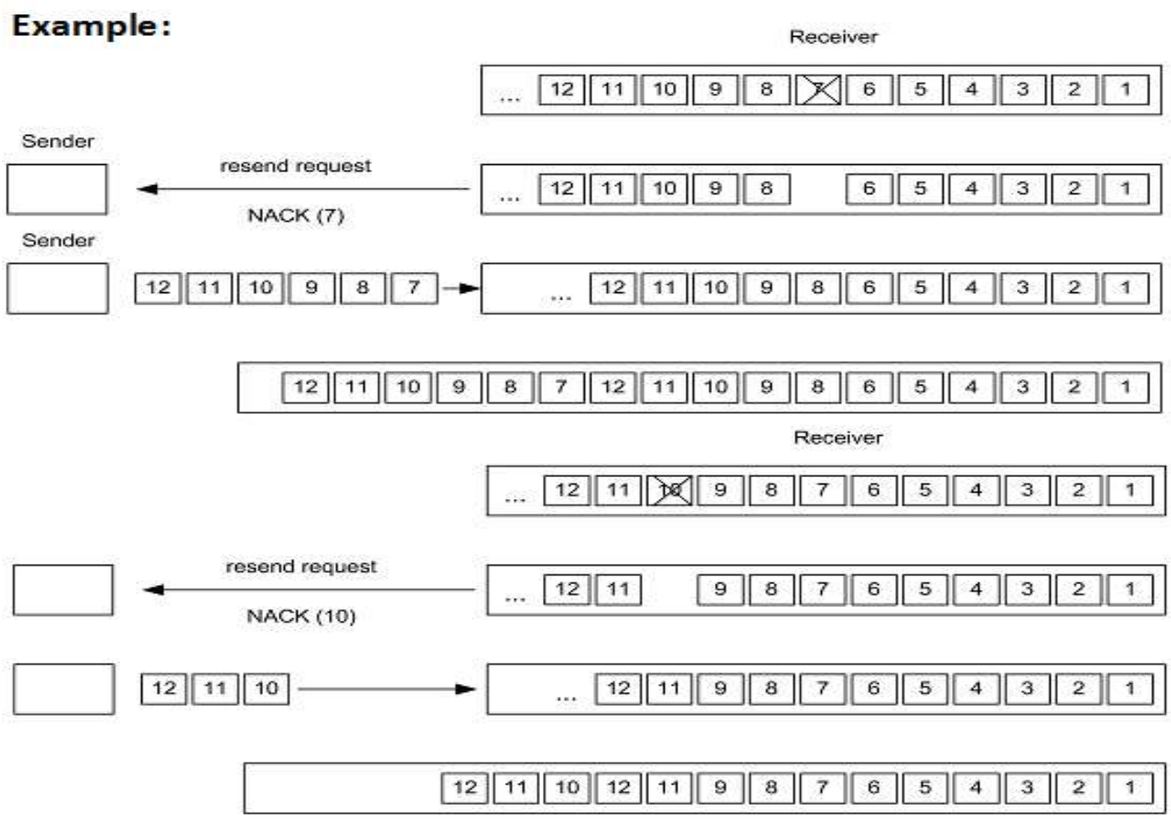
1.6.2 Priority based discarding:

- In this mechanism sources specify the priority of their packets.
- When a packet is discarded, the router chooses a low priority packet.
- **Disadvantage:** It requires hosts to participate by labeling their packets with priority levels.

1.6.3 Age based discarding

- When the router has to discard a packet, it chooses the **oldest** one in its buffer.
- This works well for **multimedia** applications which require short delays.
- This may not work so well for data/file transfer applications. because

- Discarding an **old packet is costs more than a new one**. Because if a router drops packet 7 out of 12, the sender will send packet 7 to 12 again. So, discarding one packet provokes retransmission of these six packets. Alternatively, the router can discard packet ten so that only three packets have to be resend.

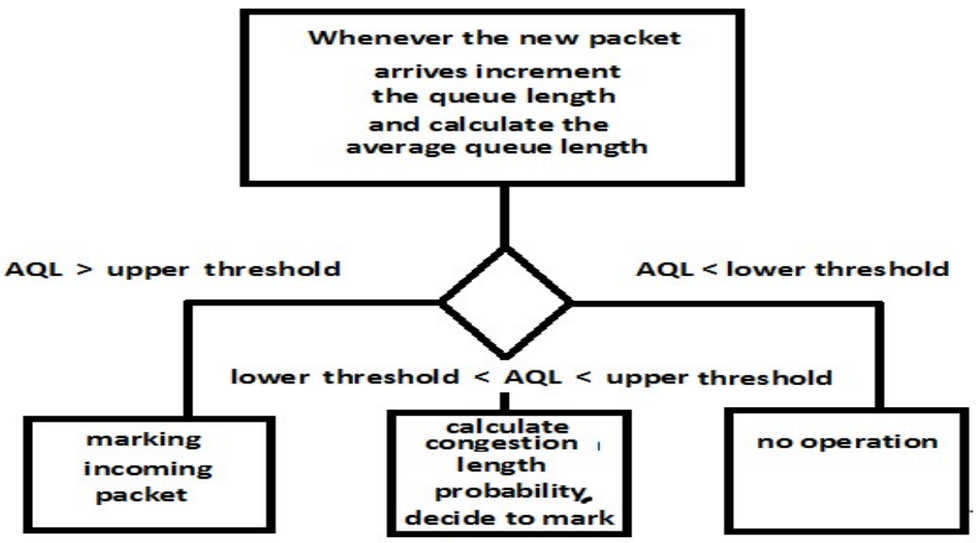


1.6.4 Random Early Detection:

- In This approach congestion will be controlled by either **marking or discarding** packets before the situation gets hopeless and congestion occurs i.e. **before all the buffer space of a router is really exhausted**.
- Each incoming packet triggers a new calculation of the average queue length(avgQueueLen) and afterwards the packet drop decision is made on

basis of two fixed thresholds, a lower and an upper one, thus the following three possible cases are distinguishable:

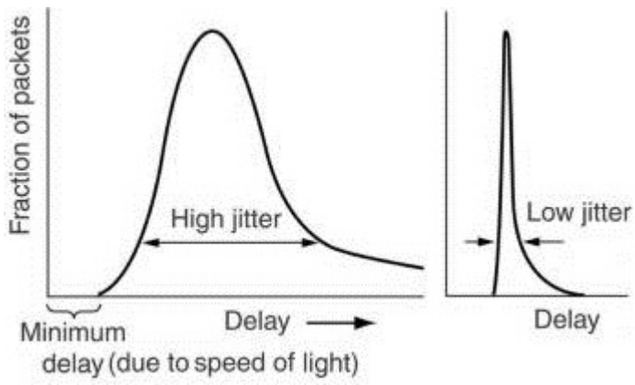
- **Step 1:** If the average queue length is below lower threshold, i.e. no congestion present.
- **Step 2:** If the average queues length is above upper threshold that means packets have to be discard/marked as a result of serious congestion state.
- **Step 3:** If average queue length is above lower threshold and below upper threshold, that means congestion is possible raising and marking/discarding probability hast calculated.



1.7 Jitter Control: The variation (i.e., standard deviation) in the packet arrival times is called **jitter**.

- Jitter will be classified into **two types**
  - **High jitter**
  - **Low jitter**
- **High jitter:** High jitter means the variation of delays is large;
- **Low jitter**: low jitter means the variation of delay is small.

Diagram:



- Suppose if four packets depart at times 0,1,2,3 and arrive at 20, 21, 22, 23, all have the same delay that is 20 units of time. On the other hand, if the above four packets arrive at 21, 23, 21, and 28, they will have different delays: 21, 22, 19, and 24.

