

CS 3591

**COMPUTER NETWORKS****COURSE OBJECTIVES:**

- To understand the concept of layering in networks.
- To know the functions of protocols of each layer of TCP/IP protocol suite.
- To visualize the end-to-end flow of information.
- To learn the functions of network layer and the various routing protocols
- To familiarize the functions and protocols of the Transport layer

**UNIT I INTRODUCTION AND APPLICATION LAYER** 10

Data Communication - Networks – Network Types – Protocol Layering – TCP/IP Protocol suite – OSI Model – Introduction to Sockets - Application Layer protocols: HTTP – FTP – Email protocols (SMTP - POP3 - IMAP - MIME) – DNS – SNMP

**UNIT II TRANSPORT LAYER** 9

Introduction - Transport-Layer Protocols: UDP – TCP: Connection Management – Flow control - Congestion Control - Congestion avoidance (DECbit, RED) – SCTP – Quality of Service

**UNIT III NETWORK LAYER** 7

Switching: Packet Switching - Internet protocol - IPV4 – IP Addressing – Subnetting - IPV6, ARP, RARP, ICMP, DHCP

**UNIT IV ROUTING** 7

Routing and protocols: Unicast routing - Distance Vector Routing - RIP - Link State Routing – OSPF – Path-vector routing - BGP - Multicast Routing: DVMRP – PIM.

**UNIT V DATA LINK AND PHYSICAL LAYERS** 12

Data Link Layer – Framing – Flow control – Error control – Data-Link Layer Protocols – HDLC – PPP - Media Access Control – Ethernet Basics – CSMA/CD – Virtual LAN – Wireless LAN (802.11) - Physical Layer: Data and Signals - Performance – Transmission media- Switching – Circuit Switching.

**TOTAL:45 PERIODS****TEXT BOOKS**

1. James F. Kurose, Keith W. Ross, Computer Networking, A Top-Down Approach Featuring the Internet, Eighth Edition, Pearson Education, 2021.
2. Behrouz A. Forouzan, Data Communications and Networking with TCP/IP Protocol Suite, Sixth Edition TMH, 2022

[www.EnggTree.com](http://www.EnggTree.com)

CS 3591 COMPUTER NETWORKS

## UNIT I - INTRODUCTION AND PHYSICAL LAYER

Data Communication - Networks – Network Types – Protocol Layering – TCP/IP Protocol suite – OSI Model – Introduction to Sockets - Application Layer protocols: HTTP – FTP – Email protocols (SMTP - POP3 - IMAP - MIME) – DNS – SNMP

### INTRODUCTION TO NETWORKS

- A network is a set of devices (often referred to as nodes) connected by communication links.
- A node can be a computer, printer, or any other device capable of sending or receiving data generated by other nodes on the network.
- When we communicate, we are sharing information. This sharing can be local or remote.

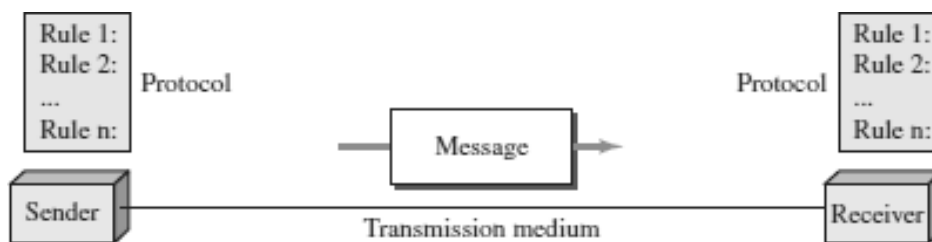
### CHARACTERISTICS OF A NETWORK

The effectiveness of a network depends on three characteristics.

1. **Delivery:** The system must deliver data to the correct destination.
2. **Accuracy:** The system must deliver data accurately.
3. **Timeliness:** The system must deliver data in a timely manner.

www.EnggTree.com

### COMPONENTS INVOLVED IN A NETWORK PROCESS



The five components are:

1. **Message** - It is the information to be communicated. Popular forms of information include text, pictures, audio, video 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.
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.

**KEY ELEMENTS OF PROTOCOL**

- **Syntax:** Refers to the structure or format of the data, meaning the order in which they are presented.
- **Semantics:** Refers to the meaning of each section of bits.
- **Timing:** Refers to two characteristics. (1). When data should be sent and (2). How fast they can be sent.

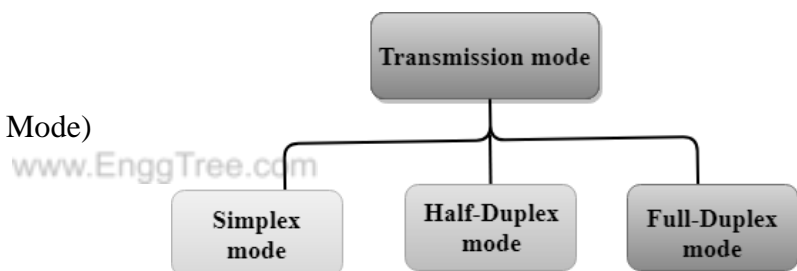
**TRANSMISSION MODES**

- The way in which data is transmitted from one device to another device is known as **transmission mode**.
- The transmission mode is also known as the communication mode.
- Each communication channel has a direction associated with it, and transmission media provide the direction. Therefore, the transmission mode is also known as a directional mode.
- The transmission mode is defined in the physical layer.

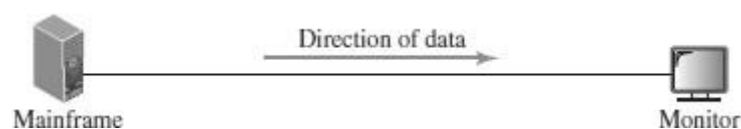
**Types of Transmission mode**

The Transmission mode is divided into three categories:

- Simplex Mode
- Half-duplex Mode
- Full-duplex mode (Duplex Mode)

**SIMPLEX MODE**

- In Simplex mode, the communication is unidirectional, i.e., the data flow in one direction.
- A device can only send the data but cannot receive it or it can receive the data but cannot send the data.
- This transmission mode is not very popular as mainly communications require the two-way exchange of data. The simplex mode is used in the business field as in sales that do not require any corresponding reply.
- The radio station is a simplex channel as it transmits the signal to the listeners but never allows them to transmit back.
- **Keyboard and Monitor** are the examples of the simplex mode as a keyboard can only accept the data from the user and monitor can only be used to display the data on the screen.
- The main advantage of the simplex mode is that the full capacity of the communication channel can be utilized during transmission.



***Advantage of Simplex mode:***

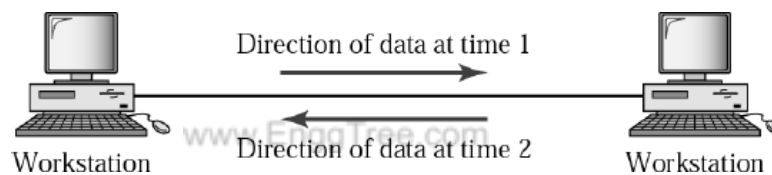
- In simplex mode, the station can utilize the entire bandwidth of the communication channel, so that more data can be transmitted at a time.

***Disadvantage of Simplex mode:***

- Communication is unidirectional, so it has no inter-communication between devices.

**HALF-DUPLEX MODE**

- In a Half-duplex channel, direction can be reversed, i.e., the station can transmit and receive the data as well.
- Messages flow in both the directions, but not at the same time.
- The entire bandwidth of the communication channel is utilized in one direction at a time.
- In half-duplex mode, it is possible to perform the error detection, and if any error occurs, then the receiver requests the sender to retransmit the data.
- A **Walkie-talkie** is an example of the Half-duplex mode.
- In Walkie-talkie, one party speaks, and another party listens. After a pause, the other speaks and first party listens. Speaking simultaneously will create the distorted sound which cannot be understood.

***Advantage of Half-duplex mode:***

- In half-duplex mode, both the devices can send and receive the data and also can utilize the entire bandwidth of the communication channel during the transmission of data.

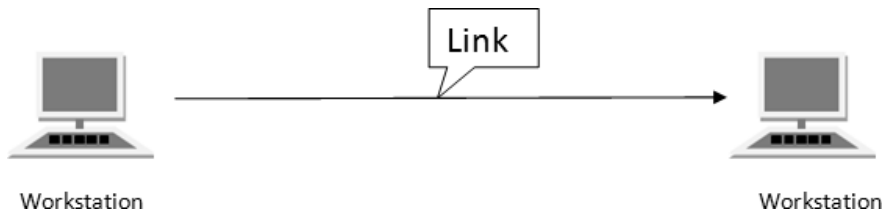
***Disadvantage of Half-Duplex mode:***

- In half-duplex mode, when one device is sending the data, then another has to wait, this causes the delay in sending the data at the right time.

**FULL-DUPLEX MODE**

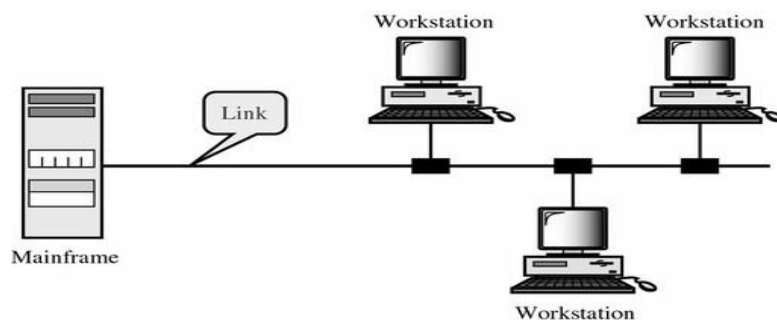
- In Full duplex mode, the communication is bi-directional, i.e., the data flow in both the directions.
- Both the stations can send and receive the message simultaneously.
- Full-duplex mode has two simplex channels. One channel has traffic moving in one direction, and another channel has traffic flowing in the opposite direction.
- The Full-duplex mode is the fastest mode of communication between devices.
- The most common example of the full-duplex mode is a **Telephone network**. When two people are communicating with each other by a telephone line, both can talk and listen at the same time.





ii. **MultiPoint:** It is also called **Multidrop** configuration. In this connection two or more devices share a single link. There are two kinds of Multipoint Connections.

- **Spatial Sharing:** If several devices can share the link simultaneously, it is called Spatially shared line configuration
- **Temporal (Time) Sharing:** If users must take turns using the link, then it's called Temporally shared or Time Shared Line Configuration.



## NETWORK TYPES

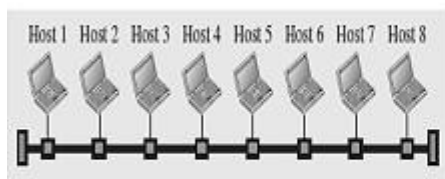
- A computer network is a group of computers linked to each other that enables the computer to communicate with another computer and share their resources, data, and applications.
- A computer network can be categorized by their size.
- A computer network is mainly of three types:
  1. Local Area Network (LAN)
  2. Wide Area Network (WAN)
  3. Metropolitan Area Network (MAN)

### **LOCAL AREA NETWORK (LAN)**

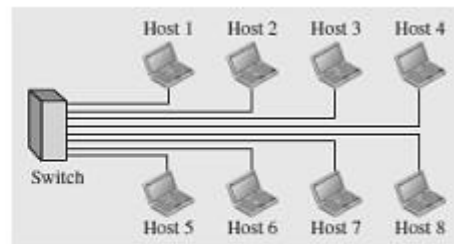
- Local Area Network is a group of computers connected to each other in a small area such as building, office.
- LAN is used for connecting two or more personal computers through a communication medium such as twisted pair, coaxial cable, etc.



- It is less costly as it is built with inexpensive hardware such as hubs, network adapters, and ethernet cables.
- The data is transferred at an extremely faster rate in Local Area Network.
- LAN can be connected using a common cable or a Switch.



a. LAN with a common cable (past)



b. LAN with a switch (today)

**Advantages of LAN**

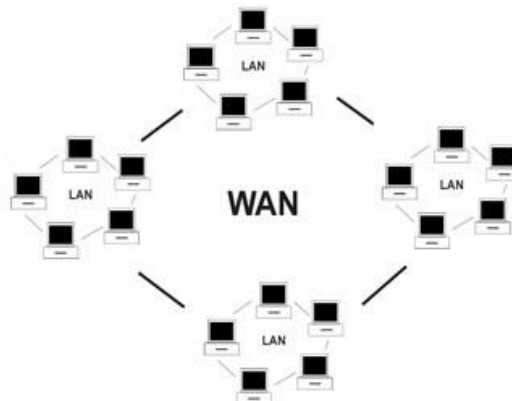
- Resource Sharing
- Software Applications Sharing.
- Easy and Cheap Communication
- Centralized Data.
- Data Security
- Internet Sharing

**Disadvantages of LAN**

- High Setup Cost
- Privacy Violations
- Data Security Threat
- LAN Maintenance Job
- Covers Limited Area

**WIDE AREA NETWORK (WAN)**

- A Wide Area Network is a network that extends over a large geographical areasuch as states or countries.
- A Wide Area Network is quite bigger network than the LAN.



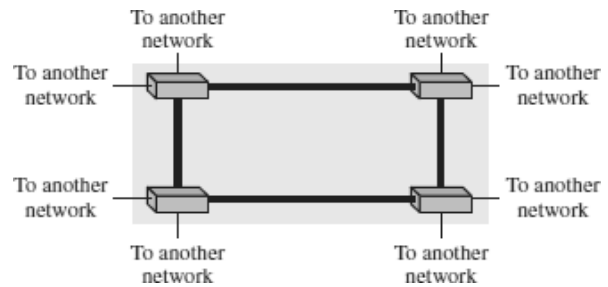


- A Wide Area Network is not limited to a single location, but it spans over a large geographical area through a telephone line, fiber optic cable or satellite links.
- The internet is one of the biggest WAN in the world.
- A Wide Area Network is widely used in the field of Business, government, and education.
- WAN can be either a point-to-point WAN or Switched WAN.

**Point-to-point WAN**



**Switched WAN**



**Advantages of Wide Area Network:**

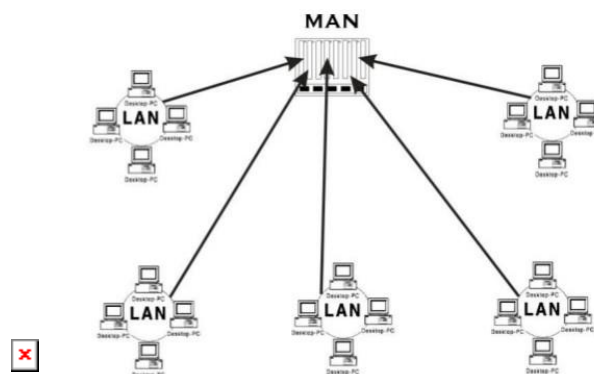
- Large Geographical area
- Centralized data
- Exchange messages
- Sharing of software and resources
- High bandwidth

**Disadvantages of Wide Area Network:**

- Security issue
- Needs Firewall & antivirus software
- High Setup cost
- Troubleshooting problems

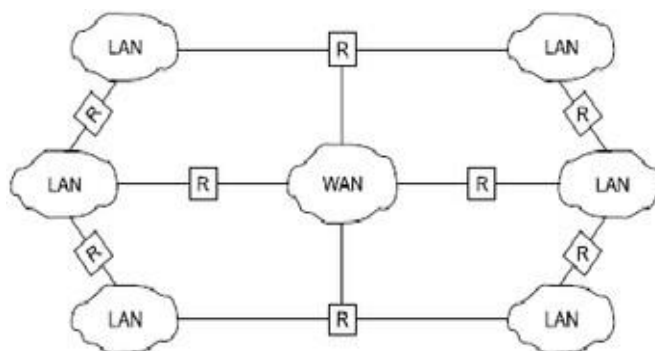
**METROPOLITAN AREA NETWORK (MAN)**

- A metropolitan area network is a network that covers a larger geographic area by interconnecting a different LAN to form a larger network.
- It generally covers towns and cities (50 km)
- In MAN, various LANs are connected to each other through a telephone exchange line.
- Communication medium used for MAN are optical fibers, cables etc.
- It has a higher range than Local Area Network (LAN). It is adequate for distributed computing applications.



## **INTERNETWORK**

- An internetwork is defined as two or more computer network LANs or WAN.
- An Internetwork can be formed by joining two or more individual networks by means of various devices such as routers, gateways and bridges.
- An interconnection between public, private, commercial, industrial, or government computer networks can also be defined as **internetworking**.



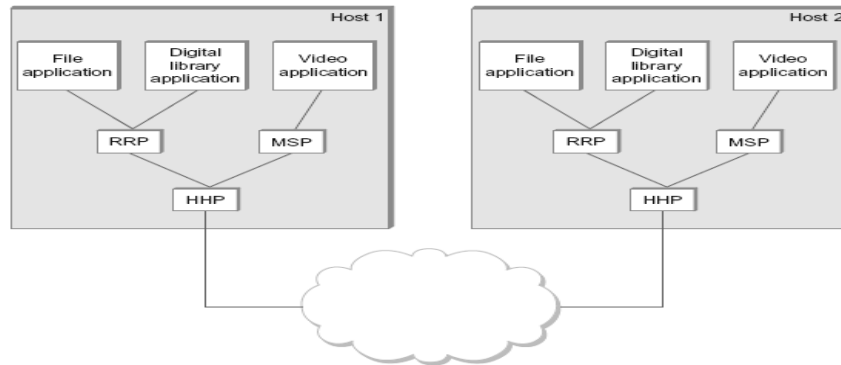
www.EnggTree.com

### **Types of Internetworks**

<u><b>Extranet</b></u>	<u><b>Intranet</b></u>
<p>An extranet is used for information sharing. The access to the extranet is restricted to only those users who have login credentials. An extranet is the lowest level of internetworking. It can be categorized as <b>MAN</b>, <b>WAN</b> or other computer networks. An extranet cannot have a single <b>LAN</b>, at least it must have one connection to the <b>external network</b>.</p>	<p>An intranet belongs to an organization which is only accessible by the <b>organization's employee</b> or members. The main aim of the intranet is to share the information and resources among the organization employees. An intranet provides the facility to work in groups and for teleconferences.</p>

## **PROTOCOL LAYERING**

- In networking, a protocol **defines the rules** that both the sender and receiver and all intermediate devices need to follow to be able **to communicate effectively**.
- A protocol provides a communication service that the process uses to exchange messages.
- When communication is simple, we may need only one simple protocol.



- When the communication is complex, we may need to divide the task between different layers, in which case we need a protocol at each layer, or **protocol layering**.
- Protocol layering is that it allows us to separate the services from the implementation.
- A layer needs to be able to receive a set of services from the lower layer and to give the services to the upper layer.
- Any modification in one layer will not affect the other layers.

### **Basic Elements of Layered Architecture**

- **Service:** It is a set of actions that a layer provides to the higher layer.
- **Protocol:** It defines a set of rules that a layer uses to exchange the information with peer entity. These rules mainly concern about both the contents and order of the messages used.
- **Interface:** It is a way through which the message is transferred from one layer to another layer.

### **Features of Protocol Layering**

1. It decomposes the problem of building a network into more manageable components.
2. It provides a more modular design.

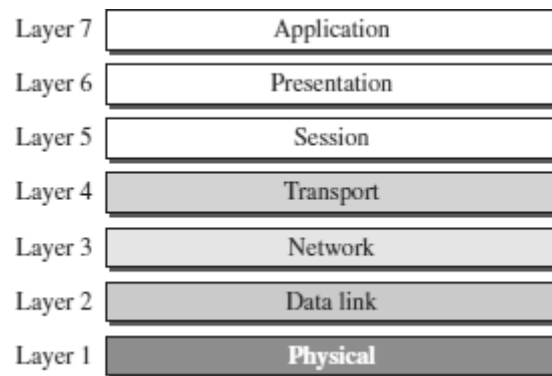
### **Principles of Protocol Layering**

1. The first principle dictates that if we want bidirectional communication, we need to make each layer so that it is able to perform two opposite tasks, one in each direction.
2. The second principle that we need to follow in protocol layering is that the two objects under each layer at both sites should be identical.

### **Protocol Graph**

- The set of protocols that make up a network system is called a **protocol graph**.
- The nodes of the graph correspond to protocols, and the edges represent a dependence relation.
- For example, the Figure below illustrates a protocol graph consists of protocols **RRP** (*Request/Reply Protocol*) and **MSP** (*Message Stream Protocol*) implement two different types of process-to-process channels, and both depend on the **HHP** (*Host-to-Host Protocol*) which provides a host-to-host connectivity service

- o OSI stands for **Open System Interconnection**.
- o It is a reference model that describes how information from a software application in one computer moves through a physical medium to the software application in another computer.
- o OSI consists of seven layers, and each layer performs a particular network function.
- o OSI model was developed by the International Organization for Standardization (ISO) in 1984, and it is now considered as an architectural model for the inter- computer communications.
- o OSI model divides the whole task into seven smaller and manageable tasks. Each layer is assigned a particular task.
- o Each layer is self-contained, so that task assigned to each layer can be performed independently.



**ORGANIZATION OF THE OSI LAYERS**

The OSI model is divided into two layers:  
**upper layers and lower layers.**

**UPPER LAYERS**

(Responsibility of the Host)

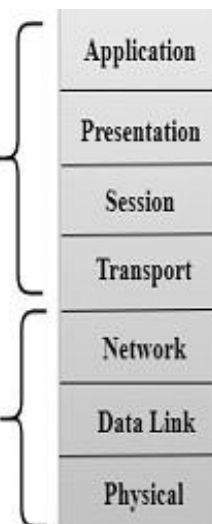
The upper layers of the OSI model mainly deals with the application related issues. They are implemented only in the software.

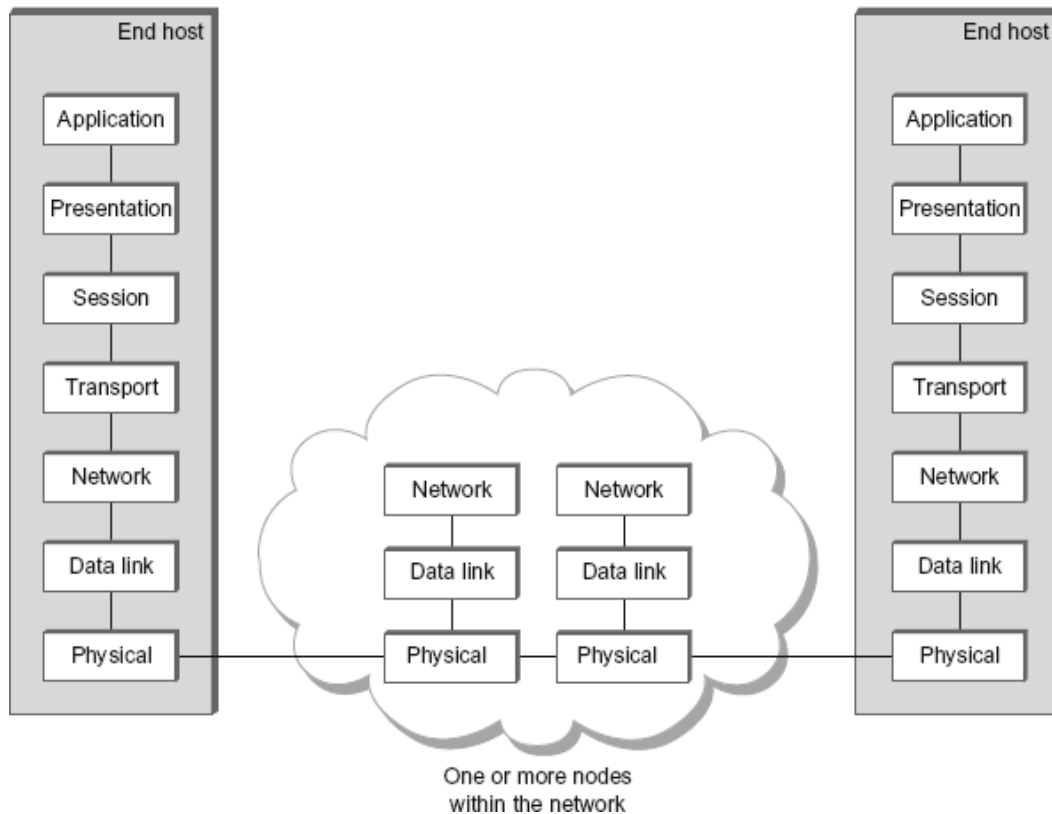


**LOWER LAYERS**

(Responsibility of the Network)

The lower layers of the OSI model deals with the data transport issues. They are implemented in hardware and software.





## FUNCTIONS OF THE OSI LAYERS

### 1. PHYSICAL LAYER

The physical layer coordinates the functions required to **transmit a bit stream over a physical medium**.

The physical layer is concerned with the following functions:

- Physical characteristics of interfaces and media** - The physical layer defines the characteristics of the interface between the devices and the transmission medium.
- Representation of bits** - To transmit the stream of bits, it must be encoded to signals. The physical layer defines the type of encoding.
- Signals:** It determines the type of the signal used for transmitting the information.
- Data Rate or Transmission rate** - The number of bits sent each second – is also defined by the physical layer.
- Synchronization of bits** - The sender and receiver must be synchronized at the bit level. Their clocks must be synchronized.
- Line Configuration** - In a point-to-point configuration, two devices are connected together through a dedicated link. In a multipoint configuration, a link is shared between several devices.
- Physical Topology** - The physical topology defines how devices are connected to make a network. Devices can be connected using a mesh, bus, star or ring topology.
- Transmission Mode** - The physical layer also defines the direction of transmission between two devices: simplex, half-duplex or full-duplex.

### 2. DATA LINK LAYER

It is responsible for **transmitting frames from one node to the next node**. The other responsibilities of this layer are

- **Framing** - Divides the stream of bits received into data units called frames.
- **Physical addressing** – If frames are to be distributed to different systems on the network, data link layer adds a header to the frame to define the sender and receiver.
- **Flow control**- If the rate at which the data are absorbed by the receiver is less than the rate produced in the sender, the Data link layer imposes a flow ctrl mechanism.
- **Error control**- Used for detecting and retransmitting damaged or lost frames and to prevent duplication of frames. This is achieved through a trailer added at the end of the frame.
- **Medium Access control** -Used to determine which device has control over the link at any given time.

### **3. NETWORK LAYER**

This layer is responsible for the **delivery of packets from source to destination**.

It determines the best path to move data from source to the destination based on the network conditions, the priority of service, and other factors.

The other responsibilities of this layer are

- **Logical addressing** - If a packet passes the network boundary, we need another addressing system for source and destination called logical address. This addressing is used to identify the device on the internet.
- **Routing** – Routing is the major component of the network layer, and it determines the best optimal path out of the multiple paths from source to the destination.

### **4. TRANSPORT LAYER**

www.EnggTree.com

It is responsible for **Process-to-Process** delivery. That is responsible for source-to- destination (end-to-end) delivery of the entire message, It also ensures whether the message arrives in order or not.

The other responsibilities of this layer are

- **Port addressing / Service Point addressing** - The header includes an address called port address / service point address. This layer gets the entire message to the correct process on that computer.
- **Segmentation and reassembly** - The message is divided into segments and each segment is assigned a sequence number. These numbers are arranged correctly on the arrival side by this layer.
- **Connection control** - This can either be **connectionless or connection oriented**.
  - The connectionless treats each segment as an individual packet and delivers to the destination.
  - The connection-oriented makes connection on the destination side before the delivery. After the delivery the termination will be terminated.
- **Flow control** - The transport layer also responsible for flow control but it is performed end-to-end rather than across a single link.
- **Error Control** - Error control is performed end-to-end rather than across the single link.

### **5. SESSION LAYER**

This layer **establishes, manages and terminates connections between applications.** The other responsibilities of this layer are

- **Dialog control** - Session layer acts as a dialog controller that creates a dialog between two processes or we can say that it allows the communication between two processes which can be either half-duplex or full-duplex.
- **Synchronization**- Session layer adds some checkpoints when transmitting the data in a sequence. If some error occurs in the middle of the transmission of data, then the transmission will take place again from the checkpoint. This process is known as Synchronization and recovery.

## **6. PRESENTATION LAYER**

It is concerned with the **syntax and semantics of information exchanged between two systems.** The other responsibilities of this layer are

- **Translation** – Different computers use different encoding system, this layer is responsible for interoperability between these different encoding methods. It will change the message into some common format.
- **Encryption and decryption**-It means that sender transforms the original information to another form and sends the resulting message over the n/w. and vice versa.
- **Compression and expansion**-Compression reduces the number of bits contained in the information particularly in text, audio and video.

## **7. APPLICATION LAYER**

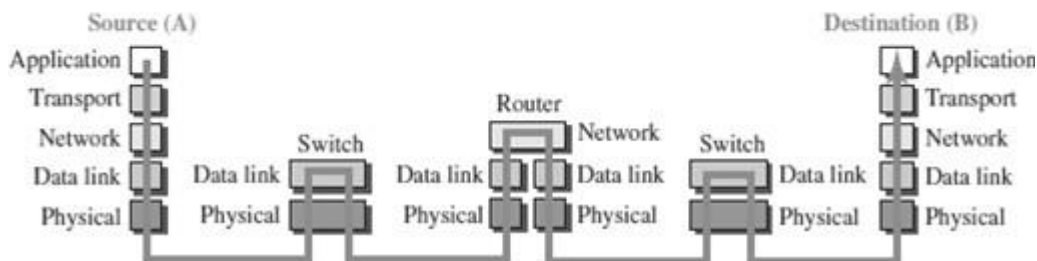
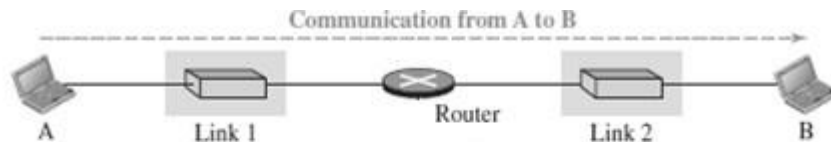
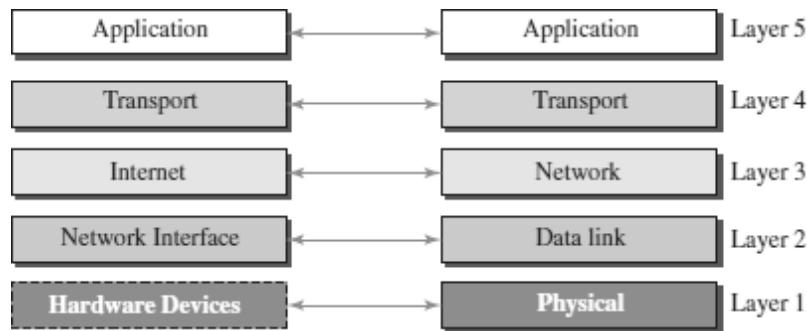
This layer **enables the user to access the network.** It handles issues such as network transparency, resource allocation, etc. This allows the user to log on to remote user.

The other responsibilities of this layer are

- **FTAM (File Transfer, Access, Management)** - Allows user to access files in a remote host.
- **Mail services** - Provides email forwarding and storage.
- **Directory services** - Provides database sources to access information about various sources and objects.

## TCP / IP PROTOCOL SUITE

- The TCP/IP architecture is also called as Internet architecture.
- It is developed by the US Defense Advanced Research Project Agency (**DARPA**) for its packet switched network (**ARPANET**).
- TCP/IP is a protocol suite used in the Internet today.
- It is a 4-layer model. The layers of TCP/IP are
  - 1. Application layer**
  - 2. Transport Layer (TCP/UDP)**
  - 3. Internet Layer**
  - 4. Network Interface Layer**



### APPLICATION LAYER

- An application layer incorporates the function of top three OSI layers. An application layer is the topmost layer in the TCP/IP model.
- It is responsible for handling high-level protocols, issues of representation.
- This layer allows the user to interact with the application.
- When one application layer protocol wants to communicate with another application layer, it forwards its data to the transport layer.
- Protocols such as FTP, HTTP, SMTP, POP3, etc running in the application layer provides service to other program running on top of application layer

### TRANSPORT LAYER

- The transport layer is responsible for the reliability, flow control, and correction of data which is being sent over the network.
- The two protocols used in the transport layer are **User Datagram protocol and Transmission control protocol.**
  - **UDP** – UDP provides connectionless service and end-to-end delivery of transmission. It is an unreliable protocol as it discovers the errors but not specify the error.
  - **TCP** – TCP provides a full transport layer services to applications. TCP is a reliable protocol as it detects the error and retransmits the damaged frames.

### INTERNET LAYER

- The internet layer is the second layer of the TCP/IP model.
- An internet layer is also known as the network layer.

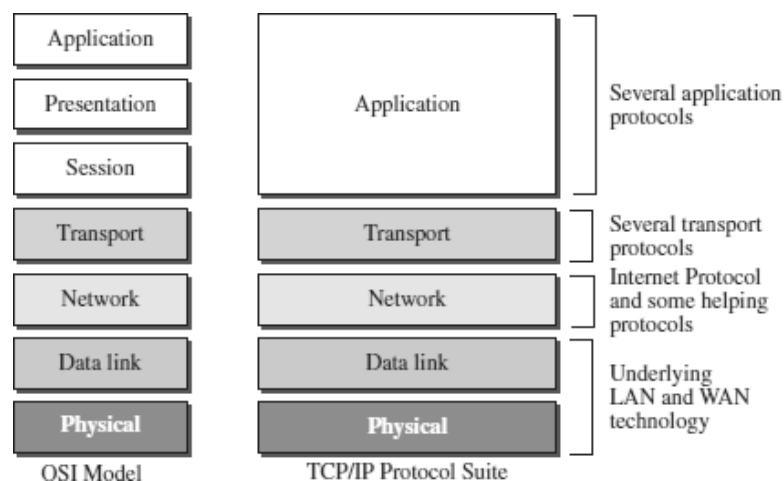


- The main responsibility of the internet layer is to send the packets from any network, and they arrive at the destination irrespective of the route they take.
- Internet layer handle the transfer of information across multiple networks through router and gateway .
- IP protocol is used in this layer, and it is the most significant part of the entire TCP/IP suite.

**NETWORK INTERFACE LAYER**

- The network interface layer is the lowest layer of the TCP/IP model.
- This layer is the combination of the Physical layer and Data Link layer defined in the OSI reference model.
- It defines how the data should be sent physically through the network.
- This layer is mainly responsible for the transmission of the data between two devices on the same network.
- The functions carried out by this layer are encapsulating the IP datagram into frames transmitted by the network and mapping of IP addresses into physical addresses.
- The protocols used by this layer are Ethernet, token ring, FDDI, X.25, frame relay.

**COMPARISON - OSI MODEL AND TCP/IP MODEL**



S.No	OSI MODEL	TCP/IP MODEL
1	Defined before advent of internet	Defined after the advent of Internet.
2	Service interface and protocols are clearly distinguished before	Service interface and protocols were not clearly distinguished before
3	Internetworking not supported	TCP/IP supports Internet working
4	Strict layering	Loosely layered
5	Protocol independent standard	Protocol Dependant standard
6	Less Credible	More Credible
7	All packets are reliably delivered	TCP reliably delivers packets, IP does not reliably deliver packets

### **SOCKETS**

A **socket** is one endpoint of a **two-way** communication link between two programs running on the network. The socket mechanism provides a means of inter-process communication (IPC) by establishing named contact points between which the communication take place.

Like 'Pipe' is used to create pipes and sockets is created using '**socket**' system call. The socket provides bidirectional **FIFO** Communication facility over the network. A socket connecting to the network is created at each end of the communication. Each socket has a specific address. This address is composed of an IP address and a port number.

Socket are generally employed in client server applications. The server creates a socket, attaches it to a network port address then waits for the client to contact it. The client creates a socket and then attempts to connect to the server socket. When the connection is established, transfer of data takes place.

**Types of Sockets:** There are two types of Sockets: the **datagram** socket and the **stream** socket.

1. **Datagram Socket:** This is a type of network which has connection less point for sending and receiving packets. It is similar to mailbox. The letters (data) posted into the box are collected and delivered (transmitted) to a letterbox (receiving socket).
2. **Stream Socket:** In Computer operating system, a stream socket is type of interprocess communications socket or network socket which provides a connection-oriented, sequenced, and unique flow of data without record boundaries with well-defined mechanisms for creating and destroying connections and for detecting errors. It is similar to phone. A connection is established between the phones (two ends) and a conversation (transfer of data) takes place.

### **HTTP (HYPERTEXT TRANSFER PROTOCOL)**

- The HyperText Transfer Protocol (HTTP) is used to define how the client-server

programs can be written to retrieve web pages from the Web.

- It is a protocol used to access the data on the World Wide Web (WWW).
  - The HTTP protocol can be used to transfer the data in the form of plain text, hypertext, audio, video, and so on.
  - HTTP is a *stateless* request/response protocol that governs client/server communication.
  - An HTTP client sends a request; an HTTP server returns a response.
  - The server uses the port number 80; the client uses a temporary port number.
  - HTTP uses the services of TCP, a connection-oriented and reliable protocol.
  - HTTP is a text-oriented protocol. It contains *embedded* URL known as links.
- 
- When hypertext is clicked, browser opens a new connection, retrieves file from the server and displays the file.
  - Each HTTP message has the general form
 

```
START_LINE <CRLF>
MESSAGE_HEADER <CRLF>
<CRLF> MESSAGE_BODY <CRLF>
```

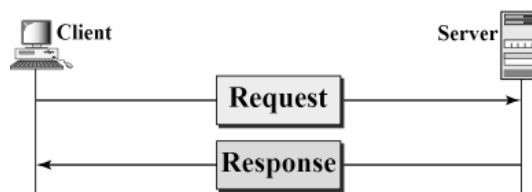
 where <CRLF> stands for carriage-return-line-feed.

### Features of HTTP

- **Connectionless protocol:**  
HTTP is a connectionless protocol. HTTP client initiates a request and waits for a response from the server. When the server receives the request, the server processes the request and sends back the response to the HTTP client after which the client disconnects the connection. The connection between client and server exist only during the current request and response time only.
- **Media independent:**  
HTTP protocol is a media independent as data can be sent as long as both the client and server know how to handle the data content. It is required for both the client and server to specify the content type in MIME-type header.
- **Stateless:**  
HTTP is a stateless protocol as both the client and server know each other only during the current request. Due to this nature of the protocol, both the client and server do not retain the information between various requests of the web pages.

### HTTP REQUEST AND RESPONSE MESSAGES

- The HTTP protocol defines the format of the request and response messages.



- Request Message:** The request message is sent by the client that consists of a request line, headers, and sometimes a body.
- Response Message:** The response message is sent by the server to the client that

consists of a status line, headers, and sometimes a body.

## HTTP REQUEST MESSAGE

<i>Request Line</i>
<i>Request Header : Value</i>
<i>Body (optional)</i>

- The first line in a request message is called a *request line*.
- After the request line, we can have zero or more *request header* lines.
- The *body* is an optional one. It contains the comment to be sent or the file to be published on the website when the method is PUT or POST.

### Request Line

- There are three fields in this request line - *Method*, *URL* and *Version*.
- The Method field defines the request types.
- The URL field defines the address and name of the corresponding web page.
- The Version field gives the version of the protocol; the most current version of HTTP is 1.1.
- Some of the Method types are

<i>Method</i>	<i>Action</i>
GET	Requests a document from the server
HEAD	Requests information about a document but not the document itself
PUT	Sends a document from the client to the server
POST	Sends some information from the client to the server
TRACE	Echoes the incoming request
DELETE	Removes the web page
CONNECT	Reserved
OPTIONS	Inquires about available options

### Request Header

- Each request header line sends additional information from the client to the server.
- Each header line has a header name, a colon, a space, and a header value.
- The value field defines the values associated with each header name.
- Headers defined for request message include

<i>Header</i>	<i>Description</i>
User-agent	Identifies the client program
Accept	Shows the media format the client can accept
Accept-charset	Shows the character set the client can handle
Accept-encoding	Shows the encoding scheme the client can handle
Accept-language	Shows the language the client can accept
Authorization	Shows what permissions the client has
Host	Shows the host and port number of the client
Date	Shows the current date
Upgrade	Specifies the preferred communication protocol
Cookie	Returns the cookie to the server
If-Modified-Since	If the file is modified since a specific date

**Body**

- The *body* can be present in a request message. It is optional.
- Usually, it contains the comment to be sent or the file to be published on the website when the method is PUT or POST.

**Conditional Request**

- A client can add a condition in its request.
- In this case, the server will send the requested web page if the condition is met or inform the client otherwise.
- One of the most common conditions imposed by the client is the time and date the web page is modified.
- The client can send the header line *If-Modified-Since* with the request to tell the server that it needs the page only if it is modified after a certain point in time.

**HTTP RESPONSE MESSAGE**

<i>Status Line</i>
<i>Response Header : Value</i>
<i>Body</i>

- The first line in a request message is called a *status line*.
- After the request line, we can have zero or more *response header* lines.
- The *body* is an optional one. The body is present unless the response is an error message

**Status Line**

- The Status line contains three fields - *HTTP version*, *Status code*, *Status phrase*
- The first field defines the version of HTTP protocol, currently 1.1.
- The status code field defines the status of the request. It classifies the HTTP result. It consists of three digits.
 

1xx–Informational,	2xx– Success,	3xx–Redirection,
4xx–Client error,	5xx–Server error	
- The Status phrase field gives brief description about status code in text form.

- Some of the Status codes are

Code	Phrase	Description
100	Continue	Initial request received, client to continue process
200	OK	Request is successful
301	Moved permanently	Requested URL is no longer in use
404	Not found	Document not found
500	Internal server error	An error such as a crash, at the server site

### **Response Header**

- Each header provides additional information to the client.
- Each header line has a header name, a colon, a space, and a header value.
- Some of the response headers are:

Response Header	Description
Content-type	specifies the MIME type
Expires	date and time up to which the document is valid
Last-modified	date and time when the document was last updated
Location	specifies location of the created or moved document

### **Body**

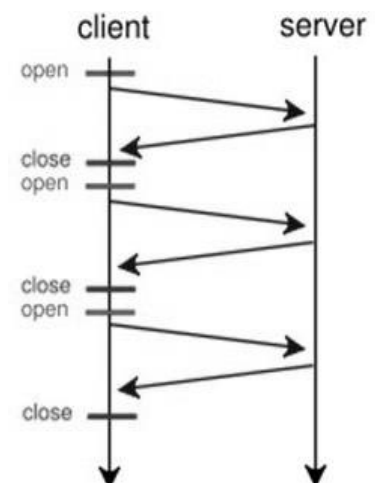
- The body contains the document to be sent from the server to the client.
- The body is present unless the response is an error message.

## **HTTP CONNECTIONS**

- HTTP Clients and Servers exchange multiple messages over the same TCP connection.
- If some of the objects are located on the same server, we have two choices: to retrieve each object using a new TCP connection or to make a TCP connection and retrieve them all.
- The first method is referred to as a *non-persistent connection*, the second as a *persistent connection*.
- HTTP 1.0 uses *non-persistent* connections and HTTP 1.1 uses *persistent* connections.

### **NON-PERSISTENT CONNECTIONS**

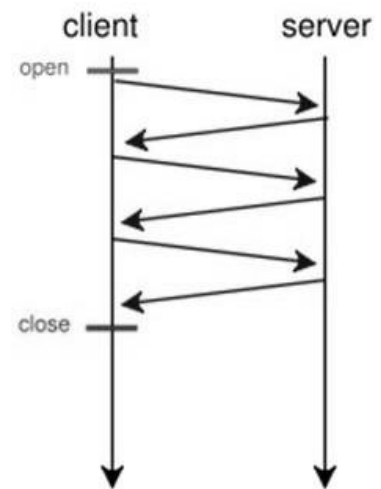
- In a non-persistent connection, one TCP connection is made for each request/response.
- Only one object can be sent over a single TCP connection
- The client opens a TCP connection and sends a request.
- The server sends the response and closes the connection.
- The client reads the data until it encounters an end-of-file marker.



- It then closes the connection.

## PERSISTENT CONNECTIONS

- HTTP version 1.1 specifies a persistent connection by default.
- Multiple objects can be sent over a single TCP connection.
- In a persistent connection, the server leaves the connection open for more requests after sending a response.
- The server can close the connection at the request of a client or if a time-out has been reached.
- Time and resources are saved using persistent connections. Only one set of buffers and variables needs to be set for the connection at each site.
- The round-trip time for connection establishment and connection termination is saved.



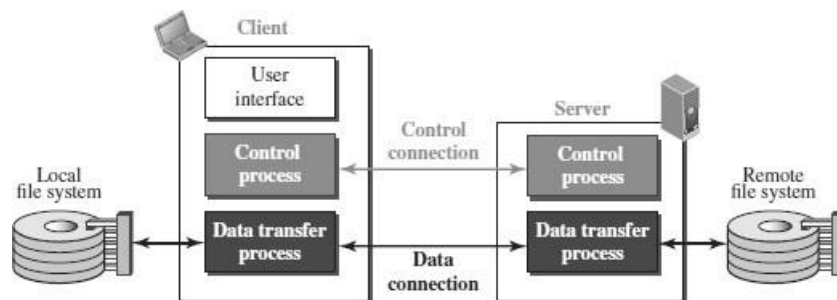
## FTP (FILE TRANSFER PROTOCOL)

- FTP stands for File transfer protocol.
- FTP is a standard internet protocol provided by TCP/IP used for transmitting the files from one host to another.
- It is mainly used for transferring the web page files from their creator to the computer that acts as a server for other computers on the internet.
- It is also used for downloading the files to computer from other servers.
- Although we can transfer files using HTTP, FTP is a better choice to transfer large files or to transfer files using different formats.

### FTP OBJECTIVES

- It provides the sharing of files.
- It is used to encourage the use of remote computers.
- It transfers the data more reliably and efficiently.

### FTP MECHANISM

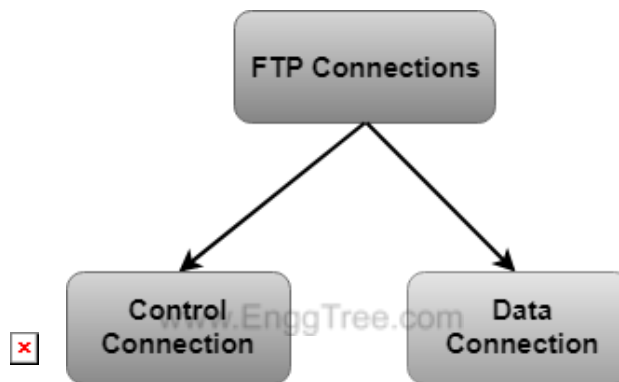


- The above figure shows the basic model of the FTP.

