RAJALAKSHMI ENGINEERING COLLEGE

An Autonomous Institution
Affiliated to Anna University, Chennai,
Rajalakshmi Nagar, Thandalam – 602 105

DEPARTMENT OF ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING

CS19541 - COMPUTER NETWORKS

Laboratory Record Note Book

Name : SIVARANJANI K

Register No. : 2116221501139

Year / Branch / Section : BTECH AIML C

Semester : V

AcademicYear: 2024-2025

RAJALAKSHMI ENGINEERING COLLEGE

An Autonomous Institution
Affiliated to Anna University, Chennai,
Rajalakshmi Nagar, Thandalam – 602 105

BONAFIDE CERTIFICATE

Name: Sivaranjani K

Academic Year: 2024-25 **Semester:** V **Branch:** AIML C

Register No. 2116221501139

Certified that this is the bonafide record of work done by the above student in the CS19541 COMPUTER NETWORKS ***Laboratory during the academic year 2024- 2025***

Signature of Faculty in-charge

Submitted for the Practical Examination held on.....20/11/24.....

Internal Examiner **External Examiner**

INDEX

EX.NO: 1	BASIC NETWORKING COMMANDS

Aim:

Study of various Network commands used in Windows and Linux.

Commands:

- **Ipconfig:** This command displays detailed configuration information about your TCP/IP connection including Router, Gateway, DNS, DHCP, and type of Ethernet adapter in your system.

Output:

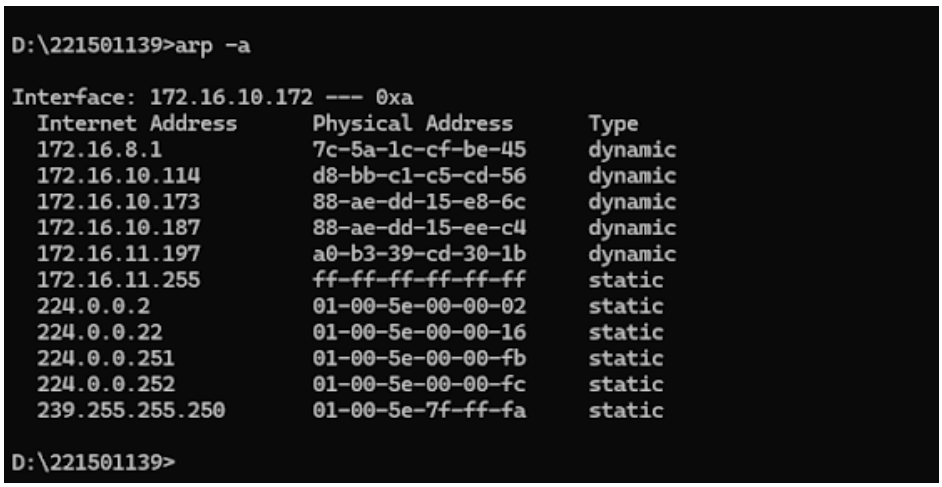
```
D:\>cd 221501139

D:\221501139>ipconfig

Windows IP Configuration


Ethernet adapter Ethernet:

   Connection-specific DNS Suffix  . :
   Link-local IPv6 Address . . . . . : fe80::1561:dd78:ed2c:4726%10
   IPv4 Address. . . . . . . . . . . : 172.16.10.172
   Subnet Mask . . . . . . . . . . . : 255.255.252.0
   Default Gateway . . . . . . . . . : 172.16.8.1

D:\221501139>
```

- **Hostname** : This is the simplest of all TCP/IP commands. It simply displays the name of your computer.

Output:

- **arp -a** : ARP is a short form of address resolution protocol.It will show the IP address of your computer along with the IP address and MAC address of your router.

Output:



- **Nbtstat -a** : This command helps solve problems with NetBIOS name resolution. (Nbt stands for NetBIOS over TCP/IP)

Output:

```
D:\221501139>nbtstat -a

Displays protocol statistics and current TCP/IP connections using NBT
(NetBIOS over TCP/IP).

NBTSTAT [ [-a RemoteName] [-A IP address] [-c] [-n]
        [-r] [-R] [-RR] [-s] [-S] [interval] ]

  -a   (adapter status) Lists the remote machine's name table given its name
  -A   (Adapter status) Lists the remote machine's name table given its
                        IP address.
  -c   (cache)          Lists NBT's cache of remote [machine] names and their IP addresses
  -n   (names)          Lists local NetBIOS names.
  -r   (resolved)       Lists names resolved by broadcast and via WINS
  -R   (Reload)         Purges and reloads the remote cache name table
  -S   (Sessions)       Lists sessions table with the destination IP addresses
  -s   (sessions)       Lists sessions table converting destination IP
                        addresses to computer NETBIOS names.
  -RR  (ReleaseRefresh) Sends Name Release packets to WINS and then, starts Refresh

  RemoteName   Remote host machine name.
  IP address   Dotted decimal representation of the IP address.
  interval     Redisplays selected statistics, pausing interval seconds
               between each display. Press Ctrl+C to stop redisplaying
               statistics.
```

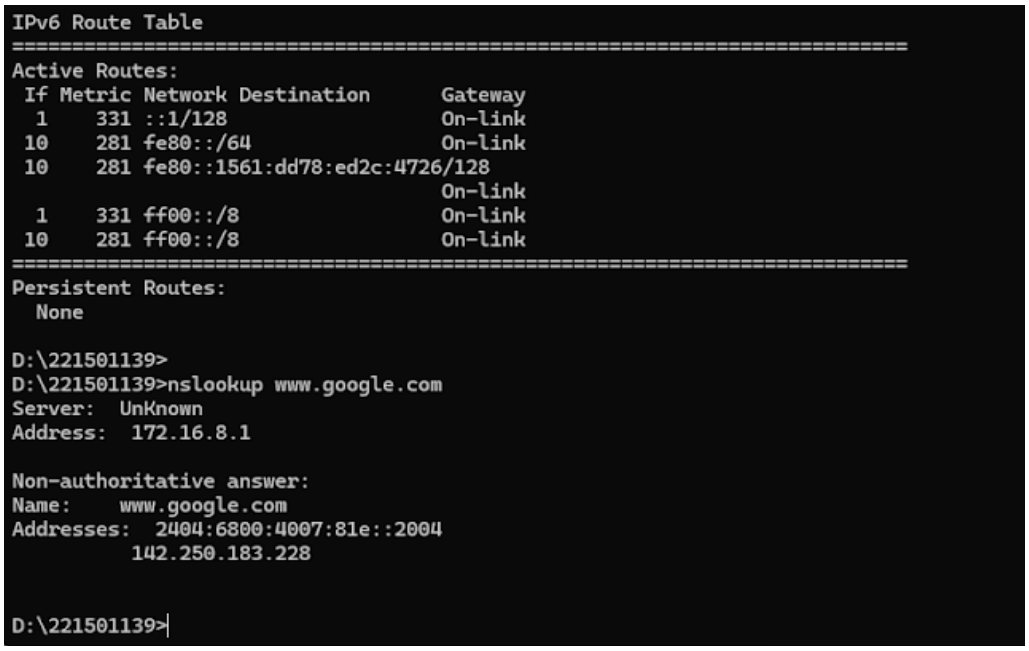
- **Netstat -r** : (network statistics) netstat displays a variety of statistics about a computers active TCP/IP connections. It is a command line tool for monitoring network connections both incoming and outgoing as well as viewing routing tables, interface statistics etc.

Output:

```
D:\221501139>netstat -r
===========================================================================
Interface List
 10...88 ae dd 14 72 47 ......Realtek PCIe GbE Family Controller
  1...........................Software Loopback Interface 1
===========================================================================

IPv4 Route Table
===========================================================================
Active Routes:
Network Destination        Netmask          Gateway       Interface  Metric
          0.0.0.0          0.0.0.0      172.16.8.1    172.16.10.172    281
        127.0.0.0        255.0.0.0         On-link         127.0.0.1    331
        127.0.0.1  255.255.255.255         On-link         127.0.0.1    331
  127.255.255.255  255.255.255.255         On-link         127.0.0.1    331
      169.254.0.0      255.255.0.0         On-link     172.16.10.172     30
  169.254.255.255  255.255.255.255         On-link     172.16.10.172    281
       172.16.8.0    255.255.252.0         On-link     172.16.10.172    281
    172.16.10.172  255.255.255.255         On-link     172.16.10.172    281
    172.16.11.255  255.255.255.255         On-link     172.16.10.172    281
        224.0.0.0        240.0.0.0         On-link         127.0.0.1    331
        224.0.0.0        240.0.0.0         On-link     172.16.10.172    281
  255.255.255.255  255.255.255.255         On-link         127.0.0.1    331
  255.255.255.255  255.255.255.255         On-link     172.16.10.172    281
===========================================================================
Persistent Routes:
  Network Address          Netmask  Gateway Address  Metric
          0.0.0.0          0.0.0.0      172.16.8.1  Default
===========================================================================
```

- **Nslookup** : It is a tool used to perform DNS lookups in Linux. It is used to display DNS details, such as the IP address of a particular computer, the MX records for a domain or the NS servers of a domain. nslookup can operate in two modes: interactive and non-interactive.

  Output:



- **Pathping** : Pathping is unique to Window's, and is basically a combination of the Ping and Tracert commands. Pathping traces the route to the destination address then launches a 25 second test of each router along the way, gathering statistics on the rate of data loss along each hop.

Output:

```
D:\221501139>pathping

Usage: pathping [-g host-list] [-h maximum_hops] [-i address] [-n]
                [-p period] [-q num_queries] [-w timeout]
                [-4] [-6] target_name

Options:
    -g host-list     Loose source route along host-list.
    -h maximum_hops  Maximum number of hops to search for target.
    -i address       Use the specified source address.
    -n               Do not resolve addresses to hostnames.
    -p period        Wait period milliseconds between pings.
    -q num_queries   Number of queries per hop.
    -w timeout       Wait timeout milliseconds for each reply.
    -4               Force using IPv4.
    -6               Force using IPv6.

D:\221501139>
```

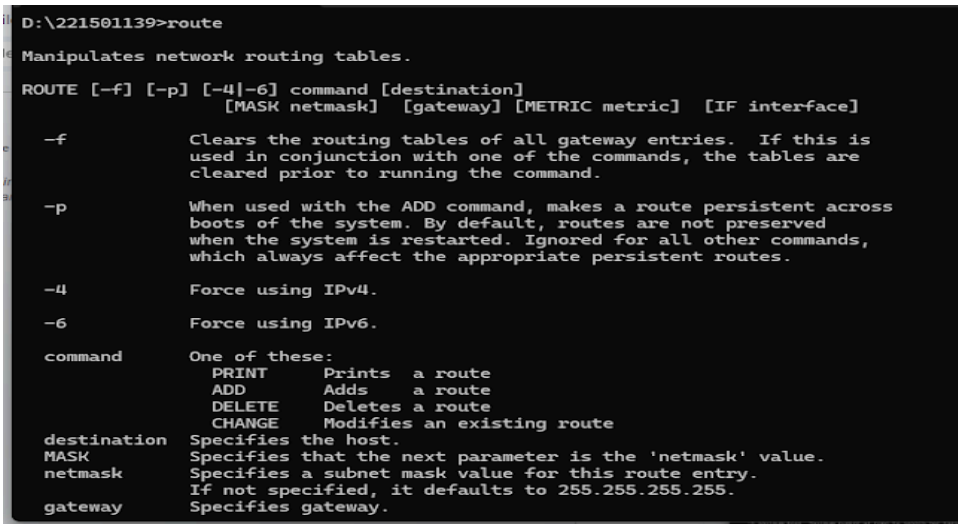
- **Ping**: (Packet Internet Groper) command is the best way to test connectivity between two nodes. Ping uses ICMP (Internet Control Message Protocol) to communicate to other devices.

