UNIT-I: INTRODUCTION

1.1. Data Communication and Components

1.1.1. *Data Communication*:

When we communicate, we are sharing information. This sharing can be local or remote. Between individuals, local communication usually occurs face to face, while remote communication takes place over distance.

Data communications are the exchange of data between two devices via some form of transmission medium such as a wire cable or wireless. For data communications, the communicating devices must be part of a communication system made up of a combination of hardware (physical equipment) and software (programs). The effectiveness of a data communications system depends on four fundamental characteristics: delivery, accuracy, timeliness, and jitter.

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

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

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

4. Jitter: Jitter refers to the variation in the packet arrival time. It is the uneven delay in the delivery of audio or video packets. For example, let us assume that video packets are sent every 30-ms. If some of the packets arrive with 30-ms delay and others with 40-ms delay, an uneven quality in the video is the result.

1.1.2. *Components*:

A data communications system has five components (Refer fig 1.1)

1. Message: The message is the information (data) to be communicated. Popular forms of information include text, numbers, pictures, audio, and video.

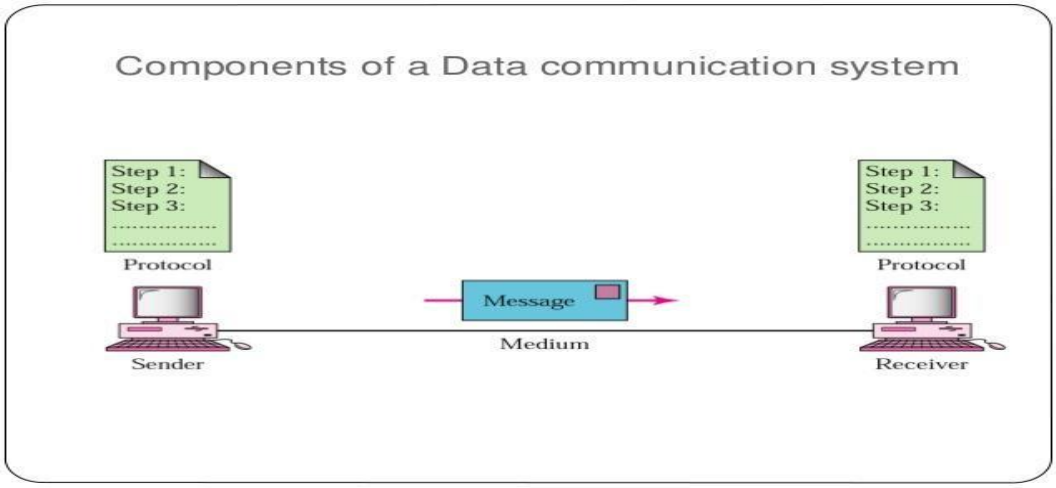


Figure 1.1

2. Sender: The sender is the device that sends the data message. It can be a computer, workstation, telephone handset, video camera, and so on.

3. Receiver: The receiver is the device that receives the message. It can be a computer, workstation, telephone handset, television, and so on.

4. Transmission medium: The transmission medium is the physical path by which a message travels from sender to receiver. Some examples of transmission media include twisted-pair wire, coaxial cable, fiber-optic cable, and radio waves.

5. Protocol: A protocol is a set of rules that govern data communications. It represents an agreement between the communicating devices. Without a protocol, two devices may be connected but not communicating, just as a person speaking French cannot be understood by a person who speaks only Japanese.

1.1.2. Data Representation

Information today comes in different forms such as text, numbers, images, audio, and video.

Text:

In data communications, text is represented as a bit pattern, a sequence of bits. Different sets of bit patterns have been designed to represent text symbols. Each set is called a code, and the process of representing symbols is called coding. Today, the prevalent coding system is called Unicode, which uses 32 bits to represent a symbol or character used in any language in the world. The American Standard Code for

Information Interchange (ASCII) developed some decades ago in the United States, now constitutes the first 127 characters in Unicode and is also referred to as Basic Latin.

Numbers:

Numbers are also represented by bit patterns. However, a code such as ASCII is not used to represent numbers; the number is directly converted to a binary number to simplify mathematical operations. Appendix B discusses several different numbering systems.

Images:

Images are also represented by bit patterns. In its simplest form, an image is composed of a matrix of pixels (picture elements), where each pixel is a small dot. The size of the pixel depends on the resolution. For example, an image can be divided into 1000 pixels or 10,000 pixels. In the second case, there is a better representation of the image (better resolution), but more memory is needed to store the image. After an image is divided into pixels, each pixel is assigned a bit pattern. The size and the value of the pattern depend on the image. For an image made of only black and white dots (e.g., a chessboard), a I-bit pattern is enough to represent a pixel. If an image is not made of pure white and pure black pixels, you can increase the size of the bit pattern to include gray scale. For example, to show four levels of gray scale, you can use 2-bit patterns. A black pixel can be represented by 00, a dark gray pixel by 01, a light gray pixel by 10, and a white pixel by 11. There are several methods to represent colour images. One method is called RGB, so called because each colour is made of a combination of three primary colours: red, green, and blue. The intensity of each colour is measured, and a bit pattern is assigned to it. Another method is called YCM, in which a colour is made of a combination of three other primary colours: yellow, cyan, and magenta.

Audio:

Audio refers to the recording or broadcasting of sound or music. Audio is by nature different from text, numbers, or images. It is continuous, not discrete. Even when we use a microphone to change voice or music to an electric signal, we create a continuous signal. In

Video:

Video refers to the recording or broadcasting of a picture or movie. Video can either be produced as a continuous entity (e.g., by a TV camera), or it can be a combination of images, each a discrete entity, arranged to convey the idea of motion. Again we can change video to a digital or an analog signal.

1.1.3. Data Flow:

Communication between two devices can be simplex, half-duplex, or full-duplex as shown in Figure

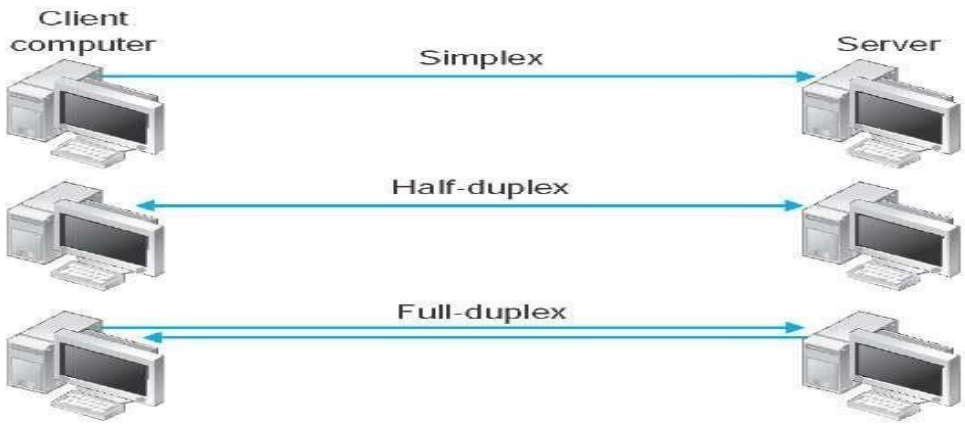


Figure 1.2

Simplex:

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

Half-Duplex:

In half-duplex mode, each station can both transmit and receive, but not at the same time. When one device is sending, the other can only receive, and vice versa. The half-duplex mode is like a one-lane road with traffic allowed in both directions. When cars are travelling in one direction, cars going the other way must wait. In a half-duplex

transmission, the entire capacity of a channel is taken over by whichever of the two devices is transmitting at the time. Walkie-talkies and CB (citizens band) radios are both half-duplex systems. The half-duplex mode is used in cases where there is no need for communication in both directions at the same time; the entire capacity of the channel can be utilized for each direction.

Full-Duplex:

In full-duplex mode (also called duplex), both stations can transmit and receive simultaneously. The full-duplex mode is like a two-way street with traffic flowing in both directions at the same time. In full-duplex mode, signals going in one direction share the capacity of the link with signals going in the other direction. This sharing can occur in two ways: Either the link must contain two physically separate transmission paths, one for sending and the other for receiving; or the capacity of the channel is divided between signals travelling in both directions. One common example of full-duplex communication is the telephone network. When two people are communicating by a telephone line, both can talk and listen at the same time. The full-duplex mode is used when communication in both directions is required all the time. The capacity of the channel, however, must be divided between the two directions.

1.2. Network Models

A network is a set of devices (often referred to as nodes) connected by communication links. A node can be a computer, printer, or any other device capable of sending and/or receiving data generated by other nodes on the network.

1.2.1. Distributed Processing

Most networks use distributed processing, in which a task is divided among multiple computers. Instead of one single large machine being responsible for all aspects of process, separate computers (usually a personal computer or workstation) handle a subset.

1.2.2. Network Criteria

A network must be able to meet a certain number of criteria. The most important of these are performance, reliability, and security.

Performance:

Performance can be measured in many ways, including transit time and response time. Transit time is the amount of time required for a message to travel from one device to another. Response time is the elapsed time between an inquiry and a response. The performance of a network depends on a number of factors, including the number of users, the type of transmission medium, the capabilities of the connected hardware, and the efficiency of the software. Performance is often evaluated by two networking metrics: throughput and delay. We often need more throughput and less delay. However, these two criteria are often contradictory. If we try to send more data to the network, we may increase throughput but we increase the delay because of traffic congestion in the network.

Reliability:

In addition to accuracy of delivery, network reliability is measured by the frequency of failure, the time it takes a link to recover from a failure, and the network's robustness in a catastrophe.

Security:

Network security issues include protecting data from unauthorized access, protecting data from damage and development, and implementing policies and procedures for recovery from breaches and data losses.

1.2.3. Physical Structure

Type of Connection

A network is two or more devices connected through links. A link is a communications pathway that transfers data from one device to another. For visualization purposes, it is simplest to imagine any link as a line drawn between two points. For communication to occur, two devices must be connected in some way to the same link at the same time. There are two possible types of connections: point-to-point and multipoint. Point-to-Point

A point-to-point connection provides a dedicated link between two devices. The entire capacity of the link is reserved for transmission between those two devices. Most point-to-point connections use an actual length of wire or cable to connect the two ends, but other options, such as microwave or satellite links, are also possible. When you change television channels by infrared remote control, you are establishing a point-to-point connection between the remote control and the television's control system.

Multipoint

A multipoint (also called multi-drop) connection is one in which more than two specific devices share a single link. In a multipoint environment, the capacity of the channel is shared, either spatially or temporally. If several devices can use the link simultaneously, it is a spatially shared connection. If users must take turns, it is a timeshared connection.

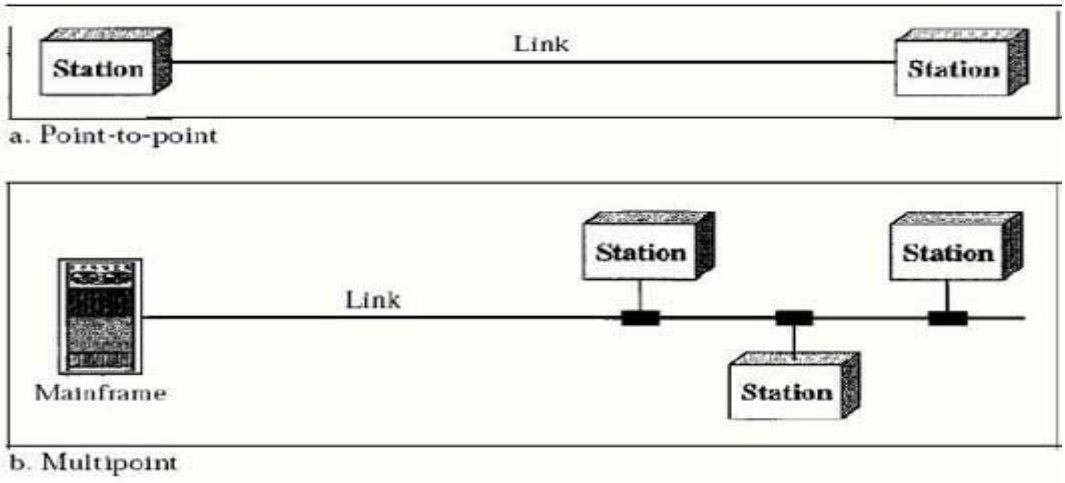


Figure 1.3

1.2.3.1. Physical Topology

The term physical topology refers to the way in which a network is laid out physically. One or more devices connect to a link; two or more links form a topology. The topology of a network is the geometric representation of the relationship of all the links and linking devices (usually called nodes) to one another. There are four basic topologies possible: bus, mesh, ring and star.

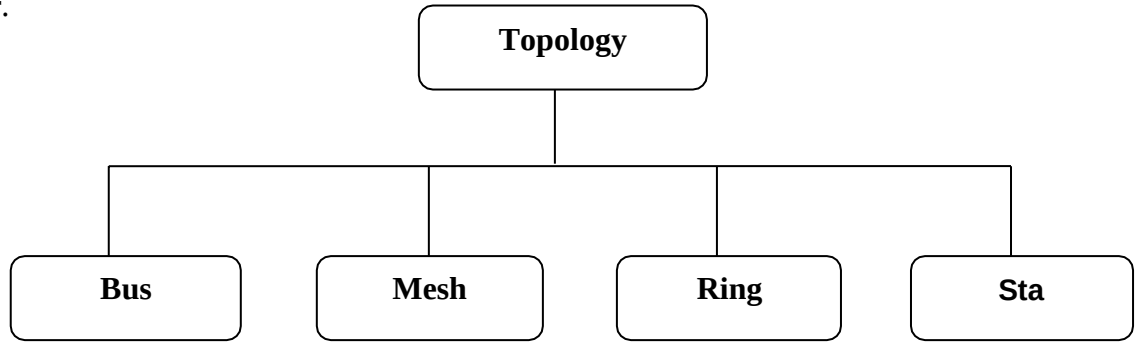


Figure 1.4

Mesh Topology:

In a mesh network, devices are connected with many redundant interconnections between network nodes. In a true mesh topology every node has a connection to every other node in the network.

There are two types of mesh topologies:

Full mesh topology: occurs when every node has a circuit connecting it to every other node in a network. Full mesh is very expensive to implement but yields the greatest amount of redundancy, so in the event that one of those nodes fails, network traffic can be directed to any of the other nodes. Full mesh is usually reserved for backbone networks.

Partial mesh topology: is less expensive to implement and yields less redundancy than full mesh topology. With partial mesh, some nodes are organized in a full mesh scheme but others are only connected to one or two in the network. Partial mesh topology is commonly found in peripheral networks connected to a full meshed backbone.

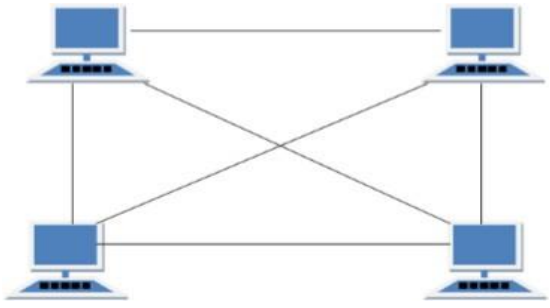


Figure 1.5

Features of Mesh Topology

1. Fully connected. 2. Robust. 3. Not flexible.

Advantages of Mesh Topology

1. Each connection can carry its own data load. 2. It is robust. 3. Fault is diagnosed easily.
4. Provides security and privacy.

Disadvantages of Mesh Topology

1. Installation and configuration is difficult. 2. Cabling cost is more. 3. Bulk wiring is required.

Star Topology:

In a star topology, each device has a dedicated point-to-point link only to a central controller, usually called a hub. The devices are not directly linked to one another. Unlike a mesh topology, a star topology does not allow direct traffic between devices. The controller acts as an exchange: If one device wants to send data to another, it sends the data to the controller, which then relays the data to the other connected device.

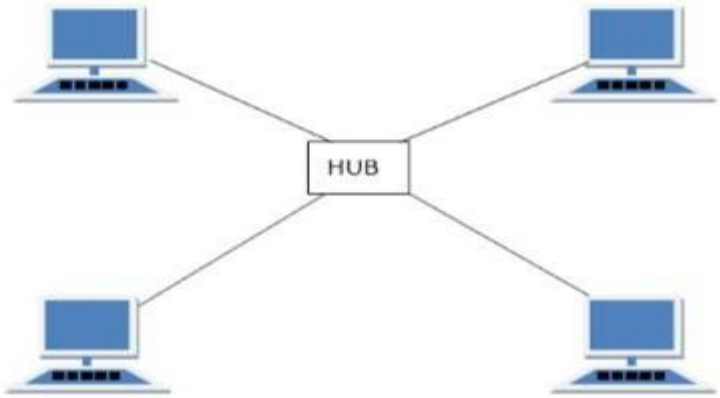


Figure 1.6

Features of Star Topology

1. Every node has its own dedicated connection to the hub. 2. Hub acts as a repeater for data flow. 3. Can be used with twisted pair, Optical Fiber or coaxial cable.

Advantages of Star Topology

1. Fast performance with few nodes and low network traffic. 2. Hub can be upgraded easily. 3. Easy to troubleshoot. 4. Easy to setup and modify. 5. Only that node is affected which has failed, rest of the nodes can work smoothly.

Disadvantages of Star Topology

1. Cost of installation is high. 2. Expensive to use. 3. If the hub fails then the whole network is stopped because all the nodes depend on the hub. 4. Performance is based on the hub that is it depends on its capacity

Bus Topology:

In networking a bus is the central cable -- the main wire -- that connects all devices on a local area network (LAN). It is also called the backbone. This is often used to describe the main network connections composing the Internet. Bus networks are relatively inexpensive and easy to install for small networks. Ethernet systems use a bus topology.

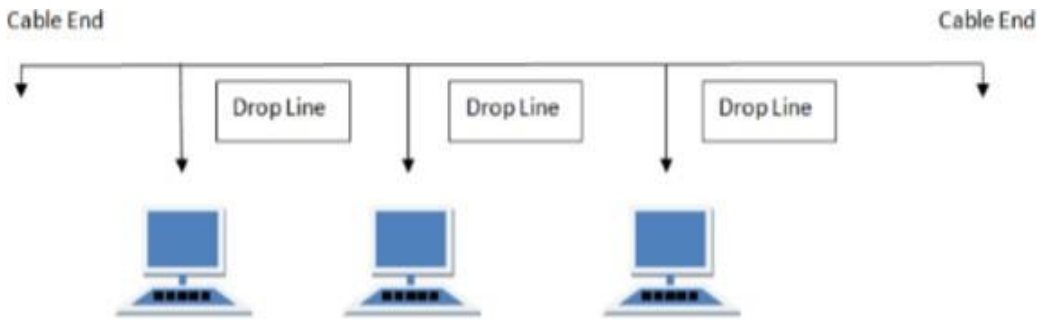


Figure 1.7

Features of Bus Topology

1. It transmits data only in one direction. 2. Every device is connected to a single cable

Advantages of Bus Topology

1. It is cost effective. 2. Cable required is least compared to other network topology. 3. Used in small networks. 4. It is easy to understand. 5. Easy to expand joining two cables together.

Disadvantages of Bus Topology

1. Cables fails then whole network fails. 2. If network traffic is heavy or nodes are more the performance of the network decreases. 3. Cable has a limited length. 4. It is slower than the ring topology.

Ring Topology:

Ring Topology: A local-area network (LAN) whose topology is a ring. That is, all of the nodes are connected in a closed loop. Messages travel around the ring, with each node reading those messages addressed to it.

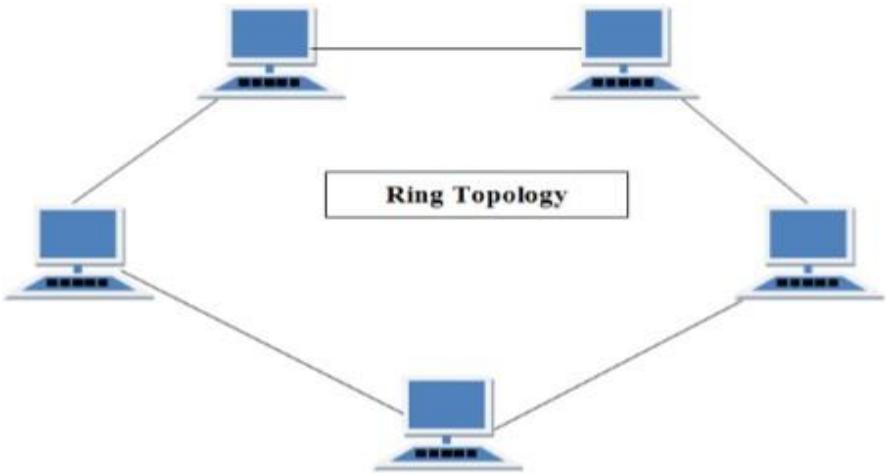


Figure 1.8

Features of Ring Topology

1. A number of repeaters are used for Ring topology with large number of nodes, because if someone wants to send some data to the last node in the ring topology with 100 nodes, then the data will have to pass through 99 nodes to reach the 100th node. Hence to prevent data loss repeaters are used in the network. 2. The transmission is unidirectional, but it can be made bidirectional by having 2 connections between each Network Node, it is called Dual Ring Topology. 3. In Dual Ring Topology, two ring networks are formed, and data flow is in opposite direction in them. Also, if one ring fails, the second ring can act as a backup, to keep the network up. 4. Data is transferred in a sequential manner that is bit by bit. Data transmitted, has to pass through each node of the network, till the destination node.

Advantages of Ring Topology

1. Transmitting network is not affected by high traffic or by adding more nodes, as only the nodes having tokens can transmit data. 2. Cheap to install and expand

Disadvantages of Ring Topology

1. Troubleshooting is difficult in ring topology. 2. Adding or deleting the computers disturbs the network activity. 3. Failure of one computer disturbs the whole network.

1.2.4. *Network Hardware*

There is no generally accepted taxonomy into which all computer networks fit, but two dimensions stand out as important: transmission technology and scale.

Classification of networks according to transmission technology:

- Broadcast networks,
- Point-to-Point networks.

Broadcast networks are networks with single communication channel shared by all the machines. Short messages (packets) sent by any machine are received by all others. An address field within the packet specifies for whom it is intended. Analogy: someone shout in the corridor with many rooms.
Broadcasting is a mode of operation in which a packet is sent to every machine using a special code in the address field.
Multicasting is sending a packet to a subset of the machines.
Point-to-point networks consist of many connections between individual pairs of machines. In these types of networks:

- A packet on its way from the source to the destination may go through intermediate machines.
- In general, multiple routes are possible - routing algorithms are necessary.

General rule (with many exceptions): smaller, geographically localized networks tends to use broadcasting, larger networks usually are point-to-point.

Classification of networks by scale: If we take as a criterion the interprocessor distance, we get on the one side of the scale data flow machines, highly parallel computers with many functional units all working on the same program. Next come the multicomputers, systems that communicate through short, very fast buses. Beyond the multicomputers are the true networks, computers communicating over longer cables. Finally, the connection of two or more networks is called an internetwork. Distance is important as a classification metric because different techniques are used at different scales.

1.2.4.1. Local Area Networks

Local area networks (LANs) are privately-owned, within a single building or campus, of up to a few kilometers in size. They are distinguished from other kind of networks by three characteristics:

- Size,

- Transmission technology,
- Topology.

LANs are restricted in size - the worst-case transmission time is known in advance, it makes possible to use certain kinds of design.

LANs transmission technology often consists of a single cable to which all machines are attached. Traditional LANs run at speed of 10 to 100 Mbps. Newer LANs may operate at higher speeds.

Possible topologies for broadcast LANs

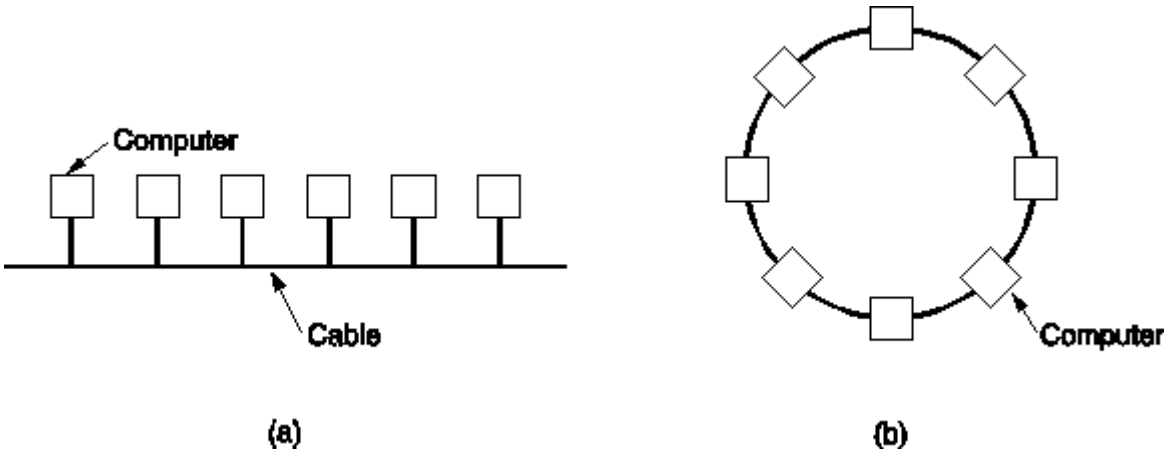


Figure 1.9. Two broadcast networks. (a) Bus. (b) Ring.

- *Bus* - At any instant one machine is the master of the bus allowed to transmit. Arbitration mechanism for resolving the conflicts when more than one machine want to transmit may be centralized or distributed. Example: Ethernet as a bus-based broadcast network with decentralized control operating at 10 or 100 Mbps.
- *Ring* - Each bit propagates around, typically it circumnavigates the entire ring in the time it takes to transmit a few bits, often before the complete packet has even be transmitted. Example: IBM token ring operating at 4 and 16 Mbps.

Broadcast networks can be, depending on how the channel is allocated, further divided into:

- *Static* - a typical would be a time division for the access to the channel and round-robin algorithms. It wastes channel capacity.
- *Dynamic* - on demand. Channel allocation could be centralized or decentralized.

LAN built using point-to-point lines is really a miniature WAN.

1.2.4.2. Metropolitan Area Networks

Metropolitan area network (MAN) is basically a bigger version of a LAN and normally uses similar technology. It might cover a group of nearby corporate offices or a city and might be either private or public. The main reason for even distinguishing MANs as a special category is that a standard has been adopted for them. It is called DQDB (Distributed Queue Dual Bus).

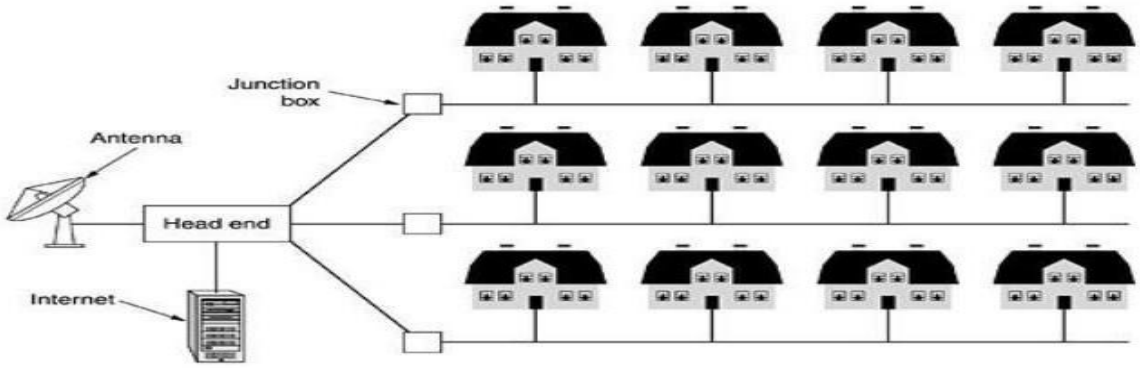


Figure 1.10 Metropolitan area network based on cable TV.

1.2.4.3. Wide Area Networks

A *wide area network* (WAN):

- Spans a large geographical area,
- Contains hosts (or end-systems) intended for running user programs,
- The hosts are connected by a *subnet* that carries messages from host to host.

The subnet usually consists of transmission lines (circuits, channels, or trunks) and switching elements. The switching elements are specialized computers used to connect two or more transmission lines. There is no standard technology used to name switching elements (e.g.

packet switching nodes, intermediate systems, data switching exchanges). As a generic term we will use the word router.

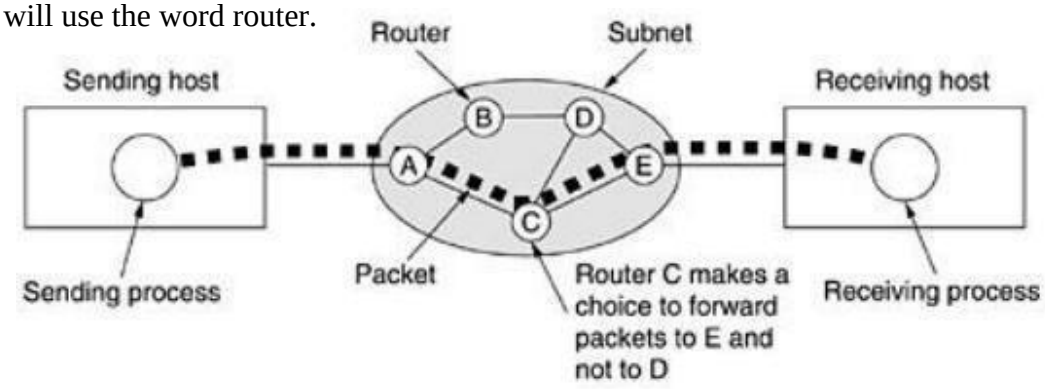


Figure 1.11. A stream of packets from sender to receiver.

If two routers that do not share a cable wish to communicate, they must do it via other routers. When a packet is sent from one router to another via intermediate routers, the packet is received at each intermediate router, stored there until the required output line is free, and then forwarded. A subnet using this principle is called point-to-point, store-and-forward, or packet-switched subnet. Nearly all wide area networks (except those using satellites) have store-and-forward subnets.

When the packets are small and all the same size, they are often called cells.

A second possibility for a WAN is a satellite or ground radio system. Each router has an antenna through which it can send and receive. All routers can hear the output from the satellite. Satellite networks are inherently broadcast.

1.2.4.4. Wireless Networks

The owners of mobile computers want to be connected to their home base when they are away from home. In case where wired connection is impossible (in cars, airplanes), the wireless networks are necessary.

The use of wireless networks:

- Portable office - sending and receiving telephone calls, faxes, e-mails, remote login, ...
- Rescue works,
- Keeping in contact,
- Military.

Wireless networking and *mobile computing* are often related but they are not identical. Portable computers are sometimes wired (e.g. at the traveler's stay in a hotel) and some wireless computer are not portable (e.g. in the old building without any network infrastructure).

Wireless LANs are easy to install but they have also some disadvantages: lower capacity (1-2 Mbps, higher error rate, possible interference of the transmissions from different computers).

Wireless networks come in many forms:

- Antennas all over the campus to allow to communicate from under the trees,
- Using a cellular (i.e. portable) telephone with a traditional analog modem,
- Direct digital cellular service called CDPD (Cellular Digital Packet Data),
- Different combinations of wired and wireless networking.

1.2.4.5. Internetworks

Internetwork or *internet* is a collection of interconnected networks. A common form of internet is a collection of LAN connected by WAN. Connecting incompatible networks together requires using machines called gateways to provide the necessary translation.

Subnets, networks and internetworks are often confused.

> Subnet makes the most sense in the context of a wide area network, where it refers to the collection of routers and communication lines. The combination of a subnet and its hosts forms a network. An internetwork is formed when distinct networks are connected together.

1.3. OSI Layers

The OSI model (minus the physical medium) is shown in Fig. This model is based on a proposal developed by the International Standards Organization (ISO) as a first step toward international standardization of the protocols used in the various layers (Day and Zimmermann, 1983). It was revised in 1995(Day, 1995). The model is called the ISO-OSI (Open Systems Interconnection) Reference Model because it deals with connecting open systems—that is, systems that are open for communication with other systems.

The OSI model has seven layers. The principles that were applied to arrive at the seven layers can be briefly summarized as follows:

1. A layer should be created where a different abstraction is needed.
2. Each layer should perform a well-defined function.

3. The function of each layer should be chosen with an eye toward defining internationally standardized protocols.
4. The layer boundaries should be chosen to minimize the information flow across the interfaces.
5. The number of layers should be large enough that distinct functions need not be thrown together in the same layer out of necessity and small enough that the architecture does not become unwieldy.

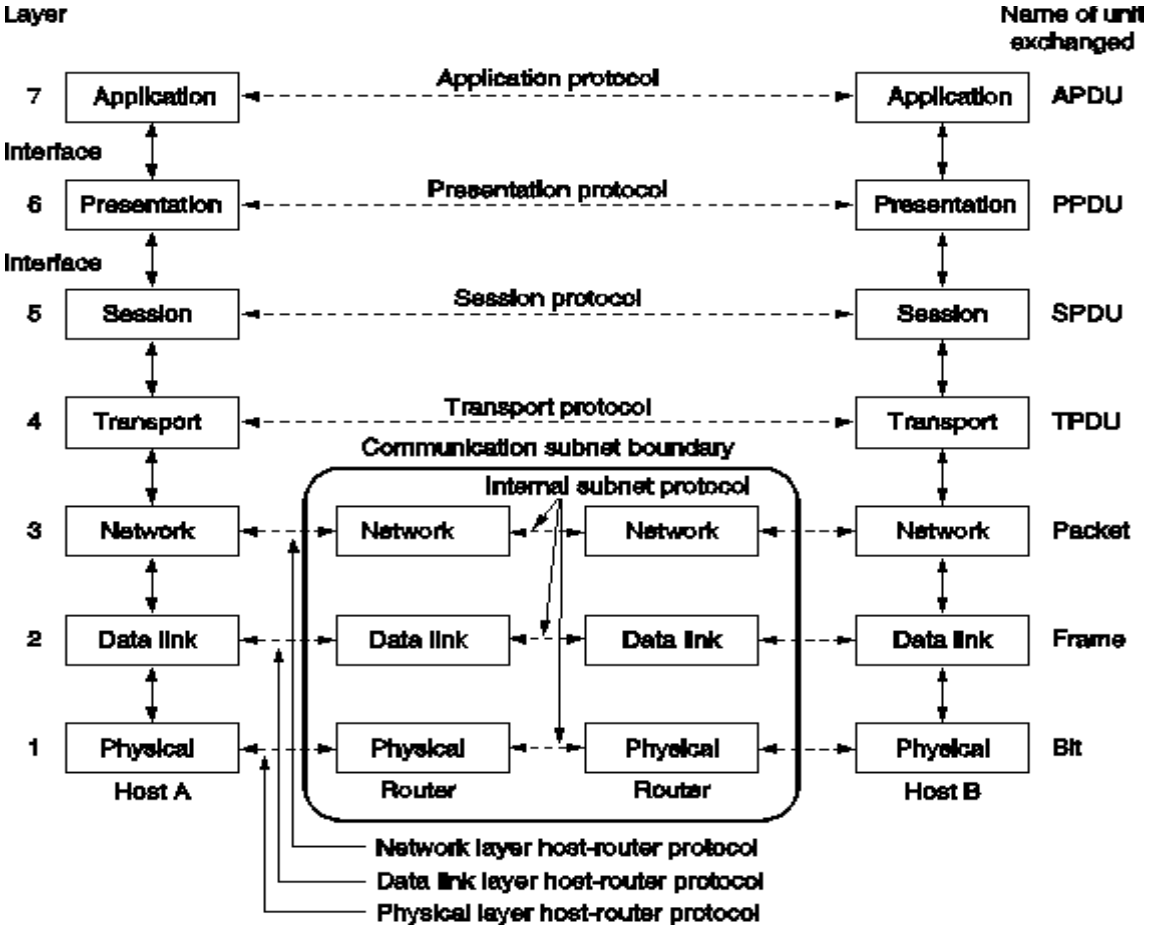


Fig 1.12a

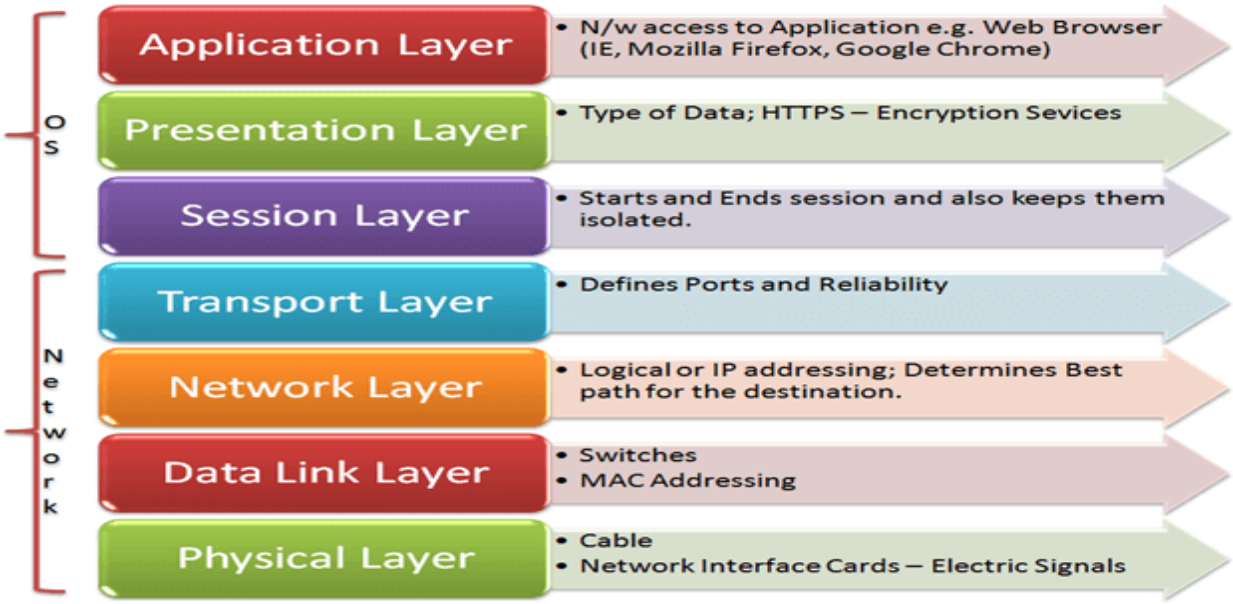


Fig 1.12b

Figure 1.12 a, b. OSI Reference model

The Physical Layer:

The physical layer is concerned with transmitting raw bits over a communication channel. The design issues have to do with making sure that when one side sends a 1 bit, it is received by the other side as a 1 bit, not as a 0 bit.

The Data Link Layer:

The main task of the data link layer is to transform a raw transmission facility into a line that appears free of undetected transmission errors to the network layer. It accomplishes this task by having the sender break up the input data into data frames (typically a few hundred or a few thousand bytes) and transmits the frames sequentially. If the service is reliable, the receiver confirms correct receipt of each frame by sending back an acknowledgement frame.

Another issue that arises in the data link layer (and most of the higher layers as well) is how to keep a fast transmitter from drowning a slow receiver in data. Some traffic regulation mechanism is often needed to let the transmitter know how much buffer space the receiver has at the moment. Frequently, this flow regulation and the error handling are integrated.

The Network Layer:

The network layer controls the operation of the subnet. A key design issue is determining how packets are routed from source to destination. Routes can be based on static tables that are "wired into" the network and rarely changed. They can also be determined at the start of each conversation, for example, a terminal session (e.g., a login to a remote machine). Finally, they can be highly dynamic, being determined anew for each packet, to reflect the current network load.

If too many packets are present in the subnet at the same time, they will get in one another's way, forming bottlenecks. The control of such congestion also belongs to the network layer. More generally, the quality of service provided (delay, transit time, jitter, etc.) is also a network layer issue.

When a packet has to travel from one network to another to get to its destination, many problems can arise. The addressing used by the second network may be different from the first one. The second one may not accept the packet at all because it is too large. The protocols may differ, and so on. It is up to the network layer to overcome all these problems to allow heterogeneous networks to be interconnected. In broadcast networks, the routing problem is simple, so the network layer is often thin or even nonexistent.

The Transport Layer:

The basic function of the transport layer is to accept data from above, split it up into smaller units if need be, pass these to the network layer, and ensure that the pieces all arrive correctly at the other end. Furthermore, all this must be done efficiently and in a way that isolates the upper layers from the inevitable changes in the hardware technology. The transport layer also determines what type of service to provide to the session layer, and, ultimately, to the users of the network. The most popular type of transport connection is an error-free point-to-point channel that delivers messages or bytes in the order in which they were sent. However, other possible kinds of transport service are the transporting of isolated messages, with no guarantee about the order of delivery, and the broadcasting of messages to multiple destinations. The type of service is determined when the connection is established.

The transport layer is a true end-to-end layer, all the way from the source to the destination. In other words, a program on the source machine carries on a conversation with a similar program on the destination machine, using the message headers and control messages. In the lower layers, the protocols are between each machine and its immediate neighbors, and not

between the ultimate source and destination machines, which may be separated by many routers.

The Session Layer:

The session layer allows users on different machines to establish sessions between them. Sessions offer various services, including dialog control (keeping track of whose turn it is to transmit), token management (preventing two parties from attempting the same critical operation at the same time), and synchronization (check pointing long transmissions to allow them to continue from where they were after a crash).

The Presentation Layer:

The presentation layer is concerned with the syntax and semantics of the information transmitted. In order to make it possible for computers with different data representations to communicate, the data structures to be exchanged can be defined in an abstract way, along with a standard encoding to be used "on the wire." The presentation layer manages these abstract data structures and allows higher-level data structures (e.g., banking records), to be defined and exchanged.

The Application Layer:

The application layer contains a variety of protocols that are commonly needed by users. One widely-used application protocol is HTTP (Hypertext Transfer Protocol), which is the basis for the World Wide Web. When a browser wants a Web page, it sends the name of the page it wants to the server using HTTP. The server then sends the page back. Other application protocols are used for file transfer, electronic mail, and network news.

1.3.1. Data Transmission in OSI Model

The key idea throughout is that although actual data transmission is vertical in Fig. 1.13, each layer is programmed as though it were horizontal. When the sending transport layer, for example, gets a message from the session layer, it attaches a transport header and sends it to the receiving transport layer. From its point of view, the fact that it must actually hand the message to the network layer on its own message is an unimportant technicality.

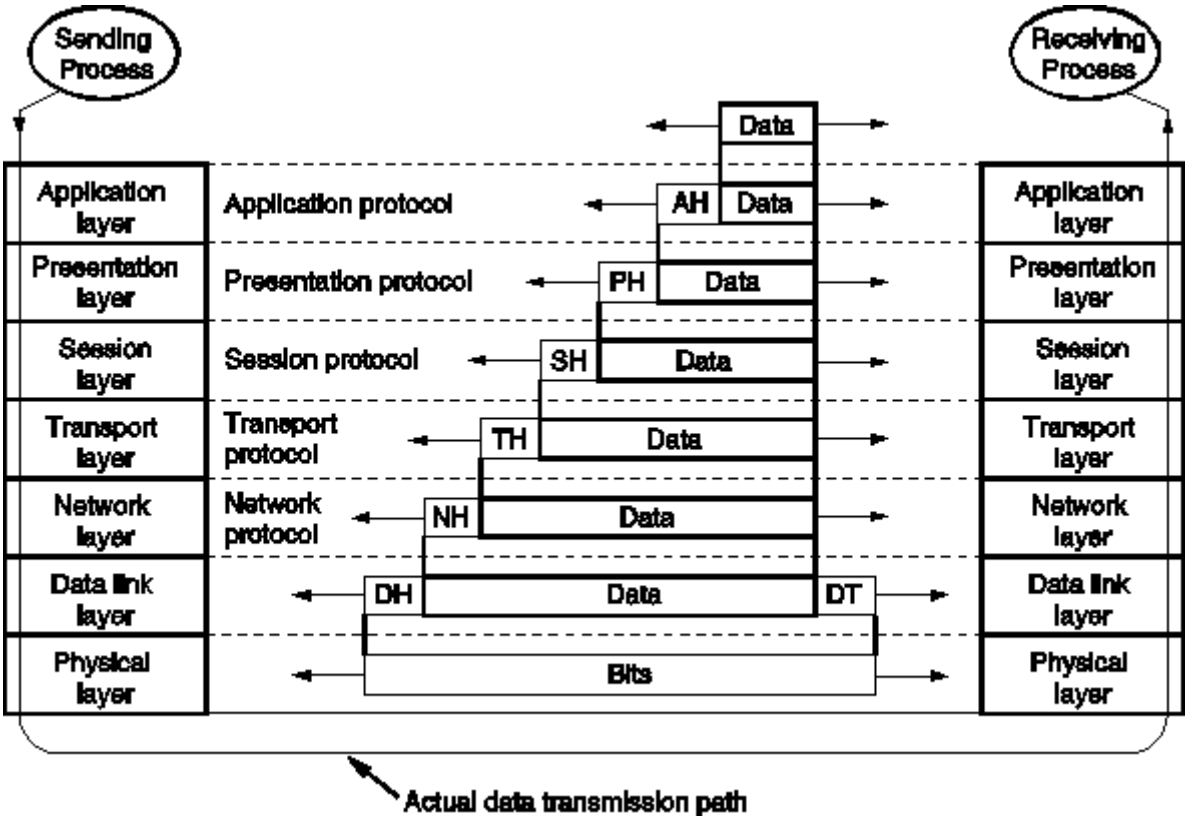


Figure 1.13 Example of how the OSI model is used. Some of the headers may be null.

1.4. TCP/IP Protocol Suit

TCP/IP reference model originates from the grandparent of all computer networks, the ARPANET and now is used in its successor, the worldwide Internet.

The name TCP/IP of the reference model is derived from two primary protocols of the corresponding network architecture.

1.4.1. The Internet Layer

The internet layer is the linchpin of the whole architecture. It is a connectionless internetwork layer forming a base for a packet-switching network. Its job is to permit hosts to inject packets into any network and have them travel independently to the destination. It works in analogy with the (snail) mail system. A person can drop a sequence of international letters into a mail box in one country, and with a little luck, most of them will be delivered to the correct address in the destination country.

The internet layer defines an official packet format and protocol called IP (Internet Protocol). The job of the internet layer is to deliver IP packets where they are supposed to go. TCP/IP internet layer is very similar in functionality to the OSI network layer (Fig. 1-18).

