BCS502

Model Question Paper-1 with effect from 2022-23 (CBCS Scheme)

USN | | | | | | | | | | |

Fifth Semester B.E. Degree Examination

Subject Title Computer Networks

TIME: 03 Hours **Max. Marks: 100**

Note: 01. Answer any **FIVE** full questions, choosing at least **ONE** question from each **MODULE**.

		Module -1	*Bloom's Taxonomy Level	COs	Marks
Q.01	a	Explain the four basic topologies used in networks. List advantages and disadvantages of each of them.	L1,L2	1, 2	7
	b	What is meant by logical connection in TCP/IP. Explain with diagram how the identical objects interact.	L1,L2	1, 2	7
	c	What is data communication? Explain its characteristics and components.	L1,L2	1, 2	6
		OR			
Q.02	a	What are guided transmission media? Explain twisted pair cable in detail.	L1,L2	1, 2	7
	b	Describe each layer of the TCP/IP and its responsibility.	L1,L2	1, 2	7
	c	Compare OSI and TCP/IP Models. What are the reasons for OSI model to fail?	L1,L2	1, 2	6
		Module-2			
Q. 03	a	What is bit oriented framing and its frame pattern. Explain with example byte stuffing and unstuffing in bit oriented framing.	L1,L2,L3	2, 3	7
	b	Explain the 3 different HDLC frames with diagram.	L1,L2,L3	2, 3	7
	c	What is the difference between ALOHA and Slotted ALOHA	L1,L2,L3	2, 3	6
		OR			
Q.04	a	Explain CRC encoder and decoder for 4 bit dataword.	L1,L2,L3	2, 3	7
	b	Explain stop and wait protocol with FSM and Flow diagram.	L1,L2,L3	2, 3	7
	c	What is PPP? What are the services provided by PPP?	L1,L2,L3	2, 3	6
		Module-3			
Q. 05	a	Explain classful addressing system with a neat diagram.	L1,L2,L3	2, 3	7
	b	Write Dijkstra's algorithm to compute shortest path with an example.	L1,L2,L3	2, 3	7
	c	Why is subnetting used and explain its importance.	L1,L2,L3	2, 3	6
		OR			
Q. 06	a	Define and explain routing and forwarding in network layer.	L1,L2,L3	2, 3	7
	b	Explain Open Shortest Path First Protocol with example.	L1,L2,L3	2, 3	7
	c	Explain DHCP and its importance?	L1,L2,L3	2, 3	6
		Module-4			
Q. 07	a	Draw the FSM diagrams for connectionless and connected oriented services offered by transport layer.	L1,L2,L3	3, 4	7
	b	List the services and applications of UDP.	L1,L2,L3	3, 4	7
	c	Explain Go-Bank-N protocol working.	L1,L2,L3	3, 4	6
		OR			
Q. 08	a	Explain connection establishment of TCP using 3-way handshaking.	L1,L2,L3	3, 4	7

	b	Explain TCP Congestion control.	L1,L2,L3	3, 4	7
	c	Explain with example stop and wait protocol.	L1,L2,L3	3, 4	6
Module-5					
Q. 09	a	Explain Standard and Non-Standard Application Layer Protocols.	L1,L2,L3	3, 4	7
	b	Differentiate client server paradigm and peer-to-peer paradigm.	L1,L2,L3	3, 4	7
	c	Differentiate between Persistent and Non Persistent connection in HTTP.	L1,L2,L3	3, 4	6
OR					
Q. 10	a	Explain how data connections happens in File Transfer Protocol.	L1,L2,L3	3, 4	7
	b	List the difference between local and remote logging.	L1,L2,L3	3, 4	7
	c	Explain briefly Domain Name System (DNS)	L1,L2,L3	3, 4	6

REVISED BLOOMS TAXONOMY LEARNING LEVEL (RBT)					
L1: Remember	L2: Understand	L3: Apply	L4: Analyze	L5: Evaluate	L6: Create

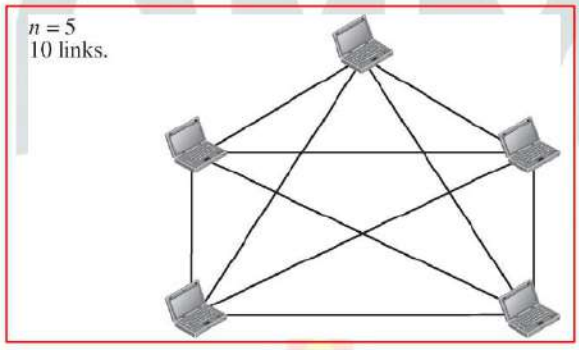
COURSE OUTCOMES (COs)	
1	**Explain** the fundamentals of computer networks.
2	**Apply** the concepts of computer networks to demonstrate the working of various layers and protocols in communication network.
3	**Analyze** the principles of protocol layering in modern communication systems.
4	**Demonstrate** various Routing protocols and their services using tools such as Cisco packet tracer.

PROGRAM OUTCOMES (POs)					
1	Engineering Knowledge	5	Modern tool usage	9	Individual and Team-Work
2	Problem Analysis	6	Engineer and Society	10	Communication
3	Design / Development Solutions	7	Environment and Sustainability	11	Project Management and Finance
4	Conduct Investigations of Complex problems	8	Ethics	12	Life-long Learning

MODEL QUESTION PAPER WITH SOLUTIONS.

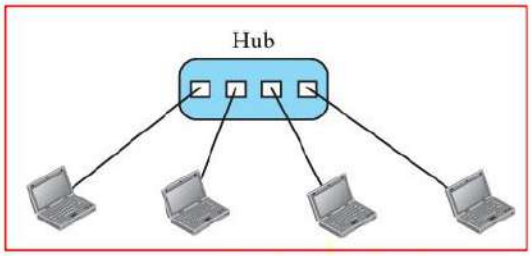
1a. Explain the four basic topologies used in networks. List advantages and disadvantages of each of them.

- **Mesh Topology:**
  - Every device is connected to every other device, requiring n(n−1)/2 links for n devices.
  - Node 1 must be connected to n – 1 nodes, node 2 must be connected to n – 1 nodes, and finally node n must be connected to n – 1 nodes. We need n (n – 1) physical links. However, if each physical link allows communication in both directions (duplex mode), we can divide the number of links by 2. In other words, we can say that in a mesh topology, we need n (n – 1) / 2 duplex-mode links.
  - Advantages: Dedicated links, fault isolation, robust, secure.
  - Disadvantages: Expensive, complex installation, excessive cabling.
  - Example: Telephone networks between regional offices.

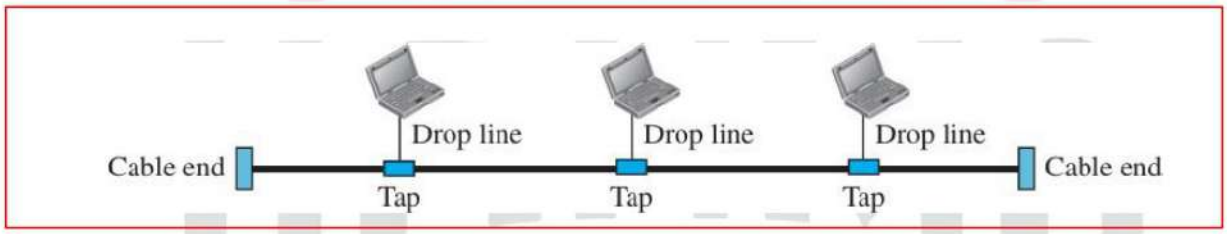


- **Star Topology**:
  - Each device is connected to a central hub, which manages communication.
  - Advantages: Easy installation and fault isolation; if a link fails, only that device is affected.
  - Disadvantages: Entire system fails if the hub goes down.

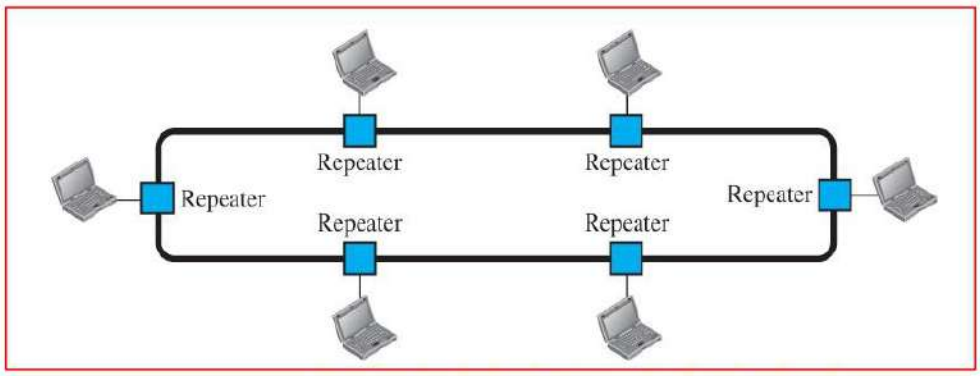
- Common in local area networks (LANs).



- **Bus Topology**:
  - All devices are connected to a single backbone cable.
  - Advantages: Easy installation, less cabling than mesh.
  - Disadvantages: Difficult to add devices, faults in the backbone disrupt the entire network.
  - Example: Early Ethernet LANs.



- **Ring Topology**:
  - Devices are connected in a loop, with signals traveling in one direction through repeaters.
  - Advantages: Easy to install, simple fault detection.
  - Disadvantages: A break in the ring can disable the entire network, though dual rings or switches can mitigate this.
  - Example: IBM's Token Ring LANs.



1b. What is meant by logical connection in TCP/IP. Explain with diagram how the identical objects interact.

Layers in the TCP/IP Protocol Suite

TCP/IP protocol suite functions and responsibilities of each layer.

Understanding the Logical Connections Between Layers

To grasp the role of each layer, it's helpful to visualize the **logical connections** between them. Figure 1.21 in the book illustrates these connections in a simple internet model.

- **End-to-End vs. Hop-to-Hop Duties:**
  - The **application, transport, and network layers** are responsible for **end-to-end** communication, meaning they manage data from one end device to the other across the network.
  - The **data-link and physical layers**, on the other hand, handle communication on a **hop-to-hop** basis, where each "hop" refers to a host or router.

This distinction is key: the top three layers operate across the entire internet, while the lower two layers manage communication on individual network segments or "links."

Data Units and Layer Responsibilities

Another important way to understand these connections is by considering the **data units** created at each layer.

- In the **top three layers**, the data units (referred to as packets) are **not modified** by routers or link-layer switches.

- In the **bottom two layers**, however, the packet created by the host can be modified by routers but **not** by link-layer switches.

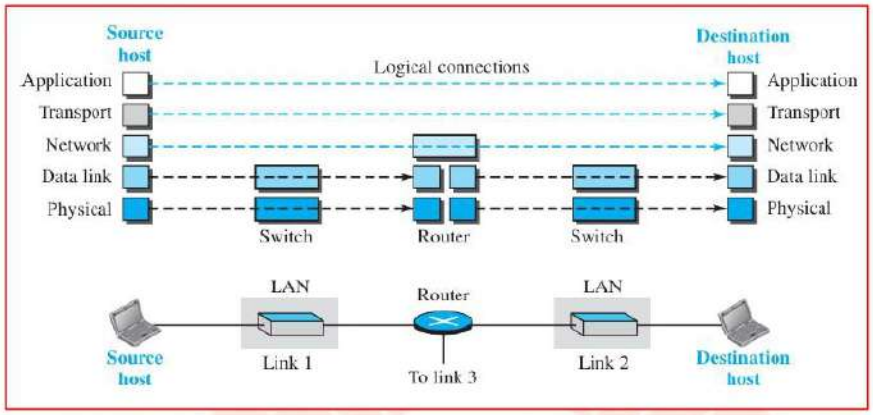
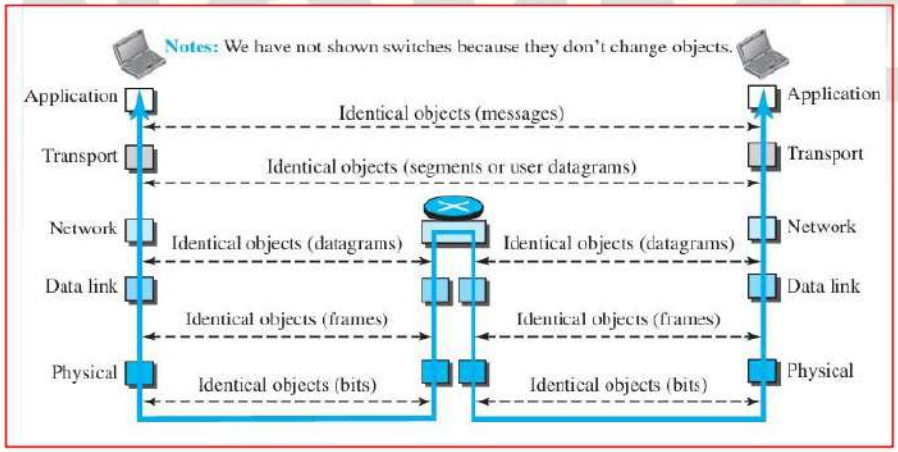


Figure shows a second principle of protocol layering: identical objects exist below each layer for connected devices.

- At the **network layer**, even though there's a logical connection between two hosts, a router might fragment packets into smaller units.
- The link between two hops does not alter these packets.

This layering approach allows for a structured, predictable method of managing data as it moves across the network.



Identical objects in the TCP/IP protocol suite

1c. What is data communication? Explain its characteristics and components

: **Data communication** is the process of transferring data from one point to another using a communication system. It involves several essential components and mechanisms to ensure the accurate and timely delivery of data.

The performance of a data communication system relies on four key characteristics: delivery, accuracy, timeliness, and jitter.

1. **Delivery**: The system must ensure that data reaches the correct destination. Only the intended recipient—whether a device or a user—should receive the data.
2. **Accuracy**: Data must be transmitted without errors. If data is altered during transmission and not corrected, it becomes unusable.
3. **Timeliness**: Data must be delivered promptly. Delayed data, especially in applications like video and audio, lose their value. For real-time transmission, data must be delivered in the same sequence and without significant delays.
4. **Jitter**: Jitter refers to the inconsistency in packet arrival times. Inconsistent delays, such as video packets arriving at varying intervals, can degrade the quality of the audio or video. For instance, if video packets are sent every 30 ms, but some arrive after 40 ms, the video quality will be affected.

