

Perfect  You've shared your **Assignment No. 3: Router Configuration using RIP/OSPF/BGP in Packet Tracer**.

Below is a complete **viva/oral preparation guide** — includes:

- All **possible questions** based on your experiment,
 - **Short answers (5–7 lines)** for oral use, and
 - **Detailed answers** for important questions, written in **very simple language**.
-

Assignment No. 3 – Configuration of 3 Router Networks using RIP in Packet Tracer

A. Important Viva / Oral Questions with Answers

1. What is a Router? (*Basic but important*)

Answer:

- A router is a **network device** that connects **two or more networks** together.
 - It works at **Layer 3 (Network Layer)** of the OSI model.
 - It uses **IP addresses** to forward data packets from one network to another.
 - Routers use **routing tables** to decide the best path for data transmission.
 - Example: In Packet Tracer, routers connect multiple LANs or subnetworks.
-

2. What is an IP Address and Default Gateway?

Answer:

- An **IP address** is a unique number assigned to each device in a network for identification and communication.
 - The **Default Gateway** is the **router's IP address** through which data leaves the local network to reach other networks.
 - Example: If PC IP is 192.168.10.2, its default gateway may be 192.168.10.1.
-

3. What is RIP (Routing Information Protocol)? (*Most important question — detailed answer below*)

Detailed Answer:

- RIP stands for **Routing Information Protocol**, one of the oldest **dynamic routing protocols**.
 - It uses the **distance-vector algorithm** and **hop count** as a metric to determine the best route.
 - Each router shares its routing table with its neighbors every **30 seconds**.
 - The **maximum hop count** is **15** — if a network is 16 hops away, it is considered unreachable.
 - RIP helps routers automatically learn routes without manual configuration.
 - It prevents routing loops using **split horizon** and **route poisoning** techniques.
 - Versions: **RIPv1** (classful, no subnet info) and **RIPv2** (classless, supports subnetting).
 - **Command Example:**
 - Router(config)#router rip
 - Router(config-router)#network 192.168.10.0
 - Router(config-router)#network 10.0.0.0
-

4. What is OSPF (Open Shortest Path First)? (*Detailed answer — important for comparison*)

Detailed Answer:

- OSPF stands for **Open Shortest Path First**.
- It is a **link-state routing protocol** that uses **Dijkstra's algorithm** to find the shortest path.
- Unlike RIP, OSPF does **not use hop count**; it uses **cost (bandwidth)** as a metric.

- OSPF is faster, more efficient, and supports large networks.
 - It updates only when there is a **change in the network**, reducing unnecessary traffic.
 - Works on the concept of **areas**, with **Area 0** being the backbone.
 - **Command Example:**
 - Router(config)#router ospf 1
 - Router(config-router)#network 192.168.10.0 0.0.0.255 area 0
-

5. What is BGP (Border Gateway Protocol)?

Answer (Detailed):

- BGP stands for **Border Gateway Protocol**.
 - It is used for routing **between different autonomous systems (AS)** — mainly used on the **internet**.
 - It is a **path-vector protocol** and makes routing decisions based on **policies and rules**.
 - BGP maintains a table of all reachable networks and paths to them.
 - It is used by **ISPs** to exchange routing information globally.
 - **Command Example:**
 - Router(config)#router bgp 100
 - Router(config-router)#neighbor 192.168.1.2 remote-as 200
-

6. What is the difference between RIP, OSPF, and BGP? (*Most frequently asked*)

Feature	RIP	OSPF	BGP
Type	Distance Vector	Link State	Path Vector
Metric	Hop Count	Cost (Bandwidth)	Policy-based
Max Hops	15	No Limit	No Limit
Speed	Slow	Fast	Very Fast
Use Case	Small Networks	Medium/Large Networks	Internet/ISPs
Algorithm	Bellman-Ford	Dijkstra	Path Selection Algorithm

Simple Summary:

- **RIP** – Easy, for small networks.
 - **OSPF** – Smart, for large networks.
 - **BGP** – Global routing on the internet.
-

7. What is Hop Count?

Answer:

- **Hop count** means the number of routers a packet must pass through to reach its destination.
 - RIP uses hop count as its **metric** — fewer hops = better route.
 - Maximum hop count in RIP is **15**; 16 means unreachable.
-

8. What are Routing Protocols?

Answer:

- Routing protocols are rules that routers use to **exchange routing information** automatically.
- They help routers **find the best path** to send data.
- Examples: RIP, OSPF, BGP, EIGRP.
- Types:
 - **Interior Gateway Protocols (IGP)** – RIP, OSPF
 - **Exterior Gateway Protocols (EGP)** – BGP

9. What is a Routing Table?

Answer:

- A **routing table** is a list inside the router that stores information about possible routes.
 - It contains destination networks, next hop, and interface details.
 - The router checks this table to decide where to forward packets.
-

10. How to verify connectivity between PCs?

Answer:

- Use the **ping command** in the command prompt.
 - Syntax: ping <destination IP>
 - If you get a **reply**, the connection works; if **request timed out**, the network is not connected.
-

11. What is Cisco Packet Tracer?

Answer:

- It is a **network simulation tool** developed by **Cisco**.
 - It allows users to **create, configure, and test** network topologies virtually.
 - You can connect routers, switches, and PCs to simulate real-world networking.
-

12. What is the purpose of subnet mask in configuration?

Answer:

- Subnet mask helps to **separate the network and host portion** of an IP address.
 - Example:
 - IP: 192.168.10.2
 - Subnet Mask: 255.255.255.0 → Network part = 192.168.10, Host = 2
-

13. Why do we use RIP instead of static routing?

Answer:

- In **static routing**, routes are added manually — difficult for large networks.
 - **RIP** automatically updates and shares routes, making it easier for **dynamic** environments.
 - If a network fails, RIP automatically updates the route table.
-

14. What is the difference between Static and Dynamic Routing?

Feature	Static Routing	Dynamic Routing
---------	----------------	-----------------

Configuration	Manual	Automatic
---------------	--------	-----------

Updates	Not Automatic	Periodic Updates
---------	---------------	------------------

Best For	Small Networks	Medium/Large Networks
----------	----------------	-----------------------

Example	ip route	RIP, OSPF, BGP
---------	----------	----------------

15. What is the role of a default route?

Answer:

- A **default route** acts as a “catch-all” path when no specific route is found in the routing table.
 - Command: ip route 0.0.0.0 0.0.0.0 <next-hop>
-

 Conclusion (Simple for Oral)

We learned how to configure **three routers** in Cisco Packet Tracer using **RIP protocol**.

We also understood **how routing works**, how to use **IP addressing, subnet masks**, and **verify connections** using the ping command.

Would you like me to make this into a **well-formatted PDF (like your previous ones)** — keeping the **same content readable, clean, and printable** for submission or viva use?

Perfect  Below is your **complete viva/oral preparation guide** for **Assignment No. 4 – Subnetting and Subnet Masks**.

It includes:

- All possible oral questions** (short and detailed)
 - Simple, clear explanations**
 - Focus on IPv4, subnetting logic, and real examples**
-

Assignment No. 4: Subnetting and Subnet Masks

◆ **1. What is Subnetting? (*Important – explain in detail*)**

Answer:

- Subnetting is the process of dividing one large network into smaller, more manageable subnetworks (subnets).
 - It is done at **Layer 3 (Network Layer)** using **IP addresses**.
 - Each subnet functions like a small network with its own range of IP addresses.
 - Subnetting helps improve performance, security, and efficient IP utilization.
 - Example: If you have a Class C network (192.168.1.0/24) and you need 4 subnets, you can borrow bits from the host portion to create smaller subnets.
-

◆ **2. What is a Subnet Mask? (*Important – explain in detail*)**

Answer:

- A **Subnet Mask** is a 32-bit number used with an IP address to separate the **network** and **host** parts.
 - It tells which bits in an IP address represent the **network** and which represent the **hosts**.
 - In binary form, 1 bits are for the network part, and 0 bits are for the host part.
 - Example:
 - IP Address: 192.168.1.10
 - Subnet Mask: 255.255.255.0
 - Network Address: 192.168.1.0
 - Host Range: 192.168.1.1 – 192.168.1.254
-

◆ **3. What is the difference between Network and Host Address?**

Answer:

- **Network Address:** Identifies the whole subnet or network (used by routers).
 - **Host Address:** Identifies individual devices (PC, router, etc.) in that network.
 - Example:
 - IP: 192.168.1.10
 - Subnet Mask: 255.255.255.0
 - Network Address: 192.168.1.0
 - Host Portion: .10
-

◆ 4. What are IP address classes?

Answer:

	Class Range (1st Octet)	Default Subnet Mask	Network Bits	Host Bits	Example
A	1–126	255.0.0.0	8	24	10.0.0.1
B	128–191	255.255.0.0	16	16	172.16.0.1
C	192–223	255.255.255.0	24	8	192.168.1.1
D	224–239	–	–	–	(Multicast)
E	240–254	–	–	–	(Experimental)

◆ 5. What is the purpose of subnetting?

Answer:

- To divide a large network into smaller networks.
- To reduce network traffic and broadcast.
- To enhance performance and security.
- To use IP addresses efficiently.
- To manage networks easily.

◆ 6. What is meant by Network ID and Broadcast ID?

Answer:

- **Network ID:** The first address in a subnet; used to identify the subnet itself.
- **Broadcast ID:** The last address in a subnet; used to send data to all devices in that subnet.
- Example for 192.168.1.0/24:
 - Network ID → 192.168.1.0
 - Broadcast ID → 192.168.1.255

◆ 7. What is Block Size in subnetting?