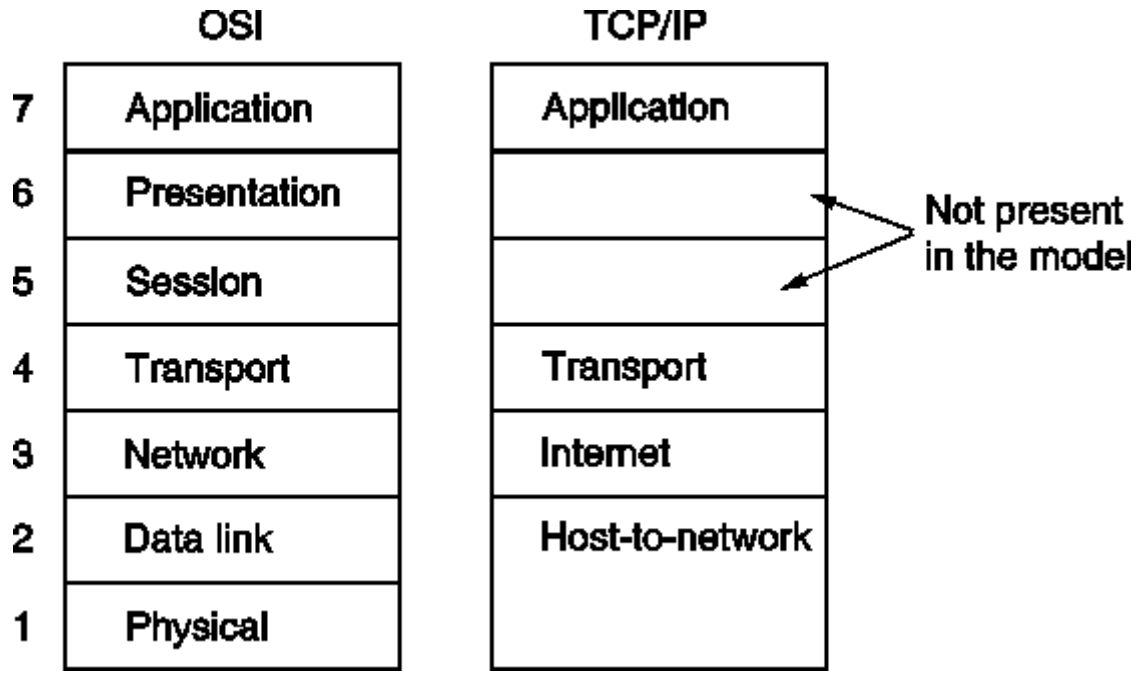


Figure 1.14. The TCP/IP reference model.

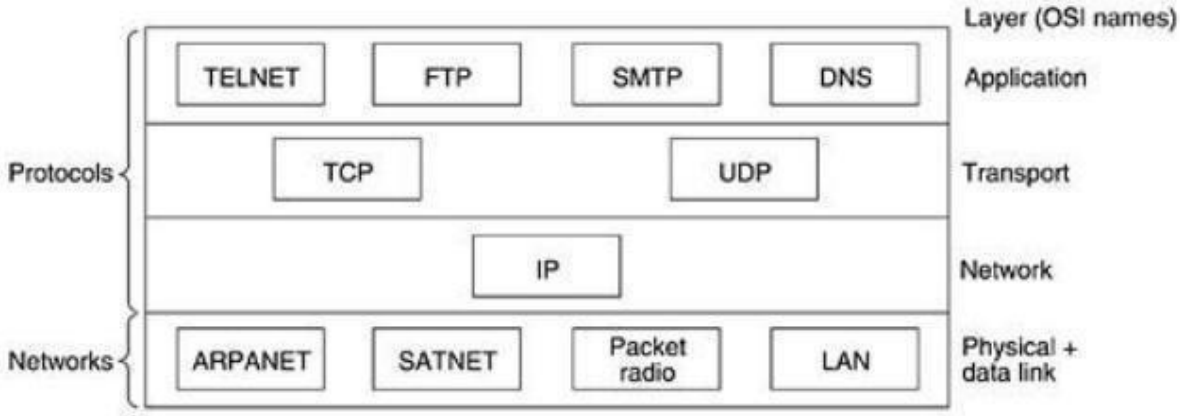


Figure 1.15 Protocols and networks in the TCP/IP model initially.

1.4.2. The Transport Layer

The layer above the internet layer in the TCP/IP model is now usually called transport layer. It is designed to allow peer entities on the source and destination hosts to carry on a

conversation, the same as in the OSI transport layer. Two end-to-end protocols have been defined here:

- *TCP* (Transmission Control Protocol) is a reliable connection-oriented protocol that allows a byte stream originating on one machine to be delivered without error on any other machine in the internet. It fragments the incoming byte stream into discrete messages and passes each one onto the internet layer. At the destination, the receiving TCP process reassembles the received messages into the output stream. TCP also handles flow control.
- *UDP* (User Datagram Protocol) is an unreliable, connectionless protocol for applications that do not want TCP's sequencing or flow control and wish to provide their own. It is also widely used for one/shot, client/server type request/reply queries and applications in which prompt delivery is more important than accurate delivery.

1.4.3. The Application Layer

The application layer is on the top of the transport layer. It contains all the higher level protocols. Some of them are:

- Virtual terminal (TELNET) - allows a user on one machine to log into a distant machine and work there.
- File transfer protocol (FTP) - provides a way to move data efficiently from one machine to another.
- Electronic mail (SMTP) - specialized protocol for electronic mail.
- Domain name service (DNS) - for mapping host names onto their network addresses.

1.4.4. The Host-to-Network Layer

Bellow the internet layer there is a great void. The TCP/IP reference model does not really say much about what happens here, except to point out that the host has to connect to the network using some protocol so it can send IP packet over it. This protocol is not defined and varies from host to host and network to network.

1.4.5. A Comparison of the OSI and TCP Reference Models

The OSI and the TCP/IP reference models have much in common:

- They are based on the concept of a stack of independent protocols,

- They have roughly similar functionality of layers,
- The layers up and including transport layer provide an end-to-end network-independent transport service to processes wishing to communicate.

The two models also have many differences (in addition to different protocols).

Probably the biggest contribution of the OSI model is that it makes the clear distinction between its three central concepts that are services, interfaces, and protocols.

Each layer performs some services for the layer above it. The service definition tells what the layer does, not how entities above it access it or how the layer works.

A layer's interface tells the processes above it how to access it including the specification of the parameters and the expected results. But it, too, says nothing about how the layer works inside.

The peer protocols used in a layer are its own business. It can use any protocol as long as it provides the offered services.

These ideas fit with modern ideas about object-oriented programming where a layer can be understood to be an object with a set of operations that processes outside the object can invoke.

The TCP/IP model did not originally clearly distinguish between service, interface, and protocol. As a consequence, the protocol in the OSI model are better hidden than in the TCP/IP model and can be replaced relatively easily as the technology changes.

The OSI reference model was devised before the protocols were invented. The positive aspect of this was that the model was made quite general, not biased toward one particular set of protocols. The negative aspect was that the designers did not have much experience with the subject and did not have a good idea of which functionality to put into which layer (e.g. some new sublayers had to be hacked into the model).

With the TCP/IP the reverse was true: the protocols came first, and the model was just a description of the existing protocols. As a consequence, the model was not useful for describing other non-TCP/IP networks.

An obvious difference between the two models is the number of layers. Another difference is in the area of connectionless versus connection-oriented communication. The OSI model

supports both types of communication in the network layer, but only connection-oriented communication in the transport layer. The TCP/IP model has only connectionless mode in the network layer but supports both modes in the transport layer. The connectionless choice is especially important for simple request-response protocols.

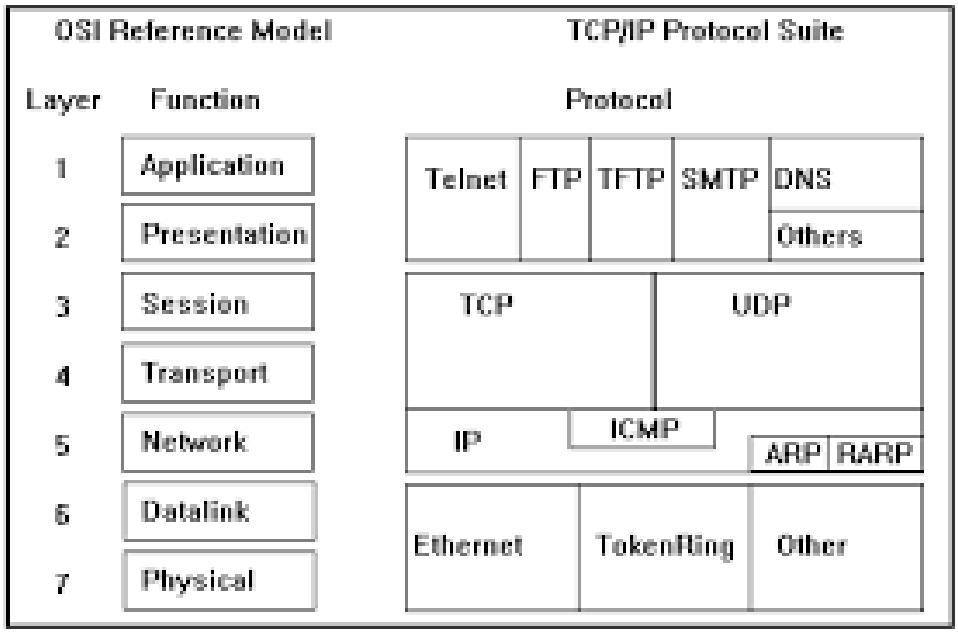


Figure 1.16 TCP/IP and OSI Model

1.5. Addressing

Four levels of addresses are used in an internet employing the *TCP/IP* protocols: physical (link) addresses, logical (IP) addresses, port addresses, and specific addresses.

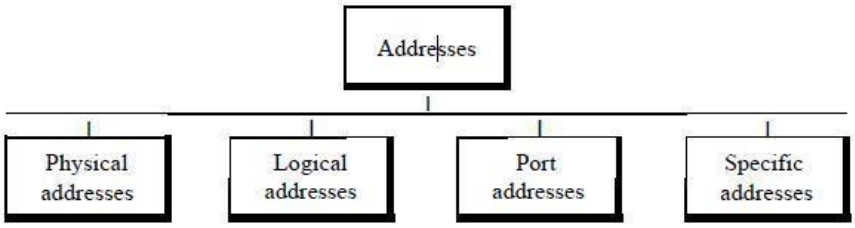


Figure 1.17

Relationship of addressing and its layers:

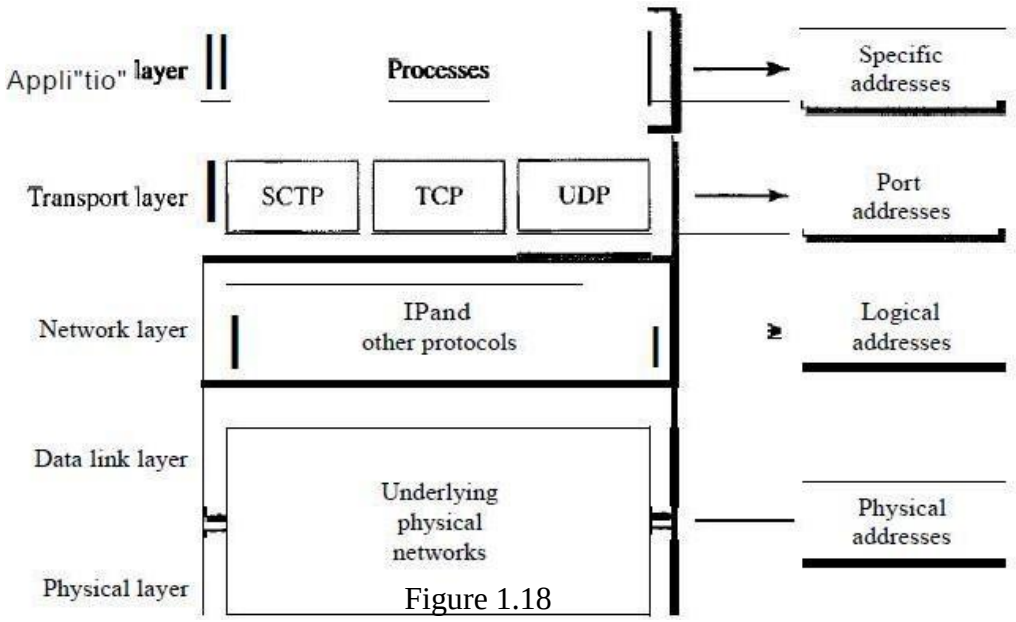


Figure 1.18

Physical Addresses

The physical address, also known as the link address, is the address of a node as defined by its LAN or WAN. It is included in the frame used by the data link layer. It is the lowest-level address.

The physical addresses have authority over the network (LAN or WAN). The size and format of these addresses vary depending on the network. For example, Ethernet uses a 6-byte (48-bit) physical address that is imprinted on the network interface card (NIC). Local Talk (Apple), however, has a I-byte dynamic address that changes each time the station comes up.

Example:

As we will see in Chapter 13, most local-area networks use a 48-bit (6-byte) physical address written as 12 hexadecimal digits; every byte (2 hexadecimal digits) is separated by a colon, as shown below:

07:01:02:01:2C:4B

A 6-byte (12 hexadecimal digits) physical address

Logical Addresses

Logical addresses are necessary for universal communications that are independent of underlying physical networks. Physical addresses are not adequate in an internetwork environment where different networks can have different address formats. A universal addressing system is needed in which each host can be identified uniquely, regardless of the underlying physical network.

The logical addresses are designed for this purpose. A logical address in the Internet is currently a 32-bit address that can uniquely define a host connected to the Internet. No two publicly addressed and visible hosts on the Internet can have the same IP address.

Port Addresses

The IP address and the physical address are necessary for a quantity of data to travel from a source to the destination host. However, arrival at the destination host is not the final objective of data communications on the Internet. A system that sends nothing but data from one computer to another is not complete. Today, computers are devices that can run multiple processes at the same time. The end objective of Internet communication is a process communicating with another process. For example, computer A can communicate with computer C by using TELNET. At the same time, computer A communicates with computer B by using the File Transfer Protocol (FTP). For these processes to receive data simultaneously, we need a method to label the different processes. In other words, they need addresses. In the TCPIP architecture, the label assigned to a process is called a port address. A port address in TCPIP is 16 bits in length.

Example: port address is a 16-bit address represented by one decimal number as shown.

753

A 16-bit port address represented as one single number

Specific Addresses

Some applications have user-friendly addresses that are designed for that specific address.

Examples include the e-mail address (for example, forouzan@fhda.edu) and the Universal Resource Locator (URL) (for example, www.mhhe.com).

1.6. Data and Signals

One of the major functions of the physical layer is to move data in the form of electromagnetic signals across a transmission medium.

Generally, the data usable to a person or application are not in a form that can be transmitted over a network. For example, a photograph must first be changed to a form that transmission media can accept. Transmission media work by conducting energy along a physical path.

1.7. Analog and Digital

Data can be analog or digital. The term **analog data** refers to information that is continuous; **digital data** refers to information that has discrete states.

Analog data, such as the sounds made by a human voice, take on continuous values. When someone speaks, an analog wave is created in the air. This can be captured by a microphone and converted to an analog signal or sampled and converted to a digital signal.

Digital data take on discrete values. For example, data are stored in computer memory in the form of Os and 1s. They can be converted to a digital signal or modulated into an analog signal for transmission across a medium.

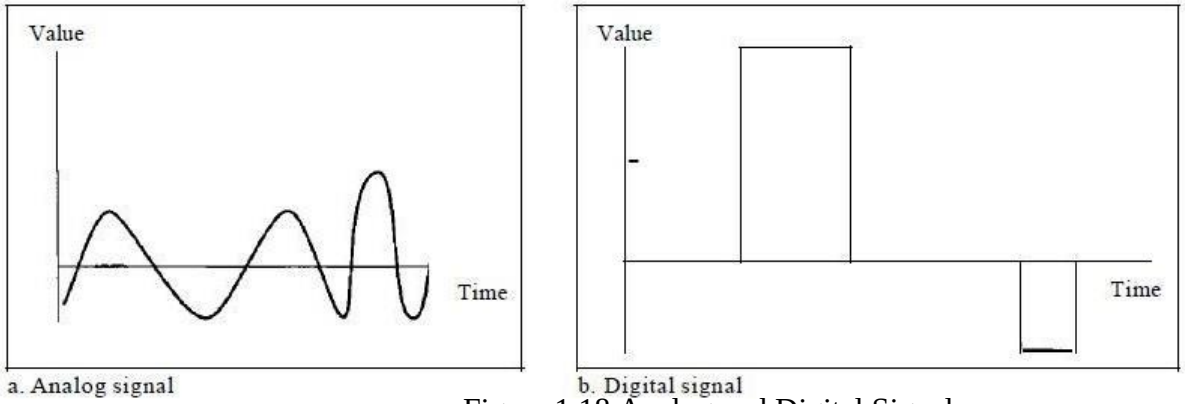


Figure 1.18 Analog and Digital Signals.

Periodic and Non-periodic Signals:

Both analog and digital signals can take one of two forms: *periodic* or *non-periodic* (sometimes refer to as *a periodic,* because the prefix *a* in Greek means "non").

A periodic signal completes a pattern within a measurable time frame, called a period, and repeats that pattern over subsequent identical periods. The completion of one full pattern is called a cycle. A non-periodic signal changes without exhibiting a pattern or cycle that repeats over time.

Both analog and digital signals can be periodic or non-periodic. In data communications, we commonly use periodic analog signals (because they need less bandwidth) and non-periodic digital signals (because they can represent variation in data).

PERIODIC ANALOG SIGNALS

A simple periodic analog signal, a sine wave, cannot be decomposed into simpler signals. A composite periodic analog signal is composed of multiple sine waves.

Sine Wave

The sine wave is the most fundamental form of a periodic analog signal. When we visualize it as a simple oscillating curve, its change over the course of a cycle is smooth and consistent, a continuous, rolling flow.

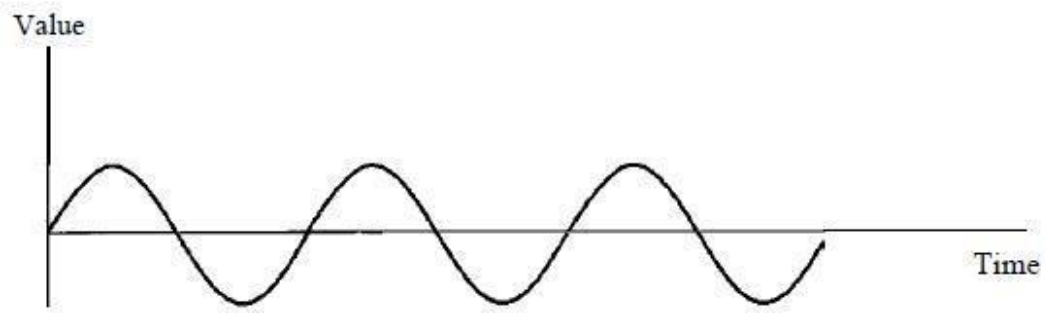


Figure 1.19 Sine wave

A sine wave can be represented by three parameters: the *peak amplitude*, the *frequency*, and the *phase.* These three parameters fully describe a sine wave.

Peak Amplitude

The peak amplitude of a signal is the absolute value of its highest intensity, proportional to the energy it carries. For electric signals, peak amplitude is normally measured in *volts.*

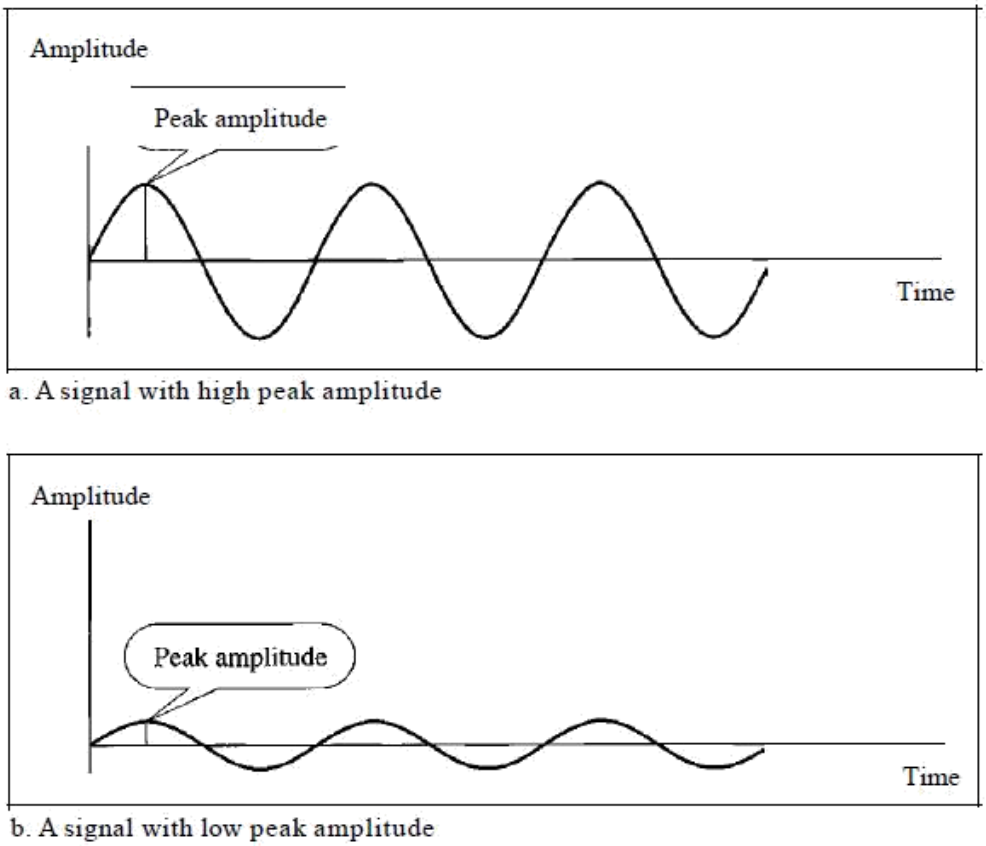


Figure 1.20 Amplitude signals

Period and Frequency

Period refers to the amount of time, in seconds, a signal needs to complete 1 cycle. Frequency refers to the number of periods in 1s. Note that period and frequency are just one characteristic defined in two ways. Period is the inverse of frequency, and frequency is the inverse of period, as the following formulas show.

$$f = \frac{1}{T} \quad \text{and} \quad T = \frac{1}{f}$$

Period is formally expressed in seconds. Frequency is formally expressed in Hertz (Hz), which is cycle per second. Units of period and frequency are shown in Table below

Unit	*Equivalent*	*Unit*	*Equivalent*
Seconds (s)	1 s	Hertz (Hz)	1 Hz
Milliseconds (ms)	10^{-3} s	Kilohertz (kHz)	10^{3} Hz
Microseconds (μs)	10-6 s	Megahertz (MHz)	10^{6} Hz
Nanoseconds (ns)	10^{-9} s	Gigahertz (GHz)	10^{9} Hz
Picoseconds (ps)	10^{-12} s	Terahertz (THz)	10^{12} Hz

Table 1.1

Wavelength

Wavelength is another characteristic of a signal traveling through a transmission medium. Wavelength binds the period or the frequency of a simple sine wave to the propagation speed of the medium.

Composite Signals

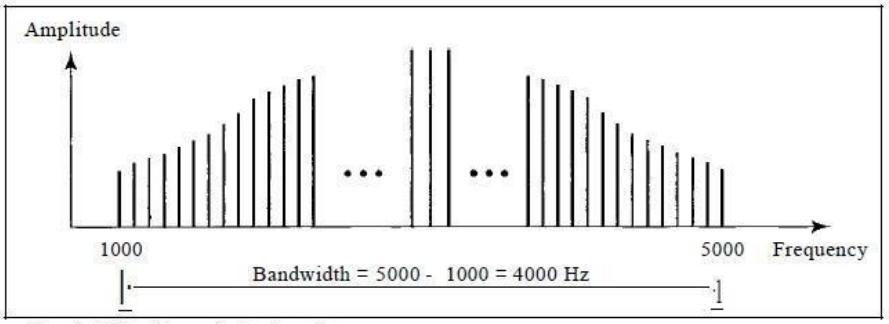
So far, we have focused on simple sine waves. Simple sine waves have many applications in daily life. We can send a single sine wave to carry electric energy from one place to another.

In a time-domain representation of this composite signal, there are an infinite number of simple sine frequencies. Although the number of frequencies in a human voice is infinite, the range is limited. A normal human being can create a continuous range of frequencies between 0 and 4 kHz.

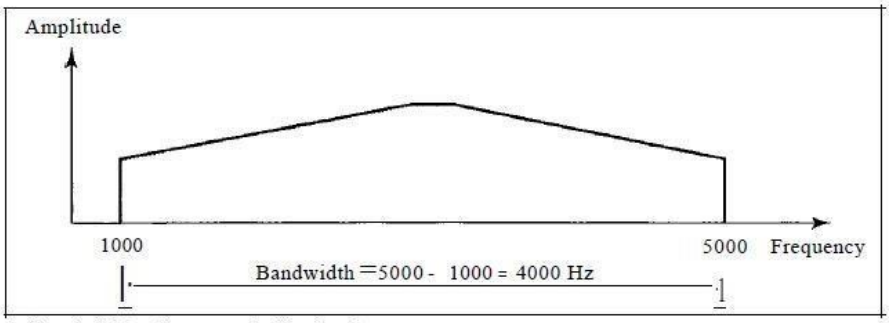
Bandwidth

The range of frequencies contained in a composite signal is its bandwidth. The bandwidth is normally a difference between two numbers. For example, if a composite signal contains frequencies between 1000 and 5000, its bandwidth is 5000-1000, or 4000.

The bandwidth of a composite signal is the difference between the highest and the lowest frequencies contained in that signal.



a. Bandwidth of a periodic signal

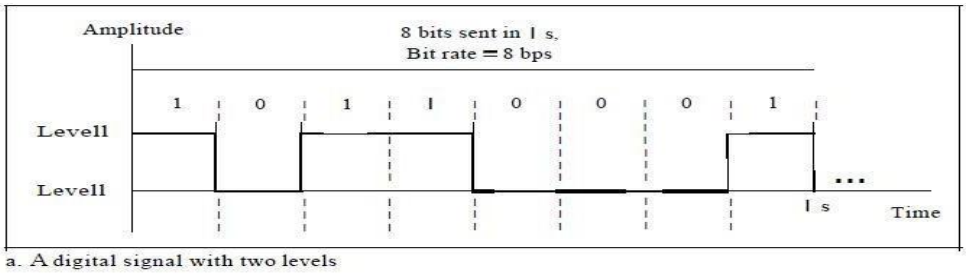


b. Bandwidth of a nonperiodic signal

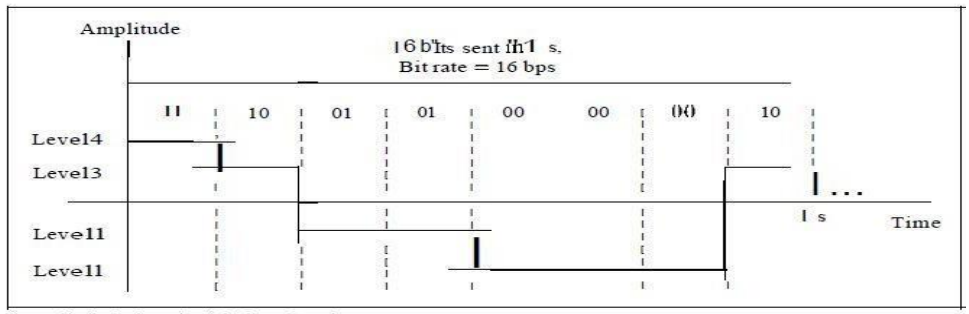
Figure 1.21 Bandwidth

DIGITAL SIGNALS:

A digital signal can have more than two levels. In this case, we can send more than 1 bit for each level.



a. A digital signal with two levels



b. A digital signal with four levels

Figure 1.22

Bit Rate

Most digital signals are non-periodic, and thus period and frequency are not appropriate characteristics. Another *term-bit rate* (instead *of frequency)-is* used to describe digital signals. The bit rate is the number of bits

sent in 1s, expressed in bits per second (bps). Figure above shows the bit rate for two signals.

Bit Length

We discussed the concept of the wavelength for an analog signal: the distance one cycle occupies on the transmission medium. We can define something similar for a digital signal: the bit length. The bit length is the distance one bit occupies on the transmission medium.

Bit length = propagation speed x bit duration

Baseband Transmission

Baseband transmission means sending a digital signal over a channel without changing the digital signal to an analog signal.

1.8. Transmission Impairment

Signals travel through transmission media, which are not perfect. The imperfection causes signal impairment. This means that the signal at the beginning of the medium is not the same as the signal at the end of the medium. What is sent is not what is received. Three causes of impairment are attenuation, distortion, and noise.

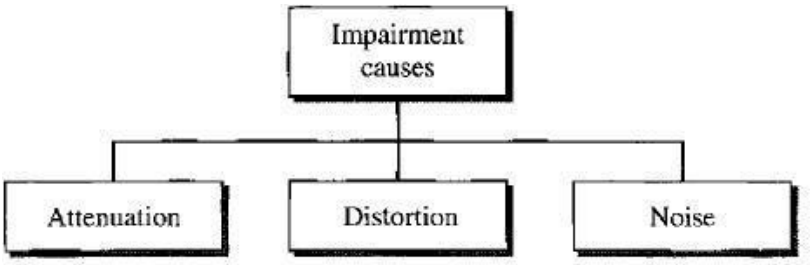


Figure 1.23

Attenuation

Attenuation means a loss of energy. When a signal, simple or composite, travels through a medium, it loses some of its energy in overcoming the resistance of the medium. That is why a wire carrying electric signals gets warm, if not hot, after a while. Some of the electrical energy in the signal is converted to heat. To compensate for this loss, amplifiers are used to amplify the signal.

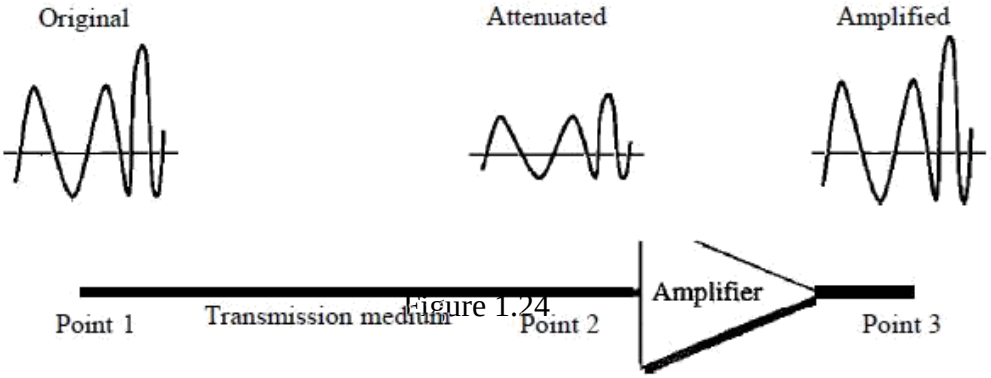


Figure 1.24

Decibel

To show that a signal has lost or gained strength, engineers use the unit of the decibel. The decibel (dB) measures the relative strengths of two signals or one signal at two different points. Note that the decibel is negative if a signal is attenuated and positive if a signal is amplified.

$$\mathbf{dB = 10 \log_{10} \frac{P_2}{P_1}}$$

Variables *PI* and *P2* are the powers of a signal at points 1 and 2, respectively.

Distortion

Distortion means that the signal changes its form or shape. Distortion can occur in a composite signal made of different frequencies. Each signal component has its own propagation speed (see the next section) through a medium and, therefore, its own delay in arriving at the final destination. Differences in delay may create a difference in phase if the delay is not exactly the same as the period duration.

The signal components at the receiver have phases different from what they had at the sender. The shape of the composite signal is therefore not the same.

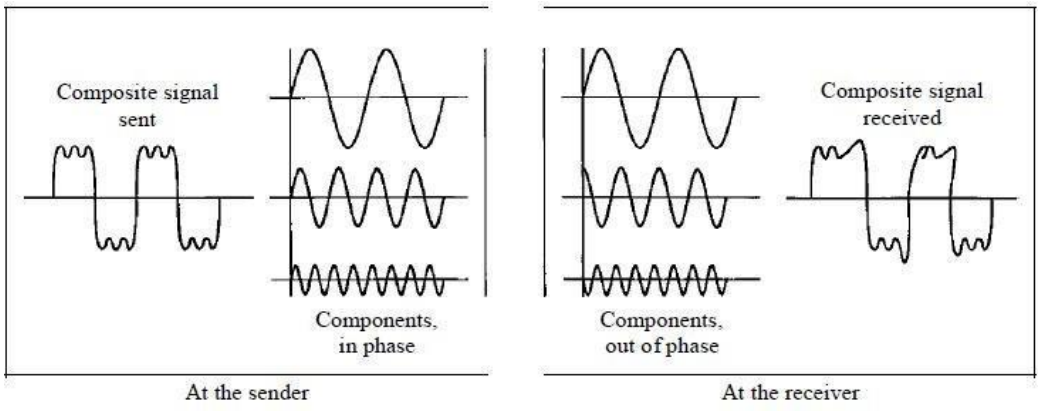


Figure 1.25

Noise

Noise is another cause of impairment. Several types of noise, such as thermal noise, induced noise, crosstalk, and impulse noise, may corrupt the signal.

Thermal noise is the random motion of electrons in a wire which creates an extra signal not originally sent by the transmitter.

Induced noise comes from sources such as motors and appliances. These devices act as a sending antenna, and the transmission medium acts as the receiving antenna.

Crosstalk is the effect of one wire on the other. One wire acts as a sending antenna and the other as the receiving antenna.

Impulse noise is a spike (a signal with high energy in a very short time) that comes from power lines, lightning, and soon.

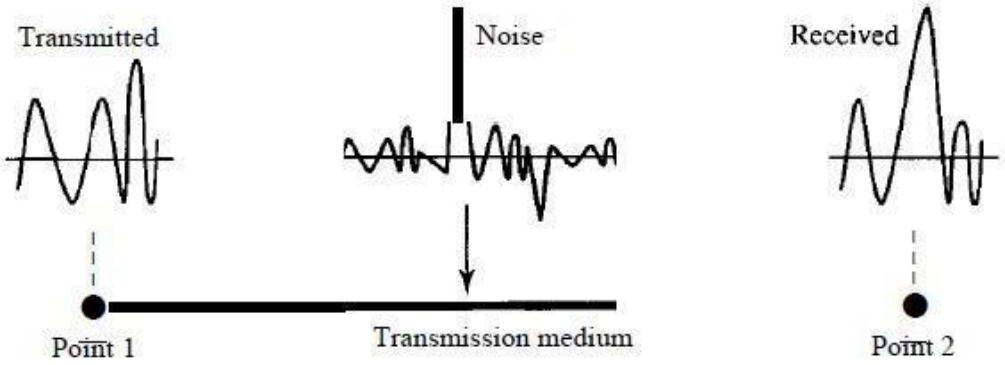


Figure 1.26

Signal-to-Noise Ratio (SNR)

As we will see later, to find the theoretical bit rate limit, we need to know the ratio of the signal power to the noise power. The signal-to-noise ratio is defined as

$$\text{SNR} = \frac{\text{average signal power}}{\text{average noise power}}$$

SNR is the ratio of two powers; it is often described in decibel units, SNRdB, defined as

$$\text{SNR}_{\text{dB}} = 10 \log_{10} \text{SNR}$$

1.9. Data rate and Channel capacity

A very important consideration in data communications is how fast we can send data, in bits per second over a channel. Data rate depends on three factors:

1. The bandwidth available
2. The level of the signals we use
3. The quality of the channel (the level of noise)

Two theoretical formulas were developed to calculate the data rate: one by Nyquist for a noiseless channels another by Shannon for a noisy channel.

Noiseless Channel: Nyquist Bit Rate

For a noiseless channel, the Nyquist bit rate formula defines the theoretical maximum bit rate

Bandwidth of the channel, *L* is the number of signal levels used to represent data, and bit rate in bits per second. Increasing the levels of a signal may reduce the reliability of the system.

Noisy Channel: Shannon Capacity

In reality, we cannot have a noiseless channel; the channel is always noisy. In 1944, Claude Shannon introduced a formula, called the Shannon capacity, to determine the theoretical highest data rate for a noisy channel:

$$\text{Capacity} = \text{bandwidth} \times \log_2 (1 + \text{SNR})$$

In this formula, bandwidth is the bandwidth of the channel, SNR is the signal-to-noise ratio, and capacity is the capacity of the channel in bits per second.

1.10. Performance

We have discussed the tools of transmitting data (signals) over a network and how the data behave. One important issue in networking is the performance of the network-how good is it? We discuss quality of service, an overall measurement of network performance,

Bandwidth

One characteristic that measures network performance is bandwidth. However, the term can be used in two different contexts with two different measuring values: bandwidth in hertz and bandwidth in bits per second.

Bandwidth in Hertz

We have discussed this concept. Bandwidth in hertz is the range of frequencies contained in a composite signal or the range of frequencies a channel can pass. For example, we can say the bandwidth of a subscriber telephone line is 4 kHz.

Bandwidth in Bits per Seconds

The term *bandwidth* can also refer to the number of bits per second that a channel, a link, or even a network can transmit. For example, one can say the bandwidth of a Fast Ethernet network (or the links in this network) is a maximum of 100 Mbps. This means that this network can send 100 Mbps.

Relationship

There is an explicit relationship between the bandwidth in hertz and bandwidth in bits per seconds. Basically, an increase in bandwidth in hertz means an increase in bandwidth in bits per second. The relationship depends on whether we have baseband transmission or transmission with modulation.

In networking, we use the term *bandwidth* in two contexts.

- The first, *bandwidth in hertz*, refers to the range of frequencies in a composite signal or the range of frequencies that a channel can pass.
- The second, *bandwidth in bits per second*, refers to the speed of bit transmission in a channel or link.

Throughput

The throughput is a measure of how fast we can actually send data through a network. Although, at first glance, bandwidth in bits per second and throughput seem the same, they are different.

Imagine a highway designed to transmit 1000 cars per minute from one point to another. However, if there is congestion on the road, this figure may be reduced to 100 cars per minute. The bandwidth is 1000 cars per minute; the throughput is 100 cars per minute.

Example:

A network with bandwidth of 10 Mbps can pass only an average of 12,000 frames per minute with each frame carrying an average of 10,000 bits. What is the throughput of this network?

Solution

We can calculate the throughput as

$$\text{Throughput} = \frac{12{,}000}{60} \times 10{,}000 = 2 \text{ Mbps}$$

The throughput is almost one-fifth of the bandwidth in this case.

Latency (Delay)

The latency or delay defines how long it takes for an entire message to completely arrive at the destination from the time the first bit is sent out from the source. We can say that latency is made of four components: propagation time, transmission time, queuing time and processing delay.

$$\text{Latency} = \text{propagation time} + \text{transmission time} + \text{queuing time} + \text{processing delay}$$

Propagation Time

Propagation time measures the time required for a bit to travel from the source to the destination. The propagation time is calculated by dividing the distance by the propagation speed.

$$\text{Propagation time} = \frac{\text{Distance}}{\text{Propagation speed}}$$

The propagation speed of electromagnetic signals depends on the medium and on the frequency of the signal For example, in a vacuum, light is propagated with a speed of 3 x 108 mfs. It is lower in air; it is much lower in cable.

Transmission Time

In data communications we don't send just 1 bit, we send a message. The first bit may take a time equal to the propagation time to reach its destination; the last bit also may take the same amount of time. However, there is a time between the first bit leaving the sender and the last bit arriving at the receiver. The first bit leaves earlier and arrives earlier; the last bit leaves later and arrives later. The time required for transmission of a message depends on the size of the message and the bandwidth of the channel.

$$\text{Transmission time} = \frac{\text{Message size}}{\text{Bandwidth}}$$

Queuing Time

The third component in latency is the queuing time, the time needed for each intermediate or end device to hold the message before it can be processed. The queuing time is not a fixed factor; it changes with the load imposed on the network. When there is heavy traffic on the network, the queuing time increases. An intermediate device, such as a router, queues, arrived messages and processes them one by one. If there are many messages, each message will have to wait.

Bandwidth-Delay Product

Bandwidth and delay are two performance metrics of a link. However, as we will see in this chapter and future chapters, what is very important in data communications is the product of the two, the bandwidth-delay product.

Jitter

Another performance issue that is related to delay is **jitter.** We can roughly say that jitter is a problem if different packets of data encounter different delays and the application using the data at the receiver site is time-sensitive (audio and video data, for example). If the delay for the first packet is 20 ms, for the second is 45 ms, and for the third is 40 ms, then the real-time application that uses the packets endures jitter.

UNIT-II PHYSICAL LAYER

2.1 DIGITAL TRANSMISSION

Physical layer is concerned with transmitting raw bits over a communication channel. The design issues have to do with making sure that when one side sends a 1 bit, it is received by the other side as 1 bit and not as 0 bit. In physical layer we deal with the communication medium used for transmission.

2.1.1 Data Encoding

Digital data to analog signals

A modem (modulator-demodulator) converts digital data to analog signal. There are 3 ways to modulate a digital signal on an analog carrier signal.

1. **Amplitude shift keying (ASK):** is a form of modulation which represents digital data as variations in the amplitude of a carrier wave. Two different amplitudes of carrier frequency represent '0', '1'.

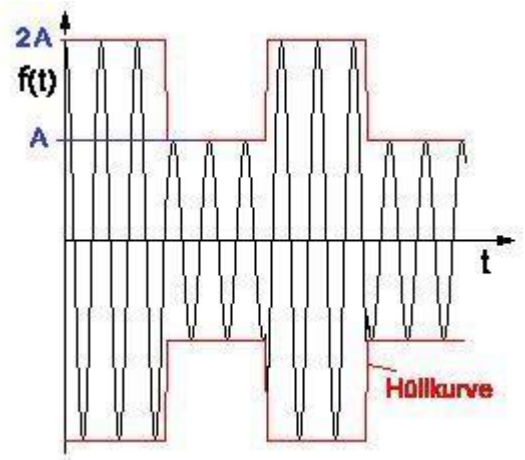


Fig 2.1 Amplitude shift keying

2. **Frequency shift keying (FSK):** In Frequency Shift Keying, the change in frequency define different digits. Two different frequencies near carrier frequency represent '0',"1'.

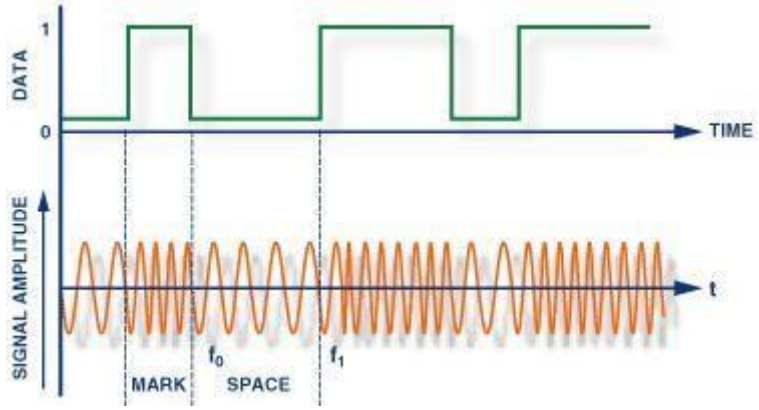


Fig 2.2 Frequency shift keying

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

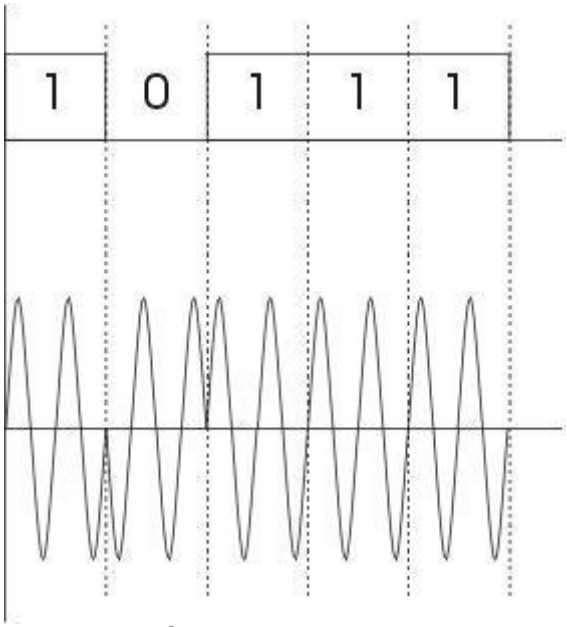


Fig 2.3Phase shift keying

2.2 Digital data to digital signals

A digital signal is sequence of discrete, discontinuous voltage pulses. Each pulses a signal element. Encoding scheme is an important factor in how successfully the receiver interprets the incoming signal.

2.2.1 Encoding Techniques

Following are several ways to map data bits to signal elements.

- **Non return to zero(NRZ)** NRZ codes share the property that voltage level is constant during a bit interval. High level voltage = bit 1 and Low level voltage = bit 0. A problem arises when there is a long sequence of 0s or 1s and the voltage level is maintained at the same value fo r a long time. This creates a problem on the receiving end because now, the clock synchronization is lost due to lack of many transitions and hence, it is difficult to determine the exact number of 0s or 1s in this sequence.

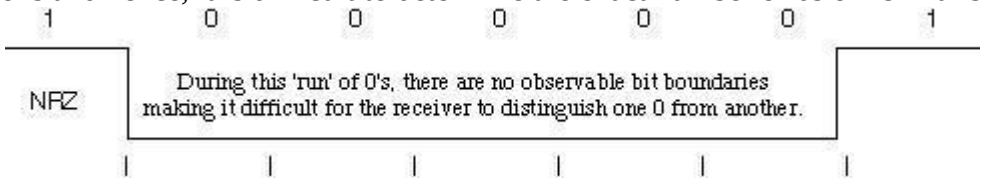


Fig 2.4 Non return to zero

The two variations are as follows:

1. **NRZ-Level:** In NRZ-L encoding, the polarity of the signal changes only when the incoming signal changes from a 1 to a 0 or from a 0 to a 1. NRZ-L method looks just like the NRZ method, except for the first input one data bit. This is because NRZ does not consider the first data bit to be a polarity change, where NRZ-L does.
2. **NRZ-Inverted:** Transition at the beginning of bit interval = bit 1 and No Transition at beginning of bit interval = bit 0 or vice versa. This technique is known as differential encoding.