Components

A data communication system includes the following components:

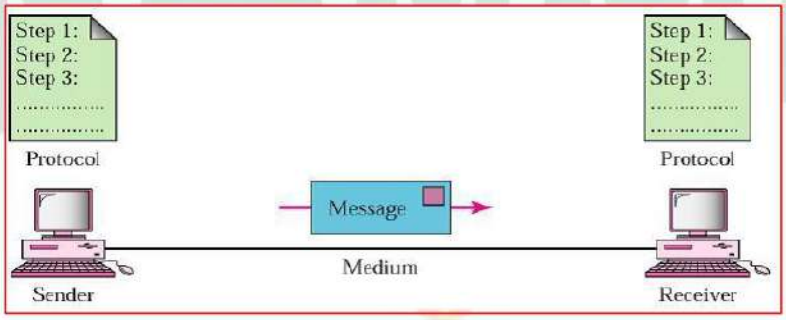


Figure 1.1: A data communications system has five components

2. **Message:** The data or information being communicated (e.g., text, images, audio).
3. **Sender:** The device that sends the message, such as a computer or smartphone.
4. **Receiver:** The device that receives the message, like another computer or a printer.
5. **Transmission Medium:** The physical path through which the data is transmitted, like

cables or radio waves.

6. **Protocol:** A set of rules that governs the communication between devices to ensure proper data exchange.

2a. What are guided transmission media? Explain twisted pair cable in detail

Guided Media: These include twisted-pair cables, coaxial cables, and fiber-optic cables.

Guided media are types of communication channels that provide a specific path for signals to travel from one device to another. These include:

1. **Twisted-Pair Cable**: This type of cable consists of pairs of insulated copper wires twisted together. The twisting helps reduce electromagnetic interference and maintains signal quality.

Twisted-Pair Cable

A twisted pair cable consists of two insulated copper conductors twisted together. Each wire in the pair serves a different function: one carries the signal to the receiver, and the other acts as a ground reference. The receiver processes the difference between the two wires to retrieve the signal.

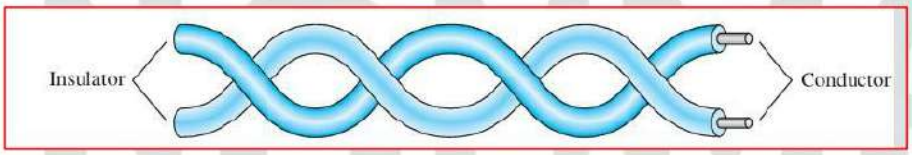


Figure 1.28: Twisted-pair cable

Noise and Interference

Twisted pair cables are designed to minimize the impact of interference (noise) and crosstalk. When the wires are parallel, noise or crosstalk can affect each wire differently due to their varying distances from the sources of interference. By twisting the wires, the cable maintains a balance. In each twist, the relative positions of the wires to the noise source change, helping to ensure that both wires experience similar levels of interference. This twisting reduces the impact of unwanted signals, as the receiver calculates the difference between the wires, canceling out most of the noise.

Shielded vs. Unshielded Twisted-Pair Cables

- **Unshielded Twisted-Pair (UTP):** The most common type used in communications, UTP cables do not have additional shielding. They are less expensive and less bulky but can be more susceptible to interference.

Shielded Twisted-Pair (STP): STP cables have an additional metal foil or braided mesh covering each pair of conductors. This shielding reduces interference and improves signal quality but makes the cables bulkier and more costly. STP is primarily used by IBM and is less common outside of their applications.

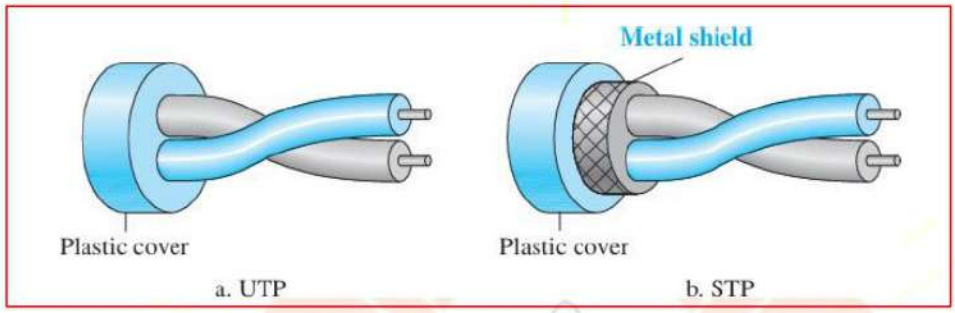


Figure 1.29: UTP and STP cables

Categories of UTP Cables

The Electronic Industries Association (EIA) classifies UTP cables into seven categories, with Category 1 being the lowest quality and Category 7 being the highest. Each category is suitable for specific applications, and the standards help ensure the cable meets certain performance criteria.

Connectors

The RJ45 connector is the most common connector for UTP cables. It is a keyed connector, meaning it can only be inserted in one direction, which ensures a proper connection.

Table 1.1: Categories of unshielded twisted-pair cables

Category	Specification	Data Rate (Mbps)	Use
1	Unshielded twisted-pair used in telephone lines	2	Telephone
2	Unshielded twisted-pair originally used in T1 lines	10	T1 Lines
3	Improved Category 2 used in LANs	20	LANs
4	Improved Category 3 used in Token Ring networks	100	Token Ring Networks
5	Cable wire is normally 24 AWG with a jacket and outside sheath	125	LANs

5E	An extension of Category 5 with additional features to minimize crosstalk and electromagnetic interference	125	LANs
6	New category with matched components from the same manufacturer; cable tested at a 200-Mbps data rate	200	LANs
7	Sometimes called SSTP (Shielded Screen Twisted-Pair); each pair is wrapped in helical metallic foil followed by a metallic foil shield	600	LANs

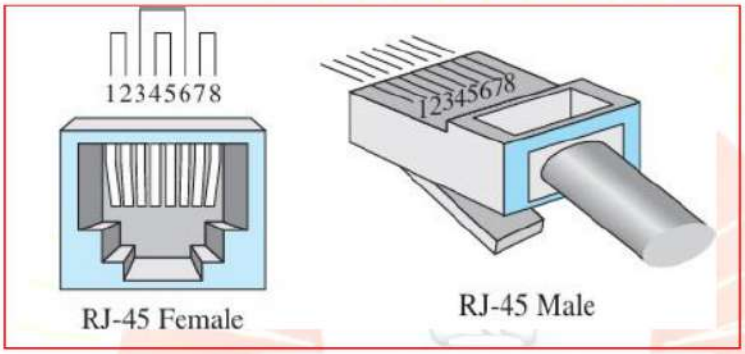


Figure 1.30: UTP connector

Performance

The performance of twisted-pair cables is often assessed by measuring attenuation (signal loss) in relation to frequency and distance. Although twisted-pair cables can handle a broad range of frequencies, attenuation increases significantly at frequencies above 100 kHz. Attenuation is measured in decibels per kilometer (dB/km), and higher frequencies result in greater signal loss.

Applications

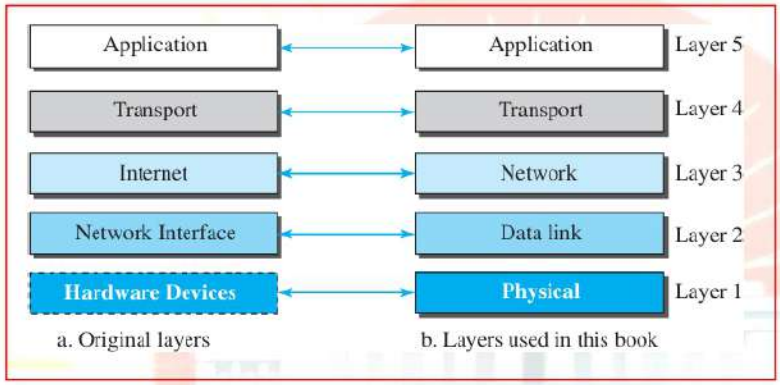
Twisted-pair cables are widely used in various applications:

- **Telephone Lines:** Used for voice and data transmission in the local loop connecting subscribers to telephone offices.
- **DSL Lines:** Provide high-data-rate connections by utilizing the high bandwidth of UTP cables.
- **Local-Area Networks (LANs):** Employed in networks such as 10Base-T and 100Base-T for data transmission.

2b. Describe each layer of the TCP/IP and its responsibility .

Layered Architecture

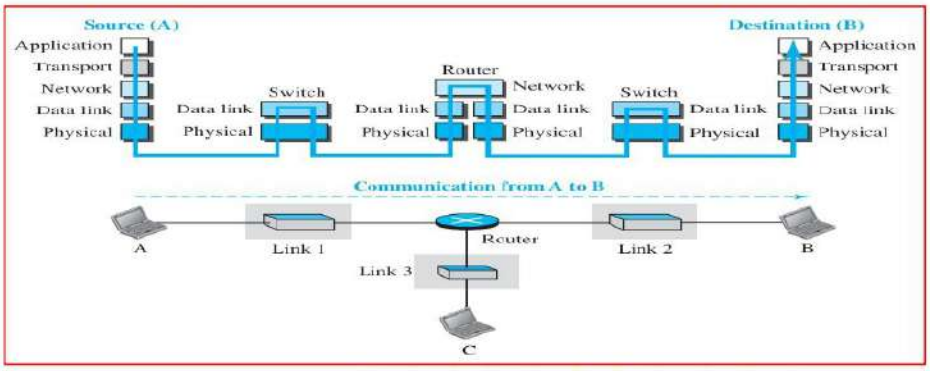
To understand how the layers in the TCP/IP protocol suite work during communication between two hosts, let's consider a small network composed of three local area networks (LANs), each connected by a link-layer switch. These LANs are also interconnected through a router.



Layers in the TCP/IP protocol suite

In this scenario, imagine that **Host A** (the source) communicates with **Host B** (the destination). The communication process involves five devices:

2. Source Host (Computer A)
3. Link-layer switch in LAN 1
4. Router
5. Link-layer switch in LAN 2
6. Destination Host (Computer B)



Each of these devices operates at different layers of the TCP/IP protocol stack, depending on its role in the network:

1. Hosts (Source and Destination)

Both **Host A** and **Host B** are involved in all five layers of the TCP/IP model:

- **Application Layer**: The source host (Host A) creates a message at the application layer and sends it down through the stack.
- **Transport Layer**: The message is passed to the transport layer, which ensures reliable delivery.
- **Network Layer**: At the network layer, the message is encapsulated into packets for transmission across the network.
- **Data Link Layer**: The packets are then prepared for transmission over the physical network in the data-link layer.
- **Physical Layer**: Finally, the message is sent through the physical medium (wires, cables, etc.) to reach the destination host.

At the destination, **Host B** receives the message at the physical layer and passes it up through the layers until it reaches the application layer for processing.

2. Router

A **router** plays a different role and operates at three layers of the TCP/IP model:

- **Network Layer**: The router's primary function is routing packets across networks. It forwards packets based on their destination IP address.
- **Data Link Layer & Physical Layer**: A router is connected to multiple links, and each link may use a different data-link and physical-layer protocol. For instance, if a packet arrives from **LAN 1** (Link 1) using one set of protocols, the router must handle it and forward it to **LAN 2** (Link 2) using another set of protocols.

Importantly, the router does not deal with the transport or application layers, as its role is solely to move packets between networks.

2. Link-Layer Switch

A **link-layer switch** operates only at the data-link and physical layers:

- **Data Link Layer**: The switch processes the data frames and ensures they are forwarded to the correct device within the same LAN.
- **Physical Layer**: The switch forwards the data through the physical medium.

Unlike routers, link-layer switches do not need to handle different sets of protocols for different links. They operate within a single LAN, using a single protocol set for the data-link and physical layers.

2c. Compare OSI and TCP/IP Models. What are the reasons for OSI model to fail?

Comparison of OSI and TCP/IP Models:

Aspect	OSI Model	TCP/IP Model
Full Form	Open Systems Interconnection	Transmission Control Protocol/Internet Protocol
Layers	7 layers: Application, Presentation, Session, Transport, Network, Data Link, Physical	4 layers: Application, Transport, Internet, Network Access
Development	Developed by ISO (International Organization for Standardization)	Developed by DARPA (Defense Advanced Research Projects Agency)
Purpose	Generalized framework for networking standards	Specific framework for internet and networking protocols
Layer Functions	Strict separation of layers with detailed definitions	Practical implementation, combining layers for simplicity
Protocol Dependency	Independent of specific protocols	Protocol-driven (e.g., TCP, IP)

Aspect	OSI Model	TCP/IP Model
Approach	Conceptual model for understanding networks	Practical model for implementing networks
Usage	Reference model for standardizing networks	Widely used in real-world network systems
Flexibility	More abstract and theoretical	Designed for real-world flexibility and performance
Examples of Protocols	Protocols are conceptualized, not defined	Protocols include HTTP, FTP, SMTP, TCP, IP, etc.

Reasons for OSI Model's Limited Adoption ("Failure")

1. **Complexity:**
   - The OSI model introduced seven layers with distinct roles, which made it more complex and less practical for implementation in the early days of networking.
2. **Performance Issues:**
   - The strict separation of layers sometimes resulted in inefficiencies, as certain functionalities overlapped across layers or required cross-layer optimizations.
3. **Slow Adoption:**
   - The OSI model's development process was bureaucratic and slow, while the TCP/IP model gained rapid adoption due to its immediate applicability and support for the growing internet.
4. **Protocol Suitability:**
   - OSI model protocols (e.g., CLNP, TP4) were not as robust, efficient, or widely adopted as TCP/IP's protocols, leading to a lack of incentive for adoption.
5. **Early Market Capture by TCP/IP:**
   - By the time the OSI model was fully developed, TCP/IP had already become the de facto standard due to its use in ARPANET and its backing by key organizations.

3a. What is bit oriented framing and its frame pattern. Explain with example byte stuffing and unstuffing in bit oriented framing.

Bit-Oriented Framing

Bit-oriented framing is a method used in data communication protocols to encapsulate frames (units of transmission) as sequences of bits. It does not rely on byte boundaries and uses specific bit patterns as delimiters to indicate the start and end of a frame.

Frame Pattern

A typical bit-oriented frame consists of:

1. **Frame Delimiters (Flags):**
   - A special bit sequence (e.g., 01111110 in HDLC) marks the beginning and end of a frame.
2. **Control Information:**
   - Contains headers like source and destination addresses, sequence numbers, or error-checking data.
3. **Payload (Data):**
   - The actual data being transmitted.
4. **Frame Check Sequence (FCS):**
   - Ensures data integrity by detecting transmission errors.

Byte Stuffing and Unstuffing in Bit-Oriented Framing

Concept

If the payload contains the same bit pattern as the frame delimiter, it may cause ambiguity. To prevent this, **byte stuffing (also called bit stuffing)** is used to differentiate the delimiter from the actual data.

- **Stuffing:** Extra bits are inserted into the payload when the delimiter pattern occurs in the data.
- **Unstuffing:** These extra bits are removed at the receiver to retrieve the original data.

