CS6551 COMPUTER NETWORKS

UNIT-I

1.1 Introduction

The Network is an interconnected set of some objects. For decades we are familiar with the Radio, Television, railway, Highway, Bank and other types of networks. In recent years, the network that is making significant impact in our day-to-day life is the Computer network. By computer network we mean an interconnected set of autonomous computers. The term autonomous implies that the computers can function independent of others. However, these computers can exchange information with each other through the communication network system. Computer networks have emerged as a result of the convergence of two technologies - Computer and Communication. The consequence of this revolutionary merger is the emergence of integrated system that transmits all types of data and information. There is no fundamental difference between data communications and data processing and there are no fundamental differences among data, voice and video communications.

1.1.1 Historical Background

The history of electronic computers is not very old. It came into existence in the early 1950s and during the first two decades of its existence it remained as a centralized system housed in a single large room. In those days the computers were large in size and were operated by trained personnel. To the users it was a remote and mysterious object having no direct communication with the users. Jobs were submitted in the form of punched cards or paper tape and outputs were collected in the form of computer printouts. The submitted jobs were executed by the computer one after the other, which is referred to as batch mode of data processing. In this scenario, there was long delay between the submission of jobs and receipt of the results.

In the 1960s, computer systems were still centralize, but users provided with direct access through interactive terminals connected by point-to-point low-speed data links with the computer. In this situation, a large number of users, some of them located in remote locations could simultaneously access the centralized computer in time-division multiplexed mode. The users could now get immediate interactive feedback from the computer and correct errors immediately. Following the introduction of on-line terminals and time-sharing operating systems, remote terminals were used to use the central computer.

With the advancement of VLSI technology, and particularly, after the invention of microprocessors in the early 1970s, the computers became smaller in size and less expensive, but with significant increase in processing power. New breed of low-cost computers known as mini and personal computers were introduced. Instead of having a single central computer, an organization could now afford to own a number of computers located in different departments and sections.

Side-by-side, riding on the same VLSI technology the communication technology also advanced leading to the worldwide deployment of telephone network, developed primarily for voice communication. An organization having computers located geographically dispersed locations wanted to have data communications for diverse applications. Communication was required among the machines of the same kind for collaboration, for the use of common software or data or for sharing of some costly resources. This led to the development of computer networks by successful integration and cross-fertilization of communications and geographically dispersed computing facilities. One significant development was the APPANET (Advanced Research Projects Agency Network). Starting with four-node experimental network in 1969, it has subsequently grown into a network several thousand computers spanning half of the globe, from Hawaii to Sweden. Most of the present-day concepts such as packet switching evolved from the ARPANET project. The low bandwidth (3KHz on a voice grade line) telephone network was the only generally available communication system available for this type of network.

1.1.2 Network Technologies

There is no generally accepted taxonomy into which all computer networks fit, but two dimensions stand out as important: Transmission Technology and Scale. The classifications based on these two basic approaches are considered in this section.

1.1.2.1 Classification Based on Transmission Technology

Computer networks can be broadly categorized into two types based on transmission technologies:

- Broadcast networks
- Point-to-point networks

Broadcast Networks:

Broadcast network have a single communication channel that is shared by all the machines on the network as shown in Figs.1.1.1 and 1.1.2. All the machines on the network receive short messages, called packets in certain contexts, sent by any machine. An address field within the packet specifies the intended recipient. Upon receiving a packet, machine checks the address field. If packet is intended for itself, it processes the packet; if packet is not intended for itself it is simply ignored.

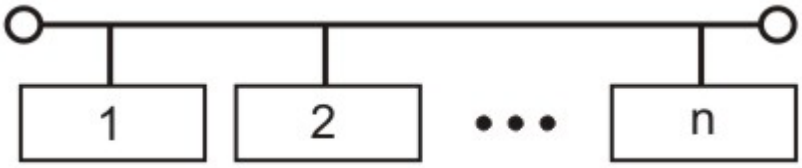


Figure 1.1.1 Example of a broadcast network based on shared bus

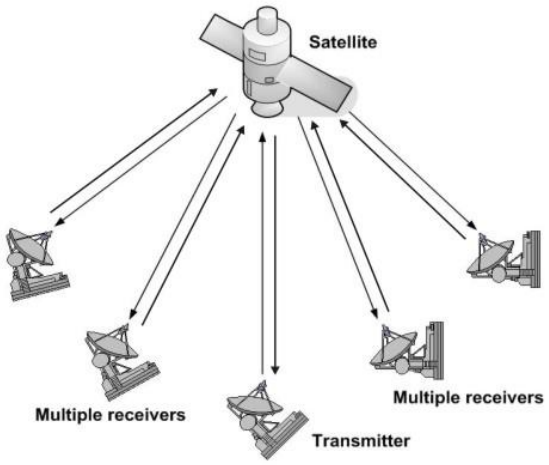


Figure 1.1.2 Example of a broadcast network based on satellite communication

This system generally also allows possibility of addressing the packet to all destinations (all nodes on the network). When such a packet is transmitted and received by all the machines on the network. This mode of operation is known as Broadcast Mode. Some Broadcast systems also support transmission to a sub-set of machines, something known as Multicasting.

Point-to-Point Networks:

A network based on point-to-point communication is shown in Fig. 1.1.3. The end devices that wish to communicate are called stations. The switching devices are called nodes. Some Nodes connect to other nodes and some to attached stations. It uses FDM or TDM for node-to-node communication. There may exist multiple paths between a source-destination pair for better network reliability. The switching nodes are not concerned with the contents of data. Their purpose is to provide a switching facility that will move data from node to node until they reach the destination.

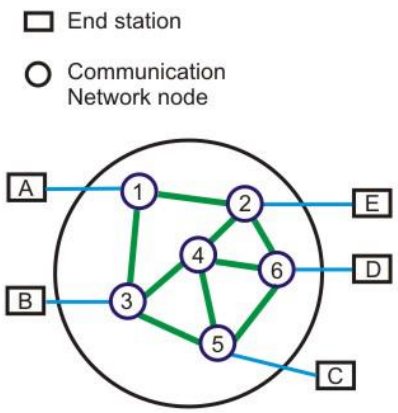


Figure 1.1.3 Communication network based on point-to-point communication

As a general rule (although there are many exceptions), smaller, geographically localized networks tend to

use broadcasting, whereas larger networks normally use are point-to-point communication.

1.1.2.2 Classification based on Scale

Alternative criteria for classifying networks are their scale. They are divided into Local Area (LAN), Metropolitan Area Network (MAN) and Wide Area Networks (WAN).

Local Area Network (LAN):

LAN is usually privately owned and links the devices in a single office, building or campus of up to few kilometers in size. These are used to share resources (may be hardware or software resources) and to exchange information. LANs are distinguished from other kinds of networks by three categories: their size, transmission technology and topology.

LANs are restricted in size, which means that their worst-case transmission time is bounded and known in advance. Hence this is more reliable as compared to MAN and WAN. Knowing this bound makes it possible to use certain kinds of design that would not otherwise be possible. It also simplifies network management.

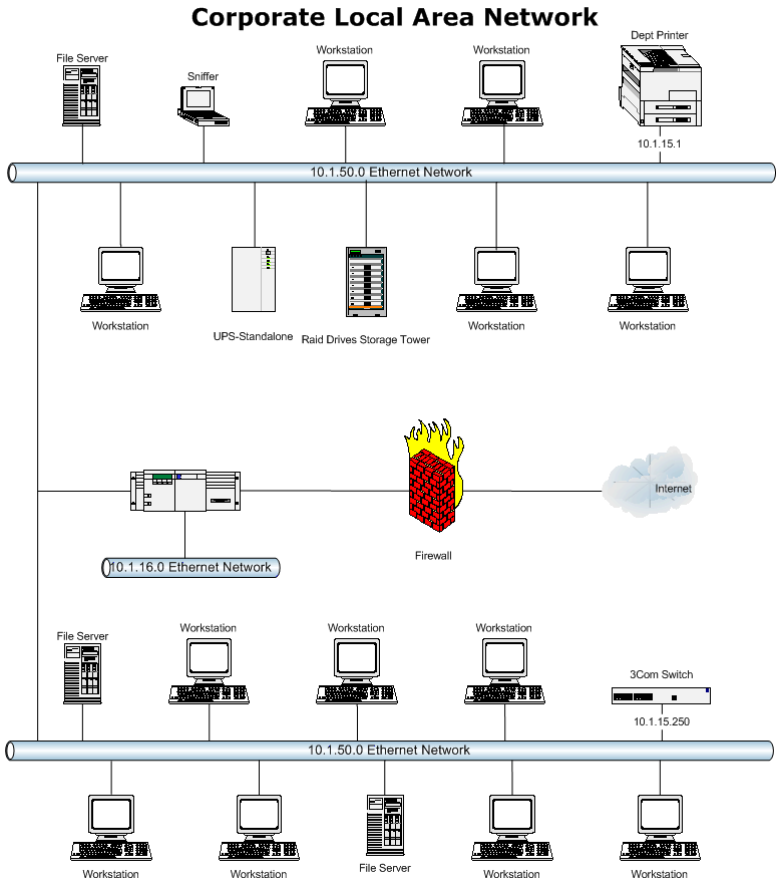


Figure 1.1.4 Local Area Network

LAN typically used transmission technology consisting of single cable to which all machines are

connected. Traditional LANs run at speeds of 10 to 100 Mbps (but now much higher speeds can be achieved). The most common LAN topologies are bus, ring and star. A typical LAN is shown in Fig. 1.1.4

Metropolitan Area Networks (MAN) :

MAN is designed to extend over the entire city. It may be a single network as a cable TV network or it may be means of connecting a number of LANs into a larger network so that resources may be shared as shown in Fig. 1.1.5. For example, a company can use a MAN to connect the LANs in all its offices in a city. MAN is wholly owned and operated by a private company or may be a service provided by a public company.

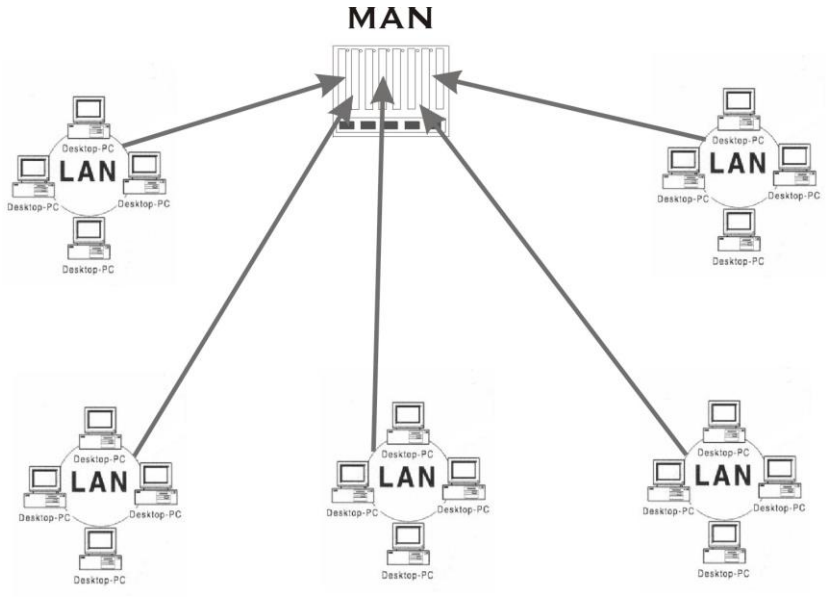


Figure 1.1.5 Metropolitan Area Networks (MAN)

The main reason for distinguishing MANs as a special category is that a standard has been adopted for them. It is DQDB (Distributed Queue Dual Bus) or IEEE 802.6.

Wide Area Network (WAN):

WAN provides long-distance transmission of data, voice, image and information over large geographical areas that may comprise a country, continent or even the whole world. In contrast to LANs, WANs may utilize public, leased or private communication devices, usually in combinations, and can therefore span an unlimited number of miles as shown in Fig. 1.1.6. A WAN that is wholly owned and used by a single company is often referred to as enterprise network.

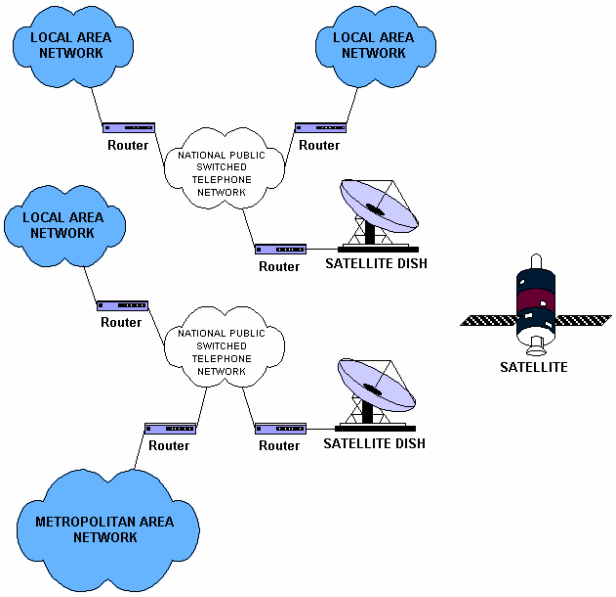


Figure 1.1.6 Wide Area Network

The Internet:

Internet is a collection of networks or network of networks. Various networks such as LAN and WAN connected through suitable hardware and software to work in a seamless manner. Schematic diagram of the Internet is shown in Fig. 1.1.7. It allows various applications such as e-mail, file transfer, remote log-in, World Wide Web, Multimedia, etc run across the internet. The basic difference between WAN and Internet is that WAN is owned by a single organization while internet is not so. But with the time the line between WAN and Internet is shrinking, and these terms are sometimes used interchangeably.

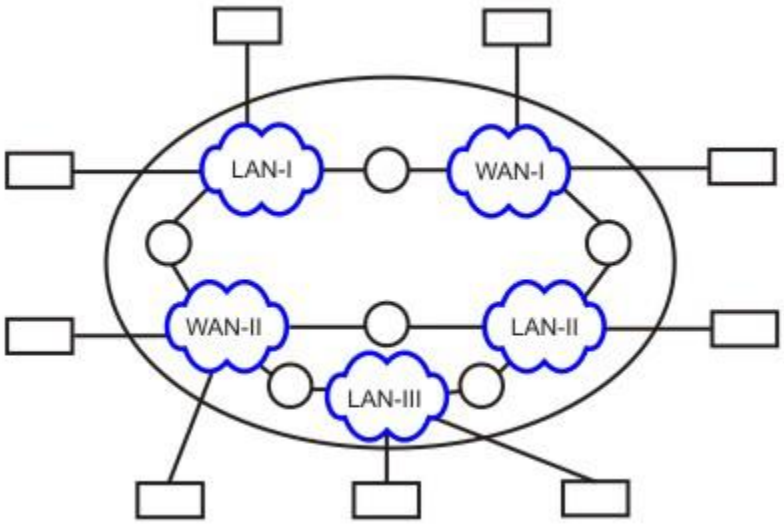


Figure 1.1.7 Internet – network of networks

1.1.3 Applications

In a short period of time computer networks have become an indispensable part of business, industry, entertainment as well as a common-man's life. These applications have changed tremendously from time and the motivation for building these networks are all essentially economic and technological.

Initially, computer network was developed for **defense purpose**, to have a secure communication network that can even withstand a nuclear attack. After a decade or so, companies, in various fields, started using computer networks for keeping track of inventories, monitor productivity, communication between their branches located at different locations. For example, Railways started using computer networks by connecting their nationwide reservation counters to provide the facility of reservation and enquiry from anywhere across the country.

And now after almost two decades, computer networks have entered a new dimension; they are now an integral part of the society and people. In 1990s, computer network started delivering services to private individuals at home. These services and motivation for using them are quite different. Some of the services are access to remote information, person-person communication, and interactive entertainment. So, some of the applications of computer networks that we can see around us today are as follows:

Marketing and sales: Computer networks are used extensively in both marketing and sales organizations. Marketing professionals use them to collect, exchange, and analyze data related to customer needs and product development cycles. Sales application includes teleshopping, which uses order-entry computers or telephones connected to order processing network, and online-reservation services for hotels, airlines and so on.

Financial services: Today's financial services are totally depended on computer networks. Application includes credit history searches, foreign exchange and investment services, and electronic fund transfer, which allow user to transfer money without going into a bank (an automated teller machine is an example of electronic fund transfer, automatic pay-check is another).

Manufacturing: Computer networks are used in many aspects of manufacturing including manufacturing process itself. Two of them that use network to provide essential services are computer-aided design (CAD) and computer-assisted manufacturing (CAM), both of which allow multiple users to work on a project simultaneously.

Directory services: Directory services allow list of files to be stored in central location to speed worldwide search operations.

Information services: A Network information service includes bulletin boards and data banks. A World Wide Web site offering technical specification for a new product is an information service.

Electronic data interchange (EDI): EDI allows business information, including documents such as purchase orders and invoices, to be transferred without using paper.

Electronic mail: probably it's the most widely used computer network application.

Teleconferencing: Teleconferencing allows conference to occur without the participants being in the same place. Applications include simple text conferencing (where participants communicate through their

normal keyboards and monitor) and video conferencing where participants can even see as well as talk to other fellow participants. Different types of equipments are used for video conferencing depending on what quality of the motion you want to capture (whether you want just to see the face of other fellow participants or do you want to see the exact facial expression).

Voice over IP: Computer networks are also used to provide voice communication. This kind of voice communication is pretty cheap as compared to the normal telephonic conversation.

Video on demand: Future services provided by the cable television networks may include video on request where a person can request for a particular movie or any clip at anytime he wish to see.

In Summary, The main area of applications can be broadly classified into following categories:

- Scientific and Technical Computing
  - Client Server Model, Distributed Processing
- Parallel Processing, Communication Media
- Commercial
  - Advertisement, Telemarketing, Teleconferencing
  - Worldwide Financial Services
  - Network for the People (this is the most widely used application nowadays)
  - Telemedicine, Distance Education, Access to Remote Information,
  - Person-to-Person Communication, Interactive Entertainment

1.2 Network architecture:

Network architecture is the collection of hardware and software components arranged to form a complete network system. It is a framework for the specification of a network's physical components and their functional organization and configuration, its operational principles and procedures, as well as data formats used in its operation.

In computing, the network architecture is a characteristic of a computer networks. The most prominent architecture today is evident in the framework of the Internet, which is based on the Internet protocol suite.

In telecommunication, the specification of a network architecture may also include a detailed description of products and services delivered via a communications network, as well as detailed rate and billing structures under which services are compensated.

In distinct usage in distributed computing, network architecture is also sometimes used as a synonym for the structure and classification of distributed application architecture, as the participating nodes in a distributed application are often referred to as a *network*. For example, the applications architecture of the public switched telephone networks (PSTN) has been termed the Advanced Intelligent Networks. There are any number of specific classifications but all lie on a continuum between the Dumb Network (e.g: Internet) and the intelligent computer networks (e.g., the telephone network). Other

networks contain various elements of these two classical types to make them suitable for various types of applications. Recently the context networks, which is a synthesis of the two, has gained much interest with its ability to combine the best elements of both.

1.2.1 The need for protocol architecture:

When computers, terminals, and/or other data processing devices exchange data, the procedures involved can be quite complex. Consider, for example, the transfer of a file between two computers. There must be a data path between the two computers, either directly or via a communication network. It is clear that there must be a high degree of cooperation between the two computer systems. Instead of implementing the logic for this as a single module, the task is broken up into subtasks, each of which is implemented separately.

In protocol architecture, the modules are arranged in a vertical stack of layers. Each layer in the stack performs a related subset of the functions required to communicate with another system. It relies on the next lower layer to perform more primitive functions and to conceal the details of those functions. It provides services to the next higher layer. Ideally, layers should be defined so that changes in one layer do not require changes in other layers. Figure 1.2.1 illustrates a 5-layer network architecture. Layer (level) *n* on one computer carries on communication with layer *n* on another computer. The set of rules and conventions that encompasses electrical, mechanical and functional characteristics of a data link, as well as the control procedures for such communication is called the *layer n protocol*. The communication between two layers at the same level (layer *n*, *n* ¡¡= 1) of two different computers is called *virtual communication*. Here, each layer passes data and control information to the layer immediately below it, until the lowest layer (layer 1). At layer 1, information from one computer is physically transferred to layer 1 of the other *(physical communication)*.

The *interface* between each pair of adjacent layers defines which operations and services the lower layer offers to the upper one. The *network architecture* thus can be defined as the set of layers and protocols.

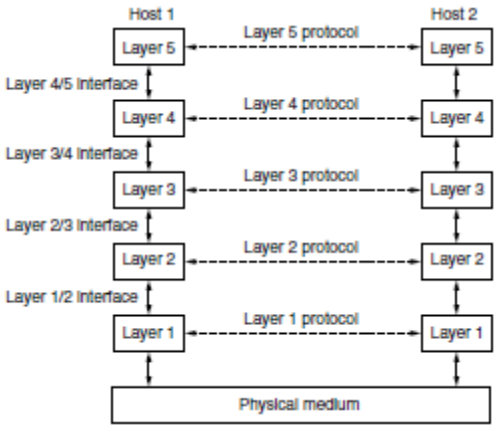


Figure 1.2.1: Layers, Protocols, interfaces

In summary, Layered architecture is needed

1. To make the design process easy by breaking unmanageable tasks into several smaller and manageable tasks (by divide-and-conquer approach).

2. Modularity and clear interfaces, so as to provide comparability between the different providers' components.

3. Ensure independence of layers, so that implementation of each layer can be changed or modified without affecting other layers.

4. Each layer can be analyzed and tested independently of all other layers.

The two most widely referenced architecture models are:

- Open System interconnection(OSI) Model
- TCP/IP or Internet Model

The ISO model:

The Open System Interconnection (OSI) reference model describes how information from a software application in one computer moves through a network medium to a software application in another computer. The OSI reference model is a conceptual model composed of seven layers, each specifying particular network functions. The model was developed by the International Organization for Standardization (ISO) in 1984, and it is now considered the primary architectural model for inter-computer communications. The OSI model divides the tasks involved with moving information between networked computers into seven smaller, more manageable task groups. A task or group of tasks is then assigned to each of the seven OSI layers. Each layer is reasonably self-contained so that the tasks assigned to each layer can be implemented independently. This enables the solutions offered by one layer to be updated without adversely affecting the other layers. Figure 1.2.2 shows the reference model of the Open Systems Interconnection (OSI).

Layer 1: The physical layer:

This layer is concerned with transmitting an electrical signal representation of data over a communication link. Typical conventions would be: voltage levels used to represent a “1” and a “0”, duration of each bit, transmission rate, mode of transmission, and functions of pins in a connector. An example of a physical layer protocol is the RS-232 standard.

Layer 2: The data link layer:

This layer is concerned with error-free transmission of data units. The data unit is an abbreviation of the official name of *data-link-service-data-units;* it is sometimes called the *data frame*. The function of the data link layer is to break the input data stream into data frames, transmit the frames sequentially, and process the *acknowledgement frame* sent back by the receiver. Data frames from this level when transferred to layer 3 are assumed to be error free.

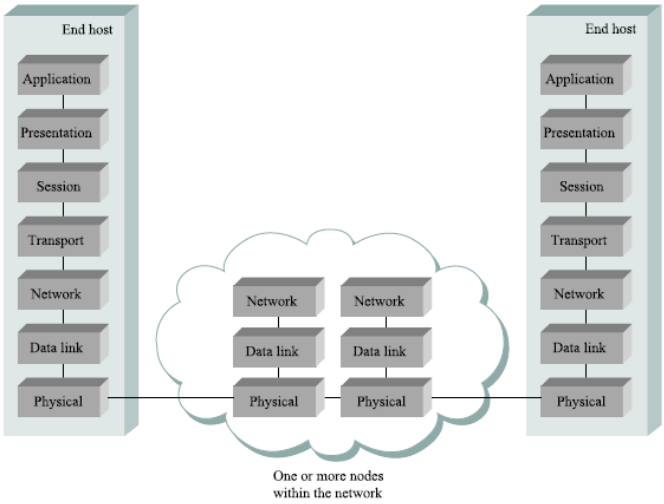


Figure 1.2.2: OSI reference model

Layer 3: The network layer:

This layer is the *network control* layer, and is sometimes called the *communication subnet layer*. It is concerned with intra-network operation such as addressing and routing within the subnet. Basically, messages from the source host are converted to *packets*. The packets are then routed to their proper destinations.

Layer 4: The transport layer:

This layer is a *transport end-to-end control layer* (i.e. source-to-destination). A program on the source computer communicates with a similar program on the destination computer using the message headers and control messages, whereas all the lower layers are only concerned with communication between a computer and its immediate neighbours, not the ultimate source and destination computers. The transport layer is often implemented as part of the operating system. The data link and physical layers are normally implemented in hardware.

Layer 5: The session layer:

The session layer is the user's interface into the network. This layer supports the dialogue through session control, if services can be allocated. A connection between users is usually called a *session*. A session might be used to allow a user to log into a system or to transfer files between two computers. A session can only be established if the user provides the remote addresses to be connected. The difference between session addresses and transport addresses is that session addresses are intended for users and their programs, whereas transport addresses are intended for transport stations.

Layer 6: The presentation layer:

This layer is concerned with transformation of transferred information. The controls include message compression, encryption, peripheral device coding and formatting.

Layer 7: The application layer:

This layer is concerned with the application and system activities. The content of the application layer is up to the individual user.

TCP/IP Model:

The TCP/IP protocol architecture is a result of protocol research and development conducted on the experimental packet-switched network, ARPANET, funded by the Defense Advanced Research Projects Agency (DARPA), and is generally referred to as the TCP/IP protocol suite. This protocol suite consists of a large collection of protocols that have been issued as Internet standards by the Internet Activities Board (IAB).

In TCP/IP model the communication task organized into five relatively independent layers as given below:

- Physical layer
- Network access layer
- Internet layer
- Host-to-host, or transport layer
- Application layer

The **physical layer** covers the physical interface between a data transmission device (e.g., workstation, computer) and a transmission medium or network. This layer is concerned with specifying the characteristics of the transmission medium, the nature of the signals, the data rate, and related matters.

The **network access layer** is concerned with the exchange of data between an end system (server, workstation, etc.) and the network to which it is attached. The sending computer must provide the network with the address of the destination computer, so that the network may route the data to the appropriate destination. The sending computer may wish to invoke certain services, such as priority, that might be provided by the network. The specific software used at this layer depends on the type of network to be used; different standards have been developed for circuit switching, packet switching (e.g., frame relay), LANs (e.g., Ethernet), and others. The network access layer is concerned with access to and routing data across a network for two end systems attached to the same network.

The **internet layer** uses internet protocol for exchange of data between two end systems (server, workstation, etc.) attached to the different networks. This protocol is implemented not only in the end systems but also in routers. A router is a processor that connects two networks and whose primary function is to relay data from one network to the other on its route from the source to the destination end system.

Regardless of the nature of the applications that are exchanging data, there is usually a requirement that data be exchanged reliably. the mechanisms for providing reliability are essentially independent of the nature of the applications. Thus, it makes sense to collect those mechanisms in a common layer shared by all applications; this is referred to as the **host-to-host layer**, or **transport layer**. The Transmission Control Protocol (TCP) is the most commonly used protocol to provide this functionality.

Finally, the **application layer** contains the logic needed to support the various user applications. For each different type of application, such as file transfer, a separate module is needed that is peculiar to that application.

1.3 PHYSICAL LINKS:

In telecommunications a **link** is the *communications channel* that connects two or more communicating devices. This link may be an actual physical link or it may be a logical link that uses one or more actual physical links. When the link is a logical link the type of physical link should always be specified (e.g., data link, uplink, downlink, fiber optic link, pt to pt link etc. etc.) This term is widely used in computer networking to refer to the communications facilities that connect nodes of a network.

1.3.1 Types of links:

- **Point-to-point**

A **point-to-point link** is a dedicated link that connects exactly two communication facilities (e.g., two nodes of a network, an intercom station at an entryway with a single internal intercom station, a radio path between two points, etc.).

- **Broadcast**

Broadcast links connect two or more nodes and support *broadcast transmission*, where one node can transmit so that all other nodes can receive the same transmission. Ethernet is an example.

- **Multipoint**

Also known as a "multidrop" link, a **multipoint link** is a link that connects *two or more* nodes. Also known as **general topology** networks, these include ATM and Frame Relay links, as well as X.25 networks when used as links for a network layer protocol like Internet protocol. Unlike broadcast links, there is no mechanism to efficiently send a single message to all other nodes without copying and retransmitting the message.

- **Point-to-multipoint**

A point to multipoint link is a specific type of *multipoint link* which consists of a central *connection endpoint* (CE) that is connected to multiple peripheral CEs. Any transmission of data that originates from the central CE is received by all of the peripheral CEs while any transmission of data that originates from any of the peripheral CEs is only received by the central CE. This term is also often used as a synonym for **multipoint**, as defined above.

Private and Public - Accessibility and Ownership:

Links are often referred to by terms which refer to the ownership and / or accessibility of the link.

- A **private link** is a link that is either owned by a specific entity or a link that is only accessible by a specific entity.
- A **public link** is a link that uses the Public switched telephone network or other public utility or entity to provide the link and which may also be accessible by anyone.

1.3.2 Transmission Media:

Transmission media can be defined as physical path between transmitter and receiver in a data transmission system. And it may be classified into two types as shown in Fig. 1.3.1.

• Guided: Transmission capacity depends critically on the medium, the length, and whether the medium is point-to-point or multipoint (e.g. LAN). Examples are co-axial cable, twisted pair, and optical fiber.

• Unguided: provides a means for transmitting electro-magnetic signals but do not guide them. Example: wireless transmission.

Characteristics and quality of data transmission are determined by medium and signal characteristics. For guided media, the medium is more important in determining the limitations of transmission. While in case of unguided media, the bandwidth of the signal produced by the transmitting antenna and the size of the antenna is more important than the medium. Signals at lower frequencies are omni-directional (propagate in all directions). For higher frequencies, focusing the signals into a directional beam is possible. These properties determine what kind of media one should use in a particular application.

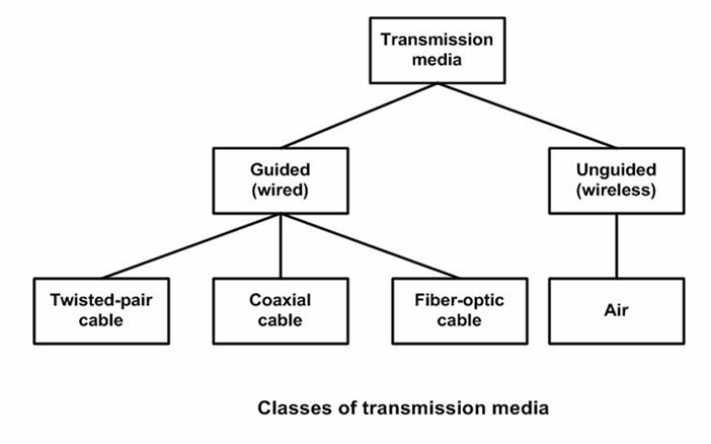


Figure 1.3.1 Classification of the transmission media

Guided transmission media

Twisted Pair:

Figure 1.3.2 CAT5 cable (twisted cable)

In twisted pair technology, two copper wires are strung between two points:

• The two wires are typically ``twisted'' together in a helix to reduce interference between the two conductors as shown in Fig.1.3.2. Twisting decreases the cross-talk interference between adjacent pairs in a cable. Typically, a number of pairs are bundled together into a cable by wrapping them in a tough protective sheath.

• Can carry both analog and digital signals. Actually, they carry only analog signals. However, the ``analog" signals can very closely correspond to the square waves representing bits, so we often think of them as carrying digital data.

• Data rates of several Mbps common.

• Spans distances of several kilometers.

• Data rate determined by wire thickness and length. In addition, shielding to eliminate interference from other wires impacts signal-to-noise ratio, and ultimately, the data rate.

• Good, low-cost communication. Indeed, many sites already have twisted pair installed in offices -- existing phone lines!

Typical characteristics: Twisted-pair can be used for both analog and digital communication. The data rate that can be supported over a twisted-pair is inversely proportional to the square of the line length. Maximum transmission distance of 1 Km can be achieved for data rates up to 1 Mb/s. For analog voice signals, amplifiers are required about every 6 Km and for digital signals, repeaters are needed for about 2 Km. To reduce interference, the twisted pair can be shielded with metallic braid. This type of wire is known as Shielded Twisted-Pair (STP) and the other form is known as Unshielded Twisted-Pair (UTP).

Use: The oldest and the most popular use of twisted pair are in telephony. In LAN it is commonly used for point-to-point short distance communication (say, 100m) within a building or a room.

Base band Coaxial:

With ``coax", the medium consists of a copper core surrounded by insulating material and a braided outer conductor as shown in Fig. 1.3.3. The term base band indicates digital transmission (as opposed to broadband analog).

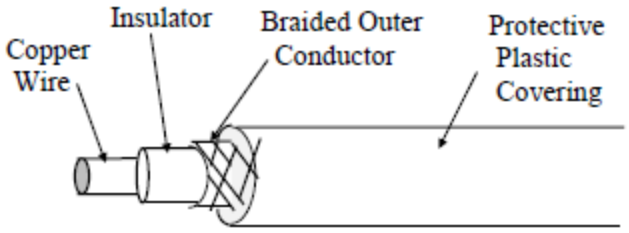


Figure 1.3.3 Co-axial cable

Physical connection consists of metal pin touching the copper core. There are two common ways to connect to a coaxial cable:

1. With vampire taps, a metal pin is inserted into the copper core. A special tool drills a hole into the cable, removing a small section of the insulation, and a special connector is screwed into the hole. The tap makes contact with the copper core.

2. With a T-junction, the cable is cut in half, and both halves connect to the T-junction. A T-connector is analogous to the signal splitters used to hook up multiple TVs to the same cable wire.

Characteristics: Co-axial cable has superior frequency characteristics compared to twisted-pair and can be

used for both analog and digital signaling. In baseband LAN, the data rates lies in the range of 1 KHz to 20 MHz over a distance in the range of 1 Km. Co-axial cables typically have a diameter of 3/8". Coaxial cables are used both for baseband and broadband communication.

For broadband CATV application coaxial cable of 1/2" diameter and 75 ohm impedance is used. This cable offers bandwidths of 300 to 400 MHz facilitating high-speed data communication with low bit-error rate. In broadband signaling, signal propagates only in one direction, in contrast to propagation in both directions in baseband signaling. Broadband cabling uses either dual-cable scheme or single-cable scheme with a head end to facilitate flow of signal in one direction. Because of the shielded, concentric construction, co-axial cable is less susceptible to interference and cross talk than the twisted-pair.

For long distance communication, repeaters are needed for every kilometer or so. Data rate depends on physical properties of cable, but 10 Mbps is typical.

Use: One of the most popular use of co-axial cable is in cable TV (CATV) for the distribution of TV signals. Another importance use of co-axial cable is in LAN.

Broadband Coaxial:

The term broadband refers to analog transmission over coaxial cable. (Note, however, that the telephone folks use broadband to refer to any channel wider than 4 kHz). The technology:

• Typically bandwidth of 300 MHz, total data rate of about 150 Mbps.

• Operates at distances up to 100 km (metropolitan area!).

• Uses analog signaling.

• Technology used in cable television. Thus, it is already available at sites such as universities that may have TV classes.

• Total available spectrum typically divided into smaller channels of 6 MHz each. That is, to get more than 6MHz of bandwidth, you have to use two smaller channels and somehow combine the signals.

• Requires amplifiers to boost signal strength; because amplifiers are one way, data flows in only one direction.

Two types of systems have emerged:

1. Dual cable systems use two cables, one for transmission in each direction:

    o One cable is used for receiving data.

    o Second cable used to communicate with headend. When a node wishes to transmit data, it sends the data to a special node called the headend. The headend then resends the data on the first cable. Thus, the headend acts as a root of the tree, and all data must be sent to the root for redistribution to the other nodes.

2. Midsplit systems divide the raw channel into two smaller channels, with each subchannel having the same purpose as above. Which is better, broadband or base band? There is rarely a simple answer to such questions. Base band is simple to install, interfaces are inexpensive, but doesn't have the same range. Broadband is more complicated, more expensive, and requires regular adjustment by a trained technician, but offers more services (e.g., it carries audio and video too).

Fiber Optics:

In fiber optic technology, the medium consists of a hair-width strand of silicon or glass, and the signal consists of pulses of light. For instance, a pulse of light means ``1", lack of pulse means ``0". It has a cylindrical shape and consists of three concentric sections: the core, the cladding, and the jacket as shown in Fig. 1.3.4.

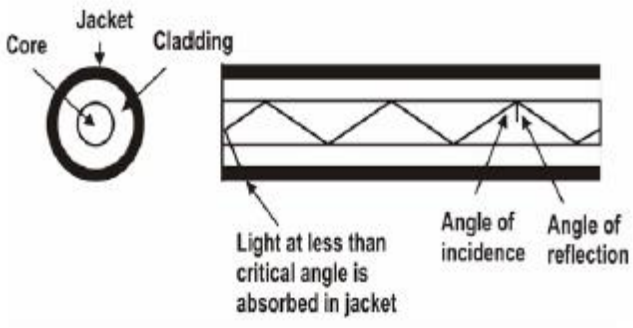


Figure 1.3.4 Optical Fiber

The core, innermost section consists of a single solid dielectric cylinder of diameter d1 and of refractive index n1. The core is surrounded by a solid dielectric cladding of refractive index n2 that is less than n1. As a consequence, the light is propagated through multiple total internal reflection. The core material is usually made of ultra pure fused silica or glass and the cladding is either made of glass or plastic. The cladding is surrounded by a jacket made of plastic. The jacket is used to protect against moisture, abrasion, crushing and other environmental hazards.

Three components are required:

1. Fiber medium: Current technology carries light pulses for tremendous distances (e.g., 100s of kilometers) with virtually no signal loss.
2. Light source: typically a Light Emitting Diode (LED) or laser diode. Running current through the material generates a pulse of light.
3. A photo diode light detector, which converts light pulses into electrical signals.

Advantages:

1. Very high data rate, low error rate. 1000 Mbps (1 Gbps) over distances of kilometers common. Error rates are so low they are almost negligible.
2. Difficult to tap, this makes it hard for unauthorized taps as well. This is responsible for higher reliability of this medium.
3. Much thinner (per logical phone line) than existing copper circuits. Because of its thinness, phone companies can replace thick copper wiring with fibers having much more capacity for same volume. This is important because it means that aggregate phone capacity can be upgraded without the need for finding more physical space to hire the new cables.

4. Not susceptible to electrical interference (lightning) or corrosion (rust).

5. Greater repeater distance than coax.

Disadvantages:

• Difficult to tap. It really is point-to-point technology. In contrast, tapping into coax is trivial. No special training or expensive tools or parts are required.

• One-way channel. Two fibers needed to get full duplex (both ways) communication.

Optical Fiber works in three different types of modes (or we can say that we have 3 types of communication using Optical fiber). Optical fibers are available in two varieties; Multi-Mode Fiber (MMF) and Single-Mode Fiber (SMF). For multi-mode fiber the core and cladding diameter lies in the range 50-200µm and 125-400µm, respectively. Whereas in single-mode fiber, the core and cladding diameters lie in the range 8-12µm and 125µm, respectively.

Single-mode fibers are also known as Mono-Mode Fiber. Moreover, both single-mode and multi-mode fibers can have two types; step index and graded index. In the former case the refractive index of the core is uniform throughout and at the core cladding boundary there is an abrupt change in refractive index. In the later case, the refractive index of the core varies radially from the centre to the core-cladding boundary from n1 to n2 in a linear manner. Fig. 1.3.5 shows the optical fiber transmission modes.

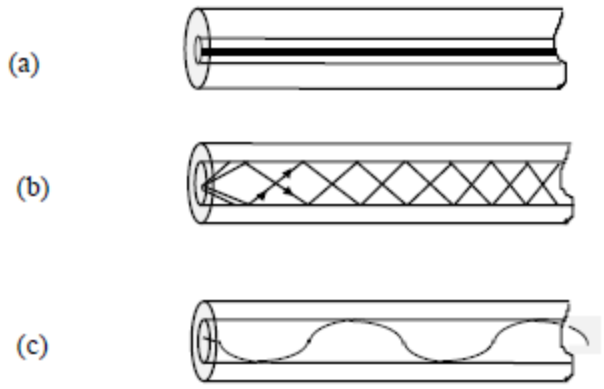


Figure 1.3.5 Schematics of three optical fiber types, (a) Single-mode step-index, (b) Multi-mode step-index, and (c) Multi-mode graded-index

Characteristics: Optical fiber acts as a dielectric waveguide that operates at optical frequencies (1014 to 1015 Hz). Three frequency bands centered around 850,1300 and 1500 nanometers are used for best results. When light is applied at one end of the optical fiber core, it reaches the other end by means of total internal reflection because of the choice of refractive index of core and cladding material (n1 > n2). The light source can be either light emitting diode (LED) or injection laser diode (ILD). These semiconductor devices emit a beam of light when a voltage is applied across the device. At the receiving end, a photodiode can be used to detect the signal-encoded light. Either PIN detector or APD (Avalanche photodiode) detector can be used as the light detector.

In a multi-mode fiber, the quality of signal-encoded light deteriorates more rapidly than single-mode fiber, because of interference of many light rays. As a consequence, single-mode fiber allows longer distances without repeater. For multi-mode fiber, the typical maximum length of the cable without a repeater is 2km, whereas for single-mode fiber it is 20km.

Fiber Uses: Because of greater bandwidth (2Gbps), smaller diameter, lighter weight, low attenuation, immunity to electromagnetic interference and longer repeater spacing, optical fiber cables are finding widespread use in long-distance telecommunications. Especially, the single mode fiber is suitable for this purpose. Fiber optic cables are also used in high-speed LAN applications. Multi-mode fiber is commonly used in LAN.

• Long-haul trunks-increasingly common in telephone network (Sprint ads)
• Metropolitan trunks-without repeaters (average 8 miles in length)
• Rural exchange trunks-link towns and villages
• Local loops-direct from central exchange to a subscriber (business or home)
• Local area networks-100Mbps ring networks.

Unguided Transmission

Unguided transmission is used when running a physical cable (either fiber or copper) between two end points is not possible. For example, running wires between buildings is probably not legal if the building is separated by a public street.

Infrared signals typically used for short distances (across the street or within same room), Microwave signals commonly used for longer distances (10's of km). Sender and receiver use some sort of dish antenna as shown in Fig. 1.3.6.

Figure 1.3.6 Communication using Terrestrial Microwave

Difficulties:

1. Weather interferes with signals. For instance, clouds, rain, lightning, etc. may adversely affect communication.
2. Radio transmissions easy to tap. A big concern for companies worried about competitors stealing plans.
3. Signals bouncing off of structures may lead to out-of-phase signals that the receiver must filter out.

Satellite Communication:

Satellite communication is based on ideas similar to those used for line-of-sight. A communication satellite is essentially a big microwave repeater or relay station in the sky. Microwave signals from a ground station is picked up by a transponder, amplifies the signal and rebroadcasts it in another frequency, which can be received by ground stations at long distances as shown in Fig. 1.3.7.

To keep the satellite stationary with respect to the ground based stations, the satellite is placed in a geostationary orbit above the equator at an altitude of about 36,000 km. As the spacing between two satellites on the equatorial plane should not be closer than 40, there can be 360/4 = 90 communication satellites in the sky at a time. A satellite can be used for point-to-point communication between two ground-based stations or it can be used to broadcast a signal received from one station to many ground-based stations as shown in Fig. 1.3.8. Number of geo-synchronous satellites limited (about 90 total, to minimize interference). International agreements regulate how satellites are used, and how frequencies are allocated. Weather affects certain frequencies. Satellite transmission differs from terrestrial communication in another important way: One-way propagation delay is roughly 270 ms. In interactive terms, propagation delay alone inserts a 1 second delay between typing a character and receiving its echo.

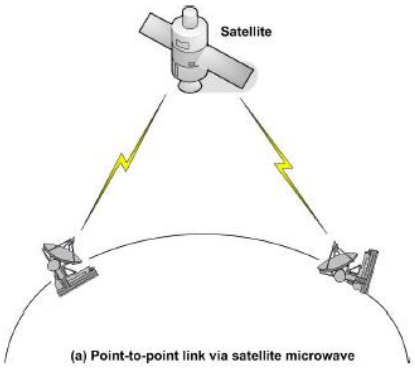


Figure 1.3.7 Satellite Microwave Communication: point –to- point

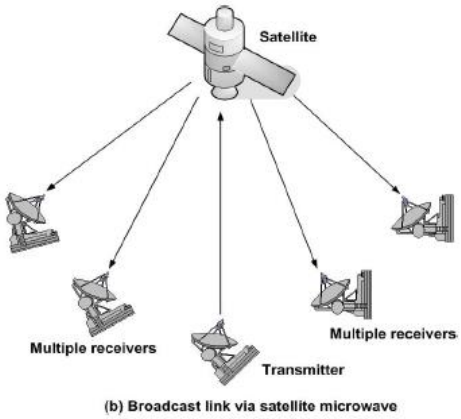


Figure 1.3.8 Satellite Microwave Communication: Broadcast links

Characteristics: Optimum frequency range for satellite communication is 1 to 10 GHz. The most popular frequency band is referred to as 4/6 band, which uses 3.7 to 4.2 GHz for down link and 5.925 to 6.425 for uplink transmissions. The 500 MHz bandwidth is usually split over a dozen transponders, each with 36 MHz bandwidth. Each 36 MHz bandwidth is shared by time division multiplexing. As this preferred band is already saturated, the next highest band available is referred to as 12/14 GHz. It uses 14 to 14.5GHz for upward transmission and 11.7 to 12.2 GHz for downward transmissions. Communication satellites have several unique properties. The most important is the long communication delay for the round trip (about 270 ms) because of the long distance (about 72,000 km) the signal has to travel between two earth stations. This poses a number of problems, which are to be tackled for successful and reliable communication.

Another interesting property of satellite communication is its broadcast capability. All stations under the downward beam can receive the transmission. It may be necessary to send encrypted data to protect against piracy.

Use: Now-a-days communication satellites are not only used to handle telephone, telex and television traffic over long distances, but are used to support various internet based services such as e-mail, FTP, World Wide Web (WWW), etc. New types of services, based on communication satellites, are emerging.

Comparison/contrast with other technologies:

1. Propagation delay very high. On LANs, for example, propagation time is in nanoseconds -- essentially negligible.
2. One of few alternatives to phone companies for long distances.
3. Uses broadcast technology over a wide area - everyone on earth could receive a message at the same time!
4. Easy to place unauthorized taps into signal.

Satellites have recently fallen out of favor relative to fiber. However, fiber has one big disadvantage: no one has it coming into their house or building, whereas anyone can place an antenna on a roof and lease a satellite channel.

1.3.3 Line coding:

Various techniques used for conversion of digital and analog data to digital signal, commonly referred to as **encoding** techniques and converts digital data to digital signal, known as **line coding.**

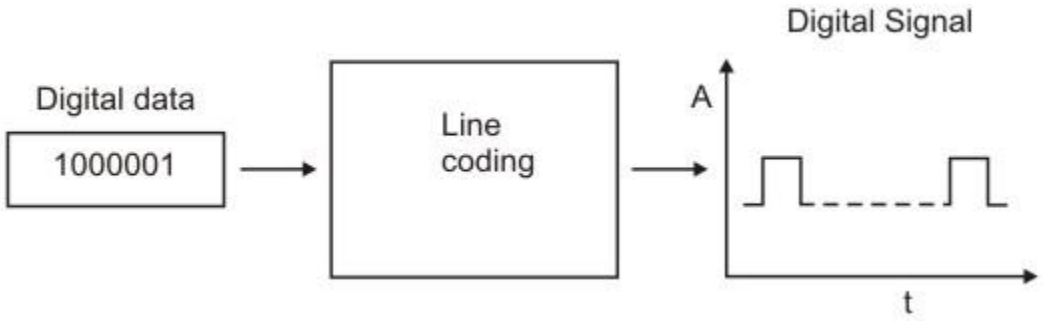
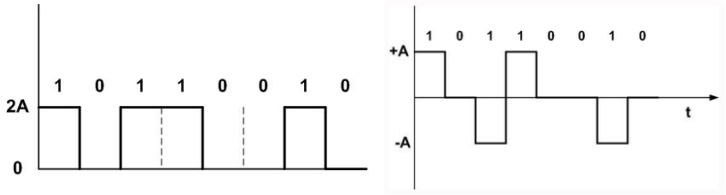


Figure 1.3.9 *Line coding to convert digital data to digital signal*

No of signal levels: This refers to the number values allowed in a signal, known as s**ignal levels**, to represent data. Figure 2.4.3(a) shows two signal levels, whereas Fig. 2.4.3(b) shows three signal levels to represent binary data.
Bit rate versus Baud rate: The **bit rate** represents the number of bits transmitted per second, whereas the **baud rate** defines the number of signal elements per second in the signal. Depending on the encoding technique used, baud rate may be more than or less than the data rate.
DC components: After line coding, the signal may have zero frequency component in the spectrum of the signal, which is known as the direct-current **(DC) component**. DC component in a signal is not desirable because the DC component does not pass through some components of a communication system such as a transformer. This leads to distortion of the signal and may create error at the output. The DC component also results in unwanted energy loss on the line.
Signal Spectrum: Different encoding of data leads to different spectrum of the signal. It is necessary to use suitable encoding technique to match with the medium so that the signal suffers minimum attenuation and distortion as it is transmitted through a medium.
Synchronization: To interpret the received signal correctly, the bit interval of the receiver should be exactly same or within certain limit of that of the transmitter. Any mismatch between the two may lead wrong interpretation of the received signal. Usually, clock is generated and synchronized from the received signal with the help of a special hardware known as Phase Lock Loop (PLL). However, this can be achieved if the received signal is self-synchronizing having frequent transitions (preferably, a minimum of one transition per bit interval) in the signal.
Cost of Implementation: It is desirable to keep the encoding technique simple enough such that it does not incur high cost of implementation.



(a) (b)

Figure 1.3.10 *(a) Signal with two voltage levels, (b) Signal with three voltage levels*

Line coding techniques can be broadly divided into three broad categories: Unipolar, Polar and Bipolar.

Unipolar: In unipolar encoding technique, only two voltage levels are used. It uses only one polarity of voltage level as shown in Fig. 1.3.11. In this encoding approach, the bit rate same as data rate. Unfortunately, DC component present in the encoded signal and there is loss of synchronization for long sequences of 0's and 1's. It is simple but obsolete.

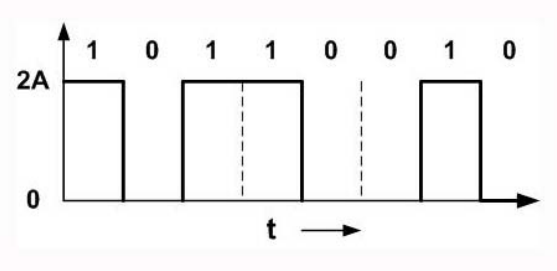


Figure 1.3.11 *Unipolar encoding with two voltage levels*

Polar: Polar encoding technique uses two voltage levels – one positive and the other one negative. Four different encoding schemes shown in Fig. 1.3.12 under this category discussed below:

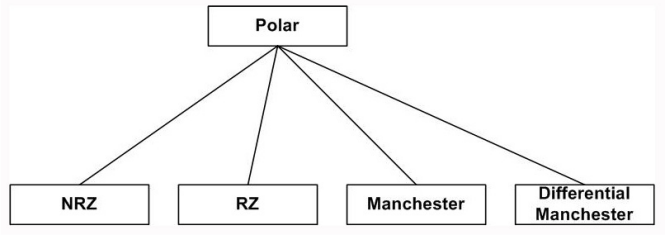
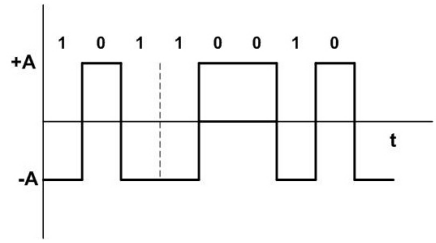


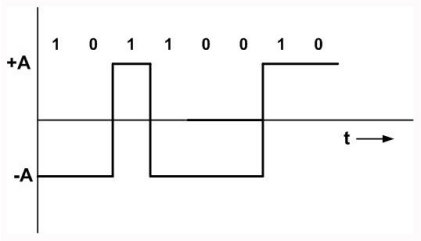
Figure 1.3.12 *Encoding Schemes under polar category*

Non Return to zero (NRZ): The most common and easiest way to transmit digital signals is to use two different voltage levels for the two binary digits. Usually a negative voltage is used to represent one binary value and a positive voltage to represent the other. The data is encoded as the presence or absence of a signal transition at the beginning of the bit time. As shown in the figure below, in NRZ encoding, the signal level remains same throughout the bit-period. There are two encoding schemes in NRZ: NRZ-L and NRZ-I, as shown in Fig. 1.3.13.



NRZ – L

1 = low level
0 = high level



NRZ – I

• For each 1 in the bit sequence, the signal level is inverted.
• A transition from one voltage level to the other represents a 1.

Figure 1.3.13 *NRZ encoding scheme*

The **advantages** of NRZ coding are:
Detecting a transition in presence of noise is more reliable than to compare a value to a threshold.
NRZ codes are easy to engineer and it makes efficient use of bandwidth.

Return to Zero RZ: To ensure synchronization, there must be a signal transition in each bit as shown in Fig. 1.3.14. Key characteristics of the RZ coding are:

• Three levels
• Bit rate is double than that of data rate
• No dc component
• Good synchronization
• Main limitation is the increase in bandwidth

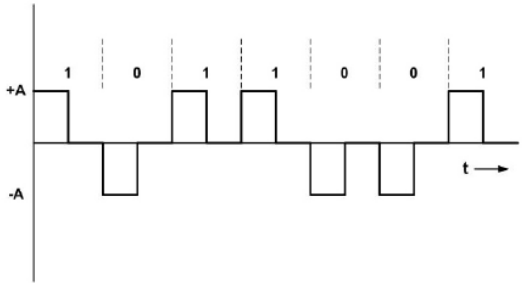


Figure 1.3.14 RZ encoding technique

Biphase: To overcome the limitations of NRZ encoding, biphase encoding techniques can be adopted. Manchester and differential Manchester Coding are the two common Biphase techniques in use, as shown in Fig. 1.3.15. In Manchester coding the mid-bit transition serves as a clocking mechanism and also as data.

In the standard Manchester coding there is a transition at the middle of each bit period. A binary 1 corresponds to a low-to-high transition and a binary 0 to a high-to-low transition in the middle.

In Differential Manchester, inversion in the middle of each bit is used for synchronization. The encoding of a 0 is represented by the presence of a transition both at the beginning and at the middle and 1 is represented by a transition only in the middle of the bit period.

Key characteristics are:

• Two levels
• No DC component
• Good synchronization
• Higher bandwidth due to doubling of bit rate with respect to data rate

A Manchester code is now very popular and has been specified for the IEEE 802.3 standard for base band coaxial cables and twisted pair CSMA/CD bus LANs.

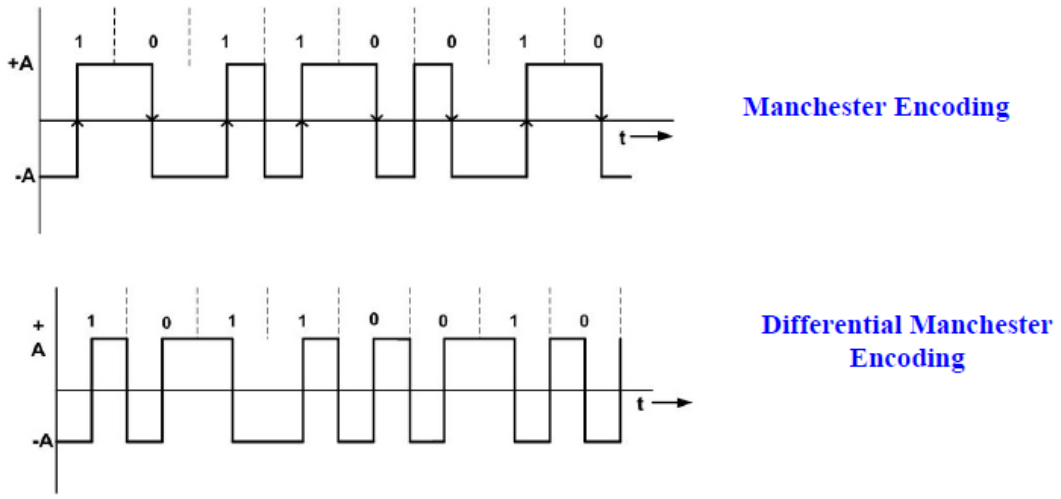


Figure 1.3.15 Manchester encoding schemes

Bipolar Encoding: Bipolar AMI uses three levels. Unlike RZ the zero level is used to represent a 0 and a binary 1's are represented by alternating positive and negative voltages, as shown in Fig 1.3.16.

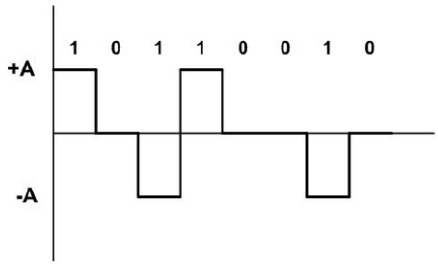


Figure1.3.16: Bipolar AMI Signal

1.4 Channel access on links:

In telecommunications and computer networks, a **channel access method** or **multiple access method** allows several terminals connected to the same multi-point transmission medium to transmit over it and to share its capacity. Examples of shared physical media are wireless networks, bus networks, ring networks, hub networks and half-duplex point-to-point links.

A channel-access scheme is based on a multiplexing method that allows several data streams or signals to share the same communication channel or physical medium. Multiplexing is in this context provided by the physical layer. But the multiplexing also may be used in full-duplex point-to-point communication between nodes in a switched network, which should not be considered as multiple access.

A channel-access scheme is also based on a multiple access protocol and control mechanism, also known as media access control (MAC). This protocol deals with issues such as addressing, assigning multiplex channels to different users, and avoiding collisions. The MAC-layer is a sub-layer in Layer 2 (Data Link Layer) of the OSI model and a component of the Link Layer of the TCP/IP model.

1.4.1 Fundamental forms of channel access schemes:

- The frequency division multiple access (FDMA) channel-access scheme is based on the frequency-division multiplex (FDM) scheme, which provides different frequency bands to different data-streams - in the FDMA case to different users or nodes. An example of FDMA systems were the first-generation *(*1G) cell-phone systems. A related technique is wave-length division multiple access (WDMA), based on wavelength division multiplex (WDM), where different users get different colors in fiber-optical communication.

- The time division multiple access (TDMA) channel access scheme is based on the time division multiplex (TDM) scheme, which provides different time-slots to different data-streams (in the TDMA case to different transmitters) in a cyclically repetitive frame structure. For example, user 1 may use time slot 1, user 2 time slot 2, etc. until the last user. Then it starts all over again.

- The code division multiple access (CDMA) scheme employs spread spectrum technology and a special coding scheme (where each transmitter is assigned a code) to allow multiple users to be multiplexed over the same physical channel at the same time. An example is the 3G cell phone system.

- Space division multiple access (SDMA) enables creating *parallel spatial pipes* next to *higher capacity pipes through spatial multiplexing and/or diversity*, by which it is able to offer superior performance in radio multiple access communication systems.

- Packet mode multiple-access is typically also based on time-domain multiplexing, but not in a cyclically repetitive frame structure, and therefore it is not considered as TDM or TDMA. Due to its random character it can be categorised as statistical multiplexing methods, making it possible to provide dynamic bandwidth allocation.

Duplexing methods

Where these methods are used for dividing forward and reverse communication channels, they are known as duplexing methods, such as:

- Time division duplex (TDD)

- Frequency division duplex (FDD)

1.4.2 Hybrid channel access techniques:

Hybrids of the above techniques can be - and frequently are - used. Some examples:

• The GSM cellular system combines the use of frequency division duplex (FDD) to prevent interference between outward and return signals, with FDMA and TDMA to allow multiple handsets to work in a single cell.

• GSM with the GPRS packet switched service combines FDD and FDMA with slotted Aloha for reservation inquiries, and a Dynamic TDMA scheme for transferring the actual data.

• Bluetooth packet mode communication combines frequency hopping (for shared channel access among several private area networks in the same room) with CSMA/CA (for shared channel access inside a medium).

• IEEE 802.11b wireless local area networks (WLANs) are based on FDMA and DS-CDMA for avoiding interference among adjacent WLAN cells or access points. This is combined with CSMA/CA for multiple access within the cell.

• HIPERLAN/2 wireless networks combine FDMA with dynamic TDMA, meaning that resource reservation is achieved by packet scheduling.

• G.hn, an ITU-T standard for high-speed networking over home wiring (power lines, phone lines and coaxial cables) employs a combination of TDMA, Token passing and CSMA/CARP to allow multiple devices to share the medium.

1.5. Issues in Data Link layer:

Link Configuration Control
• Link Discipline Control
• Link Management - bringing link up and down
• Framing
• Flow Control
• Error Control
Link Configuration Control refers to the following:
• Link Topology
• Link Duplexity

The topology of a communication link refers to the physical arrangement of the connection between the devices. In its fundamental form the topology of a data link between two devices could be:

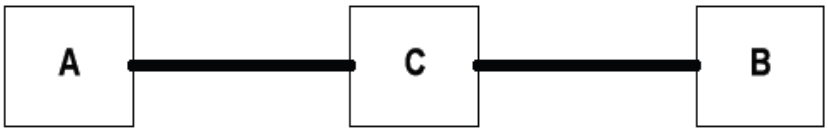
- Direct Link
- Indirect Link

DirectLink:

Two devices are connected by a direct link if there are no intermediate devices (except repeaters or amplifiers) in between them.

