

**VISVESVARAYA TECHNOLOGICAL UNIVERSITY**  
“JnanaSangama”, Belgaum -590014, Karnataka.



**LAB REPORT**  
on  
**COMPUTER NETWORKS**

*Submitted by*

**DIKSHYA ARYAL (1BM21CS058)**

*in partial fulfillment for the award of the degree of*  
**BACHELOR OF ENGINEERING**  
*in*  
**COMPUTER SCIENCE AND ENGINEERING**



**B.M.S. COLLEGE OF ENGINEERING**  
(Autonomous Institution under VTU)  
**BENGALURU-560019**  
**JUN-2023 to SEP-2023**

**B. M. S. College of Engineering,  
Bull Temple Road, Bangalore 560019**  
(Affiliated To Visvesvaraya Technological University, Belgaum)  
**Department of Computer Science and Engineering**



**CERTIFICATE**

This is to certify that the Lab work entitled “COMPUTER NETWORKS” carried out by **DIKSHYA ARYAL (1BM21CS058)**, who is a bonafide student of **B. M. S. College of Engineering**. It is in partial fulfillment for the award of **Bachelor of Engineering in Computer Science and Engineering** of the Visvesvaraya Technological University, Belgaum during the year 2023. The Lab report has been approved as it satisfies the academic requirements in respect of a **Computer Networks - (22CS4PCCON)** work prescribed for the said degree.

**Swathi Sridharan**  
Assistant Professor  
Department of CSE  
BMSCE, Bengaluru

**Dr. Jyothi S Nayak**  
Professor and Head  
Department of CSE  
BMSCE, Bengaluru

# Index

<b>Sl. No.</b>	<b>Date</b>	<b>Experiment Title</b>	<b>Page No.</b>
<b>CYCLE 1</b>			
1	15/6/23	Create a topology and simulate sending a simple PDU from source to destination using hub and switch as connecting devices and demonstrating ping messages.	4
2	22/6/23	Configure IP address to routers in packet tracer. Explore the following messages: ping responses, destination unreachable, request timed out, reply.	10
3	13/7/23	Configure default route, static route to the Router.	18
4	13/7/23	Configure DHCP within a LAN and outside LAN.	23
5	20/7/23	Configure Web Server, DNS within a LAN.	30
6	20/7/23	Configure RIP routing Protocol in Routers.	33
7	27/7/23	Configure OSPF routing protocol.	38
8	3/8/23	To construct simple LAN and understand the concept and operation of Address Resolution Protocol (ARP).	43
9	10/8/23	To construct a VLAN and make a pc communicate among VLAN.	47
10	10/8/23	Demonstrate the TTL/ Life of a Packet.	51
11	10/8/23	To construct a WLAN and make the nodes communicate wirelessly.	55
12	10/8/23	To understand the operation of TELNET by accessing the router in server room from a PC in IT office.	60
<b>CYCLE 2</b>			
13	17/8/23	Write a program for error detecting code using CRC CCITT (16-bits).	63
14	17/8/23	Write a program for congestion control using Leaky bucket algorithm.	67
15	24/8/23	Using TCP/IP sockets, write a client-server program to make client sending the file name and the server to send back the contents of the requested file if present.	70
16	24/8/23	Using UDP sockets, write a client-server program to make client sending the file name and the server to send back the contents of the requested file if present.	73
17	24/8/23	Tool Exploration -Wireshark	75

# WEEK 1

Create a topology and simulate sending a simple PDU from source to destination using hub and switch as connecting devices and demonstrate ping messages.

OBSERVATION:

Date / /  
Page 4

Create a topology and simulate sending a simple PDU from source to destination using a simple HUB and switch as connecting devices.

→ Aim  
To create a topology and simulate sending a simple PDU from source to destination using a simple HUB and switch as connecting device.

Procedure.

1. Start creating the topology.
2. Select the HUB in the logical workspace and connect three PC's to the HUB generic HUB through copper wire.
3. Set the IP address for all 3 PC's by clicking on PC.
4. Similarly select the going to config and typing the needed ip address.

Fig: PC's connected with HUB.

1. Similarly a switch is selected in the logical workspace and connect 3 PC's to the generic switch - PT through copper st. wire.
2. Set the IP address for all 3 PC's by clicking on PC, going to config and typing the needed ip address.

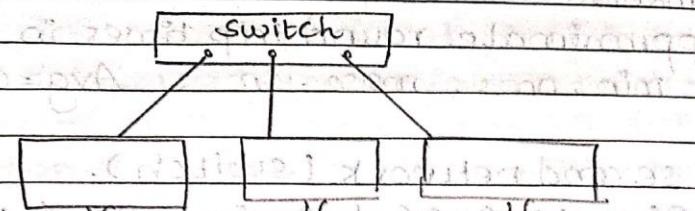


Fig: PC's connected with switch.

6. Now connect the HUB and the switch with the help of a copper-cross over wire. This completes the designing of the required topology.

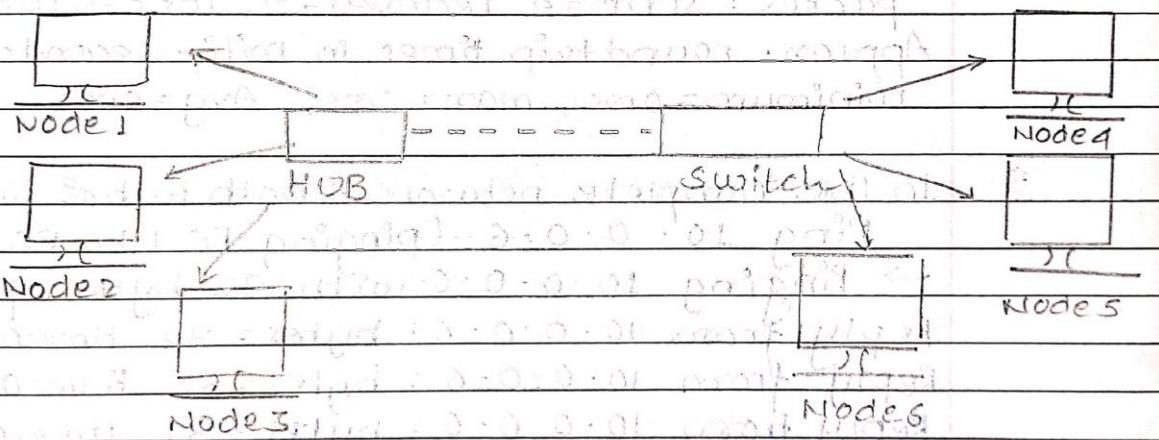


Fig: The required topology.

Output:-

- In first network (HUB)
  - ping 10.0.0.3 (Pinged PC 1 with PC 3)
  - Pinging 10.0.0.3 with 82 bytes of data

Reply from 10.0.0.3: bytes=32 time=0ms TTL=128

Pinging statistics for 10.0.0.30 with 1000

6  
 Packets: sent = 4, received = 4, lost = 0 (0% loss).  
 Approximate round trip times in milliseconds:  
 min = 0ms, max = 0ms, Avg = 0ms.

2. In second network (switch).

Ping 10.0.0.6 (pinging PC 4 to PC 6)

→ Pinging 10.0.0.6 with 32 bytes of data.

Reply from 10.0.0.6: bytes = 32 time = 0ms TTL = 128

Reply from 10.0.0.6: bytes = 32 time = 0ms TTL = 128

Reply from 10.0.0.6: bytes = 32 time = 0ms TTL = 128

Reply from 10.0.0.6: bytes = 32 time = 2ms TTL = 128

Ping statistics for 10.0.0.6

packets: sent = 4, received = 4, lost = 0 (0% loss),

Approx. round trip times in milliseconds:

minimum = 0ms, max = 2ms, Avg = 0ms.

3. In the complete network (both hub & switch).

Ping 10.0.0.6 (pinging PC 1 to PC 6)

→ Pinging 10.0.0.6 with 32 bytes of data.

Reply from 10.0.0.6: bytes = 32 time = 0ms TTL = 128

Reply from 10.0.0.6: bytes = 32 time = 0ms TTL = 128

Reply from 10.0.0.6: bytes = 32 time = 0ms TTL = 128

Reply from 10.0.0.6: bytes = 32 time = 0ms TTL = 128

Ping stats for 10.0.0.6

packets: sent = 4, received = 4, lost = 0 (0% loss)

Approximate round trip times in milliseconds:

minimum = 0ms, max = 0ms, Avg = 0ms.

Q3:

Observation: when a packet is sent from PC 1 to PC 3 that

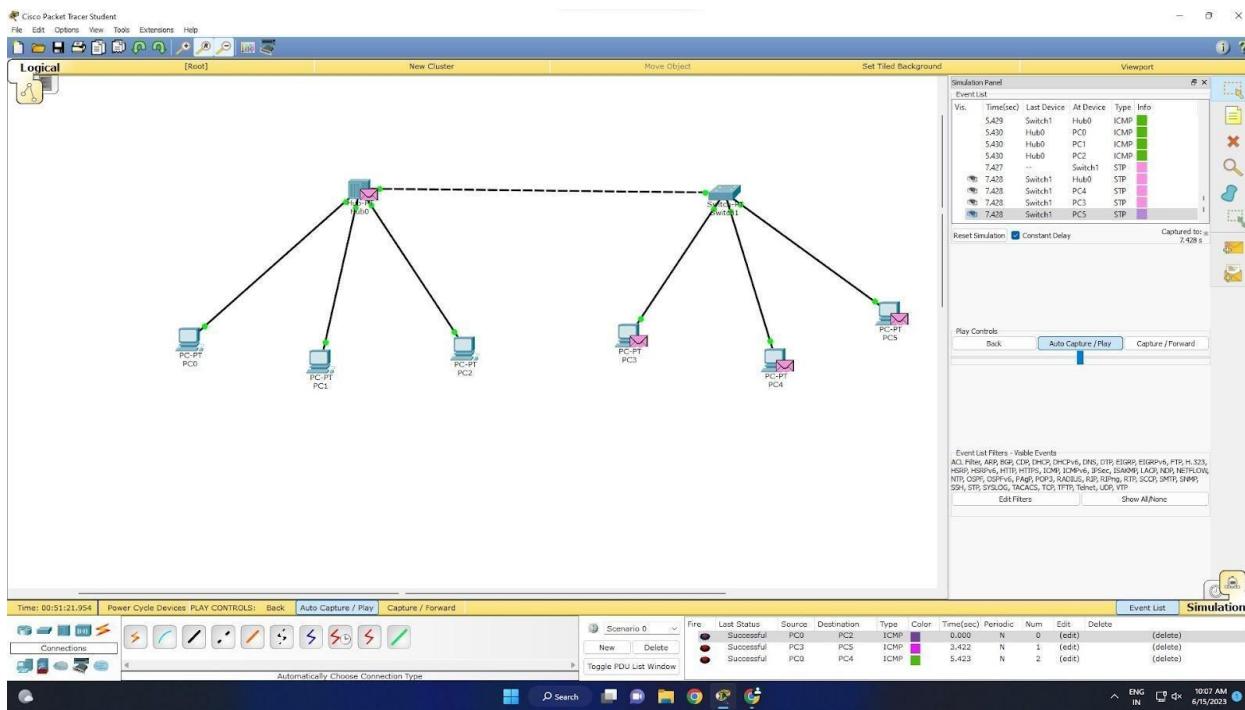
are connected to the Hub it is broadcasted to

all the PC, but acknowledgement is received from only the addressed PC.

2. When PC -> Packet is sent from PC 4 to PC 6 connected to a switch # at first it is broadcasted to all and acknowledgement is only received from the addressed PC but from next time it is only unicasted that is sent to the add. PC.
3. When packet is sent from PC 1 to PC 6 that are connected to HUB and switch, which are also connected to each other the packet is sent to all PC connected to HUB but no ackno. is received and it is only sent to the addressed PC through switch. and ack. is also received only for it.

15/1/2023

## TOPOLOGY:



## OUTPUT:

```

PC0> ping 192.160.1.5

Pinging 192.160.1.5 with 32 bytes of data:
Reply from 192.160.1.5: bytes=32 time=0ms TTL=128

Ping statistics for 192.160.1.5:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 0ms, Maximum = 0ms, Average = 0ms

PC0> ping 192.160.1.5

Pinging 192.160.1.5 with 32 bytes of data:
Request timed out.

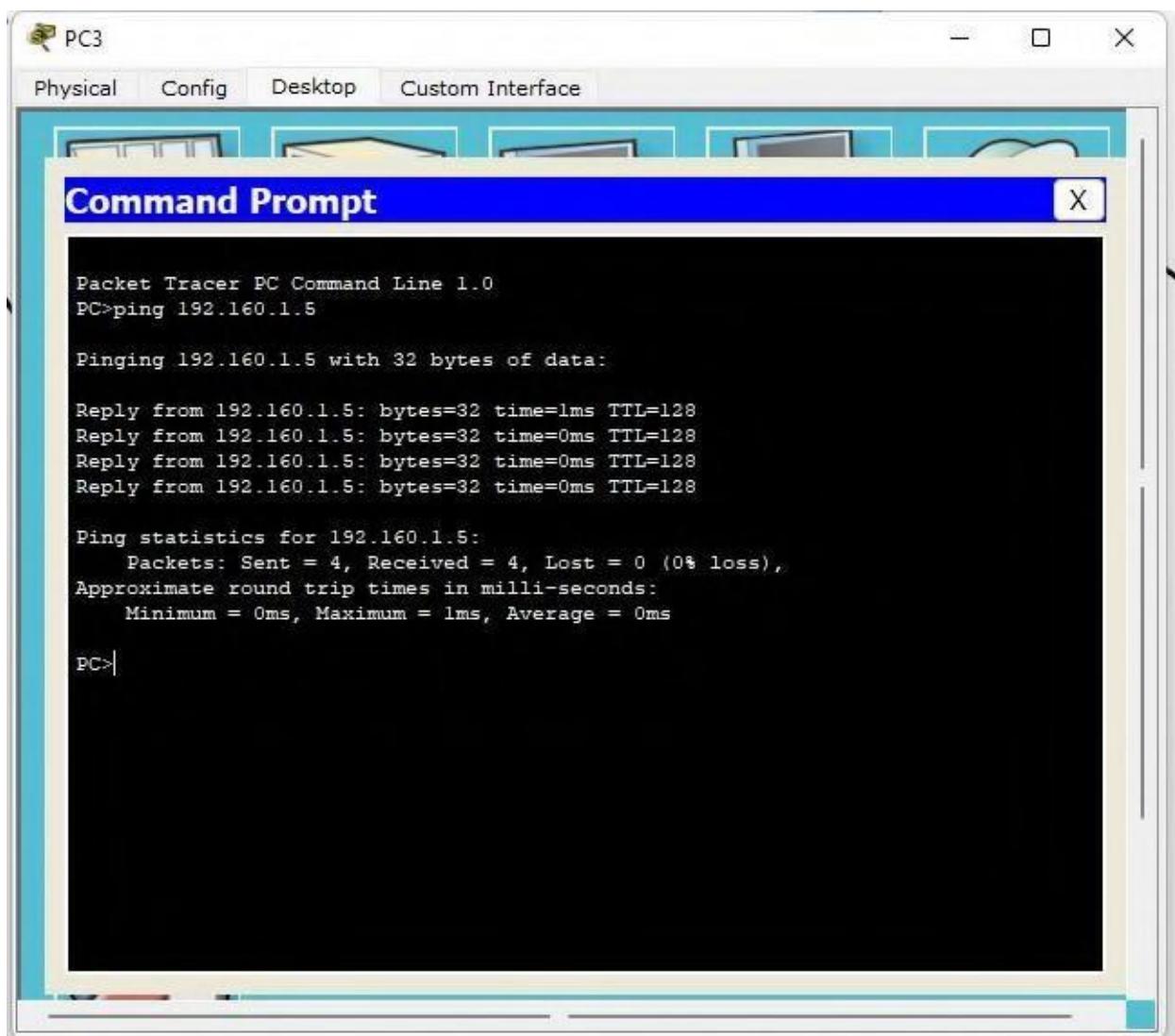
PC0> ping 192.160.1.5

Pinging 192.160.1.5 with 32 bytes of data:
Reply from 192.160.1.5: bytes=32 time=0ms TTL=128

Ping statistics for 192.160.1.5:
    Packets: Sent = 4, Received = 0, Lost = 4 (100% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 0ms, Maximum = 0ms, Average = 0ms

PC0>

```



## WEEK 2

Configure IP address to routers (one and three) in packet tracer. Explore the following messages: ping responses, destination unreachable, request timed out, reply.

### OBSERVATION:

Date 22/06/2023  
Page 8

Configure IP address to routers in packet tracer.  
Explore the following messages: ping responses, destination unreachable, request timed out, reply.  
Solution: '2a' configuration of basic config.

Aim: To config configure IP address to routers in packet tracer and get ping responses - timed out, reply.

Topology:

```
graph LR; Router0[Router 0] --- PC0[PC0]; Router0 --- PC1[PC1]; Router0[Router 0] -- "IP 10.0.0.10" --- PC0[PC0]; Router0[Router 0] -- "IP 20.0.0.20" --- PC1[PC1]
```

Procedure:

- 2 PC's are connected to a router using copper cross-over.
- IP addresses are set for PC's and router.
- IP address for routers is set by giving following commands:  

```
router >enable  
router #config  
router (config)# interface fastethernet 0/0  
router (config-if)# ip address 10.0.0.10  
255.0.0.0  
router (config-if)# no shut  
router (config-if)# exit  
router (config)# interface fastethernet 1/0  
router (config-if)# ip address 20.0.0.10  
255.0.0.0  
router (config-if)# no shut  
router (config-if)# exit
```

- After all IP are set, ping message is sent.

Result: PC > ping 20.0.0.10

Pinging 20.0.0.10 with 32 bytes of data

Request timed out.

Reply from 20.0.0.10 bytes = 32 time = 0ms TTL = 127

Reply from 20.0.0.10 bytes = 32 time = 0ms TTL = 127

Reply from 20.0.0.10 bytes = 32 time = 0ms TTL = 127

Ping statistics for 20.0.0.10.

Packet sent = 4, received = 3, lost = 1 (25% loss),  
approximate round trip times in milli-seconds.

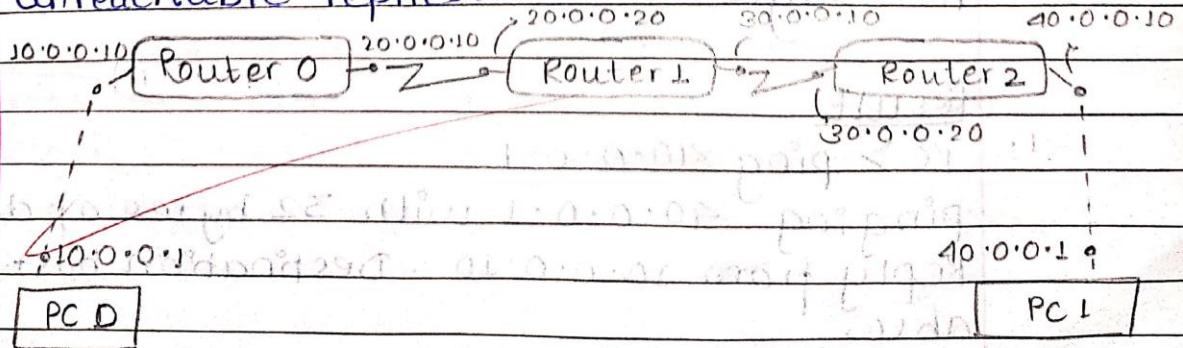
Minimum = 0ms, Maximum = 0ms, Average = 0ms.

### Observation

PC 0 is in network 10.0.0.0 and PC 1 is in 20.0.0.0. Hence we use router to connect them. When a ping message is sent from PC 10.0.0.1 to 20.0.0.1, if the message reaches the destination through router. When a message is sent, the router captures it and sends to the destination PC which is in another network.

### Solution: 2b'

Aim: To configure IP address to routers in packet tracer and get ping responses, destination host unreachable replies.



### ~~Procedure~~

- connect to corresponding routers using copper cross-over.
- connect routers using serial-DCE.
- set IP address for PC's
- configure IP address to routers by giving commands in CLI.
- After all IP's are set, ping PC to get destination host unreachable message.
- route the IP's to the adjacent IP's using following command -

for router 0 - router(config) # ip route 30.0.0.0  
255.0.0.0 20.0.0.20

router(config) # ip route 40.0.0.0  
255.0.0.0 20.0.0.20

for router 1 - router(config) # ip route 10.0.0.0  
255.0.0.0 20.0.0.10

- router(config) # ip route 40.0.0.0  
255.0.0.0 20.0.0.20

for router 2 - router(config) # ip route 10.0.0.0  
255.0.0.0 20.0.0.10

router(config) # ip route 20.0.0.0  
255.0.0.0 20.0.0.10

- After this is done, ping pc to get reply messages.

