INDEX

NAME : S.Kumaran STD : SEC. : ROLL NO. : 230701159

S.No.	Date	Title	Page No.	Teacher's Sign/ Remarks
(1)	14/7/25	Study of various network Commands used in linux and windows	1	4/8/25
(2)	21/7/25	Study of different types of network cable	3	
(3)	24/7/25	Experiment on CISCO packet Tracer.	9	14/8/25
(4)	31/7/25	Experiment on Packet Capture tool : wireshark	10	14/8/25
(A)	07/08/25	Setup and Configure LAN	9	14/10/25
(6)	14/08/25	Error Correction at Data link layer		
(8)	11/09/25	NMAP to discover live Hosts using Nmap Scans		
(9)	28/8/25	Flow control at Data link layer		

Palaniappa

S NO	DATE	Title	Page NO	Signature
(9)	15/09/25	Implementation of Subnetting in Cisco packet Tracer Simulator		
(10)	18/09/25	Internetworking with routers in Cisco packet Tracer Simulator		
(11)	22/09/25	Routing at Network Layer		
(12)	25/09/25	End-End Communication at Transport Layer		
(13)	29/09/25	Ping Program		
(14)	06/10/25	Packet Switching		

14/10/25

8

Ex: No: 1
14.07.25

Study of Various Network Commands Used in Linux and Windows

Aim: Study of various Network commands used in linux and windows.

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 routes.

Output: Interface: 192.168.0.101 --- 0x5

Internet Address	physical Address	Type
192.168.0.1	f0-27-31-c6-89-30	Dynamic
192.168.0.100	48-46-C1-22-82-d6	Dynamic
192.168.0.255	ff-ff-ff-ff-ff-ff	Static
224.0.0.22	01-00-5e-00-00-16	Static
224.0.0.251	01-00-5e-00-00-fb	Static

*hostname

This is the simplest of all TCP/IP Commands. It simply display the name of your computer.

output :- K303-83

*ipconfig/all

This command displays detailed configuration information about your TCP/IP connection including Routes, and type of ethernet adapters in your system.

output:- Windows Ip Configuration.

Host Name . . . . . :

Primary Dns Suffix . . . :

Node Type : Mixed
IP Routing enabled : No
WINS proxy enabled : No
Unknown adapter local area connection
Media State ..... Media Connected

* nbtstat - a :
This helps solve problem with NetBios name Resolution (Not stands for NetBios over TCP/IP)

output.
Displays protocol statistics and Current TCP/IP Connections using NBT (NetBios over TCP/IP)

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

-a (adapter status) Lists the Remote machine's name table given its name

-A (Adapter Status) Lists the Remote machine's name table given its IP address

-c (cache) Lists NBT's cache of Remote [machine] names and their IP address

* netstat : (network Statistics)
It displays a variety of statistics about a Computer active TCP/IP Connections.

netstat -Y
output :

Interface list
13...00 00 27 00 00 dd ...Virtual Box Host only Ethernet Adapter

3... 34 7d f6 af 69 94... Microsoft Wifi Direct Virtual Adapter

16... 86 7d f6 af 69 96... Microsoft Wi-fi Direct Virtual Adapter #2

5. 34 7d f6 af 69 96... Intel(R) Wi-Fi 6 AX201 160MHz

IPV4 Route Table

Active Routes:

Network Destination	Netmask	Gateway	Interface	Metric
127.0.0.0	255.0.0.0	On-link	127.0.0.1	331

*nslookup

Combination of interactive and non-interactive

nslookup www.google.com

output:

Server: unknown

Address: 10.207.195.43

Non-authoritative answer:

Name: www.google.com

Address: 2404:6800:4007:81f::2004

142.250.67.68

*pathping:

Basically a Combination of ping and Tracert Commands.

output:

Usage: pathping [-g host-list] [-h maximum hops]
[-i address] [-n] [-p period] [-q num-queries]
[-w timeout] [-4] [-6] target-name

*Ping (Packet Internet Groper)

Best way to test Connectivity between two nodes.

Usage: ping [-t] [-a] [-n count] [-l size] [-f] [-i TTL] [-v TOS]
[-r count] [-s count] [[-j host-list] | [-k host-list]]
[-w timeout] [-R] [-S srcaddr] [-c compartment] [-p]
[-4] [-6] target_name

Ping hostname

pinging hostname [fe80::a461:30ef:3baa:46b3%18]
with 32 bytes of data:

Reply from fe80::a461:30ef:3baa:46b3%18: time<1ms

:

Ping statistics for fe80::a461:30ef:3baa:46b3%18:
Packet: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
Minimum = 0ms, Maximum = 0ms, Avg = 0ms

*Route

Used to show the ip routing table

Route print

Output

IPv4 Route Table

Active Routes:

Network Destination	Netmask	Gateway	Interface	Metric
127.0.0.0	255.0.0.0	On-link	127.0.0.1	331
127.0.0.1	255.255.255.255	On-link	127.0.0.1	331
127.255.255.255	255.255.255.255	On-link	127.0.0.1	331
192.168.56.0	255.255.255.0	On-link	192.168.56.1	281

Persistent Routes:

None

Some important linux networking Commands

1. ip ip<OPTIONS> <OBJECT> <COMMANDS>

a. # ip address show

Output:

1. lo : <LOOPBACK, UP, LOWER_UP> mtu 65536 qdisc noqueue
   State UNKNOWN group default qlen 1000
2. enp0s3if6 : <NO-CARRIER, BROADCAST, MULTICAST, UP> mtu
   1500 qdisc fq_codel State UP group default qlen 1000
3. wlp0s0 : <BROADCAST, MULTICAST, UP, LOWER_UP> mtu 1500
   qdisc noqueue State UP group default qlen 1000

b. # Sudo ip address add 192.168.1.2/24 dev enp0s3if6
   To assign an IP to an interface

c. # Sudo ip address del 192.168.1.2/24 dev enp0s3if6
   To delete an IP on an interface

d. # Sudo ip link set enp0s3if6 up
   Alter the status of the interface by bringing the interface online

e. # Sudo ip link set enp0s3if6 down
   Alter the status of the interface by bringing the interface offline.

f. # Sudo ip link set enp0s3if6 promisc on
   Alter the Status of the interface by enabling promiscuous mode for interface.

g. #sudo ip route add default via 192.168.1.254 dev enp0s31f6

Add a default route for all addresses via the local gateway 192.168.1.254 that can be reached on device enp0s31f6.

h. # ip route add 192.168.1.0/24 via 192.168.1.254

Add a route to 192.168.1.0/24 via the gateway at 192.168.1.254

i. # sudo ip route add 192.168.1.0/24 dev enp0s31f6

Add a route to 192.168.1.0/24 that can be reached on device enp0s31f6.

j. # sudo ip route delete 192.168.1.0/24 via 192.168.1.254

Delete the route for 192.168.1.0/24 via the gateway at 192.168.1.254

k. # sudo ip route get 10.10.1.4

Display the route taken for IP 10.10.1.4

(2) ifconfig

The ifconfig command was is a staple in many Sys admin's tool belt for configuring and trouble shooting networks.

output:

enp0s31f6: flags=4855<UP, BROADCAST, PROMISC, MULTICAST> mtu 1500

inet: 192.168.1.254 netmask 255.255.255.0 broadcast 0.0.0.0

ether 20:88:10:86:79:89 txqueuelen 1000 (Ethernet)

RX packets 0 bytes 0 (0.0 B)

3. mtr

MTR (Matt's traceroute) is a program with a Command-line interface that serves as a network diagnostic and trouble shooting tool.