Answer:

- Block size represents the **total number of addresses in each subnet**, including the network and broadcast addresses.
- Formula: $256 - (\text{last octet of subnet mask})$
- Example: If mask = 255.255.255.192 → Block size = $256 - 192 = 64$

◆ 8. What is meant by Host Addresses?

Answer:

- Host addresses are the usable IPs between the **Network ID** and **Broadcast ID**.
- They can be assigned to devices like PCs, routers, or printers.
- Example:
 - Subnet: 192.168.1.0/26
 - Usable Hosts: 192.168.1.1 – 192.168.1.62

◆ 9. Explain IPv4 in simple terms.

Answer:

- IPv4 stands for **Internet Protocol version 4**.
- It uses **32-bit** addresses written in **dotted decimal notation** (e.g., 192.168.1.1).
- It has about **4.3 billion unique addresses**.
- It consists of **Network** and **Host** parts.

- IPv4 is still widely used in LANs and the Internet.
-

◆ **10. What are the ranges of IP address classes in IPv4?**

Answer:

- Class A → 1 to 126
- Class B → 128 to 191
- Class C → 192 to 223
- Class D → 224 to 239 (Multicast)
- Class E → 240 to 254 (Experimental)

(Note: 127 is reserved for loopback, and 255.255.255.255 is for broadcast.)

◆ **11. What are the advantages of Subnetting?**

Answer:

- Reduces network congestion by limiting broadcasts.
 - Improves network performance.
 - Increases security by isolating subnets.
 - Efficient IP address utilization.
 - Easier to manage smaller subnets than one large network.
-

◆ **12. What is Loopback Address and its purpose?**

Answer:

- The loopback address is **127.0.0.1**.
 - It is used to test whether the **TCP/IP stack** of a system is working properly.
 - It doesn't send data on the network, only tests internal networking.
-

◆ **13. What is Broadcast Address?**

Answer:

- It is an address used to send packets to all hosts within a network.
 - **Local Broadcast:** The last address in subnet (e.g., 192.168.1.255 for /24).
 - **Full Broadcast:** 255.255.255.255 → sends data to all devices on all networks.
-

◆ **14. What is a Directed Broadcast?**

Answer:

- Directed broadcast means sending packets to **all hosts of a specific subnet**.
 - Example: For subnet 192.168.1.0/24 → Directed broadcast is 192.168.1.255.
-

◆ **15. What is the default subnet mask for each class?**

Answer:

Class Default Subnet Mask Format

- | | | |
|---|---------------|------------------------------|
| A | 255.0.0.0 | Network.Host.Host.Host |
| B | 255.255.0.0 | Network.Network.Host.Host |
| C | 255.255.255.0 | Network.Network.Network.Host |
-

◆ **16. What is meant by “Borrowing Bits” in subnetting?**

Answer:

- It means taking bits from the **host portion** of the IP address and using them for the **network portion** to create more subnets.
 - Example: For Class C (default /24), if you borrow 2 bits → new subnet mask becomes /26 (255.255.255.192).
-

◆ **17. What happens when you borrow bits?**

Answer:

- The number of **subnets increases**, but the number of **hosts per subnet decreases**.
 - Formula:
 - No. of subnets = 2^n (where n = borrowed bits)
 - No. of hosts = $2^h - 2$ (where h = remaining host bits)
-

◆ **18. What is the formula to calculate total hosts per subnet?**

Answer:

- Formula: $2^h - 2$, where h is the number of host bits.
 - Example: /26 → 6 host bits → $2^6 - 2 = 62$ usable hosts.
-

◆ **19. Why is Subnetting important in real networks?**

Answer:

- It reduces waste of IP addresses.
 - Helps separate departments logically (e.g., HR, Admin, IT).
 - Enhances security and control of data flow.
 - Helps routers perform efficiently by reducing broadcast domains.
-

◆ **20. What are the reserved IP addresses?**

Answer:

- **0.0.0.0** → Default route (all IPs).
 - **127.0.0.1** → Loopback (testing).
 - **255.255.255.255** → Broadcast (all hosts).
-

◆ **21. What is the difference between IPv4 and IPv6?**

Answer:

Feature	IPv4	IPv6
Address Length	32-bit	128-bit
Format	Dotted decimal	Hexadecimal
Example	192.168.1.1	2001:0db8::1
No. of Addresses	~4.3 billion	Almost unlimited
Header Size	Small	Larger but efficient

◆ **22. What is meant by “CIDR Notation”?**

Answer:

- CIDR stands for **Classless Inter-Domain Routing**.
 - It represents subnet masks using a “/” followed by the number of network bits.
 - Example:
 - 192.168.1.0/24 → means 24 bits are for the network (255.255.255.0).
-

◆ **23. Example Question – Subnetting Calculation**

Q: Divide 192.168.1.0/24 into 4 subnets.

Answer:

- Borrow 2 bits → New mask /26 → 255.255.255.192
- Subnets:
 - 1** 192.168.1.0 – 192.168.1.63
 - 2** 192.168.1.64 – 192.168.1.127
 - 3** 192.168.1.128 – 192.168.1.191
 - 4** 192.168.1.192 – 192.168.1.255

◆ **24. What is the outcome of this experiment?**

Answer:

- Successfully learned how to identify **IP address classes**.
- Understood how **subnetting** divides networks and assigns subnet masks.
- Implemented subnetting examples and calculated subnet addresses.

◆ **25. What is the conclusion of this experiment?**

Answer:

We have successfully implemented subnetting and subnet masks.

Subnetting helps divide large networks into smaller ones, improves efficiency, saves IP addresses, and enhances security.

Would you like me to make this entire oral/viva preparation **PDF-ready** (formatted neatly with headings, bold text, and page style — no content change)?

I can prepare it in 1 click.

Perfect  Let's prepare your **viva/oral question and answer guide** for **Assignment No. 5 – Socket Programming using TCP and UDP (in C/C++/Java)**.

I'll give you:

- All **possible oral questions** that your teacher may ask
- Each with a **simple 5–7 line answer**
- For **important questions**, I'll give **detailed and easy-to-understand explanations** separately

 **Assignment No. 5 – Socket Programming (TCP & UDP)**

◆ **A. Basic Oral Questions (with Simple 5–7 Line Answers)**

1 **What is client-server communication?**

- Client-server communication is a model where one system (client) requests a service, and another system (server) provides it.
- The client initiates the connection, and the server waits and responds.
- Examples include web browsers (clients) connecting to web servers.
- Communication happens using IP addresses and ports.
- It can use protocols like TCP or UDP for data transfer.

2 **What is socket programming?**

- Socket programming is a way to connect two computers and exchange data over a network.
- A **socket** is an endpoint for sending and receiving data.

- It allows client-server communication using TCP or UDP.
 - In C/C++, functions like `socket()`, `bind()`, `send()`, `recv()` are used.
 - It is commonly used for chat apps, web communication, and networked systems.
-

3 What are the main socket functions used?

- `socket()` → Creates a socket (communication endpoint).
 - `bind()` → Assigns a local address (IP and port) to the socket.
 - `listen()` → Used by the TCP server to wait for client requests.
 - `accept()` → Accepts a client connection (TCP only).
 - `connect()` → Used by the client to connect to the server.
 - `send()`, `recv()` / `sendto()`, `recvfrom()` → Used to send and receive data.
 - `close()` → Closes the connection.
-

4 What is TCP?

- TCP stands for **Transmission Control Protocol**.
 - It provides a **connection-oriented**, **reliable**, and **stream-based** communication.
 - Data is guaranteed to be delivered in order and without loss.
 - It uses acknowledgments (ACKs) and retransmission in case of errors.
 - Used in applications like HTTP, FTP, and email.
-

5 What is UDP?

- UDP stands for **User Datagram Protocol**.
 - It provides **connection-less**, **unreliable**, and **fast** communication.
 - Data may arrive out of order or get lost.
 - It does not retransmit lost packets.
 - Used in applications like DNS, video streaming, and online gaming.
-

6 What is the difference between TCP and UDP?

Feature	TCP	UDP
Connection	Connection-oriented	Connection-less
Reliability	Reliable (ACKs, retransmission)	Unreliable
Speed	Slower	Faster
Ordering	Maintains order	No order guarantee
Example	HTTP, FTP	DNS, Streaming

7 What is the role of bind() function?

- The `bind()` function assigns a local address (IP + port) to a socket.
 - It is mainly used in the server program.
 - It ensures that the socket listens on a specific port number.
 - Without binding, the OS may assign a random port.
-

8 What is the role of connect() function?

- The `connect()` function is used by a **TCP client** to establish a connection with a server.
 - It requires the server's IP address and port number.
 - After connection, data can be sent and received using `send()` and `recv()`.
-

9 What is the role of accept() function?

- The accept() function is used by a **TCP server** to accept incoming client connections.
 - It waits until a client connects.
 - After accepting, it returns a new socket descriptor for communication with that client.
-

10 Why is UDP called connection-less?

- UDP does not establish a connection before sending data.
 - The client just sends data using sendto() without any handshake.
 - The server receives data using recvfrom().
 - There is no guarantee of delivery or order.
-

1 1 What are real-life examples of TCP and UDP?

- **TCP:** Web browsing (HTTP), Email (SMTP), File transfer (FTP).
 - **UDP:** Video streaming (YouTube), Voice calls (VoIP), DNS lookup, Online games.
-

1 2 What is a port number and why is it used?

