

**VISVESVARAYA TECHNOLOGICAL
UNIVERSITY**

“JnanaSangama”, Belgaum -590014, Karnataka.



**LAB REPORT
on
COMPUTER NETWORKS**

Submitted by

PRITHVI PRAKASH S (1BM21CS265)

*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 **PRITHVI PRAKASH S (1BM21CS265)**, 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.

SOWMYA T
Assistant Professor
Department of CSE
BMSCE, Bengaluru

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

Index

Sl. No.	Date	Experiment Title	Page No.
CYCLE 1			
1	16/6/23	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	23/6/23	Configure IP address to routers in packet tracer. Explore the following messages: ping responses, destination unreachable, request timed out, reply.	9
3	14/7/23	Configure default route, static route to the Router.	18
4	14/7/23	Configure DHCP within a LAN and outside LAN.	23
5	21/7/23	Configure Web Server, DNS within a LAN.	32
6	21/7/23	Configure RIP routing Protocol in Routers.	35
7	28/7/23	Configure OSPF routing protocol.	40
8	4/8/23	To construct simple LAN and understand the concept and operation of Address Resolution Protocol (ARP).	45
9	11/8/23	To construct a VLAN and make a pc communicate among VLAN.	49
10	12/8/23	Demonstrate the TTL/ Life of a Packet.	53
11	11/8/23	To construct a WLAN and make the nodes communicate wirelessly.	58
12	11/8/23	To understand the operation of TELNET by accessing the router in server room from a PC in IT office.	62
CYCLE 2			
13	18/8/23	Write a program for error detecting code using CRC CCITT (16-bits).	66
14	18/8/23	Write a program for congestion control using Leaky bucket algorithm.	72
15	25/8/23	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.	76
16	25/8/23	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.	80
17	1/9/23	Tool Exploration -Wireshark	84

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

Experiment - I - Week I

Title: X Messages

Create a topology consisting and simulate PDU from source to destination using hub and switch as connecting device and demonstrate ping message. Title:

Aim: To send messages from source to destination using hub and switch.

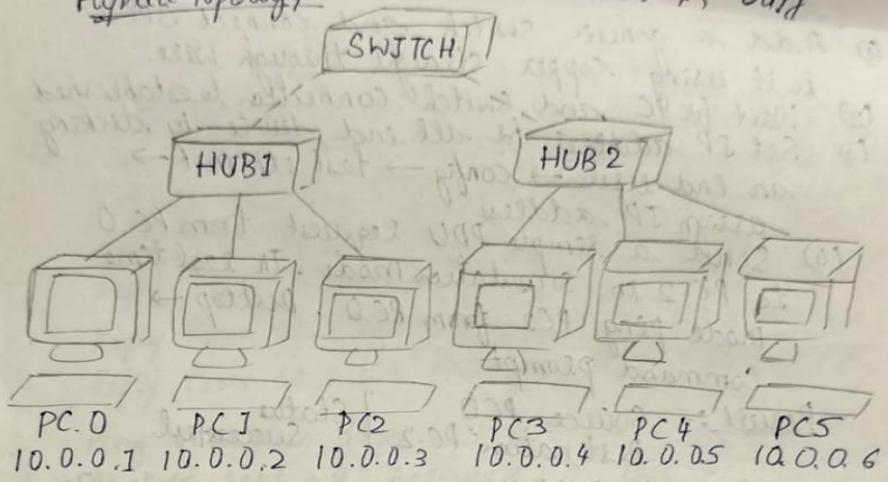
Procedure:

- ① Add a client end device and a web server end device.
- ② Connect both using a Cross-over cable
- ③ Set the clients DNS server to 192.168.0.105
- ④ Set the IP Address binder the Fast Ethernet to 192.168.0.110
- ⑤ Select the web server and IP address is to be set to 192.168.0.105 and add.
- ⑥ Select the DNS services and set the domain name as "www.firstlab.com" and IP address as 192.168.0.105 and add.
- ⑦ Ensure DNS service is ON.
- ⑧ Add Simple PDU tool is used to send a simple message to server and vice versa. The log values are displayed in the PDU list window.

