

Subject: Computer Networks

Subject Code: 17MCA31

Year: Dec.2018/Jan.2019

Q1(a)

Define Computer Networks. Explain the uses of Computer Networks in business application and home applications.

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

Business Applications:-

For smaller companies, all the computers are likely to be in a single office or perhaps a single building, but for larger ones, the computers and employees may be scattered over dozens of offices and plants in many countries. Nevertheless, a sales person in New York might sometimes need access to a product inventory.

database in Singapore. Networks called VPNs (Virtual Private Networks) may be used to join the individual networks at different sites into one extended network. In other words, the mere fact that a user happens to be 15,000 km away from his data should not prevent him from using the data as though they were local. This goal may be summarized by saying that it is an attempt to end the “tyranny of geography.”

In the simplest of terms, one can imagine a company’s information system as consisting of one or more databases with company information and some number of employees who need to access them remotely. In this model, the data are stored on powerful computers called servers. Often these are centrally housed and maintained by a system administrator. In contrast, the employees have simpler machines, called clients, on their desks, with which they access remote data, for example, to include in spreadsheets they are constructing. (Sometimes we will refer to the human user of the client machine as the “client,” but it should be clear from the context whether we mean the computer or its user.)

A second goal of setting up a computer network has to do with people rather than information or even computers. A computer network can provide a powerful communication medium among employees. Virtually every company that has two or more computers now has email (electronic mail), which employees generally use for a great deal of daily communication. In fact, a common gripe around the water cooler is how much email everyone has to deal with, much of it quite meaningless because bosses have discovered that they can send the same (often content-free) message to all their subordinates at the push of a button. Telephone calls between employees may be carried by the computer network instead of by the phone company. This technology is called IP telephony or Voice over IP (VoIP) when Internet technology is used. The microphone and speaker at each end may belong to a VoIP-enabled phone or the employee’s computer. Companies find this a wonderful way to save on their telephone bills.

A third goal for many companies is doing business electronically, especially with customers and suppliers. This new model is called e-commerce (electronic commerce) and it has grown rapidly in recent years. Airlines, bookstores, and other retailers have discovered that many customers like the convenience of shopping from home. Consequently, many companies provide catalogs of their goods and services online and take orders online. Manufacturers of automobiles, aircraft, and computers, among others, buy subsystems from a variety of suppliers and then assemble the parts. Using computer networks, manufacturers can place orders electronically as needed. This reduces the need for large inventories and enhances efficiency.

Home Applications:-

Internet access provides home users with connectivity to remote computers. As with companies, home users can access information, communicate with other people, and buy products and services with e-commerce. The main benefit now comes from connecting

	<p>outside of the home. Bob Metcalfe, the inventor of Ethernet, hypothesized that the value of a network is proportional to the square of the number of users because this is roughly the number of different connections that may be made (Gilder, 1993). This hypothesis is known as “Metcalfe’s law.” It helps to explain how the tremendous popularity of the Internet comes from its size.</p> <p>Peer-to-peer communication is often used to share music and videos. It really hit the big time around 2000 with a music sharing service called Napster that was shut down after what was probably the biggest copyright infringement case in all of recorded history (Lam and Tan, 2001; and Macedonia, 2000). Legal applications for peer-to-peer communication also exist. These include fans sharing public domain music, families sharing photos and movies, and users downloading public software packages. In fact, one of the most popular Internet applications of all, email, is inherently peer-to-peer. This form of communication is likely to grow considerably in the future.</p> <p>The Internet can be used by applications to carry audio (e.g., Internet radio stations) and video (e.g., YouTube). Besides being a cheap way to call to distant friends, these applications can provide rich experiences such as telelearning, meaning attending 8 A.M. classes without the inconvenience of having to get out of bed first. In the long run, the use of networks to enhance human-to-human communication may prove more important than any of the others. It may become hugely important to people who are geographically challenged, giving them the same access to services as people living in the middle of a big city.</p> <p>Between person-to-person communications and accessing information are social network applications. Here, the flow of information is driven by the relationships that people declare between each other. One of the most popular social networking sites is Facebook. It lets people update their personal profiles and shares the updates with other people who they have declared to be their friends. Other social networking applications can make introductions via friends of friends, send news messages to friends such as Twitter above, and much more</p>
Q1(b)	<p>Explain Local Area Network in detail.</p> <p>The next step up is the LAN (Local Area Network). A LAN is a privately owned network that operates within and nearby a single building like a home, office or factory. LANs are widely used to connect personal computers and consumer electronics to let them share resources (e.g., printers) and exchange information. When LANs are used by companies, they are called enterprise networks. Wireless LANs are very popular these days, especially in homes, older office buildings, cafeterias, and other places where it is too much trouble to install cables. In these systems, every computer has a radio modem and an antenna that it uses to communicate with other computers. In most cases, each computer talks to a device in the ceiling as shown in Fig. 1-8(a). This device, called an AP (Access Point), wireless router, or base station, relays packets between the wireless computers and also between them and the Internet. Being the AP is like being the popular kid at school because everyone wants to talk to you. However, if other computers are close enough, they can communicate directly with one another in a peer-to-peer configuration.</p> <p>There is a standard for wireless LANs called IEEE 802.11, popularly known as WiFi, which has become very widespread. It runs at speeds anywhere from 11 to hundreds of Mbps. (In this book we will adhere to tradition and measure line speeds in megabits/sec, where 1 Mbps is 1,000,000 bits/sec, and gigabits/sec, where 1 Gbps is 1,000,000,000 bits/sec.)</p> <p>It is also possible to divide one large physical LAN into two smaller logical LANs. You might wonder why this would be useful. Sometimes, the layout of the network equipment does not match the organization’s structure. For example, the engineering and finance departments of a company might have computers on the same physical LAN because they are in the same wing of the building but it might be easier to manage the system if engineering and finance logically each had its own network Virtual LAN or VLAN. In this</p>

	<p>design each port is tagged with a “color,” say green for engineering and red for finance. The switch then forwards packets so that computers attached to the green ports are separated from the computers attached to the red ports. Broadcast packets sent on a red port, for example, will not be received on a green port, just as though there were two different LANs.</p> <p>Both wireless and wired broadcast networks can be divided into static and dynamic designs, depending on how the channel is allocated. A typical static allocation would be to divide time into discrete intervals and use a round-robin algorithm, allowing each machine to broadcast only when its time slot comes up. Static allocation wastes channel capacity when a machine has nothing to say during its allocated slot, so most systems attempt to allocate the channel dynamically (i.e., on demand).</p> <p>While we could think of the home network as just another LAN, it is more likely to have different properties than other networks. First, the networked devices have to be very easy to install. Wireless routers are the most returned consumer electronic item. People buy one because they want a wireless network at home, find that it does not work “out of the box,” and then return it rather than listen to elevator music while on hold on the technical helpline.</p>
Q2(a)	<p>Describe the architecture of internet with diagram.</p> <p>The architecture of the Internet has also changed a great deal as it has grown explosively. In this section, we will attempt to give a brief overview of what it looks like today. The picture is complicated by continuous upheavals in the businesses of telephone companies (telcos), cable companies and ISPs that often make it hard to tell who is doing what. One driver of these upheavals is telecommunications convergence, in which one network is used for previously different uses. For example, in a “triple play” one company sells you telephony, TV, and Internet service over the same network connection on the assumption that this will save you money. Consequently, the description given here will be of necessity somewhat simpler than reality. And what is true today may not be true tomorrow.</p> <p>To join the Internet, the computer is connected to an Internet Service Provider, or simply ISP, from who the user purchases Internet access or connectivity. This lets the computer exchange packets with all of the other accessible hosts on the Internet. The user might send packets to surf the Web or for any of a thousand other uses, it does not matter. There are many kinds of Internet access, and they are usually distinguished by how much bandwidth they provide and how much they cost, but the most important attribute is connectivity.</p> <p>A common way to connect to an ISP is to use the phone line to your house, in which case your phone company is your ISP. DSL, short for Digital Subscriber Line, reuses the telephone line that connects to your house for digital data transmission. The computer is connected to a device called a DSL modem that converts between digital packets and analog signals that can pass unhindered over the telephone line. At the other end, a device called a DSLAM (Digital Subscriber Line Access Multiplexer) converts between signals and packets.</p> <p>Another method is to send signals over the cable TV system. Like DSL, this is a way to reuse existing infrastructure, in this case otherwise unused cable TV channels. The device at the home end is called a cable modem and the device at the cable headend is called the CMTS (Cable Modem Termination System). DSL and cable provide Internet access at rates from a small fraction of a megabit/sec to multiple megabit/sec, depending on the system. These rates are much greater than dial-up rates, which are limited to 56 kbps because of the narrow bandwidth used for voice calls. Internet access at much greater than dial-up speeds is called broadband. The name refers to the broader bandwidth that is used for faster networks, rather than any particular speed.</p> <p>The access methods mentioned so far are limited by the bandwidth of the “last mile” or last leg of transmission. By running optical fiber to residences, faster Internet access can be provided at rates on the order of 10 to 100 Mbps. This design is called FTTH (Fiber to the Home). For businesses in commercial areas, it may make sense to lease a high-speed</p>