Syntax of the command is as follows:

mtr <options> hostname/IP

(a) Basic mtr command shows you the statistics:

#mtr google.com

output: My traceroute [v0.95]

fedora (172.16.75.85) → google.com (142.251.221.206)

2025-07-14T13:42:07-0400

Keys:

Host	Packet Loss%	Pings				
		Snt	Last	Avg	Best	Wrst	StDev
1. IN 115.245.95.245	0.0%	8.3	16.2	14.2	5.8	20.2	24
2. IN 72.14.212.252	0.0%	8.4	6.4	82.8	5.5	283.9	526

(b) mtr -b google.com

Shows the numeric Ip addresses & hostname too

output:

	Packet loss%	Pings				
		Snt	last	Avg	Best	Wrst	StDev
172.16.12.122	0.0%	302	3.8	12.3	2.6	417.5	50.1
172.16.12.122	0.0%	321	6.5	15.5	5.9	368.8	25.3

(4) tcpdump

This command is designed for capturing and displaying packets.

(a) tcpdump -i wlp2s0:

This command captures the traffic on wlp2s0.

output:

dropped privs to tcpdump

tcpdump: verbose output suppressed use -v[v]... for full protocol decode.

listening on wlp2s0, link-type EN10MB (Ethernet).

snapshot length 262144 bytes
23:15 48:819779 ARP Request who-has linca.g
a.ifa

(b) tcpdump -i wlp2s0 -c 10 host 8.8.8.8

To capture traffic to and coming from one specific host

output:

dropped privs to tcpdump

tcpdump: verbose output suppressed. use -v[v]... for full protocol decode

listening on wlp2s0. link-type EN10MB (Ethernet) snapshot length 262144 bytes.

- 0 packets captured.
- 0 packets received by filter.
- 0 packets dropped by kernel.

(c) tcpdump -i wlp2s0 net 10.1.0.0 mask 255.255.255.0

To capture traffic to and from a specific network.

output:

dropped privs to tcpdump

tcpdump: verbose output suppressed, use -v[v]... for full protocol decode

listening on wlp2s0 link-type EN10MB (Ethernet)

- 0 packets captured
- 0 packets received by filter
- 0 packets dropped by kernel.

(d) tcpdump -i wlp2s0 port 53

To capture traffic to and from port number

output:

dropped privs to tcpdump

tcpdump: verbose output suppressed, use -v[v]... for full protocol decode

- Packets captured
- packets received by filter

(5) Ping

It is used to troubleshoot Connectivity reachability and name resolution

#ping google.com

output

PING google.com (142.253.221.206) 56(84) bytes of data.

From fedora (192.168.1.294) icmp_seq=1 Destination Host Unreachable

From fedora (192.168.1.294) icmp_seq=2 Destination Host unreachable.

Student observation:

(1) Which command is used to find the reachability of a host machine from your device?

ping <hostname or IP address>

Eg: ping google.com

(2) which command will give the details of hops taken by a packet to reach its destination?

mtr <hostname>

Eg: mtr google.com

(3) which Command displays the ip configuration of your machine.

On Linux: ipconfig or ip addr
On windows: ipconfig

Eg: ipconfig
ip addr show

(4) which Command displays the TCP port status in your machine?

On Linux: netstat
Eg: netstat -r

(5) Write the modify the ip configuration in a Linux Machine.

To add: Sudo ip address add 192.168.1.100/24 dev enp0s3f6

to delete: Sudo ip address del 192.168.1.100/24 dev enp0s3f6

Result:
The Networking Commands in windows and Linux have been executed Successfully.

4/8/26

21/7/25

Exp No: 2 Study of different types of network cable.

Aim: Study of different types of network cable

1. Unshielded twisted pair (UTP) cable
2. Shielded twisted pair (STP) cable
3. Coaxial cable
4. Fiber optic cable

CABLE TYPE	CATEGORY	MAXIMUM DATA TRANSMISS	ADVANTAGES / DISADVANTAGES	APPLICATIONS
UTP	Category 3	10 bps	Advantages: • cheaper • Easy to install Disadvantages: More prone to EMI	10 Base T Ethernet, Fast Ethernet, Gigabit Ethernet
	Category 5	upto 100 mbps		
	Category 5e	1 Gbps		
STP	Category 6, 6a	10 Gbps	Advantages: • Shielded • Faster than UTP Disadvantages: • Expensive • Greater installation	Gigbit Ethernet 10G Ethernet widely used in data centres
SSTP	Category 7			
Coaxial Cable	RG-6 RG-59 RG-11	10-100 Mbps	Advantages: • High Bandwidth • Immune to interference • Low loss Disadvantages: • limited distance • cost • Size & Bulky	Speed of signals is 200 m Television network high speed internet

Fibre optic cable	Single Mode Multi Mode	100 Gbps	Advantages: • High Speed • High Bandwidth • High Security Disadvantages:- • Expensive • Requires Skilled installer	Maximum distance of fibre optic cable is measured around 100 meter

STUDENT OBSERVATION:

(1) Difference between Cross Cable and Straight Cable.

A Straight cable connects different devices, while cross cable connects the similar devices by swapping transmit and receive wires.

(2) Which type of cable is used by connect two pc?

Cross cable is used to connect two pc's

(3) which cable is used to connect a router to pc?

Straight cable is used to connect a router to a pc.

(4) Find the Category of twisted pair cables used to connect the pc to network socket?

Category 5e and Category 6 are used to connect the pc to network socket.

(5) write down understanding, Challenges faced and output received?

⇒ learnt the colour codes and standards

⇒ Difficulty in arranging wires in correct order

⇒ Successfully made working cable that connected devices and enable proper network communication.

Result :-

Hence, the different types of network cables have been successfully studied.

4/8/16

24/7/25

Exp No: 3 Experiments on CISCO PACKET TRACER (Simulation tool)

Aim:-
To study the Packet tracer tool Installation and User Interface overview.

(a) To Understand Environment of CISCO PACKET TRACER to design simple network.

CISCO Packet Tracer (Simulation Tool) was Successfully Installed

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

1. From the network Component box, click and drag-and drop the below components.
   (a) 4 Generic PCs and one Hub
   (b) 4 Generic PCs and one Switch

2. click on Connections:
   (a) click on Copper Straight-Through Cable.
   (b) Select one of the PC and connect it to Hub using the cable. The link LED should glow in green, indicating that the link is up. Similarly Connect remaining 3 PCs to the Hub.
   (c) Similarly Connect 4 PCs to the Switch using Copper Straight-through cable.

3. Click on the PCs connected to hubs go to Desktop tab, click on Ip Configuration, and enter an Ip address and Subnet mask.

Here the default gateway and DNS Server information is not needed as there are only two end devices in the network.

