

Fifth Semester B.E./B.Tech. Degree Examination, Dec.2024/Jan.2025
Computer Networks

Max. Marks: 100

*Note: 1. Answer any FIVE full questions, choosing ONE full question from each module.
 2. M : Marks , L: Bloom's level , C: Course outcomes.*

Module – 1				M	L	C
Q.1	a.	What is data communication? List and explain characteristics and components of communication model.	06	L1	CO1	
	b.	Define switching. Explain Circuit Switched Network and Packet Switched Network.	06	L2	CO1	
	c.	With neat sketch, explain different layers of TCP/IP protocol suite.	08	L2	CO1	
OR						
Q.2	a.	What are guided transmission media? Explain twisted pair cable in detail.	06	L1	CO1	
	b.	What is Virtual Circuit Network (VCN)? With neat diagram, explain three phases involved in VCN.	08	L1	CO1	
	c.	Write a note on Encapsulation and decapsulation at Source Host for TCP/IP protocol suite.	06	L2	CO1	
Module – 2						
Q.3	a.	Define Redundancy. Explain CRC encoder and CRC decoder operation with block diagram.	08	L2	CO2	
	b.	Distinguish between Flow Control and Error Control. Explain Stop and Wait Protocol.	08	L2	CO2	
	c.	List and explain Control Fields of I-frames, S-frames and U-frames.	04	L2	CO2	
OR						
Q.4	a.	What is Hamming distance? With example, explain Parity Check Code.	06	L1	CO2	
	b.	Define Framing. Explain character oriented framing and bit-oriented framing.	06	L1	CO2	
	c.	With flow diagram, explain CSMA/CA.	08	L2	CO2	
Module – 3						
Q.5	a.	Explain virtual-circuit approach to route the packets in packet-switched network.	10	L2	CO3	
	b.	Illustrate the working of OSPF and BGP.	10	L3	CO3	
OR						
Q.6	a.	Explain IPv6 datagram format.	10	L2	CO3	
	b.	Write an Dijkstra's algorithm to compute shortest path through graph.	06	L1	CO3	
	c.	Write a note on Routing Information Protocol (RIP) algorithm.	04	L2	CO3	
Module – 4						
Q.7	a.	Explain Go-Back-N protocol working.	10	L2	CO4	
	b.	With neat sketch, explain three-way handshaking of TCP connection establishment.	10	L2	CO4	

OR

Q.8	a.	With an outline, explain selective repeat protocol.	10	L2	CO4
	b.	List and explain various services provided by User Datagram Protocol (UDP).	10	L2	CO4

Module – 5

Q.9	a.	Briefly explain Secure Shell (SSH).	10	L2	CO4
	b.	Write a note on Request message and response message formats of HTTP.	10	L2	CO4

OR

Q.10	a.	With neat diagram, explain the basic model of FTP.	04	L2	CO4
	b.	Describe the architecture of electronic mail (e-mail).	06	L3	CO4
	c.	Briefly explain Recursive Resolution and Iterative Resolution in DNS.	10	L2	CO4

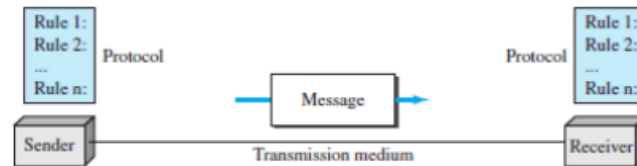
CMRIT LIBRARY
BANGALORE - 560 037



CMR INSTITUTE OF TECHNOLOGY
ACADEMIC YEAR <2024-25>
DEPARTMENT OF CSE
VTU SOLUTION
BCS502-COMPUTER NETWORKS

MODULE - 1					
1	a.	<p>What is data communication? List and explain characteristics AND components of the communication model.</p> <p>SOLUTION:</p> <h3>1.1 DATA COMMUNICATIONS</h3> <p>When we communicate, we are sharing information. This sharing can be local or remote. Between individuals, local communication usually occurs face to face, while remote communication takes place over distance. The term telecommunication, which includes telephony, telegraphy, and television, means communication at a distance (<i>tele</i> is Greek for “far”). The word data refers to information presented in whatever form is agreed upon by the parties creating and using the data.</p> <p>Data communications are the exchange of data between two devices via some form of transmission medium such as a wire cable. For data communications to occur, the communicating devices must be part of a communication system made up of a combination of hardware (physical equipment) and software (programs). The effectiveness of a data communications system depends on four fundamental characteristics: delivery, accuracy, timeliness, and jitter.</p> <ol style="list-style-type: none">1. Delivery. The system must deliver data to the correct destination. Data must be received by the intended device or user and only by that device or user.2. Accuracy. The system must deliver the data accurately. Data that have been altered in transmission and left uncorrected are unusable.3. Timeliness. The system must deliver data in a timely manner. Data delivered late are useless. In the case of video and audio, timely delivery means delivering data as they are produced, in the same order that they are produced, and without significant delay. This kind of delivery is called <i>real-time</i> transmission.4. Jitter. Jitter refers to the variation in the packet arrival time. It is the uneven delay in the delivery of audio or video packets. For example, let us assume that video packets are sent every 30 ms. If some of the packets arrive with 30-ms delay and others with 40-ms delay, an uneven quality in the video is the result.	06	L1	CO1

1.1 Five components of data communication:



1. **Message.** The **message** is the information (data) to be communicated. Popular forms of information include text, numbers, pictures, audio, and video.
2. **Sender.** The **sender** is the device that sends the data message. It can be a computer, workstation, telephone handset, video camera, and so on.
2. **Receiver.** The **receiver** is the device that receives the message. It can be a computer, workstation, telephone handset, television, and so on.
3. **Transmission medium.** The **transmission medium** is the physical path by which a message travels from sender to receiver. Some examples of transmission media include twisted-pair wire, coaxial cable, fiber-optic cable, and radio waves.
4. **Protocol.** A protocol is a set of rules that govern data communications. It represents an agreement between the communicating devices.

- b. Define switching. Explain Circuit Switched Network and Packet Switched Network.

06

L2

CO1

SOLUTION

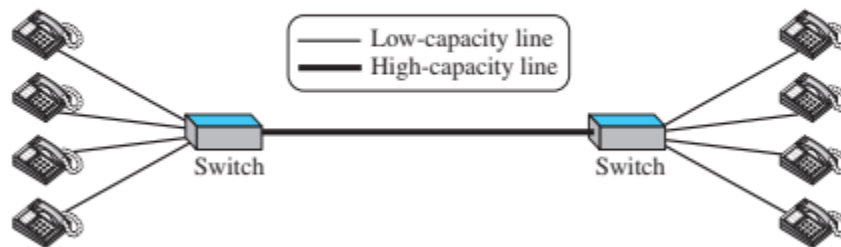
1.3.3 Switching

An internet is a **switched network** in which a switch connects at least two links together. A switch needs to forward data from a network to another network when required. The two most common types of switched networks are circuit-switched and packet-switched networks. We discuss both next.

Circuit-Switched Network

In a **circuit-switched network**, a dedicated connection, called a circuit, is always available between the two end systems; the switch can only make it active or inactive. Figure 1.13 shows a very simple switched network that connects four telephones to each end. We have used telephone sets instead of computers as an end system because circuit switching was very common in telephone networks in the past, although part of the telephone network today is a packet-switched network.

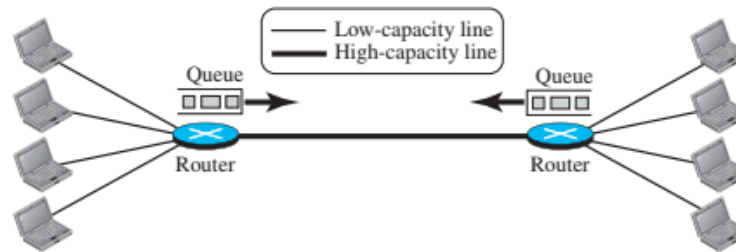
Figure 1.13 A circuit-switched network



Packet-Switched Network

In a computer network, the communication between the two ends is done in blocks of data called **packets**. In other words, instead of the continuous communication we see between two telephone sets when they are being used, we see the exchange of individual data packets between the two computers. This allows us to make the switches function for both storing and forwarding because a packet is an independent entity that can be stored and sent later. Figure 1.14 shows a small packet-switched network that connects four computers at one site to four computers at the other site.

Figure 1.14 A packet-switched network



A router in a packet-switched network has a queue that can store and forward the packet. Now assume that the capacity of the thick line is only twice the capacity of the data line connecting the computers to the routers. If only two computers (one at each site) need to communicate with each other, there is no waiting for the packets. However, if packets arrive at one router when the thick line is already working at its full capacity, the packets should be stored and forwarded in the order they arrived. The two simple examples show that a packet-switched network is more efficient than a circuit-switched network, but the packets may encounter some delays.

- c. With neat sketch, explain different layers of TCP/IP protocol suite.

SOLUTION

2.2 TCP/IP PROTOCOL SUITE

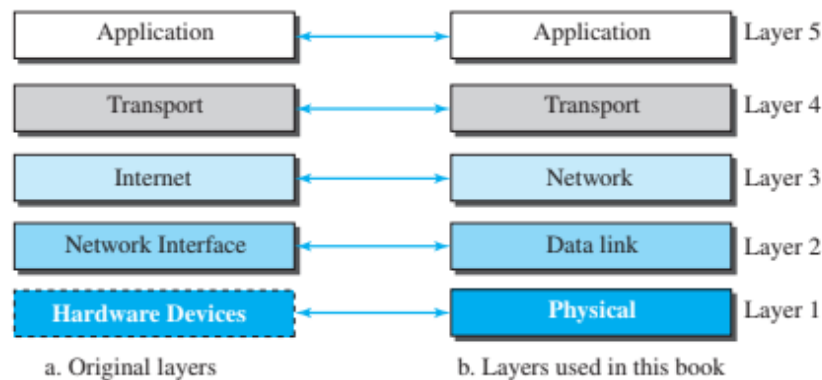
Now that we know about the concept of protocol layering and the logical communication between layers in our second scenario, we can introduce the TCP/IP (Transmission Control Protocol/Internet Protocol). TCP/IP is a protocol suite (a set of protocols organized in different layers) used in the Internet today. It is a hierarchical protocol made up of interactive modules, each of which provides a specific functionality. The term *hierarchical* means that each upper level protocol is supported by the services provided by one or more lower level protocols. The original TCP/IP protocol suite was defined as four software layers built upon the hardware. Today, however, TCP/IP is thought of as a five-layer model. Figure 2.4 shows both configurations.

08

L2

CO1

Figure 2.4 *Layers in the TCP/IP protocol suite*



2.2.3 Description of Each Layer

After understanding the concept of logical communication, we are ready to briefly discuss the duty of each layer. Our discussion in this chapter will be very brief, but we come back to the duty of each layer in next five parts of the book.

Physical Layer

We can say that the physical layer is responsible for carrying individual bits in a frame across the link. Although the physical layer is the lowest level in the TCP/IP protocol suite, the communication between two devices at the physical layer is still a logical communication because there is another, hidden layer, the transmission media, under the physical layer. Two devices are connected by a transmission medium (cable or air). We need to know that the transmission medium does not carry bits; it carries electrical or optical signals. So the bits received in a frame from the data-link layer are transformed and sent through the transmission media, but we can think that the logical unit between two physical layers in two devices is a *bit*. There are several protocols that transform a bit to a signal. We discuss them in Part II when we discuss the physical layer and the transmission media.

Data-link Layer

We have seen that an internet is made up of several links (LANs and WANs) connected by routers. There may be several overlapping sets of links that a datagram can travel from the host to the destination. The routers are responsible for choosing the *best* links. However, when the next link to travel is determined by the router, the data-link layer is responsible for taking the datagram and moving it across the link. The link can be a wired LAN with a link-layer switch, a wireless LAN, a wired WAN, or a wireless WAN. We can also have different protocols used with any link type. In each case, the data-link layer is responsible for moving the packet through the link.

TCP/IP does not define any specific protocol for the data-link layer. It supports all the standard and proprietary protocols. Any protocol that can take the datagram and carry it through the link suffices for the network layer. The data-link layer takes a datagram and encapsulates it in a packet called a *frame*.

Network Layer

The network layer is responsible for creating a connection between the source computer and the destination computer. The communication at the network layer is host-to-host. However, since there can be several routers from the source to the destination, the routers in the path are responsible for choosing the best route for each packet. We can say that the network layer is responsible for host-to-host communication and routing the packet through possible routes. Again, we may ask ourselves why we need the network layer. We could have added the routing duty to the transport layer and dropped this layer. One reason, as we said before, is the separation of different tasks between different layers. The second reason is that the routers do not need the application and transport layers. Separating the tasks allows us to use fewer protocols on the routers.

The network layer in the Internet includes the main protocol, Internet Protocol (IP), that defines the format of the packet, called a datagram at the network layer. IP also defines the format and the structure of addresses used in this layer. IP is also responsible for routing a packet from its source to its destination, which is achieved by each router forwarding the datagram to the next router in its path.


Transport Layer

The logical connection at the transport layer is also end-to-end. The transport layer at the source host gets the message from the application layer, encapsulates it in a transport-layer packet (called a *segment* or a *user datagram* in different protocols) and sends it, through the logical (imaginary) connection, to the transport layer at the destination host. In other words, the transport layer is responsible for giving services to the application layer: to get a message from an application program running on the source host and deliver it to the corresponding application program on the destination host. We may ask why we need an end-to-end transport layer when we already have an end-to-end application layer. The reason is the separation of tasks and duties, which we discussed earlier. The transport layer should be independent of the application layer. In addition, we will see that we have more than one protocol in the transport layer, which means that each application program can use the protocol that best matches its requirement.

Application Layer

As Figure 2.6 shows, the logical connection between the two application layers is end-to-end. The two application layers exchange *messages* between each other as though there were a bridge between the two layers. However, we should know that the communication is done through all the layers.

