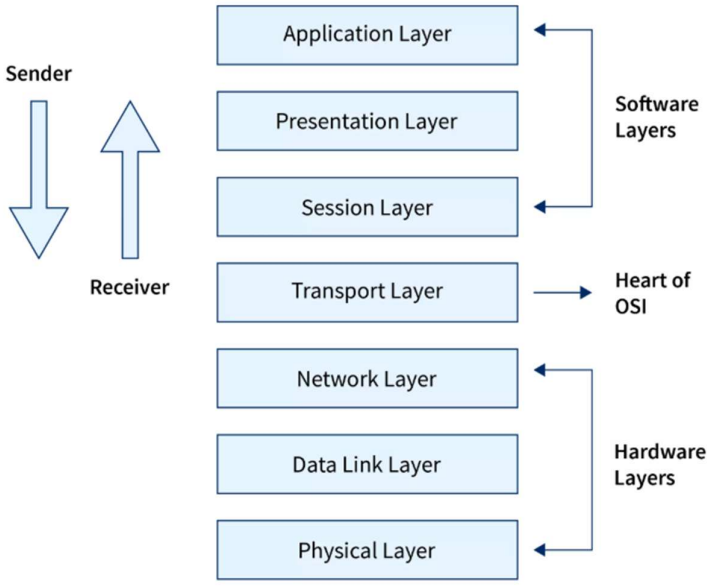
Task 1: Study **OSI & TCP/IP models**
Task 2: Learn common protocols – **HTTP, FTP, DNS, DHCP, SSH**
Task 3: Practice basic commands – ping, traceroute, netstat, ipconfig/ifconfig
Task 4: Write a short report explaining **client-server communication**

❖ Task 1: Study OSI & TCP/IP models

What is OSI Model?

The OSI model stands for Open Systems Interconnection model. The OSI model is also known as the ISO-OSI model as it was developed by ISO (International Organization for Standardization).



----------OSI Model Framework----------

1. Application Layer

The Application layer is where the end-user directly interacts with the network through software applications like browsers, email, or messaging apps.

- **User Interaction**: It's the layer where users directly interact with applications like web browsers, email, and file transfer.

- **Network Services:** Provides services such as HTTP, FTP, SMTP, and DNS that enable communication between applications across networks.
- **Data Preparation:** Ensures data is ready to be sent or received by handling things like formatting, compression, and encryption.
- Protocols: **HTTP, HTTPS, FTP, SMTP, DNS, SSH**

2. Presentation Layer

The Presentation layer acts as a translator and protector, making sure the data is properly formatted and secure before it reaches the application.

- **Translation:** Ensures that the data format used by the sender is compatible with the format the receiver understands (e.g., ASCII to EBCDIC).
- **Data Compression:** Compresses data streams to improve speed, reduce storage requirements, and make communication more efficient.
- **Decryption:** Takes encrypted data received from the network and converts it back into its original readable form for the application.
- Protocols: **SSL/TLS, JPEG, MP3, ASCII**

3. Session Layer

The Session layer establishes, manages, and ends communication sessions between two systems, like keeping track of login sessions or video calls.

- **Session Management:** Creates and manages the “session” or dialogue between two devices, ensuring smooth communication until it ends.
- **Authentication:** Confirms the identity of the user or system, such as verifying a username and password before access.
- **Authorization:** Decides what the authenticated user is allowed to do, such as accessing files, sending emails, or using certain services.
- Protocols: **NetBIOS, PPTP, RPC**

4. Transport Layer

The Transport layer ensures reliable end-to-end communication between two devices, taking care of data integrity, order, and error correction.

- **Segmentation:** Divides large chunks of data into smaller segments for easier transmission and reassembles them at the destination.
- **Flow Control:** Balances the rate of data transfer so the sender doesn't overwhelm the receiver with too much information at once.
- **Error Control:** Checks for lost or corrupted data packets, requests retransmission if needed, and ensures that only correct data is delivered.
- Protocols: **TCP, UDP**

5. Network Layer

The Network layer is responsible for deciding how data packets travel across networks from the source to the destination.

- **Logical Addressing:** Assigns IP addresses to devices so they can be uniquely identified across networks.
- **Routing:** Uses routers to choose the best possible path for the data to travel to its destination.
- **Path Determination:** Evaluates network conditions (like traffic, distance, or cost) and picks the most efficient route for data delivery.
- Protocols: **IP, ICMP, OSPF, BGP**

6. Data Link Layer

The Data Link layer ensures that data can move reliably across a physical link between two directly connected devices.

- **Framing:** Structures raw bits into frames (packets with headers and trailers) so that data is organized for transmission.
- **Addressing:** Uses MAC (Media Access Control) addresses to identify devices within the same local network (like your laptop and router).
- **Error Control:** Detects errors caused by noise or interference in the physical medium and may request retransmission.

- Protocols: **Ethernet, ARP, PPP, VLAN**

7. Physical Layer

The Physical layer deals with the hardware and the physical means of sending raw bits (0s and 1s) over the medium.

- **Transmission Media:** Defines the medium like copper cables, fiber optics, or wireless signals used for communication.
- **Bit Transmission:** Converts digital data into electrical, optical, or radio signals that can travel through the medium.
- **Hardware Components:** Involves network devices like switches, hubs, routers, NIC cards, and connectors that physically transmit signals.
- Protocols: **Ethernet, USB, Bluetooth**

TCP/IP Model:

1. Application Layer

- **Purpose:** This is the layer that users directly interact with through applications like browsers, email clients, and file transfer tools.
- **Functions:**
    - Provides services for applications to communicate over the network.
    - Handles tasks like email sending/receiving, file uploads/downloads, and web browsing.
    - Sometimes includes session management, encryption, and data translation.
- **Common Protocols:**
    - **HTTP/HTTPS:** Web browsing
    - **FTP:** File transfer
    - **SMTP, POP3, IMAP:** Email communication

- **DNS:** Converts website names to IP addresses
- **SSH, Telnet:** Remote login

2. Transport Layer

- **Purpose:** Ensures data is delivered **from one device to another correctly and efficiently**.
- **Functions:**
  - Divides large data into smaller chunks called **segments** and reassembles them at the destination.
  - Uses **port numbers** to make sure the data reaches the right application.
  - Can provide reliable delivery with error checking or faster delivery without checking.
- **Main Protocols:**
  - **TCP (Transmission Control Protocol):** Reliable, connection-oriented, ensures data is received correctly; used for web, email, and file transfers.
  - **UDP (User Datagram Protocol):** Fast, connectionless, no guarantee of delivery; used for streaming video, VoIP, and DNS queries.