Process

1. **Bit Stuffing:**
   - In protocols like HDLC, if five consecutive 1s appear in the payload, a 0 is inserted immediately after the five 1s to avoid confusion with the delimiter (01111110).

2. **Bit Unstuffing:**
    - The receiver scans the incoming bit stream. When it detects five consecutive 1s followed by a 0, it removes the 0.

Example

Original Data:

01111110 1111101110111110

Framing with Delimiters:

Add flags to the beginning and end: 01111110 | 01111110 1111101110111110 | 01111110

Bit Stuffing (Sender Side):

The second byte (11111011) has five consecutive 1s, so a 0 is inserted: 01111110 | 01111110 1111100110111110 | 01111110

Transmission Data:

01111110 01111110 1111100110111110 01111110

Bit Unstuffing (Receiver Side):

The receiver removes the stuffed 0 after five consecutive 1s: 01111110 1111101110111110

Final Frame:

Original data restored:
01111110 1111101110111110

3b. Explain the 3 different HDLC frames with diagram.

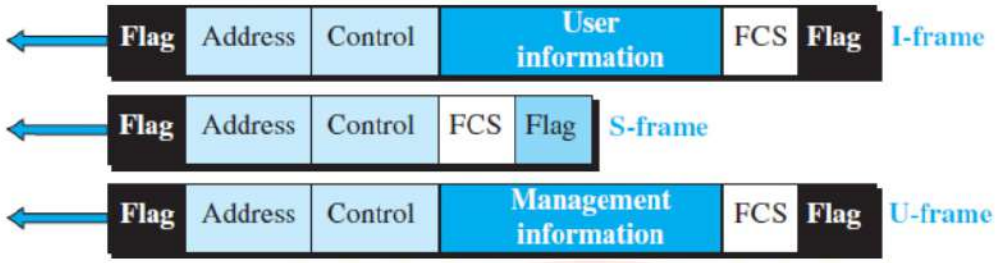
Framing

To provide the flexibility necessary to support all the options possible in the modes and configurations just described, HDLC defines three types of frames:

1. I-frames (information frames) are used to transport user data and control information relating to user data (piggybacking).
2. S-frames (supervisory frames) are used only to transport control information.

V-frames are reserved for system management.

3. U frames (unnumbered frames) Information carried by U-frames is intended for managing the link itself.



Flag field-The flag field of an HDLC frame is an 8-bit sequence with the bit pattern 01111110 that identifies both the beginning and the end of a frame and serves as a synchronization pattern for the receiver.

Address field- The second field of an HDLC frame contains the address of the secondary station. If a primary station created the frame, it contains a ***to*** address. If a secondary creates the frame, it contains *a* ***from*** address. An address field can be 1 byte or several bytes long, depending on the needs of the network.

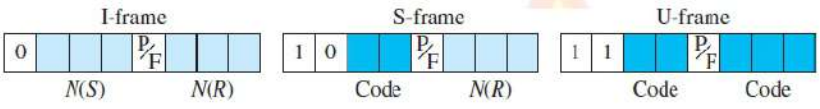
Control field -The control field is a 1- or 2-byte segment of the frame used for flow and error control. The interpretation of bits in this field depends on the frame type.

Information field-The information field contains the user's data from the network layer or management information. Its length can vary from one network to another.

FCS field-The frame check sequence (FCS) is the HDLC error detection field. It can contain either a 2- or 4-byte ITU-T CRC.

Control Field

The control field determines the type of frame and defines its functionality.



Control Field for I-Frames

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.
- The last 3 bits, called *N(R),* correspond to the acknowledgment number when piggybacking is used.
- 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 *poll* when the frame is sent by a primary station to a secondary. It means *final* when the frame is sent by a secondary to a primary.

Control Field for S-Frames

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 is 10, this means the frame is an S-frame. The last 3 bits, called *N(R),* corresponds to the acknowledgment number(ACK) or negative acknowledgment number (NAK) depending on the type of S-frame.The 2 bits called code is used to define the type of S-frame itself. With 2 bits, we can have four types of S-frames, as described below:

1. Receive ready (RR). If the value of the code subfield is 00, it is an RR S-frame. Receive not ready (RNR). If the value of the code subfield is 10, it is an RNR S-frame. The value of *NCR* is the acknowledgment number.
2. Reject (REJ). If the value of the code subfield is 01, it is a REJ S-frame. The value of *NCR)* is the negative acknowledgment number.
3. 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. The value of *N(R)* is the negative acknowledgment number.

Control Field for U-Frames

Unnumbered frames are used to exchange session management and control information between connected devices.

.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.

Figure shows how U-frames can be used for connection establishment and connection release. Node A asks for a connection with a set asynchronous balanced mode (SABM) frame; node B gives a positive response with an unnumbered acknowledgment (UA) frame. After these two exchanges, data can be transferred between the two nodes (not shown in the figure). After data transfer, node A sends a DISC (disconnect) frame to release the

connection; it is confirmed by node B responding with a UA (unnumbered acknowledgment).

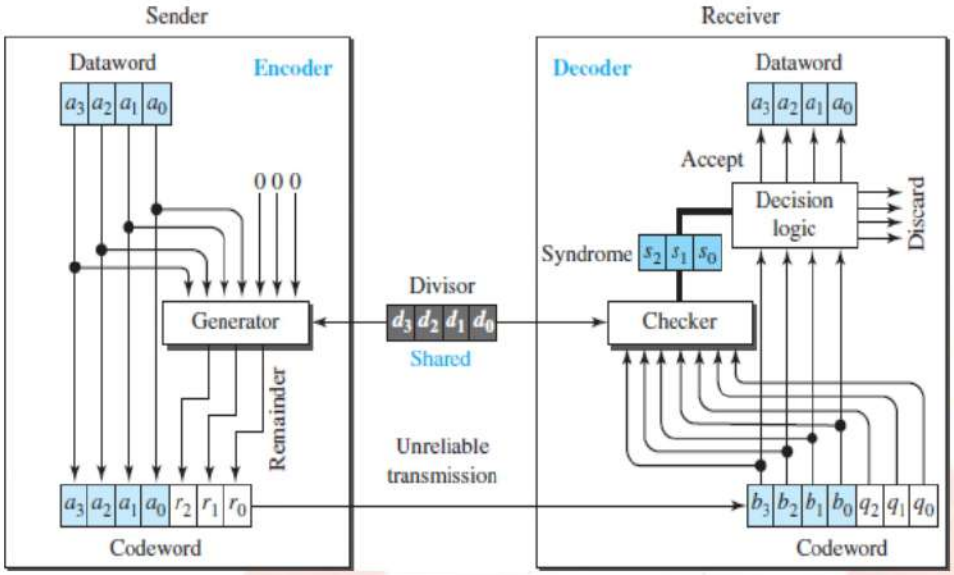
3c. What is the difference between ALOHA and Slotted ALOHA

Aspect	ALOHA	Slotted ALOHA
1. Transmission Timing	Nodes can transmit at any time.	Nodes can transmit only at the start of time slots.
2. Time Synchronization	No synchronization is required.	Requires time synchronization among nodes.
3. Collisions	Collisions can occur at any time.	Collisions can only occur within a time slot.
4. Efficiency	Maximum efficiency is 18.4%.	Maximum efficiency is 36.8%.
5. Complexity	Simpler to implement due to no slots.	Slightly more complex due to slot synchronization.
6. Channel Utilization	Lower due to higher collision probability.	Higher due to reduced collision probability.
7. Application	Suitable for low traffic networks.	Suitable for moderate traffic networks.
8. Throughput	Lower throughput due to unsynchronized access.	Higher throughput as transmissions are synchronized.
9. Collision Probability	High, as multiple nodes can transmit simultaneously.	Reduced, as nodes transmit only at defined slots.
10. Time Wastage	Time is wasted due to collisions anywhere.	Time is wasted only in collided slots.

4a. Explain CRC encoder and decoder for 4 bit data word .

Cyclic Redundancy Check

It is a subset of cyclic codes called the cyclic redundancy check (CRC), which is used in networks such as LANs and WANs.

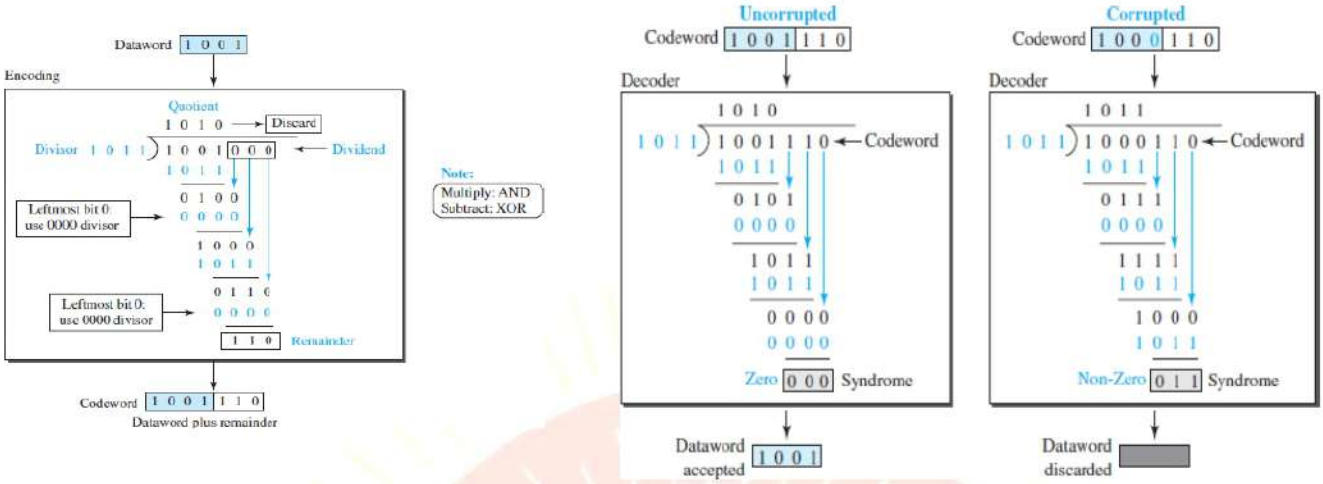


In the **encoder,**

1. The dataword has *k* bits (4 here); the codeword has *n* bits (7 here).
2. The size of the dataword is augmented by adding *n* - *k* (3 here) 0s to the right-hand sideof the word. The n-bit result is fed into the generator.
3. The generator uses a divisor of size *n* - *k* + I (4 here), predefined and agreed upon. The generator divides the augmented dataword by the divisor (modulo-2 division).
4. The quotient of the division is discarded; the remainder $(r_2 r_1 r_0)$ is appended to the dataword to create the codeword.

The decoder,

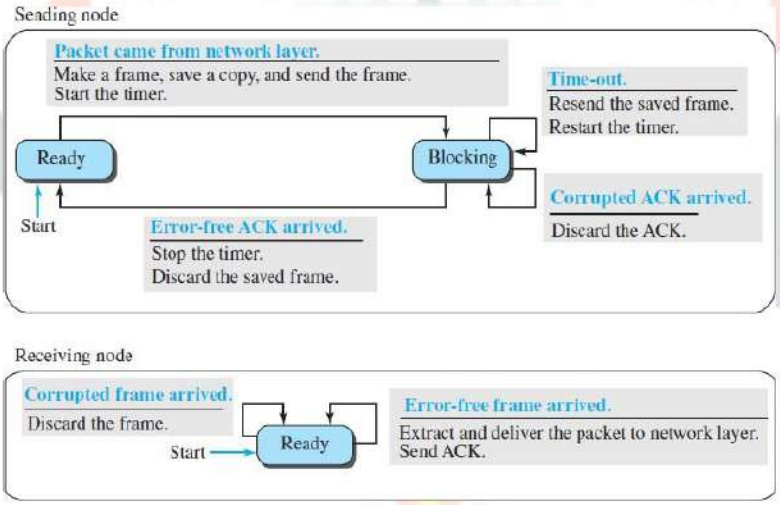
1. Receives the possibly corrupted codeword.
2. A copy of all *n* bits is fed to the checker which is a replica of the generator.
3. The remainder produced by the checker is a syndrome of *n* - *k* (3 here) bits, which is fed to the decision logic analyzer.
4. The analyzer has a simple function. If the syndrome bits are all as, the 4 leftmost bits of the codeword are accepted as the dataword (interpreted as no error); otherwise, the 4 bits are discarded (error).



4b. Explain stop and wait protocol with FSM and Flow diagram.

FSMs

Figure shows the FSMs for our primitive Stop-and-Wait protocol.



Sender States

The sender is initially in the ready state, but it can move between the ready and blocking state ***Ready State.*** When the sender is in this state, it is only waiting for a packet from the network layer. If a packet comes from the network layer, the sender creates a frame, saves a copy of the frame, starts the only timer and sends the frame. The sender then moves to the blocking state. ***Blocking State.*** When the sender is in this state, three events can occur:

a. If a time-out occurs, the sender resends the saved copy of the frame and restarts the timer.
b. If a corrupted ACK arrives, it is discarded.
c. If an error-free ACK arrives, the sender stops the timer and discards the saved copy of the frame. It then moves to the ready state.

Receiver

The receiver is always in the *ready* state. Two events may occur:

a. **If** an error-free frame arrives, the message in the frame is delivered to the network layer and an ACK is sent.

b. If a corrupted frame arrives, the frame is discarded.

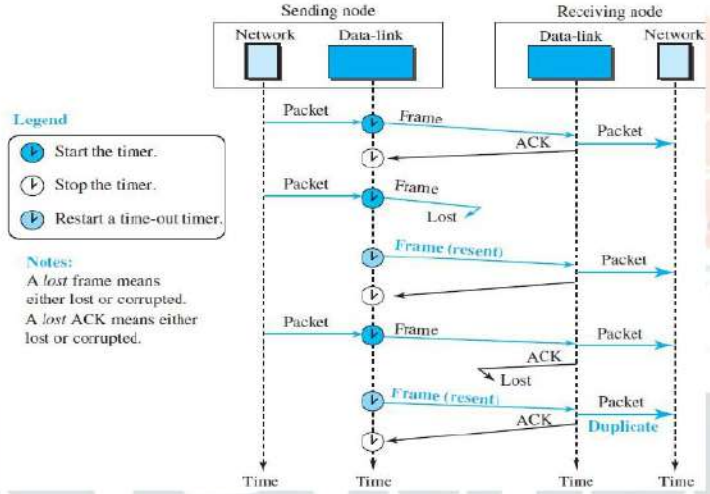


Figure shows an example. The first frame is sent and acknowledged. The second frame is sent, but lost. After time-out, it is resent. The third frame is sent and acknowledged, but the acknowledgment is lost. The frame is resent. However, there is a problem with this scheme. The

Network layer at the receiver site receives two copies of the third packet, which is not right.