- The FTP client has three components:
  - user interface, control process, and data transfer process.
- The server has two components:
  - server control process and server data transfer process.

### **FTP CONNECTIONS**

- There are two types of connections in FTP -  
**Control Connection and Data Connection.**
- The two connections in FTP have different lifetimes.
- The control connection remains connected during the entire interactive FTP session.
- The data connection is opened and then closed for each file transfer activity. When a user starts an FTP session, the control connection opens.
- While the control connection is open, the data connection can be opened and closed multiple times if several files are transferred.
- FTP uses two well-known TCP ports:
  - Port 21 is used for the control connection
  - Port 20 is used for the data connection.



- **Control Connection:**
  - The control connection uses very simple rules for communication.
  - Through control connection, we can transfer a line of command or line of response at a time.
  - The control connection is made between the control processes.
  - The control connection remains connected during the entire interactive FTP session.
- **Data Connection:**
  - The Data Connection uses very complex rules as data types may vary.
  - The data connection is made between data transfer processes.
  - The data connection opens when a command comes for transferring the files and closes when the file is transferred.

### **FTP COMMUNICATION**

- FTP Communication is achieved through commands and responses.
- FTP Commands are sent from the client to the server
- FTP responses are sent from the server to the client.
- FTP Commands are in the form of ASCII uppercase, which may or may not be followed by an argument.



- Some of the most common commands are

<i>Command</i>	<i>Description</i>
<b>ABOR</b>	Abort the previous command
<b>CDUP</b>	Change to parent directory
<b>CWD</b>	Change to another directory
<b>DELE</b>	Delete a file
<b>LIST</b>	List subdirectories or files
<b>MKD</b>	Create a new directory
<b>PASS</b>	Password
<b>PASV</b>	Server chooses a port
<b>PORT</b>	Client chooses a port
<b>PWD</b>	Display name of current directory
<b>QUIT</b>	Log out of the system
<b>RETR</b>	Retrieve files; files are transferred from server to client
<b>RMD</b>	Delete a directory
<b>RNFR</b>	Identify a file to be renamed
<b>RNTO</b>	Rename the file
<b>STOR</b>	Store files; file(s) are transferred from client to server
<b>STRU</b>	Define data organization (F: file, R: record, or P: page)
<b>TYPE</b>	Default file type (A: ASCII, E: EBCDIC, I: image)
<b>USER</b>	User information
<b>MODE</b>	Define transmission mode (S: stream, B: block, or C: compressed)

- Every FTP command generates at least one response.
- A response has two parts: a three-digit number followed by text.
- The numeric part defines the code; the text part defines needed parameter.

<i>Code</i>	<i>Description</i>	<i>Code</i>	<i>Description</i>
<b>125</b>	Data connection open	<b>250</b>	Request file action OK
<b>150</b>	File status OK	<b>331</b>	User name OK; password is needed
<b>200</b>	Command OK	<b>425</b>	Cannot open data connection
<b>220</b>	Service ready	<b>450</b>	File action not taken; file not available
<b>221</b>	Service closing	<b>452</b>	Action aborted; insufficient storage
<b>225</b>	Data connection open	<b>500</b>	Syntax error; unrecognized command
<b>226</b>	Closing data connection	<b>501</b>	Syntax error in parameters or arguments
<b>230</b>	User login OK	<b>530</b>	User not logged in

### **FTP FILE TYPE**

- FTP can transfer one of the following file types across the data connection: ASCII file, EBCDIC file, or image file.

### **FTP DATA STRUCTURE**

- FTP can transfer a file across the data connection using one of the following data structure : *file structure*, *record structure*, or *page structure*.
- The file structure format is the default one and has no structure. It is a continuous stream of bytes.
- In the record structure, the file is divided into *records*. This can be used only with text files.
- In the page structure, the file is divided into pages, with each page having a page number and a page header. The pages can be stored and accessed randomly or sequentially.

### **FTP TRANSMISSION MODE**

- FTP can transfer a file across the data connection using one of the following three

transmission modes: *stream mode*, *block mode*, or *compressed mode*.

- The stream mode is the default mode; data are delivered from FTP to TCP as a continuous stream of bytes.
- In the block mode, data can be delivered from FTP to TCP in blocks.
- In the compressed mode, data can be compressed and delivered from FTP to TCP.

### **FTP FILE TRANSFER**

- File transfer occurs over the data connection under the control of the commands sent over the control connection.
- File transfer in FTP means one of three things:
  - retrieving a file (server to client)
  - storing a file (client to server)
  - directory listing (server to client).

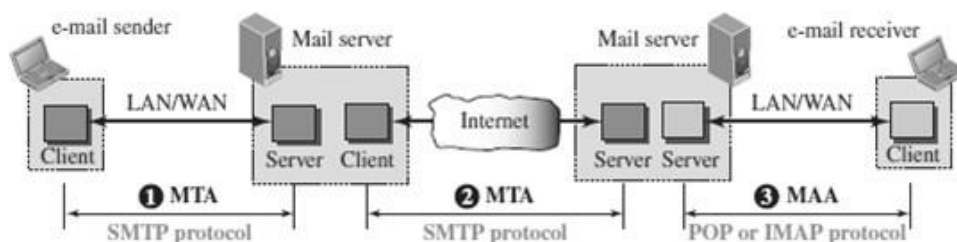
### **FTP SECURITY**

- FTP requires a password, the password is sent in plaintext which is unencrypted. This means it can be intercepted and used by an attacker.
- The data transfer connection also transfers data in plaintext, which is insecure.
- To be secure, one can add a Secure Socket Layer between the FTP application layer and the TCP layer.
- In this case FTP is called SSL-FTP.

---

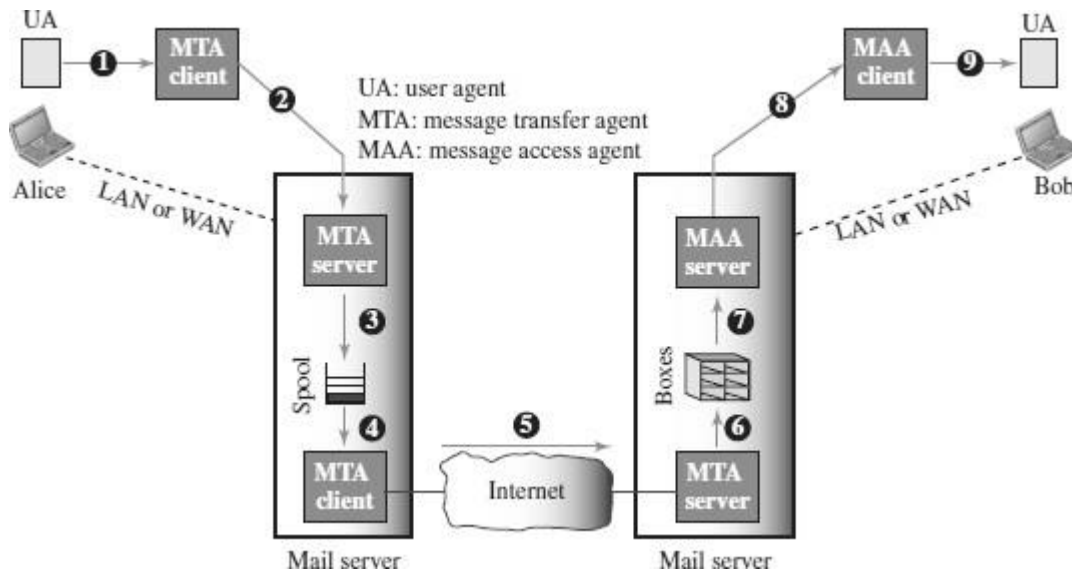
## **EMAIL (SMTP, MIME, IMAP, POP3)**

- One of the most popular Internet services is electronic mail (E-mail).
- Email is one of the oldest network applications.
- The **three main components of an Email** are
  1. User Agent (UA)
  2. Message Transfer Agent (MTA) – SMTP
  3. Message Access Agent (MAA) - IMAP ,POP



- When the sender and the receiver of an e-mail are on the same system, we need only two User Agents and no Message Transfer Agent
- When the sender and the receiver of an e-mail are on different system, we need two UA, two pairs of MTA (client and server), and two MAA (client and server).

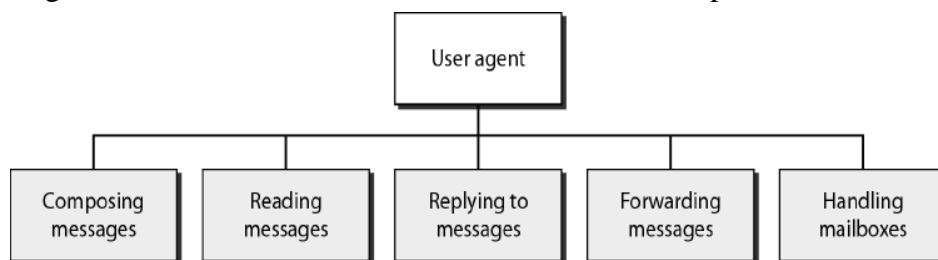
### **WORKING OF EMAIL**



- When Alice needs to send a message to Bob, she runs a UA program to prepare the message and send it to her mail server.
- The mail server at her site uses a queue (spool) to store messages waiting to be sent. The message, however, needs to be sent through the Internet from Alice's site to Bob's site using an MTA.
- Here two message transfer agents are needed: one client and one server.
- The server needs to run all the time because it does not know when a client will ask for a connection.
- The client can be triggered by the system when there is a message in the queue to be sent.
- The user agent at the Bob site allows Bob to read the received message.
- Bob later uses an MAA client to retrieve the message from an MAA server running on the second server.

### USER AGENT (UA)

- The first component of an electronic mail system is the user agent (UA).
- It provides service to the user to make the process of sending and receiving a message easier.
- A user agent is a software package that composes, reads, replies to, and forwards messages. It also handles local mailboxes on the user computers.



- There are two types of user agents: **Command-driven and GUI-based.**

### *Command driven*

- Command driven user agents belong to the early days of electronic mail.
- A command-driven user agent normally accepts a one-character command from the keyboard to perform its task.
- Some examples of command driven user agents are *mail*, *pine*, and *elm*.

### ***GUI-based***

- Modern user agents are GUI-based.
- They allow the user to interact with the software by using both the keyboard and the mouse.
- They have graphical components such as icons, menu bars, and windows that make the services easy to access.
- Some examples of GUI-based user agents are *Eudora* and *Outlook*.

### **MESSAGE TRANSFER AGENT (MTA)**

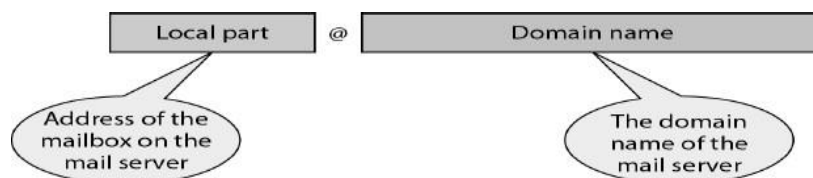
- The actual mail transfer is done through message transfer agents (MTA).
- To send mail, a system must have the client MTA, and to receive mail, a system must have a server MTA.
- The formal protocol that defines the MTA client and server in the Internet is called Simple Mail Transfer Protocol (SMTP).

### **MESSAGE ACCESS AGENT (MAA)**

- MAA is a software that pulls messages out of a mailbox.
- POP3 and IMAP4 are examples of MAA.

### **ADDRESS FORMAT OF EMAIL**

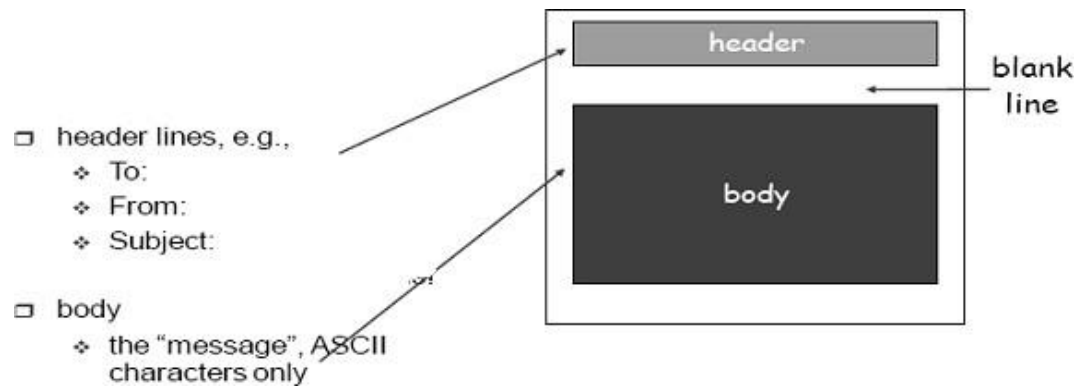
- E-mail address is *userid @ domain* where *domain* is hostname of the mail server.



### **MESSAGE FORMAT OF EMAIL**

- Email message consists of two parts namely *header* and *body*.
- Each header line contains *type* and *value* separated by a colon (:).
- Some header contents are:
  - **From:** identifier sender of the message.
  - **To:** mail address of the recipient(s).
  - **Subject:** says about purpose of the message.
  - **Date:** timestamp of when the message was transmitted.
- Header is separated from the body by a *blank* line.

- Body contains the *actual* message.

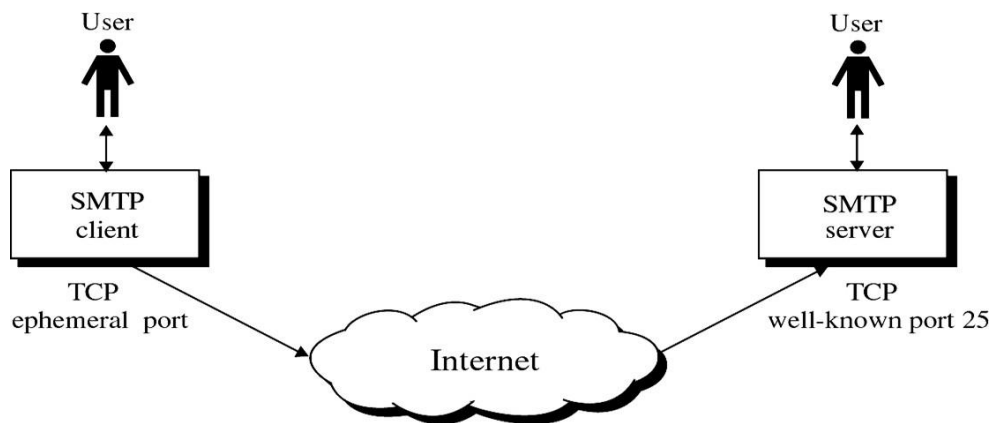


- Email was extended in 1993 to carry many different types of data: audio, video, images, Word documents, and so on.
- This extended version is known as **MIME** (Multipurpose Mail Extension).

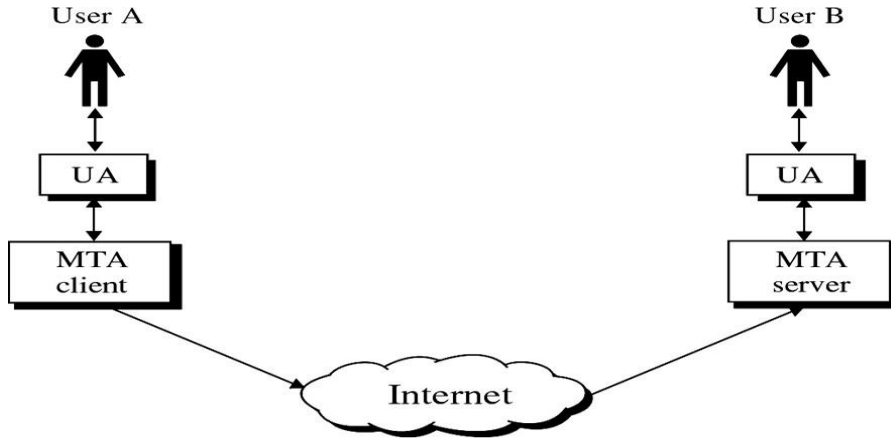
### SIMPLE MAIL TRANSFER PROTOCOL (SMTP)

- SMTP is the standard protocol for transferring mail between hosts in the TCP/IP protocol suite.
- SMTP is not concerned with the format or content of messages themselves.
- SMTP uses information written on the *envelope* of the mail (message header), but does not look at the *contents* (message body) of the envelope.

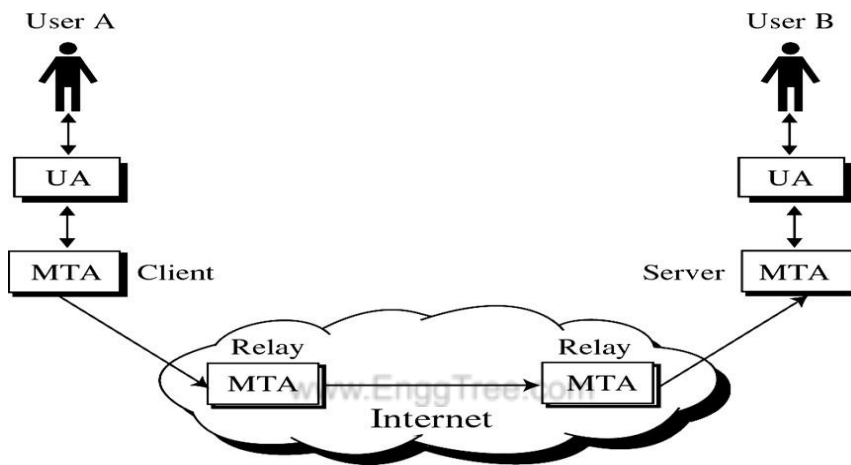
www.EnggTree.com



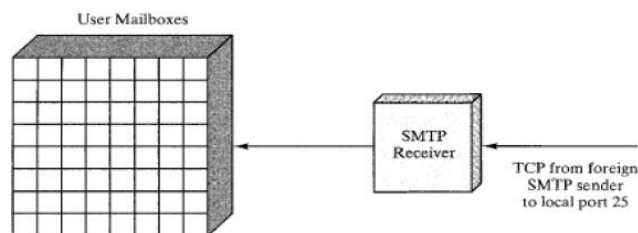
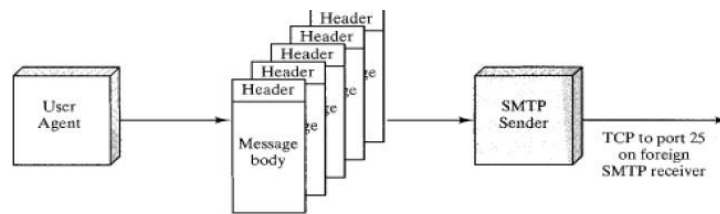
- SMTP clients and servers have two main components
  - **User Agents (UA)** – Prepares the message, encloses it in an envelope.
  - **Mail Transfer Agent (MTA)** – Transfers the mail across the internet



□ SMTP also allows the use of Relays allowing other MTAs to relay the mail.



### SMTP MAIL FLOW



- To begin, mail is created by a user-agent program in response to user input.
- Each created message consists of a header that includes the recipient's email address and other information, and a message body containing the message to be sent.
- These messages are then queued in some fashion and provided as input to an SMTP Sender program.

### SMTP COMMANDS AND RESPONSES

- The operation of SMTP consists of a series of commands and responses exchanged between the SMTP sender and SMTP receiver.
- The initiative is with the SMTP sender, who establishes the TCP connection.
- Once the connection is established, the SMTP sender sends commands over the connection to the receiver.
- The command is from an MTA client to an MTA server; the response is from an MTA server to the MTA client.

### SMTP Commands

- Commands are sent from the client to the server. It consists of a keyword followed by zero or more arguments. SMTP defines 14 commands.

*SMTP commands*

<i>Keyword</i>	<i>Argument(s)</i>	<i>Description</i>
HELO	Sender's host name	Identifies itself
MAIL FROM	Sender of the message	Identifies the sender of the message
RCPT TO	Intended recipient	Identifies the recipient of the message
DATA	Body of the mail	Sends the actual message
QUIT		Terminates the message
RSET		Aborts the current mail transaction
VERFY	Name of recipient	Verifies the address of the recipient
NOOP		Checks the status of the recipient
TURN		Switches the sender and the recipient
EXPN	Mailing list	Asks the recipient to expand the mailing list
HELP	Command name	Asks the recipient to send information about the command sent as the argument
SEND FROM	Intended recipient	Specifies that the mail be delivered only to the terminal of the recipient, and not to the mailbox
SMOL FROM	Intended recipient	Specifies that the mail be delivered to the terminal <i>or</i> the mailbox of the recipient
SMAL FROM	Intended recipient	Specifies that the mail be delivered to the terminal <i>and</i> the mailbox of the recipient

### SMTP Responses

- Responses are sent from the server to the client.

- A response is a three-digit code that may be followed by additional textual information.

*SMTP Responses*

<i>Code</i>	<i>Description</i>
<b>Positive Completion Reply</b>	
211	System status or help reply
214	Help message
220	Service ready
221	Service closing transmission channel
250	Request command completed
251	User not local; the message will be forwarded
<b>Positive Intermediate Reply</b>	
354	Start mail input
<b>Transient Negative Completion Reply</b>	
421	Service not available
450	Mailbox not available
451	Command aborted: local error
452	Command aborted; insufficient storage
<b>Permanent Negative Completion Reply</b>	
500	Syntax error; unrecognized command
501	Syntax error in parameters or arguments
502	Command not implemented
503	Bad sequence of commands
504	Command temporarily not implemented
550	Command is not executed; mailbox unavailable
551	User not local
552	Requested action aborted; exceeded storage location
553	Requested action not taken; mailbox name not allowed
554	Transaction failed

www.EnggTree.com

## SMTP OPERATIONS

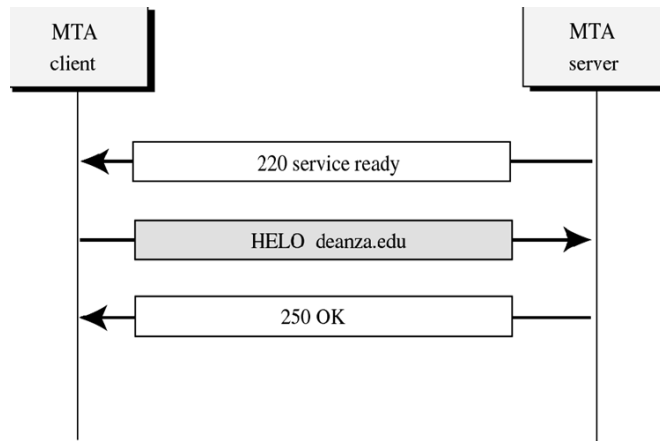
Basic SMTP operation occurs in three phases:

1. Connection Setup
2. Mail Transfer
3. Connection Termination

### Connection Setup

- An SMTP sender will attempt to set up a TCP connection with a target host when it has one or more mail messages to deliver to that host.
- The sequence is quite simple:
  1. The sender opens a TCP connection with the receiver.
  2. Once the connection is established, the receiver identifies itself with "Service Ready".
  3. The sender identifies itself with the HELO command.
  4. The receiver accepts the sender's identification with "OK".
  5. If the mail service on the destination is unavailable, the destination host returns a "Service Not Available" reply in step 2, and the process is terminated.





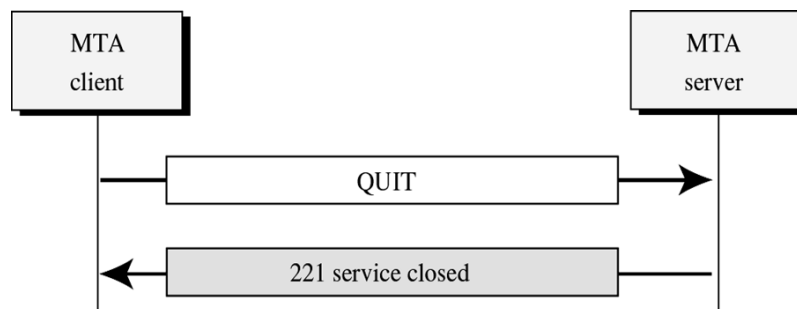
### Mail Transfer

- Once a connection has been established, the SMTP sender may send one or more messages to the SMTP receiver.
- There are three logical phases to the transfer of a message:
  1. A MAIL command identifies the originator of the message.
  2. One or more RCPT commands identify the recipients for this message.
  3. A DATA command transfers the message text.

### Connection Termination

www.EnggTree.com

- The SMTP sender closes the connection in two steps.
- First, the sender sends a QUIT command and waits for a reply.
- The second step is to initiate a TCP close operation for the TCP connection.
- The receiver initiates its TCP close after sending its reply to the QUIT command.



### LIMITATIONS OF SMTP

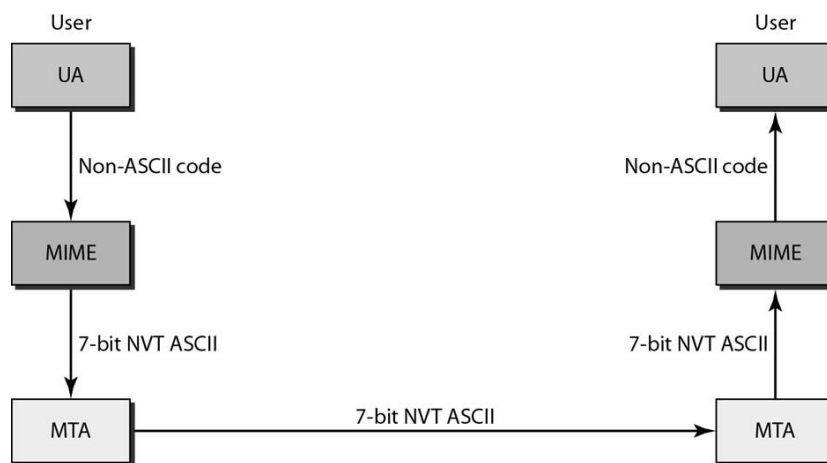
- SMTP cannot transmit executable files or other binary objects.
- SMTP cannot transmit text data that includes national language characters, as these are represented by 8-bit codes with values of 128 decimal or higher, and SMTP is limited to 7-bit ASCII.
- SMTP servers may reject mail message over a certain size.
- SMTP gateways that translate between ASCII and the character code EBCDIC do not use a consistent set of mappings, resulting in translation problems.
- Some SMTP implementations do not adhere completely to the SMTP standards defined.
- Common problems include the following:

1. Deletion, addition, or recording of carriage return and linefeed.
2. Truncating or wrapping lines longer than 76 characters.
3. Removal of trailing white space (tab and space characters).
4. Padding of lines in a message to the same length.

Conversion of tab characters into multiple-space characters

## MULTIPURPOSE INTERNET MAIL EXTENSION (MIME)

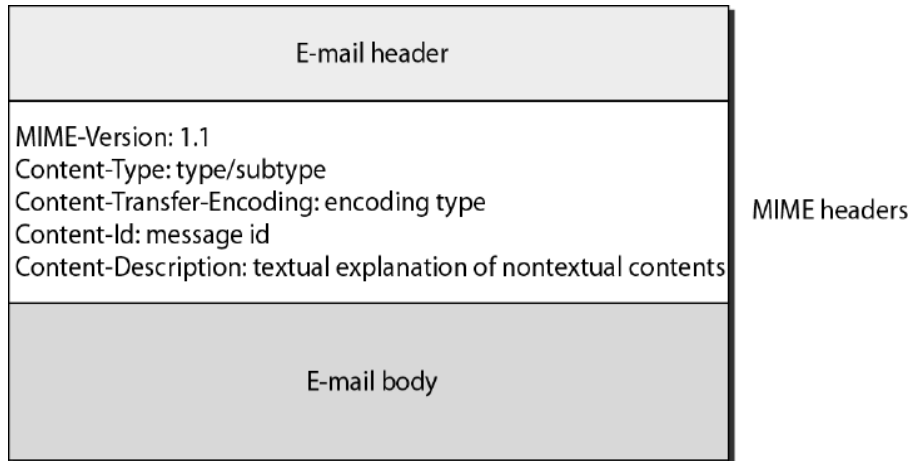
- SMTP provides a basic email service, while MIME adds multimedia capability to SMTP.
- MIME is an extension to SMTP and is used to overcome the problems and limitations of SMTP.
- Email system was designed to send messages only in *ASCII* format.
  - Languages such as French, Chinese, etc., are not supported.
  - Image, audio and video files cannot be sent.
- MIME adds the following features to email service:
  - Be able to send multiple attachments with a single message;
  - Unlimited message length;
  - Use of character sets other than ASCII code;
  - Use of rich text (layouts, fonts, colors, etc)
  - Binary attachments (executables, images, audio or video files, etc.), which may be divided if needed.
- MIME is a protocol that *converts* non-ASCII data to 7-bit NVT (Network Virtual Terminal) ASCII and vice-versa.



## MIME HEADERS

- Using headers, MIME describes the type of message content and the encoding used.
- Headers* defined in MIME are:
  - MIME-Version- current version, i.e., 1.1
  - Content-Type - message type (text/html, image/jpeg, application/pdf)
  - Content-Transfer-Encoding - message encoding scheme (eg base64).
  - Content-Id - unique identifier for the message.

- Content-Description - describes type of the message body.

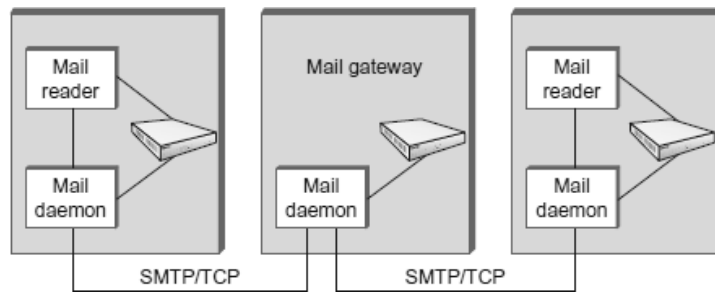


### MIME CONTENT TYPES

- There are seven different major types of content and a total of 14 subtypes.
- In general, a content type declares the general type of data, and the subtype specifies a particular format for that type of data.
- MIME also defines a multipart type that says how a message carrying more than one data type is structured.
- This is like a programming language that defines both base types (e.g., integers and floats) and compound types (e.g., structures and arrays).
- One possible multipart subtype is mixed, which says that the message contains a set of independent data pieces in a specified order.
- Each piece then has its own header line that describes the type of that piece.
- The table below lists the MIME content types:

<i>Type</i>	<i>Subtype</i>	<i>Description</i>
Text	Plain	Unformatted
	HTML	HTML format
Multipart	Mixed	Body contains ordered parts of different data types
	Parallel	Same as above, but no order
	Digest	Similar to mixed subtypes, but the default is message/RFC822
	Alternative	Parts are different versions of the same message
Message	RFC822	Body is an encapsulated message
	Partial	Body is a fragment of a bigger message
	External-Body	Body is a reference to another message
Image	JPEG	Image is in JPEG format
	GIF	Image is in GIF format
Video	MPEG	Video is in MPEG format
Audio	Basic	Single-channel encoding of voice at 8 kHz
Application	PostScript	Adobe PostScript
	Octet-stream	General binary data (8-bit bytes)

### MESSAGE TRANSFER IN MIME



- MTA is a mail daemon (sendmail) active on hosts having mailbox, used to send an email.
- Mail passes through a sequence of *gateways* before it reaches the recipient mailserv.
- Each gateway stores and forwards the mail using Simple mail transfer protocol (SMTP).
- SMTP defines communication between MTAs over TCP on port 25.
- In an SMTP session, sending MTA is *client* and receiver is *server*. In each exchange:
  - Client posts a command (HELO, MAIL, RCPT, DATA, QUIT, VRFY, etc.)
  - Server responds with a code (250, 550, 354, 221, 251 etc) and an explanation.
  - Client is identified using HELO command and verified by the server
  - Client forwards message to server, if server is willing to accept.
  - Message is terminated by a line with only single period (.) in it.
  - Eventually client terminates the connection.

### IMAP (INTERNET MAIL ACCESS PROTOCOL)

- IMAP is an Application Layer Internet protocol that allows an e-mail client to access e-mail on a remote mail server.
- It is a method of accessing electronic mail messages that are kept on a possibly shared mail server.
- IMAP is a more capable wire protocol.
- IMAP is similar to SMTP in many ways.
- IMAP is a client/server protocol running over TCP on port 143.
- IMAP allows multiple clients simultaneously connected to the same mailbox, and through flags stored on the server, different clients accessing the same mailbox at the same or different times can detect state changes made by other clients.
- In other words, it permits a "client" email program to access remote message stores as if they were local.
- For example, email stored on an IMAP server can be manipulated from a desktop computer at home, a workstation at the office, and a notebook computer while travelling, without the need to transfer messages or files back and forth between these computers.
- IMAP can support email serving in three modes:

- **Offline**

*This means that any information you put into the software isn't stored on your computer but instead on your internet connection.*

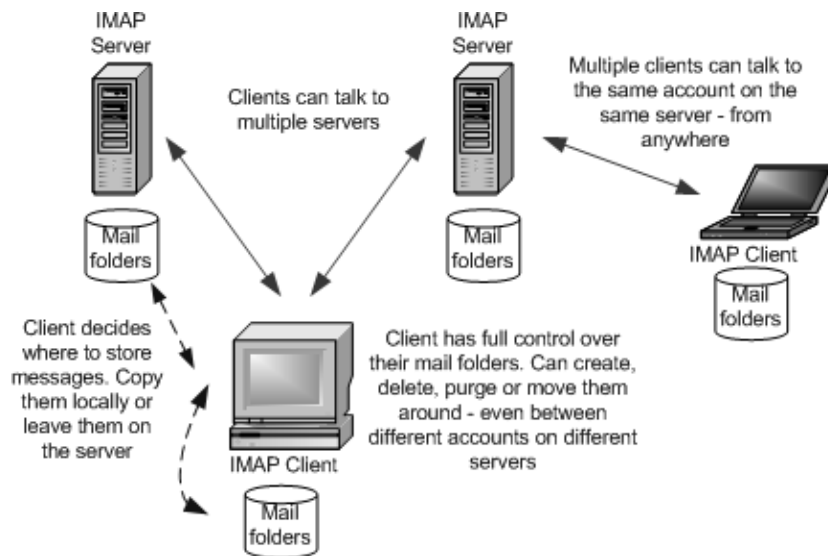
- **Online**

Users may connect to the server, look at what email is available, and access it online. This looks to the user very much like having local spool

files, but they're on the mail server.

- **Disconnected operation**

A mail client connects to the server, can make a "cache" copy of selected messages, and disconnects from the server. The user can then work on the messages offline, and connect to the server later and resynchronize the server status with the cache.



## OPERATION OF IMAP

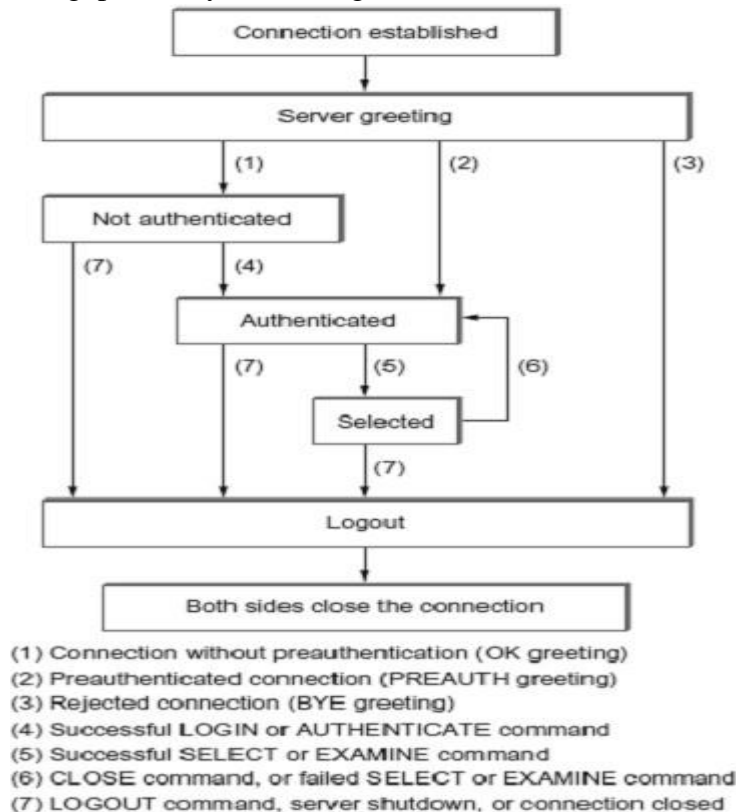
- The mail transfer begins with the client authenticating the user and identifying the mailbox they want to access.
- Client Commands**  
LOGIN, AUTHENTICATE, SELECT, EXAMINE, CLOSE, and LOGOUT
- Server Responses**  
OK, NO (no permission), BAD (incorrect command),
- When user wishes to FETCH a message, server responds in MIME format.
- Message *attributes* such as size are also exchanged.
- Flags* are used by client to report user actions.  
SEEN, ANSWERED, DELETED, RECENT

## IMAP4

- The latest version is IMAP4. IMAP4 is more powerful and more complex.
- IMAP4 provides the following extra functions:
  - A user can check the e-mail header prior to downloading.
  - A user can search the contents of the e-mail for a specific string of characters prior to downloading.
  - A user can partially download e-mail. This is especially useful if bandwidth is limited and the e-mail contains multimedia with high bandwidth requirements.
  - A user can create, delete, or rename mailboxes on the mail server.
  - A user can create a hierarchy of mailboxes in a folder for e-mail storage.

**ADVANTAGES OF IMAP**

- With IMAP, the primary storage is on the server, not on the local machine.
- Email being put away for storage can be foldered on local disk, or can be



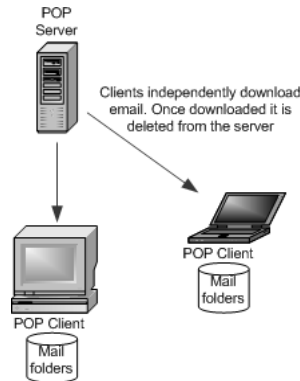
foldered on the IMAP server.

- The protocol allows full user of remote folders, including a remote folder hierarchy and multiple inboxes.
- It keeps track of explicit status of messages, and allows for user-defined status.
- Supports new mail notification explicitly.
- Extensible for non-email data, like netnews, document storage, etc.
- Selective fetching of individual MIME body parts.
- Server-based search to minimize data transfer.
- Servers may have extensions that can be negotiated.

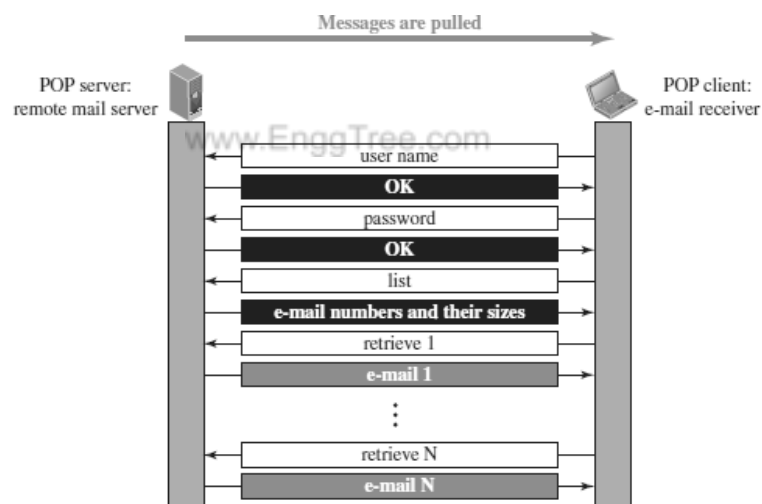
### POST OFFICE PROTOCOL (POP3)

- Post Office Protocol (POP3) is an application-layer Internet standard protocol used by local e-mail clients to retrieve e-mail from a remote server over a TCP/IP connection.
- There are two versions of POP.
  - The first, called *POP2*, became a standard in the mid-80's and requires SMTP to send messages.
  - The current version, POP3, can be used with or without SMTP. POP3 uses TCP/IP port 110.
- POP is a much simpler protocol, making implementation easier.
- POP supports offline access to the messages, thus requires less internet usagetime
- POP does not allow search facility.
- In order to access the messages, it is necessary to download them.
- It allows only one mailbox to be created on server.
- It is not suitable for accessing non mail data.
- POP mail moves the message from the email server onto the local computer, although there is usually an option to leave the messages on the email server as well.
- POP treats the mailbox as one store, and has no concept of folders.

- POP works in two modes namely, **delete** and **keep** mode.
  - In **delete mode**, mail is *deleted* from the mailbox after retrieval. The delete mode is normally used when the user is working at their permanent computer and can save and organize the received mail after reading or replying.
  - In **keep mode**, mail after reading is *kept* in mailbox for later retrieval. The keep mode is normally used when the user accesses her mail away from their primary computer.



- POP3 client is *installed* on the recipient computer and POP server on the mailserv.
- Client *opens* a connection to the server using TCP on port 110.
- Client sends username and password to *access* mailbox and to retrieve messages.



### POP3 Commands

POP commands are generally abbreviated into codes of three or four letters. The following describes some of the POP commands:

1. **UID** - This command opens the connection
2. **STAT** - It is used to display number of messages currently in the mailbox
3. **LIST** - It is used to get the summary of messages
4. **RETR** - This command helps to select a mailbox to access the messages
5. **DELE** - It is used to delete a message
6. **RSET** - It is used to reset the session to its initial state
7. **QUIT** - It is used to log off the session

---

### DIFFERENCE BETWEEN POP AND IMAP

---

SNo.	POP	IMAP
1	Generally used to support single client.	Designed to handle multiple clients.
2	Messages are accessed offline.	Messages are accessed online although it also supports offline mode.
3	POP does not allow search facility.	IMAP offers ability to search emails.
4	All the messages have to be downloaded.	It allows selective transfer of messages to the client.
5	Only one mailbox can be created on the server.	Multiple mailboxes can be created on the server.
6	Not suitable for accessing non-mail data.	Suitable for accessing non-mail data i.e. attachment.

7	POP commands are generally abbreviated into codes of three or four letters. Eg. STAT.	IMAP commands are not abbreviated, they are full. Eg. STATUS.
8	It requires minimum use of server resources.	Clients are totally dependent on server.
9	Mails once downloaded cannot be accessed from some other location.	Allows mails to be accessed from multiple locations.
10	The e-mails are not downloaded automatically.	Users can view the headings and sender of e-mails and then decide to download.
11	POP requires less internet usage time.	IMAP requires more internet usage time.

### Advantages of IMAP over POP

- IMAP is more powerful and more complex than POP.
- User can *check* the e-mail header prior to downloading.
- User can *search* e-mail for a specific string of characters prior to downloading.
- User can download *partially*, very useful in case of limited bandwidth.

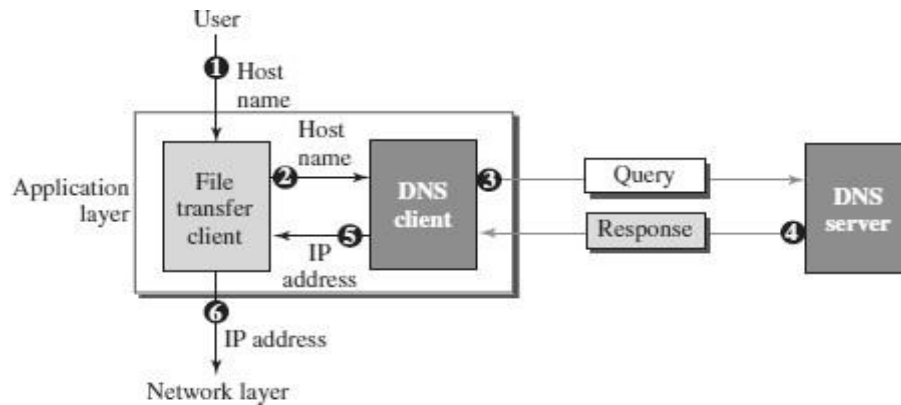
User can create, delete, or rename *mailboxes* on the mail server.

### DOMAIN NAME SYSTEM(DNS)

- Domain Name System was designed in 1984.
- DNS is used for name-to-address mapping.
- The DNS provides the protocol which allows clients and servers to communicate with each other.
- Eg: Host name like www.yahoo.com is translated into numerical IP addresses like 207.174.77.131
- Domain Name System (DNS) is a distributed database used by TCP/IP applications to map between hostnames and IP addresses and to provide electronic mail routing information.
- Each site maintains its own database of information and runs a server program that other systems across the Internet can query.



## WORKING OF DNS



The following six steps shows the working of a DNS. It maps the host name to an IP address:

1. The user passes the host name to the file transfer client.
2. The file transfer client passes the host name to the DNS client.
3. Each computer, after being booted, knows the address of one DNS server. The DNS client sends a message to a DNS server with a query that gives the file transfer server name using the known IP address of the DNS server.
4. The DNS server responds with the IP address of the desired file transfer server.
5. The DNS server passes the IP address to the file transfer client.
6. The file transfer client now uses the received IP address to access the file transfer server.

## NAME SPACE

- To be unambiguous, the names assigned to machines must be carefully selected from a name space with complete control over the binding between the names and IP address.
- The names must be unique because the addresses are unique.
- A name space that maps each address to a unique name can be organized into two ways: ***flat (or) hierarchical.***

### Flat Name Space

- In a flat name space, a name is assigned to an address.
- A name in this space is a sequence of characters without structure.
- The main disadvantage of a flat name space is that it cannot be used in a large system such as Internet because it must be centrally controlled to avoid ambiguity and duplication.