PC-PT
Client
192.168.0.110

SERVER-PT
WEB SERVER.
192.168.0.105

Hybrid topology



Procedure

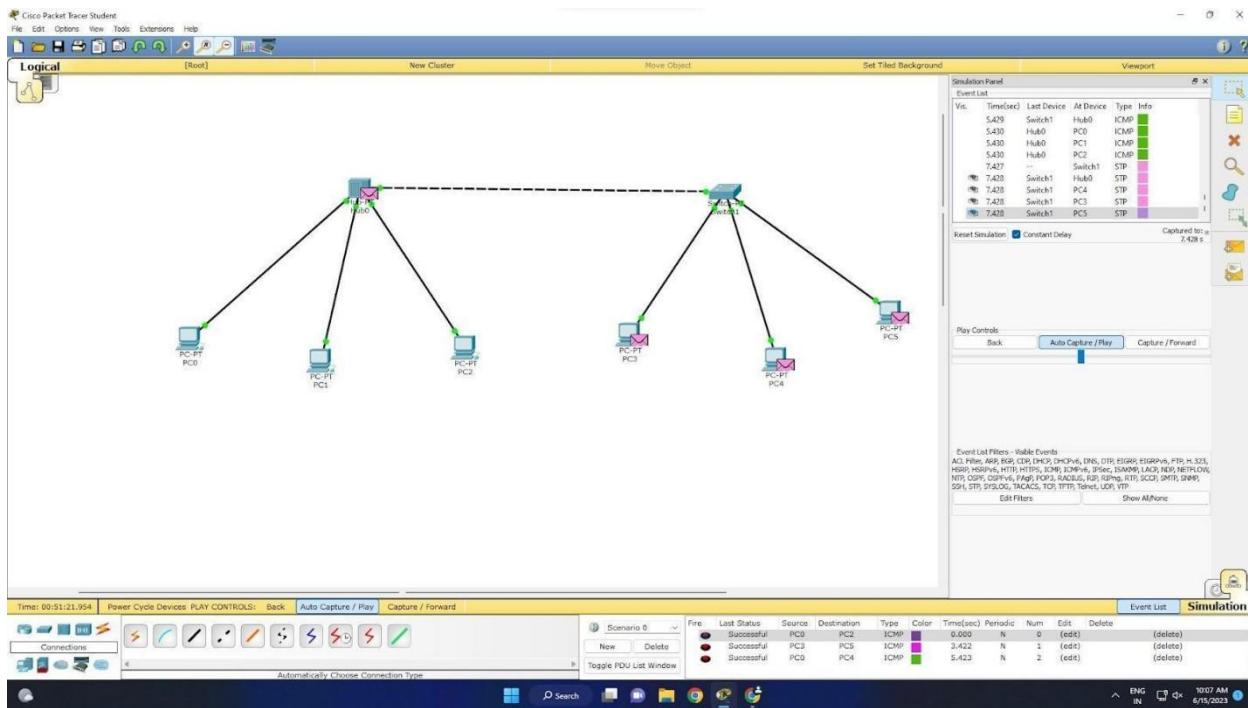
- (I) Add generic hub, generic switch and connect 3 end devices here [PC]
- (II) Connect end devices to the two different hubs then connect 2 hubs to a switch using copper cross over wire
- (III) Click on end device → Config → fast ethernet → assign IP address.
- (IV) Send PDU from source device to destination and wait for simulation to finish.
- (V) To real time mode ping devices using command prompt.

Result

Source PC0] Status
Destination PC5] Successful.