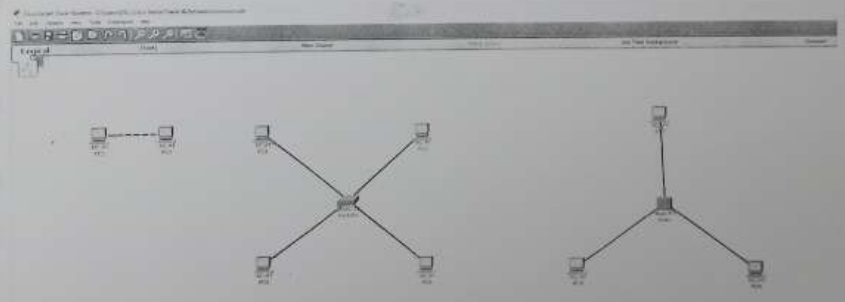
Click on the PDU from the Common tool bar.

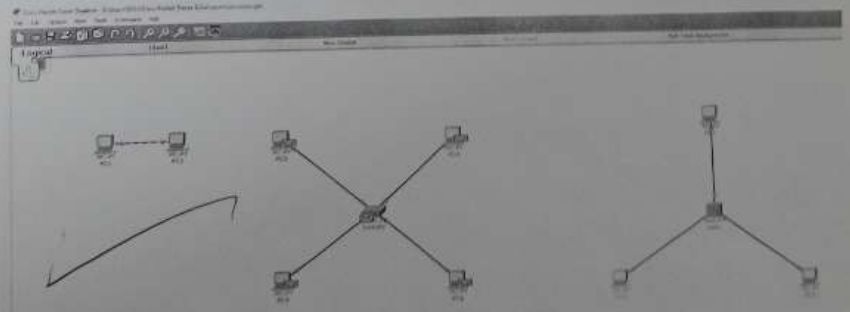
(a) Drag and drop it on one of PC and then drop it on another PC connected to the HUB.

4. Observe the flow of PDU from Source PC to destination PC by selecting the Realtime mode of Simulation.

5. Repeat Step 3 to Step 5 for the PCs connected to the Switch.

6. Observe how HUB and Switch are forwarding the PDU and about the behaviors of Switch and Hub.





Student observation:-

(a) From your observation write down the behaviour of switch and HUB in terms of forwarding the packet received by them.

A Switch forwards packets only to the specific device (port) based on MAC address, while a HUB broadcasts packets to all connected devices.

(b) Find out the network topology implimented in your college and draw and label that topology in your observation.

The network topology commonly used in colleges is Star topology - where all devices are connected to a Central Switch or hub.

Result:-

The packet tracer tool installation and user interface overview is studied.

14/8/26

Experiments on packet capture tool : wireshark

Aim: Experiments on Packet Capture tool : wireshark Exp No: 5

Packet Sniffer:

- Sniffs messages being Sent / Received from / by your Computer.
- Store and display the contents of the various Protocols / fields in the messages.
- Passive program
  - never sends packet 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

Wireshark :-

Wireshark, a network analysis tool formerly known as Ethereal, captures packets in real time and display them in human readable format.

What we can do with wireshark:

- Capture network traffic
- Decode packet protocols using dissectors
- watch smart statistics
- Analyze problems.

Wireshark used for :

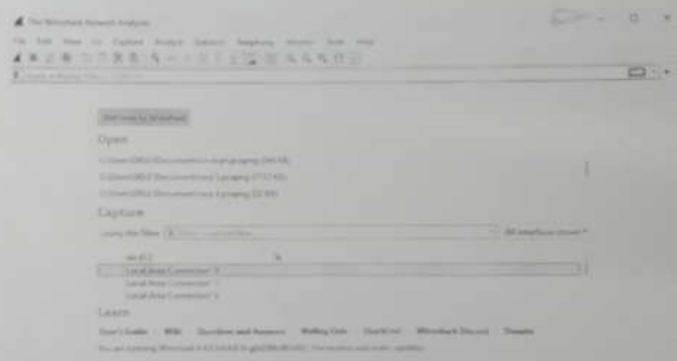
- Network administrators : Troubleshoot network problems
- Developers : Debug protocol implementations
- people : Learn network protocol internals

Getting wireshark :-

Wireshark can be downloaded for window or macOS from its official website.

Capturing packets :-

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



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.

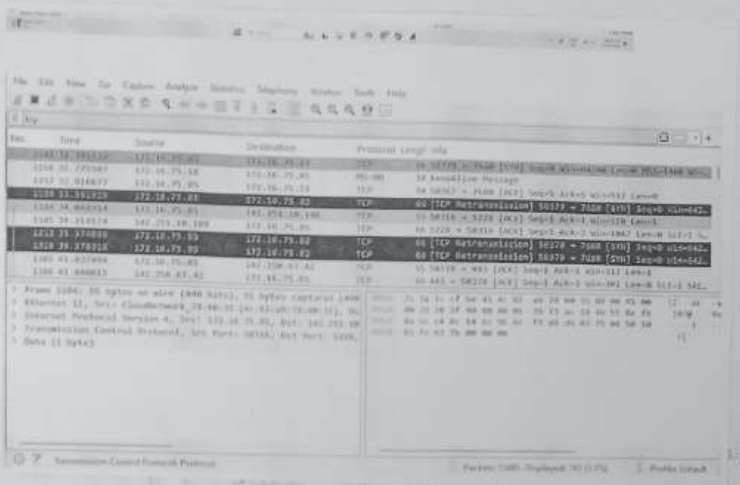
Sample Captures :-

If there's nothing interesting on your own network to inspect, Wirshark's wiki has you covered.

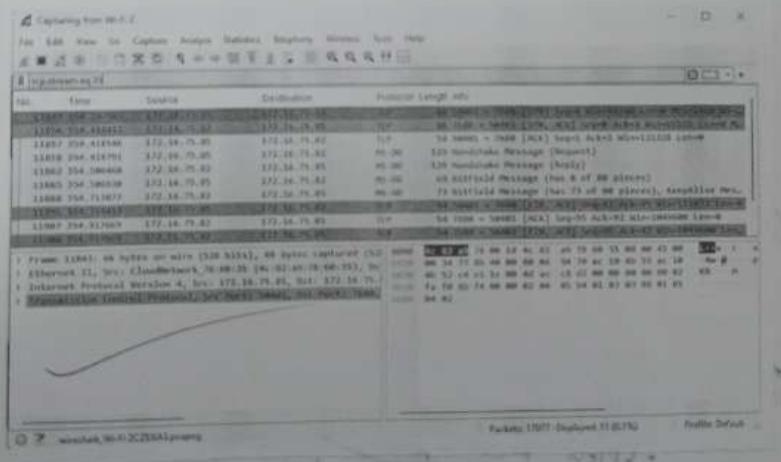
Filtering packets :-

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 network so you can narrow down the traffic

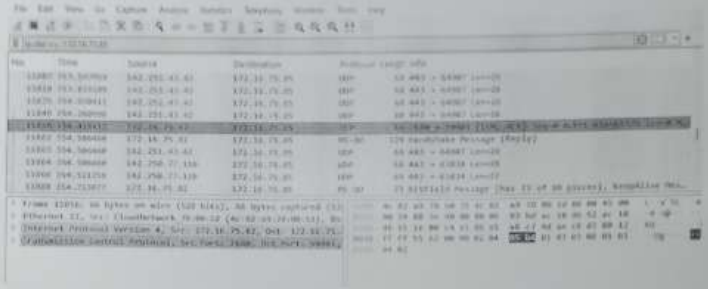
Still you'll likely have a large amount of packets to sift through. That's where wireshark's filters come in.