### Hierarchical Name Space

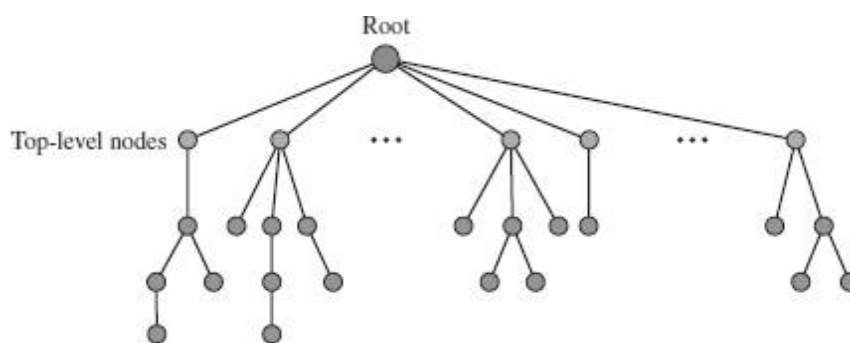
- In a hierarchical name space, each name is made of several parts.
- The first part can define the organization, the second part can define the name, the third part can define departments, and so on.
- In this case, the authority to assign and control the name spaces can be decentralized.
- A central authority can assign the part of the name that defines the nature of the organization and the name.
- The responsibility for the rest of the name can be given to the organization itself. Suffixes can be added to the name to define host or resources.
- The management of the organization need not worry that the prefix chosen for a host is taken by another organization because even if part of an address is the same, the

whole address is different.

- The names are unique without the need to be assigned by a central authority.
- The central authority controls only part of the name, not the whole name.

## DOMAIN NAME SPACE

- To have a hierarchical name space, a domain name space was designed. In this design, the names are defined in an inverted-tree structure with the root at the top.
- Each node in the tree has a label, which is a string with a maximum of 63 characters.
- The root label is a null string.
- DNS requires that children of a node have different labels, which guarantees the uniqueness of the domain names.



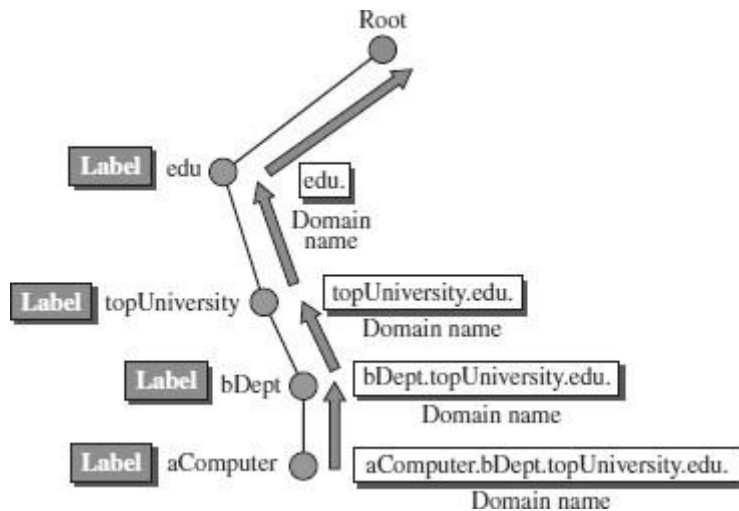
www.EnggTree.com

- Each node in the tree has a **label**, which is a string with a maximum of 63 characters.
- The root label is a null string (empty string). DNS requires that children of a node (nodes that branch from the same node) have different labels, which guarantees the uniqueness of the domain names.

## Domain Name

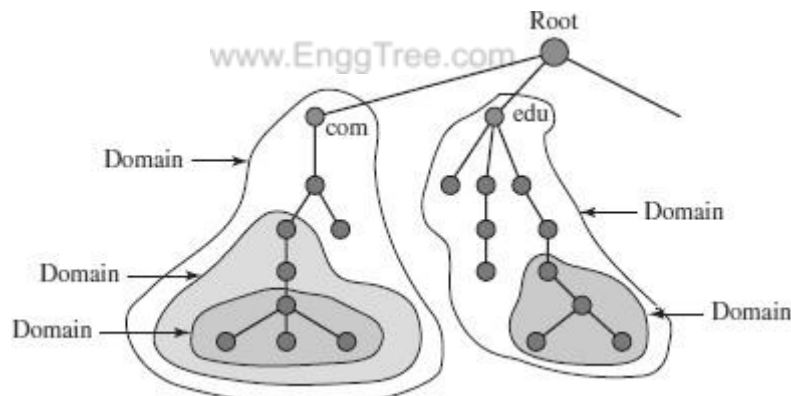
- Each node in the tree has a label called as domain name.
- A full domain name is a sequence of labels separated by dots (.)
- The domain names are always read from the node up to the root.
- The last label is the label of the root (null).
- This means that a full domain name always ends in a null label, which means the last character is a dot because the null string is nothing.
- If a label is terminated by a null string, it is called a **fully qualified domainname (FQDN)**.

- If a label is not terminated by a null string, it is called a *partially qualified domain name (PQDN)*.



## Domain

- A domain is a subtree of the domain name space.
- The name of the domain is the domain name of the node at the top of the sub-tree.
- A domain may itself be divided into domains.



## DISTRIBUTION OF NAME SPACE

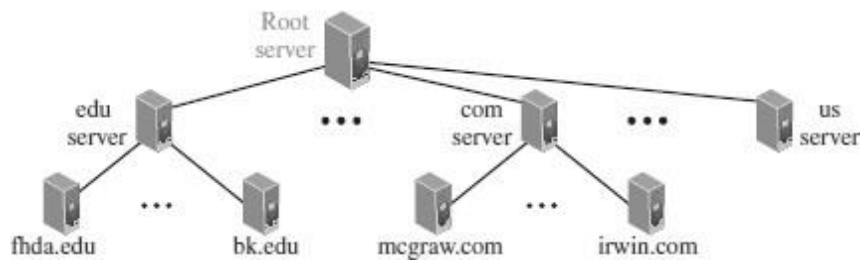
- The information contained in the domain name space must be stored.
- But it is very inefficient and also not reliable to have just one computer storesuch a huge amount of information.
- It is inefficient because responding to requests from all over the world, places aheavy load on the system.
- It is not reliable because any failure makes the data inaccessible.
- The solution to these problems is to distribute the information among many computers called *DNS servers*.

## HIERARCHY OF NAME SERVERS

- The way to distribute information among DNS servers is to divide the wholespace into many domains based on the first level.
- Let the root stand-alone and create as many domains as there are first level

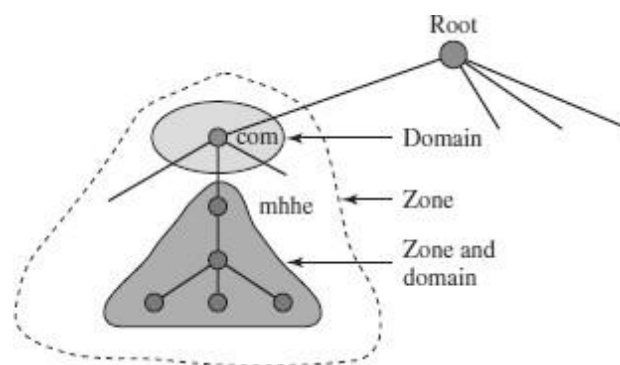
nodes.

- Because a domain created this way could be very large,
- DNS allows domains to be divided further into smaller domains.
- Thus, we have a hierarchy of servers in the same way that we have a hierarchy of names.



## ZONE

- What a server is responsible for, or has authority over, is called a *zone*.
- The server makes a database called a *zone* file and keeps all the information for every node under that domain.
- If a server accepts responsibility for a domain and does not divide the domains into smaller domains, the domain and zone refer to the same thing.
- But if a server divides its domain into sub domains and delegates parts of its authority to other servers, domain and zone refer to different things.
- The information about the nodes in the sub domains is stored in the servers at the lower levels, with the original server keeping some sort of references to these lower-level servers.
- But still, the original server does not free itself from responsibility totally.
- It still has a zone, but the detailed information is kept by the lower level servers.



## ROOT SERVER

- A root server is a server whose zone consists of the whole tree.
- A root server usually does not store any information about domains but delegates its authority to other servers, keeping references to those servers.
- Currently there are more than 13 root servers, each covering the whole domain name space.
- The servers are distributed all around the world.

## PRIMARY AND SECONDARY SERVERS

- DNS defines two types of servers: primary and secondary.
- A Primary Server is a server that stores a file about the zone for which it is an authority.

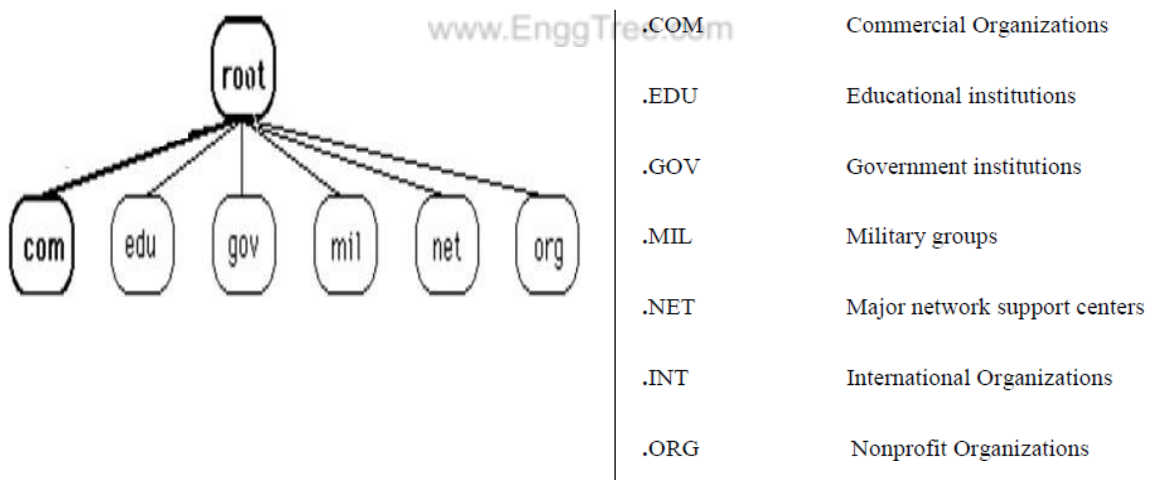
- Primary Servers are responsible for creating, maintaining, and updating the zone file.
- Primary Server stores the zone file on a local disc.
- A secondary server is a server that transfers the complete information about a zone from another server (Primary or Secondary) and stores the file on its local disc.
- If updating is required, it must be done by the primary server, which sends the updated version to the secondary.
  
- A primary server loads all information from the disk file; the secondary server loads all information from the primary server.

## **DNS IN THE INTERNET**

- DNS is a protocol that can be used in different platforms.
- In the Internet, the domain name space (tree) is divided into three different sections - *Generic domains, Country domains, and Inverse domain.*

### **Generic Domains**

- The generic domains define registered hosts according to their generic behavior.
- Each node in the tree defines a domain, which is an index to the domain namespace database.
- The first level in the generic domains section allows seven possible three character levels.
- These levels describe the organization types as listed in following table.



### **Country Domains**

- The country domains section follows the same format as the generic domains but uses two characters for country abbreviations
- E.g.; *in* for **India**, *us* for **United States** etc) in place of the three character organizational abbreviation at the first level.
- Second level labels can be organizational, or they can be more specific, national designation.
- India for example, uses state abbreviations as a subdivision of the country domain us. (e.g., ca.in.)

### **Inverse Domains**

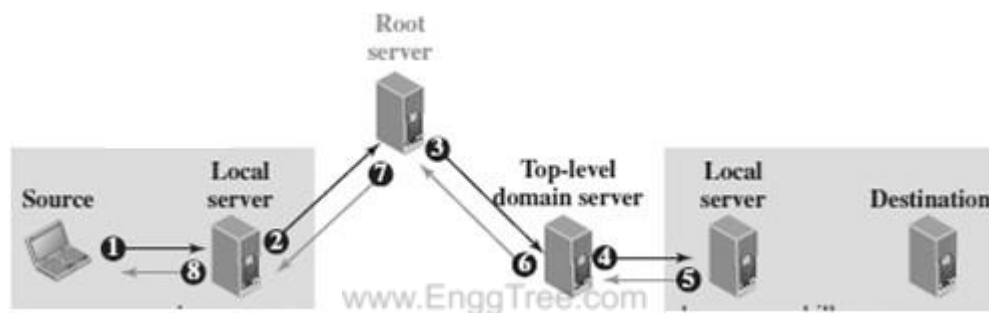
- Mapping an address to a name is called Inverse domain.

- The client can send an IP address to a server to be mapped to a domain name and it is called *PTR(Pointer) query*.
- To answer queries of this kind, DNS uses the inverse domain

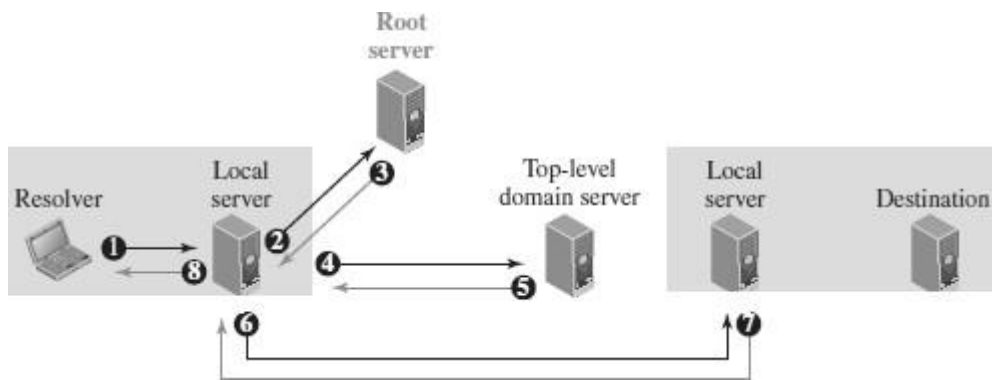
## DNS RESOLUTION

- Mapping a name to an address or an address to a name is called name address resolution.
- DNS is designed as a client server application.
- A host that needs to map an address to a name or a name to an address calls a DNS client named a *Resolver*.
- The Resolver accesses the closest DNS server with a mapping request.
- If the server has the information, it satisfies the resolver; otherwise, it either refers the resolver to other servers or asks other servers to provide the information.
- After the resolver receives the mapping, it interprets the response to see if it is areal resolution or an error and finally delivers the result to the process that requested it.
- A resolution can be either *recursive or iterative*.

### Recursive Resolution



- The application program on the source host calls the DNS resolver (client) to find the IP address of the destination host. The resolver, which does not know this address, sends the query to the local DNS server of the source (Event 1)
- The local server sends the query to a root DNS server (Event 2)
- The Root server sends the query to the top-level-DNS server (Event 3)
- The top-level DNS server knows only the IP address of the local DNS server at the destination. So, it forwards the query to the local server, which knows the IP address of the destination host (Event 4)
- The IP address of the destination host is now sent back to the top-level DNS server (Event 5) then back to the root server (Event 6), then back to the source DNS server, which may cache it for the future queries (Event 7), and finally back to the source host (Event 8).

**Iterative Resolution**

- In iterative resolution, each server that does not know the mapping, sends the IP address of the next server back to the one that requested it.
- The iterative resolution takes place between two local servers.
- The original resolver gets the final answer from the destination local server.
- The messages shown by Events 2, 4, and 6 contain the same query.
- However, the message shown by Event 3 contains the IP address of the top-level domain server.
- The message shown by Event 5 contains the IP address of the destination local DNS server.
- The message shown by Event 7 contains the IP address of the destination.
- When the Source local DNS server receives the IP address of the destination, it sends it to the resolver (Event 8).

**DNS CACHING**

- Each time a server receives a query for a name that is not in its domain, it needs to search its database for a server IP address.
- DNS handles this with a mechanism called **caching**.
- When a server asks for a mapping from another server and receives the response, it stores this information in its cache memory before sending it to the client.
- If the same or another client asks for the same mapping, it can check its cache memory and resolve the problem.
- However, to inform the client that the response is coming from the cache memory and not from an authoritative source, the server marks the response as **unauthoritative**.
- Caching speeds up resolution. Reduction of this search time would increase efficiency, but it can also be problematic.
- If a server caches a mapping for a long time, it may send an outdated mapping to the client.
- To counter this, two techniques are used.
  - ✓ First, the authoritative server always adds information to the mapping called **time to live (TTL)**. It defines the time in seconds that the receiving server can cache the information. After that time, the mapping is invalid and any query must be sent again to the authoritative server.

- ✓ Second, DNS requires that each server keep a **TTL counter** for each mapping it caches. The cache memory must be searched periodically and those mappings with an expired TTL must be purged.

### DNS RESOURCE RECORDS (RR)

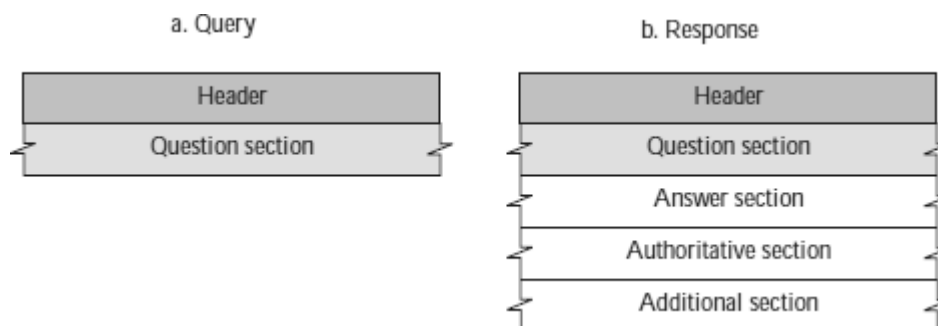
- The zone information associated with a server is implemented as a set of *resource records*.
- In other words, a name server stores a database of resource records.
- A *resource record* is a 5-tuple structure  
(**Domain Name, Type, Class, TTL, Value**)
- The domain name identifies the resource record.
- The type defines how the value should be interpreted.
- The value defines the information kept about the domain name.
- The TTL defines the number of seconds for which the information is valid.
- The class defines the type of network

### Types of Resource Records

Type	Interpretation of value
A	A 32-bit IPv4 address
NS	Identifies the authoritative servers for a zone
CNAME	Defines an alias for the official name of a host
SOA	Marks the beginning of a zone
MX	Redirects mail to a mail server
AAAA	An IPv6 address

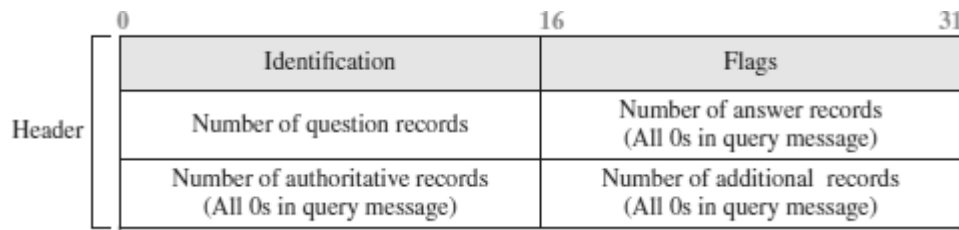
### DNS MESSAGES

- DNS has two types of messages: query and response.
- Both types have the same format.
- The query message consists of a header and question section.
- The response message consists of a header, question section, answer section, authoritative section, and additional section



- Header**
  - Both query and response messages have the same header format with some fields set to zero for the query messages.
  - The header fields are as follows:





- The identification field is used by the client to match the response with the query.
- The flag field defines whether the message is a query or response. It also includes status of error.
- The next four fields in the header define the number of each record type in the message.
- Question Section**
  - The question section consists of one or more question records. It is present in both query and response messages.
- Answer Section**
  - The answer section consists of one or more resource records. It is present only in response messages.
- Authoritative Section**
  - The authoritative section gives information (domain name) about one or more authoritative servers for the query.
- Additional Information Section**
  - The additional information section provides additional information that may help the resolver.

## DNS CONNECTIONS

- DNS can use either UDP or TCP.
- In both cases the well-known port used by the server is port 53.
- UDP is used when the size of the response message is less than 512 bytes because most UDP packages have a 512-byte packet size limit.
- If the size of the response message is more than 512 bytes, a TCP connection is used.

## DNS REGISTRARS

- New domains are added to DNS through a *registrar*. A fee is charged.
- A registrar first verifies that the requested domain name is unique and then enters it into the DNS database.
  - Today, there are many registrars; their names and addresses can be found at <http://www.intenic.net>
- To register, the organization needs to give the name of its server and the IP address of the server.
- For example, a new commercial organization named *wonderful* with a server named *ws* and IP address 200.200.200.5, needs to give the following information to one of the registrars:

**Domain name:** *ws.wonderful.com* **IP address:** *200.200.200.5*

## **SNMP (SIMPLE NETWORK MANAGEMENT PROTOCOL)**

- The **Simple Network Management Protocol (SNMP)** is a framework for managing devices in an internet using the TCP/IP protocol suite.
- SNMP is an application layer protocol that monitors and manages routers, distributed over a network.
- It provides a set of operations for monitoring and managing the internet.
- SNMP uses services of UDP on two well-known ports: 161 (Agent) and 162 (manager).
- SNMP uses the concept of *manager* and *agent*.



### **SNMP MANAGER**

- A manager is a host that runs the SNMP client program
- The manager has access to the values in the database kept by the agent.
- A manager checks the agent by requesting the information that reflects the behavior of the agent.
- A manager also forces the agent to perform a certain function by resetting values in the agent database.
- For example, a router can store in appropriate variables the number of packets received and forwarded.
- The manager can fetch and compare the values of these two variables to see if the router is congested or not.

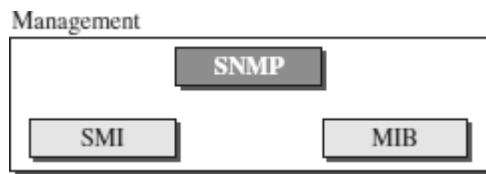
### **SNMP AGENT**

- The agent is a router that runs the SNMP server program.
- The agent is used to keep the information in a database while the manager is used to access the values in the database.
- For example, a router can store the appropriate variables such as a number of packets received and forwarded while the manager can compare these variables to determine whether the router is congested or not.
- Agents can also contribute to the management process.
- A server program on the agent checks the environment, if something goes wrong, the agent sends a warning message to the manager.

### **SNMP MANAGEMENT COMPONENTS**

- Management of the internet is achieved through simple interaction between a manager and agent.
- Management is achieved through the use of two protocols:

- Structure of Management Information (SMI)
- Management Information Base (MIB).



### Structure of Management Information (SMI)

- To use SNMP, we need rules for naming objects.
- SMI is a protocol that defines these rules.
- SMI is a guideline for SNMP
- It emphasizes three attributes to handle an object: name, data type, and encoding method.
- Its functions are:
  - ❖ To name objects.
  - ❖ To define the type of data that can be stored in an object.
  - ❖ To show how to encode data for transmission over the network.

### *Name*

- ✓ SMI requires that each managed object (such as a router, a variable in a router, a value, etc.) have a unique name. To name objects globally.
- ✓ SMI uses an **object identifier**, which is a hierarchical identifier based on a tree structure.
- ✓ The tree structure starts with an unnamed root. Each object can be defined using a sequence of integers separated by dots.
- ✓ The tree structure can also define an object using a sequence of textual names separated by dots.

### *Type of data*

- ✓ The second attribute of an object is the type of data stored in it.
- ✓ To define the data type, SMI uses **Abstract Syntax Notation One (ASN.1)** definitions.
- ✓ SMI has two broad categories of data types: *simple* and *structured*.
- ✓ The **simple data types** are atomic data types. Some of them are taken directly from ASN.1; some are added by SMI.
- ✓ SMI defines two **structured data types**: *sequence* and *sequence of*.
  - **Sequence** - A *sequence* data type is a combination of simple data types, not necessarily of the same type.
  - **Sequence of** - A *sequence of* data type is a combination of simple data types all of the same type or a combination of sequence data types all of the same type.

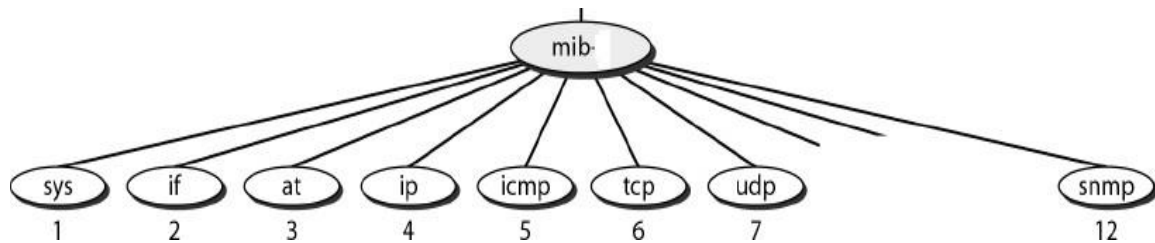
### *Encoding data*

- ✓ SMI uses another standard, **Basic Encoding Rules (BER)**, to encode data to be transmitted over the network.
- ✓ BER specifies that each piece of data be encoded in triplet format (TLV): tag, length, value

**Management Information Base (MIB)**

The Management Information Base (MIB) is the second component used in network management.

- Each agent has its own MIB, which is a *collection* of objects to be managed.
- MIB classifies objects under groups.

**MIB Variables**

MIB variables are of two types namely *simple* and *table*.

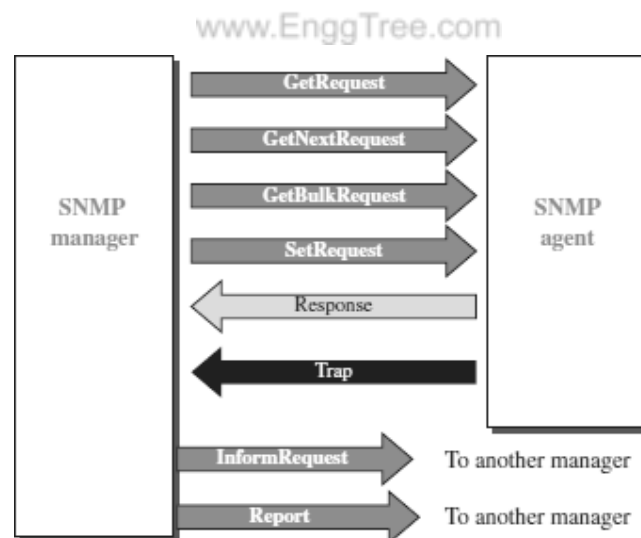
- Simple variables are accessed using *group-id* followed by *variable-id* and 0
- Tables are ordered as *column-row* rules, i.e., column by column from top to bottom. Only *leaf* elements are accessible in a table type.

**SNMP MESSAGES/PDU**

SNMP is request/reply protocol that supports various operations using PDUs.

SNMP defines eight types of protocol data units (or PDUs):

***GetRequest, GetNext-Request, GetBulkRequest, SetRequest, Response, Trap, InformRequest, and Report***

**GetRequest**

- The GetRequest PDU is sent from the manager (client) to the agent (server) to retrieve the value of a variable or a set of variables.

**GetNextRequest**

- The GetNextRequest PDU is sent from the manager to the agent to retrieve the value of a variable.

***GetBulkRequest***

- The GetBulkRequest PDU is sent from the manager to the agent to retrieve a large amount of data. It can be used instead of multiple GetRequest and GetNextRequest PDUs.

***SetRequest***

- The SetRequest PDU is sent from the manager to the agent to set (store) a value in a variable.

***Response***

- The Response PDU is sent from an agent to a manager in response to GetRequest or GetNextRequest. It contains the value(s) of the variable(s) requested by the manager.

***Trap***

- The Trap PDU is sent from the agent to the manager to report an event. For example, if the agent is rebooted, it informs the manager and reports the time of rebooting.

***InformRequest***

- The InformRequest PDU is sent from one manager to another remote manager to get the value of some variables from agents under the control of the remote manager. The remote manager responds with a Response PDU.

***Report***

- The Report PDU is designed to report some types of errors between managers.
-

## UNIT – II: TRANSPORT LAYER

Introduction - Transport-Layer Protocols: UDP – TCP: Connection

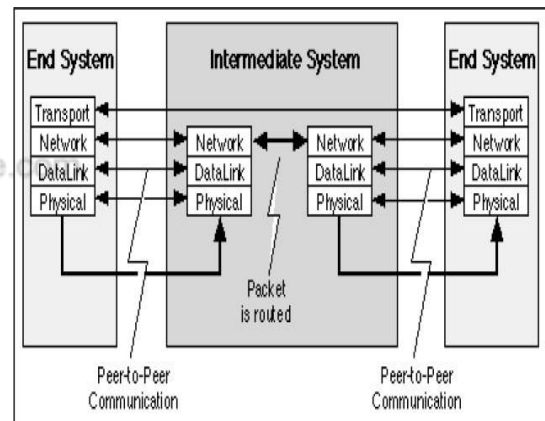
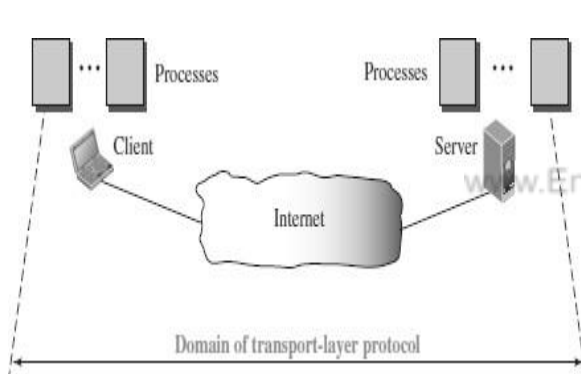
Management – Flow control -

Congestion Control - Congestion avoidance (DECbit, RED) – SCTP –

Quality of Service

### INTRODUCTION

- The transport layer is the fourth layer of the OSI model and is the core of the Internet model.
- It responds to service requests from the session layer and issues service requests to the network Layer.
- The transport layer provides transparent transfer of data between hosts.
- It provides end-to-end control and information transfer with the quality of service needed by the application program.
- It is the first true end-to-end layer, implemented in all End Systems (ES).



### TRANSPORT LAYER FUNCTIONS / SERVICES

- The transport layer is located between the network layer and the application layer.
- The transport layer is responsible for providing services to the application layer; it receives services from the network layer.
- The services that can be provided by the transport layer are
  1. Process-to-Process Communication
  2. Addressing: Port Numbers
  3. Encapsulation and Decapsulation
  4. Multiplexing and Demultiplexing
  5. Flow Control
  6. Error Control
  7. Congestion Control

### **Process-to-Process Communication**

- The Transport Layer is responsible for delivering data to the appropriate application process on the host computers.
- This involves multiplexing of data from different application processes, i.e. forming data packets, and adding source and destination port numbers in the header of each Transport Layer data packet.
- Together with the source and destination IP address, the port numbers constitute a network socket, i.e. an identification address of the process-to-process communication.

### **Addressing: Port Numbers**

- Ports are the essential ways to address multiple entities in the same location.
- Using port addressing it is possible to use more than one network-based application at the same time.
- Three types of Port numbers are used:
  - ✓ *Well-known ports* - These are permanent port numbers. They range between 0 to 1023. These port numbers are used by Server Process.
  - ✓ *Registered ports* - The ports ranging from 1024 to 49,151 are not assigned or controlled.
  - ✓ *Ephemeral ports (Dynamic Ports)* – These are temporary port numbers. They range between 49152–65535. These port numbers are used by Client Process.

### **Encapsulation and Decapsulation**

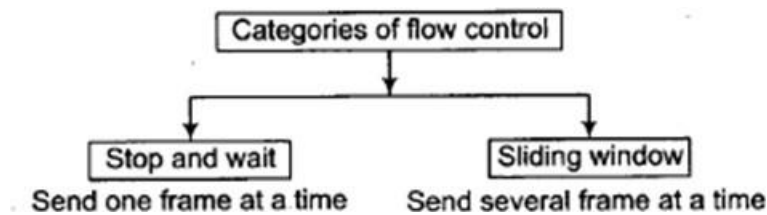
- To send a message from one process to another, the transport-layer protocol encapsulates and decapsulates messages.
- Encapsulation happens at the sender site. The transport layer receives the data and adds the transport-layer header.
- Decapsulation happens at the receiver site. When the message arrives at the destination transport layer, the header is dropped and the transport layer delivers the message to the process running at the application layer.

### **Multiplexing and Demultiplexing**

- Whenever an entity accepts items from more than one source, this is referred to as *multiplexing* (many to one).
- Whenever an entity delivers items to more than one source, this is referred to as *demultiplexing* (one to many).
- The transport layer at the source performs multiplexing
- The transport layer at the destination performs demultiplexing

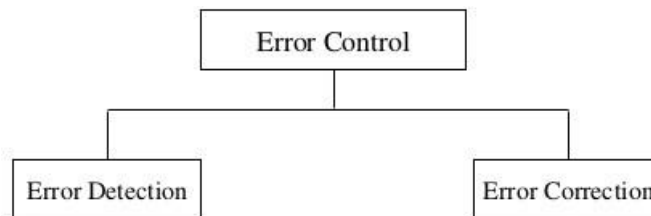
### **Flow Control**

- Flow Control is the process of managing the rate of data transmission between two nodes to prevent a fast sender from overwhelming a slow receiver.
- It provides a mechanism for the receiver to control the transmission speed, so that the receiving node is not overwhelmed with data from transmitting node.



### Error Control

- Error control at the transport layer is responsible for
  1. Detecting and discarding corrupted packets.
  2. Keeping track of lost and discarded packets and resending them.
  3. Recognizing duplicate packets and discarding them.
  4. Buffering out-of-order packets until the missing packets arrive.
- Error Control involves Error Detection and Error Correction



### Congestion Control

- Congestion in a network may occur if the *load* on the network (the number of packets sent to the network) is greater than the *capacity* of the network (the number of packets a network can handle).
- Congestion control refers to the mechanisms and techniques that control the congestion and keep the load below the capacity.
- Congestion Control refers to techniques and mechanisms that can either prevent congestion, before it happens, or remove congestion, after it has happened
- Congestion control mechanisms are divided into two categories,
  1. Open loop - prevent the congestion before it happens.
  2. Closed loop - remove the congestion after it happens.

---

## TRANSPORT LAYER PROTOCOLS

- Three protocols are associated with the Transport layer.
- They are
  - (1) **UDP –User Datagram Protocol**
  - (2) **TCP – Transmission Control Protocol**
  - (3) **SCTP - Stream Control Transmission Protocol**
- Each protocol provides a different type of service and should be used appropriately.



**UDP** - UDP is an unreliable connectionless transport-layer protocol used for its simplicity and efficiency in applications where error control can be provided by the application-layer process.

**TCP** - TCP is a reliable connection-oriented protocol that can be used in any application where reliability is important.

**SCTP** - SCTP is a new transport-layer protocol designed to combine some features of UDP and TCP in an effort to create a better protocol for multimedia communication.

*Position of transport-layer protocols in the TCP/IP protocol suite*

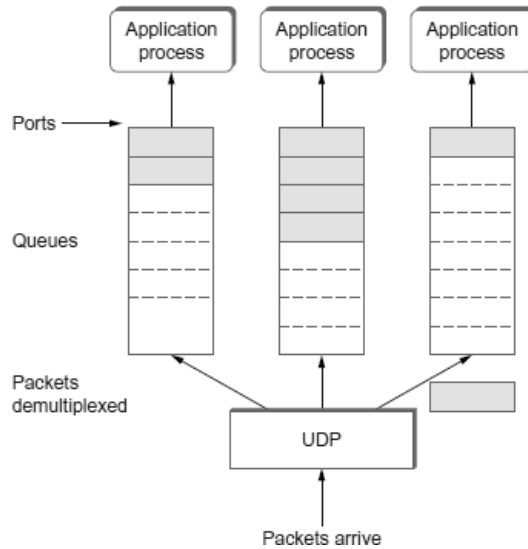


## **USER DATAGRAM PROTOCOL (UDP)**

- User Datagram Protocol (UDP) is a connectionless, unreliable transport protocol.
- UDP adds process-to-process communication to best-effort service provided by IP.
- UDP is a very simple protocol using a minimum of overhead.
- UDP is a simple demultiplexer, which allows multiple processes on each host to communicate.
- UDP does not provide flow control, reliable or ordered delivery.
- UDP can be used to send small message where reliability is not expected.
- Sending a small message using UDP takes much less interaction between the sender and receiver.
- UDP allow processes to indirectly identify each other using an abstract locator called port or mailbox

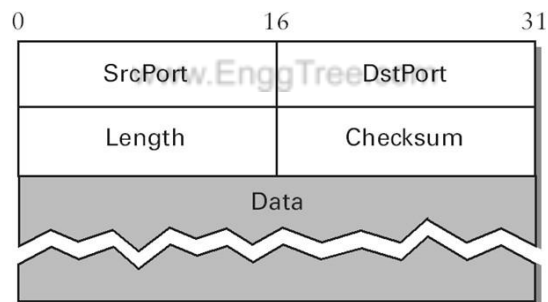
## **UDP PORTS**

- Processes (server/client) are identified by an abstract locator known as port.
- Server accepts message at *well known port*.
- Some well-known UDP ports are 7–Echo, 53–DNS, 111–RPC, 161–SNMP, etc.
- $\langle port, host \rangle$  pair is used as key for demultiplexing.
- Ports are implemented as a *message queue*.
- When a message arrives, UDP *appends* it to end of the queue.
- When queue is *full*, the message is discarded.
- When a message is *read*, it is removed from the queue.
- When an application process wants to receive a message, one is removed from the front of the queue.
- If the queue is empty, the process blocks until a message becomes available.



## UDP DATAGRAM (PACKET) FORMAT

- UDP packets are known as user *datagrams* .
- These *user datagrams*, have a fixed-size header of 8 bytes made of four fields, each of 2 bytes (16 bits).



### Source Port Number

- Port number used by process on source host with 16 bits long.
- If the source host is client (sending request) then the port number is a temporary one requested by the process and chosen by UDP.
- If the source is server (sending response) then it is a well-known port number.

### Destination Port Number

- Port number used by process on Destination host with 16 bits long.
- If the destination host is the server (a client sending request) then the port number is a well-known port number.
- If the destination host is client (a server sending response) then the port number is a temporary one copied by server from the request packet.

**Length**

- This field denotes the total length of the UDP Packet (Header plus data)
- The total length of any UDP datagram can be from 0 to 65,535 bytes.

**Checksum**

- UDP computes its checksum over the UDP header, the contents of the message body, and something called the pseudoheader.
- The pseudoheader consists of three fields from the IP header—protocol number, source IP address, destination IP address plus the UDP length field.

**Data**

- Data field defines the actual payload to be transmitted.
- Its size is variable.

**UDP SERVICES****Process-to-Process Communication**

- UDP provides process-to-process communication using socket addresses, a combination of IP addresses and port numbers.

**Connectionless Services**

- UDP provides a connectionless service.
- There is no connection establishment and no connection termination .
- Each user datagram sent by UDP is an independent datagram.
- There is no relationship between the different user datagrams even if they are
- coming from the same source process and going to the same destination program.
- The user datagrams are not numbered.
- Each user datagram can travel on a different path.

**Flow Control**

- UDP is a very simple protocol.
- There is no flow control, and hence no window mechanism.
- The receiver may overflow with incoming messages.
- The lack of flow control means that the process using UDP should provide for this service, if needed.

**Error Control**

- There is no error control mechanism in UDP except for the checksum.
- This means that the sender does not know if a message has been lost or duplicated.
- When the receiver detects an error through the checksum, the user datagram is silently discarded.

- The lack of error control means that the process using UDP should provide for this service, if needed.

### **Checksum**

- UDP checksum calculation includes three sections: a pseudoheader, the UDP header, and the data coming from the application layer.
- The pseudoheader is the part of the header in which the user datagram is to be encapsulated with some fields filled with 0s.

#### ***Optional Inclusion of Checksum***

- The sender of a UDP packet can choose not to calculate the checksum.
- In this case, the checksum field is filled with all 0s before being sent.
- In the situation where the sender decides to calculate the checksum, but it happens that the result is all 0s, the checksum is changed to all 1s before the packet is sent.
- In other words, the sender complements the sum two times.

### **Congestion Control**

- Since UDP is a connectionless protocol, it does not provide congestion control.
- UDP assumes that the packets sent are small and sporadic(occasionally or at irregular intervals) and cannot create congestion in the network.
- This assumption may or may not be true, when UDP is used for interactive real-time transfer of audio and video.

www.EnggTree.com

### **Encapsulation and Decapsulation**

- To send a message from one process to another, the UDP protocol encapsulates and decapsulates messages.

### **Queuing**

- In UDP, queues are associated with ports.
- At the client site, when a process starts, it requests a port number from the operating system.
- Some implementations create both an incoming and an outgoing queue associated with each process.
- Other implementations create only an incoming queue associated with each process.

### **Multiplexing and Demultiplexing**

- In a host running a transport protocol suite, there is only one UDP but possibly several processes that may want to use the services of UDP.
- To handle this situation, UDP multiplexes and demultiplexes.

## **APPLICATIONS OF UDP**

- UDP is used for management processes such as SNMP.
- UDP is used for route updating protocols such as RIP.
- UDP is a suitable transport protocol for multicasting. Multicasting capability is embedded in the UDP software
- UDP is suitable for a process with internal flow and error control mechanisms such as Trivial File Transfer Protocol (TFTP).
- UDP is suitable for a process that requires simple request-response communication with little concern for flow and error control.
- UDP is normally used for interactive real-time applications that cannot tolerate uneven delay between sections of a received message.

---

## **TRANSMISSION CONTROL PROTOCOL (TCP)**

- TCP is a reliable, connection-oriented, byte-stream protocol.
- TCP guarantees the reliable, in-order delivery of a stream of bytes. It is a full-duplex protocol, meaning that each TCP connection supports a pair of byte streams, one flowing in each direction.
- TCP includes a flow-control mechanism for each of these byte streams that allow the receiver to limit how much data the sender can transmit at a given time.
- TCP supports a demultiplexing mechanism that allows multiple application programs on any given host to simultaneously carry on a conversation with their peers.
- TCP also implements congestion-control mechanism. The idea of this mechanism is to prevent sender from overloading the network.
- Flow control is an end to end issue, whereas congestion control is concerned with how host and network interact.

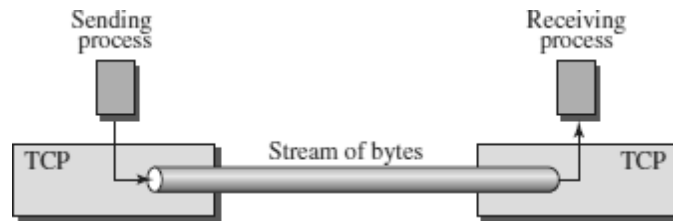
## **TCP SERVICES**

### **Process-to-Process Communication**

- TCP provides process-to-process communication using port numbers.

### **Stream Delivery Service**

- TCP is a stream-oriented protocol.
- TCP allows the sending process to deliver data as a stream of bytes and allows the receiving process to obtain data as a stream of bytes.
- TCP creates an environment in which the two processes seem to be connected by an imaginary “tube” that carries their bytes across the Internet.
- The sending process produces (writes to) the stream and the receiving process consumes (reads from) it.



### Full-Duplex Communication

- TCP offers full-duplex service, where data can flow in both directions at the same time.
- Each TCP endpoint then has its own sending and receiving buffer, and segments move in both directions.

### Multiplexing and Demultiplexing

TCP performs multiplexing at the sender and demultiplexing at the receiver.

### Connection-Oriented Service

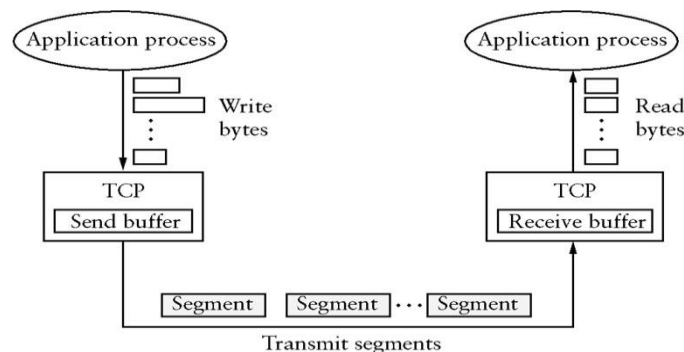
- TCP is a connection-oriented protocol.
- A connection needs to be established for each pair of processes.
- When a process at site A wants to send to and receive data from another process at site B, the following three phases occur:
  1. The two TCP's establish a logical connection between them.
  2. Data are exchanged in both directions.
  3. The connection is terminated.

### Reliable Service

- TCP is a reliable transport protocol.
- It uses an acknowledgment mechanism to check the safe and sound arrival of data.

### TCP SEGMENT

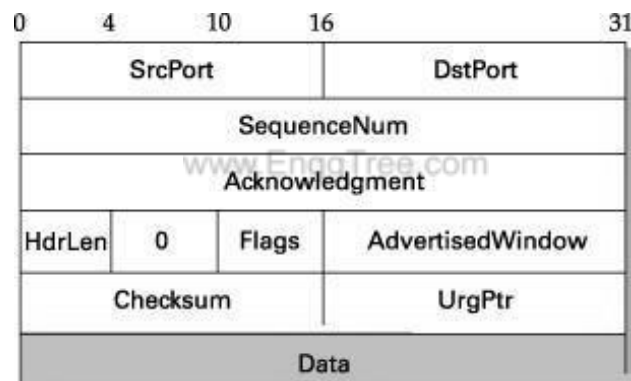
- A packet in TCP is called a segment.
- Data unit exchanged between TCP peers are called *segments*.
- A TCP segment encapsulates the data received from the application layer.
- The TCP segment is encapsulated in an IP datagram, which in turn is encapsulated in a frame at the data-link layer.



- TCP is a byte-oriented protocol, which means that the sender writes bytes into a TCP connection and the receiver reads bytes out of the TCP connection.
- TCP does not, itself, transmit individual bytes over the Internet.
- TCP on the source host buffers enough bytes from the sending process to fill a reasonably sized packet and then sends this packet to its peer on the destination host.
- TCP on the destination host then empties the contents of the packet into a receive buffer, and the receiving process reads from this buffer at its leisure.
- TCP connection supports byte streams flowing in both directions.
- The packets exchanged between TCP peers are called segments, since each one carries a segment of the byte stream.