- A port number identifies a specific process or service on a host.
 - It allows multiple network services to run simultaneously on the same machine.
 - Example: HTTP uses port 80, FTP uses port 21.
-

1 3 What is meant by “reliable communication”?

- Reliable communication means that data reaches the receiver correctly and in order.
 - TCP ensures reliability using acknowledgments and retransmissions.
 - UDP does not provide reliability.
-

1 4 What is the difference between send() and sendto() functions?

- **send()** → Used in **TCP**, where a connection exists between client and server.
 - **sendto()** → Used in **UDP**, where no connection exists; it requires the destination address each time.
-

1 5 What is checksum in UDP?

- Checksum is used for **error detection** in UDP packets.
 - It ensures that data is not corrupted during transmission.
 - If an error is detected, the datagram is discarded (not retransmitted).
-

1 6 What is the role of listen() function?

- listen() is used in TCP servers.
 - It allows the server to listen for incoming client connection requests.
 - The server specifies how many pending connections can be queued.
-

1 7 What happens when a TCP client sends data but doesn't receive an acknowledgment?

- TCP will **retransmit** the data after a timeout.
 - It will continue until the acknowledgment is received or the connection fails.
 - This ensures reliable delivery.
-

1 8 Why is UDP faster than TCP?

- Because it does not establish a connection before sending data.

- It doesn't perform acknowledgments or retransmissions.
 - This reduces delay, making it ideal for real-time communication.
-

1 9 What are the steps in TCP socket programming?

Client:

1. Create socket
2. Connect to server using connect()
3. Send/Receive data
4. Close socket

Server:

1. Create socket
 2. Bind to IP and port
 3. Listen for connections
 4. Accept client
 5. Send/Receive data
 6. Close socket
-

2 0 What are the steps in UDP socket programming?

Client:

1. Create socket
2. Send data using sendto()
3. Receive response using recvfrom()
4. Close socket

Server:

1. Create socket
 2. Bind it to IP and port
 3. Wait for data using recvfrom()
 4. Send response using sendto()
 5. Close socket
-

B. Detailed & Important Questions (Long Answers in Simple Words)

◆ Q1. Explain Socket Programming in detail.

Answer:

Socket programming allows two systems (client and server) to communicate over a network.

A **socket** acts as an endpoint for sending and receiving data.

It uses **IP address + port number** to identify applications uniquely.

Two types of sockets are used:

- **TCP sockets:** Reliable, connection-oriented.
- **UDP sockets:** Fast, connection-less.

Socket functions like socket(), bind(), listen(), accept(), connect(), send(), and recv() handle communication.

It's the base for applications like web browsers, chat apps, and file transfer tools.

◆ Q2. Explain TCP communication process in detail.

Answer:

TCP communication works in a **client-server** model.

1. Server:

- Creates a socket using `socket()`.
- Binds it to a port using `bind()`.
- Waits for client requests using `listen()`.
- Accepts a client using `accept()`.

2. Client:

- Creates a socket and connects to the server using `connect()`.
- Both can then send and receive data using `send()` and `recv()`.
TCP ensures that all data arrives correctly and in order using acknowledgments (ACKs).

This makes it **reliable** but **slower** than UDP.

◆ Q3. Explain UDP communication process in detail.

Answer:

UDP is a **connection-less** protocol, meaning it doesn't establish a connection before sending data.

- The **client** creates a socket and directly sends data to the server using `sendto()`.
- The **server** binds a socket to a port using `bind()` and waits for data using `recvfrom()`.
- Once data is received, it can reply back using `sendto()`.

There's no acknowledgment or retransmission — if a packet is lost, it's gone.

UDP is faster and used in **real-time applications** like streaming and gaming.

◆ Q4. Difference between TCP and UDP (Detailed)

Feature	TCP	UDP
Full Form	Transmission Control Protocol	User Datagram Protocol
Type	Connection-oriented	Connection-less
Reliability	Reliable – ensures delivery	Unreliable – no guarantee
Acknowledgment	Yes	No
Error Checking	Yes (retransmits lost packets)	Yes (discard errors)
Speed	Slower due to reliability	Faster, lightweight
Order	Maintains data order	No guarantee of order
Example	HTTP, FTP, Email	DNS, Streaming, Online games

◆ Q5. What are some real-time applications of socket programming?

Answer:

- **Chat applications** (like WhatsApp or Messenger).
- **Web servers** (HTTP servers communicating with browsers).
- **Online games** (use UDP for real-time response).
- **Video conferencing and VoIP apps** (use UDP for low latency).
- **File transfer tools** (use TCP for reliability).

Would you like me to include **short C/C++ code snippets** for both TCP and UDP client-server communication (for viva reference)?

They're often asked in practicals too.

Perfect  Let's prepare your **oral/viva questions and answers** for **Assignment No. 6 – TCP Socket Programming (Wired Network)**.

I'll give you:

- Important viva questions** (about 15 total)
 - Short, simple answers** (5–7 lines each)
 - Detailed explanations** for the most important ones
-

Assignment No: 6 – TCP Socket Programming (Wired Network)

Problem Statement

Write a program using TCP socket for wired network for:

- Saying Hello to each other
 - File transfer
-

Important Viva / Oral Questions and Answers

1. What is socket programming? (*Detailed Answer*)

Socket programming is a way to connect two computers or processes over a network so they can exchange data.

- A **socket** acts as an endpoint for sending and receiving data.
- It allows programs to communicate using **TCP/IP** or **UDP** protocols.
- In TCP socket programming, a **client** connects to a **server** through a specific **IP address** and **port number**.
- It is used for real-time communication such as chatting, file transfer, or web requests.

Example: Sending a “Hello” message between a client and server over TCP.

2. What is a socket?

- A socket is a combination of **IP address + port number**.
 - It acts like a **communication endpoint** between two processes.
 - Each socket is identified by a unique number called a **socket descriptor**.
 - In Unix, sockets are treated like **file descriptors** (used for read/write).
-

3. What are the different types of sockets? (*Detailed Answer*)

There are **four main types** of sockets:

1. **Stream Socket (SOCK_STREAM)** – Uses **TCP**, connection-oriented, reliable.
2. **Datagram Socket (SOCK_DGRAM)** – Uses **UDP**, connectionless, faster but unreliable.
3. **Raw Socket (SOCK_RAW)** – Gives access to low-level network protocols for developers.
4. **Sequenced Packet Socket** – Used in Network Systems (NS) for special packet-handling.

 *Most commonly used:* Stream and Datagram sockets.

4. Difference between TCP and UDP sockets?

Feature	TCP	UDP
Type	Stream socket	Datagram socket
Connection	Connection-oriented	Connectionless
Reliability	Reliable	Unreliable
Order	Maintains order	May arrive out of order
Use case	File transfer, web	Video streaming, games

5. What is TCP? (*Detailed Answer*)

- **TCP (Transmission Control Protocol)** is a **connection-oriented** and **reliable** protocol.
 - It ensures that all data packets are delivered **in order and without loss**.
 - It performs **error checking, flow control**, and **retransmission** of lost packets.
 - Common examples include **HTTP, FTP, and Email transfer**.
 - It is used in **Stream Sockets** for safe data transmission.
-

6. What are the main stages of TCP Server programming? (*Detailed Answer*)

There are **4 main stages** for a TCP server:

1. **Socket creation:** Create a socket using `socket()`.
2. **Bind:** Attach it to a specific **IP** and **port** using `bind()`.
3. **Listen:** Put the socket in passive mode with `listen()`.
4. **Accept:** Accept client requests with `accept()` and start data transfer.

After this, the server and client can **send and receive messages or files**.

7. What are the main stages of TCP Client programming?

1. **Socket creation:** Create socket using `socket()`.
 2. **Connect:** Use `connect()` to connect to the server's IP and port.
 3. **Send/Receive:** Exchange data using `send()` and `recv()`.
 4. **Close:** Terminate the connection with `close()`.
-

8. What is the purpose of the bind() function?

- `bind()` associates the socket with a **specific IP address and port**.
 - It allows the system to know where to send/receive data.
 - Without binding, the OS won't know which port the server listens on.
 - Syntax: `bind(sockfd, (struct sockaddr *)&addr, sizeof(addr));`
-

9. What is the role of the listen() function?

- `listen()` puts the server socket into **passive mode**.
 - It tells the server to **wait for client connections**.
 - It takes two arguments: the socket descriptor and **backlog** (max number of pending connections).
-

10. What does accept() function do?

- `accept()` waits for a client to connect.
 - When a request arrives, it creates a **new socket** for communication.
 - Returns a **new socket descriptor** used to send/receive data with that client.
-

11. What is connect() in client side?

- `connect()` establishes a connection to the server.
 - It takes the socket descriptor and the **server address structure** as parameters.
 - If the connection is successful, data transfer can begin using `send()` and `recv()`.
-

12. What is the difference between client and server socket?

Client Socket

Server Socket

Initiates connection using `connect()` Waits for connections using `listen()`

Knows server's IP and port

Binds to a local IP and port

Client Socket	Server Socket
Usually runs temporarily	Often runs continuously
One socket per client	One listening socket, many client sockets

13. What are the advantages of TCP socket programming?

- Reliable and connection-oriented.
 - Ensures data integrity and order.
 - Supports error recovery and retransmission.
 - Good for applications like file transfer, email, or web services.
-

14. What happens when the client and server say “Hello” to each other?

- The **client sends** a message like “Hello Server”.
 - The **server receives** it and sends back “Hello Client”.
 - This verifies that the connection and data transfer are working.
 - It’s the simplest form of message exchange via TCP.
-

15. What are the steps for file transfer using TCP?

1. Establish connection using socket and connect.
 2. Open the file in **read mode** at the sender side.
 3. Send data in **chunks** using send().
 4. Receive data using recv() and write to file at receiver side.
 5. Close both sockets after completion.
-

Conclusion (for Oral)

“In this assignment, I learned how to implement **TCP socket programming** for communication between two systems.