3. Internet Layer

- **Purpose:** Responsible for **logical addressing and routing**, making sure data finds the correct device across networks.
- **Functions:**
  - Assigns **IP addresses** to devices.
  - Chooses the **best path** for data packets to travel.
  - Breaks data into packets for transmission.
- **Main Protocols:**

- **IP (IPv4, IPv6):** Addresses and routes packets
- **ICMP (Internet Control Message Protocol):** Sends error messages and testing (like ping)
- **ARP (Address Resolution Protocol):** Converts IP addresses to MAC addresses
- **Routing protocols:** RIP, OSPF, BGP

4. Network Access Layer (Link Layer)

- **Purpose:** Handles the **physical transmission of data** over cables, Wi-Fi, or other hardware.
- **Functions:**
  - Converts packets into **frames and bits** for actual transmission.
  - Ensures devices use the network medium correctly (Ethernet, Wi-Fi, etc.).
  - Handles **hardware addressing** with MAC addresses.
- **Protocols and Technologies:**
  - **Ethernet, Wi-Fi (802.11), Bluetooth**
  - **PPP (Point-to-Point Protocol)**
  - Physical devices: cables, network interface cards, switches

❖ Task 2: Learn common protocols – HTTP, FTP, DNS, DHCP, SSH

What is Protocol?

- A network protocol is a set of rules that allows computers and devices to communicate with each other over a network. It defines how data is formatted, transmitted, and received, ensuring that all devices understand each other.

- Protocols make it possible for different devices, operating systems, and applications to work together smoothly across networks like the internet or a local network.

1. HTTP (HyperText Transfer Protocol)

- HTTP is the main protocol used whenever you access a website. Whenever you type a URL into your browser, your computer uses HTTP to request the page from the server, and the server responds with the website's data (text, images, videos).
- There's also HTTPS, which is a secure version of HTTP. It uses encryption (SSL/TLS) to protect your data so no one can spy on your activities or steal sensitive information like passwords or credit card numbers.
- **Port:** 80 for HTTP, 443 for HTTPS.
- **Example:** Opening google.com, watching videos on YouTube, or accessing web apps like Gmail.
- **Importance:** Without HTTP/HTTPS, web browsing would not be possible, and data would not flow in a structured way between browsers and servers.

2. FTP (File Transfer Protocol)

- FTP is used to move files between computers over a network. If you have a website, you often use FTP to upload files from your computer to the web server.

- FTP works in a client-server manner. You, as the client, connect to the FTP server using credentials. Once connected, you can upload or download files.
- Standard FTP is not secure, so encrypted versions like SFTP (uses SSH) or FTPS (uses SSL/TLS) are preferred.
- **Ports:** 21 for control commands, 20 for transferring data.
- **Example:** Uploading your website files, downloading large datasets from a server.
- **Importance:** Essential for managing files across networks, especially for website hosting and organizational data transfer.

3. DNS (Domain Name System)

- DNS acts like the "phonebook of the internet". Computers work with numbers (IP addresses), but humans prefer names. DNS translates domain names like google.com into IP addresses that computers can understand.
- When you type a website address, your computer asks a DNS server to provide the corresponding IP address. Once received, your browser can connect to the correct server.
- **Port:** 53.
- **Example:** You type www.example.com → DNS converts it to 93.184.216.34.
- **Importance:** Without DNS, we would have to remember numeric IP addresses for every website we visit. It makes the internet user-friendly and navigable.

4. DHCP (Dynamic Host Configuration Protocol)

- DHCP automatically assigns IP addresses and other network details to devices when they join a network.
- Imagine you connect your laptop or phone to Wi-Fi. You don't manually type an IP address; DHCP handles it automatically. It also gives you the

subnet mask, gateway, and DNS servers, so your device can communicate properly.

- **Ports:** 67 (server), 68 (client).
- **Example:** Your smartphone joins home Wi-Fi and receives 192.168.1.5 from the router automatically.
- **Importance:** Makes network management easy, prevents IP conflicts, and allows devices to join and leave networks seamlessly.

5. SSH (Secure Shell)

- SSH allows secure remote access to another computer or server. Unlike Telnet, SSH encrypts all communication so nobody can intercept sensitive commands or passwords.
- Administrators often use SSH to manage Linux servers from anywhere. You can also transfer files securely using SCP or SFTP, which use SSH encryption.
- **Port:** 22.
- **Example:** Logging into a cloud Linux server from your laptop to update software or manage files.
- **Importance:** Essential for remote administration and secure communication, especially for servers and network devices.

❖ Task 3: Practice basic commands – ping, traceroute, netstat, ipconfig/ifconfig

1. ping Command

The 'ping' command checks connectivity between your system and another device/server by sending ICMP Echo Request packets and waiting for Echo Reply.