- For applications such as audio and video, the first case is completely acceptable; the second case is not. Because in some applications, like video on demand, jitter can be compensated for by buffering at the receiver. For others, like Internet telephony or videoconferencing, the delay inherent in buffering is not acceptable.
- 99 percent of the packets be delivered with a delay in the range of 24.5 msec to 25.5 msec might be acceptable.
- The jitter can be bounded by computing the expected transit time for each hop along the path.
- When a packet arrives at a router, the router checks to see how much the packet is behind or ahead of its schedule.
- This information is stored in the packet and updated at each hop. If the packet is ahead of schedule, it is held just long enough to get it back on schedule. If it is behind schedule, the router tries to get it out the door quickly.

~~~~~~~~~~~~~~~~~~~~**END OF UNIT-2**~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~

CHAPTER-1 UNIT III

Quality of Service

❖ Introduction:

Up to now we talk about congestion control, in that we discourse about how to reduce the congestion to improve the performance of network, but with the growth of improvement in the traffic of networking, it is not only the matter of providing speedy services, it is also matter of providing qualitative services.

Quality of Service: the goal of QOS is improve the overall performance of a network. This was achieved by good design of network and protocols.

→Requirements:

- Different applications have different requirements regarding the handling of their traffic/flow in the network.
- **What is Flow?** A stream of packets from a source to a destination is called a flow.
- We already know that in a connection-oriented network, all the packets belonging to a flow follow the same route, and in a connectionless network, they may follow different routes.
- The **quality of service** that the flow is determine by using fallowing parameters those are
    - **Reliability:** each and every bit delivered to correct destination. It was achieved by **check summing** each packet and verifying the **checksum** at the destination.
    - **Delay:** the time that taken to travel a packet from source to destination.
    - **Jitter:** the arrival time difference between current packet and its next packet.

- **Bandwidth**: the amount of data that can be carried from one point to another in a given period of time (usually a second). Bandwidth is usually expressed in bits (of data) per second (bps).
- **Packet loss rate:** The rate at which packets are dropped, get lost or become corrupted (some bits are changed in the packet) while going through the network.

- The quality of service of a flow is depends on the **application type**. The following table explains the **QOS parameters** of different applications

Application	Reliability	Delay	Jitter	Bandwidth
E-mail	High	Low	Low	Low
File transfer	High	Low	Low	Medium
Web access	High	Medium	Low	Medium
Remote login	High	Medium	Medium	Low
Audio on demand	Low	Low	High	Medium
Video on demand	Low	Low	High	High
Telephony	Low	High	High	Low
Videoconferencing	Low	High	High	High

- To provide **QOS** in ATM networks, they classify their f**low** into four categories those are as follows:
  - Constant bit rate (e.g., telephony).
  - Variable bit rate
    - Real-time variable bit rate (e.g., compressed videoconferencing).
    - Non-real-time variable bit rate (e.g., watching a movie over the Internet).
  - Available bit rate (e.g., file transfer).

❖ Techniques for Achieving Good Quality of Service

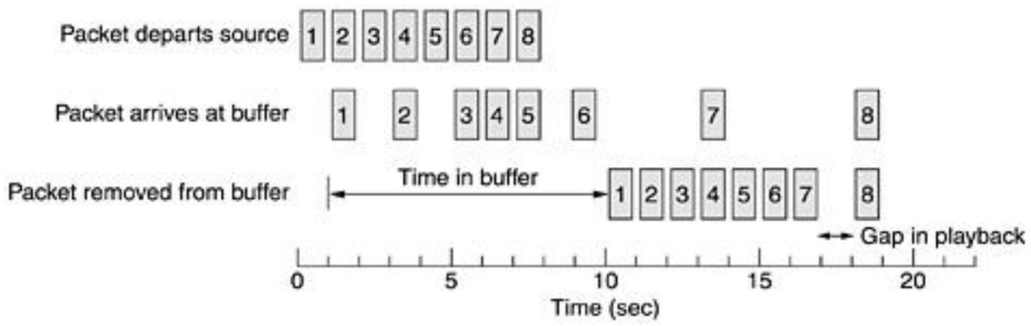
→ Over provisioning

- The name itself tells the entire story about it. This mechanism provides an easy solution to achieve QOS by provide high quality communication i.e. "**provide so much router capacity, buffer space, bandwidth and etc…**"
- **Advantage:** It works well for networks with predictable peak loads.
- **Disadvantage:** waste of resources. And predicting peak load of network.

→ Buffering

- Every packet in data flow stored/buffered on the receiving side before delivered. This may increase the delay but it reduces the jitter.

Example:



- For suppose if we want watch a video on YouTube. Whenever you click on a link of a particular video. The YouTube server (source) starts sending the data.
- Meanwhile because of the delay in network the packets are arrives at the receiver buffer like as show in above figure.
- Packet 1 is sent from the server at t = 0 sec and arrives at the client at t = 1sec. Packet 2 undergoes more delay and takes 2 sec to arrive. As the packets arrive, they are buffered on the client machine.
- After 10sec (Commercial Websites that contain streaming audio or video all use players that buffer for about 10seconds before starting to play) the video

start playing, at this time packets 1 through 6 have been buffered so that they can be delivered and removed from the buffer.

- **Disadvantage:** if any packet delayed so much that it is not available when its play slot comes up the playback must stop until it arrives.

→ **Traffic shaping:**

- In the above mechanism, source transmits the packets at a constant rate but this is not possible in all cases. For suppose if a server transmits data to multiple clients; maintain uniform transmission rate is almost impossible.
- **Traffic shaping mechanism** "Shape" the traffic before it enters the network. I.e. Smoothes out the traffic on the server side, rather than on the client side.
- Traffic shaping **controls the *rate*** at which packets are sent (not just how many).
- But the general question is Can we use the **sliding window protocol** to Limits amount transmitted? The answer is it helps to manage the buffer space at the receiver. Don't limit the rate at which it is sent.