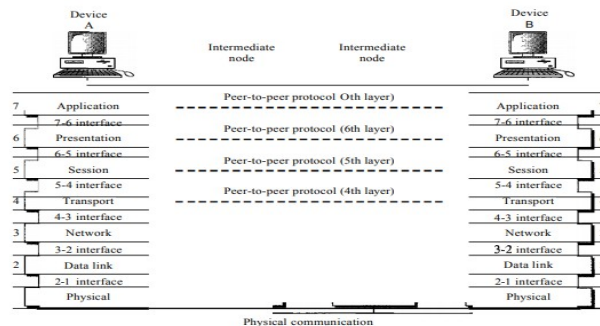
transmission line from the offices to the nearest ISP. For example, in North America, a T3 line runs at roughly 45 Mbps.

Wireless is used for Internet access too. An example we will explore shortly is that of 3G mobile phone networks. They can provide data delivery at rates of 1 Mbps or higher to mobile phones and fixed subscribers in the coverage area. We can now move packets between the home and the ISP. We call the location at which customer packets enter the ISP network for service the ISP's POP (Point of Presence). We will next explain how packets are moved between the POPs of different ISPs. From this point on, the system is fully digital and packet switched. ISP networks may be regional, national, or international in scope. We have already seen that their architecture is made up of long-distance transmission lines that interconnect routers at POPs in the different cities that the ISPs serve. This equipment is called the backbone of the ISP. If a packet is destined for a host served directly by the ISP, that packet is routed over the backbone and delivered to the host. Otherwise, it must be handed over to another ISP.

Q2(b) Explain OSI Reference model with diagram.

the International Standards Organization (ISO) is a multinational body dedicated to worldwide agreement on international standards. An ISO standard that covers all aspects of network communications is the Open Systems Interconnection model. It was first introduced in the late 1970s.

The OSI model is a layered framework for the design of network systems that allows communication between all types of computer systems. It consists of seven separate but related layers, each of which defines a part of the process of moving information across a network



LAYERS IN THE OSI MODEL

1) Physical Layer

The physical layer coordinates the functions required to carry a bit stream over a physical medium. It deals with the mechanical and electrical specifications of the interface and transmission medium.

It also defines the procedures and functions that physical devices and interfaces have to perform for transmission to occur.

Representation of bits. The physical layer data consists of a stream of bits (sequence of 0s or 1s) with no interpretation. To be transmitted, bits must be encoded into signals--electrical or optical. The physical layer defines the type of encoding (how 0s and 1s are changed to signals).

Data rate. The transmission rate--the number of bits sent each second--is also defined by the physical layer. In other words, the physical layer defines the duration of a bit, which is how long it lasts.

Synchronization of bits. The sender and receiver not only must use the same bit rate but also must be synchronized at the bit level. In other words, the sender and the receiver clocks must be synchronized.

Physical topology. The physical topology defines how devices are connected to make a network. Devices can be connected by using a mesh topology (every device is connected to

every other device), a star topology (devices are connected through a central device), a ring topology (each device is connected to the next, forming a ring), a bus topology (every device is on a common link), or a hybrid topology (this is a combination of two or more topologies).

Transmission mode. The physical layer also defines the direction of transmission between two devices: simplex, half-duplex, or full-duplex. In simplex mode, only one device can send; the other can only receive. The simplex mode is a one-way communication. In the half-duplex mode, two devices can send and receive, but not at the same time. In a full-duplex (or simply duplex) mode, two devices can send and receive at the same time.

2) Data Link Layer

The data link layer transforms the physical layer, a raw transmission facility, to a reliable link. It makes the physical layer appear error-free to the upper layer (network layer).

Other responsibilities of the data link layer include the following:

Framing. The data link layer divides the stream of bits received from the network layer into manageable data units called frames. The data link layer is responsible for moving frames from one hop (node) to the next.

Physical addressing. If frames are to be distributed to different systems on the network, the data link layer adds a header to the frame to define the sender and/or receiver of the frame. If the frame is intended for a system outside the sender's network, the receiver address is the address of the device that connects the network to the next one.

Flow control. If the rate at which the data are absorbed by the receiver is less than the rate at which data are produced in the sender, the data link layer imposes a flow control mechanism to avoid overwhelming the receiver.

Error control. The data link layer adds reliability to the physical layer by adding mechanisms to detect and retransmit damaged or lost frames. It also uses a mechanism to recognize duplicate frames. Error control is normally achieved through a trailer added to the end of the frame.

Access control. When two or more devices are connected to the same link, data link layer protocols are necessary to determine which device has control over the link at any given time

3) Network Layer

The network layer is responsible for the source-to-destination delivery of a packet, possibly across multiple networks (links). Whereas the data link layer oversees the delivery of the packet between two systems on the same network (links), the network layer ensures that each packet gets from its point of origin to its final destination.

If two systems are connected to the same link, there is usually no need for a network layer. However, if the two systems are attached to different networks (links) with connecting devices between the networks (links), there is often a need for the network layer to accomplish source-to-destination delivery. Figure shows the relationship of the network layer to the data link and transport layers.

Logical addressing. The physical addressing implemented by the data link layer handles the addressing problem locally. If a packet passes the network boundary, we need another addressing system to help distinguish the source and destination systems. The network layer adds a header to the packet coming from the upper layer that, among other things, includes the logical addresses of the sender and receiver.

Routing. When independent networks or links are connected to create internetworks (network of networks) or a large network, the connecting devices (called routers or switches) route or switch the packets to their final destination. One of the functions of the network layer is to provide this mechanism.

4) Transport Layer

The transport layer is responsible for process-to-process delivery of the entire message. A process is an application program running on a host. Whereas the network layer oversees source-to-destination delivery of individual packets, it does not recognize any relationship

between those packets. It treats each one independently, as though each piece belonged to a separate message, whether or not it does. The transport layer, on the other hand, ensures that the whole message arrives intact and in order, overseeing both error control and flow control at the source-to-destination level.

Service-point addressing.

Computers often run several programs at the same time. For this reason, source-to-destination delivery means delivery not only from one computer to the next but also from a specific process (running program) on one computer to a specific process (running program) on the other. The transport layer header must therefore include a type of address called a service-point address (or port address).

The network layer gets each packet to the correct computer; the transport layer gets the entire message to the correct process on that computer.