Output:

```
D:\221501139>ping www.facebook.com

Pinging star-mini.c10r.facebook.com [157.240.192.35] with 32 bytes of data:
Reply from 157.240.192.35: bytes=32 time=3ms TTL=59
Reply from 157.240.192.35: bytes=32 time=3ms TTL=59
Reply from 157.240.192.35: bytes=32 time=10ms TTL=59
Reply from 157.240.192.35: bytes=32 time=3ms TTL=59

Ping statistics for 157.240.192.35:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 3ms, Maximum = 10ms, Average = 4ms

D:\221501139>
```

- **Route** : route command is used to show/manipulate the IP routing table. It is primarily used to setup static routes to specific host or networks via an interface.

Output:



Result:

Thus the study of various Network commands used in Windows and Linux networks are written and output is verified.

EX.NO: 2	STUDY OF CABLES USED IN NETWORKING

Aim:

To study about the different types of cables used in networking.

Network Cabling Definition:

A physical medium used to connect devices, such as computers, servers, and switches, to form a computer network. Networking cables transmit data, voice, and video signals between devices, enabling communication, data transfer, and internet connectivity.

Functions of Networking Cable:

1. Device Connection: Connect devices to a network.
2. Data Transmission: Transmit data, voice, and video signals.
3. Internet Connectivity: Provide internet access.
4. Network Expansion: Enable network expansion and extension.

Types of Network Cabling:

- ❖ Shielded Twisted Pair (STP) Cable
- ❖ Unshielded Twisted Pair (UTP) Cable
- ❖ Fiber Optic Cable
- ❖ Coaxial Cable

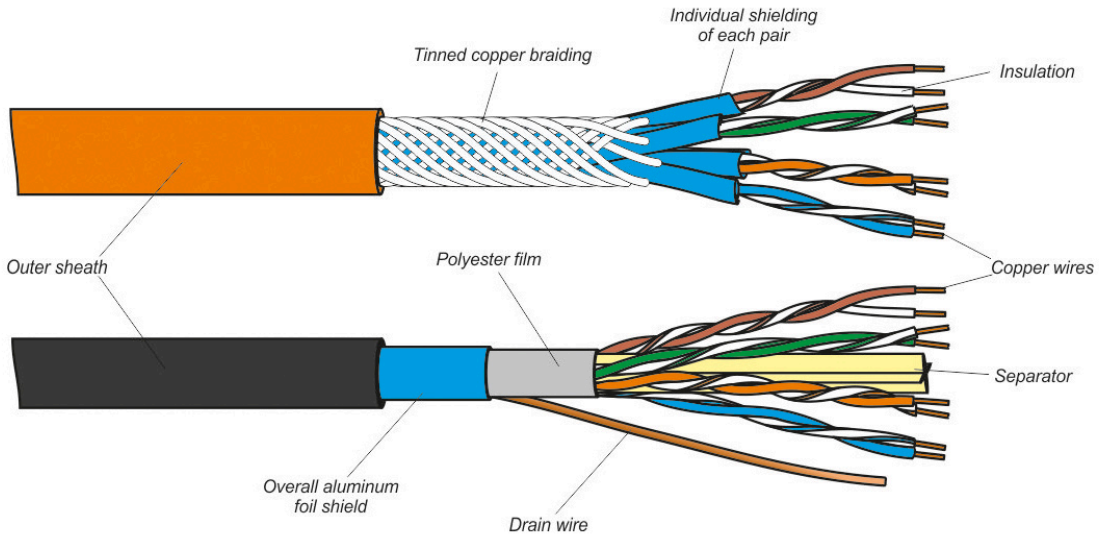
TWISTED PAIR CABLES:

Twisted-pair cabling is a cable made up of one or more pairs of copper wires. The individually insulated conductors are twisted together. Twisting also reduces electromagnetic radiation, and helps the cables resist external interference. Twisted-pair cable categories

1. Unshielded twisted pair (UTP)
2. Shielded twisted pair (STP)

1. UTP:

UTP stands for Unshielded Twisted Pair cable. UTP cable is a 100 ohm copper cable that consists of 2 to 1800 unshielded twisted pairs surrounded by an outer jacket. They have no metallic shield. This makes the cable small in diameter but unprotected against electrical interference. The twist helps to improve its immunity to electrical noise and EMI.The copper conductor of both horizontal and backbone UTP cables are either 22 AWG or 24 AWG. 24 AWG is the most common size.



Applications:

UTP cables are mostly used for LAN networks. They can be used for voice, low-speed data, high-speed data, audio and paging systems, and building automation and control systems. UTP cable can be used in both the horizontal and backbone cabling subsystems.

Advantages:

1. Cost-effective
2. Lightweight and flexible.
3. Easy installation
4. Wide availability

Disadvantages:

1. Susceptible to interference
2. Limited bandwidth
3. Security risks.
4. Crosstalk.
5. Limited length

Types of UTP cables:

1. Cat3: Older UTP cable standard for telephone and Ethernet applications
2. Cat5: UTP cable standard for Ethernet applications (up to 100 Mbps)
3. Cat5e: Enhanced UTP cable standard for Ethernet applications (up to 1 Gbps)
4. Cat6: UTP cable standard for Ethernet applications (up to 10 Gbps)

Standard, Straight-Through Wiring Diagram(both ends are the same):

RJ45 Pin #	Wire Color (T568A)	Wire Diagram (T568A)	10Base-T Signal 100Base-TX Signal	1000Base-T Signal
1	White/Green		Transmit+	BI_DA+
2	Green		Transmit-	BI_DA-
3	White/Orange		Receive+	BI_DB+
4	Blue		Unused	BI_DC+
5	White/Blue		Unused	BI_DC-
6	Orange		Receive-	BI_DB-
7	White/Brown		Unused	BI_DD+
8	Brown		Unused	BI_DD-

Straight-Through Ethernet Cable Pin Out for T568A

2. STP (Shielded Twisted Pair) cable:

STP cable is a type of twisted pair cable that has a shielded layer to protect the inner twisted pairs from electromagnetic interference (EMI).

Applications:

1. Ethernet networks (e.g., Cat5e, Cat6, Cat7)
2. Industrial control systems
3. Audio/video installations
4. Telephone systems
5. Data centers

Advantages & Disadvantages:

1. Improved noise immunity.
2. Higher data transfer rates.
3. Reduced crosstalk
4. Higher cost
5. More difficult to install

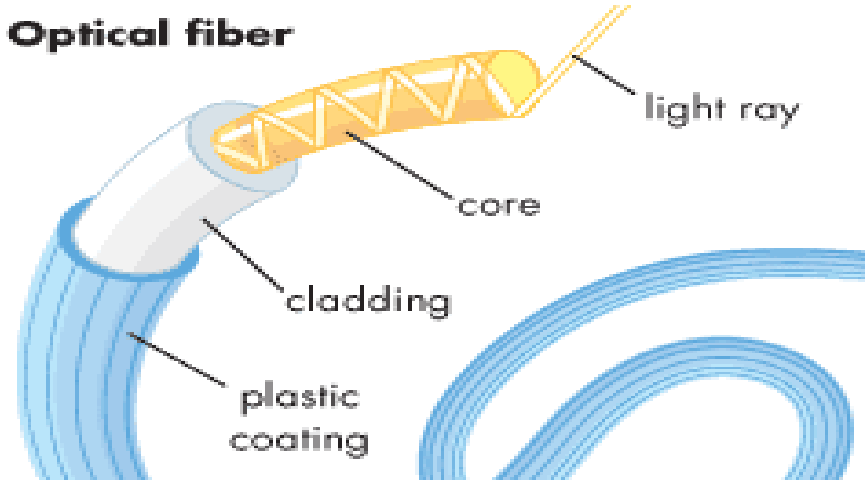
Types of STP cables:

1. Screened Twisted Pair (ScTP): A single shielded layer surrounds all twisted pairs.
2. Shielded Twisted Pair (STP): Each twisted pair has its own shielded layer.
3. Foiled Twisted Pair (FTP): A foil shield surrounds each twisted pair.

3. FIBER OPTIC CABLE:

A fiber optic cable is a thin strand of glass or plastic that uses light to transmit data as signals. It consists of three main components:

1. Core: The central part of the fiber that carries the light signal.
2. Cladding: The layer surrounding the core that helps keep the light signal within the core.
3. Coating: The outer layer that protects the fiber from damage and moisture.

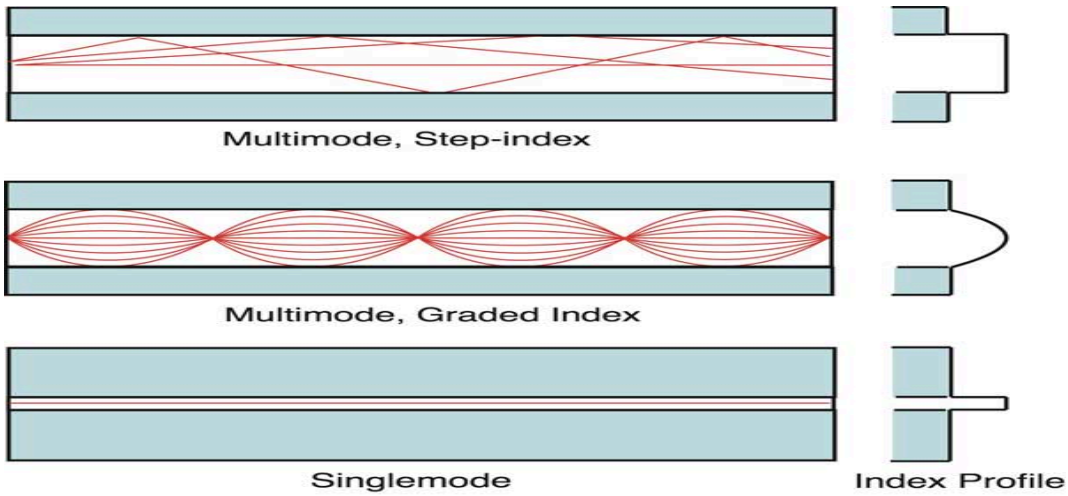


How it Works:

1. Light Transmission 2. Total Internal Reflection 3. Signal Reception

Types of Fiber Optic Cables:

1. Single-Mode Fiber (SMF): Used for long-distance, high-speed applications.
2. Multimode Fiber (MMF): Used for shorter-distance, high-speed applications.
3. Plastic Optical Fiber (POF): Used for short-distance, low-speed applications.

