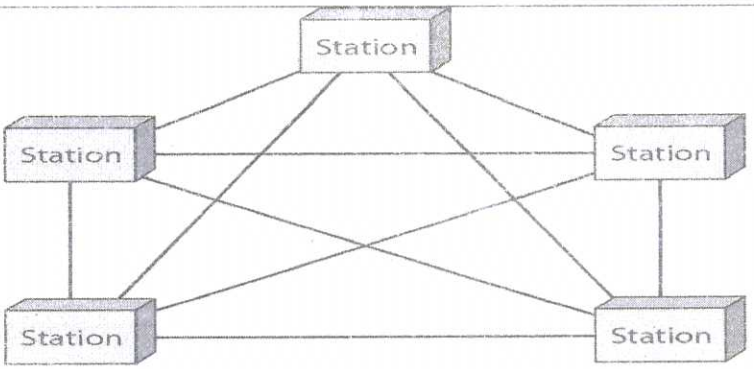
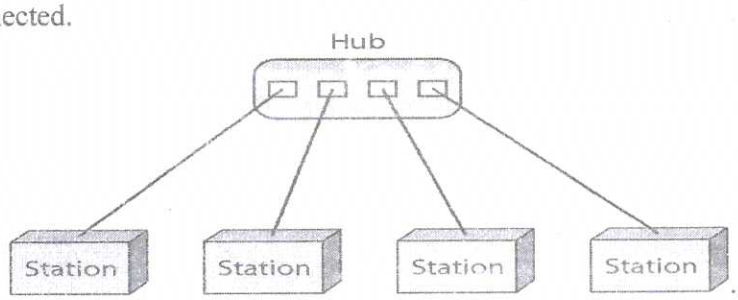
Scoring Indicators

Code: 6131 **Version:**

Qn: No	Scoring Indicators	Split score	Total score
I 1	• Physical layer • Datalink layer • Network layer • Transport layer • Application Layer (List any two 2x1=2)	2x1	2
I 2	**Ethernet** is the traditional technology for connecting wired local area networks (LANs), enabling devices to communicate with each other via a protocol.	1X 2	2
I 3	1. Backpressure 2. Choke Packet 3. Implicit Signaling 4. Explicit Signaling (List any two)	2x1	2
I 4	Congestion in a network may occur if the *load* on the network—the number of packets sent to the network—is greater than the *capacity* of the network—the number of packets a network can handle.	1x2	2
I 5	SMTP,FTP,HTTP,DNS,TELNET (List any two)	2X 1	2
II 1	**Mesh Topology** In a mesh topology, every device has a dedicated point-to-point link to every other device. The term *dedicated* means that the link carries traffic only between the two devices it connects. To find the number of physical links in a fully connected mesh network with n nodes, we first consider that each node must be connected to every other node. Node 1 must be connected to n - 1 nodes, node 2 must be connected to n – 1 nodes, and finally node n must be connected to n - 1 nodes. We need $n(n - 1)$ physical links. However, if each physical link allows communication in both directions (duplex mode), we can divide the number of links by 2. In other words, we can say that in a mesh topology, we need $n(n - 1)/2$ duplex-mode links. To accommodate that many links, every device on the network must have n – 1 input/output *(I/O)* ports to be connected to the other n - 1 stations.	3x2=6	6

	 **Star Topology** In a star topology, each device has a dedicated point-to-point link only to a central controller, usually called a hub. The devices are not directly linked to one another. Unlike a mesh topology, a star topology does not allow direct traffic between devices. The controller acts as an exchange: If one device wants to send data to another, it sends the data to the controller, which then relays the data to the other connected.  (1.5 marks for fig+1.5 marks for explanation of each topology)		
II 2	**1. Packetizing** The first duty of the network layer is definitely **packetizing:** encapsulating the payload(data received from upper layer) in a network-layer packet at the source and decapsulating the payload from the network-layer packet at the destination. The network layer is responsible for delivery of packages from a sender to a receiver without changing or using the contents. The source host receives the payload from an upper-layer protocol, adds a header that contains the source and destination addresses and some other information that is required by the network-layer protocol and delivers the packet to the data-link layer. The source is not allowed to change the content of the payload unless it is too large for delivery and needs to be fragmented. The destination host receives the network-layer packet from its data-link layer, de capsulates the packet, and delivers the payload to the corresponding upper-layer protocol. If the packet is fragmented at the source or at routers along the path, the network layer is responsible for waiting until all fragments arrive,	3x2=6	6

reassembling them, and delivering them to the upper-layer protocol.

The routers in the path are not allowed to decapsulate the packets they received unless the packets need to be fragmented.

2. Routing and Forwarding

2.1. Routing

The network layer is responsible for routing the packet from its source to the destination.A physical network is a combination of networks (LANs and WANs) and routers that connect them. This means that there is more than one route from the source to the destination. The network layer is responsible for finding the best one among these possible
routes. The network layer needs to have some specific strategies for defining the best route. In the Internet today, this is done by running some routing protocols to help the routers coordinate their knowledge about the neighborhood and to come up with consistent tables to be used when a packet arrives.

2.2. Forwarding

Forwarding can be defined as the action applied by each router when a packet arrives at one of its interfaces. The decision-making table a router normally uses for applying this action is sometimes called the forwarding table and sometimes the routing table. When a router receives a packet from one of its attached networks, it needs to forward the packet to another attached network (in unicast routing) or to some attached networks (in multicast routing). To make this decision, the router uses a piece of information in the packet header, which can be the destination address or a label, to find the corresponding output interface number in the forwarding table.

3. Error Control

The designers of the network layer, however, have added a checksum field to the datagram to control any corruption in the header, but not in the whole datagram. This checksum may prevent any
changes or corruptions in the header of the datagram.

4. Flow Control

Flow control regulates the amount of data a source can send without overloading the
receiver. If the upper layer at the source computer produces data faster than the upper layer at the destination computer can consume it, the receiver will be overload with data. To control the flow of data, the receiver needs to send some feedback to the sender to inform the latter that it is overload with data.

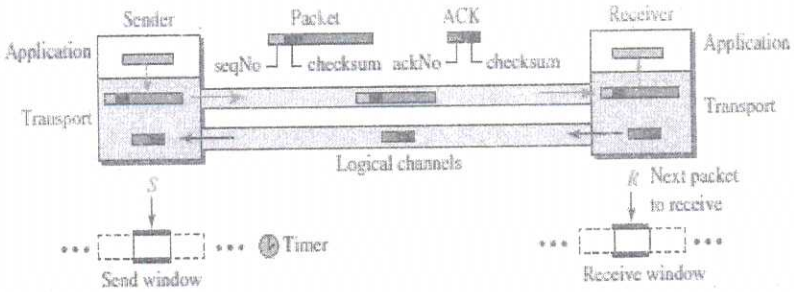
The network layer in the Internet, however, does not directly provide any flow control.
The datagrams are sent by the sender when they are ready, without any attention to
the readiness of the receiver.