- At the time of connection set up between sender and carrier (subnet), they both agree on a traffic pattern (**shape**). This agreement was called **service level agreement.**
- Two traffic shaping algorithms are:
    - Leaky Bucket Algorithm
    - Token Bucket algorithm

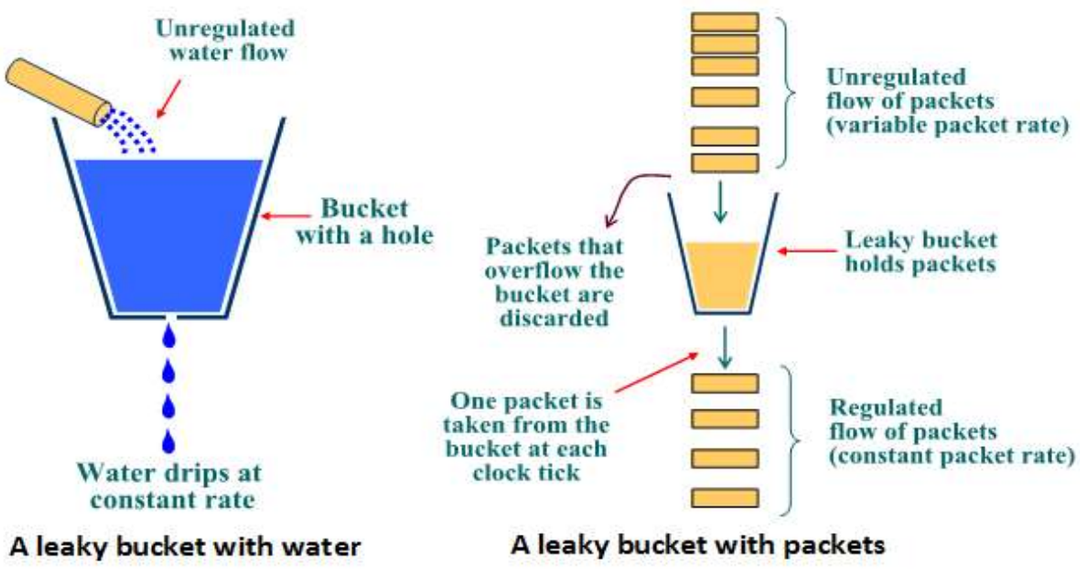
Leaky Bucket Algorithm/ peak rate limiter:

- The Leaky Bucket Algorithm used to control rate in a network. It is implemented as a single-server queue with constant service time. It was proposed a 19th century networking engineer by **Turner**.
- One day evening turner sit on his house balcony, his father is cleaning his motor cycle with a bucket of water and some cloth. That bucket has a hole

on lower part of it. The water was leaking at a constant rate, after some time the bucket was empty because of leakage. And his mother fill that bucket with water by using other bucket (it has better storage capacity) at the end of filling process the water start spills over the sides of bucket.

- From that situation he observe two important points
    - The water was leaking at a constant rate
    - Water start spills over the sides of bucket when the bucket is full.
- By using those two points he implements a new algorithm.

Diagram:



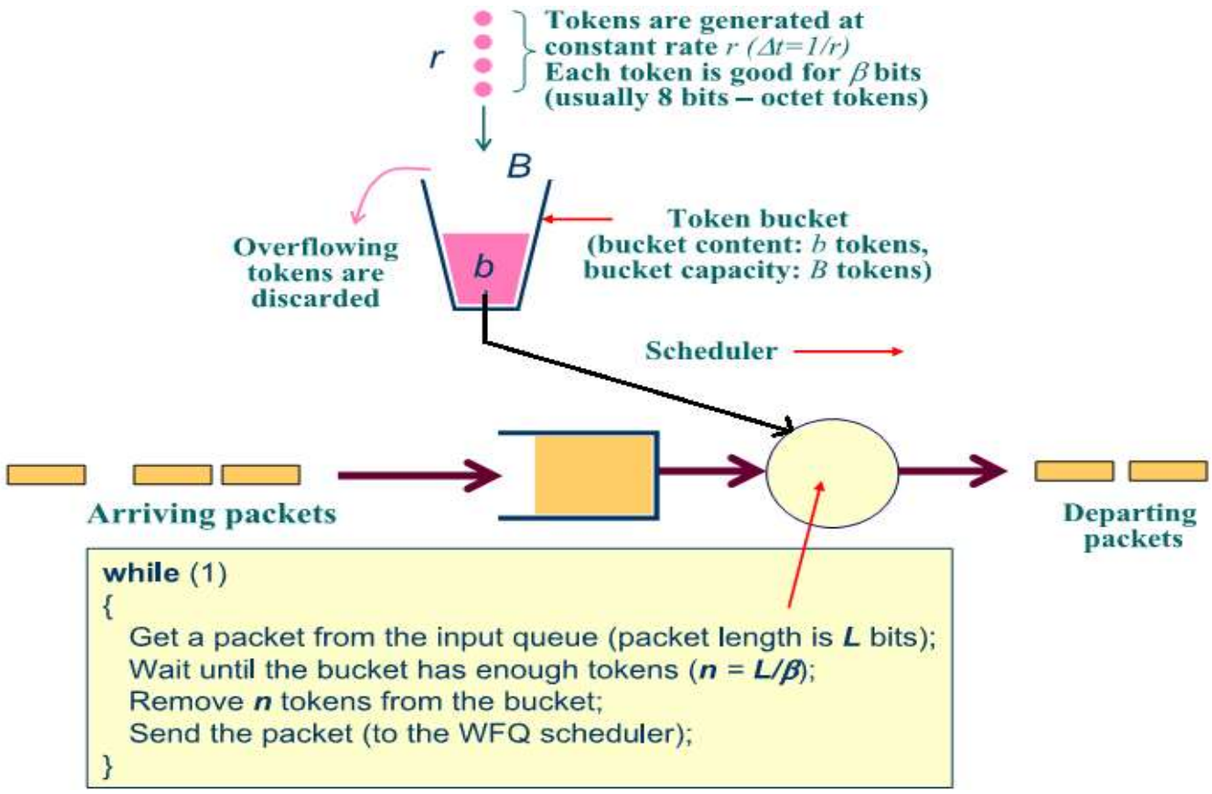
- **Algorithm:**
    - **Step 1:** Does nothing when input is idle.
    - **Step 2:** When a packet arrives, if buffer is full then discard packet. Else append to the buffer.
    - **Step 3:** At every **clock tick**, one packet is transmitted (unless the queue is empty).
- **Advantages:** control data rate in a network.
- **Disadvantage**: If data sending too fast, data is dropped.
- If all packets are of the same size the algorithm works as described in above.

- If packets have variable size, use number of bytes per clock-tick **For example** assumes the rule is 1024 bytes per clock-tick for every clock-tick the source can send: a single 1024 byte packet, or two 512 bytes packets, and etc ... it is also called **byte-counting leaky bucket.**

Token bucket algorithm:

- Leaky bucket algorithm does not allow sending **burst of packets**, but only at a specified rate. If there is no traffic for a certain period of time, the amount of unused bandwidth cannot be used for later packets; this is achieved by using a token bucket algorithm.

Algorithm:



Difference between leaky bucket and token bucket algorithm:

Sno	Leaky bucket	Token bucket
1	Bucket/buffer contains packets	Bucket/Buffer contains tokens
2	Leaky bucket algorithm does not allow idle host to save the packets	token bucket algorithm allow idle host to save the packets to send large burst latter
3	It discards the packets when bucket full	token bucket algorithm throws away tokens (i.e., transmission capacity) when the bucket fills up but never discards packets

→ Resource Reservation:

- This mechanism explains that spraying the packets over the network among multiple routers at random time does not guarantee anything.
- Set up something like virtual circuit between source and destination and all the packets that belong to the flow must follow this route.
- At the time of establish a route for flow it is possible to reserve a resources along that route. Three kinds of resources can potentially be reserved:
    - **Bandwidth:** capacity of the outgoing line.
    - **Buffer space:** buffer capacity of a router.
    - **NOTE:** we already have notes about this no need to discourse at here.
    - **CPU cycles:** how much time the router CPU takes to process the incoming packets.

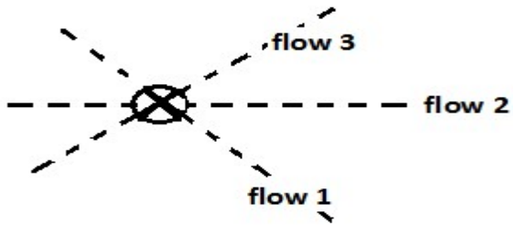
→Proportional Routing:

- Generally most routing algorithms try to find the best path form source to destination and send all traffic to that destination over the best path.

- The **proportional routing** mechanism proposed to provide a higher quality of service i.e. split the traffic for each destination over multiple paths.
- A simple method is to divide the traffic equally or in proportion to the capacity of the outgoing links.
- Example routing algorithm is OSPF.

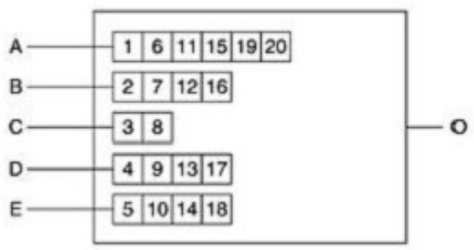
→Packet Scheduling:

- Suppose a router handle the multiple data flows, there is danger that one flow can occupies the maximum capacity of the router and starve all the other flows.



- For suppose a router follows the mechanism that is processing packets in the order of their arrival (FIFO) this is also not good enough because a speedy sender may capture the most of the capacity of router.
- There are so many packet scheduling algorithms are there, one of that algorithm is **fair queuing algorithm**.
- Generally each and every router have **separate queues for each output line**, one for **each flow;** When a line becomes **idle**, the router scans the queues **round robin**, taking the packet from the queue and given output line.
- **Disadvantage:** this algorithm gives more bandwidth to hosts that use large packets than to hosts that use small packets.
- Then in 1990 **Demers** suggest an improvement in this round robin, he said that simulate a byte-by-byte round robin, instead of a packet-by-packet round robin.

Example:



Packet	Finishing time
C	8
B	16
D	17
E	18
A	20

- **Disadvantage:** gives equal priority to all hosts and the scans queue repeatedly.
- In 1995 Sridhar and Varghese implement a new algorithm called **weighted fair queuing**. In this instead of sending byte by byte they can be given **two or more bytes** per tick.

Chapter 2 in UNIT III

Transport Layer: The Transport Service

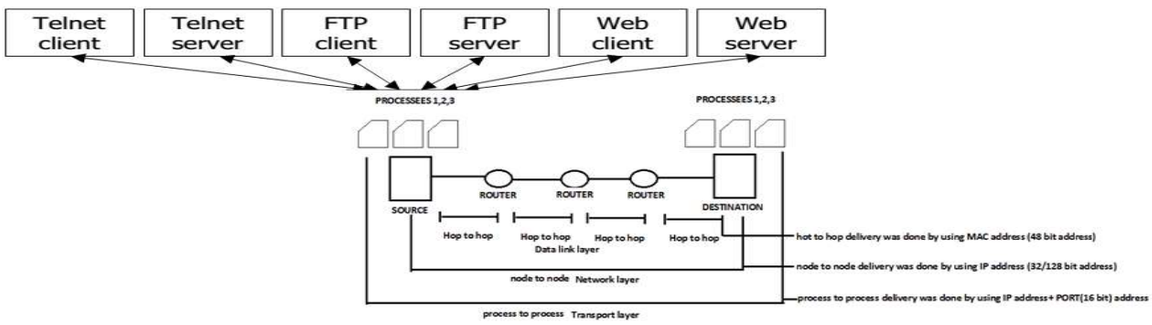
❖ Introductions:

Transport layer is just above the network layer, it is an intermediate layer. It uses the functionality of network layer and provides functionality to application layers.

The main goal of the transport layer is to provide **efficient, reliable, and cost-effective service** to its **processes.** (Process to Process delivery)