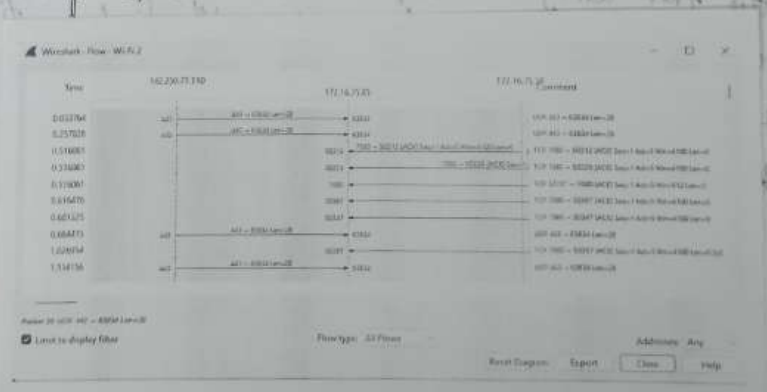
You can also click Analyze > Display Filters to choose a filter from among the default filters included in wireshark. Close the window and you'll find a filter has been applied automatically. Wireshark is showing you the packets that make up the conversation.



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 filters submenu to create a filter based on it.
Flow Graph: Gives a better understanding of what we



Capturing and analysing packets using Wireshark tool :-
To filter, capture, view, packets in Wireshark tool. Capture 10 packets from the ethernet: IEEE 802.3 LAN interface and save it.

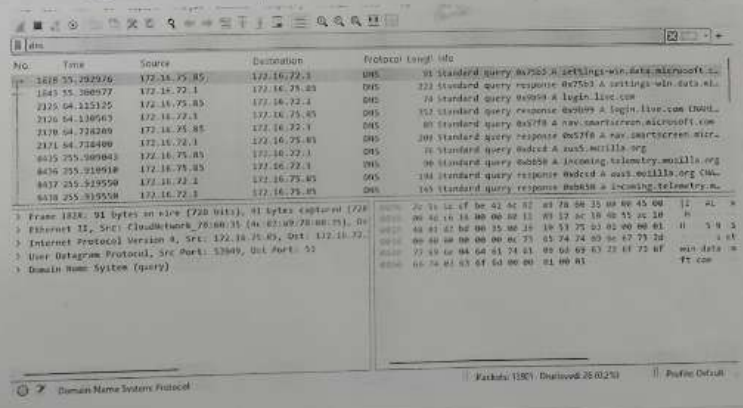
Procedure :-
- Select Local area connection in Wireshark.
- Go to capture → option.
- Select Stop capture automatically after 10 packets
- Then, click Start capture & save the packets.

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

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

3. Create a filter to display only DNS packet & provide the flow graph:

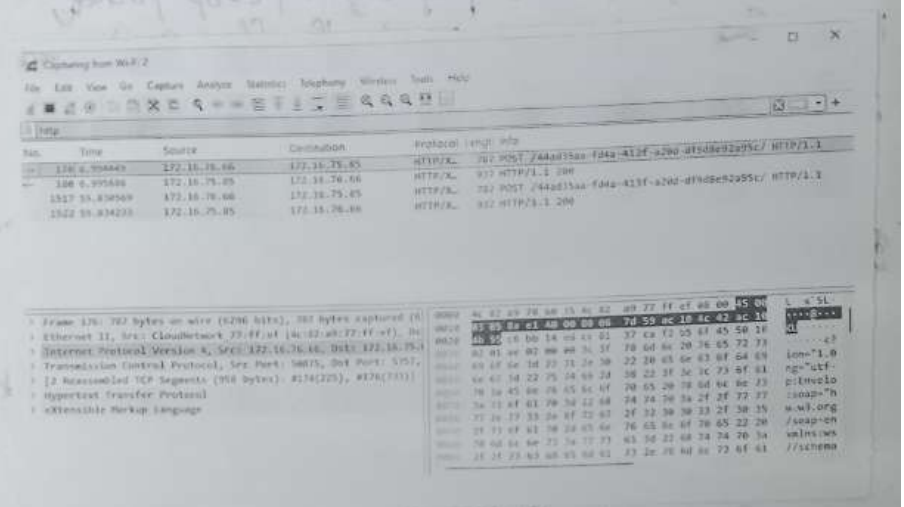
- Go to capture → option
- Select stop capture automatically after two packet
- Then click start capture.
- Search DNS packets in search bar.
- To see flow graph click Statistics → Flow Graph.
- Save the packets.



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

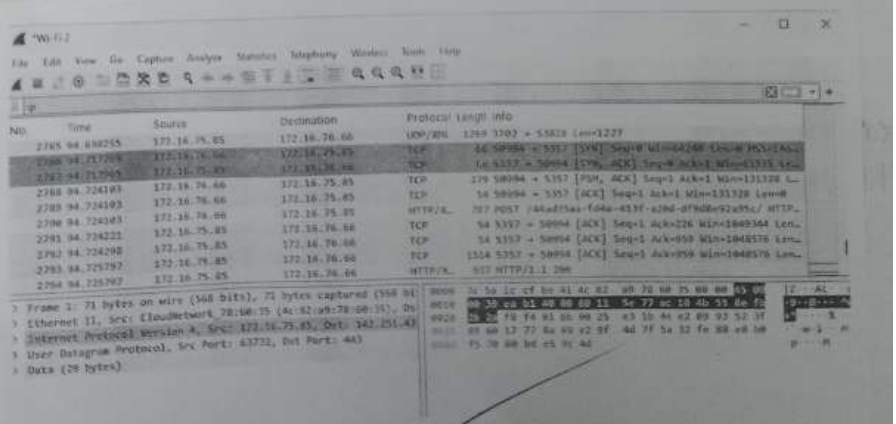
Procedure:

* Select local area connection in wireshark
* Go to Capture → option
* Select Stop capture automatically after two packet
* Then click start capture
* Search HTTP packets and save packets.



5. Create a Filter to display only Ip/ICMP Packets & inspect the packets :-

- Select Local Area connection in Wireshark.
- Go to Capture → option.
- Select Stop Capture automatically after 100 Packet.
- Then click Start capture.
- Search ICMP/IP packets in Search bar.
- Save the packets.



6. Create a filter to display only DHCP Packets and inspect the packets.

Student Observation :-

(1) What is promiscuous mode?

Promiscuous mode is setting for a network interface card (NIC) where it captures all network packet that pass through it, not just the ones addressed to it. It is used in packet sniffing and network monitoring.

(2) Does ARP Packets have transport layer header? Explain.

No, ARP (Address Resolution Protocol) packets do not have a transport layer header. ARP works at Data link layer to map an IP address to a MAC address.

(3) Which Transport layer protocol is used by DNS?

DNS can use : UDP on port 53 for most queries (faster). TCP on port 53 for tasks like zone transfers or responses exceeding 512 bytes

(4) what is the port number used by HTTP protocol?