### **TCP PACKET FORMAT**

- Each TCP segment contains the header plus the data.
- The segment consists of a header of 20 to 60 bytes, followed by data from the application program.
- The header is 20 bytes if there are no options and up to 60 bytes if it contains options.



**SrcPort and DstPort**—port number of source and destination process.

**SequenceNum**—contains sequence number, i.e. first byte of data segment.

**Acknowledgment**— byte number of segment, the receiver expects next.

**HdrLen**—Length of TCP header as 4-byte words.

**Flags**— contains *six* control bits known as flags.

- **URG** — segment contains urgent data.
- **ACK** — value of acknowledgment field is valid.
- **PUSH** — sender has invoked the push operation.
- **RESET** — receiver wants to abort the connection.
- **SYN** — synchronize sequence numbers during connection establishment.
- **FIN** — terminates the TCP connection.

**Advertised Window**—defines receiver's window size and acts as flow control.

**Checksum**—It is computed over TCP header, Data, and pseudo header containing IP fields (Length, SourceAddr & DestinationAddr).

**UrgPtr** — used when the segment contains urgent data. It defines a value that must be added to the sequence number.

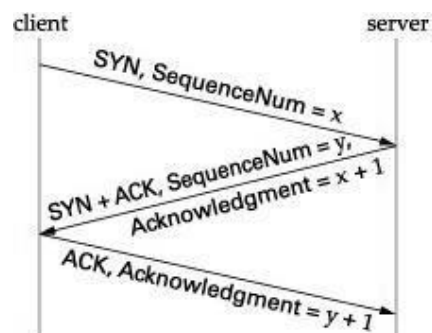
**Options - There** can be up to 40 bytes of optional information in the TCP header.

## TCP CONNECTION MANAGEMENT

- TCP is connection-oriented.
- A connection-oriented transport protocol establishes a logical path between the source and destination.
- All of the segments belonging to a message are then sent over this logical path.
- In TCP, connection-oriented transmission requires three phases: Connection Establishment, Data Transfer and Connection Termination.

### Connection Establishment

- While opening a TCP connection the two nodes (client and server) want to agree on a set of parameters.
- The parameters are the starting sequence numbers that is to be used for their respective byte streams.
- Connection establishment in TCP is a *three-way handshaking*.



1. Client sends a SYN segment to the server containing its initial sequence number (Flags = SYN, SequenceNum =  $x$ )
2. Server responds with a segment that acknowledges client's segment and specifies its initial sequence number (Flags = SYN + ACK, ACK =  $x + 1$  SequenceNum =  $y$ ).
3. Finally, client responds with a segment that acknowledges server's sequence number (Flags = ACK, ACK =  $y + 1$ ).



- The reason that each side acknowledges a sequence number that is one larger than the one sent is that the Acknowledgment field actually identifies the “next sequence number expected,”
- A timer is scheduled for each of the first two segments, and if the expected response is not received, the segment is retransmitted.

## Data Transfer

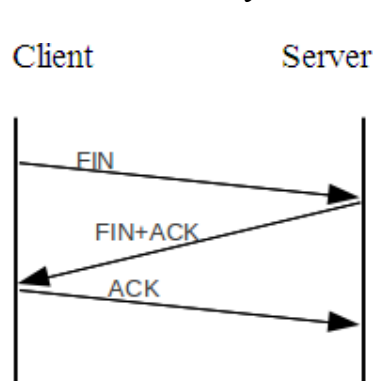
- After connection is established, bidirectional data transfer can take place.
- The client and server can send data and acknowledgments in both directions.
- The data traveling in the same direction as an acknowledgment are carried on the same segment.
- The acknowledgment is piggybacked with the data.

## Connection Termination

- Connection termination or teardown can be done in two ways:

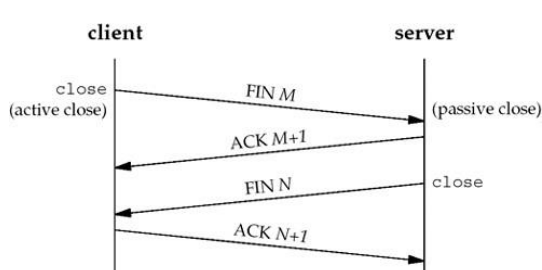
### Three-way Close and Half-Close

**Three-way Close**—Both client and server close *simultaneously*.



- Client sends a FIN segment.
- The FIN segment can include last chunk of data.
- Server responds with FIN + ACK segment to inform its closing.
- Finally, client sends an ACK segment

**Half-Close**—Client stops sending but receives data.

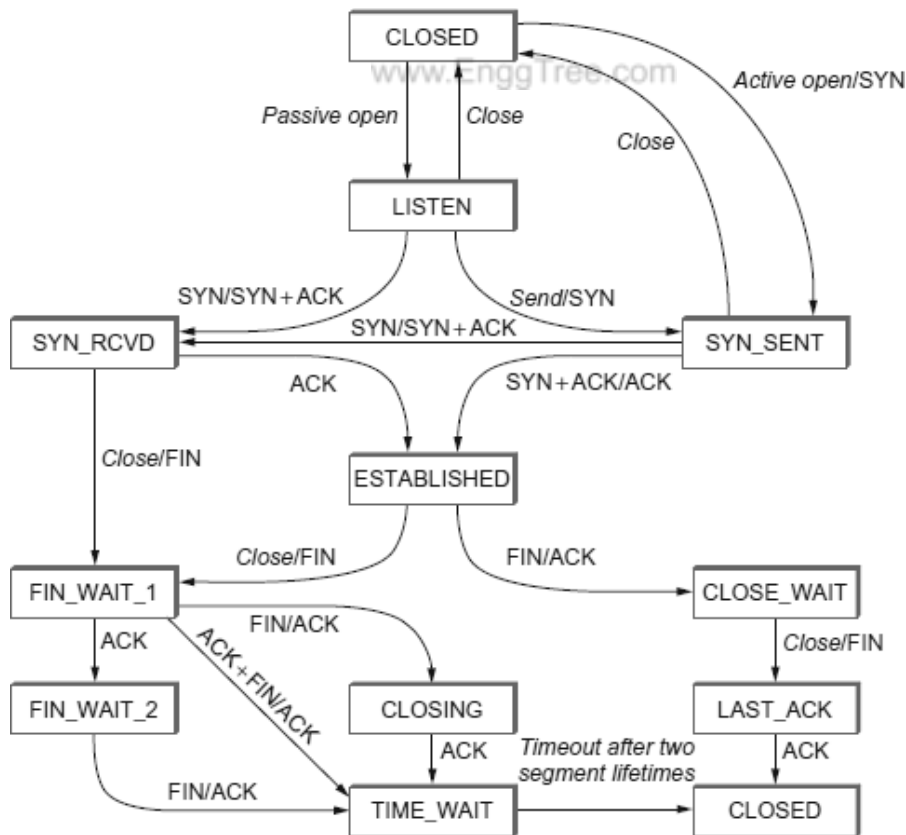


- Client half-closes the connection by sending a FIN segment.
- Server sends an ACK segment.
- Data transfer from client to the server *stops*.
- After sending all data, server sends FIN segment to client, which is acknowledged by the client.

### STATE TRANSITION DIAGRAM

- To keep track of all the different events happening during connection establishment, connection termination, and data transfer, TCP is specified as the finite state machine (FSM).
- The transition from one state to another is shown using directed lines.
- States involved in opening and closing a connection is shown above and below ESTABLISHED state respectively.
- States Involved in TCP :

State	Description
CLOSED	No connection is active or pending
LISTEN	The server is waiting for an incoming call
SYN RCVD	A connection request has arrived; wait for ACK
SYN SENT	The application has started to open a connection
ESTABLISHED	The normal data transfer state
FIN WAIT 1	The application has said it is finished
FIN WAIT 2	The other side has agreed to release
TIMED WAIT	Wait for all packets to die off
CLOSING	Both sides have tried to close simultaneously
CLOSE WAIT	The other side has initiated a release
LAST ACK	Wait for all packets to die off



## Opening a TCP Connection

1. Server invokes a *passive* open on TCP, which causes TCP to move to LISTEN state
2. Client does an *active* open, which causes its TCP to send a SYN segment to the server and move to SYN\_SENT state.
3. When SYN segment arrives at the server, it moves to SYN\_RCVD state and *responds* with a SYN + ACK segment.
4. Arrival of SYN + ACK segment causes the client to move to ESTABLISHED state and sends an ACK to the server.
5. When ACK arrives, the server finally moves to ESTABLISHED state.

## Closing a TCP Connection

1. Client / Server can independently close its half of the connection or simultaneously.

Transitions from ESTABLISHED to CLOSED state are:

### *One side closes:*

ESTABLISHED → FIN\_WAIT\_1 → FIN\_WAIT\_2 → TIME\_WAIT → CLOSED

### *Other side closes:*

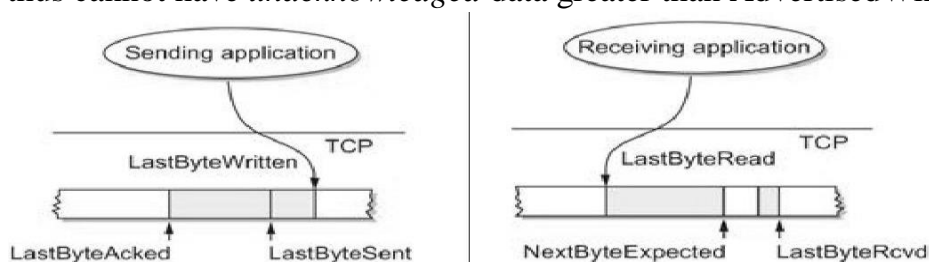
ESTABLISHED → CLOSE\_WAIT → LAST\_ACK → CLOSED

### *Simultaneous close:*

ESTABLISHED → FIN\_WAIT\_1 → CLOSING → TIME\_WAIT → CLOSED

## TCP FLOW CONTROL

- TCP uses a variant of sliding window known as adaptive flow control that:
  - \* guarantees *reliable* delivery of data
  - \* Ensures *ordered* delivery of data
  - \* Enforces *flow control* at the sender
- Receiver advertises its window size to the sender using AdvertisedWindow field.
- Sender thus cannot have *unacknowledged* data greater than AdvertisedWindow.



## Send Buffer

- Sending TCP maintains *send buffer* which contains 3 segments
  - (1) acknowledged data
  - (2) unacknowledged data
  - (3) data to be transmitted.
- Send buffer maintains three *pointers*
  - (1) LastByteAked, (2) LastByteSent, and (3) LastByteWritten
 such that:

$$\mathbf{LastByteAked \leq LastByteSent \leq LastByteWritten}$$

- A byte can be sent only *after* being written and only a sent byte *can be* acknowledged.
- Bytes to the *left* of LastByteAked are not kept as it had been acknowledged.

## Receive Buffer

- Receiving TCP maintains *receive buffer* to hold data even if it arrives out-of-order.
- Receive buffer maintains three *pointers* namely
  - (1) LastByteRead, (2) NextByteExpected, and (3) LastByteRcvd
 such that:

$$\mathbf{LastByteRead \leq NextByteExpected \leq LastByteRcvd + 1}$$

- A byte *cannot* be read until that byte and all preceding bytes have been received.
- If data is received *in order*, then NextByteExpected = LastByteRcvd + 1
- Bytes to the *left* of LastByteRead are not buffered, since it is read by the application.

## Flow Control in TCP

- Size of *send* and *receive* buffer is *MaxSendBuffer* and *MaxRcvBuffer* respectively.
- Sending TCP prevents *overflowing* of send buffer by maintaining

$$\mathbf{LastByteWritten - LastByteAked \leq MaxSendBuffer}$$

- Receiving TCP avoids *overflowing* its receive buffer by maintaining

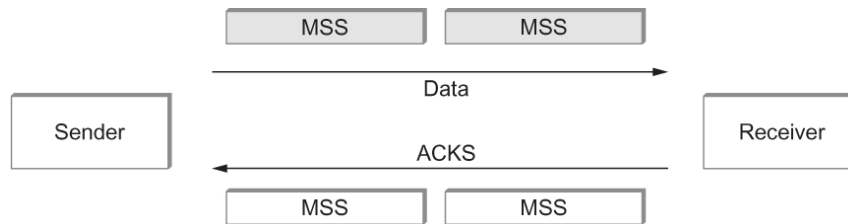
$$\mathbf{LastByteRcvd - LastByteRead \leq MaxRcvBuffer}$$

## **TCP TRANSMISSION**

- TCP has the mechanisms to trigger the transmission of a segment.
- They are
  - Maximum Segment Size (MSS) - Silly Window Syndrome
  - Timeout - Nagle's Algorithm

### **Silly Window Syndrome**

- When either the sending application program creates data slowly or the receiving application program consumes data slowly, or both, problems arise.
- Any of these situations results in the sending of data in very small segments, which reduces the efficiency of the operation.
- This problem is called the silly window syndrome.
- The sending TCP may create a silly window syndrome if it is serving an application program that creates data slowly, for example, 1 byte at a time.
- The application program writes 1 byte at a time into the buffer of the sending TCP.
- The result is a lot of 1-byte segments that are traveling through an internet.
- The solution is to prevent the sending TCP from sending the data byte by byte.
- The sending TCP must be forced to wait and collect data to send in a larger block.



### Nagle's Algorithm

- If there is data to send but is less than MSS, then we may want to wait some amount of time before sending the available data
- If we wait too long, then it may delay the process.
- If we don't wait long enough, it may end up sending small segments resulting in Silly Window Syndrome.
- The solution is to introduce a timer and to transmit when the timer expires
- Nagle introduced an algorithm for solving this problem

```

When the application produces data to send
  if (both the available data and the window) = MSS
    Send the full segment
  else
    if (there is unACKed data)
      Buffer the new data until an ACK arrives
    else
      Send all the new data now

```

## TCP CONGESTION CONTROL

- Congestion occurs if load (number of packets sent) is greater than capacity of the network (number of packets a network can handle).
- When load is less than network capacity, throughput increases proportionally.
- When load exceeds capacity, queues become full and the routers discard some packets and throughput declines sharply.
- When too many packets are contending for the same link
  - The queue overflows
  - Packets get dropped
  - Network is congested
- Network should provide a congestion control mechanism to deal with such a situation.
- TCP maintains a variable called *CongestionWindow* for each *connection*.
- TCP Congestion Control mechanisms are:

1. Additive Increase / Multiplicative Decrease (AIMD)
2. Slow Start
3. Fast Retransmit and Fast Recovery

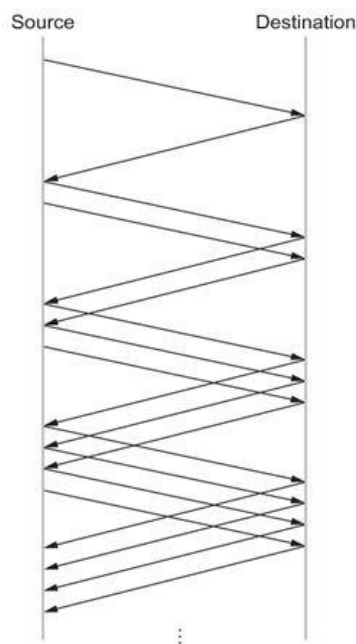
### Additive Increase / Multiplicative Decrease (AIMD)

- TCP source *initializes* CongestionWindow based on congestion level in the network.
- Source *increases* CongestionWindow when level of congestion goes down and *decreases* the same when level of congestion goes up.
- TCP interprets *timeouts* as a sign of congestion and reduces the rate of transmission.
- On timeout, source reduces its CongestionWindow by half, i.e., *multiplicative decrease*. For example, if CongestionWindow = 16 packets, after timeout it is 8.
- Value of CongestionWindow is never less than maximum segment size (MSS).
- When ACK arrives CongestionWindow is *incremented* marginally, i.e., *additive increase*.

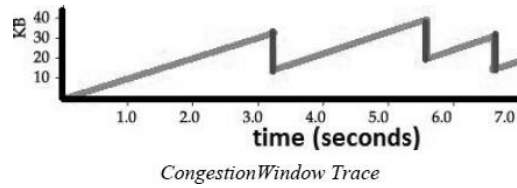
$$\text{Increment} = \text{MSS} \times (\text{MSS}/\text{CongestionWindow})$$

$$\text{CongestionWindow} += \text{Increment}$$

- For *example*, when ACK arrives for 1 packet, 2 packets are sent. When ACK for both packets arrive, 3 packets are sent and so on.
- CongestionWindow increases and decreases throughout *lifetime* of the connection.



- When CongestionWindow is plotted as a function of time, a *saw-tooth* pattern results.



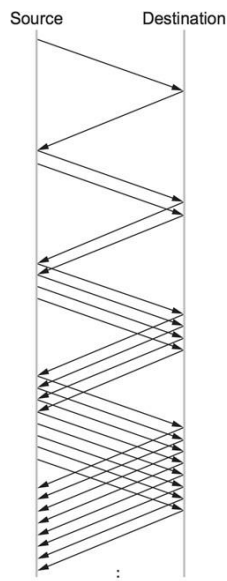
## Slow Start

- Slow start is used to increase CongestionWindow *exponentially* from a cold start.
- Source TCP *initializes* CongestionWindow to one packet.
- TCP *doubles* the number of packets sent every RTT on successful transmission.
- When ACK arrives for first packet TCP adds 1 packet to CongestionWindow and sends two packets.
- When two ACKs arrive, TCP increments CongestionWindow by 2 packets and sends four packets and so on.
- Instead of sending entire permissible packets at once (bursty traffic), packets are sent in a phased manner, i.e., *slow start*.
- Initially TCP has no idea about congestion, henceforth it increases CongestionWindow rapidly until there is a timeout. On timeout:

$$\text{CongestionThreshold} = \text{CongestionWindow} / 2$$

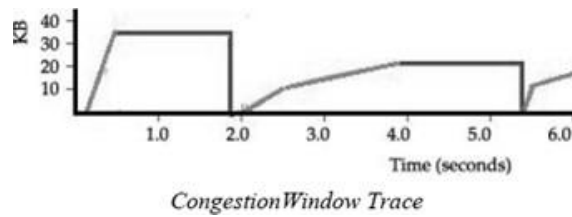
$$\text{CongestionWindow} = 1$$

- Slow start is repeated until CongestionWindow reaches CongestionThreshold and thereafter 1 packet per RTT.



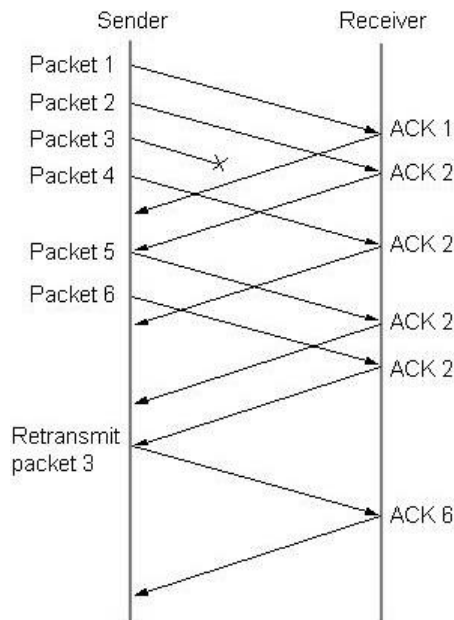


- The congestion window trace will look like

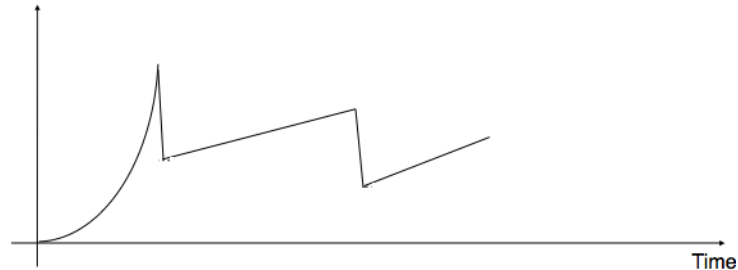


### Fast Retransmit And Fast Recovery

- TCP timeouts led to long periods of time during which the connection went dead while waiting for a timer to expire.
- Fast retransmit is a heuristic approach that *triggers* retransmission of a dropped packet sooner than the regular timeout mechanism. It *does not* replace regular timeouts.
- When a packet arrives out of order, receiving TCP resends the same acknowledgment (*duplicate ACK*) it sent last time.
- When *three duplicate ACK* arrives at the sender, it infers that corresponding packet may be lost due to congestion and retransmits that packet. This is called **fast retransmit** before regular timeout.
- When packet loss is detected using fast retransmit, the slow start phase is replaced by additive increase, multiplicative decrease method. This is known as **fast recovery**.
- Instead of setting CongestionWindow to one packet, this method uses the ACKs that are still in pipe to clock the sending of packets.
- Slow start is only used at the beginning of a connection and after *regular* timeout. At other times, it follows a pure AIMD pattern.
- 



- For example, packets 1 and 2 are received whereas packet 3 gets lost.
  - Receiver sends a duplicate ACK for packet 2 when packet 4 arrives.
  - Sender receives 3 duplicate ACKs after sending packet 6 retransmits packet 3.
  - When packet 3 is received, receiver sends cumulative ACK up to packet 6.
- The congestion window trace will look like



## TCP CONGESTION AVOIDANCE

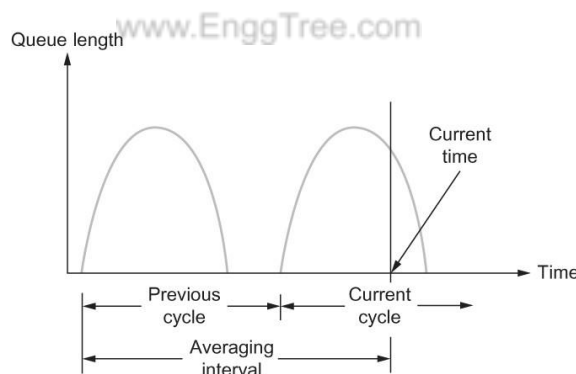
- Congestion avoidance mechanisms *prevent* congestion before it actually occurs.
- These mechanisms predict when congestion is about to happen and then to reduce the rate at which hosts send data just before packets start being discarded.
- TCP *creates* loss of packets in order to determine bandwidth of the connection.
- Routers *help* the end nodes by intimating when congestion is likely to occur.
- Congestion-avoidance mechanisms are:
  - DEC bit - Destination Experiencing Congestion Bit
  - RED - Random Early Detection

### **Dec Bit - Destination Experiencing Congestion Bit**

- The first mechanism developed for use on the Digital Network Architecture (DNA).
- The idea is to evenly split the responsibility for congestion control between the routers and the end nodes.
- Each router monitors the load it is experiencing and explicitly notifies the end nodes when congestion is about to occur.
- This notification is implemented by setting a binary congestion bit in the packets that flow through the router; hence the name DECbit.

- The destination host then copies this congestion bit into the ACK it sends back to the source.
- The Source checks *how many* ACK has DEC bit set for previous window packets.
- If less than 50% of ACK have DEC bit set, then source *increases* its congestion window by 1 packet
- Otherwise, *decreases* the congestion window by 87.5%.
- Finally, the source adjusts its sending rate so as to avoid congestion.
- *Increase by 1, decrease by 0.875* rule was based on AIMD for stabilization.
- A single congestion bit is added to the packet header.
- Using a queue length of 1 as the trigger for setting the congestion bit.
- A router sets this bit in a packet if its average queue length is greater than or equal to 1 at the time the packet arrives.

### Computing average queue length at a router using DEC bit



- Average queue length is measured over a time interval that includes the ***last busy + last idle cycle + current busy cycle.***
- It calculates the average queue length by *dividing* the curve area with time interval.

### Red - Random Early Detection

- The second mechanism of congestion avoidance is called as *Random Early Detection (RED)*.

- Each router is programmed to monitor its own queue length, and when it detects that there is congestion, it notifies the source to adjust its congestion window.
- RED differs from the DEC bit scheme by two ways:
  - a. In DECbit, explicit notification about congestion is sent to source, whereas RED implicitly notifies the source by dropping a few packets.
  - b. DECbit may lead to tail drop policy, whereas RED drops packet based on drop probability in a random manner. Drop each arriving packet with some **drop probability** whenever the queue length exceeds some *drop level*. This idea is called **early random drop**.

### Computation of average queue length using RED

- $AvgLen = (1 - Weight) \times AvgLen + Weight \times SampleLen$

where  $0 < Weight < 1$  and

SampleLen – is the length of the queue when a sample measurement is made.

- The queue length is measured every time a new packet arrives at the gateway.
- RED has two queue length thresholds that trigger certain activity: **MinThreshold and MaxThreshold**
- When a packet arrives at a gateway it compares AvgLen with these two values according to the following rules.

if  $AvgLen \leq MinThreshold$

→ queue the packet

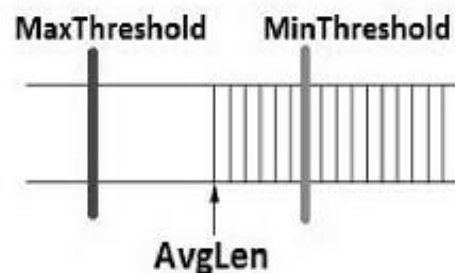
if  $MinThreshold < AvgLen < MaxThreshold$

→ calculate probability P

→ drop the arriving packet with probability P

if  $AvgLen \geq MaxThreshold$

→ drop the arriving packet



## SCTP

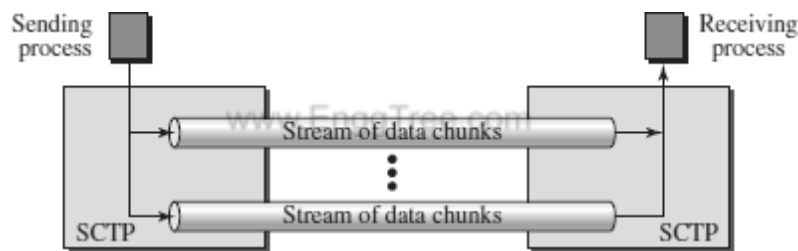
- Stream Control Transmission Protocol (SCTP) is a reliable, message-oriented transport layer protocol.
- SCTP has mixed features of TCP and UDP.
- SCTP maintains the message boundaries and detects the lost data, duplicate data as well as out-of-order data.
- SCTP provides the Congestion control as well as Flow control.
- SCTP is especially designed for internet applications as well as multimedia communication.

### Process-to-Process Communication

- SCTP provides process-to-process communication.

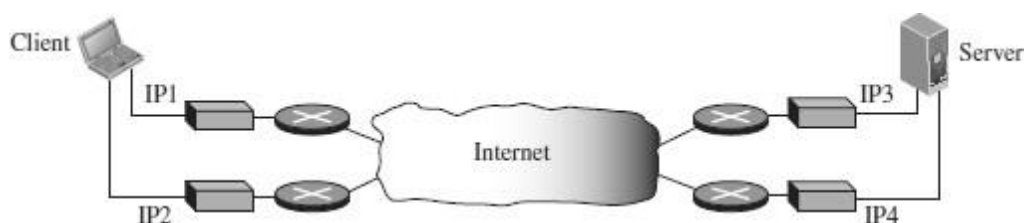
### Multiple Streams

- SCTP allows multistream service in each connection, which is called *association* in SCTP terminology.
- If one of the streams is blocked, the other streams can still deliver their data.



### Multihoming

- An SCTP association supports multihoming service.
- The sending and receiving host can define multiple IP addresses in each end for an association.
- In this fault-tolerant approach, when one path fails, another interface can be used for data delivery without interruption.



### Full-Duplex Communication

- SCTP offers full-duplex service, where data can flow in both directions at the same time. Each SCTP then has a sending and receiving buffer and packets are sent in both directions.

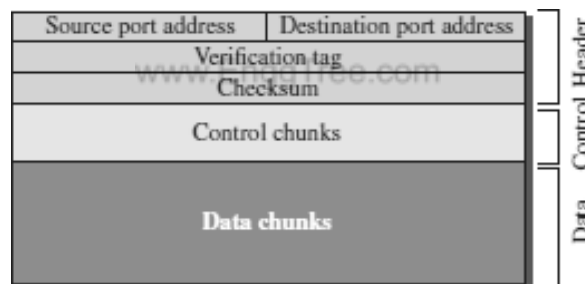
### Connection-Oriented Service

- SCTP is a connection-oriented protocol.
- In SCTP, a connection is called an *association*.
- If a client wants to send and receive message from server, the steps are :  
**Step1:** The two SCTPs establish the connection with each other.  
**Step2:** Once the connection is established, the data gets exchanged in both the directions.  
**Step3:** Finally, the association is terminated.

### Reliable Service

- SCTP is a reliable transport protocol.
- It uses an acknowledgment mechanism to check the safe and sound arrival of data.

## SCTP PACKET FORMAT



An SCTP packet has a mandatory general header and a set of blocks called chunks.

### General Header

- The *general header* (packet header) defines the end points of each association to which the packet belongs
- It guarantees that the packet belongs to a particular association
- It also preserves the integrity of the contents of the packet including the header itself.
- There are four fields in the general header.

#### *Source port*

This field identifies the sending port.

#### *Destination port*

This field identifies the receiving port that hosts use to route the packet to the appropriate endpoint/application.

**Verification tag**

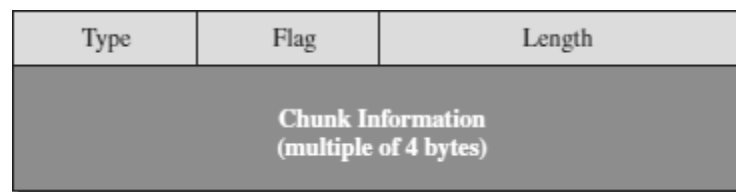
A 32-bit random value created during initialization to distinguish stale packets from a previous connection.

**Checksum**

The next field is a checksum. The size of the checksum is 32 bits. SCTP uses CRC-32 Checksum.

**Chunks**

- Control information or user data are carried in chunks.
- Chunks have a common layout.
- The first three fields are common to all chunks; the information field depends on the type of chunk.



- The type field can define up to 256 types of chunks. Only a few have been defined so far; the rest are reserved for future use.
- The flag field defines special flags that a particular chunk may need.
- The length field defines the total size of the chunk, in bytes, including the type, flag, and length fields.

www.EnggTree.com

**Types of Chunks**

- An SCTP association may send many packets, a packet may contain several chunks, and chunks may belong to different streams.
- SCTP defines two types of chunks - Control chunks and Data chunks.
- A control chunk controls and maintains the association.
- A data chunk carries user data.

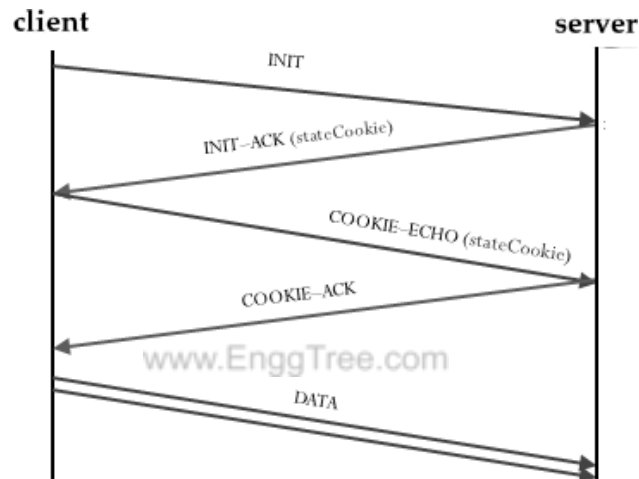
Type	Chunk	Description
0	DATA	User data
1	INIT	Sets up an association
2	INIT ACK	Acknowledges INIT chunk
3	SACK	Selective acknowledgment
4	HEARTBEAT	Probes the peer for liveness
5	HEARTBEAT ACK	Acknowledges HEARTBEAT chunk
6	ABORT	Aborts an association
7	SHUTDOWN	Terminates an association
8	SHUTDOWN ACK	Acknowledges SHUTDOWN chunk
9	ERROR	Reports errors without shutting down
10	COOKIE ECHO	Third packet in association establishment
11	COOKIE ACK	Acknowledges COOKIE ECHO chunk
14	SHUTDOWN COMPLETE	Third packet in association termination
192	FORWARD TSN	For adjusting cumulating TSN

## SCTP ASSOCIATION

- SCTP is a connection-oriented protocol.
- A connection in SCTP is called an *association* to emphasize multihoming.
- SCTP Associations consists of three phases:
  - Association Establishment
  - Data Transfer
  - Association Termination

### **Association Establishment**

- Association establishment in SCTP requires a four-way handshake.
- In this procedure, a client process wants to establish an association with a server process using SCTP as the transport-layer protocol.
- The SCTP server needs to be prepared to receive any association (passive open).
- Association establishment, however, is initiated by the client (active open).



- The client sends the first packet, which contains an INIT chunk.
- The server sends the second packet, which contains an INIT ACK chunk. The INIT ACK also sends a cookie that defines the state of the server at this moment.
- The client sends the third packet, which includes a COOKIE ECHO chunk. This is a very simple chunk that echoes, without change, the cookie sent by the server. SCTP allows the inclusion of data chunks in this packet.
- The server sends the fourth packet, which includes the COOKIE ACK chunk that acknowledges the receipt of the COOKIE ECHO chunk. SCTP allows the inclusion of data chunks with this packet.

### **Data Transfer**

- The whole purpose of an association is to transfer data between two ends.
- After the association is established, bidirectional data transfer can take place.
- The client and the server can both send data.
- SCTP supports piggybacking.



- Types of SCTP data Transfer :

### 1. Multihoming Data Transfer

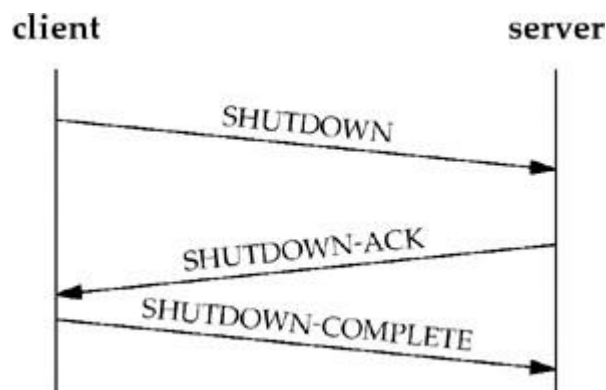
- Data transfer, by default, uses the primary address of the destination.
- If the primary is not available, one of the alternative addresses is used.
- This is called Multihoming Data Transfer.

### 2. Multistream Delivery

- SCTP can support multiple streams, which means that the sender process can define different streams and a message can belong to one of these streams.
- Each stream is assigned a stream identifier (SI) which uniquely defines that stream.
- SCTP supports two types of data delivery in each stream: *ordered* (default) and *unordered*.

### Association Termination

- In SCTP, either of the two parties involved in exchanging data (client or server) can close the connection.
- SCTP does not allow a “half closed” association. If one end closes the association, the other end must stop sending new data.
- If any data are left over in the queue of the recipient of the termination request, they are sent and the association is closed.
- Association termination uses three packets.



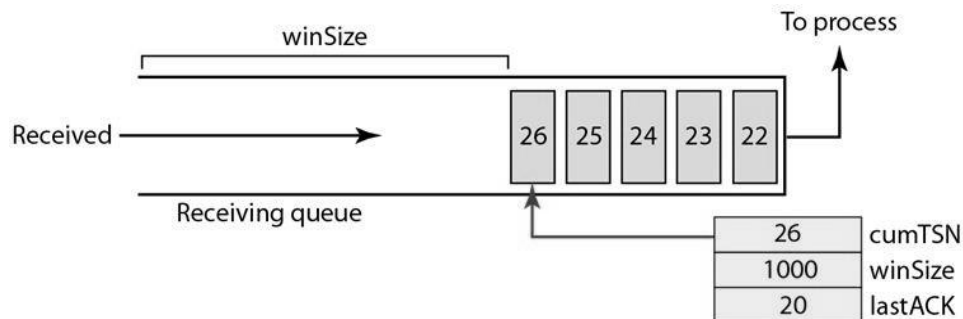
### SCTP FLOW CONTROL

- Flow control in SCTP is similar to that in TCP.
- Current SCTP implementations use a byte-oriented window for flow control.

### Receiver Site

- The receiver has one buffer (queue) and three variables.

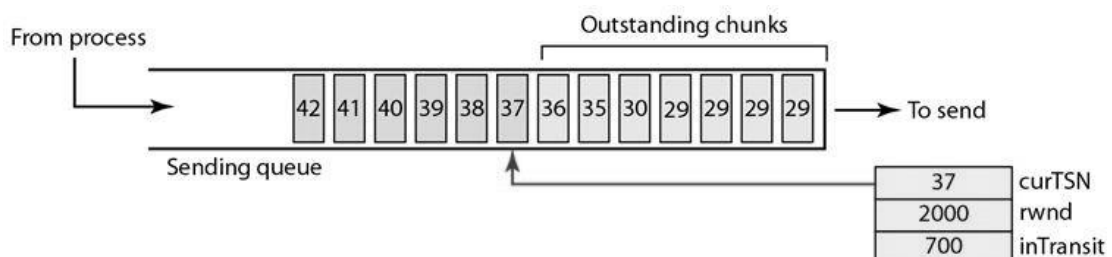
- The queue holds the received data chunks that have not yet been read by the process.
- The first variable holds the last TSN received, cumTSN.
- The second variable holds the available buffer size; winSize.
- The third variable holds the last accumulative acknowledgment, lastACK.
- The following figure shows the queue and variables at the receiver site.



- When the site receives a data chunk, it stores it at the end of the buffer (queue) and subtracts the size of the chunk from winSize.
- The TSN number of the chunk is stored in the cumTSN variable.
- When the process reads a chunk, it removes it from the queue and adds the size of the removed chunk to winSize (recycling).
- When the receiver decides to send a SACK, it checks the value of lastAck; if it is less than cumTSN, it sends a SACK with a cumulative TSN number equal to the cumTSN.
- It also includes the value of winSize as the advertised window size.

### Sender Site

- The sender has one buffer (queue) and three variables: curTSN, rwnd, and inTransit.
- We assume each chunk is 100 bytes long. The buffer holds the chunks produced by the process that either have been sent or are ready to be sent.
- The first variable, curTSN, refers to the next chunk to be sent.
- All chunks in the queue with a TSN less than this value have been sent, but not acknowledged; they are outstanding.
- The second variable, rwnd, holds the last value advertised by the receiver (in bytes).
- The third variable, inTransit, holds the number of bytes in transit, bytes sent but not yet acknowledged.
- The following figure shows the queue and variables at the sender site.



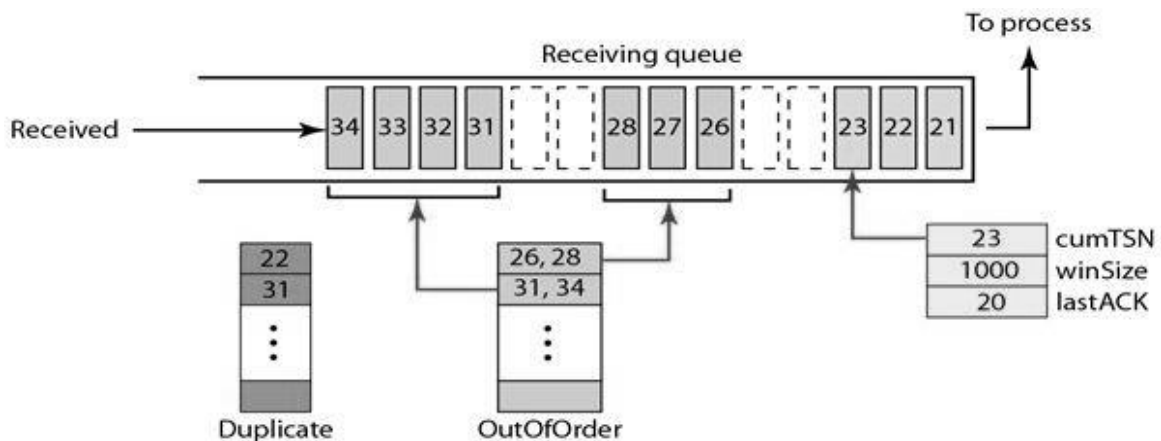
- A chunk pointed to by curTSN can be sent if the size of the data is less than or equal to the quantity  $rwnd - inTransit$ .
- After sending the chunk, the value of curTSN is incremented by 1 and now points to the next chunk to be sent.
- The value of inTransit is incremented by the size of the data in the transmitted chunk.
- When a SACK is received, the chunks with a TSN less than or equal to the cumulative TSN in the SACK are removed from the queue and discarded. The sender does not have to worry about them anymore.
- The value of inTransit is reduced by the total size of the discarded chunks.
- The value of rwnd is updated with the value of the advertised window in the SACK.

### SCTP ERROR CONTROL

- SCTP is a reliable transport layer protocol.
- It uses a SACK chunk to report the state of the receiver buffer to the sender.
- Each implementation uses a different set of entities and timers for the receiver and sender sites.

### Receiver Site

- The receiver stores all chunks that have arrived in its queue including the out-of-order ones. However, it leaves spaces for any missing chunks.
- It discards duplicate messages, but keeps track of them for reports to the sender.
- The following figure shows a typical design for the receiver site and the state of the receiving queue at a particular point in time.

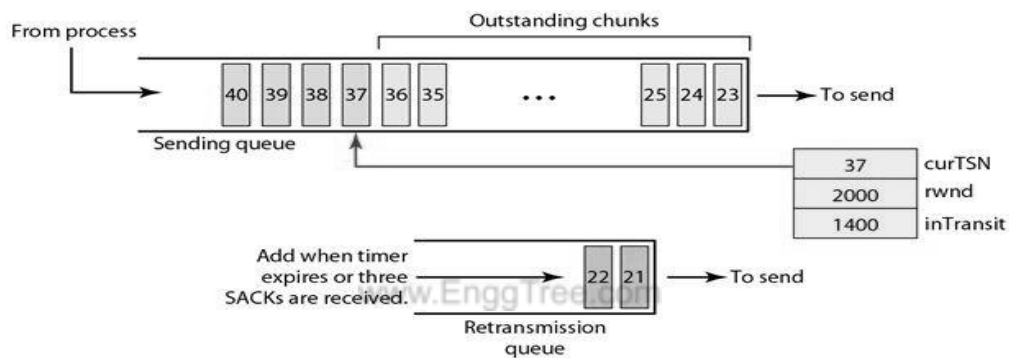


- The available window size is 1000 bytes.
- The last acknowledgment sent was for data chunk 20.
- Chunks 21 to 23 have been received in order.
- The first out-of-order block contains chunks 26 to 28.
- The second out-of-order block contains chunks 31 to 34.
- A variable holds the value of cumTSN.

- An array of variables keeps track of the beginning and the end of each block that is out of order.
- An array of variables holds the duplicate chunks received.
- There is no need for storing duplicate chunks in the queue and they will be discarded.

### Sender Site

- At the sender site, it needs two buffers (queues): a sending queue and a retransmission queue.
- Three variables were used - `rwnd`, `inTransit`, and `curTSN` as described in the previous section.
- The following figure shows a typical design.



- The sending queue holds chunks 23 to 40.
- The chunks 23 to 36 have already been sent, but not acknowledged; they are outstanding chunks.
- The `curTSN` points to the next chunk to be sent (37).
- We assume that each chunk is 100 bytes, which means that 1400 bytes of data (chunks 23 to 36) is in transit.
- The sender at this moment has a retransmission queue.
- When a packet is sent, a retransmission timer starts for that packet (all data chunks in that packet).
- Some implementations use one single timer for the entire association, but other implementations use one timer for each packet.

## UNIT III - NETWORK LAYER

Switching: Packet Switching - Internet protocol - IPV4 – IP Addressing – Subnetting - IPV6, ARP, RARP, ICMP, DHCP

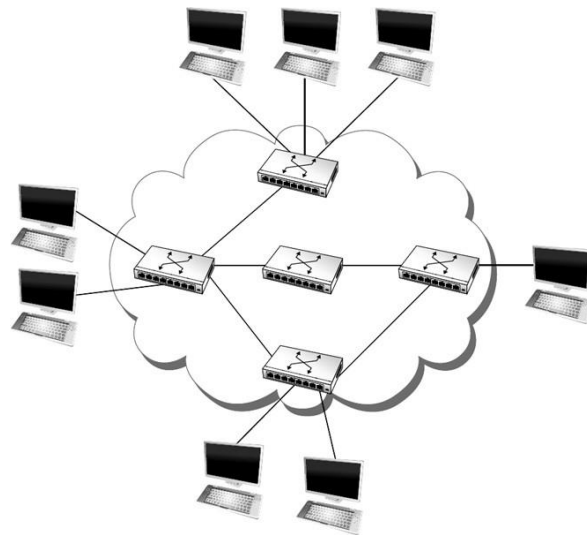
- The network layer works for the transmission of data from one host to the other located in different networks.
- It provides end to end communication by forwarding packets from source to destination.
- It also takes care of packet routing i.e., selection of the shortest path to transmit the packet, from the number of routes available.
- The sender & receiver's IP addresses are placed in the header by the network layer.

### NETWORK LAYER SERVICES:

- Packeting
- Routing and forwarding
- Addressing
- Error control
- Flow control
- Congestion control
- Quality of service

## SWITCHING

- The technique of transferring the information from one computer network to another network is known as **switching**.
- Switching in a computer network is achieved by using switches.
- A switch is a small hardware device which is used to join multiple computers together with one local area network (LAN).
- Switches are devices capable of creating temporary connections between two or more devices linked to the switch.
- Switches are used to forward the packets based on MAC addresses.
- A Switch is used to transfer the data only to the device that has been addressed. It verifies the destination address to route the packet appropriately.
- It is operated in full duplex mode.
- It does not broadcast the message as it works with limited bandwidth.



