**1. Define**Network?

**Ans**.A network is the interconnection of a set of devices capable of communication. In this definition, a device can be a host (or an *end system* as it is sometimes called) such as a large computer, desktop, laptop, workstation, cellular phone, or security system. A device in this definition can also be a connecting device such as a router, which connects the network to other networks, a switch, which connects devices together, a modem (modulator-demodulator), which changes the form of data, and so on. These devices in a network are connected using wired or wireless transmission media such as cable or air. When we connect two computers at home using a plug-and-play router, we have created a network, although very small.

2. **State**the goals of networks?

Ans.The goals of networks are:

1. Performance

2. Reliability and

3. Security.

***Performance***

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

***Reliability***

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

***Security***

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

3. **Explain**different types of connections in a network?

***Type of Connection***

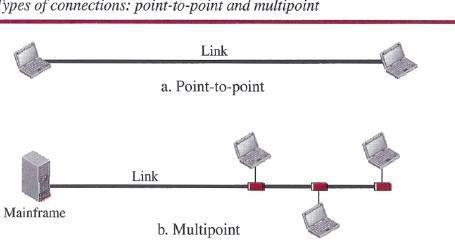
A network is two or more devices connected through links. A link is a communications pathway that transfers data from one device to another. For visualization purposes, it is simplest to imagine any link as a line drawn between two points. For communication to occur, two devices must be connected in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

***Point-to-Point***

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

***Multipoint***

A multipoint (also called multidrop) connection is one in which more than two specific devices share a single link.



In a multipoint environment, the capacity of the channel is shared, either spatially or temporally. If several devices can use the link simultaneously, it is a *spatially shared* connection. If users must take turns, it is a *timeshared* connection.

4. **Lis**t two advantages of layering principle in computer networks?

**Principles of Protocol Layering**

Let us discuss two principles of protocol layering.

***First Principle***

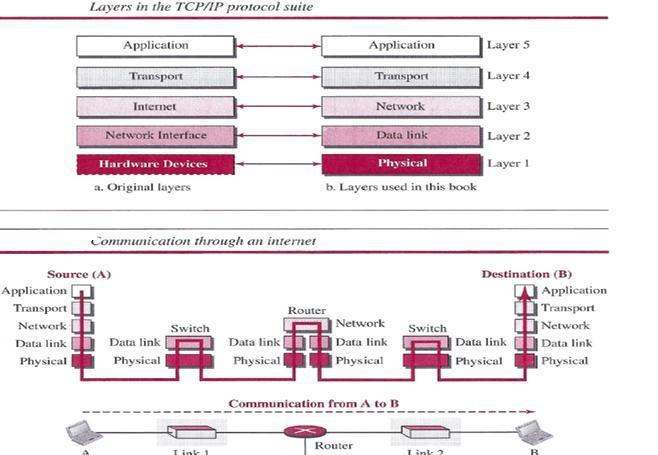
The first principle dictates that if we want bidirectional communication, we need to make each layer so that it is able to perform two opposite tasks, one in each direction. For example, the third layer task is to listen (in one direction) and *talk* (in the other direction). The second layer needs to be able to encrypt and decrypt. The first layer needs to send and receive mail.

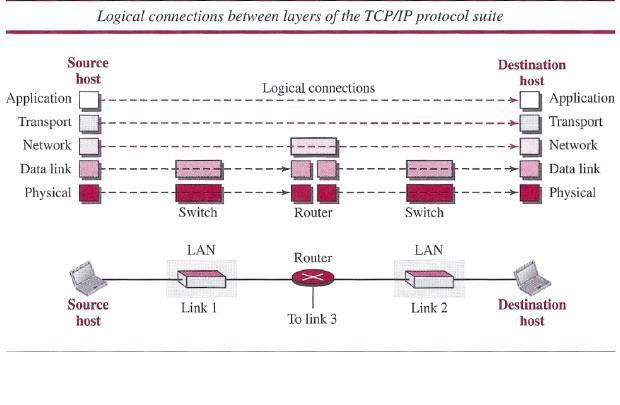
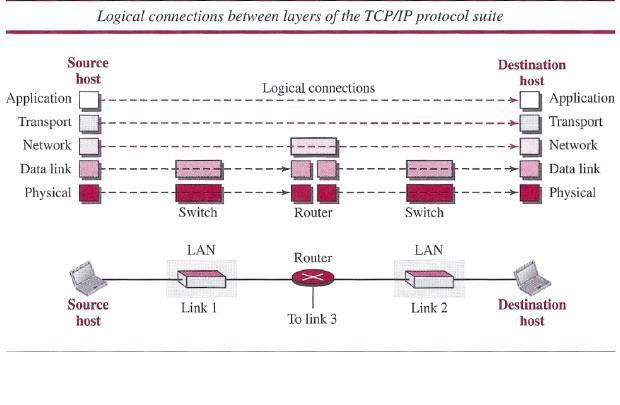
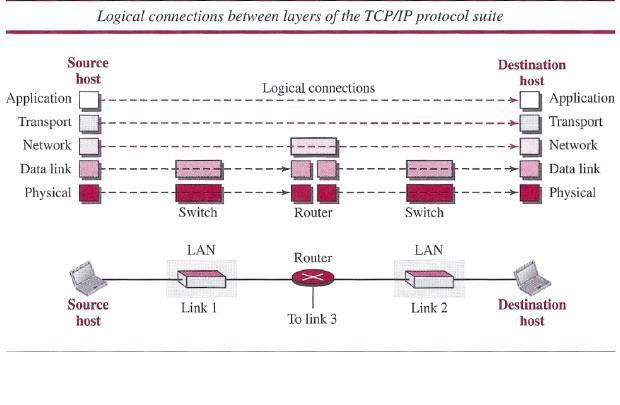
***Second Principle***

The second principle that we need to follow in protocol layering is that the two objects under each layer at both sites should be identical. For example, the object under layer 3 at both sites should be a plaintext letter. both sites should be a cipher text letter. The object under layer 1 at both sites should be a piece of mail.

5. What are Logical connections.

Ans.The logical connection between each layer as shown in below figure. This means that we have layer-to-layer communication. Maria and Ann can think that there is a logical (imaginary) connection at each layer through which they can send the object created from that layer. We will see that the concept of logical connection will help us better understand the task of layering. We encounter in data communication and networking.

6. **Classify**different types of Layers in TCP/IP protocol suite?



7. **Define**the responsibilities of data link layer?

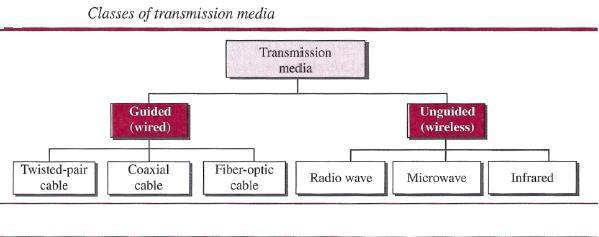
We have seen that an internet is made up of several links (LANs and WANs) connected by routers. There may be several overlapping sets of links that a datagram can travel from the hostto the destination. The routers are responsible for choosing the *best* links. However, when the next link to travel is determined by the router, the data-link layer is responsible for taking the datagram and moving it across the link. The link can be a wired LAN with a link-layer switch, a wireless LAN, a wired WAN, or a wireless WAN. We can also have different protocols used with any link type. In each case, the data-link layer is responsible for moving the packet through the link. *TCP/IP* does not define any specific protocol for the data-link layer. It supports all

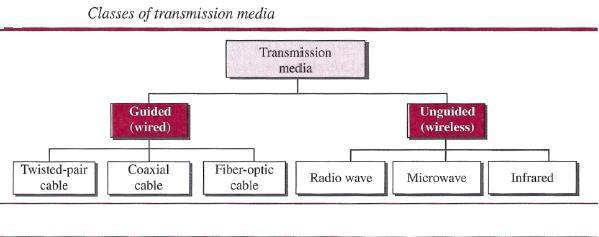
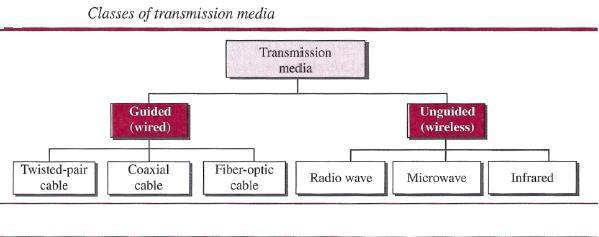
the standard and proprietary protocols. Any protocol that can take the datagram and carry it through the link suffices for the network layer. The data-link layer takes a datagram and encapsulates it in a packet called *«frame.* Each link-layer protocol may provide a different service. Some link-layer protocols provide complete error detection and correction, some provide only error correction.

8. **Explain**the role of ARPANET in computer networks?

Ans.In the mid-1960s, mainframe computers in research organizations were stand-alone devices. Computers from different manufacturers were unable to communicate with one another. The Advanced Research Projects Agency (ARPA) in the Department of Defense (DOD) was interested in finding a way to connect computers so that the researchers they funded could share their findings, thereby reducing costs and eliminating duplication of effort. In 1967, at an Association for Computing Machinery (ACM) meeting, ARPA presented its ideas for the Advanced Research Projects Agency Network (ARPANET), a small network of connected computers. The idea was that each host computer (not necessarily from the same manufacturer) would be attached to a specialized computer, called an *interface message processor* (IMP). The IMPs, in turn, would be connected to each other. Each IMP had to be able to communicate with other IMPs as well as with its own attached host

9. **List**different types of Transmission Media?





10.What are advantages of optical fiber?

***Advantages***

Fiber-optic cable has several advantages over metallic cable (twisted-pair or coaxial).

* Higher bandwidth. Fiber-optic cable can support dramatically higher bandwidths (and hence data rates) than either twisted-pair or coaxial cable. Currently, data rates and bandwidth utilization over fiber-optic cable are limited not by the medium but by the signal generation and reception technology available.
* Less signal attenuation. Fiber-optic transmission distance is significantly greater than that of other guided media. A signal can run for 50 km without requiring regeneration. We need repeaters every 5 km for coaxial or twisted-pair cable.
* D Immunity to electromagnetic interference. Electromagnetic noise cannot affect fiber-optic cables.
* D Resistance to corrosive materials. Glass is more resistant to corrosive materials than copper.
* Light weight. Fiber-optic cables are much lighter than copper cables.
* Greater immunity to tapping. Fiber-optic cables are more immune to tapping than copper cables. Copper cables create antenna effects that can easily be tapped.

11. What are disadvantages of optical fiber?

Ans.

***Disadvantages***

There are some disadvantages in the use of optical fiber.

* Installation and maintenance. Fiber-optic cable is a relatively new technology. Its installation and maintenance require expertise that is not yet available everywhere. o Unidirectional light propagation. Propagation of light is unidirectional. If we need bidirectional communication, two fibers are needed.
* Cost. The cable and the interfaces are relatively more expensive than those of other guided media. If the demand for bandwidth is not high, often the use of optical fiber cannot be justified.

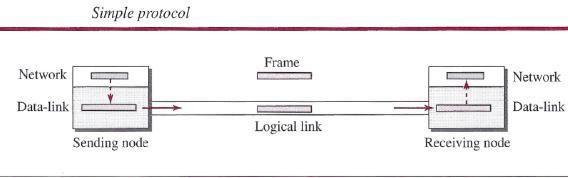
12.**Define** cyclic codes.

Ans.Cyclic codes are special linear block codes with one extra property. In a cyclic code, if a codeword is cyclically shifted (rotated), the result is another codeword. For example, if 1011000 is a codeword and we cyclically left-shift, then 0110001 is also a codeword. In this case, if we call the bits in the first word *ao* to *a6,* and the bits in the second word *bo* to *b6,* we can shift the bits by using the following: In the rightmost equation, the last bit of the first word is wrapped around and becomes the first bit of the second word.

13.**Explain** simple protocol.

Ans.

Our first protocol is a simple protocol with neither flow nor error control. We assume that the receiver can immediately handle any frame it receives. In other words, the receiver can never be overwhelmed with incoming frames. Below figure shows the layout for this protocol.



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

14.**Define** PPP?

Ans.One of the most common protocols for point-to-point access is the **Point-to-Point Protocol** **(PPP).** Today, millions of Internet users who need to connect their home computers to the serverof an Internet service provider use PPP. The majority of these users have a traditional modem; they are connected to the Internet through a telephone line, which provides the services of the physical layer. But to control and manage the transfer of data, there is a need for a point-to-point protocol at the data-link layer.

**15.What are Services provided byPPP?**

**Ans.Services**

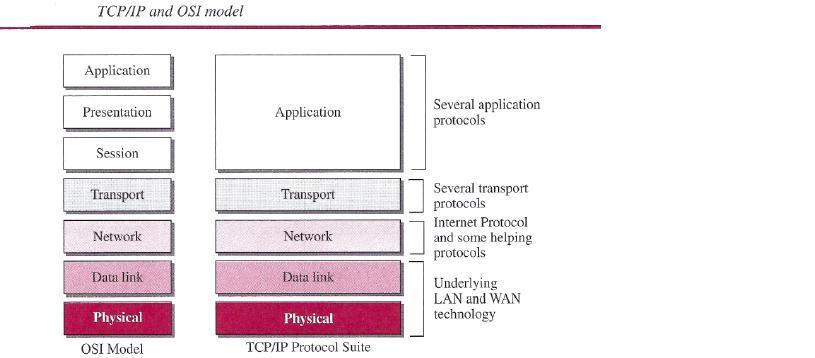
The designers of PPP have included several services to make it suitable for a point-to point protocol, but have ignored some traditional services to make it simple.

***Services Provided by PPP***

PPP defines the format of the frame to be exchanged between devices. It also defines how two devices can negotiate the establishment of the link and the exchange of data. PPP is designed to accept payloads from several network layers (not only IP). Authentication is also provided in the protocol, but it is optional. The new version of PPP, called *Multilink PPP,* provides connections over multiple links. One interesting feature of PPP is that it provides network address configuration. This is particularly useful when a home user needs a temporary network address to connect to the Internet.

16. **COMPARISION OF OSI AND TCP/IP REFERENCE MODEL**

Ans.When we compare the two models, we find that two layers, session and presentation, are missing from the *TCP/IP* protocol suite. These two layers were not added to the *TCP/IP* protocol suite after the publication of the OSI model. The application layer in the suite is usually considered to be the combination of three layers in the OSI model.



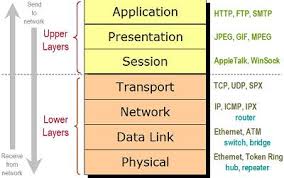
Two reasons were mentioned for this decision. First, *TCP/IP* has more than one transport-layer protocol. Some of the functionalities of the session layer are available in some of the transport-layer protocols. Second, the application layer is not only one piece of software. Many Applications can be developed at this layer. If some of the functionalities mentioned in the session and presentation layers are needed for a particular application, they can be included in the development of that piece of software.

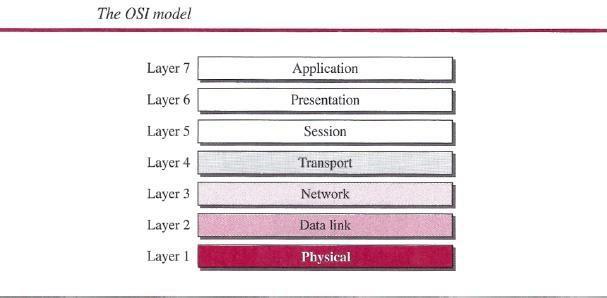
**Long Answer Questions**

1. **Explain**ISO/OSI Reference model with neat diagram?

The Open Systems Interconnect (OSI) model has seven layers. This article describes and explains them, beginning with the 'lowest' in the hierarchy (the physical) and proceeding to the 'highest' (the application). The layers are stacked this way:

* Application
* Presentation
* Session
* Transport
* Network
* Data Link
* Physical





PHYSICAL LAYER

The physical layer, the lowest layer of the OSI model, is concerned with the transmission and reception of the unstructured raw bit stream over a physical medium. It describes the electrical/optical, mechanical, and functional interfaces to the physical medium, and carries the signals for all of the higher layers. It provides:

* Data encoding: modifies the simple digital signal pattern (1s and 0s) used by the PC to better accommodate the characteristics of the physical medium, and to aid in bit and frame synchronization. It determines:  
    
  + What signal state represents a binary 1
  + How the receiving station knows when a "bit-time" starts
  + How the receiving station delimits a frame
* Physical medium attachment, accommodating various possibilities in the medium:  
    
  + Will an external transceiver (MAU) be used to connect to the medium?
  + How many pins do the connectors have and what is each pin used for?
* Transmission technique: determines whether the encoded bits will be transmitted by baseband (digital) or broadband (analog) signaling.
* Physical medium transmission: transmits bits as electrical or optical signals appropriate for the physical medium, and determines:  
    
  + What physical medium options can be used
  + How many volts/db should be used to represent a given signal state, using a given physical medium

DATA LINK LAYER

The data link layer provides error-free transfer of data frames from one node to another over the physical layer, allowing layers above it to assume virtually error-free transmission over the link. To do this, the data link layer provides:

* Link establishment and termination: establishes and terminates the logical link between two nodes.
* Frame traffic control: tells the transmitting node to "back-off" when no frame buffers are available.
* Frame sequencing: transmits/receives frames sequentially.
* Frame acknowledgment: provides/expects frame acknowledgments. Detects and recovers from errors that occur in the physical layer by retransmitting non-acknowledged frames and handling duplicate frame receipt.
* Frame delimiting: creates and recognizes frame boundaries.
* Frame error checking: checks received frames for integrity.
* Media access management: determines when the node "has the right" to use the physical medium.

NETWORK LAYER

The network layer controls the operation of the subnet, deciding which physical path the data should take based on network conditions, priority of service, and other factors. It provides: 

* Routing: routes frames among networks.
* Subnet traffic control: routers (network layer intermediate systems) can instruct a sending station to "throttle back" its frame transmission when the router's buffer fills up.
* Frame fragmentation: if it determines that a downstream router's maximum transmission unit (MTU) size is less than the frame size, a router can fragment a frame for transmission and re-assembly at the destination station.
* Logical-physical address mapping: translates logical addresses, or names, into physical addresses.
* Subnet usage accounting: has accounting functions to keep track of frames forwarded by subnet intermediate systems, to produce billing information.

Communications Subnet

The network layer software must build headers so that the network layer software residing in the subnet intermediate systems can recognize them and use them to route data to the destination address.   
  
This layer relieves the upper layers of the need to know anything about the data transmission and intermediate switching technologies used to connect systems. It establishes, maintains and terminates connections across the intervening communications facility (one or several intermediate systems in the communication subnet).   
  
In the network layer and the layers below, peer protocols exist between a node and its immediate neighbor, but the neighbor may be a node through which data is routed, not the destination station. The source and destination stations may be separated by many intermediate systems.

TRANSPORT LAYER

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 datagrams, the transport protocol should include extensive error detection and recovery.   
  
The transport layer provides:

* Message segmentation: accepts a message from the (session) layer above it, splits the message into smaller units (if not already small enough), and passes the smaller units down to the network layer. The transport layer at the destination station reassembles the message.
* Message acknowledgment: provides reliable end-to-end message delivery with acknowledgments.
* Message traffic control: tells the transmitting station to "back-off" when no message buffers are available.
* Session multiplexing: multiplexes several message streams, or sessions onto one logical link and keeps track of which messages belong to which sessions (see session layer).

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, prepending a header to each frame.   
  
The transport layer header information must then include 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.

End-to-end layers

Unlike the lower "subnet" layers whose protocol is between immediately adjacent nodes, the transport layer and the layers above are true "source to destination" or end-to-end layers, and are not concerned with the details of the underlying communications facility. Transport layer software (and software above it) on the source station carries on a conversation with similar software on the destination station by using message headers and control messages.

SESSION LAYER

The session layer allows session establishment between processes running on different stations. It provides: 

* Session establishment, maintenance and termination: allows two application processes on different machines to establish, use and terminate a connection, called a session.
* Session support: performs the functions that allow these processes to communicate over the network, performing security, name recognition, logging, and so on.

PRESENTATION LAYER

The presentation layer formats the data to be presented to the application layer. It can be viewed as the translator for the network. This layer may translate data from a format used by the application layer into a common format at the sending station, then translate the common format to a format known to the application layer at the receiving station.   
  
The presentation layer provides: 

* Character code translation: for example, ASCII to EBCDIC.
* Data conversion: bit order, CR-CR/LF, integer-floating point, and so on.
* Data compression: reduces the number of bits that need to be transmitted on the network.
* Data encryption: encrypt data for security purposes. For example, password encryption.

APPLICATION LAYER

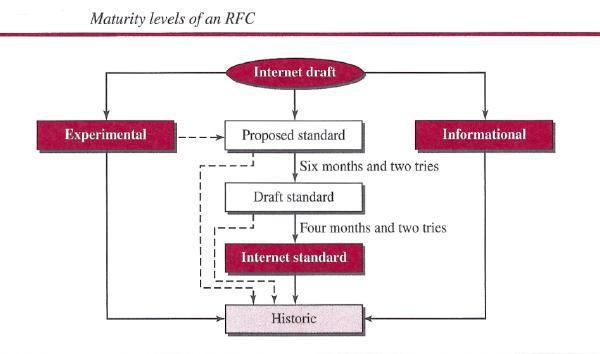
The application layer serves as the window for users and application processes to access network services. This layer contains a variety of commonly needed functions: 

* Resource sharing and device redirection
* Remote file access
* Remote printer access
* Inter-process communication
* Network management
* Directory services
* Electronic messaging (such as mail)
* Network virtual terminals.

