MCA(S5)19

KRISHNA KANTA HANDIQUI STATE OPEN UNIVERSITY

Housefed Complex, Dispur, Guwahati - 781 006

Master of Computer Applications

DATA COMMUNICATION AND COMPUTER NETWORKS

CONTENTS

Subject Expert

Prof. Anjana Kakati Mahanta, Deptt. of Computer Science, Gauhati University

Prof. Jatindra Kr. Deka, Deptt. of Computer Science and Engineering,
Indian Institute of Technology, Guwahati

Prof. Diganta Goswami, Deptt. of Computer Science and Engineering,
Indian Institute of Technology, Guwahati

Course Coordinator

Tapashi Kashyap Das, Assistant Professor, Computer Science, KKHSOU

Arabinda Saikia, Assistant Professor, Computer Science, KKHSOU

SLM Preparation Team

Units	Contributor
1	**Tapashi Kashyap Das,** KKHSOU
2	**Chakradhar Das,** Lecturer (Selection Grade), Deptt. of Electrical Engineering, Bongaigaon Polytechnic, Bongaigaon, Assam
3 & 5	**Swapnanil Gogoi** Asst. Professor, Institute of Distance and Open Learning (IDOL), Gauhati University
6	**Bornali Gogoi**, Asst. Professor, Deptt. of Computer Applications (MCA), Assam Engineering College, Jalukbari, Guwahati, Assam
4 & 7	**Pritam Medhi,** Research Scholar, Gauhati University

July 2013

Printed and published by Registrar on behalf of the Krishna Kanta Handiqui State Open University.

The university acknowledges with thanks the financial support provided by the **Distance Education Council, New Delhi**, for the preparation of this study material.

Housefed Complex, Dispur, Guwahati- 781006; Web: www.kkhsou.net

COURSE INTRODUCTION

This is a course on ***Data Communication and Computer Networks***. This course is intended to introduce the learner about the principles, design, implementation, and performance of data communication and computer networks. Networking deals with the technology and architecture of the communication networks used to interconnect communicating devices.

This course contains seven essential units. The first unit is an introductory unit on computer networks. This unit includes different categories of computer network like LAN, MAN, WAN. Concept of network topology along with their types is also covered in this unit. The second unit is on network models. Two most important ISO-OSI and TCP/IP models are described in this unit. The third unit describes Physical layer which is the lowest layer in the OSI model. It describes different networking hardware transmission media as well as transmission methods of computer networks. The fourth unit is on Data link layer. The fifth unit focuses on Network layer. The sixth unit is on Transport layer. The seventh unit is the last unit and it discusses the Application layer of ISO-OSI Model.

While going through a unit, you will notice some boxes along-side, which have been included to help you know some of the difficult, unseen terms. Some “ACTIVITY’ (s) have been included to help you apply your own thoughts. Again, we have included some relevant concepts in “LET US KNOW” along with the text. And, at the end of each section, you will get “CHECK YOUR PROGRESS” questions. These have been designed to self-check your progress of study. It will be better if you solve the given problems in these boxes immediately, after you finish reading the section in which these questions occur and then match your answers with “ANSWERS TO CHECK YOUR PROGRESS” given at the end of each unit.

MASTER OF COMPUTER APPLICATIONS

Data Communication and Computer Networks

DETAILED SYLLABUS

Unit 1: Basics of Computer Networks (Marks: 15)
Computer Network: Definition, Goals, Structure; Broadcast and Point-To-Point Networks; Network Topology and their various Types; Types of Network: LAN, MAN, WAN; Server Based LANs & Peer-to-Peer LANs; Communications Types: Synchronous, Asynchronous; Modes of Communication: Simplex, Half Duplex, Full Duplex; Protocols and Standards.

Unit 2: Network Models (Marks:15)
Design Issues of the Layer, Protocol Hierarchy, ISO-OSI Reference Model : Functions of each Layer, Various Terminology used in Computer Network, Connection-Oriented & Connectionless Services, Internet (TCP/IP) Reference Model, Comparison of ISO-OSI and TCP/IP Model

Unit 3: Physical Layer (Marks:15)
Signals: Analog and digital signals, Data rate limits, Transmission impairment, Signal measurements like throughput, propagation speed and time, wave length; Digital Transmission: Line coding, block coding, sampling, transmission mode; Analog Transmission: Modulation digital data, telephone modem, Modulation analog signals; Multiplexing: FDM, WDM, TDM; Transmission Media: Guided media, Unguided media, Circuit Switching and Telephone Network: Circuit switching, telephone network;

Unit 4: Data Link Layer (Marks:15)
Error Detection and Correction: Type of errors, detection and correction of errors; Data Link Control and Protocol: Flow & error control, Stop-And -Wait ARQ, Go-Back-N ARQ, Select Repeat ARQ, HDLC; Point-To-Point Access: Point-to-point protocol, PPP stack; Local Area Network: Traditional Ethernet, fast and gigabit Ethernets; Connecting LANs, Backbone Networks and Virtual LANs: Connecting devices, Backbone networks, Virtual LANs;

Unit 5: Network Layer (Marks: 15)
Internetworks, Addressing, Routing, Network Layer Protocols: ARP, IP, ICMP, IPV6, Unicast routing, Unicast routing protocols, Multi routing, Multicast routing protocols;

Unit 6: Transport Layer (Marks:15)
Process-To-Process delivery, user data gram, Transmission control protocol

Unit 7: Application Layer (Marks: 10)
Client-Server Model: Client-Server model, Socket interface; A brief introduction to DNS, SMTP, FTP

UNIT-1 : BASICS OF COMPUTER NETWORK

UNIT STRUCTURE

1.1 LEARNING OBJECTIVES

After going through this unit, you will be able to :

- define computer network with their goals
- learn about different switching techniques
- describe connection-less and connection-oriented services
- learn about broadcast and point-to-point networks
- describe the types of computer networks
- describe & differentiate server and peer-to-peer LANs
- learn about network topology
- describe synchronous & asynchronous types of communications
- learn about different modes of communication
- learn about protocols and standards.

1.2 INTRODUCTION

Beginning our discussion with the definition of computer network and its goals, we introduce various concepts associated with data communication. We further discuss briefly various categories of network together with the concept of network topology. Towards the end, we briefly gothrough different modes of communication followed by protocols and standards.

1.3 COMPUTER NETWORK

A computer network is a group of computers that shares information across wireless or wired technology. A computer network is a collection of several computers and terminals interconnected by one or more transmission paths. The main goal of a transmission system is the transfer and exchange of data between the computers and terminals.

The terms DTE and DCE are very common in the datacommunication. DTE stands for ***data terminal equipment*** and DCE stands for ***data communication equipment***. DTE is used to describe the end user machine, which is usually a computer or terminal. The function of communication

network is to interconnect DTEs so that they can share resources, exchange data, provide backup for each other, and allow users to perform their work from any location. The DCE is used to connect the DTEs into the communications line or channel. It also contains a portion of an application process but the primary function of the DCE remains to provide an interface to the DTE with the communication network.

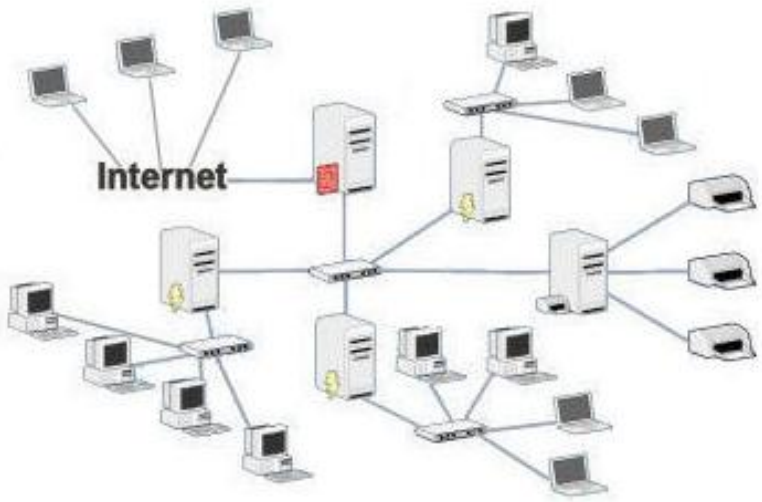


Fig. 1.1 : A typical computer network

Before discussing the technical issues in detail, it is worth devoting some time to pointing out why people are interested in computer networks and what they can be used for.

1.4 GOALS OF COMPUTER NETWORKS

Before the advent of computer networks, papers and diskettes were the only means to share information. Now a days, computer networks which comprises a number of computers as well as other devices like printers, scanners etc. are widely used to share resources in a more efficient and faster way.

In a network, whether big or small, all the devices are interconnected to transmit and receive data from each other. These connections are usually made not only by copper wires; instead, fiber optics, microwaves, infrared, and communication satellites are also used for efficient communication. The main goals of these networks can be summarized as follows :

- **Resource Sharing**

  Using computer networks, it is possible to share programs, data and other resources among several users on the network. The sharing of resources is independent of the physical location of the user and the resource. Files on a particular user's computer can be shared on the network or files can be placed on a file server, which provides a central location for all files needed by the users on the network. Users can also share devices such as printers, CD-ROM drives, and hard drives etc. It also makes upgrading an application easier because the upgrade only has to be performed on the server itself. Thus, we don't need seperate resource for each computer.

- **High Reliablity**

  A second goal is to provide high reliability by having alternative sources of supply. For example, all files could be replicatied on two or three machines, so if one of them is unavailable, the other copies could be available.

- **Cost Reduction**

  Resource sharing automatically reduces cost and hence money can be saved. For example, suppose, there are ten users and each requires a printer. If they could have been working indivually, then ten printers would have to be purchased. If these ten users are allowed to work in a network, then only two or three printers would be sufficient.

- **Communication Medium**

  Computer networks provide a powerful communication medium among peoples who are in a geographically same or different location. A file that was updated or modified on a network, can be seen by the other users on the network immediately. Thus, it becomes easy for two or more peoples living far apart to work on a same project by dividing it using a network. They can write programs, can discuss or can even change or modify some data using a network while they are far off. Otherwise, they will have to wait for several days for letter or some other media. Thus, it makes speedy co-operations and enhances human to human communication.

1.5 SWITCHING TECHNIQUES

When we have multiple work stations, a problem of connecting them for one to one communication arises. One solution for this problem is to establish point to point connections between each individual pair of stations. To interconnect ***n*** stations, ***n(n-1)/2*** individual connections are needed. For instance, to connect 10 components, total of individual 45 connections are required; for 100 components, 4950 connections are required. The requirements increases drastically as the number of stations to be interconnected increase, and for a large number of stations, it is almost impossible to maintain individual connections. A well known solution for such type of problems is, ***switching***. Let us think, how things would be if we could only use our telephone to talk to just one other person! So there are requirements for switching systems to route our calls around the world. The switches can be placed on the transmission path so that the stations do not need to be interconnected directly. In this section, we will briefly discuss various switching techniques which are in use currently.

In general, there are three methods of switching have been used: ***circuit switching***, ***message switching*** and ***packet switching***.

1.5.1 Circuit Switching

Circuit switching is the transmission technology that has been used since the first communication networks in the nineteenth century. The basic idea of circuit switching is to set up a dedicated logical path between two users or machines, prior to the commencement of data communication. In circuit switching, a caller must first establish a connection to a callee before any communication. During the connection establishment, resources are allocated between the caller and the callee. The most common application of circuit switching is *telephone* network. Circuit switching mechanism is divided into three phases: circuit establishment, data transfer and circuit disconnect

Characteristics of Circuit Switching

- An end to end dedicated path is needed.

- Connection path must be established before the beginning of data transmission.
- Channel capacity must be reserved between each pair of nodes whether that capacity is utilized or not.
- The technology is developed to handle voice data because of its key requirement that there should be no delay in the transmission and a constant signal transmission rate must be maintained.

Advantages of Circuit Switching

- It is well suited for real time communication purposes.
- The service provided by circuit switching are guaranteed since the channel remains dedicated to users throughout the communication session.

Disadvantages of Circuit Switching

- Any unused ***bandwidth*** over the allocated circuit is wasted.
- Line may be idle most of the time.
- Both sender and the receiver must function with the same speed. This limits the utility of the network in interconnecting a variety of host computers and terminals.

Bandwidth : A communication channel utilizes a specific frequency to transmit the electromagnetic energy which represents the data. Transmitting information requires more than a single frequency. And, for this purpose a band of spectrum around the nomical frequency is required, which is known as bandwidth of the signal.

1.5.2 Message Switching

Message switching refers to a switching technique involving transmission of messages from node to node through a network. In this technique, the source computer sends data or the message to the switching office first, which stores the data in its buffer. It then looks for a free link to another switching office and then sends the data to this office. This process is continued until the data are delivered to the destination computers. Owing to its working principle, it is also known as *store and forward*. That is, store first (in switching office), forward later, one jump at a time.

Advantages of Message Switching

- No waiting of the setup of path. As soon as a user has data to send, he/she may transmit it over the channel.
- The channel can be fully utilized.

- Priority of messages can be implemented.
- Broadcasting can be done very easily to all the nodes in a network.

Disadvantages of Message Switching

- Not suitable for real time transmission
- In message switching, message size is not fixed. For very large message, the nodes are required to have large buffers, which may not be practically implemented due to economical and technical constraints.
- Since there is no size limitation of the message, long messages can keep the channel blocked for a long period of time.

1.5.3 Packet Switching

Packet switching is very similar to message switching. The main difference is that the length of the units of data that may be presented to the network is limited in a packet-switches network.

With message switching, there is no limit on block size, in contrast, packet switching places a tight upper limit on block size. A fixed size of packet which can be transmitted across the network is specified. Another point of its difference from message switching is that data packets are stored on the disk in message switching whereas in packet switching, all the packets of fixed size are stored in main memory. This improves the performance as the access time (time taken to access a data packet) is reduced, thus, the throughput (measure of performance) of the network is improved.

There are two approaches to implement packet switching. These are : the datagram approach and the virtual circuit approach.

1.6 CONNECTION-ORIENTED & CONNECTION-LESS SERVICE

On the basis of *acknowledgement send by the receiver*, there are two distinct techniques used in data communications to transfer data. These are:

connection-oriented and *connection-less services*. Each service can be characterized by a quality of service. In general, a reliable service is implemented by having the receiver acknowledge the receipt of each message so the sender is sure that it arrived.

- ***Connection-oriented service*** is modeled after the telephone system. For making a call to someone, first we have to pick up the phone and dial the number. After the connection is established, we talk, and then hang up.

  Connection-oriented service requires a connection be established before any data can be sent. It set up virtual links between end systems through a network. After the connection is established, the data is transferred. As soon as the transmission is completed, the service user releases the connection. This procedure requires a specific acknowledgement for the information whether the connection is established or not. This method is often called a “reliable” network service. In some cases, when the connection is established, the sender, receiver, and subnet conduct a negotiation about parameters to be used such as maximum message size, quality of service required, and other issues. Typically, one side makes a proposal and the other side can accept it, reject it, or make a proposal as an alternative to the earlier proposal.

- The analogy of sending letters and postcards best explains the ***connection-less service.*** Each message (letter) carries a full destination address, and each one is routed through the system independent of all the others. Normally, when two messages are sent to the same destination, the first one sent will be first one to arrive. But it also possible that the first one sent can be delayed so that the second one arrives first. In a connection-less service, there is no initial end-to-end setup for a session; each packet is independently routed to its destination. The sender simply starts sending packets (called datagrams) to the destination. This service does not have the reliability of the connection-oriented method, but it is useful for periodic burst transfers. Neither system must maintain state information for the systems that they send transmission to or receive transmission from.

1.7 BROADCAST & POINT-TO-POINT NETWORKS

On the basis of *transmission technology*, computer networks can be divided into two categories: *broadcast* and *point-to-point*.

- ***Broadcast networks*** have a single communication channel which is shared by all the machines on the network. Here, messages are broken into *packets* which are then broadcast to all machines in the channel. A packet sent by any machine is received by all the machines but only that machine processes the packet for which it is intended. An address field (destination address), within the packet specifies for whom it is intended. Upon receiving a packet, a machine checks the address field. If the packet is intended for itself, it processes the packet; if it is intended for some other machine, it is just ignored. However, in broadcast networks, we have the problem of deciding who uses the channel, if there is competition for it. For this, *medium access control* (MAC) is used to determine who goes next.

  As an analogy, let us imagine a scenario where a teacher taking the attendance in a particular class. He/she will call every roll number in the classroom, all will listen but only the respective student will respond. Broadcast implies "*sending a signal where multiple parties may hear a single sender*".

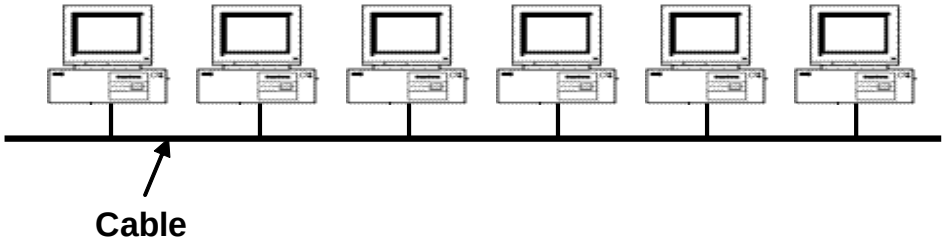


Fig.1.2 : A Broadcast Network

- Again, let us imagine a scenario that a teacher is taking the attendance in an examination hall. He/she will go to each member of the room and will take the attendance individually. Here, only one person listen and he only responds to the call. ***Point-to-point network*** is a method of communication where one "point" (person or device or entity) speaks to another entity. Point-to-point network consists of many connections between individual pairs of machines. Packet, on this

type of network, may have to visit one or more intermediate machines to reach from the source to the destination. Often multiple routes of different lengths are possible in point-to-point networks. Therefore, routing algorithms are required to determine the best routes out of multiple available routes.

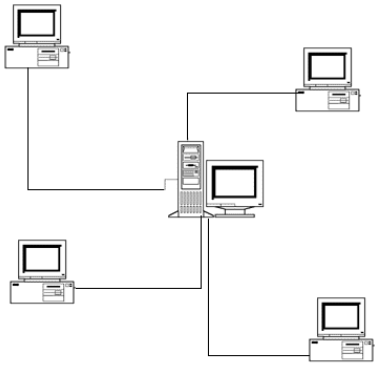


Fig. 1.3 : Point-to-Point Network

1.8 CATEGORIES OF NETWORK

Whenever we have a set of computers or networking devices to be connected, we make the connections, depending on the physical layout and our requirements. On the basis of geographical area covered, computer networks can be classified under the following three categories:

- Local Area Network
- Metropolitan Area Network
- Wide Area Network

1.8.1 Local Area Network

A ***local area network*** or **LAN**, is a high-speed data network that covers a relatively small geographic area such as a building, a laboratory, or a school. It typically connects workstations, personal computers, printers, servers, and other devices. LANs offer computer users many advantages, including shared access to devices and applications, file exchange between connected users, and communication between users via electronic mail and other applications. LANs differ in the way the computers are connected (i.e., their topology), in how information moves around the network (i.e., their transmission technology) and their size.

IEEE (Institute of Electrical and Electronic Engineers) is a US publishing and standards organization responsible for many LAN stan-

dards such as the 802 series. *IEEE 802.3,* popular called *Ethernet* is the most popular LAN, usually operating at 10 Mbps to 10 Gbps and is present in most large organizations and offices. In a typical LAN configuration, one computer is designated as the *server*. It stores all of the software that controls the network, as well as the software that can be shared by the computers attached to the network. Computers connected to the server are called *workstations*. The various components and standards associated with a LAN will be discussed in *Unit 5* of this course. Two most commonly used LAN implementations are depicted below:

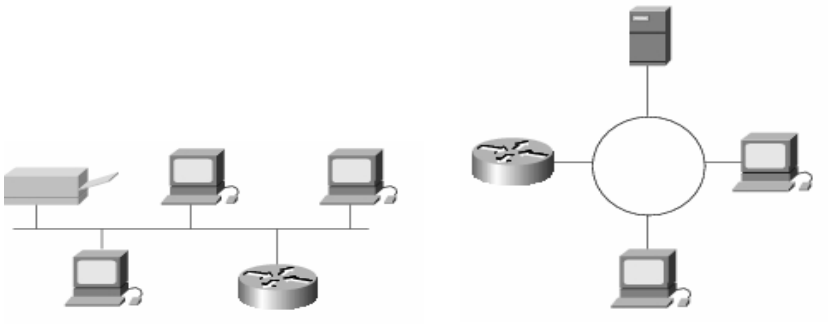


Fig.1.4 : (a) Ethernet or IEEE 802.3 (b) Token Ring or IEEE 802.5

LAN protocols function at the lowest two layers, i.e., the *physical* and the *data link layer* of the OSI reference model. We will discuss the layer concept of ISO-OSI reference model in the next unit which is “Network models”.

1.8.1.1 LAN Transmission Methods

LAN data transmissions fall into three classifications: *unicast*, *multicast*, and *broadcast*. In each type of transmission, a single packet is sent to one or more nodes.

- ***Unicast :*** Unicast is a one-to-one transmission method in which the network carries a message to one receiver, such as from a server to a LAN workstation. In a unicast environment, even though multiple users might ask for the same information from the same server at the same time, such as a video clip; duplicate data streams are sent. Unicast sends separate data streams to each computer requesting the data, in turn flooding the network with traffic.

- ***Multicast :*** Multicast is a one-to-many transmission method in which the network carries a message to multiple receivers at the same time. Multicast is similar to broadcasting, except that multicasting means sending to a specific group, whereas broadcasting implies sending to everybody, whether they want the traffic or not. When sending large amounts of data, multicast saves considerable network bandwidth because the bulk of the data is sent only once. The data travels from its source through major backbones and is then multiplied, or distributed out, at switching points closer to the end users. This is more efficient than a unicast system, in which the data is copied and forwarded to each recipient.
- ***Broadcast :*** Concept of broadcast is already discussed in previous section. *Broadcast* is a one-to-all transmission method in which the network carries a packet to all devices at the same time, but a particular machice for which the packet is intended accepts it.

1.8.1.2 Peer-to-Peer LAN & Server Based LAN

On a LAN, we expect to share files, programs, or printers, all without being particularly aware of where the physical resources we're using are actually located. LANs providing these types of services are typically set up either as "*peer-to-peer*" or "*client-server*" LANs, or perhaps as a combination of the two.

- **Peer-to-Peer LAN**

  All the machines on a peer-to-peer LAN are equal. Provided that the file's owners give permission, a file on machine A can be accessed from machine B, and vice versa. Peer-to-peer LANs do not require any one machine to be a dedicated, high-performance server; service by a peer-to-peer LAN is often cheaper for this reason. Peer-to-peer LANs work well when only a small number of machines are connected to it. But as the size of the LAN grows, peer-to-peer services can become quite disorganized. To serve all its peers, each machine on the LAN

must be powerful enough and for this reason cost increases. For larger LANs, the dedicated client-server LAN architecture becomes more cost effective.

Advantages

Less initial expense – No need for a dedicated server.

Setup – An existing operating system (such as Windows XP) of the machine may only need to be reconfigured for peer-to-peer operations.

Disadvantages

Decentralized – No central repository for files and applications.

Security – Does not provide the security available on a client/server.

- **Client-Server LAN**

A client-server LAN consists of one or more server machines on which shared files and programs reside and many client machines where people do their task. The LAN server machines are usually have higher configuration and fast because they must serve many users, while the client machines need only be fast enough for one person to use at a time. Shared printers are either attached directly to a server, or to a print server (a specialized computer attached to the network), or to a personal computer on network that acts as a print server.

Advantages

Centralized – In case of client-server architecture, resources and data security are controlled through the server.

Scalability – Any or all elements can be replaced individually as needs increase.

Flexibility – New technology can be easily integrated into system.

Accessibility – Server can be accessed remotely and across multiple platforms.

Disadvantages

Expense – Requires initial investment in dedicated server.

Maintenance – Large networks will require a staff to ensure efficient operation.

Dependence – When server goes down, operations will cease across the network.

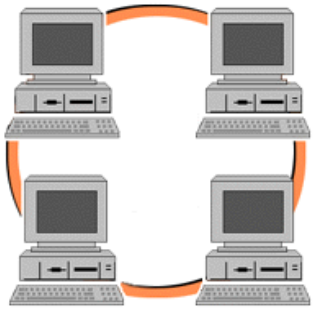


Fig.1.5 : Peer-to-Peer

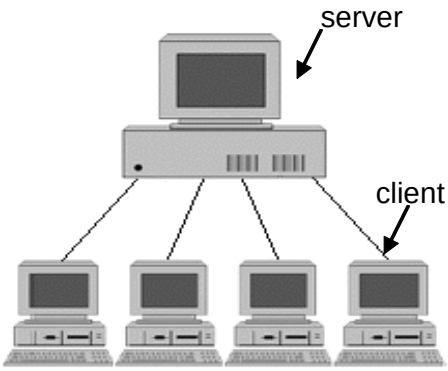


Fig. 1.6 : Client-Server

1.8.2 Metropolitan Area Network

A ***metropolitan area network,*** or **MAN**, is basically a bigger version of a LAN and normally uses similar technology. It might cover a group of nearby corporate offices or a city and might be either private or public.

A MAN often provides efficient connections as it has high-speed transmission capabilities which uses some type of telecomminication components to handle long-distance transmission. One very common example of MAN is the cable television network. Another important example is the high-speed wireless Internet access, which has been standardized as *IEEE 802.16.*

1.8.3 Wide Area Network

A ***wide area network*** or **WAN** is a computer network covering multiple distance areas, which may spread across the entire world. WANs often connect multiple smaller networks, such as local area networks (LANs) or metropolitan area networks (MANs).

Typically, a WAN consists of a number of interconnected switching nodes. A transmission from any one device is routed through these internal nodes to the specified destination device. These nodes are not concerned with the content of the data; rather, their purpose is to provide a switching facility that will move the data from node to node until they reach their destination. Traditionally, WANs have been implemented using either *circuit switching* or *packet switching* technologies. More recently, frame relay and *ATM* (*Asynchronous Transfer Mode*) networks have assumed major roles. Frame relay provides higher data rates, lowar costs, efficient handling of bursty data transmission in less expenditure. ATM offer more bandwidth to end users at less cost.

The ***Internet*** is the best known example of a WAN. Some segments of the Internet are also WANs in themselves. The Internet is a system of linked networks that are worldwide in scope and facilitate data communication services such as *remote login*, *file transfer*, *e-mail*, the *World Wide Web* etc. With the rise in demand for connectivity, the Internet has become a communications highway for millions of users. The Internet was initially restricted to military and academic institutions, but now it is a full-fledged conduit for any and all forms of information and commerce. Internet websites now provide personal, educational, political and economic resources to every corner of the planet. The last unit (Unit 6) of this course will give us the concept of Internet and various servicices provided by Internet.

CHECK YOUR PROGRESS-1

1. State True or False :
   i) Broadcast networks share a single communication channel.
   ii) For larger LANs, the client-server LAN architecture is less cost effective as compared to peer-to-peer LANs.
   iii) The world's most popular WAN is the Internet.
   iv) Cable televishion network is an example of MAN.

1.9 NETWORK TOPOLOGY

In this section we will discuss how computers and others devices are connected in a network. While discussing, we may come accross the name of various internetworking devices like hub, switch, router etc.; the role and functioning of each devices will be covered in Unit 4 of this course.

In computer networking, ***topology*** refers to the layout of connected devices. Network topologies can be *physical* or *logical*. ***Physical topology*** means the physical design of a network including the devices, location and cable installation. It defines how the systems are physically connected. Several physical topologies are in use for networks today. Some of the common include the *bus*, *ring*, *star*, *tree* and *mesh*. More complex networks can be built as *hybrids* of two or more of the above basic topologies. The ***logical topology*** defines how the systems communicate across the physical topologies. The two most common types of logical topologies are ***broadcast*** and ***token passing***.

When we decide which topology we should choose for our network, then there are few basic point that we need to take care. The factors that decide which topology we should choose are:

- ***Cost*** : Cost is a factor that plays an important role for the decision of topology. If we want to create a network for 4-5 computer, we should not expense very much in network.
- ***Scalability***: What is the size of the network that we need to make and is it possible in that kind of topology
- ***Bandwidth capacity****:* The required speed of the network that can be taken care by any particular topology.
- ***Ease of installation***: Is it easy to install the network using selected topology.
- ***Ease of fault finding and maintenance***: If we have a network we will definitely get the problem also; so will it be easy for network administrator to identify the problem and give the solution with ease and least possible time.

1.9.1 Bus Topology

A network of bus topology consists of a long single cable to which all the computers and other devices are connected. Any node attached to the bus can send signals down the cable to all the nodes of the network; that means a bus is a broadcast medium. When more than one node starts sending data through the bus, they mix with each other and the sent data become a garbage. This is called *collision.* To avoid collision there must be some agreement between the nodes so that when one computer starts to send data, others refrain themselves from sending data. To ensure correct data communication, both ends of the cable is terminated by a special device called end terminator.

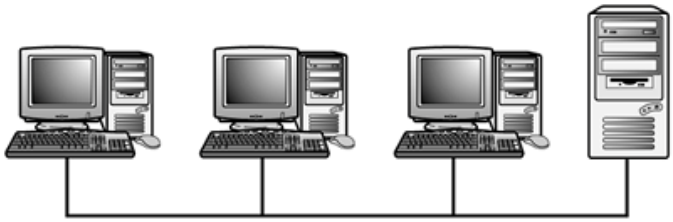


Fig.1.7 : A Bus Topology

Ethernet bus topologies are relatively easy to install and don't require much cabling compared to the alternatives. 10Base-2 (Thin) and 10Base-5 (Thick) both were popular Ethernet cabling options for bus topologies. Some advantages and disadvantages of the bus topology are listed below:

Advantages

- Bus topology is inexpensive in installation
- It requires less cable than other topologies
- Good for smaller networks not requiring higher speed.
- Easy to add systems to network.

Disadvantage

- Out-of-date technology. Bus topology was used in the early days of networking because it was inexpensive to use and relatively easy to set up.

- If the backbone cable fails, the entire network effectively becomes unusable.
- Unmanageable in a large network. If more than a few dozen computers are added to a network bus, performance problems will likely result.
- Difficult to troubleshoot.

1.9.2 Ring Topology

In ring topology the computers are connected between each other in such a way that form a close loop. In practice a cable connects the first computer to the second computer, another cable connect the second computer to the third and so on until the last computer is connected back to the first to completes the loop. It should be noted that the topology may not physically looks like a circle. The ring means the computers are connected with each other in a logical ring. The interconnecting cables may take any shape in practice. All messages travel through a ring in the same direction (either “clockwise” or “counterclockwise”).

In ring topology the communicating computers must follow some agreement between them to avoid collision as it is also a broadcast network. The main difference between the bus and ring is that the ring topology does not require termination. Because the systems are connected all together in a loop, there is no beginning and end point

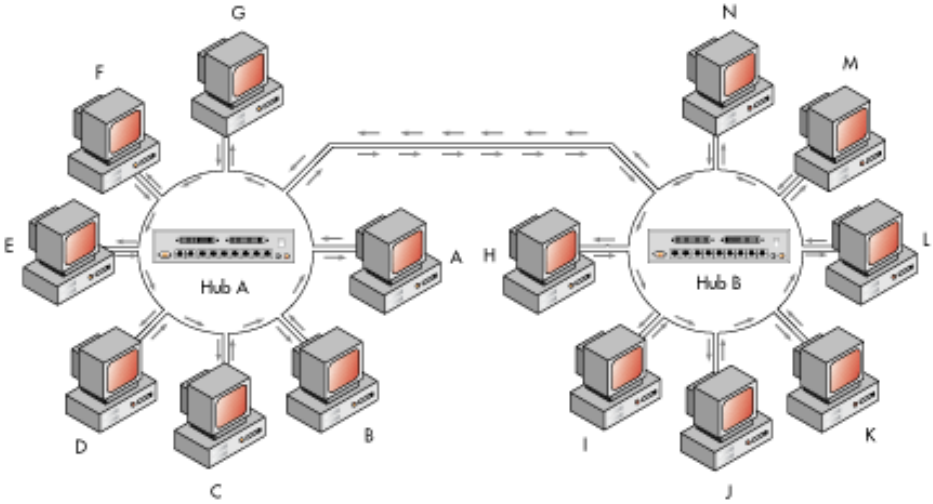


Fig. 1.8 : A dual ring topology

as there is with the bus topology. A failure in any cable or device breaks the loop and can take down the entire network.

Two major network types use the ring topology are :

i) *Fiber Distributed Data Interface* (FDDI) where a large, high speed networks use fiber optic cables in a physical ring topology.

ii) *Token-Ring networks* that use logical ring topology.

Advantages

- No collision of data as data travel in one direction only.
- Easier to fault find. If any point gets broken we can trace that easily.
- No terminator required.

Disadvantages

- Ring topology requires more cable than bus topology
- A break in ring will take the whole network down.
- Addition or removal of any node can affect entire network.

1.9.3 Star Topology

In the *star* topology, all computers and other network devices connect to a central device(controller) called a ***hub*** as depicted in the figure.1.9. Each connected device requires a single cable to be connected to the hub. A hub normally accept data from a sending computer and delivers it to the computer for which the data is addressed. Hence a star network is not a broadcast network, rather a point-to-point network.

Using a separate cable to connect to the hub allows the network to be expanded without disruption to the network. Because each computer uses a separate cable to connect to the hub, the failure of a network connection affects only the single machine concern. The other computers can continue to function normally.

Fast Ethernet (100Base-TX & 100Base-FX) in a star topology is the most commonly used LAN today. *10Base-T Ethernet* and *1000Base-TX Gigabit Ethernet* also use star topology.

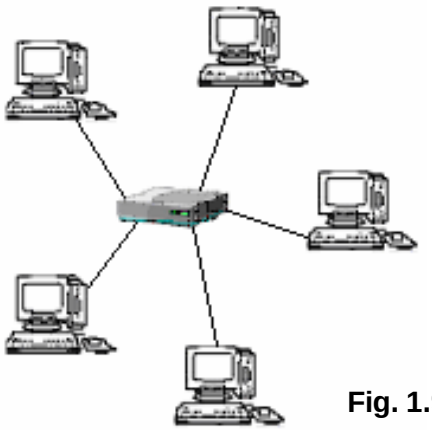


Fig. 1.9 : A Star Topology

Adavntages :

- Star networks are easily expanded without disruption to the network.
- Easy to add/remove devices to/from network
- One break does not bring whole network down. Cable failure affects only a single user.
- Easy to troubleshoot and isolate problems.
- Widely used centralized management

Disadvantages:

- Costs are usually higher than with bus or ring networks.
- Requires more cable than most of the other topologies.
- If the hub fails, any device connected to it will not be able to access the network.

In the following table, various networks with their cable types, topologies are shown:

Network Type	Standard	Cable Type	Topology
Ethernet	10Base-2	Thin(RG-58)	Bus
Ethernet	10Base-5	Thick(RG-59)coaxial	Bus
Ethernet	10Base-T	CAT3 or CAT5 UTP	Star
Fast Ethernet	100Base-TX	CAT5 UTP	Star
Gibabit Ethernet	1000BaseTX	CAT5, 5e or UTP	Star
Token Ring	all	UTP or STP	Logical Ring

Table 1.1 : Network cable types and topologies

1.9.4 Mesh Topology

The mesh topology incorporates a unique network design in which each computer on the network connects to every other, creating a point-to-point connection between every device on the network. A mesh topology is used when there can be absolutely no break in communications, for example the control systems of a nuclear power plant. The purpose of the mesh topology is to provide a high level of redundancy. If a network cable, computers or other components fails, the data always has an alternative path to get to its destination.

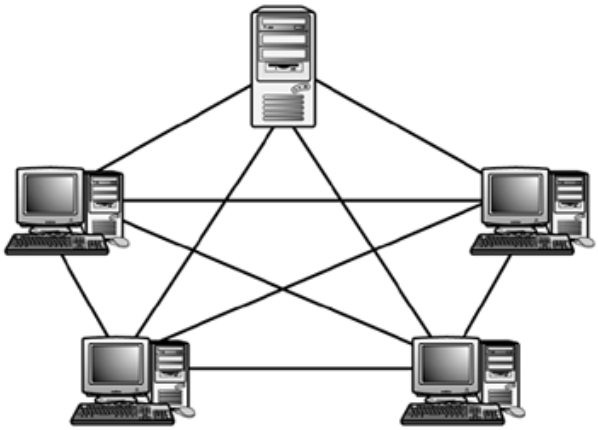


Fig. 1.10 : A Mesh Topology

A fully connected mesh network has n(n-1)/2 cables to link 'n' devices. Therefore, every device on the network must have 'n-1' input/output (I/O) ports. For example, in the figure(Fig.1.10), we have five systems that require 10 cables to create a mesh network. This topology is mainly used in environments where high availability outweighs the costs associated with this amount of interconnection. We can see in the figure that the wiring for a mesh network can be very complicated. Further, troubleshooting a failed cable can be tricky. Because of this, the mesh topology is rarely used.

Advantages:

- Provides alternative paths between devices in the network.
- The network can be expanded without disruption to current users.

Disadvantages :

- It is expensive as because a large number of cabling are required.

- Routing network traffic can be difficult because of all the different possible paths between nodes.
- It is very expensive to wire up.

There are also partial-mesh networks where some of the nodes are connected to all the others, but others are only connected to nodes with which they exchange the most data.

1.9.5 Tree Topology

A tree topology combines characteristics of bus and star topologies. It consists of groups of star-configured workstations connected to a linear bus backbone cable (fig. 1.11). This bus/star hybrid approach supports future expandability of the network much better than a bus (limited in the number of devices due to the broadcast traffic it generates) or a star (limited by the number of hub connection points) alone.

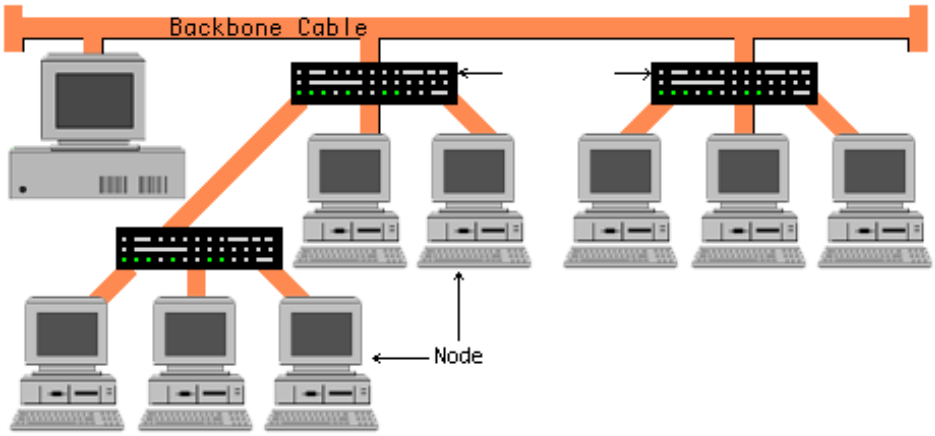


Fig. 1.11 : A Tree Topology

Advantages of Tree Topology

- The network is easy to extend by just adding another branch.
- Fault isolation is relatively easy.

Disadvantages of Tree Topology

- If the backbone cable breaks, the entire network goes down.
- More difficult to configure and wire than other topologies.
- If any hub goes down, all branches of that hub go down.

CHECK YOUR PROGRESS-2

1. Fill in the blanks:
   i) _____ defines the physical or logical arrangement of links in a network.
   ii) In _____ topology, every devices has a dedicated point-to-point link to every other device.
   iii) _____ topology is the combination of different types of topologies.
   iv) In _____ topology, each device has a dedicated point-to-point link only to a central controller.
   v) _____ topology provides alternative paths between devices in the network.
   vi) Ring topology requires _____ cables than bus topology.
   vii) The topology used in Ethernet is _____ topology.
   viii) _____ topology uses central controller or hub.
2. What topology is used by 10Base-5 and 10Base-2 Ethernet?
3. What topology is used by FDDI ?

1.10 TRANSMISSION TYPES

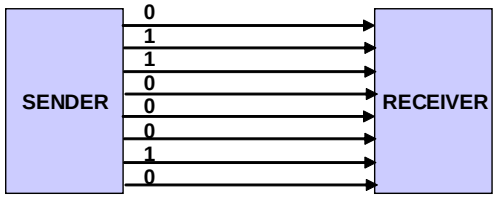
Digital transmission is the sending of information over a physical communications media in the form of digital signals. Analog signals must therefore be digitised first before being transmitted. However, as digital information cannot be sent directly in the form of 0s and 1s, it must be encoded in the form of a signal with two states, for example:

- two voltage levels with respect to earth
- the difference in voltage between two wires
- the presence/absence of current in a wire
- the presence/absence of light

Digital data transmission can occur in two basic modes:***parallel*** and ***serial.*** Data within a computer system is transmitted via parallel mode on buses with the width of the parallel bus matched to the word size of the computer system. Data between computer systems is usually transmitted in bit serial mode. Consequently, it is necessary to make a parallel-to-serial conversion at a computer interface when sending data from a computer system into a network and a serial-to-parallel conversion at a computer interface when receiving information from a network. The type of transmission mode used may also depend upon distance and required data rate.

1.10.1 Parallel Transmission

Parallel transmission communicates bits simultaneously over multiple lines(wires, frequency channels); typically the total consists of one or more bytes at a time. Parallel devices have a wider data bus than serial devices and can therefore transfer data in words of one or more bytes at a time. As a result, there is a speedup in parallel transmission bit rate over serial transmission bit rate. The timing for parallel transmission is provided by a constant clocking signal sent over a separate wire within the parallel cable; thus parallel transmission is considered synchronous. Computers are typically connected to printers and external disk drives via parallel interfaces, ports, and buses..



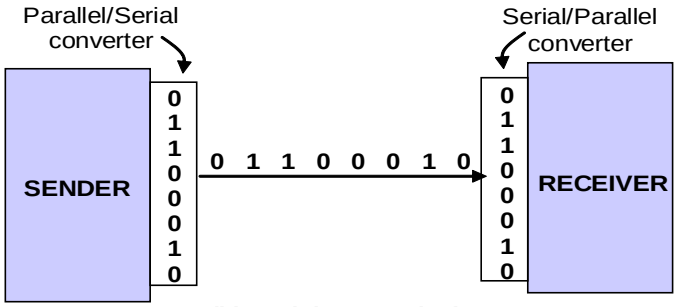
(a) Parallel Transmission;
The eight bits are sent through eight wires at a time

Fig. 1.12

1.10.2 Serial Transmission

Serial transmission sends one bit at a time over a single transmission line. The cost of communication hardware is considerably reduced since only a single wire or channel is require for the serial bit transmission which also slows the speed of transmission. Telephone lines use serial transmission for digital data, thus modems are connected to the computer via a serial port. A ***serial port*** is a socket on

a computer used to connect the serial interface to a serial line or bus. A ***serial interface*** is a data channel that transfers digital data serially; it is typically implemented as a card that plug into an expansion slot on a computer motherboard. Serial interfaces have multiple lines, but only one is used for data. An external ***serial bus*** carries serial data to any device connected to it, e.g. Ethernet. Serial transmission can be either *synchronous* or *asynchronous*.



(b) Serial Transmission;
The eight bits are sent one after another through a single wire

Fig. 1.13

Serial transmission technology is increasingly used for the transmission of digital data. A large number of up-to-date communications networks apply serial transmission. The numerous applications include computer networks for office communications, building and manufacturing automation, and finally, Internet. The transmission of a stream of bits from one device to another accross a transmission media involves a great deal of cooperation and agreement between the two sides (sender and receiver). One of the most fundamental requirements is ***synchronization***. The receiver must know the rate at which bits are being received so that it can sample the medium at appropriate intervals to determine the value of each received bit. For achieving the desired synchronization, there are two approaches. ***Synchronous*** and ***asynchronous transmissions*** are two different methods of transmission synchronization.

1.10.2.1 Asynchronous Transmission

Communication is called *asynchronous* if the transmitter and receiver do not need to synchronise before each transmission. A

sender can wait arbitrarily long between transmissions and the receiver must be ready to receive data when it arrives. Most PC serial devices such as mouse, keyboards and modems are asynchronous.

As the name implies, asynchronous communication is performed between two or more devices which operate on independent clocks. Thus, there is no guarantee that when Point A begins transmitting, Point B will begin receiving, or that Point B will continue to sample at the rate Point A transmits. In asynchronous transmission, data are transmitted one character at a time, where each character is five to eight bits in length. Timing or synchronization must only be maintained within each character; the receiver has the opportunity to resynchronize at the beginning of each new character. The following figure (Fig.1.14) depicts what happens when transmission clocks differ significantly. In fig.1.14 (a), we see that, the receiver samples at mid point of each bit of the incoming data. In fig.1.14 (b), the receiver clock is too slow; causing bit 4 to be skipped and as a result, the data is corrupted.To combat this type of timing problem, asynchronous communication requires additional bits to be added around actual data in order to maintain signal integrity. The bits of the character are transmitted beginning with the least significant bit.