I successfully made the client and server **say hello to each other** and **transfer a file** using a reliable connection.

Through this, I understood key functions like socket(), bind(), listen(), accept(), and connect() which are essential for TCP communication.”

Extra Viva Tips

- Be ready to **explain TCP vs UDP** (very common question).
 - Know the **order of socket functions** on both sides.
 - Mention **port numbers** (e.g., 8080 or 5000 commonly used).
 - Mention that TCP is **connection-oriented**, reliable, and ensures data delivery.
-

Would you like me to also include a **sample TCP client–server C/C++ code** for “Say Hello” and **File Transfer** (fully commented) to keep in your report and for oral demonstration?

Perfect  Let’s prepare your **UDP Socket Programming (Assignment No. 7) viva/oral guide** — including:

-  **20+ expected oral/viva questions**
 -  **Simple 5–7 line answers for quick oral use**
 -  **Detailed explanations** for important questions (marked )
-

Assignment No: 7 — UDP File Transfer using Sockets

Problem Statement:

Write a program using UDP sockets to enable file transfer (Script, Text, Audio, and Video one file each) between two machines.

Objective:

1. To learn how to transfer files using socket programming.
 2. To understand the working of UDP protocol for data transmission.
-

Expected Viva / Oral Questions and Answers

1. What is UDP?

Answer:

- UDP stands for *User Datagram Protocol*.
 - It is a **connectionless, unreliable**, transport-layer protocol.
 - It does not establish a connection before sending data.
 - There's **no acknowledgment, flow control, or retransmission**.
 - Each packet (called a datagram) is sent independently.
 - Used where speed is preferred over reliability (like video streaming or gaming).
-

2. What is the difference between TCP and UDP?

Answer:

Feature	TCP	UDP
Connection	Connection-oriented	Connectionless
Reliability	Reliable (acknowledgment, retransmission)	Unreliable
Flow Control	Yes	No
Speed	Slower	Faster
Example	File Transfer (FTP, HTTP)	Video, DNS, VoIP

In short, **TCP ensures delivery**, while **UDP focuses on speed**.

3. What are port numbers in UDP?

Answer:

- Port numbers identify the **specific process or service** on a host.
 - UDP uses **16-bit** port numbers (0–65535).
 - Example:
 - 53 → DNS
 - 69 → TFTP
 - 67/68 → DHCP
 - Source port = sender's port; Destination port = receiver's port.
-

4. Explain the structure of a UDP header.

Answer:

UDP header is **8 bytes** long and has four fields:

1. **Source Port (16 bits)**
2. **Destination Port (16 bits)**
3. **Length (16 bits)** – total length of UDP header + data

4. **Checksum (16 bits)** – used for error detection
→ Simple and efficient since it adds minimal overhead.
-

5. What is checksum in UDP?

Answer:

- Checksum detects **errors** in the UDP header and data.
 - It covers:
 - Pseudo-header (from IP)
 - UDP header
 - Data
 - If the checksum fails, the packet is discarded silently (no retransmission).
-

6. Why is UDP called “connectionless”?

Answer:

- UDP does not establish a session before sending data.
 - Each packet is treated **independently**.
 - There's no **handshake**, **sequence number**, or **acknowledgment**.
 - This makes it faster but less reliable.
-

7. What do you mean by multiplexing and demultiplexing in UDP?

Answer:

- **Multiplexing:** Combining data from multiple processes into one UDP layer using different port numbers.
 - **Demultiplexing:** Delivering received packets to the correct process based on destination port number.
 - This allows multiple applications to use UDP simultaneously.
-

8. What is encapsulation in UDP?

Answer:

- **Encapsulation** means wrapping data with headers as it moves down the network layers.
 - In UDP:
 - Application data → UDP adds its header → IP adds its header → Data Link adds frame header/trailer.
 - Finally, bits are sent over the physical medium.
 - This process ensures data reaches the correct process on the destination host.
-

9. What is de-encapsulation?

Answer:

- It is the reverse process of encapsulation.
 - As data moves **up** the layers:
 - Physical → Data Link → IP → UDP → Application.
 - Each layer removes its header and passes the data upward.
-

10. Explain UDP services.

Answer:

UDP provides:

1. **Process-to-process communication** using port numbers.
2. **Connectionless service** (no setup or teardown).

3. **Minimal error control** (via checksum).
 4. **No flow control** (application must handle it).
 5. **Fast delivery** due to low overhead.
-

11. What are typical applications of UDP?

Answer:

- **DNS (Domain Name System)** – fast request-response.
 - **TFTP (Trivial File Transfer Protocol)** – simple file transfer.
 - **VoIP (Voice over IP)** – real-time audio.
 - **Online games and video streaming** – low latency needed.
 - **DHCP (Dynamic Host Configuration Protocol)** – network config.
-

12. Why is UDP faster than TCP?

Answer:

- No connection setup or teardown.
- No acknowledgment or retransmission.
- Smaller header (8 bytes vs 20 bytes in TCP).
- Less processing overhead at both ends.

Hence, UDP provides **speed over reliability**.

13. What happens if a UDP packet is lost?

Answer:

- UDP does **not retransmit** lost packets.
 - The receiver silently drops erroneous or missing packets.
 - The **application layer** must handle retries if needed.
-

14. What is the use of the length field in UDP header?

Answer:

- Specifies **total size** of the UDP packet (header + data).
 - 16 bits long → max 65,535 bytes.
 - Helps the receiver know where the UDP message ends.
-

15. How does UDP achieve process-to-process communication?

Answer:

- Using **IP address + Port number** combination called a **socket**.
 - Each application is bound to a unique socket.
 - Example:
 - (192.168.1.2, 8080) → identifies a specific process on a machine.
-

16. What are the advantages of UDP?

Answer:

- Simple and faster.
 - Low overhead.
 - Suitable for broadcast/multicast communication.
 - Useful for real-time or loss-tolerant applications.
-

17. What are the disadvantages of UDP?

Answer:

- Unreliable (no acknowledgment).
 - No error recovery or retransmission.
 - No congestion or flow control.
 - Order of packets not guaranteed.
-

18. Can UDP handle large data?

Answer:

- UDP can carry up to **65,535 bytes**, but it's not ideal for very large data.
 - Data may need to be **fragmented** by IP.
 - For large reliable transfers, **TCP is preferred**.
-

19. What is the role of sockets in UDP programming?

Answer:

- A **socket** is an endpoint for communication between two processes.
 - In UDP:
 - DatagramSocket (in Java) or socket(AF_INET, SOCK_DGRAM) (in C) is used.
 - It binds to a **port number** and **IP address**.
 - Enables **sendto()** and **recvfrom()** communication.
-

20. What is meant by connectionless service?

Answer:

- Each message (datagram) is treated independently.
 - No connection setup or teardown.
 - Packets may take different paths to reach the destination.
-

21. Why is UDP preferred for multimedia transmission?

Answer:

- Real-time data (like audio/video) needs **speed and continuity**.
 - A few lost packets don't affect quality much.
 - TCP's retransmission causes **delay**, which breaks real-time flow.
-

22. What happens if the checksum fails?

Answer:

- The UDP packet is **discarded silently**.
 - No retransmission is requested.
 - This keeps UDP simple and fast.
-

23. What is the main difference between a UDP and TCP file transfer program?

Answer:

- TCP ensures **reliable**, ordered delivery with **acknowledgment**.
 - UDP sends data **without confirmation**, faster but less reliable.
 - UDP-based file transfer (like TFTP) must implement reliability manually.
-

24. What do you conclude from this experiment?

Answer:

- We successfully implemented **UDP socket programming** for file transfer.
- Understood the working of **connectionless** communication.
- Learned to use **DatagramSocket** for sending and receiving files.

- Observed that UDP is **faster** but **unreliable** compared to TCP.
-

Summary (Key Concepts to Remember for Oral)

Concept	Meaning
UDP	Connectionless, unreliable protocol
Header size	8 bytes
Key Fields	Source Port, Destination Port, Length, Checksum
Flow Control	None
Error Control	Only checksum
Reliability	Not guaranteed
Example Apps	DNS, VoIP, TFTP, DHCP

Would you like me to also include a **small diagram explanation** (UDP encapsulation + de-encapsulation flow) and **sample code summary** for oral explanation (2–3 lines each for server and client)? It's often asked in practical vivas.