Ping	10.0.0.6	from 10.0.0.6	byte = 32	time = 0ms	TTL = 128
Reply					
Reply					
Reply					

TOPOLOGY:

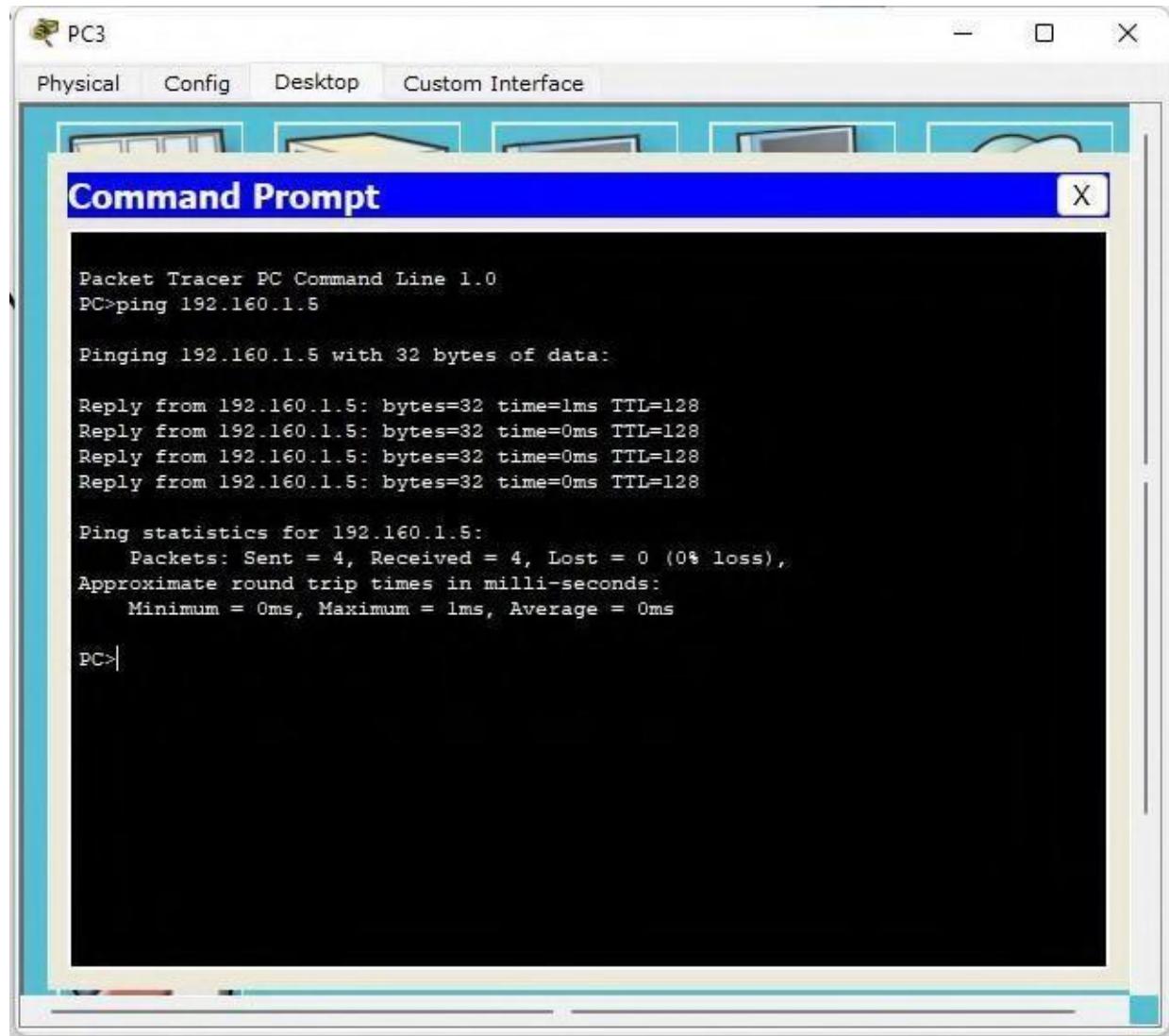


OUTPUT:

```

Packets Tracer SC Command Line 1.0
PC>ping 192.168.1.5 with 32 bytes of data:
Reply from 192.168.1.5: bytes=32 time=0ms TTL=128
Ping statistics for 192.168.1.5:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 0ms, Maximum = 0ms, Average = 0ms
PC>ping 192.168.1.6 with 32 bytes of data:
Request timed out.
Request timed out.
Request timed out.
Request timed out.