HTTP uses port 80 (TCP). for secure HTTP (HTTPS), the port is 443 (TCP).

(5) what is broadcast IP address?

A broadcast IP Address is an address used to send data to all hosts in a network simultaneously. In IPv4, it's highest address in a subnet.

Eg: For network 192.168.1.0/24, The broadcast address is 192.168.1.255.

Result :-

Experiments on packet capture tool, wireshark was successfully studied.

14/8/25

Exp No: 4 Setup and Configure LAN

Aim:- Setup and Configure a LAN (Local Area Network) using a Switch and Ethernet Cables in your Lab.

What is a LAN?

A Local Area Network (LAN) refers to a network that connects devices within a limited area, such as an office building, School or home. It enables users to share resources, including data, printers and internet access. LAN connects devices to promote collaboration and transfer information between users, such as computers, Printers, Servers and Switches. A local Area Network (LAN) Switch serves as the primary connecting device, managing and directing communication within the local Network.

How to Set up a LAN

Step 1: plan and design an appropriate network topology taking into account network requirements & equipment location.

Step 2: You can take 4 computers, a Switch with 8, 16 or 24 ports which is sufficient for networks of this sizes and 4 Ethernet Cables

Step 3: Connect your Computers to network Switch via an Ethernet cable, which is as simple as plugging one end of the Ethernet Cable into your Computer and other end to your network Switch.

Step 4: Assign Ip address to your pc's

1. Log on to the client computer as Administrator as owner.
2. click Network and Internet Connections.
3. Right click Local Area Connection / Ethernet → Go to properties

→ Select Internet protocol (TCP/IPv4) → click on Properties → Select use the following ip address option and Assign Ip Address.

Similarly Assign Ip address to all the PCs Connected to Switch.

PC 1 - Ip address: 10.1.1.1, Subnet mask 255.0.0.0.
PC 2 - Ip address: 10.1.1.2, Subnet mask 255.0.0.0.
PC 3 - Ip address: 10.1.1.3, Subnet mask 255.0.0.0.
PC 4 - Ip address: 10.1.1.4, Subnet mask 255.0.0.0.

Step 5: Configure a network Switch:

1. connect your Computer to the Switch: To Access the Switch's web interface, You will need to connect your Computer to the Switch using an Ethernet cable.

2. Login to the web interface: open a web browser and enter the Ip address of the switch in the address bar. This should bring up the login page for the Switch's web interface. Enter the username & password to log in.

3. Configure basic Settings: once you're logged in, you will be able to configure basic settings for the Switch.

4. Assign Ip Address as : 10.1.1.5, Subnet mask: 255.0.0.0

Step 6: check the connectivity between Switch & other machine by using ping command in the Command prompt of the device.

Step 7: Select a folder → go to properties → click Sharing tab → Share it with everyone on the same LAN.

Step 8: Try to access the shared folder from other Computer of the network.

Result:
Hence LAN has been Successfully Setup and Configured.

Ex No: 6
07/08/25

Error correction at Data Link layer

Aim:-

Write a program to implement Error detection & Correction using Hamming code concept.

Sender Program:

Apply Hamming Code concept on the binary data and add redundant bit to it.

```
def hamming_code(data):
    def insert_bits(data):
        m=len(data)
        r=0
        while(2**r)<(m+r+1):
            r+=1
        n=m+r
        result=['0']*n
        j=0
        for i in range(1,n+1):
            if i&(i-1)==0:
                continue
            result[-i]=data[-(j+1)]
            j+=1
            if j==m:
                break
        return result,n,r
    def calc_parity(pdata,r):
        n=len(pdata)
        result=pdata[:]
        for i in range(r):
            parity_val=0
            parity_pos=(2**i)
```

```
    for k in range(1, n+1):
        if k in parity_pos:
            parity_val = int(result[-k])
    result[parity_pos] = str(parity_val)
    return result
pbits, n, r = insert_bits(data)
code = calc_parity(pbits, r)
return ''.join(code)
input = input('Enter binary data: ')
print('Hamming Code =', hamming_code(input))
```

Receiver Program::

→ Apply hamming code on the binary data to check for errors.

→ If there is any error display the position of the error.

```
def hamming_check(hamming_code):
    n = len(hamming_code)
    r = 0
    while (2**r) < n+1:
        r += 1
    syn = 0
    parity = []
    for i in range(r):
        parity_pos = 2**i
        parity_val = 0
    for k in range(1, n+1):
        if k == parity_pos:
            parity_val = int(hamming_code[-k])
```

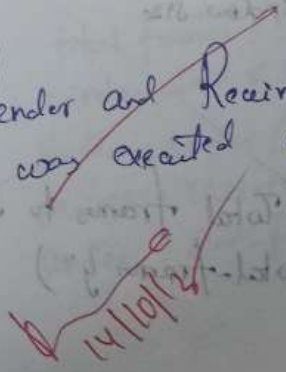
```
            parity.append(parity_val)
            syn = (parity_val << i)
        Synbits = ''.join(str(x) for x in reversed(parity))
        return Synbits, syn

Code = input("Enter Received Hamming Code :")
res, error = hamming.check(code)
print('Error bits :', res)
if error == 0:
    print("No Error detected")
else:
    print("Error detected at bit position :", error)
```

Output :

```
Enter binary data : 1001101
Hamming Code : 10011100101
Enter received Hamming Code : 10010100101
Error Syndrome bits: 0111
Error detected at bit position : 7
```

Result :

Sender and Receiver Program for hamming code concept was executed and got the output.



Ex No: 7
28/8/25

Flow control at Data link layer

Aim:

Write a program to implement flow control at data link layer using Sliding window protocol, Simulate the flow of frames from one node to another.

Features:

→ Input Window size & Message.
→ Sends window size frames at a time.
→ writes frames to Sender Buffer
→ Receiver reads frames, sends ACK or NACK to Receiver Buffer.
→ Sender - reads ACK / NACK and Continues or resends frames.
→ You can manually edit the files to simulate errors

Code:

```
import time
import random
class Sender:
    def __init__(self, total_frames, window_size):
        self.total_frames = total_frames
        self.window_size = window_size
        self.base = 0
        self.next_seq = 0
    def send_frames(self):
        print(f"\n[Sender] Total frames to Send:
                {self.total_frames}")
```

```
while self.base < self.total_frames:
    while self.next_seq < self.base + self.window_size
        and self.next_seq < self.total_frame
        print(f"[Sender] Sending frame {self.next_seq}")
        self.next_seq += 1
        time.sleep(1)
def ack_received(self, ack):
    print(f"[Sender] Acknowledgment received for Frame
                                    {ack}")
    if ack >= self.base:
        self.base = ack + 1

class Receiver:
    def receive_frame(self, frame_no, sender):
        if random.choice([True, False]):
            print(f"[Receiver] Received Frame {frame_no}")
            sender.ack_received(frame_no)
        else:
            print(f"[Receiver] Frame {frame_no} lost
                                (No Ack Sent)")

if __name__ == "__main__":
    total_frames = 5
    window_size = 3
    sender = Sender(total_frames, window_size)
    receiver = Receiver()
    sender = send_frames(receiver)
```

Output:

Enter total number of frames : 5
Enter window size : 3
[Sender] Total frames to Send : 5
[Sender] Sending Frame : 0
[Sender] Sending Frame : 1
[Sender] Sending Frame : 2
[Receiver] Successfully Received frames 0 to 2
[Sender] Acknowledgment receive for frame 2
[Sender] Sending Frame: 3
[Sender] Sending Frame: 4
[Receiver] Frame 4 is Last or Completed
[Sender] Timeout Resending Window from Frame 3
[Sender] Sending Frame : 3
[Sender] Sending Frame : 4
[Receiver] Successfully Received frames 3 to 4
[Sender] Acknowledgment received for frame 4
Transmission Completed.