Segmentation and reassembly.

A message is divided into transmittable segments, with each segment containing a sequence number. These numbers enable the transport layer to reassemble the message correctly upon arriving at the destination and to identify and replace packets that were lost in transmission.

Connection control.

The transport layer can be either connectionless or connection oriented. A connectionless transport layer treats each segment as an independent packet and delivers it to the transport layer at the destination machine. A connection oriented transport layer makes a connection with the transport layer at the destination machine first before delivering the packets. After all the data are transferred, the connection is terminated.

flow control.

at this layer is performed end to end rather than across a single link.

error control

at this layer is performed process-to process rather than across a single link. The sending transport layer makes sure that the entire message arrives at the receiving transport layer without error (damage, loss, or duplication). Error correction is usually achieved through retransmission.

5)Session Layer

The services provided by the first three layers (physical, data link, and network) are not sufficient for some processes. The session layer is the network dialog controller. It establishes, maintains, and synchronizes the interaction among communicating systems. The session layer is responsible for dialog control and synchronization.

Specific responsibilities of the session layer include the following:

Dialog control. The session layer allows two systems to enter into a dialog. It allows the communication between two processes to take place in either half duplex (one way at a time) or full-duplex (two ways at a time) mode.

Synchronization. The session layer allows a process to add checkpoints, or synchronization points, to a stream of data. For example, if a system is sending a file of 2000 pages, it is advisable to insert checkpoints after every 100 pages to ensure that each 100-page unit is received and acknowledged independently. In this case, if a crash happens during the transmission of page 523, the only pages that need to be resent after system recovery are pages 501 to 523. Pages previous to 501 need not be resent.

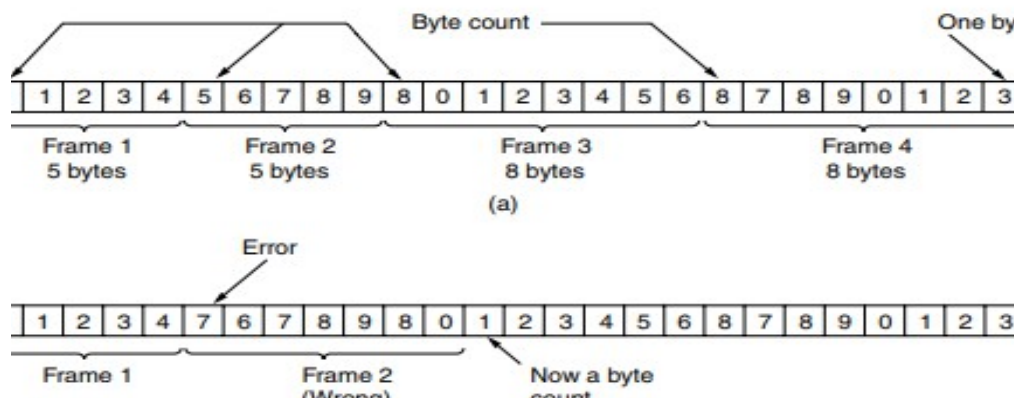
6) Presentation Layer

The presentation layer is concerned with the syntax and semantics of the information exchanged between two systems.

Translation. The processes (running programs) in two systems are usually exchanging information in the form of character strings, numbers, and so on. The information must be changed to bit streams before being transmitted. Because different computers use different encoding systems, the presentation layer is responsible for interoperability between these different encoding methods.

Encryption. To carry sensitive information, a system must be able to ensure privacy.

	<p>Encryption means that the sender transforms the original information to another form and sends the resulting message out over the network. Decryption reverses the original process to transform the message back to its original form.</p> <p>Compression. Data compression reduces the number of bits contained in the information. Data compression becomes particularly important in the transmission of multimedia such as text, audio, and video.</p> <p>7). Application Layer</p> <p>The application layer enables the user, whether human or software, to access the network. It provides user interfaces and support for services such as electronic mail, remote file access and transfer, shared database management, and other types of distributed information services.</p> <p>In Figure shows the relationship of the application layer to the user and the presentation layer. Of the many application services available, the figure shows only three: XAOO (message-handling services), X.500 (directory services), and file transfer, access, and management (FTAM). The user in this example employs XAOO to send an e-mail message. Specific services provided by the application layer include the following:</p> <p>Network virtual terminal. A network virtual terminal is a software version of a physical terminal, and it allows a user to log on to a remote host. To do so, the application creates a software emulation of a terminal at the remote host. The user's computer talks to the software terminal which, in turn, talks to the host, and vice versa. The remote host believes it is communicating with one of its own terminals and allows the user to log on.</p> <p>File transfer, access, and management. This application allows a user to access files in a remote host (to make changes or read data), to retrieve files from a remote computer for use in the local computer, and to manage or control files in a remote computer locally.</p> <p>Mail services. This application provides the basis for e-mail forwarding and storage.</p> <p>Directory services. This application provides distributed database sources and access for global information about various objects and services.</p>
Q3(a)	<p>What is framing? Explain the four framing methods</p> <p>The usual approach is for the data link layer to break up the bit stream into discrete frames, compute a short token called a checksum for each frame, and include the checksum in the frame when it is transmitted. (Checksum algorithms will be discussed later in this chapter.) When a frame arrives at the destination, the checksum is recomputed.</p> <p>If the newly computed checksum is different from the one contained in the frame, the data link layer knows that an error has occurred and takes steps to deal with it (e.g., discarding the bad frame and possibly also sending back an error report). Breaking up the bit stream into frames is more difficult than it at first appears. A good design must make it easy for a receiver to find the start of new frames while using little of the channel bandwidth.</p> <p>We will look at four methods:</p> <ol style="list-style-type: none"> 1. Byte count. 2. Flag bytes with byte stuffing. 3. Flag bits with bit stuffing. 4. Physical layer coding violations.



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

The second framing method gets around the problem of resynchronization after an error by having each frame start and end with special bytes. Often the same byte, called a flag byte, is used as both the starting and ending delimiter.

The third method of delimiting the bit stream gets around a disadvantage of byte stuffing, which is that it is tied to the use of 8-bit bytes. Framing can be also be done at the bit level, so frames can contain an arbitrary number of bits made up of units of any size. It was developed for the once very popular HDLC (Highlevel Data Link Control) protocol.

(a) 0110111111111111111111110010

(b) 01101111101111101111101001

When the receiver sees five consecutive incoming 1 bits, followed by a 0 bit, it automatically destuffs (i.e., deletes) the 0 bit. Just as byte stuffing is completely transparent to the network layer in both computers, so is bit stuffing. If the user data contain the flag pattern, 01111110, this flag is transmitted as 011111010 but stored in the receiver's memory as 01111110.

Q3(b) Explain four different error-correcting codes.

We will examine four different error-correcting codes:

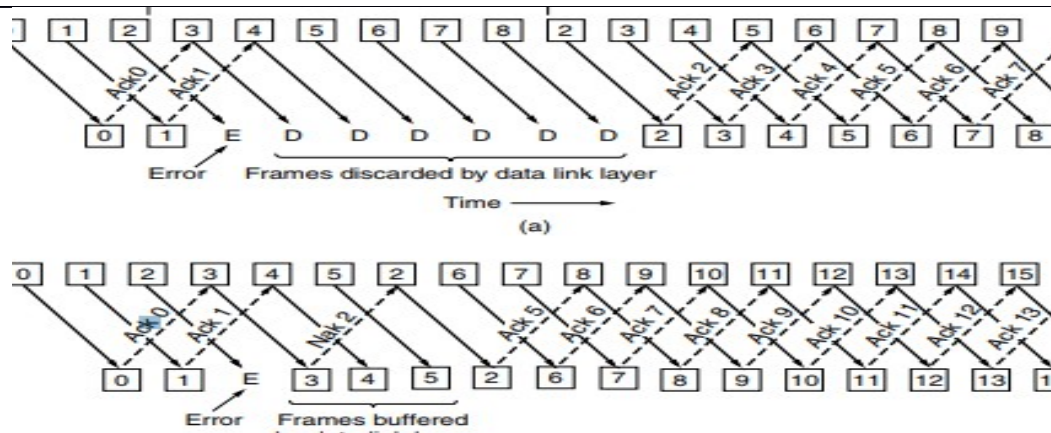
1. Hamming codes.
2. Binary convolutional codes.
3. Reed-Solomon codes.
4. Low-Density Parity Check codes.