Ping statistics for 192.168.1.6:
    Packets: Sent = 4, Received = 0, Lost = 4 (100% loss),
PC>192.168.1.5
Invalid Command.
PC>ping 192.168.1.5
Ping 192.168.1.5 with 32 bytes of data:
Reply from 192.168.1.5: bytes=32 time=0ms TTL=128
Ping statistics for 192.168.1.5:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 0ms, Maximum = 0ms, Average = 0ms
PC>

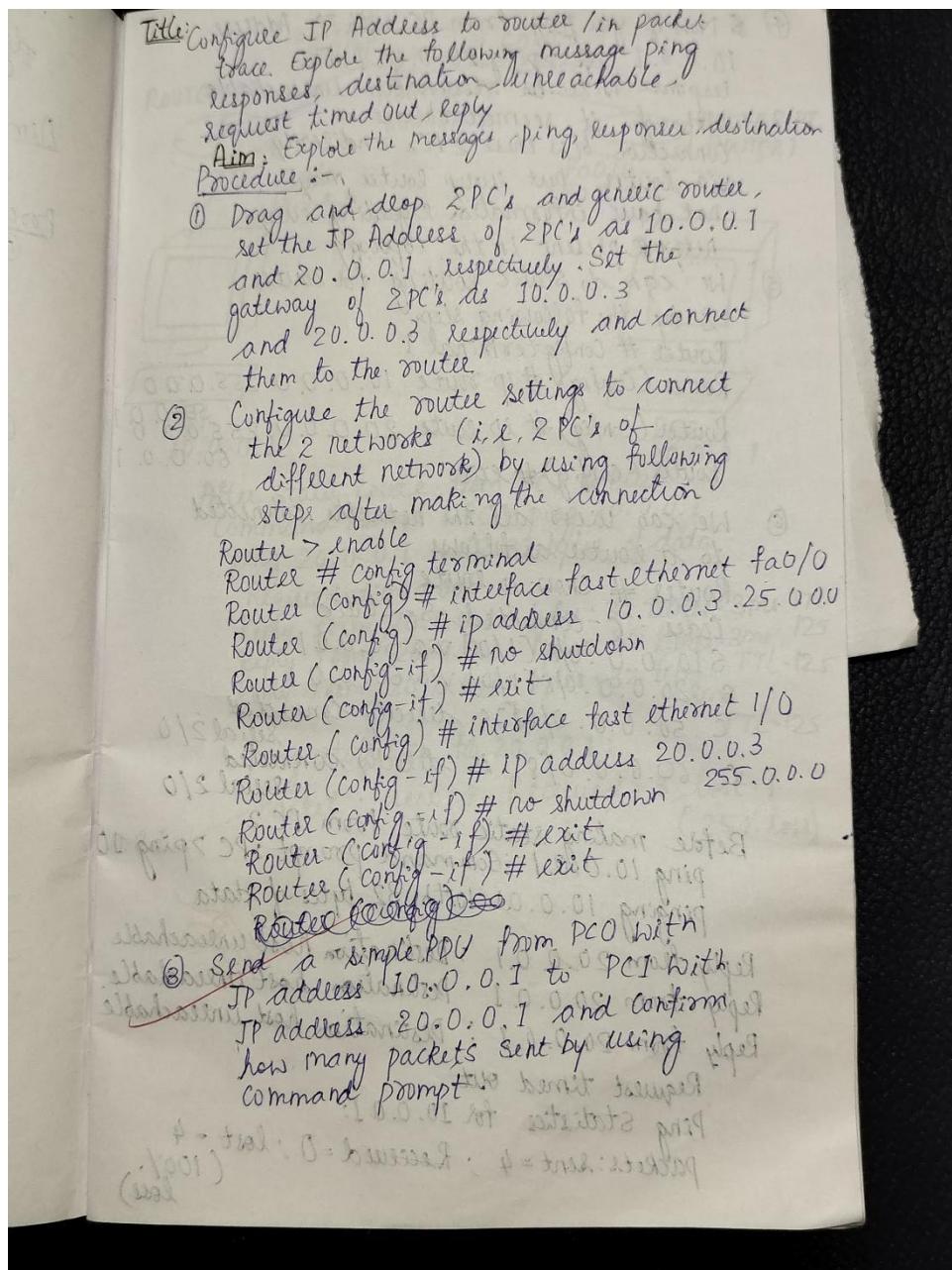
```



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:



Title: Create a topology consisting of 2 devices connected with the help of a router.