Main Uses:

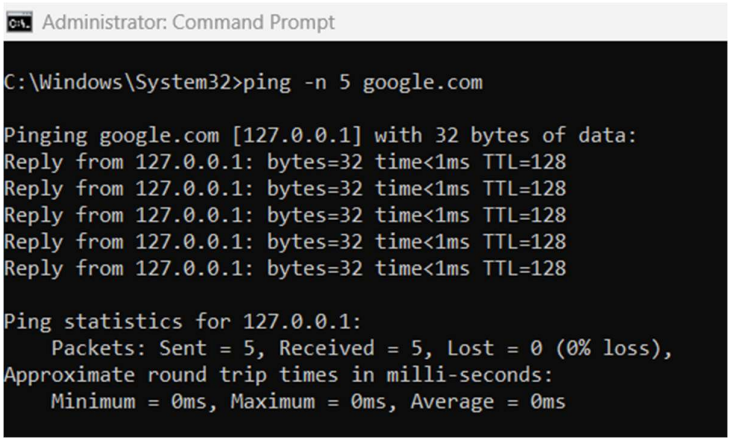
- Test if a host is reachable.
- Check for packet loss.

Commands:

Windows:

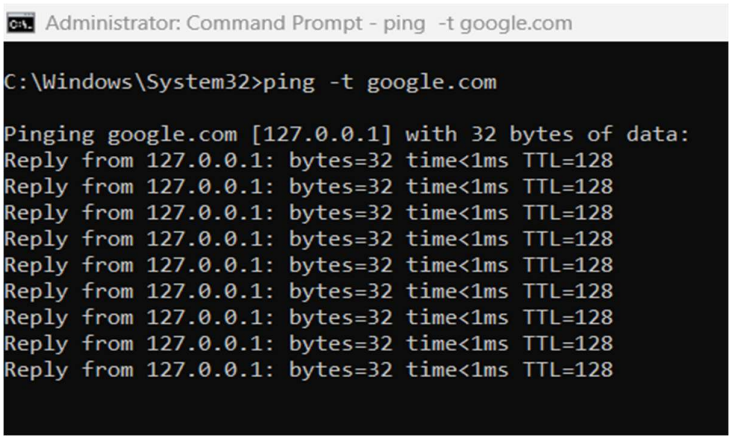
ping -n 5 google.com

- Sends 5 ping requests to google.com and then stops.



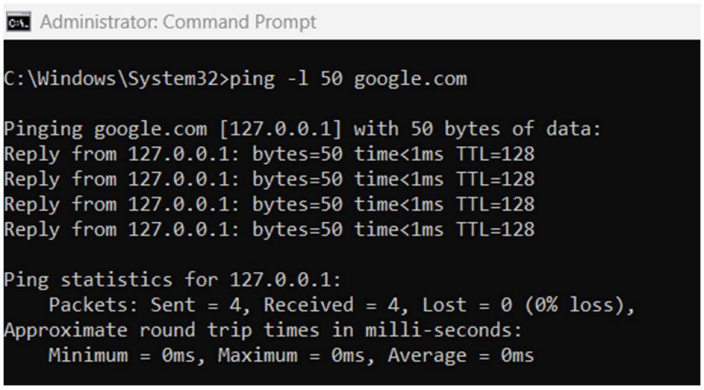
ping -t google.com

- -t = continuous ping.
- Keeps sending pings until you manually stop with Ctrl+C.



ping -l 50 google.com

- -l = packet size in bytes.
- Sends ping packets of 100 bytes instead of default 32 bytes.



Linux:

ping -c 5 google.com
ping -i 2 google.com
ping -s 100 google.com

2. tracert / traceroute Command

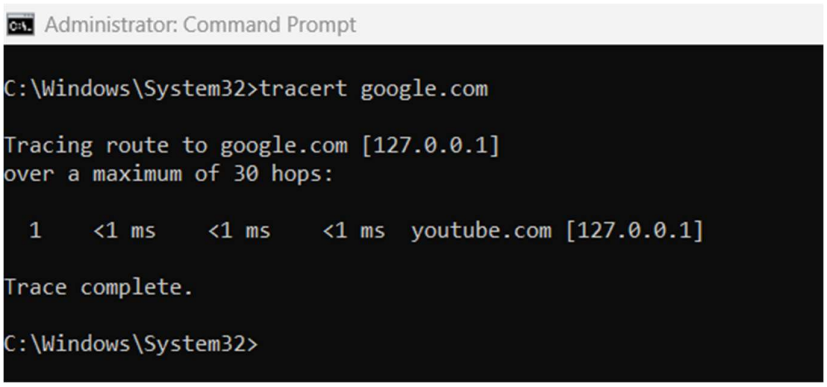
Shows the route packets take to reach a destination, including all intermediate routers.

Main Uses:
- Find where network delays occur.
- Troubleshoot routing problems.
- See how many hops are between you and a server.

Commands:

Windows:

tracert google.com



Linux:

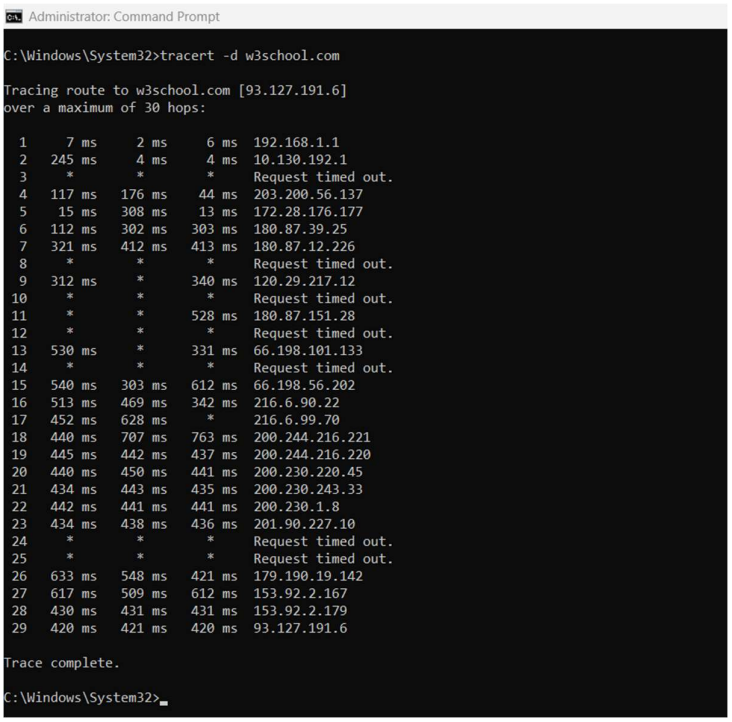
traceroute google.com [Same Like Windows]