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

To understand how errors can be handled, it is necessary to first look closely at what an error really is. Given any two codewords that may be transmitted or received—say,

	<p>10001001 and 10110001—it is possible to determine how many corresponding bits differ. In this case, 3 bits differ. To determine how many bits differ, just XOR the two codewords and count the number of 1 bits in the result. For example:</p> <pre> 10001001 10110001 00111000 </pre> <p>The number of bit positions in which two codewords differ is called the Hamming distance (Hamming, 1950). Its significance is that if two codewords are a Hamming distance d apart, it will require d single-bit errors to convert one into the other.</p> <p>The error-detecting and error-correcting properties of a block code depend on its Hamming distance. To reliably detect d errors, you need a distance $d + 1$ code because with such a code there is no way that d single-bit errors can change a valid codeword into another valid codeword. When the receiver sees an illegal codeword, it can tell that a transmission error has occurred. Similarly, to correct d errors, you need a distance $2d + 1$ code because that way the legal codewords are so far apart that even with d changes the original codeword is still closer than any other codeword. This means the original codeword can be uniquely determined based on the assumption that a larger number of errors are less likely. As a simple example of an error-correcting code, consider a code with only four valid codewords:</p> <p>000000000, 0000011111, 1111100000, and 1111111111.</p> <p>The third kind of error-correcting code we will describe is the Reed-Solomon code. Like Hamming codes, Reed-Solomon codes are linear block codes, and they are often systematic too. Unlike Hamming codes, which operate on individual bits, Reed-Solomon codes operate on m bit symbols. Naturally, the mathematics are more involved, so we will describe their operation by analogy. Reed-Solomon codes are based on the fact that every n degree polynomial is uniquely determined by $n + 1$ points. For example, a line having the form $ax + b$ is determined by two points. Extra points on the same line are redundant, which is helpful for error correction. Imagine that we have two data points that represent a line and we send those two data points plus two check points chosen to lie on the same line. If one of the points is received in error, we can still recover the data points by fitting a line to the received points. Three of the points will lie on the line, and one point, the one in error, will not. By finding the line we have corrected the error.</p> <p>The final error-correcting code we will cover is the LDPC (Low-Density Parity Check) code. LDPC codes are linear block codes that were invented by Robert Gallager in his doctoral thesis (Gallagher, 1962). Like most theses, they were promptly forgotten, only to be reinvented in 1995 when advances in computing power had made them practical. In an LDPC code, each output bit is formed from only a fraction of the input bits. This leads to a matrix representation of the code that has a low density of 1s, hence the name for the code. The received codewords are decoded with an approximation algorithm that iteratively improves on a best fit of the received data to a legal codeword. This corrects errors</p>
Q4(a)	<p>Explain the following:</p> <p>i) Simplex stop and wait protocol for error free channel.</p> <p>The communication channel is still assumed to be error free, however, and the data traffic is still simplex. One solution is to build the receiver to be powerful enough to process a continuous stream of back-to-back frames (or, equivalently, define the link layer to be slow enough that the receiver can keep up). It must have sufficient buffering and processing abilities to run at the line rate and must be able to pass the frames that are received to the network layer quickly enough. However, this is a worst-case solution. It requires dedicated hardware and can be wasteful of resources if the utilization of the link is mostly low. Moreover, it just shifts the problem of dealing with a sender that is too fast elsewhere; in this case to the network layer.</p> <p>Although data traffic in this example is simplex, going only from the sender to the receiver,</p>

	<p>frames do travel in both directions. Consequently, the communication channel between the two data link layers needs to be capable of bidirectional information transfer. However, this protocol entails a strict alternation of flow: first the sender sends a frame, then the receiver sends a frame, then the sender sends another frame, then the receiver sends another one, and so on. A halfduplex physical channel would suffice here. As in protocol 1, the sender starts out by fetching a packet from the network layer, using it to construct a frame, and sending it on its way. But now, unlike in protocol 1, the sender must wait until an acknowledgement frame arrives before looping back and fetching the next packet from the network layer. The sending data link layer need not even inspect the incoming frame as there is only one possibility. The incoming frame is always an acknowledgement. The only difference between receiver1 and receiver2 is that after delivering a packet to the network layer, receiver2 sends an acknowledgement frame back to the sender before entering the wait loop again. Because only the arrival of the frame back at the sender is important, not its contents, the receiver need not put any particular information in it.</p> <p>ii) Simplex stop and wait protocol for noisy channel.</p> <p>Frames may be either damaged or lost completely. However, we assume that if a frame is damaged in transit, the receiver hardware will detect this when it computes the checksum. If the frame is damaged in such a way that the checksum is nevertheless correct—an unlikely occurrence—this protocol (and all other protocols) can fail (i.e., deliver an incorrect packet to the network layer). At first glance it might seem that a variation of protocol 2 would work: adding a timer. The sender could send a frame, but the receiver would only send an acknowledgement frame if the data were correctly received. If a damaged frame arrived at the receiver, it would be discarded. After a while the sender would time out and send the frame again. This process would be repeated until the frame finally arrived intact.</p> <p>the network layer on B has no way of knowing that a packet has been lost or duplicated, so the data link layer must guarantee that no combination of transmission errors, however unlikely, can cause a duplicate packet to be delivered to a network layer. Consider the following scenario:</p> <ol style="list-style-type: none"> 1. The network layer on A gives packet 1 to its data link layer. The packet is correctly received at B and passed to the network layer on B. B sends an acknowledgement frame back to A. 2. The acknowledgement frame gets lost completely. It just never arrives at all. Life would be a great deal simpler if the channel mangled and lost only data frames and not control frames, but sad to say, the channel is not very discriminating. 3. The data link layer on A eventually times out. Not having received an acknowledgement, it (incorrectly) assumes that its data frame was lost or damaged and sends the frame containing packet 1 again. 4. The duplicate frame also arrives intact at the data link layer on B and is unwittingly passed to the network layer there. If A is sending a file to B, part of the file will be duplicated (i.e., the copy of the file made by B will be incorrect and the error will not have been detected). In other words, the protocol will fail.
Q4(b)	<p>Explain the Go-back-N sliding window protocol.</p> <p>Two basic approaches are available for dealing with errors in the presence of pipelining, both of which are shown in Fig.</p>



One option, called go-back-n, is for the receiver simply to discard all subsequent frames, sending no acknowledgements for the discarded frames. This strategy corresponds to a receive window of size 1. In other words, the data link layer refuses to accept any frame except the next one it must give to the network layer. If the sender's window fills up before the timer runs out, the pipeline will begin to empty. Eventually, the sender will time out and retransmit all unacknowledged frames in order, starting with the damaged or lost one. This approach can waste a lot of bandwidth if the error rate is high.

In Fig. we see go-back-n for the case in which the receiver's window is large. Frames 0 and 1 are correctly received and acknowledged. Frame 2, however, is damaged or lost. The sender, unaware of this problem, continues to send frames until the timer for frame 2 expires. Then it backs up to frame 2 and starts over with it, sending 2, 3, 4, etc. all over again.