3.**Explain** in detail Internet Standards?

Ans.An Internet standard is a thoroughly tested specification that is useful to and adhered to by thosewho work with the Internet. It is a formalized regulation that must be followed. There is a strict procedure by which a specification attains Internet standard status. A specification begins as an Internet draft. An Internet draft is a working document (a work in progress) with no official status and a six-month lifetime. Upon recommendation from the Internet authorities, a draft may be published as a Request for Comment (RFC). Each RFC is edited, assigned a number, and made available to all interested parties. RFCs go through maturity levels and are categorized according to their requirement level.

***Maturity Levels***

An RFC, during its lifetime, falls into one of six *maturity levels:* proposed standard, draft standard, Internet standard, historic, experimental, and informational. *Proposed Standard.* A proposed standard is a specification that is stable, well understood, and of sufficient interest to the Internet community. At this level, the specification is usually tested and implemented by several different group

***Draft Standard.*** A proposed standard is elevated to draft standard status after at least twosuccessful independent and interoperable implementations. Barring difficulties, a draft standard, with modifications if specific problems are encountered, normally becomes an Internet standard.

***Internet Standard****.*A draft standard reaches Internet standard status after demonstrationsof successful implementation.

***Historic*** The historic RFCs are significant from a historical perspective. They either have beensuperseded by later specifications or have never passed the necessary maturity levels to become an Internet standard.

***Experimental*** An RFC classified as experimental describes work related to an experimentalsituation that does not affect the operation of the Internet. Such an RFC should not be implemented in any functional Internet service.

***Informational*** An RFC classified as informational contains general, historical, or tutorialinformation related to the Internet. It is usually written by someone in a non-Internet organization, such as a vendor.

***Requirement Levels***

RFCs are classified into five *requirement levels:* required, recommended, elective, limited use, and not recommended.

***Required*** An RFC is labeled*required*if it must be implemented by all Internets systems toachieve minimum conformance. For example, IF and ICMP are required protocols.

***Recommended*** An RFC labeled recommended is not required for minimum conformance; it isrecommended because of its usefulness. For example, FTP and TELNET are recommended protocols.

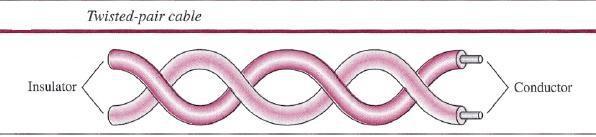
***Elective*** An RFC labeled elective is not required and not recommended. However, a system canuse it for its own benefit.

***Limited Use*** An RFC labeled limited use should be used only in limited situations. Most of theexperimental RFCs fall under this category.

***Not Recommended*** *An*RFC labeled not recommended is inappropriate for general use. Normallya historic (deprecated) RFC may fall under this category.

**4.** Explain in detail Twisted pair cable in Guided media?

Guided media, which are those that provide a conduit from one device to another, include twisted-pair cable, coaxial cable, and fiber-optic cable. A signal traveling along any of these media is directed and contained by the physical limits of the medium. Twisted-pair and coaxial cable use metallic (copper) conductors that accept and transport signals in the form of electric current. Optical fiber is a cable that accepts and transports signals in the form of light.

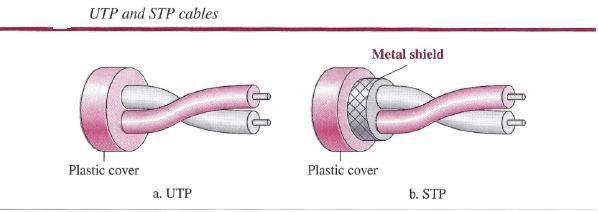


Twisted-Pair Cable

A twisted pair consists of two conductors (normally copper), each with its own plastic insulation, twisted together, as shown in following figure.

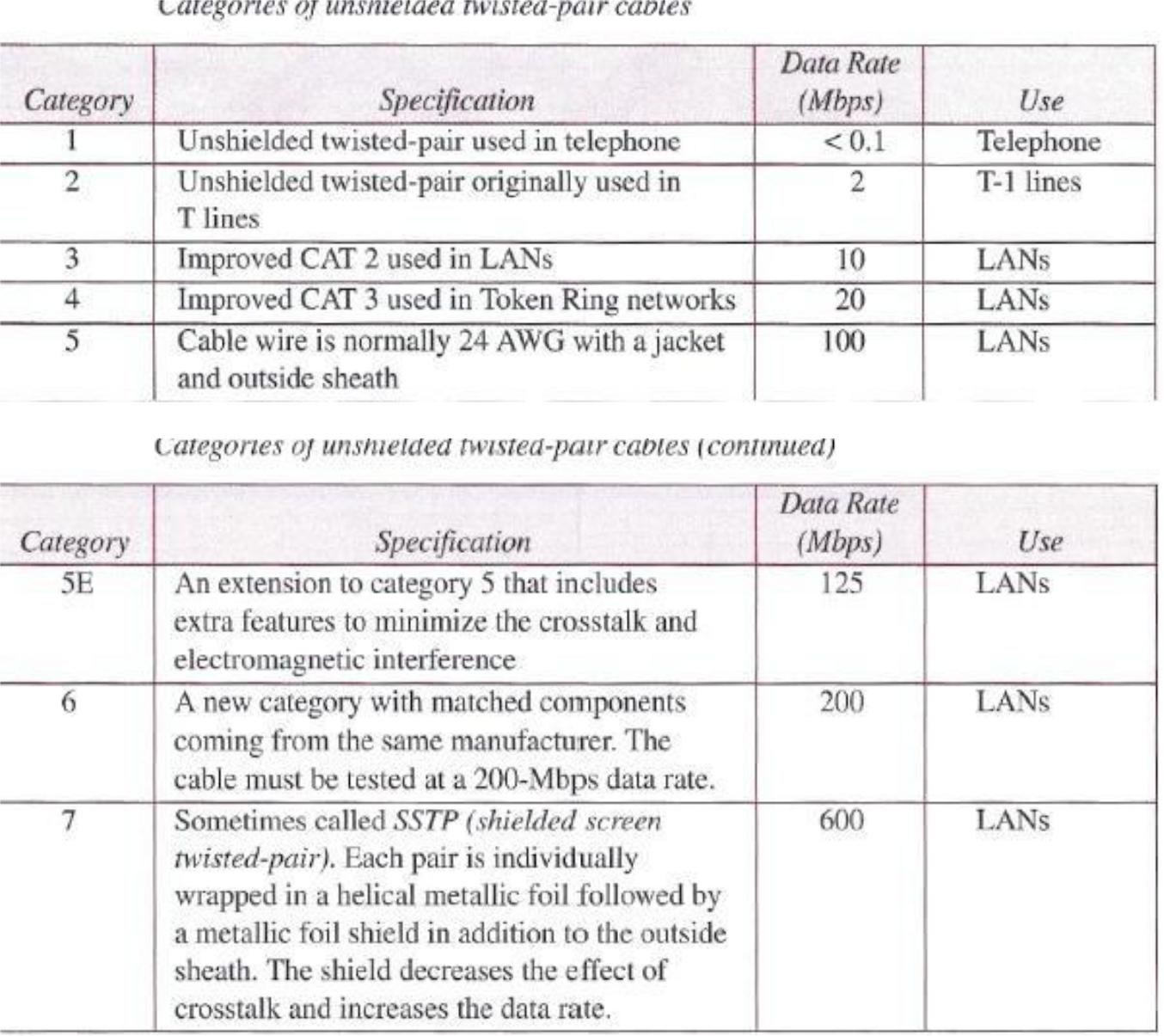
One of the wires is used to carry signals to the receiver, and the other is used only as a ground reference. The receiver uses the difference between the two.In addition to the signal sent by the sender on one of the wires, interference (noise) and crosstalk may affect both wires and create unwanted 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 (e.g., one is closer and the other is farther). This results in a difference at the receiver. By twisting the pairs, a balance is maintained. For example, suppose in one twist, one wire is closer to the noise source and the other is farther; in the next twist, the reverse is true. Twisting makes it probable that both wires are equally affected by external influences (noise or crosstalk). This means that the receiver, which calculates the difference between the two, receives no unwanted signals. The unwanted signals are mostly canceled out. From the above discussion, it is clear that the number of twists per unit of length (e.g., inch) has some effect on the quality of the cable.

**Unshielded Versus Shielded Twisted-Pair Cable**

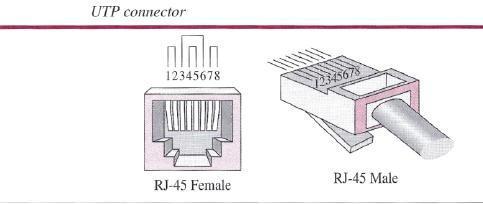
The most common twisted-pair cable used in communications is referred to as *unshielded* *twisted-pair* (UTP). IBM has also produced a version of twisted-pair cable for its use, called *shielded twisted-pair* (STP). STP cable has a metal foil or braided mesh covering that encaseseach pair of insulated conductors. Although metal casing improves the quality of cable by preventing the penetration of noise or crosstalk, it is bulkier and more expensive. Below figure

***Categories***

The Electronic Industries Association (EIA) has developed standards to classify unshielded twisted-pair cable into seven categories. Categories are determined by cable quality, with 1 as the lowest and 7 as the highest. Each EIA category is suitable for specific uses. Table below shows these categories.

***Connectors***

The most common UTP connector is **RJ45** (RJ stands for registered jack), as shown in below figure. The RJ45 is a keyed connector, meaning the connector can be inserted in only one way.



***Performance***

One way to measure the performance of twisted-pair cable is to compare attenuation versus frequency and distance. A twisted-pair cable can pass a wide range of frequencies. However, bel kilometer (dB/km), sharply increases with frequencies above 100 kHz. Note that *gauge* is a measure of the thickness of the wire.

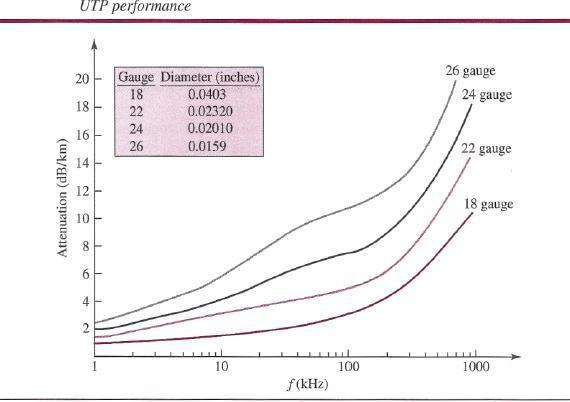
***Applications***

Twisted-pair cables are used in telephone lines to provide voice and data channels. The local loop-the line that connects subscribers to the central telephone office commonly consists of unshielded twisted-pair cables.

The DSL lines that are used by the telephone companies to provide high-data-rate connections also use the high-bandwidth capability of unshielded twisted-pair cables.

below figure shows that with increasing frequency, the attenuation, measured in decibels

Local-area networks, such as lOBase-T and lOOBase-T, also use twisted-pair cables.

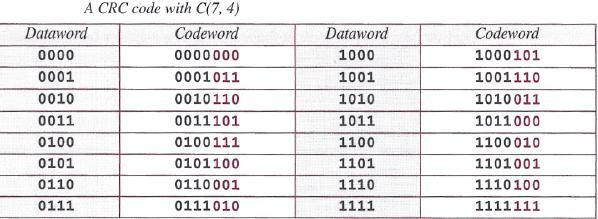


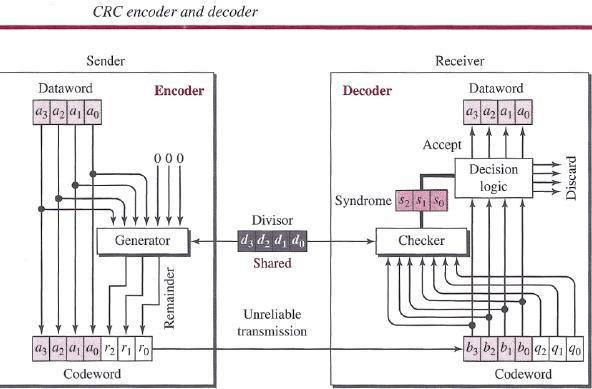
5. What are Cyclic codes? Explain CRC?

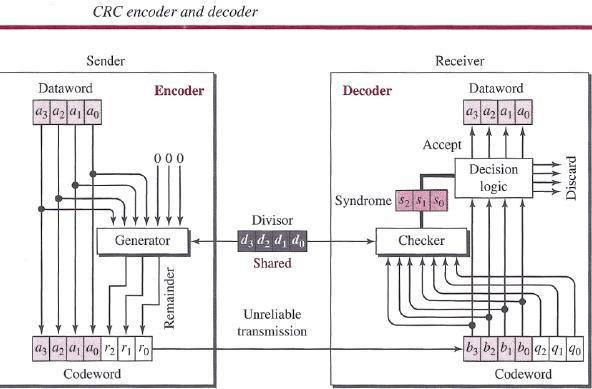
Cyclic codes are special linear block codes with one extra property. In a cyclic code, if a codeword is cyclically shifted (rotated), the result is another codeword. For example, if 1011000 is a codeword and we cyclically left-shift, then 0110001 is also a codeword. In this case, if we call the bits in the first word *ao* to *a6,* and the bits in the second word *bo* to *b6,* we can shift the bits by using the following: In the rightmost equation, the last bit of the first word is wrapped around and becomes the first bit of the second word.

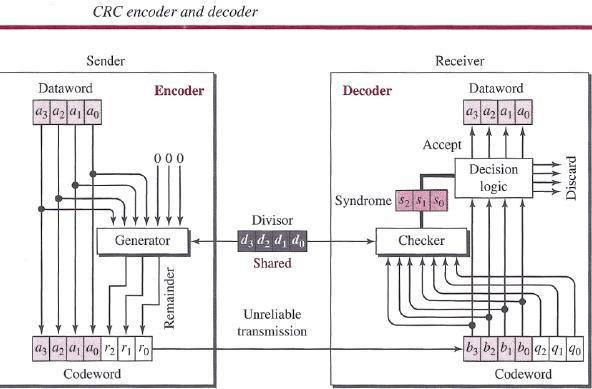
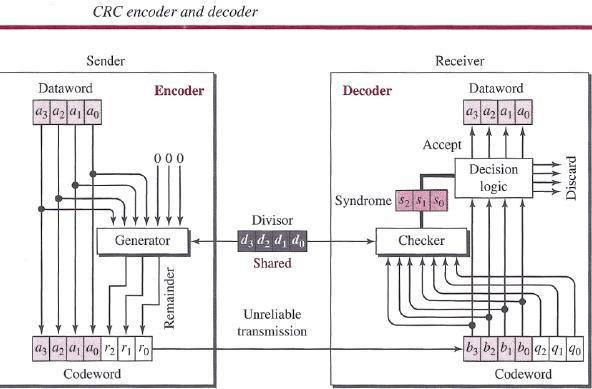
**Cyclic Redundancy Check**

We can create cyclic codes to correct errors. However, the theoretical background required is beyond the scope of this book. In this section, we simply discuss a subset of cyclic codes called the cyclic redundancy check (CRC), which is used in networks such as LANs and WANs. Table below shows an example of a CRC code. We can see both the linear and cyclic properties of this code.





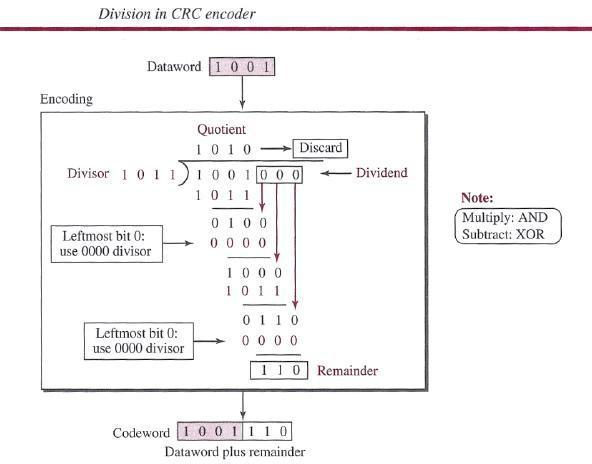




In the encoder, the dataword has *k* bits (4 here); the codeword has *n* bits (7 here). The size of the dataword is augmented by adding *n - k* (3 here) Os to the right-hand side of the word. The *n-bit* result is fed into the generator. The generator uses a divisor of size *n - k* + 1 (4 here), predefined and agreed upon. The generator divides the augmented dataword by the divisor (modulo-2 division). The quotient of the division is discarded; the remainder *(r2rlrO)* is appended to the dataword to create the codeword. The decoder receives the codeword (possibly corrupted in transition). A copy of all *n* bits is fed to the checker, which is a replica of the generator. The remainder produced by the checker is a syndrome of *n - k* (3 here) bits, which is fed to the decision logic analyzer. The analyzer has a simple function. If the syndrome bits are all Os, the 4 leftmost bits of the codeword are accepted as the dataword (interpreted as no error); otherwise, the 4 bits are discarded (error).

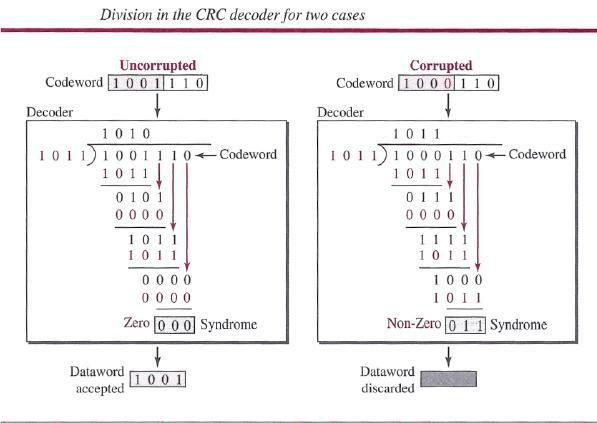
***Encoder***

Let us take a closer look at the encoder. The encoder takes a dataword and augments it with *n - k* number of Os. It then divides the augmented dataword by the divisor, as shown in below figure.



***Decoder***

The codeword can change during transmission. The decoder does the same division process as the encoder. The remainder of the division is the syndrome. If the syndrome is all Os, there is no error with a high probability; the dataword is separated from the received codeword and accepted. Otherwise, everything is discarded. Figure 10.7 shows two cases: The left-hand figure shows the value of the syndrome when no error has occurred; the syndrome is 000. The right-hand part of the figure shows the case in which there is a single error. The syndrome is not all Os (it is 011).



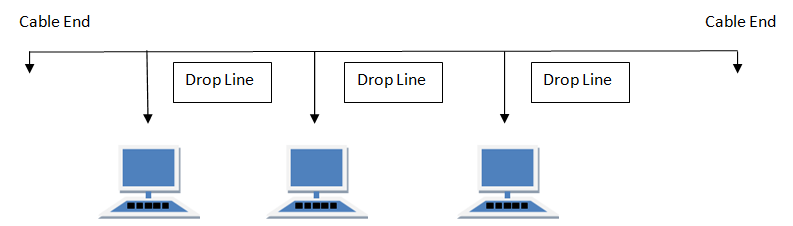
6. **Define**topology and explain the topologies of the network?

Network Topology is the schematic description of a network arrangement, connecting various nodes(sender and receiver) through lines of connection.

**Types of Network Topology**

**BUS Topology**

Bus topology is a network type in which every computer and network device is connected to single cable. When it has exactly two endpoints, then it is called **Linear Bus topology**.



**Features of Bus Topology**

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

**Advantages of Bus Topology**

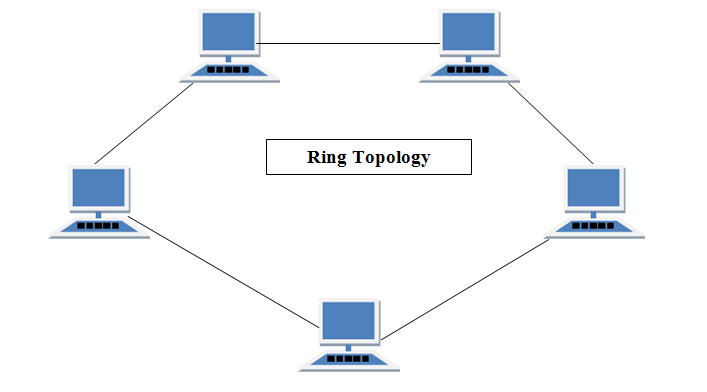
1. It is cost effective.
2. Cable required is least compared to other network topology.
3. Used in small networks.
4. It is easy to understand.
5. Easy to expand joining two cables together.

**Disadvantages of Bus Topology**

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

**RING Topology**

It is called ring topology because it forms a ring as each computer is connected to another computer, with the last one connected to the first. Exactly two neighbours for each device.