Aim :- Connecting 2 devices with the help of a router.

Procedure :-
① Drag and drop 2 generic PC's and a router. Connect 2 PCs as peripherals to router after setting the IP addresses as 10.0.0.1 and 20.0.0.1 for PCs and PC respectively and connect them.

② Configure router → enable → Configure terminal → interface fa0/0 → IP address (PC)
(subnet mask) → no shutdown

③ Repeat this for PC₂.
④ Ping from PC₁ to PC₂ and note the observation give the gateway and ping again and notedown observation

Observation :-

PC > ping 10.0.0.1
ping 10.0.0.1 with 32 bytes of data

Request timed out.

Reply from 10.0.0.1 : bytes : 32 time = 3ms TTL=125

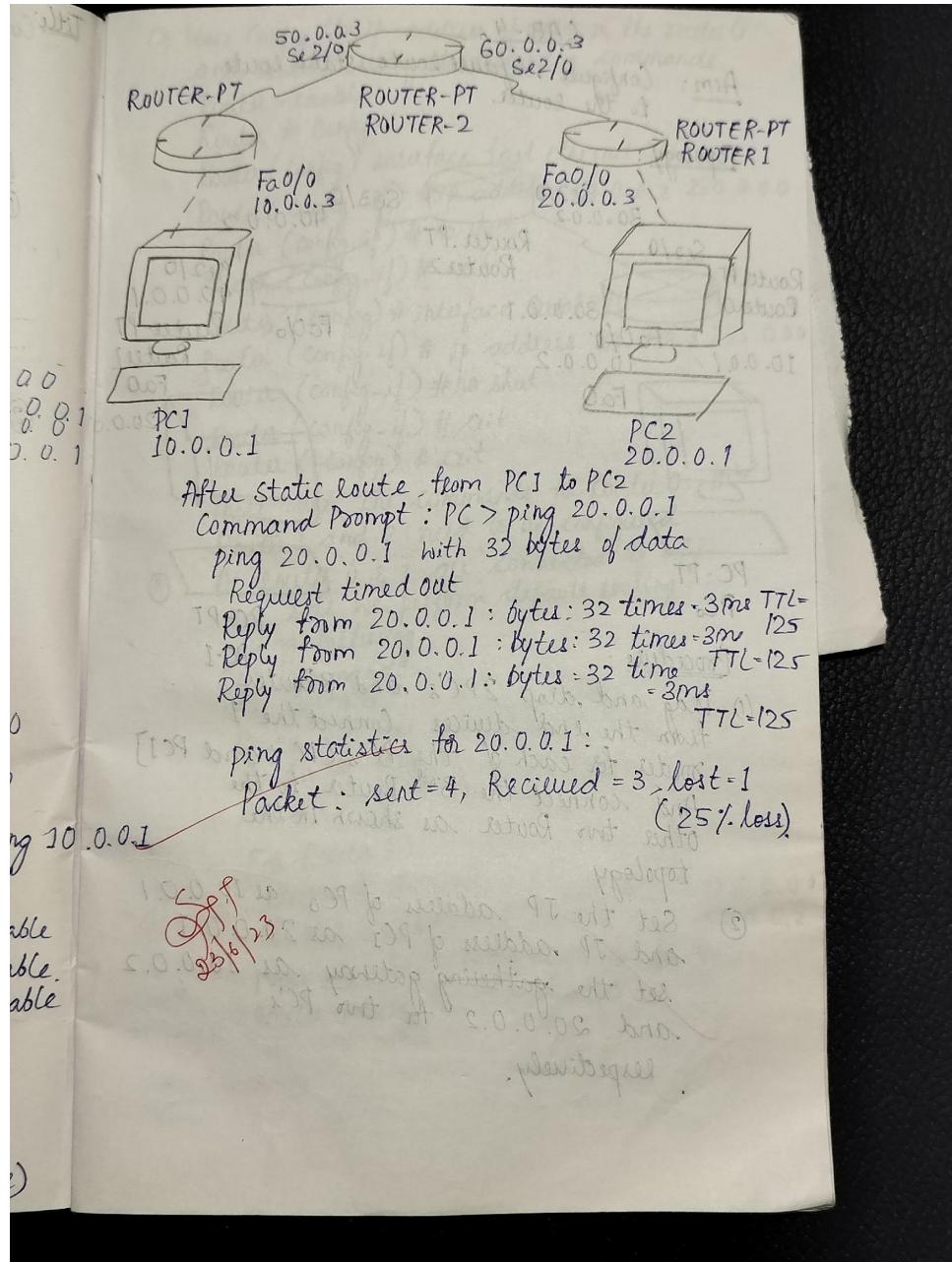
Reply from 10.0.0.1 : bytes : 32 time = 3ms TTL=125