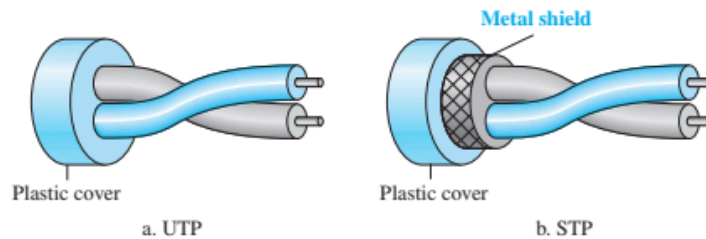
Communication at the application layer is between two *processes* (two programs running at this layer). To communicate, a process sends a request to the other process and receives a response. Process-to-process communication is the duty of the application layer. The application layer in the Internet includes many predefined protocols, but

		<p>a user can also create a pair of processes to be run at the two hosts. In Chapter 25, we explore this situation.</p> <p>The Hypertext Transfer Protocol (HTTP) is a vehicle for accessing the World Wide Web (WWW). The Simple Mail Transfer Protocol (SMTP) is the main protocol used in electronic mail (e-mail) service. The File Transfer Protocol (FTP) is used for transferring files from one host to another. The Terminal Network (TELNET) and Secure Shell (SSH) are used for accessing a site remotely. The Simple Network Management Protocol (SNMP) is used by an administrator to manage the Internet at global and local levels. The Domain Name System (DNS) is used by other protocols to find the network-layer address of a computer. The Internet Group Management Protocol</p>			
OR					
2	a.	<p>What are guided transmission media? Explain twisted pair cable in detail.</p> <p>SOLUTION</p> <h2>7.2 GUIDED MEDIA</h2> <p>Guided media, which are those that provide a conduit from one device to another, include twisted-pair cable, coaxial cable, and fiber-optic cable. A signal traveling along any of these media is directed and contained by the physical limits of the medium. Twisted-pair and coaxial cable use metallic (copper) conductors that accept and transport signals in the form of electric current. Optical fiber is a cable that accepts and transports signals in the form of light.</p> <h3>7.2.1 Twisted-Pair Cable</h3> <p>A twisted pair consists of two conductors (normally copper), each with its own plastic insulation, twisted together, as shown in Figure 7.3.</p> <p>Figure 7.3 <i>Twisted-pair cable</i></p>  <p>One of the wires is used to carry signals to the receiver, and the other is used only as a ground reference. The receiver uses the difference between the two.</p> <p>In addition to the signal sent by the sender on one of the wires, interference (noise) and crosstalk may affect both wires and create unwanted signals.</p> <p>If the two wires are parallel, the effect of these unwanted signals is not the same in both wires because they are at different locations relative to the noise or crosstalk sources (e.g., one is closer and the other is farther). This results in a difference at the receiver.</p>	06	L1	CO1

Unshielded Versus Shielded Twisted-Pair Cable

The most common twisted-pair cable used in communications is referred to as **unshielded twisted-pair (UTP)**. IBM has also produced a version of twisted-pair cable for its use, called **shielded twisted-pair (STP)**. STP cable has a metal foil or braided-mesh covering that encases each pair of insulated conductors. Although metal casing improves the quality of cable by preventing the penetration of noise or crosstalk, it is bulkier and more expensive. Figure 7.4 shows the difference between UTP and STP. Our discussion focuses primarily on UTP because STP is seldom used outside of IBM.

Figure 7.4 UTP and STP cables



Categories

The Electronic Industries Association (EIA) has developed standards to classify unshielded twisted-pair cable into seven categories. Categories are determined by cable quality, with 1 as the lowest and 7 as the highest. Each EIA category is suitable for specific uses. Table 7.1 shows these categories.

Table 7.1 Categories of unshielded twisted-pair cables

Category	Specification	Data Rate (Mbps)	Use
1	Unshielded twisted-pair used in telephone	< 0.1	Telephone
2	Unshielded twisted-pair originally used in T lines	2	T-1 lines
3	Improved CAT 2 used in LANs	10	LANs
4	Improved CAT 3 used in Token Ring networks	20	LANs
5	Cable wire is normally 24 AWG with a jacket and outside sheath	100	LANs

- b. What is Virtual Circuit Network (VCN)? With neat diagram, explain three phases involved in VCN.

SOLUTION

08

L1

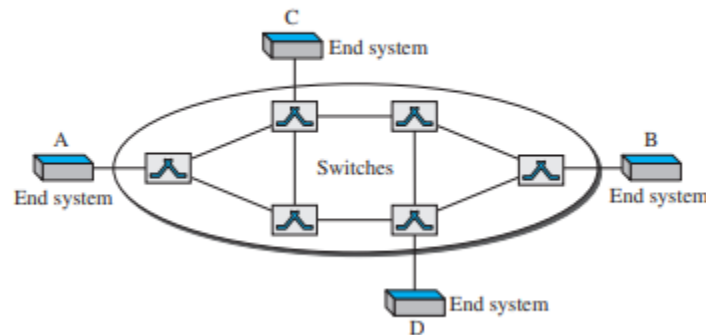
CO1

8.3.2 Virtual-Circuit Networks

A **virtual-circuit network** is a cross between a circuit-switched network and a datagram network. It has some characteristics of both.

1. As in a circuit-switched network, there are setup and teardown phases in addition to the data transfer phase.
2. Resources can be allocated during the setup phase, as in a circuit-switched network, or on demand, as in a datagram network.
3. As in a datagram network, data are packetized and each packet carries an address in the header. However, the address in the header has local jurisdiction (it defines what the next switch should be and the channel on which the packet is being carried), not end-to-end jurisdiction. The reader may ask how the intermediate switches know where to send the packet if there is no final destination address carried by a packet. The answer will be clear when we discuss virtual-circuit identifiers in the next section.
4. As in a circuit-switched network, all packets follow the same path established during the connection.

Figure 8.10 Virtual-circuit network



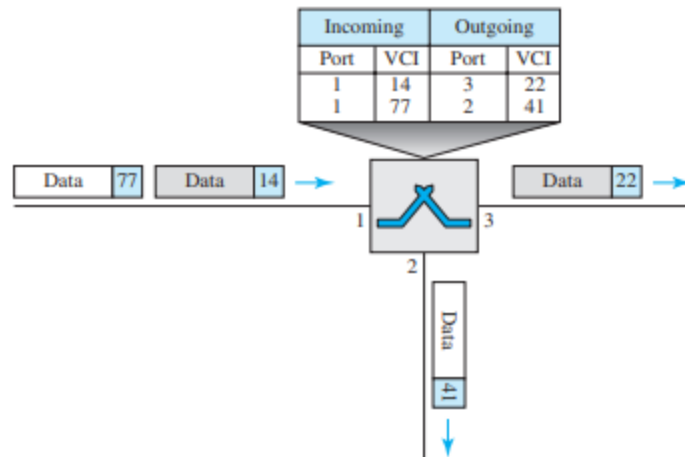
Three Phases

As in a circuit-switched network, a source and destination need to go through three phases in a virtual-circuit network: setup, data transfer, and teardown. In the setup phase, the source and destination use their global addresses to help switches make table entries for the connection. In the teardown phase, the source and destination inform the switches to delete the corresponding entry. Data transfer occurs between these two phases. We first discuss the data-transfer phase, which is more straightforward; we then talk about the setup and teardown phases.

Data-Transfer Phase

To transfer a frame from a source to its destination, all switches need to have a table entry for this virtual circuit. The table, in its simplest form, has four columns. This means that the switch holds four pieces of information for each virtual circuit that is already set up. We show later how the switches make their table entries, but for the moment we assume that each switch has a table with entries for all active virtual circuits. Figure 8.12 shows such a switch and its corresponding table.

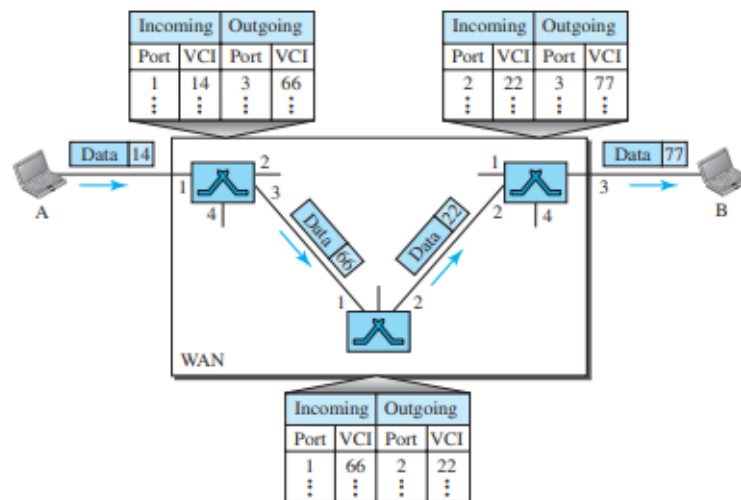
Figure 8.12 Switch and tables in a virtual-circuit network



Setup Phase

In the setup phase, a switch creates an entry for a virtual circuit. For example, suppose source A needs to create a virtual circuit to B. Two steps are required: the setup request and the acknowledgment.

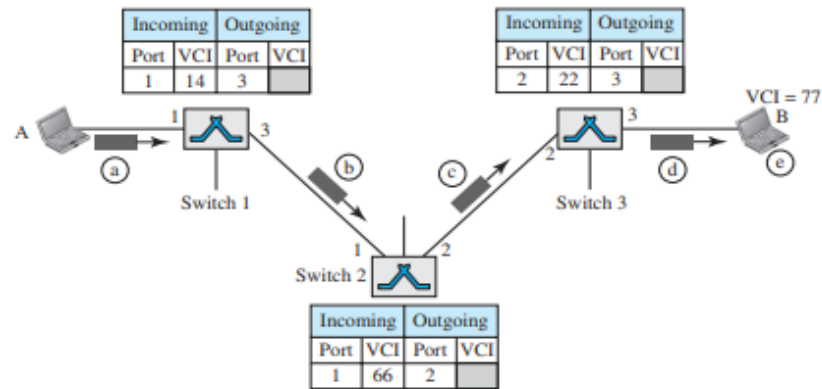
Figure 8.13 Source-to-destination data transfer in a virtual-circuit network



Setup Request

A setup request frame is sent from the source to the destination. Figure 8.14 shows the process.

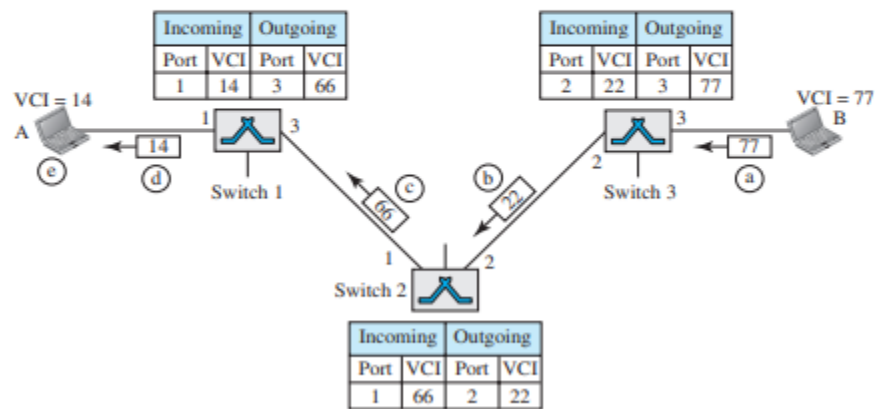
Figure 8.14 Setup request in a virtual-circuit network



Acknowledgment

A special frame, called the *acknowledgment frame*, completes the entries in the switching tables. Figure 8.15 shows the process.

Figure 8.15 Setup acknowledgment in a virtual-circuit network



Teardown Phase

In this phase, source A, after sending all frames to B, sends a special frame called a *teardown request*. Destination B responds with a *teardown confirmation* frame. All switches delete the corresponding entry from their tables.

- c. Write a note on Encapsulation and decapsulation at Source Host for TCP/IP protocol suite.

SOLUTION

06

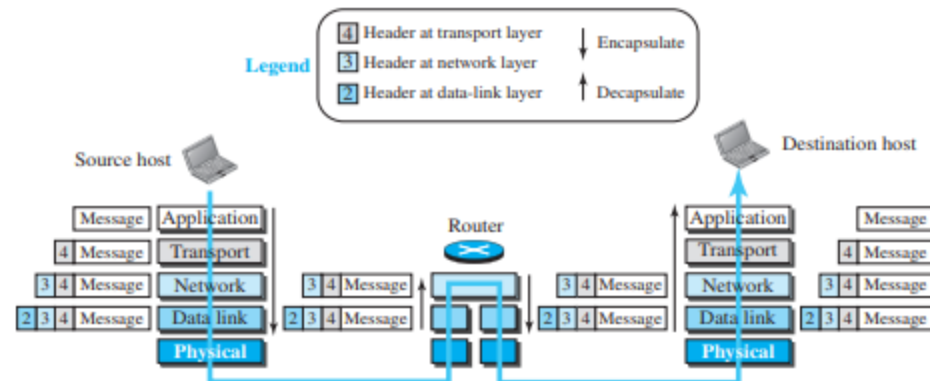
L2

CO1

2.2.4 Encapsulation and Decapsulation

One of the important concepts in protocol layering in the Internet is encapsulation/decapsulation. Figure 2.8 shows this concept for the small internet in Figure 2.5.

Figure 2.8 Encapsulation/Decapsulation



Encapsulation at the Source Host

At the source, we have only encapsulation.

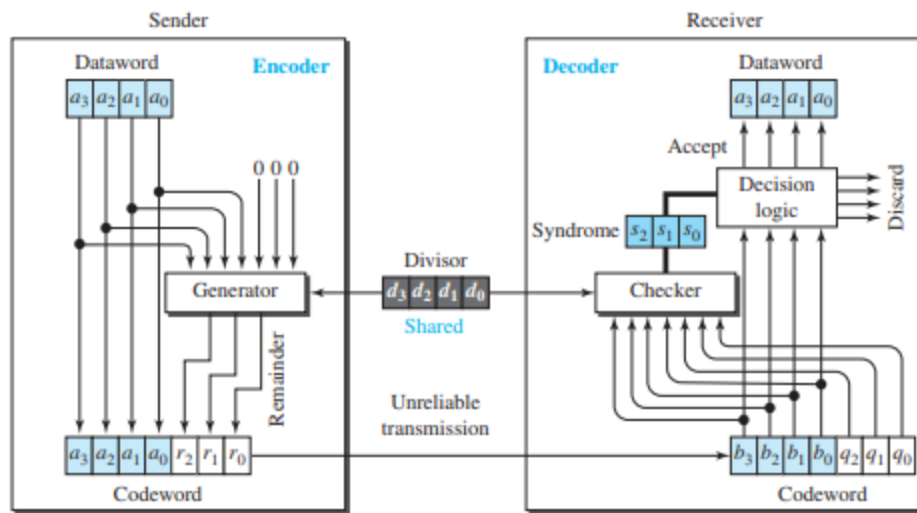
1. At the application layer, the data to be exchanged is referred to as a *message*. A message normally does not contain any header or trailer, but if it does, we refer to the whole as the message. The message is passed to the transport layer.
2. The transport layer takes the message as the payload, the load that the transport layer should take care of. It adds the transport layer header to the payload, which contains the identifiers of the source and destination application programs that

want to communicate plus some more information that is needed for the end-to-end delivery of the message, such as information needed for flow, error control, or congestion control. The result is the transport-layer packet, which is called the *segment* (in TCP) and the *user datagram* (in UDP). The transport layer then passes the packet to the network layer.

3. The network layer takes the transport-layer packet as data or payload and adds its own header to the payload. The header contains the addresses of the source and destination hosts and some more information used for error checking of the header, fragmentation information, and so on. The result is the network-layer packet, called a *datagram*. The network layer then passes the packet to the data-link layer.
4. The data-link layer takes the network-layer packet as data or payload and adds its own header, which contains the link-layer addresses of the host or the next hop (the router). The result is the link-layer packet, which is called a *frame*. The frame is passed to the physical layer for transmission.