Perfect  Let's prepare your **oral/viva question set** for **Assignment No. 8 — “Capture TCP and HTTP packets using Wireshark.”**

I'll divide it into two parts for you:

-  **Part A: Short & Simple Answers (for quick viva) — 5–7 lines each**
 -  **Part B: Detailed & Important Questions (for explanation-type viva)**
-

PART A — Important Oral / Viva Questions (Short Answers)

1 What is Wireshark?

- Wireshark is a free and open-source **network packet analyzer tool**.
 - It captures and displays packets in **real-time** from your network interface.
 - It helps analyze **network protocols, troubleshoot issues, and study communication**.
 - It supports hundreds of protocols like TCP, UDP, HTTP, DNS, etc.
 - It is widely used in **network security, research, and education**.
-

2 What is the purpose of this experiment?

- To learn how to **capture and analyze TCP and HTTP packets** using Wireshark.
 - To understand **how data is transmitted between client and server**.
 - To apply filters and analyze flags like **SYN, PSH, RST** in TCP.
 - To measure **traffic counts** and protocol usage.
-

3 What is TCP?

- TCP stands for **Transmission Control Protocol**.
 - It is a **connection-oriented** and **reliable** protocol.
 - It ensures **error-free, ordered delivery** of data between two endpoints.
 - It uses **three-way handshake** for connection setup.
 - It includes features like **flow control, error control, and congestion control**.
-

4 What is HTTP?

- HTTP stands for **HyperText Transfer Protocol**.

- It is an **application layer protocol** used for communication on the web.
 - It works on **TCP port 80** (or 443 for HTTPS).
 - HTTP is **stateless** — each request is independent.
 - It defines how a **client (browser)** and **server** exchange web resources.
-

5 What is the difference between TCP and HTTP?

Feature	TCP	HTTP
Layer	Transport Layer	Application Layer
Type	Connection-oriented	Request/Response
Purpose	Data transfer reliability	Web communication
Port	80 (default)	Uses TCP port 80
Function	Ensures delivery	Defines web data format

6 What is a TCP three-way handshake?

- TCP connection is established using a **three-step process**:
 1. **SYN** – Client requests connection.
 2. **SYN-ACK** – Server acknowledges request.
 3. **ACK** – Client confirms acknowledgment.
 - After this handshake, **data transfer begins**.
 - It ensures both ends are **ready and synchronized** for communication.
-

7 What are TCP flags?

- Flags are **1-bit indicators** that control the TCP state.
 - Common flags include:
 - **SYN** – Start connection.
 - **ACK** – Acknowledge data.
 - **PSH** – Push data immediately.
 - **RST** – Reset connection.
 - **FIN** – Terminate connection.
 - They help manage communication flow between sender and receiver.
-

8 What is the use of display filters in Wireshark?

- Display filters are used to **focus on specific packets** of interest.
 - Example filters:
 - `tcp` → all TCP packets
 - `http` → all HTTP packets
 - `tcp.flags.syn == 1` → only SYN packets
 - Filters make it easier to **analyze large captures efficiently**.
-

9 What is a TCP segment?

- A **segment** is the unit of data used by TCP.
 - It consists of a **TCP header** and **data** from the application.
 - The header includes sequence number, acknowledgment, and flags.
 - TCP uses sequence numbers to ensure **ordered data delivery**.
-

10 What is the default port number of HTTP?

- The **default port** for HTTP is **80**.
 - For **HTTPS** (secure HTTP), the default port is **443**.
-

1 1 What is the importance of capturing packets?

- It helps **monitor network traffic** and **identify performance or security issues**.
 - Packet analysis reveals how protocols behave during communication.
 - It is useful for **debugging, intrusion detection, and learning**.
-

1 2 What does SYN, PSH, and RST mean in TCP?

- **SYN (Synchronize)**: Initiates a TCP connection.
 - **PSH (Push)**: Sends data immediately to the receiving application.
 - **RST (Reset)**: Terminates or resets a connection abruptly.
-

1 3 How to count the number of packets in Wireshark?

- Go to **Statistics → Conversations or Protocol Hierarchy**.
 - Use **display filters** like `tcp` or `http`.
 - Wireshark shows the **number of captured, displayed, and filtered packets**.
-

1 4 What are the three phases of a TCP connection?

1. **Connection Establishment (Three-way handshake)**
 2. **Data Transfer**
 3. **Connection Termination**
-

1 5 What is meant by TCP flow control?

- It ensures that the **sender does not overwhelm the receiver**.
 - Controlled by the **window size** field in the TCP header.
 - It maintains **smooth and reliable data transmission**.
-

PART B — Detailed / Explanation-Type Questions

★ 1. Explain in detail the TCP packet format.

Answer:

- A TCP packet is known as a **segment**.
 - It has a **header (20–60 bytes)** followed by **data**.
 - The main fields in the TCP header are:
 - **Source Port & Destination Port**: Identify sending and receiving applications.
 - **Sequence Number**: Number assigned to each byte for ordering.
 - **Acknowledgment Number**: Confirms received data.
 - **Data Offset**: Indicates header length.
 - **Flags**: Control bits (SYN, ACK, PSH, RST, FIN, URG).
 - **Window Size**: Used for flow control.
 - **Checksum**: Error checking.
 - **Urgent Pointer**: Points to urgent data (if URG flag is set).
 - Together, these fields provide **reliability, sequencing, and error control**.
-

★ 2. Explain the HTTP request and response format.

Answer:

- **HTTP Request Message:**
 - Contains a **request line**, **header fields**, and an optional **body**.
 - Example:
 - GET /index.html HTTP/1.1
 - Host: www.facebook.com
 - User-Agent: Chrome
 - **HTTP Response Message:**
 - Contains a **status line**, **headers**, and sometimes a **body**.
 - Example:
 - HTTP/1.1 200 OK
 - Content-Type: text/html
 - Content-Length: 5120
 - This exchange represents one **HTTP transaction**, where the client requests and the server responds with the desired content.
-

★ 3. Explain how TCP connection is established using three-way handshake.

Answer:

- TCP uses a **three-way handshake** to establish a connection:
 1. **SYN**: Client sends a synchronization request to server.
 2. **SYN-ACK**: Server acknowledges and sends its own SYN.
 3. **ACK**: Client acknowledges the server's SYN.
 - After these steps, a reliable connection is established.
 - It ensures both parties are **ready, synchronized, and can start communication safely**.
-

★ 4. Explain the difference between HTTP and HTTPS.

Answer:

- **HTTP** transmits data in **plain text**, while **HTTPS** encrypts data using **SSL/TLS**.
 - **HTTPS** ensures **data security, privacy, and integrity**.
 - Default ports: HTTP → 80, HTTPS → 443.
 - **HTTPS** is used for **secure online transactions and login pages**.
-

★ 5. What are the main features of TCP?

Answer:

1. **Connection-Oriented**: Requires connection setup before data transfer.
 2. **Reliable Transmission**: Uses acknowledgment and retransmission.
 3. **Error Control**: Detects and corrects corrupted segments.
 4. **Flow Control**: Prevents receiver overload.
 5. **Congestion Control**: Adjusts sending rate to avoid network congestion.
 6. **Full Duplex**: Allows two-way simultaneous communication.
-

✓ Conclusion (for viva end):

"In this experiment, we successfully captured and analyzed **TCP and HTTP packets** using Wireshark. We observed how data flows between client and server, learned to apply **filters for SYN, PSH, RST**, and understood how TCP ensures **reliable communication** and how **HTTP handles web transactions**."

Would you like me to make a **beautiful printable PDF (with headings, colors, and table formatting)** of these viva questions and answers for submission or oral practice?

Perfect  Let's go step-by-step. Below are all **important and possible oral/viva questions** from your **Assignment No:9 (HTTP, HTTPS & FTP Protocols using Packet Tracer)** — divided into:

-  *Short/Direct questions* (2–3 lines answers)
 -  *Important/Conceptual questions* (explained in simple detailed way, 5–7 lines)
-

BASIC ORAL QUESTIONS AND ANSWERS

1 What is HTTP?

Answer:

HTTP stands for **Hypertext Transfer Protocol**.

It is used for communication between a **web client (browser)** and a **web server**.

It transfers web pages, images, and other data over the **Internet using TCP port 80**.

It is **stateless** and **connectionless**, meaning each request is independent.

2 What is HTTPS?

Answer:

HTTPS stands for **Hypertext Transfer Protocol Secure**.

It is the **secure version of HTTP**, using **SSL/TLS encryption** to protect data.

It works on **port 443** and ensures data privacy, authentication, and security.

It is mainly used in **banking, e-commerce, and login pages**.

3 What is FTP?

Answer:

FTP means **File Transfer Protocol**.

It is used to **transfer files** between client and server systems over a network.

It works on **port 21 (control)** and **port 20 (data)**.

It allows users to upload, download, delete, and rename files.

4 What are the advantages of HTTPS?

Answer:

- Provides **encryption** — protects data from hackers.
 - Ensures **authentication** of the website.
 - Builds **trust** for users, especially for online transactions.
 - Protects login credentials and sensitive information.
-

5 What are the disadvantages of HTTPS?

Answer:

- Requires purchase of an **SSL certificate**.
 - Slightly **slower** than HTTP because of encryption.
 - Internal links need to be **updated** when switching from HTTP to HTTPS.
-

6 Differentiate between HTTP and HTTPS (*important question — detailed answer*)

Answer:

Feature **HTTP**

HTTPS

Full form Hypertext Transfer Protocol Hypertext Transfer Protocol Secure

Feature	HTTP	HTTPS
Port	80	443
Security	Data is not encrypted	Data is encrypted using SSL/TLS
Speed	Faster (no encryption)	Slower (due to encryption)
Use case	Normal web pages	Banking, e-commerce, login forms
Trust level	Less secure	More secure and trusted

In short:

HTTPS = HTTP + Security (SSL/TLS). It prevents hacking and data leakage.