Just recall that layers 1, 2 and 3 are concerned with; the physical layer handles the **bits**; the data link layer deals with **local networks (hop by hop delivery)** and the network layer handles routing between hosts in networks (**node to node delivery**). In contrast, transport layer, is depends on the lower layers to handle the process of moving data between **processes (process to process delivery)**.

What is process? Process is nothing but a running program in host side. **Examples:** telnet client, server, ftp client, server, web client, web server and etc…

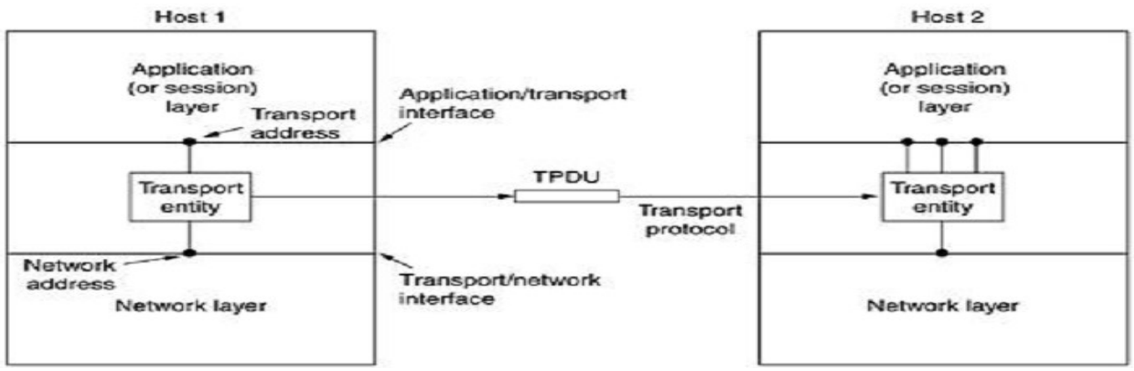


→Services provided to upper layers:

In general each and every layer in OSI or TCP/IP reference model uses the functionality of lower layer and provides functionality to upper layers. Similarly transport layer also provide its service to upper layers. That is the reason why we

can call bottom four layers as the **transport service provider,** and where as the upper layer(s) are the **transport service user**.

Diagram:



- services that the transport layer offers are
    - Addressing. (assign **port** address to individual process)
    - Connection oriented (reliable)/ connection less (unreliable) transport service.
    - Flow control, Error control.
    - Multiplexing and de-multiplexing.
    - Segmentation and reassembling.
- To achieve these services the transport layer uses the services provided by the network layer. The device that did this work was called **transport entity**.
- **Transport entity** is either software or hardware or combination of both within the transport layer. It is located in separate process of OS kernel (NIC).
- If we observe the sum of services provided by transport layer are similar to network layer and data link layer. Then the question is why we need two layers? Why not one?
- **Reason:** the transport layer code runs entirely on user machine, but the network layer code was runs on routers in network, network was operated by carrier (ISP), the problems occurs at network was not controlled by user machine.

- We can't solve these types of problems by better routers or putting more error handling techniques at data link layer. The only solution is put a layer on top of network layer that improves the quality of service.
- Network layer is responsible for logical communication between hosts and transport layer is responsible for logical communication between processes.
- **Example 1:** in a connection-oriented subnet, a transport entity sends a long transmission, in halfway through its transmission network connection has been suddenly terminated, with no indication of what has happened to the data currently in transit. Then transport entity can set up a new network connection to the remote transport entity. Using this new network connection, it can send a query to its peer asking which data arrived and which did not, and then pick up from where it left off.
- These types of problems are solved by using **SEQUENCE NUMBERS** and **ACKNOWLEDGEMENT NUMBERS**.

→Transport service primitives:

What is Primitive?

In computer programming, a primitive is a basic **interface** or **segment of code.** That can be used to build more sophisticated interfaces.

What is transport service primitive?

The primitive, which was used by the **application program/process** running on host side to access the transport services, is known as transport service primitive.

The transport service is similar to the network service, but there are also some important differences exist between them, those are

- Network services are **unreliable**, because services are depends upon real networks, and it uses the unreliable **IP protocol**.

- The transport services are **reliable**, even though the services offered by real network are unreliable by using TCP over IP, the transport layer provides reliable services. Of course it is also provide **unreliable services by using UDP**, but for limited applications like streaming, multimedia applications.
- Network services are used by only transport entities, but transport services are used by many application programs.

The primitives for simple transport service [client server application]:

The following diagram shows the primitives that are generally used to access the transport services.

Primitive	TPDU sent	Meaning
LISTEN	(none)	Block until some process tries to connect
CONNECT	CONNECTION REQ	Actively attempt to establish a connection
SEND	DATA	Send information
RECEIVE	(none)	Block until a DATA TPDU arrives
DISCONNECT	DISCONNECTION REQ	This side wants to release the connection

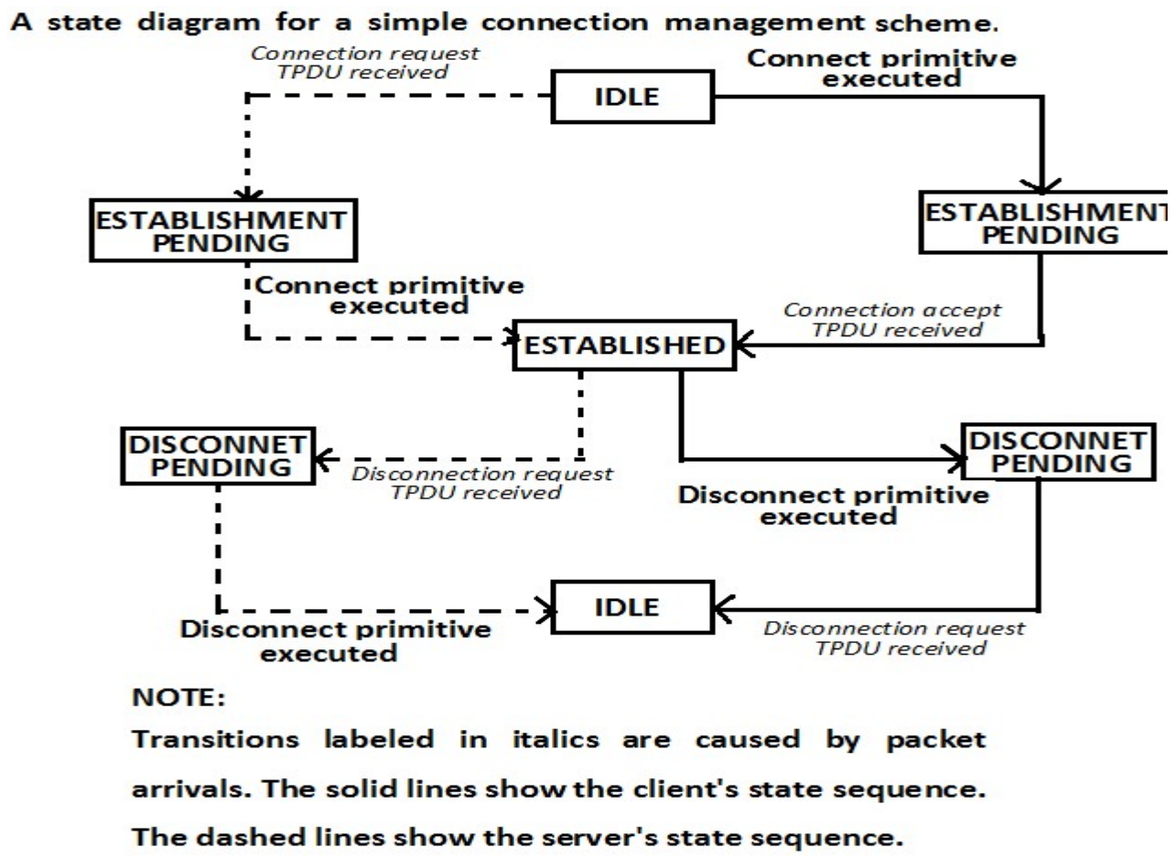
Procedure:

- **Step1:** The server executes a LISTEN primitive, by calling a library procedure that makes a system call to **block the server** until a client ready for communication.
- **Step2:** When a client ready for communicate with the server, it executes a CONNECT primitive.
- **Step 3:** The transport entity carries out this primitive by blocking the client and sending CONNECT REQ TPDU packet to the server.
- **Step 4:** When it arrives, the transport entity checks to see that the server is blocked on a LISTEN (i.e., is interested in handling requests). It then unblocks the server and sends a CONNECTION ACCEPTED TPDU back to the client.
- **Step 5:** When this TPDU arrives, the client is unblocked and the connection is established.

- **Step 6:** after establishing the connection, both are communicate by using SEND and RECEIVE primitives.
- **Step 7:** When a connection is no longer needed, it was disconnected by issuing the DISCONNECT primitive. This disconnection was done in two ways.

    - **Asymmetric:** either transport entity can issue a DISCONNECT primitive, which results in a DISCONNECT TPDU being sent to the remote transport entity.
    - **Symmetric:** each direction will be closed separately of the other. When one side issued a DISCONNECT, it means it has no more data to send but it prepared to carry on accepting data from other direction. The connection was terminated when it receives the DISCONNECT from both sides.