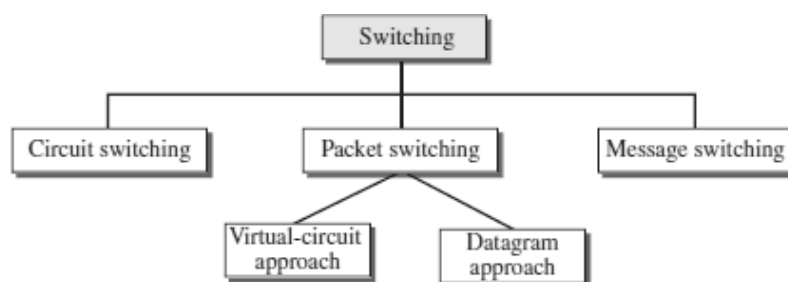
### ***Advantages of Switching:***

- Switch increases the bandwidth of the network.
- It reduces the workload on individual PCs as it sends the information to only that device which has been addressed.
- It increases the overall performance of the network by reducing the traffic on the network.
- There will be less frame collision as switch creates the collision domain for each connection.

### ***Disadvantages of Switching:***

- A Switch is more expensive than network bridges.
- A Switch cannot determine the network connectivity issues easily.
- Proper designing and configuration of the switch are required to handle multicast packets.

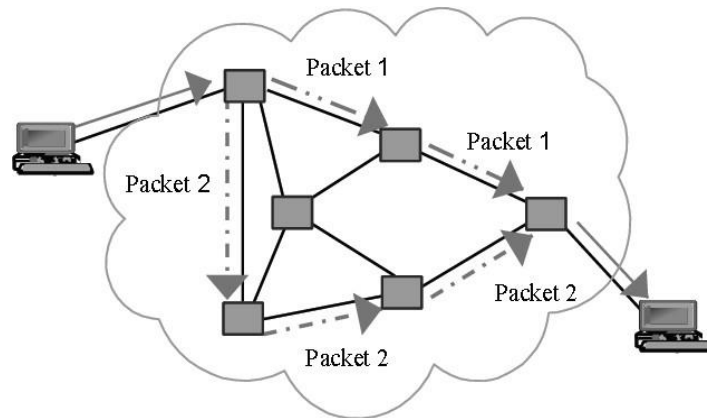
### **Types of Switching Techniques**



## **PACKET SWITCHING**

- The packet switching is a switching technique in which the message is sent in one go, but it is divided into smaller pieces, and they are sent individually.
- The message splits into smaller pieces known as packets and packets are given a unique number to identify their order at the receiving end.
- Every packet contains some information in its headers such as source address, destination address and sequence number.
- Packets will travel across the network, taking the shortest path as possible.
- All the packets are reassembled at the receiving end in correct order.

- If any packet is missing or corrupted, then the message will be sent to resend the message.
- If the correct order of the packets is reached, then the acknowledgment message will be sent.



### ***Advantages of Packet Switching:***

- **Cost-effective**
- **Reliable**
- **Efficient**

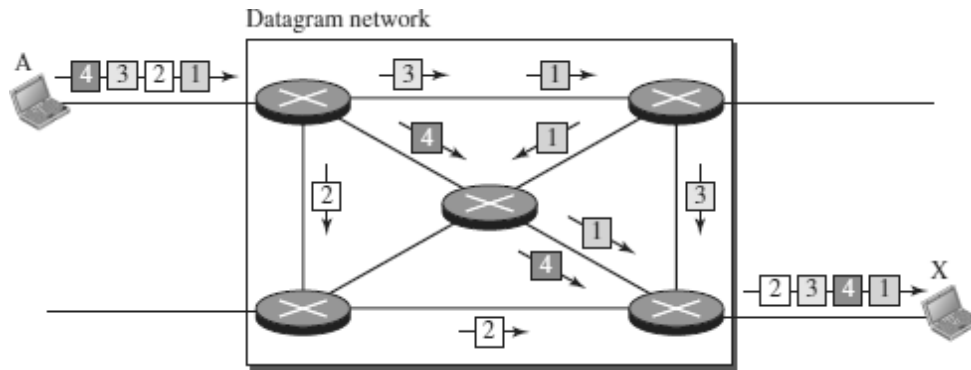
### **APPROACHES OF PACKET SWITCHING**

There are two approaches to Packet Switching:

- Datagram Packet switching (Connectionless service)
- Virtual Circuit Switching (Connection oriented service)

### **Datagram Packet switching**

- It is a packet switching technology in which packet is known as a datagram, is considered as an independent entity.
- Each packet contains the information about the destination and switch uses this information to forward the packet to the correct destination.
- The packets are reassembled at the receiving end in correct order.
- In Datagram Packet Switching technique, the path is not fixed.
- Intermediate nodes take the routing decisions to forward the packets.
- Datagram Packet Switching is also known as connectionless switching.
- There are no setup or teardown phases.
- Each packet is treated the same by a switch regardless of its source or destination.

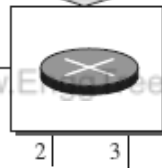


In this example, all four packets (or datagrams) belong to the same message, but may travel different paths to reach their destination.

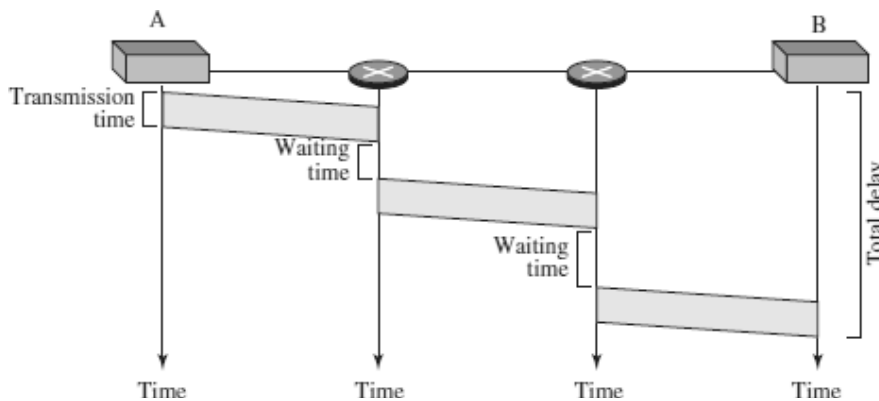
**Routing Table**

In this type of network, each switch (or packet switch) has a routing table which is based on the destination address. The routing tables are dynamic and are updated periodically. The destination addresses and the corresponding forwarding output ports are recorded in the tables.

Destination address	Output port
1232	1
4150	2
⋮	⋮
9130	3



**Delay in a datagram network**



- The packet travels through two switches.
- There are three transmission times ( $3T$ ), three propagation delays (slopes  $3t$  of the lines), and two waiting times ( $w_1 + w_2$ ).
- We ignore the processing time in each switch.

$$\text{Total delay} = 3T + 3t + w_1 + w_2$$

**Virtual Circuit Switching**

- Virtual Circuit Switching is also known as connection-oriented switching.
- In the case of Virtual circuit switching, a virtual connection is established before

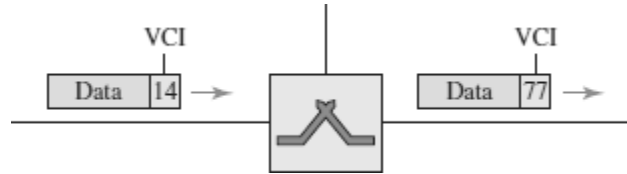


the messages are sent.

- Call request and call accept packets are used to establish the connection between sender and receiver.
- In this case, the path is fixed for the duration of a logical connection.

**Virtual Circuit Identifier (VCI)**

A virtual circuit identifier (VCI) that uniquely identifies the connection at this switch. A VCI, unlike a global address, is a small number that has only switch scope; it is used by a frame between two switches. When a frame arrives at a switch, it has a VCI; when it leaves, it has a different VCI.



**Virtual Circuit Table**

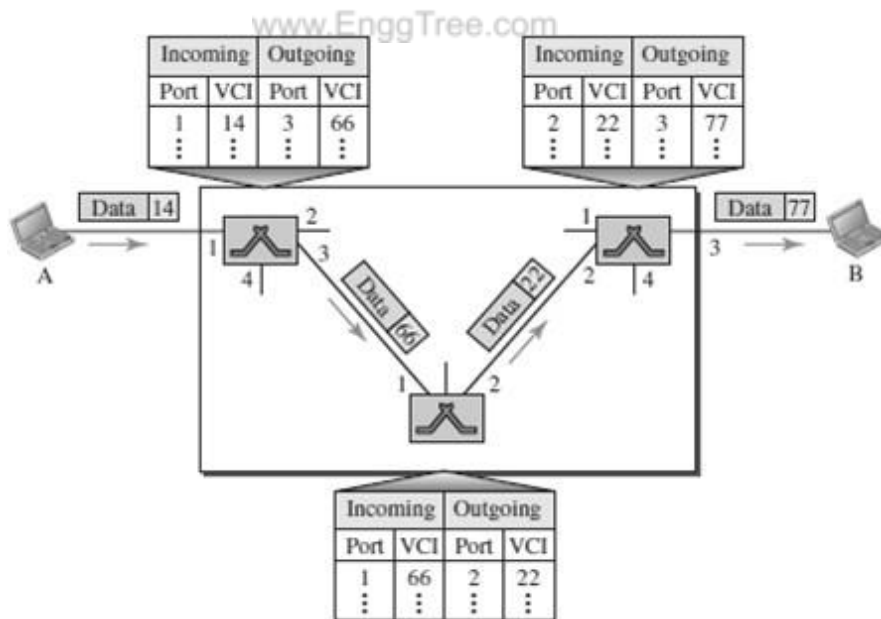
Every Virtual Circuit (VC) maintains a table called Virtual Circuit table.

One entry in the VC table on a single switch contains the following:

- An incoming interface on which packets for this VC arrive at the switch
- An outgoing interface in which packets for this VC leave the switch
- An outgoing VCI that will be used for outgoing packets

**Example:**

Source A sends a frame to Source B through Switch 1, Switch 2 and Switch 3.



## IPV4 ADDRESSES

- The identifier used in the IP layer of the TCP/IP protocol suite to identify the connection of each device to the Internet is called the Internet address or IP address.
- Internet Protocol version 4 (IPv4) is the fourth version in the development of the Internet Protocol (IP) and the first version of the protocol to be widely deployed.
- IPv4 is described in IETF publication in September 1981.
- The IP address is the address of the connection, not the host or the router. An IPv4 address is a 32-bit address that uniquely and universally defines the connection.
- If the device is moved to another network, the IP address may be changed.
- IPv4 addresses are unique in the sense that each address defines one, and only one, connection to the Internet.
- If a device has two connections to the Internet, via two networks, it has two IPv4 addresses.
- IPv4 addresses are universal in the sense that the addressing system must be accepted by any host that wants to be connected to the Internet.

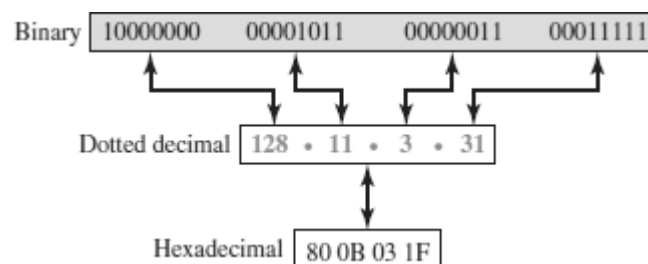
### IPV4 ADDRESS SPACE

- IPv4 defines addresses has an address space.
- An address space is the total number of addresses used by the protocol.
- If a protocol uses  $b$  bits to define an address, the address space is  $2^b$  because each bit can have two different values (0 or 1).
- IPv4 uses 32-bit addresses, which means that the address space is  $2^{32}$  or 4,294,967,296 (more than four billion).
- 4 billion devices could be connected to the Internet.

### IPV4 ADDRESS NOTATION

There are three common notations to show an IPv4 address:

- (i) binary notation (base 2),
- (ii) dotted-decimal notation (base 256), and
- (ii) hexadecimal notation (base 16).



In *binary notation*, an IPv4 address is displayed as 32 bits. To make the address more readable, one or more spaces are usually inserted between bytes (8 bits).

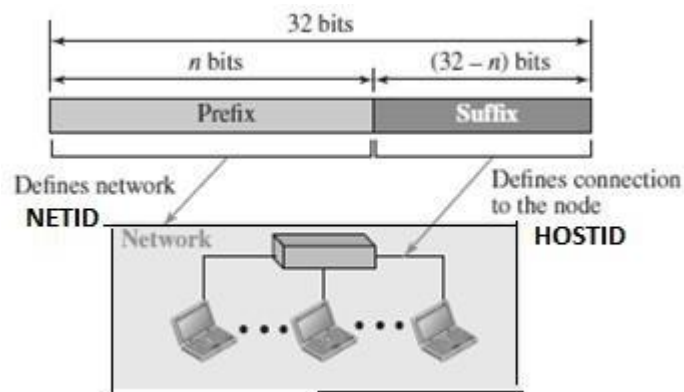
In *dotted-decimal notation*, IPv4 addresses are usually written in decimal form with a decimal point (dot) separating the bytes. Each number in the dotted-decimal notation

is between 0 and 255.

In hexadecimal notation, each hexadecimal digit is equivalent to four bits. This means that a 32-bit address has 8 hexadecimal digits. This notation is often used in network programming.

### HIERARCHY IN IPV4 ADDRESSING

- In any communication network that involves delivery, the addressing system is hierarchical.
  - A 32-bit IPv4 address is also hierarchical, but divided only into two parts.
- The first part of the address, called the *prefix*, defines the network (Net ID); the second part of the address, called the *suffix*, defines the node (Host ID).
  - The prefix length is  $n$  bits and the suffix length is  $(32 - n)$  bits.



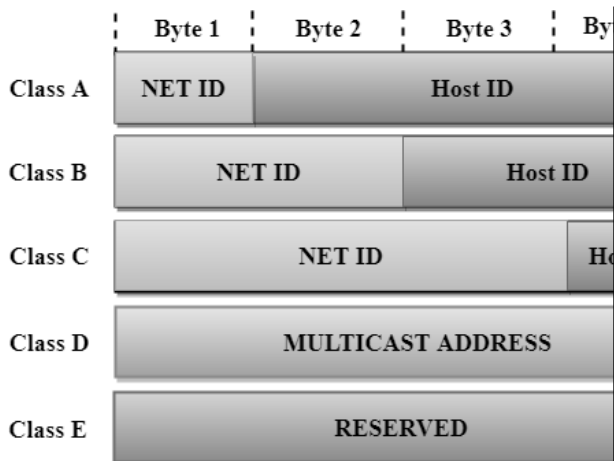
- A prefix can be fixed length or variable length.
- The network identifier in the IPv4 was first designed as a fixed-length prefix.
- This scheme is referred to as classful addressing.
- The new scheme, which is referred to as classless addressing, uses a variable-length network prefix.

### CATEGORIES OF IPV4 ADDRESSING

- There are two broad categories of IPv4 Addressing techniques.
- They are
  - Classful Addressing
  - Classless Addressing

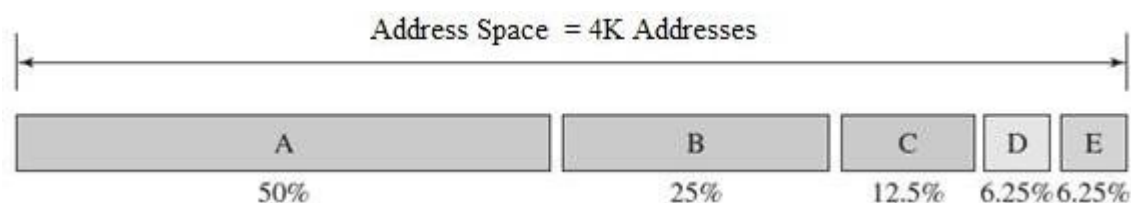
#### CLASSFUL ADDRESSING

- An IPv4 address is 32-bit long (4 bytes).
- An IPv4 address is divided into sub-classes:



Class	Prefixes	First byte
A	$n = 8$ bits	0 to 127
B	$n = 16$ bits	128 to 191
C	$n = 24$ bits	192 to 223
D	Not applicable	224 to 239
E	Not applicable	240 to 255

### Classful Network Architecture



Class	Higher bits	NET ID bits	HOST ID bits	No. of Networks	No. of hosts per network	Range
A	0	8	24	$2^7$	$2^{24}$	0.0.0.0 to 127.255.255.255
B	10	16	16	$2^{14}$	$2^{16}$	128.0.0.0 to 191.255.255.255
C	110	24	8	$2^{21}$	$2^8$	192.0.0.0 to 223.255.255.255
D	1110	Not Defined	Not Defined	Not Defined	Not Defined	224.0.0.0 to 239.255.255.255
E	1111	Not Defined	Not Defined	Not Defined	Not Defined	240.0.0.0 to 255.255.255.255

**Class A**

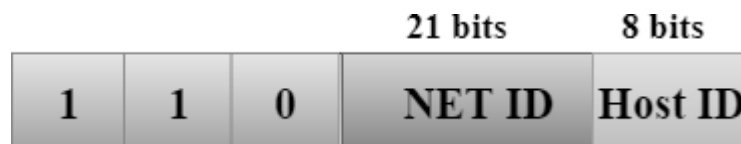
- In Class A, an IP address is assigned to those networks that contain a large number of hosts.
  - The network ID is 8 bits long.
  - The host ID is 24 bits long.
- In Class A, the first bit in higher order bits of the first octet is always set to 0 and the remaining 7 bits determine the network ID.
  - The 24 bits determine the host ID in any network.
  - The total number of networks in Class A =  $2^7 = 128$  network address
  - The total number of hosts in Class A =  $2^{24} - 2 = 16,777,214$  host address

**Class B**

- In Class B, an IP address is assigned to those networks that range from small-sized to large-sized networks.
  - The Network ID is 16 bits long.
  - The Host ID is 16 bits long.
- In Class B, the higher order bits of the first octet is always set to 01, and the remaining 14 bits determine the network ID.
  - The other 16 bits determine the Host ID.
  - The total number of networks in Class B =  $2^{14} = 16384$  network address
  - The total number of hosts in Class B =  $2^{16} - 2 = 65534$  host address

**Class C**

- In Class C, an IP address is assigned to only small-sized networks.
  - The Network ID is 24 bits long.
  - The host ID is 8 bits long.
- In Class C, the higher order bits of the first octet is always set to 110, and the remaining 21 bits determine the network ID.
  - The 8 bits of the host ID determine the host in a network.
  - The total number of networks =  $2^{21} = 2097152$  network address
  - The total number of hosts =  $2^8 - 2 = 254$  host address

**Class D**

- In Class D, an IP address is reserved for multicast addresses.
- It does not possess subnetting.

- The higher order bits of the first octet are always set to 1110, and the remaining bits determines the host ID in any network.



### **Class E**

- In Class E, an IP address is used for the future use or for the research and development purposes.
  - It does not possess any subnetting.
- The higher order bits of the first octet are always set to 1111, and the remaining bits determines the host ID in any network.



### **Address Depletion in Classful Addressing**

- The reason that classful addressing has become obsolete is address depletion.
- Since the addresses were not distributed properly, the Internet was faced with the problem of the addresses being rapidly used up.
- This results in no more addresses available for organizations and individuals that needed to be connected to the Internet.
  - To understand the problem, let us think about class A.
- This class can be assigned to only 128 organizations in the world, but each organization needs to have a single network with 16,777,216 nodes.
- Since there may be only a few organizations that are this large, most of the addresses in this class were wasted (unused).
- Class B addresses were designed for midsize organizations, but many of the addresses in this class also remained unused.
- Class C addresses have a completely different flaw in design. The number of addresses that can be used in each network (256) was so small that most companies were not comfortable using a block in this address class.
  - Class E addresses were almost never used, wasting the whole class.

### **Subnetting and Supernetting**

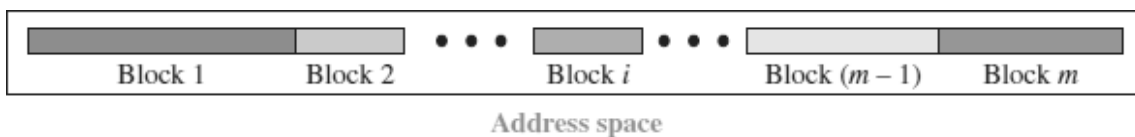
- To alleviate address depletion, two strategies were proposed and implemented:
  - (i) Subnetting      and      (ii) Supernetting.

### **Subnetting**

- In subnetting, a class A or class B block is divided into several subnets.
  - Each subnet has a larger prefix length than the original network.
- For example, if a network in class A is divided into four subnets, each subnet has a prefix of  $n_{\text{sub}} = 10$ .
- At the same time, if all of the addresses in a network are not used, subnetting allows the addresses to be divided among several organizations.

## CLASSLESS ADDRESSING

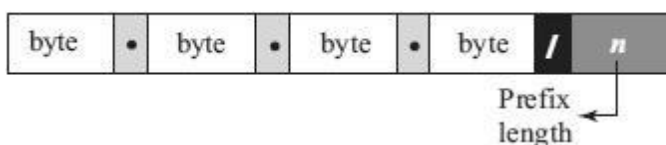
- In 1996, the Internet authorities announced a new architecture called **classless addressing**.
- In classless addressing, variable-length blocks are used that belong to no classes.
- We can have a block of 1 address, 2 addresses, 4 addresses, 128 addresses, and so on.
- In classless addressing, the whole address space is divided into variable length blocks.
- The prefix in an address defines the block (network); the suffix defines the node (device).
  - Theoretically, we can have a block of  $2^0, 2^1, 2^2, \dots, 2^{32}$  addresses.
- The number of addresses in a block needs to be a power of 2. An organization can be granted one block of addresses.



- The prefix length in classless addressing is variable.
- We can have a prefix length that ranges from 0 to 32.
- The size of the network is inversely proportional to the length of the prefix.
- A small prefix means a larger network; a large prefix means a smaller network.
- The idea of classless addressing can be easily applied to classful addressing.
- An address in class A can be thought of as a classless address in which the prefix length is 8.
- An address in class B can be thought of as a classless address in which the prefix is 16, and so on. In other words, classful addressing is a special case of classless addressing.

### Notation used in Classless Addressing

- The notation used in classless addressing is informally referred to as *slash notation* and formally as **classless interdomain routing** or **CIDR**.



Examples:  
 12.24.76.8/8  
 23.14.67.92/12  
 220.8.24.255/25

- For example, 192.168.100.14 /**24** represents the IP address 192.168.100.14 and, its subnet mask 255.255.255.0, which has 24 leading 1-bits.

### Address Aggregation

- One of the advantages of the CIDR strategy is **address aggregation** (Sometimes called *address summarization* or *route summarization*).
- When blocks of addresses are combined to create a larger block, routing can be done based on the prefix of the larger block.

- ICANN assigns a large block of addresses to an ISP.
- Each ISP in turn divides its assigned block into smaller subblocks and grants the subblocks to its customers.

### Special Addresses in IPv4

- There are five special addresses that are used for special purposes: *this-host* address, *limited-broadcast* address, *loopback* address, *private* addresses, and *multicast* addresses.

#### *This-host Address*

- ✓ The only address in the block **0.0.0.0/32** is called the *this-host* address.
- ✓ It is used whenever a host needs to send an IP datagram but it does not know its own address to use as the source address.

#### *Limited-broadcast Address*

- ✓ The only address in the block **255.255.255.255/32** is called the *limited-broadcast* address.
- ✓ It is used whenever a router or a host needs to send a datagram to all devices in a network.
- ✓ The routers in the network, however, block the packet having this address as the destination; the packet cannot travel outside the network.

#### *Loopback Address*

- ✓ The block **127.0.0.0/8** is called the *loopback* address.
- ✓ A packet with one of the addresses in this block as the destination address never leaves the host; it will remain in the host.

#### *Private Addresses*

- ✓ Four blocks are assigned as private addresses: **10.0.0.0/8**, **172.16.0.0/12**, **192.168.0.0/16**, and **169.254.0.0/16**.

#### *Multicast Addresses*

- ✓ The block **224.0.0.0/4** is reserved for multicast addresses.

## **DHCP – DYNAMIC HOST CONFIGURATION PROTOCOL**

- The dynamic host configuration protocol is used to simplify the installation and maintenance of networked computers.
- DHCP is derived from an earlier protocol called BOOTP.
- Ethernet addresses are configured into network by manufacturer and they are unique.
- IP addresses must be unique on a given internetwork but also must reflect the structure of the internetwork
- Most host Operating Systems provide a way to manually configure the IP information for the host

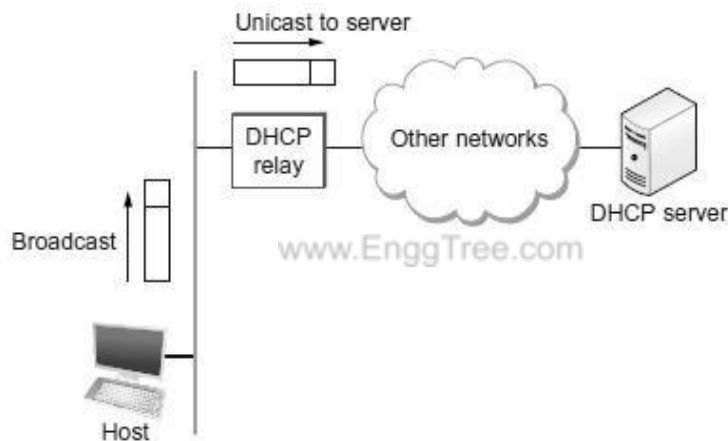
#### ➤ **Drawbacks of manual configuration:**

1. A lot of work to configure all the hosts in a large network

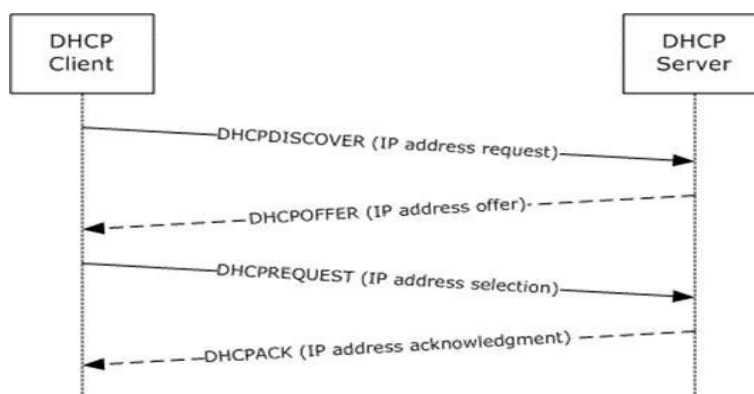


## 2. Configuration process is error-prone.

- It is necessary to ensure that every host gets the correct network number and that no two hosts receive the same IP address.
- For these reasons, automated configuration methods are required.
- The primary method uses a protocol known as the *Dynamic Host Configuration Protocol* (DHCP).
- The main goal of DHCP is to minimize the amount of manual configuration required for a host.
- If a new computer is connected to a network, DHCP can provide it with all the necessary information for full system integration into the network.
- DHCP is based on a client/server model.
- DHCP clients send a request to a DHCP server to which the server responds with an IP address
- DHCP server is responsible for providing configuration information to hosts.
- There is at least one DHCP server for an administrative domain.
- The DHCP server can function just as a centralized repository for host configuration information.
- The DHCP server maintains a pool of available addresses that it hands out to hosts on demand.



- A newly booted or attached host sends a DHCPDISCOVER message to a special IP address (255.255.255.255, which is an IP broadcast address).
- This means it will be received by all hosts and routers on that network.
- DHCP uses the concept of a *relay agent*. There is at least one relay agent on each network.
- DHCP relay agent is configured with the IP address of the DHCP server.
- When a relay agent receives a DHCPDISCOVER message, it unicasts it to the DHCP server and awaits the response, which it will then send back to the requesting client.



## DHCP Message Format

- A DHCP packet is actually sent using a protocol called the *User Datagram Protocol* (UDP).

0	8	16	24	31
Opcode	Htype	HLen	HCount	
Transaction ID				
Time elapsed		Flags		
Client IP address				
Your IP address				
Server IP address				
Gateway IP address				
Client hardware address				
Server name				
Boot file name				
Options				

Opcode: Operation code, request (1) or reply (2)  
 Htype: Hardware type (Ethernet, ...)  
 HLen: Length of hardware address  
 HCount: Maximum number of hops the packet can travel  
 Transaction ID: An integer set by the client and repeated by the server  
 Time elapsed: The number of seconds since the client started to boot  
 Flags: First bit defines unicast (0) or multicast (1); other 15 bits not used  
 Client IP address: Set to 0 if the client does not know it  
 Your IP address: The client IP address sent by the server  
 Server IP address: A broadcast IP address if client does not know it  
 Gateway IP address: The address of default router  
 Server name: A 64-byte domain name of the server  
 Boot file name: A 128-byte file name holding extra information  
 Options: A 64-byte field with dual purpose described in text

## INTERNET PROTOCOL

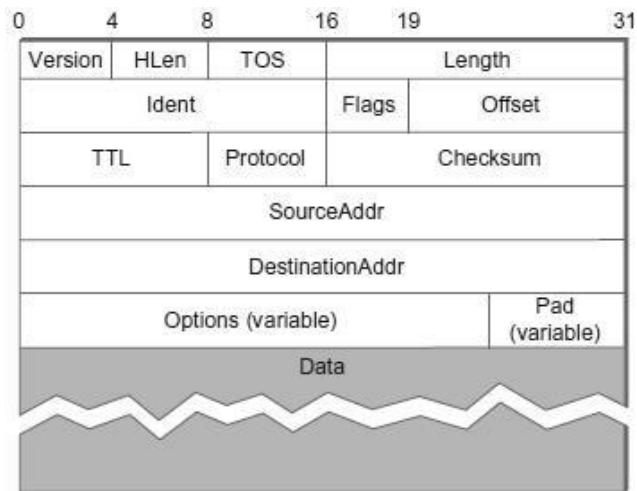
- The Internet Protocol is the key tool used today to build scalable, heterogeneous internetworks.
- IP runs on all the nodes (both hosts and routers) in a collection of networks
- IP defines the infrastructure that allows these nodes and networks to function as a single logical internetwork.

### IP SERVICE MODEL

- Service Model defines the host-to-host services that we want to provide
- The main concern in defining a service model for an internetwork is that we can provide a host-to-host service only if this service can somehow be provided over each of the underlying physical networks.
- The Internet Protocol is the key tool used today to build scalable, heterogeneous internetworks.
- The **IP service model** can be thought of as having **two parts**:
  - A **GLOBAL ADDRESSING SCHEME** - which provides a way to identify all hosts in the internetwork
  - A **DATAGRAM DELIVERY MODEL** - A connectionless model of data delivery.

## IP PACKET FORMAT / IP DATAGRAM FORMAT

- A key part of the IP service model is the type of packets that can be carried.
- The IP datagram consists of a header followed by a number of bytes of data.



FIELD	DESCRIPTION
<b>Version</b>	Specifies the version of IP. Two versions exist – IPv4 and IPv6.
<b>HLen</b>	Specifies the length of the header
<b>TOS</b> (Type of Service)	An indication of the parameters of the quality of service desired such as Precedence, Delay, Throughput and Reliability.
<b>Length</b>	Length of the entire datagram, including the header. The maximum size of an IP datagram is $65,535(2^{16})$ bytes
<b>Ident</b> (Identification)	Uniquely identifies the packet sequence number. Used for fragmentation and re-assembly.

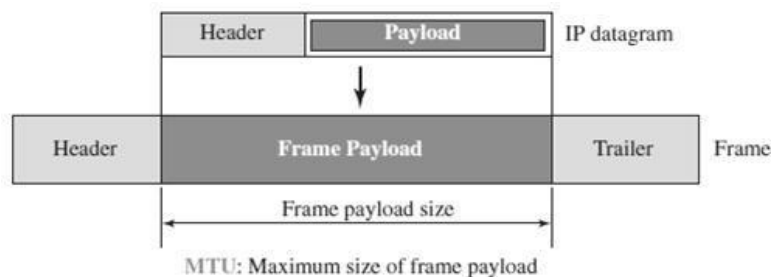
<b>Flags</b>	Used to control whether routers are allowed to fragment a packet. If a packet is fragmented, this flag value is 1. If not, flag value is 0.
<b>Offset (Fragmentation offset)</b>	Indicates where in the datagram, this fragment belongs. The fragment offset is measured in units of 8 octets (64 bits). The first fragment has offset zero.
<b>TTL (Time to Live)</b>	Indicates the maximum time the datagram is allowed to remain in the network. If this field contains the value zero, then the datagram must be destroyed.
<b>Protocol</b>	Indicates the next level protocol used in the data portion of the datagram
<b>Checksum</b>	Used to detect the processing errors introduced into the packet
<b>Source Address</b>	The IP address of the original sender of the packet.
<b>Destination Address</b>	The IP address of the final destination of the packet.
<b>Options</b>	This is optional field. These options may contain values for options such as Security, Record Route, Time Stamp, etc
<b>Pad</b>	Used to ensure that the internet header ends on a 32-bit boundary. The padding is zero.

## IP DATAGRAM - FRAGMENTATION AND REASSEMBLY

### Fragmentation:

www.EnggTree.com

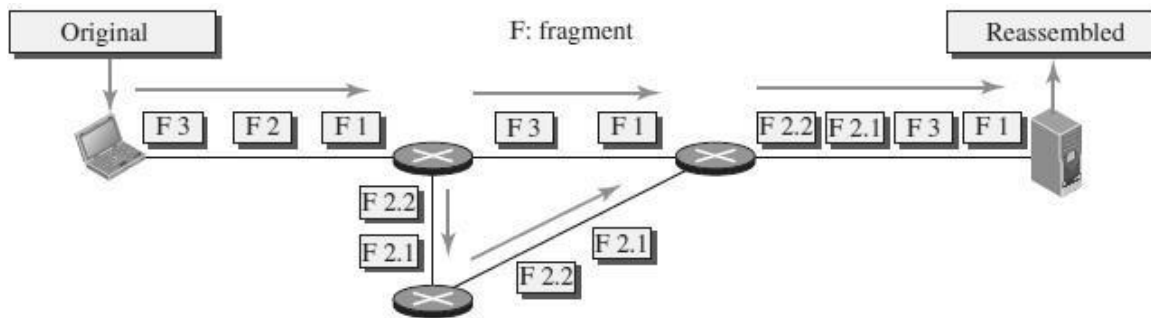
- Every network type has a *maximum transmission unit* (MTU), which is the largest IP datagram that it can carry in a frame.



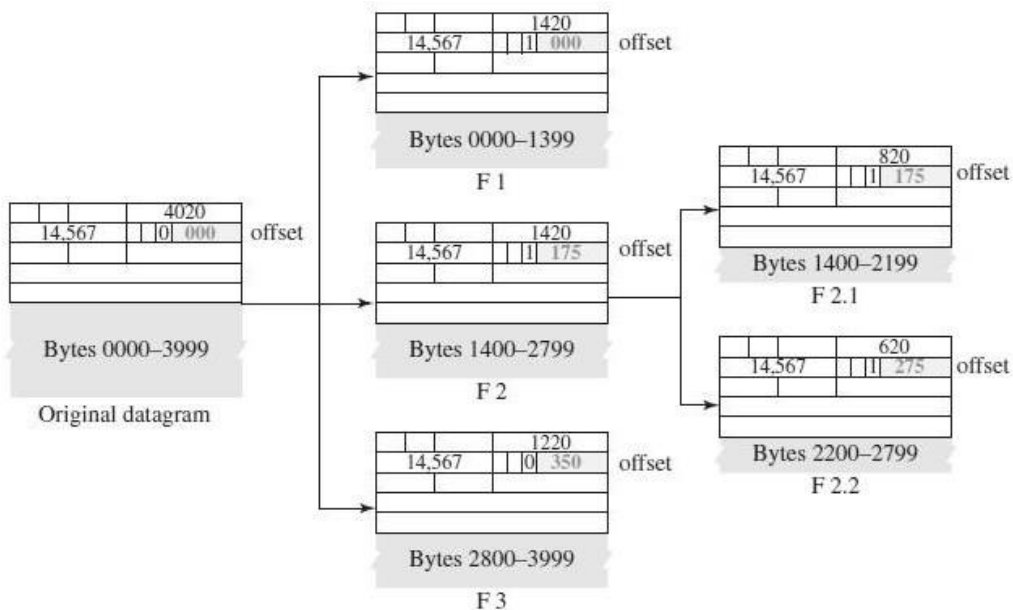
- Fragmentation of a datagram will only be necessary if the path to the destination includes a network with a smaller MTU.
- When a host sends an IP datagram, it can choose any size that it wants.
- Fragmentation typically occurs in a router when it receives a datagram that it wants to forward over a network that has an MTU that is smaller than the received datagram.
- Each fragment is itself a self-contained IP datagram that is transmitted over a sequence of physical networks, independent of the other fragments.
- Each IP datagram is re-encapsulated for each physical network over which it travels.

- For example, if we consider an Ethernet network to accept packets up to 1500 bytes long.
- This leaves two choices for the IP service model:
  - Make sure that all IP datagrams are small enough to fit inside one packet on any network technology
  - Provide a means by which packets can be fragmented and reassembled when they are too big to go over a given network technology.
- Fragmentation produces smaller, valid IP datagrams that can be readily reassembled into the original datagram upon receipt, independent of the order of their arrival.

**Example:**



- The original packet starts at the client; the fragments are reassembled at the server.
- The value of the identification field is the same in all fragments, as is the value of the flags field with the more bit set for all fragments except the last.
- Also, the value of the offset field for each fragment is shown.
- Although the fragments arrived out of order at the destination, they can be correctly reassembled.



- The value of the offset field is always relative to the original datagram.
- Even if each fragment follows a different path and arrives out of order, the final destination host can reassemble the original datagram from the fragments received (if none of them is lost) using the following strategy:
  - 1) The first fragment has an offset field value of zero.
  - 2) Divide the length of the first fragment by 8. The second fragment has an offset value equal to that result.
  - 3) Divide the total length of the first and second fragment by 8. The third fragment has an offset value equal to that result.
  - 4) Continue the process. The last fragment has its M bit set to 0.
  - 5) Continue the process. The last fragment has a *more* bit value of 0.

### **Reassembly:**

- Reassembly is done at the receiving host and not at each router.
- To enable these fragments to be reassembled at the receiving host, they all carry the same identifier in the Ident field.
- This identifier is chosen by the sending host and is intended to be unique among all the datagrams that might arrive at the destination from this source over some reasonable time period.
- Since all fragments of the original datagram contain this identifier, the reassembling host will be able to recognize those fragments that go together.
- For example, if a single fragment is lost, the receiver will still attempt to reassemble the datagram, and it will eventually give up and have to garbage-collect the resources that were used to perform the failed reassembly.
- Hosts are now strongly encouraged to perform “path MTU discovery,” a process by which fragmentation is avoided by sending packets that are small enough to traverse the link with the smallest MTU in the path from sender to receiver.

### **IP SECURITY**

There are three security issues that are particularly applicable to the IP protocol:

- (1) Packet Sniffing (2) Packet Modification and (3) IP Spoofing.

#### **Packet Sniffing**

- An intruder may intercept an IP packet and make a copy of it.
- Packet sniffing is a passive attack, in which the attacker does not change the contents of the packet.
- This type of attack is very difficult to detect because the sender and the receiver may never know that the packet has been copied.
- Although packet sniffing cannot be stopped, encryption of the packet can make the attacker’s effort useless.
- The attacker may still sniff the packet, but the content is not detectable.

#### **Packet Modification**

- The second type of attack is to modify the packet.
- The attacker intercepts the packet, changes its contents, and sends the newpacket to the receiver.
- The receiver believes that the packet is coming from the original sender.

- This type of attack can be detected using a data integrity mechanism.
- The receiver, before opening and using the contents of the message, can use this mechanism to make sure that the packet has not been changed during the transmission.

### **IP Spoofing**

- An attacker can masquerade as somebody else and create an IP packet that carries the source address of another computer.
- An attacker can send an IP packet to a bank pretending that it is coming from one of the customers.
- This type of attack can be prevented using an origin authentication mechanism

### **IP Sec**

- The IP packets today can be protected from the previously mentioned attacks using a protocol called IPsec (IP Security).
- This protocol is used in conjunction with the IP protocol.
- IPsec protocol creates a connection-oriented service between two entities in which they can exchange IP packets without worrying about the three attacks such as Packet Sniffing, Packet Modification and IP Spoofing.
- IP Sec provides the following four services:
  - 1) **Defining Algorithms and Keys:** The two entities that want to create a secure channel between themselves can agree on some available algorithms and keys to be used for security purposes.
  - 2) **Packet Encryption:** The packets exchanged between two parties can be encrypted for privacy using one of the encryption algorithms and a shared key agreed upon in the first step. This makes the packet sniffing attack useless.
  - 3) **Data Integrity:** Data integrity guarantees that the packet is not modified during the transmission. If the received packet does not pass the data integrity test, it is discarded. This prevents the second attack, packet modification.
  - 4) **Origin Authentication:** IPsec can authenticate the origin of the packet to be sure that the packet is not created by an imposter. This can prevent IP spoofing attacks.

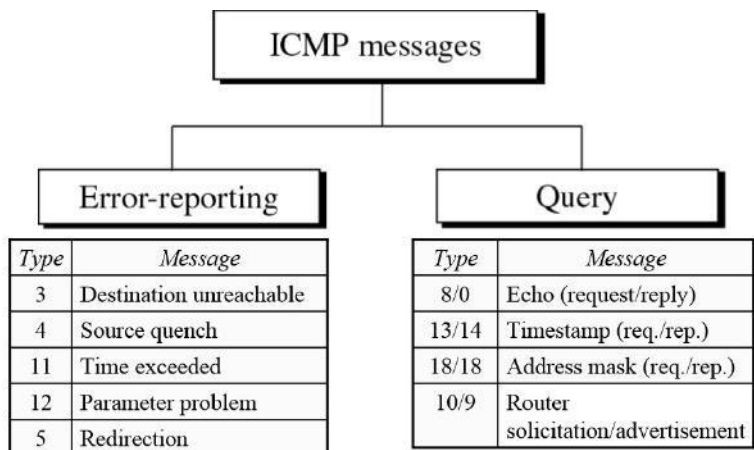
## **ICMPV4 - INTERNET CONTROL MESSAGE PROTOCOL VERSION 4**

- ICMP is a network-layer protocol.
- It is a companion to the IP protocol.
- Internet Control Message Protocol (ICMP) defines a collection of error messages that are sent back to the source host whenever a router or host is unable to process an IP datagram successfully.

### **ICMP MESSAGE TYPES**

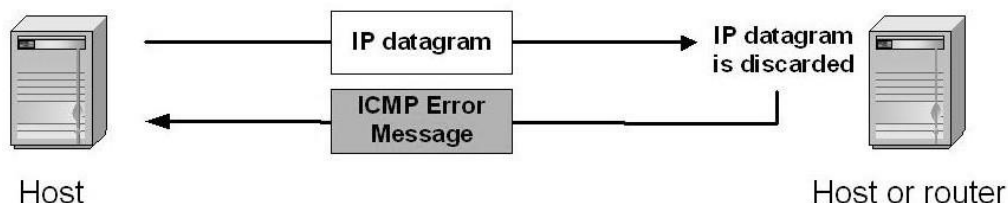
- ICMP messages are divided into two broad categories: *error-reporting messages* and *query messages*.
- The error-reporting messages report problems that a router or a host (destination) may encounter when it processes an IP packet.

- The query messages help a host or a network manager get specific information from a router or another host.



### ICMP Error – Reporting Messages

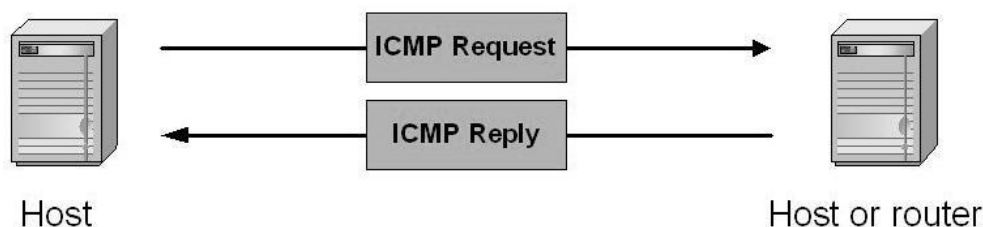
- ICMP error messages report error conditions
  - Typically sent when a datagram is discarded
  - Error message is often passed from ICMP to the application program
- **Destination Unreachable**—When a router *cannot route* a datagram, the datagram is discarded and sends a destination unreachable message to source host.
  - **Source Quench**—When a router or host discards a datagram due to *congestion*, it sends a source-quench message to the source host. This message acts as flow control.
  - **Time Exceeded**—Router discards a datagram when TTL field becomes 0 and a time exceeded message is sent to the source host.
  - **Parameter Problem**—If a router discovers ambiguous or *missing* value in any field of the datagram, it discards the datagram and sends parameter problem message to source.
  - **Redirection**—Redirect messages are sent by the default router to inform the source host to *update* its forwarding table when the packet is routed on a wrong path.



### ICMP Query Messages

- Request sent by host to a router or host
- Reply sent back to querying host

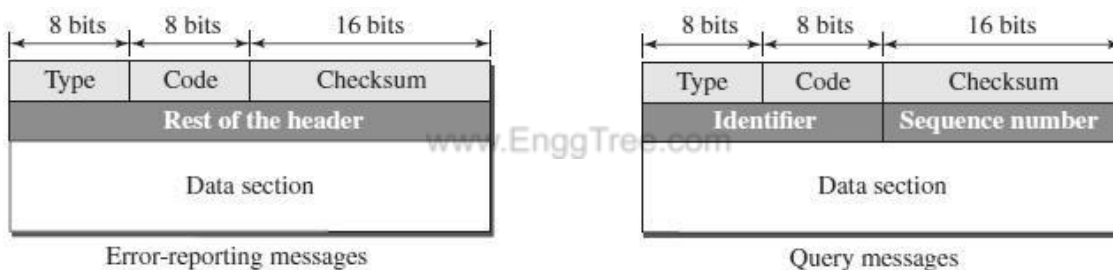




- **Echo Request & Reply**—Combination of echo request and reply messages determines whether two systems communicate or not.
- **Timestamp Request & Reply**—Two machines can use the timestamp request and reply messages to determine the round-trip time (RTT).
- **Address Mask Request & Reply**—A host to obtain its subnet mask, sends an address mask request message to the router, which responds with an address mask reply message.
- **Router Solicitation/Advertisement**—A host broadcasts a router solicitation message to know about the router. Router broadcasts its routing information with router advertisement message.