NRZ-I has an advantage over NRZ-L. Consider the situation when two data wires are wrongly

connected in each other's place. In NRZ-L all bit sequences will get reversed (B'coz voltage levels get swapped).Whereas in NAZ-I since bits are recognized by transition the bits will be correctly interpreted. A disadvantage in NRZ codes is that a string of 0's or 1's will prevent synchronization of transmitter clock with receiver clock and a separate clock line need to be provided.

- **Biphase encoding:** It has following characteristics:
  6. Modulation rate twice that of NRZ and bandwidth correspondingly greater. (Modulation is the rate at which signal level is changed).
  2. Because there is predictable transition during each bit time, the receiver can synchronize on that transition i.e. clock is extracted from the signal itself.
  3. Since there can be transition at the beginning as well as in the middle of the bit interval the clock operates at twice the data transfer rate.

  Types of Encoding -->
  - **Biphase-Manchester:** Transition from high to low in middle of interval = 1 and Transition from low to high in middle of interval = 0
  - **Differential-Manchester:** Always a transition in middle of interval. No transition at beginning of interval=1 and Transition at beginning of interval = 0

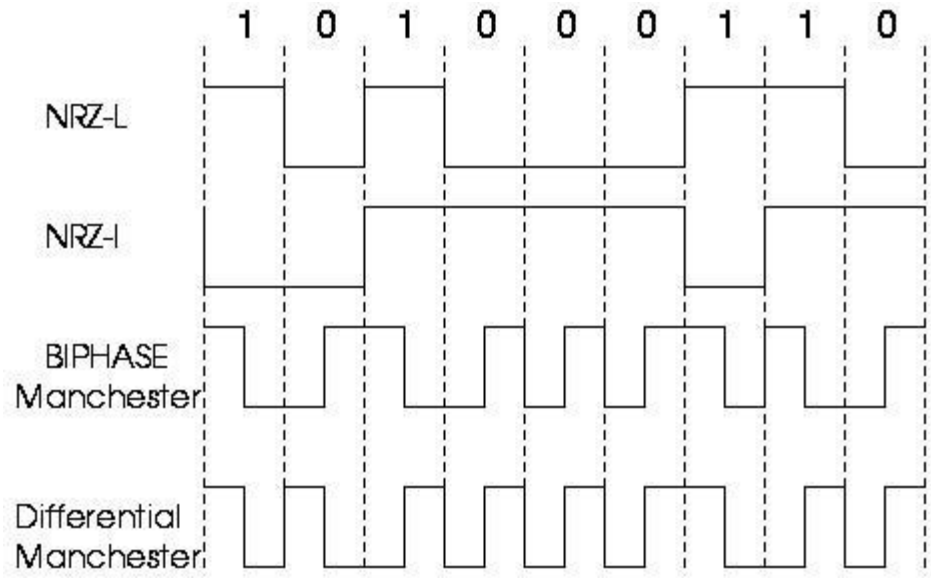


Fig 2.5 Biphase encoding

- **4B/5B Encoding:** In Manchester encoding scheme, there is a transition after every bit. It means that we must have clocks with double the speed to send same amount of data as in NRZ encodings. In other words, we may say that only 50% of the data is sent. This performance factor can be significantly improved if we use a better encoding scheme. This scheme may have a transition after fixed number of bits instead of every other bit. Like if we have a transition after every four bits, then we will be sending 80% data of actual capacity. This is a significant improvement in the performance.

  This scheme is known as **4B/5B**. So here we convert 4-bits to 5-bits, ensuring at least one transition in them. The basic idea here is that 5-bit code selected must have:
  - One leading 0
  - No more than two trailing 0s

  Thus it is ensured that we can never have more than three consecutive 0s. Now these 5-bit codes

are transmitted using NRZI coding thus problem of consecutive 1s is solved.

The exact transformation is as follows:

4-bit Data	5-bit code	4-bit Data	5-bit code
0000	11110	1000	10010
0001	01001	1001	10011
0010	10100	1010	10110
0011	10101	1011	10111
0100	01010	1100	11010
0101	01011	1101	11011
0110	01110	1110	11100
0111	01111	1111	11101

The remaining 16 codes, 7 are invalid and others are used to send some control information like line idle(11111), line dead(00000), Halt(00100) etc.

There are other variants for this scheme viz. 5B/6B, 8B/10B etc. These have self suggesting names.

- **8B/6T Encoding:** In the above schemes, we have used two/three voltage levels for a signal. But we may altogether use more than three voltage levels so that more than one-bit could be send over a single signal. Like if we use six voltage levels and we use 8-bits then the scheme is called **8B/6T**. Clearly here we have 729(3^6) combinations for signal and 256(2^8) combinations for bits.

- **Bipolar AIM:** Here we have 3 voltage levels: middle, upper, lower
  - Representation 1: Middle level =0 Upper, Lower level =1 such that successive 1's will be represented alternately on upper and lower levels.
  - Representation 2 (pseudoternary): Middle level =1 Upper, Lower level=0

2.3 Analog data to digital signal:

The process is called digitization. Sampling frequency must be at least twice that of highest frequency present in the signal so that it may be fairly regenerated. Quantization - Max and Min values of amplitude in the sample are noted. Depending on number of bits (say n) we use we divide the interval (min, max) into 2(^n) number of levels. The amplitude is then approximated to the nearest level by a 'n' bit integer. The digital signal thus consists of blocks of n bits. On reception the process is reversed to produce analog signal. But a lot of data can be lost if fewer bits are used or sampling frequency not so high.

Pulse code modulation (PCM): Here intervals are equally spaced. 8 bit PCB uses 256 different levels

- of amplitude. In non-linear encoding levels may be unequally spaced.
- **Delta Modulation (DM):** Since successive samples do not differ very much we send the differences between previous and present sample. It requires fewer bits than in PCM.

2.3.1 Digital Data Communication Techniques:

For two devices linked by a transmission medium to exchange data, a high degree of co-operation is required. Typically data is transmitted one bit at a time. The timing (rate, duration, spacing) of these bits must be same for transmitter and receiver. There are two options for transmission of bits.

1. **Parallel** All bits of a byte are transferred simultaneously on separate parallel wires. Synchronization between multiple bits is required which becomes difficult over large distance. Gives large band width but expensive. Practical only for devices close to each other.
2. **Serial** Bits transferred serially one after other. Gives less bandwidth but cheaper. Suitable for transmission over long distances.

Transmission Techniques:

1. **Asynchronous:** Small blocks of bits (generally bytes) are sent at a time without any time relation between consecutive bytes .when no transmission occurs a default state is maintained corresponding to bit 1. Due to arbitrary delay between consecutive bytes, the time occurrences of the clock pulses at the receiving end need to be synchronized for each byte. This is achieved by providing 2 extra bits start and stop.

   Start bit: It is prefixed to each byte and equals 0. Thus it ensures a transition from 1 to 0 at onset of transmission of byte. The leading edge of start bit is used as a reference for generating clock pulses at required sampling instants. Thus each onset of a byte results in resynchronization of receiver clock.

   Stop bit: To ensure that transition from 1 to 0 is always present at beginning of a byte it is necessary that default state be 1. But there may be two bytes one immediately following the other and if last bit of first byte is 0, transition from 1 to 0 will not occur. Therefore a stop bit is suffixed to each byte equaling 1. It's duration is usually 1, 1.5,2 bits.

   Asynchronous transmission is simple and cheap but requires an overhead of 3 bits i.e. for 7 bit code 2 (start ,stop bits)+1 parity bit implying 30% overhead. However % can be reduced by sending larger blocks of data but then timing errors between receiver and sender can not be tolerated beyond [50/no. of bits in block] % (assuming sampling is done at middle of bit interval). It will not only result in incorrect sampling but also misaligned bit count i.e. a data bit can be mistaken for stop bit if receiver's clock is faster.

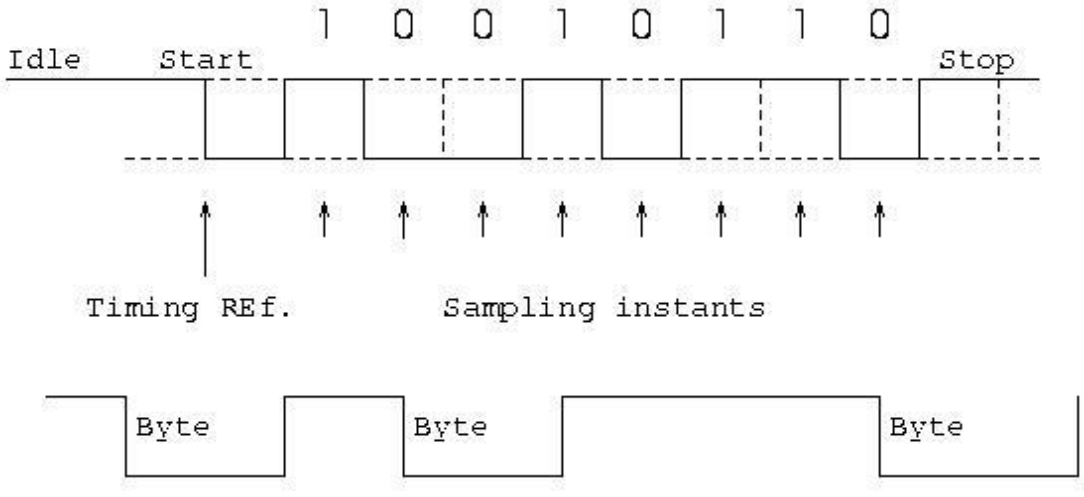


Fig 2.6 Asynchronous

4. **Synchronous** - Larger blocks of bits are successfully transmitted. Blocks of data are either treated as sequence of bits or bytes. To prevent timing drift clocks at two ends need to be synchronized. This can done in two ways:

   Provide a separate clock line between receiver and transmitter. OR

   Clocking information is embedded in data signal i.e. biphase coding for digital signals.

   Still another level of synchronization is required so that receiver determines beginning or end of block of data. Hence each block begins with a start code and ends with a stop code. These are in general same known as flag that is unique sequence of fixed no. of bits. In addition some control characters encompass data within these flags. **Data+ control information** is called a frame. Since any arbitrary bit pattern can be transmitted there is no assurance that bit pattern for flag will not appear inside the frame thus destroying frame level synchronization. So to avoid this we use bit stuffing

   Bit Stuffing: Suppose our flag bits are 01111110 (six 1's). So the transmitter will always insert an extra 0 bit after each occurrence of five 1's (except for flags). After detecting a starting flag, the receiver monitors the bit stream. If pattern of five 1's appear, the sixth is examined and if it is 0 it is deleted else if it is 1 and next is 0 the combination is accepted as a flag. Similarly byte stuffing is used for byte oriented transmission. Here we use an escape sequence to prefix a byte similar to flag and 2 escape sequences if byte is itself a escape sequence.

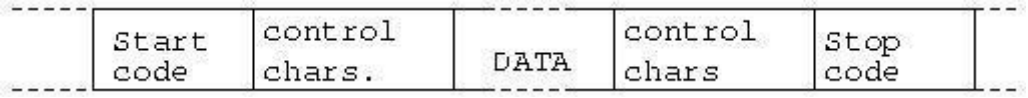


Fig 2.7 Synchronous

2.4 IEEE 802.5: Token Ring Network

- Token Ring is formed by the nodes connected in ring format as shown in the diagram below. The principle used in the token ring network is that a token is circulating in the ring and whichever node grabs that token will have right to transmit the data.

- Whenever a station wants to transmit a frame it inverts a single bit of the 3-byte token which instantaneously changes it into a normal data packet. Because there is only one token, there can at most be one transmission at a time.

- Since the token rotates in the ring it is guaranteed that every node gets the token with in some specified time. So there is an upper bound on the time of waiting to grab the token so that starvation is avoided. There is also an upper limit of 250 on the number of nodes in the network.

To distinguish the normal data packets from token (control packet) a special sequence is assigned to the token packet. When any node gets the token it first sends the data it wants to send, then recalculates the token.

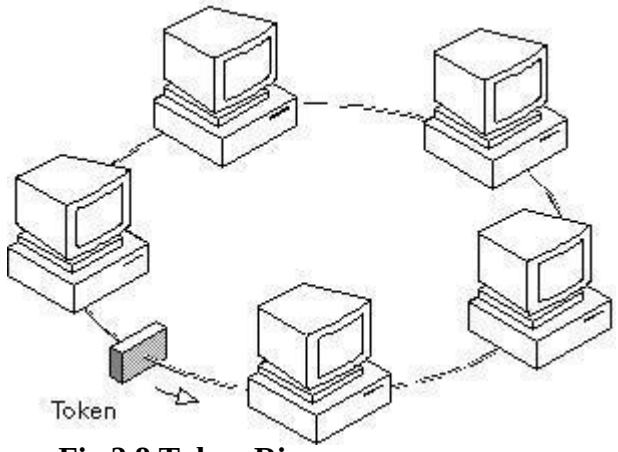


Fig 2.8 Token Ring

If a node transmits the token and nobody wants to send the data the token comes back to the sender. If the first bit of the token reaches the sender before the transmission of the last bit, then error situation arises. So to avoid this we should have:

Propagation delay + transmission of n-bits (1-bit delay in each node) > transmission of the token time

A station may hold the token for the token-holding time, which is 10 ms unless the installation sets a different value. If there is enough time left after the first frame has been transmitted to send more frames, then these frames may be sent as well. After all pending frames have been transmitted or the transmission frame would exceed the token-holding time, the station regenerates the 3-byte token frame and puts it back on the ring.

2.4.1 Modes of Operation

Listen Mode: In this mode the node listens to the data and transmits the data to the next node. In this mode there is a one-bit delay associated with the transmission.

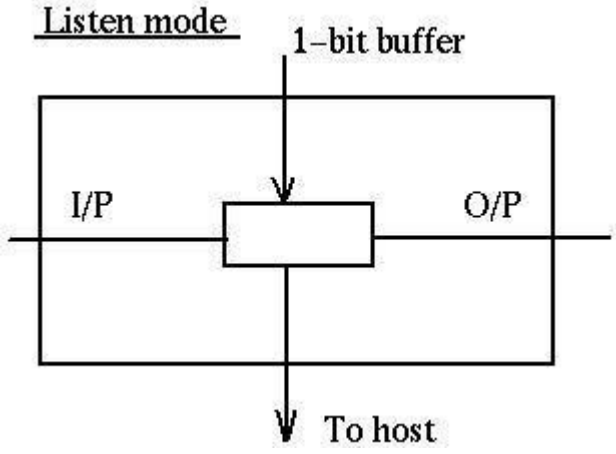


Fig 2.9 Listen mode

2. **Transmit Mode:** In this mode the node just discards the any data and puts the data onto the network.

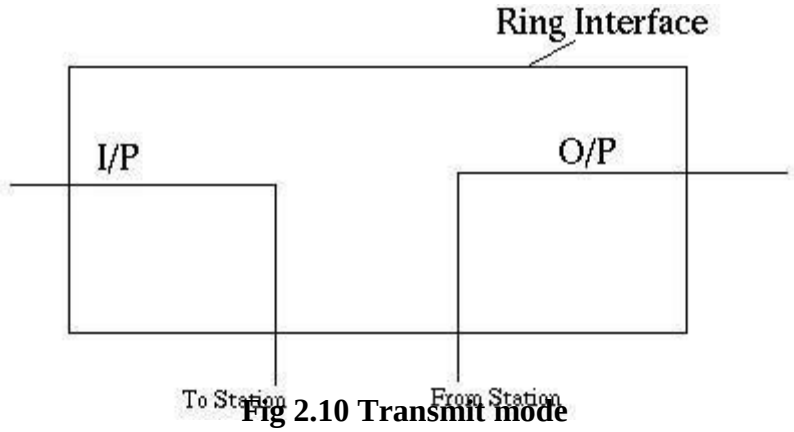


Fig 2.10 Transmit mode

1. **By-pass Mode:** In this mode reached when the node is down. Any data is just bypassed. There is no one-bit delay in this mode.

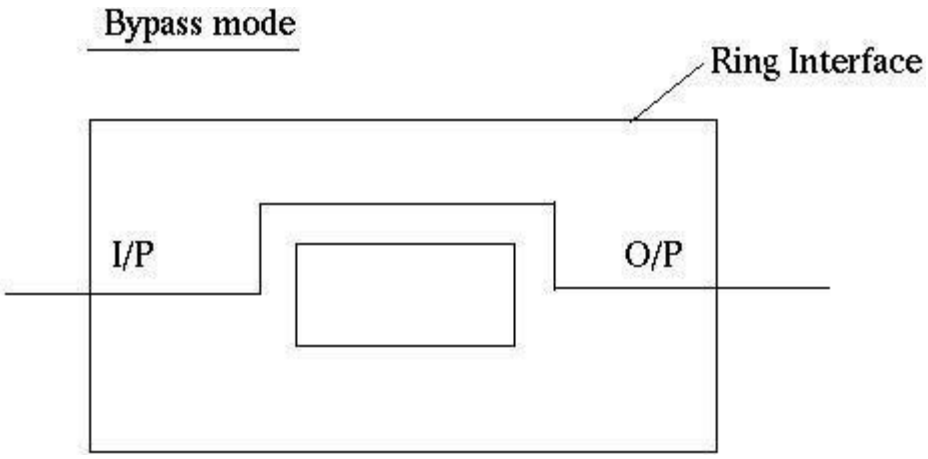


Fig 2.11 Bypass Mode

2.4.2 Token Ring Using Ring Concentrator

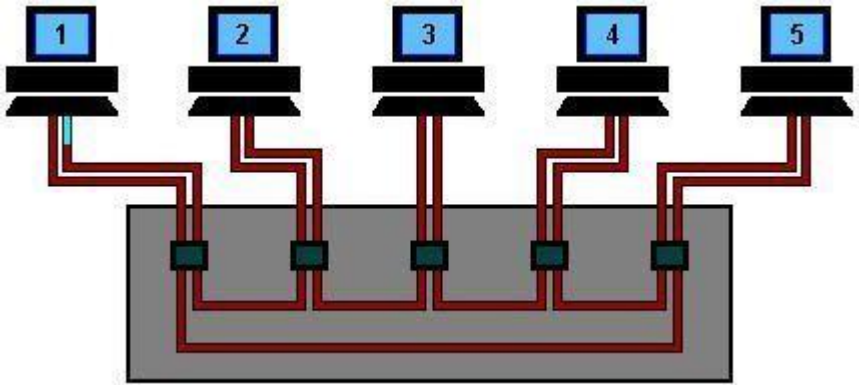


Fig 2.12 Token Ring Using Ring Concentrator

One problem with a ring network is that if the cable breaks somewhere, the ring dies. This problem is elegantly addressed by using a ring concentrator. A Token Ring concentrator simply changes the topology from a physical ring to a star wired ring. But the network still remains a ring logically. Physically, each station is connected to the ring concentrator (wire center) by a cable containing at least two twisted pairs, one for data to the station and one for data from the station. The Token still circulates around the network

and is still controlled in the same manner, however, using a hub or a switch greatly improves reliability because the hub can automatically bypass any ports that are disconnected or have a cabling fault. This is done by having bypass relays inside the concentrator that are energized by current from the stations. If the ring breaks or station goes down, loss of the drive current will release the relay and bypass the station. The ring can then continue operation with the bad segment bypassed.

Who should remove the packet from the ring?

There are 3 possibilities-

> **The source itself removes the packet after one full round in the ring**.
>
> **The destination removes it after accepting it**: This has two potential problems. Firstly, the solution won't work for broadcast or multicast, and secondly, there would be no way to acknowledge the sender about the receipt of the packet.
>
> **Have a specialized node only to discard packets**: This is a bad solution as the specialized node would know that the packet has been received by the destination only when it receives the packet the second time and by that time the packet may have actually made about one and half (or almost two in the worst case) rounds in the ring.

Thus the first solution is adopted with the source itself removing the packet from the ring after a full one round. With this scheme, broadcasting and multicasting can be handled as well as the destination can acknowledge the source about the receipt of the packet (or can tell the source about some error).

2.4.3 Token Format

The token is the shortest frame transmitted (24 bit)

MSB (Most Significant Bit) is always transmitted first - as opposed to Ethernet

SD = Starting Delimiter (1 Octet)

AC = Access Control (1 Octet)

ED = Ending Delimiter (1 Octet)

Starting Delimiter Format:

J = Code Violation

K = Code Violation

Access Control Format:

T=Token

T = 0 for Token

T = 1 for Frame

When a station with a Frame to transmit detects a token which has a priority equal to or less than the Frame to be transmitted, it may change the token to a start-of-frame sequence and transmit the Frame

P = Priority

Priority Bits indicate tokens priority, and therefore, which stations are allowed to use it. Station can transmit if its priority as at least as high as that of the token.

M = Monitor

The monitor bit is used to prevent a token whose priority is greater than 0 or any frame from continuously circulating on the ring. If an active monitor detects a frame or a high priority token with the monitor bit equal to 1, the frame or token is aborted. This bit shall be transmitted as 0 in all frame and tokens. The active monitor inspects and modifies this bit. All other stations shall repeat this bit as received.

R = Reserved bits

The reserved bits allow station with high priority Frames to request that the next token be issued at the requested priority.

Ending Delimiter Format:

J = Code Violation

K = Code Violation

I = Intermediate Frame Bit

E = Error Detected Bit

2.4.4 Frame Format:

MSB (Most Significant Bit) is always transmitted first - as opposed to Ethernet

SD=Starting Delimiter(1 octet)

AC=Access Control(1 octet)

FC = Frame Control (1 Octet)

DA = Destination Address (2 or 6 Octets)

SA = Source Address (2 or 6 Octets)

DATA = Information 0 or more octets up to 4027

CRC = Checksum(4 Octets)

ED = Ending Delimiter (1 Octet)

FS=Frame Status

Starting Delimiter Format:

J = Code Violation

K = Code Violation

Access Control Format:

T=Token

T = “0” for Token,

T = “1” for Frame.

When a station with a Frame to transmit detects a token which has a priority equal to or less than the Frame to be transmitted, it may change the token to a start-of-frame sequence and transmit the Frame.

P = Priority

Bits Priority Bits indicate tokens priority, and therefore, which stations are allowed to use it.

Station can transmit if its priority as at least as high as that of the token.

M = Monitor

The monitor bit is used to prevent a token whose priority is greater than 0 or any frame from continuously circulating on the ring. if an active monitor detects a frame or a high priority token with the monitor bit equal to 1, the frame or token is aborted. This bit shall be transmitted

as 0 in all frame and tokens. The active monitor inspects and modifies this bit. All other stations shall repeat this bit as received.

R = Reserved bits the reserved bits allow station with high priority Frames to request that the next token be issued at the requested priority

Frame Control Format:

FF= Type of Packet-Regular data packet or MAC layer packet

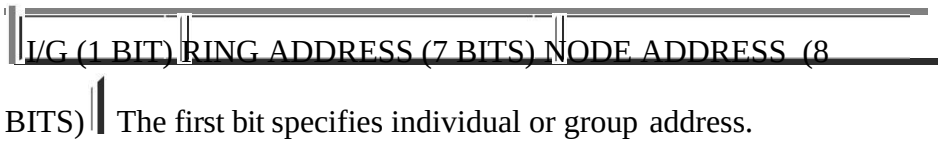
Control Bits= Used if the packet is for MAC layer protocol itself

Source and Destination Address Format:

The addresses can be of 2 bytes (local address) or 6 bytes (global address).

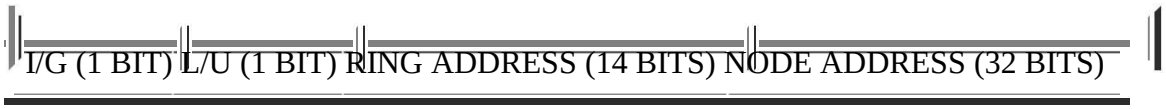
Local address format:

Alternatively



The first bit specifies individual or group address.

universal (global) address format:



The first bit specifies individual or group address.

The second bit specifies local or global (universal) address.

Local group addresses (16 bits):

The first bit specifies an individual or group address.

The second bit specifies traditional or bit signature group address.

Traditional Group Address: 2Exp14 groups can be defined.

Bit Signature Group Address: 14 grtoups are defined. A host can be a member of none or any number of them. For multicasting, those group bits are set to which the packet should go. For broadcasting, all 14 bits are set. A host receives a packet only if it is a member of a group whose corresponding bit is set to 1.

Universal group addresses (16 bits):

The description is similar to as above.

Data Format:

No upper limit on amount of data as such, but it is limited by the token holding time.

Checksum:

The source computes and sets this value. Destination too calculates this value. If the two are different, it indicates an error, otherwise the data may be correct.

Frame Status:

It contains the A and C bits.

A bit set to 1: destination recognized the packet.

C bit set to 1: destination accepted the packet.

This arrangement provides an automatic acknowledgement for each frame. The A and C bits are present twice in the Frame Status to increase reliability in as much as they are not covered by the checksum.

Ending Delimiter Format:

J = Code Violation

K = Code Violation

I = Intermediate Frame Bit

If this bit is set to 1, it indicates that this packet is an intermediate part of a bigger packet, the last packet would have this bit set to 0.

E = Error Detected Bit

This bit is set if any interface detects an error.

2.5 CONNECTING DEVICES

Hosts and networks do not normally operate in isolation. We use connecting devices to connect hosts together to make a network or to connect networks together to make an internet. Connecting devices can operate in different layers of the Internet model. We discuss three kinds of connecting devices: hubs, link-layer switches, and routers. Hubs today operate in the first layer of the Internet model. Link-layer switches operate in the first two layers. Routers operate in the first three layers.

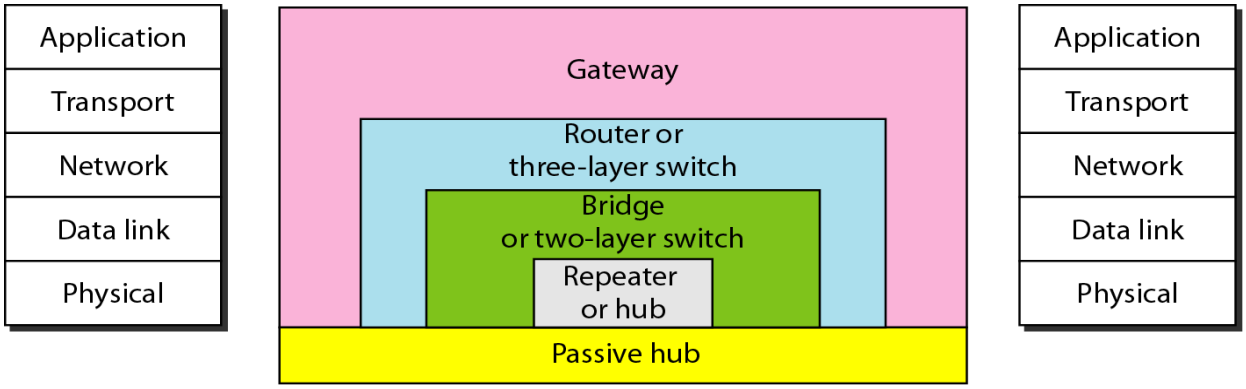


Fig: 2.13 Three categories of connecting devices

2.5.1 Hubs

A hub is a device that operates only in the physical layer. Signals that carry information within a network can travel a fixed distance before attenuation endangers the integrity of the data. A repeater receives a signal and, before it becomes too weak or corrupted, regenerates and retimes the original bit pattern. The repeater then sends the refreshed signal. In the past, when Ethernet LANs were using bus topology, a repeater was used to connect two segments of a LAN to overcome the length restriction of the coaxial cable. Today, however, Ethernet LANs use star topology. In a star topology, a repeater is a multiport device, often called a hub, which can be used to serve as the connecting point and at the same time function as a repeater. It shows that when a packet from station A to station B arrives at the hub, the signal representing the frame is regenerated to remove any possible corrupting noise, but the hub forwards the packet from all outgoing ports except the one from which the signal was received. In other words, the frame is broadcast. All stations in the LAN receive the frame, but only station B keeps it. The rest of the stations discard it. It shows the role of a repeater or a hub in a switched LAN. The figure definitely shows that a hub does not have a filtering capability; it does not have the intelligence to find from which port the frame should be sent out.

A hub or a repeater is a physical-layer device. They do not have a link-layer address and they do not check the link-layer address of the received frame. They just regenerate the corrupted bits and send them out from every port.

2.5.2 Link-Layer Switches:

A link-layer switch (or switch) operates in both the physical and the data-link layers. As a physical-layer device, it regenerates the signal it receives. As a link-layer device, the link-layer switch can check the MAC addresses (source and destination) contained in the frame.

2.5.3 Filtering

One may ask what the difference in functionality is between a link-layer switch and a hub. A link-layer switch has filtering capability. It can check the destination address of a frame and can decide from which outgoing port the frame should be sent.

Let us give an example. We have a LAN with four stations that are connected to a link-layer switch. If a frame destined for station 71:2B:13:45:61:42 arrives at port 1, the link-layer switch consults its table to find the departing port. According to its table, frames for 71:2B:13:45:61:42 should be sent out only through port 2; therefore, there is no need for forwarding the frame through other ports.

2.5.4 Transparent Switches

A transparent switch is a switch in which the stations are completely unaware of the switch's existence. If a switch is added or deleted from the system, reconfiguration of the stations is unnecessary. According to the IEEE 802.1d specification, a system equipped with transparent switches must meet three criteria:

- Frames must be forwarded from one station to another.
- The forwarding table is automatically made by learning frame movements in the network.
- Loops in the system must be prevented.

Forwarding

A transparent switch must correctly forward the frames, as discussed in the previous section. ***Learning***

The earliest switches had switching tables that were static. The system administrator would manually enter each table entry during switch setup. Although the process was simple, it was not practical. If a station was added or deleted, the table had to be modified manually. The same was true if a station's MAC address changed, which is not a rare event. For example, putting in a new network card means a new MAC address. A better solution to the static table is a dynamic table that maps addresses to ports (interfaces) automatically. To make a table dynamic, we need a switch that gradually learns from the frames' movements. To do this, the switch inspects both the destination and the source addresses in each frame that passes through the switch. The destination address is used for the forwarding decision (table lookup); the source address is used for adding entries to the table and for updating purposes. Let us elaborate on this process using.

1. When station A sends a frame to station D, the switch does not have an entry for either D or A. The frame goes out from all three ports; the frame floods the network. However, by looking at the source address, the switch learns that station A must be connected to port 1. This means that frames destined for A, in the future, must be sent out through port 1. The switch adds this entry to its table. The table has its first entry now.

2. When station D sends a frame to station B, the switch has no entry for B, so it floods the network again. However, it adds one more entry to the table related to station D.

3. The learning process continues until the table has information about every port. However, note that the learning process may take a long time. For example, if a station does not send out a frame (a rare situation), the station will never have an entry in the table.

2.5.5 Loop Problem

Transparent switches work fine as long as there are no redundant switches in the system. Systems administrators, however, like to have redundant switches (more than one switch between a pair of LANs) to make the system more reliable. If a switch fails, another switch takes over until the failed one is repaired or replaced. Redundancy can create loops in the system, which is very undesirable. Loops can be created only

when two or more broadcasting LANs (those using hubs, for example) are connected by more than one switch. Ex: A very simple example of a loop created in a system with two LANs connected by two switches.

1. Station A sends a frame to station D. The tables of both switches are empty. Both forward the frame and update their tables based on the source address A.

2. Now there are two copies of the frame on LAN 2. The copy sent out by the left switch is received by the right switch, which does not have any information about the destination address D; it forwards the frame. The copy sent out by the right switch is received by the left switch and is sent out for lack of information about D. Note that each frame is handled separately because switches, as two nodes on a broadcast network sharing the medium, use an access method such as CSMA/CD. The tables of both switches are updated, but still there is no information for destination D.

3. Now there are two copies of the frame on LAN 1. Step 2 is repeated, and both copies are sent to LAN2.

4. The process continues on and on. Note that switches are also repeaters and regenerate frames. So in each iteration, there are newly generated fresh copies of the frames.

2.5.6 Spanning Tree Algorithm

To solve the looping problem, the IEEE specification requires that switches use the spanning tree algorithm to create a loop-less topology. In graph theory, a spanning tree is a graph in which there is no loop. In a switched LAN, this means creating a topology in which each LAN can be reached from any other LAN through one path only (no loop). We cannot change the physical topology of the system because of physical connections between cables and switches, but we can create a logical topology that overlays the physical one. A system with four LANs and five switches. We have shown the physical system and its representation in graph theory. Although some textbooks represent the LANs as nodes and the switches as the connecting arcs, we have shown both LANs and switches as nodes. The connecting arcs show the connection of a LAN to a switch and vice versa. To find the spanning tree, we need to assign a cost (metric) to each arc. The interpretation of the cost is left up to the systems administrator. We have chosen the minimum hops. However, as we will see in Chapter 20, the hop count is normally 1 from a switch to the LAN and 0 in the reverse direction. The process for finding the spanning tree involves three steps:

1. Every switch has a built-in ID (normally the serial number, which is unique). Each switch broadcasts this ID so that all switches know which one has the smallest ID. The switch with the smallest ID is selected as the root switch (root of the tree). We assume that switch S1 has the smallest ID. It is, therefore, selected as the root switch.

2. The algorithm tries to find the shortest path (a path with the shortest cost) from the root switch to every other switch or LAN. The shortest path can be found by examining the total cost from the root switch to the destination.

3. The combination of the shortest paths creates the shortest tree. Based on the spanning tree, we mark the ports that are part of it, the forwarding ports, which forward a frame that the switch receives. We also mark those ports that are not part of the spanning tree, the blocking ports, which block the frames received by the switch. Figure 17.8 shows the logical systems of LANs with forwarding ports (solid lines) and blocking ports (broken lines).

Note that there is only one path from any LAN to any other LAN in the spanning tree system. This means there is only one path from one LAN to any other LAN. No loops are created. You can prove to yourself that there is only one path from LAN 1 to LAN 2, LAN 3, or LAN 4. Similarly, there is only one path from LAN 2 to LAN 1, LAN 3, and LAN 4. The same is true for LAN 3 and LAN 4. We have described the spanning tree algorithm as though it required manual entries. This is not true. Each switch is equipped with a software package that carries out this process dynamically.

Advantages of Switches

A link-layer switch has several advantages over a hub. We discuss only two of them here. *Collision Elimination*: A link-layer switch eliminates the collision. This means increasing the average bandwidth available to a host in the network. In a switched LAN, there is no need for carrier sensing and collision detection; each host can transmit at any time.

Connecting Heterogeneous Devices: A link-layer switch can connect devices that use different protocols at the physical layer (data rates) and different transmission media. As long as the format of the frame at the data-link layer does not change, a switch can receive a frame from a device that uses twisted-pair cable and sends data at 10 Mbps and deliver the frame to another device that uses fiber-optic cable and can receive data at 100 Mbps.

Routers: In this section, we mention routers to compare them with a two-layer switch and a hub. A router is a three-layer device; it operates in the physical, data-link, and network layers. As a physical-layer device, it regenerates the signal it receives. As a link-layer device, the router checks the physical addresses (source and destination) contained in the packet. As a network-layer device, a router checks the network-layer addresses.

A router can connect networks. In other words, a router is an internetworking device; it connects independent networks to form an internetwork. According to this definition, two networks connected by a router become an internetwork or an internet. There are three major differences between a router and a repeater or a switch.

1. A router has a physical and logical (IP) address for each of its interfaces.

2. A router acts only on those packets in which the link-layer destination address matches the address of the interface at which the packet arrives.

3. A router changes the link-layer address of the packet (both source and destination) when it forwards the packet.

Let us give an example. Assume an organization has two separate buildings with a Gigabit Ethernet LAN installed in each building. The organization uses switches in each LAN. The two LANs can be connected to form a larger LAN using 10 Gigabit Ethernet technologies that speeds up the connection to the Ethernet and the connection to the organization server. A router then can connect the whole system to the Internet.

2.6 Transmission Media

In data communication terminology, a transmission medium is a physical path between the transmitter and the receiver i.e it is the channel through which data is sent from one place to another. Transmission Media is broadly classified into the following types:

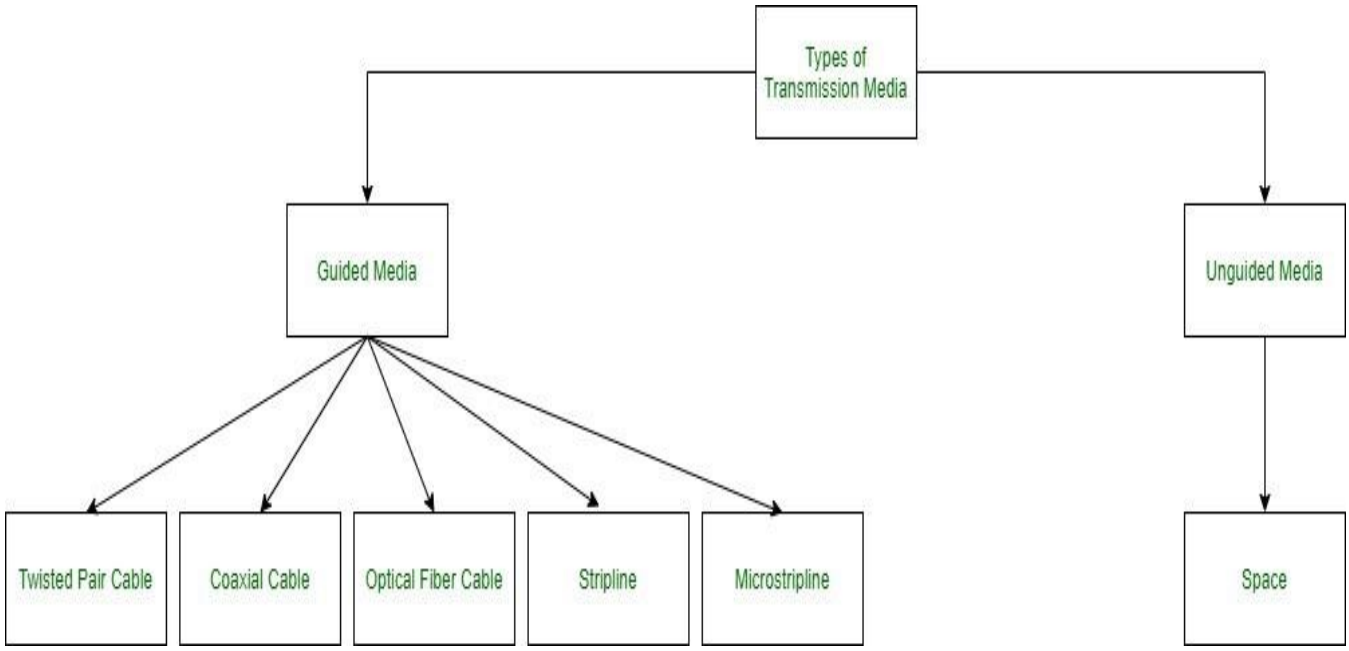


Fig 2.14 Transmission Media

2.6.1 GuidedMedia:

It is also referred to as Wired or Bounded transmission media. Signals being transmitted are directed and confined in a narrow pathway by using physical links.
Features:

- High Speed
- Secure
- Used for comparatively shorter distances

There are 3 major types of Guided Media:

(i) TwistedPairCable

It consists of 2 separately insulated conductor wires wound about each other. Generally, several such pairs are bundled together in a protective sheath. They are the most widely used Transmission Media. Twisted Pair is of two types:

1. **UnshieldedTwistedPair(UTP):**
   This type of cable has the ability to block interference and does not depend on a physical shield for this purpose. It is used for telephonic applications.
   Advantages:
   - Least expensive
   - Easy to install
   - High-speed capacity
   - Susceptible to external interference
   - Lower capacity and performance in comparison to STP
   - Short distance transmission due to attenuation
2. **ShieldedTwistedPair(STP):**
   This type of cable consists of a special jacket to block external interference. It is used in fast-data-rate Ethernet and in voice and data channels of telephone lines.
   Advantages:
   - Better performance at a higher data rate in comparison to UTP
   - Eliminates crosstalk

- Comparatively faster
- Comparatively difficult to install and manufacture
- More expensive
- Bulky

(ii) CoaxialCable

It has an outer plastic covering containing 2 parallel conductors each having a separate insulated protection cover. The coaxial cable transmits information in two modes: Baseband mode(dedicated cable bandwidth) and Broadband mode(cable bandwidth is split into separate ranges). Cable TVs and analog television networks widely use Coaxial cables.

Advantages:

- High Bandwidth
- Better noise Immunity
- Easy to install and expand
- Inexpensive

Disadvantages:

- Single cable failure can disrupt the entire network

(iii) OpticalFibreCable

It uses the concept of reflection of light through a core made up of glass or plastic. The core is surrounded by a less dense glass or plastic covering called the cladding. It is used for the transmission of large volumes of data. The cable can be unidirectional or bidirectional. The WDM (Wavelength Division Multiplexer) supports two modes, namely unidirectional and bidirectional mode.

Advantages:

- Increased capacity and bandwidth
- Lightweight
- Less signal attenuation
- Immunity to electromagnetic interference
- Resistance to corrosive materials

Disadvantages:

- Difficult to install and maintain
- High cost
- Fragile

(iv) Stripline

Stripline is a transverse electromagnetic (TEM) transmission line medium invented by Robert M. Barrett of the Air Force Cambridge Research Centre in the 1950s. Stripline is the earliest form of the planar transmission line. It uses a conducting material to transmit high-frequency waves it is also called a waveguide. This conducting material is sandwiched between two layers of the ground plane which are usually shorted to provide EMI immunity.

(v) Microstripline

In this, the conducting material is separated from the ground plane by a layer of dielectric.

2.6.2. UnguidedMedia:

It is also referred to as Wireless or Unbounded transmission media.No physical medium is required for the transmission of electromagnetic signals.

Features:

- The signal is broadcasted through air
- Less Secure

- Used for larger distances

There are 3 types of Signals transmitted through unguided media:

(i) Radiowaves

These are easy to generate and can penetrate through buildings. The sending and receiving antennas need not be aligned. Frequency Range:3KHz – 1GHz. AM and FM radios and cordless phones use Radiowaves for transmission.

Further Categorized as (i) Terrestrial and (ii) Satellite.

(ii) Microwaves

It is a line of sight transmission i.e. the sending and receiving antennas need to be properly aligned with each other. The distance covered by the signal is directly proportional to the height of the antenna. Frequency Range:1GHz – 300GHz. These are majorly used for mobile phone communication and television distribution.

(iii)Infrared

Infrared waves are used for very short distance communication. They cannot penetrate through obstacles. This prevents interference between systems. Frequency Range:300GHz – 400THz. It is used in TV remotes, wireless mouse, keyboard, printer, etc.

2.7 Switching techniques

In large networks, there can be multiple paths from sender to receiver. The switching technique will decide the best route for data transmission.

Switching technique is used to connect the systems for making one-to-one communication.

Classification Of Switching Techniques

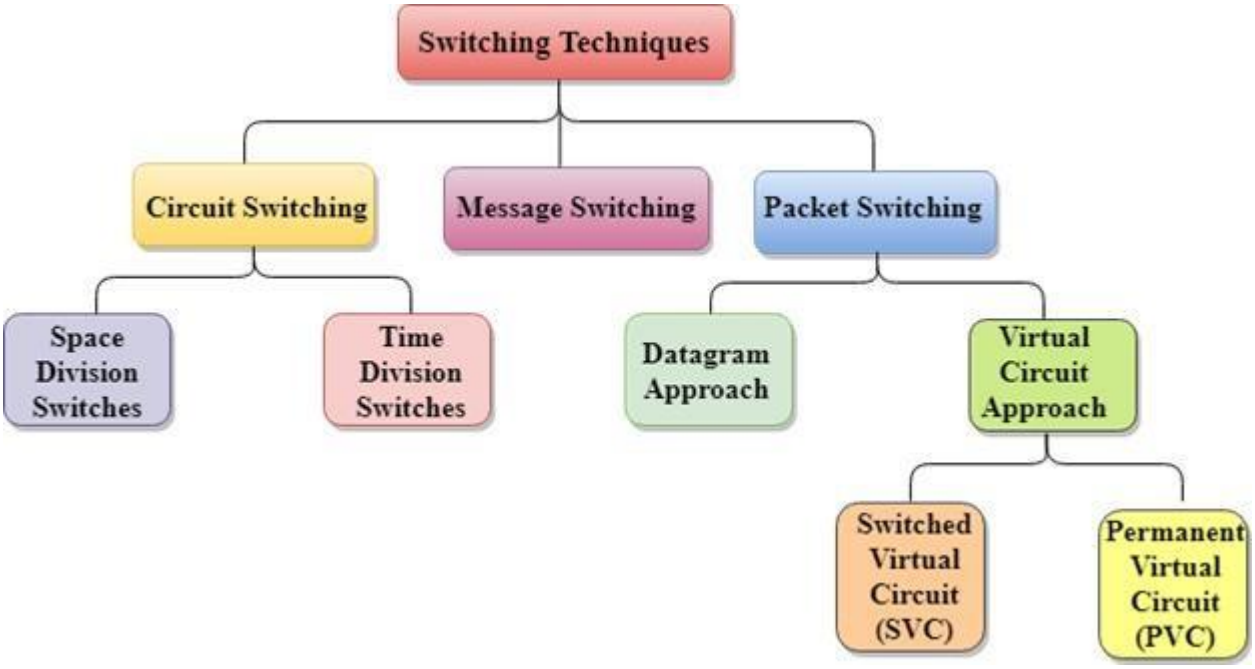


Fig 2.15 Classification Of Switching Techniques

2.7.1 Circuit Switching

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

Communication through circuit switching has 3 phases:

- Circuit establishment
- Data transfer
- Circuit Disconnect

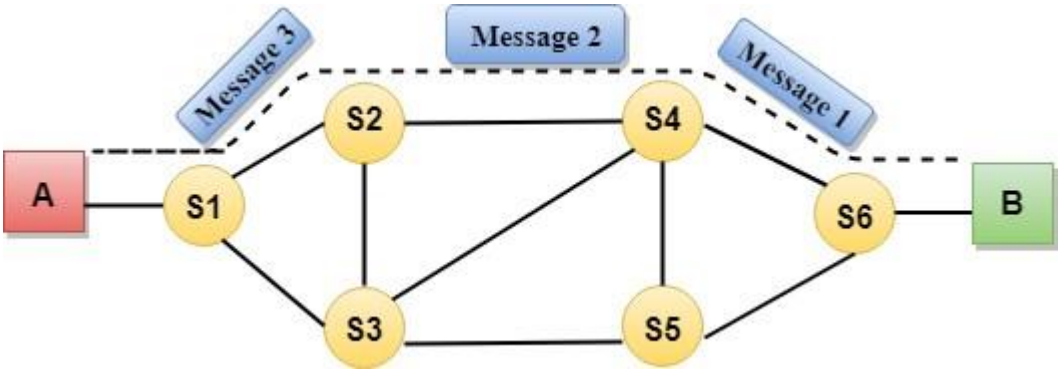


Fig 2.16 Switching Techniques

Circuit Switching can use either of the two technologies:

2.7.2 Space Division Switches:

- Space Division Switching is a circuit switching technology in which a single transmission path is accomplished in a switch by using a physically separate set of crosspoints.
- Space Division Switching can be achieved by using crossbar switch. A crossbar switch is a metallic crosspoint or semiconductor gate that can be enabled or disabled by a control unit.
- The Crossbar switch is made by using the semiconductor. For example, Xilinx crossbar switch using FPGAs.
- Space Division Switching has high speed, high capacity, and nonblocking switches.