Asynchronously transmitted data is preceded with a ***start bit*** which indicates to the receiver that a character is about to begin. The end of a character is followed by a ***stop bit***, which tells the receiver that the character has come to an end, that it should begin looking for the next start bit, and that any bits it receives before getting the start bit should be ignored. To avoid confusion with other bits, the start bit is twice the size of any other bit in the transmission. To ensure data integrity, a ***parity bit*** is often added between the last bit of data and the stop bit. The parity bit is set by the transmitter such that the total number of ones in the character, including the parity bit, is even (even parity) or odd (odd parity), depending on the convention being used. The receiver uses this

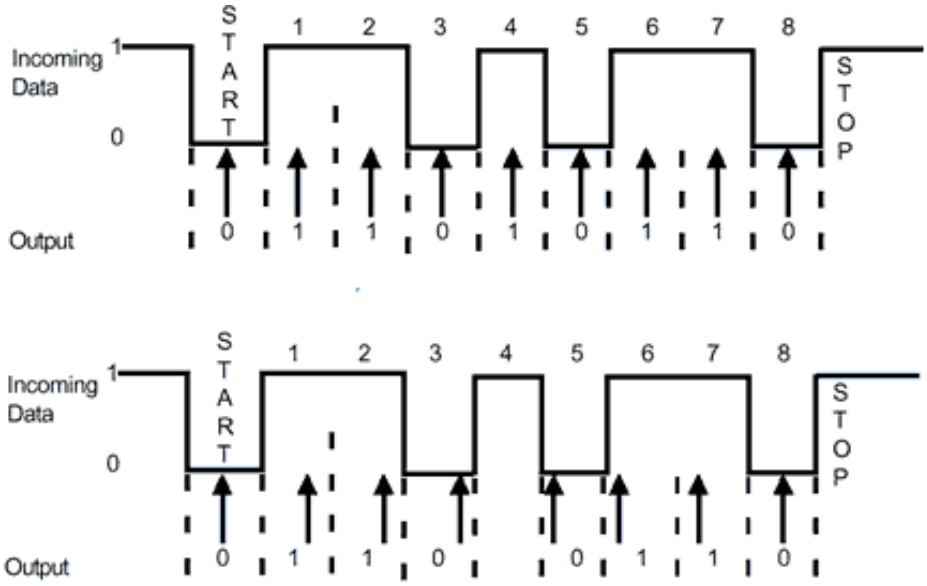


Fig. 1.14 : Asynchronous Data Sampling

bit for error detection. The parity bit makes sure that the data to be received is composed of the same number of bits in the same order in which they were sent.

1.10.2.2 Synchronous Transmission

The term synchronous is used to describe a data transfer method in which a continuous stream of data signals is accompanied by timing signals (generated by an electronic clock) to ensure that the transmitter and the receiver are synchronized with one another. These types of connections are used when large amounts of data must be transferred very quickly from one location to the other. The speed of the synchronous connection is attained by transferring data in large blocks instead of individual characters. A block of bits is transmitted in a steady stream without start and stop bits. Data or information is moved from one place to another at instants in time that are measured against the clock signal being used. This signal is usually comprised of one or more high frequency rectangular shaped waveforms, generated by special purpose clock circuitry. These pulsed waveforms are connected to all the devices that operate synchronously, allowing them to start and stop operations with respect to the clock waveform.

Typical examples of synchronous signals include the transfer and retrieval of address information within a computer via the use of an *address bus*. For example, when a processor places an address on the address bus, it will hold it there for a specific period of time. Within this interval, a particular device inside the computer will identify itself as the one being addressed and acknowledge the commencement of an operation related to that address.

In synchronous transmission, each block begins with a preamble bit pattern and generally ends with a postamble bit pattern. In addition, there are some other bits that convey control information. The data with the preamble, postamble, and control information are termed as a ***frame***. A general frame format for synchronous transmission is shown in Figure (1.15). Typically, the frame starts with a preamble called a flag, which is 8 bits long.

8-bit flag	Control fields	Data fields	Control fields	8-bit flag

Fig. 1.15 : Synchronous Frame Format

The same flag is used as a postamble. The receiver looks for the occurance of the flag pattern to signal the start of a frame. This is followed by some number of control fields, then a data field, more control fields, and finally the flag is repeated.

1.11 MODES OF COMMUNICATION

Transmission modes mean the way in which a communication is achieved between two linked devices. The device which sends data or information is called the sender and which receives the information is called the receiver. A channel can support either one way communication or two way communication at a time. Based on the way of communication link, transmission modes can be classified as : *simplex*, *half duplex* and *full duplex*.

- ***Simplex :*** A *simplex* connection is a connection in which the data flows in only one direction, from the transmitter to the receiver. This type of connection is useful if the data do not need to flow in both

directions, for example, from our computer to the printer or from the mouse to our computer. It allows one-way communication of data through the network, with the full bandwidth of the cable being used for the transmitting signal.

- ***Half-duplex :*** A *half-duplex* system provides for communication in both directions, but only one direction at a time (not simultaneously). Typically, once a party begins receiving a signal, it must wait for the transmitter to stop transmitting, before replying. This type of connection makes it possible to have bidirectional communications using the full capacity of the line. A good analogy for a half-duplex system would be a one-lane road with traffic controllers at each end. Traffic can flow in both directions, but only one direction at a time, regulated by the traffic controllers. Many networks are configured for half-duplex communication. One suitable example of a half-duplex system is a two-"walkie-talkie".

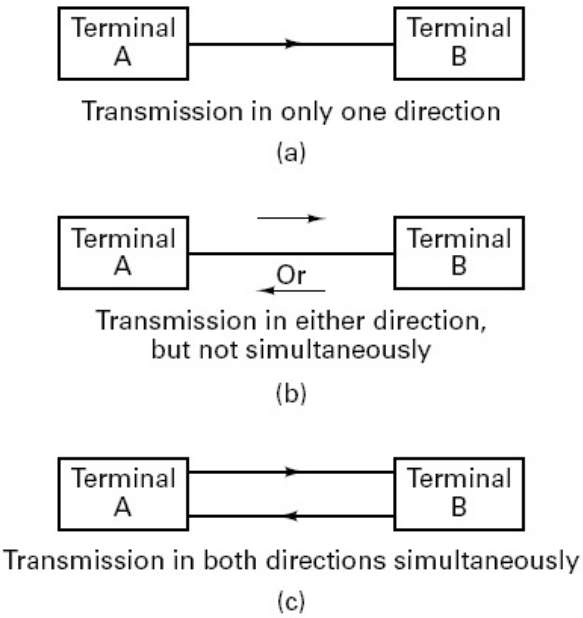


Fig.1.16 (a)Simplex mode (b) Half-duplex mode and (c) Full-duplex mode

- ***Full- duplex*** *:* The preferred transmission mode for network communication is the *full-duplex* mode. A *full-duplex*, allows communication in both directions simultaneously. Each end of the line can thus transmit and receive at the same time, which means that the bandwidth is divided in two for each direction of data transmission if the same transmission medium is used for both directions of transmis-

sion. A good analogy for a *full-duplex* system would be a two-lane road with one lane for each direction. For example, telephone networks are full-duplex, since they allow both callers to speak and be heard at the same time.

1.12 PROTOCOLS AND STANDARDS

Many different types of network protocols and standards are required to ensure that a computer can communicate with another computer located on the next desk or any location around the world.

A network protocol is a set of rules that governs the communications between computers and other devices on a network. These rules include guidelines that regulate the characteristics of a network such as access method, allowed physical topologies, types of cabling, and speed of data transfer. Some protocols also support message acknowledgement and data compression designed for reliable and/or high-performance network communication. Hundreds of different computer network protocols have been developed each designed for specific purposes and environments.

The seven-layer OSI (*Open systems Interconnection*) model, created by the ISO (*International Standards Organization*), defines internetworking environments. It provides a description of how software and hardware interact to permit communication between computers. An *interface* separates each layer from those above and below it; each layer provides services to the layer directly above it. We will learn the role and functionality of these layers in somewhat details in Unit 2.

CHECK YOUR PROGRESS-3

1. Choose the correct answer:

   i) In simplex mode of transmission

      a) Data transmission is one way

      b) Data can be transmitted to small distances only

      c) Data format is simple

d) None of the above

ii) In half-duplex data transmission

a) Data can be transmitted in one direction only

b) Data can be transmitted in both directions

c) Data can be transmitted in both directions simultaneously

d) None of the above

iii) In _____ transmission, a start bit and a stop bit form a character byte.

a) asynchronous b) synchronous

c) parallel d) none of these

iv) _____ communicates bits simultaneously over multiple lines.

a) serial transmission b) synchronous communication

c) asynchronous communication

d) parallel transmission

1.13 LET US SUM UP

This unit provides the basic concept of computer networking. **Networking** is the linking of computers (not necessarily over large distances) so they can communicate, sharing hardware and software, thus uniting processing power. Various concepts such as connection-oriented and connectionless services, broadcast and point-point networks etc. are also presented in this unit. We are also acquainted with the different types of computer networks with their relative advantages and disadvantages. It also provides the concept of network topologies. Apart from this, types of transmission and communication modes are also covered towards the end. A brief introduction of protocol and standerd are also included in this unit.

1.14 ANSWERS TO CHECK YOUR PROGRESS

CHECK YOUR PROGRESS-1

i) True ii) False iii) True iv) True

CHECK YOUR PROGRESS-2

1. i) Topology ii) mesh iii) Hybrid iv) star
   v) Mesh vi) more vii) bus viii) star
2. Bus topology 3. Ring topology

CHECK YOUR PROGRESS-3

1. i) a) Data transmission is one way
   ii) b) Data can be transmitted in both directions
   iii) a) asynchronous iv) d) parallel transmission

1.15 FURTHER READINGS

1. *"Computer Netwotks",* Andrew S. Tanenbaum, Prentice Hall India.
2. *"Data and Computer Communications",* William Stallings, Pearson Prentice Hall.

1.16 MODEL QUESTIONS

1. Define computer network. What are the goals of computer networks?
2. What is the difference between circuit switching and packet switching?
3. Explain the concept of message switching.
4. What are the advantages and disadvantages of circuit switching?
5. Compare and contrast different network topologies.
6. List the relative advantages and disadvantages of bus, ring and star topologies.
7. Differentiate between connection-less and connection-oriented networks.
8. What are the different modes of communication? Distinguish simplex, half-duplex, and full-duplex mode of transmission.
9. Distinguish between peer-to-peer LAN and client-server LAN.
10. Describe briefly the various features of serial and parallel transmission.

UNIT-2 : NETWORK MODELS

UNIT STRUCTURE

2.1 LEARNING OBJECTIVES

After going through this unit, you will be able to :

- describe design issues of network architecture.
- describe functions of various network layers.
- define various network terminologies.
- distinguish between connection-oriented and connection-less service.
- compare between OSI and TCP/IP model.

2.2 INTRODUCTION

The subject of computer network and its structure can be best understood by studying its underlying communication software and the related hardware. Historically first computer networks were designed to work on hardware which proved to be fatal afterwards. It leads to inflexibility to the computer networks. The later development of computer networks were based

primarily on highly structured software implemented on ever changing hardware. The communication software in a computer network is a much thought out software which is organized as layered software. The internal structure of the layered software is discussed in this unit.

2.3 DESIGN ISSUES OF THE LAYER

Before proceeding to learn the layered structure of network software and various aspect of computer network, we should be acquainted with different design issues of layer.

- To interchange data between two processes running on two computers there must be some addressing technique for communication correctly.
- The communication may be unidirectional (simplex communication) or in both direction alternately one after another (half-duplex communication) or bidirectional (full-duplex communication).
- Error control is also important issue because physical communication channels are not perfect, rather error prone. Now both communicating parties must agree on which error detecting and correcting code to use. The receiver should also let the sender know which message is received correctly and which is not.
- During transfer, messages are often divided into pieces and all the pieces may not be transferred in proper order. This loss of sequencing has to be taken care of by the protocol concern.
- There must be provision to refrain a fast sender from swamping a slow receiver.
- The layer concern may decide to use same connection for several unrelated conversations by multiplexing and de-multiplexing technique.
- When there are many paths between sender and receiver, a decision must be taken which route is to take. The decision may be static or dynamic.

2.4 PROTOCOL HIERARCHY

From the design issues discussed above it is apparent that the computer network design is a complex job. To reduce the complexity of design, most of the computer networks are organized as a stack of layers, one above another, taking the service offered by the layer below. In this structure, a lower layer gives certain services to the layer just above it, at the same time it shields the details of the implementation of those services from the upper layer.

In the layered organization, a particular layer on one machine talks with a peer layer on another machine. During the conversation, they follow some rules and conventions. These rules and conventions are collectively known as **protocol**. Here we have used the term **peer** which means entities comprising the corresponding layer on different machines. So, it is the peers that talk using their own protocol.

In Fig 2.1 a five layer network is shown. Here we can see dotted arrow between peers and solid arrow between any two adjacent layers. Dotted arrow indicates that layer *n* on one machine communicates with layer *n* on the other machine. During communication they follow their protocol.

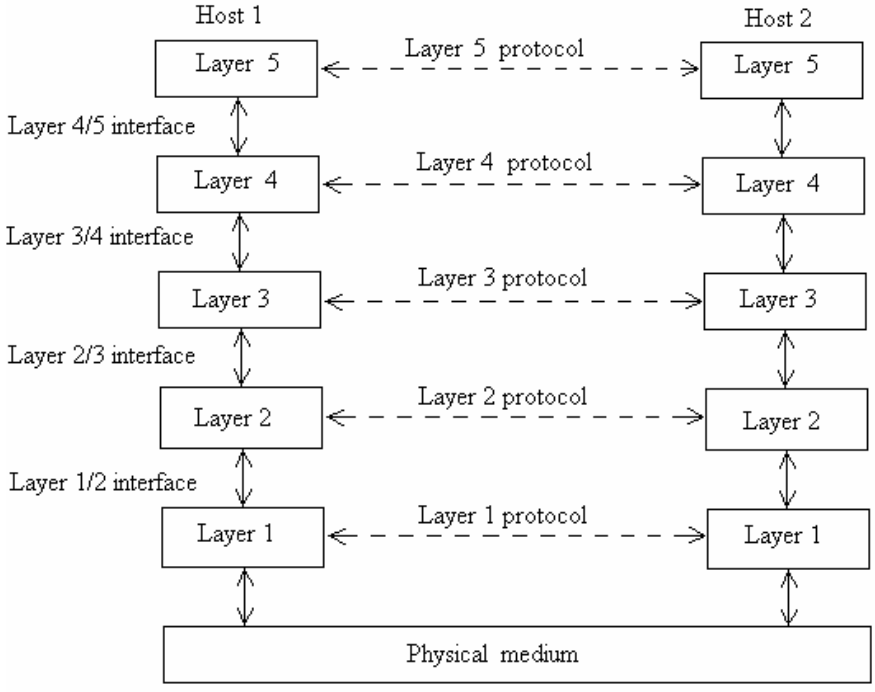


Fig. 2.1 : Layers, protocols and interfaces

Every layer communicates with the peer layer on the other machine. But all these communications are virtual since no data really moves through these dotted lines; rather data moves through solid arrow lines. In the process of communication, each layer hands over their data to the layer immediately below it. In this way data moves downward until the lowest layer, below which lies the physical medium through which actual communication takes place. This is the picture in the sending machine. The picture is just opposite in the receiving machine where data moves upward after it enters the computer from the physical medium.

In the Fig 2.1 each layer gets **service** from the layer just below it. A layer performs some primitive operations to give the service required by the immediate upper layer. The transaction between two adjacent layers takes place as per a predefined guideline where each layer performs a specific collection of functions. So, there exists an **interface** between each pair of adjacent layer through which they exchange data and information and in the process, the upper layer gets service from the lower layer. As long as the interface is unchanged, one can change the underlying hardware at any time without affecting the network operations. As for example underground copper cables are gradually replaced by fiber optic cables but it is not hampering the operations of the existing network because the interfaces between the layers remain unchanged.

The layers and their protocols are collectively known as network **architecture**. In network architecture each layer has its own protocol. The protocols used by all the layers of a network system is called **protocol stack**. Stack means something arranged one above another, like stack of bricks or something similar. In a layered network architecture, layers are arranged from bottom to top, one upon another, hence their respective protocols are also lies one upon another. That's why the name protocol stacks. In the same sense, protocol hierarchy indicates that the protocol of any layer is hierarchically above the protocol of the layer immediately below it.

2.5 ISO-OSI REFERENCE MODEL

The OSI reference model is shown in Fig 2.2. The International Standard Organization (ISO) developed a proposal for a network model and the re-

sulting model is known as ISO – OSI reference model. This model has seven layers and the layers are :

i) Application layer
ii) Presentation layer
iii) Session layer
iv) Transport layer
v) Network layer
vi) Data link layer
vii) Physical layer

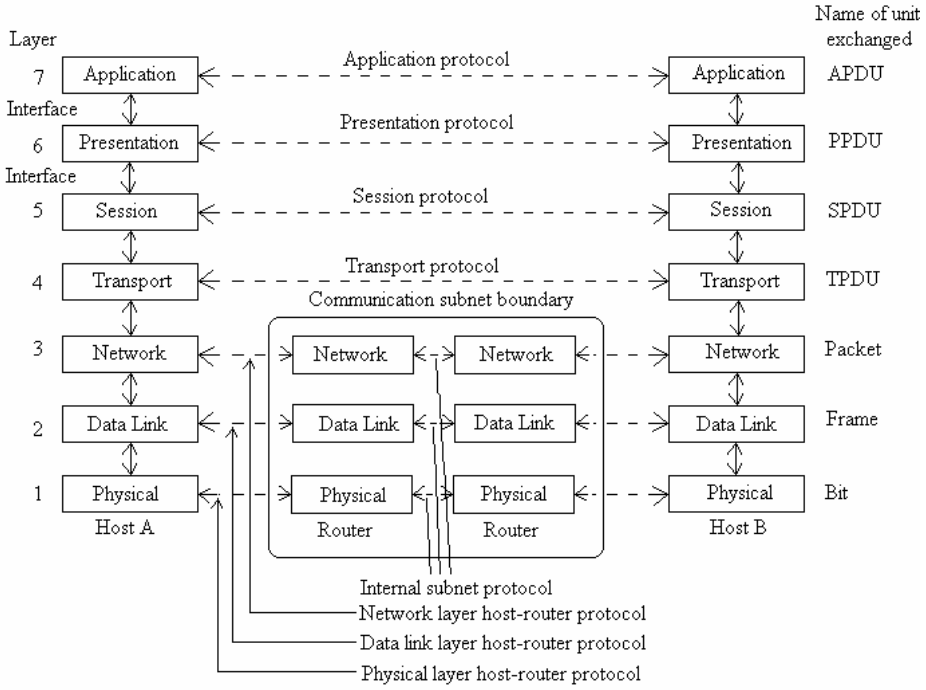


Fig. 2.1 : Layers, protocols and interfaces

(i) The Physical Layer : The physical layer is responsible for transmitting raw bits over the communication channel. This layer is to ensure how to send a 1 bit from the sending computer as 1 bit to the receiving computer; not as 0 bit. The physical layer also deals with the issues of how many bit per second will be transmitted, what level of voltage will be used to represent 1 and 0, whether transmission will be unidirectional or bidirectional, how the initial connection be established and terminated at the end, how many pins of network connector has and which pin is for what etc etc. Hence the design issues of the physical layer is mostly mechanical, electrical and procedure oriented.

(ii) The Data Link Layer : The data link layer used the raw transmission facility and transforms it to an apparently error free facility to be used by the network layer. This layer breaks the input data into frames by inserting appropriate frame boundary, transmit the frames sequentially and process the acknowledgment sent back by the receiving computer. If a frame is completely destroyed by noise burst, it is the duty of data link layer to retransmit it from the source machine. Data link layer also ensures that a fast sender be not allowed to swamp a slow receiver by sending data at a higher rate than it can be handled by the receiver. This is called flow control. In a broadcast network it is the duty of MAC (Medium Access Control) sub-layer of data link layer to decide who will access the transmission medium at a particular time.

(iii) The Network Layer : The network layer controls the operation of the subnet. The layer is to determine how packets are routed from source to destination. The routing may be static or dynamic depending on traffic load, and availability of channel.

Too many packets may cause congestion (traffic-jam) and control of such congestion is also a duty of network layer.

The subnet operation require cost, hence some accounting function is also there built into the network layer. When a packet crosses national boundary some other aspects of accounting has to be dealt with by network layer.

Packets have to travel in between heterogeneous network running on different platforms using different network protocol and network layer is also responsible to resolve all the problems arises out of such situations.

(iv) The Transport Layer : The transport layer is to accept data from the session layer, break it into smaller units if necessary, hand over these to the network layer and ensures that the pieces all delivered correctly to the receiver. The above duties must be done efficiently and in such a way that it will not effect the upper layer in case there is any change in the hardware.

Normally, the transport layer creates individual connection for each session. If high throughput is required, the transport layer may es-

tablish multiple network connections, dividing the data among individual connection thereby improving the through put.

To reduce cost the transport layer may also multiplex several transport connections on to the network connection. However, multiple connections or multiplexing must not be seen by the session layer.

The transport layer also determines the type of service given to the user of the network. Error free point-to-point connection is the most popular transport layer service where messages or bytes are delivered in order in which they were sent. Another kind of transport service is the transportation of isolated messages with no guarantee of delivery. The transport layer is a true end layer between source to destination.

The transport layer also performs flow control.

(v) The Session Layer : This layer offers facility to different users on different computer to establish session between them. A session allow an user to remotely log into a distant machine and transfer file between the two machines. Session layer perform token management to provide unidirectional communication. It also provides a service called synchronization.

(vi) The Presentation Layer : This layer performs data presentation job by following syntax and semantics rules. Before presenting data to the user, it transform data into their acceptable form. For example, say we write a name as Sri Nilimoy Choudhury, whereas it will be written as Choudhury, Mr. Nilimoy in Europe or USA. We write date as dd/mm/yyyy, currency as Rs. while in western country it is written as mm/dd/yyyy and $ etc. So, these conversions are done by the presentation layer.

(vii) The Application Layer : This layer is the nearest layer to all the network users. It offers variety of protocols that are commonly needed. It helps to transfer file. Different file systems have different meaning in different machines with different data format etc. When files are transferred from one machine to another with different file system, the application layer take necessary steps to resolve the abnormalities.

CHECK YOUR PROGRESS-1

1. What is *service* and interface?
2. What do you mean by network architecture?
3. How many layers are there in ISO-OSI model?
4. What are the prime duties of Data Link Layer?

2.6 TERMINOLOGY

When we study the subject of computer network, we encounter many terminologies associated with the subject. Already we have come across a few terminologies such as **peer, interface, protocol, service** etc. Some others are briefly described below:

- **Entity :** The active components in each layer are called entities. Entity can be a software process or hardware like an intelligent input/output device etc.
- **Peer Entity:** Entities in the same layer running on different machines are called peer entities.
- **Service provider and Service user:** In layered architecture, layer *n* provides a service which is used by layer *n+1*. Here layer *n* is service provider and layer *n+1* is service user.
- **SAP (Service Access Points):** A layer *n* offers services to layer *n+1* at a place which is called service access points. Each SAP has a unique address for identification.

2.7 CONNECTION-ORIENTED AND CONNECTIONLESS SERVICE

A layer can offers two different types of services to the layer immediately above it – connection-oriented and connectionless service.

In connection-oriented service, the service user establish a connection at first, communicates over the connection and at last release the connection. It is similar to a telephone system.

In connectionless service, no connection is established beforehand. Instead, like in a postal system, every massage carries its full destination address and each routed independently to its destination. If a big message is broken into pieces and sent to the same destination, then in connectionless service sometime it may happen that the first piece may arrive at the destination after the latter pieces. That means at the receiving end the order of delivery may not be same as the order of transmission. In connection-oriented service this never happens.

2.8 THE INTERNET (TCP/IP) REFERENCE MODEL

The Internet model of network architecture, also known as TCP/IP reference model, was developed much earlier than the OSI reference model. It was evolved from the US Department of Defense's (DoD) research network- The ARPANET. Eventually the ARPANET connected many universities and other government organizations. During the process of interconnecting all these through the existing telephone lines, satellite link and radio, the protocol used for the ARPANET had trouble and therefore, a new architecture was needed to overcome that. The architecture thus came out became known as the INTERNET or TCP/IP Reference Model. The model has four layers, namely :

1. The Host-to-Network Layer.
2. The Internet Layer.
3. The transport Layer.
4. The Application Layer.

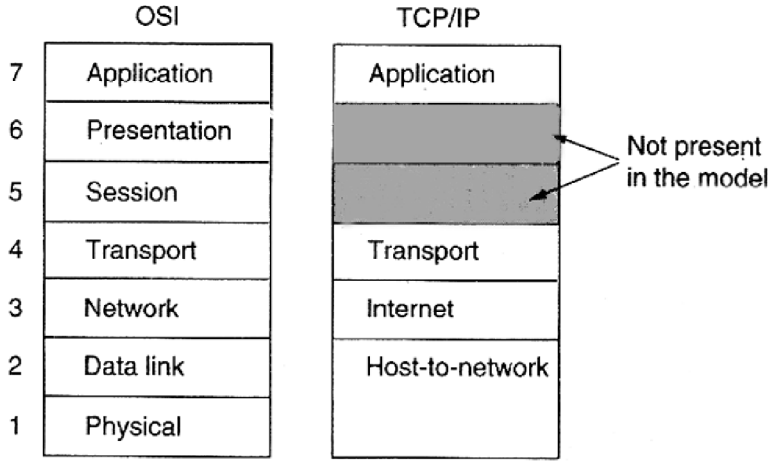


Fig. 2.3 The TCP/IP reference model

1. **The Host-to-Network Layer :** In TCP/IP reference model the bottom layer is not defined clearly. Yet we can consider that whatever is there below the internet layer is the bottom layer in the TCP/IP model. In the model, the host has to connect to the network using some protocol so that it can send IP packets over it.

2. **The Internet Layer :** The DoD planned to set up their inter-network in such a way that it had to survived even if a particular link fail in a probable war. So, this requirement led to connectionless packet-switching network instead of a connection-oriented circuit-switching network. The internet layer is designed to fulfill the goal of the architecture in such a way that it became the linchpin of the whole architecture. The job of this layer is to pump the packets from the host machine into any network and to help the packets to go to the destination independently. In this style of communication, packets may arrive in a different order than they were sent. This layer has its own protocol called **IP (Internet Protocol)** and a specified packet format. Packets routing as well as congestion avoidance are the major issues in this layer. This layer is similar to the network layer in OSI reference model.

3. **The Transport Layer :** The layer immediately above the internet layer is called the transport layer. This layer communicates with the peer layer on the other machine. It has two protocols namely **TCP and UDP.**

   TCP is a reliable connection-oriented protocol that ensures delivery of byte stream from source machine to destination machine. This layer breaks the byte stream receives from the upper layer into messages and passes them to the internet layer. At the receiving machine, these messages are reassembled by the TCP and pass them to the upper layer. Flow control is also another job of this layer to restrict a fast sender from swamping a slow receiver.

   UDP is another protocol used by this layer which offers an unreliable, connectionless service. Here, unlike TCP, sequencing and flow control is not done. When prompt delivery is more important than accurate delivery, UDP is used. It is also widely used in one time client-

server type communication. Email, video or sound transmissions are some of these applications.

4. **The Application Layer :** In OSI model, there are session and presentation layer above the transport layer. In TCP/IP model these two layers are absent. So, on top of the transport layer, the application layer is present in TCP/IP reference model. All high level protocols are present here. TELNET, FTP, SMTP are some of the early inclusions. Later, other protocols such as DNS, NNTP, HTTP are added to this layer.

2.9 COMPARISON OF ISO-OSI & TCP /IP REFERENCE MODEL

There are not much difference between the OSI and TCP/IP reference model. In both, the concept of independent stack of protocols is used. The function of the layers are also more or less same. In both model, starting from bottom up to transport layer, the functions of the layers is to provide end-to-end network independent transport service for communicating processes. The layers above the transport layer are application oriented.

On the other hand, there are some differences between the two models. In OSI reference model, three concepts are distinct. These are -

1. Services,
2. Interfaces,
3. Protocols.

The *Service* tells what a particular layer serves the layer just above it, not how these services are access by the above layer or how these are provided by the layer. The *Interface* tells how the above layer access the services provided by the layer just below it. It also does not say how these are provided. The *Protocols* used between peer layers are the set of rules agreed upon by both the layers to get the job done. Protocols can be changed without affecting the software in the higher layers.

In TCP/IP model, there is no such distinction between Service*, Interface and Protocol*. Therefore, in TCP/IP reference model the protocols are not

so hidden as in the OSI reference model. As the technology advances, protocol between a peer layer can be replaced easily in OSI model than in TCP/IP model.

In OSI model, the layered structure was thought out before the protocols were invented. The designers were new in the network technology and hence in some layer a sub-layer has to be provided latter to accommodate some new mode of communication. For example in data link layer, the MAC sub-layer was introduced latter when broadcast communication came to deal with channel allocation issues. On the other hand in TCP/IP model, the protocols came first and then the model. So there was no problem of fitting the protocols into the model.

The two models also differ in their numbers of layers; in OSI model there are seven layers whether TCP/IP model has four layers. Both have (inter)network, transport and application layers in common but the other layers are not same.

In OSI model, network layer supports both *connection-oriented and* connectionless communication and the transport layer supports only *connection-oriented* communication. On the contrary, in TCP/IP model the network layer supports only *connectionless* communication but transport layer supports both the mode i.e. *connectionless* and *connection-oriented* communication.

CHECK YOUR PROGRESS-2

1. What are the two services that can be offered by a layer to the layer above it?
2. How many layers are there in TCP/IP reference model?
3. Which model of network architecture was developed earlier – OSI or TCP/IP ?
4. Which two layers of OSI model are absent in TCP/IP model?

2.10 LET US SUM UP

- There must be some addressing technique for smooth communication between two computers.
- Simplex communication means one directional communication.
- In half-duplex communication, communication is bidirectional alternatively; not at the same time.
- Full-duplex communication means bidirectional communication at all time.
- Most of the computer networks are organized as a stack of layers, one above another, taking the service offered by the layer below.
- Protocol means a set of rules followed by two peers during communication.
- Entity: The active components in each layer are called entities. Entity can be a software process or hardware like an intelligent input/output device etc.
- Peer Entity: Entities in the same layer running on different machines are called peer entities.
- Service provider and Service user: In layered architecture, layer *n* provides a service which is used by layer *n+1*. Here layer *n* is service provider and layer *n+1* is service user.
- SAP (Service Access Points): A layer *n* offers services to layer *n+1* at a place which is called service access points. Each SAP has a unique address for identification.
- The layers and their protocols are collectively known as network architecture.
- OSI model is a seven layer network architecture.
- A layer can offers two different types of services to the layer immediately above it – connection-oriented and connectionless service.
- In TCP/IP model of network architecture, there are four layers.
- Most of the computer networks are based on TCP/IP model.

2.11 ANSWERS TO CHECK YOUR PROGRESS

CHECK YOUR PROGRESS-1

1. A layer performs some primitive operations to fulfill the requirement of the layer immediately above it. This responsibility of the lower layer is called service.

   On the other hand interface is the junction between two adjacent layers across which the lower layer provides the services to the layer above it.

2. The layers and their protocols are collectively known as network **architecture**.
3. Seven layers.
4. Framing, flow control , error control and access control.

CHECK YOUR PROGRESS-2

1. Connection-oriented service and connection-less service.
2. Four layers.
3. TCP/IP reference model was developed earlier than OSI model.
4. Session layer and presentation layer.

2.12 FURTHER READINGS

1. Tanenbaum, Andrew, Computer Networks, PHI.
2. Forouzan Behrouz A., Tata Mcgraw Hill.
3. Norton Peter, Complete Guide to Networking.
4. Comer, E. Douglas, Narayanan, M. S., Computer Networks and Internets with Internet Applications, Pearson Education.

2.13 MODEL QUESTIONS

1. What are the design issues of layered architecture of a computer network?
2. What is the principal difference between connection-less and connection-oriented communication ?

3. Describe ISO-OSI reference model of computer network.
4. Define : peer entity, protocol, protocol stack, SAP.
5. Which of the OSI layers handles each of the following:
   a) Breaking the transmitted bit stream into frames.
   b) Determining which route through the subnet to use.
6. Describe the functions of various layers of the TCP/IP model.
7. What are the responsibilities of the network layer in the TCP/IP model?
8. What are the responsibilities of the transport layer in the TCP/IP model ?
9. How do the layers of the TCP/IP model correlate to the layers of the OSI model ?
10. Match the following to one or more layers of the OSI model:
    a. Flow control
    b. Route determination
    c. Provides access for the end user
    d. Interface to the transmission media.

UNIT - 3 PHYSICAL LAYER

UNIT STRUCTURE

3.1 LEARNING OBJECTIVES

After going through this unit, you will be able to:

- learn about the basic concept of Physical Layer
- learn the concept of analog and digital signals
- describe digital and analog transmission
- learn about multiplexing and different multiplexing technique
- describe different signal transmission media
- learn about circuit switching and telephone network

3.2 INTRODUCTION TO PHYSICAL LAYER

In unit 2 we have learnt about different network models. In this unit we are going to discuss about the physical layer which is the lowest layer in the OSI model. It consists of different networking hardware transmission media and transmission methods of computer network. The physical layer provides services for the data link layer. Here transmission of raw bits over any hardware transmission medium is maintained. It provides an electrical, mechanical, and procedural interface to the network. The main design issue in this layer is to make sure that if 1 bit is sent from the source then it must be received at the other end as 1 bit.

3.3 SIGNALS

One of the major functions of the physical layer is to move data in the form of electromagnetic signals across a transmission medium. Both data and the electromagnetic signal can be either analog or digital in form.

3.3.1 ANALOG AND DIGITAL SIGNALS

An analog signal is a continuously varying electromagnetic wave for which the time varying feature of the signal is a representation of some other time varying quantity that may be propagated over a variety of media, depending on spectrum. For example, when someone speaks, an analog wav is created in the air. This can be captured by a microphone and converted to an analog signal. An analog signal has infinitely many levels of intensity over a period of time.

On the other hand a digital signal is a physical signal that has a sequence of voltage pulses that may be transmitted over a transmission medium. Data are stored in computer memory in the form of 0s and 1s that can be converted to digital signals.

The advantages of digital signaling over analog signaling are:

1. Digital signaling is generally cheaper than analog signaling
2. Digital signaling is less affected by noise interference than analog signaling.

The disadvantage of digital signaling is that digital signals suffer more from attenuation than analog signals.

Periodic and Nonperiodic signals:

Both analog and digital signals can be either periodic or nonperiodic in form.

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

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

The most fundamental form of a periodic analog signal is sine wave. A sine wave can be represented by three parameters: the peak amplitude, the frequency and the phase.

Now the absolute value of the highest intensity that proportional to the energy of a signal is called the peak amplitude. In case of electric signals, peak amplitude is measured in volts.

The amount of time required in seconds by a signal to complete 1 cycle is called period and the number of periods in one second is referred as frequency. So the period is the inverse of frequency.

Phase refers to the position of the waveform relative to time 0. Phase is measured in degrees or radians. A phase shift of 360° corresponds to a shift of a complete period and a phase shift of 180° corresponds to a shift of one-half of a period.

Wavelength is another characteristic of a signal traveling through a transmission medium. Wavelength can be calculated if we know the propagation speed and the period of the signal with the following formulae.

Wavelength=propagation speed x period

Composite signals:

Now in data communication we required to send composite signals which are made of many simple sine waves because a single frequency sine wave is not useful in data communication. According to Fourier analysis, any composite signal is a combination of simple sine waves with different frequencies, amplitudes and phases. A periodic composite signal is a collection of a series of signals with discrete frequencies and a

nonperiodic composite signal is a combination of sine waves with continuous frequencies.

Now the bandwidth of a composite signal refers to the range of frequencies contained in it. For example, if a composite signal contains frequencies between 1000 and 7000 then its bandwidth is 7000-1000 i.e. 4000.

Bit rate:

The term bit rate is used for digital signals which gives the number of bits sent in 1 second (bits per second) over a communication channel.

Bit length:

The bit length of a digital signal is the distance one bit occupies on the transmission medium which can be calculated by the following formulae.

Bit length=propagation speed X bit duration

3.3.2 DATA RATE LIMITS

In case of data communication, how fast data can be sent over a channel in bits per second is referred as data rate. Now there are three factors on which the data rate depends:

1. The bandwidth
2. The level of the signals
3. The quality of the channel

In case noiseless channel, according to Nyquist the theoretical maximum bit rate can be calculated by the following formulae:

Bit rate = 2 X B X $\log_2 N$

Here, B is the bandwidth of the channel, N is the number of signal levels and the bit rate is calculated in bits per second.

But in case of our real world data communications, it is not possible to have a noiseless channel. Claude Shannon developed a formula in 1944 to calculate the theoretical highest data rate for a noisy channel. This formula is called the Shannon capacity.

Capacity = B X $\log_2$ (1+SNR)

Here, B is the bandwidth of the channel, SNR is the signal to noise ratio and capacity is the bit rate of the channel in bits per second.

Throughput:

The throughput is the actual amount of data in bits per second that can be sent over a communication channel. So it is different from the bandwidth as bandwidth gives the total capacity of a channel in case of data transmission in bits per second. In general the throughput is always less than the bandwidth of a communication channel.

Latency:

The latency or delay is the amount of time required to arrive an entire message completely at the destination from the time the first bit is sent to the source. The latency is calculated by the following formulae:

Latency = propagation time+ transmission time+ queuing time+ processing delay

Propagation Time:

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

Propagation time= Distance / Propagation speed

The propagation speed of electromagnetic signals depends on two factors which are the communication medium and the frequency of the signal.

Transmission Time:

In data communications, the time required to arrive a complete message at the receiver from the sender is called the transmission time. In other words, we can say, the time between the first bit leaving the sender and the last bit arriving at the receiver of a message is called the transmission time. The transmission time of a message depends on the size of the message and the bandwidth of the channel.

Transmission time = Message size / Bandwidth

Queuing Time:

The queuing time is the time required for each intermediate or end device to hold the message before it can be processed. The queuing time changes with the amount of load available on the communication network. An intermediate device or end device like router processes the messages one by one in some order. If there are many messages, each message will have to wait. So if the load on the network increases, the queuing time increases.

3.3.3 TRANSMISSION IMPAIRMENT

After traveling through some transmission media, original signals at the source are changed at the destination .This means signal impairment occurs. There are three factors for signal impairment which are attenuation, distortion and noise.

Attenuation:

Attenuation means a loss of energy. When a signal is sent from source to destination over a communication channel or medium, it loses some of its energy because of the resistance present in the communication medium and some of the electrical energy in the signal is converted to heat. So this attenuation causes signal impairment. Amplifiers are used to amplify the signals and to minimize the signal attenuation. The decibel (dB) is the unit to define the relative strengths of two signals or one signal at two different points. The decibel is negative if a signal is attenuated and positive if a signal is amplified.

$$dB=10\log_{10}\frac{S_p}{S_q}$$

Variables S_p and S_q are the powers of a signal at points p and q, respectively.

Distortion:
When a signal changes its original form or shape in the time of movement from source to the final destination then it is called signal distortion . In case of a composite signal, each sine wave has its own propagation speed through the communication medium and own delay in arriving at the destination. So the delay of different sine waves have differences which means that signal components at the receiver have phases different from the components at the sender. So the shape of the composite signal is changed at the final destination.

Noise:

Noise is an unwanted signal which is the part of any recorded signal. So it is another cause of impairment. There are different types of noise like thermal noise, induced noise, crosstalk and impulse noise may corrupt the original signal. The random motion of electrons in a wire creates an extra signal called the thermal noise. Induced noises are created from sources such as motors and appliances. Crosstalk is the effect of sending antenna on the receiving antenna. Impulse noise is a signal with high energy in a very short time that comes from some sources like power lines, lightning etc.
Now the signal to noise ratio is defined as follows:

SNR=average signal power / average noise power

SNR can be described in decibel units as follows:

$SNR_{dB}=10\log_{10}SNR$

3.4 DIGITAL TRANSMISSION

In a computer network, information requires to be converted to either a digital signal or an analog signal for transmission from one machine to another machine. Now digital signal can be transmitted by using one of the two different approaches which are baseband transmission and broadband transmission.

In baseband transmission, a digital signal is transmitted over a channel without converting the digital signal to an analog signal. Baseband transmission requires a channel with a bandwidth that starts from zero called low-pass channel.

In broadband transmission, a digital signal is transmitted over a channel after converting it to an analog signal. Broadband transmission allows to use a channel with a bandwidth that does not start from zero called bandpass channel.

Now the following techniques are used to convert digital data to digital signals.

3.4.1 LINE CODING

Line coding is the process of converting digital data to digital signals. Line coding converts a sequence of bits stored in computer memory to a digital signal. At the sender end, digital data are encoded into a digital signal and at the receiver end, the digital signal is decoded into its digital data.

A data element is the smallest unit to represent a piece of information. In digital data communications, a signal element carries data elements. A signal element is the shortest unit with respect to time of a digital signal. So we can say data elements are being carried and signal elements are the carriers.

The number of signal elements sent in 1 second is called the signal rate. The unit is called baud. The signal rate can also be referred as the pulse rate or the modulation rate or the baud rate.
So one main objective in data communication is to increase the data rate which increases the speed of transmission and to decrease the signal rates which decrease the bandwidth requirement.
Now there are some problems in decoding a digital signal discussed as follows:

1. In decoding a digital signal, the signal power is evaluated against the running average of the received signal power called baseline to determine the value of the data element. Now in case of a long string of 0s and 1s can cause a drift in the baseline called baseline wandering. Due to this baseline wandering, it difficult for the receiver to decode the digital signal correctly. A good line coding scheme is required to prevent baseline wandering.

2. When the voltage level in a digital signal is constant for a while, the spectrum creates very low frequencies. These frequencies around zero are called direct current (DC) components. Now these DC components create problems for a system that cannot pass low frequencies or a system that uses electrical coupling. For these systems, a scheme without DC components is required.

3. In decoding a digital signal correctly, the receiver's bit intervals must be same with the sender's bit intervals. If the receiver clock is faster or slower than the sender then the bit intervals will not be same for them. As a result of this, the receiver might misinterpret the signals. To solve this problem, a digital signal includes timing information in the data transmitted through a communication medium.

4. The errors occurred during the signal transmission should be detected at the time of decoding digital signals. So a built-in error-detecting capability in the generated code is required to detect some of or all the errors.

5. Noise and other interferences are also should be minimized from the digital signals.

Line coding Schemes:
In general line coding schemes are divided into some categories which are discussed as follows.

Unipolar Scheme: In a unipolar scheme, all the signal levels are on one side of the time axis. In general, a unipolar scheme was designed as a non-return-to zero (NRZ) scheme. Here 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. This scheme is not used in data communications as it is very costly.

Polar scheme:
In case of polar schemes, the signal levels are on the both sides of the time axis.
In polar non-return-to zero (NRZ) encoding, two levels of voltage amplitude are used. The polar NRZ is divided into two versions which are NRZ-Level and NRZ-Invert. In NRZ-Level, the level of the voltage determines the value of the bit. On the other hand in NRZ-Invert, if there is a change in the level of the voltage then the bit is 1 and if there is no change, the bit is 0. Both NRZ-Level and NRZ-Invert have an average signal rate of N/2 Bd and have a DC component problem.

In case of NRZ encoding when the sender and receiver clocks are not synchronized then the receiver does not know when one bit has ended and the next bit is starting. So due to this problem, the return-to-zero (RZ) scheme can be used. This scheme uses three values which are positive, negative and zero. In RZ scheme, the signal changes during the bit.
The disadvantages of RZ encoding are given as follows:

1. It occupies greater bandwidth as it requires two signal changes to encode a bit.
2. It uses three levels of voltage that is more complex to create and discern.

Because of these disadvantages RZ encoding is not used in recent times.

Manchester and Differential Manchester scheme: The Manchester scheme is the combination of RZ and NRZ-Level. In Manchester encoding, the duration of the bit is divided into two halves. During the first half, the voltage remains at one level and moves to the other level in the second half. Here synchronization is maintained by the transition at the middle of the bit.
In case of Differential Manchester scheme, the idea of RZ and NRZ-Invert are combined. Here the bit values are determined at the beginning of the bit. If the next bit is 0, there is a transition otherwise if the next bit is 1 then no transition occurs.
The Manchester scheme solves several problems which are found in NRZ-Level and on the other hand differential Manchester scheme solves the problems that are associated with NRZ-Invert. In these schemes, no baseline wandering is found. Here each bit has a positive and negative voltage contribution, so no DC component is also found.
Now the disadvantage of Manchester and differential Manchester is that here the signal rate is double that for NRZ because there is always one transition at the middle of the bit and maybe one transition at the end of each bit. Manchester and Differential Manchester are also referred as biphase schemes.

Bipolar Schemes: In bipolar scheme, there are three voltage levels which are positive, negative and zero. Here the voltage level for one data element is at zero and the voltage level for the other element changes between positive and negative. The advantage of the bipolar scheme is that it has the same signal rate as NRZ with no DC component. The concentration of the energy in bipolar encoding is around frequency N/2.