Sequence and Acknowledgment Numbers

➔ Duplicate packets, as much as corrupted packets, need to be avoided. As an example, assume we are ordering some item online.
➔ If each packet defines the specification of an item to be ordered, duplicate packets mean ordering an item more than once.
➔ o correct the problem in Example, we need to add **sequence numbers** to the data frames and **acknowledgment numbers** to the ACK frames.

- ➔ Numbering in this case is very simple. Sequence numbers are 0, 1, 0, 1, 0, 1, . . . ; the acknowledgment numbers can also be 1, 0, 1, 0, 1, 0, …
- ➔ An acknowledgment number always defines the sequence number of the next frame to receive.

4c. What is PPP? What are the services provided by PPP?

Point-to-Point Protocol (PPP) is a data link layer communication protocol used to establish a direct connection between two network nodes. It provides a standard method for transporting multi-protocol data over point-to-point links, such as serial cables, phone lines, cellular networks, or fiber optic links.

PPP is by far the most common. PPP provides several services:

1. PPP defines the format of the frame to be exchanged between devices.PPP defines how two devices can negotiate the establishment of the link and the exchange of data.
2. PPP defines how network layer data are encapsulated in the data link frame.
3. PPP defines how two devices can authenticate each other.
4. PPP provides multiple network layer services supporting a variety of network layer protocols.
5. PPP provides connections over multiple links.
6. PPP provides network address configuration. This is particularly useful when a home user needs a temporary network address to connect to the Internet.

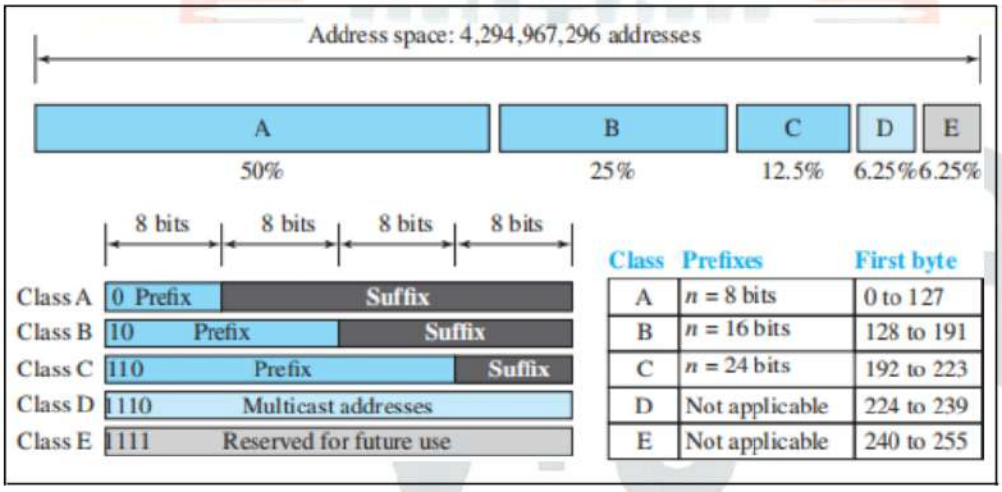
On the other hand, to keep PPP simple, several services are missing:

1. PPP does not provide flow control. A sender can send several frames one after another with no concern about overwhelming the receiver.
2. PPP has a very simple mechanism for error control.
3. PPP does not provide a sophisticated addressing mechanism to handle frames in a multipoint configuration.

5a. Explain classful addressing system with a neat diagram.

Classful Addressing

- When the Internet started, an IPv4 address was designed with a fixed-length prefix, but to accommodate both small and large networks, three fixed-length prefixes were designed instead of one (n = 8, n = 16, and n = 24). The whole address space was divided into five classes (class A, B, C, D, and E). This scheme is referred to as **classful addressing**.
- In class A, the network length is 8 bits, but since the first bit, which is 0, defines the class, we can have only seven bits as the network identifier. This means there are only $2^7 = 128$ networks in the world that can have a class A address.
- In class B, the network length is 16 bits, but since the first two bits, which are $(10)_2$, define the class, we can have only 14 bits as the network identifier. This means there are only 214 = 16,384 networks in the world that can have a class B address.
- All addresses that start with $(110)_2$ belong to class C. In class C, the network length is 24 bits, but since three bits define the class, we can have only 21 bits as the network identifier. This means there are $2^{21} = 2,097,152$ networks in the world that can have a class C address.



Class	Prefixes	First byte
A	n = 8 bits	0 to 127
B	n = 16 bits	128 to 191
C	n = 24 bits	192 to 223
D	Not applicable	224 to 239
E	Not applicable	240 to 255

Class D is not divided into prefix and suffix. It is used for multicast addresses. All addresses that start with 1111 in binary belong to class E. As in Class D, Class E is not divided into prefix and suffix and is used as reserve.

5b. Write Dijkstra's algorithm to compute shortest path with an example.

Dijkstra's Algorithm

The Dijkstra's Algorithm is a greedy algorithm that is used to find the minimum distance between a node and all other nodes in a given graph. Here we can consider node as a router and graph as a network. It uses weight of edge .ie, distance between the nodes to find a minimum distance route.

Algorithm:

1: Mark the source node current distance as 0 and all others as infinity.

2: Set the node with the smallest current distance among the non-visited nodes as the current node.

3: For each neighbor, N, of the current node:

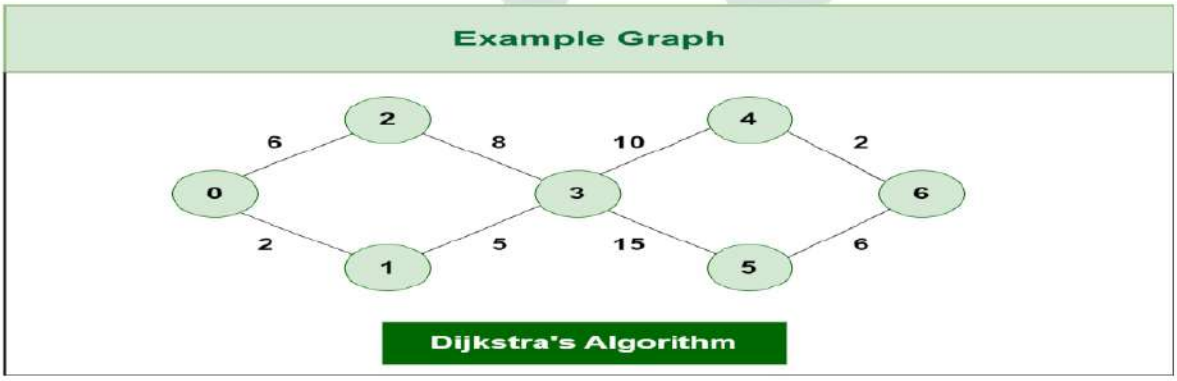
- Calculate the potential new distance by adding the current distance of the current node with the weight of the edge connecting the current node to N.
- If the potential new distance is smaller than the current distance of node N, update N's current distance with the new distance.

4: Make the current node as visited node.

5: If we find any unvisited node, go to step 2 to find the next node which has the smallest current distance and continue this process.

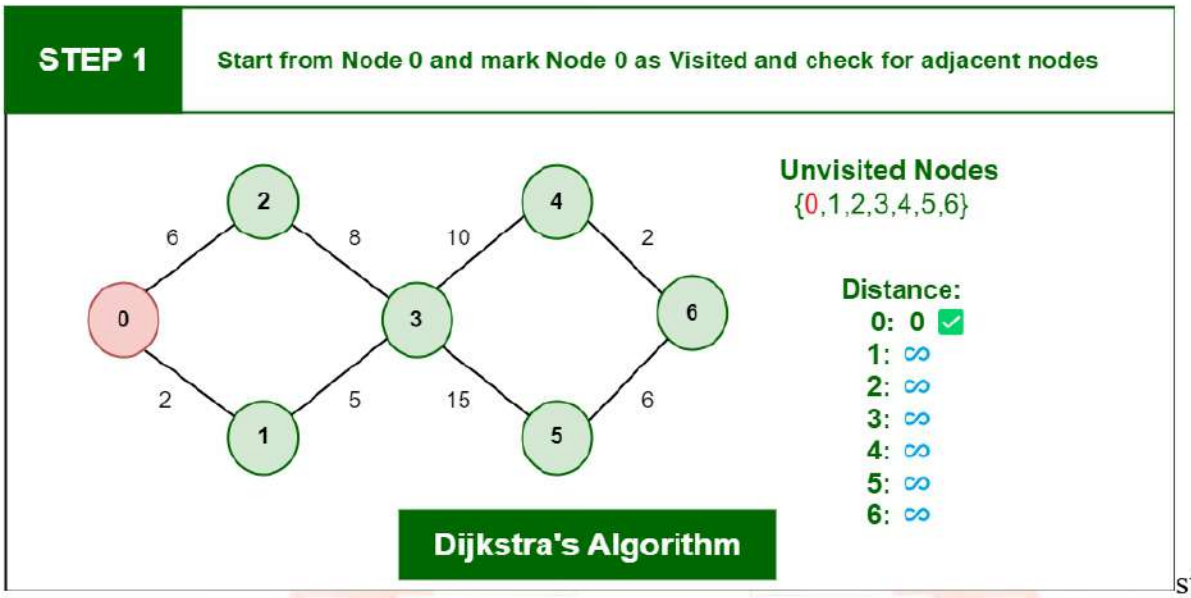
Example:

Consider the graph G:



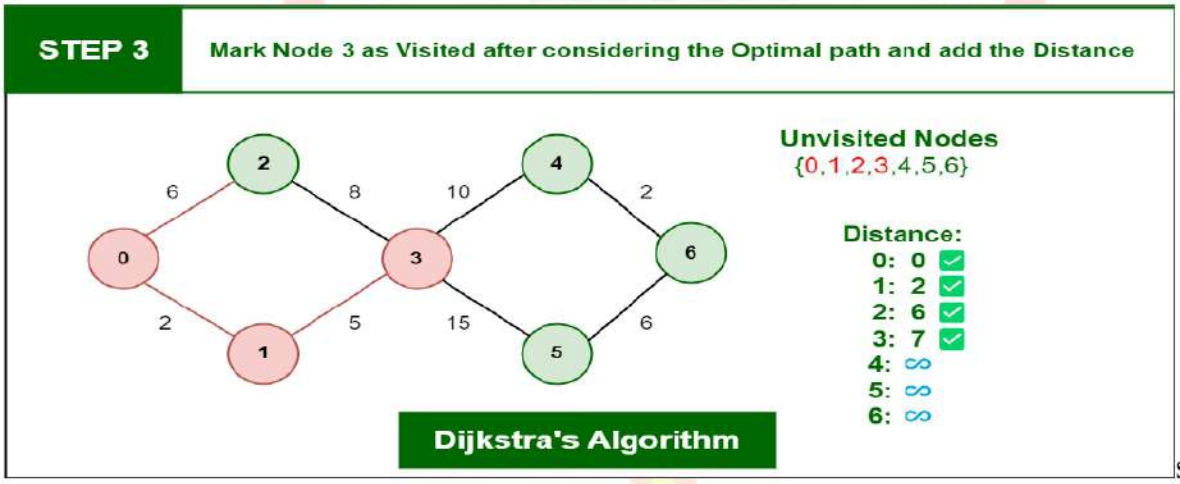
Graph G

Now, we will start normalising graph one by one starting from node 0.



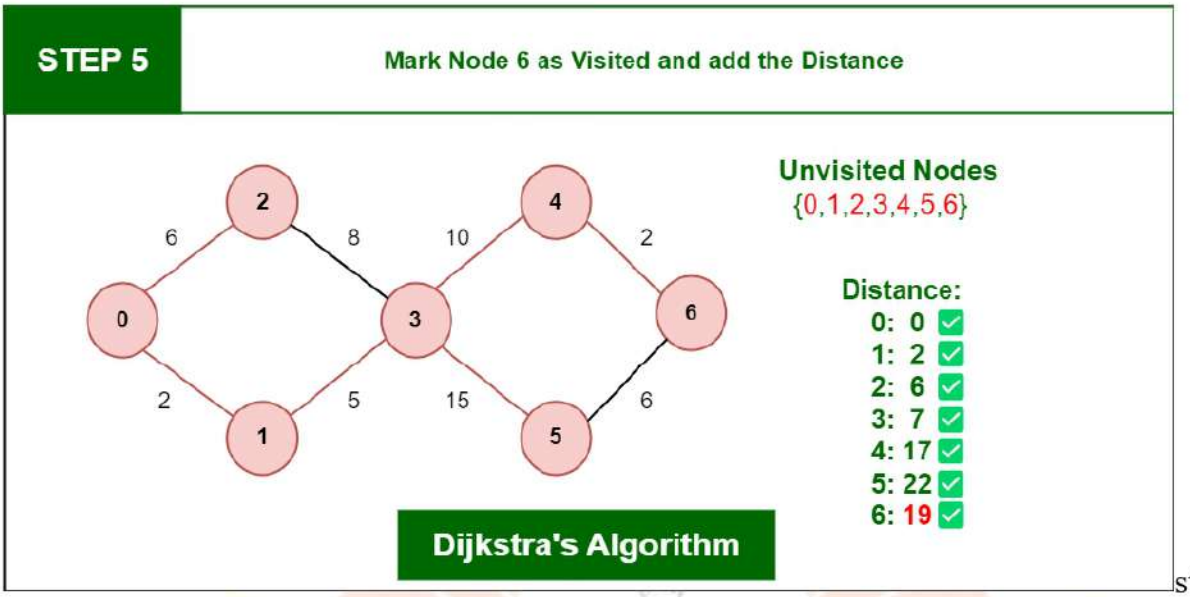
step 1

Nearest neighbour of 0 are 2 and 1 so we will normalize them first .



step 3

Similarly we will normalize other node considering it should not form a cycle and will keep track in visited nodes.



step 5

5c. Why is subnetting used and explain its importance.

What is Subnetting?

Subnetting is the process of dividing a larger IP network (or block) into smaller, more manageable sub-networks (subnets). It involves breaking down an IP address range into segments that can be assigned to smaller network groups.

Why is Subnetting Used?

1. **Efficient IP Address Allocation**:
   - Conserves IP addresses by dividing a large network into smaller networks, ensuring that only the required number of addresses are allocated to each subnet.
2. **Improved Network Performance**:
   - Reduces network traffic by limiting the scope of broadcast traffic to within a single subnet.
   - Helps avoid network congestion in large-scale networks.
3. **Enhanced Security**:

- Isolates sensitive parts of a network, ensuring that access between subnets can be controlled using routers or firewalls.

4. **Simplifies Management**:
   - Makes it easier to manage smaller subnets than one large network.
   - Helps in logically grouping devices, improving organization and troubleshooting.
5. **Facilitates Hierarchical Routing**:
   - Subnetting reduces the size of routing tables, as routers only need to know how to route traffic to a subnet rather than every individual IP address.

