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 : Sudharsan S

Register No. : 2116221501149

Year / Branch / Section : 3rd Year / 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: Sudharsan S

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

Register No. 2116221501149

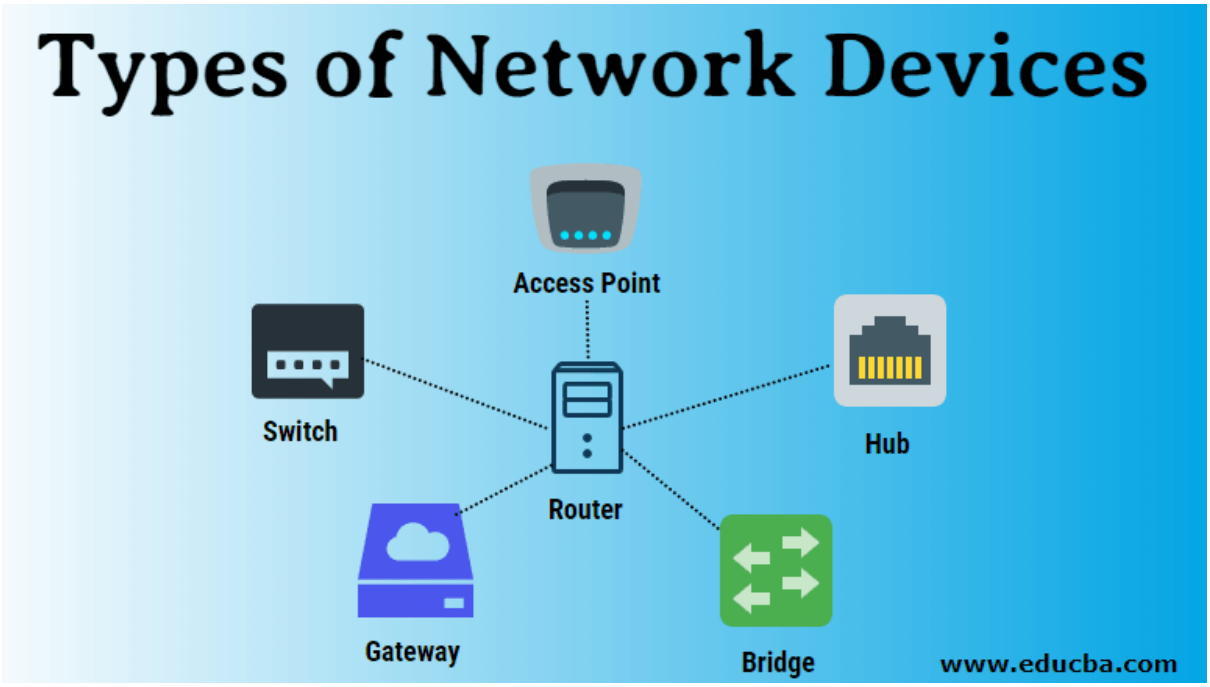
Certified that this is the bonafide record of work done by the above student in the Computer Networks (CS19341) Laboratory during the academic year 2024-2025

Signature of Faculty in-charge

Submitted for the Practical Examination held on 20.11.2024

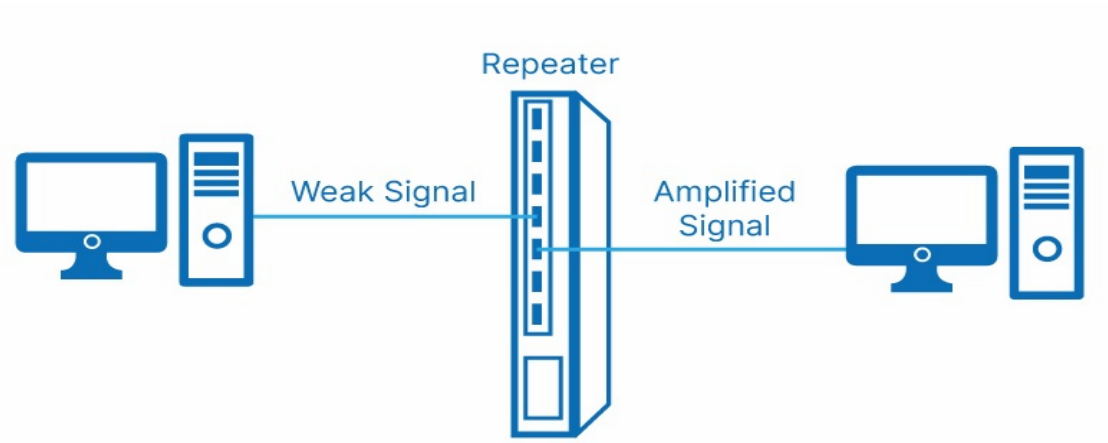
Internal Examiner **External Examiner**

Network Devices (Hub, Repeater, Bridge, Switch, Router and Gateways)



1.Repeater :

A repeater operates at the physical layer. Its job is to regenerate the signal over the same network before the signal becomes too weak or corrupted so as to extend the length to which the signal can be transmitted over the same network. An important point to be noted about repeaters is that they do no amplify the signal. When the signal becomes weak, they copy the signal bit by bit and regenerate it at the original strength. It is a 2 port device. A single Ethernet segment can have a maximum length of 500 meters with a maximum of 100 stations (in a cheapernet segment it is 185m). Functionally, a repeater can be considered as two transceivers joined together and connected to two different segments of coaxial cable. The repeater passes the digital signal bit-by-bit in both directions between the two segments. As the signal passes through a repeater, it is amplified and regenerated at the other end. The repeater does not isolate one segment from the other, if there is a collision on one segment, it is regenerated on the other segment. Therefore, the two segments form a single LAN and it is transparent to rest of the system. Ethernet allows five segments to be used in cascade to have a maximum network span of 2.5 km. It simply repeats, retimes and amplifies the bits it receives. The repeater is merely used to extend the span of a single LAN.

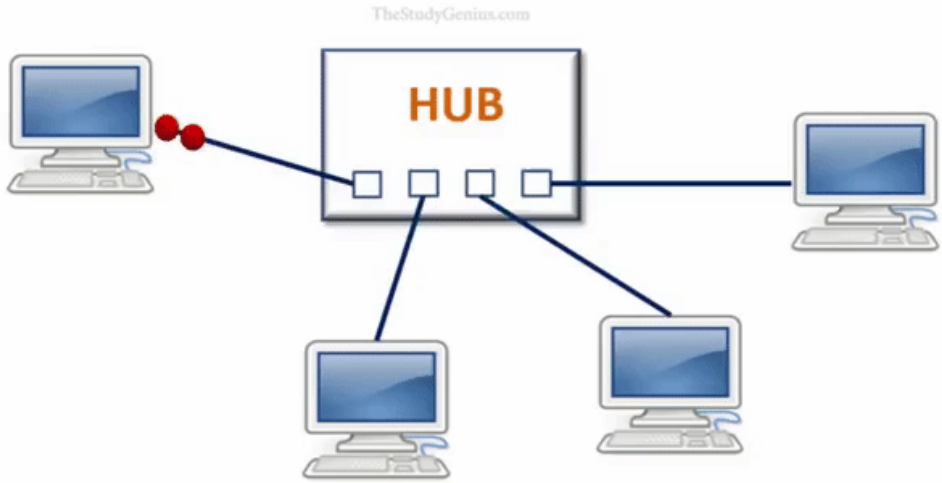


Important features of a repeater are as follows:

- A repeater connects different segments of a LAN
- A repeater forwards every frame it receives
- A repeater is a regenerator, not an amplifier
- It can be used to create a single extended LAN

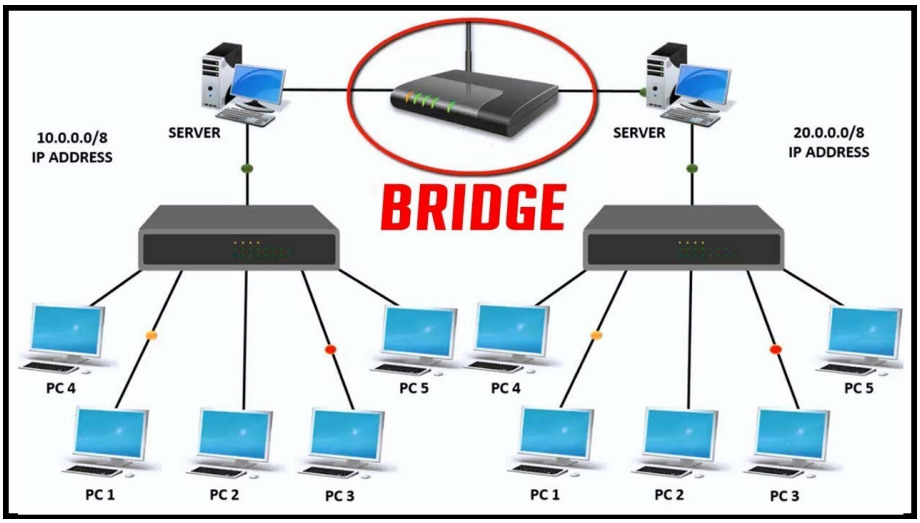
2. Hub :

A hub is basically a multiport repeater. A hub connects multiple wires coming from different branches, for example, the connector in star topology which connects different stations. Hubs cannot filter data, so data packets are sent to all connected devices. In other words, collision domain of all hosts connected through Hub remains one. Also, they do not have intelligence to find out best path for data packets which leads to inefficiencies and wastage. Hub is a generic term, but commonly refers to a multiport repeater. It can be used to create multiple levels of hierarchy of stations. The stations connect to the hub with RJ-45 connector having maximum segment length is 100 meters. This type of interconnected set of stations is easy to maintain and diagnose. Figure shows how several hubs can be connected in a hierarchical manner to realize a single LAN of bigger size with a large number of nodes.



3. Bridge :

A bridge operates at data link layer. A bridge is a repeater, with add on functionality of filtering content by reading the MAC addresses of source and destination. It is also used for interconnecting two LANs working on the same protocol. It has a single input and single output port, thus making it a 2 port device. The device that can be used to interconnect two separate LANs is known as a bridge. The bridge operates in layer 2, that is data-link layer and that is why it is called level-2 relay with reference to the OSI model. It links similar or dissimilar LANs, designed to store and forward frames, it is protocol independent and transparent to the end stations.Use of bridges offer a number of advantages, such as higher reliability, performance, security, convenience and larger geographic coverage. But, it is desirable that the quality of service (QOS) offered by a bridge should match that of a single LAN. The parameters that define the QOS include availability, frame mishaps, transit delay, frame lifetime, undetected bit errors, frame size and priority.



Key features of a bridge are mentioned below:

• A bridge operates both in physical and data-link layer

• A bridge uses a table for filtering/routing

• A bridge does not change the physical (MAC) addresses in a frame

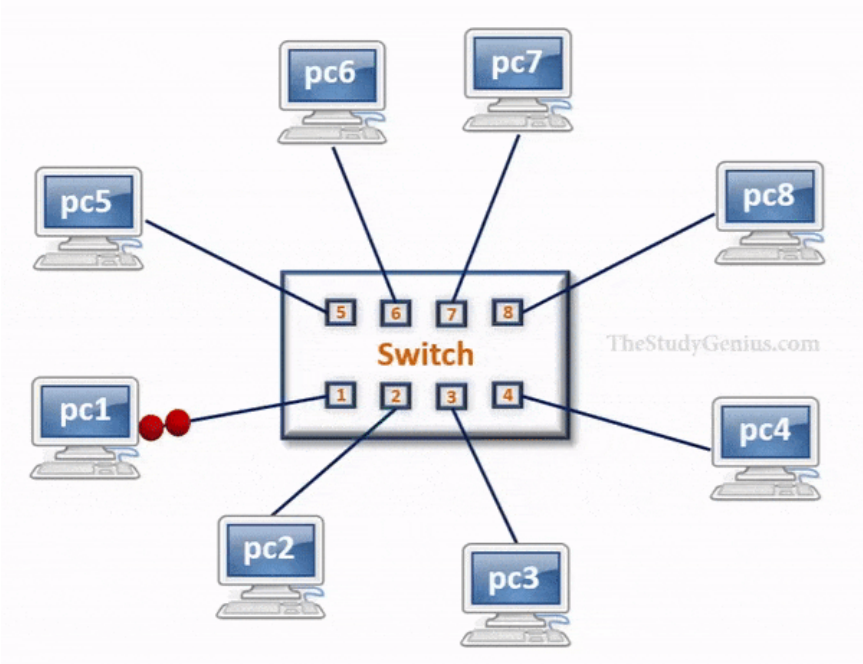
Types of bridges:

o Transparent Bridges

o Source routing bridges

4. Switch :

A switch is a multi port bridge with a buffer and a design that can boost its efficiency(large number of ports imply less traffic) and performance. Switch is data link layer device. Switch can perform error checking before forwarding data, that makes it very efficient as it does not forward packets that have errors and forward good packets selectively to correct port only. In other words, switch divides collision domain of hosts, butbroadcast domain remains same. A switch is essentially a fast bridge having additional sophistication that allows faster processing of frames.



Some of important functionalities are:

• Ports are provided with buffer

• Switch maintains a directory: #address - port#

• Each frame is forwarded after examining the #address and forwarded to the proper port#

Three possible forwarding approaches: Cut-through, Collision-free and Fullybuffered as briefly explained below.

Cut-through: A switch forwards a frame immediately after receiving the destination address. As a consequence, the switch forwards the frame without collision and error detection.

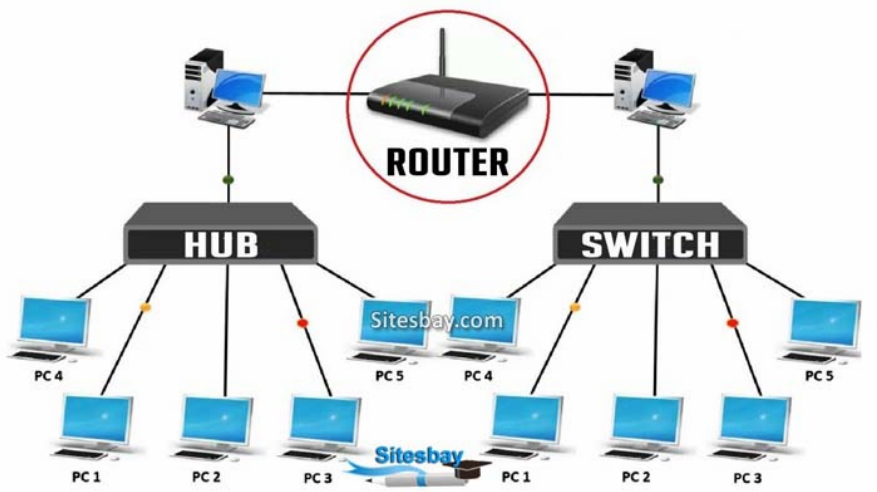
Collision-free: In this case, the switch forwards the frame after receiving 64 bytes, which allows detection of collision. However, error detection is not possible because switch is yet to receive the entire frame.

Fully buffered: In this case, the switch forwards the frame only after receiving the entire frame. So, the switch can detect both collision and error free frames are forwarded.

5. Routers :

A router is a device like a switch that routes data packets based on their IP addresses. Router is mainly a Network Layer device. Routers normally connect LANs and WANs together and have a dynamically updating routing table based on which they make decisions on routing the data packets. Router divide broadcast domains of hosts connected through it. A router is used to route data packets between two networks. It reads the information in each packet to tell where it is going. If it is destined for an immediate network it has access to, it will strip the outer packet (IP packet for example), readdress the packet to the proper ethernet address, and transmit it on that network. If it is destined for another network and must be sent to another router, it will re-package the outer packet to be received by the next router and send it to the next router. Routing occurs at the network layer of the OSI model. They can connect networks with different architectures such as Token Ring and Ethernet. Although they

can transform information at the data link level, routers cannot transform information from one data format such as TCP/IP to another such as IPX/SPX. Routers do not send broadcast packets or corrupted packets. If the routing table does not indicate the proper address of a packet, the packet is discarded.



There are two types of routers:

1. Static routers - Are configured manually and route data packets based on information in a router table.

2. Dynamic routers - Use dynamic routing algorithms.

There are two types of algorithms:

o Distance vector - Based on hop count, and periodically broadcasts the routing table to other routers which takes more network bandwidth especially with more routers. RIP uses distance vectoring. Does not work on WANs as well as it does on LANs.

o Link state - Routing tables are broadcast at startup and then only when they change. The open shortest path first (OSPF) protocol uses the link state routing method to configure routes or distance vector algorithm (DVA).

Common routing protocols include:

- IS-IS -Intermediate system to intermediate system which is a routing protocol for the OSI suite of protocols.

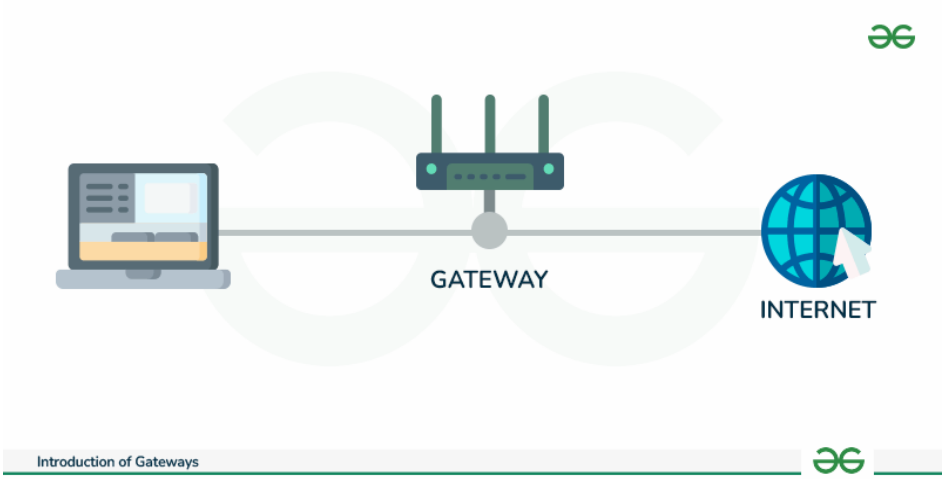
• IPX - Internet Packet Exchange. Used on Netware systems.

• NLSP - Netware Link Services protocol - Uses OSPF algorithm and is replacing IPX to provide internet capability.

• RIP - Routing information protocol uses a distance vector algorithm. There is a device called a brouter which will function similar to a bridge for network transport protocols that are not routable, and will function as a router for routable protocols. It functions at the network and data link layers of the OSI network model.

6. Gateway :

A gateway, as the name suggests, is a passage to connect two networks together that may work upon different networking models. They basically works as the messenger agents that take data from one system, interpret it, and transfer it to another system. Gateways are also called protocol converters and can operate at any network layer. Gateways are generally more complex than switch or router. A gateway can translate information between different network data formats or network architectures.



It can translate TCP/IP to AppleTalk so computers supporting TCP/IP can communicate with Apple brand computers. Most gateways operate at the application layer, but can operate at the network or session layer of the OSI model. Gateways will start at the lower level and strip information until it gets to the required level and repackage the information and work its way back toward the hardware layer of the OSI model. To confuse issues, when talking about a router that

is used to interface to another network, the word gateway is often used. This does not mean the routing machine is a gateway as defined here, although it could be.

DIFFERENT TYPES OF NETWORK CABLES

TYPES OF NETWORK CABLES:

➔ **Coaxial Cable**

a. It contains a conductor, insulator, braiding, and sheath which covers each other from the center respectively.

b. There are two types based on the number of cores present. They are

   i. Single Core Coaxial Cable

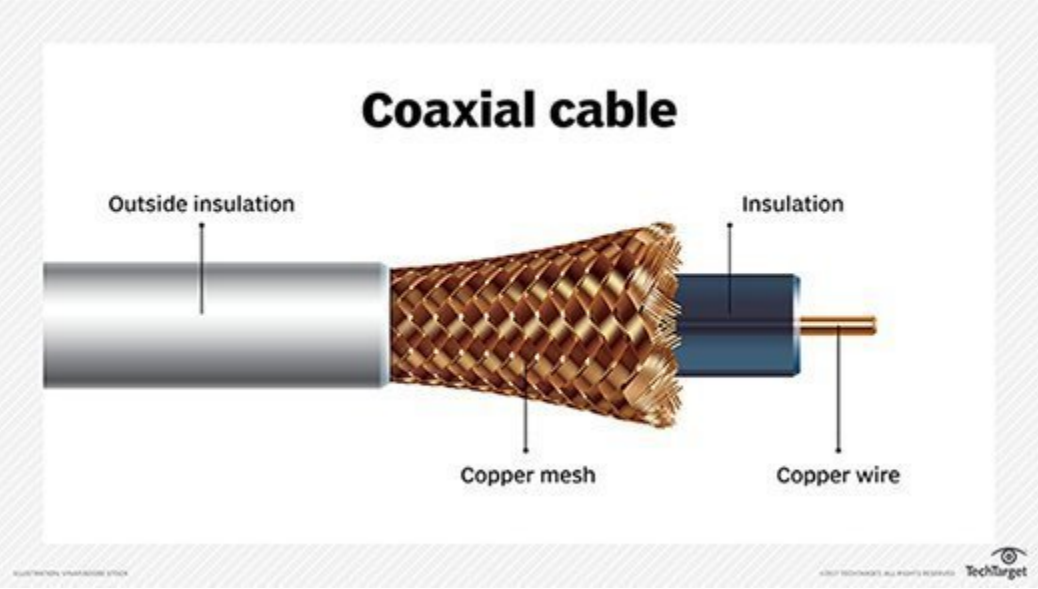
   ii. Multi Core Coaxial Cable

c. Some of the few Coaxial Cable used in Computer Networks are,

   RG-6 - Used in cable networks to provide cable Internet service and cable TV over long distances.

   RG-8 - Used in the earliest computer networks. This cable was used as the backbone cable in the bus topology. In Ethernet standards, this cable is documented as the 10base5 Thicknet cable.

   RG-58 - This cable is thinner, easier to handle and install than the RG-8 cable. This cable was used to connect a system with the backbone cable. In Ethernet standards, this



   cable is documented as the 10base2 Thinnet cable.

   RG-59 - Used in cable networks to provide short-distance service.

➔ **Twisted pair cable**

a. It has been primarily developed for computer networks commonly known as Ethernet Cable. It is commonly used for most of the Local Area Network (LAN)

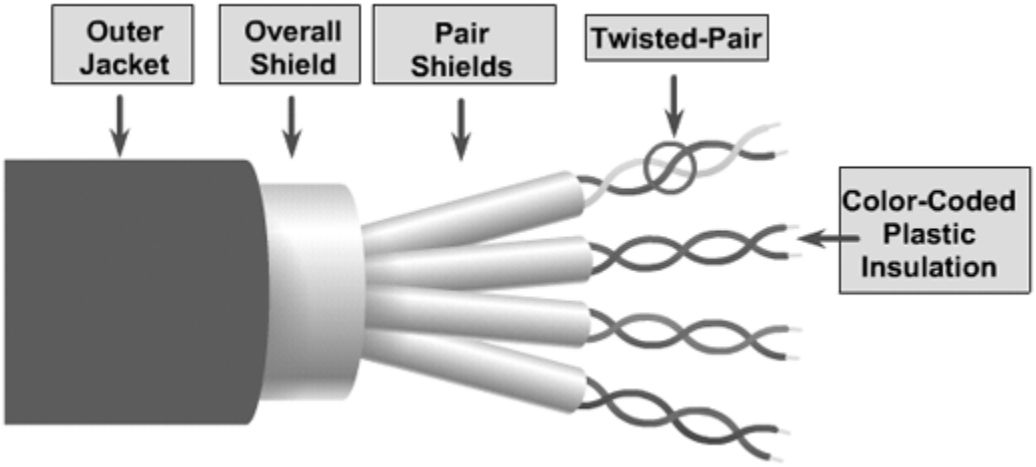
b. This cable consists of color-coded pairs of insulated copper wires. Every two wires are twisted around each other to form a pair. Usually, there are four pairs. Each pair has one solid color and one striped color wire. Solid colors are blue, brown, green, and orange. In striped color, the solid color is mixed with the white color.

c. Based on how pairs striped in the sheath , there are two types they are,

    i. UTP (Unshielded Twisted Pair)

    ii. STP (Shielded Twisted Pair)

d. Some of the Twisted Pair Cables are cat1, cat2, cat3, cat4, cat5, cat5e, cat6, cat6a, cat7 among them cat5e, cat6, cat6a are commonly used.