7 What are the main features of HTTP?

✓ Answer:

- **Connectionless** – Each request is made on a new connection.
 - **Stateless** – Server doesn't remember previous client requests.
 - **Media Independent** – Can transfer any type of data (text, images, audio, etc.).
-

8 What is the port number of HTTP, HTTPS, and FTP?

✓ Answer:

- HTTP → **Port 80**
 - HTTPS → **Port 443**
 - FTP → **Port 21** (control) and **Port 20** (data)
-

9 What do you mean by “HTTP is stateless”? ☀️ (important)

✓ Answer:

HTTP being **stateless** means each request and response is **independent**.

The server doesn't remember the client after the response is sent.

For example, if you visit a new page, it's treated as a new connection.

This makes HTTP simple but less efficient for sessions (cookies are used to manage this).

10 Why is HTTPS more secure than HTTP? ☀️ (important)

✓ Answer:

HTTPS uses **SSL (Secure Socket Layer)** or **TLS (Transport Layer Security)** to **encrypt** all data exchanged between browser and server.

This means even if someone intercepts the data, they can't read it.

It also provides **authentication**, ensuring you're connected to the right website.

That's why banking and e-commerce websites always use HTTPS.

11 What is SSL/TLS?

✓ Answer:

SSL (Secure Socket Layer) and TLS (Transport Layer Security) are **security protocols**.

They **encrypt data**, ensuring it cannot be intercepted or modified during transmission.

TLS is the newer and stronger version of SSL.

12 Explain the working of FTP. ☀️ (important)

✓ Answer:

FTP works on a **client-server model**.

- The client connects to the FTP server using **username and password**.
- It uses **port 21 for control** and **port 20 for data transfer**.
- After login, users can upload (put) or download (get) files.
- Commands like ls, delete, and rename are also supported.

In Packet Tracer, we configure an FTP server and use an FTP client (PC) to transfer files.

1 3 How do you configure FTP in Packet Tracer? ☀️ (*important and practical*)

✓ Answer:

1. Create a topology with a **Laptop** and **Server**.
2. Assign IPs:
 - Laptop → 192.168.1.1
 - Server → 192.168.1.2
3. Enable **FTP service** on the server.
4. On Laptop, open command prompt and type:
ftp 192.168.1.2
5. Login with username: cisco, password: cisco.
6. Use commands like put filename to upload a file or get filename to download.

1 4 What is the purpose of port numbers in protocols?

✓ Answer:

Port numbers help identify **specific services** on a server.

For example, HTTP uses port 80, HTTPS uses 443, FTP uses 21.

They allow multiple applications to run simultaneously on the same machine.

1 5 What is meant by client-server architecture? ☀️ (*important*)

✓ Answer:

In **client-server architecture**, the **client** requests services and the **server** provides them.

- Example: Web browser (client) requests a webpage, web server responds.
- It follows a **request-response model**.
- This architecture is used in HTTP, HTTPS, and FTP communications.

1 6 What is MIME type in HTTP?

✓ Answer:

MIME (Multipurpose Internet Mail Extensions) defines the type of data being sent.

It tells the browser how to handle the file (e.g., text/html, image/jpeg, audio/mp3).

1 7 What is the difference between HTTP being connectionless and stateless?

✓ Answer:

- **Connectionless:** A new connection is made for every request.
- **Stateless:** The server doesn't remember previous requests after closing the connection.

1 8 What tools are used to analyze HTTP, HTTPS, and FTP performance?

✓ Answer:

Tools like **Cisco Packet Tracer**, **Wireshark**, or **Postman** can be used to analyze packet flow, connection time, and data security for these protocols.

1 9 Why is HTTPS slower than HTTP?

Answer:

Because HTTPS requires **encryption and decryption** of data using SSL/TLS, which adds extra steps in communication, making it slightly slower.

2 0 What command is used to upload a file using FTP?

Answer:

put filename — This uploads the file from the client to the server.

Example: put MyFile.txt

2 1 What command is used to download a file using FTP?

Answer:

get filename — This downloads the file from the server to the client.

Example: get sample.txt

2 2 What did you conclude from this experiment? (*important*)

Answer:

We successfully analyzed the **performance and security** of HTTP, HTTPS, and FTP protocols.

We observed that:

- HTTP is simple and fast but **not secure**,
- HTTPS adds **encryption and authentication**,
- FTP allows **file transfer** between devices.

This helped in understanding **real-world internet communication** protocols.

2 3 What is the default username and password in Packet Tracer FTP service?

Answer:

Default **username: cisco, password: cisco**

2 4 Which protocol would you recommend for banking or e-commerce websites? Why?

(important)

Answer:

HTTPS, because it encrypts all transmitted data, protecting sensitive information like **credit card details or passwords**.

It ensures **data integrity, authentication, and user trust**.

2 5 What is the role of TCP in HTTP, HTTPS, and FTP?

Answer:

TCP (Transmission Control Protocol) provides a **reliable connection** for data exchange.

It ensures that packets arrive **in order and without loss** for HTTP, HTTPS, and FTP communications.

Would you like me to make this into a **PDF viva sheet** (formatted neatly with bold keywords, bullet points, and proper headers for printing or submission)?

Sure Here's a **complete list of oral/viva questions with simple, clear, and point-wise answers** for your **Assignment No. 10: SSL Protocol using Wireshark**.

I've marked **important** questions with and written **detailed answers** for them.

Basic Questions

1. What is SSL protocol?

Answer:

- SSL (Secure Sockets Layer) is an **encryption-based Internet security protocol**.
 - It was developed by **Netscape in 1995** to ensure **privacy, authentication, and data integrity**.
 - It **encrypts** data transmitted between the client (browser) and server.
 - SSL makes sure that only the intended recipient can read the data.
 - It was later replaced by a more secure version called **TLS (Transport Layer Security)**.
-

2. Why is SSL/TLS important?

Answer:

- SSL/TLS ensures **secure communication** between a user and a website.
 - It **encrypts data**, so hackers cannot read it even if intercepted.
 - It **authenticates** websites, preventing fake or malicious websites.
 - It ensures **data integrity**, meaning the data cannot be changed during transmission.
 - For example, when you enter a credit card number on a shopping site, SSL keeps it safe and hidden from attackers.
-

3. What is the main purpose of SSL?

Answer:

To provide **secure, private, and reliable communication** over the Internet by encrypting data and verifying the identity of servers and clients.

4. What does SSL stand for?

Answer: Secure Sockets Layer.

5. Who developed SSL and when?

Answer: SSL was developed by **Netscape Communications in 1995**.

6. What are the main functions of SSL?

Answer:

- **Encryption:** Protects data during transfer.
 - **Authentication:** Verifies server identity.
 - **Data Integrity:** Ensures no data tampering.
-

7. What is the SSL handshake process?

Answer (Detailed):

The **SSL handshake** is a series of steps where the client and server agree on how to communicate securely.

Steps include:

1. **Client Hello:** The client sends supported encryption methods.
 2. **Server Hello:** The server chooses the method and sends its digital certificate.
 3. **Authentication:** The client verifies the server's certificate.
 4. **Key Exchange:** Both sides generate a shared secret key.
 5. **Session Encryption:** All further communication is encrypted using that key.
-

8. What is TLS and how is it related to SSL?

Answer:

- TLS (Transport Layer Security) is the **updated and improved version of SSL**.
 - TLS was introduced by **IETF in 1999**.
 - It provides **stronger encryption and better security**.
 - Both are used for secure communication, so people often say **SSL/TLS** together.
-

9. What is the difference between SSL and TLS?

Answer:

Feature	SSL	TLS
Developer	Netscape	IETF
Introduced	1995	1999
Security	Less secure	More secure
Latest Version	SSL 3.0	TLS 1.3
Usage	Outdated	Currently used

10. What are the benefits of SSL?

Answer:

- Secures user data.
 - Builds trust and shows the site is legitimate.
 - Protects against phishing and data theft.
 - Improves SEO ranking (Google prefers HTTPS).
-

11. What is HTTPS?

Answer:

- HTTPS = HTTP + SSL/TLS.
 - It's the **secure version of HTTP**.
 - All data between browser and server is encrypted.
 - Websites using HTTPS show a **lock symbol**  in the browser.
-

12. ⭐ How does Wireshark help in studying SSL?

Answer (Detailed):

- Wireshark is a **network packet analyzer** used to capture and inspect network traffic.
 - When visiting an SSL-secured website, Wireshark captures the **handshake packets** and **encrypted data**.
 - You can observe steps like **Client Hello**, **Server Hello**, and **Certificate Exchange**.
 - It helps understand **how encryption and authentication** happen in real-time.
 - However, actual encrypted data can't be read — only the handshake details are visible.
-

13. What is encryption?

Answer:

Encryption is the process of **converting readable data (plaintext)** into an **unreadable format (ciphertext)** so that unauthorized users cannot read it.

14. What is decryption?

Answer:

Decryption is the **reverse process** of encryption — it converts the **ciphertext back into readable plaintext** using a secret key.

15. What is a digital certificate?

Answer:

- A digital certificate is like an **online identity card** issued by a trusted organization (CA).
 - It contains the **website's public key**, organization name, and domain info.
 - It helps verify that the website is **authentic and trustworthy**.
-

16. What is meant by data integrity in SSL?

Answer:

Data integrity means that the **data is not changed or tampered with** during transmission between sender and receiver.

17. What is the role of authentication in SSL?

Answer:

Authentication ensures that both the **client and server are genuine** and not imposters. This prevents man-in-the-middle attacks.

18. How can you identify a website is SSL-secured?

Answer:

- The website URL starts with **https://**
 - A **padlock (🔒)** icon appears before the URL in the browser.
 - Clicking it shows the certificate details.
-

19. ⭐ What happens if a website doesn't use SSL?