### Result

1. PC > ping 40.0.0.1

Pinging 40.0.0.1 with 32 bytes of data.

Reply from 10.0.0.10: Destination host unreachable.

Reply from 10.0.0.10: Destination host unreachable.

Reply

Reply

Ping

pa

2. PC > Pi

Pingin

Reply

Reply

Reply

Reply

Ping si

Packet

Approu  
Minimu

Observa

PC 0 is  
40.0.0.

initially

0 and 4

sent fro

the dest

After

adjacent

message

desired

destinati

Reply from 10.0.0.10 Destination host unreachable.  
Reply from 10.0.0.10 Destination host unreachable.  
Ping statistics for 10.0.0.1:  
Packets: sent = 4, Received = 0, Lost = 4 (100% loss)

## 2. PC > Ping 10.0.0.1

Pinging 10.0.0.1 with 32 bytes of data.

Reply from 10.0.0.10 bytes = 32 time = 16ms TTL = 128  
Reply from 10.0.0.10 bytes = 32 time = 1ms TTL = 128  
Reply from 10.0.0.10 bytes = 32 time = 2ms TTL = 128  
Reply from 10.0.0.10 bytes = 32 time = 4ms TTL = 128  
Ping statistics for 10.0.0.1:

Packets sent = 4, received = 4, loss = 0 (0% loss)

Approximate round trip times in milli-seconds.

Minimum = 1ms, maximum = 10ms, average = 6ms.

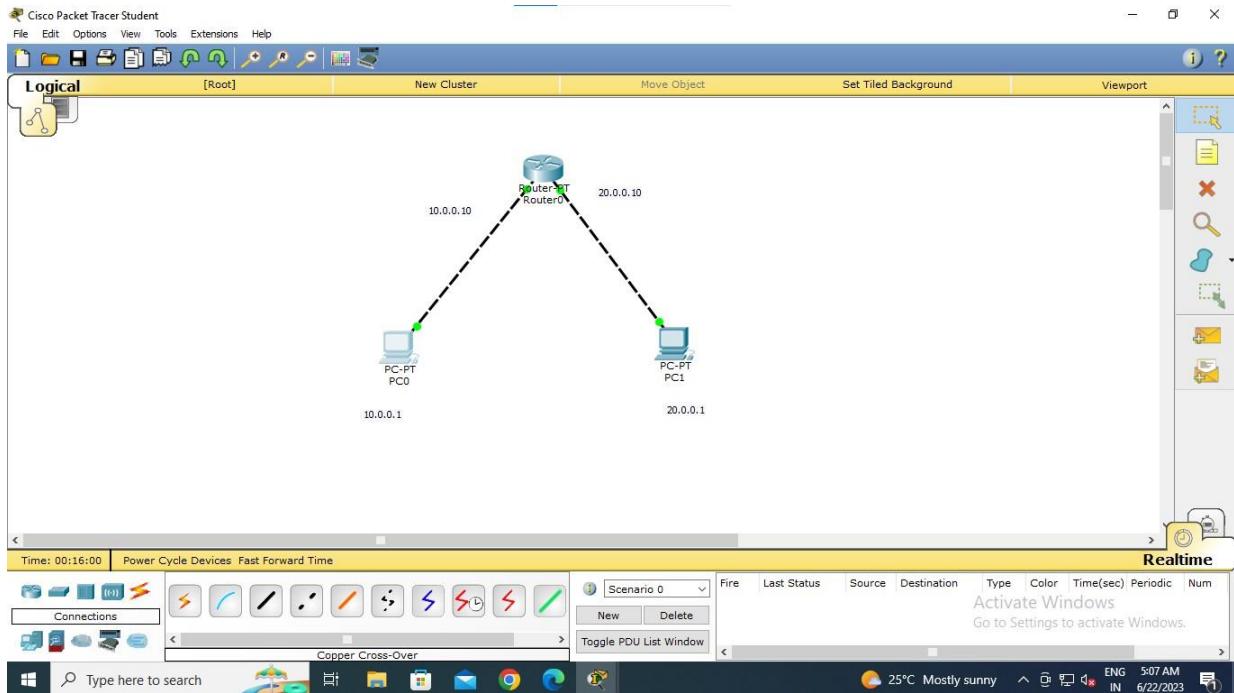
## Observations

PC 0 is in network 10.0.0.0 and PC 1 is in network 10.0.0.0. There are 3 routers in between which initially directly connect 10.0.0.0, 20.0.0.0, 30.0.0.0 and 40.0.0.0. Hence when a ping message is sent from 10.0.0.1 to 40.0.0.1, it doesn't reach the destination. Instead it only reaches the first router and gives destination host unreachable msg.

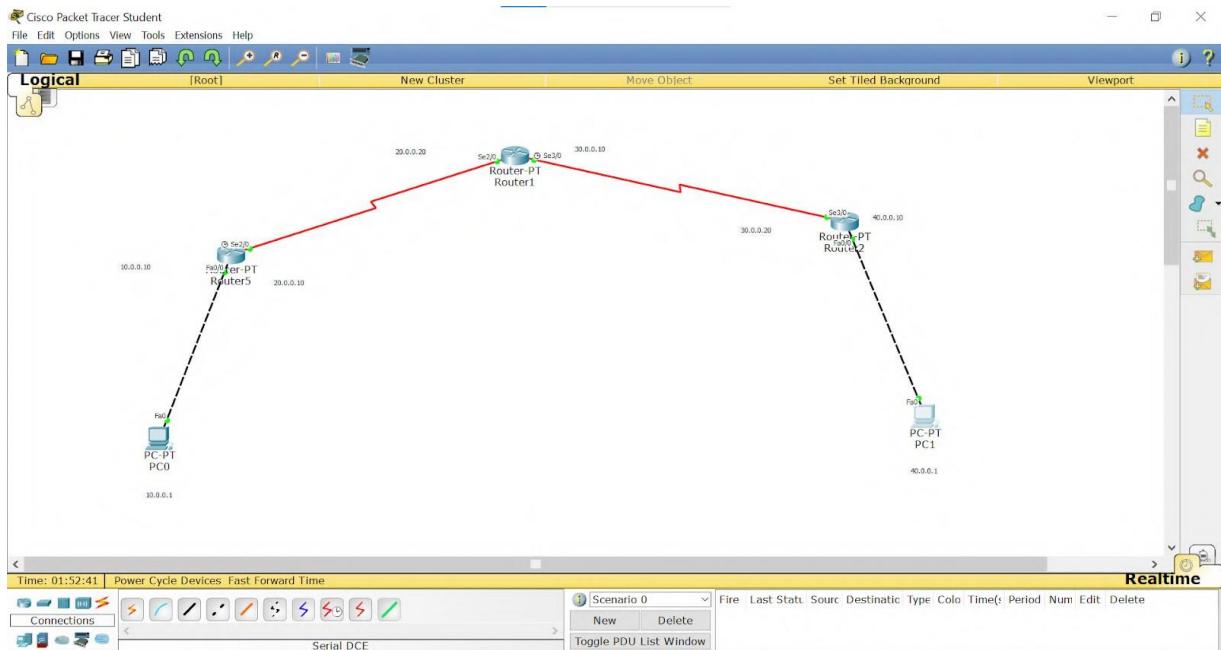
After letting the routers know about other adjacent networks, (next hop) we send a ping message from 10.0.0.1 to 10.0.0.1 to get desired result. The message reaches the destination.

## TOPOLOGY:

### PROGRAM 2.1



### PROGRAM 2.2



## OUTPUT:

### PROGRAM 2.1

The screenshot shows a Cisco Packet Tracer interface with two windows. The top window is a 'Command Prompt' showing the output of a ping command from PC0 to 20.0.0.1. The bottom window is a 'Logical' view of a network topology with three routers (Router0, Router1, Router2) and two hosts (PC0 and PC1). The network diagram includes IP addresses 10.0.0.10, 20.0.0.10, and 20.0.0.1. The Event List panel on the right shows several CDP events. The Taskbar at the bottom shows the Windows Start button, a search bar, and system icons.

```
Packet Tracer PC Command Line 1.0
PC>ping 20.0.0.1

Pinging 20.0.0.1 with 32 bytes of data:

Request timed out.
Reply from 20.0.0.1: bytes=32 time=0ms TTL=127
Reply from 20.0.0.1: bytes=32 time=0ms TTL=127
Reply from 20.0.0.1: bytes=32 time=10ms TTL=127

Ping statistics for 20.0.0.1:
  Packets: Sent = 4, Received = 3, Lost = 1 (25% loss),
Approximate round trip times in milli-seconds:
  Minimum = 0ms, Maximum = 10ms, Average = 3ms

PC>
```

Cisco Packet Tracer Student  
File Edit Options View Tools Extensions Help

Logical [Root] New Cluster Move Object Set Tiled Background Viewport

Event List Panel  
Event List  
Vis. Time(sec) Last Device At Device Type Info  
465.354 Router0 PC1 CDP  
525.353 -- Router0 CDP  
525.353 -- Router0 CDP  
525.354 Router0 PC0 CDP  
525.354 Router0 PC1 CDP  
585.355 -- Router0 CDP  
585.355 -- Router0 CDP  
585.356 Router0 PC0 CDP  
585.356 Router0 PC1 CDP

Reset Simulation  Constant Delay Captured to: 585.356 s

Play Controls Back Auto Capture / Play Capture / Forward

Event List Filters - Visible Events  
ACL, Filter, AVO, BGP, CDP, DHCP, DHCPv6, DNS, DTP, EIGRP, EIGRPv6, FTP, H.323, HSRP, HSRPv6, HTTPS, ICMP, ICMPv6, IPSec, ISAKMP, LACP, NDP, NETFLOW, NTP, OSPF, OSPFv6, PAgP, POP3, RADIUS, RIP, RIPng, RTP, SCCP, SMTP, SNMP, SSH, STP, SYSLOG, TACACS, TCP, TFTP, Telnet, UDP, VTP

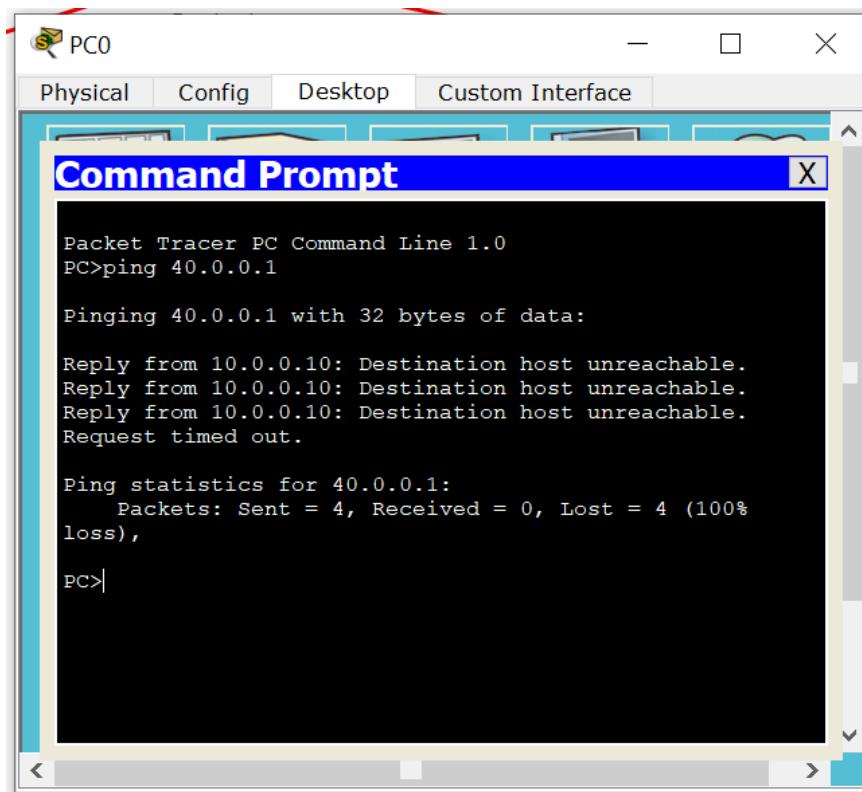
Time: 00:27:16.137 Power Cycle Devices PLAY CONTROLS: Back Auto Capture / Play Capture / Forward Scenario 0 Connections Scenario 0 New Delete Toggle PDU List Window

Fire Last Status Source Destination Type Color Time(sec) Periodic Num  
Successful PC0 PC1 ICMP 0.000 N 0

Go to Settings to activate Windows.

25°C Mostly sunny 5:10 AM ENG IN 6/22/2023

## PROGRAM 2.2



PC0

Physical Config Desktop Custom Interface

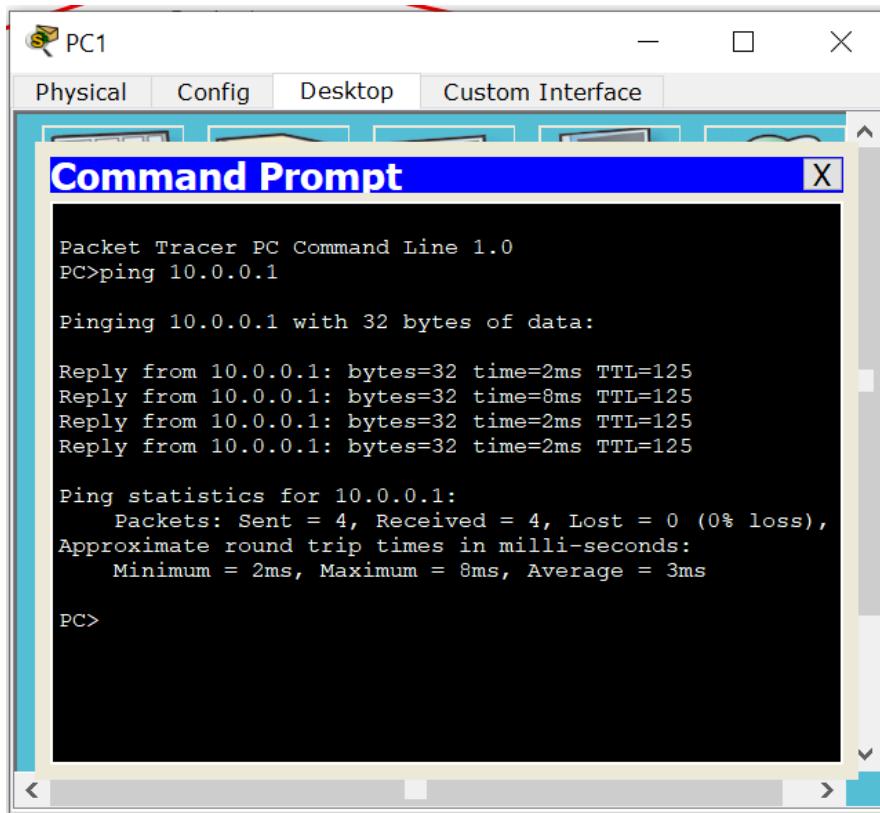
**Command Prompt**

```
Packet Tracer PC Command Line 1.0
PC>ping 40.0.0.1

Pinging 40.0.0.1 with 32 bytes of data:

Reply from 10.0.0.10: Destination host unreachable.
Reply from 10.0.0.10: Destination host unreachable.
Reply from 10.0.0.10: Destination host unreachable.
Request timed out.

Ping statistics for 40.0.0.1:
    Packets: Sent = 4, Received = 0, Lost = 4 (100% loss),
PC>
```



PC1

Physical Config Desktop Custom Interface

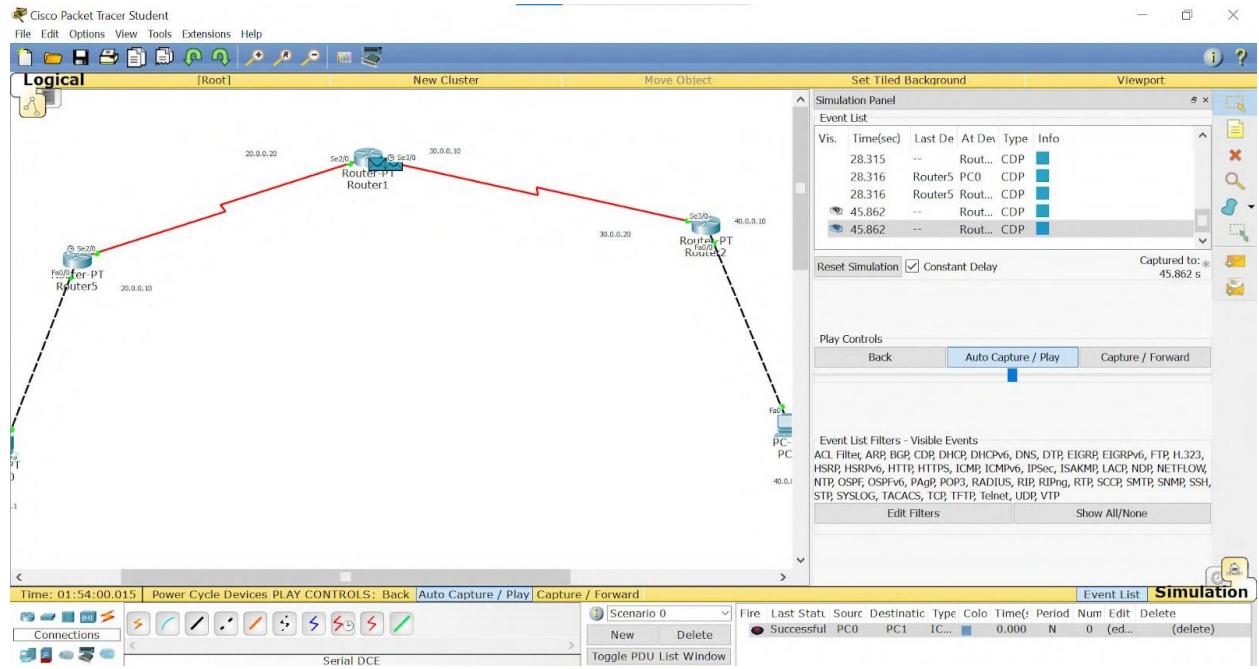
**Command Prompt**

```
Packet Tracer PC Command Line 1.0
PC>ping 10.0.0.1

Pinging 10.0.0.1 with 32 bytes of data:

Reply from 10.0.0.1: bytes=32 time=2ms TTL=125
Reply from 10.0.0.1: bytes=32 time=8ms TTL=125
Reply from 10.0.0.1: bytes=32 time=2ms TTL=125
Reply from 10.0.0.1: bytes=32 time=2ms TTL=125

Ping statistics for 10.0.0.1:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 2ms, Maximum = 8ms, Average = 3ms
PC>
```



## WEEK 3

Configure default route, static route to the Router.

OBSERVATION:

Date 19/7/23  
Page 12

**Configure default and static route for a connection of routers.**

**Aim:** Configure default and static route for a connection of routers.

**Topology:**

```
graph LR; Router0[Router 0] --- Serial[serial cable] --- Router1[Router 1]; Router1 --- FE1[Fast Ethernet 1] --- PC0[PC 0]; Router1 --- FE2[Fast Ethernet 2] --- Router2[Router 2]; Router2 --- FE3[Fast Ethernet 3] --- PC1[PC 1]; Router2 --- FE4[Fast Ethernet 4] --- Router0;
```

**Procedure.**

1. Select 3 generic routers and two PCs as end devices. Connect the PCs to different routers with copper cross-over and connect both the routers to the main router with serial cable.
2. Set IP address for PC and gateways.
3. Set the gateway IP addresses in all the routers taking as fast ethernet for the PCs and serial for routers.
4. Connect the PC's to the interfaces.
5. Config - steps:

```
>enable (router 1)
# config
# interface fastethernet 0/0
# ip address 10.0.0.10 255.0.0.0
# no shut
# exit
# interface serial 2/0.
```

# IP address 20.0.0.1 255.0.0.0

# no shut

# exit.

similarly for router - 0

> enable

# config t

# interface serial 2/0

# ip address 20.0.0.2 255.0.0.0

# no shut

# exit

# interface serial 3/0

# ip address 30.0.0.1 255.0.0.0

# no shut

# exit.

for router - 2

> enable

# config t

# interface fastethernet 0/0

# ip address 40.0.0.10 255.0.0.0

# no shut

# exit

# interface serial 2/0

# ip address 30.0.0.2 255.0.0.0

# no shut

# exit

6. we need to set IP routes for all routes via routers.

For router - 1 & router - 2, we do default routing  
and for router - 0, static routing is done.