Multilevel Schemes: The multilevel schemes are designed to increase the number of bits per baud by encoding a pattern of m data elements into a pattern of n signal elements. A group of m data elements can produce a combination of 2^m data patterns as a data can be either 1 or 0. For L different levels and n signal elements, L^n combinations of signal patterns are produced. Now if $2^m = L^n$ then each data pattern is encoded into one signal pattern and if $2^m<L^n$, data patterns occupy only a subset of signal patterns. In multilevel schemes this subset of signal patterns can be designed so that it can prevent baseline wandering and detect data transmission errors .Another advantage of this designing is that it provides synchronization. On the other hand if $2^m > L^n$ then data encoding is not possible as some of the data patterns cannot be encoded.

Two binary, one quaternary (2B1Q) is a multilevel scheme with data patterns of size 2 and it encodes the 2-bit patterns as one signal element belonging to a four-level signal. The average signal rate of 2B1Q is N/4.

Eight binary, six ternary (8B6T) is a multilevel scheme with three signal levels and it encode a pattern of 8 bits as a pattern of 6 signal elements. The average signal rate of this scheme is $\frac{1}{2}$ X N X $\frac{6}{8}$.

Four dimensional five-level pulse amplitude modulation(4D-PAM5) is a multilevel encoding scheme with five voltage levels which are -2,-1,0,1 and 2.Here data is sent over four wires at the same time. So the signal rate in this scheme can be reduced to N/8.

Multiline Transmission: The Multiline Transmission, three level scheme referred as MLT-3 is a differential encoding scheme with three levels +V, 0, and –V and three transition rules to move between the levels . The three transition rules are given as follows:

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 then the next level is 0.

3. If the next bit is 1 and the current level is 0 then the next level is the opposite of the last nonzero level.

3.4.2. BLOCK CODING

The block coding changes a block of m bits into a block of n bits. Here n is larger than m. Block coding is also referred to as an mB / nB encoding technique. Block coding is divided into three steps as follows:

1. In the first step, a sequence of bits is divided into groups of m bits.
2. In the second step, an m-bit group is substituted for an n bit group.
3. In the last step, the n-bit groups are combined together to form a stream. The new stream has more bits than the original bits.

The four binary/five binary (4B/5B) coding scheme is a block coding scheme, can be used in combination with NRZ-Invert. In case of the 4B/5B scheme, the 5-bit output replaces the 4-bit input with no more than one leading zero and no more than two trailing zeros. As a result of this, when different groups are combined to make a new sequence then more than three consecutive 0s will not occur.

The eight binary/ten binary (8B/10B) encoding is also a block coding scheme where a group of 8 bits of data is substituted by a 10-bit code. The advantage of this scheme is that here the error detection capability is more than 4B/5B. The 8B/10B block coding is actually a combination of 5B/6B and 3B/4B encoding.

3.4.3 ANALOG-TO-DIGITAL CONVERSION

For the conversion of analog signal to digital data, two techniques are available which are pulse code modulation and delta modulation.

Pulse Code Modulation (PCM):

A PCM encoder has three processes as follows

1. In this first process of PCM the analog signal is sampled at every N second, where N is the sample interval or period. The inverse of the sampling interval is called the sampling rate or sampling frequency. The sampling process is also referred to as pulse amplitude modulation .There is three sampling methods which are ideal, natural and flat-top.

   In case of ideal sampling, pulses from the analog signal are sampled. But this method is difficult to implement.

In case of natural sampling, a high-speed switch is turned on for only the small period of time when the sampling occurs.

In case of flat-top sampling, a circuit is used to create flat-top samples.

2. In the second process, the sampled signal is quantized because the result of sampling is a series of pulses with amplitude values between the maximum and minimum amplitude of the signal which can be infinite with nonintegral values between the two limits and so cannot be used in the encoding process. Now the steps in quantization are as follows:
   A. In the first step, the maximum and minimum amplitude of the original analog signal is estimated. Now let these amplitudes are V_{min} and V_{max}.
   B. In the second step, the range of the amplitudes is divided into L zones with height Δ (delta) for each zone.

   $$\Delta = \frac{V_{max} - V_{min}}{L}$$

   C. Now the quantized values are assigned from 0 to L-1 to the midpoint of each zone.
   D. In the last step, the value of the sample amplitude is approximated to the quantized values.

   Now in this process, the choice of L and the number of quantization levels are depended on the range of the amplitudes of the analog signal and the amount of accuracy required to recover the signal. If lower values of L are chosen then it will increase the quantization error if there is a lot of fluctuation in the signal. The input values to the quantization process are the real values and the output values are the approximated values.
3. In the last step, the quantized values are encoded as streams of bits that mean each sample can be changed to an n-bit code word.

Delta Modulation (DM)

As PCM technique is a very complex technique and so another technique called delta modulation (DM) can be used to convert analog signal to digital data. In DM, the difference between successive samples is encoded into n-bit data streams. Here the analog signal is approximated with a series of segments and each segment is compared to the original analog wave to determine the increase or decrease in relative amplitude. If there is no change of the signal amplitude from the previous sample then the modulated signal will remain at the same 0 or 1 state of the previous sample.

3.4.4 TRANSMISSION MODE

The transmission mode of binary data across a communication medium can be either serial or parallel mode.

Serial Transmission:

In case of serial transmission one bit of binary data is transmitted at a time, so it requires only one communication channel.

The advantage of serial over parallel transmission is that it is less costly because it requires only one communication channel. The disadvantage in this mode is that the data transfer speed is less than the speed of parallel transmission mode.

Now three ways of serial transmission are discussed as follows:

1. **Asynchronous Transmission:** In Asynchronous transmission, binary data are received and translated by some given patterns. Now these patterns are based on grouping the bit stream into bytes. In asynchronous transmission, the transmitted data is encoded with start and stop bits. Here one start bit that is 0 is sent at the beginning and one or more stop bits that are 1s are sent at the end of each byte. There may be a gap between each byte.

2. **Synchronous transmission:** In synchronous transmission, the binary data stream is grouped into some frames which may contain multiple bytes. Here the binary data for each frame are transmitted as an unbroken string of 1s and 0s.Now it is the responsibility of the receiver to separate the string into the bytes or characters and reconstruct the information. There is no gap between bits in synchronous serial transmission but there may be uneven gaps between different frames.

3. **Isochronous:** In case of isochronous transmission, the arrival of binary data is maintained at a fixed rate. This type of transmission is used in case of real-time audio and video transmission where no uneven delays between frames should be available.

Parallel Transmission:

In case of parallel transmission of binary data, n bits of data are sent at a time. So, in this mode of transmission the binary data are organized into groups of n bits each. The mechanism for parallel transmission uses n wires to send n bits at one time so that each bit has its own wire and all n bits of one group can be transmitted with each clock tick from source to the destination device.

The advantage of parallel transmission over serial transmission is its data transfer speed is more. But the disadvantage of this mode of transmission is its cost. Here cost is increased because it requires n communication lines to transmit the binary data stream.

3.5 ANALOG TRANSMISSION

In analog transmission, information are converted to analog signals. Now digital-to-analog conversion is a process to convert digital data to a bandpass analog signal and analog-to-analog conversion is a process to convert a low-pass analog signal to a bandpass analog signal.

3.5.1 MODULATION DIGITAL DATA

Digital-to-analog conversion is implemented by changing any one of the three characteristics of an analog signal based on the information in digital data. These three characteristics are amplitude, frequency, and phase. There are four mechanisms for modulating digital data into an analog signal which are amplitude shift keying (ASK), frequency shift keying (FSK), phase shift keying (PSK) and quadrature amplitude modulation (QAM).

Carrier Signal: Carrier signal is a high frequency signal that is produced by the sending device **i**n analog transmission. In analog transmission, the receiving device is tuned to the frequency of the carrier signal so that the digital information changes the carrier signal by modifying one or more of its characteristics which are amplitude, frequency, or phase.

Amplitude Shift Keying (ASK):

In case of amplitude shift keying, the amplitude of the carrier signal is varied to create signal elements. Here both frequency and phase remain constant. The two types of ASK are discussed as follows:

Binary Amplitude Shift Keying (BASK): In general, ASK is implemented using only two levels of signal elements and it is called binary amplitude shift keying or on-off keying. Here the peak amplitude of one signal level is 0 and the other is the same as the amplitude of the carrier signal.

Multilevel ASK: When there are more than two levels of signal elements available then multilevel ASK can be implemented.

Frequency Shift Keying:

In case of frequency shift keying, the frequency of the carrier signal is varied to represent signal elements. Here the frequency of the modulated signal is constant for the duration of one signal element and it changes for the next signal element if the data element changes. On the other hand both peak amplitude and phase remain constant for all signal elements.

In binary frequency shift keying (BFSK), two carrier frequencies are considered and more than two carrier frequencies are used in multilevel frequency shift keying (MFSK).

Phase Shifting Keying:

In case of phase shift keying, the phase of the carrier signal is varied to represent two or more different signal elements. Here both peak amplitude and frequency remain constant.

In binary phase shift keying (BPSK), two signal elements are used. The phase of one signal element is 0^0 and the other signal element's phase is 180^0. The advantage of binary PSK over binary ASK is that it is less susceptible to noise. Another advantage of PSK over FSK is that it does not require two carrier signals.

In quadrature phase shift keying (QPSK), two separate BPSK modulations are used to increase the baud rate.

Quadrature Amplitude Modulation (QAM):

In case of quadrature amplitude modulation, two carriers are used with different amplitude levels for each carrier. The 4-QAM scheme is a QAM. One type of 4-QAM scheme uses a unipolar NRZ signal to modulate each carrier. Here the mechanism used

is same with ASK (OOK). Another type of 4-QAM scheme uses polar NRZ and it is same with QPSK. There is an another QAM-4 which uses a signal with two positive levels to modulate each of the two carriers.

3.5.2 MODULATION ANALOG SIGNALS

Analog-to-analog conversion can be implemented in three ways which are amplitude modulation (AM), frequency modulation (FM), and phase modulation (PM).

Amplitude Modulation:

In Amplitude Modulation, the amplitude of the carrier signal varies with the changing amplitudes of the modulating signal. The frequency and phase of the carrier remain constant.

Frequency Modulation:

In Frequency Modulation, the frequency of the carrier signal is modulated so that the frequency of the carrier signal changes with the change in amplitude of the information signal. The peak amplitude and phase of the carrier signal remain constant.

Phase Modulation:

In Phase Modulation, the peak amplitude and frequency of the carrier signal remain constant, but the phase of the carrier signal changes when the amplitude of the information signal changes. In FM, the instantaneous change in the carrier frequency is proportional to the amplitude of the modulating signal. On the other hand, in PM the instantaneous change in the carrier frequency is proportional to the derivative of the amplitude of the modulating signal.

3.5.3 TELEPHONE MODEM

A **modem** is a device also called as modulator-demodulator used to modulate an analog carrier signal to encode digital information and to demodulate a carrier signal to decode the transmitted information. So modems can be used to transmit analog signals. An example of modem is a voice band modem that converts the digital data of a personal computer into modulated electrical signals in the voice frequency range of a telephone channel. Now these signals can be transmitted over telephone lines and demodulated by another modem at the receiver side to recover the digital data.

3.6 MULTIPLEXING

Multiplexing is the set of techniques used for the simultaneous transmission of multiple signals across a single data link. So in a multiplexed system, n lines share the bandwidth of one link. The three types of multiplexing techniques are discussed as follows:

3.6.1 FREQUENCY-DIVISION MULTIPLEXING

When the bandwidth of a data link is greater than the combined bandwidths of the signals to be transmitted then the frequency-division multiplexing can be used to develop a multiplexed system. In FDM, signals generated by each sending device modulate different carrier signals and these modulated signals are then combined into a single composite signal. Now the carrier signals are separated by sufficient bandwidth to accommodate the modulated signal. Channels can be separated by strips of unused bandwidth called guard bands which prevent signals from overlapping. In general FDM is considered as analog multiplexing technique but it can be used to combine sources sending digital signals also.

The demultiplexer uses a series of filters to decompose the multiplexed signal into its component signals and each signal is passed to a demodulator which separates them from their carriers and passes them to the output lines.

The telephone companies have multiplexed signals from lower-bandwidth lines onto higher-bandwidth lines to maximize their efficiency. In this case, many switched or leased lines can be combined into fewer but bigger channels and FDM is used for analog lines. Here 12 voice channels are multiplexed onto a higher-bandwidth line to create a group with a bandwidth of 48 kHz. Now, at most five groups can be multiplexed to create a composite signal called a super group with a bandwidth of 240 kHz and it supports up to 60 voice channels. Again 10 super groups are multiplexed to create a master group with a bandwidth of 2.40MHz of bandwidth and it support up to 600 voice channels. Here the guard bands between the super groups are required and so it increases the necessary bandwidth to 2.52 MHz. Now, six master groups can be combined into a jumbo group with a bandwidth of 15.12 MHz. Here also the guard bands

between the master groups are required and so it increases the necessary bandwidth to 16.984 MHz.

The FDM is used in AM and FM radio broadcasting and television broadcasting.

3.6.2 WAVELENGTH-DIVISION MULTIPLEXING

Wavelength-Division Multiplexing (WDM) is same with FDM but here the multiplexing and demultiplexing involve optical signals transmitted through fiber-optic channels. The data rate of optical fiber is higher than the data rate of metallic transmission cable. So if a fiber-optic cable is used for one single line then the available bandwidth will be wasted. So, multiplexing can be used to combine several lines into one. In WDM, multiple light sources are combined into one single light at the multiplexer and do the reverse at the demultiplexer. The combining and splitting of light sources are easily handled by a prism.

One application of WDM is the SONET network in which multiple optical fiber lines are multiplexed and demultiplexed.

Dense WDM (DWDM) is a WDM technique in which a very large number of channels are multiplexed by spacing channels very close to one another. It provides greater efficiency.

3.6.3 TIME DIVISION MULTIPLEXING

Time division multiplexing (TDM) is a digital multiplexing technique where several connections are allowed to share the high bandwidth of a link with each connection occupies a portion of time in the link. TDM is divided into two different schemes which are synchronous and statistical.

In case of synchronous TDM, the data flow of each input connection is divided into some units where each input unit has one input time slot. Here an unit can be 1 bit, one character or one block of data. Each input unit becomes one output unit with each output unit has one output time slot. Now the duration of an output time slot is n times shorter than the duration of an input time slot where n is the number of connections. Here, in this multiplexing system, a frame is formed with a collection of data units collected from each input connection in one round of input. Now a frame is divided into n time slots where n is the number of connection and one slot is allocated for each unit. So a frame consists of one complete cycle of time slots where one slot is dedicated to each sending device.

In synchronous TDM, the data rate of the link is n times faster, and the unit duration is n times shorter where n is the number of connections.

In statistical time-division multiplexing, time slots are dynamically allocated to improve bandwidth efficiency. So in this system, the number of slots in each frame is less than the number of input lines. Here the multiplexer checks each input line and it allocates a slot for an input line if the line has data to send but if the input line does not have any data to send then it skips the line and checks the next line.

In statistical TDM, a slot needs to carry data and the address of the destination and so, the ratio of the data size to address size must be in some limit to make transmission efficient. The frames in statistical TDM need not be synchronized and so it does not require synchronization bits.

In case of TDM, if input data rates are not the same then problem occurs and in this situation, three strategies or a combination of them can be used which are multilevel multiplexing, multiple-slot allocation and pulse stuffing.

Multilevel Multiplexing is a technique used when the data rate of an input line is a multiple of others.

Multiple-Slot Allocation is a technique where more than one slot are assigned in a frame to a single input line.

Pulse Stuffing: When the bit rates of sources are not multiple integers of each other then neither of the above two techniques can be used. So in this situation, pulse stuffing is used where the highest input data rate is the dominant data rate and dummy bits are added to the input lines with lower rates.

In case of TDM, synchronization between the multiplexer and demultiplexer must be maintained. If the multiplexer and the demultiplexer are not synchronized then a bit may be received by the wrong channel. So to synchronize the multiplexer and the demultiplexer, one or more synchronization bits are usually added to the beginning of each frame which are called framing bits. These framing bits follow a pattern from frame to frame which allows the demultiplexer to synchronize with the incoming data stream.

3.7 TRANSMISSION MEDIA

A transmission medium can be defined as any kind of material substance like solid, liquid, gas, or plasma that can carry any signal or information from a source to a destination. In case of electromagnetic waves like light and radio waves, vacuum is used as a transmission media. The Different types of transmission media are discussed as follows:

3.7.1 GUIDED MEDIA

In case of guided media, data transmission is maintained along a physical path from one device to another. Examples of guided media are twisted-pair cable, coaxial cable and fiber optic cable. The capacity of a guided transmission media in terms of data rate or bandwidth is depended upon the distance from the source to the receiver.

Magnetic Media:

Magnetic tapes or any removable media like recordable CDs and DVDs can be used to transport data from one computer to another. Here the data are written onto the magnetic media and physically transported it to the destination machine where the data are read back in again. The advantage of magnetic media is that high bandwidth can be achieved.

Twisted-Pair cable:

A twisted-pair cable consists of two insulated copper wires twisted together. One of the wires is used to carry signals to the receiver and the other one is used only as a ground reference. In general, a number of these pairs are bundled together into a cable by wrapping them in a protective sheath. Twisted-pair cable accepts and transport signals in the form of electric current.

In twisted-pair cable, interference and crosstalk may affect both wires and create unwanted signals. If the two wires are parallel, the effect of these unwanted signals is not the same in both wires because they are at different locations relative to the noise or crosstalk sources. So by twisting the pairs, the effect of the unwanted signals on both wires can be made same and as a result of this most of the unwanted signals are removed.

There are two types of twisted- pair cable used in data communication referred as unshielded twisted-pair (UTP) and shielded twisted-pair (STP).

Unshielded twisted pair (UTP) is ordinary telephone wire. This is the least expensive of all the transmission media used in local area networks. The disadvantage of UTP is that the pair of wires may be affected more from the external electromagnetic interference, interference from nearby twisted pair and environmental noise. The most common UTP connector is RJ45 (RJ stands for registered jack).

In STP cable, the twisted pair is shielded with a metallic braid that reduces interference. But it is bulkier and more expensive.

The Unshielded twisted pair cables are divided into seven categories given as follows:

- Category 1: It is used in telephone network. Its data rate is less than 0.1 Mbps.
- Category 2: It is used in T-lines. Its data rate is 2 Mbps.
- Category 3: It is used in LANs. Its data rate is 10 Mbps.
- Category 4: It is used in Token Ring networks. Its data rate is 20 Mbps.
- Category 5: It is used in LANs. Its data rate is 100 Mbps.
- Category 5E: It is an extension of category 5. It is also used in LANs. Its data rate is 125 Mbps.
- Category 6: It is used in LANs. Its data rate is 200 Mbps.
- Category 7: It is also called shielded screen twisted pair. It is also used in LANs. Its data rate is 600 Mbps.

Coaxial Cable:

Coaxial cable consists of a hollow outer cylindrical conductor which surrounds a single inner wire conductor. The inner conductor is the central core conductor of solid or stranded wire which is usually a solid copper surrounded by an insulating layer and all enclosed by a shield. The outer conductor is covered with a jacket or shield. This outer metallic shield protects the inner conductor from noise. The whole coaxial cable is protected by a plastic cover. A single coaxial cable has a diameter of from 1 to 2.5 cm. Coaxial cables carries signals of higher frequency ranges than twisted pair cable. The attenuation is much higher in coaxial cables than in twisted-pair cable. So in coaxial cables, the signal weakens rapidly and requires the frequent use of repeaters.

Coaxial cables are categorized by their radio government (RG) ratings. So the different categories are RG-11, RG-58 and RG-59.

Coaxial cable was used in analog telephone networks where a single coaxial network can supports 10,000 voice signals and it was also used in digital telephone networks where a single coaxial cable could carry digital data up to 600 Mbps. But today, fiber-optic cable replaces coaxial cable in most parts of the telephone networks.

Coaxial cables were used in the traditional cable TV networks. But now, cable TV providers replaced most of the media with fiber-optic cables. Coaxial cables are used only at the network boundaries in hybrid networks. RG-59 coaxial cable is used in cable TV.

Coaxial cables are also used in the traditional Ethernet LANs. RG-58 coaxial cable with Bayone-Neill-Concelman(BNC) connector is used in 10Base-2 or Thin Ethernet to transmit data at 10 Mbps with a range of 185 m. RG-11 coaxial cable is used in the 10Base5 or Thick Ethernet to transmit 10Mbps with a range of 5000 m.

Fiber-Optic Cable:

A fiber-optic cable is constructed using glass and plastic to transmit signals in the form of light. The center of the cable is the glass core through which the light propagates. In multimode fibers, the core is typically 50 microns in diameter and in single-mode fibers, the core is 8 to 10 microns. A glass cladding with a lower index of refraction than the

core surrounds the core and a thin plastic jacket is used to protect the cladding. In general fibers are grouped in bundles which are protected by an outer sheath. The practical bandwidth of this media is about 10 Gbps but its achievable bandwidth is more than 50,000 Gbps. In the laboratory, 100 Gbps has been achieved on a single fiber-optic cable.

Now the light source, the transmission medium and the detector are the most important components in the fiber-optic transmission system. Here a pulse of light means a 1 bit and the absence of light means a 0 bit. The light sources in fiber-optic cables are Light Emitting Diodes (LEDs) and Light Amplification by Stimulated Emission Radiation (Lasers). The transmission medium is an ultra-thin fiber of glass. The detector generates an electrical pulse when light falls on it. So a unidirectional data transmission system is developed by attaching a light source to one end of an optical fiber and a detector to the other end. Now this transmission system accepts an electrical signal and converts it to the light pulses to transmit it through the medium. At the receiving end, the light pulses are reconverted into an electrical signal.

Now in this optical transmission system, light can propagate through the medium for many kilometres virtually without any loss because of one principle of light. This principle is when a light ray passes from one medium to an another medium then the light ray is refracted at the boundary of the two mediums. Now the amount of refraction depends on the properties of the two media. According to the principle of light, if the angles of incidence are above a certain critical value then the light is refracted back into the first media and none of it escapes into the second media. So in case of fiber-optic cable, a light ray incident at or above the critical angle and so it is trapped inside the fiber.

Propagation Modes: There are two types of modes of propagation light along optical channels are available in the current technology. These are multimode and single mode propagation.

In case of multimode, multiple beams of light from a light source propagate through the core in different paths. Multimode can be implemented in two forms which are step-index and graded-index.

In multimode step-index fiber, the density of the core remains constant from the center to the edges .Here light moves through this constant density in a straight line to the

interface of the core and the cladding. At the interface, the angle of the light beam's motion is changed due to the lower density and as a result of this the distortion of the signal occurs as it passes through the fiber.

In case of multimode graded-index fiber, the density at the center of the core is highest and it decreases gradually to the edge. So the density at the edge is lowest. Because of these varying densities, the distortion of the signal through the cable is decreased.

In case of single mode, step-index fiber and a source of light that limits beams to a small range of angles are used. The diameter of single mode fiber is smaller than that of multimode fiber and so it has lower density. As a result of this lower density, the critical angle becomes close enough to 90^0 to make the propagation of beams almost horizontal and so here propagation of different beams is almost identical. In this case, all the beams arrive at the destination at the same time and so distortion to the signal is very less.

Advantages of fiber optic cables:

1. The bandwidth of fiber optic cable is much higher than either twisted-pair or coaxial cable.
2. Fiber optic transmission distance is much greater than that of other guided media. A signal can run for 50 km without requiring regeneration. On the other hand, repeaters are required at every 5 km for coaxial or twisted pair cable.
3. Fiber-optic cables are not affected by electromagnetic noise or power failures.
4. Glass is more resistant to corrosive materials than copper. So fiber-optic cables are not affected by corrosive chemicals in the air.
5. Fiber optic cables are much lighter than copper cables. One thousand twisted pairs copper cables of 1 km long weight 8000 kg but two fibers have more capacity and weight only 100 kg.
6. Fiber optic cables do not leak light and so difficult to tap. On the other hand, copper cables create antenna effects that can easily be tapped.

Disadvantages of fiber optic cables:

1. Fiber-optic cable is a relatively new technology. So all engineer do not have the skills required for its installation and maintenance.
2. Since propagation of light is unidirectional so for two-way communication, two fibers are required.
3. The fiber-optic cable and the interfaces are more expensive than those of other guided media.

Optical fibers are categorized by the ratio of the diameter of their core to the diameter of their cladding and both these diameters are measured in micrometers. Different types of fiber optic cables are given as follows:

1. 50/125: It is a fiber optic cable with the diameter of the core is 50.0 µm and the diameter of the cladding is 125 µm. Here the propagation mode is multimode.
2. 62.5/125: It is a fiber optic cable with the diameter of the core is 62.5 µm and the diameter of the cladding is 125 µm. Here the propagation mode is multimode.
3. 100/125: It is a fiber optic cable with the diameter of the core is 100.0 µm and the diameter of the cladding is 125 µm. Here the propagation mode is multimode.
4. 7/125: It is a fiber optic cable with the diameter of the core is 7.0 µm and the diameter of the cladding is 125 µm. Here the propagation mode is single mode.

There are three types of connectors available for fiber-optic cables.

1. The subscriber channel (SC) connector: It is used for cable TV. It uses a push/pull locking system.
2. The straight-tip (ST) connector: It is used for connecting cable to networking devices. It uses a bayonet locking system. It is more reliable than SC.
3. MT-RJ is a connector used in duplex multimode connections.

Uses of fiber-optic cables:

1. Some cable TV companies creates hybrid network by combining optical fiber and coaxial cable.
2. Local-area networks such as 100Base-FX network (Fast Ethernet) and 1000Base-X use fiber-optic cable.

3.7.2 UNGUIDED MEDIA

Wireless communication is implemented using unguided media. Unguided media does not require any physical conductor to transport electromagnetic waves. In this type of communication, signals are normally broadcast through free space and so anyone can receive these signals with a device capable of it. Unguided signals can move from the source to destination in three ways which are ground propagation, sky propagation and line-of-sight propagation.

In case of ground propagation, unguided signals move through the lowest portion of the atmosphere. For example low-frequency radio waves move in all directions from the transmitting antenna. The amount of distance covered by these signals depend on the amount of power in the signal.

In case of sky propagation, higher-frequency radio waves radiate upward into the ionosphere and then they are reflected back to earth. In this case, the unguided signals cover greater distances with lower output power.

In case of line-of-sight propagation, very high-frequency signals travel in straight lines directly from antenna to antenna. An antenna can be defined as an electrical conductor or group of conductors used either for radiating electromagnetic waves or for receiving electromagnetic waves. In line-of-sight propagation, antennas must be directional and either tall enough or close enough together so that it is not affected by the curvature of the earth.

Now wireless transmission is divided into three groups which are radio waves, micro waves and infrared waves.

Radio Waves:

The electromagnetic waves having frequencies between 3 kHz and 1GHz are called radio waves. Radio waves travel in all directions from the source and so the transmitter and receiver do not required to have any specific physical alignment. Radio waves are easy to generate, can travel long distances and can penetrate buildings easily. The low frequency radio waves can easily pass through obstacles but the power falls off sharply as the distance from the source increases. The high frequency radio waves travels in straight lines and bounce off obstacles. They are also absorbed by rain. All radio waves are affected by interference from motors and other electrical equipment. Now different bands of radio waves are discussed as follows:

- Very Low Frequency (VLF): The frequency range of this band is from 3 kHz to 30 kHz. It follows ground propagation. It is used in long range radio navigation.
- Low Frequency (LF): The frequency range of this band is from 30 kHz to 300kHz.It follows ground propagation. It is used in radio beacons and navigational locators.
- Middle Frequency (MF): The frequency range of this band is from 300 kHz to 3MHz.It follows sky propagation. AM radio broadcasting uses the MF band.
- High Frequency (HF): The frequency range of this band is from 3 MHz to 30 MHz .It follows sky propagation. It is used in ship and aircraft communication.
- Very High Frequency (VHF): The frequency range of this band is from 30 MHz to 300 MHz. It follows sky and line-of-sight propagation. It is used in VHF TV and FM radio.

Microwaves:

Electromagnetic waves with frequencies between 1 and 300 GHz are called microwaves. Microwaves are unidirectional. To transmit microwaves, the sending and receiving antennas required to be aligned. Here a pair of antennas can be aligned without interfering with another pair of aligned antennas. Microwaves follow line-of-sight propagation. Since the microwaves travel in a straight line, if the towers with the mounted antennas are too far apart then these towers required to be very tall and repeaters are needed periodically. For 100-meter-high towers, repeaters can be spaced 80 km apart. Microwaves do not pass through walls.

Microwave communication is so widely used for long-distance telephone communication, mobile phones, television distribution etc.

Different bands of microwaves are given as follows:

- Ultrahigh Frequency (UHF): The frequency range of this band is from 300 MHz to 3GHz. It is used in UHF TV, cellular phones, satellite etc.
- Superhigh Frequency (SHF): The frequency range of this band is from 3 GHz to 30 GHz. It is used in satellite communication.
- Extremely High Frequency (EHF): The frequency range of this band is from 30 GHz to 300 GHz. It is used in radar and satellite.

Infrared:

Unguided infrared and millimeter waves are used for short-range communication. For example these waves are used in the television remote controls. The frequency range of these waves is from 300GHz to 400THz. The advantage of these waves is these are relatively directional, cheap, and easy to build. But the disadvantage of these waves is that they do not pass through solid objects. Therefore, no government license is needed to operate an infrared system. Infrared communication can be used for connecting notebook computers and printers.

Lightwave Transmission

Unguided optical signals are also used in data transmission. Lasers can be used to connect the LANs in two buildings. In this application, lasers are mounted on the rooftops of the two buildings. Now the optical signalling using lasers is unidirectional and so each building needs its own laser and its own photo detector. The advantage of this type of transmission is that very high bandwidth can be achieved at very low cost. It is also relatively easy to install and does not require any license to use it.

The disadvantage of this type of data transmission is that laser beams cannot penetrate rain or thick fog and it is affected by the heat from the sun.

3.8 CIRCUIT SWITCHING

A circuit-switched network consists of a set of switches connected by physical links in which each link is divided into n channels. A connection between two stations is a dedicated path made of one or more links. Each link is normally divided into n channels by using FDM or TDM.

The communication in a circuit-switched network is achieved in three phases which are connection setup, data transfer and connection teardown.

In the connection setup phase, a dedicated circuit between the two parties trying to communicate is established before beginning the communication. So connection setup means creating dedicated channels between the switches.

In data transfer phase, data are transferred from source to destination after the establishment of the dedicated circuit.

In teardown phase, a signal is sent to each switch to release the resources when one of the parties required to disconnect.

Circuit- switched networks are not very efficient because here may be the resources are unavailable to other connections for a long time. But the advantage of a circuit-switched network is that the delay in this type of network is minimal. During data transfer the data are not delayed at each switch as the resources are allocated for the duration of the connection. Here the total delay is due to the time required to establish the connection, transfer data and disconnect the circuit. Now the delay caused by the setup is the sum of four parts which are the propagation time of the source computer request, the request signal transfer time, the propagation time of the acknowledgment from the destination computer and the signal transfer time of the acknowledgment. The delay due to data transfer is the sum of two parts which are the propagation time and data transfer time.

3.9 TELEPHONE NETWORK

The telephone network was designed in the late 1800. The plain old telephone system (POTS) was an analog system which used analog signals to transmit voice. Telephone networks use circuit switching. In recent times, the telephone network has technically changed in many areas. Now it is digital as well as analog.

The telephone network consists of three main components which are local loops, trunks and switching offices.

Local Loops:

The local loop is a twisted-pair cable which connects the subscriber telephone to the nearest end office or local central office. The bandwidth of the local loop used for voice is 4000 Hz. The first three digits of a local telephone number define the office and the next four digits provide the local loop number.

Trunks:

Trunks are the transmission media like optical fibers or satellite that handle the communication between offices. A trunk handles hundreds or thousands of connections through multiplexing.

Switching Offices:

The telephone network has several levels of switching offices like end offices, tandem offices and regional offices. The telephone company has switches located in a switching office in which a switch connects several local loops or trunks that allows a connection between different subscribers.

Telephone companies provide two types of services which are analog and digital.

Analog Services:

There are two types of analog services discussed as follows:

1. Analog switched service is the first category of the analog services. It is a dial-up service. The signal on a local loop is analog and the bandwidth is usually between 0 and 4000Hz. In general, a local call service is provided for a flat monthly rate.

The 800 service is a service where a subscriber can provide free connections for other subscribers. In this case, the call is free for the caller, but it is paid by the callee. Here the rate is less expensive than that for a normal long-distance call.

The wide-area telephone service (WATS) is a service where outbound calls are paid by the subscribers. This service is a less expensive than regular toll calls. Here charges are based on the number of calls type of the outbound calls. Here three types of outbound calls available which are outbound calls to the same state, outbound calls to several states and outbound calls to the whole country.

The 900 services is a service where the call is paid by the caller and is normally much more expensive than a normal long distance call because the carrier charges two fees. The first is the long distance toll and the second is the fee paid to the callee for each call.

2. Analog leased service the second category of the analog services. Because of this service customers can have a dedicated line also called lease line which is permanently connected to another customer.

Digital Services: In case of telephone network, digital services are less affected by noise and other forms of interference than analog services. The two most common types of digital services are switched/56 service and digital data service (DDS).

Switched/56 Service is a switched digital service that allows data rates of up to 56 kbps and to communicate through this service, both parties must subscribe. The line in a switched/56 service is digital and so subscribers do not need modems to transmit digital data. But in this case subscribers require another device called a digital service unit (DSU).

Digital data Service is the digital version of an analog leased line with a maximum data rate of 64 kbps.

CHECK YOUR PROGRESS

1. Multiple choice questions:

(I) According to Nyquist the theoretical maximum bit rate can be calculated by the following formulae:

A. Bit rate = 2 X B X $\log_2 N$

B. Capacity = B X $\log_2$ (1+SNR)

C. Bit rate = B X $\log_2 N$

D. None of above

(II) Which of the following is the cause of signal impairment?

A. Attenuation

B. Distortion

C. Noise

D. All of the above

(III) Which is not a multiplexing technique?

A. Frequency division multiplexing

B. Wavelength division multiplexing

C. Amplitude division multiplexing

D. Time division multiplexing

(IV) Which is an unguided transmission media?

A. Fiber optic cable

B. Radio wave

C. Microwave

D. Both B and C

(V) Which is not a way of serial transmission?

A. Synchronous transmission

B. Asynchronous transmission

C. Isochronous transmission

D. Symmetric transmission

(VI) Which is not a component of telephone network?

A. Trunk

B. Modem

C. Local loops

D. Switching offices

(VII) Which type of waves are used in the television remote controls?

A. Infrared

B. Lightwave

C. Microwave

D. Both A and C

(VIII) The frequency range of EHF is_____.

A. 30 GHz to 300 GHz

B. 3 GHz to 300 GHz

C. 30 MHz to 300 MHz

D. 3 MHz to 300 MHz

(IX) The electromagnetic waves ranging in frequencies between 3 kHz and 1 GHz are called_____.

A. Microwaves

B. Radio waves

C. Infrared

D. Lightwave

(X) Which is not a category of coaxial cable?

A. RG-10

B. RG-11

C. RG-58

D. RG-59

2. Fill in the blanks:

I. _____ signal is a combination of simple sine waves with different frequencies, amplitudes and phases.

II. _____ time is the amount of time required for a bit to travel from the source to the destination.

III. Line coding is the process of converting _____ to digital signals

IV. The polar NRZ is divided into two versions: _____ and _____.

V. Delta modulation can be used to convert analog signal to _____.

VI. A _____ is a device also called as modulator-demodulator.

VII. _____ is the set of techniques used for the simultaneous transmission of multiple signals across a single data link.

VIII. _____ media does not require any physical conductor to transport electromagnetic waves.

IX. Radio waves have frequencies between _____ and _____.

X. The communication in a circuit-switched network is achieved in three phases: _____, _____ and _____.

3. State whether the following statements are True of False

I. Coaxial cable consists of a hollow outer cylindrical conductor.

II. Fiber-optic cables are affected by electromagnetic noise.

III. The number of signal elements sent in 1 second is called the signal rate.

IV. When a signal changes its original form or shape in the time of movement from source to the final destination then it is called signal attenuation.

V. An analog signal is a continuously varying electromagnetic wave.

VI. Digital signal is a physical signal that has a sequence of voltage pulses that may be transmitted over a transmission medium.

VII. The amount of time required in seconds by a signal to complete 1 cycle is called phase.

VIII. In serial transmission, only one communication channel is required.

IX. In case of isochronous transmission, the arrival of binary data is maintained at a fixed rate.

X. The data rate of shielded screen twisted pair cable is 200Mbps.

3.10 LET US SUM UP

The summery of this unit is given as follows:

- The physical layer provides the transmission of raw bits over any hardware transmission medium.
- Both data and the electromagnetic signal can be either analog or digital in form.
- An analog signal is a continuously varying electromagnetic wave for which the time varying feature of the signal is a representation of some other time varying quantity that may be propagated over a variety of media, depending on spectrum.
- A digital signal is a physical signal that has a sequence of voltage pulses that may be transmitted over a transmission medium.
- Both analog and digital signals can be either periodic or nonperiodic in form.
- A periodic signal completes a pattern within a measurable time frame, called a period, and repeats that pattern over subsequent identical periods.
- A nonperiodic signal changes without exhibiting a pattern or cycle that repeats over time.
- Periodic analog signals can be classified as simple or composite.
- A simple periodic analog signal, a sine wave, cannot be decomposed into simpler signals.
- A composite periodic analog signal is composed of multiple sine waves.
- Three factors on which the data rate depends: bandwidth, the level of the signals,the quality of the channel.
- In case noiseless channel, according to Nyquist the theoretical maximum bit rate can be calculated by the following formulae: Bit rate = 2 X B X $\log_2 N$ Here, B is the bandwidth of the channel, N is the number of signal levels and the bit rate is calculated in bits per second.
- In case of noisy channel, Claude Shannon developed a formula in 1944 to calculate the theoretical highest data rate for a noisy channel. This formula is called the Shannon capacity: Capacity = B X $\log_2$ (1+SNR) Here, B is the bandwidth of the channel, SNR is the signal to noise ratio and capacity is the bit rate of the channel in bits per second.
- Three factors for signal impairment: attenuation, distortion and noise.

- Line coding is the process of converting digital data to digital signals.
- Different line coding schemes are: Unipolar Scheme, Polar scheme, Manchester and Differential Manchester scheme, Bipolar Schemes, Multilevel Schemes, Multiline Transmission:

- The block coding changes a block of m bits into a block of n bits.
- Two techniques for the conversion of analog signal to digital data are: pulse code modulation and delta modulation.
- The transmission mode of binary data across a communication medium can be either serial or parallel mode.
- In case of serial transmission one bit of binary data is transmitted at a time.
- Three ways of serial transmission are: Asynchronous Transmission, Synchronous transmission, Isochronous transmission.
- Multiplexing is the set of techniques used for the simultaneous transmission of multiple signals across a single data link.
- The three types of multiplexing techniques are: Frequency-Division Multiplexing, Wavelength-Division Multiplexing, Time Division Multiplexing.
- A transmission medium can be defined as any kind of material substance like solid, liquid, gas, or plasma that can carry any signal or information from a source to a destination.
- In case of electromagnetic waves like light and radio waves, vacuum is used as a transmission media.
- Two classes of transmission media are: Guided media and Unguided media.
- Examples of different guided media are: Magnetic Media, Twisted-Pair cable, Coaxial Cable, Fiber-Optic Cable.
- A fiber-optic cable is constructed using glass and plastic to transmit signals in the form of light.
- Unguided media does not require any physical conductor to transport electromagnetic waves.
- Unguided signals can move from the source to destination in three ways: ground propagation, sky propagation and line-of-sight propagation.
- Wireless transmission is divided into three groups which are radio waves, micro waves and infrared waves.

- Radio waves travel in all directions from the source and so the transmitter and receiver do not required to have any specific physical alignment.
- Unguided infrared and millimeter waves are used for short-range communication.
- Electromagnetic waves with frequencies between 1 and 300 GHz are called microwaves.
- Lasers mounted on the rooftops of the two buildings, can be used to connect the LANs in two buildings.
- A circuit-switched network consists of a set of switches connected by physical links in which each link is divided into n channels.
- The communication in a circuit-switched network is achieved in three phases: connection setup, data transfer and connection teardown.
- The telephone network consists of three main components: local loops, trunks and switching offices.

3.11 ANSWERS TO CHECK YOUR PROGRESS

1. (I) A , (II) D , (III) C , (IV) D , (V) D ,(VI) B ,(VII) A ,(VIII) A ,(IX) B ,(X) A

2. I. composite ,II. propagation , III. digital data , IV. NRZ-Level, NRZ-Invert , V. digital data , VI. modem , VII. multiplexing , VIII. unguided , IX. 3 kHz and 1 GHz , X. connection setup, data transfer, connection teardown.

3. I. True ,II. False , III. True , IV. False , V. True , VI. True , VII. False , VIII. True , IX. True , X. False

3.12 FURTHER READINGS

- Behrouz A Forouzan : Data Communications and Networking ,TATA McGraw Hill
- William Stallings : Data and Computer Communications, Pearson Education
- Andrew S. Tanenbaum : Computer Networks, Prentice-Hall India

3.13 MODEL QUESTIONS

1. Explain the factors for signal impairment.
2. Explain different transmission modes.
3. Explain the different types of multiplexing techniques.
4. What is telephone modem?
5. Explain fiber optic cables. What are the advantages of fiber optic cables over coaxial cables?

6. What is circuit switching?
7. Explain the services of telephone network.

UNIT - 4: DATA LINK LAYER

UNIT STRUCTURE

4.1 LEARNING OBJECTIVES

After going through this unit, you will be able to:

- learn about error control and flow control
- describe error detection and correction
- describe the error correction and error detection codes
- learn about the working of the Stop-and-Wait protocol
- illustrate the Go-Back-N protocol
- illustrate the Selective Repeat protocol
- learn about the HDLC and Point-To-Point-Protocol
- learn about Ethernet

4.2 INTRODUCTION

In the seven-layer OSI model of computer networking, the data link layer is layer 2. In TCP/IP reference model, it corresponds to, or is part of the link layer. The data link layer is the protocol layer that transfers data between adjacent network nodes in a wide area network or between nodes on the same local area network segment. The data link layer provides the functional and procedural means to transfer data between network entities and might provide the means to detect and possibly correct errors that may occur in the physical layer. The data link layer is concerned with local delivery of frames between devices on the same LAN. Data-link frames do not cross the boundaries of a local network. Inter-network routing and global addressing are higher layer functions, allowing data-link protocols to focus on local delivery, addressing, and media arbitration. The data link thus provides data transfer across the physical link. That transfer can be reliable or unreliable; many data-link protocols do not have acknowledgments of successful frame reception and acceptance, and some data-link protocols might not even have any form of checksum to check for transmission errors. In those cases, higher-level protocols must provide flow control, error checking, and acknowledgments and retransmission.

In this unit we will study the design principles for the data link layer, which deal with algorithms for achieving reliable, efficient communication of whole units of information called frames between two adjacent machines connected by a communication channel that acts conceptually like a wire. The essential property of a channel that makes it ‘‘wire-like’’ is that the bits are delivered in exactly the same order in which they are sent. Communication channels make errors occasionally. Furthermore, they have only a finite data rate, and there is a nonzero propagation delay between the time a bit is sent and the time it is received. These limitations have important implications for the efficiency of the data transfer. The protocols used for communications must take all these factors into consideration. These protocols are the subject of this unit.

4.3 FRAMING

To provide service to the network layer, the data link layer must use the service provided to it by the physical layer. What the physical layer does is accept a raw bit stream and attempt to deliver it to the destination. If the channel is noisy the physical layer will add some redundancy to its signals to reduce the bit error rate to a tolerable level. However, the bit stream received by the data link layer is not guaranteed to be error free. Some bits may have different values and the number of bits received maybe less than, equal to, or more than the number of bits transmitted. It is up to the data link layer to detect and, if necessary, correct errors. The usual approach is for the data link layer to break up the bit stream into discrete frames, compute a short token called a checksum for

each frame, and include the checksum in the frame when it is transmitted. When a frame arrives at the destination, the checksum is recomputed. If the newly computed checksum is different from the one contained in the frame, the data link layer knows that an error has occurred and takes steps to deal with it (e.g., discarding the bad frame and possibly also sending back an error report). Breaking up the bit stream into frames is more difficult than it at first appears. A good design must make it easy for a receiver to find the start of new frames while using little of the channel bandwidth. We will look at four methods:

1. Byte count.
2. Flag bytes with byte stuffing.
3. Flag bits with bit stuffing.
4. Physical layer coding violations.

The first framing method uses a field in the header to specify the number of bytes in the frame. When the data link layer at the destination sees the byte count, it knows how many bytes follow and hence where the end of the frame is. This technique is shown in **Fig. 4.1(a)** for four small example frames of sizes 5, 5, 8, and 8 bytes, respectively. The trouble with this algorithm is that the count can be garbled by a transmission error. For example, if the byte count of 5 in the second frame of **Fig. 4.1(b)** becomes a 7 due to a single bit flip, the destination will get out of synchronization. It will then be unable to locate the correct start of the next frame. Even if the checksum is incorrect so the destination knows that the frame is bad, it still has no way of telling where the next frame starts. Sending a frame back to the source asking for a retransmission does not help either, since the destination does not know how many bytes to skip over to get to the start of the retransmission. For this reason, the byte count method is rarely used by itself.