### ICMP MESSAGE FORMAT

- An ICMP message has an 8-byte header and a variable-size data section.



Type	Defines the type of the message
Code	Specifies the reason for the particular message type
Checksum	Used for error detection
Rest of the header	Specific for each message type
Data	Used to carry information
Identifier	Used to match the request with the reply
Sequence Number	Sequence Number of the ICMP packet

### ICMP DEBUGGING TOOLS

Two tools are used for debugging purpose. They are (1) Ping (2) Traceroute

#### Ping

- The *ping* program is used to find if a host is alive and responding.
- The source host sends ICMP echo-request messages; the destination, if alive, responds with ICMP echo-reply messages.

- The *ping* program sets the identifier field in the echo-request and echo-reply message and starts the sequence number from 0; this number is incremented by 1 each time a new message is sent.
- The *ping program* can calculate the round-trip time.
- It inserts the sending time in the data section of the message.
- When the packet arrives, it subtracts the arrival time from the departure time to get the round-trip time (RTT).

**\$ ping google.com**

### **Traceroute or Tracert**

- The *traceroute* program in UNIX or *tracert* in Windows can be used to trace the path of a packet from a source to the destination.
- It can find the IP addresses of all the routers that are visited along the path.
- The program is usually set to check for the maximum of 30 hops (routers) to be visited.
- The number of hops in the Internet is normally less than this.

**\$ traceroute google.com**

## **IPV6 - NEXT GENERATION IP**

- IPv6 was evolved to solve address space problem and offers rich set of services.
- Some hosts and routers will run IPv4 only, some will run IPv4 and IPv6 and some will run IPv6 only.

### **FEATURES OF IPV6**

1. **Better header format** - IPv6 uses a new header format in which options are separated from the base header and inserted, when needed, between the base header and the data. This simplifies and speeds up the routing process because most of the options do not need to be checked by routers.
2. **New options** - IPv6 has new options to allow for additional functionalities.
3. **Allowance for extension** - IPv6 is designed to allow the extension of the protocol if required by new technologies or applications.
4. **Support for resource allocation** - In IPv6, the type-of-service field has been removed, but two new fields, traffic class and flow label, have been added to enable the source to request special handling of the packet. This mechanism can be used to support traffic such as real-time audio and video.

#### **Additional Features:**

1. Need to accommodate scalable routing and addressing
2. Support for real-time services
3. Security support
4. Autoconfiguration

The ability of hosts to automatically configure themselves with such information as their own IP address and domain name.

5. Enhanced routing functionality, including support for mobile hosts

## 6. Transition from ipv4 to ipv6

**ADDRESS SPACE ALLOCATION OF IPV6**

- IPv6 provides a 128-bit address space to handle up to  $2^{128}$  nodes.
- IPv6 uses *classless* addressing, but classification is based on MSBs.
- The address space is subdivided in various ways based on the leading bits.
- The current assignment of prefixes is listed in Table

Prefix	Use
00...0 (128 bits)	Unspecified
00...1 (128 bits)	Loopback
1111 1111	Multicast addresses
1111 1110 10	Link-local unicast
Everything else	Global Unicast Addresses

- A node may be assigned an “IPv4-compatible IPv6 address” by zero-extending a 32-bit IPv4 address to 128 bits.
- A node that is only capable of understanding IPv4 can be assigned an “IPv4-mapped IPv6 address” by prefixing the 32-bit IPv4 address with 2 bytes of all 1s and then zero-extending the result to 128 bits.

**ADDRESS NOTATION OF IPV6**

- Standard representation of IPv6 address is  $x : x : x : x : x : x : x : x$  where  $x$  is a 16-bit hexadecimal address separated by colon (:).

For example,

47CD : 1234 : 4422 : ACO2 : 0022 : 1234 : A456 : 0124

Discards leading Zeros

Ex: 47CD : 1234 : 4422 : ACO2 : 0022 : 1234 : A456 : 0124

To

47CD : 1234 : 4422 : ACO2 : 22 : 1234 : A456 : 0124

Consecutive Zeros can be replaced by colon

For example,

47CD : 0000 : 0000 : 0000 : 0000 : 0000 : A456 : 0124 → 47CD :: A456 : 0124

**ADDRESS AGGREGATION OF IPV6**

- IPv6 provides *aggregation* of routing information to reduce the burden on routers.
- Aggregation is done by assigning prefixes at *continental* level.
- For *example*, if all addresses in Europe have a common prefix, then routers in other continents would need one routing table entry for all networks in Europe.

3	m	n	o	p	125-m-n-o-p
010	RegistryID	ProviderID	SubscriberID	SubnetID	InterfaceID



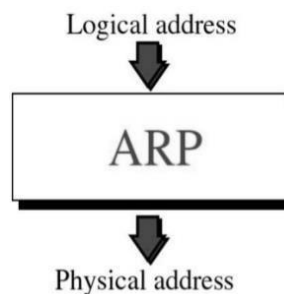
- ❖ **Fragmentation** — In IPv6, only the source host can fragment. Source uses a path MTU discovery technique to find smallest MTU on the path.
- ❖ **Authentication** — used to validate the sender and ensures data integrity.
- ❖ **ESP (Encrypted Security Payload)** — provides confidentiality against eavesdropping.

### ADVANCED CAPABILITIES OF IPV6

- ❑ **Auto Configuration** — Auto or stateless configuration of IP address to hosts without the need for a DHCP server, i.e., plug and play.
- ❑ **Advanced Routing** — Enhanced routing support for mobile hosts is provided.
- ❑ **Additional Functions** — Enhanced routing functionality with support for mobile hosts.
- ❑ **Security** — Encryption and authentication options provide confidentiality and integrity.
- ❑ **Resource allocation** — Flow label enables the source to request special handling of real-time audio and video packets

### ADDRESS RESOLUTION PROTOCOL(ARP)

- ARP stands for Address Resolution Protocol.
- ARP is the most important protocol of the Data Link Layer.
- ARP is a network layer protocol used to **convert a IP address (Network/Logical address) into a MAC Address (Hardware /Physical address)**.

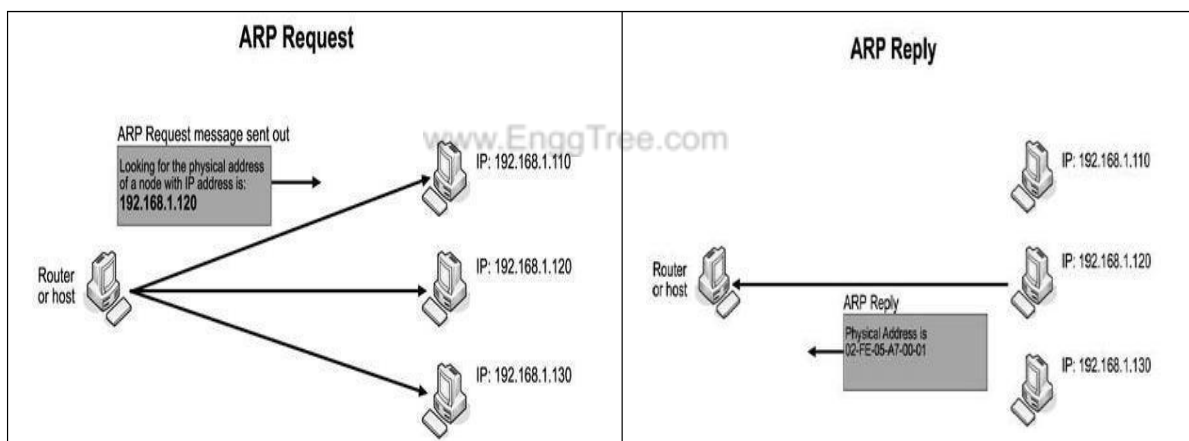


- The computer programs/applications use logical address (IP address) to send/receive messages, however the actual communication happens over the physical address (MAC address).
- To send a datagram over a network, we need both the logical and physical address.
- IP addresses are made up of 32 bits whereas MAC addresses are made up of 48 bits.
- ARP enables each host to build a table of IP address and corresponding physical address.
- ARP relies on broadcast support from physical networks.
- The Address Resolution Protocol is a request and response protocol.
- The types of ARP messages are:
  1. ARP request

## 2. ARP reply

### ARP Operation

- ARP maintains a cache table in which MAC addresses are mapped to IP addresses.
- If a host wants to send an IP datagram to a host, it first checks for a mapping in the cache table.
- If no mapping is found, it needs to invoke the Address Resolution Protocol over the network.
- It does this by broadcasting an ARP query onto the network.
- This query contains the target IP address.
- Each host receives the query and checks to see if it matches its IP address.
- If it does match, the host sends a response message that contains its link-layer address (MAC Address) back to the originator of the query.
- The originator adds the information contained in this response to its ARP table.
- For example,
  - To determine system B's physical (MAC) address, system A broadcasts an ARP request containing B's IP address to all machines on its network.



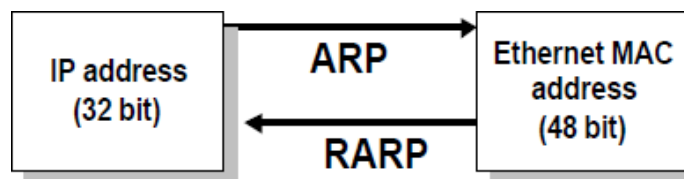
- All nodes except the destination discard the packet but update their ARP table.
- Destination host (System B) constructs an ARP Response packet
- ARP Response is unicast and sent back to the source host (System A).
- Source stores target Logical & Physical address pair in its ARP table from ARP Response.
- If target node does not exist on same network, ARP request is sent to default router.

## ARP Packet

Hardware Type		Protocol Type
Hardware length	Protocol length	Operation Request:1, Reply:2
Source hardware address		
Source protocol address		
Destination hardware address (Empty in request)		
Destination protocol address		

## RARP – Reverse ARP

- Reverse Address Resolution protocol (RARP) allows a host to convert its MAC address to the corresponding IP address.



www.EnggTree.com

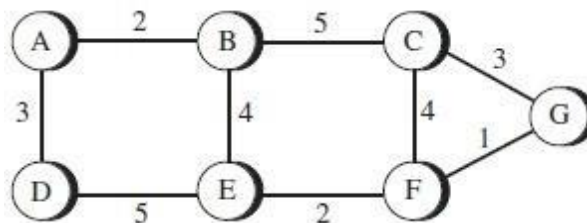
## UNIT IV – ROUTING

Routing and protocols: Unicast routing - Distance Vector Routing - RIP - Link State Routing – OSPF – Path-vector routing - BGP - Multicast Routing: DVMRP – PIM.

- Routing is the process of selecting best paths in a network.
- In unicast routing, a packet is routed, hop by hop, from its source to its destination by the help of forwarding tables.
- Routing a packet from its source to its destination means routing the packet from a *source router* (the default router of the source host) to a *destination router* (the router connected to the destination network).
- The source host needs no forwarding table because it delivers its packet to the default router in its local network.
- The destination host needs no forwarding table either because it receives the packet from its default router in its local network.
- Only the intermediate routers in the networks need forwarding tables.

### NETWORK AS A GRAPH [www.EnggTree.com](http://www.EnggTree.com)

➤ The Figure below shows a graph representing a network.



- The nodes of the graph, labeled A through G, may be hosts, switches, routers, or networks.
- The edges of the graph correspond to the network links.
- Each edge has an associated *cost*.
- The basic problem of routing is to find the lowest-cost path between any two nodes, where the cost of a path equals the sum of the costs of all the edges that make up the path.
- This static approach has several problems:
  - ❖ It does not deal with node or link failures.
  - ❖ It does not consider the addition of new nodes or links.
  - ❖ It implies that edge costs cannot change.
- For these reasons, routing is achieved by running routing protocols among the



nodes.

- These protocols provide a distributed, dynamic way to solve the problem of finding the lowest-cost path in the presence of link and node failures and changing edge costs.

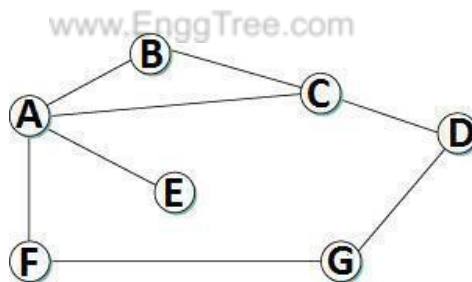
### UNICAST ROUTING ALGORITHMS

- There are three main classes of routing protocols:
  - 1) **Distance Vector Routing Algorithm – Routing Information Protocol**
  - 2) **Link State Routing Algorithm – Open Shortest Path First Protocol**
  - 3) **Path-Vector Routing Algorithm - Border Gateway Protocol**

#### DISTANCE VECTOR ROUTING (DSR) ROUTING INFORMATION PROTOCOL (RIP) BELLMAN - FORD ALGORITHM

- Distance vector routing is *distributed*, i.e., algorithm is run on all nodes.
- Each node *knows* the distance (cost) to each of its directly connected neighbors.
- Nodes construct a *vector* (Destination, Cost, NextHop) and distributes to its neighbors.
- Nodes compute routing table of *minimum* distance to every other node via NextHop using information obtained from its neighbors.

#### Initial State



- In given network, *cost* of each link is 1 hop.
- Each node sets a distance of 1 (hop) to its *immediate* neighbor and cost to itself as 0.
- Distance for non-neighbors is marked as *unreachable* with value  $\infty$  (infinity).
- For node A, nodes B, C, E and F are *reachable*, whereas nodes D and G are *unreachable*.

Destination	Cost	NextHop
A	0	A
B	1	B
C	1	C
D	$\infty$	—
E	1	E
F	1	F
G	$\infty$	—

Node A's initial table

Destination	Cost	NextHop
A	1	A
B	1	B
C	0	C
D	1	D
E	$\infty$	—
F	$\infty$	—
G	$\infty$	—

Node C's initial table

Destination	Cost	NextHop
A	1	A
B	$\infty$	—
C	$\infty$	—
D	$\infty$	—
E	$\infty$	—
F	0	F
G	1	G

Node F's initial table

- The initial table for all the nodes are given below

Initial Distances Stored at Each Node (Global View)							
Information Stored at Node	Distance to Reach Node						
	A	B	C	D	E	F	G
A	0	1	1	∞	1	1	∞
B	1	0	1	∞	∞	∞	∞
C	1	1	0	1	∞	∞	∞
D	∞	∞	1	0	∞	∞	1
E	1	∞	∞	∞	0	∞	∞
F	1	∞	∞	∞	∞	0	1
G	∞	∞	∞	1	∞	1	0

- Each node *sends* its initial table (distance vector) to neighbors and receives their estimate.
- Node A sends its table to nodes B, C, E & F and receives tables from nodes B, C, E & F.
- Each node *updates* its routing table by comparing with each of its neighbor's table
- For each destination, Total Cost is computed as:
- **Total Cost** = Cost (Node to Neighbor) + Cost (Neighbor to Destination)
- If Total Cost < Cost then
- **Cost** = Total Cost and NextHop = Neighbor
- Node A *learns* from C's table to reach node D and from F's table to reach node G.
- Total Cost to reach node D via C = Cost (A to C) + Cost(C to D)  
Cost = 1 + 1 = 2.
- Since  $2 < \infty$ , entry for destination D in A's table is changed to (D, 2, C)
  - Total Cost to reach node G via F = Cost(A to F) + Cost(F to G) = 1 + 1 = 2
  - Since  $2 < \infty$ , entry for destination G in A's table is changed to (G, 2, F)
- Each node builds *complete* routing table after few exchanges amongst its neighbors.

*Node A's final routing table*

Destination	Cost	NextHop
A	0	A
B	1	B
C	1	C
D	2	C
E	1	E
F	1	F
G	2	F

- System stabilizes when all nodes have complete routing information, i.e., **convergence**.
- Routing tables are exchanged *periodically or in case of triggered update*.
- The final distances stored at each node is given below:

Final Distances Stored at Each Node (Global View)							
Information Stored at Node	Distance to Reach Node						
	A	B	C	D	E	F	G
A	0	1	1	2	1	1	2
B	1	0	1	2	2	2	3
C	1	1	0	1	2	2	2
D	2	2	1	0	3	2	1
E	1	2	2	3	0	2	3
F	1	2	2	2	2	0	1
G	2	3	2	1	3	1	0

### Updation of Routing Tables

There are two different circumstances under which a given node decides to send a routing update to its neighbors.

#### Periodic Update

- In this case, each node automatically sends an update message every so often, even if nothing has changed.
- The frequency of these periodic updates varies from protocol to protocol, but it is typically on the order of several seconds to several minutes.

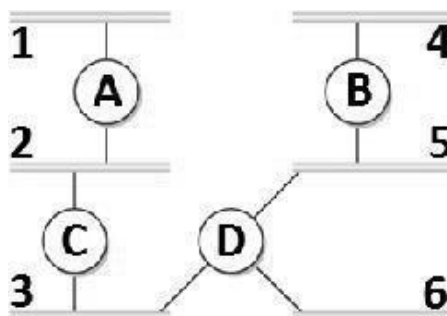
#### Triggered Update

- In this case, whenever a node notices a link failure or receives an update from one of its neighbors that causes it to change one of the routes in its routing table.
- Whenever a node's routing table changes, it sends an update to its neighbors, which may lead to a change in their tables, causing them to send an update to their neighbors.

### **ROUTING INFORMATION PROTOCOL (RIP)**

- RIP is an intra-domain routing protocol based on distance-vector algorithm.

#### **Example**



- Routers *advertise* the cost of reaching networks. Cost of reaching each link is 1 hop. For example, router *C* advertises to *A* that it can reach network 2, 3 at cost 0 (directly connected), networks 5, 6 at cost 1 and network 4 at cost 2.
- Each router *updates* cost and next hop for each network number.
- Infinity is defined as 16, i.e., any route cannot have more than 15 hops. Therefore RIP can be implemented on small-sized networks only.
- Advertisements are sent every 30 seconds or in case of triggered update.

0	7	15	31
command		version	must be zero
address family identifier		must be zero	
IP address			
must be zero			
must be zero			
metric			

- **Command** - It indicates the packet type.  
Value 1 represents a request packet. Value 2 represents a response packet.
- **Version** - It indicates the RIP version number. For RIPv1, the value is 0x01.
- **Address Family Identifier** - When the value is 2, it represents the IP protocol.
- **IP Address** - It indicates the destination IP address of the route. It can be the addresses of only the natural network segment.
- **Metric** - It indicates the hop count of a route to its destination.

### Count-To-Infinity (or) Loop Instability Problem

- Suppose link from node *A* to *E* goes *down*.
  - ❖Node *A* advertises a distance of  $\infty$  to *E* to its neighbors
  - ❖Node *B* receives periodic update from *C* before *A*'s update reaches *B*
  - ❖Node *B* updated by *C*, concludes that *E* can be reached in 3 hops via *C*
  - ❖Node *B* advertises to *A* as 3 hops to reach *E*
  - ❖Node *A* in turn updates *C* with a distance of 4 hops to *E* and so on
- Thus, nodes update each other until cost to *E* reaches *infinity*, i.e., *no convergence*.
- Routing table does not stabilize.
- This problem is called *loop instability* or *count to infinity*

### **Solution to Count-To-Infinity (or) Loop Instability Problem:**

- Infinity* is redefined to a small number, say 16.
- Distance between any two nodes can be 15 hops maximum. Thus, distancevector routing *cannot be used* in large networks.
- When a node updates its neighbors, it does not send those routes it learned

from each neighbor back to that neighbor. This is known as **split horizon**.

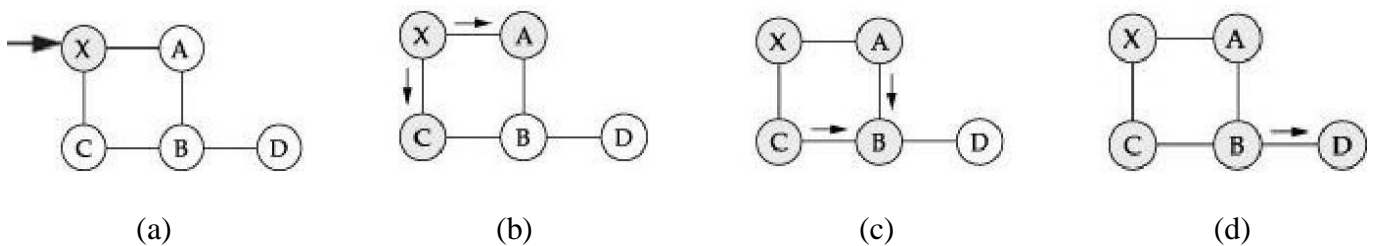
- **Split horizon with poison reverse** allows nodes to advertise routes it learnt from a node back to that node, but with a warning message.

## LINK STATE ROUTING (LSR) OPEN SHORTEST PATH PROTOCOL (OSPF) DIJKSTRA'S ALGORITHM

- Each node knows *state* of link to its neighbors and *cost*.
- Nodes create an update packet called *link-state packet* (LSP) that contains:
  - ID of the node
  - List of neighbors for that node and associated cost
  - 64-bit Sequence number
  - Time to live
- Link-State routing protocols rely on two mechanisms:
  - **Reliable flooding** of link-state information to all other nodes
  - **Route calculation** from the accumulated link-state knowledge

### Reliable Flooding

- Each node *sends* its LSP out on each of its directly connected links.
- When a node receives LSP of another node, checks if it has an LSP already for that node.
- If not, it stores and forwards the LSP on all other links except the incoming one.
- Else if the received LSP has a *bigger* sequence number, then it is stored and forwarded. Older LSP for that node is *discarded*.
- Otherwise discard the received LSP, since it is not latest for that node.
- Thus, recent LSP of a node eventually *reaches* all nodes, i.e., *reliable flooding*.



- Flooding of LSP in a small network is as follows:
  - When node X receives Y's LSP (*fig a*), it floods onto its neighbors A and C (*fig b*)
  - Nodes A and C forward it to B, but does not send it back to X (*fig c*).
  - Node B receives two copies of LSP with same sequence number.
  - Accepts one LSP and forwards it to D (*fig d*). Flooding is complete.
- LSP is generated either *periodically* or when there is a *change* in the topology.

### Route Calculation

- Each node knows the entire topology, once it has LSP from every other node.
- Forward search algorithm is used to compute routing table from the received LSPs.
- Each node maintains two lists, namely Tentative and Confirmed with entries of the form (Destination, Cost, NextHop).

### DIJKSTRA'S SHORTEST PATH ALGORITHM (FORWARD SEARCH ALGORITHM)

1. Each host maintains two lists, known as *Tentative* and *Confirmed*
2. Initialize the Confirmed list with an entry for the Node (Cost = 0).
3. Node just added to Confirmed list is called Next. Its LSP is examined.
4. For each neighbor of Next, calculate cost to reach each neighbor as Cost (Node to Next) + Cost (Next to Neighbor).
  - a. If Neighbor is neither in Confirmed nor in Tentative list, then add (Neighbor, Cost, NextHop) to Tentative list.
  - b. If Neighbor is in Tentative list, and Cost is less than existing cost, then replace the entry with (Neighbor, Cost, NextHop).
5. If Tentative list is empty then *stop*, otherwise move *least* cost entry from Tentative list to Confirmed list. Go to *Step* 2.

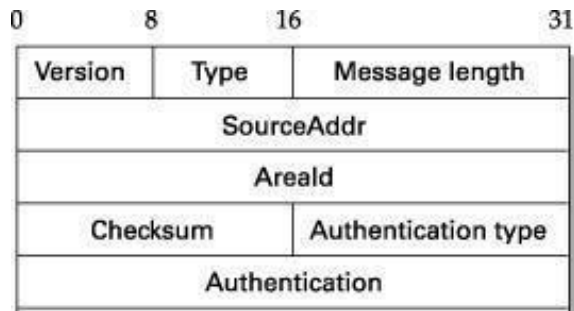
#### Example :

	Step	Confirmed	Tentative	Comments
	1	(D,0,-)		Since D is the only new member of the confirmed list, look at its LSP.
	2	(D,0,-)	(B,11,B) (C,2,C)	D's LSP says we can reach B through B at cost 11, which is better than anything else on either list, so put it on Tentative list; same for C.
	3	(D,0,-) (C,2,C)	(B,11,B)	Put lowest-cost member of Tentative (C) onto Confirmed list. Next, examine LSP of newly confirmed member (C).
	4	(D,0,-) (C,2,C)	(B,5,C) (A,12,C)	Cost to reach B through C is 5, so replace (B,11,B). C's LSP tells us that we can reach A at cost 12.
	5	(D,0,-) (C,2,C) (B,5,C)	(A,12,C)	Move lowest-cost member of Tentative (B) to Confirmed, then look at its LSP.
	6	(D,0,-) (C,2,C) (B,5,C)	(A,10,C)	Since we can reach A at cost 5 through B, replace the Tentative entry.
	7	(D,0,-) (C,2,C) (B,5,C) (A,10,C)		Move lowest-cost member of Tentative (A) to Confirmed, and we are all done.

### OPEN SHORTEST PATH FIRST PROTOCOL (OSPF)

- OSPF is a non-proprietary widely used link-state routing protocol.
- OSPF Features are:
  - **Authentication**—Malicious host can collapse a network by advertising to reach every host with cost 0. Such disasters are averted by authenticating routing updates.
  - **Additional hierarchy**—Domain is partitioned into areas, i.e., OSPF is more scalable.
  - **Load balancing**—Multiple routes to the same place are assigned same cost. Thus, traffic is distributed evenly.

## Link State Packet Format



- **Version** — represents the current version, i.e., 2.
- **Type** — represents the type (1–5) of OSPF message.
  - Type 1 - “hello” message,      Type 2 - request,      Type 3 – send,
  - Type 4 - acknowledge the receipt of link state messages,
  - Type 5 - reserved
- **SourceAddr** — identifies the sender
- **AreaId** — 32-bit identifier of the area in which the node is located
- **Checksum** — 16-bit internet checksum
- **Authentication type** — 1 (simple password), 2 (cryptographic authentication).
- **Authentication** — contains password or cryptographic checksum

www.EnggTree.com

## Difference Between Distance-Vector and Link-State Algorithms

<i>Distance vector Routing</i>	<i>Link state Routing</i>
Each node talks only to its directly connected neighbors, but it tells them everything it has learned (i.e., distance to all nodes).	Each node talks to all other nodes, but it tells them only what it knows for sure (i.e., only the state of its directly connected links).

## PATH VECTOR ROUTING (PVR) BORDER GATEWAY PROTOCOL (BGP)

- Path-vector routing is an asynchronous and distributed routing algorithm.
- The Path-vector routing is not based on least-cost routing.
- The best route is determined by the source using the policy it imposes on the route.
- In other words, the source can control the path.
- Path-vector routing is not actually used in an internet, and is mostly designed to

route a packet between ISPs.

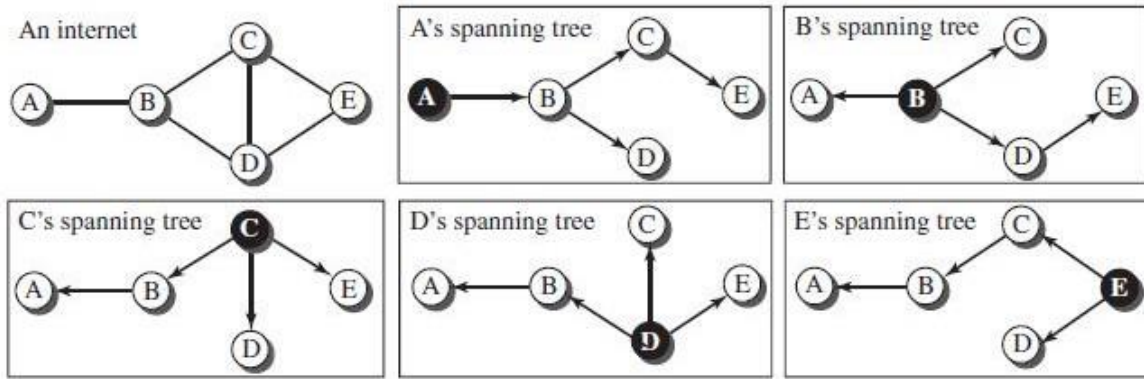
### Spanning Trees

- In path-vector routing, the path from a source to all destinations is determined by the *best* spanning tree.
- The best spanning tree is not the least-cost tree.
- It is the tree determined by the source when it imposes its own policy.
- If there is more than one route to a destination, the source can choose the route that meets its policy best.
- A source may apply several policies at the same time.
- One of the common policies uses the minimum number of nodes to be visited. Another common policy is to avoid some nodes as the middle node in a route.
- The spanning trees are made, gradually and asynchronously, by each node. When a node is booted, it creates a *path vector* based on the information it can obtain about its immediate neighbor.
- A node sends greeting messages to its immediate neighbors to collect these pieces of information.
- Each node, after the creation of the initial path vector, sends it to all its immediate neighbors.
- Each node, when it receives a path vector from a neighbor, updates its path vector using the formula
 
$$\text{Path}(x, y) = \text{best} \{ \text{Path}(x, y), [x + \text{Path}(v, y)] \} \quad \text{for all } v\text{'s in the internet.}$$
- The policy is defined by selecting the *best* of multiple paths.
- Path-vector routing also imposes one more condition on this equation.
- If Path (v, y) includes x, that path is discarded to avoid a loop in the path.
- In other words, x does not want to visit itself when it selects a path to y.

### Example:

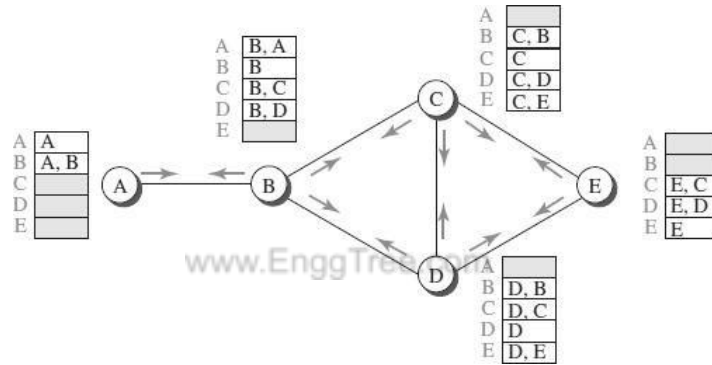
- The Figure below shows a small internet with only five nodes.
- Each source has created its own spanning tree that meets its policy.
- The policy imposed by all sources is to use the minimum number of nodes to reach a destination.
- The spanning tree selected by A and E is such that the communication does not pass through D as a middle node.
- Similarly, the spanning tree selected by B is such that the communication does not pass through C as a middle node.





### Path Vectors made at booting time

- The Figure below shows all of these path vectors for the example.
- Not all of these tables are created simultaneously.
- They are created when each node is booted.
- The figure also shows how these path vectors are sent to immediate neighbors after they have been created.



### Updating Path Vectors

- The Figure below shows the path vector of node C after two events.
- In the first event, node C receives a copy of B's vector, which improves its vector now it knows how to reach node A.
- In the second event, node C receives a copy of D's vector, which does not change its vector.
- The vector for node C after the first event is stabilized and serves as its forwarding table.

New C		Old C		B	
A	C, B, A	A		A	B, A
B	C, B	B	C, B	B	B
C	C	C	C	C	B, C
D	C, D	D	C, D	D	B, D
E	C, E	E	C, E	E	

$C[] = \text{best}(C[], C + B[])$

Event 1: C receives a copy of B's vector

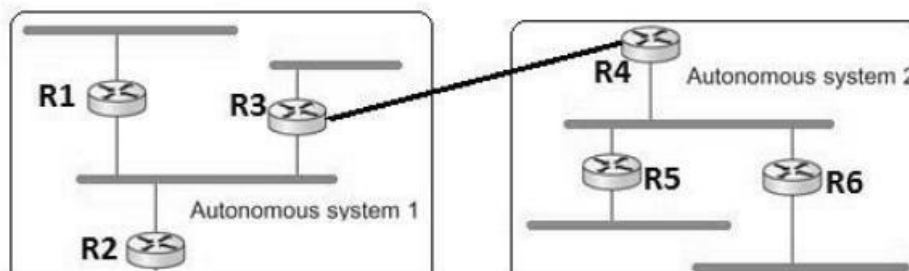
New C		Old C		D	
A	C, B, A	A	C, B, A	A	
B	C, B	B	C, B	B	D, B
C	C	C	C	C	D, C
D	C, D	D	C, D	D	D
E	C, E	E	C, E	E	D, E

$C[] = \text{best}(C[], C + D[])$

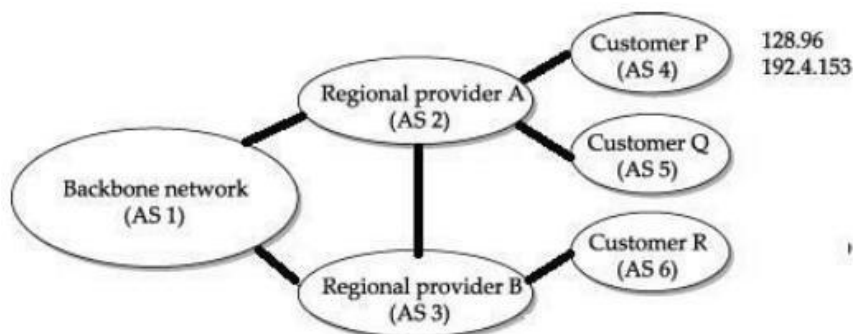
Event 2: C receives a copy of D's vector

## BORDER GATEWAY PROTOCOL (BGP)

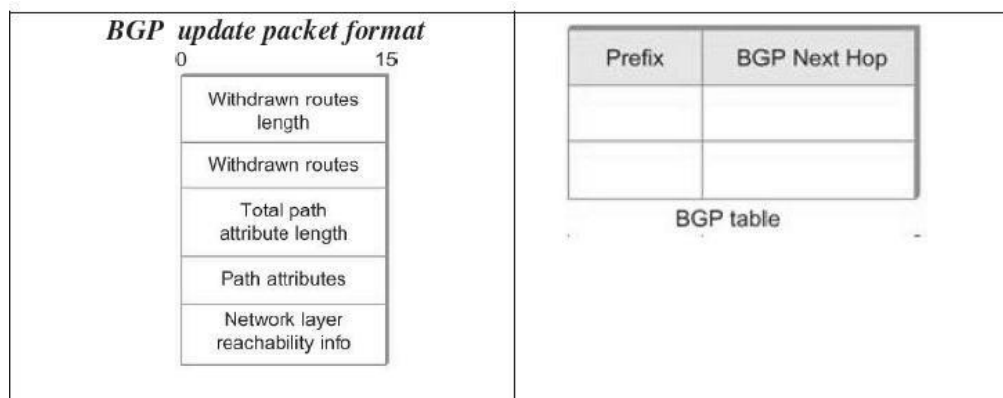
- The Border Gateway Protocol version (BGP) is the only interdomain routing protocol used in the Internet today.
- BGP4 is based on the path-vector algorithm. It provides information about the reachability of networks in the Internet.
- BGP views internet as a set of autonomous systems interconnected arbitrarily.



- Each AS have a *border router* (gateway), by which packets enter and leave that AS. In above figure, R3 and R4 are border routers.
- One of the routers in each autonomous system is designated as BGP *speaker*.
- BGP Speaker *exchange* reachability information with other BGP speakers, known as *external* BGP session.
- BGP advertises complete *path* as enumerated list of AS (path vector) to reach a particular network.
- Paths must be without any *loop*, i.e., AS list is unique.
- For *example*, backbone network advertises that network 128.96 and 192.4.153 can be reached along the path <AS1, AS2, AS4>.



- If there are *multiple* routes to a destination, BGP speaker chooses one based on policy.
- Speakers *need not* advertise any route to a destination, even if one exists.
- Advertised paths can be *cancelled*, if a link/node on the path goes down. This negative advertisement is known as *withdrawn* route.
- Routes are not repeatedly sent. If there is no change, *keep alive* messages are sent.



## iBGP - interior BGP

- 
- Used by routers to update routing information learnt from other speakers to routers inside the autonomous system.
- Each router in the AS is able to determine the appropriate next hop for all prefixes.

## MULTICAST ROUTING

- To support multicast, a router must additionally have multicast forwarding tables that indicate, based on multicast address, which links to use to forward the multicast packet.
- Unicast forwarding tables collectively specify a set of paths.
- Multicast forwarding tables collectively specify a set of trees - Multicast distribution trees.
- Multicast routing is the process by which multicast distribution trees are determined.
- To support multicasting, routers *additionally* build multicast forwarding tables.
- Multicast forwarding table is a tree structure, known as **multicast distribution trees**.
- Internet multicast is implemented on physical networks that support broadcasting by *extending* forwarding functions.

## MULTICAST DISTRIBUTION TREES

There are two types of Multicast Distribution Trees used in multicast routing.

They are

- **Source-Based Tree: (DVMRP)**
  - For each combination of (source, group), there is a shortest path spanning tree.
  - **Flood and prune**
    - Send multicast traffic everywhere
    - Prune edges that are not actively subscribed to group
  - **Link-state**
    - Routers flood groups they would like to receive
    - Compute shortest-path trees on demand
- **Shared Tree (PIM)**

- Single distributed tree shared among all sources
- Does not include its own topology discovery mechanism, but instead uses routing information supplied by other routing protocols
- Specify rendezvous point (RP) for group
- Senders send packets to RP, receivers join at RP
  
- RP multicasts to receivers; Fix-up tree for optimization
- **Rendezvous-Point Tree**: one router is the center of the group and therefore, the root of the tree.

## MULTICAST ROUTING PROTOCOLS

- Internet multicast is implemented on physical networks that support broadcasting by *extending forwarding functions*.
- Major multicast routing protocols are:
  1. Distance-Vector Multicast Routing Protocol (DVMRP)
  2. Protocol Independent Multicast (PIM)

### 1. Distance Vector Multicast Routing Protocol

- The DVMRP, is a routing protocol used to share information between routers to facilitate the transportation of IP multicast packets among networks.
- It formed the basis of the Internet's historic multicast backbone.
- Distance vector routing for unicast is extended to support multicast routing.
- Each router maintains a routing table for all destination through exchange of distance vectors. [www.EnggTree.com](http://www.EnggTree.com)
- DVMRP is also known as ***flood-and-prune protocol***.
- DVMRP consists of two major components:
  - A conventional distance-vector routing protocol, like RIP
  - A protocol for determining how to forward multicast packets, based on the routing table
- DVMRP router forwards a packet if
  - The packet arrived from the link used to reach the source of the packet
  - If downstream links have not pruned the tree
- DVMRP protocol uses the **basic packet types** as follows:

- **DVMRP Probes**
  - for DVMRP Neighbor Discovery
- **DVMRP Reports**
  - for Multicast Route Exchange
- **DVMRP Prunes**
  - for pruning multicast delivery trees
- **DVMRP Grafts**
  - for grafting multicast delivery trees
- **DVMRP Graft Ack's**
  - for acknowledging graft msgs

- The **forwarding table** of DVMRP is as follows:

<u>Source Subnet</u>	<u>Multicast Group</u>	<u>TTL</u>	<u>InPort</u>	<u>OutPorts</u>
128.1.0.0	224.1.1.1	200	1 Pr	2p 3p
	224.2.2.2	100	1	2p 3
	224.3.3.3	250	1	2
128.2.0.0	224.1.1.1	150	2	2p 3

- Multicasting is added to distance-vector routing in four stages.

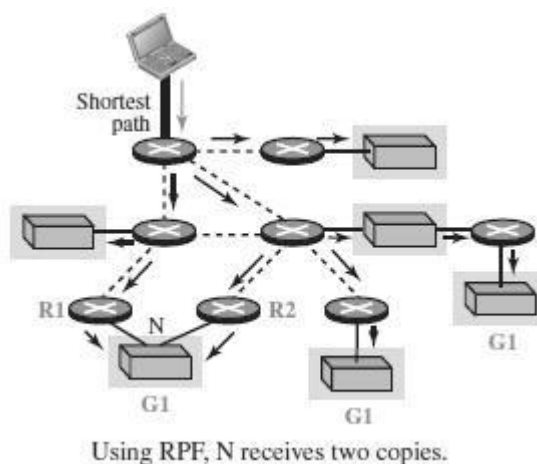
- Flooding
- Reverse Path Forwarding (RPF)
- Reverse Path Broadcasting (RPB)
- Reverse Path Multicast (RPM)

### Flooding

- ❑ Router on receiving a multicast packet from source  $S$  to a Destination from NextHop, forwards the packet on all out-going links.
- ❑ Packet is flooded and looped back to  $S$ .
- ❑ The drawbacks are:
  - It floods a network, even if it has *no members* for that group.
  - Packets are forwarded by each router connected to a LAN, i.e., *duplicate flooding*

### Reverse Path Forwarding (RPF)

- ❑ RPF eliminates the looping problem in the flooding process.
- ❑ Only one copy is forwarded and the other copies are discarded.
- ❑ RPF forces the router to forward a multicast packet from one specific interface: the one which has come through the shortest path from the source to the router.
- ❑ Packet is flooded but not looped back to  $S$ .

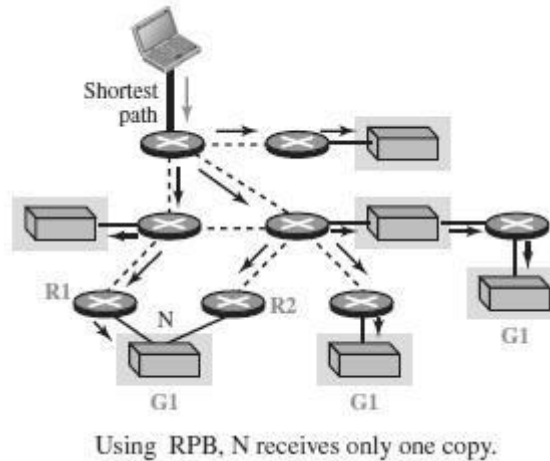


### Reverse-Path Broadcasting (RPB)

- ❑ RPB does not multicast the packet, it broadcasts it.
- ❑ RPB creates a shortest path broadcast tree from the source to each destination.
- ❑ It guarantees that each destination receives one and only one copy of the packet.
- ❑ We need to prevent each network from receiving more than one copy of the

packet.

- If a network is connected to more than one router, it may receive a copy of the packet from each router.
- One router identified as parent called designated Router (DR).
- Only parent router *forwards* multicast packets from source *S* to the attached network.
- When a router that is not the parent of the attached network receives a multicast packet, it simply drops the packet.

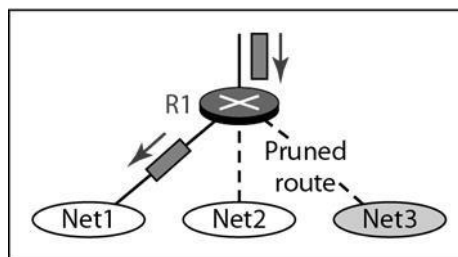


**Reverse-Path Multicasting (RPM)**

- To increase efficiency, the multicast packet must reach only those networks that have active members for that particular group.
- RPM adds pruning and grafting to RPB to create a multicast shortest path tree that supports dynamic membership changes.

**Pruning:**

- Sent from routers receiving multicast traffic for which they have no active group members
- “Prunes” the tree created by DVMRP
- Stops needless data from being sent

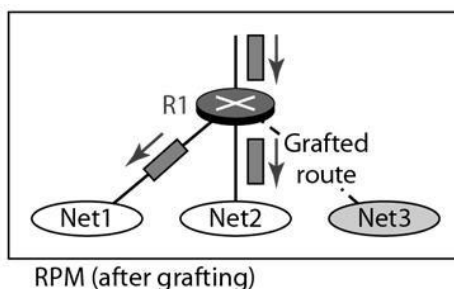


RPM (after pruning)

**Grafting:**

- Used after a branch has been pruned back
- Sent by a router that has a host that joins a multicast group
- Goes from router to router until a router active on the multicast group is reached
- Sent for the following cases
  - A new host member joins a group

- A new dependent router joins a pruned branch
- A dependent router restarts on a pruned branch



## 2. Protocol Independent Multicast (PIM)

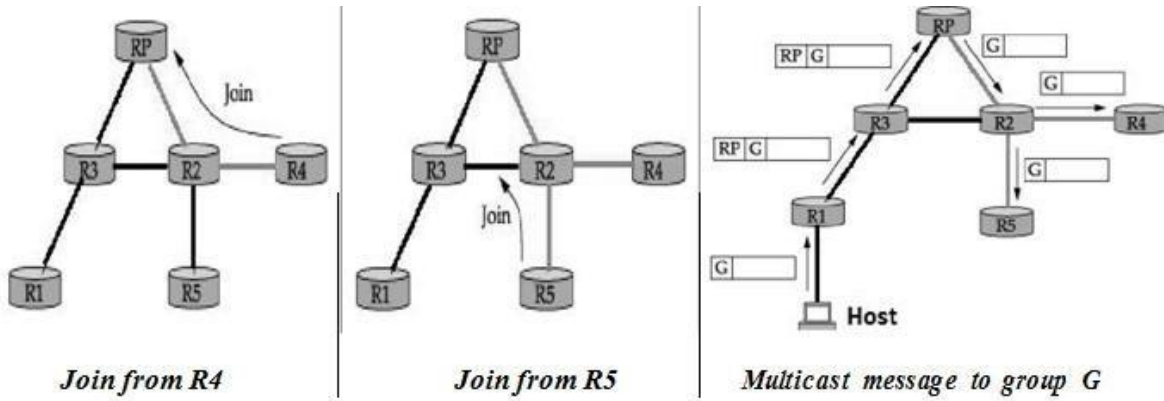
- PIM divides multicast routing problem into *sparse* and *dense* mode.
- PIM sparse mode (PIM-SM) is widely used.
- PIM does not rely on any type of unicast routing protocol, hence protocol independent.
- Routers explicitly join and leave multicast group using ***Join and Prune messages***.
- One of the router is designated as *rendezvous point* (RP) for each group in a domain to receive PIM messages.
- Multicast forwarding *tree* is built as a result of routers sending Join messages to RP.
- Two types of trees to be constructed:
  - ***Shared tree*** - used by all senders
  - ***Source-specific tree*** - used only by a specific sending host
- The normal mode of operation creates the shared tree first, followed by one or more source-specific trees

### Shared Tree

- When a router sends Join message for group  $G$  to RP, it goes through a set of routers.
- Join message is *wildcarded* (\*), i.e., it is applicable to all senders.
- Routers create an *entry* (\*,  $G$ ) in its forwarding table for the shared tree.
- Interface* on which the Join arrived is marked to forward packets for that group.
- Forwards* Join towards rendezvous router RP.
- Eventually, the message arrives at RP. Thus a shared tree with RP as *root* is formed.