Options:

Windows:

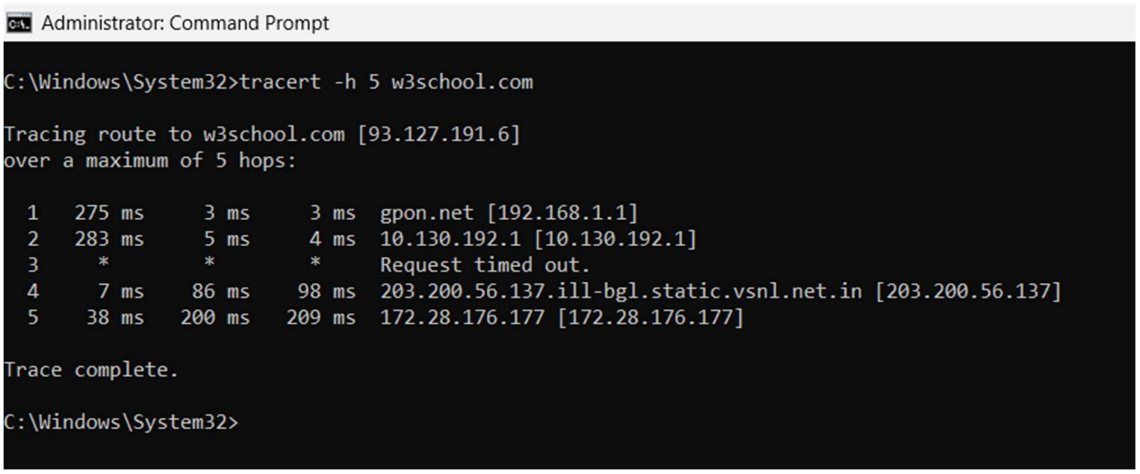
1. tracert -d google.com

- Normally tracert tries to convert each IP address into a hostname (via DNS lookup).
- With -d, it skips DNS resolution and only shows raw IP addresses.



tracert -h 5 google.com

- Limits the maximum number of hops (routers) it will trace to 5.
- Default is 30 hops in Windows.



Linux:

traceroute -n google.com

traceroute -m 10 google.com

traceroute -I google.com

3. netstat Command

Displays network connections, routing tables, interface statistics, and listening ports.

Main Uses:

- See active TCP/UDP connections.
- Find which ports are open and listening.
- Detect suspicious or unknown connections.

Common TCP Connection States

1. LISTENING

- Port is open and waiting for a connection.

- Example: A web server waiting on port 80.
  👉 Like a shopkeeper waiting for customers.

2. ESTABLISHED

- Connection is active, data is flowing.
- Example: Your browser connected to Google on port 443.
  👉 Like customer and shopkeeper talking & exchanging items.

3. SYN_SENT

- Your computer has sent a request to connect (SYN packet) but is waiting for reply.
- Example: You dial a phone number but the other person hasn't picked up yet.
  👉 Like a knock on the door, waiting for it to open.

4. SYN_RECEIVED

- The server received your SYN request and replied with SYN+ACK, now waiting for your ACK.
- Example: The other person picked up the call and said "Hello?", waiting for you to reply.
  👉 Like door opened, waiting for you to enter.

5. FIN_WAIT_1

- Your side started closing the connection (sent a FIN packet).
- Waiting for acknowledgment.
  👉 Like saying "Goodbye" but waiting to hear back.

6. FIN_WAIT_2

- Your FIN was acknowledged, now waiting for the other side to also send its FIN.
  👉 Like you hung up, but still holding the phone waiting for them to hang up too.

7. CLOSE_WAIT

- The other side sent a FIN (wants to close).

- Your side hasn't closed yet — maybe the application is still running.
  👉 Like the other person said goodbye, but you haven't replied yet.

8. TIME_WAIT

- Connection is closed, but socket stays reserved for some time.
- Ensures late/delayed packets don't confuse future connections.
  👉 Like keeping the door unlocked for a few seconds after closing it.

9. CLOSING

- Both sides sent FIN almost at the same time.
- Rare state.
  👉 Like both people saying goodbye at the same time.