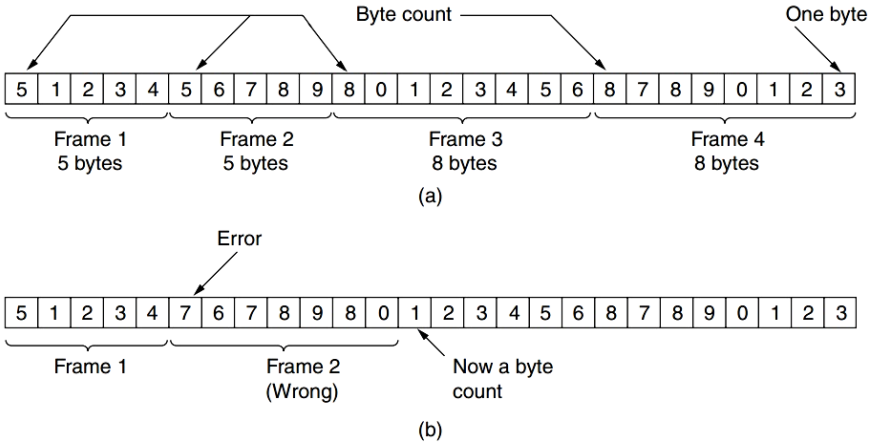


Fig 4.1: A byte stream. (a) Without errors. (b) With one error.

The second framing method gets around the problem of resynchronization after an error by having each frame start and end with special bytes. Often the same byte, called a flag byte, is used as both the starting and ending delimiter. This byte is shown in **Fig. 4.2(a)** as **FLAG**. Two consecutive *flag bytes* indicate the

end of one frame and the start of the next. Thus, if the receiver ever loses synchronization it can just search for two *flag bytes* to find the end of the current frame and the start of the next frame. However, there is a still a problem we have to solve. It may happen that the *flag byte* occurs in the data, especially when binary data such as photographs or songs are being transmitted. This situation would interfere with the framing. One way to solve this problem is to have the sender's data link layer insert a special escape byte (**ESC**) just before each ''**accidental**'' *flag byte* in the data. Thus, a framing *flag byte* can be distinguished from one in the data by the absence or presence of an *escape byte* before it. The data link layer on the receiving end removes the escape bytes before giving the data to the network layer. This technique is called **byte stuffing**. Of course, the next question is: what happens if an *escape byte* occurs in the middle of the data? The answer is that it, too, is stuffed with an *escape byte*. At the receiver, the first *escape byte* is removed, leaving the data byte that follows it. Some examples are shown in **Fig. 4.2(b)**. In all cases, the byte sequence delivered after de-stuffing is exactly the same as the original byte sequence. We can still search for a frame boundary by looking for two *flag bytes* in a row, without bothering to undo escapes. The byte-stuffing scheme depicted in **Fig. 4.2** is a slight simplification of the one used in **PPP** (Point-to-Point Protocol), which is used to carry packets over communications links.

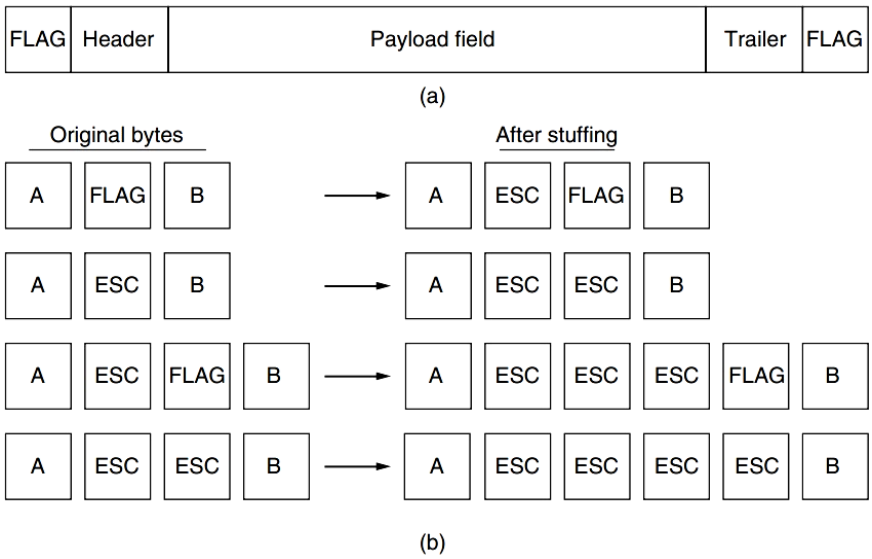


Fig 4.2: (a) A frame delimited by flag bytes. (b) Four examples of byte sequences before and after byte stuffing.

The third method of delimiting the bit stream gets around a disadvantage of byte stuffing, which is that it is tied to the use of 8-bit bytes. Framing can also be done at the bit level, so frames can contain an arbitrary number of bits made up of units of any size. It was developed for the once very popular **HDLC** (High-level Data Link Control) protocol. Each frame begins and ends with a special bit pattern, 01111110 or 0x7E in hexadecimal. This pattern is a *flag byte*. Whenever the sender's data link layer encounters five consecutive 1s in the data, it automatically stuffs a 0 bit into the

outgoing bit stream. This bit stuffing is analogous to byte stuffing, in which an *escape byte* is stuffed into the outgoing character stream before a *flag byte* in the data. It also ensures a minimum density of transitions that help the physical layer maintain synchronization. USB (Universal Serial Bus) uses bit stuffing for this reason. When the receiver sees five consecutive incoming 1 bits, followed by a 0 bit, it automatically de-stuffs (deletes) the 0 bit. Just as byte stuffing is completely transparent to the network layer in both computers, so is bit stuffing. If the user data contain the flag pattern, 01111110, this flag is transmitted as 011111010 but stored in the receiver's memory as 01111110. **Figure 4.3** gives an example of bit stuffing. With bit stuffing, the boundary between two frames can be unambiguously recognized by the flag pattern. Thus, if the receiver loses track of where it is, all it has to do is scan the input for flag sequences, since they can only occur at frame boundaries and never within the data.

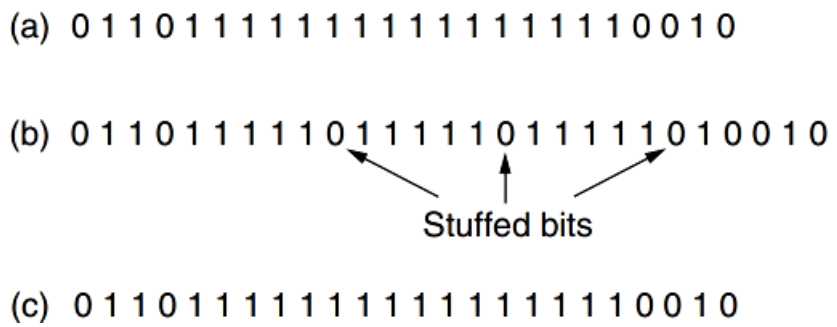


Fig 4.3: Bit stuffing. (a) The original data. (b) The data as they appear on the line. (c) The data as they are stored in the receiver's memory after de-stuffing

With both bit and byte stuffing, a side effect is that the length of a frame now depends on the contents of the data it carries. For instance, if there are no *flag bytes* in the data, 100 bytes might be carried in a frame of roughly 100 bytes. If, however, the data consists solely of *flag bytes*, each flag byte will be escaped and the frame will become roughly 200 bytes long. With bit stuffing, the increase would be roughly 12.5% as 1 bit is added to every byte. The last method of framing is to use a shortcut from the physical layer. Encoding of bits as signals often includes redundancy to help the receiver which means that some signals will not occur in regular data. For example, in the 4B/5B line code 4 data bits are mapped to 5 signal bits to ensure sufficient bit transitions. This means that 16 out of the 32 signal possibilities are not used. We can use some reserved signals to indicate the start and end of frames. In effect, we are using ''**coding violations**'' to delimit frames. The beauty of this scheme is that, because they are reserved signals, it is easy to find the start and end of frames and there is no need to stuff the data. Many data link protocols use a combination of these methods for safety. A common pattern used for Ethernet and 802.11 is to have a frame begin with a well-defined pattern called a **preamble**. This pattern might be quite long (72 bits is typical for 802.11) to allow the receiver to prepare for an incoming packet. The preamble is then followed by a length (i.e., count) field in the header that is used to locate the end of the frame.

4.4 ERROR CONTROL

Having solved the problem of marking the start and end of each frame, we come to the next problem: how to make sure all frames are eventually delivered to the network layer at the destination and in the proper order. Assume for the moment that the receiver can tell whether a frame that it receives contains correct or faulty information. For unacknowledged connectionless service it might be fine if the sender just kept outputting frames without regard to whether they were arriving properly. But for reliable, connection-oriented service it would not be fine at all.

The usual way to ensure reliable delivery is to provide the sender with some feedback about what is happening at the other end of the line. Typically, the protocol calls for the receiver to send back special control frames bearing positive or negative acknowledgements about the incoming frames. If the sender receives a positive acknowledgement about a frame, it knows the frame has arrived safely. On the other hand, a negative acknowledgement means that something has gone wrong and the frame must be transmitted again.

An additional complication comes from the possibility that hardware troubles may cause a frame to vanish completely. In this case, the receiver will not react at all, since it has no reason to react. Similarly, if the acknowledgement frame is lost, the sender will not know how to proceed. It should be clear that a protocol in which the sender transmits a frame and then waits for an acknowledgement, positive or negative, will hang forever if a frame is ever lost due to, for example, malfunctioning hardware or a faulty communication channel. This possibility is dealt with by introducing timers into the data link layer. When the sender transmits a frame, it generally also starts a timer. The timer is set to expire after an interval long enough for the frame to reach the destination, be processed there, and have the acknowledgement propagate back to the sender. Normally, the frame will be correctly received and the acknowledgement will get back before the timer runs out, in which case the timer will be canceled. However, if either the frame or the acknowledgement is lost, the timer will go off, alerting the sender to a potential problem. The obvious solution is to just transmit the frame again. However, when frames may be transmitted multiple times there is a danger that the receiver will accept the same frame two or more times and pass it to the network layer more than once. To prevent this from happening, it is generally necessary to assign sequence numbers to outgoing frames, so that the receiver can distinguish retransmissions from originals.

The whole issue of managing the timers and sequence numbers so as to ensure that each frame is ultimately passed to the network layer at the destination exactly once, no more and no less, is an important part of the duties of the data link layer (and higher layers).

4.5 FLOW CONTROL

Another important design issue that occurs in the data link layer is what to do with a sender that systematically wants to transmit frames faster than the receiver can accept them. This situation can occur when the sender is running on a fast, powerful computer and the receiver is running on a slow, low-end machine. Even if the transmission is error free, the receiver may be unable to handle the frames as fast as they arrive and will lose some. Clearly, something has to be done to prevent this situation. Two approaches are commonly used. In the first one, **feedback-based flow control**, the receiver sends back information to the sender giving it permission to send more data, or at least telling the sender how the receiver is doing. In the second one, **rate-based flow control**, the protocol has a built-in mechanism that limits the rate at which senders may transmit data, without using feedback from the receiver. Feedback-based schemes are seen at both the link layer and higher layers. The latter is more common these days, in which case the link layer hardware is designed to run fast enough that it does not cause loss. For example, hardware implementations of the link layer as NICs (Network Interface Cards) are sometimes said to run at ‘‘wire speed,’’ meaning that they can handle frames as fast as they can arrive on the link. Any overruns are then not a link problem, so they are handled by higher layers. Various feedback-based flow control schemes are known, but most of them use the same basic principle. The protocol contains well-defined rules about when a sender may transmit the next frame. These rules often prohibit frames from being sent until the receiver has granted permission, either implicitly or explicitly.

4.6 ERROR DETECTION AND CORRECTION

Communication channels have a range of characteristics. Some channels, like optical fiber in telecommunications networks, have tiny error rates so that transmission errors are a rare occurrence. But other channels, especially wireless links and aging local loops, have error rates that are orders of magnitude larger. For these links, transmission errors are the norm. They cannot be avoided at a reasonable expense or cost in terms of performance. The conclusion is that transmission errors are here to stay. We have to learn how to deal with them.

Network designers have developed two basic strategies for dealing with errors. Both add redundant information to the data that is sent. One strategy is to include enough redundant information to enable the receiver to deduce what the transmitted data must have been. The other is to include only enough redundancy to allow the receiver to deduce that an error has occurred and have it request a retransmission. The former strategy uses **error-correcting codes** and the latter uses **error-detecting codes**. The use of error-correcting codes is often referred to as **FEC** (Forward Error Correction).

Each of these techniques occupies a different ecological niche. On channels that are highly reliable, such as fiber, it is cheaper to use an error-detecting code and just retransmit the occasional block found to be faulty. However, on channels such as wireless links that make many errors, it is better to add redundancy to each block so that the receiver is able to figure out what the originally transmitted block was. FEC is used on noisy channels because retransmissions are just as likely to be in error as the first transmission. A key consideration for these codes is the type of errors that are likely to occur. Neither error-correcting codes nor error-detecting codes can handle all possible errors since the redundant bits that offer protection are as likely to be received in error as the data bits. It would be nice if the channel treated redundant bits differently than data bits, but it does not. They are all just bits to the channel. This means that to avoid undetected errors the code must be strong enough to handle the expected errors.

One model is that errors are caused by extreme values of thermal noise that overwhelm the signal briefly and occasionally, giving rise to isolated single-bit errors. Another model is that errors tend to come in bursts rather than singly. This model follows from the physical processes that generate them—such as a deep fade on a wireless channel or transient electrical interference on a wired channel. Both models matter in practice, and they have different trade-offs. Having the errors come in bursts has both advantages and disadvantages over isolated single-bit errors. On the advantage side, computer data are always sent in blocks of bits. Suppose that the block size was 1000 bits and the error rate was 0.001 per bit. If errors were independent, most blocks would contain an error. If the errors came in bursts of 100, however, only one block in 100 would be affected, on average. The disadvantage of burst errors is that when they do occur they are much harder to correct than isolated errors. Other types of errors also exist. Sometimes, the location of an error will be known, perhaps because the physical layer received an analog signal that was far from the expected value for a 0 or 1 and declared the bit to be lost. This situation is called an **erasure channel**. It is easier to correct errors in erasure channels than in channels that flip bits because even if the value of the bit has been lost, at least we know which bit is in error. However, we often do not have the benefit of erasures.

We will examine both error-correcting codes and error-detecting codes next. It is important to keep two points in mind, though. First, we cover these codes in the link layer because this is the first place that we have run up against the problem of reliably transmitting groups of bits. However, the codes are widely used because reliability is an overall concern. Error-correcting codes are also seen in the physical layer, particularly for noisy channels, and in higher layers, particularly for real-time media and content distribution. Error-detecting codes are commonly used in link, network, and transport layers. The second point to bear in mind is that error codes are applied mathematics. Unless you are particularly adept at Galois fields or the properties of sparse matrices, you should get codes with good properties from a reliable source rather than making up your own. In fact, this is what many

protocol standards do, with the same codes coming up again and again.

4.6.1 ERROR CORRECTING CODES

We will examine four different error-correcting codes:

1. Hamming codes.
2. Binary convolutional codes.
3. Reed-Solomon codes.
4. Low-Density Parity Check codes.

All of these codes add redundancy to the information that is sent. A frame consists of ***m*** data (i.e., message) bits and ***r*** redundant (i.e. check) bits. In a block code, the ***r*** check bits are computed solely as a function of the ***m*** data bits with which they are associated, as though the ***m*** bits were looked up in a large table to find their corresponding ***r*** check bits. In a systematic code, the ***m*** data bits are sent directly, along with the check bits, rather than being encoded themselves before they are sent. In a linear code, the ***r*** check bits are computed as a linear function of the ***m*** data bits. Exclusive OR (XOR) or modulo 2 addition is a popular choice. This means that encoding can be done with operations such as matrix multiplications or simple logic circuits. The codes we will look at in this section are linear, systematic block codes unless otherwise noted.

Let the total length of a block be ***n*** (i.e., ***n=m+r***). We will describe this as an (***n***, ***m***) code. An ***n***-bit unit containing data and check bits is referred to as an ***n***-bit code word. The code rate, or simply rate, is the fraction of the code word that carries information that is not redundant, or ***m/n***. The rates used in practice vary widely. They might be 1/2 for a noisy channel, in which case half of the received information is redundant, or close to 1 for a high-quality channel, with only a small number of check bits added to a large message. To understand how errors can be handled, it is necessary to first look closely at what an error really is. Given any two code words that may be transmitted or received—say, 10001001 and 10110001—it is possible to determine how many corresponding bits differ. In this case, 3 bits differ. To determine how many bits differ, just XOR the two code words and count the number of 1 bits in the result.

For example:

```
  10001001
  10110001
  --------
  00111000
```

The number of bit positions in which two code words differ is called the **Hamming distance** (Hamming, 1950). Its significance is that if two code words are a Hamming distance ***d*** apart, it will require ***d*** single-bit errors to convert one into the other. Given the algorithm for computing the check bits, it is possible to construct a complete list of the legal code words, and from this list to find the two code words with the smallest Hamming distance. This distance is the Hamming distance of the complete code. In most data transmission applications, all $\mathbf{2^m}$ possible data messages are legal,

but due to the way the check bits are computed, not all of the 2^n possible code words are used. In fact, when there are ***r*** check bits, only the small fraction of $2^m/2^n$ or $1/2^r$ of the possible messages will be legal code-words. It is the sparseness with which the message is embedded in the space of code-words that allows the receiver to detect and correct errors. The error-detecting and error-correcting properties of a block code depend on its **Hamming distance**. To reliably detect ***d*** errors, you need a distance ***d*+1** code because with such a code there is no way that ***d*** single-bit errors can change a valid code word into another valid code word. When the receiver sees an illegal code-word, it can tell that a transmission error has occurred. Similarly, to correct ***d*** errors, you need a distance **2*d*+1** code because that way the legal code-words are so far apart that even with ***d*** changes the original code-word is still closer than any other code-word. This means the original code-word can be uniquely determined based on the assumption that a larger number of errors are less likely.

As a simple example of an error-correcting code, consider a code with only four valid code-words:

0000000000, 0000011111, 1111100000, and 1111111111

This code has a distance of 5, which means that it can correct double errors or detect quadruple errors. If the code-word 0000000111 arrives and we expect only single- or double-bit errors, the receiver will know that the original must have been 0000011111. If, however, a triple error changes 0000000000 into 0000000111, the error will not be corrected properly. Alternatively, if we expect all of these errors, we can detect them. None of the received code-words are legal code-words so an error must have occurred. It should be apparent that in this example we cannot both correct double errors and detect quadruple errors because this would require us to interpret a received code-word in two different ways. In our example, the task of decoding by finding the legal code-word that is closest to the received code-word can be done by inspection. Unfortunately, in the most general case where all code-words need to be evaluated as candidates, this task can be a time-consuming search. Instead, practical codes are designed so that they admit shortcuts to find what was likely the original code-word. Imagine that we want to design a code with ***m*** message bits and ***r*** check bits that will allow all single errors to be corrected. Each of the 2^m legal messages has ***n*** illegal code words at a distance of 1 from it. These are formed by systematically inverting each of the ***n*** bits in the ***n***-bit code-word formed from it. Thus, each of the 2^m legal messages requires ***n*+1** bit patterns dedicated to it. Since the total number of bit patterns is 2^n, we must have $(n+1)2^m \leq 2^n$. Using $n=m+r$, this requirement becomes

$$(m+r+1) \leq 2^r \quad \text{——— 4.1}$$

Given ***m***, this puts a lower limit on the number of check bits needed to correct single errors. This theoretical lower limit can, in fact, be achieved using a method due to Hamming (1950). In Hamming codes the bits of the code-word are numbered consecutively, starting with bit 1 at the left end, bit 2 to its immediate right, and so on. The bits that are powers of 2 (1, 2, 4,

8, 16, etc.) are check bits. The rest (3, 5, 6, 7, 9, etc.) are filled up with the ***m*** data bits. This pattern is shown for an (11,7) Hamming code with 7 data bits and 4 check bits in **Fig. 4.4**. Each check bit forces the modulo 2 sum, or parity, of some collection of bits, including itself, to be even (or odd). A bit may be included in several check bit computations. To see which check bits the data bit in position ***k*** contributes to, rewrite ***k*** as a sum of powers of 2. For example, 11 = 1 + 2 + 8 and 29 = 1 + 4 + 8 + 16. A bit is checked by just those check bits occurring in its expansion (e.g., bit 11 is checked by bits 1, 2, and 8). In the example, the check bits are computed for even parity sums for a message that is the ASCII letter ‘‘**A**’’.

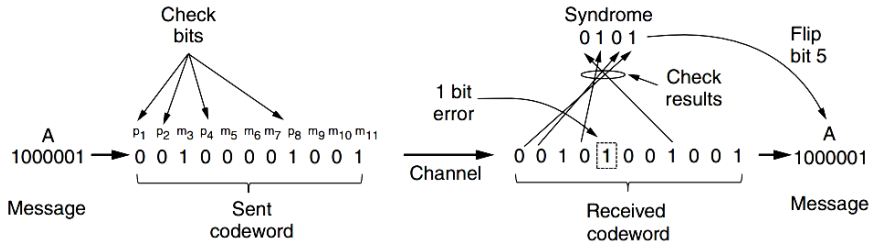


Fig 4.4: Example of an (11, 7) Hamming code correcting a single-bit error.

This construction gives a code with a Hamming distance of 3, which means that it can correct single errors (or detect double errors). The reason for the very careful numbering of message and check bits becomes apparent in the decoding process. When a code-word arrives, the receiver redoes the check bit computations including the values of the received check bits. We call these the check results. If the check bits are correct then, for even parity sums, each check result should be zero. In this case the code-word is accepted as valid. If the check results are not all zero, however, an error has been detected. The set of check results forms the **error syndrome** that is used to pinpoint and correct the error. In **Fig. 4.4**, a single-bit error occurred on the channel so the check results are 0, 1, 0, and 1 fork = 8, 4, 2, and 1, respectively. This gives a syndrome of 0101 or 4+1=5. By the design of the scheme, this means that the fifth bit is in error. Flipping the incorrect bit (which might be a check bit or a data bit) and discarding the check bits gives the correct message of an ASCII ‘‘**A**’’. Hamming distances are valuable for understanding block codes, and Hamming codes are used in error-correcting memory. However, most networks use stronger codes. The second code we will look at is a **convolutional** code. This code is the only one we will cover that is not a block code. In a convolutional code, an encoder processes a sequence of input bits and generates a sequence of output bits. There is no natural message size or encoding boundary as in a block code. The output depends on the current and previous input bits. That is, the encoder has memory. The number of previous bits on which the output depends is called the constraint length of the code. Convolutional codes are specified in terms of their rate and constraint length. Convolutional codes are widely used in deployed networks, for example, as part of the GSM mobile phone system, in satellite communications, and

in 802.11. As an example, a popular convolutional code is shown in **Fig. 4.5**. This code is known as the NASA convolutional code of **r=1/2** and **k=7**, since it was first used for the Voyager space missions starting in 1977. Since then it has been liberally reused, for example, as part of 802.11.

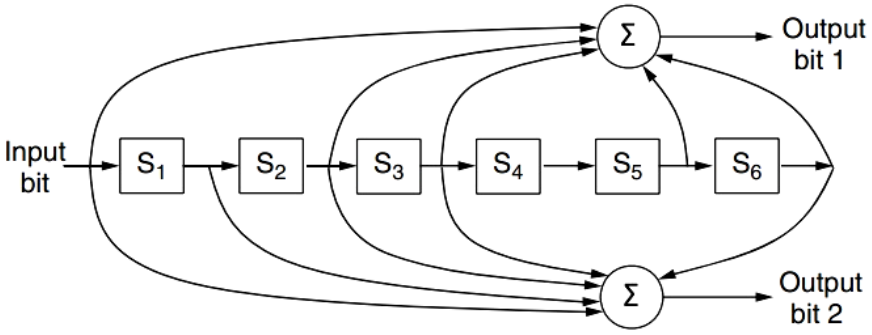


Fig 4.5: The NASA binary convolutional code used in 802.11

In **Fig. 4.5**, each input bit on the left-hand side produces two output bits on the right-hand side that are XOR sums of the input and internal state. Since it deals with bits and performs linear operations, this is a binary, linear convolutional code. Since 1 input bit produces 2 output bits, the code rate is 1/2. It is not systematic since none of the output bits is simply the input bit. The internal state is kept in six memory registers. Each time another bit is input the values in the registers are shifted to the right. For example, if 111 is input and the initial state is all zeros, the internal state, written left to right, will become 100000, 110000, and 111000 after the first, second, and third bits have been input. The output bits will be 11, followed by 10, and then 01. It takes seven shifts to flush an input completely so that it does not affect the output. The constraint length of this code is thus **k=7**. A convolutional code is decoded by finding the sequence of input bits that is most likely to have produced the observed sequence of output bits (which includes any errors). For small values of ***k***, this is done with a widely used algorithm developed by Viterbi (Forney, 1973). The algorithm walks the observed sequence, keeping for each step and for each possible internal state the input sequence that would have produced the observed sequence with the fewest errors. The input sequence requiring the fewest errors at the end is the most likely message. Convolutional codes have been popular in practice because it is easy to factor the uncertainty of a bit being a 0 or a 1 into the decoding. For example, suppose **−1V** is the logical 0 level and **+1V** is the logical 1 level, we might receive 0.9V and−0.1V for 2 bits. Instead of mapping these signals to 1 and 0 right away, we would like to treat 0.9V as ‘‘very likely a 1’’ and −0.1V as ‘‘maybe a 0’’ and correct the sequence as a whole. Extensions of the Viterbi algorithm can work with these uncertainties to provide stronger error correction. This approach of working with the uncertainty of a bit is called **soft-decision decoding**. Conversely, deciding whether each bit is a 0 or a 1 before subsequent error correction is called **hard-decision decoding**.

The third kind of error-correcting code we will describe is the **Reed-Solomon code**. Like Hamming codes, Reed-Solomon

codes are linear block codes, and they are often systematic too. Unlike Hamming codes, which operate on individual bits, Reed-Solomon codes operate on ***m*** bit symbols. Naturally, the mathematics are more involved, so we will describe their operation by analogy. Reed-Solomon codes are based on the fact that every ***n*** degree polynomial is uniquely determined by **n+1** points. For example, a line having the form **ax+b** is determined by two points. Extra points on the same line are redundant, which is helpful for error correction. Imagine that we have two data points that represent a line and we send those two data points plus two check points chosen to lie on the same line. If one of the points is received in error, we can still recover the data points by fitting a line to the received points. Three of the points will lie on the line, and one point, the one in error, will not. By finding the line we have corrected the error. Reed-Solomon codes are actually defined as polynomials that operate over finite fields, but they work in a similar manner. For ***m*** bit symbols, the code-words are $\mathbf{2^m-1}$ symbols long. A popular choice is to make **m=8** so that symbols are bytes. A code-word is then 255 bytes long. The (255, 233) code is widely used; it adds 32 redundant symbols to 233 data symbols. Decoding with error correction is done with an algorithm developed by Berlekamp and Massey that can efficiently perform the fitting task for moderate-length codes (Massey, 1969). Reed-Solomon codes are widely used in practice because of their strong error-correction properties, particularly for burst errors. They are used for DSL, data over cable, satellite communications, and perhaps most ubiquitously on CDs, DVDs, and Blu-ray discs. Because they are based on ***m*** bit symbols, a single-bit error and an ***m*-bit** burst error are both treated simply as one symbol error. When ***2t*** redundant symbols are added, a Reed-Solomon code is able to correct up to ***t*** errors in any of the transmitted symbols. This means, for example, that the (255, 233) code, which has 32 redundant symbols, can correct up to 16 symbol errors. Since the symbols may be consecutive and they are each 8 bits, an error burst of up to 128 bits can be corrected. The situation is even better if the error model is one of erasures (e.g., a scratch on a CD that obliterates some symbols). In this case, up to ***2t*** errors can be corrected. Reed-Solomon codes are often used in combination with other codes such as a convolutional code. The thinking is as follows. Convolutional codes are effective at handling isolated bit errors, but they will fail, likely with a burst of errors, if there are too many errors in the received bit stream. By adding a Reed-Solomon code within the convolutional code, the Reed-Solomon decoding can mop up the error bursts, a task at which it is very good. The overall code then provides good protection against both single and burst errors.

The final error-correcting code we will cover is the **LDPC** (**Low-Density Parity Check**) code. LDPC codes are linear block codes that were invented by Robert Gallagher in his doctoral thesis (Gallagher, 1962). Like most theses, they were promptly forgotten, only to be reinvented in 1995 when advances in computing power had made them practical. In an LDPC code, each output bit is formed from only a fraction of the input bits. This leads to a matrix representation of the code that has a low density of 1s, hence the name for the code. The received code-words are decoded with an

approximation algorithm that iteratively improves on a best fit of the received data to a legal code-word. This corrects errors. LDPC codes are practical for large block sizes and have excellent error-correction abilities that outperform many other codes in practice. For this reason they are rapidly being included in new protocols. They are part of the standard for digital video broadcasting, 10 Gbps Ethernet, power-line networks, and the latest version of 802.11.

4.6.2 ERROR DETECTING CODES

Error-correcting codes are widely used on wireless links, which are notoriously noisy and error prone when compared to optical fibers. Without error-correcting codes, it would be hard to get anything through. However, over fiber or high-quality copper, the error rate is much lower, so error detection and retransmission is usually more efficient therefor dealing with the occasional error.

We will examine three different error-detecting codes. They are all linear, systematic block codes:

1. Parity.
2. Checksums.
3. Cyclic Redundancy Checks (CRCs).

To see how they can be more efficient than error-correcting codes, consider the first error-detecting code, in which a single **parity bit** is appended to the data. The parity bit is chosen so that the number of 1 bits in the code-word is even (or odd). Doing this is equivalent to computing the (even) parity bit as the modulo 2 sum or XOR of the data bits. For example, when 1011010 is sent in even parity, a bit is added to the end to make it 10110100. With odd parity 1011010 becomes 10110101. A code with a single parity bit has a distance of 2, since any single-bit error produces a code-word with the wrong parity. This means that it can detect single-bit errors. Consider a channel on which errors are isolated and the error rate is $\mathbf{10^{-6}}$ per bit. This may seem a tiny error rate, but it is at best a fair rate for a long wired cable that is challenging for error detection. Typical LAN links provide bit error rates of $\mathbf{10^{-10}}$. Let the block size be 1000 bits. To provide error correction for 1000-bit blocks, we know from **Equation 4.1** that 10 check bits are needed. Thus, a megabit of data would require 10,000 check bits. To merely detect a block with a single 1-bit error, one parity bit per block will suffice. Once every 1000 blocks, a block will be found to be in error and an extra block (1001 bits) will have to be transmitted to repair the error. The total overhead for the error detection and re-transmission method is only 2001 bits per megabit of data, versus 10,000 bits for a Hamming code. One difficulty with this scheme is that a single parity bit can only reliably detect a single-bit error in the block. If the block is badly garbled by a long burst error, the probability that the error will be detected is only 0.5, which is hardly acceptable. The odds can be improved considerably if each block to be sent is regarded as a rectangular matrix $\boldsymbol{n}$ bits wide and $\boldsymbol{k}$ bits high. Now, if we compute and send one parity bit for each row, up to $\boldsymbol{k}$ bit errors will be reliably detected as long as there is at most one error per row. However,

there is something else we can do that provides better protection against burst errors: we can compute the parity bits over the data in a different order than the order in which the data bits are transmitted. Doing so is called **interleaving**. In this case, we will compute a parity bit for each of the ***n*** columns and send all the data bits as ***k*** rows, sending the rows from top to bottom and the bits in each row from left to right in the usual manner. At the last row, we send the ***n*** parity bits. This transmission order is shown in **Fig. 4.6** for n=7 and k=7.

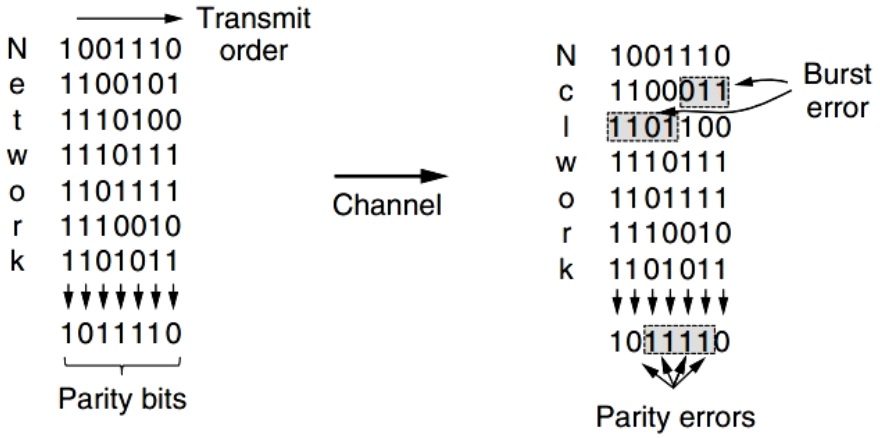


Fig 4.6: Interleaving of parity bits to detect a burst error

Interleaving is a general technique to convert a code that detects (or corrects) isolated errors into a code that detects (or corrects) burst errors. In **Fig. 4.6**, when a burst error of length **n=7** occurs, the bits that are in error are spread across different columns. (A burst error does not imply that all the bits are wrong; it just implies that at least the first and last are wrong. In **Fig. 4.6**, 4 bits were flipped over a range of 7 bits.) At most 1 bit in each of the ***n*** columns will be affected, so the parity bits on those columns will detect the error. This method uses ***n*** parity bits on blocks of ***kn*** data bits to detect a single burst error of length ***n*** or less. A burst of length **n+1** will pass undetected, however, if the first bit is inverted, the last bit is inverted, and all the other bits are correct. If the block is badly garbled by a long burst or by multiple shorter bursts, the probability that any of the ***n*** columns will have the correct parity by accident is 0.5, so the probability of a bad block being accepted when it should not be is $\mathbf{2^{-n}}$.

The second kind of error-detecting code, the **checksum**, is closely related to groups of parity bits. The word ‘‘checksum’’ is often used to mean a group of check bits associated with a message, regardless of how are calculated. A group of parity bits is one example of a checksum. However, there are other, stronger checksums based on a running sum of the data bits of the message. The checksum is usually placed at the end of the message, as the complement of the sum function. This way, errors may be detected by summing the entire received code-word, both data bits and checksum. If the result comes out to be zero, no error has been detected. One example of a checksum is the 16-bit Internet checksum used on all Internet packets as part of the IP protocol (Braden et al., 1988). This checksum is a sum of the message bits divided into16-bit words. Because this method

operates on words rather than on bits, as in parity, errors that leave the parity unchanged can still alter the sum and be detected. For example, if the lowest order bit in two different words is flipped from a 0 to a 1, a parity check across these bits would fail to detect an error. However, two 1s will be added to the 16-bit checksum to produce a different result. The error can then be detected. The Internet checksum is computed in one's complement arithmetic instead of as the modulo $\mathbf{2^{16}}$ sum. In one's complement arithmetic, a negative number is the bitwise complement of its positive counterpart. Modern computers run two's complement arithmetic, in which a negative number is the one's complement plus one. On a two's complement computer, the one's complement sum is equivalent to taking the sum modulo $\mathbf{2^{16}}$ and adding any overflow of the high order bits back into the low-order bits. This algorithm gives a more uniform coverage of the data by the checksum bits. Otherwise, two high-order bits can be added, overflow, and be lost without changing the sum. There is another benefit, too. One's complement has two representations of zero, all 0s and all 1s. This allows one value (e.g., all 0s) to indicate that there is no checksum, without the need for another field. For decades, it has always been assumed that frames to be check-summed contain random bits. All analyses of checksum algorithms have been made under this assumption. Inspection of real data by Partridge et al. (1995) has shown this assumption to be quite wrong. As a consequence, undetected errors are in some cases much more common than had been previously thought. The Internet checksum in particular is efficient and simple but provides weak protection in some cases precisely because it is a simple sum. It does not detect the deletion or addition of zero data, nor swapping parts of the message, and it provides weak protection against message splices in which parts of two packets are put together. These errors may seem very unlikely to occur by random processes, but they are just the sort of errors that can occur with buggy hardware. A better choice is **Fletcher's checksum** (Fletcher, 1982). It includes a positional component, adding the product of the data and its position to the running sum. This provides stronger detection of changes in the position of data.

Although the two preceding schemes may sometimes be adequate at higher layers, in practice, a third and stronger kind of error-detecting code is in wide-spread use at the link layer: the **CRC** (**Cyclic Redundancy Check**), also known as a **polynomial code**. Polynomial codes are based upon treating bit strings as representations of polynomials with coefficients of 0 and 1 only. A **k-bit** frame is regarded as the coefficient list for a polynomial with $\boldsymbol{k}$ terms, ranging from $\mathbf{x^{k-1}}$ to $\mathbf{x^0}$. Such a polynomial is said to be of degree **k−1**. The high-order (leftmost) bit is the coefficient of $\mathbf{x^{k-1}}$, the next bit is the coefficient of $\mathbf{x^{k-2}}$, and so on. For example, 110001 has 6 bits and thus represents a six-term polynomial with coefficients 1, 1, 0, 0, 0, and 1: $\mathbf{1}x^5+\mathbf{1}x^4+\mathbf{0}x^3+\mathbf{0}x^2+\mathbf{0}x^1+\mathbf{1}x^0$. Polynomial arithmetic is done modulo 2, according to the rules of algebraic field theory. It does not have carries for addition or borrows for subtraction. Both addition and subtraction are identical to exclusive OR. For example:

```
  10011011      00110011      11110000      01010101
+ 11001010    + 11001101    - 10100110    - 10101111
----------    ----------    ----------    ----------
  01010001      11111110      01010110      11111010
```

Long division is carried out in exactly the same way as it is in binary except that the subtraction is again done modulo 2. A divisor is said ‘‘to go into’’ a dividend if the dividend has as many bits as the divisor. When the polynomial code method is employed, the sender and receiver must agree upon a **generator polynomial**, G(*x*), in advance. Both the high- and low-order bits of the generator must be 1. To compute the CRC for some frame with ***m*** bits corresponding to the polynomial ***M*(*x*)**, the frame must be longer than the generator polynomial. The idea is to append a CRC to the end of the frame in such a way that the polynomial represented by the check-summed frame is divisible by ***G*(*x*)**. When the receiver gets the check-summed frame, it tries dividing it by ***G*(*x*)**. If there is a remainder, there has been a transmission error.

The algorithm for computing the CRC is as follows:

1. Let ***r*** be the degree of ***G*(*x*)**. Append ***r*** zero bits to the low-order end of the frame so it now contains ***m+r*** bits and corresponds to the polynomial $x^r M(x)$.
2. Divide the bit string corresponding to **G(x)** into the bit string corresponding to $x^r M(x)$, using modulo 2 division.
3. Subtract the remainder (which is always ***r*** or fewer bits) from the bit string corresponding to $x^r M(x)$ using modulo 2 subtraction. The result is the check-summed frame to be transmitted. Call its polynomial **T(x)**.

Figure 4.7 illustrates the calculation for a frame 1101011111 using the generator $G(x)=x^4+x+1$.

It should be clear that **T(x)** is divisible (modulo 2) by **G(x)**. In any division problem, if you diminish the dividend by the remainder, what is left over is divisible by the divisor. For example, in base 10, if you divide 210,278 by 10,941, the remainder is 2399. If you then subtract 2399 from 210,278, what is left over (207,879) is divisible by 10,941. Now let us analyze the power of this method. What kinds of errors will be detected? Imagine that a transmission error occurs, so that instead of the bit string for **T(x)** arriving, **T(x)+E(x)** arrives. Each 1 bit in **E(x)** corresponds to a bit that has been inverted. If there are k 1 bits in **E(x)**, ***k*** single-bit errors have occurred. A single burst error is characterized by an initial 1, a mixture of 0s and 1s, and a final 1, with all other bits being 0. Upon receiving the check-summed frame, the receiver divides it by **G(x)**; that is, it computes **[T(x)+E(x)]/G(x)**. **T(x)/G(x)** is 0, so the result of the computation is simply **E(x)/G(x)**. Those errors that happen to correspond to polynomials containing **G(x)** as a factor will slip by; all other errors will be caught. If there has been a single-bit error, $E(x)=x^i$, where **i** determines which bit is in error. If **G(x)** contains two or more terms, it will never divide into **E(x)**, so all single-bit errors will be detected.

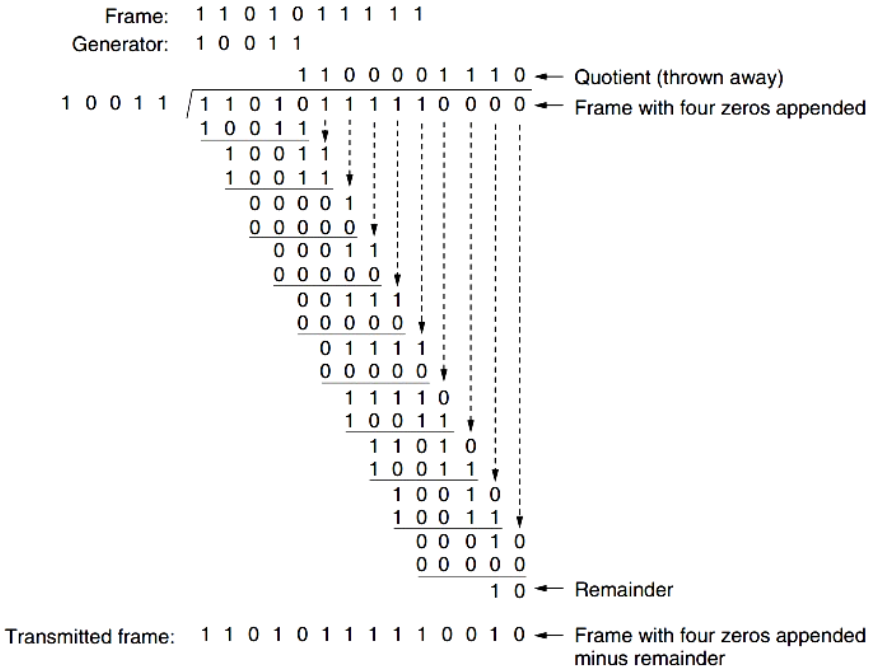


Fig 4.7: Example calculation of the CRC

If there have been two isolated single-bit errors, $\mathbf{E(x)=x^i+x^j}$, where **i>j**. Alternatively, this can be written as $\mathbf{E(x)=x^j(x^{i-j}+1)}$. If we assume that **G(x)** is not divisible by **x**, a sufficient condition for all double errors to be detected is that **G(x)** does not divide $\mathbf{x^k+1}$ for any **k** up to the maximum value of **i−j** (i.e., up to the maximum frame length). Simple, low-degree polynomials that give protection to long frames are known. For example, $\mathbf{x^{15}+x^{14}+1}$ will not divide $\mathbf{x^k+1}$ for any value of **k** below 32,768. If there are an odd number of bits in error, **E(X)** contains an odd number of terms (e.g., x^5+x^2+1, but not x^2+1). Interestingly, no polynomial with an odd number of terms has **x+1** as a factor in the modulo 2 system. By making **x+1** a factor of **G(x)**, we can catch all errors with an odd number of inverted bits.

Finally, and importantly, a polynomial code with ***r*** check bits will detect all burst errors of length **≤r**. A burst error of length ***k*** can be represented by $\mathbf{x^i(x^{k-1}+...+1)}$, where **i** determines how far from the right-hand end of the received frame the burst is located. If **G(x)** contains an $\mathbf{x^0}$ term, it will not have $\mathbf{x^i}$ as a factor, so if the degree of the parenthesized expression is less than the degree of **G(x)**, the remainder can never be zero. If the burst length is **r+1**, the remainder of the division by **G(x)** will be zero if and only if the burst is identical to **G(x)**. By definition of a burst, the first and last bits must be 1, so whether it matches depends on the **r−1** intermediate bits. If all combinations are regarded as equally likely, the probability of such an incorrect frame being accepted as valid is $½^{r-1}$. It can also be shown that when an error burst longer than **r+1** bits occurs or when several shorter bursts occur, the probability of a bad frame getting through unnoticed is $½^r$, assuming that all bit patterns are equally likely.

Certain polynomials have become international standards. The one used in IEEE 802 followed the example of Ethernet and is

$$x^{32}+x^{26}+x^{23}+x^{22}+x^{16}+x^{12}+x^{11}+x^{10}+x^8+x^7+x^5+x^4+x^2+x^1+1$$

Among other desirable properties, it has the property that it detects all bursts of length 32 or less and all bursts affecting an odd number of bits. It has been used widely since the 1980s. However, this does not mean it is the best choice. Using an exhaustive computational search, Castagnoli et al. (1993) and Koopman (2002) found the best CRCs. These CRCs have a Hamming distance of 6 for typical message sizes, while the IEEE standard CRC-32 has a Hamming distance of only 4. Although the calculation required to compute the CRC may seem complicated, it is easy to compute and verify CRCs in hardware with simple shift register circuits (Peterson and Brown, 1961). In practice, this hardware is nearly always used. Dozens of networking standards include various CRCs, including virtually all LANs (e.g., Ethernet, 802.11) and point-to-point links (e.g., packets over SONET).