		<p><i>Decapsulation and Encapsulation at the Router</i></p> <p>At the router, we have both decapsulation and encapsulation because the router is connected to two or more links.</p> <ol style="list-style-type: none"> 1. After the set of bits are delivered to the data-link layer, this layer decapsulates the datagram from the frame and passes it to the network layer. 2. The network layer only inspects the source and destination addresses in the datagram header and consults its forwarding table to find the next hop to which the datagram is to be delivered. The contents of the datagram should not be changed by the network layer in the router unless there is a need to fragment the datagram if it is too big to be passed through the next link. The datagram is then passed to the data-link layer of the next link. 3. The data-link layer of the next link encapsulates the datagram in a frame and passes it to the physical layer for transmission. <p><i>Decapsulation at the Destination Host</i></p> <p>At the destination host, each layer only decapsulates the packet received, removes the payload, and delivers the payload to the next-higher layer protocol until the message reaches the application layer. It is necessary to say that decapsulation in the host involves error checking.</p>			
MODULE -2					
3	a.	<p>Define Redundancy. Explain CRC encoder and CRC decoder operation with block diagram.</p> <p>SOLUTION</p> <p>Redundancy: Redundancy in CRC refers to the extra bits (also called redundant bits or check bits) that are added to the original data to enable error detection. CRC works by appending a checksum (redundant bits) to the transmitted message, which allows the receiver to verify data integrity and detect errors during transmission.</p> <p>10.3.1 Cyclic Redundancy Check</p> <p>We can create cyclic codes to correct errors. However, the theoretical background required is beyond the scope of this book. In this section, we simply discuss a subset of</p>	08	L2	CO2

Figure 10.5 CRC encoder and decoder



In the encoder, the dataword has k bits (4 here); the codeword has n bits (7 here). The size of the dataword is augmented by adding $n - k$ (3 here) 0s to the right-hand side of the word. The n -bit result is fed into the generator. The generator uses a divisor of size $n - k + 1$ (4 here), predefined and agreed upon. The generator divides the augmented dataword by the divisor (modulo-2 division). The quotient of the division is discarded; the remainder ($r_2r_1r_0$) is appended to the dataword to create the codeword.

The decoder receives the codeword (possibly corrupted in transition). A copy of all n bits is fed to the checker, which is a replica of the generator. The remainder produced

by the checker is a syndrome of $n - k$ (3 here) bits, which is fed to the decision logic analyzer. The analyzer has a simple function. If the syndrome bits are all 0s, the 4 leftmost bits of the codeword are accepted as the dataword (interpreted as no error); otherwise, the 4 bits are discarded (error).

Encoder

Let us take a closer look at the encoder. The encoder takes a dataword and augments it with $n - k$ number of 0s. It then divides the augmented dataword by the divisor, as shown in Figure 10.6.

The process of modulo-2 binary division is the same as the familiar division process we use for decimal numbers. However, addition and subtraction in this case are the same; we use the XOR operation to do both.

As in decimal division, the process is done step by step. In each step, a copy of the divisor is XORed with the 4 bits of the dividend. The result of the XOR operation (remainder) is 3 bits (in this case), which is used for the next step after 1 extra bit is pulled down to make it 4 bits long. There is one important point we need to remember in this type of division. If the leftmost bit of the dividend (or the part used in each step) is 0, the step cannot use the regular divisor; we need to use an all-0s divisor.

When there are no bits left to pull down, we have a result. The 3-bit remainder forms the **check bits** (r_2 , r_1 , and r_0). They are appended to the dataword to create the codeword.

		<p><i>Decoder</i></p> <p>The codeword can change during transmission. The decoder does the same division process as the encoder. The remainder of the division is the syndrome. If the syndrome is all 0s, there is no error with a high probability; the dataword is separated from the received codeword and accepted. Otherwise, everything is discarded. Figure 10.7 shows two cases: The left-hand figure shows the value of the syndrome when no error has occurred; the syndrome is 000. The right-hand part of the figure shows the case in which there is a single error. The syndrome is not all 0s (it is 011).</p>			
	b.	<p>Distinguish between Flow Control and Error Control. Explain Stop and Wait Protocol.</p> <p>SOLUTION</p> <p>Flow Control: Manages the rate at which data is transmitted between sender and receiver to prevent data loss due to buffer overflow. Ensures that the sender does not overwhelm the receiver with data. Does not deal with error correction; only controls data transmission speed. Works at the Data Link Layer and Transport Layer.</p> <p>Error Control: Detects and corrects errors in transmitted data to ensure accurate communication. Ensures that the received data is accurate and free from errors. Detects and corrects errors in the received data. Works at the Data Link Layer and Transport Layer.</p> <p>11.2.2 Stop-and-Wait Protocol</p> <p>Our second protocol is called the Stop-and-Wait protocol, which uses both flow and error control. We show a primitive version of this protocol here, but we discuss the more sophisticated version in Chapter 23 when we have learned about sliding windows. In this protocol, the sender sends one frame at a time and waits for an acknowledgment before sending the next one. To detect corrupted frames, we need to add a CRC (see Chapter 10) to each data frame. When a frame arrives at the receiver site, it is checked. If its CRC is incorrect, the frame is corrupted and silently discarded. The silence of the receiver is a signal for the sender that a frame was either corrupted or lost. Every time the sender sends a frame, it starts a timer. If an acknowledgment arrives before the timer expires, the timer is stopped and the sender sends the next frame (if it has one to send). If the timer expires, the sender resends the previous frame, assuming that the frame was either lost or corrupted. This means that the sender needs to keep a copy of the frame until its acknowledgment arrives. When the corresponding acknowledgment arrives, the sender discards the copy and sends the next frame if it is ready. Figure 11.10 shows the outline for the Stop-and-Wait protocol. Note that only one frame and one acknowledgment can be in the channels at any time.</p>	08	L2	CO2

Figure 11.10 Stop-and-Wait protocol

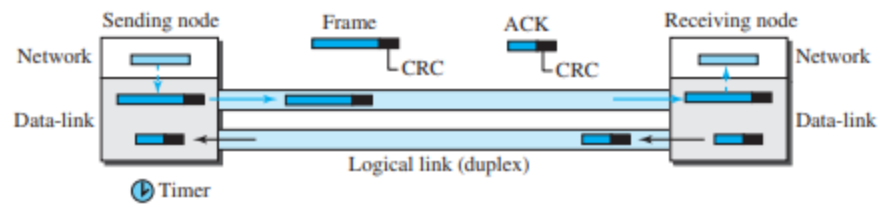
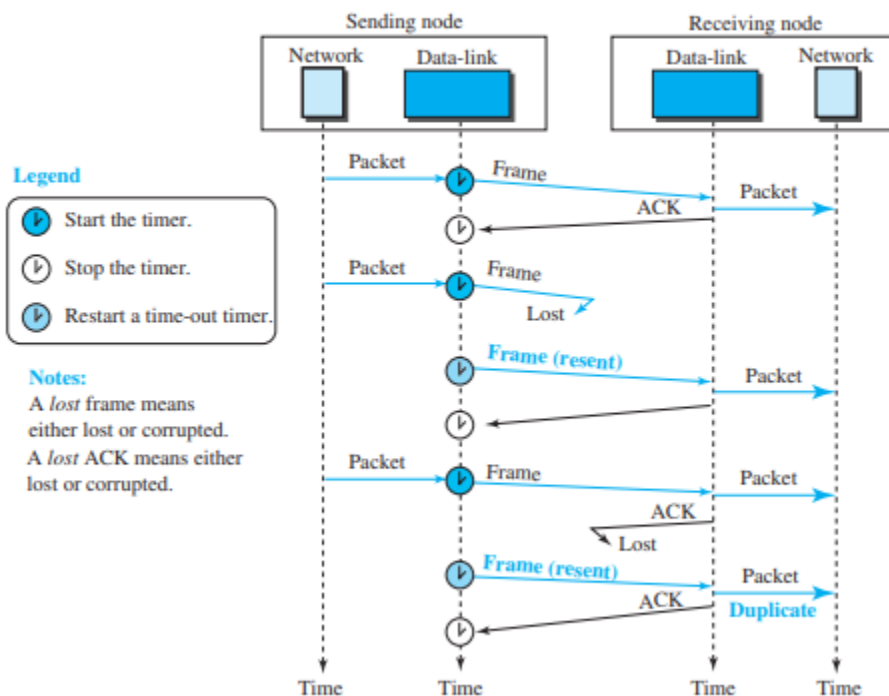


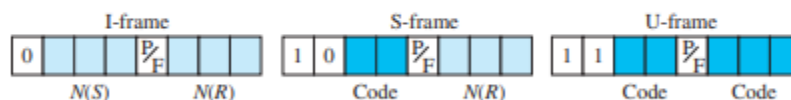
Figure 11.12 Flow diagram for Example 11.3



c. List and explain Control Fields of I-frames, S-frames and U-frames.

SOLUTION

Figure 11.17 Control field format for the different frame types



		<p>Control Field for I-Frames</p> <p>I-frames are designed to carry user data from the network layer. In addition, they can include flow- and error-control information (piggybacking). The subfields in the control field are used to define these functions. The first bit defines the type. If the first bit of the control field is 0, this means the frame is an I-frame. The next 3 bits, called $N(S)$, define the sequence number of the frame. Note that with 3 bits, we can define a sequence number between 0 and 7. The last 3 bits, called $N(R)$, correspond to the acknowledgment number when piggybacking is used. The single bit between $N(S)$ and $N(R)$ is called the P/F bit. The P/F field is a single bit with a dual purpose. It has meaning only when it is set (bit = 1) and can mean poll or final. It means <i>poll</i> when the frame is sent by a primary station to a secondary (when the address field contains the address of the receiver). It means <i>final</i> when the frame is sent by a secondary to a primary (when the address field contains the address of the sender).</p> <p>Control Field for S-Frames</p> <p>Supervisory frames are used for flow and error control whenever piggybacking is either impossible or inappropriate. S-frames do not have information fields. If the first 2 bits of the control field are 10, this means the frame is an S-frame. The last 3 bits, called $N(R)$, correspond to the acknowledgment number (ACK) or negative acknowledgment number (NAK), depending on the type of S-frame. The 2 bits called <i>code</i> are used to define the type of S-frame itself. With 2 bits, we can have four types of S-frames, as described below:</p> <ul style="list-style-type: none"> ❑ Receive ready (RR). If the value of the code subfield is 00, it is an RR S-frame. This kind of frame acknowledges the receipt of a safe and sound frame or group of frames. In this case, the value of the $N(R)$ field defines the acknowledgment number. ❑ Receive not ready (RNR). If the value of the code subfield is 10, it is an RNR S-frame. This kind of frame is an RR frame with additional functions. It acknowledges the receipt of a frame or group of frames, and it announces that the receiver is busy and cannot receive more frames. It acts as a kind of congestion-control mechanism by asking the sender to slow down. The value of $N(R)$ is the acknowledgment number. ❑ Reject (REJ). If the value of the code subfield is 01, it is an REJ S-frame. This is a NAK frame, but not like the one used for Selective Repeat ARQ. It is a NAK that can be used in Go-Back-N ARQ to improve the efficiency of the process by informing the sender, before the sender timer expires, that the last frame is lost or damaged. The value of $N(R)$ is the negative acknowledgment number. ❑ Selective reject (SREJ). If the value of the code subfield is 11, it is an SREJ S-frame. This is a NAK frame used in Selective Repeat ARQ. Note that the HDLC Protocol uses the term <i>selective reject</i> instead of <i>selective repeat</i>. The value of $N(R)$ is the negative acknowledgment number. 		
--	--	-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	--	--

		<p>Control Field for U-Frames</p> <p>Unnumbered frames are used to exchange session management and control information between connected devices. Unlike S-frames, U-frames contain an information field, but one used for system management information, not user data. As with S-frames, however, much of the information carried by U-frames is contained in codes included in the control field. U-frame codes are divided into two sections: a 2-bit prefix before the P/F bit and a 3-bit suffix after the P/F bit. Together, these two segments (5 bits) can be used to create up to 32 different types of U-frames.</p> <p>Control Field for U-Frames</p> <p>Unnumbered frames are used to exchange session management and control information between connected devices. Unlike S-frames, U-frames contain an information field, but one used for system management information, not user data. As with S-frames, however, much of the information carried by U-frames is contained in codes included in the control field. U-frame codes are divided into two sections: a 2-bit prefix before the P/F bit and a 3-bit suffix after the P/F bit. Together, these two segments (5 bits) can be used to create up to 32 different types of U-frames.</p>			
OR					
4	a.	<p>What is Hamming distance? With example, explain Parity Check Code.</p> <p>SOLUTION</p> <p>Hamming Distance</p> <p>One of the central concepts in coding for error control is the idea of the Hamming distance. The Hamming distance between two words (of the same size) is the number of differences between the corresponding bits. We show the Hamming distance between two words x and y as $d(x, y)$. We may wonder why Hamming distance is important for error detection. The reason is that the Hamming distance between the received codeword and the sent codeword is the number of bits that are corrupted during transmission. For example, if the codeword 00000 is sent and 01101 is received, 3 bits are in error and the Hamming distance between the two is $d(00000, 01101) = 3$. In other words, if the Hamming distance between the sent and the received codeword is not zero, the codeword has been corrupted during transmission.</p> <p>The Hamming distance can easily be found if we apply the XOR operation (\oplus) on the two words and count the number of 1s in the result. Note that the Hamming distance is a value greater than or equal to zero.</p> <p>Parity-Check Code</p> <p>Perhaps the most familiar error-detecting code is the parity-check code. This code is a linear block code. In this code, a k-bit dataword is changed to an n-bit codeword where $n = k + 1$. The extra bit, called the <i>parity bit</i>, is selected to make the total number of 1s in the codeword even. Although some implementations specify an odd number of 1s, we discuss the even case. The minimum Hamming distance for this category is $d_{\min} = 2$, which means that the code is a single-bit error-detecting code. Our first code (Table 10.1) is a parity-check code ($k = 2$ and $n = 3$). The code in Table 10.2 is also a parity-check code with $k = 4$ and $n = 5$.</p>	06	L1	CO2

Table 10.2 Simple parity-check code $C(5, 4)$

Dataword	Codeword	Dataword	Codeword
0000	00000	1000	10001
0001	00011	1001	10010
0010	00101	1010	10100
0011	00110	1011	10111
0100	01001	1100	11000
0101	01010	1101	11011
0110	01100	1110	11101
0111	01111	1111	11110

- b. Define Framing. Explain character oriented framing and bit-oriented framing

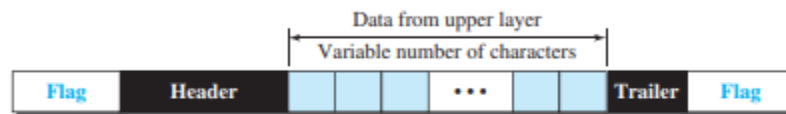
06 L1 CO2

SOLUTION

Character-Oriented Framing

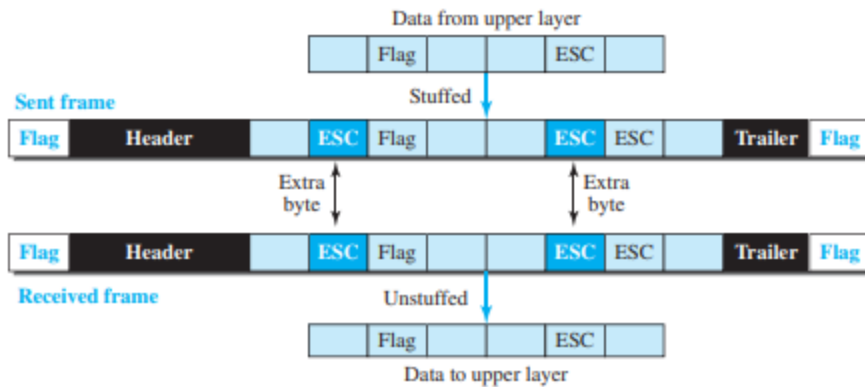
In *character-oriented (or byte-oriented) framing*, data to be carried are 8-bit characters from a coding system such as ASCII (see Appendix A). The header, which normally carries the source and destination addresses and other control information, and the trailer, which carries error detection redundant bits, are also multiples of 8 bits. To separate one frame from the next, an 8-bit (1-byte) **flag** is added at the beginning and the end of a frame. The flag, composed of protocol-dependent special characters, signals the start or end of a frame. Figure 11.1 shows the format of a frame in a character-oriented protocol.