For router - 1

# config t

# ip route 0.0.0.0 0.0.0.0 20.0.0.2

# no shut

#exit

show ip route

C 10.0.0.0/8 is directly connected, fastethernet0/0

C 20.0.0.0/8 is directly connected, serial 2/0

S 0.0.0.0/0 [1/0] via 20.0.0.2

Similarly for router 2

#config t

# ip route 0.0.0.0 0.0.0.0 30.0.0.1

#exit

Show ip route.

for router-0 (static routing)

#config t

# ip route 10.0.0.0 255.0.0.0 20.0.0.0

# ip route 40.0.0.0 255.0.0.0 30.0.0.0

#exit

show ip route.

S 10.0.0.0/8 [1/0] via 20.0.0.0

C 20.0.0.0/8 is directly connected, serial 2/0

C 30.0.0.0/8 is directly connected, serial 3/0

S 40.0.0.0/8 [1/0] via 30.0.0.0

→ Now we ping 10.0.0.1 from the command

prompt of 40.0.0.1

Request timed out.

Reply from 10.0.0.1 bytes=32 time=2ms TTL=128

Reply from 10.0.0.1 bytes=32 time=2ms TTL=128

Reply from 10.0.0.1 bytes=32 time=2ms TTL=128

Ping status for 10.0.0.1

Packets sent = 9, Received = 3 (lost = 1 (25% loss)).

Observation

Through default routing we configure the network here.

Config Aim

the ne

Topolo

PCO

Proced

1. Select

switch

and su

2. Connec

3. Set IF

4. Now (

curren

5. Now (I

network

6. Now ir

zenab)

# config

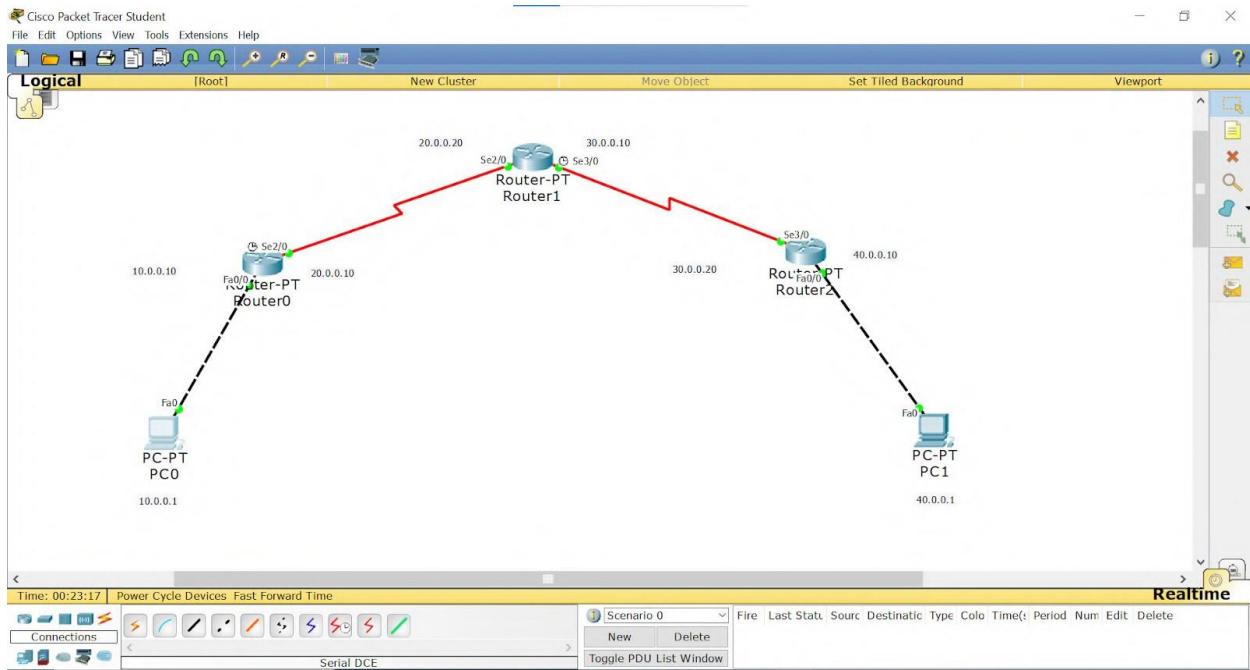
# inter

# ip o

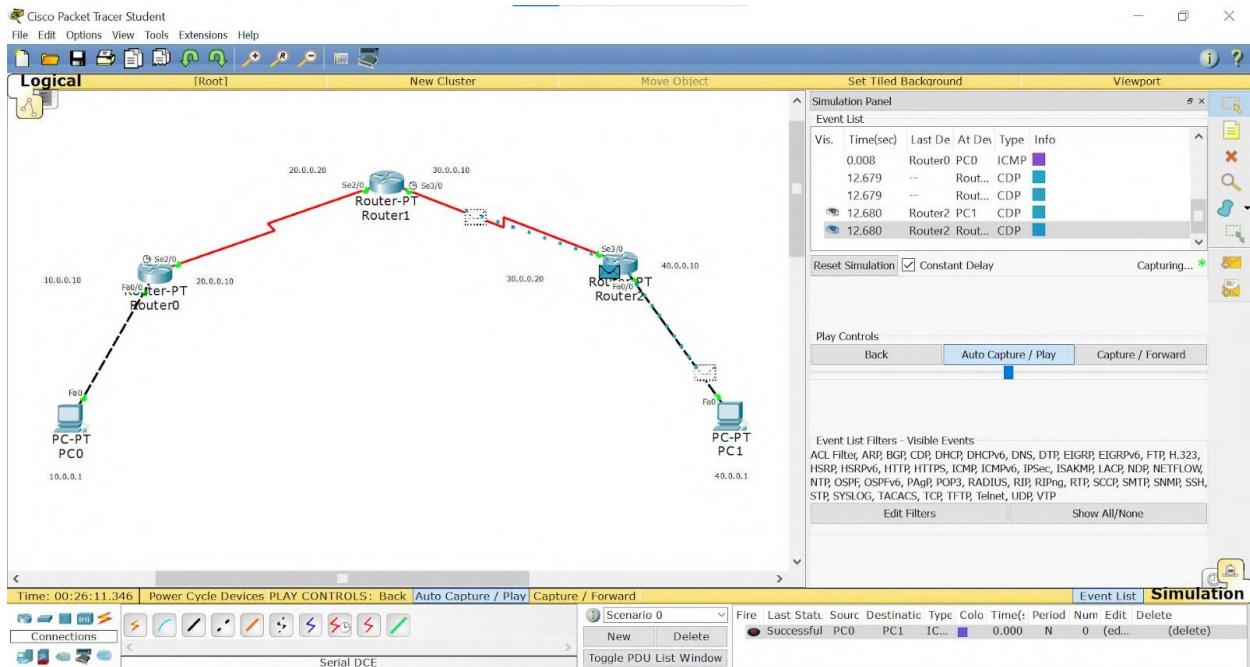
# no

# exit

## TOPOLOGY:



## OUTPUT:



 PC0

Physical Config Desktop Custom Interface

## Command Prompt

```
Packet Tracer PC Command Line 1.0
PC>ping 40.0.0.1

Pinging 40.0.0.1 with 32 bytes of data:

Request timed out.
Reply from 40.0.0.1: bytes=32 time=2ms TTL=125
Reply from 40.0.0.1: bytes=32 time=16ms TTL=125
Reply from 40.0.0.1: bytes=32 time=2ms TTL=125

Ping statistics for 40.0.0.1:
    Packets: Sent = 4, Received = 3, Lost = 1 (25% loss),
Approximate round trip times in milli-seconds:
    Minimum = 2ms, Maximum = 16ms, Average = 6ms

PC>ping 40.0.0.1

Pinging 40.0.0.1 with 32 bytes of data:

Reply from 40.0.0.1: bytes=32 time=21ms TTL=125
Reply from 40.0.0.1: bytes=32 time=9ms TTL=125
Reply from 40.0.0.1: bytes=32 time=2ms TTL=125
Reply from 40.0.0.1: bytes=32 time=4ms TTL=125

Ping statistics for 40.0.0.1:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 2ms, Maximum = 21ms, Average = 9ms

PC>
```

# WEEK 4

Configure DHCP within a LAN and outside LAN.

OBSERVATION:

Date 11  
Page 15

configure DHCP within a LAN and outside the LAN.  
Aim : Connection of server LAN within and outside the network. using switches and routers.

Topology

Router (labeled 'O')  
IP: 20.0.0.20  
Switch (top): IP 10.0.0.20  
Switch (bottom): IP 10.0.0.1  
PC0: IP 10.0.0.1  
PC1: IP 10.0.0.1  
Server: IP 10.0.0.1  
PC2: IP 10.0.0.1  
PC3: IP 10.0.0.1

Procedure

1. Select two or more PC and a server connecting to switch. and another network with only end devices and switch.
2. Connect both switches to router.
3. Set IP address of server as 10.0.0.1.
4. Now go to services < select DHCP < save the current IP address 20.0.0.20.
5. Now check the IP addresses of other devices in the network in the IP configuration in desktop.
6. Now in the CLI of router enable follow steps:  
~~#enable~~  
~~#config terminal~~  
~~#interface fastethernet 0/0~~  
~~#ip address 10.0.0.10 255.0.0.0~~  
~~#no shutdown~~  
~~#exit~~  
~~#interface fastethernet 0/0~~

# ip address 20.0.0.20 255.0.0.0

# no shut

# exit.

7. Go to server < config < gateway to 0.0.0.20

8. Now in router, we need to set ip address of server.

# config t

# fastethernet 0/0.

# ip helper address 10.0.0.1

# no shut

# exit

9. Now go to server < services < DHCP < add new IP address 20.0.0.2.

10. To check the connection, go to the IP configuration of PC outside the network and click on DHCP and IP gateway will be visible.

### Result

From server - from PC2 to PC0 whose ip address is 10.0.0.2

PC > ping 10.0.0.2

pinging 10.0.0.2 with 52 bytes of data:

Request timed out.

Reply from 10.0.0.2 : bytes = 32 time=6ms TTL=125

Reply from 10.0.0.2 : bytes = 32 time=2ms TTL=125

Reply from 10.0.0.2 : bytes = 32 time = 12ms TTL=125

ping statistics for 10.0.0.2

Packets: sent = 4 , Received = 3 , lost = 1

Approximate round trip time in milliseconds.

Minimum = 2ms , Maximum = 12ms , Average = 6ms.

### Observation

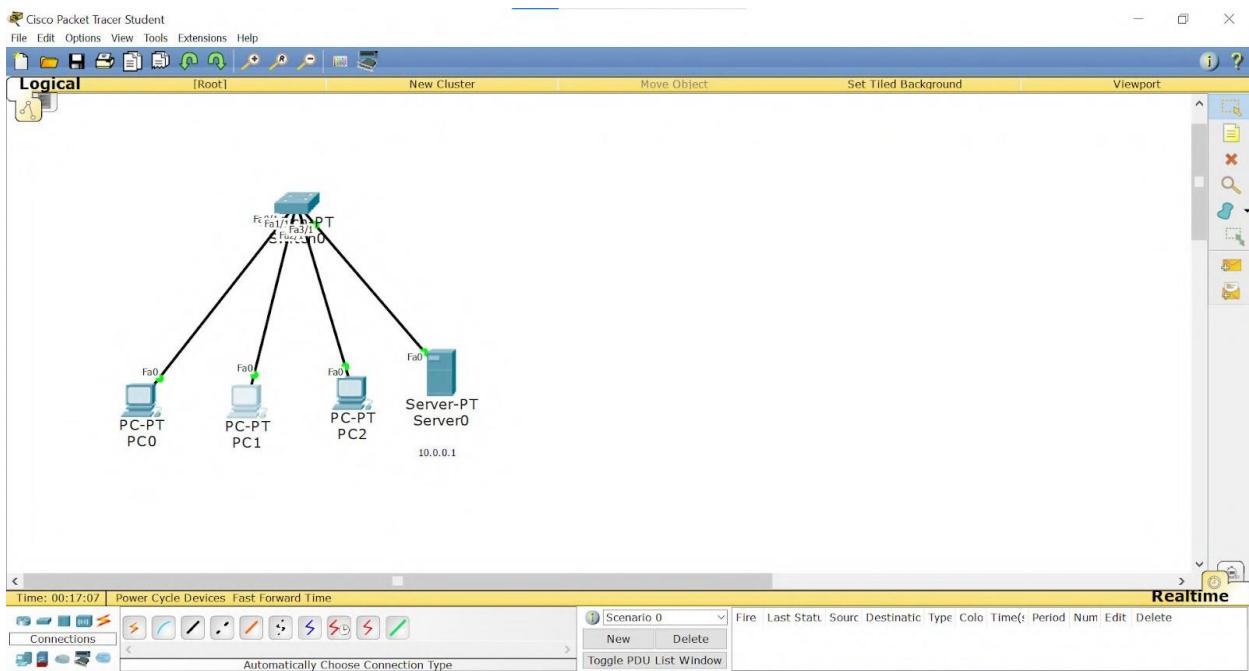
IP address of PC 2 and PC 3 are also automatically set by the server. IP address of PC 2 to 10.0.0.2 and PC 3 to 20.0.0.3.

We could successfully ping PC 2 to PC 0 without loss.