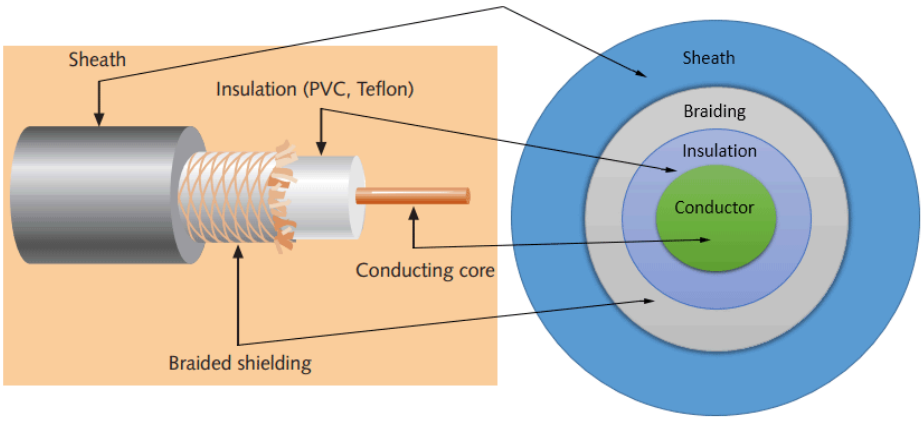


Advantages & Disadvantages:

1. High Bandwidth
2. Long-Distance Transmission
3. High Security.
4. High Cost
5. Difficult Installation
6. Fragility

4. Coaxial Cable:

This type of cable is commonly used in older buildings. It uses a copper core surrounded by an insulating material, then a metal sheath. The metal sheath gives it strength while keeping out outside elements such as water or dirt. This type of cable is not recommended for use in commercial buildings because it can only transmit signals over short distances.



Result:

Thus the different types of network cables are studied.

EX.NO: 3 a	CISCO PACKET TRACER

a) To understand the environment of CISCO PACKET TRACER to design a simple network.

Aim:

To design a simple network using Cisco Packet Tracer.

Cisco Packet Tracer is a network simulation tool that allows users to create network topologies and simulate various networking devices and protocols. It's widely used in education, especially for learning and practicing networking concepts, and for designing simple to complex networks**.**

UI OVERVIEW

1. Menu Bar: Located at the top, providing access to file management, simulation modes, and options.

2. Toolbar: Below the menu, offering quick access to essential tools like device selection, simulation controls, and zoom.

3. Device Selection Panel: On the left, allows you to drag and drop devices (routers, switches, PCs) and cables into the workspace.

4. Workspace: The central area where you design and visualize the network by placing and connecting devices.

5. Device Configuration Panel: Appears when a device is clicked, offering configuration options (e.g., IP settings, CLI commands).

6. Simulation Mode/Real-Time Mode Toggle: Switch between observing packet flow (Simulation Mode) or real-time device behavior (Real-Time Mode).

7. Bottom Panel: Displays simulation controls and packet information in Simulation Mode.

8. Status Bar: At the bottom, showing the current mode and device status.

Steps to Design a Simple Network:

1. Place Devices:

- Drag two PCs, one switch, and one router onto the workspace.

2. Connect Devices:

- Use copper straight-through cables to connect the PCs to the switch and the switch to the router.

3. Configure IP Addresses:

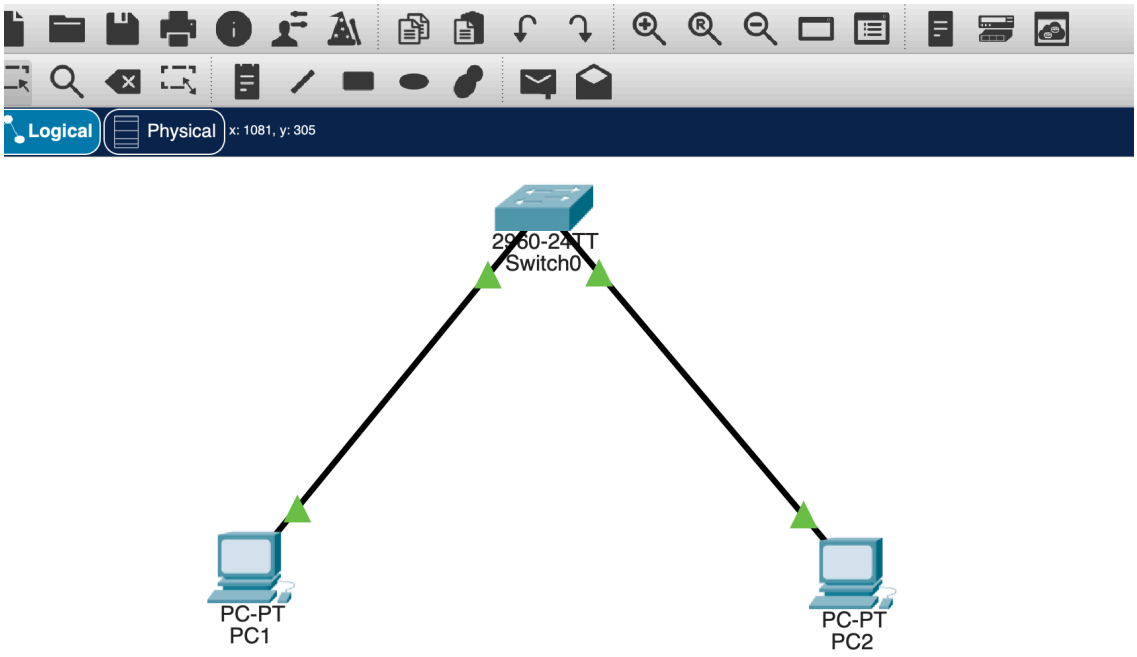
- Assign IP addresses to the PCs (e.g., PC1: 192.168.1.1, PC2: 192.168.1.2).
- Configure the router's interface with an IP address for the network (e.g., 192.168.1.254).

4. Test with Ping:

- Ping from PC1 to PC2 to check connectivity.

5. Simulation:

- Go to simulation mode and observe how the ping request travels through the network.



Result:

Thus a basic network setup created and simulated using Cisco Packet Tracer.

EX.NO: 3 b	CISCO PACKET TRACER

b) Analyse the behaviour of network devices using CISCO PACKET TRACER simulator.

1. Network Using Hub (4 to 6 Hosts):

- **Design:**

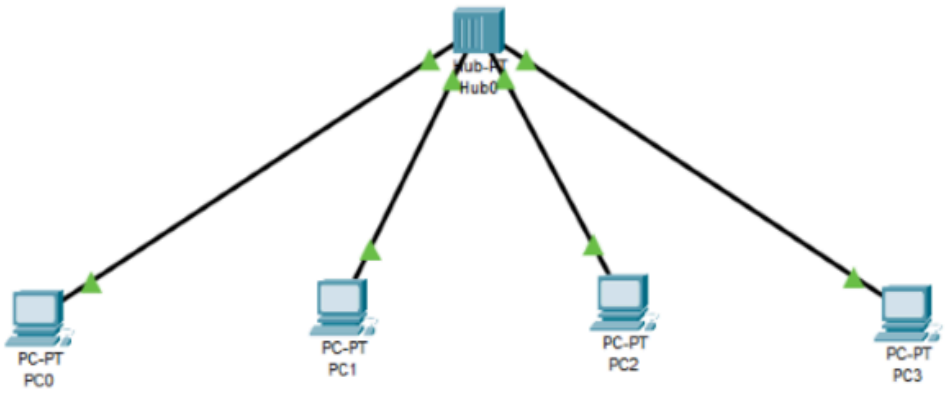
  Add 4 to 6 PCs and a hub from the device library.

  Connect each PC to the hub using copper straight-through cables.

- **Simulation:**

  In Simulation Mode, send a ping from one PC to another and observe how all devices receive the broadcast packet.

  Since a hub broadcasts data to all devices on the network, all PCs will receive the data even if it's not intended for them.



2. Network Using Switch (4 to 6 Hosts):

- **Design:**

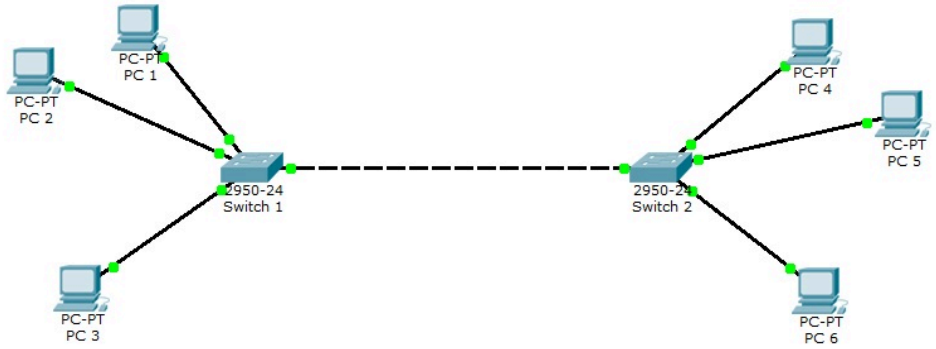
  Add 4 to 6 PCs and a switch from the device library.

  Connect each PC to the switch using copper straight-through cables.

- **Simulation:**

  In Simulation Mode, ping between PCs.

  The switch will intelligently forward packets only to the device with the matching MAC address, ensuring efficient communication.



Result:

Thus a basic network setup using hub and switch are created and simulated using Cisco Packet Tracer.

EX.NO: 4	CISCO PACKET TRACER

Aim:

To Setup and configure a LAN (Local area network) using Switch and Ethernet cables.

Procedure:

1. Connect Host Machines to Switch:

- Use Ethernet cables to connect each of the 3-4 host machines to the switch.

2. Assign IP Addresses:

- Assign static IP addresses to each host machine (e.g., 192.168.1.2, 192.168.1.3, 192.168.1.4, etc.) through their network configuration settings.

3. Test Connectivity Using Ping Command:

- Open a command prompt on each host machine.
- Use the `ping` command to test connectivity between the machines (e.g., `ping 192.168.1.3` from host 1 to host 2).
- If the network is correctly set up, the ping should return successful replies.

4. Share Files and Folders:

- On each host, configure file sharing in the network settings.
- Set permissions and share folders or files.
- Access the shared files from the other host machines by browsing the network in File Explorer.

EX.NO: 5	WIRESHARK

Aim:

To understand the features of wireshark as a packet capture tool and understand encapsulation of information at various layers of a Protocol stack.

Wireshark is a network protocol analyzer used to capture and inspect data packets traveling across a network. It helps in analyzing network traffic and diagnosing issues at different protocol layers.

Basic Features of Wireshark:

1. Packet Capture: Wireshark can capture live network traffic from wired or wireless interfaces and display detailed packet-level information.