Diagram:



→Berkeley sockets:

Berkeley sockets are another **set of primitives** (or) API (Application Program Interface) i.e. a set of function calls used by processes to access the **transport services**.

It was developed in the early **1980s** at the **University Of California, at Berkeley**. Originally released with 4.2BSD (Berkeley Software Distribution) UNIX based operating system in 1983.

What is socket?

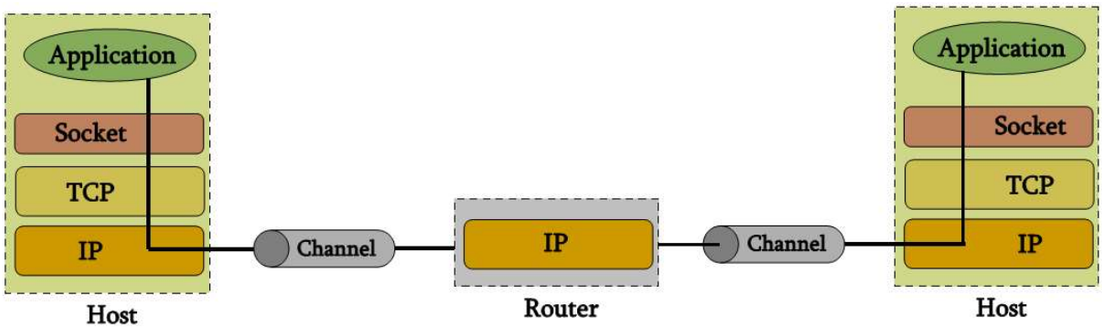
End point of communication is known as Socket. It was identified by using socket adders [IP address: port number]. Each process establishes its own socket.

Example: 192.168.1.1**:** 21

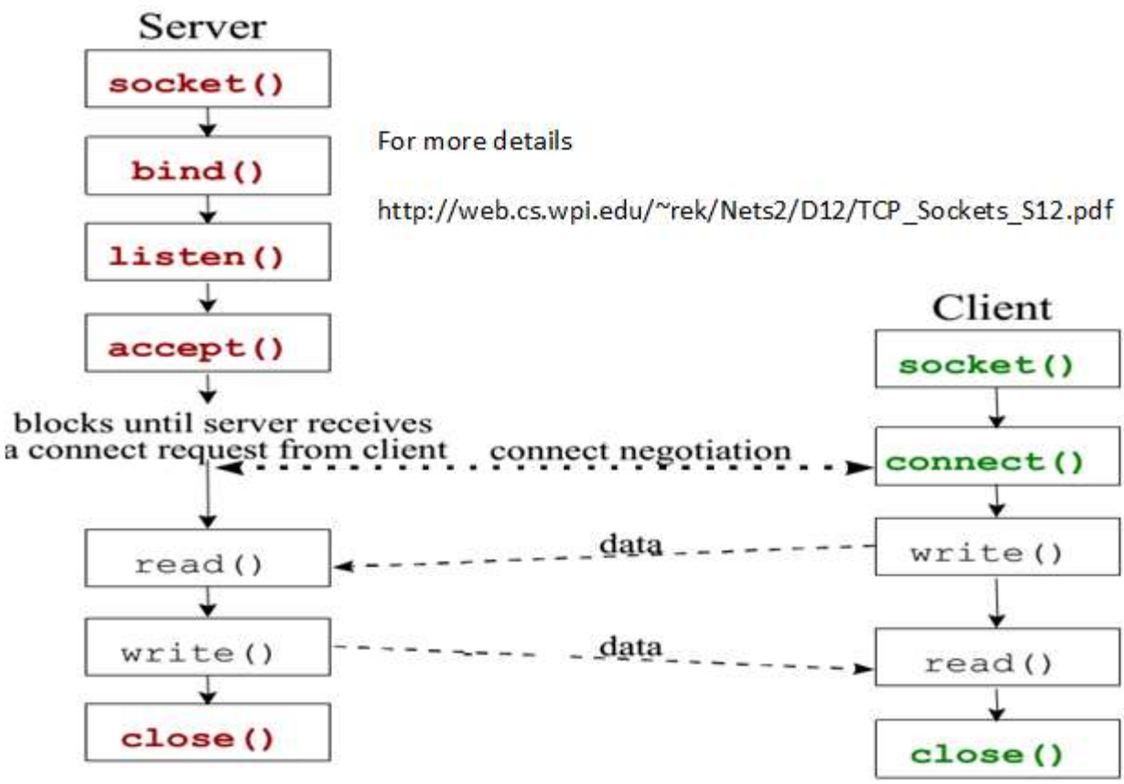
The following diagram contains the list of socket primitives for TCP:

Primitive	Meaning
SOCKET	Create a new communication end point
BIND	Attach a local address to a socket
LISTEN	Announce willingness to accept connections; gize
ACCEPT	Block the caller until a connection attempt arriv
CONNECT	Actively attempt to establish a connection
SEND	Send some data over the connection
RECEIVE	Receive some data from the connection
CLOSE	Release the connection

The socket interface:



The TCP socket system calls:



Example: Berkeley client server model

NOTE: the client needs to know of the existence of and the address of the server, but the server does not need to know the address of (or even the existence of) the client prior to the connection being established.

The system calls for establishing a connection are somewhat different for the client and the server.

The steps involved in establishing a socket on the ***server* side** are as follows:

1. Create a socket with the **socket ()** system call.
2. Bind an address to socket using the **bind ()** system call. An address consists of a port number on the host machine.
3. Listen for connections with the **listen()** system call
4. Accept a connection with **accept ()** system call. This call typically blocks until a client connects with the server.

   NOTE: When you call **listen** on a socket, it tells the networking stack in the operating system to take necessary actions so that the client connections are buffered and are ready to be accepted by your application. When you call accept, it asks the networking stack to create a new socket for the first client connection that is pending in the queue and return it.

5. Send and receive data
6. Terminate the connection using **close ()** system call.

The steps involved in establishing a socket on the ***client* side** are as follows:

1. Create a socket with the **socket()** system call
2. Connect the socket to the address of the server using the **connect()** system call
3. Send and receive data. There are a number of ways to do this, but the simplest is to use the **read ()** and **write ()** system calls.
4. Terminate the connection using **close ()** system call.

Chapter 3 in UNIT III

Elements of Transport Protocols

Concepts: *Addressing, Connection Establishment, Connection Release, Flow Control and Buffering, Multiplexing, Crash Recovery,* ~~***Simple transport Protocol.***~~

❖ Introduction:

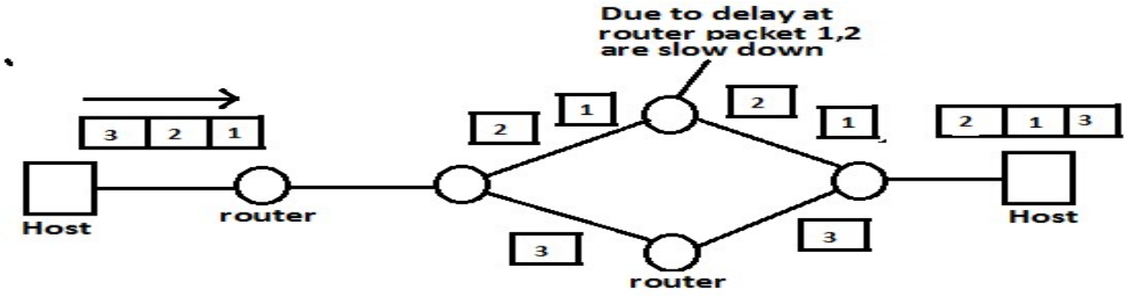
What is transport protocol?

The protocol that was used between two transport entities to implement the services of transport layer is known as **transport protocol.**

In the previous section we ware talk about some of the similar functionalities performed in both network layer and transport layer and how they are different. Similarly DLL and TL are also performs some same functionalities like flow and error control. However there are significant differences between them.

Difference between DLL and TL

- **Mechanism:** DLL is responsible for node to node delivery. TL is responsible for process to process delivery.
- **Environment:** in DLL the communication entities are connected by using a physical channel. Where as in TL, physical channel was replaced by subnet.
- **Reliability:** DLL is responsible for node to node error control; TL is responsible for process to process error control. Node to node error control dose not guarantees the process to process error control.



❖ Elements of transport protocol:

The following are the list of elements we talk about related to transport protocol concern.

- Addressing.
- Connection Establishment.
- Connection Release.

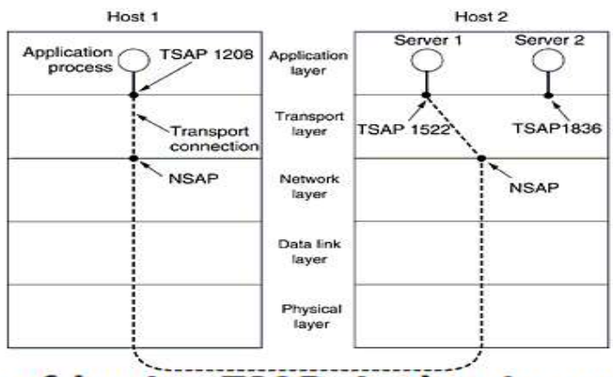
- Flow Control and Buffering.
- Multiplexing, and
- Crash Recovery.

→ Addressing:

From the basics we know that transport layer is responsible for process to process to delivery. I.e. the application process in one host establishes a transport connection with the remote process. It needs to specify which process to connect to connect by using transport address. Generally that address is known as **port address**.

We will use the generic term **TSAP** (Transport Service Access Point) examples: FTP 21, HTTP 80, HTTPS 443,110 POP3, DNS 53 and etc…. similarly send points in the network layer ware called **NSAP**s. IP addresses are examples of NSAPs.

Relationship between TSAP and NSAP:



The purpose of having TSAPs is that in some networks, each computer has a single NSAP, so some way is needed to distinguish multiple transport end points that share that NSAP.

Example: Transport connection scenario between client and day of time server.

Step 1: A time of day server process on host 2 attaches itself (assigned by operating system) to TSAP 1522, and wait for an incoming call.

Step 2: client process on host 1 find out time of day server process on host 2, and establish a connection by specifying TSAP 1208 as the source and TSAP 1522 as the destination.

Step 3: Then client process then sends over a request for the time.

Step 4: The time server process responds with the current time.