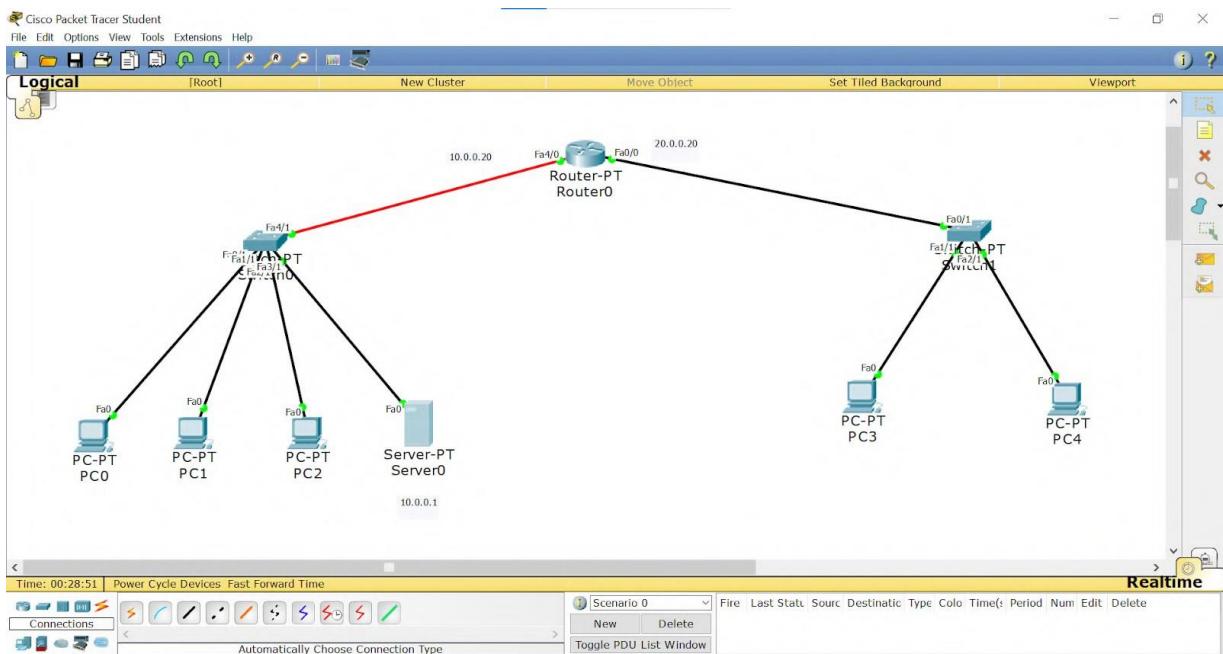
~~S~~

## TOPOLOGY:

### PROGRAM 4.1:

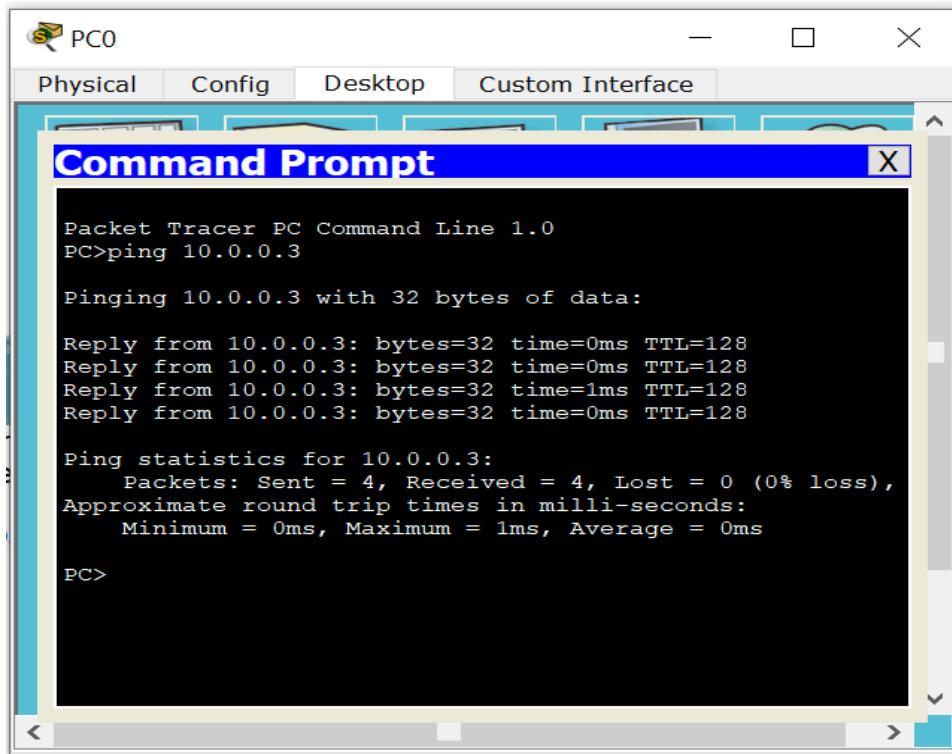


### PROGRAM 4.2:



## OUTPUT:

### PROGRAM 4.1:



PC0

Physical Config Desktop Custom Interface

**Command Prompt**

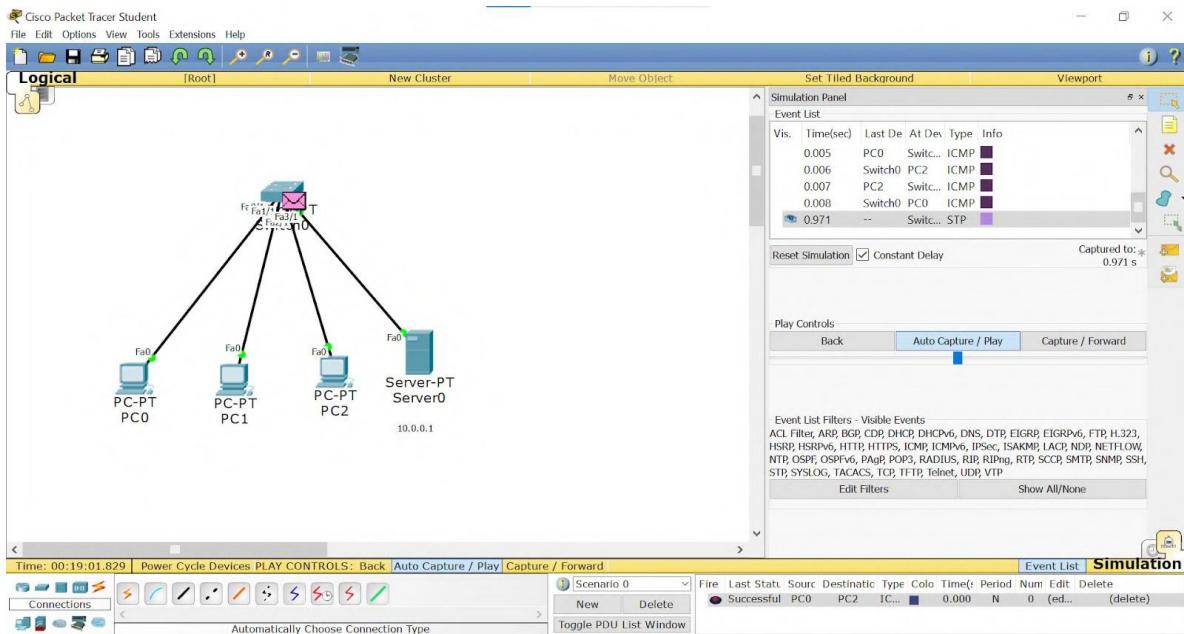
```
Packet Tracer PC Command Line 1.0
PC>ping 10.0.0.3

Pinging 10.0.0.3 with 32 bytes of data:

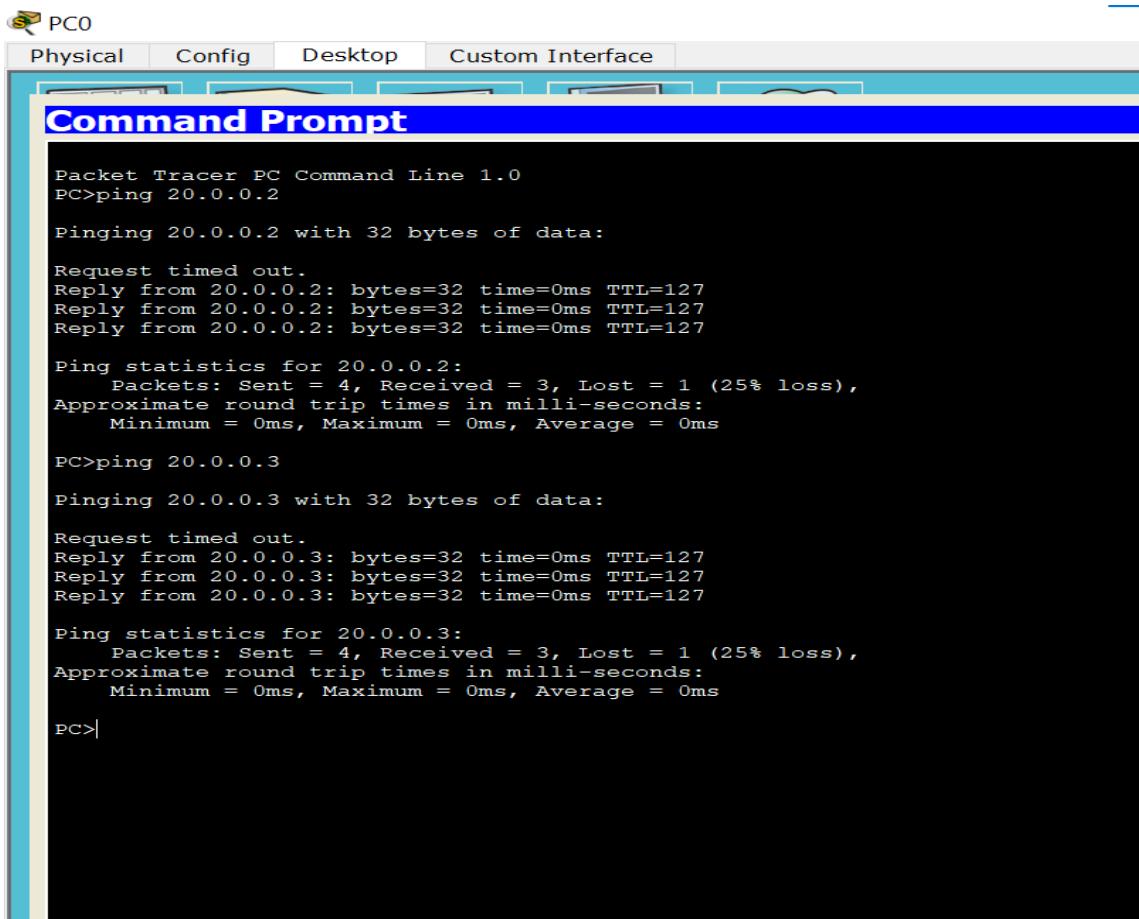
Reply from 10.0.0.3: bytes=32 time=0ms TTL=128
Reply from 10.0.0.3: bytes=32 time=0ms TTL=128
Reply from 10.0.0.3: bytes=32 time=1ms TTL=128
Reply from 10.0.0.3: bytes=32 time=0ms TTL=128

Ping statistics for 10.0.0.3:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 0ms, Maximum = 1ms, Average = 0ms

PC>
```



## PROGRAM 4.2:



The screenshot shows a software interface titled "PC0" at the top left. Below it is a menu bar with four tabs: "Physical", "Config", "Desktop", and "Custom Interface". The "Custom Interface" tab is currently selected. A blue header bar across the top of the main window reads "Command Prompt". The main area contains a black terminal window displaying the output of ping commands. The terminal output is as follows:

```
Packet Tracer PC Command Line 1.0
PC>ping 20.0.0.2

Pinging 20.0.0.2 with 32 bytes of data:

Request timed out.
Reply from 20.0.0.2: bytes=32 time=0ms TTL=127
Reply from 20.0.0.2: bytes=32 time=0ms TTL=127
Reply from 20.0.0.2: bytes=32 time=0ms TTL=127

Ping statistics for 20.0.0.2:
    Packets: Sent = 4, Received = 3, Lost = 1 (25% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 0ms, Average = 0ms

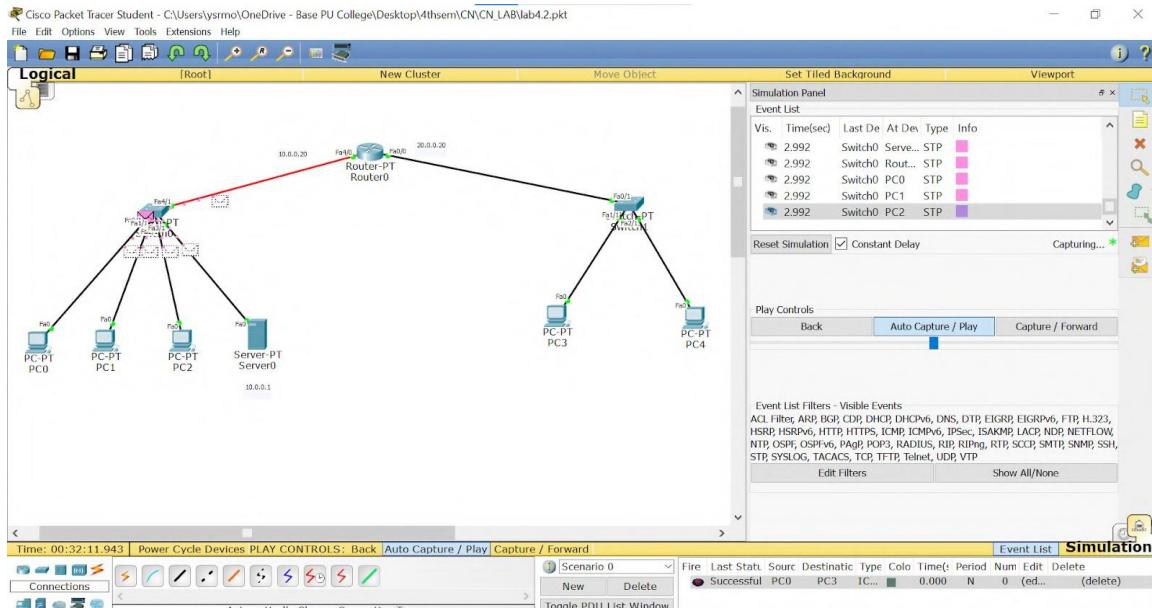
PC>ping 20.0.0.3

Pinging 20.0.0.3 with 32 bytes of data:

Request timed out.
Reply from 20.0.0.3: bytes=32 time=0ms TTL=127
Reply from 20.0.0.3: bytes=32 time=0ms TTL=127
Reply from 20.0.0.3: bytes=32 time=0ms TTL=127

Ping statistics for 20.0.0.3:
    Packets: Sent = 4, Received = 3, Lost = 1 (25% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 0ms, Average = 0ms

PC>
```



# WEEK 5

Configure Web Server, DNS within a LAN.

OBSERVATION:

Date 20/07/23  
Page 12

Configure web server, DNS within a LAN.

Aim: Configure web server, DNS within a LAN.

Topology: A topology diagram showing a PC (PC0) connected to a switch via port Fa0. The switch has two ports: Fa0/0 and Fa1/0. Both ports are connected to a server (server0) via ports Fa0 and Fa1 respectively. The server is labeled "server0" and "serverA".

Procedure:

1. Connect a switch, PC and a server to form a LAN.
2. Set PC's IP address by checking on it and go to config, there in fastethernet option set IP address as 10.0.0.1 and subnet mask.
3. Set server's IP address as 10.0.0.2 and subnet mask respectively.
4. Go to PC's desktop and click on web browser, in the URL tab type 10.0.0.2 you will get a default display.
5. To make a CV here, we need to make changes in server services.
6. Go to server → Services → HTTP → index.html. There create the CV and click on save.
7. Again go to PC → Desktop → webbrowser and type 10.0.0.2 you will see the CV or content that is changed.
8. Next, go to server → Services → DNS and switch on IP address as 10.0.0.2. Press add and save it.
9. Again go back to PC → desktop → webbrowser &

type the given domain name. Here we can see the CV which had been created earlier.

### Output

web browser

[<] [>] URL | http://dikshya.cv | [Go] [stop]

CV

Name: Dikshya Aryal

USN: IBM21CS058

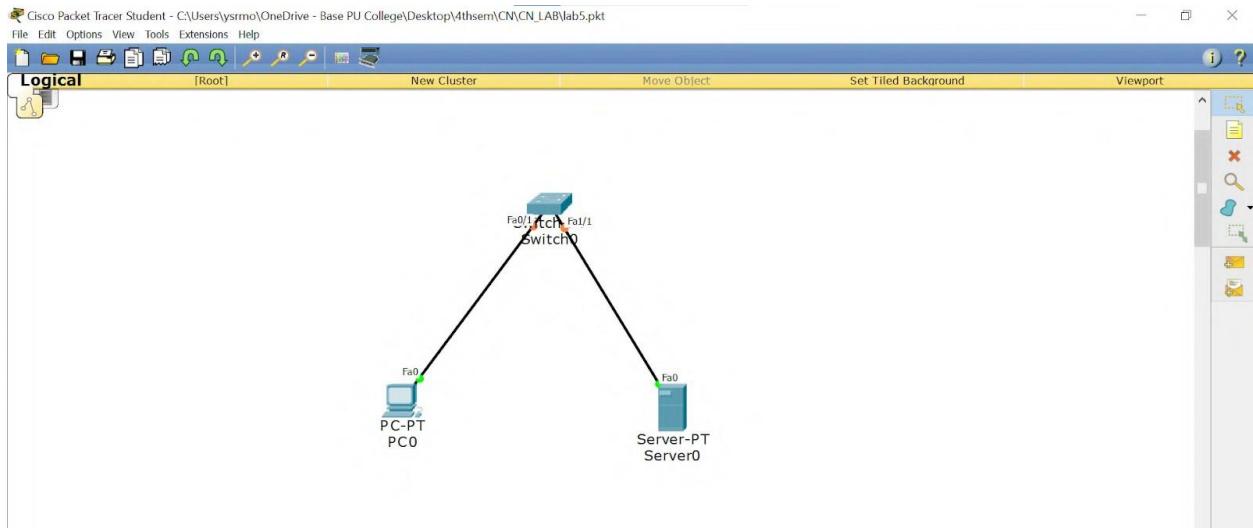
Languages: C/Java/PHP

Image:

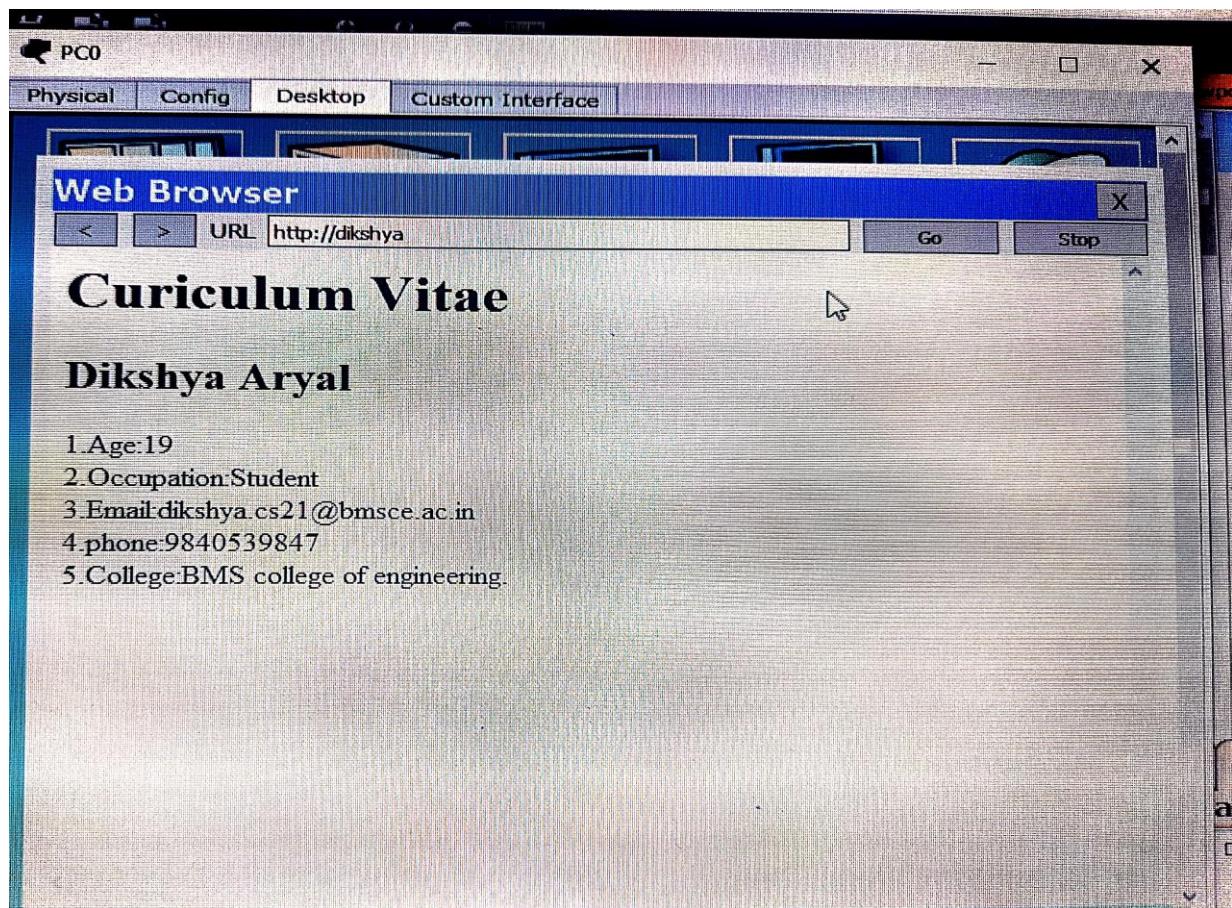
### Observation.

1. If you wanted to go to a certain website, you would open web browser and type domain name of that website or else, you can also type the IP address instead of you know that website IP address.
2. Since we can't remember IP address of all websites DNS server will search through its cache to find a matching IP address for that domain name & when it finds a matching IP address it will resolve that domain name to IP address of website, once that is done then computer is able to communicate with a webserver & retrieve the webpage.

## TOPOLOGY:



## OUTPUT:



# WEEK 6

Configure RIP routing Protocol in Routers.

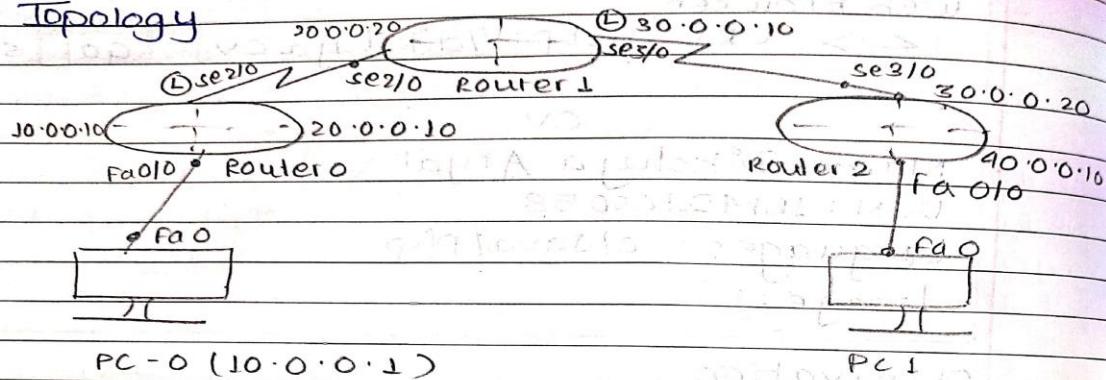
OBSERVATION:

Date 1/1  
Page 20

Configure RIP routing protocols in routers.

Aim: configure RIP routing protocols in routers

Topology



Procedure

1. Create a Network using 3 routers and 2 PCs. Connect routers using serial DCE cable and PC to router using copper-crossover cable.
2. Set the IP address and gateway no for both PC's as 10.0.0.1 - IP, 10.0.0.10 - gateway → PC 0  
40.0.0.1 - IP, 40.0.0.10 - gateway → PC 1 respectively.
3. Go to router → CLI mode, and execute following commands.
  1. No
  2. Enable
  3. config t
  4. Interface fast-ethernet 0/0.
  5. IP address 10.0.0.10 255.0.0.0
  6. No shut
  7. Exit
  8. Interface se 2/0
  9. IP address 20.0.0.10 255.0.0.0

10. Encapsulation PPP
11. Clock rate 69000
12. No Shut.
4. Here for router with fast ethernet execute only step 11 and type no shut.
5. Only for router to router connection execute all steps, also execute the step 11 only for the router connection which has a clock symbol at start. Repeat the steps for all routers.
6. Again go to router 0 → CLI mode. and type these steps.
  1. config t
  2. router rip
  3. Network 10.0.0.0
  4. Network 20.0.0.0
  5. Exit
7. Repeat these steps for all the routers.
8. At last now go to each router and type show IP route. Here the IP addresses associated with the router will be labelled as C and other IP addresses will be labelled as R.
9. Lastly go to PC0 and ping a message to PC1 using ping destination IP address command.

### Ping output

Packet tracer PC command line 1>

PC > Ping 40.0.0.1

Pinging 40.0.0.1 with 32 bytes of data  
Request timed out

Reply from 40.0.0.1 bytes=32 time=8ms TTL=125

Reply from 40.0.0.1 bytes=32 time=5ms TTL=125

Reply from 40.0.0.1 bytes=32 time=10ms TTL=125

Ping statistics for 90.0.0.1

Packet: sent 4 received = 3 lost = 1 (25% loss)

Avgon. round trip time 8 in milliseconds

Minimum = 5ms, Maximum = 10ms, Average = 7ms

### Observation

Routing Info protocol is a dynamic routing protocol that uses hop count as a routing metric to find the best path between source and destination. It is a distance-vector routing protocol.

Hop count is the no. of routers coming between the source and destination. The path with least hop count is selected.

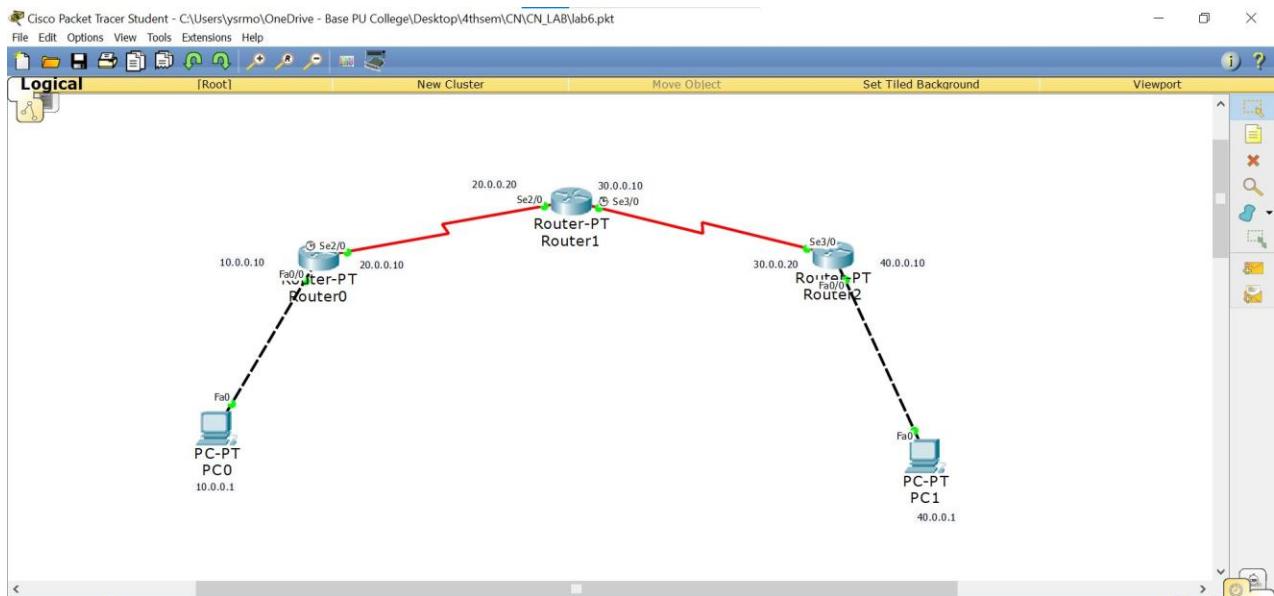
Updates of network are exchanged periodically.