Space Division Switches can be categorized in two ways:

- **Crossbar Switch**
- **Multistage Switch**

Crossbar Switch

The Crossbar switch is a switch that has n input lines and n output lines. The crossbar switch has n^2 intersection points known as **crosspoints.**

Disadvantage of Crossbar switch:

The number of crosspoints increases as the number of stations is increased. Therefore, it becomes very expensive for a large switch. The solution to this is to use a multistage switch.

Multistage Switch

- Multistage Switch is made by splitting the crossbar switch into the smaller units and then interconnecting them.
- It reduces the number of crosspoints.
- If one path fails, then there will be an availability of another path.

Advantages Of Circuit Switching:

- In the case of Circuit Switching technique, the communication channel is dedicated.
- It has fixed bandwidth.

Disadvantages Of Circuit Switching:

- Once the dedicated path is established, the only delay occurs in the speed of data transmission.
- It takes a long time to establish a connection approx 10 seconds during which no data can be transmitted.
- It is more expensive than other switching techniques as a dedicated path is required for each connection.
- It is inefficient to use because once the path is established and no data is transferred, then the capacity of the path is wasted.
- In this case, the connection is dedicated therefore no other data can be transferred even if the channel is free.

2.7.3 Message Switching

- Message Switching is a switching technique in which a message is transferred as a complete unit and routed through intermediate nodes at which it is stored and forwarded.
- In Message Switching technique, there is no establishment of a dedicated path between the sender and receiver.
- The destination address is appended to the message. Message Switching provides a dynamic routing as the message is routed through the intermediate nodes based on the information available in the message.

- Message switches are programmed in such a way so that they can provide the most efficient routes.
- Each and every node stores the entire message and then forward it to the next node. This type of network is known as **store and forward network.**
- Message switching treats each message as an independent entity.

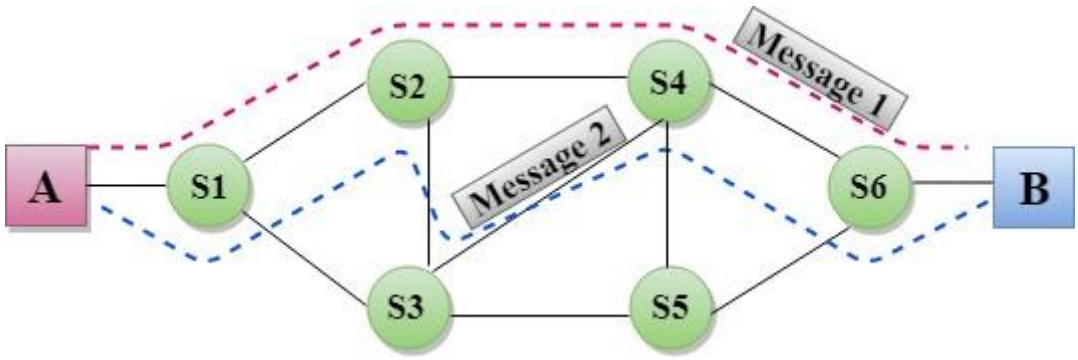


Fig 2.17 Message Switching

Advantages Of Message Switching

- Data channels are shared among the communicating devices that improve the efficiency of using available bandwidth.
- Traffic congestion can be reduced because the message is temporarily stored in the nodes.
- Message priority can be used to manage the network.
- The size of the message which is sent over the network can be varied. Therefore, it supports the data of unlimited size.

Disadvantages Of Message Switching

- The message switches must be equipped with sufficient storage to enable them to store the messages until the message is forwarded.
- The Long delay can occur due to the storing and forwarding facility provided by the message switching technique.

Packet Switching

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

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

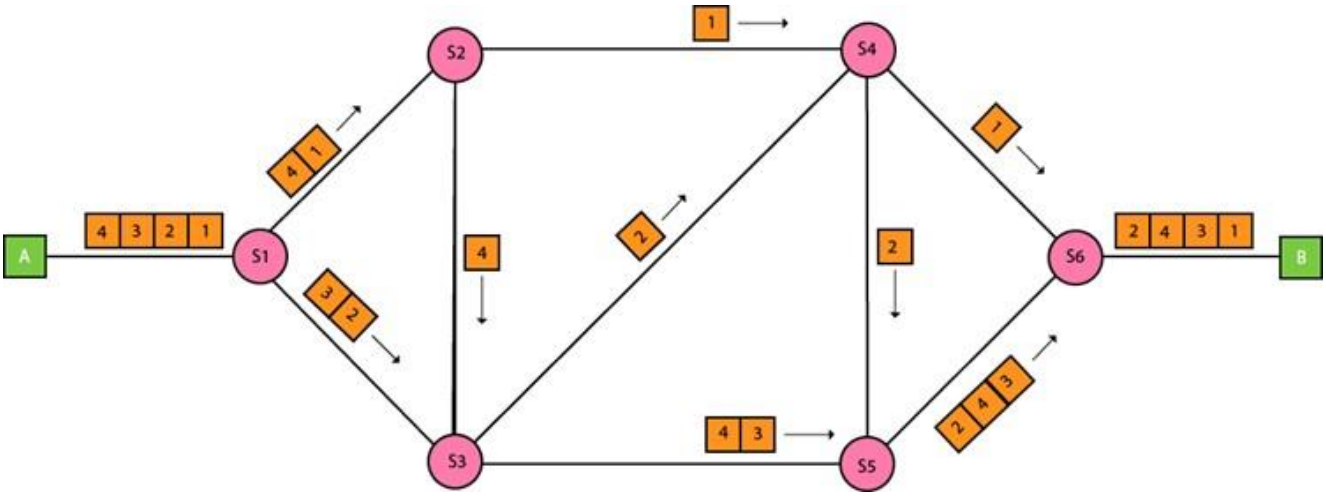


Fig 2.18 Packet Switching

Approaches Of Packet Switching:

There are two approaches to Packet Switching:

Datagram Packet switching:

- It is a packet switching technology in which packet is known as a datagram, is considered as an independent entity. Each packet contains the information about the destination and switch uses this information to forward the packet to the correct destination.
- The packets are reassembled at the receiving end in correct order.
- In Datagram Packet Switching technique, the path is not fixed.
- Intermediate nodes take the routing decisions to forward the packets.
- Datagram Packet Switching is also known as connectionless switching.

Virtual Circuit Switching

- Virtual Circuit Switching is also known as connection-oriented switching.
- In the case of Virtual circuit switching, a preplanned route is established before the messages are sent.
- Call request and call accept packets are used to establish the connection between sender and receiver.
- In this case, the path is fixed for the duration of a logical connection.

Let's understand the concept of virtual circuit switching through a diagram:

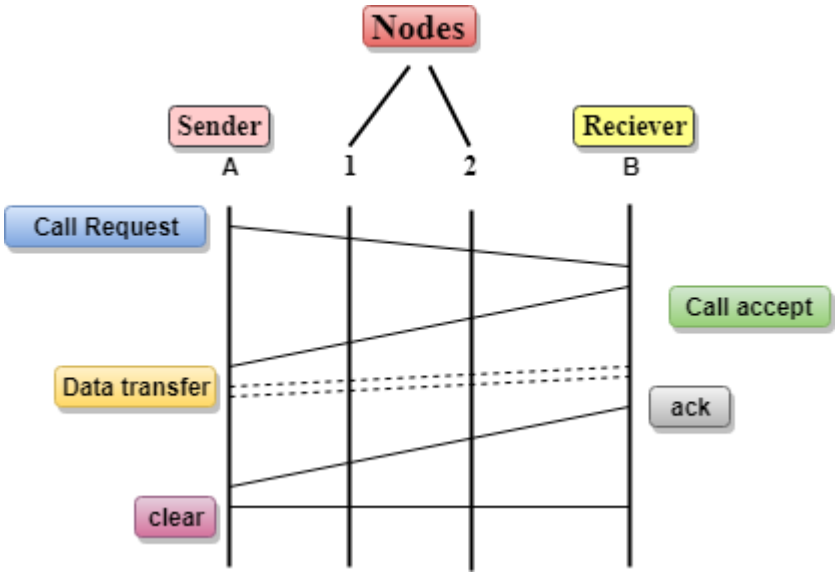


Fig 2.19 Virtual Circuit Switching

- In the above diagram, A and B are the sender and receiver respectively. 1 and 2 are the nodes.
- Call request and call accept packets are used to establish a connection between the sender and receiver.
- When a route is established, data will be transferred.
- After transmission of data, an acknowledgment signal is sent by the receiver that the message has been received.
- If the user wants to terminate the connection, a clear signal is sent for the termination.

Differences b/w Datagram approach and Virtual Circuit approach

Datagram approach	Virtual Circuit approach
Node takes routing decisions to forward the packets.	Node does not take any routing decision.
Congestion cannot occur as all the packets travel in different directions.	Congestion can occur when the node is busy, and it do other packets to pass through.
It is more flexible as all the packets are treated as an independent entity.	It is not very flexible.

Advantages Of Packet Switching:

- **Cost-effective:** In packet switching technique, switching devices do not require massive secondary storage to store the packets, so cost is minimized to some extent. Therefore, we can say that the packet switching technique is a cost-effective technique.
- **Reliable:** If any node is busy, then the packets can be rerouted. This ensures that the Packet Switching technique provides reliable communication.
- **Efficient:** Packet Switching is an efficient technique. It does not require any established path prior to the transmission, and many users can use the same communication channel simultaneously, hence makes use of available bandwidth very efficiently.

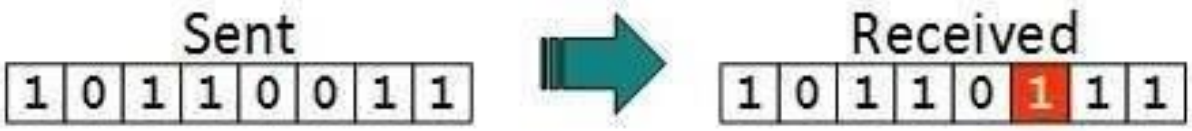
Disadvantages Of Packet Switching:

- Packet Switching technique cannot be implemented in those applications that require low delay and high-quality services.
- The protocols used in a packet switching technique are very complex and requires high implementation cost.
- If the network is overloaded or corrupted, then it requires retransmission of lost packets. It can also lead to the loss of critical information if errors are nor recovered.

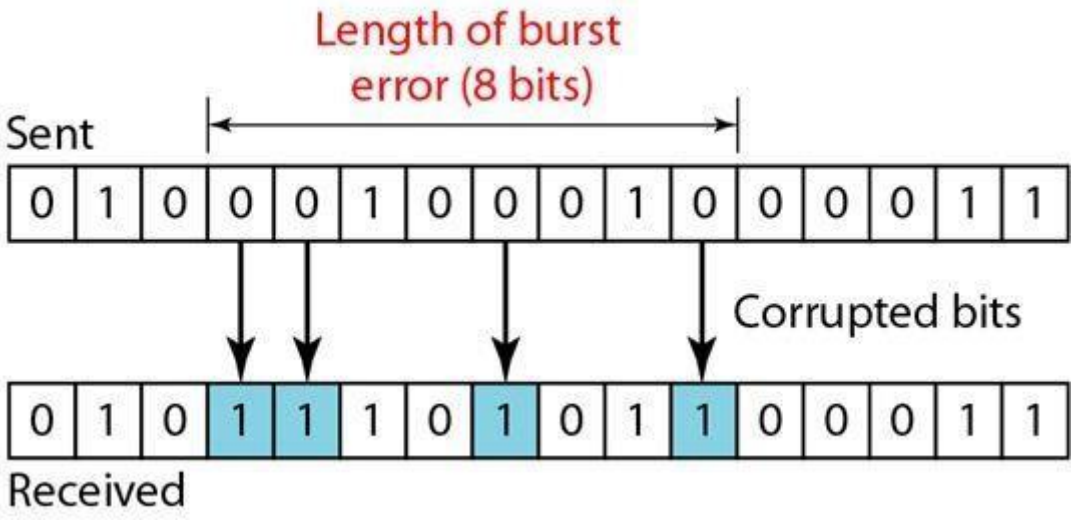
UNIT III DATA LINK LAYER

Errors:

- Whenever bits flow from one point to another, they are subject to unpredictable changes because of interference.
- This interference can change the shape of the signal.
- There may be three types of errors:
    - Single bit error
    - Burst error
- In a single-bit error, only 1 bit in the data unit has changed.



- A Burst error means that 2 or more bits in the data unit have changed.



- The central concept in detecting or correcting errors is redundancy.
- These redundant bits are added by the sender and removed by the receiver.
- To detect or correct errors, extra bits are added with data.

ERROR DETECTION:

- In error detection, the error in the data is detected.
- A single-bit error is the same as that of the burst error.
- **Example:** CRC – Cyclic Redundancy Check

ERROR CORRECTION:

- The exact number of bits that are corrupted.

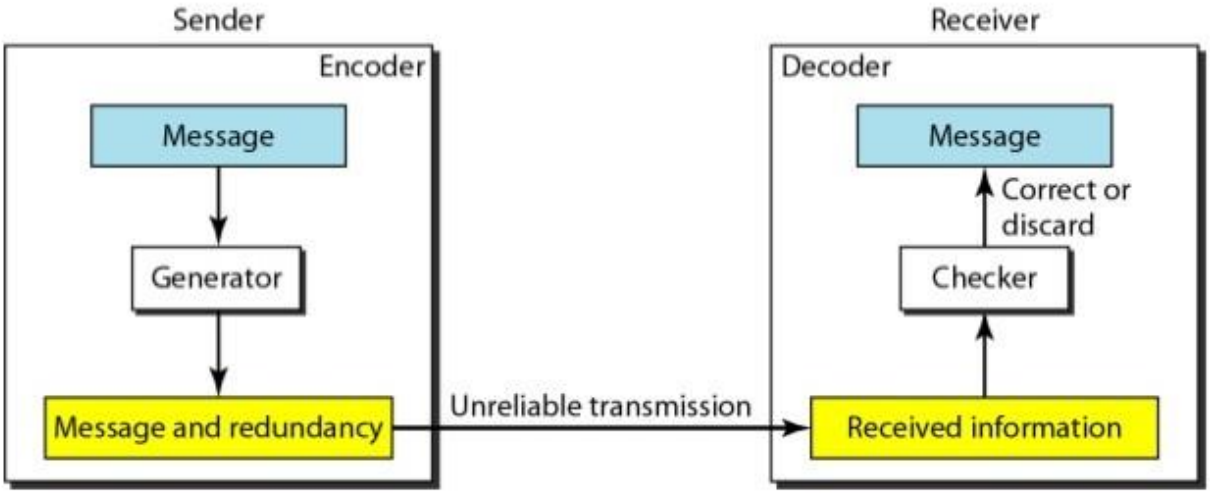
- Corrupted bits location in the message.
- Hence the number of errors and size of the message are important factors.

➢ There are two types of error correction.

  - Forward Error Correction
  - Retransmission

- It is the process in which the receiver tries to guess the message by using redundant bits.
- It is a technique in which the receiver detects the occurrence of an error and asks the sender to resend the message.
- Resending is repeated until a message arrives that the receiver believes is error-free.
- Not all errors can be detected.
- Redundancy is achieved through various coding schemes.
- The sender adds redundant bits through a process that creates a relationship between the redundant bits and the actual data bits.
- The receiver checks the relationship between the two sets of bits to detect or correct errors.

Figure 10.3 *The structure of encoder and decoder*



- Coding schemes are divided into two broad categories:

  ➢ Block Coding

  ➢ Convolution Coding

VRC:

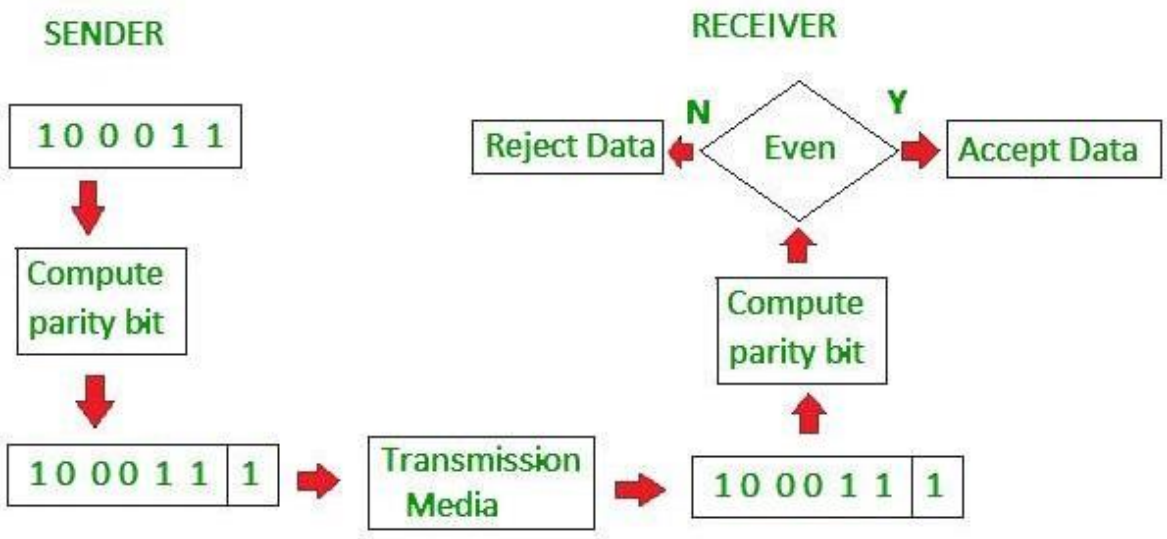
- Vertical Redundancy Check
- It is also known as parity check.
- Parity bit means nothing but an additional bit added to the data at the transmitter before transmitting the data.
- Before adding the parity bit, number of 1's or zeros is calculated in the data.
- Based on this calculation of data an extra bit is added to the actual information / data.
- The addition of parity bit to the data will result in the change of data string size.
- There is two types of parity bits in error detection, they are
    - Even parity
    - Odd parity

Even Parity:

- If the data has even number of 1's, the parity bit is 0.
- **Example**: 10000001 -> parity bit 0
- Odd number of 1's, the parity bit is 1.
- **Example**: 10010001 -> parity bit 1

Odd Parity:

- If the data has odd number of 1's, the parity bit is 0.
- **Example:** 10011101 -> parity bit 0
- Even number of 1's, the parity bit is 1.
- **Example:** 10010101 -> parity bit 1



3 bit data			Message with even parity		Message with odd parity	
A	B	C	Message	Parity	Message	Parity
0	0	0	000	0	000	1
0	0	1	001	1	001	0
0	1	0	010	1	010	0
0	1	1	011	0	011	1
1	0	0	100	1	100	0
1	0	1	101	0	101	1
1	1	0	110	0	110	1
1	1	1	111	1	111	0

PERFORMANCE:

- It can detect single bit error.
- It can detect burst errors only if the number of errors is odd.

Example:

Sender: 1100001 – parity bit -> 1

Transmitted data: 11100001

Receiver: Received data: 10100001 - parity bit -> 0

Rejects the data

Example:

Sender: 1100001 – parity bit -> 1

Transmitted data: 11100001

Receiver: Received data: 10100100 - parity bit -> 0

Rejects the data

Receiver: Received data: 10100101 - parity bit -> 1

Accepts the data

Example:

Append the even and odd parity bit after each block shown below using VRC

1110110 1101111 1110010

Even parity:

1110110 – parity bit -> 1

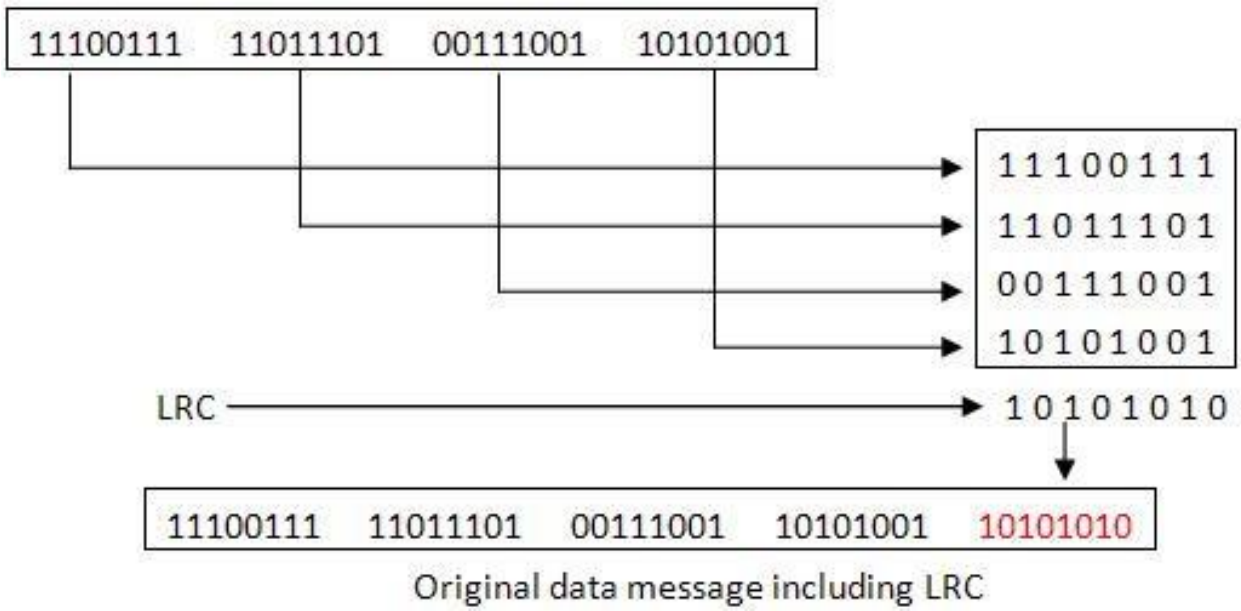
1101111 – parity bit -> 0

1110010 – parity bit -> 0

1	1110110	0	1101111	0	1110010

LRC:

- Longitudinal Redundancy Check.
- It is also known as two-dimensional parity check.
- In longitudinal redundancy method, a BLOCK of bits are arranged in a table format (in rows and columns) and then the parity bit is calculated for each column separately.
- The set of these parity bits are also sent along with our original data bits.
- Longitudinal redundancy check is a bit by bit parity computation.



Performance:

- This method can easily detect burst errors and single bit errors.
- It fails to detect the 2 bit errors occurred in same vertical slice.

CRC:

Cyclic Redundancy Check (CRC) is a block code invented by W. Wesley Peterson in 1961.

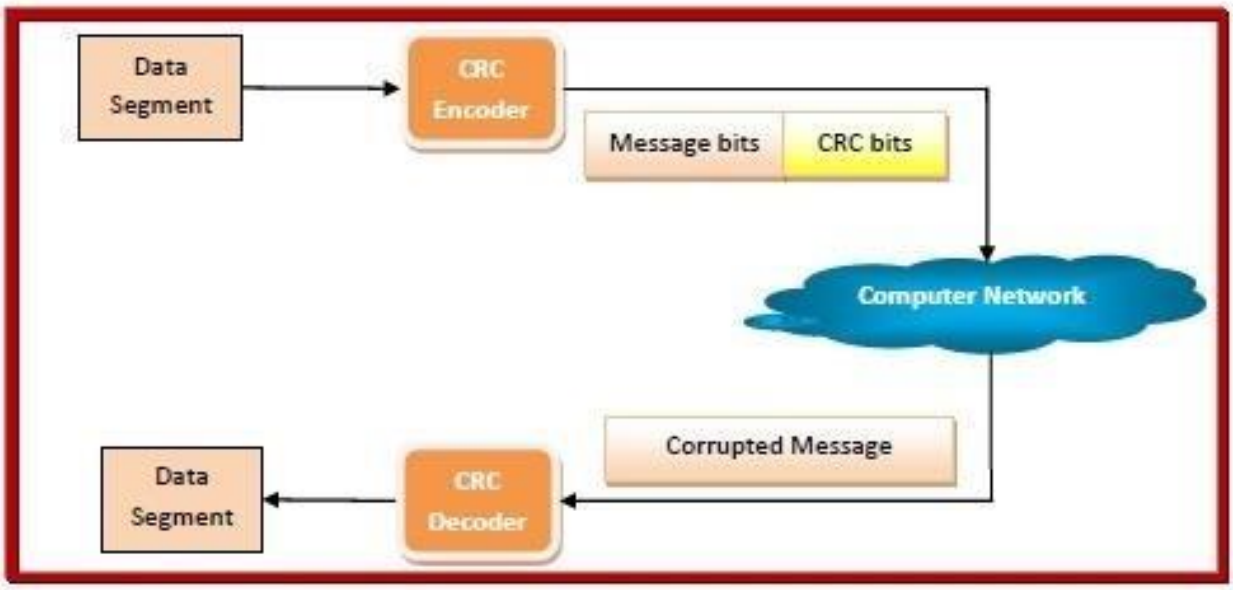
It is commonly used to detect accidental changes to data transmitted via telecommunications networks

and storage devices.

CRC involves binary division of the data bits being sent by a predetermined divisor agreed upon by the communicating system.

The divisor is generated using polynomials.

So, CRC is also called polynomial code checksum.



Encoding using CRC:

The communicating parties agrees upon the size of message block and the CRC divisor.

For example, the block chosen may be CRC (7, 4), where 7 is the total length of the block and 4 is the number of bits in the data segment.

The divisor chosen may be 1011.

The sender performs binary division of the data segment by the divisor.

It then appends the remainder called CRC bits to the end of data segment. This makes the resulting data unit exactly divisible by the divisor.

Decoding:

The receiver divides the incoming data unit by the divisor.

If there is no remainder, the data unit is assumed to be correct and is accepted.

Otherwise, it is understood that the data is corrupted and is therefore rejected.

The receiver may then send an erroneous acknowledgement back to the sender for retransmission.

CRC uses **Generator Polynomial** which is available on both sender and receiver side.

An example generator polynomial is of the form like $x^3 + x + 1$.

This generator polynomial represents key 1011.

Another example is $x^2 + 1$ that represents key 101.

n : Number of bits in data to be sent from sender side.

k : Number of bits in the key obtained from generator polynomial.

Sender Side:

The binary data is first augmented by adding k-1 zeros in the end of the data

Use ***modulo-2 binary division*** to divide binary data by the key and store remainder of division.

Append the remainder at the end of the data to form the encoded data and send the same

Receiver Side:

Perform modulo-2 division again and if the remainder is 0, then there are no errors.

Modulo-2 division:

The process of modulo-2 binary division is the same as the familiar division process for decimal numbers.

Just that instead of subtraction, XOR is used.

In each step, a copy of the divisor (or data) is XORed with the k bits of the dividend (or key).

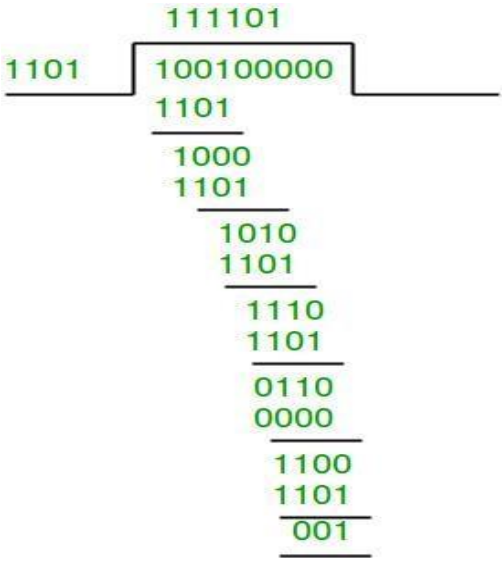
The result of the XOR operation (remainder) is (n-1) bits, which is used for the next step after 1 extra bit is pulled down to make it n bits long.

When there are no bits left to pull down, we have a result. The (n-1)-bit remainder which is appended at the sender side.

Example:

Data word to be sent - 100100

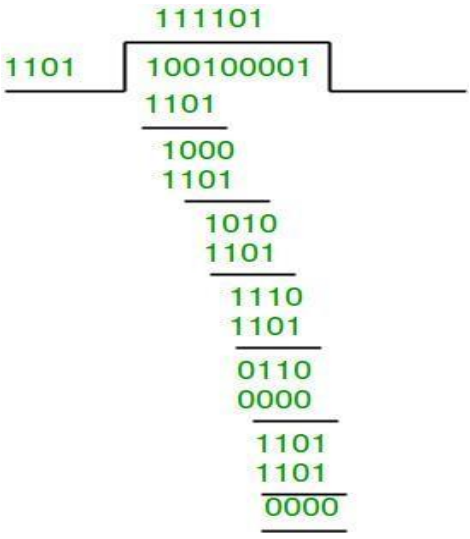
Key - 1101 [Or generator polynomial $x^3 + x^2 + 1$]



Therefore, the remainder is 001 and hence the encoded data sent is 100100001.

Receiver Side:

Code word received at the receiver side 100100001



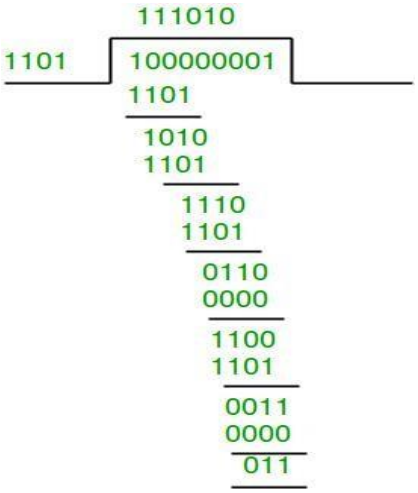
Therefore, the remainder is all zeros.

Hence, the data received has no error.

Receiver Side:

Let there be an error in transmission media

Code word received at the receiver side - 100000001



Since the remainder is not all zeroes, the error is detected at the receiver side.

HDLC:

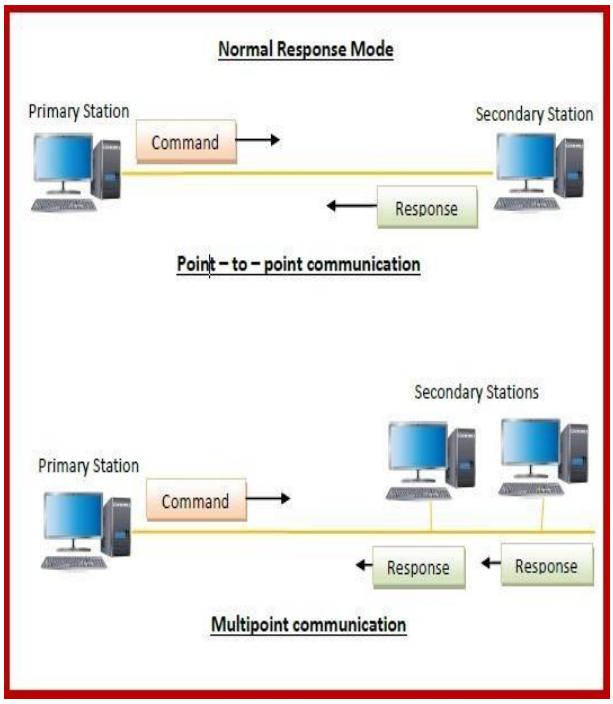
- It is a bit - oriented protocol for communication over point - to - point and multipoint communications.
- It implements ARQ mechanisms.

Configuration and Transfer Modes

- HDLC supports two types of transfer modes that can be used in different configurations:
    1. Normal response mode (NRM)
    2. Asynchronous balanced mode (ABM)

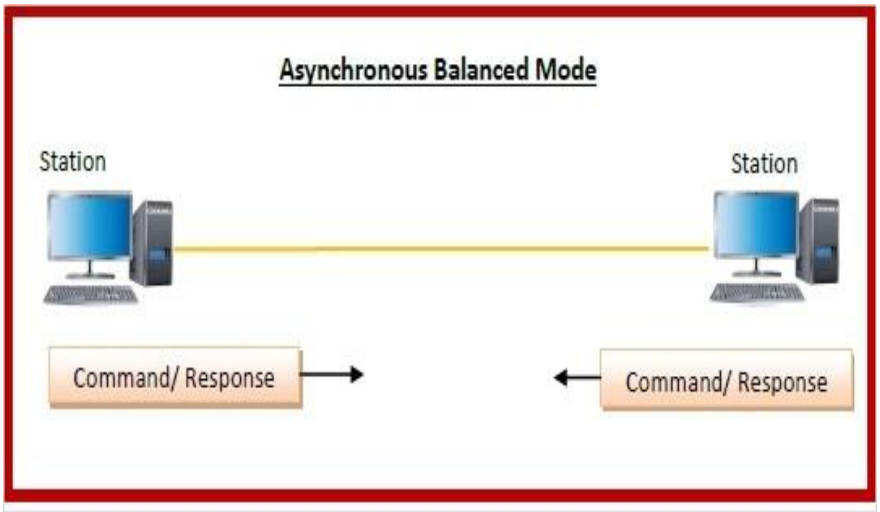
Normal Response Mode:

- In NRM, the station configuration is unbalanced.
- One primary station and multiple secondary stations.
- A Primary station can send commands.
- A secondary station can only respond.
- NRM is used for both point-to-point and multi-point links.



Asynchronous Balanced Mode:

- In ABM, the station configuration is balanced.
- The link is point-to-point.
- Each station can function as a primary and a secondary.
- It is the common mode.



Frames:

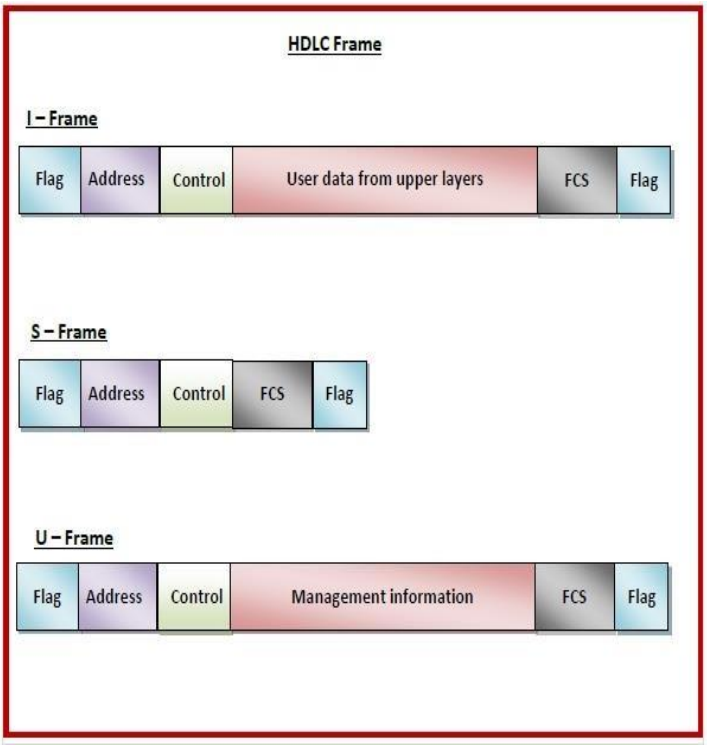
- HDLC defines three types of frames.

    1. Information frames (I-frames)
    2. Supervisory frames (S-frames)

3. Unnumbered frames (U-frames)

Frames:

- Each type of frame serves as an envelope for the transmission of a different type of message.
- I-frames are used to transport user data and control information relating to user data (piggybacking).
- S-frames are used only to transport control information.
- U-frames are reserved for system management (managing the links).



UNIT IV Network layer

4.1 Logical Addressing - IPv4 Addresses

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. IPv4 addresses are unique. They are unique in the sense that each address defines one, and only one, connection to the Internet. Two devices on the Internet can never have the same address at the same time. The IPv4 addresses are universal in the sense that the addressing system must be accepted by any host that wants to be connected to the Internet.

4.1.1 Address Space

A protocol such as IPv4 that defines addresses has an address space. An address space is the total number of addresses used by the protocol. If a protocol uses *N* bits to define an address, the address space is *2N* because each bit can have two different values (0 or 1) and *N* bits can have *2N* values.

IPv4 uses 32-bit addresses, which means that the address space is 232 or 4,294,967,296 (more than 4 billion). This means that, theoretically, if there were no restrictions, more than 4 billion devices could be connected to the Internet.

4.1.2 Notations

There are two prevalent notations to show an IPv4 address: binary notation and dotted decimal notation.

a. Binary Notation

In binary notation, the IPv4 address is displayed as 32 bits. Each octet is often referred to as a byte. So it is common to hear an IPv4 address referred to as a 32-bit address or a 4-byte address. The following is an example of an IPv4 address in binary notation.

01110101 10010101 00011101 00000010

b. Dotted-Decimal Notation

To make the IPv4 address more compact and easier to read, Internet addresses are usually written in decimal form with a decimal point (dot) separating the bytes. The following is the dotted-decimal notation of the above address:

117.149.29.2

Figure 4.1 shows an IPv4 address in both binary and dotted-decimal notation. Note that because each byte (octet) is 8 bits, each number in dotted-decimal notation is a value ranging from 0 to 255

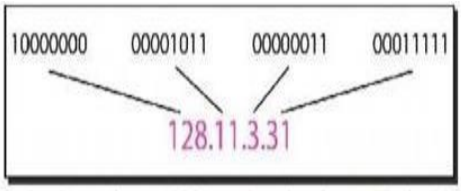


Figure 4.1 Dotted-decimal notation and binary notation for an IPv4 address

Example 4.1

Change the following IPv4 addresses from binary notation to dotted-decimal notation.

a)10000001 00001011 00001011 11101111

b)11000001 10000011 00011011 11111111

Solution

We replace each group of 8 bits with its equivalent decimal number and add dots for separation.

a)129.11.11.239 b)193.131.27.25

Example:

Change the following IPv4 addresses from

dotted-decimal notation to binary notation.

a)111.56.45.78

b)221.34.7.82

Solution

We replace each decimal number with its binary equivalent. a)01101111 00111000

00101101 01001110

b)11011101 00100010 00000111 01010010

Example 4.3

Find the error, if any, in the following IPv4 addresses.

a)111.56.045.78

b)221.34.7.8.20

c)75.45.301.14

d)11100010.23.14.67

Solution

a) There must be no leading zero (045).

b) There can be no more than four numbers in an IPv4 address.

c) Each number needs to be less than or equal to 255 (301 is outside this range).

d) A mixture of binary notation and dotted-decimal notation is not allowed.

4.1.3 Classful Addressing

IPv4 addressing, at its inception, used the concept of classes. This architecture is called classful addressing. In classful addressing, the address space is divided into five classes: A, B, C, D, and E.

We can find the class of an address when given the address in binary notation or dotted-decimal notation. If the address is given in binary notation, the first few bits can immediately tell us the class of the address. If the address is given in decimal- dotted notation, the first byte defines the class. Both methods are shown in Figure 3.2

Example 4.4

Find the class of each address.

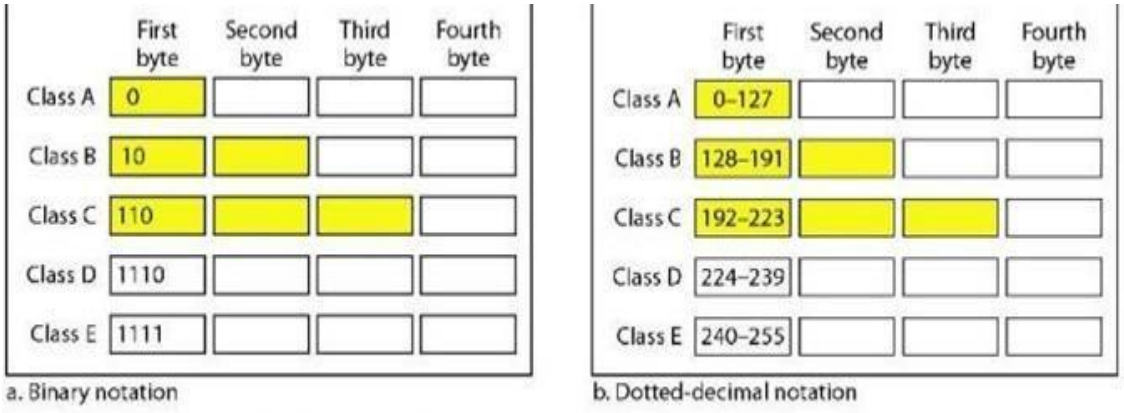


Figure 4.2 Finding the classes in binary and dotted-decimal notation

a)00000001 00001011 00001011 11101111

b)11000001 10000011 00011011 11111111

c) 14.23.120.8

d)252.5.15.111

Solution

a) The first bit is O. This is a class A address.

b) The first 2 bits are 1; the third bit is O. This is a class C address.

c) The first byte is 14 (between 0 and 127); the class is A.

d) The first byte is 252 (between 240 and 255); the class is E.

4.1.4 Classes and Blocks

One problem with classful addressing is that each class is divided into a fixed number of blocks with each block having a fixed size as shown in Table 3.1.

Class A addresses were designed for large organizations with a large number of attached hosts or routers. Class B addresses was designed for midsize organizations with tens of thousands of attached hosts or routers. Class C addresses were designed for small organizations with a small number of attached hosts or routers. Class D addresses were designed for multicasting. The class E addresses were reserved for future use. In c1assfnl addressing, a large part of the available

Table 4.1 Number of blocks and block size in classful IPv4 addressing

Class	*Number of Blocks*	*Block Size*	*Application*
A	128	16,777,216	Unicast
B	16,384	65,536	Unicast
C	2,097,152	256	Unicast
D	1	268,435,456	Multicast
E	1	268,435,456	Reserved

addresses were wasted

Netid and Hostid

In classful addressing, an IP address in class A, B, or C is divided into netid and hostid.

These parts are of varying lengths, depending on the class of the address. The netid is in color, the hostid is in white. Note that the concept does not apply to classes D and E. In class A, one byte defines the netid and three bytes define the hostid. In class B, two bytes define the netid and two bytes define the hostid. In class C, three bytes define the netid and one byte defines the hostid.

Mask

Although the length of the netid and hostid (in bits) is predetermined in classful addressing, we can also use a mask (also called the default mask), a 32-bit number made of contiguous Is followed by contiguous as. The masks for classes A, B, and C are shown in Table 4.2. The concept does not apply to classes D and E

Table 4.2 Default mask for classful addressing

Class	*Binary*	*Dotted-Decimal*	*CIDR*
A	11111111 00000000 00000000 00000000	255.0.0.0	/8
B	11111111 11111111 00000000 00000000	255.255.0.0	/16
C	11111111 11111111 11111111 00000000	255.255.255.0	/24

The mask can help us to find the netid and the hostid. For example, the mask for a class A address has eight 1s, which means the first 8 bits of any address in class A define the netid; the next 24 bits define the hostid. The last column of Table 4.2 shows the mask in the form */n* where *n* can be 8, 16, or 24 in classful addressing. This notation is also called slash notation orClassless Interdomain Routing (CIDR) notation. The notation is used in classless addressing,

Subnetting

If an organization was granted a large block in class A or B, it could divide the addressesinto several contiguous groups and assign each group to smaller networks (called subnets). Subnetting increases the number of Is in the mask.

Supernetting

In supernetting, an organization can combine several class C blocks to create a larger range of addresses. In other words, several networks are combined to create a supernetwork or a supemet. An organization can apply for a set of class C blocks instead of just one.

Address Depletion

The flaws in classful addressing scheme combined with the fast growth of the Internet led to the near depletion of the available addresses. Yet the number of devices on the Internet is much less than the 232 address space. We have run out of class A and B addresses, and a class C block is too small for most midsize organizations. One solution that has alleviated the problem is the idea of classless addressing.

4.1.5 Classless Addressing

To overcome address depletion and give more organizations access to the Internet, classless addressing was designed and implemented. In this scheme, there are no classes, but the addresses are still granted in blocks.

Address Blocks

In classless addressing, when an entity, small or large, needs to be connected to the Internet, it is granted a block (range) of addresses. The size of the block varies based on the nature and size of the entity. For example, a household may be given only two addresses; a large organization may be given thousands of addresses. An ISP, as the Internet service provider, may be given thousands or hundreds of thousands based on the number of customers it may serve.

Restrictions to simplify the handling of addresses, the Internet authorities impose three restrictions on classless address blocks:

1. The addresses in a block must be contiguous, one after another.
2. The number of addresses in a block must be a power of 2 (I, 2, 4, 8 ...).
3. The first address must be evenly divisible by the number of addresses.

Example 4.5

Figure 3.3 shows a block of addresses, in both binary and dotted-decimal notation, granted to a small business that needs 16 addresses.

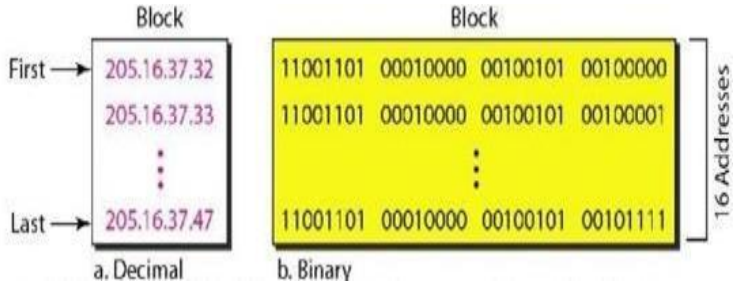


Fig 4.A block of 16 bit address granted to a small organization

We can see that the restrictions are applied to this block. The addresses are contiguous. The number of addresses is a power of 2 (16 = 24), and the first address is divisible by 16. The first address, when converted to a decimal number, is 3,440,387,360, which when divided by 16 results in 215,024,210.