In Fig. frames 0 and 1 are again correctly received and acknowledged and frame 2 is lost. When frame 3 arrives at the receiver, the data link layer there notices that it has missed a frame, so it sends back a NAK for 2 but buffers 3. When frames 4 and 5 arrive, they, too, are buffered by the data link layer instead of being passed to the network layer. Eventually, the NAK 2 gets back to the sender, which immediately resends frame 2. When that arrives, the data link layer now has 2, 3, 4, and 5 and can pass all of them to the network layer in the correct order. It can also acknowledge all frames up to and including 5, as shown in the figure. If the NAK should get lost, eventually the sender will time out for frame 2 and send it (and only it) of its own accord, but that may be a quite a while later.

These two alternative approaches are trade-offs between efficient use of bandwidth and data link layer buffer space. Depending on which resource is scarcer, one or the other can be used. Fig. shows a go-back-n protocol in which the receiving data link layer only accepts frames in order; frames following an error are discarded. In this protocol, for the first time we have dropped the assumption that the network layer always has an infinite supply of packets to send. When the network layer has a packet it wants to send, it can cause a network layer ready event to happen. However, to enforce the flow control limit on the sender window or the number of unacknowledged frames that may be outstanding at any time, the data link layer must be able to keep the network layer from bothering it with more work. The library procedures enable network layer and disable network layer do this job.

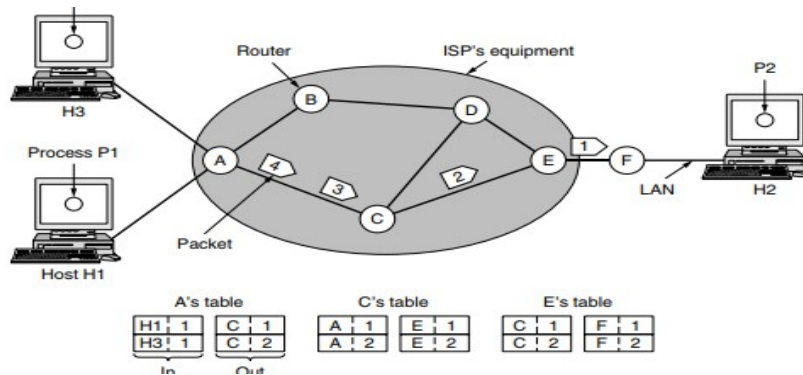
Q5(a)

How connection oriented services is implemented in network layer?

For connection-oriented service, we need a virtual-circuit network. Let us see how that works. The idea behind virtual circuits is to avoid having to choose a new route for every packet sent.

when a connection is established, a route from the source machine to the destination machine is chosen as part of the connection setup and stored in tables inside the routers. That route is used for all traffic flowing over the connection, exactly the same way that the telephone system works. When the connection is released, the virtual circuit is also terminated. With connection-oriented service, each packet carries an identifier telling which virtual circuit it belongs to.

As an example, consider the situation shown in Fig. Here, host H1 has established connection 1 with host H2. This connection is remembered as the first entry in each of the routing tables. The first line of A's table says that if a packet bearing connection identifier 1 comes in from H1, it is to be sent to router C and given connection identifier 1. Similarly, the first entry at C routes the packet to E, also with connection identifier 1.



Now let us consider what happens if H3 also wants to establish a connection to H2. It chooses connection identifier 1 (because it is initiating the connection and this is its only connection) and tells the network to establish the virtual circuit. This leads to the second row in the tables. Note that we have a conflict here because although A can easily distinguish connection 1 packets from H1 from connection 1 packets from H3, C cannot do this. For this reason, A assigns a different connection identifier to the outgoing traffic for the second connection. Avoiding conflicts of this kind is why routers need the ability to replace connection identifiers in outgoing packets.

Q5(b) Discuss about distance vector routing algorithm with suitable network.

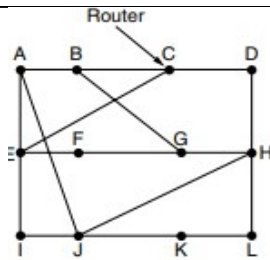
A distance vector routing algorithm operates by having each router maintain a table (i.e., a vector) giving the best known distance to each destination and which link to use to get there. These tables are updated by exchanging information with the neighbors. Eventually, every router knows the best link 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 and Fulkerson, 1962). It was the original ARPANET routing algorithm and was also used in the Internet under the name RIP.

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 msec, 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 X_i being X's estimate of how long it takes to get to router i. If the router knows that the delay to X is m msec, it also knows that it can reach router i via X in $X_i + m$ 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 msec, respectively.



To	A	I	H	K	delay from J	
A	0	24	20	21	8	A
B	12	36	31	28	20	A
C	25	18	19	36	28	I
D	40	27	8	24	20	H
E	14	7	30	22	17	I
F	23	20	19	40	30	I
G	18	31	6	31	18	H
H	17	20	0	19	12	H
I	21	0	14	22	10	I
J	9	11	7	10	0	-
K	24	22	22	0	6	K
L	29	33	9	9	15	K
JA	8	JI	JH	JK	New routing table for J	
delay is	8	delay is	delay is	delay is		
		10	12	6		

Vectors received from J's four neighbors

(a)

(b)

Consider how J computes its new route to router G. It knows that it can get to A in 8 msec, and furthermore A claims to be able to get to G in 18 msec, so J knows it can count on a delay of 26 msec 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) msec, respectively. The best of these values is 18, so it makes an entry in its routing table that the delay to G is 18 msec and that the route to use is via H.

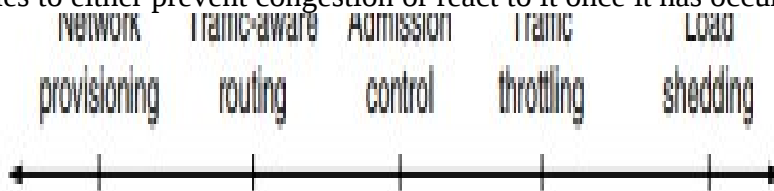
Q6(a)

What is congestion control? Discuss the approaches to congestion control.

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.

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. 5-22, these solutions are usually applied on different time scales to either prevent congestion or react to it once it has occurred.



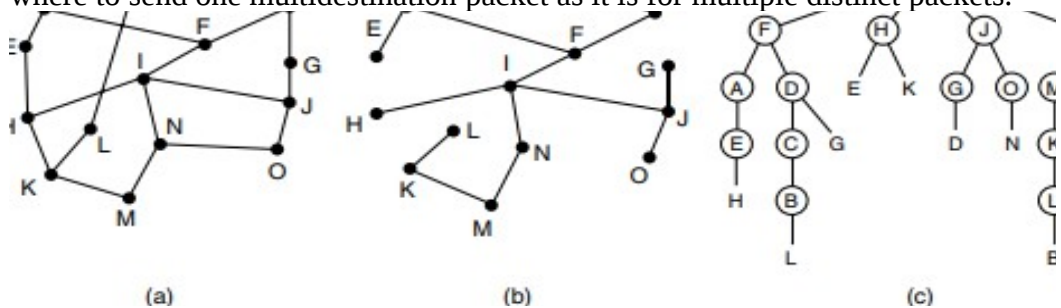
The most basic way to avoid congestion is to build a network that is well matched to the traffic that it carries. If there is a low-bandwidth link on the path along which most traffic is directed, congestion is likely. Sometimes resources can be added dynamically when there is serious congestion, for example, turning 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

routes can be tailored to traffic patterns that change during the day as network users 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.

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**.
 s, 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.