Updates of routing info are always broadcast.

Full routing tables are sent in updates.

Routers always trust routing Pinfo. received from neighbour routers.

## TOPOLOGY:



## OUTPUT:

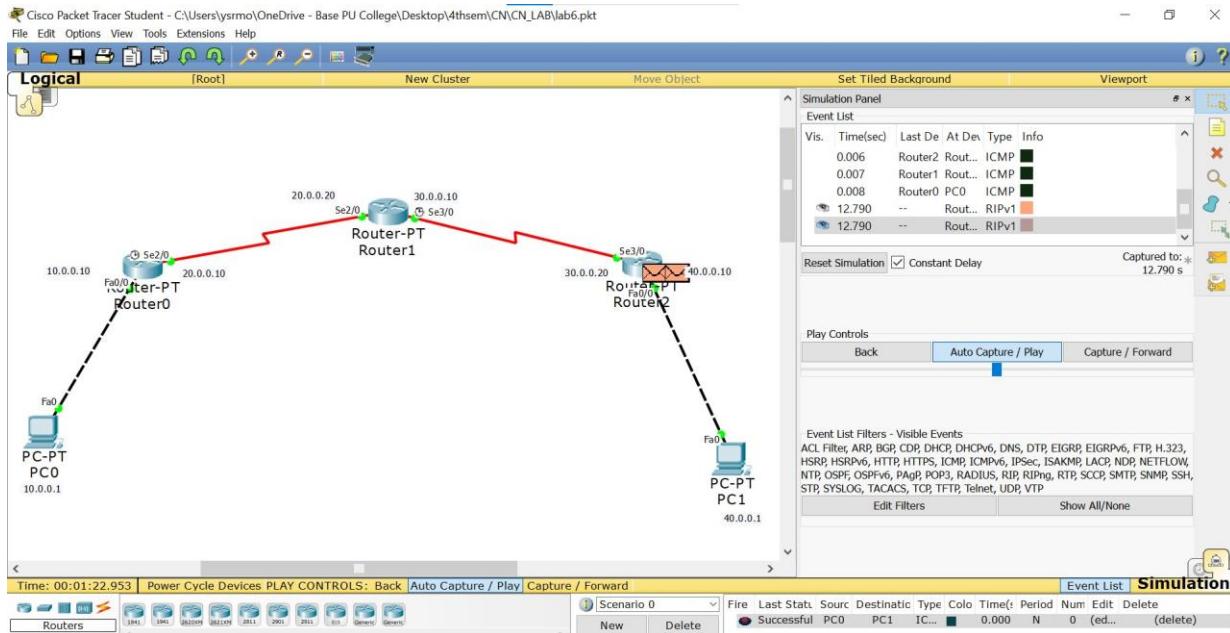
```
PC0
Physical Config Desktop Custom Interface
Command Prompt X
Packet Tracer PC Command Line 1.0
PC>ping 40.0.0.1

Pinging 40.0.0.1 with 32 bytes of data:

Request timed out.
Reply from 40.0.0.1: bytes=32 time=8ms TTL=125
Reply from 40.0.0.1: bytes=32 time=5ms TTL=125
Reply from 40.0.0.1: bytes=32 time=10ms TTL=125

Ping statistics for 40.0.0.1:
    Packets: Sent = 4, Received = 3, Lost = 1 (25% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 5ms, Maximum = 10ms, Average = 7ms

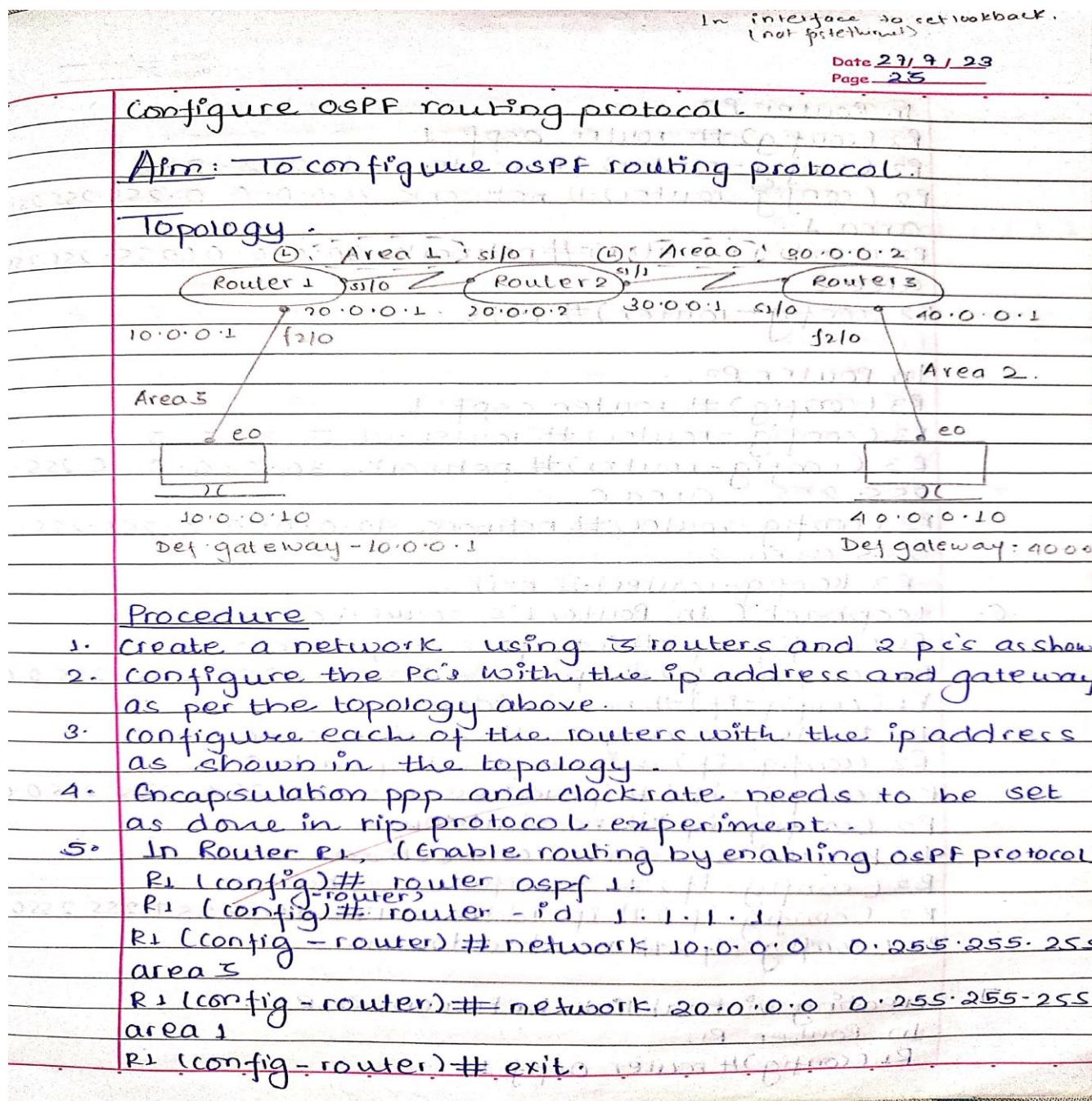
PC>
```



# WEEK 7

Configure OSPF routing protocol.

OBSERVATION:



In Router R2

```
R2 (config) # router ospf 1
R2 (config-router) # router-id 2.2.2.2
R2 (config-router) # network 20.0.0.0 0.255.255.255 area 1
R2 (config-router) # network 30.0.0.0 0.0.255.255 area 0
R2 (config-router) # exit
```

In Router R3

```
R3 (config) # router ospf 1
R3 (config-router) # router-id 3.3.3.3
R3 (config-router) # network 30.0.0.0 0.255.255.255 area 0
R3 (config-router) # network 40.0.0.0 0.255.255.255 area 2
R3 (config-router) # exit
```

6. Loopback [ In Router 1's serial connection].

```
R1 (config-if) # interface loopback 0
R1 (config-if) # ip address 172.16.1.252 255.255.0.0
R1 (config-if) # no shutdown
```

In Router 2's any of the serial connection

```
R2 (config-if) # interface loopback 0
R2 (config-if) # ip address 172.16.1.253 255.255.0.0
R2 (config-if) # no shutdown
```

In Router 3's serial connection

```
R3 (config-if) # interface loopback 0
R3 (config-if) # ip address 172.16.1.254 255.255.0.0
R3 (config-if) # no shutdown
```

7: Creating virtual link between R1, R2

In Router R1.

```
R1 (config) # router ospf 1
```

R1 (config - router) # area 1 virtual-link 2.2.2.2  
some info appears. R2 (config #)

In Router 2.

R2 (config) # router ospf 1.

R2 (config - router) # area 1 virtual link 1.1.1.1

R2 (config - router) # exit .

R2 (config #) some info.

8. for every router, we do show ip - route command.

Output :

Ping 40.0.0.10 with 32 bytes of data.

Pinging 40.0.0.10 with 32 bytes of data.

Reply from 40.0.0.10 bytes = 32 time = 11ms TTL = 125

Reply from 40.0.0.10 bytes = 32 time = 8ms TTL = 125

Reply from 40.0.0.10 bytes = 32 time = 2ms TTL = 125

Reply from 40.0.0.10 bytes = 32 time = 8ms TTL = 125

Ping statistics for 40.0.0.10

Packet(s) sent = 4 Received = 4 Lost = 0 (0% loss)

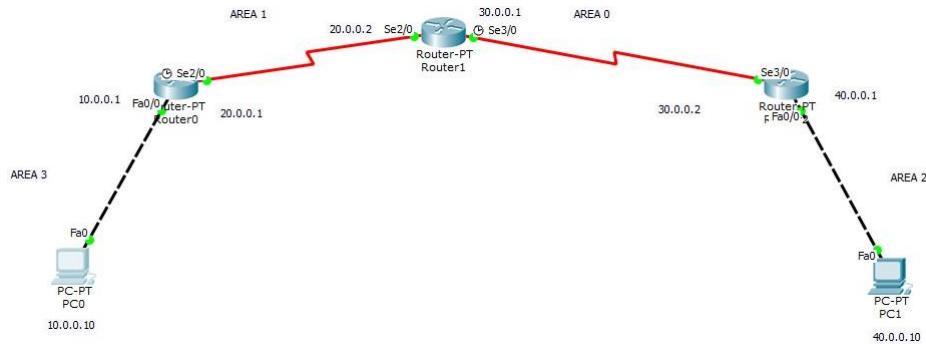
Approximate round trip time in milliseconds.

Minimum = 2ms Maximum = 11ms Average = 7ms

### Observation - link

- OS is a state routing protocol that is used to find the best path between source and destination using its own algorithm.
- This network is divided into 1 areas where area 0 is the backbone.
- After we make the virtual-link between the area which isn't connected to the backbone area, we can ping messages successfully.

## TOPOLOGY:



## OUTPUT:

```
Packet Tracer PC Command Line 1.0
PC>ping 40.0.0.10

Pinging 40.0.0.10 with 32 bytes of data:

Reply from 10.0.0.1: Destination host unreachable.

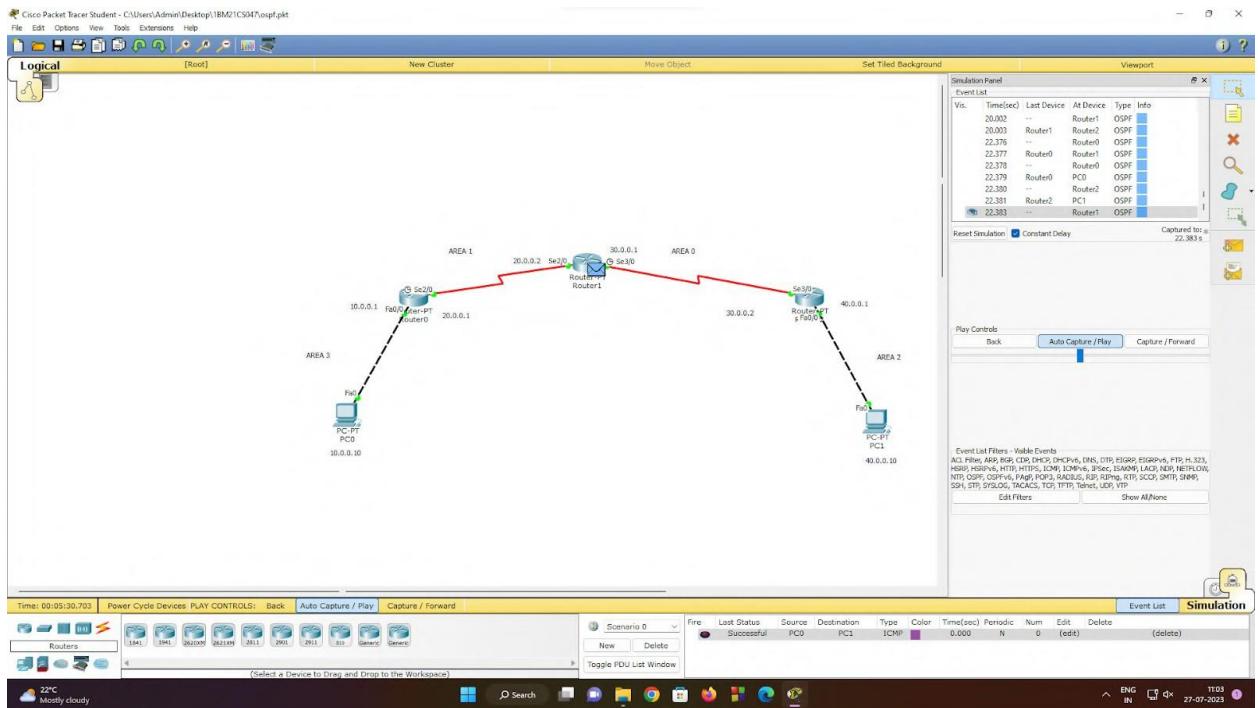
Ping statistics for 40.0.0.10:
    Packets: Sent = 4, Received = 0, Lost = 4 (100% loss),
PC>ping 40.0.0.10

Pinging 40.0.0.10 with 32 bytes of data:

Request timed out.
Reply from 40.0.0.10: bytes=32 time=4ms TTL=125
Reply from 40.0.0.10: bytes=32 time=6ms TTL=125
Reply from 40.0.0.10: bytes=32 time=12ms TTL=125

Ping statistics for 40.0.0.10:
    Packets: Sent = 4, Received = 3, Lost = 1 (25% loss),
Approximate round trip times in milli-seconds:
    Minimum = 4ms, Maximum = 12ms, Average = 7ms

PC>
```



## WEEK 8

To construct a simple LAN and understand the concept and operation of Address Resolution Protocol (ARP).

OBSERVATION:

Date 3/8/23  
Page 27

To construct simple LAN and understand the concept and operation of Address Resolution Protocol (ARP)

Aim: to construct simple LAN and understand the concept & operation of ARP.

Switch - pl

Fa0/1      Fa0/2      Fa0/3      Fa0

PC1 (PC-1)      10.0.0.1

PC2 (PC-2)      10.0.0.2

PC3 (PC-3)      10.0.0.3

Server - pl      Server 0      10.0.0.5

Procedure

1. Create a topology of 4 PCs and a Server.
2. IP address assigned to all.
3. Connect them through a switch.
4. Use a inspect tool to click on a PC to see the ARP Table.
5. Command in CLI for the same ls arp -a (PC1 - CLI)
6. Initially ARP table
7. Also in CLI for the same of switch, the command show mac address-table can be given on every transaction to see how the switch learns from transactions and build the add-table.
8. Use the capture button in the simulation panel to go step by step so that the changes in ARP can be clearly noted.
9. Observe the switch as well as the nodes update the ARP tables as and when a new communication starts.

### Ping output

PC > ping 10.0.0.4

Pinging 10.0.0.4 with 32 bytes of data.

Reply from 10.0.0.4: bytes = 32 time = 0ms TTL = 128

Reply from 10.0.0.4: bytes = 32 time = 0ms TTL = 128

Reply from 10.0.0.4: bytes = 32 time = 0ms TTL = 128

Reply from 10.0.0.4: bytes = 32 time = 0ms TTL = 128

Reply from 10.0.0.4: bytes = 32 time = 0ms TTL = 128

Ping statistics for 10.0.0.4.

Packets: sent = 4, received = 4, lost = 0 (0% loss)

Approx round trip times in milliseconds:

minimum = 0ms, max = 0ms, Avg = 0ms.

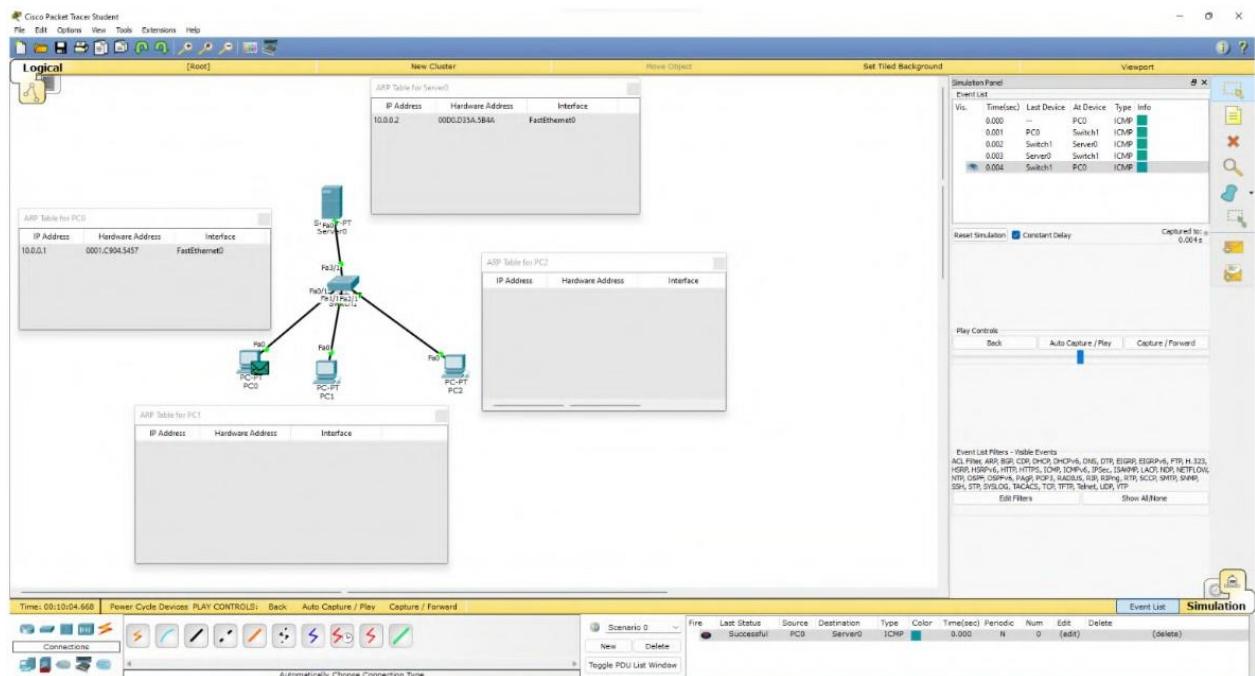
PC > arp - a

Internet address	physical add.	Type
10.0.0.4	00:02:2f:a0:32:9d	dynamic.