➔ **Fiber cable**

a. It consists of core, cladding, buffer and jacket.

    i. Core carries the data signals in the form of light.

    ii. Cladding reflects light back to the core.

    iii. Buffer protects the light from leaking.

    iv. The jacket protects the cable from physical damage.

b. Fiber optic cable is completely immune to EMI (Electromagnetic Inference)and RFI (Radio Frequency Inference)

c. Based on how many beams of light transmit there are two types they are,

i. **SMF (** Single mode Fiber**)** Optical Cable

ii. **MMF (**Multi mode Fiber**)** Optical Cable

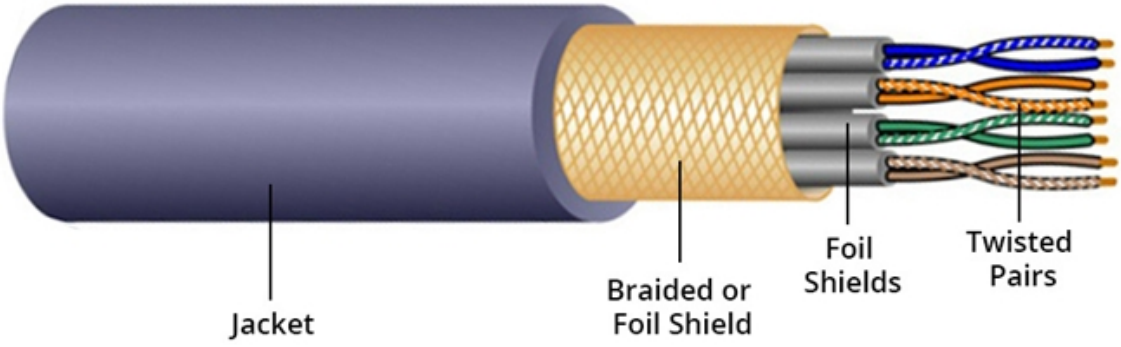
d.

i. SMF (Single-mode fiber) optical cable

1. This cable carries only a single beam of light. This is more reliable and supports much higher bandwidth and longer distances than the MMF cable. This cable uses a laser as the light source and transmits 1300 or 1550 nano-meter wavelengths of light.

ii. MMF (multi-mode fiber) optical cable

1. This cable carries multiple beams of light. Because of multiple beams, this cable carries much more data than the SMF cable. This cable is used for shorter distances. This cable uses an LED as the light source and transmits 850 or 1300 nano-meter



wavelengths of light.

Difference Between Twisted Pair , Fibre Optical and Coaxial Cables:-

Characteristics	UTP	STP	Coaxial Cables	Fiber Optic Cables
Bandwidth	10 Mbps - 100 Mbps	10 Mbps - 100 Mbps	10 Mbps	100 Mbps -1 Gbps
Maximum cable segment	100 meters	100 meters	200 - 500 meters	2 k.m. - 100 k.m.
Interference rating	Poor	Better than UTP	Better than Twisted Pair Cable	Very good as compared to any other cable
Installation cost	Cheap	Costly than UTP	Costlier than twisted pair wires	Costliest to install
Bend radius	360 degrees / feet	360 degrees / feet	360 degrees / feet or 30 degrees / feet	30 degrees / feet
Security	Low	Low	Low	High

BASIC NETWORKING COMMANDS:

arp –a:-
ARP is 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.

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

ipconfig /all:
This command displays detailed configuration information about your TCP/IPconnection including Router, Gateway, DNS, DHCP, and type of ethernet adapter in your system.

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

Netstat:
(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.
e.g.:- netstat -r

Nslookup:
(name server lookup) 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.
e.g.:- nslookup www.google.com

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.

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

1. #ping hostname(ping localhost)
2. #ping ip address (ping 4.2.2.2)
3. #ping fully qualified domain name(ping www.facebook.com)

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.

Building Networks

The concept of a network can encompass a physical infrastructure connecting devices or a web of social connections linking individuals. Understanding how to create and connect these networks is valuable in both technical and social spheres. This detailed guide explores the processes involved in forming and connecting both physical networks, commonly used in homes and offices, and social networks, essential for building communities and fostering collaboration.

Delving Deeper into Physical Networks: The Backbone of Connectivity

The Building Blocks of a Physical Network:

- **Cables:** These are the physical pathways through which data travels. The two most common types used in LANs are:
    - **Twisted-Pair Cable:** This consists of four pairs of insulated copper wires twisted together to reduce interference. It's a cost-effective and widely used option, offering good performance for typical home and office networks.
    - **Fiber Optic Cable:** This transmits data using light pulses instead of electrical signals. Fiber optic cables offer superior speed, bandwidth, and immunity to interference compared to twisted-pair cables, but they are generally more expensive and require specialized equipment for installation.
- **Network Interface Cards (NICs):** These are expansion cards installed in computers or embedded chips in laptops and other devices. NICs convert digital data from the device into electrical signals (for twisted-pair) or light pulses (for fiber optic) for transmission over the network cable and vice versa.
- **Network Devices:** These are the hardware components that manage and direct data flow within the network. Some key players include:
    - **Routers:** As mentioned earlier, routers act as traffic directors, receiving data packets from one device and forwarding them to the designated recipient on the network or the wider internet (if applicable).
    - **Switches:** Switches provide dedicated connections between devices. They are particularly beneficial in larger networks where multiple devices compete for bandwidth on a single router port. A switch learns the MAC addresses (unique identifiers) of connected devices and forwards data packets only to the intended recipient, improving network efficiency.
    - **Hubs:** While less common in modern networks, hubs simply broadcast all data packets received on one port to all other connected devices. This can be inefficient, especially in larger networks, as devices receive irrelevant data traffic.

Network Topologies: Mapping the Connections

The physical layout or arrangement of connections between devices in a network is called its topology. Here are some common topologies:

- **Bus:** All devices are connected to a single central cable. This is a simple and cost-effective option, but a failure in the central cable disrupts the entire network.
- **Star:** Each device has a dedicated connection to a central hub or switch. This is a more reliable and scalable option compared to a bus topology. However, the performance of the network depends on the central device's capacity.
- **Mesh:** Devices connect to each other in a web-like fashion, creating multiple pathways for data flow. This offers redundancy and improved fault tolerance, but mesh networks can be more complex to set up and manage.

Optimizing Your Network Performance:

Once your network is up and running, consider these factors to ensure smooth operation:

- **Bandwidth:** This refers to the amount of data that can be transmitted over the network per unit of time. Higher bandwidth allows for faster data transfer speeds. The type of cable used and the capabilities of your network devices determine your network's bandwidth.
- **Latency:** This refers to the time it takes for data to travel from one point to another on the network. Lower latency is desirable for applications like real-time video conferencing and online gaming.
- **Network Congestion:** When multiple devices try to transmit data simultaneously, it can lead to congestion, slowing down the network. Using bandwidth-intensive applications like video streaming can contribute to congestion. Techniques like Quality of Service (QoS) can be implemented on routers to prioritize certain types of traffic and ensure smoother performance for critical applications.

HAMMING CODE

Hamming Code is an error-detection and error-correction technique used in data communication and storage systems to ensure data integrity. It was developed by Richard Hamming in 1950 and is a popular method for single-bit error correction and detection of two-bit errors.

Steps to Encode Using Hamming Code

1. Calculate the number of parity bits (rrr).
2. Place parity bits at 2n2^n2n positions.
3. Fill remaining positions with the data bits.
4. Calculate each parity bit using XOR.

Steps to Decode Using Hamming Code

1. Recalculate parity bits based on received data.
2. Compare recalculated parity with transmitted parity.
3. Identify the error position (if any).
4. Correct the error by flipping the erroneous bit.

Python program implementing the Hamming code for error detection and correction

```
def calculate_parity_bits(data):
    """Calculate the parity bits and return encoded data with parity bits."""
    m = len(data)
    r = 1
    # Calculate the number of parity bits required
    while (2 ** r) < (m + r + 1):
        r += 1
```

```
    # Create an array to hold the data and parity bits
    total_length = m + r
    encoded_data = ['0'] * total_length

    # Place data bits in the correct positions
    j = 0
    for i in range(1, total_length + 1):
        # Positioning parity bits at power of 2 positions
        if (i & (i - 1)) == 0:
            continue
        encoded_data[i - 1] = data[j]
        j += 1

    # Calculate parity bits
    for i in range(r):
        parity_position = 2 ** i
        parity_value = 0
        for j in range(1, total_length + 1):
            if j & parity_position and j != parity_position:
                parity_value ^= int(encoded_data[j - 1])
        encoded_data[parity_position - 1] = str(parity_value)

    return ''.join(encoded_data)


def detect_and_correct_error(received_data):
    """Detect and correct a single-bit error in the received data."""
```

```
    total_length = len(received_data)
    error_position = 0

    # Calculate parity bits to detect error
    for i in range(total_length):
        if (i + 1) & (i + 1):
            parity = 0
            for j in range(total_length):
                if ((j + 1) & (i + 1)) and ((j + 1) != (i + 1)):
                    parity ^= int(received_data[j])
            if parity != int(received_data[i]):
                error_position += (i + 1)

    # Correct the error if detected
    corrected_data = list(received_data)
    if error_position > 0:
        corrected_data[error_position - 1] = '1' if corrected_data[error_position - 1]
== '0' else '0'

    return ''.join(corrected_data), error_position



# Test run
if __name__ == "__main__":
    # Input data stream
    input_data = input("Enter binary data to encode (e.g., 1011): ")
    encoded_data = calculate_parity_bits(input_data)
```

```
    print(f"Encoded data with parity bits: {encoded_data}")

    # Simulate error in the encoded data
    error_index = int(input("Enter position to introduce an error (1-indexed, 0 for no error): "))
    if error_index > 0:
        received_data = list(encoded_data)
        received_data[error_index - 1] = '1' if received_data[error_index - 1] == '0' else '0'
        received_data = ''.join(received_data)
    else:
        received_data = encoded_data

    print(f"Received data: {received_data}")

    # Detect and correct error
    corrected_data, error_position = detect_and_correct_error(received_data)
    if error_position > 0:
        print(f"Error detected and corrected at position: {error_position}")
    else:
        print("No error detected.")
    print(f"Corrected data: {corrected_data}")
```

OUTPUT:

- Input Data: 1011
- Encoded Data: 1011011
- Received Data (with error): 1001011
- Error Position: 3
- Corrected Data: 1011011

sliding window

The **Sliding Window Protocol** is a method of flow control used in data link layer and transport layer protocols to efficiently manage the transmission of data packets (or frames) between two devices (sender and receiver) over a network. It ensures reliable and orderly delivery of data while making optimal use of the available bandwidth.

How Sliding Window Works

1. **Sender's Sliding Window**:
    - The sender keeps a set of sequence numbers corresponding to the frames it is allowed to send. This is the sender's window.
    - Once a frame is acknowledged, the window slides forward.
2. **Receiver's Sliding Window**:
    - The receiver also has a window of sequence numbers corresponding to the frames it expects to receive.
    - It sends acknowledgments for successfully received frames.

Python program to simulate the **Sliding Window Protocol** for flow control at the data link layer.

```
import time
import random

def sliding_window_protocol(window_size, num_frames):
    """Simulate sliding window protocol for frame transmission."""
    sent_frames = 0  # Frames sent by the sender
    acknowledged_frames = 0  # Frames acknowledged by the receiver
```

```
    print(f"Starting Sliding Window Protocol with window size {window_size} and {num_frames} frames.\n")

    while acknowledged_frames < num_frames:
        # Sender sends frames within the window size
        for i in range(window_size):
            if sent_frames < num_frames:
                print(f"Sender: Sending frame {sent_frames}")
                sent_frames += 1
            else:
                break

        # Simulate acknowledgement
        for i in range(window_size):
            if acknowledged_frames < num_frames:
                if random.choice([True, False]):  # Randomly simulate success/failure
                    print(f"Receiver: Frame {acknowledged_frames} received and acknowledged.")
                    acknowledged_frames += 1
                else:
                    print(f"Receiver: Frame {acknowledged_frames} lost. Retransmitting from this frame.")
                    sent_frames = acknowledged_frames  # Resend from the lost frame
                    break

        time.sleep(1)  # Pause to simulate transmission delay

    print("\nAll frames have been successfully transmitted and acknowledged.")
```

```
# Simulation parameters
if __name__ == "__main__":
    window_size = int(input("Enter the sliding window size: "))
    num_frames = int(input("Enter the total number of frames to send: "))
    sliding_window_protocol(window_size, num_frames)
```

OUTPUT:

Starting Sliding Window Protocol with window size 3 and 7 frames.

Sender: Sending frame 0

Sender: Sending frame 1

Sender: Sending frame 2

Receiver: Frame 0 received and acknowledged.

Receiver: Frame 1 received and acknowledged.

Receiver: Frame 2 lost. Retransmitting from this frame.

Sender: Sending frame 2

Sender: Sending frame 3

Sender: Sending frame 4

Receiver: Frame 2 received and acknowledged.

Receiver: Frame 3 received and acknowledged.

Receiver: Frame 4 lost. Retransmitting from this frame.

...

All frames have been successfully transmitted and acknowledged.

TCP Echo Server

```
import socket
# Create a TCP socket
server_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
server_socket.bind(('localhost', 12345))
server_socket.listen(1)

print("TCP Echo Server is running...")

while True:
    conn, addr = server_socket.accept()
    print(f"Connected by {addr}")
    while True:
        data = conn.recv(1024)
        if not data:
            break
        print(f"Received: {data.decode()}")
        conn.sendall(data)  # Echo back
    conn.close()
```

TCP Echo Client

```
import socket
client_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
client_socket.connect(('localhost', 12345))
while True:
    message = input("Enter message to send (type 'exit' to quit): ")
    if message.lower() == 'exit':
```

```
        break
    client_socket.sendall(message.encode())
    response = client_socket.recv(1024)
    print(f"Received from server: {response.decode()}")
client_socket.close()
```

UDP Echo Server

```
import socket

# Create a UDP socket
server_socket = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
server_socket.bind(('localhost', 12345))

print("UDP Echo Server is running...")

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

UDP Echo Client

```
import socket

client_socket = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)

while True:
    message = input("Enter message to send (type 'exit' to quit): ")
```

```
    if message.lower() == 'exit':
        break
    client_socket.sendto(message.encode(), ('localhost', 12345))
    response, _ = client_socket.recvfrom(1024)
    print(f"Received from server: {response.decode()}")

client_socket.close()
```

b) Chat Program Using TCP and UDP

TCP Chat Program:

```
import socket
import threading

def handle_client(conn, addr):
    print(f"Connected by {addr}")
    while True:
        data = conn.recv(1024)
        if not data:
            break
        print(f"{addr} says: {data.decode()}")
        conn.sendall(data)
    conn.close()

server_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
server_socket.bind(('localhost', 12345))
server_socket.listen(5)
```

```
print("TCP Chat Server is running...")

while True:
    conn, addr = server_socket.accept()
    threading.Thread(target=handle_client, args=(conn, addr)).start()
```

Client:

```
import socket

client_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
client_socket.connect(('localhost', 12345))

print("Connected to the chat server!")

while True:
    message = input("You: ")
    if message.lower() == 'exit':
        break
    client_socket.sendall(message.encode())
    response = client_socket.recv(1024)
    print(f"Server: {response.decode()}")

client_socket.close()
```

UDP Chat Program

Server:

```
import socket
```

```
server_socket = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
server_socket.bind(('localhost', 12345))

print("UDP Chat Server is running...")

while True:
    data, addr = server_socket.recvfrom(1024)
    print(f"{addr} says: {data.decode()}")
    response = f"Echo: {data.decode()}"
    server_socket.sendto(response.encode(), addr)
```

Client:

```
import socket

client_socket = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
server_addr = ('localhost', 12345)

print("Connected to the chat server!")

while True:
    message = input("You: ")
    if message.lower() == 'exit':
        break
    client_socket.sendto(message.encode(), server_addr)
    response, _ = client_socket.recvfrom(1024)
```

```
    print(f"Server: {response.decode()}")