Figure 11.1 A frame in a character-oriented protocol



Character-oriented framing was popular when only text was exchanged by the data-link layers. The flag could be selected to be any character not used for text communication. Now, however, we send other types of information such as graphs, audio, and video; any character used for the flag could also be part of the information. If this happens, the receiver, when it encounters this pattern in the middle of the data, thinks it has reached the end of the frame. To fix this problem, a byte-stuffing strategy was added to character-oriented framing. In **byte stuffing** (or character stuffing), a special byte is added to the data section of the frame when there is a character with the same pattern as the flag. The data section is stuffed with an extra byte. This byte is usually called the *escape character (ESC)* and has a predefined bit pattern. Whenever the receiver encounters the ESC character, it removes it from the data section and treats the next character as data, not as a delimiting flag. Figure 11.2 shows the situation.

Figure 11.2 Byte stuffing and unstuffing

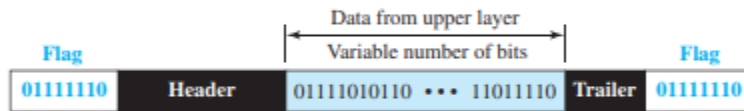


Byte stuffing is the process of adding one extra byte whenever there is a flag or escape character in the text.

Bit-Oriented Framing

In *bit-oriented framing*, the data section of a frame is a sequence of bits to be interpreted by the upper layer as text, graphic, audio, video, and so on. However, in addition to headers (and possible trailers), we still need a delimiter to separate one frame from the other. Most protocols use a special 8-bit pattern flag, 01111110, as the delimiter to define the beginning and the end of the frame, as shown in Figure 11.3.

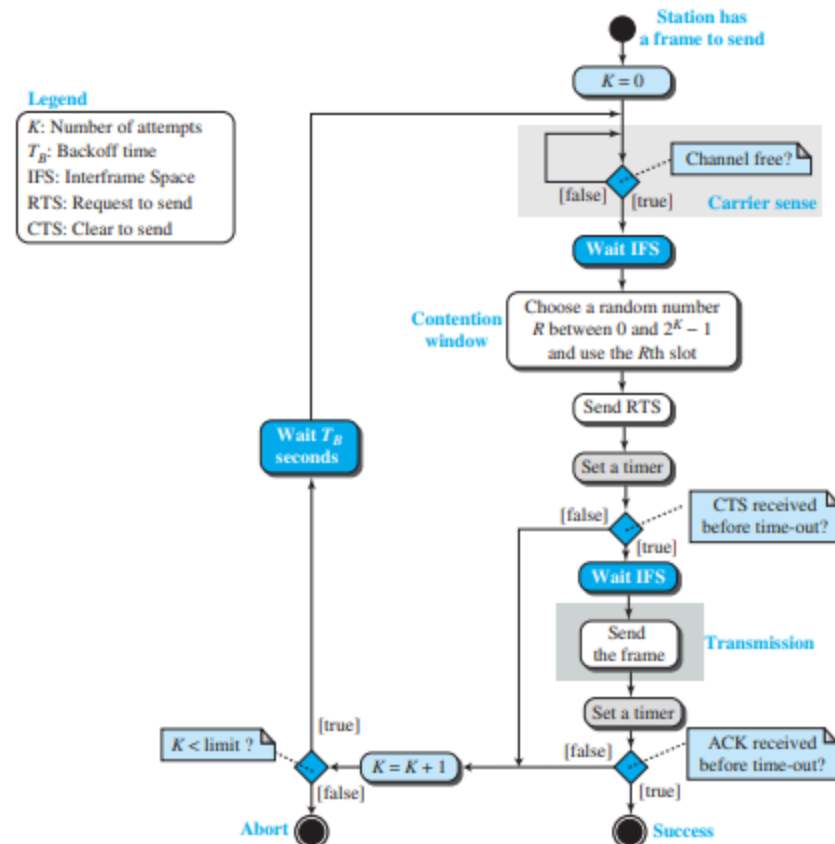
Figure 11.3 A frame in a bit-oriented protocol



This flag can create the same type of problem we saw in the character-oriented protocols. That is, if the flag pattern appears in the data, we need to somehow inform the receiver that this is not the end of the frame. We do this by stuffing 1 single bit (instead of 1 byte) to prevent the pattern from looking like a flag. The strategy is called **bit stuffing**. In bit stuffing, if a 0 and five consecutive 1 bits are encountered, an extra 0 is added. This extra stuffed bit is eventually removed from the data by the receiver. Note that the extra bit is added after one 0 followed by five 1s regardless of the value of the next bit. This guarantees that the flag field sequence does not inadvertently appear in the frame.

		<div><p>Figure 11.4 <i>Bit stuffing and unstuffing</i></p><p>The diagram illustrates the bit stuffing process. It starts with 'Data from upper layer' (0001111111001111101000). This data is 'Stuffed' with a zero to create a 'Frame sent'. The frame consists of a 'Flag' (00011111), a 'Header' (1011001111), the stuffed data (00011111110011111001000), a 'Trailer' (1001000), and another 'Flag' (00011111). The 'Frame received' is identical. Two arrows labeled 'Two extra bits' point to the stuffed data and the trailer. The frame is then 'Unstuffed' to retrieve the 'Data to upper layer' (0001111111001111101000).</p></div>			
	c.	<p>With flow diagram, explain CSMA/CA.</p> <p>SOLUTION</p> <p>12.1.4 CSMA/CA</p> <p>Carrier sense multiple access with collision avoidance (CSMA/CA) was invented for wireless networks. Collisions are avoided through the use of CSMA/CA's three strategies: the interframe space, the contention window, and acknowledgments, as shown in Figure 12.15. We discuss RTS and CTS frames later.</p>	08	L2	CO2

Figure 12.15 Flow diagram of CSMA/CA

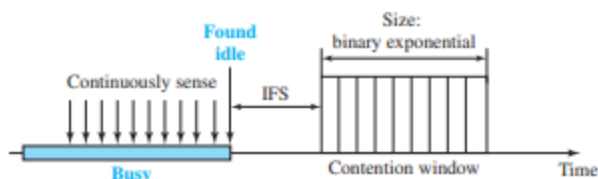


- ❑ **Interframe Space (IFS).** First, collisions are avoided by deferring transmission even if the channel is found idle. When an idle channel is found, the station does not send immediately. It waits for a period of time called the *interframe space* or *IFS*. Even though the channel may appear idle when it is sensed, a distant station may have already started transmitting. The distant station's signal has not yet reached this

station. The IFS time allows the front of the transmitted signal by the distant station to reach this station. After waiting an IFS time, if the channel is still idle, the station can send, but it still needs to wait a time equal to the contention window (described next). The IFS variable can also be used to prioritize stations or frame types. For example, a station that is assigned a shorter IFS has a higher priority.

- ❑ **Contention Window.** The **contention window** is an amount of time divided into slots. A station that is ready to send chooses a random number of slots as its wait time. The number of slots in the window changes according to the binary exponential backoff strategy. This means that it is set to one slot the first time and then doubles each time the station cannot detect an idle channel after the IFS time. This is very similar to the p -persistent method except that a random outcome defines the number of slots taken by the waiting station. One interesting point about the contention window is that the station needs to sense the channel after each time slot. However, if the station finds the channel busy, it does not restart the process; it just stops the timer and restarts it when the channel is sensed as idle. This gives priority to the station with the longest waiting time. See Figure 12.16.

Figure 12.16 Contention window



- ❑ **Acknowledgment.** With all these precautions, there still may be a collision resulting in destroyed data. In addition, the data may be corrupted during the transmission. The positive acknowledgment and the time-out timer can help guarantee that the receiver has received the frame.

Frame Exchange Time Line

Figure 12.17 shows the exchange of data and control frames in time.

1. Before sending a frame, the source station senses the medium by checking the energy level at the carrier frequency.
 - a. The channel uses a persistence strategy with backoff until the channel is idle.
 - b. After the station is found to be idle, the station waits for a period of time called the **DCF interframe space (DIFS)**; then the station sends a control frame called the **request to send (RTS)**.
2. After receiving the RTS and waiting a period of time called the **short interframe space (SIFS)**, the destination station sends a control frame, called the **clear to send (CTS)**, to the source station. This control frame indicates that the destination station is ready to receive data.
3. The source station sends data after waiting an amount of time equal to SIFS.
4. The destination station, after waiting an amount of time equal to SIFS, sends an acknowledgment to show that the frame has been received. Acknowledgment is needed in this protocol because the station does not have any means to check for the successful arrival of its data at the destination. On the other hand, the lack of collision in CSMA/CD is a kind of indication to the source that data have arrived.

MODULE - 03

5	a.	Explain virtual-circuit approach to route the packets in packet-switched network.	10	L2	CO3
		SOLUTION			

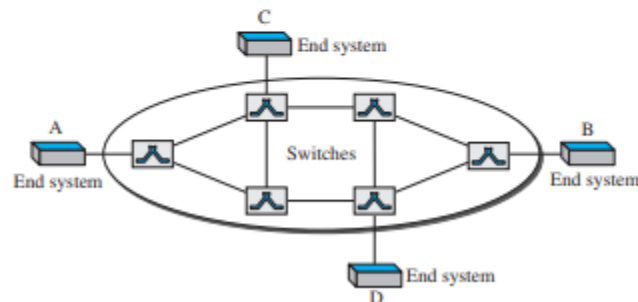
8.3.2 Virtual-Circuit Networks

A **virtual-circuit network** is a cross between a circuit-switched network and a datagram network. It has some characteristics of both.

1. As in a circuit-switched network, there are setup and teardown phases in addition to the data transfer phase.
2. Resources can be allocated during the setup phase, as in a circuit-switched network, or on demand, as in a datagram network.
3. As in a datagram network, data are packetized and each packet carries an address in the header. However, the address in the header has local jurisdiction (it defines what the next switch should be and the channel on which the packet is being carried), not end-to-end jurisdiction. The reader may ask how the intermediate switches know where to send the packet if there is no final destination address carried by a packet. The answer will be clear when we discuss virtual-circuit identifiers in the next section.
4. As in a circuit-switched network, all packets follow the same path established during the connection.
5. A virtual-circuit network is normally implemented in the data-link layer, while a circuit-switched network is implemented in the physical layer and a datagram network in the network layer. But this may change in the future.

Figure 8.10 is an example of a virtual-circuit network. The network has switches that allow traffic from sources to destinations. A source or destination can be a computer, packet switch, bridge, or any other device that connects other networks.

Figure 8.10 Virtual-circuit network



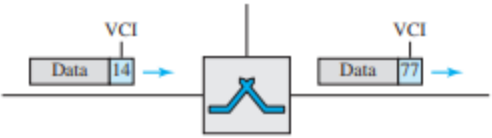
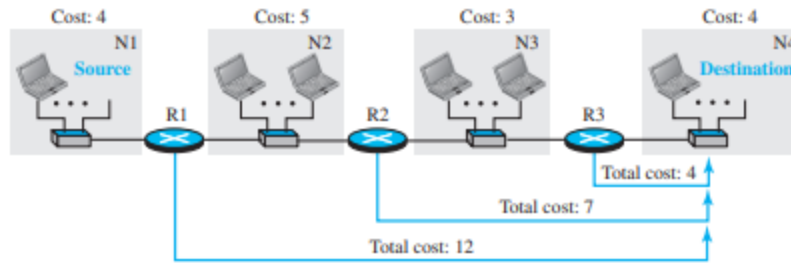
		<p>Addressing</p> <p>In a virtual-circuit network, two types of addressing are involved: global and local (virtual-circuit identifier).</p> <p>Global Addressing</p> <p>A source or a destination needs to have a global address—an address that can be unique in the scope of the network or internationally if the network is part of an international network. However, we will see that a global address in virtual-circuit networks is used only to create a virtual-circuit identifier, as discussed next.</p> <p>Virtual-Circuit Identifier</p> <p>The identifier that is actually used for data transfer is called the <i>virtual-circuit identifier (VCI)</i> or the <i>label</i>. A VCI, unlike a global address, is a small number that has only switch scope; it is used by a frame between two switches. When a frame arrives at a switch, it has a VCI; when it leaves, it has a different VCI. Figure 8.11 shows how the VCI in a data frame changes from one switch to another. Note that a VCI does not need to be a large number since each switch can use its own unique set of VCIs.</p> <hr/> <p>Figure 8.11 Virtual-circuit identifier</p> <hr/>  <hr/>			
	b.	<p>Illustrate the working of OSPF and BGP.</p> <p>SOLUTION</p> <p>20.3.3 Open Shortest Path First (OSPF)</p> <p>Open Shortest Path First (OSPF) is also an intradomain routing protocol like RIP, but it is based on the link-state routing protocol we described earlier in the chapter. OSPF is an <i>open</i> protocol, which means that the specification is a public document.</p> <p>Metric</p> <p>In OSPF, like RIP, the cost of reaching a destination from the host is calculated from the source router to the destination network. However, each link (network) can be assigned a weight based on the throughput, round-trip time, reliability, and so on. An administration can also decide to use the hop count as the cost. An interesting point about the cost in OSPF is that different service types (TOSs) can have different weights as the cost. Figure 20.19 shows the idea of the cost from a router to the destination host network. We can compare the figure with Figure 20.15 for the RIP.</p>	10	L3	CO3

Figure 20.19 Metric in OSPF



Forwarding Tables

Each OSPF router can create a forwarding table after finding the shortest-path tree between itself and the destination using Dijkstra's algorithm, described earlier in the chapter. Figure 20.20 shows the forwarding tables for the simple AS in Figure 20.19. Comparing the forwarding tables for the OSPF and RIP in the same AS, we find that the only difference is the cost values. In other words, if we use the hop count for OSPF, the tables will be exactly the same. The reason for this consistency is that both protocols use the shortest-path trees to define the best route from a source to a destination.