5. Congestion Control

Another issue in a network-layer protocol is congestion control. Congestion in the network layer is a situation in which too

	many datagrams are present in an area of the Internet. Congestion may occur if the number of datagrams sent by source computers is beyond the capacity of the network or routers. In this situation, some routers may drop some of the datagrams. However, as more datagrams are dropped, the situation may become worse because, due to the error control mechanism at the upper layers, the sender may send duplicates of the lost packets. If the congestion continues, sometimes a situation may reach a point where the system collapses and no datagrams are delivered. **6. Quality of Service** As the Internet has allowed new applications such as multimedia communication (in particular real-time communication of audio and video), the quality of service (QoS) of the communication has become more and more important. The Internet should provide better quality of service to support these applications **7. Security** Security was not a concern when the Internet was originally designed because it was used by an small number of users at universities for research activities; other people had no access to the Internet. The network layer was designed with no security provision. (write any 3 services)		
II 3	**1. Delay** The delays in a network can be divided into four types: • transmission delay • propagation delay • processing delay • queuing delay **1.1. Transmission Delay** A source host or a router cannot send a packet instantaneously. A sender needs to put the bits in a packet on the line one by one. If the first bit of the packet is put on the line at time t1 and the last bit is put on the line at time t2, transmission delay of the packet is (t2 − t1). Definitely, the transmission delay is longer for a longer packet and shorter if the sender can transmit faster. **Delaytr = (Packet length) / (Transmission rate)** **1.2. Propagation Delay** Propagation delay is the time it takes for a bit to travel from point A to point B in the transmission media. The propagation delay for a packet-switched network depends on the propagation delay of each network (LAN or WAN). The propagation delay depends on the propagation speed of the media, which is 3 × 108 meters/second in a vacuum and normally much less in a wired medium; it also depends on the distance of the link. In other words, propagation delay is **Delaytr = (Packet length) / (Transmission rate).**	1.5X4= 6	6

	Delaypg = (Distance) / (Propagation speed). **1.3. Processing Delay** The processing delay is the time required for a router or a destination host to receive a packet from its input port, remove the header, perform an error detection procedure, and deliver the packet to the output port (in the case of a router) or deliver the packet to the upper-layer protocol (in the case of the destination host). The processing delay may be different for each packet, but normally is calculated as an average. **Delaypr = Time required to process a packet in a router or a destination host** **1.4. Queuing Delay** Queuing delay can normally happen in a router.A router has an input queue connected to each of its input ports to store packets waiting to be processed; the router also has an output queue connected to each of its output ports to store packets waiting to be transmitted. The queuing delay for a packet in a router is measured as the time a packet waits in the input queue and output queue of a router. **Delayqu = The time a packet waits in input and output queues in a router** **1.5. Total Delay** Assuming equal delays for the sender, routers, and receiver, the total delay (source-todestination delay) a packet encounters can be calculated if we know the number of routers, n, in the whole path. **Total delay = (*n* + 1) (Delaytr + Delaypg + Delaypr) + (*n*) (Delayqu)** (1.5 Marks for each delay)		
II 4	**2. Stop-and-Wait Protocol** Our second protocol is a connection-oriented protocol called the **Stop-and-Waitprotocol,** which uses both flow and error control. Both the sender and the receiver use a sliding window of size 1. The sender sends one packet at a time and waits for an acknowledgment before sending the next one. To detect corrupted packets, we need to add a checksum to each data packet. When a packet arrives at the receiver site, it is checked. If its checksum is incorrect, the packet is corrupted and silently discarded. The silence of the receiver is a signal for the sender that a packet was either corrupted or lost. Every time the sender sends a packet, it starts a timer. If an acknowledgment arrives before the timer expires, the timer is stopped and the sender sends the next packet. If the timer expires, the sender resends the previous packet, assuming that the packet was either lost or corrupted. This means that the sender needs to keep a copy of the packet until its acknowledgment arrives. Figure shows the outline for the Stop-and-Wait protocol. Note that only one packet and one acknowledgment can be in the channels at any time.	2+4=6	6



Stop-and-Wait protocol

Sequence Numbers

To prevent duplicate packets, the protocol uses sequence numbers and acknowledgment numbers. A field is added to the packet header to hold the sequence number of that packet. Smallest range of sequence numbers is used that provides unambiguous communication.

Assume we have used x as a sequence number; we only need to use $x + 1$ after that.
There is no need for $x + 2$. To show this, assume that the sender has sent the packet with sequence number x. Three things can happen.

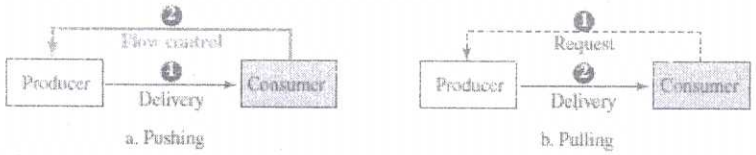
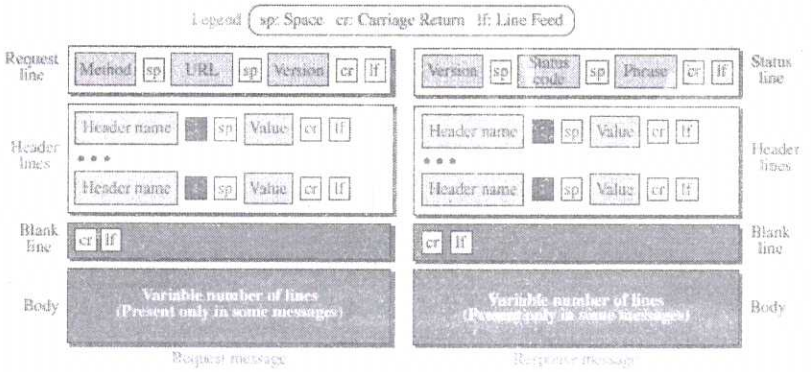
1. The packet arrives safe and sound at the receiver site; the receiver sends an acknowledgment. The acknowledgment arrives at the sender site, causing the sender to send the next packet numbered $x + 1$.

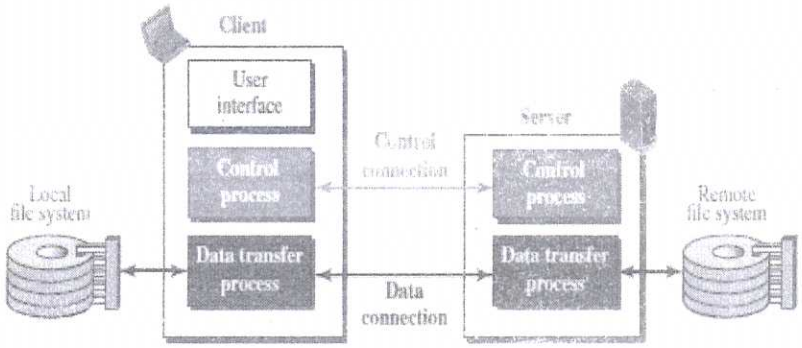
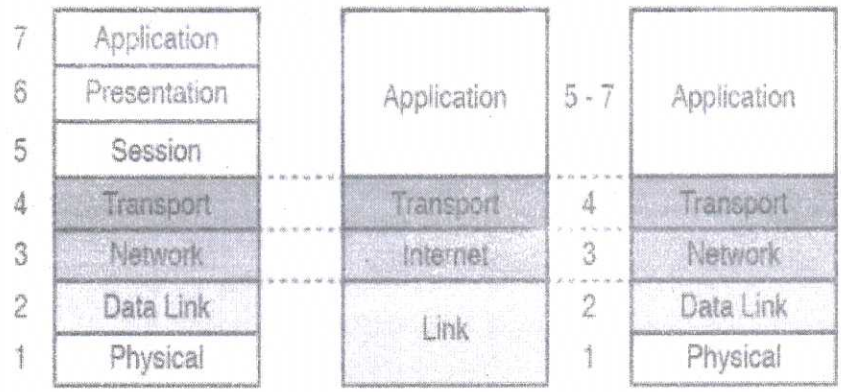
2. The packet is corrupted or never arrives at the receiver site; the sender resends the packet (numbered x) after the time-out. The receiver returns an acknowledgment.