CHECK YOUR PROGRESS

1. Fill in the blanks:

   (a) The _____ layer is the protocol layer that transfers data between adjacent network nodes in a wide area network.

   (b) The _____ flow control protocol has a built-in mechanism that limits data transmission rate.

   (c) _____ codes include enough redundant information to enable the receiver to deduce what the transmitted data must have been.

   (d) _____ codes include only enough redundancy to allow the receiver to deduce that an error has occurred and have it request a retransmission.

   (e) The error-detecting and error-correcting properties of a block code depend on its _____.

   (f) In a _____, an encoder processes a sequence of input bits and generates a sequence of output bits.

   (g) _____ are based on the fact that every n degree polynomial is determined by n+1 points.

   (h) _____ are practical for large block sizes and have excellent error-correction abilities.

   (i) The _____ is usually placed at the end of the message.

   (j) _____ are based upon treating bit strings as representations of polynomials with coefficients of 0 and 1 only.

4.7 DATA LINK PROTOCOLS

To introduce the subject of protocols, we will begin by looking at three protocols of increasing complexity. Before we look at the protocols, it is useful to make explicit some of the assumptions underlying the model of communication. To start with, we assume that the physical layer, data link layer, and network layer are independent processes that communicate by passing messages back and forth. A common implementation is shown in **Fig. 4.8**. The physical layer process and some of the data link layer process run on dedicate hardware called a NIC (**Network Interface Card**). The rest of the link layer process and the network layer process run on the main CPU as part of the operating system, with the software for the link layer process often taking the form of a device driver. However, other implementations are also possible (e.g., three processes offloaded to dedicated hardware called a **network accelerator**, or three processes running on the main CPU on a software-defined ratio). Actually, the preferred implementation changes from decade to decade with technology trade-offs. In any event, treating the three layers as separate processes makes the discussion conceptually cleaner and also serves to emphasize the independence of the layers.

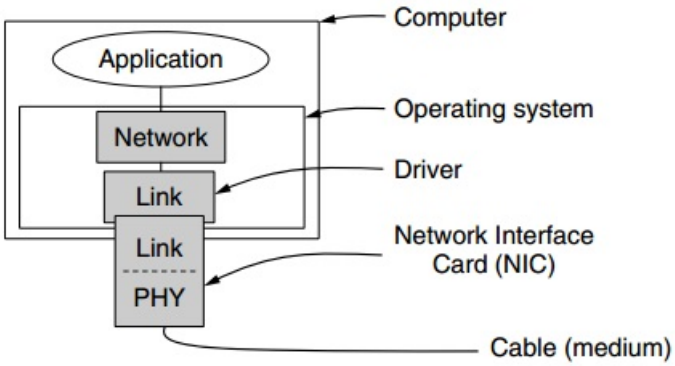


Fig 4.8 Implementation of the physical, data link, and network layers

Another key assumption is that machine A wants to send a long stream of data to machine B, using a reliable, connection-oriented service. A is assumed to have an infinite supply of data ready to send and never has to wait for data to be produced. Instead, when A's data link layer asks for data, the network layer is always able to comply immediately. We also assume that machines do not crash. That is, these protocols deal with communication errors, but not the problems caused by computers crashing and rebooting. As far as the data link layer is concerned, the packet passed across the interface to it from the network layer is pure data, whose every bit is to be delivered to the destination's network layer. The fact that the destination's network layer may interpret part of the packet as a header is of no concern to the data link layer. When the data link layer accepts a packet, it encapsulates the packet in a frame by adding a data link header and trailer to it. Thus, a frame consists of

an embedded packet, some control information (in the header), and a checksum (in the trailer). The frame is then transmitted to the data link layer on the other machine. We will assume that there exist suitable library procedures *to physical layer* to send a frame and *from physical layer* to receive a frame. These procedures compute and append or check the checksum (which is usually done in hardware) so that we do not need to worry about it as part of the protocols we develop in this section for example. Initially, the receiver has nothing to do. It just sits around waiting for something to happen. We indicate that the data link layer is waiting for something to happen by the procedure call ***wait_for_event* (&*event*)**. This procedure only returns when something has happened (e.g., a frame has arrived). Upon return, the variable *event* tells what happened. The set of possible events differs for the various protocols to be described and will be defined separately for each protocol. Note that in a more realistic situation, the data link layer will not sit in a tight loop waiting for an event, as we have suggested, but will receive an interrupt, which will cause it to stop whatever it was doing and go handle the incoming frame. Nevertheless, for simplicity we will ignore all the details of parallel activity within the data link layer and assume that it is dedicated full time to handling just our one channel.

```
#define MAX_PKT 1024                                    /* determines packet size in bytes */

typedef enum {false, true} boolean;                     /* boolean type */
typedef unsigned int seq_nr;                            /* sequence or ack numbers */
typedef struct {unsigned char data[MAX_PKT];} packet;   /* packet definition */
typedef enum {data, ack, nak} frame_kind;               /* frame_kind definition */

typedef struct {                                        /* frames are transported in this layer */
  frame_kind kind;                                      /* what kind of frame is it? */
  seq_nr seq;                                           /* sequence number */
  seq_nr ack;                                           /* acknowledgement number */
  packet info;                                          /* the network layer packet */
} frame;


/* Wait for an event to happen; return its type in event. */
void wait_for_event(event_type *event);

/* Fetch a packet from the network layer for transmission on the channel. */
void from_network_layer(packet *p);

/* Deliver information from an inbound frame to the network layer. */
void to_network_layer(packet *p);

/* Go get an inbound frame from the physical layer and copy it to r. */
void from_physical_layer(frame *r);

/* Pass the frame to the physical layer for transmission. */
void to_physical_layer(frame *s);

/* Start the clock running and enable the timeout event. */
void start_timer(seq_nr k);

/* Stop the clock and disable the timeout event. */
void stop_timer(seq_nr k);

/* Start an auxiliary timer and enable the ack_timeout event. */
void start_ack_timer(void);

/* Stop the auxiliary timer and disable the ack_timeout event. */
void stop_ack_timer(void);

/* Allow the network layer to cause a network_layer_ready event. */
void enable_network_layer(void);

/* Forbid the network layer from causing a network_layer_ready event. */
void disable_network_layer(void);

/* Macro inc is expanded in-line: increment k circularly. */
#define inc(k) if (k < MAX_SEQ) k = k + 1; else k = 0
```

Fig 4.9: Some definitions needed in the protocols to follow. These definitions are located in the file protocol.h

When a frame arrives at the receiver, the checksum is recomputed. If the checksum in the frame is incorrect (i.e., there was a transmission error), the data link layer is so informed (***event=cksum err***). If the inbound frame arrived undamaged, the data link layer is also informed (***event=frame_arrival***) so that it can acquire the frame for inspection using ***from_physical_layer***. As soon as the receiving data link layer has acquired an undamaged frame, it checks the control information in the header, and, if everything is all right, passes the packet portion to the network layer. Under no circumstances is a frame header ever given to a network layer. There is a good reason why the network layer must never be given any part of the frame header: to keep the network and data link protocols completely separate. As long as the network layer knows nothing at all about the data link protocol or the frame format, these things can be changed without requiring changes to the network layer's software. This happens whenever a new NIC is installed in a computer. Providing a rigid interface between the network and data link layers greatly simplifies the design task because communication protocols in different layers can evolve independently. **Figure 4.9** shows some declarations (in C) common to many of the protocols to be discussed later. Five data structures are defined there: ***boolean***, ***seq_nr***, ***packet***, ***frame_kind***, and ***frame***. A *boolean* is an enumerated type and can take on the values *true* and *false*. A *seq_nr* is a small integer used to number the frames so that we can tell them apart. These sequence numbers run from 0 up to and including *MAX_SEQ*, which is defined in each protocol needing it. A *packet* is the unit of information exchanged between the network layer and the data link layer on the same machine, or between network layer peers. In our model it always contains *MAX_PKT* bytes, but more realistically it would be of variable length. A frame is composed of four fields: ***kind***, ***seq***, ***ack***, and ***info***, the first three of which contain control information and the last of which may contain actual data to be transferred. These control fields are collectively called the frame header. The *kind* field tells whether there are any data in the frame, because some of the protocols distinguish frames containing only control information from those containing data as well. The *seq* and *ack* fields are used for sequence numbers and acknowledgements, respectively. The *info* field of a data frame contains a single packet; the *info* field of a control frame is not used. A more realistic implementation would use a variable-length *info* field, omitting it altogether for control frames.

Again, it is important to understand the relationship between a packet and a frame. The network layer builds a packet by taking a message from the transport layer and adding the network layer header to it. This packet is passed to the data link layer for inclusion in the info field of an outgoing frame. When the frame arrives at the destination, the data link layer extracts the packet from the frame and passes the packet to the network layer. In this manner, the network layer can act as though machines can exchange packets directly. A number of procedures are also listed in **Fig. 4.9**. These are library routines whose details are implementation dependent. The procedure *wait_for_event* sits in a tight loop waiting for something to happen. The procedures

to_network_layer and *from_network_layer* are used by the data link layer to pass packets to the network layer and accept packets from the network layer, respectively. Note that from physical layer and to physical layer pass frames between the data link layer and the physical layer. In other words, *to_network_layer* and *from_network_layer* deal with the interface between layers 2 and 3, whereas from physical layer and to physical layer deal with the interface between layers 1 and 2.

In most of the protocols, we assume that the channel is unreliable and loses entire frames upon occasion. To be able to recover from such calamities, the sending data link layer must start an internal timer or clock whenever it sends a frame. If no reply has been received within a certain predetermined time interval, the clock times out and the datalink layer receives an interrupt signal. In our protocols this is handled by allowing the procedure *wait_for_event* to return *event=timeout*. The procedures *start_timer* and *stop_timer* turn the timer on and off, respectively. Timeout events are possible only when the timer is running and before *stop_timer* is called. It is explicitly permitted to call *start_timer* while the timer is running; such a call simply resets the clock to cause the next timeout after a full timer interval has elapsed (unless it is reset or turned off). The procedures *start_ack_timer* and *stop_ack_timer* control an auxiliary timer used to generate acknowledgements under certain conditions. The procedures *enable_network_layer* and *disable_network_layer* are used in the more sophisticated protocols, where we no longer assume that the network layer always has packets to send. When the data link layer enables the network layer, the network layer is then permitted to interrupt when it has a packet to be sent. We indicate this with *event = network_layer_ready*. When the network layer is disabled, it may not cause such events. By being careful about when it enables and disables its network layer, the data link layer can prevent the network layer from swamping it with packets for which it has no buffer space. Frame sequence numbers are always in the range 0 to *MAX_SEQ*, where *MAX_SEQ* is different for the different protocols. It is frequently necessary to advance a sequence number by 1 circularly (i.e., *MAX_SEQ* is followed by 0). The macro *inc* performs this incrementing. It has been defined as a macro because it is used in-line within the critical path. The factor limiting network performance is often protocol processing, so defining simple operations like this as macros does not affect the readability of the code but does improve performance.

4.7.1 STOP AND WAIT ARQ

Now we will tackle the problem of preventing the sender from flooding the receiver with frames faster than the latter is able to process them. This situation can easily happen in practice so being able to prevent it is of great importance. The communication channel is still assumed to be error free, however, and the data traffic is still simplex. One solution is to build the receiver to be powerful enough to process a continuous stream of back-to-back frames. It must have sufficient buffering and processing abilities to run at the line rate and must be able to pass the frames that are

received to the network layer quickly enough. However, this is a worst-case solution. It requires dedicated hardware and can be wasteful of resources if the utilization of the link is mostly low. Moreover, it just shifts the problem of dealing with a sender that is too fast elsewhere; in this case to the network layer. A more general solution to this problem is to have the receiver provide feedback to the sender. After having passed a packet to its network layer, the receiver sends a little dummy frame back to the sender which, in effect, gives the sender permission to transmit the next frame. After having sent a frame, the sender is required by the protocol to bide its time until the little dummy (acknowledgement) frame arrives. This delay is a simple example of a flow control protocol. Protocols in which the sender sends one frame and then waits for an acknowledgement before proceeding are called **stop-and-wait**. **Figure 4.10** gives an example of a simplex stop-and-wait protocol. Although data traffic in this example is simplex, going only from the sender to the receiver, frames do travel in both directions. Consequently, the communication channel between the two data link layers needs to be capable of bidirectional information transfer. However, this protocol entails a strict alternation of flow: first the sender sends a frame, then the receiver sends a frame, then the sender sends another frame, then the receiver sends another one, and so on. A half-duplex physical channel would suffice here. As in protocol 1, the sender starts out by fetching a packet from the network layer, using it to construct a frame, and sending it on its way. But now, unlike in protocol 1, the sender must wait until an acknowledgement frame arrives before looping back and fetching the next packet from the network layer. The sending data link layer need not even inspect the incoming frame as there is only one possibility. The incoming frame is always an acknowledgement. The only difference between receiver1 and receiver2 is that after delivering a packet to the network layer, receiver2 sends an acknowledgement frame back to the sender before entering the wait loop again. Because only the arrival of the frame back at the sender is important, not its contents, the receiver need not put any particular information in it.

```
/* Protocol 2 (Stop-and-wait) also provides for a one-directional flow of data from
   sender to receiver. The communication channel is once again assumed to be error
   free, as in protocol 1. However, this time the receiver has only a finite buffer
   capacity and a finite processing speed, so the protocol must explicitly prevent
   the sender from flooding the receiver with data faster than it can be handled. */

typedef enum {frame_arrival} event_type;
#include "protocol.h"

void sender2(void)
{
  frame s;                          /* buffer for an outbound frame */
  packet buffer;                    /* buffer for an outbound packet */
  event_type event;                 /* frame_arrival is the only possibility */

  while (true) {
      from_network_layer(&buffer);  /* go get something to send */
      s.info = buffer;              /* copy it into s for transmission */
      to_physical_layer(&s);        /* bye-bye little frame */
      wait_for_event(&event);       /* do not proceed until given the go ahead */
  }
}

void receiver2(void)
{
  frame r, s;                       /* buffers for frames */
  event_type event;                 /* frame_arrival is the only possibility */
  while (true) {
      wait_for_event(&event);       /* only possibility is frame_arrival */
      from_physical_layer(&r);      /* go get the inbound frame */
      to_network_layer(&r.info);    /* pass the data to the network layer */
      to_physical_layer(&s);        /* send a dummy frame to awaken sender */
  }
}
```

Fig 4.10: A simplex stop-and-wait protocol

4.7.2 GO BACK N ARQ

Until now we have made the tacit assumption that the transmission time required for a frame to arrive at the receiver plus the transmission time for the acknowledgement to comeback is negligible. Sometimes this assumption is clearly false. In these situations the long round-trip time can have important implications for the efficiency of the bandwidth utilization. As an example, consider a 50-kbps satellite channel with a 500-msec round-trip propagation delay. Let us imagine trying to use protocol 4 to send 1000-bit frames via the satellite. At t=0 the sender starts sending the first frame. At t=20 msec the frame has been completely sent. Not until t=270 msec has the frame fully arrived at the receiver, and not until t=520 msec has the acknowledgement arrived back at the sender, under the best of circumstances. This means that the sender was blocked 500/520 or 96% of the time. In other words, only 4% of the available bandwidth was used. Clearly, the combination of a long transit time, high bandwidth, and short frame length is disastrous in terms of efficiency.

The problem described here can be viewed as a consequence of the rule requiring a sender to wait for an acknowledgement before sending another frame. If we relax that restriction, much better efficiency can be achieved. Basically, the solution lies in allowing the sender to transmit up to ***w*** frames

before blocking, instead of just 1. With a large enough choice of ***w*** the sender will be able to continuously transmit frames since the acknowledgements will arrive for previous frames before the window becomes full, preventing the sender from blocking. To find an appropriate value for ***w*** we need to know how many frames can fit inside the channel as they propagate from sender to receiver. This capacity is determined by the bandwidth in bits/sec multiplied by the one-way transit time, or the bandwidth-delay product of the link. We can divide this quantity by the number of bits in a frame to express it as a number of frames. Call this quantity BD. Then ***w*** should be set to **2*BD*+1**. Twice the bandwidth-delay is the number of frames that can be outstanding if the sender continuously sends frames when the round-trip time to receive an acknowledgement is considered. The ‘‘+1’’ is because an acknowledgement frame will not be sent until after a complete frame is received.

For the example link with a bandwidth of 50 kbps and a one-way transit time of 250 msec, the bandwidth-delay product is 12.5 kbit or 12.5 frames of 1000 bits each. ***2BD+1*** is then 26 frames. Assume the sender begins sending frame 0 as before and sends a new frame every 20 msec. By the time it has finished sending 26 frames, at t=520 msec, the acknowledgement for frame 0 will have just arrived. Thereafter, acknowledgements will arrive every 20 msec, so the sender will always get permission to continue just when it needs it. From then onwards, 25 or 26 unacknowledged frames will always be outstanding. Put in other terms, the sender’s maximum window size is 26. For smaller window sizes, the utilization of the link will be less than 100% since the sender will be blocked sometimes. We can write the utilization as the fraction of time that the sender is not blocked:

link utilization ≤ w / (1+2BD)

This value is an upper-bound because it does not allow for any frame processing time and treats the acknowledgement frame as having zero length, since it is usually short. The equation shows the need for having a large window ***w*** whenever the bandwidth-delay product is large. If the delay is high, the sender will rapidly exhaust its window even for a moderate bandwidth, as in the satellite example. If the bandwidth is high, even for a moderate delay the sender will exhaust its window quickly unless it has a large window (e.g., a 1-Gbps link with 1-msec delay holds 1 megabit). With stop-and-wait for which **w=1**, if there is even one frame’s worth of propagation delay the efficiency will be less than 50%. This technique of keeping multiple frames in flight is an example of pipelining. Pipelining frames over an unreliable communication channel raises some serious issues. First, what happens if a frame in the middle of a long stream is damaged or lost? Large numbers of succeeding frames will arrive at the receiver before the sender even finds out that anything is wrong. When a damaged frame arrives at the receiver, it obviously should be discarded, but what should the receiver do with all the correct frames following it? Remember that the receiving data link layer is obligated to hand packets to the network layer in sequence.

Two basic approaches are available for dealing with errors in the presence of pipelining, both of which are shown in **Fig. 4.11**.

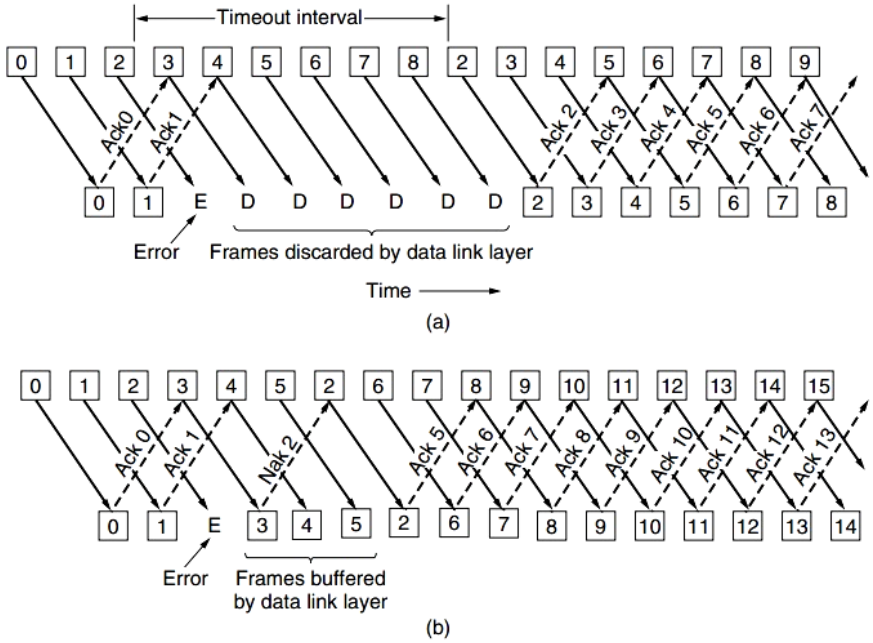


FIG 4.11: Pipelining and error recovery. Effect of an error when (a) receiver's window size is 1 and (b) receiver's window size is large

One option, called **Go-Back-N**, is for the receiver simply to discard all subsequent frames, sending no acknowledgements for the discarded frames. This strategy corresponds to a *receive_window* of size 1. In other words, the data link layer refuses to accept any frame except the next one it must give to the network layer. If the sender's window fills up before the timer runs out, the pipeline will begin to empty. Eventually, the sender will timeout and retransmit all unacknowledged frames in order, starting with the damaged or lost one. This approach can waste a lot of bandwidth if the error rate is high. In **Fig. 4.11 (b)** we see go-back-n for the case in which the receiver's window is large. Frames 0 and 1 are correctly received and acknowledged. Frame 2, however, is damaged or lost. The sender, unaware of this problem, continues to send frames until the timer for frame 2 expires. Then it backs up to frame 2 and starts over with it, sending 2, 3, 4, etc. all over again. The next strategy used is the Selective Repeat protocol and shall be discussed in the next section.

4.7.3 SELECTIVE REPEAT ARQ

The go-back-n protocol works well if errors are rare, but if the line is poor it wastes a lot of bandwidth on retransmitted frames. An alternative strategy, the **selective repeat** protocol, is to allow the receiver to accept and buffer the frames following a damaged or lost one. In this protocol, both sender and receiver maintain a window of outstanding and acceptable sequence numbers, respectively. The sender's window size starts out at 0 and grows to some predefined maximum. The receiver's window, in contrast, is always fixed in size and equal to the predetermined maximum. The

receiver has a buffer reserved for each sequence number within its fixed window. Associated with each buffer is a bit telling whether the buffer is full or empty. Whenever a frame arrives, its sequence number is checked by the function between to see if it falls within the window. If so and if it has not already been received, it is accepted and stored. This action is taken without regard to whether or not the frame contains the next packet expected by the network layer. Of course, it must be kept within the data link layer and not passed to the network layer until all the lower-numbered frames have already been delivered to the network layer in the correct order. Non-sequential receive introduces further constraints on frame sequence numbers compared to protocols in which frames are only accepted in order. We can illustrate the trouble most easily with an example. Suppose that we have a 3-bit sequence number, so that the sender is permitted to transmit up to seven frames before being required to wait for an acknowledgement. The sender now transmits frames 0 through 6. The receiver's window allows it to accept any frame with a sequence number between 0 and 6 inclusive. All seven frames arrive correctly, so the receiver acknowledges them and advances its window to allow receipt of 7, 0, 1, 2, 3, 4, or 5. All seven buffers are marked empty. It is at this point that disaster strikes in the form of a lightning bolt hitting the telephone pole and wiping out all the acknowledgements. The protocol should operate correctly despite this disaster. The sender eventually times out and re-transmits frame 0. When this frame arrives at the receiver, a check is made to see if it falls within the receiver's window. Unfortunately frame 0 is within the new window, so it is accepted as a new frame. The receiver also sends a (piggybacked) acknowledgement for frame 6, since 0 through 6 have been received. The sender is happy to learn that all its transmitted frames did actually arrive correctly, so it advances its window and immediately sends frames 7, 0, 1, 2, 3, 4, and 5. Frame 7 will be accepted by the receiver and its packet will be passed directly to the network layer. Immediately thereafter, the receiving data link layer checks to see if it has a valid frame 0 already, discovers that it does, and passes the old buffered packet to the network layer as if it were a new packet. Consequently, the network layer gets an incorrect packet, and the protocol fails.

The essence of the problem is that after the receiver advanced its window, the new range of valid sequence numbers overlapped the old one. Consequently, the following batch of frames might be either duplicates (if all the acknowledgements were lost) or new ones (if all the acknowledgements were received). The receiver has no way of distinguishing these two cases. The way out of this dilemma lies in making sure that after the receiver has advanced its window there is no overlap with the original window. To ensure that there is no overlap, the maximum window size should be at most half the range of the sequence numbers.

4.7.4 HDLC PROTOCOL

High-Level Data Link Control (HDLC) is a bit-oriented code-transparent synchronous data link layer protocol developed by the

International Organization for Standardization (ISO). HDLC provides both connection-oriented and connectionless service. HDLC can be used for point to multipoint connections, but is now used almost exclusively to connect one device to another, using what is known as **Asynchronous Balanced Mode** (ABM). HDLC frames can be transmitted over synchronous or asynchronous links. Those links have no mechanism to mark the beginning or end of a frame, so the beginning and end of each frame has to be identified. This is done by using a frame delimiter, or flag, which is a unique sequence of bits that is guaranteed not to be seen inside a frame. This sequence is '01111110', or, in hexadecimal notation, 0x7E. Each frame begins and ends with a frame delimiter. A frame delimiter at the end of a frame may also mark the start of the next frame. A sequence of 7 or more consecutive 1-bits within a frame will cause the frame to be aborted.

The contents of an HDLC frame are shown in the following table:

Flag	Address	Control	Information	FCS	Flag
8 bits	8 or more bits	8 or 16 bits	Variable length, 0 or more bits	16 or 32 bits	8 bits

Note that the end flag of one frame may be (but does not have to be) the beginning (start) flag of the next frame. Data is usually sent in multiples of 8 bits, but only some variants require this; others theoretically permit data alignments on other than 8-bit boundaries. The **frame check sequence** (FCS) is a 16-bit CRC-CCITT or a 32-bit CRC-32 computed over the Address, Control, and Information fields. It provides a means by which the receiver can detect errors that may have been induced during the transmission of the frame, such as lost bits, flipped bits, and extraneous bits. However, given that the algorithms used to calculate the FCS are such that the probability of certain types of transmission errors going undetected increases with the length of the data being checked for errors, the FCS can implicitly limit the practical size of the frame. If the receiver's calculation of the FCS does not match that of the sender's, indicating that the frame contains errors, the receiver can either send a negative acknowledge packet to the sender, or send nothing. After either receiving a negative acknowledge packet or timing out waiting for a positive acknowledge packet, the sender can retransmit the failed frame. The FCS was implemented because many early communication links had a relatively high bit error rate, and the FCS could readily be computed by simple, fast circuitry or software. More effective forward error correction schemes are now widely used by other protocols

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.

4.7.5 POINT-TO-POINT PROTOCOL

In networking, the **Point-to-Point Protocol** (PPP) is a data link protocol commonly used in establishing a direct connection between two networking nodes. It can provide connection authentication, transmission encryption (using ECP, RFC 1968), and compression. It is used over many types of physical networks including serial cable, phone line, trunk line, cellular telephone, specialized radio links, and fiber optic links such as SONET. PPP is also used over Internet access connections (now marketed as "broadband"). Internet service providers (ISPs) have used PPP for customer dial-up access to the Internet, since IP packets cannot be transmitted over a modem line on their own, without some data link protocol. Two derivatives of PPP, Point-to-Point Protocol over Ethernet (PPPoE) and Point-to-Point Protocol over ATM (PPPoA), are used most commonly by Internet Service Providers (ISPs) to establish a Digital Subscriber Line (DSL) Internet service connection with customers. PPP is commonly used as a data link layer protocol for connection over synchronous and asynchronous circuits, where it has largely superseded the older Serial Line Internet Protocol (SLIP) and telephone company mandated standards (such as Link Access Protocol, Balanced (LAPB) in the X.25 protocol suite). PPP was designed to work with numerous network layer protocols, including Internet Protocol (IP), TRILL, Novell's Internetwork Packet Exchange (IPX), NBF and AppleTalk. PPP permits multiple network layer protocols to operate on the same communication link. For every network layer protocol used, a separate Network Control Protocol (NCP) is provided in order to encapsulate and negotiate options for the multiple network layer protocols. It negotiates network-layer information, e.g. network address or compression options, after the connection has been established.

Structure of a PPP frame

Name	Number of bytes	Description
Protocol	1 or 2	setting of protocol in data field
Information	variable (0 or more)	datagram
Padding	variable (0 or more)	optional padding

The **Protocol** field indicates the type of payload packet (e.g. LCP, NCP, IP, IPX, AppleTalk, etc.).

The **Information** field contains the PPP payload; it has a variable length with a negotiated maximum called the Maximum Transmission Unit. By default, the maximum is 1500 octets. It might be padded on transmission; if the information for a particular

protocol can be padded, that protocol must allow information to be distinguished from padding.

PPP frames are encapsulated in a lower-layer protocol that provides framing and may provide other functions such as a checksum to detect transmission errors. PPP on serial links is usually encapsulated in a framing similar to HDLC.

Name	Number of bytes	Description
Flag	1	indicates frame's begin or end
Address	1	broadcast address
Control	1	control byte
Protocol	1 or 2	l in information field
Information	variable (0 or more)	datagram
Padding	variable (0 or more)	optional padding
FCS	2 (or 4)	error check

The **Flag** field is present when PPP with HDLC-like framing is used.

The **Address** and **Control** fields always have the value hex FF (for "all stations") and hex 03 (for "unnumbered information"), and can be omitted whenever PPP LCP Address-and-Control-Field-Compression (ACFC) is negotiated.

The **Frame Check Sequence** (FCS) field is used for determining whether an individual frame has an error. It contains a checksum computed over the frame to provide basic protection against errors in transmission. This is a CRC code similar to the one used for other layer two protocol error protection schemes such as the one used in Ethernet. According to RFC 1662, it can be either 16 bits (2 bytes) or 32 bits (4 bytes) in size (default is 16 bits - Polynomial x16 + x12 + x5 + 1). The FCS is calculated over the Address, Control, Protocol, Information and Padding fields after the message has been encapsulated.

4.8 ETHERNET

Ethernet was designed in the 1970s at the Palo Alto Research Center. The first prototype used a coaxial cable as the shared medium and 3 Mbps of bandwidth. Ethernet was improved during the late 1970s and in the 1980s, Digital Equipment, Intel and Xerox published the first official Ethernet specification. This specification defines several important parameters for Ethernet networks. The first decision was to standardize the commercial Ethernet at 10 Mbps. The second decision was the duration of the slot time. In Ethernet, a long slot time enables networks to span a long distance but forces the host to use a larger minimum frame size. The compromise was a slot time of 51.2 microseconds, which corresponds to a minimum frame size of 64 bytes.

Fast Ethernet –

At the same time that switches were becoming popular, the speed of 10-Mbps Ethernet was coming under pressure. At first, 10 Mbps seemed like heaven, just as cable modems seemed like heaven to the users of telephone modems. But the novelty wore off quickly. It seemed that data expanded to fill the bandwidth available for their transmission. Many installations needed more bandwidth and thus had numerous 10-Mbps LANs connected by a maze of repeaters, hubs, and switches, although to the network managers it sometimes felt that they were being held together by bubble gum and chicken wire. But even with Ethernet switches, the maximum bandwidth of a single computer was limited by the cable that connected it to the switch port. It was in this environment that IEEE reconvened the 802.3 committee in 1992 with instructions to come up with a faster LAN. One proposal was to keep 802.3 exactly as it was, but just make it go faster. Another proposal was to redo it totally and give it lots of new features, such as real-time traffic and digitized voice, but just keep the old name. After some wrangling, the committee decided to keep 802.3 the way it was, and just make it go faster. This strategy would get the job done before the technology changed and avoid unforeseen problems with a brand new design. The new design would also be backward-compatible with existing Ethernet LANs. The people behind the losing proposal did what any self-respecting computer-industry people would have done under these circumstances: they stomped off and formed their own committee and standardized their LAN anyway (eventually as 802.12). It flopped miserably. The work was done quickly, and the result, 802.3u, was approved by IEEE in June 1995. Technically, 802.3u is not a new standard, but an addendum to the existing 802.3 standard. This strategy is used a lot. Since practically everyone calls it fast Ethernet, rather than 802.3u, we will do that, too. The basic idea behind fast Ethernet was simple: keep all the old frame formats, interfaces, and procedural rules, but reduce the bit time from 100 nsec to 10 nsec. Technically, it would have been possible to copy 10-Mbps classic Ethernet and still detect collisions on time by just reducing the maximum cable length by a factor of 10. However, the advantages of twisted-pair wiring were so overwhelming that fast Ethernet is based entirely on this design. Thus, all fast Ethernet systems use hubs and switches; multi-drop cables with vampire taps or BNC connectors are not permitted. Nevertheless, some choices still had to be made, the most important being which wire types to support. One contender was Category 3 twisted pair. The argument for it was that practically every office in the Western world had at least four Category 3 twisted pairs running from it to a telephone wiring closet within 100 meters. Sometimes two such cables existed. Thus, using Category 3 twisted pair would make it possible to wire up desktop computers using fast Ethernet without having to rewire the building, an enormous advantage for many organizations. The main disadvantage of a Category 3 twisted pair is its inability to carry 100 Mbps over 100 meters, the maximum computer-to-hub distance specified for 10-Mbps hubs. In contrast, Category 5 twisted pair wiring can handle 100 m easily, and fiber can go much farther. The compromise chosen was to allow all three possibilities, as shown in **Fig. 4.12**, but to pep up the Category 3 solution to give it the additional carrying capacity needed. The Category 3

UTP scheme, called 100 Base-T4, used a signaling speed of 25 MHz, only 25% faster than standard Ethernet's 20 MHz. However, to achieve the necessary bit rate, 100 Base-T4 requires four twisted pairs. Of the four pairs, one is always to the hub, one is always from the hub, and the other two are switchable to the current transmission direction. To get 100 Mbps out of the three twisted pairs in the transmission direction, a fairly involved scheme is used on each twisted pair. It involves sending ternary digits with three different voltage levels. This scheme is not likely to win any prizes for elegance, and we will skip the details. However, since standard telephone wiring for decades has had four twisted pairs per cable, most offices are able to use the existing wiring plant. 100Base-T4 fell by the wayside as many office buildings were rewired with Category 5 UTP for 100 Base-TX Ethernet, which came to dominate the market. This design is simpler because the wires can handle clock rates of 125 MHz. Only two twisted pairs per station are used, one to the hub and one from it.

Name	Cable	Max. segment	Advantages
100Base-T4	Twisted pair	100 m	Uses category 3 UTP
100Base-TX	Twisted pair	100 m	Full duplex at 100 Mbps (Cat 5 UTP)
100Base-FX	Fiber optics	2000 m	Full duplex at 100 Mbps; long runs

Fig 4.12: The original fast Ethernet cabling

Neither straight binary coding (i.e., NRZ) nor Manchester coding is used. Instead, the 4B/5B encoding we described in Sec 2.5 is used. 4 data bits are encoded as 5 signal bits and sent at 125 MHz to provide 100 Mbps. This scheme is simple but has sufficient transitions for synchronization and uses the bandwidth of the wire relatively well. The 100 Base-TX system is full duplex; stations can transmit at 100 Mbps on one twisted pair and receive at 100 Mbps on another twisted pair at the same time. The last option, 100 Base-FX, uses two strands of multimode fiber, one for each direction, so it, too, can run full duplex with 100 Mbps in each direction. In this setup, the distance between a station and the switch can be up to 2 km. Fast Ethernet allows interconnection by either hubs or switches. To ensure that the CSMA/CD algorithm continues to work, the relationship between the minimum frame size and maximum cable length must be maintained as the network speed goes up from 10 Mbps to 100 Mbps. So, either the minimum frame size of 64 bytes must go up or the maximum cable length of 2500 m must come down, proportionally. The easy choice was for the maximum distance between any two stations to come down by a factor of 10, since a hub with 100-m cables falls within this new maximum already. However, 2-km 100 Base-FX cables are too long to permit a 100-Mbps hub with the normal Ethernet collision algorithm. These cables must instead be connected to a switch and operate in a full-duplex mode so that there are no collisions. Users quickly started to deploy fast Ethernet, but they were not about to throw away 10-Mbps Ethernet cards on older computers. As a consequence, virtually all fast Ethernet switches can handle a mix of 10-Mbps and 100-Mbps stations. To make upgrading easy, the standard itself provides a mechanism called auto-negotiation that lets two stations automatically negotiate the

optimum speed (10 or 100 Mbps) and duplexity (half or full). It works well most of the time but is known to lead to duplex mismatch problems when one end of the link auto negotiates but the other end does not and is set to full-duplex mode. Most Ethernet products use this feature to configure themselves.

Gigabit Ethernet –

The ink was barely dry on the fast Ethernet standard when the 802 committee began working on a yet faster Ethernet, quickly dubbed gigabit Ethernet. IEEE ratified the most popular form as 802.3ab in 1999. The committee's goals for gigabit Ethernet were essentially the same as the committee's goals for fast Ethernet: increase performance tenfold while maintaining compatibility with all existing Ethernet standards. In particular, gigabit Ethernet had to offer unacknowledged datagram service with both unicast and broadcast, use the same 48-bit addressing scheme already in use, and maintain the same frame format, including the minimum and maximum frame sizes. The final standard met all these goals. Like fast Ethernet, all configurations of gigabit Ethernet use point-to-point links. In the simplest configuration, illustrated in **Fig 4.13(a)**, two computers are directly connected to each other. The more common case, however, uses a switch or a hub connected to multiple computers and possibly additional switches or hubs, as shown in **Fig 4.13(b)**. In both configurations, each individual Ethernet cable has exactly two devices on it, no more and no fewer. Also like fast Ethernet, gigabit Ethernet supports two different modes of operation: full-duplex mode and half-duplex mode. The "normal" mode is full-duplex mode, which allows traffic in both directions at the same time. This mode is used when there is a central switch connected to computers (or other switches) on the periphery. In this configuration, all lines are buffered so each computer and switch is free to send frames whenever it wants to. The sender does not have to sense the channel to see if anybody else is using it because contention is impossible. On the line between a computer and a switch, the computer is the only possible sender to the switch, and the transmission will succeed even if the switch is currently sending a frame to the computer (because the line is full duplex).

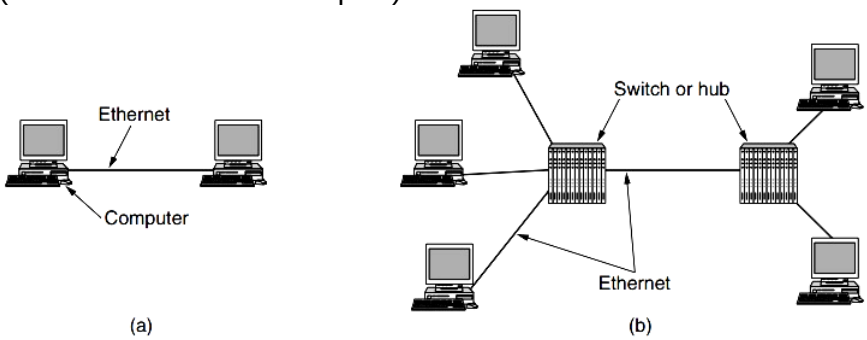


Fig 4.13: (a) A two-station Ethernet. (b) A multistation Ethernet

Since no contention is possible, the CSMA/CD protocol is not used, so the maximum length of the cable is determined by signal strength issues rather than by how long it takes for a noise burst to propagate back to the sender in the worst case. Switches are free

to mix and match speeds. Auto negotiation is supported just as in fast Ethernet, only now the choice is among 10, 100, and 1000 Mbps. The other mode of operation, half-duplex, is used when the computers are connected to a hub rather than a switch. A hub does not buffer incoming frames. Instead, it electrically connects all the lines internally, simulating the multi drop cable used in classic Ethernet. In this mode, collisions are possible, so the standard CSMA/CD protocol is required. Because a 64-byte frame (the shortest allowed) can now be transmitted 100 times faster than in classic Ethernet, the maximum cable length must be 100 times less, or 25 meters, to maintain the essential property that the sender is still transmitting when the noise burst gets back to it, even in the worst case. With a 2500-meter-long cable, the sender of a 64-byte frame at 1 Gbps would be long finished before the frame got even a tenth of the way to the other end, let alone to the end and back. This length restriction was painful enough that two features were added to the standard to increase the maximum cable length to 200 meters, which is probably enough for most offices. The first feature, called carrier extension, essentially tells the hardware to add its own padding after the normal frame to extend the frame to 512 bytes. Since this padding is added by the sending hardware and removed by the receiving hardware, the software is unaware of it, meaning that no changes are needed to existing software. The downside is that using 512 bytes worth of bandwidth to transmit 46 bytes of user data (the payload of a 64-byte frame) has a line efficiency of only 9%. The second feature, called frame bursting, allows a sender to transmit a concatenated sequence of multiple frames in a single transmission. If the total burst is less than 512 bytes, the hardware pads it again. If enough frames are waiting for transmission, this scheme is very efficient and preferred over carrier extension. In all fairness, it is hard to imagine an organization buying modern computers with gigabit Ethernet cards and then connecting them with an old-fashioned hub to simulate classic Ethernet with all its collisions. Gigabit Ethernet interfaces and switches used to be expensive, but their prices fell rapidly as sales volumes picked up. Still, backward compatibility is sacred in the computer industry, so the committee was required to put it in. Today, most computers ship with an Ethernet interface that is capable of 10-, 100-, and 1000-Mbps operation and compatible with all of them. Gigabit Ethernet supports both copper and fiber cabling, as listed in **Fig. 4.14**. Signaling at or near 1 Gbps requires encoding and sending a bit every nanosecond. This trick was initially accomplished with short, shielded copper cables (the 1000 Base-CX version) and optical fibers. For the optical fibers, two wavelengths are permitted and result in two different versions: 0.85 microns (short, for1000Base-SX) and 1.3microns (long, for 1000Base-LX).

Name	Cable	Max. segment	Advantages
1000Base-SX	Fiber optics	550 m	Multimode fiber (50, 62.5 microns)
1000Base-LX	Fiber optics	5000 m	Single (10 μ) or multimode (50, 62.5 μ)
1000Base-CX	2 Pairs of STP	25 m	Shielded twisted pair
1000Base-T	4 Pairs of UTP	100 m	Standard category 5 UTP

Fig 4.14: Gigabit Ethernet cabling

Signaling at the short wavelength can be achieved with cheaper LEDs. It is used with multimode fiber and is useful for connections within a building, as it can run up to 500 m for 50-micron fiber. Signaling at the long wavelength requires more expensive lasers. On the other hand, when combined with single-mode (10-micron) fiber, the cable length can be up to 5 km. This limit allows long distance connections between buildings, such as for a campus backbone, as a dedicated point-to-point link. Later variations of the standard allowed even longer links over single-mode fiber. To send bits over these versions of gigabit Ethernet, the 8B/10B encoding was borrowed from another networking technology called Fibre Channel. That scheme encodes 8 bits of data into 10-bit codewords that are sent over the wire or fiber, hence the name 8B/10B. The codewords were chosen so that they could be balanced (i.e., have the same number of 0s and 1s) with sufficient transitions for clock recovery. Sending the coded bits with NRZ requires a signaling bandwidth of 25% more than that required for the uncoded bits, a big improvement over the 100% expansion of Manchester coding. However, all of these options required new copper or fiber cables to support the faster signaling. None of them made use of the large amount of Category 5 UTP that had been installed along with fast Ethernet. Within a year, 1000 Base-T came along to fill this gap, and it has been the most popular form of gigabit Ethernet ever since. People apparently dislike rewiring their buildings. More complicated signaling is needed to make Ethernet run at 1000 Mbps over Category 5 wires. To start, all four twisted pairs in the cable are used, and each pair is used in both directions at the same time by using digital signal processing to separate signals. Over each wire, five voltage levels that carry 2 bits are used for signaling at 125 M symbols/sec. The mapping to produce the symbols from the bits is not straightforward. It involves scrambling, for transitions, followed by an error correcting code in which four values are embedded into five signal levels. A speed of 1 Gbps is quite fast. For example, if a receiver is busy with some other task for even 1 msec and does not empty the input buffer on some line, up to 1953 frames may have accumulated in that gap. Also, when a computer on a gigabit Ethernet is shipping data down the line to a computer on a classic Ethernet, buffer overruns are very likely. As a consequence of these two observations, gigabit Ethernet supports flow control. The mechanism consists of one end sending a special control frame to the other end telling it to pause for some period of time. These PAUSE control frames are normal Ethernet frames containing a type of 0x8808. Pauses are given in units of the minimum frame time. For gigabit Ethernet, the time unit is 512 nsec, allowing for pauses as long as 33.6 msec. There is one more extension that was introduced along with gigabit Ethernet. Jumbo frames allow for frames to be longer than 1500 bytes, usually up to 9 KB. This extension is proprietary. It is not recognized by the standard because if it is used then Ethernet is no longer compatible with earlier versions, but most vendors support it anyway. The rationale is that 1500 bytes is a short unit at gigabit speeds. By manipulating larger blocks of information, the frame rate can be decreased, along with the processing associated with it, such as interrupting the processor to say that a frame has

arrived, or splitting up and recombining messages that were too long to fit in one Ethernet frame.

CHECK YOUR PROGRESS

2. Fill in the blanks:

(a) The physical layer process and some of the data link layer process run on dedicate hardware called a _____.

(b) Protocols in which the sender sends one frame and then waits for an acknowledgement before proceeding are called _____.

(c) In the _____ protocol the receiver simply discards all subsequent frames, sending no acknowledgements for the discarded frames.