Step 5: And finally transport connection is then released.

Up to now everything is ok, but the question is **how does the user process on host 1 know that the time-of-day server is attached to TSAP 1522?**

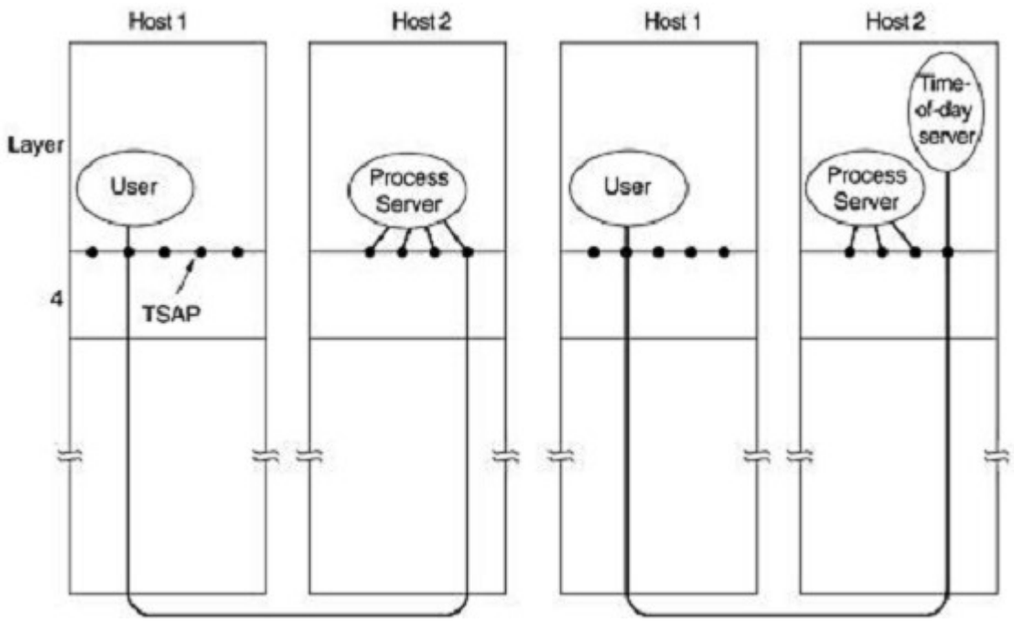
Possibility 1: [initial connection protocol] all the process has the stable TSAP's (well known ports) but this is not as much efficient as we think. Because there are only 2^{16} ports available and some of them are already reserved. If there are potentially many server processes, most of which are rarely used, and it is wasteful to have each of them active and listening to a stable TSAP address. So we need a better solution.

Possibility 2: [process server / port maper] Instead of every conceivable server listening at a well-known TSAP, each machine that wishes to offer services to remote users has a special **process server** that acts as a proxy for less heavily used server.

At the time of registration, the server registers it service name and corresponding TSAP.

- **Step 1:** listen on well-known TSAP.
- **Step 2:** user connect to process server and specify the service name.
- **Step 3:** process server sends back TSAP address of that server.

Example: How a user process in host 1 establishes a connection with a time-of-day server on host 2.



→Connection Establishment:

Generally connection establishment is looks like a simple task like transport entity to just send a CONNECTION REQUEST TPDU to the destination and wait for a CONNECTION ACCEPTED reply. But it is not that easy. It suffers from the serious problems occurs in network those are

- **Packet loss.**
- **Duplicate packets.**
- **Store**: Traffic (heavy load) in network.

Example:

Let us take an example, a user1 want to transfer money to other user1. For that, user establishes a connection with a bank, sends message to bank for transfer money to the account of user2, and then releases connection.

In mean while the packet ware **duplicated and stored** in network due to some problems. After connection release these duplicate packets ware popped out from the subnet and reaches the bank. It has no reason what so ever to tell that these are duplicates, it simply assumes that this is a second, independent transaction, and transfers the money again.

In the above example we notify the problems of delayed duplicates, with special emphasis on algorithms for establishing connections in a reliable way.

1. Connection identifier:

Step 1: At initiating side, give a unique identifier for each connection (sequence number), when new connection it was incremented by 1.

Step 2: After connection was released, both entity updates its tables.

Step 3: Whenever a new connection request comes in, it could be checked against the table and identifies if is old one or new one.

Disadvantage: If transport entity crash/loss its memory, it will never know what are the previous connection identifiers.

2. Kill aged packets:

Main idea behind this concept is "**no packet lives longer than some maximum time"**. Packet lifetime can be restricted to a known maximum using one (or more) of the following techniques:

- **Restricted subnet design:** include new method to prevent looping
- **Putting a hop counter in each packet:** -1 at each hop
- **Time stamping each packet:** clock must be synchronized

Disadvantage: these methods are also suffered from their individual problems, consider a time stamp technique, it requires each and every router must synchronize with clock. And etc…

3. **'T' method:**

For reliable connection establishment, we will need to guarantee not only that a packet is dead, but also that all ACK to it are also dead. For that let T be the maximum life time of a packet, **whenever a packet is sent wait for a time T before sent next packet.** This will guarantee that all data and ACKs were dead.

Disadvantage: waiting for a time 'T' may cost the performance of network.

4. Tomlinson method:

With packet lifetimes bounded, it is possible to devise a **foolproof** way to establish connections safely. And in 1975 Tomlinson proposed a new method that solve problems but introduce a new oddness of its own.

Step 1: each and every host must have **a time-of-day clock;**

- It is running continually **even if the host crash.** It increment for every uniform interval.
- The clock is stored in the form of binary counter. And that too **the number of bits in the counter must greater than or equal to the number of bits in the sequence numbers**.

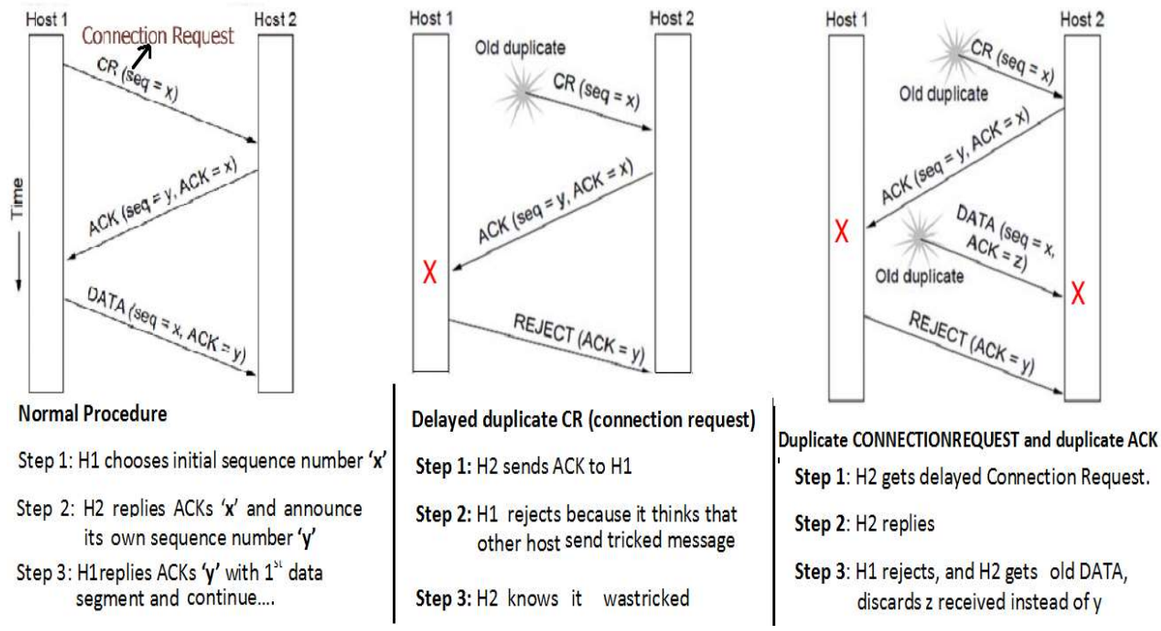
Step 2: When a connection is set up, the low-order k bits of the clock are used as the initial sequence number (also k bits). i.e. Never reuse a sequence number x within the lifetime T for the packet with sequence number x.

Disadvantage: getting both sides to agree on the initial sequence number. Example : Suppose, for example, that connections are established by having host1 send a CONNECTION REQUEST TPDU containing the proposed initial sequence number and destination port number to a remote peer, host 2. The receiver, host 2, then acknowledges this request by sending a CONNECTION ACCEPTED TPDU back. If the CONNECTION REQUEST TPDU is lost but a delayed duplicate CONNECTION REQUEST suddenly shows up at host 2, the connection will be established incorrectly.

5. Three way handshake:

To solve these types of problems **Tomlinson** introduce a new mechanism called "**THREE WAY HANDSHAKE".**

Each and every entity checks with other entity that the connection request is correct or not by using the TCP 32-bit sequence number. No clocks are used in TCP because attackers can predict clock.



→Connection release:

Compare with the connection establishment, releasing is easy task. As we discourse in earlier, there are two styles of terminating a connection.

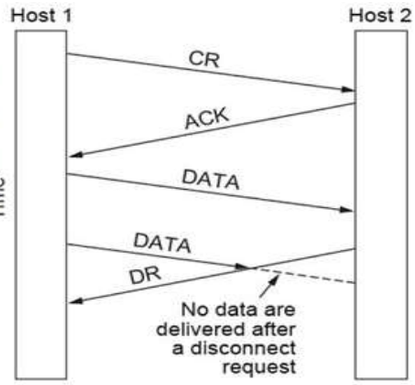
Asymmetric release: any one of them hangs up that's it, the connection is broken