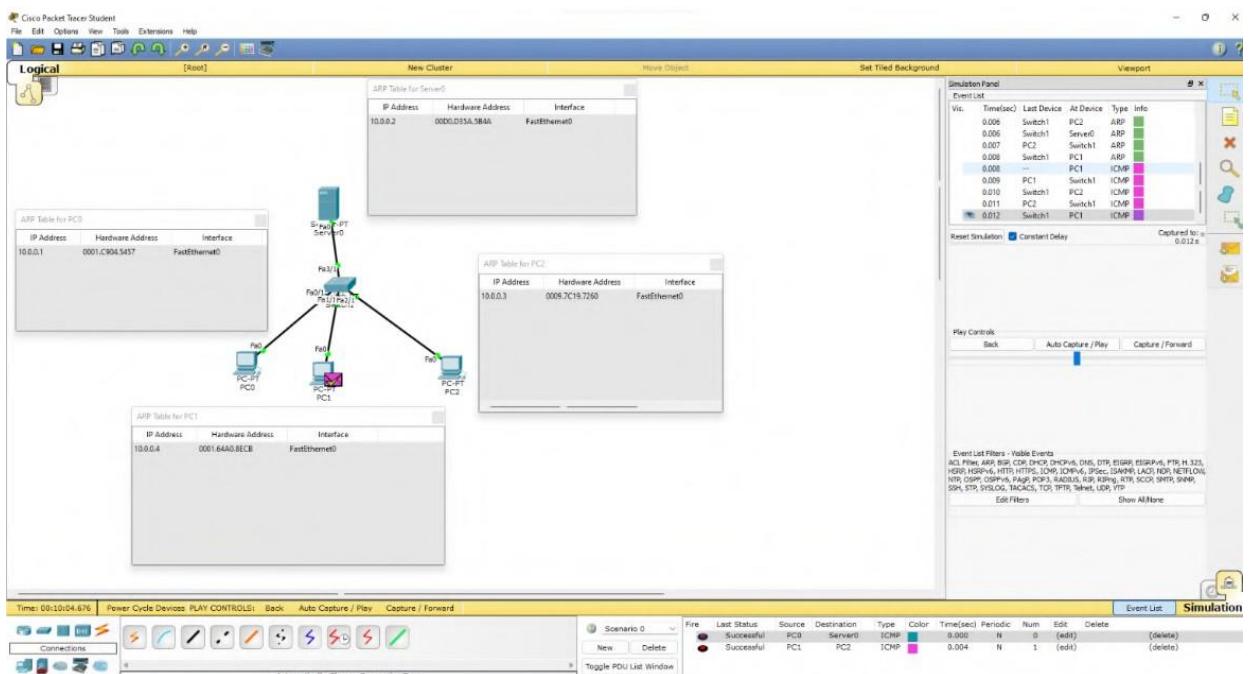
### Observation

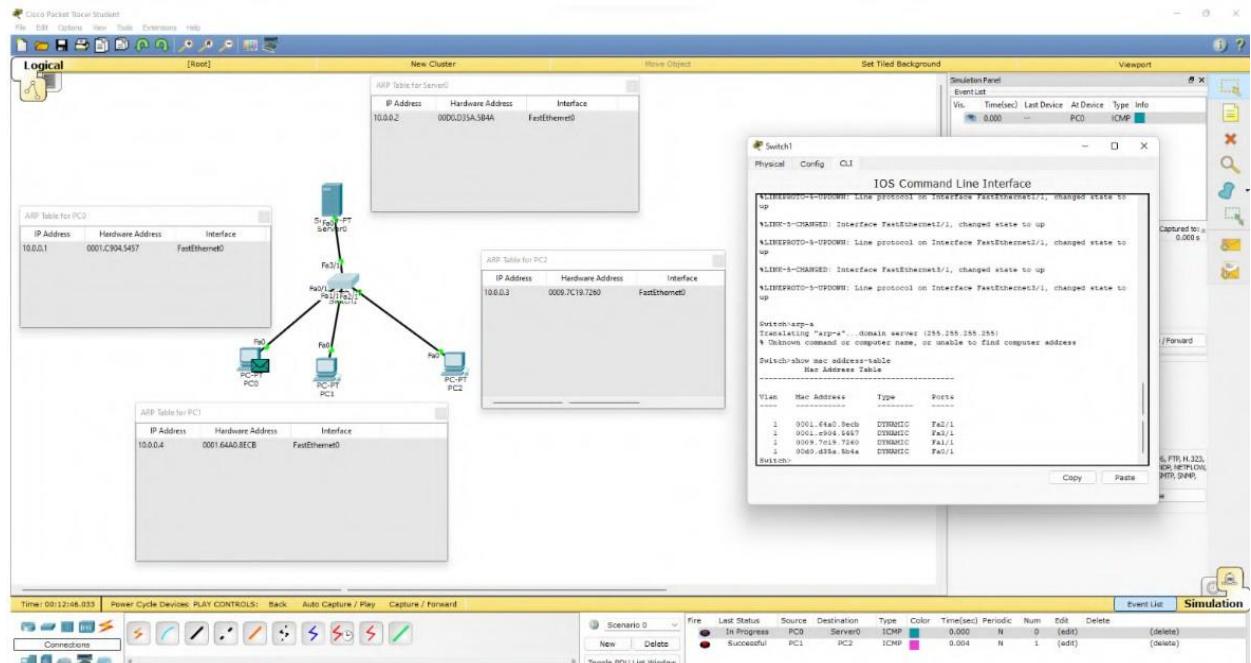
- When we ping 1 PC or server the address of servers is known to pc & vice versa.
- When we ping between other two PCs simultaneously the address of each other are unknown.
- Everytime a host requests a MAC address in order to send a packet to another host in LAN, it checks its ARP cache to see if the ip to Mac address translation address already exists. If the translation doesn't exist it performs ARP.

## TOPOLOGY:



## OUTPUT:

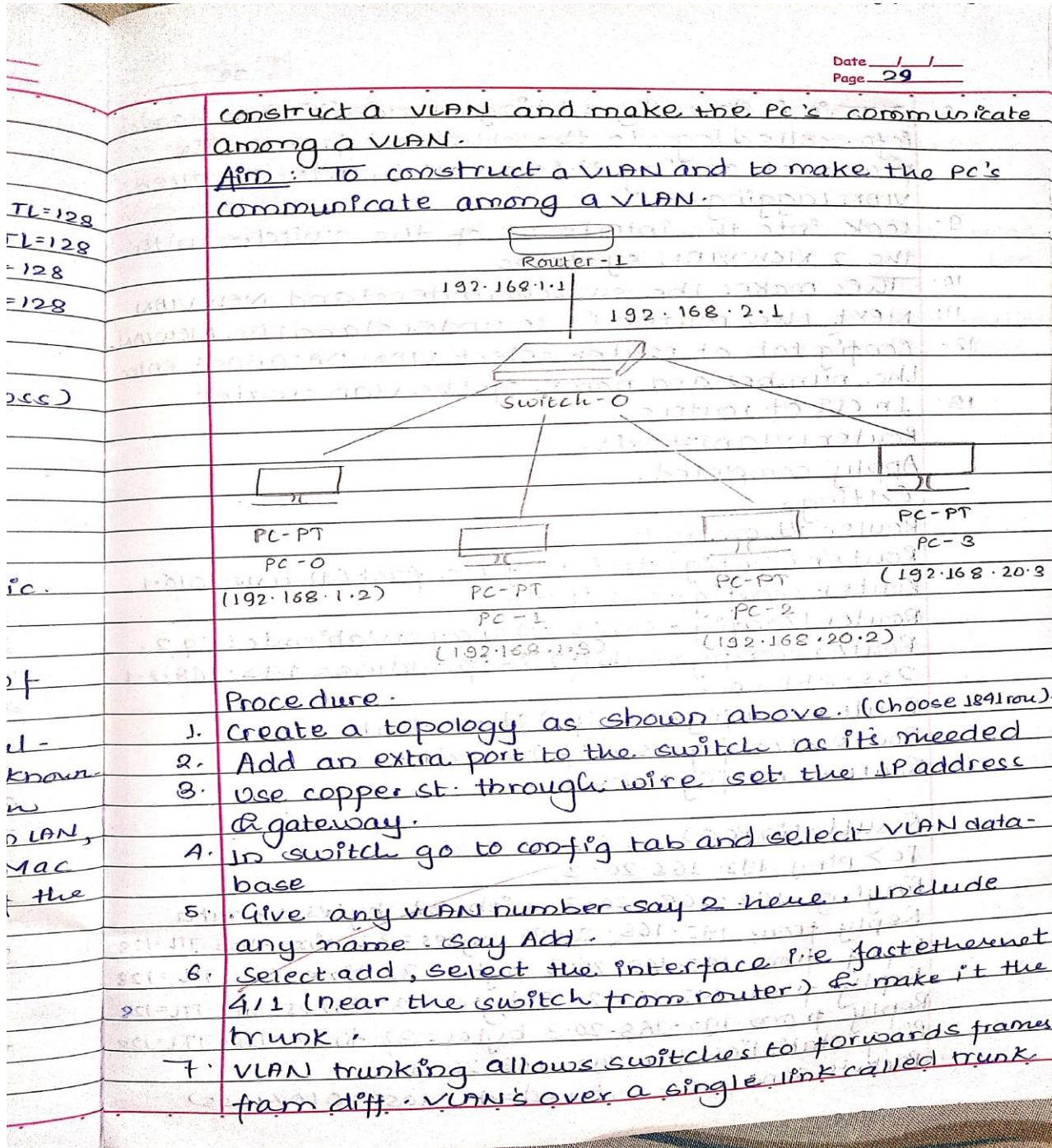




# WEEK 9

To construct a VLAN and make a pc communicate among VLAN.

## OBSERVATION:



8. This is done by adding an additional header info. called tag to the ethernet frame. The process of adding this small header is called VLAN tagging.
9. Look into the interfaces of the switcher with the 2 NEWVLAN systems
10. This makes the switch understand NEWVLAN.
11. Next the router has to understand the NEWVLAN.
12. Config tab of router select VLAN DATABASE enter the number and name of the VLAN created
13. In CLI of routers,  
Router (vlan) # exit.  
Apply completed.  
Exiting.

Router # config  
Router (config) # interface fast Ethernet 0/0/1  
Router (config-subif) #  
Router (config-subif) # encapsulation dot1q 2.  
Router (config-subif) # ip address 192.168.2.1  
255.255.0.

Router (config-subif) # no shutdown  
Router (config-subif) # exit  
Router (config) # exit

Result (in PC0)

PC > Ping 192.168.20.3

Pinging 192.168.20.3 with 32 bytes of data  
Reply from 192.168.20.3 bytes = 32 time = 0ms TTL = 128  
Reply from 192.168.20.3 bytes = 32 time = 0ms TTL = 128  
Reply from 192.168.20.3 bytes = 32 time = 0ms TTL = 128  
Reply from 192.168.20.3 bytes = 32 time = 0ms TTL = 128  
Ping statistics for 192.168.20.3

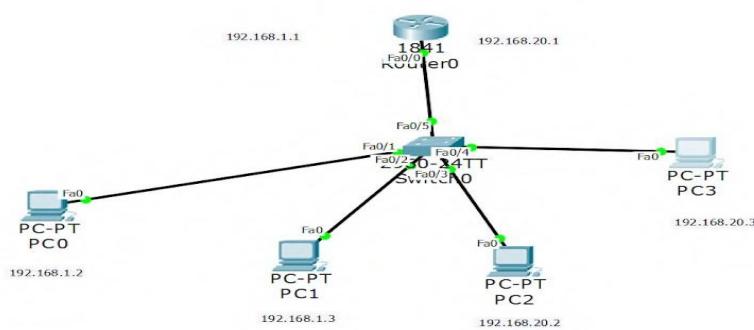
Packet sent = 4, received = 4, lost = 0 (0% loss)

Approximate round trip times in milliseconds  
minimum = 0ms, maximum = 1ms, Average = 0ms.

### Observations

- (i) VLAN - virtual local area network is any broadcast domain that is partitioned and isolated in a computer network at the data link layer.
- (ii) It is virtualized connection that connects multiple devices and network nodes from diff. LANs into one logical network.

## TOPOLOGY:



## OUTPUT:

```

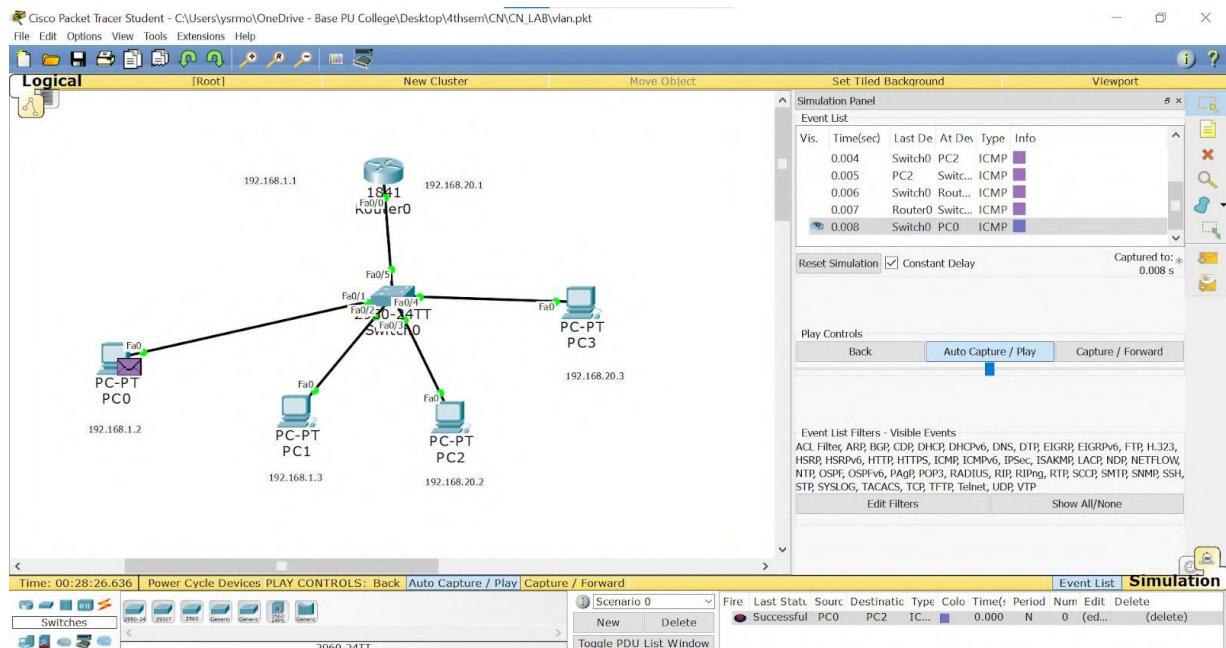
Packet Tracer PC Command Line 1.0
PC>ping 192.168.20.3

Pinging 192.168.20.3 with 32 bytes of data:

Request timed out.
Reply from 192.168.20.3: bytes=32 time=0ms TTL=127
Reply from 192.168.20.3: bytes=32 time=5ms TTL=127
Reply from 192.168.20.3: bytes=32 time=0ms TTL=127

Ping statistics for 192.168.20.3:
    Packets: Sent = 4, Received = 3, Lost = 1 (25% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 5ms, Average = 1ms

PC>
  
```



# WEEK 10

Demonstrate the TTL/ Life of a Packet.

OBSERVATION:

When the TTL value sends an ICMP message.

Date \_\_\_\_\_  
Page \_\_\_\_\_

Demonstrate the TTL/life of a packet.

Aim: To demonstrate the TTL/life of a packet.

20.0.0.20 Router 1  
20.0.0.10 30.0.0.20 30.0.0.10 Router 2  
10.0.0.10 Router 0  
10.0.0.1

Procedure:

1. Create a topology as shown above with 2 PCs & 3 routers.
2. Set the IP address & gateway for all PCs.
3. Configure the routers either static/default routing way.
4. In simulation mode send a simple PDU from 1 to another PC.
5. Use capture button to capture every transfer.
6. Click on the PDU during every transfer to see the inbound & outbound PDU details.