(d) In the Selective Repeat protocol both _____ and _____ maintain a window of outstanding and acceptable sequence numbers, respectively

(e) _____ frames can be transmitted over synchronous or asynchronous links.

4.9 LET US SUM UP

- In the seven-layer OSI model of computer networking, the ***Data Link*** layer is layer 2.
- The ***byte count framing*** method uses a field in the header to specify the number of bytes in the frame.
- ***Error control*** makes sure all frames are eventually delivered to the network layer at the destination and in the proper order.
- In ***feedback-based flow control***, the receiver sends back information to the sender giving it permission to send more data.
- The number of bit positions in which two code words differ is called the ***Hamming distance***.

- ***Convolutional codes*** are specified in terms of their rate and constraint length.

- ***Reed-Solomon*** codes are widely used in practice because of their strong error-correction properties, particularly for burst errors.

- ***Selective Repeat*** protocol, is to allow the receiver to accept and buffer the frames following a damaged or lost one.

- ***HDLC*** provides both connection-oriented and connectionless service.

- ***Point-to-Point Protocol*** (PPP) is a data link protocol commonly used in establishing a direct connection between two networking nodes.

4.10 ANSWERS TO CHECK YOUR PROGRESS

1.
(a) data link
(b) Rate-based
(c) Error-correcting
(d) Error-detecting
(e) Hamming distance
(f) convolutional code
(g) Reed-Solomon codes
(h) LDPC codes
(i) checksum
(j) Polynomial codes.

2.
(a) Network Interface Card
(b) stop-and-wait
(c) Go-Back-N
(d) sender, receiver
(e) HDLC.

4.11 FURTHER READINGS

Computer Networks

– Andrew S. Tanenbaum, David J. Wetherall

PRENTICE HALL

Computer Networking: Principles, Protocols and Practice

– Olivier Bonaventure

4.12 MODEL QUESTIONS

1. What is the Data Link layer of the OSI model? Describe its functions.
2. What is Framing in the Data link layer? Describe the different framing methods used.
3. Describe the concept of Error control in the data-link layer.
4. What is Flow control in the data-link layer? What are the available Flow control techniques?
5. What do you understand by Error detection and Error correction in the Data link layer?
6. Illustrate with examples three Error correcting codes present in the Data link layer.
7. What are Error detecting codes in the Data link layer? Describe.
8. Illustrate the Stop-and-Wait protocol of the Data link layer.
9. Describe the Go-Back-N protocol of the Data link layer.
10. Describe the working of the Selective Repeat protocol of the Data link layer.
11. Write a short note on the HDLC protocol of the Data link layer.
12. Explain the Point-to-Point protocol of the Data link layer.
13. Write a note on Ethernet and its other types.

UNIT - 5 NETWORK LAYER

UNIT STRUCTURE

5.1 LEARNING OBJECTIVES

After going through this unit, you will be able to:

- learn about the basic concept of network layer
- learn about internetworks and network addressing
- describe the concept of routing and different routing protocols
- learn about different network layer protocols.

5.2 INTRODUCTION TO NETWORK LAYER

In this unit we will discuss about the network layer of OSI model. The network layer provides the mechanism of transferring variable length data sequences from a source host on one network to a destination host on a different network .In this process, the quality of service requested by the transport layer is also maintained. The network layer adds a header that includes the logical addresses of the sender and receiver to the packet coming from the upper layer.

The network layer performs network routing functions. Routers are used in this layer for sending data throughout the different networks.

The network layer must know about the topology of the communication subnet and choose appropriate paths through it to perform its job properly. It must carefully choose routes to avoid overloading some of the communication lines and routers. When the source and destination are in different networks then different problems are occurred. It is up to the network layer to deal with these problems.

5.3 INTERNETWORKS

A collection of interconnected networks, permitting data to move freely among these large number of different networks and populations is called an internetwork or internet. So people connected to one network can communicate with people attached to a different network with the help of internetwork. The gateways are used to make the connection amongst different networks and provide the necessary translation for hardware and software. In other words internet is a collection of LANs connected by a WAN. Different organizations construct different parts of the internetwork and each organization maintains its own part.

The internetwork is a packet-switched network at the network layer. The internetwork uses universal addresses defined in the network layer to route packets from the source to the destination. Delivery of a packet can be accomplished by using either a connection-oriented or a connectionless network service. In connectionless service, the network layer protocol treats each packet independently. The packets in a message may or may not travel the same path to their destination. This type of service is used in the datagram approach to packet switching. The Internet has chosen this type of service at the network layer because the internet is made of different types of networks that it is impossible to create a connection from the source to the destination without knowing the nature of the networks in advance.

5.4 ADDRESSING

In network layer, the packet transmitted by the sending computer may travel through different LANs or WANs in the way to the destination computer. Now in this part of communication, a global addressing scheme is used which is called the logical addressing. The term IP address is used to mean a logical address in the network layer of the TCP/IP protocol suite.

The internet addresses used in current times are 32 bits in length which are referred as IPv4 (IP version 4) addresses or simply IP addresses. So in case of IPv4 address, the maximum 2^{32} addresses are possible.
Now a new version of internet address is designed due to the requirement of more than 2^{32} addresses. This new version is referred as IPv6 (IP version 6). In this version, the internet uses 128-bit addresses which are called IPv6 addresses. So in case of IPv6 address, the maximum 2^{128} addresses are possible.

IPv4 Addresses:

An IPv4 address is a 32 bit address which uniquely and globally defines the connection of a device to the internet. So, two devices on the internet can never have the same IPv4 address at the same time and this addressing scheme must be accepted by any host which wants to be connected to the internet.

The length of an IPv4 address is 32-bit. Each bit of an IPv4 address can have two different values which are 0 or 1. So the address space of IPv4 is 2^{32} or 4,294,967,296.

To show an IPv4 address, two types of notations can be used which are binary notation and dotted-decimal notation.

In binary notation, the IPv4 address is displayed as 32 bits. For example:

00111000 10110011 00111011 00001100

In dotted-decimal notation, IPv4 addresses are written in decimal form with a decimal point separating the bytes. For example the dotted-decimal notation of the above address is given below:

56.179.59.12

In classful addressing, there are five classes of IPv4 addresses which are A, B, C, D and E. Each class occupies some part of the address space. If the address is given in binary notation, the class of the address can be found with the leftmost first few bits as follows:

- If the leftmost bit of the address is 0 then it is in class A. For example: 00000100 10001001 00010010 11101101 is class A address.
- If the leftmost two bits of the address are 10 then it is in class B. For example: 10100001 00010010 11100010 10111001 is a class B address.
- If the leftmost three bits of the address are 110 then it is in Class C. For example: 11000001 10001001 00001010 11111010 is a class C address.

- If the leftmost four bits of the address are 1110 then it is in class D. For example: 11101010 00010101 10111010 11111111 is a class D address.
- If the leftmost four bits of the address are 1111 then it is in class E. For example: 11111001 00010110 11101101 00011101 is a class E address.

Now, if the address is given in decimal-dotted notation, the decimal value of the first byte defines the class as follows:

- If the decimal value of the first byte of the address is in the range 0 to 127 then it is in class A. For example: 4.137.18.237 is a class A address.
- If the decimal value of the first byte of the address is in the range 128 to 191 then it is in class B. For example: 161.18.226.185 is a class B address.
- If the decimal value of the first byte of the address is in the range 192 to 223 then it is in class C. For example: 193.137.20.250 is a class C address.
- If the decimal value of the first byte of the address is in the range 224 to 239 then it is in class D. For example: 234.21.186.255 is a class D address.
- If the decimal value of the first byte of the address is in the range 240 to 255 then it is in class E. For example: 249.22.237.29 is a class E address.

Class A addresses were designed for large organizations with large number of attached hosts or routers. Class B addresses were designed for midsize organizations with tens of thousands attached hosts or routers. Class C addresses were designed for small organizations with a small number of attached hosts or routers. Class D addresses were designed for multicasting. The class E addresses were reserved for future use.

In clasful addressing, an IP address in class A, B or C divided into netid and hostid. In class A, one byte defines the netid and three bytes define the hostid. In class B, two bytes define the netid and two bytes define the hostid. In class C, three bytes define the netid and one byte defines the hostid.

In classful addressing, mask or default mask are used for classes A, B and C. A mask is a 32-bit number in which the n leftmost bits are 1s and the 32-n rightmost bits are 0s. Class D and class E does not have any default mask. The masks for classes A, B, and C are given in the following table.

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

In the given table, the last column shows the mask in the form /n where n can be 8, 16 or 24 in classful addressing. This notation is also called slash notation or Classless Interdomain Routing (CIDR) notation. This notation is used in classless addressing.

Due to the fast growing internet, depletion of the available addresses may occur in case of the classful addressing scheme. In recent times, the number of devices on the Internet is less than the 2^{32} address space but the availability of class A and class B addresses is decreasing and a class C block is too small for most midsize organizations.

To solve the drawback of classful addressing scheme, classless addressing was designed and implemented. This addressing scheme overcome address depletion and gives more organization access to the internet. In this scheme there are no classes.

In classless addressing, when a device needs to be connected to the Internet then it is granted a block of addresses. The size of the block varies based on the nature and size of the device.
The internet authorities impose three restrictions on classless address blocks to simplify the handling of addresses which are:

1. The addresses in a block must be contiguous.
2. The number of addresses in a block must be a power of 2 like 2^1, 2^2, 2^4 etc.
3. The first address must be evenly divisible by the number of addresses.

In classless addressing the mask for a block can take any value from 0 to 32.

In IPv4 addressing, a block of addresses can be defined as x.y.z.t/n where x.y.z.t defines one of the addresses and the /n defines the mask.

Now the first address in the block can be found by setting the 32-n rightmost bits in the binary notation of the address to 0s and the Last address in the block can be found by setting the 32-n rightmost bits in the binary notation of the address to 1s.The number of addresses in the block is the difference between the last and first address. It can be found by using the formula 2^{32-n}.

Network Address Translation (NAT)

In near future, the address space of IPv4 will not be sufficient to accommodate all the devices in the internet because of the fast growth of internet. Now a solution to this problem is Network Address Translation (NAT). It is described in RFC 3022.

The basic concept of NAT is to assign each company a large set of addresses internally and one address or a small set of addresses externally. Each computer inside the company gets a unique IP address from the large set of addresses. When a data packet comes out from the company and goes to the ISP then an address translation takes place. In this scheme, three ranges of IP addresses have declared as private. These addresses can be used internally by different companies but packets containing these addresses can not appear on the internet itself. These three reserved ranges of addresses are given as follows:

```
10.0.0.0     to  10.255.255.255    (16,777,216 hosts)
172.16.0.0   to  172.31.255.255    (1,048,576 hosts)
192.168.0.0  to  192.168.255.255   (65,536 hosts)
```

The first range provides for 16,777,216 addresses .In general most companies choose this range of addresses. So using first range of addresses, every computer inside the company has a unique address of the form 10.x.y.z. When a packet comes out from the company network, it passes through a NAT box which converts the internal IP source address to the company's true IP address.

IPv6 Addresses:

Internet Protocol version 6 (**IPv6**) is the latest version of internet addresses. IPv6 was developed by the Internet Engineering Task Force (IETF).

IPv6 has an address space of 2^{128} addresses because it uses a 128-bit address. So the address space of IPv6 is much larger than IPv4 address space.

Hexadecimal Colon Notation: IPv6 specifies hexadecimal colon notation for its addresses where 128 bits is divided into eight sections and each 2 bytes in length. Two bytes in hexadecimal notation requires four hexadecimal digits. So an IPv6 address consists of 32 hexadecimal digits with every four digits separated by a colon. For example: EFED: 07E4: 0070: 0040: 00E0: D0FF: 0000: EEEE is an IPv6 address in hexadecimal colon notation.

In case of IPv6, the IP addresses are divided into several categories. A few leftmost bits in each IP address which are called the type prefix define the category of each address. The type prefix is designed such that no code is identical to the first part of any other code. So when an address is given then the type prefix can easily be determined. The following table shows the type prefix and its type of the address

Type Prefix	Type
0000 0000	Reserved
0000 0001	Unassigned
0000 001	ISO network addresses
0000 010	IPX(Novell) network addresses
0000 011	Unassigned
0000 1	Unassigned
0001	Reserved
001	Reserved
010	Provider-based unicast addresses
011	Unassigned
100	Geographic-based unicast addresses
101	Unassigned
110	Unassigned
1110	Unassigned
1111 0	Unassigned
1111 10	Unassigned
1111 110	Unassigned

1111 1110 0	Unassigned
1111 1110 10	Link local addresses
1111 1110 11	Site local addresses
1111 1111	Multicast addresses

The different categories of IPv6 addresses are discussed as follows:

Unicast Addresses:

A unicast address is used to define a single computer. The packet sent to a unicast address must be delivered to a specific computer. IPv6 defines two types of unicast addresses which are geographically based and provider based.

Multicast addresses:

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

Anycast Addresses:

IPv6 defines anycast addresses where each anycast address defines a group of nodes. Now a packet sent to an anycast address is delivered to only one of the members of the anycast group which is the nearest one with the shortest route. Anycast addresses can be assigned to all routers of an ISP that covers a large logical area in the Internet. No block is assigned for anycast addresses.

Reserved Addresses:

Reserved addresses start with eight 0s. There are some subcategories in this category which are unspecified address, loopback address, compatible address and mapped address.

In the unspecified address, all 128 bits are 0s. This address is used when a host does not know its own address and sends an inquiry to find its address.

In the loopback address, only the right most bit is 1 and all other 127 bits are 0s. This address is used by a host to test itself without going into the network.

In a compatible address, the right most 32 bits is an IPv4 address and all other 96 bits are 0s. These addresses are used when a host using IPv6 wants to send message to another host using IPv6 but the message needs to travel through a part of the network that is still using IPv4.

In a mapped address, all leftmost 80 bits are 0s and next 16 bits to these 80 bits are 1s. Here the rightmost 32 bits is an IPv4 address. These addresses are used when a host that has migrated to IPv6 wants to send a message to a host still using IPv4.

Local Addresses:

Local addresses are used when an organization wants to use IPv6 protocol without being connected to the global Internet. So in this case, no messages can be sent from outside the organization to the nodes using these addresses. There are two types of local addresses in IPv6 which are link local address and site local address.

A link local address is used in an isolated subnet and a site local address is used in an isolated site with several subnets.

5.5 ROUTING

The main function of the network layer is to transfer variable length data sequence from source machine to destination machine through the best possible path which is also called routing IP packets from the source machine to the destination machine. Now the algorithms called routing algorithms which are the part of network layer software responsible for deciding the routes and the data structures to transmit the incoming packets. If the subnet uses datagrams internally, this decision must be a new one for each time at every arrival of data packet because the best route may have changed since last time. If the subnet uses virtual circuits internally then routing decisions are made only when a new virtual circuit is being set up. This is also called as session routing because a route remains same for an entire user session.

The different goals of a routing algorithm are discussed as follows:

1. **Correctness:** The routing should be done in a proper way so that correctness can be maintained for sending the packets to their proper destination.
2. **Simplicity:** Simplicity should be maintained in the development of a routing algorithm so that the overhead is as low as possible.
3. **Robustness:** Once a major network becomes operative, it may be expected to run continuously for years without any failures. So the routing algorithms should be robust enough to handle hardware and software failures and should be able to cope with changes in the topology and traffic without requiring all jobs in all hosts to be aborted and the network rebooted every time when some router goes down.
4. **Stability:** The routing algorithms should be stable under all possible situations.
5. **Fairness:** Every node connected to the network should get a fair chance of transmitting their packets.
6. **Optimality:** The routing algorithms should be optimal in case of throughput and minimizing mean packet delays.

Routing algorithms can be divided into two major classes which are nonadaptive and adaptive algorithms.

Nonadaptive algorithms are static routing algorithms. Here the choice of the route to transmit IP packets from one node to another node is computed in advance and downloaded to the routers when the network is booted. It is not depended on the measurements of the current traffic and topology.

On the other hand, in adaptive algorithms, the routing decisions are changed whenever there is a change in the topology and the network traffic. Now the different adaptive algorithms have differences amongst them in case of the following points:

(a) From where they get their information.

(b) When they change the routes

(c) When the load changes or when the topology changes.

(d) Type of metric used for optimization.

The Optimality Principle

The optimality principle is used in routing algorithms. It states that if router B is on the optimal path from router A to router C then the optimal path from B to C also falls along the same route.

So according to the optimality principle, the set of optimal routes from all sources to a particular destination form a tree rooted at that destination called a sink tree. A sink tree is not necessarily unique because other trees with the same path lengths may exist. The goal of all routing algorithms is to discover and use the sink trees for all routers.

Routing Protocols

Routing protocols are used to continuously update the routing tables that are consulted for forwarding and routing IP packets. A routing protocol is a combination of rules and procedures that allows routers to share whatever they know about the internet or their neighbourhood. Routing protocols are divided into categories which are unicast and multicast protocols.

5.5.1 UNICAST ROUTING

A host or a router has a table called routing table which stores one entry for each destination or a group of destinations host to route IP packets. A routing table can be either static or dynamic.

A static table is one with manual entries which are entered by the network administrator. It cannot be updated automatically when there is a change in the Internet. It is the responsibility of the administrator to update the table manually.

On the other hand, a dynamic routing table is updated automatically when there is a change in the internet. In recent times, dynamic routing tables are required in the internet for better performance.

In unicast routing, when a router receives a packet to route then it needs to find the shortest path to the destination of the packet. The router uses its routing table to find the shortest path for that particular destination. Now the next-hop entry corresponding to the destination in the routing table is the start of the shortest path. The router has a shortest path tree to optimally reach all destinations. In unicast routing, each router needs only one shortest path tree to forward a packet.

Distance Vector Routing

Distance vector routing is a dynamic routing algorithm. This algorithm is also referred as the distributed Bellman-Ford routing algorithm or the Ford-Fulkerson algorithm. It was developed by Bellman in 1957 and Ford and Fulkerson in 1962.

In this algorithm, each router maintains a routing table which provide the best known distance to each destination and which path to use to reach there. Each entry in a routing table of a router contains two parts which are (a) the preferred outgoing line to use for the destination and (b) an estimate of the time or distance to the destination. So here, with the help of the routing tables, the route source host to destination host with minimum distance is calculated. Now the routing tables of the routers are automatically updated by exchanging information with the neighbours. Initially each router can know only the distance between itself and its immediate neighbours. Later, each router shares its routing table with its immediate neighbours periodically and when there is a change. By doing this, the routing tables are updated automatically and path between any two routers with minimum distance can be found with the help of updated routing tables.

The Count-to-Infinity Problem

The count to infinity is a problem that may occur in distance vector routing. The count-to-infinity problem happens when a router tells another router that it has a path somewhere but there is no way for the second router to know that he is a part of the path. The count to infinity problem is caused by link failures that partition the network into two or more segments.

Link State Routing

Link state routing is also a dynamic routing algorithm. In this algorithm, each router must perform the following five steps:

1. Each router must discover its neighbours and learn their network addresses. It is done by sending a special HELLO packet from a router to all its neighbours connected to it. Now the router on the other end need to send back a reply telling who it is. These names must be globally unique.
2. Each router must have a reasonable estimate of the delay or cost to each of its neighbours. It can be done by sending a special ECHO packet to the neighbours of a router. Now on receiving an ECHO packet by a router, it is required to send back immediately to the sending router. By measuring the round-trip time and dividing it by two, the sending router can get a reasonable estimate of the delay. For better results, this test can be done several times and the average is used. In this method, the delays are assumed to be symmetric. Now here the load of the network traffic plays an important role. If the load is considered as a factor in this method then the round-trip timer must be started when the ECHO packet is

queued and if the load is ignored then the timer should be started when the ECHO packet reaches the front of the queue.

3. After collecting the necessary information required to exchange, each router must construct a packet called link state packet, containing all these information. This packet contains the identity of the sender, a sequence number, age, a list of neighbours and the delay to each neighbour. Now the most important point is when these packets should be constructed. In some cases, the link state packets are built at regular intervals. Another possibility is to build these packets when some significant event occurs like a neighbour going down or coming back up again or changing its properties.

4. Each router must distribute its link state packet to all other routers. The routers getting the first packet will change their routes. Now the different routers may be using different versions of the topology so some problems like inconsistencies, loops, unreachable machines may occur. So in this distribution process a distribution algorithm is used to distributing the link state packets reliably. The basic idea is to use flooding to distribute the packets. Here each packet contains a sequence number that is incremented for each new packet sent. When a new packet comes in, it is checked against the list of packets already arrived. If it is new one then it is forwarded on all lines except the one it comes in. Now if it is a duplicate packet then it is discarded. If an incoming packet has a sequence number which is lower than the highest sequence of any packet arrived earlier then it is rejected because the router now has more recent data. Now this algorithm has some problems. First problem is it may be possible that at some time the value of the sequence number will reach its maximum value. The solution for this problem is to use a 32 bit sequence number which will be large enough. Second problem is if a router crashes then it will lose track of its sequence number. If it starts again at 0 then the next packet will be rejected as a duplicate one. Third problem is if a sequence number is corrupted. In this case it may be possible that some new packets will be rejected. For example if the actual sequence number of a new packet is 8 but it is corrupted to 640 then packets with sequence numbers from 9 to 640 will be rejected as obsolete because the current sequence number is thought to be 640. The solution to these problems is to include the age of each packet after the

sequence number and decrement it once per second. When the age hits zero then the information from that router is discarded. The Age field is also decremented by each router during the initial flooding process to make sure no packet can get lost and live for an indefinite period of time. Some modifications can be done to this algorithm. First modification is when a packet comes in to a router for flooding, it is not queued for transmission immediately. In this case, it is first put in a holding area to wait a short while. If another packet from the same source comes in before the first packet is transmitted, their sequence numbers are compared. If they are equal then the duplicate is discarded and if they are different then the older one is flooded. Second modification is all link state packets are acknowledged. It will help to handle errors occurred on the router to router lines. In this case when a line goes idle, the holding area is scanned in round-robin order to select a packet for acknowledgement to send.

5. Each router must compute the shortest path to every other router. In this step, Dijkstra's algorithm can be run locally to construct the shortest path to all possible destinations. The results of this algorithm can be installed in the routing tables. Now in case of this algorithm, for large subnets, the requirement of memory to store input data can be a problem because if a subnet has n routers and each router has k neighbours then the memory required to store the input data is proportional to k*n. The computation time can also be a problem in this algorithm.

5.5.2 UNICAST ROUTING PROTOCOLS

Two unicast routing protocols are discussed as follows:

Routing Information Protocol

The Routing Information Protocol (RIP) is an intradomain routing protocol used inside an autonomous system based on distance vector routing. RIP implements distance vector routing with some consideration:

1. An autonomous system has both routers and networks. Here the routers have routing tables but networks do not have any table.
2. The first column in routing tables defines a network address.
3. The metric used by RIP is the distance which is the number of networks to reach the destination and it is called hop count.
4. Any route in an autonomous system using RIP cannot have more than 15 hops.

5. The next-node column in a routing defines the address of the router to which the packet is to be sent to reach its destination.

Open Shortest Path First Protocol (OSPF):

The open shortest path protocol is an intradomain unicast routing protocol based on link state routing.

The OSPF divides an autonomous system into some areas to handle routing efficiently and in a timely manner. An area is a collection of networks, hosts, and routers of an autonomous system. All networks inside an area must be connected. Routers inside an area required to distribute the routing information in the area. There are some special routers at the border of an area called area border routers which summarize the information about the area and pass it to other areas. There is a special area inside an autonomous system which is called the backbone and the routers inside the backbone are called the backbone routers. All other areas inside an autonomous system must be connected to the backbone.

When for some problem the connectivity between a backbone and an area is broken then administrator must create a virtual link between routers to allow continuity of the functions of the backbone as the primary area.

The OSPF protocol allows the administrator to assign a metric to each route. The metric is actually a cost which is based on a type of service such as minimum delay, maximum throughput etc. So a router can have multiple routing tables each based on a different type of service in OSPF protocol.

In OSPF protocol four types of links have been specified which are point to point, transient, stub and virtual.

A point to point link connects two routers without any other host or router in between.

A transient link is a network with several routers attached to it. The data can enter through any of the routers and exit the network through any router.

A stub link is a network that is connected to only one router. The data packets enter the network through this single router and exit the network through this same router.

When the link between two routers is broken, the administrator may create a virtual link between them using a longer path which possibly passes through several routers.

5.5.3 MULTICAST ROUTING

In multicast routing, router receives multicast packet for routing to the destinations in more than one network. Now in this situation routing of a single multicast packet to each member of a group requires a shortest path tree. So if we there are n groups then n shortest path trees are required. In this case, the complexity of multicast routing is increased. Now to solve the problem, source-based trees and group-shared trees approaches are used.

Source-Based Tree: In the source-based tree approach, each router needs to have one shortest path tree for each group.

Group Shared Tree: In the group-shared tree approach, only one designated router, called the center core has m shortest path trees in its routing table. Here if a router receives a multicast packet then it encapsulates the packet in a unicast packet and

sends it to the core router. The core router removes the multicast packet from the unicast packet and route the multicast packet with the help of its routing table.

Multicast Link State Routing:

Multicast link state routing is an extension of unicast link state routing. This multicast routing algorithm uses a source based tree approach. In unicast link state routing, each node needs to advertise the state of its links. Now in multicast routing, state of a link specifies the groups which are active on that link. Here a node advertises every group which has any member on the link. Now the information about the group can be received by running Internet Group Management Protocol (IGMP).

When a router receives all the link state packets then it constructs n topologies from which *n* shortest path trees are made by using Dijkstra's algorithm. Here n is the number of groups. So each router has a routing table that specifies n shortest path trees.

Now the problem with this routing algorithm is that it requires more time and space to create and save the many shortest path trees than the unicast routing. The solution for this problem is that creation of the trees should be done only when it is required. When a router receives a packet with a multicast destination address then it use Dijkstra's algorithm to calculate the shortest path tree for that group. The result can be cached in case there are additional packets for that destination.

Multicast Distance Vector Routing:

Multicast distance vector routing is the extension of unicast distance vector routing. In case of multicast routing, routers are not allowed to send its routing table to its neighbours. Here the idea is to create a table by using the information from the unicast distance vector tables.

Multicast distance vector routing uses source based trees but here the router never actually constructs a routing table. In this algorithm when a router receives a multicast packet then it forwards the packet as though it is consulting a routing table. After a packet is forwarded the table is destroyed.

The multicast distance vector algorithm uses a process to forward packets based on four decision making strategies discussed as follows:

1. **Flooding:** Flooding is a static algorithm. In this strategy, every incoming IP packet is passed on every outgoing line except the one it arrived on. Flooding generates vast numbers of duplicate packets. One solution to solve this problem is to have a hop counter contained in the header of each packet which is decremented at each hop. When the counter reaches zero then the packet is discarded. In general, the hop counter should be initialized to the length of the path from source to destination but if the sender does not know the length then it can initialize the counter to the maximum length of any source to any destination.

An another technique can be used to keep track of which packets have been flooded and avoid sending them second time. In this technique the source router put a sequence number in each packet it receives from its hosts. Here each router requires a list of sequence numbers per source router originating at that source which are already seen. If the sequence number of an incoming packet is on the list then it is discarded.

2. **Reverse Path Forwarding (RPF):** In RPF, a router forwards only the copy of a packet that has travelled the shortest path from the source to the router to prevent loops. Now RPF uses the unicast routing table to find this copy. The other copies of the packet are discarded.
3. **Reverse Path Broadcasting (RPB):** In RPF scheme, a network may receive two or more copies of a packet because here forwarding is based on the source address. Now to eliminate this duplication, only one parent router for each network should be specified. So for this, a restriction can be made by which a network can receive a multicast packet from a particular source only through a designated parent router. This scheme is called reverse path broadcasting (RPB). Now the designated parent router can be the router with the shortest path to the source.
4. **Reverse Path Multicasting (RPM):** In RPB, broadcast of packets are done which is not efficient. So to increase efficiency, in RPM scheme the multicast packet is forwarded to only those networks that have active members for the particular group.

5.5.4 MULTICAST ROUTING PROTOCOLS

Two multicast routing protocols are discussed as follows:

Multicast Open Shortest Path First (MOSPF) Protocol:

MOSPF protocol is an extension of the OSPF protocol. It uses multicast link state routing to create source-based trees. In this protocol, a tree is constructed that contains all the hosts belonging to a particular group. In this construction, the unicast address of the host is used. For efficiency, the router calculates the shortest path trees and the tree can be saved in cache memory for future use by the same source and group pair. MOSPF is a data-driven protocol. So an MOSPF router sees a datagram with a given source and group address for the first time, it constructs the Dijkstra shortest path tree.

Distance Vector Multicast Routing Protocol (DVMRP):

DVMRP is a multicast routing protocol which uses multicast distance vector routing. It is a source-based routing protocol based on RIP.

5.6 NETWORK LAYER PROTOCOLS

Four network layer protocols are discussed as follows:

5.6.1 INTERNET PROTOCOL

The main network protocol in the internet model is the Internet Protocol (IP). The Internet Protocol version 4(IPv4) is used by the TCP/IP protocol.

IPv4 is an unreliable and connectionless protocol for a packet-switching network that uses the datagram approach. Here each datagram is handled independently and each datagram can follow a different route from source to the destination. So in this case, datagrams sent by the same source to the same destination may not arrive in the same order as the sending order. Some datagrams may also be lost or corrupted during transmission. The IPv4 does not provide any error control and flow control mechanism. It provides error detection on the header. IPv4 must be paired with a reliable protocol such as TCP to provide reliability.

A datagram in IPv4 is a variable-length packet consisting of two parts that are header and data. The header is 20 to 60 bytes in length and contains necessary information for routing and delivery. The different fields of header are given as follows:

Version (VER): It is a 4-bit field which defines the version of the IPv4 protocol. Currently the version is 4 but it may be replaced by version 6 in the future. This field tells the IPv4 software that the format of datagram is version 4. If the machine is using some other version of IPv4 then the datagram is discarded.

Header length (HLEN): It is a 4-bit field which defines the total length of the datagram header in 4-byte words. This field is required because the length of the header is variable. It is 20 to 60 bytes in length. Now when there are no options then the header length is 20 bytes and so the value of this field is 5 that mean 5 words with each word of 4 bytes in length. When the option field is at its maximum size then the value of this field is 15 that mean 15 words with each word of 4 bytes in length.

Services: It is an 8-bit field. Earlier this field is called service type and now it is called differentiated services.

Service Type: In this interpretation, the first 3 bits are called precedence bits. The next 4 bits are called type of service (TOS) bits and the last bit is not used.

Precedence bits is ranging from 0(000 in binary) to 7(111 in binary). The precedence bits provides the priority of the datagram in case of some problems like congestion. If a router is congested and needs to discard some datagrams then the datagrams with lowest precedence will be discarded first.

TOS bits is a 4-bit subfield with one and only one of the bits can have the value of 1 in each datagram. Each TOS bits has a special meaning given in the following table:

TOS Bits	Description
0000	Normal
0001	Minimize cost
0010	Maximize reliability
0100	Maximize throughput
1000	Minimize delay

Differentiated Services: In this interpretation, the first 6 bits make up the code point subfield and the last 2 bits are not used. Now the code point subfield can be used in two different ways given as follows:

a. If the 3 rightmost bits are 0s then the 3 leftmost bits are interpreted the same as the precedence bits in the service type interpretation.

b. If the 3 rightmost bits are not all 0s then the 6 bits defines 64 services based on the priority assignment by the Internet or local authorities. There are three categories of services. The first category contains 32 service types. The second and the third each contain 16 service types. The numbers for the first category are 0, 2, 4,…., 62 It is assigned by the Internet authorities (IETF). The numbers for the second category are 3, 7, 11, 15, and 63. It can be used by local authorities. The numbers for the third category are 1, 5, 9, 13, 17,…, 61. It is temporary and can be used for experimental purposes.

Total length: It is a 16 bit field which defines the total length of the IPv4 datagram in bytes.

Identification: It is a 16 bit field which identifies a datagram originating from the source host.

Flags: It is a 3 bit field. Here the first bit is reserved. The second bit is called “do not fragment bit”. If its value is 1 then the machine must not fragment the datagram. If it cannot pass the datagram through any available physical network then it discards the datagram and sends an ICMP error message to the source host. If its value is 0 then the datagram can be fragmented if necessary. The third bit is called the “more fragment bit”. If its value is 1 then it means that the datagram is not the last fragment. If its value is 0 then it means that this is the last or only fragment.

Fragmentation offset: It is a 13 bit field which shows the relative position of the fragment with respect to the whole datagram. It is the offset of the data in the original datagram measured in units of 8 bytes.

Time-to-Live (TTL): It is an 8 bit field. It stores a value which specifies the number of router hops the packet is yet allowed to travel before it must be discarded or returned. When a source host sends a datagram, it stores a value in this field which is approximately 2 times the maximum number of routes between any two hosts. Each router that processes the datagram decrement the value of this field by 1 and now if this value is zero then the router discards the datagram.

This field is required because sometimes routing tables in the internet may be corrupted. As a result of this a datagram may travel between two or more routers infinitely without getting delivered to the destination host. So this field is used to limit the journey of the packet.

Protocol: It is an 8-bit field which defines the higher-level protocol that uses the services of the IPv4 layer. An IPv4 datagram can encapsulate data from several higher-level protocols such as TCP, UDP, ICMP and IGMP. This field specifies the final destination protocol to which the IPv4 datagram is delivered. The following table shows the possible values in this field and corresponding higher level protocols:

Value	Protocol
1	ICMP
2	IGMP
6	TCP
17	UDP
89	OSPF

Source Address: It is a 32-bit field which provides the IPv4 address of the source. This field must remain same during the time the IPv4 datagram travels from the source host to the destination host.

Destination address: It is a 32-bit field which provides the IPv4 address of the destination. This field must remain same during the time the IPv4 datagram travels from the source host to the destination.

IPv4 has some drawbacks and as a result of these, it will not be suitable for the fast growing Internet.

- Due to the fast growth of the internet, in near future the address space of IPv4 will not be sufficient to accommodate all the devices in the internet.
- The Internet must accommodate real-time audio and video transmission which requires minimum delay strategies and reservation of resources. But IPv4 does not provide these.
- IPv4 does not provide any encryption and authentication mechanism.

5.6.2 IPV6

IPv6 (Internetworking Protocol, version 6) is the new version of internet layer protocol for packet-switched internetworking and it provides end-to-end datagram transmission across multiple IP networks. IPv6 was first formally described in Internet standard document RFC 2460. IPv6 is also referred as IPng(Internetworking Protocol, next generation). In IPv6, the packet format and the length of the IP address were changed. Here related protocols like ICMP were also modified and other protocols in the network

layer like ARP, RARP and IGMP were either deleted or included in the ICMPv6 protocol. Routing protocols like RIP and OSPF were also modified.

The Advantages of IPv6 are discussed as follows:

1. The length of IPv6 address is 128 bits and so the address space, 2^{128} is much larger than the address space, 2^{32} of IPv4.
2. IPv6 uses a new header format. In this format options are separated from the base header. In IPv6 the options are inserted when needed between the base header and the upper-layer data. So it simplifies and increases the speed of the routing process because most of the options do not need to be checked by routers.
3. IPv6 has new options to allow additional functionalities.
4. IPv6 allows the extension of the protocol if required by new technologies or applications.
5. In IPv6, new mechanism has been added to support traffic like real-time audio and video.
6. The encryption and authentication options are available in IPv6 which provide confidentiality and integrity of the packet.

Packet Format:

The IPv6 packet is consists of a base header followed by the payload. The payload consists of two parts which are optional extension headers and data from an upper layer. The length of the base header is 40 bytes and the payload contain up to 65535 bytes of information.

The base header is consists of eight fields discussed as follows:

1. **Version:** It is a 4 bit field which defines the version number of the IP.
2. **Priority:** It is a 4 bit field which defines the priority of the packet with respect to traffic congestion. The following table shows different possible priority values and its meaning.

Priority	Meaning
0	No specific traffic
1	Background data
2	Unattended data traffic
3	Reserved
4	Attended bulk data traffic
5	Reserved
6	Interactive traffic
7	Control traffic

3. **Flow label**: It is a 24 bit field which is designed to provide special handling for a particular flow of data.
4. **Payload length:** It is a 16 bit field which defines the length of the IP datagram excluding the base header.
5. **Next header:** It is an 8 bit field which defines the header that follows the base header in the datagram.
6. **Hop limit:** It is an 8 bit field which stores the number of network segments on which the packet is allowed to travel before being discarded by a router. The hop limit is set by the sending host and is used to prevent packets from infinitely circulating on an IPv6 internetwork. At the time of forwarding an IPv6 packet, IPv6 routers are required to decrease the Hop Limit by 1 and the IPv6 packet is discarded when the Hop Limit is 0.
7. **Source address:** It is a 128 bit field which stores a 128 bit internet address to identify the original source of the datagram.
8. **Destination address:** It is a 128 bit field which stores a 128 bit internet address used usually to identify the final destination of the datagram. But if source routing is used, this field contains the address of the next router.

Extension Headers:

There are six extension headers in IPv6 packet format discussed as follows:

Hop-by-Hop option: The hop-by-hop option is used when the source needs to send information to all routers which are visited by the datagram.

Source Routing: The source routing extension header is used to specify a list of intermediate nodes for a packet to travel on its path to its final destination.

Fragmentation: The fragmentation extension header is used in the purpose of fragmentation of datagram in IPv6. In IPv6, only the original source can fragment a datagram.

Authentication: The authentication extension header is used to validate the message sender and ensures the integrity of data.

Encrypted Security Payload: The encrypted security payload provides confidentiality.

Destination Option: The destination option is used when the source needs to send message to the destination only and this message is not permitted to access by the other routers.

5.6.3 ADDRESS RESOLUTION PROTOCOL (ARP)

The Address Resolution Protocol (ARP) is designed to create a mapping between physical and logical addresses. IP packets are encapsulated in a frame. IP packets use logical addresses and frame requires physical addresses. In internet, a packet starting from a source host may pass through different physical networks on the way to the destination host. Now the hosts and routers are recognized at the network level by their logical (IP) addresses and at the physical level, the hosts and routers are recognized by their physical addresses.

A physical address is a local address and so it must be unique in a local network but it may not unique universally. It is called a physical address because it is usually implemented in hardware.

Now delivery of a packet to a host or a router requires both logical and physical addressing. The mapping of a logical address to its corresponding physical address and vice versa can be done by using either static or dynamic mapping.

In static mapping, a table is created which associates a logical address with a physical address. This table is stored in each machine on the network. Static mapping table must be updated periodically because physical addresses may change in the following ways:

1. When a machine changes its NIC then a new physical address is created.
2. The physical address changes every time the computer is turned on in case of some LANs. For example: LocalTalk
3. In case of a mobile computer, physical address changes whenever it move from one physical network to another.

Now updating the static mapping table could degrade the network performance.

In case of dynamic mapping, ARP is used by a machine to find a logical address if it knows corresponding physical address and vice versa. When a host or a router has an IP datagram to send to another host or router, it has the logical (IP) address of the receiver. If the sender is the host then the logical (IP) address is obtained from the DNS and if the sender is a router then the logical address is obtained in a routing table. Now the sender requires the physical address of the receiver. So the host or the router sends an ARP query packet which includes the physical and IP addresses of the sender and the IP address of the receiver. This query is broadcast over the network and so every host or router on the network receives the ARP query packet. Now the actual recipient recognizes its IP address in the ARP query packet and sends back an ARP response packet. This ARP response packet contains the logical address (IP) and the physical address of the recipient. The ARP response packet is sent directly only to the sender of the ARP query packet by using the physical address received in the query packet.

The performance of ARP can be improved by using cache memory to store the ARP reply packets because a system normally sends several packets to the same destination. A system that stores ARP reply packets in the cache memory always checks the cache memory to find the required mapping before sending an ARP request.

The different fields of an ARP packet are as follows:

1. Hardware type is a 16-bit field defining the type of the network on which ARP is running. Each LAN has been specified by an integer based on its type.
2. Protocol type is a 16-bit field defining the protocol.
3. Hardware length is an 8-bit field defining the length of the physical address in bytes.
4. Protocol length is an 8-bit field defining the length of the logical address in bytes.
5. Operation is a 16-bit field defining the type of packet which can be either ARP request (1) or ARP reply (2).
6. Sender hardware address is a variable-length field defining the physical address of the sender.

7. Sender protocol address is a variable-length field defining the logical address of the sender. In case of IP protocol, the length of this field is 4 bytes.
8. Target hardware address is a variable-length field defining the physical address of the receiver.
9. Target protocol address is a variable-length field defining the logical address of the receiver. In case of IPv4 protocol, the length of this field is 4 bytes.

Proxy ARP:

A proxy ARP is an ARP that acts on behalf of a set of hosts. When a router running a proxy ARP receives an ARP request looking for the IP address of one of these hosts then the router sends an ARP reply with its own physical address. After the router receives the actual IP packet, it sends the packet to the appropriate host or router.

So using proxy ARP a network can be extended without the knowledge of the upstream router.

5.6.4 INTERNET CONTROL MESSAGE PROTOCOL (ICMP)

The IP protocol does not have any error-reporting or error-correcting mechanism and it also does not have any mechanism for host and management queries. Now the Internet Control Message Protocol (ICMP) is designed to provide these mechanisms.

ICMP messages are divided into two categories which are error-reporting messages and query messages.

The error-reporting messages report problems which are occurred when a router or a destination host processes an IP packet.

A host or a network manager gets specific information from a router or another host from query messages which are occurred in pairs.

An ICMP message has two parts that are an 8-byte header and a variable-size data section. Now the first 4-bytes of the header are common for all messages. The first field of the header is ICMP type which defines the type of the message. The second field of the header is code field that specifies the reason for the particular message type and the last common field is the checksum field. The remaining part of the header is specific for each message type.

The data section in error message contains the information for finding the original packet where the error is occurred. The data section in query messages contains information based on the type of the query.

ICMP uses the source IP address to send the error message to the source of the datagram. In general, ICMP reports five types of errors which are destination unreachable, source quench, time exceeded parameter problems and redirection.

When a router cannot route a datagram or a host cannot deliver a datagram then the datagram is discarded and the router or the host sends a destination-unreachable message created by either a router or the destination host, to the source host that has produced the datagram.

In case of IP protocol, there is no communication between the source host, the routers and the destination host. So in this situation, congestion can occur because IP does not

have any flow control mechanism. If the datagrams are received much faster than they can be forwarded or processed, the buffer of a router or a host may overflow as its size is limited. In this case, the router or the host must discard some of the datagrams. Now the source-quench message in ICMP provides a kind of flow control to the IP. When a router or host discards a datagram due to congestion, it sends a source-quench message to the sender of the datagram. This message informs the source that the datagram has been discarded and warns the source that it should slow down the sending process as there is congestion somewhere in the path from the source to the destination host.

Routers use routing tables to find the next router for sending packets. Now if there are errors in one or more routing tables then sometimes a packet may travel in a loop from one router to the next or a series of routers infinitely. So in this situation, each datagram contains a field called time to live to control it. When a datagram visits a router, the value of this field is decremented by 1. When the time-to-live value reaches 0, the router discards the datagram and a time-exceeded message must be sent by the router to the original source. Another time-exceeded message is also generated when all parts of a message does not arrive at the destination host within a certain time limit.

If a router or the destination host discovers an ambiguous or missing value in any part of the datagram then it discards the datagram and sends a parameter-problem message back to the source.

In IP, both routers and hosts require a routing table to find the address of the router or the next router. Routers update the routing tables constantly but hosts do not take part in the routing update process as the number of hosts in internet is much more than routers. So in general, the host uses static routing with a routing table that has a limited number of entries. As a result of this, the host may send a datagram to the wrong router. In this case, the router that receives the datagram will forward the datagram to the correct router and send a redirection message to the host for updating the routing table of the host.

A query message is encapsulated in an IP packet and the IP packet is encapsulated in a data link layer frame. The four pairs of query messages are discussed as follows:

The echo-request and echo-reply messages are used by network managers and users to identify network problems. This pair of messages can be used to determine the presence of communication at the IP level. ICMP messages are encapsulated in IP datagrams, so if a machine which sent an echo request, receive an echo-reply message then it means that the IP protocols in the sender and receiver are communicating with each other using the IP datagram and the intermediate routers are also receiving, processing and forwarding IP datagrams.

The timestamp request and timestamp reply messages are used by any two machines to determine the round-trip time needed for an IP datagram to travel between them. This pair of query messages is also used to synchronize the clocks in two machines.

The address-mask request and reply messages are used by a host to obtain its mask. A host sends an address-mask-request message to a router on the LAN to obtain its mask. Now if the host knows the address of the router then it sends the request directly to the router otherwise it broadcasts the message. After receiving the address-mask-request message, the router sends an address-mask-reply message, providing the necessary mask for the host.

A host that wants to send data to a host on another network can obtain the address of routers connected to its network by using the router solicitation and advertisement query messages. This pair of query messages is also used to know if the routers are alive and functioning. A host can broadcast a router-solicitation message. The routers receive the solicitation message and broadcast their routing information using the router-advertisement message.

There are two tools that are ping and traceroute can be used in the Internet for debugging. These tools use ICMP for debugging.

Ping program is used to determine if a host is alive and responding.