3. The packet arrives safe and sound at the receiver site; the receiver sends an acknowledgment, but the acknowledgment is corrupted or lost. The sender resends the packet numbered x after the time-out. The packet here is a duplicate. The receiver can recognize this fact because it expects packet $x + 1$ but packet x
was received.

Acknowledgment Numbers

Since the sequence numbers must be suitable for both data packets and acknowledgments, we use this convention: The acknowledgment numbers always announce the sequence number of the *next packet expected* by the receiver. For example, if packet 0 has arrived safe and sound, the receiver sends an ACK with acknowledgment 1 (meaning packet 1 is expected next). If packet 1 has arrived safe and sound, the receiver sends an ACK with acknowledgment 0 (meaning packet 0 is expected).

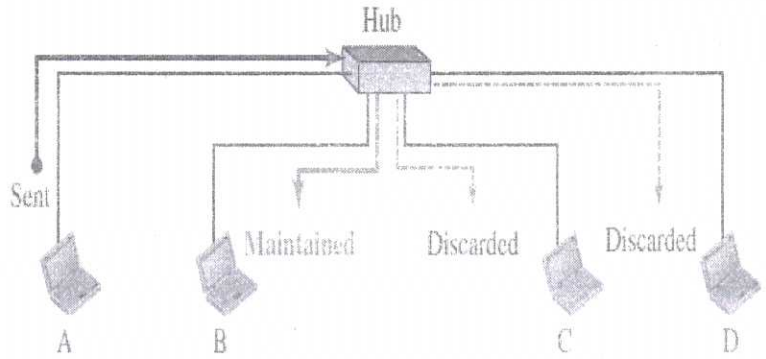
	(2 marks for figure+4 marks for explanation)		
II 5	**Pushing or Pulling** Delivery of items from a producer to a consumer can occur in one of two ways: pushing or pulling. If the sender delivers items whenever they are produced without a prior request from the consumer the delivery is referred to as **pushing**. If the producer delivers the items after the consumer has requested them, the delivery is referred to as **pulling**. Figure 23.9 shows these two types of delivery.  ***pushing or pulling*** When the producer *pushes* the items, the consumer may be overwhelmed and there is a need for flow control, in the opposite direction, to prevent discarding of the items. In other words, the consumer needs to warn the producer to stop the delivery and to inform the producer when it is again ready to receive the items. When the consumer pulls the items, it requests them when it is ready. In this case, there is no need for flow control. (1 Mark for figure+ 2 marks for explanations)	(1+2)x2 =6	6
II 6	**Message Formats** The HTTP protocol defines the format of the request and response messages, as shown in Figure . Each message is made of four sections. The first section in the request message is called the *request line;* the first section in the response message is called the *status line*. The other three sections have the same names in the request and response messages. (3 Marks)  (3 Marks)	3+3=6	6
II	**File Transfer Protocol (FTP)** is the standard protocol provided by	3+3=6	6

7	TCP/IP for copying a file from one host to another  Figure shows the basic model of FTP. The client has three components: **the user interface, the client control process, and the client data transfer process.** The server has two components: **the server control process and the server data transfer process.** The control connection is made between the control processes. The data connection is made between the data transfer processes. The control connection uses very simple rules of communication. We need to transfer only a line of command or a line of response at a time. The data connection, on the other hand, needs more complex rules due to the variety of data types transferred. (3 Marks for fig+3 Marks for explanation)		
III a	 ***Fig: Layers in TCP/IP Protocol Suit*** **Physical Layer** It is responsible for the transmission and reception of bits • **Data rate** – the number of bits per second that can be sent • **Bit Synchronization** – The sender and receiver must be synchronized at	3+2x3= 9	15

	the symbol level so that the number of bits expected per unit time is the same. • **Configuration** – Defines point to point or multipoint link • **Topology** – Defines different types of topology like mesh,star,ring ,bus • **Mode** – Defines simplex, half duplex and full duplex **Data link Layer** • **Framing** – Divides the stream of bits to frames • **Physical Addressing** – Adds the address to the header of the frame to identify the system. • **Flow Control** – Controls the flow of data from sender to receiver • **Error Control** – Mechanism to detect damage and lost frames • **Access Control** – Decides which devices can access the link **Network Layer** • Responsible for source to destination delivery of packet • **Logical Addressing** • Adds the universal address to the to the packet to identify the system • **Routing** • Routes the packet from source to destination **Transport Layer** • **Resposible for Process to process Delivery**.		

	• **Adds port addresses** – To identify specific process • **Segmentation and reassembly** – Message is divided in to transmittable segments with seq no. • **Connection control** – Connection oriented(TCP-Reliable) – connectionless services(UDP-Unreliable) • **Flow control** – Controls the flow of data from sender to receiver • **Error Control** – Mechanism to detect damage,lost,duplicate message **Application Layer** • **Session Establishment** o Establishes and maintains the interaction between two systems • **Data conversions** o Converts the data from sender dependant format to receiver dependant format • **Compresssion** o Reduce the no of bits contained in the information • **Encryption** o For privacy sender transforms the information into coded format • **Remote Access** o Allows application to access the files in the remote host (3 marks for fig+3 marks for each layer)		
III b	**HUB** A **hub** is a device that operates only in the physical layer. Ethernet LANs use star topology. In a star topology, a repeater(A **repeater** receives a signal and, before it becomes too weak or	(2 X 3)=6	

corrupted, *regenerates* and *retimes* the original bit pattern) is a multiport device, often called a *hub*, that can be used to serve as the connecting point and at the same time function as a repeater.



All stations in the LAN receive the frame, but only the receiving station keeps it. The rest of the stations discard it.

A hub does not have a filtering capability; it does not have the intelligence to find from which port the frame should be sent out.

Link-Layer Switches

A **link-layer switch** (or ***switch***) operates in both the physical and the data-link layers.

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

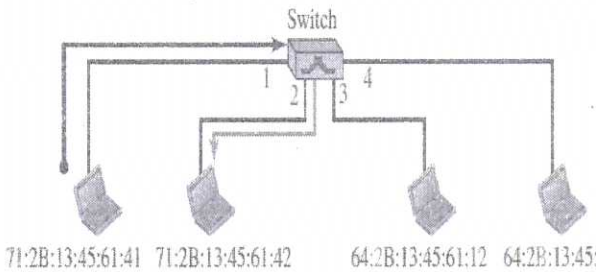
Filtering

A link-layer switch has **filtering** capability. It can check the destination address of

a frame and can decide from which outgoing port the frame should be sent. Let us give an example. In Figure below we have a LAN with four stations that are connected to a link-layer switch. If a frame destined for station 71:2B:13:45:61:42 arrives at port 1, the link-layer switch consults its table to find the departing port.

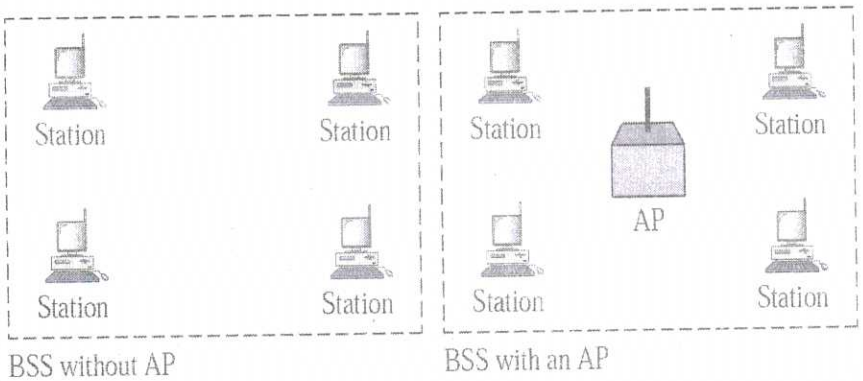
According to its table, frames for 71:2B:13:45:61:42 should be sent out only through

port 2; therefore, there is no need for forwarding the frame through other ports.