2. Packet Panel: The captured packets are displayed in a panel with three main sections:

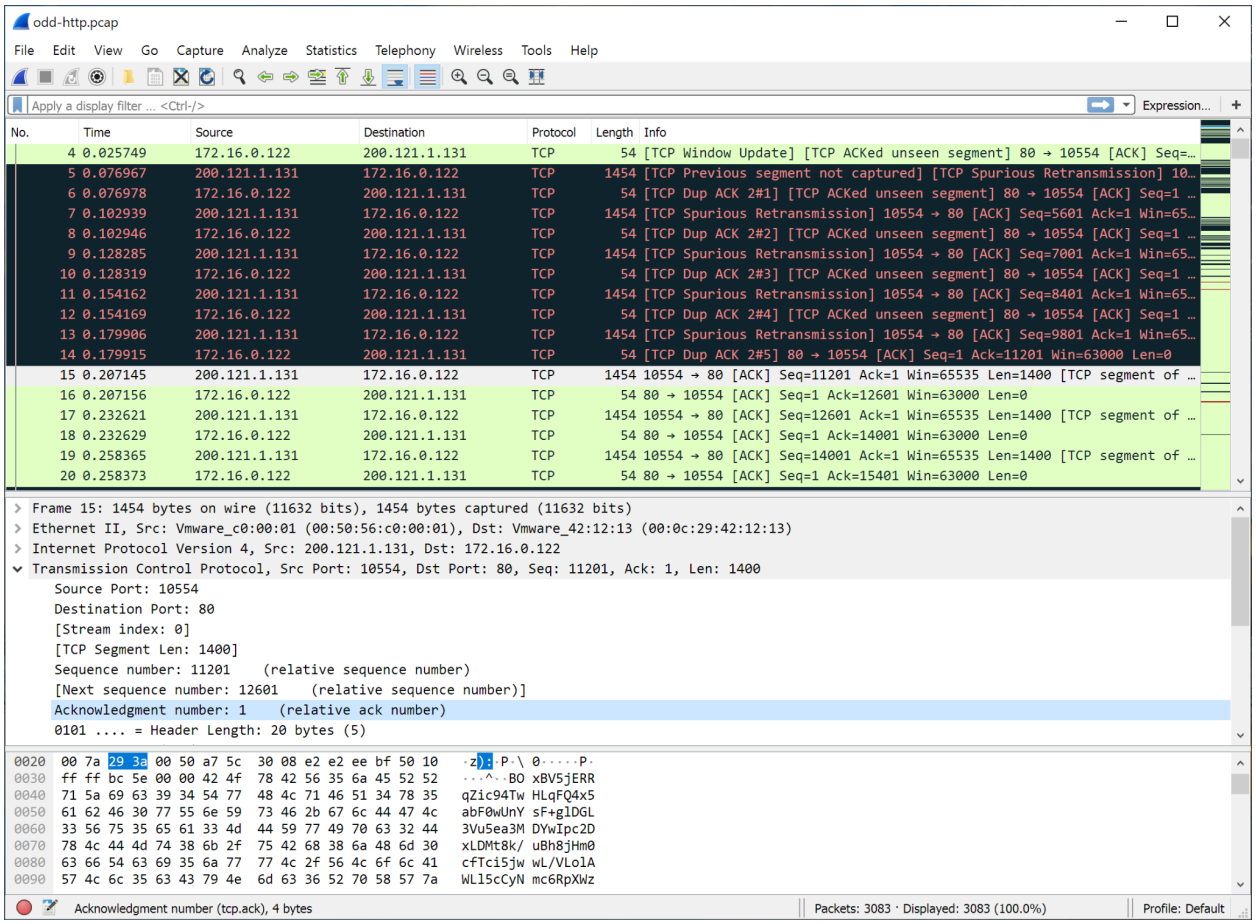
- Packet List: A summary of captured packets with time, source, destination, protocol, and packet length.
- Packet Details: A hierarchical breakdown of the selected packet, showing headers and protocol information for each layer (Ethernet, IP, TCP/UDP, etc.).
- Packet Bytes: Raw byte data of the selected packet, shown in hexadecimal and ASCII.

3. Protocol Decoding: Wireshark decodes packets and presents information in a readable format, allowing you to analyze data exchanges from the Physical Layer to the Application Layer.

4. Filters: Display filters allow you to search and focus on specific packets or protocols, like filtering by IP address, port number, or protocol type.

5. Reassembly: Wireshark can reassemble fragmented packets and TCP streams for complete analysis of data transfers.

6. Statistics: Provides insights like protocol distribution, flow graphs, and network performance analysis.



Result:

Thus the output is verified.

EX.NO: 6	ERROR CORRECTION AT DATA LINK LAYER

Aim:

To write a program to implement error detection and correction using the hamming code concept.

Program:

```
def calculate_parity_bits(data_bits):
    m = len(data_bits)
    r = 1
    while (2**r) < (m + r + 1):
        r += 1
    return r
def generate_hamming_code(data_bits):
    m = len(data_bits)
    r = calculate_parity_bits(data_bits)
    total_length = m + r
    hamming_code = ['0'] * total_length
    j = 0
    for i in range(1, total_length + 1):
        if not (i & (i - 1)) == 0:  # Not a power of 2
            hamming_code[i - 1] = data_bits[j]
            j += 1
    for i in range(r):
        parity_pos = 2**i
        parity = 0
        for j in range(parity_pos - 1, total_length, 2 * parity_pos):
            parity ^= sum(int(hamming_code[k]) for k in range(j, min(j + parity_pos,
total_length)))
```

```
        hamming_code[parity_pos - 1] = str(parity)
    return ''.join(hamming_code)

def detect_and_correct(hamming_code):
    total_length = len(hamming_code)
    r = calculate_parity_bits(hamming_code)
    error_position = 0
    for i in range(r):
        parity_pos = 2**i
        parity = 0
        for j in range(parity_pos - 1, total_length, 2 * parity_pos):
            parity ^= sum(int(hamming_code[k]) for k in range(j, min(j + parity_pos,
total_length)))
        if parity:
            error_position += parity_pos
    if error_position:
        print(f"Error detected at position: {error_position}")
        corrected_code = list(hamming_code)
        corrected_code[error_position - 1] = '1' if corrected_code[error_position - 1]
== '0' else '0'
        return ''.join(corrected_code)
    else:
        print("No errors detected.")
        return hamming_code
data = "1011"
hamming_code = generate_hamming_code(data)
print(f"Hamming Code: {hamming_code}")
received_code = hamming_code[:3] + ('1' if hamming_code[3] == '0' else '0') +
hamming_code[4:]  # Introduce error
print(f"Received Code with Error: {received_code}")
corrected_code = detect_and_correct(received_code)
print(f"Corrected Hamming Code: {corrected_code}")
```

Output:

Hamming Code: 0110011
Received Code with Error: 0111011
Error detected at position: 4
Corrected Hamming Code: 0110011

Result:

Thus the program is written and implemented for error detection and correction using the hamming code.

EX.NO: 7	FLOW CONTROL AT DATA LINK LAYER

Aim:

To write a program to implement flow control at the data link layer using SLIDING WINDOW PROTOCOL.

Program:

```
def sliding_window_protocol(frames, window_size):
    sent = 0
    ack = 0
    while sent < len(frames):

        for i in range(window_size):
            if sent + i < len(frames):
                print(f"Sent frame {frames[sent + i]}")

        ack = min(len(frames), sent + window_size)
        print(f"Acknowledged up to frame {frames[ack - 1]}")
        sent = ack

frames = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
window_size = 4
sliding_window_protocol(frames, window_size)
```

Output:

```
Sent frame 1
Sent frame 2
Sent frame 3
Sent frame 4
Acknowledged up to frame 4
Sent frame 5
Sent frame 6
Sent frame 7
Sent frame 8
Acknowledged up to frame 8
Sent frame 9
Sent frame 10
Acknowledged up to frame 10
```

Result:

Thus the program is written to implement flow control at the data link layer using SLIDING WINDOW PROTOCOL.

EX.NO: 8 a	VIRTUAL LAN

Aim:
To simulate Virtual LAN configuration using CISCO Packet Tracer Simulation.

Procedure:

1. Set Up Topology:

- Add 1 switch and 4 PCs to the workspace.
- Connect PCs to the switch using straight-through cables.

2. Assign IP Addresses:

- PC1, PC2 (VLAN 10): Assign `192.168.10.1` and `192.168.10.2`.
- PC3, PC4 (VLAN 20): Assign `192.168.20.1` and `192.168.20.2`.

3. Configure VLANs on Switch:

- Open the **CLI** of the switch and enter:

```plaintext
enable
configure terminal
vlan 10
name Sales
vlan 20
name Marketing
exit
```

- Assign ports to VLANs:

```plaintext
interface fastethernet 0/1
switchport mode access
switchport access vlan 10
```

```
exit
interface fastethernet 0/3
switchport mode access
switchport access vlan 20
exit
```

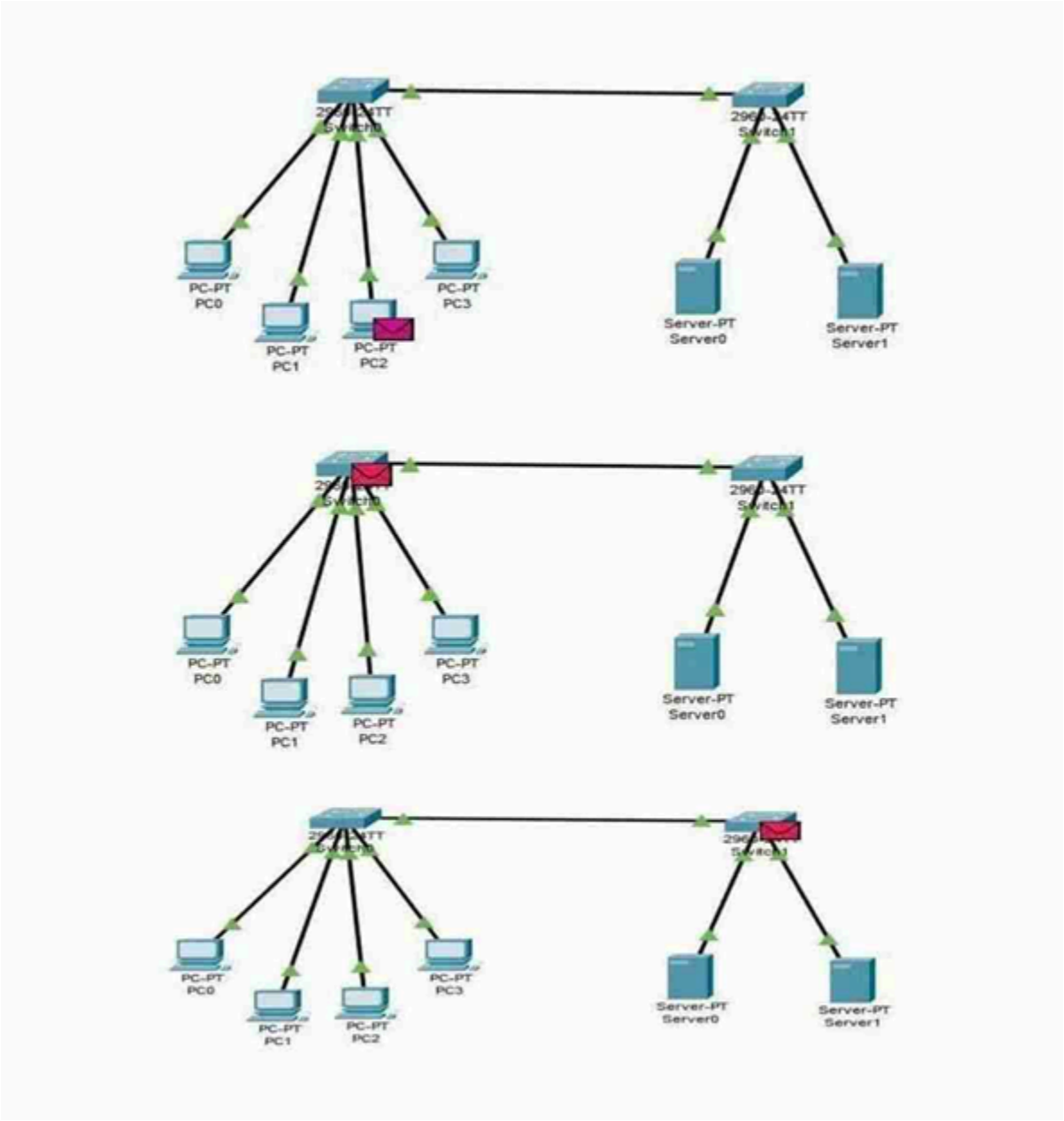
4. Verify Configuration:

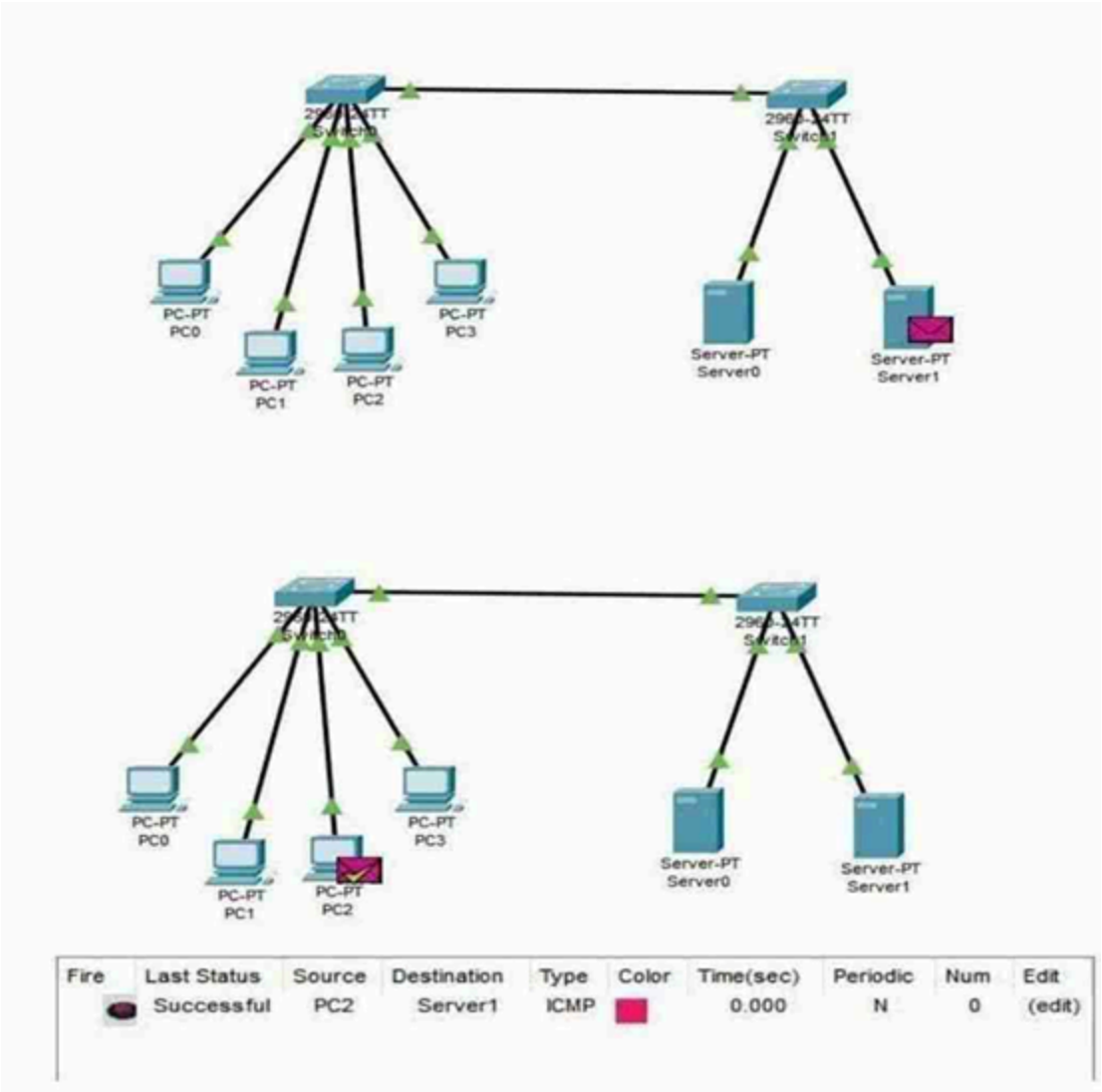
- Use the `show vlan brief` command to check VLAN and port assignments.

5. Test Communication:

- Ping between PCs in the same VLAN (successful).
- Ping between PCs in different VLANs (unsuccessful unless inter-VLAN routing is configured).

Output:


PC-PT
PC0
PC-PT
PC1
PC-PT
PC2
PC-PT
PC3
Server-PT
Server0
Server-PT
Server1
PC-PT
PC0
PC-PT
PC1
PC-PT
PC2
PC-PT
PC3
Server-PT
Server0
Server-PT
Server1
PC-PT
PC0
PC-PT
PC1
PC-PT
PC2
PC-PT
PC3
Server-PT
Server0
Server-PT
Server1



Result:

Thus the program is written and output is verified.

EX.NO: 8 b	WIRELESS LAN

Aim:

To Configuration of Wireless LAN using CISCO Packet Tracer.

Procedure:

1. Set Up Devices:

- Add a Wireless Router, a Server, and multiple Wireless PCs to the workspace.
- Use the Connections tab to connect the server to the wireless router using a straight-through cable.

2. Configure the Wireless Router:

- Click on the router and go to the GUI tab.
- Under Wireless Settings, set:
- SSID: e.g., `MyWirelessLAN`
- Channel: Select any available channel.
- Security Mode: WPA2-PSK.
- Passphrase: Set a password (e.g., `securepassword`).

3. Configure IP Address:

- Under LAN Setup, set:
- Router IP: e.g., `192.168.1.1`.
- DHCP Range: Enable DHCP and define the range (e.g., `192.168.1.2` to `192.168.1.50`).

4. Configure the Server:

- Set the server's IP address manually (e.g., `192.168.1.100`).

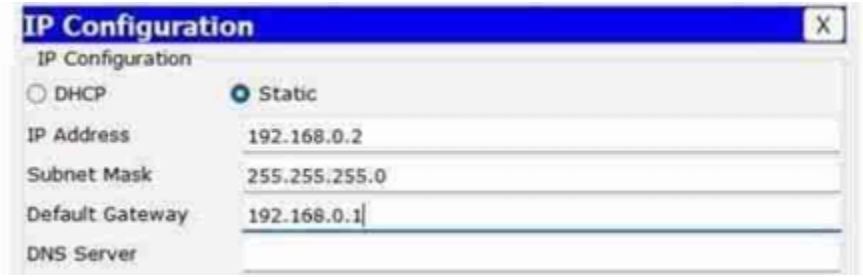
5. Connect Wireless PCs:

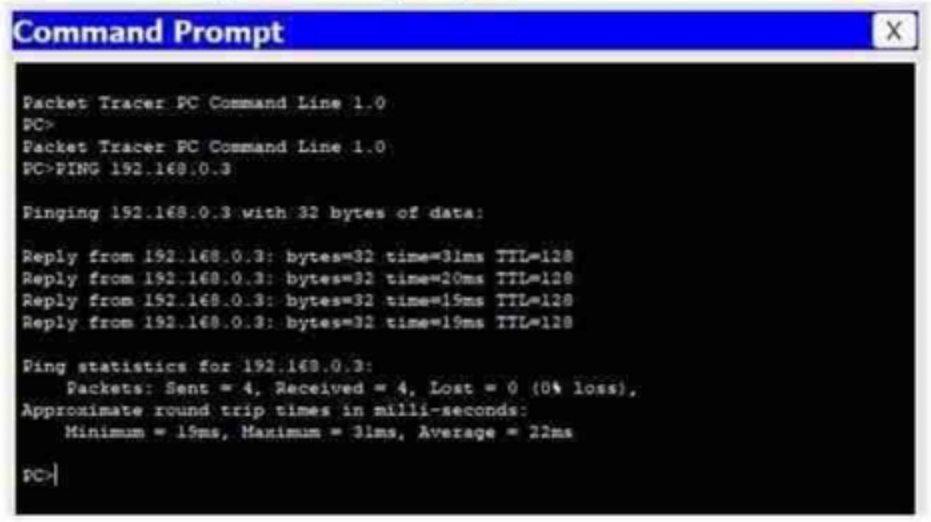
- Click on each wireless PC, go to Desktop > PC Wireless.
- Scan for the network, select SSID (MyWirelessLAN), and enter the password.

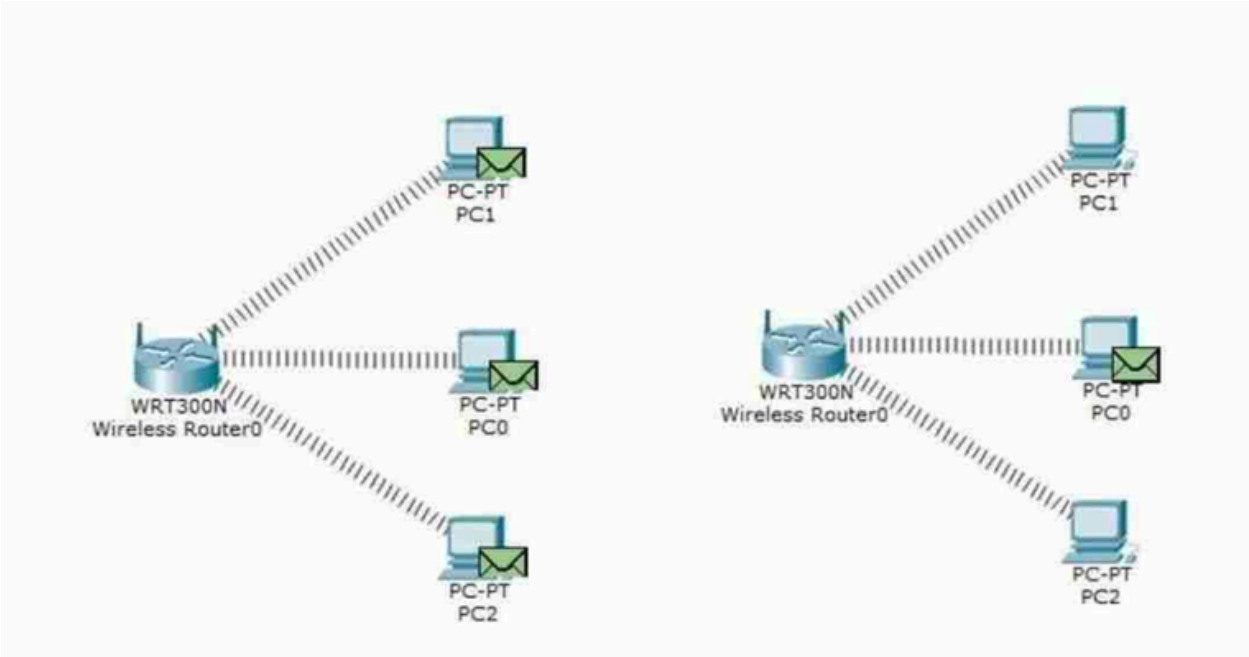
6. Test Connectivity:

- Ping the server from any wireless PC to verify connectivity.
- Ensure PCs connected to the same SSID can communicate.

Output:







Result:

Thus the output is verified for configuration of Wireless LAN using CISCO Packet Tracer.

EX.NO: 9	IMPLEMENTATION OF SUBNETTING

Aim:

To implement subnetting in CISCO PACKET TRACER simulator. a) Design multiple subnet with suitable number of hosts. b) Assign static IP address across all subnet and connect the subnets via Router. c) Simulate packet transmission across the subnets

Procedure:

1. Design Subnets:

- Calculate the number of subnets and hosts using CIDR.
- For example, for 4 subnets with at least 50 hosts each, use `192.168.1.0/24` divided into subnets like:
  - Subnet 1: `192.168.1.0/26`
  - Subnet 2: `192.168.1.64/26`
  - Subnet 3: `192.168.1.128/26`
  - Subnet 4: `192.168.1.192/26`

2. Set Up Network Topology:

- Add 4 PCs and 2 switches.
- Connect each PC to a switch using straight-through cables.
- Connect both switches to a router using straight-through cables.