client_socket.close()
```

Practical-5

AIM Experiments on Packet capture tool: Wireshark

Packet Sniffer

- Sniffs messages being sent/received from/by your computer
- Store and display the contents of the various protocol fields in the messages
- Passive program
  - never sends packets itself
  - no packets addressed to it
  - receives a copy of all packets (sent/received)

Packet Sniffer Structure Diagnostic Tools

- Tcpdump
  - E.g. tcpdump -enx host 10.129.41.2 -w exe3.out
- Wireshark
  - wireshark -r exe3.out

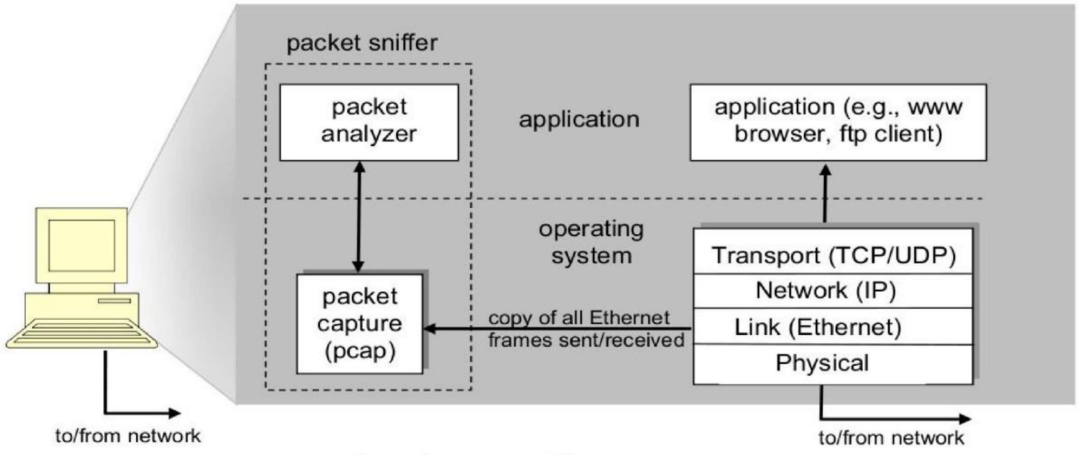


Figure 1: Packet sniffer structure

DESCRIPTION:

WIRESHARK

Wireshark, a network analysis tool formerly known as Ethereal, captures packets in real time and display them in human-readable format. Wireshark includes filters, color coding, and other features that let you dig deep into network traffic and inspect individual packets. You can use Wireshark to inspect a suspicious program's network traffic, analyze the traffic flow on your network, or troubleshoot network problems.

What we can do with Wireshark:

- Capture network traffic
- Decode packet protocols using dissectors
- Define filters – capture and display
- Watch smart statistics
- Analyze problems
- Interactively browse that traffic

Wireshark used for:

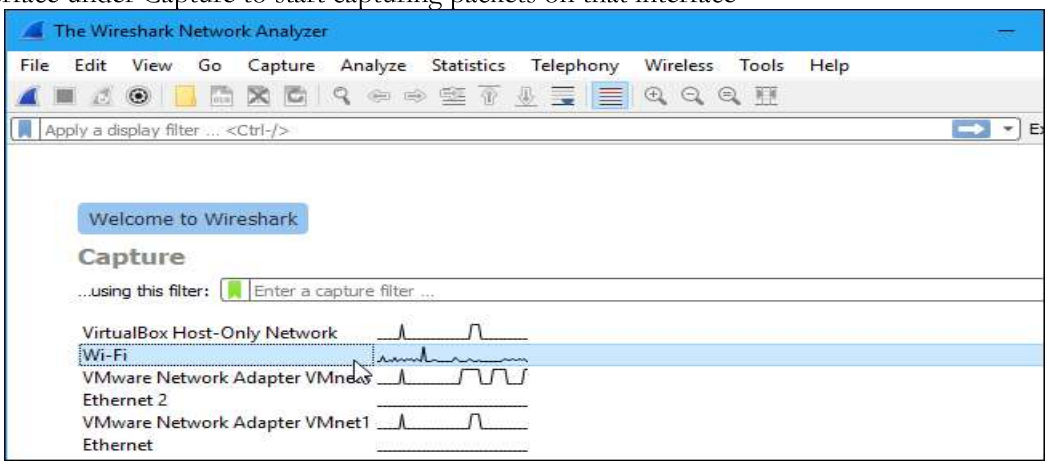
- Network administrators: troubleshoot network problems
- Network security engineers: examine security problems
- Developers: debug protocol implementations
- People: learn **network protocol internals**

Getting Wireshark

Wireshark can be downloaded for Windows or macOS from its official website. For Linux or another UNIX-like system, Wireshark will be found in its package repositories. For Ubuntu, Wireshark will be found in the Ubuntu Software Center.

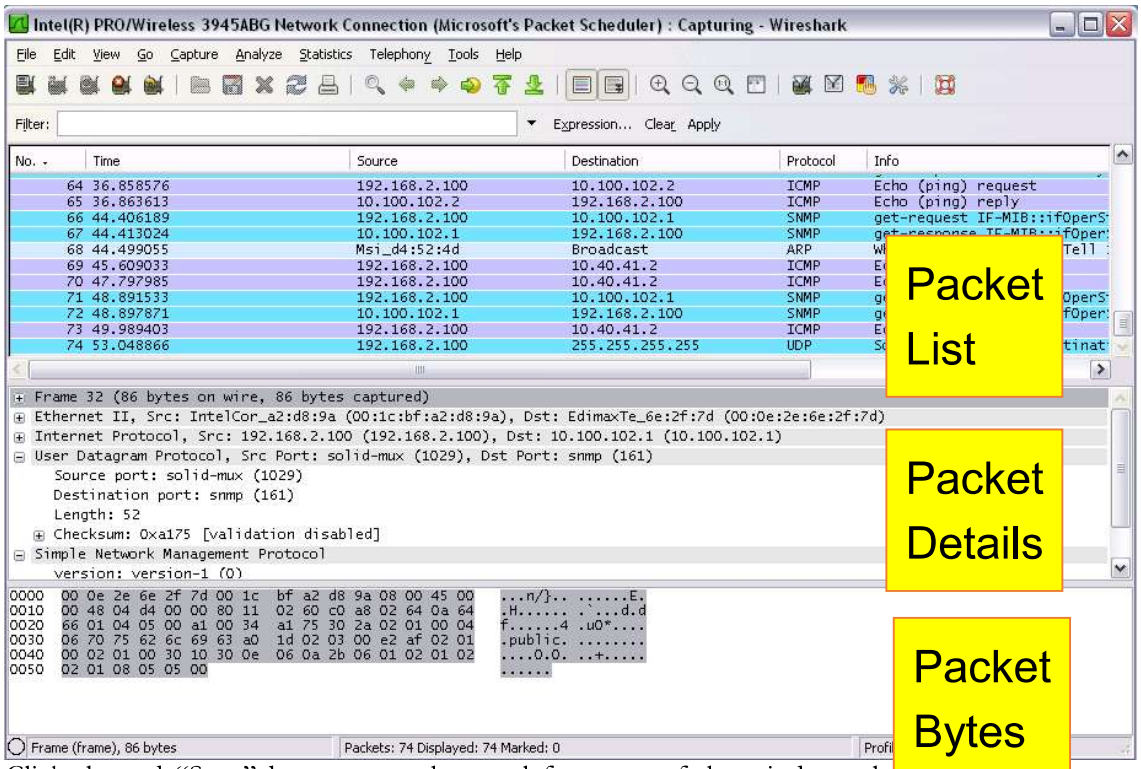
Capturing Packets

After downloading and installing Wireshark, launch it and double-click the name of a network interface under Capture to start capturing packets on that interface



As soon as you click the interface's name, you'll see the packets start to appear in real time. Wireshark captures each packet sent to or from your system.

If you have promiscuous mode enabled—it's enabled by default—you'll also see all the other packets on the network instead of only packets addressed to your network adapter. To check if promiscuous mode is enabled, click Capture > Options and verify the "Enable promiscuous mode on all interfaces" checkbox is activated at the bottom of this window.



Click the red "Stop" button near the top left corner of the window when you want to stop capturing traffic.

The "Packet List" Pane

The packet list pane displays all the packets in the current capture file. The "Packet List" pane Each line in the packet list corresponds to one packet in the capture file. If you select a line in this pane, more details will be displayed in the "Packet Details" and "Packet Bytes" panes.

The "Packet Details" Pane

The packet details pane shows the current packet (selected in the "Packet List" pane) in a more detailed form. This pane shows the protocols and protocol fields of the packet selected in the "Packet List" pane. The protocols and fields of the packet shown in a tree which can be expanded and collapsed.

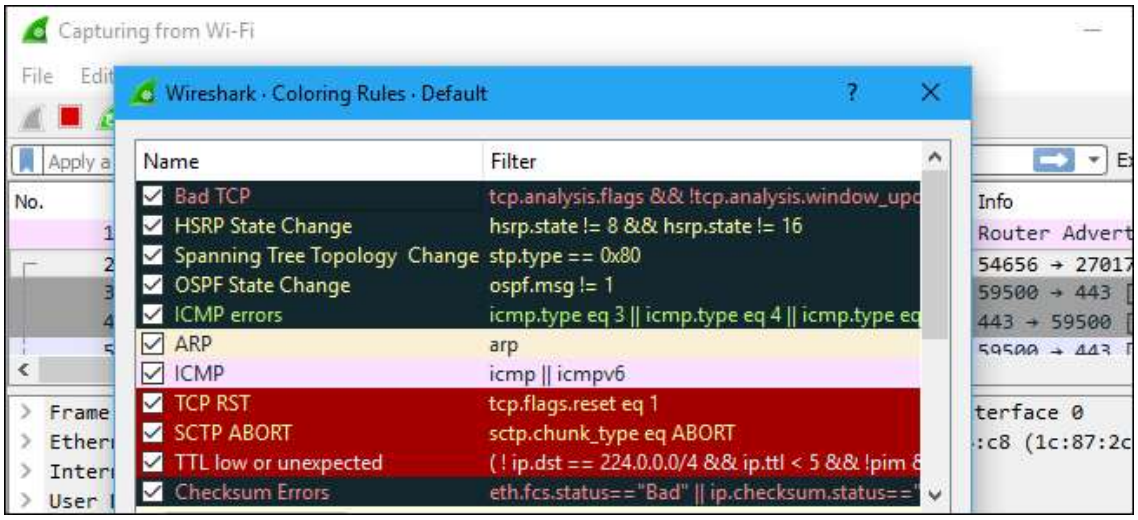
The "Packet Bytes" Pane

The packet bytes pane shows the data of the current packet (selected in the "Packet List" pane) in a hexdump style.

Color Coding

You'll probably see packets highlighted in a variety of different colors. Wireshark uses colors to help you identify the types of traffic at a glance. By default, light purple is TCP traffic, light blue is UDP traffic, and black identifies packets with errors—for example, they could have been delivered out of order.

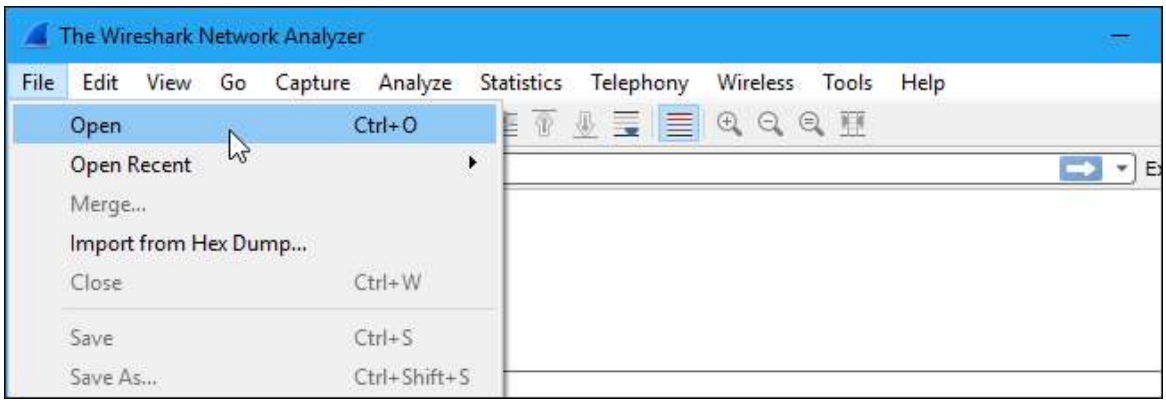
To view exactly what the color codes mean, click View > Coloring Rules. You can also customize and modify the coloring rules from here, if you like.



Sample Captures

If there's nothing interesting on your own network to inspect, Wireshark's wiki has you covered. The wiki contains a page of sample capture files that you can load and inspect. Click File > Open in Wireshark and browse for your downloaded file to open one.

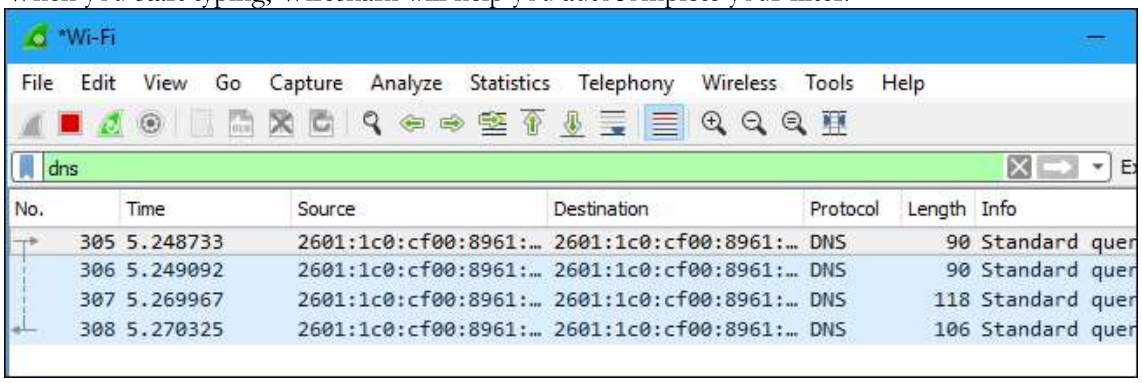
You can also save your own captures in Wireshark and open them later. Click File > Save to save your captured packets.



Filtering Packets

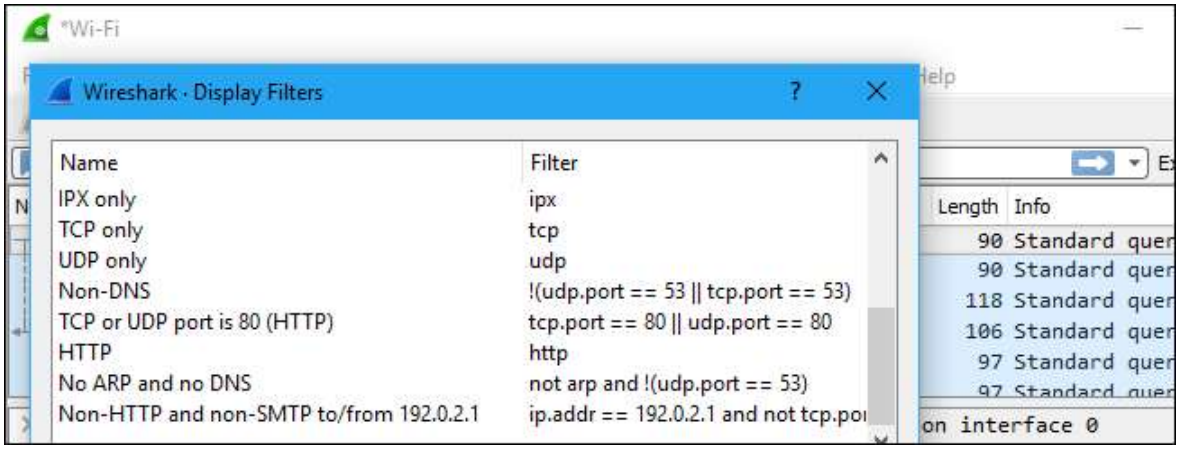
If you're trying to inspect something specific, such as the traffic a program sends when phoning home, it helps to close down all other applications using the network so you can narrow down the traffic. Still, you'll likely have a large amount of packets to sift through. That's where Wireshark's filters come in.

The most basic way to apply a filter is by typing it into the filter box at the top of the window and clicking Apply (or pressing Enter). For example, type "dns" and you'll see only DNS packets. When you start typing, Wireshark will help you autocomplete your filter.

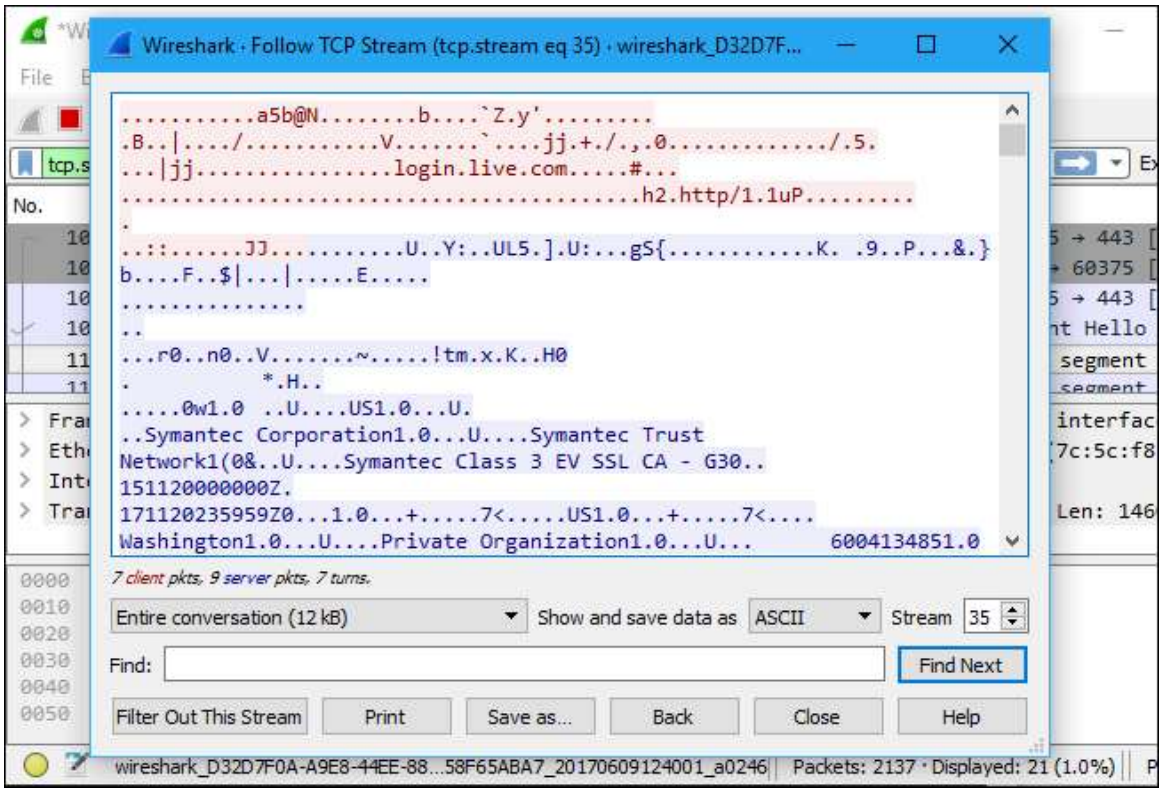


You can also click Analyze > Display Filters to choose a filter from among the default filters included in Wireshark. From here, you can add your own custom filters and save them to easily access them in the future.

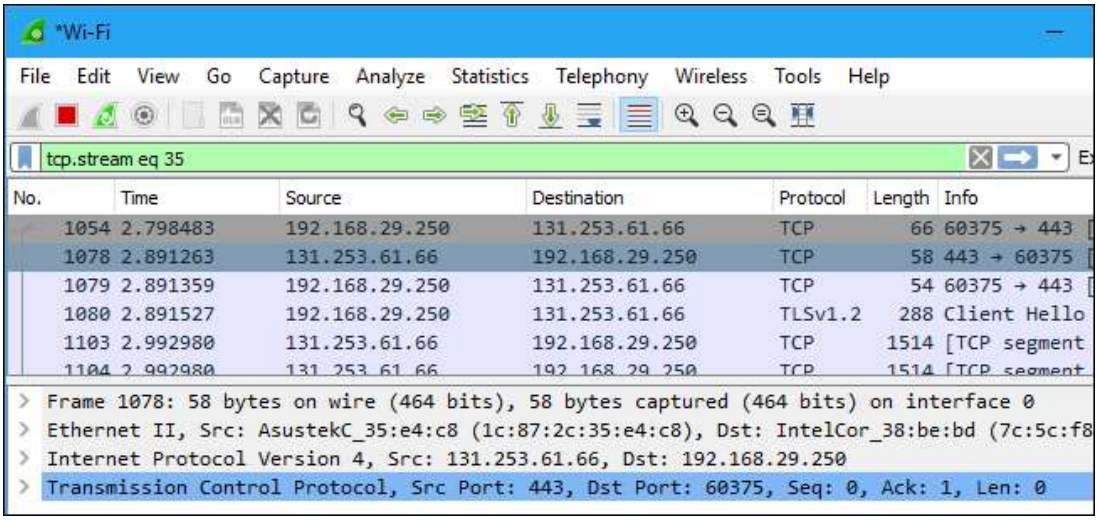
For more information on Wireshark's display filtering language, read the Building display filter expressions page in the official Wireshark documentation.



Another interesting thing you can do is right-click a packet and select Follow > TCP Stream. You'll see the full TCP conversation between the client and the server. You can also click other protocols in the Follow menu to see the full conversations for other protocols, if applicable.

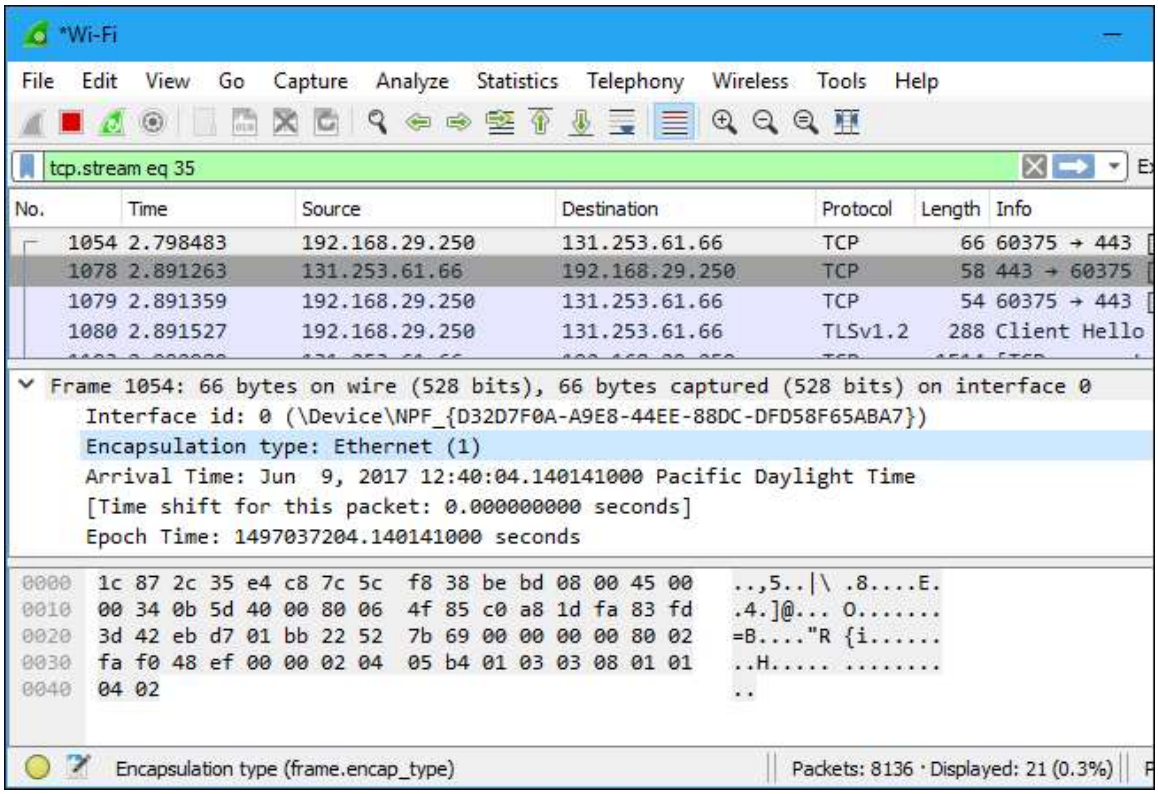


Close the window and you'll find a filter has been applied automatically. Wireshark is showing you the packets that make up the conversation.

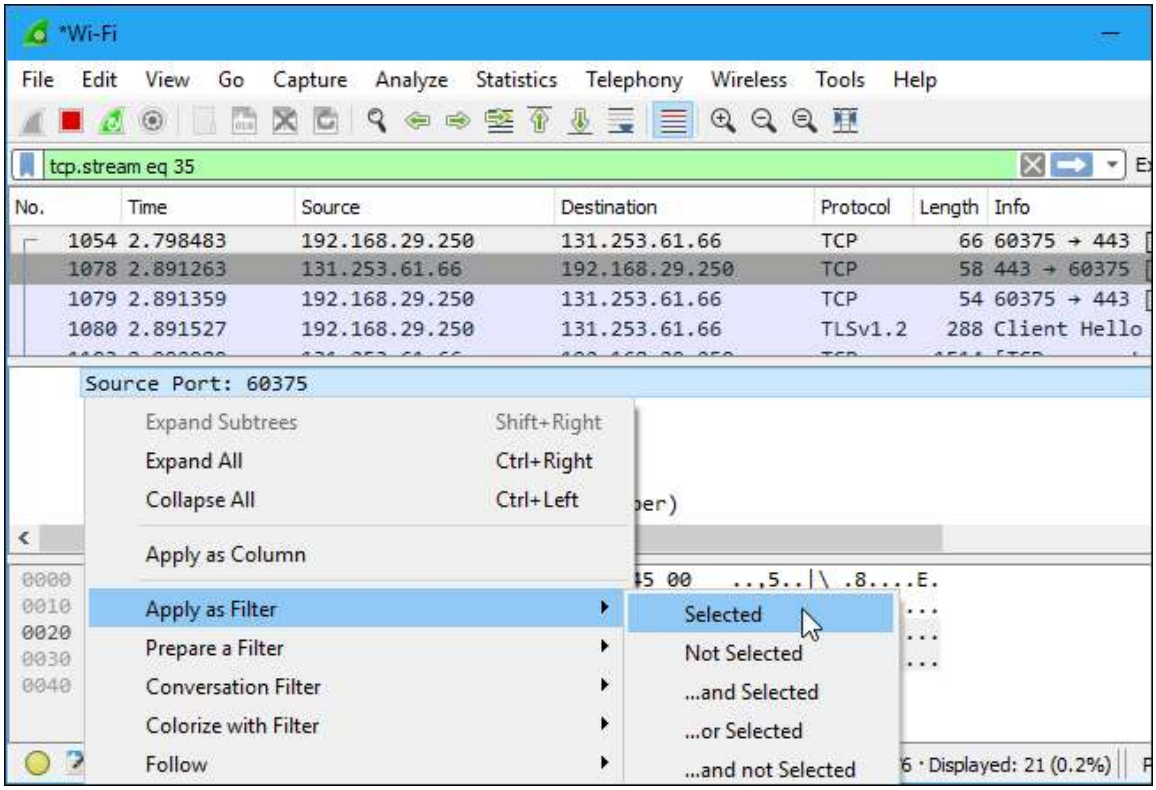


Inspecting Packets

Click a packet to select it and you can dig down to view its details.

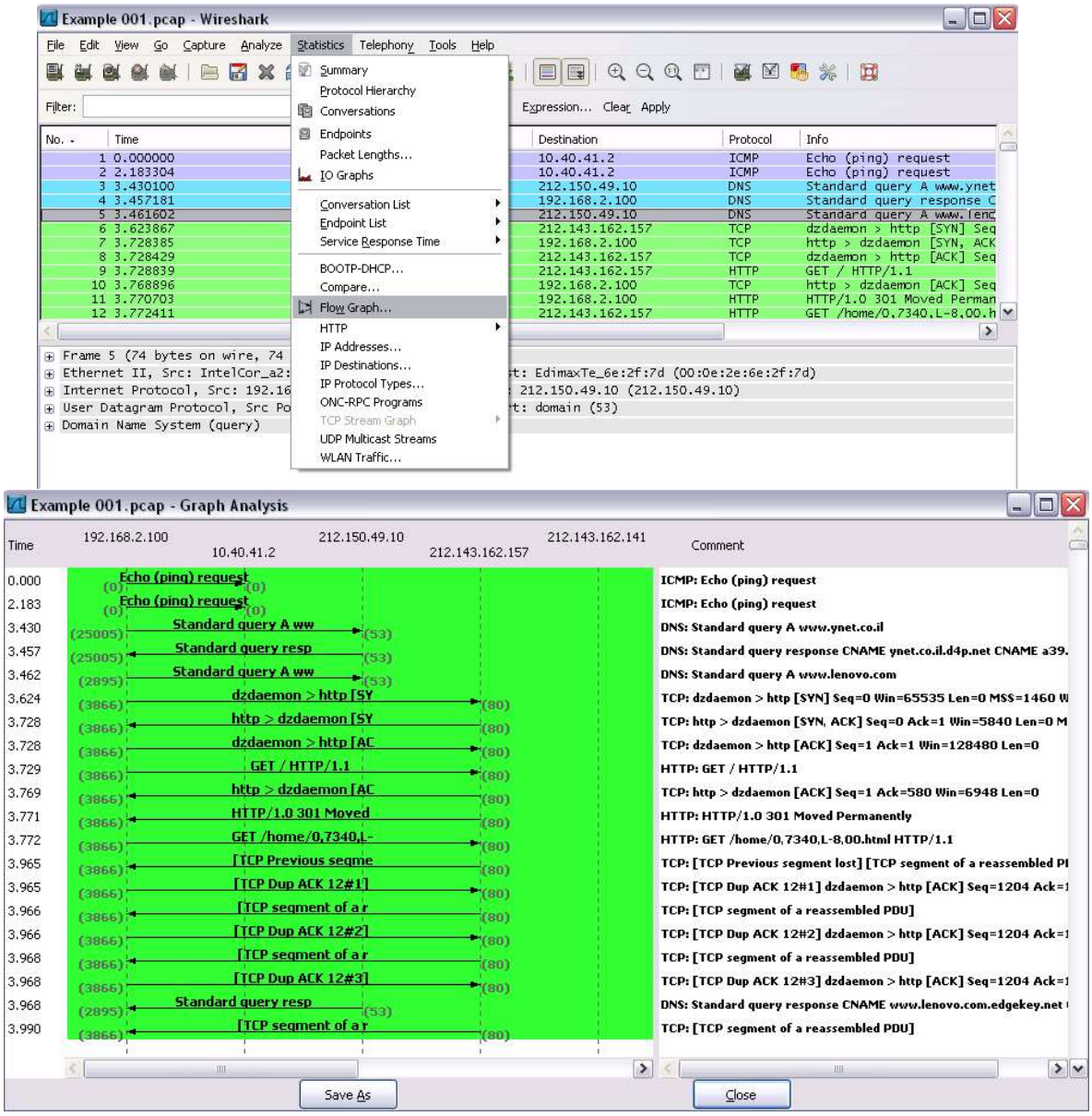


You can also create filters from here — just right-click one of the details and use the Apply as Filter submenu to create a filter based on it.