Switching table

Address	Port
71:2B:13:45:61:41	1
71:2B:13:45:61:42	2
64:2B:13:45:61:12	3
64:2B:13:45:61:13	4

	Routers A **router** is a three-layer device; it operates in the physical, data-link, and network layers.As a physical-layer device, it regenerates the signal it receives. As a link-layer device, the router checks the physical addresses (source and destination) contained in the packet. As a network-layer device, a router checks the network-layer addresses. A router can connect networks. In other words, a router is an internetworking device; it connects independent networks to form an internetwork. According to this definition, two networks connected by a router become an internetwork or an internet. There are three major differences between a router and a repeater or a switch. **1.** A router has a physical and logical (IP) address for each of its interfaces. **2.** A router acts only on those packets in which the link-layer destination address matches the address of the interface at which the packet arrives. **3.** A router changes the link-layer address of the packet (both source and destination) when it forwards the packet. (Answer any two.3 marks for each device)		
IV a	**IEEE 802.11 PROJECT** IEEE has defined the specifications for a wireless LAN, called IEEE 802.11. It is sometimes called *wireless Ethernet.* **Architecture** The standard defines two kinds of services: the basic service set (BSS) and the extended service set (ESS). **Basic Service Set**  BSS without AP BSS with an AP IEEE 802.11 defines the **basic service set (BSS)** as the building blocks of a wireless LAN. A basic service set is made of stationary or mobile wireless	3+3+3= 9	15

	stations and an optional central base station, known as the *access point (AP).* The BSS without an AP is a stand-alone network and cannot send data to other BSSs. It is called an *ad hoc architecture*. In this architecture, stations can form a network without the need of an AP; they can locate one another and agree to be part of a BSS. A BSS with an AP is sometimes referred to as an *infrastructure BSS.* **Extended Service Set** An **extended service set (ESS)** is made up of two or more BSSs with APs. In this case, the BSSs are connected through a *distribution system,* which is a wired or a wireless network. The distribution system connects the APs in the BSSs. IEEE 802.11 does not restrict the distribution system; it can be any IEEE LAN such as an Ethernet. When BSSs are connected, the stations within reach of one another can communicate without the use of an AP. However, communication between a station in a BSS and the outside BSS occurs via the AP. (3 marks for figure + 3 marks for BSS+3 marks for ESS)		
IV b	**Membership** **Interface Numbers** Some VLAN vendors use switch interface numbers as a membership characteristic. For example, the administrator can define that stations connecting to ports 1, 2, 3, and 7 belong to VLAN 1, stations connecting to ports 4, 10, and 12 belong to VLAN 2, and so on. **MAC Addresses** Some VLAN vendors use the 48-bit MAC address as a membership characteristic. For example, the administrator can stipulate that stations having MAC addresses E2:13:42:A1:23:34 and F2:A1:23:BC:D3:41 belong to VLAN 1. **IP Addresses** Some VLAN vendors use the 32-bit IP address (see Chapter 18) as a membership characteristic. For example, the administrator can stipulate that stations having IP addresses 181.34.23.67, 181.34.23.72, 181.34.23.98, and 181.34.23.112 belong to VLAN 1. **Multicast IP Addresses** Some VLAN vendors use the multicast IP address as a membership characteristic. Multicasting at the IP layer is now translated to multicasting at the datalink layer. (1.5 marks for each membership)	1.5x4=6	
V	When the Internet started, an IPv4 address was divided into five	6+3=9	15

a

classes (class A, B, C, D, and E). This scheme is referred to as **classful addressing.**

In class A,

The network length is 8 bits, but since the first bit, which is 0, defines the class, we can have only seven bits as the network identifier. This means there are only $2^7 = 128$ networks in the world that can have a class A address.

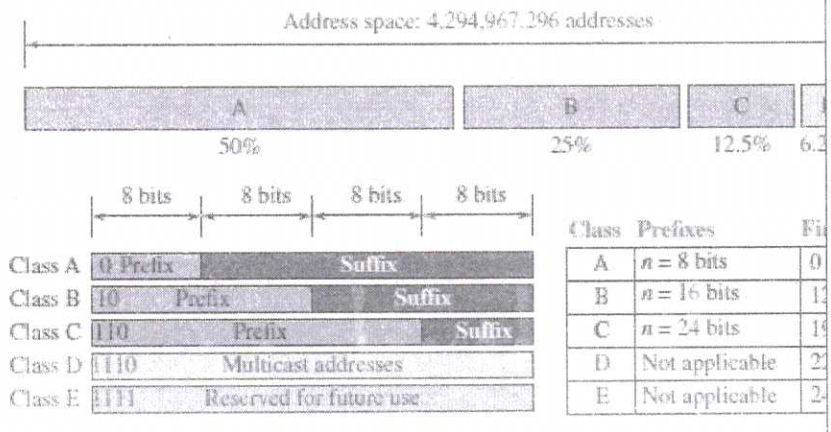
In class B,

the network length is 16 bits, but since the first two bits, which are (10), define the class, we can have only 14 bits as the network identifier. This means there are only $2^{14} = 16,384$ networks in the world that can have a class B address.

In class C

All addresses that start with (110)2 belong to class C. In class C, the network length is 24 bits, but since three bits define the class, we can have only 21 bits as the network identifier. This means there are $2^{21} = 2,097,152$ networks in the world that can have a class C address.

Figure 18.18 *Occupation of the address space in classful addressing*



Class	Prefixes	Fi
A	$n = 8$ bits	0
B	$n = 16$ bits	1
C	$n = 24$ bits	1
D	Not applicable	2
E	Not applicable	2

Class D is not divided into prefix and suffix. It is used for multicast addresses. All

addresses that start with 1111 in binary belong to class E.

Class E is not divided into prefix and suffix and is used. as reserve

In **Classless Addressing**, the class is removed from the distribution to compensate for the address depletion.

In classless addressing, the whole address space is divided into variable length

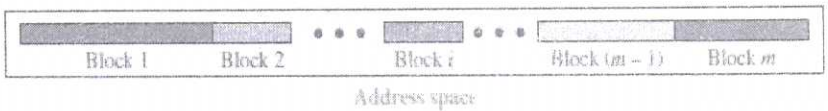
blocks. The prefix in an address defines the block (network); the suffix defines the node

(device). Theoretically, we can have a block of 2^0, 2^1, 2^2,...2^{32} addresses. One of the

restrictions, is that the number of addresses in a block needs to be a power of 2. An organization can be granted one block of addresses. Figure 18.19 shows

the division of the whole address space into nonoverlapping blocks.

Figure 18.19 *Variable-length blocks in classless addressing*



Unlike classful addressing, the prefix length in classless addressing is variable. We can have a prefix length that ranges from 0 to 32.