3. Assign IPs:

- Assign static IPs to PCs in each subnet:
  - PC1: `192.168.1.1/26`, PC2: `192.168.1.2/26`.
  - PC3: `192.168.1.65/26`, PC4: `192.168.1.66/26`.

- Set the default gateway for each subnet to the router interface in the respective subnet.

4. Configure the Router:

- Access the router CLI and configure interfaces:

```plaintext
interface gigabitEthernet 0/0
ip address 192.168.1.1 255.255.255.192
no shutdown
exit
interface gigabitEthernet 0/1
ip address 192.168.1.65 255.255.255.192
no shutdown
exit
```

5. Test Connectivity:

- Ping PCs within the same subnet (successful).
- Ping PCs across subnets (successful, as the router connects them).

6. Simulate Packet Transmission:

- Use Simulation Mode in Packet Tracer.
- Observe packet flow between subnets via the router.

Output:

Fire	Last Status	Source	Destination	Type	Color	Time(sec)	Periodic	Num
	Successful	PC4(2)	Router2	ICMP		0.000	N	12
	Successful	PC4(2)	PC2(1)(1)	ICMP		0.000	N	13
	Successful	PC0	Router0	ICMP		0.000	N	14

Result:

Thus the output is verified.

EX.NO: 10	NETWORKING WITH ROUTERS USING CISCO

Aim:

To implement internetworking with routers in CISCO PACKET TRACER simulator.

Procedure:

(a) Simple Internetwork Using a Router

1. Design the Network:

- Create two networks, e.g., Network 1: `192.168.1.0/24` and Network 2: `192.168.2.0/24`.
- Add a router, two switches, and 3-4 PCs per network.

2. Connect Devices:

- Connect each PC to its respective switch using straight-through cables.
- Connect each switch to a different router interface using straight-through cables.

3. Assign IPs:

- Assign static IPs to PCs:
  - Network 1: `192.168.1.1` to `192.168.1.3`, gateway `192.168.1.254`.
  - Network 2: `192.168.2.1` to `192.168.2.3`, gateway `192.168.2.254`.
- Configure router interfaces:

```plaintext
interface gigabitEthernet 0/0
ip address 192.168.1.254 255.255.255.0
no shutdown
exit
```

```
interface gigabitEthernet 0/1
ip address 192.168.2.254 255.255.255.0
no shutdown
exit
```

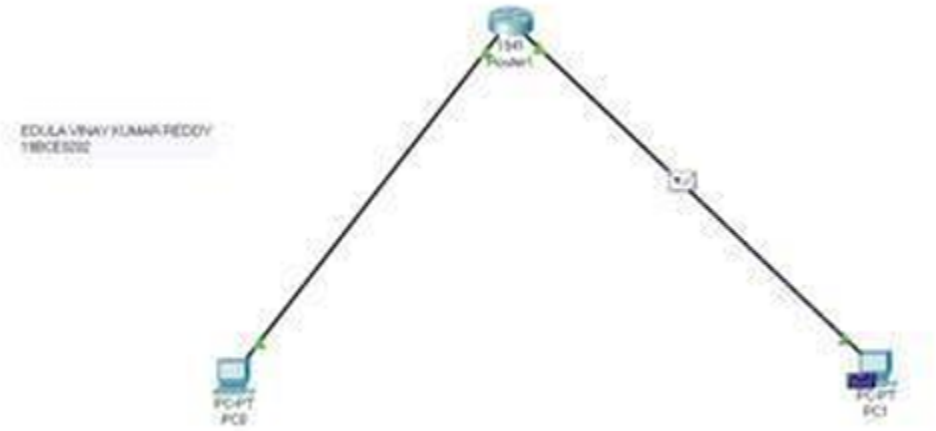
4. Test Connectivity:

- Ping between PCs in the same network.
- Ping between PCs in different networks to verify routing.

5. Simulate Packet Transmission:

- Switch to Simulation Mode, initiate a ping, and observe the routing path via the router.

Output:



(b) Internetwork Using Wireless Router, DHCP Server, and Internet Cloud

1. Set Up the Topology:

- Add a Wireless Router, a DHCP Server, an Internet Cloud, and 3-4 wireless PCs.

2. Configure Wireless Router:

- Set SSID, WPA2-PSK Security, and DHCP:
  - DHCP range: `192.168.0.100` to `192.168.0.200`.
- Assign Gateway: `192.168.0.1`.

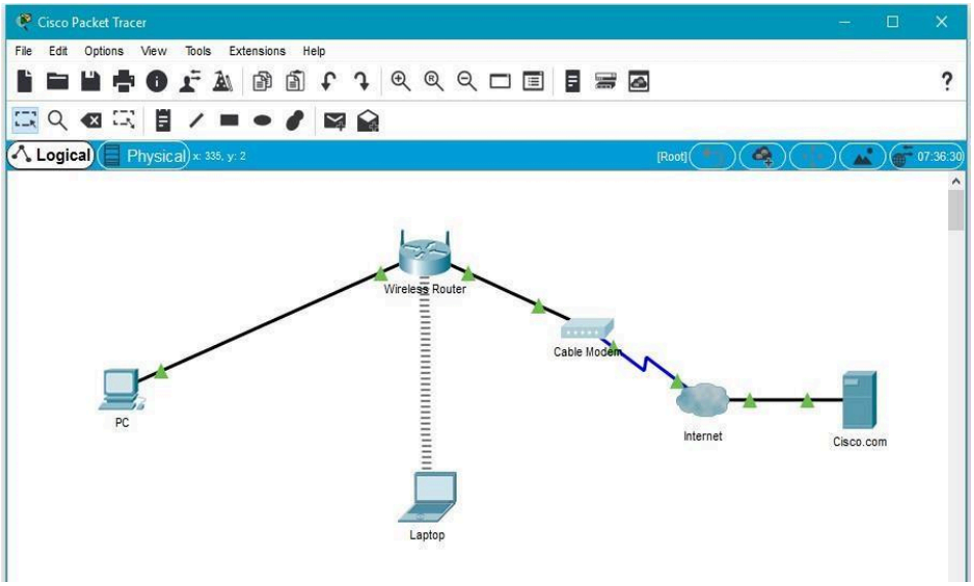
3. Configure the DHCP Server:

- Connect it to the router and set an IP (e.g., `192.168.0.2`).

4. Test Internet Access:

- Connect PCs to the wireless network.
- Verify they obtain IPs dynamically from the DHCP server.
- Use a cloud for internet simulation and ping external IPs.

Output:



(c) Internetwork in a Lab with Switch, Router, and Ethernet Cables

1. Set Up Devices:

- Add a Router, a Switch, and 3-4 PCs. Connect them with straight-through cables.

2. Configure the Router:

- Assign an IP to the router's interface (e.g., `192.168.3.254`).

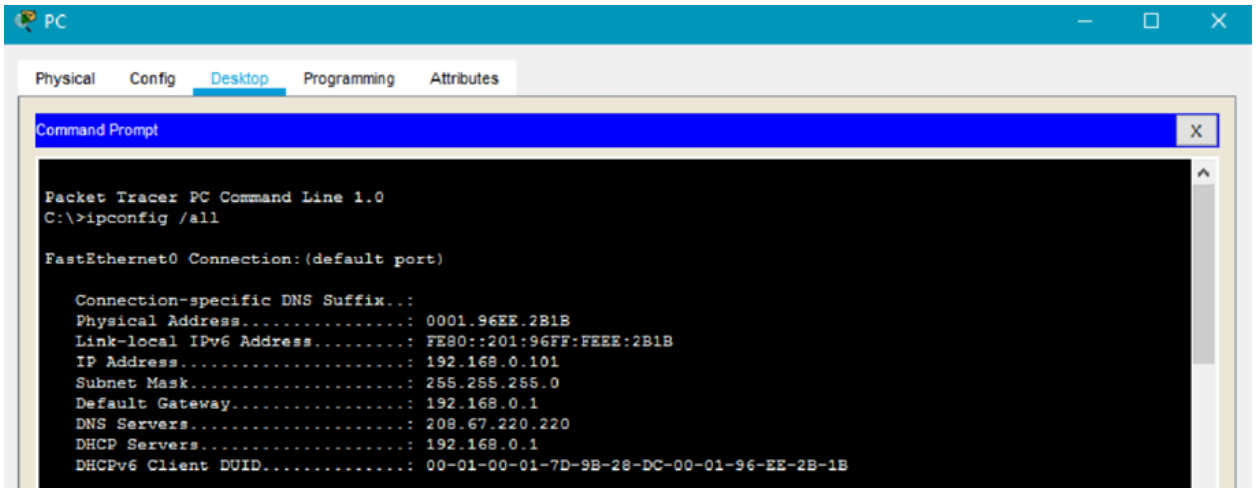
3. Assign IPs to PCs:

- Use static IPs (e.g., `192.168.3.1`, `192.168.3.2`, etc.).
- Set the default gateway for PCs to the router's IP.

4. Verify Connections:

- Ping between PCs to ensure connectivity.
- Simulate packet transmission and observe traffic via the router.

Output:



Result:

Thus the output is verified

EX.NO: 11	ROUTING AT NETWORK LAYER

Aim:

To implement routing at Network Layer.

a) Simulate Static Routing Protocol Configuration using CISCO Packet Tracer.

Procedure:

1.Set Up Topology:

- Add 2-3 routers, switches, and PCs for different networks.
- Connect routers with serial cables and PCs to switches using straight-through cables.

2. Assign IPs:

- Assign static IPs to PCs and router interfaces.
- Example:
  - Router1: `192.168.1.1` (LAN), `10.0.0.1` (WAN).
  - Router2: `192.168.2.1` (LAN), `10.0.0.2` (WAN).

3. Configure Static Routes:

- On Router1:

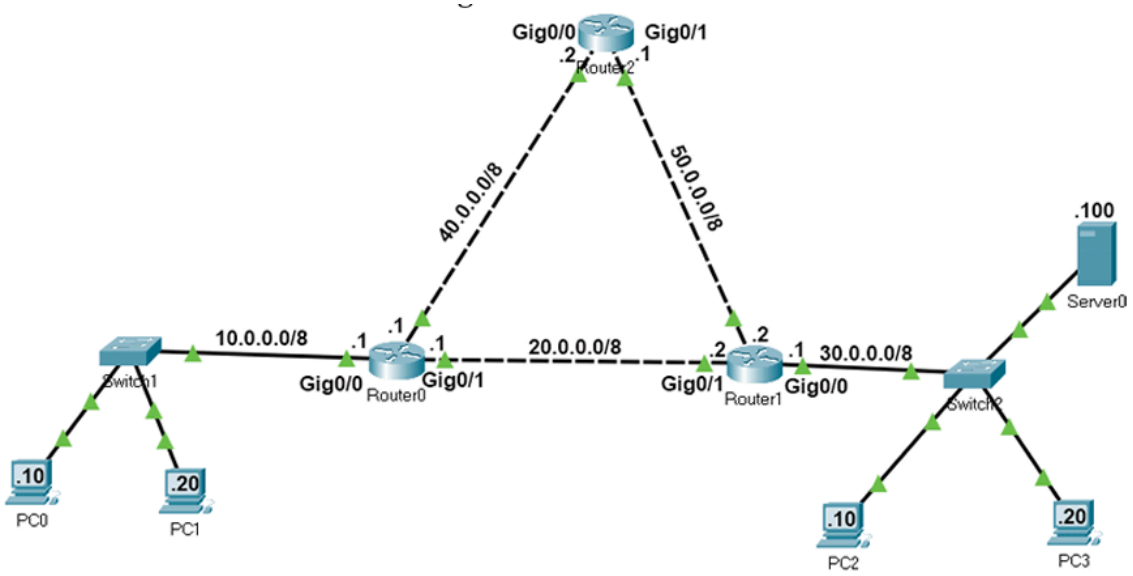
```plaintext
ip route 192.168.2.0 255.255.255.0 10.0.0.2
```