Indirect Link

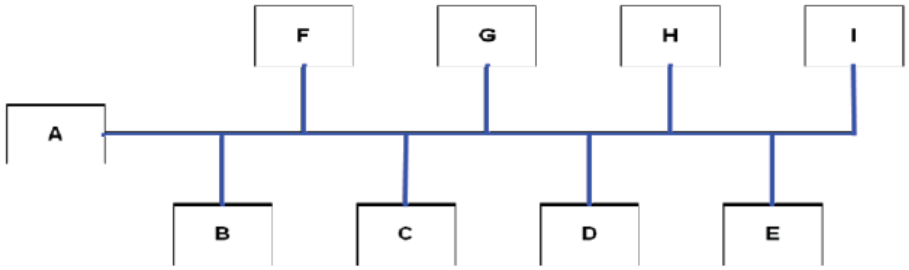
If there are one or more intermediate devices between two devices then the link between them is referred to as an indirect link. Devices A and C have a direct link between them whereas devices A and B have an indirect link between them



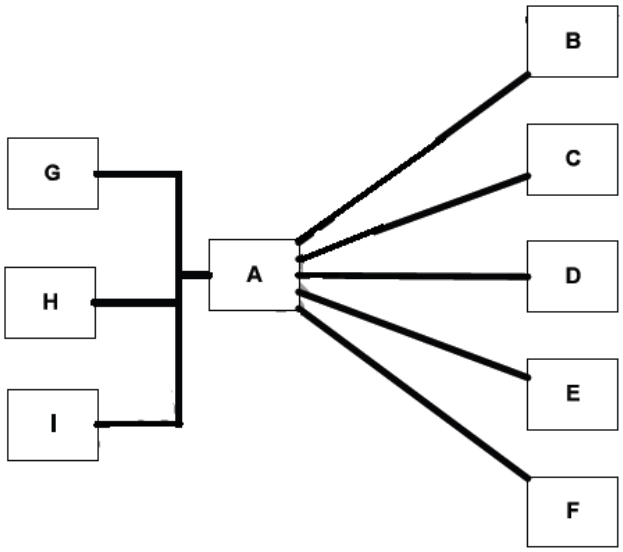
There are two possibilities with a direct link:

- Multipoint Link
- Point to Point Link

A direct link is called a multipoint link if there are more than two devices sharing the link.



A direct link between two devices is called a point to point link if and only if they are the only two devices sharing link.



Link Duplexity:
Duplexity refers to the fact that either one station can transmit at a time (half duplex) or both can transmit simultaneously (full duplex)

Simplex:

- Only one device can transmit to the other i.e. only transmit in one direction
- Not real communication, just one way communication, rarely used in data communications
- Examples: ordinary television, radio e.g., receiving signals from the radio station or CATV
- The sending station has only one transmitter the receiving station has only one receiver

Half Duplex:

- Both devices can transmit to each other but not simultaneously, data may travel in both directions, but only in one direction at a time
- Devices take turns to speak
- Usually implies single path for both transmission and reception
- Computers use control signals to negotiate when to send and when to receive
- The time it takes to switch between sending and receiving is called turnaround time

Full Duplex:

- Both devices can transmit simultaneously
- Usually implies separate transmit and receive paths
- Complete two-way simultaneous transmission
- Faster than half-duplex communication because no turnaround time is needed Link Discipline Control

Link discipline is dependent on three things:

- The topology of the link
- Duplexity of the link
- Relationship of the devices on the link i.e. Peer to Peer or Primary - Secondary
    1. Primary - Secondary is the old terminal to host environment
    2. Peer - Peer is the modern computer network environment

Link Discipline with Point to Point Links

- Simple
- One device may send an ENQ message to see if the other is ready
- On receiving an ACK the DATA frame may be sent Line Discipline with Multipoint Links

There are two possibilities:

- Designated Primary Station

    This Primary-secondary relationship is the case for terminal host setup

- No Designated Primary Station i.e. Peer - Peer

    Peer to Peer relationship holds in computer network.

Designated Primary Station

- The Primary station controls the link
- Primary either polls the secondaries or selects one of them to transmit

No Designated Primary Station

- Various Medium Access Control (MAC) schemes are in use in Local Area Networks
- Examples CSMA/CD, Token Ring.

1.5 Framing:

Normally, units of data transfer are larger than a single analog or digital encoding symbol. It is necessary to recover clock information for both the signal (so we can recover the right number of symbols and recover each symbol as accurately as possible), and obtain synchronization for larger units of data (such as data words and frames). It is necessary to recover the data in words or blocks because this is the only way the receiver process will be able to interpret the data received; for a given bit stream. Depending on the byte boundaries, there will be seven or eight ways to interpret the bit stream as ASCII characters, and these are likely to be very different. So, it is necessary to add other bits to the block that convey control information used in the data link control procedures. The data along with preamble, postamble, and control information forms a **frame**.

Synchronization

Data sent by a sender in bit-serial form through a medium must be correctly interpreted at the receiving end. This requires that the beginning, the end and logic level and duration of each bit as sent at the transmitting end must be recognized at the receiving end. There are three synchronization levels: *Bit, Character and Frame.* Moreover, to achieve synchronization, two approaches known as *asynchronous* and *synchronous* transmissions are used.

Frame synchronization is the process by which incoming frame alignment signals (i.e., distinctive bit sequences) are identified, i.e. distinguished from data bits, permitting the data bits within the frame to be extracted for decoding or retransmission. The usual practice is to insert, in a dedicated time slot within the frame, a non-information bit that is used for the actual synchronization of the incoming data with the receiver.
In order to receive bits in the first place, the receiver must be able to determine how fast bits are being sent and when it has received a signal symbol. Further, the receiver needs to be able to determine what the relationship of the bits in the received stream have to one another, that is, what the logical units of transfer are, and where each received bit fits into the logical units. We call these logical units *frames*. This means that in addition to bit (or transmission symbol) synchronization, the receiver needs word and frame synchronization.

Synchronous communication (bit-oriented)
Timing is recovered from the signal itself (by the carrier if the signal is analog, or by regular transitions in the data signal or by a separate clock line if the signal is digital). Scrambling is often used to ensure frequent transitions needed. The data transmitted may be of any bit length, but is often constrained by the frame transfer protocol (data link or MAC protocol).
Bit-oriented framing only assumes that bit synchronization has been achieved by the underlying hardware, and the incoming bit stream is scanned at all possible bit positions for special patterns generated by the sender. The sender uses a special pattern (a flag pattern) to delimit frames (one flag at each end), and has to provide for data transparency by use of bit stuffing (see below). A commonly used flag pattern is HDLC's 01111110 flag as shown in Fig. 1.5.1. The bit sequence 01111110 is used for both preamble and postamble for the purpose of synchronization. A frame format for bit-oriented synchronous frame is shown in Fig. 1.5.2. Apart from the flag bits there are control fields. This field contains the commands, responses and sequences numbers used to maintain the data flow accountability of the link, defines the functions of the frame and initiates the logic to control the movement of traffic between sending and receiving stations.

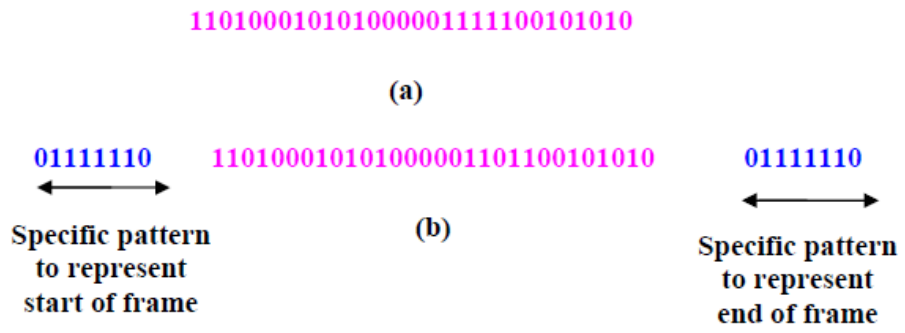


Figure 1.5.1 Bit oriented framing (a) Data to be sent to the peer, (b) Data after being character stuffed

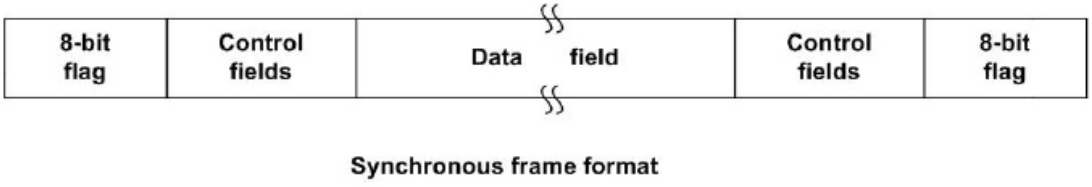


Figure 1.5.2 Frame format for synchronous communication

Summary of the approach:

• Initially 1 or 2 synchronization characters are sent
• Data characters are then continuously sent without any extra bits
• At the end, some error detection data is sent

Advantages:
• Much less overhead
• No overhead is incurred except for synchronization characters

Disadvantages:
• No tolerance in clock frequency is allowed
• The clock frequency should be same at both the sending and receiving ends

Bit stuffing: If the flag pattern appears anywhere in the header or data of a frame, then the receiver may prematurely detect the start or end of the received frame. To overcome this problem, the sender makes sure that the frame body it sends has no flags in it at any position (note that since there is no character synchronization, the flag pattern can start at any bit location within the stream). It does this by bit stuffing, inserting an extra bit in any pattern that is beginning to look like a flag. In HDLC, whenever 5 consecutive 1's are encountered in the data, a 0 is inserted after the 5th 1, regardless of the next bit in the data as shown in Fig. 1.5.3. On the receiving end, the bit stream is piped through a shift register as the receiver looks for the flag pattern. If 5 consecutive 1's followed by a 0 is seen, then the 0 is dropped before sending the data on (the receiver destuffs the stream). If 6 1's and a 0 are seen, it is a flag and either the current frame are ended or a new frame is started, depending on the current state of the receiver. If more than 6 consecutive 1's are seen, then the receiver has detected an invalid pattern, and usually the current frame, if any, is discarded.

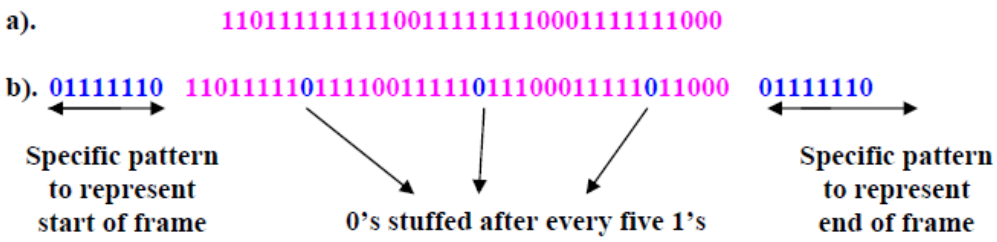


Figure 1.5.3 Bit oriented (a) Data to be sent to the peer, (b) Data after being bit stuffed.

With bit stuffing, the boundary between two frames can be unambiguously recognized by the flag pattern. Thus, if receiver loses track of where it is, all it has to do is to scan the input for flag sequence, since they can only occur at frame boundaries and never within data. In addition to receiving the data in logical units called frames, the receiver should have some way of determining if the data has been corrupted or not. If it has been corrupted, it is desirable not only to realize that, but also to make an attempt to obtain the correct data. This process is called error detection and error correction.

Asynchronous communication (word-oriented):

In asynchronous communication, small, fixed-length words (usually 5 to 9 bits long) are transferred without any clock line or clock is recovered from the signal itself. Each word has a start bit (usually as a 0) before the first data bit of the word and a stop bit (usually as a 1) after the last data bit of the word, as shown in Fig. 1.5.4. The receiver's local clock is started when the receiver detects the 1-0 transition of the start bit, and the line is sampled in the middle of the fixed bit intervals (a bit interval is the inverse of the data rate). The sender outputs the bit at the agreed-upon rate, holding the line in the appropriate state for one bit interval for each bit, but using its own local clock to determine the length of these bit intervals. The receiver's clock and the sender's clock may not run at the same speed, so that there is a relative clock drift (this may be caused by variations in the crystals used, temperature, voltage, etc.). If the receiver's clock drifts too much relative to the sender's clock, then the bits may be sampled while the line is in transition from one state to another, causing the receiver to misinterpret the received data. There can be variable amount of gap between two frames as shown in Fig. 1.5.5.

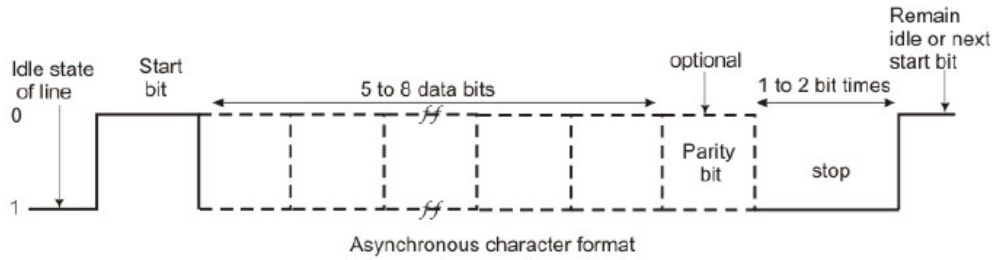


Figure 1.5.4 Character or word oriented format for asynchronous mode

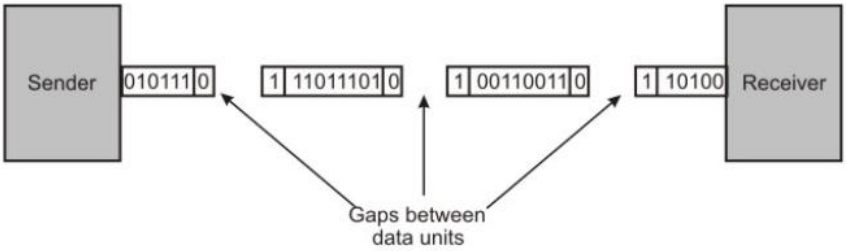


Figure 1.5.5 Character or word oriented format for asynchronous mode

Advantages of asynchronous character oriented mode of communication are summarized below:

- Simple to implement
- Self synchronization; Clock signal need not be sent
- Tolerance in clock frequency is possible
- The bits are sensed in the middle hence ± ½ bit tolerance is provided

This mode of data communication, however, suffers from high overhead incurred in data transmission. Data must be sent in multiples of the data length of the word, and the two or more bits of synchronization overhead compared to the relatively short data length causes the effective data rate to be rather low. For example, 11 bits are required to transmit 8 bits of data. In other words, baud rate (number of signal elements) is higher than data rate.

Character Oriented Framing:

The first framing method uses a field in the header to specify the number of characters in the frame. When the data link-layer sees the character count, it knows how many characters follow, and hence where the end of the frame is. This technique is shown in Fig. 1.5.6 for frames of size 6, 4, and 8 characters, respectively. The trouble with this algorithm is that the count can be garbled by a transmission error. For example, if the character count of 4 in the second frame becomes 5, as shown in Fig. 1.5.6(b), the destination will get out of synchronization and will be unable to locate the start of next frame. Even if the checksum is incorrect so the destination knows that the frame is bad, it still had no way of telling where the next frame starts. Sending a frame back to the source and asking for retransmission does not help either, since the destination doesn't know how many characters to skip over to the start of retransmission. For this reason the character count method is rarely used.

Character-oriented framing assumes that character synchronization has already been achieved by the hardware. The sender uses special characters to indicate the start and end of frames, and may also use them to indicate header boundaries and to assist the receiver gain character synchronization. Frames must be of an integral character length. Data transparency must be preserved by use of character as shown in Fig. 1.5.7.

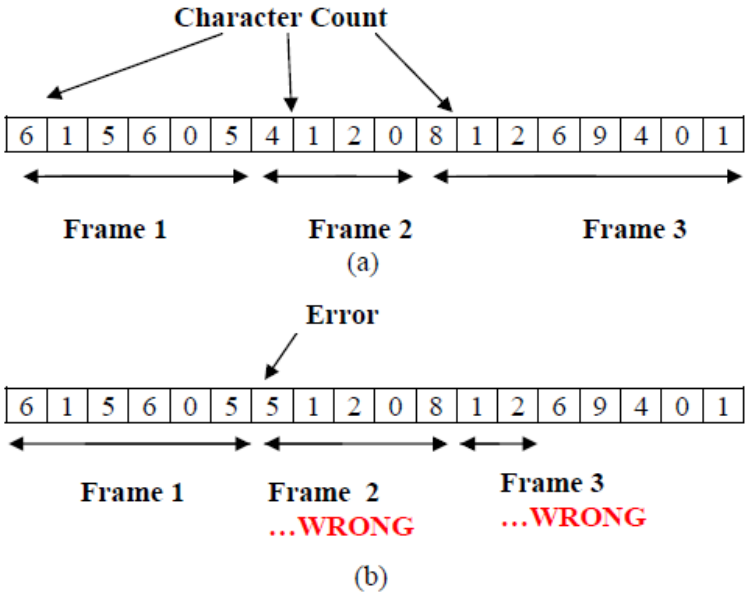


Figure 1.5.6 A Character Stream (a) Without error and (b) with error

Most commonly, a DLE (data link escape) character is used to signal that the next character is a control character, with DLE SOH (start of header) used to indicate the start of the frame (it starts with a header), DLE STX (start of text) used to indicate the end of the header and start of the data portion, and DLE ETX (end of text) used to indicate the end of the frame.

Figure 1.5.7 Character Oriented (a) Data to be send to the peer, (b) Data after being character stuffed

A serious problem occurs with this method when binary data, such as object program are being transmitted. It may easily happen when the characters for DLE STX or DLE ETX occur in the data, which will interfere with the framing. One way to overcome this problem is to use character stuffing discussed below.

Character stuffing:
When a DLE character occurs in the header or the data portion of a frame, the sender must somehow let the receiver know that it is not intended to signal a control character. The sender does this by inserting an extra DLE character after the one occurring inside the frame, so that when the receiver encounters two DLEs in a row, it immediately deletes one and interpret the other as header or data. This is shown in Fig. 1.5.8. Note that since the receiver has character synchronization, it will not mistake a DLE pattern that crosses a byte boundary as a DLE signal.

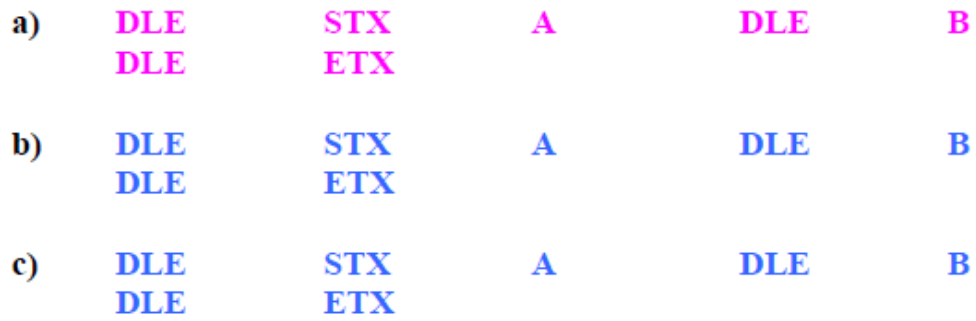


Figure 1.5.8 Character Stuffing (a). Data send by network layer, (b) Data after being character stuffed by the data link layer. (c) Data passed to the network layer on the receiver side.

The main disadvantage of this method is that it is closely tied to 8-bit characters in general and the ASCII character code in particular. As networks grow, this disadvantage of embedding the character code in framing mechanism becomes more and more obvious, so a new technique had to be developed to allow arbitrary sized character. Bit-oriented frame synchronization and bit stuffing is used that allow data frames to contain an arbitrary number of bits and allow character code with arbitrary number of bits per character.

1.6 Error Correction and Detection:

Environmental interference and physical defects in the communication medium can cause random bit errors during data transmission. Error coding is a method of detecting and correcting these errors to ensure information is transferred intact from its source to its destination. Error coding is used for fault tolerant computing in computer memory, magnetic and optical data storage media, satellite and deep space communications, network communications, cellular telephone networks, and almost any other form of digital data communication. Error coding uses mathematical formulas to encode data bits at the source into longer bit words for transmission. The "code word" can then be decoded at the destination to retrieve the information. The extra bits in the code word provide *redundancy* that, according to the coding scheme used, will allow the destination to use the decoding process to determine if the communication medium introduced errors and in some cases correct them so that the data need not be retransmitted. Different error coding schemes are chosen depending on the types of errors expected, the communication medium's expected error rate, and whether or not data retransmission is possible. Faster processors and better communications technology make more complex coding schemes, with better error detecting and correcting capabilities, possible for smaller embedded systems, allowing for more robust communications. However, tradeoffs between bandwidth and coding overhead, coding complexity and allowable coding delay between transmissions, must be considered for each application.

There are two basic strategies for dealing with errors. One way is to include enough redundant information (extra bits are introduced into the data stream at the transmitter on a regular and logical basis) along with each block of data sent to enable the receiver to deduce what the transmitted character must have been. The other way is to include only enough redundancy to allow the receiver to deduce that error has occurred, but not which error has occurred and the receiver asks for a retransmission. The former strategy uses Error-Correcting Codes and latter uses Error-detecting Codes.

Error Detecting Codes:

Basic approach used for error detection is the use of redundancy, where additional bits are added to facilitate detection and correction of errors. Popular techniques are:

- Simple Parity check
- Two-dimensional Parity check
- Checksum
- Cyclic redundancy check

Simple Parity Checking or One-dimension Parity Check :

- The most common and least expensive mechanism for error- detection is the simple parity check. In this technique, a redundant bit called **parity bit**, is appended to every data unit so that the number of 1s in the unit (including the parity becomes even).
- Blocks of data from the source are subjected to a check bit or *Parity bit* generator form, where a parity of 1 is added to the block if it contains an odd number of 1's (ON bits) and 0 is added if it contains an even number of 1's. At the receiving end the parity bit is computed from the received data bits and compared with the received parity bit, as shown in Fig. 1.6.1. This scheme makes the total number of 1's even, that is why it is called *even parity checking*.

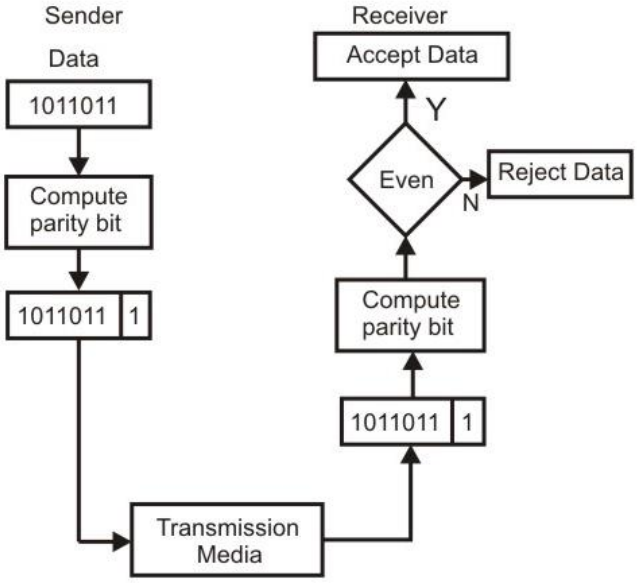


Figure 1.6.1 Even-parity checking scheme

Two-dimension Parity Check:

Performance can be improved by using two-dimensional parity check, which organizes the block of bits in the form of a table. Parity check bits are calculated for each row, which is equivalent to a simple parity check bit. Parity check bits are also calculated for all columns then both are sent along with the data. At the receiving end these are compared with the parity bits calculated on the received data. This is illustrated in Fig. 1.6.2.

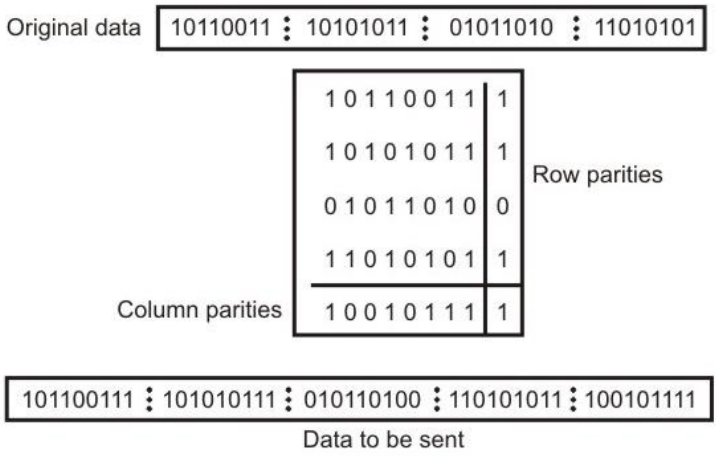


Figure 1.6.2: Two-dimension Parity Checking

Performance

Two- Dimension Parity Checking increases the likelihood of detecting burst errors. As we have shown in Fig. 1.6.2 that a 2-D Parity check of n bits can detect a burst error of n bits. A burst error of more than n bits is also detected by 2-D Parity check with a high-probability. There is, however, one pattern of error that remains elusive. If two bits in one data unit are damaged and two bits in exactly same position in another data unit are also damaged, the 2-D Parity check checker will not detect an error. For example, if two data units: 11001100 and 10101100. If first and second from last bits in each of them is changed, making the data units as 01001110 and 00101110, the error cannot be detected by 2-D Parity check.

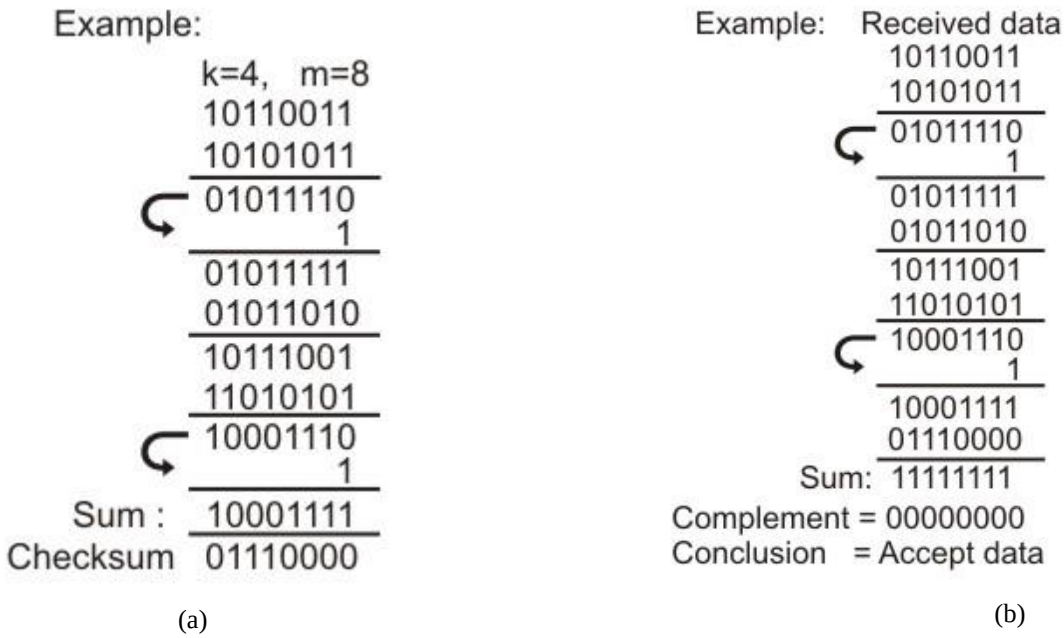


Figure 1.6.3 (a) Sender's end for the calculation of the checksum, (b) Receiving end for checking the checksum

Checksum:

In checksum error detection scheme, the data is divided into k segments each of m bits. In the sender's end the segments are added using 1's complement arithmetic to get the sum. The sum is complemented to get the checksum. The checksum segment is sent along with the data segments as shown in Fig. 1.6.3 (a). At the receiver's end, all received segments are added using 1's complement arithmetic to get the sum. The sum is complemented. If the result is zero, the received data is accepted; otherwise discarded, as shown in Fig. 1.6.3 (b).

Performance

The checksum detects all errors involving an odd number of bits. It also detects most errors involving even number of bits.

Cyclic Redundancy Checks (CRC):

This Cyclic Redundancy Check is the most powerful and easy to implement technique. Unlike checksum scheme, which is based on addition, CRC is based on binary division. In CRC, a sequence of redundant bits, called **cyclic redundancy check bits**, are appended to the end of data unit so that the resulting data unit becomes exactly divisible by a second, predetermined binary number. At the destination, the incoming data unit is divided by the same number. If at this step there is no remainder, the data unit is assumed to be correct and is therefore accepted. A remainder indicates that the data unit has been damaged in transit and therefore must be rejected. The generalized technique can be explained as follows.

If a *k* bit message is to be transmitted, the transmitter generates an *r*-bit sequence, known as *Frame Check Sequence* (FCS) so that the *(k+r)* bits are actually being transmitted. Now this *r*-bit FCS is generated by dividing the original number, appended by *r* zeros, by *a* predetermined number. This number, which is *(r+1)* bit in length, can also be considered as the coefficients of a polynomial, called G*enerator Polynomial.* The remainder of this division process generates the *r*-bit FCS. On receiving the packet, the receiver divides the *(k+r)* bit frame by the same predetermined number and if it produces no remainder, it can be assumed that no error has occurred during the transmission. Operations at both the sender and receiver end are shown in Fig. 1.6.4.

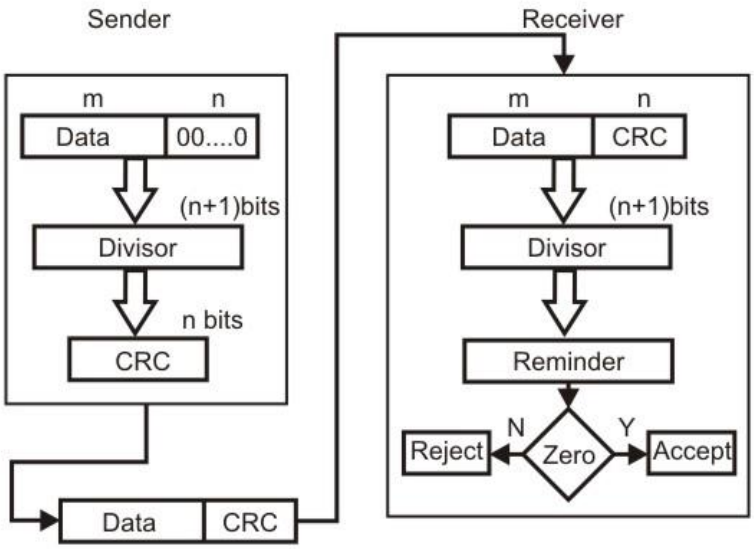


Figure 1.6.4 Basic scheme for Cyclic Redundancy Checking

This mathematical operation performed is illustrated in Fig. 1.6.5 by dividing a sample 4-bit number by the coefficient of the generator polynomial x3+x+1, which is 1011, using the modulo-2 arithmetic. Modulo-2 arithmetic is a binary addition process without any carry over, which is just the Exclusive-OR operation. Consider the case where k=1101. Hence we have to divide 1101000 (i.e. *k* appended by 3 zeros) by 1011, which produces the remainder r=001, so that the bit frame *(k+r)* =1101001 is actually being transmitted through the communication channel. At the receiving end, if the received number, i.e., 1101001 is divided by the same generator polynomial 1011 to get the remainder as 000, it can be assumed that the data is free of errors.

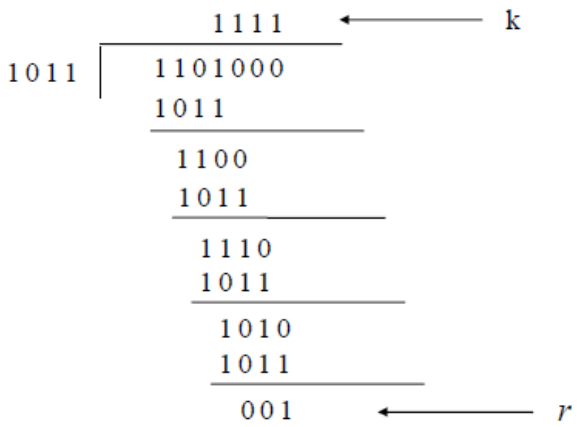


Figure1.6.5 Cyclic Redundancy Checks (CRC)

The transmitter can generate the CRC by using a feedback shift register circuit. The same circuit can also be used at the receiving end to check whether any error has occurred. All the values can be expressed as polynomials of a dummy variable X. For example, for P = 11001 the corresponding polynomial is X4+X3+1. A polynomial is selected to have at least the following properties:

- It should not be divisible by X.
- It should not be divisible by (X+1).

The first condition guarantees that all burst errors of a length equal to the degree of polynomial are detected. The second condition guarantees that all burst errors affecting an odd number of bits are detected.

CRC process can be expressed as XnM(X)/P(X) = Q(X) + R(X) / P(X)

Commonly used divisor polynomials are:

CRC-16 = X16 + X15 + X2 + 1

CRC-CCITT = X16 + X12 + X5 + 1

CRC-32 = X32 + X26 + X23 + X22 + X16 + X12 + X11 + X10 + X8 + X7 + X5 + X4 + X2 + 1

Performance

CRC is a very effective error detection technique. If the divisor is chosen according to the previously mentioned rules, its performance can be summarized as follows:

CRC can detect all single-bit errors

CRC can detect all double-bit errors (three 1's)

CRC can detect any odd number of errors (X+1)

CRC can detect all burst errors of less than the degree of the polynomial.

CRC detects most of the larger burst errors with a high probability.

• For example CRC-12 detects 99.97% of errors with a length 12 or more.

Error Correcting Codes:

The techniques that we have discussed so far can detect errors, but do not correct them. **Error Correction** can be handled in two ways.

o One is when an error is discovered; the receiver can have the sender retransmit the entire data unit. This is known as **backward error correction**.

o In the other, receiver can use an error-correcting code, which automatically corrects certain errors. This is known as **forward error correction**.

In theory it is possible to correct any number of errors atomically. Error-correcting codes are more sophisticated than error detecting codes and require more redundant bits. The number of bits required to correct multiple-bit or burst error is so high that in most of the cases it is inefficient to do so. For this reason, most error correction is limited to one, two or at the most three-bit errors.

1.7 Link-level Flow Control:

For reliable and efficient data communication a great deal of coordination is necessary between at least two machines. Some of these are necessary because of the following constraints:

Both sender and receiver have limited speed

Both sender and receiver have limited memory

It is necessary to satisfy the following requirements:

- o A fast sender should not overwhelm a slow receiver, which must perform a certain amount of processing before passing the data on to the higher-level software.
- o If error occur during transmission, it is necessary to devise mechanism to correct it

The most important functions of Data Link layer to satisfy the above requirements are **error control** and **flow control**. Collectively, these functions are known as **data link control.**

Flow Control is a technique so that transmitter and receiver with different speed characteristics can communicate with each other. Flow control ensures that a transmitting station, such as a server with higher processing capability, does not overwhelm a receiving station, such as a desktop system, with lesser processing capability. This is where there is an orderly flow of transmitted data between the source and the destination.

Error Control involves both error detection and error correction. It is necessary because errors are inevitable in data communication, in spite of the use of better equipment and reliable transmission media based on the current technology. When an error is detected, the receiver can have the specified frame retransmitted by the sender. This process is commonly known as **Automatic Repeat Request (ARQ).**

Flow Control

Modern data networks are designed to support a diverse range of hosts and communication mediums. Consider a 933 MHz Pentium-based host transmitting data to a 90 MHz 80486/SX. Obviously, the Pentium will be able to drown the slower processor with data. Likewise, consider two hosts, each using an Ethernet LAN, but with the two Ethernets connected by a 56 Kbps modem link. If one host begins transmitting to the other at Ethernet speeds, the modem link will quickly become overwhelmed. In both cases, *flow control* is needed to pace the data transfer at an acceptable speed.

Flow Control is a set of procedures that tells the sender how much data it can transmit before it must wait for an acknowledgment from the receiver. The flow of data should not be allowed to overwhelm the receiver. Receiver should also be able to inform the transmitter before its limits (this limit may be amount of memory used to store the incoming data or the processing power at the receiver end) are reached and the sender must send fewer frames. Hence, **Flow control** refers to the set of procedures used to restrict the amount of data the transmitter can send before waiting for acknowledgment.

There are two methods developed for flow control namely **Stop-and-wait** and **Sliding-window**. Stop-and-wait is also known as Request/reply sometimes. Request/reply (Stop-and-wait) flow control requires each data packet to be acknowledged by the remote host before the next packet is sent. **Sliding window** algorithms, used by TCP, permit multiple data packets to be in simultaneous transit, making more efficient use of network bandwidth.

Stop-and-Wait:

This is the simplest form of flow control where a sender transmits a data frame. After receiving the frame, the receiver indicates its willingness to accept another frame by sending back an ACK frame acknowledging the frame just received. The sender must wait until it receives the ACK frame before sending the next data frame. This is sometimes referred to as *ping-pong* behavior, request/reply is simple to understand and easy to implement, but not very efficient. In LAN environment with fast links, this isn't much of a concern, but WAN links will spend most of their time idle, especially if several hops are required.
Figure 1.7.1 illustrates the operation of the stop-and-wait protocol. The downward arrows show the sequence of data frames being sent across the link from the sender (top to the receiver (bottom). The protocol relies on two-way transmission (full duplex or half duplex) to allow the receiver at the remote node to return frames acknowledging the successful transmission. The acknowledgements are shown in upward arrows in the diagram, and flow back to the original sender. A small processing delay may be introduced between reception of the last byte of a Data PDU and generation of the corresponding ACK.

Major drawback of Stop-and-Wait Flow Control is that only one frame can be in transmission at a time, this leads to inefficiency if propagation delay is much longer than the transmission delay.

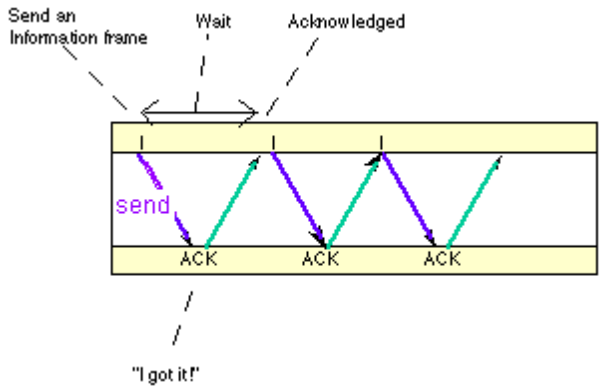


Figure 1. 7.1 Stop-and Wait protocol

Link Utilization in Stop-and-Wait
Let us assume the following:
Transmission time: The time it takes for a station to transmit a frame (normalized to a value of 1).
Propagation delay: The time it takes for a bit to travel from sender to receiver (expressed as *a*).
a < 1 :The frame is sufficiently long such that the first bits of the frame arrive at the destination before the source has completed transmission of the frame.
a > 1: Sender completes transmission of the entire frame before the leading bits of the frame arrive at the receiver.
The link utilization U = 1/(1+2a),

a = Propagation time / transmission time

It is evident from the above equation that the link utilization is strongly dependent on the ratio of the propagation time to the transmission time. When the propagation time is small, as in case of LAN environment, the link utilization is good. But, in case of long propagation delays, as in case of satellite communication, the utilization can be very poor. To improve the link utilization, we can use the following (sliding-window) protocol instead of using stop-and-wait protocol.

Sliding Window :

With the use of multiple frames for a single message, the stop-and-wait protocol does not perform well. Only one frame at a time can be in transit. In stop-and-wait flow control, if *a > 1*, serious inefficiencies result. Efficiency can be greatly improved by allowing multiple frames to be in transit at the same time. Efficiency can also be improved by making use of the full-duplex line. To keep track of the frames, sender station sends sequentially numbered frames. Since the sequence number to be used occupies a field in the frame, it should be of limited size. If the header of the frame allows k bits, the sequence numbers range from 0 to 2k – 1. Sender maintains a list of sequence numbers that it is allowed to send (sender window). The size of the sender's window is at most 2k – 1. The sender is provided with a buffer equal to the window size. Receiver also maintains a window of size 2k – 1. The receiver acknowledges a frame by sending an ACK frame that includes the sequence number of the next frame expected. This also explicitly announces that it is prepared to receive the next N frames, beginning with the number specified. This scheme can be used to acknowledge multiple frames. It could receive frames 2, 3, 4 but withhold ACK until frame 4 has arrived. By returning an ACK with sequence number 5, it acknowledges frames 2, 3, 4 in one go. The receiver needs a buffer of size 1.

Sliding window algorithm is a method of flow control for network data transfers. TCP, the Internet's stream transfer protocol, uses a sliding window algorithm.

A sliding window algorithm places a buffer between the application program and the network data flow. For TCP, the buffer is typically in the operating system kernel, but this is more of an implementation detail than a hard-and-fast requirement.

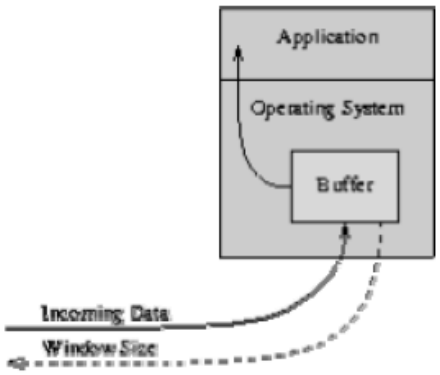


Figure 1.7.2 Buffer in sliding window

Data received from the network is stored in the buffer, from where the application can read at its own pace. As the application reads data, buffer space is freed up to accept more input from the network. The *window* is the amount of data that can be "read ahead" - the size of the buffer, less the amount of valid data stored in it. *Window announcements* are used to inform the remote host of the current *window size*.

Sender sliding Window: At any instant, the sender is permitted to send frames with sequence numbers in a certain range (the sending window) as shown in Fig. 1.7.3.

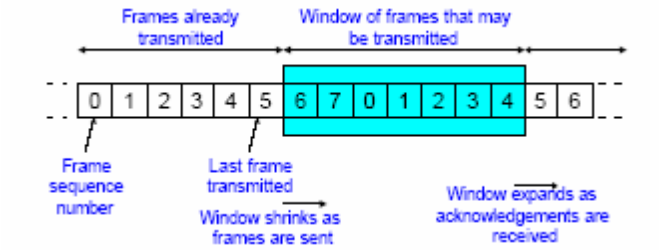


Figure 1.7.3 Sender's window

Receiver sliding Window: The receiver always maintains a window of size 1 as shown in Fig. 1.7.4. It looks for a specific frame (frame 4 as shown in the figure) to arrive in a specific order. If it receives any other frame (out of order), it is discarded and it needs to be resent. However, the receiver window also slides by one as the specific frame is received and accepted as shown in the figure. The receiver acknowledges a frame by sending an ACK frame that includes the sequence number of the next frame expected. This also explicitly announces that it is prepared to receive the next N frames, beginning with the number specified. This scheme can be used to acknowledge multiple frames. It could receive frames 2, 3, 4 but withhold ACK until frame 4 has arrived. By returning an ACK with sequence number 5, it acknowledges frames 2, 3, 4 at one time. The receiver needs a buffer of size 1.

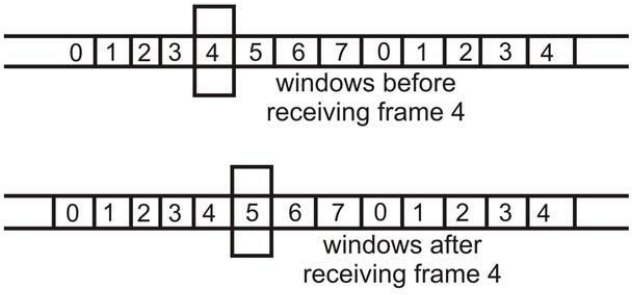


Figure 1.7.4 Receiver sliding window

On the other hand, if the local application can process data at the rate it's being transferred; sliding window still gives us an advantage. If the window size is larger than the packet size, then multiple packets can be outstanding in the network, since the sender knows that buffer space is available on the receiver to hold all of them. Ideally, a steady-state condition can be reached where a series of packets (in the forward direction) and window announcements (in the reverse direction) are constantly in transit. As each new window announcement is received by the sender, more data packets are transmitted. As the application reads data from the buffer (remember, we're assuming the application can keep up with the network), more window announcements are generated. Keeping a series of data packets in transit ensures the efficient use of network resources.

Hence, Sliding Window Flow Control

- Allows transmission of multiple frames
- Assigns each frame a k-bit sequence number
- Range of sequence number is [0…2k-1], i.e., frames are counted modulo 2k.

The link utilization in case of Sliding Window Protocol

U = 1, for N > 2a + 1

N/(1+2a), for N < 2a + 1

Where N = the window size, and a = Propagation time / transmission time

Error Control Techniques

When an error is detected in a message, the receiver sends a request to the transmitter to retransmit the ill-fated message or packet. The most popular retransmission scheme is known as Automatic-Repeat-Request (ARQ). Such schemes, where receiver asks transmitter to re-transmit if it detects an error, are known as reverse error correction techniques. There exist three popular ARQ techniques, as shown in Fig. 1.7.5.

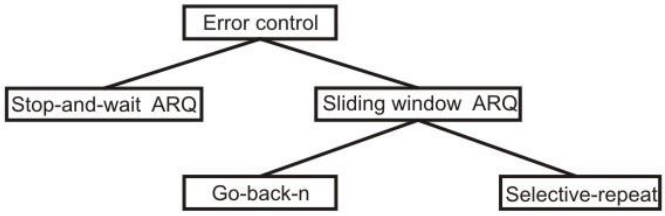


Figure 1.7.5 Error control techniques

Stop-and-Wait ARQ:

In Stop-and-Wait ARQ, which is simplest among all protocols, the sender (say station A) transmits a frame and then waits till it receives positive acknowledgement (ACK) or negative acknowledgement (NACK) from the receiver (say station B). Station B sends an ACK if the frame is received correctly, otherwise it sends NACK. Station A sends a new frame after receiving ACK; otherwise it retransmits the old frame, if it receives a NACK. This is illustrated in Fig 1.7.6.

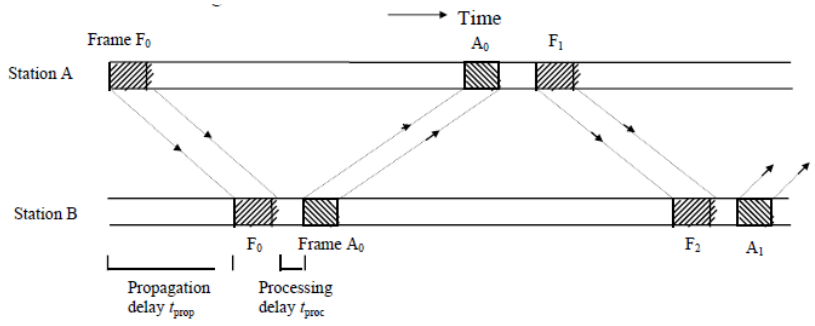


Figure 1.7.6 Stop-And-Wait ARQ technique

To tackle the problem of a lost or damaged frame, the sender is equipped with a timer. In case of a lost ACK, the sender transmits the old frame. In the Fig. 1.7.7, the second PDU of Data is lost during transmission. The sender is unaware of this loss, but starts a timer after sending each PDU. Normally an ACK PDU is received before the timer expires. In this case no ACK is received, and the timer counts down to zero and triggers retransmission of the same PDU by the sender. The sender always starts a timer following transmission, but in the second transmission receives an ACK PDU before the timer expires, finally indicating that the data has now been received by the remote node.

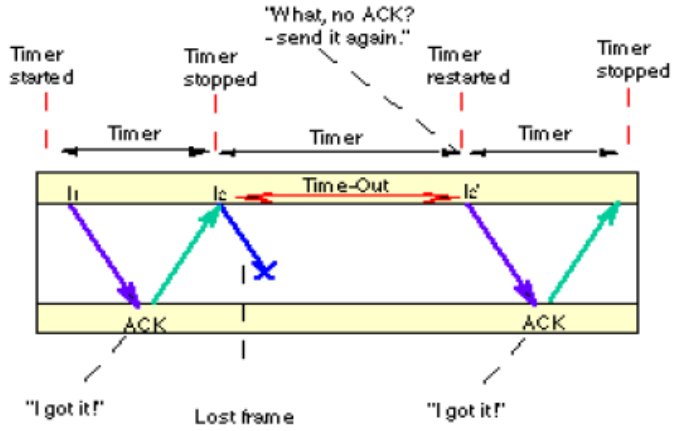


Figure 1.7.7 Retransmission due to lost frame

The receiver now can identify that it has received a duplicate frame from the label of the frame and it is discarded. To tackle the problem of damaged frames, say a frame that has been corrupted during the transmission due to noise, there is a concept of NACK frames, i.e. Negative Acknowledge frames. Receiver transmits a NACK frame to the sender if it founds the received frame to be corrupted. When a NACK is received by a transmitter before the time-out, the old frame is sent again as shown in Fig. 1.7.8.

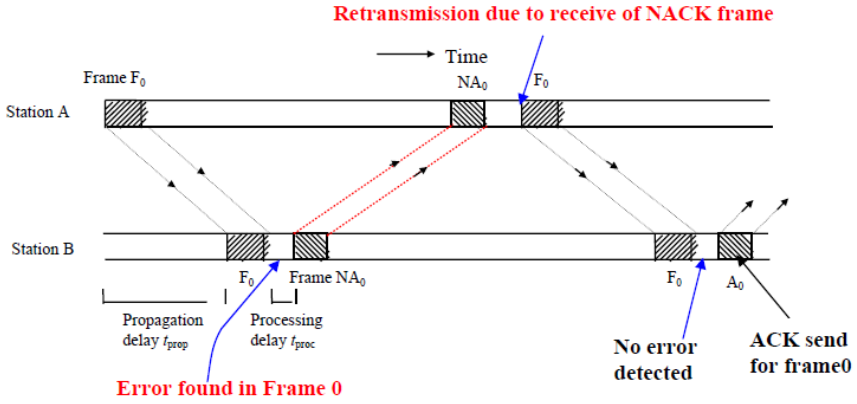


Figure1.7.8 Retransmission due to damaged frame

The main advantage of stop-and-wait ARQ is its simplicity. It also requires minimum buffer size. However, it makes highly inefficient use of communication links, particularly when 'a' is large.

Go-back-N ARQ:

The most popular ARQ protocol is the go-back-N ARQ, where the sender sends the frames continuously without waiting for acknowledgement. That is why it is also called as *continuous ARQ*. As the receiver receives the frames, it keeps on sending ACKs or a NACK, in case a frame is incorrectly received. When the sender receives a NACK, it retransmits the frame in error plus all the succeeding frames as shown in Fig.1.7.9. Hence, the name of the protocol is go-back-N ARQ. If a frame is lost, the receiver sends NAK after receiving the next frame as shown in Fig. 1.7.10. In case there is long delay before sending the NAK, the sender will resend the lost frame after its timer times out. If the ACK frame sent by the receiver is lost, the sender resends the frames after its timer times out as shown in Fig. 1.7.11.

Assuming full-duplex transmission, the receiving end sends piggybacked acknowledgement by using some number in the ACK field of its data frame. Let us assume that a 3-bit sequence number is used and suppose that a station sends frame 0 and gets back an RR1, and then sends frames 1, 2, 3, 4, 5, 6, 7, 0 and gets another RR1.This might either mean that RR1 is a cumulative ACK or all 8 frames were damaged. This ambiguity can be overcome if the maximum window size is limited to 7, i.e. for a k-bit sequence number field it is limited to 2k-1. The number N (=2k-1) specifies how many frames can be sent without receiving acknowledgement.

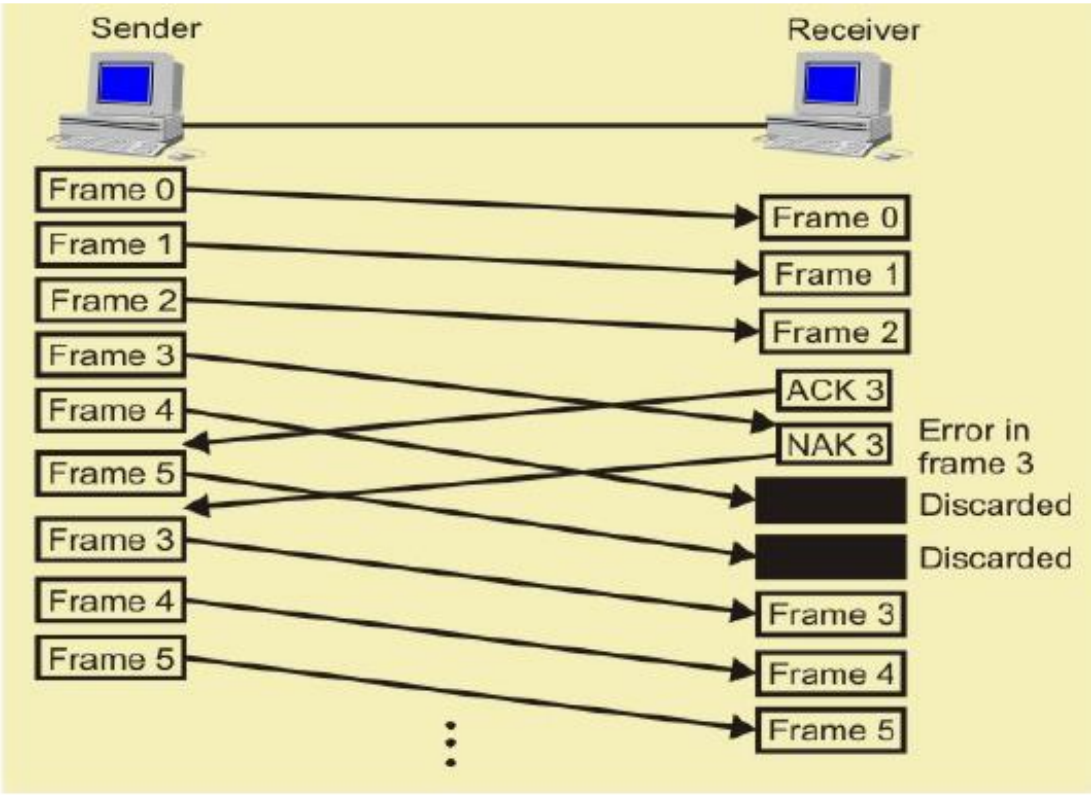


Figure 1.7.9 Frames in error in go-Back-N ARQ

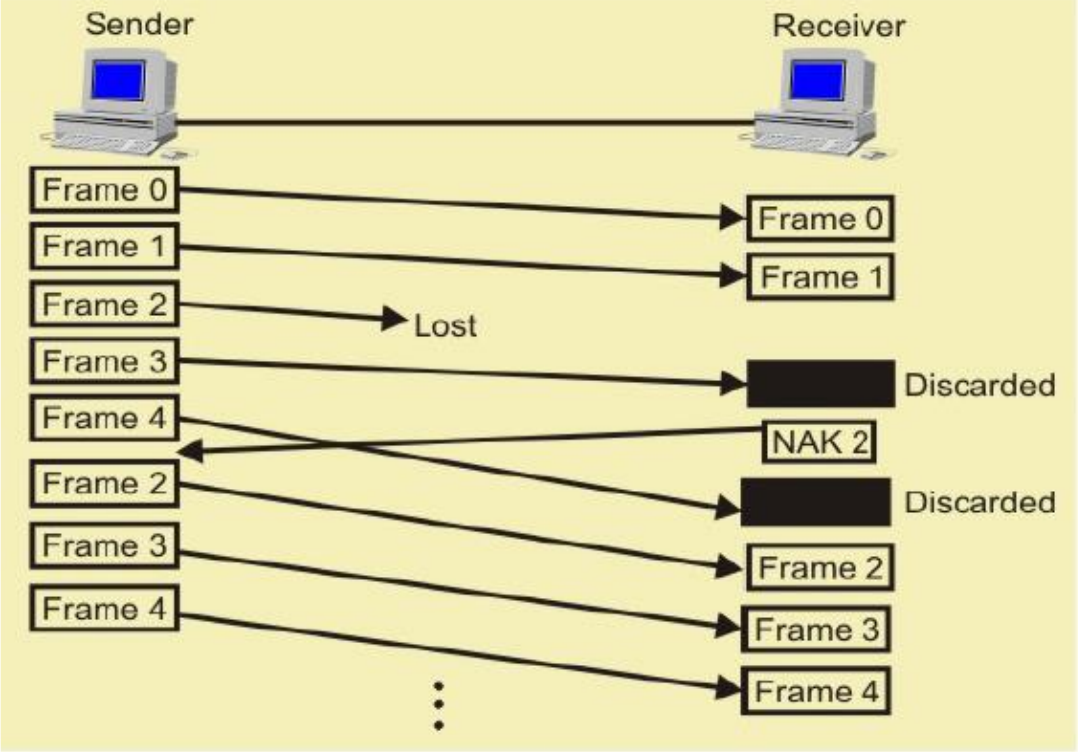


Figure 1.7.10 Lost Frames in Go-Back-N ARQ

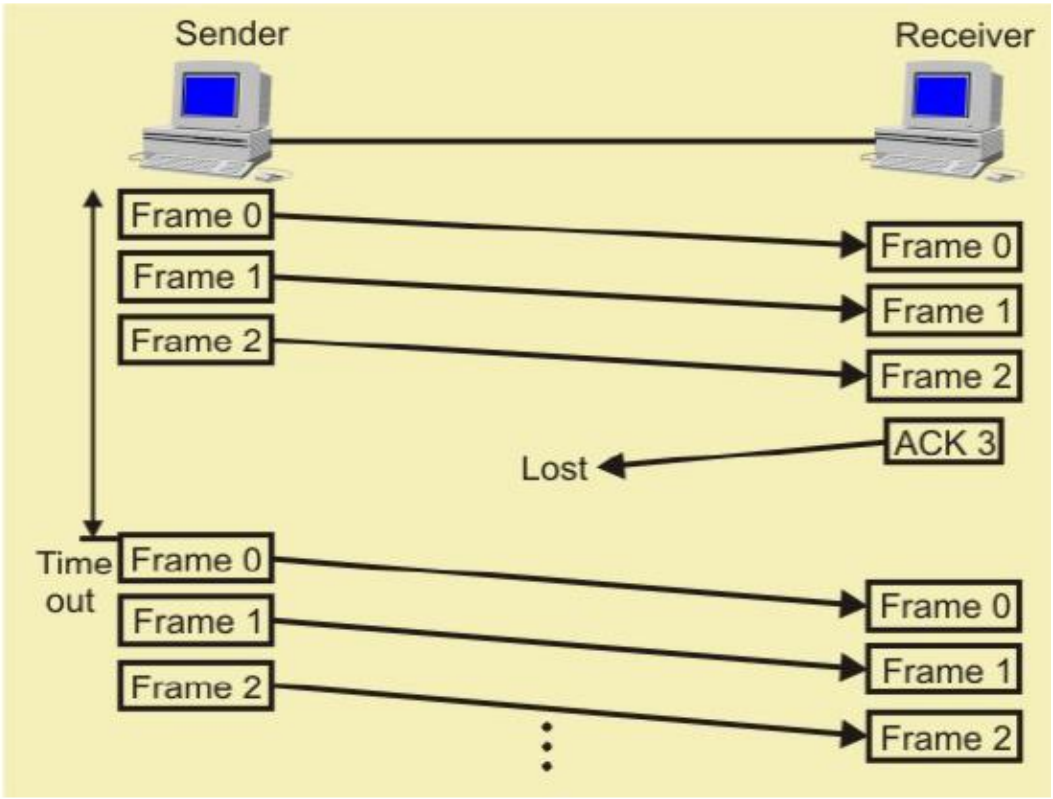


Figure 1.7.11 Lost ACK in Go-Back-N ARQ

If no acknowledgement is received after sending N frames, the sender takes the help of a timer. After the time-out, it resumes retransmission. The go-back-N protocol also takes care of damaged frames and damaged ACKs. This scheme is little more complex than the previous one but gives much higher throughput.

Assuming full-duplex transmission, the receiving end sends piggybacked acknowledgement by using some number in the ACK field of its data frame. Let us assume that a 3-bit sequence number is used and suppose that a station sends frame 0 and gets back an RR1, and then sends frames 1, 2, 3, 4, 5, 6, 7, 0 and gets another RR1.This might either mean that RR1 is a cumulative ACK or all 8 frames were damaged. This ambiguity can be overcome if the maximum window size is limited to 7, i.e. for a k-bit sequence number field it is limited to 2k-1. The number N (=2k-1) specifies how many frames can be sent without receiving acknowledgement. If no acknowledgement is received after sending N frames, the sender takes the help of a timer. After the time-out, it resumes retransmission. The go-back-N protocol also takes care of damaged frames and damaged ACKs. This scheme is little more complex than the previous one but gives much higher throughput.

Selective-Repeat ARQ:

The selective-repetitive ARQ scheme retransmits only those for which NAKs are received or for which timer has expired, this is shown in the Fig.1.7.12. This is the most efficient among the ARQ schemes, but the sender must be more complex so that it can send out-of-order frames. The receiver also must have storage space to store the post-NAK frames and processing power to reinsert frames in proper sequence.

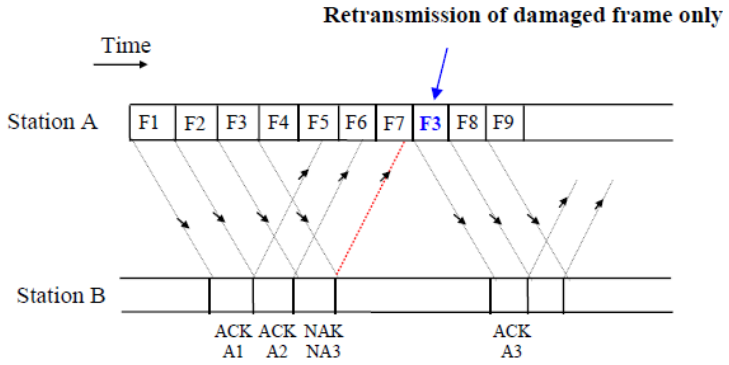


Figure 1.7.12 Selective-repeat Reject

HIGH LEVEL DATA LINK CONTROL:

The most important data link control protocol is HDLC (ISO 3009, ISO 4335). Not only is HDLC widely used, but it is the basis for many other important data link control protocols, which use the same or similar formats and the same mechanisms as employed in HDLC.

Basic Characteristics

To satisfy a variety of applications, HDLC defines three types of stations, two link configurations, and three data transfer modes of operation.

The three station types are:

• **Primary station:** Responsible for controlling the operation of the link. Frames issued by the primary are called commands.

• **Secondary station:** Operates under the control of the primary station. Frames issued by a secondary are called responses. The primary maintains a separate logical link with each secondary station on the line.

• **Combined station:** Combines the features of primary and secondary. A combined station may issue both commands and responses.

The two link configurations are:
• **Unbalanced configuration:** Consists of one primary and one or more secondary stations and supports both full-duplex and half-duplex transmission.
• **Balanced configuration:** Consists of two combined stations and supports both full-duplex and half-duplex transmission.