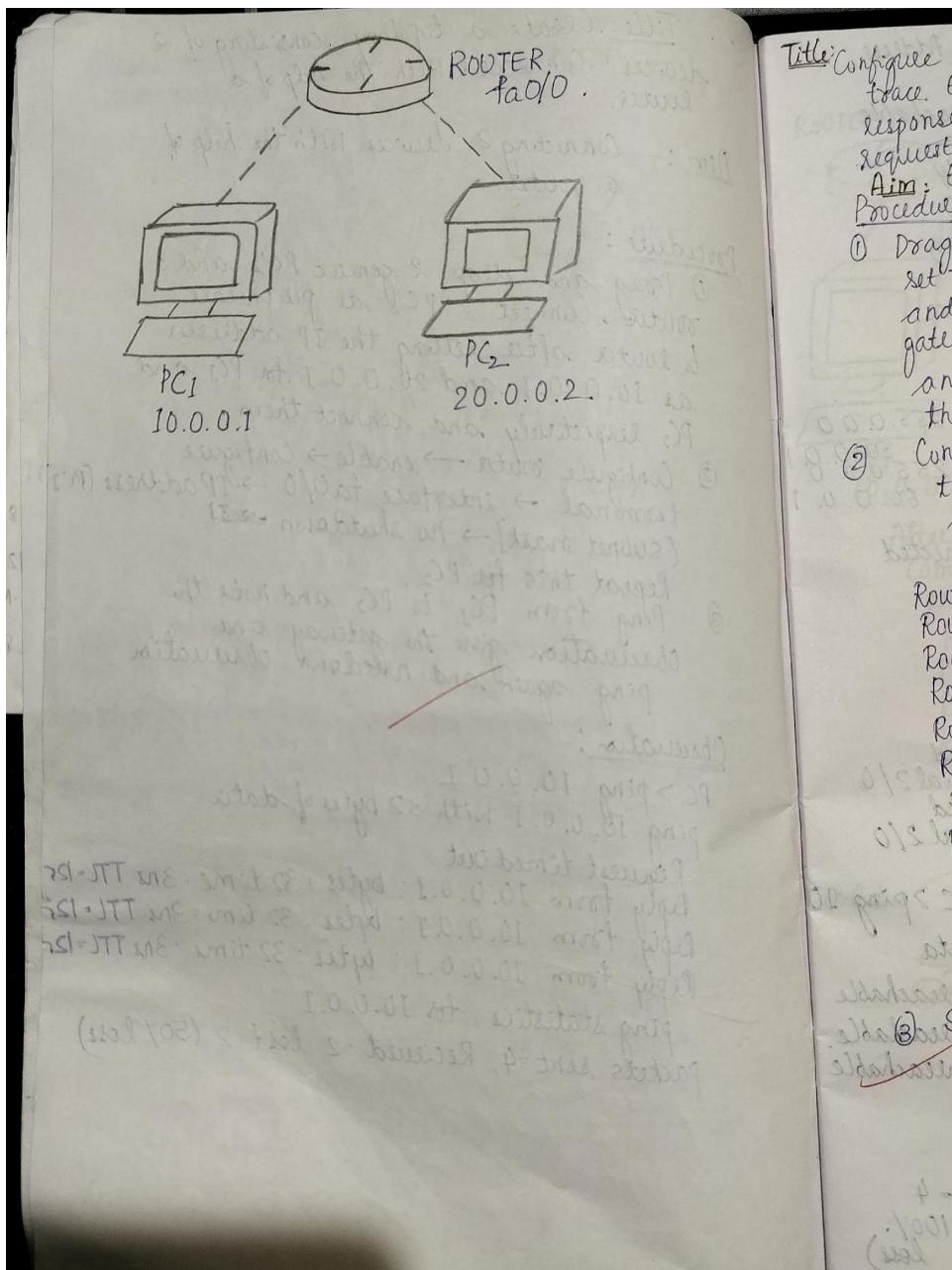
Reply from 10.0.0.1 : bytes : 32 time = 3ms TTL=125