Wireshark is an extremely powerful tool, and this tutorial is just scratching the surface of what you can do with it. Professionals use it to debug network protocol implementations, examine security problems and inspect network protocol internals.

Flow Graph: Gives a better understanding of what we see.



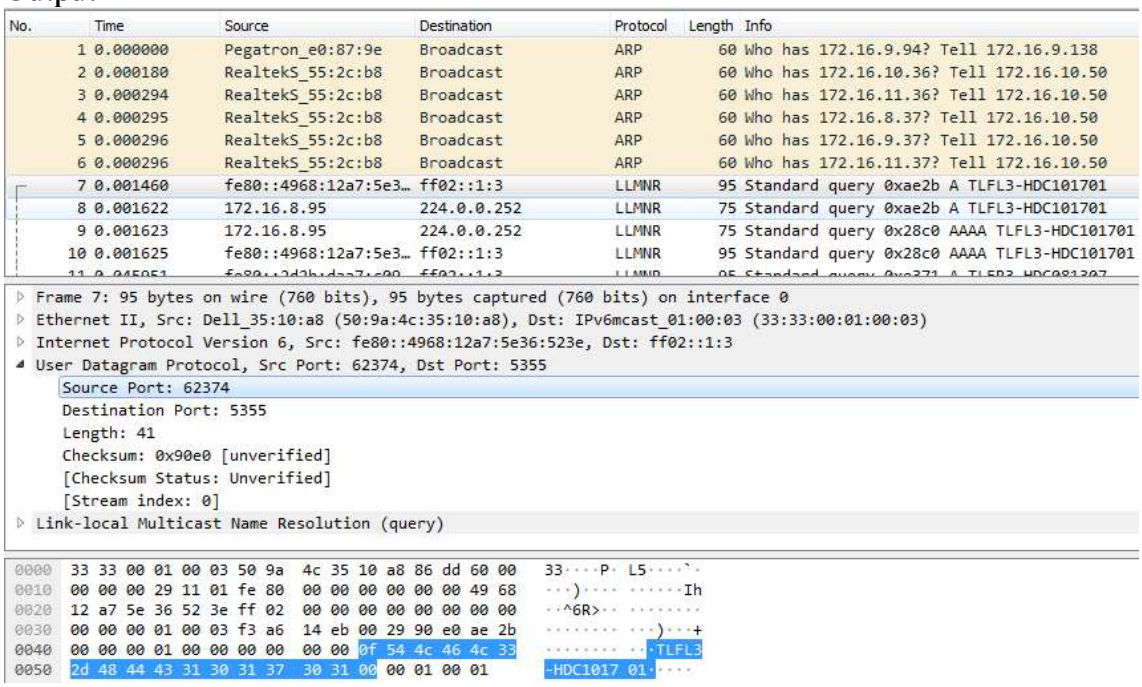
CAPTURING AND ANALYSING PACKETS USING WIRESHARK TOOL

To filter, capture, view, packets in Wireshark Tool.
Capture 100 packets from the Ethernet: IEEE 802.3 LAN Interface and save it.

Procedure

- Select Local Area Connection in Wireshark.
- Go to capture → option
- Select stop capture automatically after 100 packets.
- Then click Start capture.
- Save the packets.

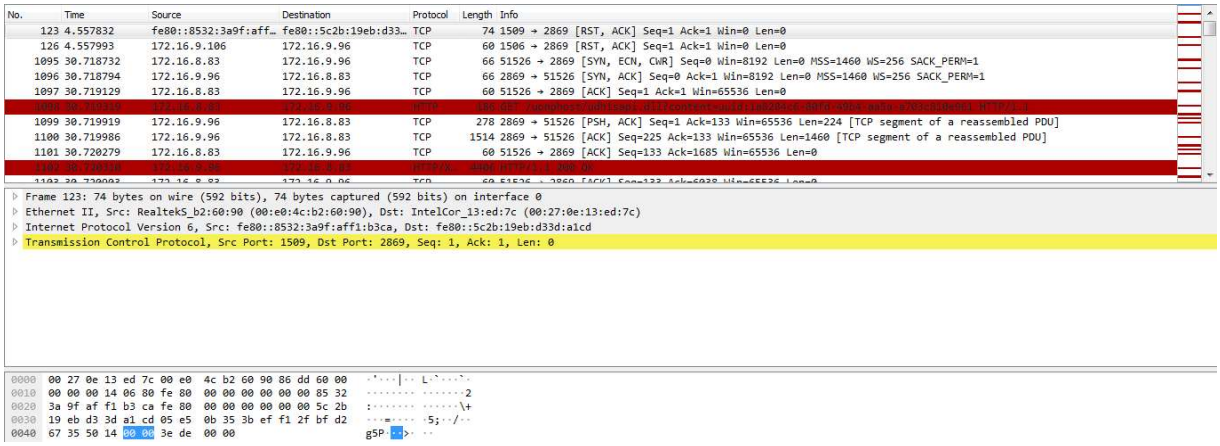
Output



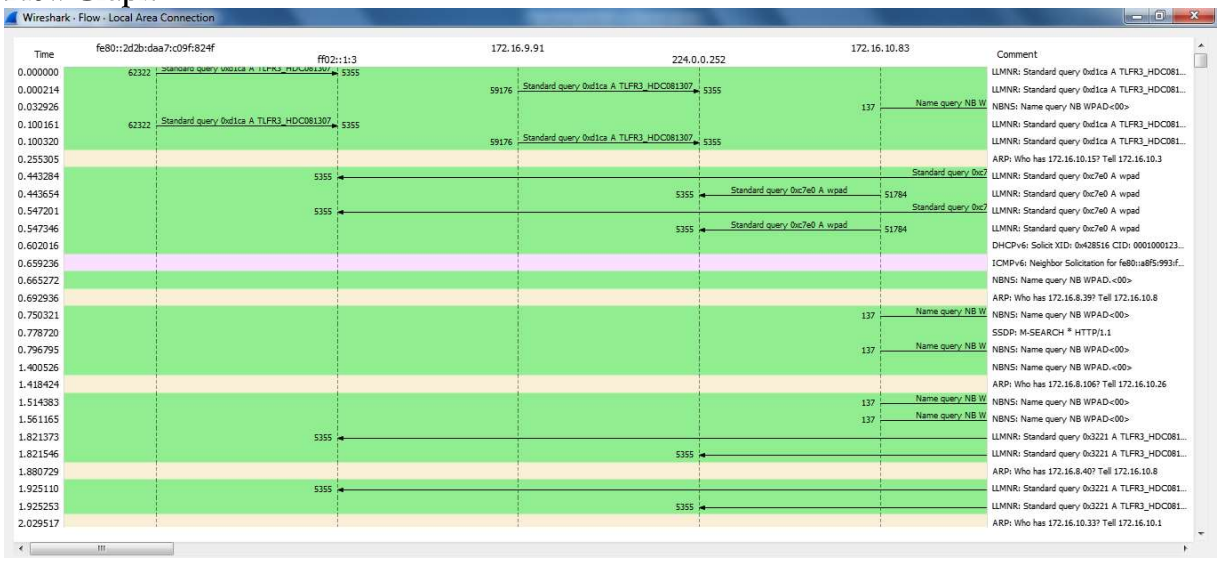
1. **Create a Filter to display only TCP/UDP packets, inspect the packets and provide the flow graph**

Procedure

- Select Local Area Connection in Wireshark.
- Go to capture → option
- Select stop capture automatically after 100 packets.
- Then click Start capture.
- Search TCP packets in search bar.
- To see flow graph click Statistics→Flow graph.
- Save the packets.



Flow Graph

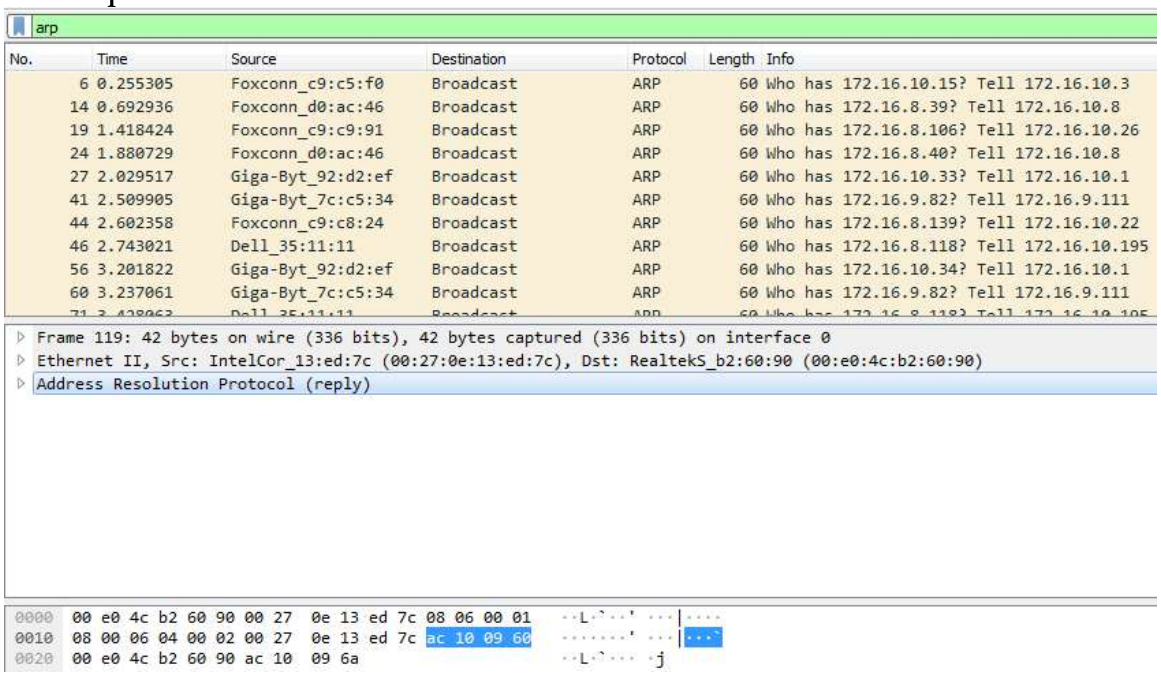


2. Create a Filter to display only ARP packets and inspect the packets.

Procedure

- Go to capture → option
- Select stop capture automatically after 100 packets.
- Then click Start capture.
- Search ARP packets in search bar.
- Save the packets.

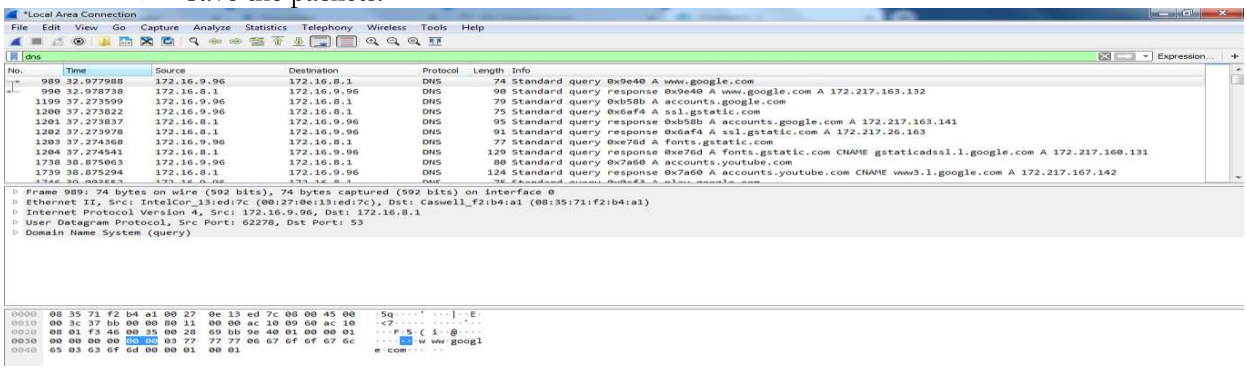
Output

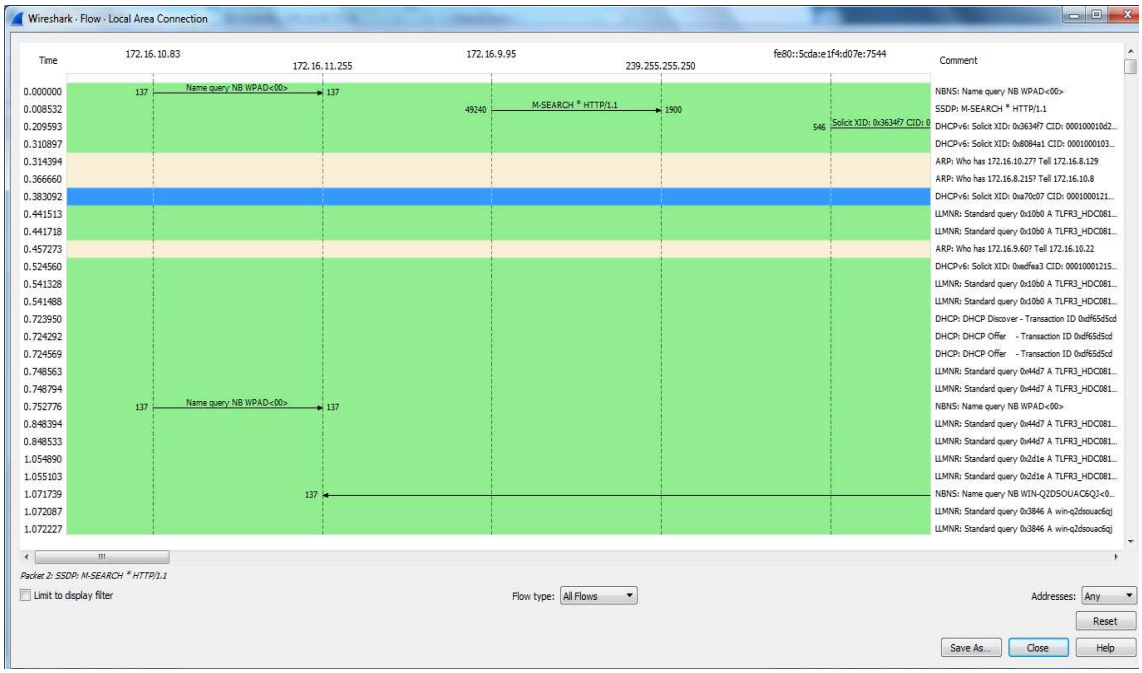


3. Create a Filter to display only DNS packets and provide the flow graph.

Procedure

- Go to capture → option
- Select stop capture automatically after 100 packets.
- Then click Start capture.
- Search DNS packets in search bar.
- To see flow graph click Statistics→Flow graph.
- Save the packets.

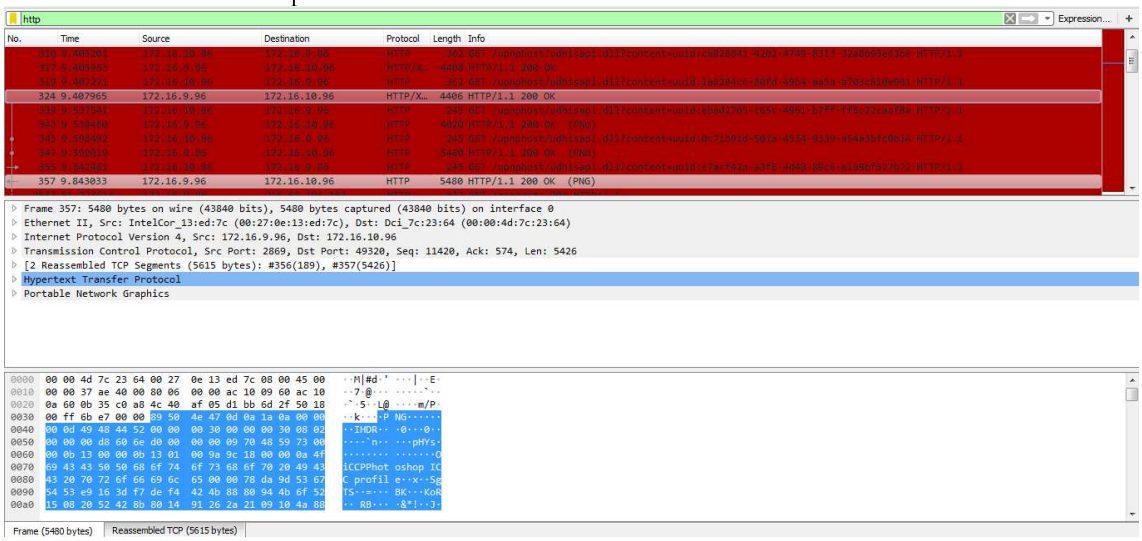




4. Create a Filter to display only HTTP packets and inspect the packets

Procedure

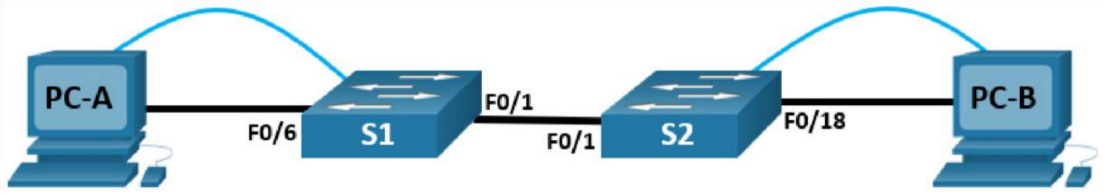
- Select Local Area Connection in Wireshark.
- Go to capture → option
- Select stop capture automatically after 100 packets.
- Then click Start capture.
- Search HTTP packets in search bar.
- Save the packets.



Practical-8

AIM: - a) Simulate Virtual LAN configuration using CISCO Packet Tracer Simulation.

Packet Tracer - Configure VLANs and Trunking - Physical Mode Topology



Addressing Table

Device	Interface	IP Address	Subnet Mask	Default Gateway
S1	VLAN 1	192.168.1.11	255.255.255.0	N/A
S2	VLAN 1	192.168.1.12	255.255.255.0	N/A
PC-A	NIC	192.168.10.3	255.255.255.0	192.168.10.1
PC-B	NIC	192.168.10.4	255.255.255.0	192.168.10.1

Blank Line - no additional information

Objectives

Part 1: Build the Network and Configure Basic Device Settings
Part 2: Create VLANs and Assign Switch Ports
Part 3: Maintain VLAN Port Assignments and the VLAN Database Part 4: Configure an 802.1Q Trunk between the Switches

Background / Scenario

Modern switches use virtual local-area networks (VLANs) to improve network performance by separating large Layer 2 broadcast domains into smaller ones. VLANs can also be used as a security measure by controlling which hosts can communicate. In general, VLANs make it easier to design a network to support the goals of an organization.

VLAN trunks are used to span VLANs across multiple devices. Trunks allow the traffic from multiple VLANS to travel over a single link, while keeping the VLAN identification and segmentation intact.

In this Packet Tracer Physical Mode (PTPM) activity, you will create VLANs on both switches in the topology, assign VLANs to switch access ports, and verify that VLANs are working as expected. You will then create a VLAN trunk between the two switches to allow hosts in the same VLAN to communicate through the trunk, regardless of which switch to which the host is attached.

Instructions

Part 1: Build the Network and Configure Basic Device Settings

Step 1: Build the network as shown in the topology.

Attach the devices as shown in the topology diagram, and cable as necessary. a. Click and drag both switch **S1** and **S2** to the **Rack**.

b. Click and drag both **PC-A** and **PC-B** to the **Table** and use the power button to turn them on.

c. Provide network connectivity by connecting **Copper Straight-through** cables, as shown in the topology.
d. Connect **Console Cable** from device **PC-A** to **S1** and from device **PC-B** to **S2**.

Step 2: Configure basic settings for each switch.

a. From the **Desktop Tab** on each PC, use the **Terminal** to console into each switch and enable privileged EXEC mode.
   Open configuration window
b. Enter configuration mode.
c. Assign a device name to each switch.
d. Assign **class** as the privileged EXEC encrypted password.
e. Assign **cisco** as the console password and enable login.
f. Assign **cisco** as the vty password and enable login.
g. Encrypt the plaintext passwords.
h. Create a banner that warns anyone accessing the device that unauthorized access is prohibited.
i. Configure the IP address listed in the Addressing Table for VLAN 1 on the switch.
   Note: The VLAN 1 address is not grade because you will remove it later in the activity. However, you will need VLAN 1 to test connectivity later in this Part.
j. Shut down all interfaces that will not be used.
k. Set the clock on each switch.
   Note: The clock setting cannot be graded in Packet Tracer.
l. Save the running configuration to the startup configuration file.
   Close configuration window

Step 3: Configure PC hosts.

From the **Desktop** tab on each **PC**, click IP Configuration and enter the addressing information as displayed in the Addressing Table.

Step 4: Test connectivity.

Test network connectivity by attempting to ping between each of the cabled devices.

Questions:

Can PC-A ping PC-B?
Can PC-A ping S1?
Can PC-B ping S2?
Can S1 ping S2?

Close configuration window

Part 2: Create VLANs and Assign Switch Ports

In Part 2, you will create Management, Operations, Parking Lot, and Native VLANs on both switches. You will then assign the VLANs to the appropriate interface. The **show vlan** command is used to verify your configuration settings.

Step 1: Create VLANs on the switches.

From the **Desktop Tab** on each **PC**, use Terminal to continue configuring both network switches.

Open configuration window

a. Create the VLANs on **S1**.