The three data transfer modes are:
• **Normal response mode (NRM):** Used with an unbalanced configuration. The primary may initiate data transfer to a secondary, but a secondary may only transmit data in response to a command from the primary.
• **Asynchronous balanced mode (ABM):** Used with a balanced configuration. Either combined station may initiate transmission without receiving permission from the other combined station.
• **Asynchronous response mode (ARM):** Used with an unbalanced configuration. The secondary may initiate transmission without explicit permission of the primary. The primary still retains responsibility for the line, including initialization, error recovery, and logical disconnection.

NRM is used on multidrop lines, in which a number of terminals are connected to a host computer. The computer polls each terminal for input. NRM is also sometimes used on point-to-point links, particularly if the link connects a terminal or other peripheral to a computer. ABM is the most widely used of the three modes; it makes more efficient use of a full-duplex point-to-point link because there is no polling overhead. ARM is rarely used; it is applicable to some special situations in which a secondary may need to initiate transmission.

Frame Structure:
HDLC uses synchronous transmission. All transmissions are in the form of frames, and a single frame format suffices for all types of data and control exchanges. Figure1.7.13 depicts the structure of the HDLC frame.
The flag, address, and control fields that precede the information field are known as a **header**. The FCS and flag fields following the data field are referred to as a **trailer**. **Flag Fields** Flag fields delimit the frame at both ends with the unique pattern 01111110.A single flag may be used as the closing flag for one frame and the opening flag for the next. On both sides of the user-network interface, receivers are continuously hunting for the flag sequence to synchronize on the start of a frame. While receiving a frame, a station continues to hunt for that sequence to determine the end of the frame. Because the protocol allows the presence of arbitrary bit patterns (i.e., there are no restrictions on the content of the various fields imposed by the link protocol), there is no assurance that the pattern 01111110 will not appear somewhere inside the frame, thus destroying synchronization. To avoid this problem, a procedure known as *bit stuffing* is used. For all bits between the starting and ending flags, the transmitter inserts an extra 0 bit after each occurrence of five 1s in the frame. After detecting a starting flag, the receiver monitors the bit stream. When a pattern of five 1s appears, the sixth bit is examined. If this bit is 0, it is deleted. If the sixth bit is a 1and the seventh bit is a 0, the combination is accepted as a flag. If the sixth and seventh bits are both 1, the sender is indicating an abort condition. With the use of bit stuffing, arbitrary bit patterns can be inserted into the data field of the frame. This property is known as **data transparency**. Note that in the first two cases, the extra 0 is not strictly necessary for avoiding a flag pattern but is necessary for the operation of the algorithm.

Address Field The address field identifies the secondary station that transmitted or is to receive the frame. This field is not needed for point-to-point links but is always included for the sake of uniformity. The address field is usually 8 bits long but, by prior agreement, an extended format may be used in which the actual address length is a multiple of 7 bits. The leftmost bit of each octet is 1 or 0 according as it is or is not the last octet of the address field. The remaining 7 bits of each octet form part of the address. The single-octet address of 11111111 is interpreted as the all-stations address in both basic and extended formats. It is used to allow the primary to broadcast a frame for reception by all secondaries.

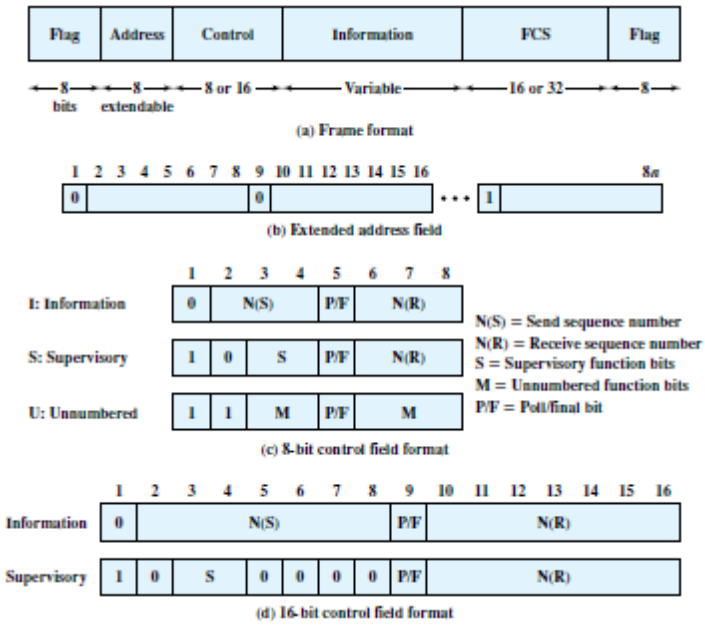


Figure 1.7.13 The structure of the HDLC frame.

Control Field HDLC defines three types of frames, each with a different control field format.

Information frames (I-frames) carry the data to be transmitted for the user (the logic above HDLC that is using HDLC). Additionally, flow and error control data, using the ARQ mechanism, are piggybacked on an information frame.

Supervisory frames (S-frames) provide the ARQ mechanism when piggybacking is not used. **Unnumbered frames** (U-frames) provide supplemental link control functions. The first one or two bits of the control field serves to identify the frame type. The remaining bit positions are organized into subfields as indicated in Figures 1.7.13c and d.

All of the control field formats contain the poll/final (P/F) bit. Its use depends on context. Typically, in command frames, it is referred to as the P bit and is set to 1 to solicit (poll) a response frame from the peer HDLC entity. In response frames, it is referred to as the F bit and is set to 1 to indicate the response frame transmitted as a result of a soliciting command. Note that the basic control field for S- and I-frames uses 3-bit sequence numbers. With the appropriate set-mode command, an extended control field can be used for S- and I-frames that employs 7-bit sequence numbers. U-frames always contain an 8-bit control field.

Information Field The information field is present only in I-frames and some U-frames. The field can contain any sequence of bits but must consist of an integral number of octets. The length of the information field is variable up to some system defined maximum.

Frame Check Sequence Field The frame check sequence (FCS) is an error detecting code calculated from the remaining bits of the frame, exclusive of flags. The normal code is the 16-bit CRC-CCITT. An optional 32-bit FCS, using CRC-32, may be employed if the frame length or the line reliability dictates this choice.

Operation

HDLC operation consists of the exchange of I-frames, S-frames, and U-frames between two stations. The various commands and responses defined for these frame types are listed in Table 1. 7.1. In describing HDLC operation, we will discuss these three types of frames.

Table 1.7.1 HDLC Commands and Responses

Name	Command/ Response	Description
Information (I)	C/R	Exchange user data
Supervisory (S)		
Receive ready (RR)	C/R	Positive acknowledgment; ready to receive I-frame
Receive not ready (RNR)	C/R	Positive acknowledgment; not ready to receive
Reject (REJ)	C/R	Negative acknowledgment; go back N
Selective reject (SREJ)	C/R	Negative acknowledgment; selective reject
Unnumbered (U)		
Set normal response/extended mode (SNRM/SNRME)	C	Set mode; extended = 7-bit sequence numbers
Set asynchronous response/extended mode (SARM/SARME)	C	Set mode; extended = 7-bit sequence numbers
Set asynchronous balanced/extended mode (SABM, SABME)	C	Set mode; extended = 7-bit sequence numbers
Set initialization mode (SIM)	C	Initialize link control functions in addressed station
Disconnect (DISC)	C	Terminate logical link connection
Unnumbered Acknowledgment (UA)	R	Acknowledge acceptance of one of the set-mode commands
Disconnected mode (DM)	R	Responder is in disconnected mode
Request disconnect (RD)	R	Request for DISC command
Request initialization mode (RIM)	R	Initialization needed; request for SIM command
Unnumbered information (UI)	C/R	Used to exchange control information
Unnumbered poll (UP)	C	Used to solicit control information
Reset (RSET)	C	Used for recovery; resets N(R), N(S)
Exchange identification (XID)	C/R	Used to request/report status
Test (TEST)	C/R	Exchange identical information fields for testing
Frame reject (FRMR)	R	Report receipt of unacceptable frame

The operation of HDLC involves three phases. First, one side or another initializes the data link so that frames may be exchanged in an orderly fashion. During this phase, the options that are to be used are agreed upon. After initialization, the two sides exchange user data and the control information to exercise flow and error control. Finally, one of the two sides signals the termination of the operation.

Initialization Either side may request initialization by issuing one of the six setmode commands. This command serves three purposes:

1. It signals the other side that initialization is requested.

2. It specifies which of the three modes (NRM,ABM,ARM) is requested.

3. It specifies whether 3- or 7-bit sequence numbers are to be used.

If the other side accepts this request, then the HDLC module on that end transmits an unnumbered acknowledged (UA) frame back to the initiating side. If the request is rejected, then a disconnected mode (DM) frame is sent.

Data Transfer When the initialization has been requested and accepted, then a logical connection is established. Both sides may begin to send user data in I frames, starting with sequence number 0.The N(S) and N(R) fields of the I-frame are sequence numbers that support flow control and error control. An HDLC module sending a sequence of I-frames will number them sequentially, modulo 8 or 128, depending on whether 3- or 7-bit sequence numbers are used, and place the sequence number in N(S). N(R) is the acknowledgment for I-frames received; it enables the HDLC module to indicate which number I-frame it expects to receive next.

S-frames are also used for flow control and error control. The receive ready (RR) frame acknowledges the last I-frame received by indicating the next I-frame expected. The RR is used when there is no reverse user data traffic (I-frames) to carry an acknowledgment. Receive not ready (RNR) acknowledges an I-frame, as with RR, but also asks the peer entity to suspend transmission of I-frames. When the entity that issued RNR is again ready, it sends an RR. REJ initiates the go-back-N ARQ. It indicates that the last I-frame received has been rejected and that retransmission of all I-frames beginning with number N(R) is required. Selective reject(SREJ) is used to request retransmission of just a single frame.

Disconnect Either HDLC module can initiate a disconnect, either on its own initiative if there is some sort of fault, or at the request of its higher-layer user. HDLC issues a disconnect by sending a disconnect (DISC) frame. The remote entity must accept the disconnect by replying with a UA and informing its layer 3 user that the connection has been terminated. Any outstanding unacknowledged I-frames may be lost, and their recovery is the responsibility of higher layers.

Unit-II

2.1 Introduction:

A network of computers based on multi-access medium requires a protocol for effective sharing of the media. As only one node can send or transmit signal at a time using the broadcast mode, the main problem here is how different nodes get control of the medium to send data, that is *"who goes next?".* The protocols used for this purpose are known as *Medium Access Control (MAC) techniques*. The key issues involved here are - *Where* and *How* the control is exercised.

'Where' refers to whether the control is exercised in a *centralised* or *distributed* manner. In a centralised system a master node grants access of the medium to other nodes. A centralized scheme has a number of advantages as mentioned below:

- Greater control to provide features like priority, overrides, and guaranteed bandwidth.
- Simpler logic at each node.
- Easy coordination.

Although this approach is easier to implement, it is vulnerable to the failure of the master node and reduces efficiency. On the other hand, in a distributed approach all the nodes collectively perform a medium access control function and dynamically decide which node to be granted access. This approach is more reliable than the former one.

'How' refers to in what manner the control is exercised. It is constrained by the topology and trade off between cost-performance and complexity. Various approaches for medium access control are shown in Fig. 2.1. The MAC techniques can be broadly divided into four categories; *Contention-based*, *Round-Robin*, *Reservation-based* and. *Channelization-based*. Under these four broad categories there are specific techniques, as shown in Fig. 2.1. The first two categories are used in the legacy LANs of the IEEE standard. The CSMA/CA, a collision-free protocol used in wireless LAN. Channelization-based MACs are used in cellular telephone networks and the reservation-based MACs, which are used in satellite networks.

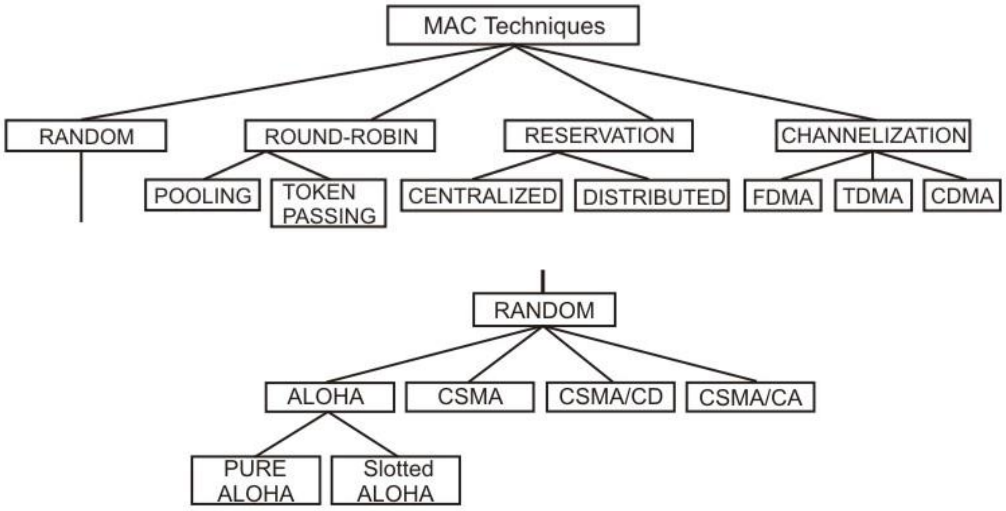


Figure 2.1 Possible MAC techniques

2.2 Goals of MACs:

Medium Access Control techniques are designed with the following goals in mind.

Initialisation: The technique enables network stations, upon power-up, to enter the state required for operation.

Fairness: The technique should treat each station fairly in terms of the time it is made to wait until it gains entry to the network, access time and the time it is allowed to spend for transmission.

Priority: In managing access and communications time, the technique should be able to give priority to some stations over other stations to facilitate different type of services needed.

Limitations to one station: The techniques should allow transmission by one station at a time. **Error Limitation**: The method should be capable of encompassing an appropriate error detection scheme.

Receipt: The technique should ensure that message packets are actually received (no lost packets) and delivered only once (no duplicate packets), and are received in the proper order. **Recovery**: If two packets collide (are present on the network at the same time), or if notice of a collision appears, the method should be able to recover, i.e. be able to halt all the transmissions and select one station to retransmit.

Reconfigurability: The technique should enable a network to accommodate the addition or deletion of a station with no more than a noise transient from which the network station can recover.

Compatibility: The technique should accommodate equipment from all vendors who build to its specification.

Reliability: The technique should enable a network to confine operating inspite of a failure of one or several stations.

2.3 Round Robin Techniques:

In Round Robin techniques, each and every node is given the chance to send or transmit by rotation. When a node gets its turn to send, it may either decline to send, if it has no data or may send if it has got data to send. After getting the opportunity to send, it must relinquish its turn after some maximum period of time. The right to send then passes to the next node based on a predetermined logical sequence. The right to send may be controlled in a centralised or distributed manner. *Polling* is an example of centralised control and *token passing* is an example of distributed control as discussed below.

2.3.1 Polling

The mechanism of polling is similar to the roll-call performed in a classroom. Just like the teacher, a controller sends a message to each node in turn. The message contains the address of the node being selected for granting access. Although all nodes receive the message, only the addressed node responds and then it sends data, if any. If there is no data, usually a *"poll reject"* message is sent

back. In this way, each node is interrogated in a round-robin fashion, one after the other, for granting access to the medium. The first node is again polled when the controller finishes with the remaining codes.

The polling scheme has the flexibility of either giving equal access to all the nodes, or some nodes may be given higher priority than others. In other words, priority of access can be easily implemented. Polling can be done using a central controller, which may use a frequency band to send outbound messages as shown in Fig. 5.2.2.

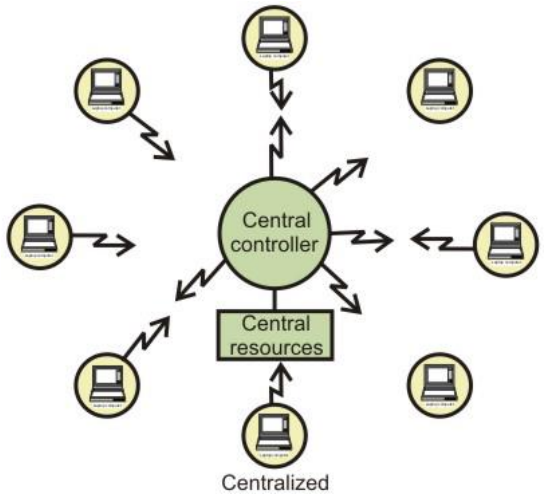


Figure 2.2 Polling using a central controller

Other stations share a different frequency to send inbound messages. The technique is called frequency-division duplex approach (FDD). Main drawbacks of the polling scheme are high overhead of the polling messages and high dependence on the reliability of the controller.

Polling can also be accomplished without a central controller. Here, all stations receive signals from other stations as shown in Fig. 2.3. Stations develop a polling order list, using some protocol.

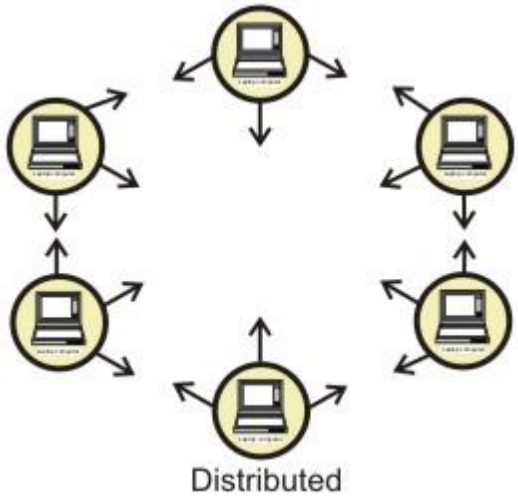


Figure 2.3 Polling in a distributed manner

2.3.2 Token Passing

In token passing scheme, all stations are logically connected in the form of a ring and control of the access to the medium is performed using a *token*. A *token* is a special bit pattern or a small packet, usually several bits in length, which circulate from node to node. Token passing can be used with both broadcast (token bus) and sequentially connected (token ring) type of networks with some variation in the details as considered in the next lesson.

In case of token ring, token is passed from a node to the physically adjacent node. On the other hand, in the token bus, token is passed with the help of the address of the nodes, which form a logical ring. In either case a node currently holding the token has the 'right to transmit'. When it has got data to send, it removes the token and transmits the data and then forwards the token to the next logical or physical node in the ring. If a node currently holding the token has no data to send, it simply forwards the token to the next node. The token passing scheme is efficient compared to the polling technique, but it relies on the correct and reliable operation of all the nodes. There exists a number of potential problems, such as *lost token*, *duplicate token*, *and insertion of a node*, *removal of a node*, which must be tackled for correct and reliable operation of this scheme.

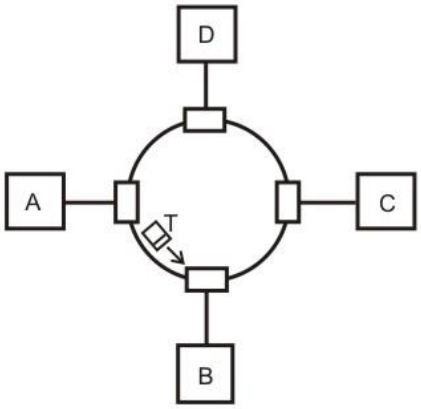


Figure 2.4 a. A token ring network

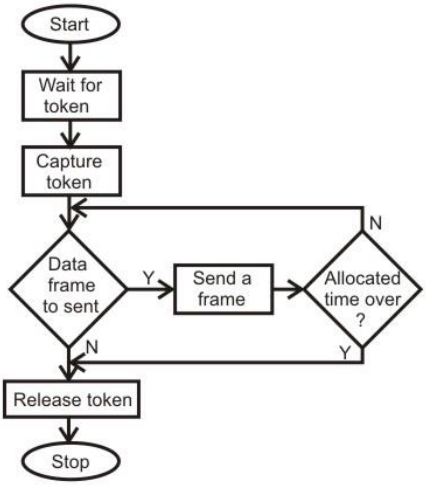


Figure.2.4 b. Token passing mechanism

Performance: Performance of a token ring network can be represented by two parameters; *throughput*, which is a measure of the successful traffic, and *delay*, which is a measure of time between when a packet is ready and when it is delivered. A station starts sending a packet at t = t0, completes transmission at t = t0 + a, receives the tail at t0 + 1 + a. So, the average time (delay) required to send a token to the next station = a/N. and throughput, $S = 1/(1 + a/N)$ for $a<1$ and $S = 1/a(1 + 1/N)$ for $a>1$.

2.4 Contention-based Approaches

Round-Robin techniques work efficiently when majority of the stations have data to send most of the time. But, in situations where only a few nodes have data to send for brief periods of time, Round-Robin techniques are unsuitable. Contention techniques are suitable for bursty nature of traffic. In contention techniques, there is no centralised control and when a node has data to send, it contends for gaining control of the medium. The principle advantage of contention techniques is their simplicity. They can be easily implemented in each node. The techniques work efficiently under light to moderate load, but performance rapidly falls under heavy load.

2.4.1 ALOHA

The ALOHA scheme was invented by Abramson in 1970 for a packet radio network connecting remote stations to a central computer and various data terminals at the campus of the university of Hawaii. A simplified situation is shown in Fig. 2.5. Users are allowed random access of the central computer through a common radio frequency band f_1 and the computer centre broadcasts all received signals on a different frequency band f_2. This enables the users to monitor packet collisions, if any. The protocol followed by the users is simplest; whenever a node has a packet to sent, it simply does so. The scheme, known as *Pure ALOHA*, is truly a *free-for-all* scheme. Of course, frames will suffer collision and colliding frames will be destroyed. By monitoring the signal sent by the central computer, after the maximum round-trip propagation time, an user comes to know whether the packet sent by him has suffered a collision or not.

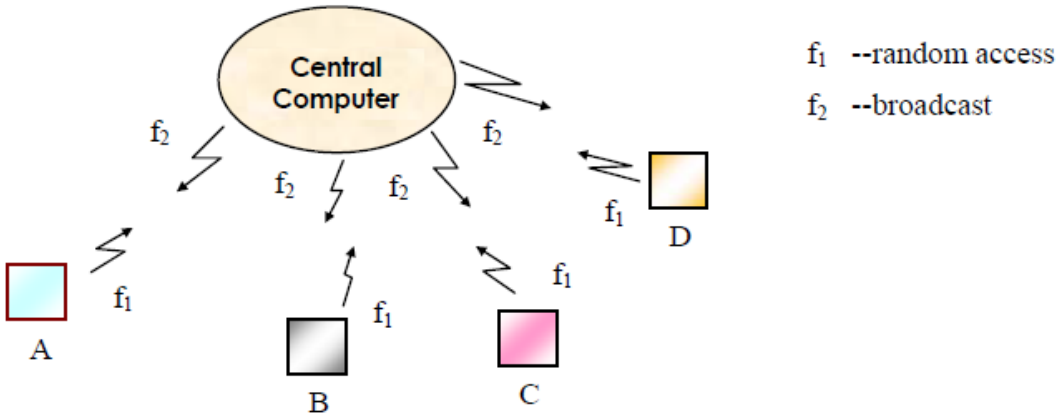


Figure 2.5 Simplified ALOHA scheme for a packet radio system

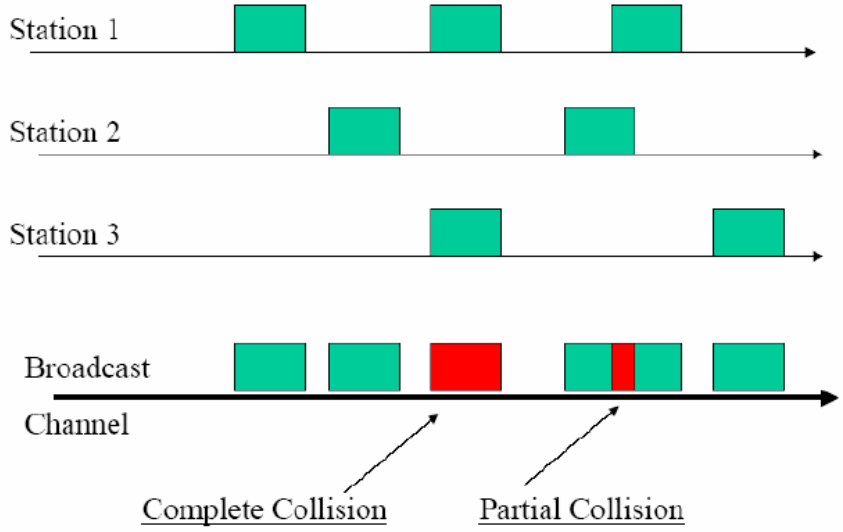


Figure 2.6 Collision in Pure ALOHA

It may be noted that if all packets have a fixed duration of τ (shown as F in Figure 2.7), then a given packet A will suffer collision if another user starts to transmit at any time from τ before to until τ after the start of the packet A as shown in Fig. 2.6. This gives a vulnerable period of 2τ. Based on this assumption, the channel utilization can be computed. The channel utilization, expressed as throughput S, in terms of the offered load G is given by $S=Ge^{-2G}$.

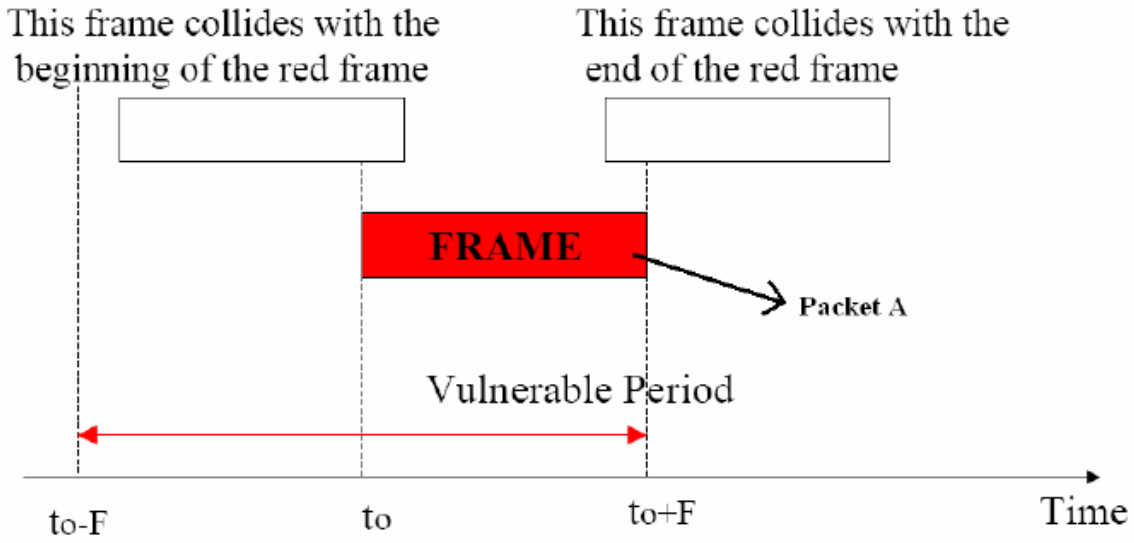


Figure 2.7 Vulnerable period in Pure ALOHA

Based on this, the best channel utilisation of 18% can be obtained at 50 percent of the offered load as shown in Fig. 2.8. At smaller offered load, channel capacity is underused and at higher offered load too many collisions occur reducing the throughput. The result is not encouraging, but for such a simple scheme high throughput was also not expected. Subsequently, in a new scheme, known as *Slotted ALOHA*, was suggested to improve upon the efficiency of pure ALOHA. In this scheme, the channel is divided into slots equal to τ and packet transmission can start only at the beginning of a slot as shown in Fig. 5.2.9. This reduces the vulnerable period from 2τ to τ and improves efficiency by reducing the probability of collision as shown in Fig. 2.10. This gives a maximum throughput of 37% at 100 percent of offered load, as shown in Figure 2.8.

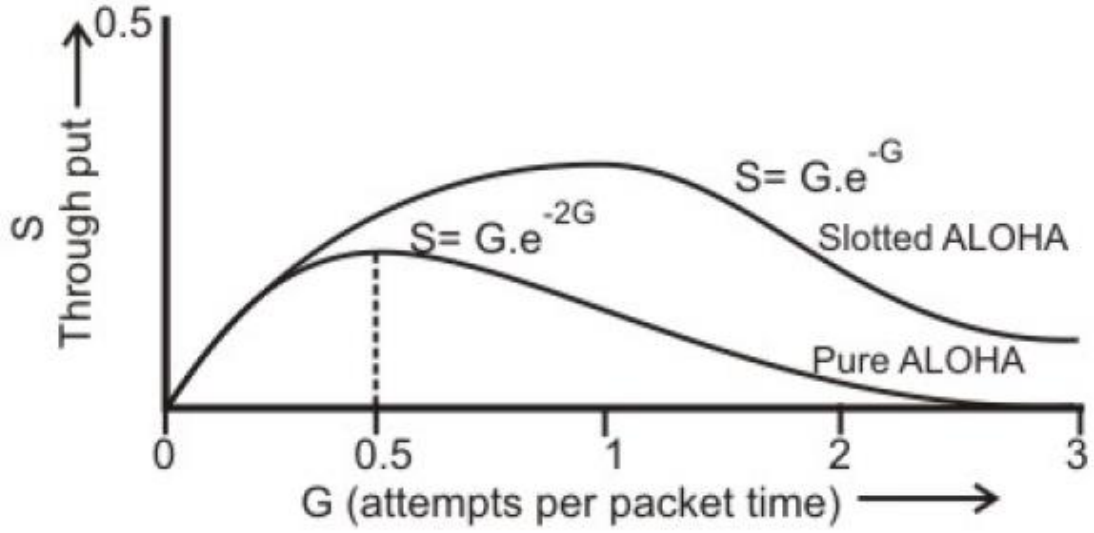


Figure 2.8 Throughput versus offered load for ALOHA protocol

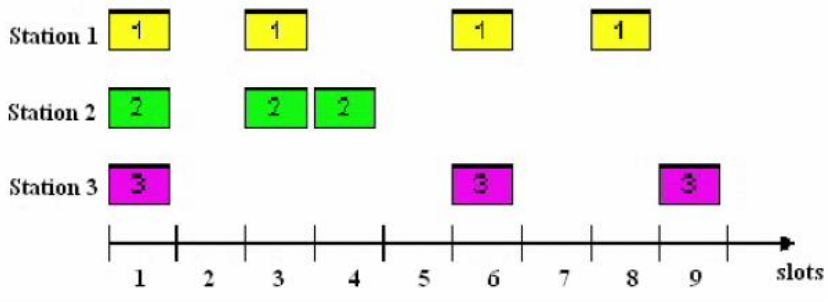


Figure **2**.9 Slotted ALOHA: Single active node can continuously transmit at full rate of channel

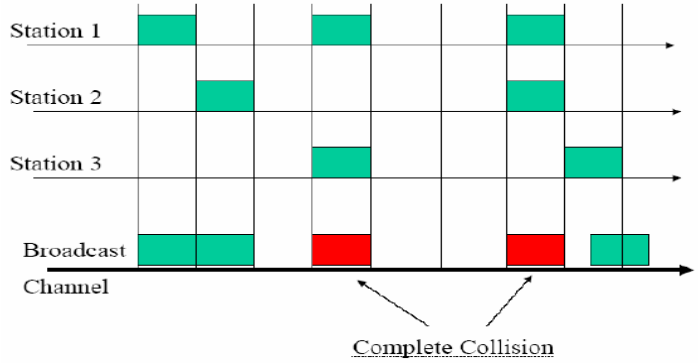


Figure 2.10 Collision in Slotted ALOHA

2.5 CSMA

The poor efficiency of the ALOHA scheme can be attributed to the fact that a node start transmission without paying any attention to what others are doing. In situations where propagation delay of the signal between two nodes is small compared to the transmission time of a packet, all other nodes will know very quickly when a node starts transmission. This observation is the basis of the *carrier-sense multiple-access* (CSMA) protocol. In this scheme, a node having data to transmit first listens to the medium to check whether another transmission is in progress or not. The node starts sending only when the channel is free, that is there is no carrier. That is why the scheme is also known as *listen-before-talk*. There are three variations of this basic scheme as outlined below.

(i) 1-persistent CSMA: In this case, a node having data to send, start sending, if the channel is sensed free. If the medium is busy, the node continues to monitor until the channel is idle. Then it starts sending data.

(ii) Non-persistent CSMA: If the channel is sensed free, the node starts sending the packet. Otherwise, the node waits for a random amount of time and then monitors the channel.

(iii) p-persistent CSMA: If the channel is free, a node starts sending the packet. Otherwise the node continues to monitor until the channel is free and then it sends with probability *p*.

The efficiency of CSMA scheme depends on the propagation delay, which is represented by a parameter a, as defined below:

a=Propagation delay/Packet transmission time.

The throughput of 1-persistent CSMA scheme is shown in Fig. 2.11 for different values of a. It may be noted that smaller the value of propagation delay, lower is the vulnerable period and higher is the efficiency.

2.6 CSMA/CD

CSMA/CD protocol can be considered as a refinement over the CSMA scheme. It has evolved to overcome one glaring inefficiency of CSMA. In CSMA scheme, when two packets collide the channel remains unutilized for the entire duration of transmission time of both the packets. If the propagation time is small (which is usually the case) compared to the packet transmission time, wasted channel capacity can be considerable. This wastage of channel capacity can be reduced if the nodes continue to monitor the channel while transmitting a packet and immediately cease transmission when collision is detected. This refined scheme is known as *Carrier Sensed Multiple Access with Collision Detection* (CSMA/CD) or *Listen-While-Talk.*

On top of the CSMA, the following rules are added to convert it into CSMA/CD:

(i) If a collision is detected during transmission of a packet, the node immediately ceases transmission and it transmits jamming signal for a brief duration to ensure that all stations know that collision has occurred.

(ii) After transmitting the jamming signal, the node waits for a random amount of time and then transmission is resumed.

The random delay ensures that the nodes, which were involved in the collision are not likely to have a collision at the time of retransmissions. To achieve stability in the back off scheme, a technique known as *binary exponential back off* is used. A node will attempt to transmit repeatedly in the face of repeated collisions, but after each collision, the mean value of the random delay is doubled. After 15 retries (excluding the original try), the unlucky packet is discarded and the node reports an error. A flowchart representing the binary exponential back off algorithm is given in Fig. 2.11.

Performance Comparisons: The throughput of the three contention based schemes with respect to the offered load is given in Fig 2.12. The figure shows that pure ALHOA gives a maximum throughput of only 18 percent and is suitable only for very low offered load. The slotted ALHOA gives a modest improvement over pure ALHOA with a maximum throughput of 36 percent. Non persistent CSMA gives a better throughput than 1-persistent CSMA because of smaller probability of collision for the retransmitted packets. The non-persistent CSMA/CD provides a high throughput and can tolerate a very heavy offered load. Figure 2.13 provides a plot of the offered load versus throughput for the value of a = 0.01.

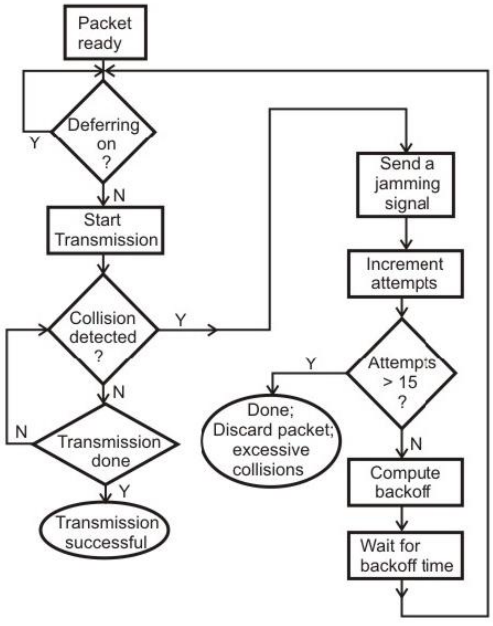


Figure 2.11 Binary exponential back off algorithm used in CSMA/CD

Protocol	Throughput
ALOHA	$S = Ge^{-2G}$
Slotted ALOHA	$S = Ge^{-G}$
Nonpersistent CSMA	$S = \frac{Ge^{-aG}}{[G(1+2a)+e^{-aG}]}$
Nonpersistent CSMA/CD	$S = \frac{Ge^{-aG}}{[Ge^{-aG}+3aG(1-e^{-aG})+(2-e^{-aG})]}$

Figure 2.12 Comparison of the throughputs for the contention-based MACs

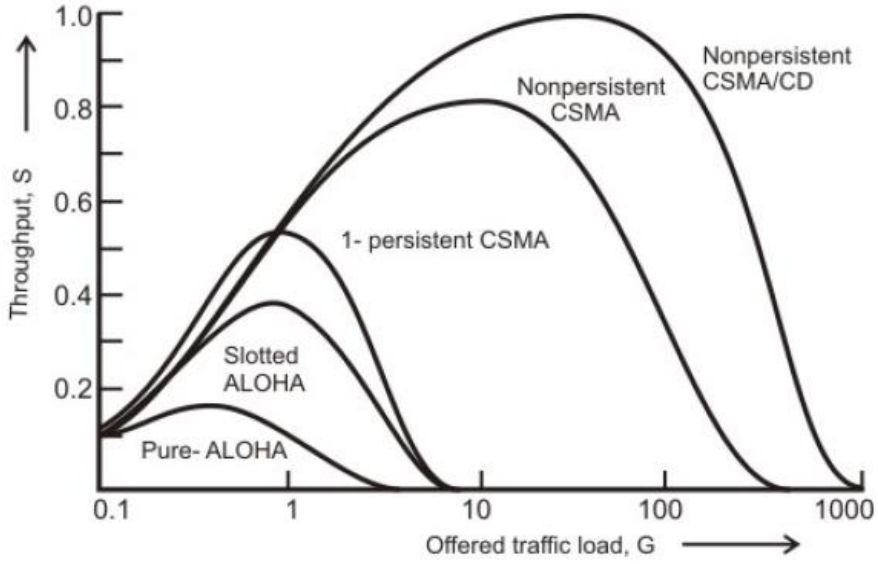


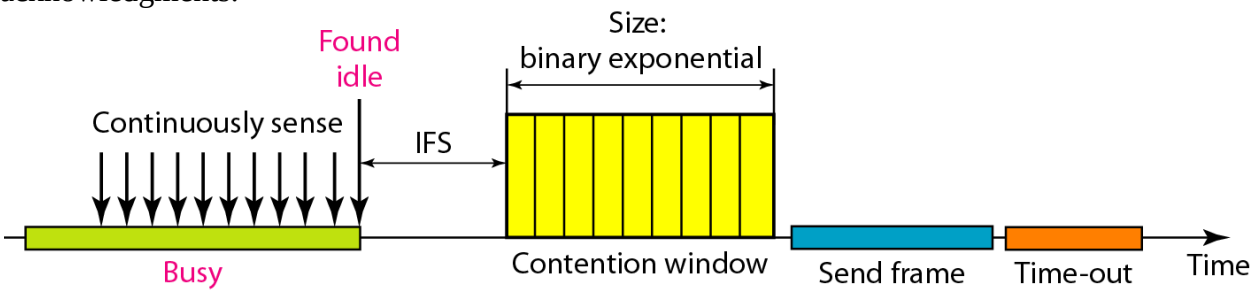
Figure 2.13 A plot of the offered load versus throughput for the value of a = 0.01

Performance Comparison between CSMA/CD and Token ring: It has been observed that smaller the mean packet length, the higher the maximum mean throughput rate for token passing compared to that of CSMA/CD. The token ring is also least sensitive to workload and propagation effects compared to CSMS/CD protocol. The CSMA/CD has the shortest delay under light load conditions, but is most sensitive to variations to load, particularly when the load is heavy. In CSMA/CD, the delay is not deterministic and a packet may be dropped after fifteen collisions based on binary exponential back off algorithm. As a consequence, CSMA/CD is not suitable for real-time traffic.

2.7 CSMA/CA:

- In wireless network, much of the sent energy is lost in transmission. The received signal is weak and hence not useful for effective collision detection.

- Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA) was invented for wireless network.
- Collisions are avoided through the use of inter-frame space, contention window and acknowledgments.



Inter frame Space (IFS)

- First, collisions are avoided by deferring transmission even if the channel is found idle. It waits for a period of time called the IFS.
- The IFS time allows first bit of the transmitted signal by the distant station to reach this station.
- If after IFS time the channel is still idle, the station can send but waits for a time equal to the contention time.

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 back-off strategy.
- The station senses the channel after each time slot.
- If the channel is busy, the timer is stopped and restarted when the channel is idle. This gives priority to station with longest waiting time.

Acknowledgment

- Despite IFS and contention window, there still may be a collision.
- In addition, the data may be corrupted during the transmission.
- The positive acknowledgment and the time-out mechanism guarantee that the receiver has received the frame correctly.

Why CSMA/CD cannot be used in wireless environment?

1. Collision detection requires costly stations and increased bandwidth.
2. Collision may not be detected because of the hidden station problem.
3. Signal fading can prevent a station at one end from hearing a collision at other end.

2.8 Ethernet:

A LAN consists of shared transmission medium and a set of hardware and software for interfacing devices to the medium and regulating the ordering access to the medium. These are used to share resources (may be hardware or software resources) and to exchange information. LAN protocols function at the lowest two layers of the OSI reference model: the physical and data-link layers. The IEEE 802 LAN is a shared medium peer-to-peer communications network that broadcasts information for all stations to receive. As a consequence, it does not inherently provide privacy. A LAN enables stations to communicate directly using a common physical medium on a point-to-point basis without any intermediate switching node being required. There is always need for an access sub layer in order to arbitrate the access to the shared medium.

To satisfy diverse requirements, the standard includes CSMA/CD, Token bus, Token Ring medium access control techniques along with different topologies. All these standards differ at the physical layer and MAC sub layer, but are compatible at the data link layer.

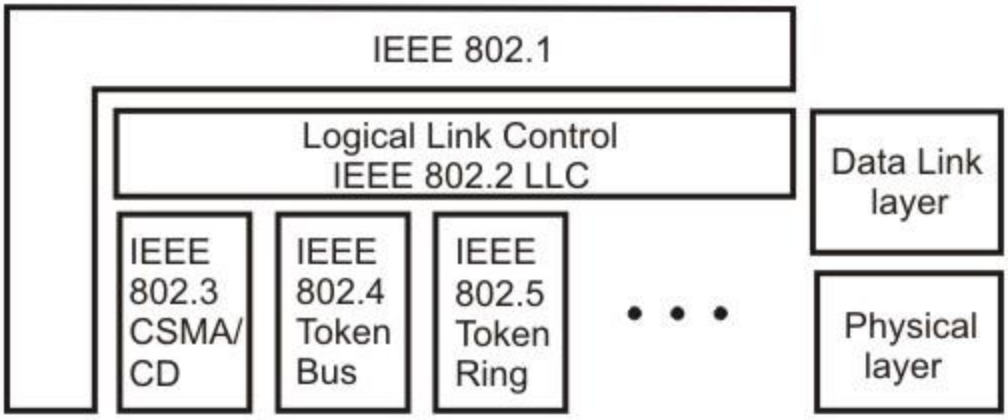


Figure 2.8.1 IEEE 802 Legacy LANs

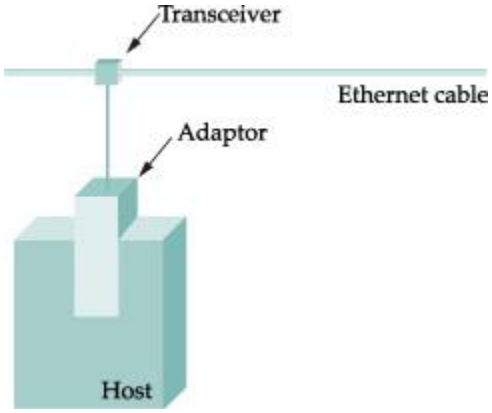
The **802.1** sublayer gives an introduction to set of standards and gives the details of the interface primitives. It provides relationship between the OSI model and the 802 standards. The **802.2** sublayer describes the **LLC** (logical link layer), which is the upper part of the data link layer. LLC facilitate error control and flow control for reliable communication. It appends a header containing sequence number and acknowledgement number. And offers the following three types of services:
Unreliable datagram service
Acknowledged datagram service
Reliable connection oriental service

The standards 802.3, 802.4 and 802.5 describe three LAN standards based on the CSMA/CD, token bus and token ring, respectively. Each standard covers the physical layer and MAC sub layer protocols.

IEEE 802.3 and Ethernet:

- Digital Equipment and Intel Corporation joined Xerox to define Ethernet standard in 1978. It then formed the basis for IEEE standard 802.3
- Standard Ethernet has a data rate of 10 Mbps. It is easy to administer and maintain.
- Ethernet has evolved to Fast Ethernet (100 Mbps), Gigabit Ethernet (1 Gbps) and Ten-Gigabit Ethernet (10 Gbps).
- Ethernet is limited to supporting a maximum of 1024 hosts.
- Ethernet has a total reach of only 2500 m by using repeaters.
- Ethernet produces better throughput only under lightly loaded conditions.

Physical Properties



- Ethernet uses Manchester encoding scheme and digital signaling (baseband) at 10 Mbps.
- Hosts connect to the Ethernet segment by tapping, each at least 2.5 m apart.
- The transceiver is responsible for transmitting/receiving frames and collision detection.
- Protocol logic is implemented in the adaptor.
- The various forms of Standard Ethernet are 10Base5, 10Base2, 10Base-T & 10Base-F

10Base5 (*Thick Ethernet*)

- Thick Ethernet uses bus topology with an external *transceiver*
- Collision occurs only in the thick coaxial cable.
- The maximum length of the cable must not exceed 500m.

10Base2 (*Thin Ethernet*)

- Thin Ethernet also uses bus topology, but the cable is much thinner and more flexible.
- The transceiver is part of the network interface card (NIC).
- The length of each segment should not exceed 185m and is less expensive.

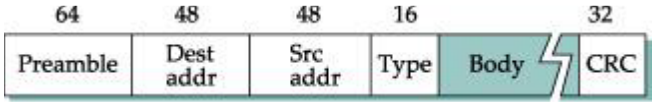
10Base-T (*Twisted-Pair Ethernet*)

- 10Base-T uses a star topology.
- The stations are connected to a hub via two pairs of twisted cable.
- The maximum length of the twisted cable is 100m
- Collision happens in the hub only.

10Base-F (*Fiber Ethernet*)

- 10Base-F uses star topology to connect stations to a hub.
- Stations are connected to the hub using two fiber-optic cables with a max. length of 2000m.

Frame Format



- *Preamble*—contains alternating 0s and 1s that alerts the receiving system and enables it to synchronize its input timing.
- *Destination address*—contains physical address of the destination host.
- *Source address*—contains the physical address of the sender.
- *Type/Length*—It may contain either type of the upper layer protocol or frame length.
- *Data*—carries data (46–1500 bytes) encapsulated from the upper-layer protocols.
- *CRC*—the last field contains error detection information (CRC-32).

Addressing

- Each host on the Ethernet network has its own network interface card (NIC).
- NIC provides a globally unique 6-byte physical address (in hex, delimited by colon).
  - Each manufacturer is given a unique prefix (3 bytes).
- If LSB of the first byte in a destination address is 0, then it is *unicast* else *multicast*.
- In *broadcast* address, all bits are 1s.

Transmitter algorithm

- Ethernet is a working example of CSMA/CD.
- Minimum frame length (64 bytes) is required for operation of CSMA/CD.
- Signals placed on the ethernet propagate in both directions and is broadcasted over the entire network.
- Ethernet is said to be a 1-persistent protocol. When the adaptor has a frame to send:
  - If line is idle, it transmits the frame immediately.
  - If line is busy, it waits for the line to go idle and then transmits immediately.
- It is possible for two (or more) adaptors to begin transmitting at the same time.
  - In such case, the frames collide
  - A 96-bit *runt* frame (64-bit preamble + 32-bit jamming sequence) is sent and transmission is aborted.
- Retransmission is attempted after a back-off procedure ($k \times 51.2\mu s$)

Receiver algorithm

- Each frame transmitted on an Ethernet is received by every adaptor on that network.
- Ethernet receives broadcast frames, frames addressed to it and multicast frames if it belongs to that group. Otherwise frames are discarded
- Receives all frames, if it runs in promiscuous mode.
- Ethernet does not provide any mechanism for acknowledging received frames, making it as an unreliable medium.

What is fast ethernet

- Fast Ethernet is IEEE 802.3u standard and was invented compete with LAN protocols such as FDDI.
- The data rate in Fast ethernet is 100 Mbps.
- Fast ethernet uses star topology.

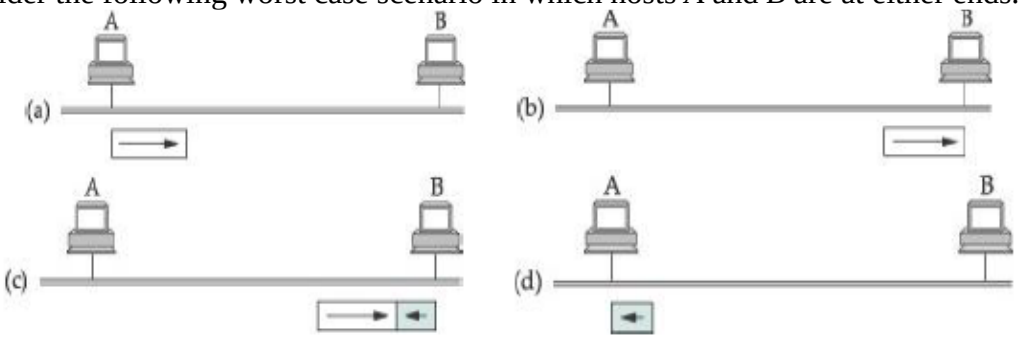
List the function of a repeater?

- A repeater is a device that connects segments of a LAN.
- A repeater reconstructs the received weak signal to its original strength and forwards it.
- It operates in the physical layer.

 It can extend the length of a LAN network.

Why the minimum frame length in Ethernet should be at least 64 bytes?

 Consider the following worst case scenario in which hosts *A* and *B* are at either ends.



 Host A begins transmitting a frame at time t, as shown in (a).

 It takes one link latency (say d) for the frame to reach host B. Thus, the first bit of A's frame arrives at B at time t + d, as shown in (b).

 Suppose an instant before host A's frame arrives, B senses it idle line, host B begins to transmit its own frame.

 B's frame will immediately collide with A's frame, and this collision will be detected by host B (c). Host B will send the 32-bit jamming sequence.

 Host A will not know that the collision occurred until B's frame reaches it, i.e., at time t + 2d, as shown in (d).

 On a maximally configured Ethernet, the round-trip delay is 51.2 μs, i.e., 512 bits (64 bytes)

Successors of Ethernet

On a regular Ethernet segment, all stations share the available bandwidth of 10 Mb/s. With the increase in traffic, the number of packet collisions goes up, lowering the overall throughput. In such a scenario, there are two basic approaches to increase the bandwidth.

One is to replace the Ethernet with a higher speed version of Ethernet. Use of Fast Ethernet operating at 100 Mb/s and Gigabit Ethernet operating at 1000 Mb/s belong to this category. This approach requires replacement of the old network interface cards (NICs) in each station by new ones.

The other approach is to use Ethernet switches (let us call it switched Ethernet approach) that use a high-speed internal bus to switch packets between multiple (8 to 32) cable segments and offer dedicated 10 Mb/s bandwidth on each segment/ports. In this approach, there is no need to replace the NICs; replacement of the hub by a switch serves the purpose. This approach is discussed in the following section.

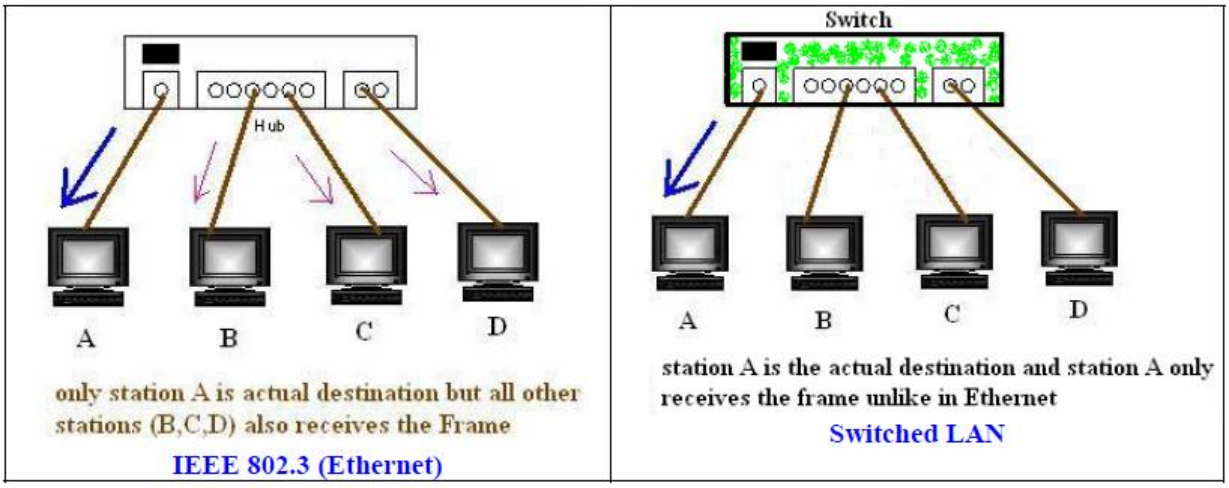
Switched Ethernet

Switched Ethernet gives dedicated 10 Mb/s bandwidth on each of its ports. On each of the ports one can connect either a thick/thin segment or a computer.

In Ethernet (IEEE 802.3) the topology, though physically is star but logically is BUS, i.e. the collision domain of all the nodes in a LAN is common. In this situation only one station can send the frame. If more than one station sends the frame, there is a collision. A comparison between the two is shown in Fig below

In Switched Ethernet, the collision domain is separated. The hub is replaced by a switch, which functions as a fast bridge. It can recognize the destination address of the received frame and can forward the frame to the port to which the destination station is connected. The other ports are not involved in the transmission process. The switch can receive another frame from another station at

the same time and can route this frame to its own final destination. In this case, both the physical and logical topologies are star.



There are two possible forwarding techniques that can be used in the implementation of Ethernet switches: *store-and-forward* and *cut-through*. In the first case, the entire frame is captured at the incoming port, stored in the switch's memory, and after an address lookup to determine the LAN destination port, forwarded to the appropriate port. The lookup table is automatically built up. On the other hand, a cut-through switch begins to transmit the frame to the destination port as soon as it decodes the destination address from the frame header.

Store-and-forward approach provides a greater level of error detection because damaged frames are not forwarded to the destination port. But, it introduces longer delay of about 1.2 msec for forwarding a frame and suffers from the chance of loosing data due to reliance on buffer memory. The cut-through switches, on the other hand, has reduced latency but has higher switch cost.

The throughput can be further increased on switched Ethernet by using full-duplex technique, which uses separate wire pairs for transmitting and receiving. Thus a station can transmit and receive simultaneously, effectively doubling the throughput to 20 Mb/s on each port.

Fast Ethernet:

The 802.u or the fast Ethernet, as it is commonly known, was approved by the IEEE 802 Committee in June 1995. It may not be considered as a new standard but an addendum to the existing 802.3 standard. The fast Ethernet uses the same frame format, same CSMA/CD protocol and same interface as the 802.3, but uses a data transfer rate of 100 Mb/s instead of 10 Mb/s. However, fast Ethernet is based entirely on 10-Base-T, because of its advantages (Although technically 10-BASE-5 or 10-BASE-2 can be used with shorter segment length).

Fortunately, the Ethernet is designed in such a way that the speed can be increased if collision domain is decreased. The only two changes made in the MAC layer are the data rate and the collision domain. The data rate is increased by a factor of 10 and collision domain is decreased by a factor of 10. To increase the data rate without changing the minimum size of the frame (576 bits or 76 bytes in IEEE 802.3), it is necessary to decrease the round-trip delay time. With the speed of 100Mbps the round-trip time reduce to 5.76 microseconds (576 bits/100 Mbps; which was 57.6 microsecond for 10Mbps Normal Ethernet). This means that the collision domain is decreased 10 fold from 2500 meters (in IEEE802.3) to 250 meters (fast Ethernet).

IEEE has designed two categories of Fast Ethernet: 100Base-X and 100Base-T4. 100Base-X uses two-wire interface between a hub and a station while 100Base-T4 uses four-wire interface. 100-Base-X itself is divided into two: 100Base-TX and 100base-FX

Gigabit Ethernet:

As applications increased, the demand on the network, newer, high-speed protocols such as FDDI and ATM became available. However, in the last couple of years, Fast Ethernet has become the backbone of choice because it's simplicity and its reliance on Ethernet. The primary goal of Gigabit Ethernet is to build on that topology and knowledge base to build a higher-speed protocol without forcing customers to throw away existing networking equipment.

In March 1996, the IEEE 802.3 committee approved the 802.3z Gigabit Ethernet Standardization project. At that time as many as 54 companies expressed there intent to participate in the standardization project. The Gigabit Ethernet Alliance was formed in May 1996 by 11 companies. The Alliance represents a multi-vendor effort to provide open and inter-operable Gigabit Ethernet products. The objectives of the alliance are:

Supporting extension of existing Ethernet and Fast Ethernet technology in response to demand for higher network bandwidth.

Developing technical proposals for the inclusion in the standard

Establishment of inter-operability test procedures and processes

Similarities and advances over Ethernet (IEEE 802.3)

As its name implies, Gigabit Ethernet - officially known as 802.3z - is the 1 Gb/s extension of the 802.3 standard already defined for 10 and 100 Mb/s service. Gigabit Ethernet builds on top of the Ethernet protocol, but increases speed tenfold over Fast Ethernet to 1000 Mbps, or 1 gigabit per second (Gbps). It retains the Carrier Sense Multiple Access/ Collision Detection (CSMA/CD) as the access method. It supports full duplex as well as half duplex modes of operation. Initially, single-mode and multi mode fiber and short-haul coaxial cable were supported. Standards for twisted pair cables were subsequently added. The standard uses physical signaling technology used in Fiber Channel to support Gigabit rates over optical fibers. Since Gigabit Ethernet significantly leverages on Ethernet, customers will be able to leverage their existing knowledge base to manage and maintain gigabit networks. Initially, Gigabit Ethernet was expected to be used as a backbone system in existing networks. It can be used to aggregate traffic between clients and "server farms", and for connecting Fast Ethernet switches. It can also be used for connecting workstations and servers for high-bandwidth applications such as medical imaging or CAD. But, gigabit Ethernet is not simply a straight Ethernet running at 1 Gb/s. In fact, the ways it differs from its predecessors may be more important than its similarities. Some of the important differences are highlighted below

(i) The cabling requirement of gigabit Ethernet is very different. The technology is based on fiber optic cable. Multi-mode fiber is able to transmit at gigabit rate to at least 580 meters and with single-mode runs exceeding 3 km. Fiber optic cabling is costly. In order to reduce the cost of cabling, the 802.3z working group also proposed the use of twisted-pair or cable or coaxial cable for distances up to 30 meters.

(ii) Gigabit Ethernet also relies on a modified MAC layer. At gigabit speed, two stations 200 meters apart will not detect a collision, when both simultaneously send 64-byte frames. This inability to detect collision leads to network instability. A mechanism known as *carrier extension* has been proposed for frames shorter than 512 bytes. The number of repeater hops is also restricted to only one in place of two for 100 Base-T.

(iii) Flow Control is a major concern in gigabit Ethernet because of buffer overflow and junked frames in heavily loaded condition. The solution proposed by IEEE subcommittee is the 802.3x. The X-on/X-off protocol works over any full-duplex Ethernet, fast Ethernet or gigabit Ethernet link. When a switch buffer is close to capacity, the receiving device signals the sending station and tells it to stop transmitting until the buffer becomes empty.

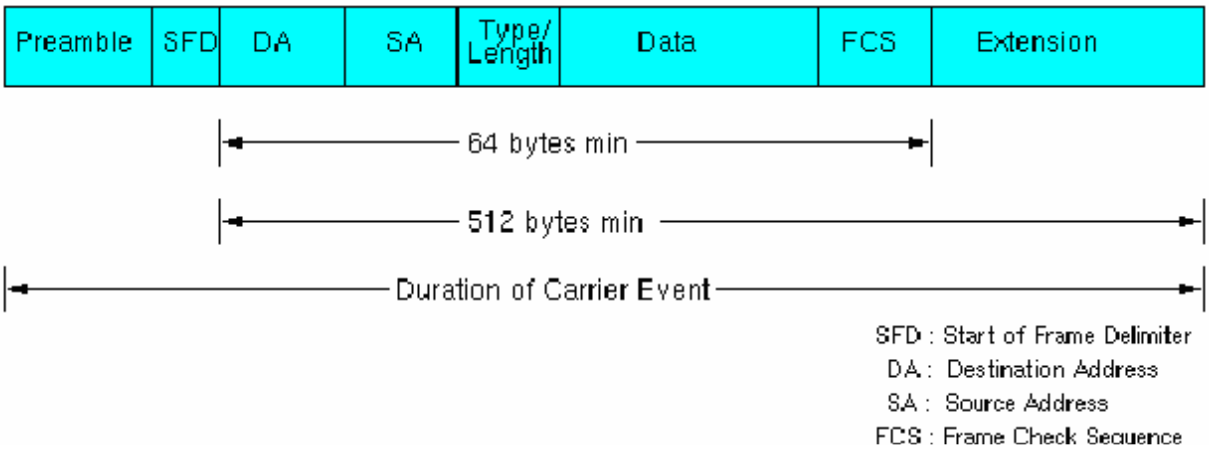
(iv) Finally, one important feature, which Ethernet technology lacks, is the Quality of Service (QoS). The gigabit Ethernet is a connectionless technology that transmits variable length frames. As such, it simply cannot guarantee that the real-time packets get the preferential treatment they require. The IEEE subcommittee developed two specifications that will help Ethernet provide the required QoS. 802.lq tags traffic for VLANs and for prioritization. 802.lp is a signaling scheme that lets end station request priority and allows switches to pass these requests along the path.

The gigabit Ethernet comes into its own as an internetworking switch link (ISL) that aggregates 10-and100-Mb/s feeds from the desktops and servers. Presently, gigabit Ethernet is already matured with a large installation base as a backbone network technology.

Gigabit Ethernet has been designed to adhere to the standard Ethernet frame format. This setup maintains compatibility with the installed base of Ethernet and Fast Ethernet products, requiring no frame translation. Gigabit Ethernet maintains the minimum and maximum frame sizes of Ethernet. Since, Gigabit Ethernet is 10 times faster than Fast Ethernet, to maintain the same slot size, maximum cable length would have to be reduced to about 10 meters, which is not very useful. Instead, Gigabit Ethernet uses a bigger slot size of 512 bytes (In Ethernet, the slot size is 64 bytes, the minimum frame length). To maintain compatibility with Ethernet, the minimum frame size is not increased, but the "carrier event" is extended. If the frame is shorter than 512 bytes, then it is padded with extension symbols. These are special symbols, which cannot occur in the payload. This process is called *Carrier Extension*

Carrier Extension

Gigabit Ethernet should be inter-operable with existing 802.3 networks. Carrier Extension is a way of maintaining 802.3 minimum and maximum frame sizes with meaningful cabling distances.