We need to emphasize that the idea of classless addressing can be easily applied to
classful addressing. An address in class A can be thought of as a classless address in which the prefix length is 8. An address in class B can be thought of as a classless address in which the prefix is 16, and so on. Classful addressing is a special case of classless addressing.
(6 marks for classful addressing+ 3 Marks for classless addressing)

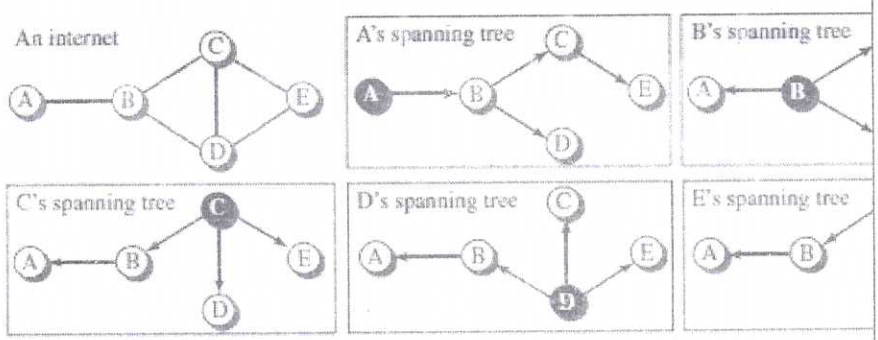
V b

Path-Vector Routing

Spanning Trees

In path-vector routing, the path from a source to all destinations is also determined by the *best* spanning tree. The best spanning tree, however, is not the least-cost tree; it is the tree determined by the source when it imposes its own policy. If there is more than one route to a destination, the source can choose the route that meets its policy best. A source may apply several policies at the same time. One of the common policies uses the minimum number of nodes to be visited. Another common policy is to avoid some nodes as the middle node in a route.
Figure 20.11 shows a small internet with only five nodes. Each source has created its own spanning tree that meets its policy. The policy imposed by all sources is to use the minimum number of nodes to reach a destination. The spanning tree selected by A and E is such that the communication does not pass through D as a middle node. Similarly, the spanning tree selected by B is such that the communication does not pass through C as a middle node. (3 + 3)=6

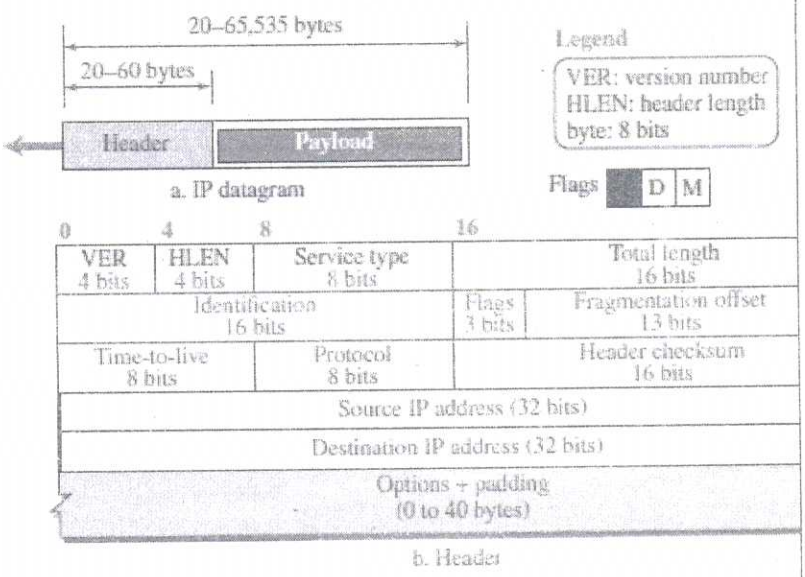
Figure 20.11 *Spanning trees in path-vector routing*



Creation of Spanning Trees

When a node is booted, it creates a *path vector* based on the information it can obtain about its immediate neighbor. A node sends greeting messages to its immediate

	neighbors to collect these pieces of information. Note, however, that we do not mean that all of these tables are created simultaneously; they are created when each node is booted. The figure also shows how these path vectors are sent to immediate neighbors after they have been created (arrows). Each node, after the creation of the initial path vector, sends it to all its immediate neighbors. Each node, when it receives a path vector from a neighbor, updates its path vector using an equation similar to the Bellman-Ford, but applying its own policy instead of looking for the least cost. We can define this equation as $Path(x, y) = best\ \{Path(x, y), [(x + Path(v, y)]\}$ for all v's in the internet. (3 Marks for fig +3 Marks for explanation)		
VI a	**Datagram Format** Packets used by the IP are called *datagrams*. Figure 19.2 shows the IPv4 datagram format. A datagram is a variable-length packet consisting of two parts: header and payload (data). The header is 20 to 60 bytes in length and contains information essential to routing and delivery. ❑ ***Version Number.*** The 4-bit version number (VER) field defines the version of the IPv4 protocol, which, obviously, has the value of 4. ❑ ***Header Length.*** The 4-bit header length (HLEN) field defines the total length of the datagram header in 4-byte words. The IPv4 datagram has a variable-length header. ❑ ***Service Type.*** In the original design of the IP header, this field was referred to as type of service (TOS), which defined how the datagram should be handled. ❑ ***Total Length.*** This 16-bit field defines the total length (header plus data) of the IP datagram in bytes. A 16-bit number can define a total length of up to 65,535 (when all bits are 1s). ❑ ***Identification, Flags, and Fragmentation Offset.*** These three fields are related to the fragmentation of the IP datagram when the size of the datagram is larger than the underlying network can carry.	3+6=9	15

	Figure 19.2 *IP datagram*  ❑ ***Time-to-live***. The time-to-live (TTL) field is used to control the maximum number of hops (routers) visited by the datagram. When a source host sends the datagram, it stores a number in this field. This value is approximately two times the maximum number of routers between any two hosts. Each router that processes the datagram decrements this number by one. If this value, after being decremented, is zero, the router discards the datagram. ❑ ***Protocol.*** In TCP/IP, the data section of a packet, called the *payload,* carries the whole packet from another protocol. A datagram, for example, can carry a packet belonging to any transport-layer protocol such as UDP or TCP. (3 Marks for fig+6 Marks for explanation)		
VI b	**Security of IPv4 Datagrams** There are three security issues that are particularly applicable to the IP protocol: packet sniffing, packet modification, and IP spoofing. ***1. Packet Sniffing*** An intruder may intercept an IP packet and make a copy of it. Packet sniffing is a passive attack, in which the attacker does not change the contents of the packet. This type of attack is very difficult to detect because the sender and the receiver may never know that the packet has been copied. Although packet sniffing cannot be stopped, encryption of the packet can make the attacker's effort useless. ***2. Packet Modification*** The second type of attack is to modify the packet. The attacker intercepts the packet,changes its contents, and sends the new packet to the receiver. The receiver believes that the packet is coming from the original sender. This type of attack can be detected using a data integrity mechanism. ***3. IP Spoofing***	4x1.5=6	

	An attacker can masquerade as somebody else and create an IP packet that carries the source address of another computer. An attacker can send an IP packet to a bank pretending that it is coming from one of the customers. This type of attack can be prevented using an origin authentication mechanism. *4. IPSec* The IP packets today can be protected from the previously mentioned attacks using a protocol called IPSec (IP Security). This protocol, which is used in conjunction with the IP protocol, creates a connection-oriented service between two entities in which they can exchange IP packets without worrying about the three attacks discussed above. IPSec provides the following four services: ❑ ***Defining Algorithms and Keys.*** ❑ ***Packet Encryption.*** ❑ ***Data Integrity.*** ❑ ***Origin Authentication.*** (1.5 marks for each)		
VI I a	**Connectionless Service** In a connectionless service, the source process (application program) needs to divide its message into chunks of data of the size acceptable by the transport layer and deliver them to the transport layer one by one. The transport layer treats each chunk as a single unit without any relation between the chunks. When a chunk arrives from the application layer, the transport layer encapsulates it in a packet and sends it. To show the independency of packets, assume that a client process has three chunks of messages to send to a server process. The chunks are handed over to the connectionless transport protocol in order. However, since there is no dependency between the packets at the transport layer, the packets may arrive out of order at the destination and will be delivered out of order to the server process. The figure shows that at the client site, the three chunks of messages are delivered to the client transport layer in order (0, 1, and 2). Because of the extra delay in transportation of the second packet, the delivery of messages at the server is not in order (0, 2, 1). If these three chunks of data belong to the same message, the server process may have received a strange message. The situation would be worse if one of the packets were lost. Since there is no numbering on the packets, the receiving transport layer has no idea that one of the messages has been lost. It just delivers two chunks of data to the server process.	(2+2)x2 =8	15