**Features of Ring Topology**

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

**Advantages of Ring Topology**

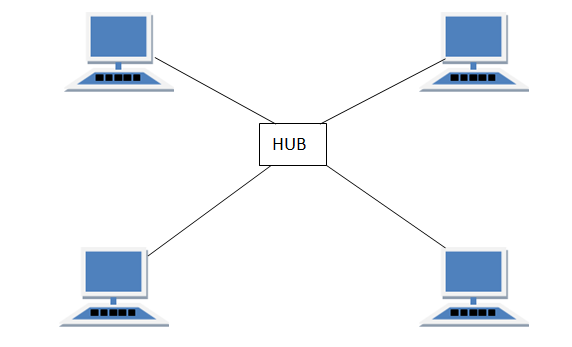
1. Transmitting network is not affected by high traffic or by adding more nodes, as only the nodes having tokens can transmit data.
2. Cheap to install and expand

**Disadvantages of Ring Topology**

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

**STAR Topology**

In this type of topology all the computers are connected to a single hub through a cable. This hub is the central node and all others nodes are connected to the central node.



**Features of Star Topology**

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

**Advantages of Star Topology**

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

**Disadvantages of Star Topology**

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

**MESH Topology**

It is a point-to-point connection to other nodes or devices. All the network nodes are connected to each other. Mesh has n(n-2)/2 physical channels to link n devices.

There are two techniques to transmit data over the Mesh topology, they are :

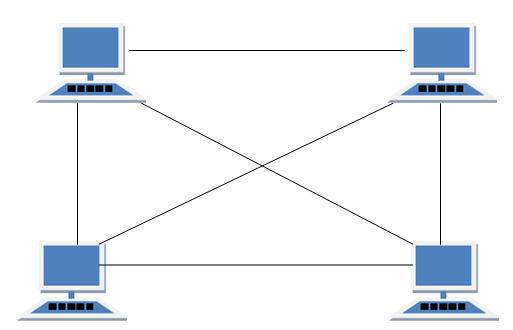
1. Routing
2. Flooding

**Routing**

In routing, the nodes have a routing logic, as per the network requirements. Like routing logic to direct the data to reach the destination using the shortest distance. Or, routing logic which has information about the broken links, and it avoids those node etc. We can even have routing logic, to re-configure the failed nodes.

**Flooding**

In flooding, the same data is transmitted to all the network nodes, hence no routing logic is required. The network is robust, and the its very unlikely to lose the data. But it leads to unwanted load over the network.



**Types of Mesh Topology**

1. **Partial Mesh Topology :**In this topology some of the systems are connected in the same fashion as mesh topology but some devices are only connected to two or three devices.
2. **Full Mesh Topology :**Each and every nodes or devices are connected to each other.

**Features of Mesh Topology**

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

**Advantages of Mesh Topology**

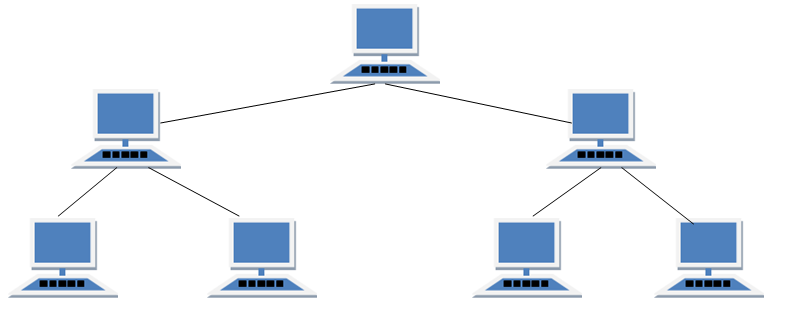
1. Each connection can carry its own data load.
2. It is robust.
3. Fault is diagnosed easily.
4. Provides security and privacy.

**Disadvantages of Mesh Topology**

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

**TREE Topology**

It has a root node and all other nodes are connected to it forming a hierarchy. It is also called hierarchical topology. It should at least have three levels to the hierarchy.



**Features of Tree Topology**

1. Ideal if workstations are located in groups.
2. Used in Wide Area Network.

**Advantages of Tree Topology**

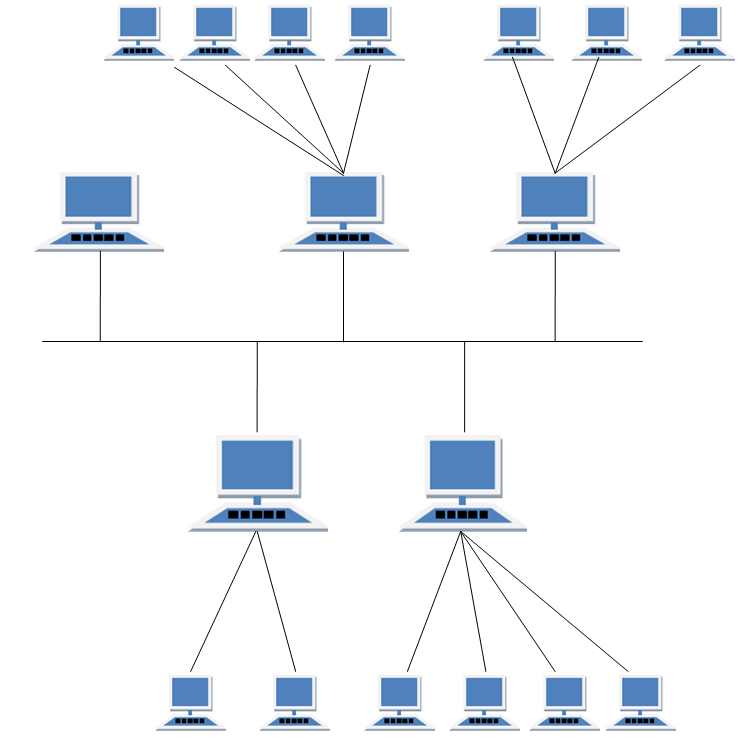
1. Extension of bus and star topologies.
2. Expansion of nodes is possible and easy.
3. Easily managed and maintained.
4. Error detection is easily done.

**Disadvantages of Tree Topology**

1. Heavily cabled.
2. Costly.
3. If more nodes are added maintenance is difficult.
4. Central hub fails, network fails.

**HYBRID Topology**

It is two different types of topologies which is a mixture of two or more topologies. For example if in an office in one department ring topology is used and in another star topology is used, connecting these topologies will result in Hybrid Topology (ring topology and star topology).



**Features of Hybrid Topology**

1. It is a combination of two or topologies
2. Inherits the advantages and disadvantages of the topologies included

**Advantages of Hybrid Topology**

1. Reliable as Error detecting and trouble shooting is easy.
2. Effective.
3. Scalable as size can be increased easily.
4. Flexible.

**Disadvantages of Hybrid Topology**

1. Complex in design.
2. Costly.

7. Explain Sliding window Protocol?

In sliding window method, multiple frames are sent by sender at a time before needing an acknowledgment.

 Multiple frames sent by source are acknowledged by receiver using a single ACK frame.

## Sliding Window

• Sliding window refers to an imaginary boxes that hold the frames on both sender and receiver side.

• It provides the upper limit on the number of frames that can be transmitted before requiring an acknowledgment.

• Frames may be acknowledged by receiver at any point even when window is not full on receiver side.

• Frames may be transmitted by source even when window is not yet full on sender side.

• The windows have a specific size in which the frames are numbered modulo- n, which means they are numbered from 0 to n-l. For e.g. if n = 8, the frames are numbered 0, 1,2,3,4,5,6, 7, 0, 1,2,3,4,5,6, 7, 0, 1, ....

• The size of window is n-1. For e.g. In this case it is 7. Therefore, a maximum of n-l frames may be sent before an acknowledgment.

• When the receiver sends an ACK, it includes the number of next frame it expects to receive. For example in order to acknowledge the group of frames ending in frame 4, the receiver sends an ACK containing the number 5. When sender sees an ACK with number 5, it comes to know that all the frames up to number 4 have been received.

[](http://ecomputernotes.com/images/Sliding-Window.jpg)

## Sliding Window on Sender Side

• At the beginning of a transmission, the sender's window contains n-l frames.

• As the frames are sent by source, the left boundary of the window moves inward, shrinking the size of window. This means if window size is w, if four frames are sent by source after the last acknowledgment, then the number of frames left in window is w-4.

• When the receiver sends an ACK, the source's window expand i.e. (right boundary moves outward) to allow in a number of new frames equal to the number of frames acknowledged by that ACK.

• For example, Let the window size is 7 (see diagram (a)), if frames 0 through 3 have been sent and no acknowledgment has been received, then the sender's window contains three frames - 4,5,6.

• Now, if an ACK numbered 3 is received by source, it means three frames (0, 1, 2) have been received by receiver and are undamaged.

• The sender's window will now expand to include the next three frames in its buffer. At this point the sender's window will contain six frames (4, 5, 6, 7, 0, 1). (See diagram (b)).

## Sliding Window on Receiver Side

• At the beginning of transmission, the receiver's window contains n-1 spaces for frame but not the frames.

• As the new frames come in, the size of window shrinks.

• Therefore the receiver window represents not the number of frames received but the number of frames that may still be received without an acknowledgment ACK must be sent.

• Given a window of size w, if three frames are received without an ACK being returned, the number of spaces in a window is w-3.

• As soon as acknowledgment is sent, window expands to include the number of frames equal to the number of frames acknowledged.

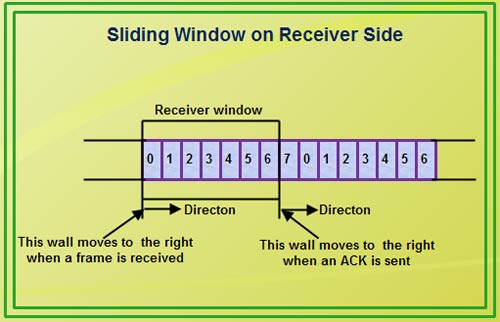
• For example, let the size of receiver's window is 7 as shown in diagram. It means window contains spaces for 7 frames.

• With the arrival of the first frame, the receiving window shrinks, moving the boundary from space 0 to 1. Now, window has shrunk by one, so the receiver may accept six more frame before it is required to send an ACK.

• If frames 0 through 3 have arrived but have DOC been acknowledged, the window will contain three frame spaces.

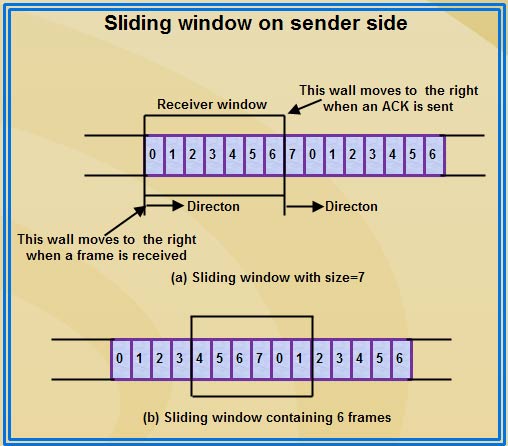
• As receiver sends an ACK, the window of the receiver expands to include as many new placeholders as newly acknowledged frames.

• The window expands to include a number of new frame spaces equal to the number of the most recently acknowledged frame minus the number of previously acknowledged frame. For *e.g.,*If window size is 7 and if prior ACK was for frame 2 & the current ACK is for frame 5 the window expands by three (5-2).

[](http://ecomputernotes.com/images/Sliding-Window-on-Receiver-Side.jpg)

• Therefore, the sliding window of sender shrinks from left when frames of data are sending. The sliding window of the sender expands to right when acknowledgments are received.

• The sliding window of the receiver shrinks from left when frames of data are received. The sliding window of the receiver expands to the right when acknowledgement is sent.

[](http://ecomputernotes.com/images/Sliding-window-on-sender-side.jpg)

unit-2

Short Answer Questions

1. Define ALOHA?

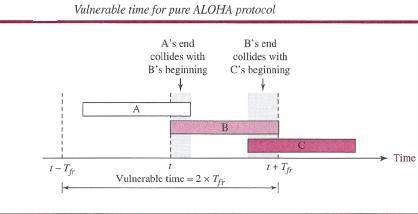
Ans. The original ALOHA protocol is called *pure ALOHA.* This is a simple but elegant protocol. The idea is that each station sends a frame whenever it has a frame to send (multiple access). However, since there is only one channel to share, there is the possibility of collision between frames from different stations. Below figure shows an example of frame collisions in pure ALOHA.

2.Define MAC?

ans. When nodes or stations are connected and use a common link, called a *multipoint* or *broadcast* *link,* we need a multiple-access protocol to coordinate access to the link.

Many protocols have been devised to handle access to a shared link. All of these protocols belong to a sub layer in the data-link layer called *media access control (MAC).*

3.Explain Vulnerable Time?

***ans.vulnerable time,*** the length of time in which there is a possibility of collision

**Pure ALOHA vulnerable time = 2 x *Tfr***

4.Explain how throughput is improved in slotted ALOHA overpure ALOHA?

ans. It can be proven that the average number of successful transmissions for slotted ALOHA is S =

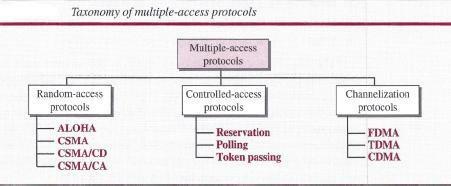
G x *e-G.*The maximum throughput *Smax* is 0.368, when G = 1. In other words, if one frame is generated during one frame transmission time, then 36.8 percent of these frames reach their destination successfully. We expect G = 1 to produce maximum throughput because the vulnerable time is equal to the frame transmission time. Therefore, if a station generates only one frame in this vulnerable time (and no other station generates a frame during this time), the frame will reach its destination successfully.

**The throughput for slotted ALOHA is S = G x e-G**

**The maximum throughput *Smax* = 0.368 when G = 1.**

5.List three categories of multiple access protocols?

ans. We categorize them into three groups:



**7. What is 100Base-FX**

**ans. It** uses two pairs of fiber-optic cables. Optical fiber can easily handle high bandwidthrequirements by using simple encoding schemes. The designers of 100Base-FX selected the NRZ-I encoding scheme for this implementation. However, NRZ-I has a bit synchronization problem for long sequences of 0s (or Is, based on the encoding). To overcome this problem, the designers used 4B/5B block encoding, as we described for 100Base- TX. The block encoding increases the bit rate from 100 to 125 Mbps, which can easily be handled by fiber-optic cable. A 100Base-TX network can provide a data rate of 100 Mbps, but it requires the use of category 5 UTP or STP cable. This is not cost-efficient for buildings that have already been wired for voice-grade twisted-pair (category 3). A new standard, called ***lOOBase-T4,*** was designed to use category 3 or higher UTP. The implementation uses four pairs of UTP for transmitting 100 Mbps. Encoding/decoding in 100Base-T4 is more complicated. As this implementation uses category 3 UTP, each twisted-pair cannot easily handle more than 25 Mbaud. In this design, one pair switches between sending and receiving. Three pairs of UTP category 3, however, can handle only 75 Mbaud (25 Mbaud) each. We need to use an encoding scheme that converts 100 Mbps to a 75 Mbaud signal. 8B/6T satisfies this requirement. In 8B/6T, eight data elements are encoded as six signal elements. This means that 100 Mbps uses only (6/8) x 100 Mbps, or 75 Mbaud.

**8.Explain 100Base- TX**

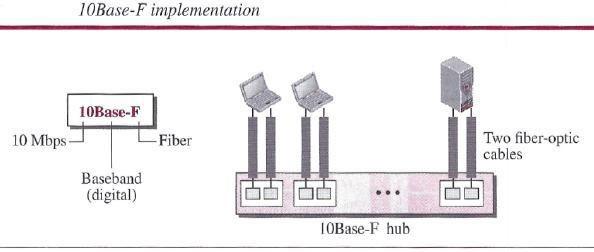
**ans.It** uses two pairs of twisted-pair cable (either category S UTP or STP). For thisimplementation, the MLT-3 scheme was selected since it has good bandwidth performance. (However, since MLT-3 is not a self-synchronous line coding scheme, 4B/SB block coding is used to provide bit synchronization by preventing the occurrence of a long sequence of Os and Is. This creates a data rate of 125 Mbps, which is fed into MLT-3 for encoding.

9. ***what is Topology***

Fast Ethernet is designed to connect two or more stations. If there are only two stations, they can be connected point-to-point. Three or more stations need to be connected in a star topology with a hub or a switch at the center.

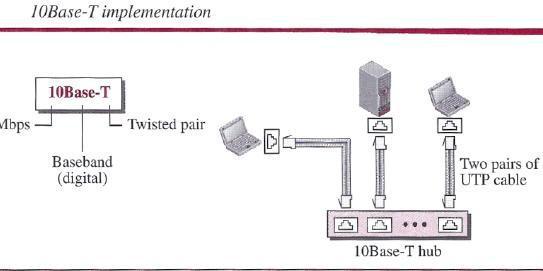
10.what is ***10Base-F: Fiber Ethernet***

Although there are several types of optical fiber 10-Mbps Ethernet, the most common is called *lOBase-F.* lOBase-F uses a star topology to connect stations to a hub. The stations are connectedto the hub using two fiber-optic cables, as shown in Figure.



11. what is ***10Base-T: Twisted-Pair Ethernet***

The third implementation is called *1OBase- T* or *twisted-pair Ethernet.* 1OBase-T uses a physical star topology. The stations are connected to a hub via two pairs of twisted cable, as shown in Figure.

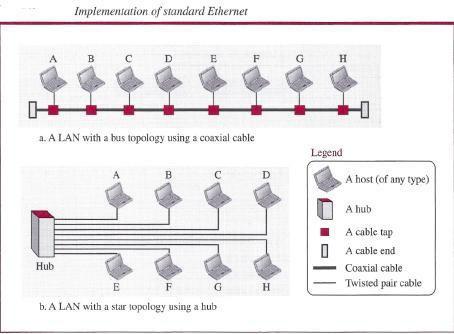


Note that two pairs of twisted cable create two paths (one for sending and one for receiving) between the station and the hub. Any collision here happens in the hub. Compared to lOBase5 or lOBase2, we can see that the hub actually replaces the coaxial cable as far as a collision is concerned. The maximum length of the twisted cable here is defined as 100 m, to minimize the effect of attenuation in the twisted cable.

12. ***Distinguish Between Unicast, Multicast, and Broadcast Transmission***

ans.Standard Ethernet uses a coaxial cable (bus topology) or a set of twisted-pair cables with a hub (star topology) as shown in Figure. We need to know that transmission in the standard Ethernet is always broadcast, no matter if the intention is unicast, multicast, or broadcast. In the bus topology, when station A sends a frame to station B, all stations will receive it. In the star topology, when station A sends a frame to station B, the hub will receive it. Since the hub is a passive element, it does not check the destination address of the frame; it regenerates the bits (if they have been weakened) and sends them to all stations except station A. In fact, it floods the network with the frame. The question is, then, how the actual unicast, multicast, and broadcast transmissions are distinguished from each other. The answer is in the way the frames are kept or dropped.

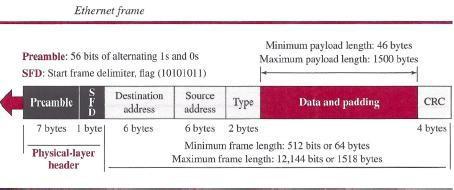
* In a unicast transmission, all stations will receive the frame, the intended recipient keeps and handles the frame; the rest discard it.
* In a multicast transmission, all stations will receive the frame, the stations that are members of the group keep and handle it; the rest discard it.
* In a broadcast transmission, all stations (except the sender) will receive the frame and all stations (except the sender) keep and handle it.



13.what is Ethernet Frame format.

ans.***Frame Format***

The Ethernet frame contains seven fields, as shown in below figure.



\

Long Answer Questions.

1. Discuss the MAC layer functions of IEEE 802.11?

ans. ***Media Access Control (MAC)***

Earlier we discussed multiple access methods including random access, controlled access, and channelization. IEEE Project 802 has created a sublayer called *media access control* that defines the specific access method for each LAN. For example, it defines *CSMA/CD* as the media access method for Ethernet LANs and defines the token-passing method for Token Ring and Token Bus LANs. As we mentioned in the previous section, part of the framing function is also handled by the MAC layer.

**STANDARD ETHERNET**

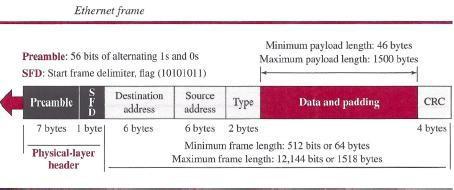
The original Ethernet technology with the data rate of 10 Mbps as the *Standard Ethernetis* *referred .* Although most implementations have moved to other technologies in the Ethernetevolution, there are some features of the Standard Ethernet that have not changed during the evolution.

**Characteristics**

Let us first discuss some characteristics of the Standard Ethernet.

***Frame Format***

The Ethernet frame contains seven fields, as shown in below figure.



***Preamble*** This field contains 7 bytes (56 bits) of alternating Os and Is that alert the receivingsystem to the coming frame and enable it to synchronize its clock if it's out of synchronization. The pattern provides only an alert and a timing pulse. The 56-bit pattern allows the stations to miss some bits at the beginning of the frame. The *preamble* is actually added at the physical layer and is not (formally) part of the frame.

***Start frame delimiter (SFD)*** This field (1 byte: 10101011) signals the beginning of the frame.The SFD warns the station or stations that this is the last chance for synchronization. The last 2 bits are (llh and alert the receiver that the next field is the destination address. This field is actually a flag that defines the beginning of the frame. We need to remember that an Ethernet frame is a variable-length frame. It needs a flag to define the beginning of the frame. The SFD field is also added at the physical layer.