10. CLOSED

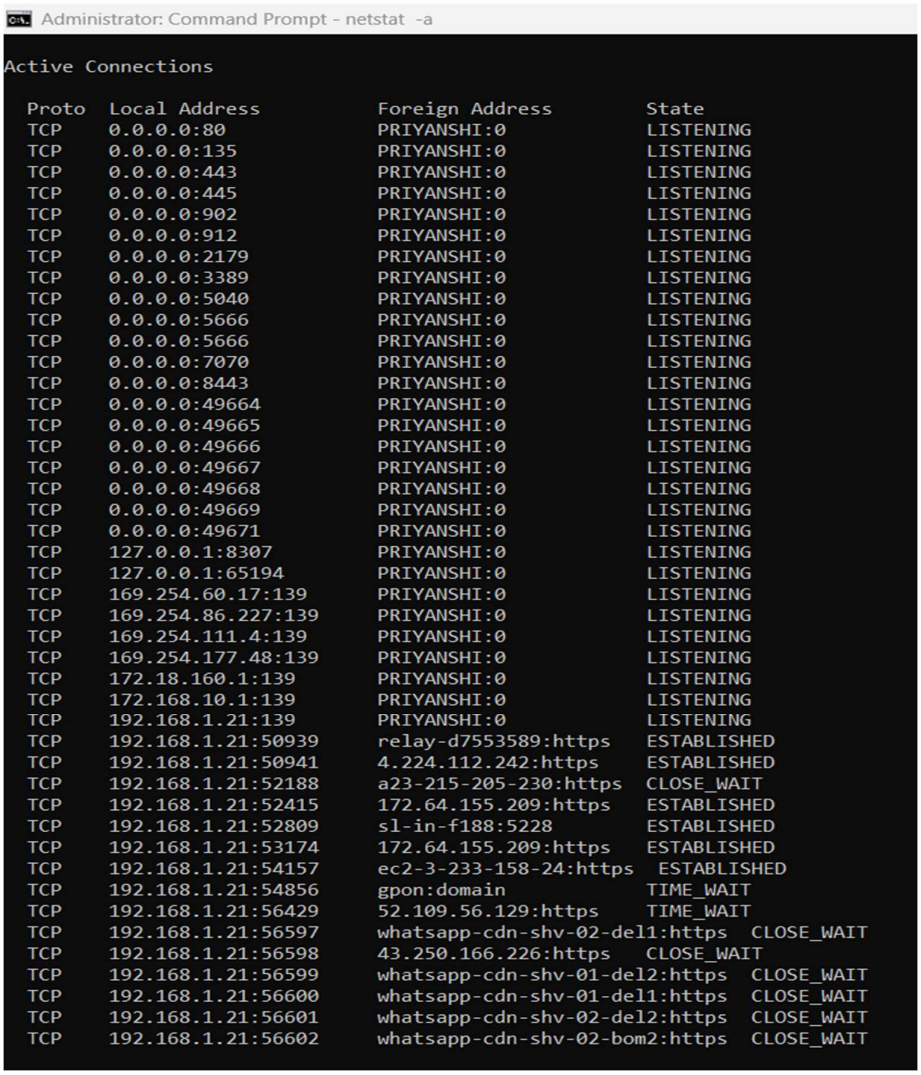
- No connection exists.
- Final state.
  👉 Like door locked and no one inside.

Common Commands:

Windows:

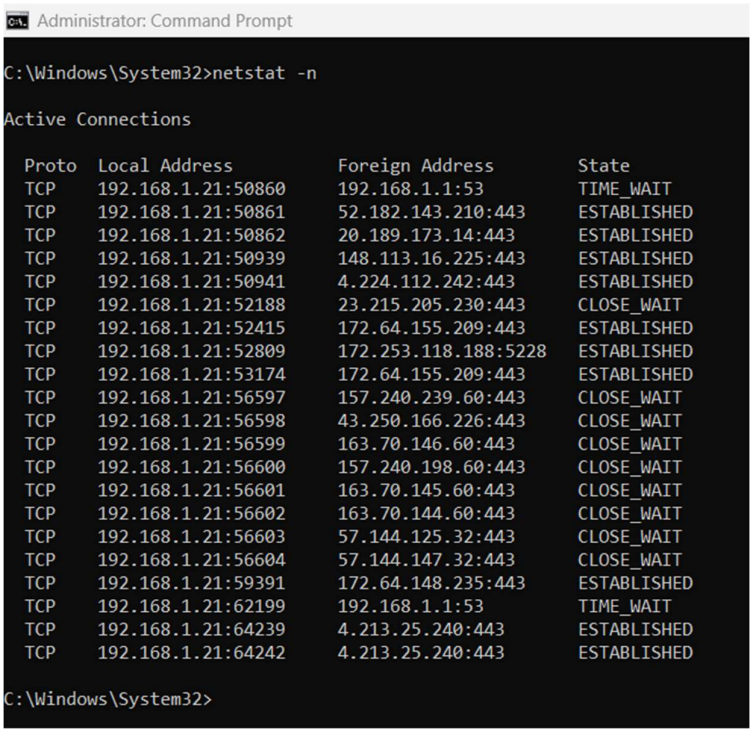
netstat -a

- Displays all active connections (TCP & UDP) and the ports your system is currently listening on.



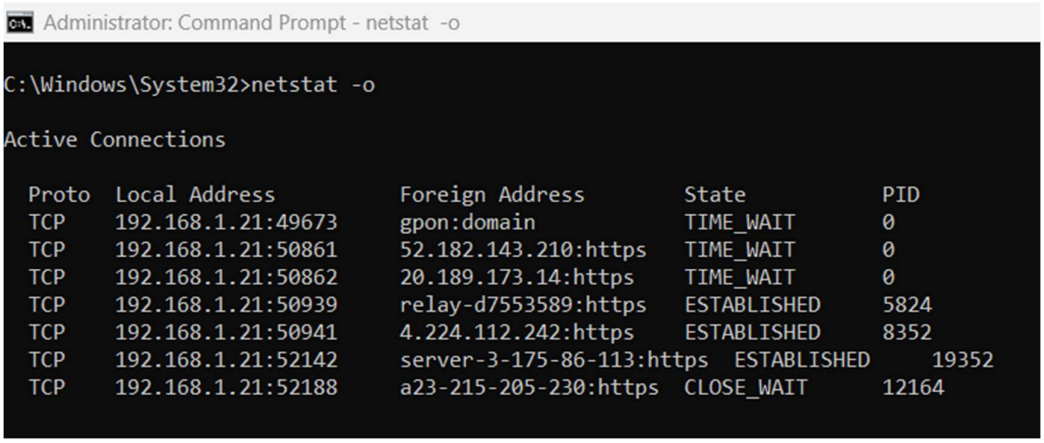
netstat -n

- By default, netstat tries to resolve hostnames and services (e.g., shows http instead of :80).
- With -n, it skips that and shows raw IP addresses and port numbers.



