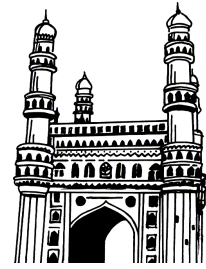


Rahul's ✓
Topper's Voice



M.C.A.

II Year III Sem

(Osmania University)

Latest 2024 Edition

COMPUTER NETWORKS

- ☞ Study Manual
- ☞ Important Questions
- ☞ Solved Model Papers
- ☞ Solved Previous Question Papers

- by -

WELL EXPERIENCED LECTURER

Price
199-00



Rahul Publications TM

Hyderabad. Cell : 9391018098, 9505799122

All disputes are subjects to Hyderabad Jurisdiction only

M.C.A.

II Year III Sem
(*Osmania University*)

COMPUTER NETWORKS

Inspite of many efforts taken to present this book without errors, some errors might have crept in. Therefore we do not take any legal responsibility for such errors and omissions. However, if they are brought to our notice, they will be corrected in the next edition.

© No part of this publication should be reproduced, stored in a retrieval system, or transmitted in any form or by any means, electronic, mechanical, photocopying, recording and/or otherwise without the prior written permission of the publisher

Price ` . 199 -00

Sole Distributors :

Cell : 9391018098, 9505799122

VASU BOOK CENTRE

Shop No. 2, Beside Gokul Chat, Koti, Hyderabad.

Maternity Hospital Opp. Lane, Narayan Naik Complex, Koti, Hyderabad.

Near Andhra Bank, Subway, Sultan Bazar, Koti, Hyderabad -195.

COMPUTER NETWORKS

STUDY MANUAL

Important Questions	IV - VIII
Unit - I	1 - 58
Unit - II	59 - 98
Unit - III	99 - 144
Unit - IV	145 - 166
Unit - V	167 - 190
Lab Programs	191 - 228

SOLVED MODEL PAPERS

Model Paper - I	229 - 229
Model Paper - II	230 - 230
Model Paper - III	230 - 231

PREVIOUS QUESTION PAPERS

April / May - 2023	232 - 233
October / November - 2023	234 - 234

Contents

UNIT - I

Topic	Page No.
1.1 Data Communication	1
1.2 Uses of Computer Networks	3
1.3 Components of Computer Networks	5
1.4 Types of Networks	10
1.4.1 Local Area Network (LAN)	11
1.4.2 Metropolitan Area Network (MAN)	13
1.4.3 Wide Area Network (WAN)	14
1.4.4 Wireless Network	15
1.4.4.1 Connection Oriented and Connection less Services	16
1.5 OSI Reference Model	18
1.5.1 TCP/IP	23
1.5.2 TCP/IP Reference Model	24
1.5.2.1 Similarities between OSI Reference and TCP/IP Reference Model	27
1.6 Network Software	28
1.6.1 Issues	29
1.7 Network Topology	30
1.8 Transmission Media	36
1.8.1 Coaxial Cable	37
1.8.2 Fiber Optics	40
1.8.3 Line Coding	43
1.8.4 Modem	50
1.8.4.1 Asynchronous & Synchronous Modems	54
1.9 RS232 Interfacing	55

UNIT - II

2.1 Data Link Layer	59
2.1.1 Error Detection and Correction	61
2.2 Cyclic Redundancy Check (CRC)	63
2.3 Hamming Code	67
2.4 Flow Control & Error Control	71

Topic	Page No.
2.4.1 Stop and Wait protocol, Sliding Window protocol-go back-N ARQ - selective repeat ARQ	71
2.5 MAC Layer	76
2.5.1 The Channel Allocation Problem	78
2.5.2 Dynamic Channel Allocation in LANs and MANs	79
2.6 ALOHA	80
2.6.1 Pure ALOHA	81
2.6.2 Slotted ALOHA	82
2.6.3 Pure ALOHA Vs Slotted ALOHA	84
2.7 Ethernet IEEE 802.3 LAN Ethernet Efficiency Calculation	84
2.8 Bridges	87
2.9 ARP	95
2.10 RARP	96
UNIT - III	
3.1 Network Layer	99
3.2 Routers	108
3.2.1 Distance vector Routing	110
3.3 Link-State Routing	114
3.4 IPv4 Addressing	117
3.5 Subnetting	121
3.6 CIDR	122
3.7 Introduction to IPv6	129
3.7.1 ICM	131
3.7.2 IGMP	132
3.7.3 OSPF	135
3.7.4 Border Gateway Protocol (BGP)	140
UNIT - IV	
4.1 Transport layer	145
4.1.1 Services of transport layer	145
4.2 Multiplexing	147
4.3 Transmission Control Protocol	153

Topic	Page No.
4.3.1 Congestion Control	155
4.3.2 Timer Management	159
4.4 Quality of Services	161
4.5 User Datagram Protocol (UDP)	164

UNIT - V

5.1 Socket Programming	167
5.1.1 Primitive and Advance System Calls	167
5.2 Iterative and Concurrent Client Server Programs	169
5.3 Application Layer	173
5.3.1 DNS (Domain Name System)	174
5.3.2 SMTP	184
5.3.3 File Transfer Protocol (FTP)	185
5.3.4 HTTP	187

Important Questions

UNIT - I

1. What are the different uses of Networks?

Ans :

Refer Unit-I, Q.No. 4

2. Explain the basic components of computer networking?

Ans :

Refer Unit-I, Q.No. 5

3. Define key requirements and functions of Wireless LANs?

Ans :

Refer Unit-I, Q.No. 11

4. Explain connection oriented and connection less networks ?

Ans :

Refer Unit-I, Q.No. 12

5. What is mean by TCP/IP?

Ans :

Refer Unit-I, Q.No. 14

6. What are the design issues for the Layers?

Ans :

Refer Unit-I, Q.No. 18

7. What is transmission media? State different types of transmission media.

Ans :

Refer Unit-I, Q.No. 20

8. What is a Modem? Explain different types of Modems.

Ans :

Refer Unit-I, Q.No. 27

9. Explain the concept of RS-232 interfacing.

Ans :

Refer Unit-I, Q.No. 29

UNIT - II

1. What is DLL? State the functions of DLL.

Ans :

Refer Unit-II, Q.No. 1

2. Define CRC. State the process of CRC.

Ans :

Refer Unit-II, Q.No. 4

3. Explain the concept of Hamming code.

Ans :

Refer Unit-II, Q.No. 5

4. Explain in detail about the working of flow control?

Ans :

Refer Unit-II, Q.No. 6

5. Explain and discuss the Architecture of HDLC Protocol ?

Ans :

Refer Unit-II, Q.No. 8

6. Define MAC & state the functions of MAC?

Ans :

Refer Unit-II, Q.No. 9

7. Explain Dynamic channel allocation in LAN and MAN ?

Ans :

Refer Unit-II, Q.No. 11

8. What is ALOHA? Different types of Aloha?

Ans :

Refer Unit-II, Q.No. 12

9. Explain the concept of Bridges.

Ans :

Refer Unit-II, Q.No. 16

10. Explain the concept of Transparent Bridges.

Ans :

Refer Unit-II, Q.No. 17

11. What are the differences between ARP and RARP?

Ans :

Refer Unit-II, Q.No. 23

UNIT - III

1. Define networking layer in TCP/IP Model.

Ans :

Refer Unit-III, Q.No. 1

2. What is Router ? Explain the Characteristics of Routers/Router Protocols.

Ans :

Refer Unit-III, Q.No. 7

3. What is Distance Vector Protocol Routing ?

Ans :

Refer Unit-III, Q.No. 8

4. What is Link State Routing Protocol ?

Ans :

Refer Unit-III, Q.No. 9

5. Explain IPv4 addressing methods.

Ans :

Refer Unit-III, Q.No. 11

6. Explain the concept of subnetting. State its advantages and disadvantages.

Ans :

Refer Unit-III, Q.No. 12

7. What is CIDR ? Discuss with an Example.

Ans :

Refer Unit-III, Q.No. 13

8. How Network Address Translation Works ?

Ans :

Refer Unit-III, Q.No. 15

9. What are the differences between IPv4 and IPv6 ?

Ans :

Refer Unit-III, Q.No. 18

10. Discuss about OSPF in detail.

Ans :

Refer Unit-III, Q.No. 22

11. Explain the concept of Border Gateway Protocol (BGP).

Ans :

Refer Unit-III, Q.No. 23

UNIT - IV

1. Define transport layer. Explain the services of transport layer.

Ans :

Refer Unit-IV, Q.No. 1

2. What is multiplexing? Explain its methods.

Ans :

Refer Unit-IV, Q.No. 2

3. Define Transmission Control Protocol. State the various services of TCP.

Ans :

Refer Unit-IV, Q.No. 3

4. Define QoS. Explain the specific purpose of QoS.

Ans :

Refer Unit-IV, Q.No. 6

5. Explain UDP and its Attributes.

Ans :

Refer Unit-IV, Q.No. 7

UNIT - V

1. Describe various services of transport layer.

Ans :

Refer Unit-V, Q.No. 2

2. Explain Application Layer and its Functions.

Ans :

Refer Unit-V, Q.No. 5

3. Explain in concept of Domain Name System.

Ans :

Refer Unit-V, Q.No. 6

4. How Simple Mail Transfer Protocol (SMTP) works?

Ans :

Refer Unit-V, Q.No. 11

5. What is FTP - File Transfer Protocol?

Ans :

Refer Unit-V, Q.No. 12

6. What is HTTP? Explain the features of HTTP.

Ans :

Refer Unit-V, Q.No. 13

7. How to Check HTTP Request and Response?

Ans :

Refer Unit-V, Q.No. 15

UNIT I

Data Communications : Components - Direction of Data flow - networks - Components and Categories- types of Connections - Topologies -Protocols and Standards - ISO/OSI model, TCP/IP. Transmission Media - Coaxial Cable - Fiber Optics - Line Coding - Modems - RS232 Interfacing.

1.1 DATA COMMUNICATION

Q1. Explain the concept of network.

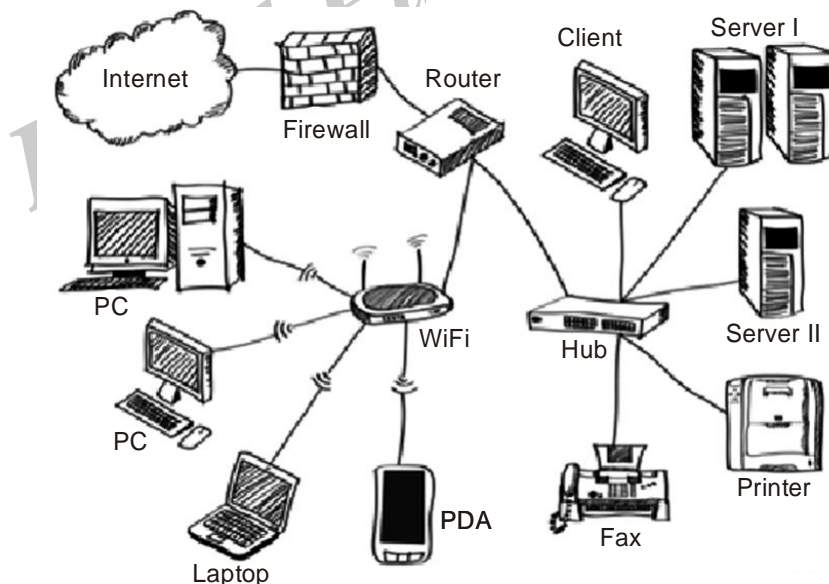
Ans :

Data Communication and network have changed the way business and other daily affair works. Now, they highly rely on computer networks and internetwork.

A set of devices often mentioned as nodes connected by media link is called a Network.

A node can be a device which is capable of sending or receiving data generated by other nodes on the network like a computer, printer etc. These links connecting the devices are called

Communication Channels



Computer network is a telecommunication channel using which we can share data with other computers or devices, connected to the same network. It is also called Data Network.

Computer network does not mean a system with one Control Unit connected to multiple other systems as its slave. That is Distributed system, not Computer Network.

A network must be able to meet certain criterias, these are mentioned below:

1. Performance
2. Reliability
3. Scalability

A computer network is a group of computer systems and other computing hardware devices that are linked together through communication channels to facilitate communication and resource-sharing among a wide range of users. Networks are commonly categorized based on their characteristics.

Networks are used to:

1. Facilitate communication via email, video conferencing, instant messaging, etc.
2. Enable multiple users to share a single hardware device like a printer or scanner.
3. Enable file sharing across the network.
4. Allow for the sharing of software or operating programs on remote systems.
5. Make information easier to access and maintain among network users.

There are many types of networks, including:

- Local Area Networks (LAN)
- Personal Area Networks (PAN)
- Home Area Networks (HAN)
- Wide Area Networks (WAN)
- Campus Networks
- Metropolitan Area Networks (MAN)
- Enterprise Private Networks
- Internetworks
- Backbone Networks (BBN)
- Global Area Networks (GAN)
- The Internet

Q2. State the characteristics and advantages of networks.

Ans :

Characteristics

1. **Sharing Resources** from one Computer to another Computer over a network.

2. **Performance** by measuring the speed of data transmission with number of users, connectivity and the software used.
3. **Reliability** makes easy to use an alternative source for data communication in case of hardware failure or connectivity issues.
4. **Scalability** increases the system performance by adding more processors.
5. **Security** is the main characteristics of Computer network where you can take necessary steps for protecting your data from unauthorized access.

Advantages

The following are the distinct notes in favor of computer network.

1. The computers, staff and information can be well managed.
2. A network provides the means to exchange data among the computers and to make programs and data available to people.
3. It permits the sharing of the resources of the machine.
4. Networking also provides the function of back-up.
5. Networking provides a flexible networking environment. Employees can work at home by using through networks ties through networks into the computer at office.

Q3. Explain briefly about Data Communications.

Ans :

Data communications (DC) is the process of using computing and communication technologies to transfer data from one place to another, and vice versa. It enables the movement of electronic or digital data between two or more nodes, regardless of geographical location, technological medium or data contents.

Data communications incorporates several techniques and technologies with the primary objective of enabling any form of electronic communication. These technologies include telecommunications, computer networking and radio/satellite communication. Data communication

usually requires existence of a transportation or communication medium between the nodes wanting to communicate with each other, such as copper wire, fiber optic cables or wireless signals.

Some devices/technologies used in data communications are known as data communication equipment (DCE) and data terminal equipment (DTE). DCE is used at the sending node, and DTE is used at the receiving node.

Different media are employed for transmitting data from one computer terminal to the central computer or to other computer systems inside some kind of network.

There are two forms of communication media:

- **Analog:** Includes conventional radio, telephonic and television transmissions
- **Digital:** Computer-mediated communication, computer networking and telegraphy

The most commonly used data communication media include

- Wire pairs
- Coaxial cable
- Microwave transmission
- Communication satellites
- Fiber optics

The communication media acts as a channel for linking various computing devices so that they may interact with each other. Contemporary communication media facilitate communication and data exchange among a large number of individuals across long distances via email, teleconferencing, internet forums and many other forms of communication.

1.2 USES OF COMPUTER NETWORKS

Q4. What are the different uses of Networks?

Ans : (Imp.)

Computer networks help users on the network to share the resources and in communi-

cation. Can you imagine a world now without emails, online newspapers, blogs, chat and the other services offered by the internet? The following are the important uses and benefits of a computer network.

1. File Sharing

Networking of computers helps the network users to share data files.

2. Hardware Sharing

Users can share devices such as printers, scanners, CD-ROM drives, hard drives etc. Without computer networks, device sharing is not possible.

3. Application Sharing

Applications can be shared over the network, and this allows to implement client/server applications.

4. User Communication

Networks allow users to communicate using e-mail, news groups, and video conferencing etc.

5. Network Gaming

A lot of network games are available, which allow multi-users to play from different locations.

6. Voice over IP (VoIP)

Voice over Internet Protocol (IP) is a revolutionary change in telecommunication which allows to send telephone calls (voice data) using standard Internet Protocol (IP) rather than by traditional PSTN.

Business Applications

1. Resource Sharing

The goal is to make all programs, equipment's, and especially data, available to anyone on the network without regard to the physical location of the resource and the user.

2. Server-Client Model

One can imagine a company's information system as consisting of one or more databases and some number of employees who need to access them remotely. In this model, the

data is stored on powerful computers called servers. Often these are centrally housed and maintained by a system administrator. In contrast, the employees have simple machines, called clients, on their desks, with which they access remote data.

3. Communication Medium

A computer network can provide a powerful communication medium among employees. Virtually every company that has two or more computers now has e-mail (electronic mail), which employees generally use for a great deal of daily communication

4. E-Commerce

A goal that is starting to become more important is doing business with consumers over the Internet. Airlines, bookstores and music vendors have discovered that many customers like the convenience of shopping from home.

Tag and Full Name	Example
B2C - Business-to-Consumer	Ordering books on-line
B2B - Business-to-Business	Car manufacture ordering tires from supplier
C2C - Consumer-to-Consumer	Auctioning second-hand products on line
G2C - Government-to-Consumer	Government Distributing tax forms electronically
P2P - Peer-to-Peer	File sharing

Home Applications

- Access to Remote Information** : Computer Network facilitates users to access information that is distant away by staying at home remotely.
- Person-to-person Communication** : Users can use Computer Network in their home to communicate with other peoples by telephone, video chat, etc.
- Interactive Entertainment** : Computer Network is used in multiplayer gaming. It is also used in social networking sites like face book, twitter, etc to connect people.

Mobile Users

Mobile computers, such as notebook computers and personal digital assistants (PDAs), are one of the fastest-growing segments of the computer industry. Many owners of these computers have desktop machines back at the office and want to be connected to their home base even when away from home or enroute. Since having a wired connection is impossible in cars and airplanes, there is a lot of interest in wireless networks. In this section we will briefly look at some of the uses of wireless networks.

Why would anyone want one? A common reason is the portable office. People on the road often want to use their portable electronic equipment to send and receive telephone calls, faxes, and electronic mail, surf the Web, access remote files, and log on to remote machines. And they want to do this from anywhere on land, sea, or air. For example, at computer conferences these days, the organizers often set up a wireless network in the conference area. Anyone with a notebook computer and a wireless modem can just turn the computer on and be connected to the Internet, as though the computer were plugged into a wired network. Similarly, some universities have installed wireless networks on campus so students can sit under the trees and consult the library's card catalog or read their e-mail.

Wireless networks are of great value to fleets of trucks, taxis, delivery vehicles, and repair persons for keeping in contact with home. For example, in many cities, taxi drivers are independent businessmen, rather than being employees of a taxi company. In some of these cities, the taxis have a display the driver

can see. When a customer calls up, a central dispatcher types in the pickup and destination points. This information is displayed on the drivers' displays and a beep sounds. The first driver to hit a button on the display gets the call.

Although wireless networking and mobile computing are often related, they are not identical, as the below figure shows.

Wireless	Mobile	Applications
No	No	Desktop computers in offices
No	Yes	A note book computer used in a hotel room
Yes	No	Networks in older, unwired buildings.
Yes	No	Portable office, PDA for store inventory

Social Issues

The wide spread introduction of networking has introduced new social, ethical, and political problems. Let us just briefly mention a few of them; a thorough study would require a full book, at least. A popular feature of many networks is news groups or bulletin boards whereby people can exchange messages with like-minded individuals. As long as the subjects are restricted to technical topics or hobbies like gardening, not too many problems will arise.

The trouble comes when news groups are set up on topics that people actually care about, like politics, religion, or sex. Views posted to such groups may be deeply offensive to some people. Worse yet, they may not be politically correct. Further more, messages need not be limited to text. High-resolution color photographs and even short video clips can now easily be transmitted over computer networks. Some people take a live-and-let-live view, but others feel that posting certain material (e.g., attacks on particular countries or religions, pornography, etc.) is simply unacceptable and must be censored. Different countries have different and conflicting laws in this area. Thus, the debate rages.

People have sued network operators, claiming that they are responsible for the contents of what they carry, just as newspapers and magazines are. The inevitable response is that a network is like a telephone company or the post office and cannot be expected to police what its users say. Stronger yet, were network operators to censor messages, they would likely delete everything containing even the slightest possibility of them being sued, and thus violate their users' rights to free speech. It is probably safe to say that this debate will go on for a while.

1.3 COMPONENTS OF COMPUTER NETWORKS

Q5. Explain the basic components of computer networking.

Ans :

(Imp.)

Computer network components include the major parts that are needed to install a network both at the office and home level. Before delving into the installation process, you should be familiar with each part so that you could choose and buy the right component that fits with your network system.

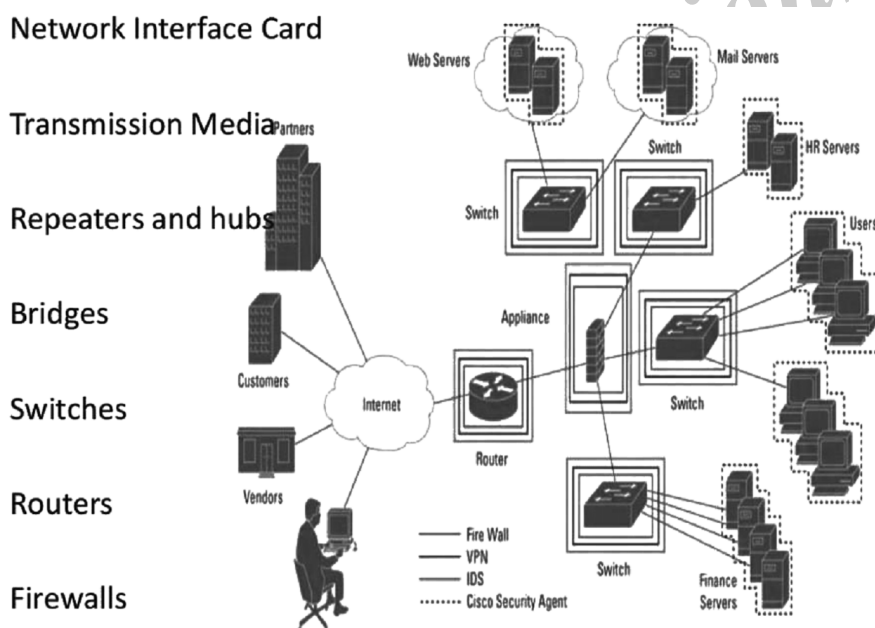
These hardware components include cable, Hub, Switch, NIC (network interface card), modem and router. Depending on the type of network you are going to install, some of the parts can be eliminated. For example, in a wireless network you don't need cables, hubs so on.

We will discuss about the main computer network components required to install simple computer network, often called LAN (local area network).

In computer network technology, there are several types of networks that range from simple to complex level. However, in any case in order to connect computers with each other or to the existing network or planning to install from scratch, the required devices and rules (protocols) are mostly the same.

Computer network requires the following devices (some of them are optional).

1. Network Interface Card (NIC)
2. Hub
3. Switches
4. Cables and connectors
5. Router
6. Modem



Network adapter is a device that enables a computer to talk with other computer/network. Using unique hardware addresses (MAC address) encoded on the card chip, the data-link protocol employs these addresses to discover other systems on the network so that it can transfer data to the right destination.

There are two types of network cards: wired and wireless. The wired NIC uses cables and connectors as a medium to transfer data, whereas in the wireless card, the connection is made using antenna that employs radio wave technology. All modern laptop computers incorporated wireless NIC in addition to the wired adapter.

Network Card Speed

Network Interface card, one of the main computer network components, comes with different speeds, 10Mbps, 100Mbps, and 1000Mbps, so on. Recent standard network cards built with Gigabit (1000Mbps) connection speed. It also supports to connect slower speeds such as 10Mbps and 100Mbps. However, the speed of the card depends on your LAN speed.

For example, if you have a switch that supports up to 100Mbps, your NIC will also transfer a data with this same speed even though your computer NIC has still the capability to transfer data at 1000Mbps (1Gbps). In modern computers, network adapter is integrated with a computer motherboard. However if you want advanced and fast Ethernet card, you may buy and install on your computer using the PCI slot found on the motherboard (desktop) and Express Card slots on laptop.

2. Hub

Hub is a device that splits a network connection into multiple computers. It is like a distribution center. When a computer request information from a network or a specific computer, it sends the request to the hub through a cable. The hub will receive the request and transmit it to the entire network.

Currently Hubs are becoming obsolete and replaced by more advanced communication devices such as Switchs and Routers.

3. Switches

Switch is a telecommunication device grouped as one of computer network components. Switch is like a Hub but built in with advanced features. It uses physical device addresses in each incoming messages so that it can deliver the message to the right destination or port.

Like Hub, switch don't broadcast the received message to entire network, rather before sending it checks to which system or port should the message be sent. In other words switch connects the source and

destination directly which increases the speed of the network. Both switch and hub have common features: Multiple RJ-45 ports, power supply and connection lights.

4. Cables and Connectors

Cable is one way of transmission media which can transmit communication signals. The wired network typology uses special type of cable to connect computers on a network.

There are a number of solid transmission Media types, which are listed below. - Twisted pair wire

It is classified as Category 1, 2, 3, 4, 5, 5E, 6 and 7. Category 5E, 6 and 7 are high-speed cables that can transmit 1Gbps or more.

Coaxial cable

Coaxial cable more resembles like TV installation cable. It is more expensive than twisted-pair cable but provide high data transmission speed.

Fiber-optic cable

It is a high-speed cable which transmits data using light beams through a glass bound fibers. Fiber-optic cable is high data transmission cable comparing to the other cable types. But the cost of fiber optics is very expensive which can only be purchased and installed on governmental level.

5. Router

When we talk about computer network components, the other device that used to connect a LAN with an internet connection is called Router. When you have two distinct networks (LANs) or want to share a single internet connection to multiple computers, we use a Router.

In most cases, recent routers also include a switch which in other words can be used as a switch. You don't need to buy both switch and router, particularly if you are installing small business and home networks.

There are two types of Router: wired and wireless. The choice depends on your physical office/home setting, speed and cost.

6. Modems

A modem enables you to connect your computer to the available internet connection over the existing telephone line. Like NIC, Modem is not integrated with a computer motherboard. It comes as separate part which can be installed on the PCI slots found on motherboard.

A modem is not necessary for LAN, but required for internet connection such as dial-up and DSL.

There are some types of modems, which differs in speed and transmission rate. Standard PC modem or Dial-up modems (56Kb data transmission speed), Cellular modem (used in a laptop that enables to connect while on the go), cable modem (500 times faster than standard modem) and DSL Modems are the most popular.

1.3.1 Advantages

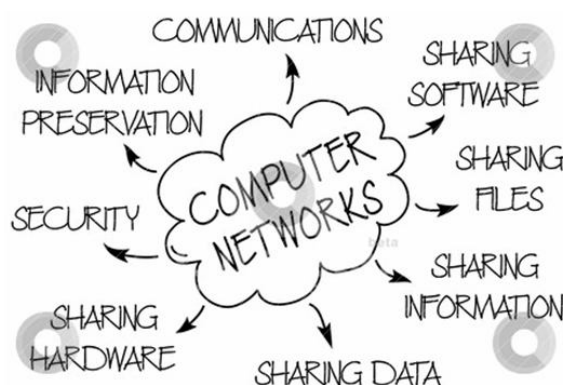
Q6. Explain the advantages of computer networking.

Ans :

(Imp.)

One of the delights of the digital era is widely spread ease of communication from one end to the vast distant location. We have seen a revolution in communication, digitalization, globalization, socialization, access to fast internet, video calling, wireless data transfer, high definition media content. It all becomes a possibility because of NETWORKING.

Few of the obvious advantages of networking that have apparently changed our lives for Good.



(i) Communication

The biggest leap forward in technology that we all witnessed in last decade is "Communication". Ease of communication is one of the biggest advantages of computer networking.

"To be precise, an effective and faster communication between computers, which enable us to communicate effectively"

Computer networking technology has improved the way of communication between people from the same or different organizations; they can communicate in a matter of seconds for collaborating the work activities. In offices, networked computers are serving as the backbone for the daily communication from top to bottom level of organization. Different types of software can be installed which are useful for transmitting messages and emails at fast speed.

Different types of software can be installed which are useful for transmitting messages and emails and video calls at fast speed. And that is all because of computer networking.

(ii) Data Sharing

Another wonderful advantage of computer networks is the data sharing. Data such as documents, file, accounts information, reports, presentation files, videos, images etc can be shared within a local network or remotely connected networks. Hardware and application sharing is also a possibility for organizations such as banks.

Back in the day, high-resolution images and videos that use to take hours to load on our local internet, now they can be shared, transferred from one location to another far more rapidly.

(iii) Instant and Multiple Accesses

One of the other advantages of computer networking is that it enables multiple users to access same data at the same time from a same or remote location. One of the real time examples is a world wide web. Everyone can

access a web page from a different location and read the same information at a same time.

It enables us to send different commands such as Print Command from a remote location. Computer networking gave us an ability of ease of accessibility. All these features couldn't have possible with advancements in computer networking.

(iv) Video Conferencing

Video calling is getting a norm in corporate culture. LAN and WN have made it possible for the organizations to communication on live video streaming from different geographical locations. It has reduced the cost of communicating between two entities.

(v) Internet Service

One high-speed internet connection is distributed among 100's of computers within the organization. Every computer can receive and send email, access World Wide Web and share data with other computers within the network. It is all because of the ability of the computer to communicate with a server to form a network.

(vi) Broad Casting

With the help of computer network, news, important messages, and information can be broadcasted in just a matter of seconds to the selected or entire group; it saves a lot of time and effort on a day to day task within an organization. People, can exchange messages immediately over the network any time or we can say 24 hour.

(vii) Cost Effective

Building a computer network, save a lot of cost for any organizations in different ways. Building "uplinks" through the computer networking, immediately transfers files and messages to the other people which reduced transportation and communication expense. It also raises the standard of the organization because of the advanced technologies that re used in networking.

(viii) Remote Access

Employees of organization can connect to the local network from their home to access and editing files. All they require is an interface or network remote IP or web portal to log in. Remote accessing is also possible for single users.

Remote accessing is also possible for single users. An individual student can access his home computer (Connected to internet) from his university computer to access data from home computer and vice versa.

(ix) Flexible

Computer network technology is quite flexible. Based on requirements of the organization, the network can be formed by choosing appropriate computer networking topology.

(x) Reliable

With proper defense mechanism is install using a firewall, antivirus programs etc, computer networks are considered reliable and safe. If one of the attached computers is collapsed, same data can be accessed from another computer that is connected to the same network.

(xi) Data Transmission

Data is transferred at the fast speed, even in the scenarios of one or two terminals machine fails to work properly. Data transmission in seldom affected in the computer networks. Almost complete communication can be achieved in critical scenarios too.

The number of advantages of computer networking mentioned above is just tip of an ice burg. Whereas, computer networking brings us its fruit in all aspects, without us realizing. It is just not limited to our homes or offices. It helps us in traveling with GPS, it helps us mobile communication, and all feilds of science.

1.4 TYPES OF NETWORKS

Q7. Classify different types of Networks.

Ans :

(Imp.)

There are about different types of networks which are used world wide these days, both in houses and commercially. These networks are used on the bases of their scale and scope, historical reasons, preferences for networking industries, and their design and implementation issues. LAN and WAN are mostly known and used widely. LAN, local area network was first invented for communication between two computers.

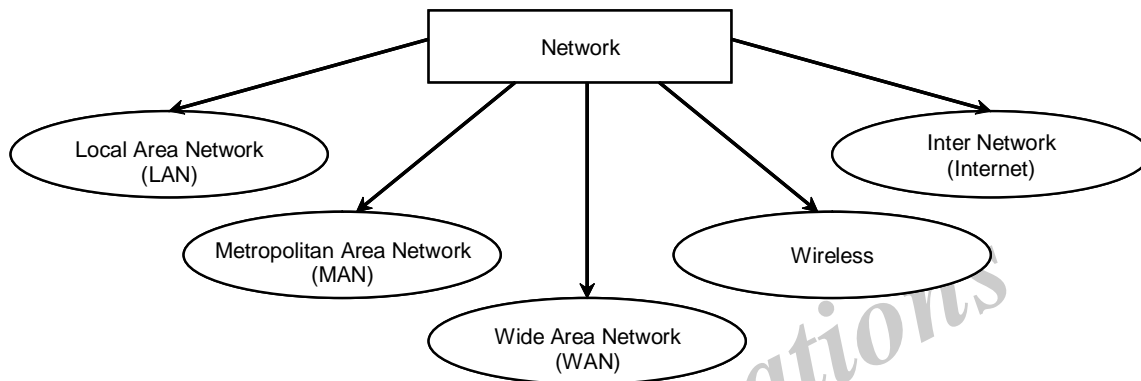


Fig. : Classification of Network

LAN operates through cables and network cards. Later WLAN, Wireless local area network was formed through LAN concept, there are no wires involved in communication between computers, and Wireless LAN cards are required to connect to wireless network. LAN is the original network out of which other networks are formed according to requirements. The network types are as follow.

1. LAN – Local Area Networks

LAN connects networking devices with in short spam of area, i.e. small offices, home, internet cafes etc. LAN uses TCP/IP network protocol for communication between computers. It is often but not always implemented as a single IP subnet. Since LAN is operated in short area so It can be control and administrate by single person or organization.

2. WAN - Wide Area Networks

As "word" Wide implies, WAN, wide area network cover large distance for communication between computers. The Internet itself is the biggest example of Wide area network, WAN, which is covering the entire earth. WAN is distributed collection of geographically LANs. A network connecting device router connects LANs to WANs. WAN used network protocols like ATM, X.25, and Frame Relay for long distance connectivity.

3. Wireless - Local Area Network

A LAN, local area networks based on wireless network technology mostly referred as Wi-Fi. Unlike LAN, in WLAN no wires are used, but radio signals are the medium for communication. Wireless network cards are required to be installed in the systems for accessing any wireless network around. Mostly wireless cards connect to wireless routers for communication among computers or accessing WAN, internet.

4. MAN - Metropolitan Area Network

This kind of network is not mostly used but it has its own importance for some government bodies and organizations on larger scale. MAN, metropolitan area network falls in middle of LAN and WAN, It covers large span of physical area than LAN but smaller than WAN, such as a city.

5. CAN - Campus Area Network

Networking spanning with multiple LANs but smaller than a Metropolitan area network, MAN. This kind of network mostly used in relatively large universities or local business offices and buildings.

6. SAN - Storage Area Network

SAN technology is used for data storage and it has no use for most of the organization but data oriented organizations. Storage area network connects servers to data storage devices by using Fiber channel technology.

7. SAN - System Area Network

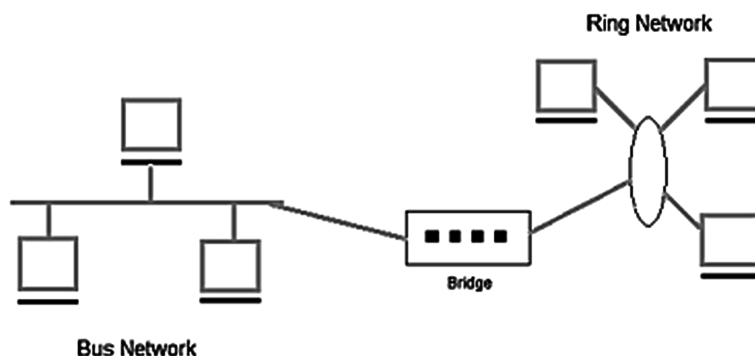
SAN, system area networks are also known as cluster area network and it connects high performance computers with high speed connections in cluster configuration.

1.4.1 Local Area Network (LAN)**Q8. Explain briefly about LAN.**

Ans :

It is also called LAN and designed for small physical areas such as an office, group of buildings or a factory. LANs are used widely as it is easy to design and to troubleshoot. Personal computers and workstations are connected to each other through LANs. We can use different types of topologies through LAN, these are Star, Ring, Bus, Tree etc.

LAN can be a simple network like connecting two computers, to share files and network among each other while it can also be as complex as interconnecting an entire building.



LAN networks are also widely used to share resources like printers, shared hard-drive etc.

Characteristics

1. LAN's are private networks, not subject to tariffs or other regulatory controls.
2. LAN's operate at relatively high speed when compared to the typical WAN.

3. There are different types of Media Access Control methods in a LAN, the prominent ones are Ethernet, Token ring.
4. It connects computers in a single building, block or campus, i.e. they work in a restricted geographical area.

Applications

1. One of the computer in a network can become a server serving all the remaining computers called clients. Software can be stored on the server and it can be used by the remaining clients.
2. Connecting Locally all the workstations in a building to let them communicate with each other locally without any internet access.
3. Sharing common resources like printers etc are some common applications of LAN.

Advantages

1. Resource Sharing

Computer resources like printers, modems, DVD-ROM drives and hard disks can be shared with the help of local area networks. This reduces cost and hardware purchases.

2. Software Applications Sharing

It is cheaper to use same software over network instead of purchasing separate licensed software for each client a network.

3. Easy and Cheap Communication

Data and messages can easily be transferred over networked computers.

4. Centralized Data

The data of all network users can be saved on hard disk of the server computer. This will help users to use any workstation in a network to access their data. Because data is not stored on workstations locally.

5. Data Security

Since, data is stored on server computer centrally, it will be easy to manage data at only one place and the data will be more secure too.

6. Internet Sharing

Local Area Network provides the facility to share a single internet connection among all the LAN users. In Net Cafes, single internet connection sharing system keeps the internet expenses cheaper.

Disadvantages

1. High Setup Cost

Although the LAN will save cost over time due to shared computer resources, but the initial setup costs of installing Local Area Networks is high.

2. Privacy Violations

The LAN administrator has the rights to check personal data files of each and every LAN user. Moreover he can check the internet history and computer use history of the LAN user.

3. Data Security Threat

Unauthorised users can access important data of an organization if centralized data repository is not secured properly by the LAN administrator.

4. LAN Maintenance Job

Local Area Network requires a LAN Administrator because, there are problems of software installations or hardware failures or cable disturbances in Local Area Network. A LAN Administrator is needed at this full time job.

5. Covers Limited Area

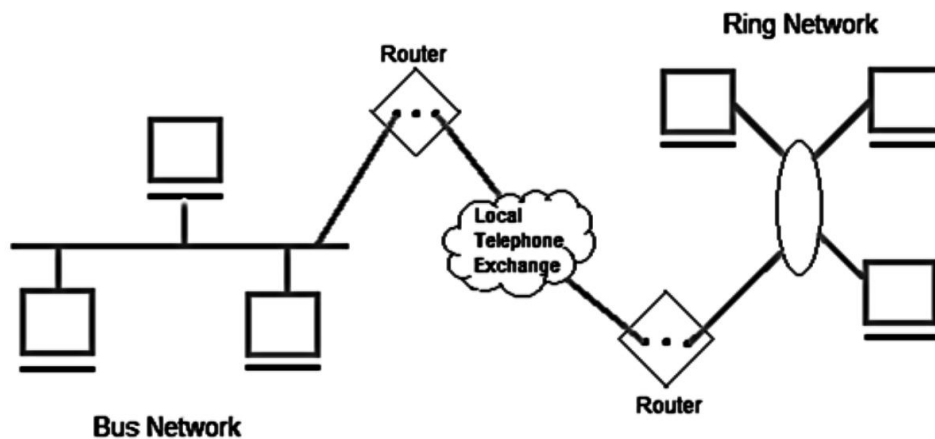
Local Area Network covers a small area like one office, one building or a group of nearby buildings.

1.4.2 Metropolitan Area Network (MAN)

Q9. Explain briefly about MAN.

Ans :

It was developed in 1980s. It is basically a bigger version of LAN. It is also called MAN and uses the similar technology as LAN. It is designed to extend over the entire city. It can be means to connecting a number of LANs into a larger network or it can be a single cable. It is mainly hold and operated by single private company or a public company.



Characteristics

1. It generally covers towns and cities (50 km).
2. Communication medium used for MAN are optical fibers, cables etc.
3. Data rates adequate for distributed computing applications.

Advantages

1. Extremely efficient and provide fast communication via high-speed carriers, such as fibre optic cables.
2. It provides a good back bone for large network and provides greater access to WANs.
3. The dual bus used in MAN helps the transmission of data in both directions simultaneously.
4. A MAN usually encompasses several blocks of a city or an entire city.

Disadvantages

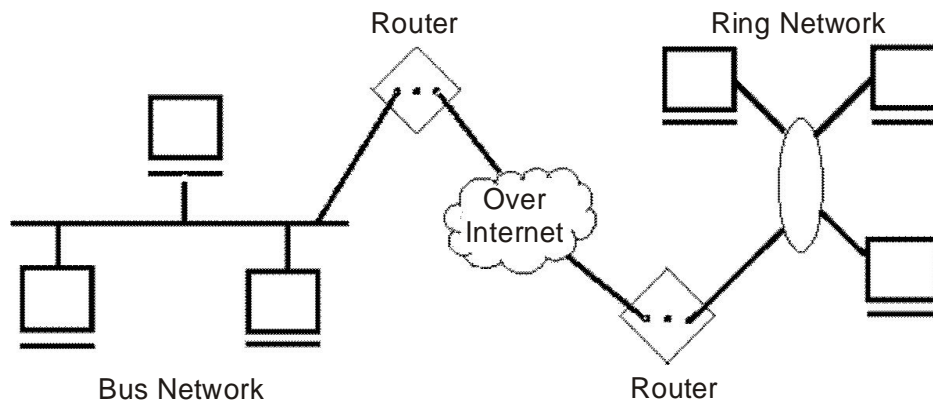
1. More cable required for a MAN connection from one place to another.
2. It is difficult to make the system secure from hackers and industrial espionage (spying) graphical regions.

1.4.3 Wide Area Network (WAN)

Q10. Explain briefly about Wide Area Network.

Ans :

It is also called WAN. WAN can be private or it can be public leased network. It is used for the network that covers large distance such as cover states of a country. It is not easy to design and maintain. Communication medium used by WAN are PSTN or Satellite links. WAN operates on low data rates.



Characteristics

1. It generally covers large distances (states, countries, continents).
2. Communication medium used are satellite, public telephone networks which are connected by routers.

Advantages

1. Covers a large geographical area so long distance business can connect on the one network.
2. Shares software and resources with connecting workstations.
3. Messages can be sent very quickly to anyone else on the network. These messages can have picture, sounds or data included with them (called attachments).
4. Expensive things (such as printers or phone lines to the internet) can be shared by all the computers on the network without having to buy a different peripheral for each computer.
5. Everyone on the network can use the same data. This avoids problems where some users may have older information than others.

Disadvantages

1. Need a good firewall to restrict outsiders from entering and disrupting the network.
2. Setting up a network can be an expensive, slow and complicated. The bigger the network the more expensive it is.
3. Once set up, maintaining a network is a full-time job which requires network supervisors and technicians to be employed.
4. Security is a real issue when many different people have the ability to use information from other computers. Protection against hackers and viruses adds more complexity and expense.

1.4.4 Wireless Network

Q11. Explain about Wireless Network.

OR

Define key requirements and functions of Wireless LANs.

Ans :

Digital wireless communication is not a new idea. Earlier, Morse code was used to implement wireless networks. Modern digital wireless systems have better performance, but the basic idea is the same.

Wireless Networks can be divided into three main categories:

1. System interconnection
2. Wireless LANs
3. Wireless WANs

1. System Interconnection

System interconnection is all about interconnecting the components of a computer using short-range radio. Some companies got together to design a short-range wireless network called Bluetooth to connect various components such as monitor, keyboard, mouse and printer, to the main unit, without wires. Bluetooth also allows digital cameras, headsets, scanners and other devices to connect to a computer by merely being brought within range.

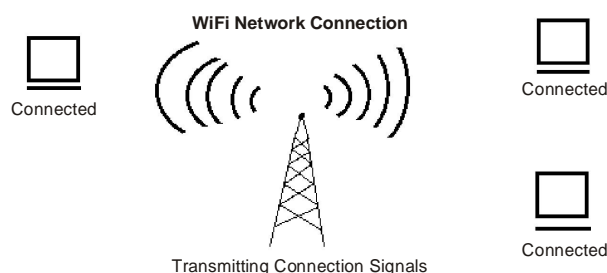
In simplest form, system interconnection networks use the master-slave concept. The system unit is normally the master, talking to the mouse, keyboard, etc. as slaves.

2. Wireless LANs

System interconnection is all about interconnecting the components of a computer using short-range radio. Some companies got together to design a short-range wireless network called Bluetooth to connect various components such as monitor, keyboard, mouse and printer, to the main unit, without wires. Bluetooth also allows

digital cameras, headsets, scanners and other devices to connect to a computer by merely being brought within range.

In simplest form, system interconnection networks use the master-slave concept. The system unit is normally the master, talking to the mouse, keyboard, etc. as slaves.



These are the systems in which every computer has a radio modem and antenna with which it can communicate with other systems. Wireless LANs are becoming increasingly common in small offices and homes, where installing Ethernet is considered too much trouble. There is a standard for wireless LANs called IEEE 802.11, which most systems implement and which is becoming very widespread.

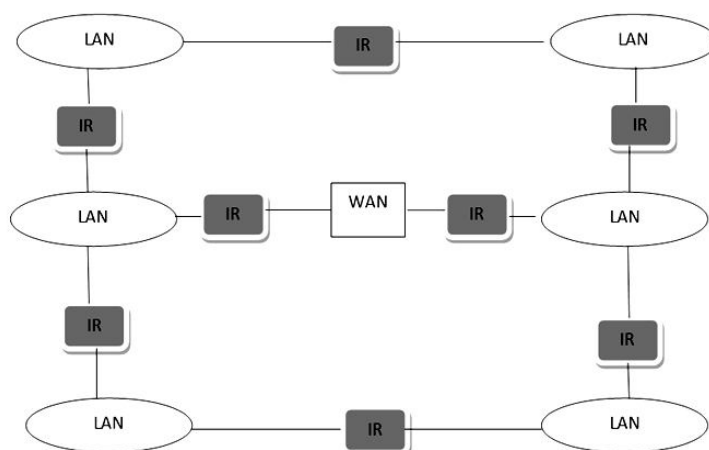
3. Wireless WANs

The radio network used for cellular telephones is an example of a low-bandwidth wireless WAN. This system has already gone through three generations.

1. The first generation was analog and for voice only.
2. The second generation was digital and for voice only.
3. The third generation is digital and is for both voice and data.

Inter Network

Inter Network or Internet is a combination of two or more networks. Inter network can be formed by joining two or more individual networks by means of various devices such as routers, gateways and bridges.



1.4.4.1 Connection Oriented and Connection less Services

Q12. Explain different types of services by computer networks.

OR

Explain connection oriented and connection less networks.

Ans :

(Imp.)

These are the two services given by the layers to layers above them. These services are :

1. Connection Oriented Service
2. Connection less Services

1. Connection Oriented Services

There is a sequence of operation to be followed by the users of connection oriented service. These are:

- (i) Connection is established
- (ii) Information is sent
- (iii) Connection is released

In connection oriented service we have to establish a connection before starting the communication. When connection is established we send the message or the information and then we release the connection.

Connection oriented service is more reliable than connection less service. We can send the message in connection oriented service if there is an error at the receivers end. Example of connection oriented is TCP (Transmission Control Protocol) protocol.

2. Connection Less Services

It is similar to the postal services, as it carries the full address where the message (letter) is to be carried. Each message is routed independently from source to destination. The order of message sent can be different from the order received.

In connectionless the data is transferred in one direction from source to destination without checking that destination is still there or not or if it prepared to accept the message. Authentication is not needed in this. Example of Connectionless service is UDP (User Datagram Protocol) protocol.

Difference between Connection oriented service and Connectionless service

1. In connection oriented service authentication is needed while connectionless service does not need any authentication.
2. Connection oriented protocol makes a connection and checks whether message is received or not and sends again if an error occurs connectionless service protocol does not guarantees a delivery.
3. Connection oriented service is more reliable than connectionless service.
4. Connection oriented service interface is stream based and connectionless is message based.

Service Primitives

A service is formally specified by a set of primitives (operations) available to a user process to access the service. These primitives tell the service to perform some action or report on an action taken by a peer entity. If the protocol stack is located in the operating system, as it often is, the primitives are normally system calls. These calls cause a trap to kernel mode, which then turns control of the machine over to the operating system to send the necessary packets. The set of primitives available depends on the nature of the service being provided. The primitives for connection-oriented service are different from those of connection-less service.

There are five types of service primitives:

1. **LISTEN:** When a server is ready to accept an incoming connection it executes the LISTEN primitive. It blocks waiting for an incoming connection.
2. **CONNECT:** It connects the server by establishing a connection. Response is awaited.
3. **RECIEVE:** Then the RECIEVE call blocks the server.
4. **SEND:** Then the client executes SEND primitive to transmit its request followed by the execution of RECIEVE to get the reply. Send the message.
5. **DISCONNECT:** This primitive is used for terminating the connection. After this primitive one can't send any message. When the client sends DISCONNECT packet then the server also sends the DISCONNECT packet to acknowledge the client. When the server package is received by client then the process is terminated.

Connection Oriented Service Primitives

There are 5 types of primitives for Connection Oriented Service :

LISTEN	Block waiting for an incoming connection
CONNECTION	Establish a connection with a waiting peer
RECEIVE	Block waiting for an incoming message
SEND	Sending a message to the peer
DISCONNECT	Terminate a connection

Connectionless Oriented Service Primitives

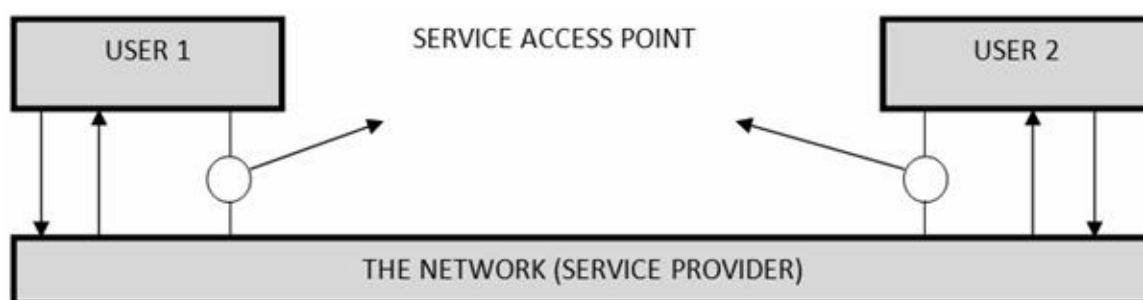
There are 4 types of primitives for Connectionless Oriented Service:

UNIDATA	This primitive sends a packet of data
FACILITY, REPORT	Primitive for enquiring about the performance of the network, like delivery statistics.

Relationship of Services to Protocol

Services

These are the operations that a layer can provide to the layer above it. It defines the operation and states a layer is ready to perform but it does not specify anything about the implementation of these operations.



Protocols

These are set of rules that govern the format and meaning of frames, messages or packets that are exchanged between the server and client.

1.5 OSI REFERENCE MODEL

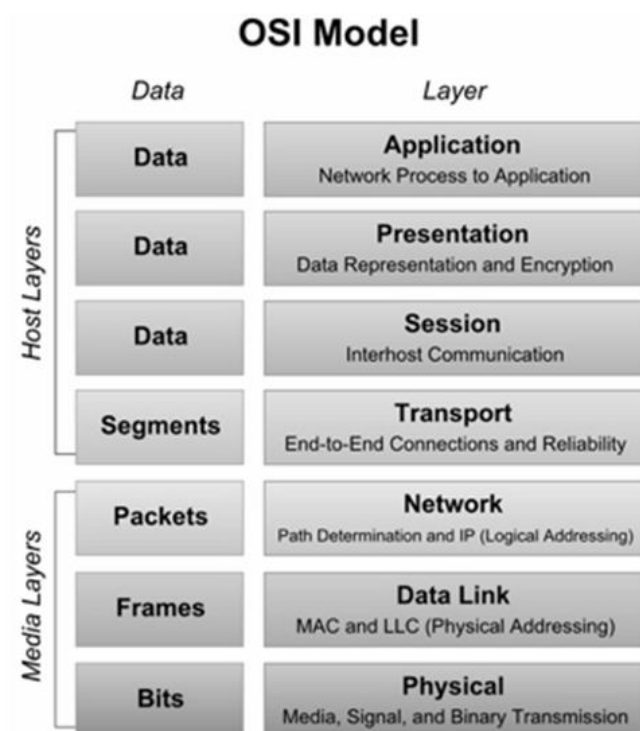
Q13. What is OSI reference model?

Ans :

(Imp.)

There are n numbers of users who use computer network and are located over the world. So to ensure, national and worldwide data communication, systems must be developed which are compatible to communicate with each other ISO has developed a standard. ISO stands for International organization of Standardization. This is called a model for Open System Interconnection (OSI) and is commonly known as OSI model.

The ISO-OSI model is seven layer architecture. It defines seven layers or levels in a complete communication system.

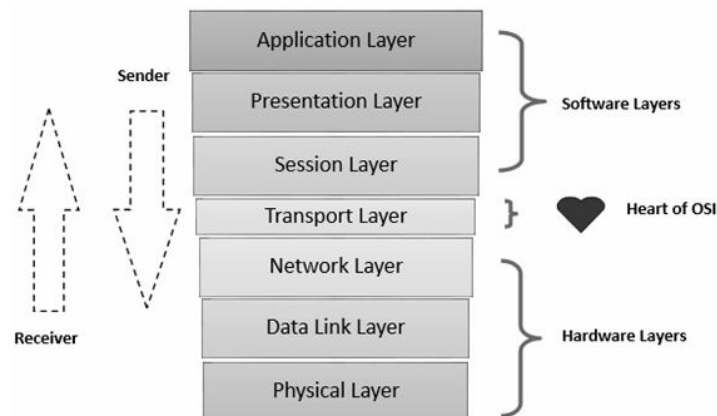


Layer	Name of Protocol	Name of Unit Exchanged
Application	Application Protocol	APDU-Application Protocol Data Unit
Presentation	Presentation Protocol	PPDU - Presentation Protocol Data Unit
Session	Session Protocol	SPDU - Session Protocol Data Unit
Transport	Transport Protocol	TPDU - Transport Protocol Data Unit
Network	Network layer host-router Protocol	Packet
Data Link	Data link layer host-router Protocol	Frame
Physical	Physical layer host-router Protocol	Bit

Principles of OSI Reference Model

The OSI reference model has 7 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 architecture does not become unwieldy.



1. Physical Layer (Layer 1)

The lowest layer of the OSI reference model is the physical layer. It is responsible for the actual physical connection between the devices. The physical layer contains information in the form of bits. It is responsible for the actual physical connection between the devices. When receiving data, this layer will get the signal received and convert it into 0s and 1s and send them to the Data Link layer, which will put the frame back together.

1100 0111 0011

The functions of the physical layer are:

- (i) **Bit Synchronization:** The physical layer provides the synchronization of the bits by providing a clock. This clock controls both sender and receiver thus providing synchronization at bit level.
- (ii) **Bit rate Control:** The Physical layer also defines the transmission rate i.e. the number of bits sent per second.
- (iii) **Physical Topologies:** Physical layer specifies the way in which the different, devices/nodes are arranged in a network i.e. bus, star or mesh topology.
- (iv) **Transmission Mode:** Physical layer also defines the way in which the data flows between the two connected devices. The various transmission modes possible are: Simplex, half-duplex and full-duplex.

Note: Hub, Repeater, Modem, Cables are Physical Layer devices.

Note: Network Layer, Data Link Layer and Physical Layer are also known as **Lower Layers** or **Hardware Layers**.

2. Data Link Layer (DLL) (Layer 2)

The data link layer is responsible for the node to node delivery of the message. The main function of this layer is to make sure data transfer is error free from one node to another, over the physical layer. When a packet arrives in a network, it is the responsibility of DLL to transmit it to the Host using its MAC address.

Data Link Layer is divided into two sub layers

- (i) Logical Link Control (LLC)
- (ii) Media Access Control (MAC)

Packet received from Network layer is further divided into frames depending on the frame size of NIC (Network Interface Card). DLL also encapsulates Sender and Receiver's MAC address in the header.

The Receiver's MAC address is obtained by placing an ARP (Address Resolution Protocol) request onto the wire asking "Who has that IP address?" and the destination host will reply with its MAC address.



The functions of the data Link layer are :

- (i) **Framing:** Framing is a function of the data link layer. It provides a way for a sender to transmit a set of bits that are meaningful to the receiver. This can be accomplished by attaching special bit patterns to the beginning and end of the frame.
- (ii) **Physical Addressing:** After creating frames, Data link layer adds physical addresses (MAC address) of sender and/or receiver in the header of each frame.
- (iii) **Error Control:** Data link layer provides the mechanism of error control in which it detects and retransmits damaged or lost frames.
- (iv) **Flow Control:** The data rate must be constant on both sides else the data may get corrupted thus , flow control coordinates that amount of data that can be sent before receiving acknowledgment.
- (v) **Access Control:** When a single communication channel is shared by multiple devices, MAC sub-layer of data link layer helps to determine which device has control over the channel at a given time.

* Packet in Data Link layer is referred as **Frame**.

** Data Link layer is handled by the NIC (Network Interface Card) and device drivers of host machines.

*** *Switch & Bridge are Data Link Layer devices.*

3. Network Layer (Layer 3)

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

The functions of the Network layer are :

- (i) **Routing:** The network layer protocols determine which route is suitable from source to destination. This function of network layer is known as routing.
- (ii) **Logical Addressing:** In order to identify each device on internetwork uniquely, network layer defines an addressing scheme. The sender & receiver's IP address are placed in the header by network layer. Such an address distinguishes each device uniquely and universally.

* *Segment* in Network layer is referred as **Packet**.

** Network layer is implemented by networking devices such as routers.

4. Transport Layer (Layer 4)

Transport layer provides services to application layer and takes services from network layer. The data in the transport layer is referred to as *Segments*. It is responsible for the End to End delivery of the complete message. Transport layer also provides the acknowledgment of the successful data transmission and re-transmits the data if error is found.

At sender's Side

Transport layer receives the formatted data from the upper layers, performs **Segmentation** and also implements **Flow & Error control** to ensure proper data transmission. It also adds Source and Destination port number in its header and forwards the segmented data to the Network Layer.

Note: The sender need to know the port number associated with the receiver's application.

Generally this destination port number is configured, either by default or manually. For example, when a web application makes a request to a web server, it typically uses port number 80, because this is the default port assigned to web applications. Many applications have default port assigned.

At Receiver's Side

Transport Layer reads the port number from its header and forwards the Data which it has received to the respective application. It also performs sequencing and reassembling of the segmented data.

The functions of the transport layer are :

I. Segmentation and Reassembly

This layer accepts the message from the (session) layer, breaks the message into smaller units. Each of the segment produced has a header associated with it. The transport layer at the destination station reassembles the message.

II. Service Point Addressing

In order to deliver the message to correct process, transport layer header includes a type of address called service point address or port address. Thus by specifying this address, transport layer makes sure that the message is delivered to the correct process.

The services provided by transport layer :

Connection Oriented Service

It is a three phase process which include

- Connection Establishment
- Data Transfer
- Termination/disconnection

In this type of transmission the receiving device sends an acknowledgment, back to the source after a packet or group of packet is received. This type of transmission is reliable and secure.

Connection Less Service

It is a one phase process and includes Data Transfer. In this type of transmission the receiver does not acknowledge receipt of a packet. This approach allows for much faster communication between devices. Connection oriented Service is more reliable than connection less Service.

* Data in the Transport Layer is called as Segments.

** Transport layer is operated by the Operating System. It is a part of the OS and communicates with the Application Layer by making system calls.

Transport Layer is called as Heart of OSI model.

5. Session Layer (Layer 5)

This layer is responsible for establishment of connection, maintenance of sessions, authentication and also ensures security.

The functions of the session layer are :

(i) **Session establishment, maintenance and termination:** The layer allows the two processes to establish, use and terminate a connection.

(ii) **Synchronization:** This layer allows a process to add checkpoints which are considered as synchronization points into the data. These synchronization point help to identify the error so that the data is re-synchronized properly, and ends of the messages are not cut prematurely and data loss is avoided.

(iii) **Dialog Controller:** The session layer determines which device will communicate first and the amount of data that will be sent.

**All the above 3 layers are integrated as a single layer in TCP/IP model as "Application Layer".

**Implementation of above 3 layers is done by the network application itself. These are also known as Upper Layers or Software Layers.

6. Presentation Layer (Layer 6)

Presentation layer is also called the Translation layer. The data from the application layer is extracted here and manipulated as per the required format to transmit over the network.

The functions of the presentation layer are :

(i) **Translation:** For example, ASCII to EBCDIC.

(ii) Encryption/ Decryption: Data encryption translates the data into another form or code. The encrypted data is known as the cipher text and the decrypted data is known as plain text. A key value is used for encrypting as well as decrypting data.

(iii) Compression: Reduces the number of bits that need to be transmitted on the network.

7. Application Layer (Layer 7)

At the very top of the OSI Reference Model stack of layers, we find Application layer which is implemented by the network applications. These applications produce the data, which has to be transferred over the network. This layer also serves as window for the application services to access the network and for displaying the received information to the user.

Example :

Application - Browsers, Skype Messenger etc.

****Application Layer is also called as Desktop Layer.**

The functions of the Application layer are :

- (i) Network Virtual Terminal
- (ii) FTAM-File transfer access and management
- (iii) Mail Services
- (iv) Directory Services

OSI model acts as a reference model and is not implemented in Internet because of its late invention. Current model being used is the TCP/IP model.

Merits of OSI reference Model

- (i) OSI model distinguishes well between the services, interfaces and protocols.
- (ii) Protocols of OSI model are very well hidden.
- (iii) Protocols can be replaced by new protocols as technology changes.
- (iv) Supports connection oriented services as well as connectionless service.

Demerits of OSI Reference Model

- (i) Model was devised before the invention of protocols.
- (ii) Fitting of protocols is tedious task.
- (iii) It is just used as a reference model.

1.5.1 TCP/IP

Q14. What is mean by TCP/IP?

Ans :

(Imp.)

TCP/IP is shorthand for a suite of protocols that run on top of IP. IP is the Internet Protocol, and TCP is the most important protocol that runs on top of IP. Any application that can communicate over the Internet is using IP, and these days most internal networks are also based on TCP/IP.

Protocols that run on top of IP include: TCP, UDP and ICMP. Most TCP/IP implementations support all three of these protocols. We'll talk more about them later.

Protocols that run underneath IP include: SLIP and PPP. These protocols allow IP to run across telecommunications lines.

TCP/IP protocols work together to break data into packets that can be routed efficiently by the network. In addition to the data, packets contain addressing, sequencing, and error checking information. This allows TCP/IP to accurately reconstruct the data at the other end.

Here's an analogy of what TCP/IP does. Say you're moving across the country. You pack your boxes and put your new address on them. The moving company picks them up, makes a list of the boxes, and ships them across the country using the most efficient route. That might even mean putting different boxes on different trucks. When the boxes arrive at your new home, you check the list to make sure everything has arrived (and in good shape), and then you unpack the boxes and "reassemble" your house.

- A suite of protocols
- Rules that dictate how packets of information are sent across - multiple networks
- Addressing
- Error checking

IP

Every computer on the Internet has at least one address that uniquely identifies it from all other computers on the Internet (aptly called it's IP address!). When you send or receive data - say an email message or web page - the message gets divided into little chunks called packets or data grams. Each of these packets contains both the source IP address and the destination IP address.

IP looks at the destination address to decide what to do next. If the destination is on the local network, IP delivers the packet directly. If the destination is not on the local network, then IP passes the packet to a gateway - usually a router.

Computers usually have a single default gateway. Routers frequently have several gateways from which to choose. A packet may get passed through several gateways before reaching one that is on a local network with the destination.

Along the way, any router may break the IP packet into several smaller packets based on transmission medium. For example, Ethernet usually allows packets of up to 1500 bytes, but it is not uncommon for modem-based PPP connections to only allow packets of 256 bytes. The last system in the chain (the destination) reassembles the original IP packet.

21 → FTP - File Transfer Protocol

23 → Telnet

25 → SMTP - Simple Mail Transfer Protocol

37 → Time

69 → TFTP - Trivial File Transfer Protocol

79 → Finger

103 → X400

161 → SNMP - Simple Network Management Protocol

162 → SNMPTRAP

After TCP/IP was invented and deployed, the OSI layered network model was accepted as a standard. OSI neatly divides network protocols into seven layers; the bottom four layers are shown in this diagram. The idea was that TCP/IP was an interesting experiment, but that it would be replaced by protocols based on the OSI model.

As it turned out, TCP/IP grew like wildfire, and OSI-based protocols only caught on in certain segments of the manufacturing community. These days, while everyone uses TCP/IP, it is common to use the OSI vocabulary.

1.5.2 TCP/IP Reference Model

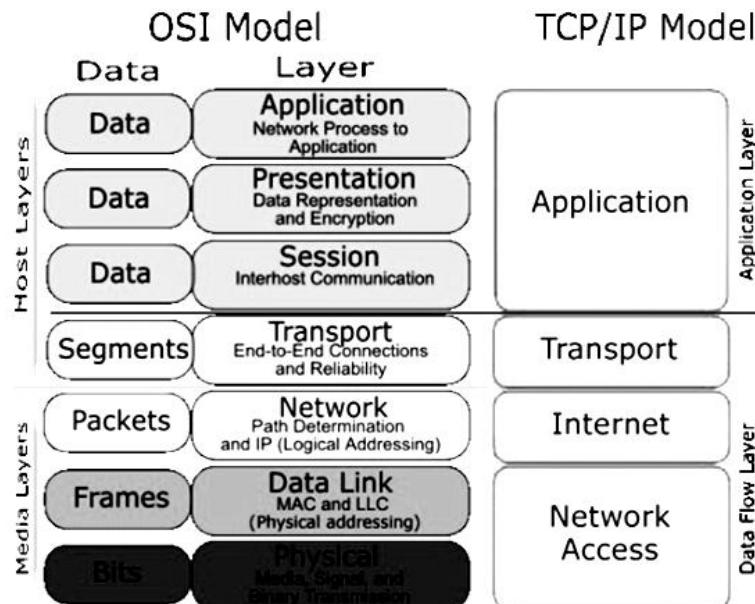
Q15. Explain layering architecture of TCP/IP model.

Ans :

(Imp.)

TCP/IP means Transmission Control Protocol and Internet Protocol. It is the network model used in the current Internet architecture as well. Protocols are set of rules which govern every possible communication over a network. These protocols describe the movement of data between the source and destination or the internet. These protocols offer simple naming and addressing schemes. The TCP/IP model is a concise version of the OSI model. It contains four layers, unlike seven layers in the OSI model. The layers are:

1. Process/Application Layer
2. Host-to-Host/Transport Layer
3. Internet Layer
4. Network Access/Link Layer



Protocols and networks in the TCP/IP model initially:

TCP/IP that is Transmission Control Protocol and Internet Protocol was developed by Department of Defence's Project Research Agency (ARPA, later DARPA) as a part of a research project of network interconnection to connect remote machines.

The features that stood out during the research, which led to making the TCP/IP reference model were:

- Support for a flexible architecture. Adding more machines to a network was easy.
- The network was robust, and connections remained intact until the source and destination machines were functioning.

The overall idea was to allow one application on one computer to talk to (send data packets) another application running on different computer.

The first layer is the Process layer on the behalf of sender and Network Access layer on the behalf of receiver.

1. Network Access Layer

This layer corresponds to the combination of Data Link Layer and Physical Layer of the OSI model. It looks out for hardware addressing and the protocols present in this layer allows for physical transmission of data.

2. Internet Layer

This layer parallels the functions of OSI's Network layer. It defines the protocols which are responsible for logical transmission of data over the entire network. The main protocols residing at this layer are:

- (i) **IP:** Stands for Internet Protocol and it is responsible for delivering packets from the source host to the destination host by looking at the IP addresses in the packet headers. IP has 2 versions:
- (ii) **IPv4 and IPv6:** IPv4 is the one that most of the websites are using currently. But IPv6 is growing as the number of IPv4 addresses is limited in number when compared to the number of users.
- (iii) **ICMP:** Stands for Internet Control Message Protocol. It is encapsulated within IP datagrams and is responsible for providing hosts with information about network problems.
- (iv) **ARP:** Stands for Address Resolution Protocol. It's job is to find the hardware address of a host from a known IP address. ARP has several types: Reverse ARP, Proxy ARP, Gratuitous ARP and Inverse ARP.

3. Host-to-Host Layer

This layer is analogous to the transport layer of the OSI model. It is responsible for end-to-end communication and error-free delivery of data. It shields the upper-layer applications from the complexities of data. The two main protocols present in this layer are:

- (i) **Transmission Control Protocol (TCP):** It is known to provide reliable and error-free communication between end systems. It performs sequencing and segmentation of data. It also has acknowledgment feature and controls the flow of the data through flow control mechanism. It is a very effective protocol but has a lot of overhead due to such features. Increased overhead leads to increased cost.
- (ii) **User Datagram Protocol (UDP):** On the other hand does not provide any such features. It is the go to protocol if your application does not require reliable transport as it is very cost-effective. Unlike TCP, which is connection-oriented protocol, UDP is connectionless.