***Destination address (DA)*** This field is six bytes (48 bits) and contains the linklayer address ofthe destination station or stations to receive the packet. When the receiver sees its own link-layer address, or a multicast address for a group that the receiver is a member of, or a broadcast address, it decapsulates the data from the frame and passes the data to the upperlayer protocol defined by the value of the type field.

***Source address (SA)*** This field is also six bytes and contains the link-layer address of the senderof the packet.

***Type*** This field defines the upper-layer protocol whose packet is encapsulated in the frame. Thisprotocol can be IP, ARP, OSPF, and so on. In other words, it serves the same purpose as the protocol field in a datagram and the port number in a segment or user datagram. It is used for multiplexing and demultiplexing.

***Data*** This field carries data encapsulated from the upper-layer protocols. It is a minimum of 46and a maximum of 1500 bytes. We discuss the reason for these minimum and maximum valuesshortly. If the data coming from the upper layer is more than 1500 bytes, it should be fragmented and encapsulated in more than one frame. If it is less than 46 bytes, it needs to be padded with extra Os. A padded data frame is delivered to the upper-layer protocol as it is (without removing the padding), which means that it is the responsibility of the upper layer to remove or, in the case of the sender, to add the padding. The upper-layer protocol needs to know the length of its data. For example, a datagram has a field that defines the length of the data.

*CRC* The last field contains error detection information, in this case a CRC-32. The CRC iscalculated over the addresses, types, and data field. If the receiver calculates the CRC and finds that it is not zero (corruption in transmission), it discards the frame.

***Frame Length***

Ethernet has imposed restrictions on both the minimum and maximum lengths of a frame. The minimum length restriction is required for the correct operation of CSMAlCD. An Ethernet frame needs to have a minimum length of 512 bits or 64 bytes. Part of this length is the header and the trailer. If we count 18 bytes of header and trailer (6 bytes of source address, 6 bytes of destination address, 2 bytes of length or type, and 4 bytes of CRC), then the minimum length of data from the upper layer is 64 - 18 = 46 bytes. If the upper-layer packet is less than 46 bytes, padding is added to make up the difference.

The standard defines the maximum length of a frame (without preamble and SFD field) as 1518 bytes. If we subtract the 18 bytes of header and trailer, the maximum length of the payload is 1500 bytes. The maximum length restriction has two historical reasons. First, memory was very expensive when Ethernet was designed; a maximum length restriction helped to reduce the size of the buffer. Second, the maximum length restriction prevents one station from monopolizing the shared medium, blocking other stations that have data to send.

**Minimum frame length: 64 bytes Maximum frame length: 1518 bytes**

**Minimum data length: 46 bytes Maximum data length: 1500 byte**

**Addressing**

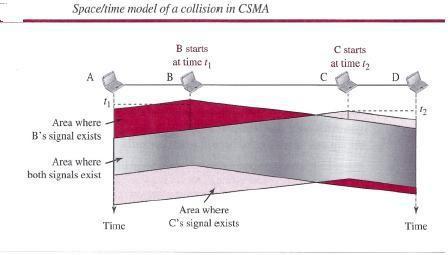
Each station on an Ethernet network (such as a PC, workstation, or printer) has its own network interface card (NIC). The NIC fits inside the station and provides the station with a link-layer address. The Ethernet address is 6 bytes (48 bits), normally written in hexadecimal notation, with a colon between the bytes. For example, the following shows an Ethernet MAC address:

4A:30:10:21:10:1A

***Transmission of Address Bits***

The way the addresses are sent out online is different from the way they are written in hexadecimal notation. The transmission is left to right, byte by byte; however, for each byte, the least significant bit is sent first and the most significant bit is sent last. This means that the bit that defines an address as unicast or multicast arrives first at the receiver. This helps the receiver to immediately know if the packet is unicast or multicast.

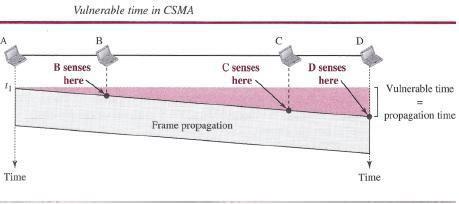
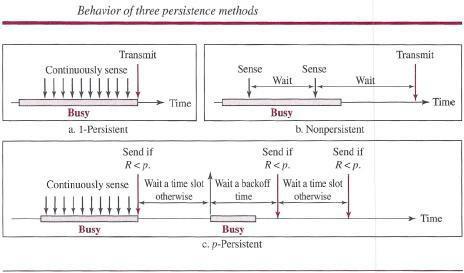
2. Explain the Concept of **CSMA?**

ans:To minimize the chance of collision and, therefore, increase the performance, the CSMA method was developed. The chance of collision can be reduced if a station senses the medium before trying to use it. Carrier sense multiple access (CSMA) requires that each station first listen to the medium (or check the state of the medium) before sending. In other words, CSMA is based on the principle "sense before transmit" or "listen before talk." CSMA can reduce the possibility

At time fl, station B senses the medium and finds it idle, so it sends a frame. At time *t2 (t2* > *tl),* station C senses the medium and finds it idle because, at this time, the first bits from station B have not reached station C. Station C also sends a frame. The two signals collide and both frames are destroyed.

***Vulnerable Time***

The vulnerable time for CSMA is the *propagation time Tp.* This is the time needed for a signal to propagate from one end of the medium to the other. When a station sends a frame and any other station tries to send a frame during this time, a collision will result. But if the first bi t of the frame reaches the end of the medium, every station will already have heard the bit and will refrain from sending. Below Figure shows the worst case. The leftmost station, A, sends a frame at time *fl,* which reaches the rightmost station, D, at time *tl* + *Tp.* The gray area shows the vulnerableareaintim



Above Figure shows the flow diagrams for these methods.

***I-Persistent***

The *l-persistent method* is simple and straightforward. In this method, after the station finds the line idle, it sends its frame immediately (with probability 1). This method has the highest chance of collision because two or more stations may find the line idle and send their frames immediately. We will see later that Ethernet uses this method.

***Non persistent***

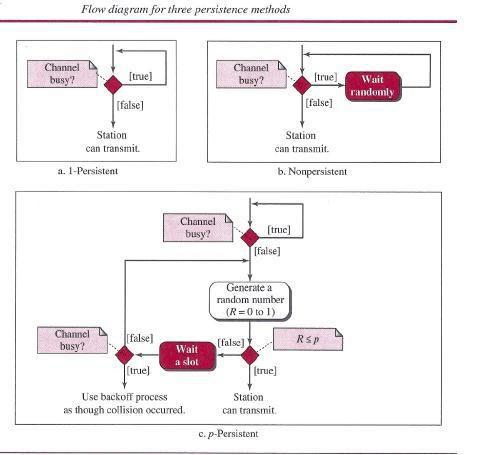
In the *nonpersistent method,* a station that has a frame to send senses the line. If the line is idle, it sends immediately. If the line is not idle, it waits a random amount of time and then senses the line again. The nonpersistent approach reduces the chance of collision because it is unlikely that two or more stations will wait the same amount of time and retry to send simultaneously. However, this method reduces the efficiency of the network because the medium remains idle when there may be stations with frames to send.

***P- Persistent***

The *p-persistent method* is used if the channel has time slots with a slot duration equal to or greater than the maximum propagation time. The p-persistent approach combines the advantages of the other two strategies. It reduces the chance of collision and improves efficiency. In this method, after the station finds the line idle it follows these

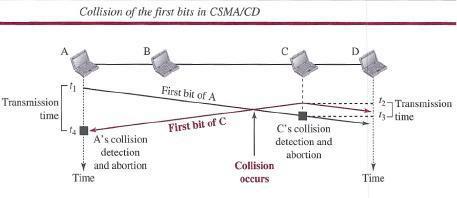
Steps: With probability *p,* the station sends its frame.

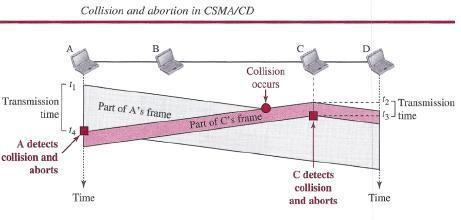
1. With probability *q* = 1- *p,* the station waits for the beginning of the next time slot and checks the line again.
2. If the line is idle, it goes to step 1.
3. If the line is busy, it acts as though a collision has occurred and uses the back off procedure.



**4.Explain CSMA/CD?**

The CSMA method does not specify the procedure following a collision. Carrier sense multiple access with collision detection *(CSMA/CD)* augments the algorithm to handle the collision. In this method, a station monitors the medium after it sends a frame to see if the transmission was successful. If so, the station is finished. If, however, there is a collision, the frame is sent again.

To better understand *CSMA/CD,* let us look at the first bits transmitted by the two stations involved in the collision. Although each station continues to send bits in the frame until it detects the collision, we show what happens as the first bits collide. In below figure, stations A and Care involved in the collision.

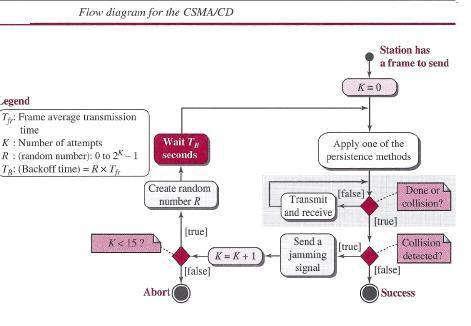
At time *t*1, station A has executed its persistence procedure and starts sending the bits of its frame. At time *t2,* station C has not yet sensed the first bit sent by A. Station C executes its persistence procedure and starts sending the bits in its frame, which propagate both to the left and to the right. The collision occurs sometime after time *t2'* Station C detects a collision at time *t3* when it receives the first bit of A's frame. Station C immediately (or after a short time, but weassume immediately) aborts transmission. Station A detects collision at time *t4* when it receives the first bit of C's frame; it also immediately aborts transmission. Looking at the figure, we see that A transmits for the duration *t4 - t1;* C transmits for the duration *t3 - t2'* Now that we know the time durations for the two transmissions, we can show a more complete graph in below figure.

***Minimum Frame Size***

For *CSMAJCD* to work, we need a restriction on the frame size. Before sending the last bit of the frame, the sending station must detect a collision, if any, and abort the transmission. This is so because the station, once the entire frame is sent, does not keep a copy of the frame and does not monitor the line for collision detection. Therefore, the frame transmission time *Tfr* must be at least two times the maximum propagation time *Tp.* To understand the reason, let us think about the worst-case scenario. If the two stations involved in a collision are the maximum distance apart, the signal from the first takes time *Tp* to reach the second, and the effect of the collision takes another time *Tp* to reach the first. So the requirement is that the first station must still be transmitting after *2Tp.*

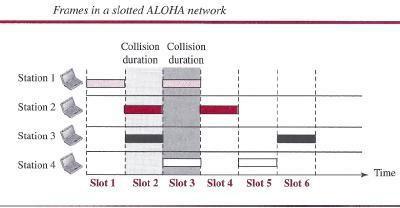
***Procedure***

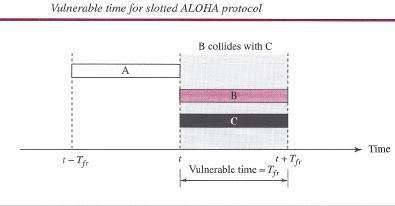
Now let us look at the flow diagram for *CSMA/CD* in Figure. It is similar to the one for the ALOHA protocol, but there are differences.



***5.Explain in detail Slotted ALOHA***

Pure ALOHA has a vulnerable time of 2 x *Tp* This is so because there is no rule that defines when the station can send. A station may send soon after another station has started or just before another station has finished. Slotted ALOHA was invented to improve the efficiency of pure ALOHA. In slotted ALOHA we divide the time into slots of *Tfr* seconds and force the station to send only at the beginning of the time slot. Below figure shows an example of frame collisions in slotted ALOHA.



Because a station is allowed to send only at the beginning of the synchronized time slot, if a station misses this moment, it must wait until the beginning of the next time slot. This means that the station which started at the beginning of this slot has already finished sending its frame. Of course, there is still the possibility of collision if two stations try to send at the beginning of the same time slot. However, the vulnerable time is now reduced to one-half, equal to *Tjr. Below* *f*igure shows the situation.

**Slotted ALOHA vulnerable time = *Tlr***

***Throughput***

It can be proven that the average number of successful transmissions for slotted ALOHA is S =

G x *e-G.*The maximum throughput *Smax* is 0.368, when G = 1. In other words, if one frame is generated during one frame transmission time, then 36.8 percent of these frames reach their destination successfully. We expect G = 1 to produce maximum throughput because the vulnerable time is equal to the frame transmission time. Therefore, if a station generates only one frame in this vulnerable time (and no other station generates a frame during this time), the frame will reach its destination successfully.

**The throughput for slotted ALOHA is S = G x e-G**

**The maximum throughput *Smax* = 0.368 when G = 1.**

The first difference is the addition of the persistence process. We need to sense the channel before we start sending the frame by using one of the persistence processes we discussed previously (non persistent, l-persistent, or p-persistent). The second difference is the frame transmission. In ALOHA, we first transmit the entire frame and then wait for an acknowledgment. In *CSMA/CD,* transmission and collision detection are continuous processes. We do not send the entire frame and then look for a collision. The station transmits and receives continuously and simultaneously (using two different ports or a bidirectional port). We use a loop to show that transmission is a continuous process. We constantly monitor in order to detect one of two conditions: either transmission is finished or a collision is detected. Either event stops transmission. When we come out of the loop, if a collision has not been detected, it means that transmission is complete; the entire frame is transmitted. Otherwise, a collision has occurred. The third difference is the sending of a short jamming signal to make sure that all other stations become aware of the collision.

**6.What is ALOHA.Describe ALOHA Procedure.**

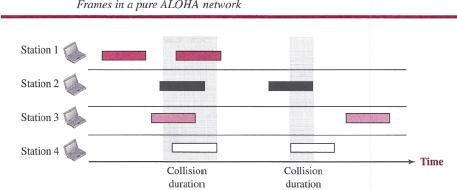
ALOHA, the earliest random access method was developed at the University of Hawaii in early 1970. It was designed for a radio (wireless) LAN, but it can be used on any shared medium.

It is obvious that there are potential collisions in this arrangement. The medium is shared between the stations. When a station sends data, another station may attempt to do so at the same time. The data from the two stations collide and become garbled.

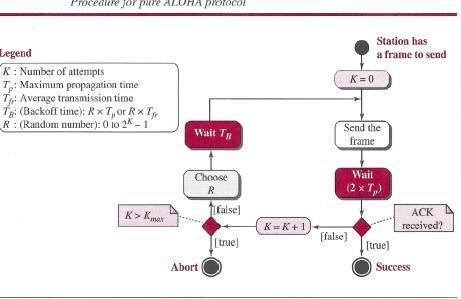
***Pure ALOHA***

The original ALOHA protocol is called *pure ALOHA.* This is a simple but elegant protocol. The idea is that each station sends a frame whenever it has a frame to send (multiple access). However, since there is only one channel to share, there is the possibility of collision between frames from different stations. Below figure shows an example of frame collisions in pure

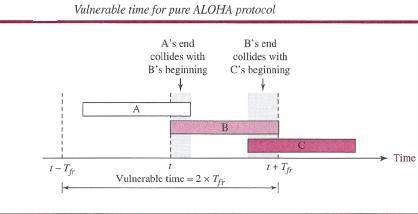
ALOHA.



There are four stations (unrealistic assumption) that contend with one another for access to the shared channel. The figure shows that each station sends two frames; there are a total of eight frames on the shared medium. Some of these frames collide because multiple frames are in contention for the shared channel. Above Figure shows that only two frames survive: one frame from station 1 and one frame from station 3. We need to mention that even if one bit of a frame coexists on the channel with one bit from another frame, there is a collision and both will be destroyed. It is obvious that we need to resend the frames that have been destroyed during transmission. The pure ALOHA protocol relies on acknowledgments from the receiver. When a station sends a frame, it expects the receiver to send an acknowledgment. If the acknowledgment does not arrive after a time-out period, the station assumes that the frame (or the acknowledgment) has been destroyed and resends the frame. A collision involves two or more stations. If all these stations try to resend their frames after the time-out, the frames will collide again. Pure ALOHA dictates that when the time-out period passes, each station waits a random amount of time before resending its frame. The randomness will help avoid more collisions. We call this time the *back off time Ts.*

Pure ALOHA has a second method to prevent congesting the channel with retransmitted frames. After a maximum number of retransmission attempts *Kmax'* a station must give up and try later. The time-out period is equal to the maximum possible round-trip propagation delay, which is twice the amount of time required to send a frame between the two most widely separated stations (2 x *Tp).*The backoff time *Ts* is a random value that normally depends on *K* (the number of attempted unsuccessful transmissions). The formula for *Ts* depends on the implementation. One common formula is the *binary exponential backoff.* In this method, for each retransmission, a multiplier *R* = 0 to *2K -* 1 is randomly chosen and multiplied by *Tp*(maximum propagation time) or *Tfr* (the average time required to send out a frame) to find *Ts.* Note that in this rocedure, the range of the random numbers increases after each collision. The value of *Kmax* is usually chosen as 15.

***Vulnerable time***

Let us find the ***vulnerable time,*** the length of time in which there is a possibility of collision.We assumes that the stations send fixed-length frames with each frame taking *Tjr* seconds to send. Following figure shows the vulnerable time for station B.

Station B starts to send a frame at time *t.* Now imagine station A has started to send its frame after *t - TjT'* This leads to a collision between the frames from station Band station A. On the other hand, suppose that station C starts to send a frame before time *t* + *Tjr-* Here, there is also a collision between frames from station B and station C. Looking at Figure 12.4, we see that the vulnerable time during which a collision may occur in pure ALOHA is 2 times the frame transmission time.

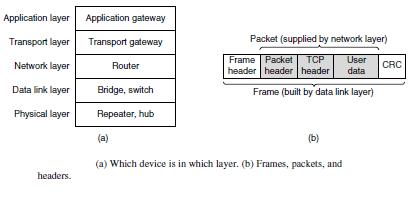
**Pure ALOHA vulnerable time = 2 x *Tfr***

**8.Explain about Connnecting devices.**

**ans.Repeaters, Hubs, Bridges, Switches, Routers, and Gateways**

The key to understanding these devices is to realize that they operate in different layers, as illustrated in Fig. 4-45(a). The layer matters because different devices use different pieces of information to decide how to switch. In a typical scenario, the user generates some data to be sent to a remote machine. Those data are passed to the transport layer, which then adds a header (for example, a TCP header) and passes the resulting unit down to the network layer. The etwork layer adds its own header to form a network layer packet (e.g., an IP packet). In Fig. we see the IP packet shaded in gray. Then the packet goes to the data link layer, which adds its own header and checksum (CRC) and gives the resulting frame to the physical layer for transmission, for example, over a LAN

.



Now let us look at the switching devices and see how they relate to the packets and frames. At the bottom, in the physical layer, we find the repeaters. These are analog devices that work with signals on the cables to which they are connected. A signal appearing on one cable is cleaned up, amplified, and put out on another cable. Repeaters do not understand frames, packets, or headers. They understand the symbols that encode bits as volts. Classic Ethernet, for example, was

designed to allow four repeaters that would boost the signal to extend the maximum cable length from 500 meters to 2500 meters. Next we come to the hubs. A hub has a number of input lines that it joins electrically. Frames arriving on any of the lines are sent out on all the others. If two frames arrive at the same time, they will collide, just as on a coaxial cable. All the lines coming into a hub must operate at the same speed. Hubs differ from repeaters in that they do not (usually) amplify the incoming signals and are designed for multiple input lines, but the differences are slight. Like repeaters, hubs are physical layer devices that do not examine the link layer addresses or use them in any way.

Now let us move up to the data link layer, where we find bridges and switches. We just studied bridges at some length. A bridge connects two or more LANs. Like a hub, a modern bridge has multiple ports, usually enough for 4 to 48 input lines of a certain type. Unlike in a hub, each port is isolated to be its own collision domain; if the port has a full-duplex point-to-point line, the CSMA/CD algorithm is not needed. When a frame arrives, the bridge extracts the destination address from the frame header and looks it up in a table to see where to send the frame. For Ethernet, this address is the 48-bit destination address shown in Fig. The bridge only outputs the frame on the port where it is needed and can forward multiple frames at the same time.

Bridges offer much better performance than hubs, and the isolation between bridge ports also means that the input lines may run at different speeds, possibly even with different network types. A common example is a bridge with ports that connect to 10-, 100-, and 1000-Mbps Ethernet. Buffering within the bridge is needed to accept a frame on one port and transmit the frame out on a different port. If frames come in faster than they can be retransmitted, the bridge may run out of buffer space and have to start discarding frames. For example, if a gigabit Ethernet is pouring bits into a 10-Mbps Ethernet at top speed, the bridge will have to buffer them, hoping not to run out of memory. This problem still exists even if all the ports run at the same speed because more than one port may be sending frames to a given destination port.

Bridges were originally intended to be able to join different kinds of LANs, for example, an Ethernet and a Token Ring LAN. However, this never worked well because of differences between the LANs. Different frame formats require copying and reformatting, which takes CPU time, requires a new checksum calculation, and introduces the possibility of undetected errors due to bad bits in the bridge’s memory. Different maximum frame lengths are also a serious problem with no good solution. Basically, frames that are too large to be forwarded must be discarded. So much for transparency.