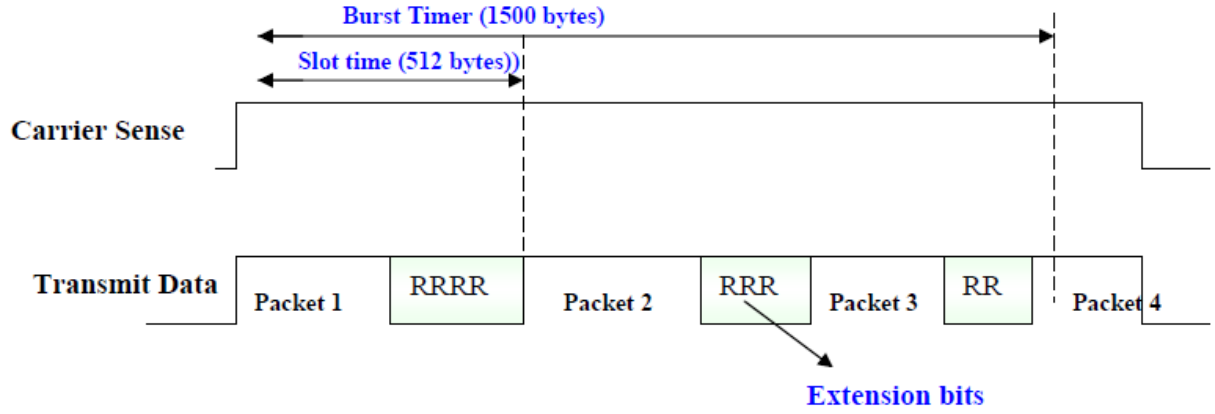
For carrier extended frames, the non-data extension symbols are included in the "collision window", that is, the entire extended frame is considered for collision and dropped. However, the Frame Check Sequence (FCS) is calculated only on the original (without extension symbols) frame. The extension symbols are removed before the FCS is checked by the receiver. So the LLC (Logical Link Control)

layer is not even aware of the carrier extension. Figure above shows the Ethernet frame format when Carrier Extension is used.

Packet Bursting

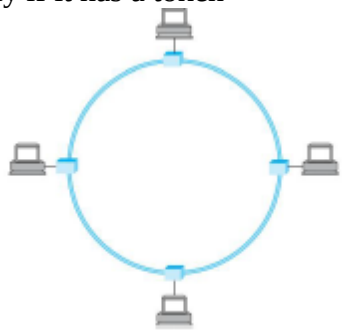
Carrier Extension is a simple solution, but it wastes bandwidth. Up to 448 padding bytes may be sent for small packets. This results in lower throughput. In fact, for a large number of small packets, the throughput is only marginally better than Fast Ethernet.

Packet Bursting is an extension of Carrier Extension. Packet Bursting is "Carrier Extension plus a burst of packets". When a station has a number of packets to transmit, the first packet is padded to the slot time if necessary using carrier extension. Subsequent packets are transmitted back to back, with the minimum Inter-packet gap (IPG) until a burst timer (of 1500 bytes) expires. Packet Bursting substantially increases the throughput. Figure below shows how Packet Bursting works.



2.9 Token Ring:

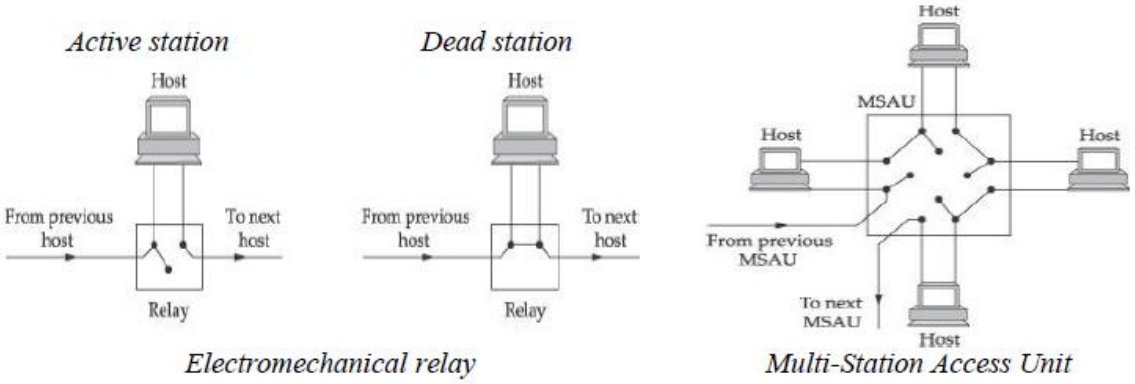
- Developed by IBM and later became standard IEEE 802.5
- A token ring network consists of a set of nodes connected in a ring
- Data flow is unidirectional
- It is based on a small frame called *token* that circulates around the ring.
- A station can transmit data only if it has a token



Physical Properties

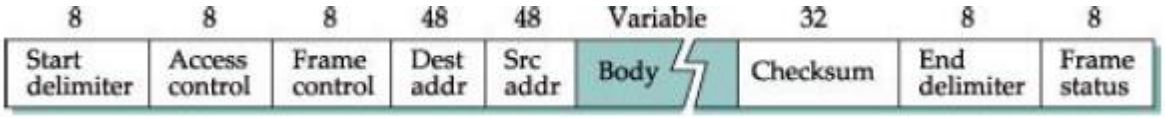
- In a ring topology, any link or node failure would render the whole network useless
- Therefore each station is connected to the ring using an *electromechanical relay*.
  - When the station is healthy, relay is open and the station is included in the ring.
  - If the station goes down, relay closes and the ring bypasses the station.

- Multiple relays packed in a single unit is known as a multi-station access unit (*MSAU*).
- MSAU makes it easy to add and remove stations from the network
- The data rate is 4 or 16 Mbps and uses differential Manchester encoding.
- Twisted-pair is widely used as the physical medium.
- A maximum of 250 stations can be included in the ring.



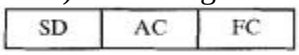
Electromechanical relay *Multi-Station Access Unit*

Frame format



- *Start/End delimiter*—contains Manchester codes that indicates start/end of a frame
- *Access control*—includes 3 priority bits (P) and 3 reservation bits (R). The T bit indicates whether the frame is token or data. M is monitor bit.
- *Frame control*—is a demux key that identifies the higher-layer protocol
- *Destination* and *Source address*—contains 6 bytes of source and destination address
- *Checksum*—is a 32-bit CRC used for error detection
- *Frame status*—Contains address recognized (A) and frame-copied (C) bits

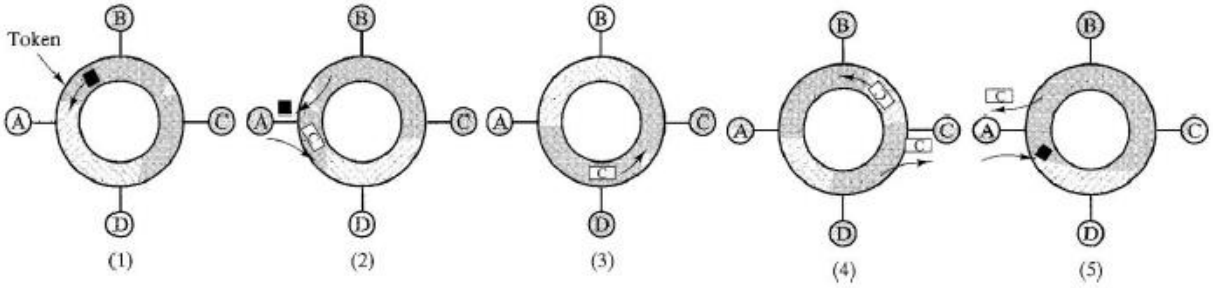
The *token frame* is a 3-byte frame (24 bits) containing the first three fields of the data frame.



Medium Access Control (MAC)

- The network adaptor contains a transmitter, receiver and data storage.
- As the token circulates around the ring, any station that has data to send, seizes the token by changing a bit in the token so that it becomes *start delimiter* for a data frame
- Once a station has the token, it is allowed to send one or more frames. Each frame contains the destination address of the intended receiver(s).
- Each node checks the data frame to see if it is the intended recipient, as frame passes through.
  - If so, it copies the packet into a buffer as it flows through the network adaptor.
- The sender has the responsibility of removing the packet from the ring.
  - It inserts a new token on the ring after the frame reaches it back.

The following example shows token capture by station A, its subsequent data transmission and eventual token release.



- *Token Holding Time* (THT) specifies how long a given node is allowed to hold the token
  - Allowing a node to send as much as it wants will result in better utilization of the ring. This strategy will fail if multiple nodes have data to send
  - The default THT is 10 ms
- *Token Rotation Time* (TRT) is the amount of time taken by the token to traverse the ring
  - TRT = ActiveNodes × THT + RingLatency
  - ActiveNodes is number of nodes that wish to transmit data
  - RingLatency refers to time taken for token to circulate the ring when no station wishes to transmit data
- *Reliable* delivery is provided using 2 bits in the packet trailer (A and C bits).
  - The intended recipient sets the A bit when the frame comes to it
  - It sets the C bit after copying the frame. For frame received A = 1 and C = 1
  - If destination station is non-existent or inactive, then A = 0 and C = 0
  - If destination station exists but frame not copied, then A = 1 and C = 0
- The token ring supports different levels of priority by 3-bit *priority* and *reservation* field.
  - Each station that wants to send a frame assigns a priority to that frame.
  - The station can seize the token, only if its *frame priority* ≥ *token priority*
  - A station having a higher priority frame to transmit than the current frame can *reserve* the next token for its priority level as the frame passes by.
  - When the next token is issued, it will be at the *reserved priority* level.
  - When the station that had upgraded, sees a token at the higher priority once again, it presumes all high priority traffic is over and restores the token priority to its original value.
  - Priority may lead to starvation of low-priority packets.

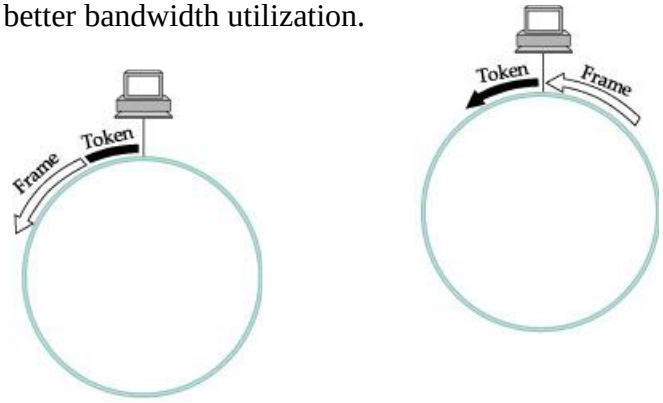
Monitor

- Token rings have one of the stations designated as a *monitor.*
- The monitor's job is to ensure that the ring stays alive.
- A monitor periodically announces its presence with a special *control message*.
  - If a station fails to see that message for some period of time, it assumes that the monitor has failed and will try to become the monitor as follows:
  - It transmits a *claim token* frame
  - If that token circulates back to itself, then the station becomes monitor.
  - If more than one station competes to become monitor, then highest address wins.

- Monitor ensures that either token circulates in the ring or is held by some station.
  - To detect *missing token*, the monitor watches for a passing token within the interval NumStations × THT + RingLatency.
  - If it fails to see the token, it inserts a new token
- The monitor also checks for *corrupted* or *orphaned* frames.
  - The monitor sets a header bit in the data frame when it passes through once.
  - When monitor sees the header bit set in a frame, it indicates the frame is looping(*corrupted*) and is removed from the ring
- Detection of *dead stations* is done by sending a *beacon* frame to the suspect destination.
  - Based on how far the frame goes, the status of the ring is determined.
  - The malfunctioning stations are bypassed using relays in MSAU.

Distinguish between early and delayed release?

- The sender can insert the token back onto the ring immediately following its frame (*early* release) or after the frame it transmits gets back to it (*delayed* release).
- Early release allows better bandwidth utilization.



Early release *Delayed release*

2.10 FDDI:

Physical Properties

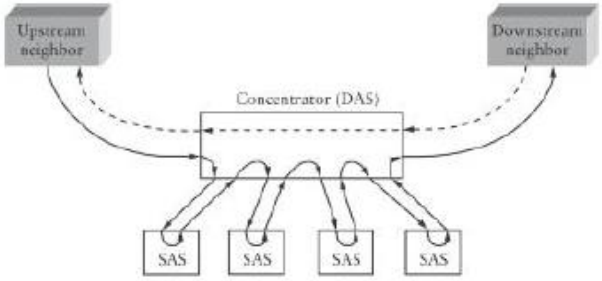
- FDDI network consists of two independent rings that transmit data in opposite directions.
- The second ring is used only if the primary fails.
- FDDI network is fault tolerant to a link or a station failure.
- FDDI has a data rate of 100 Mbps and is designed for LAN and MAN.

Normal operation *Failure of primary ring*

- Since dual rings are expensive, FDDI also allows station to connect to the network by

means of a single cable. Such stations are known as *single attachment station* (SAS).

- Dual cable connected stations are called *dual attachment stations* (DAS).
- A *concentrator* is used to attach several SAS to the dual ring.
- The concentrator is able to detect failure of a SAS and isolates it using optical bypass. Thus the ring is always connected.

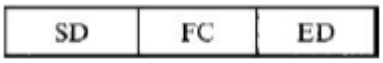


- The buffer in network adaptor for each station can hold up to 9–80 bits.
- FDDI is a 100-Mbps network with a 10 ns bit time.
- A FDDI network can have at most 500 stations with a maximum distance of 2 km between any pair of stations.
- FDDI uses 4B/5B encoding.
- Even though the network is limited to a total of 200 km of fiber, it is actually 100 km due to dual nature of ring.
- The physical medium is not mandated to be fiber-optic, could be coax or twisted pair.

Frame Format

- *Start/End of frame*Indicates start/end of frame. It contains 4B/5B control symbol.
- *Frame control*indicates whether the traffic is synchronous or asynchronous.
- *Source / Destination address*specifies 48-bit source and destination address.
- *CRC*contains 32-bit CRC code used for error detection.
- *Frame status*contains error detected (E), address recognized (A), frame copied (F) bits.

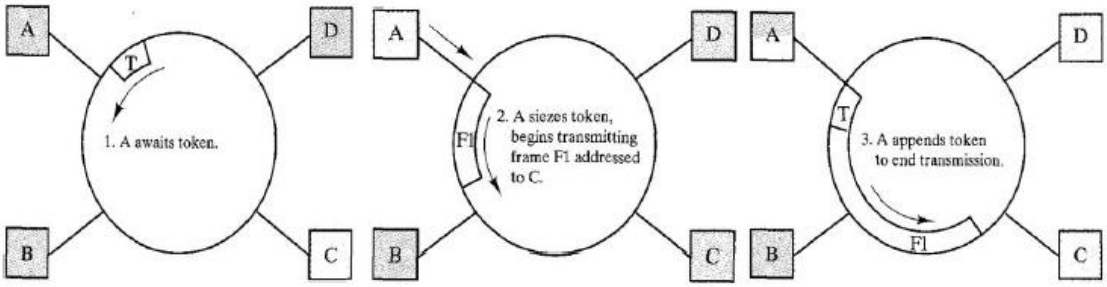
The FDDI token frame has 24 bits with control field value as either 10000000 or 11000000



FDDI MAC

- In FDDI, a station after capturing the token begins transmitting one or more data frames. It does not alter any bit as in case of token ring.
- The station after data transmission releases a new token immediately (*early release*).
- As in token ring, the intended recipients sets the following bits in *frame status* field:
  - The A bit is set, if it detects its own address in the *destination* address field.
  - The C bit is set, after copying the frame.
  - The E bit is set, if an error is detected.
- FDDI have no priority or reservation mechanism to capture a token.

The following example shows that after station A has seized the token, it transmits frame F1, and immediately transmits a new token.



Timed Token Algorithm

- The THT for each node is defined according to the network.
- All nodes agree to a *target token rotation time* (TTRT)
- Each node measures the time between successive arrivals of token as *measured* TRT.
- If a node's measured TRT > TTRT, then the token is late, and does not transmit any data.
- Otherwise the token is early, and the node is allowed to hold the token for the duration TTRT - TRT. It transmits data, if it can send a full frame.
- FDDI defines two classes of traffic: *synchronous* (delay sensitive) and *asynchronous* (throughput-based):

o When a node receives a token, it is allowed to send synchronous (delay sensitive) data, even if the token is late.

o The total amount of synchronous data that can be sent during one token rotation is also bounded by TTRT

o A node can send asynchronous traffic only when the token is early.

o When a node has both types of data to transmit, it transmits asynchronous traffic up to a TTRT time and then synchronous traffic for another TTRT

o Thus a single rotation of token can take a maximum 2 × TTRT

Token Maintenance

- All nodes on an FDDI ring monitor the ring to be sure that the token has not been lost.
- The idle time for a node in FDDI should be 2.5 ms.

o Each node sets a timer to 2.5 ms after it notices a data frame or token frame.

o If this timer expires, the node suspects that something has gone wrong and transmits a *claim frame*.

o The claim frame contains the node's TTRT estimate based on requirements of application running on it.

- When a node receives a claim frame

o If the TTRT bid in the frame is less than its own, then it's estimate is changed to TTRT bid and forwards the claim frame.

o If the bid TTRT is greater than the node's estimate, then claim frame is removed and the node inserts its own claim frame.

o If the TTRT bid and the node's estimate are the same, then one with the higher address wins.

- If this claim frame makes it all the way around the ring, then

o The node removes the claim frame and inserts a new token on the ring

- The node's estimate becomes TTRT.

Difference between Token ring and FDDI network.

- Token ring is a single ring whereas FDDI uses double ring.
- Token ring uses twisted pair cable whereas FDDI uses fiber-optic cable.
- Token ring uses Manchester coding whereas FDDI uses 4B/5B encoding.
- The data rate is 16 Mbps for token ring whereas FDDI data rate is 100 Mbps.
- For healthy ring, FDDI uses concentrators whereas token ring uses MSAU.
- Token ring includes bits for priority and reservation, whereas FDDI does not.

2.11 Wireless LAN

Wireless LAN 802.11 is designed for use in a limited geographical area (office, campus, etc)

Physical Properties

- 802.11 was designed to run over three physical media namely FHSS, DSSS and infra red.

Spread spectrum (FHSS/DSSS)

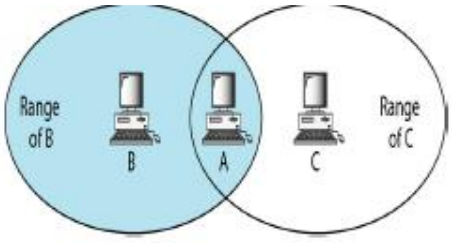
- Operates in 2.4GHz ISM band with a data rate of 11 Mbps.
- In FHSS, signal is transmitted over a random sequence of frequencies
- In DSSS for each data bit, the sender transmits the XOR of that bit and *n* random bits.
- A pseudorandom number generator is used to select the hopping sequence/random bits
- The modulation techniques used are frequency shift keying (FSK) and phase shift keying (PSK) respectively.
- In both methods except for intended receiver, it would remain as noise for other nodes.

Infrared

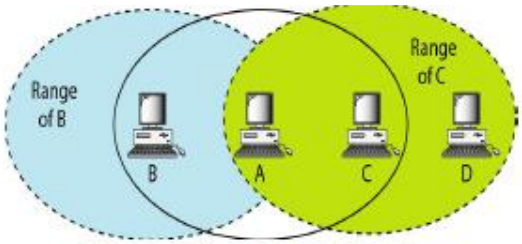
- It uses infrared light in the range of 800 to 950 nm.
- The modulation technique is called pulse position modulation (PPM).
- The sender and receiver do not need to have a clear line of sight.
- It has a range of up to about 10 m and is limited to the inside of buildings only.
- The data rate is up to 2 Mbps.

Collision Avoidance

- Collision detection is not feasible, since all nodes are not within the reach of each other.
- The two major problems are *hidden* and *exposed* terminals.
- In figure, each node is able to send and receive signals from nodes to its immediate left and right.



Hidden Station



Exposed Station

Hidden Station

- Suppose station *B* is sending data to *A*. At the same time, station *C* also has data to send to station *A*.
- Since *B* is not within the range of *C*, it thinks the medium is free and sends its data to *A*.

Frames from *B* and *C* collide at *A.* Stations *B* and *C* are *hidden* from each other.

Exposed Station

- Suppose station A is transmitting to station B and station C has some data to send to station D, which can be sent without interfering the transmission from A to B.
- Station C is exposed to transmission from A and it hears what A is sending and thus refrains from sending, even if the channel is available

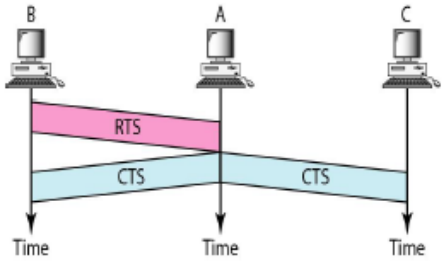
Multiple Access with Collision Avoidance (MACA)

- The idea is for the sender and receiver to exchange short *control frames* with each other, so that stations nearby can avoid transmitting for the duration of the data frame.
- The control frames used for collision avoidance is *Request to Send* (RTS) and *Clear to Send* (CTS).
- Any station hearing RTS is close to sender and remains silent long enough for CTS to be transmitted back.
- Any station hearing CTS remains silent during the upcoming data transmission.
- The receiver sends an ACK frame to the sender after successfully receiving a frame.
- If RTS frames from two or more stations collide, then they do not receive CTS. Each node waits for a random amount of time and then tries to send RTS again

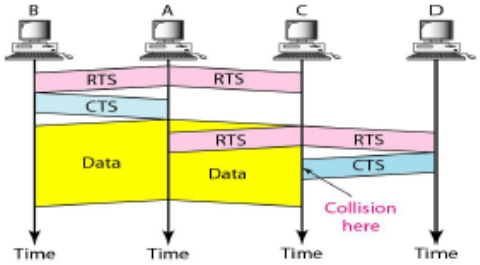
Handshake for hidden & exposed station

- *B* sends an RTS containing name of sender, receiver & duration of transmission.
- It reaches *A*, but not *C*.
- The receiver *A* acknowledges with a CTS message back to the sender *B* echoing the duration of transmission and other information.
- The CTS from *A* is received by both *B* and *C*. *B* starts to transmit data to *A.*
- *C* knows that some *hidden station* is using the channel and refrains from transmitting.
- The handshaking messages RTS and CTS *does not help in exposed stations* because C does not receive CTS from D as it collides with data sent by A.

Solution for hidden node Handshake for exposed node



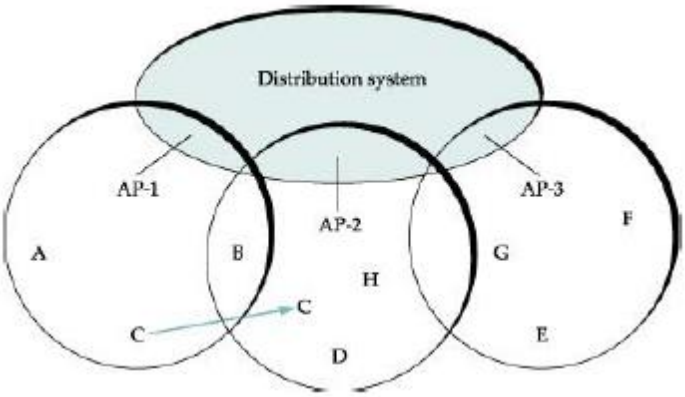
Solution for hidden node



Handshake for exposed node

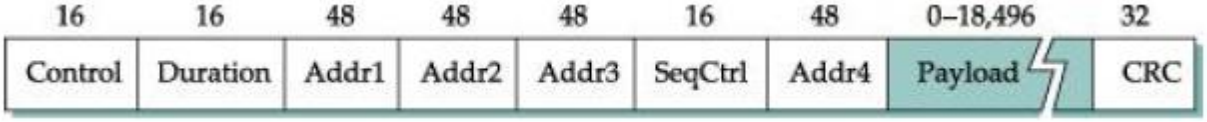
Distribution System

- In wireless network, nodes are mobile and the set of reachable nodes change with time.
- Mobile nodes are connected to a wired network infrastructure called *access points* (AP)
- Access points are connected to each other by a *distribution system* (DS) such as ethernet, token ring, etc



- Two nodes communicate directly with each other if they are reachable (eg, A and C)
- Communication between two stations in different APs occurs via two APs (eg, A and E)
- The technique for selecting an AP is called *active scanning.* It is done whenever a node joins a network or switches over to another AP.
  - The node sends a Probe frame.
  - All APs within reach reply with a Probe Response frame.
  - The node selects one of the APs and sends an Association Request frame.
  - The AP replies with an Association Response frame
- APs also periodically send a Beacon frame that advertises its features such as transmission rate. This is known as *passive scanning.*

Frame Format



- *Control*—contains subfields that includes *type* (management, control or data), *subtype* (RTS, CTS or ACK) and pair of 1-bit fields ToDS and FromDS.
- *Duration*—specifies duration of frame transmission.
- *Addresses*—The *four* address fields depend on value of ToDS and FromDS subfields.
  - When one node is sending directly to another, both DS bits are 0, Addr1 identifies the *target* node, and Addr2 identifies the *source* node
  - When both DS bits are set to 1, the message went from a node onto the distribution system, and then from the distribution system to another node. Addr1 contains *ultimate destination*, Addr2 contains *immediate sender*, Addr3 contains *intermediate destination* and Addr4 contains *original source*.
- *Sequence Control*—defines sequence number of the frame to be used in flow control.
- *Payload*—can contain a maximum of 2312 bytes and is based on the type and the subtype defined in the *Control* field.

- *CRC* contains CRC-32 error detection sequence.

2.12 Repeaters

A single Ethernet segment can have a maximum length of 500 meters with a maximum of 100 stations (in a cheapernet segment it is 185m). To extend the length of the network, a *repeater* may be used as shown in Fig. below. Functionally, a repeater can be considered as two transceivers joined together and connected to two different segments of coaxial cable. The repeater passes the digital signal bit-by-bit in both directions between the two segments. As the signal passes through a repeater, it is amplified and regenerated at the other end. The repeater does not isolate one segment from the other, if there is a collision on one segment, it is regenerated on the other segment. Therefore, the two segments form a single LAN and it is transparent to rest of the system. Ethernet allows five segments to be used in cascade to have a maximum network span of 2.5 km. With reference of the ISO model, a repeater is considered as a *level-1 relay* as depicted in Fig. below. It simply repeats, retimes and amplifies the bits it receives. The repeater is merely used to extend the span of a single LAN. Important features of a repeater are as follows:

A repeater connects different segments of a LAN
A repeater forwards every frame it receives
A repeater is a regenerator, not an amplifier
It can be used to create a single extended LAN

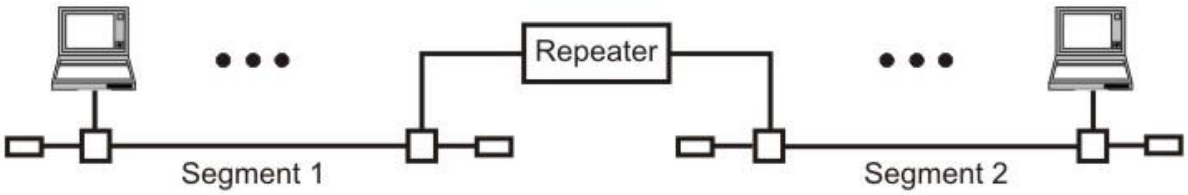


Figure :Repeater connecting two LAN segments

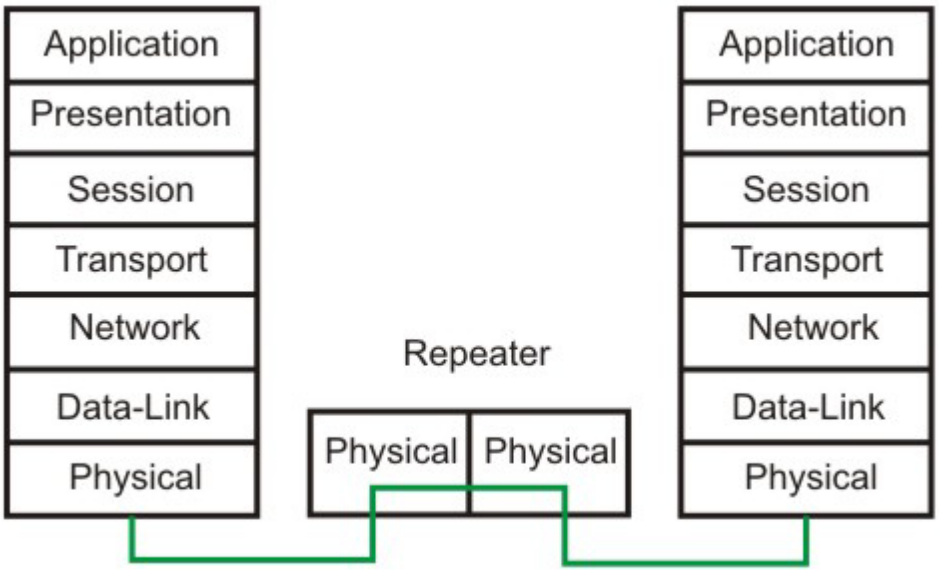


Figure :Operation of a repeater as a level-1 relay

2.13 Hubs

Hub is a generic term, but commonly refers to a multiport repeater. It can be used to create multiple levels of hierarchy of stations. The stations connect to the hub with RJ-45 connector having maximum segment length is 100 meters. This type of interconnected set of stations is easy to maintain and diagnose. Figure below shows how several hubs can be connected in a hierarchical manner to realize a single LAN of bigger size with a large number of nodes.

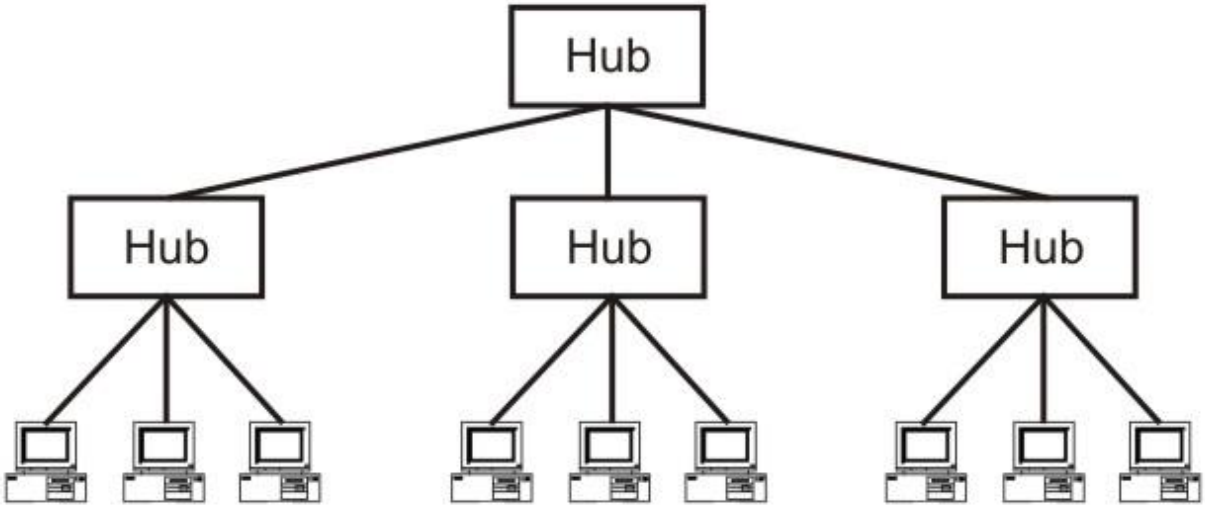


Figure : Hub as a multi-port repeater can be connected in a hierarchical manner to form a single LAN with many nodes

2.14 Bridge:

The device that can be used to interconnect two separate LANs is known as a *bridge*. It is commonly used to connect two similar or dissimilar LANs as shown in Fig. below. The bridge operates in layer 2, that is data-link layer and that is why it is called *level-2 relay* with reference to the OSI model. It links similar or dissimilar LANs, designed to store and forward frames, it is protocol independent and transparent to the end stations. The flow of information through a bridge is shown in Fig. below. Use of bridges offer a number of advantages, such as higher reliability, performance, security, convenience and larger geographic coverage. But, it is desirable that the quality of service (QOS) offered by a bridge should match that of a single LAN. The parameters that define the QOS include *availability, frame mishaps, transit delay*, *frame lifetime*, *undetected bit errors, frame size* and *priority*. Key features of a bridge are mentioned below:

- A bridge operates both in physical and data-link layer
- A bridge uses a table for filtering/routing
- A bridge does not change the physical (MAC) addresses in a frame

Types of bridges:

- o Transparent Bridges

o Source routing bridges

A bridge must contain addressing and routing capability. Two routing algorithms have been proposed for a bridged LAN environment. The first, produced as an extension of IEEE 802.1 and applicable to all IEEE 802 LANs, is known as *transparent bridge*. And the other, developed for the IEEE 802.5 token rings, is based on *source routing approach*. It applies to many types of LAN including token ring, token bus and CSMA/CD bus.

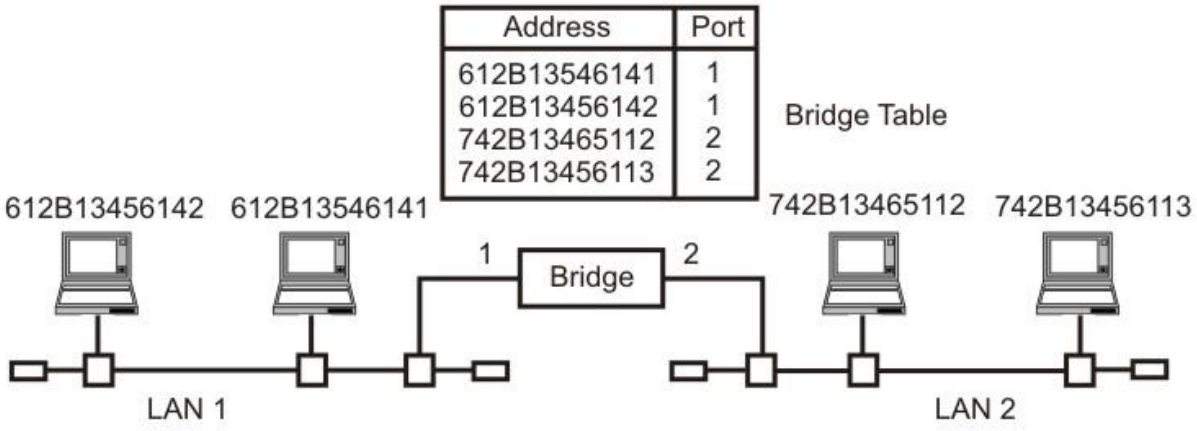


Figure : A bridge connecting two separate LANs

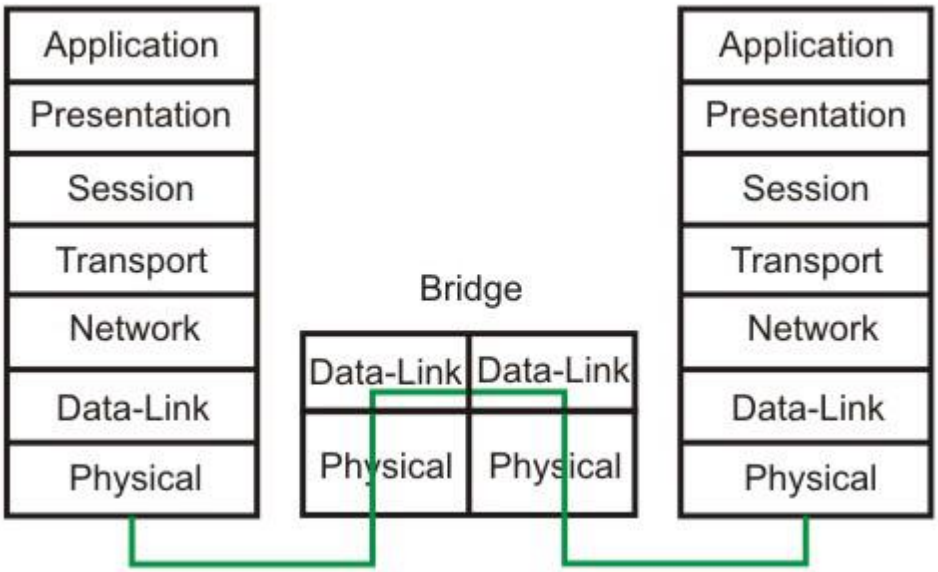


Figure: Information flow through a bridge

Transparent Bridges

The transparent bridge uses two processes known as **bridge forwarding** and **bridge learning**. If the destination address is present in the forwarding database already created, the packet is forwarded to the port number to which the destination host is attached. If it is not present, forwarding is done on all parts (flooding). This process is known as *bridge forwarding*. Moreover, as each frame arrives, its source address indicates where a particular host is situated, so that the bridge learns which way to forward frames to that address. This process is known as *bridge learning*. Key features of a transparent bridge are:

- The stations are unaware of the presence of a transparent bridge
- Reconfiguration of the bridge is not necessary: it can add/remove without being noticed
- It performs two functions:
    - Forwarding frames
    - Learning to create the forwarding table

Bridge forwarding:

Bridge forwarding operation is explained with help of flowchart and basic functions of the bridge forwarding are given below:

- Discard the frame if source and destination addresses are same
- Forward the frame if the source and destination address are different and destination address is present in the table
- Use flooding if destination address is not present in the table

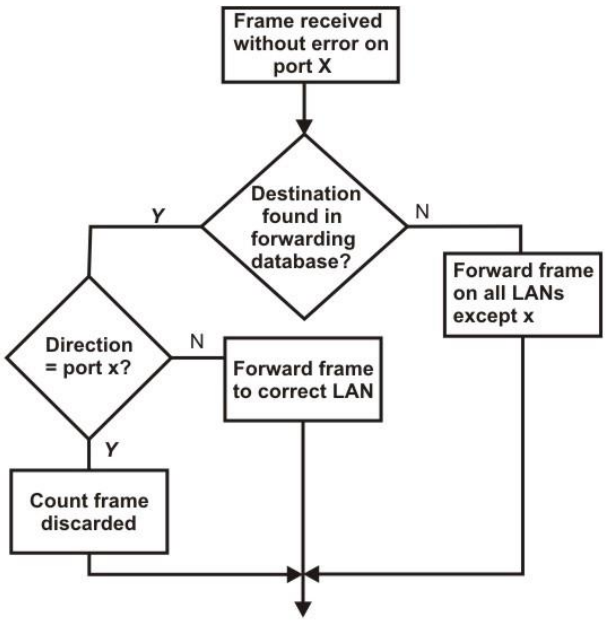


Figure: Bridge Forwarding

Bridge learning:

At the time installation of the transparent bridge, the database in the form of table is empty. As a packet is encountered the bridge is checks its source address and build up a table by associating a source address with a port address to which it is connected. The flowchart explains the learning process. The table building up operation is illustrated in Fig. below.

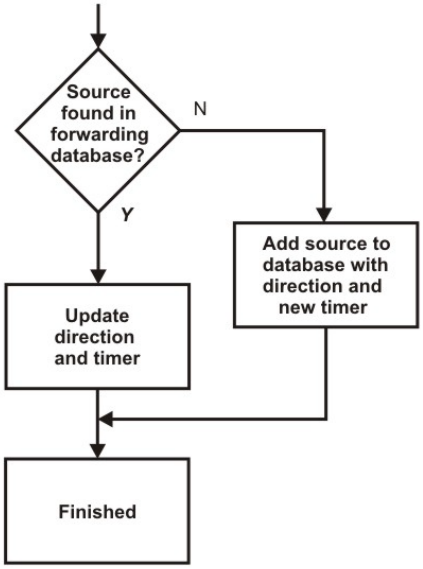


Figure : Bridge learning

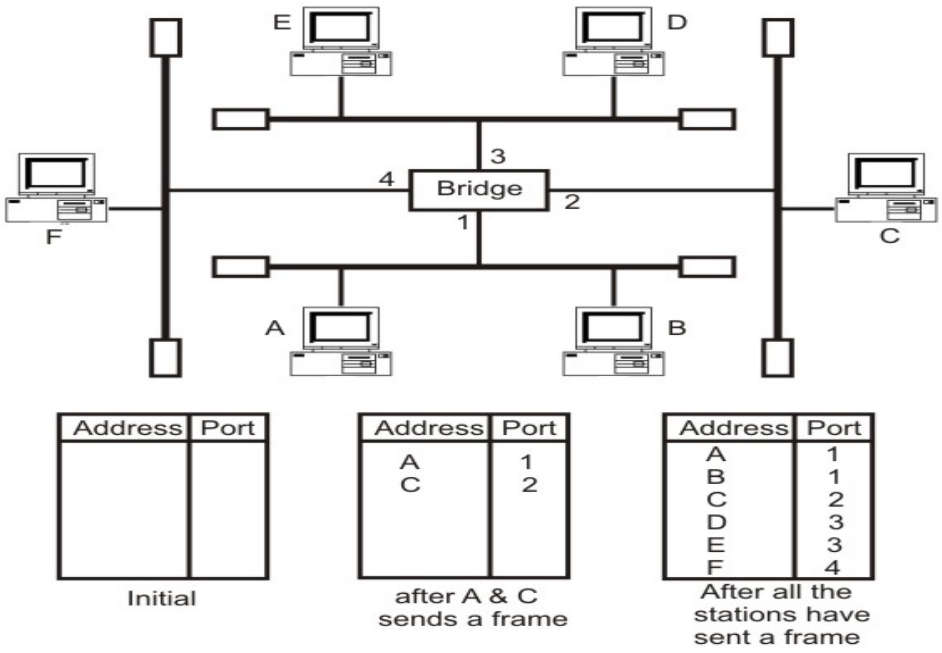


Figure : Creation of a bridge-forwarding table

Loop Problem

Forwarding and learning processes work without any problem as long as there is no redundant bridge in the system. On the other hand, redundancy is desirable from the viewpoint of reliability, so that the function of a failed bridge is taken over by a redundant bridge. The existence of redundant bridges creates the so-called *loop problem* as illustrated with the help of Fig. below. Assuming that after initialization tables in both the bridges are empty let us consider the following steps:

Step 1. Station-A sends a frame to Station-B. Both the bridges forward the frame to LAN Y and update the table with the source address of A.

Step 2. Now there are two copies of the frame on LAN-Y. The copy sent by Bridge-a is received by Bridge-b and vice versa. As both the bridges have no information about Station B, both will forward the frames to LAN-X.

Step 3. Again both the bridges will forward the frames to LAN-Y because of the lack of information of the Station B in their database and again Step-2 will be repeated, and so on.

So, the frame will continue to loop around the two LANs indefinitely.

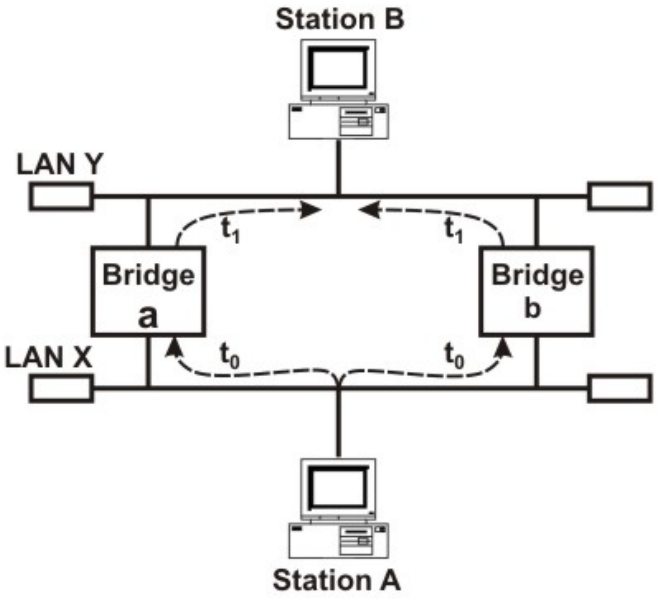


Figure : Loop problem in a network using bridges

Spanning Tree

As redundancy creates loop problem in the system, it is very undesirable. To prevent loop problem and proper working of the forwarding and learning processes, there must be only one path between any pair of bridges and LANs between any two segments in the entire bridged LAN. The IEEE specification requires that the bridges use a special topology. Such a topology is known as *spanning tree* (a graph where there is no loop) topology. The methodology for setting up a spanning tree is known as spanning tree algorithm, which creates a tree out of a graph. Without changing the physical topology, a logical topology is created that overlay on the physical one by using the following steps:

Select a bridge as *Root-bridge*, which has the smallest ID. Select *Root ports* for all the bridges, except for the root bridge, which has least-cost path (say minimum number of hops) to the root bridge. Choose a *Designated bridge*, which has least-cost path to the Root-bridge, in each LAN.

Select a port as *Designated port* that gives least-cost path from the Designated bridge to the Root bridge.

Mark the designated port and the root ports as *Forwarding ports* and the remaining ones as *Blocking ports.*

The spanning tree of a network of bridges is shown in Fig.below. The forwarding ports are shown as solid lines, whereas the blocked ports are shown as dotted lines.

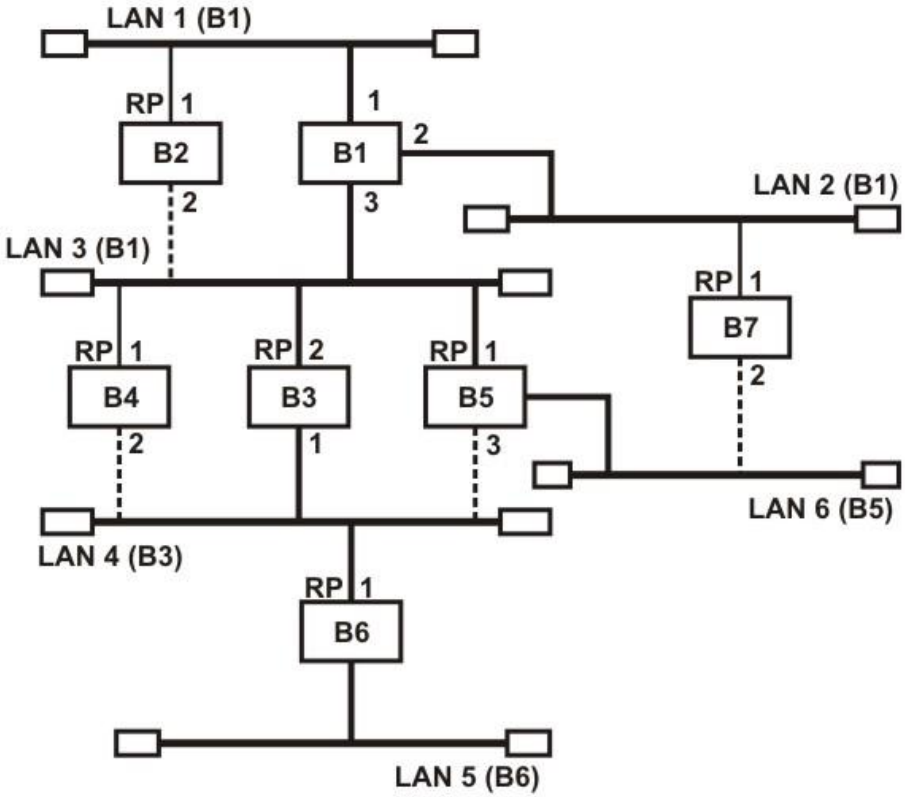


Figure : Spanning tree of a network of bridges

Source Routing Bridges

The second approach, known as *source routing,* where the routing operation is performed by the source host and the frame specifies which route the frame is to follow. A host can discover a route by sending a *discovery frame***,** which spreads through the entire network using all possible paths to the destination. Each frame gradually gathers addresses as it goes. The destination responds to each frame and the source host chooses an appropriate route from these responses. For example, a route with minimum hop-count can be chosen. Whereas transparent bridges do not modify a frame, a source routing bridge adds a routing information field to the frame. Source routing approach provides a shortest path at the cost of the proliferation of discovery frames, which can put a serious extra burden on the network. Figure below shows the frame format of a source routing bridge.

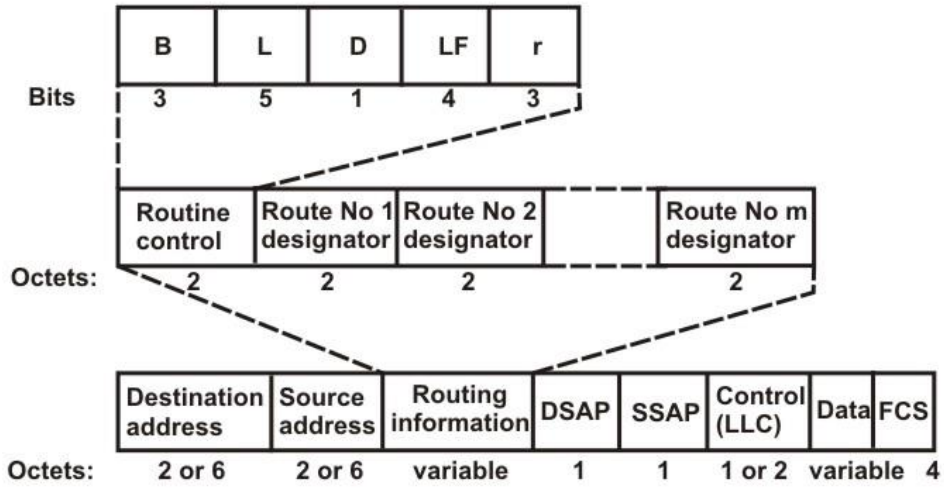


Figure : Source routing frame

the advantages of bridge.

- It *increases* total bandwidth of the network. For eg, while a single Ethernet segment can carry only 10 Mbps of total traffic, an Ethernet bridge can carry as much as 10*n* Mbps, where *n* is the number of ports on the bridge.
- Another advantage of bridge is separation of the collision domain as few stations contend for access to the medium. Thus the probability of collision is reduced.
- The networks can be connected without the end hosts having to run any additional protocols.

the limitations of a bridge.

- It is not realistic to connect more than a few LANs by means of bridges. Broadcast does not scale well, i.e., extended LANs do not scale.
- Bridges can support only networks that have exactly the same format for addresses. Bridges can be used to connect Ethernets to Ethernets, 802.5 to 802.5, and Ethernets to 802.5 rings. However, it cannot be used to connect ATM networks.

2.15 Switches

A switch is essentially a fast bridge having additional sophistication that allows faster processing of frames. Some of important functionalities are:

Ports are provided with buffer
Switch maintains a directory: #address - port#
Each frame is forwarded after examining the #address and forwarded to the proper port#
Three possible forwarding approaches: Cut-through, Collision-free and Fully-buffered as briefly explained below.

Cut-through: A switch forwards a frame immediately after receiving the destination address. As a consequence, the switch forwards the frame without collision and error detection.

Collision-free: In this case, the switch forwards the frame after receiving 64 bytes, which allows detection of collision. However, error detection is not possible because switch is yet to receive the entire frame.

Fully buffered: In this case, the switch forwards the frame only after receiving the entire frame. So, the switch can detect both collision and error free frames are forwarded.

Comparison between a switch and a hub

Although a hub and a switch apparently look similar, they have significant differences. As shown in Fig. 6.1.12, both can be used to realize physical star topology, the hubs works like a logical bus, because the same signal is repeated on all the ports. On the other hand, a switch functions like a logical star with the possibility of the communication of separate signals between any pair of port lines. As a consequence, all the ports of a hub belong to the same collision domain, and in case of a switch each port operates on separate collision domain. Moreover, in case of a hub, the bandwidth is shared by all the stations connected to all the ports. On the other hand, in case of a switch, each port has dedicated bandwidth. Therefore, switches can be used to increase the bandwidth of a hub-based network by replacing the hubs by switches.

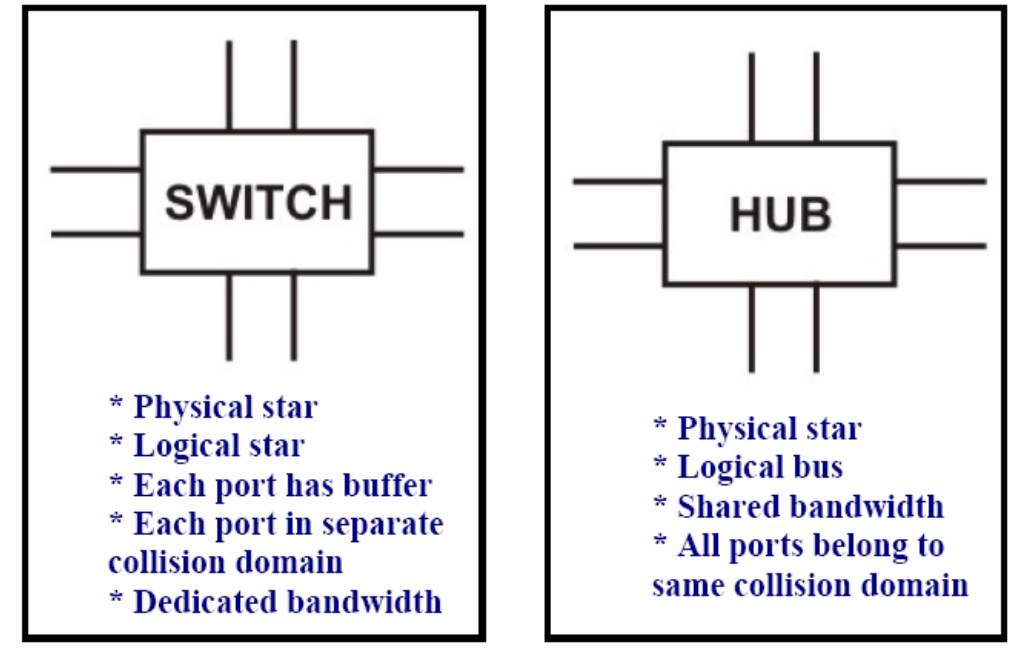


Figure : Difference between a switch and a bridge

Difference between LAN and extended LAN (*bridge****).***

- If a bridge becomes congested, it drops frames, whereas Ethernet does not drop frames
- The latency between any pair of hosts on an extended LAN becomes both larger and more highly variable than in Ethernet
- Frame order is not shuffled in ethernet, whereas reordering is possible in extended LAN.

3.1 INTER NETWORKING BASICS:

For transmission of data beyond a local area, communication is typically achieved by transmitting data from source to destination through a network of intermediate switching nodes; this switched network design is typically used to implement LANs as well. The switching nodes are not concerned with the content of the data; rather, their purpose is to provide a switching facility that will move the data from node to node until they reach their destination. Figure 3.1 illustrates a simple network. The devices attached to the network may be referred to as *stations.* The stations may be computers, terminals, telephones, or other communicating devices. We refer to the switching devices whose purpose is to provide communication as *nodes.* Nodes are connected to one another in some topology by transmission links. Each station attaches to a node, and the collection of nodes is referred to as a *communications network.*

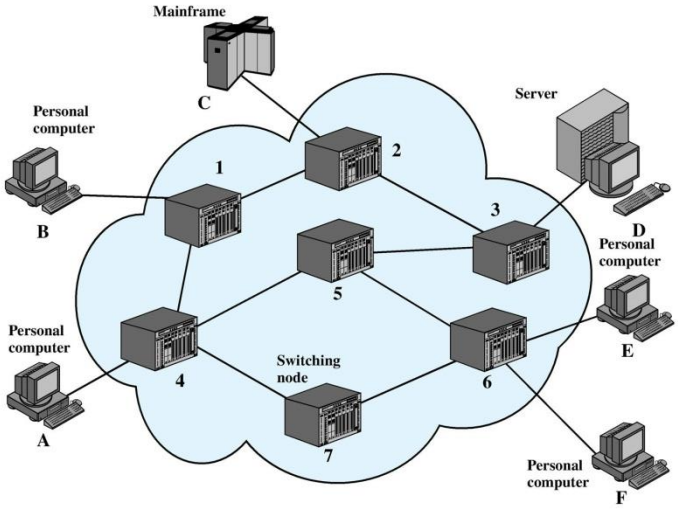


Figure 3.1: a simple network

Circuit Switching vs. Packet Switching:

One fundamental way of differentiating networking technologies is on the basis of the method they use to determine the path between devices over which information will flow. In highly simplified terms, there are two approaches: either a path can be set up between the devices in advance, or the data can be sent as individual data elements over a variable path.

Circuit Switching In this networking method, a connection called a *circuit* is set up between two devices, which is used for the whole communication. Information about the nature of the circuit is maintained by the network. The circuit may either be a fixed one that is always present, or it may be a circuit that is created on an as-needed basis. Even if many potential paths through intermediate devices may exist between the two devices communicating, only one path will be used for any given dialog. Thus a dedicated physical link exists between a source and a destination and data are sent as stream of bits. This is illustrated in Figure 3.2

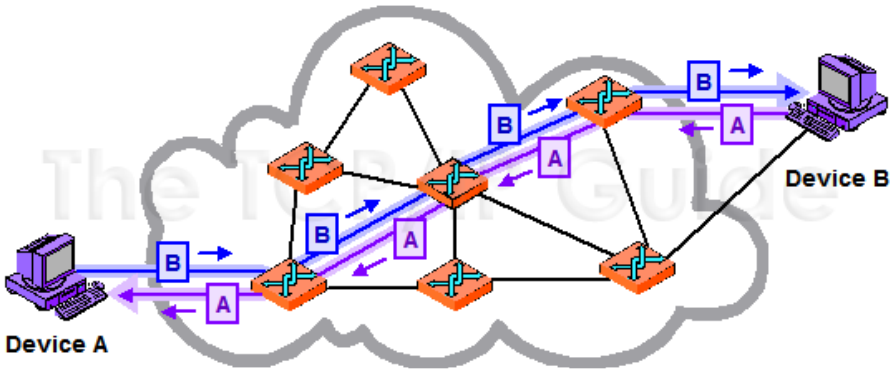


Figure3.2: Circuit Switching

The classic example of a circuit-switched network is the telephone system. When we call someone and they answer, we establish a circuit connection and can pass data between us, in a steady stream if desired. That circuit functions the same way regardless of how many intermediate devices are used to carry your voice. we use it for as long as we need it, and then terminate the circuit. The next time we call, you get a new circuit, which may (probably will) use different hardware than the first circuit did, depending on what's available at that time in the network

Packet Switching

In this network type, no specific path is used for data transfer. Instead, the data is chopped up into small pieces called *packets* and sent over the network. The packets can be routed, combined or fragmented, as required to get them to their eventual destination. On the receiving end, the process is reversed—the data is read from the packets and re-assembled into the form of the original data. A packet-switched network is more analogous to the postal system than it is to the telephone system (though the comparison isn't perfect.) An example is shown in Figure 3.3.

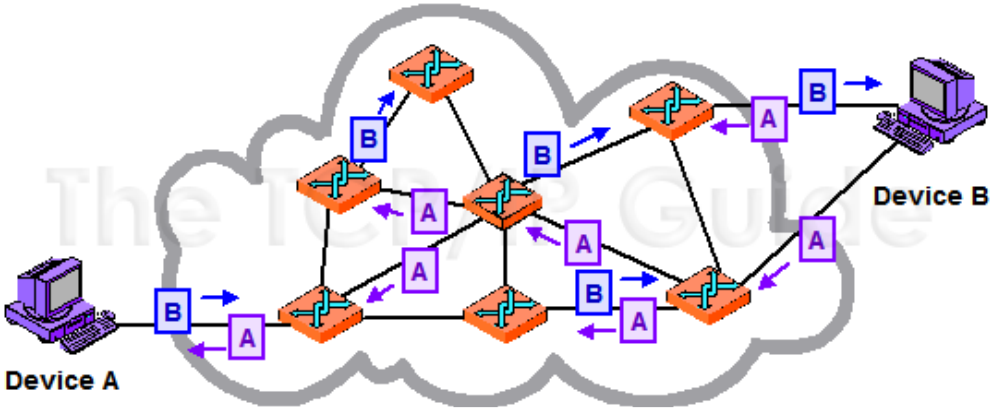


Figure3.3: Packet Switching

In a packet-switched network, no circuit is set up prior to sending data between devices. Blocks of data, even from the same file or communication, may take any number of paths as it journeys from one device to another. **Packet switching** is a communications method in which packets (discrete blocks of data) are routed between nodes over data links shared with other traffic. In each network node, packets are queued or buffered, resulting in variable delay. Packet switching is used to optimize the use of the channel capacity available in digital telecommunication networks such as computer networks, to minimize the transmission latency, and to increase robustness of communication. The most well-known use of packet switching is the Internet and local area networks. The Internet uses the Internet protocol suite over a variety of data link layer protocols. For example, Ethernet and frame relay are very common. Newer mobile phone technologies also use packet switching.

The following paradigms are available for packet switching:

o **Virtual Circuit Switching**: A connection is setup before the packets are transmitted. All the packets follow the same path. The connection could either be permanent (manually setup by network administrator) or switched (dynamically setup through control signals).

o **Datagram Switching**: No connection is setup each packet is forwarded independent of the other. An initial setup phase is used to set up a route between the intermediate nodes for all the packets passed during the session between the two end nodes. In each intermediate node, an entry is registered in a table to indicate the route for the connection that has been set up. Thus, packets passed through this route, can have short headers, containing only a virtual circuit identifier (VCI), and not their destination. Each intermediate node passes the packets according to the information that was stored in it, in the setup phase.

In this way, packets arrive at the destination in the correct sequence, and it is guaranteed that essentially there will not be errors. This approach is slower than Circuit Switching, since different virtual circuits may compete over the same resources, and an initial setup phase is needed to initiate the circuit. As in Circuit Switching, if an intermediate node fails, all virtual circuits that pass through it are lost.

The most common forms of Virtual Circuit networks are X.25 and Frame Relay, which are commonly used for public data networks (PDN). X.25 is a notable use of packet switching in that, despite being based on packet switching methods, it provided virtual circuits to the user. These virtual circuits carry variable-length packets. X.25 was used to provide the first international and commercial packet switching network, the International Packet Switched Service (IPSS). Asynchronous Transfer Mode (ATM) also is a virtual circuit technology, which uses fixed-length cell relay connection oriented packet switching.