Importance of Subnetting

1. **Scalability**:
   - Allows a network to grow efficiently while maintaining its structure and performance.
2. **Broadcast Control**:
   - Broadcast traffic (e.g., ARP requests) is contained within subnets, improving overall network efficiency.
3. **Security Isolation**:
   - Devices in one subnet can be restricted from accessing another subnet unless explicitly allowed.
4. **Cost Savings**:
   - Makes better use of available IP address space, reducing waste and the need to acquire additional IP blocks.
5. **Support for Different Network Sizes**:
   - Allows networks of different sizes to coexist, tailored to specific needs (e.g., smaller subnets for isolated servers, larger subnets for general user devices).

6a. Define and explain routing and forwarding in network layer.

Routing and Forwarding in the Network Layer

1. Routing

Definition:

Routing is the process of determining the best path or route for data packets to travel from the source to the destination across interconnected networks. It is a global decision-making process that occurs at the network level.

Key Points:

- **Goal**: To identify the optimal path between nodes based on specific criteria like shortest distance, least cost, or minimum delay.
- **Dynamic or Static**:
  - **Dynamic Routing**: Uses algorithms to update routing tables automatically based on network conditions (e.g., OSPF, RIP, BGP).
  - **Static Routing**: Manually configured routes that do not change unless explicitly updated.
- **Algorithms**: Implements routing algorithms to compute routes, such as:
  - **Distance Vector Routing**: Shares the cost to reach destinations with neighbors (e.g., RIP).
  - **Link State Routing**: Builds a complete map of the network (e.g., OSPF).
- **Routing Table**: A data structure maintained by routers containing information about routes to various network destinations.

Example:

A router using OSPF (Open Shortest Path First) calculates the shortest path to a destination based on the link-state information it gathers from other routers.

---

2. Forwarding

Definition:

Forwarding is the process of transferring a data packet from an input interface of a router to the appropriate output interface based on the routing table. It is a local, per-packet operation performed at each hop in the network.

Key Points:

- **Goal**: To move packets toward their destination efficiently, using information in the routing table.
- **Decentralized**: Forwarding decisions are made at each router based on the packet's destination address and the router's forwarding table.
- **Forwarding Table**:

- Derived from the routing table.
- Contains mappings of destination addresses or subnets to the next hop or output interface.

- **Fast Process**: Optimized for speed, often implemented in hardware (e.g., using Ternary Content Addressable Memory (TCAM)).

Example:

When a router receives a packet destined for 192.168.1.0/24, it checks its forwarding table to determine the next hop or interface and sends the packet accordingly.

Difference Between Routing and Forwarding

Aspect	Routing	Forwarding
Purpose	Determines the best path for data packets.	Moves packets along the determined path.
Scope	Global or network-wide process.	Local, per-router process.
Frequency	Performed periodically or when network changes.	Performed for every incoming packet.
Focus	Focused on path computation and optimization.	Focused on delivering packets to the next hop.
Data Structure	Routing Table.	Forwarding Table.

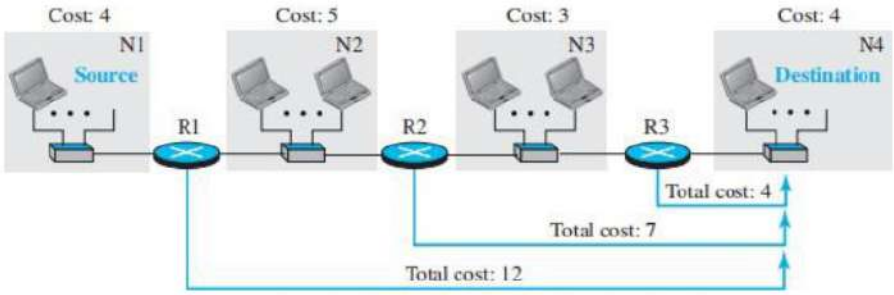
6b. Explain Open Shortest Path First Protocol with example.

Open Shortest Path First (OSPF)

Open Shortest Path First (OSPF) is an intradomain routing protocol like RIP but is based on the link-state routing protocol. It is an open protocol, meaning the specification is publicly available. Unlike RIP, OSPF allows each link to be assigned a weight based on factors such as throughput, round-trip time, or reliability, though administrators can use hop count as a cost. Different Types of Service (TOS) can have different weights for cost calculation.

Metric:
OSPF calculates the cost of reaching a destination from a source by considering link weights, which can vary based on the type of service. This is shown in Figure below, where the total

cost to the destination is calculated by summing the costs of individual links.



Forwarding Tables:

OSPF routers use Dijkstra's algorithm to create forwarding tables by building the shortest-path tree to destinations. The difference between OSPF and RIP forwarding tables is mainly in the cost values. If OSPF uses hop count as its metric, its forwarding tables would be identical to those of RIP. Both protocols determine the best route using shortest-path trees.

Forwarding table for R1

Destination network	Next router	Cost
N1	——	4
N2	——	5
N3	R2	8
N4	R2	12

Forwarding table for R2

Destination network	Next router	Cost
N1	R1	9
N2	——	5
N3	——	3
N4	R3	7

Forwarding table for R3

Destination network	Next router	Cost
N1	R2	12
N2	R2	8
N3	——	3
N4	——	4

Areas:

OSPF is designed for both small and large autonomous systems (AS). In large ASs, flooding link-state packets (LSPs) across the entire network can cause congestion, so OSPF introduces areas to localize LSP flooding. The AS is divided into smaller sections called areas, with one backbone area (Area 0) responsible for inter-area communication.

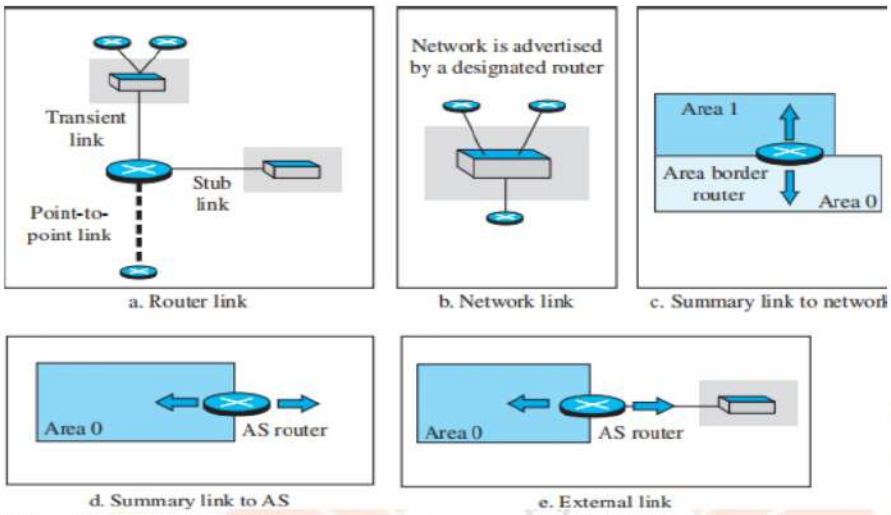
Link-State Advertisement (LSA):

OSPF routers advertise their link states to neighbors for forming a global link-state database (LSDB). Unlike the simple graph model, OSPF distinguishes between different types of nodes and links, requiring various types of advertisements:

Router Link: Announces router existence and its connection to other entities.
Network Link: Advertises the existence of a network, but with no associated cost.
Summary Link to Network: Advertised by area border routers to summarize links between areas.

Summary Link to AS: Announced by AS boundary routers to inform other areas of external AS links.

External Link: Advertises external network routes to the AS.



OSPF Implementation:
OSPF operates at the network layer and uses IP for message propagation. OSPF messages are encapsulated in IP datagrams with a protocol field value of 89. OSPF has two versions, with version 2 being the most widely implemented.

OSPF Algorithm:
OSPF uses a modified link-state routing algorithm. After routers form their shortest-path trees, they create corresponding routing tables. The algorithm also handles OSPF message exchange.

Performance:

Update Messages: OSPF's LSPs are complex and can create heavy traffic in large areas, using considerable bandwidth.
Convergence of Forwarding Tables: OSPF converges relatively quickly once flooding is complete, although Dijkstra's algorithm can take time to run.

Robustness: OSPF is more robust than RIP since routers operate independently after constructing their LSDBs. A failure in one router has less impact on the overall network.

6c. Explain DHCP and its importance?

Dynamic Host Configuration Protocol (DHCP)

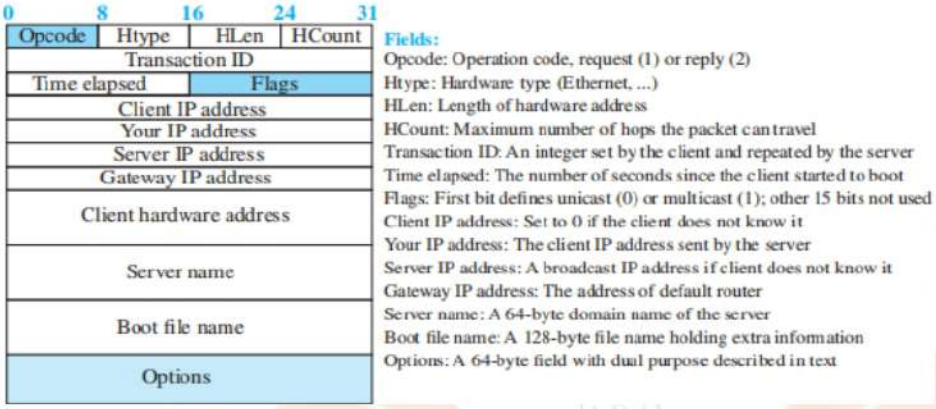
DHCP is an application-layer protocol that automates IP address assignment, using a client-server model. Widely used on the Internet, it is often referred to as a "plug-and-play" protocol.

- **Address Assignment:** DHCP can assign permanent or temporary IP addresses. For

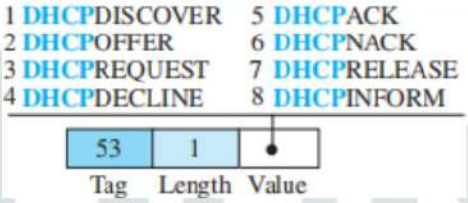
example, ISPs can use DHCP to provide temporary addresses to users, allowing limited IP resources to serve more customers.

- **Additional Information:** DHCP can also provide essential details like the network prefix, default router address, and name server address.

DHCP Message Format



The 64-byte option field in DHCP serves two purposes: carrying additional or vendor-specific information. A special value, called a "magic cookie" (99.130.83.99), helps the client recognize options in the message. The next 60 bytes contain options, structured in three fields: a 1-byte tag, 1-byte length, and variable-length value. The tag field (e.g., 53) can indicate one of the 8 DHCP message types used by the protocol.



DHCP Operation

1. The host creates a **DHCPDISCOVER** message with only a random transaction-ID, as it doesn't know its own IP address or the server's. The message is sent using UDP (source port 68, destination port 67) and broadcasted (source IP: 0.0.0.0, destination IP: 255.255.255.255).
2. The DHCP server responds with a **DHCPOFFER** message, containing the offered IP address, server address, and lease time. This message is sent with the same port numbers but reversed, using a broadcast address so other servers can also offer better options.
3. The host selects the best offer and sends a **DHCPREQUEST** to the server. The message includes the chosen IP address and is sent as a broadcast (source: new client IP, destination: 255.255.255.255) to notify other servers that their offers were declined.
4. The selected server responds with a **DHCPACK** if the IP address is valid, completing the process. If the IP address is unavailable, a **DHCPNACK** is sent, and the host must restart the process.

Two Well-Known Ports

DHCP uses well-known ports (68 for the client, 67 for the server) to avoid conflicts with other services using ephemeral ports. Since DHCP responses are broadcast, using a well-

known port (68) ensures the DHCP response is only delivered to the correct client, preventing confusion with other clients (e.g., DAYTIME) that might be using the same temporary port. If two DHCP clients are running simultaneously (e.g., after a power failure), they are distinguished by their unique transaction IDs.

Using FTP

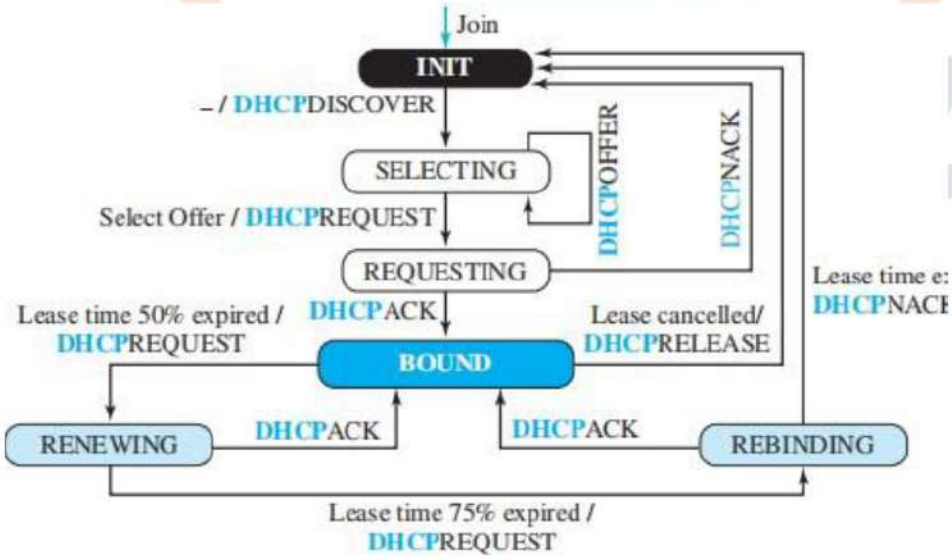
The DHCPACK message includes a pathname to a file with additional information (e.g., DNS server address). The client uses FTP to retrieve this information.

Error Control

Since DHCP relies on unreliable UDP, it ensures error control by requiring UDP checksums and using timers with a retransmission policy. To avoid traffic congestion (e.g., after a power failure), clients use random timers for retransmission.

Transition States

The operation of the DHCP were very simple. To provide dynamic address allocation, the DHCP client acts as a state machine that performs transitions from one state to another depending on the messages it receives or sends.