Two other areas where LANs can differ are security and quality of service. Some LANs have link-layer encryption, for example 802.11, and some do not, for example Ethernet. Some LANs have quality of service features such as priorities, for example 802.11, and some do not, for example Ethernet. Consequently, when a frame must travel between these LANs, the security or quality of service expected by the sender may not be able to be provided. For all of these reasons, modern bridges usually work for one network type, and routers, which we will come to soon, are used instead to join networks of different types. Switches are modern bridges by another name.

UNIT-3

Short Answer Questions

1. Explain Design Issues Of Network layer?

ans. These issues include the service provided to the transport layer and the internal design of the network.

**Services Provided to the Transport Layer**

The network layer provides services to the transport layer at the network layer/transport layer interface. An important question is precisely what kind of services the network layer provides to the transport layer. The services need to be carefully designed with the following goals in mind:

1.The services should be independent of the router technology.

2.The transport layer should be shielded from the number, type, and topology of the routers

Present.

2. Define virtual circuit?

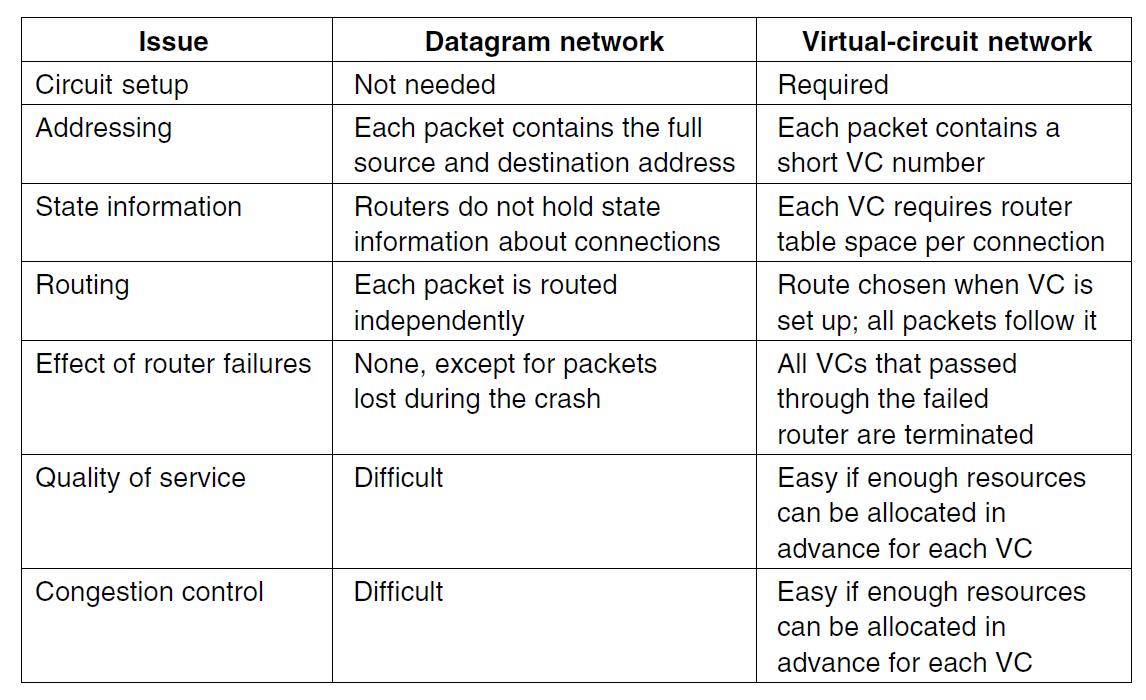
ans. . If connection-oriented service is used, a path from the source router all the way to the destination router must be established before any data packets can be sent. This connection is called a **VC** (**virtual circuit**), in analogy with the physical circuits set up by the

telephone system, and the network is called a **virtual-circuit network.**

**3.** Define datagram’s?

ans.If connectionless service is offered, packets are injected into the network individually and routed independently of each other. No advance setup is needed. In this context, the packets are frequently called **datagrams** (in analogy with telegrams) and the network is called a **datagram network**.

**4.Comparison of Virtual-Circuit and Datagram Networks**

****

5. Define packet switching?

ans.packet switching, only packet switching is used at the network layer because the unit of data at this layer is a packet. At the Network layer, a message from the upper layer is divided into manageable packets and each packet is sent through the network. The source of the message sends the packets one by one; the destination of the message receives the packets one by one. The destination waits for all packets belonging to the same message to arrive before delivering the message to the upper layer.

6. State circuit switching?

ans. circuit switching and packet switching, only packet switching is used at the network layer because the unit of data at this layer is a packet. Circuit switching is mostly used lat the physical layer; the electrical switch mentioned earlier is a kind of circuit switch.

7. Explain Optimality principle?

Before we get into specific algorithms, it may be helpful to note that one can make a general statement about optimal routes without regard to network topology or traffic. This statement is known as the **optimality principle** (Bellman, 1957).

8.what is flooding.  
 A simple local technique is **flooding**, in which every incoming packet is sent out on every outgoing line except the one it arrived on. Flooding obviously generates vast numbers of duplicate packets, in fact, an infinite number unless some measures are taken to damp the process.

9. Define Congestion.

Too many packets present in (a part of) the network causes packet delay and loss that degrades performance. This situation is called **congestion**. The network and transport layers share the responsibility for handling congestion.

10.what is hierachial routing.

ans. As networks grow in size, the router routing tables grow proportionally. Not only is router memory consumed by ever-increasing tables, but more CPU time is needed to scan them and more bandwidth is needed to send status reports about them. At a certain point, the network may grow to the point where it is no longer feasible for every router to have an entry for every other router, so the routing will have to be done hierarchically, as it is in the telephone network. When hierarchical routing is used, the routers are divided into what we will call **regions**

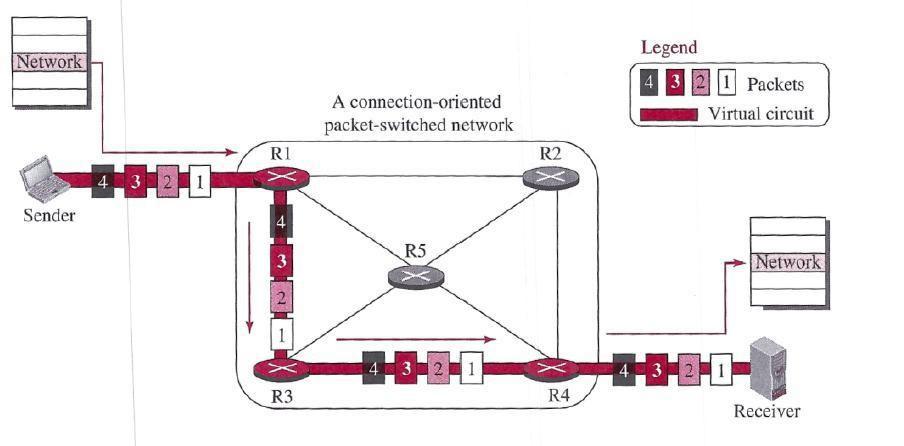
Long Answer Questions.

1. **What is Virtual-Circuit Approach?**

**ans.Virtual-Circuit Approach:**

**Connection-Oriented Service**

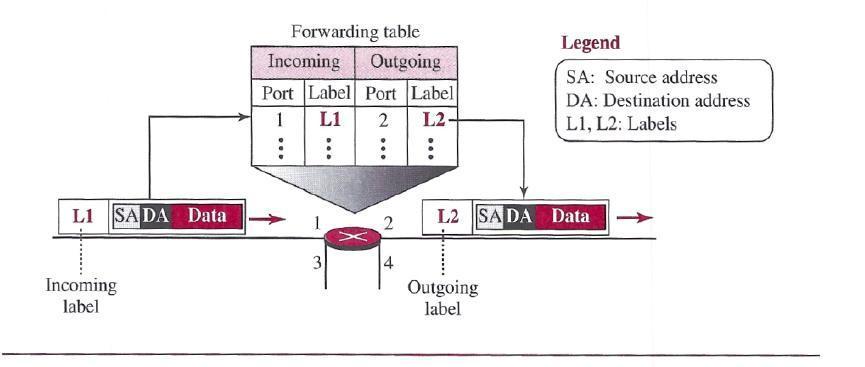
In a connection-oriented service (also called *virtual-circuit approach),* there is a relationship between all packets belonging to a message. Before all datagrams in a message can be sent, a virtual connection should be set up to define the path for the datagrams. After connection setup, the datagrams can all follow the same path. In this type of service, not only must the packet contain the source and destination addresses, it must also contain a flow label, a virtual circuit identifier that defines the virtual path the packet should follow. Shortly, we will show how this flow label is determined, but for the moment, we assume that the packet carries this label. Although it looks as though the use of the label may make the source and destination addresses unnecessary during the data transfer phase, parts of the Internet at the network layer still keep these addresses. One reason is that part of the packet path may still be using the connectionless service. Another reason is that the protocol at the network layer is designed with these addresses, and it may take a while before they can be changed. Figure shows the concept of connection-oriented service.



A virtual-circuit packet-switched network

Each packet is forwarded based on the label in the packet. To follow the idea of connection-oriented design to be used in the Internet, we assume that the packet has a label when it reaches the router. Figure 18.6 shows the idea. In this case, the forwarding decision is based on the value of the label, or *virtual circuit identifier,* as it is sometimes called. To create a connection-oriented service, a three-phase process is used: setup, data transfer, and teardown. In the setup phase, the source and destination addresses of the sender and receiver are used to make table entries for the connection-oriented service. In the teardown phase, the source and destination inform the router to delete the corresponding entries. Data transfer occurs between these two phases.

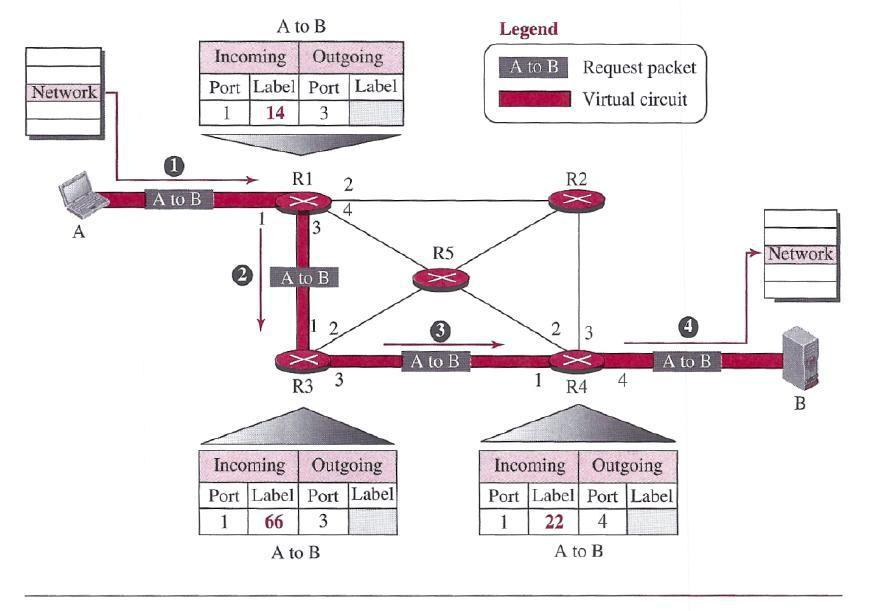
***Setup Phase***

In the setup phase, a router creates an entry for a virtual circuit. For example, suppose source .A needs to create a virtual circuit to destination B. Two auxiliary packets need to be exchanged between the sender and the receiver: the request packet and the acknowledgment packet

Forwarding process in a router when used in a virtual-circuit network

***Request packet***

A request packet is sent from the source to the destination. This auxiliary packet carries the source and destination addresses. Figure shows the process.



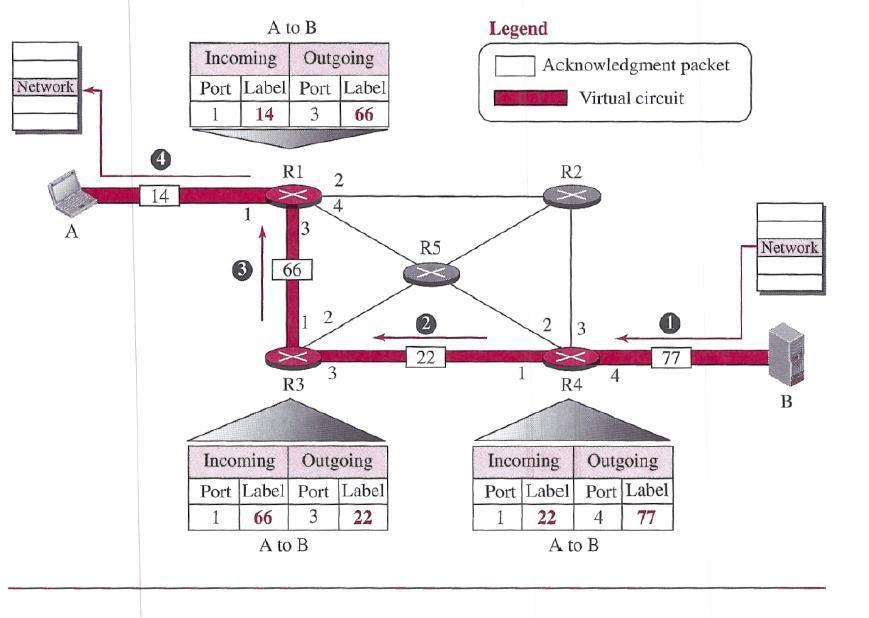
Sending request packet in a virtual-circuit network

1. Source A sends a request packet to router Rl.

Router Rl receives the request packet. It knows that a packet going from A to B goes 0rt through port 3. How the router has obtained this information is a point covered later. For the moment, assume that it knows the output port. The router creates an entry in its table for this virtual circuit, but it is only able to fill three of the four columns. The router assigns the incoming port (1) and chooses an available in rooming label (14) and the outgoing port (3). It does not yet know the outgoing label, which will be found during the acknowledgment step. The router then forwards the packet through port 3 to router R3.

1. Router iR3 receives the setup request packet. The same events happen here as at router Rl; three columns of the table are completed: in this case, incoming port (1)incoming label (66), and outgoing port (3).
2. Router R4 receives the setup request packet. Again, three columns are completed: incoming port (1), incoming label (22), and outgoing port (4).
3. Destination B receives the setup packet, and if it is ready to receive packets from A, it assigns a label to the incoming packets that come from A, in this case 77, as shown in Figure. This label lets the destination know that the packets come from A,I and not from other sources.

***Acknowledgment Packet***

A special Packet, called the acknowledgment packet, completes the entries in the switching tables. Figure shows the process.

Sending acknowledgments in a virtual-circuit network

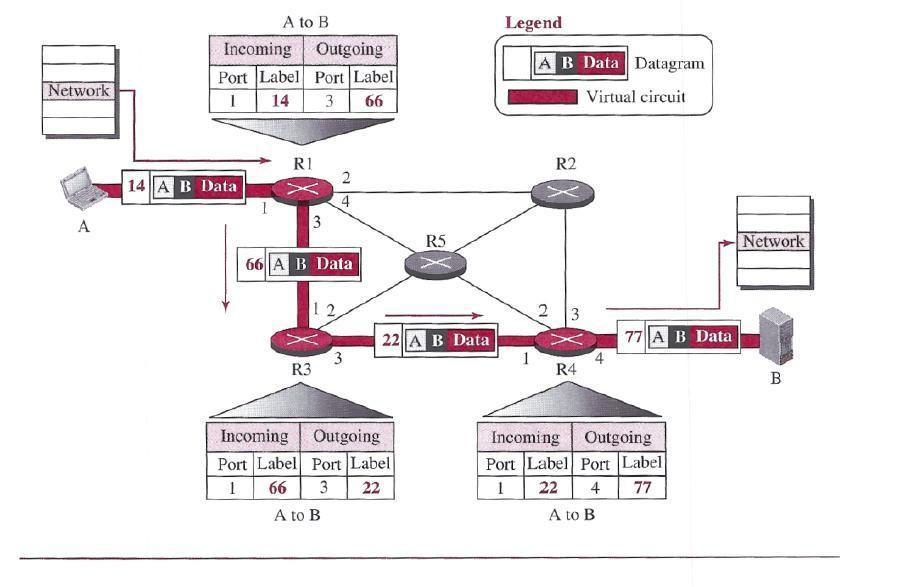
1. The destination sends an acknowledgment to router R4. The acknowledgment carries the global source and destination addresses so the router knows which entry in the table is to be completed. The packet also carries label 77, chosen by the destination as the incoming label for packets from A. Router R4 uses this label to complete the outgoing label column for this entry. Note that 77 is the incoming label for destination B, but the outgoing label for router R4.
   1. Router R4 sends an acknowledgment to router R3 that contains its incoming label in the

table, chosen in the setup phase. Router R3 uses this as the outgoing label in the table.

1. Router R3 sends an acknowledgment to router Rl that contains its incoming label in the table, chosen in the setup phase. Router RI uses this as the outgoing label in the table.
2. Finally router Rl sends an acknowledgment to source A that contains its incoming label in the table, chosen in the setup phase.
3. The source uses this as the outgoing label for the data packets to be sent to destination B.

**Data- Transfer Phase**

The second phase is called the data-transfer phase. After all routers have created their forwarding table for a specific virtual circuit, then the network-layer packets belonging to one message can be sent one after another. In Figure, we show the flow of a single packet, but the process is the same for 1, 2, or 100 packets. The source computer uses the label 14, which it has received from router Rl in the setup.



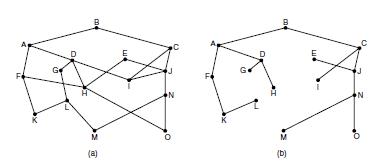
Flow of one packet in an established virtual circuit

phase. Router Rl forwards the packet to router R3, but changes the label to 66.Router R3I forwards the packet to router R4, but changes the label to 22. Finally, router R4 delivers the packet to its final destination with the label 77. All the packets in the message follow the same sequence of labels, and the packets arrive in order at the destination. *Teardown Phase* In the teardown phase, source A, after sending all packets to B, sends a special packet called a teardown packet. Destination B responds with a confirmation packet. All routers delete the corresponding entries from their tables.

2.Explain optimality Principle?

**The Optimality Principle**

Before we get into specific algorithms, it may be helpful to note that one can make a general statement about optimal routes without regard to network topology or traffic. This statement is known as the **optimality principle** (Bellman, 1957). It states that if router *J* is on the optimal path from router *I* to router *K*, then the optimal path from *J* to *K* also falls along the same route. To see this, call the part of the route from *I* to *J r*1 and the rest of the route *r* 2*.* If a route better than *r* 2 existed from *J* to *K*, it could be concatenated with *r* 1 to improve the route from *I* to *K*, contradicting our statement that *r* 1*r* 2 is optimal. As a direct consequence of the optimality principle, we can see that the set of optimal routes from all sources to a given destination form a tree rooted at the destination. Such a tree is called a **sink tree** and is illustrated in Fig. where the distance metric is the number of hops. The goal of all routing algorithms is to discover and use the sink trees for all routers.



(a) A network. (b) A sink tree for router *B*.

Note that a sink tree is not necessarily unique; other trees with the same path lengths may exist. If we allow all of the possible paths to be chosen, the tree becomes a more general structure called a **DAG** (**Directed Acyclic Graph**). DAGs have no loops. We will use sink trees as convenient shorthand for both cases. Both cases also depend on the technical assumption that the paths do not interfere with each other so, for example, a traffic jam on one path will not cause another path to divert. Since a sink tree is indeed a tree, it does not contain any loops, so each

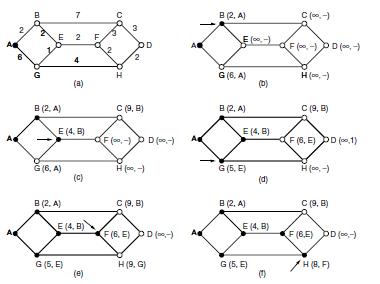
packet will be delivered within a finite and bounded number of hops. In practice, life is not quite this easy. Links and routers can go down and come back up during operation, so different routers may have different ideas about the current topology. Also, we have quietly finessed the issue of whether each router has to individually acquire the information on which to base its sink tree computation or whether this information is collected by some other means. We will come back to these issues shortly. Nevertheless, the optimality principle and the sink tree provide a benchmark against which other routing algorithms can be measured.

routing tables. This process is **forwarding**. The other process is responsible for filling in and updating the routing tables. That is where the routing algorithm comes into play.

4.Explain  **Shortest Path Algorithm**

Let us begin our study of routing algorithms with a simple technique for computing optimal paths given a complete picture of the network. These paths are the ones that we want a distributed routing algorithm to find, even though not all routers may know all of the details of the network. The idea is to build a graph of the network, with each node of the graph representing a router and each edge of the graph representing a communication line, or link. To choose a route between a given pair of routers, the algorithm just finds the shortest path between them on the graph.

The concept of a **shortest path** deserves some explanation. One way of measuring path length is the number of hops. Using this metric, the paths *ABC* and *ABE* in Fig. are equally long. Another metric is the geographic distance in kilometers, in which case *ABC* is clearly much longer than *ABE* (assuming the figure is drawn to scale).