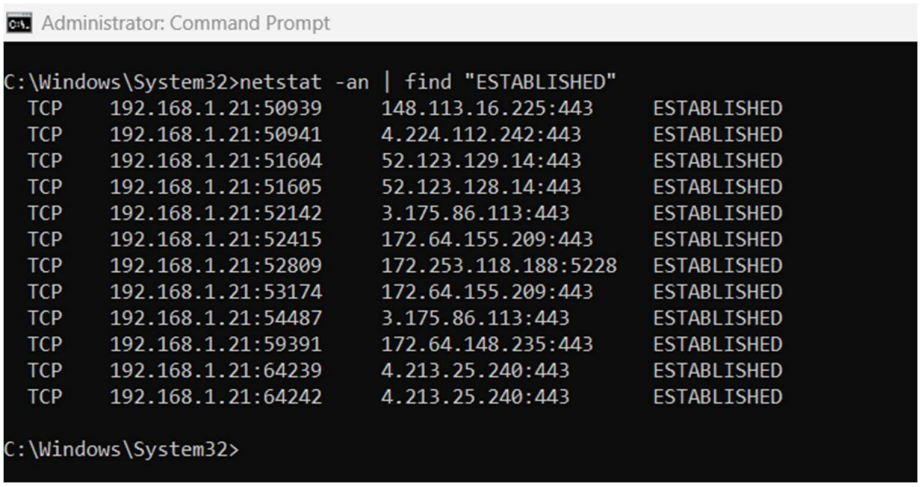
netstat -o

- Adds the process ID (PID) that is using each connection.



netstat -an | find "ESTABLISHED"

- netstat -an = all connections, numeric form.
- | find "ESTABLISHED" = only show results containing the word "ESTABLISHED".



Linux:

netstat -tuln

netstat -tulnp

netstat -s

netstat -r

4. ipconfig (Windows) / ifconfig (Linux)

Displays IP address, subnet mask, gateway, and other network info for your device.

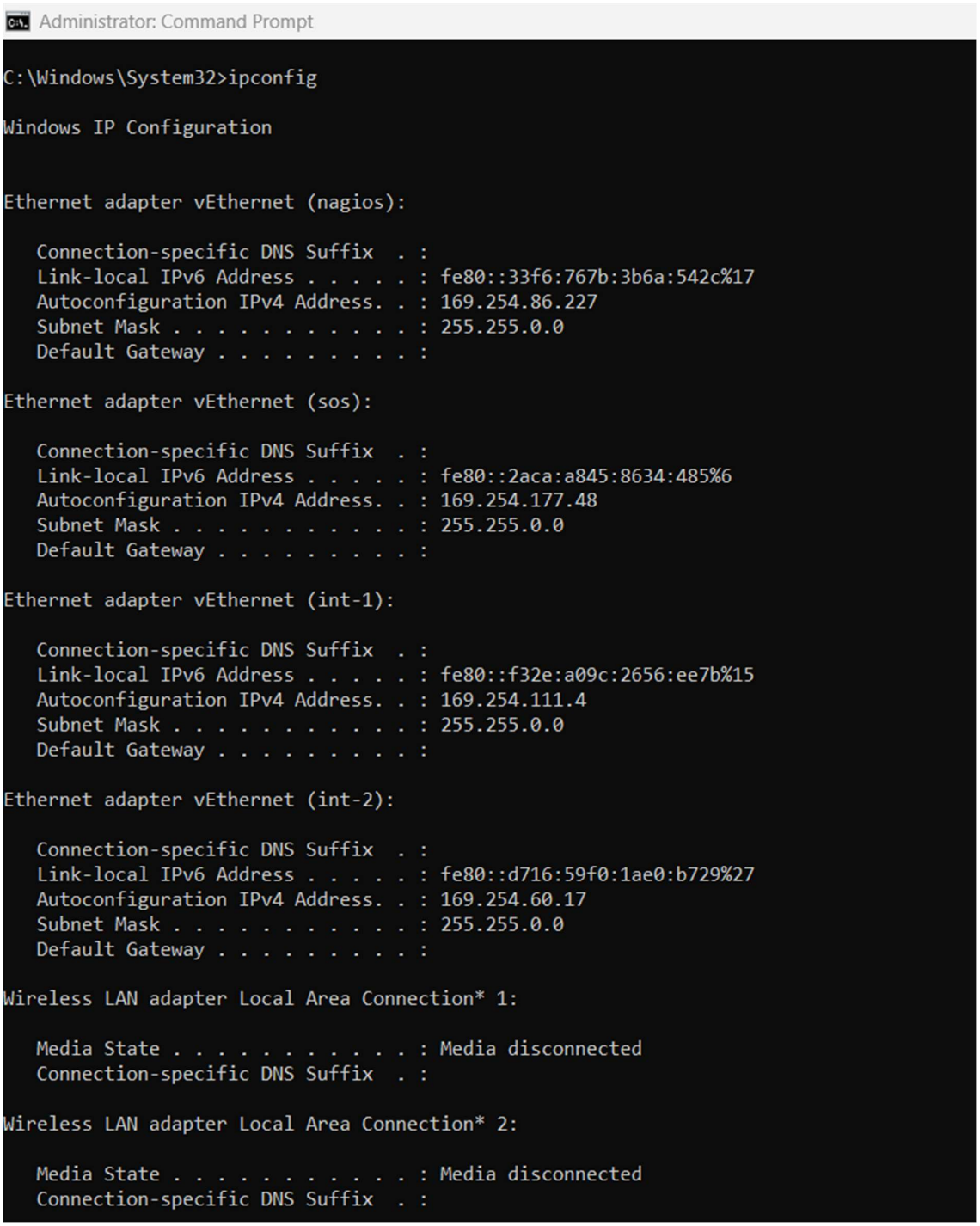
Main Uses:

- Find device IP and MAC address.
- Check DNS and DHCP info.
- Troubleshoot network connectivity.