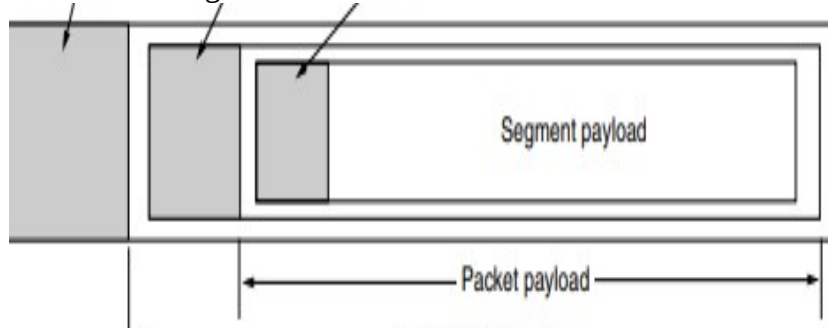
Q6(b) Discuss about Broadcast routing in network layer.
 Sending a packet to all destinations simultaneously is called broadcasting. Various methods have been proposed for doing it.
 One broadcasting method that requires no special features from the network is for the source to simply send a distinct packet to each destination. Not only is the method wasteful of bandwidth and slow, but it also requires the source to have a complete list of all destinations. This method is not desirable in practice, even though it is widely applicable.
 An improvement is multidestination routing, in which each packet contains either a list of destinations or a bit map indicating the desired destinations. When a packet arrives at a router, the router checks all the destinations to determine the set of output lines that will be needed. (An output line is needed if it is the best route to at least one of the destinations.) The router generates a new copy of the packet for each output line to be used and includes in each packet only those destinations that are to use the line. In effect, the destination set is partitioned among the output lines. After a sufficient number of hops, each packet will carry only one destination like a normal packet.
 Multidestination routing is like using separately addressed packets, except that when several packets must follow the same route, one of them pays full fare and the rest ride free. The network bandwidth is therefore used more efficiently. However, this scheme still requires the source to know all the destinations, plus it is as much work for a router to determine where to send one multidestination packet as it is for multiple distinct packets.



An example of reverse path forwarding is shown in Fig. . Part (a) shows a network, part (b) shows a sink tree for router I of that network, and part (c) shows how the reverse path algorithm works. On the first hop, I sends packets to F, H, J, and N, as indicated by the second row of the tree. Each of these packets arrives on the preferred path to I (assuming that the preferred path falls along the sink tree) and is so indicated by a circle around the letter. On the second hop, eight packets are generated, two by each of the routers that received a packet on the first hop. As it turns out, all eight of these arrive at previously unvisited routers, and five of these arrive along the preferred line. Of the six packets generated on the third hop, only three arrive on the preferred path (at C, E, and K); the others are duplicates. After five hops and 24 packets, the broadcasting terminates, compared with four hops and 14 packets had the sink tree been followed exactly.

Q7(a) Explain the following:
i) Nesting of segments, packets and frames.
 segments (exchanged by the transport layer) are contained in packets (exchanged by the network layer). In turn, these packets are contained in frames (exchanged by the data link layer). When a frame arrives, the data link layer processes the frame header and, if the destination address matches for local delivery, passes the contents of the frame payload field up to the network entity. The network entity similarly processes the packet header and

then passes the contents of the packet payload up to the transport entity. This nesting is illustrated in Fig.



Getting back to our client-server example, the client's CONNECT call causes a CONNECTION REQUEST segment to be sent to the server. When it arrives, the transport entity checks to see that the server is blocked on a LISTEN (i.e., is interested in handling requests). If so, it then unblocks the server and sends a CONNECTION ACCEPTED segment back to the client. When this segment arrives, the client is unblocked and the connection is established. Data can now be exchanged using the SEND and RECEIVE primitives. In the simplest form, either party can do a (blocking) RECEIVE to wait for the other party to do a SEND. When the segment arrives, the receiver is unblocked. It can then process the segment and send a reply.

ii) Socket primitives in TCP

The primitives are listed in Fig. Roughly speaking, they follow the model of our first example but offer more features and flexibility. We will not look at the corresponding segments here. That discussion will come later.

Primitive	Meaning
SOCKET	Create a new communication endpoint
BIND	Associate a local address with a socket
LISTEN	Announce willingness to accept connections; give queue size
ACCEPT	Passively establish an incoming connection
CONNECT	Actively attempt to establish a connection
SEND	Send some data over the connection
RECEIVE	Receive some data from the connection

The first four primitives in the list are executed in that order by servers. The SOCKET primitive creates a new endpoint and allocates table space for it within the transport entity. The parameters of the call specify the addressing format to be used, the type of service desired (e.g., reliable byte stream), and the protocol.

A successful SOCKET call returns an ordinary file descriptor for use in succeeding calls, the same way an OPEN call on a file does. Newly created sockets do not have network addresses. These are assigned using the BIND primitive. Once a server has bound an address to a socket, remote clients can connect to it.

The reason for not having the SOCKET call create an address directly is that some processes care about their addresses (e.g., they have been using the same address for years and everyone knows this address), whereas others do not. Next comes the LISTEN call, which allocates space to queue incoming calls for the case that several clients try to connect at the same time. In contrast to LISTEN in our first example, in the socket model LISTEN is not a blocking call.

Q7(b)	<p>Write socket programming with respect to internet file server.</p> <p>The code has many limitations (discussed below), but in principle the server code can be compiled and run on any UNIX system connected to the Internet. The client code can be compiled and run on any other UNIX machine on the Internet, anywhere in the world. The client code can be executed with appropriate parameters to fetch any file to which the server has access on its machine.</p> <p>The file is written to standard output, which, of course, can be redirected to a file or pipe. Let us look at the server code first. It starts out by including some standard headers, the last three of which contain the main Internet-related definitions and data structures. Next comes a definition of SERVER PORT as 12345.</p> <p>This number was chosen arbitrarily. Any number between 1024 and 65535 will work just as well, as long as it is not in use by some other process; ports below 1023 are reserved for privileged users. The next two lines in the server define two constants needed. The first one determines the chunk size in bytes used for the file transfer. The second one determines how many pending connections can be held before additional ones are discarded upon arrival.</p> <p>the server creates a socket and checks for errors (indicated by $s < 0$). In a production version of the code, the error message could be a trifle more explanatory. The call to <code>setsockopt</code> is needed to allow the port to be reused so the server can run indefinitely, fielding request after request. Now the IP address is bound to the socket and a check is made to see if the call to <code>bind</code> succeeded. The final step in the initialization is the call to <code>listen</code> to announce the server's willingness to accept incoming calls and tell the system to hold up to <code>QUEUE SIZE</code> of them in case new requests arrive while the server is still processing the current one. If the queue is full and additional requests arrive, they are quietly discarded.</p> <p>At this point, the server enters its main loop, which it never leaves. The only way to stop it is to kill it from outside. The call to <code>accept</code> blocks the server until some client tries to establish a connection with it. If the <code>accept</code> call succeeds, it returns a socket descriptor that can be used for reading and writing, analogous to how file descriptors can be used to read from and write to pipes. However, unlike pipes, which are unidirectional, sockets are bidirectional, so <code>sa</code> (the accepted socket) can be used for reading from the connection and also for writing to it. A pipe file descriptor is for reading or writing but not both.</p> <p>After the connection is established, the server reads the file name from it. If the name is not yet available, the server blocks waiting for it. After getting the file name, the server opens the file and enters a loop that alternately reads blocks from the file and writes them to the socket until the entire file has been copied. Then the server closes the file and the connection and waits for the next connection to show up. It repeats this loop forever.</p>
Q8(a)	<p>Write short notes on : i) TCP ii) UDP</p> <p>i)TCP</p> <p>A key feature of TCP, and one that dominates the protocol design, is that every byte on a TCP connection has its own 32-bit sequence number. When the Internet began, the lines between routers were mostly 56-kbps leased lines, so a host blasting away at full speed took over 1 week to cycle through the sequence numbers. At modern network speeds, the sequence numbers can be consumed at an alarming rate, as we will see later. Separate 32-bit sequence numbers are carried on packets for the sliding window position in one direction and for acknowledgements in the reverse direction, as discussed below. The sending and receiving TCP entities exchange data in the form of segments. A TCP segment consists of a</p>

fixed 20-byte header (plus an optional part) followed by zero or more data bytes. The TCP software decides how big segments should be. It can accumulate data from several writes into one segment or can split data from one write over multiple segments. Two limits restrict the segment size.