```
S1(config)# vlan 10
S1(config-vlan)# name Operations
S1(config-vlan)# vlan 20
S1(config-vlan)# name Parking_Lot
S1(config-vlan)# vlan 99
S1(config-vlan)# name Management
S1(config-vlan)# vlan 1000
S1(config-vlan)# name Native
S1(config-vlan)# end
```

b. Create the same VLANs on S2.

c. Issue the **show vlan brief** command to view the list of VLANs on **S1**.

```
S1# show vlan brief

VLAN Name                              Status    Ports
---- --------------------------------- --------- -------------------------------
1    default                           active    Fa0/1, Fa0/2, Fa0/3, Fa0/4
                                                 Fa0/5, Fa0/6, Fa0/7, Fa0/8
                                                 Fa0/9, Fa0/10, Fa0/11, Fa0/12
                                                 Fa0/13, Fa0/14, Fa0/15, Fa0/16
                                                 Fa0/17, Fa0/18, Fa0/19, Fa0/20
                                                 Fa0/21, Fa0/22, Fa0/23, Fa0/24
                                                 Gi0/1, Gi0/2
10   Operations                        active
20   Parking_Lot                       active
99   Management                        active
1000 Native                            active
1002 fddi-default                      active
1003 token-ring-default                active
1004 fddinet-default                   active
1005 trnet-default                     active
```

Questions:

What is the default VLAN?

What ports are assigned to the default VLAN?

Step 2: Assign VLANs to the correct switch interfaces.

a. Assign VLANs to the interfaces on **S1**.
   1) Assign PC-A to the Operation VLAN.
      S1(config)# **interface f0/6**
      S1(config-if)# **switchport mode access**
      S1(config-if)# **switchport access vlan 10**
   2) From VLAN 1, remove the management IP address and configure it on VLAN 99.
      S1(config)# **interface vlan 1**
      S1(config-if)# **no ip address**
      S1(config-if)# **interface vlan 99**
      S1(config-if)# **ip address 192.168.1.11 255.255.255.0**
      S1(config-if)# **end**

b. Issue the **show vlan brief** command and verify that the VLANs are assigned to the correct interfaces. c. Issue the **show ip interface brief** command.
   Question:
   What is the status of VLAN 99? Explain.

d. Assign **PC-B** to the Operations VLAN on **S2**.
e. From VLAN 1, remove the management IP address and configure it on VLAN 99 according to the Addressing Table.
f. Use the **show vlan brief** command to verify that the VLANs are assigned to the correct interfaces.
   Questions:
   Is S1 able to ping S2? Explain.
   Is PC-A able to ping PC-B? Explain.

Part 3: Maintain VLAN Port Assignments and the VLAN Database

In Part 3, you will change port VLAN assignments and remove VLANs from the VLAN database.

Step 1: Assign a VLAN to multiple interfaces.

From the **Desktop Tab** on each **PC**, use **Terminal** to continue configuring both network switches.

Open configuration window

a. On S1, assign interfaces F0/11 – 24 to VLAN99.
   S1(config)# **interface range f0/11-24**
   S1(config-if-range)# **switchport mode access**
   S1(config-if-range)# **switchport access vlan 99**
   S1(config-if-range)# **end**
b. Issue the **show vlan brief** command to verify VLAN assignments.
c. Reassign F0/11 and F0/21 to VLAN 10.
d. Verify that VLAN assignments are correct.

Step 2: Remove a VLAN assignment from an interface.

a. Use the **no switchport access vlan** command to remove the VLAN 99 assignment to F0/24.
   S1(config)# **interface f0/24**
   S1(config-if)# **no switchport access vlan**
   S1(config-if)# **end**
b. Verify that the VLAN change was made.
   Question:

Which VLAN is F0/24 now associated with?

Step 3: Remove a VLAN ID from the VLAN database.

a. Add VLAN 30 to interface F0/24 without issuing the global VLAN command.
S1(config)# **interface f0/24**
S1(config-if)# **switchport access vlan 30**
% Access VLAN does not exist. Creating vlan 30
Note: Current switch technology no longer requires that the **vlan** command be issued to add a VLAN to the database. By assigning an unknown VLAN to a port, the VLAN will be created and added to the VLAN database.

b. Verify that the new VLAN is displayed in the VLAN table.
Question:
What is the default name of VLAN 30?

c. Use the **no vlan 30** command to remove VLAN 30 from the VLAN database.
S1(config)# **no vlan 30**
S1(config)# **end**

d. Issue the **show vlan brief** command. F0/24 was assigned to VLAN 30.
Question:
After deleting VLAN 30 from the VLAN database, why is F0/24 no longer displayed in the output of the **show vlan brief** command? What VLAN is port F0/24 now assigned to? What happens to the traffic destined to the host that is attached to F0/24?

e. On interface F0/24, issue the **no switchport access vlan** command.

f. Issue the **show vlan brief** command to determine the VLAN assignment for F0/24.
Questions:
To which VLAN is F0/24 assigned?

Note: Before removing a VLAN from the database, it is recommended that you reassign all the ports assigned to that VLAN.

Why should you reassign a port to another VLAN before removing the VLAN from the VLAN database?

Close configuration window.

Part 4: Configure an 802.1Q Trunk Between the Switches

In Part 4, you will configure interface F0/1 to use the Dynamic Trunking Protocol (DTP) to allow it to negotiate the trunk mode. After this has been accomplished and verified, you will disable DTP on interface F0/1 and manually configure it as a trunk.

Step 1: Use DTP to initiate trunking on F0/1.

The default DTP mode of a 2960 switch port is dynamic auto. This allows the interface to convert the link to a trunk if the neighboring interface is set to trunk or dynamic desirable mode.

Open configuration window

a. On **S1**, set F0/1 to negotiate trunk mode.

S1(config)# **interface f0/1**
S1(config-if)# **switchport mode dynamic desirable**

Sep 19 02:51:47.257: %LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet0/1, changed state to up
Sep 19 02:51:47.291: %LINEPROTO-5-UPDOWN: Line protocol on Interface Vlan99, changed state to up
You should also receive link status messages on S2.
S2#
Sep 19 02:42:19.424: %LINK-3-UPDOWN: Interface FastEthernet0/1, changed state to up
Sep 19 02:42:21.454: %LINEPROTO-5-UPDOWN: Line protocol on Interface Vlan99, changed state to up
Sep 19 02:42:22.419: %LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet0/1, changed state to up

b. On **S1** and **S2**, issue the **show vlan brief** command. Interface F0/1 is no longer assigned to VLAN 1. Trunked interfaces are not listed in the VLAN table.
c. Issue the **show interfaces trunk** command to view trunked interfaces. Notice that the mode on **S1** is set to desirable, and the mode on **S2** is set to auto.
S1# **show interfaces trunk**

S2# **show interfaces trunk**
Note: By default, all VLANs are allowed on a trunk. The **switchport trunk** command allows you to control what VLANs have access to the trunk. For this activity, keep the default settings. This allows all VLANs to traverse F0/1.

Close configuration window

d. Verify that VLAN traffic is traveling over trunk interface F0/1.
Questions:
Can S1 ping S2?
Can PC-A ping PC-B?
Can PC-A ping S1?
Can PC-B ping S2?

Step 2: Manually configure trunk interface F0/1.

The **switchport mode trunk** command is used to manually configure a port as a trunk. This command should be issued on both ends of the link.

a. On interface F0/1, change the switchport mode to force trunking. Make sure to do this on both switches.

Open configuration window

S1(config)# **interface f0/1**
S1(config-if)# **switchport mode trunk**

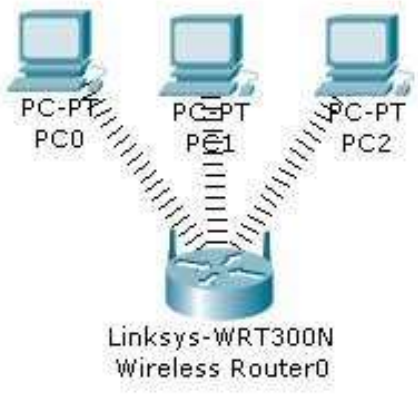
b. Issue the **show interfaces trunk** command to view the trunk mode. Notice that the mode changed from **desirable** to **on**.
S1# **show interfaces trunk**
c. Modify the trunk configuration on both switches by changing the native VLAN from VLAN 1 to VLAN 1000.
S1(config)# **interface f0/1**
S1(config-if)# **switchport trunk native vlan 1000**
d. Issue the **show interfaces trunk** command to view the trunk. Notice the Native VLAN information is updated.
S2# **show interfaces trunk**
Questions:
Why might you want to manually configure an interface to trunk mode instead of using DTP?
Why might you want to change the native VLAN on a trunk?

Close configuration window

Practical-8

AIM:-b) Configuration of Wireless LAN using CISCO Packet Tracer.

Design a topology with three PCs connected from Linksys Wireless routers.



Perform following configuration:-

- Configure Static IP on PC and Wireless Router
- Set SSID to MotherNetwork
- Set IP address of router to 192.168.0.1, PC0 to 192.168.0.2, PC1 to 192.168.0.3 and PC2 to 192.168.0.4.
- Secure your network by configuring WAP key on Router
- Connect PC by using WAP key

To complete these tasks follow these step by step instructions:-

Step1:- Click on wireless router,

- Select Administration tab from top Menu, set username and password to admin and click on Save Setting.

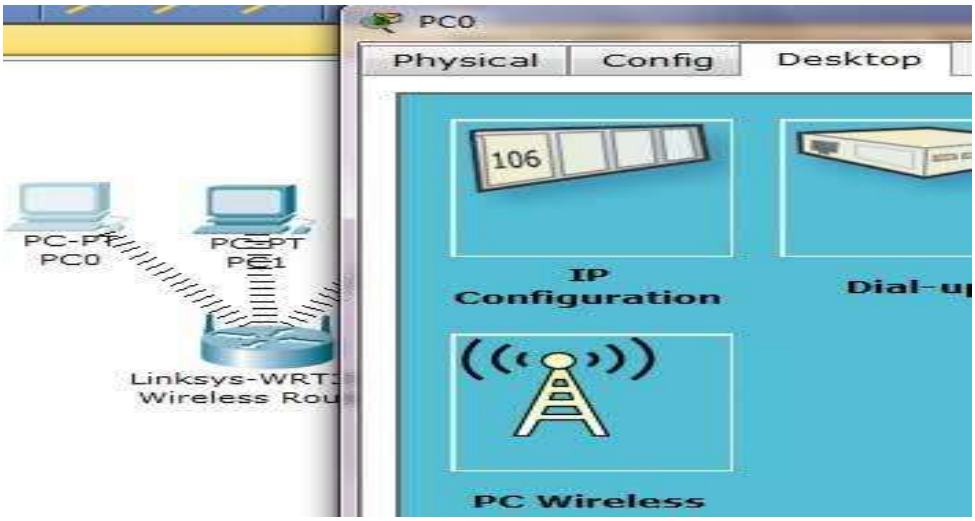
- Next click on wireless tab and set default SSID to MotherNetwork.
- Now Select wireless security and change Security Mode to WEP

- Set Key1 to 0123456789

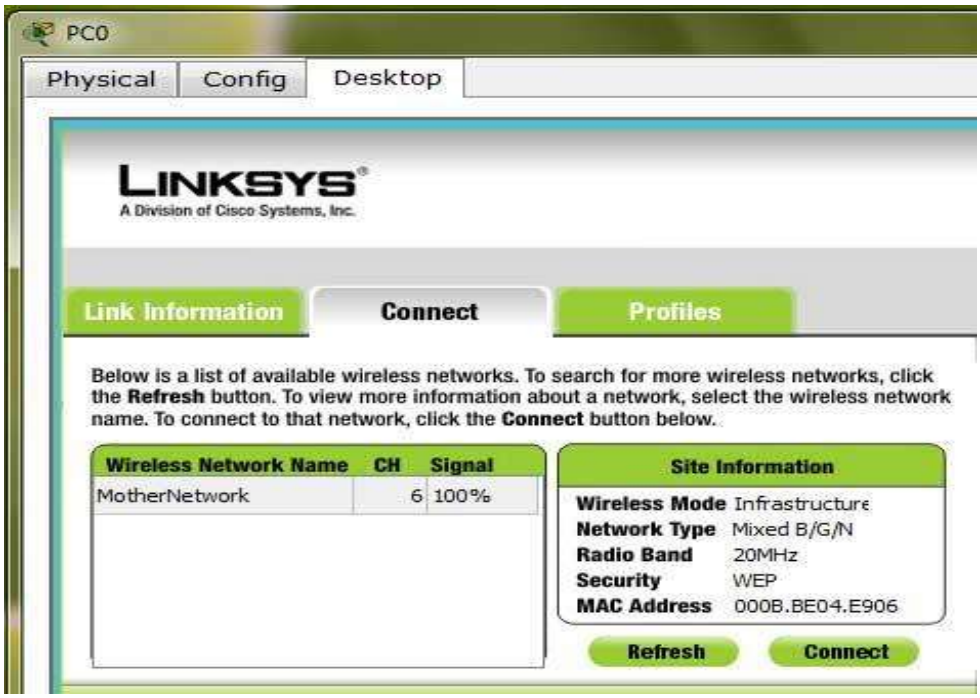
- Again go in the end of page and Click on Save Setting
- Now we have completed all given task on Wireless router. Now configure the static IP on all three PC's
- Double click on pc select Desktop tab click on IP configuration select Static IP and set IP as given below

PC	IP	Subnet Mask	Default Gateway
PC0	192.168.0.2	255.255.255.0	192.168.0.1
PC1	192.168.0.3	255.255.255.0	192.168.0.1
PC2	192.168.0.4	255.255.255.0	192.168.0.1

- Now it's time to connect PC's from Wireless router. To do so click PC select Desktop click on PC Wireless

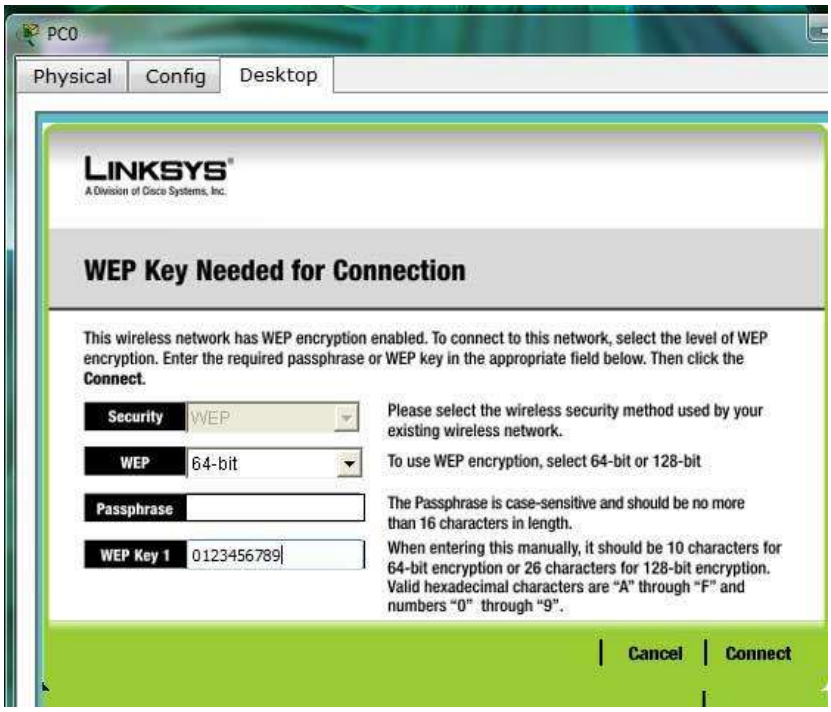


- Click on connect tab and click on Refresh button



As you can see in image that Wireless device is accessing MotherNetwork on CH 6 and signal strength is 100%. In left side you can see that WEP security is configured in network. Click on connect button to connect MotherNetwork

- It will ask for WAP key insert 0123456789 and click connect



It will connect you with wireless router.

As you can see in image below that system is connected. And PCI card is active.

- Repeat same process on PC1 and PC2.

Practical-9

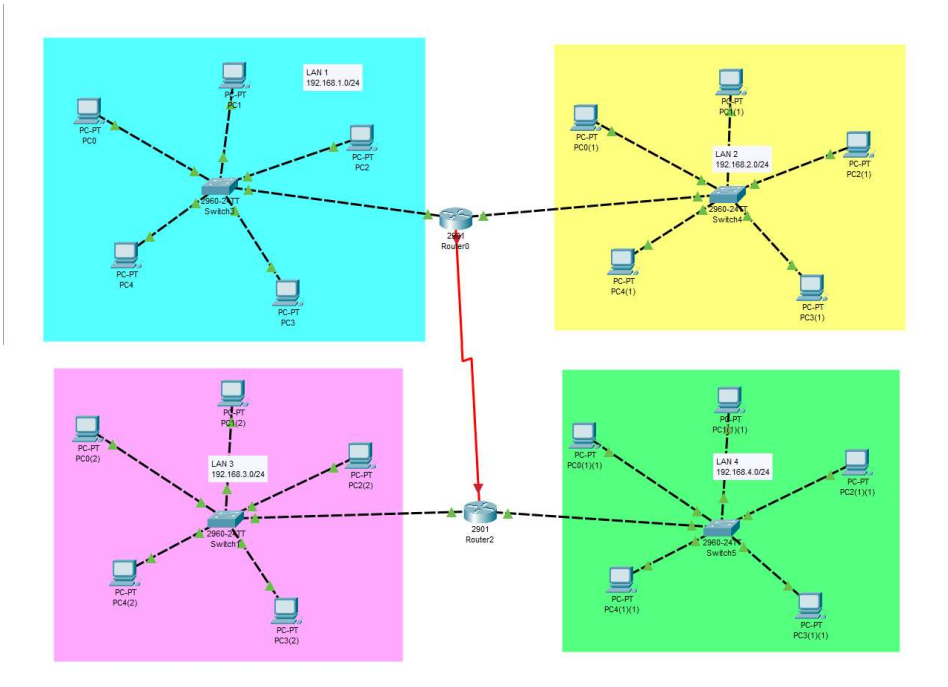
AIM:-Implementation of SUBNETTING in CISCO PACKET TRACER simulator.

Classless IP subnetting is a technique that allows for more efficient use of IP addresses by allowing for subnet masks that are not just the default masks for each IP class. This means that we can divide our IP address space into smaller subnets, which can be useful when we have a limited number of IP addresses but need to create multiple networks.

CREATING A NETWORK TOPOLOGY:
The first step in implementing classless IP subnetting is to create a network topology in Packet Tracer. To create a network topology in Packet Tracer, select the "New" button in the top left corner, then select "Network" and "Generic". This will create a blank network topology that we can use to add devices.

ADDING THE DEVICES:
Once we have created our network topology, we can add devices to it. Here, we will be adding routers, witches, and PCs. To add a device, select the device from the bottom left corner and drag it onto the network topology. Then, connect the devices by dragging a cable from one device's port to another device's port.

SUBNETTING:
To subnet the network address of 192.168.1.0/24 to provide enough space for at least 5 addresses for end devices, the switch, and the router, we can use a /27 subnet mask. This will give us 8 subnets with 30 host addresses each.



The IP addressing for the network shown in the topology can be as follows:

- Router R1:
- GigabitEthernet0/0: 192.168.1.1
- GigabitEthernet0/1: 192.168.2.1
- Switch S1:
- FastEthernet0/1: 192.168.1.0/27
- PC1: 192.168.1.11
- PC2: 192.168.1.12
- PC3: 192.168.1.13
- PC4: 192.168.1.14
- PC5: 192.168.1.15
- FastEthernet0/2: 192.168.2.0/27
- PC1: 192.168.2.11
- PC2: 192.168.2.12
- PC3: 192.168.2.13
- PC4: 192.168.2.14
- PC5: 192.168.2.15
- Router R2:
- FastEthernet0/0: 192.168.3.1
- FastEthernet0/1: 192.168.4.1
- Switch S2:
- FastEthernet0/1: 192.168.3.0/27
- PC1: 192.168.3.11
- PC2: 192.168.3.12
- PC3: 192.168.3.13
- PC4: 192.168.3.14
- PC5: 192.168.3.15
- FastEthernet0/2: 192.168.4.0/27
- PC1: 192.168.4.11
- PC2: 192.168.4.12
- PC3: 192.168.4.13
- PC4: 192.168.4.14
- PC5: 192.168.4.15

CONFIGURING THE DEVICES:

Now that we have added our devices and connected them, we can start configuring them. We will start by configuring the router. Right-click on the router and select "CLI". This will open the command-line interface (CLI) for the router. In the CLI, enter the following commands:

#enable
#configure terminal
#interface FastEthernet0/0
#ip address {IP address} {subnet mask}
#no shutdown
#exit

interface FastEthernet0/1
ip address {IP address} {subnet mask}

no shutdown
exit

Replace "{IP address}" and "{subnet mask}" with your desired IP address and subnet mask. The first interface, FastEthernet0/0, will be connected to the switch, while the second interface, FastEthernet0/1, will be connected to one of the PCs. These commands configure the router's interfaces with IP addresses and subnet masks.
Next, we will configure the switch. Right-click on the switch and select "CLI". In the CLI, enter the following commands:

```
enable
configure terminal
interface FastEthernet0/1
switchport mode access
exit

interface FastEthernet0/2
switchport mode access
exit
```

These commands configure the switch to operate in access mode on its two ports, which are connected to the two PCs.
Finally, we will configure the PCs. Right-click on each PC and select "Config". In the configuration window, enter the IP address, subnet mask, default gateway, and DNS server information. The IP address and subnet mask should be within the same subnet as the router's FastEthernet0/1 interface.

To configure the GigabitEthernet interface on the router, you can follow these steps:

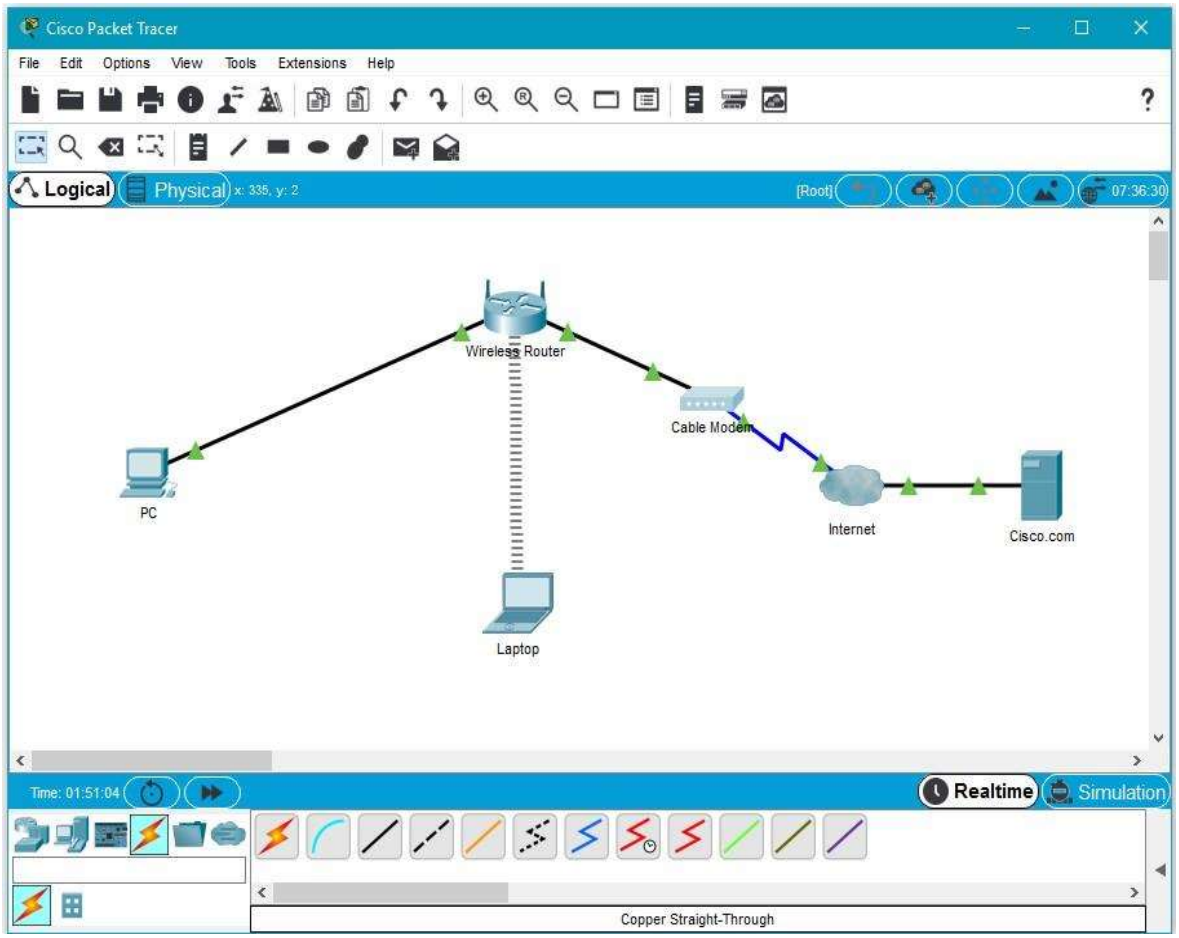
1. Right-click on the router and select "CLI".
2. Enter the following commands:

```
enable
configure terminal
interface GigabitEthernet0/0
ip address {IP address} {subnet mask}
no shutdown
exit
```

Replace "{IP address}" and "{subnet mask}" with your desired IP address and subnet mask. These commands configure the GigabitEthernet interface with an IP address and subnet mask, and enable the interface.

Practical 10

AIM:- b) Design and configure an internetwork using wireless router, DHCP server and internet cloud.



Addressing Table

Device	Interface	IP Address	Subnet Mask	Default Gateway
PC	Ethernet0	DHCP		192.168.0.1
Wireless Router	LAN	192.168.0.1	255.255.255.0	
Wireless Router	Internet	DHCP		
Cisco.com Server	Ethernet0	208.67.220.220	255.255.255.0	
Laptop	Wireless0	DHCP		

Objectives

Part 1: Build a Simple Network in the Logical Topology Workspace

Part 2: Configure the Network Devices
Part 3: Test Connectivity between Network Devices Part 4: Save the File and Close Packet Tracer

Part 1: Build a Simple Network in the Logical Topology Workspace

Step 1: Launch Packet Tracer.

Step 2: Build the topology

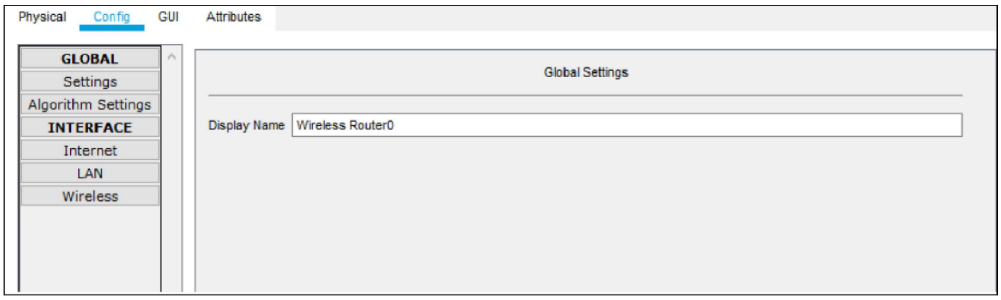
a. Add network devices to the workspace.

Using the device selection box, add the network devices to the workspace as shown in the topology diagram.

To place a device onto the workspace, first choose a device type from the **Device-Type Selection** box. Then, click on the desired device model from the **Device-Specific Selection** box. Finally, click on a location in the workspace to put your device in that location. If you want to cancel your selection, click the **Cancel** icon for that device. Alternatively, you can click and drag a device from the **Device-Specific Selection** box onto the workspace.

b. Change display names of the network devices.

To change the display names of the network devices click on the device icon on the Packet Tracer **Logical** workspace, then click on the **Config** tab in the device configuration window. Type the new name of the device into the **Display Name** box as show in the figure below.



c. Add the physical cabling between devices on the workspace

Using the device selection box, add the physical cabling between devices on the workspace as shown in the topology diagram.

The PC will need a copper straight-through cable to connect to the wireless router. Select the copper straight-through cable in the device selection box and attach it to the FastEthernet0 interface of the PC and the Ethernet 1 interface of the wireless router.

The wireless router will need a copper straight-through cable to connect to the cable modem. Select the copper straight-through cable in the device-selection box and attach it to the Internet interface of the wireless router and the Port 1 interface of the cable modem.

The cable modem will need a coaxial cable to connect to the Internet cloud. Select the coaxial cable in the device-selection box and attach it to the Port 0 interface of the cable modem and the coaxial interface of the Internet cloud.

The Interne cloud will need copper straight-through cable to connect to the Cisco.com server. Select the copper straight-through cable in the device-selection box and attach it to the Ethernet interface of the Internet cloud and the FastEthernet0 interface of the Cisco.com server.

Part 2: Configure the Network Devices

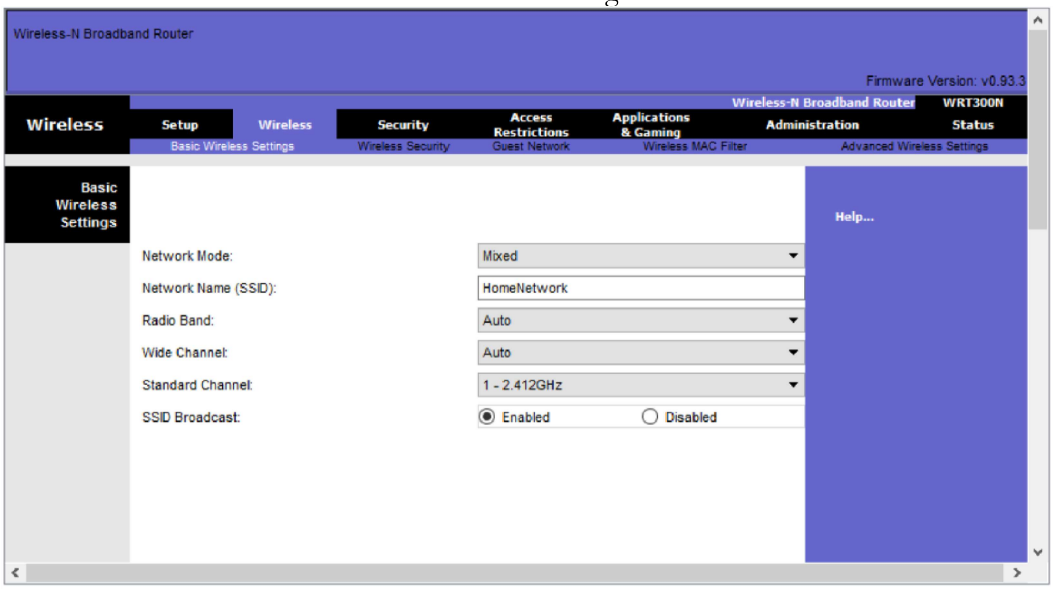
Step 1: Configure the wireless router

a. Create the wireless network on the wireless router

Click on the **Wireless Router** icon on the Packet Tracer **Logical** workspace to open the device configuration window.

In the wireless router configuration window, click on the **GUI** tab to view configuration options for the wireless router.

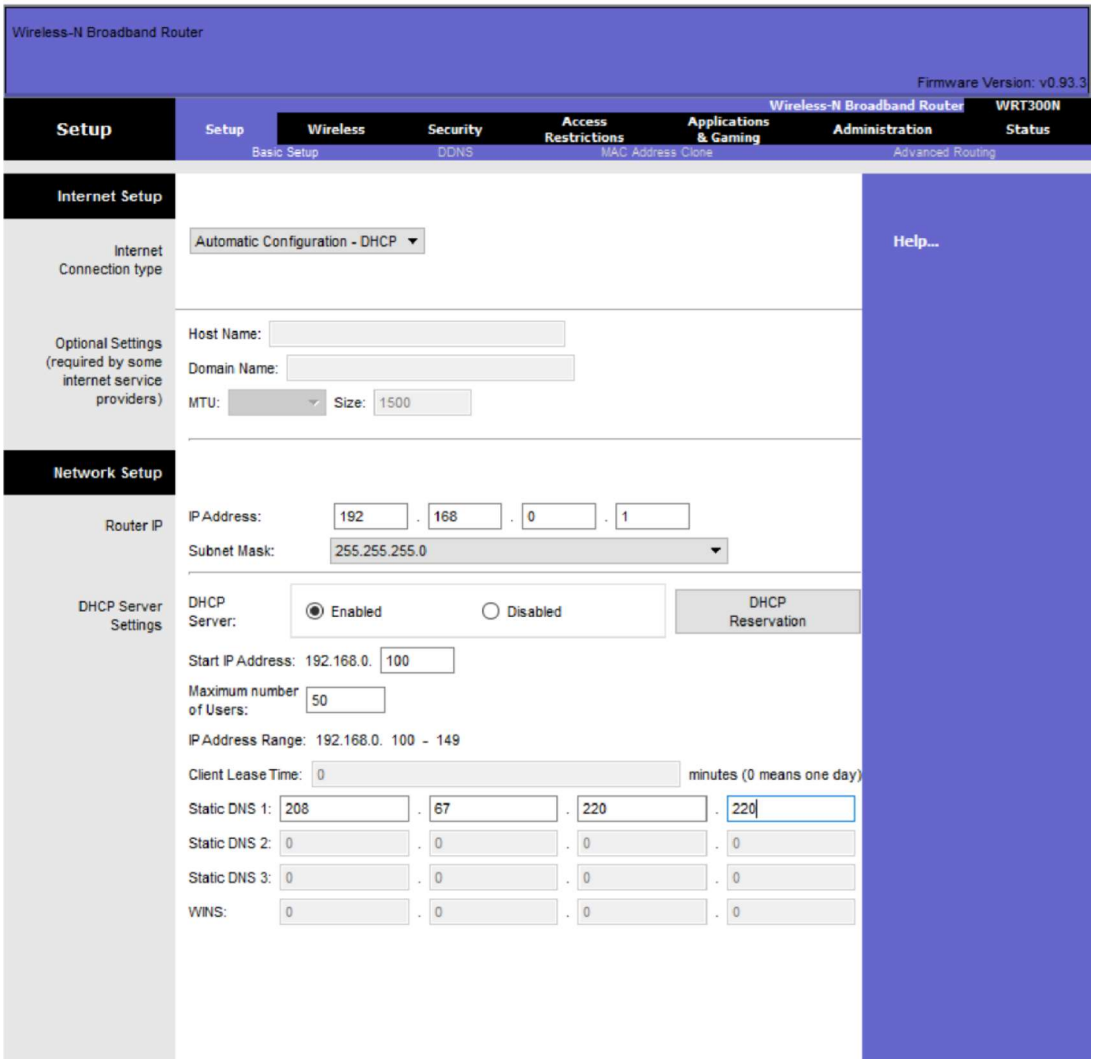
Next, click on the **Wireless** tab in the GUI to view the wireless settings. The only setting that needs to be changed from the defaults is the **Network Name (SSID)**. Here, type the name "HomeNetwork" as shown in the figure.



Configure the Internet connection on the wireless router
Click on the **Setup** tab in the wireless router GUI.

In the **DHCP Server** settings verify that the **Enabled** button is selected and configure the static IP address of the DNS server as 208.67.220.220 as shown in the figure.

b. Click on the **Save Settings** tab.



Step 2: Configure the laptop

a. Configure the Laptop to access the wireless network

Click on the Laptop icon on the Packet Tracer **Logical** workspace and in the laptop configuration windows select the **Physical** tab.

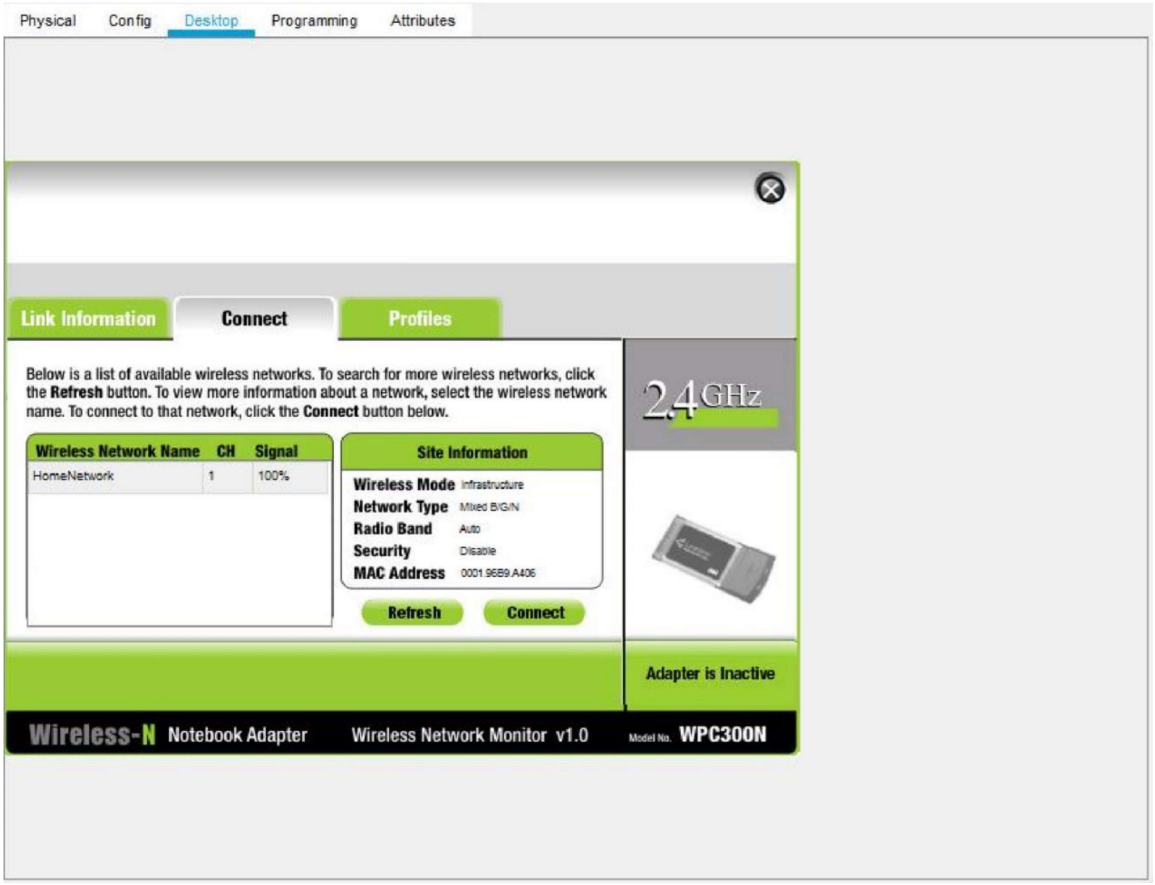
In the **Physical** tab you will need to remove the Ethernet copper module and replace it with the Wireless WPC300N module.

To do this, you first power the Laptop off by clicking the power button on the side of the laptop. Then remove the currently installed Ethernet copper module by clicking on the module on the side of the laptop and dragging it to the **MODULES** pane on the left of the laptop window. Then install the Wireless WPC300N module by clicking on it in the **MODULES** pane and dragging it to the empty module port on the side of the laptop. Power the laptop back on by clicking on the Laptop power button again.

With the wireless module installed, the next task is to connect the laptop to the wireless network.

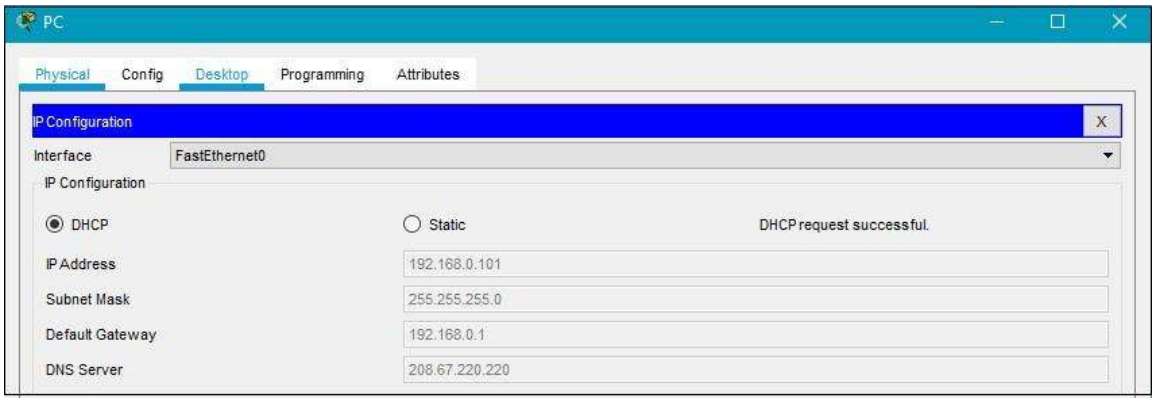
Click on the **Desktop** tab at the top of the Laptop configuration window and select the **PC Wireless** icon.

Once the Wireless-N Notebook Adapter settings are visible, select the **Connect** tab. The wireless network "HomeNetwork" should be visible in the list of wireless networks as shown in the figure.
Select the network, and click on the **Connect** tab found below the **Site Information pane.**

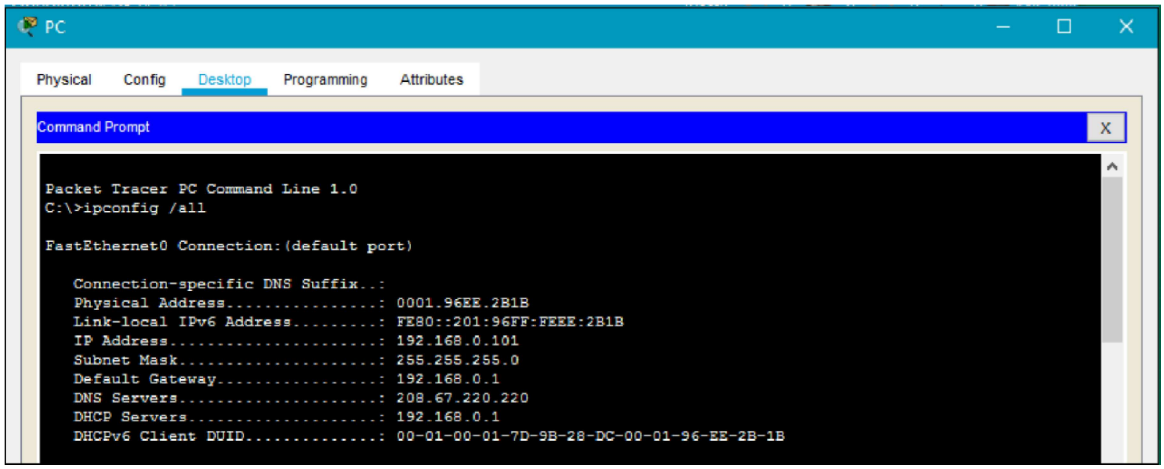


Step 3: Configure the PC

a. Configure the PC for the wired network
Click on the **PC** icon on the Packet Tracer **Logical** workspace and select the **Desktop** tab and then the **IP Configuration** icon.
In the IP Configuration window, select the **DCHP** radio button as shown in the figure so that the PC will use DCHP to receive an IPv4 address from the wireless router. Close the IP Configuration window.



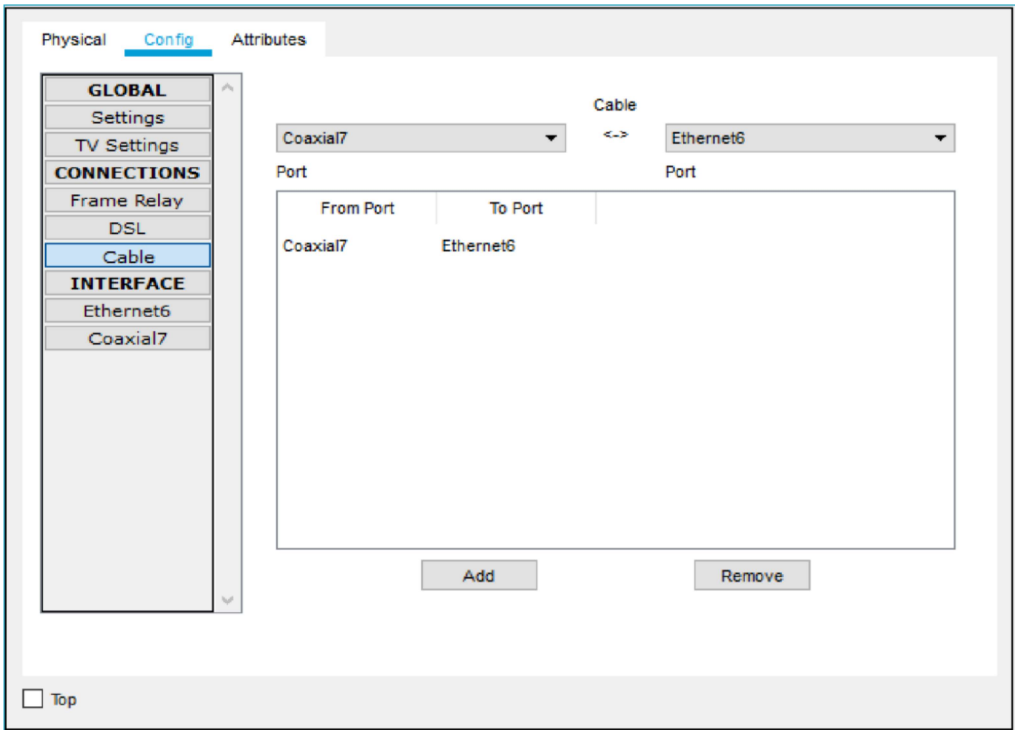
Click on the Command Prompt icon. Verify that the PC has received an IPv4 address by issuing the **ipconfig /all** command from the command prompt as shown in the figure. The PC should receive an IPv4 address in the 192.168.0.x range.



Step 4: Configure the Internet cloud

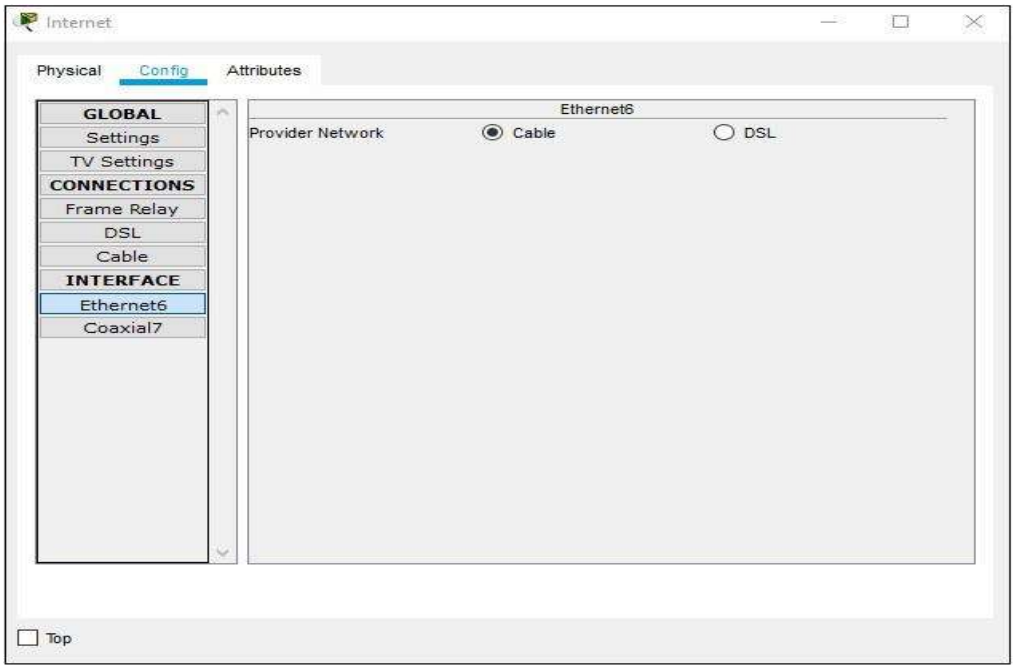
a. Install network modules if necessary
Click on the **Internet Cloud** icon on the Packet Tracer **Logical** workspace and then click on the **Physical** tab. The cloud device will need two modules if they are not already installed. The PT-CLOUD-NM-1CX which is for the cable modem service connection and the PT-CLOUD-NM-1CFE which is for a copper Ethernet cable connection. If these modules are missing, power off the physical cloud devices by clicking on the power button and drag each module to an empty module port on the device and then power the device back on.

b. Identify the From and To Ports
Click on the **Config** tab in the Cloud device window. In the left pane click on **Cable** under **CONNECTIONS**. In the first drop down box choose Coaxial and in the second drop down box choose
Ethernet then click the **Add** button to add these as the **From Port** and **To Port** as shown in the figure.



c. Identify the type of provider

While still in the **Config** tab click Ethernet under **INTERFACE** in the left pane. In the Ethernet configuration window select **Cable** as the Provider Network as shown in the figure.



Step 5: Configure the Cisco.com server

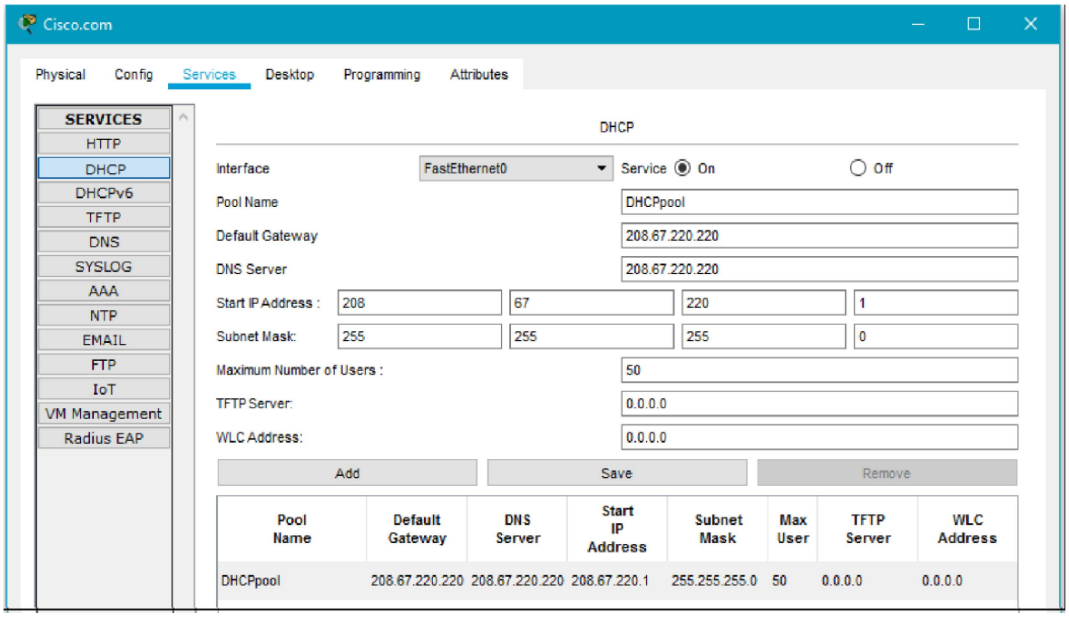
a. Configure the Cisco.com server as a DHCP server

Click on the Cisco.com server icon on the Packet Tracer **Logical** workspace and select the **Services** tab. Select **DHCP** from the **SERVICES** list in the left pane.

In the DHCP configuration window, configure a DHCP as shown in the figure with the following settings.

- Click **On** to turn the DCHP service on
- Pool name: DHCPpool
- Default Gateway: 208.67.220.220
- DNS Server: 208.67.220.220
- Starting IP Address: 208.67.220.1
- Subnet Mask 255.255.255.0
- Maximum number of Users: 50

Click **Add** to add the pool



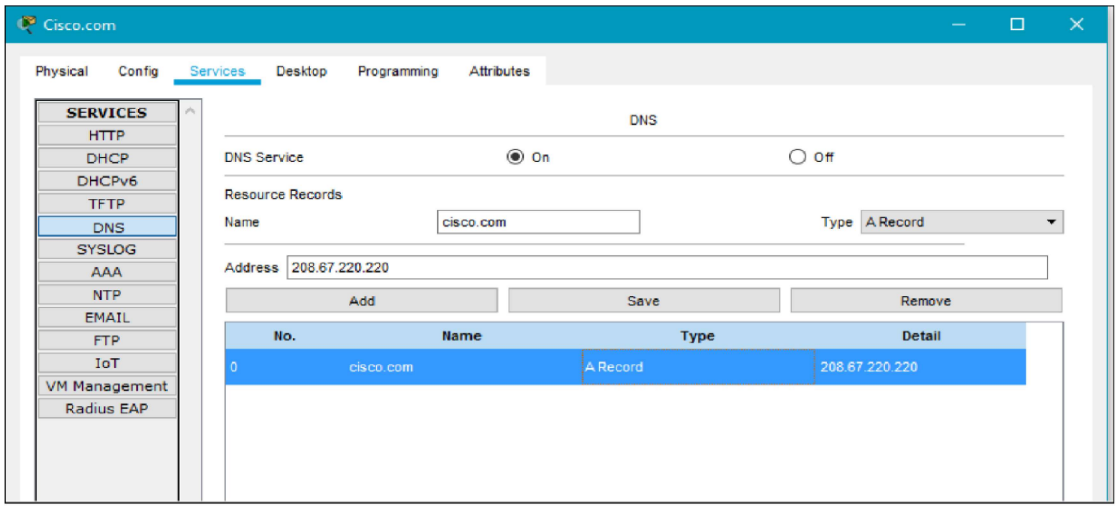
b. Configure the Cisco.com server as a DNS server to provide domain name to IPv4 address resolution.

While still in the **Services** tab, select **DNS** from the **SERVICES** listed in the left pane.

Configure the DNS service using the following settings as shown in the figure.

- Click **On** to turn the DNS service on
- Name: Cisco.com
- Type: A Record
- Address: 208.67.220.220

Click **Add** to add the DNS service settings



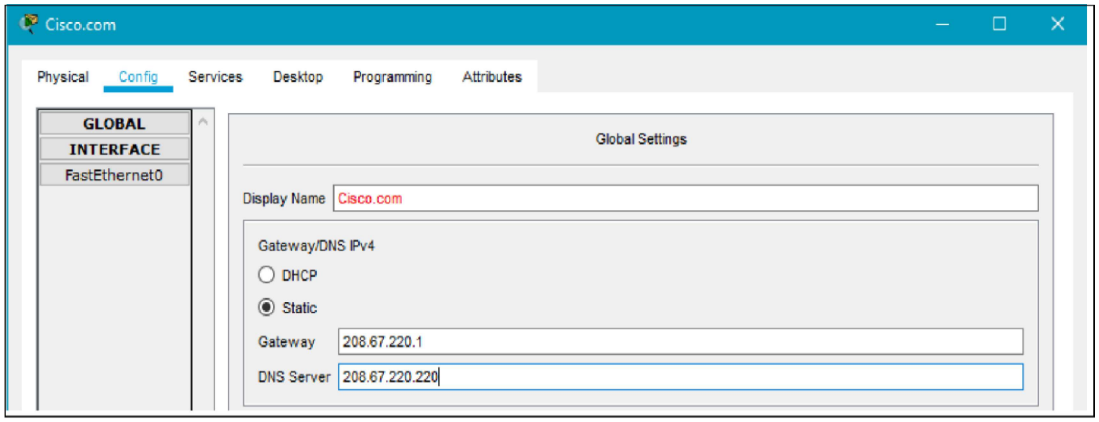
c. Configure the Cisco.com server Global settings.

Select the **Config** tab.

Click on **Settings** in left pane.

Configure the Global settings of the server as follows:

- Select **Static**
- Gateway: 208.67.220.1
- DNS Server: 208.67.220.220

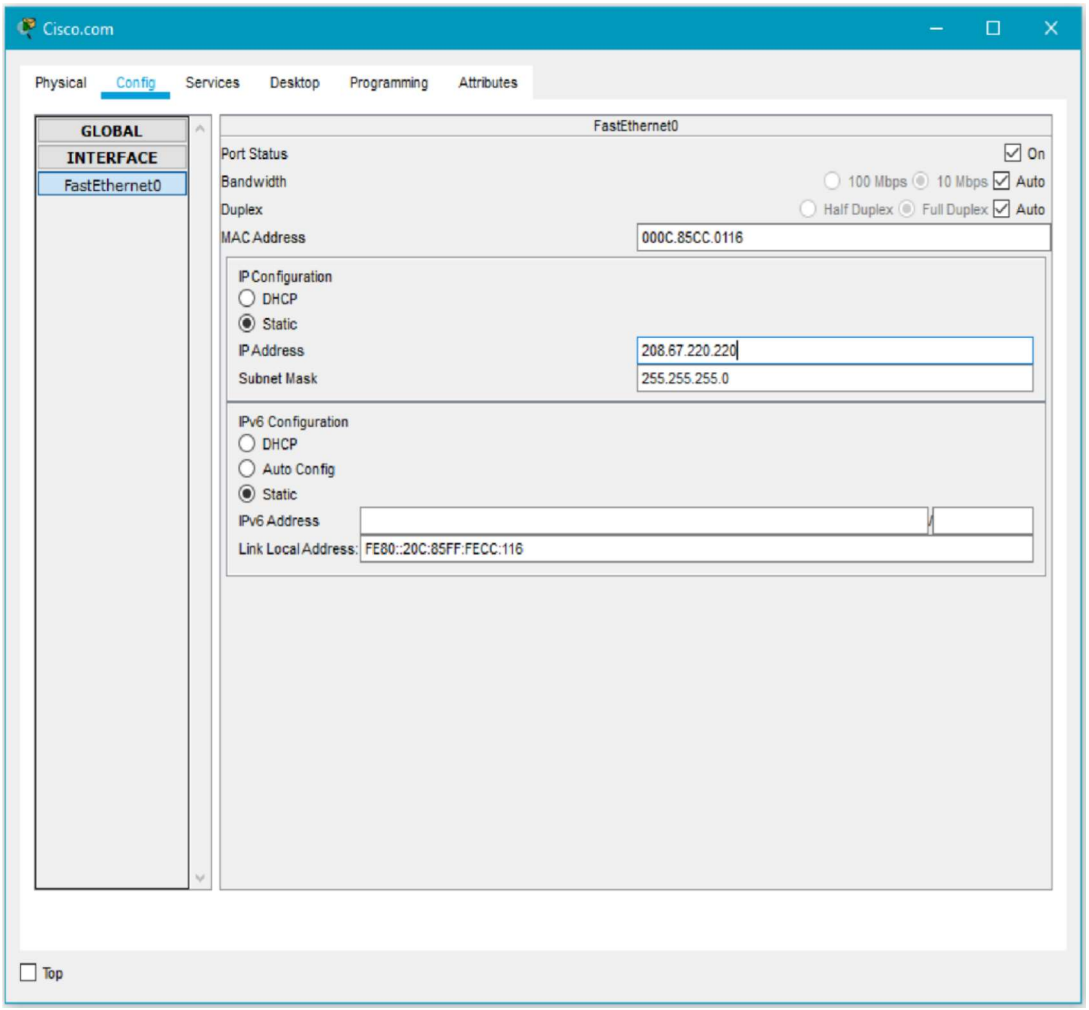


d. Configure the Cisco.com server FastEthernet0 Interface settings.

Click on **Fast Ethernet** in left pane of the **Config** tab

Configure the Fast Ethernet Interface settings of the server as follows:

- Select **Static** under IP Configuration
- IP Address: 208.67.220.220
- Subnet Mask: 255.255.255.0



Part 3: Verify Connectivity

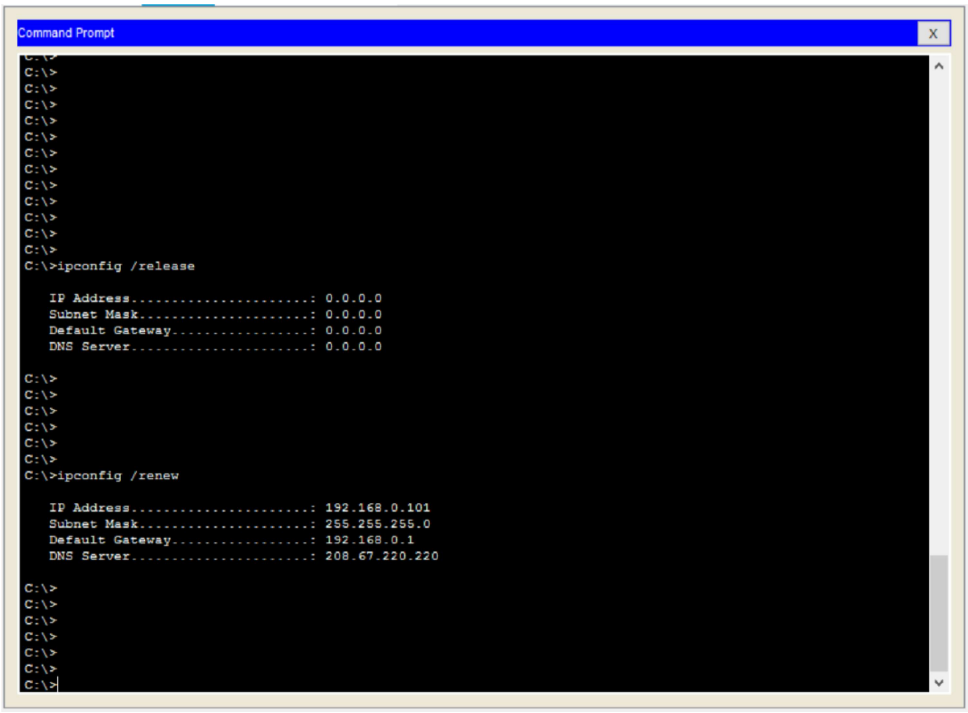
Step 1: Refresh the IPv4 settings on the PC

a) Verify that the PC is receiving IPv4 configuration information from DHCP.

Click on the **PC** on the Packet Tracer **Logical** workspace and then the select the **Desktop** tab of the PC configuration window.

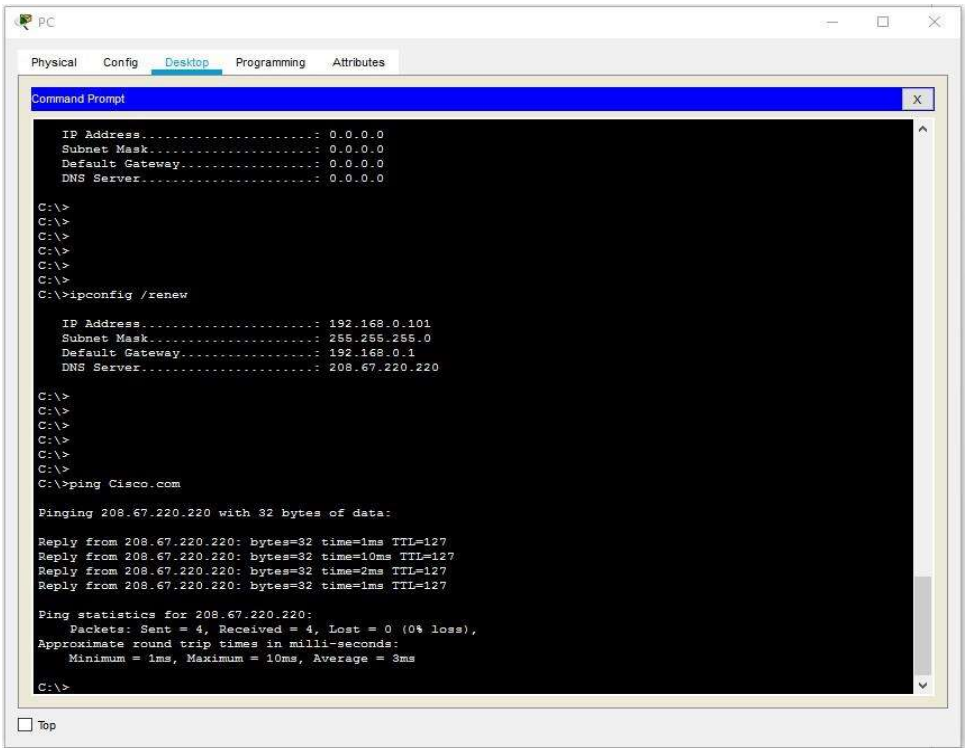
Click on the **Command Prompt** icon

In the command prompt refresh the IP settings by issuing the commands **ipconfig /release** and then **ipconfig /renew.** The output should show that the PC has an IP address in the 192.168.0.x range, a subnet mask, a default gateway, and DNS server address as shown in the figure.



b) Test connectivity to the Cisco.com server from the PC

From the command prompt, issue the command **ping Cisco.com.** It may take a few seconds for the ping to return. Four replies should be received as shown in the figure.



Practical 11

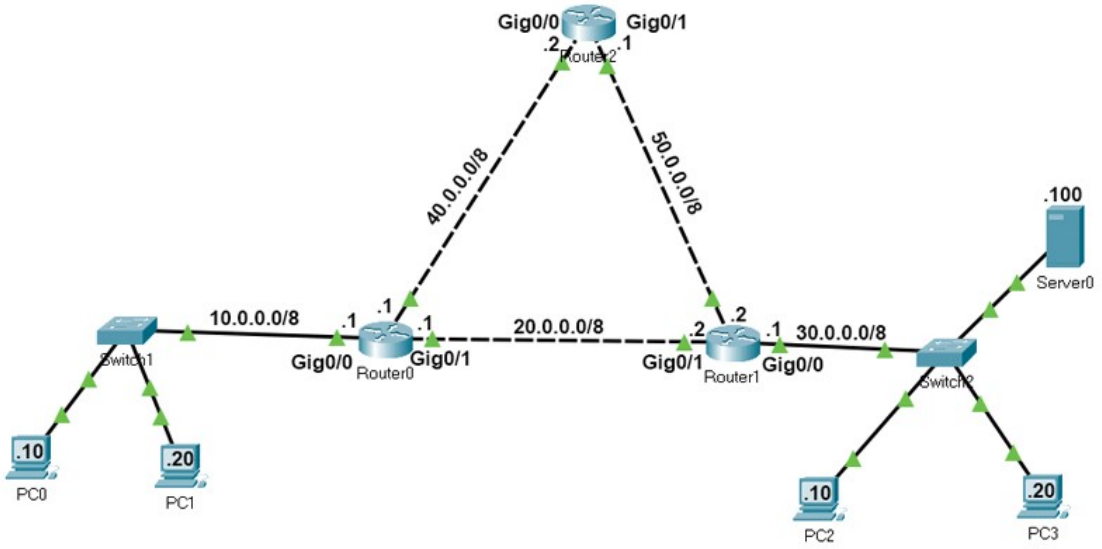
AIM:- a)Simulate Static Routing Configuration using CISCO Packet Tracer

Static routes are the routes you manually add to the router's routing table. The process of adding static routes to the routing table is known as static routing. Let's take a packet tracer example to understand how to use static routing to create and add a static route to the routing table.

Setting up a practice lab

Create a packet tracer lab as shown in the following image or download the following pre-created lab and load it on Packet Tracer.

Packet Tracer Lab with Initial IP Configuration



In this lab, each network has two routes to reach. We will configure one route as the main route and another route as the backup route. If the link bandwidth of all routes is the same, we use the route that has the least number of routers as the main route. If the link bandwidth and the number of routers are the same, we can use any route as the main route and another route as the backup route.

If we specify two routes for the same destination, the router automatically selects the best route for the destination and adds the route to the routing table. If you manually want to select a route that the router should add to the routing table, you have to set the AD value of the route lower than other routes. For example, if you use the following commands to create two static routes for network 30.0.0/8, the route will place the first route to the routing table.