Symmetric release: it treats the connection as two separate unidirectional connections and requires each one to be released separately.

1. Asymmetric release suffers from the **data loss**, consider the following

Example:

H1 sends a CONNECTION REQ and H2 response with ACK. The connection was established successfully. Now H1 sends TPDU1 to H2, its came safely. Now H1 send other TPDU2, Unfortunately, H2 issues a DISCONNECT before TPDU2 arrives. This causes data loss.



2. Symmetric release: asymmetric release causes the data loss. So a more sophisticated release is needed to avoid data loss. One such release is symmetric release.

The working of symmetric release is H1 says: **I am done. Are you done too?** If H2 responds: **I am done too.** Then the connection can be safely released. Unfortunately this does not always work.

Example: two army problem

Let us assume there are two army groups (white, blue and both are oppositions). The white army is larger than either of the blue armies alone, but together the blue armies are larger than the white army. I.e. if blue army wants win against white army, both blue armies fight together. For that they must communicate each other but the channel is unreliable.

The two-army problem.

The question is Does a protocol exist that allows the blue armies to win?

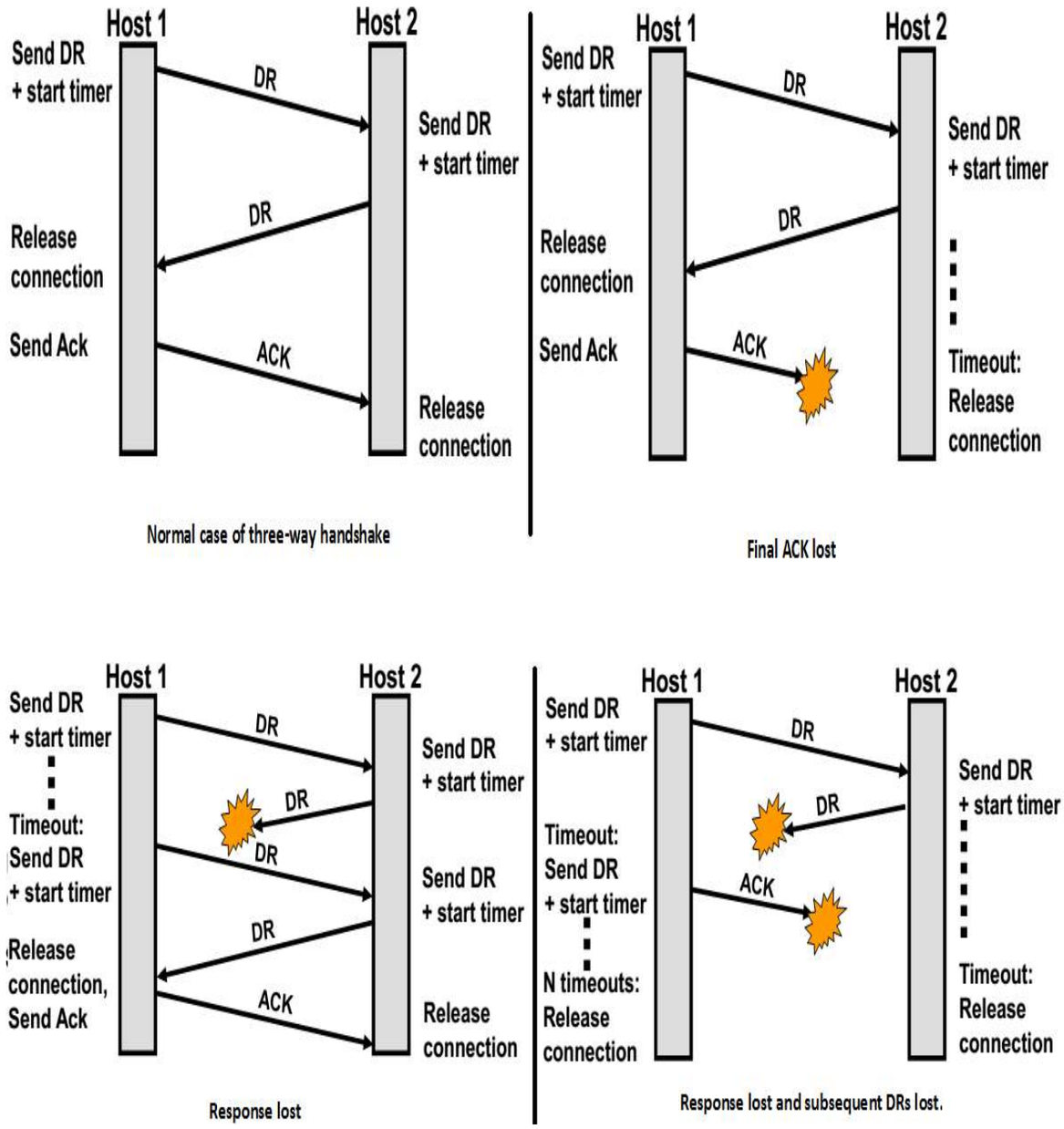
Case 1: [two way handshake] suppose that the commander of blue army #1 sends a message "can we attack white army today?" And his reply gets safely back to blue army #1. But the commander of army #2 does not know his replay reach army #1 or not. And he does not commit to attack.

Case 2: [three way handshake] the idea is "**initiator of the original proposal must acknowledge the response"** but that does not help either.

Case 3: [last message is essential] the idea is the last message that was send by the any one is fixed. But the problem is the sender of the final message can never be sure of its arrival. So it will not attack either.

Just substitute ''disconnect'' for ''attack'' In above. If neither side is prepared to disconnect until it is convinced that the other side is prepared to disconnect too, the disconnection will never happen.

Four scenarios of releasing using a three-way handshake



Solution:

The final solution is "**automatically disconnect if no segment was received in a time X sec".** The timer must reset after each segment. And send dummy packets to keep connection alive. TCP generally uses the symmetric close with each side independently close half connection.

4.Flow Control and Buffering

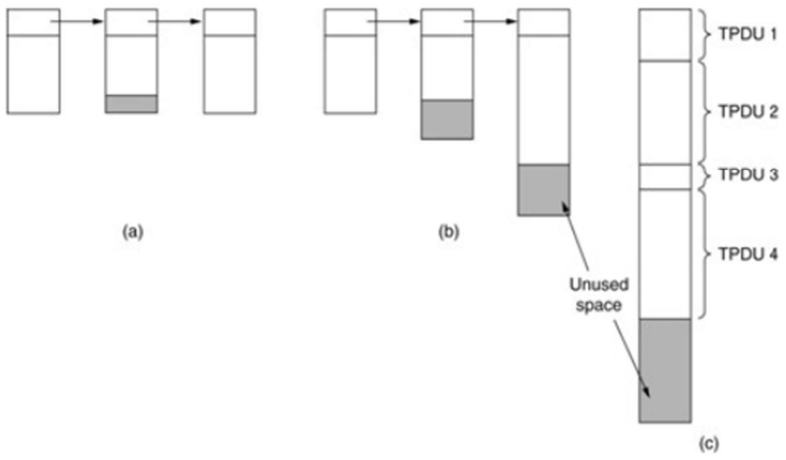
- How connections are managed when in use:

Flow control

- In some ways the flow control problem in the transport layer is the same as in the data link layer, but in other ways it is different.
- The basic similarity is that in both layers a sliding window or other scheme is needed on each connection to keep a fast transmitter from overrunning a slow receiver.
- The main difference is that a router usually has relatively few lines, whereas a host may have numerous connections.
- This difference makes it impractical to implement the data link buffering strategy in the transport layer.

- if the network service is unreliable, the sender must buffer all TPDUs sent, just as in the data link layer.
- with reliable network service, other trade-offs become possible.
- In particular, if the sender knows that the receiver always has buffer space, it need not retain copies of the TPDUs it sends.
- if the receiver cannot guarantee that every incoming TPDU will be accepted, the sender will have to buffer anyway.
- In the latter case, the sender cannot trust the network layer's acknowledgement, because the acknowledgement means only that the TPDU arrived, not that it was accepted.

- Even if the receiver has agreed to do the buffering, there still remains the question of the buffer size.
- If most TPDUs are nearly the same size, it is natural to organize the buffers as a pool of identically-sized buffers, with one TPDU per buffer, as in Fig (a).
- if there is wide variation in TPDU size, a pool of fixed-sized buffers presents problems.
- If the buffer size is chosen equal to the largest possible TPDU, space will be wasted whenever a short TPDU arrives.
- If the buffer size is chosen less than the maximum TPDU size, multiple buffers will be needed for long TPDUs, with the attendant complexity.



(a) Chained fixed-size buffers. (b) Chained variable-sized buffers.
(c) One large circular buffer per connection.

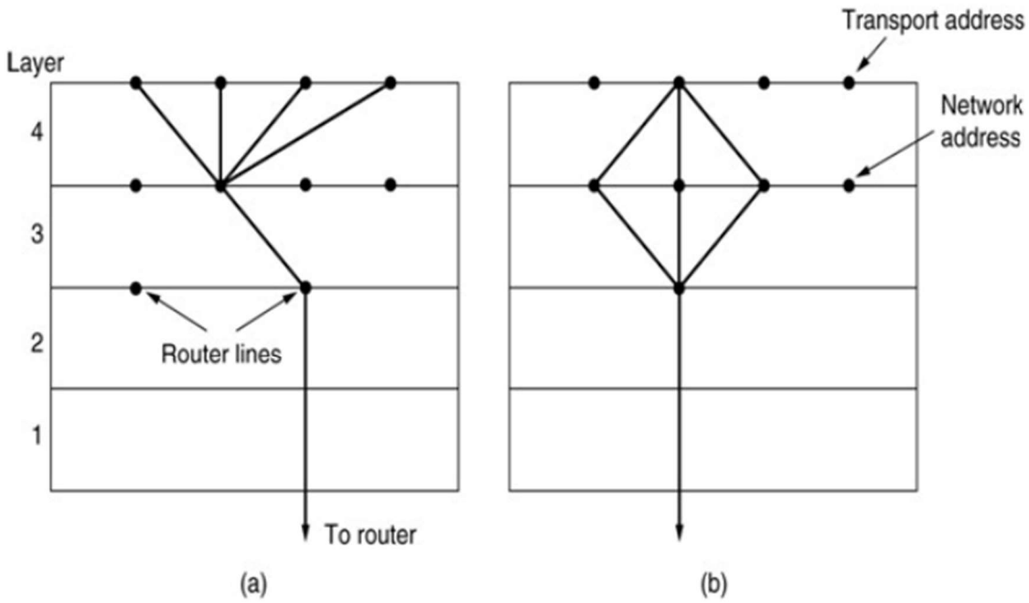
www.cshstudent.co.nz

- Another approach to the buffer size problem is to use variable-sized buffers, as in Fig(b).
- The advantage here is better memory utilization, at the price of more complicated buffer management.
- A third possibility is to dedicate a single large circular buffer per connection, as in Fig. (c).
- This system also makes good use of memory, provided that all connections are heavily loaded, but is poor if some connections are lightly loaded.

5.Multiplexing

- In the transport layer the need for multiplexing can arise in a number of ways.
- For Eg, if only one network address is available on a host, all transport connections on that machine have to use it.
- When a TPDU comes in, some way is needed to tell which process to give it to.
- This situation, called **upward multiplexing** , is shown in fig a.
- In this figure, 4 distinct transport connections all use the same network connection (e.g., IP address) to the remote host.
- If a user needs more bandwidth than one virtual circuit can provide, a way out is to open multiple network connections and distribute the traffic among them on a round-robin basis, as indicated in fig b.
- This modus operandi is called **downward multiplexing**

(a) Upward multiplexing. (b) Downward multiplexing.