### ***Example***

- Router  $R4$  sends Join message for group  $G$  to rendezvous router RP.
- Join message is received by router  $R2$ . It makes an entry (\*,  $G$ ) in its table and forwards the message to  $RP$ .
- When  $R5$  sends Join message for group  $G$ ,  $R2$  does not forwards the Join. It *adds* an outgoing interface to the forwarding table created for that group.

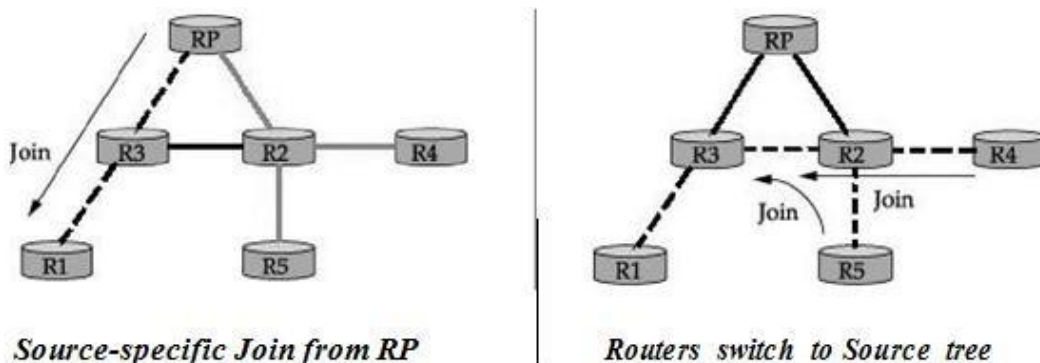


- As routers send Join message for a group, branches are *added* to the tree, i.e., shared.
- Multicast packets sent from hosts are forwarded to *designated* router RP.
- Suppose router *R1*, receives a message to group *G*.
  - *R1* has no state for group *G*.
  - Encapsulates the multicast packet in a Register message.
  - Multicast packet is tunneled along the way to RP.
- RP decapsulates the packet and sends multicast packet onto the shared tree, towards *R2*.
- *R2* forwards the multicast packet to routers *R4* and *R5* that have members for group *G*.

**Source-Specific Tree**

- RP can force routers to know about group *G*, by sending Join message to the sending host, so that tunneling can be avoided.
- Intermediary routers create *sender-specific* entry (*S, G*) in their tables. Thus a source-specific route from *R1* to RP is formed.
- If there is high rate of packets sent from a sender to a group *G*, then shared-tree is *replaced* by source-specific tree with sender as root.

**Example**



- Rendezvous router RP sends a Join message to the host router *R1*.
- Router *R3* learns about group *G* through the message sent by RP.
- Router *R4* send a source-specific Join due to high rate of packets from sender.
- Router *R2* learns about group *G* through the message sent by *R4*.



- Eventually a source-specific tree is formed with  $R1$  as root.

### **Analysis of PIM**

- Protocol independent because, tree is based on Join messages via *shortest* path.
  - Shared trees are more *scalable* than source-specific trees.
  - Source-specific trees enable *efficient* routing than shared trees.
-

## UNIT V – DATA LINK AND PHYSICAL LAYERS

Data Link Layer – Framing – Flow control – Error control – Data-Link Layer Protocols – HDLC – PPP - Media Access Control – Ethernet Basics – CSMA/CD – Virtual LAN – Wireless LAN (802.11) - Physical Layer: Data and Signals - Performance – Transmission media - Switching – Circuit Switching.

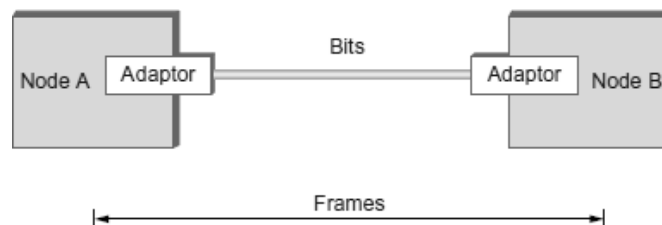
### DATA LINK LAYER

- The data link control (DLC) deals with procedures for communication between two adjacent nodes—node-to-node communication—no matter whether the link is dedicated or broadcast.
- Data link control service include
  - (1) Framing (2) Flow Control (3) Error Control

#### 1.

### FRAMING

- The data-link layer packs the bits of a message into frames, so that each frame is distinguishable from another.



- Although the whole message could be packed in one frame, that is not normally done.
- One reason is that a frame can be very large, making flow and error control very inefficient.
- When a message is carried in one very large frame, even a single-bit error would require the retransmission of the whole frame.
- When a message is divided into smaller frames, a single-bit error affects only that small frame.
- Framing in the data-link layer separates a message from one source to a destination by adding a sender address and a destination address.
- The destination address defines where the packet is to go; the sender address helps the recipient acknowledge the receipt.

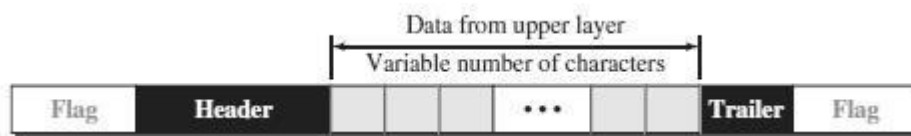
#### **Frame Size**

- Frames can be of fixed or variable size.

- Frames of fixed size are called cells. In fixed-size framing, there is no need for defining the boundaries of the frames; the size itself can be used as a delimiter.
- In variable-size framing, we need a way to define the end of one frame and the beginning of the next. Two approaches were used for this purpose: a character-oriented approach and a bit-oriented approach.

### Character-Oriented Framing

- In character-oriented (or byte-oriented) framing, data to be carried are 8-bit characters.
- To separate one frame from the next, an 8-bit (1-byte) flag is added at the beginning and the end of a frame.
- The flag, composed of protocol-dependent special characters, signals the start or end of a frame.

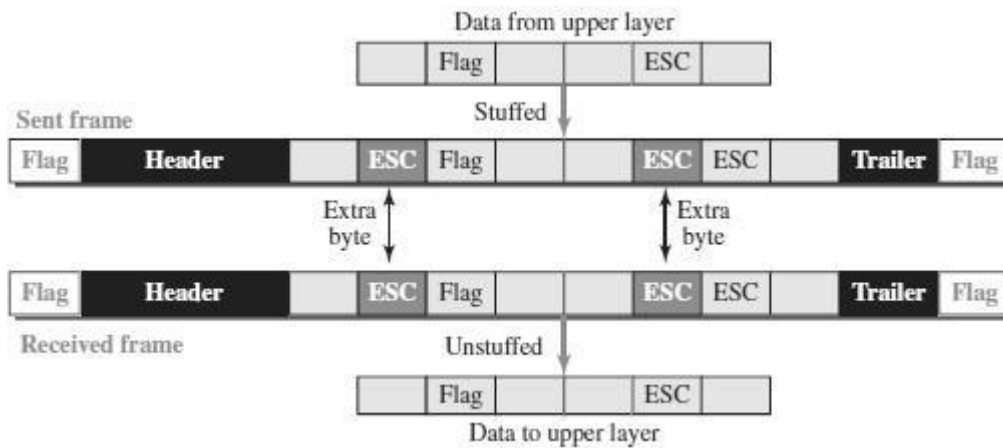


- Any character used for the flag could also be part of the information.
- If this happens, when it encounters this pattern in the middle of the data, the receiver thinks it has reached the end of the frame.
- To fix this problem, a **byte-stuffing** strategy was added to character-oriented framing.

www.EnggTree.com

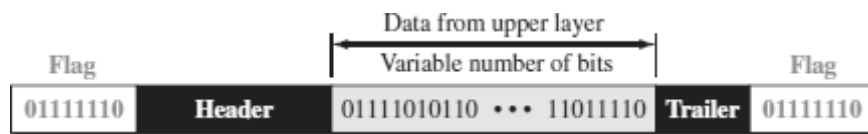
### Byte Stuffing (or) Character Stuffing

- **Byte stuffing is the process of adding one extra byte whenever there is a flag or escape character in the text.**
- In byte stuffing, a special byte is added to the data section of the frame when there is a character with the same pattern as the flag.
- The data section is stuffed with an extra byte. This byte is usually called the escape character (ESC) and has a predefined bit pattern.
- Whenever the receiver encounters the ESC character, it removes it from the data section and treats the next character as data, not as a delimiting flag.



### Bit-Oriented Framing

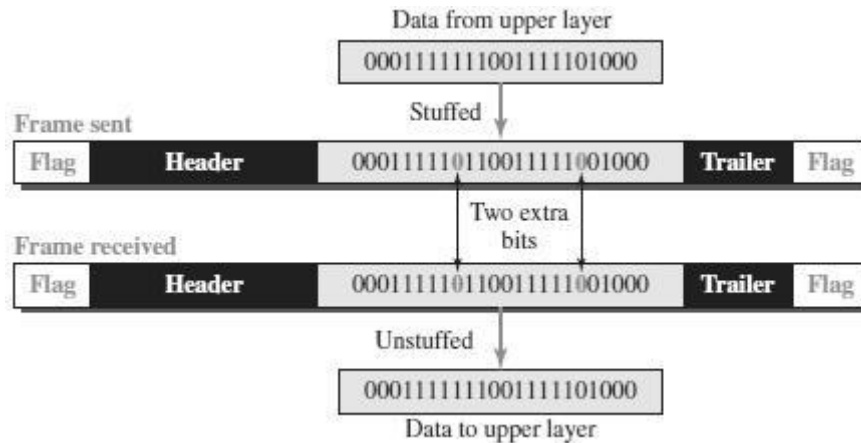
- In bit-oriented framing, the data section of a frame is a sequence of bits to be interpreted by the upper layer as text, graphic, audio, video, and so on.
- In addition to headers and trailers, we still need a delimiter to separate one frame from the other.
- Most protocols use a special 8-bit pattern flag, 01111110, as the delimiter to define the beginning and the end of the frame



- If the flag pattern appears in the data, the receiver must be informed that this is not the end of the frame.
- This is done by stuffing 1 single bit (instead of 1 byte) to prevent the pattern from looking like a flag. The strategy is called **bit stuffing**.

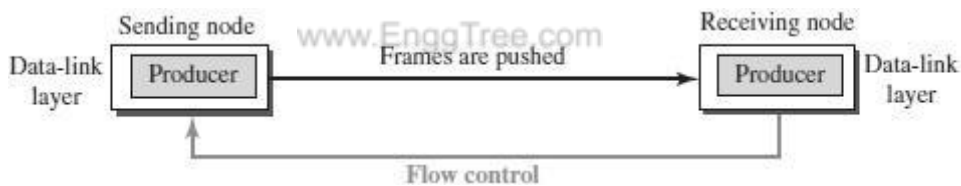
### Bit Stuffing

- **Bit stuffing is the process of adding one extra 0 whenever five consecutive 1s follow a 0 in the data, so that the receiver does not mistake the pattern 01111110 for a flag.**
- In bit stuffing, if 0 and five consecutive 1 bits are encountered, an extra 0 is added.
- This extra stuffed bit is eventually removed from the data by the receiver.
- The extra bit is added after one 0 followed by five 1's regardless of the value of the next bit.
- This guarantees that the flag field sequence does not inadvertently appear in the frame.



### FLOW CONTROL

- **Flow control refers to a set of procedures used to restrict the amount of data that the sender can send before waiting for acknowledgment.**
- The receiving device has limited speed and limited memory to store the data.
- Therefore, the receiving device must be able to inform the sending device to stop the transmission temporarily before the limits are reached.
- It requires a buffer, a block of memory for storing the information until they are processed.

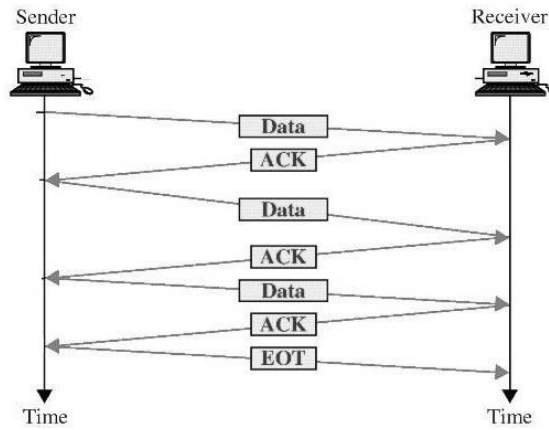


**Two methods have been developed to control the flow of data:**

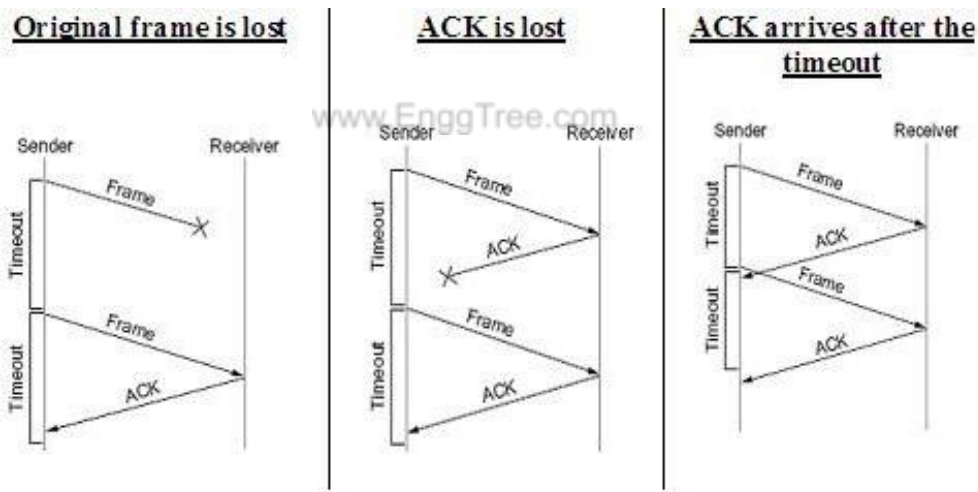
- Stop-and-Wait
- Sliding Window
- 

#### **STOP-AND-WAIT**

- The simplest scheme is the stop-and-wait algorithm.
- In the Stop-and-wait method, the sender waits for an acknowledgement after every frame it sends.
- When acknowledgement is received, then only next frame is sent.
- The process of alternately sending and waiting of a frame continues until the sender transmits the EOT (End of transmission) frame.



- If the acknowledgement is not received within the allotted time, then the sender assumes that the frame is lost during the transmission, so it will retransmit the frame.
- The acknowledgement may not arrive because of the following three scenarios :
  1. Original frame is lost
  2. ACK is lost
  3. ACK arrives after the timeout



**Advantage of Stop-and-wait**

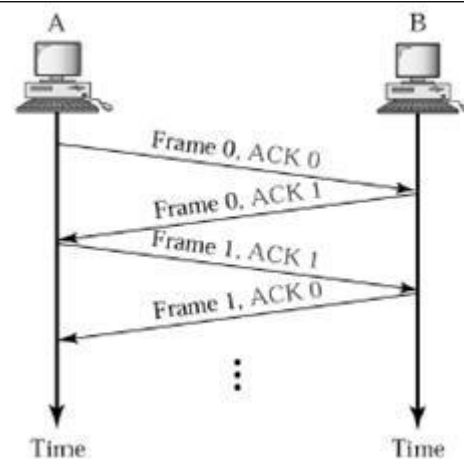
- The Stop-and-wait method is simple as each frame is checked and acknowledged before the next frame is sent

**Disadvantages of Stop-And-Wait**

- In stop-and-wait, at any point in time, there is only one frame that is sent and waiting to be acknowledged.
- This is not a good use of transmission medium.
- To improve efficiency, multiple frames should be in transition while waiting for ACK.

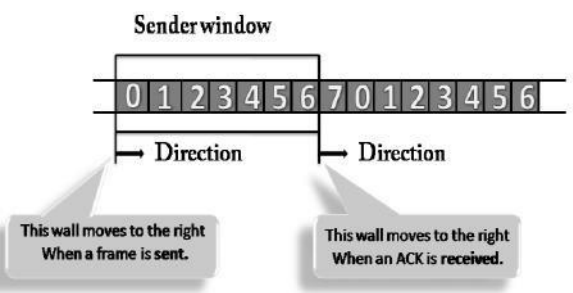
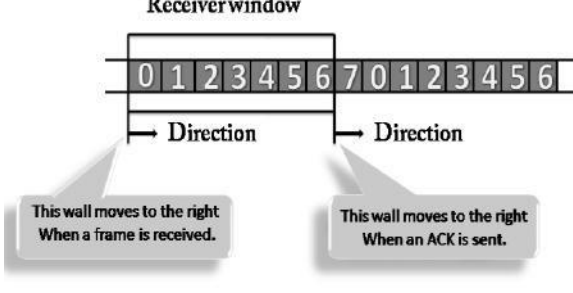
**PIGGYBACKING**

- A method to combine a data frame with ACK.
- Piggybacking saves bandwidth
- Station A and B both have data to send.
- Instead of sending separately, station A sends a data frame that includes an ACK.
- Station B does the same thing.

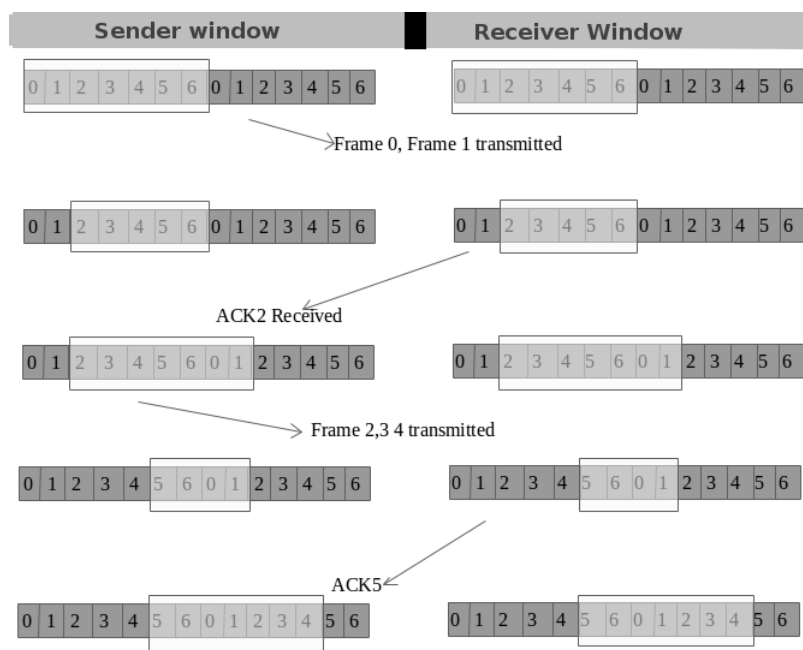


## SLIDING WINDOW

- The Sliding Window is a method of flow control in which a sender can transmit the several frames before getting an acknowledgement.
- In Sliding Window Control, multiple frames can be sent one after the another due to which capacity of the communication channel can be utilized efficiently.
- A single ACK acknowledge multiple frames.
- Sliding Window refers to imaginary boxes at both the sender and receiver end.
- The window can hold the frames at either end, and it provides the upper limit on the number of frames that can be transmitted before the acknowledgement.
- Frames can be acknowledged even when the window is not completely filled.
- The window has a specific size in which they are numbered as modulo-n means that they are numbered from 0 to n-1.
- For example, if  $n = 8$ , the frames are numbered from  
0,1,2,3,4,5,6,7,0,1,2,3,4,5,6,7,0,1.....
- The size of the window is represented as n-1. Therefore, maximum n-1 frames can be sent before acknowledgement.
- When the receiver sends the ACK, it includes the number of the next frame that it wants to receive.
- For example, to acknowledge the string of frames ending with frame number 4, the receiver will send the ACK containing the number 5.
- When the sender sees the ACK with the number 5, it got to know that the frames from 0 through 4 have been received.

Sender Window	Receiver Window
	
<ul style="list-style-type: none"> <li>At the beginning of a transmission, the sender window contains <math>n-1</math> frames.</li> <li>When a frame is sent, the size of the window shrinks.</li> <li>For example, if the size of the window is 'w' and if three frames are sent out, then the number of frames left out in the sender window is <math>w-3</math>.</li> <li>Once the ACK has arrived, then the sender window expands to the number which will be equal to the number of frames acknowledged by ACK.</li> </ul>	<ul style="list-style-type: none"> <li>At the beginning of transmission, the receiver window does not contain n frames, but it contains <math>n-1</math> spaces for frames.</li> <li>When the new frame arrives, the size of the window shrinks.</li> <li>For example, the size of the window is w and if three frames are received then the number of spaces available in the window is <math>(w-3)</math>.</li> <li>Once the acknowledgement is sent, the receiver window expands by the number equal to the number of frames acknowledged.</li> </ul>

**Example of Sliding Window**





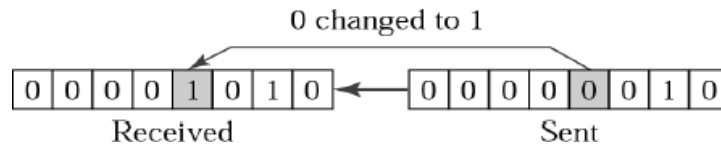
### 3. ERROR CONTROL

Data can be corrupted during transmission. For reliable communication, errors must be detected and corrected. Error Control is a technique of error detection and retransmission.

#### **TYPES OF ERRORS**

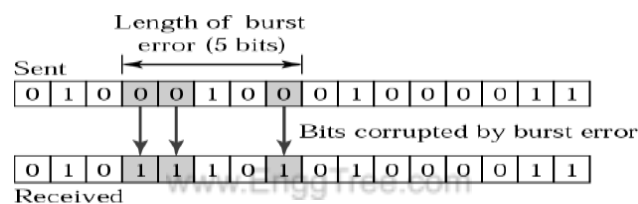
##### **SINGLE-BIT ERROR**

The term Single-bit error means that only one bit of a given data unit (such as byte, character, data unit or packet) is changed from 1 to 0 or from 0 to 1.



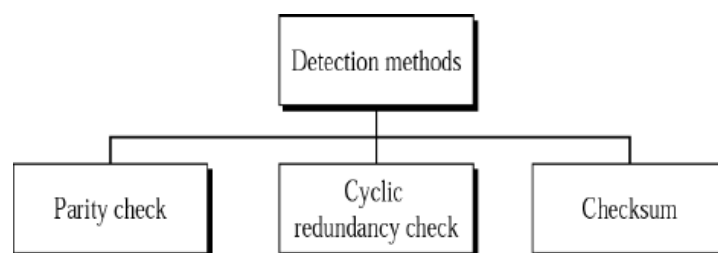
##### **BURST ERROR**

The term Burst Error means that two or more bits in the data unit have changed from 1 to 0 or from 0 to 1.



#### **ERROR DETECTION TECHNIQUES / METHODS**

The basic idea behind any error detection scheme is to add additional information to a frame that can be used to determine if errors have been introduced.



##### **PARITY CHECK**

- One bit, called parity bit is added to every data unit so that the total number of 1's in the data unit becomes even (or) odd.
- The source then transmits this data via a link, and bits are checked and verified at the destination.
- Data is considered accurate if the number of bits (even or odd) matches the number transmitted from the source.
- This techniques is the most common and least complex method.

1. **Even parity** – Maintain even number of 1s

E.g., 1011 → 1011 **1**

2. **Odd parity** – Maintain odd number of 1s

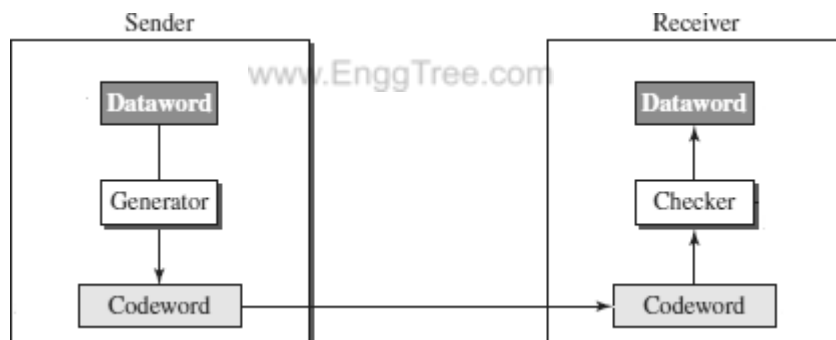
E.g., 1011 → 1011 **0**

### CYCLIC REDUNDANCY CHECK

- Cyclic codes refer to encoding messages by adding a fixed-length check value.
- CRCs are popular because they are simple to implement, easy to analyze mathematically and particularly good at detecting common errors caused in transmission channels.

#### Steps Involved:

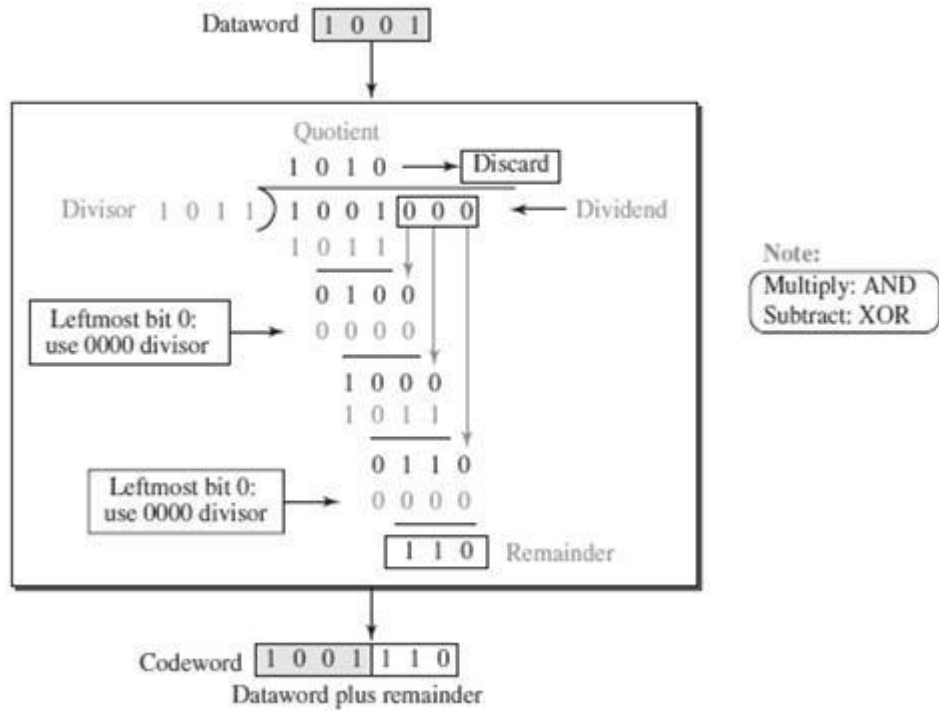
- Consider the original message (data word) as  $M(x)$  consisting of 'k' bits and the divisor as  $C(x)$  consists of 'n+1' bits.
- The original message  $M(x)$  is appended by 'n' bits of zeros. Let us call this zero-extended message as  $T(x)$ .
- Divide  $T(x)$  by  $C(x)$  and find the remainder.
- The division operation is performed using XOR operation.
- The resultant remainder is appended to the original message  $M(x)$  as CRC and sent by the sender (codeword).



#### Example 1:

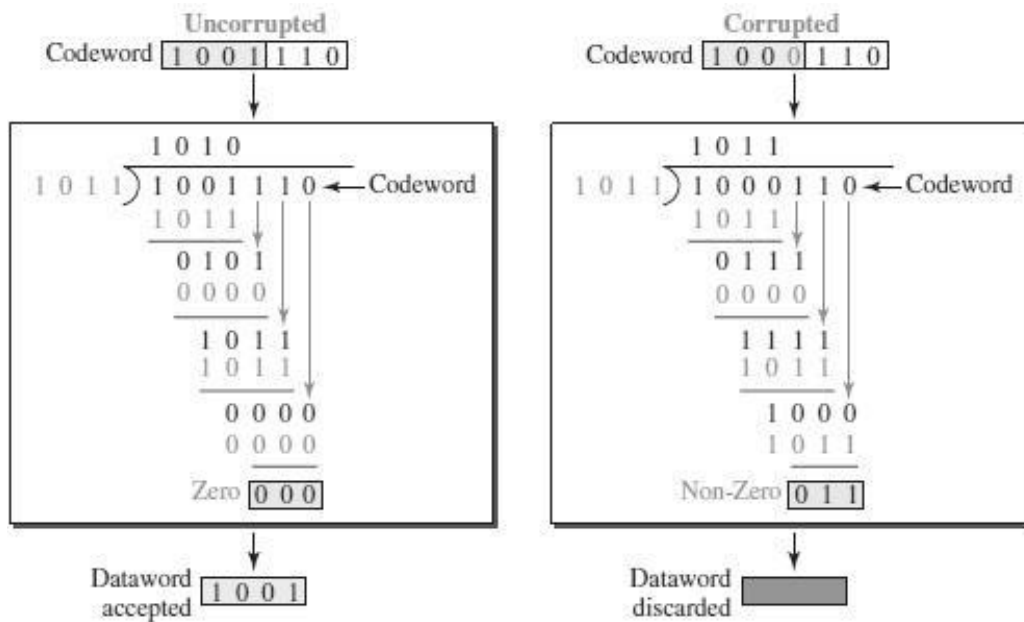
- Consider the Dataword / Message  $M(x) = 1001$
- Divisor  $C(x) = 1011$  ( $n+1=4$ )
- Appending 'n' zeros to the original Message  $M(x)$ .
- The resultant message is called  $T(x) = 1001$  **000**. (here  $n=3$ )
- Divide  $T(x)$  by the divisor  $C(x)$  using XOR operation.

**Sender Side:**



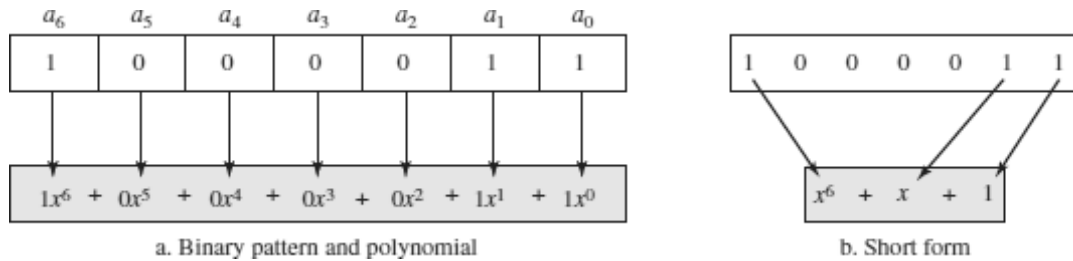
**Receiver Side:**

(For Both Case – Without Error and With Error)



### Polynomials

- A pattern of 0s and 1s can be represented as a **polynomial** with coefficients of 0 and 1.
- The power of each term shows the position of the bit; the coefficient shows the value of the bit.

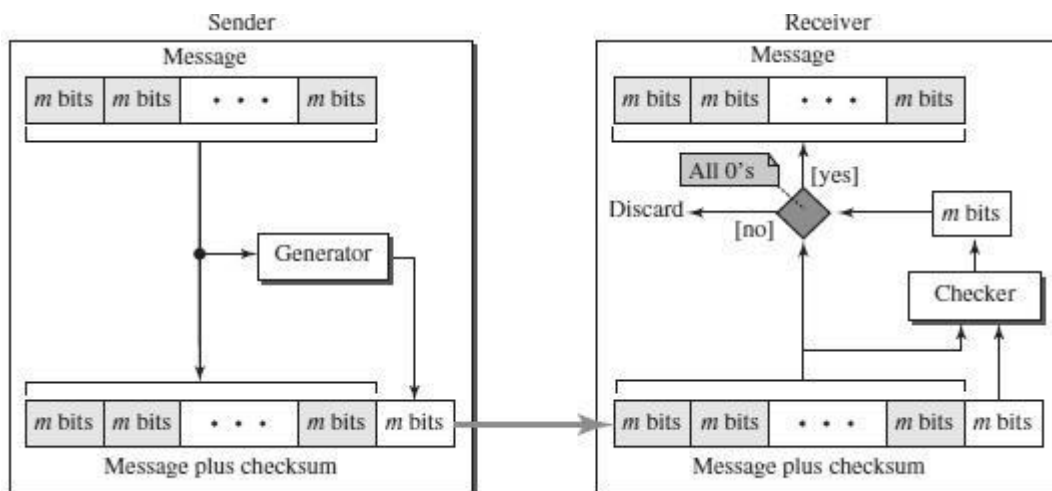


### INTERNET CHECKSUM

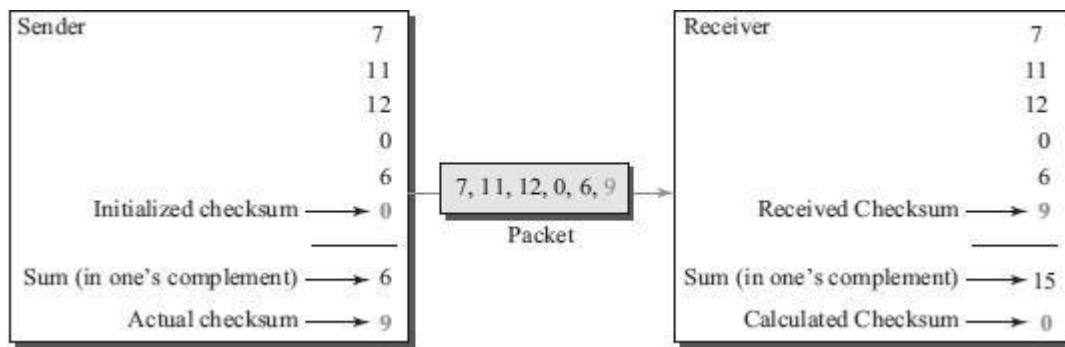
- Checksum is a calculated value that is used to determine the integrity of data.

*Procedure to calculate the traditional checksum*

Sender	Receiver
1. The message is divided into 16-bit words.	1. The message and the checksum are received.
2. The value of the checksum word is initially set to zero.	2. The message is divided into 16-bit words.
3. All words including the checksum are added using one's complement addition.	3. All words are added using one's complement addition.
4. The sum is complemented and becomes the checksum.	4. The sum is complemented and becomes the new checksum.
5. The checksum is sent with the data.	5. If the value of the checksum is 0, the message is accepted; otherwise, it is rejected.



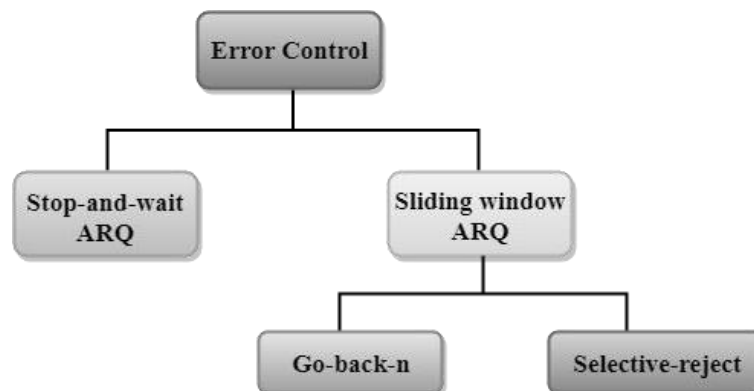
**Example:** Let the message to be transmitted be 7,11,12,0,6.



### ERROR CONTROL

- Error control includes both error detection and error correction.
- Whenever an error is detected, specified frames are retransmitted
- It allows the receiver to inform the sender if a frame is lost or damaged during transmission and coordinates the retransmission of those frames by the sender.
- Includes the following actions:
  - **Error detection**
  - Positive Acknowledgement (**ACK**): if the frame arrived with no errors
  - Negative Acknowledgement (**NAK**): if the frame arrived with errors
  - Retransmissions after **Timeout**: Frame is retransmitted after certain amount of time if no acknowledgement was received
- Error control in the data link layer is based on automatic repeat request (ARQ).

### Categories of Error Control

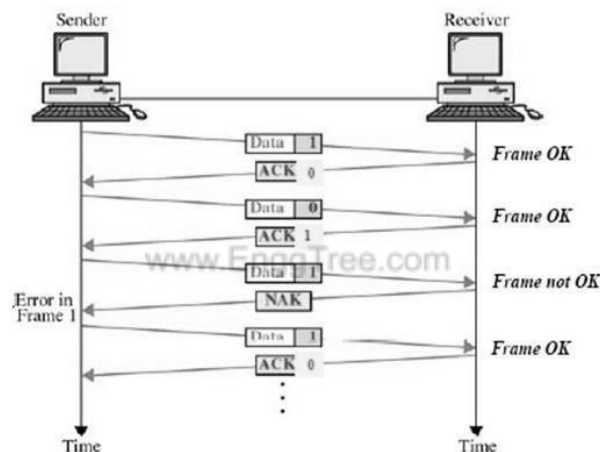


### STOP-AND-WAIT ARQ

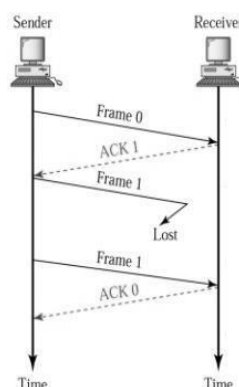
- Stop-and-wait ARQ is a technique used to retransmit the data in case of damaged or lost frames.
- This technique works on the principle that the sender will not transmit the next frame until it receives the acknowledgement of the last transmitted frame.

#### Two possibilities of the retransmission in Stop and Wait ARQ:

- **Damaged Frame:** When the receiver receives a damaged frame(i.e., the frame contains an error), then it returns the NAK frame. For example, when the frame DATA 1 is sent, and then the receiver sends the ACK 0 frame means that the data 1 has arrived correctly. The sender transmits the next frame: DATA 0. It reaches undamaged, and the receiver returns ACK 1. The sender transmits the third frame: DATA 1. The receiver reports an error and returns the NAK frame. The sender retransmits the DATA 1 frame.



- **Lost Frame:** Sender is equipped with the timer and starts when the frame is transmitted. Sometimes the frame has not arrived at the receiving end so that it cannot be acknowledged either positively or negatively. The sender waits for acknowledgement until the timer goes off. If the timer goes off, it retransmits the last transmitted frame.



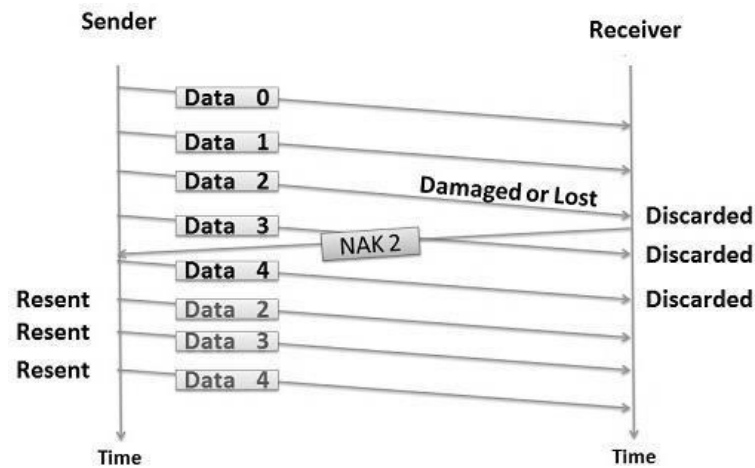
## SLIDING WINDOW ARQ

Sliding Window ARQ is a technique used for continuous transmission error control.

### Two protocols used in sliding window ARQ:

#### 1.GO-BACK-N ARQ

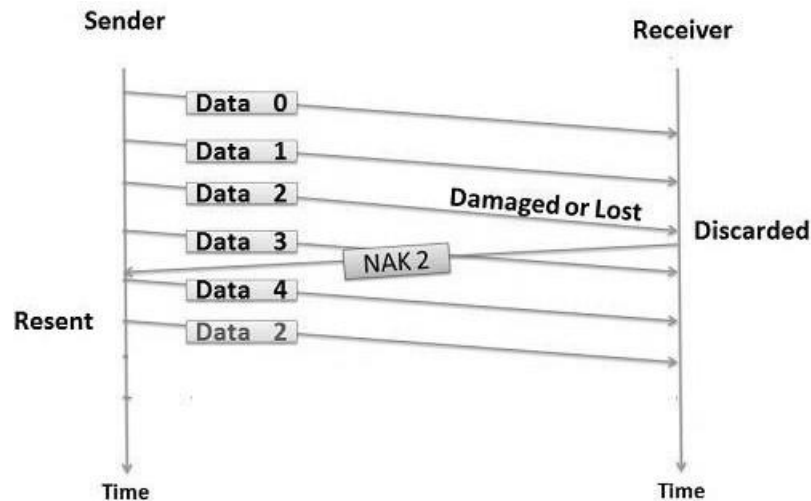
- In Go-Back-N ARQ protocol, if one frame is lost or damaged, then it retransmits all the frames after which it does not receive the positive ACK.



- In the above figure, three frames (Data 0,1,2) have been transmitted before an error discovered in the third frame.
- The receiver discovers the error in Data 2 frame, so it returns the NAK 2 frame.
- All the frames including the damaged frame (Data 2,3,4) are discarded as it is transmitted after the damaged frame.
- Therefore, the sender retransmits the frames (Data2,3,4).

#### 2.SELECTIVE-REJECT(REPEAT) ARQ

- Selective-Reject ARQ technique is more efficient than Go-Back-n ARQ.
- In this technique, only those frames are retransmitted for which negative acknowledgement (NAK) has been received.
- The receiver storage buffer keeps all the damaged frames on hold until the frame in error is correctly received.
- The receiver must have an appropriate logic for reinserting the frames in a correct order.
- The sender must consist of a searching mechanism that selects only the requested frame for retransmission.



- In the above figure, three frames (Data 0,1,2) have been transmitted before an error discovered in the third frame.
- The receiver discovers the error in Data 2 frame, so it returns the NAK 2 frame.
- The damaged frame only (Data 2) is discarded.
- The other subsequent frames (Data 3,4) are accepted.
- Therefore, the sender retransmits only the damaged frame (Data2).

www.EnggTree.com

Four protocols have been defined for the data-link layer controls.

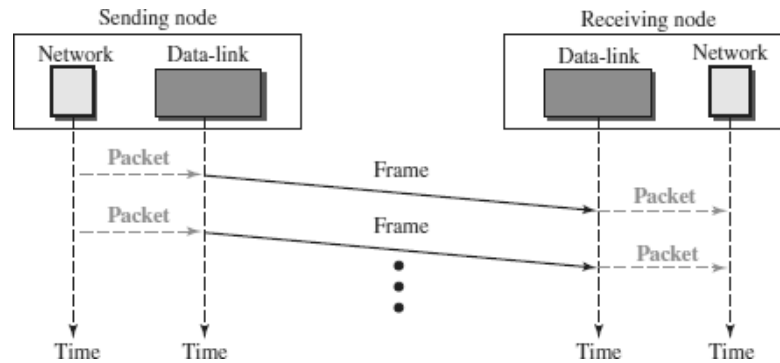
They are

1. Simple Protocol
2. Stop-and-Wait Protocol
3. Go-Back-N Protocol
4. Selective-Repeat Protocol

### **1.SIMPLE PROTOCOL**

- The first protocol is a simple protocol with neither flow nor error control.
- We assume that the receiver can immediately handle any frame it receives.
- In other words, the receiver can never be overwhelmed with incoming frames.
- The data-link layers of the sender and receiver provide transmission services for their network layers.





- The data-link layer at the sender gets a packet from its network layer, makes a frame out of it, and sends the frame.
- The data-link layer at the receiver receives a frame from the link, extracts the packet from the frame, and delivers the packet to its network layer.

**NOTE :**

**2. STOP-AND-WAIT PROTOCOL**

REFER STOP AND WAIT FROM FLOW CONTROL

**3. GO-BACK-N PROTOCOL**

REFER GO-BACK-N ARQ FROM ERROR CONTROL

**4. SELECTIVE-REPEAT PROTOCOL**

REFER SELECTIVE-REPEAT ARQ FROM ERROR CONTROL

## 5. HDLC (HIGH-LEVEL DATA LINK CONTROL)

- High-level Data Link Control (HDLC) is a bit-oriented protocol
- HDLC is used for communication over point-to-point and multipoint links.
- HDLC implements the Stop-and-Wait protocol.

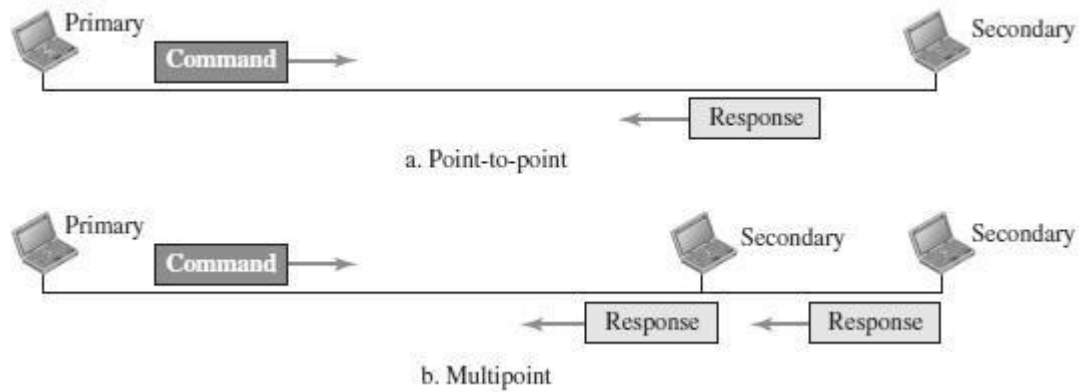
### HDLC CONFIGURATIONS AND TRANSFER MODES

HDLC provides two common transfer modes that can be used in different configurations:

1. Normal response mode (NRM)
2. Asynchronous balanced mode (ABM).

**Normal response mode (NRM)**

- In normal response mode (NRM), the station configuration is unbalanced.
- We have one primary station and multiple secondary stations.
- A *primary station* can send commands; a *secondary station* can only respond.
- The NRM is used for both point-to-point and multipoint links.