Datagram packet switching is also called connectionless networking because no connections are established. Technologies such as Multiprotocol Label Switching (MPLS) and the Resource Reservation Protocol (RSVP) create virtual circuits on top of datagram networks. Virtual circuits are especially useful in building robust failover mechanisms and allocating bandwidth for delay sensitive applications. MPLS and its predecessors, as well as ATM, have been called "fast packet" technologies. MPLS, indeed, has been called "ATM without cells". Modern routers, however, do not require these technologies to be able to forward variable-length packets at multi-gigabit speeds across the network. This approach uses a different, more dynamic scheme, to determine the route through the network links. Each packet is treated as an independent entity, and its header contains full information about the destination of the packet. The intermediate nodes examine the header of the packet, and decide to which node to send the packet so that it will reach its destination. In the decision two factors are taken into account:

• The shortest way to pass the packet to its destination - protocols such as RIP/OSPF issued to determine the shortest path to the destination.

• Finding a free node to pass the packet to - in this way, bottle necks are eliminated, since packets can reach the destination in alternate routes.

Thus, in this method, the packets don't follow a pre-established route, and the intermediate nodes (the routers) don't have pre-defined knowledge of the routes that the packets should be passed through. Packets can follow different routes to the destination, and delivery is not guaranteed (although packets usually do follow the same route, and are reliably sent). Due to the nature of this method, the packets can reach the destination in a different order than they were sent, thus they must be sorted at the destination to form the original message. This approach is time consuming since every router has to decide where to send each packet. The main implementation of Datagram Switching network is the Internet which uses the IP network protocol.

Switching at the network layer in the Internet is done using the datagram approach to packet switching. The communication at the network layer in the Internet is connectionless. The Figure3.4 shows how datagram approach can be used for delivering four packets from station A to station X. It could be viewed that though all the packets belong to the same message they can take different paths to reach their destination.

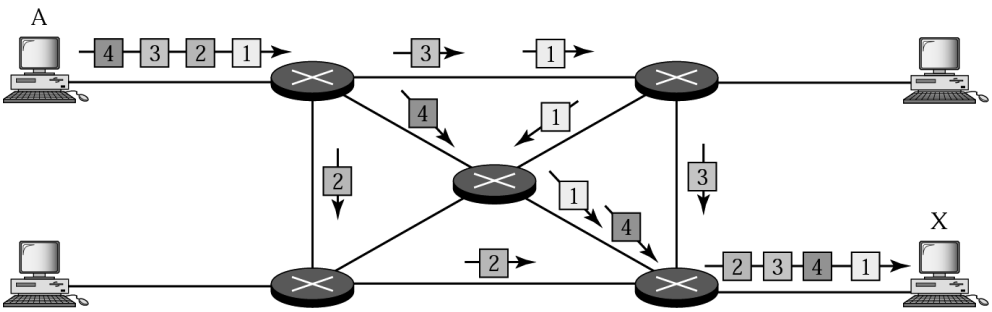


Figure 3.4: Datagram Approach

The ability to have many devices communicate simultaneously without dedicated data paths is one reason why packet switching is becoming predominant today. However, there are some disadvantages of packet switching compared to circuit switching. One is that since all data does not take the same, predictable path between devices, it is possible that some pieces of data may get lost in transit, or show up in the incorrect order. In some situations this does not matter, while in others it is very important indeed.

While the theoretical difference between circuit and packet switching is pretty clear-cut, understanding how they are used is a bit more complicated. One of the major issues is that in modern networks, they are often combined. For example, suppose you connect to the Internet using a dial-up modem. You will be using IP datagrams (packets) to carry higher-layer data, but it will be over the circuit-switched telephone network. Yet the data may be sent over the telephone system in digital packetized form. So in some ways, both circuit switching and packet switching are being used concurrently.

IP:

The **Internet Protocol** (**IP**) is a protocol used for communicating data across packet switched internetwork using the Internet Protocol Suite, also referred to as TCP/IP.IP is the primary protocol in the Internet Layer of the Internet Protocol Suite and has the task of delivering distinguished protocol datagrams (packets) from the source host to the destination host solely based on their addresses. For this purpose the Internet Protocol defines addressing methods and structures for datagram encapsulation. The first major version of addressing structure, now referred to as Internet Protocol Version 4 (IPv4) is still the dominant protocol of the Internet, although the successor, Internet Protocol Version 6 (IPv6) is being deployed actively worldwide

IP encapsulation

Data from an upper layer protocol is encapsulated as packets/datagrams (the terms are synonymous in IP). Circuit setup is not needed before a host may send packets to another host that it has previously not communicated with (a characteristic of packet- switched networks), thus IP is a connectionless protocol. This is in contrast to public switched telephone networks that require the setup of a circuit for each phone call (*connection-oriented* protocol).

Services provided by IP

Because of the abstraction provided by encapsulation, IP can be used over a heterogeneous network, i.e., a network connecting computers may consist of a combination of Ethernet, ATM, FDDI, Wi-Fi, token ring, or others. Each link layer implementation may have its own method of addressing (or possibly the complete lack of it), with a corresponding need to resolve IP addresses to data link addresses. This address resolution is handled by the Address Resolution Protocol (ARP) for IPv4 and Neighbor Discovery Protocol (NDP) for IPv6.

Reliability

The design principles of the Internet protocols assume that the network infrastructure is inherently unreliable at any single network element or transmission medium and that it is dynamic in terms of availability of links and nodes. No central monitoring or performance measurement facility exists that tracks or maintains the state of the network. For the benefit of reducing network complexity, the intelligence in the network is purposely mostly located in the end nodes of each data transmission, cf. end-to-end principle. Routers in the transmission path simply forward packets to next known local gateway matching the routing prefix for the destination address.

As a consequence of this design, the Internet Protocol only provides best effort delivery and its service can also be characterized as *unreliable*. In network architectural language it is a *connection-less* protocol, in contrast to so-called connection-oriented modes of transmission. The lack of reliability allows any of the following fault events to occur:

- data corruption
- lost data packets
- duplicate arrival
- out-of-order packet delivery; meaning, if packet 'A' is sent before packet 'B', packet 'B' may arrive before packet 'A'. Since routing is dynamic and there is no memory in the network about the path of prior packets, it is possible that the first packet sent takes a longer path to its destination.

The only assistance that the Internet Protocol provides in Version 4 (IPv4) is to ensure that the IP packet header is error-free through computation of a checksum at the routing nodes. This has the side-effect of discarding packets with bad headers on the spot. In this case no notification is required to be sent to either end node, although a facility exists in the Internet Control Message Protocol (ICMP) to do so.

IPv6, on the other hand, has abandoned the use of IP header checksums for the benefit of rapid forwarding through routing elements in the network.

The resolution or correction of any of these reliability issues is the responsibility of an upper layer protocol. For example, to ensure in-order delivery the upper layer may have to cache data until it can be passed to the application.

In addition to issues of reliability, this dynamic nature and the diversity of the Internet and its components provide no guarantee that any particular path is actually capable of, or suitable for performing the data transmission requested, even if the path is available and reliable. One of the technical constraints is the size of data packets allowed on a given link. An application must assure that it uses proper transmission characteristics. Some of this responsibility lies also in the upper layer protocols between application and IP. Facilities exist to examine the maximum transmission unit (MTU) size of the local link, as well as for the entire projected path to the destination when using IPv6. The IPv4 internetworking layer has the capability to automatically fragment the original datagram into smaller units for transmission. In this case, IP does provide re-ordering of fragments delivered out-of-order.
Transmission Control Protocol (TCP) is an example of a protocol that will adjust its segment size to be smaller than the MTU. User Datagram Protocol (UDP) and Internet Control Message Protocol (ICMP) disregard MTU size thereby forcing IP to fragment oversized datagrams.

IP Packet format

The IP datagram, like most packets, consists of a header followed by a number of bytes of data. The format of the header is shown in Figure 3.5.

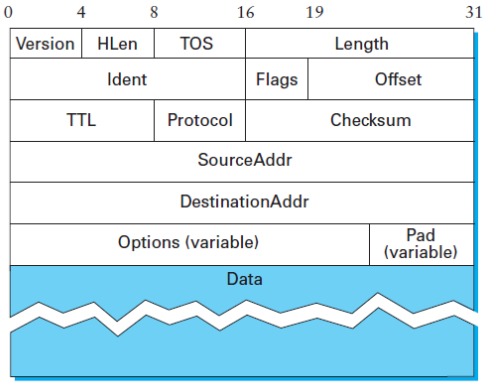


Fig. 3.5:IPv4 Packet Header

Version: The **Version field** specifies the version of IP. The current version of IP is 4, and it is sometimes called IPv4.
HLen : The next field, **HLen,** specifies the length of the header in 32-bit words. When there are no options, which is most of the time, the header is 5 words (20 bytes) long.
TOS : The 8-bit **TOS (**type of service) field has had a number of different definitions over the years, but its basic function is to allow packets to be treated differently based on application needs. For example, the TOS value might determine whether or not a packet should be placed in a special queue that receives low delay.
Length : The next 16 bits of the header contain the **Length** of the datagram, including the header. Unlike the HLen field, the Length field counts bytes rather than words. Thus, the maximum size of an IP datagram is 65,535 bytes. The physical network over which IP is running, however, may not support such long packets. For this reason, IP supports a **fragmentation and reassembly process**. The second word of the header contains information about fragmentation, and the details of its use are presented under "Fragmentation and Reassembly" below.
TTL : Moving on to the third word of the header, the next byte is the **TTL (time to live)** field. Its name reflects its historical meaning rather than the way it is commonly used today. The intent of the field is to catch packets that have been going around in routing loops and discard them, rather than let them consume resources indefinitely. Originally, TTL was set to a specific number of seconds that the packet would be allowed to live, and routers along the path would decrement this field until it reached 0. However, since it was rare for a packet to sit for as long as 1 second in a router, and routers did not all have access to a common clock, most routers just decremented the TTL by 1 as they forwarded the packet. Thus, it became more of a hop count than a timer, which is still a perfectly good way to catch packets that are stuck in routing loops. One subtlety is in the initial setting of this field by the sending host: Set it too high and packets could circulate rather a lot before getting dropped; set it too low and they may not reach their destination. The value 64 is the current default.
Ident: It allows the destination host to determine which datagram a newly arrived fragment belongs to. All the fragment of a datagram contain the same identification value
Flags : DF – Don't fragment, MF- More fragment
Offset : Max 8192 fragment per datagram
Protocol : This field is simply a de-multiplexing key that identifies the higher-level protocol to which this IP packet should be passed. There are values defined for TCP (6), UDP (17), and many other protocols that may sit above IP in the protocol graph.
Checksum : This field is calculated by considering the entire IP header as a sequence of 16-bit words, adding them up using ones complement arithmetic, and taking the ones complement of the result. Thus, if any bit in the header is corrupted in transit, the checksum will not contain the correct value upon receipt of the packet. Since a corrupted header may contain an error in the destination address—and, as a result, may have been misdelivered—it makes sense to discard any packet that fails the checksum. It should be noted that this type of checksum does not have the same strong error detection properties as a CRC, but it is much easier to calculate in software.
The last two required fields in the header are the **SourceAddr and the DestinationAddr for the packet.** The latter is the key to datagram delivery: Every packet contains a full address for its intended destination so that forwarding decisions can be made at each router. The source address is required to allow recipients to decide if they want to accept the packet and to enable them to reply.
Finally, there may be a number of **options** at the end of the header. The presence or absence of options may be determined by examining the header length (HLen) field. While options are used fairly rarely, a complete IP implementation must handle them all.

Fragmentation and Reassembly
One of the problems of providing a uniform host-to-host service model over a heterogeneous collection of networks is that each network technology tends to have its own idea of how large a packet can be. For example, an Ethernet can accept packets up to 1500 bytes long, while FDDI packets may be 4500 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, or provide a means by which packets can be fragmented and reassembled when they are too big to go over a given network technology
The central idea here is that every network type has a ***maximum transmission unit* (MTU),** which is the largest IP datagram that it can carry in a frame. When a host sends an IP datagram, therefore, it can choose any size that it wants. A reasonable choice is the MTU of the network to which the host is directly attached. Then fragmentation will only be necessary if the path to the destination includes a network with a smaller MTU. Should the transport protocol that sits on top of IP give IP a packet larger than the local MTU, however, then the source host must

fragment it. 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.

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. Should all the fragments not arrive at the receiving host, the host gives up on the reassembly process and discards the fragments that did arrive. IP does not attempt to recover from missing fragments.

IP addressing

Perhaps the most complex aspects of IP are IP addressing and routing. Addressing refers to how end hosts become assigned IP addresses and how subnetworks of IP host addresses are divided and grouped together. IP routing is performed by all hosts, but most importantly by internetwork routers, which typically use either interior gateway protocols (IGPs) or external gateway protocols (EGPs) to help make IP datagram forwarding decisions across IP connected networks.

An IPv4 address is a 32-bit address that *uniquely* and *universally* defines the connection of a device (for example, a computer or a router) to the Internet.

An IP address is a 32 bit binary number usually represented as 4 decimal values, each representing 8 bits, in the range 0 to 255 (known as octets) separated by decimal points. This is known as "dotted decimal" notation. Figure 3.6 shows the **binary and dotted decimal** representation of the IP address ***128.11.3.31***.

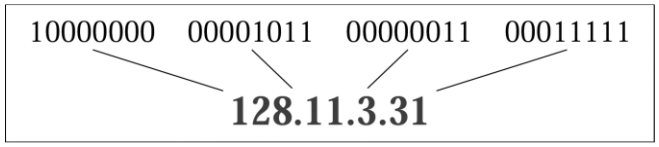


Figure 3.6: Binary and dotted decimal notation of IP address

There are three modes of addressing namely, Uni-cast, Multi-cast and Broad-cast addresses**.** ***Uni-Cast*** address is simply a **send-to one** addressing mode. ***Broad-Cast*** address is simply a **send-to all** addressing mode and cannot be used as source address. ***Multi-Cast*** is simply a **send-to group** addressing mode and can never be used as a source address.

Every IP address consists of two parts, one identifying the network and the other identifying the node. The Class of the address and the subnet mask determine which part belongs to the network address and which part belongs to the node address.

There are 5 different address classes namely A, B, C, D and E. It is possible to determine to which class a given IP address belong to by examining the most significant four bits of the IP address.

- **Class A** addresses begin with **0xxx**, or **1 to 127** decimal.
- **Class B** addresses begin with **10xx**, or **128 to 191** decimal.
- **Class C** addresses begin with **110x**, or **192 to 223** decimal.
- **Class D** addresses begin with **1110**, or **224 to 239** decimal.
- **Class E** addresses begin with **1111**, or **240 to 255** decimal.

The class of the IP address determines the default number of bits used for the network identification and host identification within the network. The netid and the hostid bytes for all the classes are shown in the Figure 3.7

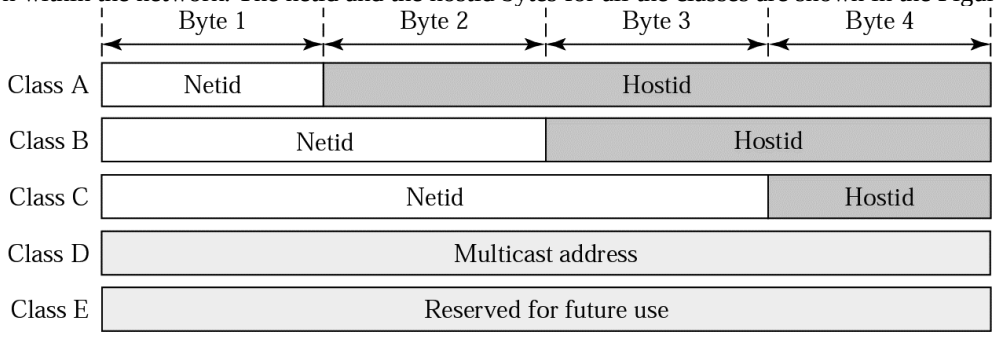


Figure 3.7: netid and hostid bits

Class A networks have 7 bits for the network part and 24 bits for the host part. There can be only 126 (0 and 127 are reserved) Class A networks. Each of them can accommodate 224 -2 hosts. Class A addresses were designed for large organizations with large number of hosts or routers attached to their network. Class B networks have 14 bits for the network part and16 bits for the host part. Class B networks can accommodate 65,534 hosts. Class B addresses were designed for midsize organizations that may have tens of thousands of hosts or routers attached to their networks. Class C networks have 8 bits for the network part and 21 bits for the host part. Class C networks can accommodate only 256 hosts. Class C addresses were designed for small organizations with a small number of hosts or routers attached to their network. There is just one block of Class D addresses, which is designed for multicasting. There is just one block of Class E addresses, which is designed for use as reserved addresses.

Subnetting

To filter packets for a particular network, a router uses a concept known as *masking*, which filters out the net id part (by ANDing with all 1's) by removing the host id part (by ANDing with all 0's). The net id part is then compared with the network address as shown in Figure3.8. All the hosts in a network must have the same network number. This property of IP addressing causes problem as the network grows. To overcome this problem, a concept known as *subnets* is used, which splits a network into several parts for internal use, but still acts like a single network to the outside world. To facilitate routing, a concept known as *subnet mask* is used. As shown in Figure3.9, a part of hostid is used as subnet address with a corresponding subnet mask. Subnetting reduces router table space by creating a three-level hierarchy; net id, subnet id followed by hosted.

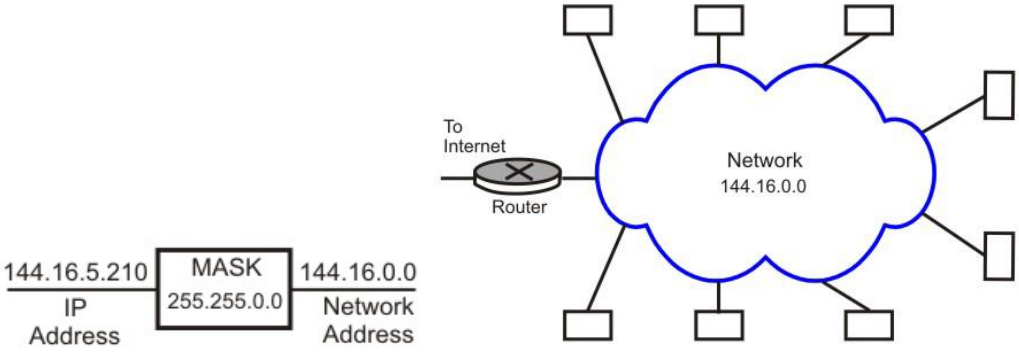


Figure 3.8: Masking with the help of router

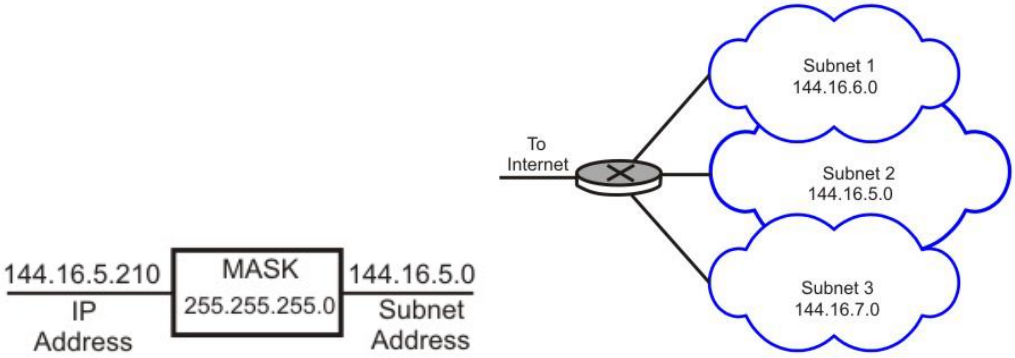


Figure 3.9: Subnet masking with the help of router

Network Address Translation (NAT)

With the increasing number of internet users requiring an unique IP address for each host, there is an acute shortage of IP addresses (until everybody moves to IPV6). The *Network Address Translation* (NAT) approach is a quick interim solution to this problem. NAT allows a large set of IP addresses to be used in an internal (private) network and a handful of addresses to be used for the external internet. The internet authorities has set aside three sets of addresses to be used as private addresses as shown in Table 3.1. It may be noted that these addresses can be reused within different internal networks simultaneously, which in effect has helped to increase the lifespan of the IPV4. However, to make use of the concept, it is necessary to have a router to perform the operation of address translation between the private network and the internet. As shown in Figure3.10, the NAT router maintains a table with a pair of entries for private and internet address. The source address of all outgoing packets passing through the NAT router gets replaced by an internet address based on table look up. Similarly, the destination address of all incoming packets passing through the NAT router gets replaced by the corresponding private address, as shown in the figure. The NAT can use a pool of internet addresses to have internet access by a limited number of stations of the private network at a time.

Table 3.1 Addresses for Private Network

Range of addresses	Total number
10.0.0.0 to 10.255.255.255	224
172.16.0.0 to 172.31.255.255	220
192.168.0.0 to 192.168.255.255	216

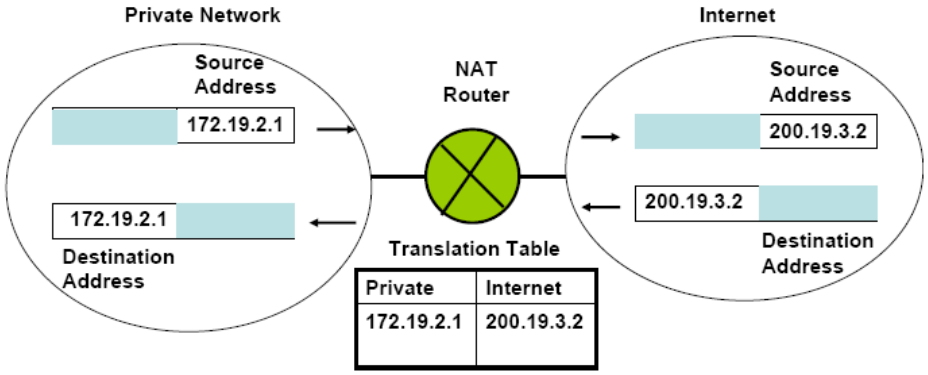


Figure 3.10: NAT Address translation

ADDRESS MAPPING

An internet is made of a combination of physical networks connected by internetworking devices such as routers. A packet starting from a source host may pass through several different physical networks before finally reaching the destination host. The hosts and routers are recognized at the network level by their logical (IP) addresses. However, packets pass through physical networks to reach these hosts and routers. At the physical level, the hosts and routers are recognized by their physical addresses.

A physical address is a local address. Its jurisdiction is a local network. It must be unique locally, but is not necessarily unique universally. It is called a *physical* address because it is usually (but not always) implemented in hardware. An example of a physical address is the 48-bit MAC address in the Ethernet protocol, which is imprinted on the NIC installed in the host or router. The physical address and the logical address are two different identifiers. We need both because a physical network such as Ethernet can have two different protocols at the network layer such as IP and IPX (Novell) at the same time. Likewise, a packet at a network layer such as IP may pass through different physical networks such as Ethernet and LocalTalk (Apple). This means that delivery of a packet to a host or a router requires two levels of addressing: logical and physical.

We need to be able to **map a logical address to its corresponding physical address and vice versa.** These can be done by using either **static or dynamic mapping**.

Static mapping involves in the creation of a table that associates a logical address with a physical address. This table is stored in each machine on the network. Each machine that knows, for example, the IP address of another

machine but not its physical address can look it up in the table. This has some limitations because physical addresses may change in the following ways:
1. A machine could change its NIC, resulting in a new physical address.
2. In some LANs, such as LocalTalk, the physical address changes every time the computer is turned on.
3. A mobile computer can move from one physical network to another, resulting in a change in its physical address.
To implement these changes, a static mapping table must be updated periodically. This overhead could affect network performance.
In dynamic mapping each time a machine knows one of the two addresses (logical or physical), it can use a protocol to find the other one.

3.3.1 ARP- Address Resolution Protocol

Mapping Logical to Physical Address: ARP
Anytime a host or a router has an IP datagram to send to another host or router, it has the logical (IP) address of the receiver. The logical (IP) address is obtained from the DNS if the sender is the host or it is found in a routing table if the sender is a router. But the IP datagram must be encapsulated in a frame to be able to pass through the physical network. This means that the sender needs the physical address of the receiver.
The host or the router sends an ARP query packet. The packet includes the physical and IP addresses of the sender and the IP address of the receiver. Because the sender does not know the physical address of the receiver, the query is broadcast over the network (see Figure 3.11).
Every host or router on the network receives and processes the ARP query packet, but only the intended recipient recognizes its IP address and sends back an ARP response packet. The response packet contains the recipient's IP and physical addresses. The packet is unicast directly to the inquirer by using the physical address received in the query packet.
In Figure 3.11 a, the system on the left (A) has a packet that needs to be delivered to another system (B) with IP address 141.23.56.23. System A needs to pass the packet to its data link layer for the actual delivery, but it does not know the physical address of the recipient. It uses the services of ARP by asking the ARP protocol to send a broadcast ARP request packet to ask for the physical address of a system with an IP address of 141.23.56.23.
This packet is received by every system on the physical network, but only system B will answer it, as shown in Figure 3.11 b. System B sends an ARP reply packet that includes its physical address. Now system A can send all the packets it has for this destination by using the physical address it received.

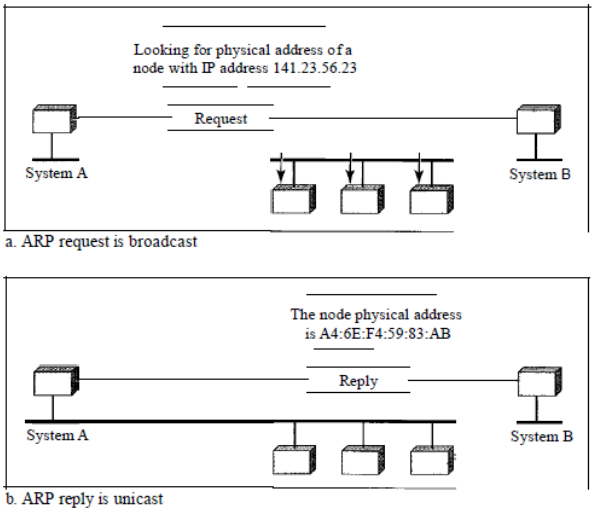


Figure 3.11: *ARP operation*

Cache Memory

Using ARP is inefficient if system A needs to broadcast an ARP request for each IP packet it needs to send to system B. It could have broadcast the IP packet itself. ARP can be useful if the ARP reply is cached (kept in cache memory for a while) because a system normally sends several packets to the same destination. A system that receives an ARP reply stores the mapping in the cache memory and keeps it for 20 to 30 minutes unless the space in the cache is exhausted. Before sending an ARP request, the system first checks its cache to see if it can find the mapping.

Packet Format

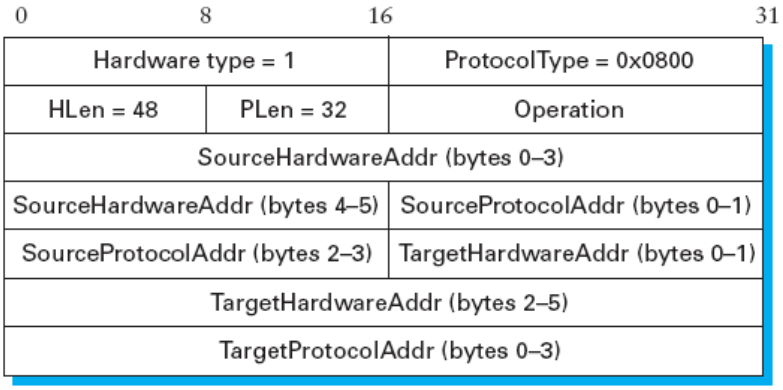


Figure 3.12:Format of an ARP packet for mapping IP address into Ethernet addresses.

The fields are as follows:

• **Hardware type :** This is a 16-bit field defining the type of the network on which ARP is running. Each LAN has been assigned an integer based on its type. For example, Ethernet is given type 1. ARP can be used on any physical network.

• **Protocol type :** This is a 16-bit field defining the protocol. For example, the value of this field for the IPv4 protocol is 080016, ARP can be used with any higher-level protocol.

• **HLen(" hardware" address length) and PLen(" protocol" address length):** These fields specify the length of the link-layer address and higher-layer protocol address respectively.

• **Operation**. This field specifies whether this is a request or a response.

• **The Source and Target hardware(Ethernet) and protocol(IP) addresses**.

Encapsulation

An ARP packet is encapsulated directly into a data link frame. For example, an ARP packet can be encapsulated in an Ethernet frame. Note that the type field indicates that the data carried by the frame are an ARP packet.

Operation

Let us see how ARP functions on a typical internet. First we describe the steps involved.

Then we discuss the four cases in which a host or router needs to use ARP. These are the steps involved in an ARP process:

1. The sender knows the IP address of the target. We will see how the sender obtains this shortly.
2. IP asks ARP to create an ARP request message, filling in the sender physical address, the sender IP address, and the target IP address. The target physical address field is filled with Os.
3. The message is passed to the data link layer where it is encapsulated in a frame by using the physical address of the sender as the source address and the physical broadcast address as the destination address.
4. Every host or router receives the frame. Because the frame contains a broadcast destination address, all stations remove the message and pass it to ARP. All machines except the one targeted drop the packet. The target machine recognizes its IP address.
5. The target machine replies with an ARP reply message that contains its physical address. The message is unicast.
6. The sender receives the reply message. It now knows the physical address of the target machine.
7. The IP datagram, which carries data for the target machine, is now encapsulated in a frame and is unicast to the destination.

ProxyARP

A technique called *proxy* ARP is used to create a subnetting effect. A proxy ARP is an ARP that acts on behalf of a set of hosts. Whenever a router running a proxy ARP receives an ARP request looking for the IP address of one of these hosts, the router sends an ARP reply announcing its own hardware (physical) address. After the router receives the actual IP packet, it sends the packet to the appropriate host or router.

Mapping Physical to Logical Address: RARP, BOOTP, and DHCP:

There are occasions in which a host knows its physical address, but needs to know its logical address. This may happen in two cases:

1. A diskless station is just booted. The station can find its physical address by checking its interface, but it does not know its IP address.
2. An organization does not have enough IP addresses to assign to each station; it needs to assign IP addresses on demand. The station can send its physical address and ask for a short time lease.

RARP

Reverse Address Resolution Protocol (RARP) finds the logical address for a machine that knows only its physical address. Each host or router is assigned one or more logical (IP) addresses, which are unique and independent of the physical (hardware) address of the machine. To create an IP datagram, a host or a router needs to know its own IP address or addresses. The IP address of a machine is usually read from its configuration file stored on a disk file. However, a diskless machine is usually booted from ROM, which has minimum booting information. The ROM is installed by the manufacturer. It cannot include the IP address because the IP addresses on a network are assigned by the network administrator. The machine can get its physical address (by reading its NIC, for example), which is unique locally. It can then use the physical address to get the logical address by using the RARP protocol. A RARP request is created and broadcast on the local network. Another machine on the local network that knows all the IP addresses will respond with a RARP reply. The requesting machine must be running a RARP client program; the responding machine must be running a RARP server program.

There is a serious problem with RARP: Broadcasting is done at the data link layer. The physical broadcast address, all is in the case of Ethernet, does not pass the boundaries of a network. This means that if an administrator has several networks or several subnets, it needs to assign a RARP server for each network or subnet. This is the reason that RARP is almost obsolete. Protocols like BOOTP and DHCP, are replacing RARP.

BOOTP

The Bootstrap Protocol (BOOTP) is a client/server protocol designed to provide physical address to logical address mapping. BOOTP is an application layer protocol. The administrator may put the client and the server on the same network or on different networks .BOOTP messages are encapsulated in a UDP packet, and the UDP packet itself is encapsulated in an IP packet.

DHCP- Dynamic Host Configuration Protocol

DHCP dynamically assigns IP addresses to hosts. That is DHCP allows addresses to be “leased” for some period of time. Once the lease expires, the server is free to return that address to its pool. DHCP relies on the existence of a DHCP server that is responsible for providing configuration information to hosts. Since the goal of DHCP is to minimize the amount of manual configuration required for a host to function, it would rather defeat the purpose if each host had to be configured with the address of a DHCP server. Thus, the first problem faced by DHCP is that of server discovery.

To contact a DHCP server, a newly booted or attached host sends a DHCPDISCOVER message to a special IP address (255.255.255.255) that is an IP broadcast address. This means it will be received by all hosts and routers on that network. (Routers do not forward such packets onto other networks, preventing broadcast to the entire Internet.) In the simplest case, one of these nodes is the DHCP server for the network. The server would then reply to the host that generated the discovery message (all the other nodes would ignore it). However, it is not really desirable to require one DHCP server on every network because this still creates a potentially large number of servers that need to be correctly and consistently configured.

Thus, DHCP uses the concept of a *relay agent*. There is at least one relay agent on each network, and it is configured with just one piece of information: 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. The process of relaying a message from a host to a remote DHCP server is shown in Figure 3.13.

Depending on implementation, the DHCP server may have three methods of allocating IP-addresses:
 dynamic allocation: A network administrator assigns a range of IP addresses to DHCP, and each client computer on the LAN has its IP software configured to request an IP address from the DHCP server during network initialization. The request-and-grant process uses a lease concept with a controllable time period, allowing the DHCP server to reclaim (and then reallocate) IP addresses that are not renewed (dynamic re-use of IP addresses).
 automatic allocation: The DHCP server permanently assigns a free IP address to a requesting client from the range defined by the administrator. This is like dynamic allocation, but the DHCP server keeps a table of past IP address assignments, so that it can preferentially assign to a client the same IP address that the client previously had.
 static allocation: The DHCP server allocates an IP address based on a table with MAC address/IP address pairs, which are manually filled in (perhaps by a network administrator). Only requesting clients with a MAC address listed in this table will be allocated an IP address. This feature (which is not supported by all devices) is variously called *Static DHCP Assignment* (by DD-WRT), *fixed-address* (by the dhcpd documentation), *DHCP reservation* or *Static DHCP* (by Cisco/Linksys), and *IP reservation* or *MAC/IP binding* (by various other router manufacturers).

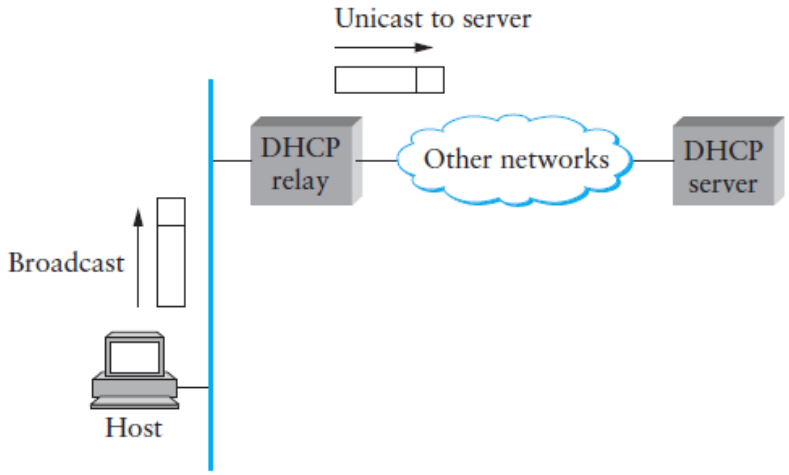


Figure 3.13: A DHCP relay agent receives a broadcast DHCPDISCOVER message from a host and sends a unicast DHCPDISCOVER to the DHCP server

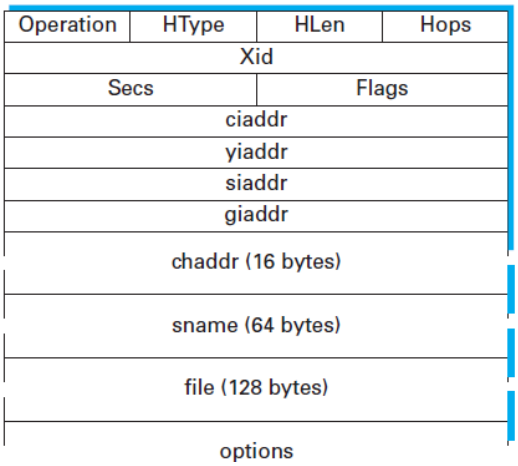


Fig 3.14: DHCP Packet Format

Figure 3.14 shows the format of a DHCP message. The message is actually sent using a protocol called UDP (the User Datagram Protocol) that runs over IP. DHCP is derived from an earlier protocol called BOOTP, and some of the packet fields are thus not strictly relevant to host configuration. When trying to obtain configuration information, the client puts its hardware address (e.g., its Ethernet address) in the chaddr field. The DHCP server replies by filling in the yiaddr ("your" IP address) field and sending it to the client. Other information such as the default router to be used by this client can be included in the options field.

In the case where DHCP dynamically assigns IP addresses to hosts, it is clear that hosts cannot keep addresses indefinitely, as this would eventually cause the server to exhaust its address pool. At the same time, a host cannot be depended upon to give back its address, since it might have crashed, been unplugged from the network, or been turned off. Thus, DHCP allows addresses to be "leased" for some period of time. Once the lease expires, the server is free to return that address to its pool. A host with a leased address clearly needs to renew the lease periodically if in fact it is still connected to the network and functioning correctly. Note that DHCP may also introduce some more complexity into network management, since it makes the binding between physical hosts and IP addresses much more dynamic.

ICMP- Internet Control Message Protocol

To make efficient use of the network resources, IP was designed to provide unreliable and connectionless best-effort datagram delivery service. As a consequence, IP has no error-control mechanism and also lacks mechanism for host and management queries. A companion protocol known as *Internet Control Message Protocol* (ICMP), has been designed to compensate these two deficiencies. ICMP messages can be broadly divided into two broad categories: error reporting messages and query messages as follows.

- Error reporting Messages: Destination unreachable, Time exceeded, Source quench, Parameter problems, Redirect

- Query: Echo request and reply, Timestamp request and reply, Address mask request and reply

The frame formats of these query and messages are shown in Figure. 3.15.

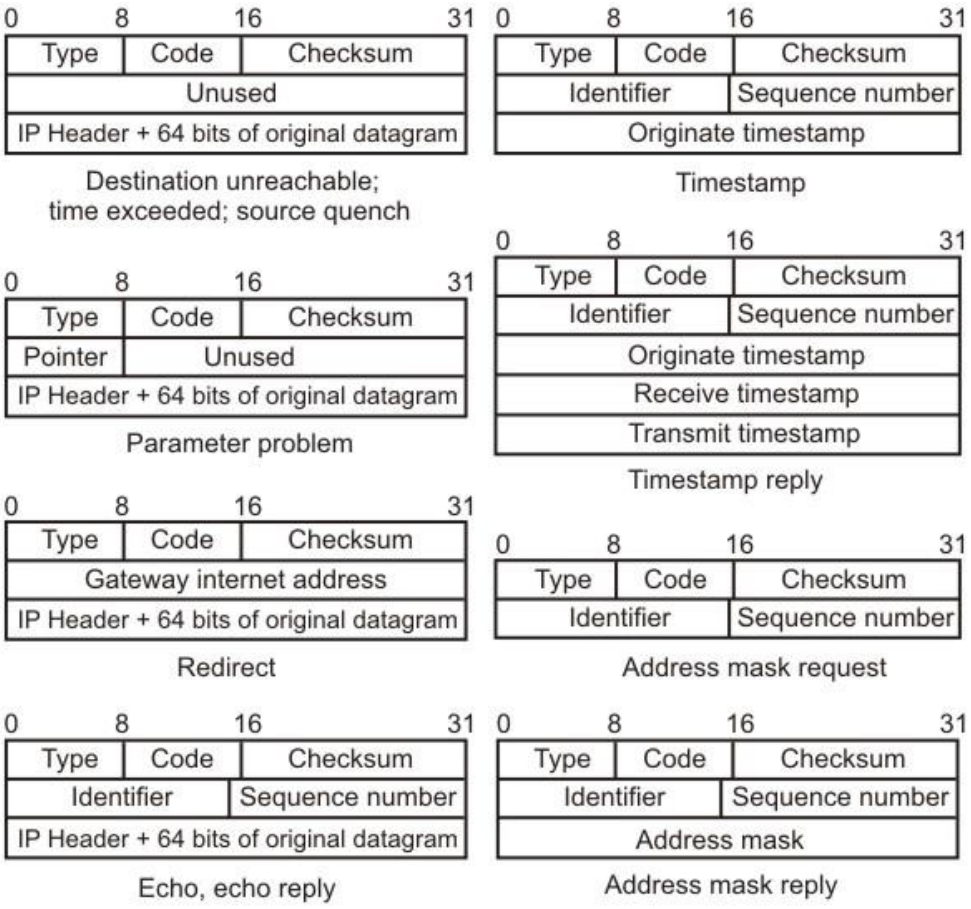


Figure 3.15. .ICMP Query and Message Formats

UNIT-III

Routing:

Introduction

Routing is the act of moving information across an inter-network from a source to a destination. Along the way, at least one intermediate node typically is encountered. It's also referred to as the process of choosing a path over which to send the packets. Routing is often contrasted with bridging, which might seem to accomplish precisely the same thing to the casual observer. The primary difference between the two is that bridging occurs at Layer 2 (the data link layer) of the OSI reference model, whereas routing occurs at Layer 3 (the network layer). This distinction provides routing and bridging with different information to use in the process of moving information from source to destination, so the two functions accomplish their tasks in different ways. The routing algorithm is the part of the network layer software responsible for deciding which output line an incoming packet should be transmitted on, i.e. what should be the next intermediate node for the packet.

Routing protocols use metrics to evaluate what path will be the best for a packet to travel. A *metric* is a standard of measurement; such as path bandwidth, reliability, delay, current load on that path etc; that is used by routing algorithms to determine the optimal path to a destination. To aid the process of path determination, routing algorithms initialize and maintain routing tables, which contain route information. Route information varies depending on the routing algorithm used.

Routing algorithms fill routing tables with a variety of information. Mainly Destination/Next hop associations tell a router that a particular destination can be reached optimally by sending the packet to a particular node representing the "next hop" on the way to the final destination. When a router receives an incoming packet, it checks the destination address and attempts to associate this address with a next hop. Some of the routing algorithm allows a router to have multiple "next hop" for a single destination depending upon best with regard to different metrics. For example, let's say router R2 is be best next hop for destination "D", if path length is considered as the metric; while Router R3 is the best for the same destination if delay is considered as the metric for making the routing decision.

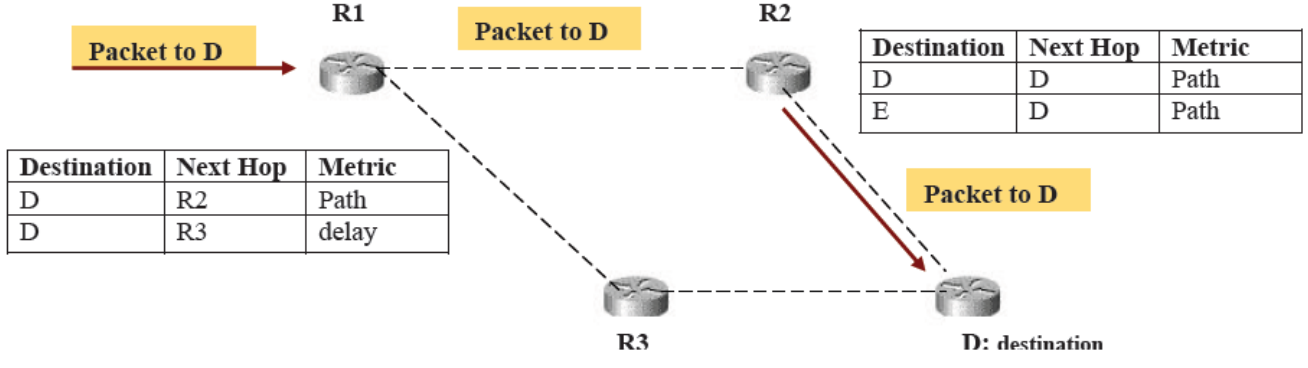


Figure 3.16 Typical routing in a small network

Figure 3.16 shows a small part of a network where packet destined for node "D", arrives at router R1, and based on the path metric i.e. the shortest path to destination is forwarded to router R2 which forward it to the final destination. Routing tables also can contain other information, such as data about the desirability of a path. Routers compare metrics to determine optimal routes, and these metrics differ depending on the design of the routing algorithm used. Routers communicate with one another and maintain their routing tables through the transmission of a variety of messages. The routing *update message* is one such message that generally consists of all or a portion of a routing table. By analyzing routing updates from all other routers, a router can build a detailed picture of network topology. A *link-state advertisement*, another example of a message sent between routers, informs other routers of the state of the sender's links. Link information also can be used to build a complete picture of network topology to enable routers to determine optimal routes to network destinations.

Desirable properties of a router are as follows:

- **Correctness and simplicity:** The packets are to be correctly delivered. Simpler the routing
- algorithm, it is better.
- **Robustness:** Ability of the network to deliver packets via some route even in the face of failures.
- **Stability:** The algorithm should converge to equilibrium fast in the face of changing conditions in the network.
- **Fairness and optimality:** obvious requirements, but conflicting.
- **Efficiency:** Minimum overhead

While designing a routing protocol it is necessary to take into account the following design parameters:

- **Performance Criteria:** Number of hops, Cost, Delay, Throughput, etc
- **Decision Time:** Per packet basis (Datagram) or per session (Virtual-circuit) basis
- **Decision Place:** Each node (distributed), Central node (centralized), Originated node (source)
- **Network Information Source:** None, Local, Adjacent node, Nodes along route, All nodes
- **Network Information Update Timing:** Continuous, Periodic, Major load change, Topology
- change

Classification of Routers

Routing algorithms can be classified based on the following criteria:
Static versus Adaptive
Single-path versus multi-path
Intra-domain versus inter-domain
Flat versus hierarchical
Link-state versus distance vector
Host-intelligent versus router-intelligent

Static versus Adaptive

This category is based on how and when the routing tables are set-up and how they can be modified, if at all. Adaptive routing is also referred as **dynamic routing** and Non-adaptive is also known as **static routing** algorithms. *Static routing algorithms* are hardly algorithms at all; the table mappings are established by the network administrator before the beginning of routing. These mappings do not change unless the network administrator alters them. Algorithms that use static routes are simple to design and work well in environments where network traffic is relatively predictable and where network design is relatively simple. Routing decisions in these algorithms are in no way based on current topology or traffic.

Because static routing systems cannot react to network changes, they generally are considered unsuitable for today's large, constantly changing networks. Most of the dominant routing algorithms today are *dynamic routing algorithms*, which adjust to changing network circumstances by analyzing incoming routing update messages. If the message indicates that a network change has occurred, the routing software recalculates routes and sends out new routing update messages. These messages permeate the network, stimulating routers to rerun their algorithms and change their routing tables accordingly. Dynamic routing algorithms can be supplemented with static routes where appropriate.

Single-Path versus Multi-path

This division is based upon the number of paths a router stores for a single destination.

Single path algorithms are where only a single path (or rather single next hop) is stored in the routing table. Some sophisticated routing protocols support multiple paths to the same destination; these are known as multi-path algorithms. Unlike single-path algorithms, these multipath algorithms permit traffic multiplexing over multiple lines. The advantages of multipath algorithms are obvious: They can provide substantially better throughput and reliability. This is generally called load sharing.

Intradomain versus Interdomain

Some routing algorithms work only within domains; others work within and between domains. The nature of these two algorithm types is different. It stands to reason, therefore, that an optimal intra-domain-routing algorithm would not necessarily be an optimal inter-domain-routing algorithm.

Flat Versus Hierarchical

Some routing algorithms operate in a flat space, while others use routing hierarchies. In a *flat routing system*, the routers are peers of all others. In a hierarchical routing system, some routers form what amounts to a routing backbone. Packets from non-backbone routers travel to the backbone routers, where they are sent through the backbone until they reach the general area of the destination. At this point, they travel from the last backbone router through one or more non-backbone routers to the final destination.

Routing systems often designate logical groups of nodes, called domains, autonomous systems, or areas. In *hierarchical systems*, some routers in a domain can communicate with routers in other domains, while others can communicate only with routers within their domain. In very large networks, additional hierarchical levels may exist, with routers at the highest hierarchical level forming the routing backbone.

The primary advantage of hierarchical routing is that it mimics the organization of most companies and therefore supports their traffic patterns well. Most network communication occurs within small company groups (domains). Because intradomain routers need to know only about other routers within their domain, their routing algorithms can be simplified, and, depending on the routing algorithm being used, routing update traffic can be reduced accordingly.

Link-State versus Distance Vector

This category is based on the way the routing tables are updated.

Distance vector algorithms (also known as Bellman-Ford algorithms): Key features of the distance vector routing are as follows:

The routers share the knowledge of the entire autonomous system
Sharing of information takes place only with the neighbors
Sharing of information takes place at fixed regular intervals, say every 30 seconds.

Link-state algorithms (also known as shortest path first algorithms) have the following key feature

The routers share the knowledge only about their neighbors compared to all the routers in the autonomous system
Sharing of information takes place only with all the routers in the internet, by sending small updates using flooding compared to sending larger updates to their neighbors
Sharing of information takes place only when there is a change, which leads to lesser internet traffic compared to distance vector routing

Because convergence takes place more quickly in link-state algorithms, these are somewhat less prone to routing loops than distance vector algorithms. On the other hand, link-state algorithms require more processing power and memory than distance vector algorithms. Link-state algorithms, therefore, can be more expensive to implement and support. Link-state protocols are generally more scalable than distance vector protocols.

Host-Intelligent Versus Router-Intelligent

This division is on the basis of whether the source knows about the entire route or just about the next-hop where to forward the packet. Some routing algorithms assume that the source end node will determine the entire route. This is usually referred to as **source routing**. In source-routing systems, routers merely act as store-and-forward devices, mindlessly sending the packet to the next stop. These algorithms are also referred to as **Host-Intelligent Routing,** as entire route is specified by the source node.

Other algorithms assume that hosts know nothing about routes. In these algorithms, routers determine the path through the internet based on their own own strategy. In the first system, the hosts have the routing intelligence. In the latter system, routers have the routing intelligence.

Routing Algorithm Metrics:

Routing tables contain information used by switching software to select the best route. In this section we will discuss the different nature of information they contain, and the way they determine that one route is preferable to others?

Routing algorithms have used many different metrics to determine the best route. Sophisticated routing algorithms can base route selection on multiple metrics, combining them in a single (hybrid) metric. All the following metrics have been used:

Path length
Delay
Bandwidth
Load
Communication cost
Reliability

Path length is the most common routing metric. Some routing protocols allow network administrators to assign arbitrary costs to each network link. In this case, path length is the sum of the costs associated with each link traversed. Other routing protocols define **hop count**, a metric that specifies the number of passes through internetworking products, such as routers, that a packet must pass through in a route from a source to a destination.

Routing delay refers to the length of time required to move a packet from source to destination through the internet. Delay depends on many factors, including the bandwidth of intermediate network links, the port queues (receive and transmit queues that are there in the routers) at each router along the way, network congestion on all intermediate network links, and the physical distance to be traveled. Because delay is a conglomeration of several important variables, it is a common and useful metric.

Bandwidth refers to the available traffic capacity of a link. All other things being equal, a 10-Mbps Ethernet link would be preferable to a 64-kbps leased line. Although bandwidth is a rating of the maximum attainable throughput on a link, routes through links with greater bandwidth do not necessarily provide better routes than routes through slower links. For example, if a faster link is busier, the actual time required to send a packet to the destination could be greater.

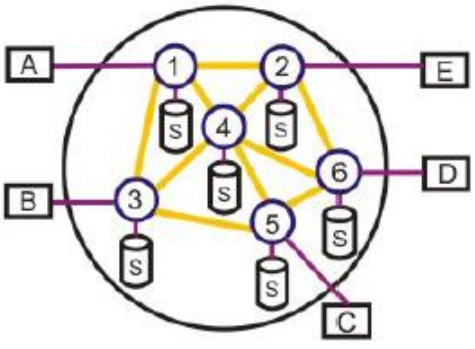
Load refers to the degree to which a network resource, such as a router, is busy. Load can be calculated in a variety of ways, including CPU utilization and packets processed per second. Monitoring these parameters on a continual basis can be resource-intensive itself.

Communication cost is another important metric, especially because some companies may not care about performance as much as they care about operating expenditures. Although line delay may be longer, they will send packets over their own lines rather than through the public lines that cost money for usage time.

Reliability, in the context of routing algorithms, refers to the dependability (usually described in terms of the bit-error rate) of each network link. Some network links might go down more often than others. After a network fails, certain network links might be repaired more easily or more quickly than other links. Any reliability factor can be taken into account in the assignment of the reliability ratings, which are arbitrary numeric values, usually assigned to network links by network administrators.

Fixed or Static Routing

In fixed routing a route is selected for each source-destination pair of nodes in the network. The routes are fixed; they may only change if there is a change in the topology of the network. A central routing matrix is created based on least-cost path, which is stored at a network control center. The matrix shows, for each source-destination pair of nodes, the identity of the next node on the route. Figure 3.17(a) shows a simple packet switching network with six nodes (routers), and Figure 3.17 (b) shows the central routing table created based on least-cost path algorithm. Figure 3.18 shows the routing tables that can be distributed in different nodes of the network.



Central Routing Directory

From node \ To node	1	2	3	4	5	6
1	-	2	3	2	2	2
2	1	-	1	6	6	6
3	1	1	-	4	5	1
4	6	6	3	-	5	6
5	4	4	3	4	-	4
6	2	2	2	4	4	-

Figures 3.17 (a) A simple packet switching network with six nodes (routers), (b) The central routing table created based on least-cost path

Node 1 Directory

Destination	Next Node
2	2
3	3
4	2
5	2
6	2

Node 2 Directory

Destination	Next Node
1	1
3	1
4	6
5	6
6	6

Node 3 Directory

Destination	Next Node
1	1
2	1
4	4
5	5
6	1

Node 4 Directory

Destination	Next Node
1	6
2	6
3	3
5	5
6	6

Node 5 Directory

Destination	Next Node
1	4
2	4
3	3
4	4
6	4

Node 6 Directory

Destination	Next Node
1	2
2	2
3	2
4	4
5	4

Figures 3.18 Routing tables that can be stored in different nodes of the network.

Flooding

Flooding requires no network information whatsoever. Every incoming packet to a node is sent out on every outgoing line except the one it arrived on. All possible routes between source and destination are tried. A packet will always get through if a path exists. As all routes are tried, at least one packet will pass through the shortest route. All nodes, directly or indirectly connected, are visited. Main limitation flooding is that it generates vast number of duplicate packets. It is necessary to use suitable damping mechanism to overcome this limitation. One simple is to use *hop-count*; a hop counter may be contained in the packet header, which is decremented at each hop, with the packet being discarded when the counter becomes zero. The sender initializes the hop counter. If no estimate is known, it is set to the full diameter of the subnet. Another approach is keep track of packets, which are responsible for flooding using a sequence number and avoid sending them out a second time. A variation, which is slightly more practical, is *selective flooding*. The routers do not send every incoming packet out on every line, only on those lines that go in approximately in the direction of destination. Some of the important utilities of flooding are:

Flooding is highly robust, and could be used to send emergency messages (e.g., military applications).

It may be used to initially set up the route in a virtual circuit.
Flooding always chooses the shortest path, since it explores every possible path in parallel.
Can be useful for the dissemination of important information to all nodes (e.g., routing information).

Intradomain versus Interdomain

In this section we shall discuss the difference between inter-domain and intra-domain routing algorithms or as they are commonly known as Exterior-gateway protocols and Interior gateway protocols respectively. Before going into the details of each of these routing algorithms, let's discuss the concept of Autonomous systems, which is the major differentiator between the two.

Autonomous Systems

As internet is a network of network that spans the entire world and because it's not under the control of a single organization or body, one cannot think of forcing a single policy for routing over it. Thus, comes the concept of autonomous system.

An **Autonomous System (AS)** is a connected segment of a network topology that consists of a collection of subnetworks (with hosts attached) interconnected by a set of routes. The subnetworks and the routers are expected to be under the control of a single operations and maintenance (O&M) organization i.e., an AS is under the same administrative authority. These ASs share a common routing strategy. An AS has a single "interior" routing protocol and policy. Internal routing information is shared among routers within the AS, but not with systems outside the AS. However, an AS announces the network addresses of its internal networks to other ASs that it is linked to. An AS is identified by an Autonomous System number.

Border gateway protocols: To make the network that is hidden behind the autonomous systems reachable throughout the internet each autonomous system agrees to advertise network reachability information to other Autonomous systems. An autonomous system shares routing information with other autonomous systems using the *Border Gateway Protocol* (BGP). Previously, the Exterior Gateway Protocol (EGP) was used. When two routers exchange network reachability information, the message carry the AS identifier (AS number) that router represents.

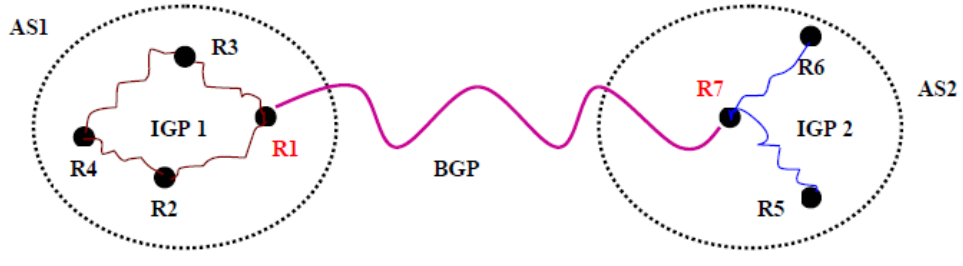


Figure 3.19 Two AS, each of which are using different IGPs internally and one BGP to communicate between each other

Figure 3.19 shows a conceptual view of two Autonomous systems (AS1 and AS2), each of which is using a different Interior gateway protocol (IGP1 and IGP2) as a routing protocol internally to the respective AS, while one router from each of the autonomous systems (R1 and R7) communicate among themselves to exchange the information of their respective Autonomous systems using a Border Gateway protocol, BGP. These two routers (R1 and R7) understands both interior and border gateway protocols.

Interior gateway protocols: In small and slowly changing network the network administrator can establish or modify routes by hand i.e. manually. Administrator keeps a table of networks and updates the table whenever a network is added or deleted from the autonomous system. The disadvantage of the manual system is obvious; such systems are neither scalable nor adaptable to changes. Automated methods must be used to improve reliability and response to failure. To automate the task this task, interior router (within a autonomous system) usually communicate with one another, exchanging network routing information from which reachability can be deduced. These routing methods are known as Interior gateway Protocols (IGP).

In the following lessons we shall discuss two Interior gateway protocols namely, routing Information Protocol (RIP) and Open Shortest path first (OSPF) and a border gateway protocol, for the better understanding of Routing.

Routing Information Protocol

Routing Information Protocol (RIP) is a simple routing protocol, originally defined in 1988 as RFC 1058 and more recently as RFC 1723, based upon the original ARPANET routing algorithm. RIP involves a router calculating the best route to all other routers in a network using a **shortest path** algorithm attributable to Bellman (1957) and Ford and Fulkerson (1962). The shortest path in this case is the one that passes through the least number of routers. Each router traversed is known as a **hop**. Therefore the shortest path is described by a **hop count**, or **distance vector**. This is a crude measure of distance or cost to reach a destination. It takes no account of other factors such as propagation delay or available bandwidth. RIP then builds a routing database that contain stables of the best routes to all the other routers. Each router then advertises its own routing tables to all other routers. Although RIP is simple to implement it is only efficient in small networks since, as the size of a network grows, RIP datagrams can become very long, thus consuming substantial amounts of bandwidth.

The algorithm is distributed because it involves a number of nodes (routers) within an Autonomous system. It consists of the following steps:

Each node calculates the distances between itself and all other nodes within the AS and stores this information as a table.
Each node sends its table to all neighbouring nodes.
When a node receives distance tables from its neighbours, it calculates the shortest routes to all other nodes and updates its own table to reflect any changes.

The main disadvantages of Bellman-Ford algorithm in this setting are

Does not scale well
Changes in network topology are not reflected quickly since updates are spread node-by-node.
Counting to infinity
Few modifications, which will be discussed later in this section, are made in Bellman-ford algorithm to overcame the abovementioned disadvantages.

RIP partitions participants (node within the AS) into *active* and *passive* (slient) nodes. Active routers advertise their routes to others; passive node just listen and updates their routes based on the advertisements. Passive nodes donot advertise. Only routers can run RIP in active mode; other host run RIP in passive mode. A router running in active mode broadcasts a message or advertisement every 30 seconds. The message contains information taken from the router's current routing database. Each message consists of pairs, where each pair contains a IP network address and a integer distance to that network in terms of number hops to reach the destination. All active and passive nodes listen to the advertisements and updates their route tables. Lets discuss an example for better understanding. Consider the Autonomous system consisting of 4 routers (R1, R2, R3, R4) shown in Figure. 3.20.

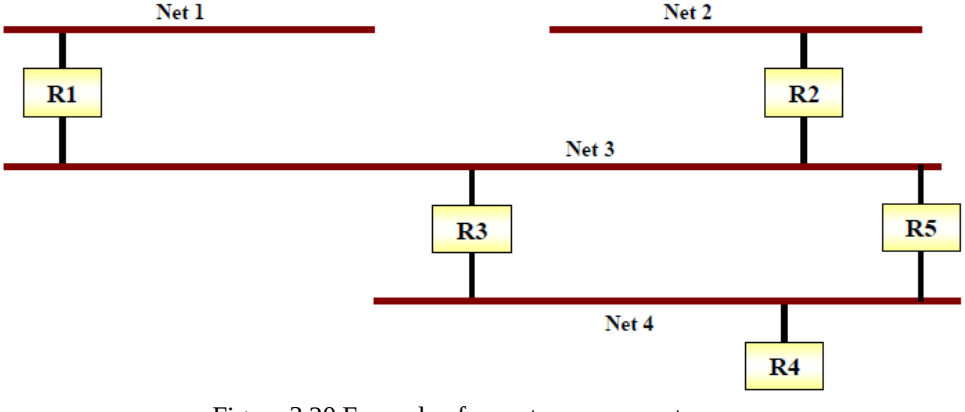


Figure 3.20 Example of an autonomous system

R2 will broadcast a message on network 3 (Net 3) containing a pair (2, 1), meaning that it can reach network 2 at a cost of 1. Router R1 and R3 will receive this broadcast and install a route for network 2 (Net 2) in their respective routing tables, through R2 (at a cost of 2, as now there are two routers in between either (R1 or R2) or (R2 and R3). Later on Router R3 will broadcast a message with pair (2, 2) on network 4 (Net 4). Eventually all router will have a entry for Network 2 (Net 2) in their routing tables, and same is the case with the routes for other networks too.