Mask

A better way to define a block of addresses is to select any address in the block and the mask. As we discussed before, a mask is a 32-bit number in which the n leftmost bits are 1s and the 32 - n rightmost bits are 0s. However, in classless addressing the mask for a block can take any value from 0 to 32. It is very convenient to give just the value of n preceded by a slash.

First Address: The first address in the block can be found by setting the 32 - n rightmost bits inthe binary notation of the address to 0s. The first address in the block can be found by setting the rightmost 32 - n bits to 0s.

Example 4.6

A block of addresses is granted to a small organization. We know that one of the addresses is 205.16.37.39/28. What is the first address in the block?

Solution

The binary representation of the given address is 11001101 00010000 00100101 00100 I

11. If we set 32 - 28 rightmost bits to 0, we get 11001101 000100000100101 0010000 or 205.16.37.32.

Last Address: The last address in the block can be found by setting the 32 - n rightmost bits inthe binary notation of the address to Is. The last address in the block can be found by setting the rightmost 32 - n bits to Is.

Example 4.7

Find the last address for the block in Example 3.6.

Solution

The binary representation of the given address is 11001101 00010000001001010010 0111. If we set 32 - 28 rightmost bits to 1, we get 11001101 00010000 001001010010 1111 or 205.16.37.47.

Number of Addresses: The number of addresses in the block is the difference between the lastand first address. It can easily be found using the formula 232- n.

Example 4.8

Find the number of addresses in Example 3.6.

Solution

The value of n is 28, which means that number of addresses is 232- 28 or 16.

Example 4.9

Another way to find the first address, the last address, and the number of addresses is to represent the mask as a 32-bit binary (or 8-digit hexadecimal) number. This is particularly useful when we are writing a program to find these pieces of information. In Example 19.5 the /28 can be represented as 11111111 11111111 11111111 11110000 (twenty-eight 1s and four 0s).

Find a. The first address

b. The last address

c. The number of addresses

Solution

a. The first address can be found by ANDing the given addresses with the mask. ANDing here is done bit by bit. The result of ANDing 2 bits is 1 if both bits are Is; the result is 0 otherwise.

Address: 11001101 00010000 00100101 00100111

Mask: 11111111 11111111 11111111 11110000

First address: 11001101 00010000 00100101 00100000

b. The last address can be found by ORing the given addresses with the complement of the

mask. ORing here is done bit by bit. The result of ORing 2 bits is 0 if both bits are 0s; the result is 1 otherwise. The complement of a number is found by changing each 1 to 0 and each 0 to 1.

Address: 11001101 00010000 00100101 00100111

Mask complement: 00000000 00000000 00000000 00001111

Last address: 11001101 00010000 00100101 00101111

c. The number of addresses can be found by complementing the mask, interpreting it as a decimal number, and adding 1 to it.

Mask complement: 000000000 00000000 00000000 00001111

Number of addresses: 15 + 1 =16

Network Addresses

A very important concept in IP addressing is the network address. When an organization is given a block of addresses, the organization is free to allocate the addresses to the devices that need to be connected to the Internet. The first address in the class, however, is normally (not always) treated as a special address. The first address is called the network address and defines the organization network. It defines the organization itself to the rest of the world.

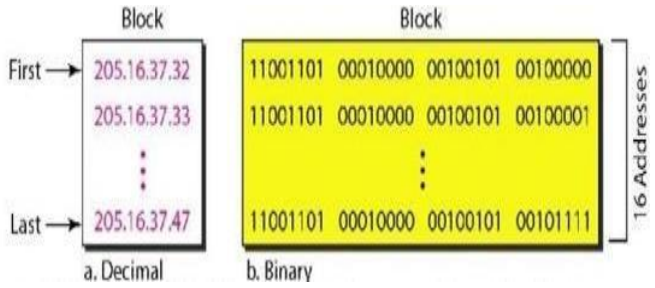


Fig 4.5 A network for the block 205.16.37.32/28

The organization network is connected to the Internet via a router. The router has two addresses. One belongs to the granted block; the other belongs to the network that is at the other side of the router. We call the second address x.y.z.t/n because we do not know anything about the network it is connected to at the other side. All messages destined for addresses in the organization block (205.16.37.32 to 205.16.37.47) are sent, directly or indirectly, to x.y.z.t/n. We say directly or indirectly because we do not know the structure of the network to which the other side of the router is connected. The first address in a block is normally not assigned to any device; it is used as the network address that represents the organization to the rest of the world.

Hierarchy

IP addresses, like other addresses or identifiers we encounter these days, have levels of

hierarchy. For example, a telephone network in North America has three levels of hierarchy. The leftmost three digits define the area code, the next three digits define the exchange, the last four digits define the connection of the local loop to the central office. Figure 3.5 shows the structure of a hierarchical telephone number

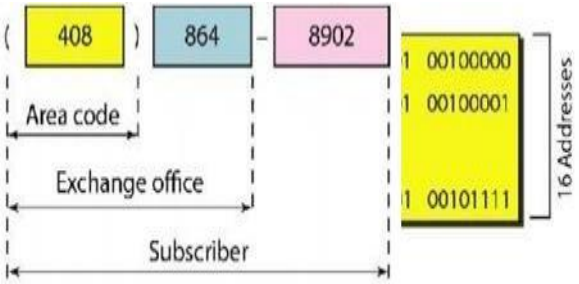


Figure 4.5 Hierarchy in a telephone network in North America

Two-Level Hierarchy: No Subnetting

An IP address can define only two levels of hierarchy when not subnetted. The n leftmost bits of the address x.y.z.tJn define the network (organization network); the 32 – n rightmost bits define the particular host (computer or router) to the network. The two common terms are prefix and suffix. The part of the address that defines the network is called the prefix; the part that defines the host is called the suffix. Figure 3.6 shows the hierarchical structure of an IPv4 address.

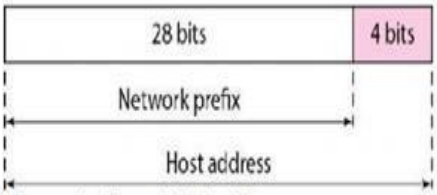


Figure4 .6 Two levels of hierarchy in an IPv4 address

The prefix is common to all addresses in the network; the suffix changes from one device to another. Each address in the block can be considered as a two-level hierarchical

structure: the leftmost n bits (prefix) define the network; the rightmost 32 - n bits define the host. Note that applying the mask of the network, /26 to any of the addresses gives us the network address 17.12.14.0/26. We leave this proof to the reader. We can say that through subnetting, we

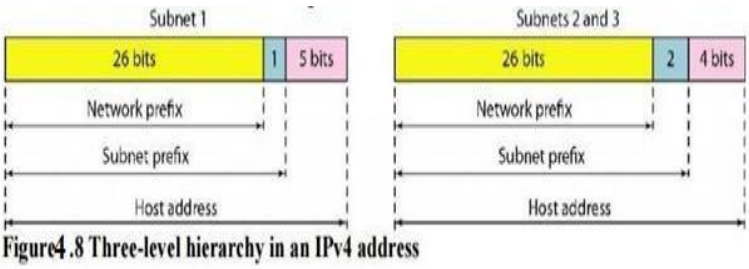


Figure4 .8 Three-level hierarchy in an IPv4 address

have three levels of hierarchy. Note that in our example, the subnet prefix length can differ for the subnets as shown in Figure 4.8.

More Levels of Hierarchy

The structure of classless addressing does not restrict the number of hierarchical levels. An organization can divide the granted block of addresses into subblocks. Each subblock can in turn be divided into smaller subblocks. And so on. One example of this is seen in the ISPs. A national ISP can divide a granted large block into smaller blocks and assign each of them to a regional ISP. A regional ISP can divide the block received from the national ISP into smaller blocks and assign each one to a local ISP. A local ISP can divide the block received from the regional ISP into smaller blocks and assign each one to a different organization. Finally, an organization can divide the received block and make several subnets out of it.

4.1.6 Address Allocation

The next issue in classless addressing is address allocation. The ultimate responsibility of address allocation is given to a global authority called the Internet Corporation for Assigned Names and Addresses (ICANN). However, ICANN does not normally allocate addresses to individual organizations. It assigns a large block of addresses to an ISP. Each ISP, in turn, divides its assigned block into smaller subblocks and grants the subblocks to its customers. In other words, an ISP receives one large block to be distributed to its Internet users. This is called address aggregation: many blocks of addresses are aggregated in one block and granted to one ISP.

Example 4.10

An ISP is granted a block of addresses starting with 190.100.0.0/16 (65,536 addresses). The ISP needs to distribute these addresses to three groups of customers as follows:The first group has 64 customers; each needs 256 addresses.

a) The second group has 128 customers; each needs 128 addresses.

b) The third group has 128 customers; each needs 64 addresses.

Design the subblocks and find out how many addresses are still available after these allocations.

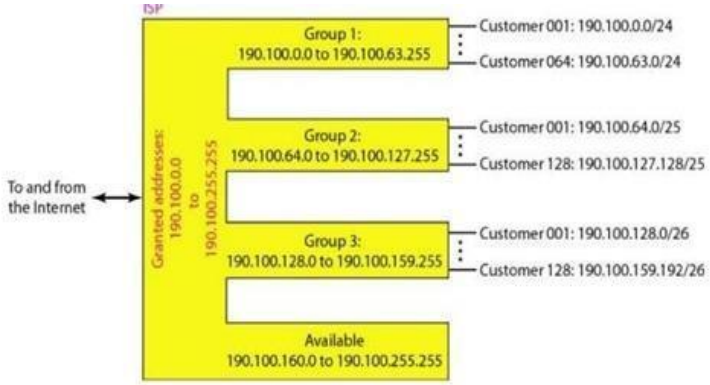


Figure 4.9 An example of address allocation and distribution by an ISP

Solution

Figure 4.9 shows the situation.

1. Group 1

For this group, each customer needs 256 addresses. This means that 8 (log2256) bits are needed to define each host. The prefix length is then 32 - 8 =24. The addresses are

1st Customer: 190.100.0.0/24 100.0.255/24

2nd Customer: 190.100.1.0/24190 190.100.1.255/24

64th Customer: 190.100.63.0/24 190.100.63.255/24 Total =64

X 256 =16,384

2. Group2

For this group, each customer needs 128 addresses. This means that 7 (10g2 128) bits are needed to define each host. The prefix length is then 32 - 7 =25. The addresses are

3. Group3

For this group, each customer needs 64 addresses. This means that 6 (log2 64) bits are needed to each host. The prefix length is then 32 - 6 =26. The addresses are

1st Customer: 190.100.128.0/26 190.100.128.63/26

2nd Customer: 190.100.128.64/26 190.100.128.127/26

128th Customer: 190.100.159.192/26 190.100.159.255/26

Total =128 X 64 =8192

Number of granted addresses to the ISP: 65,536 Number of allocated addresses by the ISP: 40,960 Number of available addresses: 24,576

Despite all short-term solutions, such as classless addressing, Dynamic Host Configuration Protocol (DHCP) and NAT, address depletion is still a long-term problem for the Internet. This and other problems in the IP protocol itself, such as lack of accommodation for real-time audio and video transmission, and encryption and authentication of data for some applications, have been the motivation for IPv6.

Structure

An IPv6 address consists of 16 bytes (octets); it is 128 bits long. An IPv6 address is 128 bits long.

Hexadecimal Colon Notation

To make addresses more readable, IPv6 specifies hexadecimal colon notation. In this notation, 128 bits is divided into eight sections, each 2 bytes in length. Two bytes in hexadecimal notation

requires four hexadecimal digits. Therefore, the address consists of 32 hexadecimal digits, with every four digits separated by a colon

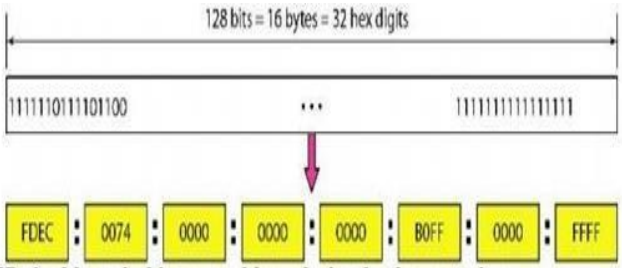


Figure 4.10 IPv6 address in binary and hexadecimal colon notation

Abbreviation

Although the IP address, even in hexadecimal format, is very long, many of the digits are zeros. In this case, we can abbreviate the address. The leading zeros of a section (four digits between two colons) can be omitted. Only the leading zeros can be dropped, not the trailing zeros**. Using this form of abbreviation, 0074 can be written as 74, 000F as F, and 0000 as 0.**

Example 4.11

Expand the address 0:15:: 1:12:1213 to its original.

Solution

We first need to align the left side of the double colon to the left of the original pattern and the

right side of the double colon to the right of the original pattern to find now many Os we need to replace the double colon.

xxxx: xxxx: xxxx: xxxx: xxxx: xxxx: xxxx: xxxx 0: 15: :

l: 12: 1213

This means that the original address is

0000: 0015: 0000: 0000: 0000: 0001: 0012: 1213

1. Address Space

IPv6 has a much larger address space; 2128 addresses are available. The designers of IPv6 divided the address into several categories. A few leftmost bits, called the *type prefix,* in each address define its category. The type prefix is variable in length, but it is designed such that no code is identical to the first part of any other code. In this way, there is no ambiguity; when an address is given, the type prefix can easily be determined.

2. Unicast Addresses

A **unicast address** defines a single computer. The packet sent to a unicast address must be delivered to that specific computer. IPv6 defines two types of unicast addresses: geographically based and provider-based. We discuss the second type here; the first type is left for future

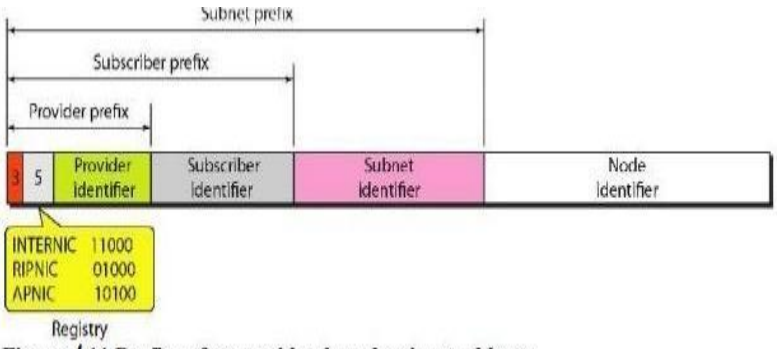


Figure 4.11 Prefixes for provider-based unicast address

definition. The provider-based address is generally used by a normal host as a unicast address. Fields for the provider-based address are as follows:

- **Type identifier:** This 3-bit field defines the address as a provider-based address.

- **Registry identifier.** This 5-bit field indicates the agency that has registered the

Type Prefix	*Type*	*Fraction*
0000 0000	Reserved	1/256
0000 0001	Unassigned	1/256
0000 001	ISO network addresses	1/128
0000 010	IPX (Novell) network addresses	1/128
0000 011	Unassigned	1/128
0000 1	Unassigned	1/32
0001	Reserved	1/16
001	Reserved	1/8
010	**Provider-based unicast addresses**	**1/8**
011	Unassigned	1/8
100	Geographic-based unicast addresses	1/8
101	Unassigned	1/8
110	Unassigned	1/8
1110	Unassigned	1/16
1111 0	Unassigned	1/32
1111 10	Unassigned	1/64
1111 110	Unassigned	1/128
1111 1110 0	Unassigned	1/512
1111 1110 10	Link local addresses	1/1024
1111 1110 11	Site local addresses	1/1024
1111 1111	Multicast addresses	1/256

address.Currently three registry centers have been defined. INTERNIC (code 11000) is the center for North America; RIPNIC (code 01000) is the center for European registration; and APNIC (code 10100) is for Asian and Pacific countries.

- **Provider identifier.** This variable-length field identifies the provider for Internet access(such as an ISP). A 16-bit length is recommended for this field.

- **Subscriber identifier.** When an organization subscribes to the Internet through a provider, itis assigned subscriber identification. A 24-bit length is recommended for this field.

- **Subnet identifier.** Each subscriber can have many different subnetworks, and eachsubnetwork can have an identifier. The subnet identifier defines a specific subnetwork under the territory of the subscriber. A 32-bit length is recommended for this field.

- **Node identifier.** The last field defines the identity of the node connected to a subnet. Alength of 48 bits is recommended for this field to make it compatible with the 48-bit link (physical) address used by Ethernet. In the future, this link address will probably be the same as the node physical address.

4.2.1 Multicast Addresses

Multicast addresses are used to define a group of hosts instead of just one. A packet sent to a multicast address must be delivered to each member of the group.

The second field is a flag that defines the group address as either permanent or transient. A permanent group address is defined by the Internet authorities and can be accessed at all times. A

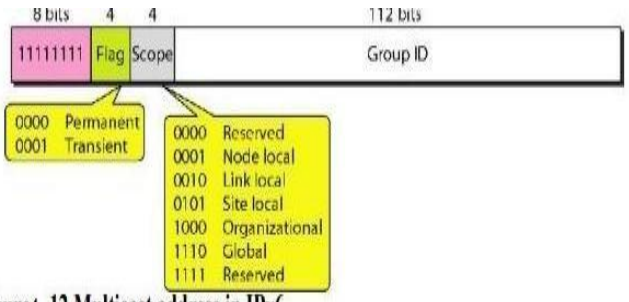


Figure4 .12 Multicast address in IPv6

transient group address, on the other hand, is used only temporarily. Systems engaged in a teleconference, for example, can use a transient group address. The third field defines the scope of the group address. Many different scopes have been defined.

4.2.2 All cast Addresses

IPv6 also defines anycast addresses. An anycast address, like a multicast address, also defines a group of nodes. However, a packet destined for an anycast address is delivered to only one of the members of the anycast group, the nearest one (the one with the shortest route). Although the definition of an anycast address is still debatable, one possible use is to assign an anycast address to all routers of an ISP that covers a large logical area in the Internet.

4.2.3 Reserved Addresses

Another category in the address space is the reserved address. These addresses start with eight Os (type prefix is 00000000). A few subcategories are defined in this category.

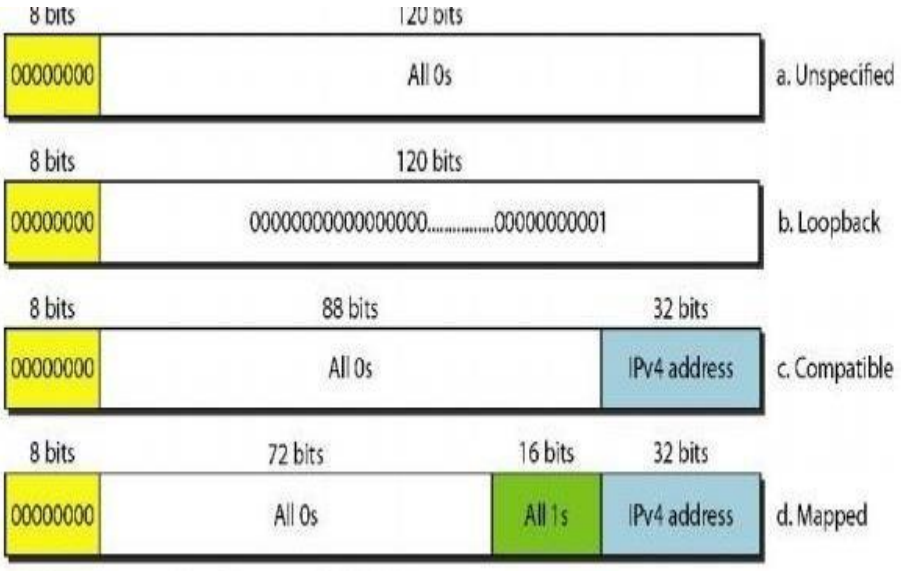


Figure 4.13 Reserved addresses in IPv6

An unspecified address is used when a host does not know its own address and sends an inquiry to find its address. A loopback address is used by a host to test itself without going into the network. A compatible address is used during the transition from IPv4 to IPv6.

4.2.5 Local Addresses

These addresses are used when an organization wants to use IPv6 protocol without being connected to the global Internet. In other words, they provide addressing for private networks. Nobody outside the organization can send a message to the nodes using these addresses. Two types of addresses are defined for this purpose.

Figure 4.14 Local addresses in IPv6

4.3 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.

Limitations

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.

4.3.1 Mapping Logical to Physical Address: ARP

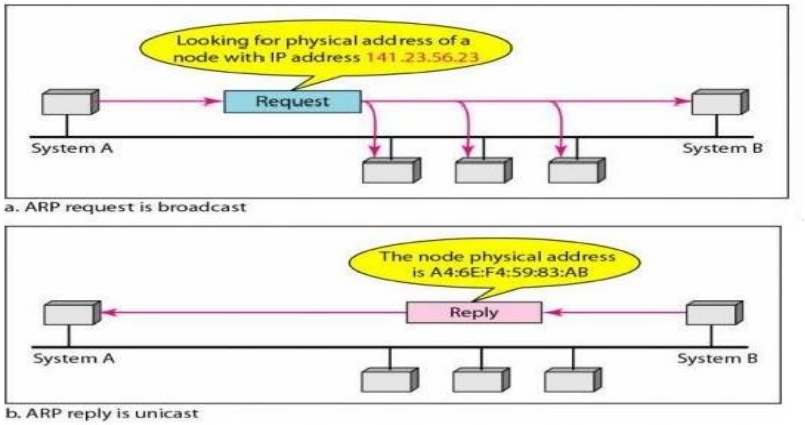


Fig 4.15 ARP Operation

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

4.3.1.1 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 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

The fields are as follows:

Hardware type: This is a 16-bit field defining the type of the network on which ARP isrunning. 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

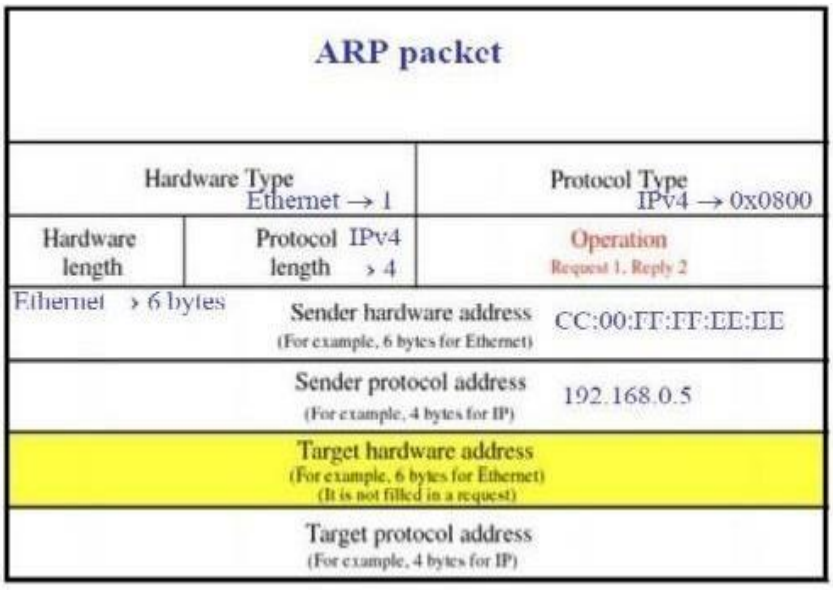


Fig 4.16 ARP Packet

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

Hardware length: This is an 8-bit field defining the length of the physical address inbytes. For example, for Ethernet the value is 6.

Protocol length: This is an 8-bit field defining the length of the logical address inbytes.For example, for the IPv4 protocol the value is 4.

Operation: This is a 16-bit field defining the type of packet. Two packet types aredefined: ARP request (1) and ARP reply (2).

Sender hardware address: This is a variable-length field defining the physical addressof the sender. For example, for Ethernet this field is 6 bytes long.

Sender protocol address: This is a variable-length field defining the logical (forexample, IP) address of the sender. For the IP protocol, this field is 4 bytes long.

Target hardware address: This is a variable-length field defining the physical addressof the target. For example, for Ethernet this field is 6 bytes long. For an ARP request message, this field is alIOs because the sender does not know the physical address of the target.

Target protocol address: This is a variable-length field defining the logical (forexample, IP) address of the target. For the IPv4 protocol, this field is 4 bytes long.

4.3.2 Encapsulation

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

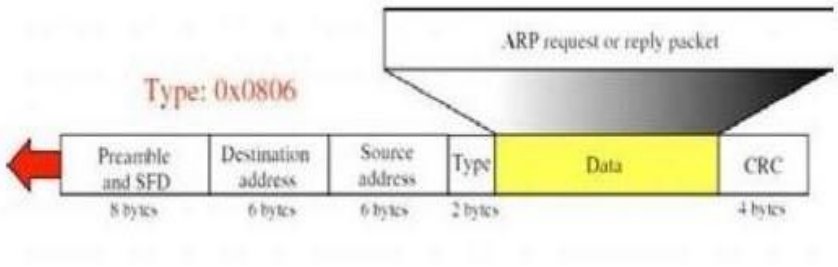


Fig 4.18 Encapsulation

Operation

The steps involved in an ARP process:

1. The sender knows the IP address of the target.
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 0s.
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.

4. Four Different Cases:

The following are four different cases in which the services of ARP can be used

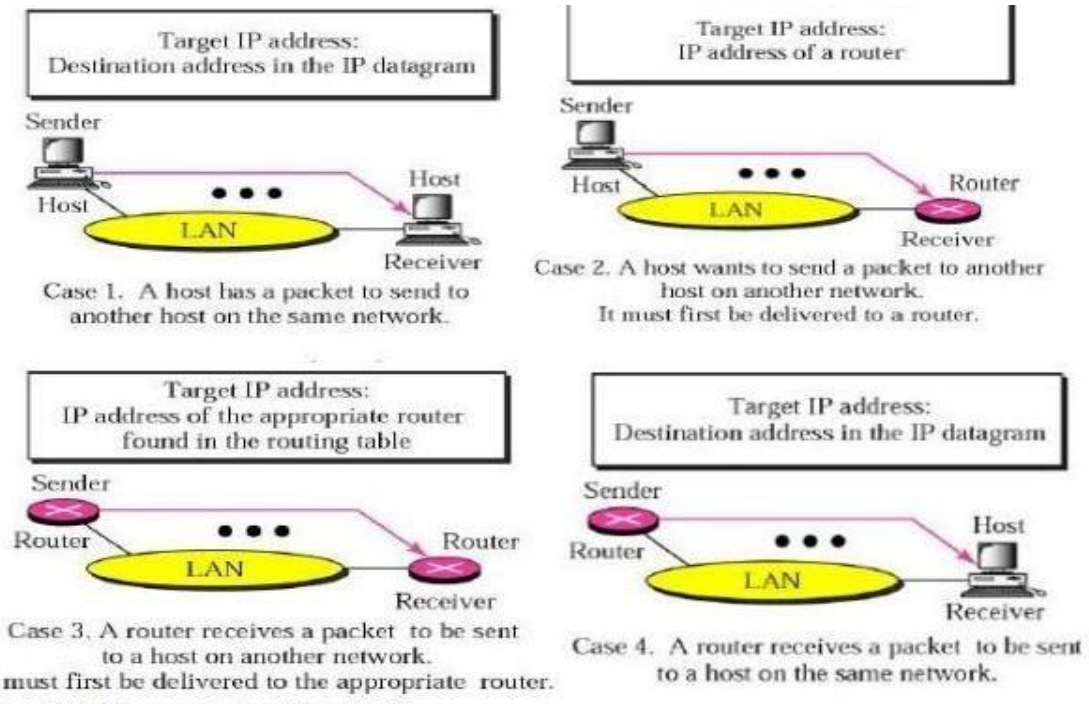


Figure 4.19 Four cases using ARP

The sender is a host and wants to send a packet to another host on the same network. In this case, the logical address that must be mapped to a physical address is the destination IP address in the datagram header.

1. The sender is a host and wants to send a packet to another host on another network. In this case, the host looks at its routing table and finds the IP address of the next hop (router) for this destination. If it does not have a routing table, it looks for the IP address of the default router. The IP address of the router becomes the logical address that must be mapped to a physical address.
2. The sender is a router that has received a datagram destined for a host on another network. It checks its routing table and finds the IP address of the next router. The IP address of the next router becomes the logical address that must be mapped to a physical address.
3. The sender is a router that has received a datagram destined for a host on the same network. The destination IP address of the datagram becomes the logical address that must be mapped to a physical address.

Example 3.17

A host with IP addresses 130.23.43.20 and physical address B2:34:55: 10:22: 10 has a packet to send to another host with IP address 130.23.43.25 and physical address A4:6E:F4:59:83:AB. The two hosts are on the same Ethernet network. Show the ARP request and reply packets encapsulated in Ethernet frames.

Solution

The ARP request and reply packets. Note that the ARP data field in this case is 28 bytes, and that the individual addresses do not fit in the 4-byte boundary. That is why we do not show the

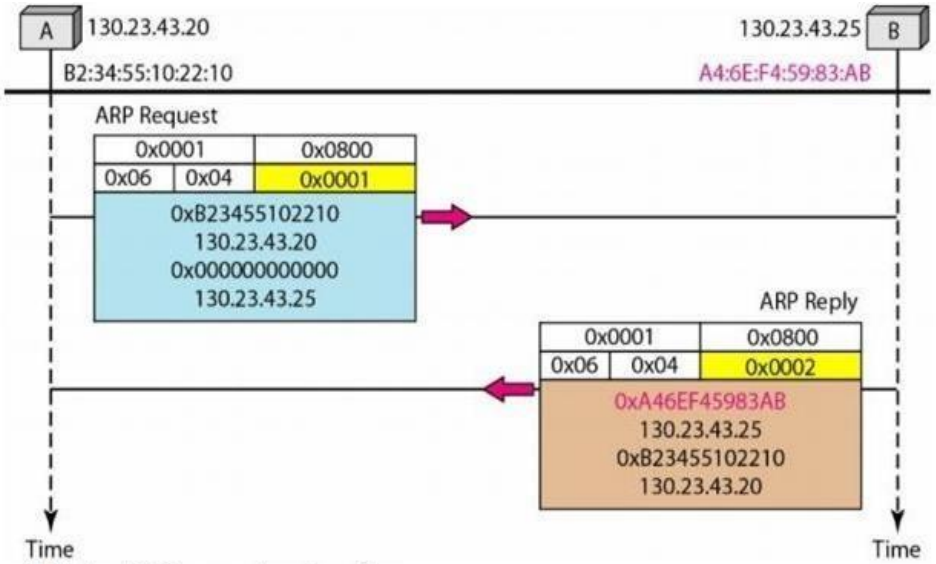


Figure 4.20 An ARP request and reply

regular 4-byte boundaries for these addresses.

4.3.3 Proxy ARP

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. Let us give an example.

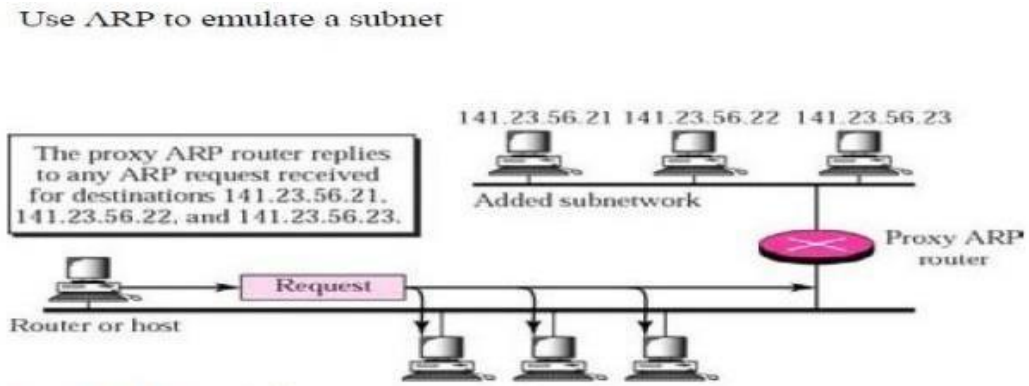


Figure 4.21 Proxy ARP

However, the administrator may need to create a subnet without changing the whole system to recognize subnetted addresses. One solution is to add a router running a proxy ARP. In this case, the router acts on behalf of all the hosts installed on the subnet. When it receives an ARP request with a target IP address that matches the address of one of its proteges (141.23.56.21, 141.23.56.22, or 141.23.56.23), it sends an ARP reply and announces its hardware address as the target hardware address. When the router receives the IP packet, it sends the packet to the appropriate host. This may happen in two cases:

44 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.

45 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.

4.4 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

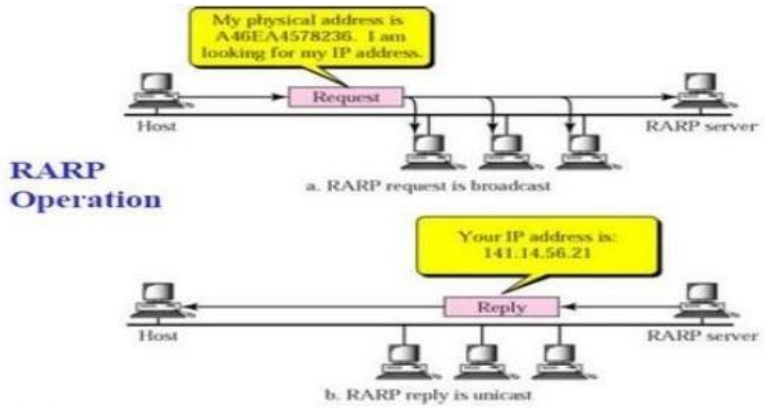
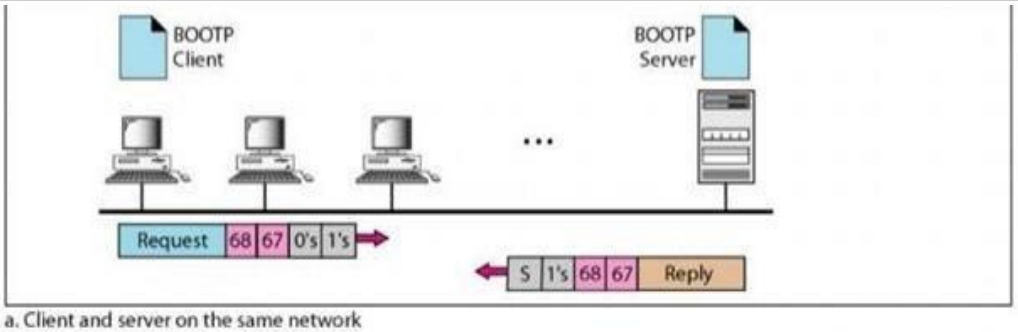


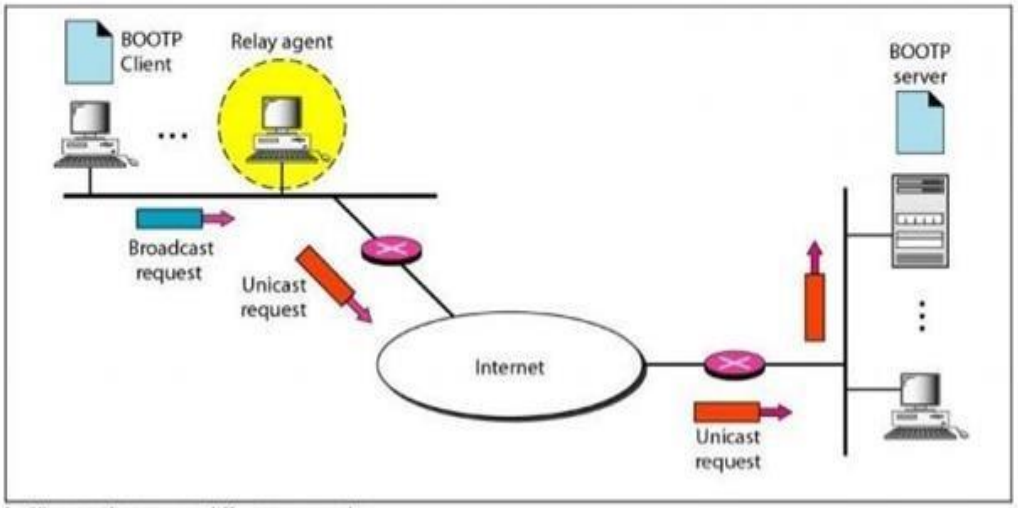
Figure 4.22 RARP Operation

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. Two protocols, BOOTP and DHCP, are replacing RARP

4.5 Bootstrap Protocol (BOOTP)



a. Client and server on the same network

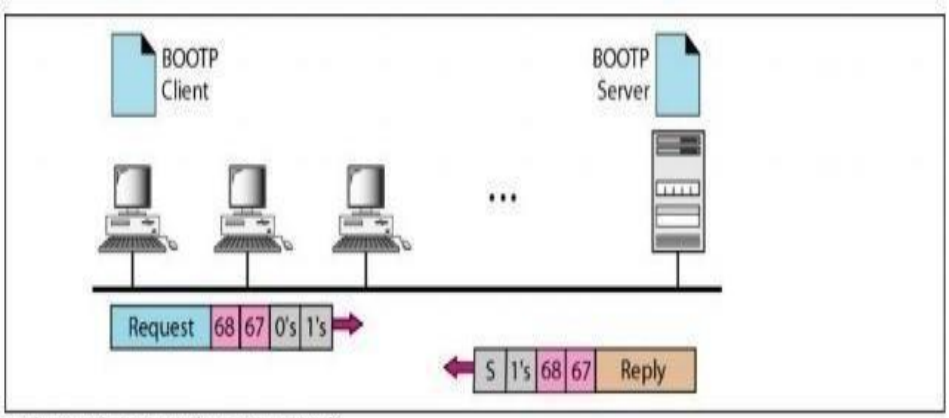


b. Client and server on different networks

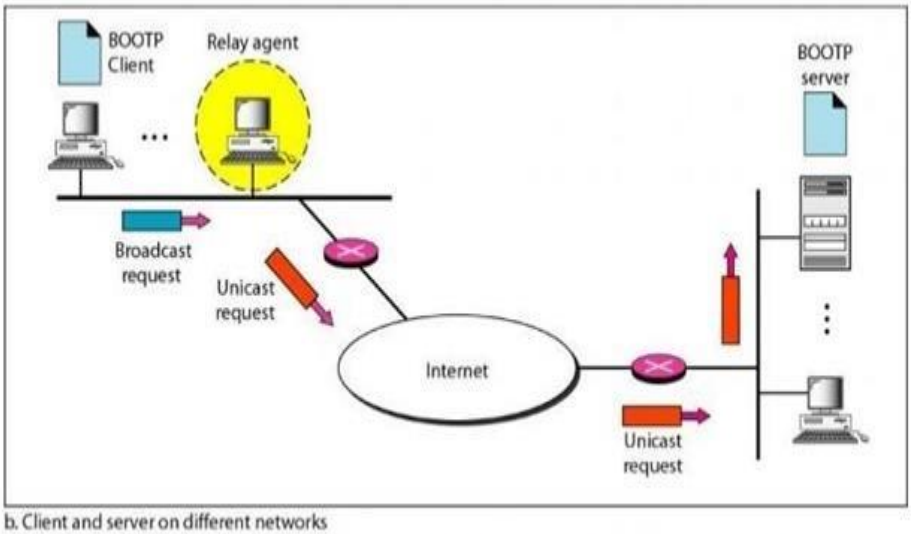
Figure 4.23 BOOTP client and server on the same and different networks

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.



a. Client and server on the same network



b. Client and server on different networks

Figure 4.23 BOOTP client and server on the same and different networks

One of the advantages of BOOTP over RARP is that the client and server are application-layer processes. As in other application-layer processes, a client can be in one network and the server in another, separated by several other networks. However, there is one problem that must be solved. The BOOTP request is broadcast because the client does not know the IP address of the server. A broadcast IP datagram cannot pass through any router. To solve the problem, there is a need for an intermediary. One of the hosts (or a router that can be configured to operate at the application layer) can be used as a relay. The host in this case is called a relay agent. The relay agent knows the unicast address of a BOOTP server. When it receives this type of packet, it encapsulates the message in a unicast datagram and sends the request to the BOOTP server. The packet, carrying a unicast destination address, is routed by any router and reaches the BOOTP server. The BOOTP server knows the message comes from a relay agent because one of the fields in the request message defines the IP address of the relay agent. The relay agent, after receiving the reply, sends it to the BOOTP client.

4.6 The Dynamic Host Configuration Protocol (DHCP)

The Dynamic Host Configuration Protocol (DHCP) has been devised to provide static and dynamic address allocation that can be manual or automatic. DHCP provides static and dynamic address allocation that can be manual or automatic. Static Address Allocation In this capacity DHCP acts as BOOTP does. It is backward compatible with BOOTP, which means a host running the BOOTP client can request a static address from a DHCP server. A DHCP server has a database that statically binds physical addresses to IP addresses.

Dynamic Address Allocation: DHCP has a second database with a pool of available IP addresses. This second database makes DHCP dynamic. When a DHCP client requests a temporary IP address, the DHCP server goes to the pool of available (unused) IP addresses and assigns an IP address for a negotiable period of time. When a DHCP client sends a request to a DHCP server, the server first checks its static database. If an entry with the requested physical address exists in the static database, the permanent IP address of the client is returned. On the other hand, if the entry does not exist in the static database, the server selects an IP address from the available pool, assigns the address to the client, and adds the entry to the dynamic database.

Manual and Automatic Configuration:

One major problem with the BOOTP protocol is that the table mapping the IP addresses to physical addresses needs to be manually configured. This means that every time there is a change in a physical or IP address, the administrator needs to manually enter the changes. DHCP, on the other hand, allows both manual and automatic configurations. Static addresses are created manually~ dynamic addresses are created automatically.

4.7 ICMP

The IP provides unreliable and connectionless datagram delivery. It was designed this way to make efficient use of network resources. The IP protocol is a best-effort delivery service that delivers a datagram from its original source to its final destination. However, it has two deficiencies: lack of error control and lack of assistance mechanisms.

The IP protocol also lacks a mechanism for host and management queries. A host sometimes needs to determine if a router or another host is alive. And sometimes a network administrator needs information from another host or router. The Internet Control Message Protocol (ICMP) has been designed to compensate for the above two deficiencies. It is a companion to the IP protocol.

4.7.1 Types of Messages

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

The query messages, which occur in pairs, help a host or a network manager get specific information from a router or another host. For example, nodes can discover their neighbors. Also, hosts can discover and learn about routers on their network, and routers can help a node redirect its messages.

4.7.2 Message Format

An ICMP message has an 8-byte header and a variable-size data section. Although the general format of the header is different for each message type, the first 4 bytes are common to all. As the first field, ICMP type, defines the type of the message. The code field specifies the reason for the particular message type. The last common field is the checksum field (to be discussed later in the chapter). The rest of the header is specific for each message type. The data section in error messages carries information for finding the original packet that had the error. In query messages, the data section carries extra information based on the type of the query.

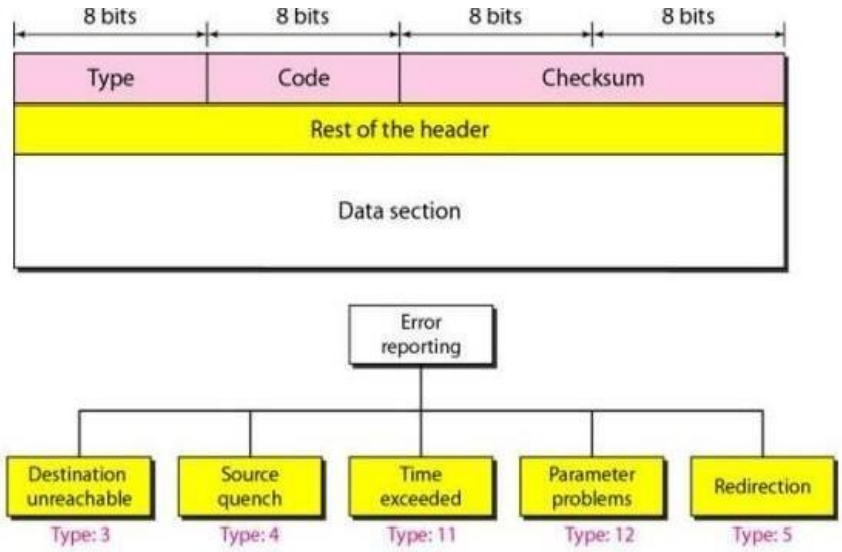


Figure 4.24 General format of ICMP messages

4.8 Unicast Routing Protocols

A routing table can be either static or dynamic. A static table is one with manual entries. A dynamic table, on the other hand, is one that is updated automatically when there is a change somewhere in the internet. Today, an internet needs dynamic routing tables. The tables need to be updated as soon as there is a change in the internet. For instance, they need to be updated when a router is down, and they need to be updated whenever a better route has been found.

Optimization

A router receives a packet from a network and passes it to another network. A router is usually attached to several networks. One approach is to assign a cost for passing through a network. We call this cost a metric. However, the metric

assigned to each network depends on the type of protocol. Some simple protocols, such as the Routing Information Protocol (RIP), treat all networks as equals. The cost of passing through a network is the same; it is one hop count. So if a packet passes through 10 networks to reach the destination, the total cost is 10 hop counts.

4.8.1 Intra- and Inter-domain Routing

An internet can be so large that one routing protocol cannot handle the task of updating the routing tables of all routers. For this reason, an internet is divided into autonomous systems. An autonomous system (AS) is a group of networks and routers under the authority of a single administration. Routing inside an autonomous system is referred to as intradomain routing. Routing between autonomous systems is referred to as interdomain routing.