Areas

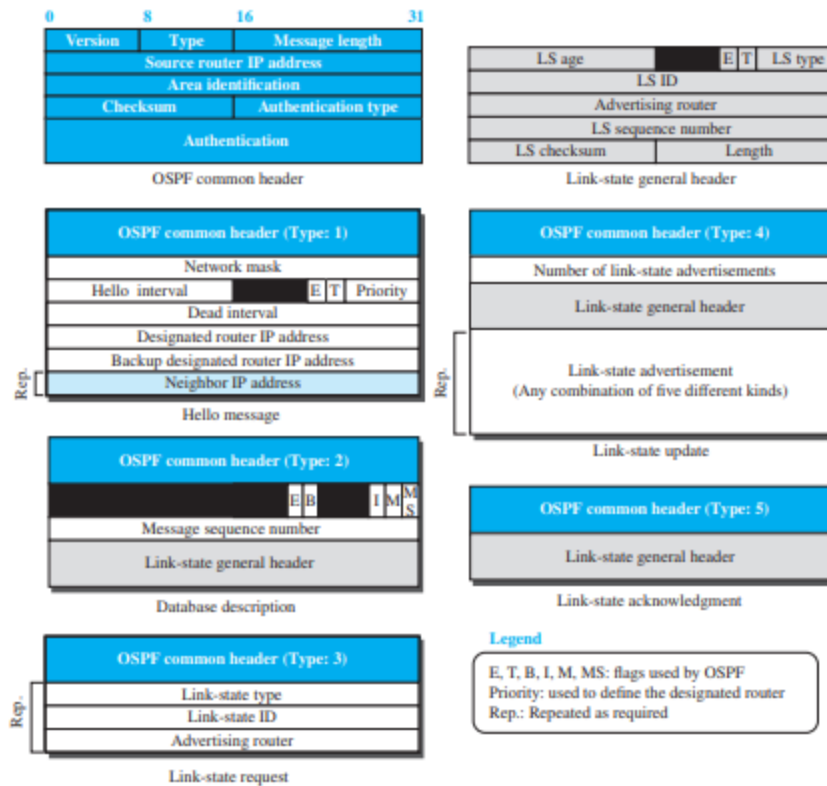
Compared with RIP, which is normally used in small ASs, OSPF was designed to be able to handle routing in a small or large autonomous system. However, the formation of shortest-path trees in OSPF requires that all routers flood the whole AS with their LSPs to create the global LSDB. Although this may not create a problem in a small AS, it may have created a huge volume of traffic in a large AS. To prevent this, the AS needs to be divided into small sections called *areas*. Each area acts as a small independent domain for flooding LSPs. In other words, OSPF uses another level of hierarchy in routing: the first level is the autonomous system, the second is the area.

Figure 20.20 Forwarding tables in OSPF

Forwarding table for R1			Forwarding table for R2			Forwarding table for R3		
Destination network	Next router	Cost	Destination network	Next router	Cost	Destination network	Next router	Cost
N1	—	4	N1	R1	9	N1	R2	12
N2	—	5	N2	—	5	N2	R2	8
N3	R2	8	N3	—	3	N3	—	3
N4	R2	12	N4	R3	7	N4	—	4

However, each router in an area needs to know the information about the link states not only in its area but also in other areas. For this reason, one of the areas in the AS is designated as the *backbone area*, responsible for gluing the areas together. The routers in the backbone area are responsible for passing the information collected by each area to all other areas. In this way, a router in an area can receive all LSPs generated in other areas. For the purpose of communication, each area has an area identification. The area identification of the backbone is zero. Figure 20.21 shows an autonomous system and its areas.

Figure 20.23 OSPF message formats



20.3.4 Border Gateway Protocol Version 4 (BGP4)

The **Border Gateway Protocol version 4 (BGP4)** is the only interdomain routing protocol used in the Internet today. BGP4 is based on the path-vector algorithm we described before, but it is tailored to provide information about the reachability of networks in the Internet.

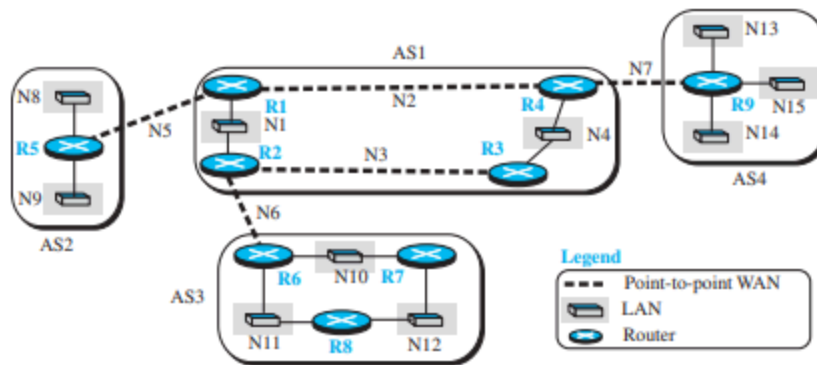
Introduction

BGP, and in particular BGP4, is a complex protocol. In this section, we introduce the basics of BGP and its relationship with intradomain routing protocols (RIP or OSPF). Figure 20.24 shows an example of an internet with four autonomous systems. AS2, AS3, and AS4 are *stub* autonomous systems; AS1 is a *transient* one. In our example, data exchange between AS2, AS3, and AS4 should pass through AS1.

Introduction

BGP, and in particular BGP4, is a complex protocol. In this section, we introduce the basics of BGP and its relationship with intradomain routing protocols (RIP or OSPF). Figure 20.24 shows an example of an internet with four autonomous systems. AS2, AS3, and AS4 are *stub* autonomous systems; AS1 is a *transient* one. In our example, data exchange between AS2, AS3, and AS4 should pass through AS1.

Figure 20.24 A sample internet with four ASs



Each autonomous system in this figure uses one of the two common intradomain protocols, RIP or OSPF. Each router in each AS knows how to reach a network that is in its own AS, but it does not know how to reach a network in another AS.

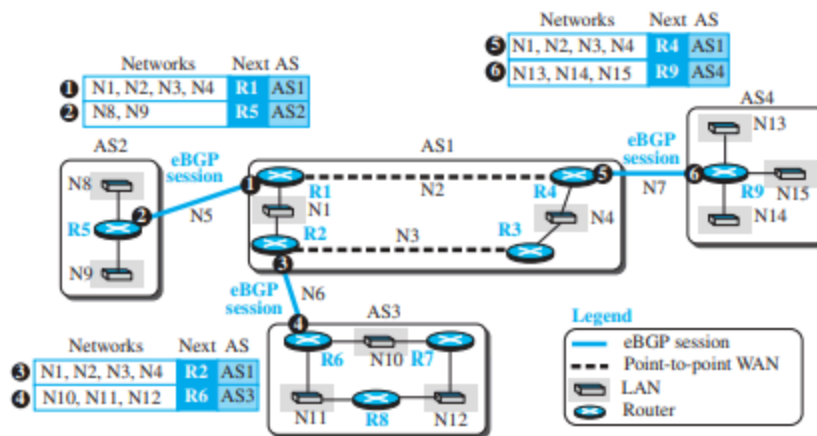
To enable each router to route a packet to any network in the internet, we first install a variation of BGP4, called *external BGP (eBGP)*, on each *border router* (the one at the edge of each AS which is connected to a router at another AS). We then install the second variation of BGP, called *internal BGP (iBGP)*, on all routers. This means that the border routers will be running three routing protocols (intradomain, eBGP, and iBGP), but other routers are running two protocols (intradomain and iBGP). We discuss the effect of each BGP variation separately.

Operation of External BGP (eBGP)

We can say that BGP is a kind of point-to-point protocol. When the software is installed on two routers, they try to create a TCP connection using the well-known port 179. In other words, a pair of client and server processes continuously communicate with each other to exchange messages. The two routers that run the BGP processes are called *BGP peers* or *BGP speakers*. We discuss different types of messages exchanged between two peers, but for the moment we are interested in only the update messages (discussed later) that announce reachability of networks in each AS.

The eBGP variation of BGP allows two physically connected border routers in two different ASs to form pairs of eBGP speakers and exchange messages. The routers that are eligible in our example in Figure 20.24 form three pairs: R1-R5, R2-R6, and R4-R9. The connection between these pairs is established over three physical WANs (N5, N6, and N7). However, there is a need for a logical TCP connection to be created over the physical connection to make the exchange of information possible. Each logical connection in BGP parlance is referred to as a *session*. This means that we need three sessions in our example, as shown in Figure 20.25.

Figure 20.25 eBGP operation



Operation of Internal BGP (iBGP)

The iBGP protocol is similar to the eBGP protocol in that it uses the service of TCP on the well-known port 179, but it creates a session between any possible pair of routers inside an autonomous system. However, some points should be made clear. First, if an AS has only one router, there cannot be an iBGP session. For example, we cannot create an iBGP session inside AS2 or AS4 in our internet. Second, if there are n routers in an autonomous system, there should be $[n \times (n - 1) / 2]$ iBGP sessions in that autonomous system (a fully connected mesh) to prevent loops in the system. In other words, each router needs to advertise its own reachability to the peer in the session instead of flooding what it receives from another peer in another session. Figure 20.26 shows the combination of eBGP and iBGP sessions in our internet.

Figure 20.26 Combination of eBGP and iBGP sessions in our internet

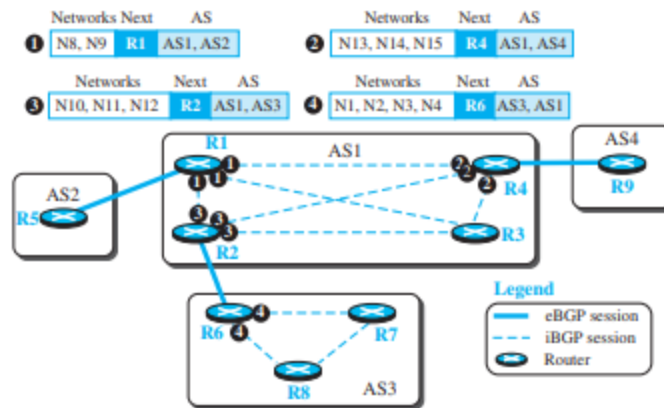
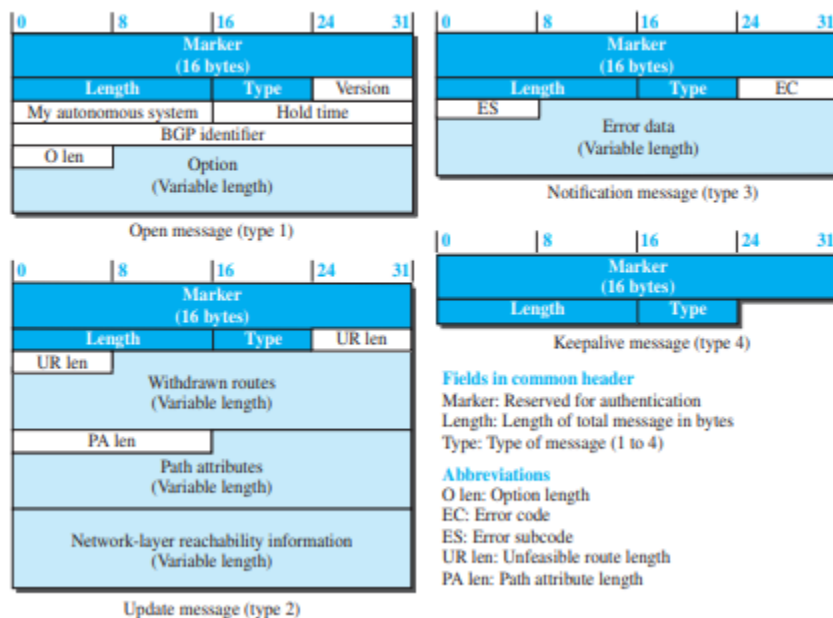


Figure 20.31 BGP messages



6	a.	<p>Explain IPv6 datagram format.</p> <p>An IPv6 datagram consists of two main parts:</p> <ol style="list-style-type: none">1. IPv6 Header: Contains essential control information for packet routing.2. Payload: The actual data being transmitted (e.g., application data, upper-layer protocol data like TCP or UDP). <p>IPv6 Datagram Header Format</p> <p>The IPv6 header is fixed-length (40 bytes) and contains the following fields:</p> <table><tr><th>Field</th><th>bits)</th><th>Description</th></tr><tr><td>Version</td><td>4 bits</td><td>Indicates the IP version; for IPv6, it is set to 6.</td></tr><tr><td>Traffic Class</td><td>8 bits</td><td>Used for quality of service (QoS) management; differentiates the data packets based on priority.</td></tr><tr><td>Flow Label</td><td>20 bits</td><td>Used to identify packets that belong to the same flow (stream of packets with specific QoS requirements).</td></tr><tr><td>Payload Length</td><td>16 bits</td><td>The length of the payload (data) section, i.e., the data after the header (not including the header).</td></tr><tr><td>Next Header</td><td>8 bits</td><td>Identifies the type of the next layer protocol (e.g., TCP, UDP, ICMPv6).</td></tr><tr><td>Hop Limit</td><td>8 bits</td><td>Similar to TTL (Time to Live) in IPv4; it limits the number of hops (routers) a packet can travel before being discarded.</td></tr><tr><td>Source Address</td><td>128 bits</td><td>The 128-bit IPv6 address of the sender (source) of the packet.</td></tr></table> <p>IPv6 Datagram Header Detailed Breakdown:</p> <ol style="list-style-type: none">1. Version (4 bits):<ul style="list-style-type: none">o This field specifies the version of the Internet Protocol. For IPv6, it is always 6.1. Traffic Class (8 bits):<ul style="list-style-type: none">o Similar to the Type of Service field in IPv4, this field is used to specify the priority or class of service (CoS). It helps in defining packet precedence and service level for data transfer (e.g., for QoS policies).2. Flow Label (20 bits):<p>This is a new field introduced in IPv6 that allows for identifying packets belonging to the same flow. A flow is a set of packets that have the same source, destination, and traffic characteristics.</p><p>It can be used by routers to give preferential treatment to certain types of traffic (e.g., real-time data like voice or video).</p>	Field	bits)	Description	Version	4 bits	Indicates the IP version; for IPv6, it is set to 6.	Traffic Class	8 bits	Used for quality of service (QoS) management; differentiates the data packets based on priority.	Flow Label	20 bits	Used to identify packets that belong to the same flow (stream of packets with specific QoS requirements).	Payload Length	16 bits	The length of the payload (data) section, i.e., the data after the header (not including the header).	Next Header	8 bits	Identifies the type of the next layer protocol (e.g., TCP, UDP, ICMPv6).	Hop Limit	8 bits	Similar to TTL (Time to Live) in IPv4; it limits the number of hops (routers) a packet can travel before being discarded.	Source Address	128 bits	The 128-bit IPv6 address of the sender (source) of the packet.	10	L2	CO3
Field	bits)	Description																											
Version	4 bits	Indicates the IP version; for IPv6, it is set to 6.																											
Traffic Class	8 bits	Used for quality of service (QoS) management; differentiates the data packets based on priority.																											
Flow Label	20 bits	Used to identify packets that belong to the same flow (stream of packets with specific QoS requirements).																											
Payload Length	16 bits	The length of the payload (data) section, i.e., the data after the header (not including the header).																											
Next Header	8 bits	Identifies the type of the next layer protocol (e.g., TCP, UDP, ICMPv6).																											
Hop Limit	8 bits	Similar to TTL (Time to Live) in IPv4; it limits the number of hops (routers) a packet can travel before being discarded.																											
Source Address	128 bits	The 128-bit IPv6 address of the sender (source) of the packet.																											