4. Process Layer

This layer performs the functions of top three layers of the OSI model: Application, Presentation and Session Layer. It is responsible for node-to-node communication and controls user-interface specifications. Some of the protocols present in this layer are : HTTP, HTTPS, FTP, TFTP, Telnet, SSH, SMTP, SNMP, NTP, DNS, DHCP, NFS, X Window, LPD. Have a look at Protocols in Application Layer for some information about these protocols. Protocols other than those present in the linked article are :

- (i) **HTTP and HTTPS :** HTTP stands for Hyper-text transfer protocol. It is used by the World Wide Web to manage communications between web browsers and servers. HTTPS stands for HTTP-Secure. It is a combination of HTTP with SSL(Secure Socket Layer). It is efficient in cases where the browser need to fill out forms, sign in, authenticate and carry out bank transactions.
- (ii) **SSH :** SSH stands for Secure Shell. It is terminal emulations software similar to Telnet. The reason SSH is more preferred is because of its ability to maintain encrypted connection. It sets up a secure session over a TCP/IP connection.
- (iii) **NTP :** NTP stands for Network Time Protocol. It is used to synchronize the clocks on our computer to one standard time source. It is very useful in situations like bank transactions. Assume the following situation without the presence of NTP. Suppose you carry out a transaction, where your computer reads the time at 2:30 PM while the server records it at 2:28 PM. The server can crash very badly if it's out of sync.

Merits of TCP/IP Model

1. It operated independently.
2. It is scalable.
3. Client/server architecture.
4. Supports a number of routing protocols.
5. Can be used to establish a connection between two computers.

Demerits of TCP/IP

1. In this, the transport layer does not guarantee delivery of packets.
2. The model cannot be used in any other application.
3. Replacing protocol is not easy.
4. It has not clearly separated its services, interfaces and protocols.

1.5.2.1 Similarities between OSI Reference and TCP/IP Reference Model**Q16. Explain similarities and distinguish between TCP/IP and OSI Model.***Ans :***(Imp.)****Similarities**

Following are some similarities between OSI Reference Model and TCP/IP Reference Model.

1. Both are layered architecture.
2. Layers provide similar functionalities.
3. Both are protocol stack.
4. Both are reference models.

Comparison of OSI Reference Model and TCP/IP Reference Model

Following are some major differences between OSI Reference Model and TCP/IP Reference Model, with diagrammatic comparison below.

S.No.	OSI(Open System Interconnection)	TCP/IP(Transmission Control Protocol / Internet Protocol)
1	OSI is a generic, protocol independent standard, acting as a communication gateway between the network and end user.	TCP/IP model is based on standard protocols around which the Internet has developed. It is a communication protocol, which allows connection of hosts over a network.
2	In OSI model the transport layer guarantees the delivery of packets.	In TCP/IP model the transport layer does not guarantees delivery of packets. Still the TCP/IP model is more reliable.
3	Follows vertical approach.	Follows horizontal approach.
4	OSI model has a separate Presentation layer and Session layer.	TCP/IP does not have a separate Presentation layer or Session layer.
5	Transport Layer is Connection Oriented.	Transport Layer is both Connection Oriented and Connection less.
6	Network Layer is both Connection Oriented and Connection less.	Network Layer is Connection less.
7	OSI is a reference model around which the networks are built. Generally it is used as a guidance tool.	TCP/IP model is, in a way implementation of the OSI model.
8	Network layer of OSI model provides both connection oriented and connectionless service.	The Network layer in TCP/IP model provides connectionless service.
9	OSI model has a problem of fitting the protocols into the model.	TCP/IP model does not fit any protocol
10	Protocols are hidden in OSI model and are easily replaced as the technology changes.	In TCP/IP replacing protocol is not easy.
11	OSI model defines services, interfaces and protocols very clearly and makes clear distinction between them. It is protocol independent.	In TCP/IP, services, interfaces and protocols are not clearly separated. It is also protocol dependent.
12	It has 7 layers	It has 4 layers

1.6 NETWORK SOFTWARE

Q17. Define Network software.

Ans :

The first computer networks were designed with the hardware as the main concern and the software as an afterthought. This strategy no longer works. Network software is now highly structured. To reduce their design complexity, most networks are organized as a series or hierarchy of layers or levels. The number of layers, the name of each layer, the contents of each layer, and the function of each layer differ from network to network. Layer n on one machine communicates with layer n on another machine on the network using a some rules known as the layer n protocol.

A protocol is an agreement between the communicating parties on how the communication is to proceed.

The entities comprising the corresponding layers on two communicating machines over the network are called peers.

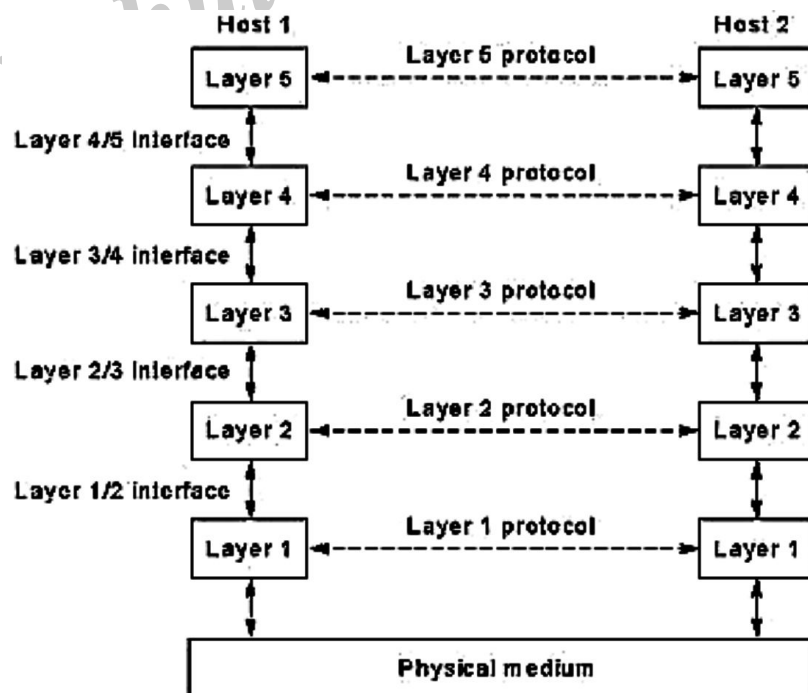
In reality, no data is transferred from layer n on any two machines. Instead, each data and control information is passed to the layer below.

Additional information including protocol control information may be appended by each layer to data as it travels from higher to lower layers in the form of layer headers.

Below layer 1 is the physical medium through which actual communication occur over communication channels.

Between each pair of adjacent layers there is an interface. The interface defines which primitive operations and services the lower layer offers to the upper layer.

The set of layers and associated protocols is called network architecture.



1.6.1 Issues**Q18. What are the design issues for the Layers?**

Ans : (Imp.)

The following are the design issues for the layers:

1. Reliability

It is a design issue of making a network that operates correctly even when it is made up of unreliable components.

2. Addressing

There are multiple processes running on one machine. Every layer needs a mechanism to identify senders and receivers.

3. Error Control

It is an important issue because physical communication circuits are not perfect. Many error detecting and error correcting codes are available. Both sending and receiving ends must agree to use any one code.

4. Flow Control

If there is a fast sender at one end sending data to a slow receiver, then there must be flow control mechanism to control the loss of data by slow receivers. There are several mechanisms used for flow control such as increasing buffer size at receivers, slow down the fast sender, and so on. Some process will not be in position to accept arbitrarily long messages. This property leads to mechanisms for disassembling, transmitting and the reassembling messages.

5. Multiplexing and De-multiplexing

If the data has to be transmitted on transmission media separately, it is inconvenient or expensive to setup separate connection for each pair of communicating processes. So, multiplexing is needed in the physical layer at sender end and de-multiplexing is need at the receiver end.

6. Scalability

When network gets large, new problem arises. Thus scalability is important so that network can continue to work well when it gets large.

7. Routing

When there are multiple paths between source and destination, only one route must be chosen. This decision is made on the basis of several routing algorithms, which chooses optimized route to the destination.

8. Confidentiality and Integrity

Network security is the most important factor. Mechanisms that provide confidentiality defend against threats like eavesdropping. Mechanisms for integrity prevent faulty changes to messages.

Addressing Level

Level in architecture at which entity is named

1. Unique address for each end system (computer) and each intermediate system (router)
2. Network level address
3. IP or internet address (TCP/IP)
4. Network service access point or NSAP (OSI)
5. Process within the system
6. Port number (TCP/IP)
7. Service access point or SAP

Addressing Scope**1. Global non-ambiguity**

Global address identifies unique system There is only one system with address X

2. Global Applicability

It is possible at any system (any address) to identify any other system (address) by the global address of the other system Address X identifies that system from anywhere on the network

E.g. MAC address on IEEE 802 networks

Connection Identifiers

1. Connection oriented data transfer (virtual circuits).
2. Allocates a connection name during the transfer phase.

The advantages are:

1. Reduced overhead as connection identifiers are shorter than global addresses.
2. Routing may be fixed and identified by connection name.
3. Entities may want multiple connections – multiplexing.
4. State information.

Error Control

Guard against loss or damage of data and control information Error control is implemented as two separate functions:

1. Error detection
2. Sender inserts error detecting bits
3. Receiver checks these bits
4. If OK, acknowledge
5. If error, discard packet
6. Retransmission
7. If no acknowledge in given time, re-transmit
8. Performed at various layers of protocol

Flow Control

1. Done by receiving entity.
2. Function to limit amount or rate of data sent by a transmitting entity.
3. Simplest form: stop-and-wait procedure.
4. More efficient protocols: Credit systems Sliding window.
5. Needed at application as well as network layers.

Multiplexing

1. Supporting multiple connections on one machine.
2. Mapping of multiple connections at one level to a single connection at another.
3. Carrying a number of connections on one fiber optic cable
4. Aggregating or bonding ISDN lines to gain bandwidth.

Routing

Determine path or route that packets will follow.

1. Use routing protocol based on a routing algorithm.
2. "Good" path should be least cost path
3. Cost : depends on the following factors.
4. Average queuing delay
5. Propagation delay
6. Bandwidth, mean queue length, etc.
7. End systems and routers maintain routing tables.
8. Dynamic or static.

1.7 NETWORK TOPOLOGY

Q19. Define Topology and explain topologies of computer network.

Ans :

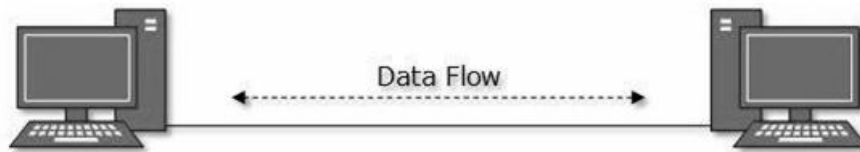
A Network Topology is the arrangement with which computer systems or network devices are connected to each other. Topologies may define both physical and logical aspect of the network. Both logical and physical topologies could be same or different in a same network.

The following factors are considered while selecting a topology:

1. Cost
2. Reliability
3. Scalability
4. Bandwidth capacity
5. Ease of installation
6. Ease of troubleshooting
7. Delay involved in routing information from one node to another.

Point-to-Point

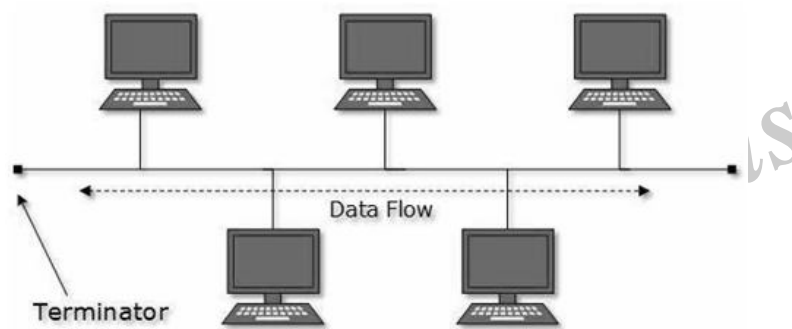
Point-to-point networks contains exactly two hosts such as computer, switches or routers, servers connected back to back using a single piece of cable. Often, the receiving end of one host is connected to sending end of the other and vice-versa.



If the hosts are connected point-to-point logically, then may have multiple intermediate devices. But the end hosts are unaware of underlying network and see each other as if they are connected directly.

1. Bus Topology

In case of Bus topology, all devices share single communication line or cable. Bus topology may have problem while multiple hosts sending data at the same time. Therefore, Bus topology either uses CSMA/CD technology or recognizes one host as Bus Master to solve the issue. It is one of the simple forms of networking where a failure of a device does not affect the other devices. But failure of the shared communication line can make all other devices stop functioning.



Both ends of the shared channel have line terminator. The data is sent in only one direction and as soon as it reaches the extreme end, the terminator removes the data from the line.

Advantages

The advantages of physical bus topology are:

1. It uses established standards and it is relatively easy to install and the use for small networks.
2. It requires fewer medium than other topologies.
3. Failure of one node does not affect the network functioning.
4. Cost is less as only one main cable is required and least amount of cable is required to connect computers.
5. Expansion is easier. New node can be easily added by using a connector.

Disadvantages

The disadvantages of bus Topology are:

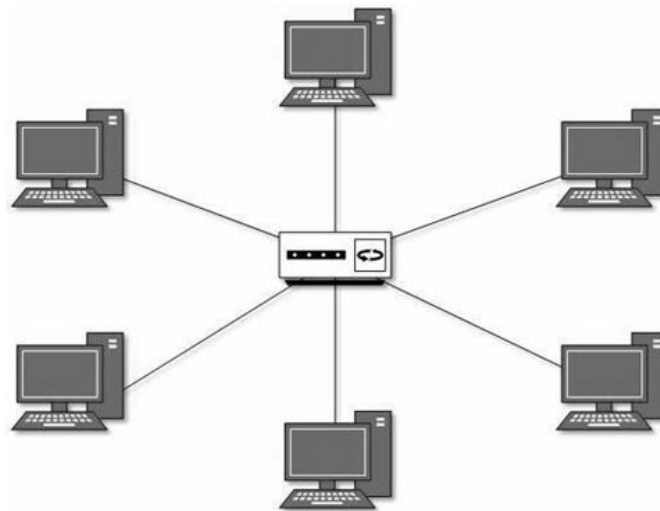
1. If the main central line fails the entire network collapses.
2. The bus networks are difficult to reconfigure, especially when the acceptable number of connections or maximum distances have been reached.
3. They are also difficult to troubleshoot because everything happens on a single media segment. This can have dangerous consequences because any break in the cabling brings the network to its knee.

4. Sharing a single communication channel results in slower access time.
5. In this topology, higher network traffic slows down the bus speed. Only one device transmits at a time, other devices wait for their turn. As a result there is no coordination between the devices for reservation of transmission time slots, so data collisions are frequent.

2. Star Topology

All hosts in Star topology are connected to a central device, known as hub device, using a point-to-point connection. That is, there exists a point to point connection between hosts and hub. The hub device can be any of the following:

- Layer-1 device such as hub or repeater
- Layer-2 device such as switch or bridge
- Layer-3 device such as router or gateway



As in Bus topology, hub acts as single point of failure. If hub fails, connectivity of all hosts to all other hosts fails. Every communication between hosts, takes place through only the hub. Star topology is not expensive as to connect one more host, only one cable is required and configuration is simple.

Advantages

The benefits of star topology are:

1. It is easier to add new node or modify any existing node without disturbing network i.e. expansion is easier.
2. Addition of new node does not increase communication delay.
3. If any local computer or link fails, the entire system does not collapse. Only that link or computer is affected.
4. It is easy to find device and cable problems i.e. fault identification and isolation is easier.
5. Media faults are automatically isolated to the failed segment.

Disadvantages

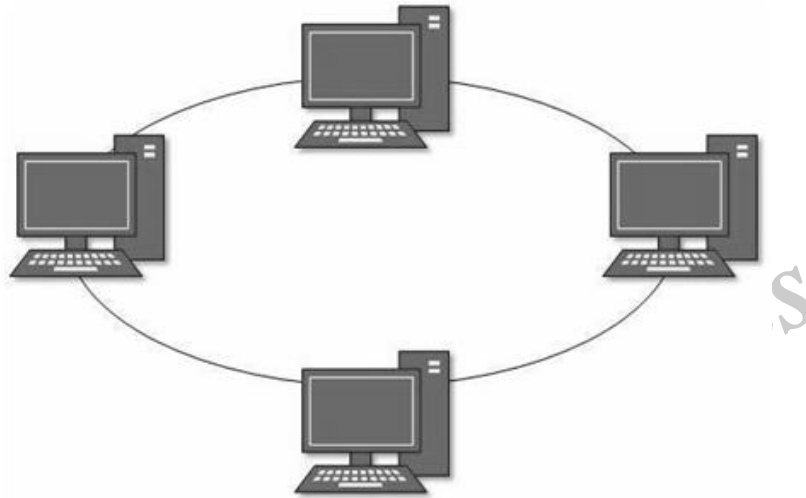
The disadvantages are considered as follows :

1. If the central controller or hub fails, entire system collapses.

2. Cabling cost is more as each node is connected individually to the hub.
3. Requires more cable than most topologies
4. Moderately difficult to install

3. Ring Topology

In ring topology, each host machine connects to exactly two other machines, creating a circular network structure. When one host tries to communicate or send message to a host which is not adjacent to it, the data travels through all intermediate hosts. To connect one more host in the existing structure, the administrator may need only one more extra cable.



Failure of any host results in failure of the whole ring. Thus, every connection in the ring is a point of failure. There are methods which employ one more backup ring.

There are two kinds of ring topologies:

1. **Single ring** : In single ring network, a single cable is shared by all the devices and data travel only in one direction. Each device waits for its turn and then transmits. When the data reaches its destination, another device can transmit.
2. **Dual ring** : This topology uses two rings to send the data, each in different direction. Thus allowing more packets to be sent over the network.

Advantages

The advantages of Ring Topology are:

1. They are very easy to troubleshoot because each device incorporates a repeater.
2. A special internal feature called becoming allows troubled workstations to identify themselves quickly.
3. There is no master computer or controller. Every computer has equal chance to place the data and access the token.
4. There are no collisions.
5. Data packets travel at greater speeds.

6. It is easier to locate the problems with device and cable i.e. fault isolation is simplified. If one device does not receive a signal within a specified time, it can issue an alarm. This alarm alerts the network operator to the problem and its location.

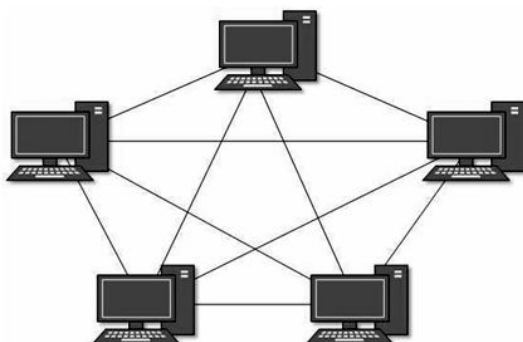
Disadvantages

The disadvantages of ring topologies are:

1. A ring network requires more cable than a bus network.
2. A break in cable ring brings down entire network (in case of single ring).
3. Adding or removing the node disturbs the network activity.
4. In ring network, communication delay is directly proportional to the number of nodes in the network. Hence addition of new nodes in the network also increases communication delay.
5. It is considerably difficult to install and reconfigure ring Topology.
6. Media failure on unidirectional or single loop causes complete network failure.

4. Mesh Topology

In this type of topology, a host is connected to one or multiple hosts. This topology has hosts in point-to-point connection with every other host or may also have hosts which are in point-to-point connection to few hosts only.



Hosts in Mesh topology also work as relay for other hosts which do not have direct point-to-point links. Mesh technology comes into two types:

Full Mesh

All hosts have a point-to-point connection to every other host in the network. Thus for every new host $n(n-1)/2$ connections are required. It provides the most reliable network structure among all network topologies.

Partially Mesh

Not all hosts have point-to-point connection to every other host. Hosts connect to each other in some arbitrarily fashion. This topology exists where we need to provide reliability to some hosts out of all.

Advantages

1. It is robust as the failure of one node does not collapse the entire system. If one link fails, the entire system continues to work.
2. There is no traffic congestion problem as dedicated links are being used.
3. Dedicated links ensure faster transmission without any delay.
4. Dedicated links also ensure data privacy and security.
5. Point to point links makes fault identification and isolation easier.

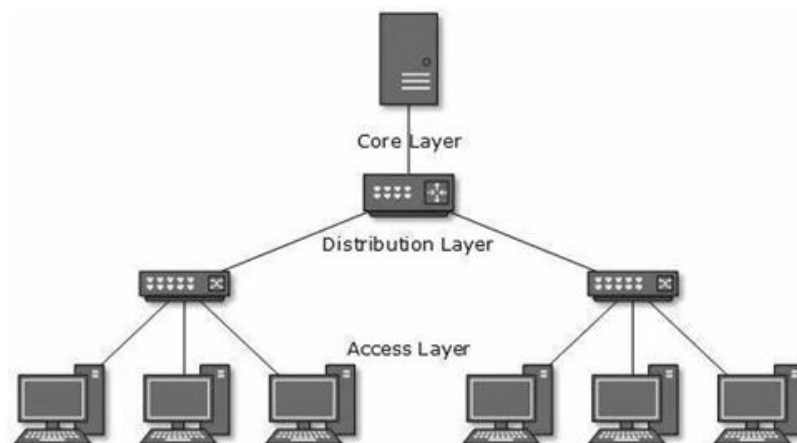
Disadvantages

1. Connecting each device to every other device in the network makes installation and reconfiguration difficult.
2. It has high cabling cost as $n(n-1)/2$ links are required to connect n nodes.

5. Tree Topology

Also known as Hierarchical Topology, this is the most common form of network topology in use presently. This topology imitates as extended Star topology and inherits properties of bus topology.

This topology divides the network in to multiple levels/layers of network. Mainly in LANs, a network is bifurcated into three types of network devices. The lowermost is access-layer where computers are attached. The middle layer is known as distribution layer, which works as mediator between upper layer and lower layer. The highest layer is known as core layer, and is central point of the network, i.e. root of the tree from which all nodes fork.



All neighboring hosts have point-to-point connection between them. Similar to the Bus topology, if the root goes down, then the entire network suffers even though it is not the single point of failure. Every connection serves as point of failure, failing of which divides the network into unreachable segment.

Advantages

1. Supported by several hardware and software vendors.
2. It allows more devices to be attached to a single central hub and can therefore increase the distance a signal can travel between devices.
3. It allows the network to isolate and prioritize communication from different computers i.e. the computers attached to one secondary hub can be given priority over the computers attached to another secondary hub.

Disadvantages

1. Overall length of each segment is limited by the type of cabling used.
2. If the backbone line breaks, the entire segment goes down.
3. More difficult to configure and wire than other topologies.
4. It has higher cabling cost in setting up a tree structure.

6. Daisy Chain

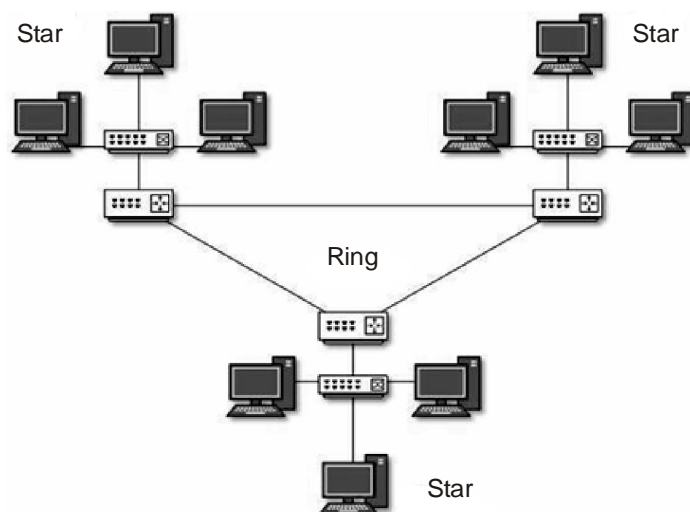
This topology connects all the hosts in a linear fashion. Similar to Ring topology, all hosts are connected to two hosts only, except the end hosts. Means, if the end hosts in daisy chain are connected then it represents Ring topology.



Each link in daisy chain topology represents single point of failure. Every link failure splits the network into two segments. Every intermediate host works as relay for its immediate hosts.

7. Hybrid Topology

A network structure whose design contains more than one topology is said to be hybrid topology. Hybrid topology inherits merits and demerits of all the incorporating topologies.



The above picture represents an arbitrarily hybrid topology. The combining topologies may contain attributes of Star, Ring, Bus, and Daisy-chain topologies. Most WANs are connected by means of Dual-Ring topology and networks connected to them are mostly Star topology networks. Internet is the best example of largest Hybrid topology.

1.8 TRANSMISSION MEDIA

Q20. What is transmission media? State different types of transmission media.

Ans :

(Imp.)

Transmission media is a pathway that carries the information from sender to receiver. We use different types of cables or waves to transmit data. Data is transmitted normally through electrical or electromagnetic signals.

An electrical signal is in the form of current. An electromagnetic signal is series of electromagnetic energy pulses at various frequencies. These signals can be transmitted through copper wires, optical fibers, atmosphere, water and vacuum. Different Media have different properties like bandwidth, delay, cost and ease of installation and maintenance. Transmission media is also called Communication channel.

Types

Transmission media is broadly classified into two groups.

1. **Wired or Guided Media or Bound Transmission Media**

Bound transmission media are the cables that are tangible or have physical existence and are limited by the physical geography. Popular bound transmission media in use are twisted pair cable, co-axial cable and fiber optical cable. Each of them has its own characteristics like transmission speed, effect of noise, physical appearance, cost etc.

2. **Wireless or Unguided Media or Unbound Transmission Media**

Unbound transmission media are the ways of transmitting data without using any cables. These media are not bounded by physical geography. This type of transmission is called Wireless communication. Nowadays wireless communication is becoming popular. Wireless LANs are being installed in office and college campuses. This transmission uses Microwave, Radio wave, Infra-red are some of popular unbound transmission media.

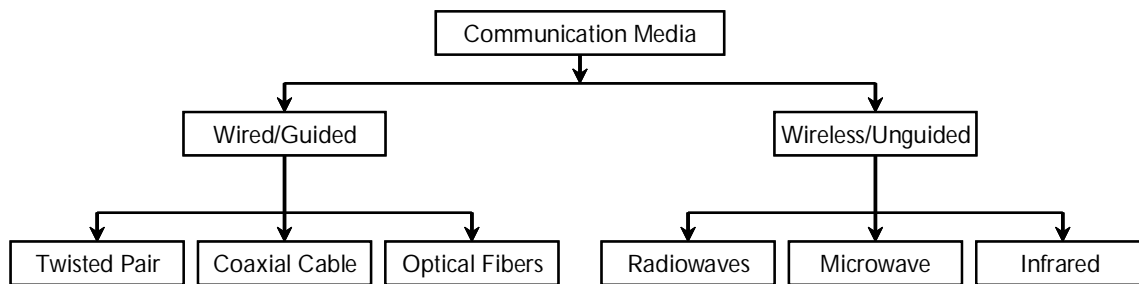


Fig. : Communication Media

The data transmission capabilities of various Medias vary differently depending upon the various factors. These factors are:

1. **Bandwidth:** It refers to the data carrying capacity of a channel or medium. Higher bandwidth communication channels support higher data rates.
2. **Radiation:** It refers to the leakage of signal from the medium due to undesirable electrical characteristics of the medium.
3. **Noise Absorption:** It refers to the susceptibility of the media to external electrical noise that can cause distortion of data signal.
4. **Attenuation:** It refers to loss of energy as signal propagates outwards. The amount of energy lost depends on frequency. Radiations and physical characteristics of media contribute to attenuation.

1.8.1 Coaxial Cable

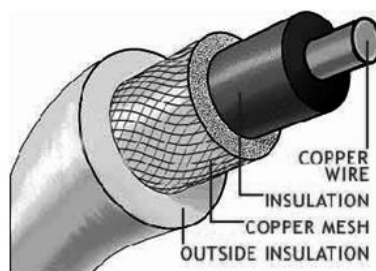
Q21. Explain the concept of Coaxial Cable.

Ans. :

(Imp.)

Coaxial cables are the guided media that carries the signal of higher frequency range compared to twisted pair cable. Coaxial cables are also called *coax*. (short form). Two types of coaxial cables are widely used: 50 ohm cable and 75 ohm cable. 50 ohm cable is used for digital transmission and 75 ohm cable is used for analog transmission. Due to the shield provided, this cable has excellent noise immunity. It has a large bandwidth and low losses. Co-axial cables are easy to install. They are often installed either in a device to device daisy chain (Ethernet) or a star (ARC net).

A coaxial cable consists of many small cables in a protective cover. The cover shields the cable from physical dangers as well as from electromagnetic interference. Within the cover, the various cables are shielded from interference with one another. Coaxial cables are used in communication networks that require many simultaneous communication links. Each coaxial cable can provide more than 5000 links. It has a data rate of 10 Mbps which can be increased with the increase in diameter of the inner conductor. The specified maximum number of nodes on a thin net segment is 30 nodes and on a thicknet it is 100 nodes.

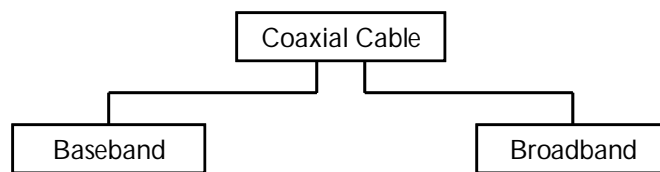


Coaxial cable is a two-conductor cable in which one conductor forms an electromagnetic shield around the other. The two conductors are separated by insulation. It is a constant impedance transmission cable. This media is used in base band and broadband transmission.

Coaxial cables do not produce external electric and magnetic fields and are not affected by them. This makes them ideally suited, although more expensive, for transmitting signals. This cable is suitable for point to point or point to multipoint applications. In fact this is the most widely used medium for local area networks. These cables are costlier than twisted pair cables but they are cheaper than the optical fiber cables.

Types of Coaxial Cables

There are two types of coaxial cables:



1. Baseband

A baseband coaxial cable transmits a single signal at a time at very high speed. A broadband coaxial cable can transmit many simultaneous signals using different frequencies. A baseband cable is mainly used for LANs.



Baseband coaxial cables support frequency range of a-4kHz and are used for digital signaling. Broadband coaxial cable supports the frequency range above 4kHz and are used for analog signals. So it must be used with a modem. The digital signal inserted on these cables is encoded using Manchester or Differential Manchester coding. The digital signal consumes the entire frequency spectrum of the cable. So it is not possible to transmit multiple channels using FDM. The transmission of digital signal on the cable is bi-directional.

Baseband coaxial cables are 50 ohm cables used for 'digital transmission'. For 1Km cables the bandwidth is 1-2 Gbps. Longer cables can be used with low data rates or periodic amplifiers. Broadband coaxial cables are 75 ohm cables used for analog transmission. The baseband co-axial cable was originally used for the Ethernet system that operates at 10 Mbps. They use standard cable television technology. To transmit digital signals on an analog network, each interface must have converters i.e. analog to digital for outgoing bit stream n vice versa another difference between baseband and broadband is that broadband systems have developed dual cables. The maximum length of baseband co-axial cable between two repeaters is dependent on the data rates.

2. Broadband

Since broadband is used for large area, it requires amplifiers which are unidirectional. In dual band systems two identical cables run together, one used for outgoing data, one for incoming data. Different bandwidths are given for inbound and outbound cables. Eg: for 300MHz, 5-30MHz for inbound and 40-300MHz for outbound. There are two systems that use the bus LANs namely 10 BASE 5 and 10 BASE 2 which are compared based on various factors in Table.

IEEE-802.3-Specification-for-10Mbps-baseband-co-axial cable Bus Lan

Sr. No.	Parameter	10 BASE 5	10 BASE 2
1	DataRate	10 Mbps	10 Mbps
2	Maximum Segment	500 m	185 m
3	Length	2500 m	1000 m
4	Network span	100 m	30 m
5	Nodes per	2.5 m	0.5 m
6	Segment Node spacing Cable Diameter	1 cm	0.5 cm

- **Features:** It provides better immunity than twisted pair. This cable is able to transmit data at higher rates.
- **Limitations:** High installation cost. High maintenance cost.

Advantages of Coaxial Cables

1. It can be used for both analog and digital transmission.
2. It offers higher bandwidth as compared to twisted pair cable and can span longer distances.
3. Because of better shielding in coaxial cable, loss of signal or attenuation is less.
4. Better shielding also offers good noise immunity.
5. It is relatively inexpensive as compared to optical fibers.
6. It has lower error rates as compared to twisted pair.
7. It is not as easy to tap as twisted pair because copper wire is contained in plastic jacket.

Disadvantages of Coaxial Cables

It is usually more expensive than twisted pair.

Applications of Co-axial Cables

1. Analog telephone networks.
2. Digital telephone network.
3. Cable TV
4. Traditional Ethernet LANs
5. Digital transmission
6. Thick Ethernet

1.8.2 Fiber Optics

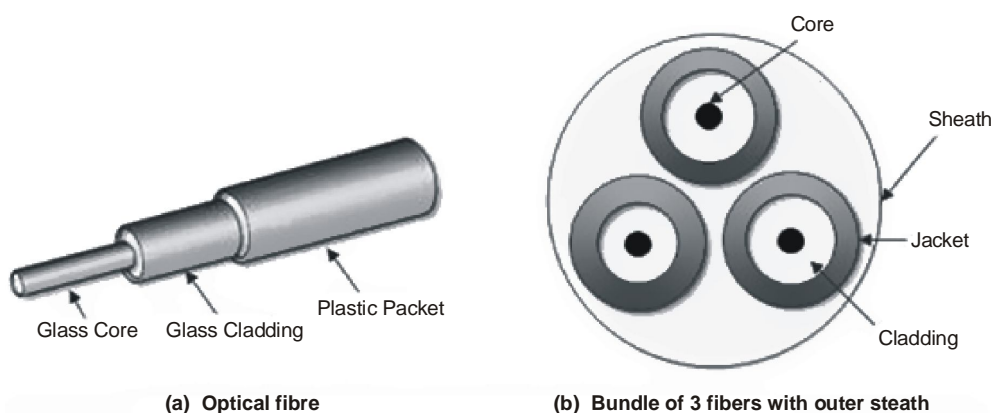
Q22. What are Optical Fibers?

Ans :

Optical fiber consists of thin glass fibers or plastic that can carry information at frequencies in the visible light spectrum and beyond. The typical optical fiber consists of a very narrow strand of glass called the core. Around the core is a concentric layer of glass called the cladding?

An optical transmission system has three basic components

- **Light source:** In such a system a pulse of light indicates bit 1 and the absence of light indicates bit 0. Light source can be an LED or a laser beam.
- **Transmission medium:** Transmission medium is the ultra-thin fiber of glass.
- **Detector:** A detector generates an electrical pulse when the light falls on it,

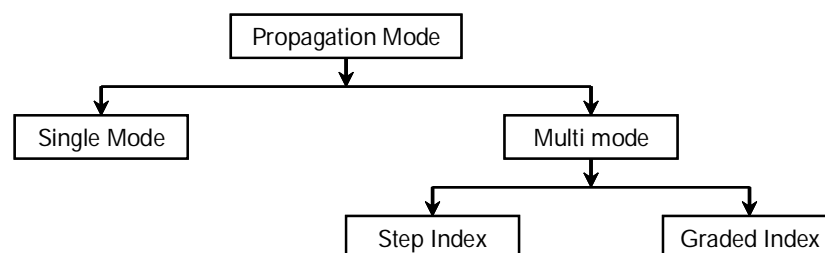


A typical core diameter is 62.5 microns. Typically cladding has a diameter of 125 microns. 100 microwatts power roughly) a light emitting diode can couple into an optical fiber. Coating the cladding is a protective coating consisting of plastic, it is called the Jacket.

The loss in signal power as light travels down the fiber is called attenuation. An important characteristic of fiber optics is refraction. Refraction is the characteristic of a material to either pass or reflect light. When light passes through a medium, it "bends" as it passes from one medium to the other. An example of this is when we look into a pond of water. If the angle of incidence is small, the light rays are reflected and do not pass into the water.

If the angle of incident is great, light passes through the media but is bent or refracted. Optical fibers work on the principle that the core refracts the light and the cladding reflects the light. The core refracts the light and guides the light along its path. The cladding reflects any light back into the core and stops light from escaping through it - it bounds the medium! Fast data transmission rate is an advantage to using fiber optics data transmission.

Types of Optical Fibers



1. Single Mode Fiber

The various characteristics of Single mode fiber are:-

1. The diameter of glass core in single mode fiber is very small ranging from 8 to 10 microns.
2. In this mode, light can propagate only in a straight line, without bouncing.
3. Fiber glass has lower density (index of refraction) that creates a critical angle close enough to 90° such that the beam propagates in a straight line.
4. In this case, propagation of different beams is almost identical and delays are negligible. The beams arrive at destination together and can be recombined with little distortion to the signal.
5. Single mode fibers are more expensive and are widely used for long distance communication.
6. These types of fibers can transmit data at 50 Gbps for 100 kilometers without amplification.

2. Multimode Fiber

1. In multimode fiber, multiple beams travel in the core in different paths.
2. In multimode fiber, the diameter of core is about 50 microns.
3. Multimode fibers are further categorized into Step index fibers and Graded index fibers.

(a) Step Index Fiber

1. Density of core remains constant from the centre to the edges.
2. A beam of light moves in a straight line in this medium until it reaches the interface of core and the cladding.
3. At this interface, the angle of ray is changed due to the change in density.
4. In this mode, some beams travel in a straight line through the core and reaches destination without reflection or refraction.

5. The beams that strike the interface of core and cladding at an angle smaller than critical angle penetrate the cladding and are lost.
6. The beams striking at an angle greater than critical angle are reflected back in core and form total internal reflection.
7. In this fiber, a ray with smaller angle of incidence requires more bounces thus will take more time to reach the destination whereas the ray with high angle of incidence will require less number of bounces and will reach the destination in lesser time.

(b) Graded Index Fiber

1. A graded index fiber has different densities at the core and at the edges. Density is highest at the centre of the core and decreases gradually to its lowest at the edge.
2. Because of this difference in densities, different beams refract at different angles into a curve.
3. Only the horizontal beams move in a straight line due to constant density at the centre.

Q23. State the advantages and disadvantages of Optical Fiber.

Ans :

Advantages

1. They are not affected by electrical and magnetic interference as the data travel in form of light.
2. Optical fiber offers higher bandwidth than twisted pair or coaxial cable.
3. Optical fibers are thin, lighter in weight and small in size as compared to other wired Medias. It is easier to group several optical fibers in one bundle.
4. Glass is more resistant to corrosive materials as compared to copper. Hence can be laid in different environments.

5. In optical fibers, attenuation (loss of signal) is very low. Therefore these fibers can run several kilometers without amplification.
6. Fibers do not leak light and are quite difficult to tap. So they provide security against potential wiretappers.
7. There is no cross-talk problem in optical fibers.
8. They are highly suitable for environments where speed is needed with full accuracy.
9. Photons in fiber do not affect one another (as they have no charge) and are not affected by stray photons outside the fiber. But when electrons move in a wire they affect each other and are themselves affected by electrons outside the wire.
10. The size (diameter) of the optical fibers is very small (it is comparable to the diameter of human hair). Therefore a large number of optical fibers can fit into a cable of small diameter.
11. The material used for the manufacturing of optical fibers is "silica glass". This material is easily available. So the optical fibers cost lower than the cables with metallic conductors.
12. As the light rays have a very high frequency in the GHz range, the bandwidth of the optical fiber is extremely large. This allows transmission of more number of channels. Therefore the information carrying capacity of an optical fiber is much higher than that of a co-axial cable.

Disadvantages

1. Fiber optics cables are fragile i.e. more easily broken than wires.
2. Being fragile, optical fibers need to be put deep into the land. This causes a lot of installation cost. Also the interface used for these fibers are expensive.
3. Optical fibers are unidirectional for two-way communication, two fibers are required.
4. It is a newer technology and requires skilled people to administer and maintain them.

Q24. State the characteristics and applications of optical fibers.

Ans :

Characteristics

Fiber optic cables have the following characteristics:

1. Fiber optic cabling can provide extremely high bandwidths in the range from 100 mbps to 2 gigabits because light has a much higher frequency than electricity.
2. The number of nodes which a fiber optic can support does not depend on its length but on the hub or hubs that connect cables together.
3. Fiber optic cable has much lower attenuation and can carry signal to longer distances without using amplifiers and repeaters in between.
4. Fiber optic cable is not affected by EMI effects and can be used in areas where high voltages are passing by.
5. The cost of fiber optic cable is more compared to twisted pair and co-axial.
6. The installation of fiber optic cables is difficult and tedious.

Applications

1. Optical fiber transmission systems are widely used in the backbone of networks. Current optical fiber systems provide transmission rates from 45 Mb/s to 9.6 Gb/s using the single wavelength transmission.
2. The installation cost of optical fibers is higher than that for the co-axial or twisted wire cables.
3. Optical fibers are now used in the telephone systems.
4. In the Local Area Networks (LANs).

1.8.3 Line Coding

Q25. Describe the characteristics of line coding.

Ans :

1. Line coding is the process of converting digital data to digital signals.
2. The data may be in the form of text, numbers, graphical images, audio, or video, are stored in computer memory as sequences of bits.
3. Line coding converts a sequence of bits to a digital signal. At the sender, digital data are encoded into a digital signal; at the receiver, the digital data are recreated by decoding the digital signal.

Characteristics

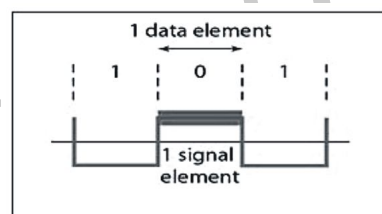
The different characteristics of Line Coding Technique are as follows:

1. Signal Element versus Data Element

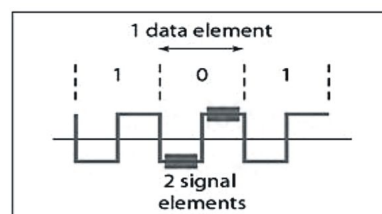
A data element is the smallest entity that can represent a piece of information. This is the bit. In digital data communications, a signal element carries data elements. A signal element is the shortest unit (time wise) of a digital signal. In other words, data elements are what we need to send; signal elements are what we can send. Data elements are being carried; signal elements are the carriers.

We define a ratio r which is the number of data elements carried by each signal element. The shows several situations with different values of r .

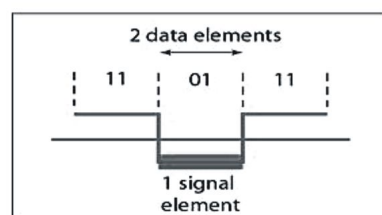
In part of the figure, one data element is carried by one signal element ($r = 1$). In part b of the figure, we need two signal elements (two transitions) to carry each data element ($r = 1/2$). In part c of the figure, a signal element carries two data elements ($r = 2$). In part d, a group of 4 bits is being carried by a group of three signal elements ($r = 4/3$). For every line coding scheme r value should be defined.



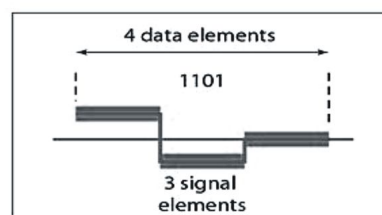
a. One data element per one signal element ($r = 1$)



b. One data element per two signal elements ($r = \frac{1}{2}$)



c. Two data elements per one signal element ($r = 2$)



d. Four data elements per three signal elements ($r = \frac{4}{3}$)

2. Data Rate Versus Signal Rate

The data rate defines the number of data elements (bits) sent in 1s. The unit is bits per second (bps). The signal rate is the number of signal elements sent in 1s. The unit is the baud. The data rate is sometimes called the bit rate; the signal rate is sometimes called the pulse rate, the modulation rate, or the baud rate.

One goal in data communications is to increase the data rate while decreasing the signal rate. Increasing the data rate increases the speed of transmission; decreasing the signal rate decreases the bandwidth requirement.

3. Bandwidth

Digital signal that carries information is non-periodic. The bandwidth of a non-periodic signal is continuous with an infinite range. However, most digital signals we encounter in real life have a bandwidth with finite values. In other words, the bandwidth is theoretically infinite, but many of the components have such small amplitude that they can be ignored. The effective bandwidth is finite.

4. Baseline Wandering

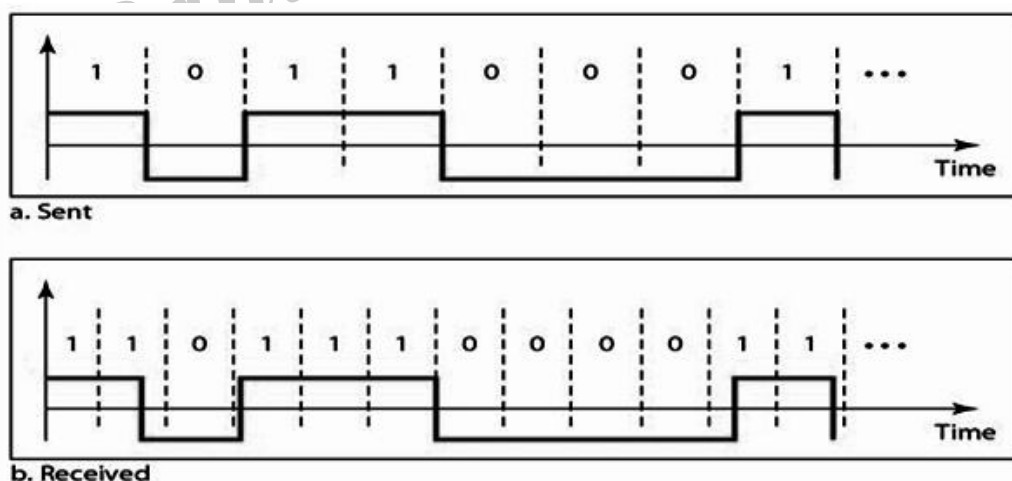
In decoding a digital signal, the receiver calculates a running average of the received signal power. This average is called the baseline. The incoming signal power is evaluated against this baseline to determine the value of the data element. A long string of 0s or 1s can cause a drift in the baseline (baseline wandering) and make it difficult for the receiver to decode correctly. A good line coding scheme needs to prevent baseline wandering.

5. DC Components

When the voltage level in a digital signal is constant for a while, the spectrum creates very low frequencies (results of Fourier analysis). These frequencies around zero, called DC (direct-current) components, present problems for a system that cannot pass low frequencies or a system that uses electrical coupling (via a transformer).

6. Self-synchronization

To correctly interpret the signals received from the sender, the receiver's bit intervals must correspond exactly to the sender's bit intervals. If the receiver clock is faster or slower, the bit intervals are not matched and the receiver might misinterpret the signals. The following figure represents the synchronization problem.



7. Built-in Error Detection

It is desirable to have a built-in error-detecting capability in the generated code to detect some of or all the errors that occurred during transmission. Some encoding schemes that we will discuss have this capability to some extent.

8. Immunity to Noise and Interference

Another desirable code characteristic is a code that is immune to noise and other interferences.

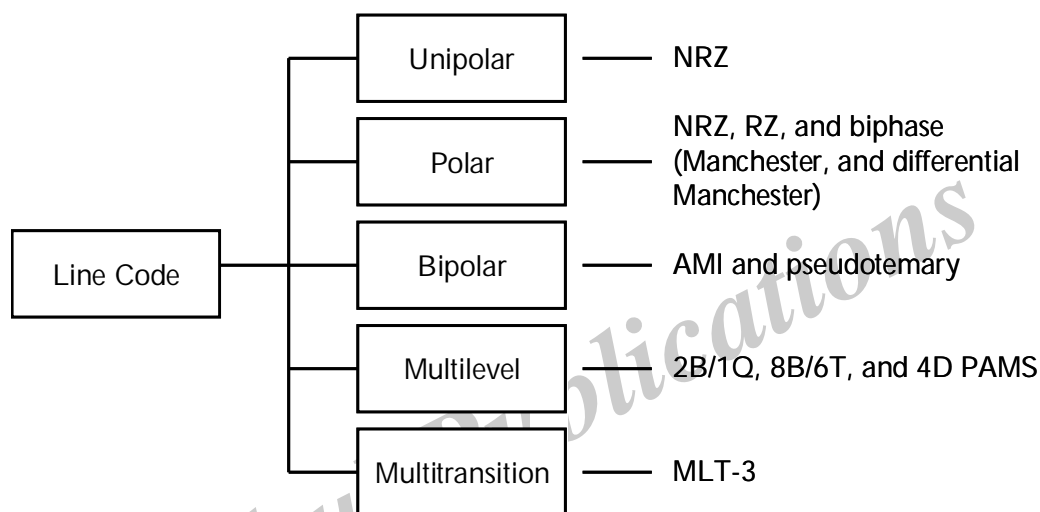
9. Complexity

A complex scheme is more costly to implement than a simple one. For example, a scheme that uses four signal levels is more difficult to interpret than one that uses only two levels.

Q26. Explain different line coding techniques.

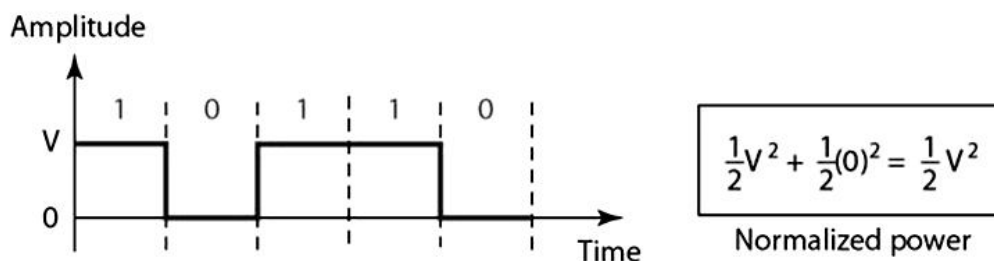
Ans :

The Line Coding schemes are categorized as shown in the following figure:



Unipolar Scheme: In a unipolar scheme, all the signal levels are on one side of the time axis, either above or below.

NRZ (Non-Return-to-Zero): Traditionally, a unipolar scheme was designed as a non-return-to-zero (NRZ) scheme in which the positive voltage defines bit 1 and the zero voltage defines bit 0. It is called NRZ because the signal does not return to zero at the middle of the bit. The following figure shows a unipolar NRZ scheme.



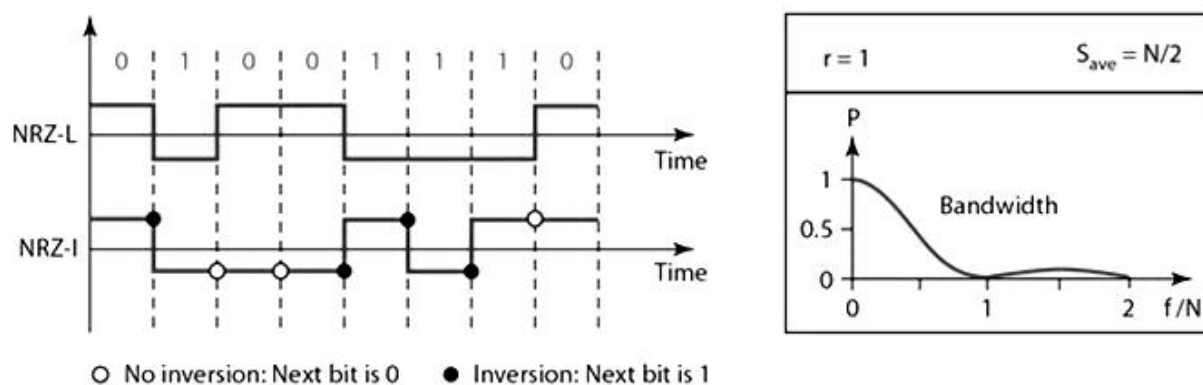
Compared with its polar counterpart, the normalized power (power needed to send 1 bit per unit line resistance) is double that for polar NRZ. For this reason, this scheme is normally not used in data communications today.

Polar Schemes

In polar schemes, the voltages are on the both sides of the time axis. For example, the voltage level for 0 can be positive and the voltage level for 1 can be negative.

1. Non-Return-to-Zero (NRZ)

In polar NRZ encoding, we use two levels of voltage amplitude. We can have two versions of polar NRZ: NRZ-L and NRZ-I, as shown in the following Figure. The figure also shows the value of r , the average baud rate, and the bandwidth.



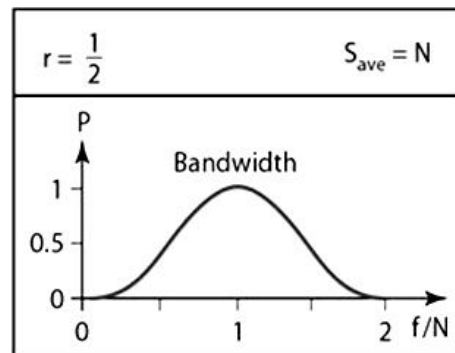
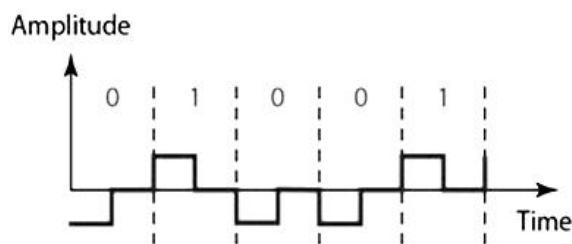
In the first variation, NRZ-L (NRZ-Level), the level of the voltage determines the value of the bit. In the second variation, NRZ-I (NRZ-Invert), the change or lack of change in the level of the voltage determines the value of the bit. If there is no change, the bit is 0; if there is a change, the bit is 1.

Drawbacks

1. Baseline wandering is a problem for both variations; it is twice as severe in NRZ-L. If there is a long sequence of 0s or 1s in NRZ-L, the average signal power becomes skewed. The receiver might have difficulty discerning the bit value.

In NRZ-I this problem occurs only for a long sequence of 0s. If we eliminate the long sequence of 0s, we can avoid baseline wandering.

2. The synchronization problem (sender and receiver clocks are not synchronized) also exists in both schemes. Again, this problem is more serious in NRZ-L than in NRZ-I. While a long sequence of 0s can cause a problem in both schemes, a long sequence of 1s affects only NRZ-L.
3. Another problem with NRZ-L occurs when there is a sudden change of polarity in the system. For example, if twisted-pair cable is the medium, a change in the polarity of the wire results in all 0s interpreted as 1s and all 1s interpreted as 0s. NRZ-I does not have this problem. Both schemes have an average signal rate of $N/2$.
4. Return to Zero (RZ): The main problem with NRZ encoding occurs when the sender and receiver clocks are not synchronized. The receiver does not know when one bit has ended and the next bit is starting. One solution is the return-to-zero (RZ) scheme, which uses three values: positive, negative, and zero. In RZ, the signal changes not between bits but during the bit. In the following figure, we see that the signal goes to 0 in the middle of each bit. It remains there until the beginning of the next bit.



Drawbacks

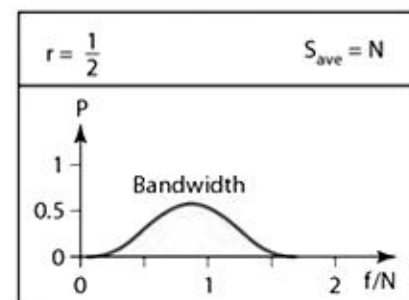
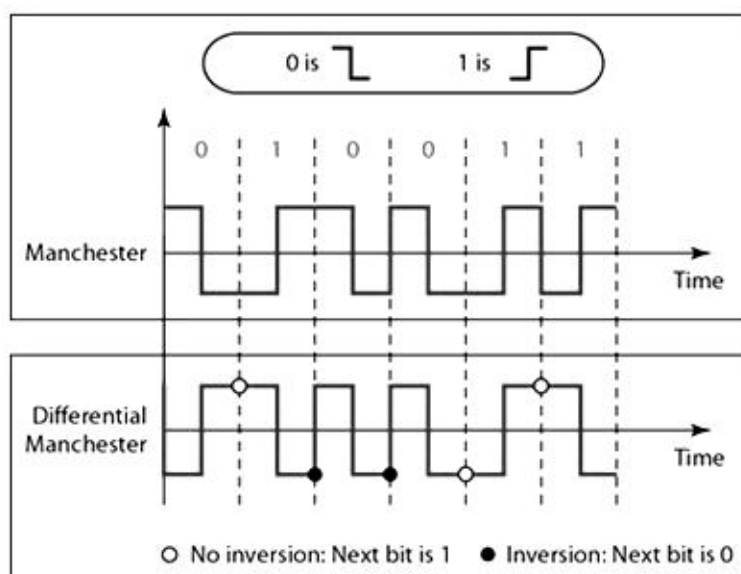
- (i) The main disadvantage of RZ encoding is that it requires two signal changes to encode a bit and therefore occupies greater bandwidth.
- (ii) Sudden change of polarity resulting in all 0s interpreted as 1s and all 1s interpreted as 0s, still exist here, but there is no DC component problem.
- (iii) Another problem is the complexity: RZ uses three levels of voltage, which is more complex to create and discern. As a result of all these deficiencies, the scheme is not used today.

Bi-phase Manchester and Differential Manchester

The idea of RZ (transition at the middle of the bit) and the idea of NRZ-L are combined into the Manchester scheme.

In Manchester encoding, the duration of the bit is divided into two halves. The voltage remains at one level during the first half and moves to the other level in the second half. The transition at the middle of the bit provides synchronization.

Differential Manchester, on the other hand, combines the ideas of RZ and NRZ-I. There is always a transition at the middle of the bit, but the bit values are determined at the beginning of the bit. If the next bit is 0, there is a transition; if the next bit is 1, there is none. The following figure shows both Manchester and differential Manchester encoding.

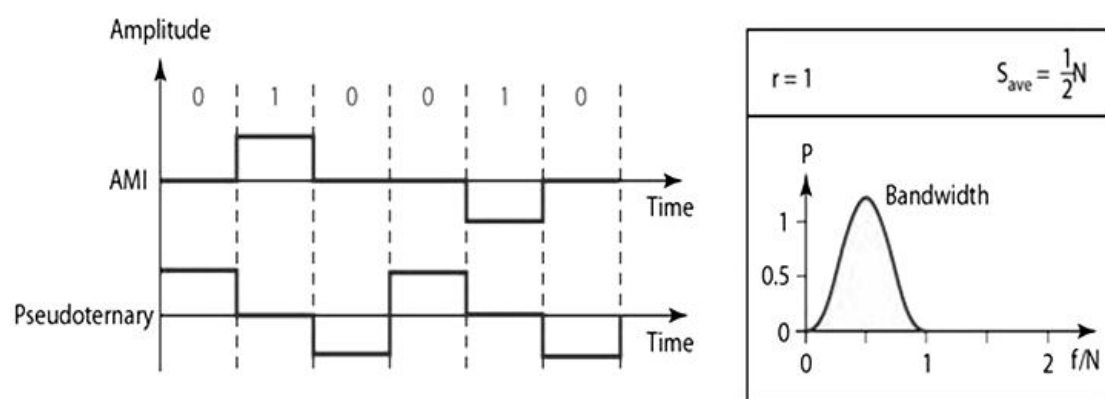


The Manchester scheme overcomes several problems associated with NRZ-L, and differential Manchester overcomes several problems associated with NRZ-I. First, there is no baseline wandering. There is no DC component because each bit has a positive and negative voltage contribution. The only drawback is the signal rate. The signal rate for Manchester and differential Manchester is double that for NRZ. The reason is that there is always one transition at the middle of the bit and maybe one transition at the end of each bit.

Bipolar Schemes

In bipolar encoding (sometimes called multilevel binary), there are three voltage levels, positive, negative, and zero. The voltage level for one data element is at zero, while the voltage level for the other element alternates between positive and negative.

AMI and Pseudoternary The following figure shows two variations of bipolar encoding: AMI and pseudoternary.



A common bipolar encoding scheme is called bipolar alternate mark inversion (AMI). In alternate mark inversion, a neutral zero voltage represents binary 0. Binary 1s are represented by alternating positive and negative voltages.

A variation of AMI encoding is called Pseudoternary in which the 1 bit is encoded as a zero voltage and the 0 bit is encoded as alternating positive and negative voltages.

The bipolar scheme was developed as an alternative to NRZ. The bipolar scheme has the same signal rate as NRZ, but there is no DC component. The NRZ scheme has most of its energy concentrated near zero frequency, which makes it unsuitable for transmission over channels with poor performance around this frequency. The concentration of the energy in bipolar encoding is around frequency $N/2$.

Multilevel Schemes

The desire to increase the data speed or decrease the required bandwidth has resulted in the creation of many schemes. The goal is to increase the number of bits per baud by encoding a pattern of m data elements into a pattern of n signal elements. We only have two types of data elements (0's and 1's), which means that a group of m data elements can produce a combination of 2^m data patterns.

We can have different types of signal elements by allowing different signal levels. If we have L different levels, then we can produce L^n combinations of signal patterns.

If $2^m = L^n$, then each data pattern is encoded into one signal pattern. If $2^m < L^n$, data patterns occupy only a subset of signal patterns. The subset can be carefully designed to prevent baseline wandering, to provide synchronization, and to detect errors that occurred during data transmission. Data encoding is not possible if $2^m > L^n$ because some of the data patterns cannot be encoded.

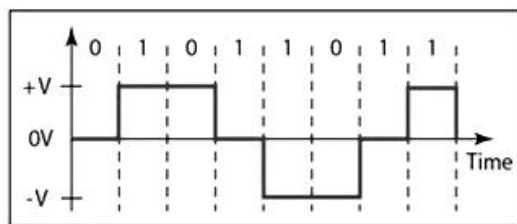
The code designers have classified these types of coding as mBnL, where m is the length of the binary pattern, B means binary data, n is the length of the signal pattern, and L is the number of levels in the signaling. A letter is often used in place of L: B(binary) for L=2, T (ternary) for L = 3, and Q (quaternary) for L = 4. Note that the first two letters define the data pattern, and the second two define the signal pattern.

Multiline Transmission: MLT-3:

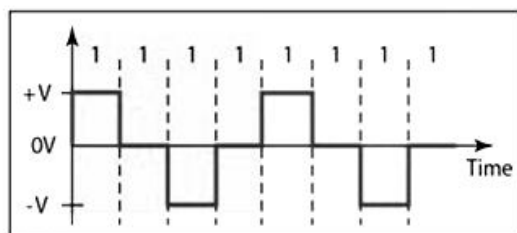
NRZ-I and differential Manchester are classified as differential encoding but use two transition rules to encode binary data (no inversion, inversion). If we have a signal with more than two levels, we can design a differential encoding scheme with more than two transition rules. MLT-3 is one of them. The multiline transmission, three level (MLT-3) scheme uses three levels (+V, 0 and -V) and three transition rules to move between the levels.

1. If the next bit is 0, there is no transition.
2. If the next bit is 1 and the current level is not 0, the next level is 0.
3. If the next bit is 1 and the current level is 0, the next level is the opposite of the last nonzero level.

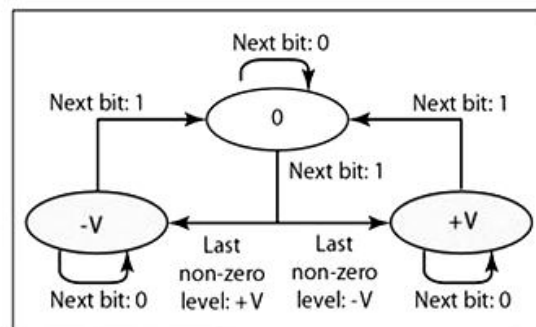
The behavior of MLT-3 can best be described by the state diagram shown in the following figure. The three voltage levels (-V, 0, and +V) are shown by three states (ovals). The transition from one state (level) to another is shown by the connecting lines. The following figure also shows two examples of an MLT-3 signal.



a. Typical case



b. Worse case



c. Transition states

The signal rate is the same as that for NRZ-I, but with greater complexity (three levels and complex transition rules). It turns out that the shape of the signal in this scheme helps to reduce the required bandwidth. Let us look at the worst-case scenario, a sequence of 1s. In this case, the signal element pattern +V 0 -V 0 is repeated every 4 bits.

A non-periodic signal has changed to a periodic signal with the period equal to 4 times the bit duration. This worst-case situation can be simulated as an analog signal with a frequency one-fourth of the bit rate. In other words, the signal rate for MLT-3 is one-fourth the bit rate. This makes MLT-3 a suitable choice when we need to send 100 Mbps on a copper wire that cannot support more than 32 MHz (frequencies above this level create electromagnetic emissions).

1.8.4 Modem

Q27. What is a Modem? Explain different types of Modems.

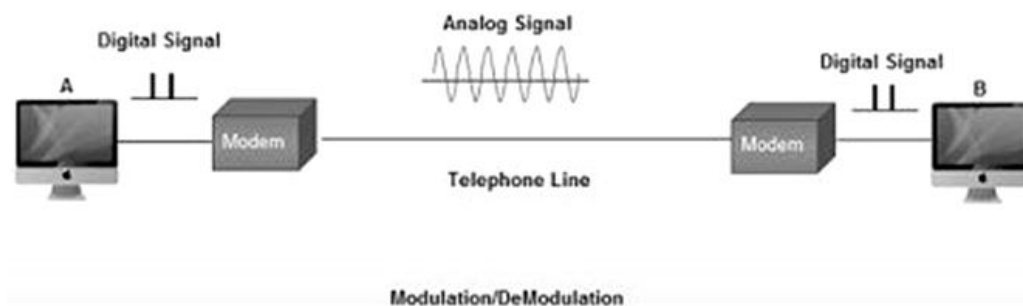
Ans :

(Imp.)

Modems are used for data transfer from one computer network to another computer network through telephone lines. The computer network works in digital mode, while analog technology is used for carrying messages across phone lines.

Modulator converts information from digital mode to analog mode at the transmitting end and demodulator converts the same from analog to digital at receiving end. The process of converting analog signals of one computer network into digital signals of another computer network so they can be processed by a receiving computer is referred to as digitizing.

When an analog facility is used for data communication between two digital devices called Data Terminal Equipment (DTE), modems are used at each end. DTE can be a terminal or a computer.



The modem at the transmitting end converts the digital signal generated by DTE into an analog signal by modulating a carrier. This modem at the receiving end demodulates the carrier and hand over the demodulated digital signal to the DTE.

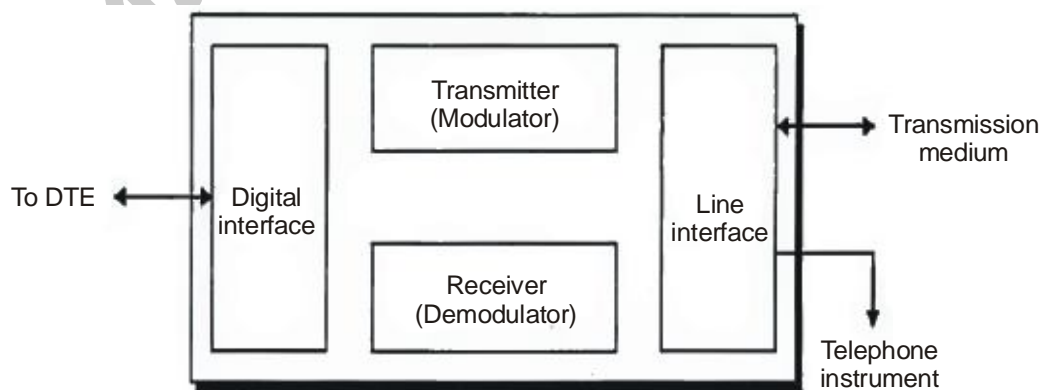


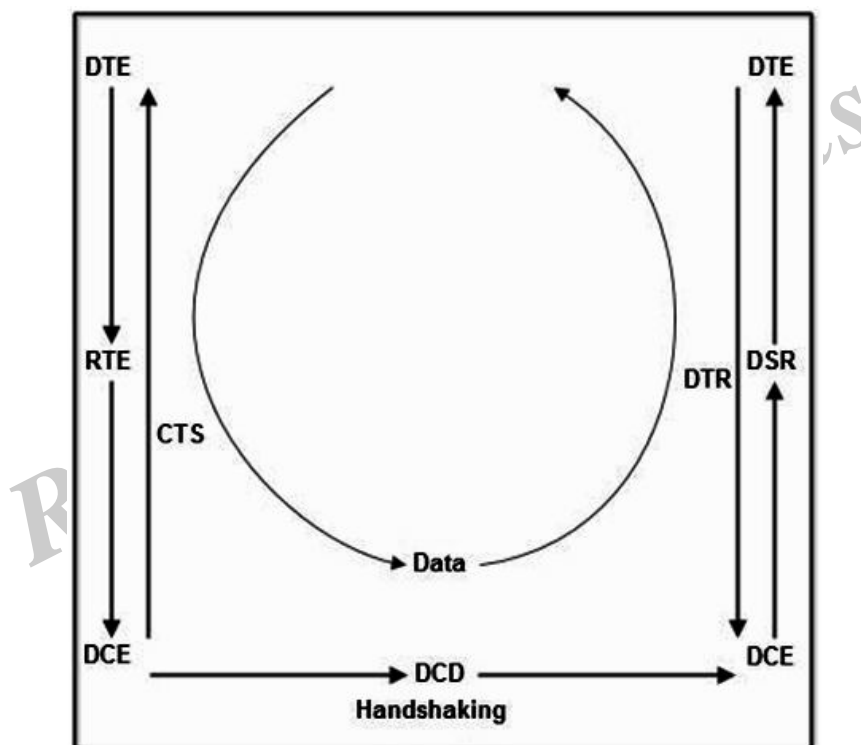
Fig. : Building blocks of a modem

The transmission medium between the two modems can be dedicated circuit or a switched telephone circuit. If a switched telephone circuit is used, then the modems are connected to the local telephone exchanges. Whenever data transmission is required connection between the modems is established through telephone exchanges.