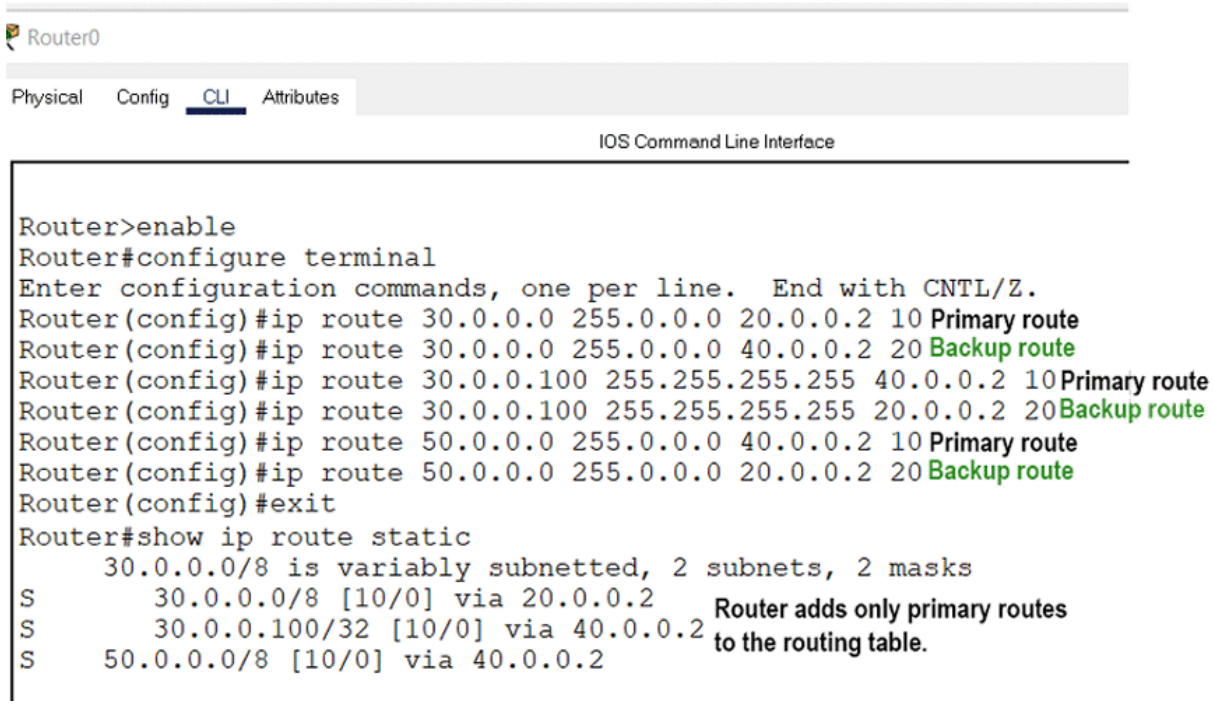
- On Router2:

```plaintext
ip route 192.168.1.0 255.255.255.0 10.0.0.1
```

4. Test Connectivity:

- Ping between PCs in different networks to verify routing.





Result:

Thus the program is written to simulate Static Routing Protocol Configuration using CISCO Packet Tracer.

b) Simulate RIP using CISCO Packet Tracer.

Procedure:

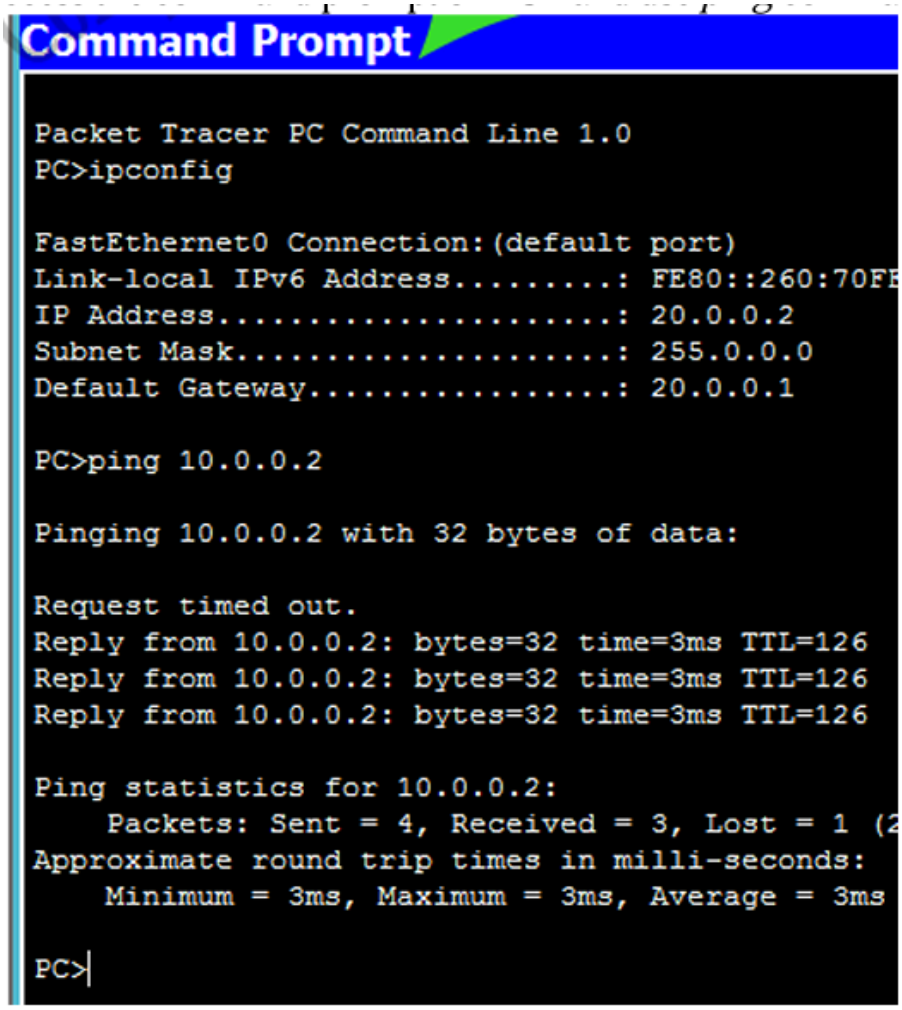
1. Set Up Topology:
   - Add 2-3 routers, switches, and PCs, and connect them as in static routing.

2. Assign IPs:
   - Assign static IPs to PCs and router interfaces.

3. Enable RIP:
   - On each router:
     ```plaintext
     router rip
     version 2
     network 192.168.1.0
     network 10.0.0.0
     ```
     (Add all relevant networks connected to the router).

4. Test Connectivity:
   - Ping between PCs in different networks to verify routing via RIP.

Output:



Result:

Thus the output is verified.

EX.NO: 12	ROUTING AT NETWORK LAYER

(a) Echo Client-Server Using TCP and UDP

Aim:

To implement Echo Client-Server Using TCP and UDP.

Procedure:

Server:

- Create a socket, bind it to a port, and start listening.
- For TCP, establish a connection; for UDP, receive data directly.
- Send received data back to the client.

Client:

- Create a socket, connect to the server (TCP) or directly send data (UDP).
- Display the response from the server.

Code:

TCP

Server:

```
import socket
server = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
server.bind(('localhost', 12345))
server.listen(1)
print("Server is ready.")
conn, addr = server.accept()
print(f"Connection from {addr}")
data = conn.recv(1024).decode()
print(f"Received: {data}")
conn.send(data.encode())
```

```
conn.close()
```

Output:

```
Server is ready.
Connection from ('127.0.0.1', <port>)
Received: Hello TCP!
```

Client:

```
import socket
client = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
client.connect(('localhost', 12345))
client.send(b'Hello TCP!')
print("Echoed:", client.recv(1024).decode())
client.close()
```

Output:

```
Echoed: Hello TCP!
```

UDP:

Server:

```
import socket
server = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
server.bind(('localhost', 12345))
print("Server is ready.")
data, addr = server.recvfrom(1024)
print(f"Received from {addr}: {data.decode()}")
server.sendto(data, addr)
```

Output:

Server is ready.
Received from ('127.0.0.1', <port>): Hello UDP!

Client:

```
import socket
client = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
client.sendto(b'Hello UDP!', ('localhost', 12345))
data, addr = client.recvfrom(1024)
print("Echoed:", data.decode())
```

Output:

Echoed: Hello UDP!

(b) Chat Program Using Socket Programming

Aim:

To implement chat program using socket programming

Procedure:

Server:

- Create a socket and bind it to a port.
- Wait for a client connection.
- Continuously receive and respond to messages until either party sends "exit".

Client:

- Connect to the server and exchange messages.
- Terminate the connection when "exit" is sent.

Program:

Server:

```python
import socket
server = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
server.bind(('localhost', 12345))
server.listen(1)
print("Server is ready.")
conn, addr = server.accept()
print(f"Connected to {addr}")
while True:
    msg = conn.recv(1024).decode()
    if msg.lower() == "exit":
        print("Client disconnected.")
        break
    print(f"Client: {msg}")
    conn.send(input("Server: ").encode())
conn.close()
```

Output:

Server is ready.
Connected to ('127.0.0.1', <port>)
Client: Hi, Server!
Server: How are you?
Client: exit
Client disconnected.

Client:

```
import socket
client = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
client.connect(('localhost', 12345))
print("Connected to the server.")
while True:
    msg = input("Client: ")
    client.send(msg.encode())
    if msg.lower() == "exit":
        print("Disconnected.")
        break
    print(f"Server: {client.recv(1024).decode()}")
client.close()
```

Output:

Connected to the server.
Client: Hi, Server!
Server: How are you?
Client: exit
Disconnected.

Result:

Thus the output is verified.

EX.NO: 13	PING PROGRAM

Aim:

To write a python program for implementing ping program

Procedure:

1. Understand ICMP and Ping:
   - Ping sends ICMP Echo Requests to a destination and listens for ICMP Echo Replies.
   - It measures the round-trip time (RTT) for packets.

2. Set Up Socket:
   - Use Python's `socket` module to create raw sockets for ICMP packets.

3. Create ICMP Packet:
   - Build an ICMP Echo Request packet manually with appropriate headers and a checksum.

4. Send and Receive Packets:
   - Send the ICMP packet to the target IP address.
   - Record the time before sending and after receiving the response to calculate RTT.