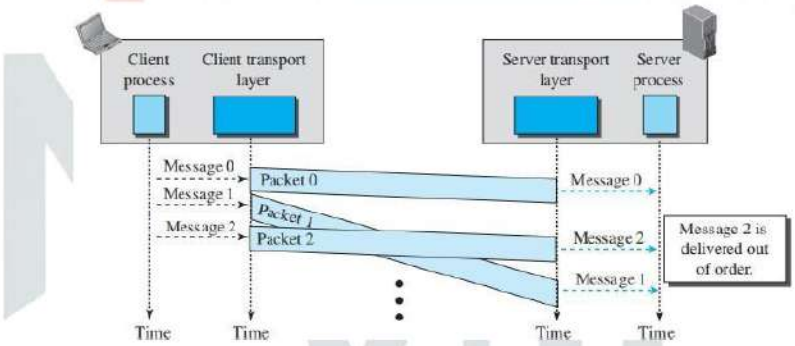
1. **INIT state**: The client starts here and sends a *Discover* message to find a DHCP server.
2. **SELECTING state**: After receiving one or more *Offer* messages, the client selects one offer.
3. **REQUESTING state**: The client sends a *Request* message to the selected server and waits.

4. **BOUND state**: If the server responds with an *ACK* message, the client uses the assigned IP address.
5. **RENEWING state**: When 50% of the lease time is expired, the client tries to renew the lease by contacting the server. If successful, it stays in the BOUND state.
6. **REBINDING state**: If the lease is 75% expired and no response is received, the client tries to contact any DHCP server. If the server responds, it stays BOUND; otherwise, it goes back to INIT to request a new IP.

7a. Draw the FSM diagrams for connectionless and connected oriented services offered by transport layer.

Connectionless Service

In a **connectionless service**, the source application divides its message into chunks of data and sends them to the **transport layer**, which treats each chunk independently. There is no relationship between the chunks, so they may arrive out of order at the destination. For example, a client might send three chunks (0, 1, and 2), but due to delays, the server could receive them out of order (0, 2, 1), as shown in **Figure** . This could result in a garbled message. If one packet is lost, since there is no numbering or coordination between the transport layers, the receiving side won't know and will deliver incomplete data. This lack of **flow control**, **error control**, and **congestion control** makes the system inefficient.

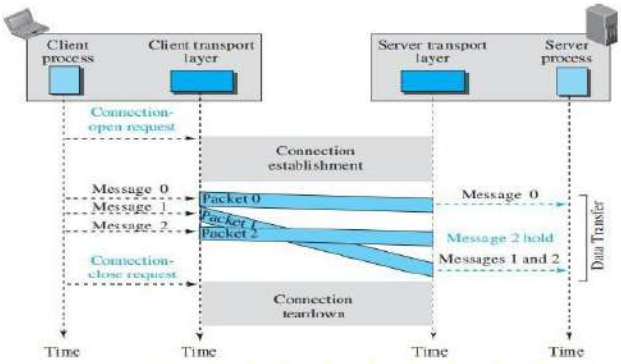


Connection-Oriented Service

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

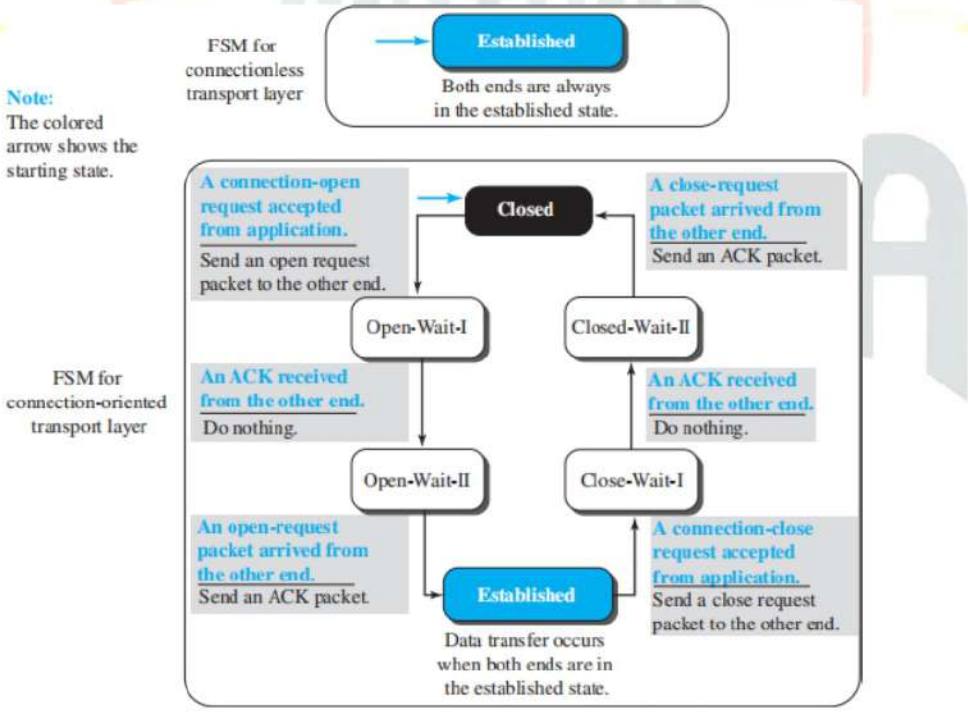
Figure shows the connection establishment, data-transfer, and tear-down phases in a connection-oriented service at the transport layer.

We can implement flow control, error control, and congestion control in a connection oriented protocol.



The behavior of a transport-layer protocol, both when it provides a connectionless and when it provides a connection-oriented protocol, can be better shown as a **finite state machine (FSM).** Using this tool, each transport layer (sender or receiver) is taught as a machine with a finite number of states.

Every FSM must have an initial state, which is where the machine starts when it turns on. In diagrams, rounded rectangles are used to represent states, colored text indicates events, and black text shows actions. A horizontal line or a slash separates the event from the action, and arrows depict the transition to the next state.



In a **connectionless transport layer**, the FSM has only one state: the established state. The machines on both the client and server sides remain in the established state, always ready to send and receive transport-layer packets.

A **connection-oriented FSM** requires several states for **establishment** and **termination**:

- **Closed State**: The FSM starts here when there is no connection. It stays in this state until it receives an **open request** from the local process.
- The machine sends an **open request packet** to the remote transport layer and transitions to **open-wait-I**.
- When the acknowledgment is received, it moves to **open-wait-II**. At this point, a **unidirectional connection** is established.

7b. List the services and applications of UDP.

UDP Services

Process-to-Process Communication

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

Connectionless Services

UDP provides a *connectionless service.* This means that each user datagram sent by UDP is an independent datagram. There is no relationship between the different user datagrams even if they are coming from the same source process and going to the same destination program. The user datagrams are not numbered. There is no connection establishment and no connection termination. This means that each user datagram can travel on a different path.

Flow Control

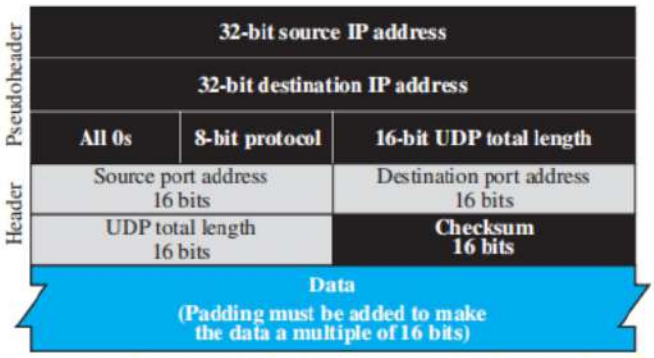
UDP is a very simple protocol. There is no *flow control,* and hence no window mechanism. The receiver may overflow with incoming messages. The lack of flow control means that the process using UDP should provide for this service, if needed.

Error Control

There is no *error control* mechanism in UDP except for the checksum. This means that the sender does not know if a message has been lost or duplicated. When the receiver detects an error through the checksum, the user datagram is silently discarded.

Checksum

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



If the checksum does not include the pseudoheader, a user datagram may arrive safe and sound. However, if the IP header is corrupted, it may be delivered to the wrong host. The protocol field is added to ensure that the packet belongs to UDP, and not to TCP. The value of the protocol field for UDP is 17. If this value is changed during transmission, the checksum calculation at the receiver will detect it and UDP drops the packet. It is not delivered to the wrong protocol.

Congestion Control

Since UDP is a connectionless protocol, it does not provide congestion control. UDP assumes that the packets sent are small and sporadic and cannot create congestion in the network.

Encapsulation and Decapsulation

To send a message from one process to another, the UDP protocol encapsulates and decapsulates messages.

Queuing

In UDP, queues are associated with ports. At the client site, when a process starts, it requests a port number from the operating system. Some implementations create both an incoming and an outgoing queue associated with each process. Other implementations create only an incoming queue associated with each process.

Multiplexing and Demultiplexing

In a host running a TCP/IP protocol suite, there is only one UDP but possibly several processes that may want to use the services of UDP. To handle this situation, UDP multiplexes and demultiplexes.

Applications of UDP

1. **Real-Time Communication:**
    - UDP is ideal for time-sensitive applications where speed is more critical than reliability.
    - Examples:
        - Voice over IP (VoIP)

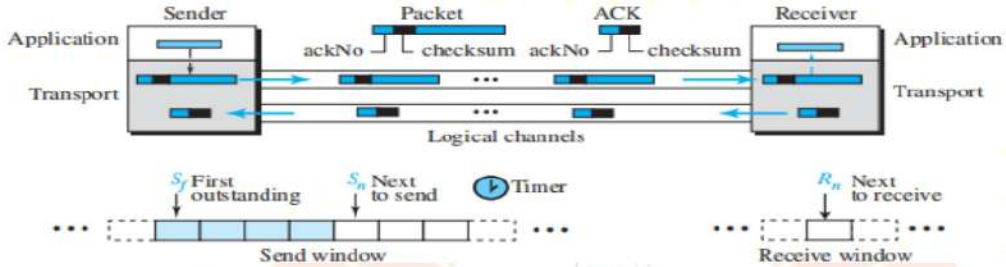
- Video conferencing (e.g., Zoom, WebRTC)
- Online gaming

2. **Streaming Media:**
   - Used in live audio/video streaming, where occasional packet loss is acceptable but low latency is essential.
   - Examples:
     - IPTV
     - YouTube live streams
     - Netflix (in some cases for adaptive streaming)
3. **Broadcast and Multicast Services:**
   - UDP supports broadcast and multicast communication, which TCP does not.
   - Examples:
     - Routing protocols (e.g., RIP, OSPF)
     - Multimedia applications (e.g., live TV distribution)
4. **DNS and DHCP:**
   - Domain Name System (DNS) queries and Dynamic Host Configuration Protocol (DHCP) operations use UDP for efficiency since the communication typically involves small amounts of data.
   - Examples:
     - Resolving domain names to IP addresses.
     - Assigning IP addresses to devices on a network.
5. **Online Gaming:**
   - Real-time multiplayer games rely on UDP to minimize latency and ensure fast communication between players.
   - Examples:
     - First-person shooters
     - Real-time strategy games

7c. Explain Go-Bank-N protocol working.

Go-Back-*N* Protocol (GBN)

The key to Go-back-*N* is that we can send several packets before receiving acknowledgments, but the receiver can only buffer one packet. We keep a copy of the sent packets until the acknowledgments arrive. Figure shows the outline of the protocol. Note that several data packets and acknowledgments can be in the channel at the same time.



Sequence Numbers

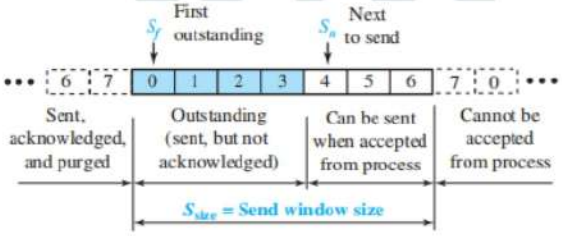
The sequence numbers are modulo $2m$, where m is the size of the sequence number field in bits.

Acknowledgment Numbers

An acknowledgment number in this protocol is cumulative and defines the sequence number of the next packet expected. For example, if the acknowledgment number (ackNo) is 7, it means all packets with sequence number up to 6 have arrived, safe and sound, and the receiver is expecting the packet with sequence number 7.

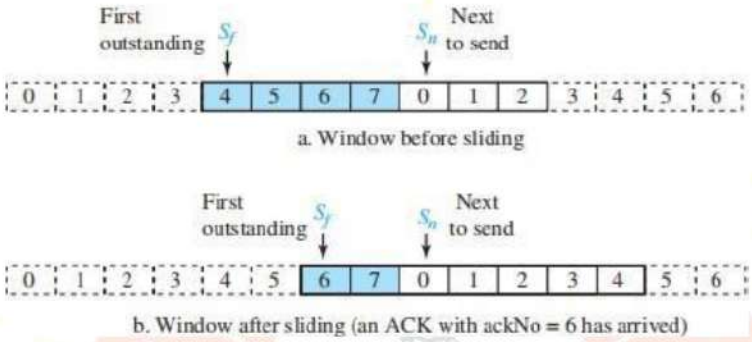
Send Window

The send window is an imaginary box covering the sequence numbers of the data packets that can be in transit or can be sent. In each window position, some of these sequence numbers define the packets that have been sent; others define those that can be sent. The maximum size of the window is $2m - 1$.



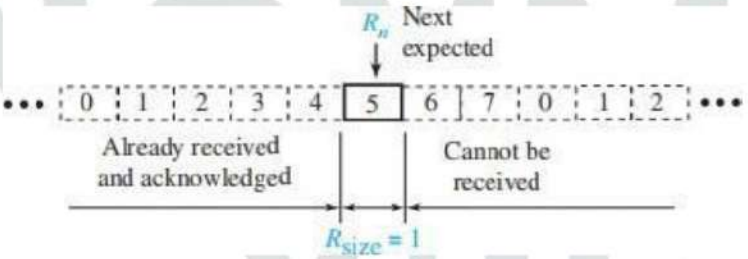
The send window at any time divides sequence numbers into four regions. The first region includes acknowledged packets, which the sender no longer tracks. The second region contains outstanding packets that have been sent but have an unknown status. The third region defines sequence numbers for packets that can be sent but for which data hasn't been received from the application layer. The fourth region consists of sequence numbers that cannot be used until the window slides.

The window itself is an abstraction; three variables define its size and location at any time. We call these variables S_f (send window, the first outstanding packet), *Sn* (send window, the next packet to be sent), and *S*size (send window, size). The variable S_f defines the sequence number of the first (oldest) outstanding packet. The variable *Sn* holds the sequence number that will be assigned to the next packet to be sent. Finally, the variable *S*size defines the size of the window, which is fixed in our protocol.



Receive Window

The receive window ensures correct packet reception and acknowledgment. In Go-Back-N, its size is always 1, as the receiver expects a specific packet. Out-of-order packets are discarded and must be resent. Only packets matching the expected sequence number, Rn, are accepted and acknowledged. The window slides by one slot upon receiving the correct packet, with Rn updated as (Rn + 1) modulo 2m.



Timers

Although there can be a timer for each packet that is sent, in our protocol we use only one. The reason is that the timer for the first outstanding packet always expires first. We resend all outstanding packets when this timer expires.