Answer (Detailed):

- Data travels in **plain text**, which hackers can easily read.
 - Attackers can **steal sensitive information** like passwords and credit card numbers.
 - Fake websites can **impersonate** real ones and perform phishing attacks.
 - Browsers mark such sites as "**Not Secure**", reducing trust.
-

20. What type of attacks does SSL/TLS prevent?

Answer:

- **Eavesdropping (data theft)**
 - **Man-in-the-middle attacks**
 - **Data tampering**
 - **Phishing (fake websites)**
-

21. What is the current version of TLS?

Answer: TLS 1.3 (released in 2018).

22. What tools can be used to analyze SSL packets?

Answer: Wireshark, Fiddler, and OpenSSL are commonly used tools.

23. What layer does SSL/TLS operate at?

Answer:

SSL/TLS operates between the **Application layer** and **Transport layer** — often called the **Presentation layer** in the OSI model.

24. What are the key elements of SSL communication?

Answer:

- Certificates
 - Keys (public/private)
 - Encryption algorithms
 - Handshake process
-

25. ⭐ What did you observe in Wireshark while visiting an SSL website?

Answer (Detailed):

- The **Client Hello** packet from the browser.
 - The **Server Hello** and **Certificate** packets from the website.
 - Key exchange and encryption negotiation.
 - Data packets showing **encrypted content (TLSv1.2 or TLSv1.3)**.
 - The communication remains **secure and unreadable** to outsiders.
-

26. What is meant by “certificate authority”?

Answer:

A **Certificate Authority (CA)** is a trusted organization that issues digital certificates (e.g., DigiCert, Let's Encrypt, GoDaddy).

27. What happens if a certificate is invalid or expired?

Answer:

- The browser shows a **warning** like “Your connection is not private.”
 - It means the website may not be trustworthy or secure.
-

28. What are the versions of SSL?

Answer:

- SSL 1.0 (never released)
 - SSL 2.0
 - SSL 3.0 (final version before TLS)
-

29. What replaced SSL?

Answer: TLS (Transport Layer Security) replaced SSL for better security.

30. What is meant by “cipher suite” in SSL/TLS?

Answer:

A **cipher suite** is a set of algorithms that define how encryption, decryption, and authentication are performed in SSL/TLS communication.

✓ CONCLUSION (For Oral Summary)

In this experiment, we studied how the SSL/TLS protocol provides secure communication between client and server.

Using **Wireshark**, we captured and analyzed SSL handshake packets.

We observed how **data encryption**, **authentication**, and **integrity** protect user information from

hackers.

Finally, we understood the importance of **HTTPS** and why **TLS has replaced SSL** in modern security systems.

Would you like me to make a **short oral summary paragraph (2–3 mins explanation)** for you to speak during viva?

Perfect  — this is a **Computer Networks Lab Assignment on Installing and Configuring a DHCP Server**. Below are **all important oral/viva questions** that can be asked on this experiment, along with **clear and simple answers**. I've also given **detailed explanations** for important ones.

ORAL QUESTIONS & ANSWERS (DHCP SERVER EXPERIMENT)

1. What is DHCP? (*Important – explain clearly*)

Answer:

- DHCP stands for **Dynamic Host Configuration Protocol**.
 - It is a **network management protocol** used to **automatically assign IP addresses** and other network parameters to client devices.
 - It eliminates the need for **manual IP configuration** on each computer.
 - The DHCP server dynamically gives IP addresses, subnet mask, default gateway, and DNS information to clients.
 - It ensures that each client gets a **unique IP address** on the network.
-

2. Why do we need DHCP? (*Important*)

Answer:

- Without DHCP, the **network administrator must manually assign IP addresses**, which is time-consuming and error-prone.
 - DHCP **automates this process**, saving time and reducing IP conflicts.
 - It's especially useful for **large networks** with many computers.
 - It allows easy **IP reuse** when devices disconnect.
 - It ensures proper **centralized management** of IP addressing.
-

3. How does DHCP work? (*Very Important – detailed answer*)

Answer:

DHCP works through a **four-step process** known as the **DORA process**:

1. **D – Discover:**
The client broadcasts a “DHCP Discover” message to find available DHCP servers.
2. **O – Offer:**
The DHCP server responds with a “DHCP Offer” message containing an available IP address and configuration.
3. **R – Request:**
The client sends a “DHCP Request” message to accept the offered IP address.
4. **A – Acknowledge:**
The server sends a “DHCP Acknowledgement” confirming that the IP is assigned.

 After this, the client can use the assigned IP to communicate in the network.

4. What are the components of DHCP?

Answer:

1. **DHCP Server:** Assigns IPs and network configurations.
 2. **DHCP Client:** The device that requests and receives IP configuration.
 3. **IP Address Pool:** Range of IPs available for distribution.
 4. **Lease:** Duration for which an IP is assigned.
 5. **Scope:** Defines a range of IPs and related options on the DHCP server.
-

5. What is a DHCP Scope? (*Important*)

Answer:

- A **scope** is a **range of IP addresses** that the DHCP server can assign to clients.
 - It defines **start and end IP addresses, subnet mask, gateway, and DNS settings**.
 - Each DHCP server should have **at least one scope** that does not overlap with others.
 - Example: Scope 192.168.100.1 – 192.168.100.100.
-

6. What is a DHCP Lease?

Answer:

- A **lease** is the **time period** for which an IP address is assigned to a client.
 - After the lease expires, the client must **renew** it to continue using the IP.
 - Example: Lease time = 8 hours or 1 day.
-

7. What is the difference between DHCP and Static IP Addressing? (*Important*)

Feature	DHCP	Static IP
IP Assignment	Automatic by DHCP server	Manually configured
Management	Centralized, easy	Time-consuming
IP Conflict	Less chance	High if manual error
Best For	Large networks	Small or specific servers
Example	Offices, schools	Routers, servers

8. What are the main advantages of DHCP?

Answer:

- Reduces manual configuration.
 - Avoids IP conflicts.
 - Efficient IP address management.
 - Supports mobility (devices can move easily).
 - Centralized network management.
-

9. What are the limitations or disadvantages of DHCP?

Answer:

- If the DHCP server fails, clients cannot get IPs.
 - Not suitable for critical systems requiring permanent IPs.
 - Security risk — **rogue DHCP servers** can assign wrong configurations.
-

10. What is a Rogue DHCP Server? (*Important for oral*)

Answer:

- A **rogue DHCP server** is an **unauthorized server** that gives wrong IP configurations.
- It can cause **network issues** or **redirect traffic** for malicious purposes.

- In Windows Server, DHCP servers must be **authorized in Active Directory** to prevent rogue servers.
-

11. What information does DHCP assign to clients?

Answer:

DHCP assigns the following:

1. IP Address
 2. Subnet Mask
 3. Default Gateway
 4. DNS Server Address
 5. WINS Server (optional)
 6. Lease Time
-

12. What happens if the DHCP server is down?

Answer:

- New clients **cannot obtain IP addresses**.
 - Existing clients continue using their IPs until the **lease expires**.
 - After expiry, they lose connectivity until the DHCP server comes back online.
-

13. What is IP Address Exclusion in DHCP?

Answer:

- Exclusion means **reserving certain IP addresses** within the scope that the DHCP server should **not assign dynamically**.
 - These IPs can be used for **servers, printers, routers** with static IPs.
 - Example: Exclude 192.168.100.1 – 192.168.100.10.
-

14. What are the steps to install the DHCP service on Windows Server 2003?

Answer:

1. Go to **Control Panel** → **Add or Remove Programs** → **Add/Remove Windows Components**.
 2. Select **Networking Services** → **Details** → **DHCP**.
 3. Click **Next**, insert CD if required, and wait for installation.
 4. Click **Finish** once setup completes.
-

15. What are the steps to configure a DHCP scope?

Answer:

1. Open **Administrative Tools** → **DHCP**.
 2. Right-click the server → **New Scope** → **Next**.
 3. Enter **scope name and IP range**.
 4. Add **exclusion range, lease duration, and gateway**.
 5. Configure **DNS and WINS** if required.
 6. **Activate scope and authorize** the DHCP server.
-

16. What is DHCP Authorization and why is it required? (*Important*)

Answer:

- **Authorization** ensures that only trusted DHCP servers in a domain can assign IP addresses.
- It prevents **rogue DHCP servers** from operating.
- In Windows Server, only **domain administrators** can authorize a DHCP server through **Active Directory**.

17. What is a Default Gateway?

Answer:

- It is the **router's IP address** that connects the local network to other networks or the internet.
 - DHCP provides this address automatically to clients.
-

18. What is the difference between DHCP Discover and DHCP Offer?

Answer:

- **DHCP Discover:** Message sent by the client to find available DHCP servers.
 - **DHCP Offer:** Reply sent by the DHCP server offering an available IP address.
-

19. Which port number does DHCP use?

Answer:

- DHCP uses **UDP Port 67** for the server and **UDP Port 68** for the client.
-

20. What is a DHCP Relay Agent?

Answer:

- A device or router that **forwards DHCP requests** from clients in one network to a DHCP server in another network.
 - Used when the client and server are on **different subnets**.
-

21. What is the function of the DHCP database?

Answer:

- It stores information about **active leases, reservations, and scopes**.
 - Helps the server remember which IPs are in use and which are free.
-

22. What is the lease renewal process in DHCP?

Answer:

- Clients request renewal at **50% of the lease time** by sending a **DHCP Request** directly to the server.
 - If no response, it retries at **87.5% of lease time**.
 - If still no response, the client stops using the IP after lease expiry.
-

23. What are DHCP Options?

Answer:

- Additional parameters that the DHCP server provides to clients.
 - Includes **gateway address, DNS, domain name, WINS, etc.**
-

24. What is the difference between DHCP and DNS?

DHCP

DNS

Assigns IP addresses automatically Translates domain names to IP addresses

Works during IP assignment

Works during name resolution

Uses UDP Ports 67, 68

Uses UDP/TCP Port 53

25. What is the conclusion of this experiment?

Answer:

- We successfully installed and configured a **DHCP server**.
 - We learned how to create scopes, assign IPs, and authorize the DHCP server.
 - The clients were automatically assigned IP addresses without manual configuration.
-

Would you like me to make a **printable PDF version** of these viva questions and answers (formatted neatly for submission or oral preparation)?