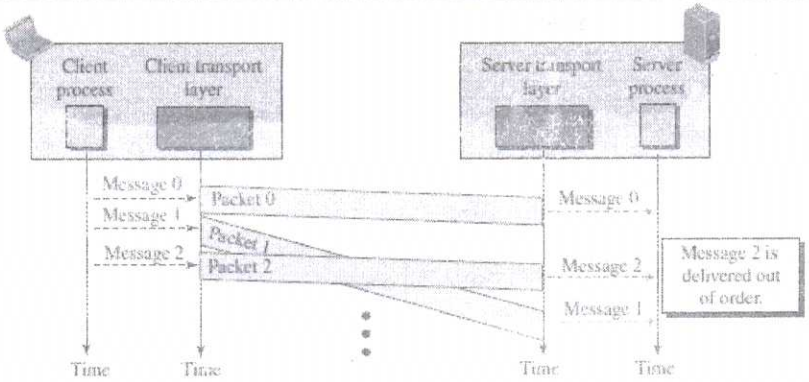


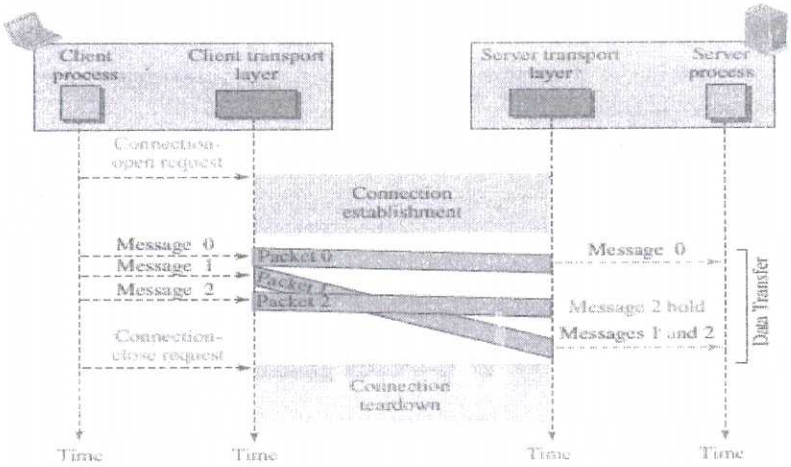
Fig:Connectionless service

The above two problems arise from the fact that the two transport layers do not
coordinate with each other. The receiving transport layer does not know when the first packet will come nor when all of the packets have arrived.We can say that no flow control, error control, or congestion control can be effectively implemented in a connectionless service.

Connection-Oriented Service

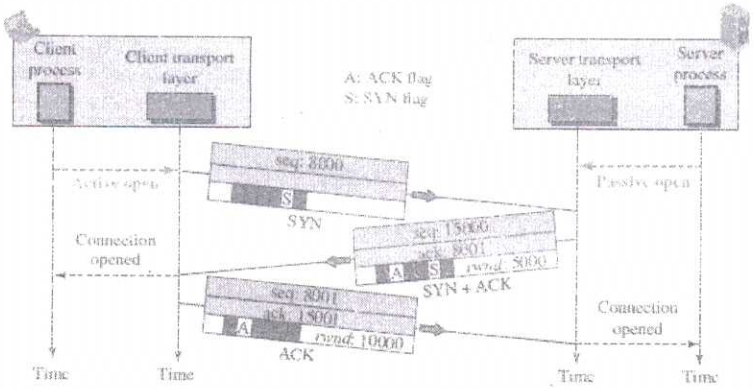
In a connection-oriented service, the client and the server first need to establish a
logical connection between themselves. The data exchange can only happen after the connection establishment. After data exchange, the connection needs to be torn down.

At the transport layer, connection-oriented service involves only the two hosts; the service is end to end. Figure shows the connection establishment, data-transfer, and tear-down phases in a connection-oriented service at the transport layer. We can implement flow control, error control, and congestion control in a connection oriented protocol.



Connection-oriented service

	(2 marks for fig+2 marks for explanation)		
VI I b	**A TCP Connection** TCP is connection-oriented. A connection-oriented transport protocol establishes a logical path between the source and destination. All of the segments belonging to a message are then sent over this logical path. Using a single logical pathway for the entire message facilitates the acknowledgment process as well as retransmission of damaged or lost frames. TCP operates at a higher level. TCP uses the services of IP to deliver individual segments to the receiver, but it controls the connection itself. If a segment is lost or corrupted, it is retransmitted. In TCP, connection-oriented transmission requires three phases: connection establishment, data transfer, and connection termination. **Connection Establishment** TCP transmits data in full-duplex mode. When two TCPs in two machines are connected, they are able to send segments to each other simultaneously. This implies that each party must initialize communication and get approval from the other party before any data are transferred. **Three-Way Handshaking** The connection establishment in TCP is called ***three-way handshaking.*** In our example, an application program, called the *client,* wants to make a connection with another application program, called the *server,* using TCP as the transport-layer protocol. The process starts with the server. The server program tells its TCP that it is ready to accept a connection. This request is called a *passive open.* Although the server TCP is ready to accept a connection from any machine in the world, it cannot make the connection itself. The client program issues a request for an *active open.* A client that wishes to connect to an open server tells its TCP to connect to a particular server. TCP can now start the three-way handshaking process, as shown in Figure.	3+4=7	

	 ***Connection establishment using three-way handshaking*** To show the process we use time lines. The three steps in this phase are as follows. 1. The client sends the first segment, a SYN segment, in which only the SYN flag is set. This segment is for synchronization of sequence numbers. The client in our example chooses a random number as the first sequence number and sends this number to the server. This sequence number is called the *initial sequence number (ISN)*. The SYN segment is a control segment and carries no data. However, it consumes one sequence number because it needs to be acknowledged. **2.** The server sends the second segment, a SYN [illegible] ACK segment with two flag bits set as: SYN and ACK. This segment has a dual purpose. First, it is a SYN segment for communication in the other direction. The server uses this segment to initialize a sequence number for numbering the bytes sent from the server to the client. The server also acknowledges the receipt of the SYN segment from the client by setting the ACK flag and displaying the next sequence number it expects to receive from the client. 2. The client sends the third segment. This is just an ACK segment. It acknowledges the receipt of the second segment with the ACK flag and acknowledgment number field. (3 Marks for figure+4 Marks for explanation)		
VIII a	**SCTP Features** The following shows the general features of SCTP. **1. Transmission Sequence Number (TSN)** The unit of data in SCTP is a data chunk, which may or may not have a one-to-one relationship with the message coming from the process because of fragmentation. Data transfer in SCTP is controlled by	1+1.5x4 =7	15

numbering the data chunks. SCTP uses a **transmission sequence number (TSN)** to number the data chunks. In other words, the TSN in SCTP plays a role analogous to the sequence number in TCP.TSNs are 32 bits long and randomly initialized between 0 and 232 – 1. Each data chunk
must carry the corresponding TSN in its header.