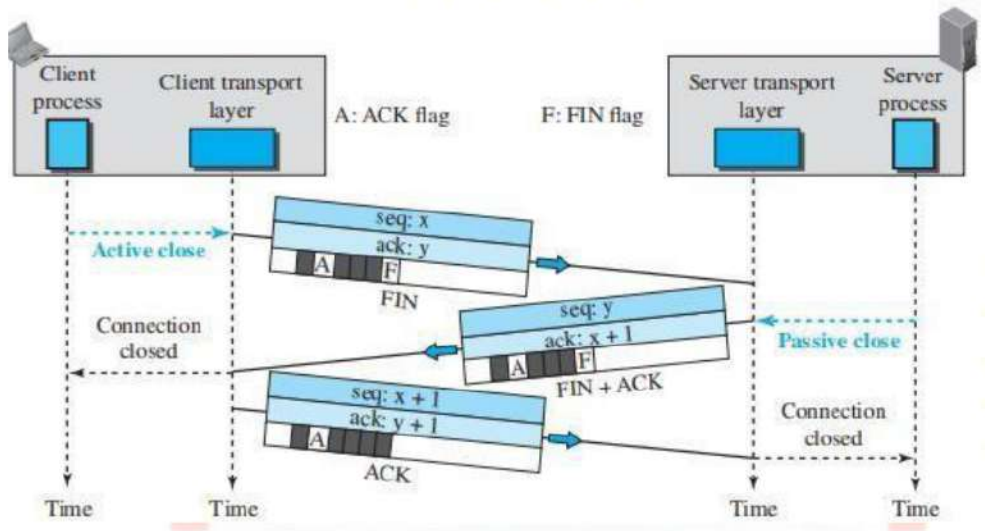
Resending packets

When the timer expires, the sender resends all outstanding packets. For example, suppose the sender has already sent packet 6 (*Sn* = 7), but the only timer expires. If *Sf* = 3, this means that packets 3, 4, 5, and 6 have not been acknowledged; the sender goes back and resends packets 3, 4, 5, and 6. That is why the protocol is called *Go-Back*-N. On a time-out, the machine goes back *N* locations and resends all packets.

8a. Explain connection establishment of TCP using 3-way handshaking.

Three-Way Handshaking

Most implementations today allow *three-way handshaking* for connection termination.



1. In this situation, the client TCP, after receiving a close command from the client process, sends the first segment, a FIN segment in which the FIN flag is set.
2. The server TCP, after receiving the FIN segment, informs its process of the situation and sends the second segment, a FIN + ACK segment, to confirm the receipt of the FIN segment from the client and at the same time to announce the closing of the connection in the other direction.
3. The client TCP sends the last segment, an ACK segment, to confirm the receipt of the FIN segment from the TCP server. This segment contains the acknowledgment number, which is one plus the sequence number received in the FIN segment from the server. This segment cannot carry data and consumes no sequence numbers.

8b. Explain TCP Congestion control.

. TCP Congestion Control

TCP uses different policies to handle the congestion in the network.

Congestion Window

To control the number of segments to transmit, TCP uses another variable called a *congestion window, cwnd,* whose size is controlled by the congestion situation in the network. The *cwnd* variable and the *rwnd* variable together define the size of the send window in TCP. The first is related to the congestion in the middle (network); the second is related to the congestion at the end. The actual size of the window is the minimum of these two.

Actual window size = minimum (*rwnd*, *cwnd*)

Congestion Detection

TCP detects network congestion through two main events: time-outs and the receipt of three duplicate ACKs. A time-out occurs when the sender does not receive an ACK for a segment or group of segments before the timer expires, signaling the likelihood of severe congestion and possible segment loss. On the other hand, receiving three duplicate ACKs (four identical ACKs) indicates that one segment is missing, but others have been received, suggesting mild congestion or network recovery. While early TCP versions like **Tahoe** treated both events the same, later versions like **Reno** distinguish between the two, with time-outs indicating stronger congestion than duplicate ACKs. TCP relies on ACKs as the only feedback to detect congestion, where missing or delayed ACKs serve as indicators of network conditions.

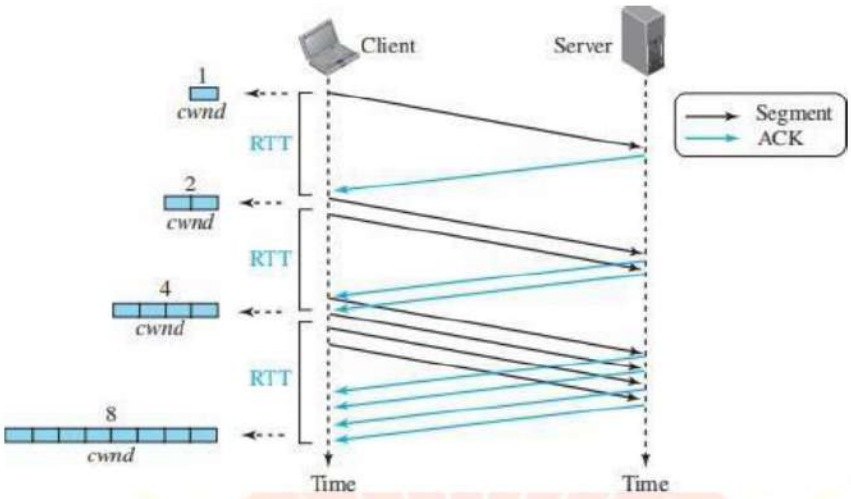
Congestion Policies

TCP's general policy for handling congestion is based on three algorithms: slow start, congestion avoidance, and fast recovery.

Slow Start: Exponential Increase

The slow-start algorithm begins with the congestion window (cwnd) set to one maximum segment size (MSS) and increases by one MSS for each received acknowledgment. The MSS is negotiated during connection establishment. Despite the name, the algorithm grows exponentially. Initially, the sender transmits one segment, and upon receiving the ACK, the cwnd increases by 1, allowing the sender to transmit two segments. Each acknowledgment further increases cwnd, doubling the number of segments the sender can transmit, resulting in rapid growth as long as no congestion is detected. The size of the congestion window in this algorithm is a function of the number of ACKs arrived and can be determined as follows.

If an ACK arrives, *cwnd* = *cwnd* + 1.



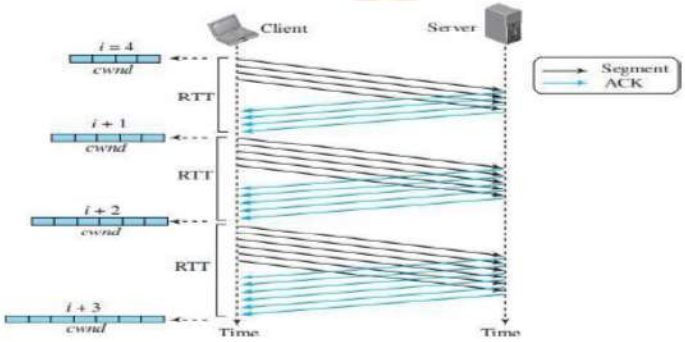
If we look at the size of the *cwnd* in terms of round-trip times (RTTs), we find that the growth rate is exponential in terms of each round trip time, which is a very aggressive approach:

Start	→	$cwnd = 1 \rightarrow 2^0$
After 1 RTT	→	$cwnd = cwnd + 1 = 1 + 1 = 2 \rightarrow 2^1$
After 2 RTT	→	$cwnd = cwnd + 2 = 2 + 2 = 4 \rightarrow 2^2$
After 3 RTT	→	$cwnd = cwnd + 4 = 4 + 4 = 8 \rightarrow 2^3$

A slow start cannot continue indefinitely. There must be a threshold to stop this phase. The sender keeps track of a variable named *ssthresh* (slow-start threshold). When the size of the window in bytes reaches this threshold, slow start stops and the next phase starts.

Congestion Avoidance: Additive Increase

The slow-start algorithm in TCP increases the size of the congestion window (cwnd) exponentially, which can lead to congestion. To mitigate this, TCP employs the congestion avoidance algorithm, which increases cwnd additively rather than exponentially. When cwnd reaches the slow-start threshold (i), the slow-start phase ends, and the additive phase begins. In this algorithm, for each set of acknowledged segments (the entire "window"), cwnd is incremented by one. A window represents the number of segments sent during a round-trip time (RTT).



For example, if the sender starts with cwnd = 4, it can send four segments. Upon receiving four ACKs, one segment slot opens up, increasing cwnd to 5. After sending five segments and receiving five acknowledgments, cwnd increases to 6, and so on.
The congestion window can be expressed as:

If an ACK arrives, *cwnd* = *cwnd* + (1/*cwnd*).

The window increases by (1/*cwnd*) portion of the Maximum Segment Size (MSS) in bytes. Thus, all segments in the previous window must be acknowledged to increase cwnd by 1 MSS byte. This results in a linear growth rate of cwnd with each round-trip time (RTT), making it a more conservative approach than slow-start.

Start	→	$cwnd = i$
After 1 RTT	→	$cwnd = i + 1$
After 2 RTT	→	$cwnd = i + 2$
After 3 RTT	→	$cwnd = i + 3$

Fast Recovery

The **fast-recovery** algorithm is optional in TCP. The old version of TCP did not use it, but the new versions try to use it. It starts when three duplicate ACKs arrive, which is interpreted as light congestion in the network. Like congestion avoidance, this algorithm is also an additive increase, but it increases the size of the congestion window when a duplicate ACK arrives (after the three duplicate ACKs that trigger the use of this algorithm). We can say

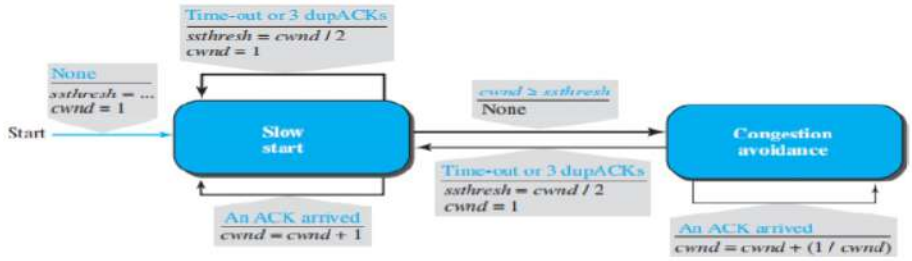
If a duplicate ACK arrives, *cwnd* == *cwnd* + (1 / *cwnd)*

Policy Transition

We discussed three congestion policies in TCP. Now the question is when each of these policies is used and when TCP moves from one policy to another. To answer these questions, we need to refer to three versions of TCP: Taho TCP, Reno TCP, and New Reno TCP.

Taho TCP

The early TCP, known as *Taho TCP,* used only two different algorithms in their congestion policy: *slow start* and *congestion avoidance*.



Taho TCP treats the two signs used for congestion detection, time-out and three duplicate ACKs, in the same way. In this version, when the connection is established, TCP starts the slow-start algorithm and sets the ssthresh variable to a pre-agreed value(normally a multiple of MSS) and the cwnd to 1 MSS.
If congestion is detected (occurrence of time-out or arrival of three duplicate ACKs), TCP immediately interrupts this aggressive growth and restarts a new slow start algorithm by limiting the threshold to half of the current *cwnd* and resetting the congestion window to 1
If no congestion is detected while reaching the threshold, TCP learns that the ceiling of its ambition is reached; it should not continue at this speed. It moves to the congestion avoidance state and continues in that state.
In the congestion-avoidance state, the size of the congestion window is increased by 1 each time a number of ACKs equal to the current size of the window has been received.

Reno TCP

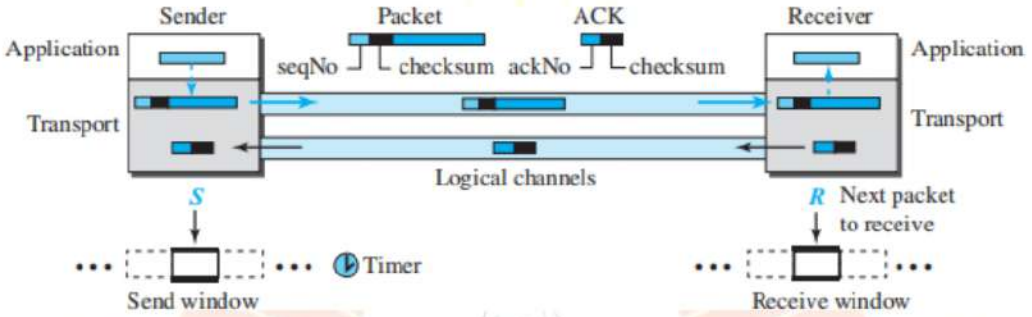
A newer version of TCP, called *Reno TCP,* added a new state to the congestion-control FSM, called the fast-recovery state. This version treated the two signals of congestion, time-out and the arrival of three duplicate ACKs, differently. In this version, if a time-out occurs, TCP moves to the slow-start state (or starts a new round if it is already in this state); on the other hand, if three duplicate ACKs arrive, TCP moves to the fast-recovery state and remains there as long as more duplicate ACKs arrive.
When TCP enters the fast-recovery state, three major events may occur. If duplicate ACKs continue to arrive, TCP stays in this state, but the *cwnd* grows exponentially. If a time-out occurs, TCP assumes that there is real congestion in the network and moves to the slow-start state. If a new (non duplicate) ACK arrives, TCP moves to the congestion-avoidance state, but deflates the size of the *cwnd* to the *ssthresh* value, as though the three duplicate ACKs have not occurred, and transition is from the slow-start state to the congestion-avoidance state.

8c. Explain with example stop and wait protocol.

Stop-and-Wait Protocol

- ➔ Stop-and-Wait is a connection-oriented protocol, which uses both **flow and error control**.
- ➔ Both the sender and the receiver use a **sliding window of size 1**. The sender sends one packet at a time and waits for an acknowledgment before sending the next one.
- ➔ To detect corrupted packets, we need to add a **checksum** to each data packet. When a packet arrives at the receiver site, it is checked. If its checksum is incorrect, the packet is corrupted and silently discarded.

- ➔ The silence of the receiver is a signal for the sender that a packet was either corrupted or lost. Every time the sender sends a packet, it starts a timer.
- ➔ If an acknowledgment arrives before the timer expires, the timer is stopped and the sender sends the next packet (if it has one to send).
- ➔ If the timer expires, the sender resends the previous packet, assuming that the packet was either lost or corrupted. This means that the sender needs to keep a copy of the packet until its acknowledgment arrives.



Sequence Numbers

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

Acknowledgment Numbers

Since the sequence numbers must be suitable for both data packets and acknowledgments, we use this convention: The acknowledgment numbers always announce the sequence number of the *next packet expected* by the receiver.