Result:

Sliding Window protocol is executed Successfully.

EX NO: 8

Experiment on Outlining the process in NMAP Before port Scanning, to find outline System.

Aim:
To attempt the port Scan offline systems and Recognizes the waste time & the created unneeded network (because it is active recon)

The ARP Scan:
This Scan uses ARP Requests to discover live hosts 3.

ICMP Scan:
This Scan uses ICMP Requests to identify live hosts 3.

TCP/UDP ping Scan.
This Scan Sends packets to TCP ports and UDP ports to determine live hosts.

There will be 2 Scanners introduced:
(1) arp-scan
(2) mass Scan

NMAP (Network Mapper) - It is a well known tool for mapping networks, locating live hosts and detecting running services. NMAP's Scripting Engine can be used to extend its capabilities such as fingerprinting services and exploiting flaws. The Scans typically follows the steps represented in the image below, but several are optional and are conditional on the "Command-line options provided prior to the Scan:

Step 1: Enumerate the targets
Step 2: Discover live hosts
Step 3: Reverse DNS Lookup
Step 4: Scan ports
Step 5: Detect versions
Step 6: Detect OS
Step 7: Trace Route
Step 8: Scripts
Step 9: Write output

Result:-
Hence the experiment on outlining the Processes in NMAP port scanning is done successfully.

EX NO: 9
15/9/25

SUBNETTING

AIM: Implementation of connectin 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 default masks for each IP class. This means that we can divide a IP address Space into Smaller Subnets which can be useful when we have a limited no. of addresses but need to create multiple networks.

Creating a Network Topology

The First Step is to create a network topology in Packet Tracer. For that, Select the "New" button in the top left corner, then Select "Network" and "Generic". This will create a blank network topology that we use to add devices.

Adding the Devices

Here, we add routers, Switches and PCs. Select the device & add it onto the network topology. Then, Connect the devices by dragging a cable from one device's port to another device's port.

Subnetting

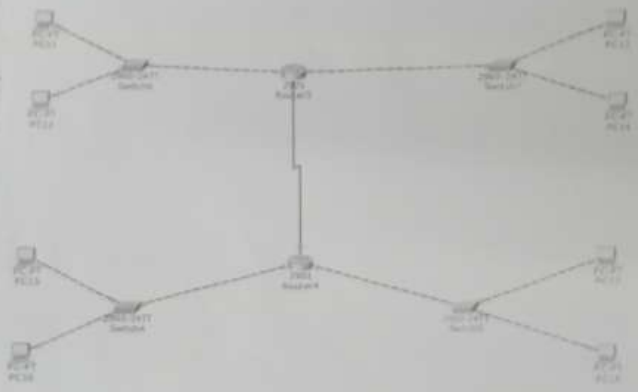
To Subnet the network address of 192.168.1.0/24 to provide enough Space for atleast 5 addresses for end devices, the Switch & the router, we can use a /27 Subnet mask. This will give us 8 Subnets with 30 host addresses each.

Configuring the Devices

Now that we added our devices and connected them, we can start configuring them, we will start by configuring the routers, the switch and then the PCs.

Testing the Network

Now, we can test the network. open a command prompt on each pc and try to ping the other pc. If the ping is successful, the the network is functioning properly.



Student observation

(1) write down your understanding of Subnetting?

Subnetting is dividing a larger network into smaller sub networks for efficient IP Usage and Management.

(2) What is the advantage of implementing Subnetting with Network

It improves IP Utilization, reduces broadcast traffic, enhances Security, & makes networks easier to manage.

Result:

Subnetting was successfully implemented in CisCo Packet Tracer. network devices Communicated properly using the assigned Subnetted IP Addresses.

Ex 10a

AIM:-

(a) Internetworking with routers in CISCO PACKET TRACER Simulator.

Design and Configure a Simple internetwork using a router.

In this network, a router and 2 PCs are used Computers are connected with routers using a Copper Straight-through cable. After forming the network, to check network connectivity a Single PDU is transferred from pc0 to pc1.

Procedure :-

Step-1 (Configuring Router 1):

1. Select the router and Open CLI.
2. Press Enter to Start and Configuring Router 1.
3. Type 'enable' to activate the Privileged mode.

Step-2 (Configuring PCs):

1. Assign IP Addresses to every PC in the network.
2. Select the PC, Go to Desktop and Select IP configuration and assign an IP address, Default gateway Subnet Mask.
3. Assign the default gateway of PC0 as 192.168.10.1.
4. Assign the default gateway of PC1 as 192.168.10.1.

Step-3 (connecting PCs with Router):

1. Connect FastEthernet0 port of PC0 with FastEthernet 0/0 port of Router 1 using a Copper Straight-through cable.
2. Connect FastEthernet0 port of PC1 with FastEthernet 0/1 port of Router 1 using a Copper Straight-through cable.

Router Configuration Table:

Device Name	Ip address FastEthernet 0/0	Subnet Mask	IP address Fast Ethernet 0/1	Subnet Mask
Router1	192.168.10.1	255.255.255.0	192.168.20.1	255.255.255.0

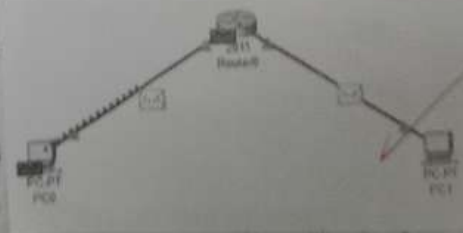
PC configuration Table:

Device Name	Ip Address	Subnet Mask	Gateway
pc0	192.168.10.2	255.255.255.0	192.168.10.1
pc1	192.168.20.2	255.255.255.0	192.168.20.1

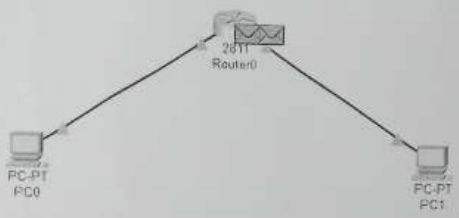
Designed Network Topology:

Simulated of Designed Network Topology:

Sending a PDU From pc0 to pc1:



Acknowledgement from PC1 to PC0 :

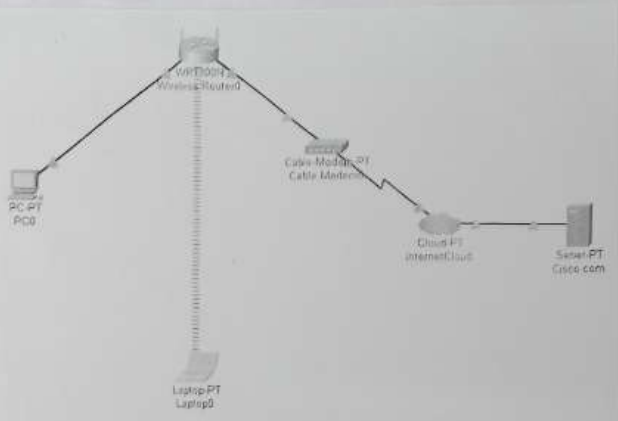


Result :-

The network was successfully interconnected using routers in Cisco Packet Tracer, Enabling communication between different networks.