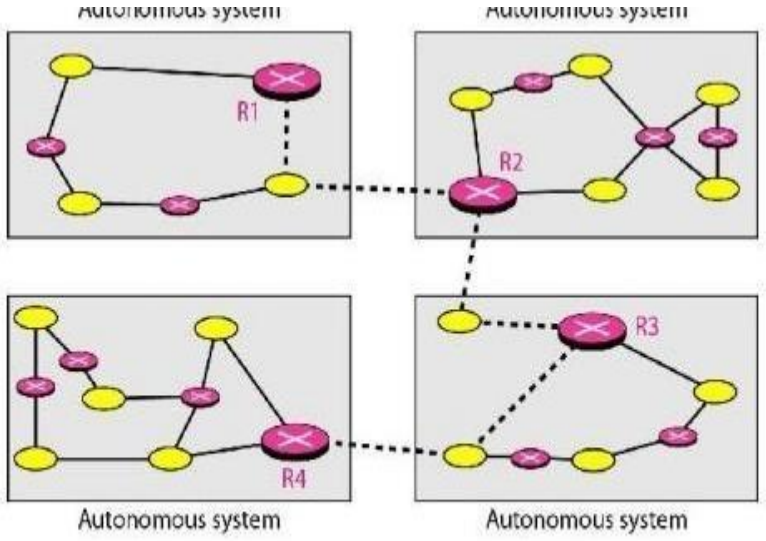


Figure 4.25 Autonomous systems

Several intradomain and interdomain routing protocols are in use.

O Two intradomain routing protocols: Distance vector and link state.

O One interdomain routing protocol: path vector.

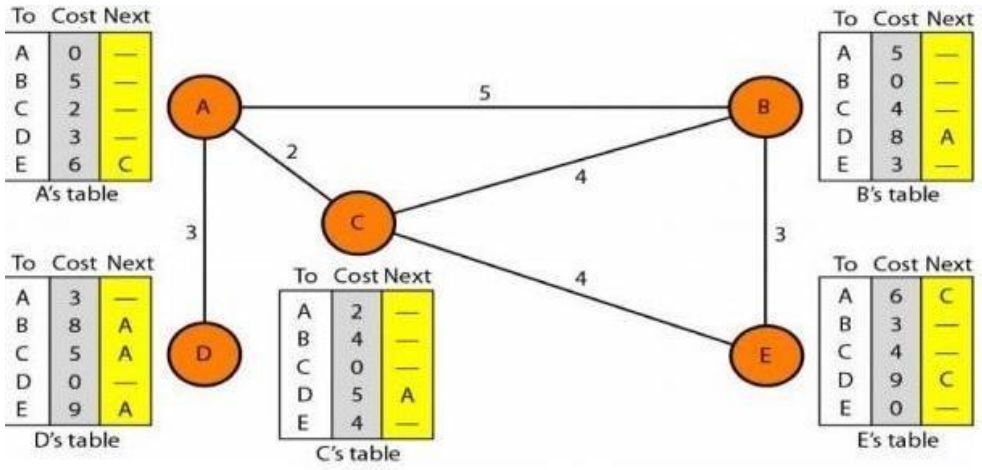


Figure . 4.27 Distance vector routing tables

Routing Information Protocol (RIP) is an implementation of the distance vector protocol. Open Shortest Path First (OSPF) is an implementation of the link state protocol. Border Gateway Protocol (BGP) is an implementation of the path vector protocol.

4.8.2 Distance Vector Routing

In distance vector routing, the least-cost route between any two nodes is the route with minimum distance. In this protocol, as the name implies, each node maintains a vector (table) of minimum distances to every node. The table at each node also guides the packets to the desired node by showing the next stop in the route (next- hop routing).

The table for node A shows how we can reach any node from this node. For example, our least cost to reach node E is 6. The route passes through C.

Initialization

The tables in Figure 3.45 are stable; each node knows how to reach any other node and the cost. At the beginning, however, this is not the case. Each node can know only the distance between itself and its immediate neighbors, those directly connected to it. So for the moment, we assume that each node can send a message to the immediate neighbors and find the distance between itself and these neighbors. The distance for any entry that is not a neighbor is marked as infinite (unreachable).

Sharing

The whole idea of distance vector routing is the sharing of information between neighbors. Although node A does not know about node E, node C does. So if node C shares its routing table with A, node A can also know how to reach node E. On the other hand, node C does not know how to reach node D, but node A does. If node A shares its routing table with node C, node C also knows how to reach node

D. In other words, nodes A and C, as immediate neighbors, can improve their routing tables if they help each other.

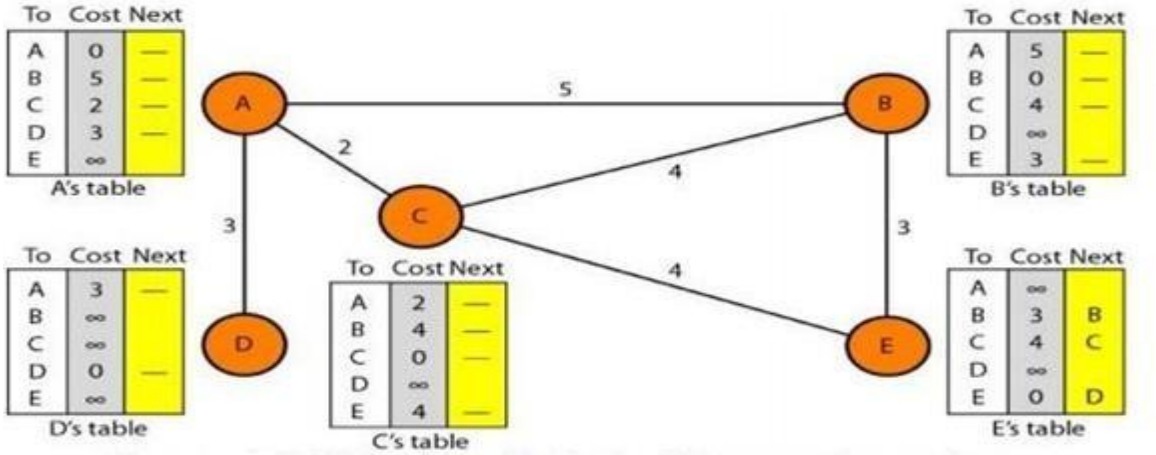


Figure 4.29 Initialization of tables in distance vector routing

Updating

When a node receives a two-column table from a neighbor, it needs to update its routing table. Updating takes three steps:

1. The receiving node needs to add the cost between itself and the sending node to each value in the second column. The logic is clear. If node C claims that its distance to a destination is *x* mi, and the distance between A and C is *y* mi, then the distance between A and that destination, via C, is *x* + *y*mi.

2. The receiving node needs to add the name of the sending node to each row as the third column if the receiving node uses information from any row. The sending node is the next node in the route.

3. The receiving node needs to compare each row of its old table with the corresponding row of the modified version of the received table.

a) If the next-node entry is different, the receiving node chooses the row with the smaller cost. If there is a tie, the old one is kept.

b) b) If the next-node entry is the same, the receiving node chooses the new row. For example, suppose node C has previously advertised a route to node X with distance 3.

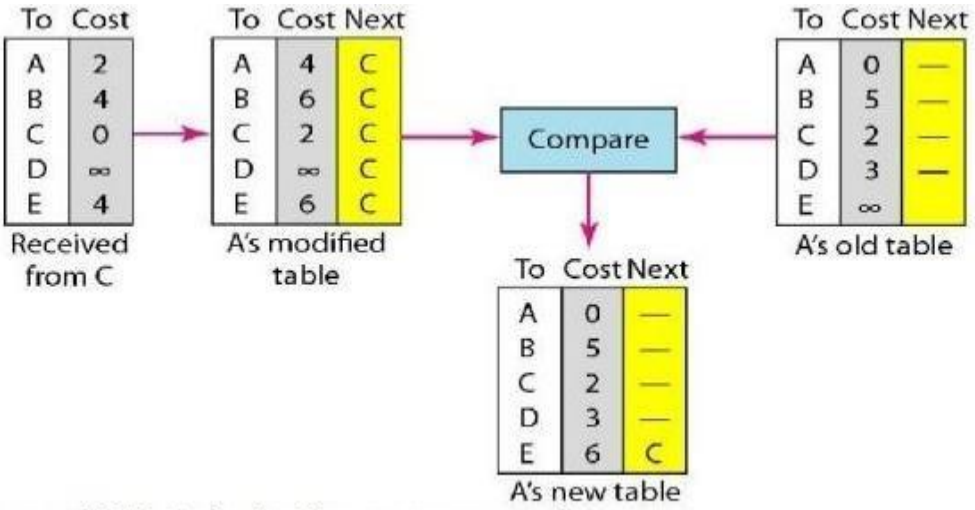


Figure 4.30 Updating in distance vector routing

Two-Node Loop Instability

A problem with distance vector routing is instability, which means that a network using this protocol can become unstable. To understand the problem, let us look at the scenario depicted.

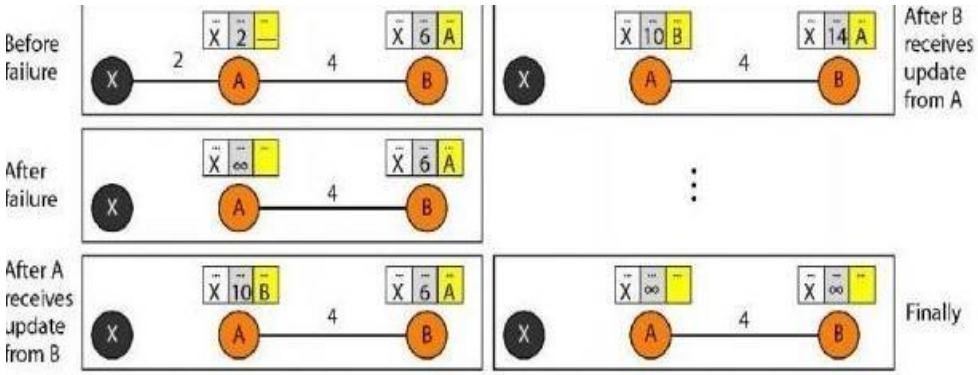


Figure 4.31 Two-node instability

Defining Infinity The first obvious solution is to redefine infinity to a smaller number, such as100. For our previous scenario, the system will be stable in less than 20 update s. As a matter of fact, most implementations of the distance vector protocol define the distance between each node to be I and define 16 as infinity. However, this means that the distance vector routing cannot be used in large systems. The size of the network, in each direction, cannot exceed 15 hops.

Split Horizon Another solution is called split horizon. In this strategy, instead of flooding thetable through each interface, each node sends only part of its table through each interface. If, according to its table, node B thinks that the optimum

route to reach X is via A, it does not need to advertise this piece of information to A; the information has corne from A (A already knows). Taking information from node A, modifying it, and sending it back to node A creates the confusion. In our scenario, node B eliminates the last line of its routing table before it sends it to A. In this case, node A keeps the value of infinity as the distance to X.

4.8.3 Link State Routing

Link state routing has a different philosophy from that of distance vector routing. In link state routing, if each node in the domain has the entire topology of the domain the list of nodes and links, how they are connected including the type, cost (metric), and condition of the links (up or down)-the node can use Dijkstra's algorithm to build a routing table.

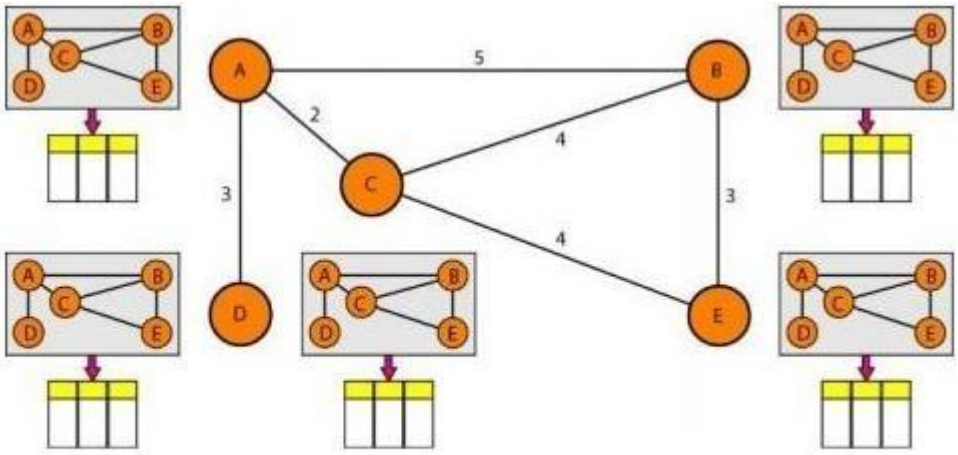


Figure 4.32 Concept of link state routing

The figure shows a simple domain with five nodes. Each node uses the same topology to create a routing table, but the routing table for each node is unique because the calculations are based on different interpretations of the topology. This is analogous to a city map. While each person may have the same map, each needs to take a different route to reach her specific destination

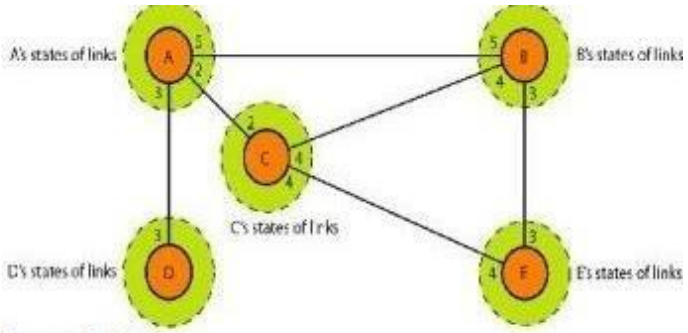


Figure 4.33 Link state knowledge

4.8.4 Building Routing Tables:

In link state routing, four sets of actions are required to ensure that each node has therouting table showing the least-cost node to every other node.

a) Creation of the states of the links by each node, called the link state packet (LSP).

b) Dissemination of LSPs to every other router, called **flooding,** in an efficient and reliable way.

c) Formation of a shortest path tree for each node.

d) Calculation of a routing table based on the shortest path tree.

Types of Links

In OSPF terminology, a connection is called a *link.* Four types of links have been defined: point-to-point, transient, stub, and virtual.

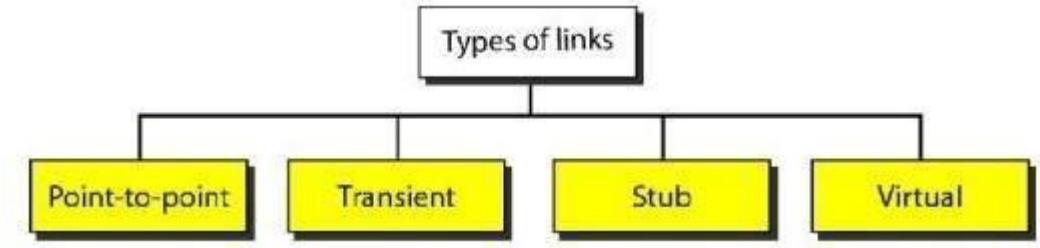


Figure 4.34 Types of links

A point-to-point link connects two routers without any other host or router in between. In other words, the purpose of the link (network) is just to connect the two routers. An example of this type of link is two routers connected by a telephone line or a T line. There is no need to assign a network address to this type of link. Graphically, the routers are represented by nodes, and the link is represented by a bidirectional edge connecting the nodes. The metrics, which are usually the same, are shown at the two ends, one for each direction. In other words, each router has only one neighbor at the other side of the link.

4.8.5 Path Vector Routing

Distance vector and link state routing are both intradomain routing protocols. They can be used inside an autonomous system, but not between autonomous systems. These two protocols are not suitable for interdomain routing mostly because of scalability. Both of these routing protocols become intractable when the domain of operation becomes large. Distance vector routing is subject to instability if there are more than a few hops in the domain of operation. Link state routing needs a huge amount of resources to calculate routing tables. It also creates heavy traffic because of flooding. There is a need for a third routing protocol which we call path vector routing.

Path vector routing proved to be useful for interdomain routing. The principle of path vector routing is similar to that of distance vector routing. In path vector routing, we assume that there is one node in each autonomous system that acts on behalf of the entire autonomous system.

Initialization

At the beginning, each speaker node can know only the reach ability of nodes inside its autonomous system

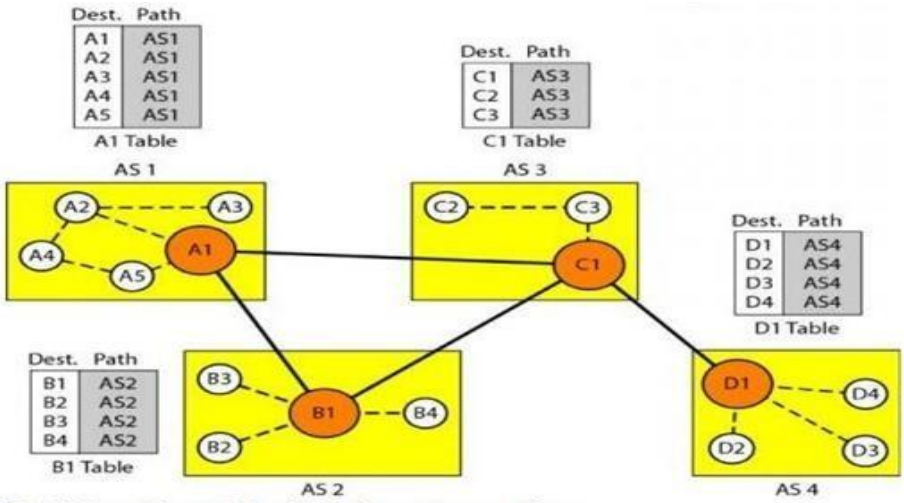


Figure 4.35 Initial routing tables in path vector routing

Node Al is the speaker node for AS1, B1 for AS2, C1 for AS3, and Dl for AS4. Node Al creates an initial table that shows Al to A5 are located in ASI and can be reached through it. Node B1 advertises that Bl to B4 are located in AS2 and can be reached through Bl. And so on.

UNIT-V Transport Layer and Application Layer

5.1 Process to Process Delivery

- The **data link layer** is responsible for delivery of frames between two neighboring nodes over a link. This is called **node-to-node delivery**.
- The **network layer** is responsible for delivery of datagrams between two hosts. This is **called host-to-host delivery**.
- Real communication takes place between two processes (application programs). We need **process-to-process delivery**. The transport layer is responsible for process-to-process delivery-the delivery of a packet, part of a message, from one process to another.

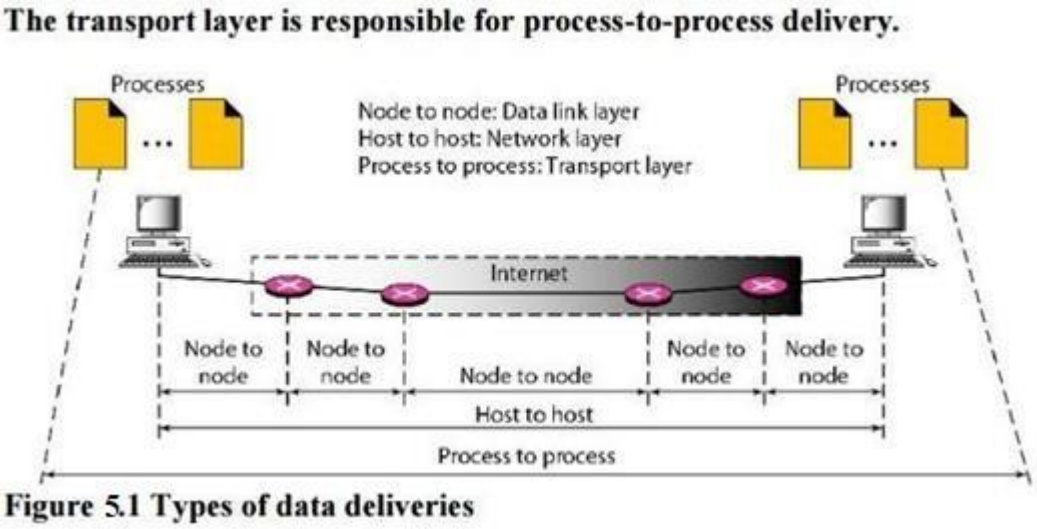


Figure 5.1 Types of data deliveries

Figure 5.1 shows these three types of deliveries and their domains

5.1.1 Client/Server Paradigm

Although there are several ways to achieve process-to-process communication, the most common one is through the client/server paradigm. A process on the **local host, called a client**, needs services from a process usually on the **remote host, called a server**. Both processes (client and server) have the same name. For example, to get the day and time from a remote machine, we need a Daytime client process running on the local host and a Daytime server process running on a

remote machine. For communication, we must define the following:

1. Local host
2. Local process
3. Remote host
4. Remote process

5.1.2 Addressing

Whenever we need to **deliver something to one specific destination among many, we need an address**. At the **data link layer, we need a MAC address** to choose one node among several nodes if the connection is not point-to-point. A frame in the data link layer needs a Destination MAC address for delivery and a source address for the next node's reply.

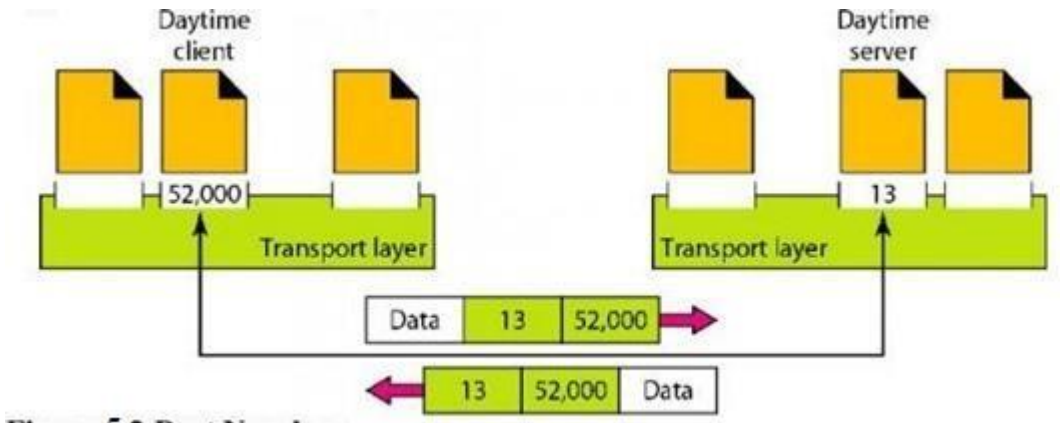


Figure 5.2 Port Numbers

The IP addresses and port numbers play different roles in selecting the final destination of data. The destination IP address defines the host among the different hosts in the world. After the host has been selected, the port number defines one of the processes on this particular host (see Figure 4.3).

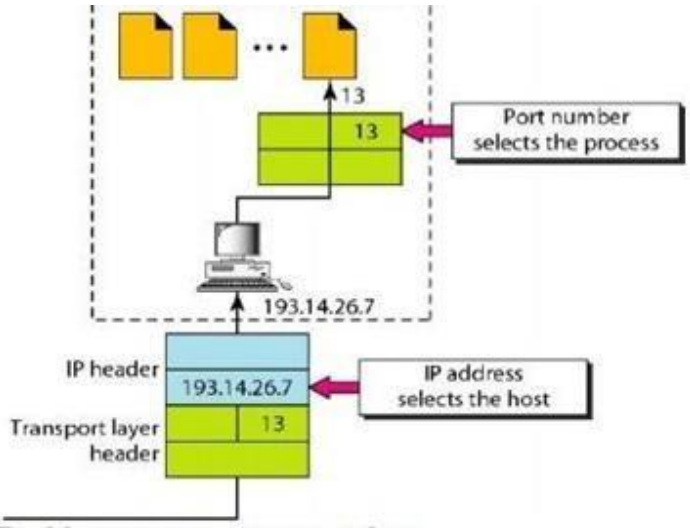


Figure5 .3 IP addresses versus port numbers

5.1.3 lANA Ranges

The lANA (Internet Assigned Number Authority) has divided the port numbers into three ranges: well known, registered, and dynamic (or private), as shown in Figure 4.4.

- **Well-known ports**. The ports ranging from 0 to 1023 are assigned and **controlled** by lANA. These are the well-known ports.

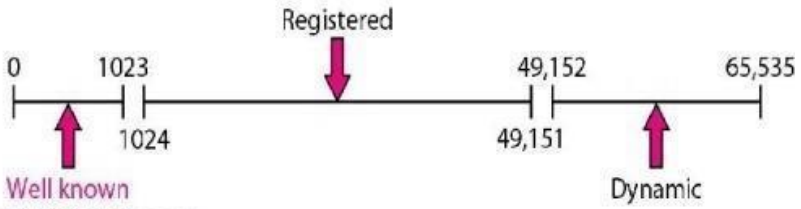


Figure 4.4 lANA Ranges

- **Registered ports.** The ports ranging from 1024 to 49,151 are **not assigned or controlled** by lANA. They can only be registered with lANA to prevent duplication.
- **Dynamic ports**. The ports ranging from 49,152 to 65,535 are n**either controlled nor registered**. They can be used by any process. These are the ephemeral ports.

5.1.4 Socket Addresses

Process-to-process delivery needs two identifiers, IP address and the port number, at each end to make a connection. The **combination of an IP address and a port number is called a socket address.** The **client socket address defines the client process** uniquely just as **the server socket address defines the server process** uniquely (see Figure 4.5).

UDP or TCP header contains the port numbers.

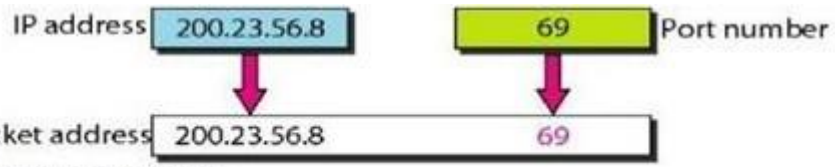


Figure 5.5 Socket Address

5.1.5 Multiplexing and Demultiplexing

The addressing mechanism allows multiplexing and demultiplexing by the transport layer, as shown in Figure 4.6.

Multiplexing

At the sender site, there may be several processes that need to send packets. However, there is only

one transport layer protocol at any time. This is a **many-to-one** relationship and requires multiplexing.

Demultiplexing

At the receiver site, the relationship is **one-to-many** and requires demultiplexing. The transport layer receives datagrams from the network layer. After error checking and dropping of the header, the transport layer delivers each message to the appropriate process based on the port number.

5.1.6 Connectionless Versus Connection-Oriented Service

A transport layer protocol can either be connectionless or connection-oriented.

Connectionless Service

In a connectionless service, **the packets are sent from one party to another with no need for connection establishment or connection release.** The packets are not numbered; they may be delayed or lost or may arrive out of sequence. There is **no acknowledgment** either.

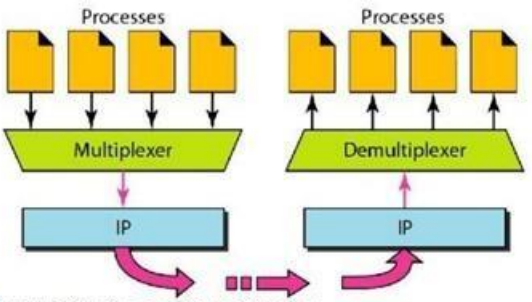


Figure5 .6 Multiplexing and Demultiplexing

Connection~Oriented Service

In a connection-oriented service, a connection is **first established** between the sender and the receiver. Data are transferred. At the end, the connection is released.

5.1.7Reliable Versus Unreliable

- The transport layer service can be **reliable or unreliable**. If the application layer program needs reliability, we use a reliable transport layer protocol by implementing **flow and error control at the transport layer**. This means a slower and more complex service.
- In the Internet, there are three common different transport layer protocols. **UDP is connectionless and unreliable; TCP and SCTP are connection oriented and reliable.** These three can respond to the demands of the application layer programs.
- The **network layer in the Internet is unreliable** (best-effort delivery), we need to

implement **reliability at the transport layer**. To understand that error control at the data link layer does not guarantee error control at the transport layer, let us look at Figure 4.7.

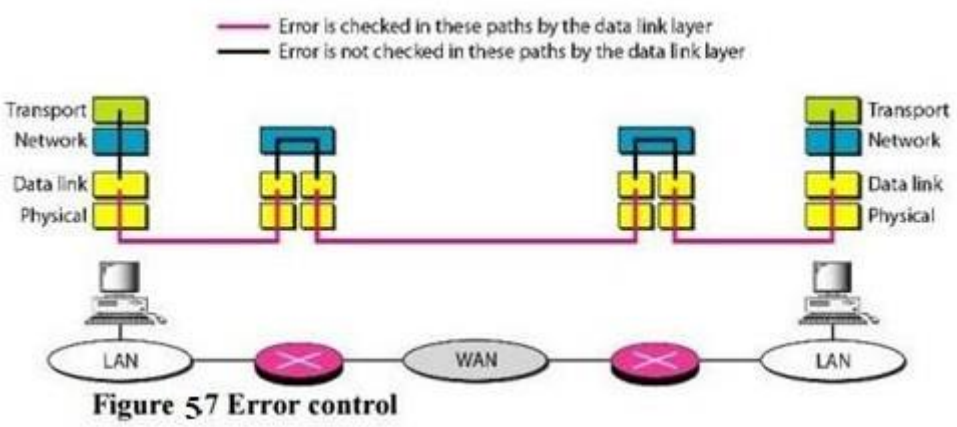


Figure 5.7 Error control

Three Protocols

The original TCP/IP protocol suite specifies two protocols for the transport layer: UDP and TCP. We first focus on UDP, the simpler of the two, before discussing TCP. A new transport layer protocol, SCTP, has been designed. Figure 4.8 shows the position of these protocols in the TCP/IP protocol suite.

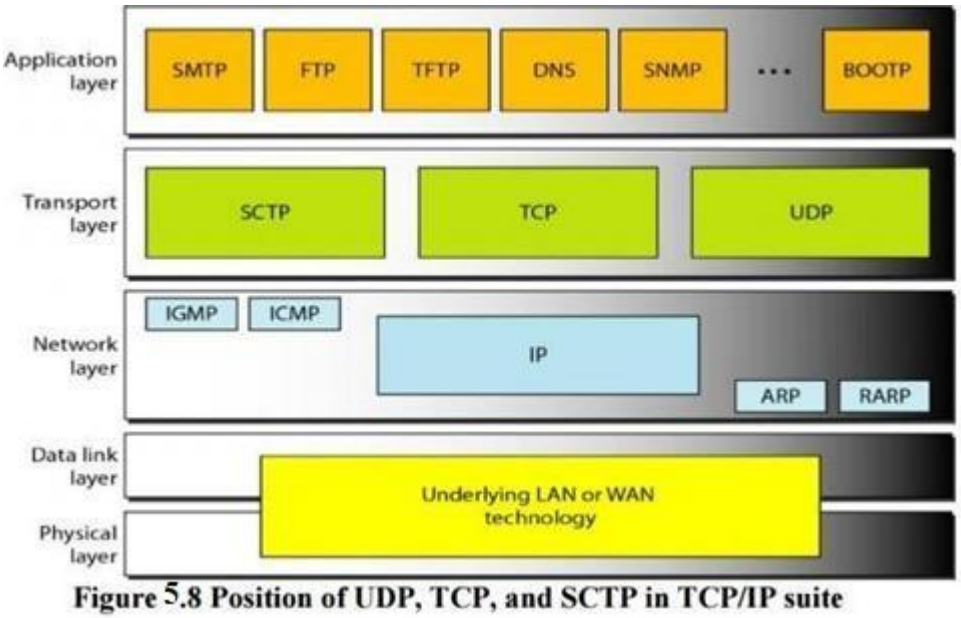


Figure 5.8 Position of UDP, TCP, and SCTP in TCP/IP suite

5.2 USER DATAGRAM PROTOCOL (UDP)

- The User Datagram Protocol (UDP) is called a **connectionless, unreliable transport**

protocol. It does not add anything to the services of IP except to provide process-to process communication instead of host-to-host communication. Also, it performs very **limited error checking**.

- UDP is a very **simple protocol** using a minimum of overhead. If a process wants to send a small message and does not care much about reliability, it can use UDP. Sending a small message by using UDP takes much **less interaction** between the sender and receiver than using TCP or SCTP.

1. Well-Known Ports for UDP:

Table 4.1 shows some well-known port numbers used by UDP. Some port numbers can be used by both UDP and TCP1

Table 4.1 Well-known port numbers used by UDP

Port	Protocol	Description
1	Echo	Echoes a received datagram back to the sender
9	Discard	Discards any datagram that is received
11	Users	Active users
13	Daytime	Returns the date and the time
17	Quote	Returns a quote of the day
19	Chargen	Returns a string of characters
53	Name server	Domain Name Service
67	BOOTPs	Server port to download bootstrap information
68	BOOTPc	Client port to download bootstrap information
69	TFTP	Trivial File Transfer Protocol
111	RPC	Remote Procedure Call
123	NTP	Network Time Protocol
161	SNMP	Simple Network Management Protocol
162	SNMP	Simple Network Management Protocol (trap)

5.2.2 User Datagram

UDP packets, called user datagram's have a fixed-size header of 8 bytes. Figure 4.9 shows the format of a user datagram.

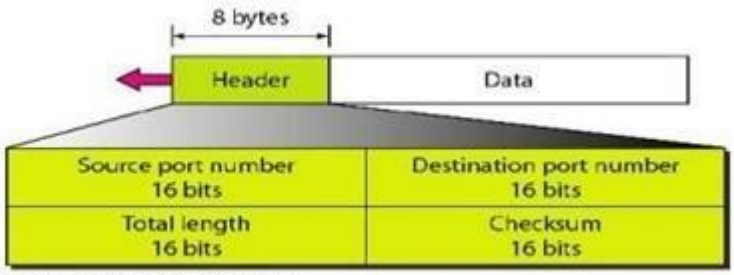


Figure5 .9 User datagram format

The fields are as follows:

- **Source port number**. This is **the port number used by** the process running on the **source host**. It is 16 bits long, which means that the port number can range from 0 to 65,535.
- **Destination port number**. This is the **port number used** by the process running on the d**estination** host. It is also 16 bits long.
- **Length**. This is a **16-bit field** that defines the total length of the user datagram, header plus data. The 16 bits can define a total length of 0 to 65,535 bytes.

UDP length = IP length - IP header's length

- **Checksum**. This field is used to **detect errors** over the entire user datagram (header plus data).

5.2.3 Checksum

The UDP checksum calculation is different from the one for IP and ICMP. Here the checksum includes three sections: a **pseudo header, the UDP header, and the data coming from the application layer**

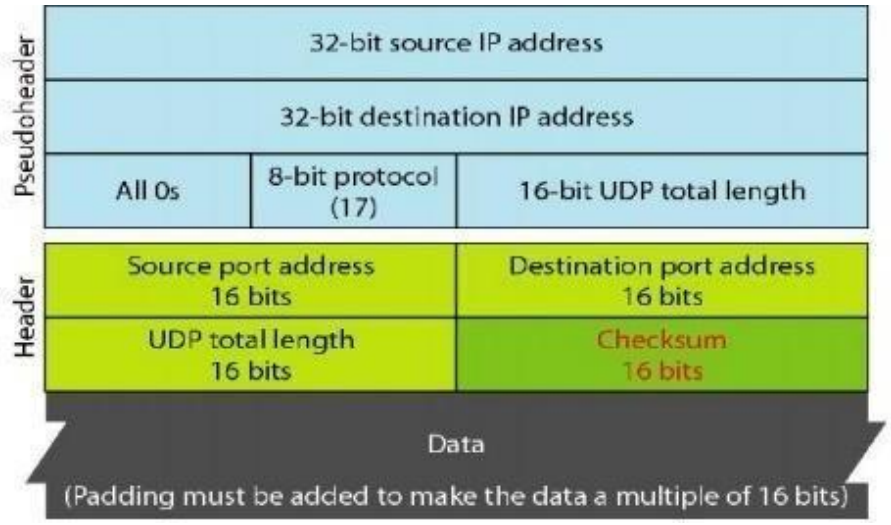


Figure 5.10 Pseudoheader for checksum calculation

The **pseudoheader** is the part of the header of the IP packet in which the user datagram is to be **encapsulated w**ith some fields filled with **Os** (see Figure 4.10).

Example 5.1

Figure 4.11 shows the checksum calculation for a very small user datagram with only 7 bytes of data. Because the number of bytes of data is odd, padding is added for checksum calculation. The pseudoheader as well as the padding will be dropped when the user datagram is delivered to IP.

Optional use of the checksum:

The calculation of the checksum and its inclusion in a user datagram are optional. If the checksum is not calculated, the field is filled with 1s. Note that a calculated checksum can never be all 1s because this implies that the sum is all 0s, which is impossible because it requires that the value of fields to be 0s.

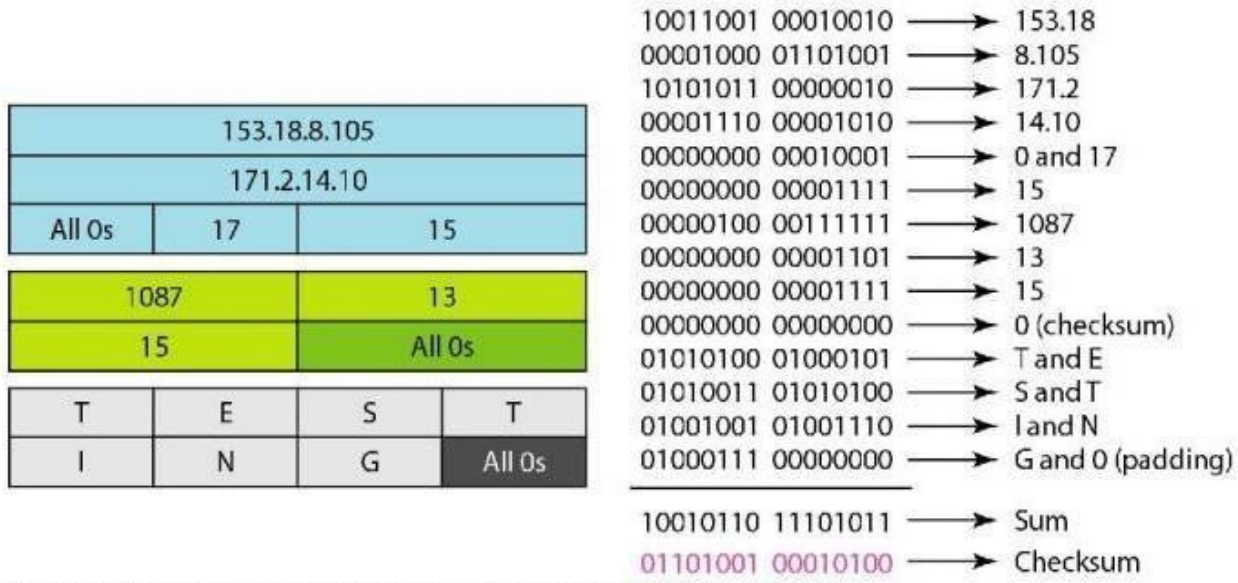


Figure 4.11 Checksum calculation of a simple UDP user datagram

5.2.4 UDP Operation:

UDP uses concepts common to the transport layer.

Connectionless Services

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

Flow and Error Control

UDP is a very **simple, unreliable** transport protocol. There is no flow control and hence no window mechanism. The receiver may **overflow** with incoming messages. There is **no error control mechanism in UDP except for the checksum**. This means that the sender does not know if a message has been **lost or duplicated**. When the receiver detects an error through the checksum, the user datagram is silently **discarded**. The lack of flow control and error control means that the process using UDP should provide these mechanisms.

Encapsulation and Decapsulation

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

messages in an IP datagram.

Queuing

In UDP, queues are associated with ports. (Figure 4.12).

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

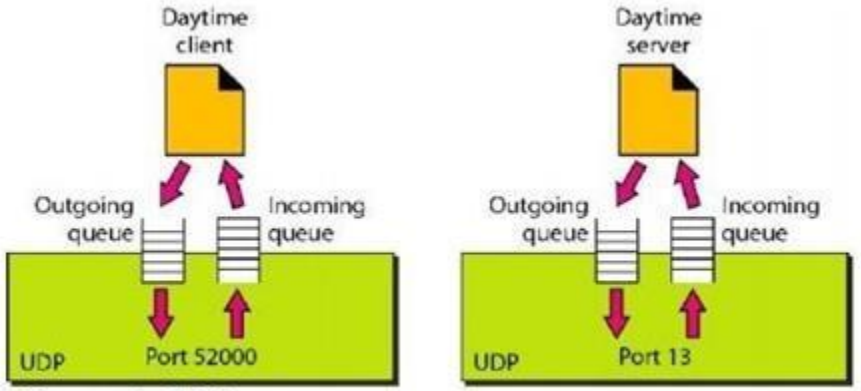


Figure5 .12 Queues in UDP

Note that even if a process wants to communicate with multiple processes, it obtains only one port number and eventually one outgoing and one incoming queue. The queues opened by the client are, in most cases, identified by ephemeral port numbers. The queues function as long as the process is running. When the **process terminates, the queues are destroyed.**

The client process can send messages to the outgoing queue by using the source port number specified in the request. UDP removes the messages one by one and, after adding the UDP header, delivers them to IP. An outgoing queue can overflow. If this happens, the operating system can ask the client process to wait before sending any more messages.

5.2.5 Use of UDP

The following lists some uses of the UDP protocol:

5.3 UDP is suitable for a process that requires simple request-response communication with little concern for flow and error control. It is not usually used for a process such as FTP that needs

to send bulk data.

5.4 UDP is suitable for a process with internal flow and error control mechanisms. For example, the Trivial File Transfer Protocol (TFTP) process includes flow and error control. It can easily use UDP.

5.5 UDP is a suitable transport protocol for multicasting. Multicasting capability is embedded in the UDP software but not in the TCP software.

5.6 UDP is used for management processes such as SNMP.

5.7 UDP is used for some route updating protocols such as Routing Information Protocol (RIP).

5.3 Transmission Control Protocol (TCP)

TCP is a process-to-process (program-to-program) protocol. TCP is called a **connection-oriented, reliable transport protocol**. TCP uses flow and error control mechanisms at the transport level.

5.3.1 TCP Services

The services offered by TCP to the processes at the application layer.

Process-to-Process Communication

TCP provides process-to-process communication using port numbers. Table 4.2 lists some well-known port numbers used by TCP.

Table 4.2 Well-Known Port used by TCP

Port	Protocol	Description
1	Echo	Echoes a received datagram back to the sender
9	Discard	Discards any datagram that is received
11	Users	Active users
13	Daytime	Returns the date and the time
17	Quote	Returns a quote of the day
19	Chargen	Returns a string of characters

20	FTP, Data	File transfer protocol (data connection)
21	FTP, Control	File transfer protocol (control connection)
23	TELNET	Terminal network
25	SMTP	Simple Mail Transfer protocol
53	DNS	Domain name server
67	BOOTP	Bootsrap protocol
79	Finger	Finger
80	HTTP	Hypertext transfer protocol
111	RPC	Remote procedure Call

5.3.2 Stream Delivery Service

TCP is a **stream-oriented protocol**. TCP allows the sending process to deliver data as a **stream of bytes** and allows the receiving process to obtain data as a stream of bytes. TCP creates an environment in which the two processes seem to be connected by an imaginary "tube" that carries their data across the Internet. This imaginary environment is depicted in Figure 4.13. The sending process produces (writes to) the stream of bytes, and the receiving process consumes (reads from) them.

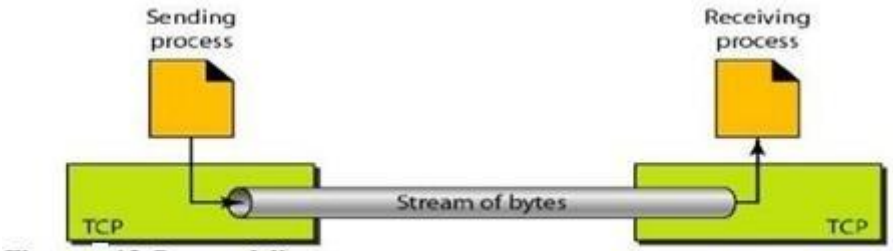


Figure 5.13 ***Stream delivery***

Sending and Receiving Buffers:

Because the sending and the receiving processes may not write or read data at the same speed, TCP needs **buffers for storage**. There are two buffers, the **sending buffer and the receiving buffer, one for each direction**. One way to implement a buffer is to use a circular array of I-byte locations as shown in Figure 4.14. For simplicity, we have shown two buffers of 20 bytes each; normally the buffers are hundreds or thousands of bytes, depending on the implementation. We also show the buffers as the same size, which is not always the case.

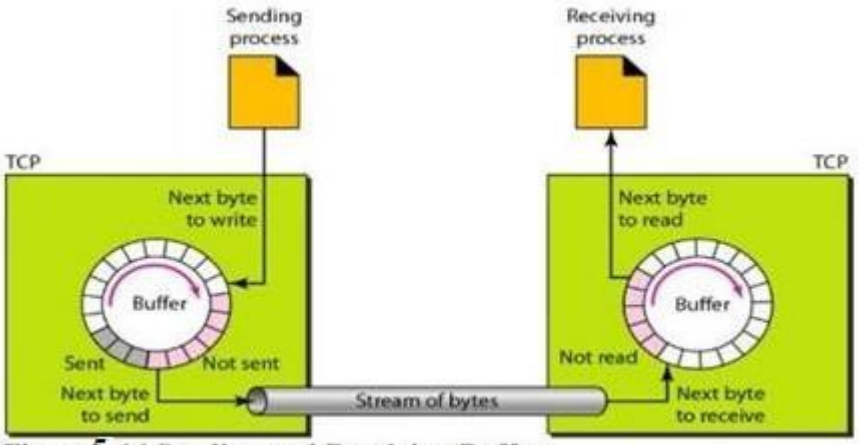


Figure 5.14 Sending and Receiving Buffers

Figure 4.14 shows the movement of the data in one direction. At the sending site, the buffer has three types of chambers. The white section contains empty chambers that can be filled by the sending process (producer). The gray area holds bytes that have been sent but not yet acknowledged. TCP keeps these bytes in the buffer until it receives an acknowledgment. The colored area contains bytes to be sent by the sending TCP.

5.3.4 Segments

Segments although buffering handles the disparity between the speeds of the producing and consuming processes, we need one more step before we can send data. **The IP layer, as a service provider for TCP, needs to send data in packets, not as a stream of bytes. At the transport layer, TCP groups a number of bytes together into a packet called a segment**. Figure 4.15 show segments are created from the bytes in the buffers.