		<p>Payload Length (16 bits):</p> <p>This field specifies the length of the payload, i.e., the data portion of the packet excluding the header. The maximum payload length can be 65,535 bytes. If the payload exceeds this size, the Jumbo Payload Option is used, which allows larger payloads</p> <p>Next Header (8 bits):</p> <ul style="list-style-type: none"> o This field indicates the protocol used in the next layer (above the IP layer). It specifies whether the next layer is TCP, UDP, ICMPv6, or any other protocol (such as IPv6 Extension Headers). <p>Common values:</p> <p>6 = TCP</p> <p>17 = UDP</p> <p>58 = ICMPv6</p> <p>41 = IPv6 encapsulated in IPv4 (for tunnelling)</p> <p>1. Hop Limit (8 bits):</p> <ul style="list-style-type: none"> o Similar to the TTL (Time to Live) field in IPv4, this field limits the number of hops (routers) a packet can make. Each router that forwards the packet decrements the hop limit. If the hop limit reaches 0, the packet is discarded. This helps prevent packets from circulating endlessly due to routing loops. <p>2. Source Address (128 bits):</p> <ul style="list-style-type: none"> o This is the IPv6 address of the sender. It is 128 bits long and is written in hexadecimal notation (e.g., 2001:0db8:85a3:0000:0000:8a2e:0370:7334). <p>3. Destination Address (128 bits):</p> <p>This is the IPv6 address of the receiver. It is also 128 bits long and is similarly written in hexadecimal notation</p> <p>Dijkstra's algorithm is a greedy algorithm that solves the single-source shortest path problem for a graph with non-negative edge weights. It finds the shortest path from a source vertex to all other vertices in the graph.</p>			
	b.	<p>Write an Dijkstra's algorithm to compute shortest path through graph.</p> <p>Dijkstra's algorithm is a greedy algorithm that solves the single-source shortest path problem for a graph with non-negative edge weights. It finds the shortest path from a source vertex to all other vertices in the graph.</p> <p>Steps of Dijkstra's Algorithm</p> <p>1. Initialization:</p> <ul style="list-style-type: none"> o Mark the distance to the source node as 0. o Mark the distance to all other nodes as infinity. 	06	L1	CO3

- o Set the source node as the current node.
- o Set all nodes as unvisited.

2. Visit the Current Node:

- o For the current node, consider all of its unvisited neighbors.
- o Calculate their tentative distances from the source. The tentative distance is the sum of the distance from the source to the current node and the edge weight from the current node to the neighbor.
- o If the calculated tentative distance is smaller than the current recorded distance, update the shortest distance for the neighbor.

3. Mark the Current Node as Visited:

After considering all of the unvisited neighbors, mark the current node as visited. A visited node will not be checked again.

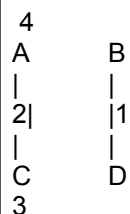
4. Select the Next Node:

Choose the unvisited node with the smallest tentative distance and make it the new current node.

Repeat steps 2-4 until all nodes are visited or the smallest tentative distance among the unvisited nodes is infinity (which means the remaining unvisited nodes are unreachable from the source).

1. Result:

Once all nodes have been visited, the algorithm will have calculated the shortest path from the source to each node. Consider the following graph with nodes labeled A, B, C, D, E:



Graph Representation (edge weights between nodes) graph = {
 'A': {'B': 4, 'C': 2},
 'B': {'A': 4, 'D': 1},
 'C': {'A': 2, 'D': 3},

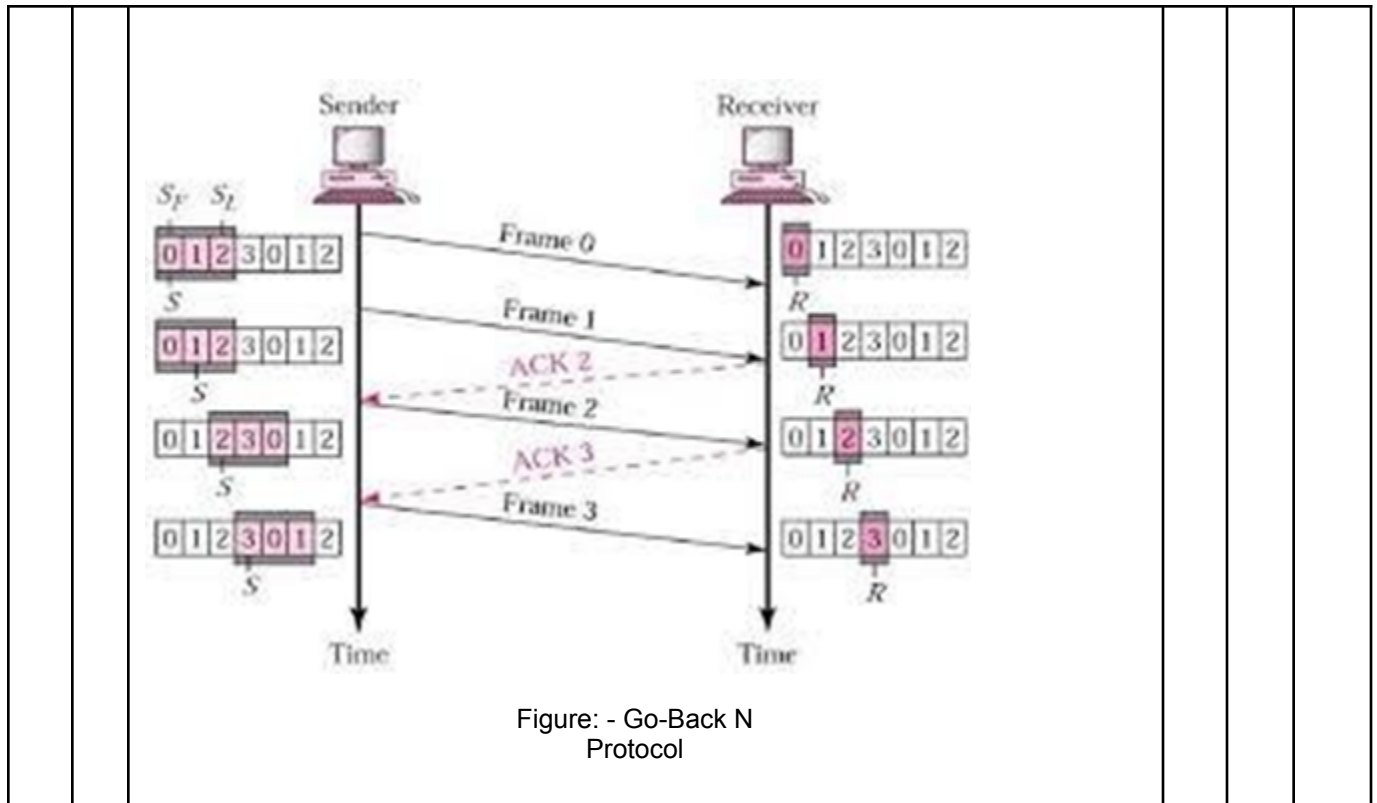
	<div>'D': {'B': 1, 'C': 3}</div> <div>}</div> <div>· Start: A</div> <div>Step-by-Step Execution:</div> <div>1. Initialization:</div> <div><div>o dist = {'A': 0, 'B': inf, 'C': inf, 'D': inf}</div><div>o prev = {'A': None, 'B': None, 'C': None, 'D': None}</div><div>o unvisited = ['A', 'B', 'C', 'D']</div></div> <div>2. Visit Node A:</div> <div><div>o Tentative distances for neighbors of A:</div><div>§ B: 4 (A → B)</div><div>§ C: 2 (A → C)</div><div>o Update dist and prev:</div><div>§ dist = {'A': 0, 'B': 4, 'C': 2, 'D': inf}</div><div>§ prev = {'A': None, 'B': 'A', 'C': 'A', 'D': None}</div></div> <div>3. Visit Node C (next minimum):</div> <div><div>o Tentative distances for neighbors of C:</div><div>§ A: 4 (C → A) (no update needed)</div><div>§ D: 5 (C → D)</div><div>o Update dist and prev:</div><div>§ dist = {'A': 0, 'B': 4, 'C': 2, 'D': 5}</div><div>§ prev = {'A': None, 'B': 'A', 'C': 'A', 'D': 'C'}</div></div> <div>4. Visit Node B (next minimum):</div> <div><div>o Tentative distances for neighbors of B:</div><div>§ A: 8 (B → A) (no update needed)</div><div>§ D: 5 (B → D) (no update needed)</div><div>o No updates.</div></div> <div>5. Visit Node D (last node):</div>			
--	-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	--	--	--

		<ul style="list-style-type: none"> o No more updates needed as all nodes are visited. <p>6. Result:</p> <ul style="list-style-type: none"> o dist = {'A': 0, 'B': 4, 'C': 2, 'D': 5} o prev = {'A': None, 'B': 'A', 'C': 'A', 'D': 'C'} <p>Reconstructing the Shortest Path</p> <p>To find the shortest path from the source (A) to any node, trace the prep dictionary:</p> <ul style="list-style-type: none"> · Path to B: A → B · Path to C: A → C · Path to D: A → C → D <p>Explanation-3 Marks</p>			
	c.	<p>Write a note on Routing Information Protocol (RIP) algorithm.</p> <p>Routing Information Protocol (RIP) is one of the oldest distance-vector routing protocols used in IP networks. RIP helps routers determine the best route to reach a destination within a network. It is based on the Bellman-Ford algorithm and uses hop count as the metric to determine the shortest path to a destination.</p> <p>RIP is typically used in small to medium-sized networks due to its simplicity, but it has limitations in scalability and speed. There are two main versions of RIP:</p> <p>1. RIP version 1 (RIP v1)</p> <p>.RIP</p> <p>RIP is a distance-vector protocol, meaning each router sends information about its known routes to its neighbors periodically. Each router maintains a routing table, which lists the routes to all known destinations along with the hop count to reach each destination.</p> <p>Key Characteristics of RIP:</p> <p>Hop Count Metric: RIP uses hop count as the metric for determining the best path. A hop is defined as the passage of data through one router.</p> <ul style="list-style-type: none"> o The maximum number of hops allowed by RIP is 15. This means any destination that requires more than 15 hops is considered unreachable. <p>Periodic Updates: RIP routers send updates about their routing tables to their neighbors every 30 seconds. This helps routers to maintain an up-to-date view of the network.</p>	04	L2	CO3

		<p>Routing Tables: Each router in a RIP-based network maintains a routing table with the destination network and the corresponding next-hop router to reach that network, along with the number of hops.</p> <p>Split Horizon and Poison Reverse: These are techniques used to prevent routing loops and to improve the convergence time in RIP.</p> <ul style="list-style-type: none"> o Split Horizon: A router will not advertise a route back to the router from which it learned the route. o Poison Reverse: A router advertises that a route has an infinite hop count (16 hops) if it is no longer valid, to avoid routing loops. <p>RIP Algorithm Process</p> <ol style="list-style-type: none"> 1. Initial State: <ul style="list-style-type: none"> o Each router initializes its routing table with directly connected routes. <p>The hop count to a directly connected network is set to 1, and all other routes are set to infinity (16 hops).</p> 2. Periodic Updates: <ul style="list-style-type: none"> o Every 30 seconds, routers exchange routing information with their neighbors. Each router sends its entire routing table to its neighbors. o The routers update their routing tables based on the received information. If a better route (with a lower hop count) is found, the routing table is updated accordingly. 3. Convergence: <ul style="list-style-type: none"> o RIP routers eventually reach a stable state where all routers in the network have the same routing information. o The protocol tries to ensure that the network converges quickly by propagating changes in routing information. 2. Route Failures: <ul style="list-style-type: none"> o If a route becomes unreachable, the router will update its routing table by setting the hop count to 16 (infinity) for that route. This is the count to infinity problem, and techniques like Split Horizon and Poison Reverse are used to avoid loops. <p>RIP Versions</p> <p>RIP v1 (Classful Routing)</p> <ul style="list-style-type: none"> · Classful: RIP v1 does not send subnet mask information in its routing updates, meaning it assumes all IP addresses belong to a classful network (A, B, or C). · Broadcast: RIP v1 uses broadcast messages for updates, which can create unnecessary traffic in large networks. 			
--	--	-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	--	--	--

		<p>No Authentication: RIP v1 does not support authentication of routing updates, making it less secure.</p> <p>RIP v2 (Classless Routing)</p> <p>Classless: RIP v2 supports CIDR (Classless Inter-Domain Routing) and sends the subnet mask along with the IP address, allowing more flexibility in network addressing.</p> <p>Multicast: RIP v2 uses multicast (address 224.0.0.9) for updates, which is more efficient than broadcasting.</p> <p>Authentication: RIP v2 supports simple password authentication to secure routing updates.</p>			
MODULE 4					
7	a.	<p>Explain Go-Back-N protocol working.</p> <p>The Go-Back-N Protocol is a sliding window protocol used in computer networks to ensure reliable data transmission over an unreliable network. It is commonly employed at the transport layer. This protocol manages error detection, retransmission, and flow control. Here's a detailed explanation of its working:</p> <p>Key Components:</p> <ol style="list-style-type: none"> Sender Window Size (N): <p>The sender maintains a window of size N that determines how many frames can be sent before waiting for an acknowledgment (ACK).</p> <ol style="list-style-type: none"> Receiver Window Size: <p>The receiver can only accept the next expected frame (size = 1). If a frame is out of order, it discards it and does not send an acknowledgment for it.</p> <ol style="list-style-type: none"> Sequence Numbers: <p>Each frame is assigned a unique sequence number that wraps around after a certain range.</p> <p>Working:</p> <ol style="list-style-type: none"> Sending Data: <ul style="list-style-type: none"> The sender can transmit up to N frames without waiting for an acknowledgment. Frames are sent continuously as long as the window is not full. Acknowledgments: 	10	L2	CO4

		<ul style="list-style-type: none"> o The receiver sends a cumulative acknowledgment (ACK) for the last correctly received, in-order frame. o For example, if frames 0, 1, and 2 are received correctly, the receiver sends ACK for frame 2, meaning it has received all frames up to 2.3. <p>Error Handling:</p> <ul style="list-style-type: none"> o If a frame is lost or an error occurs, the receiver discards all subsequent frames (out-of-order frames). o It sends no acknowledgment for the lost frame, which eventually causes the sender to detect a timeout. <p>4. Retransmission:</p> <ul style="list-style-type: none"> o When the sender's timer for a specific frame expires (due to missing ACK), it retransmits all frames starting from the unacknowledged frame. o This is where "Go-Back-N" gets its name, as the sender goes back and retransmits N frames. <p>Example:</p> <p>Parameters: Window size = 4</p> <ol style="list-style-type: none"> 1. Sender sends frames 0, 1, 2, 3. 2. Receiver acknowledges frame 0 (ACK 0). 3. Frame 1 is lost during transmission. 4. Receiver discards frames 2 and 3 (out-of-order frames) and does not send ACK for them. 5. Sender times out waiting for ACK for frame 1. 6. Sender retransmits frames 1, 2, 3. <p>Pros:</p> <ul style="list-style-type: none"> · Efficient use of bandwidth: Allows sending multiple frames without waiting for individual ACKs. · Simpler receiver design: Since the receiver only accepts in-order frames, its implementation is less complex. <p>Cons:</p> <ul style="list-style-type: none"> · Wasted bandwidth on retransmission: If a single frame is lost, all subsequent frames in the window are retransmitted, even if they were received correctly. · Not suitable for high-latency links: Retransmitting many frames can lead to inefficiency. 			
--	--	-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	--	--	--