Perfect  Let's prepare a **complete oral preparation guide** for your **Assignment No. 12 – DNS Lookup**.

Below are **all important viva/oral questions with simple, clear, and detailed answers (where required)** — easy to speak and remember 

Assignment 12: DNS Lookup

◆ **1. What is DNS? (Most Important)**

Answer:

- DNS stands for **Domain Name System**.
 - It is like the **phonebook of the Internet**.
 - It converts **domain names (like google.com)** into **IP addresses (like 142.250.190.78)** and vice versa.
 - Computers understand IP addresses, not names. DNS helps users access websites using easy names instead of numbers.
 - Without DNS, we would have to remember long IP addresses for every website.
-

◆ **2. Why do we need DNS?**

Answer:

- Humans prefer **names** (like yahoo.com), but computers use **IP addresses**.
 - DNS provides a **mapping between names and IP addresses**.
 - Earlier, mapping was stored in a **hosts.txt** file, but as the Internet grew, it became impossible to maintain manually.
 - So, DNS was created to **distribute this mapping** among many servers for efficiency.
-

◆ **3. What is DNS Lookup?**

Answer:

- DNS lookup means **finding the IP address of a domain name** or **finding the domain name from an IP address**.
 - Example:
 - Forward lookup: www.google.com → 142.250.190.78
 - Reverse lookup: 142.250.190.78 → www.google.com
-

◆ **4. What are the two types of DNS lookups?**

Answer:

1. **Forward Lookup:** Domain name → IP address.
 2. **Reverse Lookup:** IP address → Domain name (uses **PTR records**).
-

◆ **5. What is the objective of this experiment?**

Answer:

- To learn how DNS works.
- To get the **hostname and IP address** of a system.

- To perform **mapping between name and IP address** (both directions).
-

◆ **6. What is the outcome of the experiment?**

Answer:

- Students can summarize DNS lookup.
 - They learn how to get hostnames and IP addresses using DNS functions or commands.
 - Understand how Internet name resolution works.
-

◆ **7. What is a Host File and why is it not used today?**

Answer:

- A host file is a **simple text file** containing name–address pairs.
 - It was used in early networks for name resolution.
 - But now the Internet has **millions of hosts**, so maintaining one global file is **impossible**.
 - DNS replaced it with a **distributed system** that stores mappings on multiple servers.
-

◆ **8. What are the main components of DNS?**

Answer:

1. **Domain Name Space:** The structure or hierarchy of domain names.
 2. **DNS Servers:** Store and manage domain name information.
 3. **Resolvers:** Client programs that send DNS queries and receive responses.
-

◆ **9. What is Name Space in DNS?**

Answer:

- The **Name Space** defines the way names are structured and assigned.
 - It can be of two types:
 1. **Flat Name Space:** Simple, no structure (unsuitable for large networks).
 2. **Hierarchical Name Space:** Organized like a tree, with subdomains and top-level domains.
-

◆ **10. What is a Domain Name Space? (Detailed Answer)**

Answer:

- Domain Name Space is a **hierarchical structure** used to organize all domain names.
 - It looks like an **inverted tree**, where:
 - The **root** is at the top (represented by a dot ".").
 - Below root are **Top-Level Domains (TLDs)** like .com, .org, .in, etc.
 - Then come **Second-Level Domains** (like google.com) and subdomains (like mail.google.com).
 - This structure ensures **uniqueness** and **easy delegation** of domain control.
-

◆ **11. What is a Domain?**

Answer:

- A **domain** is a subtree of the domain name space.
 - Example: In “example.com”, “example” is a domain under the top-level domain “.com”.
 - Domains can have **subdomains** like “mail.example.com”.
-

◆ **12. What is a Resolver in DNS? (Important)**

Answer:

- A **resolver** is a **DNS client program** that sends queries to DNS servers.
 - When you type a website name, your computer's resolver asks a DNS server for its IP address.
 - It interprets the server's response and gives the final result to the application (like browser).
-

◆ 13. What is Name Resolution?

Answer:

- The process of **mapping a name to an address or an address to a name**.
 - Example:
 - Name to address → www.google.com → 142.250.190.78
 - Address to name → 142.250.190.78 → www.google.com
-

◆ 14. What is Recursive Resolution? (Detailed)

Answer:

- In recursive resolution, the **DNS server** does **all the searching** for the client.
 - If the server doesn't have the answer, it asks other DNS servers until it gets the final result.
 - Once the answer is found, it sends it back to the resolver.
 - It's like the server doing all the work for the client.
-

◆ 15. What is Iterative Resolution?

Answer:

- In iterative resolution, the **DNS server gives the best answer it has**.
 - If it doesn't know the final answer, it refers the client to another DNS server.
 - The **client** then queries that referred server.
 - So here, the client does the work.
-

◆ 16. What is a DNS Server?

Answer:

- A DNS server is a **computer that stores and manages domain name records**.
 - It helps convert domain names into IP addresses.
 - Common types:
 - **Root Server**
 - **Top-Level Domain (TLD) Server**
 - **Authoritative DNS Server**
 - **Local DNS Server**
-

◆ 17. What are Root Servers?

Answer:

- Root servers are **top-level DNS servers** that know where to find information about all top-level domains (.com, .org, .net, etc.).
 - There are **13 main root servers** worldwide.
-

◆ 18. What is a DNS Record?

Answer:

- DNS records are stored in zone files on DNS servers.
- They store information about domains.
 - **A Record:** Maps domain name to IP address.

- **PTR Record:** Maps IP address to domain name (reverse lookup).
 - **MX Record:** Mail exchange record (email).
 - **CNAME:** Canonical name (alias).
 - **NS:** Name server record.
-

◆ **19. Explain the DNS Resolution Process (Step-by-Step)**

Answer (Detailed & Important):

1. User enters a website name (e.g., www.example.com) in the browser.
 2. The resolver checks if the IP is cached locally.
 3. If not, it sends a request to the **local DNS server**.
 4. The local DNS server asks the **root server**.
 5. The root server refers it to a **TLD server (.com)**.
 6. The TLD server refers it to the **authoritative DNS server** for example.com.
 7. The authoritative server responds with the **IP address**.
 8. The local DNS server caches it and sends it to the resolver.
 9. The browser connects to the IP and loads the site.
-

◆ **20. What is the difference between Flat and Hierarchical Name Space?**

Feature Flat Name Space Hierarchical Name Space

Structure No structure Tree-like structure

Suitable for Small networks Large Internet

Control Centralized Distributed

Example Early host file DNS system

◆ **21. What is the difference between Forward and Reverse DNS Lookup?**

Type	Input	Output	Record Type
------	-------	--------	-------------

Forward Lookup	Domain name	IP address	A record
----------------	-------------	------------	----------

Reverse Lookup	IP address	Domain name	PTR record
----------------	------------	-------------	------------

◆ **22. What happens when you type www.google.com in your browser?**

Answer (Short version):

- The browser asks the local resolver for the IP.
 - If not found, resolver contacts DNS servers (root → TLD → authoritative).
 - IP address is returned.
 - Browser connects to that IP and loads Google's website.
-

◆ **23. What is the practical program for DNS lookup?**

Answer:

A simple Python example:

```
import socket
host = input("Enter website name or IP: ")
try:
    print("IP Address:", socket.gethostbyname(host))
except:
    print("Host Name:", socket.gethostbyaddr(host))
```

It performs forward or reverse lookup using socket library.

◆ **24. What is a Top-Level Domain (TLD)?**

Answer:

- The last part of a domain name (like .com, .org, .in).
 - It defines the **type or country** of the organization.
 - Examples: .edu (education), .gov (government), .in (India).
-

◆ **25. What is a Subdomain?**

Answer:

- A smaller part of a main domain.
 - Example: mail.google.com → “mail” is a subdomain of “google.com”.
-

◆ **26. What are the advantages of DNS?**

Answer:

- Easy to remember names instead of IPs.
 - Decentralized system → less load.
 - Automatic name resolution.
 - Easily scalable for Internet growth.
-

◆ **27. What is a PTR record?**

Answer:

- A **Pointer Record (PTR)** is used for **reverse DNS lookup**.
 - It maps an IP address to a domain name.
-

◆ **28. What is Caching in DNS?**

Answer:

- DNS servers store recently resolved domain names temporarily.
 - It **reduces lookup time** and **server load** for repeated requests.
-

◆ **29. What is the conclusion of your experiment?**

Answer:

- The DNS Lookup program was successfully executed.
 - It helped understand how names are mapped to IP addresses and vice versa.
 - We learned about DNS structure, resolution, and lookup process.
-

Would you like me to include a **short viva summary sheet (1-page version)** that you can revise before oral — containing **only most expected 10 questions** with crisp answers?