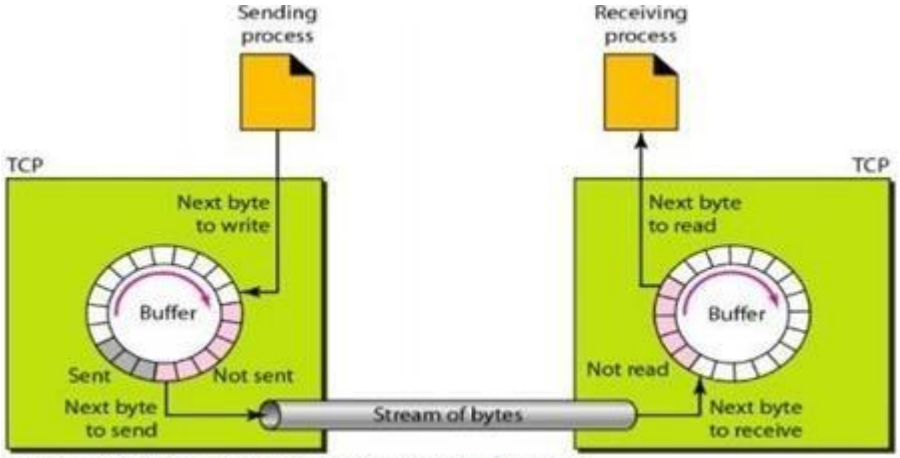


Figure 5.14 Sending and Receiving Buffers

Full-Duplex Communication

TCP offers full-duplex service, in which data can **flow in both directions at the same time**. Each TCP then has a sending and receiving buffer, and segments move in both directions.

5.3.5 Connection-Oriented Service

TCP is a connection-oriented protocol. When a process at site A wants to send and receive data from another process at site B, the following occurs:

The two TCPs establish a connection between them.

Data are exchanged in both directions.

The connection is terminated.

Reliable Service

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

TCP Features

TCP has several features.

Numbering System

Although the TCP software keeps track of the segments being transmitted or received, there is no field for a segment number value in the segment header. Instead, there are two fields called the **sequence number and the acknowledgment number**. These two fields refer to the byte number and not the segment number.

Byte Number

TCP numbers all data bytes that are transmitted in a connection. Numbering is independent in each direction. When TCP receives bytes of data from a process, it stores them in the sending buffer and numbers them.

The bytes of data being transferred in each connection are numbered by TCP. The numbering starts with a randomly generated number.

Sequence Number

After the bytes have been numbered, TCP **assigns a sequence number to each segment** that is being sent. The sequence number for each segment is the number of the first byte carried in that segment.

The value in the sequence number field of a segment defines the number of the first data byte contained in that segment.

When a segment carries **a combination of data and control** information (piggybacking), it uses a sequence number. If a segment does not carry user data, it does not logically define a sequence number. The field is there, but the value is not valid. However, some segments, when carrying **only control information**, need a sequence number to allow an acknowledgment from the receiver. These segments are used for **connection establishment, termination, or abortion**.

Acknowledgment Number

Communication in TCP is full duplex; **when a connection is established, both parties can send and receive data at the same time.** Each party numbers the bytes, usually with a different starting byte number.

The sequence number in each direction shows the number of the first byte carried by the segment. Each party also uses an acknowledgment number to confirm the bytes it has received. However, the acknowledgment number defines the number of the next byte that the party expects

to receive. In addition, the acknowledgment number is cumulative, which means that the party takes the number of the last byte that it has received, safe and sound, adds I to it, and announces this sum as the acknowledgment number.

Flow Control

TCP provides flow control. **The receiver of the data controls the amount of data that are to be sent by the sender.** This is done to **prevent the receiver from being overwhelmed** with data. The numbering system allows TCP to use a byte-oriented flow control.

Error Control

To provide reliable service, TCP implements **an error control mechanism**. Although error control considers a segment as the unit of data for error detection (loss or corrupted segments), error control is byte-oriented.

Congestion Control

TCP, unlike UDP, takes into account congestion in the network. The amount of data sent by a sender is not only controlled by the receiver (flow control), but is also **determined by the level of congestion in the network.**

Segment

A packet in TCP is called a segment.

Format

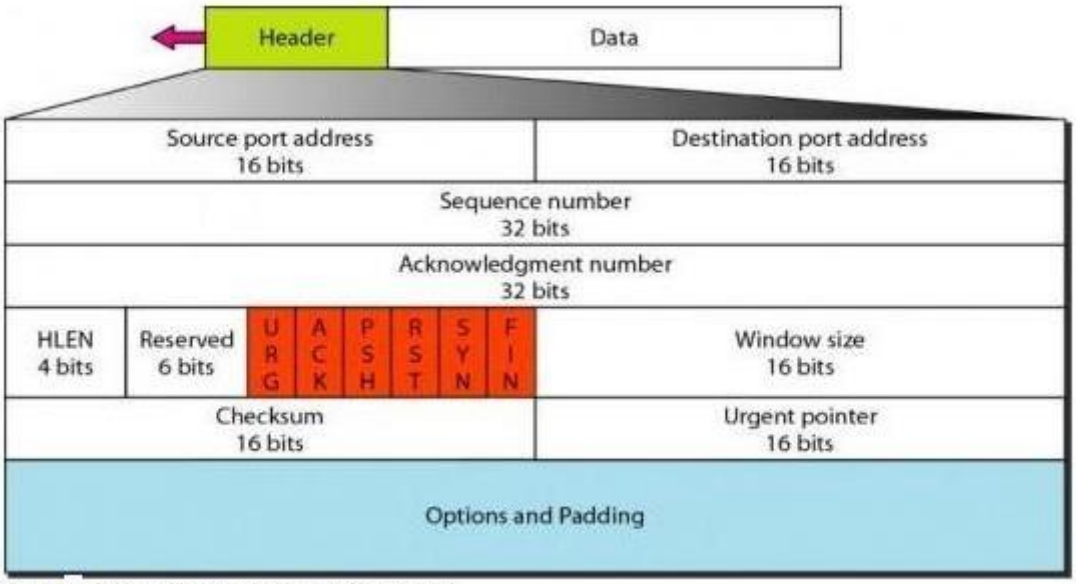


Figure 5.16 TCP Segment format

The format of a segment is shown in Figure 4.16

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

- **Source port address**. This is a 16-bit field that defines the port number of the application program in the host that is **sending the segment**. This serves the same purpose as the source port address in the UDP header.
- **Destination port address**. This is a 16-bit field that defines the port number of the application program in the host that is **receiving the segment**. This serves the same purpose as the destination port address in the UDP header.
- **Sequence number**. This 32-bit field defines the number assigned to the first byte of data contained in this segment. As we said before, TCP is a stream transport protocol. To ensure connectivity, each byte to be transmitted is numbered. The **sequence number tells the destination which byte in this sequence comprises the first byte in the segment**.
- **Acknowledgment number**. This 32-bit field defines the byte number that the receiver of the segment is expecting to receive from the other party.
- **Header length**. This 4-bit field indicates the number of 4-byte words in the TCP header. The length of the header can be between 20 and 60 bytes. Therefore, the value of this field can be between 5 (5 x 4 =20) and 15 (15 x 4 =60).
- **Reserved**. This is a 6-bit field reserved for future use.
- **Control**. This field defines 6 different control bits or flags as shown in Figure 4.17.One or more of these bits can be set at a time.

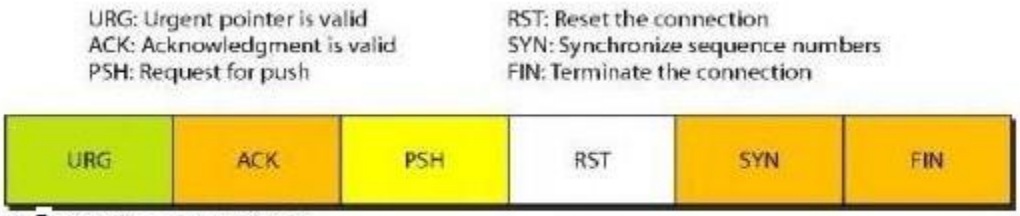


Figure 5.17 Control Field

These bits enable flow control, connection establishment and termination, connection abortion, and the mode of data transfer in TCP. A brief description of each bit is shown in Table 4.3

Table 4.3 Description of flags in the control field

Flag	Description
URG	The value of the urgent pointer field is valid.
ACK	The value of the acknowledge field is valid.
PSH	Push the data.
RST	Reset the connection.
SYN	Synchronize sequence numbers during connection
FIN	Terminate the connection.

- **Window size**. This field defines the size of the window, in bytes, that the other partymust maintain. Note that the length of this field is 16 bits, which means that the maximum size of the window is 65,535 bytes. This value is normally referred to as the receiving window (rwnd) and is determined by the receiver. The sender must obey the dictation of the receiver in this case.
- **Checksum**. This 16-bit field contains the checksum. The inclusion of the checksum forTCP is mandatory. For the TCP pseudoheader, the value for the protocol field is 6.
- **Urgent pointer**. This l6-bit field, which is valid, only if the urgent flag is set, is usedwhen the segment contains urgent data. It defines the number that must be added to the sequence number to obtain the number of the last urgent byte in the data section of the segment.
- **Options.** There can be up to 40 bytes of optional information in the TCP header.

5.3.6 A TCP Connection

TCP is connection-oriented. A connection-oriented transport protocol **establishes a virtual path between the source and destination**. All the segments belonging to a message are then sent over this virtual path. Using a single virtual pathway for the entire message facilitates the acknowledgment process as well as **retransmission of damaged or lost frames**.

***a.* Connection Establishment**

TCP transmits data in **full-duplex mode**. When two TCPs in two machines are connected, they are able to send segments to each other simultaneously. This implies that each party must initialize communication and get approval from the other party before any data are transferred.

Three-Way Handshaking:

- The **connection establishment** in TCP is called three way handshaking. In our example, an application program, called the client, wants to make a connection with another application program, called the server, using TCP as the transport layer protocol.
- The process starts with the server. The server program tells its TCP that it is ready to **accept a connection**. This is called a request for a passive open. Although the server TCP is ready to accept any connection from any machine in the world, it cannot make the connection itself.
- The client program issues a request for an active open. A client that wishes to connect to an open server tells its TCP that it needs to be connected to that particular server. TCP can now start the three-way handshaking process as shown in Figure 4.18.

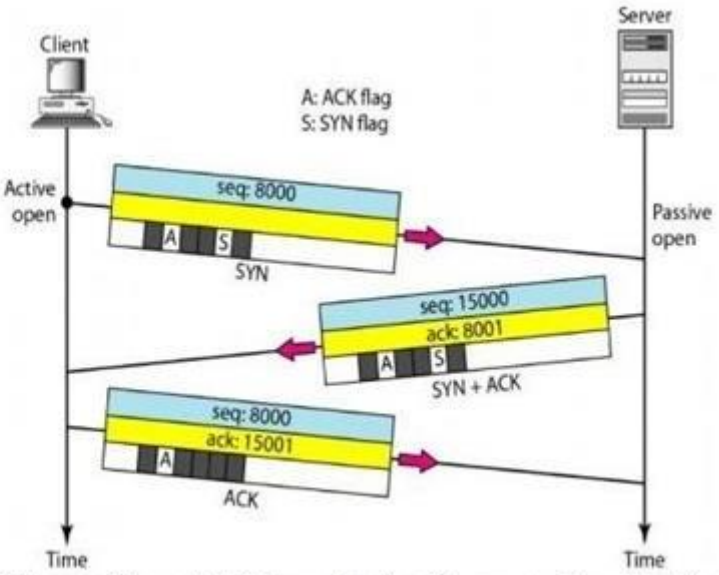


Figure 5. 18 Connection establishment using three-way handshaking

To show the process, we use two time lines: one at each site. Each segment has values for all its header fields and perhaps for some of its option fields, too. However, we show only the few fields necessary to understand each phase. We show the sequence number, the acknowledgment number, the control flags (only those that are set), and the window size, if not empty. The three steps in this phase are as follows.

1. The client sends the first segment, a SYN segment, in which only the SYN flag

is set. This segment is for synchronization of sequence numbers. It consumes one sequence number. When the **data transfer start, the sequence number is incremented by 1.** We can say that the SYN segment carries no real data, but we can think of it as containing 1 imaginary byte.A SYN segment cannot carry data, but it consumes one sequence number.

2. The server sends the second segment, a SYN +ACK segment, with 2 flag bits set: SYN and ACK. This segment has a dual purpose. It is a SYN segment for communication in the other direction and serves as the acknowledgment for the SYN segment. It consumes one sequence number.

A SYN +ACK segment cannot carry data, but does consume one sequence number.

3. The client sends the third segment. This is just an ACK segment. It acknowledges the receipt of the second segment with the ACK flag and acknowledgment number field. Note that the sequence number in this segment is the same as the one in the SYN segment; the ACK segment does not consume any sequence numbers.

An ACK segment, if carrying no data, consumes no sequence number.

Simultaneous Open

A rare situation, called a simultaneous open, may occur when both processes issue an active open. In this case, both TCPs transmit a SYN + ACK segment to each other, and one single connection is established between them.

SYN Flooding Attack

The connection establishment procedure in TCP is susceptible to a serious security problem called the SYN flooding attack. This happens when a **malicious attacker** sends a large number of SYN segments to a server, pretending that each of them is corning from a different client by faking the source IP addresses in the datagrams.

5.3.7 Data Transfer

- A**fter connection is established, bidirectional data transfer can take place.** The client and server can both send data and acknowledgments. The acknowledgment is piggybacked with the data. Figure 4.19 shows an example.
- In this example, after connection is established (not shown in the figure), the client sends 2000 bytes of data in two segments. The server then sends 2000 bytes in one segment.

- The client sends one more segment. The **first three segments carry both data and acknowledgment, but the last segment carries only an acknowledgment because there are no more data to be sent.** Note the values of the sequence and acknowledgment numbers. The data segments sent by the client have the PSH (push) flag set so that the server TCP knows to deliver data to the server process as soon as they are received. The segment from the server, on the other hand, does not set the push flag. Most TCP implementations have the option to set or not set this flag.

Pushing Data

The sending TCP uses a buffer to **store the stream of data** coming from the sending application program. The **sending TCP can select the segment size**. The **receiving TCP also buffers** the data when they arrive and delivers them to the application program when the application program is ready or when it is convenient for the receiving TCP. This type of flexibility increases the efficiency of TCP. However, on occasion the application program has no need for this flexibility.

TCP can handle such a situation. The application program at the sending site can request a push operation. This means that the **sending TCP must not wait for the window to be filled**. It must **create a segment and send it immediately**. The sending TCP must also set the push bit (PSH) to

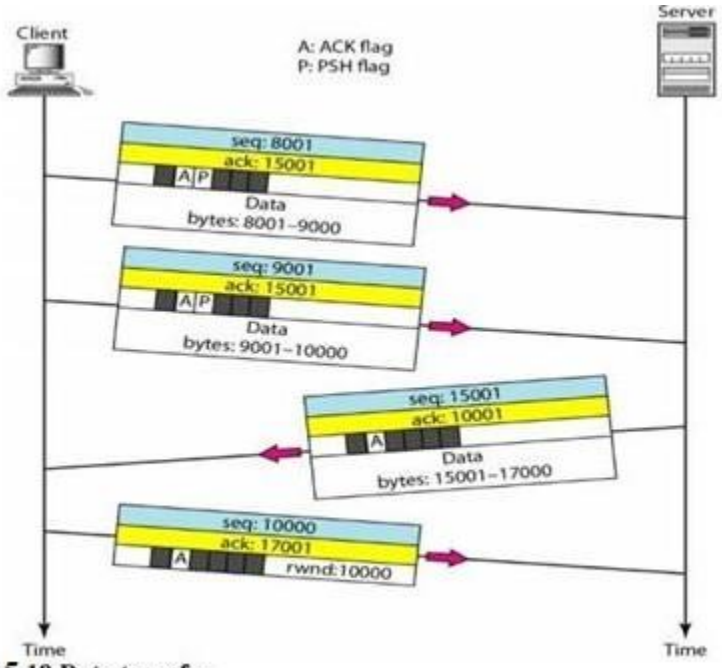


Figure 5.19 Data transfer

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.

Although the push operation can be requested by the application program, most current implementations ignore such requests. TCP can choose whether or not to use this feature.

5.3.8 Urgent Data

TCP is a stream-oriented protocol. This means that the data are presented from the application program to TCP as a stream of bytes. Each byte of data has a position in the stream. However, on occasion an **application program needs to send urgent bytes**. This means that the sending application program wants a piece of data to be read out of order by the receiving application program. As an example, suppose that the sending application program is sending data to be processed by the receiving application program. **When the result of processing comes back, the sending application program finds that everything is wrong. It wants to abort the process, but it has already sent a huge amount of data. If it issues an abort command, these two characters will be stored at the end of the receiving TCP buffer.** It will be delivered to the receiving application program after all the data have been processed.

Connection Termination

Any of the two parties involved in **exchanging data** (client or server) can close the connection,

although it is usually initiated by the client. Most implementations today allow two Options for connection termination: **three-way handshaking and four-way handshaking** with a half-close option.

Three-Way Handshaking

Most implementations today allow three-way handshaking for connection termination.

1. In a normal situation, the client TCP, after receiving a close command from the client process, sends the first segment, a FIN segment in which the FIN flag is set. Note that a FIN segment can include the last chunk of data sent by the client, or it can be just a control segment. If it is only a control segment, it consumes only one sequence number.

The FIN segment consumes one sequence number if it does not carry data.

2. The server TCP, after receiving the FIN segment, informs its process of the situation and sends the second segment, a FIN +ACK segment, to confirm the receipt of the FIN segment from the client and at the same time to announce the closing of the connection in the other direction. This segment can also contain the last chunk of data from the server. If it does not carry data, it consumes only one sequence number.

The FIN +ACK segment consumes one sequence number if it does not carry data.

3. The client TCP sends the last segment, an ACK segment, to confirm the receipt of the FIN segment from the TCP server. This segment contains the acknowledgment number, which is 1 plus the sequence number received in the FIN segment from the server. This segment cannot carry data and consumes no sequence numbers.

Half-Close

In TCP, one end can stop sending data while still receiving data. This is called a half-close.

Although either end can issue a half-close, it is normally initiated by the client. It can occur when

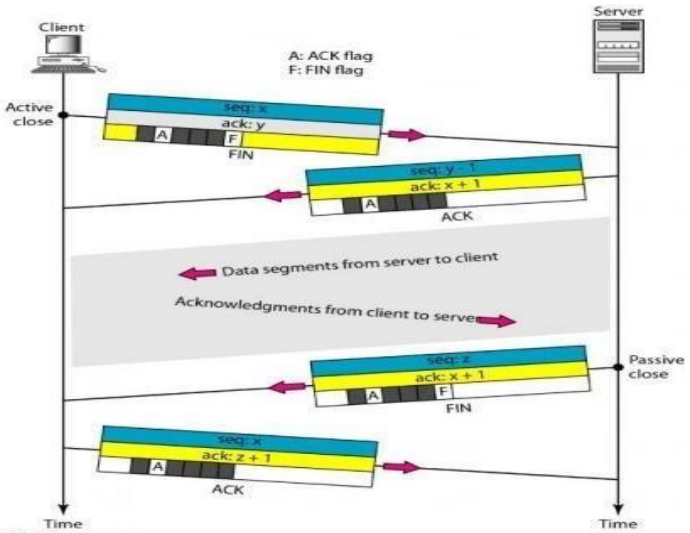


Figure 4.21 Half close

the server needs all the data before processing can begin. A good example is sorting. When the client sends data to the server to be sorted, the server needs to receive all the data before sorting can start. This means the client, after sending all the data, can close the connection in the outbound direction. However, the inbound direction must remain open to receive the sorted data. The server, after receiving the data, still needs time for sorting; its outbound direction must remain open.

Figure 4.21 shows an example of a half-close. The client half-closes the connection by sending a FIN segment. The server accepts the half-close by sending the ACK segment. The data transfer from the client to the server stops. The server, however, can still send data. When the server has sent all the processed data, it sends a FIN segment, which is acknowledged by an ACK from the client. After half-closing of the connection, data can travel from the server to the client and acknowledgments can travel from the client to the server. The client cannot send any more data to the server. Note the sequence numbers we have used. The second segment (ACK) consumes no sequence number. Although the client has received sequence number y - 1 and is expecting y, the server sequence number is still y - 1. When the connection finally closes, the sequence number of the last ACK segment is still x, because no sequence numbers are consumed during data transfer in that direction.

5.3.8 Flow Control

TCP uses a sliding window to handle flow control. The sliding window protocol used by TCP, however, is something between the Go-Back-N and Selective Repeat sliding window. The sliding window protocol in TCP looks like the Go-Back-N protocol because it does not use NAKs; it looks like Selective Repeat because the receiver holds the out-of-order segments until the missing ones arrive.

The window is opened, closed, or shrunk. These three activities, as we will see, are in the control of the receiver (and depend on congestion in the network), not the sender. The sender must obey the commands of the receiver in this matter.

Opening a window means moving the right wall to the right. This allows more new bytes in the buffer that are eligible for sending. Closing the window means moving the left wall to the right. This means that some bytes have been acknowledged and the sender need not worry about them anymore. Shrinking the window means moving the right wall to the left. This is strongly discouraged and not allowed in some implementations because it means revoking the eligibility of some bytes for sending. This is a problem if the sender has already sent these bytes. Note

that the left wall cannot move to the left because this would revoke some of the previously sent acknowledgments.

A sliding window is used to make transmission more efficient as well as to control the flow of data so that the destination does not become overwhelmed with data. TCP sliding windows are byte-oriented.

The size of the window at one end is determined by the lesser of two values:

1. Receiver window (rwnd) or
2. Congestion window (cwnd).

The receiver window is the value advertised by the opposite end in a segment containing acknowledgment. It is the number of bytes the other end can accept before its buffer overflows and data are discarded. The congestion window is a value determined by the network to avoid congestion.

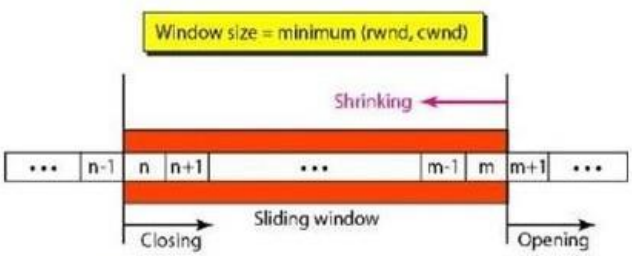


Figure 5.22 Sliding Window

Some points about TCP sliding windows:

The size of the window is the lesser of *rwnd* and *cwnd.*

The source does not have to send a full window's worth of data.

The window can be opened or closed by the receiver, but should not be shrunk.

The destination can send an acknowledgment at any time as long as it does not result in a shrinking window.

The receiver can temporarily shut down the window; the sender, however, can always send a segment of 1 byte after the window is shut down.

Error Control

TCP is a reliable transport layer protocol. This means that an application program that delivers a stream of data to TCP relies on TCP to deliver the entire stream to the application program on the other end in order, without error, and without any part lost or duplicated.

TCP provides reliability using error control. Error control includes mechanisms for detecting corrupted segments, lost segments, out-of-order segments, and duplicated segments. Error control also includes a mechanism for correcting errors after they are detected. Error detection and correction in TCP is achieved through the use of three simple tools: checksum, acknowledgment, and time-out.

a. Checksum

Each segment includes a checksum field which is used to check for a corrupted segment. If the segment is corrupted, it is discarded by the destination TCP and is considered as lost. TCP uses a 16-bit checksum that is mandatory in every segment. The 16-bit checksum is considered inadequate for the new transport layer, SCTP. However, it cannot be changed for TCP because this would involve reconfiguration of the entire header format.

b. Acknowledgment

TCP uses acknowledgments to confirm the receipt of data segments. Control segments that carry no data but consume a sequence number are also acknowledged. ACK segments are never acknowledged.

ACK segments do not consume sequence numbers and are not acknowledged.

c. Retransmission

The heart of the error control mechanism is the retransmission of segments. When a segment is corrupted, lost, or delayed, it is retransmitted. In modern implementations, a segment is retransmitted on two occasions: when a retransmission timer expires or when the sender receives three duplicate ACKs.

In modern implementations, a retransmission occurs if the retransmission timer expires or three duplicate ACK segments have arrived.

Note that no retransmission occurs for segments that do not consume sequence numbers. In particular, there is no transmission for an ACK segment.

No retransmission timer is set for an ACK segment.

i. Retransmission After RTO

A recent implementation of TCP maintains one retransmission time-out (RTO) timer for all outstanding (sent, but not acknowledged) segments. When the timer matures, the earliest outstanding segment is retransmitted even though lack of a received ACK can be due to a delayed segment, a delayed ACK, or a lost acknowledgment. Note that no time-out timer is set for a segment that carries only an acknowledgment, which means that no such segment is resent.

ii. Retransmission After Three Duplicate ACK Segments

The previous rule about retransmission of a segment is sufficient if the value of RTO is not very large. Sometimes, however, one segment is lost and the receiver receives so many out-of-order segments that they cannot be saved (limited buffer size).

iii. Out-of-Order Segments

When a segment is delayed, lost, or discarded, the segments following that segment arrive out of order. Originally, TCP was designed to discard all out-of-order segments, resulting in the retransmission of the missing segment and the following segments. Most implementations today do not discard the out-of-order segments. They store them temporarily and flag them as out-of- order segments until the missing segment arrives.

Some Scenarios

In these scenarios, we show a segment by a rectangle. If the segment carries data, we show the range of byte numbers and the value of the acknowledgment field. If it carries only an acknowledgment, we show only the acknowledgment number in a smaller box.

a. Normal Operation

The first scenario shows bidirectional data transfer between two systems, as in Figure .

The client TCP sends one segment; the server TCP sends three. The figure shows which rule applies to each acknowledgment.

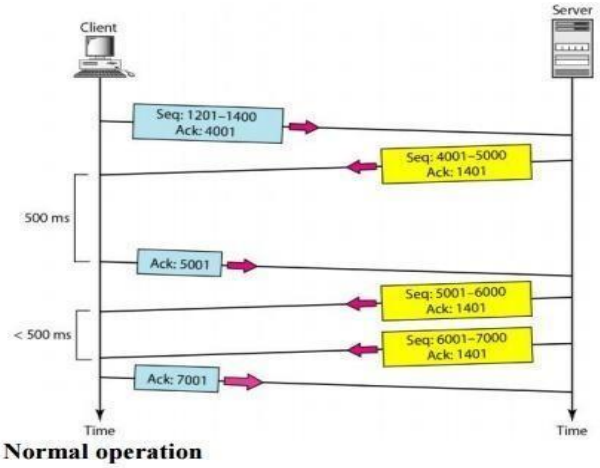


Figure 4.23 Normal operation

There are data to be sent, so the segment displays the next byte expected. When the client receives the first segment from the server, it does not have any more data to send; it sends only an ACK segment

b. Lost Segment

In this scenario, we show what happens when a segment is lost or corrupted. A lost segment and a corrupted segment are treated the same way by the receiver. A lost segment is discarded somewhere in the network; a corrupted segment is discarded by the receiver itself. Both are considered lost. Figure 4.24 shows a situation in which a segment is lost and discarded by some router in the network, perhaps due to congestion.

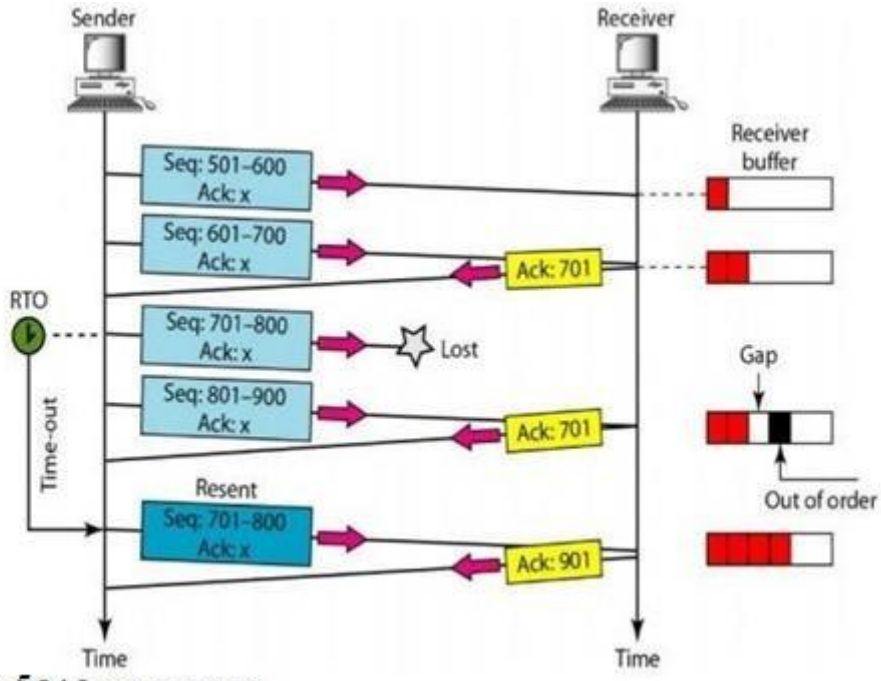


Figure 5.24 Lost segment.

We are assuming that data transfer is unidirectional: one site is sending and the other is receiving. In our scenario, the sender sends segments 1 and 2, which are acknowledged immediately by an ACK. Segment 3, however, are lost. The receiver receives segment 4, which is out of order. The receiver stores the data in the segment in its buffer but leaves a gap to indicate that there is no continuity in the data. The receiver immediately sends an acknowledgment to the sender, displaying the next byte it expects. Note that the receiver stores bytes 801 to 900, but never delivers these bytes to the application until the gap is filled.The receiver TCP delivers only ordered data to the process.

c. Fast Retransmission

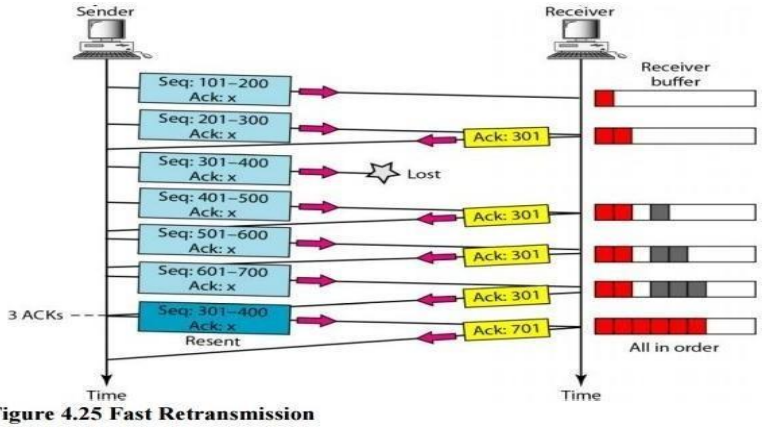


Figure 4.25 Fast Retransmission

In this scenario, we want to show the idea of fast retransmission. Our scenario is the same as the second except that the RTO has a higher value (see Figure 4.25).

When the receiver receives the fourth, fifth, and sixth segments, it triggers an acknowledgment. The

sender receives four acknowledgments with the same value (three duplicates). Although the timer for segment 3 has not matured yet, the fast transmission requires that segment 3, the segment that is expected by all these acknowledgments, be resent immediately.

5.4 Congestion Control: Open Loop and Closed Loop

Congestion control refers to techniques and mechanisms that can either **prevent congestion, before it happens, or remove congestion, after it has happened.**

In general, we can divide congestion control mechanisms into two broad categories: open-loop congestion control (prevention) and closed-loop congestion control (removal) as shown in Figure 4.27.

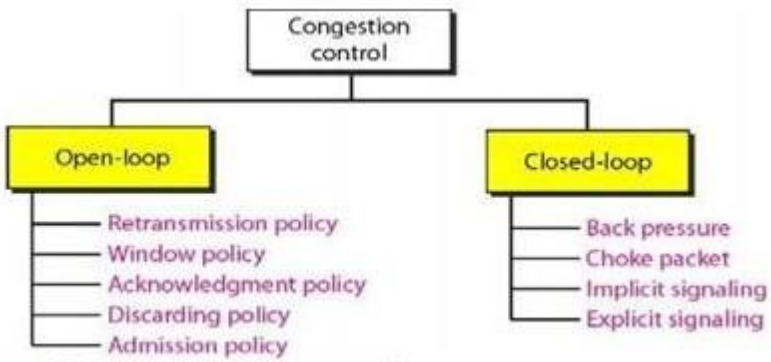


Figure 5. 27 Congestion control categories

1. Open-Loop Congestion Control

In open-loop congestion control, policies are applied to **prevent congestion before it happens**. In these mechanisms, congestion control is **handled by either the source or the destination**.

a. **Retransmission Policy -** Retransmission is sometimes unavoidable. If the sender feels that a sent **packet is lost or corrupted**, the packet needs to be retransmitted. Retransmission in general may i**ncrease congestion** in the network. However, a **good retransmission policy can prevent congestion**. The retransmission policy and the retransmission timers must be designed to **optimize efficiency and at the same time prevent congestion**. For example, the retransmission policy used by TCP (explained later) is designed to prevent or alleviate congestion.

b. **Window Policy -** The type of window at the sender may also affect congestion. The **Selective Repeat window is better than the Go-Back-N** window for congestion control. In the Go-Back-N window, when the timer for a packet times out, several packets may be resent, although some may have arrived safe and sound at the receiver. This duplication may make the congestion worse. The Selective Repeat window, on the other hand, tries to **send the specific packets that have been lost or corrupted.**

c. **Acknowledgment Policy -** The acknowledgment policy imposed by the **receive**r may also

affect congestion. **If the receiver does not acknowledge every packet it receives, it may slow down the sender and help prevent congestion**. Several approaches are used in this case. A receiver may send an acknowledgment only if it has a packet to be sent or a special timer expires. A receiver may decide to acknowledge only N packets at a time. We need to know that the **acknowledgments are also part of the load in a network.** Sending fewer acknowledgments means imposing fewer loads on the network.

d. Discarding Policy - A good discarding policy by the routers may prevent congestion and at the same time may **not harm** the integrity of the transmission. For example, in audio transmission, if the policy is to discard less sensitive packets when congestion is likely to happen, **the quality of sound is still preserved and congestion is prevented or alleviated.**

e. Admission Policy - An admission policy, which is a quality-of-service mechanism, can also **prevent congestion in virtual-circuit networks**. Switches in a flow, **first check the resource requirement of a flow before admitting it to the network.** A router can **deny** establishing a virtual circuit connection **if there is congestion in the network** or if there is a possibility of future congestion.

5.4.2 Closed-Loop Congestion Control

Closed-loop congestion control mechanisms **try to alleviate congestion after it happens.** Several mechanisms have been used by different protocols.

a. Backpressure

The technique of backpressure refers to a congestion control mechanism in which a **congested node stops receiving data from the immediate upstream node or nodes**. This may cause the upstream node or nodes to become congested, and they, in turn, **reject data from their upstream nodes or nodes.** And so on. Backpressure is a node-to-node congestion control that starts with a node and propagates, in the opposite direction of data flow, to the source. The **backpressure technique can be applied only to virtual circuit networks**, in which each node knows the upstream node from which a **flow of data is corning**. Figure 4.28 shows the idea of backpressure.

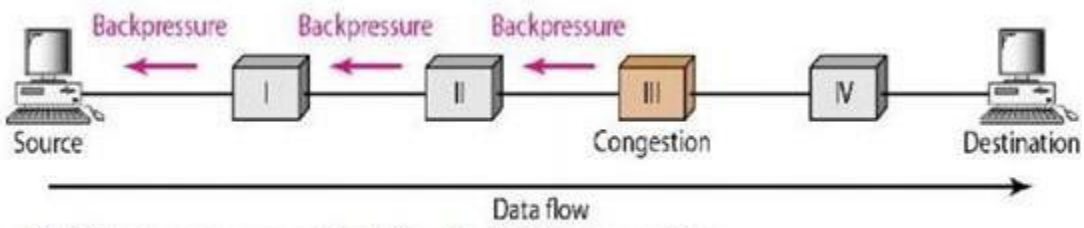


Figure 5.28 Backpressure method for alleviating congestion

Node III in the figure has more input data than it can handle. It drops some packets in its input buffer and informs node II to slow down. Node II, in turn, may be congested because it is slowing down the output flow of data. If node II is congested, it informs node I to slow down, which in turn may create congestion. If so, node I inform the source of data to slow down. This, in time, alleviates the congestion. Note that the pressure on node III is moved backward to the source to remove the congestion.

b. Choke Packet

A choke packet is a **packet sent by a node to the source to inform it of congestion**. Note the difference between the backpressure and choke packet methods. In backpressure, the warning is from one node to its upstream node, although the warning may eventually reach the source station. In the choke packet method, **the warning is from the router, which has encountered congestion, to the source station directly.** The **intermediate nodes through which the packet has traveled are not warned.** We have seen an example of this type of control in ICMP. When a router in the Internet is **overwhelmed with IP datagrams**, it may discard some of them; but it informs the source host, using a source **quench ICMP** message. The warning message goes directly to the source station; the intermediate routers, and does not take any action. Figure 4.29 shows the idea of a choke packet.

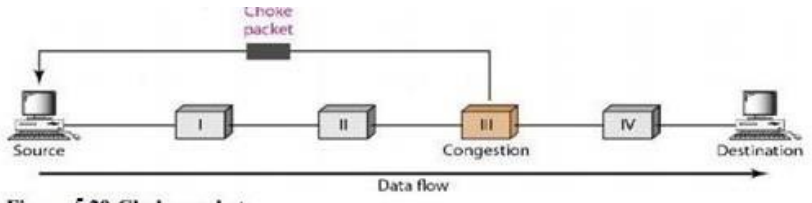


Figure 5.29 Choke packet

Implicit Signaling

In implicit signaling, there is no communication between the congested node or nodes and the source. The **source guesses** that there is congestion somewhere in the network from other symptoms. For example, when a **source sends several packets and there is no acknowledgment for a while, one assumption is that the network is congested**. The delay in receiving an acknowledgment is interpreted as congestion in the network; the source should slow down.

c. Explicit Signaling

The node that experiences congestion can **explicitly send a signal to the source or destination**. The explicit signaling method, however, is different from the choke packet method. In the choke packet method, a separate packet is used for this purpose; in the explicit signaling

method, the **signal is included in the packets that carry data**. Explicit signaling, as we will see in Frame Relay congestion control, can occur in either the **forward or the backward direction**.

i. **Backward Signaling**

A bit can be set in a packet moving in the **direction opposite to the congestion**. This bit can warn the source that there is congestion and that it needs to **slow down to avoid the discarding of packets.**

ii. **Forward Signaling**

A bit can be set in a packet moving in the **direction of the congestion**. This bit can **warn the destination that there is congestion**. The receiver in this case can use policies, such as slowing down the acknowledgments, to alleviate the congestion.

5.5 Quality of services

Quality of service (QoS) is an internetworking issue that has been discussed more than defined. We can informally define quality of service as something a flow seeks to attain.

5.5.1 Flow Characteristics

Traditionally, four types of characteristics are attributed to a flow: reliability, delay, Jitter, and bandwidth, as shown in Figure 4.30

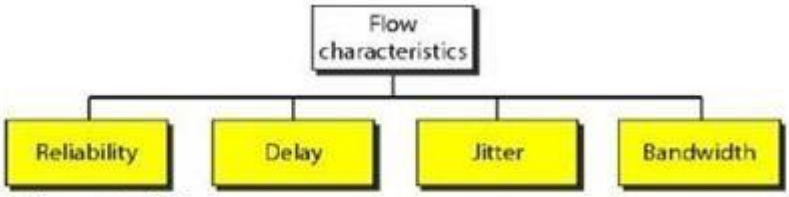


Figure 530 Flow characteristics

a. Reliability

Reliability is a characteristic that a flow needs. Lack of reliability means **losing a packet or acknowledgment, which entails retransmission**. However, the sensitivity of application programs to reliability is not the same. For example, it is more important that **electronic mail, file transfer, and Internet access have reliable transmissions than telephony or audio conferencing**.

b. Delay

Source-to-destination delay is another flow characteristic. Again applications can tolerate delay in different degrees. In this case**, telephony, audio conferencing, video conferencing, and remote log-in need minimum delay, while delay in file transfer or e-mail is less important.**

c. Jitter

- Jitter is **the variation in delay** for packets belonging to the same flow. For example, if four packets depart at times 0, 1, 2, 3 and arrive at 20, 21, 22, 23, all have the same delay, 20 units of time. On the other hand, if the above four packets arrive at 21, 23, 21, and 28, they will have different delays: 21, 22, 19, and 24.
- Jitter is defined as the variation in the packet delay. **High jitter** means the difference between **delays is large**; **low jitter** means the variation is **small**. If the jitter is high, some action is needed in order to use the received data.

d. Bandwidth

- Different applications need different bandwidths. **In video conferencing we need to send millions of bits per second to refresh a color screen while the total number of bits in an e-mail may not reach even a million.**

5.5.2 Flow Classes

- Based on the flow characteristics, we can classify flows into groups, with each group having similar levels of characteristics. This categorization is not formal or universal; some protocols such as ATM have defined classes.

5 . 6 Techniques to Improve QoS

Some techniques that can be used to **improve the quality of service**. The four common methods: scheduling, traffic shaping, admission control, and resource reservation.

e. Scheduling

Packets from different flows arrive at a switch or router for processing. A good scheduling technique treats the different flows in a fair and appropriate manner. Several scheduling techniques are designed to improve the quality of service. We discuss three of them here: FIFO queuing, priority queuing, and weighted fair queuing.

i. FIFO Queuing

In first-in, first-out (FIFO) queuing, packets wait in a buffer (queue) until the node (router or switch) is ready to process them. **If the average arrival rate is higher than the average processing rate, the queue will fill up and new packets will be discarded**. A FIFO queue is familiar to those who have had to wait for a bus at a bus stop.

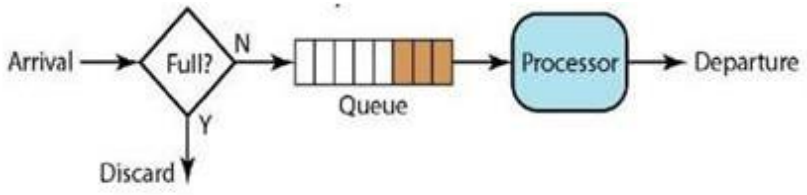


Figure 5.31 FIFO queue

ii. Priority Queuing

In priority queuing, packets are first assigned to a priority class. Each priority class has its own queue. **The packets in the highest-priority queue are processed first. Packets in the lowest- priority queue are processed last**. Note that the system does not stop serving a queue until it is empty. Figure 4.32 shows priority queuing with two priority levels (for simplicity).

A priority queue can provide better QoS than the FIFO queue because higher priority traffic, such as multimedia, can reach the destination with less delay. However, there is a potential drawback. **If there is a continuous flow in a high-priority queue**, the packets in the **lower-priority queues will never have a chance to be processed**. This is a condition called **starvation**

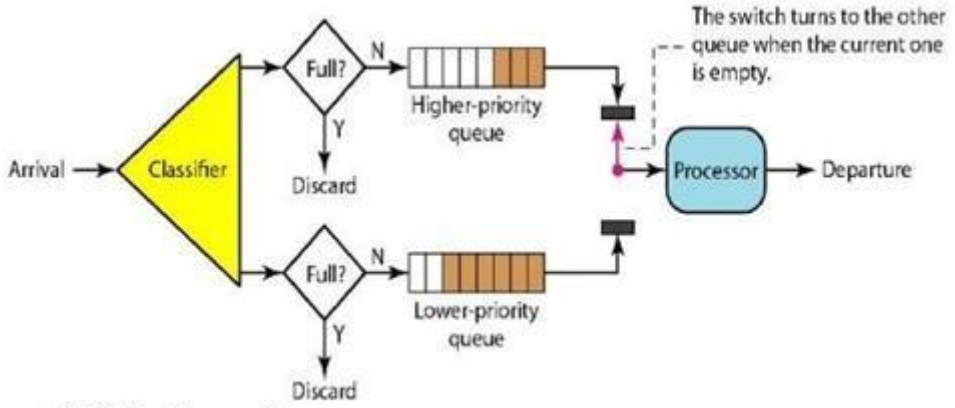


Figure 5.32 Priority queuing

iii. Weighted Fair Queuing

A better scheduling method is weighted fair queuing. In this technique, **the packets are still assigned to different classes and admitted to different queues**. The queues, however, are **weighted based on the priority of the queues**; higher priority means a higher weight. The system processes packets in each queue in a round-robin fashion with the number of packets selected from each queue based on the corresponding weight. For example, if the weights are 3, 2, and 1, three packets are processed from the first queue, two from the second queue, and one from the third queue. If the system does not impose priority on the classes, all weights can be equal. In this way, we have fair queuing with priority. Figure 4.33 shows the technique with three classes.

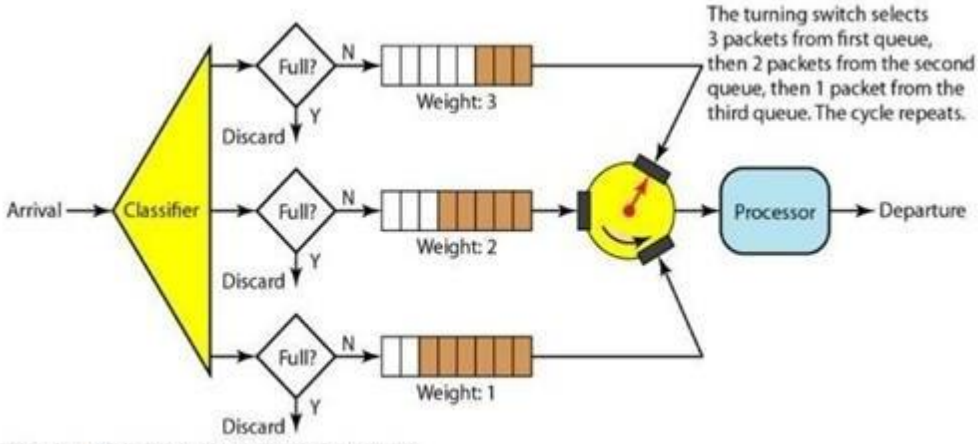


Figure 533 Weighted Fair Queuing

f. Traffic Shaping

Traffic shaping is a mechanism to **control the amount and the rate of the traffic** sent to the network. Two techniques can shape traffic: **leaky bucket and token bucket**

i. Leaky Bucket

If a bucket has a small hole at the bottom, the water leaks from the bucket at a constant rate as long as there is water in the bucket. The rate at which the water leaks does not depend on the rate at which the water is input to the bucket unless the bucket is empty**. The input rate can vary, but the output rate remains constant.** Similarly, in networking, a technique called leaky bucket can smooth out bursty traffic. Bursty chunks are stored in the bucket and sent out at an average rate. Figure 4.34 shows a leaky bucket and its effects.