Ex:No:10 b)

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



Addressing Table:

Device	Interface	Ip address	Subnet Mask	Default Gateway
pc	Ethernet 0	DHCP		192.168.0.1
Wireless Router	LAN	192.168.0.1	255.255.255.0	
Wireless Router	Internet	DHCP		
cisco.com Server	Ethernet 0	208.67.220.220	255.255.255.0	
Laptop	Ethernet 0	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 device

Part 4: Save the file & close packet Tracer.

Part 1:-

Step 1: Launch Packet Tracer

Step 2: Build the Topology.

(a) Add network devices to the workspace.

(b) Change display names of the network devices.

(c) Add the physical cabling between devices on the workspace.

Part 2:-

Step 1:- Configure the wireless router.

(a) Create the wireless network on the wireless router.

(b) Click on the Save Settings tab

Step 2: Configure the Laptop

(a) Configure the Laptop to access the wireless network.

Step 3: Configure the PC.

(a) Configure the PC for the wired network.

Step 4: Configure the Internet cloud.

(a) Install network modules if necessary.

(b) Identify the From and To ports.

(c) Identify the type of provider.

Step 5: Configure the cisco.com Server.

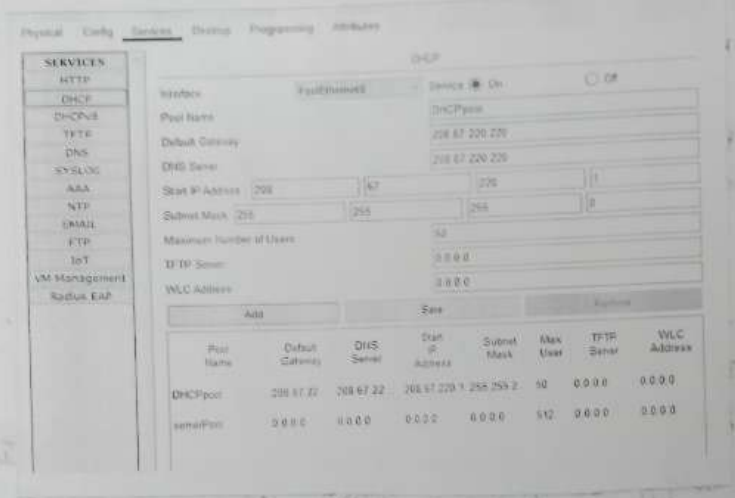
(a) configure the cisco.com Server as a DHCP Server.

(b) configure the cisco.com Server as a DNS Server.

to provide domain name to IPv4 address resolution.

(c) Configure the cisco.com Server Global Settings.

(d) configure the cisco.com Server FastEthernet0 Interface Settings.



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.

(b) Test connectivity to the cisco.com Server from PC.

```
C:\>ping 192.168.0.1

Pinging 192.168.0.1 with 32 bytes of data:

Reply from 192.168.0.1: bytes=32 time=26ms TTL=255
Reply from 192.168.0.1: bytes=32 time=9ms TTL=255
Reply from 192.168.0.1: bytes=32 time=8ms TTL=255
Reply from 192.168.0.1: bytes=32 time=11ms TTL=255

Ping statistics for 192.168.0.1:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 8ms, Maximum = 26ms, Average = 13ms
```

Student observation :-

1) write down the Key Features of configuring Wireless router & DHCP Server.

Wireless router configuration includes setting SSID, Security Key, IP range, and enabling DHCP for automatic IP Assignment.

2) What is the significance of DHCP Server in internetworking?

DHCP Server simplifies internetworking by automatically assigning IP addresses, reducing manual configuration errors.

3) Design & configure an inter-network in your lab using Switch, router and Ethernet cables.

A Network was designed using a router, Switch & PCs connected via Ethernet cables, each device configured with unique IP Addresses for communication.

Result :-

The internetwork was Successfully designed and configured using a wireless router, DHCP Server, and internet cloud.

Ex 11:- Routing at Network Layer

Aim:- (a) Simulate Static Routing Configuration using Cisco Packet Tracer.

The process of adding static routes to the routing table is known as static routing. To understand how to use static routing to create and add a static routes to routing table.

Network Setup:-

* Three routers: Router 0, Router 1, Router 2.
* Each router has directly connected networks & requires static routes for unreachable networks.

Router 0 Configuration:-

* Routes to 30.0.0.0/8 via Router 1 (main) & Router 2 (backup).
* Float Route 30.0.0.0/100 via Router 2 (main) & Router 1 (backup).
* Routes to 20.0.0.0/8 via Router 2 (main) and Router 1 (backup).

Router 1 configuration:-

* Routes to 10.0.0.0/8 via Router 0 (main) and Router 2 (Backup).
* Routes to 400.0.0/8 via Router 0 (main) and Router 2 (Backup).

Router 2 Configuration:-

* Routes to 10.0.0.0/8 and 30.0.0.0/8 networks.

Verification:-

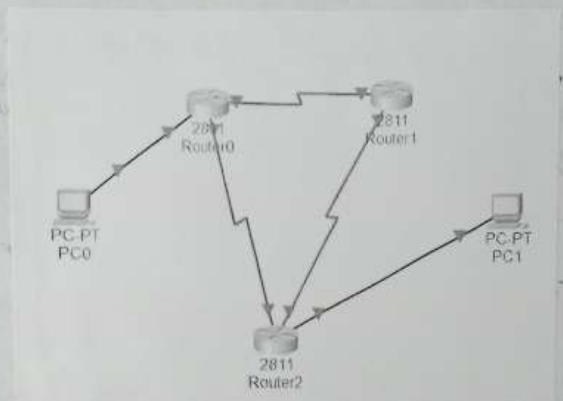
* Show ip route static used to verify routing table entries.
* ping and tracert used to confirm data path.

Failure Simulation:-

* Link between Router 0 and Router 1 removed to trigger failover. * Router switched to backup Route successfully.