Ready to Send

To begin with the Data Terminal Equipment or DTE (better known as a computer) sends a Ready To Send or RTS signal to the Data Communication Equipment or DCE (better known as a modem). This is sometimes known as a wakeup call and results in the modem sending a Data Carrier Detect or DCD signal to the receiving modem. There then follows a series of signals passed between the two until the communication channel has been established. This process is known as handshaking and helps to explain why, even now, some companies like Composure use the symbol of two hands grasping each other to mean being on-line. Of course, after that all it takes is for the second modem to send a Data Set Ready or DSR signal to its computer and wait for the Data Terminal Ready or DTR reply. When that happens the first modem sends a Clear To Send or CTS signal to the computer that started the whole process off and data can then be transmitted. It is as simple as that.

Alternatively, for anyone confused by what the entire Internet industry dubs TLA's which means Three Letter Acronyms, the following diagram should help.



It only looks confusing. Take a second look and everything will soon become obvious.

By way of completeness, these signals are all sent through different pins in the plug which is why the handbooks for all modems and printers carry a pin diagram somewhere in the section on troubleshooting. They are also standardized after the industry leaders met to agree standards for a whole range of peripheral equipment. The Recommended Standard for cable was number 232 which explains that one technical term probably everybody has heard of: RS 232.

Of course, that still leaves the question of exactly how data is transferred from one computer to another; something that is more of a problem than might first appear mainly because the phone lines are analogue while computers are digital. In simple terms this means a telephone signal is constantly changing.

To understand that just think of a sine wave as produced on an oscilloscope. The signal might be constant, but it is constantly changing from positive to negative and back again in a series of smooth curves. Computers, on the other hand, can only understand information when it is presented as a string of binary digits so the idea is to map digital output onto an analogue signal.

Without going into technical details this is done by superimposing different frequencies onto the analogue signal (which then becomes known as the carrier wave). Different frequencies can then represent different groups of binary digits in a process which is known as modulation when it is being transmitted and demodulation when it is decoded at the receiving end. Naturally two way communication is achieved by having a single device being capable of both modulation and demodulation, from which the unit takes its name: the modem.

From this it becomes obvious that the more frequencies that can be superimposed on the carrier wave the faster data can be transmitted. Alternatively, to take a different point of view, the more data there is to be transmitted so the more frequencies are needed.

Unfortunately it is only possible to send a limited number of frequencies at the same time, known as the bandwidth, which means communication takes that much longer as the size of the signals steadily increases. Now that pictures, sound and even video sequences are transmitted over the Internet on a regular basis, and as these all call for massive data files, the amount of available bandwidth is likely to be a problem for some time.

Finally, as the whole process comes down to sending binary digits or bits over a phone line the speed of the system is expressed as Bits Per Second or BPS which is a figure quoted by all the modem manufacturers.

Unfortunately when it comes to data communications there is a lot more involved than just how fast bits can be sent down a phone line. There is also the problem of what those bits mean and how they can be assembled into something intelligible at the far end. Here a whole range of issues need to be addressed and so it might be a good idea to briefly look at the first of these which are the transmission protocols.

Types of Modems

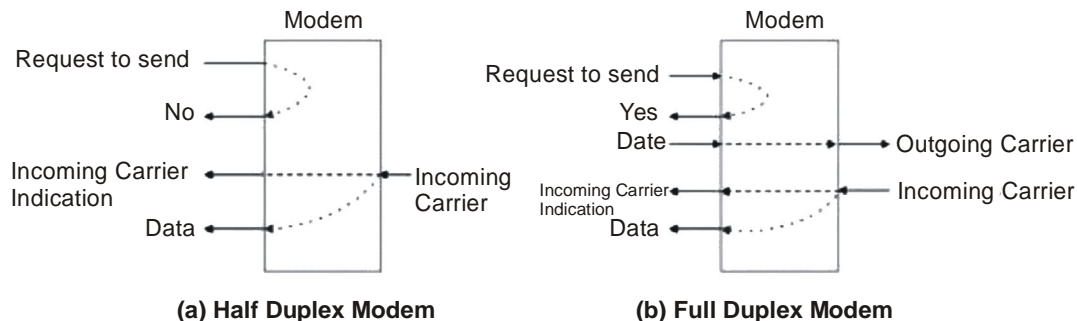
1. Modems can be of several types and they can be categorized in a number of ways.
2. Categorization is usually based on the following basic modem features:
 - (a) Directional capacity: half duplex modem and full duplex modem.
 - (b) Connection to the line: 2-wire modem and 4-wire modem.
 - (c) Transmission mode: asynchronous modem and synchronous modem.

Half Duplex and Full Duplex Modems

Half Duplex

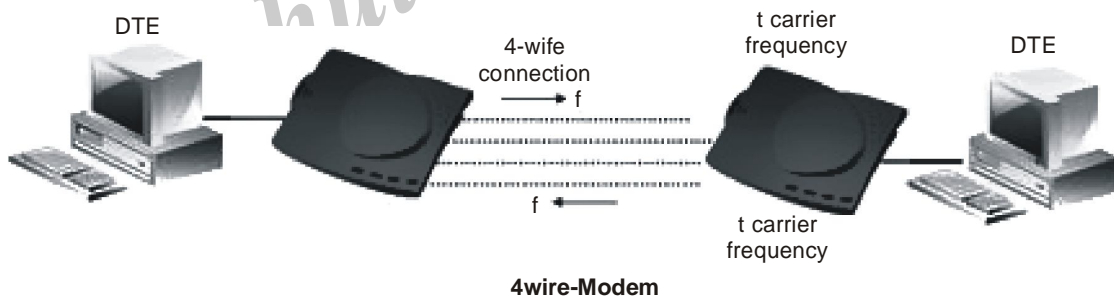
1. A half duplex modem permits transmission in one direction at a time.
2. If a carrier is detected on the line by the modem, it gives an indication of the incoming carrier to the DTE through a control signal of its digital interface.

- As long as they are being received; the modem does not give permission to the DTE to transmit data.



Full Duplex

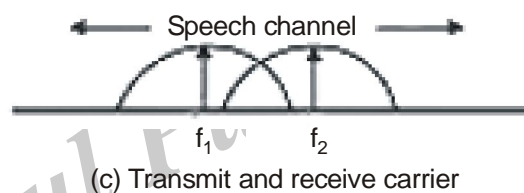
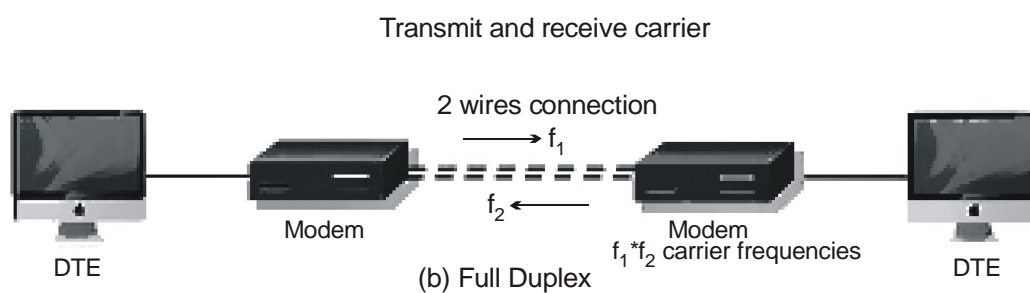
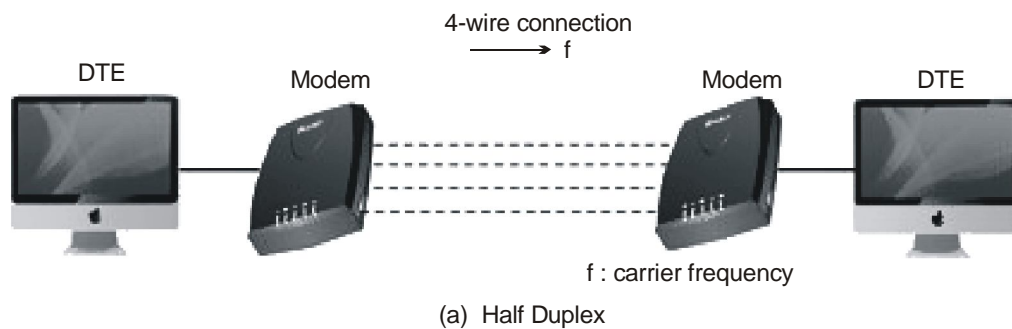
- A full duplex modem allows simultaneous transmission in both directions.
- Therefore, there are two carriers on the line, one outgoing and the other incoming. Wire and 4-wire Modems
- The line interface of the modem can have a 2-wire or a 4-wire connection to transmission medium. 4-wire Modem
- In a 4-wire connection, one pair of wires is used for the outgoing carrier and the other pair is used for incoming carrier.
- Full duplex and half duplex modes of data transmission are possible on a 4-wire connection.
- As the physical transmission path for each direction is separate, the same carrier frequency can be used for both the directions.



2-wire Modem

- 2-wire modems use the same pair of wires for outgoing and incoming carriers.
- A leased 2-wire connection is usually cheaper than a 4-wire connection as only one pair of wires is extended to the subscriber's premises.
- The data connection established through telephone exchange is also a 2-wire connection.
- In 2-wire modems, half duplex mode of transmission that uses the same frequency for the incoming and outgoing carriers can be easily implemented.
- For full duplex mode of operation, it is necessary to have two transmission channels, one for transmit direction and the other for receive direction.

This is achieved by frequency division multiplexing of two different carrier frequencies. These carriers are placed within the bandwidth of the speech channel.



1.8.4.1 Asynchronous & Synchronous Modems

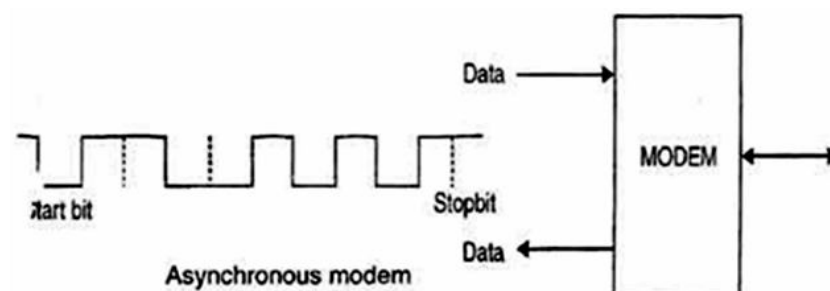
Q28. Explain Asynchronous & Synchronous Modems.

Ans :

(Imp.)

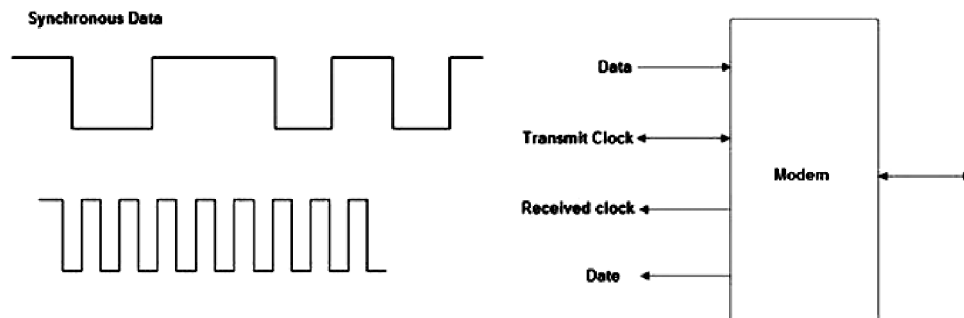
Asynchronous Modem

1. Asynchronous modems can handle data bytes with start and stop bits.
2. There is no separate timing signal or clock between the modem and the DTE.
3. The internal timing pulses are synchronized repeatedly to the leading edge of the start pulse.



Synchronous Modem

1. Synchronous modems can handle a continuous stream of data bits but requires a clock signal.
2. The data bits are always synchronized to the clock signal.
3. There are separate clocks for the data bits being transmitted and received.
4. For synchronous transmission of data bits, the DTE can use its internal clock and supply the same to the modem.

**Modulation techniques used for Modem**

The basic modulation techniques used by a modem to convert digital data to analog signals are :

- Amplitude shift keying (ASK)
- Frequency shift keying (FSK)
- Phase shift keying (PSK)
- Differential PSK (DPSK)

These techniques are known as the binary continuous wave (CW) modulation.

1. Modems are always used in pairs. Any system whether simplex, half duplex or full duplex requires a modem at the transmitting as well as the receiving end.
2. Thus a modem acts as the electronic bridge between two worlds - the world of purely digital signals and the established analog world.

1.9 RS232 INTERFACING

Q29. What is RS-232C?

(OR)

Explain the concept of RS-232 interfacing.

Ans :

(Imp.)

RS-232C is a long-established standard ("C" is the current version) that describes the physical interface and protocol for relatively low-speed serial data communication Networks between computers and related devices.

RS-232C is the interface that your computer uses to talk to and exchange data with your modem and other serial devices. RS-232C is the interface between your Communication networks and other communication networks.

Somewhere in your PC, typically on a Universal Asynchronous Receiver/Transmitter (UART) chip on your motherboard, the data from your computer is transmitted to an internal or external modem (or other serial device) from its Data Terminal Equipment (DTE) interface. Since data in your computer flows along parallel circuits and serial devices can handle only one bit at a time, the UART chip converts the groups of bits in parallel to a serial stream of bits. As your PC's DTE agent, it also communicates with the modem or other serial device, which, in accordance with the RS-232C standard, has a complementary interface called the Data Communications Equipment (DCE) interface. RTS/CTS is the way the DTE indicates that it is ready to transmit data and the way the DCW indicates that it is ready to accept data

RS232C, a standard interface approved by the Electronic Industries Alliance (EIA) for connecting serial devices. In 1987, the EIA released a new version of the standard and changed the name to EIA-232-D. And in 1991, the EIA teamed up with Telecommunications Industry association (TIA) and issued a new version of the standard called EIA/TIA-232-E. Many people, however, still refer to the standard as RS-232C, or just RS-232.

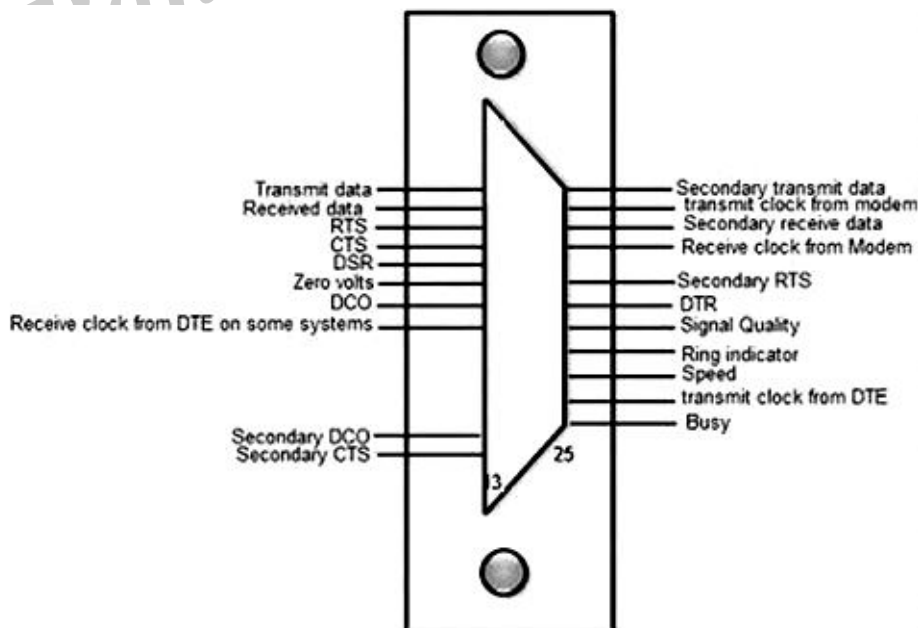
Almost all modems conform to the EIA-232 standard and most personal computers have an EIA-232 port for connecting a modem or other device. In addition to modems, many display screens, mice, and serial printers are designed to connect to a EIA-232 port. In EIA-232 parlance, the device that connects to the interface is called a Data Communications Equipment (DCE) and the device to which it connects (e.g., the computer) is called a Data Terminal Equipment (DTE).

The EIA-232 standard supports two types of connectors — a 25-pin D-type connector (DB-25) and a 9-pin D-type connector (DB-9). The type of serial communications used by PCs requires only 9 pins so either type of connector will work equally well.

Although EIA-232 is still the most common standard for serial communication, the EIA has recently defined successors to EIA-232 called RS-422 and RS-423. The new standards are backward compatible so that RS-232 devices can connect to an RS-422 port.

RS-232 Mechanical Specification

There is a standardized pin out for RS-232 on a DB25 connector, as shown in Figure.

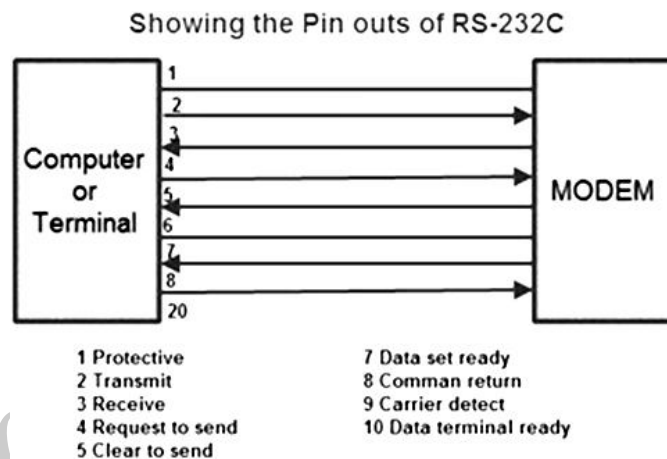


The essential feature of RS-232 is that the signals are carried as single voltages referred to a common ground on pin 7. In its simplest form, the RS-232:C interface consist of only two wires for data and ground. The ground is the absolute voltage reference for all the interface circuitry, the point in the circuit from which all voltages are measured. Data on pin 2 of the DTE is transmitted, while the same data on pin 2 of a DCE (modem) is received data as shown in Figure.

Data is transmitted and received on pins 2 and 3, respectively. Data Set Ready (DSR) is an indication from the Data set (the modem or DSU/CSU) that it is on. Similarly, DTR indicates that the DTE is on. Carrier Detect (CD) indicates that carrier for the transmission data is on.

Pins 4 and 5 carry the Request to Send (RTS) and Clear to Send (CTS) signals. In most situations, RTS and CTS are constantly on the communication session. However, where the DTE is connected to a multipoint line, RTS is used to turn the carrier on the modem on and off. On a multipoint line, it is imperative that only one station is transmitting at a time. When a station wants to transmit, it raises RTS. The modem turns on carrier, typically waits a few milliseconds for carrier to stabilize, and raises CTS.

The DTE transmits when it sees CTS up. When the station has finished its transmission, it drops RTS and the modem drops CTS and carrier together.



Terminals and modems usually communicate bidirectional. Bidirectional interchange between the two devices is directly analogous to the connection of two telephones. The differences between the DTE and DCE are that DTEs transmit on pin 2 and receive on pin 3. DCEs transmit on pin 3 and receive on pin 2. When the modem is communicating with another modem, three essential links are established. These are:

➤ **DTE to DCE** The DTE or terminal and modem or DCE talk to each other.

DCE to DCE The two modems on the link talk to each other. When a DCE to DCE connection is established the modem will send a connect message to the computer. The connect message specifies the computer about the baud rate being used by the two modems for communication.

In case of modems with dissimilar baud rate, the modem of higher baud rate is set to send the data at the same rate of the other modem using the computer's communication program.

DTE to DCE The host computer talks with its modem. The output signal level usually swings between +12v and -12v. RS-232 is simple, universal, well understood and supported everywhere. However, it has some serious shortcomings as an electrical interface.

First, the interface presupposes a common ground between the DTE and DCE. This is a reasonable assumption where a short cable connects a DTE and DCE in the same room, but with longer lines and

connections between devices that may be on different electrical buses, this may not be true. Second, a signal on a single line is impossible to screen effectively for noise. By screening the entire cable one can reduce the influence of outside noise, but internally generated noise remains a problem. As the baud rate and line length increase, the effect of capacitance between the cables introduces serious cross talk until a point is reached where the data itself is unreadable. Low capacitance cable can reduce cross talk. Also, as it is the higher frequencies that are the problem control of slew rate in the signal (i.e. making the signal more rounded, rather than square) also decreases the cross talk. The original specifications for RS-232 had no specification for maximum slew rate.

The standards for RS-232 and similar interfaces usually restrict RS-232 to 20 kbps or less and line lengths of 15m (50 ft) or less. These restrictions are mostly throwbacks to the days when 20 kbps were considered a very high line speed, and cables were thick, with high capacitance. However, in practice, RS-232 is far more robust than the traditional specified limits of 20 kbps over a 15m line would imply. Most 56 kbps DSUs are supplied with both V.35 and RS-232 ports because RS-232 is perfectly adequate at speeds up to 200 kbps. The 15 m limitation for cable length can be stretched to about 30 m for ordinary cable, if well screened and grounded, and about 100 m if the cable is low capacitance as well.

Rahul Publications

UNIT II

Datalink Layer: Error detection and correction, CRC, Hamming code, Flow Control and Error control, Stop and Wait protocol, Sliding Window protocol - go back-N ARQ - selective repeat ARQ.

MAC Layer: LAN - Pure and Slotted ALOHA, Ethernet IEEE 802.3 LAN Ethernet Efficiency Calculation, Bridges. ARP, RARP

2.1 DATA LINK LAYER

Q1. What is DLL? State the functions of DLL.

Ans :

(Imp.)

Meaning

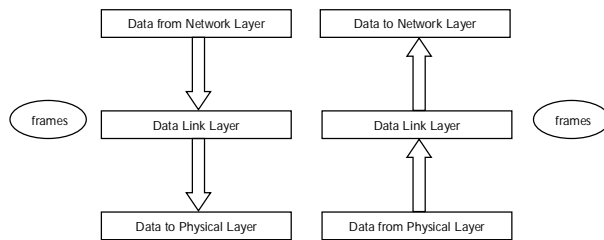
Data link layer is most reliable node to node delivery of data. It forms frames from the packets that are received from network layer and gives it to physical layer. It also synchronizes the information which is to be transmitted over the data. Error controlling is easily done. The encoded data are then passed to physical.

Error detection bits are used by the data link layer. It also corrects the errors. Outgoing messages are assembled into frames. Then the system waits for the acknowledgements to be received after the transmission. It is reliable to send message.

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 few thousand bytes) and transmit the frames sequentially. If the service is reliable, the receiver confirms correct receipt of each frame by send back an acknowledgement frame.

Functions

1. **Framing:** Frames are the streams of bits received from the network layer into manageable data units. This division of stream of bits is done by Data Link Layer.
2. **Physical Addressing:** The Data Link layer adds a header to the frame in order to define physical address of the sender or receiver of the frame, if the frames are to be distributed to different systems on the network.
3. **Flow Control:** A flow control mechanism to avoid a fast transmitter from running a slow receiver by buffering the extra bit is provided by flow control. This prevents traffic jam at the receiver side.
4. **Error Control:** Error control is achieved by adding a trailer at the end of the frame. Duplication of frames are also prevented by using this mechanism. Data Link Layers adds mechanism to prevent duplication of frames.
5. **Access Control:** Protocols of this layer determine which of the devices has control over the link at any given time, when two or more devices are connected to the same link.



Q2. Explain design issues with data link layer.

Ans :

The 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, the flow regulation and the error handling are integrated.

Broadcast networks have an additional issue in the data link layer: How to control access to the shared channel. A special sublayer of the data link layer, the Medium Access Control (MAC) sublayer, deals with this problem.

Since the physical layer merely accepts and transmits a stream of bits without any regard to meaning or structure, it is up to the data link layer to create and recognize frame boundaries. This can be accomplished by attaching special bit patterns to the beginning and end of the frame. If these bit patterns can accidentally occur in data, special care must be taken to make sure these patterns are not incorrectly interpreted as frame delimiters. The four framing methods that are widely used are

- i) Character count
- ii) Starting and ending characters, with character stuffing
- iii) Starting and ending flags, with bit stuffing
- iv) Physical layer coding violations

i) Character Count

This method uses a field in the header to specify the number of characters in the frame. When the data link layer at the destination sees the character count, it knows how many characters follow, and hence where the end of the frame is.

The disadvantage is that if the count is garbled by a transmission error, the destination will lose synchronization and will be unable to locate the start of the next frame. So, this method is rarely used.

ii) Character stuffing

In the second method, each frame starts with the ASCII character sequence DLE STX and ends with the sequence DLE ETX. (where DLE is Data Link Escape, STX is Start of Text and ETX is End of Text.) This method overcomes the drawbacks of the character count method. If the destination ever loses synchronization, it only has to look for DLE STX and DLE ETX characters. If however, binary data is being transmitted then there exists a possibility of the characters DLE STX and DLE ETX occurring in the data. Since this can interfere with the framing, a technique called character stuffing is used. The sender's data link layer inserts an ASCII DLE character just before the DLE character in the data. The receiver's data link layer removes this DLE before this data is given to the network layer. However character stuffing is closely associated with 8-bit characters and this is a major hurdle in transmitting arbitrary sized characters.

iii) Bit stuffing

The third method allows data frames to contain an arbitrary number of bits and allows character codes with an arbitrary number of bits per character. At the start and end of each frame is a flag byte consisting of the special bit pattern 01111110. Whenever the sender's data link layer encounters five consecutive 1s in the data, it automatically stuffs a zero bit into the outgoing bit stream. This technique is called bit stuffing. When the receiver sees five consecutive 1s in the incoming data stream, followed by a zero bit, it automatically destuffs the 0 bit. The boundary between two frames can be determined by locating the flag pattern.

iv) Physical layer coding violations

The final framing method is physical layer coding violations and is applicable to networks in which the encoding on the physical medium contains some redundancy. In such cases normally, a 1 bit is a high-low pair and a 0 bit is a low-high pair. The combinations of low-low and high-high which are not used for data may be used for marking frame boundaries.

2.1.1 Error Detection and Correction

Q3. What is Error Correction and Detection?

Ans :

Error detection and correction has great practical importance in maintaining data (information) integrity across noisy Communication Networks channels and less-than-reliable storage media.

I) Error Correction

Send additional information so incorrect data can be corrected and accepted. Error correction is the additional ability to reconstruct the original, error-free data.

There are two basic ways to design the channel code and protocol for an error correcting system:

- **Automatic Repeat-Request (ARQ):** The transmitter sends the data and also an error detection code, which the receiver uses to check for errors, and request retransmission of erroneous data. In many cases, the request is implicit; the receiver sends an acknowledgment (ACK) of correctly received data, and the transmitter resends anything not acknowledged within a reasonable period of time.
- **Forward Error Correction (FEC):** The transmitter encodes the data with an error-correcting code (ECC) and sends the coded message. The receiver never sends any messages back to the transmitter. The receiver decodes what it receives into the "most likely" data. The codes are designed so that it would take an "unreasonable" amount of noise to trick the receiver into misinterpreting the data.

II) Error Detection

Send additional information so incorrect data can be detected and rejected. Error detection is the ability to detect the presence of errors caused by noise or other impairments during transmission from the transmitter to the receiver.

Schemes

In telecommunication, a redundancy check is extra data added to a message for the purposes of error detection. Several schemes exist to achieve error detection, and are generally quite simple. All error detection codes transmit more bits than were in the original data. Most codes are "systematic": the transmitter sends a fixed number of original data bits, followed by fixed number of check bits usually referred to as redundancy which are derived from the data bits by some deterministic algorithm.

The receiver applies the same algorithm to the received data bits and compares its output to the received check bits; if the values do not match, an error has occurred at some point during the transmission. In a system that uses a "non-systematic" code, such as some raptor codes, data bits are transformed into at least as many code bits, and the transmitter sends only the code bits.

Repetition Schemes

Variations on this theme exist. Given a stream of data that is to be sent, the data is broken up into blocks of bits, and in sending, each block is sent some predetermined number of times. For example, if we want to send "1011", we may repeat this block three times each. Suppose we send "1011 1011 1011", and this is received as "1010 1011 1011".

As one group is not the same as the other two, we can determine that an error has occurred. This scheme is not very efficient, and can be susceptible to problems if the error occurs in exactly the same place for each group e.g. "1010 1010 1010" in the example above will be detected as correct in this scheme. The scheme however is extremely simple, and is in fact used in some transmissions of numbers stations.

Parity Schemes

A parity bit is an error detection mechanism. A parity bit is an extra bit transmitted with a data item, chosen to give the resulting bit even or odd parity. Parity refers to the number of bits set to 1 in the data item. There are 2 types of parity

- **Even parity :** an even number of bits are 1
Even parity - data: 10010001, parity bit 1.
- **Odd parity :** an odd number of bits are 1
Odd parity - data: 10010111, parity bit 0.

The stream of data is broken up into blocks of bits, and the number of 1 bits is counted. Then, a "parity bit" is set (or cleared) if the number of one bits is odd (or even). This scheme is called even parity; odd parity can also be used. There is a limitation to parity schemes. A parity bit is only guaranteed to detect an odd number of bit errors (one, three, five, and so on). If an even number of bits (two, four, six and so on) are flipped, the parity bit appears to be correct, even though the data is corrupt. For example

- **Original data and parity:** 10010001 + 1 (even parity)
- **Incorrect data:** 10110011 + 1 (even parity!)
- Parity usually used to catch one-bit errors

Checksum

A checksum of a message is an arithmetic sum of message code words of a certain word length, for example byte values, and their carry value. The sum is negated by means of ones-complement, and stored or transferred as an extra code word extending the message. On the receiver side, a new checksum may be calculated, from the extended message.

If the new checksum is not 0, error are detected. Checksum schemes include parity bits, check digits, and longitudinal redundancy check. Suppose we have a fairly long message, which can reasonably be divided into shorter words (a 128 byte message, for instance). We can introduce an accumulator with the same width as a word (one byte, for instance), and as each word comes in, add it to the accumulator.

When the last word has been added, the contents of the accumulator are appended to the message (as a 129th byte, in this case). The added word is called a checksum. Now, the receiver performs the same operation, and checks the checksum. If the checksums agree, we assume the message was sent without error.

Hamming Distance Based Checks

If we want to detect d bit errors in an n bit word we can map every n bit word into a bigger $n+d+1$ bit word so that the minimum Hamming distance between each valid mapping is $d+1$. This way, if one receives $n+d+1$ bit word that doesn't

match any word in the mapping (with a Hamming distance $x \leq d+1$ from any word in the mapping) it can successfully detect it as an errored word. Even more, d or fewer errors will never transform a valid word into another, because the Hamming distance between each valid word is at least $d+1$, and such errors only lead to invalid words that are detected correctly.

Given a stream of $m \times n$ bits, we can detect $x \leq d$ bit errors successfully using the above method on every n bit word. In fact, we can detect a maximum of $m \times d$ errors if every n word is transmitted with maximum d errors. The Hamming distance between two bit strings is the number of bits you have to change to convert one to the other. The basic idea of an error correcting code is to use extra bits to increase the dimensionality of the hypercube, and make sure the Hamming distance between any two valid points is greater than one.

- If the Hamming distance between valid strings is only one, a single bit error results in another valid string. This means we can't detect an error.
- If it's two, then changing one bit results in an invalid string, and can be detected as an error. Unfortunately, changing just one more bit can result in another valid string, which means we can't detect which bit was wrong; so we can detect an error but not correct it.
- If the Hamming distance between valid strings is three, then changing one bit leaves us only one bit away from the original error, but two bits away from any other valid string. This means if we have a one-bit error, we can figure out which bit is the error; but if we have a two-bit error, it looks like one bit from the other direction. So we can have single bit correction, but that's all.
- Finally, if the Hamming distance is four, then we can correct a single-bit error and detect a double-bit error. This is frequently referred to as a SECDED (Single Error Correct, Double Error Detect) scheme.

Cyclic Redundancy Checks

For CRC following some of Peterson & Brown's notation here.....

- **k** is the length of the message we want to send, i.e., the number of information bits.
- **n** is the total length of the message we will end up sending the information bits followed by the check bits. Peterson and Brown call this a *code polynomial*.
- **n-k** is the number of check bits. It is also the degree of the generating polynomial. The basic (mathematical) idea is that we're going to pick the n-k check digits in such a way that the code polynomial is divisible by the generating polynomial. Then we send the data, and at the other end we look to see whether it's still divisible by the generating polynomial; if it's not then we know we have an error, if it is, we hope there was no error. The way we calculate a CRC is we establish some predefined n-k + 1 bit number P (called the Polynomial, for reasons relating to the fact that modulo-2 arithmetic is a special case of polynomial arithmetic). Now we append n-k 0's to our message, and divide the result by P using modulo-2 arithmetic. The remainder is called the Frame Check Sequence. Now we ship off the message with the remainder appended in place of the 0's. The receiver can either recompute the FCS or see if it gets the same answer, or it can just divide the whole message (including the FCS) by P and see if it gets a remainder of 0. As an example, let's set a 5-bit polynomial of 11001, and compute the CRC of a 16 bit message:

```

-----
11001)10011101010101100000
11001
-----
1010101010101100000
11001
-----
110001010101100000
11001
-----
00011010101100000
11001
-----

```

```

0011101100000
11001
-----
100100000
11001
-----
10110000
11001
-----
1111000
11001
-----
11100
11001
-----
0101

```

In division don't bother to keep track of the quotient; we don't care about the quotient. Our only goal here is to get the remainder (0101), which is the FCS. CRC's can actually be computed in hardware using a shift register and some number of exclusive-or gates.

2.2 CYCLIC REDUNDANCY CHECK (CRC)

Q4. Define CRC. State the process of CRC.

Ans : (Imp.)

Cyclic Redundancy Check (CRC) An error detection mechanism in which a special number is appended to a block of data in order to detect any changes introduced during storage (or transmission). The CRC is recalculated on retrieval (or reception) and compared to the value originally transmitted, which can reveal certain types of error. For example, a single corrupted bit in the data results in a one-bit change in the calculated CRC, but multiple corrupt bits may cancel each other out.

A CRC is derived using a more complex algorithm than the simple CHECKSUM, involving MODULO ARITHMETIC (hence the 'cyclic' name) and treating each input word as a set of coefficients for a polynomial.

CRC is more powerful than VRC and LRC in detecting errors.

- It is not based on binary addition like VRC and LRC. Rather it is based on binary division.
- At the sender side, the data unit to be transmitted is divided by a predetermined divisor (binary number) in order to obtain the remainder. This remainder is called CRC.
- The CRC has one bit less than the divisor. It means that if CRC is of n bits, divisor is of $n + 1$ bit.
- The sender appends this CRC to the end of data unit such that the resulting data unit becomes exactly divisible by predetermined divisor i.e. remainder becomes zero.
- At the destination, the incoming data unit i.e. data + CRC is divided by the same number (predetermined binary divisor).
- If the remainder after division is zero then there is no error in the data unit & receiver accepts it.
- If remainder after division is not zero, it indicates that the data unit has been damaged in transit and therefore it is rejected.
- This technique is more powerful than the parity check and checksum error detection.

CRC is based on binary division. A sequence of redundant bits called CRC or CRC remainder is appended at the end of a data unit such as byte.

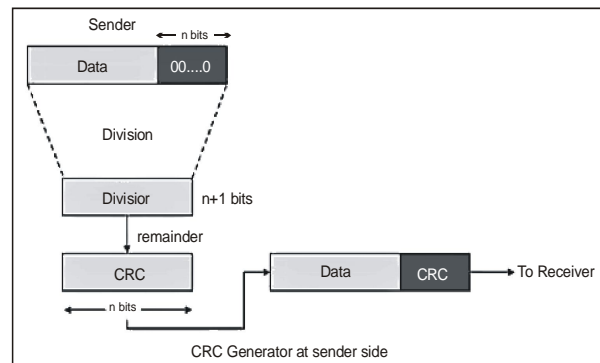
A CRC will be valid if and only if it satisfies the following requirements:

1. It should have exactly one less bit than divisor.
2. Appending the CRC to the end of the data unit should result in the bit sequence which is exactly divisible by the divisor.

Process

The various steps followed in the CRC method are

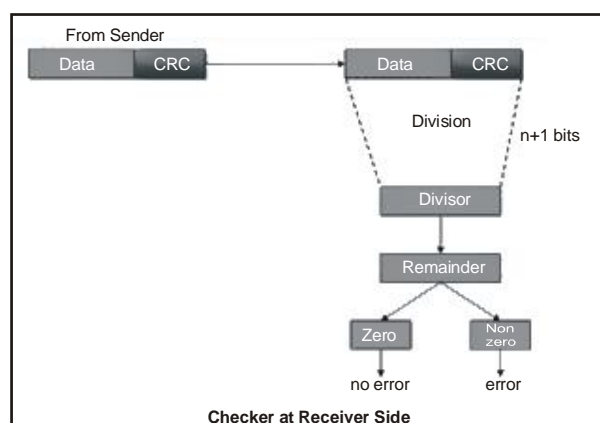
1. A string of n as is appended to the data unit. The length of predetermined divisor is $n + 1$.
2. The newly formed data unit i.e. original data + string of n as are divided by the divisor using binary division and remainder is obtained. This remainder is called CRC.



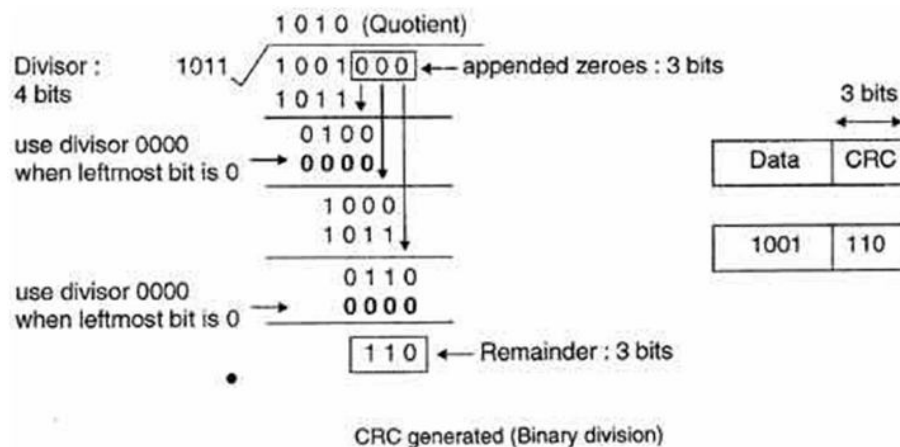
1. Now, string of n 0's appended to data unit is replaced by the CRC remainder (which is also of n bit).
2. The data unit + CRC is then transmitted to receiver.
3. The receiver on receiving it divides data unit + CRC by the same divisor & checks the remainder.
4. If the remainder of division is zero, receiver assumes that there is no error in data and it accepts it.
5. If remainder is non-zero then there is an error in data and receiver rejects it.

For example, if data to be transmitted is 1001 and predetermined divisor is 1011. The procedure given below is used:

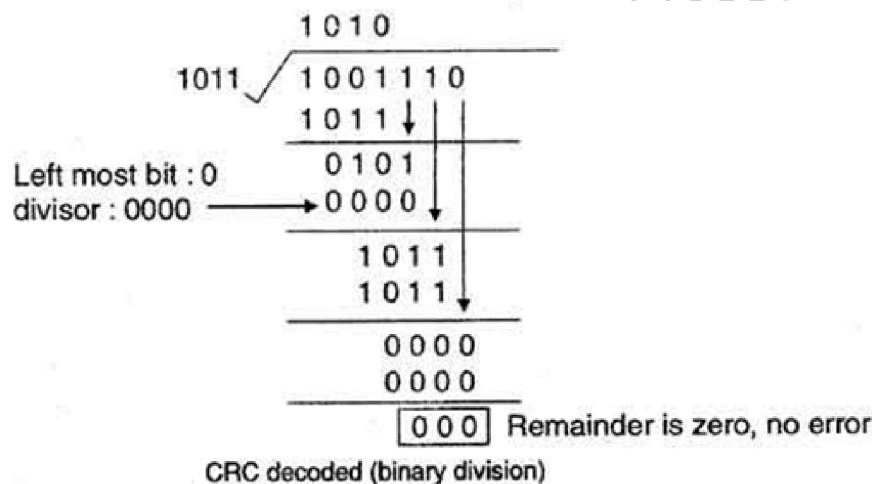
String of 3 zeroes is appended to 1011 as divisor is of 4 bits. Now newly formed data is 1011000.



Data unit 1011000 is divided by 1011.



- During this process of division, whenever the leftmost bit of dividend or remainder is 0, we use a string of 0s of same length as divisor. Thus in this case divisor 1011 is replaced by 0000.
 - At the receiver side, data received is 1001110.
 - This data is again divided by a divisor 1011.
- The remainder obtained is 000; it means there is no error.

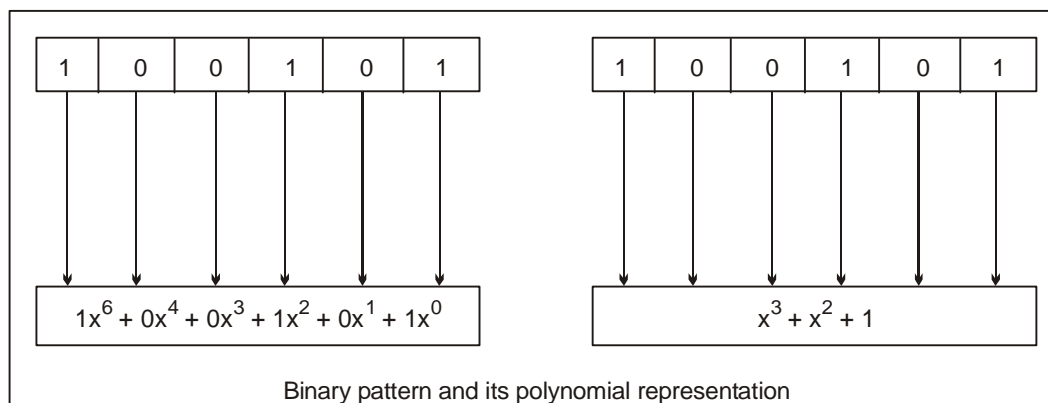


- CRC can detect all the burst errors that affect an odd number of bits.
- The probability of error detection and the types of detectable errors depends on the choice of divisor.
- Thus two major requirement of CRC are:
 - CRC should have exactly one bit less than divisor.
 - Appending the CRC to the end of the data unit should result in the bit sequence which is exactly divisible by the divisor.

Polynomial codes

- A pattern of 0's and 1s can be represented as a polynomial with coefficient of 0 and 1.
- Here, the power of each term shows the position of the bit and the coefficient shows the values of the bit.

For example, if binary pattern is 100101, its corresponding polynomial representation is $x^5 + x^2 + 1$. Figure shows the polynomial where all the terms with zero coefficient are removed and x^j is replaced by x and x^0 by 1.



- The benefit of using polynomial codes is that it produces short codes. For example here a 6-bit pattern is replaced by 3 terms.
- In polynomial codes, the degree is 1 less than the number of bits in the binary pattern. The degree of polynomial is the highest power in polynomial. For example as shown in fig degree of polynomial $x^5 + x^2 + 1$ are 5. The bit pattern in this case is 6.
- Addition of two polynomials is based on modulo-2 method. In such as case, addition and subtraction is same.
- Addition or subtraction is done by combining terms and deleting pairs of identical terms. For example adding $x^5 + x^4 + x^2$ and $x^6 + x^4 + x^2$ give $x^6 + x^5$. The terms x^4 and x^2 are deleted.
- If three polynomials are to be added and if we get a same term three times, a pair of them is detected and the third term is kept. For example, if there is x^2 three times then we keep only one x^2
- In case of multiplication of two polynomials, their powers are added. For example, multiplying $x^5 + x^3 + x^2 + x$ with $x^2 + x + 1$ yields:
- $(x^5 + x^3 + x^2 + x)(x^2 + x + 1)$
- $= x^7 + x^6 + x^5 + x^5 + x^4 + x^3 + x^4 + x^3 + x^2 + x^2 + x$
- $= x^7 + x^6 + x^3 + x$
- In this, first polynomial is multiplied by all terms of second. The result is then simplified and pairs of equal terms are deleted.
- In case of division, the two polynomials are divided as per the rules of binary division, until the degree of dividend is less than that of divisor.

CRC generator using polynomials

- If we consider the data unit 1001 and divisor or polynomial generator 1011 their polynomial representation is:

Diagram illustrating the division process:

$$\begin{array}{r}
 \text{Divisor } x^3 + x + 1 \overline{) x^6 + x^4 + x^3} \\
 \underline{x^6 + x^4 + x^3} \\
 x^4 + x^2 + x \\
 \underline{x^4 + x^2 + x} \\
 0
 \end{array}$$

The result shows the quotient $x^3 + x$ and the remainder $x^2 + x$. The remainder is boxed and labeled "Remainder (degree is less than that of divisor)".

The final data unit to be transmitted is shown as:

$x^6 + x^3$	$x^2 + x$
Data	Remainder

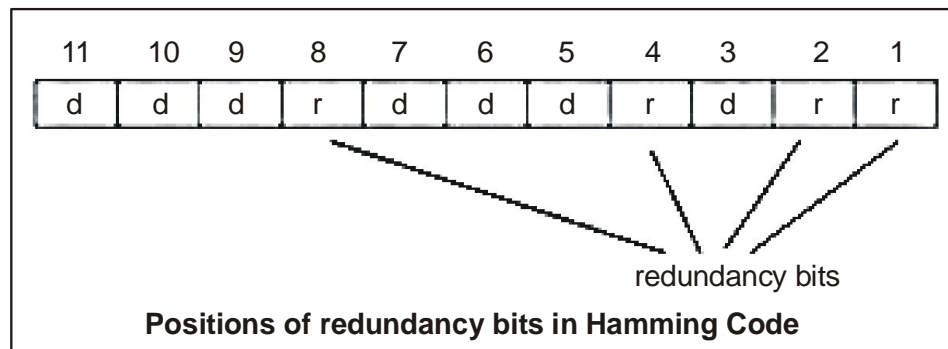
- Now string of n 0s (one less than that of divisor) is appended to data. Now data is 1001000 and its corresponding polynomial representation is $x^6 + x^3$.
- The division of $x^6 + x^3$ by $x^3 + x + 1$ is shown in fig.
- The polynomial generator should have following properties:
 - It should have at least two terms.
 - The coefficient of the term x^0 should be 1.
 - It should not be divisible by x .
 - It should be divisible by $x + 1$.

Name	Polynomial	Application
CRC-8	$x^3 + x^2 + x + 1$	ATM header
CRC-10	$x^{10} + x^9 + x^5 + x^4 + x^2 + 1$	ATM AAL
CRC-16	$x^{16} + x^{12} + x^5 + 1$	HDLC
CRC-32	$x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1$	LANs

Rahul Publications

The redundancy bits are placed at the positions which correspond to the power of 2.

For example in case of 7 bit data, 4 redundancy bits are required, so making total number of bits as 11. The redundancy bits are placed in position 1, 2, 4 and 8 as shown in figure.



Generating Parity Information

- In Hamming code, each r bit is the VRC for one combination of data bits. r_1 is the VRC bit for one combination of data bits, r_2 is the VRC for another combination of data bits and so on.
- Each data bit may be included in more than one VRC calculation.
- r_1 bit is calculated using all bits positions whose binary representation includes a 1 in the rightmost position.
- r_2 bit calculated using all the bit positions with a 1 in the second position and so on.

Therefore the various r bits are parity bits for different combination of bits.

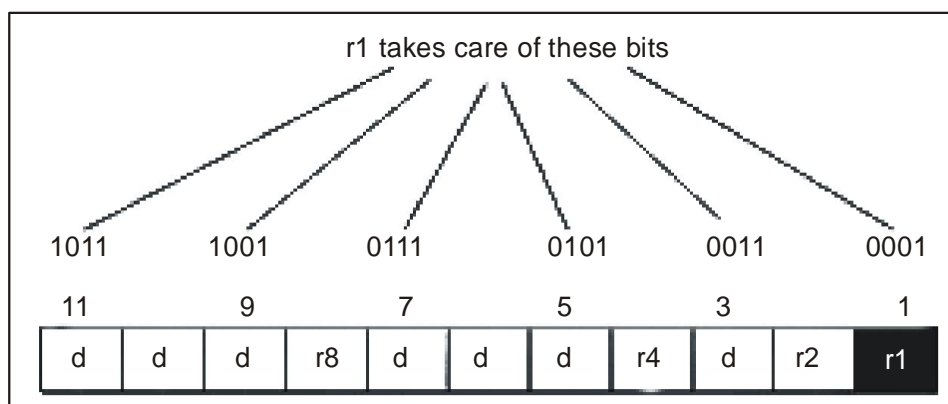
The various combinations are:

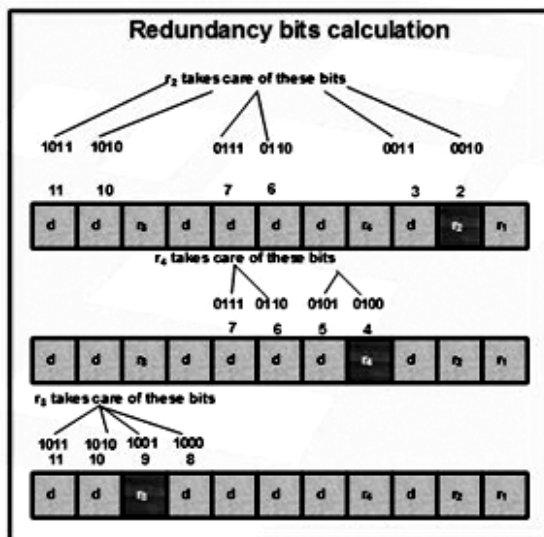
r_1 : bits 1,3,5, 7, 9, 11

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

r_4 : bits 4, 5, 6, 7

r_8 : bits 8, 9, 10, 11





Example of Hamming Code Generation

Suppose a binary data 1001101 is to be transmitted. To implement hamming code for this, following steps are used:

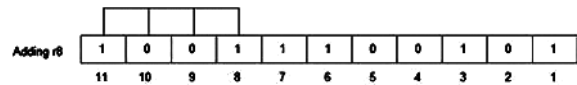
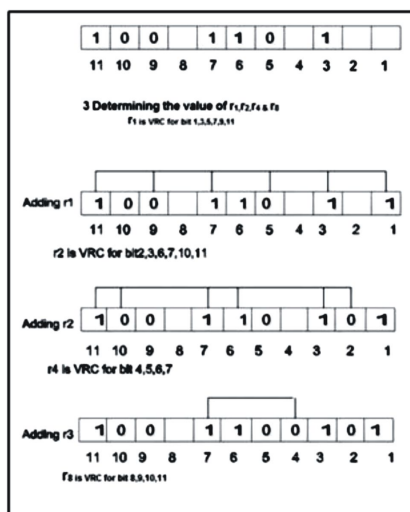
1. Calculating the number of redundancy bits required. Since number of data bits is 7, the value of r is calculated as

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

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

Therefore no. of redundancy bits = 4

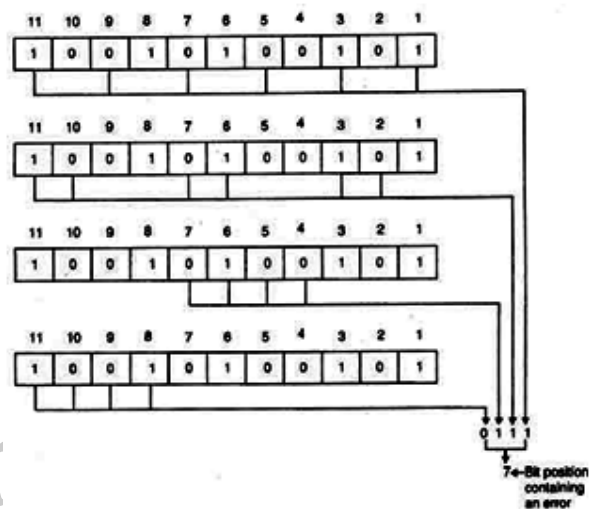
2. Determining the positions of various data bits and redundancy bits. The various r bits are placed at the position that corresponds to the power of 2 i.e. 1, 2, 4, 8



3. Thus data 1 0 0 1 1 1 0 0 1 0 1 with be transmitted.

Error Detection & Correction

Considering a case of above discussed example, if bit number 7 has been changed from 1 to 0. The data will be erroneous.



Data sent: 1 0 0 1 1 1 0 0 1 0 1

Data received: 1 0 0 1 0 1 0 0 1 0 1 (seventh bit changed).

The receive takes the transmission and recalculates four new VRCs using the same set of bits used by sender plus the relevant parity (r) bit for each set as shown in fig.

Then it assembles the new parity values into a binary number in order of r position (r_8, r_4, r_2, r_1). In this example, this step gives us the binary number 0111. This corresponds to decimal 7.

Therefore bit number 7 contains an error. To correct this error, bit 7 is reversed from 0 to 1.

Q6. Write and explain the algorithm for Hamming Code ?

Ans :

The Hamming Code is simply the use of extra parity bits to allow the identification of an error.

1. Write the bit positions starting from 1 in binary form (1, 10, 11, 100, etc).
2. All the bit positions that are a power of 2 are marked as parity bits (1, 2, 4, 8, etc).
3. All the other bit positions are marked as data bits.
4. Each data bit is included in a unique set of parity bits, as determined its bit position in binary form.

- Parity bit 1 covers all the bits positions whose binary representation includes a 1 in the least significant position (1, 3, 5, 7, 9, 11, etc).
- Parity bit 2 covers all the bits positions whose binary representation includes a 1 in the second position from the least significant bit (2, 3, 6, 7, 10, 11, etc).
- Parity bit 4 covers all the bits positions whose binary representation includes a 1 in the third position from the least significant bit (4–7, 12–15, 20–23, etc).
- Parity bit 8 covers all the bits positions whose binary representation includes a 1 in the fourth position from the least significant bit (8–15, 24–31, 40–47, etc).
- In general each parity bit covers all bits where the bitwise AND of the parity position and the bit position is non-zero.

5. Since we check for even parity set a parity bit to 1 if the total number of ones in the positions it checks is odd.

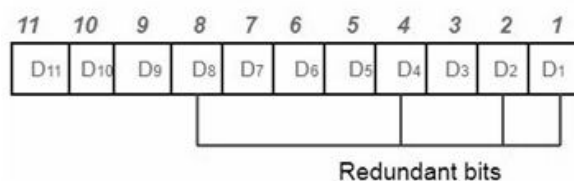
6. Set a parity bit to 0 if the total number of ones in the positions it checks is even.

Determining the position of redundant bits –

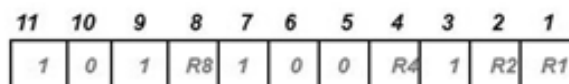
These redundancy bits are placed at the positions which correspond to the power of 2.

As in the above example:

1. The number of data bits = 7
2. The number of redundant bits = 4
3. The total number of bits = 11
4. The redundant bits are placed at positions corresponding to power of 2- 1, 2, 4, and 8.



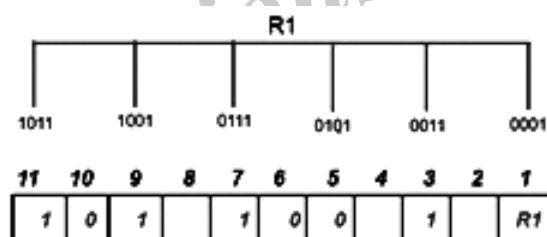
Suppose the data to be transmitted is 1011001, the bits will be placed as follows:



Determining the Parity bits –

1. R₁ bit is calculated using parity check at all the bits positions whose binary representation includes a 1 in the least significant position.

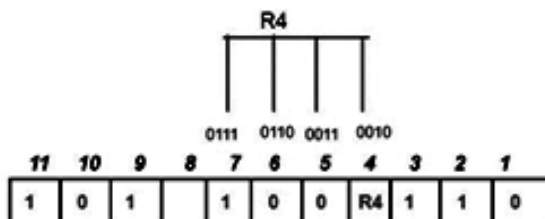
R₁: bits 1, 3, 5, 7, 9, 11



To find the redundant bit R2, we check for even parity. Since the total number of 1's in all the bit positions corresponding to R2 is an odd number the value of R2 (parity bit's value) = 1.

3. R4 bit is calculated using parity check at all the bits positions whose binary representation includes a 1 in the third position from the least significant bit.

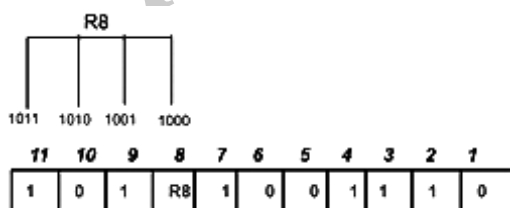
R4: bits 4, 5, 6, 7



To find the redundant bit R4, we check for even parity. Since the total number of 1's in all the bit positions corresponding to R4 is an odd number the value of R4 (parity bit's value) = 1

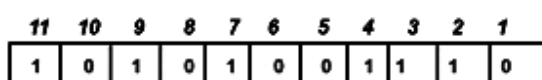
4. R8 bit is calculated using parity check at all the bits positions whose binary representation includes a 1 in the fourth position from the least significant bit.

R8: bit 8,9,10,11



To find the redundant bit R8, we check for even parity. Since the total number of 1's in all the bit positions corresponding to R8 is an even number the value of R8 (parity bit's value) = 0.

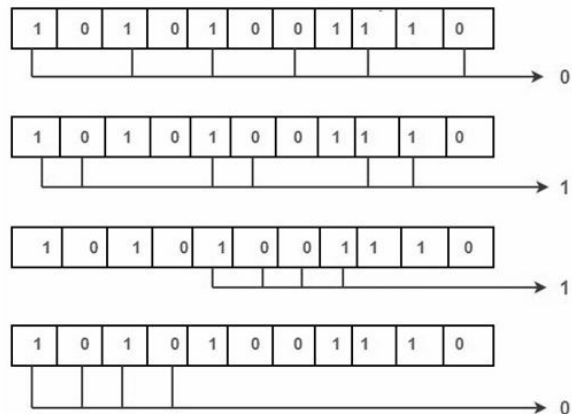
Thus, the data transferred is:



Error detection and correction

Suppose in the above example the 6th bit is changed from 0 to 1 during data transmission, then it gives new parity values in the binary number:

The bits give the binary number as 0110 whose decimal representation is 6. Thus, the bit 6 contains an error. To correct the error the 6th bit is changed from 1 to 0.



2.4 FLOW CONTROL & ERROR CONTROL

2.4.1 Stop and Wait protocol, Sliding Window protocol-go back-N ARQ - selective repeat ARQ

Q6. Explain in detail about the working of flow control?

Ans :

Data-link layer is responsible for implementation of point-to-point.

1. Flow Control
2. Error control Mechanism.

1. Flow Control

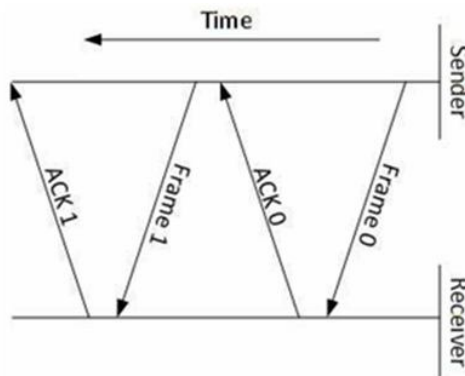
When a data frame (Layer-2 data) is sent from one host to another over a single medium, it is required that the sender and receiver should work at the same speed. That is, sender sends at a speed on which the receiver can process and accept the data. What if the speed (hardware/software) of the sender or receiver differs? If sender is sending too fast the receiver may be overloaded, (swamped) and data may be lost.

Two types of mechanisms can be deployed to control the flow:

- i) **Stop and Wait**
- ii) **Sliding Window**

i) Stop and Wait

This flow control mechanism forces the sender after transmitting a data frame to stop and wait until the acknowledgement of the data-frame sent is received.



ii) Sliding Window

In this flow control mechanism, both sender and receiver agree on the number of data-frames after which the acknowledgement should be sent. As we learnt, stop and wait flow control mechanism wastes resources, this protocol tries to make use of underlying resources as much as possible.

2. Error Control

When data-frame is transmitted, there is a probability that data-frame may be lost in the transit or it is received corrupted. In both cases, the receiver does not receive the correct data-frame and sender does not know anything about any loss. In such case, both sender and receiver are equipped with some protocols which helps them to detect transit errors such as loss of data-frame. Hence, either the sender retransmits the data-frame or the receiver may request to resend the previous data-frame.

Requirements for error control mechanism:

- **Error detection:** The sender and receiver, either both or any, must ascertain that there is some error in the transit.
- **Positive ACK:** When the receiver receives a correct frame, it should acknowledge it.

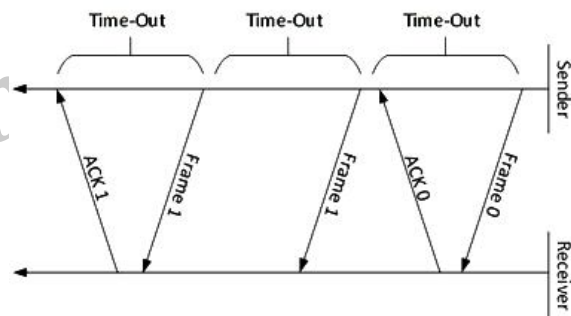
- **Negative ACK:** When the receiver receives a damaged frame or a duplicate frame, it sends a NACK back to the sender and the sender must retransmit the correct frame.

- **Retransmission:** The sender maintains a clock and sets a timeout period. If an acknowledgement of a data-frame previously transmitted does not arrive before the timeout the sender retransmits the frame, thinking that the frame or its acknowledgement is lost in transit.

- There are three types of techniques available which Data-link layer may deploy to control the errors by Automatic Repeat Requests (ARQ):

- i) Stop-and-wait ARQ
- ii) Go-Back-N ARQ
- iii) Selective Repeat ARQ

i) Stop-and-wait ARQ



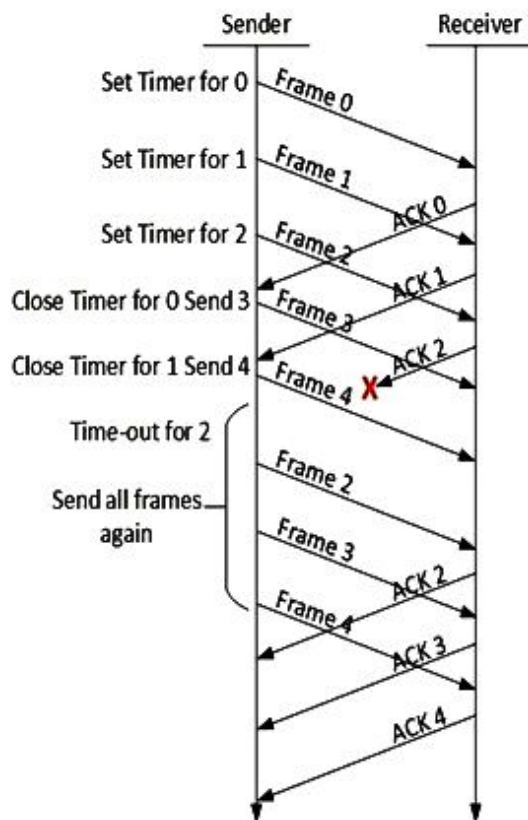
The following transition may occur in Stop-and-Wait ARQ:

- The sender maintains a timeout counter.
- When a frame is sent, the sender starts the timeout counter.
- If acknowledgement of frame comes in time, the sender transmits the next frame in queue.
- If acknowledgement does not come in time, the sender assumes that either the frame or its acknowledgement is lost in transit. Sender retransmits the frame and starts the timeout counter.
- If a negative acknowledgement is received, the sender retransmits the frame.

ii) Go-Back-N ARQ

Stop and wait ARQ mechanism does not utilize the resources at their best. When the

acknowledgement is received, the sender sits idle and does nothing. In Go-Back-N ARQ method, both sender and receiver maintain a window.



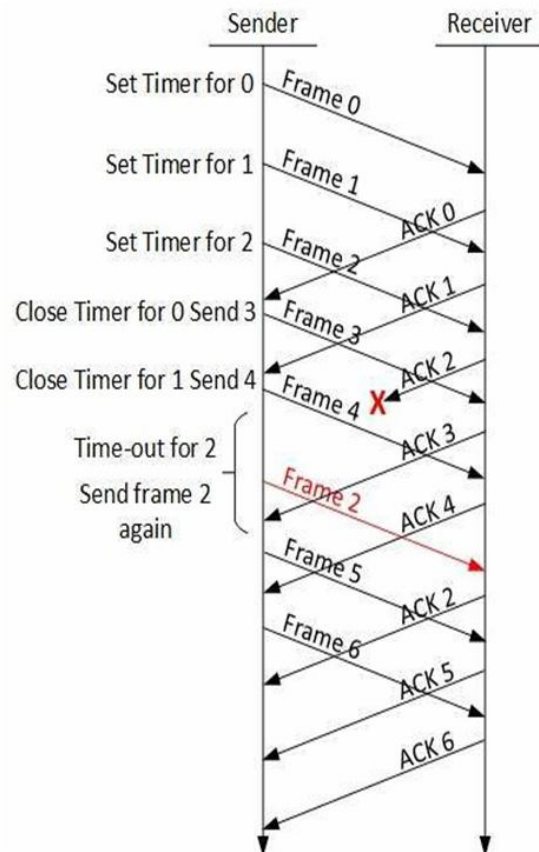
The sending-window size enables the sender to send multiple frames without receiving the acknowledgement of the previous ones. The receiving-window enables the receiver to receive multiple frames and acknowledge them. The receiver keeps track of incoming frame's sequence number.

When the sender sends all the frames in window, it checks up to what sequence number it has received positive acknowledgement. If all frames are positively acknowledged, the sender sends next set of frames. If sender finds that it has received NACK or has not receive any ACK for a particular frame, it retransmits all the frames after which it does not receive any positive ACK.

iii) Selective Repeat ARQ

In Go-back-N ARQ, it is assumed that the receiver does not have any buffer space for its window size and has to process each frame as it

comes. This enforces the sender to retransmit all the frames which are not acknowledged.



In Selective-Repeat ARQ, the receiver while keeping track of sequence numbers, buffers the frames in memory and sends NACK for only frame which is missing or damaged.

The sender in this case, sends only packet for which NACK is received.

Q7. Define HDLC Protocol?

Ans :

The HDLC protocol is a general purpose protocol which operates at the data link layer of the OSI reference model. The protocol uses the services of a physical layer, and provides either a best effort or reliable communications path between the transmitter and receiver (i.e. with acknowledged data transfer). The type of service provided depends upon the HDLC mode which is used.

Each piece of data is encapsulated in an HDLC frame by adding a trailer and a header. The

header contains an HDLC address and an HDLC control field. The trailer is found at the end of the frame, and contains a Cyclic Redundancy Check (CRC) which detects any errors which may occur during transmission. The frames are separated by HDLC flag sequences which are transmitted between each frame and whenever there is no data to be transmitted.

It is a transmission protocol used at the data link layer (layer 2) of the OSI seven layer model for data communications. The HDLC protocol embeds information in a data frame that allows devices to control data flow and correct errors. HDLC is an ISO standard developed from the Synchronous Data Link Control (SDLC) standard proposed by IBM in the 1970's.

For any HDLC communications session, one station is designated primary and the other secondary. A session can use one of the following connection modes, which determine how the primary and secondary stations interact.

- **Normal unbalanced** : The secondary station responds only to the primary station.

Asynchronous : The secondary station can initiate a message.

- **Asynchronous balanced** : Both stations send and receive over its part of a duplex line. This mode is used for X.25 packet-switching networks.

There are three fundamental types of HDLC frames.

- Information frames, or **I-frames**, transport user data from the network layer. In addition they can also include flow and error control information piggybacked on data.
- Supervisory Frames, or **S-frames**, are used for flow and error control whenever piggybacking is impossible or inappropriate, such as when a station does not have data to send. S-frames **do not** have information fields.

Unnumbered frames, or **U-frames**, are used for various miscellaneous purposes, including link management. Some U-frames contain an information field, depending on the type.

Q8. Explain and discuss the Architecture of HDLC Protocol ?

Ans :

Short for High-level Data Link Control, a transmission protocol used at the data link layer (layer 2) of the OSI seven layer models for data communications. The HDLC protocol embeds information in a data frame that allows devices to control data flow and correct errors. HDLC is an ISO standard developed from the Synchronous Data Link Control (SDLC) standard proposed by IBM in the 1970's. HDLC NRM (also known as SDLC).

HDLC is a bit oriented protocol that supports both half-duplex and full-duplex communication over point to point & multipoint link.

For any **HDLC communications session**, one station is designated primary and the other secondary. A session can use one of the following connection modes, which determine how the primary and secondary stations interact.

- **Normal unbalanced**: The secondary station responds only to the primary station.

- **Asynchronous**: The secondary station can initiate a message.

- **Asynchronous balanced**: Both stations send and receive over its part of a duplex line.

This mode is used for X.25 packet-switching networks.