```
#ip route 30.0.0.0 255.0.0.0 20.0.0.2 10
#ip route 30.0.0.0 255.0.0.0 40.0.0.2 20
```

If the first route fails, the router automatically adds the second route to the routing table.

Creating, adding, verifying static routes

Routers automatically learn their connected networks. We only need to add routes for the networks that are not available on the router's interfaces. For example, network 10.0.0.0/8, 20.0.0.0/8 and 40.0.0.0/8 are directly connected to Router0. Thus, we don't need to configure routes for these

networks. Network 30.0.0.0/8 and network 50.0.0.0/8 are not available on Router0. We have to create and add routes only for these networks.

The following table lists the connected networks of each router.

Router	Available networks on local interfaces	Networks available on other routers' interfaces
Router0	10.0.0.0/8, 20.0.0.0/8, 40.0.0.0/8	30.0.0.0/8, 50.0.0.0/8
Router1	20.0.0.0/8, 30.0.0.0/8, 50.0.0.0/8	10.0.0.0/8, 40.0.0.0/8
Router2	40.0.0.0/8, 50.0.0.0/8	10.0.0.0/8, 20.0.0.0/8, 30.0.0.0/8

Let's create static routes on each router for networks that are not available on the router.

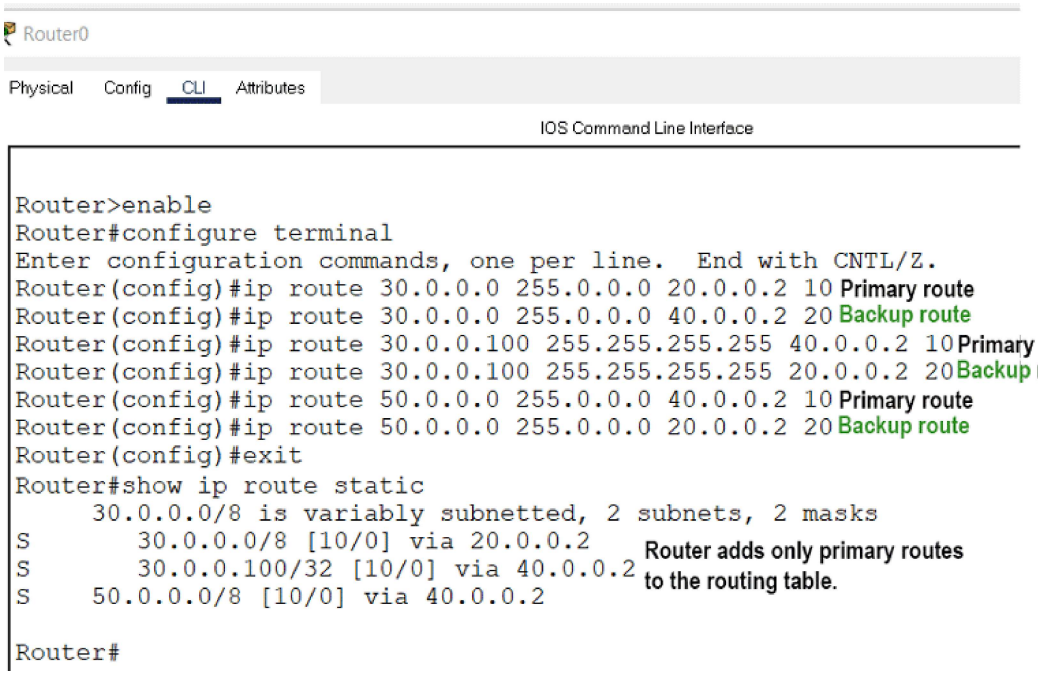
Router0 requirements

- Create two routes for network 30.0.0.0/8 and configure the first route (via -Router1) as the main route and the second route (via-Router2) as a backup route.
- Create two routes for the host 30.0.0.100/8 and configure the first route (via -Router2) as the main route and the second route (via-Router1) as a backup route.
- Create two routes for network 50.0.0.0/8 and configure the first route (via -Router2) as the main route and the second route (via-Router1) as a backup route.
- Verify the router adds only main routes to the routing table.

Router0 configuration

Access the CLI prompt of Router0 and run the following commands.

```
Router>enable
Router#configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
Router(config)#ip route 30.0.0.0 255.0.0.0 20.0.0.2 10
Router(config)#ip route 30.0.0.0 255.0.0.0 40.0.0.2 20
Router(config)#ip route 30.0.0.100 255.255.255.255 40.0.0.2 10
Router(config)#ip route 30.0.0.100 255.255.255.255 20.0.0.2 20
Router(config)#ip route 50.0.0.0 255.0.0.0 40.0.0.2 10
Router(config)#ip route 50.0.0.0 255.0.0.0 20.0.0.2 20
Router(config)#exit
Router#show ip route static
30.0.0.0/8 is variably subnetted, 2 subnets, 2 masks
S 30.0.0.0/8 [10/0] via 20.0.0.2
S 30.0.0.100/32 [10/0] via 40.0.0.2
S 50.0.0.0/8 [10/0] via 40.0.0.2
Router#
```

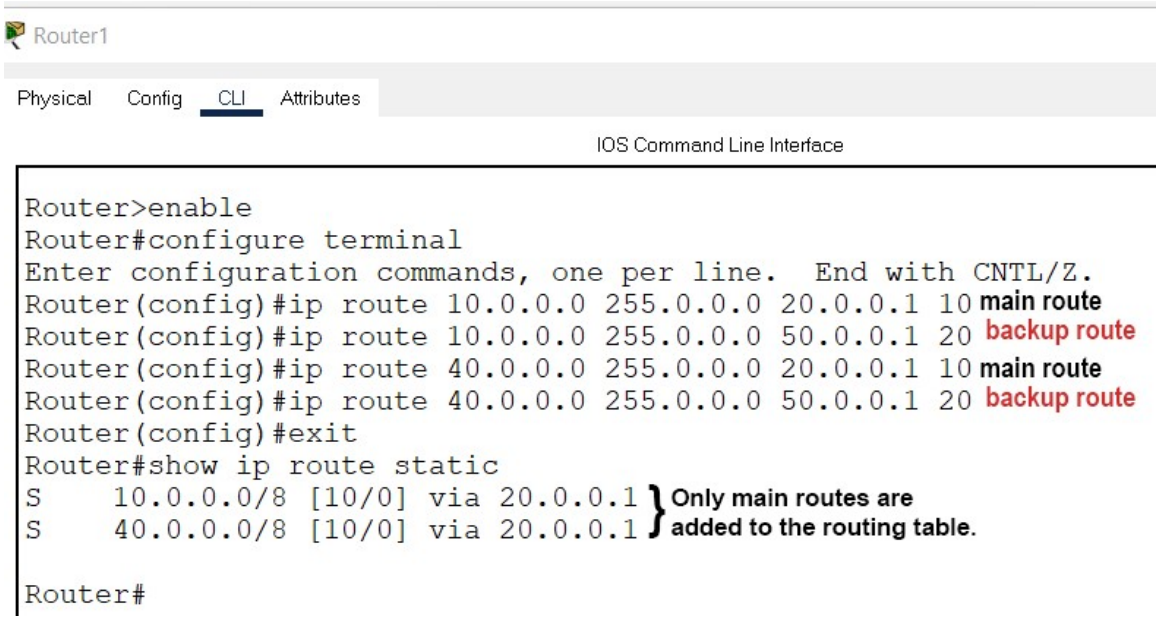


Router1 requirements

- Create two routes for network 10.0.0.0/8 and configure the first route (via -Router0) as the main route and the second route (via-Router1) as a backup route.
- Create two routes for network 40.0.0.0/8 and configure the first route (via -Router0) as the main route and the second route (via-Router2) as a backup route.
- Verify the router adds only main routes to the routing table.

Router1 configuration

```
Router>enable
Router#configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
Router(config)#ip route 10.0.0.0 255.0.0.0 20.0.0.1 10
Router(config)#ip route 10.0.0.0 255.0.0.0 50.0.0.1 20
Router(config)#ip route 40.0.0.0 255.0.0.0 20.0.0.1 10
Router(config)#ip route 40.0.0.0 255.0.0.0 50.0.0.1 20
Router(config)#exit
Router#show ip route static
S 10.0.0.0/8 [10/0] via 20.0.0.1
S 40.0.0.0/8 [10/0] via 20.0.0.1
Router#
```

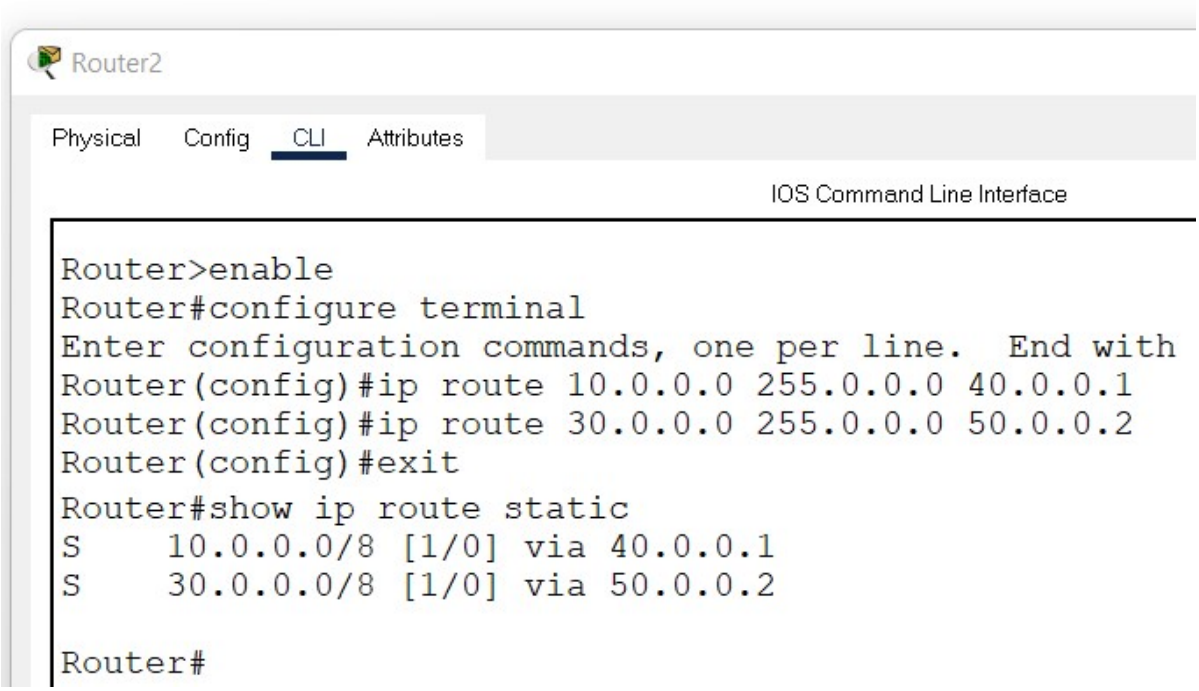


Router2 requirements

Create static routes for network 10.0.0.0/8 and network 30.0.0.0/8 and verify the router adds both routes to the routing table.

Router2 configuration