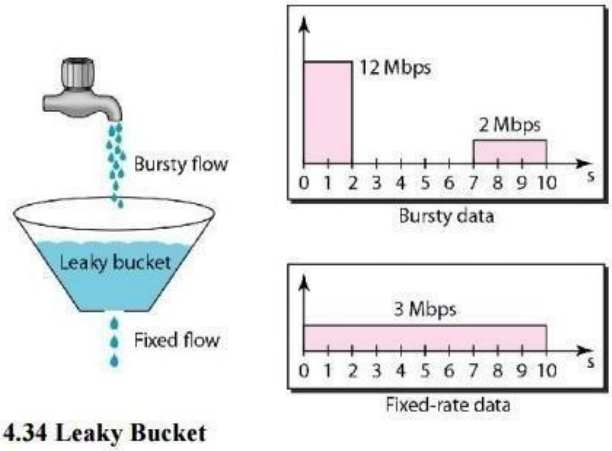


Figure 4.34 Leaky Bucket

In the figure, we assume that the network has committed a bandwidth of 3 Mbps for a host. The use of the leaky bucket shapes the input traffic to make it conform to this commitment. In Figure

the host sends a burst of data at a rate of 12 Mbps for 2 s, for a total of 24 Mbits of data. The host is silent for 5 s and then sends data at a rate of 2 Mbps for 3 s, for a total of 6 Mbits of data. In

all, the host has sent 30 Mbits of data in lOs. The leaky bucket smooth's the traffic by sending out data at a rate of 3 Mbps during the same 10 s.

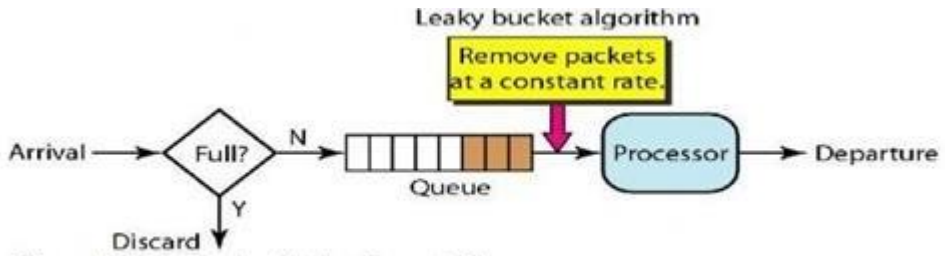


Figure 535 Leaky bucket implementation

A simple leaky bucket implementation is shown in Figure 4.35. A FIFO queue holds the packets. If the traffic consists of fixed-size packets (e.g., cells in ATM networks), the process removes a fixed number of packets from the queue at each tick of the clock. If the traffic consists of variable-length packets, the fixed output rate must be based on the number of bytes or bits.

The following is an algorithm for variable-length packets:

- Initialize a counter to n at the tick of the clock.
- If n is greater than the size of the packet, send the packet and decrement the counter by the packet size. Repeat this step until n is smaller than the packet size.
- Reset the counter and go to step 1.

A leaky bucket algorithm shapes bursty traffic into fixed-rate traffic by averaging the data rate. It may drop the packets if the bucket is full.

ii. Token Bucket

The leaky bucket is **very restrictive**. It does not credit an idle host. For example, if a host is not sending for a while, its bucket becomes empty. Now if the host has bursty data, the leaky bucket allows only an average rate. The time when the host was idle is not taken into account. On the other hand**, the token bucket algorithm allows idle hosts to accumulate credit for the future in the form of tokens.** For each tick of the clock, the system sends n tokens to the bucket. The system removes one token for every cell (or byte) of data sent. For example, if n is 100 and the host is idle for 100 ticks, the bucket collects 10,000 tokens.

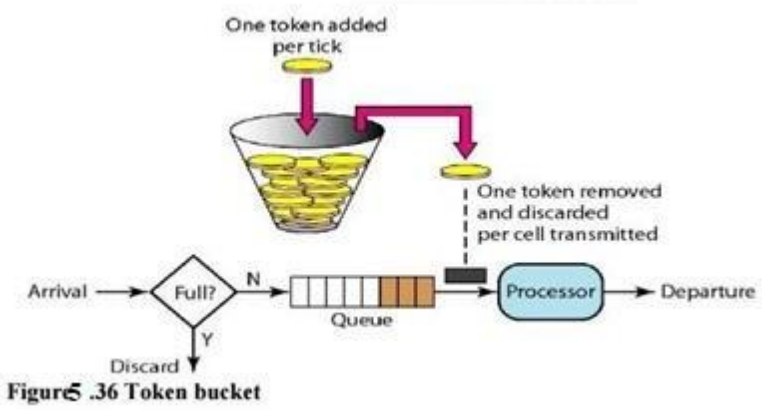


Figure 5.36 Token bucket

The token bucket can easily be implemented with a counter. The token is initialized to zero. Each time a token is added, the counter is incremented by 1. Each time a unit of data is sent, the counter is decremented by 1. When the counter is zero, the host cannot send data.

The token bucket allows bursty traffic at a regulated maximum rate.

Combining Token Bucket and Leaky Bucket

The two techniques can be combined to credit an idle host and at the same time regulate the traffic. The leaky bucket is applied after the token bucket; the rate of the leaky bucket needs to be higher than the rate of tokens dropped in the bucket.

g. Resource Reservation

A flow of data needs resources such as a buffer, bandwidth, CPU time, and so on. The quality of service is improved if these resources are reserved beforehand. We discuss in this section one QoS model called Integrated Services, which depends heavily on resource reservation to improve the quality of service.

h. Admission Control

Admission control refers to the mechanism used by a router, or a switch, to accept or reject a flow based on predefined parameters called flow specifications. Before a router accepts a flow for processing, it checks the flow specifications to see if its capacity (in terms of bandwidth, buffer size, CPU speed, etc.) and its previous commitments to other flows can handle the new flow.

Application Layer Protocol

5.6 Remote Logging - Telnet

It is a network protocol used on the Internet or local area networks to provide a **bidirectional interactive communications** facility. Typically, telnet provides access to a command-line interface on a remote host via a virtual terminal connection which consists of an 8-bit byte oriented data connection over the Transmission Control Protocol (TCP). User data is interspersed in-band

with TELNET control information. The user's computer, which initiates the connection, is referred to as the local computer.

- It is an application layer protocol, which can be used on the internet or LAN(Local Area Network).
- It provides a bi-directional interactive text oriented communication service by using **virtual terminal connection.** Telnet is basically a client server protocol, which is based on a reliable connection- oriented transport. **It uses a port number 23, to establish the connection with TCP (Transmission Control Protocol).**

OVERVIEW OF THE TELNET PROTOCOL FOR REMOTE LOGIN SESSIONS

5.6.1 What is TELNET?

• TELNET (RFC854) is a protocol providing **platform independent, bi-directional byte-oriented communication between hosts** (unlike rlogin which is Unix based).
• Most often TELNET is used for **remote login** to hosts on the Internet.
• TELNET is basically a TCP connection with interspersed TELNET control information.
• TELNET may use option negotiation for providing additional services.

The network terminal protocol (TELNET) allows a **user to log in on any other computer on the network.** We can start a remote session by specifying a computer to connect to. From that time until we finish the session, anything we type is sent to the other computer.

- The Telnet program runs on the computer and connects your PC to a server on the network. We can then enter commands through the Telnet program and they will be executed as if we were entering them directly on the server console. **This enables we to control the server and communicate with other servers on the network.** To start a **Telnet session, we must log in to a server by entering a valid username and password.** Telnet is a common way to remotely control Web servers.
- The term **telnet** also refers to software which implements the client part of the protocol. TELNET clients have been available on most Unix systems for many years and are available virtually for all platforms. Most network equipment and OSs with a TCP/IP stack support some kind of TELNET service server for their remote configuration including ones based on Windows NT. TELNET is a client server protocol, based on a reliable connection oriented transport. Typically this protocol used to **establish a connection to TCP port 23,** where a getty-equivalent program (telnetd) is listening, although TELNET predates.
- Telnet was developed in 1969 to aid in remote connectivity between computers over a

network. Telnet can connect to a remote machine that on a network and is port listening. Most common ports to which one can connect to through telnet are:

Port 21 ~ File Transfer Protocol

Port 22 - SSH Remote Login Protocol

Port 23 - Telnet Server

Port 25 - Simple Mail Transfer Protocol (SMTP)

Port 53 - Domain Name Server (DNS)

Port 69 - Trivial File Transfer Protocol (TFTP)

Port 70 - Gopher

Port 80 - Hyper Text Transfer Protocol (HTTP)

Port 110 - Post Office Protocol 3 (POP3)

- Telnet can be used to connect to virtually any machine that listens on ports. In other words, you can connect to any machine that has certain ports open. Once connected to a machine, you need to issue UNIX based commands to interact with the remote service. For example, a user don't need to login, check and send mails only through his e-mail service provider's interface but this can be achieved using simple telnet commands.
- It is because of this reason that many hackers can send spoofed emails or access information such as which services are running on the remote machine. This is also called banner grabbing or daemon tracking. Black hat hackers can also use telnet to sniff network packets which might contain sensitive information such as usernames and passwords. This is achieved by using telnet and network utilities such as TCP dump and wire shark.
- Telnet client and server functionality comes built-in in most operating systems. However, there are several third-party applications like putty client that enable remote connectivity. A user can connect to a remote machine through several access modes such as raw access, SSH access, etc. SSH mode offers encryption and security and hence can prevent eavesdropping by hackers. This is by far the most secure way of connecting to a machine.
- However, it is necessary that the remote machine supports SSH login to make use of the encryption and security features. On windows machines, telnet client can simply be started by issuing the telnet command in windows command shell. The following example would help you connect to a remote machine on the HTTP Port 80 and issue a GET command which would fetch a file as your web browser does it behind scenes:

Command Prompt> Telnet

Command Prompt> open (somedomain.com or ip address) 80

At this stage, you would be connected to somedomain.com or ip address on port 80 and the daemon that is running on port 80 (most probably HTTP Server) would be waiting for HTTP requests.

GET/HTTP/1.1host: Issuing the command above would make the HTTP Server return the file requested, in this case it would be the default file at the root location, most applications and embedded devices make use of the telnet technology to connect to remote server machines and provide end user functionality. The most common use of telnet stands to enable remote authentication and access,

Establishing Telnet Connection

- To use Telnet, you need to know the address of the host whose resources you want to use. Your Telnet client contacts the host using its internet address. When you contact the host, the distant computer and your computer negotiate how they will communicate with each other. They decide which terminal emulation will be used. Telnet emulation determines how your keyboard will transmit information to the distant computer and how information will be displayed on your screen. For example, it determines how a back space key <- will work.
- Type text in a Telnet session accumulates in a buffer on your computer. When a complete line of data is ready for transmission, or when you give a command to transmit data (such as pressing the Enter key), the data is sent across the Internet from your **Network Virtual Terminal (NVT)** keyboard. Along with the data is the host's IP address, which makes sure that the packet is sent to the proper location.
- Your IP address is also sent so that information can be routed back to you. Additionally, specific Telnet commands that the other NVT will use, are sent to decide what to do with the data, or how to respond to the data. E.g. when data is sent from one NVT to another and certain information must be sent back to the originating NVT for a process to proceed, the Telnet Go Ahead (GA) command is sent.
- After Telnet host receives data you have sent it, processes the data and returns to your screen and give the results of using the data or running the command on a distant computer.

Connecting to a Remote Host

Follow these steps to connect to a remote host using Telnet

1. Open Telnet by clicking on Start menu and choose run. Now type Telnet, and press Enter key from the keyboard or by clicking on the OK button.
2. From the Menu, choose Connect. Remote
3. Enter the name or IP address of the system that you want to connect to in the Host Name Field.
4. If required, a port in the Port field.
5. In the term Type, select the type of terminal that you want Telnet to emulate.
6. After you are finished with the remote host, you can disconnect from a remote host by choosing Connect, Disconnect.

Telnet Protocol Characteristics

There are the various characteristics of Telnet which are described below:

- Telnet is a terminal emulation protocol. When you start installing and configuring native TCP/IP devices, you are going to need some way to connect to the device to issue its commands.
- Telnet is versatile. You can establish Telnet sessions over the phone. If there is no phone connection and your device is accessible to the Internet, you can establish a Telnet session over the Internet. In any of these conditions you can establish a Telnet session with a remote host.

Terminal Emulation

- A personal computer can connect via Modem to a large computer and run a terminal emulation program. The most common terminal emulation is the VT100. The computer works like a dumb terminal, except it is connected via a phone line instead of a direct connection. Often, you will not be able to use graphics on the Internet, such as the WWW (World Wide Web), this kind of access, although you will be able to browse the text-only portion of the Web.
- This kind of Internet account is sometimes called "Shell" account. This shell account is available with VSNL for students in India. Many terminal emulation programs can emulate DEC terminals, including the VT52 and VT200 series terminals. For example, tty pathname of your terminal's device file.

The syntax for this command is **tty [option]**

The options are

1. -1 Prints the synchronous line number.
2. -s Causes tty not to print any output but sets the exit status to 0 if the standard input file

is a terminal, and to 1 if it is not.

TELNET is generally used with the following applications

(1) Enterprise networks to access host applications, e.g. on IBM Mainframes.

(2) Administration of network elements, e.g., in commissioning, integration and maintenance of core network elements in mobile communication networks.

(3) MUD games played over the Internet, as well as talkers, MUSHes, MUCKs, MOOes, and the resurgent BBS community.

(4) embedded systems.

5.7 Electronic Mail

One of the most popular Internet services is electronic mail (e-mail). The designers of the Internet probably never imagined the popularity of this application program. Its architecture consists of several components. At the beginning of the Internet era, the messages sent by electronic mail were short and consisted of text only; they let people exchange quick memos.

1. Architecture

To explain the architecture of e-mail, we give four scenarios. We begin with the simplest situation and add complexity as we proceed. The fourth scenario is the most common in the exchange of email.

First Scenario

In the first scenario, the sender and the receiver of the e-mail are users (or application programs) on the same system; they are directly connected to a shared system. The administrator has created one mailbox for each user where the received messages are stored. A mailbox is part of a local hard drive, a special file with permission restrictions. Only the owner of the mailbox has access to it. When Alice, a user, needs to send a message to Bob, another user, Alice runs a user agent (VA) program to prepare the message and store it in Bob's mailbox. The message has the sender and recipient mailbox addresses (names of files). Bob can retrieve and read the contents of his mailbox at his convenience, using a user agent.

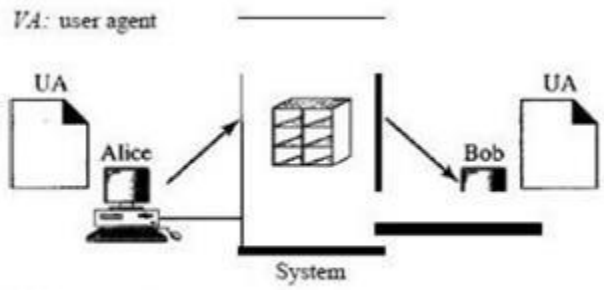


Figure 5.37 First Scenario in e-mail

Second Scenario

In the second scenario, the sender and the receiver of the e-mail are users (or application programs) on two different systems. The message needs to be sent over the Internet. Here we need user agents (VAs) and message transfer agents (MTAs).

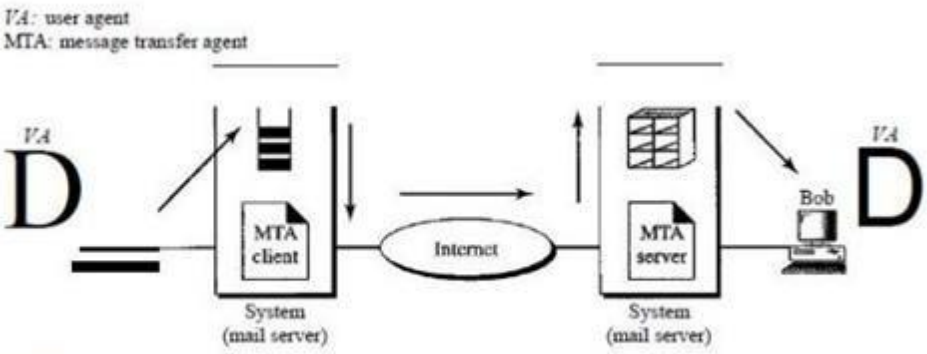


Figure 5.38 Second Scenario in e-mail.

Third Scenario

In the third scenario, Bob, as in the second scenario, is directly connected to his system. Alice, however, is separated from her system. Either Alice is connected to the system via a point-topoint WAN, such as a dial-up modem, a DSL, or a cable modem; or she is connected to a LAN in an organization that uses one mail server for handling e-mails-all users need to send their messages to this mail server.

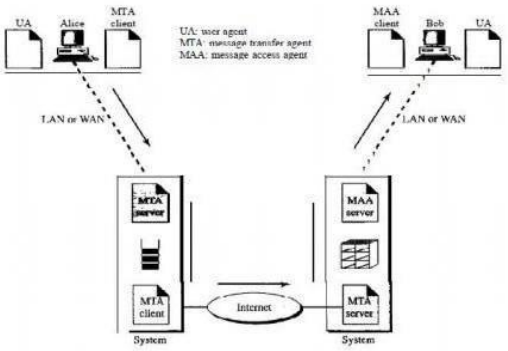


Figure 5.10 Fourth Scenario in e-mail

Fourth Scenario

In the fourth and most common scenario, Bob is also connected to his mail server by a WAN or a LAN. After the message has arrived at Bob's mail server, Bob needs to retrieve it. Here, we need another set of client/server agents, which we call message access agents (MAAs). Bob uses an MAA client to retrieve his messages. The client sends a request to the MAA server, which is running all the time, and requests the transfer of the messages.

2. User Agent

The first component of an electronic mail system is the user agent (VA). It provides service to the user to make the process of sending and receiving a message easier.

Services Provided by a User Agent

A user agent is a software package (program) that composes reads, replies to, and forwards messages. It also handles mailboxes.

Composing Messages A user agent helps the user compose the e-mail message to be sent out. Most user agents provide a template on the screen to be filled in by the user. Some even have a built-in editor that can do spell checking, grammar checking, and other tasks expected from a sophisticated word processor. A user, of course, could alternatively use his or her favorite text editor or word processor to create the message and import it, or cut and paste it, into the user agent template.

Reading Messages The second duty of the user agent is to read the incoming messages. When a user invokes a user agent, it first checks the mail in the incoming mailbox. Most user agents show a one-line summary of each received mail. Each e-mail contains the following fields.

1. A number field.
2. A flag field that shows the status of the mail such as new, already read but not replied to, or read and replied to.
3. The size of the message.
4. The sender.
5. The optional subject field.

Replying to Messages After reading a message, a user can use the user agent to reply to a message. A user agent usually allows the user to reply to the original sender or to reply to all recipients of the message. The reply message may contain the original message (for quick reference) and the new message.

Forwarding Messages Replying is defined as sending a message to the sender a message to the sender or recipients of the copy. Forwarding is defined as sending the message to a third party. A user agent allows the receiver to forward the message, with or without extra comments, to a third party.

3. User Agent Types

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

Command-Driven

Command-driven user agents belong to the early days of electronic mail. They are still present as the underlying user agents in servers. A command-driven user agent normally accepts a one-character command from the keyboard to perform its task. For example, a user can type the character r, at the command prompt, to reply to the sender of the message, or type the character R to reply to the sender and all recipients. Some examples of command-driven user agents are mail, pine, and elm.

GUI-Based Modem user agents are GUI-based. They contain graphical-user interface (GUI)

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

Some examples of GUI based user agents are Eudora, Outlook, and Netscape.

4. Message Transfer Agent: SMTP

The actual mail transfer is done through message transfer agents. To send mail, a system must have the client MTA, and to receive mail, a system must have a server MTA. The formal protocol that defines the MTA client and server in the Internet is called the Simple Mail Transfer Protocol (SMTP). As we said before, two pairs of MTA client/server programs are used in the most common situation (fourth scenario).

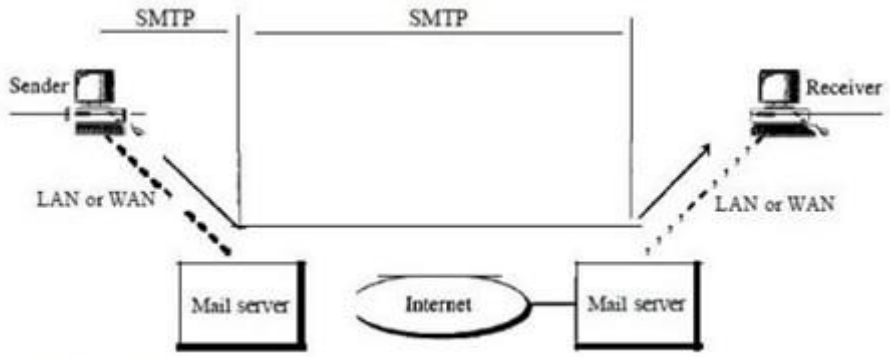


Figure 5.41 SMTP Range

SMTP is used two times, between the sender and the sender's mail server and between the two mail servers. As we will see shortly, another protocol is needed between the mail server and the receiver.

SMTP simply defines how commands and responses must be sent back and forth. Each network is free to choose a software package for implementation.

Commands and Responses

- SMTP uses commands and responses to transfer messages between an MTA client and an MTA server. Each command or reply is terminated by a two-character (carriage return and line feed) end-of-line token.

Commands: Commands are sent from the client to the server. It consists of a keyword followed by zero or more arguments. SMTP defines 14 commands. The first five are mandatory; every implementation must support these five commands. The next three are often used and highly

recommended. The last six are seldom used.

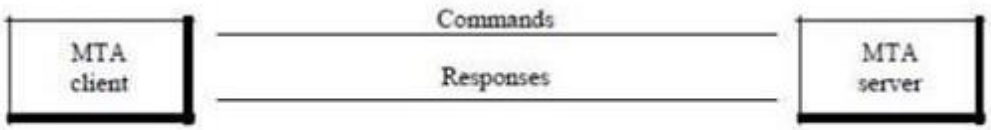


Figure 5.42 Commands and responses

Responses: Responses are sent from the server to the client. A response is a three digit code that may be followed by additional textual information.

5. Mail Transfer Phases

- The process of transferring a mail message occurs in three phases: connection establishment, mail transfer, and connection termination.
- Currently two message access protocols are available: Post Office Protocol, version 3 (POP3) and Internet Mail Access Protocol, version 4 (IMAP4).

POP3

Post Office Protocol, version 3 (POP3) is simple and limited in functionality. The client POP3 software is installed on the recipient computer; the server POP3 software is installed on the mail server. Mail access starts with the client when the user needs to download e-mail from the mailbox on the mail server. POP3 has two modes: **the delete mode and the keep mode**. In the delete mode, the mail is deleted from the mailbox after each retrieval. In the keep mode, the mail remains in the mailbox after retrieval. The delete mode is normally used when the user isworking at her permanent computer and can save and organize the received mail after reading or

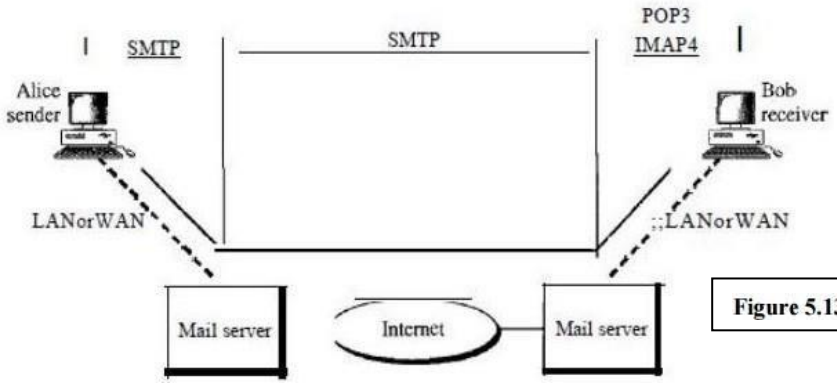


Figure 5.13 POP3 and IMAP4

replying.

The keep mode is normally used when the user accesses her mail away from her primary computer (e.g., a laptop). The mail is read but kept in the system for later retrieval and organizing.

IMAP4

Another mail access protocol is Internet Mail Access Protocol, version 4 (IMAP4). IMAP4 is similar to POP3, but it has more features; IMAP4 is more powerful and more complex

5.8 Domain Name System

There are several applications in the application layer of the Internet model that follow the client/server paradigm. The client/server programs can be divided into two categories: those that can be directly used by the user, such as e-mail, and those that support other application programs. The Domain Name System (DNS) is a supporting program that is used by other programs such as e-mail.

1. Need for DNS

To identify an entity, TCP/IP Protocols use the IP address, which uniquely identifies the connection of a host to an internet. In the case of ARPANET, a file named hosts.txt is used to list all hosts and their IP addresses, this work suitable for small network but not for large network due to heavy load and latency. Therefore, people prefer to use names instead of addresses that is, we need a system that can map a name to an address and conversely an address to a name. Thus, preferred system is called as Domain Name system.

2. DNS in the Internet:

DNS is a protocol that can be used in different platforms. In the Internet, the domain name space is divided into three sections are

Generic domains

1. Country domains and
2. Inverse domain

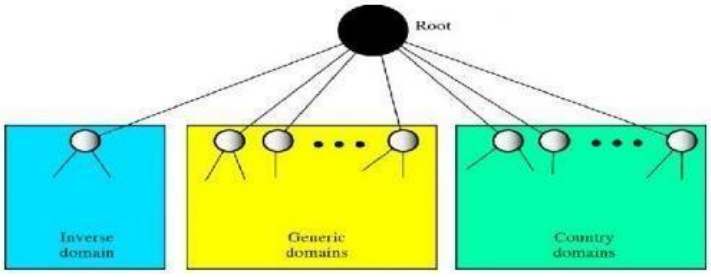


Figure 5.42 DNS in the Internet

1. Generic domain:

There are 14 generic domains, each specifying an organization type. The generic domain defines registered hosts according to their generic behavior.

Each node in the tree defines a domain, which is an index to the domain mane space data base. Looking at the tree, we see that the first level in the generic domain section allows seven possible three-character labels. These labels describe the organization types as shown below

Label	Description
Com	Commercial organizations
Edu	Educational Institutions
Gov	Government Institutions
Int	International organizations
Mil	Military groups
Net	Network support centers
Org	Non profit organizations

Recently a few more first level labels are proposed as,

Label	Description
Arts	Cultural Organizations
Firm	Business or firms
Info	Information service providers

Nom	Personal Nomenclatures
Rec	Recreation/Entertainment Organization
Store	Business offering goods to purchase
web	Web related organizations

2. Country domains:

Each country domain specifies a country. This section follows the same format as the generic domains but uses two-character country abbreviations in place of three character organizational abbreviations at the first level. Second level labels can be organizational, or they can be more specific, national designations. The following figure 5.3 shows the country domain section the address cs.Keio.ac.jp refers to computer science department of Keio University in Japan. To create a new domain, permission is required of the domain in which it will be included.

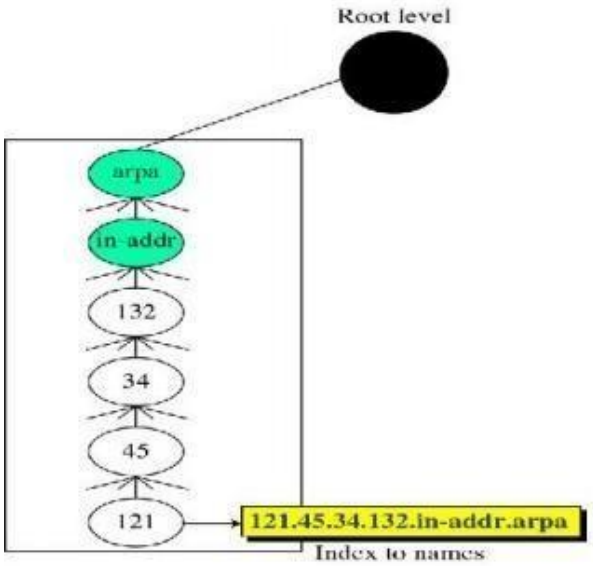
For example, if a new university is chartered, say the University of Chennai, it must ask the manager of the edu domain to assign it unc.edu, in order to avoid conflicts and each domain can keep track of all its subdomains

Once a new domain has been created and registered, it can create subdomains, such as cs.unc.edu,

without getting permission from anybody higher up the tree.

3. Inverse domain:

The inverse domain finds a domain name for a given IP address. This is called address-to-name resolution. It is used to map an address to a name. This may happen, for example, when a server lists only the IP address of the client. To determine if the client is on the authorized list, it can be



send a query to the DNS server and ask for a mapping of address to name in figure 5.4 Types of Records:

There are two types of DNS records:

1. Question records
2. Resource records

Question Records:

The question records are used in the question section of the query and response messages. It is used by the client to get information from a server.

Resource Records:

Every domain whether it is a single host or a top level domain, can have a set of resource records associated with it. For a single host, the most common resource record is just its IP address, but many other kinds also exist. When a resolver gives a domain name to DNS, what it gets back are the resource records associated with that name. Thus, the primary function of DNS is to map domain names onto resource records. The server database consists of resource records. This record is used in the answer, authoritative and additional information sections of the response message.

4. Domain Name space:

DNS can be pictured as an inverted hierarchical tree structure with one root node at the top and a maximum of 128 levels.

Labels:

Each node in the tree has a label, which is string with a maximum of 63 characters.

Domain Name:

Each node in the tree has a domain name. A full domain name is a sequence of labels separated by dots (.).

Fully Qualified Domain Name (FQDN):

A FQDN is a domain name consisting of labels beginning with the host and going back through each level to the root node. Exà Challenger.atc.fh.da.Edu

Partially Qualified Domain Name (PQDN):

In PQDN is a domain name that does not include all the levels between the host and the root node. Exà Challenger.

5. Name Server:

In theory at least, a single name server could contain the entire DNS database and respond to all queries about it. In practice, this server would be so overloaded as to be useless. To avoid problems associated with having only a single source of information, the DNS name space is divided into non-overlapping zones. One possible way to divide the name space, where the zone boundaries are placed within a zone is up to that zones administrator. This decision is made in larger part based on how many name servers are desired. To improve reliability, some servers for a zone can be located outside the zone.

The DNS client, called a resolver, maps a name to an address, or an address to a name. When a resolver has a query about the domain name, it passes the query to one of the local name servers. If the domain being sought falls under the jurisdiction of the name server, such as ai.cs.yale.edu falling under cs.yale.edu, it returns the authoritative resource records. An authoritative record is one that comes from the authority that manages the record and it thus always correct. While, DNS helps in mapping names onto their IP addresses. It does not help locate

people, resources, services or objects in general. For locating these things, another directory service has been defined, called LDAP (Light Weight Directory Access protocol).

6. DNS Messages:

These are two types of DNS Messages queries and responses. Both types have the same format. Queries Messages:

The query message consists of a header and question

Header
Question section

Figure 5.5 Query Message

Response Messages:

The response message consists of a header, question records, answer records, authoritative records

Identification	Flags
Number of questions records	Number of answers records (All 0s in query message)
Number of authoritative records. (All 0s in query message)	Number of additional records. (All 0s in query message)

and additional records.

7. Header Format:

Both are having the same header format. The header is 12 bytes.

Identification

1. Number of questions records
2. Number of authoritative records. (All 0s in query message)

Flags

1. Number of answers records (All 0s in query message)
2. Number of additional records. (All 0s in query message)

- The **identification subfield** is used by the client to match the response with the query.
- The **flag subfield** is a collection of subfields that define the types of the message, the type of answer requested, and the type of desired resolution and so on.
- The **Number of question records subfield** contains the number of queries in the question section of the message
- The **number of answer records subfield** contains the number of answer records in the

answer section of the response message. Its value is zero in the query message.

- The **number of authoritative records subfield** contains the number of authoritative records in the authoritative section of a response message. Its value is zero in the query section.
- The **number of additional records subfield** contains the number of additional records in the additional section of a response message. Its value is zero in the query message.

5.9 SMTP –SIMPLE MAIL TRANSFER PROTOCOL

Next we look at SMTP- the protocol used to transfer messages from one host to another. To place SMTP in the right context, we need to identify the key players. First, users interact with a mail reader when they compose, file, search, and read their email. There are countless mail readers available, just like there are many web browsers now include a mail reader. Second, there is a mail daemon running on each host. You can think of this process as playing the role of a post office: mail readers give the daemon messages they want to send to others users, the daemon uses SMTP running over TCP to transmit the message into a daemon running on another machine, and the daemon puts incoming messages into the user's mailbox. Since SMTP is a protocol that anyone could implement, in theory there could be many different implementations of the mail daemon. It runs out, though that the mail daemon running on most hosts is derived from the send mail program originally implemented on berkely u nix.

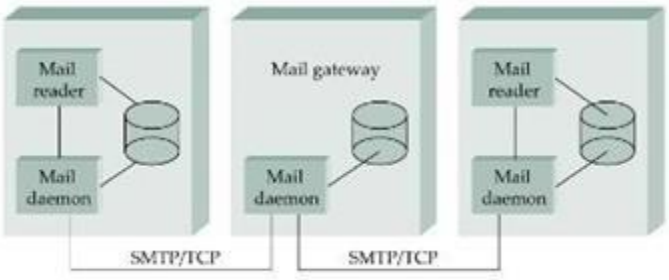


Figure 5.43 Sequence of mail gateways store and forward email messages.

While it is certainly possible that the send mail program on a s ender's machine establishes an SMTP/TCP connection to the send mail program on the recipient's machine, in many cases the mail traverses one or more mail gateways on its route from the sender's host to the receiver's host. Like the end hosts, these gateways also run a send-mail process. It's not an accident that these intermediate nodes are called "gateways" since their job is to store and forward email messages.

Mail Reader:

The final step is for the user to actually receive her messages from the mail box, read them, reply to them, and possibly save a copy for future reference .The user performs all the actions by interacting with a mail reader. In many cases ,this reader is just a program running on the same machine as the user's mailbox resides, in which case it simply reads and writes the file that

implements the mailbox .I n other cases ,the user accesses her mailbox from a remote machine using yet another protocol, such as the Post Office Protocol(POP) or the Internet Message Access Control(IMAP).It is beyond the scope of this book to discuss the user interface aspects of the mail reader but it is definitely within our scope to talk about the access protocol. We consider IMAP, in particular.

IMAP is similar to S MTP in many ways .It is a client/server protocol running over TCP, where the client (running on the user's desktop machine) issues commands in the form of <CRLF> terminated ASCII text lines and the mail server(running on the machine that maintains the user's mailbox) responds in-kind. The exchange begins with the client authenticating herself, an d identifying the mailbox she wants to acc ess. This can be represented by the simple state tr ansaction diagram shown in the figure. In this d iagram, LOGIN, AUTHENTICATE, SELECT, EXAMINE, CLOSE and LOGOUT are exampl e commands that the client can issue, while OK is one possible server response. Other common c ommands include FETCH, STORE, DELETE, an d EXPUNGE, with the obvious meanings. Additional server responds include NO (client d oes not have permission to perform that operation) and BAD (command is ill-formed).

When the user asks to FETCH a message, the server returns it in MIM E format and the mail reader decodes it. In addition to the message itself, IMAP also defines a set of message attributes that are exchanged as part of other commands, independent of transfer ring the message itself. Message attributes include information like the size of the message, but more interestingly, various flags associated with a message, such as Seen, Answered, Deleted and Recent. These flags are used to keep the client and server synchronized, that is, when the user deletes a message in the mail reader, the client needs to report this fact to the mail server. Later, should the user decide to expunge all deleted messages, the client issues an EXPUNGE command to the server, which knows to actually remove all earlier deleted messages from the mail box.

Finally, note that when the user replies to a message,or sends a new message, the mail reader does not forward the message from the client to the mail server using IMAP ,but it instead uses SMTP .This means that the user‟s mail server is effectively the first mail gateway traversed along the path from the desktop to the recipient‟s mail box.

TCP/IP protocol suite specifies a standard for the exchange of mail between machines. It was derived from the (MTP) Mail Transfer Protocol. it deals with how the underlying mail delivery system passes messages across a link from one machine to another. The mail is enclosed in what is called an**envelope.** The envelope contains t he To and From fields and these are followed by the mail . The mail consists of two parts namely the Header and the Data. The Header has the To and From fields. If Headers are defined by us they should start with X. The standard headers do not

start with X. In SMTP data portion can contain only printable ASCII characters The old method of sending a binary file was to send it in uuencoded form but there was no way to distinguish between the many types of binary files possible eg. .tar , .gz , .dvi etc.

5.10 File Transfer Protocol (FTP):

- Transferring files from one computer to another is one of the most common tasks expected from a networking or internetworking environment. As a matter of fact, the greatest volume of data exchange in the Internet today is due to file transfer.
- File Transfer Protocol (FTP) is the standard mechanism provided by TCP/IP for copying a file from one host to another. Although transferring files from one system to another seems simple and straightforward, some problems must be dealt with first.
- FTP differs from other client/server applications in that it establishes two connections between the hosts. One connection is used for data transfer, the other for control information (commands and responses). Separation of commands and data transfer makes FTP more efficient. The control connection uses very simple rules of communication. We need to transfer only a line of command or a line of response at a time. The data connection, on the other hand, needs more complex rules due to the variety of data types transferred. However, the difference in complexity is at the FTP level, not TCP. For TCP, both connections are treated the same. FTP uses two well-known TCP ports: Port 21 is used for the control connection, and port 20 is used for the data connection.
- The client has three components: user interface, client control process, and the client data transfer process. The server has two components: the server control process and the server data transfer process. The control connection is made between the control processes. The data connection is made between the data transfer processes.

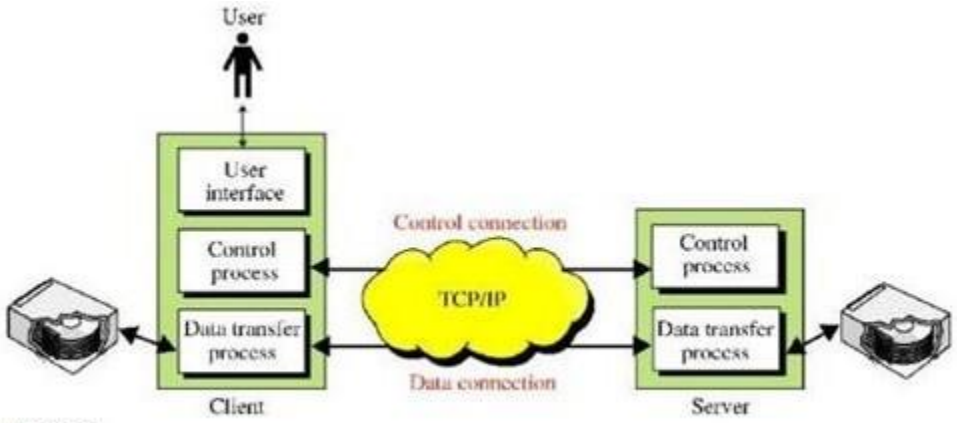


Figure 5. 44FTP

- The control connection remains connected during the entire interactive FTP session. The data connection is opened and then closed for each file transferred. It opens each time commands that involve transferring files are used, and it closes when the file is transferred. In other words, 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.

1. Communication over Control Connection

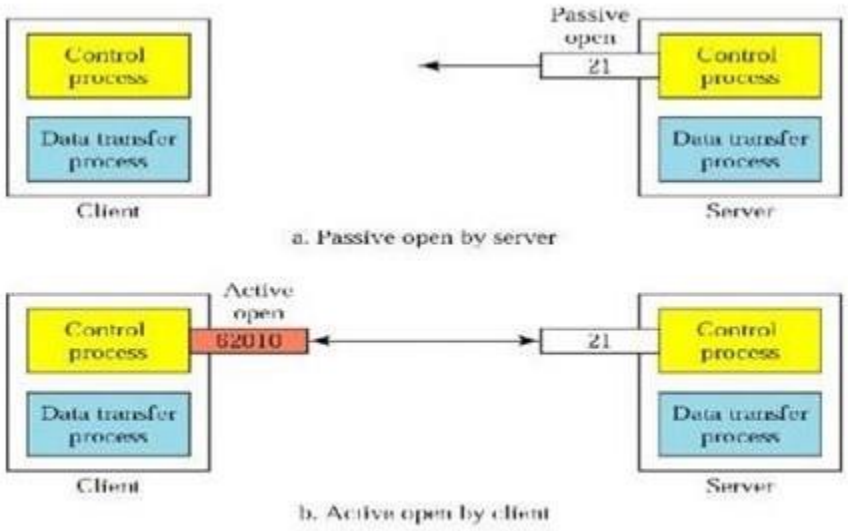


Figure 5. 45Control Connection

FTP uses the same approach as SMTP to communicate across the control connection. It uses the 7-bit ASCII character set. Communication is achieved through commands and responses. This simple method is adequate for the control connection because we send one command (or response) at a time. Each command or response is only one short line, so we need not worry about file format or file structure. Each line is terminated with a two-character (carriage return and line feed) end-of-

line token.

2. Communication over Data Connection

The purpose of the data connection is different from that of the control connection. We want to transfer files through the data connection. File transfer occurs over the data connection under the control of the commands sent over the control connection. However, we should remember that file transfer in FTP means one of three things:

- A file is to be copied from the server to the client. This is called retrieving aft/e. 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 aft/e. 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. Note that FTP treats a list of directory or file names as a file. It is sent over the data connection.

The client must define the type of file to be transferred, the structure of the data, and the transmission mode. Before sending the file through the data connection, we prepare for transmission through the control connection. The heterogeneity problem is resolved by defining three attributes of communication: file type, data structure, and transmission mode

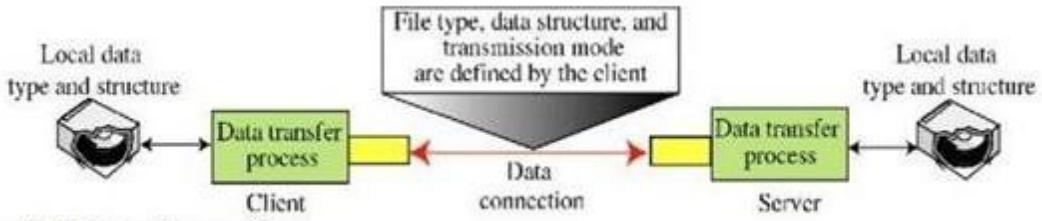


Figure 5.46 Data Connection

File Type:

FTP can transfer one of the following file types across the data connection: an ASCII file, EBCDIC file, or image file. The ASCII file is the default format for transferring text files. Each character is encoded using 7-bit ASCII. The sender transforms the file from its own representation into ASCII characters, and the receiver transforms the ASCII characters to its own representation.

Data Structure

FTP can transfer a file across the data connection by using one of the following interpretations about the structure of the data: file structure, record structure, and page structure. In the file structure format, the file is a continuous stream of bytes. In the record structure, the file is divided into records.

Transmission Mode

FTP can transfer a file across the data connection by using one of the following three transmission modes: stream mode, block mode, and compressed mode. The stream mode is the default mode. Data are delivered from FTP to TCP as a continuous stream of bytes. TCP is responsible for chopping data into segments of appropriate size. If the data are simply a stream of bytes (file structure), no end-of-file is needed. End-of-file in this case is the closing of the data connection by the sender.

Hypertext Transfer Protocol (HTTP)

The Hypertext Transfer Protocol (HTTP) is a protocol used mainly to access data on the World Wide Web.

HTTP functions as a combination of FTP and SMTP. It is similar to FTP because it transfers files and uses the services of TCP. However, it is much simpler than FTP because it uses only one TCP connection. There is no separate control connection; only data are transferred between the client and the server. HTTP is like SMTP because the data transferred between the client and the server look like SMTP messages. In addition, the format of the messages is controlled by MIME-like headers. Unlike SMTP, the HTTP messages are not destined to be read by humans; they are read and interpreted by the HTTP server and HTTP client (browser). SMTP messages are stored and forwarded, but HTTP messages are delivered immediately. The commands from the client to the server are embedded in a request message. The contents of the requested file

or other information are embedded in a response message. HTTP uses the services of TCP on well- known port 80.

1. **HTTP Transaction**

Although HTTP uses the services of TCP, HTTP itself is a stateless protocol. The client initializes the transaction by sending a request message. The server replies by sending a response.

Messages

The formats of the request and response messages are similar. A request message consists of a request line, a header, and sometimes a body. A response message consists of a status line, a header, and sometimes a body. Request and Status Lines The first line in a request message is called a request line; the first line in the response message is called the status line.

- **Request type:** This field is used in the request message. In version1.1of HTTP, several request types are defined.
- **Version:** The most current version of HTTP is 1.1.
- **Status code**: This field is used in the response message. The status code field is similar to those in the FTP and the SMTP protocols. It consists of three digits.
- **Status phrase**: This field is used in the response message. It explains the status code in text form.
- **Header:** The header exchanges additional information between the client and the server. For example, the client can request that the document be sent in a special format, or the server can send extra information about the document. The header can consist of one or more header lines. Each header line has a header name, a colon, a space, and a header value.
- **Body**: The body can be present in a request or response message. Usually, it contains the document to be sent or received.

2. Persistent versus Non persistent Connection

HTTP prior to version 1.1 specified a non persistent connection, while a persistent connection is the default in version 1.1.

Non persistent Connection

In a non persistent connection, one TCP connection is made for each request/response. The following lists the steps in this strategy:

1. The client opens a TCP connection and sends a request.
2. The server sends the response and closes the connection.
3. The client reads the data until it encounters an end-of-file marker; it then closes the connection.

Persistent Connection

HTTP version 1.1 specifies a persistent connection by default. In a persistent connection, the server leaves the connection open for more requests after sending a response. The server can close the connection at the request of a client or if a time-out has been reached. The sender usually sends the length of the data with each response. However, there are some occasions when the sender does not know the length of the data. This is the case when a document is created dynamically or actively.