The Link Access Procedure-Balanced (LAP-B) and Link Access Procedure D-channel (LAP-D) protocols are subsets of HDLC.

LAPB is a bit-oriented synchronous protocol that provides complete data transparency in a full-duplex point-to-point operation. It supports a peer-to-peer link in that neither end of the link plays the role of the permanent master station. HDLC NRM, on the other hand, has a permanent primary station with one or more secondary stations.

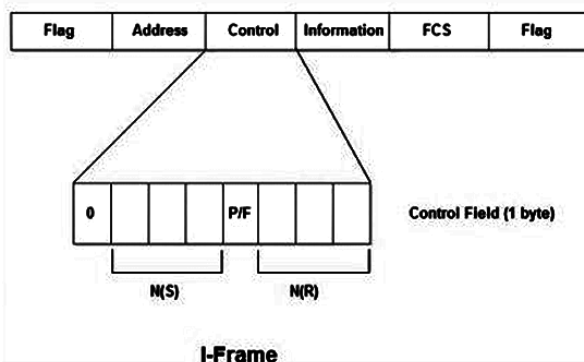
HDLCLAPB is a very efficient protocol, which requires a minimum of overhead to ensure flow control, error detection and recovery. If data is flowing in both directions (full duplex), the data frames themselves carry all the information required to ensure data integrity.

The concept of a frame window is used to send multiple frames before receiving confirmation that the first frame has been correctly received. This means that data can continue to flow in situations where there may be long "turn-around" time lags without stopping to wait for an acknowledgement. This kind of situation occurs, for instance in satellite communication.

Types

HDLCL defines three types of frames:

- Information frames (I-frame)
- Supervisory frame (S-frame)



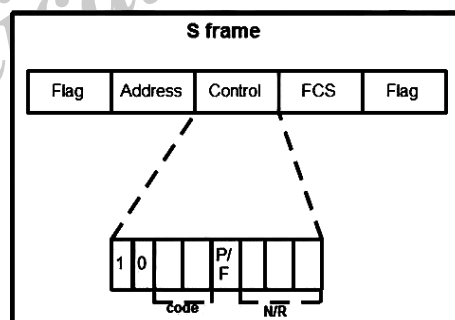
- Information frame (I frame)
 - The first bit of control field is always zero, i.e. the presence of zero at this place indicates that it is I-frame.
 - Bit number 2, 3 & 4 in control field is called N(S) that specifies the sequence number of the frame. Thus it specifies the number of the frame that is currently being sent. Since it is a 3-bit field, only eight sequence numbers are possible 0, 1, 2, 3, 4, 5, 6, 7 (000 to 111).
 - Bit number 5 in control field is P/F i.e. Poll/Final and is used for these two purposes. It has meaning only when it is set i.e. when P/F = 1. It can represent the following two cases.
 - It means poll when frame is sent by a primary station to secondary (when address field contains the address of receiver).

- It means final when frame is sent by secondary to a primary (when the address field contains the address of the sender).
- Bit number 6, 7, and 8 in control field specifies N(R) i.e. the sequence number of the frame expected in return in two-way communication.

If last frame received was error-free then N(R) number will be that of the next frame in sequence. If the last frame was not received correctly, the N(R) number will be the number of the damaged frame, asking for its retransmission.

2. Supervisory frame

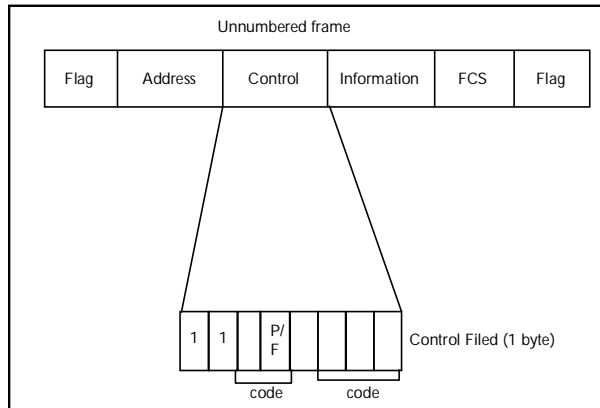
- S-frame carries control information, primarily data link layer flow and error controls.
- It does not contain information field.
- The format of S-frame is shown in diagram.



- The first two bits in the control field of S-frame are always 10.
- Then there is a bit code field that specifies four types of S-frame with combination 00, 01, 10, 11 as shown.

Table: Types of S-frame

Code	Command
00	RR Receive Ready
01	REJ Reject
10	RNR Receive Not Ready
11	SREJ Selective Reject



- U-frame contains two code fields, one two hit and other three bit.
- These five bits can create upto 32 different U-frames.
- P/F bit in control field has same purpose in V-frame as discussed earlier.
- Protocol Structure - HDLC: High Level Data Link Control
- **Flag** - The value of the flag is always (0x7E).
- **Address field** - Defines the address of the secondary station which is sending the frame or the destination of the frame sent by the primary station. It contains Service Access Point (6bits), a Command/Response bit to indicate whether the frame relates to information frames (I-frames) being sent from the node or received by the node, and an address extension bit which is usually set to true to indicate that the address is of length one byte. When set to false it indicates an additional byte follows.
- **Extended address** - HDLC provides another type of extension to the basic format. The address field may be extended to more than one byte by agreement between the involved parties.
- **Control field** - Serves to identify the type of the frame. In addition, it includes sequence numbers, control features and error tracking according to the frame type.

- **FCS** - The Frame Check Sequence (FCS) enables a high level of physical error control by allowing the integrity of the transmitted frame data to be checked.

2.5 MAC LAYER

Q9. Define MAC & state the functions of MAC?

Ans :

(Imp.)

Meaning

Media access control (MAC) and logical link control (LLC) are the sub layers of the data link layer (Layer 2) in OSI Reference Model. 'MAC' is also refer to as MAC layer. It use MAC protocols to provides unique addressing identification and channel access control mechanism for network nodes to communicate with other nodes across a shared channel.

MAC describes the process that is employed to control the basis on which devices can access the shared network. Some level of control is required to ensure the ability of all devices to access the network within a reasonable period of time, thereby resulting in acceptable access and response times.

It is also important that some method exists to either detect or avoid data collisions, which are caused by multiple transmissions being placed on the shared medium simultaneously. Media Access Control can be accomplished on either a centralized or decentralized basis, and can be characterized as either deterministic or non-deterministic in nature.

Functions

i) Centralized Control

A centralized controller polls devices to determine when access and transmission by each station is allowed to occur. Stations transmit when requested to do so, or when a station transmission request is acknowledged and granted. This process of polling requires the passing of control packets, adding overhead and reducing the amount of throughput relative to the raw bandwidth available. Additionally, the failure of the central controller will disrupt the entire network; in such an event, the controller is taken off-line and a backup controller assumes responsibility. Centrally, controlled networks generally employ deterministic access control; Token Ring and FDDI networks are centrally controlled.

ii) Deterministic Access

Deterministic access is a media access control convention that allows both the centralized master station and each slaved station to determine the maximum length of time which will pass before access is provided to the network. In other words, each station can be guaranteed the right to communicate within a certain time frame. Additionally, the system administrator can assign access priorities. Deterministic access is also known as non-contentious, because the devices do not contend for access, rather access is controlled on a centralized basis.

Deterministic access employs token passing. The token, which consists of a specific bit pattern, indicates the status of the network whether it is available or unavailable. The token is generated by a centralized master control station and transmitted across the network. The station in possession of the token is in control of access to the network. It may transmit or may require other stations to respond. After transmitting, the station will pass the token to a successor station in a predetermined sequence while the process is complex and overhead intensive, it yields careful control over the network.

Deterministic access is especially effective in high-traffic environments where a lack of control would cause chaos in the form of frequent data collisions.

General characteristics of token-based networks include a high level of access control, which is centralized. Access delay is measured and assured, with priority access being supported. Throughput is very close to raw bandwidth, as data collisions are avoided; throughput also improves under load, although absolute overhead is higher than is the case with non-deterministic access techniques. Deterministic access standards include Token-Passing Ring, IBM Token Ring, and Token-Passing Bus.

Token-based LAN technologies are somewhat overhead intensive, due to the token passing and management processes. However, they can more than compensate for that fact by virtue of the avoidance of data collisions. Token Ring, for instance comes in 4, 16 and 20 Mbps. In each case bandwidth utilization is virtually 100%.

iii) Non-deterministic Access

Non-deterministic media access control, places access control responsibilities on the individual stations. This is popularly known as Carrier Sense Multiple Access (CSMA), and is most effective in low-traffic environments. There are two variations, CSMA/CD and CSMA/CA.

CSMA is a decentralized, contentious media access control method used in Ethernet and other bus oriented LANs. Each of multiple stations, or nodes, must sense the carrier to determine network availability before access to the medium to transmit data: further, each station must monitor the network to determine if a collision has occurred. Collisions render the transmission invalid and require retransmission. In the event of a busy condition, the station will back off the network for a calculated random time interval before attempting subsequent access.

CSMA is implemented in two standard means, CSMA/CD and CSMA/CA. In either case, latency and throughput degrade under heavy loads of traffic. For example, an Ethernet network running at a theoretical speed of 10Mbps typically provides about 4 to 6 Mbps throughput. While it is less costly than Token Ring networking, it also delivers less efficient use of bandwidth.

Carrier Sense Multiple Access with Collision Detection (CSMA/CD). This is the most common media access control method used in bus networks. At that point, all devices back off the network, calculating a random time interval before attempting a retransmission.

Carrier Sense Multiple Access/Collision Avoid (CSMA/CA). This includes a priority scheme to guarantee the transmission privileges of high-priority stations. CSMA/CA requires a delay in network activity after each transmission is completed. That delay is proportionate to the priority level of each device, with high-priority nodes programmed for short delays and with low-priority nodes programmed for relatively long delays. As collisions may still occur, they are managed either through Collision Detect or through retransmission after receipt of a Negative Acknowledgment (NAK). CSMA/CA is more expensive to implement, as it requires that additional programmed logic be

embedded in each device or NIC. CSMA/CA does, however, offer the advantage of improved access control, which serves to reduce collisions and, thereby, improve the overall performance of the network.

2.5.1 The Channel Allocation Problem

Q10. Explain the channel Allocation problem?

Ans :

The traditional way of allocating a single channel, such as a telephone trunk, among multiple competing users is Frequency Division Multiplexing (FDM). If there are N users, the bandwidth is divided into N equal-sized portions, each user being assigned one portion. Since each user has a private frequency band, there is no interference between users. When there are only a small and constant number of users, each of which has a heavy (buffered) load of traffic (e.g., carriers' switching offices), FDM is a simple and efficient allocation mechanism.

However, when the number of senders is large and continuously varying or the traffic is busty FDM presents some problems. If the spectrum is cut up into N regions and fewer than N users are currently interested in communicating, a large piece of valuable spectrum will be wasted. If more than N users want to communicate, some of them will be denied permission for lack of bandwidth, even if some of the users who have been assigned a frequency band hardly ever transmit or receive anything.

However, even assuming that the number of users could somehow be held constant at N , dividing the single available channel into static sub-channels is inherently inefficient. The basic problem is that when some users are quiescent, their bandwidth is simply lost. They are not using it, and no one else is allowed to use it either. Furthermore, in most computer systems, data traffic is extremely busty (peak traffic to mean traffic ratios of 1000:1 are common). Consequently, most of the channels will be idle most of the time.

The poor performance of static FDM can easily be seen from a simple queuing theory calculation. Let us start with the mean time delay, T , for a channel of capacity C bps, with an arrival rate of 1 frames/sec, each frame having a length drawn from an exponential probability density function with mean $1/\mu$ bits/frame. With these parameters the arrival rate is 1 frames/sec and the service rate is μC frames/sec. From queuing theory it can be shown that for Poisson arrival and service times,

For example, if C is 100 Mbps, the mean frame length, $1/\mu$, is 10,000 bits, and the frame arrival rate, 1, is 5000 frames/sec, then $T = 200 \mu\text{sec}$. Note that if we ignored the queuing delay and just asked how long it takes to send a 10,000 bit frame on a 100-Mbps network, we would get the (incorrect) answer of $100 \mu\text{sec}$. That result only holds when there is no contention for the channel.

Now let us divide the single channel into N independent sub-channels, each with capacity C/N bps. The mean input rate on each of the sub-channels will now be $1/N$.

The mean delay using FDM is N times worse than if all the frames were somehow magically arranged orderly in a big central queue.

Precisely the same arguments that apply to FDM also apply to time division multiplexing (TDM). Each user is statically allocated every N^{th} time slot. If a user does not use the allocated slot, it just lies fallow. The same holds if we split up the networks physically. Using our previous example again, if we were to replace the 100-Mbps network with 10 networks of 10 Mbps each and statically allocate each user to one of them, the mean delay would jump from $200 \mu\text{sec}$ to 2 msec.

Since none of the traditional static channel allocation methods work well with busty traffic, we will now explore dynamic methods.

2.5.2 Dynamic Channel Allocation in LANs and MANs

Q11. Explain Dynamic channel allocation in LAN and MAN ?

Ans :

(Imp.)

Before we get into the first of the many channel allocation methods to be discussed in this chapter, it is worth while carefully formulating the allocation problem. Underlying all the work done in this area are five key assumptions, described below.

The model consists of N independent stations (e.g., computers, telephones, or personal communicators), each with a program or user that generates frames for transmission. Stations are sometimes called terminals. The probability of a frame being generated in an interval of length Δt is $\lambda \Delta t$, where λ is a constant (the arrival rate of new frames). Once a frame has been generated, the station is blocked and does nothing until the frame has been successfully transmitted.

i) Single Channel Assumption

A single channel is available for all communication. All stations can transmit on it and all can receive from it. As far as the hardware is concerned, all stations are equivalent, although protocol software may assign priorities to them.

The single channel assumption is the heart of the model. There are no external ways to communicate. Stations cannot raise their hands to request that the teacher call on them.

ii) Collision Assumption

If two frames are transmitted simultaneously, they overlap in time and the resulting signal is garbled. This event is called a collision. All stations can detect collisions. A collided frame must be transmitted again later. There are no errors other than those generated by collisions.

1. **Continuous Time:** Frame transmission can begin at any instant. There is no master clock dividing time into discrete intervals.
2. **Slotted Time:** Time is divided into discrete intervals (slots). Frame transmissions always begin at the start of a slot. A slot may contain 0, 1, or more frames, corresponding to an idle slot, a successful transmission, or a collision, respectively.

I) Carrier Sense

Stations can tell if the channel is in use before trying to use it. If the channel is sensed as busy, no station will attempt to use it until it goes idle.

II) No Carrier Sense

Stations cannot sense the channel before trying to use it. They just go ahead and transmit. Only later can they determine whether the transmission was successful.

The first one says that stations are independent and that work is generated at a constant rate. It also implicitly assumes that each station only has one program or user, so while the station is blocked, no new work is generated. More sophisticated models allow multi-programmed stations that can generate work while a station is blocked, but the analysis of these stations is much more complex.

The collision assumption is also basic, although in some systems (notably spread spectrum), this assumption is relaxed, with surprising results. Also, some LANs, such as token rings, pass a special token from station to station, possession of which allows the current holder to transmit a frame. But in the coming sections we will stick to the single channel with contention and collisions model.

Two alternative assumptions about time are possible. Either it is continuous (1) or it is slotted (2). Some systems use one and some systems use the other, so we will discuss and analyze both. For a given system, only one of them holds.

Similarly, a network can either have carrier sensing (I) or not have it (II). LANs generally have carrier sense. However, wireless networks cannot use it effectively because not every station may be within radio range of every other station. Stations on wired carrier sense networks can terminate their transmission prematurely if they discover that it is colliding with another transmission. Collision detection is rarely done on wireless networks, for engineering reasons. Note that the word "carrier" in this sense refers to an electrical signal on the cable and has nothing to do with the common carriers (e.g., telephone companies) that date back to the Pony Express days.

2.6 ALOHA

Q12. What is ALOHA? Explain different types of Aloha?

Ans :

(Imp.)

Meaning

ALOHA stands for Abramson's Logic of Hiring Access. Aloha, also called the Aloha method, refers to a simple communications scheme in which each source (transmitter) in a network sends data whenever there is a frame to send.

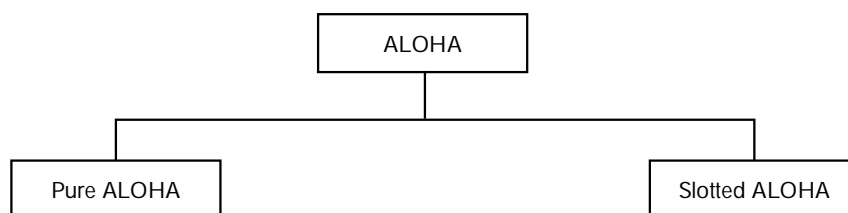
ALOHA is a system for coordinating and arbitrating access to a shared communication Networks channel. It was developed in the 1970s by Norman Abramson and his colleagues at the University of Hawaii. The original system used for ground based radio broadcasting, but the system has been implemented in satellite communication systems.

A shared communication system like ALOHA requires a method of handling collisions that occur when two or more systems attempt to transmit on the channel at the same time. In the ALOHA system, a node transmits whenever data is available to send. If another node transmits at the same time, a collision occurs, and the frames that were transmitted are lost. However, a node can listen to broadcasts on the medium, even its own, and determine whether the frames were transmitted.

Aloha is a multiple access protocol at the datalink layer and proposes how multiple terminals access the medium without interference or collision. In 1972 Roberts developed a protocol that would increase the capacity of aloha two fold. The Slotted Aloha protocol involves dividing the time interval into discrete slots and each slot interval corresponds to the time period of one frame. This method requires synchronization between the sending nodes to prevent collisions.

Types

There are two different versions of ALOHA



Types of ALOHA

i) Pure ALOHA

Pure ALOHA is introduced by Norman Abramson and his associates at the University of Hawaii in early 1970. The Pure ALOHA just allows every station to transmit the data whenever they have the data to be sent. When every station transmits the data without checking whether the channel is free or not there is always the possibility of the collision of data frames. If the acknowledgment arrived for the received frame, then it is ok or else if the two frames collide (Overlap), they are damaged.

If a frame is damaged, then the stations wait for a random amount of time and retransmit the frame till it transmits successfully. The waiting time of the each station must be random and it must not be same just to avoid the collision of the frames again and again. The throughput of the Pure ALOHA is maximized when the frames are of uniform length. The formula to calculate the throughput of the Pure ALOHA is $S = G * e^{-2G}$, the throughput is maximum when $G = 1/2$ which is 18% of the total transmitted data frames.

ii) Slotted ALOHA

After the pure ALOHA in 1970, Roberts introduced another method to improve the capacity of the Pure ALOHA which is called Slotted ALOHA. He proposed to divide the time into discrete intervals called time slots. Each time slot corresponds to the length of the frame. In contrast to the Pure ALOHA, Slotted ALOHA does not allow to transmit the data whenever the station has the data to be send. The Slotted ALOHA makes the station to wait till the next time slot begins and allow each data frame to be transmitted in the new time slot.

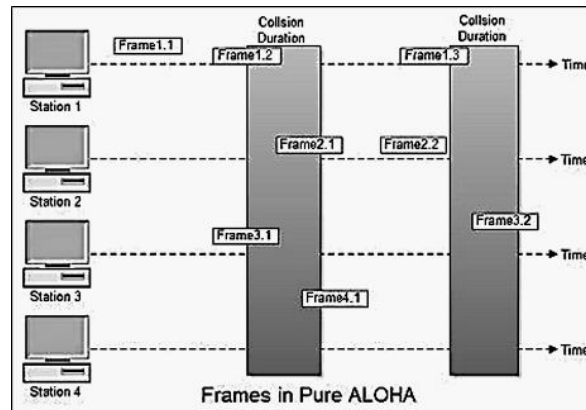
Synchronization can be achieved in Slotted ALOHA with the help of a special station that emits a pip at the beginning of every time slot as a clock does. The formula to calculate the throughput of the Slotted ALOHA is $S = G * e^{-G}$, the throughput is maximum when $G = 1$ which is 37% of the total transmitted data frames. In Slotted ALOHA, 37% of the time slot is empty, 37% successes and 26% collision.

2.6.1 Pure ALOHA**Q13. What is pure ALOHA?**

Ans :

- In pure ALOHA, the stations transmit frames whenever they have data to send.
- When two or more stations transmit simultaneously, there is collision and the frames are destroyed.
- In pure ALOHA, whenever any station transmits a frame, it expects the acknowledgment from the receiver.
- If acknowledgement is not received within specified time, the station assumes that the frame (or acknowledgement) has been destroyed.
- If the frame is destroyed because of collision the station waits for a random amount of time and sends it again. This waiting time must be random otherwise same frames will collide again and again.
- Therefore pure ALOHA dictates that when time-out period passes, each station must wait for a random amount of time before resending its frame. This randomness will help avoid more collisions.

Figure shows an example of frame collisions in pure ALOHA.



- In fig there are four stations that contend with one another for access to shared channel. All these stations are transmitting frames. Some of these frames collide because multiple frames are in contention for the shared channel. Only two frames, frame 1.1 and frame 2.2 survive. All other frames are destroyed.
- Whenever two frames try to occupy the channel at the same time, there will be a collision and both will be damaged. If first bit of a new frame overlaps with just the last bit of a frame almost finished, both frames will be totally destroyed and both will have to be retransmitted.

Operations

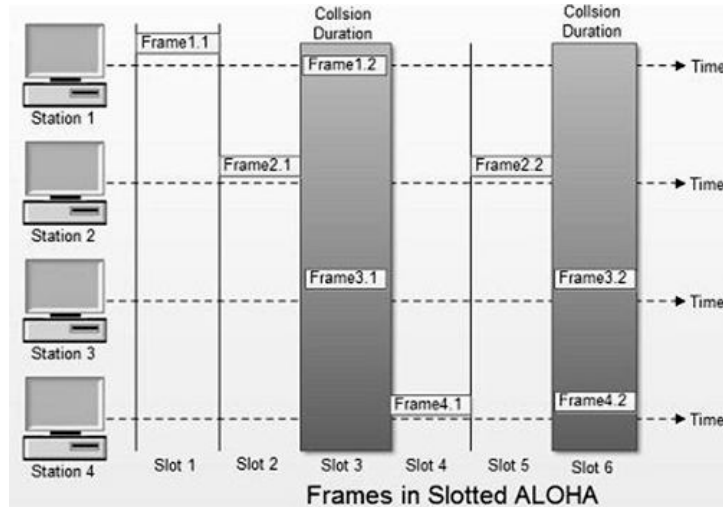
1. The frames are transmitted on the shared channel by all the nodes that wish to send.
2. An acknowledgement is awaited from the receiving nodes, confirming the successful delivery of the frames.
3. If collision appears, the nodes come to know about it after a time-out period when no acknowledgement is received.
4. After the passing of time-out period, the nodes should wait for random amount of time (back-off time) before transmitting again. This random delay of time is likely to differ thus making an attempt to avoid collision.
5. If after retransmission, again a collision occurs then above step is repeated.
6. After a predetermined maximum number of retransmissions, the station is required to give up and try later.

2.6.2 Slotted ALOHA

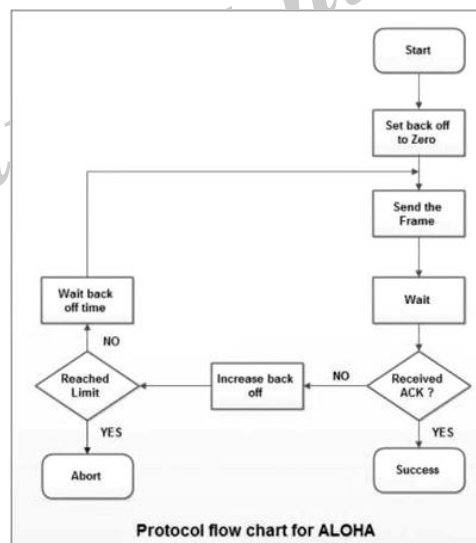
Q14. Explain the concept of Slotted Aloha.

Ans :

- Slotted ALOHA was invented to improve the efficiency of pure ALOHA as chances of collision in pure ALOHA are very high.
- In slotted ALOHA, the time of the shared channel is divided into discrete intervals called slots.
- The stations can send a frame only at the beginning of the slot and only one frame is sent in each slot.

**Frame 1.1**

- In slotted ALOHA, if any station is not able to place the frame onto the channel at the beginning of the slot *i.e.* it misses the time slot then the station has to wait until the beginning of the next time slot.
- In slotted ALOHA, there is still a possibility of collision if two stations try to send at the beginning of the same time slot as shown in fig.
- Slotted ALOHA still has an edge over pure ALOHA as chances of collision are reduced to one-half.

Protocol Flow Chart for ALOHA**Frame 2.2****Operations**

1. When a node has a frame to send, it waits for the beginning of the next slot and then transmit the frame in that slot.
2. If a collision occurs, it is sensed before the end of the slot. Then the node waits for the next subsequent slot to begin and then retransmits.
3. If again a collision is encountered, above step is repeated. If not, then it is the sign of successful transmission.

2.6.3 Pure ALOHA Vs Slotted ALOHA

Q15. What are the differences between Pure ALOHA and Slotted ALOHA.

Ans :

(Imp.)

BASIS FOR COMPARISON	PURE ALOHA	SLOTTED ALOHA
Introduced	Introduced by Norman Abramson and his associates at the University of Hawaii in 1970.	Introduced by Roberts in 1972.
Frame Transmission	The user can transmit the data frame whenever the station has the data to be transmitted.	The user has to wait till the next time slot start, to transmit the data frame.
Time	In Pure ALOHA the time is continuous.	In Slotted ALOHA the time is discrete.
Successful Transmission	The probability of successful transmission of the data frame is: $S = G * e^{-2G}$	The probability of successful transmission of the data frame is: $S = G * e^{-G}$
Synchronization	The time is not globally synchronized.	The time here is globally synchronized.
Throughput	The maximum throughput occurs at $G = 1/2$ which is 18%.	The maximum throughput occurs at $G = 1$ which is 37%.

2.7 ETHERNET IEEE 802.3 LAN ETHERNET EFFICIENCY CALCULATION

Q16. Explain about Ethernet work routing?

Ans :

(Imp.)

Ethernet is the most popular physical layer LAN technology in use today. It defines the number of conductors that are required for a connection, the performance thresholds that can be expected, and provides the framework for data transmission. A standard Ethernet network can transmit data at a rate up to 10 Megabits per second (10 Mbps). Other LAN types include Token Ring, Fast Ethernet, Gigabit Ethernet, 10 Gigabit Ethernet, Fiber Distributed Data Interface (FDDI), Asynchronous Transfer Mode (ATM) and LocalTalk.

Ethernet is popular because it strikes a good balance between speed, cost and ease of installation. These benefits, combined with wide acceptance in the computer marketplace and the ability to support virtually all popular network protocols, make Ethernet an ideal networking technology for most computer users today.

The Institute for Electrical and Electronic Engineers developed an Ethernet standard known as IEEE Standard 802.3. This standard defines rules for configuring an Ethernet network and also specifies how the elements in an Ethernet network interact with one another. By adhering to the IEEE standard, network equipment and network protocols can communicate efficiently.

Fast Ethernet

The Fast Ethernet standard (IEEE 802.3u) has been established for Ethernet networks that need higher transmission speeds. This standard raises the Ethernet speed limit from 10 Mbps to 100 Mbps with only minimal changes to the existing cable structure. Fast Ethernet provides faster throughput for video, multimedia, graphics, Internet surfing and stronger error detection and correction.

There are three types of Fast Ethernet: 100BASE-TX for use with level 5 UTP cable; 100BASE-FX for use with fiber-optic cable; and 100BASE-T4 which utilizes an extra two wires for use with level 3 UTP cable. The 100BASE-TX standard has become the most popular due to its close compatibility with the 10BASE-T Ethernet standard.

Network managers who want to incorporate Fast Ethernet into an existing configuration are required to make many decisions. The number of users in each site on the network that need the higher throughput must be determined; which segments of the backbone need to be reconfigured specifically for 100BASE-T; plus what hardware is necessary in order to connect the 100BASE-T segments with existing 10BASE-T segments. Gigabit Ethernet is a future technology that promises a migration path beyond Fast Ethernet so the next generation of networks will support even higher data transfer speeds.

Gigabit Ethernet

Gigabit Ethernet was developed to meet the need for faster communication networks with applications such as multimedia and Voice over IP (VoIP). Also known as "gigabit-Ethernet-over-copper" or 1000Base-T, GigE is a version of Ethernet that runs at speeds 10 times faster than 100Base-T. It is defined in the IEEE 802.3 standard and is currently used as an enterprise backbone. Existing Ethernet LANs with 10 and 100 Mbps cards can feed into a Gigabit Ethernet backbone to interconnect high performance switches, routers and servers.

From the data link layer of the OSI model upward, the look and implementation of Gigabit Ethernet is identical to that of Ethernet. The most important differences between Gigabit Ethernet and Fast Ethernet include the additional support of full duplex operation in the MAC layer and the data rates.

10 Gigabit Ethernet

10 Gigabit Ethernet is the fastest and most recent of the Ethernet standards. IEEE 802.3ae defines a version of Ethernet with a nominal rate of 10Gbps/s that makes it 10 times faster than

Gigabit Ethernet.

Unlike other Ethernet systems, 10 Gigabit Ethernet is based entirely on the use of optical fiber connections. This developing standard is moving away from a LAN design that broadcasts to all nodes, toward a system which includes some elements of wide area routing. As it is still very new, which of the standards will gain commercial acceptance has yet to be determined.

Asynchronous Transfer Mode (ATM)

ATM is a cell-based fast-packet communication technique that can support data-transfer rates from sub-T1 speeds to 10 Gbps. ATM achieves its high speeds in part by transmitting data in fixed-size cells and dispensing with error-correction protocols. It relies on the inherent integrity of digital lines to ensure data integrity.

ATM can be integrated into an existing network as needed without having to update the entire network. Its fixed-length cell-relay operation is the signaling technology of the future and offers more predictable performance than variable length frames. Networks are extremely versatile and an ATM network can connect points in a building, or across the country, and still be treated as a single network.

Power over Ethernet (PoE)

PoE is a solution in which an electrical current is run to networking hardware over the Ethernet Category 5 cable or higher. This solution does not require an extra AC power cord at the product location. This minimizes the amount of cable needed as well as eliminates the difficulties and cost of installing extra outlets.

LAN Technology Specifications

Name	IEEE Standard	Data Rate	Media Type	Maximum Distance
Ethernet	802.3	10 Mbps	10Base-T	100 meters
Fast Ethernet/ 100Base-T	802.3u	100 Mbps	100Base-TX 100Base-FX	100 meters 2000 meters
Gigabit Ethernet/ GigE	802.3z	1000 Mbps	1000Base-T 1000Base-SX 1000Base-LX	100 meters 275/550 meters 550/5000 meters
10 Gigabit Ethernet	IEEE 802.3ae	10 Gbps	10GBase-SR 10GBase-LX4 10GBase-LR/ER 10GBase-SW/LW/EW	300 meters 300m MMF/ 10km SMF 10km/40km 300m/10km/40km

Token Ring

Token Ring is another form of network configuration. It differs from Ethernet in that all messages are transferred in one direction along the ring at all times. Token Ring networks sequentially pass a “token” to each connected device. When the token arrives at a particular computer (or device), the recipient is allowed to transmit data onto the network. Since only one device may be transmitting at any given time, no data collisions occur. Access to the network is guaranteed, and time-sensitive applications can be supported. However, these benefits come at a price. Component costs are usually higher, and the networks themselves are considered to be more complex and difficult to implement. Various PC vendors have been proponents of Token Ring networks.

Networking and Ethernet Basics**Protocols**

After a physical connection has been established, network protocols define the standards that allow computers to communicate. A protocol establishes the rules and encoding specifications for sending data. This defines how computers identify one another on a network, the form that the data should take in transit, and how this information is processed once it reaches its final destination. Protocols also define procedures for determining the type of error checking that will be used, the data compression method, if one is needed, how the sending device will indicate that it has finished sending a message, how the receiving device will indicate that it has received a message, and the handling of lost or damaged transmissions or “packets”.

The main types of network protocols in use today are: TCP/IP (for UNIX, Windows NT, Windows 95 and other platforms); IPX (for Novell NetWare); DECnet (for networking Digital Equipment Corp. computers); AppleTalk (for Macintosh computers), and NetBIOS/NetBEUI (for LAN Manager and Windows NT networks).

Although each network protocol is different, they all share the same physical cabling. This common method of accessing the physical network allows multiple protocols to peacefully coexist over the network media, and allows the builder of a network to use common hardware for a variety of protocols. This concept is known as “protocol independence,” which means that devices which are compatible at the physical and data link layers allow the user to run many different protocols over the same medium.

2.8 BRIDGES

Q17. Explain the concept of Bridges.

Ans :

(Imp.)

Many organizations have multiple LANs and wish to connect them. LANs can be connected by devices called bridges, which operate in the data link layer. This statement means that bridges do not examine the network layer header and can thus copy IP, IPX, and OSI packets equally well. In contrast, a pure IP, IPX, or OSI router can handle only its own native packets.

In the following sections we will look at bridge design, especially for connect-ing 802.3, 802.4, and 802.5 LANs. For a comprehensive treatment of bridges and related topics, see (Perlman, 1992). Before getting into the technology of bridges, it is worthwhile taking a look at some common situations in which bridges are used. We will mention six reasons why a single organization may end up with multiple LANs. First, many university and corporate departments have their own LANs, primarily to connect their own personal computers, workstations, and servers. Since the goals of the various departments differ, different departments choose different LANs, without regard to what other departments are doing. Sooner or later, there is a need for interaction, so bridges are needed. In this example, multiple LANs came into existence due to the autonomy of their owners.

Second, the organization may be geographically spread over several buildings separated by considerable distances. It may be cheaper to have separate LANs in each building and connect them with bridges and infrared links than to run a single coaxial cable over the entire site.

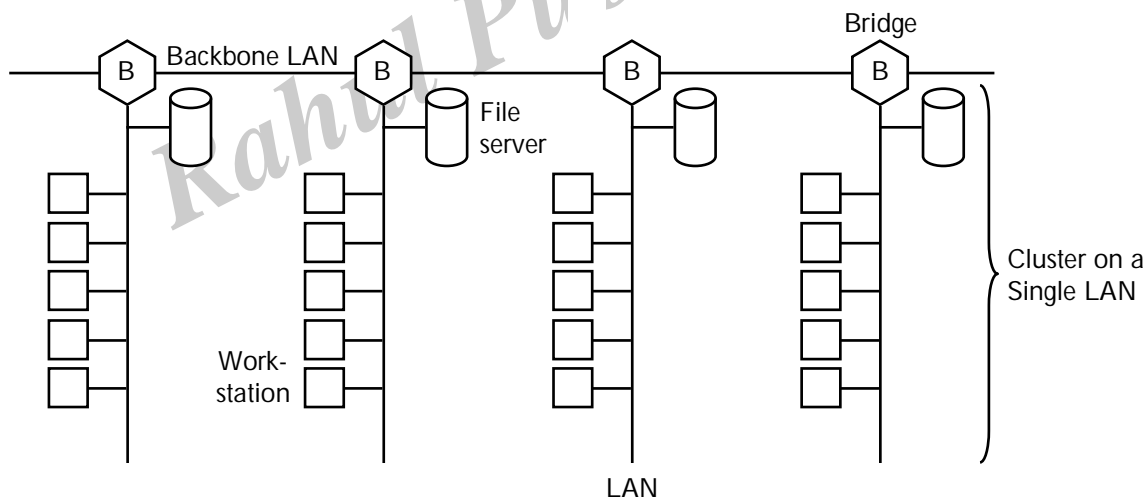


Fig.: Multiple LANs connected by a backbone to handle a total load higher than the capacity of a single LAN

Third, it may be necessary to split what is logically a single LAN into separate LANs to accommodate the load. At many universities, for example, thousands of workstations are available for student and faculty computing. Files are normally kept on file server machines, and are downloaded to users' machines upon request. The enormous scale of this system precludes putting all the work-stations on a single LAN—the total bandwidth needed is far too high. Instead multiple LANs connected by bridges are used, as shown in Figure Each LAN contains a cluster of workstations with its own file server, so that most traffic is restricted to a single LAN and does not add load to the backbone.

Fourth, in some situations, a single LAN would be adequate in terms of the load, but the physical distance between the most distant machines is too great (e.g., more than 2.5 km for 802.3). Even if laying the cable is easy to do, the network would not work due to the excessively long round-trip delay. The only solution is to partition the LAN and install bridges between the segments. Using bridges, the total physical distance covered can be increased.

Fifth, there is the matter of reliability. On a single LAN, a defective node that keeps outputting a continuous stream of garbage will cripple the LAN. Bridges can be inserted at critical places, like fire doors in a building, to prevent a single node which has gone berserk from bringing down the entire system. Unlike a repeater, which just copies whatever it sees, a bridge can be programmed to exercise some discretion about what it forwards and what it does not forward.

Sixth, and last, bridges can contribute to the organization's security. Most LAN interfaces have a promiscuous mode, in which all frames are given to the computer, not just those addressed to it. Spies and busybodies love this feature. By inserting bridges at various places and being careful not to forward sensitive traffic, it is possible to isolate parts of the network so that its traffic cannot escape and fall into the wrong hands.

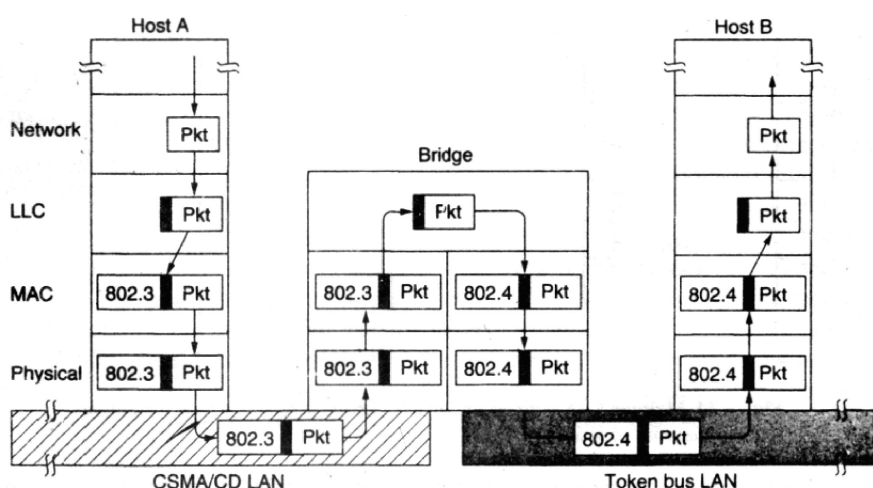


Fig.: Operation of a LAN bridge from 802.3 to 802.4

Having seen why bridges are needed, let us now turn to the question of how they work. Figure 35 illustrates the operation of a simple two-port bridge. Host A has a packet to send. The packet descends into the LLC sublayer and acquires an LLC header. Then it passes into the MAC sublayer and an 802.3 header is prepended to it (also a trailer, not shown in the figure). This unit goes out onto the cable and eventually is passed up to the MAC sublayer in the bridge, where the 802.3 header is stripped off. The bare packet (with LLC header) is then handed off to the LLC sublayer in the bridge. In this example, the packet is destined for an 802.4 subnet connected to the bridge, so it works its way down the 802.4 side of the bridge and off it goes. Note that a bridge connecting k different LANs will have k different MAC sublayers and k different physical layers, one for each type.

Bridges from 802.x to 802.y

Naively think that a bridge from one 802 LAN to another one would be completely trivial. Such is not the case. In the remainder of this section we will point out some of the difficulties that will be encountered when trying to build a bridge between the various 802 LANs.

Each of the nine combinations of 802.x to 802.y has its own unique set of problems. However, before dealing with these one at a time, let us look at some general problems common to all the bridges. To start with, each of the LANs uses a different frame format (see Figure). There is no valid technical

reason for this incompatibility. It is just that none of the corporations supporting the three standards (Xerox, GM, and IBM) wanted to change *theirs*. As a result, any copy-ing between different LANs requires reformatting, which takes CPU time, requires a new checksum calculation, and introduces the possibility of undetected errors due to bad bits in the bridge's memory. None of this would have been necessary if the three committees had been able to agree on a single format.

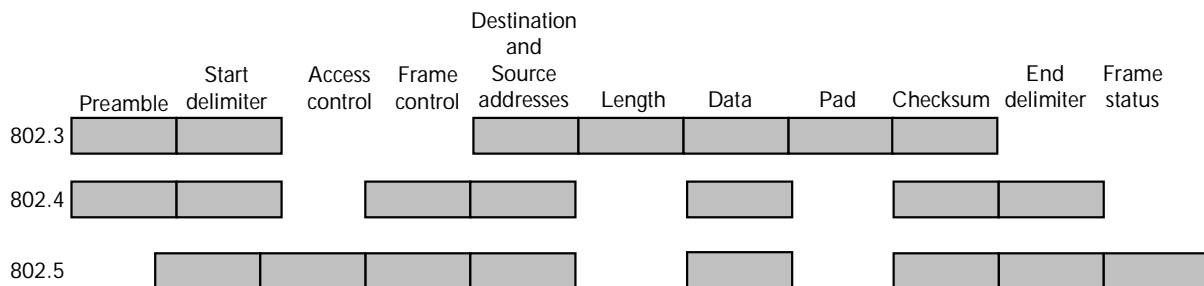


Fig.: The IEEE 802 frame formats

A second problem is that interconnected LANs do not necessarily run at the same data rate. When forwarding a long run of back-to-back frames from a fast LAN to a slower one, the bridge will not be able to get rid of the frames as fast as they come in. It will have to buffer them, hoping not to run out of memory. The problem also exists from 802.4 to 802.3 at 10 Mbps to some extent because some of 802.3's bandwidth is lost to collisions. It does not really have 10 Mbps, whereas 802.4 really does (well, almost). Bridges that connect three or more LANs have a similar problem when several LANs are trying to feed the same out-put LAN at the same time.

A subtle, but important problem related to the bridge-as-bottleneck problem is the value of timers in the higher layers. Suppose that the network layer on an 802.4 LAN is trying to send a very long message as a sequence of frames. After sending the last one it starts a timer to wait for an acknowledgement. If the mes-sage has to transit a bridge to a slower 802.5 LAN, there is a danger that the timer will go off before the last frame has been forwarded onto the slower LAN. The network layer will assume the problem is due to a lost frame and just retransmit the entire sequence again. After n failed attempts it may give up and tell the transport layer that the destination is dead.

A third, and potentially most serious problem of all, is that all three 802 LANs have a different maximum frame length. For 802.3 it depends on the parameters of the configuration, but for the standard 10-Mbps system the payload is a max-imum of 1500 bytes. For 802.4 it is fixed at 8191 bytes. For 802.5 there is no upper limit, except that a station may not transmit longer than the token-holding time. With the default value of 10 msec, the maximum frame length is 5000 bytes.

An obvious problem arises when a long frame must be forwarded onto a LAN that cannot accept it. Splitting the frame into pieces is out of the question in this layer. All the protocols assume that frames either arrive or they do not. There is no provision for reassembling frames out of smaller units. This is not to say that such protocols could not be devised. They could be and have been. It is just that 802 does not provide this feature. Basically, there is no solution. Frames that are too large to be forwarded must be discarded. So much for transparency.

Now let us briefly consider each of the nine cases of 802.x to 802.y bridges to see what other problems are lurking in the shadows. From 802.3 to 802.3 is easy. The only thing that can go wrong is that the destination LAN is so heavily loaded that frames keep pouring into the bridge, but the bridge cannot get rid of them. If this situation persists long enough, the bridge might run out of buffer space and begin dropping frames. Since this problem is always potentially present when forwarding onto an 802.3 LAN, we will not mention it further. With the other two LANs, each station, including the bridge is guaranteed to acquire the token periodically and cannot be shut out for long intervals.

From 802.4 to 802.3 two problems exist. First, 802.4 frames carry priority bits that 802.3 frames do not have. As a result, if two 802.4 LANs communicate via an 802.3 LAN, the priority will be lost by the intermediate LAN.

The second problem is caused by a specific feature in 802.4: temporary token handoff. It is possible for an 802.4 frame to have a header bit set to 1 to temporarily pass the token to the destination, to let it send an acknowledgement frame. However, if such a frame is forwarded by a bridge, what should the bridge do? If it sends an acknowledgement frame itself, it is lying because the frame really has not been delivered yet. In fact, the destination may be dead.

On the other hand, if it does not generate the acknowledgement, the sender will almost assuredly conclude that the destination is dead and report back failure to its superiors. There does not seem to be any way to solve this problem.

From 802.5 to 802.3 we have a similar problem. The 802.5 frame format has A and C bits in the frame status byte. These bits are set by the destination to tell the sender whether the station addressed saw the frame, and whether it copied it. Here again, the bridge can lie and say the frame has been copied, but if it later turns out that the destination is down, serious problems may arise. In essence, the insertion of a bridge into the network has changed the semantics of the bits. It is hard to imagine a proper solution to this problem.

From 802.3 to 802.4 we have the problem of what to put in the priority bits. A good case can be made for having the bridge retransmit all frames at the highest priority, because they have probably suffered enough delay already.

From 802.4 to 802.4 the only problem is what to do with the temporary token handoff. At least here we have the possibility of the bridge managing to forward the frame fast enough to get the response before the timer runs out. Still it is a gamble. By forwarding the frame at the highest priority, the bridge is telling a little white lie, but it thereby increases the probability of getting the response in time.

From 802.5 to 802.4 we have the same problem with the A and C bits as before. Also, the definition of the priority bits is different for the two LANs, but beggars can't be choosers. At least the two LANs have the same number of priority bits. All the bridge can do is copy the priority bits across and hope for the best.

From 802.3 to 802.5 the bridge must generate priority bits, but there are no other special problems. From 802.4 to 802.5 there is a potential problem with frames that are too long and the token handoff problem is present again. Finally, from 802.5 to 802.5 the problem is what to do with the A and C bits again. Figure 4-37 summarizes the various problems we have been discussing.

When the IEEE 802 committee set out to come up with a LAN standard, it was unable to agree on a single standard, so it produced *three* incompatible standards, as we have just seen in some detail. For this failure, it has been roundly criticized. When it was later assigned the job of designing a standard for bridges to interconnect its three incompatible LANs, it resolved to do better. It did. It came up with *two* incompatible bridge designs. So far nobody has asked it to design a gateway standard to connect its two incompatible bridges, but at least the trend is in the right direction.

This section has dealt with the problems encountered in connecting two IEEE 802 LANs via a single bridge. The next two sections deal with the problems of connecting large internetworks containing many LANs and many bridges and the two IEEE approaches to designing these bridges!

Q18. Explain the concept of Transparent Bridges.

Ans :

The first 802 bridge is a transparent bridge or spanning tree bridge (Perlman, 1992). The overriding concern of the people who supported this design was complete transparency. In their view, a site with

multiple LANs should be able to go out and buy bridges designed to the IEEE standard, plug the connectors into the bridges, and everything should work perfectly, instantly. There should be no hardware changes required, no software changes required, no setting of address switches, no downloading of routing tables or parameters, nothing. Just plug in the cables and walk away. Furthermore, the operation of the existing LANs should not be affected by the bridges at all. Surprisingly enough, they actually succeeded.

A transparent bridge operates in promiscuous mode, accepting every frame transmitted on all the LANs to which it is attached. As an example, consider the configuration of Figure. Bridge B1 is connected to LANs 1 and 2, and bridge B2 is connected to LANs 2, 3, and 4. A frame arriving at bridge B1 on LAN 1 destined for A can be discarded immediately, because it is already on the right LAN, but a frame arriving on LAN 1 for C or F must be forwarded.

When a frame arrives, a bridge must decide whether to discard or forward it, and if the latter, on which LAN to put the frame.

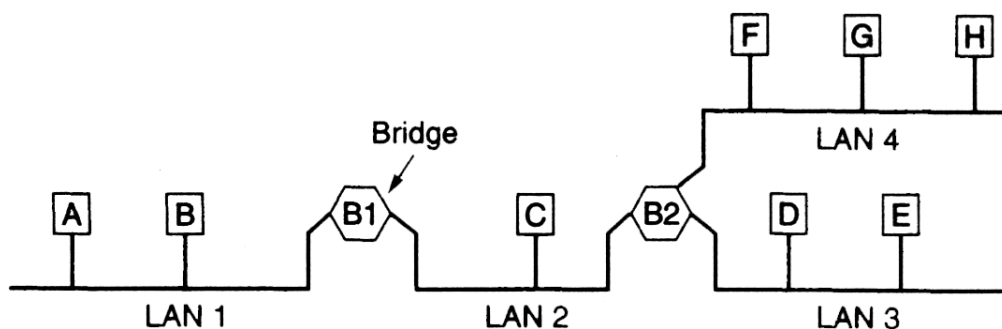


Fig.: A configuration with four LANs and two bridges

This decision is made by looking up the destination address in a big (hash) table inside the bridge. The table can list each possible destination and tell which output line (LAN) it belongs on. For example, B2's table would list A as belonging to LAN 2, since all B2 has to know is which LAN to put frames for A on. That, in fact, more forwarding happens later is not of interest to it.

When the bridges are first plugged in, all the hash tables are empty. None of the bridges know where any of the destinations are, so they use the flooding algorithm: every incoming frame for an unknown destination is output on all the LANs to which the bridge is connected except the one it arrived on. As time goes on, the bridges learn where destinations are, as described below. Once a destination is known, frames destined for it are put on only the proper LAN and are not flooded.

The algorithm used by the transparent bridges is backward learning. As mentioned above, the bridges operate in promiscuous mode, so they see every frame sent on any of their LANs. By looking at the source address, they can tell which machine is accessible on which LAN. For example, if bridge B1 in Figure sees a frame on LAN 2 coming from C, it knows that C must be reachable via LAN 2, so it makes an entry in its hash table noting that frames going to C should use LAN 2. Any subsequent frame addressed to C coming in on LAN 1 will be forwarded, but a frame for C coming in on LAN 2 will be discarded.

The topology can change as machines and bridges are powered up and down and moved around. To handle dynamic topologies, whenever a hash table entry is made, the arrival time of the frame is noted in the entry. Whenever a frame whose destination is already in the table arrives, its entry is updated with the current time. Thus the time associated with every entry tells the last time a frame from that machine was seen.

Periodically, a process in the bridge scans the hash table and purges all entries more than a few minutes old. In this way, if a computer is unplugged from its LAN, moved around the building, and

replugged in somewhere else, within a few minutes it will be back in normal operation, without any manual intervention. This algorithm also means that if a machine is quiet for a few minutes, any traffic sent to it will have to be flooded, until it next sends a frame itself.

The routing procedure for an incoming frame depends on the LAN it arrives on (the source LAN) and the LAN its destination is on (the destination LAN), as follows:

1. If destination and source LANs are the same, discard the frame.
2. If the destination and source LANs are different, forward the frame.
3. If the destination LAN is unknown, use flooding.

As each frame arrives, this algorithm must be applied. Special purpose VLSI chips exist to do the lookup and update the table entry, all in a few microseconds.

To increase reliability, some sites use two or more bridges in parallel between pairs of LANs, as shown in Figure. This arrangement, however, also introduces some additional problems because it creates loops in the topology.

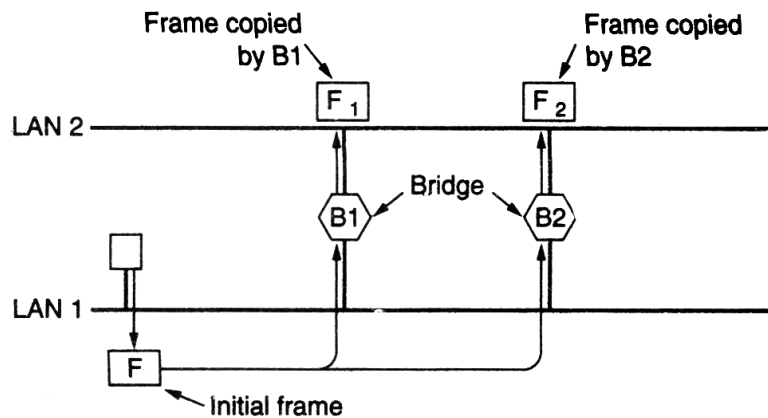


Fig.: Two parallel transparent bridges

A simple example of these problems can be seen by observing how a frame, F , with unknown destination is handled in Figure. Each bridge, following the normal rules for handling unknown destinations, uses flooding, which in this example, just means copying it to LAN 2. Shortly thereafter, bridge 1 sees F_2 , a frame with an unknown destination, which it copies to LAN 1, generating F_3 (not shown). Similarly, bridge 2 copies F_1 to LAN 1 generating F_4 (also not shown). Bridge 1 now forwards F_4 and bridge 2 copies F_3 . This cycle goes on forever.

Q19. Explain the concept of spanning tree Bridges.

Ans :

(Imp.)

The solution to this difficulty is for the bridges to communicate with each other and overlay the actual topology with a spanning tree that reaches every LAN. In effect, some potential connections between LANs are ignored in the interest of constructing a fictitious loop-free topology. For example, in Figure (a) we see nine LANs interconnected by ten bridges. This configuration can be abstracted into a graph with the LANs as the nodes. An arc connects any two LANs that are connected by a bridge. The graph can be reduced to a spanning tree by dropping the arcs shown as dotted lines in Figure (b). Using this spanning tree, there is exactly one path from every LAN to every other LAN. Once the bridges have agreed on the spanning tree, all forwarding between LANs follows the spanning tree. Since there is a unique path from each source to each destination, loops are impossible.

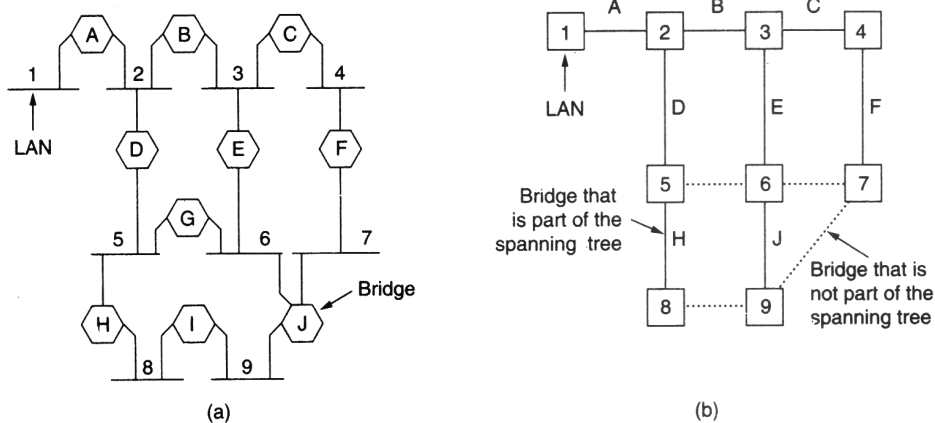


Fig.: (a) Interconnected LANs, (b) A spanning tree covering the LANs. The dotted lines are not part of the spanning tree.

To build the spanning tree, first the bridges have to choose one bridge to be the root of the tree. They make this choice by having each one broadcast its serial number, installed by the manufacturer, and guaranteed to be unique worldwide. The bridge with the lowest serial number becomes the root. Next, a tree of short-test paths from the root to every bridge and LAN is constructed. This tree is the spanning tree. If a bridge or LAN fails, a new one is computed.

The result of this algorithm is that a unique path is established from every LAN to the root, and thus to every other LAN. Although the tree spans all the LANs, not all the bridges are necessarily present in the tree (to prevent loops). Even after the spanning tree has been established, the algorithm continues to run in order to automatically detect topology changes and update the tree. The distributed algorithm used for constructing the spanning tree was invented by Perlman and is described in detail in (Perlman, 1992).

Bridges can also be used to connect LANs that are widely separated. In this model, each site consists of a collection of LANs and bridges, one of which has a connection to a WAN. Frames for remote LANs travel over the WAN. The basic spanning tree algorithm can be used, preferably with certain optimizations to select a tree that minimizes the amount of WAN traffic.

Q20. Explain the concept of source routing Bridges.

Ans :

Transparent bridges have the advantage of being easy to install. You just plug them in and walk away. On the other hand, they do not make optimal use of the bandwidth, since they only use a subset of the topology (the spanning tree). The relative importance of these two (and other) factors led to a split within the 802 committees (Pitt, 1988). The CSMA/CD and token bus people chose the transparent bridge. The ring people (with encouragement from IBM) preferred a scheme called source routing, which we will now describe. For additional details, see (Dixon, 1987).

Reduced to its barest essentials, source routing assumes that the sender of each frame knows whether or not the destination is on its own LAN. When sending a frame to a different LAN, the source machine sets the high-order bit of the source address to 1, to mark it. Furthermore, it includes in the frame header the exact path that the frame will follow.

This path is constructed as follows. Each LAN has a unique 12-bit number, and each bridge has a 4-bit number that uniquely identifies it in the context of its LANs. Thus, two bridges far apart may both have number 3, but two bridges between the same two LANs must have different bridge numbers. A route is then a sequence of bridge, LAN, bridge, LAN, ... numbers. Referring to Fig. 4-38, the route from A to D would be (L1, B1, L2, B2, L3).

A source routing bridge is only interested in those frames with the high-order bit of the destination set to 1. For each such frame that it sees, it scans the route looking for the number of the LAN on which the frame arrived. If this LAN number is followed by its own bridge number, the bridge forwards the frame onto the LAN whose number follows its bridge number in the route. If the incoming LAN number is followed by the number of some other bridge, it does not forward the frame.

This algorithm lends itself to three possible implementations:

1. **Software:** the bridge runs in promiscuous mode, copying all frames to its memory to see if they have the high-order destination bit set to 1. If so, the frame is inspected further; otherwise it is not.
2. **Hybrid:** the bridge's LAN interface inspects the high-order destination bit and only accepts frames with the bit set. This interface is easy to build into hardware and greatly reduces the number of frames the bridge must inspect.
3. **Hardware:** the bridge's LAN interface not only checks the high-order destination bit, but it also scans the route to see if this bridge must do forwarding. Only frames that must actually be forwarded are given to the bridge. This implementation requires the most complex hardware but wastes no bridge CPU cycles because all irrelevant frames are screened out.

These three implementations vary in their cost and performance. The first one has no additional hardware cost for the interface but may require a very fast CPU to handle all the frames. The last one requires a special VLSI chip but offloads much of the processing from the bridge to the chip, so that a slower CPU can be used, or alternatively, the bridge can handle more LANs.

Implicit in the design of source routing is that every machine in the internet-work knows, or can find, the best path to every other machine. How these routes are discovered is an important part of the source routing algorithm. The basic idea is that if a destination is unknown, the source issues a broadcast frame asking where it is. This discovery frame is forwarded by every bridge so that it reaches every LAN on the internet work. When the reply comes back, the bridges record their identity in it, so that the original sender can see the exact route taken and ultimately choose the best route.

While this algorithm clearly finds the best route (it finds all routes), it suffers from a frame explosion. Consider the configuration of Figure, with N LANs linearly connected by triple bridges. Each discovery frame sent by station 1 is copied by each of the three bridges on LAN 1, yielding three discovery frames on LAN 2. Each of these is copied by each of the bridges on LAN 2, resulting in nine frames on LAN 3. By the time we reach LAN N , 3^{N-1} frames are circulating. If a dozen sets of bridges are traversed, more than half a million discovery frames will have to be injected into the last LAN, causing severe congestion.

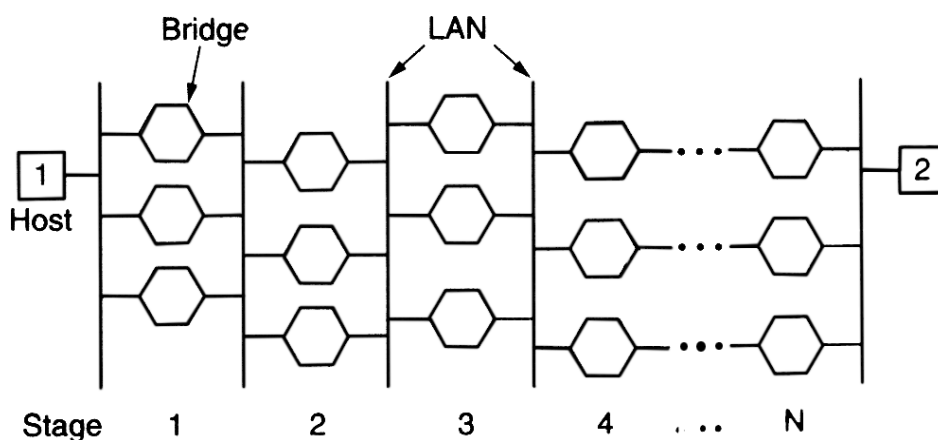


Fig.: A series of LANs connected by triple bridges

A somewhat analogous process happens with the transparent bridge, only it is not nearly so severe. When an unknown frame arrives, it is flooded, but only along the spanning tree, so the total volume of frames sent is linear with the size of the network, not exponential.

Once a host has discovered a route to a certain destination, it stores the route in a cache, so that the discovery process will not have to be run next time. While this approach greatly limits the impact of the frame explosion, it does put some administrative burden on all the hosts, and the whole algorithm is definitely not transparent, which was one of the original goals, as we mentioned above.

Q21. Compare and contrast transparent and Source routing Bridges.

Ans :

S.No.	Issue	Transparent bridge	Source routing bridge
1.	Orientation	Connectionless	Connection-oriented
2.	Transparency	Fully transparent	Not transparent
3.	Configuration	Automatic	Manual
4.	Routing	Suboptimal	Optimal
5.	Locating	Backward learning	Discovery frames
6.	Failures	Handled by the bridges	Handled by the hosts
7.	Complexity	In the bridges	In the hosts

2.9 ARP

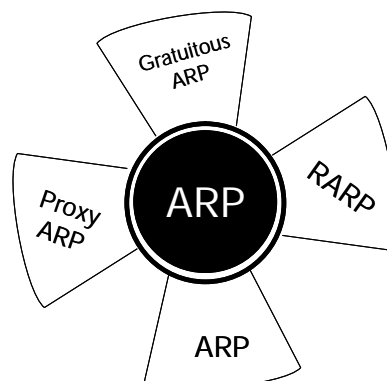
Q22. Define ARP. Explain different types of ARP.

Ans :

(Imp.)

There are four types of Address Resolution Protocol, which is given below:

1. Proxy ARP
2. Gratuitous ARP
3. Reverse ARP (RARP)
4. Inverse ARP



- i) **Proxy ARP:** Proxy ARP is a method through which a Layer 3 devices may respond to ARP requests for a target that is in a different network from the sender. The Proxy ARP configured router responds to the ARP and map the MAC address of the router with the target IP address and fool the sender that it is reached at its destination.

At the backend, the proxy router sends its packets to the appropriate destination because the packets contain the necessary information.

Example - If Host A wants to transmit data to Host B, which is on the different network, then Host A sends an ARP request message to receive a MAC address for Host B. The router responds to Host A with its own MAC address pretend itself as a destination. When the data is transmitted to the destination by Host A, it will send to the gateway so that it sends to Host B. This is known as proxy ARP.

- ii) **Gratuitous ARP** - Gratuitous ARP is an ARP request of the host that helps to identify the duplicate IP address. It is a broadcast request for the IP address of the router. If an ARP request is sent by a switch or router to get its IP address and no ARP responses are received, so all other nodes cannot use the IP address allocated to that switch or router. Yet if a router or switch sends an ARP request for its IP address and receives an ARP response, another node uses the IP address allocated to the switch or router.

There are some primary use cases of gratuitous ARP that are given below:

- The gratuitous ARP is used to update the ARP table of other devices.
- It also checks whether the host is using the original IP address or a duplicate one.

- iii) **Reverse ARP (RARP)** - It is a networking protocol used by the client system in a local area network (LAN) to request its IPv4 address from the ARP gateway router table. A table is created by the network administrator in the gateway-router that is used to find out the MAC address to the corresponding IP address.

When a new system is set up or any machine that has no memory to store the IP address, then the user has to find the IP address of the device. The device sends a RARP broadcast packet, including its own MAC address in the address field of both the sender and the receiver hardware. A host installed inside of the local network called the RARP-server is prepared to respond to such type of broadcast packet. The RARP server is then trying to locate a mapping table entry in the IP to MAC address. If any entry matches the item in the table, then the RARP server sends the response packet along with the IP address to the requesting computer.

- iv) **Inverse ARP (InARP)** - Inverse ARP is inverse of the ARP, and it is used to find the IP addresses of the nodes from the data link layer addresses. These are mainly used for the frame relays, and ATM networks, where Layer 2 virtual circuit addressing are often acquired from Layer 2 signaling. When using these virtual circuits, the relevant Layer 3 addresses are available.

ARP conversions Layer 3 addresses to Layer 2 addresses. However, its opposite address can be defined by InARP. The InARP has a similar packet format as ARP, but operational codes are different.

2.10 RARP

Q23. What is meant by RARP?

(OR)

Explain the concept of RARP.

Ans :

(Imp.)

Meaning

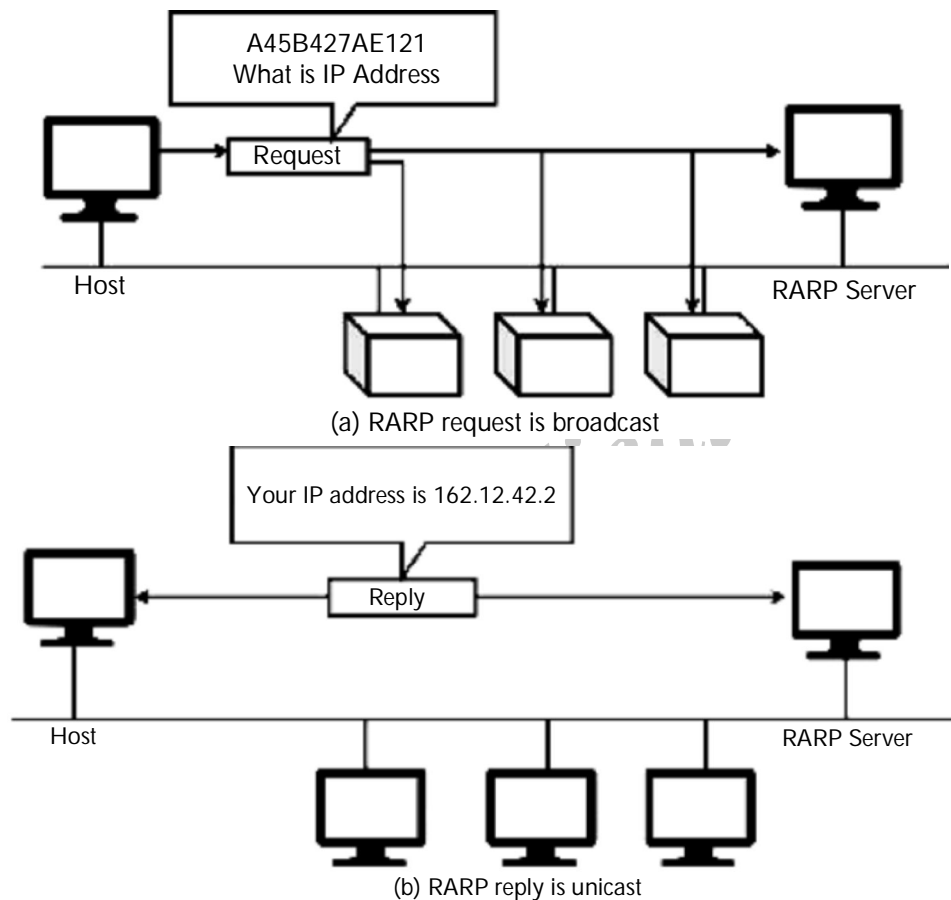
Reverse Address Resolution Protocol (RARP) is a network-specific standard protocol. It is described in RFC 903. Some network hosts, such as a diskless workstation, do not know their own IP address when they are booted. To determine their own IP address, they use a mechanism similar to ARP, but now the hardware address of the host is the known parameter, and the IP address is the queried parameter.

The reverse address resolution is performed the same way as the ARP address resolution. The same packet format is used for the ARP.

An exception is the operation code field that now takes the following values -

- for RARP request
- for RARP reply

The physical header of the frame will now indicate RARP as the higher-level protocol (8035 hex) instead of ARP (0806 hex) or IP (0800 hex) in the Ether type field.



When a framework with a local disk is bootstrapped, it generally accepts its IP address from a configuration document that's read from a disk file. But a system without a disk, including an X terminal or a diskless workstation, needs some other way to accept its IP address.

The feature of RARP is for the diskless framework to read its specific hardware address from the interface card and send a RARP request asking for someone to reply with the diskless systems IP address.

The format of a RARP packet is almost identical to an ARP packet. The only difference is that the frame type is 0X8035 for a RARP request or reply, and the op-field has a value of 3 for a RARP request and 4 for a RARP reply.

The problem with RARP includes its use of a link-layer broadcast, preventing most routers from forwarding a RARP request, and the minimal information returned just the system's IP address. While the RARP concept is easy, the implementation of a RARP server is system dependent.

Q24. What are the differences between ARP and RARP?

Ans :

(Imp.)