2. Stream Identifier (SI)

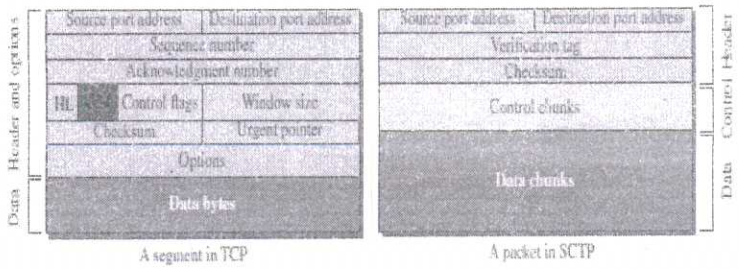
In SCTP, there may be several streams in each association. Each stream in SCTP needs to be identified using a **stream identifier (SI).** Each data chunk must carry the SI in its header so that when it arrives at the destination, it can be properly placed in its stream. The SI is a 16-bit number starting from 0.

3. Stream Sequence Number (SSN)

When a data chunk arrives at the destination SCTP, it is delivered to the appropriate stream and in the proper order. This means that, in addition to an SI, SCTP defines each data chunk in each stream with a **stream sequence number (SSN).**

4. Packets

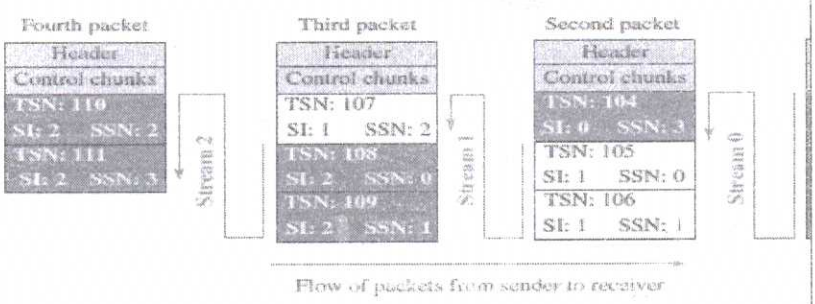
In TCP, a segment carries data and control information. Data are carried as a collection of bytes; control information is defined by six control flags in the header. The design of SCTP is totally different: data are carried as data chunks, control information as control chunks. Several control chunks and data chunks can be packed together in a packet. A packet in SCTP plays the same role as a segment in TCP. Figure compares a segment in TCP and a packet in SCTP.



Comparison between a TCP segment and an SCTP packet

In SCTP, we have data chunks, streams, and packets. An association may send
many packets, a packet may contain several chunks, and chunks may belong to different streams.

Note that each data chunk needs three identifiers: TSN, SI, and SSN. TSN is acumulative number and is used. SI defines the stream to

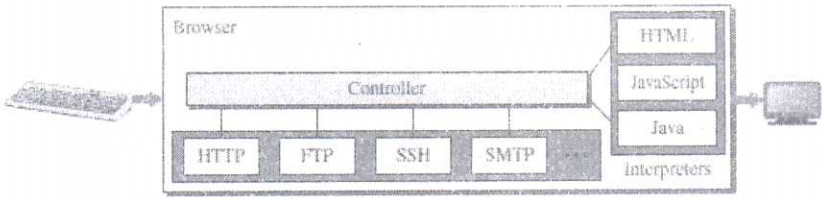
	which the chunk belongs. SSN defines the chunk's order in a particular stream. In our example, SSN starts from 0 for each stream. ***Packets, data chunks, and streams***  **5. Acknowledgment Number** TCP acknowledgments numbers are byte-oriented and refer to the sequence numbers.SCTP acknowledgment numbers are chunk-oriented. They refer to the TSN. A second difference between TCP and SCTP acknowledgments is the control information. This information is part of the segment header in TCP. To acknowledge segments that carry only control information, TCP uses a sequence number and acknowledgment number .In SCTP, however, the control information is carried by control chunks, which do not need a TSN. These control chunks are acknowledged by another control chunk of the appropriate type (some need no acknowledgment). (1 mark for listing+1.5 Marks for each features(answer any 4 features)		
VI II b	**UDP Services** **1. Process-to-Process Communication** UDP provides process-to-process communication using **socket addresses,** a combination of IP addresses and port numbers. **2. Connectionless Services** UDP provides a *connectionless service.* This means that each user datagram sent by UDP is an independent datagram. There is no relationship between the different user datagrams even if they are coming from the same source process and going to the same destination program. The user datagrams are not numbered. Also, unlike TCP, there is no connection establishment and no connection termination. This means that each user datagram can travel on a different path. Only those processes sending short messages, messages less than 65,507 bytes (65,535 minus 8 bytes for the UDP header and minus 20 bytes for the IP header), can use UDP. **3. Flow Control**	8x1=8	

	UDP is a very simple protocol. There is no *flow control,* and hence no window mechanism.The receiver may overflow with incoming messages. The lack of flow control means that the process using UDP should provide for this service, if needed. **4. 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. When the receiver detects an error through the checksum, the user datagram is silently discarded. The lack of error control means that the process using UDP should provide for this service, if needed. **5. Checksum** UDP checksum calculation includes three sections: a pseudoheader, the UDP header, and the data coming from the application layer. The *pseudoheader* 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. **6. Congestion Control** Since UDP is a connectionless protocol, it does not provide congestion control. UDP assumes that the packets sent are small and sporadic and cannot create congestion in the network. This assumption may or may not be true today, when UDP is used for interactive real-time transfer of audio and video. **7. Encapsulation and Decapsulation** To send a message from one process to another, the UDP protocol encapsulates and decapsulates messages. **8. Queuing** In UDP, queues are associated with ports. At the client site, when a process starts, it requests a port number from the operating system. Some implementations create both an incoming and an outgoing queue associated with each process. Other implementations create only an incoming queue associated with each process. **9. Multiplexing and Demultiplexing** In a host running a TCP/IP protocol suite, there is only one UDP but possibly several processes that may want to use the services of UDP. To handle this situation, UDP multiplexes and demultiplexes. (List any 8 services)		
IX a	**Architecture** The WWW today is a distributed client-server service, in which a client using a browser can access a service using a server. However, the service provided is distributed over many locations called *sites.* Each site holds one or more web pages. Each web page, however, can contain some links to other web pages in the same or other sites. In other words, a web page can be simple or composite. A simple web	3+2+3= 8	15

page has no links to other web pages; a composite web page has one or more links to other web pages. Each web page is a file with a name and address.

Web Client (Browser)

A variety of vendors offer commercial browsers that interpret and display a web
page, and all of them use nearly the same architecture. Each browser usually consists of three parts: a controller, client protocols, and interpreters.



The controller receives input from the keyboard or the mouse and uses the client programs to access the document. After the document has been accessed, the controller uses one of the interpreters to display the document on the screen. The client protocol can be one of the protocols, such as HTTP or FTP. The interpreter can be HTML, Java, or JavaScript, depending on the type of document. Some commercial browsers include Internet Explorer, Netscape Navigator, and Firefox.

(3 Marks(2 marks for fig+1 mark for explanation))

Web Server

The web page is stored at the server. Each time a request arrives, the corresponding document is sent to the client. A server can also become more efficient through multithreading or multiprocessing. In this case, a server can answer more than one request at a time. Some popular web servers include Apache and Microsoft Internet Information Server.