The first six steps used in computing the shortest path from *A* to *D*. The arrows indicate the working node.

However, many other metrics besides hops and physical distance are also possible. For example, each edge could be labeled with the mean delay of a standard test packet, as measured by hourly runs. With this graph labeling, the shortest path is the fastest path rather than the path with the fewest edges or kilometers. In the general case, the labels on the edges could be computed as a function of the distance, bandwidth, average traffic, communication cost, measured delay, and other factors. By changing the weighting function, the algorithm would then compute the

‘‘shortest’’ path measured according to any one of a number of criteria or to a combination of criteria. Several algorithms for computing the shortest path between two nodes of a graph are known. This one is due to Dijkstra (1959) and finds the shortest paths between a source and all destinations in the network. Each node is labeled (in parentheses) with its distance from the source node along the best known path. The distances must be non-negative, as they will be if they are based on real quantities like bandwidth and delay. Initially, no paths are known, so all nodes are labeled with infinity. As the algorithm proceeds and paths are found, the labels may change, reflecting better paths. A label may be either tentative or permanent. Initially, all labels are tentative. When it is discovered that a label represents the shortest possible path from the source to that node, it is made permanent and never changed thereafter. To illustrate how the labeling algorithm works, look at the weighted, undirected graph of Fig., where the weights represent, for example, distance. We want to find the shortest path from *A* to *D*. We start out by marking node *A* as permanent, indicated by a filled-in circle. Then we examine, in turn, each of the nodes adjacent to *A* (the working node), relabeling each one with the distance to *A*. Whenever a node is relabeled, we also label it with the node from which the probe was made so that we can reconstruct the final path later. If the network had more than one shortest path from *A* to *D* and we wanted to find all of them, we would need to remember all of the probe nodes that could reach a node with the same distance. Having examined each of the nodes adjacent to *A*, we examine all the tentatively labeled nodes in the whole graph and make the one with the smallest label permanent, as shown in Fig. (b). this one becomes the new working node. We now start at *B* and examine all nodes adjacent to it. If the sum of the label on *B* and the distance from *B* to thenode being considered is less than the label on that node, we have a shorter path, so the node is relabeled. After all the nodes adjacent to the working node have been inspected and the tentative labels changed if possible, the entire graph is searched for the tentatively labeled node with the smallest value. This node is made permanent and becomes the working node for the next round. Figure shows the first six steps of the algorithm. To see why the algorithm works, look at Fig.

(c). At this point we have just made *E* permanent. Suppose that there were a shorter path than *ABE*, say *AXYZE* (for some *X* and *Y*). There are two possibilities: either node *Z* has already beenmade permanent, or it has not been. If it has, then *E* has already been probed (on the round following the one when *Z* was made permanent), so the *AXYZE* path has not escaped our attention and thus cannot be a shorter path. Now consider the case where *Z* is still tentatively labeled. If the label at *Z* is greater than or equal to that at *E*, then *AXYZE* cannot be a shorter path than *ABE*. If the label is less than that of *E*, then *Z* and not *E* will become permanent first, allowing *E* to be probed from *Z*. This algorithm is given in Fig. The global variables *n* and *dist* describe the graph and are initialized before *shortest path* is called. The only difference between the program and the algorithm described above is that in Fig., we compute the shortest path starting at the terminal node, *t*, rather than at the source node, *s*. Since the shortest paths from *t* to *s* in an undirected graph are the same as the shortest paths from *s* to *t*, it does not matter at whichend we begin. The reason for searching backward is that each node is labeled with its predecessor rather than its successor. When the final path is copied into the output variable, *path*, the path is thus reversed. The two reversal effects cancel, and the answer is produced in the correct order.

5. **Explain the concept of Flooding?**

When a routing algorithm is implemented, each router must make decisions based on local knowledge, not the complete picture of the network. A simple local technique is **flooding**, in

which every incoming packet is sent out on every outgoing line except the one it arrived on. Flooding obviously generates vast numbers of duplicate packets, in fact, an infinite number unless some measures are taken to damp the process. One such measure is to have a hop counter contained in the header of each packet that is decremented at each hop, with the packet being discarded when the counter reaches zero. Ideally, the hop counter should be initialized to the length of the path from source to destination. If the sender does not know how long the path is, it can initialize the counter to the worst case, namely, the full diameter of the network.

Flooding with a hop count can produce an exponential number of duplicate packets as the hop count grows and routers duplicate packets they have seen before. A better technique for damming the flood is to have routers keep track of which packets have been flooded, to avoid sending them out a second time. One way to achieve this goal is to have the source router put a sequence number in each packet it receives from its hosts. Each router then needs a list per source router telling which sequence numbers originating at that source have already been seen. If an incoming packet is on the list, it is not flooded.

#define MAX NODES 1024 /\* maximum number of nodes \*/

#define INFINITY 1000000000 /\* a number larger than every maximum path \*/ int n, dist[MAX NODES][MAX NODES]; /\* dist[i][j] is the distance from i to j \*/ void shortest path(int s, int t, int path[])

{ struct state { /\* the path being worked on \*/ int predecessor; /\* previous node \*/

int length; /\* length from source to this node \*/ enum {permanent, tentative} label; /\* label state \*/ } state[MAX NODES];

int i, k, min; struct state \*p;

for (p = &state[0]; p < &state[n]; p++) { /\* initialize state \*/ p->predecessor = −1;

p->length = INFINITY; p->label = tentative;

}

state[t].length = 0; state[t].label = permanent; k = t; /\* k is the initial working node \*/

do { /\* Is there a better path from k? \*/

for (i = 0; i < n; i++) /\* this graph has n nodes \*/ if (dist[k][i] != 0 && state[i].label == tentative) { if (state[k].length + dist[k][i] < state[i].length) { state[i].predecessor = k;

state[i].length = state[k].length + dist[k][i];

}

}

/\* Find the tentatively labeled node with the smallest label. \*/ k = 0; min = INFINITY;

for (i = 0; i < n; i++)

if (state[i].label == tentative && state[i].length < min) { min = state[i].length;

k = i;

}

state[k].label = permanent; } while (k != s);

/\* Copy the path into the output array. \*/ i = 0; k = s;

do {path[i++] = k; k = state[k].predecessor; } while (k >= 0);

}

Dijkstra’s algorithm to compute the shortest path through a graph

To prevent the list from growing without bound, each list should be augmented by a counter, *k*, meaning that all sequence numbers through *k* have been seen. When a packet comes in, it is easy to check if the packet has already been flooded (by comparing its sequence number to *k*; if so, it is discarded. Furthermore, the full list below *k* is not needed, since *k* effectively summarizes it. Flooding is not practical for sending most packets, but it does have some important uses. First, it ensures that a packet is delivered to every node in the network. This may be wasteful if there is a single destination that needs the packet, but it is effective for broadcasting information. In wireless networks, all messages transmitted by a station can be received by all other stations within its radio range, which is, in fact, flooding, and some algorithms utilize this property. Second, flooding is tremendously robust. Even if large numbers of routers are blown to bits (e.g., in a military network located in a war zone), flooding will find a path if one exists, to get a packet to its destination. Flooding also requires little in the way of setup. The routers only need to know their neighbors. This means that flooding can be used as a building block for other routing algorithms that are

more efficient but need more in the way of setup. Flooding can also be used as a metric against which other routing algorithms can be compared. Flooding always chooses the shortest path because it chooses every possible path in parallel. Consequently, no other algorithm can produce a shorter delay (if we ignore the overhead generated by the flooding process itself).

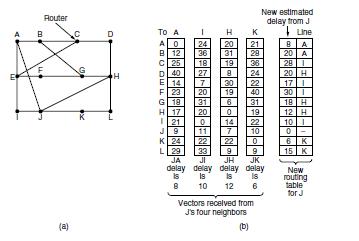
6. Describe **Distance Vector Routing?**

Computer networks generally use dynamic routing algorithms that are more complex than flooding, but more efficient because they find shortest paths for the current topology. Two dynamic algorithms in particular, distance vector routing and link state routing, are the most popular. In this section, we will look at the former algorithm. In the following section, we will study the latter algorithm. A **distance vector routing** algorithm operates by having each router maintain a table (i.e., a vector) giving the best known distance to each destination and which link to use to get there. These tables are updated by exchanging information with the neighbors. Eventually, every router knows the best link to reach each destination. The distance vector routing algorithm is sometimes called by other names, most commonly the distributed **Bellman-Ford** routing algorithm, after the researchers who developed it (Bellman, 1957; and Ford andFulkerson, 1962). It was the original ARPANET routing algorithm and was also used in the Internet under the name RIP. In distance vector routing, each router maintains a routing table indexed by, and containing one entry for each router in the network. This entry has two parts: the preferred outgoing line to use for that destination and an estimate of the distance to that destination. The distance might be measured as the number of hops or using another metric, as we discussed for computing shortest paths. The router is assumed to know the ‘‘distance’’ to

each of its neighbors. If the metric is hops, the distance is just one hop. If the metric is propagation delay, the router can measure it directly with special ECHO packets that the receiver just timestamps and sends back as fast as it can. As an example, assume that delay is used as a metric and that the router knows the delay to each of its neighbors. Once every *T* m sec, each router sends to each neighbor a list of its estimated delays to each destination. It also receives a similar list from each neighbor. Imagine that one of these tables has just come in from neighbor *X*, with *Xi* being *X*’sestimate of how long it takes to get to router *i*.If the router knows that thedelay to *X* is *m* m sec, it also knows that it can reach router *i* via *X* in *Xi m* msec. By performing this calculation for each neighbor, a router can find out which estimate seems the best and use that estimate and the corresponding link in its new routing table. Note that the old routing table is not used in the calculation. This updating process is illustrated in Fig. 5-9. Part



(a) shows a network. The first four columns of part (b) show the delay vectors received from the neighbors of router *J*. *A* claims to have a 12-msec delay to *B*, a 25-msec delay to *C*, a 40-msec delay to *D*, etc. Suppose that *J* has measured or estimated its delay to its neighbors, *A*, *I, H*, and *K*, as 8, 10, 12, and 6 m sec, respectively.



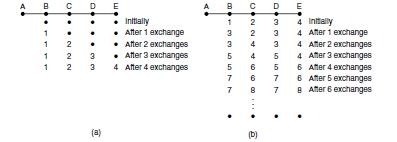
(a) A network. (b) Input from *A*, *I*, *H*, *K*, and the new routing table for *J*.

Consider how *J* computes its new route to router *G*. It knows that it can get to *A* in 8 m sec, and furthermore *A* claims to be able to get to *G* in 18 m sec, so *J* knows it can count on a delay of 26 m sec to *G* if it forwards packets bound for *G* to *A*. Similarly, it computes the delay to *G* via *I*, *H*, and *K* as 41 (31 + 10), 18 (6 + 12), and 37 (31 + 6) m sec, respectively. The best of these values is 18, so it makes an entry in its routing table that the delay to *G* is 18 m sec and that the route to use is via *H*. The same calculation is performed for all the other destinations, with the new routing table shown in the last column of the figure.

7. **Explain The Count-to-Infinity Problem?**

The settling of routes to best paths across the network is called **convergence**. Distance vector routing is useful as a simple technique by which routers can collectively compute shortest paths, but it has a serious drawback in practice: although it converges to the correct answer, it may do so slowly. In particular, it reacts rapidly to good news, but leisurely to bad news. Consider a router whose best route to destination *X* is long. If, on the next exchange, neighbor *A* suddenly reports a short delay to *X*, the router just switches over to using the line to *A* to send traffic to *X*. In one vector exchange, the good news is processed. To see how fast good news propagates

consider the five-node (linear) network of Fig. 5-10, where the delay metric is the number of hops. Suppose *A* is down initially and all the other routers know this. In other words, they have all recorded the delay to *A* as infinity.



The count-to-infinity problem.

When *A* comes up, the other routers learn about it via the vector exchanges. For simplicity, we will assume that there is a gigantic going somewhere that is struck periodically to initiate a vector exchange at all routers simultaneously. At the time of the first exchange, *B* learns that its left-hand neighbor has zero delay to *A*. *B* now makes an entry in

its routing table indicating that *A* is one hop away to the left. All the other routers still think that *A* is down. At this point, therouting table entries for *A* are as shown in the second row of Fig. (a). On the next exchange, *C* learns that *B* has a path of length 1 to *A*, so it updates its routing table to indicate a path of length 2, but *D* and *E* do not hear the good news until later. Clearly, the good news is spreading at the rate of one hop per exchange. In a network whose longest path is of length *N* hops, within *N* exchanges everyone will know about newly revived links and routers. Now let us consider the situation of (b), in which all the links and routers are initially up. Routers *B*, *C*, *D*, and *E* have distances to *A* of 1, 2, 3, and 4 hops, respectively. Suddenly, either *A* goes down or the link between *A* and *B* is cut (which is effectively the same thing from *B*’s point of view).At the first packet exchange, *B* does not hear anything from *A*. Fortunately, *C* says ‘‘Do not worry; I have a path to *A* of length 2.’’ Little does *B* suspect that *C*’s path runs through *B* itself . For all *B* knows, *C* might have ten links all with separate paths to *A* of length 2. As a result, *B* thinks it can reach *A* via *C*, with a path length of 3. *D* and *E* do not update their entries for *A* on the first exchange.

On the second exchange, *C* notices that each of its neighbors claims to have a path to *A* of length 3. It picks one of them at random and makes its new distance to *A* 4, as shown in the third row of Fig. 5-10(b). Subsequent exchanges produce the history shown in the rest of Fig. 5-10(b).From this figure, it should be clear why bad news travels Slowly: no router ever has a value more than one higher than the minimum of all its neighbors. Gradually, all routers work their way up to infinity, but the number of exchanges required depends on the numerical value used for infinity. For this reason, it is wise to set infinity to the longest path plus 1.Not entirely surprisingly, this problem is known as the **count-to-infinity** problem. There have been many attempts to solve it, for example, preventing routers from advertising their best paths back to the neighbors from which they heard them with the split horizon with poisoned reverse rule discussed in RFC 1058.However, none of these heuristics work well in practice despite the colorful names. The core of the problem is that when *X* tells *Y* that it has a path somewhere, *Y* has no way of knowing whether it itself is on the path.

8.Descibe  **Hierarchical Routing**

As networks grow in size, the router routing tables grow proportionally. Not only is router memory consumed by ever-increasing tables, but more CPU time is needed to scan them and more bandwidth is needed to send status reports about them. At a certain point, the network may grow to the point where it is no longer feasible for every router to have an entry for every other router, so the routing will have to be done hierarchically, as it is in the telephone network. When hierarchical routing is used, the routers are divided into what we will call **regions**. Each router knows all the details about how to route packets to destinations within its own region but knows nothing about the internal structure of other regions. When different networks are interconnected, it is natural to regard each one as a separate region to free the routers in one network from having to know the topological structure of the other ones. For huge networks, a two-level hierarchy may be insufficient; it may be necessary to group the regions into clusters, the clusters into zones, the zones into groups, and so on, until we run out of names for aggregations. As an example of a multilevel hierarchy, consider how a packet might be routed from Berkeley, California, to Malindi, Kenya. The Berkeley router would know the detailed topology within California but would send all out-of-state traffic to the Los Angeles router. The Los Angeles router would be able to route traffic directly to other domestic routers but would send all foreign traffic to New York. The New York router would be programmed to direct all traffic to the router in the destination country responsible for handling foreign traffic, say, in Nairobi. Finally, the packet would work its way down the tree in Kenya until it got to Malindi. Figure gives a quantitative example of routing in a two-level hierarchy with five regions. The full routing table for router *1A* has 17 entries, as shown in Fig. (b). When routing is done hierarchically, as in Fig. 5-14(c), there are entries for all the local routers, as before, but all other regions are condensed into a single router, so all traffic for region 2 goes via the *1B-2A* line, but the rest of the remote traffic goes via the *1C-3B* line. Hierarchical routing has reduced the table from 17 to 7 entries. As the ratio of the number of regions to the number of routers per region grows, the savings in table space increase. Unfortunately, these gains in space are not free. There is a penalty to be paid: increased path length. For example, the best route from *1A* to *5C* is via region 2,but with hierarchical routing all traffic to region 5 goes via region 3, because that is better for most destinations in region 5.When a single network becomes very large, an interesting question is

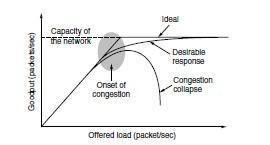
‘‘how many levels should the hierarchy have?’’ For example, consider a network with 720 routers. If there is no hierarchy, each router needs 720 routing table entries. If the network is partitioned into 24 regions of 30 routers each, each router needs 30 local entries plus 23 remote entries for a total of 53 entries. If a three-level hierarchy is chosen, with 8 clusters each containing 9 regions of 10 routers, each router needs 10 entries for local routers, 8 entries for routing to other regions within its own cluster, and 7 entries for distant clusters, for a total of 25 entries.Kamoun and Kleinrock (1979) discovered that the optimal number of levels for an *N* router network is ln *N*, requiring a total of *e* ln *N* entries per router. They have also shown that the increase in effective mean path length caused by hierarchical routing is sufficiently small that it is usually acceptable Hierarchical routing.

.



9. **Explain congestion control algorithms**

Ans.Too many packets present in (a part of) the network causes packet delay and loss that degrades performance. This situation is called **congestion**. The network and transport layers share the responsibility for handling congestion. Since congestion occurs within the network, it is the network layer that directly experiences it and must ultimately determine what to do with the excess packets. However, the most effective way to control congestion is to reduce the load that the transport layer is placing on the network. This requires the network and transport layers to work together. In this chapter we will look at the network aspects of congestion. In Chap. 6, we will complete the topic by covering the transport aspects of congestion. Figure depicts the onset of congestion. When the number of packets hosts send into the network is well within its carrying capacity, the number delivered is proportional to the number sent. If twice as many are sent, twice as many are delivered. However, as the offered load approaches the carrying capacity, bursts of traffic occasionally fill up the buffers inside routers and some packets are lost. These lost packets consume some of the capacity, so the number of delivered packets falls below the ideal curve. The network is now congested.



With too much traffic, performance drops sharply.

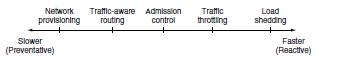
Unless the network is well designed, it may experience a **congestion collapse**, in which performance plummets as the offered load increases beyond the capacity. This can happen because packets can be sufficiently delayed inside the network that they are no longer useful when they leave the network. For example, in the early Internet, the time a packet spent waiting for a backlog of packets ahead of it to be sent over a slow 56-kbps link could reach the maximum time it was allowed to remain in the network. It then had to be thrown away. A different failure mode occurs when senders retransmit packets that are greatly delayed, thinking that they have been lost. In this case, copies of the same packet will be delivered by the network, again wasting its capacity. To capture these factors, the y-axis of Fig. is given as **good put**, which is the rate at which *useful* packets are delivered by the network. We would like to design networks that avoid congestion where possible and do not suffer from congestion collapse if they do become congested. Unfortunately, congestion cannot wholly be avoided. If all of a sudden, streams of packets begin arriving on three or four input lines and all need the same output line, a queue will build up. If there is insufficient memory to hold all of them, packets will be lost. Adding more memory may help up to a point, but Nagle (1987) realized that if routers have an infinite amount of memory, congestion gets worse, not better. This is because by the time packets get to the front of the queue, they have already timed out (repeatedly) and duplicates have been sent. This makes matters worse, not better—it leads to congestion collapse. Low-bandwidth links or routers that process packets more slowly than the line rate can also become congested. In this case, the situation can be improved by directing some of the traffic away from the bottleneck to other parts of the network. Eventually, however, all regions of the network will be congested. In this situation, there is no alternative but to shed load or build a faster network. It is worth pointing out the difference between congestion control and flow control, as the relationship is a very subtle one. Congestion control has to do with making sure the network is able to carry the offered traffic. It is a global issue, involving the behavior of all the hosts and routers. Flow control, in contrast, relates to the traffic between a particular sender and a particular receiver. Its job is to make sure that a fast sender cannot continually transmit data faster than the receiver is able to absorb it. To see the difference between these two concepts, consider a network made up of 100-Gbps fiber optic links on which a supercomputer is trying to force feed a large file to a personal computer that is capable of handling only 1 Gbps. Although there is no congestion (the network itself is not in trouble), flow control is needed to force the supercomputer to stop frequently to give the personal computer chance to breathe. At the other extreme, consider a network with 1-Mbps lines and 1000 large computers, half of which are trying to transfer files at 100 kbps to the other half. Here, the problem is not that of fast senders overpowering slow receivers, but that the total offered traffic exceeds what the network can handle.

The reason congestion control and flow control are often confused is that the best way to handle both problems is to get the host to slow down. Thus, a host can get a ‘‘slow down’’ message either because the receiver cannot handle the load or because the network cannot handle it. We will come back to this point in Chap. 6. We will start our study of congestion control by looking at the approaches that can be used at different time scales. Then we will look at approaches to preventing congestion from occurring in the first place, followed by approaches for coping with it once it has set in.

**Approaches to Congestion Control**

The presence of congestion means that the load is (temporarily) greater than the resources (in a part of the network) can handle. Two solutions come to mind: increase the resources or decrease

the load. As shown in Fig., these solutions are usually applied on different time scales to either prevent congestion or react to it once it has occurred.



Timescales of approaches to congestion control.