Sl. No	ARP	RARP
1.	ARP stands for Address Resolution Protocol.	Whereas RARP stands for Reverse Address Resolution Protocol.
2.	Through ARP, (32-bit) IP address mapped into (48-bit) MAC address.	Whereas through RARP, (48-bit) MAC address of 48 bits mapped into (32-bit) IP address.
3.	In ARP, broadcast MAC address is used.	While in RARP, broadcast IP address is used.
4.	In ARP, ARP table is managed or maintained by local host.	While in RARP, RARP table is managed or maintained by RARP server.
5.	In Address Resolution Protocol, Receiver's MAC address is fetched.	While in RARP, IP address is fetched.
6.	In ARP, ARP table uses ARP reply for its updation.	While in RARP, RARP table uses RARP reply for configuration of IP addresses .
7.	Hosts and routers uses ARP for knowing the MAC address of other hosts and routers in the networks.	While RARP is used by small users having less facilities.

UNIT III

Network Layer: - Distance Vector Routing, Link State Routing,
IP v4 addressing, Subnetting, CIDR., Introduction to IPv6
ICMP , IGMP, OSPF and BGP.

3.1 NETWORK LAYER

Q1. Define networking layer in TCP/IP Model.

Ans :

(Imp.)

Meaning

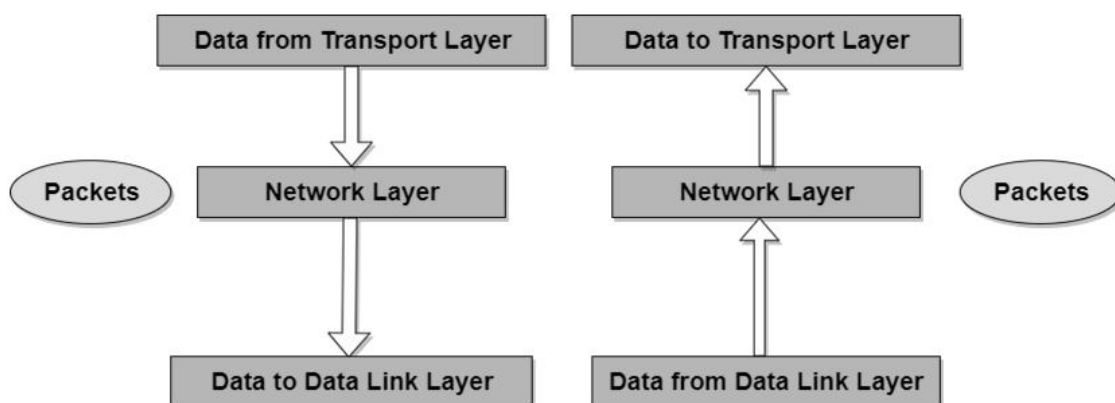
The network Layer controls the operation of the subnet. The main aim of this layer is to deliver packets from source to destination across multiple links (networks). If two computers (system) are connected on the same link, then there is no need for a network layer. It routes the signal through different channels to the other end and acts as a network controller.

It also divides the outgoing messages into packets and to assemble incoming packets into messages for higher levels.

In broadcast networks, the routing problem is simple, so the network layer is often thin or even non-existent.

Functions

1. It translates logical network address into physical address. Concerned with circuit, message or packet switching.
2. Routers and gateways operate in the network layer. Mechanism is provided by Network Layer for routing the packets to final destination.
3. Connection services are provided including network layer flow control, network layer error control and packet sequence control.
4. Breaks larger packets into small packets.



Design Issues with Network Layer

- 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 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 into one another's way, forming bottlenecks. The control of such congestion also belongs to the network layer.
- Moreover, the quality of service provided (delay, transmit 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 such as:
 - 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.

Q2. What is Internet working and explain its types ?

Ans :

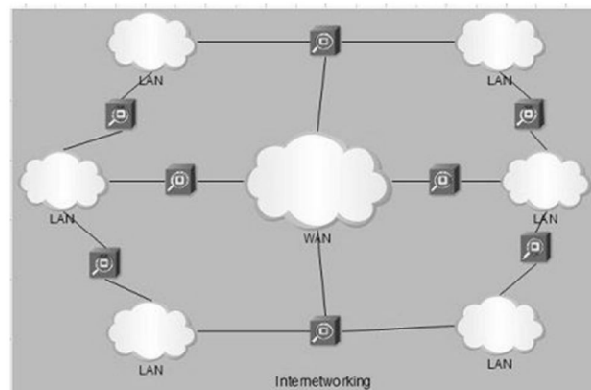
Meaning

Internet working started as a way to connect disparate types of computer networking technology. Computer network term is used to describe two or more computers that are linked to each other. When two or more computer LANs or WANs or computer network segments are connected using devices such as a router and configure by logical addressing scheme with a protocol such as IP, then it is called as computer internet working.

Internet working is a term used by Cisco. Any inter connection among or between public, private, commercial, industrial, or governmental computer

networks may also be defined as an inter network or "Internet working".

In modern practice, the interconnected computer networks or Inter networking use the Internet Protocol. Two architectural models are commonly used to describe the protocols and methods used in internet working. The standard reference model for internet working is Open Systems Inter connection (OSI).



Types

Internet working is implemented in Layer 3 (Network Layer) of this model. The most notable example of internet working is the Internet (capitalized). There are three variants of internetwork or Internet working, depending on who administers and who participates in them :

- (i) Extranet
- (ii) Intranet
- (iii) Internet

Intranets and extranets may or may not have connections to the Internet. If connected to the Internet, the intranet or extranet is normally protected from being accessed from the Internet without proper authorization. The Internet is not considered to be a part of the intranet or extranet, although it may serve as a portal for access to portions of an extranet.

(i) Extranet

An extranet is a network of internetwork or Internetworking that is limited in scope to a single organization or entity but which also has limited connections to the networks of one or more other usually, but not

necessarily, trusted organizations or entities. Technically, an extranet may also be categorized as a MAN, WAN, or other type of network, although, by definition, an extranet cannot consist of a single LAN; it must have at least one connection with an external network.

(ii) Intranet

An intranet is a set of interconnected networks or Internetworking, using the Internet Protocol and uses IP-based tools such as web browsers and ftp tools, that is under the control of a single administrative entity. That administrative entity closes the intranet to the rest of the world, and allows only specific users. Most commonly, an intranet is the internal network of a company or other enterprise. A large intranet will typically have its own web server to provide users with browse able information.

(iii) Internet

A specific Internetworking, consisting of a worldwide interconnection of governmental, academic, public, and private networks based upon the Advanced Research Projects Agency Network (ARPANET) developed by ARPA of the U.S. Department of Defense also home to the World Wide Web (WWW) and referred to as the 'Internet' with a capital 'I' to distinguish it from other generic internetworks. Participants in the Internet, or their service providers, use IP Addresses obtained from address registries that control assignments.

Q3. Define Packet Switching and explain its advantages and disadvantages.

Ans :

Meaning

Packet switching is a method of transferring the data to a network in form of packets. In order to transfer the file fast and efficient manner over the network and minimize the transmission latency, the data is broken into small pieces of variable length, called Packet. At the destination, all these small-parts (packets) has to be reassembled, belonging to the same file. A packet composes of payload and various control information. No pre-setup or reservation of resources is needed.

Packet Switching uses Store and Forward technique while switching the packets; while forwarding the packet each hop first store that packet then forward. This technique is very beneficial because packets may get discarded at any hop due to some reason. More than one path is possible between a pair of source and destination. Each packet contains Source and destination address using which they independently travel through the network. In other words, packets belonging to the same file may or may not travel through the same path. If there is congestion at some path, packets are allowed to choose different path possible over existing network.

Packet-Switched networks were designed to overcome the *weaknesses* of Circuit-Switched networks, since circuit-switched networks were not very effective for small messages.

Advantage

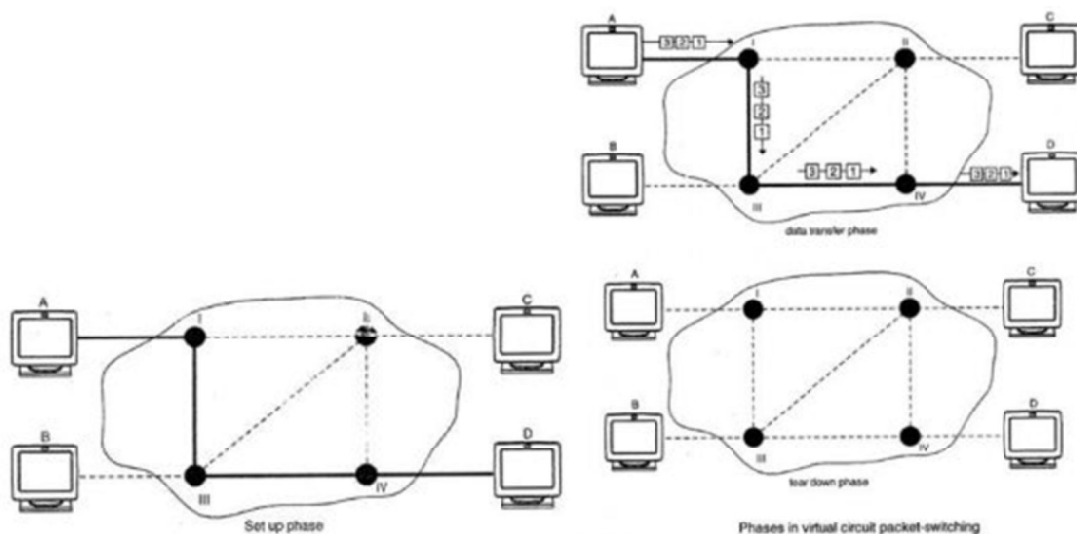
- More efficient in terms of bandwidth, since the concept of reserving circuit is not there.
- Minimal transmission latency.
- More reliable as destination can detect the missing packet.
- More fault tolerant because packets may follow different path in case any link is down, Unlike Circuit Switching.
- Cost effective and comparatively cheaper to implement.

Disadvantage

- Packet Switching don't give packets in order, whereas Circuit Switching provides ordered delivery of packets because all the packets follow the same path.
- Since the packets are unordered, we need to provide sequence numbers to each packet.
- Complexity is more at each node because of the facility to follow multiple path.
- Transmission delay is more because of rerouting.
- Packet Switching is beneficial only for small messages, but for bursty data (large messages) Circuit Switching is better.

Q4. Describe various Modes of Packet Switching.*Ans. :***1) Connection-oriented Packet Switching (Virtual Circuit)**

Before starting the transmission, it establishes a logical path or virtual connection using signalling protocol, between sender and receiver and all packets belongs to this flow will follow this predefined route. Virtual Circuit ID is provided by switches/routers to uniquely identify this virtual connection. Data is divided into small units and all these small units are appended with help of sequence number. Overall, three phases takes place here- Setup, data transfer and tear down phase.



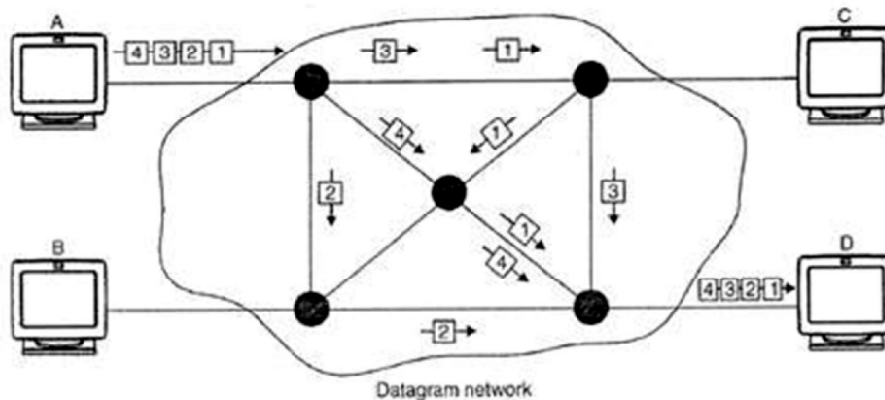
All address information is only transferred during setup phase. Once the route to destination is discovered, entry is added to switching table of each intermediate node. During data transfer, packet header (local header) may contain information such as length, time stamp, sequence number etc.

Connection-oriented switching is very useful in switched WAN. Some popular protocols which uses Virtual Circuit Switching approach are X.25, Frame-Relay, ATM and MPLS(Multi Protocol Label Switching).

2) Connectionless Packet Switching (Datagram)

Unlike Connection-oriented packet switching, In Connectionless Packet Switching each packet contains all necessary addressing information such as source address, destination address and port numbers etc. In Datagram Packet Switching, each packet is treated independently. Packets belongs to one flow may take different routes because routing decisions are made dynamically, so the packets arrived to destination might be out of order. It has no connection setup and tear down phase, like Virtual Circuits.

Packet delivery is not guaranteed in connectionless packet switching, so the reliable delivery must be provided by end systems using additional protocols.



Datagram Packet Switching

- a) A—R1—R2—B
- b) A is the sender (start)
- c) R1,R2 are two router that store and forward data
- d) B is receiver(destination)

To send packet from A to B there is delays since this is a Store and Forward network.

Delays in Packet switching:

- (i) Transmission Delay
- (ii) Propagation Delay
- (iii) Queuing Delay
- (iv) Processing Delay

i) Transmission Delay:

Time taken to put a packet onto link. In other words, it is simply time required to put data bits on the wire/communication medium. It depends on length of packet and bandwidth of network.

$$\text{Transmission Delay} = \text{Data size} / \text{bandwidth} = (L/B) \text{ second}$$

ii) Propagation delay :

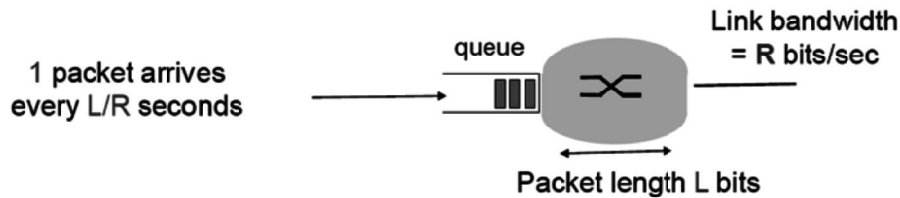
Time taken by the first bit to travel from sender to receiver end of the link. In other words, it is simply the time required for bits to reach destination from start point. Factors on which Propagation delay depends are Distance and propagation speed.

$$\text{Propagation delay} = \text{distance/transmission speed} = d/s$$

iii) Queuing Delay:

Queuing delay is the time a job waits in a queue until it can be executed. It depends on congestion. It the time difference between when packet arrived Destination and when the packet data was processed or executed. It may be caused by mainly three reasons i.e. originating switches, intermediate switches or call receiver servicing switches.

Queueing delay: illustration



$$\text{Arrival rate: } a = 1/(L/R) = R/L \text{ (packet/second)}$$

$$\text{Traffic intensity} = aL/R = (R/L)(L/R) = 1$$

$$\text{Average queueing delay} = 0$$

(queue is initially empty)

$$\text{Average Queueing delay} = (N-1)L/(2 \cdot R)$$

Where,

N = no. of packets

L = size of packet

R = bandwidth

iv) Processing Delay

Processing delay is the time it takes routers to process the packet header. Processing of packets help in detecting bit-level errors that occur during transmission of packet to destination. Processing delays in high-speed routers are typically on the order of microseconds or less.

In simple words, it is just the time taken to process packets.

Total time or End-to-End time = Transmission delay + Propagation delay + Queuing delay + Processing delay

For M hops and N packets

Total delay = $M \cdot (\text{Transmission delay} + \text{propagation delay}) + (M-1) \cdot (\text{Processing delay} + \text{Queuing delay}) + (N-1) \cdot (\text{Transmission delay})$

For N connecting link in the circuit

Transmission delay = $N \cdot L/R$

Propagation delay = $N \cdot (d/s)$

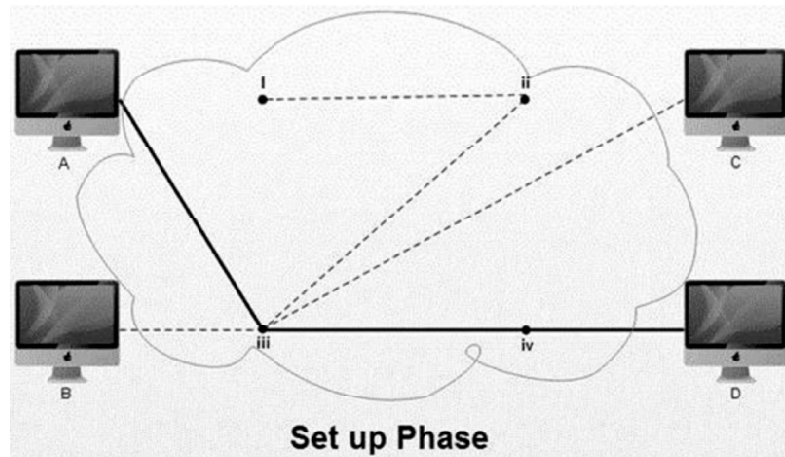
Q5. Describe various types of Packet Switching.

Ans :

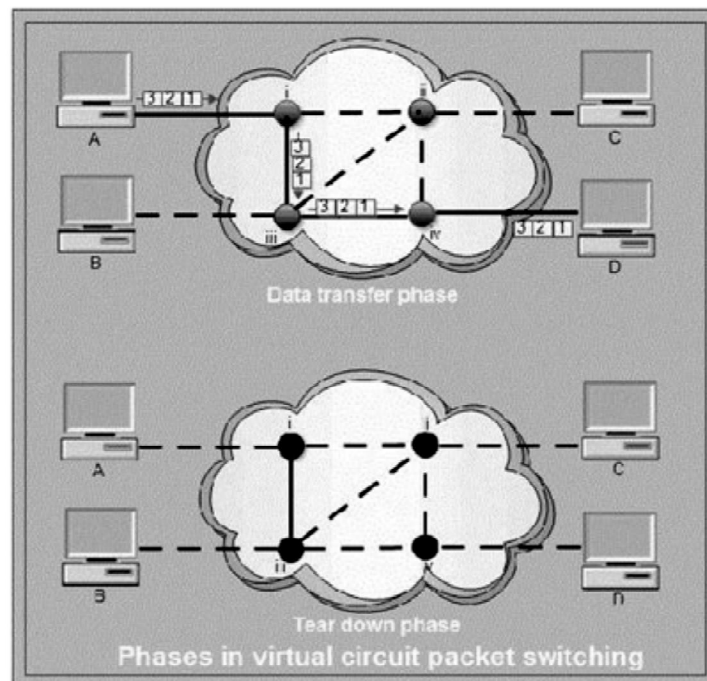
The packet switching has two approaches: Virtual Circuit approach and Datagram approach. WAN, ATM, frame relay and telephone networks use connection oriented virtual circuit approach; whereas internet relies on connectionless datagram based packet switching.

1) Virtual Circuit Packet Switching

In virtual circuit packet switching, a single route is chosen between the sender and receiver and all the packets are sent through this route. Every packet contains the virtual circuit number. As in circuit switching, virtual circuit needs call setup before actual transmission can be started. The routing is based on the virtual circuit number.



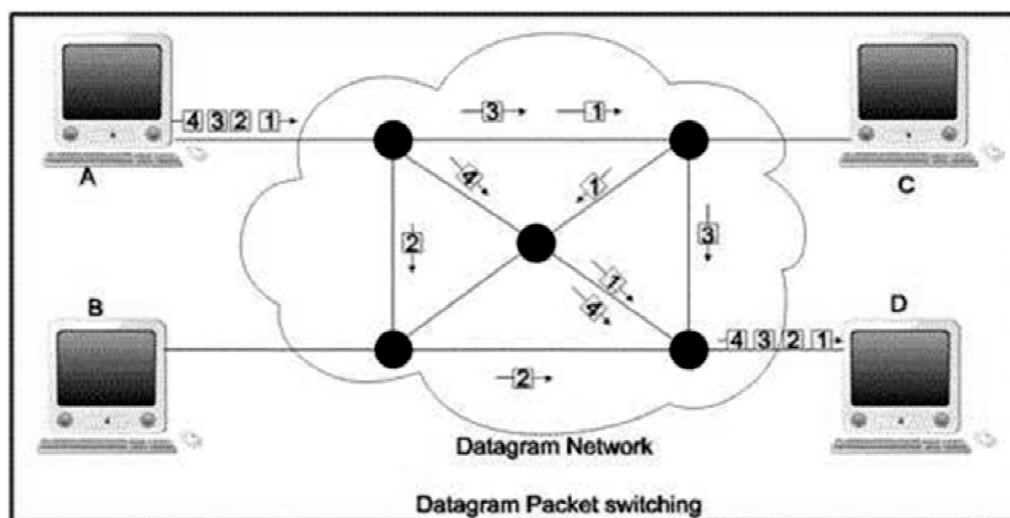
This approach preserves the relationship between all the packets belonging to a message. Just like circuit switching, virtual circuit approach has a setup, data transfer and tear down phases. Resources can be allocated during the set up phase, as in circuit switched networks or on demand, as in a datagram network. All the packets of a message follow the same path established during the connection. A virtual circuit network is normally implemented in the data link layer, while a circuit switched network is implemented in the physical layer and a datagram network in the network layer.



2) Datagram Packet Switching

In datagram packet switching each packet is transmitted without any regard to other packets. Every packet contains full packet of source and destination. Every packet is treated as individual, independent transmission.

Even if a packet is a part of multi-packet transmission the network treats it as though it existed alone. Packets in this approach are called datagrams. Datagram switching is done at the network layer. Figure shows how a datagram approach is used to deliver four packets from station A to station D. All the four packets belong to same message but they may travel via different paths to reach the destination i.e. station D.



Datagram approach can cause the datagrams to arrive at their destination out of order with different delays between the packets. Packets may also be lost or dropped because of lack of resources. The datagram networks are also referred to as connectionless networks. Here connectionless means that the switch does not keep information about connection state. There are no connection establishment or tear down phases.

The datagram can arrive at the destination with a different order from the order in which they were sent. The source and destination address are used by the routers to decide the route for packets. Internet uses datagram approach at the network layer.

Q6. List the differences between Virtual Circuits & Datagram Networks.

Ans :

Computer networks that provide connection-oriented service are called Virtual Circuits while those providing connection-less services are called as Datagram networks. For prior knowledge, the Internet which we use is actually based on Datagram network (connection-less) at network level as all packets from a source to a destination do not follow the same path.

Let us see what are the highlighting differences between these two hot debated topics here:

Virtual Circuits

1. It is connection-oriented simply meaning that there is a reservation of resources like buffers, CPU, bandwidth, etc. for the time in which the newly setup VC is going to be used by a data transfer session.
2. First packet goes and reserves resources for the subsequent packets which as a result follow the same path for the whole connection time.
3. Since all the packets are going to follow the same path, a global header is required only for the first packet of the connection and other packets generally don't require global headers.
4. Since data follows a particular dedicated path, packets reach in order to the destination.
5. From above points, it can be concluded that Virtual Circuits are highly reliable means of transfer.
6. Since each time a new connection has to be setup with reservation of resources and extra information handling at routers, it's simply costly to implement Virtual Circuits.

Datagram Networks

1. It is connectionless service. There is no need of reservation of resources as there is no dedicated path for a connection session.
2. All packets are free to go to any path on any intermediate router which is decided on the go by dynamically changing routing tables on routers.
3. Since every packet is free to choose any path, all packets must be associated with a header with proper information about source and the upper layer data.
4. The connectionless property makes data packets reach destination in any order, means they need not reach in the order in which they were sent.
5. Datagram networks are not reliable as Virtual Circuits.
6. But it is always easy and cost efficient to implement datagram networks as there is no extra headache of reserving resources and making a dedicated each time an application has to communicate.

S.No	Nature	Virtual Circuit	Data gram Networks
1.	Addressing	Packet contains full source and destination address	Packet contains short virtual circuit number identifier.
2.	State information	None other than router table containing destination network	Each virtual circuit number entered to table on setup, used for routing.
3.	Routing	Packets routed independently	Route established at setup, all packets follow same route.
4.	Effect of router failure	Only on packets lost during crash	All virtual circuits passing through failed router terminated.
5.	Congestion control	Difficult since all packets routed independently router resource requirements can vary.	Simple by pre -allocating enough buffers to each virtual circuit at setup, since maximum number of circuits fixed.

3.2 ROUTERS

Q7. What is Router ? Explain the Characteristics of Routers/Router Protocols.

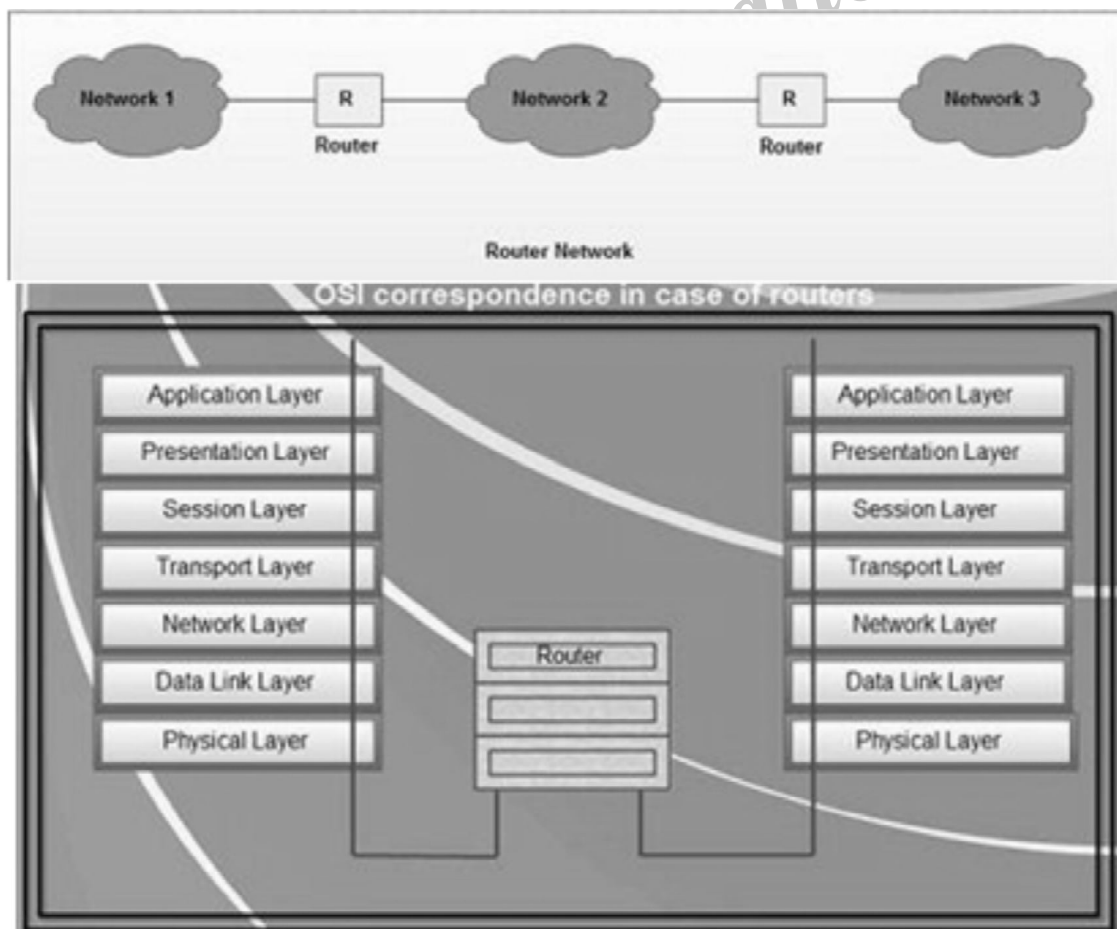
Ans :

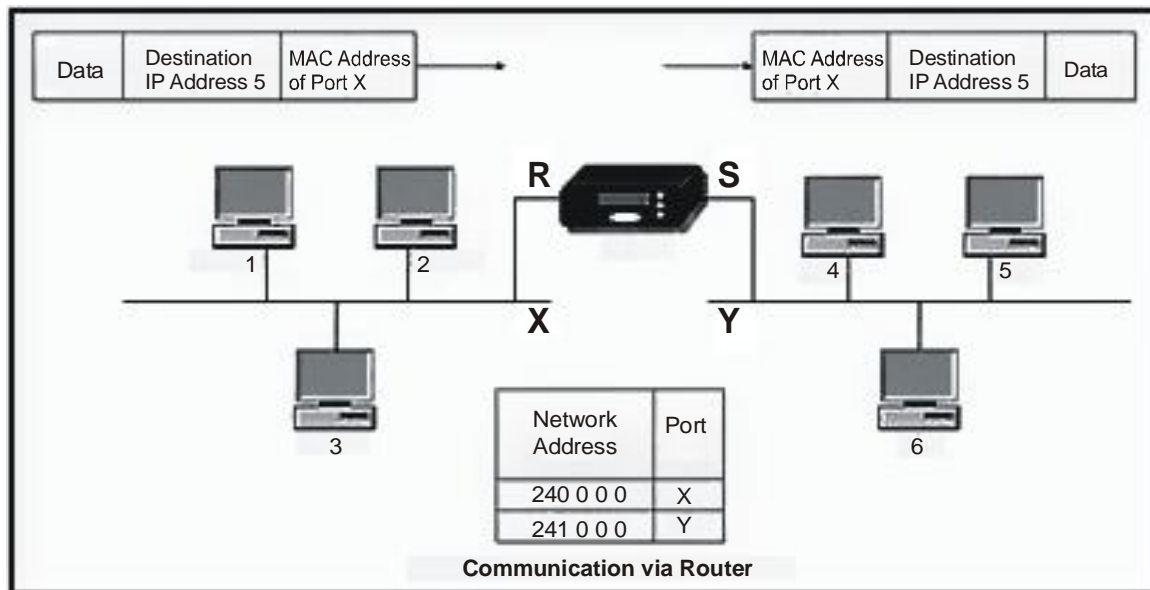
(Imp.)

Routers are used to connect both similar and dissimilar LANs. Router operates on the network layer of OSI model using the physical layer, data link layer and network layer to provide connectivity, addressing and switching. These are highly intelligent devices. In case of TCP/IP network, Internet Protocol (IP) is used as addresses for network; this is the router which interprets the IP address and delivers the packet reliably.

Now, we may say that router transmits the network layer data and therefore, provides transmission of data between LANs that use different data link protocols but using the same network layer protocol. Because of this, Ethernet can be connected with token ring network using routers. Additionally, routers provide connectivity to LAN (SMDS) and WAN (X.25, Frame Relay and ATM). Routers are protocol sensitive; typically supporting multiple protocols and large and varying packet sizes such as might be involved in supporting both Ethernet and Token Ring.

A network consisting of routers can have multiple paths unlike bridges. Normally, the shortest of all paths in the network is used to transfer packets.





Consider a case for data transmission from computer 1 to computer 5 on the network. When computer 1 starts sending the data, it compares its IP address with computer 5, i.e. destination computer addresses to know whether computer 5 lies on its own network or not. When computer 1 finds that it is not on its network, it transmits a data packet containing the MAC address R of router in this case. When router receives this packet, it sets the MAC address of computer 5 and sends packet to the port which has the same IP destination address as given in data packet. In this manner computer 5 receives the data packet.

Characteristics

1. Routers are multiport devices with high - speed backbones
2. Routers also support filtering and encapsulation like bridges
3. Like bridges routers are also self-learning, as they can communicate their existence. to other devices and can learn of the existence of new routers, nodes and LAN segments
4. As explained earlier, they route traffic by considering the network as a whole. It shows that they use a high level of intelligence to accomplish this task. This characteristic makes them superior than hubs and bridges because they simply view the network on a link-by-link basis
5. The packet handled by router may include destination address, packet priority level, least-cost route, minimum route delay, minimum route distance, and route congestion level
6. Routers constantly monitor the condition of the network, as a whole to dynamically adapt to changes in the condition of the network
7. They typically provide some level of redundancy so that they are less susceptible to catastrophic failure.

Router Protocols

Router protocols consist of both bridging and routing protocols as listed below:

➤ **Inter-router Protocols**

These are router-to-router protocols that can operate over dissimilar networks. This protocol routes information and stores data packets during periods of idleness.

➤ **Serial Line Protocols**

This protocol is widely used over serial or dialup links connecting unlike routers. Examples include HDLC, SLIP (Serial Line Interface Protocol), and PPP (Point-to-Point Protocol).

➤ **Protocol Stack Routing and Bridging Protocols**

This advises the router as to which packets should be routed and which should be bridged.

3.2.1 Distance vector Routing

Q8. What is Distance Vector Protocol Routing ?

Ans :

(Imp.)

The distance-vector routing Protocol is a type of algorithm used by routing protocols to discover routes on an interconnected network. The primary distance-vector routing protocol algorithm is the Bellman-Ford algorithm. Another type of routing protocol algorithm is the *link-state* approach.

Routing protocols that use distance-vector routing protocols include RIP (Routing Information Protocol), Cisco's IGRP (Internet Gateway Routing Protocol), and Apple's RTMP (Routing Table Maintenance Protocol). The most common link-state routing protocol is OSPF (Open Shortest Path First). Dynamic routing, as opposed to static (manually entered) routing, requires routing protocol algorithms.

- (i) **Dynamic routing protocols** assist in the automatic creation of routing tables. Network topologies are subject to change at any time. A link may fail unexpectedly, or a new link may be added. A dynamic routing protocol must discover these changes, automatically adjust its routing tables, and inform other routers of the changes.

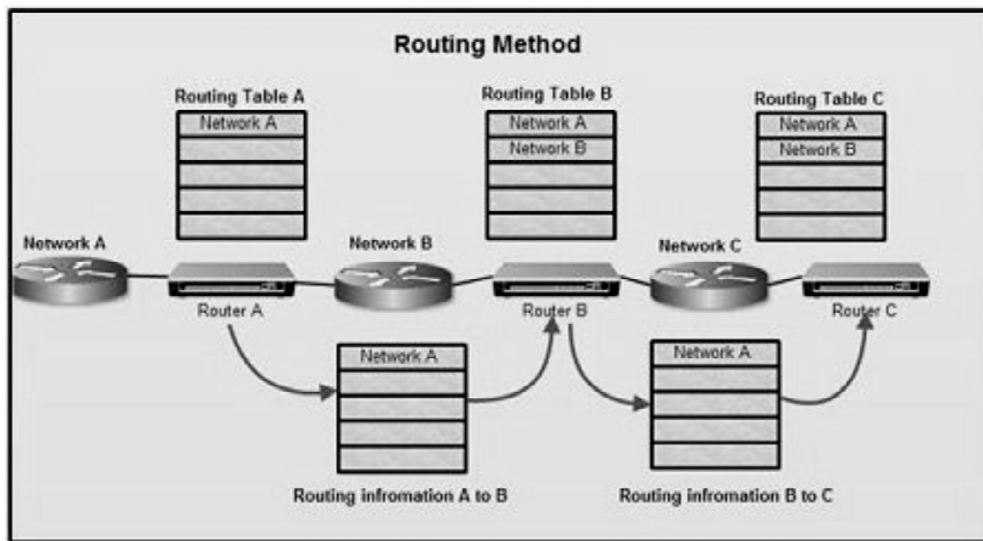
The process of rebuilding the routing tables based on new information is called *convergence*.

- (ii) **Distance-vector routing** refers to a method for exchanging route information. A router will advertise a route as a vector of *direction* and *distance*.

Direction refers to a port that leads to the next router along the path to the destination, and distance is a metric that indicates the number of hops to the destination, although it may also be an arbitrary value that gives one route precedence over another. Inter network routers exchange this vector information and build route lookup tables from it.

- (iii) **Distance vector protocols are RIP, Interior Gateway Routing Protocol (IGPR).**

Algorithm where each router exchanges its routing table with each of its neighbors. Each router will then merge the received routing tables with its own table, and then transmit the merged table to its neighbors. This occurs dynamically after a fixed time interval by default, thus requiring significant link overhead.



There are problems, however, such as:

- If exchanging data among routers every 90 seconds for example, it takes 90×10 seconds that a router detects a problem in router 10, routers ahead and the route cannot be changed during this period.
- Traffic increases since routing information is continually exchanged.
- There is a limit to the maximum amount of routing information (15 for RIP), and routing is not possible on networks where the number of hops exceeds this maximum.
- Cost data is only the number of hops, and so selecting the best path is difficult.
- However, routing processing is simple, and it is used in small-scale networks in which the points mentioned above are not a problem.

RIP (Routing Information Protocol)

RIP is the most widely used routing protocol of distance-vector type today. It has been originally designed based on the routing protocol applied to XNS and PUP protocol systems of Xerox (RFC 1058).

- RIP request is used, by a router upon startup to inquire of its neighbor router about route information to obtain routing information.
- RIP response includes a destination host address and cost information in the address part. Response is sent to the neighbor router in case of the following:

1. Receipt of RIP request
2. Regularly

Response is sent every 30 seconds even if no RIP request is issued. All routers delete route information from their routing table if no route information is received within a specified period of time. This is intended to allow detection of fault of neighbor router.

- In case of changes made to routing table contents. If changes are made to the routing table because changes to the network configuration have been detected, information relating to these changes is sent to the neighbor router.

Distance Vector Algorithm

1. A router transmits its distance vector to each of its neighbors in a routing packet.
2. Each router receives and saves the most recently received distance vector from each of its neighbors.
3. A router recalculates its distance vector when:
 - It receives a distance vector from a neighbor containing different information than before.
 - It discovers that a link to a neighbor has gone down.
 - The DV calculation is based on minimizing the cost to each destination

$D_x(y)$ = Estimate of least cost from x to y

$C(x,v)$ = Node x knows cost to each neighbor v

$D_x = [D_x(y): y \in N]$ = Node x maintains distance vector

Node x also maintains its neighbors' distance vectors

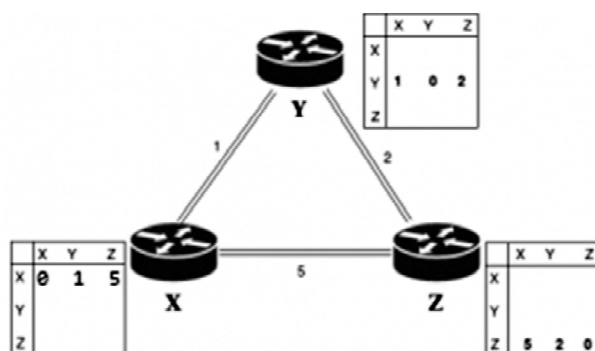
– For each neighbor v, x maintains $D_v = [D_v(y): y \in N]$

Note

- From time-to-time, each node sends its own distance vector estimate to neighbors.
- When a node x receives new DV estimate from any neighbor v, it saves v's distance vector and it updates its own DV using B-F equation:

$$D_x(y) = \min \{ C(x,v) + D_v(y) \} \text{ for each node } y \in N$$

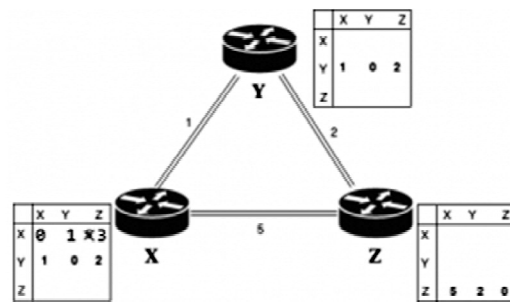
Example: Consider 3-routers X, Y and Z as shown in figure. Each router have their routing table. Every routing table will contain distance to the destination nodes.



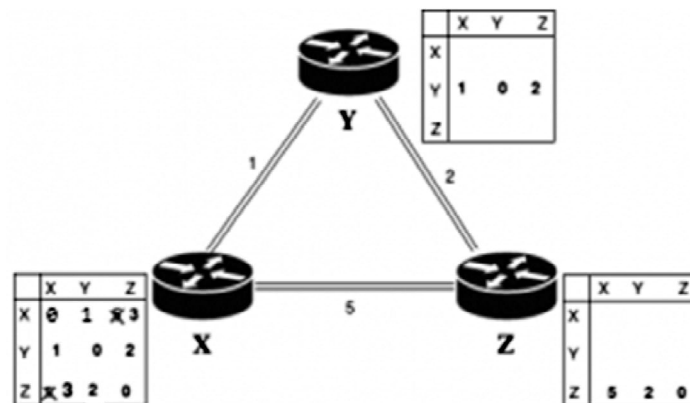
Consider router X, X will share its routing table to neighbors and neighbors will share their routing table to it and distance from node X to destination will be calculated using Bellman-Ford equation.

$$D_x(y) = \min \{ C(x,v) + D_v(y) \} \text{ for each node } y \in N$$

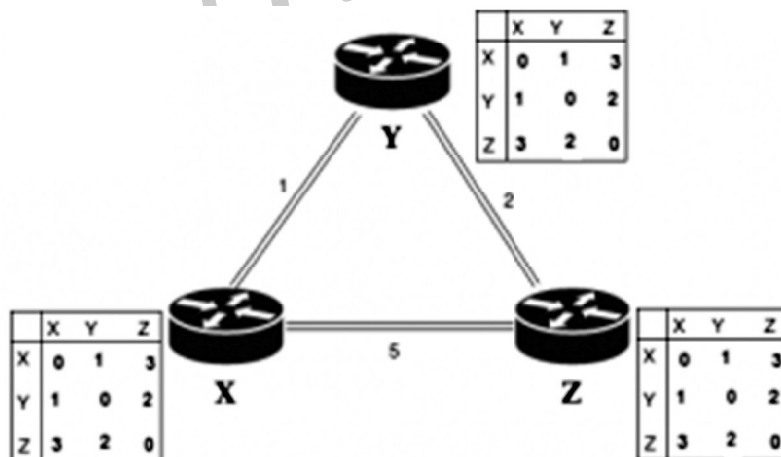
As we can see that distance will be less going from X to Z when Y is intermediate node(hop) so it will be updated in routing table X.



Similarly for Z also –



Finally the routing table for all –



Advantage

- It is simpler to configure and maintain than link state routing.

Disadvantages

- It is slower to converge than link state.
- It is at risk from the count-to-infinity problem.

- It creates more traffic than link state since a hop count change must be propagated to all routers and processed on each router. Hop count updates take place on a periodic basis, even if there are no changes in the network topology, so bandwidth-wasting broadcasts still occur.
- For larger networks, distance vector routing results in larger routing tables than link state since each router must know about all other routers. This can also lead to congestion on WAN links.

Note – Distance Vector routing uses UDP(User datagram protocol) for transportation.

3.3 LINK-STATE ROUTING

Q9. What is Link State Routing Protocol ?

Ans :

(Imp.)

Link-state algorithms (also known as shortest path first algorithms) flood only incremental changes that have occurred since the last routing table update. During this incremental update, each router sends only that portion of the routing table that describes the state of its own links, as opposed to its entire routing table.

Link-state routing protocols require routers to periodically send routing updates to their neighboring routers in the internetwork. In addition, link-state routing protocols are quick to converge their routing updates across the network in comparison to distance vector protocols.

The speed at which they converge makes link-state protocols less prone to routing loops than distance vector protocols. However, link-state protocols also require more CPU power and system memory. One of the primary reasons that additional CPU power and memory are needed is that link-state protocols are based on the distributed map concept, which means that every router has a copy of the network map that is regularly updated. In addition to the size of the routing table, the number of routers in an area and the number of adjacencies amongst routers also has an effect on router memory and CPU usage in link state protocols. These factors were obvious in the old fully meshed asynchronous transfer mode (ATM) networks, where some routers had 50 or more OSPF adjacent peers and performed poorly.

Link-state protocols are based on link-state algorithms, which are also called shortest path first (SPF) algorithms or Dijkstra algorithms. "SPF in Operation," later in this tutorial, covers the SPF algorithm in more detail.

A simple way to understand how link-state technology operates is to picture the network as a large jigsaw puzzle; the number of pieces in your puzzle depends on the size of your network. Each piece of the puzzle holds only one router or one LAN. Each router "draws" itself on that jigsaw piece, including arrows to other routers and LANs. Those pieces are then replicated and sent throughout the network from router to router (via link-state advertisements [LSAs]), until each router has a complete and accurate copy of each piece of the puzzle. Each router then assembles these pieces by using the SPF algorithm.

NOTE

The principle of link-state routing is that all the routers within an area maintain an identical copy of the network topology. From this map, each router performs a series of calculations that determine the best routes. This network topology is contained within a link-state database, where each record represents the links to a particular node in the network.

The steps in link state routing mainly consist of five parts. Each router in the network must do the following things to make it work:

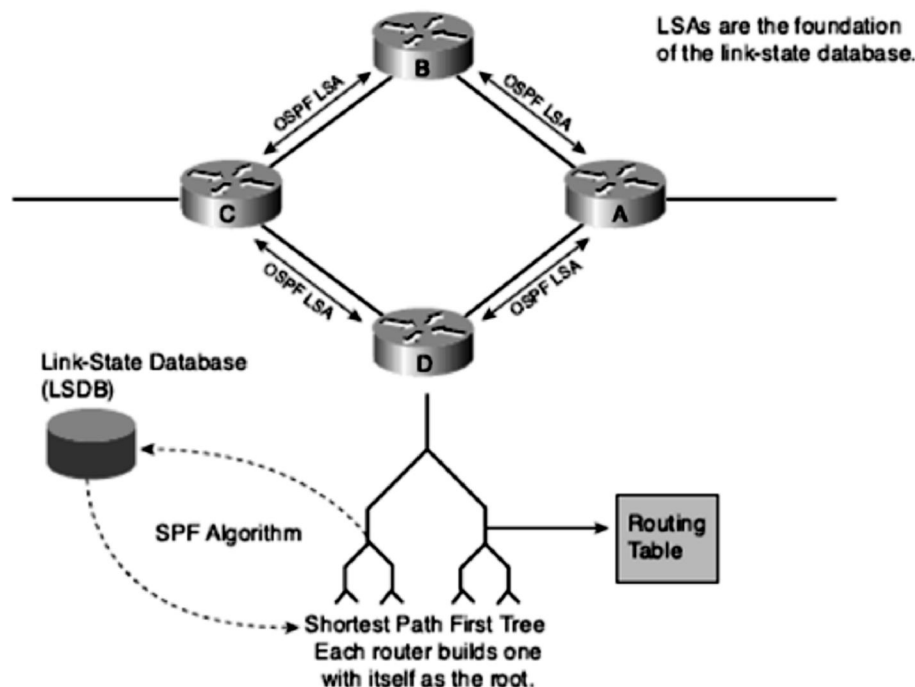
1. Discovering its neighbors and learning their network addresses.
2. Setting the distance or cost metric to each of its neighbors.
3. Constructing a packet telling all it has just learned.
4. Sending this packet to and receive packets from all other routers.
5. Computing the shortest path to every other router.

Each record contains the following pieces of information:

- Interface identifier
- Link number
- Metric information regarding the state of the link

Armed with that information, each router can quickly compute the shortest path from itself to all other routers.

The SPF algorithm determines how the various pieces of the puzzle fit together. Figure below illustrates all of these pieces put together in operation.



Link-state protocols such as OSPF flood all the routing information when they first become active in link-state packets. After the network converges, they send only small updates via link-state packets

Advantages

Link-state routing protocols have several advantages compared to distance vector routing protocols.

➤ **Builds a topological map**

A topological map or SPF tree of the network topology is created by Link-state routing protocols. Since link-state routing protocols exchange link-states, SPF tree of the network can be developed using SPF algorithm. By the help of SPF tree, the shortest path to every network can be found by each router independently.

➤ **Fast convergence**

During the process of receiving an LSP, link-state routing protocols immediately flood the LSP out all interfaces except for the interface from which it receives LSP. In contrast, RIP requires processing each routing update and updating its routing table before it floods the routing update on other interfaces.

➤ **Event-driven updates**

When the initial flooding of LSPs is carried out, link-state routing protocols only transmit an LSP whenever there is a modification in the network topology. The LSP contains only the information that is related to affected link. Comparing to distance vector routing, link-state routing protocols do not send updates at fixed intervals of time.

➤ **Hierarchical design**

Link-state routing protocols are based on the idea of areas. Multiple areas develop a hierarchical design to networks that allow for better route aggregation and the separation of routing issues within an area.

Disadvantages

➤ Link-state protocols also have some demerits compared to distance vector routing protocols.

➤ **Memory requirements**

Additional memory to develop and maintain the link-state database and SPF tree is needed by Link-state protocols.

➤ **Processing requirements**

It also requires more CPU processing as compared to distance vector routing protocols. The Shortest Path Algorithm (SPF algorithm) needs more CPU time as compared to distance vector algorithms, especially Bellman-Ford because link-state protocols make a full map of the topology.

➤ **Bandwidth requirements**

While flooding the link-state packets, it may have the effect on the available bandwidth on a network.

Q10. What are the differences between distance Vector Routing and link State Routing ?

Ans :

S.No.	Distance Vector Routing	Link State Routing
1.	Distance vector protocols use a distance calculation plus an outgoing network interface (a vector) to choose the best path to a destination network	Link State protocols track the status and connection type of each link and produces a calculated metric based on these and other factors, including some set by the network administrator
2.	Each router maintains routing table indexed by and containing one entry for each router in the subnet	It is the advanced version of distance vector routing
3.	Algorithm took too large to converge	Algorithm is faster
4.	Distance Vector routing protocols support dis-contiguous subnets	Link State routing protocols support contiguous subnets
5.	Distance Vector routing protocols uses hop count and composite metric	Cost is the metric of the Link State routing protocols
6.	Bandwidth is less	Wide bandwidth is available
7.	Router measure delay directly with special ECHO packets	All delays measured and distributed to every router
8.	It doesn't take line bandwidth into account when choosing the routes	It considers the line bandwidth into account when choosing the routes
9.	Distance Vector routing protocols are less scalable such as RIP supports 16 hops and IGRP has a maximum of 100 hops	Link State routing protocols are very much scalable; supports infinite hops
10.	Distance vector require less memory	Link state require more memory

3.4 IPv4 ADDRESSING

Q11. Explain IPv4 addressing methods.

Ans :

(Imp.)

Every location or device on a network must be *addressable*. This is simply a term that means that it can be reached by referencing its designation under a predefined system of addresses. In the normal TCP/IP model of network layering, this is handled on a few different layers, but usually, when we refer to an address on a network, we are talking about an IP address.

IP addresses allow network resources to be reached through a network interface. If one computer wants to communicate with another computer, it can address the information to the remote computer's IP address. Assuming that the two computers are on the same network, or that the different computers and devices in between can translate requests across networks, the computers should be able to reach each other and send information.

Each IP address must be unique on its own network. Networks can be isolated from one another, and they can be bridged and translated to provide access between distinct networks. A system called Network Address Translation, allows the addresses to be rewritten when packets traverse network borders to allow them to continue on to their correct destination. This allows the same IP address to be used on multiple, isolated networks while still allowing these to communicate with each other if configured correctly.

The difference between IPv4 and IPv6

There are two revisions of the IP protocol that are widely implemented on systems today. IPv4, which is the fourth version of the protocol, currently is what the majority of systems support. The newer, sixth revision, called IPv6, is being rolled out with greater frequency due to improvements in the protocol and the limitations of IPv4 address space. Simply put, the world now has too many internet-connected devices for the amount of addresses available through IPv4.

IPv4 addresses are 32-bit addresses. Each byte, or 8-bit segment of the address, is divided by a period and typically expressed as a number 0-255. Even though these numbers are typically expressed in decimal to aid in human comprehension, each segment is usually referred to as an **octet** to express the fact that it is a representation of 8 bits.

A typical IPv4 address looks something like this:

192.168.0.5

The lowest value in each octet is a 0, and the highest value is 255.

We can also express this in binary to get a better idea of how the four octets will look. We will separate each 4 bits by a space for readability and replace the dots with dashes:

1100 0000 - 1010 1000 - 0000 0000 - 0000 0101

Recognizing that these two formats represent the same number will be important for understanding concepts later on.

Although there are some other differences in the protocol and background functionality of IPv4 and IPv6, the most noticeable difference is the address space. IPv6 expresses addresses as an 128-bit number. To put that into perspective, this means that IPv6 has space for more than 7.9×10^{28} times the amount of addresses as IPv4.

To express this extended address range, IPv6 is generally written out as eight segments of four hexadecimal digits. Hexadecimal numbers represent the numbers 0-15 by using the digits 0-9, as well as the numbers a-f to express the higher values. A typical IPv6 address might look something like this:

1203:8fe0:fe80:b897:8990:8a7c:99bf:323d

You may also see these addresses written in a compact format. The rules of IPv6 allow you to remove any leading zeros from each octet, and to replace a single range of zeroed groups with a double colon (::).

For instance, if you have one group in an IPv6 address that looks like this:

...:00bc:...

You could instead just type:

...:bc:...

To demonstrate the second case, if you have a range in an IPv6 address with multiple groups as zeroes, like this:

...:18bc:0000:0000:0000:00ff:...

You could compact this like so (also removing the leading zeros of the group like we did above):

...:18bc::ff...

You can do this only once per address, or else the full address will be unable to be reconstructed.

While IPv6 is becoming more common every day, in this guide, we will be exploring the remaining concepts using IPv4 addresses because it is easier to discuss with a smaller address space.

IPv4 Addresses Classes and Reserved Ranges

IP addresses are typically made of two separate components. The first part of the address is used to identify the network that the address is a part of. The part that comes afterwards is used to specify a specific host within that network.

Where the network specification ends and the host specification begins depends on how the network is configured. We will discuss this more thoroughly momentarily.

IPv4 addresses were traditionally divided into five different "classes", named A through E, meant to differentiate segments of the available addressable IPv4 space. These are defined by the first four bits of each address. You can identify what class an IP address belongs to by looking at these bits.

Here is a translation table that defines the addresses based on their leading bits:

➤ **Class A**

0 — : If the first bit of an IPv4 address is "0", this means that the address is part of class A. This means that any address from 0.0.0.0 to 127.255.255.255 is in class A.

➤ **Class B**

10— : Class B includes any address from 128.0.0.0 to 191.255.255.255. This represents the addresses that have a "1" for their first bit, but don't have a "1" for their second bit.

➤ **Class C**

110- : Class C is defined as the addresses ranging from 192.0.0.0 to 223.255.255.255. This represents all of the addresses with a "1" for their first two bits, but without a "1" for their third bit.

➤ **Class D**

1110 : This class includes addresses that have "111" as their first three bits, but a "0" for the next bit. This address range includes addresses from 224.0.0.0 to 239.255.255.255.

➤ **Class E**

1111 : This class defines addresses between **240.0.0.0 and 255.255.255.255**. Any address that begins with four "1" bits is included in this class.

Class D addresses are reserved for multi-casting protocols, which allow a packet to be sent to a group of hosts in one movement. Class E addresses are reserved for future and experimental use, and are largely not used.

Traditionally, each of the regular classes (A-C) divided the networking and host portions of the address differently to accommodate different sized networks. Class A addresses used the remainder of the first octet to represent the network and the rest of the address to define hosts. This was good for defining a few networks with a lot of hosts each.

The class B addresses used the first two octets (the remainder of the first, and the entire second) to define the network and the rest to define the hosts on each network. The class C addresses used the first three octets to define the network and the last octet to define hosts within that network.

The division of large portions of IP space into classes is now almost a legacy concept. Originally, this was implemented as a stop-gap for the problem of rapidly depleting IPv4 addresses (you can have multiple computers with the same host if they are in separate networks). This was replaced largely by later schemes that we will discuss below.

Reserved Private Ranges

There are also some portions of the IPv4 space that are reserved for specific uses.

One of the most useful reserved ranges is the loop back range specified by addresses from 127.0.0.0 to 127.255.255.255. This range is used by each host to test networking to itself. Typically, this is expressed by the first address in this range: 127.0.0.1.

Each of the normal classes also have a range within them that is used to designate private network addresses. For instance, for class A addresses, the addresses from 10.0.0.0 to 10.255.255.255 are reserved for private network assignment. For class B, this range is 172.16.0.0 to 172.31.255.255. For class C, the range of 192.168.0.0 to 192.168.255.255 is reserved for private usage.

Any computer that is not hooked up to the internet directly (any computer that goes through a router or other NAT system) can use these addresses at will.

There are additional address ranges reserved for specific use-cases. You can find a summary of reserved addresses [here](#).

Net masks and Subnets

The process of dividing a network into smaller network sections is called subnetting. This can be useful for many different purposes and helps isolate groups of hosts together and deal with them easily.

As we discussed above, each address space is divided into a network portion and a host portion. The amount the address that each of these take up is dependent on the class that the address belongs to. For instance, for class C addresses, the first 3 octets are used to describe the network. For the address 192.168.0.15, the 192.168.0 portion describes the network and the 15 describes the host.

By default, each network has only one subnet, which contains all of the host addresses defined within. A netmask is basically a specification of the amount of address bits that are used for the network portion. A subnet mask is another netmask within used to further divide the network.

Each bit of the address that is considered significant for describing the network should be represented as a "1" in the netmask.

For instance, the address we discussed above, 192.168.0.15 can be expressed like this, in binary:

1100 0000 - 1010 1000 - 0000 0000 - 0000 1111

As we described above, the network portion for class C addresses is the first 3 octets, or the first 24 bits. Since these are the significant bits that we want to preserve, the netmask would be:

1111 1111 - 1111 1111 - 1111 1111 - 0000 0000

This can be written in the normal IPv4 format as 255.255.255.0. Any bit that is a "0" in the binary representation of the netmask is considered part of the host portion of the address and can be variable. The bits that are "1" are static, however, for the network or subnetwork that is being discussed.

We determine the network portion of the address by applying a bitwise AND operation to between the address and the netmask. A bitwise AND operation will basically save the networking portion of the address and discard the host portion. The result of this on our above example that represents our network is:

1100 0000 - 1010 1000 - 0000 0000 - 0000 0000

This can be expressed as 192.168.0.0. The host specification is then the difference between these original value and the host portion. In our case, the host is "0000 1111" or 15.

The idea of subnetting is to take a portion of the host space of an address, and use it as an additional networking specification to divide the address space again.

For instance, a netmask of 255.255.255.0 as we saw above leaves us with 254 hosts in the network (you cannot end in 0 or 255 because these are reserved). If we wanted to divide this into two subnetworks, we could use one bit of the conventional host portion of the address as the subnet mask.

So, continuing with our example, the networking portion is:

1100 0000 - 1010 1000 - 0000 0000

The host portion is:

0000 1111

We can use the first bit of our host to designate a subnetwork. We can do this by adjusting the subnet mask from this:

1111 1111 - 1111 1111 - 1111 1111 - 0000 0000

To this:

1111 1111 - 1111 1111 - 1111 1111 - 1000 0000

In traditional IPv4 notation, this would be expressed as 192.168.0.128. What we have done here is to designate the first bit of the last octet as significant in addressing the network. This effectively produces two sub-networks. The first sub-network is from 192.168.0.1 to 192.168.0.127. The second sub-network contains the hosts 192.168.0.129 to 192.168.0.255. Traditionally, the subnet itself must not be used as an address.

If we use more bits out of the host space for networking, we can get more and more sub-networks.

3.5 SUBNETTING

Q12. Explain the concept of subnetting. State its advantages and disadvantages.

Ans :

(Imp.)

Subnetting is a process of dividing a single large network in multiple smaller networks. A single large network is just like a town without any sector and street address. In such a town, a postman may take 3 to 4 days in finding a single address. While if town is divided in sectors and streets, he can easily find any address in less than one hour.

Computer networks also follow the same concept. In computer networking, Subnetting is used to divide a large IP network in smaller IP networks known as subnets.

A default class A, B and C network provides 16777214, 65534, 254 hosts respectively. Having so many hosts in a single network always creates several issues such as broadcast, collision, congestion, etc.

Let's take a simple example. In a company there are four departments; sales, production, development and management. In each department there are 50 users. Company used a private class C IP network. Without any Subnetting, all computers will work in a single large network.

Advantages

- Subnetting allows us to break a single large network in smaller networks. Small networks are easy to manage.
- Subnetting reduces network traffic by allowing only the broadcast traffic which is relevant to the subnet.
- By reducing unnecessary traffic, Subnetting improves overall performance of the network.
- By blocking a subnet' traffic in subnet, Subnetting increases security of the network.
- Subnetting reduces the requirement of IP range.

Disadvantages

- Different subnets need an intermediate device known as router to communicate with each other.
- Since each subnet uses its own network address and broadcast address, more subnets mean more wastage of IP addresses.
- Subnetting adds complexity in network. An experienced network administrator is required to manage the subnetted network.

3.6 CIDR**Q13. What is CIDR ? Discuss with an Example.***Ans :***(Imp.)**

CIDR is an acronym for Classless Inter-Domain Routing. CIDR was developed in the 1990s as a standard scheme for routing network traffic across the internet.

Even if blocks of IP addresses are allocated for the efficient utilization of addresses, still a problem remain and i.e. routing table explosion. Routers that are used in organizations at the edge of a network say a university, require to have an entry for each of their subnets, notifying the router which lines to use to reach that network. For routes that have destinations outside the boundary of the organization, they can use the simple default rule of transmitting the packets on the line toward the ISP that provides the connection to the organization to the rest of the Internet.

Routers that remain in ISPs and backbones in the middle of the Internet do not experience such facility. They must be known about the way to go to reach every network and no simple default rule of transmitting will work. These main routers remain in the default-free zone of the Internet. Nobody really knows number of networks that are connected to the Internet anymore, but probably it is at least a million. Very large table may be made for this. If we view by computer standards, it may not sound large but in practice that routers must perform a lookup in this table to forward each and every packet. At large ISPs, routers may transmit up to millions of packets per second. It needs specialized hardware and fast memory for the processing of packets at these rates, general purpose computer cannot perform this.

Additionally, routing algorithms need each router to exchange information about the addresses by the help of which it can reach with other routers. As the table gets larger, more information is required to be communicated and processed. At least, the processing grows linearly with the table size. Greater communication increases the chances of getting some parts lost, or temporarily, it at least leads to routing instabilities.

The solution to the routing table problem could be going to a deeper hierarchy, like in the telephone network. For instance, having each IP address contain a country, state/province, city, network, and host

field might work. Then, each router would only require knowing the idea to get to each country, the states or provinces in its own country, the cities in its state or province, and the networks in its city. Unluckily, this solution may require significantly more than 32 bits for IP addresses and may use addresses in an inefficient manner.

Luckily, routing table sizes can be reduced by some ways. We can use the same approach as subnetting: routers at various locations can be familiar with a given IP address as belonging to prefixes of different sizes. However, instead of dividing an address block into subnets, here we mix multiple small prefixes into one larger prefix. We called this as route aggregation. The resultant larger prefix is sometimes called a supernet, to contrast with subnets as the division of blocks of addresses.

Using aggregation, prefixes contain IP address of varying sizes. The same IP address that one router view as part of a /22 (a block that contains 210 addresses) may be viewed by another router as part of a larger /20 (containing 212 addresses). It is the task of the router to have the matching prefix information. This design works with subnetting and is known as CIDR (Classless Inter- Domain Routing). The most recent version of it is specified in RFC 4632 (Fuller and Li, 2006). The name focuses the contrast with addresses that program hierarchy with classes.

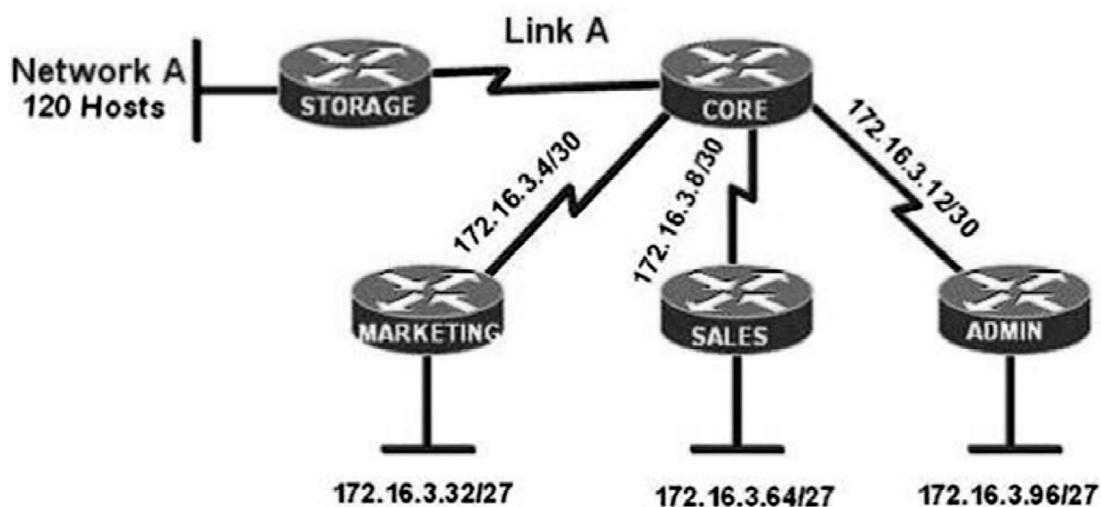
To easily understand CIDR, let us take an example in which a block of 8192 IP addresses is available starting at 194.24.0.0. Let us suppose that Cambridge University needs 2048 addresses and is assigned the addresses 194.24.0. through 194.24.7.255, along with mask 255.255.248.0 and this is a /21 prefix. Another, Oxford University asks for 4096 addresses. Oxford cannot be provided addresses starting at 194.24.8.0. because a block of 4096 addresses must lie on a 4096-byte boundary. Instead, it is provided 194.24.16.0 through 194.24.31.255, along with subnet mask 255.255.240.0. At last, the University of Edinburgh requests for 1024 addresses and is assigned addresses 194.24.8.0 through 194.24.11.255 and mask 255.255.252.0.

10.1.1.0/30

172.16.1.16/28

192.168.1.32/27 etc.

CIDR / VLSM Network addressing topology example



CIDR uses VLSM (Variable Length Subnet Masks) to allocate IP addresses to subnetworks according to need rather than class. VLSM allows for subnets to be further divided or subnetted into even smaller subnets. Simply, VLSM is just subnetting a subnet.

With CIDR, address classes (Class A, B, and C) became meaningless. The network address was no longer determined by the value of the first octet, but assigned prefix length (subnet mask) address space. The number of hosts on a network, could now be assigned a specific prefix depending upon the number of hosts needed for that network.

Propagating CIDR super-nets or VLSM subnets require a classless Routing Protocols – . A classless routing protocol includes the subnet mask along with the network address in the routing update.

CIDR Advantages

With the introduction of CIDR and VLSM, ISPs could now assign one part of a classful network to one customer and different part to another customer. With the introduction of VLSM and CIDR, network administrators had to use additional sub-netting skills.

The table below shows allowed subnet and Hosts IP address for all The Classes

Class A

No. of bits	Subnet Mask	CIDR	No. of Subnets	No. of Hosts	Nets * Hosts
2	255.192.0.0	/10	2	4194302	8388604
3	255.224.0.0	/11	6	2097150	12582900
4	255.240.0.0	/12	14	1048574	14680036
5	255.248.0.0	/13	30	524286	15728580
6	255.252.0.0	/14	62	262142	16252804
7	255.254.0.0	/15	126	131070	16514820
8	255.255.0.0	/16	254	65534	16645636
9	255.255.128.0	/17	510	32766	16710660
10	255.255.192.0	/18	1022	16382	16742404
11	255.255.224.0	/19	2046	8190	16756740
12	255.255.240.0	/20	4094	4094	16760836
13	255.255.248.0	/21	8190	2046	16756740
14	255.255.252.0	/22	16382	1022	16742404
15	255.255.254.0	/23	32766	510	16710660
16	255.255.255.0	/24	65534	254	16645636
17	255.255.255.128	/25	131070	126	16514820
18	255.255.255.192	/26	262142	62	16252804
19	255.255.255.224	/27	524286	30	15728580
20	255.255.255.240	/28	1048574	14	14680036
21	255.255.255.248	/29	2097150	6	12582900
22	255.255.255.252	/30	4194302	2	8388604

Class B

No. of bits	Subnet Mask	CIDR	No. of Subnets	No. of Hosts	Nets * Hosts
2	255.255.192.0	/18	2	16382	32764
3	255.255.224.0	/19	6	8190	49140
4	255.255.240.0	/20	14	4094	57316
5	255.255.248.0	/21	30	2046	61380
6	255.255.252.0	/22	62	1022	63364
7	255.255.254.0	/23	126	510	64260
8	255.255.255.0	/24	254	254	64516
9	255.255.255.128	/25	510	126	64260
10	255.255.255.192	/26	1022	62	63364
11	255.255.255.224	/27	2046	30	61380
12	255.255.255.240	/28	4094	14	57316
13	255.255.255.248	/29	8190	6	49140
14	255.255.255.252	/30	16382	2	32764

Class C

No. of bits	Subnet Mask	CIDR	#No. of Subnets	No. of Hosts	Nets * Hosts
2	255.255.255.192	/26	2	62	124
3	255.255.255.224	/27	6	30	180
4	255.255.255.240	/28	14	14	196
5	255.255.255.248	/29	30	6	180
6	255.255.255.252	/30	62	2	124

Q14. Why we use CIDR (Classless Inter Domain Routing) in sub netting ?

Ans :

Before CIDR technology was developed, internet routers managed network traffic based on the class of IP addresses. In this system, the value of an IP address determines its subnetwork for the purposes of routing.

CIDR is an alternative to traditional IP subnetting.

It organizes IP addresses into subnetworks independent of the value of the addresses themselves. CIDR is also known as supernetting, as it effectively allows multiple subnets to be grouped together for network routing.