~~~~~~~~~~~~~~~~~~~~END OF UNIT-3~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~

Chapter 1 in UNIT IV

Internet Transport Protocols (UDP)

Concepts: *Introduction to UDP, Remote Procedure Call, the Real-Time Transport Protocol.*

❖ Introduction:

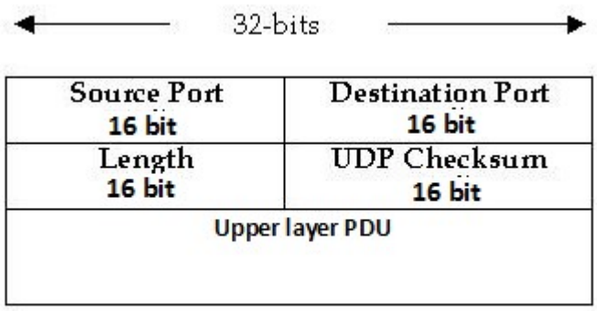
According to the old knowledge transport layer is responsible for process to process delivery. For that, transport layer uses the following four protocols

- UDP (User Datagram Protocol)
- TCP (Transmission Control Protocol)
- SCTP (Stream Control Transmission Protocol)
- DCCP (Datagram Congestion Control Protocol)

→ UDP [User Datagram Protocol]:

We already know that transport layer provides both connection oriented services and connection less services.

- UDP is a protocol that was used to provide connection less service. I.e. send the segments without establishing connection. The protocol was designed by **David P. Reed in 1980.**
- It provides a **best-effort** datagram service between processes; "Best Effort" service is one which **does not provide full reliability**. It usually performs limited **error control** (e.g. discarding all frames which may have been corrupted) and may also provide **limited retransmission** (e.g. CSMA/CD).
- UDP does not guarantee of **delivery, ordering, or duplicate protection**.
- Each and every UDP segment consists of 8 byte header followed by upper layer PDU.
- **UDP header:**



UDP segment structure

- **Source** and **destination port [16-bit decimal]:** source port indicates the port address/ TSPA of the sender process (application program). Similarly destination port indicates the port address of the destination process.

  According to basics we know that port address is of length 16 bit. I.e. totally 2^{16} =65535 port numbers are possible. In that

  - Port numbers 0 – 1023 are **Well Known Ports**. (run by root)
  - Port numbers1024 – 49151 are **Registered Ports** (run by user)
  - Port numbers 49152 – 65535 are **Private or Dynamic Ports**.

  NOTE: the sender port address was assigned by OS in sender machine by randomly selecting port number from the dynamic ports.

- **Length filed [16-bit decimal]:** length field represents the length of the UDP datagram (both **UDP header + upper layer PDU**).
  Maximum UDP data gram size: in UDP header, length field contains 16 bits. I.e. maximum size is up to **2^{16}-1 = 65535 bytes.** But segment Includes 8 bytes UDP header and 20 byte IP header 20+8=28. I.e. 65535 - 28 = 65507 bytes. The maximum size of UDP datagram is 65507 bytes.
- **Checksum [16 bit]:** similar mechanism that was used in IP datagram (addition of 1's complement).

→Remote Procedure Call:

Introduction:

Sending a message to a remote host and getting reply back is a lot like making a function call in a programming language. This observation has led people to arrange request-reply interactions on networks in the form of procedure calls.

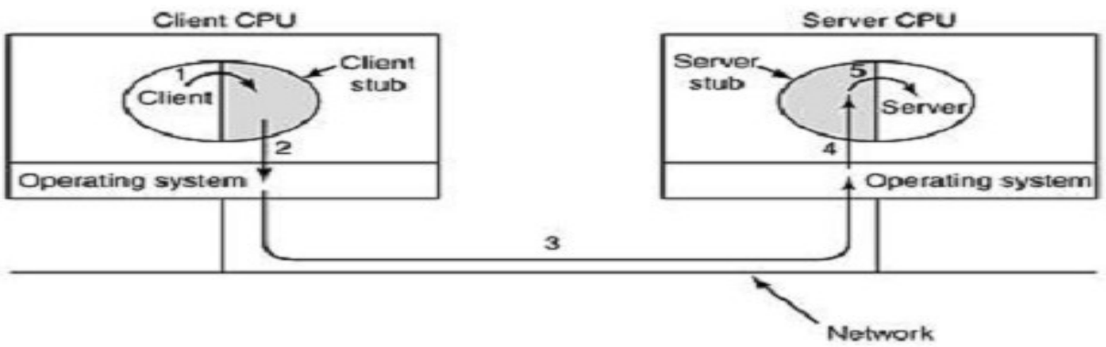
This idea was taking on to the cards **by Birrell and Nelson** in 1984. They suggest that, allowing programs to call procedures located on remote hosts.

What is remote procedure call?

Remote Procedure Calls are a collection of **library functions** was developed by Sun Microsystems. RPCs are used to **allow programs on one system execute code in another address.** The other address can be a remote system on a network or different processor in the local system.

Before we start the RPC procedures, let's look at some important terms

Remote Procedure Calls are performed in the following order:



Steps in making a remote procedure call

Client: The program on a system requesting the execution of code

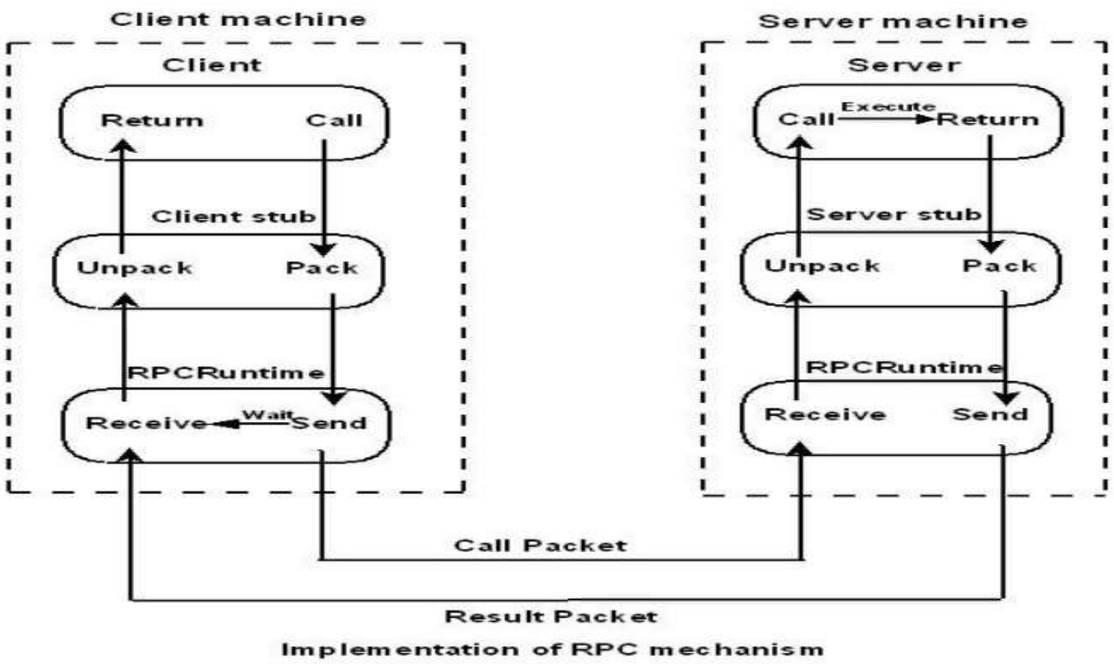
Server: The program accepting the request from the client on a system where the remote code is to be executed

Stub: Portion of code that convert addresses and parameters between Client and Server systems

Marshalling: Placing parameters and addresses in a message to send to the Server

Un-marshalling: Removes parameters and addresses from a message when received by the Server.

Working of RPC's:



Implementation of RPC mechanism

Step 1: A client invokes a client stub procedure, passing parameters in the usual way. The client stub resides within the client's own address space.

Step 2: The client stub marshalling the parameters into a message. Marshalling includes converting the representation of the parameters into a standard format, and copying each parameter into the message.

Step 3: The client stub passes the message to local OS. Then local OS pass this message to the transport layer, which sends it to the remote server machine.

Step 4: On the server, the transport layer passes the message to a server stub, which de-marshalling the parameters and calls the desired server routine.

Step 5: When the server procedure completes, it returns to the server stub. Which marshals the return values into a message? The server stub then hands the message to the transport layer.

Step 6: The transport layer sends the result message back to the client transport layer, which hands the message back to the client stub.

Step 7: The client stub de-marshalling the return parameters and execution returns to the caller.

Problems with RPC:

- Global variables are not shared in remote procedure calls because both client stub and server stub are in different address spaces.
- Passing the pointer as an argument was not allowed in RPC because both client stub and server stub are in different address spaces.
- Most popular languages like C, C++, java and etc…. are dose not provide any RPC mechanisms. If we want to use those languages for RPC we need to design special compilers for both client and server stubs.

NOTE: These problems are not meant to suggest that RPC is hopeless. In fact, it is widely used, but some restrictions are needed to make it work well in practice.