The most basic way to avoid congestion is to build a network that is well matched to the traffic that it carries. If there is a low-bandwidth link on the path along which most traffic is directed, congestion is likely. Sometimes resources on spare routers or enabling lines that are normally used only as backups (to make the system fault tolerant) or purchasing bandwidth on the open market. More often, links and routers that are regularly heavily utilized are upgraded at the earliest opportunity. This is called **provisioning** and happens on a time scale of months, driven by long-term traffic trends. To make the most of the existing network capacity, routes can be tailored to traffic patterns that change during the day as network user’s wake and sleep in different time zones. For example, routes may be changed to shift traffic away from heavily used paths by changing the shortest path weights. Some local radio stations have helicopters flying around their cities to report on road congestion to make it possible for their mobile listeners to route their packets (cars) around hotspots. This is called **traffic-aware routing**. Splitting traffic across multiple paths is also helpful. However, sometimes it is not possible to increase capacity. The only way then to beat back the congestion is to decrease the load. In a virtual-circuit network, new connections can be refused if they would cause the network to become congested. This is called **admission control**. At a finer granularity, when congestion is imminent the network can deliver feedback to the sources whose traffic flows are responsible for the problem. The network can request these sources to throttle their traffic, or it can slow down the traffic itself. Two difficulties with this approach are how to identify the onset of congestion, and how to inform the source that needs to slow down. To tackle the first issue, routers can monitor the average load, queuing delay, or packet loss. In all cases, rising numbers indicate growing congestion. To tackle the second issue, routers must participate in a feedback loop with the sources. For a scheme to work correctly, the time scale must be adjusted carefully. If every time two packets arrive in a row, a router yells STOP and every time a router is idle for 20 sec, it yells GO, the system will oscillate wildly and never converge. On the other hand, if it waits 30 minutes to make sure before saying anything, the congestion-control mechanism will react too sluggishly to be of any use. Delivering timely feedback is a nontrivial matter. An added concern is having routers send more messages when the network is already congested. Finally, when all else fails, the network is forced to discard packets that it cannot deliver. The general name for this is **load shedding**. A good policy for choosing which packets to discard can help to prevent congestion collapse.

Bridges were developed when classic Ethernet was in use, so they tend to join relatively few

LANs and thus have relatively few ports. The term ‘‘switch’’ is more popular nowadays. Also, modern installations all use point-to-point links, such as twisted-pair cables, so individual computers plug directly into a switch and thus the switch will tend to have many ports. Finally, ‘‘switch’’ is also used as a general term. With a bridge, the functionality is clear. On the other hand, a switch may refer to an Ethernet switch or a completely different kind of device that makes forwarding decisions, such as a telephone switch. So far, we have seen repeaters and hubs, which are actually quite similar, as well as bridges and switches, which are even more similar to each other. Now we move up to routers, which are different from all of the above. When a packet comes into a router, the frame header and trailer are stripped off and the packet located in the frame’s payload field is passed to the routing software. This software uses the packet header to choose an output line. For an IP packet, the packet header will contain a 32-bit (IPv4) or 128-bit (IPv6) address, but not a 48-bit IEEE 802 address. The routing software does not see the frame addresses and does not even know whether the packet came in on a LAN or a point-to-point line. Up another layer, we find transport gateways. These connect two computers that use different connection-oriented transport protocols. For example, suppose a computer using the connection-oriented TCP/IP protocol needs to talk to a computer using a different connection-oriented transport protocol called SCTP. The transport gateway can copy the packets from one connection to the other, reformatting them as need be.

Finally, application gateways understand the format and contents of the data and can translate messages from one format to another. An email gateway could translate Internet messages into

SMS messages for mobile phones, for example. Like ‘‘switch,’’ ‘‘gateway’’ is somewhat of a general term. It refers to a forwarding process that runs at a high layer.

**UNIT-4**

**1.What is Internetwork Routing?**

Ans.Routing through an internet poses the same basic problem as routing within a single network, but with some added complications. To start, the networks may internally use different routing algorithms. For example, one network may use link state routing and another distance vector routing. Since link state algorithms need to know the topology but distance vector algorithms do not, this difference alone would make it unclear how to find the shortest paths across the internet. Networks run by different operators lead to bigger problems. First, the operators may have different ideas about what is a good path through the network. One operator may want the route with the least delay, while another may want the most inexpensive route. This will lead the operators to use different quantities to set the shortest-path costs (e.g., milliseconds of delay vs. monetary cost). The weights will not be comparable across networks, so shortest paths on the internet will not be well defined. Worse yet, one operator may not want another operator to even know the details of the paths in its network, perhaps because the weights and paths may reflect sensitive information (such as the monetary cost) that represents a competitive business advantage. Finally, the internet may be much larger than any of the networks that comprise it. It may therefore require routing algorithms that scale well by using a hierarchy; even if none of the individual networks need to use a hierarchy. All of these considerations lead to a two-level routing algorithm. Within each network, an **intra domain** or **interior gateway protocol.**

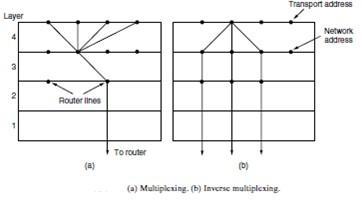
**2.Define Packet Fragmentation**

Each network or link imposes some maximum size on its packets. These limits have various causes, among them

1. Hardware (e.g., the size of an Ethernet frame).
2. Operating system (e.g., all buffers are 512 bytes).
3. Protocols (e.g., the number of bits in the packet length field).
4. Compliance with some (inter)national standard.
5. Desire to reduce error-induced retransmissions to some level.
6. Desire to prevent one packet from occupying the channel too long.

**3. What is Crash Recovery?**

If hosts and routers are subject to crashes or connections are long-lived (e.g., large software or media downloads) recovery from these crashes becomes an issue. If the transport entity is entirely within the hosts, recovery from network

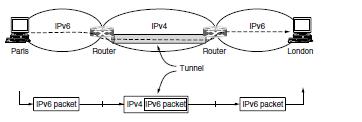


and router crashes is straightforward. The transport entities expect lost segments all the time and know how to cope with them by using retransmissions. A more troublesome problem is how to recover from host crashes. In particular, it may be desirable for clients to be able to continue working when servers crash and quickly reboot. To illustrate the difficulty, let us assume that one host, the client, is sending a long file to another host, the file server, using a simple Stop-and-wait protocol. The transport layer on the server just passes the incoming segments to the transport user, one by one. Partway through the transmission, the server crashes.

**4. Define Tunneling**

|  |  |
| --- | --- |
|  |  |

Ans.Handling the general case of making two different networks interwork is exceedingly difficult. However, there is a common special case that is manageable even for different network protocols. This case is where the source and destination hosts are on the same type of network, but there is a different network in between. As an example, think of an international bank with an IPv6 network in Paris, an IPv6 network in London and connectivity between the offices via the IPv4 Internet.

This situation is shown in Fig.

**5.what are Limits of Packet fragmentation?**

**Ans.**

**Packet Fragmentation**

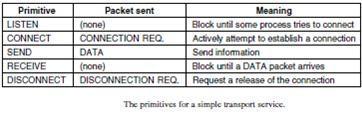
Each network or link imposes some maximum size on its packets. These limits have various causes, among them

1. Hardware (e.g., the size of an Ethernet frame).
2. Operating system (e.g., all buffers are 512 bytes).
3. Protocols (e.g., the number of bits in the packet length field).
4. Compliance with some (inter)national standard.
5. Desire to reduce error-induced retransmissions to some level.
6. Desire to prevent one packet from occupying the channel too long.

**6.what are the Services Provided to the Upper Layers?**

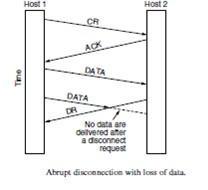
The ultimate goal of the transport layer is to provide efficient, reliable, and cost-effective data transmission service to its users, normally processes in the application layer. To achieve this, the transport layer makes use of the services provided by the network layer. The software and/or hardware within the transport layer that does the work is called the **transport entity**. The transport entity can be located in the operating system kernel, in a library package bound into network applications, in a separate user process, or even on the network interface card. The first two options are most common on the Internet.

**7.What are primitives of simple transport service?**



**8. Explain Connection Release?**

Ans.Releasing a connection is easier than establishing one. Nevertheless, there are more pitfalls than one might expect here. As we mentioned earlier, there are two styles of terminating a connection: asymmetric release and symmetric release Asymmetric release is the way the telephone system works: when one party hangs up, the connection is broken. Symmetric release treats the connection as two separate unidirectional connections and requires each one to be released separately. Asymmetric release is abrupt and may result in data loss. Consider the scenario of Fig. After the connection is established, host 1 sends a segment that arrives properly at host 2. Then host 1 sends another segment. Unfortunately, host 2 issues a DISCONNECT before the second segment arrives. The result is that the connection is released and data are lost.



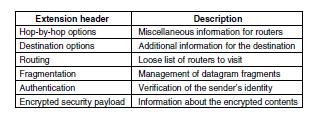
**9.What is the lifetime of packet?**

Ans. Packet lifetime can be restricted to a known maximum using one (or more) of the following techniques:

1. Restricted network design.
2. Putting a hop counter in each packet.
3. Time stamping each packet.

**10. What are Extension Headers?**

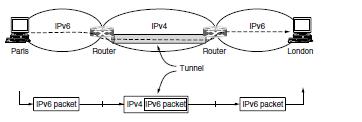
Ans.Some of the missing IPv4 fields are occasionally still needed, so IPv6 introduces the concept of (optional) **extension headers**. These headers can be supplied to provide extra information, but encoded in an efficient way. Six kinds of extension headers are defined at present, as listed in Fig. Each one is optional, but if more than one is present they must appear directly after the fixed header, and preferably in the order listed.



IPv6 extension headers

Long Answer Questions

**1.Explain the concept of Tunneling?**

Ans.Handling the general case of making two different networks interwork is exceedingly difficult. However, there is a common special case that is manageable even for different network protocols. This case is where the source and destination hosts are on the same type of network, but there is a different network in between. As an example, think of an international bank with an IPv6 network in Paris, an IPv6 network in London and connectivity between the offices via the IPv4 Internet. This situation is shown in Fig

Tunneling a packet from Paris to London.

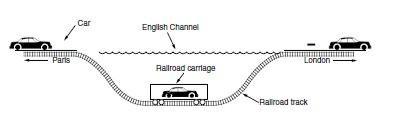
The solution to this problem is a technique called **tunneling**. To send an IP packet to a host in the London office, a host in the Paris office constructs the packet containing an IPv6 address in London, and sends it to the multiprotocol router that connects the Paris IPv6 network to the IPv4 Internet. When this router gets the IPv6 packet, it encapsulates the packet with an IPv4 header addressed to the IPv4 side of the multiprotocol router that connects to the London IPv6 network.

That is, the router puts a (IPv6) packet inside a (IPv4) packet. When this wrapped packet arrives, the London router removes the original IPv6 packet and sends it onward to the destination host.

The path through the IPv4 Internet can be seen as a big tunnel extending from one multiprotocol router to the other. The IPv6 packet just travels from one end of the tunnel to the other, snug in its nice box. It does not have to worry about dealing with IPv4 at all. Neither do the hosts in Paris or London. Only the multiprotocol routers have to understand both IPv4 and IPv6 packets. In effect, the entire trip from one multiprotocol router to the other is like a hop over a single link.

An analogy may make tunneling clearer. Consider a person driving her car from Paris to London. Within France, the car moves under its own power, but when it hits the English Channel, it is loaded onto a high-speed train and transported to England through the Chunnel (cars are not permitted to drive through the Chunnel). Effectively, the car is being carried as freight, as depicted in Fig. At the far end, the car is let loose on the English roads and once again continues to move under its own power. Tunneling of packets through a foreign network works the same way. Tunneling is widely used to connect isolated hosts and networks using other networks. The network that results is called an **overlay** since it has effectively been overlaid on the base network. Deployment of a network protocol with a new feature is a common reason, as our

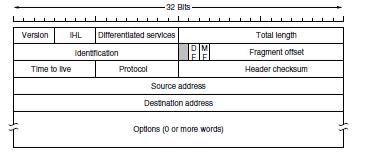
‘‘IPv6 over IPv4’’ example shows. The disadvantage of tunneling is that none of the hosts on the network that are tunneled over can be reached because the packets cannot escape in the middle of the tunnel.

Tunneling a car from France to England.

However, this limitation of tunnels is turned into an advantage with **VPNs** (**Virtual Private** **Networks**). A VPN is simply an overlay that is used to provide a measure of security.

**2. Describe the IP Version 4 Protocol**

Ans. An appropriate place to start our study of the network layer in the Internet is with the format of the IP data grams themselves. An IPv4 datagram consists of a header part and a body or payload part. The header has a 20-byte fixed part and a variable-length optional part. The header format is shown in Fig. 5-46. The bits are transmitted from left to right and top to bottom, with the high-order bit of the *Version* field going first. (This is a ‘‘big-endian’’ network byte order. On little endian machines, such as Intel x86 computers, a software conversion is required on both transmission and reception.) In retrospect, little endian would have been a better choice, but at the time IP was designed, no one knew it would come to dominate computing.



The IPv4 (Internet Protocol) header.

The *Version* field keeps track of which version of the protocol the datagram belongs to. Version 4 dominates the Internet today, and that is where we have started our discussion. By including the version at the start of each datagram, it becomes possible to have a transition between versions over a long period of time. In fact, IPv6, the next version of IP, was defined more than a decade ago, yet is only just beginning to be deployed. We will describe it later in this section. Its use will eventually be forced when each of China’s almost 231 people has a desktop PC, a laptop, and an IP phone. As an aside on numbering, IPv5 was an experimental real-time stream protocol that was never widely used.

Since the header length is not constant, a field in the header, *IHL*, is provided to tell how long the header is, in 32-bit words. The minimum value is 5, which applies when no options are present. The maximum value of this 4-bit field is 15, which limits the header to 60 bytes, and thus the *Options* field to 40 bytes. For some options, such as one that records the route a packet has taken,40 bytes is far too small, making those options useless. The *Differentiated services* field is one of the few fields that has changed its meaning (slightly) over the years. Originally, it was called the *Type of service* field. It was and still is intended to distinguish between different classes ofservice.

Various combinations of reliability and speed are possible. For digitized voice, fast delivery beats accurate delivery. For file transfer, error-free transmission is more important than fast transmission. The *Type of service* field provided 3 bits to signal priority and 3 bits to signal whether a host cared more about delay, throughput, or reliability. However, no one really knew what to do with these bits at routers, so they were left unused for many years. When differentiated services were designed, IETF threw in the towel and reused this field. Now, the top 6 bits are used to mark the packet with its service class; we described the expedited and assured

Services.The bottom 2 bits are used to carry explicit congestion notification information, such as whether the packet has experienced congestion; we described explicit congestion notification as part of congestion control earlier in this chapter. The *Total* *length* includes everything in the datagram—both header and data. The maximum length is65,535 bytes. At present, this upper limit is tolerable, but with future networks, larger datagrams may be needed. The *Identification* field is needed to allow the destination host to determine which packet a newly arrived fragment belongs to. All the fragments of a packet contain the same *Identification* value.

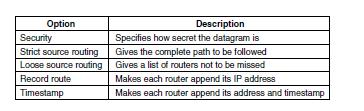
Next comes an unused bit, which is surprising, as available real estate in the IP header is extremely scarce. As an April fool’s joke, Bellovin (2003) proposed using this bit to detect malicious traffic. This would greatly simplify security, as packets with the ‘‘evil’’ bit set would be known to have been sent by attackers and could just be discarded. Unfortunately, network security is not this simple. Then come two 1-bit fields related to fragmentation. *DF* stands for Don’t Fragment. It is an order to the routers not to fragment the packet. Originally, it was intended to support hosts incapable of putting the pieces back together again. Now it is used as part of the process to discover the path MTU, which is the largest packet that can travel along a path without being fragmented. By marking the datagram with the *DF* bit, the sender knows it will either arrive in one piece, or an error message will be returned to the sender. *MF* stands for More Fragments. All fragments except the last one have this bit set. It is needed to know when all fragments of a datagram have arrived. The *Fragment offset* tells where in the current packet this fragment belongs.

All fragments except the last one in a datagram must be a multiple of 8 bytes, the elementary fragment unit. Since 13 bits are provided, there is a maximum of 8192 fragments per datagram, supporting a maximum packet length up to the limit of the *Total length* field. Working together, the *Identification*, *MF*, and *Fragment offset* fields are used to implement fragmentation as described in Sec. 5.5.5.The *TtL (Time to live)* field is a counter used to limit packet lifetimes. It was originally supposed to count time in seconds, allowing a maximum lifetime of 255 sec. It must be decremented on each hop and is supposed to be decremented multiple times when a packet is queued for a long time in a router. In practice, it just counts hops. When it hits zero, the packet is discarded and a warning packet is sent back to the source host. This feature prevents packets from wandering around forever, something that otherwise might happen if the routing tables ever become corrupted.

When the network layer has assembled a complete packet, it needs to know what to do with it. The *Protocol* field tells it which transport process to give the packet to. TCP is one possibility, but so are UDP and some others. The numbering of protocols is global across the entire Internet. Protocols and other assigned numbers were formerly listed in RFC 1700, but nowadays they are contained in an online database located at [*www.iana.org*.](http://www.iana.org/) Since the header carries vital information such as addresses, it rates its own checksum for protection, the *Header checksum*. The algorithm is to add up all the 16-bit half words of the header as they arrive, using one’s complement arithmetic, and then take the one’s complement of the result. For purposes of this algorithm, the *Header checksum* is assumed to be zero upon arrival. Such a checksum is useful for detecting errors while the packet travels through the network. Note that it must be recomputed at each hop because at least one field always changes (the *Time to live* field), but

tricks can be used to speed up the computation. The *Source address* and *Destination address* indicate the IP address of the source and destination network interfaces. The *Options* field was designed to provide an escape to allow subsequent versions of the protocol to include information not present in the original design, to permit experimenters to try out new ideas, and to avoid allocating header bits to information that is rarely needed. The options are of variable length. Each begins with a 1-byte code identifying the option. Some options are followed by a 1-byte option length field, and then one or more data bytes. The *Options* field is padded out to a multiple of 4 bytes. Originally, the five options listed in Fig. were defined.

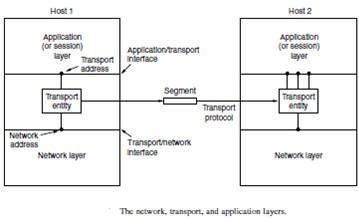
The *Security* option tells how secret the information is. In theory, a military router might use this field to specify not to route packets through certain countries the military considers to be ‘‘bad guys.’’ In practice, all routers ignore it, so its only practical function is to help spies find the good stuff more easily. The *Strict source routing* option gives the complete path from source to destination as a sequence of IP addresses. The datagram is required to follow that



Some of the IP options exact route. It is most useful for system managers who need to send emergency packets when the routing tables have been corrupted, or for making timing measurements. The *Loose source* *routing* option requires the packet to traverse the list of routers specified, in the order specified,but it is allowed to pass through other routers on the way. Normally, this option will provide only a few routers, to force a particular path. For example, to force a packet from London to Sydney to go west instead of east, this option might specify routers in New York, Los Angeles, and Honolulu. This option is most useful when political or economic considerations dictate passing through or avoiding certain countries.

The *Record route* option tells each router along the path to append its IP address to the *Options* field. This allows system managers to track down bugs in the routing algorithms (‘‘Why are packets from Houston to Dallas visiting Tokyo first?’’). When the ARPANET was first set up, no packet ever passed through more than nine routers, so 40 bytes of options was plenty. As mentioned above, now it is too small. Finally, the *Timestamp* option is like the *Record route* option, except that in addition to recording its 32-bit IP address, each router also records a 32-bit timestamp. This option, too, is mostly useful for network measurement. Today, IP options have fallen out of favor. Many routers ignore them or do not process them efficiently, shunting them to the side as an uncommon case. That is, they are only partly supported and they are rarely used.

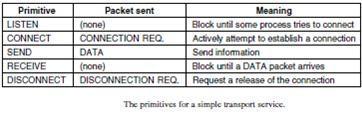
**3. What are the Services provided to the Upper Layers**

The ultimate goal of the transport layer is to provide efficient, reliable, and cost-effective data transmission service to its users, normally processes in the application layer. To achieve this, the transport layer makes use of the services provided by the network layer. The software and/or hardware within the transport layer that does the work is called the **transport entity**. The transport entity can be located in the operating system kernel, in a library package bound into network applications, in a separate user process, or even on the network interface card. The first two options are most common on the Internet. The (logical) relationship of the network, transport, and application layers is illustrated in Fig.

there are also two types of transport service. The connection-oriented transport service is similar to the connection-oriented network service in many ways. In both cases, connections have three phases: establishment, data transfer, and release. Addressing and flow control are also similar in both layers. Furthermore, the connectionless transport service is also very similar to the connectionless network service. However, note that it can be difficult to provide a connectionless transport service on top of a connection-oriented network service, since it is inefficient to set up a connection to send a single packet and then tear it down immediately afterwards. The obvious question is this: if the transport layer service is so similar to the network layer service, why are there two distinct layers? Why is one layer not Problems occur, that’s what? The users have no real control over the network layer, so they cannot solve the problem of poor service by using better routers or putting more error handling in the data link layer because they don’t own the routers. The only possibility is to put on top of the network layer another layer that improves the quality of the service. If, in a connectionless network, packets are lost or mangled, the transport entity can detect the problem and compensate for it by using retransmissions. If, in a connection-oriented network, a transport entity is informed halfway through a long transmission that its network connection has been abruptly terminated, with no indication of what has happened to the data currently in transit, it can set up a new network connection to the remote transport entity. Using this new network connection, it can send a query to its peer asking which data arrived and which did not, and knowing where it was, pick up from where it left off.