	<p>b. With neat sketch, explain three-way handshaking of TCP connection establishment.</p> <p>The three-way handshake is the process used by the Transmission Control Protocol (TCP) to establish a reliable connection between a client and a server. It ensures that both parties are ready to communicate and that they agree on the sequence numbers for data transfer. Here's a detailed explanation:</p> <p>Steps in the Three-Way Handshake:</p> <ol style="list-style-type: none"> Step 1: SYN (Synchronization) <ul style="list-style-type: none"> The client initiates the connection by sending a SYN (synchronize) packet to the server. This packet includes: <ul style="list-style-type: none"> § An initial sequence number (ISN) chosen by the client. § A request to synchronize sequence numbers with the server. <p>Example:</p> <p>Client → Server: SYN (ISN = X)</p> Step 2: SYN-ACK (Synchronization + Acknowledgment) <ul style="list-style-type: none"> The server responds with a SYN-ACK packet. 	10	L2	CO4
--	----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	----	----	-----

		<ul style="list-style-type: none"> o This packet includes: <ul style="list-style-type: none"> § Its own ISN (Y) to synchronize sequence numbers with the client. § An ACK (acknowledgment) for the client's ISN, indicating it received the SYN packet. <p>Example:</p> <p>Server → Client: SYN-ACK (ISN = Y, ACK = X + 1)</p> <p>3. Step 3: ACK (Acknowledgment)</p> <ul style="list-style-type: none"> o The client sends an ACK packet back to the server. o This packet acknowledges the server's ISN, confirming the connection is established. <p>Example:</p> <p>Client → Server: ACK (ACK = Y + 1)</p> <p>Final State:</p> <p>After the handshake, both the client and server are in a connected state, ready to exchange data.</p> <p>Example Sequence:</p> <ul style="list-style-type: none"> · Client: SYN (ISN = 100) · Server: SYN-ACK (ISN = 200, ACK = 101) · Client: ACK (ACK = 201) <p>Key Features:</p> <ol style="list-style-type: none"> 1. Reliability: Ensures both parties are ready for data transfer. 2. Sequence Number Agreement: Allows both sides to track the data flow and detect lost packets. 3. Full-Duplex Communication Setup: Establishes a bidirectional connection. <p>Importance:</p> <ul style="list-style-type: none"> · The three-way handshake is essential for establishing a reliable connection, avoiding issues like duplicate packets or connection mismatches. 			
OR					
8	a.	With an outline, explain selective repeat protocol.	10	L2	CO4

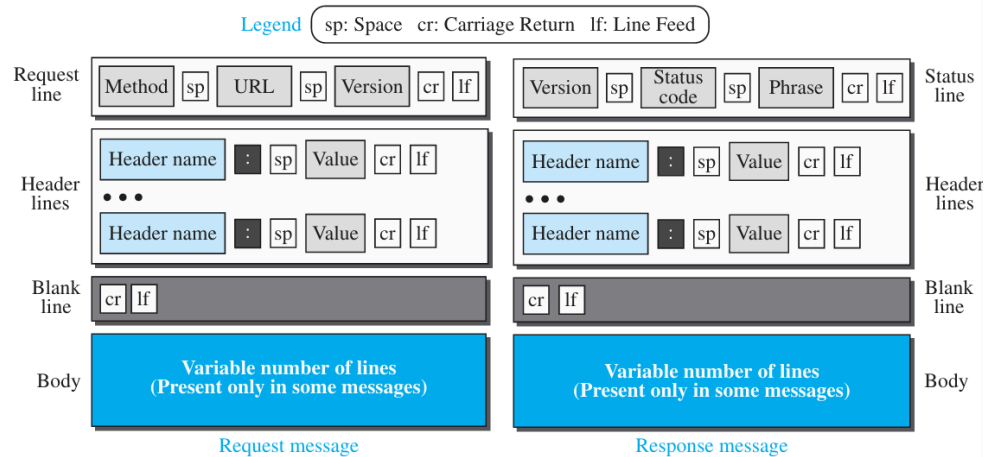
		<p>The Selective Repeat (SR) protocol is a sliding window protocol used for reliable data transfer. Unlike the Go-Back-N protocol, it retransmits only the frames that are lost or corrupted, not all subsequent frames. This makes it more efficient in handling errors, especially in networks with high latency or error rates.</p> <p>Key Features of the Protocol:</p> <ol style="list-style-type: none"> Sliding Window Mechanism: <ul style="list-style-type: none"> Both the sender and receiver maintain a window of size N. The sender can transmit up to N frames without waiting for an acknowledgment. The receiver can buffer out-of-order frames and acknowledge each frame individually. Acknowledgments (ACK): <ul style="list-style-type: none"> The receiver sends individual acknowledgments for correctly received frames, even if they are out of order. Cumulative acknowledgment is not used. Retransmissions: <ul style="list-style-type: none"> Only the frames that are reported as lost or corrupted are retransmitted. This prevents unnecessary retransmission of correctly received frames. <p>Working of Selective Repeat Protocol:</p> <ol style="list-style-type: none"> Sender Side: <ul style="list-style-type: none"> The sender transmits frames up to the window size. It starts a timer for each frame sent. If a frame's acknowledgment is not received before the timer expires, only that specific frame is retransmitted. Receiver Side: <ul style="list-style-type: none"> The receiver accepts frames within the receiver window, even if they are out of order. It buffers out-of-order frames and delivers them to the application layer in the correct sequence once missing frames are received. Error Handling: <ul style="list-style-type: none"> If a frame is lost or corrupted, the receiver discards it and waits for the sender to retransmit it. 			
--	--	---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	--	--	--

		Acknowledgments are sent only for successfully received frames.			
	b.	<p>List and explain various services provided by User Datagram Protocol (UDP).</p> <p>Defination-2 Marks</p> <p>Services & explanation-8 Marks</p> <p>User Datagram Protocol (UDP) provides several services to support the transmission of data between applications. Here are the key services provided by UDP:</p> <ol style="list-style-type: none"> 1. Connectionless Communication: <ul style="list-style-type: none"> o UDP is a connectionless protocol, meaning there is no need to establish a connection before sending data. Each packet (datagram) is sent independently, and there is no session or handshake process involved. o This makes UDP faster, but it comes at the cost of reliability and ordering. 2. Unreliable Data Delivery: <ul style="list-style-type: none"> o UDP does not guarantee delivery of packets. If a packet is lost during transmission, it is not retransmitted. The sender is not notified about lost packets, making UDP an unreliable protocol. o This is suitable for applications where speed is more critical than reliability, such as streaming or real-time applications. 3. No Acknowledgments: <ul style="list-style-type: none"> o Unlike Transmission Control Protocol (TCP), UDP does not require any acknowledgment from the receiver for the data sent. This reduces the overhead and increases transmission speed. 4. No Flow Control: <ul style="list-style-type: none"> o UDP does not implement flow control mechanisms to regulate the rate of data transmission between sender and receiver. It relies on the application to handle congestion or buffer overflow situations. 5. No Error Recovery: <ul style="list-style-type: none"> o UDP includes a checksum for error detection in the header of each datagram to check for integrity, but it does not offer any error recovery mechanisms. If a datagram is corrupted, it is discarded without any retransmission. . Data Integrity (Checksum): <ul style="list-style-type: none"> o UDP uses a checksum to verify the integrity of the data in each datagram. This ensures that the data received by the destination is not corrupted during transmission, but there is no mechanism to fix it (other than dropping the datagram). 7. Multiplexing: <ul style="list-style-type: none"> o UDP allows multiple applications to use the same network interface 	10	L2	CO4

		<p>simultaneously. It uses port numbers in its header to identify the sending and receiving applications, allowing multiplexing of communication on the same machine.</p> <ol style="list-style-type: none"> Low Overhead: <ul style="list-style-type: none"> The UDP header is simple and small, containing only the source port, destination port, length, and checksum. This makes it lightweight compared to TCP, which has more fields and requires more processing. Broadcast and Multicast: <ul style="list-style-type: none"> UDP supports broadcasting and multicasting, which allows data to be sent to multiple receivers in a single transmission. This is useful in applications like video conferencing, live streaming, and DNS queries. Faster Data Transmission: <p>Due to its minimalistic design, UDP has lower latency compared to TCP, which is suitable for time-sensitive applications like VoIP, online gaming, and video streaming, where a slight delay in transmission could degrade the quality of service.</p> 			
MODULE - 5					
9	a.	<p>Briefly explain Secure Shell (SSH).</p> <p>Secure Shell (SSH) is a cryptographic network protocol used to securely access and manage devices over an unsecured network, such as the internet. It provides a secure channel for communication between a client and a server, ensuring confidentiality, integrity, and authenticity of the transmitted data. Here's a brief overview of its key features:</p> <ol style="list-style-type: none"> Secure Remote Login: <ul style="list-style-type: none"> SSH allows users to log into remote systems securely, replacing older, less secure protocols like Telnet or rlogin. It encrypts the entire session, preventing unauthorized interception of data. Encryption: <ul style="list-style-type: none"> SSH uses strong encryption techniques (such as AES or RSA) to ensure that the data exchanged between the client and server remains confidential and is not readable by attackers. Authentication: <ul style="list-style-type: none"> SSH supports multiple forms of authentication, including password-based and public-key authentication. Public-key authentication provides a more secure way of authenticating, as it uses a pair of cryptographic keys (public and private) rather than relying on passwords. Data Integrity: <ul style="list-style-type: none"> SSH ensures data integrity by using cryptographic hash functions. This 	10	L2	CO4

	<p>prevents data from being altered during transmission.</p> <p>5. Secure File Transfer:</p> <ul style="list-style-type: none"> SSH also supports secure file transfer protocols like SFTP (Secure File Transfer Protocol) and SCP (Secure Copy Protocol), enabling secure transfer of files between systems. <p>6. Port Forwarding:</p> <ul style="list-style-type: none"> SSH can tunnel other protocols through its secure connection via port forwarding, allowing encrypted communication for otherwise insecure protocols, such as HTTP or VNC. <p>7. Versatility:</p> <ul style="list-style-type: none"> SSH is commonly used for managing remote servers, transferring files, and securing network services, making it a versatile tool for system administrators and developers. <p>Due to these features, SSH is widely used for remote administration, secure file transfers, and protecting sensitive data communications over untrusted networks.</p>			
	<p>Figure 26.25 <i>Components of SSH</i></p> <pre> graph TD subgraph SSH Application[Application] SSHCONN[SSH-CONN] SSHAUTH[SSH-AUTH] SSHTRANS[SSH-TRANS] end TCP[TCP] SSH --- TCP </pre>			
b.	Write a note on Request message and response message formats of HTTP.	10	L2	CO4

Figure 26.5 *Formats of the request and response messages*



- There are three fields in this line separated by one space and terminated by two characters.
- Method: Defines the request type.

Table 26.1 *Methods*

Method	Action
GET	Requests a document from the server
HEAD	Requests information about a document but not the document itself
PUT	Sends a document from the client to the server
POST	Sends some information from the client to the server
TRACE	Echoes the incoming request
DELETE	Removes the web page
CONNECT	Reserved
OPTIONS	Inquires about available options

Table 26.3 *Response header names*

Header	Description
Date	Shows the current date
Upgrade	Specifies the preferred communication protocol
Server	Gives information about the server
Set-Cookie	The server asks the client to save a cookie
Content-Encoding	Specifies the encoding scheme
Content-Language	Specifies the language
Content-Length	Shows the length of the document
Content-Type	Specifies the media type
Location	To ask the client to send the request to another site
Accept-Ranges	The server will accept the requested byte-ranges
Last-modified	Gives the date and time of the last change

OR

10

a.

With neat diagram, explain the basic model of FTP.

04

L2

CO4

FTP (File Transfer Protocol) is used for transferring files between a client and a server over a TCP/IP network. It follows a client-server model, where the client requests files or services, and the server provides them. The FTP model typically involves two main connections between the client and the server:

1. Control Connection (Command Channel): Used for sending commands and receiving responses.
2. Data Connection: Used for transferring the actual file data.

- File Transfer Protocol (FTP) is the standard protocol provided by TCP/IP for copying a file from one host to another.

Figure 26.10 *FTP*

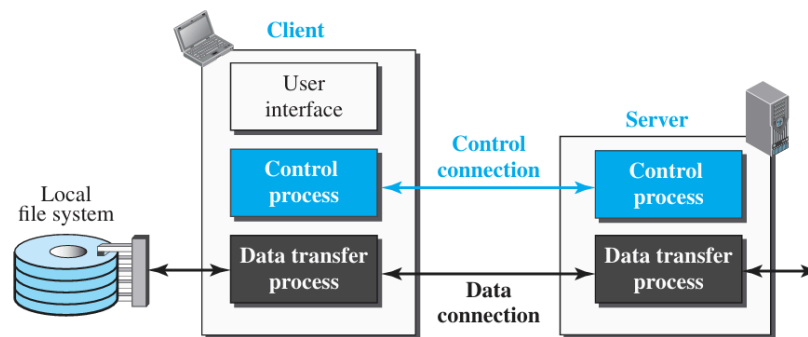


Figure: - File Transfer Protocol

Components of the FTP Model:

1. Control Connection (Port 21):

- o Purpose: This is the channel through which the client and server communicate for control purposes. It is used to send FTP commands (like login, file commands) and receive responses.

Communication: The client opens the connection to port 21 of the server, where it sends commands such as USER, PASS, LIST, RETR, and so on. The server responds with status codes and messages.