5. Handle Permissions:
   - Root/Administrator privileges are required to create raw sockets.

Program:

Server:

```
import socket

server_socket = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
server_socket.bind(('localhost', 12345))
print("Server is ready to receive pings.")

while True:
    data, addr = server_socket.recvfrom(1024)  # Receive ping message
    print(f"Received '{data.decode()}' from {addr}")
    server_socket.sendto(data, addr)  # Echo the message back
```

Client:

```
import socket
import time

# Client setup
client_socket = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
server_address = ('localhost', 12345)

# Send "ping" request and measure RTT
message = "ping"
start_time = time.time()
client_socket.sendto(message.encode(), server_address)
data, server = client_socket.recvfrom(1024)  # Receive echo from server
end_time = time.time()

rtt = (end_time - start_time) * 1000  # Round-trip time in milliseconds
print(f"Reply from {server}: '{data.decode()}' RTT = {rtt:.2f}ms")
```

Output:

Server:

```
========== RESTART: D:/server.py ==========
===
UDP Server running on 127.0.0.1:12345
Received message from ('127.0.0.1', 63929): Ping
Received message from ('127.0.0.1', 63922): Ping
Received message from ('127.0.0.1', 56820): Ping
```

Client:

```
>>>
========== RESTART: D:/client.py ==========
===
Received Ping from ('127.0.0.1', 12345) in 0.01 seconds
>>>
========== RESTART: D:/client.py ==========
Received Ping from ('127.0.0.1', 12345) in 0.07 seconds
>>>
========== RESTART: D:/client.py ==========
Received Ping from ('127.0.0.1', 12345) in 0.01 seconds
>>>
```

Result:

Thus the output is verified

EX.NO: 15	TYPES OF SERVER USING WEBALIZER TOOL

Aim:

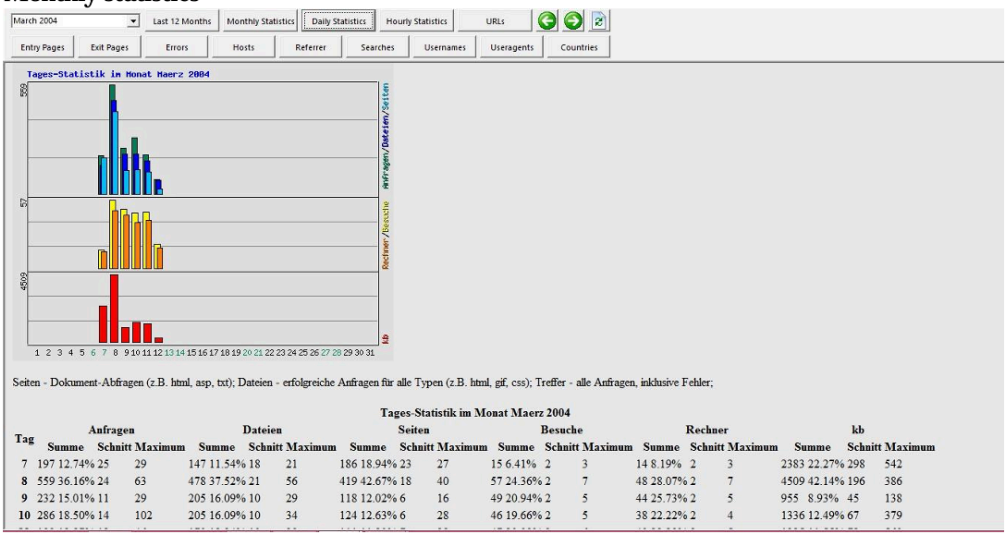
To analyze the different types of web logs using Webalizer tool.

Procedure:

- Step1: Run webalizer windows version
- Step2. Input web log file (down load from web)
- Step3: Press Run webalizer

Output:

Monthly statistics

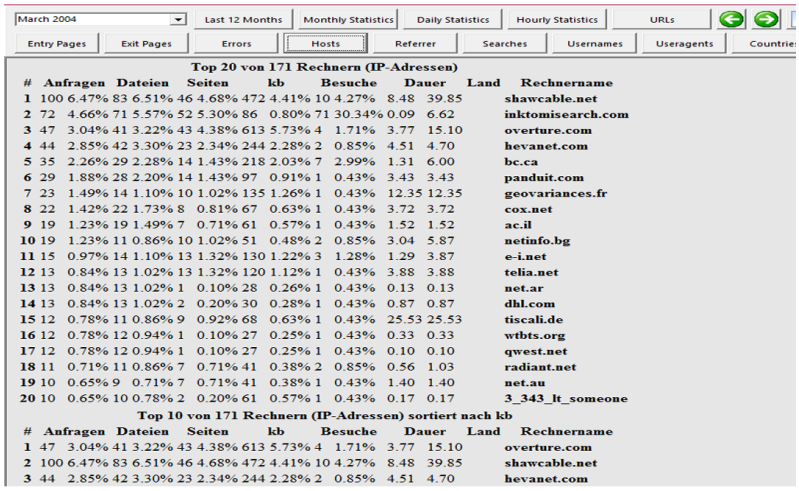


Seiten - Dokument-Abfragen (z.B. html, asp, txt); Dateien - erfolgreiche Anfragen für alle Typen (z.B. html, gif, css); Treffer - alle Anfragen, inklusive Fehler;

Tages-Statistik im Monat Maerz 2004

Tag	Anfragen			Dateien			Seiten			Besuche			Rechner			kb		
	Summe	Schnitt	Maximum	Summe	Schnitt	Maximum	Summe	Schnitt	Maximum	Summe	Schnitt	Maximum	Summe	Schnitt	Maximum	Summe	Schnitt	Maximum
7	197 12.74%	25	29	147 11.54%	18	21	186 18.94%	23	27	15 6.41%	2	3	14 8.19%	2	3	2383 22.27%	298	542
8	559 36.16%	24	63	478 37.52%	21	56	419 42.67%	18	40	57 24.36%	2	7	48 28.07%	2	7	4509 42.14%	196	386
9	232 15.01%	11	29	205 16.09%	10	29	118 12.02%	6	16	49 20.94%	2	5	44 25.73%	2	5	955 8.93%	45	138
10	286 18.50%	14	102	205 16.09%	10	34	124 12.63%	6	28	46 19.66%	2	5	38 22.22%	2	4	1336 12.49%	67	379

Hosts



March 2004 | Last 12 Months | Monthly Statistics | Daily Statistics | Hourly Statistics | URLs

Entry Pages | Exit Pages | Errors | Hosts | Referrer | Searches | Usernames | Useragents | Countries

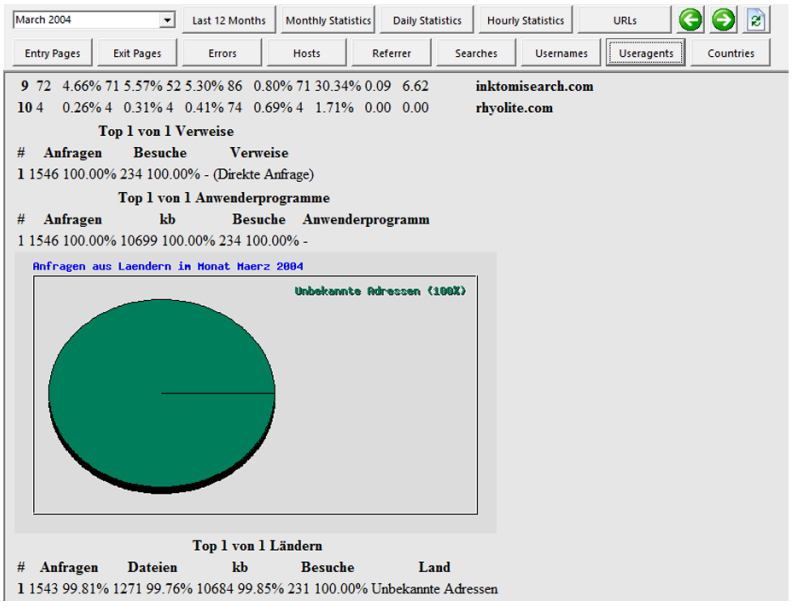
Top 20 von 171 Rechnern (IP-Adressen)

#	Anfragen		Dateien		Seiten		kb		Besuche		Dauer	Land	Rechnername
1	100	6.47%	83	6.51%	46	4.68%	472	4.41%	10	4.27%	8.48	39.85	shawcable.net
2	72	4.66%	71	5.57%	52	5.30%	86	0.80%	71	30.34%	0.09	6.62	inktomisearch.com
3	47	3.04%	41	3.22%	43	4.38%	613	5.73%	4	1.71%	3.77	15.10	overture.com
4	44	2.85%	42	3.30%	23	2.34%	244	2.28%	2	0.85%	4.51	4.70	hevanet.com
5	35	2.26%	29	2.28%	14	1.43%	218	2.03%	7	2.99%	1.31	6.00	bc.ca
6	29	1.88%	28	2.20%	14	1.43%	97	0.91%	1	0.43%	3.43	3.43	panduit.com
7	23	1.49%	14	1.10%	10	1.02%	135	1.26%	1	0.43%	12.35	12.35	geovariances.fr
8	22	1.42%	22	1.73%	8	0.81%	67	0.63%	1	0.43%	3.72	3.72	cox.net
9	19	1.23%	19	1.49%	7	0.71%	61	0.57%	1	0.43%	1.52	1.52	ac.il
10	19	1.23%	11	0.86%	10	1.02%	51	0.48%	2	0.85%	3.04	5.87	netinfo.bg
11	15	0.97%	14	1.10%	13	1.32%	130	1.22%	3	1.28%	1.29	3.87	e-i.net
12	13	0.84%	13	1.02%	13	1.32%	120	1.12%	1	0.43%	3.88	3.88	telia.net
13	13	0.84%	13	1.02%	1	0.10%	28	0.26%	1	0.43%	0.13	0.13	net.ar
14	13	0.84%	13	1.02%	2	0.20%	30	0.28%	1	0.43%	0.87	0.87	dhl.com
15	12	0.78%	11	0.86%	9	0.92%	68	0.63%	1	0.43%	25.53	25.53	tiscali.de
16	12	0.78%	12	0.94%	1	0.10%	27	0.25%	1	0.43%	0.33	0.33	wtbts.org
17	12	0.78%	12	0.94%	1	0.10%	27	0.25%	1	0.43%	0.10	0.10	qwest.net
18	11	0.71%	11	0.86%	7	0.71%	41	0.38%	2	0.85%	0.56	1.03	radiant.net
19	10	0.65%	9	0.71%	7	0.71%	41	0.38%	1	0.43%	1.40	1.40	net.au
20	10	0.65%	10	0.78%	2	0.20%	61	0.57%	1	0.43%	0.17	0.17	3_343_lt_someone

Top 10 von 171 Rechnern (IP-Adressen) sortiert nach kb

#	Anfragen		Dateien		Seiten		kb		Besuche		Dauer	Land	Rechnername
1	47	3.04%	41	3.22%	43	4.38%	613	5.73%	4	1.71%	3.77	15.10	overture.com
2	100	6.47%	83	6.51%	46	4.68%	472	4.41%	10	4.27%	8.48	39.85	shawcable.net
3	44	2.85%	42	3.30%	23	2.34%	244	2.28%	2	0.85%	4.51	4.70	hevanet.com

User-agents



March 2004 | Last 12 Months | Monthly Statistics | Daily Statistics | Hourly Statistics | URLs

Entry Pages | Exit Pages | Errors | Hosts | Referrer | Searches | Usernames | Useragents | Countries

9	72	4.66%	71	5.57%	52	5.30%	86	0.80%	71	30.34%	0.09	6.62	inktomisearch.com
10	4	0.26%	4	0.31%	4	0.41%	74	0.69%	4	1.71%	0.00	0.00	rhyolite.com

Top 1 von 1 Verweise

#	Anfragen		Besuche		Verweise
1	1546	100.00%	234	100.00%	- (Direkte Anfrage)

Top 1 von 1 Anwenderprogramme

#	Anfragen		kb		Besuche		Anwenderprogramm
1	1546	100.00%	10699	100.00%	234	100.00%	-

Top 1 von 1 Ländern

#	Anfragen		Dateien		kb		Besuche		Land
1	1543	99.81%	1271	99.76%	10684	99.85%	231	100.00%	Unbekannte Adressen

Result:

Thus the different types of web logs using Webalizer tool are studied.