**Transport Service Primitives**

To allow users to access the transport service, the transport layer must provide some operations to application programs, that is, a transport service interface. Each transport service has its own interface. In this section, we will first examine a simple (hypothetical) transport service and its interface to see the bare essentials. In the following section, we will look at a real example. The transport service is similar to the network service, but there are also some important differences. The main difference is that the network service is intended to model the service offered by real networks, warts and all. Real networks can lose packets, so the network service is generally unreliable.



**4.Explain the terms**

**a.Connection Establishment**

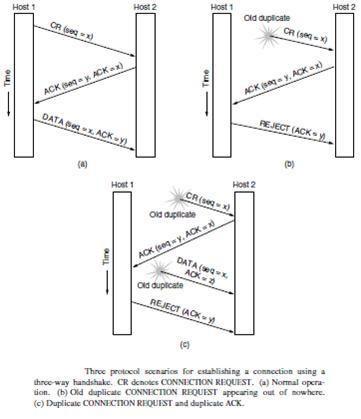
Establishing a connection sounds easy, but it is actually surprisingly tricky. At first glance, it would seem sufficient for one transport entity to just send a CONNECTION REQUEST segment to the destination and wait for a CONNECTION ACCEPTED reply. The problem occurs when the network can lose, delay, corrupt, and duplicate packets. This behaviour causes serious complications. Imagine a network that is so congested that acknowledgements hardly ever get back in time and each packet times out and is retransmitted two or three times. Suppose that the network uses datagrams inside and that every packet follows a different route. Some of the packets might get stuck in a traffic jam inside the network and take a long time to arrive. That is, they may be delayed in the network and pop out much later, when the sender thought that they had been lost. The worst possible nightmare is as follows. A user establishes a connection with a bank, sends messages telling the bank to transfer a large amount of money to the account of a not-entirely-trustworthy person. Unfortunately, the packets decide to take the scenic route to the destination and go off exploring a remote corner of the network. The sender then times out and sends them all again. This time the packets take the shortest route and are delivered quickly so the sender releases the connection.

Packet lifetime can be restricted to a known maximum using one (or more) of the following techniques:

1. Restricted network design.
2. Putting a hop counter in each packet.
3. Time stamping each packet.

The first technique includes any method that prevents packets from looping, combined with some way of bounding delay including congestion over the (now known) longest possible path. It is difficult, given that internets may range from a single city to international in scope. The second method consists of having the hop count initialized to some appropriate value and decremented each time the packet is forwarded. The network protocol simply discards any packet whose hop counter becomes zero. The third method requires each packet to bear the time it was created, with the routers agreeing to discard any packet older than some agreed-upon time. This latter method requires the router clocks to be synchronized, which itself is a nontrivial task, and in practice a hop counter is a close enough approximation to age.

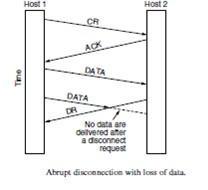
TCP uses this three-way handshake to establish connections. Within a connection, a timestamp is used to extend the 32-bit sequence number so that it will not wrap within the maximum packet lifetime, even for gigabit-per-second connections. This mechanism is a fix to TCP that was needed as it was used on faster and faster links. It is described in RFC 1323 and called **PAWS**

(**Protection Against Wrapped Sequence numbers**). Across connections, for the initial sequence numbers and before PAWS can come into play, TCP originally use the clock-based scheme just described. However, this turned out to have security vulnerability. The clock made it easy for an attacker to predict the next initial sequence number and send packets that tricked the three-way handshake and established a forged connection. To close this hole, pseudorandom initial sequence numbers are used for connections in practice.

the initial sequence numbers not repeat for an interval even though they appear random to an observer. Otherwise, delayed duplicates can wreak havoc.

**Connection Release**

Releasing a connection is easier than establishing one. Nevertheless, there are more pitfalls than one might expect here. As we mentioned earlier, there are two styles of terminating a connection: asymmetric release and symmetric release Asymmetric release is the way the telephone system works: when one party hangs up, the connection is broken. Symmetric release treats the connection as two separate unidirectional connections and requires each one to be released separately. Asymmetric release is abrupt and may result in data loss. Consider the scenario of Fig. After the connection is established, host 1 sends a segment that arrives properly at host 2. Then host 1 sends another segment. Unfortunately, host 2 issues a DISCONNECT before the second segment arrives. The result is that the connection is released and data are lost.

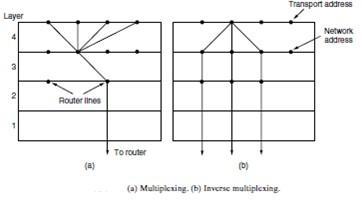


**6. What is crash recovery?**

Ans.

**Crash Recovery**

If hosts and routers are subject to crashes or connections are long-lived (e.g., large software or media downloads) recovery from these crashes becomes an issue. If the transport entity is entirely within the hosts, recovery from network



and router crashes is straightforward. The transport entities expect lost segments all the time and know how to cope with them by using retransmissions. A more troublesome problem is how to recover from host crashes. In particular, it may be desirable for clients to be able to continue working when servers crash and quickly reboot. To illustrate the difficulty, let us assume that one host, the client, is sending a long file to another host, the file server, using a simple Stop-and-wait protocol. The transport layer on the server just passes the incoming segments to the transport user, one by one. Partway through the transmission, the server crashes. When it comes

back up, its tables are reinitialized, so it no longer knows precisely where it was. In an attempt to recover its previous status, the server might send a broadcast segment to all other hosts, announcing that it has just crashed and requesting that its clients inform it of the status of all open connections. Each client can be in one of two states: one segment outstanding, *S1*, or no segments outstanding, *S0*. Based on only this state information, the client must decide whether to retransmit the most recent segment.

**Unit-V**

**Short Answer Questions**

**1. Define** UDP?

**Ans.**

UDP provides connectionless, unreliable, datagram service. Connectionless service means that there is no logical connection between the two ends exchanging messages. Each message is an independent entity encapsulated in a datagram.UDP does not see any relation (connection) between consequent datagram coming from the same source and going to the same destination.

**2.What is TCP?**

**Ans. Transmission Control Protocol** (TCP) is a connection-oriented, reliable protocol. TCPexplicitly defines connection establishment, data transfer, and connection teardown phases to provide a connection-oriented service.

1. **What is HTTP?**

Ans.The Hyper Text Transfer Protocol (HTTP) is used to define how the client-server programs can be written to retrieve web pages from the Web.An HTTP client sends a request; an HTTP server returns a response. The server uses the port number 80; the client uses a temporary port number.

**5.** **What is FTP.**

File Transfer Protocol (FTP) is the standard protocol provided by *TCP/IP* for copying a file from one host to another.Two systems may have different directory structures. All of these problems have been solved by FTP in a very simple and elegant approach.

1. **What is EMAIL**

Ans.Electronic mail (or e-mail) allows users to exchange messages.

In an application such as HTTP or FTP, the server program is running all the time, waiting for arequest from a client.

When the request arrives, the server provides the service. There is a request and there is a response.

In the case of electronic mail, the situation is different. First, e-mail is considered a one-way transaction

The users run only client programs when they want and the intermediate servers apply the client/server paradigm.

**6.What is TELNET.**

One of the original remote logging protocols is **TELNET,** which is an abbreviation for *Terminal* *Network.*

A hacker can eavesdrop and obtain the logging name and password. Because of this security issue, the use of TELNET has diminished in favor of another protocol, Secure Shell (SSH).

**7. What is SSH**

Ans.Secure Shell (SSH) is a secure application program that can be used today for several purposes such as remote logging and file transfer; it was originally designed to replace TELNET.

There are two versions of SSH: SSH-l and SSH-2

8. **What is DOMAIN NAME SYSTEM (DNS)**

The host that needs mapping can contact the closest computer holding the needed information. This method is used by the Domain Name System (DNS).



A user wants to use a file transfer client to access the corresponding file transfer server running on a remote host.

The user knows only the file transfer server name, such as *afilesource.com.*

**9. Expalin WWW?**

**Ans.**

The term **World Wide Web (WWW)**refers to the collection of public Web sites connected to the Internet worldwide, together with the [client](http://compnetworking.about.com/od/basicnetworkingfaqs/a/client-server.htm) devices such as computers and cell phones that access its content. For many years it has become known simply as "the Web."

**10.Define SMTP?**

SMTP (Simple Mail Transfer Protocol) is a [TCP/IP](http://searchnetworking.techtarget.com/definition/TCP-IP) [protocol](http://searchnetworking.techtarget.com/definition/protocol) used in sending and receiving e-mail. However, since it is limited in its ability to [queue](http://searchcio-midmarket.techtarget.com/definition/queue) messages at the receiving end, it is usually used with one of two other protocols, [POP3](http://searchexchange.techtarget.com/definition/POP3) or [IMAP](http://searchexchange.techtarget.com/definition/IMAP), that let the user save messages in a server mailbox and download them periodically from the server. In other words, users typically use a program that uses SMTP for sending e-mail and either POP3 or IMAP for receiving e-mail. On [Unix](http://searchenterpriselinux.techtarget.com/definition/Unix)-based systems, [sendmail](http://searchenterpriselinux.techtarget.com/definition/sendmail) is the most widely-used SMTP server for e-mail. A commercial package, Sendmail, includes a POP3 server. Microsoft [Exchange](http://searchexchange.techtarget.com/definition/Exchange) includes an SMTP server and can also be set up to include POP3 support.

Long Answer Questions

1. **Explain the real transport protocol of UDP and how will you calculate checksum in UDP?**

**Ans. *UDP Protocol***

UDP provides connectionless, unreliable, datagram service. Connectionless service means that there is no logical connection between the two ends exchanging messages. Each message is an independent entity encapsulated in a datagram.

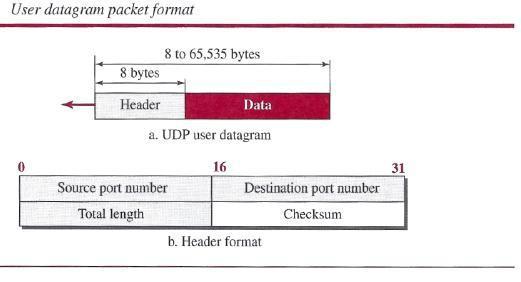
UDP does not see any relation (connection) between consequent datagram coming from the same source and going to the same destination.

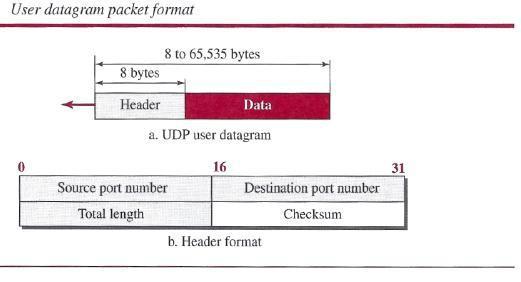
UDP has an advantage: it is message-oriented. It gives boundaries to the messages exchanged. An application program may be designed to use UDP if it is sending small messages and the simplicity and speed is more important for the application than reliability.

**User Datagram**

UDP packets, called *user datagram,* have a fixed-size header of 8 bytes made of four fields, each of 2 bytes (16 bits).

. The 16 bits can define a total length of 0 to 65,535 bytes. However, the total length needs to be less because a UDP user datagram is stored in an IP datagram with the total length of 65,535 bytes. The last field can carry the optional checksum



****

**UDP Services**

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

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

***Connectionless Services***

As mentioned previously, UDP provides a *connection less service.* This means that each user datagram sent by UDP is an independent datagram. There is no relationship between the different user data grams even if they are coming from the same source process and going to the same destination program.

***Flow Control***

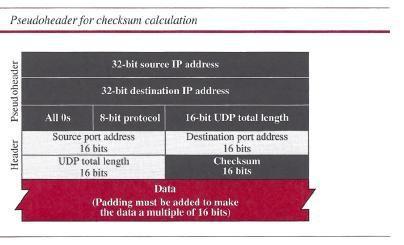
UDP is a very simple protocol. There is no *flow control,* and hence no window mechanism. The receiver may overflow with incoming messages.

***Error Control***

There is no *error control* mechanism in UDP except for the checksum. This means that the sender does not know if a message has been lost or duplicated.

***Checksum***

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



# 2. Expalin big-endian and little-endian

Big-endian and little-endian are terms that describe the order in which a sequence of [byte](http://searchstorage.techtarget.com/definition/byte)s are stored in computer memory. Big-endian is an order in which the "big end" (most significant value in the sequence) is stored first (at the lowest storage address). Little-endian is an order in which the "little end" (least significant value in the sequence) is stored first. For example, in a big-endian computer, the two bytes required for the [hexadecimal](http://searchcio-midmarket.techtarget.com/definition/hexadecimal) number 4F52 would be stored as 4F52 in storage (if 4F is stored at storage address 1000, for example, 52 will be at address 1001). In a little-endian system, it would be stored as 524F (52 at address 1000, 4F at 1001).

IBM's 370 mainframes, most [RISC](http://search400.techtarget.com/definition/RISC)-based computers, and Motorola microprocessors use the big-endian approach. [TCP/IP](http://searchnetworking.techtarget.com/definition/TCP-IP) also uses the big-endian approach (and thus big-endian is sometimes called *network order*). For people who use languages that read left-to-right, this seems like the natural way to think of a storing a string of characters or numbers - in the same order you expect to see it presented to you. Many of us would thus think of big-endian as storing something in *forward* fashion, just as we read.

On the other hand, Intel [processor](http://searchcio-midmarket.techtarget.com/definition/processor)s (CPUs) and DEC Alphas and at least some programs that run on them are little-endian. An argument for little-endian order is that as you increase a numeric value, you may need to add digits to the left (a higher non-exponential number has more digits). Thus, an addition of two numbers often requires moving all the digits of a big-endian ordered number in storage, moving everything to the right. In a number stored in little-endian fashion, the least significant bytes can stay where they are and new digits can be added to the right at a higher address. This means that some computer operations may be simpler and faster to perform.

Language [compiler](http://whatis.techtarget.com/definition/compiler)s such as that of [Java](http://searchsoa.techtarget.com/definition/Java) or [FORTRAN](http://whatis.techtarget.com/definition/FORTRAN-FORmula-TRANslation) have to know which way the object code they develop is going to be stored. Converters can be used to change one kind of endian to the other when necessary.

3. **Explain TRANSMISSION CONTROL PROTOCOL**

**Transmission Control Protocol** (TCP) is a connection-oriented, reliable protocol. TCPexplicitly defines connection establishment, data transfer, and connection teardown phases to provide a connection-oriented service.

**TCP Services**

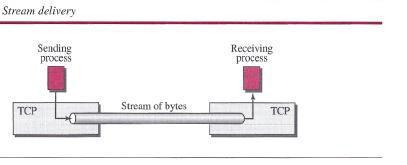
***Process-to-Process Communication***

As with UDP, TCP provides process-to-process communication using port numbers. We have already given some of the port numbers used by TCP.

***Stream Delivery Service***

In UDP, a process sends messages with predefined boundaries to UDP for delivery. UDP adds its own header to each of these messages and delivers it to IP for transmission.

TCP, on the other hand, allows the sending process to deliver data as a stream of bytes and allows the receiving process to obtain data as a stream of bytes.

TCP creates an environment in which the two processes seem to be connected by an imaginary "tube" that carries their bytes across the Internet.

***Sending and Receiving Buffers***

Because the sending and the receiving processes may not necessarily write or read data at the same rate, TCP needs buffers for storage.

There are two buffers, the sending buffer and the receiving buffer, one for each direction.

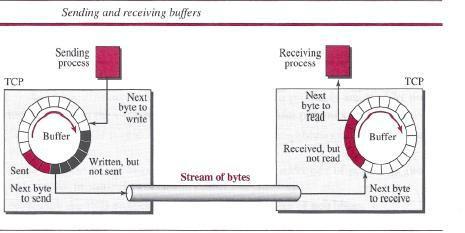
At the sender, the buffer has three types of chambers. The white section contains empty chambers that can be filled by the sending process (producer).

The colored area holds bytes that have been sent but not yet acknowledged.

The TCP sender keeps these bytes in the buffer until it receives an acknowledgment. The shaded area contains bytes to be sent by the sending TCP.

The operation of the buffer at the receiver is simpler. The circular buffer is divided into two areas (shown as white and colored).

The white area contains empty chambers to be filled by bytes received from the network.

The colored sections contain received bytes that can be read by the receiving process. When a byte is read by the receiving process, the chamber is recycled and added to the pool of empty chambers.

***Segments***

Although buffering handles the disparity between the speed of the producing and consuming Processes, we need one more step before we can send data.

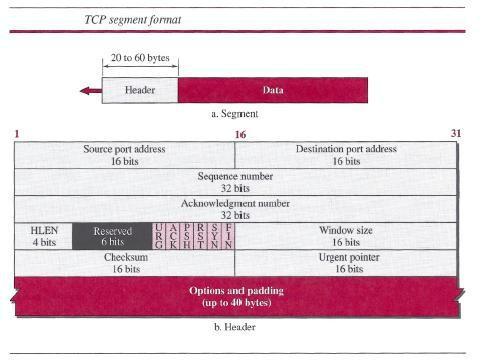
The network layer, as a service provider for TCP, needs to send data in packets, not as a stream of bytes. At the transport layer, TCP groups a number of bytes together into a packet called a *segment.*

The segments are encapsulated in an IP datagram and transmitted. This entire operation is transparent to the receiving process.

***Format***

The segment consists of a header of 20 to 60 bytes, followed by data from the application program.

The header is 20 bytes if there are no options and up to 60 bytes if it contains options.



***Source port address*** This is a 16-bit field that defines the port number of the application programin the host that is sending the segment.

***Destination port address*** This is a 16-bit field that defines the port number of the applicationprogram in the host that is receiving the segment.

***Sequence number*** This 32-bit field defines the number assigned to the first byte of datacontained in this segment.

***Acknowledgment number*** This 32-bit field defines the byte number that the receiver of thesegment is expecting to receive from the other party.

***Header length*** This 4-bit field indicates the number of 4-byte words in the TCP header. Thelength of the header can be between 20 and 60 bytes.

1. **Explain three domains of the Domain Name Space?**

**Ans.DOMAIN NAME SYSTEM (DNS)**

The host that needs mapping can contact the closest computer holding the needed information. This method is used by the Domain Name System (DNS).



A user wants to use a file transfer client to access the corresponding file transfer server running on a remote host.

The user knows only the file transfer server name, such as *afilesource.com.*

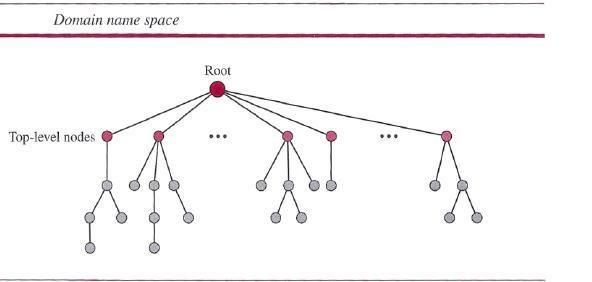
**Name Space**

A **name** space that maps each address to a unique name can be organized in two ways: flat or hierarchical.

In a *flat name space,* a name is assigned to an address. A name in this space is a sequence of characters without structure.

In a *hierarchical name space,* each name is made of several parts.

***Domain Name Space***



***Domain Name Space***

To have a hierarchical name space, a domain name space was designed. In this design the names are defined in an inverted-tree structure with the root at the top.

***Label***

Each node in the tree has a label, which is a string with a maximum of 63 characters. The root label is a null string (empty string).

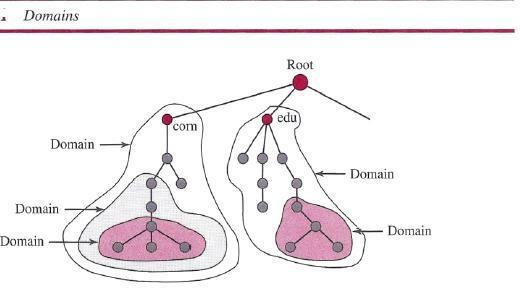
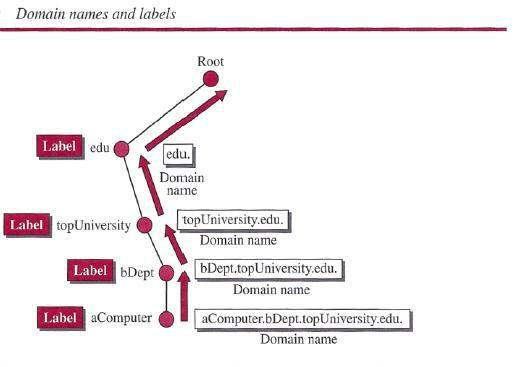
***Domain Name***

If a label is terminated by a null string, it is called a fully qualified domain name (FQDN).

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

***Domain***

A domain is a sub tree of the domain name space. The name of the domain is the name of the node at the top of the sub tree.





,