Output

XP	4	8	16	17	TTL: 28
4	TTL	DSCH	0X	0X	
0					

ID: 0X6

TTL: 255 PRO: 0X1 checksum

SRC IP: 10.0.0.1

DST IP: 40.0.0.1

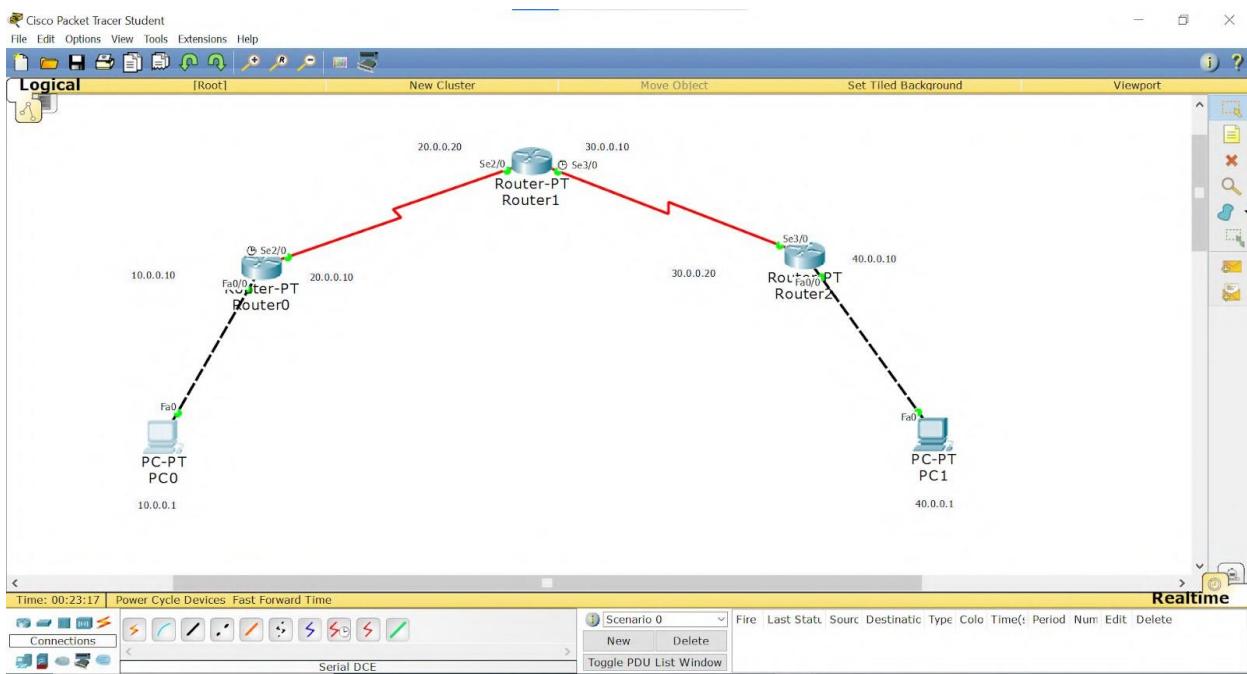
OPT: 0X0

DATA variable length 0X0

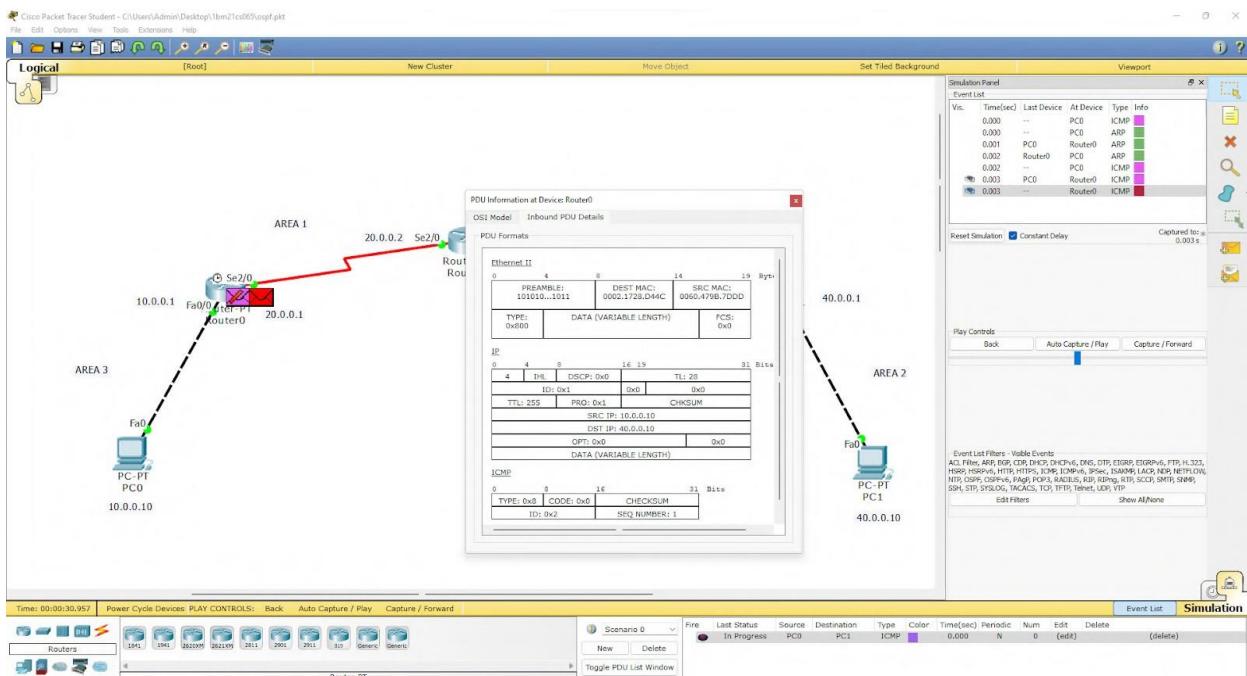
Observation

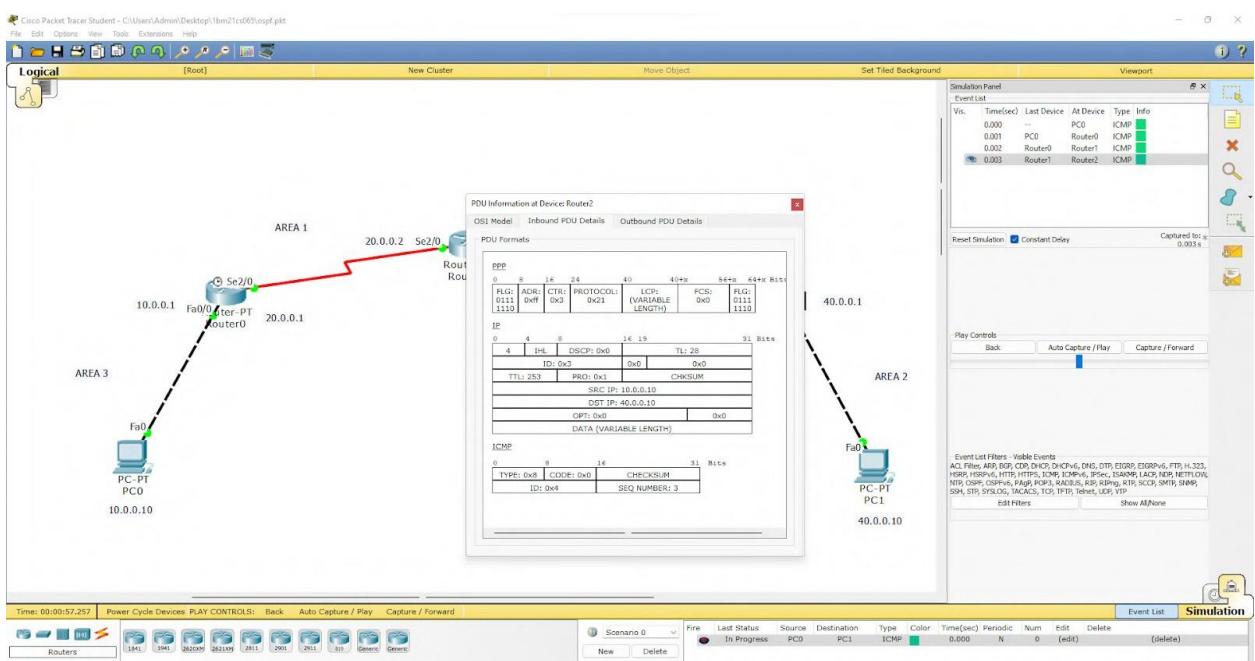
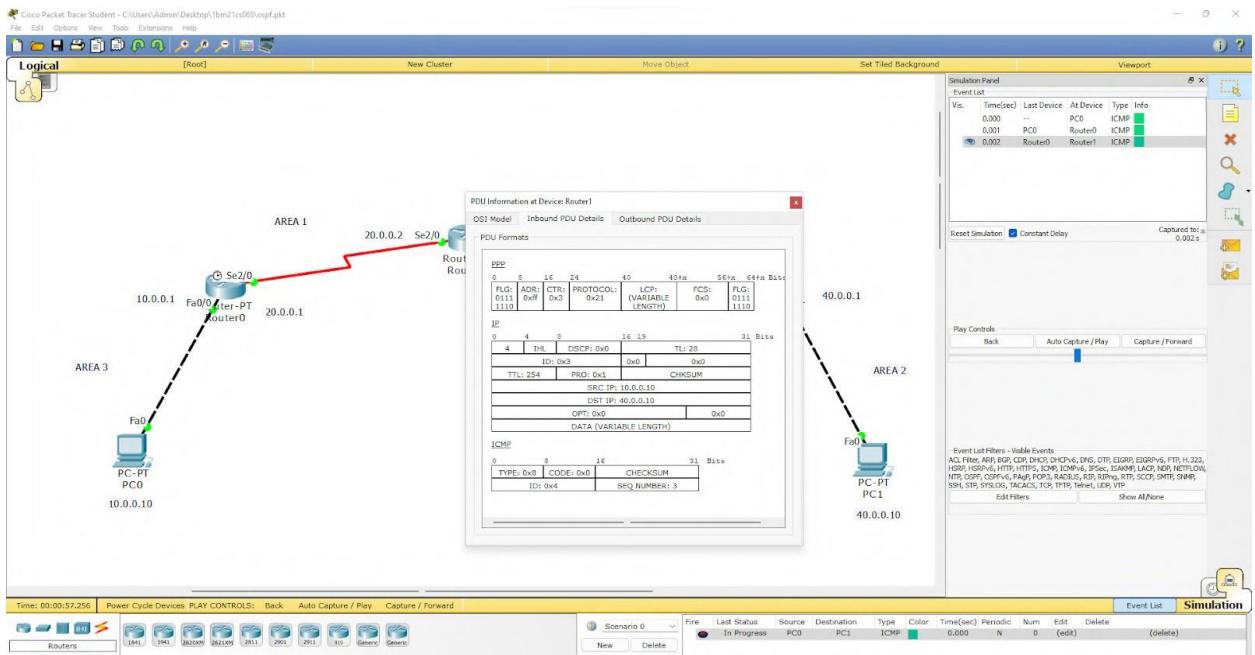
1. The no. of hops the packet travel before being discarded as TTL reaches zero.
2. Datagram TTL field is set by the sender & reduced by each router along the path to its destination.
3. The router reduces TTL value by some value forming the packets.

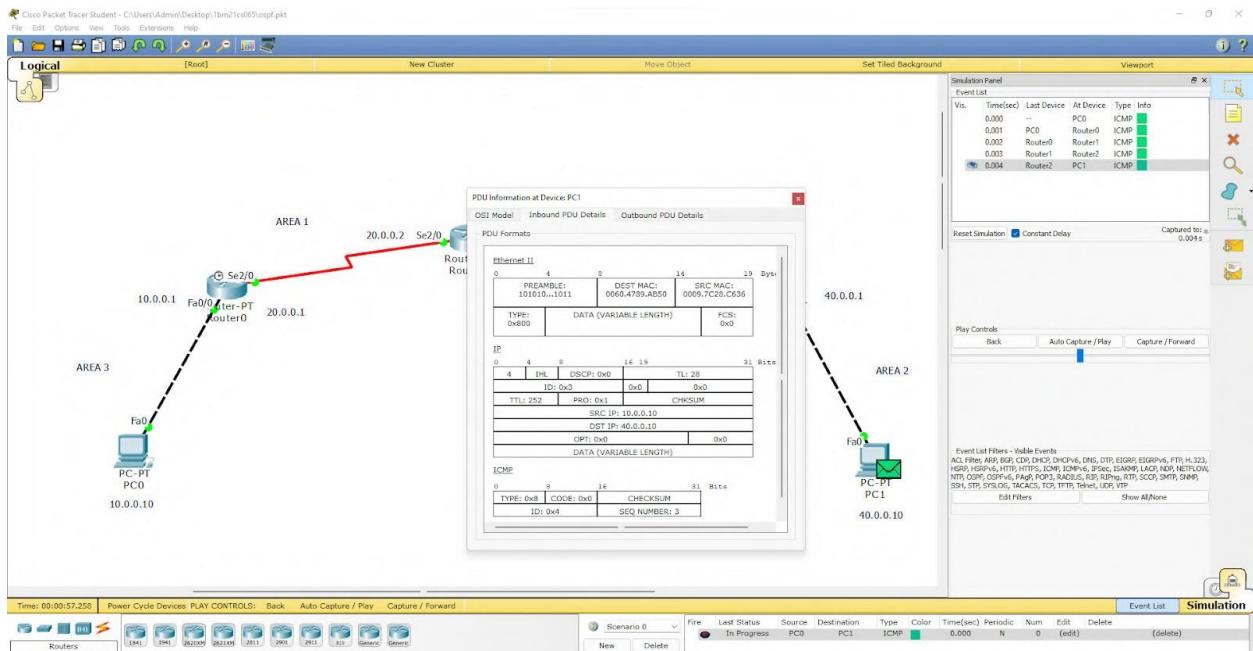
## TOPOLOGY:



## OUTPUT:







## WEEK 11

To construct a WLAN and make the nodes communicate wirelessly

OBSERVATION:

Date / /  
Page 5A

To construct a WLAN and make the nodes communicate wirelessly.

Aim : To construct a WLAN and make the nodes communicate wirelessly.

Router  
10.0.0.10

switch 0

Access point

laptop  
10.0.0.4

PC 0  
10.0.0.2

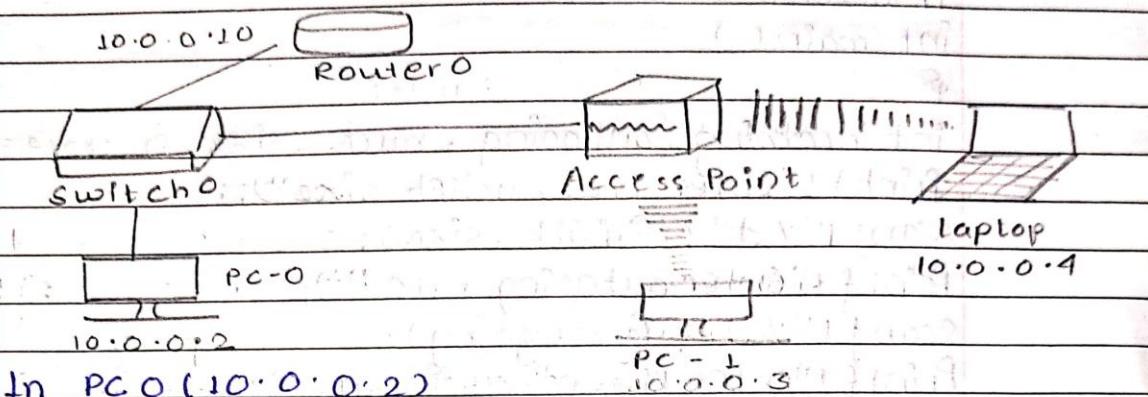
PC 1  
10.0.0.3

Procedure.

1. Construct above topology. Use access point - ppt connect that to router. Set the IP address of the PC connected with wireless.
2. Configure PC 0 and Router 1 as is normally done
3. Configure Access Point 1 - port 1 → SSID Name - any name (WLAN here)
4. Select WEP and give any 10 digit hex key - 1234567890 here
5. Configure PC 1 & laptop with wireless standards
6. Switch off the device. Drag the existing PT-HOST - NM - IAM to the component listed in the LHS. Drag WMP30GN wireless interface to the empty port. switch on the device.
7. In the config tab a new wireless interface could have been added. Now configure SSID, WEP, WEP Key, IP address and gateway (as normally done) to the device.
8. Ping from each device to every other device and see the results.

## Result

### Topology



In PC0 (10.0.0.2)

Pc > ping 10.0.0.3

Pinging 10.0.0.3 with 32 bytes of data

Reply from 10.0.0.3 bytes = 32 time = 11 ms TTL=120

Reply from 10.0.0.3 bytes = 32 time = 12ms TTL=120

Reply from 10.0.0.3 bytes = 32 time = 6ms TTL=120

Reply from 10.0.0.3 bytes = 32 time = 0ms TTL=120

Ping statistics for 10.0.0.3

Packet s: Sent = 4, Received = 4, Lost = 0

Approx round trip time in milliseconds

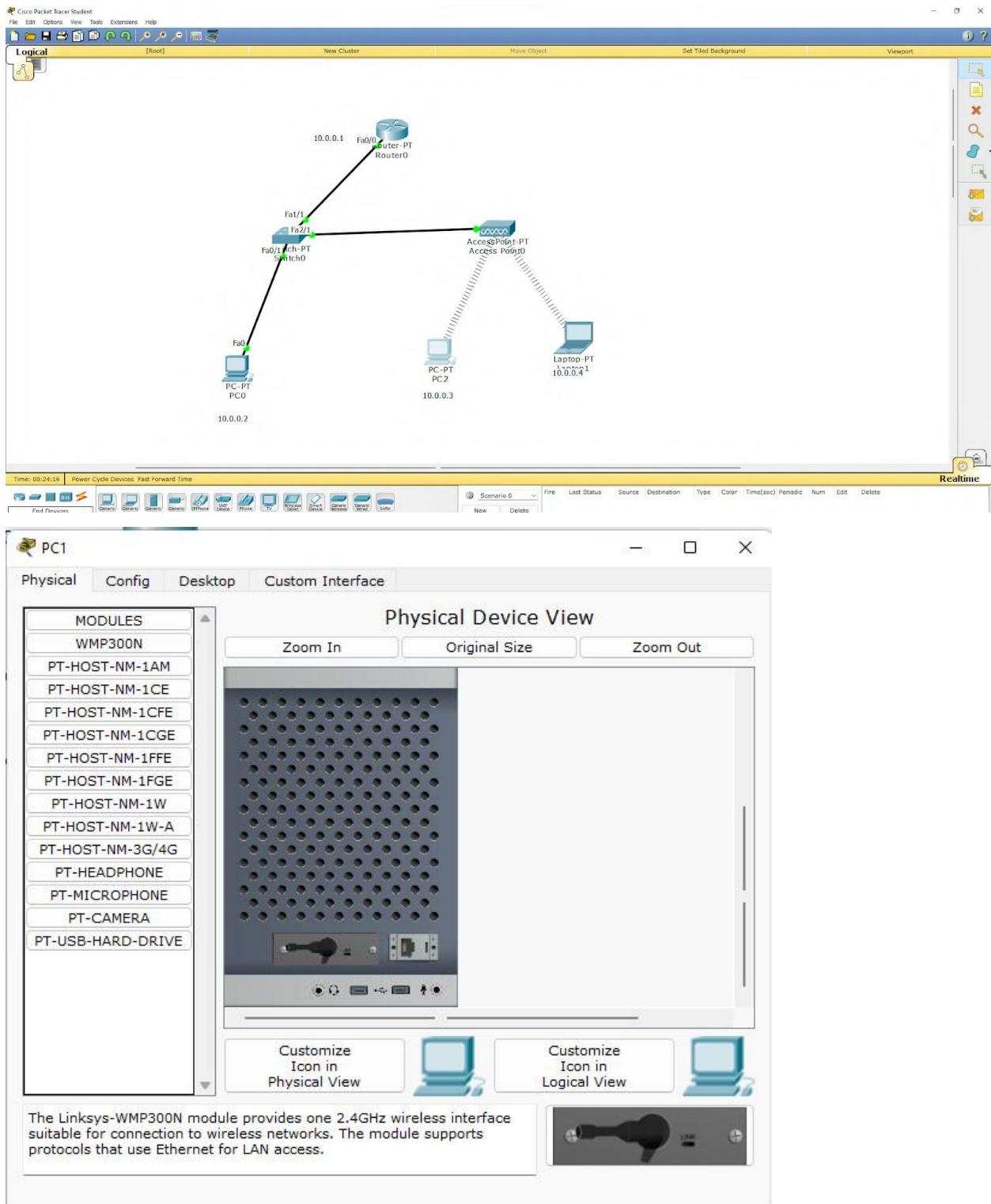
minimum = 6ms, Max = 21ms, Avg = 12 ms.

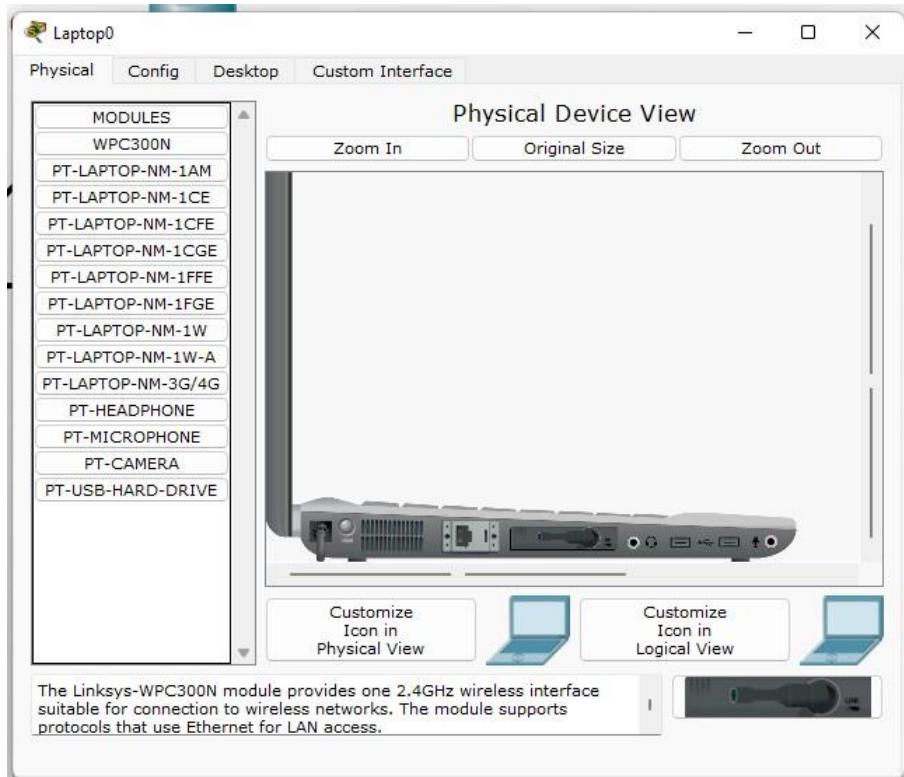
### Observation

Wireless local area network (WLAN) is a group of wirelessly connected computers or other devices that form a network based on radio transmission rather than wire connections.

After the WLAN is set up, the lined connection appears in the topology from the access point

## TOPOLOGY:





## OUTPUT:

The screenshot shows a terminal window titled "Command Prompt". The window has tabs at the top: Physical, Config, Desktop, and Custom Interface. The "Physical" tab is selected. The main area of the window displays the output of several ping commands. The output is as follows:

```
Packets: Sent = 4, Received = 0, Lost = 4 (100% loss),
PC>ping 10.0.0.3
Pinging 10.0.0.3 with 32 bytes of data:
Request timed out.
Request timed out.
Request timed out.
Request timed out.

Ping statistics for 10.0.0.3:
  Packets: Sent = 4, Received = 0, Lost = 4 (100% loss),
PC>ping 10.0.0.3
Pinging 10.0.0.3 with 32 bytes of data:
Reply from 10.0.0.3: bytes=32 time=21ms TTL=128
Reply from 10.0.0.3: bytes=32 time=7ms TTL=128
Reply from 10.0.0.3: bytes=32 time=9ms TTL=128
Reply from 10.0.0.3: bytes=32 time=10ms TTL=128

Ping statistics for 10.0.0.3:
  Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
  Approximate round trip times in milli-seconds:
    Minimum = 7ms, Maximum = 21ms, Average = 11ms
PC>
```

## WEEK 12

To understand the operation of TELNET by accessing the router in server room from a PC in IT office.

OBSERVATION:

Date 10/8/23  
Page 52

To understand the operation of TELNET by accessing the router in server room from a PC in IT office.

Aim : To understand the operation of TELNET by accessing the router in server room from a PC in IT office.

**Topology**

PC - PT  
PC 0  
(10.0.0.2)

Router - PT  
router 1  
10.0.0.1

**Procedure**

1. Create a topology as shown above.
2. Configure the IP address & gateway for PC 0
3. Configure the router by executing the following commands:
  1. Enable.
  2. config t
  3. Host name s1
  4. enable secret p1
  5. interface fastethernet 0/0
  6. ip address 10.0.0.1 255.0.0.0
  7. no shut
  8. time vty 05
  9. login
  10. password pa
  11. exit, Exit.
  12. w&

**Ping message to router.**

password for user verification is po  
password for enable is pl.

Accessing router CLI from PC  
show IP route.

### Ping output

Packet tracer PC command line 1.0

PC > ping 10.0.0.1

Pinging 10.0.0.1 with 32 bytes of data.