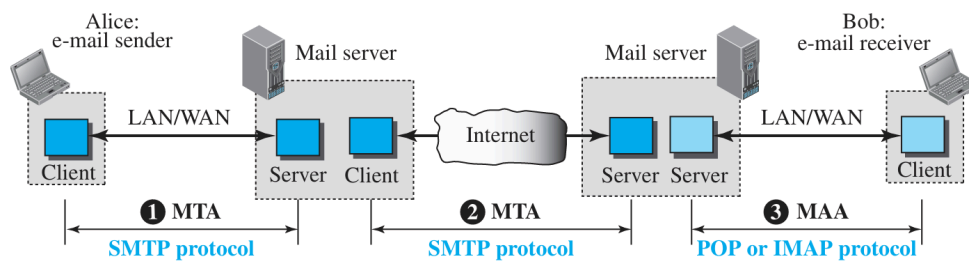
1. Data Connection (Port 20 or Passive Mode):

- o Purpose: This channel is used for transferring actual file data between the

		<p>client and server.</p> <ul style="list-style-type: none"> o Communication: <p>§ Active Mode: The client establishes the control connection on port 21, and the server opens a data connection from port 20 to the client. This method is less common due to firewall restrictions.</p> <p>§ Passive Mode: The server opens a random port and tells the client to connect to it for data transfer. This mode is more common, especially when the client is behind a firewall or NAT (Network Address Translation).</p> <p>Flow of FTP Communication:</p> <ol style="list-style-type: none"> 1. Connection Establishment: <ul style="list-style-type: none"> o The client opens a control connection to the server on port 21 and sends the login credentials (username and password) for authentication. 2. Command and Response: <ul style="list-style-type: none"> o The client sends FTP commands like LIST, RETR (retrieve file), STOR (store file), etc., through the control connection. o The server responds with status codes (e.g., 200 OK, 550 File Not Found). 3. Data Transfer: <ul style="list-style-type: none"> o Once the command is processed (e.g., RETR for downloading a file), a data connection is established either in active or passive mode. o The file is transferred over this data connection. 4. Termination: <ul style="list-style-type: none"> o Once the file transfer is complete, the data connection is closed, and the control connection can also be terminated when the session ends. 			
	b.	<p>Describe the architecture of electronic mail (e-mail). The architecture of email involves several components that work together to ensure the proper delivery, receipt, and management of messages. Email follows a client-server model, and the entire process can be broken down into key components and protocols used for sending, receiving, and storing email.</p> <ol style="list-style-type: none"> 1. Email Clients (User Interface) <p>Purpose: Email clients are the software applications or programs used by end users to send, receive, and manage email messages. They can be either desktop-based (like Microsoft Outlook, Thunderbird) or web-based (like Gmail, Yahoo Mail).</p> <p>Responsibilities:</p> <ul style="list-style-type: none"> o Composing and reading emails 	06	L3	CO4

		<ul style="list-style-type: none"> o Managing inbox and folders o Sending attachments o Syncing email data with the server <p>2. Mail Servers (Email Servers)</p> <p>Purpose: Mail servers are responsible for handling the email messages sent and received by users. There are typically two types of mail servers involved:</p> <ul style="list-style-type: none"> o Incoming Mail Servers (for receiving messages) o Outgoing Mail Servers (for sending messages) <p>Incoming Mail Servers:</p> <p>These servers are responsible for receiving and storing emails for users. They follow protocols such as:</p> <p>POP3 (Post Office Protocol v3):</p> <ul style="list-style-type: none"> o Used for retrieving emails from the server. POP3 downloads emails from the server and stores them on the client's device. It typically deletes the email from the server after download. <p>IMAP (Internet Message Access Protocol):</p> <ul style="list-style-type: none"> o Used for retrieving emails while keeping them stored on the server. IMAP allows users to access their emails from multiple devices without deleting the messages from the server, providing a more flexible way of managing emails. <p>Outgoing Mail Servers:</p> <p>These servers are responsible for sending email messages from clients to recipients. They use protocols like:</p> <p>SMTP (Simple Mail Transfer Protocol):</p>			
--	--	----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	--	--	--

Figure 26.15 *Protocols used in electronic mail*



- o The protocol used to send outgoing emails. SMTP is responsible for the delivery of email messages between servers. It works by relaying email to the destination server using the recipient's domain.

3. Mail Transfer Agents (MTAs)

Purpose: MTAs are responsible for transferring email messages between mail servers. When you send an email, the email client contacts the SMTP server (outgoing mail server). The SMTP server then forwards the message to the destination server (recipient's mail server) using a series of MTAs.

Responsibilities:

- o Relaying email from the sender's email server to the recipient's email server.
- o Ensuring the correct routing of the email message to the recipient's server.

4. Mail Delivery Agents (MDAs)

Purpose: MDAs are responsible for the final delivery of the email to the recipient's mailbox on the incoming mail server.

Responsibilities:

- o After the MTA forwards the email to the recipient's server, the MDA delivers the email to the user's mailbox.
- o Examples include Dovecot and Procmail.

1. Email Protocols

The following protocols are essential for the functioning of email:

SMTP (Simple Mail Transfer Protocol):

- o Used for sending emails from the sender's email client to the email server and between email servers (to deliver the message to the recipient's server).

POP3 (Post Office Protocol v3):

- o Used for retrieving emails from the server. POP3 downloads emails to the client and removes them from the server, so they cannot be accessed from another device.

	<p>IMAP (Internet Message Access Protocol):</p> <ul style="list-style-type: none"> o Used for retrieving emails from the server while keeping them on the server. IMAP allows users to organize and manage their emails from different devices and retains the emails on the server for synchronization. <p>2. Email Addressing</p> <p>Email Address: A unique identifier for an email recipient, typically in the form of username@domain.com. The username represents the individual recipient, while the domain identifies the mail server.</p> <p>Example: john.doe@example.com</p> <p>3. Email Flow (Sending and Receiving Process)</p> <p>The process of sending and receiving email can be broken down into the following steps:</p> <p>Sending an Email:</p> <ol style="list-style-type: none"> 1. The user creates and sends an email via an email client (e.g., Outlook or Gmail). 2. The email client contacts the SMTP server (outgoing mail server) using SMTP. 3. The SMTP server checks the recipient's domain and finds the appropriate MTA. 4. The email is passed through the MTA to the recipient's incoming mail server. 5. The email is then handed over to the MDA (Mail Delivery Agent), which places it in the recipient's mailbox. <p>Receiving an Email:</p> <ol style="list-style-type: none"> 1. The recipient's email client contacts the incoming mail server using either POP3 or IMAP to check for new messages. 2. The mail server retrieves the email from the user's mailbox and forwards it to the client. 3. The email client displays the message to the recipient. <p>4. Security Measures in Email</p> <p>Email systems often implement security measures to protect the integrity and confidentiality of email communication:</p> <p>TLS (Transport Layer Security): Ensures encrypted communication between mail clients and servers.</p> <p>SPF (Sender Policy Framework): Helps verify the sender's domain to prevent email spoofing.</p>			
--	------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	--	--	--

DKIM (DomainKeys Identified Mail): Uses cryptographic signatures to verify the authenticity of the sender's domain.

DMARC (Domain-based Message Authentication, Reporting & Conformance): A policy framework that works with SPF and DKIM to prevent email abuse and phishing.

Figure 26.16 Example 26.12

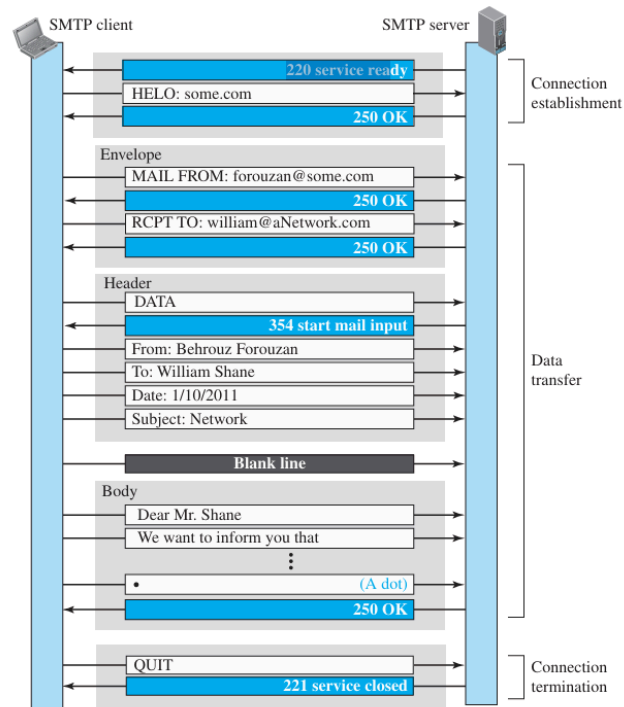


Figure 26.12 Common scenario

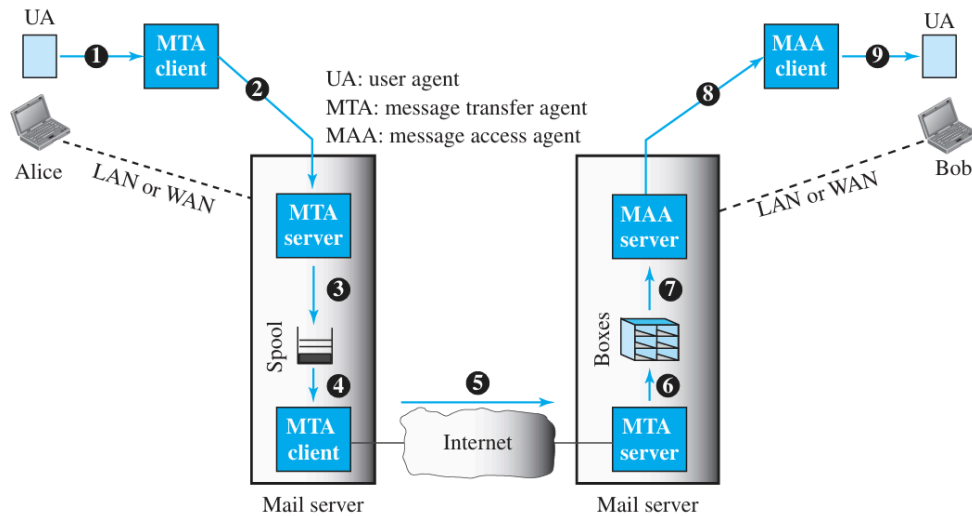


Figure 26.13 Format of an e-mail

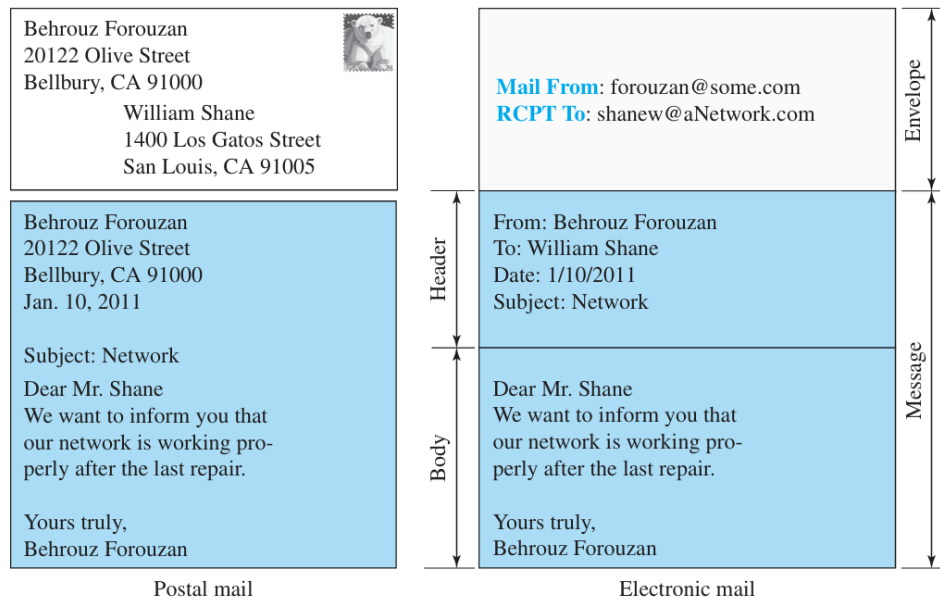
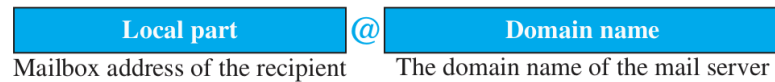


Figure 26.14 E-mail address



c. Briefly explain Recursive Resolution and Iterative Resolution in DNS.

10

L2

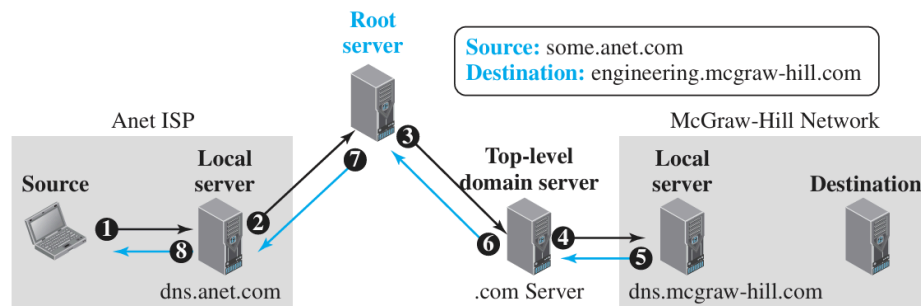
CO4

DNS (Domain Name System) is used to translate human-readable domain names (like `www.example.com`) into IP addresses that computers can use to communicate with each other. When a client (e.g., a web browser) needs to resolve a domain name, the DNS query can be processed in one of two ways: recursive resolution or iterative resolution.

Resolution can be of two types:
Recursive Resolution
Iterative Resolution

1. Recursive Resolution:

Figure 26.36 Recursive resolution



Definition: In recursive resolution, the DNS client (usually a resolver) sends a query to a DNS server, and the server is responsible for fully resolving the query by either returning the final answer (IP address) or querying other DNS servers on behalf of the client until the answer is found.

Process:

- o The client sends a query to a recursive DNS resolver.
- o If the resolver does not have the answer in its cache, it queries other DNS servers, starting from the root server to the TLD (Top-Level Domain) server, and then to the authoritative name server for the domain.

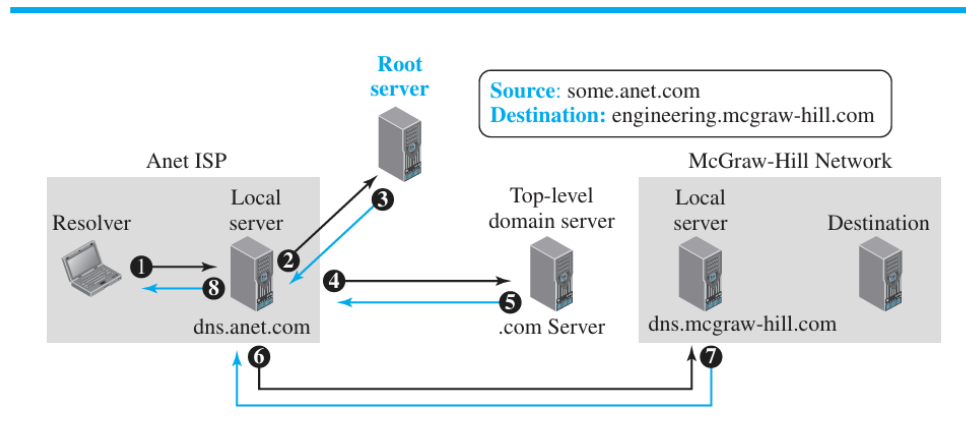
- o The recursive DNS server continues to query servers until it gets the final IP address and returns it to the client.

Key Points:

- o The client sends only one request, and the server handles the entire process.
- o The server must perform all the steps to resolve the query.
- o It's slower than iterative resolution but provides a complete answer to the client.

2. Iterative Resolution:

Figure 26.37 *Iterative resolution*



Definition: In iterative resolution, the DNS client queries a DNS server, and the server provides the best answer it has (which may not be the final answer). If the server doesn't know the answer, it provides a referral to another server that may know more.

Process:

- o The client sends a query to a DNS resolver.
- o If the resolver does not know the answer, it sends a referral to a server that is closer to the final answer (e.g., a root server or a TLD server).
- o The client then sends the query to the next server (according to the referral) and repeats the process until it receives the final IP address.

Key Points:

- o The client must follow the referrals and may have to send multiple queries to different servers.
- o The server only provides partial answers or referrals to other servers.
- o It's faster than recursive resolution, but the client is more involved in the resolution process.