CIDR Notation

CIDR specifies an IP address range using a combination of an IP address and its associated network mask. CIDR notation uses the following format:

- xxx.xxx.xxx.xxx/n
- where n is the number of (leftmost) '1' bits in the mask. For example:
- 192.168.12.0/23
- applies the network mask 255.255.254.0 to the 192.168 network, starting at 192.168.12.0. This notation represents the address range 192.168.12.0 - 192.168.13.255. Compared to traditional class-based networking, 192.168.12.0/23 represents an aggregation of the two Class C subnets 192.168.12.0 and 192.168.13.0 each having a subnet mask of 255.255.255.0. In other words:
- $192.168.12.0/23 = 192.168.12.0/24 + 192.168.13.0/24$
- Additionally, CIDR supports internet address allocation and message routing independent of the traditional class of a given IP address range.

For example:

- 10.4.12.0/22 represents the address range 10.4.12.0 - 10.4.15.255 (network mask 255.255.252.0). This allocates the equivalent of four Class C networks within the much larger Class A space.
- You will sometimes see CIDR notation used even for non-CIDR networks. In non-CIDR IP subnetting, however, the value of n is restricted to either 8 (Class A), 16 (Class B) or 24 (Class C).

Examples:

- 10.0.0.0/8
- 172.16.0.0/16
- 192.168.3.0/24

How CIDR Works

CIDR implementations require certain support be embedded within the network routing protocols. When first implemented on the internet, the core routing protocols like BGP (Border Gateway Protocol) and OSPF (Open Shortest Path First) were updated to support CIDR. Obsolete or less popular routing protocols may not support CIDR.

CIDR aggregation requires the network segments involved to be contiguous numerically adjacent in the address space. CIDR cannot, for example, aggregate 192.168.12.0 and 192.168.15.0 into a single route unless the intermediate .13 and .14 address ranges are included.

Internet WAN or backbone routers those that manage traffic between Internet Service Providers all generally support CIDR to achieve the goal of conserving IP address space. Mainstream consumer routers often do not support CIDR, therefore private networks including home networks and even small public networks (LANs) often do not employ it.

Q15. What is Network Address Translation ?

OR

How Network Address Translation Works ?

Ans :

(Imp.)

Network Address Translation (NAT) is the process where a network device, usually a firewall, assigns a public address to a computer (or group of computers) inside a private network. The main use of NAT is to limit the number of public IP addresses an organization or company must use, for both economy and security purposes.

The most common form of network translation involves a large private network using addresses in a private range (10.0.0.0 to 10.255.255.255, 172.16.0.0 to 172.31.255.255, or 192.168.0.0 to 192.168.255.255). The private addressing scheme works well for computers that only have to access resources inside the network, like workstations needing access to file servers and printers. Routers inside the private network can route traffic between private addresses with no trouble.

However, to access resources outside the network, like the Internet, these computers have to have a public address in order for responses to their requests to return to them. This is where NAT comes into play.

Internet requests that require Network Address Translation (NAT) are quite complex but happen so rapidly that the end user rarely knows it has occurred. A workstation inside a network makes a request to a computer on the Internet. Routers within the network recognize that the request is not for a resource inside the network, so they send the request to the firewall. The firewall sees the request from the computer with the internal IP. It then makes the same request to the Internet using its own public address, and returns the response from the Internet resource to the computer inside the private network. From the perspective of the resource on the Internet, it is sending information to the address of the firewall. From the perspective of the workstation, it appears that communication is directly with the site on the Internet. When NAT is used in this way, all users inside the private network access the Internet have the same public IP address when they use the Internet. That means only one public addresses is needed for hundreds or even thousands of users.

Most modern firewalls are stateful - that is, they are able to set up the connection between the internal workstation and the Internet resource. They can keep track of the details of the connection, like ports, packet order, and the IP addresses involved. This is called keeping track of the state of the connection. In this way, they are able to keep track of the session composed of communication between the workstation and the firewall, and the firewall with the Internet. When the session ends, the firewall discards all of the information about the connection.

There are other uses for Network Address Translation (NAT) beyond simply allowing workstations with internal IP addresses to access the Internet. In large networks, some servers may act as Web servers and require access from the Internet. These servers are assigned public IP addresses on the firewall, allowing the public to access the servers only through that IP address. However, as an additional layer of security, the firewall acts as the intermediary between the outside world and the protected internal network. Additional rules can be

added, including which ports can be accessed at that IP address. Using NAT in this way allows network engineers to more efficiently route internal network traffic to the same resources, and allow access to more ports, while restricting access at the firewall. It also allows detailed logging of communications between the network and the outside world.

Additionally, NAT can be used to allow selective access to the outside of the network, too. Workstations or other computers requiring special access outside the network can be assigned specific external IPs using NAT, allowing them to communicate with computers and applications that require a unique public IP address. Again, the firewall acts as the intermediary, and can control the session in both directions, restricting port access and protocols.

NAT is a very important aspect of firewall security. It conserves the number of public addresses used within an organization, and it allows for stricter control of access to resources on both sides of the firewall.

Q16. Describe the mechanism of Network address Translation.

Ans :

NAT is like the receptionist in a large office. Let's say you have left instructions with the receptionist not to forward any calls to you unless you request it. Later on, you call a potential client and leave a message for that client to call you back. You tell the receptionist that you are expecting a call from this client and to put her through.

The client calls the main number to your office, which is the only number the client knows. When the client tells the receptionist that she is looking for you, the receptionist checks a lookup table that matches your name with your extension. The receptionist knows that you requested this call, and therefore forwards the caller to your extension.

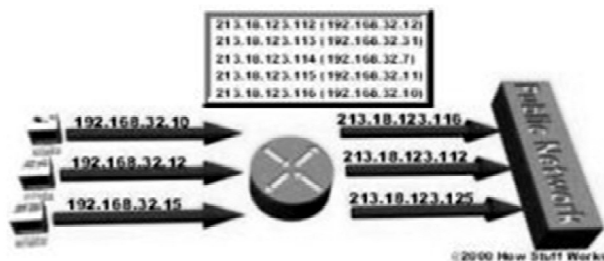
Developed by Cisco, Network Address Translation is used by a device (firewall, router or computer that sits between an internal network and the rest of the world. NAT has many forms and can work in several ways:

Static NAT - Mapping an unregistered IP address to a registered IP address on a one-to-one basis. Particularly useful when a device needs to be accessible from outside the network.



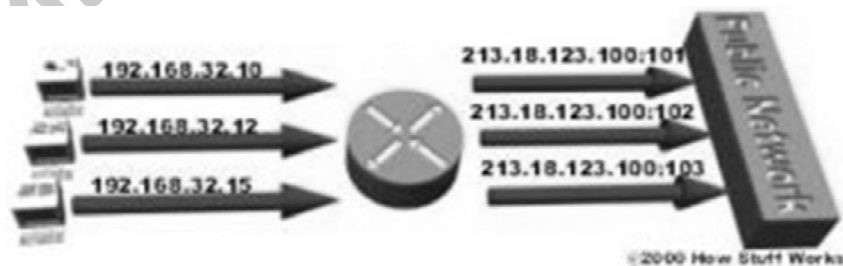
In static NAT, the computer with the IP address of 192.168.32.10 will always translate to 213.18.123.110.

Dynamic NAT - Maps an unregistered IP address to a registered IP address from a group of registered IP addresses.



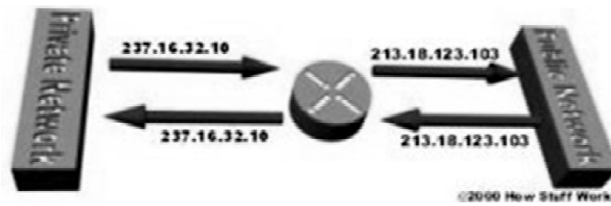
In dynamic NAT, the computer with the IP address 192.168.32.10 will translate to the first available address in the range from 213.18.123.100 to 213.18.123.150.

- **Overloading** - A form of dynamic NAT that maps multiple unregistered IP addresses to a single registered IP address by using different ports. This is known also as PAT (Port Address Translation), single address NAT or port-level multiplexed NAT.



- In overloading, each computer on the private network is translated to the same IP address (213.18.123.100), but with a different port number assignment.

Overlapping - When the IP addresses used on your internal network are registered IP addresses in use on another network, the router must maintain a lookup table of these addresses so that it can intercept them and replace them with registered unique IP addresses. It is important to note that the NAT router must translate the "internal" addresses to registered unique addresses as well as translate the "external" registered addresses to addresses that are unique to the private network. This can be done either through static NAT or by using DNS and implementing dynamic NAT.



- The internal IP range (237.16.32.xx) is also a registered range used by another network. Therefore, the router is translating the addresses to avoid a potential conflict with another network. It will also translate the registered global IP addresses back to the unregistered local IP addresses when information is sent to the internal network.
- The internal network is usually a LAN (Local Area Network), commonly referred to as the stub domain. A stub domain is a LAN that uses IP addresses internally. Most of the network traffic in a stub domain is local, so it doesn't travel outside the internal network. A stub domain can include both registered and unregistered IP addresses. Of course, any computers that use unregistered IP addresses must use Network Address Translation to communicate with the rest of the world.

3.7 INTRODUCTION TO IPv6

Q17. What is IPv6 ?

Ans :

IPv4 produces 4 billion addresses, and the developers think that these addresses are enough, but they were wrong. IPv6 is the next generation of IP addresses. The main difference between IPv4 and IPv6 is the address size of IP addresses. The IPv4 is a 32-bit address, whereas IPv6 is a 128-bit hexadecimal address. IPv6 provides a large address space, and it contains a simple header as compared to IPv4.

It provides transition strategies that convert IPv4 into IPv6, and these strategies are as follows:

- **Dual stacking:** It allows us to have both the versions, i.e., IPv4 and IPv6, on the same device.
- **Tunneling:** In this approach, all the users have IPv6 communicates with an IPv4 network to reach IPv6.
- **Network Address Translation:** The translation allows the communication between the hosts having a different version of IP.

This hexadecimal address contains both numbers and alphabets. Due to the usage of both the numbers and alphabets, IPv6 is capable of producing over 340 undecillion (3.4×10^{38}) addresses.

IPv6 is a 128-bit hexadecimal address made up of 8 sets of 16 bits each, and these 8 sets are separated by a colon. In IPv6, each hexadecimal character represents 4 bits. So, we need to convert 4 bits to a hexadecimal number at a time

Address format

The address format of IPv4:



The address format of IPv6:



The above diagram shows the address format of IPv4 and IPv6. An IPv4 is a 32-bit decimal address. It contains 4 octets or fields separated by 'dot', and each field is 8-bit in size. The number that each field contains should be in the range of 0-255. Whereas an IPv6 is a 128-bit hexadecimal address. It contains 8 fields separated by a colon, and each field is 16-bit in size.

Q18. What are the differences between IPv4 and IPv6 ?

Ans :

(Imp.)

S.No.	Nature	IPv4	IPv6
1.	Address length	IPv4 is a 32-bit address. IPv4 is a numeric address that consists of 4 fields which are separated by dot(.).	IPv6 is a 128-bit address.
2.	Fields	IPv4 is a numeric address that consists of 4 fields which are separated by dot (.).	IPv6 is an alphanumeric address that consists of 8 fields, which are separated by colon.
3.	Classes	IPv4 has 5 different classes of IP address that includes Class A, Class B, Class C, Class D, and Class E.	
4.	Number of IP address	IPv4 has a limited number of IP addresses.	IPv6 has a large number of IP addresses.
5.	VLSM	It supports VLSM (Virtual Length Subnet Mask). Here, VLSM means that Ipv4 converts IP addresses into a subnet of different sizes.	It does not support VLSM
6.	Address configuration	It supports manual and DHCP configuration.	It supports manual, DHCP, auto-configuration, and renumbering.
7.	Address space	It generates 4 billion unique addresses	It generates 340 undecillion unique addresses.

8.	End-to-end connection integrity	In IPv4, end-to-end connection integrity is unachievable.	In the case of IPv6, end-to-end connection integrity is achievable.
9.	Security features	In IPv4, security depends on the application. This IP address is not developed in keeping the security feature in mind.	In IPv6, IPSEC is developed for security purposes.
10.	Address representation	In IPv4, the IP address is represented in decimal	In IPv6, the representation of the IP address in hexadecimal.
11.	Fragmentation	Fragmentation is done by the senders and the forwarding routers.	Fragmentation is done by the senders only.
12.	Packet flow identification	It does not provide any mechanism for packet flow identification.	It uses flow label field in the header for the packet flow identification.
13.	Checksum field	The checksum field is available in IPv4.	The checksum field is not available in IPv6.
14.	Transmission scheme	IPv4 is broadcasting.	On the other hand, IPv6 is multicasting, which provides efficient network operations.
15.	Encryption and Authentication	It does not provide encryption and authentication.	It Provides encryption and authentication.
16.	Number of octets	It consists of 4 octets	It consists of 8 field, and each field contains 2 octets. Therefore, the total number of octets in IPv6 is 16.

3.7.1 ICMP

Q19. Explain the concept of ICMP.

Ans :

The ICMP stands for Internet Control Message Protocol. It is a network layer protocol. It is used for error handling in the network layer, and it is primarily used on network devices such as routers. As different types of errors can exist in the network layer, so ICMP can be used to report these errors and to debug those errors.

For example, some sender wants to send the message to some destination, but the router couldn't send the message to the destination. In this case, the router sends the message to the sender that I could not send the message to that destination.

The IP protocol does not have any error-reporting or error-correcting mechanism, so it uses a message to convey the information. For example, if someone sends the message to the destination, the message is somehow stolen between the sender and the destination. If no one reports the error, then the sender might think that the message has reached the destination. If someone in-between reports the error, then the sender will resend the message very quickly.

The ICMP resides in the IP Layer, as shown in the below diagram.



The ICMP messages are usually divided into two categories:

Category	Type	Message
Error-reporting messages	3	Destination unreachable
	4	Source quench
	11	Time exceeded
	12	Parameter problem
	5	Redirection
Query messages	8 or 0	Echo request or reply
	13 or 14	Timestamp request or reply

(i) **Error-reporting messages**

The error-reporting message means that the router encounters a problem when it processes an IP packet then it reports a message.

(ii) **Query messages**

The query messages are those messages that help the host to get the specific information of another host. For example, suppose there are a client and a server, and the client wants to know whether the server is live or not, then it sends the ICMP message to the server.

3.7.2 IGMP

Q20. What is IGMP in the computer network ?

Ans :

IGMP is acronym for Internet Group Management Protocol. IGMP is a communication protocol used by hosts and adjacent routers for multicasting communication with IP networks and uses the resources efficiently to transmit the message/data packets. Multicast communication can have single or multiple senders and receivers and thus, IGMP can be used in streaming videos, gaming or web conferencing tools. This protocol is used on IPv4 networks and for using this on IPv6, multicasting is managed by Multicast Listener Discovery (MLD). Like other network protocols, IGMP is used on network layer. MLDv1 is almost same in functioning as IGMPv2 and MLDv2 is almost similar to IGMPv3.

Applications

(i) **Streaming**

Multicast routing protocol are used for audio and video streaming over the network i.e., either one-to-many or many-to-many.

(ii) Gaming

Internet group management protocol is often used in simulation games which has multiple users over the network such as online games.

(iii) Web Conferencing tools

Video conferencing is a new method to meet people from your own convenience and IGMP connects to the users for conferencing and transfers the message/data packets efficiently.

Q21. Explain various types of IGMP.

Ans :

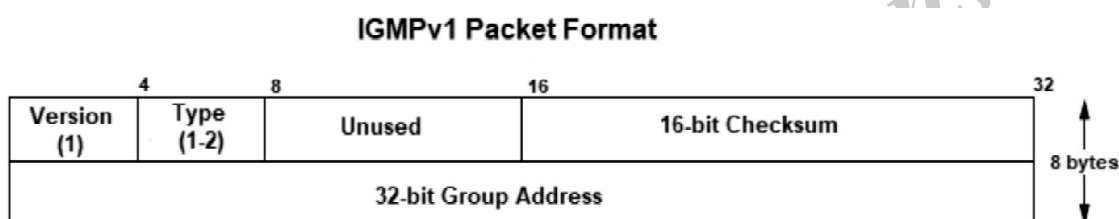
(Imp.)

There are 3 versions of IGMP. These versions are backward compatible. Following are the versions of IGMP:

1. IGMPv1

The version of IGMP communication protocol allows all the supporting hosts to join the multicast groups using membership request and include some basic features. But, host cannot leave the group on their own and have to wait for a timeout to leave the group.

The message packet format in IGMPv1:



➤ **Version –**

Set to 1.

➤ **Type –**

1 for Host Membership Query and Host Membership Report.

➤ **Unused –**

8-bits of zero which are of no use.

➤ **Checksum –**

It is the one's complement of the one's complement of the sum of IGMP message.

➤ **Group Address –**

The group address field is zero when sent and ignored when received in membership query message. In a membership report message, the group address field takes the IP host group address of the group being reported.

2. IGMPv2 :

IGMPv2 is the revised version of IGMPv1 communication protocol. It has added functionality of leaving the multicast group using group membership.

The message packet format in IGMPv2:

IGMPv2 Packet Format

4	8	16	32
Type	Max Response Time	Checksum	
32-bit Group Address			

Type –

0x11 for Membership Query

0x12 for IGMPv1 Membership Report

0x16 for IGMPv2 Membership Report

0x22 for IGMPv3 Membership Report

0x17 for Leave Group

➤ **Max Response Time –**

This field is ignored for message types other than membership query. For membership query type, it is the maximum time allowed before sending a response report. The value is in units of 0.1 seconds.

➤ **Checksum –**

It is the one's complement of the one's complement of the sum of IGMP message.

➤ **Group Address –**

It is set as 0 when sending a general query. Otherwise, multicast address for group-specific or source-specific queries.

3. IGMPv3 :

IGMPv2 was revised to IGMPv3 and added source-specific multicast and membership report aggregation. These reports are sent to 224.0.0.22.

The message packet format in IGMPv3:

IGMPv3 Packet Format

Bit Offset	0-3	4	5-7	8-15	16-31
0	Type = 0x11			Max Response Code	Checksum
32	Group Address				
64	Resv	S	QRV	QQIC	Number of Sources (N)
96	Source Address[1]				
128	Source Address[2]				
	Source Address[N]				

➤ **Max Response Time –**

This field is ignored for message types other than membership query. For membership query type, it is the maximum time allowed before sending a response report. The value is in units of 0.1 seconds.

➤ **Checksum –**

It is the one's complement of the one's complement of the sum of IGMP message.

➤ **Group Address –**

It is set as 0 when sending a general query. Otherwise, multicast address for group-specific or source-specific queries.

➤ **Resv –**

It is set zero of sent and ignored when received.

➤ **S flag –**

It represents Suppress Router-side Processing flag. When the flag is set, it indicates to suppress the timer updates that multicast routers perform upon receiving any query.

➤ **QRV –**

It represents Querier's Robustness Variable. Routers keep on retrieving the QRV value from the most recently received query as their own value until the most recently received QRV is zero.

➤ **QQIC –**

It represents Querier's Query Interval Code.

➤ **Number of sources –**

It represents the number of source addresses present in the query. For general query or group-specific query, this field is zero and for group-and-source-specific query, this field is non-zero.

➤ **Source Address[i] –**

It represents the IP unicast address for N fields.

3.7.3 OSPF**Q22. Discuss about OSPF in detail.**

Ans :

(Imp.)

The routers in an internet are accountable for receiving and transmitting packets through the organized set of networks. Each router does its routing decision on the basis of knowledge of the topology and traffic/delay situations of the internet. A fixed routing scheme is possible in a simple internet. In case of complex internets, a level of dynamic collaboration is required among the routers. Particularly, the router has to stay away from parts of the network that have been unsuccessful and should avoid parts of the network that are congested. For making such type of dynamic routing decisions, routers exchange routing details by using a special routing protocol. Information is required about the condition of the internet by the help of which networks can be reached by which path and the delay properties of different paths. Mainly these two concepts are considered main for the routing functions:

- Routing information: It is the information about the topology and delays of the internet
- Routing algorithm: It is the algorithm used to make a routing decision for a particular datagram on the basis of current routing information

Interior Routing Protocol: OSPF

The OSPF (Open Shortest Path First) protocol comes under the category of IP Routing protocols, and is also an Interior Gateway Protocol (IGP) for the Internet, that distribute IP routing details throughout a single Autonomous System (AS) in an IP network.

Open-Shortest-Path-First (OSPF) is the most widely used interior gateway protocol routing protocol on the world because it is a public (non-proprietary) routing protocol while its biggest rival, EIGRP, is a Cisco proprietary protocol so other vendors can't use it. OSPF is a complex link-state routing protocol. Link-state routing protocols generate routing updates only when a change occurs

in the network topology. When a link changes state, the device that detected the change creates a link-state advertisement (LSA) concerning that link and sends to all neighboring devices using a special multicast address. Each routing device takes a copy of the LSA, updates its link-state database (LSDB), and forwards the LSA to all neighboring devices.

The OSPF protocol can be considered as a link-state routing protocol, it means that the routers exchange topology information with the neighbor router. The information about the topology is flooded throughout the AS, so that every router that lies inside the AS has a complete view of the topology of the AS. This view can be then used to compute end-to-end paths through the AS, usually utilizing a variation of the Dijkstra algorithm. So, in case of link-state routing protocol, the next hop address to which data is transmitted is determined by choosing the best end-to-end path to the final destination.

The main benefit of a link state routing protocol such as OSPF is that the full knowledge of topology let routers to compute paths that satisfy certain criteria. It can be beneficial for traffic engineering purposes, where routes can be forced to meet certain quality of service needs. The main disadvantage of a link state routing protocol is that it has no proper ability to scale well in case of adding more routers to the routing domain. The size and frequency of the topology updates increase with the increasing number of routers, and the duration it takes to compute end-to-end routes also increases with it. This lack of scalability implies that a link state routing protocol is inappropriate for routing across the large internet. This is also the reason why IGP's only route traffic within a single AS.

Each router in the OSPF provides details about its local state and the details may be usable interfaces and reachable neighbors and the cost of using each interface. These details are provided to other routers by the help of a Link State Advertisement (LSA) message. Each router uses the received messages to make up the same database that gives information about the topology of the AS.

Using this database, each router computes its own routing table using a Shortest Path First (SPF) or Dijkstra algorithm. This routing table includes all the destinations the routing protocol is known about, related to a next hop IP address and outgoing interface.

- The protocol recalculates routes as the network topology changes by the help of Dijkstra algorithm and reduces the routing protocol traffic that it develops.
- It gives support for multiple paths of equal cost.
- It provides a multi-level hierarchy also called "area routing," so that details about the topology within a defined area of the AS is protected and hidden from routers that lie outside this area. This allows a supplementary degree of routing security and a minimization in routing protocol traffic.
- All protocol exchanges can be used only by the authorized routers that mean only trusted routers can be involved in the routing exchanges for the AS.
- OSPF version 2 (OSPFv2) is compatible with IPv4. OSPFv3 has been modified for compatibility with IPv6's. However, this is not the only variation in OSPFv2 and OSPFv3. Other modifications in OSPFv3, as defined in RFC 2740, include
- Protocol processing based on per-link net on per-subnet
- Addition of flooding scope, which may be link-local, area or AS-wide
- Erase of opaque LSAs
- Support for multiple instances of OSPF per link

Different packet and LSA format modifications

The OSPF protocol (RFC 2328) is presently widely used as the interior router protocol in TCP/IP networks. OSPF calculates a route by the help of the internet that incurs the minimum cost based on

a user-manageable metric of cost. The user can arrange the cost to state a function of delay, data rate, dollar cost, or other different factors. OSPF has the ability to equalize loads over multiple equal-cost paths.

Each router maintains a database that reveals the known topology of the autonomous system of which it is a part. The topology is articulated using a directed graph. The graph contains the following:

Vertices, or nodes, of two types:

1. Router
2. Network, which is also of two types

Transit, these are the networks that can route data that is neither created nor finished on an end system connected to this network

Stub, these are networks that are not considered as transit network

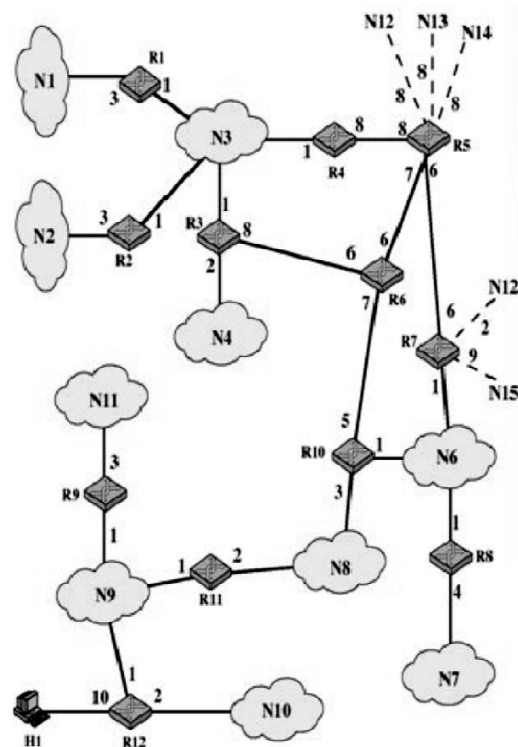
Edges of two types

1. Graph edges that join two router vertices when the particular routers are connected to each other by a direct point-to-point link.
2. Graph edges that join a router vertex to a network vertex when the router is directly connected to the network.

The mapping is straightforward

- Two routers connected by a point-to-point link are shown in the graph as being directly joined by a pair of edges, one in each direction (such as routers 6 and 10).
- When many routers are connected to a network (for instance, a LAN or packet switching network), the directed graph reveals all routers bi-directionally joined to the network vertex such as routers 1, 2, 3, and 4 all join to network 3).
- If a single router is joined to a network, the network will be seen in the graph as an end connection (such as network 7).
- An end system, also called as a host, can be directly joined to a router, in which case it is represented in the corresponding graph (such as host 1).
- If a router is joined to other autonomous systems, then the route cost to each network in the other system must be gained by some

exterior router protocol (ERP). Each such network is shown on the graph by a stub and an edge to the router with the known path cost (such as networks 12 through 15).

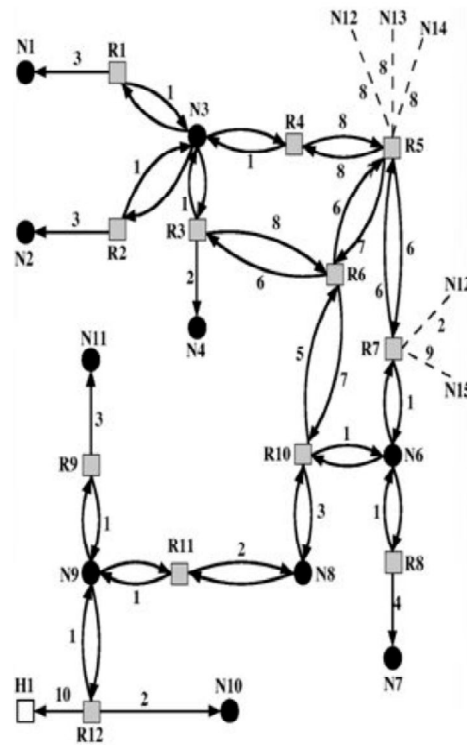


A Sample Autonomous System

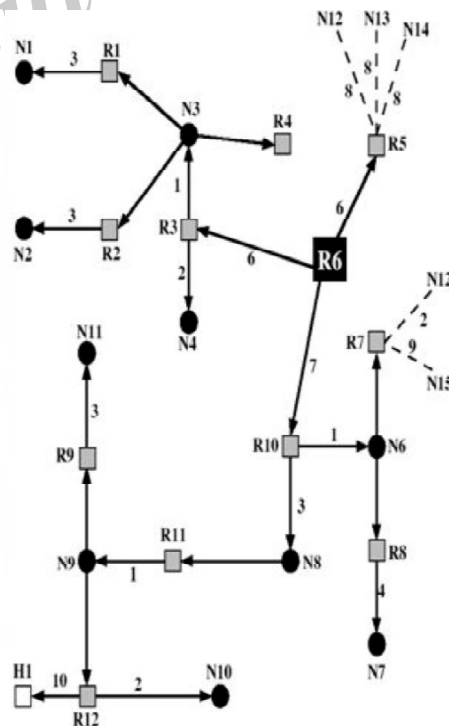
A cost is related to the output part of each router interface. The system administrator manages this cost. Arcs on the graph are labeled with the cost of the router that corresponds to the output interface. Arcs that have no labeled cost have a cost of 0.

It should be noted that arcs leading from networks to routers always have a cost of 0. A database that corresponds to the directed graph is managed by each router. It is pieced jointly from link state messages from other routers on the internet. By the help of Dijkstra Algorithm, a router computes the minimum-cost path to all destination networks. The result for router 6 is shown as a tree, being R6 as the root of the tree. The tree provides the whole route to any destination network or host. However, only the next hop to the destination is used in the routing forward process. The resulting routing table for router 6 is shown in Table 1. The

table contains entries for routers advertising external routes (i.e. routers 5 and 7). For external networks whose detail is known, entries are also provided.



Directed graph of autonomous system



SPF tree for router R6

Destination	Next Hop	Distance
N1	R3	10
N2	R3	10
N3	R3	7
N4	R3	8
N6	R10	8
N7	R10	12
N8	R10	10
N9	R10	11
N10	R10	13
N11	R10	14
H1	R10	21
R5	R5	6
R7	R10	8
N12	R10	10
N13	R5	14
N14	R5	14
N15	R10	17

Features and advantage of OSPF

- It supports both IPv4 and IPv6 routed protocols.
- It supports load balancing with equal cost routes for same destination.
- Since it is based on open standards, it will run on most routers.
- It provides a loop free topology using SPF algorithm.
- It is a classless protocol.
- It supports VLSM and route summarization.
- It supports unlimited hop counts.
- It scales enterprise size network easily with area concept.
- It supports trigger updates for fast convergence.
- Just like other routing protocols, OSPF also has its negatives.

Disadvantage of OSPF

- It requires extra CPU process to run SPF algorithm.
- It requires more RAM to store adjacency topology.
- It is more complex to setup and hard to troubleshoot.

3.7.4 Border Gateway Protocol (BGP)

Q23. Explain the concept of Border Gateway Protocol (BGP).

Ans :

(Imp.)

Definition

Border Gateway Protocol (BGP) is a routing protocol used to transfer data and information between different host gateways, the Internet or autonomous systems. BGP is a Path Vector Protocol (PVP), which maintains paths to different hosts, networks and gateway routers and determines the routing decision based on that. It does not use Interior Gateway Protocol (IGP) metrics for routing decisions, but only decides the route based on path, network policies and rule sets.

Sometimes, BGP is described as a reachability protocol rather than a routing protocol.

BGP roles include

- Because it is a PVP, BGP communicates the entire autonomous system/network path topology to other networks
- Maintains its routing table with topologies of all externally connected networks
- Supports classless interdomain routing (CIDR), which allocates Internet Protocol (IP) addresses to connected Internet devices

When used to facilitate communication between different autonomous systems, BGP is referred to as External BGP (EBGP). When used at host networks/autonomous systems, BGP is referred to as Internal BGP (IBGP).

We really want to show you why we need BGP first but it is very difficult to explain without understanding a bit about BGP. So we will learn some basic knowledge about BGP first.

First we need to understand about the difference between Interior Gateway Protocol and Exterior Gateway Protocol. The difference between them is shown below:



➤ Interior Gateway Protocol (IGP)

A routing protocol operating within an Autonomous System (AS) like OSPF, EIGRP... Usually routers running IGP are under the same administration (of a company, corporation, individual)

➤ Exterior Gateway Protocol (EGP)

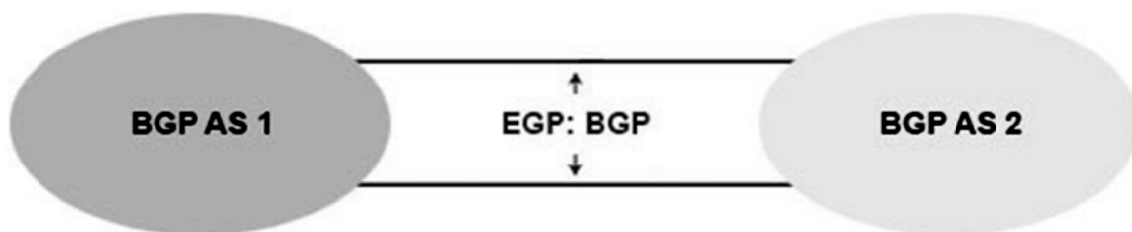
A routing protocol operating between different AS. BGP is the only EGP used nowadays

In the topology above R1, R2 and R3 should run an IGP to communicate with each other because they are in the same AS. But to connect with other routers in another AS (like a different ISP), R1 and R3 must use an EGP.

With BGP, the term *autonomous system* (AS) refers to a network that operates separately from other networks and usually operates within a single administrative domain. Each AS is represented by an AS number. It is similar to EIGRP AS in this aspect. BGP is used mainly by the Internet Service Provider (ISP) all over the world. Each ISP usually has one BGP AS number (some very big ISP may have a few AS numbers). BGP AS numbers can be between 1 to 65,535.

In the topology above R1 and R3 are operating in BGP AS 1. If an AS connects to the public Internet using an EGP, then it must be assigned a unique AS number which is managed by the Internet Assigned Numbers Authority (IANA). IANA manages the AS numbers from 1 to 64,512 for public use (similar to public IP addresses) while 64,512 to 65,535 numbers are reserved for private use (similar to private IP addresses).

If we don't want to show the routers inside each AS we can simply ignore them:



In fact, the Internet that we are going “online” every day is a collection of inter connected autonomous systems and BGP is running to provide routing between them.

Other BGP terms that you should learn are listed below:

- **BGP speaker:** a router running BGP
- **BGP peer or BGP neighbor**

Any two routers that have formed a TCP connection to exchange BGP routing information (as BGP runs over TCP on port 179, not UDP)

Prefix

Maybe you learned the word “subnet”. In BGP world, it is usually called “prefix” because BGP usually does not advertise small subnets. It advertises blocks of large subnets so “prefix” is often used instead

- **Internal BGP (iBGP)**

refers to the BGP neighbor relationship within the same AS. The iBGP neighbor does not have to be directly connected

- **External BGP (eBGP)**

refers to the BGP neighbor relationship between two peers belongs to different AS. It is recommended that eBGP should be directly connected. Never run an IGP between eBGP peers.

In the below topology suppose all routers are running BGP then R1 is considered internal BGP to R2 and R3 (as they are running same AS 1) but is external BGP to R4. R5 is internal to R4 and R6 but external to R3.



Q24. Explain different types of Connections to ISP.

Ans :

BGP is often used to connect to the ISP so we list here all the type of connection to the ISP.

Single homed

Your company may connect to ISP in several ways. The most popular and simple way is single homed with a single link between the company and the ISP. With this design, only one possible next-hop router exists for all routes to the Internet.



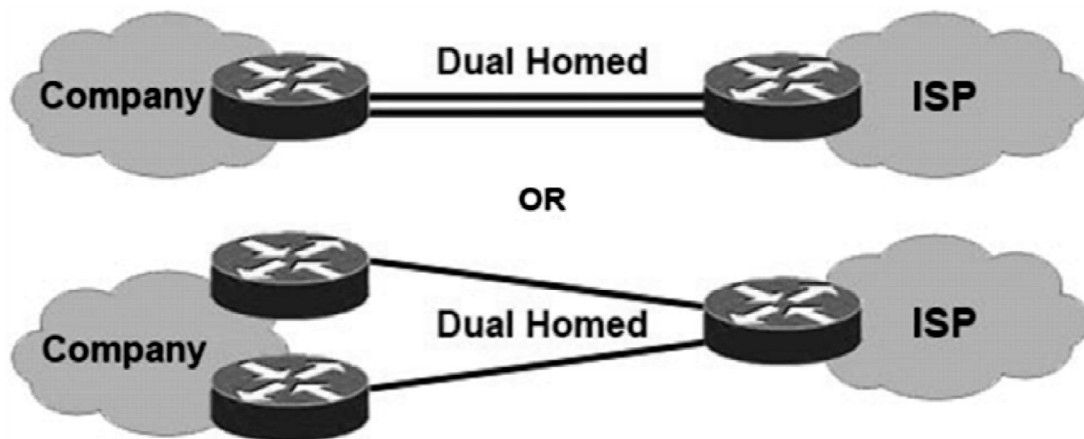
A big disadvantage of this design is when the link fails or either of the routers fails, the connection to the Internet fails as well. But of course, this design saves money comparing to multiple connections to the Internet designs and in fact it is the only reason for small company to accept this design.

With this design we don't need BGP in fact, all things we need are:

- A default route from the company to the ISP
- A static route from the ISP to the company's public address range

Dual homed

The next design is called "dual homed", in which the "dual" word refers to the designs with two links to the same router.

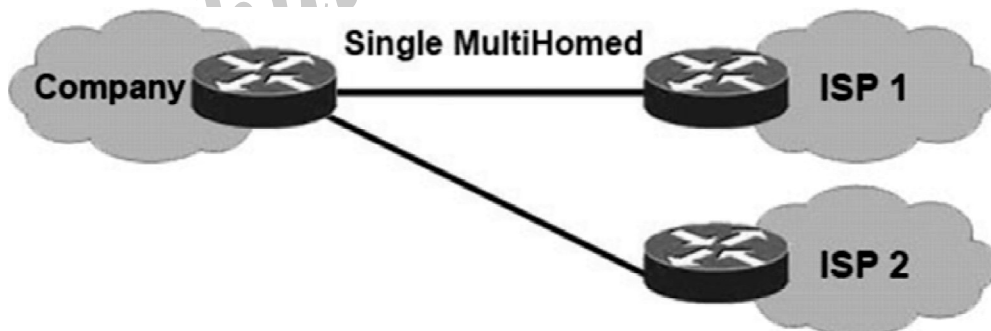


In this design we can use BGP to share the traffic between two routers of the company with our specific ratio (load balancing) or fail over. Of course this design is better in redundancy than the first one but it still has a “single point of failure” at the ISP router.

Single Multi-homed

The next design is called “single multi-homed” refers to:

- Having connections to multiple ISPs from one router at the company
- Single link per ISP.

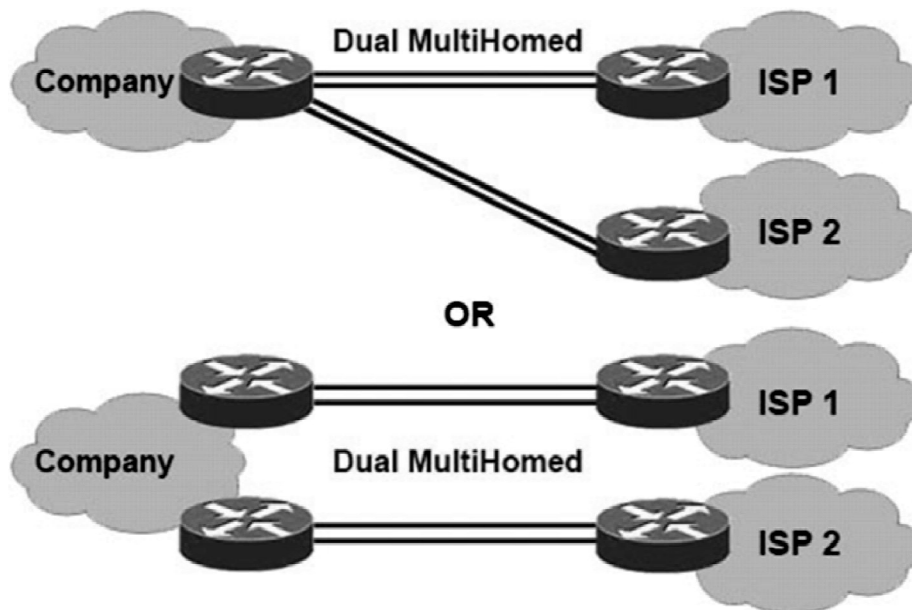


This design is good if we want to separate important traffic to a specific ISP while still has the other ISP as the fail over path.

Dual Multi-homed

And the last design is called “dual multihomed” refers to:

- + Multiple links per ISP
- + Multiple links to Company



If your company has a strong budget then Dual MultiHomed design is ideal to make sure your connection to outside is always up.

UNIT IV

TRANSPORT LAYER :

Services of transport layer, Multiplexing. Transmission Control Protocol (TCP)
Congestion Control, timer management, Quality of services (QOS) and User
Datagram Protocol (UDP)

4.1 TRANSPORT LAYER

4.1.1 Services of transport layer

Q1. Define transport layer. Explain the services of transport layer.

Ans :

(Imp.)

Meaning

The transport layer is located between the application layer and the network layer. It provides a process-to-process communication between two application layers, one at the local host and the other at the remote host. Communication is provided using a logical connection, which means that the two application layers, which can be located in different parts of the globe, assume that there is an imaginary direct connection through which they can send and receive messages.

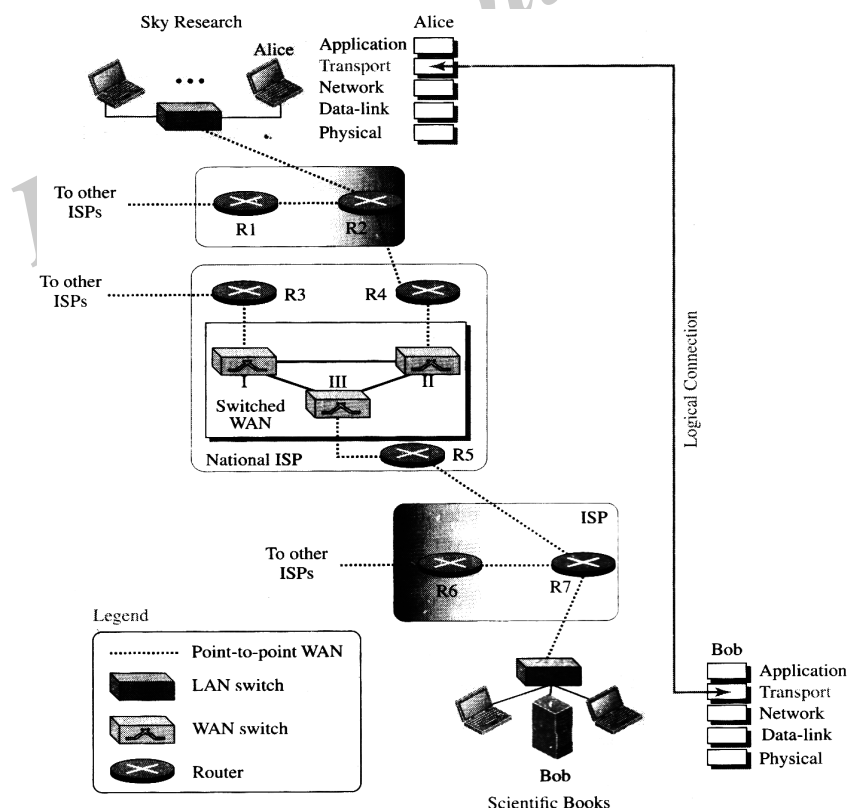


Fig.: Logical connection at the transport layer

The transport layer is located between the network layer and the application layer. The transport layer is responsible for providing services to the application layer; it receives services from the network layer. In this section, we discuss the services that can be provided by the transport layer.

(i) Process-to-Process Communication

The first duty of a transport-layer protocol is to provide process-to-process communication. A process is an application-layer entity (running program) that uses the services of the transport layer. Before we discuss how process-to-process communication can be accomplished, we need to understand the difference between host-to-host communication and process-to-process communication.

(ii) Addressing: Port Numbers

Although there are a few ways to achieve process-to-process communication, the most common 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.

However, operating systems today support both multiuser and multiprogramming environments. A remote computer can run several server programs at the same time, just as several local computers can run one or more client programs at the same time. For communication, we must define the local host, local process, remote host, and remote process. The local host and the remote host are defined using IP addresses discussed in. To define the processes, we need second identifiers, called port numbers. In the TCP/IP protocol suite, the port numbers are integers between 0 and 65,535 (16 bits).

(iii) CANN Ranges

ICANN has divided the port numbers into three ranges: well-known, registered, and dynamic (or private), as shown in fig.

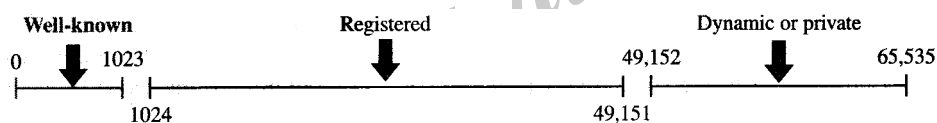


Fig.: CANN Ranges

- **Well-known ports:** The ports ranging from 0 to 1023 are assigned and controlled by ICANN. These are the well-known ports.
- **Registered ports:** The ports ranging from 1024 to 49,151 are not assigned or controlled by ICANN. They can only be registered with ICANN to prevent duplication.
- **Dynamic ports:** The ports ranging from 49,152 to 65,535 are neither controlled nor registered. They can be used as temporary or private port numbers.

(iv) Socket Addresses

A transport-layer protocol in the TCP suite needs both the 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.

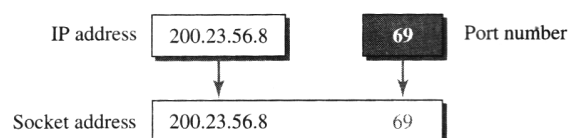


Fig.: Socket address

(v) Encapsulation and Decapsulation

To send a message from one process to another, the transport-layer protocol encapsulates and decapsulates messages. Encapsulation happens at the sender site. When a process has a message to send, it passes the message to the transport layer along with a pair of socket addresses and some other pieces of information, which depend on the transport-layer protocol. The transport layer receives the data and adds the transport-layer header. The packets at the transport layer in the Internet are called user datagrams, segments, or packets, depending on what transport-layer protocol we use. In general discussion, we refer to transport-layer payloads as packets.

Decapsulation happens at the receiver site. When the message arrives at the destination transport layer, the header is dropped and the transport layer delivers the message to the process running at the application layer. The sender socket address is passed to the process in case it needs to respond to the message received.

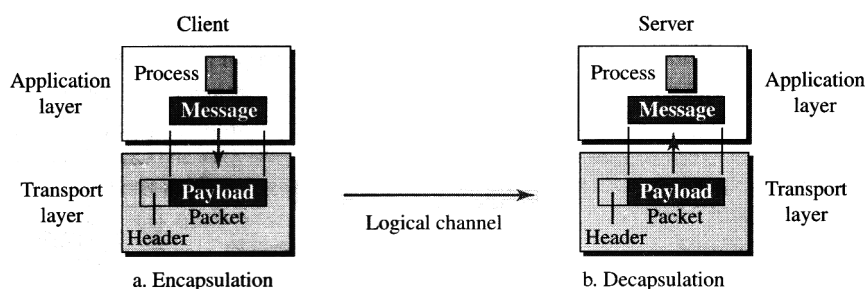


Fig.: Encapsulation and decapsulation

4.2 MULTIPLEXING

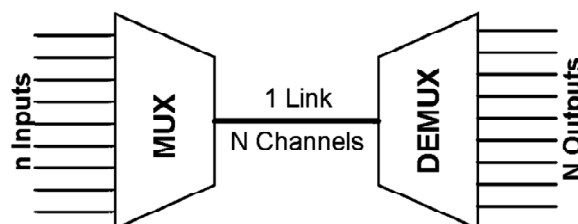
Q2. What is multiplexing? Explain its methods.

Ans :

(Imp.)

Meaning

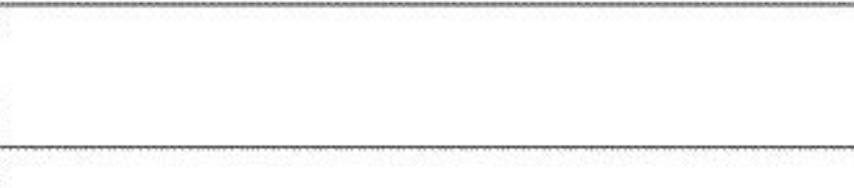
Channel multiplexing is the process of splitting or sharing the capacity of a high speed channel/telecommunication link to form multiple low capacity/low speed sub-channels. Each such sub-channel can then be used by multiple end nodes as dedicated links. Multiplexing can usually be done in different domains like time, frequency and space (and even combinations of these).



For computer communication, though multiplexing techniques like TDM, FDM were initially used mainly in backbone links connecting multiple data exchanges, later they have percolated widely into the access/last mile links too, including inside home networks.

Advantages

If no multiplexing is used between the users at two different sites that are distance apart, then separate communication lines would be required.



Separate Channels/links between the devices

Receiving Locations

Division Multiplexing (TDM)

a high speed data channel/link is made to carry data of multiple users in a round robin fashion. TDM is similar in concept to multitasking. The processor carries out multiple tasks simultaneously. In multitasking, the processor executes one task at any instant of time and keeps shuttling between tasks. In TDM, a high speed data channel in which it executes, each task thinks as though it is the only task being executed.

Division Multiplexing (TDM)

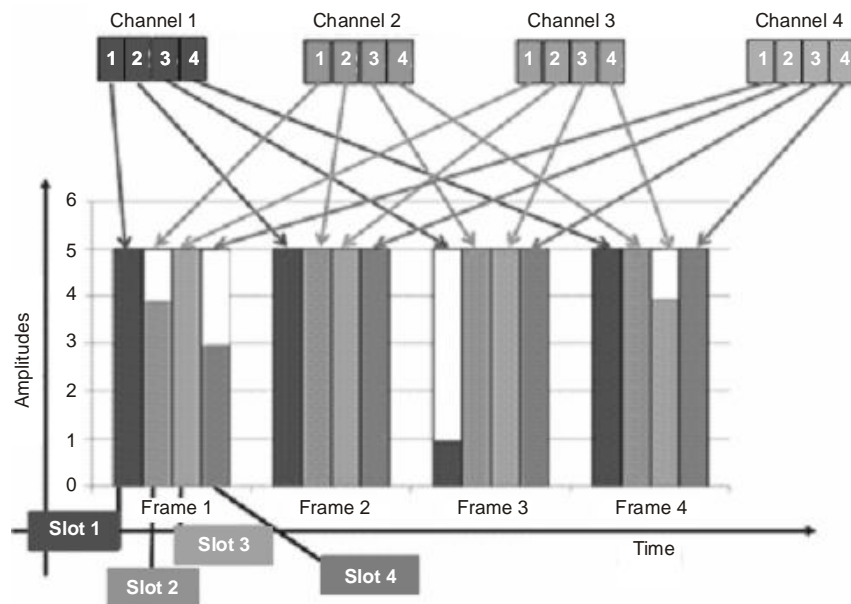
a high speed data channel/link is made to carry data of multiple users in a round robin fashion. TDM is similar in concept to multitasking. The processor carries out multiple tasks simultaneously. In multitasking, the processor executes one task at any instant of time and keeps shuttling between tasks. In TDM, a high speed data channel in which it executes, each task thinks as though it is the only task being executed.

in TDM, data of each connection is segmented into smaller units, so that they fit inside mini link transmits these small units of data from multiple connections in a round robin fashion, allotting a mini time slot for each user, in the time domain.

the basic repeating unit is a frame. A TDM frame consists of a fixed number of time slots. Inside a frame carries data belonging to a specific end node/connection. Thus multiple channels/links are created inside a single channel. It is also possible to give multiple slots within same user, thereby having the provision of having different capacity sub-channels within

g that there are “n” end users, each requiring a link with a capacity of X Kbps, then to multiplex these each end users on a channel, the channel’s capacity needs to be atleast equal to nX Kbps.

are given below illustrates a sample TDM scheme with 4 users being served in a round robin time domain.



An example TDM frame with 4 time slots serving 4 different users

In the example given in the figure, the TDM main channel serves a total of four users and hence creates four sub-channels. Each user's data is carried in a specific slot inside each frame. For e.g. Channel-1's (User 1) data is always carried in the first slot of each frame.

The basic principle of any TDM based protocol remains the same as described above, though there are multiple variants, based on

- The transmission speed
- Number of frames generated per second
- The number of time slots within each frame
- The frame structure etc.

TDM is typically used in WAN digital transmission links, in both trunk and access networks. ISDN (Integrated Services Digital Network) is an example of a protocol using TDM at the access network, to connect home users to their nearest ISP, using the local loop (telephone link). In ISDN, there are a total of 3 sub-channels, with two of them known as B-Channels (Bearer Channels), each with a capacity of 64Kbps being used to carry data and the third known as D-Channel with a capacity of 16Kbps being used to carry signalling information.

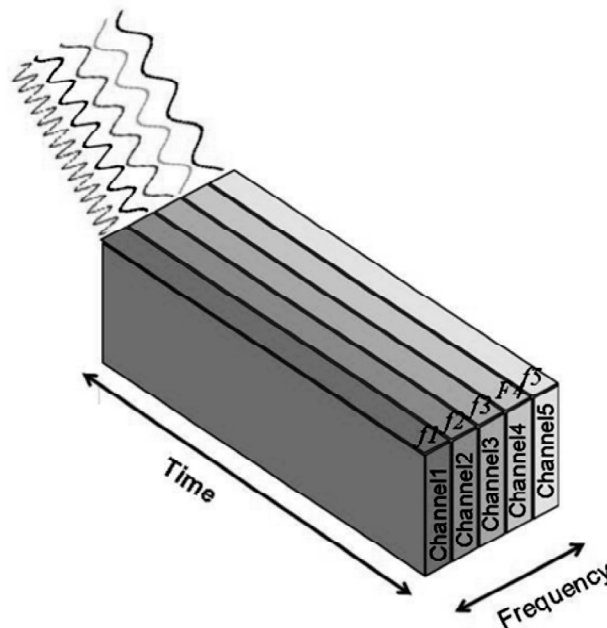
Standard T1/E1 serial links are classical examples of TDM based protocols and are used as trunk links between data exchanges. While T1 supports an aggregate rate of 1.54 Mbps with support for 24 sub-channels, each with a capacity of 64Kbps, E1 supports an aggregate rate of 2.08 Mbps with support for 32 sub-channels, each with a capacity of 64Kbps. TDM links with higher capacity include T2, T3 and SONET Optical links.

Time Division Duplex (TDD) is a form of TDM, where within the same TDM frame, some slots are used for uplink direction (end nodes to network) and some slots are used for downlink direction (network to end nodes), thereby enabling full duplex communication using the same TDM link.

2. Frequency Division Multiplexing (FDM)

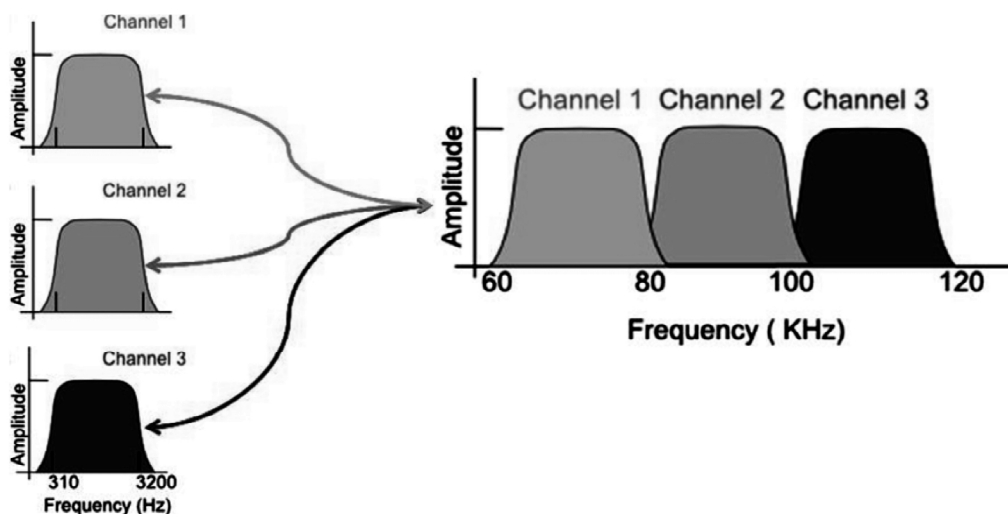
In FDM, the spectrum (frequency range) of a high capacity link is divided into different non-overlapping intervals/carriers. Data of different end nodes are then modulated using these different carriers, so that the resultant signal of each end node occupies a different region in the frequency domain. Between each adjacent carrier, a small guard band is left unused, so as not to cause interference between closely separated carriers.

In FDM, at any instant of time, we would have electromagnetic signals corresponding to each node/sub-channel, unlike in TDM, where at any instant of time, the channel would only have electromagnetic signal belonging to one end node/sub-channel. This is shown in the diagram given below.



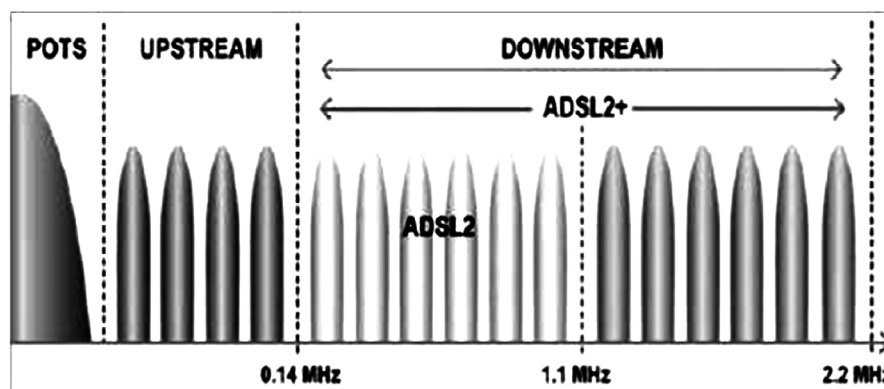
An example FDM with 6 different frequency carriers coexisting simultaneously in the time domain

Traditional FM Radio and Broadcast TV are classical examples of applications using FDM, where data belonging to each radio station/ TV channel is modulated over a different carrier, as shown in the diagram given below.



In computer communication, the concept of basic FDM and variants of FDM are widely used both in LAN and WAN environments. DSL and cable modem links are typical examples of physical layer protocols using FDM for achieving high data rates. In DSL, which also uses the standard telephone last mile local loop line, multiple sub-carriers, each with a band width of 4KHz. are used to carry users data. The base band region from 0 to 4KHz is left for basic POTS voice calls. Above this, some number of sub-carriers are allotted for upstream traffic and a higher number of sub-carriers are allotted for downstream traffic. Similarly, cable modem has a separate frequency band for upstream traffic and a range of sub-carriers for downstream traffic.

An example diagram showing the sub-carrier spectrum allocation for POTS, DSL upstream and downstream directions are given in the diagram below :



FDM being used in ADSL, with different frequency sub-carriers for POTS, ADSL upstream and ADSL downstream.

In DSL, to achieve high data rates, a line coding technique like QAM is used on top of each sub-carrier. Thus both FDM and line coding techniques are combined at the physical layer to achieve high broadband data rates.

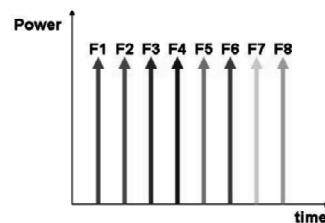
FDM is also used in some variants of Fast Ethernet (100 Mbps) and Gigabit Ethernet (1000 Mbps) LAN protocols, where multiple carriers are used to achieve the overall data rate supported by the underlying physical layer.

Variants of FDM

- Wavelength Division Multiplexing (WDM) and DWDM (Dense-WDM) used in optical Networks, are based on principles similar to FDM, except that their carriers are based on different wavelengths instead of different frequencies
- Frequency Division Duplexing (FDD) is a form of FDM, where some set of frequencies/carriers are used for carrying uplink direction traffic and some other set of frequencies are used for carrying downlink traffic, thereby enabling full duplex communication using FDM.
- Spread Spectrum techniques are variants of FDM, where the data is carried or spread over a wide range of frequency spectrum. In normal FDM, a single carrier is used to carry data corresponding to an end node. But in Spread spectrum techniques, multiple carriers are used to carry data corresponding to an end node, with each carrier carrying a small piece of data. FHSS (Frequency Hopping Spread Spectrum), DSS (Direct Sequence Spread Spectrum) and OFDM (Orthogonal Frequency Division Multiplexing) are different types of spread spectrum techniques.

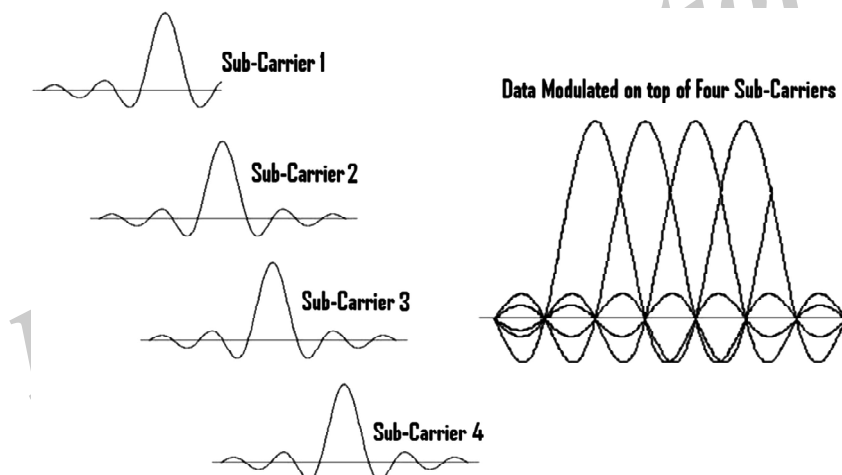
- In FHSS, the frequency of the carrier varies from instant to instant, whereas in DSS, data is split into smaller units and simultaneously carried by multiple carriers, as shown in the diagram given below:

In frequency hopping, the carrier frequency follows a pattern which is known only to the sender and receiver



A FHSS where 8 different carriers are used, with data hopping on top of different carriers in a pre-determined order

- **CDMA** (Code Division Multiple Access) is a form of DSS, where a codeword is combined with data to spread the signal over a wide range of spectrum.
- **OFDM** is a form of DSS that is widely used in Wireless LAN protocols (802.11 a/g), wherein a set of carriers that are orthogonal (do not interfere with each other) are used to carry the data signal, as shown in the diagram below.



An OFDM scheme where 4 orthogonal frequency sub-carriers are used to spread the information over a wider spectrum.

Orthogonal Frequency Division Multiplexing (OFDM) is used for both wired and wireless networks.

3. Space Division Multiplexing (SDM)

In SDM, the same set of frequencies or same set of TDM signals are used in two different places that are geographically wide apart in space, so that one does not interfere with the other. Cellular communication, where the same set of carrier frequencies are reused (frequency reuse) in cells that are not close to one another is a classical example of SDM. Another example of SDM is the FM radio broadcast, where the same set of carrier frequencies are used in different cities that are geographically apart. These are examples of techniques where SDM and FDM are combined.

Combination of FDM, TDM and SDM

GSM (Global System for Mobile Communication) protocol combines both TDM and FDM, to achieve full duplex wireless communication between the mobile handsets and the base stations. While one set of frequencies are used as base carriers from mobile handsets to base station, another set of non-overlapping frequencies are used as base carriers from base stations to mobile handsets, thereby making use of FDM principles. Within each carrier, GSM uses TDM, to carry voice and data belonging to multiple mobile users simultaneously, each in different time slots. Additionally the same set of carrier frequencies and TDM schemes are reused beyond a certain minimum distance, thereby making use of SDM. Thus GSM is an example protocol that uses FDM, TDM and SDM

4.3 TRANSMISSION CONTROL PROTOCOL

Q3. Define Transmission Control Protocol. State the various services of TCP.

Ans :

(Imp.)

Meaning

Transmission Control Protocol (TCP) is a connection-oriented, reliable protocol. TCP explicitly defines connection establishment, data transfer, and connection tear-down phases to provide a connection-oriented service. TCP uses a combination of GBN and SR protocols to provide reliability. To achieve this goal, TCP uses checksum (for error detection), retransmission of lost or corrupted packets, cumulative and selective acknowledgments, and timers. In this section, we first discuss the services provided by TCP; we then discuss the TCP features in more detail. TCP is the most common transport-layer protocol in the Internet.

Services

(i) Process-to-Process Communication

As with UDP, TCP provides process-to-process communication using port numbers.

(ii) Stream Delivery Service

TCP, unlike UDP, is a stream-oriented protocol. In UDP, a process sends messages with predefined boundaries to UDP for delivery. UDP adds its own header to each of these messages and delivers it to IP for transmission. Each message from the process is called a user datagram, and becomes, eventually, one IP datagram. Neither IP nor UDP recognizes any relationship between the datagrams.

TCP, on the other hand, allows the sending process to deliver data as a stream of bytes and allows the receiving process to obtain data as a stream of bytes. TCP creates an environment in which the two processes seem to be connected by an imaginary "tube" that carries their bytes across the Internet.

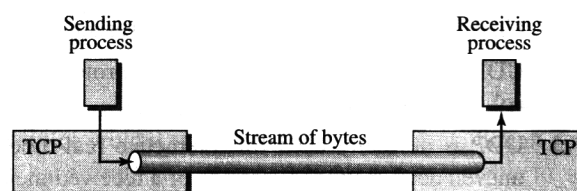


Fig.: Stream delivery

(iii) Sending and Receiving Buffers

Because the sending and the receiving processes may not necessarily write or read data at the same rate, TCP needs buffers for storage. There are two buffers, the sending buffer and the receiving buffer,

one for each direction. We will see later that these buffers are also necessary for flow- and error-control mechanisms used by TCP. One way to implement a buffer is to use a circular array of 1-byte locations as shown in fig., 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.

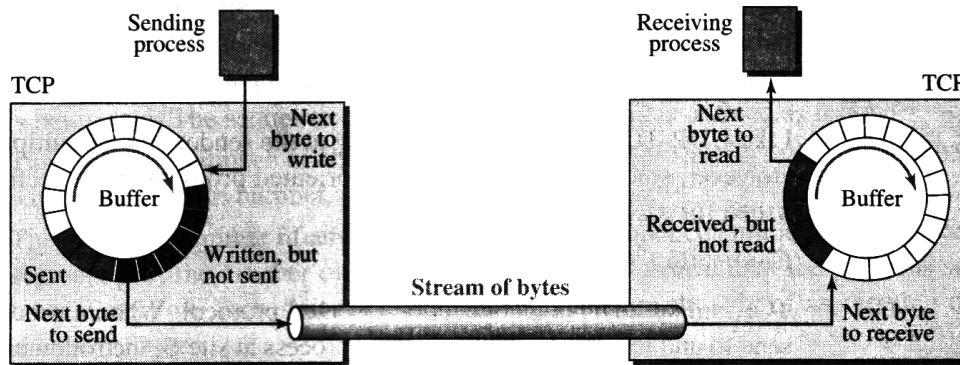


Fig.: Sending and receiving buffers

(iv) Segments

Although buffering handles the disparity between the speed of the producing and consuming processes, we need one more step before we can send data. The network 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. TCP adds a header to each segment (for control purposes) and delivers the segment to the network layer for transmission. The segments are encapsulated in an IP datagram and transmitted. This entire operation is transparent to the receiving process. Later we will see that segments may be received out of order, lost or corrupted, and resent. All of these are handled by the TCP receiver with the receiving application process unaware of TCP's activities.

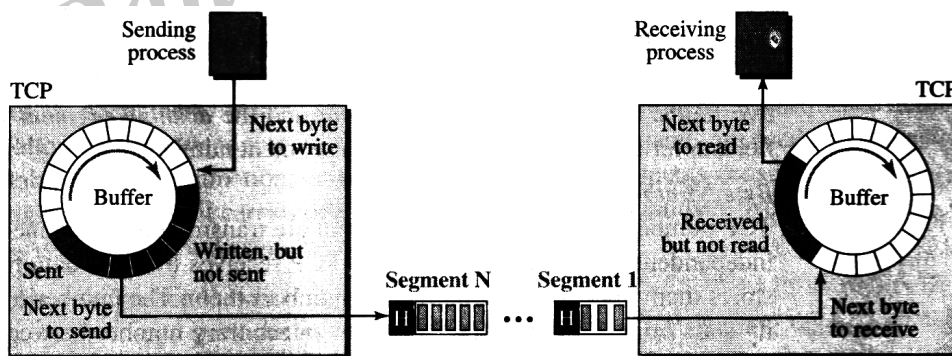


Fig.: TCP segments

Note that segments are not necessarily all the same size. In the figure, for simplicity we show one segment carrying 3 bytes and the other carrying 5 bytes. In reality, segments carry hundreds, if not thousands, of bytes.

(v) Full-Duplex Communication

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

(vi) Multiplexing and Demultiplexing

Like UDP, TCP performs multiplexing at the sender and demultiplexing at the receiver. However, since TCP is a connection-oriented protocol, a connection needs to be established for each pair of processes.

(vii) Connection-Oriented Service

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

1. The two TCP's establish a logical connection between them.
2. Data are exchanged in both directions.
3. The connection is terminated.

Note that this is a logical connection, not a physical connection. The TCP segment is encapsulated in an IP datagram and can be sent out of order, or lost or corrupted, and then resent. Each may be routed over a different path to reach the destination. There is no physical connection. TCP creates a stream-oriented environment in which it accepts the responsibility of delivering the bytes in order to the other site.

(viii) Reliable Service

TCP is a reliable transport protocol. It uses an acknowledgment mechanism to check the safe and sound arrival of data. We will discuss this feature further in the section on error control.

4.3.1 Congestion Control**Q4. Describe in detail TCP Congestion Control.**

Ans :

TCP uses different policies to handle the congestion in the network.