The traceroute program in UNIX or tracert in Windows can be used to trace the route of a packet from the source to the destination.

CHECK YOUR PROGRESS

1. Multiple choice questions:

(I) The length of IPv4 address is

A. 32 bits

B. 128 bits

C. 32 bytes

D. None of above

(II) The size of the address space of IPv6 is

A. 2^{16}

B. 2^{32}

C. 2^{128}

D. Both B and C

(III) Which is not a subcategory of reserved addresses in IPv6?

A. Loopback address

B. Mapped address

C. Local address

D. Compatible address

(IV) Which is the decision making strategy of Multicast distance vector routing algorithm?

A. Flooding
B. Reverse path forwarding
C. Reverse path broadcasting
D. All of the above

(V) Which one is a unicast routing protocol?

A. Open Shortest Path First protocol
B. Address Resolution Protocol.
C. DVMRP protocol
D. None of the above

(VI) Which protocol is designed to create mapping between physical and logical addresses.

A. Internet Control Message Protocol
B. Internet Protocol
C. Address Resolution Protocol.
D. Both B and C

(VII) The length of the header part in an Internet Control Message Protocol message

A. 2 byte
B. 4 byte
C. 8 byte
D. 16 byte

(VIII) Routing Information Protocol is a

A. Multicast routing protocol
B. Broadcast routing protocol
C. Unicast routing protocol
D. None of the above

(IX) Which is not a type of link specified in OSPF protocol?

A. Point to point
B. Transient
C. Virtual
D. Backbone

(X) Which is not a dynamic routing algorithm?

A. Flooding
B. Distance vector routing algorithm
C. Link state routing algorithm
D. Both B and C

2. Fill in the blanks:

I. The internetwork is a _____ network.
II. _____service is used in the datagram approach to packet switching.
III. The term IP address is used to mean a _____ address in the network layer of the TCP/IP protocol suite.
IV. The length of an IPv6 address is _____ bytes.
V. In IPv6, reserved addresses start with _____s.
VI. In case of OSPF protocol, there is a special area inside an autonomous system called _____.
VII. Distance Vector Multicast Routing Protocol (DVMRP) is a source-based routing protocol based on _____.
VIII. The performance of ARP can be improved by using _____memory.
IX. An Internet Control Message Protocol message has two parts: _____ and a _____.
X. If the decimal value of the first byte of an IPv4 address is in the range _____ then it is a class B address.

3. State whether the following statements are True of False
   I. If the decimal value of the first byte of an IPv4 address is in the range 192 to 223 then it is a class A address.
   II. The basic concept of NAT is to assign each company a large set of addresses internally and one address or a small set of addresses externally.
   III. Link state routing is a static routing algorithm.
   IV. Hardware type is a 16-bit field defining the type of the network on which ARP is running.
   V. MOSPF protocol uses multicast link state routing to create source-based trees.
   VI. IPv4 is an unreliable and connection oriented protocol.
   VII. The IPv6 packet is consists of a base header followed by the payload.
   VIII. Using proxy ARP a network can be extended without the knowledge of the upstream router.
   IX. The IP protocol has an error-reporting or error-correcting mechanism.
   X. Ping program is used to determine if a host is alive and responding.

5.7 LET US SUM UP

The summery of this unit is given as follows:

- The network layer provides the mechanism of transferring variable length data sequences from a source host on one network to a destination host on a different network.
- A collection of interconnected networks, permitting data to move freely among different networks is called an internetwork or internet.
- An IPv4 address is a 32 bit address which uniquely and globally defines the connection of a device to the internet.
- Two types of notations are used to show an IPv4 address: binary notation and dotted-decimal notation.
- There are two addressing scheme in IPv4: classful addressing and classless addressing.
- Network Address Translation (NAT) is a solution for address depletion in IPv4 addressing.
- IPv6 has an address space of 2^{128} addresses because it uses a 128-bit address.
- IPv6 specifies hexadecimal colon notation for its addresses where 128 bits is divided into eight sections and each 2 bytes in length.

- In case of IPv6, the IP addresses are divided into several categories: Unicast Addresses, Multicast addresses, Anycast Addresses, Reserved Addresses and Local Addresses.
- The routing algorithms are the part of network layer software responsible for deciding the routes and the data structures to transmit the incoming packets efficiently.
- The different goals of a routing algorithm are: Correctness, Simplicity, Robustness, Stability, Fairness and Optimality.
- Routing algorithms can be divided into two major classes: nonadaptive and adaptive algorithms.
- Nonadaptive algorithms are static routing algorithms where the choice of the route to transmit IP packets from one node to another node is computed in advance and downloaded to the routers when the network is booted.
- Adaptive algorithms are dynamic algorithms where the routing decisions are changed whenever there is a change in the topology and the network traffic.
- Routing protocols are divided into categories: unicast and multicast protocols.
- In unicast routing, when a router receives a packet to route then it needs to find the shortest path to the destination of the packet.
- In multicast routing, router receives multicast packet for routing to the destinations in more than one network.
- Distance vector routing and link state routing are two dynamic routing algorithm.
- The Routing Information Protocol (RIP) is an intradomain routing protocol used inside an autonomous system based on distance vector routing.
- The open shortest path protocol is an intradomain unicast routing protocol based on link state routing.
- Multicast link state routing is an extension of unicast link state routing.
- Multicast distance vector routing is the extension of unicast distance vector routing.
- Multicast Open Shortest Path First (MOSPF) Protocol is an extension of the OSPF protocol.
- DVMRP is a multicast routing protocol which uses multicast distance vector routing.
- The Internet Protocol version 4(IPv4) is used by the TCP/IP protocol.
- IPv4 is an unreliable and connectionless protocol for a packet-switching network that uses the datagram approach.
- IPv6 (Internetworking Protocol, version 6) is the new version of internet layer protocol for packet-switched internetworking and it provides end-to-end datagram transmission across multiple IP networks.
- The IPv6 packet is consists of a base header followed by the payload.
- The Address Resolution Protocol (ARP) is designed to create a mapping between physical and logical addresses.

- The Internet Control Message Protocol (ICMP) is designed to provide error-reporting or error-correcting mechanism. It also provides mechanism for host and management queries.
- ICMP messages are divided into two categories which are error-reporting messages and query messages.

5.8 ANSWERS TO CHECK YOUR PROGRESS

1. (I) A , (II) C , (III) C , (IV) D , (V) A ,(VI) C ,(VII) C ,(VIII) C ,(IX) D ,(X) A

2. I. packet-switched ,II. connectionless , III. logical , IV. 16 , V. eight 0 , VI. backbone, VII. RIP , VIII. cache , IX. Header, data section , X. 128 to 191.

3. I. False ,II. True , III. False , IV. True , V. True , VI. False , VII. True , VIII. True , IX. False , X. True

5.9 FURTHER READINGS

- Behrouz A Forouzan : Data Communications and Networking ,TATA McGraw Hill
- William Stallings : Data and Computer Communications, Pearson Education
- Andrew S. Tanenbaum : Computer Networks, Prentice-Hall India

5.10 MODEL QUESTIONS

- Explain Different addressing scheme in Pv4.
- Define Routing. What are the different goals of routing algorithm?
- Explain distance vector routing and link state routing.
- Explain the different fields of an IPv4 packet.
- Explain Address Resolution Protocol (ARP).

UNIT 6: THE TRANSPORT LAYER

UNIT STRUCTURE

6.1 INTRODUCTION

Transport Layer is 4th layer of the network OSI model. The transport layer of the OSI model offers end-to-end communication between end devices through a network. It also offers application-to-application communication between the two applications in the both host. Depending on the application, the transport layer offers reliable, connection-oriented or connectionless, best-effort communications.

The two most common transport layer protocols are the connection-oriented TCP Transmission Control Protocol (TCP) and the connectionless UDP User Datagram Protocol (UDP).

The transport layer ensures that messages are delivered error-free, in sequence, and with no losses or duplications. It relieves the higher layer protocols from any concern with the transfer of data between them and their peers. The size and complexity of a transport protocol depends on the type of service it can get from the network layer. For a reliable network layer with virtual circuit capability, a minimal transport layer is required. If the network layer is unreliable and/or only supports datagram, the transport protocol should include extensive error detection and recovery.

The transport layer provides transfer of data between end users, providing reliable data transfer services to the upper layers. It controls the reliability of a given link through flow control, segmentation/de-segmentation, and error control. The transport layer can keep track of the segments and retransmit those that fail. The transport layer also provides the acknowledgement of the successful data transmission and sends the next data if no errors occurred.

The transport layer provides:

- **Message segmentation:** It accepts a message from the (session) layer above it, splits the message into smaller units, if required, and passes the smaller units down to the network layer. The transport layer at the destination end re-assembles the message to get the original message back.
- **Message acknowledgment:** It provides reliable end-to-end message delivery with acknowledgments.
- **Message traffic control:** It tells the transmitting host to ‘wait-for-sometime’ when no message buffers are available in the receiving host.
- **Session multiplexing:** It multiplexes several message streams, or sessions onto one logical link and keeps track of which messages belong to which sessions.

Typically, the transport layer can accept relatively large messages, but there are strict message size limits imposed by the network (or lower) layer. Consequently, the transport layer must break up the messages into smaller units, or frames, put a transport layer header to each frame. The transport layer header includes control information, such as message start and message end flags, to enable the transport layer on the other end to recognize

message boundaries. In addition, if the lower layers do not maintain sequence, the transport header must contain sequence information to enable the transport layer on the receiving end to get the pieces back together in the right order before handing the received message up to the layer above.

Residing between the application and network layers, the transport layer has the critical role of providing communication services directly to the application processes running on different hosts. In this chapter we'll examine the possible services provided by a transport layer protocol and the principles underlying various approaches towards providing these services.

6.1.1 Relationship between Transport and Network Layers

In theory, the transport layer and network layer are distinct, but in practice, they are often very closely related to each other. We can see this easily just by looking at the names of common protocol stacks—they are often named after the layer three and four protocols in the suite, implying their close relationship. For example, the name “TCP/IP” comes from the suite’s most commonly used transport layer protocol (TCP) and network layer protocol (IP). Similarly, the Novell NetWare suite is often called “IPX/SPX” for its layer three (IPX) and layer four (SPX) protocols. Typically, specific transport layer protocols consider the network layers in the same family.

In the sending end, the transport layer of the network software takes input from the above layer. In case of TCP/IP protocol suit, it is the session layer. Then the transport layer processes them according to its own mechanism. After the process, the frames are handed over to the next lower layer in the protocol suit. For TCP/IP, it is the network layer. The network layer prepares them to transfer through the physical medium used in the network.

In the receiving end, the reverse process is executed. The incoming frames are accepted by the network layer. Network layer removes the network layer header and process those frames accordingly. Once this layer’s processing is finished, the frames are handed over to the transport layer.

The most commonly used transport layer protocols are the Transmission Control Protocol (TCP) and User Datagram Protocol (UDP) in the TCP/IP suite; the Sequenced Packet Exchange (SPX) protocol in the NetWare protocol suite; and the NetBEUI protocol in the NetBIOS/NetBEUI/NBF suite.

6.1.2 Overview of the Transport Layer in the Internet

The backbone of Internet is the TCP/IP protocol suit. This protocol suit makes data transfer possible among heterogeneous networks. It is using two distinct transport-layer protocols and makes them available to the application layer. One is **UDP** (User Datagram Protocol), which provides an unreliable, connectionless service to the invoking application. The second one is **TCP** (Transmission Control Protocol), which provides a reliable & connection-oriented service to the invoking application. When designing a network application using TCP/IP protocol suit, the application developer must specify any one of these two transport layer protocols.

6.1.3 Ports and sockets

This section introduces the concepts of **port** and **socket**, which are needed to determine which local application in the sending end actually wants to communicates with which remote application in the receiving end.

6.1.3.1 Ports

A port is a 16-bit number, used by the host-to-host protocol to identify to which higher level protocol or application process it must deliver incoming messages.

There are two types of port:

• **Well-known:** Well-known ports belong to standard server applications. For example, Telnet uses well-known port number 23, HTTP uses well-known port number 80 etc. Well-known port numbers range between 1 and 1023. Most servers require only a single port. Exceptions are the FTP server, which uses two: 20 and 21. The well-known ports are controlled and assigned by the Internet Assigned Number Authority (IANA) and on most systems can only be used by system processes or by programs executed by privileged users. The reason for well-known ports is to allow clients to be able to find servers without configuration information. The well-known port numbers are defined in STD 2 – Assigned Internet Numbers

• **Ephemeral:** Clients do not need well-known port numbers because they initiate communication with servers and the port number they are using is contained in the TCP/UDP datagram sent to the server. Each client application is allocated a port number for as long as it needs it by the host it is running on. Ephemeral port numbers have values greater than 1023, normally in the range 1024 to 65535. A client can use any number allocated to it, as long as the combination *of <transport protocol, IP address, port number>* is unique. Ephemeral ports are not controlled by IANA and can be used by ordinary user-developed programs on most systems.

Due to two different applications trying to use the same port numbers on one host, confusion is avoided by writing those applications to request an available port from TCP/IP. Because this port number is dynamically assigned, it may differ from one invocation of an application to the next. UDP & TCP all use the same port principle. To the best possible extent, the same port numbers are used for the same services on top of UDP & TCP.

6.1.3.2 Sockets

Since transport layer at the receiving host delivers data to the socket, there should be a unique identifier for each socket.

Socket identifier is called socket address.

Socket address = IP address : Port number (Figure 1)

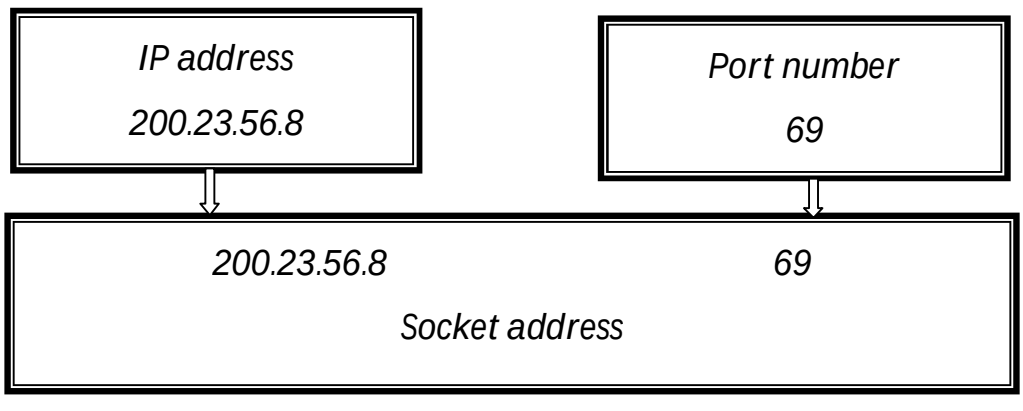


Figure 1: Socket Address

• A socket is a special type of file handle, which is used by an application to request network services from the operating system.

• A socket address is: <IP-address and local-application>

For example, in the TCP/IP suite: <193.44.234.3, 12345>

A socket is an endpoint for communication that can be named and addressed in a network. Two applications in two remote PCs communicate via TCP sockets. The socket model provides an application a connection to another application. These facilities are provided by TCP. TCP uses the same port principle as UDP to provide the communication. Like UDP, TCP uses well-known and ephemeral ports. If two applications are communicating over TCP, they have a logical connection that is uniquely identifiable by the two sockets involved, that is, by the combination <local IP address, local port, remote IP address, remote port>.

6.2 Process-To-Process delivery

The transport layer is responsible for process-to-process delivery—the delivery of a packet, part of a message, from one application to another over network. Two processes communicate in a client /server architecture. UDP and TCP are transport-layer protocols that create a process-to-process communication. UDP is an unreliable and connectionless protocol that requires little overhead and offers fast delivery.

In the client-server paradigm, an application program on the local host, called the client, needs services from an application program on the remote host, called a server. Each application program has a unique port number that distinguishes it from other programs running at the same time on the same machine. The client program is assigned a random port number called the ephemeral port number. The server program is assigned a universal port number called a well-known port number. The combination of the IP address and the port number, called the socket address, uniquely defines a process and a host. TCP uses a sliding window mechanism for flow control.

6.3 User Datagram Protocol (UDP)

UDP is a connectionless, unreliable transport protocol. It does not add anything to the services of IP except to provide process-to process communication instead of host-to-host communication. UDP is a very simple protocol using a minimum of overhead. If a process wants to send a small message and does not care much about reliability, it can use UDP.

UDP uses a simple 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. As this is normally IP over unreliable media, there is no guarantee of delivery, ordering or duplicate protection. UDP provides Checksum for data integrity, and port number for addressing different functions at the source and destination of the datagram. UDP is suitable for purposes where error checking and correction is either not necessary or 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. If error correction facilities are needed at the network interface level, an application may use the Transmission Control Protocol (TCP) which is designed for this purpose.

UDP datagram is having 8-byte header which is divided into four fields each with 2 bytes. (Figure 2)

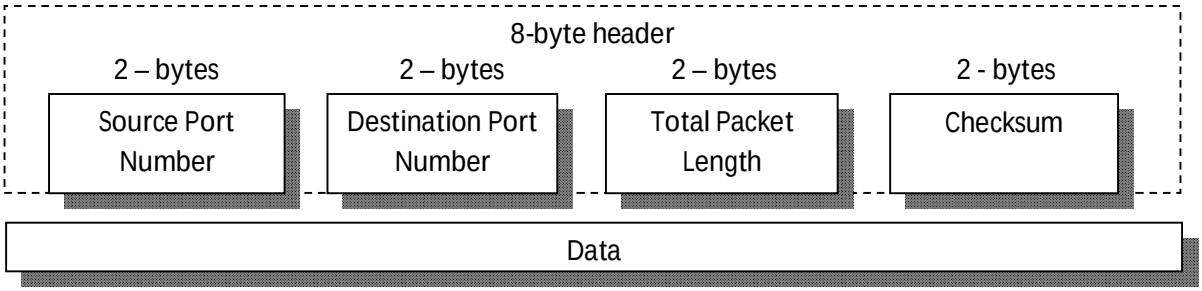


Figure 2: UDP datagram

1. **Source port number:** This is the port number used by the process running on the source host.

   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 is the port number used by the process running on the destination host. 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.

3. **Total Packet Length:** It defines the total length of the user datagram. A field that specifies the length in bytes of the entire datagram: header and data. The minimum length is 8 bytes since that's 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).

4. **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. This field is not optional for IPv6.

6.3.1 Applications of UDP

Numerous key Internet applications use UDP, including: the Domain Name System (DNS), where queries must be fast and only consist of a single request followed by a single reply packet, the Simple Network Management Protocol (SNMP), the Routine Information Protocol (RIP) and the Dynamic Host Configuration Protocol (DHCP).Voice and video traffic is generally transmitted using UDP. Real-time video and audio streaming protocols are designed to handle occasional lost packets, so only slight degradation in quality occurs, rather than large delays if lost packets were retransmitted.

6.3.2 Suitability of UDP for certain application

- It is **transaction-oriented**, suitable for simple query-response protocols such as the Domain Name Systems or the Network Time Protocol.
- It provides datagrams, 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 (VoIP), 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.
- Sending a small message by using UDP takes much less interaction between the sender and receiver than using TCP or SCTP. UDP packets, called user datagrams, have a fixed size header of 8 bytes.

6.3.3 Some drawback of UDP

- There is no flow control: The receiver may overflow with incoming messages.
- There is no error control mechanism in UDP except for the checksum.
- The sender does not know if a message has been lost or duplicated.
- When the receiver detects an error through the checksum, the user datagram is discarded.
- Each user datagram can travel on a different path. No relationship between the different user datagram even if they are coming from the same source process and going to the same destination program. Also, there is no connection establishment and no connection termination.

- UDP provides no guarantees to the upper layer protocol for message delivery and the UDP protocol layer retains no state of UDP messages once sent. For this reason, UDP is sometimes referred to as unreliable Datagram Protocol.

6.4 Transmission Control Protocol (TCP)

The **Transmission Control Protocol** (**TCP**) is one of the two original core protocols of the Internet protocol suite (IP), and is so common that the entire suite is often called TCP/IP. TCP provides reliable, ordered, error-checked delivery of a stream of octets between programs running on computers connected to an intranet or the public Internet. Browsers use it when they connect to servers on the World Wide Web sites, and it is used to accurately deliver email and transfer files from one location to another. Applications that do not require the reliability of a TCP connection may instead use the connectionless User Datagram Protocol (UDP).

The protocol corresponds to the transport layer of TCP/IP suite. TCP provides a communication service at an intermediate level between an application program and the Internet Protocol (IP). That is, when an application program desires to send a large chunk of data across the Internet using IP, instead of breaking the data into IP-sized pieces and issuing a series of IP requests, the software can issue a single request to TCP and let TCP handle the IP details.

Due to network congestion, traffic load balancing, or other unpredictable network behavior, IP packets can be lost, duplicated, or delivered out of order. TCP detects these problems, requests retransmission of lost data, rearranges out-of-order data, and even helps minimize network congestion to reduce the occurrence of the other problems. Once the TCP receiver has reassembled the sequence of octets originally transmitted, it passes them to the application program. Thus, TCP abstracts the application's communication from the underlying networking details.

TCP is utilized extensively by many of the Internet's most popular applications, including the World Wide Web (WWW), E-mail, File Transfer Protocol, Secure Shell, peer-to-peer file sharing, and some streaming media applications.

TCP is optimized for accurate delivery rather than timely delivery, and therefore, TCP sometimes incurs relatively long delays (in the order of seconds) while waiting for out-of-order messages or retransmissions of lost messages. It is not particularly suitable for real-time applications such as VoIP. For such applications, protocols like the Real-time Transport Protocol (RTP) running over the User Datagram Protocol (UDP) are usually recommended instead.

TCP is a reliable stream delivery service that guarantees that all bytes received will be identical with bytes sent and in the correct order. Since packet transfer is not reliable, a technique known as positive acknowledgment with retransmission is used to guarantee reliability of packet transfers. This fundamental technique requires the receiver to respond with an acknowledgment message as it receives the data. The sender keeps a record of each packet it sends. The sender also keeps a timer from when the packet was sent, and

retransmits a packet if the timer expires before the message has been acknowledged. The timer is needed in case a packet gets lost or corrupted.

TCP consists of a set of rules: for the protocol, that are used with the Internet Protocol, and for the IP, to send data "in a form of message units" between computers over the Internet. While IP handles actual delivery of the data, TCP keeps track of the individual units of data transmission, called segments that a message is divided into for efficient routing through the network. For example, when an HTML file is sent from a Web server, the TCP software layer of that server divides the sequence of octets of the file into segments and forwards them individually to the IP software layer (Internet Layer). The Internet Layer encapsulates each TCP segment into an IP packet by adding a header that includes (among other data) the destination IP address. Even though every packet has the same destination address, they can be routed on different paths through the network. When the client program on the destination computer receives them, the TCP layer (Transport Layer) reassembles the individual segments and ensures they are correctly ordered and error free as it streams them to an application.

6.4.1. TCP can be characterized by the following facilities it provides for the applications using it:

- **Stream Data Transfer:**

From the application's viewpoint, TCP transfers a continuous stream of bytes through the network. TCP does this by grouping the bytes in TCP segments, which are passed to IP for transmission to the destination. Also, TCP itself decides how to segment the data and it can forward the data at its own convenience. Sometimes, an application needs to be sure that all the data passed to TCP has actually been transmitted to the destination. For that reason, a push function is defined. It will push all remaining TCP segment still in storage to the destination host. The normal close connection function also pushes the data to the destination.

• Reliability:

TCP uses *a sequence number* to identify each byte of data. The sequence number identifies the order of the bytes sent from each computer so that the data can be reconstructed in order, regardless of any fragmentation, disordering, or packet loss that may occur during transmission. For every payload byte transmitted, the sequence number must be incremented. In the first two steps of the 3-way handshake, both computers exchange an initial sequence number (ISN). This number can be arbitrary, and should in fact be unpredictable to defend against TCP sequence prediction attacks.

TCP primarily uses a *cumulative acknowledgment* scheme, where the receiver sends an acknowledgment signifying that the receiver has received all data preceding the acknowledged sequence number. The sender sets the sequence number field to the sequence number of the first payload byte in the segment's data field, and the receiver sends an acknowledgment specifying the sequence number of the next byte they expect to receive. For example, if a sending computer sends a packet containing four payload bytes with a sequence number field of 100, then the sequence numbers of the four payload bytes are 100,

101, 102 and 103. When this packet arrives at the receiving computer, it would send back an acknowledgment number of 104 since that is the sequence number of the next byte it expects to receive in the next packet.

In addition to cumulative acknowledgments, TCP receivers can also send *selective acknowledgments* to provide further information. If the sender infers that data has been lost in the network, it retransmits the data.

- **Flow Control**

The receiving TCP, when sending an acknowledgment back to the sender, also indicates to the sender the number of bytes it can receive beyond the last received TCP segment, without causing overrun and overflow in its internal buffers. This is sent in the acknowledgment in the form of the highest sequence number it can receive without problems. This mechanism is also referred to as a *window-mechanism*.

TCP uses an end-to-end flow control protocol to avoid having the sender send data too fast for the TCP receiver to receive and process it reliably. Having a mechanism for flow control is essential in an environment where machines of diverse network speeds communicate. For example, if a PC sends data to a Smartphone that is slowly processing received data, the Smartphone must regulate the data flow so as not to be overwhelmed.

TCP uses a *sliding window flow control protocol*. In each TCP segment, the receiver specifies in the *receive window* field the amount of additionally received data (in bytes) that it is willing to buffer for the connection. The sending host can send only up to that amount of data before it must wait for an acknowledgment and window update from the receiving host.

TCP sequence numbers and receive windows behave very much like a clock. The receive window shifts each time the receiver receives and acknowledges a new segment of data. Once it runs out of sequence numbers, the sequence number loops back to 0.

When a receiver advertises a window size of 0, the sender stops sending data and starts the *persist timer*. The persist timer is used to protect TCP from a deadlock situation that could arise if a subsequent window size update from the receiver is lost, and the sender cannot send more data until receiving a new window size update from the receiver. When the persist timer expires, the TCP sender attempts recovery by sending a small packet so that the receiver responds by sending another acknowledgement containing the new window size.

- **Congestion control**

Another main aspect of TCP is congestion control. TCP uses a number of mechanisms to achieve high performance and avoid congestion collapse, where network performance can fall by several orders of magnitude. These mechanisms control the rate of data entering the network, keeping the data flow below a rate that would trigger collapse. They also yield an approximately max-min fair allocation between flows.

Acknowledgments for data sent, or lack of acknowledgments, are used by senders to infer network conditions between the TCP sender and receiver. Coupled with timers, TCP senders

and receivers can alter the behavior of the flow of data. This is more generally referred to as congestion control and/or network congestion avoidance.

Enhancing TCP to reliably handle loss, minimize errors, manage congestion and go fast in very high-speed environments are ongoing areas of research and standards development. As a result, there are a number of TCP congestion avoidance algorithm variations.

- **Maximum segment size**

The maximum segment size (MSS) is the largest amount of data, specified in bytes, that TCP is willing to receive in a single segment. For best performance, the MSS should be set small enough to avoid IP fragmentation, which can lead to packet loss and excessive retransmissions. To try to accomplish this, typically the MSS is announced by each side using the MSS option when the TCP connection is established, in which case it is derived from the maximum transmission unit (MTU) size of the data link layer of the networks to which the sender and receiver are directly attached. Furthermore, TCP senders can use path MTU discovery to infer the minimum MTU along the network path between the sender and receiver, and use this to dynamically adjust the MSS to avoid IP fragmentation within the network.

MSS announcement is also often called "*MSS negotiation*". Strictly speaking, the MSS is not "*negotiated*" between the originator and the receiver, because that would imply that both originator and receiver will negotiate and agree upon a single, unified MSS that applies to all communication in both directions of the connection. In fact, two completely independent values of MSS are permitted for the two directions of data flow in a TCP connection. This situation may arise, for example, if one of the devices participating in a connection has an extremely limited amount of memory reserved (perhaps even smaller than the overall discovered Path MTU) for processing incoming TCP segments.

- **Selective acknowledgments**

Relying purely on the cumulative acknowledgment scheme employed by the original TCP protocol can lead to inefficiencies when packets are lost. For example, suppose 10,000 bytes are sent in 10 different TCP packets, and the first packet is lost during transmission. In a pure cumulative acknowledgment protocol, the receiver cannot say that it received bytes 1,000 to 9,999 successfully, but failed to receive the first packet, containing bytes 0 to 999. Thus, the sender may then have to resend all 10,000 bytes.

To solve this problem TCP employs the *selective acknowledgment (SACK)* option, which allows the receiver to acknowledge discontinuous blocks of packets that were received correctly, in addition to the sequence number of the last contiguous byte received successively, as in the basic TCP acknowledgment. The acknowledgement can specify a number of SACK blocks, where each SACK block is conveyed by the starting and ending sequence numbers of a contiguous range that the receiver correctly received. In the example above, the receiver would send SACK with sequence numbers 1000 and 9999. The sender thus retransmits only the first packet, bytes 0 to 999.

An extension to the SACK option is the duplicate-SACK option. An out-of-order packet delivery can often falsely indicate the TCP sender of lost packet and, in turn, the TCP sender retransmits the suspected-to-be-lost packet and slows down the data delivery to prevent network congestion. The TCP sender undoes the action of slow-down that is a recovery of the original pace of data transmission, upon receiving a D-SACK that indicates the retransmitted packet is duplicated.

The SACK option is optional and used only if both parties support it. This is negotiated when connection is established. SACK uses the optional part of the TCP header. The use of SACK is widespread — all popular TCP stacks support it. Selective acknowledgment is also used in *Stream Control Transmission Protocol (SCTP)*.

- **Multiplexing:** Achieved through the use of ports, just as with UDP.
- **Logical Connections:**

The reliability and flow control mechanisms described above require that TCP initializes and maintains certain status information for each data stream. The combination of this status, including sockets, sequence numbers and window sizes, is called a *logical connection*. Each connection is uniquely identified by the pair of sockets used by the sending and receiving processes.

• **Full Duplex:** TCP provides for concurrent data streams in both directions.

6.4.2. TCP SEGMENT STRUCTURE

Transmission Control Protocol accepts data from a data stream, divides it into chunks, and adds a TCP header creating a TCP segment. The TCP segment is then encapsulated into an Internet Protocol (IP) datagram. A TCP segment is the packet of information that TCP uses to exchange data with its peers.

The term TCP packet is not in line with current terminology, where segment refers to the TCP Protocol Data Unit (PDU), datagram to the IP PDU and frame to the data link layer PDU.

Processes transmit data by calling on the TCP and passing buffers of data as arguments. The TCP packages the data from these buffers into segments and calls on the internet module, e.g. IP, to transmit each segment to the destination TCP.

A TCP segment consists of *a segment header* and *a data section*. The TCP header contains 10 mandatory fields, and an optional extension field. The data section follows the header. Its contents are the payload data carried for the application. The length of the data section is not specified in the TCP segment header. It can be calculated by subtracting the combined length of the TCP header and the encapsulating IP header from the total IP datagram length***.***

The following figure 3 shows the layout of a TCP segment header. The header is of size 20 to 60 bytes. The 20 bytes is reserved for the fixed fields and the other 40 bytes are reserved for options, which is the additional information that the header carries to the destination. These 40 bytes are not mandatory.

Let us discuss briefly these header fields:

1. **Source Port Number:** This is 2-bytes port number of the application on the source computer which wants to send the TCP segment.

2. **Destination Port Number:** This is 2-bytes port number of the application on the destination computer which is expected to receive the TCP segment.

3. **Sequence Number:** TCP sends multiple segments from source to destination in one TCP connection. So, it becomes important to number those segments in an increasing sequence to maintain the connectivity. The 4 bytes sequence number is assigned a number to the first byte of data portion in the TCP segment. It makes the destination host clear about the first byte of the TCP segment.

   During the TCP connection phase, the source and destination host randomly generate different unique numbers. Let's assume that for the source host, this random unique number is 2120 and first TCP segment is carrying 2000 bytes of data. The number 2120 and 2121 will be used in establishing the TCP connection with the destination host. Then the next number 2122 will be used in the sequence number field of the first TCP segment which will carry 2000 bytes of data. The second segment will carry the sequence number 4122 (= 2122+2000).

4. **Acknowledgement Number:** If the destination host receives a segment with sequence number X correctly, it sends 4 bytes acknowledgement number X+1 back to the source host which acknowledge the correct receipt of the previous segment from the source.

5. **Header Length:** This 4 bits field specifies the number of 4-bytes words in the TCP segment. Since the size of the TCP header can be 20 to 60 bytes, therefore, the value of this field can be between 5 and 15.

   If it is 5 then the length of the TCP header is 5 X 4 = 20 bytes, which is the minimum size it can have. Else if it is 15 then the size of the TCP header is 15 X 4 = 60, which is the maximum in size.

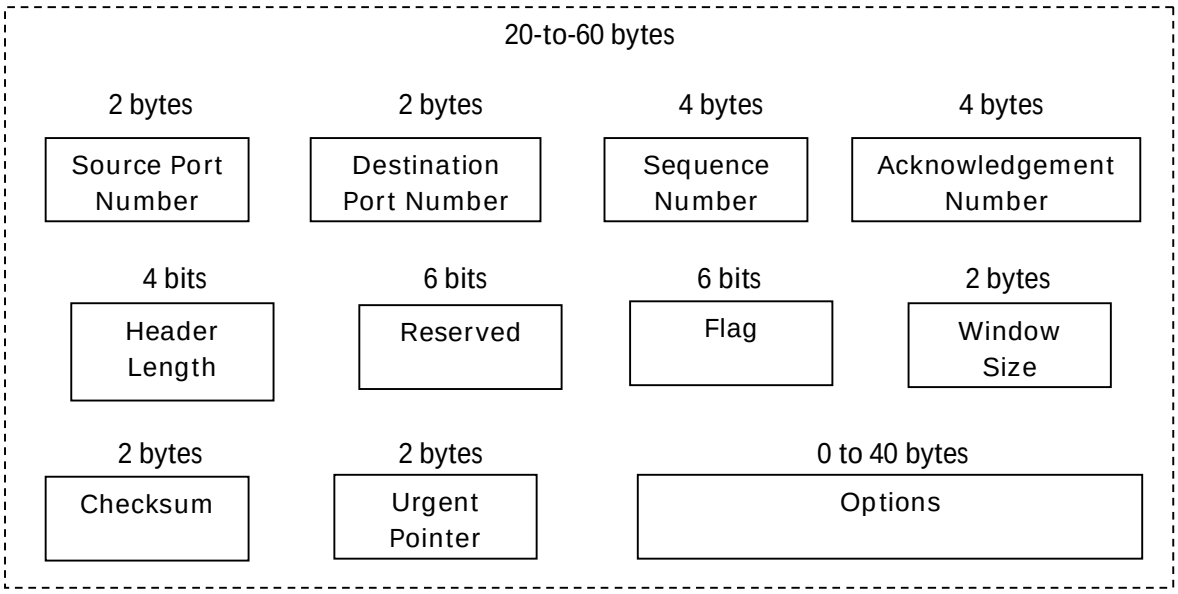


Figure 3: Layout of TCP segment header

6. **Reserved:** This 6-bits field is reserved for future use.

7. **Flag:** This 6-bits field defines 6 different control flag where each of them occupy 1 bit. They are

    a. **URG:** Set to 1 if *Urgent Pointer* is in use. It is used to indicate a byte offset from the current sequence number at which urgent data are to be found.

    b. **ACK:** Set to 1 to indicate that the *Acknowledge Number* is valid. If it is set to 0, *Acknowledgement Number* field is ignored.

    c. **PSH:** It is used to indicate *Pushed Data.* If it is s*et* to 1, it instructed the receiver to deliver the data to the appropriate application once received. Else receiver waits till the buffer is full.

    d. **RST:** Used to reset a connection that has become confused due to some reason. It is also used to reject a segment or refuse to open a connection.

    e. **SYN:** It is used to establish a connection. A request for a connection contains SYN = 1 and ACK = 0. Reply to this request bears SYN = 1 and ACK = 1.

    f. **FIN:** It is used to close a connection.

8. **Window Size:** This 2-bytes field determines the size of the sliding window that the other end must maintain.

9. **Checksum:** This 2-bytes field contains the checksum which is used for error detection and correction.

10. **Urgent Pointer:** This 2-bytes field is used to indicate that some data in a TCP segment is more urgent than others in the same connection.

6.4.3 TCP PROTOCOL OPERATION

TCP protocol operations may be divided into three phases.

Phase 1: Connections must be properly established in a multi-step handshake process **(connection establishment)** before entering phase 2.

Phase 2: The **data transfer** phase. After data transmission is completed, phase 3 is executed.

Phase 3: The **connection termination** closes established virtual circuits and releases all allocated resources.

A TCP connection is managed by an operating system through a programming interface that represents the local end-point for communications, the *Internet socket*. During the lifetime of a TCP connection the local end-point undergoes a series of state changes:

a. LISTEN: (Server) represents waiting for a connection request from any remote TCP and port.

b. SYN-SENT: (Client) represents waiting for a matching connection request after having sent a connection request.

c. SYN-RECEIVED: (Server) represents waiting for a confirming connection request acknowledgment after having both received and sent a connection request.

d. ESTABLISHED: (Both server and client) represents an open connection, data received can be delivered to the user. The normal state for the data transfer phase of the connection.

e. FIN-WAIT-1: (Both server and client) represents waiting for a connection termination request from the remote TCP, or an acknowledgment of the connection termination request previously sent.

f. FIN-WAIT-2: (Both server and client) represents waiting for a connection termination request from the remote TCP.

g. CLOSE-WAIT: (Both server and client) represents waiting for a connection termination request from the local user.

h. CLOSING: (Both server and client) represents waiting for a connection termination request acknowledgment from the remote TCP.

i. LAST-ACK: (Both server and client) represents waiting for an acknowledgment of the connection termination request previously sent to the remote TCP (which includes an acknowledgment of its connection termination request).

j. TIME-WAIT: (Either server or client) represents waiting for enough time to pass to be sure the remote TCP received the acknowledgment of its connection termination request.

k. CLOSED: (Both server and client) represents no connection state at all.

6.4.4 Connection establishment

To establish a connection, TCP uses a *three-way handshake*. Before a client attempts to connect with a server, the server must first bind to and listen at a port to open it up for connections. This is called a *passive open*. Once the passive open is established, a client may initiate an *active open*. To establish a connection, the three-way (or 3-step) handshake occurs:

1. **SYN**: The active open is performed by the client sending a SYN to the server. The client sets the segment's sequence number to a random value A.

2. **SYN-ACK**: In response, the server replies with a SYN-ACK. The acknowledgment number is set to one more than the received sequence number (A + 1), and the sequence number that the server chooses for the packet is another random number, B.

3. **ACK**: Finally, the client sends an ACK back to the server. The sequence number is set to the received acknowledgement value i.e. A + 1, and the acknowledgement number is set to one more than the received sequence number i.e. B + 1.

At this point, both the client and server have received an acknowledgment of the connection. The steps 1, 2 establish the connection parameter (sequence number) for one direction and it is acknowledged. The steps 2, 3 establish the connection parameter (sequence number) for the other direction and it is acknowledged. With these, a full-duplex communication is established.

6.4.5 CONNECTION TERMINATION

The connection termination phase uses *a four-way handshake*, with each side of the connection terminating independently. When an endpoint wishes to stop its half of the connection, it transmits a FIN packet, which the other end acknowledges with an ACK. Therefore, a typical tear-down requires a pair of FIN and ACK segments from each TCP endpoint. After both FIN/ACK exchanges are concluded, the side which sent the first FIN before receiving one waits for a timeout before finally closing the connection, during which time the local port is unavailable for new connections; this prevents confusion due to delayed packets being delivered during subsequent connections.

A connection can be "*half-open*", in which case one side has terminated its end, but the other has not. The side that has terminated can no longer send any data into the connection, but the other side can. The terminating side should continue reading the data until the other side terminates as well.

It is also possible to terminate the connection by a *3-way handshake*, when host A sends a FIN and host B replies with a FIN & ACK (merely combines 2 steps into one) and host A replies with an ACK. This is perhaps the most common method.

It is possible for both hosts to send FINs simultaneously then both just have to ACK. This could possibly be considered a 2-way handshake since the FIN/ACK sequence is done in parallel for both directions.

Some host TCP stacks may implement a half-duplex close sequence, as Linux or HP-UX do. If such a host actively closes a connection but still has not read all the incoming data the stack already received from the link, this host sends a RST instead of a FIN. This allows a TCP application to be sure the remote application has read all the data the former sent—waiting the FIN from the remote side, when it actively closes the connection. However, the remote TCP stack cannot distinguish between a *Connection Aborting RST* and this *Data Loss RST*. Both cause the remote stack to throw away all the data it received, but that the application still didn't read.

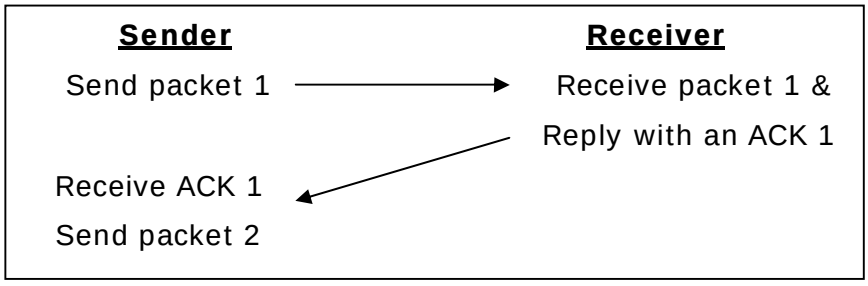
Some application protocols may violate the OSI model layers, using the TCP open/close handshaking for the application protocol open/close handshaking — these may find the RST problem on active close.

For a usual program flow like above, a TCP/IP stack like that described above does not guarantee that all the data arrives to the other application.

6.4.6 SLIDING WINDOW

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

A simple transport protocol might use the following principle: send a packet and then wait for an acknowledgment from the receiver before sending the next packet. If the ACK is not received within a certain amount of time, retransmit the packet. See Figure for more details



While this mechanism ensures reliability, it only uses a part of the available network bandwidth. Now, consider a protocol where the sender groups its packets to be transmitted, and uses the following rules:

• The sender can send all packets within the window without receiving an ACK, but must start a timeout timer for each of them.• The receiver must acknowledge each packet received, indicating the sequence number of the last well-received packet. The sender slides the window on each ACK received.

This window mechanism ensures:

a. Reliable transmission.

b. Better use of the network bandwidth (better throughput).

c. Flow-control, since the receiver may delay replying to a packet with an acknowledgment, knowing its free buffers are available and the window-size of the communication.

6.4.7 COMPARISON OF UDP AND TCP

Transmission Control Protocol is a connection-oriented protocol, which means that it requires handshaking to set up end-to-end communications. Once a connection is set up user data may be sent bi-directionally over the connection.

- *Reliable* – TCP manages message acknowledgment, retransmission and timeout. Multiple attempts to deliver the message are made. If it gets lost along the way, the server will re-request the lost part. In TCP, there's either no missing data, or, in case of multiple timeouts, the connection is dropped.

- *Ordered* – if two messages are sent over a connection in sequence, the first message will reach the receiving application first. When data segments arrive in the wrong order, TCP buffers delay the out-of-order data until all data can be properly re-ordered and delivered to the application.

- *Heavyweight* – TCP requires three packets to set up a socket connection, before any user data can be sent. TCP handles reliability and congestion control.

- *Streaming* – Data is read as a byte stream, no distinguishing indications are transmitted to signal message (segment) boundaries.

UDP is a simpler message-based connectionless protocol. Connectionless protocols do not set up a dedicated end-to-end connection. Communication is achieved by transmitting information in one direction from source to destination without verifying the readiness or state of the receiver. However, one primary benefit of UDP over TCP is the application to voice over internet protocol (VoIP) where latency and jitter are the primary concerns. It is assumed in VoIP UDP that the end users provide any necessary real time confirmation that the message has been received.

- *Unreliable* – When a message is sent, it cannot be known if it will reach its destination; it could get lost along the way. There is no concept of acknowledgment, retransmission, or timeout.

- *Not ordered* – If two messages are sent to the same recipient, the order in which they arrive cannot be predicted.

- *Lightweight* – There is no ordering of messages, no tracking connections, etc. It is a small transport layer designed on top of IP.

- *Datagrams* – Packets are sent individually and are checked for integrity only if they arrive. Packets have definite boundaries which are honored upon receipt, meaning a read operation at the receiver socket will yield an entire message as it was originally sent.

- *No congestion control* – UDP itself does not avoid congestion, and it's possible for high bandwidth applications to trigger congestion collapse, unless they implement congestion control measures at the application level.

6.5 LET US SUM UP

- The transport layer provides Message segmentation, Message acknowledgment, Message traffic control & Session multiplexing.

- This layer uses two protocols to provide services. They are Transmission Control Protocol (TCP) and User Datagram Protocol (UDP).

- This layer uses the concepts of ports and sockets to provide services to the applications running in both the host computers.

- A port is a 16-bit number. It is used by the host-to-host protocol to identify to which higher level protocol or application process it must deliver incoming messages.

- TCP is connection oriented protocol as TCP always creates a virtual connection between the two hosts which are trying to communicate with each other using three way hand shaking method.