First, each segment, including the TCP header, must fit in the 65,515- byte IP payload. Second, each link has an MTU (Maximum Transfer Unit). Each segment must fit in the MTU at the sender and receiver so that it can be sent and received in a single, unfragmented packet. In practice, the MTU is generally 1500 bytes (the Ethernet payload size) and thus defines the upper bound on segment size.

When the segment arrives at the destination, the receiving TCP entity sends back a segment (with data if any exist, and otherwise without) bearing an acknowledgement number equal to the next sequence number it expects to receive and the remaining window size. If the sender's timer goes off before the acknowledgement is received, the sender transmits the segment again.

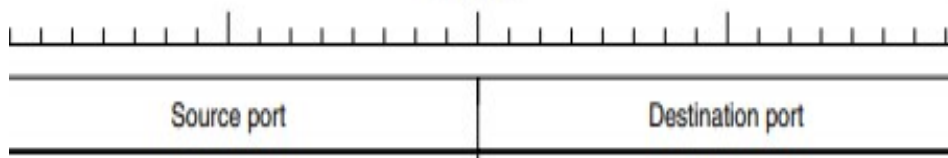
TCP must be prepared to deal with these problems and solve them in an efficient way. A considerable amount of effort has gone into optimizing the performance of TCP streams, even in the face of network problems. A number of the algorithms used by many TCP implementations will be discussed below.

ii)UDP:-

The Internet protocol suite supports a connectionless transport protocol called UDP (User Datagram Protocol). UDP provides a way for applications to send encapsulated IP datagrams without having to establish a connection. UDP is described in RFC 768.

UDP transmits segments consisting of an 8-byte header followed by the payload. The header is shown in Fig. 6-27. The two ports serve to identify the endpoints within the source and destination machines. When a UDP packet arrives, its payload is handed to the process attached to the destination port.

UDP). Think of ports as mailboxes that applications can rent to receive packets. We will have more to say about them when we describe TCP, which also uses ports. In fact, the main value of UDP over just using raw IP is the addition of the source and destination ports. Without the port fields, the transport layer would not know what to do with each incoming packet. With them, it delivers the embedded segment to the correct application.



The source port is primarily needed when a reply must be sent back to the source. By copying the Source port field from the incoming segment into the Destination port field of the outgoing segment, the process sending the reply can specify which process on the sending machine is to get it. The UDP length field includes the 8-byte header and the data. The minimum length is 8 bytes, to cover the header. The maximum length is 65,515 bytes, which is lower than the largest number that will fit in 16 bits because of the size limit on IP packets.

An optional Checksum is also provided for extra reliability. It checksums the header, the data, and a conceptual IP pseudoheader. When performing this computation, the Checksum field is set to zero and the data field is padded out with an additional zero byte if its length is an odd number. The checksum algorithm is simply to add up all the 16-bit words in one's complement and to take the one's complement of the sum.

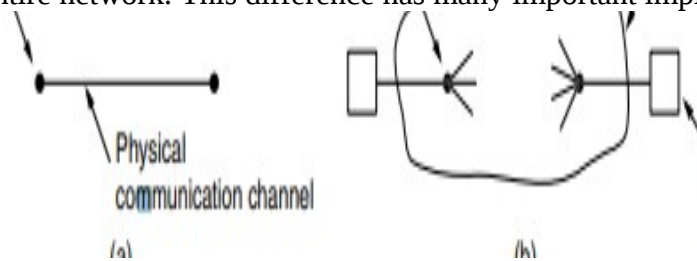
As a consequence, when the receiver performs the calculation on the entire segment, including the Checksum field, the result should be 0. If the checksum is not computed, it is stored as a 0, since by a happy coincidence of one's complement arithmetic a true computed 0 is stored as all 1s. However, turning it off is foolish unless the quality of the data does not matter (e.g., for digitized speech).

Q8(b)

Discuss about the elements of transport protocols.

The transport service is implemented by a transport protocol used between the two transport entities.

At the data link layer, two routers communicate directly via a physical channel, whether wired or wireless, whereas at the transport layer, this physical channel is replaced by the entire network. This difference has many important implications for the protocols.



For one thing, over point-to-point links such as wires or optical fiber, it is usually not necessary for a router to specify which router it wants to talk to—each outgoing line leads directly to a particular router. In the transport layer, explicit addressing of destinations is required.

Another (exceedingly annoying) difference between the data link layer and the transport layer is the potential existence of storage capacity in the network. When a router sends a packet over a link, it may arrive or be lost, but it cannot bounce around for a while, go into hiding in a far corner of the world, and suddenly emerge after other packets that were sent much later. If the network uses datagrams, which are independently routed inside, there is a nonnegligible probability that a packet may take the scenic route and arrive late and out of the expected order, or even that duplicates of the packet will arrive. The consequences of the network's ability to delay and duplicate packets can sometimes be disastrous and can require the use of special protocols to correctly transport information.

A final difference between the data link and transport layers is one of degree rather than of kind. Buffering and flow control are needed in both layers, but the presence in the transport layer of a large and varying number of connections with bandwidth that fluctuates as the connections compete with each other may require a different approach than we used in the data link layer.

Q9(a)

Explain the following i) Domain Name System ii) Electronic Mail.

Domain Name System

Although programs theoretically could refer to Web pages, mailboxes, and other resources by using the network (e.g., IP) addresses of the computers on which they are stored, these addresses are hard for people to remember. Also, browsing a company's Web pages from 128.111.24.41 means that if the company moves the Web server to a different machine with a different IP address, everyone needs to be told the new IP address. Consequently, high-level, readable names were introduced in order to decouple machine names from machine addresses.

this way, the company's Web server might be known as www.cs.washington.edu regardless of its IP address. Nevertheless, since the network itself understands only numerical addresses, some mechanism is required to convert the names to network addresses. In the following sections, we will study how this mapping is accomplished in the Internet.

However, well before many millions of PCs were connected to the Internet, everyone involved with it realized that this approach could not continue to work forever. For one thing, the size of the file would become too large. However, even more importantly, host name conflicts would occur constantly unless names were centrally managed, something unthinkable in a huge international network due to the load and latency. To solve these problems, DNS (Domain Name System) was invented in 1983. It has been a key part of the Internet ever since.