1. Congestion Window

When we discussed flow control in TCP, we mentioned that the size of the send window is controlled by the receiver using the value of *rwnd*, which is advertised in each segment traveling in the opposite direction. The use of this strategy guarantees that the receive window is never overflowed with the received bytes (no end congestion). This, however, does not mean that the intermediate buffers, buffers in the routers, do not become congested. A router may receive data from more than one sender. No matter how large the buffers of a router may be, it may be overwhelmed with data, which results in dropping some segments sent by a specific TCP sender. In other words, there is no congestion at the other end. but there may be congestion in the middle. TCP needs to worry about congestion in the middle because many segments lost may seriously affect the error control. More segment loss means resending the same segments again, resulting in worsening the congestion, and finally the collapse of the communication.

TCP is an end-to-end protocol that uses the service of IP. The congestion in the router is in the IP territory and should be taken care of by IP.

TCP cannot ignore the congestion in the network; it cannot aggressively send segments to the network. The result of such aggressiveness would hurt the TCP itself, as we mentioned before. TCP cannot be very conservative, either, sending a small number of segments in each time interval, because this means not utilizing the available band-width of the network. TCP needs to define policies that accelerate the data transmission when there is no congestion and decelerate the transmission when congestion is detected.

To control the number of segments to transmit, TCP uses another variable called a congestion window, *cwnd*, whose size is controlled by the congestion situation in the network (as we will explain shortly). The *cwnd* variable and the *rwnd* variable together define the size of the send window in TCP. The first is related to the congestion in the middle (network); the second is related to the congestion at the end. The actual size of the window is the minimum of these two.

$$\text{Actual window size} = \text{minimum}(\text{rwnd}, \text{cwnd})$$

2. Congestion Detection

Before discussing how the value of *cwnd* should be set and changed, we need to describe how a TCP sender can detect the possible existence of congestion in the network. The TCP sender uses the occurrence of two events as signs of congestion in the network: time-out and receiving three duplicate ACKs.

The first is the time-out. If a TCP sender does not receive an ACK for a segment or a group of segments before the time-out occurs, it assumes that the corresponding segment or segments are lost and the loss is due to congestion.

Another event is the receiving of three duplicate ACKs (four ACKs with the same acknowledgment number). Recall that when a TCP receiver sends a duplicate ACK, it is the sign that a segment has been delayed, but sending three duplicate ACKs is the sign of a missing segment, which can be due to congestion in the network. However, the congestion in the case of three duplicate ACKs can be less severe than in the case of time-out. When a receiver sends three duplicate ACKs, it means that one segment is missing, but three segments have been received. The network is either slightly congested or has recovered from the congestion.

We will show later that an earlier version of TCP, called Tahoe TCP, treated both events (time-out and three duplicate ACKs) similarly, but the later version of TCP, called Reno TCP, treats these two signs differently.

A very interesting point in TCP congestion is that the TCP sender uses only one feedback from the other end to detect congestion: ACKs. The lack of regular, timely receipt of ACKs, which results in a time-out, is the sign of a strong congestion; the receiving of three duplicate ACKs is the sign of a weak congestion in the network.

3. Congestion Policies

TCP's general policy for handling congestion is based on three algorithms: slow start, congestion avoidance, and fast recovery. We first discuss each algorithm before showing how TCP switches from one to the other in a connection.

4. Slow Start: Exponential Increase

The slow-start algorithm is based on the idea that the size of the congestion window (*cwnd*) starts with one maximum segment size (MSS), but it increases one MSS each time an acknowledgment arrives. As we discussed before, the MSS is a value negotiated during the connection establishment, using an option of the same name.

The name of this algorithm is misleading; the algorithm starts slowly, but grows exponentially. To show the idea, let us look at Figure. We assume that *rwnd* is much larger than *cwnd*, so that the sender window size always equals *cwnd*. We also assume that each segment is of the same size and carries MSS bytes. For simplicity, we also ignore the delayed-ACK policy and assume that each segment is acknowledged individually.

The sender starts with *cwnd* = 1. This means that the sender can send only one segment. After the first ACK arrives, the acknowledged segment is purged from the window, which means there is now one

empty segment slot in the window. The size of the congestion window is also increased by 1 because the arrival of the acknowledgment is a good sign that there is no congestion in the network. The size of the window is now 2. After sending two segments and receiving two individual acknowledgments for them, the size of the congestion window now becomes 4, and so on. In other words.

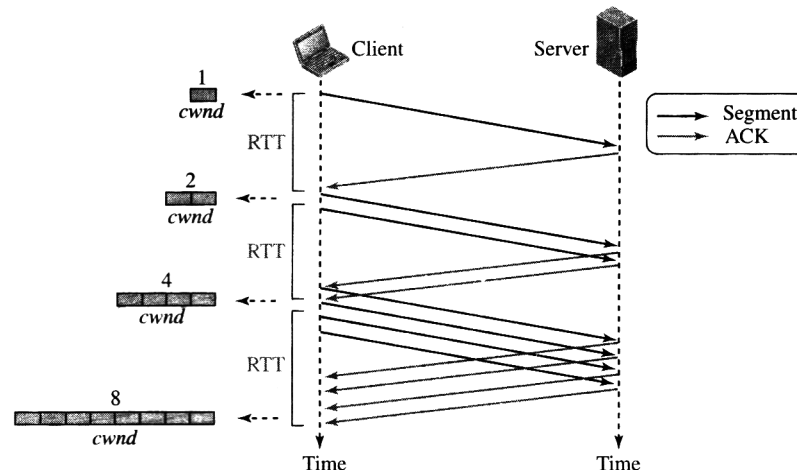


Fig.: Slow start, exponential increase

the size of the congestion window in this algorithm is a function of the number of ACKs arrived and can be determined as follows.

If an ACK arrives, $cwnd = cwnd + 1$.

If we look at the size of the cwnd in terms of round-trip times (RTTs), we find that the growth rate is exponential in terms of each round trip time, which is a very aggressive approach:

Start $\rightarrow cwnd = 1 \rightarrow 2^0$
 After 1 RTT $\rightarrow cwnd = cwnd + 1 = 1 + 1 = 2 \rightarrow 2^1$
 After 2 RTT $\rightarrow cwnd = cwnd + 2 = 2 + 2 = 4 \rightarrow 2^2$
 After 3 RTT $\rightarrow cwnd = cwnd + 4 = 4 + 4 = 8 \rightarrow 2^3$

A slow start cannot continue indefinitely. There must be a threshold to stop this phase. The sender keeps track of a variable named ssthresh (slow-start threshold). When the size of the window in bytes reaches this threshold, slow start stops and the next phase starts.

In the slow-start algorithm, the size of the congestion window increases exponentially until it reaches a threshold

We need, however, to mention that the slow-start strategy is slower in the case of delayed acknowledgments. Remember, for each ACK, the cwnd is increased by only 1. Hence, if two segments are acknowledged cumulatively, the size of the cwnd increases by only 1, not 2. The growth is still exponential, but it is not a power of 2. With one ACK for every two segments, it is a power of 1.5.

5. Congestion Avoidance: Additive Increase

If we continue with the slow-start algorithm, the size of the congestion window increases exponentially. To avoid congestion before it happens, we must slow down this exponential growth. TCP defines another algorithm called congestion avoidance, which increases the cwnd additively instead of exponentially. When the size of the congestion window reaches the slow-start threshold in the case where $cwnd = i$, the slow-start phase stops and the additive phase begins. In this algorithm, each time the whole "window" of segments is acknowledged, the size of the congestion window is increased by one.

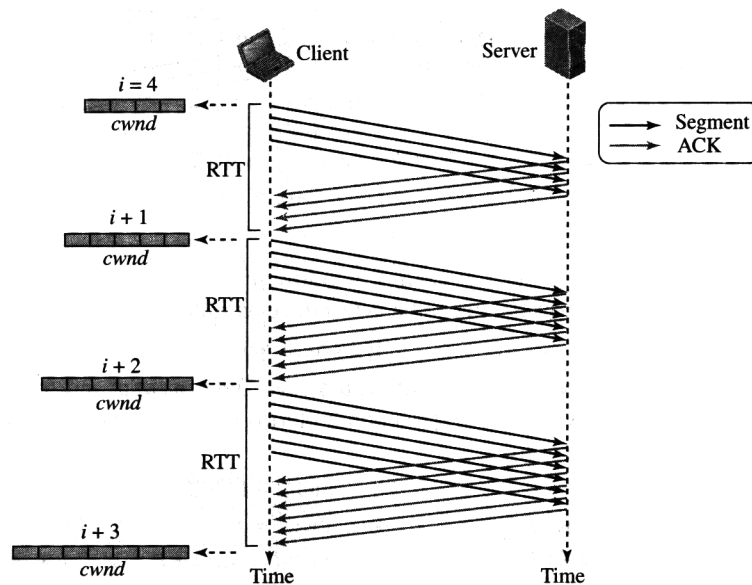


Fig.: Congestion avoidance, additive increase

The sender starts with $cwnd = 4$. This means that the sender can send only four segments. After four ACKs arrive, the acknowledged segments are purged from the window, which means there is now one extra empty segment slot in the window. The size of the congestion window is also increased by 1. The size of window is now 5. After sending five segments and receiving five acknowledgments for them, the size of the congestion window now becomes 6, and so on. In other words, the size of the congestion window in this algorithm is also a function of the number of ACKs that have arrived and can be determined as follows:

If an ACK arrives, $cwnd = cwnd + (1 / cwnd)$.

The size of the window increases only $1/cwnd$ portion of MSS (in bytes). In other words, all segments in the previous window should be acknowledged to increase the window 1 MSS bytes.

If we look at the size of the $cwnd$ in terms of round-trip times (RTTs), we find that the growth rate is linear in terms of each round-trip time, which is much more conservative than the slow-start approach.

Start $\rightarrow cwnd = i$
 After 1 RTT $\rightarrow cwnd = i + 1$
 After 2 RTT $\rightarrow cwnd = i + 2$
 After 3 RTT $\rightarrow cwnd = i + 3$

In the congestion-avoidance algorithm, the size of the congestion window increases additively until congestion is detected

Fast Recovery The fast-recovery algorithm is optional in TCP. The old version of TCP did not use it, but the new versions try to use it. It starts when three duplicate ACKs arrive, which is interpreted as light congestion in the network. Like congestion avoidance, this algorithm is also an additive increase, but it increases the size of the congestion window when a duplicate ACK arrives (after the three duplicate ACKs that trigger the use of this algorithm). We can say

If a duplicate ACK arrives, $cwnd = cwnd + (1 / cwnd)$

Policy Transition

We discussed three congestion policies in TCP. Now the questions when each of these policies is used and when TCP moves from one policy to another. To answer these questions, we need to refer to three versions of TCP: Tahoe TCP, Reno TCP, and New Reno TCP.

Tahoe TCP

The early TCP, known as Tahoe TCP, used only two different algorithms in their congestion policy: slow start and congestion avoidance. We use fig. to show the FSM for this version of TCP. However, we need to mention that we have deleted some small trivial actions, such as incrementing and resetting the number of duplicate ACKs, to make the FSM less crowded and simpler.

4.3.2 Timer Management

Q5. Discuss about TCP timer.

Ans :

To perform their operations smoothly, most TCP implementations use at least four timers: retransmission, persistence, keepalive, and TIME-WAIT.

1. Retransmission Timer

To retransmit lost segments, TCP employs one retransmission timer (for the whole connection period) that handles the retransmission time-out (RTO), the waiting time for an acknowledgment of a segment. We can define the following rules for the retransmission timer:

1. When TCP sends the segment in front of the sending queue, it starts the timer.
2. When the timer expires, TCP resends the first segment in front of the queue, and restarts the timer.
3. When a segment or segments are cumulatively acknowledged, the segment or segments are purged from the queue.
4. If the queue is empty, TCP stops the timer; otherwise, TCP restarts the timer.

Round-Trip Time (RTT)

To calculate the retransmission time-out (RTO), we first need to calculate the round-trip time (RTT). However, calculating RTT in TCP is an involved process.

- **Measured RTT:** We need to find how long it takes to send a segment and receive an acknowledgment for it. This is the measured RTT. We need to remember that the segments and their acknowledgments do not have a one-to-one relationship; several segments may be acknowledged together. The measured round-trip time for a segment is the time required for the segment to reach the destination and be acknowledged, although the acknowledgment may include other segments. Note that in TCP only one RTT measurement can be in progress at any time. This means that if an RTT measurement is started, no other measurement starts until the value of this RTT is finalized. We use the notation RTT_M to stand for measured RTT.

In TCP, there can be only one RTT measurement in progress at any time.

- **Smoothed RTT:** The measured RTT, RTT_M , is likely to change for each round trip. The fluctuation is so high in today's Internet that a single measurement alone cannot be used for retransmission time-out purposes. Most implementations use a smoothed RTT, called RTT_S , which is a weighted average of RTT_M and the previous RTT_S , as shown below:

Initially → No value

After first measurement → $RTT_s = RTT_M$

After each measurement → $RTT_s = (1 - \alpha) RTT_s + \alpha \times RTT_M$

The value of α is implementation-dependent, but it is normally set to 1/8. In other words, the new RTT_s is calculated as 7/8 of the old RTT_s and 1/8 of the current RTT_M .

- **RTT Deviation:** Most implementations do not just use RTT_s ; they also calculate the RTT deviation, called RTT_D , based on the RTT_s and RTT_M , using the following formulas. (The value of β is also implementation-dependent, but is usually set to 1/4.)

Initially → No value

After first measurement → $RTT_D = RTT_M/2$

After each measurement → $RTT_D = (1 - \beta) RTT_D + \beta \times |RTT_s - RTT_M|$

Retransmission Time-out (RTO) The value of RTO is based on the smoothed round-trip time and its deviation. Most implementations use the following formula to calculate the RTO:

Original → Initial value

After any measurement → $RTO = RTT_s + 4 \times RTT_D$

In other words, take the running smoothed average value of RTT_s and add four times the running smoothed average value of RTT_D (normally a small value).

2. Persistence Timer

To deal with a zero-window-size advertisement, TCP needs another timer. If the receiving TCP announces a window size of zero, the sending TCP stops transmitting segments until the receiving TCP sends an ACK segment announcing a nonzero window size. This ACK segment can be lost. Remember that ACK segments are not acknowledged nor retransmitted in TCP. If this acknowledgment is lost, the receiving TCP thinks that it has done its job and waits for the sending TCP to send more segments. There is no retransmission timer for a segment containing only an acknowledgment. The sending TCP has not received an acknowledgment and waits for the other TCP to send an acknowledgment advertising the size of the window. Both TCP's might continue to wait for each other forever (a deadlock).

To correct this deadlock, TCP uses a persistence timer for each connection. When the sending TCP receives an acknowledgment with a window size of zero, it starts a persistence timer. When the persistence timer goes off, the sending TCP sends a special segment called a probe. This segment contains only 1 byte of new data. It has a sequence number, but its sequence number is never acknowledged; it is even ignored in calculating the sequence number for the rest of the data. The probe causes the receiving TCP to resend the acknowledgment.

3. Keepalive Timer

A keepalive timer is used in some implementations to prevent a long idle connection between two TCPs. Suppose that a client opens a TCP connection to a server, transfers some data, and becomes silent. Perhaps the client has crashed. In this case, the connection remains open forever.

To remedy this situation, most implementations equip a server with a keepalive timer. Each time the server hears from a client, it resets this timer. The time-out is usually 2 hours. If the server does not hear from the client after 2 hours, it sends a probe segment. If there is no response after 10 probes, each of which is 75 seconds apart, it assumes that the client is down and terminates the connection.

TIME-WAIT Timer

The TIME-WAIT (2MSL) timer is used during connection termination. The maximum segment lifetime (MSL) is the amount of time any segment can exist in a network before being discarded. The implementation needs to choose a value for MSL. Common values are 30 seconds, 1 minute, or even 2 minutes. The 2MSL timer is used when TCP performs an active close and sends the final ACK. The connection must stay open for 2 MSL amount of time to allow TCP to resend the final ACK in case the ACK is lost. This requires that the RTO timer at the other end times out and new FIN and ACK segments are resent.

4.4 QUALIFY OF SERVICES

Q6. Define QoS. Explain the specific purpose of QoS.

Ans :

(Imp.)

Meaning

Quality of service (QoS) is the overall performance of a telephony or computer network, particularly the performance seen by the users of the network. To quantitatively measure quality of service, several related aspects of the network service are often considered, such as error rates, bandwidth, throughput, transmission delay, availability, jitter, etc. Quality of service is particularly important for the transport of traffic with special requirements. In particular, much technology has been developed to allow computer networks to become as useful as telephone networks for audio conversations, as well as supporting new applications with even stricter service demands.

In the field of telephony, quality of service was defined by the ITU in 1994. Quality of service comprises requirements on all the aspects of a connection, such as service response time, loss, signal-to-noise ratio, crosstalk, echo, interrupts, frequency response, loudness levels, and so on. A

subset of telephony QoS is grade of service (GoS) requirements, which comprises aspects of a connection relating to capacity and coverage of a network, for example guaranteed maximum blocking probability and outage probability.

In the field of computer networking and other packet-switched telecommunication networks, the traffic engineering term refers to resource reservation control mechanisms rather than the achieved service quality. Quality of service is the ability to provide different priority to different applications, users, or data flows, or to guarantee a certain level of performance to a data flow.

For example, a required bit rate, delay, jitter, packet dropping probability and/or bit error rate may be guaranteed. Quality of service guarantees are important if the network capacity is insufficient, especially for real-time streaming multimedia applications such as voice over IP, online games and IP-TV, since these often require fixed bit rate and are delay sensitive, and in networks where the capacity is a limited resource, for example in cellular data communication.

A network or protocol that supports QoS may agree on a traffic contract with the application software and reserve capacity in the network nodes, for example during a session establishment phase. During the session it may monitor the achieved level of performance, for example the data rate and delay, and dynamically control scheduling priorities in the network nodes. It may release the reserved capacity during a tear down phase.

A best-effort network or service does not support quality of service. An alternative to complex QoS control mechanisms is to provide high quality communication over a best-effort network by over-provisioning the capacity so that it is sufficient for the expected peak traffic load. The resulting absence of network congestion eliminates the need for QoS mechanisms.

QoS is sometimes used as a quality measure, with many alternative definitions, rather than referring to the ability to reserve resources. Quality of service sometimes refers to the level of quality of service, i.e. the guaranteed service quality.[3] High QoS is often confused with a high level of performance or achieved service quality, for example high bit rate, low latency and low bit error probability.

An alternative and disputable definition of QoS, used especially in application layer services such as telephony and streaming video, is requirements on a metric that reflects or predicts the subjectively experienced quality. In this context, QoS is the acceptable cumulative effect on subscriber satisfaction of all imperfections affecting the service. Other terms with similar meaning are the quality of experience (QoE) subjective business concept, the required "user perceived performance", [4] the required "degree of satisfaction of the user" or the targeted "number of happy customers". Examples of measures and measurement methods are mean opinion score (MOS), perceptual speech quality measure (PSQM) and perceptual evaluation of video quality (PEVQ). See also subjective video quality.

Qualities of Traffic

In packet-switched networks, quality of service is affected by various factors, which can be divided into "human" and "technical" factors. Human factors include: stability of service, availability of service, delays, user information. Technical factors include: reliability, scalability, effectiveness, maintainability, grade of service, etc. Many things can happen to packets as they travel from origin to destination, resulting in the following problems as seen from the point of view of the sender and receiver:

1. Low throughput

Due to varying load from disparate users sharing the same network resources, the bit

rate (the maximum throughput) that can be provided to a certain data stream may be too low for real-time multimedia services if all data streams get the same scheduling priority. Dropped packets The routers might fail to deliver (drop) some packets if their data loads are corrupted, or the packets arrive when the router buffers are already full. The receiving application may ask for this information to be retransmitted, possibly causing severe delays in the overall transmission.

2. Errors

Sometimes packets are corrupted due to bit errors caused by noise and interference, especially in wireless communications and long copper wires. The receiver has to detect this and, just as if the packet was dropped, may ask for this information to be retransmitted. Latency It might take a long time for each packet to reach its destination, because it gets held up in long queues, or it takes a less direct route to avoid congestion. This is different from throughput, as the delay can build up over time, even if the throughput is almost normal. In some cases, excessive latency can render an application such as VoIP or online gaming unusable.

3. Jitter

Packets from the source will reach the destination with different delays. A packet's delay varies with its position in the queues of the routers along the path between source and destination and this position can vary unpredictably. This variation in delay is known as jitter and can seriously affect the quality of streaming audio and/or video.

4. Out-of-order delivery

When a collection of related packets is routed through a network, different packets may take different routes, each resulting in a different delay. The result is that the packets arrive in a different order than they were sent. This

problem requires special additional protocols responsible for rearranging out-of-order packets to an isochronous state once they reach their destination. This is especially important for video and VoIP streams where quality is dramatically affected by both latency and lack of sequence.

Applications

A defined quality of service may be desired or required for certain types of network traffic, for example :

- Streaming media specifically
 - Internet protocol television (IPTV)
 - Audio over Ethernet
 - Audio over IP
- IP telephony also known as Voice over IP (VoIP)
- Videoconferencing
- Tele-presence
- Storage applications such as iSCSI and FCoE
- Circuit Emulation Service
- Safety-critical applications such as remote surgery where availability issues can be hazardous
- Network operations support systems either for the network itself, or for customers' business critical needs
- Online games where real-time lag can be a factor
- Industrial control systems protocols such as Ethernet/IP which are used for real-time control of machinery

These types of service are called inelastic, meaning that they require a certain minimum level of bandwidth and a certain maximum latency to function. By contrast, elastic applications can take

advantage of however much or little bandwidth is available. Bulk file transfer applications that rely on TCP are generally elastic.

Application Layer

At the top of the TCP/IP protocol architecture is the Application Layer. This layer includes all processes that use the Transport Layer protocols to deliver data. There are many applications protocols. Most provide user services, and new services are always being added to this layer.

- The most widely known and implemented applications protocols are: Telnet
- The Network Terminal Protocol, which provides remote login over the network. FTP
- The File Transfer Protocol, which is used for interactive file transfer. SMTP
- The Simple Mail Transfer Protocol, which delivers electronic mail. HTTP

The Hypertext Transfer Protocol, which delivers Web pages over the network. While HTTP, FTP, SMTP, and telnet are the most widely implemented TCP/IP applications, you will work with many others as both a user and a system administrator. Some other commonly used TCP/IP applications are: *Domain Name Service (DNS)*

Open Shortest Path First (OSPF)

Routing is central to the way TCP/IP works. OSPF is used by network devices to exchange routing information. Routing is also a major topic of this book. Network File system (NFS)

This protocol allows files to be shared by various hosts on the network some protocols, such as telnet and FTP, can only be used if the user has some knowledge of the network. Other protocols, like OSPF, run without the user even knowing that they exist. As system administrator, you are aware of all these applications and all the protocols in the other TCP/IP layers.

4.5 USER DATAGRAM PROTOCOL (UDP)

Q7. Explain UDP and its Attributes.

Ans : (Imp.)

The User Datagram Protocol (UDP) is one of the core members of the Internet protocol suite. The protocol was designed by David P. Reed in 1980 and formally defined in RFC 768.

UDP uses a simple connectionless transmission model with a minimum of protocol mechanism. It has no handshaking dialogues, and thus exposes any unreliability of the underlying network protocol to the user's program. There is no guarantee of delivery, ordering, or duplicate protection. UDP provides checksums for data integrity, and port numbers for addressing different functions at the source and destination of the datagram.

With UDP, computer applications can send messages, in this case referred to as datagram, to other hosts on an Internet Protocol (IP) network without prior communications to set up special transmission channels or data paths. UDP is suitable for purposes where error checking and correction is either not necessary or is performed in the application, avoiding the overhead of such processing at the network interface level. Time-sensitive applications often use UDP because dropping packets is preferable to waiting for delayed packets, which may not be an option in a real-time system.[1] If error correction facilities are needed at the network interface level, an application may use the Transmission Control Protocol (TCP) or Stream Control Transmission Protocol (SCTP).

UDP (User Datagram Protocol) is an alternative communications protocol to Transmission Control Protocol (TCP) used primarily for establishing

low-latency and loss tolerating connections between applications on the Internet. Both UDP and TCP run on top of the Internet Protocol (IP) and are sometimes referred to as UDP/IP or TCP/IP. Both protocols send short packets of data, called datagram.

UDP provides two services not provided by the IP layer. It provides port numbers to help distinguish different user requests and, optionally, a checksum capability to verify that the data arrived intact.

TCP has emerged as the dominant protocol used for the bulk of Internet connectivity owing to services for breaking large data sets into individual packets, checking for and resending lost packets and reassembling packets into the correct sequence. But these additional services come at a cost in terms of additional data overhead, and delays called latency.

UDP is an ideal protocol for network applications in which perceived latency is critical such as gaming, voice and video communications, which can suffer some data loss without adversely affecting perceived quality. In some cases, forward error correction techniques are used to improve audio and video quality in spite of some loss.

UDP can also be used in applications that require lossless data transmission when the application is configured to manage the process of retransmitting lost packets and correctly arranging received packets. This approach can help to improve the data transfer rate of large files compared with TCP.

Attributes

A number of UDP's attributes make it especially suited for certain applications.

- It is transaction-oriented, suitable for simple query-response protocols such as the Domain Name System or the Network Time Protocol.
- It provides datagram, suitable for modeling other protocols such as in IP tunneling or Remote Procedure Call and the Network File System.
- It is simple, suitable for bootstrapping or other purposes without a full protocol stack, such as the DHCP and Trivial File Transfer Protocol.
- It is stateless, suitable for very large numbers of clients, such as in streaming media applications for example IPTV
- The lack of retransmission delays makes it suitable for real-time applications such as Voice over IP, online games, and many protocols built on top of the Real Time Streaming Protocol.
- Works well in unidirectional communication, suitable for broadcast information such as in many kinds of service discovery and shared information such as broadcast time or Routing Information Protocol
- UDP provides application multiplexing (via port numbers) and integrity verification (via checksum) of the header and payload.[4] If transmission reliability is desired, it must be implemented in the user's application.

The UDP header consists of 4 fields, each of which is 2 bytes (16 bits).[1] The use of the fields "Checksum" and "Source port" is optional in IPv4 (pink background in table). In IPv6 only the source port is optional (see below).

- **Source port number:** This field identifies the sender's port when meaningful and should be assumed to be the port to reply to if needed. If not used, then it should be zero. If the source host is the client, the port number is likely to be an ephemeral port number. If the source host is the server, the port number is likely to be a well-known port number.[2]
- **Destination port number:** This field identifies the receiver's port and is required. Similar to source port number, if the client is the destination host then the port number will likely be an ephemeral port number and if the destination host is the server then the port number will likely be a well-known port number.
- **Length:** A field that specifies the length in bytes of the UDP header and UDP data. The minimum length is 8 bytes because that is the length of the header. The field size sets a theoretical limit of 65,535 bytes (8 byte header + 65,527 bytes of data) for a UDP datagram. The practical limit for the data length which is imposed by the underlying IPv4 protocol is 65,507 bytes (65,535 " 8 byte UDP header " 20 byte IP header).
- In IPv6 Jumbo grams it is possible to have UDP packets of size greater than 65,535 bytes.[5] RFC 2675 specifies that the length field is set to zero if the length of the UDP header plus UDP data is greater than 65,535.
- **Checksum:** The checksum field is used for error-checking of the header and data. If no checksum is generated by the transmitter, the field uses the value all-zeros.[6] This field is not optional for IPv6.[7]

Reliable byte stream (TCP)

A reliable byte stream is a common service paradigm in computer networking; it refers to a byte stream in which the bytes which emerge from the communication channel at the recipient are exactly the same, and in exactly the same order, as they were when the sender inserted them into the channel.

The classic example of a reliable byte stream communication protocol is the Transmission Control Protocol, one of the major building blocks of the Internet.

A reliable byte stream is not the only reliable service paradigm which computer network communication protocols provide, however; other protocols (e.g. SCTP) provide a reliable message stream, i.e. the data is divided up into distinct units, which are provided to the consumer of the data as discrete objects.

Connection-oriented (TCP)

- **Flow control:** keep sender from overrunning receiver
- **Congestion control:** keep sender from overrunning network.

Rahul Publications

UNIT V

Socket Programming: Primitive and Advance System calls, Iterative and concurrent client server programs

Application Layer: Domain Name Space (DNS) - SMTP - FTP - HTTP

5.1 SOCKET PROGRAMMING

5.1.1 Primitive and Advance System Calls

Q1. Explain the concept of Socket Programming.

Ans :

Socket interface started in the early 1980s at UC Berkeley as part of a UNIX environment. The socket interface is a set of instructions that provide communication between the application layer and the operating system, as shown in fig. It is a set of instructions that can be used by a process to communicate with another process.

The idea of sockets allows us to use the set of all instructions already designed in a programming language for other sources and sinks. For example, in most computer languages, like C, C++, or Java, we have several instructions that can read and write data to other sources and sinks such as a keyboard (a source), a monitor (a sink), or a file (source and sink). We can use the same instructions to read from or write to sockets. In other words, we are adding only new sources and sinks to the programming language

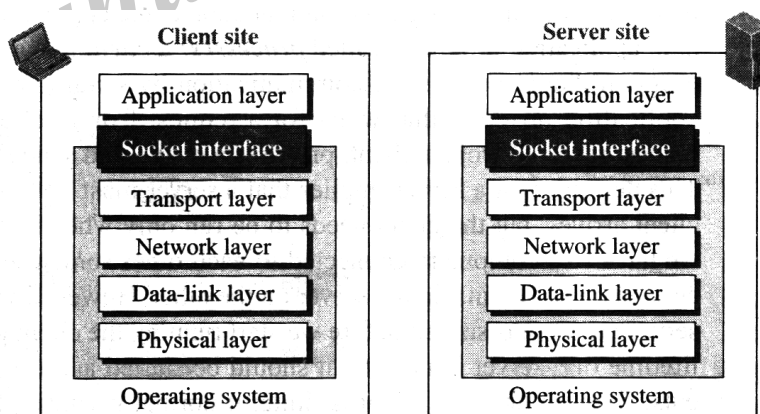


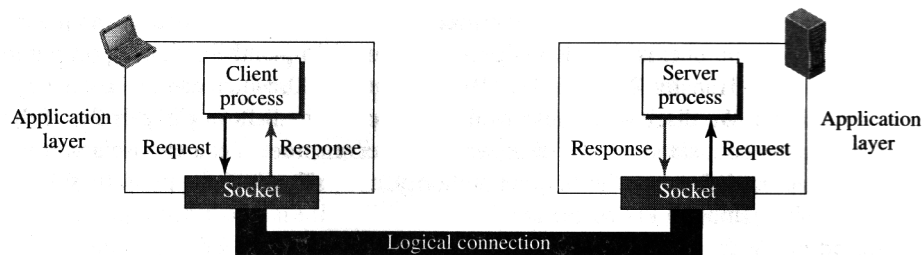
Fig.: Position of the socket interface

without changing the way we send data or receive data fig. shows the idea and compares the sockets with other sources and sinks.

Sockets

Although a socket is supposed to behave like a terminal or a file, it is not a physical entity like them; it is an abstraction. It is an object that is created and used by the application program.

We can say that, as far as the application layer is concerned, communication between a client process and a server process is communication between two sockets, created at two ends, as shown in Figure 25.6. The client thinks that the socket is the entity that receives the request and gives the response; the server thinks that the socket is the one that has a request and needs the response. If we create two sockets, one at each end, and define the source and destination addresses correctly, we can use the available instructions to send and receive data. The rest is the responsibility of the operating system and the embedded TCP/IP protocol.



Socket Addresses

The interaction between a client and a server is two-way communication. In a two-way communication, we need a pair of addresses: local (sender) and remote (receiver). The local address in one direction is the remote address in the other direction and vice versa. Since communication in the client-server paradigm is between two sockets, we need a pair of socket addresses for communication: a local socket address and a remote socket address. However, we need to define a socket address in terms of identifiers used in the TCP/IP protocol suite.

A socket address should first define the computer on which a client or a server is running. As we discussed in Chapter 18, a computer in the Internet is uniquely defined by its IP address, a 32-bit integer in the current Internet version. However, several client or server processes may be running at the same time on a computer, which means that we need another identifier to define the specific client or server involved in the communication.

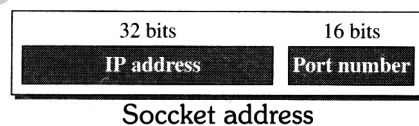


Fig.: A socket address

Since a socket defines the end-point of the communication, we can say that a socket is identified by a pair of socket addresses, a local and a remote.

Finding Socket Addresses

How can a client or a server find a pair of socket addresses for communication? The situation is different for each site.

Server Site

The server needs a local (server) and a remote (client) socket address for communication.

Client Site

The client also needs a local (client) and a remote (server) socket address for communication.

Local Socket Address The local (client) socket address is also provided by the operating system.

The operating system knows the IP address of the computer on which the client is running. The port number, however, is a 16-bit temporary integer that is assigned to a client process each time the process

needs to start the communication. The port number, however, needs to be assigned from a set of integers defined by the Internet authority and called the ephemeral (temporary) port numbers, which we discussed in Chapter 24. The operating system, however, needs to guarantee that the new port number is not used by any other running client process. The operating system needs to remember the port number to be able to redirect the response received from the server process to the client process that sent the request.

Remote Socket Address

Finding the remote (server) socket address for a client, how-ever, needs more work. When a client process starts, it should know the socket address of the server it wants to connect to. We will have two situations in this case.

- (i) Sometimes, the user who starts the client process knows both the server port number and IP address of the computer on which the server is running. This usually occurs in situations when we have written client and server applications and we want to test them. For example, at the end of this chapter we write some simple client and server programs and we test them using this approach. In this situation, the programmer can provide these two pieces of information when he runs the client program.
- (ii) Although each standard application has a well-known port number, most of the time, we do not know the IP address. This happens in situations such as when we need to contact a web page, send an e-mail to a friend, copy a file from a remote site, and so on. In these situations, the server has a name, an identifier that uniquely defines the server process. Examples of these identifiers are URLs, such as `www.xxx.yyy`, or e-mail addresses, such as `xxxx@yyyy.com`. The client process should now change this identifier (name) to the corresponding server socket address. The client process normally knows the port number because it should be a well-known port number, but the IP address can be obtained using another client-server application called the Domain Name System (DNS).

5.2 ITERATIVE AND CONCURRENT CLIENT SERVER PROGRAMS

Q2. Describe various services of transport layer.

Ans :

(Imp.)

A pair of processes provide services to the users of the Internet, human or programs. A pair of processes, however, need to use the services provided by the transport layer for communication because there is no physical communication at the application layer.

UDP Protocol

UDP provides connectionless, unreliable, datagram service. Connectionless service means that there is no logical connection between the two ends exchanging messages. Each message is an independent entity encapsulated in a datagram. UDP does not see any relation (connection) between consequent datagrams coming from the same source and going to the same destination.

UDP is not a reliable protocol. Although it may check that the data is not corrupted during the transmission, it does not ask the sender to resend the corrupted or lost data-gram. For some applications, UDP has an advantage: it is message-oriented. It gives boundaries to the messages exchanged. An application program may be designed to use UDP if it is sending small messages and the simplicity and speed is more important for the application than reliability. For example, some management and multimedia applications fit in this category.

TCP Protocol

TCP provides connection-oriented, reliable, byte-stream service. TCP requires that two ends first create a logical connection between themselves by exchanging some connection-establishment packets.

This phase, which is sometimes called **handshaking**, establishes some parameters between the two ends, including the size of the data packets to be exchanged, the size of buffers to be used for holding the chunks of data until the whole message arrives, and so on. After the handshaking process, the two ends can send chunks of data in segments in each direction. By numbering the bytes exchanged, the continuity of the bytes can be checked. For example, if some bytes are lost or corrupted, the receiver can request the resending of those bytes, which makes TCP a reliable protocol. TCP also can provide flow control and congestion control, as we saw in Chapter 24. One problem with the TCP protocol is that it is not message-oriented; it does not put boundaries on the messages exchanged. Most of the standard applications that need to send long messages and require reliability may benefit from the service of the TCP.

SCTP Protocol

SCTP provides a service which is a combination of the two other protocols. Like TCP, SCTP provides a connection-oriented, reliable service, but it is not byte-stream oriented. It is a message-oriented protocol like UDP. In addition, SCTP can provide multi-stream service by providing multiple network-layer connections. SCTP is normally suitable for any application that needs reliability and at the same time needs to remain connected, even if a failure occurs in one network-layer connection.

Q3. Explain iterative communication by using UDP.

Ans :

(Imp.)

Communication between a client program and a server program can occur iteratively or concurrently. Although several client programs can access the same server program at the same time, the server program can be designed to respond iteratively or concurrently. An iterative server can process one client request at a time; it receives a request, processes it, and sends the response to the requestor before handling another request. When the server is handling the request from a client, the requests from other clients, and even other requests from the same client, need to be queued at the server site and wait for the server to be freed. The received and queued requests are handled in the first-in, first-out fashion. In this section, we discuss iterative communication using UDP.

Sockets Used for UDP

In UDP communication, the client and server use only one socket each. The socket created at the server site lasts forever; the socket created at the client site is closed (destroyed) when the client process terminates. Figure shows the lifetime of the sockets in the server and client processes. In other words, different clients use different sockets, but the server creates only one socket and changes only the remote socket address each time a new client makes a connection. This is logical, because the server does know its own socket address, but does not know the socket addresses of the clients who need its services; it needs to wait for the client to connect before filling this part of the socket address.

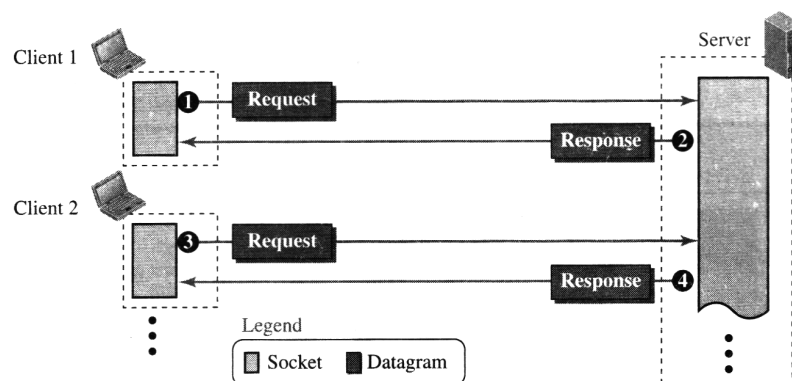


Fig.: Sockets for UDP communication

Flow Diagram

As UDP provides a connectionless service, in which a client sends a request and the server sends back a response. Figure shows a simplified flow diagram for iterative communication. There are multiple clients, but only one server. Each client is served in each iteration of the loop in the server. Note that there is no connection establishment or connection termination. Each client sends a single datagram and receives a single datagram. In other words, if a client wants to send two datagrams, it is considered as two clients for the server. The second datagram needs to wait for its turn. The diagram also shows the status of the socket after each action.

Server Process

The server makes a passive open, in which it becomes ready for the communication, but it waits until a client process makes the connection. It creates an empty socket. It then binds the socket to the server and the well-known port, in which only part of the socket (the server socket address) is filled (binding can happen at the time of creation depending on the underlying language). The server then issues a receive request command, which blocks until it receives a request from a client. The server then fills the rest of the socket (the client socket section) from the information obtained in the request. The request is the process and the response is sent back to the client. The server now starts another iteration waiting for another request to arrive (an infinite loop). Note that in each iteration, the socket becomes only half-filled again; the client socket address is erased. It is totally filled only when a request arrives.

Client Process

The client process makes an active open. In other words, it starts a connection. It creates an empty socket and then issues the send command, which fully fills the socket, and sends the request. The client then issues a receive command, which is blocked until a response arrives from the server. The response is then handled and the socket is destroyed.

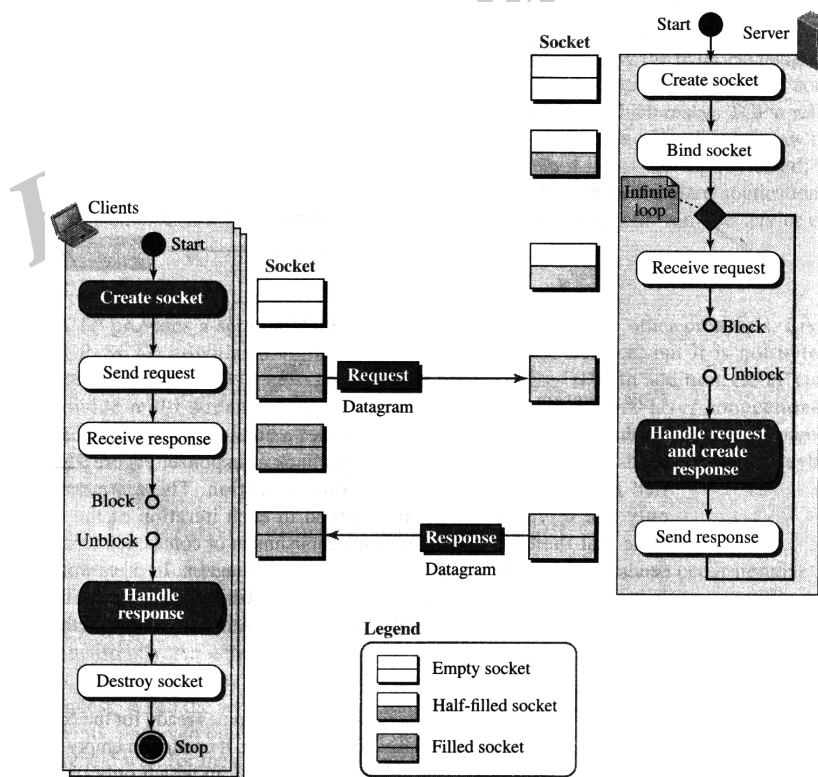


Fig.: Flow diagram for iterative UDP communication

Q4. Explain iterative communication by using TCP.*Ans.:***(Imp.)**

TCP is a connection-oriented protocol. Before sending or receiving data, a connection needs to be established between the client and the server. After the connection is established, the two parties can send and receive chunks of data as long as they have data to do so. Although iterative communication using TCP is not very common, because it is simpler we discuss this type of communication in this section.

Sockets Used in TCP

The TCP server uses two different sockets, one for connection establishment and the other for data transfer. We call the first one the listen socket and the second the socket. The reason for having two types of sockets is to separate the connection phase from the data exchange phase. A server uses a listen socket to listen for a new client trying to establish connection. After the connection is established, the server creates a socket to exchange data with the client and finally to terminate the connection. The client uses only one socket for both connection establishment and data exchange.

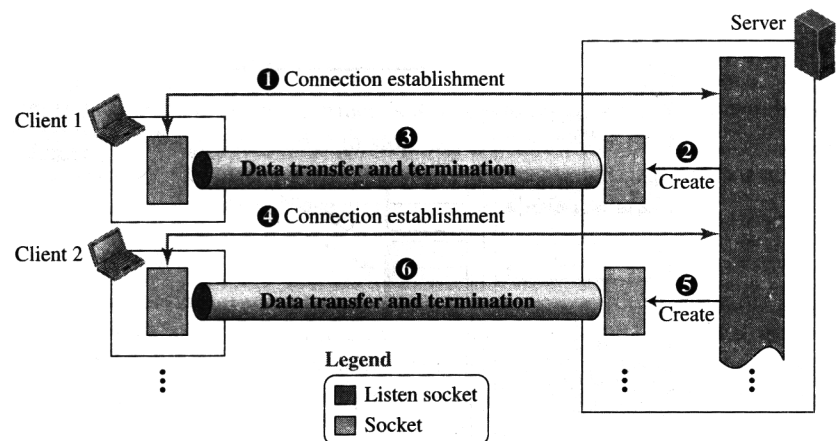
**Fig.: Sockets used in TCP communication****Flow Diagram**

Figure shows a simplified flow diagram for iterative communication using TCP. There are multiple clients, but only one server. Each client is served in each iteration of the loop. The flow diagram is almost similar to the one for UDP, but there are differences that we explain for each site.

Server Process

In Figure the TCP server process, like the UDP server process, creates a socket and binds it, but these two commands create the listen socket to be used only for the connection establishment phase. The server process then calls the listen procedure, to allow the operating system to start accepting the clients, completing the connection phase, and putting them in the waiting list to be served.

The server process now starts a loop and serves the clients one by one. In each iteration, the server process issues the accept procedure that removes one client from the waiting list of the connected clients for serving. If the list is empty, the accept procedure blocks until there is a client to be served. When the accept procedure returns, it creates a new socket for data transfer. The server process now uses the client socket address obtained during the connection establishment to fill the remote socket address field in the newly created socket. At this time the client and server can exchange data.

Client Process

The client flow diagram is almost similar to the UDP version except that the client data-transfer box needs to be defined for each specific case. We do so when we write a specific program later.

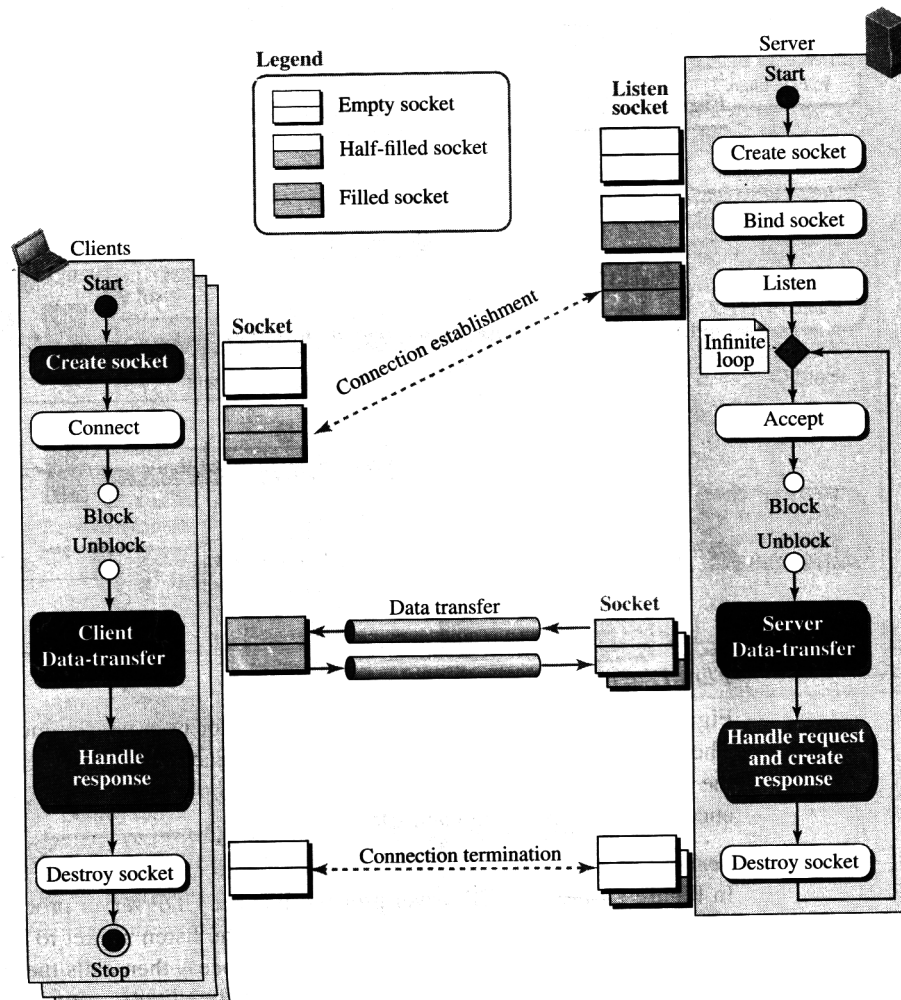


Fig.: Flow diagram for iterative TCP communication

5.3 APPLICATION LAYER

Q5. Explain Application Layer and its Functions.

Ans :

(Imp.)

It is the top most layer of OSI Model. Manipulation of data(information) in various ways is done in this layer which enables user or software to get access to the network. Some services provided by this layer includes: E-Mail, transferring files, distributing the results to user, directory services, network resources, etc.

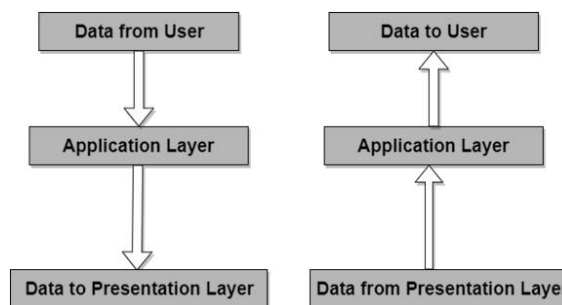
The Application Layer contains a variety of protocols that are commonly needed by users. One widely-used application protocol is HTTP(HyperText Transfer Protocol), which is the basis for the World

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 that are used are: File Transfer Protocol(FTP), Trivial File Transfer Protocol(TFTP), Simple Mail Transfer Protocol (SMTP), TELNET, Domain Name System (DNS) etc.

Functions of Application Layer

1. **Mail Services:** This layer provides the basis for E-mail forwarding and storage.
2. **Network Virtual Terminal:** It allows a user to log on to a remote host. The application creates software emulation of a terminal at the remote host. User's computer talks to the software terminal which in turn talks to the host and vice versa. Then the remote host believes it is communicating with one of its own terminals and allows user to log on.
3. **Directory Services:** This layer provides access for global information about various services.
4. **File Transfer, Access and Management (FTAM) :** It is a standard mechanism to access files and manages it. Users can access files in a remote computer and manage it. They can also retrieve files from a remote computer.



Design Issues with Application Layer

There are commonly reoccurring problems that occur in the design and implementation of Application Layer protocols and can be addressed by patterns from several different pattern languages:

- Pattern Language for Application-level Communication Protocols
- Service Design Patterns
- Patterns of Enterprise Application Architecture
- Pattern-Oriented Software Architecture.

5.3.1 DNS (Domain Name System)

Q6. Explain in concept of Domain Name System.

Ans :

(Imp.)

Programs rarely refer to hosts, mailboxes, and other resources by their bonar network addresses. Instead of binary numbers, they use ASCII strings, such as tana@art.ucsb.edu. Nevertheless, the network itself only understands bmsn addresses, so some mechanism is required to convert the ASCII strings to network addresses. In the following sections we will study how this mapping is accomplished plished in the Internet.

Way back in the ARPANET, there was simply a file, `hosts.txt`, that listed all the hosts and their IP addresses. Every night, all the hosts would fetch it from the site at which it was maintained. For a network of a few hundred large timesharing machines, this approach worked reasonably well.

However, when thousands of workstations were connected to the net, every-one realized that this approach could not continue to work forever. For one thing, the size of the file would become too large. However, even more important, host name conflicts would occur constantly unless names were centrally managed, something unthinkable in a huge international network. To solve these problems, DNS (the Domain Name System) was invented.

Conceptually, the Internet is divided into several hundred top-level domains, K where each domain covers many hosts. Each domain is partitioned into sub-domains, and these are further partitioned, and so on. All these domains can be represented by a tree, as shown in Fig. The leaves of the tree represent domains that have no subdomains (but do contain machines, of course) A leaf domain may contain a single host, or it may represent a company and contains thousands of hosts.

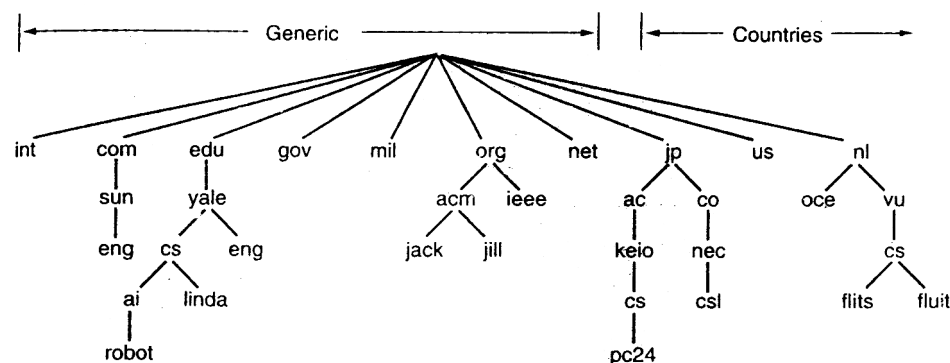


Fig.: A portion of the Internet domain name space.

The top-level domains come in two flavors: generic and countries. The generic domains are `com` (commercial), `edu` (educational institutions), `gov` (the U.S. federal government), `int` (certain international organizations), `mil` (the U.S. armed forces), `net` (net/ork providers), and `org` (nonprofit organizations). The country domains include one entry for every country, as defined in ISO 3166.

Each domain is named by the path upward from it to the (unnamed) root. The components are separated by periods (pronounced “dot”). Thus Sun Microsystems engineering department might be `eng.sun.com.`, rather than a UNIX-style name such as `/com/sun/eng`. Notice that this hierarchical naming means that `eng.sun.com.` does not conflict with a potential use of `eng` in `eng.yale.edu.`, which might be used by the Yale English department.

Domain names can be either absolute or relative. An absolute domain name ends with a period (e.g., `eng.sun.com.`), whereas a relative one does not. Relative names have to be interpreted in some context to uniquely determine their true meaning. In both cases, a named domain refers to a specific node in the tree and all the nodes under it.

Domain names are case insensitive, so `edu` and `EDU` mean the same thing. Component names can be up to 63 characters long, and full path names must not exceed 255 characters.

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 of resource records 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 real function of DNS is to map domain names onto resource records.

A resource record is a Five-tuple. Although they are encoded in binary for efficiency, in most expositions resource records are presented as ASCII text, one line per resource record. The format we will use is as follows:

Domain_name Time_to_live Type Class Value

1. The Domain_name tells the domain to which this record applies. Normally, many Records exist for each domain and each copy of the database holds information about multiple domains. This field is thus the primary search key used to satisfy queries. The order of the records in the database is not significant. When a query is made about a domain, all the matching records of the class requested are returned.
2. The Time_to_live field gives an indication of how stable the record is. Information that is highly stable is assigned a large value, such as 86400 (the number of seconds in 1 day). Information that is highly volatile is assigned a small value, such as 60 (1 minute). We will come back to this point later when we have discussed caching.
3. The Type field tells what kind of record this is. The most important types are listed in Fig.

Type	Meaning	Value
SOA	Start of Authority	Parameters for this zone
A	IP address of a host	32-Bit integer
MX	Mail exchange	Priority, domain willing to accept email
NS	Name Server	Name of a server for this domain
CNAME	Canonical name	Domain name
PTR	Pointer	Alias for an IP address
HINFO	Host description	CPU and OS in ASCII
TXT	Text	Uninterpreted ASCII text

Fig.: The principal DNS resource record types

- An SOA record provides the name of the primary source of information about the name server's zone (described below), the email address of its administrator, a unique serial number, and various flags and timeouts.
- The most important record type is the A (Address) record. It holds a 32-bit IP address for some host. Every Internet host must have at least one IP address, so other machines can communicate with it. Some hosts have two or more network connections, in which case they will have one type A resource record per network connection (and thus per IP address).
- The next most important record type is the MX record. It specifies the name of the domain prepared to accept email for the specified domain. A common use of this record is to allow a machine that is not on the Internet to receive email from Internet sites. Delivery is accomplished by having the non-Internet site make an arrangement with some Internet site to accept email for it and forward it using whatever protocol the two of them agree on.

Name Servers

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. Furthermore, if it ever went down, the entire Internet would be crippled.

To avoid the problems associated with having only a single source of information, the DNS name space is divided up into nonoverlapping zones. One possible way to divide up the name space of Fig. is shown in Fig. Each zone contains some part of the tree and also contains name servers holding the authoritative information about that zone. Normally, a zone will have one primary name server, which gets its information from a file on its disk, and one or more secondary name servers, which get their information from the primary name server. To improve reliability, some servers for a zone can be located outside the zone.

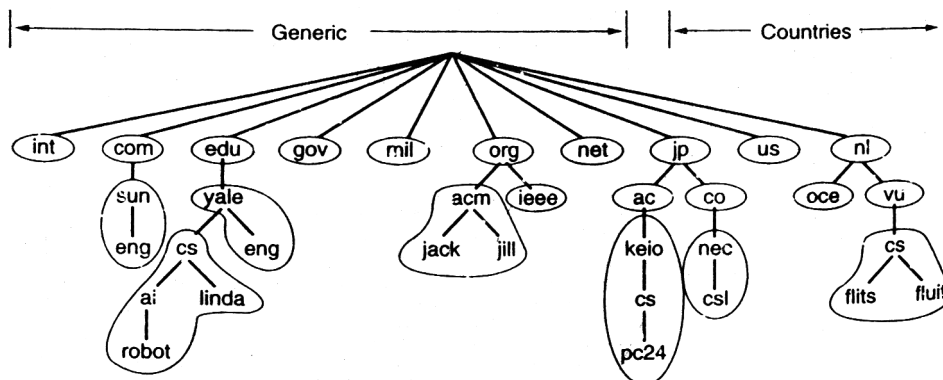


Fig.: Part of the DNS name space showing the division into zones.

Where the zone boundaries are placed within a zone is up to that zone administrator. This decision is made in large part based on how many name servers are desired, and where. For example, in Fig. Yale has a server for yale.edu that handles eng.yale.edu but not cs.yale.edu, which is a separate zone with its own name servers. Such a decision might be made when a department such as English does not wish to run its own name server, but a department such as computer science does. Consequently, cs.yale.edu is a separate zone but eng.yale.edu is not.

When a resolver has a query about a 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 is thus always correct. Authoritative records are in contrast to cached records, which may be out of date.

If, however, the domain is remote and no information about the requested domain is available locally, the name server sends a query message to the top-level name server for the domain requested. To make this process clearer, consider the example of Fig. Here, a resolver on flits.cs.vu.nl wants to know the IP address of the host linda.cs.yale.edu. In step 1 it sends a query to the local name server, cs.vu.nl. This query contains the domain name sought, the type (A) and the class (IN).

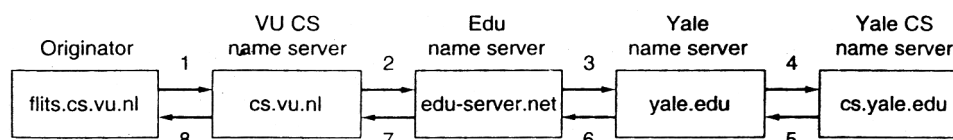


Fig.: How a resolver looks up a remote name in eight steps

Let us suppose the local name server has never had a query for this domain before and knows nothing about it. It may ask a few other nearby name servers, but if none of them know, it sends a UDP packet to the server for edu given in its database ee Fig. edu-server.net. It is unlikely that this server knows

the address of linda.cs.yale.edu. and probably does not know cs.yale.edu either, but it must know all of its own children, so it forwards the request to the name server for yale.edu (step 3). In turn, this one forwards the request to cs.yale.edu (step 4), which must have the authoritative resource records. Since each request is from a client to a server, the resource record requested works its way back in steps 5 through 8.

Q7. Discuss about the components of e-mail.

Ans :

(Imp.)

Email is a service which allows us to send the message in electronic mode over the internet. It offers an efficient, inexpensive and real time mean of distributing information among people.

E-Mail Address

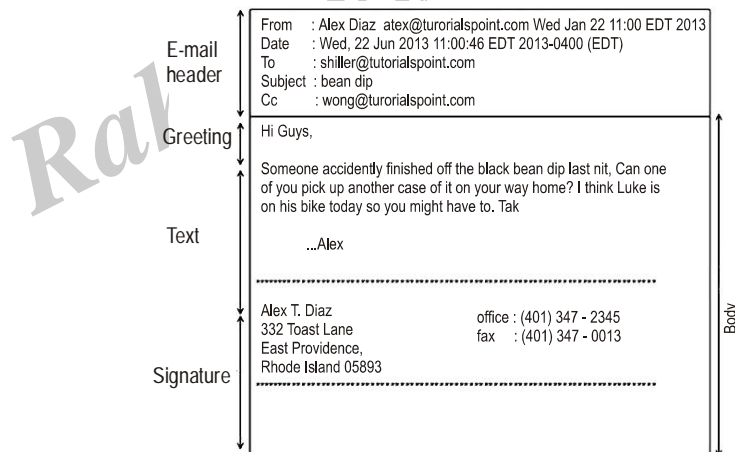
Each user of email is assigned a unique name for his email account. This name is known as E-mail address. Different users can send and receive messages according to the e-mail address.

Mail is generally of the form username@ domainname. For example, webmaster@tutorials .com is an e-mail address where webmaster is username and tutorials.com is domain name.

- The username and the domain name are separated by @ (at) symbol.
- E-mail addresses are not case sensitive.
- Spaces are not allowed in e-mail address.

E-mail Message Components

E-mail message comprises of different components: E-mail Header, Greeting, Text, and Signature. These components are described in the following diagram:



E-mail Header

The first five lines of an E-mail message is called E-mail header. The header part comprises of following fields:

- **FROM:** The **From** field indicates the sender's address i.e. who sent the e-mail.
- **DATE:** The **Date** field indicates the date when the e-mail was sent.
- **TO:** The **To** field indicates the recipient's address i.e. to whom the e-mail is sent.
- **SUBJECT:** The **Subject** field indicates the purpose of e-mail. It should be precise and to the point.

- **CC:** **CC** stands for Carbon copy. It includes those recipient addresses whom we want to keep informed but not exactly the intended recipient.
- **BCC:** **BCC** stands for Black Carbon Copy. It is used when we do not want one or more of the recipients to know that someone else was copied on the message.
- **GREETING:** Greeting is the opening of the actual message. Eg. Hi Sir or Hi Guys etc.
- **TEXT:** It represents the actual content of the message.
- **SIGNATURE:** This is the final part of an e-mail message. It includes Name of Sender, Address, and Contact Number.

Advantages

E-mail has proved to be powerful and reliable medium of communication. Here are the benefits of **E-mail**:

Reliable

Many of the mail systems notify the sender if e-mail message was undeliverable.

- **Convenience:** There is no requirement of stationary and stamps. One does not have to go to post office. But all these things are not required for sending or receiving an mail.
- **Speed:** E-mail is very fast. However, the speed also depends upon the underlying network.
- **Inexpensive:** The cost of sending e-mail is very low.
- **Printable:** It is easy to obtain a hardcopy of an e-mail. Also an electronic copy of an e-mail can also be saved for records.
- **Global:** E-mail can be sent and received by a person sitting across the globe.
- **Generality:** It is also possible to send graphics, programs and sounds with an e-mail.

Disadvantages

Apart from several benefits of E-mail, there also exist some disadvantages as discussed below:

- **Forgery:** E-mail doesn't prevent from forgery, that is, someone impersonating the sender, since sender is usually not authenticated in any way.
- **Overload:** Convenience of E-mail may result in a flood of mail.
- **Misdirection:** It is possible that you may send e-mail to an unintended recipient.
- **Junk:** Junk emails are undesirable and inappropriate emails. Junk emails are sometimes referred to as spam.
- **No Response:** It may be frustrating when the recipient does not read the e-mail and respond on a regular basis.

Q8. Explain the working of e-mail System.

Ans :

E-mail system comprises of the following three components:

- **Mailer:** It is also called mail program, mail application or mail client. It allows us to manage, read and compose e-mail.
- **Mail Server:** The function of mail server is to receive, store and deliver the email. It is must for mail servers to be sunning all the time because if it crashes or is down, email can be lost.
- **Mailboxes:** Mailbox is generally a folder that contains emails and information about them.

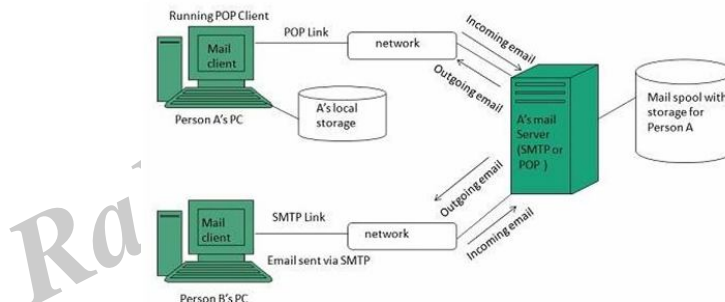
Working of E-mail

Email working follows the client server approach. In this client is the mailer i.e. the mail application or mail program and server is a device that manages emails.

Following example will take you through the basic steps involved in sending and receiving emails and will give you a better understanding of working of email system:

- Suppose person A wants to send an email message to person B.
- Person A composes the messages using a mailer program i.e. mail client and then select Send option.
- The message is routed to **Simple Mail Transfer Protocol** to person B's mail server.
- The mail server stores the email message on disk in an area designated for person B.
- The disk space area on mail server is called mail spool.
- Now, suppose person B is running a POP client and knows how to communicate with B's mail server.
- It will periodically poll the POP server to check if any new email has arrived for B. As in this case; person B has sent an email for person B, so email is forwarded over the network to B's PC. This message is now stored on person B's PC.

The following diagram gives pictorial representation of the steps discussed above:



Q9. List and Explain different types of protocols used in e-mail system.

Ans :

E-mail Protocols are set of rules that help the client to properly transmit the information to or from the mail server. Here in this tutorial, we will discuss various protocols such as SMTP, POP, and IMAP.

SMTP: SMTP stands for Simple Mail Transfer Protocol. It was first proposed in 1982. It is a standard protocol used for sending e-mail efficiently and reliably over the internet.

Key Points

- SMTP is application level protocol.
- SMTP is connection oriented protocol.
- SMTP is text based protocol.
- It handles exchange of messages between e-mail servers over TCP/IP network.

- Apart from transferring e-mail, SMTP also provides notification regarding incoming mail.
- When you send e-mail, your e-mail client sends it to your e-mail server which further contacts the recipient mail server using SMTP client.
- These SMTP commands specify the sender's and receiver's e-mail address, along with the message to be send.
- The exchange of commands between servers is carried out without intervention of any user.
- In case, message cannot be delivered, an error report is sent to the sender which makes SMTP a reliable protocol.

SMTP Commands

The following table describes some of the SMTP commands:

S. No.	Command Description
1.	HELLO: This command initiates the SMTP conversation.
2.	EHELLO: This is an alternative command to initiate the conversation. ESMTP indicates that the sender server wants to use extended SMTP protocol.
3.	MAIL FROM: This indicates the sender's address.
4.	RCPT TO: It identifies the recipient of the mail. In order to deliver similar message to multiple users this command can be repeated multiple times.
5.	SIZE: This command let the server know the size of attached message in bytes.
6.	DATA: The DATA command signifies that a stream of data will follow. Here stream of data refers to the body of the message.
7.	QUIT: This commands is used to terminate the SMTP connection.
8.	VERIFY: This command is used by the receiving server in order to verify whether the given username is valid or not.
9.	EXPN: It is same as VRFY, except it will list all the users name when it used with a distribution list.

IMAP: IMAP stands for Internet Mail Access Protocol. It was first proposed in 1986. There exist five versions of IMAP as follows:

1. Original IMAP
2. IMAP2
3. IMAP3
4. IMAP2bis
5. IMAP4

Key Points

- IMAP allows the client program to manipulate the e-mail message on the server without downloading them on the local computer.
- The e-mail is hold and maintained by the remote server.
- It enables us to take any action such as downloading, delete the mail without reading the mail. It enables us to create, manipulate and delete remote message folders called mail boxes.
- IMAP enables the users to search the e-mails.
- It allows concurrent access to multiple mailboxes on multiple mail servers.

IMAP Commands

The following table describes some of the IMAP commands:

S.No.	Command Description
1.	IMAP_LOGIN : This command opens the connection.
2.	CAPABILITY : This command requests for listing the capabilities that the server supports.
3.	NOOP : This command is used as a periodic poll for new messages or message status updates during a period of inactivity.
4.	SELECT : This command helps to select a mailbox to access the messages.
5.	EXAMINE : It is same as SELECT command except no change to the mailbox is permitted.
6.	CREATE : It is used to create mailbox with a specified name.
7.	DELETE : It is used to permanently delete a mailbox with a given name.
8.	RENAME : It is used to change the name of a mailbox.
9.	LOGOUT : This command informs the server that client is done with the session. The server must send BYE untagged response before the OK response and then close the network connection.

POP

POP stands for Post Office Protocol. It is generally used to support a single client. There are several versions of POP but the POP 3 is the current standard.

Key Points

- POP is an application layer internet standard protocol.
- Since POP supports offline access to the messages, thus requires less internet usage time.
- POP does not allow search facility.
- In order to access the messaged, it is necessary to download them.
- It allows only one mailbox to be created on server.
- It is not suitable for accessing non mail data.
- POP commands are generally abbreviated into codes of three or four letters. Eg. STAT.

POP Commands

The following table describes some of the POP commands:

S.No.	Command Description
1.	LOGIN: This command opens the connection.
2.	STAT: It is used to display number of messages currently in the mailbox.
3.	LIST: It is used to get the summary of messages where each message summary is shown.
4.	RETR: This command helps to select a mailbox to access the messages.
5.	DELE: It is used to delete a message.
6.	RSET: It is used to reset the session to its initial state.
7.	QUIT: It is used to log off the session. ³

Q10. State the differences between POP and IMAP.

Ans :

(Imp.)