- TCP is reliable protocol as TCP carries out different mechanisms to detect errors like packet duplicacy, packet loss, sequencing and re-sequencing etc.

- UDP is connectionless and not reliable protocol.

CHECK YOUR PROGRESS

1. Which layer of the OSI model provides error correction and flow control?

    a) Presentation

    b) Transport

    c) Network

    d) Data link

2. The addressing especially used by Transport Layer is

    a) Station address

    b) Network address

    c) Application port address

d) Dialog address

3. Both TCP and UDP belong to which layer of the OSI model?

a) Session layer

b) Transport layer

c) Network layer

d) Data Link layer

4. Which Protocol Data Unit (PDU) is employed at the Transport Layer?

a. Bits

b. Frames

c. Packets

d. Segments

5. What are the responsibilities of Transport Layer?

6. Which processes does TCP, but not UDP, use?

a. Windowing

b. Acknowledgements

c. Source Port

d. Destination Port

7. Which layer is responsible for providing mechanisms for multiplexing upper-layer application, session establishment, and tear down of virtual circuits?

a. Session

b. Network

c. Physical

d. Transport

e. Application

f. Presentation

8. Which two of the following protocols are used at the Transport layer?

a) ARP

b) UDP

c) ICMP

d)RARP

e)TCP

f) BootP

6.6 Answers to Check Your Progress

1. b
2. c
3. b
4. c
5. Write about reliability, flow control, congestion control, multiplexing, duplicacy control, message segmentation, loss control & message acknowledgement.
6. b
7. d
8. b & e

6.7 FURTHER READINGS

1. Tanenbaum A.S., Computer Network, PHI (EEE)
2. Stalling, Data and Computer Communication, PHI (EEE)
3. Stevens, UNIX Network Programming, PHI (EEE)
4. Forouzan, Data communication and Networking, TMGH

6.8 MODEL QUESTIONS

1. What is the data unit of "Transport layer"? What are the functions of a Transport Layer in OSI Model?
2. What are Ports and sockets?
3. Explain on Process-To-Process delivery.
4. What is User Datagram? Explain the application of UDP and some drawback of UDP.
5. What is the source port number?
6. What is the destination port number?
7. What is sequence number?
8. What is the acknowledgment number?
9. What is the length of the header?

10. What is the type of the segment?

9. What is Transmission control protocol?

10. What is the TCP Protocol operation?

11. Explain on Connection Establishment and connection termination in terms of Transmission control protocol.

12. What is Sliding Window?

13. What is the window size?

14. What is Checksum?

15. What are the difference between TCP and UDP?

16. What is the way to establish a TCP connection?

17. Mark one of the most important differences between TCP and UDP.

18. Which layer of OSI is responsible for end-to-end communication?

UNIT - 7: APPLICATION LAYER

UNIT STRUCTURE

7.1 LEARNING OBJECTIVES

After going through this unit, you will be able to:

- know the basics of the client server model
- get an overview of the Domain Name System
- know the DNS namespace
- learn about Name servers and their use
- know the SMTP protocol
- know about FTP

7.2 INTRODUCTION

Having finished all the preliminaries, we now come to the layer where all the Applications are found. The Application Layer is the most important and most visible layer in computer networks. Applications reside in this layer and human users interact via those applications through the network. The layers below the application layer are there to provide Transport services, but they do not do real work for users. In this chapter, we will study some real network applications. However, even in the application layer there is a need for support protocols, to allow the applications to function.

Accordingly, we will look at an important one of these before starting with the applications themselves. The item in question is DNS, which handles naming within the internet. We will learn about two very simple protocols namely SMTP and FTP which are the earlier ones to come up in the application layer.

Within the Internet, email is delivered by having the sending computer establish a TCP connection to port 25 of the receiving computer. Listening to this port is a mail server that speaks SMTP (Simple Mail Transfer Protocol). This server accepts incoming connections, subject to some security checks, and accepts messages for delivery. The FTP protocol is used to access files by FTP, the internet's file transfer protocol. FTP predates the web and has been in use for more than three decades. The web makes it easy to obtain files placed on numerous FTP servers throughout the world by providing a simple, clickable interface instead of a command-line interface. This improved access to information is one reason for the spectacular growth of the web.

7.3 CLIENT SERVER MODEL

A Client-Server model is the oldest model used to organize a networked application. In this model, a server provides services to clients that exchange information with it. This model is highly asymmetrical: clients send requests and servers perform actions and return responses. It is illustrated in the figure below.

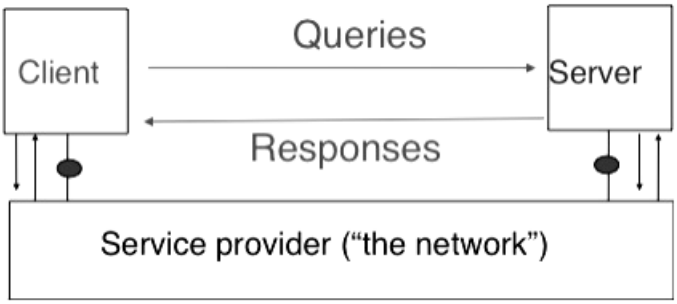


Fig 7.1: The Client-Server model

The client-server model was the first model to be used to develop networked applications. This model comes naturally from the mainframes and minicomputers that were the only networked computers used until the 1980s. A minicomputer is a multi-user system that is used by tens or more users at the same time. Each user interacts with the minicomputer by using a terminal. Those terminals were mainly a screen, a keyboard and a cable directly connected to the minicomputer. There are various types of servers as well as various types of clients. A web server provides information in response to the query sent by its clients. A print server prints documents sent as queries by the client. An Email server will forward towards their recipient the email messages sent as queries while a music server will deliver the music requested by the client. From the viewpoint of the application developer, the client and the server applications directly exchange messages (the horizontal arrows labeled queries and responses in the above

figure), but in practice these messages are exchanged thanks to the underlying layers (the vertical arrows in the above figure). Networked applications do not exchange random messages. In order to ensure that the server is able to understand the queries sent by a client, and also that the client is able to understand the responses sent by the server, they must both agree on a set of syntactical and semantic rules. These rules define the format of the messages exchanged as well as their ordering. This set of rules is called an application-level protocol. An application-level protocol is similar to a structured conversation between humans. Assume that Alice wants to know the current time but does not have a watch. If Bob passes close by, the following conversation could take place:

- Alice: Hello
- Bob: Hello
- Alice: What time is it?
- Bob: 11:55
- Alice: Thank you
- Bob: You're welcome

Such a conversation succeeds if both Alice and Bob speak the same language. If Alice meets Tchang who only speaks Chinese, she won't be able to ask him the current time. A conversation between humans can be more complex. For example, assume that Bob is a security guard whose duty is to only allow trusted secret agents to enter a meeting room. If all agents know a secret password, the conversation between Bob and Trudy could be as follows:

- Bob: What is the secret password?
- Trudy: 1234
- Bob: This is the correct password, you're welcome

If Alice wants to enter the meeting room but does not know the password, her conversation could be as follows:

- Bob: What is the secret password?
- Alice: 3.1415
- Bob: This is not the correct password.

Human conversations can be very formal, e.g. when soldiers communicate with their hierarchy, or informal such as when friends discuss. Computers that communicate are more akin to soldiers and require well-defined rules to ensure a successful exchange of information. There are two types of rules that define how information can be exchanged between computers:

- Syntactical rules that precisely define the format of the messages that are exchanged. As computers only process bits, the syntactical rules specify how information is encoded as bit strings.
- Organisation of the information flow. For many applications, the flow of information must be structured and there are precedence relationships between the different types of information. In the time example above, Alice must greet Bob before asking for the current time. Alice would not ask for the current time first and greet Bob afterwards. Such precedence relationships exist in networked applications as

well. For example, a server must receive a username and a valid password before accepting more complex commands from its clients.

7.4 DNS (DOMAIN NAME SYSTEM)

Although programs theoretically could refer to Web pages, mailboxes, and other resources by using the network (e.g., IP) addresses of the computers on which they are stored, these addresses are hard for people to remember. Also, browsing a company's Web pages from 128.111.24.41 means that if the company moves the Web server to a different machine with a different IP address, everyone needs to be told the new IP address. Consequently, high-level, readable names were introduced in order to decouple machine names from machine addresses. In this way, the company's Web server might be known as *www.cs.washington.edu* regardless of its IP address. Nevertheless, since the network itself understands only numerical addresses, some mechanism is required to convert the names to network addresses. In the following sections, we will study how this mapping is accomplished in the Internet. Way back in the ARPANET days, there was simply a file, hosts.txt, that listed all the computer names 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, well before many millions of PCs were connected to the Internet, everyone involved with it 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 importantly, host name conflicts would occur constantly unless names were centrally managed, something unthinkable in a huge international network due to the load and latency. To solve these problems, **DNS** (**Domain Name System**) was invented in 1983. It has been a key part of the Internet ever since. The essence of DNS is the invention of a hierarchical, domain-based naming scheme and a distributed database system for implementing this naming scheme. It is primarily used for mapping host names to IP addresses but can also be used for other purposes. DNS is defined in RFCs 1034, 1035, 2181, and further elaborated in many others. Very briefly, the way DNS is used is as follows. To map a name onto an IP address, an application program calls a library procedure called the resolver, passing it the name as a parameter. The resolver sends a query containing the name to a local DNS server, which looks up the name and returns a response containing the IP address to the resolver, which then returns it to the caller. The query and response messages are sent as UDP packets. Armed with the IP address, the program can then establish a TCP connection with the host or send it UDP packets.

7.4.1 THE DNS NAME SPACE

The Domain Name System is a distributed database that allows to map names on IP addresses. Managing a large and constantly changing set of names is a nontrivial problem. In the postal system, name management is done by requiring letters to specify (implicitly or explicitly) the country, state or province, city, street address, and name of the addressee. Using this kind of hierarchical addressing ensures that there is no confusion between the Marvin Anderson on Main St. in White Plains, N.Y. and the Marvin Anderson on Main St. in Austin, Texas. DNS works the same way. For the Internet, the top of the naming hierarchy is managed by an organization called **ICANN** (**Internet Corporation for Assigned Names and Numbers**). ICANN was created for this purpose in 1998, as part of the maturing of the Internet to a worldwide, economic concern. Conceptually, the Internet is divided into over 250 top-level domains, where each domain covers many hosts. Each domain is partitioned into subdomains, and these are further partitioned, and so on. All these domains can be represented by a tree, as shown in **Fig. 7.2**. The leaves of the tree represent domains that have no subdomains. A leaf domain may contain a single host, or it may represent a company and contain thousands of hosts.

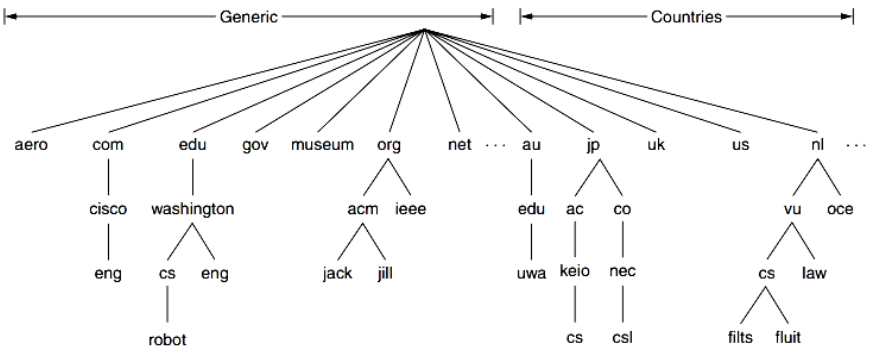


Fig 7.2: A portion of the Internet domain name space

The top-level domains come in two flavors: generic and countries. The generic domains, listed in **Fig. 7.3**, include original domains from the 1980s and domains introduced via applications to ICANN. Other generic top-level domains will be added in the future. The country domains include one entry for every country, as defined in ISO 3166. Internationalized country domain names that use non-Latin alphabets were introduced in 2010. These domains let people name hosts in Arabic, Cyrillic, Chinese, or other languages. Getting a second-level domain, such as name-of-company.com, is easy. The top-level domains are run by registrars appointed by ICANN. Getting a name merely requires going to a corresponding registrar (for com in this case) to check if the desired name is available and not somebody else's trademark. If there are no problems, the requester pays the registrar a small annual fee and gets the name.

However, as the Internet has become more commercial and more international, it has also become more contentious,

especially in matters related to naming. This controversy includes ICANN itself. For example, the creation of the *xxx* domain took several years and court cases to resolve. Is voluntarily placing adult content in its own domain a good or a bad thing? (Some people did not want adult content available at all on the Internet while others wanted to put it all in one domain so nanny filters could easily find and block it from children). Some of the domains self-organise, while others have restrictions on who can obtain a name, as noted in **Fig. 7.3**. But what restrictions are appropriate? Take the *pro* domain,

Domain	Intended use	Start date	Restricted?
com	Commercial	1985	No
edu	Educational institutions	1985	Yes
gov	Government	1985	Yes
int	International organizations	1988	Yes
mil	Military	1985	Yes
net	Network providers	1985	No
org	Non-profit organizations	1985	No
aero	Air transport	2001	Yes
biz	Businesses	2001	No
coop	Cooperatives	2001	Yes
info	Informational	2002	No
museum	Museums	2002	Yes
name	People	2002	No
pro	Professionals	2002	Yes
cat	Catalan	2005	Yes
jobs	Employment	2005	Yes
mobi	Mobile devices	2005	Yes
tel	Contact details	2005	Yes
travel	Travel industry	2005	Yes
xxx	Sex industry	2010	No

Fig 7.3: Generic top-level domains

for example. It is for qualified professionals. But who is a professional? Doctors and lawyers clearly are professionals. But what about freelance photographers, piano teachers, magicians, plumbers, barbers, exterminators, tattoo artists, and mercenaries? Are these occupations eligible? According to whom?

There is also money in names. Tuvalu (the country) sold a lease on its tv domain for $50 million, all because the country code is well-suited to advertising television sites. Virtually every common (English) word has been taken in the *com* domain, along with the most common misspellings. Try household articles, animals, plants, body parts, etc. The practice of registering a domain only to turn around and sell it off to an interested party at a much higher price even has a name. It is called cybersquatting. Many companies that were slow off the mark when the Internet era

began found their obvious domain names already taken when they tried to acquire them. In general, as long as no trademarks are being violated and no fraud is involved, it is first-come, first-served with names. Nevertheless, policies to resolve naming disputes are still being refined. Each domain is named by the path upward from it to the (unnamed) root. The components are separated by periods. Thus, the engineering department at Cisco might be *eng.cisco.com*, rather than a UNIX-style name such as */com/cisco/eng*. Notice that this hierarchical naming means that *eng.cisco.com* does not conflict with a potential use of *eng* in *eng.washington.edu*, which might be used by the English department at the University of Washington.

Domain names can be either absolute or relative. An absolute domain name always ends with a period (e.g., *eng.cisco.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*, *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.

In principle, domains can be inserted into the tree in either generic or country domains. For example, *cs.washington.edu* could equally well be listed under the *us* country domain as *cs.washington.wa.us*. In practice, however, most organizations in the United States are under generic domains, and most outside the United States are under the domain of their country. There is no rule against registering under multiple top-level domains. Large companies often do so (e.g., *sony.com*, *sony.net*, and *sony.nl*). Each domain controls how it allocates the domains under it. For example, Japan has domains *ac.jp* and *co.jp* that mirror *edu* and *com*. The Netherlands does not make this distinction and puts all organizations directly under *nl*. Thus, all three of the following are university computer science departments:

1. *cs.washington.edu* (University of Washington, in the U.S.).
2. *cs.vu.nl* (Vrije Universiteit, in The Netherlands).
3. *cs.keio.ac.jp* (Keio University, in Japan).

To create a new domain, permission is required of the domain in which it will be included. For example, if a VLSI group is started at the University of Washington and wants to be known as *vlsi.cs.washington.edu*, it has to get permission from whoever manages *cs.washington.edu*. Similarly, if a new university is chartered, say, the University of Northern South Dakota, it must ask the manager of the *edu* domain to assign it *unsd.edu* (if that is still available). In this way, name conflicts are avoided and each domain can keep track of all its subdomains. Once a new domain has been created and registered, it can create subdomains, such as *cs.unsd.edu*, without getting permission from anybody higher up the tree. Naming follows organizational boundaries, not physical networks. For example, if the computer science and electrical engineering departments are located in the same building and share the same LAN, they can nevertheless have distinct domains. Similarly, even if computer science is split over Babbage Hall and

Turing Hall, the hosts in both buildings will normally belong to the same domain.

CHECK YOUR PROGRESS

1. Fill in the blanks:

   (a) In the _____ model a server provides services to clients that exchange information with it.

   (b) The client-server model was the first model to be used to develop _____.

   (c) The DNS is primarily used for mapping _____ to _____.

   (d) For the Internet, the top of the naming hierarchy is managed by an organization called _____.

   (e) A leaf domain may contain a _____ host, or it may represent a company and contain thousands of _____.

   (f) The top-level domains come in two flavors of _____ and _____.

   (g) Domain names can be either _____ or _____.

7.4.2 DOMAIN 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. These records are the DNS database. 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 primary 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 Class Type Value

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. 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).

The third field of every resource record is the *Class*. For Internet information, it is always *IN*. For non-Internet information, other codes can be used, but in practice these are rarely seen.

The *Type* field tells what kind of record this is. There are many kinds of DNS records. The important types are listed in **Fig. 7.4**.

An SOA record provides the name of the primary source of information about the name server's zone, 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 IPv4 address of an interface for some host. The corresponding *AAAA*, or ''quadA,'' record holds a 128-bit IPv6 address. Every Internet host must have at least one IP address so that other machines can communicate with it. Some hosts have two or more network interfaces, in which case they will have two or more type *A* or *AAAA* resource records. Consequently, DNS can return multiple addresses for a single name.

A common record type is the *MX* record. It specifies the name of the host prepared to accept email for the specified domain. It is used because not every machine is prepared to accept email. If someone wants to send email to, for example, *bill@microsoft.com*, the sending host needs to find some mail server located at *microsoft.com* that is willing to accept email. The MX record can provide this information.

Another important record type is the *NS* record. It specifies a name server for the domain or subdomain. This is a host that has a copy of the database for a domain. It is used as part of the process to look up names.

Type	Meaning	Value
SOA	Start of authority	Parameters for this zone
A	IPv4 address of a host	32-Bit integer
AAAA	IPv6 address of a host	128-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
SPF	Sender policy framework	Text encoding of mail sending policy
SRV	Service	Host that provides it
TXT	Text	Descriptive ASCII text

Fig 7.4: The principal DNS resource record types

CNAME records allow aliases to be created. For example, a person familiar with Internet naming in general and wanting to send a message to user *paul* in the computer science department at M.I.T. might guess that *paul@cs.mit.edu* will work. Actually, this address will not work, because the domain for M.I.T.'s computer science department is *csail.mit.edu*. However, as a service to people who do not know this, M.I.T. could create a ***CNAME*** entry to point people and programs in the right direction. An entry like this one might do the job:

cs.mit.edu 86400 IN CNAME csail.mit.edu

Like ***CNAME***, ***PTR*** points to another name. However, unlike ***CNAME***, which is really just a macro definition (i.e., a mechanism to replace one string by another), ***PTR*** is a regular DNS data type whose interpretation depends on the context in which it is found. In practice, it is nearly always used to associate a name with an **IP** address to allow lookups of the IP address and return the name of the corresponding machine. These are called **reverse lookups**. ***SRV*** is a newer type of record that allows a host to be identified for a given service in a domain. For example, the Web server for *cs.washington.edu* could be identified as *cockatoo.cs.washington.edu*. This record generalizes the *MX* record that performs the same task but it is just for mail servers. *SPF* is also a newer type of record. It lets a domain encode information about what machines in the domain will send mail to the rest of the Internet. This helps receiving machines check that mail is valid. If mail is being received from a machine that calls itself *dodgy* but the domain records say that mail will only be sent out of the domain by a machine called *smtp*, chances are that the mail is forged junk mail. Last on the list, *TXT* records were originally provided to allow domains to identify themselves in arbitrary ways. Nowadays, they usually encode machine-readable information, typically the *SPF* information.

Finally, we have the *Value* field. This field can be a number, a domain name, or an ASCII string. The semantics depend on the record type. A short description of the *Value* fields for each of the principal record types is given in **Fig. 7.4**.

For an example of the kind of information one might find in the DNS database of a domain of **Fig. 7.5**. This figure depicts part of a (hypothetical) database for the *cs.vu.nl* domain shown in **Fig. 7.2**. The database contains seven types of resource records.

```
; Authoritative data for cs.vu.nl
cs.vu.nl.       86400   IN   SOA     star boss (9527,
cs.vu.nl.       86400   IN   MX      1 zephyr
cs.vu.nl.       86400   IN   MX      2 top
cs.vu.nl.       86400   IN   NS      star

star            86400   IN   A       130.37.56.205
zephyr          86400   IN   A       130.37.20.10
top             86400   IN   A       130.37.20.11
www             86400   IN   CNAME   star.cs.vu.nl
ftp             86400   IN   CNAME   zephyr.cs.vu.nl

flits           86400   IN   A       130.37.16.112
flits           86400   IN   A       192.31.231.165
flits           86400   IN   MX      1 flits
flits           86400   IN   MX      2 zephyr
flits           86400   IN   MX      3 top

rowboat                 IN   A       130.37.56.201
                        IN   MX      1 rowboat
                        IN   MX      2 zephyr

little-sister           IN   A       130.37.62.23

laserjet                IN   A       192.31.231.216
```

Fig 7.5: A portion of a possible DNS database for *cs.vu.nl*

The first non-comment line of **Fig. 7.5** gives some basic information about the domain, which will not concern us further. Then come two entries giving the first and second places to try to deliver email sent to *person@cs.vu.nl*. The *zephyr* (a specific machine) should be tried first. If that fails, the *top* should be tried as the next choice. The next line identifies the name server for the domain as *star*.

After the blank line (added for readability) come lines giving the IP addresses for the *star*, *zephyr*, and *top*. These are followed by an alias, *www.cs.vu.nl*, so that this address can be used without designating a specific machine. Creating this alias allows *cs.vu.nl* to change its World Wide Web server without invalidating the address people use to get to it. A similar argument holds for *ftp.cs.vu.nl*.

The section for the machine *flits* lists two IP addresses and three choices are given for handling email sent to *flits.cs.vu.nl*. First choice is naturally the *flits* itself, but if it is down, the *zephyr* and *top* are the second and third choices.

The next three lines contain a typical entry for a computer, in this case, *rowboat.cs.vu.nl*. The information provided contains the IP address and the primary and secondary mail drops. Then comes an entry for a computer that is not capable of receiving mail itself, followed by an entry that is likely for a printer that is connected to the Internet.

7.4.3 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 into non-overlapping **zones**. One possible way to divide the namespace of **Fig. 7.2** is shown in **Fig. 7.6**. Each circled zone contains some part of the tree.

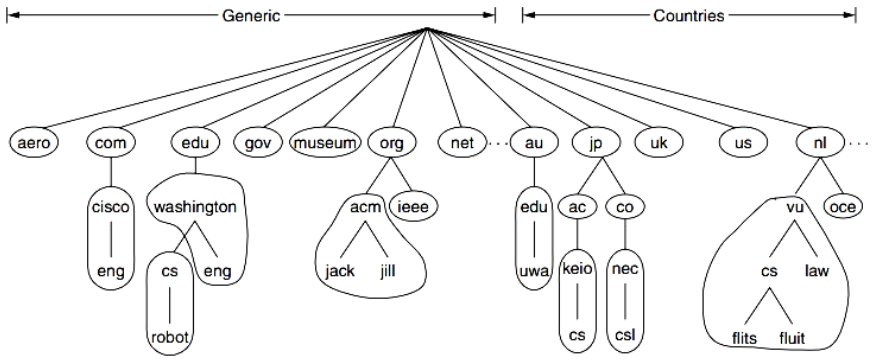


Fig 7.6: Part of the DNS name space divided into (circled) zones

Where the zone boundaries are placed within a zone is up to that zone's administrator. This decision is made in large part based on how many name servers are desired, and where. For example, in **Fig. 7.6**, the University of Washington has a zone for *washington.edu* that handles *eng.washington.edu* but does not handle *cs.washington.edu*. That 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.

Each zone is also associated with one or more name servers. These are hosts that hold the database for the 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 of the name servers can be located outside the zone. The process of looking up a name and finding an address is called **name resolution**. When a resolver has a query about a domain name, it passes the query to a local name server. If the domain being sought falls under the jurisdiction of the name server, such as *top.cs.vu.nl* falling under *cs.vu.nl*, 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. What happens when the domain is remote, such as when *flits.cs.vu.nl* wants to find the IP address of *robot.cs.washington.edu* at UW (University of Washington)? In this case, and if there is no cached information about the domain available locally, the name server begins a remote query. This query follows the process shown in

Fig. 7.7. Step 1 shows the query that is sent to the local name server. The query contains the domain name sought, the type (A), and the class (IN).

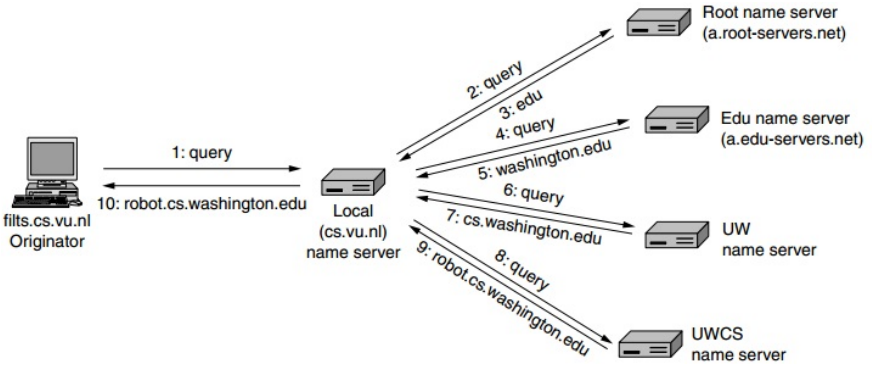


Fig 7.7: Example of a resolver looking up a remote name in 10 steps

The next step is to start at the top of the name hierarchy by asking one of the **root name servers**. These name servers have information about each top-level domain. This is shown as step 2 in **Fig.7.7**. To contact a root server, each name server must have information about one or more root name servers. This information is normally present in a system configuration file that is loaded into the DNS cache when the DNS server is started. It is simply a list of NS records for the root and the corresponding A records. There are 13 root DNS servers, unimaginatively called *a- root-servers.net* through *m.root-servers.net*. Each root server could logically be a single computer. However, since the entire Internet depends on the root servers, they are powerful and heavily replicated computers. Most of the servers are present in multiple geographical locations and reached using any cast routing, in which a packet is delivered to the nearest instance of a destination address.

The root name server is unlikely to know the address of a machine at UW, and probably does not know the name server for UW either. But it must know the name server for the *edu* domain, in which *cs.washington.edu* is located. It returns the name and IP address for that part of the answer in step 3. The local name server then continues its quest. It sends the entire query to the *edu* name server (*a.edu-servers.net*). That name server returns the name server for UW. This is shown in steps 4 and 5. Closer now, the local name server sends the query to the UW name server (step 6). If the domain name being sought was in the English department, the answer would be found, as the UW zone includes the English department. But the Computer Science department has chosen to run its own name server. The query returns the name and IP address of the UW Computer Science name server (step 7).

Finally, the local name server queries the UW Computer Science name server (step 8). This server is authoritative for the domain *cs.washington.edu*, so it must have the answer. It returns the final answer (step 9), which the local name server forwards as a response to flits.cs.vu.nl (step 10). The name has been resolved.

You can explore this process using standard tools such as the *dig* program that is installed on most UNIX systems. For example, typing

dig@a.edu-servers.net robot.cs.washington.edu

will send a query for *robot.cs.washington.edu* to the *a.edu-servers.net* name server and print out the result. This will show you the information obtained in step 4 in the example above, and you will learn the name and IP address of the UW name servers.

There are three technical points to discuss about this long scenario. First, two different query mechanisms are at work in **Fig. 7.7**. When the host *flits.cs.vu.nl* sends its query to the local name server, that name server handles the resolution on behalf of *flits* until it has the desired answer to return. It does not return partial answers. They might be helpful, but they are not what the query was seeking. This mechanism is called a **recursive query**.

On the other hand, the root name server (and each subsequent name server) does not recursively continue the query for the local name server. It just returns a partial answer and moves on to the next query. The local name server is responsible for continuing the resolution by issuing further queries. This mechanism is called an **iterative query**. One name resolution can involve both mechanisms, as this example showed. A recursive query may always seem preferable, but many name servers (especially the root) will not handle them. They are too busy. Iterative queries put the burden on the originator. The rationale for the local name server supporting a recursive query is that it is providing a service to hosts in its domain. Those hosts do not have to be configured to run a full name server, just to reach the local one.

The second point is caching. All of the answers, including all the partial answers returned, are cached. In this way, if another *cs.vu.nl* host queries for *robot.cs.washington.edu* the answer will already be known. Even better, if a host queries for a different host in the same domain, say *galah.cs.washington.edu*, the query can be sent directly to the authoritative name server. Similarly, queries for other domains in *washington.edu* can start directly from the *washington.edu* name server. Using cached answers greatly reduces the steps in a query and improves performance. The original scenario we sketched is in fact the worst case that occurs when no useful information is cached. However, cached answers are not authoritative, since changes made at *cs.washington.edu* will not be propagated to all the caches in the world that may know about it. For this reason, cache entries should not live too long. This is the reason that the *Time to live* field is included in each resource record. It tells remote name servers how long to cache records. If a certain machine has had the same IP address for years, it may be safe to cache that information for 1 day. For more volatile information, it might be safer to purge the records after a few seconds or a minute.

The third issue is the transport protocol that is used for the queries and responses. It is UDP. DNS messages are sent in UDP packets with a simple format for queries, answers, and name servers that can be used to continue the resolution. We will not go into the details of this format. If no response arrives within a short

time, the DNS client repeats the query, trying another server for the domain after a small number of retries. This process is designed to handle the case of the server being down as well as the query or response packet getting lost. A 16-bit identifier is included in each query and copied to the response so that a name server can match answers to the corresponding query, even if multiple queries are outstanding at the same time. Even though its purpose is simple, it should be clear that DNS is a large and complex distributed system that is comprised of millions of name servers that work together. It forms a key link between human-readable domain names and the IP addresses of machines. It includes replication and caching for performance and reliability and is designed to be highly robust.

There is also application demand to use names in more flexible ways, for example, by naming content and resolving to the IP address of a nearby host that has the content. This fits the model of searching for and downloading a movie. It is the movie that matters, not the computer that has a copy of it, so all that is wanted is the IP address of any nearby computer that has a copy of the movie. Content distribution networks are one way to accomplish this mapping.

7.5 SMTP–SIMPLE MAIL TRANSFER PROTOCOL

Simple Mail Transfer Protocol (SMTP) is an Internet standard for electronic mail (e-mail) transmission across Internet Protocol (IP) networks. SMTP was first defined by RFC 821 (1982, eventually declared STD 10), and last updated by RFC 5321 (2008) which includes the Extended SMTP (ESMTP) additions, and is the protocol in widespread use today. SMTP uses TCP port 25. The protocol for new submissions (MSA) is effectively the same as SMTP, but it uses port 587 instead. SMTP connections secured by SSL are known by the shorthand SMTPS, though SMTPS is not a protocol in its own right. While electronic mail servers and other mail transfer agents use SMTP to send and receive mail messages, user-level client mail applications typically use SMTP only for sending messages to a mail server for relaying. For receiving messages, client applications usually use either the Post Office Protocol (POP) or the Internet Message Access Protocol (IMAP) or a proprietary system (such as Microsoft Exchange or Lotus Notes/Domino) to access their mail box accounts on a mail server. Various forms of one-to-one electronic messaging were used in the 1960s. People communicated with one another using systems developed for specific mainframe computers. As more computers were interconnected, especially in the US Government's ARPANET, standards were developed to allow users of different systems to e-mail one another. SMTP grew out of these standards developed during the 1970s. SMTP can trace its roots to two implementations described in 1971: the Mail Box Protocol, whose implementation has been disputed, but is discussed in RFC 196 and other RFCs, and the SNDMSG program, which, according to RFC 2235, Ray Tomlinson of BBN invented for TENEX computers to send mail messages across the

ARPANET. Fewer than 50 hosts were connected to the ARPANET at this time. Further implementations include FTP Mail and Mail Protocol, both from 1973. Development work continued throughout the 1970s, until the ARPANET converted into the modern Internet around 1980. Jon Postel then proposed a Mail Transfer Protocol in 1980 that began to remove the mail's reliance on FTP. SMTP was published as RFC 788 in November 1981, also by Postel. The SMTP standard was developed around the same time as Usenet, a one-to-many communication network with some similarities. SMTP became widely used in the early 1980s. At the time, it was a complement to Unix to Unix Copy Program (UUCP) mail, which was better suited for handling e-mail transfers between machines that were intermittently connected. SMTP, on the other hand, works best when both the sending and receiving machines are connected to the network all the time. Both use a store and forward mechanism and are examples of push technology. Though Usenet's newsgroups are still propagated with UUCP between servers, UUCP mail has virtually disappeared along with the "bang paths" it used as message routing headers.
Released with 4.1cBSD, right after RFC 788, Sendmail was one of the first (if not the first) mail transfer agents to implement SMTP. Over time, as BSD Unix became the most popular operating system on the Internet, sendmail became the most common MTA (mail transfer agent). Some other popular SMTP server programs include Postfix, qmail, Novell GroupWise, Exim, Novell NetMail, Microsoft Exchange Server, Sun Java System Messaging Server. Message submission (RFC 2476) and SMTP-AUTH (RFC 2554) were introduced in 1998 and 1999, both describing new trends in e-mail delivery. Originally, SMTP servers were typically internal to an organization, receiving mail for the organization from the outside, and relaying messages from the organization to the outside. But as time went on, SMTP servers (mail transfer agents), in practice, were expanding their roles to become message submission agents for Mail user agents, some of which were now relaying mail from the outside of an organization. This issue, a consequence of the rapid expansion and popularity of the World Wide Web, meant that SMTP had to include specific rules and methods for relaying mail and authenticating users to prevent abuses such as relaying of unsolicited e-mail (spam). Work on message submission (RFC 2476) was originally started because popular mail servers would often rewrite mail in an attempt to fix problems in it, for example, adding a domain name to an unqualified address. This behavior is helpful when the message being fixed is an initial submission, but dangerous and harmful when the message originated elsewhere and is being relayed. Cleanly separating mail into submission and relay was seen as a way to permit and encourage rewriting submissions while prohibiting rewriting relay. As spam became more prevalent, it was also seen as a way to provide authorization for mail being sent out from an organization, as well as traceability. This separation of relay and submission quickly became a foundation for modern email security practices.

As this protocol started out purely as ASCII text-based, it did not deal well with binary files, or characters in many non-English languages. Standards such as **Multipurpose Internet**

Mail Extensions (MIME) were developed to encode binary files for transfer through SMTP. Mail transfer agents (MTAs) developed after Sendmail also tended to be implemented 8-bit-clean, so that the alternate "just send eight" strategy could be used to transmit arbitrary text data (in any 8-bit ASCII-like character encoding) via SMTP. Mojibake was still a problem due to differing character set mappings between vendors, although the email addresses themselves still allowed only ASCII. 8-bit-clean MTAs today tend to support the 8 BITMIME extension, permitting binary files to be transmitted almost as easily as plain text. Recently the SMTPUTF8 extension was created to support UTF-8 text, allowing international content and addresses in non-Latin scripts like Cyrillic or Chines.

SMTP V/S Mail Retrieval

SMTP is a delivery protocol only. In normal use, mail is "pushed" to a destination mail server (or next-hop mail server) as it arrives. Mail is routed based on the destination server, not the individual user(s) to which it is addressed. Other protocols, such as the Post Office Protocol (POP) and the Internet Message Access Protocol (IMAP) are specifically designed for use by individual users retrieving messages and managing mail boxes. To permit an intermittently-connected mail server to pull messages from a remote server on demand, SMTP has a feature to initiate mail queue processing on a remote server. POP and IMAP are unsuitable protocols for relaying mail by intermittently-connected machines; they are designed to operate after final delivery, when information critical to the correct operation of mail relay (the "mail envelope") has been removed.

SMTP transport example

A typical example of sending a message via SMTP to two mailboxes (alice and theboss) located in the same mail domain (example.com or localhost.com) is reproduced in the following session exchange. (In this example, the conversation parts are prefixed with S: and C:, for server and client, respectively; these labels are not part of the exchange.) After the message sender (SMTP client) establishes a reliable communications channel to the message receiver (SMTP server), the session is opened with a greeting by the server, usually containing its fully qualified domain name (FQDN), in this case *smtp.example.com*. The client initiates its dialog by responding with a HELO command identifying itself in the command's parameter with its FQDN (or an address literal if none is available).

```
S:   220 smtp.example.com ESMTP Postfix
C:   HELO relay.example.org
S:   250 Hello relay.example.org, I am glad to meet you
C:   MAIL FROM:<bob@example.org>
S:   250 Ok
C:   RCPT TO:<alice@example.com>
S:   250 Ok
C:   RCPT TO:<theboss@example.com>
S:   250 Ok
C:   DATA
S:   354 End data with <CR><LF>.<CR><LF>
```

```
C:   From: "Bob Example" <bob@example.org>
C:   To: "Alice Example" <alice@example.com>
C:   Cc: theboss@example.com
C:   Date: Tue, 15 January 2008 16:02:43 -0500
C:   Subject: Test message
C:
C:   Hello Alice.
C:   This is a test message with 5 header fields and 4
     lines in the message body.
C:   Your friend,
C:   Bob
C:   .
S:   250 Ok: queued as 12345
C:   QUIT
S:   221 Bye
{The server closes the connection}
```

The client notifies the receiver of the originating email address of the message in a **MAIL FROM** command. In this example, the email message is sent to two mailboxes on the same SMTP server: one each for each recipient listed in the **To** and **Cc** header fields. The corresponding SMTP command is **RCPT TO**. Each successful reception and execution of a command is acknowledged by the server with a result code and response message (e.g., 250 Ok). The transmission of the body of the mail message is initiated with a DATA command after which it is transmitted verbatim line by line and is terminated with an end-of-data sequence. This sequence consists of a new-line (<CR><LF>), a single full stop (period), followed by another new-line. Since a message body can contain a line with just a period as part of the text, the client sends two periods every time a line starts with a period; correspondingly, the server replaces every sequence of two periods at the beginning of a line with a single one. Such escaping method is called **dot-stuffing**. The server's positive reply to the end-of-data, as exemplified, implies that the server has taken the responsibility of delivering the message. A message can be doubled if there is a communication failure at this time, e.g. due to a power shortage: Until the sender has received that 250 reply, it must assume the message was not delivered. On the other hand, after the receiver has decided to accept the message, it must assume the message has been delivered to it. Thus, during this time span, both agents have active copies of the message that they will try to deliver. The probability that a communication failure occurs exactly at this step is directly proportional to the amount of filtering that the server performs on the message body, most often for anti-spam purposes. The limiting timeout is specified to be 10 minutes. The QUIT command ends the session. If the email has other recipients located elsewhere, the client would QUIT and connect to an appropriate SMTP server for subsequent recipients after the current destination(s) had been queued. The information that the client sends in the HELO and MAIL FROM commands are added as additional header fields to the message by the receiving server. It adds a Received and Return-Path header field, respectively. Some clients are implemented to close the connection after the message is accepted (250 Ok: queued as 12345), so the last two lines may actually be omitted. This causes an error on the server when trying to send the 221 reply.

7.6 FTP-FILE TRANSFER PROTOCOL

File Transfer Protocol (FTP) is a standard network protocol used to transfer files from one host to another host over a TCP-based network, such as the Internet. FTP is built on a client-server architecture and uses separate control and data connections between the client and the server. FTP users may authenticate themselves using a clear-text sign-in protocol, normally in the form of a username and password, but can connect anonymously if the server is configured to allow it. For secure transmission that hides (encrypts) the username and password, and encrypts the content, FTP is often secured with SSL/TLS ("FTPS"). SSH File Transfer Protocol ("SFTP") is sometimes also used instead, but is technologically different. The first FTP client applications were command-line applications developed before operating systems had graphical user interfaces, and are still shipped with most Windows, Unix, and Linux operating systems. Dozens of FTP clients and automation utilities have since been developed for desktops, servers, mobile devices, and hardware, and FTP has been incorporated into hundreds of productivity applications, such as Web page editors.

Communication and data transfer

FTP may run in active or passive mode, which determines how the data connection is established. In active mode, the client creates a TCP control connection. In situations where the client is behind a firewall and unable to accept incoming TCP connections, passive mode may be used. In this mode, the client uses the control connection to send a PASV command to the server and then receives a server IP address and server port number from the server, which the client then uses to open a data connection from an arbitrary client port to the server IP address and server port number received. Both modes were updated in September 1998 to support IPv6. Further changes were introduced to the passive mode at that time, updating it to extended passive mode. The server responds over the control connection with three-digit status codes in ASCII with an optional text message. For example "200" (or "200 OK") means that the last command was successful. The numbers represent the code for the response and the optional text represents a human-readable explanation or request (e.g. <Need account for storing file>). An ongoing transfer of file data over the data connection can be aborted using an interrupt message sent over the control connection.

While transferring data over the network, four data representations can be used:

- **ASCII mode**: used for text. Data is converted, if needed, from the sending host's character representation to "8-bit ASCII" before transmission, and (again, if necessary) to the receiving host's character representation. As a consequence, this mode is inappropriate for files that contain data other than plain text.
- **Image mode** (commonly called Binary mode): the sending machine sends each file byte for byte, and the recipient

stores the bytestream as it receives it. (Image mode support has been recommended for all implementations of FTP).

- **EBCDIC mode**: use for plain text between hosts using the EBCDIC character set. This mode is otherwise like ASCII mode.
- **Local mode**: Allows two computers with identical setups to send data in a proprietary format without the need to convert it to ASCII

For text files, different format control and record structure options are provided. These features were designed to facilitate files containing Telnet or ASA.

Data transfer can be done in any of three modes:

- **Stream mode**: Data is sent as a continuous stream, relieving FTP from doing any processing. Rather, all processing is left up to TCP. No End-of-file indicator is needed, unless the data is divided into records.
- **Block mode**: FTP breaks the data into several blocks (block header, byte count, and data field) and then passes it on to TCP.
- **Compressed mode**: Data is compressed using a single algorithm.

CHECK YOUR PROGRESS

2. Fill in the blanks:

   (a) The primary function of DNS is to map _____ onto _____.

   (b) The process of looking up a name and finding an address is called _____.

   (c) DNS messages are sent in _____ packets with a simple format for queries.

   (d) SMTP connections secured by SSL are known by the shorthand _____.

   (e) _____ has a feature to initiate mail queue processing on a remote server

   (f) _____ is built on a client-server architecture and uses separate _____ and _____ between the client and the server.

   (g) _____ users may authenticate themselves using a clear-text sign-in _____, normally in the form of a username and password.

4.9 LET US SUM UP

- The ***Application Layer*** is the most important and most visible layer in computer networks.
- A ***Client-Server model*** is the oldest model used to organize a networked application.
- The ***Domain Name System*** is a distributed database that allows to map names on IP addresses.
- The essence of ***DNS*** is the invention of a hierarchical, domain-based naming scheme and a distributed database system for implementing this naming scheme.
- Every domain, whether it is a single host or a top-level domain, can have a set of ***resource records*** associated with it.
- The ***DNS*** name space is divided into non-overlapping zones.
- ***Simple Mail Transfer Protocol*** is an Internet standard for electronic mail transmission across Internet Protocol networks.
- ***File Transfer Protocol*** is a standard network protocol used to transfer files from one host to another host over a TCP-based network.
- ***Multipurpose Internet Mail Extensions*** (MIME) were developed to encode binary files for transfer through SMTP.
- ***FTP*** may run in active or passive mode, which determines how the data connection is established.

4.10 ANSWERS TO CHECK YOUR PROGRESS

1.
   (a) Client Server.
   (b) networked applications.
   (c) host names, IP addresses.
   (d) Internet Corporation for Assigned Names and Numbers.
   (e) single, hosts.
   (f) generic, countries.
   (g) absolute, relative.

2.
   (a) domain names, resource records
   (b) name resolution.
   (c) UDP.
   (d) SMTPS.
   (e) SMTP.
   (f) FTP, control, data connections.
   (g) FTP, protocol.

4.11 FURTHER READINGS

Computer Networks

- Andrew S. Tanenbaum, David J. Wetherall

  PRENTICE HALL

Computer Networking: Principles, Protocols and Practice

- Olivier Bonaventure

4.11 MODEL QUESTIONS

1. Explain the concept of a Client Server model.
2. What is the Domain Name System? Explain its significance.
3. Explain the DNS Namespace with diagram.
4. What are Domain Resource Records? Explain their usefulness.
5. Describe the concept of Name servers.

6. Explain the concept of the Simple Mail Transfer Protocol giving suitable examples.

7. What is the File Transfer Protocol? Describe its application.