RIP specifies that once a router learns a route from another router, it must keep that route until it learns a better one. In our example, if router R3 and R5 both advertise network 2 (Net 2) or network 1 (Net 1) at cost of 2; router R2 will install a route through the one that happens to advertise first. Hence, to prevent routes from oscillating between two or more equal cost paths, RIP specifies that existing routes should be retained until a new route has strictly lower cost.

As RIP is a distance vector routing protocol, it represents the routing information in terms of the cost of reaching the specific destination. Circuit priorities are represented using numbers between 1 and 15. This scale establishes the order of use of links. The router decides the path to use base on the priority list. Once the priorities are established, the information is stored in a RIP routing table. Each entry in a RIP routing table provides a variety of information, including the ultimate destination, the next hop on the way to that destination, and a metric. The metric indicates the distance in number of hops to the destination. Other information can also be present in the routing table, including various timers associated with the route.

A distance-vector routing protocol uses the Bellman-Ford algorithm to calculate paths. A distance-vector routing protocol requires that a router informs its neighbors of topology changes periodically and, in some cases, when a change is detected in the topology of a network. Compared to link-state protocols, which require a router to inform all the nodes in a network of topology changes, distance-vector routing protocols have less computational complexity and message overhead. Distance Vector means that Routers are advertised as vector of distance and direction. 'Direction' is represented by next hop address and exit interface, whereas 'Distance' uses metrics such as hop count.

Routers using distance vector protocol do not have knowledge of the entire path to a destination. Instead DV uses two methods:

1. Direction in which or interface to which a packet should be forwarded.
2. Distance from its destination.

Open Shortest Path First

Open Shortest Path First (OSPF) is another Interior Gateway Protocol. It is a routing protocol developed for Internet Protocol (IP) networks by the Interior Gateway Protocol (IGP) working group of the Internet Engineering Task Force (IETF). The working group was formed in 1988 to design an IGP based on the Shortest Path First (SPF) algorithm for use in the Internet. OSPF was created because in the mid-1980s, the Routing Information Protocol (RIP) was increasingly incapable of serving large, heterogeneous internetworks. OSPF being a SPF algorithm scales better than RIP. Few of the important features of OSPF are as follows:

- This protocol is *open*, which means that its specification is in the public domain. It means that anyone can implement it without paying license fees. The OSPF specification is published as Request For Comments (RFC) 1247.
- OSPF is based on the *SPF algorithm*, which is also referred to as the Dijkstra's algorithm, named after the person credited with its creation.
- OSPF is a *link-state routing protocol* that calls for the sending of link-state advertisements (LSAs) to all other routers within the same hierarchical area. Information on attached interfaces, metrics used, and other variables are included in OSPF LSAs. As a link-state routing protocol, OSPF contrasts with RIP, which are distance-vector routing protocols. Routers running the distance-vector algorithm send all or a portion of their routing tables in routing-update messages only to their neighbors.
- OSPF specifies that all the exchanges between routers must be *authenticated*. It allows a variety of authentication schemes, even different areas can choose different authentication schemes. The idea behind authentication is that only authorized router are allowed to advertise routing information.
- OSPF include Type of service Routing. It can calculate separate routes for each *Type of Service (TOS)*, for example it can maintain separate routes to a single destination based on hop-count and high throughput.
- OSPF provides *Load Balancing*. When several equal-cost routes to a destination exist, traffic is distributed equally among them.
- OSPF allows supports host-specific routes, Subnet-specific routes and also network-specific routes. OSPF allows sets of networks to be grouped together. Such a grouping is called an *Area*. Each Area is self-contained; the topology of an area is hidden from the rest of the Autonomous System and from other Areas too. This information hiding enables a significant reduction in routing traffic.
- OSPF uses different message formats to distinguish the information acquired from within the network (internal sources) with that which is acquired from a router outside (external sources).

Just like any other Link state routing, OSPF also has the following features:
Advertise about neighborhood: Instead of sending its entire routing table, a router sends information about its neighborhood only.
Flooding: Each router sends this information to every other router on the internetwork, not just to its neighbors. It does so by a process of flooding. In Flooding, a router sends its information to all its neighbors (through all of its output ports). Every router sends such messages to each of its neighbor, and every router that receives the packet sends copies to its neighbor. Finally, every router has a copy of same information.
Active response: Each outer sends out information about the neighbor when there is a change.

Initialization: When an SPF router is powered up, it initializes its routing-protocol data structures and then waits for indications from lower-layer protocols that its interfaces are functional.

After a router is assured that its interfaces are functioning, it uses the OSPF Hello protocol (sends greeting messages) to acquire neighbors, which are routers with interfaces to a common network. The router sends hello packets to its neighbors and receives their hello packets. These messages are also known as greeting messages. It then prepares an LSP (Link State packet) based on the results of this Hello protocol.
An example of an internet is shown in Figure.3.21, where R1 is a neighbor of R2 and R4, R2 is a neighbor of R1, R3 and R4**,** R3 is a neighbor of R2 and R4**,** R4 is a neighbor of R1, R2 and R3**.** So each router will send greeting messages to its entire neighbors.

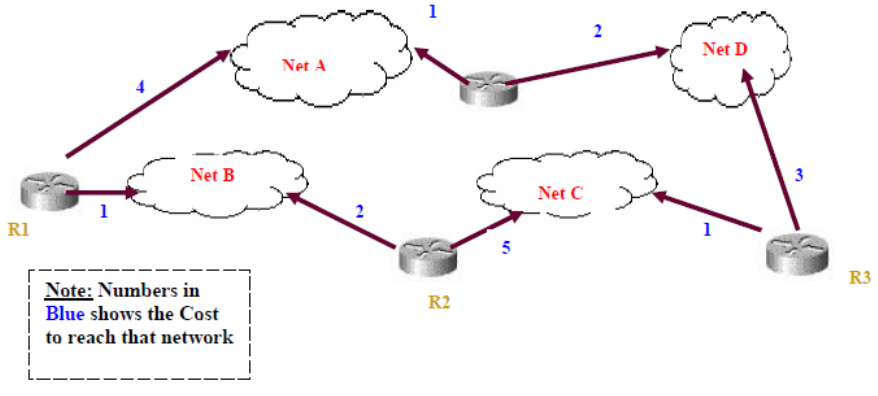


Figure 3.21 An example internet

Information from neighbors: A router gets its information about its neighbor by periodically sending them a short greeting packet (this is known as *Hello Message* format). If neighbor responds to this greeting message as expected, it is assumed to be alive and functioning. If it does not, a change is assumed to have occurred and the sending router then alerts the rest of the network in its next LSP, about this neighbor being down. These Greeting messages are small enough that they do not use network resources to a significant amount, unlike the routing table updates in case of a vector-distance algorithm.
Link state packet: The process of router flooding the network with information about its neighborhood is known as *Advertising*. The basis of advertising is a short packet called a *Link state Packet (LSP).* An LSP usually contains 4 fields: the ID of the advertiser (Identifier of the router which advertises the message), ID of the destination network, The cost, and the ID of the neighbor router. Figure 3.22 shows the LSP of a router found after the *Hello* protocol and Fig. 3.23 shows the basic fields of LSP.

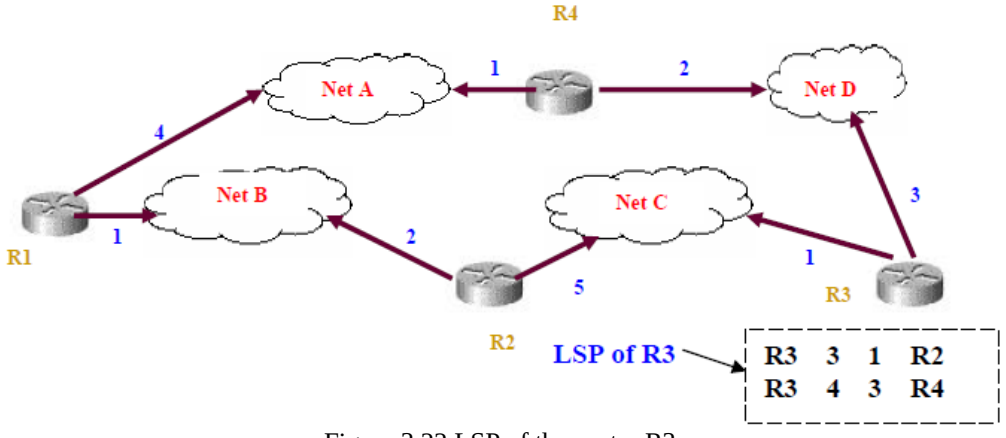


Figure 3.22 LSP of the router R3

Advertiser	Network	Cost	Neighbor
----------	----------	----------	----------
----------	----------	----------	----------

Figure 3.23: The LSP fields

Link State Database: Every router receives every LSP and then prepares a database, which represents a complete network topology. This Database is known as Link State Database. Figure 2.24 shows the database of our sample internetwork. These databases are also known as *topological database.*

Advertiser	Network	Cost	Neighbor
R1	A	4	R4
R1	B	1	R2
R2	B	2	R1
R2	C	5	R3
R3	C	1	R2
R3	D	3	R4
R4	A	1	R1
R4	D	2	R3

Figure3.24:Link state Database

Because every router receives the same LSPs, every router builds the same database. Every router uses it to calculate its routing table. If a router is added or deleted from the system, the whole database must be changed accordingly in all routers.

Shortest Path calculation: After gathering the Link State database, each router applies an algorithm called the Dijkstra algorithm to calculate the shortest distance between any two nodes. The Dijkstra's algorithm calculates the shortest path between two points on a network using a graph made up of nodes and arcs, where nodes are the Routers and the network, while connection between router and network is refer to as arcs.
The algorithm begins to build a tree by identifying its root as shown in Figure. 3.25. The router is the root itself. The algorithm then attaches all other nodes that can be reached from that router; this is done with the help of the Link state database.

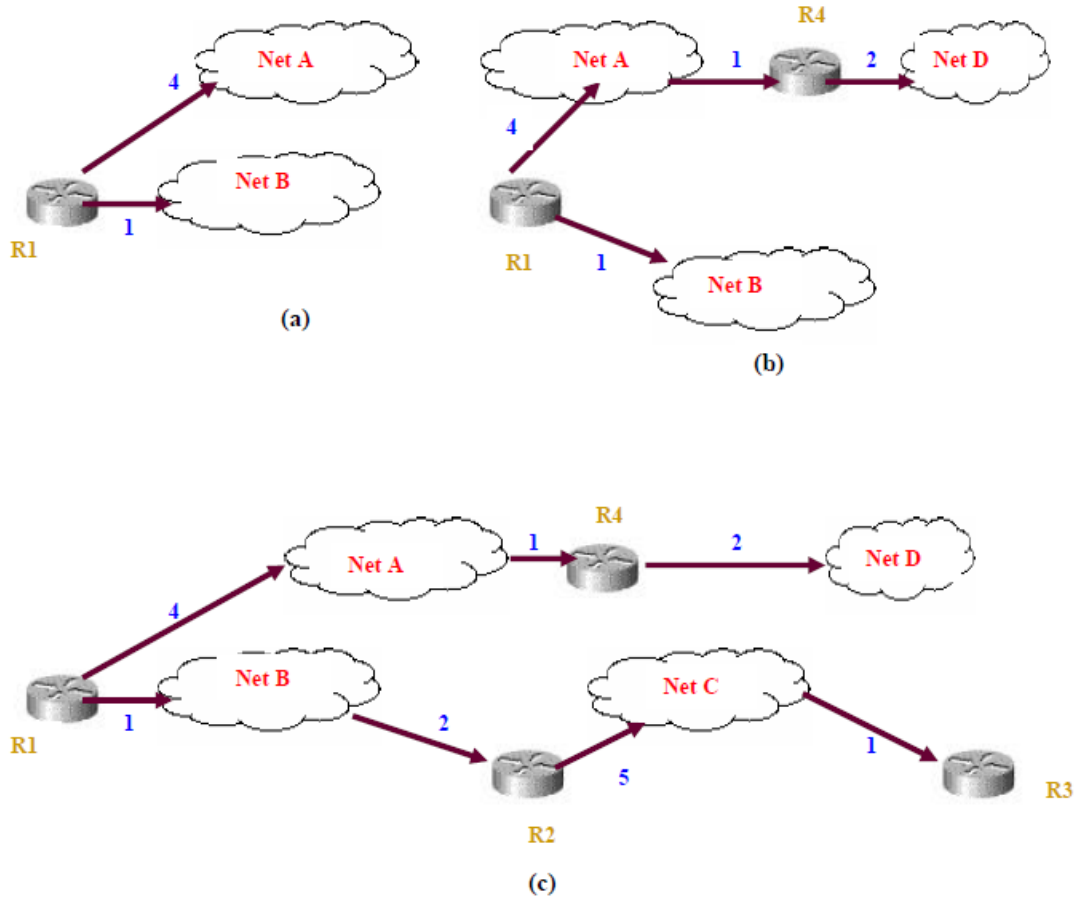


Figure 3.25 Path calculation for router R1

From this shortest path calculation each router makes its routing table, as per our example internet table for router R1 is given in Fig. 7.26. All other routers too have a similar routing table made up after this point.

Network	Cost	Next Router
A	4	-----
B	1	-----
C	8	R2
D	7	R4

Figure3.26: Routing table example

Subnets and subnet routing

Many Internet networks, in particular type A and type B, can be quite large with many hosts, they must be separated into *sub-nets*, because it is not workable to have thousands of hosts on one physical LAN. In many ways one administrative internet network (an *autonomous system*) with subnets is itself an *internet*, there must be subnet routers.
Subnet addresses
The first problem is to divide the host address space, this must (like type A, B and C nets) be a power of two. Consider figure 3.27. So if the herts.ac.uk net address is 147.197.0.0, 16 bits give the network address and 16 bits the host, the host is further divided into a 5 bit subnet number (giving upto 32 subnets) and an 11 bit host address (giving upto 2048 hosts on each subnet).

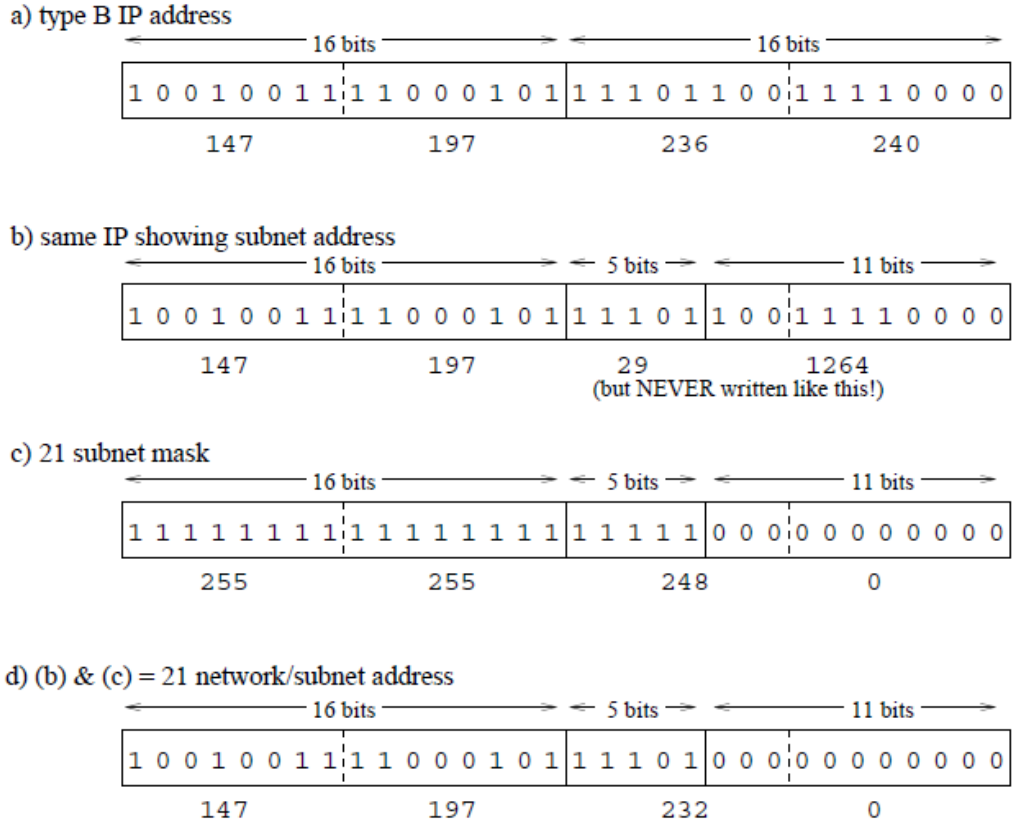


Figure3.27: Subnet addressing

An example herts.ac.uk address (B type) is given in part (a) of the picture, 147.197.236.240. The subnet part is shown in part (b), note that this is just a simple 32 bit number, it is only by convention that it is written as four 8 bit numbers in decimal, therefore we could say this is subnet 29 (the 5 bits), host 1264 (given by the 11 bits), but that would be confusing so it is still written conventionally.

Packet forwarding with subnets

The rest of the Internet doesn't know or care about the subnets on individual networks, routing from outside is still to the whole network but all the systems on the network must be aware of the subnets—they must forward to the correct subnet. The way that packet forwarding occurs is to compare the *network* part of the address with entries in the forwarding table to select the destination.

It is only possible to send *directly* to a system on a LAN if it is on the same *subnet*, so it is necessary to examine the net and subnet number, at Hatfield the network and subnet part is 21 bits long, It must be provided with a *mask* that when and-ed with the address leaves only the net+subnet part which can be compared with the network numbers. For the Hatfield subnet the subnet mask is 21 bits long, when written in conventional IP notation is is 255.255.248.0, sub-picture (c) shows the binary value of the mask. The result of and-ing the mask with the example address 147.197.236.240 is shown in binary in (d), in conventional IP notation it is 147.197.232.0.

It is also possible to examine the forwarding table on host 149.197.236.240, it shows the local subnet number and the subnet mask applied to destination addresses.

Destination	Gateway	Genmask	Flags	Metric	Use	Iface
147.197.232.0	0.0.0.0	255.255.248.0	U	0	0	eth1
0.0.0.0	147.197.232.1	0.0.0.0	UG	0	0	eth1

If this machine 149.197.236.240 sends to 149.197.239.69 then the forwarding table will mask the destination address with the subnetmask 255.255.248.0 giving 147.197.232.0 which will be sent out directly (no gateway). If, however, the destination is 147.197.200.44 the mask will produce 147.197.200.0, this won't match the first network destination so the last line will be used instead and the packet will be forwarded to the gateway 147.197.232.1. Note that this treats the problem of routing to other subnets and to other networks in the same way, in both cases the packets go to the gateway and it must decide to forward to another subnet or go out to the Internet.

Another notation for subnet addresses

Note that forwarding with subnets blurs the distinction between the *network* and *host* parts of an address. If subnets are used it is not enough to recognise a type A, B or C address and know what the network address is. Consequently there is a different way to write network addresses that makes absolutely clear what the network (maybe with subnet) part is:

full-network-address/*number-of-bits-of-network-part*

for example the address of the subnet my machine uses is: 147.197.232.0/21 which gives the length of the network+subnet part. It gives two things: the subnet mask (length 21), ie. 255.255.248.0, and it gives the value of the 21 bits—the network number.

CIDR:

CIDR (Classless Inter-Domain Routing, sometimes known as *supernetting*) is a way to allocate and specify the Internet addresses used in inter-domain routing more flexibly than with the original system of Internet Protocol (IP) address classes. As a result, the number of available Internet addresses has been greatly increased. CIDR is now the routing system used by virtually all gateway hosts on the Internet's backbone network. The Internet's regulating authorities now expect every Internet service provider (ISP) to use it for routing.

The original Internet Protocol defines IP addresses in four major classes of address structure, Classes A through D. Each of these classes allocates one portion of the 32-bit Internet address format to a network address and the remaining portion to the specific host machines within the network specified by the address. One of the most commonly used classes is (or was) Class B, which allocates space for up to 65,533 host addresses. A company who needed more than 254 host machines but far fewer than the 65,533 host addresses possible would essentially be "wasting" most of the block of addresses allocated. For this reason, the Internet was, until the arrival of CIDR, running out of address space much more quickly than necessary. CIDR effectively solved the problem by providing a new and more flexible way to specify network addresses in routers. (With a new version of the Internet Protocol - IPv6 - a 128-bit address is possible, greatly expanding the number of possible addresses on the Internet. However, it will be some time before IPv6 is in widespread use.)

Using CIDR, each IP address has a *network prefix* that identifies either an aggregation of network gateways or an individual gateway. The length of the network prefix is also specified as part of the IP address and varies depending on the number of bits that are needed (rather than any arbitrary class assignment structure). A destination IP address or route that describes many possible destinations has a shorter prefix and is said to be less specific. A longer prefix describes a destination gateway more specifically. Routers are required to use the most specific or longest network prefix in the routing table when forwarding packets.

A CIDR network address looks like this:

192.30.250.00/18

The "192.30.250.00" is the network address itself and the "18" says that the first 18 bits are the network part of the address, leaving the last 14 bits for specific host addresses. CIDR lets one routing table entry represent an aggregation of networks that exist in the forward path that don't need to be specified on that particular gateway, much as the public telephone system uses area codes to channel calls toward a certain part of the network. This aggregation of networks in a single address is sometimes referred to as a *supernet*.

CIDR is supported by the Border Gateway Protocol, the prevailing exterior (interdomain) gateway protocol. (The older exterior or interdomain gateway protocols, Exterior Gateway Protocol and Routing Information Protocol, do not support CIDR.) CIDR is also supported by the OSPF interior or intradomain gateway protocol.

Inter-domain routing

The interdomain routing involves AS sharing their reachability information with each other AS.
 The goal of interdomain routing is *reachability* and not optimality.
 The two major interdomain routing protocols are Exterior Gateway Protocol (*EGP*) and Border Gateway Protocol (*BGP*).

The problems in interdomain routing are:

 An internet backbone must be able to route packets to any destination, i.e., there should be a match in the routing/forwarding table.
 Each AS has its own intradomain routing protocols and chooses the metric assigns to path. This varies from one AS to another.
 Autonomous systems may not trust each other.

BGP:

The **Border Gateway Protocol (BGP)** is an inter-autonomous system routing protocol. An autonomous system (AS) is a network or group of networks under a common administration and with common routing policies. BGP is used to exchange routing information for the Internet and is the protocol used between Internet service providers (ISP), which are different ASes.

One of the most important characteristics of BGP is its *flexibility*. The protocol can connect together any internetwork of autonomous systems using an arbitrary topology. The only requirement is that each AS have at least one router that is able to run BGP and that this router connect to at least one other AS's BGP router. Beyond that, "the sky's the limit," as they say. BGP can handle a set of ASs connected in a full mesh topology (each AS to each other AS), a partial mesh, a chain of ASes linked one to the next, or any other configuration. It also handles changes to topology that may occur over time.

The primary function of a BGP speaking system is to exchange network reachability information with other BGP systems. This network reachability information includes information on the list of Autonomous Systems (ASs) that reachability information traverses. BGP constructs a graph of autonomous systems based on the information exchanged between BGP routers. As far as BGP is concerned, whole Internet is a graph of ASs, with each AS identified by a Unique AS number. Connections between two ASs together form a path and the collection of path information forms a route to reach a specific destination. BGP uses the path information to ensure the loop-free inter-domain routing.

Another important assumption that BGP makes is that it doesn't know anything about what happens within the AS. This is of course an important prerequisite to the notion of an AS being *autonomous* - it has its own internal topology and uses its own choice of routing protocols to determine routes. BGP only takes the information conveyed to it from the AS and shares it with other ASs.

When a pair of autonomous systems agrees to exchange routing information, each must designate a router that will speak BGP on its behalf; the two routers are said to become *BGP peers* of one another. As a router speaking BGP must communicate with a peer in another autonomous system, usually a machine, which is near to the edge (Border) of the autonomous system is selected for this. Hence, BGP terminology calls the machine a *Border Gateway Router*

BGP Characteristics

BGP is different from other routing protocols in several ways. Most important being that BGP is neither a pure distance vector protocol nor a pure link state protocol. Let's have a look at some of the characteristics that stands BGP apart from other protocols.

- **Inter-Autonomous System Configuration**: BGP's primary role is to provide communication between two autonomous systems.
- **Next-Hop paradigm**: Like RIP, BGP supplies next hop information for each destination.
- **Coordination among multiple BGP speakers within the autonomous system**: If an Autonomous system has multiple routers each communicating with a peer in other autonomous system, BGP can be used to coordinate among these routers, in order to ensure that they all propagate consistent information.

- **Path information**: BGP advertisements also include path information, along with the reachable destination and next destination pair, which allows a receiver to learn a series of autonomous system along the path to the destination.
- **Policy support**: Unlike most of the distance-vector based routing, BGP can implement policies that can be configured by the administrator. For Example, a router running BGP can be configured to distinguish between the routes that are known from within the Autonomous system and that which are known from outside the autonomous system.
- **Runs over TCP**: BGP uses TCP for all communication. So the reliability issues are taken care by TCP.
- **Conserve network bandwidth**: BGP doesn't pass full information in each update message. Instead full information is just passed on once and thereafter successive messages only carries the incremental changes called **deltas**. By doing so a lot of network Bandwidth is saved. BGP also conserves bandwidth by allowing sender to aggregate route information and send single entry to represent multiple, related destinations.
- **Support for CIDR**: BGP supports classless addressing (CIDR). That it supports a way to send the network mask along with the addresses.
- **Security**: BGP allows a receiver to authenticate messages, so that the identity of the sender can be verified.

BGP Functionality and Route Information Management

The job of the Border Gateway Protocol is to facilitate the exchange of route information between BGP devices, so that each router can determine efficient routes to each of the networks on an IP internetwork. This means that descriptions of routes are the key data that BGP devices work with. But in a broader aspect, BGP peers perform three basic functions. The First function consists of initial peer acquisition and authentication. Both the peers establish a TCP connection and perform message exchange that guarantees both sides have agreed to communicate. The second function primarily focus on sending of negative or positive reachability information, this step is of major concern. The Third function provides ongoing verification that the peers and the network connection between them are functioning correctly. Every BGP speaker is responsible for managing route descriptions according to specific guidelines established in the BGP standards.

BGP Route Information Management Functions

Conceptually, the overall activity of route information management can be considered to encompass four main tasks:

Route Storage: Each BGP stores information about how to reach networks in a set of special databases. It also uses databases to hold routing information received from other devices.
Route Update: When a BGP device receives an *Update* from one of its peers, it must decide how to use this information. Special techniques are applied to determine when and how to use the information received from peers to properly update the device's knowledge of routes.

Route Selection: Each BGP uses the information in its route databases to select good routes to each network on the internetwork.

Route Advertisement: Each BGP speaker regularly tells its peers what it knows about various networks and methods to reach them. This is called *route advertisement* and is accomplished using BGP *Update* messages.

BGP Attributes

BGP Attributes are the properties associated with the routes that are learned from BGP and used to determine the best route to a destination, when multiple routes are available. An understanding of how BGP attributes influence route selection is required for the design of robust networks. Following attributes are used by BGP in the route selection process:

- AS_path
- Next hop
- Weight
- Local preference
- Multi-exit discriminator
- Origin

- Community

BGP Path Selection

BGP could possibly receive multiple advertisements for the same route from multiple sources. BGP selects only one path as the best path. When the path is selected, BGP puts the selected path in the IP routing table and propagates the path to its neighbors. BGP uses the following criteria, in the order presented, to select a path for a destination:

- If the path specifies a next hop that is inaccessible, drop the update.
- Prefer the path with the largest weight.
- If the weights are the same, prefer the path with the largest local preference.
- If the local preferences are the same, prefer the path that was originated by BGP running on this router.
- If no route was originated, prefer the route that has the shortest AS_path.
- If all paths have the same AS_path length, prefer the path with the lowest origin type (where IGP is lower than EGP, and EGP is lower than incomplete).
- If the origin codes are the same, prefer the path with the lowest MED attribute.
- If the paths have the same MED, prefer the external path to the internal path.
- If the paths are still the same, prefer the path through the closest IGP neighbor.
- Prefer the path with the lowest IP address, as specified by the BGP router ID.
- CIDR and subneting could not solve the address exhaustion faced by IPv4. IPv6 was evolved to solve this problem.

IPv6

The striking features of IPv6 are:

o support for real-time services
o security support
o auto configuration
o enhanced routing functionality, including support for mobile hosts

Addresses Space

 IPv6 provides a 128-bit address space as opposed to IPv4's 32-bit.
 IPv6 addresses do not have classes, but classification is based on the leading bits.
 The IPv4's classes A, B and C start with 001 prefix (unicast addresses).
 Large chunks of address space are left *unassigned* to allow for new features in the future.
 The address spaces (0000 001 and 0000 010) are *reserved* for non-IP address such as IPX
 Link local address enables a host to configure address automatically that works on the local network.
 Site local address allows valid addresses on a private network which is not connected to internet.
 Multicast address (start with a byte of 1s) serves the purpose of class D address.The common prefix (excluding unassigned) and their usage is tabulated below:

Prefix	**Usage**
0000 0000	Reserved
0000 001	Reserved for ISO protocol
0000 010	Reserved for Novell network layer
001	Aggregated Global Unicast Addresses (Class A, B and C)
010	Provider-based unicast addresses
100	Geographic-based unicast addresses
1111 1110 10	Link local use addresses
1111 1110 11	Site local use addresses
1111 1111	Multicast addresses

Address Notation

 The standard representation is **x:x:x:x:x:x:x:x** where **x** is a hexadecimal representation of a 16-bit address separated by colon (:) as shown below

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

 An IPv6 address with a large number of contiguous 0s is written compactly by omitting the 0 fields as shown below

47CD:0000:0000:0000:0000:0000:A456:0124 is written as **47CD::A456:0124**

 An IPv4 address can be mapped to IPv6 address by prefixing the 32-bit IPv4 address with 2 bytes of all 1s and then zero-extending the result to 128 bits.

128. 96.33.81 is written as **::FFFF:128.96.33.81**

Aggregatable Global Unicast Addresses

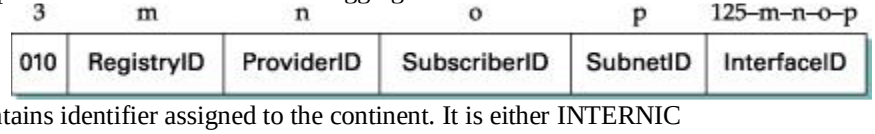
 The goal of the IPv6 address allocation plan is to provide aggregation of routing information to reduce the burden on intradomain routers.

 Aggregation is done by assigning prefixes at continental level.

 Continental boundaries form natural divisions in the Internet topology

o 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.

 The format for provider-based unicast address aggregation is shown below.



o RegistryID contains identifier assigned to the continent. It is either INTERNIC (North America), RIPNIC (Europe) or APNIC (Asia and Pacific)

o ProviderID variable-length field identifies the provider for Internet access such as an ISP.

o SubscriberID specifies the assigned subscriber identifier

o SubnetID defines a specific subnet under the territory of subscriber.

o InterfaceID contains the link level or physical address.

Anycast, Multicast and Reserved addressing

 Multicast address as in IPv4 is used to address a group of hosts.

 IPv6 also defines *Anycast* addresses. A packet destined for an anycast address is delivered to only one member of the anycast group (the nearest one).

 Reserved addresses start with prefix of eight 0s. It is classified into

- *unspecified* address is used when a host does not know its address
- *compatible* address is used when IPv6 hosts communicate through IPv4 network
- *mapped* address is used when a IPv6 host communicates with a IPv4 host.

 IPv6 header defines *Local* addresses for private networks. It is classified into

o *Site* local address for use in a isolated site with several subnets.

o *Link* local address for use in a isolated subnet

Packet Format

The IPv6 base header is always 40 bytes long. The packet format and field description follows:

 Version—specifies the IP version, i.e., 6.

 TrafficClass—defines the priority of the packet with respect to traffic congestion. It is either congestion-controlled or non-congestion controlled

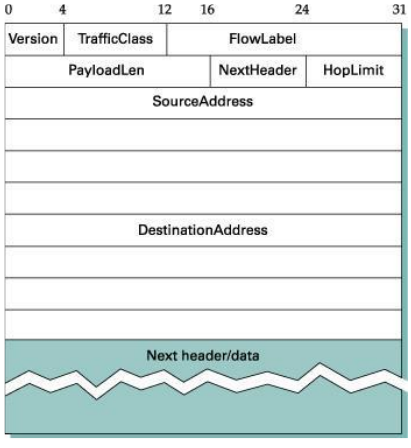
 FlowLabel—is designed to provide special handling for a particular flow of data. The router handles flow with the help of a flow table.

 PayloadLen—gives the length of the packet, excluding the IPv6 header

 NextHeader—If options are required, then it is specified in one or more special headers following the IP header, its value is contained in NextHeader field. Otherwise, it identifies the higher-level protocol (TCP/UDP).

 HopLimit—This field serves the same purpose as TTL field in IPv4.

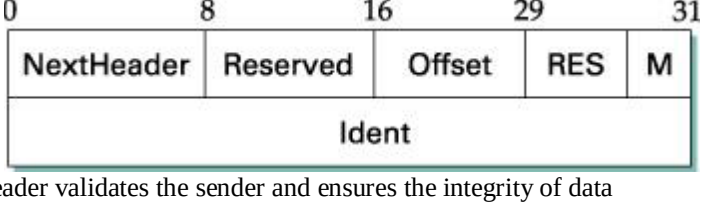
 SourceAddress and DestinationAddress—contains 16-byte address of the source and destination host respectively.



Extension Header

 To provide greater functionality to IP datagram, the base header can be followed by up to six extension headers.
 IPv6 treats options as extension headers, if present must appear in a specific order.
 Each option has its own type of extension header. The six types of extension header are:

o *Hop-by-Hop*—This header is used when the source needs to pass information to all routers visited by the datagram.
o *Source Routing*—This header accounts for both strict and loose source routing.
o *Fragmentation*—In IPv6, only the original source can fragment. A source must use a path MTU discovery technique to find the smallest MTU on the path. This header is present only if fragmentation exists. The header format is



o *Authentication*—This header validates the sender and ensures the integrity of data
o *Encrypted Security Payload*—The ESP header provides confidentiality and guards against eavesdropping.
o *Destination*—This header is used when source needs to pass information to destination only. Intermediate routers cannot access this information.

Other features

 IPv6 provides a new form of *autoconfiguration* called *stateless* auto-configuration, which does not require a DHCP server.

The advantages of IPv6.

 *Large address space*An IPv6 address is 128 bits long. Compared with the 32-bit address of IPv4, this is a huge (296) increase in the address space.
 *Better header format*IPv6 uses a new header format in which options are separated from the base header and inserted, when needed.
 *New options*IPv6 has new options to allow for additional functionalities.
 *Allowance for extension*IPv6 is designed to allow the extension of the protocol if required by new technologies or applications.
 *Support for resource allocation*In IPv6, flow label has been added to enable the source to request special handling of the packet such as real-time audio and video.
 *Support for more security*The encryption and authentication options in IPv6 provide confidentiality and integrity of the packet.

Multicast Routing

Some applications, such as a multiplayer game or live video of a sports event streamed to many viewing locations, send packets to multiple receivers. Unless the group is very small, sending a distinct packet to each receiver is expensive. On the other hand, broadcasting a packet is wasteful if the group consists of, say, 1000 machines on a million-node network, so that most receivers are not interested in the message (or worse yet, they are definitely interested but are not supposed to see it).

Thus, we need a way to send messages to well-defined groups that are numerically large in size but small compared to the network as a whole. Sending a message to such a group is called **multicasting**, and the routing algorithm used is called **multicast routing**.

All multicasting schemes require some way to create and destroy groups and to identify which routers are members of a group. Multicast routing schemes build on the broadcast routing schemes, sends packets along spanning trees to deliver the packets to the members of the group while making efficient use of bandwidth. However, the best spanning tree to use depends on whether the group is dense, with receivers scattered over most of the network, or sparse, with much of the network not belonging to the group. In this section we will consider both cases.
If the group is dense, broadcast is a good start because it efficiently gets the packet to all parts of the network. But broadcast will reach some routers that are not members of the group, which is wasteful. The solution explored by Deering and Cheriton is to prune the broadcast spanning tree by removing links that do not lead to members.
The result is an efficient multicast spanning tree.

As an example, consider the two groups, 1 and 2, in the network shown in Figure3.28 (a). Some routers are attached to hosts that belong to one or both of these groups, as indicated in the figure. A spanning tree for the leftmost router is shown in Figure3.28 (b). This tree can be used for broadcast but is overkill for multicast, as can be seen from the two pruned versions that are shown next. In figure3.28(c), all the links that do not lead to hosts that are members of group 1 have been removed. The result is the multicast spanning tree for the leftmost router to send to group 1. Packets are forwarded only along this spanning tree, which is more efficient than the broadcast tree because there are 7 links instead of 10. Figure3.28 (d) shows the multicast spanning tree after pruning for group 2. It is efficient too, with only five links this time. It also shows that different multicast groups have different spanning trees.

Various ways of pruning the spanning tree are possible. The simplest one can be used if link state routing is used and each router is aware of the complete topology, including which hosts belong to which groups. Each router can then construct its own pruned spanning tree for each sender to the group in question by constructing a sink tree for the sender as usual and then removing all links that do not connect group members to the sink node. **MOSPF** (**Multicast OSPF**) is an example of a link state protocol that works in this way.

With distance vector routing, a different pruning strategy can be followed. The basic algorithm is reverse path forwarding. However, whenever a router with no hosts interested in a particular group and no connections to other routers receives a multicast message for that group, it responds with a PRUNE message, telling the neighbor that sent the message not to send it any more multicasts from the sender for that group. When a router with no group members among its own hosts has received such messages on all the lines to which it sends the multicast, it, too, can respond with a PRUNE message. In this way, the spanning tree is recursively pruned. **DVMRP** (**Distance Vector Multicast Routing Protocol**) is an example of a multicast routing protocol that works this way.
Pruning results in efficient spanning trees that use only the links that are actually needed to reach members of the group. One potential disadvantage is that itis lots of work for routers, especially for large networks. Suppose that a network has *n* groups, each with an average of *m* nodes.

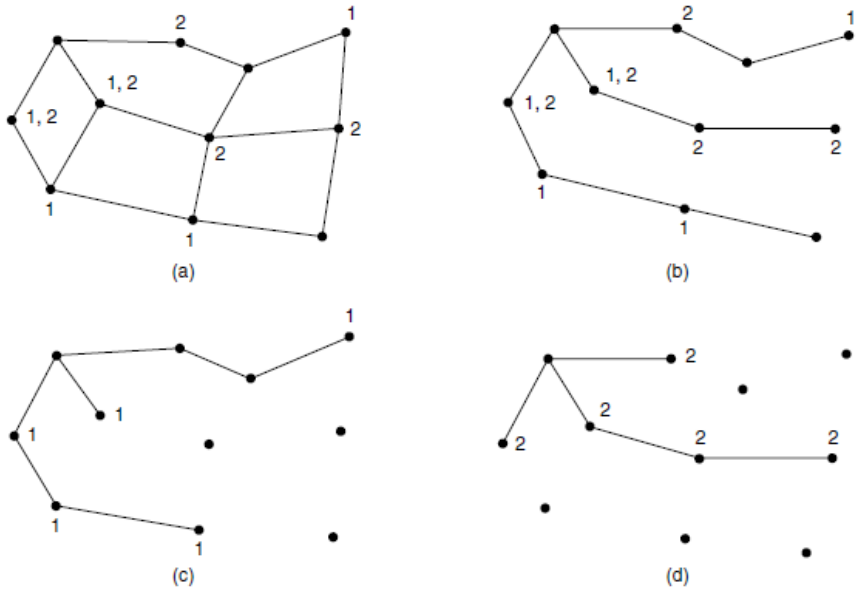


Figure 3.28. (a) A network. (b) A spanning tree for the leftmost router. (c) A multicast tree for group 1. (d) A multicast tree for group 2.

At each router and for each group, *m* pruned spanning trees must be stored, for a total of *mn* trees. For example, Figure 3.28(c) gives the spanning tree for the leftmost router to send to group 1. The spanning tree for the rightmost router to send to group 1 (not shown) will look quite different, as packets will head directly for group members rather than via the left side of the graph. This in turn means that routers must forward packets destined to group 1 in different directions depending on which node is sending to the group. When many large groups with many senders exist, considerable storage is needed to store all the trees.

An alternative design uses **core-based trees** to compute a single spanning tree for the group. All of the routers agree on a root (called the **core** or **rendezvous point**) and build the tree by sending a packet from each member to the root. The tree is the union of the paths traced by these packets. Figure3.29 (a) shows a core-based tree for group 1. To send to this group, a sender sends a packet to the core. When the packet reaches the core, it is forwarded down the tree. This is shown in figure3.29 (b) for the sender on the right hand side of the network. As a performance optimization, packets destined for the group do not need to reach the core before they are multicast. As soon as a packet reaches the tree, it can be forwarded up toward the root, as well as down all the other branches. This is the case for the sender at the top of figure3.29 (b).

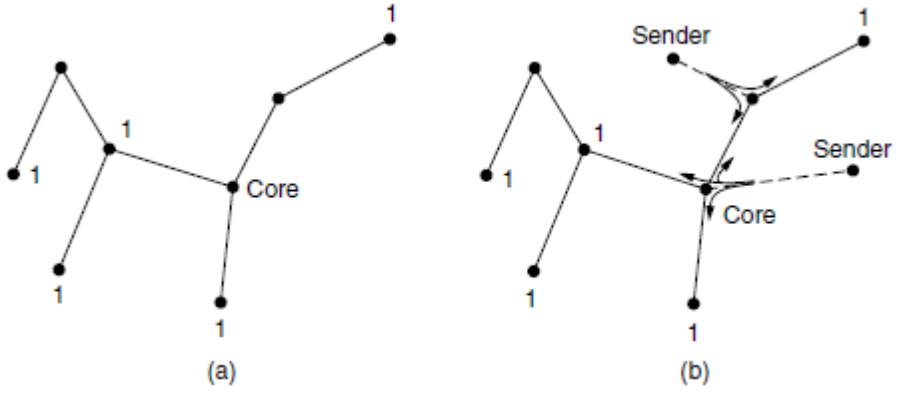


Figure 3.29: (a) Core-based tree for group 1. (b) Sending to group 1.

Having a shared tree is not optimal for all sources. For example, in figure3.29 (b), the packet from the sender on the right hand side reaches the top-right group member via the core in three hops, instead of directly. The inefficiency depends on where the core and senders are located, but often it is reasonable when the core is in the middle of the senders. When there is only a single sender, as in a video that is streamed to a group, using the sender as the core is

optimal. Also of note is that shared trees can be a major savings in storage costs, messages sent, and computation. Each router has to keep only one tree per group, instead of *m* trees. Further, routers that are not part of the tree do no work at all to support the group. For this reason, shared tree approaches like core-based trees are used for multicasting to sparse groups in the Internet as part of popular protocols such as **PIM** (**Protocol Independent Multicast**).

UNIT-IV

Illustrate and explain UDP and its packet format.

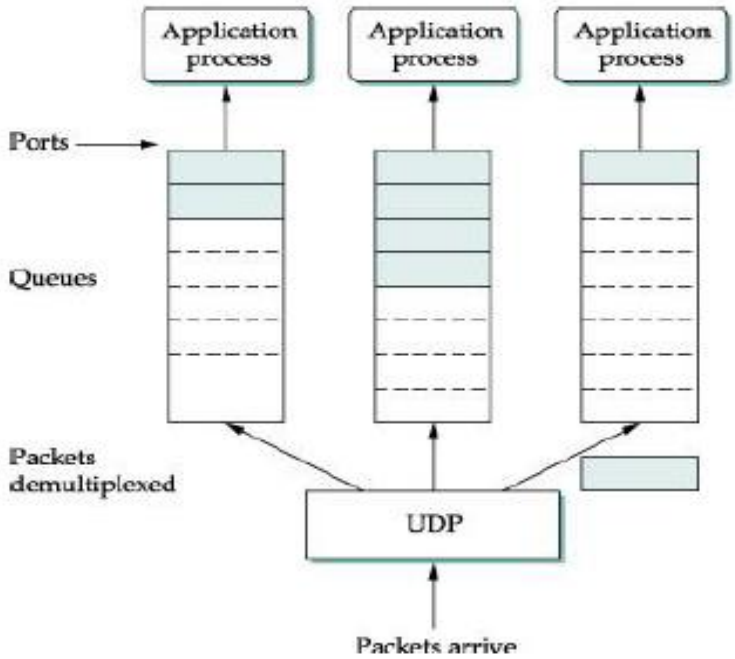
 User Datagram Protocol (UDP) is a connectionless, unreliable transport protocol.
 It does not add anything to the services of IP except process-to-process communication.
 UDP is a simple multiplexer/demultiplexer that allow multiple processes on each host to share the network.
 UDP does not implement flow control or reliable/ordered delivery.
 UDP ensures correctness of the message by the use of a checksum.
 If a process wants to send a small message and does not require reliability, UDP is used.

Port Number

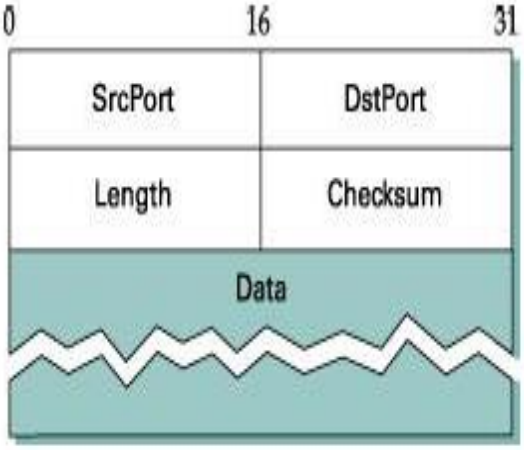
 Each process is assigned a unique 16-bit port number on that host.
 Processes are identified by (host, port) pair.
 Processes can be classified as either as *client* / *server*.
    - o Client process usually initiates exchange of information with the server
    - o Server process is identified by a well-known port number (0 – 1023).
    - o Client process is assigned an ephemeral port number (49152 – 65,535) by the OS.
    - o Some well known UDP ports are:

Port	Protocol
7	Echo
13	Daytime
53	DNS
111	RPC
161	SNMP

 Ports are usually implemented as a message queue.
    - o When a message arrives, UDP appends the message to the end of the queue.
    - o When queue is full, the message is discarded.
    - o When a message is read, it is removed from the queue.
    - o When queue is empty the process is blocked



UDP Header



 UDP packets, called user datagrams, have a fixed-size header of 8 bytes.
  SrcPort and *DstPort*—Contains port number for both the sender (*source*) and receiver (*destination*) of the message.
  Length—This 16-bit field defines total length of the user datagram, header plus data.
The total length is less than 65,535 bytes as it is encapsulated in an IP datagram.
UDP length = IP length - IP header's length
  Checksum—It is computed over pseudo header, UDP header and message content to ensure that message is correctly delivered to the exact recipient.

- The *pseudoheader* consists of three fields from the IP header (protocol number i.e., *17*, source and destination IP address), plus the UDP length field.

Bring out the classification of port numbers.
  Well-known ports range from 0 to 1023 are assigned and controlled by IANA.
  Registered ports range from 1024 to 49,151 are not assigned or controlled by IANA.

They can only be registered with IANA to prevent duplication.
 Ephemeral (dynamic) ports range from 49,152 to 65,535 is neither controlled nor registered. It is usually assigned to a client process by the operating system.

Distinguish between network and transport layer

Network layer	*Transport layer*
The network layer is responsible for *host*-to-*host* delivery	The transport layer is responsible for *process*-to-*process* delivery of a packet
Host address is required for delivery	Host address and port number is required for delivery
Error detection is not offered	Error detection is done using checksum
Flow control is not done	Flow control is not done
Multicasting capability is not inbuilt	Multicasting is embedded into UDP

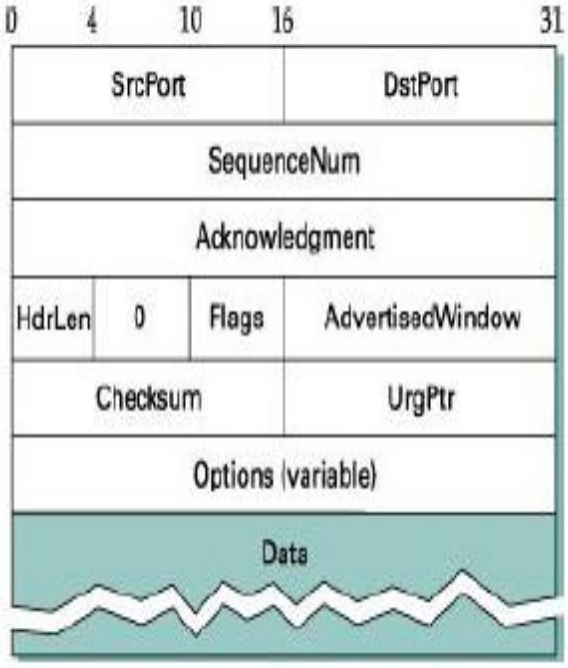
List some applications of UDP.
 UDP is used for management processes such as SNMP.
 UDP is used for some route updating protocols such as RIP.
 UDP is a suitable transport protocol for multicasting.
 UDP is suitable for a process with internal flow and error control mechanisms such as Trivial File Transfer Protocol (TFTP).

With a neat architecture, explain TCP in detail.
 Transmission Control Protocol (TCP) offers a reliable, connection-oriented, byte stream service
 TCP guarantees the reliable, in-order delivery of a stream of bytes
 It is a full-duplex protocol
 TCP supports demultiplexing mechanism for process-to-process communication.
 TCP has built-in congestion-control mechanism, i.e., sender is prevented from overloading the network.

Process-to-Process Communication
 Like UDP, TCP provides process-to-process communication. A TCP connection is identified a 4-tuple (SrcPort, SrcIPAddr, DstPort, DstIPAddr).
 Some well-known port numbers used by TCP are

Port	Protocol
23	TELNET
25	SMTP
80	HTTP

Segment Format

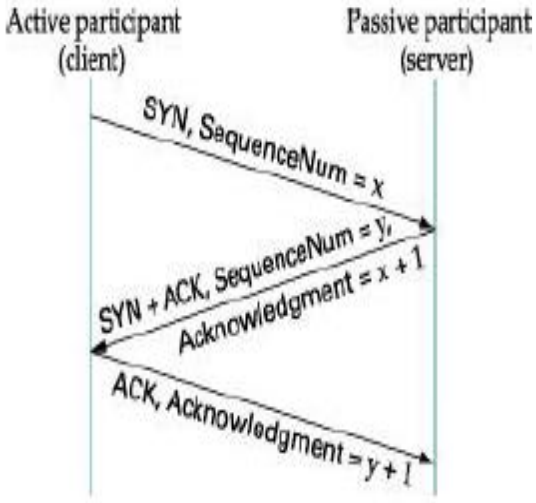


 TCP is a byte-oriented protocol, i.e. the sender writes bytes into a TCP connection and the receiver reads bytes out of the TCP connection.
 TCP groups a number of bytes together into a packet called *segment* and adds a header onto each segment. Segment is encapsulated in a IP datagram and transmitted.
 SrcPort and DstPort fields identify the source and destination ports.
 SequenceNum field contains sequence number, i.e. first byte of data in that segment.
 Acknowledgment defines byte number of the segment, the receiver expects next.
 HdrLen field specifies the number of 4-byte words in the TCP header.
 Flags field contains six control bits or flags. They are set to indicate:
o *URG*—indicates that the segment contains urgent data.
o *ACK*—the value of acknowledgment field is valid.
o *PSH*—indicates sender has invoked the push operation.
o *RESET*—signifies that receiver wants to abort the connection.
o *SYN*—synchronize sequence numbers during connection establishment.
o *FIN*—terminates the connection
 AdvertisedWindow field defines the receiver window and acts as flow control.
 Checksum field is computed over the TCP header, the TCP data, and pseudoheader.
 UrgPtr field indicates where the non-urgent data contained in the segment begins.
 Optional information (max. 40 bytes) can be contained in the header.

Connection Establishment

The connection establishment in TCP is called *three-way handshaking* as shown below:

1. The client (active participant) sends a segment to the server (passive participant) stating the initial sequence number it is to use (Flags = SYN, SequenceNum = *x*)
2. The server responds with a single segment that both acknowledges the client's sequence number (Flags = ACK, Ack = *x* + 1) and states its own beginning sequence number (Flags = SYN, SequenceNum = *y*).
3. Finally, the client responds with a segment that acknowledges the server's sequence number (Flags = ACK, Ack = *y* + 1).



State Transition Diagram

 The states involved in opening and closing a connection is shown above and below ESTABLISHED state respectively.

 The operation of sliding window (i.e., retransmission) is not shown.

 The two events that trigger a state transition is:

o a segment arrives from its peer.

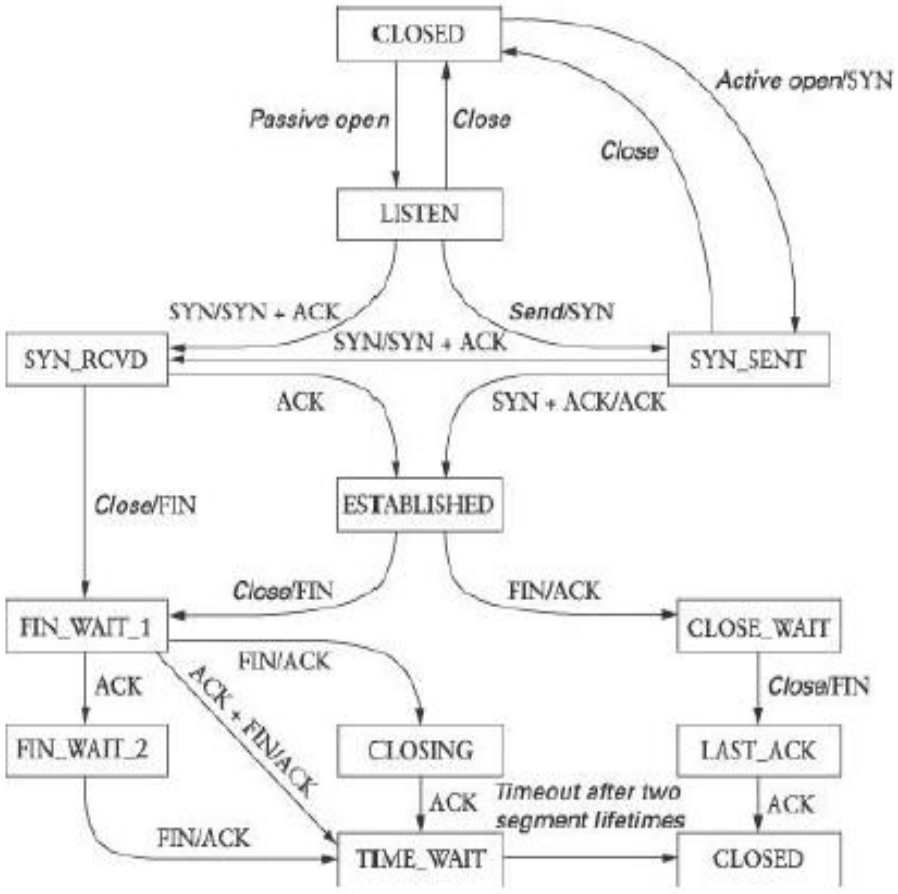
o the local application process invokes an operation on TCP

 TCP's state transition diagram defines the semantics of both its *peer-to-peer* interface and its *service* interface.

The state transition involved in opening a connection is as follows:

1. The server first invokes a *passive* open on TCP, which causes TCP to move to LISTEN state
2. Later, the client does an *active* open, which causes its end of the connection to send a SYN segment to the server and to move to the SYN_SENT state.
3. When the SYN segment arrives at the server, it moves to SYN_RCVD state and responds with a SYN + ACK segment.
4. The arrival of this segment causes the client to move to the ESTABLISHED state and to send an ACK back to the server.
5. When this ACK arrives, the server finally moves to the ESTABLISHED state.

a. Even if the client's ACK gets lost, sever will move to ESTABLISHED state when the first data segment from client arrives.



In TCP, the application process on both sides of the connection can independently close its half of the connection. The combinations of transitions from the ESTABLISHED state to CLOSED state are:

□ ESTABLISHED□ □ FIN_WAIT_1□ □ FIN_WAIT_2□ □ TIME_WAIT□ □ CLOSED (this side closes first)

□ ESTABLISHED□□ CLOSE_WAIT□□ LAST_ACK□□ CLOSED (other side closes first)

□ ESTABLISHED□□ FIN_WAIT_1□□ CLOSING□□ TIME_WAIT□□ CLOSED (both side close simultaneously)

Connection Termination

Three-way Handshaking—Most implementation follow three-way handshaking as shown.

1. The client TCP after receiving a Close command from the client process sends a FIN segment. A FIN segment can include the last chunk of data.
2. The server TCP responds with FIN + ACK segment to inform its closing.
3. The client TCP finally sends an ACK segment.

Four-way Half-Close—In TCP, one end can stop sending data while still receiving data, known as *half-close*. For instance, submit its data to the server initially for processing and close its connection. At a later time, the client receives the processed data from the server.

1. The client TCP half-closes the connection by sending a FIN segment.
2. The server TCP accepts the half-close by sending the ACK segment. The data transfer from the client to the server stops.

3. The server can send data to the client and acknowledgement can come from the client.
4. When the server has sent all the processed data, it sends a FIN segment to the client.
5. The FIN segment is acknowledged by the client.

Write short notes on urgent data in TCP?

 TCP is a stream-oriented protocol, i.e., each byte of data has a position in the stream.
 At times an application may need to send urgent data, i.e., sending process wants a piece of data to be read out of order by the receiving process. For example, to abort the process by issuing Ctrl + C keystroke.
 The above scenario is handled by setting the URG bit.
 The sending TCP inserts the urgent data at beginning of the segment.
 The urgent pointer field in the header defines start of normal data.
 When the receiving TCP receives a segment with the URG bit set, it delivers urgent data out of order to the receiving application.

What is push operation in TCP?

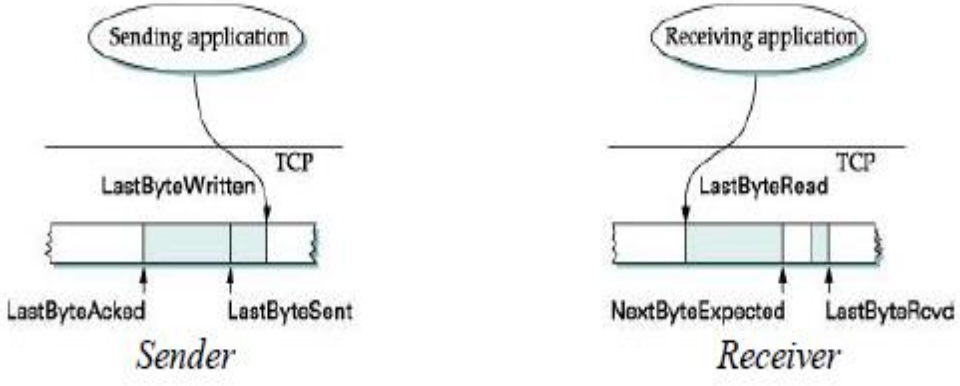
 The receiving TCP buffers the data when they arrive and delivers them to the application program when ready or when it is convenient for the receiving process.
 In case of interactive applications, delayed delivery of data is not acceptable.
 TCP handles as follows:

o The application program at the sending site can request a Push operation.
o This instructs the sending TCP not to wait for the window to be filled. It must create a segment and send it immediately.
o The sending TCP also sets the push bit (PSH) to let the receiving TCP know that the segment includes data that must be delivered to the receiving application program as soon as possible and not to wait for more data to come.

Explain TCPs adaptive control and its uses.

 TCP uses a variant of sliding window known as adaptive flow control that:
    - o guarantees the reliable delivery of data in ordered manner
    - o enforces flow control at the sender
 The receiver advertises a window size to the sender using AdvertisedWindow field in the TCP header
 The sender cannot have unacknowledged data greater than value of AdvertisedWindow

Reliable and Ordered Delivery



Sender Receiver

 TCP on the sending side maintains a *send* buffer that is divided into 3 segments namely acknowledged data, unacknowledged data and data to be transmitted

 Similarly TCP on the receiving side maintains a *receive* buffer to hold data even if it arrives of order.

 The send buffer maintains three variables namely LastByteAcked, LastByteSent, and LastByteWritten as shown above. The relation between them is obviousLastByteAcked ≤ LastByteSent and LastByteSent ≤ LastByteWritten

 The bytes to the left of LastByteAcked are not kept as it had been acknowledged.

 The receive buffer maintains three variables namely LastByteRead, NextByteExpected, and LastByteRcvd. The relation between them is LastByteRead < NextByteExpected and NextByteExpected ≤LastByteRcvd + 1

 If data are received in order, NextByteExpected is the next byte after LastByteRcvd

 Bytes to the left of LastByteRead is not buffered as it has been read by the application

Flow Control

 The capacity of *send* and *receiver* buffer is MaxSendBuffer and MaxRcvBuffer respectively.

 The sending TCP prevents overflowing of its buffer by maintaining

LastByteWritten - LastByteAcked ≤ MaxSendBuffer

 The receiving TCP avoids overflowing its receive buffer by maintaining

LastByteRcvd - LastByteRead ≤ MaxRcvBuffer

 The receiver throttles the sender by advertising a window that is no larger than the amount of *free* space that it can buffer as

AdvertisedWindow = MaxRcvBuffer - ((NextByteExpected - 1) ? LastByteRead)

 When data arrives, the receiver acknowledges it as long as preceding bytes have arrived.

- LastByteRcvd moves to its right (incremented), and the advertised window shrinks

 The advertised window expands when the data is read by the application

- It data is read as fast as it arrives then AdvertisedWindow = MaxRcvBuffer
- If it is read slow, it eventually leads to a AdvertisedWindow of size 0.

 The sending TCP adheres to the advertised window by computing *effective* window, that limits how much data it should send as

EffectiveWindow = AdvertisedWindow - (LastByteSent -LastByteAcked)

 When a acknowledgement arrives for *x* bytes, LastByteAcked is incremented by *x* and the buffer space is freed accordingly

Fast Sender vs. Slow Receiver

 A slow receiver prevents being swamped with data from a fast receiver by using AdvertisedWindow field

 Initially the fast sender transmits at a higher rate.