S.No.	POP	IMAP
1.	Generally used to support single client.	Designed to handle multiple clients.
2.	Messages are accessed offline.	Messages are accessed online although it also supports offline mode.
3.	POP does not allow search facility.	It offers ability to search emails.
4.	All the messages have to be downloaded.	It allows selective transfer of messages to the client.
5.	Only one mailbox can be created on the server.	Multiple mailboxes can be created on the server.
6.	Not suitable for accessing non-mail data.	Suitable for accessing non-mail data i.e. attachment.
7.	POP commands are generally abbreviated into codes of three or four letters. Eg. STAT.	IMAP commands are not abbreviated, they are full. Eg. STATUS.
8.	It requires minimum use of server resources.	Clients are totally dependent on server.
9.	Mails once downloaded cannot be accessed from some other location.	Allows mails to be accessed from multiple locations.
10.	The e-mails are not downloaded automatically.	Users can view the headings and sender of e-mails and then decide to download.
11.	POP requires less internet usage time.	IMAP requires more internet usage time.

5.3.2 SMTP

Q11. How Simple Mail Transfer Protocol (SMTP) works?

Ans :

(Imp.)

Simple Mail Transfer Protocol (SMTP) is based on end-to-end message delivery. An Simple Mail Transfer Protocol (SMTP) client contacts the destination host's Simple Mail Transfer Protocol (SMTP) server on well-known port 25, to deliver the mail. The client then waits for the server to send a 220 READY FOR MAIL message. Upon receipt of the 220 message, the client sends a HELO command. The server then responds with a "250 Requested mail action okay" message.

After this, the mail transaction will begin with a MAIL command that gives the sender identification as well as a FROM: field that contains the address to which errors should be reported.

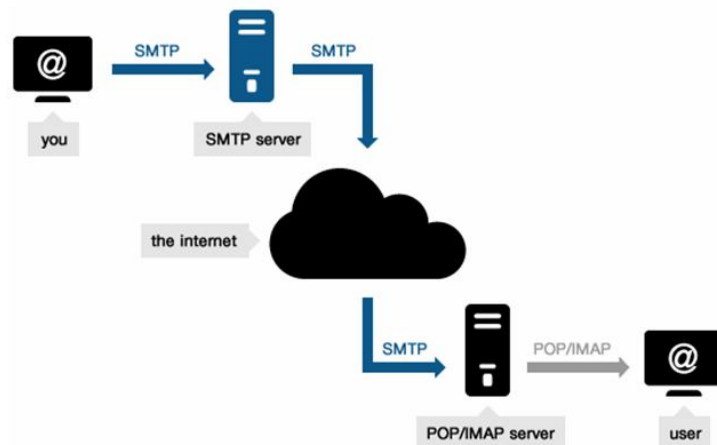
After a successful MAIL command, the sender issues a series of RCPT commands those identify recipients of the mail message. The receiver will acknowledge each RCPT command by sending 250 OK or by sending the error message 550 No such user here.

After all RCPT commands have been acknowledged, the sender issues a DATA command to inform the receiver that the sender is ready to transfer a complete mail message. The receiver responds with message 354 Start mail command with an ending sequence that the sender should use to terminate the message data. The termination sequence consists of 5 characters: carriage return, line feed, period, carriage return, and line feed (<CRLF>.<CRLF>).

The client now sends the data line by line, ending with the 5-character sequence <CRLF>.<CRLF> line, upon which the receiver will acknowledge with a 250 OK, or an appropriate error message if anything went wrong.

After the sending is completed, the client can follow any of these actions.

- **Terminate Session:** If the current Simple Mail Transfer Protocol (SMTP) client has no more messages to send, the connection can be closed with a QUIT command, which will be answered with a 221 Service closing transmission channel reply.
- **Exchange Roles:** If the current Simple Mail Transfer Protocol (SMTP) client has no more messages to send, but is ready to receive any messages from the current Simple Mail Transfer Protocol (SMTP) server, it can issue the TURN command. Now the SMTP client and the SMTP server will switch their role of sender/receiver, and the sender (previous receiver) can now send messages by issuing a MAIL command.
- **Send Another Mail:** If the Simple Mail Transfer Protocol (SMTP) client (sender) has another message to send, it can issue a new MAIL command.



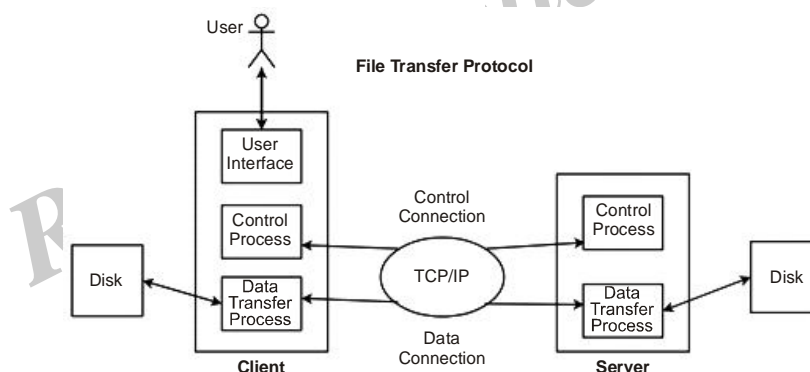
5.3.3 File Transfer Protocol (FTP)

Q12. What is FTP - File Transfer Protocol?

Ans :

(Imp.)

File Transfer Protocol(FTP) is an application layer protocol which moves files between local and remote file systems. It runs on the top of TCP, like HTTP. To transfer a file, 2 TCP connections are used by FTP in parallel: control connection and data connection.



Control connection

For sending control information like user identification, password, commands to change the remote directory, commands to retrieve and store files etc., FTP makes use of control connection. Control connection is initiated on port number 21.

Data connection

For sending the actual file, FTP makes use of data connection. Data connection is initiated on port number 20.

FTP sends the control information out-of-band as it uses a separate control connection. Some protocols send their request and response header lines and the data in the same TCP connection. For this reason, they are said to send their control information in-band. HTTP and SMTP are such examples.

FTP Session

When a FTP session is started between a client and a server, the client initiates a control TCP connection with the server side. The client sends the control information over this. When the server receives this, it initiates a data connection to the client side. Only one file can be sent over one data connection. But the control connection remains active throughout the user session. As we know HTTP is stateless i.e. it does not have to keep track of any user state. But FTP needs to maintain a state about its user throughout the session.

Data Structures

FTP allows three types of data structures

1. **File Structure:** In file-structure there is no internal structure and the file is considered to be a continuous sequence of data bytes.
2. **Record Structure:** In record-structure the file is made up of sequential records.
3. **Page Structure:** In page-structure the file is made up of independent indexed pages.

FTP Commands

Some of the FTP commands are :

- **USER:** This command sends the user identification to the server.
- **PASS:** This command sends the user password to the server.
- **CWD:** This command allows the user to work with a different directory or dataset for file storage or retrieval without altering his login or accounting information.
- **RMD:** This command causes the directory specified in the path-name to be removed as a directory.
- **MKD:** This command causes the directory specified in the path name to be created as a directory.
- **PWD:** This command causes the name of the current working directory to be returned in the reply.
- **RETR:** This command causes the remote host to initiate a data connection and to send the requested file over the data connection.
- **STOR:** This command causes to store a file into the current directory of the remote host.
- **LIST:** Sends a request to display the list of all the files present in the directory.
- **ABOR:** This command tells the server to abort the previous FTP service command and any associated transfer of data.
- **QUIT:** This command terminates a USER and if file transfer is not in progress, the server closes the control connection.

FTP Replies – Some of the FTP replies are

- 200 → Command okay.
- 530 → Not logged in.
- 331 → User name okay, need password.
- 225 → Data connection open; no transfer in progress.

- 221 → Service closing control connection.
- 551 → Requested action aborted: page type unknown.
- 502 → Command not implemented.
- 503 → Bad sequence of commands.
- 504 → Command not implemented for that parameter.

Anonymous FTP

Anonymous FTP is enabled on some sites whose files are available for public access. A user can access these files without having any username or password. Instead, username is set to anonymous and password to guest by default. Here, the user access is very limited. For example, the user can be allowed to copy the files but not to navigate through directories.

5.3.4 HTTP

Q13. What is HTTP? Explain the features of HTTP.

Ans :

The Hypertext Transfer Protocol (HTTP) is an application-level protocol for distributed, collaborative, hypermedia information systems. This is the foundation for data communication for the World Wide Web (i.e. internet) since 1990. HTTP is a generic and stateless protocol which can be used for other purposes as well using extensions of its request methods, error codes, and headers.

Basically, HTTP is a TCP/IP based communication protocol, that is used to deliver data (HTML files, image files, query results, etc.) on the World Wide Web. The default port is TCP 80, but other ports can be used as well. It provides a standardized way for computers to communicate with each other. HTTP specification specifies how clients' request data will be constructed and sent to the server, and how the servers respond to these requests.

Basic Features

There are three basic features that make HTTP a simple but powerful protocol:

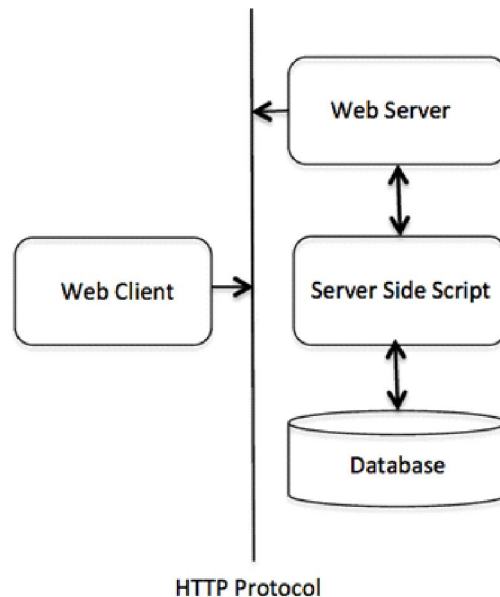
- **HTTP is connectionless:** The HTTP client, i.e., a browser initiates an HTTP request and after a request is made, the client disconnects from the server and waits for a response. The server processes the request and re-establishes the connection with the client to send a response back.
- **HTTP is media independent:** It means, any type of data can be sent by HTTP as long as both the client and the server know how to handle the data content. It is required for the client as well as the server to specify the content type using appropriate MIME-type.
- **HTTP is stateless:** As mentioned above, HTTP is connectionless and it is a direct result of HTTP being a stateless protocol. The server and client are aware of each other only during a current request. Afterwards, both of them forget about each other. Due to this nature of the protocol, neither the client nor the browser can retain information between different requests across the web pages.

HTTP/1.0 uses a new connection for each request/response exchange, where as HTTP/1.1 connection may be used for one or more request/response exchanges.

Q14. Explain the Architecture of HTTP.

Ans :

The following diagram shows a very basic architecture of a web application and depicts where HTTP sits:



The HTTP protocol is a request/response protocol based on the client/server based architecture where web browsers, robots and search engines, etc. act like HTTP clients, and the Web server acts as a server.

Client

The HTTP client sends a request to the server in the form of a request method, URI, and protocol version, followed by a MIME-like message containing request modifiers, client information, and possible body content over a TCP/IP connection.

Server

The HTTP server responds with a status line, including the message's protocol version and a success or error code, followed by a MIME-like message containing server information, entity meta information, and possible entity-body content.

The Basic Characteristics of HTTP (Hyper Text Transfer Protocol):

- It is the protocol that allows web servers and browsers to exchange data over the web.
- It is a request response protocol.
- It uses the reliable TCP connections by default on TCP port 80.

It is stateless means each request is considered as the new request. In other words, server doesn't recognize the user by default.

Q15. How to Check HTTP Request and Response?*Ans :***(Imp.)**

The request sends by the computer to a web server that contains all sorts of potentially interesting information is known as HTTP requests.

The HTTP client sends the request to the server in the form of request message which includes following information:

- The Request-line
- The analysis of source IP address, proxy and port
- The analysis of destination IP address, protocol, port and host
- The Requested URI (Uniform Resource Identifier)
- The Request method and Content
- The User-Agent header
- The Connection control header
- The Cache control header

The HTTP request method indicates the method to be performed on the resource identified by the **Requested URI (Uniform Resource Identifier)**. This method is case-sensitive and should be used in uppercase.

The HTTP request methods are :

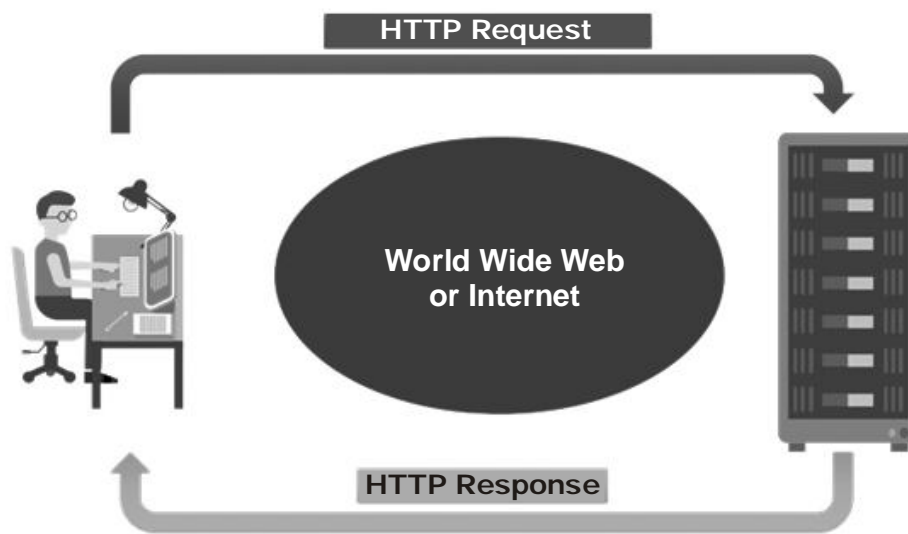
HTTP Request	Description
GET	Asks to get the resource at the requested URL.
POST	Asks the server to accept the body info attached. It is like GET request with extra info sent with the request.
HEAD	Asks for only the header part of whatever a GET would return. Just like GET but with no body.
TRACE	Asks for the loopback of the request message, for testing or troubleshooting.
PUT	Says to put the enclosed info (the body) at the requested URL.
DELETE	Says to delete the resource at the requested URL.
OPTIONS	Asks for a list of the HTTP methods to which the thing at the request URL can respond

A simple response from the server contains the following components :

- HTTP Status Code (For example HTTP/1.1 301 Moved Permanently, means the requested resource was permanently moved and redirecting to some other resource).

- Headers (Example – Content-Type: html)
- An empty line.
- A message body which is optional.

All the lines in the server response should end with a carriage return and line feed. Similar to request, the empty line in a response also should only have carriage return and line feed without any spaces.



LAB PRACTICALS

1. Understanding and using of commands like ifconfig, netstat, ping, arp, telnet, ftp, finger, traceroute, whois.

Ans :

1. **Ifconfig**

The command ifconfig stands for interface configurator. This command enables us to initialize an interface, assign IP address, enable or disable an interface. It display route and network interface.

You can view IP address, MAC address and MTU (Maximum Transmission Unit) with ifconfig command

Syntax: ifconfig

Assigning IP address and Gateway

You can assign IP address and Gateway to an interface but these settings will be disabled after system reboot.

Syntax: ifconfig eth0 <address> netmask <address>

2. **netstat**

Linux netstat command stands for Network statistics. It displays information about different interface statistics, including open sockets, routing tables, and connection information. Further, it can be used to displays all the socket connections (including TCP, UDP). Apart from connected sockets, it also displays the sockets that are pending for connections. It is a handy tool for network and system administrators.

Syntax : netstat

Options : It supports multiple command-line options to print information about the Linux networking subsystem. The output is controlled by the first argument. Let's see the list of the first arguments:

(none) : If no option is specified, it will execute the default command that displays a list of open sockets of all configured address families.

- route, -r : It is used to print the kernel routing tables. The "netstat -r" command and "route -e" command will produce the same output.

- groups, -g : It is used to display multicast group membership information different IP versions (Ipv4 and IPV6).

-interfaces, -i : It is used to display all network interfaces.

-masquerade, -M : It displays masqueraded connections.

-statistics, -s : This option displays the summary statistics for each protocol

Installation of the netstat command

```
sudo apt install net-tools
```

Examples of the netstat command

Display All Connections

The '-a' option is used to display all the existing connections. Execute the netstat command as follows:

```
netstat-a
```

The above command will list all the existing connections

Display only TCP or UDP Connections

We can list only the TCP or UDP connections. To display only the TCP connection, execute the command with the 't' option as follows:

```
netstat -at
```

The above command will list all the TCP connections.

3. Ping

Linux ping command stands for (Packet Internet Groper). It checks connectivity between two nodes to see if a server is available. It sends ICMP ECHO_REQUEST packets to network hosts and displays the data on the remote server's response. It checks if a remote host is up, or that network interfaces can be reached. Further, it is used to check if a network connection is available between two devices. It is also handy tool for checking your network connection and verifying network issues.

Ping command keeps executing and sends the packet until you interrupt.

To stop the execution, press "**CTRL**+**C**" keys.

Syntax: ping <option> <destination>

Options:

The ping command supports the following command-line options:

-4: It used to use IPv4 only.

-6: It is used to use IPv6 only.

-a: It is used for the audible ping.

-A: It is used for an adaptive ping.

-b: It is used to ping a broadcast address.

-B: It is used for not changing the source address of probes.

-c count: It is used to stop after sending count ECHO_REQUEST packets.

d: It is used to set the SO_DEBUG option on the socket being used.

-D: It is used to print the timestamp before each line.

-f: It stands for flood ping. It prints a period for every sent ECHO_REQUEST and backspaces for every received ECHO_REPLY.

-F flow label: It is used for IPv6 only. It allocates a 20-bit flow label (in hex) on echo request packets.

-h: It is used to display the help manual having a brief description of the usage and support options.

Examples of the ping Command

Ping using DNS

To check the connectivity using DNS, execute the command as follows:

```
ping <destination>
```

Ping using IP address

We can use the IP address instead of DNS with a ping command. To use the IP address to ping a destination, execute the command as follows:

```
ping <IP address>
```

Consider the below command:

```
ping 2.2.2.2
```

The above command will check the connectivity with the given IP address.

4. **Arp**

The command arp stands for **A**ddress **R**esolution **P**rotocol. It allows us to view or add content into kernel's ARP table.

Syntax: arp

5. **telnet**

In Linux, the `telnet` command is used to create a remote connection with a system over a TCP/IP network. It allows us to administrate other systems by the terminal. We can run a program to conduct administration.

The syntax for the telnet is as Follows:

```
telnet hostname/IP address
```

Install Telnet on Linux (Ubuntu)

Installing telnet on Linux is a straight forward process. We can install it by executing the following commands:

```
sudo apt update
```

The above command will prompt for the user password. Type the password and press **ENTER** key; it will start a daemon process and take a while to update your system.

To install the telnet, execute the below command:

```
sudo apt install telnetd -y
```

The above command will install the required package for the telnet protocol.

6. **ftp:Internet file transfer program**

tp is the user interface to the Internet standard File Transfer Protocol. The program allows a user to transfer files to and from a remote network site.

OPTIONS

TAG	DESCRIPTION
-A'	Use active mode for data transfers. This is useful for transmissions to servers which do not support passive connections (for whatever reason.).
-p'	Use passive mode for data transfers. Allows use of ftp in environments where a firewall prevents connections from the outside world back to the client machine. Requires that the ftp server support the PASV command. This is the default now for all clients (ftp and pftp) due to security concerns using the PORT transfer mode. The flag is kept for compatibility only and has no effect anymore.
-i'	Turns off interactive prompting during multiple file transfers.
-n'	Restrains ftp from attempting "auto-login" upon initial connection. If auto-login is enabled, ftp will check the .netrc (see netrc(5)) file in the user's home directory for an entry describing an account on the remote machine. If no entry exists, ftp will prompt for the remote machine login name (default is the user identity on the local machine), and, if necessary, prompt for a password and an account with which to login.
-e'	Disables command editing and history support, if it was compiled into the ftp executable. Otherwise, does nothing
-g'	Disables file name globbing
-m'	The default requires that ftp explicitly binds to the same interface for the data channel as the control channel in passive mode. Useful on multi-homed clients. This option disables this behavior. Files cannot be extracted from a thin ftpchive.
-v'	Verbose option forces ftp to show all responses from the remote server, as well as report on data transfer statistics
-d'	Enables debugging.

Examples

To see help of all available commands in ftp

```
$ ftp
```

```
ftp> help
```

Output

```
$ ftp
```

```
ftp> help
```

7. Finger

Finger command is a user information lookup command which gives details of all the users logged in. This tool is generally used by system administrators. It provides details like login name, user name, idle time, login time, and in some cases their email address even.

To install finger tool use the following commands as per your Linux distribution.

In case of Debian/Ubuntu

```
$sudo apt-get install finger
```

In case of CentOS/RedHat

```
$sudo yum install finger
```

8. Traceroute

Linux traceroute command is a network troubleshooting utility that helps us determine the number of hops and packets traveling path required to reach a destination. It is used to display how the data transmitted from a local machine to a remote machine. Loading a web page is one of the common examples of the traceroute. A web page loading transfers data through a network and routers. The traceroute can display the routes, IP addresses, and hostnames of routers over a network. It can be useful for diagnosing network issues.

Syntax

traceroute [OPTION...] HOST

Options

The following command-line options are supported by the traceroute command:

-f, -first-hop=NUM: It is used to set the initial hop distance.

-g, -gateways=GATES: It is used to display a list of gateways for loose source routing.

-I, -icmp: It is specified to use ICMP ECHO as a probe.

-m, -max-hop=NUM: It is used to set maximal hop count, the default is 64.

-M, -type=METHOD: It specifies the METHOD (icmp or udp) for traceroute operations, the default method is udp.

-p, -port=PORT: It is defined to use destination PORT port, the default PORT is 33434.

-q, -tries=NUM: It is used to forward NUM probe packets per hop, the default is 3.

-resolve-hostnames: It is used to resolve the hostnames.

-t, -tos=NUM: It is used to set the type of service (TOS) to NUM.

-w, -wait=NUM: It is used to wait in seconds for a response, the default is 3.

Install the traceroute Command

The traceroute is not a default utility of the Linux system. To use the traceroute, we need to install it manually. To install it, execute one of the following commands:

```
sudo apt install inetutils-traceroute
```

```
sudo apt install traceroute
```

10. whois

The whois command displays information about a website's record. You may get all the information about a website regarding its registration and owner's information.

Syntax:

whois <websiteName>

Example:

whois javat.com

2. Socket Programming: Implementation of Connection-Oriented Service using standard ports.

Ans :

TCP Server

```
#include <netdb.h>
#include <netinet/in.h>
#include <stdlib.h>
#include <string.h>
#include <sys/socket.h>
#include <sys/types.h>
#define MAX 80
#define PORT 8080
#define SA struct sockaddr
// Function designed for chat between client and server.
void func(int sockfd)
{
    char buff[MAX];
    int n;
    // infinite loop for chat
    for(;;) {
        bzero(buff, MAX); // read the message from client and copy it in buffer
        read(sockfd, buff, sizeof(buff)); // print buffer which contains the client contents
        printf("From client: %s To client : ", buff);
        bzero(buff, MAX);
        n = 0; // copy server message in the buffer
        while((buff[n++] = getchar()) != '\n'); // and send that buffer to client
        write(sockfd, buff, sizeof(buff)); // if msg contains "Exit" then server exit and chat ended.
        if(strncmp("exit", buff, 4) == 0) {
            printf("Server Exit...");
        }
        break;
    }
}
// Driver function
int main()
{
```

```
int sockfd, connfd, len;
struct sockaddr_in servaddr, cli;           // socket create and verification
sockfd = socket(AF_INET, SOCK_STREAM, 0);
if(sockfd == -1) {
    printf("socket creation failed...\n");
    exit(0);
}
else
    printf("Socket successfully created..\n");
bzero(&servaddr, sizeof(servaddr));        // assign IP, PORT
servaddr.sin_family = AF_INET;
servaddr.sin_addr.s_addr = htonl(INADDR_ANY);
servaddr.sin_port = htons(PORT);          // Binding newly created socket to given IP and verification
if((bind(sockfd, (SA*)&servaddr, sizeof(servaddr))) != 0) {
    printf("socket bind failed...\n");
    exit(0);
}
else
    printf("Socket successfully binded..\n"); // Now server is ready to listen and verification
if((listen(sockfd, 5)) != 0) {
    printf("Listen failed...\n");
    exit(0);
}
else
    printf("Server listening..\n");
len = sizeof(cli);                          // Accept the data packet from client and verification
connfd = accept(sockfd, (SA*)&cli, &len);
if(connfd < 0) {
    printf("server accept failed...\n");
    exit(0);
}
else
    printf("server accept the client...\n"); // Function for chatting between client and server
func(connfd);                               // After chatting close the socket
close(sockfd);
}
```

TCP Client

// Write CPP code here

```
#include <netdb.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <sys/socket.h>
#define MAX 80
#define PORT 8080
#define SA struct sockaddr
void func(int sockfd)
{
    char buff[MAX];
    int n;
    for(;;) {
        bzero(buff, sizeof(buff));
        printf("Enter the string : ");
        n = 0;
        while((buff[n++] = getchar()) != '\n');
        write(sockfd, buff, sizeof(buff));
        bzero(buff, sizeof(buff));
        read(sockfd, buff, sizeof(buff));
        printf("From Server : %s", buff);
        if((strcmp(buff, "exit", 4)) == 0) {
            printf("Client Exit...");
            break;
        }
    }
}

int main()
{
    int sockfd, connfd;
    struct sockaddr_in servaddr, cli;
    // socket create and verification
    sockfd = socket(AF_INET, SOCK_STREAM, 0);
    if(sockfd == -1) {
        printf("socket creation failed...");
        exit(0);
    }
```

```
else
printf("Socket successfully created..");
bzero(&servaddr, sizeof(servaddr));
// assign IP, PORT
servaddr.sin_family = AF_INET;
servaddr.sin_addr.s_addr
    = inet_addr("127.0.0.1");
servaddr.sin_port = htons(PORT);
// connect the client socket to server socket
if(connect(sockfd, (SA*)&servaddr, sizeof(servaddr)) != 0) {
printf("connection with the server failed...");
exit(0);
}
else
printf("connected to the server..");
// function for chat
func(sockfd);
// close the socket
close(sockfd);
}
```

Compilation

Server side:

```
gcc server.c -o server
./server
```

Client side:

```
gcc client.c -o client
./client
```

Output

Server side:

Socket successfully created..

Socket successfully binded..

Server listening..

server accept the client...

From client: hi

To client : hello

From client: exit

To client : exit

Server Exit...

Client side:

Socket successfully created..

connected to the server..
Enter the string : hi
From Server : hello
Enter the string : exit
From Server : exit
Client Exit...

3. Implementation of Connection-Less Service using standard ports.

Ans :

UDPServer.c

// Server side implementation of UDP client-server model

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <string.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <arpa/inet.h>
#include <netinet/in.h>
#define PORT 8080
#define MAXLINE 1024

// Driver code
int main() {
    int sockfd;
    char buffer[MAXLINE];
    char *hello = "Hello from server";
    struct sockaddr_in servaddr, cliaddr;

    // Creating socket file descriptor
    if( (sockfd = socket(AF_INET, SOCK_DGRAM, 0)) < 0 ) {
        perror("socket creation failed");
        exit(EXIT_FAILURE);
    }

    memset(&servaddr, 0, sizeof(servaddr));
    memset(&cliaddr, 0, sizeof(cliaddr));

    // Filling server information
    servaddr.sin_family = AF_INET; // IPv4
    servaddr.sin_addr.s_addr = INADDR_ANY;
    servaddr.sin_port = htons(PORT);

    // Bind the socket with the server address
```

```
    if( bind(sockfd, (const struct sockaddr *)&servaddr,
        sizeof(servaddr)) < 0 )
    {
        perror("bind failed");
        exit(EXIT_FAILURE);
    }

    int len, n;
    n = recvfrom(sockfd, (char*)buffer, MAXLINE,
        MSG_WAITALL, ( struct sockaddr *) &cliaddr, &len);
    buffer[n] = '\0';
    printf("Client : %s", buffer);
    sendto(sockfd, (const char*)hello, strlen(hello),
        MSG_CONFIRM, (const struct sockaddr *) &cliaddr, len);
    printf("Hello message sent.");
    return 0;
}
```

UDPClient.c

// Client side implementation of UDP client-server model

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <string.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <arpa/inet.h>
#include <netinet/in.h>
#define PORT 8080
#define MAXLINE 1024
// Driver code
int main() {
    int sockfd;
    char buffer[MAXLINE];
    char* hello = "Hello from client";
    struct sockaddr_in servaddr;
    // Creating socket file descriptor
    if( (sockfd = socket(AF_INET, SOCK_DGRAM, 0)) < 0 ) {
        perror("socket creation failed");
        exit(EXIT_FAILURE);
    }
```



```

memset(&servaddr, 0, sizeof(servaddr));
// Filling server information
servaddr.sin_family = AF_INET;
servaddr.sin_port = htons(PORT);
servaddr.sin_addr.s_addr = INADDR_ANY;
intn, len;
sendto(sockfd, (constchar*)hello, strlen(hello),
MSG_CONFIRM, (conststructsockaddr *) &servaddr,
sizeof(servaddr));
printf("Hello message sent. ");
n = recvfrom(sockfd, (char*)buffer,
MAXLINE, MSG_WAITALL,
(structsockaddr *) &servaddr, &len);
buffer[n] = "\n";
printf("Server : %s", buffer);
close(sockfd);
return 0;
}

```

Output

\$./server

Client : Hello from client

Hello message sent.

\$./client

Hello message sent.

Server : Hello from server

4. Implementation of Connection-Oriented Iterative Echo-Server, date and time, character generation using user-defined ports.

Ans :

TCP ITERATIVE ECHO SERVER

TCP Echo Client

```

#include <stdlib.h>
#include <stdio.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <string.h>
#include <arpa/inet.h>
#define MAXLINE 4096 /*max text line length*/
#define SERV_PORT 3000 /*port*/
int
main(int argc, char **argv)
{
    int sockfd;

```

```

        struct sockaddr_in servaddr;
        char sendline[MAXLINE], recvline[MAXLINE];
        //basic check of the arguments
        //additional checks can be inserted
        if (argc != 2) {
            perror("Usage: TCPClient <IP address of the server>");
            exit(1);
        }
        //Create a socket for the client
        //If sockfd < 0 there was an error in the creation of the socket
        if ((sockfd = socket (AF_INET, SOCK_STREAM, 0)) < 0) {
            perror("Problem in creating the socket");
            exit(2);
        }
        //Creation of the socket
        memset(&servaddr, 0, sizeof(servaddr));
        servaddr.sin_family = AF_INET;
        servaddr.sin_addr.s_addr = inet_addr(argv[1]);
        servaddr.sin_port = htons(SERV_PORT); //convert to big-endian order
        //Connection of the client to the socket
        if (connect(sockfd, (struct sockaddr *) &servaddr, sizeof(servaddr)) < 0) {
            perror("Problem in connecting to the server");
            exit(3);
        }
        while (fgets(sendline, MAXLINE, stdin) != NULL) {
            send(sockfd, sendline, strlen(sendline), 0);
            if (recv(sockfd, recvline, MAXLINE, 0) == 0) {
                //error: server terminated prematurely
                perror("The server terminated prematurely");
                exit(4);
            }
            printf("%s", "String received from the server: ");
            fputs(recvline, stdout);
        }
        exit(0);
    }
}

```

TCP Iterative Server

```

#include <stdlib.h>
#include <stdio.h>
#include <sys/types.h>
#include <sys/socket.h>

```

```
#include <netinet/in.h>
#include <string.h>
#include <unistd.h>
#define MAXLINE 4096 /*max text line length*/
#define SERV_PORT 3000 /*port*/
#define LISTENQ 8 /*maximum number of client connections */
int main (int argc, char **argv)
{
    int listenfd, connfd, n;
    socklen_t clilen;
    char buf[MAXLINE];
    struct sockaddr_in cliaddr, servaddr;
    //creation of the socket
    listenfd = socket (AF_INET, SOCK_STREAM, 0);
    //preparation of the socket address
    servaddr.sin_family = AF_INET;
    servaddr.sin_addr.s_addr = htonl(INADDR_ANY);
    servaddr.sin_port = htons(SERV_PORT);
    bind (listenfd, (struct sockaddr *) &servaddr, sizeof(servaddr));
    listen (listenfd, LISTENQ);
    printf("%s\n", "Server running...waiting for connections.");
    for ( ; ; ) {
        clilen = sizeof(cliaddr);
        connfd = accept (listenfd, (struct sockaddr *) &cliaddr, &clilen);
        printf("%s\n", "Received request...");
        while ( (n = recv(connfd, buf, MAXLINE, 0)) > 0) {
            printf("%s", "String received from and resent to the client.");
            puts(buf);
            send(connfd, buf, n, 0);
        }
        if (n < 0) {
            perror("Read error");
            exit(1);
        }
        close(connfd);
    }
    //close listening socket
    close (listenfd);
}
```

**TCP DATE and TIME SERVER
SERVER**

```
#include <stdio.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <time.h>
#include <string.h>
#include <stdlib.h>
#define max 30
#define PORT 2100
int main()
{
    int sersoc, clisoc, conn, len, wri;
    char str[max];
    pid_t pid;
    time_t ticks;
    socklen_t clilen;
    struct sockaddr_in servaddr, cliaddr;
    servaddr.sin_family = AF_INET;
    servaddr.sin_port = htons(PORT);
    servaddr.sin_addr.s_addr = htonl(INADDR_ANY);
    if((sersoc = socket(AF_INET, SOCK_STREAM, 0)) < 0)
{
    perror("Socket Error");
    exit(0);
}

    if(bind(sersoc, (struct sockaddr *)&servaddr, sizeof(servaddr)) < 0)
    {
        perror("Bind Error");
        exit(0);
    }
    listen(sersoc, 10);
    for(;;)
    {
```

```

len=sizeof(cliaddr);
conn=(accept(sersoc,(struct sockaddr *) &clisoc,&len));
if((pid=fork())==0)
{
close(sersoc);
ticks=time(NULL);
strcpy(str,ctime(&ticks));
if(wri==(write(conn,str,sizeof(str),0))<0)
{
printf("Write Error");
exit(0);
}
close(conn);
exit(0);
}
close(conn);
}
close(sersoc);
return 0;
}

```

OUTPUT

FROM SERVER SIDE

\$ cc tcptimeserver.c

\$./a.out

FROM CLIENT SIDE

\$ cc tcptimeclient.c

\$./a.out 127.0.0.1

The Current Date and Time : Wed Mar 23 05:55:11 2011

\$

5. Implementation of Connectionless Iterative Echo-server, date and time, character.*Ans :*

generation using user-defined ports.

/*udp Server.c*/

#include <unistd.h>

#include <sys/socket.h>

#include <stdio.h>

#include <string.h>

#include <netinet/in.h>

```
#include <sys/types.h>
#include <errno.h>
#define MAXLINE 5555
void main(int argc, char *argv[])
{
    int sockfd, n;
    unsigned short port;
    socklen_t len;
    char mesg[1024];
    struct sockaddr_in servaddr, cliaddr;
    port = (unsigned short)atoi(argv[1]);
    if((sockfd = socket(AF_INET, SOCK_DGRAM, 0)) < 0)
        printf("error in socket creation");
    bzero(&servaddr, sizeof(servaddr));
    servaddr.sin_family = AF_INET;
    servaddr.sin_addr.s_addr = htonl(INADDR_ANY);
    servaddr.sin_port = htons(port);
    len = sizeof(servaddr);
    if(bind(sockfd, (struct sockaddr *)&servaddr, len) < 0)
        perror("\nbind");
    for(;;)
    {
        len = sizeof(cliaddr);
        n = recvfrom(sockfd, &mesg, MAXLINE, 0, (struct sockaddr *)&cliaddr, &len);
        sendto(sockfd, &mesg, n, 0, (struct sockaddr *)&cliaddr, len);
    }
}
```

Output:

/home/anil/~ cc udpserver.c

/home/anil/~ ./a.out 4455

/* open another new terminal for udp client program */

/* udpclient.c */

```
#include <stdio.h>
#include <unistd.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <string.h>
#include <netinet/in.h>
#include <errno.h>
#define MAXLINE 5566
void main(int argc, char **argv[])
```

```
{
    int sockfd,n,len;
    char mesg[1024];
    unsigned short port;
    char sendline[MAXLINE],recvline[MAXLINE];
    struct sockaddr_in servaddr;
    if((sockfd=socket(AF_INET,SOCK_DGRAM,0))<0)
    {
        perror("socket");
    }
    bzero(&servaddr,sizeof(servaddr));
    servaddr.sin_family=AF_INET;
    port=(unsigned short)atoi(argv[1]);
    servaddr.sin_port=htons(port);
    if((inet_pton(AF_INET,argv[2],&servaddr.sin_addr))<0)
    {
        perror("inet_pton");
    }
    len=sizeof(servaddr);
    while(fgets(sendline,MAXLINE,stdin)!=NULL)
    {
        sendto(sockfd,&sendline,strlen(sendline),0,(struct sockaddr *)&servaddr,len);
        n=recvfrom(sockfd,&recvline,MAXLINE,0,(struct sockaddr *)&servaddr,&len);
        recvline[n]=0;
        printf("\n %s",recvline);
    }
}
```

Output

```
/home/anil/~ cc udpclient.c
/home/anil/~ ./a.out 4455 127.0.0.1
hi
hi
hello
hello
```

6. Implementation of Connection-Oriented Concurrent Echo-server, date and time, character generation using user-defined ports.

Ans :

TCP Echo Client

```
#include <stdlib.h>
#include <stdio.h>
#include <sys/types.h>
```

```
#include <sys/socket.h>
#include <netinet/in.h>
#include <string.h>
#include <arpa/inet.h>
#define MAXLINE 4096 /*max text line length*/
#define SERV_PORT 3000 /*port*/
int
main(int argc, char **argv)
{
    int sockfd;
    struct sockaddr_in servaddr;
    char sendline[MAXLINE], recvline[MAXLINE];
    //basic check of the arguments
    //additional checks can be inserted
    if (argc !=2)
    {
        perror("Usage: TCPClient <IP address of the server>");
        exit(1);
    }

    //Create a socket for the client
    //If sockfd<0 there was an error in the creation of the socket
    if ((sockfd = socket (AF_INET, SOCK_STREAM, 0)) <0)
    {
        perror("Problem in creating the socket");
        exit(2);
    }

    //Creation of the socket
    memset(&servaddr, 0, sizeof(servaddr));
    servaddr.sin_family = AF_INET;
    servaddr.sin_addr.s_addr= inet_addr(argv[1]);
    servaddr.sin_port = htons(SERV_PORT); //convert to big-endian order
    //Connection of the client to the socket
    if (connect(sockfd, (struct sockaddr *) &servaddr, sizeof(servaddr))<0)
    {
        perror("Problem in connecting to the server");
        exit(3);
    }

    while (fgets(sendline, MAXLINE, stdin) != NULL)
    {
        send(sockfd, sendline, strlen(sendline), 0);
        if (recv(sockfd, recvline, MAXLINE,0) == 0)
```



```

    { //error: server terminated prematurely
      perror("The server terminated prematurely");
      exit(4);
    }
    printf("%s", "String received from the server: ");
    fputs(recvline, stdout);
  }
  exit(0);
}

```

TCP Concurrent Echo Server

```

#include <stdlib.h>
#include <stdio.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <string.h>
#include <unistd.h>
#define MAXLINE 4096 /*max text line length*/
#define SERV_PORT 3000 /*port*/
#define LISTENQ 8 /*maximum number of client connections*/

int main (int argc, char **argv)
{
    int listenfd, connfd, n;
    pid_t chldpid;
    socklen_t clen;
    char buf[MAXLINE];
    struct sockaddr_in cliaddr, servaddr;

    //Create a socket for the socket
    //If sockfd<0 there was an error in the creation of the socket
    if ((listenfd = socket (AF_INET, SOCK_STREAM, 0)) < 0) {
        perror("Problem in creating the socket");
        exit(2);
    }

    //preparation of the socket address
    servaddr.sin_family = AF_INET;
    servaddr.sin_addr.s_addr = htonl(INADDR_ANY);
    servaddr.sin_port = htons(SERV_PORT);

    //bind the socket
    bind (listenfd, (struct sockaddr*)&servaddr, sizeof(servaddr));

    //listen to the socket by creating a connection queue, then wait for clients

```

```

listen (listenfd, LISTENQ);
printf("%s\n","Server running...waiting for connections.");
for ( ; ; )
{
    clilen = sizeof(cliaddr);
    //accept a connection
    connfd = accept (listenfd, (struct sockaddr *) &cliaddr, &clilen);
    printf("%s\n","Received request...");
    if ((childpid = fork ()) == 0 ) { //if it's 0, it's child process
        printf ("%s\n","Child created for dealing with client requests");
        //close listening socket
        close (listenfd);
        while ( (n = recv(connfd, buf, MAXLINE,0)) > 0)
        {
            printf("%s","String received from and resent to the client:");
            puts(buf);
            send(connfd, buf, n, 0);
        }
        if (n < 0)
            printf("%s\n", "Read error");
        exit(0);
    }
    //close socket of the server
    close(connfd);
}
}

```

7. Program for connection-oriented Iterative Service in which server reverses the string sent by the client and sends it back.

Ans :

```

// C client code to send string to reverse
#include <arpa/inet.h>
#include <netinet/in.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <sys/socket.h>
#include <unistd.h>
#define PORT 8090
// Driver code
int main()
{

```

```
struct sockaddr_in address;
int sock = 0, valread;
struct sockaddr_in serv_addr;
char str[100];

printf("\nInput the string:");
scanf("%[^\n]s", str);
char buffer[1024] = { 0 };
// Creating socket file descriptor
if ((sock = socket(AF_INET, SOCK_STREAM, 0))
    < 0) {
    printf("\n Socket creation error \n");
    return -1;
}
memset(&serv_addr, '0', sizeof(serv_addr));
serv_addr.sin_family = AF_INET;
serv_addr.sin_port = htons(PORT);
// Convert IPv4 and IPv6 addresses from
// text to binary form 127.0.0.1 is local
// host IP address, this address should be
// your system local host IP address
if (inet_pton(AF_INET, "127.0.0.1", &serv_addr.sin_addr)
    <= 0) {
    printf("\nAddress not supported \n");
    return -1;
}
// connect the socket
if (connect(sock, (struct sockaddr*)&serv_addr,
    sizeof(serv_addr))
    < 0) {
    printf("\nConnection Failed \n");
    return -1;
}
int l = strlen(str);
// send string to server side
send(sock, str, sizeof(str), 0);
// read string sent by server
valread = read(sock, str, l);
printf("%s\n", str);
return 0;
}
```

```
// Server C code to reverse a
// string by sent from client
#include <netinet/in.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <sys/socket.h>
#include <unistd.h>
#define PORT 8090
// Driver code
int main()
{
    int server_fd, new_socket, valread;
    struct sockaddr_in address;
    char str[100];
    int addrlen = sizeof(address);
    char buffer[1024] = { 0 };
    char* hello = "Hello from server";    // Creating socket file descriptor
    if ((server_fd = socket(AF_INET, SOCK_STREAM, 0)) == 0)
    {
        perror("socket failed");
        exit(EXIT_FAILURE);
    }
    address.sin_family = AF_INET;
    address.sin_addr.s_addr = INADDR_ANY;
    address.sin_port = htons(PORT);    // Forcefully attaching socket to the port 8090
    if (bind(server_fd, (struct sockaddr*)&address, sizeof(address)) < 0)
    {
        perror("bind failed");
        exit(EXIT_FAILURE);
    }
    // puts the server socket in passive mode

    if (listen(server_fd, 3) < 0) {
        perror("listen");
        exit(EXIT_FAILURE);
    }
    if ((new_socket = accept(server_fd, (struct sockaddr*)&address, socklen_t*)&addrlen)) < 0)
    {
```

```
perror("accept");
exit(EXIT_FAILURE);
}
// read string send by client
valread = read(new_socket, str, sizeof(str));
int i, j, temp;
int l = strlen(str);
printf("\nString sent by client:%s\n", str); // loop to reverse the string
for (i = 0, j = l - 1; i < j; i++, j--)
{
    temp = str[i];
    str[i] = str[j];
    str[j] = temp;
}
// send reversed string to client
// by send system call
send(new_socket, str, sizeof(str), 0);
printf("\nModified string sent to client\n");
return 0;
}
```

Input : welcome

Output :emoclew

-
8. Program for connection-oriented Iterative service in which server changes the case of the strings sent by the client and sends back (Case Server).

Ans :

Server Code

```
#include <stdio.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <string.h>
#include <stdlib.h>
int main(){
    int welcomeSocket, newSocket, portNum, clientLen, nBytes;
    char buffer[1024];
    struct sockaddr_in serverAddr;
```

```
struct sockaddr_storage serverStorage;
socklen_t addr_size;
int i;
welcomeSocket = socket(PF_INET, SOCK_STREAM, 0);
portNum = 7891;
serverAddr.sin_family = AF_INET;
serverAddr.sin_port = htons(portNum);
serverAddr.sin_addr.s_addr = inet_addr("127.0.0.1");
memset(serverAddr.sin_zero, '\0', sizeof serverAddr.sin_zero);
bind(welcomeSocket, (struct sockaddr *) &serverAddr, sizeof(serverAddr));
if(listen(welcomeSocket,5)==0)
    printf("Listening\n");
else
    printf("Error\n");
addr_size = sizeof serverStorage;
/*loop to keep accepting new connections*/
while(1){
newSocket = accept(welcomeSocket, (struct sockaddr *) &serverStorage, &addr_size);
/*fork a child process to handle the new connection*/
if(!fork()){
    nBytes = 1;
    /*loop while connection is live*/
    while(nBytes!=0){
        nBytes = recv(newSocket,buffer,1024,0);
        for (i=0;i<nBytes-1;i++){
            buffer[i] = toupper(buffer[i]);
        }
        send(newSocket,buffer,nBytes,0);
    }
    close(newSocket);
    exit(0);
}
/*if parent, close the socket and go back to listening new requests*/
else{
    close(newSocket);
}
}
return 0;
}
```

Client Code

```
#include <stdio.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <string.h>
int main(){
    int clientSocket, portNum, nBytes;
    char buffer[1024];
    struct sockaddr_in serverAddr;
    socklen_t addr_size;
    clientSocket = socket(PF_INET, SOCK_STREAM, 0);
    portNum = 7891;
    serverAddr.sin_family = AF_INET;
    serverAddr.sin_port = htons(portNum);
    serverAddr.sin_addr.s_addr = inet_addr("127.0.0.1");
    memset(serverAddr.sin_zero, '\0', sizeof serverAddr.sin_zero);
    addr_size = sizeof serverAddr;
    connect(clientSocket, (struct sockaddr *) &serverAddr, addr_size);
    while(1){
        printf("Type a sentence to send to server:\n");
        fgets(buffer, 1024, stdin);
        printf("You typed: %s", buffer);
        nBytes = strlen(buffer) + 1;
        send(clientSocket, buffer, nBytes, 0);
        recv(clientSocket, buffer, 1024, 0);
        printf("Received from server: %s\n\n", buffer);
    }
    return 0;
}
```

OUTPUT

Type a sentence to send to server: welcome

Received from server: WELCOME

9. Program for Connection-Oriented Iterative service in which server calculates the netsalary of an employee based on the following details sent by the client i) basic ii) hra iii) da iv) pt v) epf vi) net-salary=basic+hra+da-pt-epf).

Ans ;

Server Side

```
#include <sys/types.h>
#include <sys/socket.h>
#include <stdio.h>
#include <netinet/in.h>
#include <unistd.h>
#include <string.h>
#include <arpa/inet.h>

void main()
{
    int b, sockfd, connfd, sin_size, l, n, len;
    if((sockfd=socket(AF_INET, SOCK_STREAM, 0)) > 0)
        printf("socket created successfully\n");           //socket creation
                                                    //printf("%d\n", sockfd);
                                                    //on success 0 otherwise -1

    struct sockaddr_in servaddr;
    struct sockaddr_in clientaddr;
    servaddr.sin_family = AF_INET;
    servaddr.sin_addr.s_addr = inet_addr("127.0.0.1");
    servaddr.sin_port = 6006;
    if((bind(sockfd, (struct sockaddr *)&servaddr, sizeof(servaddr))) == 0)
        printf("bind successful\n");                       // bind() assigns the
// address specified by addr to the socket referred to by the file
// descriptor sockfd. addrlen specifies the size, in bytes, of the
// address structure pointed to by addr. Traditionally, this operation is
// called "assigning a name to a socket".
        printf("%d\n", b);

    if((listen(sockfd, 5)) == 0)                           //listen for connections on a socket
        printf("listen successful\n");                     //printf("%d\n", l);

    sin_size = sizeof(struct sockaddr_in);
    if((connfd=accept(sockfd, (struct sockaddr *)&clientaddr, &sin_size)) > 0);
```



```

printf("accept sucessful\n");           //printf("%d\n",connfd);
                                        //variable to store values

float basic, da, hra, ta,pf,epf;
float net_salary;                       //input required fields
printf("Enter Basic Salary ($) : ");
read(connfd, &basic);
printf("Enter HRA ($) : ");
read(connfd,&hra);
printf("Enter TA ($) : ");
read(connfd, &ta);
printf("Enter Pf ($) : ");
read(connfd,&Pf);
printf("Enter EPf ($) : ");
read(connfd,&epf);                      //calculate net salary
net_salary = basic + da + hra + ta + pf+epf; //printing Net salary
printf("Net Salary is: $ %.02f\n");
write(connfd,&net_salary);
close(sockfd);
}

```

Client Side :

```

#include <sys/types.h>
#include <sys/socket.h>
#include <stdio.h>
#include <netinet/in.h>
#include <unistd.h>
#include <string.h>
#include <strings.h>
#include <arpa/inet.h>
//define bufsize 150
void main()
{
int b,sockfd,sin_size,con,n,len;
//char buff[256];
float basic, hra, da, ta, pf , epf, net_salary;
if((sockfd=socket(AF_INET,SOCK_STREAM,0))>0)
printf("socket created sucessfully\n");

```

```
//printf("%d\n", sockfd);
struct sockaddr_in servaddr;
servaddr.sin_family=AF_INET;
servaddr.sin_addr.s_addr=inet_addr("127.0.0.1");
servaddr.sin_port=6006;
sin_size = sizeof(struct sockaddr_in);
if((con=connect(sockfd,(struct sockaddr *) &servaddr, sin_size))==0);
    //initiate a connection on a socket
printf("connect sucessful\n");
write(sockfd,&basic);
write(sockfd,&hra);
write(sockfd,&da);
write(sockfd,&ta);
write(sockfd,&pf);
write(sockfd,&epf);
read(sockfd,&net_salary);
printf("Net salary from the server=%d\n",net_salary);
close(sockfd);
}
```

Server side Output

Enter Basic Salary (\$): 5000

Enter HRA (\$):2000

Enter TA (\$): 1500

Enter Pf (\$): 2000

Enter EPf (\$): 1200

Client side out put

Net salary from the server= 17000

10. Program for file access using sockets.

Ans :

```
#include <netinet/in.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <stdio.h>
#include <arpa/inet.h>
```

```
#include <string.h>
#include <fcntl.h>
main() {
    int sd, cd;
    charbuf[1000] = "", fname[10];
    struct sockaddr_in ser;
    sd = socket(AF_INET, SOCK_STREAM, 0);
    if (sd < 0)
        printf("SOCKET NOT CREATED\n");
    bzero(&ser, sizeof (struct sockaddr_in));
    ser.sin_family = AF_INET;
    ser.sin_port = htons(1101);
    inet_aton("localhost", &ser.sin_addr);
    int b = bind(sd, (struct sockaddr *) &ser, sizeof (ser));
    printf("BIND VALUE:%d\n", b);
    listen(sd, 5);
    for (;;) {
        cd = accept(sd, NULL, NULL);
        int pid = fork();
        if (pid == 0) {
            printf("accept value %d\n", cd);
            read(cd, buf, 1000);
            int fd = open(buf, O_RDONLY);
            read(fd, buf, 1000);
            write(cd, buf, strlen(buf));
            printf("MESSAGE FROM CLIENT:%s\n", buf);
            close(cd);
        }
    }
    close(sd);
}
```

CLIENT

```
#include <netinet/in.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <stdio.h>
#include <arpa/inet.h>
```

```
#include <string.h>
#include <fcntl.h>
main() {
    int sd, cd;
    charbuf[1000] = "", buf1[1000] = "";
    struct sockaddr_in ser;
    sd = socket(AF_INET, SOCK_STREAM, 0);
    if (sd < 0)
        printf("SOCKET NOT CREATED\n");
    bzero(&ser, sizeof (struct sockaddr_in));
    ser.sin_family = AF_INET;
    ser.sin_port = htons(1101);
    inet_aton("localhost", &ser.sin_addr);
    connect(sd, (struct sockaddr *) &ser, sizeof
        (ser));
    for (;;) {
        printf("ENTER THE MESSAGE");
        scanf("%s", buf);
        write(sd, buf, strlen(buf));
        read(sd, buf, 1000);
        printf("RECEIVED FROM SERVER%s\n", buf);
    }
    close(sd);
}
```

Sample Server Output:

cc udp.c - o udp

. / udp

BIND VALUE :0

accept value 4

MESSAGE FROM CLIENT :

```
#include <stdio.h>
```

```
main() {
    printf("hello");
}
```

Sample Client Output:

```
cc ftclient.c - o ftcli
```

```
./ ftcli
```

```
ENTER THE MESSAGE
```

```
hello.c
```

```
RECEIVED FROM SERVER
```

```
#include<stdio.h>
```

```
main() {
```

```
    printf("hello");
```

```
}
```

11. Program for Remote Command Execution using sockets.

Ans :

Server program

```
#include<stdio.h>
```

```
#include<stdlib.h>
```

```
#include<string.h>
```

```
#include<unistd.h>
```

```
#include<netinet/in.h>
```

```
#include<arpa/inet.h>
```

```
#include<sys/types.h>
```

```
#include<sys/socket.h>
```

```
#include<errno.h>
```

```
int main()
```

```
{
```

```
    int sd,acpt,len,bytes,port;
```

```
    char send[50],receiv[50];
```

```
    struct sockaddr_in serv,cli;
```

```
    if((sd=socket(AF_INET,SOCK_
        STREAM,0))<0)
```

```
{
```

```
    printf("Error in socket\n");
```

```
    exit(0);
```

```
}
```

```
    bzero(&serv,sizeof(serv));
```

```
printf("Enter the port number:");
scanf("%d",&port);
serv.sin_family=AF_INET;
serv.sin_port=htons(port);
serv.sin_addr.s_addr=htonl(INADDR_ANY);
if(bind(sd,(struct sockaddr *)&serv,sizeof(serv))<0)
{
printf("Error in bind\n");
exit(0);
}
if(listen(sd,3)<0)
{
printf("Error in listen\n");
exit(0);
}
if((acpt=accept(sd,(struct sockaddr*)&NULL,NULL))<0)
{
printf("\n\t Error in accept");
exit(0);
}
while(1)
{
bytes=recv(acpt,receiv,50,0);
receiv[bytes]='\0';
if(strcmp(receiv,"end")==0)
{
close(acpt);
close(sd);
exit(0);
}
else
{
printf("Command received : %s",receiv);
system(receiv);
printf("\n");
}
}
}
```

Client program

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <errno.h>

int main()
{
    int sd,acpt,len,bytes,port;
    char send1[50],receiv[50];
    struct sockaddr_in serv,cli;
    if((sd=socket(AF_INET,SOCK_STREAM,0))<0)
    {
        printf("Error in socket\n");
        exit(0);
    }
    bzero(&serv,sizeof(serv));
    printf("Enter the port number : ");
    scanf("%d",&port);
    serv.sin_family=AF_INET;
    serv.sin_port=htons(port);
    serv.sin_addr.s_addr=htonl(INADDR_ANY);
    if(connect(sd,(struct sockaddr *)&serv,sizeof(serv))<0)
    {
        printf("Error in connection\n");
        exit(0);
    }
    while(1)
    {
        printf("Enter the command:");
        gets(send1);
```

```

        if(strcmp(send1,"end")!=0)
        {
            send(sd,send1,50,0);
        }
        else
        {
            send(sd,send1,50,0);
            close(sd);
            break;
        }
    }
}

```

OUTPUT**SERVER**

[11ca013@mc linux network]\$ cc 8rcpser.c

[11ca013@mc linux network]\$./a.out

Enter the port number : 8596

Command received : ls

1.c	4tcpcli.c	7pingcli.c	exno2a.c	hello.txt	sevan.txt
2.c	4tcpser.c	7pingser.c	exno2client.c	mca1.txt	tcpchclient.c
3tcpcli.c	4udpcli.c	8rcpcli.c	exno2server.c	menu.sh	tcpchserver.c
3tcpser.c	4udpser.c	8rcpser.c	exnola.sh	payroll.sh	third.txt
3udpcli.c	6tcpftpcli.c	9rcpcli.c	febser.sh	rajiv	

Command received : cat sample.txt

rajivgandhi

hi

how are you

Command received : date

Fri Mar 1 11:39:14 IST 2013

Command received : cal

March 2013

Su	Mo	Tu	We	Th	Fr	Sa
					1	2
3	4	5	6	7	8	9
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24	25	26	27	28	29	30
31						

Command received : end

CLIENT

[11ca013@mc linux network]\$ cc 8rcpcli.c

[11ca013@mc linux network]\$./a.out

Enter the port number: 1806

Enter the command: Enter the command: ls

Enter the command: cat sample.txt

Enter the command: date

Enter the command: cal

Enter the command: end

Q12. Implementation of DNS.

Ans :

// **Client Program : dnsc.c**

```
#include <stdio.h>
```

```
#include <sys/stat.h>
```

```
#include <sys/types.h>
```

```
#include <sys/socket.h>
```

```
#include <arpa/inet.h>
```

```
#include <netinet/in.h>
```

```
main()
```

```
{
```

```
    struct sockaddr_in server, client;
```

```
    int s, n;
```

```
    char b1[100], b2[100];
```

```
    s = socket(AF_INET, SOCK_DGRAM, 0);
```

```
    server.sin_family = AF_INET;
```

```
    server.sin_port = 3000;
```

```
    server.sin_addr.s_addr = inet_addr("127.0.0.1");
```

```
    n = sizeof(server);
```

```
    printf("\nEnter canonical address: ");
```

```
    scanf("%s", b2);
```

```
    sendto(s, b2, sizeof(b2), 0, (struct sockaddr*)&server, n);
```

```
    recvfrom(s, b1, sizeof(b1), 0, NULL, NULL);
```

```
    printf("%s\n", b1);
```

```
}
```

// **Server Program : dnss.c**

```
#include<stdio.h>
#include<sys/stat.h>
#include<sys/types.h>
#include<sys/socket.h>
#include<netinet/in.h>
#include<arpa/inet.h>
#include<string.h>
main()
{
    FILE *fp;
    struct sockaddr_in server,client;
    int s,n;
    char b1[100],b2[100],a[100];
    s=socket(AF_INET,SOCK_DGRAM,0);
    server.sin_family=AF_INET;
    server.sin_port=3000;
    server.sin_addr.s_addr=inet_addr("127.0.0.1");
    bind(s,(struct sockaddr *)&server,sizeof(server));
    n=sizeof(client);
    while(1)
    {
        strcpy(b2,"");
        fp=fopen("dns.txt","r");
        recvfrom(s,b1,sizeof b1, 0,(struct sockaddr *)&client,&n);
        while(!feof(fp))
        {
            fscanf(fp,"%s",a);
            if(strcmp(a,b1)==0)
            {
                fscanf(fp,"%s",b2);
                break;
            }
        }
        if(strcmp(b2,"")==0)
```

```
{  
    strcpy(b2,"Not found...");  
}  
fclose(fp);  
sendto(s,b2,sizeof b2,0,(struct sockaddr*)&client,n);  
}  
}
```

/* (DNS.txt)

```
www.2k8cs.tk 93.170.52.20  
www.google.com 192.168.0.7  
www.yahoo.com 192.168.0.16  
www.2k8618.blogspot.com 66.102.13.191  
*/
```

Output:

Terminal1: (Client)

```
nn@linuxmint ~ $ gcc dnsc.c -o client  
nn@linuxmint ~ $ ./client  
Enter canonical address: www.2k8cs.tk  
93.170.52.20  
nn@linuxmint ~ $ ./client  
Enter canonical address: www.2k8618.blogspot.com  
66.102.13.191  
nn@linuxmint ~ $
```

Terminal 2: (Server)

```
nn@linuxmint ~ $ gcc dnss.c -o server  
nn@linuxmint ~ $ ./server
```

FACULTY OF INFORMATICS
M.C.A II Year III - Semester Examination
MODEL PAPER - I
COMPUTER NETWORKS

Time : 3 Hours]

[Max. Marks : 70

Note: Answer all the question according to the internal choice (5 × 14 = 70)

ANSWERS

1. (a) State the characteristics and advantages of networks. (Unit-I, Q.No.2)
(b) Explain briefly about MAN. (Unit-I, Q.No.9)

OR

2. (a) Define Topology and explain topologies of computer network. (Unit-I, Q.No.19)
(b) Explain Asynchronous & Synchronous Modems. (Unit-I, Q.No.28)
3. (a) Explain design issues with data link layer. (Unit-II, Q.No.2)
(b) What is Error Correction and Detection? (Unit-II, Q.No.3)

OR

4. (a) Explain Dynamic channel allocation in LAN and MAN. (Unit-II, Q.No.11)
(b) Explain the concept of Transparent Bridges. (Unit-II, Q.No.18)
5. (a) What is Internet working and explain its types? (Unit-III, Q.No.2)
(b) What is Link State Routing Protocol? (Unit-III, Q.No.9)

OR

6. (a) Explain IPv4 Addressing methods. (Unit-III, Q.No.11)
(b) How Network Address Translation Works? (Unit-III, Q.No.15)
7. What is multiplexing? Explain its methods. (Unit-IV, Q.No.2)

OR

8. Explain UDP and its Attributes. (Unit-IV, Q.No.7)
9. Describe various services of transport layer. (Unit-V, Q.No.2)

OR

10. Explain in concept of Domain Name System. (Unit-V, Q.No.6)

FACULTY OF INFORMATICS
M.C.A II Year III - Semester Examination
MODEL PAPER - II
COMPUTER NETWORKS

Time : 3 Hours]

[Max. Marks : 70

Note: Answer all the question according to the internal choice (5 × 14 = 70)

ANSWERS

1. (a) What are the different uses of Networks? (Unit-I, Q.No.4)
(b) Explain layering architecture of TCP/IP model. (Unit-I, Q.No.15)

OR

2. (a) Explain briefly about LAN. (Unit-I, Q.No.8)
(b) What are Optical Fibers? (Unit-I, Q.No.22)
3. (a) Explain in detail about the working of flow control. (Unit-II, Q.No.6)
(b) What are the differences between ARP and RARP? (Unit-II, Q.No.24)

OR

4. (a) Define ARP. Explain different types of ARP. (Unit-II, Q.No.22)
(b) Explain the concept of Hamming code. (Unit-II, Q.No.5)
5. (a) Define networking layer in TCP/IP Model. (Unit-III, Q.No.1)
(b) What is Distance Vector Protocol Routing ? (Unit-III, Q.No.8)

OR

6. (a) Explain the concept of subnetting. State its advantages and disadvantages. (Unit-III, Q.No.12)
(b) What are the differences between IPv4 and IPv6 ? (Unit-III, Q.No.18)
7. Define transport layer. Explain the services of transport layer. (Unit-IV, Q.No.1)

OR

8. Discuss about TCP timer. (Unit-IV, Q.No.5)
9. Explain iterative communication by using TCP. (Unit-V, Q.No.4)

OR

10. What is FTP - File Transfer Protocol? (Unit-V, Q.No.12)

FACULTY OF INFORMATICS
M.C.A II Year III - Semester Examination
MODEL PAPER - III
COMPUTER NETWORKS

Time : 3 Hours]

[Max. Marks : 70

Note: Answer all the question according to the internal choice (5 × 14 = 70)

ANSWERS

1. (a) Explain the basic components of computer networking. (Unit-I, Q.No.5)
(b) Describe the characteristics of line coding. (Unit-I, Q.No.25)

OR

2. (a) Explain briefly about Wide Area Network. (Unit-I, Q.No.10)
(b) Define Network software. (Unit-I, Q.No.17)
3. (a) Explain the concept of Bridges. (Unit-II, Q.No.17)
(b) Explain the concept of Slotted Aloha. (Unit-II, Q.No.14)

OR

4. (a) What is ALOHA? Explain different types of Aloha? (Unit-II, Q.No.12)
(b) Define HDLC Protocol. (Unit-II, Q.No.7)
5. (a) Define Packet Switching and explain its advantages and disadvantages. (Unit-III, Q.No.3)
(b) What are the differences between distance Vector Routing and link State Routing? (Unit-III, Q.No.10)

OR

6. (a) What is Router? Explain the Characteristics of Routers/Router Protocols. (Unit-III, Q.No.7)
(b) What is CIDR? Discuss with an Example. (Unit-III, Q.No.13)
7. Define Transmission Control Protocol. State the various services of TCP. (Unit-IV, Q.No.3)

OR

8. Define QoS. Explain the specific purpose of QoS. (Unit-IV, Q.No.6)
9. What is HTTP? Explain the features of HTTP. (Unit-V, Q.No.13)

OR

10. Explain Application Layer and its Functions. (Unit-V, Q.No.5)

FACULTY OF INFORMATICS
M.C.A. (2 years Course) III - Semester (CBCS) (Main) Examination
April/May - 2023
COMPUTER NETWORKS

Time : 3 Hours

Max. Marks : 70

- Note :** I. Answer one question from each unit. All questions carry equal marks.
II. Missing data, if any, may be suitably assumed.

Answers

Unit-I

1. (a) Discuss about components of computer networks. (Unit-I, Q.No. 5)
(b) Briefly explain the different types of transmission media. (Unit-I, Q.No. 20)

(OR)

2. (a) Describe the ISO/OSI reference model. (Unit-I, Q.No. 13)
(b) Explain the different types of modems. (Unit-I, Q.No. 27)

Unit-II

3. (a) Discuss the process of CRC. (Unit-II, Q.No. 4)
(b) Briefly explain the concept of bridges. (Unit-II, Q.No. 16)

(OR)

4. (a) Discuss the functions of MAC. (Unit-II, Q.No. 9)
(b) Explain the concept of RARP in detail. (Unit-II, Q.No. 23)

Unit - III

5. (a) Discuss about the link state routing protocol. (Unit-III, Q.No. 9)
(b) Explain the differences between IPv4 and IPv6. (Unit-III, Q.No. 18)

(OR)

6. (a) Describe the CIDR with an example. (Unit-III, Q.No. 13)
(b) Briefly explain various types of IGMP. (Unit-III, Q.No. 20)

Unit-IV

7. (a) Discuss the services of transport layer. (Unit-IV, Q.No. 1)
(b) Explain the User datagram protocol. (Unit-IV, Q.No. 7)

(OR)

8. (a) Describe the Frequency Division multiplexing in detail. (Unit-IV, Q.No. 2)
(b) Briefly explain the TCP congestion control. (Unit-IV, Q.No. 4)

Unit-V

9. (a) Discuss about the concept of socket programming in detail. **(Unit-V, Q.No. 1)**
(b) Explain the concept of DNS. **(Unit-V, Q.No. 6)**

(OR)

10. (a) Describe an iterative communication by using TCP. **(Unit-V, Q.No. 4)**
(b) Briefly explain the FTP. **(Unit-V, Q.No. 12)**

Rahul Publications

FACULTY OF INFORMATICS
MCA III - Semester Examinations
October / November - 2023
COMPUTER NETWORKS

Time : 3 Hours]

[Max. Marks : 70

Note:

I. Answer one question from each unit. All questions carry equal marks.

II. Missing data, if any, may be suitably assumed.

UNIT - I

1. (a) Explain the different types of network topologies. (Unit-I, Q.No.19)
(b) Describe the concept of RS-232 interfacing. (Unit-I, Q.No.29)

OR

2. (a) Briefly explain the TCP/IP reference model. (Unit-I, Q.No.15)
(b) Discuss the different line coding techniques. (Unit-I, Q.No.26)

UNIT - II

3. (a) Briefly discuss the concept of Hamming code. (Unit-II, Q.No.5)
(b) Explain the different types of ALOHA. (Unit-II, Q.No.12)

OR

4. (a) Describe the Error control mechanism. (Unit-II, Q.No.6)
(b) Explain different types of ARP in detail. (Unit-II, Q.No.22)

UNIT - III

5. (a) Discuss the Distance Vector Routing protocol. (Unit-III, Q.No.8)
(b) Briefly explain the concept of ICMP. (Unit-III, Q.No.21)

OR

6. (a) Describe the IPv4 addressing methods in detail. (Unit-III, Q.No.11)
(b) Explain the concept of BGP. (Unit-III, Q.No.23)

UNIT - IV

7. (a) Describe the Time Division Multiplexing in detail. (Unit-IV, Q.No.2)
(b) Briefly explain the TCP Timer management. (Unit-IV, Q.No.5)

OR

8. (a) Discuss the various services of TCP. (Unit-IV, Q.No.3)
(b) Explain the Quality of Services. (Unit-IV, Q.No.6)

UNIT - V

9. (a) Describe an iterative communication by using UDP. (Unit-V, Q.No.3)
(b) Explain the SMTP works in detail. (Unit-V, Q.No.11)

OR

10. (a) Discuss the functions of Application Layer. (Unit-V, Q.No.5)
(b) Explain the HTTP in detail. (Unit-V, Q.No.13)