The essence of DNS is the invention of a hierarchical, domain-based naming scheme and a distributed database system for implementing this naming scheme. It is primarily used for

	<p>mapping host names to IP addresses but can also be used for other purposes. DNS is defined in RFCs 1034, 1035, 2181, and further elaborated in many others</p> <p>Electronic Mail</p> <p>Email, like most other forms of communication, has developed its own conventions and styles. It is very informal and has a low threshold of use. People who would never dream of calling up or even writing a letter to a Very Important Person do not hesitate for a second to send a sloppily written email to him or her. By eliminating most cues associated with rank, age, and gender, email debates often focus on content, not status. With email, a brilliant idea from a summer student can have more impact than a dumb one from an executive vice president.</p> <p>we will provide an overview of how email systems are organized and what they can do. The architecture of the email system is shown in Fig. 7-7. It consists of two kinds of subsystems: the user agents, which allow people to read and send email, and the message transfer agents, which move the messages from the source to the destination. We will also refer to message transfer agents informally as mail servers.</p> <p>The message transfer agents are typically system processes. They run in the background on mail server machines and are intended to be always available. Their job is to automatically move email through the system from the originator to the recipient with SMTP (Simple Mail Transfer Protocol). This is the message transfer step.</p> <p>Message transfer agents also implement mailing lists, in which an identical copy of a message is delivered to everyone on a list of email addresses. Other advanced features are carbon copies, blind carbon copies, high-priority email, secret (i.e., encrypted) email, alternative recipients if the primary one is not currently available, and the ability for assistants to read and answer their bosses' email.</p>
<p>Q9(b)</p>	<p>Explain the DNS resource record types.</p> <p>Every domain, whether it is a single host or a top-level domain, can have a set of resource records associated with it. These records are the DNS database. For a single host, the most common resource record is just its IP address, but many other kinds of resource records also exist.</p> <p>When a resolver gives a domain name to DNS, what it gets back are the resource records associated with that name. Thus, the primary function of DNS is to map domain names onto resource records.</p> <p>A resource record is a five-tuple. Although they are encoded in binary for efficiency, in most expositions resource records are presented as ASCII text, one line per resource record. The format we will use is as follows:</p> <p>Domain name Time to live Class Type Value</p> <p>The Domain name tells the domain to which this record applies. Normally, many records exist for each domain and each copy of the database holds information about multiple domains. This field is thus the primary search key used to satisfy queries. The order of the records in the database is not significant.</p> <p>The Time to live field gives an indication of how stable the record is. Information that is highly stable is assigned a large value, such as 86400 (the number of seconds in 1 day). Information that is highly volatile is assigned a small value, such as 60 (1 minute). We will come back to this point later when we have discussed caching.</p> <p>The third field of every resource record is the Class. For Internet information, it is always IN. For non-Internet information, other codes can be used, but in practice these are rarely seen.</p> <p>The Type field tells what kind of record this is. There are many kinds of DNS records. The important types are listed in Fig. 7-3. An SOA record provides the name of the primary source of information about the name server's zone (described below), the email address of its administrator, a unique serial number, and various flags and timeouts.</p> <p>The most important record type is the A (Address) record. It holds a 32-bit IPv4 address of</p>

	<p>an interface for some host. The corresponding AAAA, or “quad A,” record holds a 128-bit IPv6 address. Every Internet host must have at least one IP address so that other machines can communicate with it. Some hosts have two or more network interfaces, in which case they will have two or more type A or AAAA resource records. Consequently, DNS can return multiple addresses for a single name.</p>
<p>Q10(a))</p>	<p>Discuss about static and dynamic web pages.</p> <p>static web page:-</p> <p>The basis of the Web is transferring Web pages from server to client. In the simplest form, Web pages are static. That is, they are just files sitting on some server that present themselves in the same way each time they are fetched and viewed. Just because they are static does not mean that the pages are inert at the browser, however. A page containing a video can be a static Web page. As mentioned earlier, the lingua franca of the Web, in which most pages are written, is HTML. The home pages of teachers are usually static HTML pages.</p> <p>The home pages of companies are usually dynamic pages put together by a Web design company. In this section, we will take a brief look at static HTML pages as a foundation for later material. Readers already familiar with HTML can skip ahead to the next section, where we describe dynamic content and Web services.</p> <p>dynamic web page:-</p> <p>The static page model we have used so far treats pages as multimedia documents that are conveniently linked together. It was a fitting model in the early days of the Web, as vast amounts of information were put online. Nowadays, much of the excitement around the Web is using it for applications and services. Examples include buying products on e-commerce sites, searching library catalogs, exploring maps, reading and sending email, and collaborating on documents. These new uses are like traditional application software (e.g., mail readers and word processors). The twist is that these applications run inside the browser, with user data stored on servers in Internet data centers. They use Web protocols to access information via the Internet, and the browser to display a user interface. The advantage of this approach is that users do not need to install separate application programs, and user data can be accessed from different computers and backed up by the service operator. It is proving so successful that it is rivaling traditional application software. Of course, the fact that these applications are offered for free by large providers helps. This model is the prevalent form of cloud computing, in which computing moves off individual desktop computers and into shared clusters of servers in the Internet.</p> <p>There is more to dynamic content, however. The page that is returned may itself contain programs that run in the browser. In our map example, the program would let the user find routes and explore nearby areas at different levels of detail. It would update the page, zooming in or out as directed by the user (step 4). To handle some interactions, the program may need more data from the server. In this case, the program will send a request to the server (step 5) that will retrieve more information from the database (step 6) and return a response (step 7). The program will then continue updating the page (step 4). The requests and responses happen in the background; the user may not even be aware of them because the page URL and title typically do not change. By including client-side programs, the page can present a more responsive interface than with server-side programs alone.</p>
<p>Q10(b)</p>	<p>Explain in detail about server farm and web proxies.</p> <p>Server Farms</p> <p>It can only serve so many Web requests before the load is too great. The solution in this case is to use more than one computer to make a Web server.</p> <p>The difficulty with this seemingly simple model is that the set of computers that make up the server farm must look like a single logical Web site to clients. If they do not, we have just set up different Web sites that run in parallel. There are several possible solutions to make the set of servers appear to be one Web site. All of the solutions assume that any of the servers can handle a request from any client. To do this, each server must have a copy of</p>

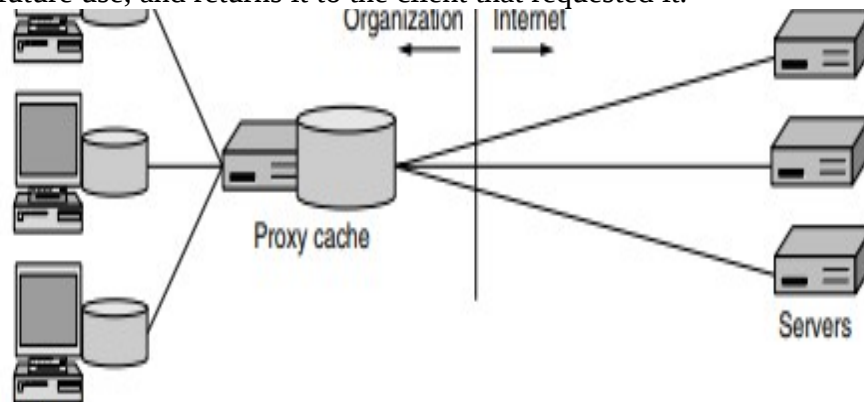
the Web site. The servers are shown as connected to a common back-end database by a dashed line for this purpose.

Web Proxies

Web requests and responses are sent using HTTP. A Web proxy is used to share a cache among users. A proxy is an agent that acts on behalf of someone else, such as the user. There are many kinds of proxies. For instance, an ARP proxy replies to ARP requests on behalf of a user who is elsewhere (and cannot reply for himself). A Web proxy fetches Web requests on behalf of its users. It normally provides caching of the Web responses, and since it is shared across users it has a substantially larger cache than a browser.

When a proxy is used, the typical setup is for an organization to operate one Web proxy for all of its users. The organization might be a company or an ISP. Both stand to benefit by speeding up Web requests for its users and reducing its bandwidth needs. While flat pricing, independent of usage, is common for end users, most companies and ISPs are charged according to the bandwidth that they use.

This setup is shown in Fig. To use the proxy, each browser is configured to make page requests to the proxy instead of to the page's real server. If the proxy has the page, it returns the page immediately. If not, it fetches the page from the server, adds it to the cache for future use, and returns it to the client that requested it.



As well as sending Web requests to the proxy instead of the real server, clients perform their own caching using its browser cache. The proxy is only consulted after the browser has tried to satisfy the request from its own cache. That is, the proxy provides a second level of caching.