Commands:

Windows:

ipconfig

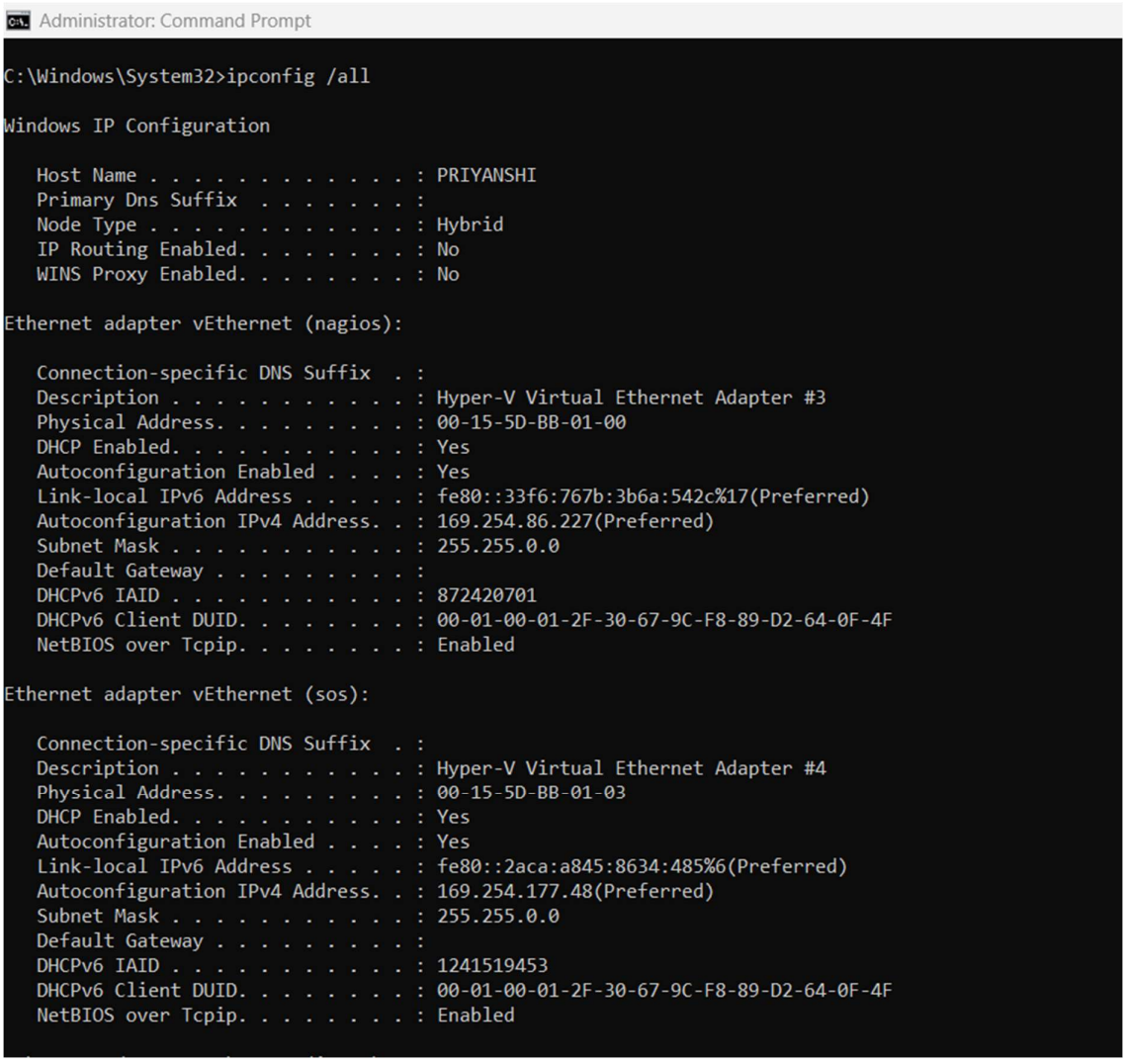


ipconfig /all

Shows detailed info including:

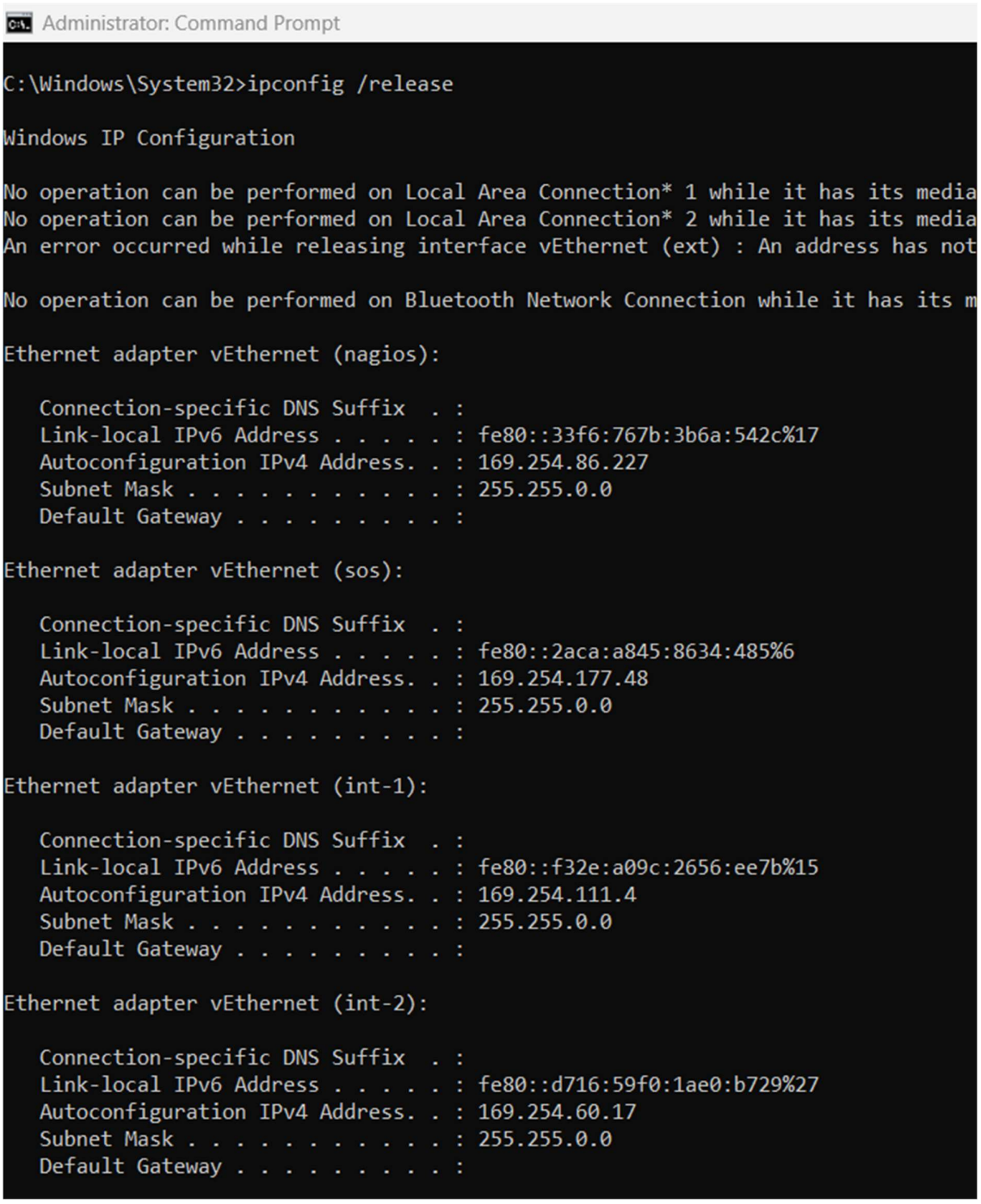
- IP address
- Subnet mask

- Default gateway
- DNS servers
- MAC (Physical) address
- DHCP info



ipconfig /release

- Releases current IP address obtained from DHCP server.

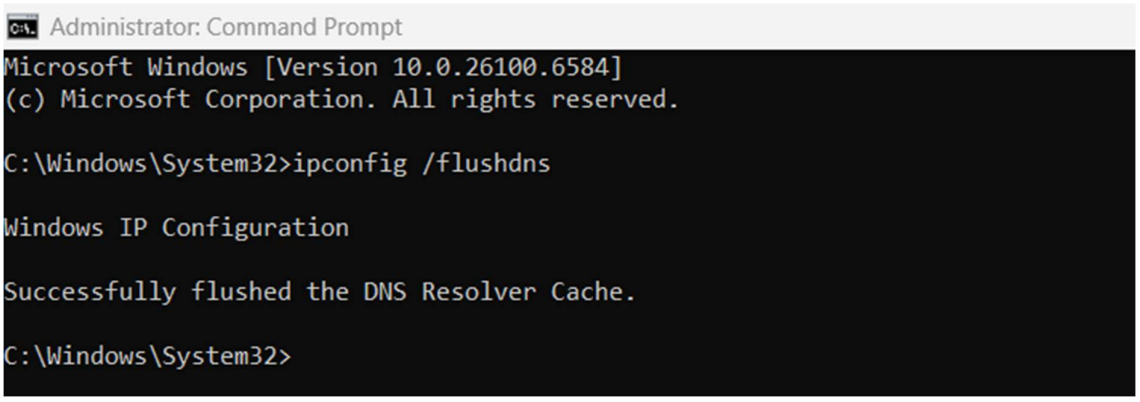


ipconfig /renew

- Requests a new IP address from the DHCP server.
- Used when internet is not working due to wrong IP.

ipconfig /flushdns

- Clears (flushes) the DNS cache.



Linux:

ifconfig

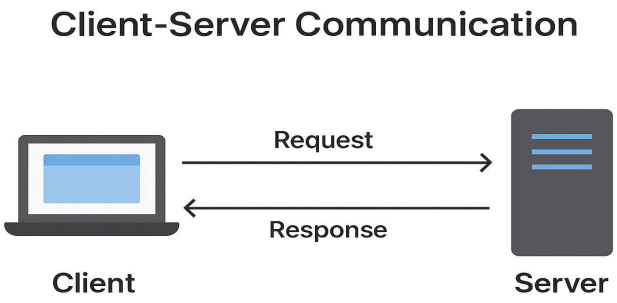
ifconfig eth0 up/down

ip addr show

ip route show [Routing Table]

❖ Task 4: Write a short report explaining client-server communication

- Client-server communication is when a computer (client) asks another computer (server) for information or services, and the server sends back the answer.
- It helps computers share data and resources over a network easily.



How Client-Server Communication Works

1. **Client Sends a Request:**
    - The client (computer, smartphone, or browser) starts the communication by asking the server for a service or data.
    - This could be a web page, a file, an email, or any other online service.
    - The client is basically saying, *"Please give me this information or perform this task."*

2. **Server Receives & Processes the Request:**
    - The server gets the request and checks what is being asked.
    - It retrieves the needed data, runs programs, or performs tasks to prepare the response.
    - The server makes sure the request is handled correctly before sending it back.

3. **Server Sends Back the Response:**

- After processing, the server sends the requested data, information, or result back to the client.
- This can be a web page, an email, a file, or confirmation of a service.
- If something goes wrong, the server may also send an error message to the client.

4. **Client Uses the Response:**
   - The client receives the response and displays or uses it.
   - Examples: a browser shows a web page, an email app shows new emails, or a game updates the screen.
   - The client may send more requests based on what it received, continuing the communication process.

Example:

Online Shopping Website:

- When you visit an online store like Amazon, your web browser acts as the **client**.
- It sends requests to the store's **server** to show product pages, check prices, or add items to the cart.
- The server processes these requests, fetches the information from databases, and sends it back to your browser.
- This allows you to view products, place orders, and track deliveries seamlessly.