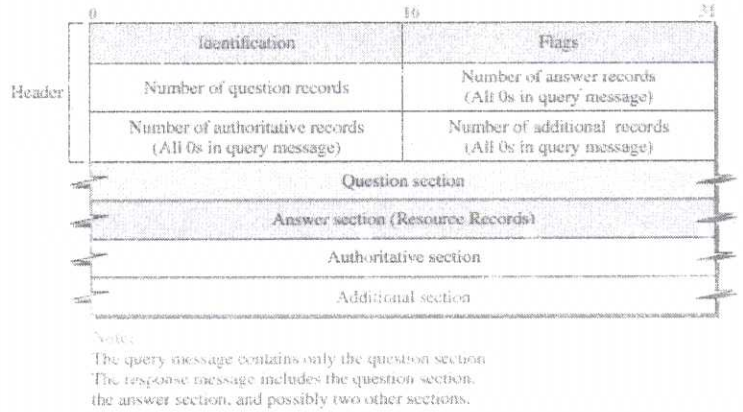
(2 Marks)

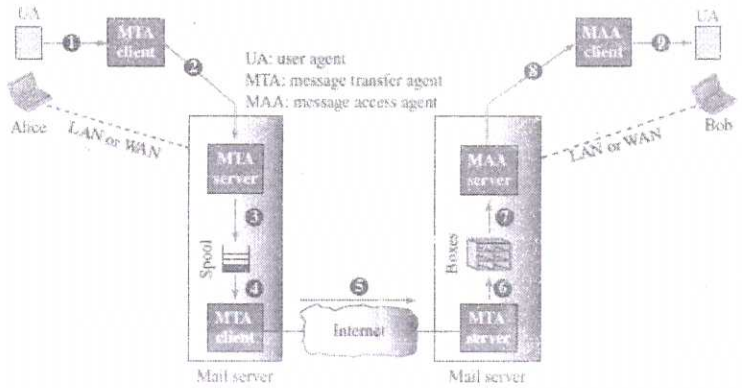
Uniform Resource Locator (URL)

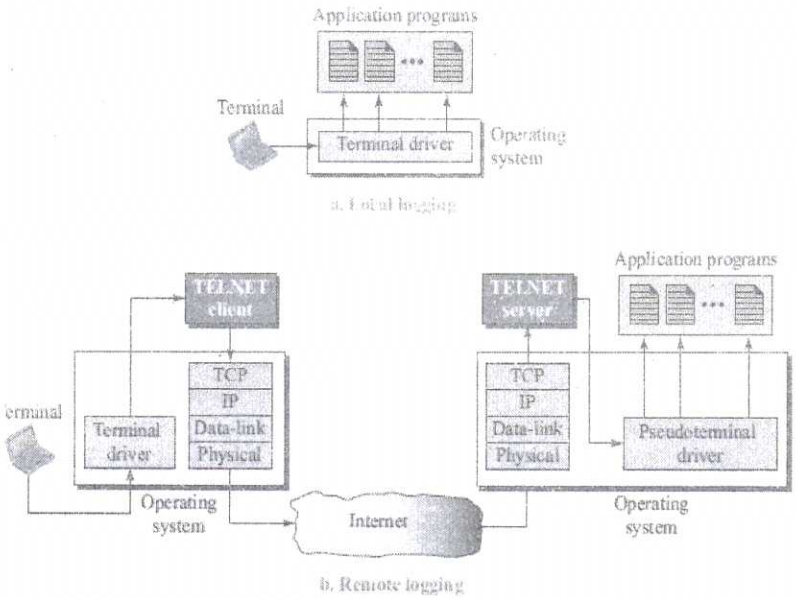
A web page, as a file, needs to have a unique identifier to distinguish it from other web pages. To define a web page, we need three identifiers: ***host, port***, and ***path***.

However, before defining the web page, we need to tell the browser what client server application we want to use, which is called the *protocol*. This means we need four identifiers to define the web page. The first is the type of vehicle to be used to fetch the web page; the last three make up the combination that defines the destination object (web page).

❑ Protocol. The first identifier is the abbreviation for the client-server program that we need in order to access the web page. Although most of the time the protocol is HTTP (HyperText Transfer Protocol), we can also use other protocols such as FTP (File Transfer Protocol).

	❑ **Host.** The host identifier can be the IP address of the server or the unique name given to the server. IP addresses can be defined in dotted decimal notation, ; the name is normally the domain name that uniquely defines the host. ❑ **Port.** The port, a 16-bit integer, is normally predefined for the client-server application.For example, if the HTTP protocol is used for accessing the web page, the well-known port number is 80. However, if a different port is used, the number can be explicitly given. ❑ **Path.** The path identifies the location and the name of the file in the underlying operating system. **protocol://host/path -Used most of the time** **protocol://host:port/path -Used when port number is needed** **(3 Marks)**		
IX b	**DNS Messages** To retrieve information about hosts, DNS uses two types of messages: *query* and *response*. Both types have the same format as shown in Figure.  Notes: The query message contains only the question section. The response message includes the question section, the answer section, and possibly two other sections. **Fields in a DNS message.** • The identification field is used by the client to match the response with the query. • The flag field defines whether the message is a query or response. It also includes status of error. • The next four fields in the header define the number of each record type in the message. The question section consists of one or more question records. It is present in both query and response messages.	3+4=7	

	• The answer section consists of one or more resource records. It is present only in response messages. • The authoritative section gives information (domain name) about one or more authoritative servers for the query. • The additional information section provides additional information that may help the resolver. (3 marks for fig+4 marks for explanation)		
X a	 Consider a common scenario, as shown in Figure . Another possibility is the case in which Alice or Bob is directly connected to the corresponding mail server, in which LAN or WAN connection is not required. In the common scenario, the sender and the receiver of the e-mail, Alice and Bob respectively, are connected via a LAN or a WAN to two mail servers. The administrator has created one mailbox for each user where the received messages are stored. A *mailbox* is part of a server hard drive, a special file with permission restrictions. Only the owner of the mailbox has access to it. The administrator has also created a queue (spool) to store messages waiting to be sent. A simple e-mail from Alice to Bob takes nine different steps, as shown in the figure. Alice and Bob use three different *agents*: a **user agent (UA),** a **message transfer agent (MTA),** and a **message access agent (MAA).** When Alice needs to send a message to Bob, she runs a UA program to prepare the message and send it to her mail server. The mail server at her site uses a queue to store messages waiting to be sent. The message, however, needs to be sent through the Internet from Alice's site to Bob's site using an MTA. Here two message transfer agents are needed: one client and one server. Like most client-server programs on the Internet, the server needs to run all the time because it does not know when a client will ask for a connection. The client, on the other hand, can be triggered by the system when there is a message in the queue to be sent. The user agent at the Bob site allows Bob to read the received message. Bob later uses an MAA	3+6=9	15

	client toretrieve the message from an MAA server running on the second server. (3 Marks for fig+6 Marks for explaining steps)		
X b	**TELNET** One of the original remote logging protocols is **TELNET**, which is an abbreviationfor *TErminaL NETwork*. Although TELNET requires a logging name and password, it is vulnerable to hacking because it sends all data including the password in plaintext (not encrypted). A hacker can eavesdrop and obtain the logging name and password. **Local versus Remote Logging** When a user logs into a local system, it is called *local logging*. As a user types at a terminal or at a workstation running a terminal emulator, the keystrokes are accepted by the terminal driver. The terminal driver passes the characters to the operating system. The operating system, in turn, interprets the combination of characters and invokes the desired application program or utility. However, when a user wants to access an application program or utility located on a remote machine, she performs *remote logging*. Here the TELNET client and server programs come into use. The user sends the keystrokes to the terminal driver where the local operating system accepts the characters but does not interpret them. The characters are sent to the TELNET client, which transforms the characters into a universal character set called *Network Virtual Terminal* (NVT) characters and delivers them to the local TCP/IP stack.  (3 marks for fig+3 marks for explanation)	3+3=6	