 The receiver's buffer gets filled up. Hence, AdvertisedWindow shrinks, eventually to 0.

 When the receiver advertises window of size 0, sender cannot transmit any further data. Therefore, the TCP at the sender blocks the sending process.

 When the receiving process reads some data, those bytes are acknowledged. Thus the AdvertisedWindow expands.

 The LastByteAcked is incremented and buffer space is freed to that extent,

 The sending process becomes unblocked and is allowed to fill up the free space.

Checking AdvertisedWindow status

 TCP always sends a segment in response that contains the latest values for the Acknowledge and AdvertisedWindow fields, even if these values have not changed.

 Thus the sender can come to know the status of AdvertisedWindow even after the receiver advertises a window of size 0.

AdvertisedWindow

 The TCP's AdvertisedWindow field is 16 bits long, half the size of SequenceNum

 The length of 16-bits ensures that it does not wrap around

 The length of AdvertisedWindow is designed such that it allows the sender to keep the pipe full.

 The 16-bit length also accounts for product of delay × bandwidth. It is not big enough, in case of a T3 connection, but taken care by TCP extension headers.

What is adaptive retransmission? Explain the algorithms used?

 TCP guarantees reliability through retransmission.

- o Retransmission due to timeout before ACK.
- o Timeout is a function of RTT.
- o RTT is highly variable between any two hosts on the internet.
- o Appropriate timeout is chosen using adaptive retransmission.

Original Algorithm

 TCP estimates SampleRTT by computing the duration between sending of a packet and arrival of its ACK.

 TCP then computes EstimatedRTT as a weighted average between the previous and current estimate as

$$\text{EstimatedRTT} = \alpha \times \text{EstimatedRTT} + (1 - \alpha) \times \text{SampleRTT}$$

where α is the smoothening factor and its value is in the range 0.8–0.9

 Timeout is twice the EstimatedRTT

$$\text{TimeOut} = 2 \times \text{EstimatedRTT}$$

Karn/Partridge Algorithm

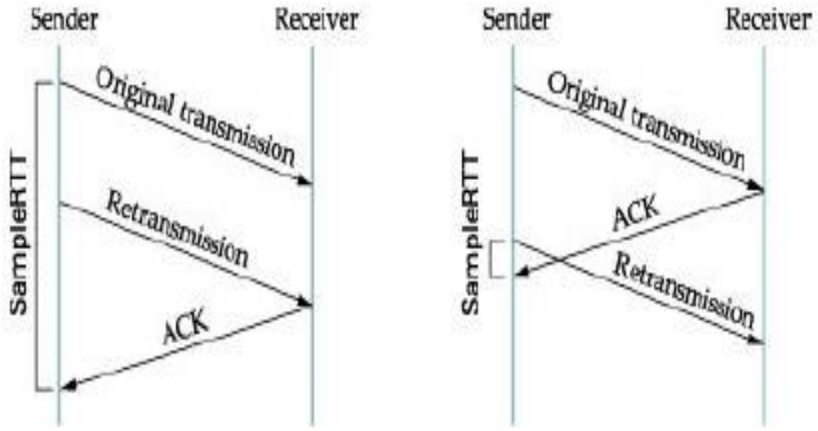
 The flaw discovered in original algorithm after years of use is

- o whether an ACK should be associated with the original or retransmission segment
- o If ACK is associated with original one, then SampleRTT becomes too large
- o If ACK is associated with retransmission, then SampleRTT becomes too small

 Karn/Partridge proposed a solution to the above by making changes to the timeout mechanism.

 Each time TCP retransmits, it sets the next timeout to be twice the last timeout.

- o Loss of segments is mostly due to congestion and hence TCP source does not react aggressively to a timeout.



Jacobson/Karels Algorithm

 The main problem with original algorithm is that variance of the sample RTTs is not taken into account.

- o if variation among samples is small, then EstimatedRTT can be trusted
- o otherwise timeout should not be tightly coupled with the EstimatedRTT

 In this new approach, the sender measures a new SampleRTT as before.

 The Deviation amongst RTTs is computed as follows:

Difference = SampleRTT - EstimatedRTT

EstimatedRTT = EstimatedRTT + (δ × Difference)

Deviation = Deviation + (|Difference| ? Deviation)

where δ is a fraction between 0 and 1

 TCP now computes TimeOut as a function of both EstimatedRTT and Deviation as listed:

TimeOut = μ × EstimatedRTT + ϕ × Deviation

where μ = 1 and ϕ = 4 usually

 When variance is small, difference between TimeOut and EstimatedRTT is negligible.

 When variance is larger, Deviation plays a greater role in deciding TimeOut.

What is silly window syndrome? When should TCP transmit a segment?

 When an ACK arrives, the window enlarges for transmission.

 Even if window size is less than one MSS, TCP decides to go ahead and transmit a half-full segment.

 The strategy of aggressively taking advantage of any available window leads to a situation now known as the *silly window syndrome*.

 If the sender aggressively fills, then any small segments introduced into the system remains in the system indefinitely as it does not combine with adjacent segments to create larger ones as shown.

Nagle's Algorithm

Nagle's suggests a solution as to what the sending TCP should do when there is data to send and window size is less than one MSS. The algorithm is listed below:

When the application produces data to send

if both the available data and the window ≥ MSS

send a full segment
else
if there is unACKed data in flight
buffer the new data until an ACK arrives
else
send all the new data now

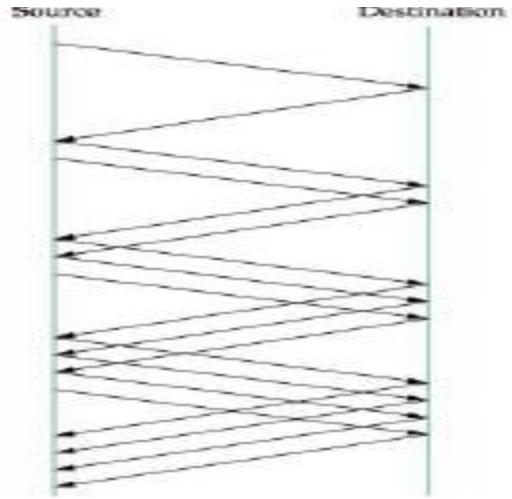
 It's always OK to send a full segment if the window allows.
 It's also OK to immediately send a small amount of data if there are currently no segments in transit, but if there is anything in flight, the sender must wait for an ACK before transmitting the next segment.

Explain TCP congestion control techniques in detail.

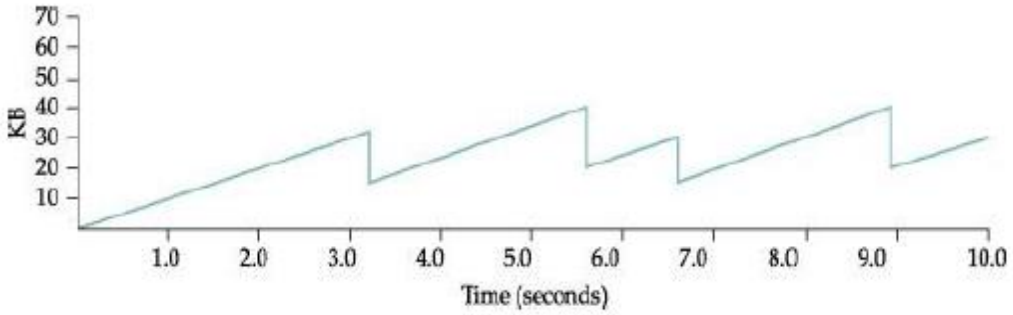
 In TCP congestion control, each source has to determine the available capacity in the network, so that it can send packets without loss.
 By using ACKs to pace transmission of packets, TCP is said to be *self-clocking*.
 TCP maintains a state variable CongestionWindow for each connection. Therefore
MaxWindow = MIN(CongestionWindow, AdvertisedWindow)
EffectiveWindow = MaxWindow - (LastByteSent - LastByteAcked)
 Thus, a TCP source is allowed to send no faster than *network* or *destination* host
 The problem is that available bandwidth changes over time. The three congestion control mechanism are:
o Additive Increase/Multiplicative Decrease
o Slow Start
o Fast Retransmit and Fast Recovery

Additive Increase/Multiplicative Decrease (AIMD)

 TCP source sets the CongestionWindow based on the level of congestion it perceives to exist in the network.
 The additive increase/multiplicative decrease (AIMD) mechanism works as follows:
o The source increases CongestionWindow when level of congestion goes down and decreases CongestionWindow when level of congestion goes up.
 TCP interprets timeouts as a sign of congestion and reduces the rate at which it is transmitting.
o Each time a timeout occurs, the source sets CongestionWindow to half of its previous value. This is known as *multiplicative decrease*.
o For example, if CongestionWindow is set to 16 packets, after a packet loss, it is set to 8.
o The CongestionWindow is not allowed to fall below one packet size or MSS, irrespective of the level of congestion.
 Every time, the source successfully sends one packet, CongestionWindow is increased by a fraction (*additive increase*).
o An ACK acknowledges receipt of MSS bytes, the increment is computed as
Increment = MSS × (MSS/CongestionWindow)
CongestionWindow += Increment



 This pattern of continually increasing and decreasing the congestion window continues throughout the lifetime of the connection

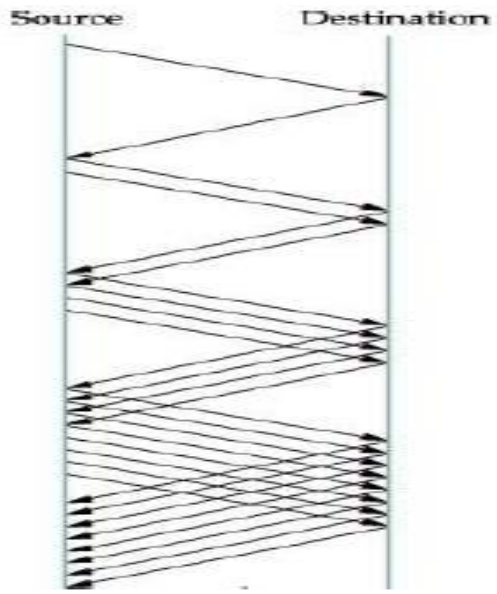


 When the current value of CongestionWindow as a function of time, it results as a saw-tooth pattern.
 AIMD decreases its CongestionWindow aggressively but increases conservatively.
- o Having small CongestionWindow only results in less probability of packets being dropped.
- o Thus congestion control mechanism becomes stable.

 Since timeout is an indication of congestion that triggers multiplicative decrease, TCP needs the most accurate timeout mechanism.
 AIMD is appropriate only when source is operating close to network capacity.

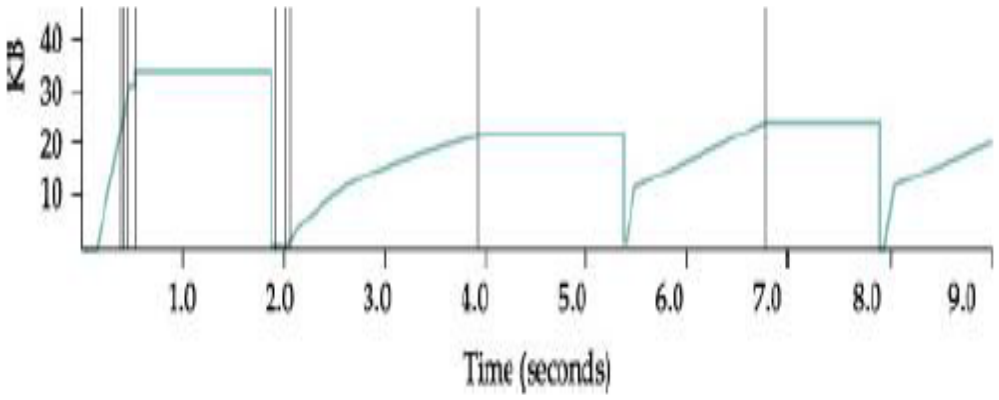
Slow Start

 Slow start increases the congestion window exponentially, rather than linearly. It is usually used from cold start.
 The source starts by setting CongestionWindow to one packet.
- o When ACK arrives, TCP adds 1 to CongestionWindow and sends two packets.
- o Upon receiving two ACKs, TCP increments CongestionWindow by 2 and sends four packets.
- o Thus TCP doubles the number of packets every RTT as shown.



 Slow start provides exponential growth and is designed to avoid bursty nature of TCP.
 Initially TCP has no idea about congestion, henceforth it increases CongestionWindow rapidly until there is a packet loss.
 When a packet is lost:
- o TCP immediately decreases CongestionWindow by half (*multiplicative decrease*).
- o It stores the current value of CongestionWindow as CongestionThreshold and resets to CongestionWindow one packet
- o The CongestionWindow is incremented one packet for each ACK arrived until it reaches CongestionThreshold and thereafter one packet per RTT.

 In initial stages, TCP loses more packets because it attempts to learn the available bandwidth quickly through exponential increase
 An alternate strategy to slow start is known as *packet pair*
- o Send packets without space and then observe timings of their ACKs.
- o The difference between ACKs is taken as a measure of congestion in the network



In the above example, initial slow start causes increase in CongestionWindow (34KB). The trace then flattens at 2 sec due to loss of packets. CongestionThreshold is set to 17KB (34/2) and CongestionWindow to 1 packet. Thereafter additive increase is followed

Fast Retransmit and Fast Recovery

 Fast retransmit is a heuristic that triggers the retransmission of a dropped packet sooner than the regular timeout mechanism. It does not replace regular timeouts.

 When a packet arrives out of order, the receiving TCP resends the same acknowledgment (*duplicate ACK*) it sent the last time.

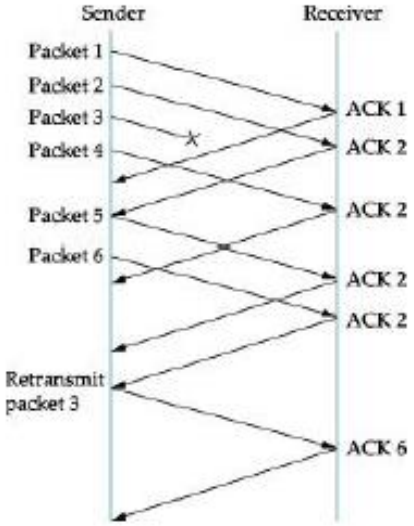
 The sending TCP waits for three duplicate ACK, to confirm that the packet is lost before retransmitting the lost packet. This is known as *fast retransmit* and it signals congestion.

 Instead of setting CongestionWindow to one packet, this method uses the ACKs that are still in the pipe to clock the sending of packets. This mechanism is called *fast recovery*.

 The fast recovery mechanism removes the slow start phase and follows additive increase.

 The fast retransmit/recovery results increase in throughput by 20%.

The following example shows transmission of packets in which the third packet gets lost. The sender on receiving three duplicate ACKs (ACK 2) retransmits the third packet as shown below. On receiving the lost one, the receiver acknowledges the packet with highest number.



In this strategy:

 Slow start is only used at the beginning of a connection and when the regular timeout occurs.

 At other times, the congestion window follows a pure additive increase/multiplicative decrease pattern

 TCP's fast retransmit can detect up to three dropped packets per window.

Explain in detail about TCP congestion avoidance algorithms.

 Congestion avoidance refers to mechanisms that prevent congestion before it actually occurs.

 TCP increases the load and when congestion is likely to occur, it decreases load on the network.

 TCP creates loss of packets in order to determine bandwidth of the connection

 The three congestion-avoidance mechanisms are:

o DECbit

o Random Early Detection (RED)

o Source-based congestion avoidance

DECbit

 Was developed for use on Digital Network Architecture

 In DEC bit, each router monitors the load it is experiencing and explicitly notifies the end nodes when congestion is about to occur by setting a binary congestion bit called **DECbit** in packets that flow through it.

 The destination host copies the DECbit into the ACK and sends back to the source.

 Eventually the source reduces its transmission rate and congestion is avoided.

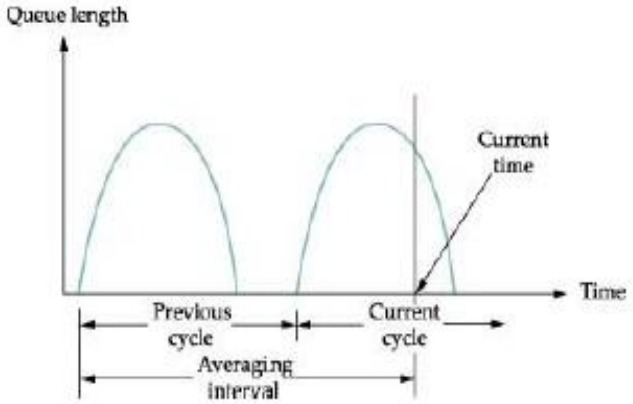
Algorithm

 A single congestion bit is added to the packet header.

 A router sets this bit in a packet if its average queue length is 1 when the packet arrives.

 The average queue length is measured over a time interval that spans the last busy + last idle cycle + current busy cycle as shown below.

 Router calculates average queue length by dividing the curve area by time interval



 The source computes how many ACK has the DECbit set for the previous window packets it has sent.

o If less than 50% of the packets had its DECbit set, then source increases its congestion window by 1 packet.

o Otherwise, source decrease the congestion window by 87.5% (multiply its previous value by 0.875)

 "Increase by 1, decrease by 0.875" rule is its additive increase/multiplicative decrease strategy.

Random Early Detection (RED)

 Proposed by Floyd and Jackson

 Each router monitors its own queue length.

 In RED, router implicitly notifies the source that congestion is likely to occur by dropping one of its packets.

 The source is notified by timeout or duplicate ACK.

 The router drops a few packets earlier before it runs out of space, so that it need not drop more packets later.

 Each incoming packet is dropped with a probability known as *drop probability* when the queue length exceeds *drop level*.

Algorithm

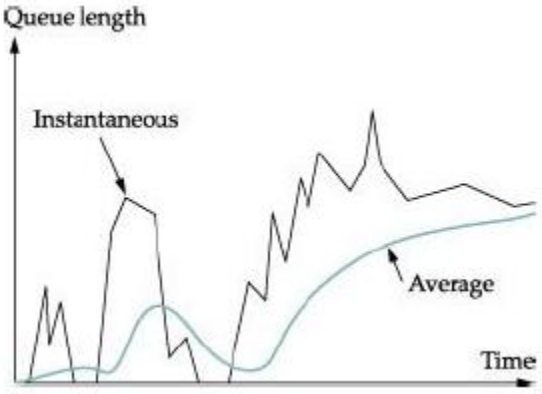
 RED computes average queue length using a weighted running average as follows:

AvgLen = (1 Weight) × AvgLen + Weight × SampleLen

o where 0 < Weight < 1 and SampleLen is length of the queue when a sample measurement is made.

o Because of the bursty nature of Internet traffic, queues can become full very quickly and then empty again.

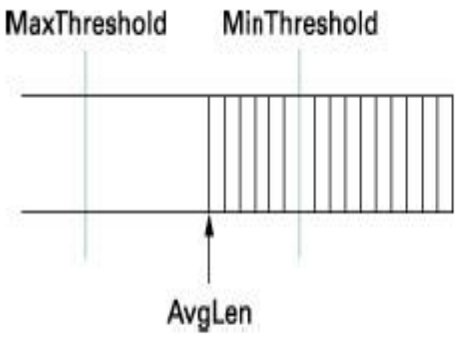
o The weighted running average detects long-lived congestion as shown below



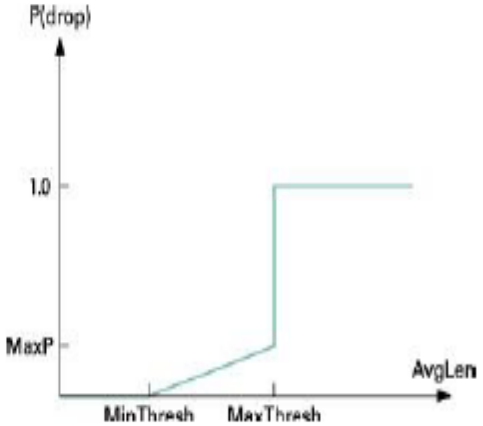
 RED has two queue length thresholds MinThreshold and MaxThreshold. When a packet arrives at the gateway, RED compares the current AvgLen with these thresholds and decides whether to queue or drop the packet as follows:

```
if AvgLen  MinThreshold
queue the packet
if MinThreshold < AvgLen < MaxThreshold
calculate probability P
drop the arriving packet with probability P
if MaxThreshold  AvgLen
drop the arriving packet
```

o The probability of drop increases slowly when AvgLen is between the two thresholds, reaching MaxP at the upper threshold, at which point it jumps to unity as shown.



RED thresholds



Drop probability function

RED thresholds Drop probability function

o P is a function of both AvgLen and how long it has been since the last packet was dropped. It is computed as

TempP = MaxP × (AvgLen - MinThreshold)/(MaxThreshold - MinThreshold)

P = TempP/(1 - count × TempP)

 Because RED drops packets randomly, the probability that RED decides to drop a flow's packet(s) is roughly proportional to the share of the bandwidth for that flow.

 MaxThreshold is set to twice of MinThreshold as it works well for the Internet traffic.

 There should be enough free buffer space above MaxThreshold to absorb bursty traffic.

Source-Based Congestion Avoidance

 The source looks for signs of congestion on the network, for example, a considerable increase in the RTT, indicate queuing at a router.

Some mechanisms

1. Every two round-trip delays, it checks to see if the current RTT is greater than the average of the minimum and maximum RTTs.

a. If it is, then the algorithm decreases the congestion window by one-eighth.

b. Otherwise the normal increase as in TCP.

2. The window is adjusted once every two round-trip delays based on the product

(CurrentWindow - OldWindow) × (CurrentRTT - OldRTT)

a. If the result is positive, the source decreases the window size by one-eighth

b. Otherwise, the source increases the window by one maximum packet size.

3. Every RTT, it increases the window size by one packet and compares the throughput achieved to the throughput when the window was one packet smaller.

a. If the difference is less than one-half the throughput achieved when only one packet was in transit, it decreases the window by one packet.

TCP Vegas

 In standard TCP, it was observed that throughput increases as congestion window increases, but not beyond the available bandwidth.

 Any further increase in the window size only results in packets taking up buffer space at the bottleneck router

 TCP Vegas uses this idea to measure and control the right amount of extra data in transit.

 If a source is sending too much extra data, it will cause long delays and possibly lead to congestion.

 TCP Vegas's congestion-avoidance actions are based on changes in the estimated amount of extra data in the network.

 A flow's BaseRTT is set to the minimum of all RTTs and is mostly the first packet sent.

 The expected throughput is given by ExpectedRate = CongestionWindow/BaseRTT

 The sending rate, ActualRate is computed by dividing number of bytes transmitted during a RTT by that RTT.

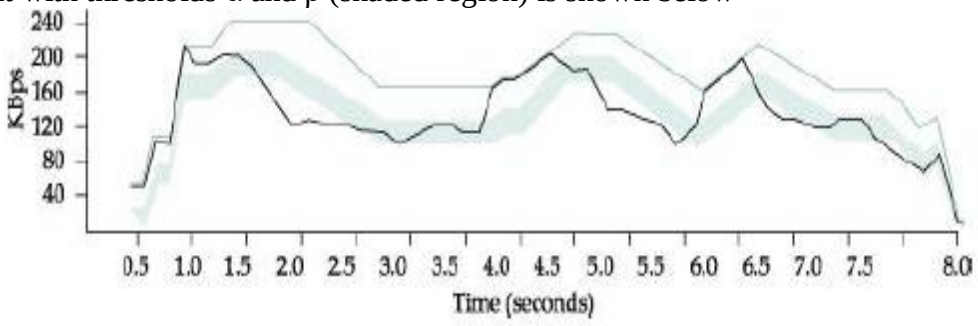
 The difference between two rates is computed, say Diff = ExpectedRate – ActualRate

 Two thresholds α and β are defined such that $\alpha < \beta$

o When Diff < α, the congestion window is linearly increased during the next RTT

o When Diff > β, the congestion window is linearly decreased during the next RTT

o When α < Diff < β, the congestion window is unchanged
 When actual and expected output varies significantly, the congestion window is reduced as it indicates congestion in the network.
 When actual and expected output is almost the same, the congestion window is increased to utilize the available bandwidth.
 The overall goal is to keep between α and β extra bytes in the network. The expected & actual throughput with thresholds α and β (shaded region) is shown below



What is meant by quality of service?
 QoS is defined as a set of attributes pertaining to the performance of a connection.
 The attributes may be either user or network oriented.
 QoS on the Internet can be broadly classified into
- o *Integrated Services* (IntSrv)
- o *Differentiated Services*

Explain how QoS is provided through integrated services.
 Integrated Services *IntSrv* is a flow-based QoS model, i.e., user creates flow from source to destination and informs all routers of the resource requirement.

Service Classes
 The two classes of service defined are:
- o *Guaranteed* service in which the network assures that delay will not be beyond some maximum if flow stays within TSpec.
- o *Controlled load* service meets the need of tolerant, adaptive applications which requests low-loss or no-loss such as file transfer, e-mail, etc.

Flowspec
 The set of information given to the network for a given flow is called *flowspec*. It has two parts namely
- o Tspec defines the traffic characterization of the flow
- o Rspec defines resources that the flow needs to reserve (buffer, bandwidth, etc.)

TSpec
 The bandwidth of real-time application varies constantly for most application.
 The average rate of flows cannot be taken into account as variable bit rate applications exceed the average rate. This leads to queuing and subsequent delay/loss of packets.

Token Bucket

 The solution to manage varying bandwidth is to use *token bucket* filter that can describe bandwidth characteristics of a source/flow.

 The two parameters used are token rate r and a bucket depth B

 A token is required to send a byte of data.

 A source can accumulate tokens at rate r/second, but not more than B tokens.

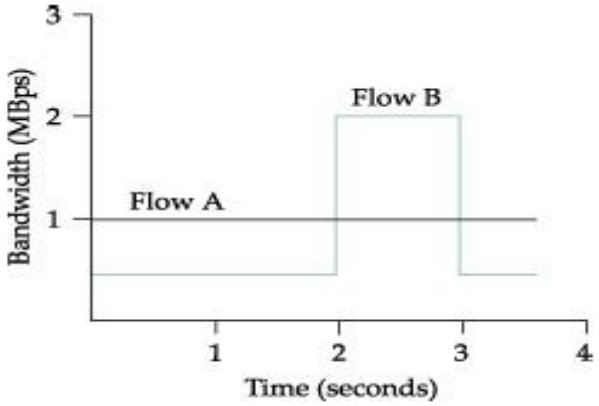
 Bursty data of more than r bytes per second is not permitted. Therefore bursty data should be spread over a long interval.

 The token bucket provides information that is used by admission control algorithm to determine whether or not to consider the new request for service.

The following example shows two flows with equal average rates but different token bucket descriptions.

 Flow A generates data at a steady rate of 1 Mbps, which is described using a token bucket filter with rate r = 1 Mbps and a bucket depth B = 1 byte.

 Flow B sends at rate of 0.5 Mbps for 2 seconds and then at 2 Mbps for 1 second, which is described using a token bucket filter with rate r = 1 Mbps and a bucket depth B = 1 MB. The additional depth allows it to accumulate tokens when it sends 0.5 Mbps (2 × 0.5 = 1 MB) and uses the same to send for bursty data of 2 Mbps.



Admission Control

 When a flow requests a level of service, admission control examines *TSpec* and *RSpec* of the flow.

 It checks to see whether the desired service can be provided with currently available resources, without causing any worse service to previously admitted flows.

- o If it can provide the service, the flow is admitted otherwise denied.

 The decision to allow/deny a service can be *heuristic* such as "currently delays are within bounds, therefore another service can be admitted."

 Admission control is closely related to *policy*. For example, a network admin will allow CEO to make reservations and forbid requests from other employees.

Reservation Protocol (RSVP)

 The Resource Reservation Protocol (RSVP) is a signaling protocol to help IP create a flow and make a resource reservation.

 RSVP provides resource reservations for all kinds of traffic including multimedia which uses multicasting. RSVP supports both unicast and multicast flows.

 RSVP is a robust protocol that relies on *soft state* in the routers.

- o Soft state unlike hard state (as in ATM, VC), times out after a short period if it is not refreshed. It does not require to be deleted.
- o The default interval is 30 ms.

 Since multicasting involves large number of receivers than senders, RSVP follows *receiver-oriented* approach that makes receivers to keep track of their requirements.

RSVP Messages

 To make a reservation, the receiver needs to know:

o What traffic the sender is likely to send so as to make an appropriate reservation, i.e., *TSpec*.

o Secondly, what path the packets will travel.

 The sender sends a PATH message to all receivers (*downstream*) containing *TSpec*.

 A PATH message stores necessary information for the receivers on the way. PATH messages are sent about every 30 seconds.

 The receiver sends a reservation request as a RESV message back to the sender (*upstream*), containing sender's *TSpec* and receiver requirement *RSpec*.

 Each router on the path looks at the RESV request and tries to allocate necessary resources to satisfy and passes the request onto the next router.

o If allocation is not feasible, the router sends an *error* message to the receiver

 If there is any failure in the link a new path is discovered between sender and the receiver. The RESV message follows the new path thereafter.

 A router reserves resources as long as it receives RESV message, otherwise released.

 If a router does not support RSVP, then best-effort delivery is followed.

Reservation Merging

 In RSVP, the resources are not reserved for each receiver in a flow, but merged.

 When a RESV message travels from receiver up the multicast tree, it is likely to come across a router where reservations have already been made for some other flow.

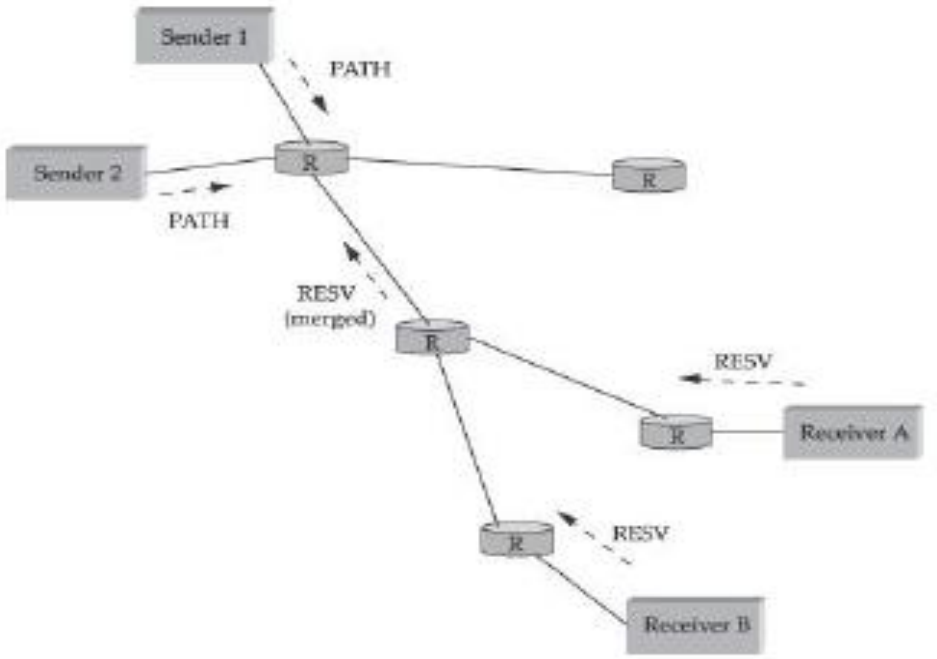
 If the new resource requirements can be met using existing allocations, then new allocations need not be made.

o For example, receiver *A* has already made a request for a guaranteed delay of less than 100 ms. If *B* comes with a new request for a delay of less than 200 ms, then no new reservations are made.

o Another example shows router R3 merging requests from Rc1, Rc2 and Rc3 before making bandwidth reservation.

 A router that handles multiple requests with one reservation is known as *merge point*. This is because, different receivers require different quality.

 Reservation merging meets the needs of all receivers downstream of the *merge point*.



Packet Classifying and Scheduling

 Packet classification refers to the process of associating each packet with corresponding reservation.

- o This is done by examining the fields *source address*, *destination address*, *protocol number*, *source port* and *destination port* in the packet header.

 Scheduling refers to the process of managing packets in queues to ensure that they get the requested service.

- o Weighted fair queuing or a combination of queuing disciplines can be used.

List the disadvantages of integrated services

 *Scalability*IntSrv requires router to maintain information for each flow, which is not feasible for today's internet growth

 *Service type limitation*Only two types of services are provided. Certain applications may require more than the offered services.

Explain how QoS is provided through differentiated services

Differentiated Services (DiffServ) is a class-based QoS model designed for IP.

Premium class

 The default *best-effort* model is enhanced as a new class called *premium*.

 The premium packets have bits set (*marked*) in the header by the organization gateway router or by the ISP router.

 IETF has defined a set of behaviors for routers known as per-hop behaviors (*PHB*).

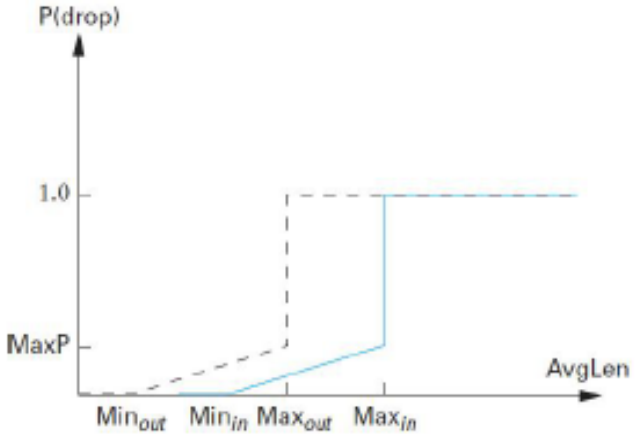
 IETF has replaced the existing TOS field in IPv4 or Class field in IPv6 with 6-bit DiffServ code points (*DSCP*) and remaining 2 bits unused.

 6-bit DSCP can be used to define 64 PHB that could be applied to a packet.

 The three PHBs defined are *default* PHB (DE PHB), *expedited forwarding* PHB (EF PHB) and *assured forwarding* PHB (AF PHB).
 The DE PHB is the same as best-effort delivery and is compatible with TOS

Expedited Forwarding (EF PHB)
 Packets marked for EF treatment should be forwarded by the router with minimal delay (*latency*) and loss by ensuring required bandwidth.
 A router guarantees EF, only if arrival rate of EF packets is less than forwarding rate
 The rate limiting of EF packets is achieved by configuring routers at the edge of an administrative domain to ensure that it is less than bandwidth of the slowest link.
 Queuing can be either using strict priority or weighted fair queuing.
- o In strict priority, EF packets are preferred over others, leaving less chance for other packets to go through.
- o In weighted fair queuing, other packets are given a chance, but there is a possibility of EF packets being dropped, if there is excessive EF traffic.

Assured Forwarding
 The AF PHB is based on RED with In and Out (*RIO*) algorithm.
 In RIO, the drop probability increases as the average queue length increases.
The following example shows RIO with two classes named *in* and *out*.



 The *out* curve has a lower MinThreshold than *in* curve, therefore under low levels of congestion, only packets marked *out* will be discarded.
 If the average queue length exceeds Min*in*, packets marked *in* are also dropped.
 The terms *in* and *out* are explained with the example "Customer X is allowed to send up to *y* Mbps of assured traffic".
- o If the customer sends packets less than *y* Mbps then packets are marked *in*.
- o When the customer exceeds *y* Mbps, the excess packets are marked *out*.

 Thus combination of profile meter at the edge router and RIO in all routers, assures (*but does not guarantee*) the customer that packets within the profile will be delivered
 RIO does not change the delivery order of *in* and *out* packets.
 If weighted fair queuing is used, then weight for the premium queue is chosen using the formula. It is based on the load of premium packets.

*B*premium = *W*premium / (*W*premium + *W*best-effort)

o For example, if weight of premium queue is 1 and best-effort is 4, then only 20% of the link is reserved for premium packets.

How differentiated services overcome the limitations of integrated services?

1. The main processing was moved from the core of the network to edge of the network (*scalability*). Thus routers need not store information about flows. The applications define the type of service they need each time when a packet is sent.
2. The per-flow service is changed to per-class service. The router routes the packet based on class of service defined in the packet, not the flow. Different types of classes (*services*) based on the needs of applications.

Write short notes on ATM QoS.

The five ATM service classes are:
1. constant bit rate (*CBR*)
2. variable bit rate—real-time (*VBR-rt*)
3. variable bit rate—non-real-time (*VBR-nrt*)
4. available bit rate (*ABR*)
5. unspecified bit rate (*UBR*)

Constant Bit Rate

- Sources of CBR traffic are expected to send at a constant rate.
- The source's peak rate and average rate of transmission are equal.
- CBR class is designed for customers who need real-time audio or video services.
- CBR is a relatively easy service for implementation

Variable Bit Rate

- The VBR class is divided into two subclasses: real-time (*VBR-rt*) and non-real-time (*VBR-nrt*).
- *VBR-rt* is designed for users who need real-time services (such as voice and video transmission) and use compression techniques to create a variable bit rate.
- The traffic generated by the source is characterized by a token bucket, and the maximum total delay required through the network is specified.
- *VBR-nrt* bears some similarity to IP's controlled load service. The source traffic is specified by a token bucket.
- *VBR-nrt* is designed for users who do not need real-time services but use compression techniques to create a variable bit rate

Unspecified Bit Rate

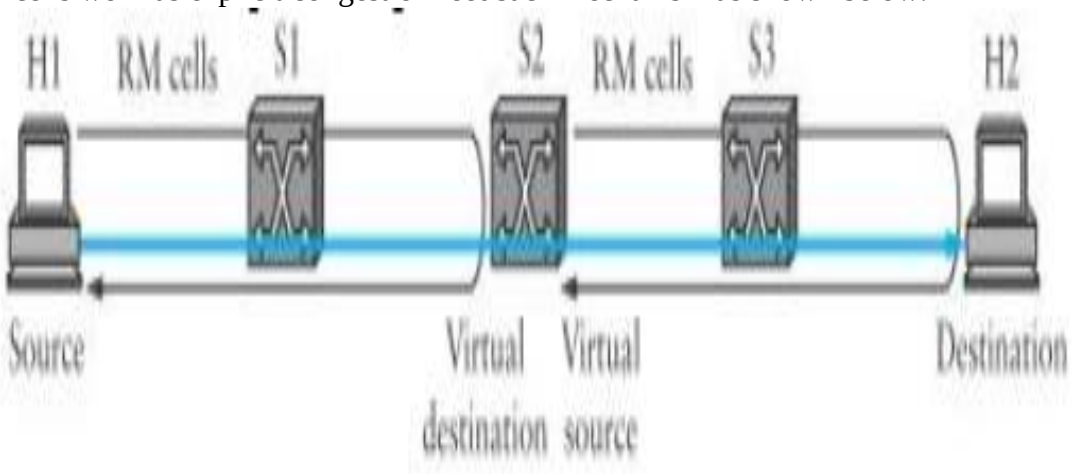
- UBR class is a best-effort delivery service that does not guarantee anything.
- UBR allows the source to specify a maximum rate at which it will send.

o Switches may make use of this information to decide whether to admit or reject or negotiate with the source for a less peak rate.

Available Bit Rate

- ABR apart from being a service class also defines a set of congestion-control mechanism.
- The ABR mechanisms operate over a virtual circuit by exchanging special ATM cells called resource management (RM) cells between the source and destination.

 RM cells work as explicit congestion feedback mechanism as shown below.



 ABR allows a source to increase or decrease its allotted rate as conditions dictate.
 ABR class delivers cells at a minimum rate. If more network capacity is available, this minimum rate can be exceeded.
 ABR is suitable for applications that are bursty in nature.

What is equation based congestion control?
 TCP's congestion-control algorithm is not appropriate for real-time applications.
 A smooth transmission rate is obtained by ensuring that flow's behavior adheres to an equation that models TCP's behavior.

$$\text{Rate} = \left(\frac{1}{\text{RTT} \times \sqrt{\rho}} \right)$$

 To be TCP-friendly, the transmission rate must be inversely proportional to the roundtrip time (RTT) and the square root of the loss rate (ρ).

Queuing discipline:

Introduction

As Internet can be considered as a *Queue of packets*, where transmitting nodes are constantly adding packets and some of them (receiving nodes) are removing packets from the queue. So, consider a situation where too many packets are present in this queue (or internet or a part of internet), such that constantly transmitting nodes are pouring packets at a higher rate than receiving nodes are removing them. This degrades the performance, and such a situation is termed as *Congestion*. Main reason of congestion is more number of packets into the network than it can handle. So, the objective of congestion control can be summarized as to maintain the number of packets in the network below the level at which performance falls off dramatically. The nature of a Packet switching network can be summarized in following points:

- A network of queues
- At each node, there is a queue of packets for each outgoing channel
- If packet arrival rate exceeds the packet transmission rate, the queue size grows without bound
- When the line for which packets are queuing becomes more than 80% utilized, the queue length grows alarmingly

When the number of packets dumped into the network is within the carrying capacity, they all are delivered, expect a few that have too be rejected due to transmission errors). And then the number delivered is proportional to the number of packets sent. However, as traffic increases too far, the routers are no longer able to cope, and they begin to lose packets. This tends to make matter worse. At very high traffic, performance collapse completely, and almost no packet is delivered. In the following sections, the causes of congestion, the effects of congestion and various congestion control techniques are discussed in detail.

Causes of Congestion

Congestion can occur due to several reasons. For example, if all of a sudden a stream of packets arrive on several input lines and need to be out on the same output line, then a long queue will be build up for that output. If there is *insufficient memory* to hold these packets, then packets will be lost (dropped). Adding more memory also may not help in certain situations. If router have an infinite amount of memory even then instead of congestion being reduced, it gets worse; because by the time packets gets at the head of the queue, to be dispatched out to the output line, they have already timed-out (repeatedly), and duplicates may also be present. All the packets will be forwarded to next router up to the destination, all the way only increasing the load to the network more and more. Finally when it arrives at the destination, the packet will be discarded, due to time out, so instead of been dropped at any intermediate router (in case memory is restricted) such a packet goes all the way up to the destination, increasing the network load throughout and then finally gets dropped there.

Slow processors also cause Congestion. If the router CPU is slow at performing the task required for them (Queuing buffers, updating tables, reporting any exceptions etc.), queue can build up even if there is excess of line capacity. Similarly, *Low-Bandwidth* lines can also cause congestion. Upgrading lines but not changing slow processors, or vice-versa, often helps a little; these can just shift the bottleneck to some other point. The real problem is the mismatch between different parts of the system.

Congestion tends to feed upon itself to get even worse. Routers respond to overloading by dropping packets. When these packets contain TCP segments, the segments don't reach their destination, and they are therefore left unacknowledged, which eventually leads to timeout and retransmission. So, the major cause of congestion is often the *bursty* nature of traffic. If the hosts could be made to transmit at a uniform rate, then congestion problem will be less common and all other causes will not even led to congestion because other causes just act as an enzyme which boosts up the congestion when the traffic is bursty (i.e., other causes just add on to make the problem more serious, main cause is the bursty traffic).

This means that when a device sends a packet and does not receive an acknowledgment from the receiver, in most the cases it can be assumed that the packets have been dropped by intermediate devices due to congestion. By detecting the rate at which segments are sent and not acknowledged, the source or an intermediate router can infer the level of congestion on the network. In the following section we shall discuss the ill effects of congestion.

Effects of Congestion

Congestion affects two vital parameters of the network performance, namely *throughput* and *delay*. In simple terms, the throughput can be defined as the percentage utilization of the network capacity. Figure 3.30(a) shows how throughput is affected as offered load increases. Initially throughput increases linearly with offered load, because utilization of the network increases. However, as the offered load increases beyond certain limit, say 60% of the capacity of the network, the throughput drops. If the offered load increases further, a point is reached when not a single packet is delivered to any destination, which is commonly known as *deadlock* situation. There are three curves in Figure. 3.30(a), the ideal one corresponds to the situation when all the packets introduced are delivered to their destination up to the maximum capacity of the network. The second one corresponds to the situation when there is no congestion control. The third one is the case when some congestion control technique is used. This prevents the throughput collapse, but provides lesser throughput than the ideal condition due to overhead of the congestion control technique.

The delay also increases with offered load, as shown in Figure. 3.30(b). And no matter what technique is used for congestion control, the delay grows without bound as the load approaches the capacity of the system. It may be noted that initially there is longer delay when congestion control policy is applied. However, the network without any congestion control will saturate at a lower offered load.

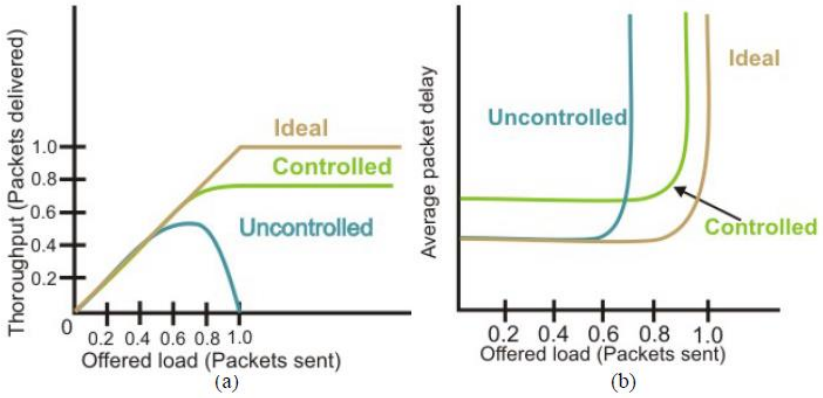


Figure 3.30 (a) Effect of congestion on throughput (b) Effect of congestion on delay

Congestion Control Techniques

Congestion control refers to the mechanisms and techniques used to control congestion and keep the traffic below the capacity of the network. As shown in Figure. 3.31, the congestion control techniques can be broadly classified two broad categories:

Open loop: Protocols to prevent or avoid congestion, ensuring that the system (or network under consideration) never enters a Congested State.

Close loop: Protocols that allow system to enter congested state, detect it, and remove it.

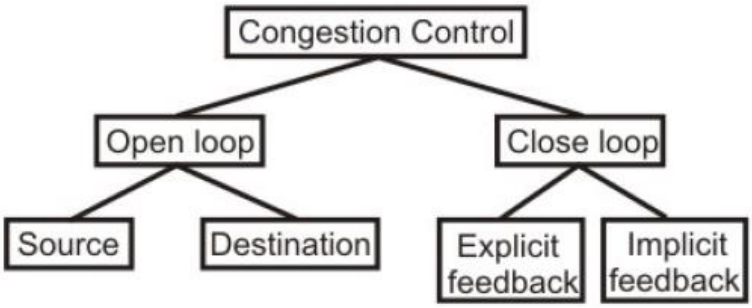


Figure 3.31 Congestion control categories

The first category of solutions or protocols attempt to solve the problem by a good design, at first, to make sure that it doesn't occur at all. Once system is up and running midcourse corrections are not made. These solutions are somewhat static in nature, as the policies to control congestion don't change much according to the current state of the system. Such Protocols are also known as *Open Loop* solutions. These rules or policies include deciding upon when to accept traffic, when to discard it, making scheduling decisions and so on. Main point here is that they make decision without taking into consideration the current state of the network. The open loop algorithms are further divided on the basis of whether these acts on source versus that act upon destination.

The second category is based on the concept of feedback. During operation, some system parameters are measured and feed back to portions of the subnet that can take action to reduce the congestion. This approach can be divided into 3 steps:

- Monitor the system (network) to detect whether the network is congested or not and what's the actual location and devices involved.
- To pass this information to the places where actions can be taken

- Adjust the system operation to correct the problem.

These solutions are known as *Closed Loop* solutions**.** Various Metrics can be used to monitor the network for congestion. Some of them are: the average queue length, number of packets that are timed-out, average packet delay, number of packets discarded due to lack of buffer space, etc. A general feedback step would be, say a router, which detects the congestion send special packets to the source (responsible for the congestion) announcing the problem. These extra packets increase the load at that moment of time, but are necessary to bring down the congestion at a later time. Other approaches are also used at times to curtail down the congestion. For example, hosts or routers send out probe packets at regular intervals to explicitly ask about the congestion and source itself regulate its transmission rate, if congestion is detected in the network. This kind of approach is a *pro-active* one, as source tries to get knowledge about congestion in the network and act accordingly.

Yet another approach may be where instead of sending information back to the source an intermediate router which detects the congestion send the information about the congestion to rest of the network, piggy backed to the outgoing packets. This approach will in no way put an extra load on the network (by not sending any kind of special packet for feedback). Once the congestion has been detected and this information has been passed to a place where the action needed to be done, then there are two basic approaches that can overcome the problem. These are: either to increase the resources or to decrease the load. For example, separate dial-up lines or alternate links can be used to increase the bandwidth between two points, where congestion occurs. Another example could be to decrease the rate at which a particular sender in transmitting packets out into the network.

The closed loop algorithms can also be divided into two categories, namely *explicit feedback* and *implicit feedback* algorithms. In the explicit approach, special packets are sent back to the sources to curtail down the congestion. While in implicit approach, the source itself acts pro-actively and tries to deduce the existence of congestion by making local observations. In the following sections we shall discuss about some of the popular algorithms from the above categories.

Leaky Bucket Algorithm

Consider a Bucket with a small hole at the bottom, whatever may be the rate of water pouring into the bucket, the rate at which water comes out from that small hole is constant. This scenario is depicted in figure 3.32(a). Once the bucket is full, any additional water entering it spills over the sides and is lost (i.e. it doesn't appear in the output stream through the hole underneath).

The same idea of leaky bucket can be applied to packets, as shown in Figure3.32(b). Conceptually each network interface contains a leaky bucket. And the following steps are performed:

- When the host has to send a packet, the packet is thrown into the bucket.
- The bucket leaks at a constant rate, meaning the network interface transmits packets at a constant rate.
- Bursty traffic is converted to a uniform traffic by the leaky bucket.
- In practice the bucket is a finite queue that outputs at a finite rate.

This arrangement can be simulated in the operating system or can be built into the hardware. Implementation of this algorithm is easy and consists of a finite queue. Whenever a packet arrives, if there is room in the queue it is queued up and if there is no room then the packet is discarded.

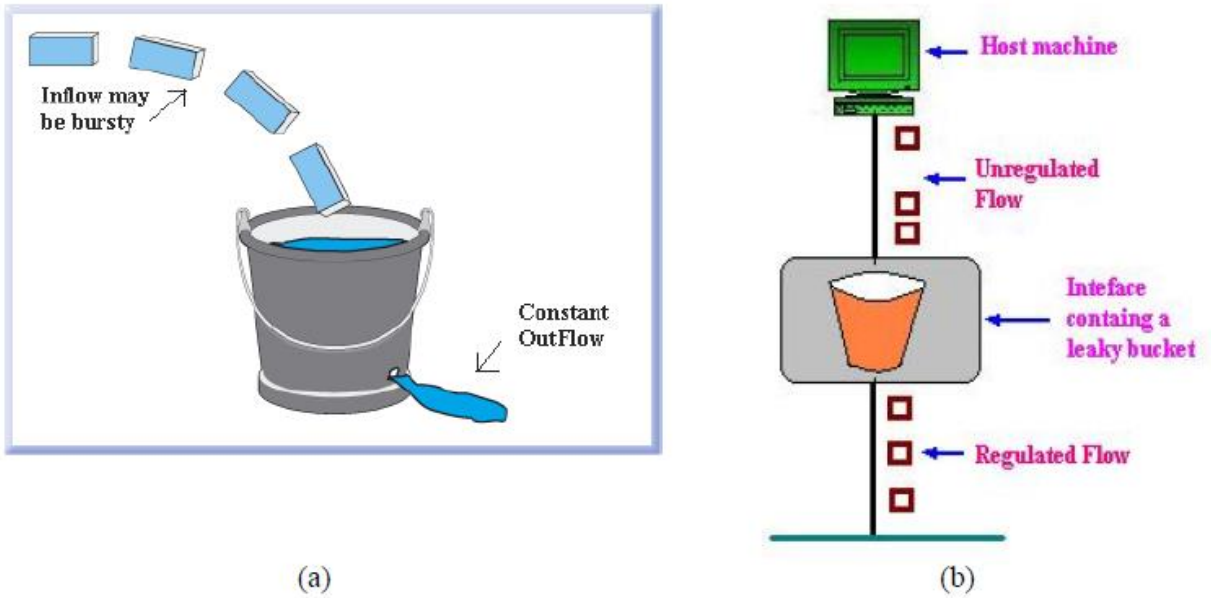


Figure 3.32(a) Leaky bucket (b) Leaky bucket implementation

Token Bucket Algorithm

The leaky bucket algorithm described above, enforces a rigid pattern at the output stream, irrespective of the pattern of the input. For many applications it is better to allow the output to speed up somewhat when a larger burst arrives than to loose the data. Token Bucket algorithm provides such a solution. In this algorithm leaky bucket holds token, generated at regular intervals. Main steps of this algorithm can be described as follows:

- In regular intervals tokens are thrown into the bucket.
- The bucket has a maximum capacity.
- If there is a ready packet, a token is removed from the bucket, and the packet is send.
- If there is no token in the bucket, the packet cannot be send.

Figure 3.33 shows the two scenarios before and after the tokens present in the bucket have been consumed. In Figure 3.33(a) the bucket holds two tokens, and three packets are waiting to be sent out of the interface, in Figure3.33(b) two packets have been sent out by consuming two tokens, and 1 packet is still left.

The token bucket algorithm is less restrictive than the leaky bucket algorithm, in a sense that it allows bursty traffic. However, the limit of burst is restricted by the number of tokens available in the bucket at a particular instant of time.

The implementation of basic token bucket algorithm is simple; a variable is used just to count the tokens. This counter is incremented every t seconds and is decremented whenever a packet is sent. Whenever this counter reaches zero, no further packet is sent out as shown in Figure3.34.

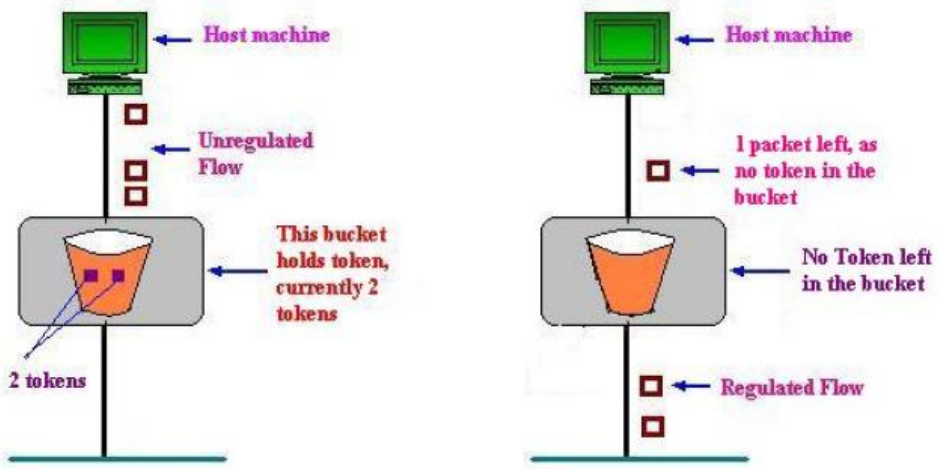


Figure 3.33(a) Token bucket holding two tokens, before packets are send out, (b) Token bucket after two packets are send, one packet still remains as no token is left

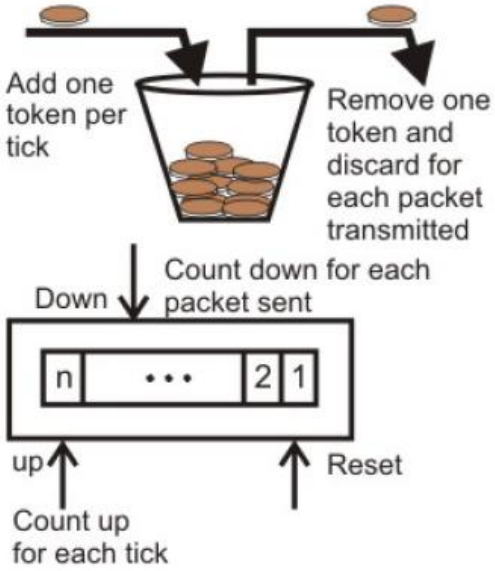


Figure3.34: Implementation of the Token bucket algorithm

Congestion control in virtual Circuit

In open loop algorithms, the policy decisions are made in the beginning, irrespective of the current state. Both leaky bucket algorithm and token bucket algorithm are open loop algorithms. In this section we shall have a look at how the congestion is tackled in a virtual-circuit network. *Admission control* is one such closed-loop technique, where action is taken once congestion is detected in the network. Different approaches can be followed:

Simpler one being: do not set-up new connections, once the congestion is signaled. This type of approach is often used in normal telephone networks. When the exchange is overloaded, then no new calls are established.

Another approach, which can be followed is: to allow new virtual connections, but route these carefully so that none of the congested router (or none of the problem area) is a part of this route.

Yet another approach can be: To negotiate different parameters between the host and the network, when the connection is setup. During the setup time itself, Host specifies the volume and shape of traffic, quality of service, maximum delay and other parameters, related to the traffic it would be offering to the network. Once the host specifies its requirement, the resources needed are reserved along the path, before the actual packet follows.

Choke Packet Technique

The *choke packet* technique, a closed loop control technique**,** can be applied in both virtual circuit and datagram subnets. Each router monitors its resources and the utilization at each of its output line. There is a threshold set by the administrator, and whenever any of the resource utilization crosses this threshold and action is taken to curtail down this. Actually each output line has a utilization associated with it, and whenever this utilization crosses the threshold, the output line enters a "warning" state. If so, the router sends a *choke packet* back to the source, giving it a feedback to reduce the traffic. And the original packet is tagged (a bit is manipulated in the header field) so that it will not generate other choke packets by other intermediate router, which comes in place and is forwarded in usual way. It means that the first router (along the way of a packet), which detects any kind of congestion, is the only one that sends the choke packets.

When the source host gets the choke packet, it is required to reduce down the traffic send out to that particular destination (choke packet contains the destination to which the original packet was send out). After receiving the choke packet the source reduces the traffic by a particular fixed percentage, and this percentage decreases as the subsequent choke packets are received. Figure 3.35 depicts the functioning of choke packets.

For Example, when source A receives a choke packet with destination B at first, it will curtail down the traffic to destination B by 50%, and if again after affixed duration of time interval it receives the choke packet again for the same destination, it will further curtail down the traffic by 25% more and so on. As stated above that a source will entertain another subsequent choke packet only after a fixed interval of time, not before that. The reason for this is that when the first choke packet arrives at that point of time other packets destined to the same destination would also be there in the network and they will generate other choke packets too, the host should ignore these choke packets which refer to the same destination for a fixed time interval.

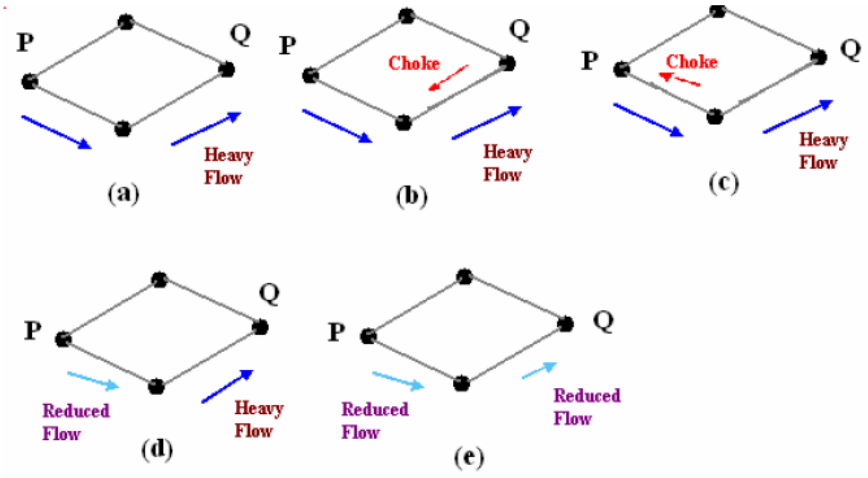


Figure 3.35 Depicts the functioning of choke packets, (a) Heavy traffic between nodes P and Q, (b) Node Q sends the Choke packet to P, (c) Choke packet reaches P, (d) P reduces the flow and send a reduced flow out, (e) Reduced flow reaches node Q

Hop-by Hop Choke Packets

This technique is advancement over Choked packet method. At high speed over long distances, sending a packet all the way back to the source doesn't help much, because by the time choke packet reach the source, already a lot of packets destined to the same original destination would be out from the source. So to help this, Hop-by-Hop Choke packets are used. In this approach, the choke packet affects each and every intermediate router through which it passes by. Here, as soon as choke packet reaches a router back to its path to the source, it curtails down the traffic between those intermediate routers. In this scenario, intermediate nodes must dedicate few more buffers for the incoming traffic as the outflow through that node will be curtailed down immediately as choke packet arrives it, but

the input traffic flow will only be curtailed down when choke packet reaches the node which is before it in the original path. This method is illustrated in Figure3.36.

As compared to choke packet technique, hop-by-hop choke packet algorithm is able to restrict the flow rapidly. As can been seen from Figures 3.35 and 3.36, one-step reduction is seen in controlling the traffic, this single step advantage is because in our example there is only one intermediate router. Hence, in a more complicated network, one can achieve a significant advantage by using hop-by-hop choke packet method.

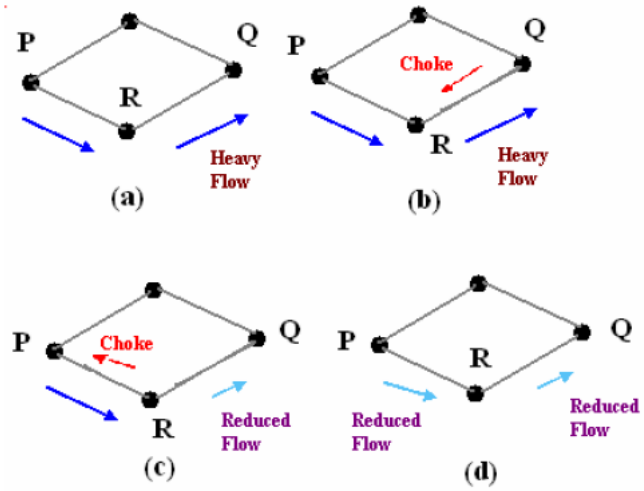


Figure 7.5.7 Depicts the functioning of Hop-by-Hop choke packets, (a) Heavy traffic between nodes P and Q, (b) Node Q sends the Choke packet to P, (c) Choke packet reaches R, and the flow between R and Q is curtail down, Choke packer reaches P, and P reduces the flow out

Load Shedding

Another simple closed loop technique is *Load Shedding***;** it is one of the simplest and more effective techniques. In this method, whenever a router finds that there is congestion in the network, it simply starts dropping out the packets. There are different methods by which a host can find out which packets to drop. Simplest way can be just choose the packets randomly which has to be dropped. More effective ways are there but they require some kind of cooperation from the sender too. For many applications, some packets are more important than others. So, sender can mark the packets in priority classes to indicate how important they are. If such a priority policy is implemented than intermediate nodes can drop packets from the lower priority classes and use the available bandwidth for the more important packets.