**Asynchronous balanced mode (ABM)**

- In ABM, the configuration is balanced.
- The link is point-to-point, and each station can function as a primary and a secondary (acting as peers).
- This is the common mode today.

**HDLC FRAMES**

HDLC defines three types of frames:

1. Information frames (I-frames) - used to carry user data
2. Supervisory frames (S-frames) - used to carry control information
3. Unnumbered frames (U-frames) – reserved for system management

Each type of frame serves as an envelope for the transmission of a different type of message.

Each frame in HDLC may contain up to six fields:

1. Beginning flag field
2. Address field
3. Control field
4. Information field (User Information/ Management Information)
5. Frame check sequence (FCS) field
6. Ending flag field

In multiple-frame transmissions, the ending flag of one frame can serve as the beginning flag of the next frame.

**I – Frame**



**S – Frame**



**U – Frame**

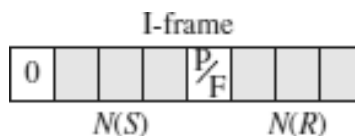


- **Flag field** - This field contains synchronization pattern 01111110, which identifies both the beginning and the end of a frame.
- **Address field** - This field contains the address of the secondary station. If a primary station created the frame, it contains a ‘to’ address. If a secondary station creates the frame, it contains a ‘from’ address. The address field can be one byte or several bytes long, depending on the needs of the network.
- **Control field**. The control field is one or two bytes used for flow and error control.
- **Information field**. The information field contains the user’s data from the network layer or management information. Its length can vary from one network to another.
- **FCS field**. The frame check sequence (FCS) is the HDLC error detection field. It can contain either a 16- bit or 32-bit CRC.

**CONTROL FIELD FORMAT FOR THE DIFFERENT FRAME TYPES**

**Control Field for I-Frames**

- I-frames are designed to carry user data from the network layer. In addition, they can include flow-control and error-control information

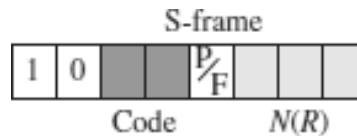


- The first bit defines the type. If the first bit of the control field is 0, this means the frame is an I-frame.
- The next 3 bits, called N(S), define the sequence number of the frame.
- The last 3 bits, called N(R), correspond to the acknowledgment number when piggybacking is used.
- The single bit between N(S) and N(R) is called the P/F bit. If this bit is 1 it means poll (the frame is sent by a primary station to a secondary).

- If this bit is 0 it means final (the frame is sent by a secondary to a Primary).

### Control Field for S-Frames

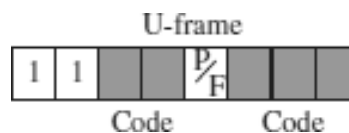
- Supervisory frames are used for flow and error control whenever piggybacking is either impossible or inappropriate.
- S-frames do not have information fields



- If the first 2 bits of the control field are 10, this means the frame is an S-frame.
- The last 3 bits, called N(R), correspond to the acknowledgment number (ACK) or negative acknowledgment number (NAK), depending on the type of S-frame.
- The 2 bits called code are used to define the type of S-frame itself.
- With 2 bits, we can have four types of S-frames – **Receive ready (RR), Receive not ready (RNR), Reject (REJ) and Selective reject (SREJ).**

### Control Field for U-Frames

- Unnumbered frames are used to exchange session management and control information between connected devices.
- U-frames contain an information field, but used only for system management information and not user data.



- If the first 2 bits of the control field are 11, this means the frame is an U-frame.
  - U-frame codes are divided into two sections: a 2-bit prefix before the P/F bit and a 3-bit suffix after the P/F bit.
  - Together, these two segments (5 bits) can be used to create up to 32 different types of U-frames.
-

## 6. POINT-TO-POINT PROTOCOL (PPP)

- Point-to-Point Protocol (PPP) was devised by IETF (Internet Engineering Task Force) in 1990 as a Serial Line Internet Protocol (SLIP).
- PPP is a data link layer communications protocol used to establish a direct connection between two nodes.
- It connects two routers directly without any host or any other networking device in between.
- It is used to connect the Home PC to the server of ISP via a modem.
- It is a byte - oriented protocol that is widely used in broadband communications having heavy loads and high speeds.
- Since it is a data link layer protocol, data is transmitted in frames. It is also known as RFC 1661.

### Services Provided by PPP

The main services provided by Point - to - Point Protocol are –

1. Defining the frame format of the data to be transmitted.
2. Defining the procedure of establishing link between two points and exchange of data.
3. Stating the method of encapsulation of network layer data in the frame.
4. Stating authentication rules of the communicating devices.
5. Providing address for network communication.
6. Providing connections over multiple links.
7. Supporting a variety of network layer protocols by providing a range of services.

### PPP Frame

PPP is a byte - oriented protocol where each field of the frame is composed of one or more bytes.



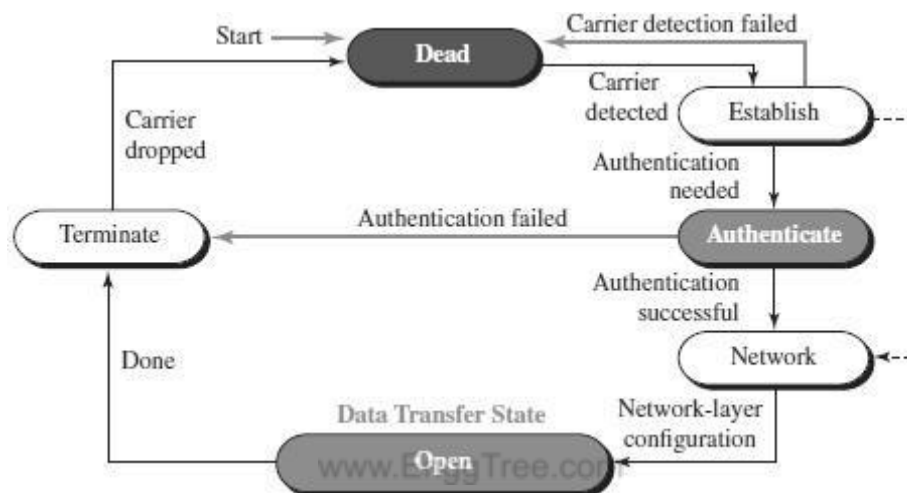
1. **Flag** – 1 byte that marks the beginning and the end of the frame. The bit pattern of the flag is 01111110.
2. **Address** – 1 byte which is set to 11111111 in case of broadcast.
3. **Control** – 1 byte set to a constant value of 11000000.
4. **Protocol** – 1 or 2 bytes that define the type of data contained in the payload field.
5. **Payload** – This carries the data from the network layer. The maximum length of the payload field is 1500 bytes.
6. **FCS** – It is a 2-byte(16-bit) or 4 bytes(32-bit) frame check sequence for error detection. The standard code used is CRC.

## Byte Stuffing in PPP Frame

Byte stuffing is used in PPP payload field whenever the flag sequence appears in the message, so that the receiver does not consider it as the end of the frame. The escape byte, 01111101, is stuffed before every byte that contains the same byte as the flag byte or the escape byte. The receiver on receiving the message removes the escape byte before passing it onto the network layer.

## Transition Phases in PPP

The PPP connection goes through different states as shown in a *transition phase* diagram.



- ❖ **Dead:** In dead phase the link is not used. There is no active carrier and the line is quiet.
- ❖ **Establish:** Connection goes into this phase when one of the nodes start communication. In this phase, two parties negotiate the options. If negotiation is successful, the system goes into authentication phase or directly to networking phase.
- ❖ **Authenticate:** This phase is optional. The two nodes may decide whether they need this phase during the establishment phase. If they decide to proceed with authentication, they send several authentication packets. If the result is successful, the connection goes to the networking phase; otherwise, it goes to the termination phase.
- ❖ **Network:** In network phase, negotiation for the network layer protocols takes place. PPP specifies that two nodes establish a network layer agreement before data at the network layer can be exchanged. This is because PPP supports several protocols at network layer. If a node is running multiple protocols simultaneously at the network layer, the receiving node needs to know which protocol will receive the data.
- ❖ **Open:** In this phase, data transfer takes place. The connection remains in this phase until one of the endpoints wants to end the connection.
- ❖ **Terminate:** In this phase connection is terminated.

## Components/Protocols of PPP

Three sets of components/protocols are defined to make PPP powerful:

- ❖ Link Control Protocol (LCP)
- ❖ Authentication Protocols (AP)
- ❖ Network Control Protocols (NCP)

**Link Control Protocol (LCP)** – It is responsible for establishing, configuring, testing, maintaining and terminating links for transmission. It also provides negotiation mechanisms to set options between the two endpoints. Both endpoints of the link must reach an agreement about the options before the link can be established.

**Authentication Protocols (AP)** – Authentication means validating the identity of a user who needs to access a set of resources. PPP has created two protocols for authentication - Password Authentication Protocol and Challenge Handshake Authentication Protocol.

### *PAP*

The Password Authentication Protocol (PAP) is a simple authentication procedure with a two-step process:

- a. The user who wants to access a system sends an authentication identification (usually the user name) and a password.
- b. The system checks the validity of the identification and password and either accepts or denies connection.

### *CHAP*

The Challenge Handshake Authentication Protocol (CHAP) is a three-way handshaking authentication protocol that provides greater security than PAP. In this method, the password is kept secret; it is never sent online.

- a. The system sends the user a challenge packet containing a challenge value.
- b. The user applies a predefined function that takes the challenge value and the user's own password and creates a result. The user sends the result in the response packet to the system.
- c. The system does the same. It applies the same function to the password of the user (known to the system) and the challenge value to create a result. If the result created is the same as the result sent in the response packet, access is granted; otherwise, it is denied.

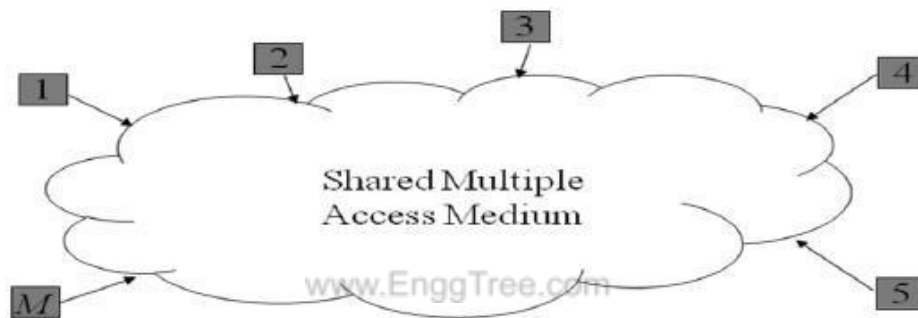
CHAP is more secure than PAP, especially if the system continuously changes the challenge value. Even if the intruder learns the challenge value and the result, the password is still secret.

**Network Control Protocols (NCP)** – PPP is a multiple-network-layer protocol. It can carry a network-layer data packet from protocols defined by the Internet. PPP

has defined a specific Network Control Protocol for each network protocol. These protocols are used for negotiating the parameters and facilities for the network layer. For every higher-layer protocol supported by PPP, one NCP is there.

## 7. MEDIA ACCESS CONTROL (MAC)

- When two or more nodes transmit data at the same time, their frames will collide and the link bandwidth is wasted during collision.
- To coordinate the access of multiple sending/receiving nodes to the shared link, we need a protocol to coordinate the transmission.
- These protocols are called Medium or Multiple Access Control (MAC) Protocols. MAC belongs to the data link layer of OSI model
- MAC defines rules for orderly access to the shared medium. It tries to ensure that no two nodes are interfering with each other's transmissions, and deals with the situation when they do.



### Issues involved in MAC

The key issues involved are –

- **Where** the control is exercised - refers to whether the control is exercised in a centralized or distributed manner
- **How** the control is exercised - refers to in what manner the control is exercised

### Goals of MAC

1. Fairness in sharing
2. Efficient sharing of bandwidth
3. Need to avoid packet collisions at the receiver due to interference

### MAC Management

- Medium allocation (collision avoidance)
- Contention resolution (collision handling)



## MAC Types

- **Round-Robin:** – Each station is given opportunity to transmit in turns. Either a central controller polls a station to permit to go, or stations can coordinate among themselves.
- **Reservation:** - Station wishing to transmit makes reservations for time slots in advance. (Centralized or distributed).
- **Contention (Random Access):** - No control on who tries; If collision occurs, retransmission takes place.

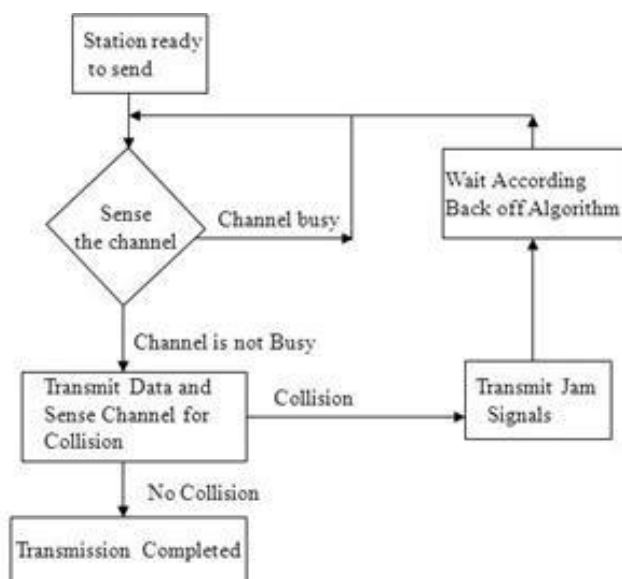
## MECHANISMS USED

- Wired Networks:
  - CSMA / CD – Carrier Sense Multiple Access / Collision Detection
- Wireless Networks:
  - CSMA / CA – Carrier Sense Multiple Access / Collision Avoidance

## CARRIER SENSE MULTIPLE ACCESS / COLLISION DETECTION (CSMA / CD)

- **Carrier Sense** in CSMA/CD means that all the nodes sense the medium to check whether it is idle or busy.
  - If the carrier sensed is idle, then the node transmits the entire frame.
  - If the carrier sensed is busy, the transmission is postponed.
- **Collision Detect** means that a node listens as it transmits and can therefore detect when a frame it is transmitting has collided with a frame transmitted by another node.

### Flowchart of CSMA/CD Operation

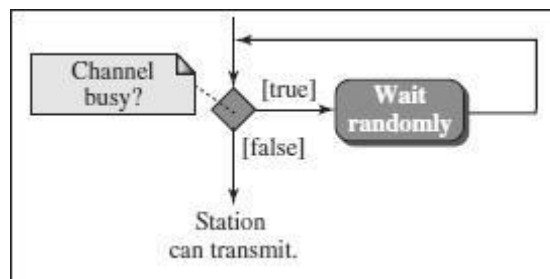


## Transmitter Algorithm in CSMA/CD

- Transmitter Algorithm defines the procedures for a node that senses a busy medium.
- Three types of Transmitter Algorithm exist.
- They are
  1. Non-Persistent Strategy
  2. Persistent Strategy: 1-Persistent & P-Persistent

### **Non-Persistent Strategy**

- In the non-persistent method, a station that has a frame to send senses the line.
- If the line is idle, it sends immediately.
- If the line is not idle, it waits a random amount of time and then senses the line again.



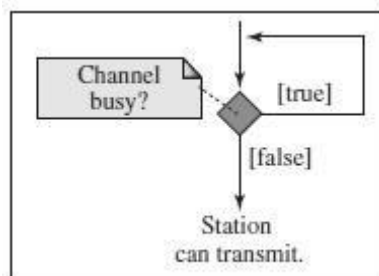
www.EnggTree.com

- The non-persistent approach reduces the chance of collision because it is unlikely that two or more stations will wait the same amount of time and retry to send simultaneously.
- However, this method reduces the efficiency of the network because the medium remains idle when there may be stations with frames to send.

### **Persistent Strategy**

#### 1-Persistent:

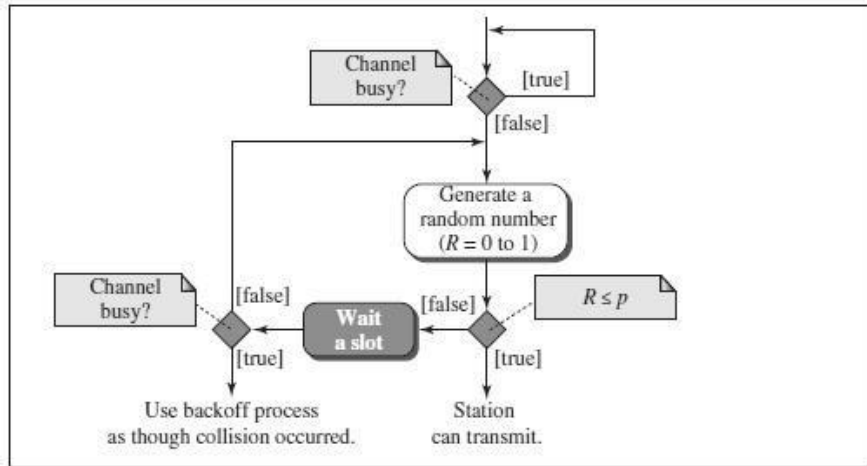
- The 1-persistent method is simple and straightforward.
- In this method, after the station finds the line idle, it sends its frame immediately (with probability 1).



- This method has the highest chance of collision because two or more stations may find the line idle and send their frames immediately.

**P-Persistent:**

- In this method, after the station finds the line idle it follows these steps:
- With probability  $p$ , the station sends its frame.
- With probability  $q = 1 - p$ , the station waits for the beginning of the next time slot and checks the line again.

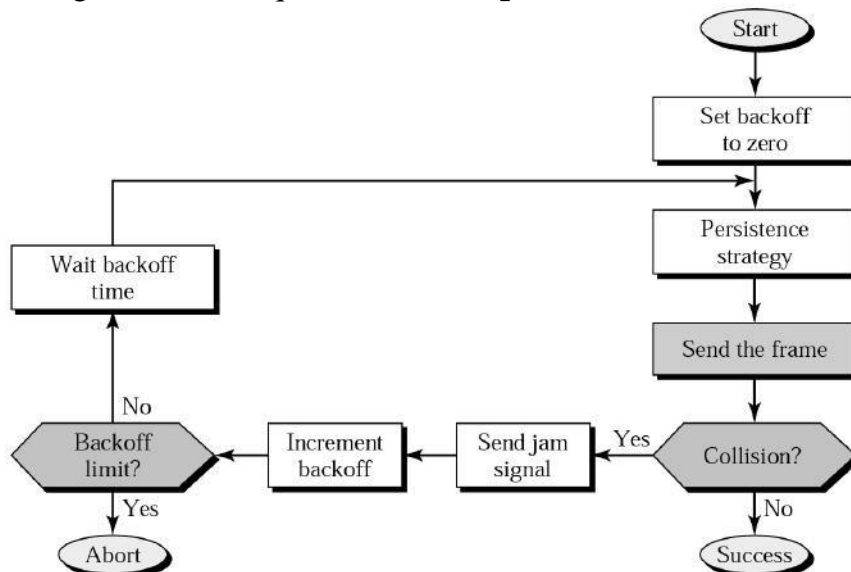


- The p-persistent method is used if the channel has time slots with a slot duration equal to or greater than the maximum propagation time.
- The p-persistent approach combines the advantages of the other two strategies. It reduces the chance of collision and improves efficiency.

www.EnggTree.com

**EXPONENTIAL BACK-OFF**

- Once an adaptor has detected a collision and stopped its transmission, it waits a certain amount of time and tries again.
- Each time it tries to transmit but fails, the adaptor doubles the amount of time it waits before trying again.
- This strategy of doubling the delay interval between each retransmission attempt is a general technique known as **exponential back-off**.



## CARRIER SENSE MULTIPLE ACCESS / COLLISION AVOIDANCE (CSMA/CA)

- Carrier sense multiple access with collision avoidance (CSMA/CA) was invented for wireless networks.
- Wireless protocol would follow exactly the same algorithm as the Ethernet— Wait until the link becomes idle before transmitting and back off should a collision occur.
- Collisions are avoided through the use of CSMA/CA's three strategies: the interframe space, the contention window, and acknowledgments

**Interframe Space (IFS)** - First, collisions are avoided by deferring transmission even if the channel is found idle. When an idle channel is found, the station does not send immediately. It waits for a period of time called the *interframe space* or *IFS*.

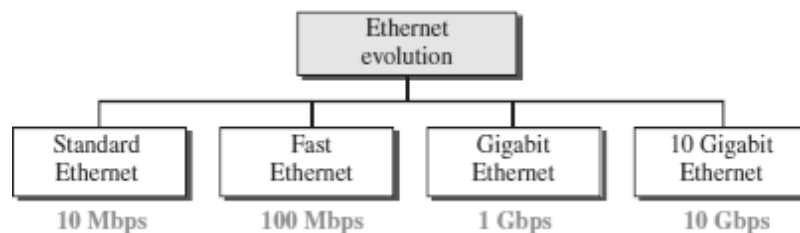
**Contention Window** - The **contention window** is an amount of time divided into slots. A station that is ready to send chooses a random number of slots as its wait time. The number of slots in the window changes according to the binary exponential backoff strategy. This means that it is set to one slot the first time and then doubles each time the station cannot detect an idle channel after the IFS time.

**Acknowledgment** - In addition, the data may be corrupted during the transmission. The positive acknowledgment and the time-out timer can help guarantee that the receiver has received the frame.

### ETHERNET BASICS

- Ethernet was developed in the mid-1970's at the Xerox Palo Alto Research Center (PARC),
- IEEE controls the Ethernet standards.
- The Ethernet is the most successful local area networking technology, that uses bus topology.
- The Ethernet is **multiple-access networks** that is set of nodes send and receive frames over a shared link.
- Ethernet uses the **CSMA / CD (C a r r i e r Sense Multiple Access with Collision Detection)** mechanism.

## EVOLUTION OF ETHERNET

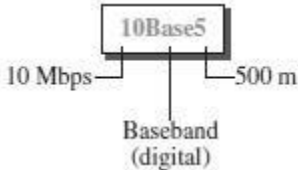
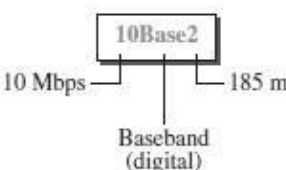
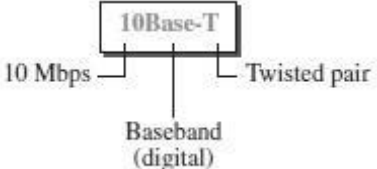
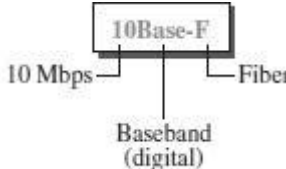


### Standard Ethernet (10 Mbps)

The original Ethernet technology with the data rate of 10 Mbps as the Standard Ethernet.

Standard Ethernet types are

1. 10Base5: Thick Ethernet,
2. 10Base2: Thin Ethernet,
3. 10Base-T: Twisted-Pair Ethernet
4. 10Base-F: Fiber Ethernet.

<p><b><u>10Base5: Thick Ethernet</u></b></p>  <ul style="list-style-type: none"> <li>• The first implementation is called <b>10Base5, thick Ethernet, or Thicknet.</b></li> <li>• 10Base5 was the first Ethernet specification to use a bus topology with an external <b>transceiver</b>(transmitter/receiver) connected via a tap to a thick coaxial cable.</li> </ul>	<p><b><u>10Base2: Thin Ethernet</u></b></p>  <ul style="list-style-type: none"> <li>• The second implementation is called <b>10Base2, thin Ethernet, or Cheapernet.</b></li> <li>• 10Base2 also uses a bus topology, but the cable is much thinner and more flexible.</li> <li>• In this case, the transceiver is normally part of the network interface card (NIC), which is installed inside the station.</li> </ul>
<p><b><u>10Base-T: Twisted-Pair Ethernet</u></b></p>  <ul style="list-style-type: none"> <li>• The third implementation is called <b>10Base-T or twisted-pair Ethernet.</b></li> <li>• 10Base-T uses a physical star topology. The stations are connected to a hub via two pairs of twisted cable.</li> </ul>	<p><b><u>10Base-F: Fiber Ethernet</u></b></p>  <ul style="list-style-type: none"> <li>• Although there are several types of optical fiber 10-Mbps Ethernet, the most common is called <b>10Base-F.</b></li> <li>• 10Base-F uses a star topology to connect stations to a hub.</li> <li>• The stations are connected to the hub using two fiber-optic cables.</li> </ul>

**Fast Ethernet (100 Mbps)**

Fast Ethernet or 100BASE-T provides transmission speeds up to 100 megabits per second and is typically used for LAN backbone systems.

The 100BASE-T standard consists of three different component specifications –

1. 100 BASE-TX
2. 100BASE-T4
3. 100BASE-FX

<b><u>100 BASE-TX</u></b>	<b><u>100BASE-T4</u></b>	<b><u>100BASE-FX</u></b>
<b>100Base-TX</b> uses two pairs of twisted-pair cable either UTP or STP. A 100Base-TX network can provide a data rate of 100 Mbps.	A new standard, called <b>100Base-T4</b> , was designed to use four pairs of UTP for transmitting 100 Mbps.	<b>100Base-FX</b> uses two pairs of fiber-optic cables. Optical fiber can easily handle high bandwidth requirements.

**Gigabit Ethernet (1 Gbps)**

- The Gigabit Ethernet upgrades the data rate to 1 Gbps (1000 Mbps).
- Gigabit Ethernet can be categorized as either a two-wire or a four-wire implementation.
- The two-wire implementations use fiber-optic cable (**1000Base-SX**, short-wave, or **1000Base-LX**, long-wave), or STP (**1000Base-CX**).
- The four-wire version uses category 5 twisted-pair cable (**1000Base-T**).

**10 Gigabit Ethernet (10 Gbps)**

- 10 Gigabit Ethernet is an upcoming Ethernet technology that transmits at 10 Gbps.
- 10 Gigabit Ethernet enables a familiar network technology to be used in LAN, MAN and WAN architectures.
- 10 Gigabit Ethernet uses multimode optical fiber up to 300 meters and single mode fiber up to 40 kilometers.
- Four implementations are the most common: **10GBase-SR**, **10GBase-LR**, **10GBase-EW**, and **10GBase-X4**.

## ACCESS METHOD/ PROTOCOL OF ETHERNET

The access method of Ethernet is CSMA/CD.

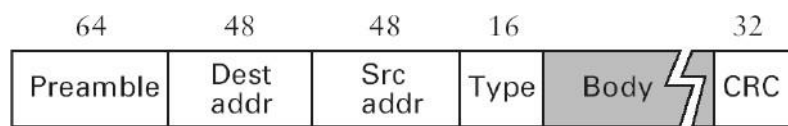
**Note: Refer CSMA/CD from MAC**

## COLLISION DETECTION IN ETHERNET

- As the Ethernet supports collision detection, senders are able to determine a collision.
- At the moment an adaptor detects that its frame is colliding with another, it first makes sure to transmit a **32-bit jamming sequence** along with the **64-bit preamble** (totally 96 bits) and then stops the transmission.
- These **96 bits** are sometimes called **Runt Frame**.

## FRAME FORMAT OF ETHERNET

The Ethernet frame is defined by the format given in the Fig.



- The 64-bit *preamble* allows the receiver to synchronize with the signal; it is a sequence of alternating 0's and 1's.
- Both the *source and destination* hosts are identified with a 48-bit *address*.
- The packet *type* field serves as the demultiplexing key.
- Each frame contains up to 1500 bytes of *data(Body)*.
- *CRC* is used for Error detection

## Ethernet Addresses

- Every Ethernet host has a unique Ethernet address (48 bits – 6 bytes).
- Ethernet address is represented by sequence of six numbers separated by colons.
- Each number corresponds to 1 byte of the 6-byte address and is given by pair of hexadecimal digits.
- **Eg: 8:0:2b:e4:b1:2** is the representation of  
00001000 00000000 00101011 11100100 10110001 00000010
- Each frame transmitted on an Ethernet is received by every adaptor connected to the Ethernet.
- In addition to *unicast* addresses an Ethernet address consisting of *all 1s* is treated as *broadcast* address.
- Similarly, the address that has the *first bit set to 1* but it is not the broadcast address is called *multicast* address.

## ADVANTAGES OF ETHERNET

Ethernets are successful because

- It is extremely *easy to administer and maintain*. There are no switches that can fail, no routing or configuration tables that have to be kept up-to-date, and it is easy to add a new host to the network.
- It is *inexpensive*: Cable is cheap, and the only other cost is the network adaptor on each host.

---

## WIRELESS LAN (IEEE 802.11)

- Wireless communication is one of the fastest-growing technologies.
- The demand for connecting devices without the use of cables is increasing everywhere.
- Wireless LANs can be found on college campuses, in office buildings, and in many public areas.

## ADVANTAGES OF WLAN / 802.11

1. **Flexibility:** Within radio coverage, nodes can access each other as radio waves can penetrate even partition walls.
2. **Planning:** No prior planning is required for connectivity as long as devices follow standard convention
3. **Design:** Allows to design and develop mobile devices.
4. **Robustness:** Wireless network can survive disaster. If the devices survive, communication can still be established.

## DISADVANTAGES OF WLAN / 802.11

1. **Quality of Service:** Low bandwidth (1 – 10 Mbps), higher error rates due to interference, delay due to error correction and detection.
2. **Cost:** Wireless LAN adapters are costly compared to wired adapters.
3. **Proprietary Solution:** Due to slow standardization process, many solutions are proprietary that limit the homogeneity of operation.
4. **Restriction:** Individual countries have their own radio spectral policies. This restricts the development of the technology
5. **Safety and Security:** Wireless Radio waves may interfere with other devices. Eg; In a hospital, radio waves may interfere with high-tech equipment.

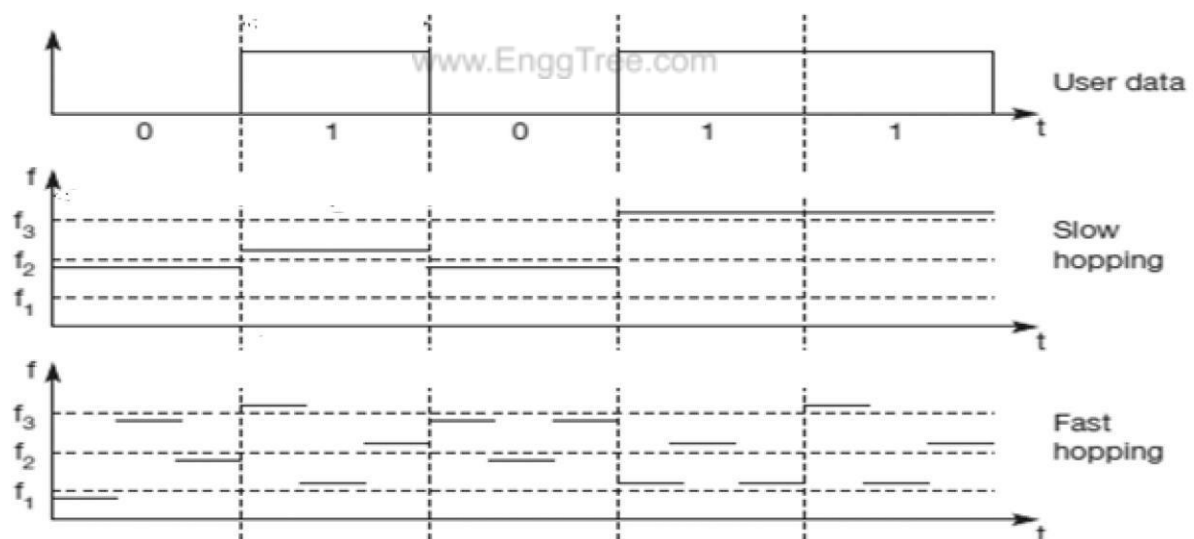


## TECHNOLOGY USED IN WLAN / 802.11

- WLAN's uses Spread Spectrum (SS) technology.
- The idea behind Spread spectrum technique is to spread the signal over a wider frequency band than normal, so as to minimize the impact of interference from other devices.
- There are two types of Spread Spectrum:
  - Frequency Hopping Spread Spectrum (FHSS)
  - Direct Sequence Spread Spectrum (DSSS)

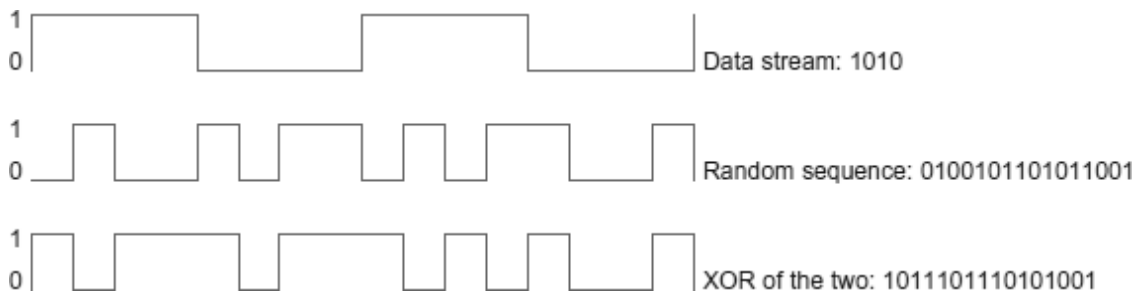
### Frequency Hopping Spread Spectrum (FHSS)

- Frequency hopping is a spread spectrum technique that involves transmitting the signal over a random sequence of frequencies.
- That is, first transmitting at one frequency, then a second, then a third, and so on.
- The random sequence of frequencies is computed by a pseudorandom number generator.
- The receiver uses the same algorithm as the sender and initializes it with the same seed and hence is able to hop frequencies in sync with the transmitter to correctly receive the frame.



### Direct Sequence Spread Spectrum (DSSS)

- Each bit of data is represented by multiple bits in the transmitted signal.
- DSSS takes a user data stream and performs an XOR operation with a pseudo-random number.
- This pseudo random number is called as *chipping sequence*.



**TOPOLOGY IN WLAN / 802.11**

WLANs can be built with either of the following two topologies /architecture:

- Infra-Structure Network Topology
- Ad Hoc Network Topology

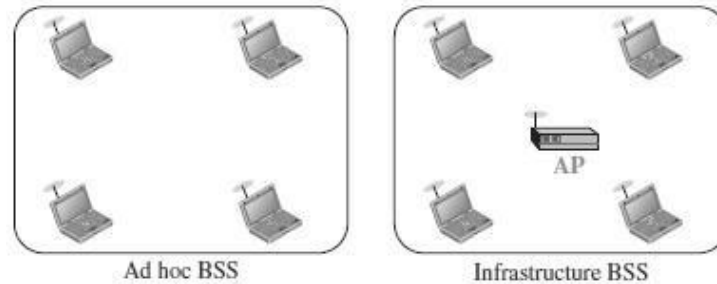
<p><b><u>Infra-Structure Topology</u></b> <b><u>(AP based Topology)</u></b></p> <p>Infrastructure BSS</p> <ul style="list-style-type: none"> <li>• An infrastructure network is the network architecture for providing communication between wireless clients and wired network resources.</li> <li>• The transition of data from the wireless to wired medium occurs via a Base Station called AP(Access Point).</li> <li>• An AP and its associated wireless clients define the coverage area.</li> </ul>	<p><b><u>Ad-Hoc Topology</u></b> <b><u>(Peer-to-Peer Topology)</u></b></p> <p>Ad hoc BSS</p> <ul style="list-style-type: none"> <li>• An adhoc network is the architecture that is used to support mutual communication between wireless clients.</li> <li>• Typically, an ad- hoc network is created spontaneously and does not support access to wired networks.</li> <li>• An adhoc network does not require an AP.</li> </ul>
---	---

## ARCHITECTURE OF WLAN / 802.11

- The standard defines two kinds of services: the Basic Service Set (BSS) and the Extended Service Set (ESS).

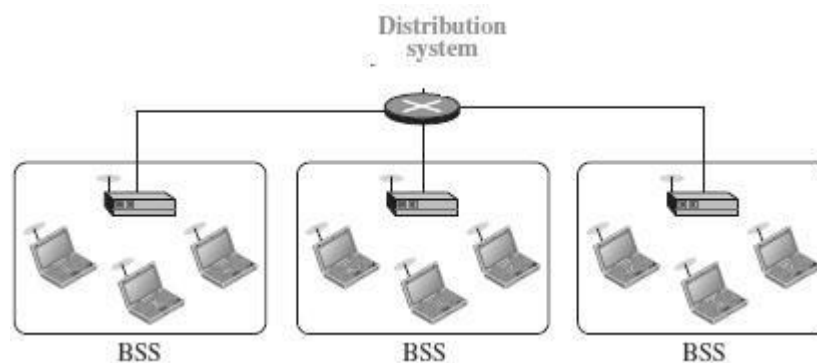
### Basic Service Set (BSS)

- IEEE 802.11 defines the **basic service set (BSS)** as the building blocks of a wireless LAN.
- A basic service set is made of stationary or mobile wireless stations and an optional central base station, known as the *access point (AP)*.



### Extended Service Set (ESS)

- An extended service set (ESS) is made up of two or more BSSs with APs.
- In this case, the BSSs are connected through a *distribution system*, which is a wired or a wireless network.
- The distribution system connects the APs in the BSSs. The extended service set uses two types of stations: mobile and stationary.
- The mobile stations are normal stations inside a BSS.
- The stationary stations are AP stations that are part of a wired LAN.



### Station Types

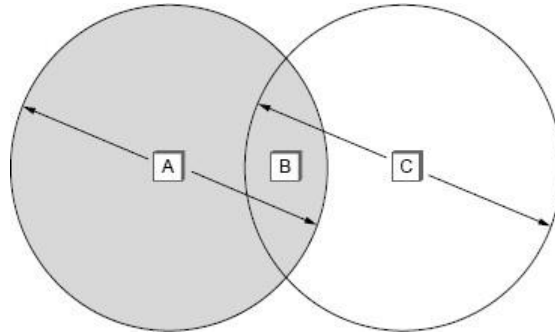
IEEE 802.11 defines three types of stations based on their mobility in a wireless LAN:

1. **No-transition** - A station with no-transition mobility is either stationary (not moving) or moving only inside a BSS.
  2. **BSS-transition** - A station with BSS-transition mobility can move from one BSS to another, but the movement is confined inside one ESS
- ESS-transition** - A station with ESS-transition mobility can move from one ESS to another.

## COLLISION AVOIDANCE IN WLAN / 802.11

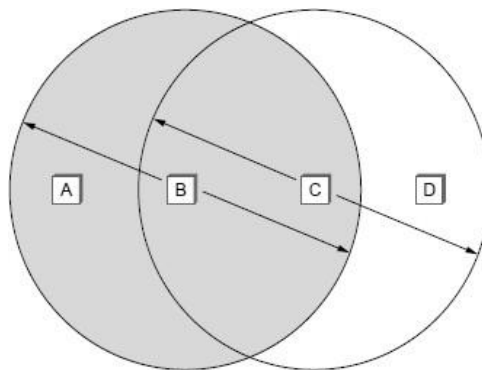
Wireless protocol would follow exactly the same algorithm as the Ethernet—Wait until the link becomes idle before transmitting and back off should a collision occur.

### Hidden Node Problem



- Consider the situation shown in the Figure.
- Here A and C are both within range of B but not with each other.
- Suppose both A and C want to communicate with B and so they each send a frame to B.
- A and C are unaware of each other since their signals do not carry that far.
- These two frames collide with each other at B, but neither A nor C is aware of this collision.
- A and C are said to be *hidden nodes* with respect to each other.

### Exposed Node Problem

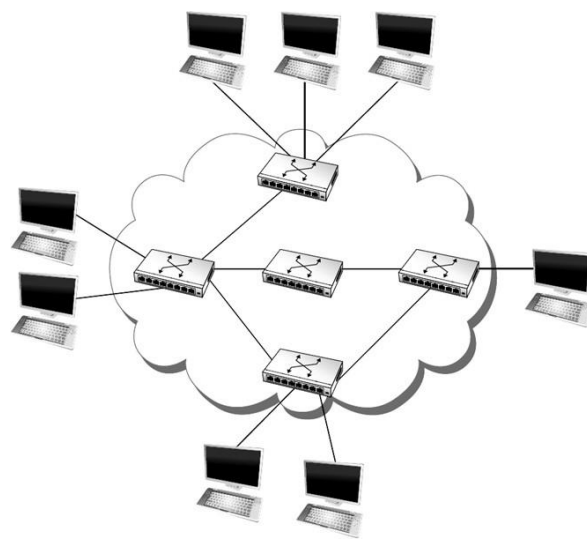


- Each of the four nodes is able to send and receive signals that reach just the nodes to its immediate left and right.
- For example, B can exchange frames with A and C but it cannot reach D, while C can reach B and D but not A.

- Suppose B is sending to A. Node C is aware of this communication because it hears B's transmission.
- If at the same time, C wants to transmit to node D.
- It would be a mistake, however, for C to conclude that it cannot transmit to anyone just because it can hear B's transmission.
- This is not a problem since C's transmission to D will not interfere with A's ability to receive from B.
- This is called exposed problem.
- Although B and C are exposed to each other's signals, there is no interference if B transmits to A while C transmits to D.

## SWITCHING

- The technique of transferring the information from one computer network to another network is known as **switching**.
- Switching in a computer network is achieved by using switches.
- A switch is a small hardware device which is used to join multiple computers together with one local area network (LAN).
- Switches are devices capable of creating temporary connections between two or more devices linked to the switch.
- Switches are used to forward the packets based on MAC addresses.
- A Switch is used to transfer the data only to the device that has been addressed. It verifies the destination address to route the packet appropriately.
- It is operated in full duplex mode.
- It does not broadcast the message as it works with limited bandwidth.



### *Advantages of Switching:*

- Switch increases the bandwidth of the network.
- It reduces the workload on individual PCs as it sends the information to only that device which has been addressed.
- It increases the overall performance of the network by reducing the traffic on the

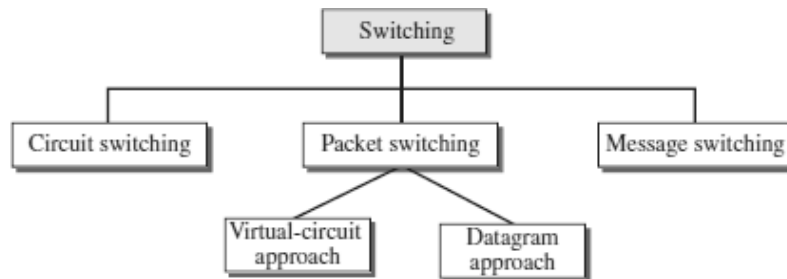
network.

- There will be less frame collision as switch creates the collision domain for each connection.

### ***Disadvantages of Switching:***

- A Switch is more expensive than network bridges.
- A Switch cannot determine the network connectivity issues easily.
- Proper designing and configuration of the switch are required to handle multicast packets.

### **Types of Switching Techniques**



## **CIRCUIT SWITCHING**

- Circuit switching is a switching technique that establishes a dedicated path between sender and receiver.
- In the Circuit Switching Technique, once the connection is established then the dedicated path will remain to exist until the connection is terminated.
- Circuit switching in a network operates in a similar way as the telephone works.
- A complete end-to-end path must exist before the communication takes place.
- In case of circuit switching technique, when any user wants to send the data, voice, video, a request signal is sent to the receiver then the receiver sends back the acknowledgment to ensure the availability of the dedicated path. After receiving the acknowledgment, dedicated path transfers the data.
- Circuit switching is used in public telephone network. It is used for voice transmission.
- Fixed data can be transferred at a time in circuit switching technology.

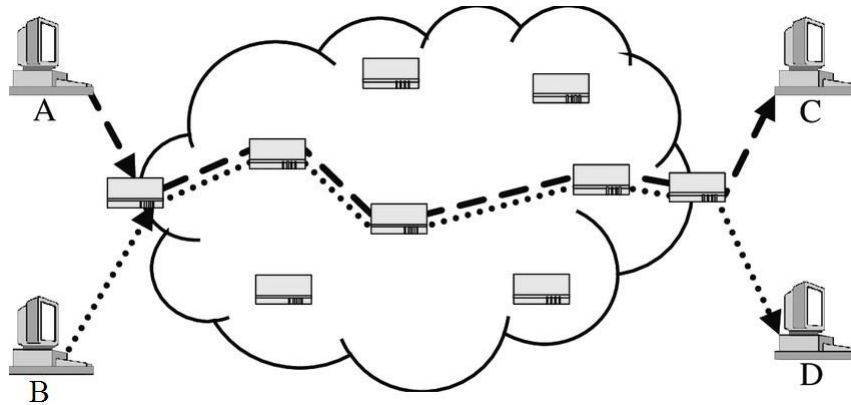
### **Phases in Circuit Switching**

Communication through circuit switching has 3 phases:

1. ***Connection Setup / Establishment*** - In this phase, a dedicated circuit is established from the source to the destination through a number of intermediate switching centres. The sender and receiver transmits communication signals to request and acknowledge establishment of circuits.
2. ***Data transfer*** - Once the circuit has been established, data and voice are transferred

from the source to the destination. The dedicated connection remains as long as the end parties communicate.

3. **Connection teardown / Termination** - When data transfer is complete, the connection is relinquished. The disconnection is initiated by any one of the user. Disconnection involves removal of all intermediate links from the sender to the receiver.



#### **Advantages**

- It is suitable for long continuous transmission, since a continuous transmission route is established, that remains throughout the conversation.
- The dedicated path ensures a steady data rate of communication.
- No intermediate delays are found once the circuit is established. So, they are suitable for real time communication of both voice and data transmission.

#### **Disadvantages**

- Circuit switching establishes a dedicated connection between the end parties. This dedicated connection cannot be used for transmitting any other data, even if the data load is very low.
- Bandwidth requirement is high even in cases of low data volume.
- There is underutilization of system resources. Once resources are allocated to a particular connection, they cannot be used for other connections.
- Time required to establish connection may be high.
- It is more expensive than other switching techniques as a dedicated path is required for each connection.