Deleting a Static Route:

- Use show ip route static to view routes.
- Remove Route using no ip route command.
- Backup route becomes the new main route if available.




```
Router>enable
Router#configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)#interface s0/0/0
Router(config-if)#ip address 192.168.1.250 255.255.255.252
Router(config-if)#no shutdown

Router(config-if)#
%LINK-5-CHANGED: Interface Serial0/0/0, changed state to up

Router(config-if)#exit
Router(config)#interfa
%LINEPROTO-5-UPDOWN: Line protocol on Interface Serial0/0/0, changed state to up

% Incomplete command.
Router(config)#interface s0/0/1
Router(config-if)#ip address 192.168.1.246 255.255.255.252
Router(config-if)#no shutdown

%LINK-5-CHANGED: Interface Serial0/0/1, changed state to down
Router(config-if)#exit
Router(config)#
%LINK-5-CHANGED: Interface Serial0/0/1, changed state to up

%LINEPROTO-5-UPDOWN: Line protocol on Interface Serial0/0/1, changed state to up

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

1b) RIP Simulation

AIM:- To Simulate RIP (Routing Information Protocol) in Cisco Packet Tracer and verify dynamic routing between routers.

Initial IP Configuration:-

* PCs and routers assigned IP addresses on FastEthernet and Serial interfaces as per topology.
* Serial interfaces on DCE side Configured with clock rate & Bandwidth.
* Interfaces brought up using no Shutdown command.

RIP Configuration:-

* Enable RIP on each router using router rip comma
* Advertise directly connected networks using netwo <network-address> Command.

Network Connectivity Verification:-

* Use ping from PC0 to PC1 to verify end-to-end Connectivity.
* Two routes Exist between PC0 & PC1; RIP Selects the route with least hop count by default.
* Use tracert to verify the path taken by packets.

Dynami Route Failover:-

* Simulated link failure by disconnecting Router Serial 0/0/1 to Router 2 Serial 0/0/1
* RIP automatically rerouted traffic via alternate path (Through Router 1) without manual intervention.

Command Prompt

```
Cisco Packet Tracer PC Command Line 1.0
C:\>ping 10.0.0.2

Pinging 10.0.0.2 with 32 bytes of data:

Reply from 10.0.0.2: bytes=32 time=2ms TTL=126
Reply from 10.0.0.2: bytes=32 time=21ms TTL=126
Reply from 10.0.0.2: bytes=32 time=18ms TTL=126
Reply from 10.0.0.2: bytes=32 time=17ms TTL=126

Ping statistics for 10.0.0.2:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 2ms, Maximum = 21ms, Average = 14ms

C:\>tracert 10.0.0.2

Tracing route to 10.0.0.2 over a maximum of 30 hops:

  1    0 ms      0 ms      0 ms      20.0.0.1
  2    1 ms      2 ms      2 ms      192.168.1.254
  3    0 ms      20 ms     0 ms      10.0.0.2

Trace complete.

C:\>
```

Result :-

Static routing and RIP were successfully configured in Cisco Packet Tracer; connectivity between PC's was verified, and backup/alternate routes worked correctly during link failure.

Ex 12:-

Experiment 12

Aim:- a) Implement echo client Server using TCP/UDP Socket

Algorithm:-

```
import Socket
import time
def ping_Server(host = '127.0.0.1', port = 12345):
    with Socket.Socket(Socket.AF_INET, Socket.SOCK_DGRAM)
                                        as S:

        try:
            S.SendTo(b'Hello', (host, port))
        except S.timeout:
            print("Request timed out")
ping.Server()
```

Result:-

The Message hello is Successfully Sent to the Server.

(b) Implement chat client server using TCP/UDP Sockets.

Algorithm:-

```
import socket
def start_server(host='127.0.0.1', port=12345):
    with socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
            as s:
        s.bind((host, port))
        print(f"UDP Server running on {host}:{port}")
        while True:
            data, addr = s.recvfrom(1024)
            print(f"Received Message from {addr}:
                  {data.decode()}")

start_server()
```

Output:

```
UDP Server running on 127.0.0.1:12345
Received Message from ('127.0.0.1', 52345): Hello
```

Result:-

The Message hello is Successfully Received in the Server

Experiment 13

(a)

Aim:- Implement Your own Ping Program.

Algorithm:-

(1) Create a UDP Socket
(2) Set a timeout of 2 seconds
(3) Record Start time.
(4) Send the message 'ping' to the Server
(5) Wait to receive a response from the Server
> If received, record end time and display the reply with round-trip time.
> If timeout occurs, print "Request time out".
(6) Close the Socket

Input:

```
import Socket
import time

def ping_server(host = '127.0.0.1', port = 12345):
    with Socket.Socket (Socket.AF_INET, Socket.SOCK_DGRAM
                                                    as S:
        try:
            S.SetTimeout(2)
            Start = time.time()
            S.sendto('ping'.(host, port))
            data, addr = S.recvfrom(1024)
            end = time.time()
            print(f"Received {data.decode()} from
                        {addr} in {end.start : .2f} Seconds")
    except socket.timeout:
            Print ("Request time out")
```

```
if __name__ == "__main__":
    ping_serverly
```

Output:

Received ping from ('127.0.0.1', 12345) in 0.00 second

Request timed out

(b) Algorithm:

(1) Create a UDP Socket
(2) Bind the Socket to Server Ip and port
(3) Continuously wait for incoming Messages.
(4) when a Message is Received:
    > Display the Message & Client Address
    > Send back "Pong" to the client.
(5) Repeat Step 3 indefinitely.

Input:

```
import socket
def start_server(host = '127.0.0.1', port = 12345):
    with socket.socket(socket.AF_INET, socket.SOCK_DGRAM
                                        as s:
        s.bind((host, port))
        print(f"UDP Server running on {host}:{port}")
        while True:
            data, addr = s.recvfrom(1024)
            print(f"Received message from {addr}:
                                        {data.decode()}")
            s.sendto(b'pong', addr)

if __name__ == "__main__":
    start_server()
```

O/p:

The Server running on 127.0.0.1:12345
Received message from ('127.0.0.1', 12345): ping

Ex NO: 14

PACKET SNIFFING

AIM:

Write a code using RAW Sockets to implement Packet Sniffing.

Algorithm:-

(1) Start the program.
(2) Capture network packets from the Wi-Fi interface
(3) For each packet received:
  (a) Check if it contains an IP layer.
  (b) Get the Source and destination IP Addresses
  (c) Identify the protocol (TCP/UDP/ICMP).
  (d) Display the protocol name and IP Addresses
(4) Repeat Continuously until Stopped.

INPUT:

```
from Scapy.all import sniff
from Scapy.layers.inet import IP, TCP, UDP, ICMP
def packet_callback(packet):
    if IP in packet:
        ip_layer = packet[IP]
        protocol = ip_layer.proto
        src_ip = ip_layer.src
        dst_ip = ip_layer.dst
        # determine the protocol
        protocol_name =
        if protocol == 1:
            protocol_name = "ICMP"
```

```
elif protocol == 6:
    protocol_name = "TCP"
elif protocol == 17:
    protocol_name = "UDP"
else:
    protocol_name = "Unknown Protocol"
print(f"Protocol: {protocol_name}")
print(f"Source IP: {src_ip}")
print(f"Destination IP: {dst_ip}")
print("-" * 50)

sniff(iface='Wi-Fi', prn=packet_callback, filter="ip", store=0)
```

Output:

```
Protocol: TCP
Source IP: 192.168.1.10
Destination IP: 172.217.167.78
--------------------------------------------------
Protocol: UDP
Source IP: 192.168.1.10
Destination IP: 8.8.8.8
--------------------------------------------------
Protocol: ICMP
Source IP: 192.168.1.10
Destination IP: 192.168.1.1
```

Result:

Packet Sniffing using Raw Sockets has been successfully implemented.