```
Router>enable
Router#configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
Router(config)#ip route 10.0.0.0 255.0.0.0 40.0.0.1
Router(config)#ip route 30.0.0.0 255.0.0.0 50.0.0.2
Router(config)#exit
Router#show ip route static
S 10.0.0.0/8 [1/0] via 40.0.0.1
S 30.0.0.0/8 [1/0] via 50.0.0.2
Router#
```



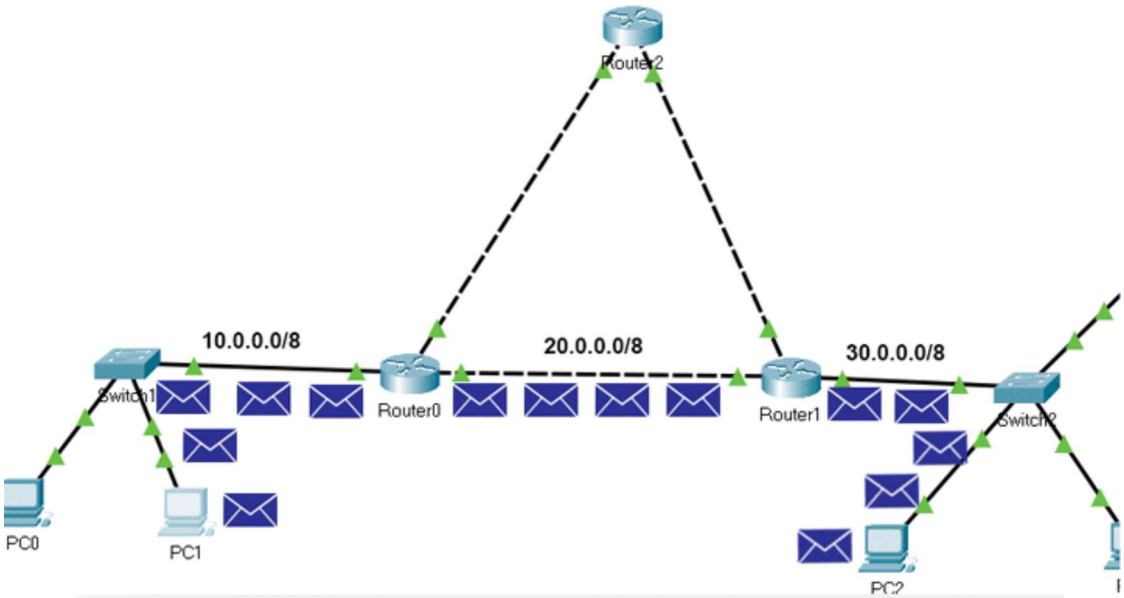
Verifying static routing

On Router0, we configured two routes for network 30.0.0.0/8. These routes are via Router1 and via Router2. We set the first route (via-Router1) as the main route and the second route as the backup route. We can verify this configuration in two ways.

By sending ping requests to a PC of network 30.0.0.0/8 and tracing the path they take to reach the network 30.0.0.0/8. For this, you can use **'tracert'** command on a PC of network 10.0.0.0/8. The **'tracert'** command sends ping requests to the destination host and tracks the path they take to reach the destination.

By listing the routing table entries on Router0. Since a router uses the routing table to forward data packets, you can check the routing table to figure out the route the router uses to forward data packets for each destination.

The following image shows the above testing.



Router0

Physical Config CLI Attributes

IOS Command Line Interface

```
Router>enable
Router#show ip route static
     30.0.0.0/8 is variably subnetted, 2 subnets, 2 masks
S       30.0.0.0/8 [10/0] via 20.0.0.2
S       30.0.0.100/32 [10/0] via 40.0.0.2
S    50.0.0.0/8 [10/0] via 40.0.0.2

Router#
```

PC1

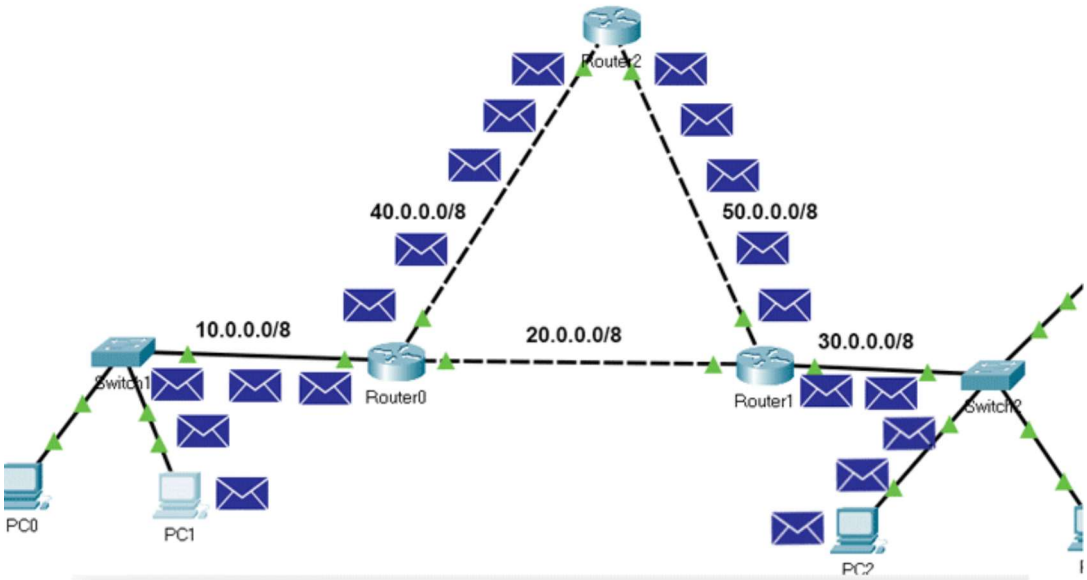
Physical Config Desktop Programming Attributes

Command Prompt

```
C:\>tracert 30.0.0.20

Tracing route to 30.0.0.20 over a maximum of 30
hops:

  1    0 ms      0 ms      0 ms      10.0.0.1
  2    0 ms      0 ms      0 ms      20.0.0.2
  3    *         0 ms      0 ms      30.0.0.20

Trace complete.

C:\>
```

We also configured a separate static host route for the host 30.0.0.100/8. The router must use this route to forward data packets to the host 30.0.0.100/8. To verify this, you can do the same testing for the host 30.0.0.100/8.

The following image shows this testing.



Router0

Physical Config CLI Attributes

IOS Command Line Interface

```
Router>enable
Router#show ip route static
     30.0.0.0/8 is variably subnetted, 2 subnets, 2 masks
S       30.0.0.0/8 [10/0] via 20.0.0.2
S       30.0.0.100/32 [10/0] via 40.0.0.2
S    50.0.0.0/8 [10/0] via 40.0.0.2

Router#
```

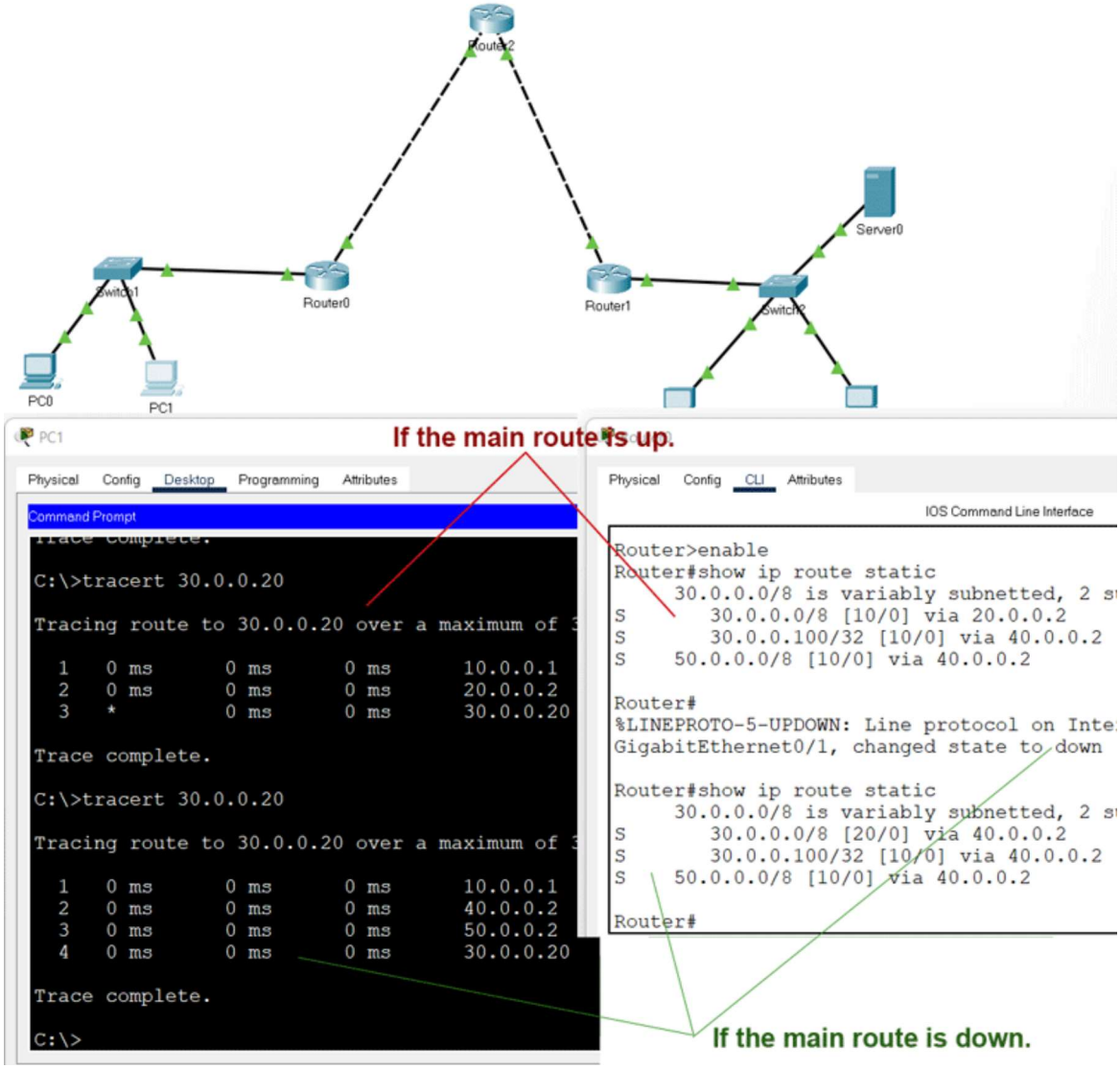
PC1

Physical Config Desktop Programming Attributes

```
Packet Tracer PC Command Line 1.0
C:\>tracert 30.0.0.100

Tracing route to 30.0.0.100 over a maximum of 30
hops:

  1    1 ms       0 ms       0 ms       10.0.0.1
  2    *          0 ms       0 ms       40.0.0.2
  3    *          *          0 ms       20.0.0.2
  4    *          0 ms       0 ms       30.0.0.100

Trace complete.

C:\>
```

We also configured a backup route for network 30.0.0.0/8. The router must put the backup route to the routing table and use it to forward data packets to network 30.0.0.0/8 when the main route fails. To verify this, we have to simulate the failure of the main route.

To simulate the failure of the main route, you can delete the link between Router0 and Router1. After deleting the link, do the same testing again for the network 30.0.0.0/8.



The following link provides the configured packet tracer lab of the above example.
Packet Tracer Lab with Static Routing Configuration

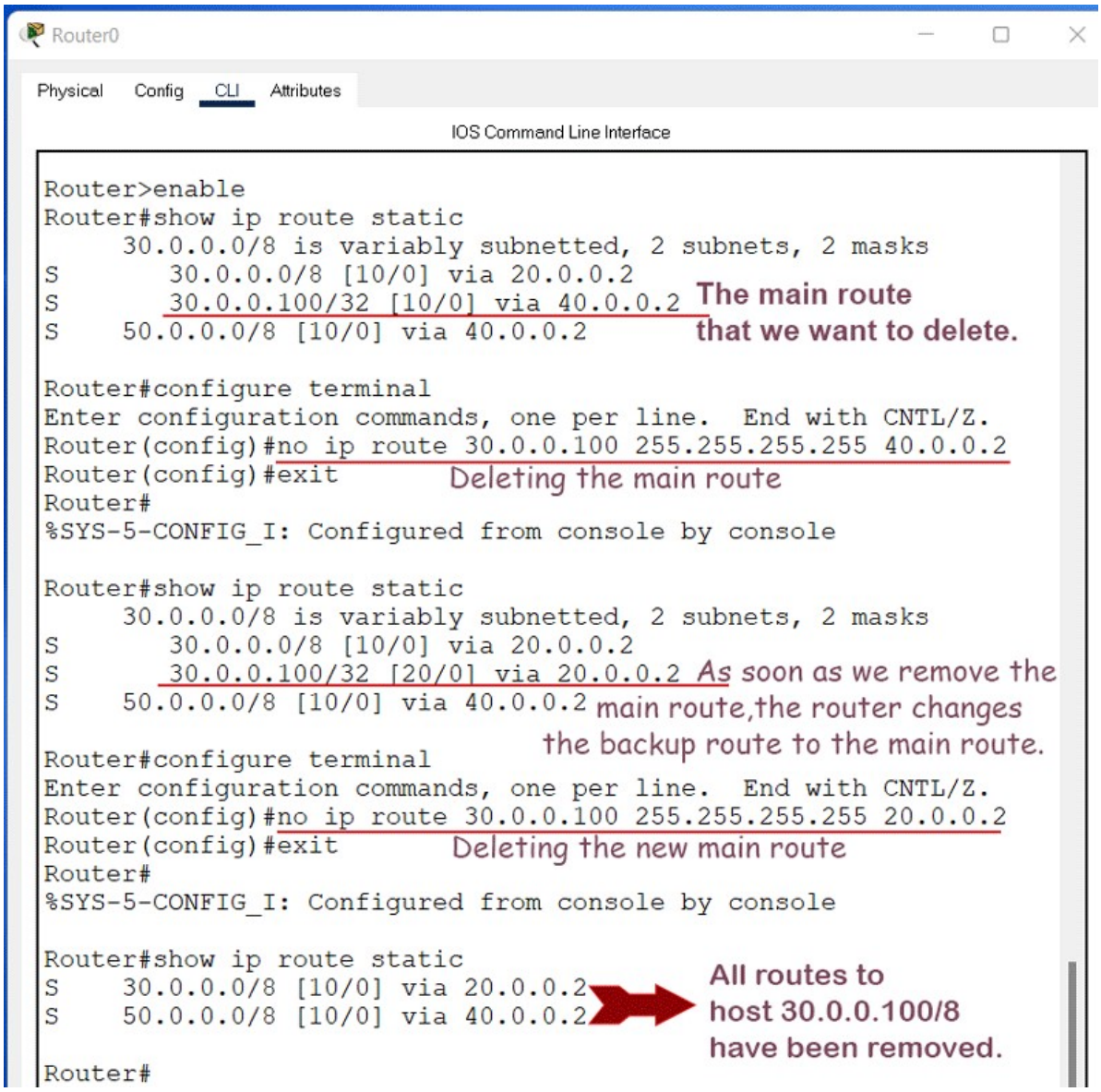
Deleting a static route

To delete a static route, use the following steps.

- Use the **'show ip route static'** command to print all static routes.
- Note down the route you want to delete.
- Use the **'no ip route'** command to delete the route.

If you have a backup route, the backup route becomes the main route when you delete the main route.

In our example, we have a backup route and a main route for the host 30.0.0.100/8. The following image shows how to delete both routes.


Router0
Physical
Config
CLI
Attributes
IOS Command Line Interface
Router>enable
Router#show ip route static
30.0.0.0/8 is variably subnetted, 2 subnets, 2 masks
S 30.0.0.0/8 [10/0] via 20.0.0.2
S 30.0.0.100/32 [10/0] via 40.0.0.2
S 50.0.0.0/8 [10/0] via 40.0.0.2
The main route that we want to delete.
Router#configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
Router(config)#no ip route 30.0.0.100 255.255.255.255 40.0.0.2
Router(config)#exit
Deleting the main route
Router#
%SYS-5-CONFIG_I: Configured from console by console
Router#show ip route static
30.0.0.0/8 is variably subnetted, 2 subnets, 2 masks
S 30.0.0.0/8 [10/0] via 20.0.0.2
S 30.0.0.100/32 [20/0] via 20.0.0.2
S 50.0.0.0/8 [10/0] via 40.0.0.2
As soon as we remove the main route,the router changes the backup route to the main route.
Router#configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
Router(config)#no ip route 30.0.0.100 255.255.255.255 20.0.0.2
Router(config)#exit
Deleting the new main route
Router#
%SYS-5-CONFIG_I: Configured from console by console
Router#show ip route static
S 30.0.0.0/8 [10/0] via 20.0.0.2
S 50.0.0.0/8 [10/0] via 40.0.0.2
All routes to host 30.0.0.100/8 have been removed.
Router#

Practical 11

AIM:- **b)Simulate RIP using CISCO Packet Tracer**

Initial IP configuration

Device	Interface	IP Configuration	Connected with
PC0	Fast Ethernet	10.0.0.2/8	Router0's Fa0/1
Router0	Fa0/1	10.0.0.1/8	PC0's Fast Ethernet
Router0	S0/0/1	192.168.1.254/30	Router2's S0/0/1
Router0	S0/0/0	192.168.1.249/30	Router1's S0/0/0
Router1	S0/0/0	192.168.1.250/30	Router0's S0/0/0
Router1	S0/0/1	192.168.1.246/30	Router2's S0/0/0
Router2	S0/0/0	192.168.1.245/30	Router1's S0/0/1
Router2	S0/0/1	192.168.1.253/30	Router0's S0/0/1
Router2	Fa0/1	20.0.0.1/30	PC1's Fast Ethernet
PC1	Fast Ethernet	20.0.0.2/30	Router2's Fa0/1

Assign IP address to PCs

Double click **PCs** and click **Desktop** menu item and click **IP Configuration**. Assign IP address referring the above table.

Assign IP address to interfaces of routers

Double click **Router0** and click **CLI** and press **Enter key** to access the command prompt of **Router0**.

We need to configure IP address and other parameters on interfaces before we could actually use them for routing. Interface mode is used to assign IP address and other parameters. Interface mode can be accessed from global configuration mode. Following commands are used to access the global configuration mode.

```
Router>enable
Router#configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)#
```

From global configuration mode we can enter in interface mode. From there we can configure the interface. Following commands will assign IP address on FastEthernet0/0.

```
Router(config)#interface fastEthernet 0/0
Router(config-if)#ip address 10.0.0.1 255.0.0.0
Router(config-if)#no shutdown
Router(config-if)#exit
Router(config)#
```

interface *fastEthernet 0/0* command is used to enter in interface mode.
ip address *10.0.0.1 255.0.0.0* command will assign IP address to interface.

no shutdown command will bring the interface up.
exit command is used to return in global configuration mode.
Serial interface needs two additional parameters **clock rate** and **bandwidth**. Every serial cable has two ends DTE and DCE. These parameters are always configured at DCE end.
We can use **show controllers interface** command from privilege mode to check the cable's end.

```
Router#show controllers serial 0/0/0
Interface Serial0/0/0
Hardware is PowerQUICC MPC860
DCE V.35, clock rate 2000000
[Output omitted]
```

Fourth line of output confirms that DCE end of serial cable is attached. If you see DTE here instead of DCE skip these parameters.
Now we have necessary information let's assign IP address to serial interface.

```
Router#configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)#interface serial 0/0/0
Router(config-if)#ip address 192.168.1.249 255.255.255.252
Router(config-if)#clock rate 64000
Router(config-if)#bandwidth 64
Router(config-if)#no shutdown
Router(config-if)#exit
Router(config)#interface serial 0/0/1
Router(config-if)#ip address 192.168.1.254 255.255.255.252
Router(config-if)#clock rate 64000
Router(config-if)#bandwidth 64
Router(config-if)#no shutdown
Router(config-if)#exit
Router(config)#
```

Router#configure terminal Command is used to enter in global configuration mode.
Router(config)#interface *serial 0/0/0* Command is used to enter in interface mode.
Router(config-if)#ip address *192.168.1.249 255.255.255.252* Command assigns IP address to interface. For serial link we usually use IP address from /30 subnet.
Router(config-if)#clock rate *64000* And **Router(config-if)#bandwidth *64*** In real life environment these parameters control the data flow between serial links and need to be set at service providers end. In lab environment we need not to worry about these values. We can use these values.
Router(config-if)#no shutdown Command brings interface up.
Router(config-if)#exit Command is used to return in global configuration mode.
We will use same commands to assign IP addresses on interfaces of remaining routers. We need to provided clock rate and bandwidth only on DCE side of serial interface. Following command will assign IP addresses on interface of Router1.

Router1

```
Router>enable
Router#configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)#interface serial 0/0/0
Router(config-if)#ip address 192.168.1.250 255.255.255.252
```

```
Router(config-if)#no shutdown
Router(config-if)#exit
Router(config)#interface serial 0/0/1
Router(config-if)#ip address 192.168.1.246 255.255.255.252
Router(config-if)#clock rate 64000
Router(config-if)#bandwidth 64
Router(config-if)#no shutdown
Router(config-if)#exit
```

Use same commands to assign IP addresses on interfaces of Router2.

Router2

```
Router>enable
Router#configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)#interface fastEthernet 0/0
Router(config-if)#ip address 20.0.0.1 255.0.0.0
Router(config-if)#no shutdown
Router(config-if)#exit
Router(config)#interface serial 0/0/0
Router(config-if)#ip address 192.168.1.245 255.255.255.252
Router(config-if)#no shutdown
Router(config-if)#exit
Router(config)#interface serial 0/0/1
Router(config-if)#ip address 192.168.1.253 255.255.255.252
Router(config-if)#no shutdown
Router(config-if)#exit
```

Now routers have information about the networks that they have on their own interfaces. Routers will not exchange this information between them on their own. We need to implement RIP routing protocol that will insist them to share this information.

Configure RIP routing protocol

Configuration of RIP protocol is much easier than you think. It requires only two steps to configure the RIP routing.

- Enable RIP routing protocol from global configuration mode.
- Tell RIP routing protocol which networks you want to advertise.

Let's configure it in Router0

Router0

```
Router0(config)#router rip
Router0(config-router)# network 10.0.0.0
Router0(config-router)# network 192.168.1.252
Router0(config-router)# network 192.168.1.248
```

router rip command tell router to enable the RIP routing protocol.

network command allows us to specify the networks which we want to advertise. We only need to specify the networks which are directly connected with the router.

That's all we need to configure the RIP. Follow same steps on remaining routers.

Router1

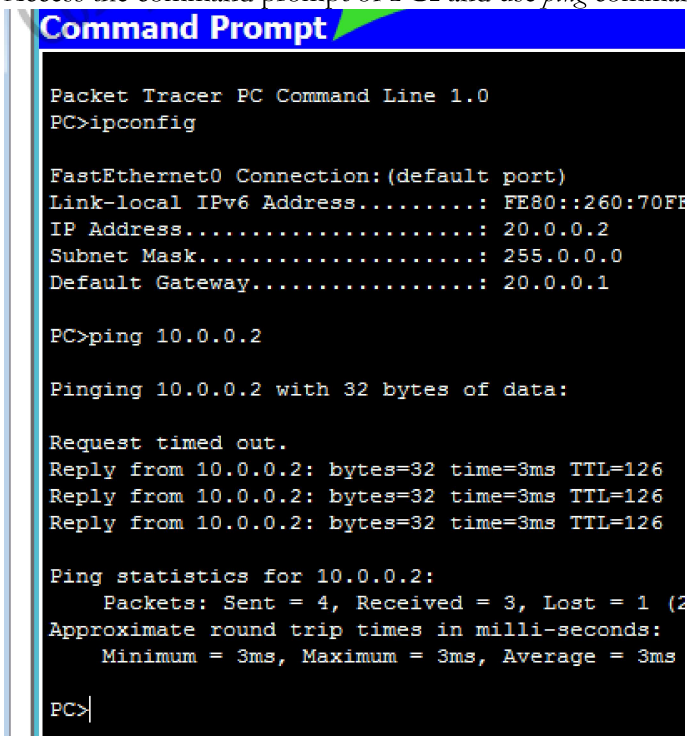
```
Router1(config)#router rip
Router1(config-router)# network 192.168.1.244
Router1(config-router)# network 192.168.1.248
```

Router2

```
Router2(config)#router rip
```

```
Router2(config-router)# network 20.0.0.0
Router2(config-router)# network 192.168.1.252
Router2(config-router)# network 192.168.1.244
```

That's it. Our network is ready to take the advantage of RIP routing. To verify the setup we will use ping command. ping command is used to test the connectivity between two devices.
Access the command prompt of **PC1** and use *ping* command to test the connectivity from **PC0**.



RIP protocol automatically manage all routes for us. If one route goes down, it automatically switches to another available. To explain this process more clearly we have added one more route in our network.
Currently there are two routes between PC0 and PC1.

Route 1

PC0 [Source / destination – 10.0.0.2] <==> Router0 [FastEthernet0/1 – 10.0.0.1] <==> Router0 [Serial0/0/1 – 192.168.1.254] <==> Router2 [Serial 0/0/1 – 192.168.1.253] <==> Router2 [FastEthernet0/0 – 20.0.0.1] <==> PC1 [Destination /source – 20.0.0.2]

Route 2

PC0 [Source / destination – 10.0.0.2] <==> Router0 [FastEthernet0/1 – 10.0.0.1] <==> Router0 [Serial0/0/0 – 192.168.1.249] <==> Router1 [Serial 0/0/0 – 192.168.1.250] <==> Router1 [Serial 0/0/1 – 192.168.1.246] <==> Router2 [Serial 0/0/0 – 192.168.1.245] <==> Router2 [FastEthernet0/0 – 20.0.0.1] <==> PC1 [Destination /source – 20.0.0.2]

By default RIP will use the route that has low hops counts between source and destination. In our network route1 has low hops counts, so it will be selected. We can use *tracert* command to verify it.
Now suppose route1 is down. We can simulate this situation by removing the cable attached between **Router0 [s0/0/1]** and **Router2 [s0/0/1]**.

What will happen now? There is no need to worry. RIP will automatically reroute the traffic.
Use ***tracert*** command again to see the magic of dynamic routing.