Slow Start - a Pro-active technique

This is one of the pro-active techniques, which is used to avoid congestion. In the original implementation of TCP, as soon as a connection was established between two devices, they could each go "hog wild", sending segments as fast as they liked as long as there was room in the other devices receive window. In a busy internet, the sudden appearance of a large amount of new traffic could aggravate any existing congestion.

To alleviate this, modern TCP devices are restrained in the rate at which they initially send segments. Each sender is at first restricted to sending only an amount of data equal to one "full-sized" segment—that is, equal to the MSS (maximum segment size) value for the connection. Each time an acknowledgment is received, the amount of data the device can send is increased by the size of another full-sized segment. Thus, the device "starts slow" in terms of how much data it can send, with the amount it sends increasing until either the full window size is reached or congestion is detected on the link. In the latter case, the congestion avoidance feature is used.

When potential congestion is detected on a TCP link, a device responds by throttling back the rate at which it sends segments. A special algorithm is used that allows the device to drop the rate at which segments are sent quickly when congestion occurs. The device then uses the *Slow Start* algorithm just above to gradually increase the transmission rate back up again to try to maximize throughput without congestion occurring again.

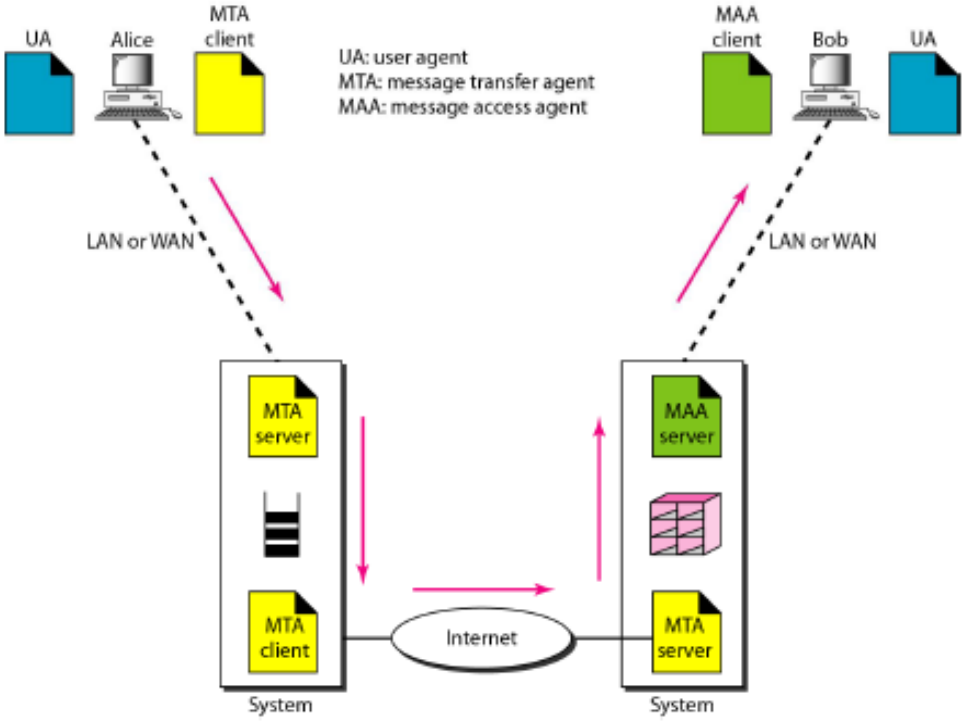
UNIT-V

Explain the various components of an email system and the protocols used.

E-mail is one of the most popular internet services than it was when it was envisaged.

Common Architecture

The following shows components of e-mail system involved in Alice sending a message to Bob



Components

1. User Agent
2. Message
3. Message Transfer Agent
4. Message Access Agent

User Agent

 A user agent (UA) is software that is either *command* (eg. pine, elm) or *GUI* based (eg. Microsoft Outlook, Netscape). It facilitates:

- *Composing messages*UA helps to compose messages by providing a template that comes with a built-in editor.
- *Reading messages*UA checks mail in the incoming box and apart from message provides information such as sender, size, subject and flag (read, new).
- *Replying to messages*UA allows user to reply (send message) back to sender
- *Forwarding messages*UA facilitates forwarding message to a third party.
- *Handling mailboxes*UA creates two mailboxes for each user, namely *inbox* (to store received emails) and *outbox* (to keep all sent mails).

Message Format

 RFC822 defines message to have two parts namely *header* and a *body*.

 The message header is a series of <CRLF> terminated lines. Each header line contains an *type* and *value* separated by a colon (:). It is filled by the user/system. Some of them are:

- From□user who sent the message
- To□identifies the message recipient(s).
- Subject□says something about the purpose of the message
- Date□when the message was transmitted
- E-mail address consists of *user_name@domain_name* where domain_name is hostname of the *mail server*.

 The body of the message contains the actual information

- The header is separated from the message body by a *blank* line.

 Initially email system was designed to send messages only in NVT 7-bit ASCII format.

- Languages such as French, German, Chinese, Japanese were not supported.
- Image, audio and video files cannot be sent.

Multipurpose Internet Mail Extensions (MIME)

 MIME is a supplementary protocol that allows *non-ASCII* data to be sent through e-mail.

 MIME transforms non-ASCII data to NVT ASCII and delivers to client MTA. The NVT ASCII data is converted back to non-ASCII form at the recipient mail server.

 MIME defines five headers. They are:

- MIME-Version□specifies the current version 1.1
- Content-Type□specifies message type such as *text* (plain, html), *image* (jpeg, gif), *audio*, *video* and *application* (postscript, msword). If more than one type exists, then it is termed as *multipart* (mixed).
- Content-Transfer-Encoding□defines how data in the message body is encoded such as *binary*, *base64*, *7-bit*, etc.
- Content-Id□unique identifier the whole message in a multiple message type.
- Content-Description□describes type of the message body.

For example, a message containing plain text and an image file looks like:

```
MIME-Version: 1.0
Content-Type: multipart/mixed; boundary="-------417CA6E2DE4ABCAFBC5"
From: Alice Smith <Alice@cisco.com>
To: Bob@cs.Princeton.edu
Subject: promised material
Date: Mon, 07 Sep 1998 19:45:19 -0400
---------417CA6E2DE4ABCAFBC5
Content-Type: text/plain; charset=us-ascii
Content-Transfer-Encoding: 7bit
…
---------417CA6E2DE4ABCAFBC5
Content-Type: image/jpeg
Content-Transfer-Encoding: base64
```

Message Transfer Agent (MTA): SMTP

 Message Transfer Agent (MTA) is a mail daemon (a version of sendmail program) that helps to transmit/receive message over the network.

 To send mail a system must have the client MTA, and to receive mail a system must have a server MTA.

 Simple Mail Transfer Protocol (SMTP) defines communication between client/server MTA.

 SMTP defines how commands and responses must be sent back and forth.

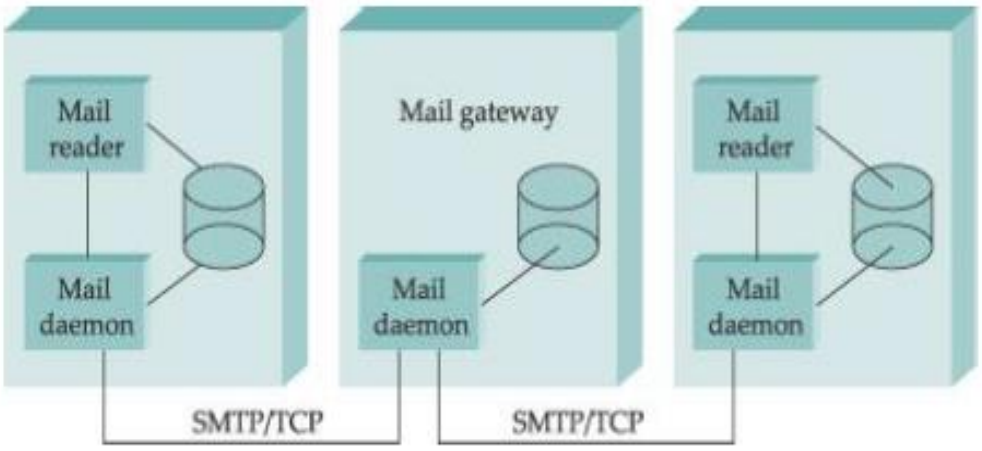
 Some commands sent from client MTA are:

Command	Description
MAIL FROM	Sender of the message
RCPTTO	Recipient of the message
DATA	Body of the mail
QUIT	Terminate
VRFY	Name of recipient to be verified before forwarding
EXPN	Mailing list to be expanded

- Common responses sent from server MTA are:

Code	Description
220	Service ready
250	Request completed
354	Start mail input
450	Mailbox not available
500	Syntax error; unrecognized command
551	User not local

 SMTP uses TCP connection on port 25 to forward the entire message and store at intermediate mail servers/mail gateways until it reaches the recipient mail server.



The following *example* shows commands and responses using SMTP protocol

HELO cs.princeton.edu
250 Hello daemon@mail.cs.princeton.edu [128.12.169.24]
MAIL FROM:<Bob@cs.princeton.edu>
250 OK
RCPT TO:<Alice@cisco.com>
250 OK
RCPT TO:<Tom@cisco.com>
550 No such user here
DATA
354 Start mail input; end with <CRLF>.<CRLF>
Blah blah blah...
...etc. etc. etc.
<CRLF>.<CRLF>
250 OK
QUIT
221 Closing connection

In each exchange, the client posts a command and the server responds with a code. The server also returns a human-readable explanation for the code. After the commands and responses, client sends the message which is ended by a period (.) and terminates the connection.

Message Access Agent (MAA)/Mail Reader: POP and IMAP

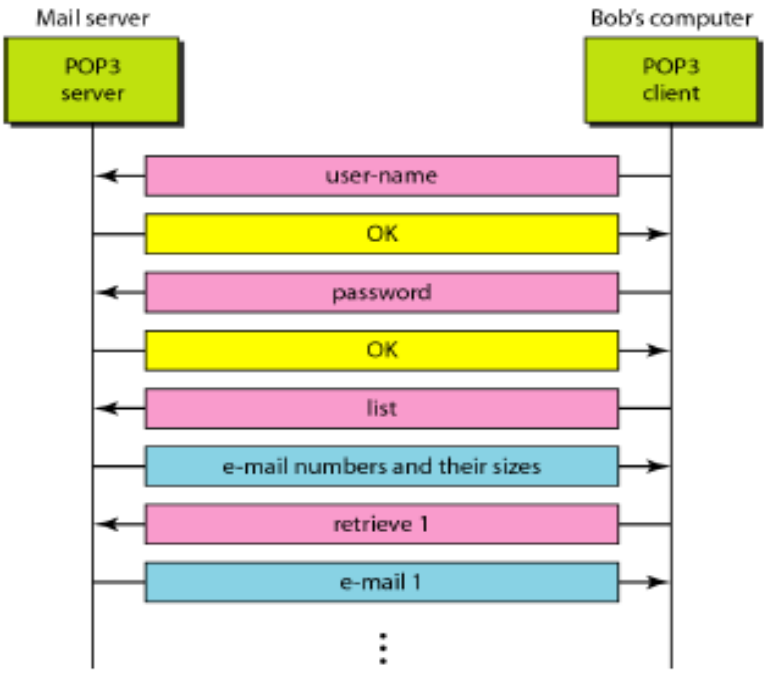
 MAA or mail reader allows user to retrieve messages from the mailbox, so that user can perform actions such as reply, forwarding, etc.
 The two message access protocols are:
- Post Office Protocol, version 3 (POP3)
- Internet Mail Access Protocol, version 4 (IMAP4)

 SMTP is a push type protocol whereas POP3 and IMAP4 are pop type protocol.

POP3

 POP3 is simple and limited in functionality
 POP3 client is installed on the recipient computer and POP3 server on the mail server.
 The client opens a connection to the server on TCP port 110.
 The client sends username and password to access the mailbox and retrieve the messages.
 POP3 works in two modes namely, *delete* and *keep* mode.
- In delete mode, mail is deleted from the mailbox after retrieval
- In keep mode, mail after reading is kept in mailbox for later retrieval.

Downloading message using POP3 is shown below:

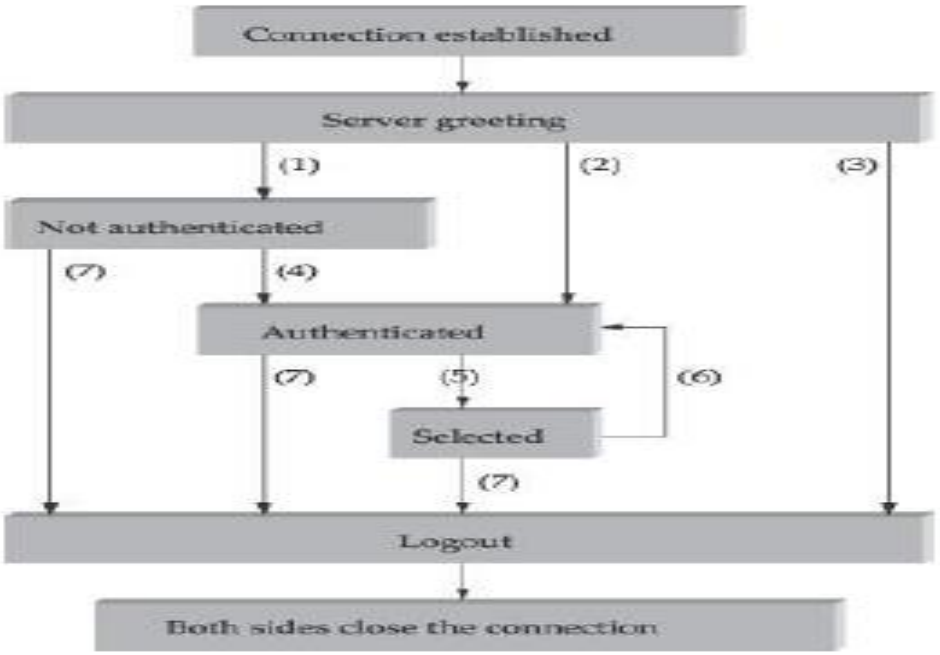


IMAP4

 IMAP is a client/server protocol running over TCP. The client issues commands and the mail server responds.

- The client can issue commands such as LOGIN, AUTHENTICATE, SELECT, EXAMINE, CLOSE, LOGOUT, etc.
- Server responses include OK, FETCH, STORE, DELETE, EXPUNGE, NO, BAD, etc.

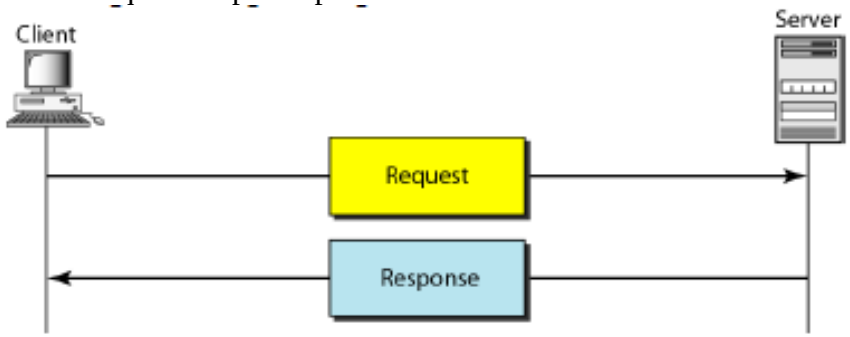
 The exchange begins with the client authenticating itself to access the mailbox. This is represented as a state transition diagram as shown below.



 When the user asks to FETCH a message, server returns it in MIME format and the mail reader decodes it.
 IMAP also defines message *attributes* such as size and *flags* such as Seen, Answered, Deleted and Recent.

Explain HTTP protocol in detail.

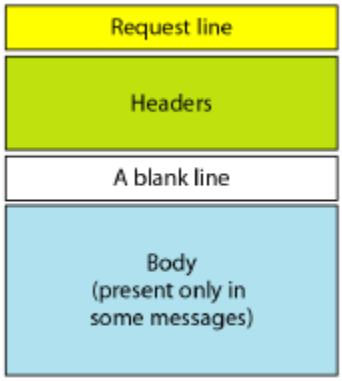
 WWW is a distributed client/server service, in which a client (*web browser*) can access a service through a server, where the service is distributed over many locations called *sites*.
 Both the client and server use Hypertext Transfer Protocol (HTTP).
 Web browsers allow users to access files (repository of information) through uniform resource locator (URL).
 When user enters URL in the web browser, the browser forms a *request* message and sends to the server.
 The server retrieves the requested URL and sends it as a *response* message.
 The browser displays the response in HTML / appropriate format.
 HTTP uses one TCP connection on well known port 80 to transfer data between client and the server.
 HTTP is a stateless request/response protocol as shown.



 The general form of message is shown below:

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

Request Message



Request line

The request line specifies three elements:

- *HTTP version* specifies current version of the protocol i.e., 1.1
- *URL* specifies path (absolute/relative) along with document name.
- The *Request type* specifies methods that operate on the URL are:

Method	Description
GET	retrieve document specified as URL
HEAD	retrieve meta-information about the URL document
POST	send information from client to the server
PUT	store document under specified URL
TRACE	echoes the incoming request
OPTION	request information about available options
DELETE	delete specified URL

For example, the request line to retrieve file index.html on host cs.princeton.edu is GET

http://www.cs.princeton.edu/index.html HTTP/1.1

Request Header

Request Header specifies client's configuration and preferred document format:

Request Header Description

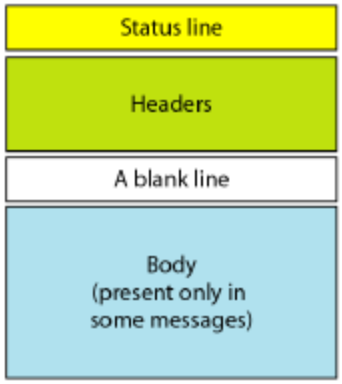
Request Header	Description
Accept-charset	specifies the character set the client can handle
Authorization	specifies what permissions the client has
From	specifies e-mail address of the user
Host	specifies host name and port number of the server
If-modified-since	server sends the URL if it is newer than specified date
Referrer	specifies URL of the linked document
User-agent	specifies name of the browser

The above example using request header is specified as

GET index.html HTTP/1.1

Host: www.cs.princeton.edu

Response Messages



Status line

 The status code field consists of three digits (1xx–Informational, 2xx–Success, 3xx–Redirection, 4xx–Client Error, 5xx–Server Error)

 The status phrase explains the status code in text form. Some of them are:

Code	Phrase	Description
100	Continue	Initial request received, client to continue process
200	OK	Request is successful
201	Created	A new URL is created.
204	No content	There is no content in the body.
301	Moved permanently	The requested URL is no longer in use
304	Not modified	The document has not been modified
401	Unauthorized	The request lacks proper authorization
404	Not found	The document is not found
500	Internal server error	There is an error, such as a crash, at the server site

For example, the server reports as follows, if the requested file is not found

HTTP/1.1 404 Not Found

Response Header

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

The response for a moved page is given below.

HTTP/1.1 301 Moved Permanently

Location: http://www.princeton.edu/cs/index.html.

Distinguish between persistent and non-persistent connection.

Non-persistent connection

 A TCP connection is required for each request/response

 Imposes high overhead on the server because the server needs N buffers for N URL pointers and TCP overhead for each connection

Persistent connection

 Client and server can exchange multiple request/response messages over the same TCP connection

 Eliminates the connection setup overhead and load on the server

 TCP's congestion window mechanism is able to operate more efficiently.

 The problem is that how long the connection should be kept open.

- The server times out, if there is no request from the client for a specified period

Write short note on caching.

 Caching enables the client to retrieve document faster and reduces load on the server.

 Caching can be implemented at different places

- For example, the ISP router can cache pages. Further such request coming from its clients, the ISP responds.
- Proxy server is a host that keeps copies responses to recent requests. The client sends request to the proxy server. The proxy server either responds to client or forwards the request to the server.
- The browser also can cache pages.

 Server assigns expiration date (using Expires header field) to each page, beyond which the page should not be cached.

 Therefore prior to caching a page, its expiration date is checked. If a cached page reaches its expiration, then the page is deleted.

 The proxy node also can verify whether it has the latest document by using If-Modified- Since header.

 A page can have cache directives that must be adhered by all caching nodes. (for example, a no-cache page).

Explain the role of DNS on a computer network.

 We remember domain-names rather than IP address of a host, since it is user-friendly.

 Thus, need for a system to map domain name to an IP address that includes:

- A *namespace* to define domain names without conflict.
- Binding of domain names to IP address
- A *name server* that returns IP address for a given name

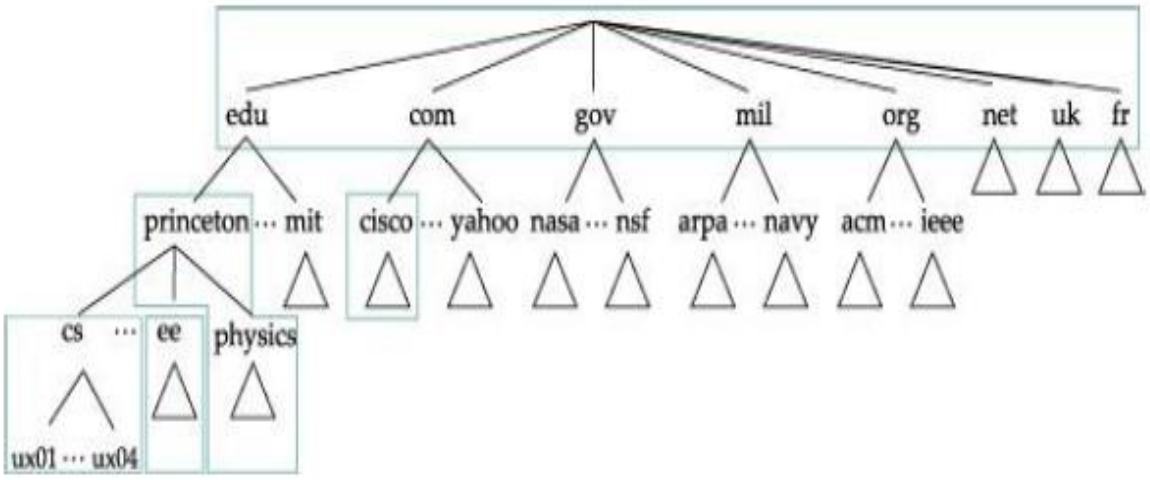
 During early days of internet, there were only few hundred hosts

- A central authority called the Network Information Center (NIC) maintained name-to-address bindings in a flat-file called *hosts.txt*
- A new host that joins the internet would mail its name and IP address to NIC.
- NIC updates *hosts.txt* and mails to all hosts.
- Name server resolved domain names using a simple lookup hosts.text

 As hosts grew to thousands and millions, the flat file approach failed, leading to evolution of DNS in mid 1980s.

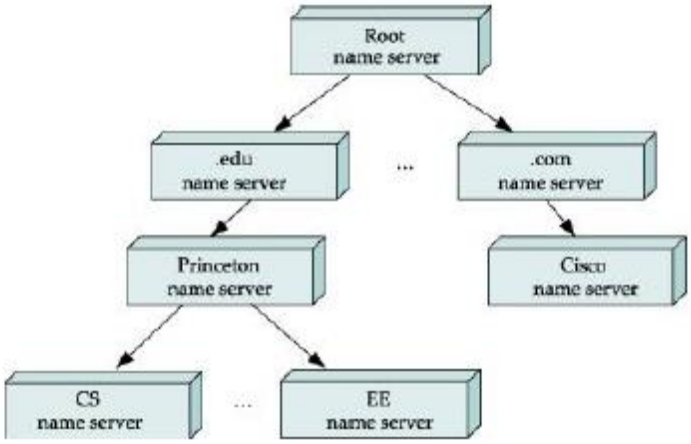
Name Hierarchy

 DNS was originally funded by ARPA
 DNS uses hierarchical name space for domains in the Internet.
 Hierarchical naming permits use of same sub-domain name in different domains.
 Domain names are case insensitive and can be up to 63 characters
 DNS names are processed from right to left and use periods as the separator.
 DNS can be used to map names to values, not necessarily from domain names to IP address.
 DNS hierarchy can be visualized as a tree, where each *node* in the tree corresponds to a domain and the *leaves* relate to hosts.
 Six big domains are .edu (education) .com (commercial) .gov (US government) .mil (US military) .org (non-profitable organization) and .net (network providers).
 Top level domain exist one for each country .uk (united kingdom) .fr (france) .in (india), etc.



Name Servers

 The domain hierarchy is partitioned into *zones*.
 Each zone acts as central authority for that part of the sub-tree.
 The topmost domains are managed by NIC.
 In the .edu hierarchy, *princeton* is a zone.
 Each zone can be further sub-divided that manage using their own name servers such as CS department under princeton university. The hierarchy of name servers is shown below.



 Each zone information is implemented on at least two name servers.
 Clients send queries to name servers, and name servers respond to it.
 The response contains either the host IP address or address of another name server
 Each name server contains a collection of *resource records*.
 A resource record is a name-to-value binding and is a 5-tuple with the following fields

Name	Value	Type	Class	TTL

 Name tells the domain to which this record applies. It is the primary search key, used to satisfy queries
 The Type field tells what kind of record it is. Some commonly used types are:
  - NS *Value* field contains a name server
  - CNAME *Value* field contains canonical name for the host. Used to define aliases.
  - MX *Value* field contains a mail server that accepts messages for the domain.
  - A *Value* field contains an IP address
 The Value field can be a number, a domain name, or an ASCII string. The semantics depend on the record type
 For internet information, the Class field is always IN.
 The TTL field gives an indication of how long the resource record is valid.

Root name server
 The root name server contains an NS record for each second-level server.
 It also has an A record that translates this name into corresponding IP address.
The following shows part of .edu root name server

(princeton.edu, cit.princeton.edu, NS, IN)
(cit.princeton.edu, 128.196.128.233, A, IN)
…

Zone name server
 The zone name server princeton.edu has a name server available on host cit.princeton.edu that contains the following records.
 Some records contain A records, whereas others point to next level name servers.

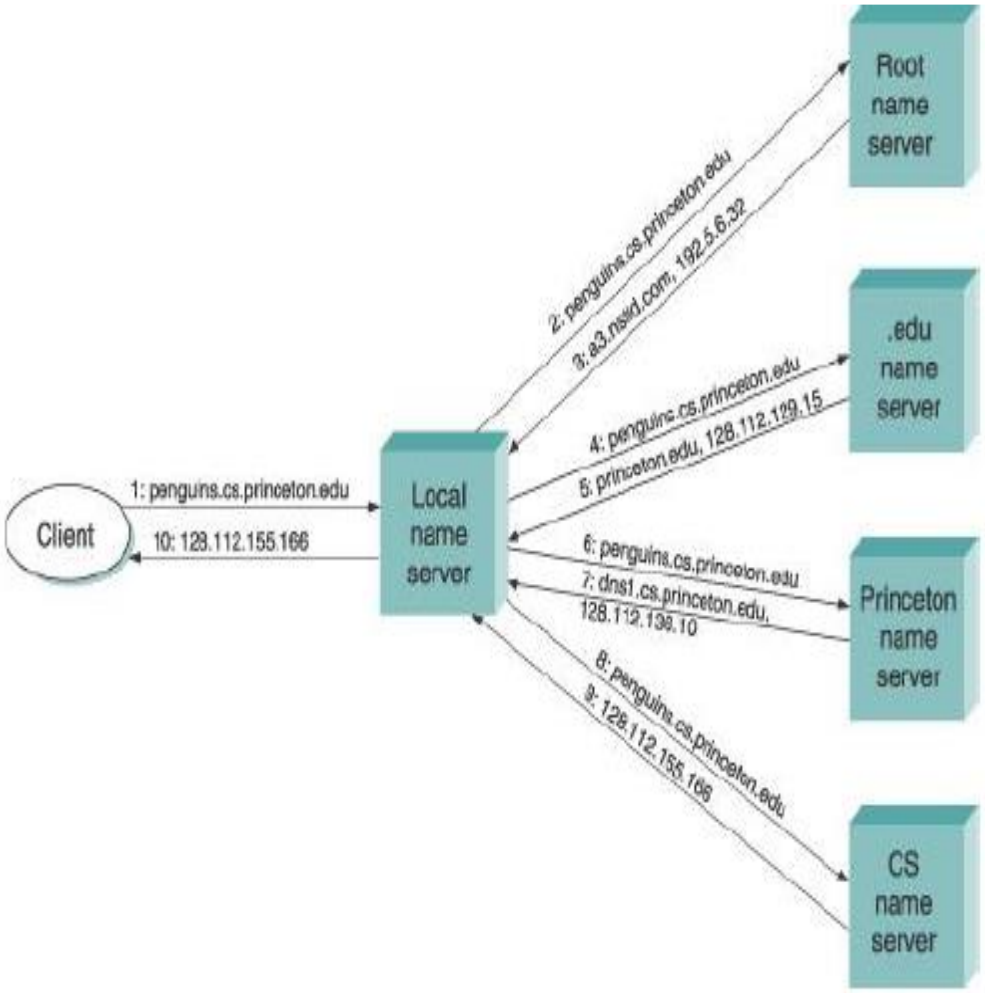
(cs.princeton.edu, gnat.cs.princeton.edu, NS, IN)
(gnat.cs.princeton.edu, 192.12.69.5, A, IN)
…

Eventually, third-level name server, such as the domain cs.princeton.edu, contains A records for all of its hosts.
(cs.princeton.edu, gnat.cs.princeton.edu, MX, IN)
(cicada.cs.princeton.edu, 192.12.69.60, A, IN)
(cic.cs.princeton.edu, cicada.cs.princeton.edu, CNAME, IN)
(gnat.cs.princeton.edu, 192.12.69.5, A, IN)

Name Resolution
For example, the step involved in the lookup for name cicada.cs.princeton.edu is as follows:
 The client first sends a query containing cicada.cs.princeton.edu to the *root* server.
 The root server, does not finds an exact match, but locates the *NS* record for princeton.edu

 The root returns the A record for princeton.edu back to the client.
 The client sends the same query to 128.196.128.233 and receives the A record for cs.princeton.edu
 Finally the client sends the query to 192.12.69.5 and gets the A record for cicada.cs.princeton.edu

The drawback with this lookup is:
 All hosts should know the root name server, which is not feasible.
 Instead, the client can send query to the local name server that it knows
 The local name server can query the root name server on behalf of the client.
 Once the local NS gets the required response, it caches the A record based on TTL and sends the record to the client.



Why is POP not preferred?
 It does not allow the user to organize their mail on the server
 The user cannot have different folders on the server
 It does not allow the user to partially check the contents of the mail before downloading

List the advantages of IMAP over POP
IMAP4 is more powerful and more complex than POP3. The additional features provided are:

 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.

Explain how SNMP is used to manage nodes on the network (SNMP)
 Simple Network Management Protocol (SNMP) is a framework for managing devices in an internet using TCP/IP.
 It provides a set of fundamental operations for monitoring and maintaining an internet
 SNMP uses the concept of *manager* and *agent*.
- A manager is a host that runs the SNMP client program.
- A managed station called an agent, is a router that runs the SNMP server program

 SNMP is an application layer protocol, therefore it can monitor devices of different manufacturers installed on different physical networks.
 SNMP management includes:
- A manager that checks an agent by requests information on behavior of the agent.
- A manager forces an agent to perform a task by setting/resetting values in the agent database.
- An agent warns the manager of an unusual situation.

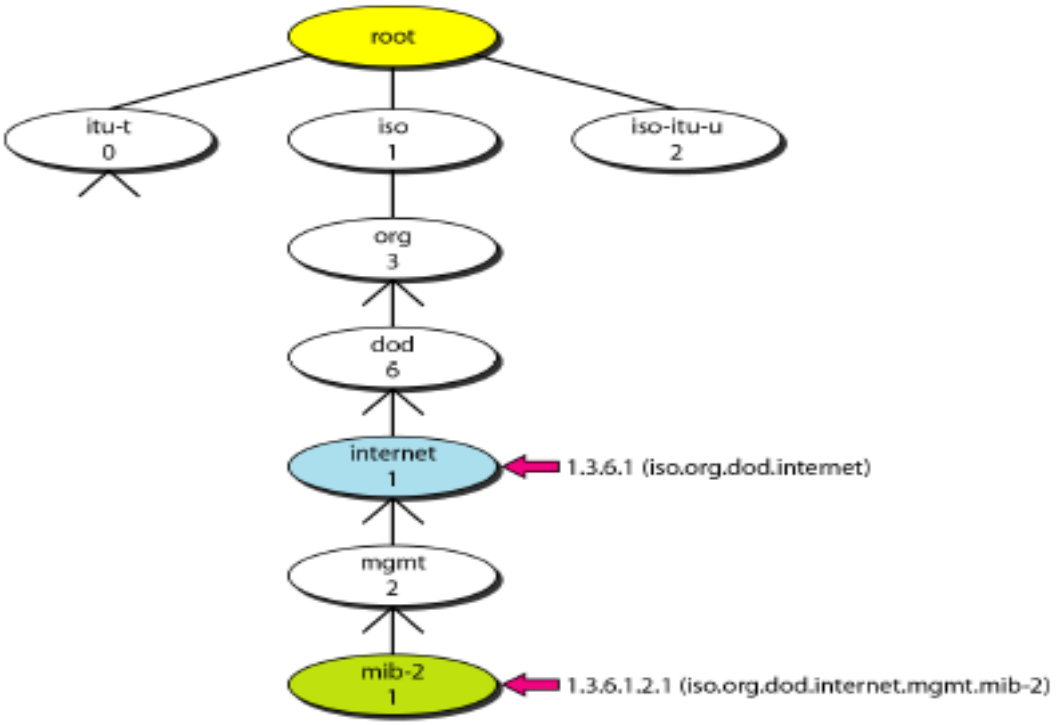
 SNMP uses services of UDP on two well-known ports, 161 (agent) and 162 (manager).
 SNMP is supported by two other protocols in Internet Network management. They are:
- Structure of Management Information (SMI)
- Management Information Base (MIB)

 The role of SNMP is to
- Define format of the packet to be sent from a manager to an agent and vice versa.
- Interprets the result and creates statistics
- Responsible for reading and setting object values

 The role of SMI is to
- Define rules for naming objects and object types.
- Uses *Basic Encoding Rules* to encode data to be transmitted over the network.

 The role of MIB is to
- creates a collection of named objects, their types, and their relationships to each other in an entity to be managed

Object Identifier
 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 by using a sequence of integers separated by *dots*.
 The objects that are used in SNMP are located under the mib-2 object, so their identifiers always start with 1.3.6.1.2.1
 Object identifiers follow lexographic ordering.

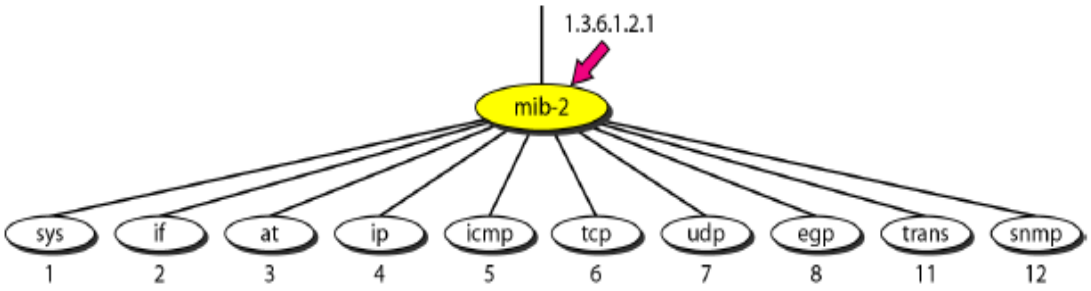


MIB Groups

 Each agent has its own MIB2 (version 2), which is a collection of all the objects that the manager can manage.

 The objects in MIB2 are categorized under 10 different groups namely *system, interface, address translation, ip, icmp,* tcp, *udp, egp, transmission,* and *snmp*.

- sys (*system* defines general information about the node such as the name, location, and lifetime.
- if (*interface* defines information about all the interfaces of the node such as physical address and IP address, packets sent and received on each interface, etc.
- at (*address translation* defines information about the ARP table
- ip defines information related to IP such as the routing table, statistics on datagram forwarding, reassembling and drop, etc.
- tcp defines general information related to TCP, such as the connection table, time-out value, number of ports, and number of packets sent and received.
- udp information on UDP traffic such as total number of UDP packets sent and received.

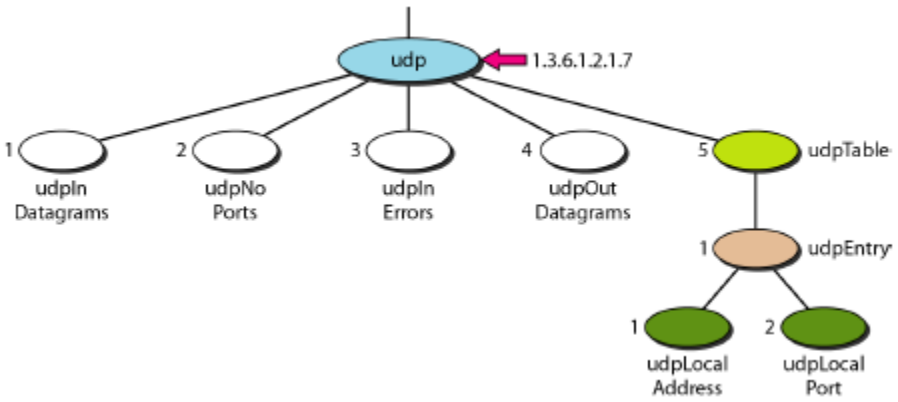


MIB variables

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

 To access any of the simple variable content, use *id* of the *group* (1.3.6.1.2.1.7) followed by the id of the *variable* and an instance suffix, which is 0.

- For example, variable *udpInDatagrams* is accessed as 1.3.6.1.2.1.7.1.0



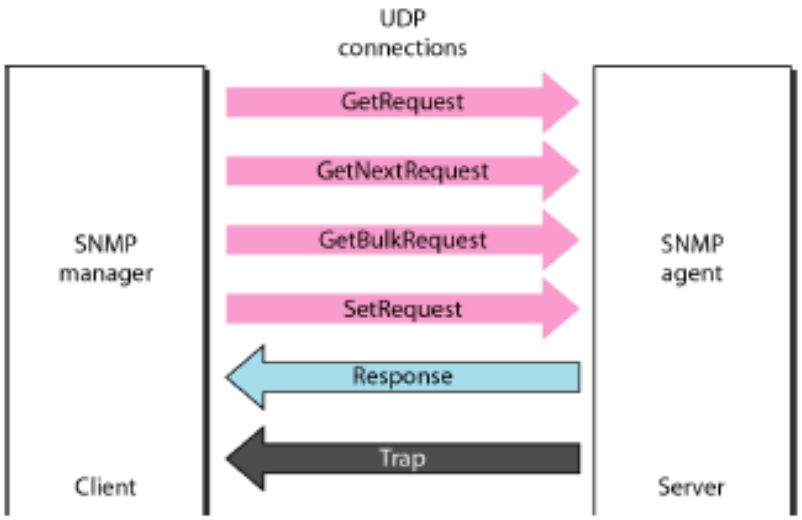
 In case of table, only leaf elements are accessible.

- In this case, the group id is followed by table id and so on up to the leaf element.
- To access a specific instance (row) of the table, add the index to the above ids.
- The indexes are based on the value of one or more fields in the entries.
- Tables are ordered according to column-row rules, i.e one should go column by column from top to bottom.

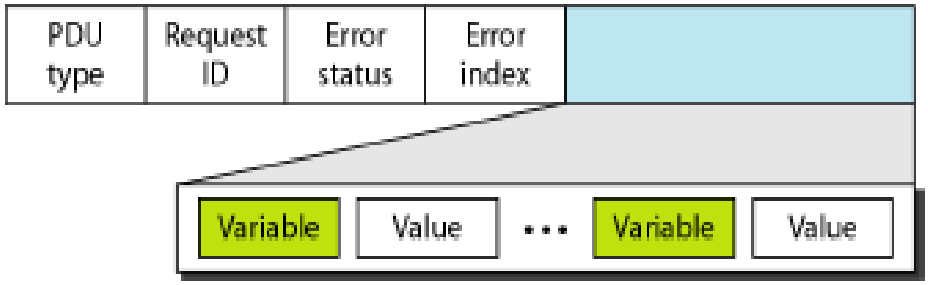
SNMPv3 PDU

 SNMP is request/reply protocol that defines PDUs GetRequest, GetNextRequest, GetBulkRequest, SetRequest, Response and Trap.

- GetRequest used by manager to retrieve value of agent's variable(s)
- GetNextRequest used by manager to retrieve next entries in a agent's table
- SetRequest used by manager to set value of an agent's variable
- Response sent from an agent to manager in response to GetRequest/ GetNextRequest that contains value of variables
- Trap sent from an agent to the manager to report an event such as reboot.



The PDU format is shown below:



 The SNMP client puts the identifier for the MIB variable it wants to get into the request message, and sends this message to the server.
 The server then maps this identifier into a local variable, retrieves the current value held in this variable, and uses BER to encode the value it sends back to the client.

Discuss Telnet in detail

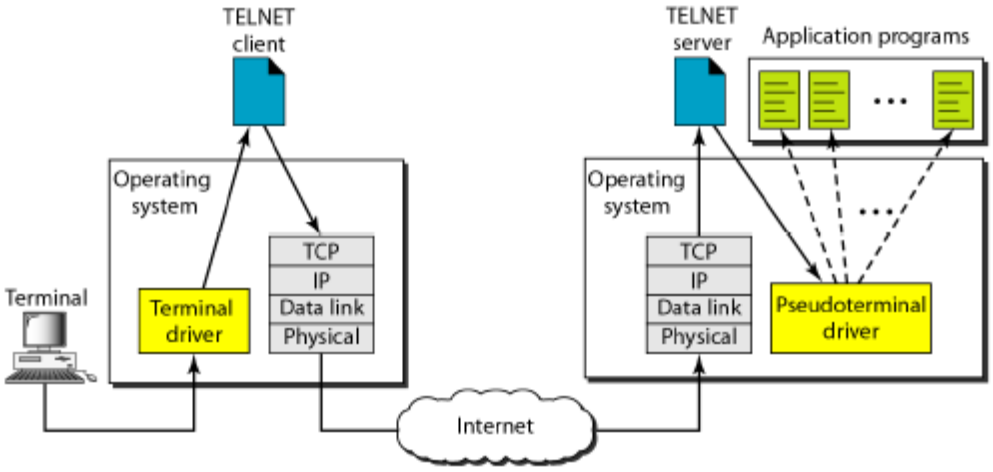
 TErminaL NETwork (TELNET) is a general-purpose client/server application program.
 TELNET is the standard TCP/IP protocol for virtual terminal.
 TELNET enables connection to a remote system in such a way that the local terminal appears to be a terminal at the remote system.
 TELNET was designed during days of time-sharing environment in which a large computer supported multiple users.

- The interaction between a user and the computer occurs through a terminal (keyboard + monitor + mouse).

 Each user has an identification name and a password.
 To access the system, the user logs into the system with a user id or log-in name.
 The user is authenticated using password and hence unauthorized access is prevented.

Remote Logon

The process of remote login using TELNET client and server program is depicted below.

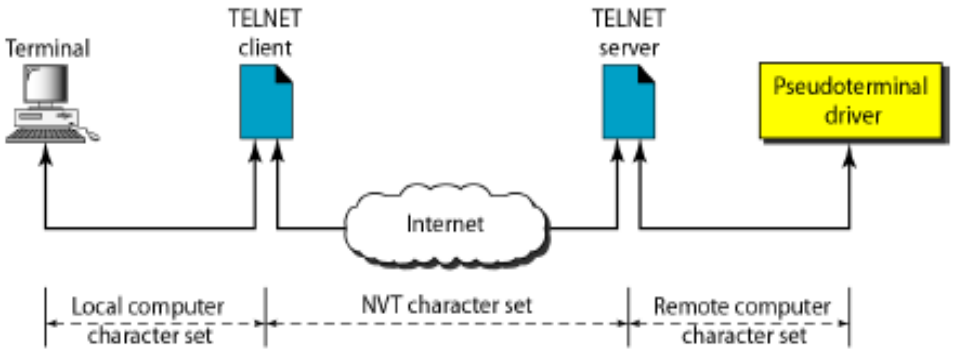


 The user keystrokes are sent to the terminal driver, where the local operating system accepts the characters but does not interpret them.
 The characters are sent to the *TELNET client*, which transforms the characters to a universal character set called *Network Virtual Terminal* (NVT) characters and puts it over the network.
 The commands/text in NVT form reaches the remote host.
 The *TELNET server* at well-known port 23, converts NVT characters onto remote character set.
 Since the operating system is not designed to receive data from TELNET server, data is redirected via a pseudo terminal driver to the remote OS.
 The remote OS passes the data to the corresponding applications.

NVT Character Set

 Every operating system use a special combination of characters as tokens
  - For example, the *end-of-file* token in DOS is Ctrl+z, whereas in UNIX it is Ctrl+d.

 TELNET solves the problem of heterogeneity, by defining a universal interface called the network virtual terminal (NVT) character set.
 Data transmitted over the network is NVT, whereas at the host level data is processed using its character set as shown below.



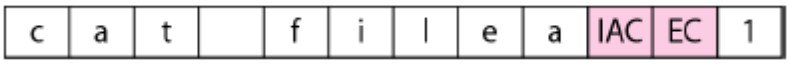
 NVT uses two sets of 8-bit characters, one for data and the other for control.

- For data, the MSB is 0 and for control it is 1.

 Some NVT control characters are:

Character	Purpose
EOF	End of file
EOR	End of record
IP	Interrupt process
AYT	Are you there
EC	Erase character
EL	Erase line
IAC	Interrupt as control

 TELNET uses the same connection to send both data and control characters.
 To distinguish data from control characters, each sequence of control characters is preceded by a special control character called IAC.
 For example, to display file1, the command is cat file1, by mistake the user types cat filea<*backspace*>1.



Options

 TELNET lets the client and server negotiate options before or during the session.
 Options are extra features available with a more sophisticated terminal whereas simple terminals use default features. Some options are

Options	Purpose
Echo	Echo the received data to the sender
Status	Request the status of TELNET
Line mode	Change to line mode.

 The control characters used for option negotiation are WILL, WONT, DO and DONT.

Modes

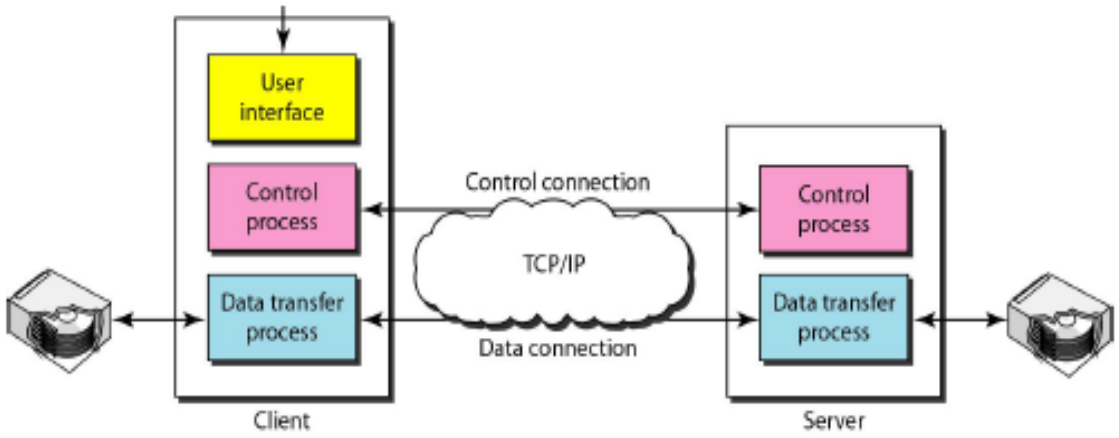
 TELNET operate in three modes namely *default*, *character* and *line* mode.

- In *default* mode, the client sends characters only after the line is typed.
- In *character* mode, each character typed is sent by the client to the server.
- In *line* mode, line editing is done by the client and sends after a line is typed

Briefly explain the transfer of file contents using FTP.

 File Transfer Protocol (FTP) is the standard provided by TCP/IP for copying a file from one host to another.
 FTP establishes two connections between hosts

- Data connection is used for data transfer
- Control connection is used for control information.
- FTP uses two well-known TCP ports, 21 for control and 20 for data connection.

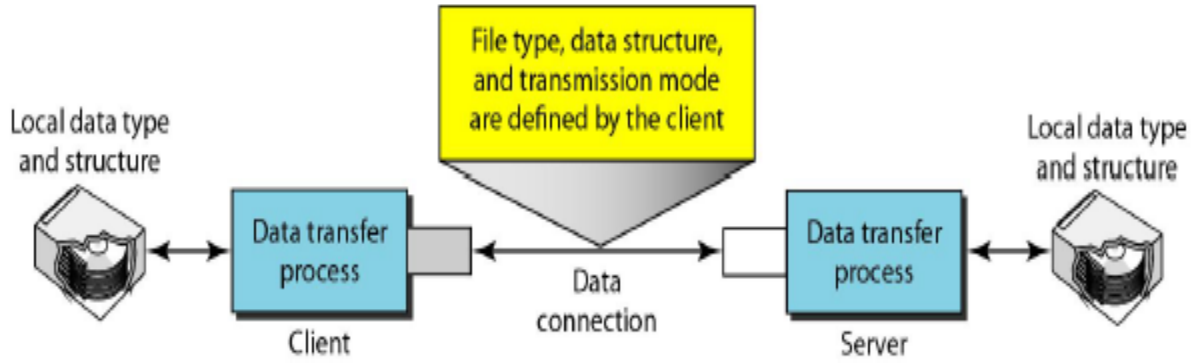


Control Connection

 FTP uses 7-bit NVT ASCII character set to communicate across the control connection.
 Communication is achieved through commands and responses.
 Each command or response is only one short line terminated with <CRLF>
 FTP uses well-known TCP port 21 for control communication
 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.

Data Connection

 File transfer occurs over the data connection under the control of the commands sent over the control connection.
 A file transfer in FTP means one of the following:
  - A file is to be copied from the server to the client. This is called *retrieving* a file. It is done under the supervision of the RETR command
  - A file is to be copied from the client to the server. This is called *storing* a file. It is done under the supervision of the STOR command.
  - A list of directory or file names is to be sent from the server to the client. This is done under the supervision of the LIST command.
 The client defines the *type* of file to be transferred, the *structure* of the data, and the *transmission* mode.
 Before sending the file through the data connection it is prepared for transmission through the control connection.



File Type

 FTP can transfer either an *ASCII* file, *EBCDIC* file, or *image* file.

- ASCII file is the default format for transferring text files.
- IBM uses EBCDIC encoding.
- The image file is the default format for transferring binary files.

Data Structure

 FTP interprets file's data structure as either *file*, *record* or *page* structure.

- In file structure, the file is a continuous stream of bytes.
- In record structure, the file is divided into records (used only for text files)
- In page structure, the file is divided into pages. Each page has a page number and header. Page access can be random or sequential.

Transmission Mode

 FTP uses *stream* (default), *block* or *compressed* mode of transmission.

- In stream mode, data is delivered to TCP as a continuous stream of bytes. If it's a file structure, end-of-file (EOF) is not needed. In case of record structure, each record is marked by a end-of-record (EOR) and the end of the file has a EOF character.
- In block mode, data is delivered to TCP in blocks, where each block is preceded by a 3-byte header. The first byte is the block descriptor and next 2 bytes define the size.
- In compressed mode, the compression used is run-length encoding. Consecutive appearance of character is replaced by an occurrence and count of repetitions.

Example

```
$ ftp voyager.deanza.tbda.edu
Connected to voyager.deanza.tbda.edu.
220 (vsFTPd 1.2.1)
530 Please login with USER and PASS.
Name (voyager.deanza.tbda.edu:forouzan): forouzan
331 Please specify the password.
Password:
230 Login successful.
ftp> Is reports
150 Here comes the directory listing.
drwxr-xr-x 23027 411 4096 Sep 24 2002 business
drwxr-xr-x 23027 411 4096 Sep 24 2002 personal
drwxr-xr-x 23027 411 4096 Sep 24 2002 school
226 Directory send OK.
```

What is anonymous FTP?

 To use FTP, a user should know user name and password on the remote server.
 Some sites have a set of files available for public access, to enable anonymous FTP.
 To access these files, a user does not need to have an account.
 User access to the system is very limited. For example, most sites allow the user to download files.

Write short notes on PGP.

 Pretty Good Privacy (PGP) is a popular approach in providing encryption and authentication capabilities for e-mail.

 PGP takes note that each user has his own set of criteria by which he/she wants to trust the keys certified by someone else.

- For example, one may trust signed certificates of co-workers than a renowned politician and vice-versa.

 PGP provides tools needed to manage the level of trust put in these certificates.

 PGP allows certification relationships to form an *arbitrary mesh* and not a rigid hierarchy as in Privacy Enhanced Mail (PEM).

 PGP allows each user to decide for themselves how much trust they wish to place in a given certificate

- As the number of trust-worthy signatures for a public key increase, validity for the same and the user's confidence level increases.

 PGP key-signing parties are a regular feature of network community meetings such as IETF. The activities include:

- Collect public keys from known persons.
- Share their public key with others
- Get their public key signed by others
- Sign public key of others
- Collect certificate from trust-worthy persons.

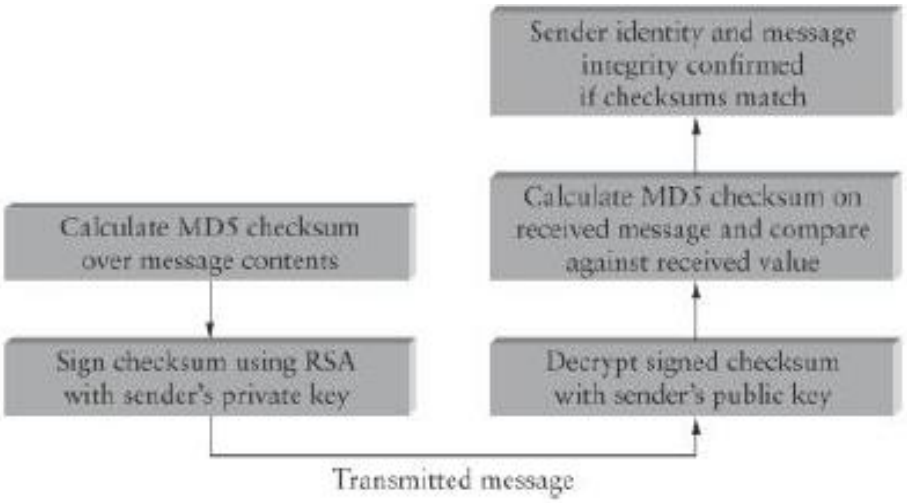
 PGP stores the set of collected certificates in a file called *key ring*.

 PGP allows a wide variety of different cryptographic algorithms to be used

- The actual algorithms used in a message are specified in header fields

 PGP allows a user to list his preferred algorithms in the file that contains his/her public key.

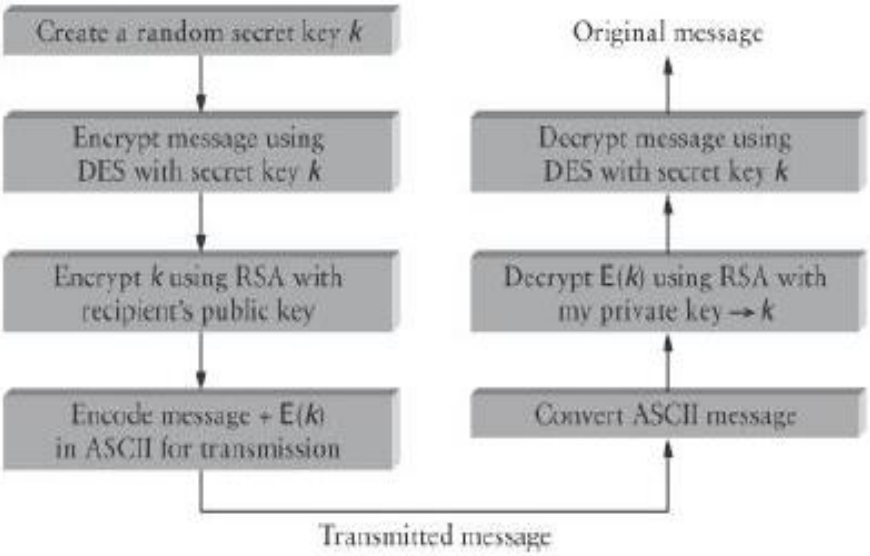
Message Transmission



1. A creates a cryptographic checksum over the message body, such as MD5 and then encrypts the checksum using A's private key.
2. On receipt of the message, B uses PGP's key management software to search his key ring for *A*'s public key.
3. If the key is found

a. Checksum of the received message is calculated
b. Encrypted checksum is decrypted using A's public key,
c. The two checksums are compared. If both are same, then it confirms that A has sent the message and its integrity.
4. If key is not found, the sender and authenticity of the message cannot be verified.
5. Apart from signature verification, PGP tells B the level of trust previously assigned to this public key

Message Encryption



1. A randomly picks a per-message key k to encrypt the message using a symmetric algorithm such as DES
2. The per-message key k is encrypted using B's public key
3. PGP obtains B's public key from A's key ring and notifies A of the level of trust assigned to this key.
4. On receipt, B uses its private key to decrypt the per-message key k.
5. The same algorithm is applied to decrypt the message using per-message key k.

Write short notes on SSH.

- Secure Shell (SSH) provides a remote login service in a secure manner
- SSH uses well-known port 22.
- SSH is used to provide strong client/server authentication
  - Passwords are not sent as clear text over the network. It is sent in encrypted form.
  - Thus sending password through un-trusted network is not a problem
- Unlike Telnet and rlogin, SSH supports message integrity and confidentiality
- SSH version 2 consist of the following protocols
  - Transport layer protocol SSH-TRANS
  - Authentication protocol SSH-AUTH
  - Connection protocol SSH-CONN

SSH-TRANS

 SSH-TRANS provides an encrypted channel for communication.
 It runs on top of a TCP connection.
 Client and server establish secure channel by having the client authenticate the server using RSA.

- Server informs the client of its public key at the time of connection
- Client warns the user when it tries to connect to the server for the first time, since it does not know the server

 Once authenticated, the client and server establish a session key that they will use to encrypt any data sent over the channel.

- Client remembers the server's public key
- For future connection, the client compares server's response with the saved key

 SSH-TRANS includes a negotiation of the encryption algorithm the two sides are going to use. For example, 3DES is commonly selected.
 SSH-TRANS includes a message integrity check of all data exchanged over the channel.

SSH-AUTH

 Server is authenticated during setup of SSH-TRANS channel by default
 User can authenticate using any of the three mechanism

1) *Login* with username and password. Password is sent in encrypted form
2) *Public key* encryption by asking the user to store user's public key on the server
3) *Host based* authentication requires the client to be authenticated when it connects to server for the first time. Further connection from a trusted host is believed to be from the same user.

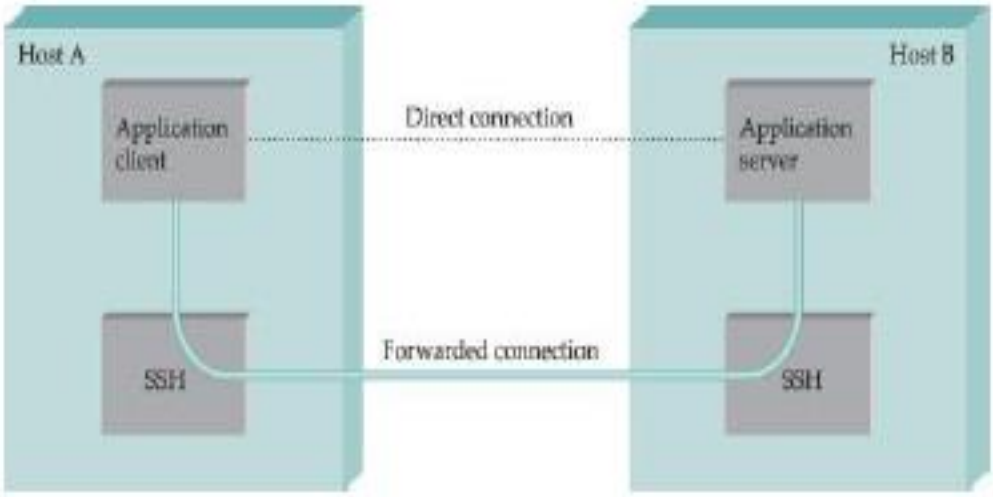
 In UNIX,

- /.ssh/known_hosts records the keys for all the hosts the user has logged into
- /.ssh/authorized_keys contains the public keys needed to authenticate the user when he or she logs into this machine
- /.ssh/identity contains the private keys for authenticating user on remote machine

SSH-CONN

 SSH can be extended to support insecure TCP applications such as X Windows, IMAP mail readers, etc using SSH-CONN.
 Insecure applications are run by tunneling through SSH, known as *port forwarding*.

- Client on host *A* communicates with server on host *B* using SSH.
- Client data sent through SSH is encrypted at sender side
- The receiving SSH at well-known port decrypts the contents
- Content is forwarded to the actual port on which the server is listening



What is Web-based mail?

 E-mail is such a common application that some websites today provide this service to anyone who accesses the site such as Hotmail, Yahoo, etc.
 Mail transfer from Alice's browser to her mail server is done through HTTP
 The message transfer from sending mail server to receiving mail server is through SMTP
 Finally, the message from the receiving Web server to Bob's browser is done using HTTP
 The website sends a form to be filled in by Bob, which includes log-in id and password.
 If the credentials match, the e-mail is transferred from Web server to Bob's browser in HTML format.