Reply from 10.0.0.1 bytes=32 time=10ms TTL=255

### Ping statistics for 10.0.0.1

Packets sent = 4 received = 4 lost = 0 (0% loss)

Approx round trip times in milliseconds.

minimum = 0ms , max = 0ms , Average = 0ms

PC > telnet 10.0.0.1

Typing 10.0.0.1 ... open

User access verification

password: po

Pl > enable

password: pl

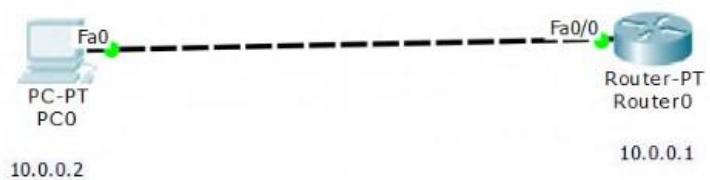
r1 # show ip route

C 10.0.0.0/8 is directly connected, fa 0/0.

### Observation

1. Telnet stands for Teletype, net work, it is a type of protocol that enables one comp to connect to local comp.
2. It is used as standard TCP/IP pro. for virtual terminal service provided by ISO.
3. During TELNET operation, whatever is being performed on the remote comp. will be displayed by local comp. Telnet

## TOPOLOGY:



## OUTPUT:

The screenshot shows the "Command Prompt" window of the Packet Tracer software. The window title is "Command Prompt". The content of the window is as follows:

```
Packet Tracer PC Command Line 1.0
PC>ping 10.0.0.1

Pinging 10.0.0.1 with 32 bytes of data:

Reply from 10.0.0.1: bytes=32 time=1ms TTL=255
Reply from 10.0.0.1: bytes=32 time=0ms TTL=255
Reply from 10.0.0.1: bytes=32 time=0ms TTL=255
Reply from 10.0.0.1: bytes=32 time=0ms TTL=255

Ping statistics for 10.0.0.1:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 0ms, Maximum = 1ms, Average = 0ms

PC>telnet 10.0.0.1
Trying 10.0.0.1 ...Open

User Access Verification

Password:
* Password: timeout expired!

[Connection to 10.0.0.1 closed by foreign host]
PC>telnet 10.0.0.1
Trying 10.0.0.1 ...Open

User Access Verification

Password:
Password:
Password:

[Connection to 10.0.0.1 closed by foreign host]
PC>telnet 10.0.0.1
Trying 10.0.0.1 ...Open

User Access Verification

Password:
rl>enable
Password:
rl#show ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
      D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
      N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
      E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
      i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, ia - IS-IS inter area
      * - candidate default, U - per-user static route, o - ODR
      P - periodic downloaded static route

Gateway of last resort is not set

C    10.0.0.0/8 is directly connected, FastEthernet0/0
rl#
```

## WEEK 13

Write a program for error detecting code using CRC- CCITT (16-bits).

OBSERVATION:

Date \_\_\_\_\_  
Page 38

write a program for error detecting code  
using CRC (16-bits)

```
#include <stdio.h>
#include <string.h>
#define N strlen (gen-poly)
char data [28];
char check-value [28];
char gen-poly [10];
int data-length, i, j;
void XOR() {
    for (j = 1; j < N; j++)
        check-value [j] = ((check-value [j]) == gen-poly[j])
            ? '0' : '1';
}
void receiver() {
    printf ("Enter the received data: ");
    scanf ("%s", data);
    printf ("Data received: %s", data);
    for (i = 0; i < N - 1) if (check-value [i] != '1') i++;
    if (i < N - 1)
        printf ("\nError detected\n\n");
    else
        printf ("\nError not detected\n\n");
}
void CRC() {
    for (i = 0; i < N; i++)
        check-value [i] = data [i];
    do {
        if (check-value [0] == '1')
            XOR();
        for (j = 0; j < N - 1; j++)
            check-value [j] = check-value [j + 1];
    } while (check-value [0] == '1');
}
```

```
check-value [j] = data [i++];  
3 while (i <= data - length + N - 1);  
3  
int main()  
{  
    printf ("\n Enter data to be transmitted :\n");  
    scanf ("%s", &data);  
    printf ("\n Enter the divisor polynomial :\n");  
    scanf ("%s", &gen-poly);  
    data-length = strlen (data);  
    for (i = data-length; i < data-length + N - 1; i++)  
        data [i] = 0;  
    printf ("\n Data padded with n-1 zeros : %s", data);  
    crc();  
    printf ("\n CRC value is : %s", check-value);  
    for (i = data-length; i < data-length + N - 1; i++)  
        data [i] = check-value [i - data-length];  
    printf ("\n Final codeword to be sent : %s", data);  
    printf ("\n ---\n");  
    receiver();  
    return 0;  
}
```

Output

Enter the dataword: 11001010111001001  
Calculated CRC = 111010010111001

OUTPUT:

```
C:\Users\Admin\Desktop\1BM21CS047\ADA\CRC16\bin\Debug\CRC16.exe
Enter the dataword
1 0 1 1 0 0 1 1 1 1 0 0 1 0 1 1 1
Enter dividend
1 0 0 0 1 0 0 0 0 0 1 0 0 0 1 1
Codeword: 1011001111001011100000000000011011
At receiver end
Codeword: 10110011110010111000000000000000
Process returned 1 (0x1) execution time : 49.507 s
Press any key to continue.
```

## WEEK 14

Write a program for congestion control using leaky bucket algorithm  
CODE:

Date / /  
Page 36

write a program for error congestion control  
using leaky bucket algorithms

```
#include <stdio.h>
int main()
{
    int incoming, outgoing, buck_size, n, store=0;
    printf("Enter the bucket size");
    scanf("%d", &buck_size);
    printf("Enter outgoing rate:");
    scanf("%d", &outgoing);
    printf("Enter the no. of inputs:");
    scanf("%d", &n);
    while (n != 0)
    {
        printf("Enter the incoming packet size");
        scanf("%d", &incoming);
        if (incoming <= (buck_size - store))
        {
            store += incoming;
            printf("Bucket buffer size %d out of %d\n", store, buck_size);
        }
        else
        {
            printf("Dropped %d no. of packets\n", incoming - (buck_size - store));
            printf("Bucket buffer size %d out of %d\n", store, buck_size);
            store = buck_size;
        }
        store = store - outgoing;
        printf("After outgoing %d packets left out of %d in buffer\n", store, buck_size);
        n--;
    }
}
```

### Output

Enter no. of queries, buffer size, input and output packet size

4

4

4

1

Packet is accepted

rem. space = 3

Packet is accepted

rem. space = 0

Packet not accepted.

Rem space 1

Packet not accepted

rem. space = 2

", store,

ing -

store,

+ of v.d

```
PS D:\VS Code> cd "d:\vs code\OS\" ; if ($?) { gcc bucket.c -o bucket } ; if ($?) { .\bucket }

Enter Bucket size and outstream size
2000
100
Packet of 41 bytes accepted
Remaining bytes: 2000
If you want to stop input, press 0, otherwise, press 1
1
Packet of 467 bytes accepted
Remaining bytes: 1633
If you want to stop input, press 0, otherwise, press 1
1
Packet of 334 bytes accepted
Remaining bytes: 1399
If you want to stop input, press 0, otherwise, press 1
1
Packet of 500 bytes accepted
Remaining bytes: 999
If you want to stop input, press 0, otherwise, press 1
1
Packet of 169 bytes accepted
Remaining bytes: 930
If you want to stop input, press 0, otherwise, press 1
1
Packet of 724 bytes accepted
Remaining bytes: 306
If you want to stop input, press 0, otherwise, press 1
1
Packet of 478 bytes is discarded
Remaining bytes: 406
If you want to stop input, press 0, otherwise, press 1
1
Packet of 358 bytes accepted
Remaining bytes: 148
If you want to stop input, press 0, otherwise, press 1
1
Packet of 962 bytes is discarded
Remaining bytes: 248
If you want to stop input, press 0, otherwise, press 1
0
Remaining bytes: 348
Remaining bytes: 448
Remaining bytes: 548
Remaining bytes: 648
Remaining bytes: 748
```

```
Remaining bytes: 348
Remaining bytes: 448
Remaining bytes: 548
Remaining bytes: 648
Remaining bytes: 748
Remaining bytes: 848
Remaining bytes: 948
Remaining bytes: 1048
Remaining bytes: 1148
Remaining bytes: 1248
Remaining bytes: 1348
Remaining bytes: 1448
Remaining bytes: 1548
Remaining bytes: 1648
Remaining bytes: 1748
Remaining bytes: 1848
Remaining bytes: 1948
Remaining bytes: 2000
PS D:\vs Code\os> █
```

# WEEK 15

Using TCP/IP sockets, write a client-server program to make client sending the filename and the server to send back the contents of the requested file if present.

OBSERVATION:

Date 21/08/23  
Page 40

1. - to send back the contents of the requested file if present  
Using TCP/IP sockets, write a client-server program to make client sending the file name to the server  
Solution

Client TCP .py

```
from socket import *
serverName = "127.0.0.1"
serverPort = 12000
clientSocket = Socket (AF_INET, SOCK_STREAM)
clientSocket .connect ((serverName , serverPort))
sentence = input (" \n enter file Name ")
clientSocket .send (sentence .encode ())
fileContents = clientSocket .recv (1024) .decode ()
print (" \n from server: \n " + fileContents)
clientSocket .close ()
```

Server TCP .py

```
from socket import *
serverName = "127.0.0.1"
serverPort = 12000
serverSocket = socket (AF_INET, SOCK_STREAM)
serverSocket .bind ((serverName , serverPort))
serverSocket .listen (1)
while True:
    print ("The server is ready to receive")
    connectionSocket, addr = serverSocket .accept ()
    sentence = connectionSocket .recv (1024) .decode ()
    file = open (sentence, "r")
    f = file.read (1024)
    connectionSocket .send (f .encode ())
    print ("\n Sent contents of " + sentence)
    file .close ()
```

Output

= Restart : c:/users/Admin/AppData/Local/Programs/  
Python/Python310/server-tcp.py

The Server is ready to receive.  
Sent contents of server-tcp.py

The server is ready to receive.

= Restart : c:/users/Admin/AppData/Local/Programs/  
Python/Python310/client-tcp.py

enter file name : server-tcp.py

from socket import \*

Server Name = "127.0.0.1"

Server Port = 12000

Server Socket = socket (AF\_INET, SOCK\_STREAM)

>Contents sent by the server displayed here.

+-----+  
| 127.0.0.1 | 12000 |

+-----+  
| 127.0.0.1 | 12000 |

+-----+  
| 127.0.0.1 | 12000 |

+-----+  
| 127.0.0.1 | 12000 |

+-----+  
| 127.0.0.1 | 12000 |

+-----+  
| 127.0.0.1 | 12000 |

+-----+  
| 127.0.0.1 | 12000 |

+-----+  
| 127.0.0.1 | 12000 |

+-----+  
| 127.0.0.1 | 12000 |

+-----+  
| 127.0.0.1 | 12000 |

+-----+  
| 127.0.0.1 | 12000 |

+-----+  
| 127.0.0.1 | 12000 |

+-----+  
| 127.0.0.1 | 12000 |

+-----+  
| 127.0.0.1 | 12000 |

+-----+  
| 127.0.0.1 | 12000 |

+-----+  
| 127.0.0.1 | 12000 |

+-----+  
| 127.0.0.1 | 12000 |

+-----+  
| 127.0.0.1 | 12000 |

## OUTPUT:

The screenshot shows a Windows desktop environment with four windows open:

- server-tcp.py**: A code editor window containing the Python server code. It prints "The server is ready to receive" and reads from a file named 'sentence'.
- client-tcp.py**: A code editor window containing the Python client code. It connects to the server, sends a file name, receives contents, and prints them.
- IDLE Shell 3.10.8**: An interactive Python shell window where the server script is run, showing it's ready to receive. It then receives the client's file content and prints it.
- IDLE Shell 3.10.8**: Another interactive Python shell window where the client script is run, showing the connection to the server and the received file content.

The desktop taskbar at the bottom shows icons for File Explorer, Task View, and other standard Windows applications.

```
server-tcp.py - C:/Users/Admin/AppData/Local/Programs/Python/Python310/server-tcp.py (3.10.8)
File Edit Format Run Options Window Help
from socket import *
serverName="127.0.0.1"
serverPort = 12000
serverSocket = socket(AF_INET,SOCK_STREAM)
serverSocket.bind((serverName,serverPort))
serverSocket.listen(1)
while 1:
    print ("The server is ready to receive")
    connectionSocket, addr = serverSocket.accept()
    sentence = connectionSocket.recv(1024).decode()
    file=open(sentence,"r")
    l=file.read(1024)
    connectionSocket.send(l.encode())
    connectionSocket.send(l.encode())
    print ('\nSent contents of ' + sentence)
    file.close()
    connectionSocket.close()

client-tcp.py - C:/Users/Admin/AppData/Local/Programs/Python/Python310/client-tcp.py (3.10.8)
File Edit Format Run Options Window Help
from socket import *
serverName = '127.0.0.1'
serverPort = 12000
clientSocket = socket(AF_INET, SOCK_STREAM)
clientSocket.connect((serverName,serverPort))
sentence = input("\nEnter file name: ")
clientSocket.send(sentence.encode())
filecontents = clientSocket.recv(1024).decode()
print ('\nFrom Server:\n')
print(filecontents)
clientSocket.close()

IDLE Shell 3.10.8
File Edit Shell Debug Options Window Help
Python 3.10.8 (tags/v3.10.8:aaaf517, Oct 11 2022, 16:50:30) [MSC v.1933 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> = RESTART: C:/Users/Admin/AppData/Local/Programs/Python/Python310/server-tcp.py
The server is ready to receive
Sent contents of server-tcp.py
The server is ready to receive

IDLE Shell 3.10.8
File Edit Shell Debug Options Window Help
Python 3.10.8 (tags/v3.10.8:aaaf517, Oct 11 2022, 16:50:30) [MSC v.1933 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> Enter file name: server-tcp.py
From Server:
from socket import *
serverName="127.0.0.1"
serverPort = 12000
serverSocket = socket(AF_INET,SOCK_STREAM)
serverSocket.bind((serverName,serverPort))
serverSocket.listen(1)
while 1:
    print ("The server is ready to receive")
    connectionSocket, addr = serverSocket.accept()
    sentence = connectionSocket.recv(1024).decode()
    file=open(sentence,"r")
    l=file.read(1024)
    connectionSocket.send(l.encode())
    connectionSocket.send(l.encode())
    print ('\nSent contents of ' + sentence)
    file.close()
    connectionSocket.close()
```

## WEEK 16

Using UDP sockets, write a client-server program to make the client send the file name and the server to send back the contents of the requested file if present.

OBSERVATION:

Date / /  
Page 92

2. Using UDP sockets, write a client - server program to make client sending the file name to server to send back contents of requested file if present.

→ client UDP . Py

```
from socket import *
serverName = "127.0.0.1"
serverPort = 12000
clientSocket = socket(AF_INET, SOCK_DGRAM)
sentence = input("\n Enter file name: ")
clientSocket.sendto(sentence.encode("utf-8"), (serverName, serverPort))
filecontents, serverAddress = clientSocket.recvfrom(2048)
print("\n Reply from server: \n")
print(filecontents.decode("utf-8"))
# for i in filecontents:
#     print(str(i), end="")
clientSocket.close()
clientSocket.close()
```

server UDP . Py

```
from socket import *
serverPort = 12000
serverSocket = socket(AF_INET, SOCK_DGRAM)
serverSocket.bind(("127.0.0.1", serverPort))
print("The server is ready to receive")
while True:
    sentence, clientAddress = serverSocket.recvfrom(2048)
    sentence = sentence.decode("utf-8")
    file = open(sentence, "r")
    con = file.read(2048)
    serverSocket.sendto(con.encode("utf-8"), clientAddress)
    print("\n sent contents of ", end="")
    print(sentence)
```

```
# for i in sentence:  
#     print lstr(i), end=""  
file.close()
```

## Output

The server is ready to receive.  
sent contents of server UDP - py

Client sends message to server. Server is ready to receive.

client server is ready to receive.

The contents requested by server is displayed.

100

# OUTPUT:

The image shows four windows from the Python IDLE interface. The top-left window contains the code for `serverudp.py`, which creates a UDP server at port 12000. It reads a sentence from a file, encodes it to UTF-8, and sends it to a client. The top-right window contains the code for `clientudp.py`, which connects to the server at port 12000, receives the sentence, decodes it from UTF-8, and prints it. The bottom-left window is an IDLE Shell session where the server code is run, and the bottom-right window shows the output of the client code, which prints "The server is ready to receive".

```
serverudp.py - C:/Users/Admin/AppData/Local/Programs/Python/Python311/serverudp.py (3.11.4)
File Edit Format Run Options Window Help
from socket import *
serverPort = 12000
serverSocket = socket(AF_INET, SOCK_DGRAM)
serverSocket.bind(("127.0.0.1", serverPort))
print ("The server is ready to receive")
while 1:
    sentence, clientAddress = serverSocket.recvfrom(2048)
    sentence = sentence.decode("utf-8")
    file=open(sentence, "r")
    con=file.read(2048)
    serverSocket.sendto(bytes(con,"utf-8"),clientAddress)
    print ('\nSent contents of', end = ' ')
    print (sentence)
    # for i in sentence:
    #     print (str(i), end = "")
    file.close()

clientudp.py - C:/Users/Admin/Desktop/serupy/clientudp.py (3.11.4)
File Edit Format Run Options Window Help
from socket import *
serverName = "127.0.0.1"
serverPort = 12000
clientSocket = socket(AF_INET, SOCK_DGRAM)
sentence = input("\nEnter file name:")
clientSocket.sendto(bytes(sentence,"utf-8"),(serverName, serverPort))
filecontents,serverAddress = clientSocket.recvfrom(2048)
print ("\nReply from Server:\n")
print (filecontents.decode("utf-8"))
# for i in filecontents:
#     print(str(i), end = "")
clientSocket.close()
clientSocket.close()

Ln:14 Col:0

IDLEShell 3.11.4
File Edit Shell Debug Options Window Help
Python 3.11.4 (tags/v3.11.4:d239ef, Jun  7 2023, 05:45:37) [MSC v.1934 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> ===== RESTART: C:/Users/Admin/Desktop/serupy/clientudp.py =====

Enter file name:serverudp.py
Reply from Server:
from socket import *
serverPort = 12000
serverSocket = socket(AF_INET, SOCK_DGRAM)
serverSocket.bind(("127.0.0.1", serverPort))
print ("The server is ready to receive")
while 1:
    sentence, clientAddress = serverSocket.recvfrom(2048)
    sentence = sentence.decode("utf-8")
    file=open(sentence,"r")
    con=file.read(2048)
    serverSocket.sendto(bytes(con,"utf-8"),clientAddress)
    print ('\nSent contents of', end = ' ')
    print (sentence)
    # for i in sentence:
    #     print (str(i), end = "")
    file.close()

>>>

IDLEShell 3.11.4*
File Edit Shell Debug Options Window Help
Python 3.11.4 (tags/v3.11.4:d239ef, Jun  7 2023, 05:45:37) [MSC v.1934 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> = RESTART: C:/Users/Admin/AppData/Local/Programs/Python/Python311/serverudp.py =
The server is ready to receive
Sent contents of serverudp.py
```

# WEEK 17

## Tool Exploration - Wireshark

### OBSERVATION:

Date 24/08/23  
Page 49

Write on Wireshark - Tool Exploration - Wireshark

Wireshark is an open source packet analyzer which is used for education analysis, software development and network communication protocol development and network troubleshooting. It is used to track packets so that each one is filtered to meet our specific needs. It is commonly called as a sniffer, network protocol analyzer, network analyzer. It is also used by network security engineers to examine security problems.

Wireshark is a free application used to apprehend data back and forth. It is also called as a free packet sniffer computer application, puts network card into a unselective mode i.e. to accept all packets which it receives.

Uses

- It is used by network security engineers to examine security problems.
- It is used by network engineers to troubleshoot network uses.
- It is also used to analyze dropped packets.
- It helps to troubleshoot latency malicious activities on the network.
- It helps us to know how all devices like laptop, mobile phones, desktop switch routers communicate in a local network or the rest of the world.

Functionality of wireshark.

It is similar to a TCP dump in networking. It has a graphic, end, sort and filtering functions. It

also monitors the unicast traffic which is not sent to network's MAC address interface. The port mirroring is a method to monitor the network traffic. When it is enabled switch sends copies of all network packets present at one port to another port.

#### Features of Wireshark:

- It is a multiplatform software i.e. it can run on the Linux, Windows, OS X, True BSD, NetBSD, etc.
- It is a standard three pane packet browser.
- It performs deep inspection of kind of protocols.
- It even has sort and filter option which makes ease to user to view the data.
- It can capture raw USB traffic.
- It is useful in IP analysis.
- It also involves live analysis i.e. from different types of network like Ethernet, loopback etc. through which we can read live data.