FSMs

Since the protocol is a connection-oriented protocol, both ends should be in the *established* state before exchanging data packets. The states are actually nested in the *established* state.

Sender

The sender is initially in the ready state, but it can move between the ready and blocking state. The variable S is initialized to 0.

Ready state. When the sender is in this state, it is only waiting for one event to occur. If a request comes from the application layer, the sender creates a packet with the sequence number set to S. A copy of the packet is stored, and the packet is sent. The sender then starts the only timer. The sender then moves to the blocking state.

Blocking state. When the sender is in this state, three events can occur:

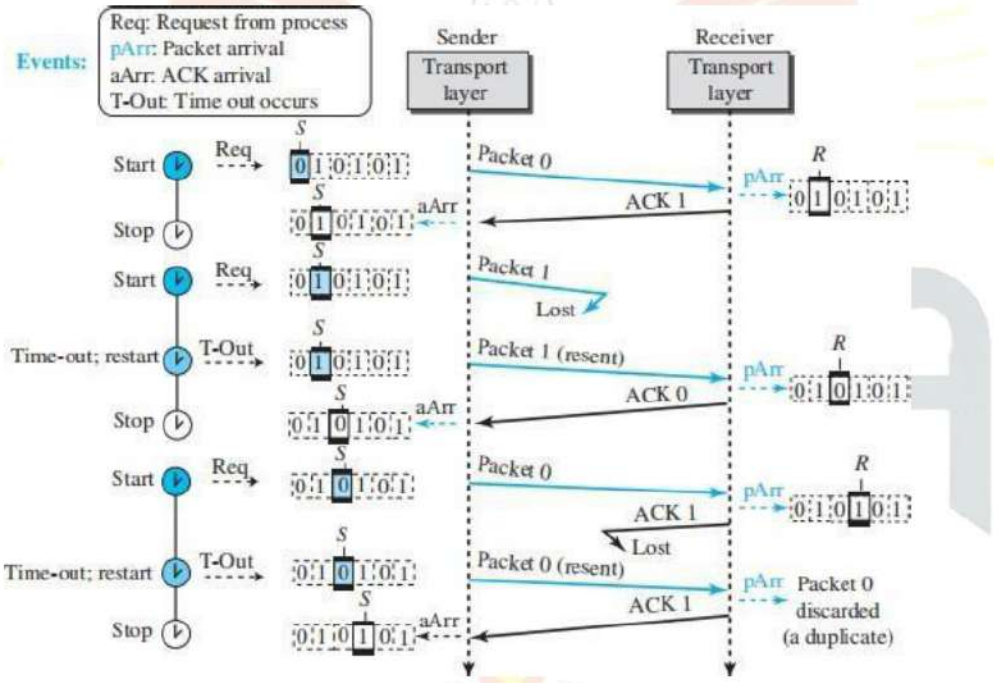
a. If an error-free ACK arrives with the ackNo related to the next packet to be sent, which means ackNo = (S + 1) modulo 2, then the timer is stopped. The window slides, $S = (S + 1)$ modulo 2. Finally, the sender moves to the ready state.
b. If a corrupted ACK or an error-free ACK with the ackNo ≠ (S + 1) modulo 2 arrives, the ACK

is discarded.
c. If a time-out occurs, the sender resends the only outstanding packet and restarts the timer.

Receiver

The receiver is always in the *ready* state. Three events may occur:
a. If an error-free packet with seqNo = *R* arrives, the message in the packet is delivered to the application layer. The window then slides, *R* = (*R* + 1) modulo 2. Finally an ACK with ackNo = *R* is sent.
b. If an error-free packet with seqNo ≠ *R* arrives, the packet is discarded, but an ACK with ackNo = *R* is sent.
If a corrupted packet arrives, the packet is discarded.

Figure shows an example of the Stop-and-Wait protocol. Packet 0 is sent and acknowledged. Packet 1 is lost and resent after the time-out. The resent packet 1 is acknowledged and the timer stops. Packet 0 is sent and acknowledged, but the acknowledgment is lost. The sender has no idea if the packet or the acknowledgment is lost, so after the time-out, it resends packet 0, which is acknowledged.



9a. Explain Standard and Non-Standard Application Layer Protocols.

Standard and Nonstandard Protocols

For smooth Internet operation, the protocols in the first four layers of the TCP/IP suite need to be standardized and documented. These standard protocols are typically part of operating systems like Windows or UNIX. However, the application-layer protocols can be both standard and nonstandard for added flexibility.

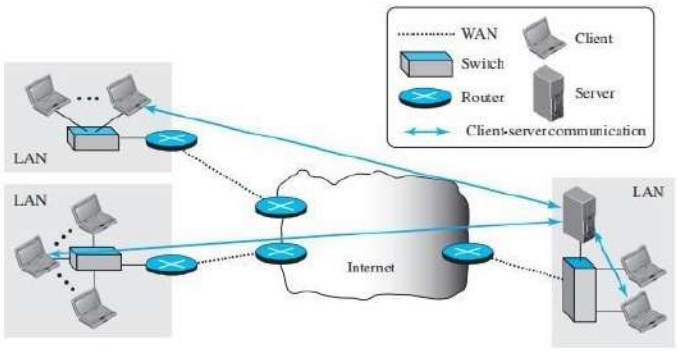
Standard Application-Layer Protocols-Standard application-layer protocols are well documented and standardized by Internet authorities. They are widely used in daily Internet interactions.

Nonstandard Application-Layer Protocols-Programmers can also create nonstandard (or proprietary) application-layer programs by developing two programs that provide services through the transport layer.

9b. Differentiate client server paradigm and peer-to-peer paradigm.

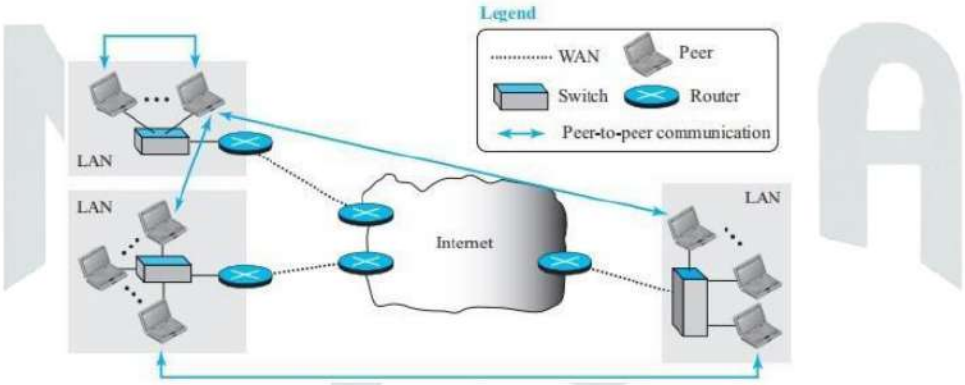
Traditional Paradigm: Client-Server

- In client-server architecture, there is an **always-on** host, called the **server**, which provides services when it receives requests from many other hosts, called **clients**.
- **Example:** In Web application Web server services requests from browsers running on client hosts. When a Web server receives a request for an object from a client host, it responds by sending the requested object to the client host.
- In client-server architecture, clients do not directly communicate with each other.
- The server has a fixed, well-known address, called an **IP address**. Because the server has a fixed, well-known address, and is always on, a client can always contact the server by sending a packet to the server's IP address.
- Some of the better-known applications with client-server architecture include the Web, FTP, Telnet, and e-mail.
- The most popular Internet services—such as search engines (e.g., Google and Bing), Internet commerce (e.g., Amazon and e-Bay), Web-based email (e.g., Gmail and Yahoo Mail), social networking (e.g., Facebook and Twitter)— employ one or more data centers.



New Paradigm: Peer-to-Peer

- In P2P architecture, there is minimal dependence on dedicated servers in data centers.
- The application employs direct communication between pairs of intermittently connected hosts, called peers.
- The peers are not owned by the service provider, but are instead desktops and laptops controlled by users, with most of the peers residing in homes, universities, and offices.
- Many of today's most popular and traffic-intensive applications are based on P2P architectures. These applications include file sharing (e.g., BitTorrent), Internet Telephony (e.g., Skype), and IPTV (e.g., Kankan and PPstream).



9c. Differentiate between Persistent and Non Persistent connection in HTTP.

In HTTP, **Persistent** and **Non-Persistent** connections refer to how connections between a client (e.g., browser) and a server are managed during data transfer. Here are the key differences:

Aspect	Persistent Connection	Non-Persistent Connection
Definition	The connection between client and server remains open for multiple requests and responses.	A new connection is established for each request/response pair.
Efficiency	Reduces overhead by reusing the same connection for multiple requests, improving efficiency.	Higher overhead due to repeated connection setup and teardown.
Connection Handling	The Connection: keep-alive header is used to keep the connection open.	The connection closes automatically after a single request/response cycle.
Performance	Faster for multiple requests, as there is no need to repeatedly establish and close connections.	Slower due to the additional time required to set up and close connections for each request.
Use Case	Suitable for modern web applications with multiple resources (e.g., images, scripts).	Suitable for simple, single-resource requests or older protocols.
Resource Utilization	More efficient in terms of network resources and server processing.	Consumes more resources due to frequent connection handling.
HTTP Versions	Default in HTTP/1.1 and later versions unless explicitly disabled.	Default behavior in HTTP/1.0 unless explicitly configured.

10a. Explain how data connections happens in File Transfer Protocol

Data Connection

The data connection uses the well-known port 20 at the server site. However, the creation of a data connection is different from the control connection. The following shows the steps:

1. The client, not the server, issues a passive open using an ephemeral port. This must be done by the client because it is the client that issues the commands for transferring files.
2. Using the PORT command the client sends this port number to the server.
3. The server receives the port number and issues an active open using the well-known port 20 and the received ephemeral port number.

Communication over Data Connection

The purpose and implementation of the data connection are to transfer files through the data connection. The client must define the type of file to be transferred, the structure of the data, and the transmission mode.

Before sending the file through the data connection, we prepare for transmission through the control connection. The heterogeneity problem is resolved by defining three attributes of

communication: file type, data structure, and transmission mode.

File Type

FTP can transfer one of the following file types across the data connection: ASCII file, EBCDIC file, or image file.

Data Structure

FTP can transfer a file across the data connection using one of the following interpretations of the structure of the data: *file structure, record structure,* or *page structure*. ***Transmission Mode*** FTP can transfer a file across the data connection using one of the following three transmission modes: *stream mode, block mode,* or *compressed mode*. The stream mode is the default mode; data are delivered from FTP to TCP as a continuous stream of bytes. In the block mode, data can be delivered from FTP to TCP in blocks.

File Transfer

File transfer occurs over the data connection under the control of the commands sent over the control connection. However, we should remember that file transfer in FTP means one of three things: *retrieving a file* (server to client), *storing a file* (client to server), and *directory listing* (server to client)

Security for FTP

The FTP protocol was designed when security was not a big issue. Although FTP requires a password, the password is sent in plaintext (unencrypted), which means it can be intercepted and used by an attacker. The data transfer connection also transfers data in plain text, which is insecure. To be secure, one can add a Secure Socket Layer between the FTP application layer and the TCP layer. In this case FTP is called SSL-FTP.

10b. List the difference between local and remote logging.

Here is a comparison of **local logging** and **remote logging**, which differ based on where and how log data is stored and accessed:

Aspect	Local Logging	Remote Logging
Definition	Logs are stored on the same machine or server where the application or process is running.	Logs are sent and stored on a different machine or centralized server.
Storage Location	Logs are saved locally on disk or within the local file system of the host machine.	Logs are stored on a remote server, often using a centralized logging system.

Aspect	Local Logging	Remote Logging
Access	Direct access to the local machine is needed to view logs.	Logs can be accessed from any system with appropriate credentials and tools.
Reliability	Susceptible to data loss if the local system crashes or experiences storage failure.	Provides greater reliability since logs are stored externally, independent of the host system.
Scalability	Limited scalability, especially in distributed systems with multiple hosts.	Highly scalable, as centralized systems can handle logs from many sources.
Performance Impact	Minimal network overhead but may affect local disk usage and performance.	Introduces network overhead but offloads storage and processing from the local system.
Centralization	Logs are decentralized and stored on individual systems.	Logs are centralized, simplifying monitoring and analysis.
Analysis and Monitoring	Difficult to aggregate and analyze logs across multiple machines.	Facilitates real-time monitoring and easier log analysis using centralized tools.
Security	Logs are stored locally and may be more vulnerable to local attacks or unauthorized access.	Typically more secure, as logs can be stored on secure remote systems with controlled access.
Use Cases	Useful for single-system applications or during development.	Essential for distributed systems, large-scale environments, or production systems.

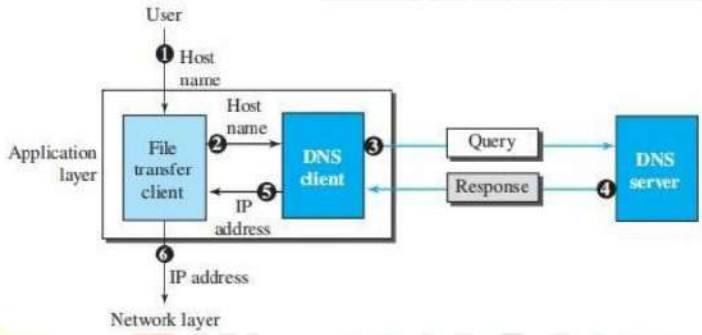
10c. Explain briefly Domain Name System (DNS)

DOMAIN NAME SYSTEM (DNS)

The Domain Name System (DNS) was developed to simplify access to Internet resources by mapping human-friendly names to IP addresses, which are needed for network identification. Similar to how a telephone directory helps map names to numbers, DNS serves as a directory for the Internet, allowing users to remember domain names rather than numeric IP addresses. A central directory for the entire Internet would be impractical due to

its vast scale and vulnerability to failures. Instead, DNS information is distributed across multiple servers worldwide. When a host requires name-to-IP mapping, it contacts the nearest DNS server with the necessary information. This distributed structure enhances reliability and efficiency.

Figure shows how TCP/IP uses a DNS client and a DNS server to map a name to an address. A user wants to use a file transfer client to access the corresponding file transfer server running on a remote host. The user knows only the file transfer server name, such as *afilesource.com*. The TCP/IP suite needs the IP address of the file transfer server to make the connection.



The following six steps map the host name to an IP address:

1. The user passes the host name to the file transfer client.
2. The file transfer client passes the host name to the DNS client.
3. Each computer, after being booted, knows the address of one DNS server. The DNS client sends a message to a DNS server with a query that gives the file transfer server name using the known IP address of the DNS server.
4. The DNS server responds with the IP address of the desired file transfer server.
5. The DNS server passes the IP address to the file transfer client.
6. The file transfer client now uses the received IP address to access the file transfer server.

NAMMA
VTU