Reply from 10.0.0.1 : bytes : 32 time = 3ms TTL=125

ping statistics for 10.0.0.1

packets sent = 4, Received = 2 lost = 2 (50% loss)





Title: Configure trace route request

Aim:

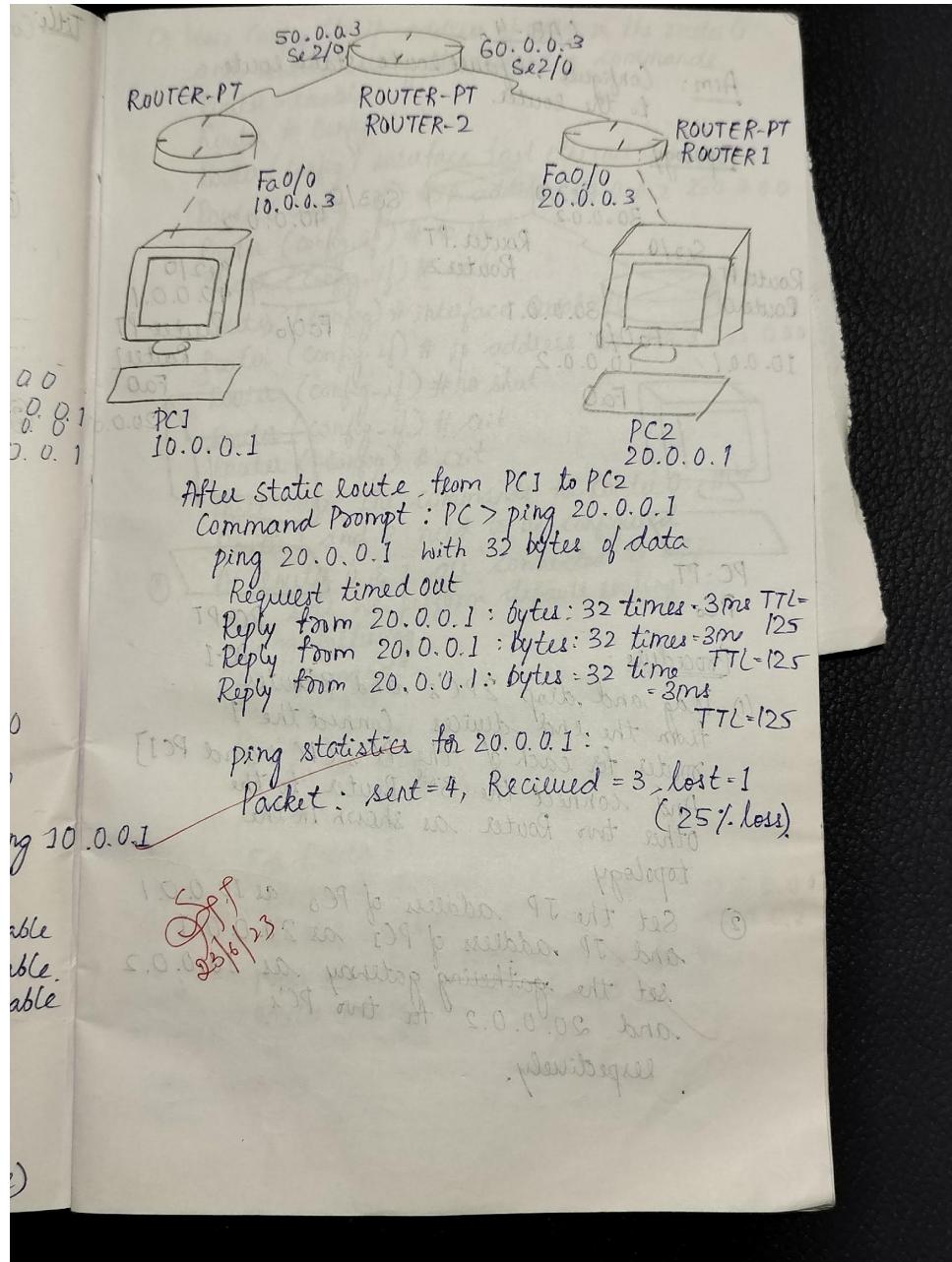
Procedures

① Drag set and gather and the

② Conf th

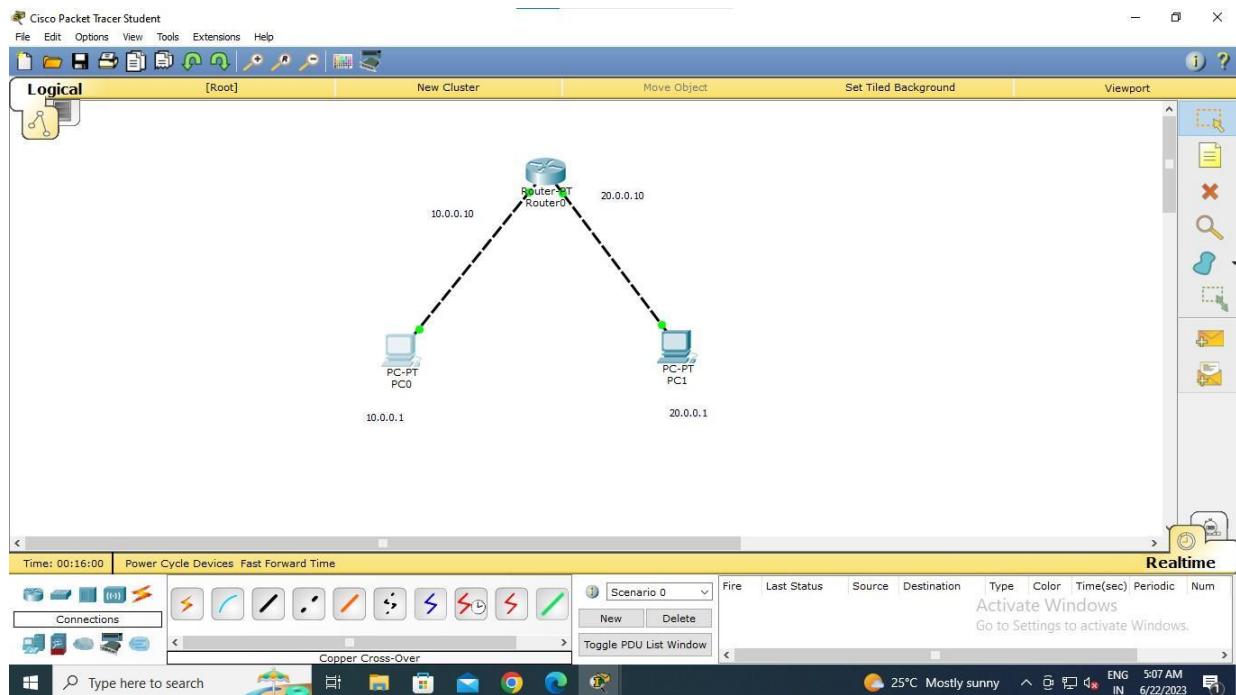
Rout

Rou

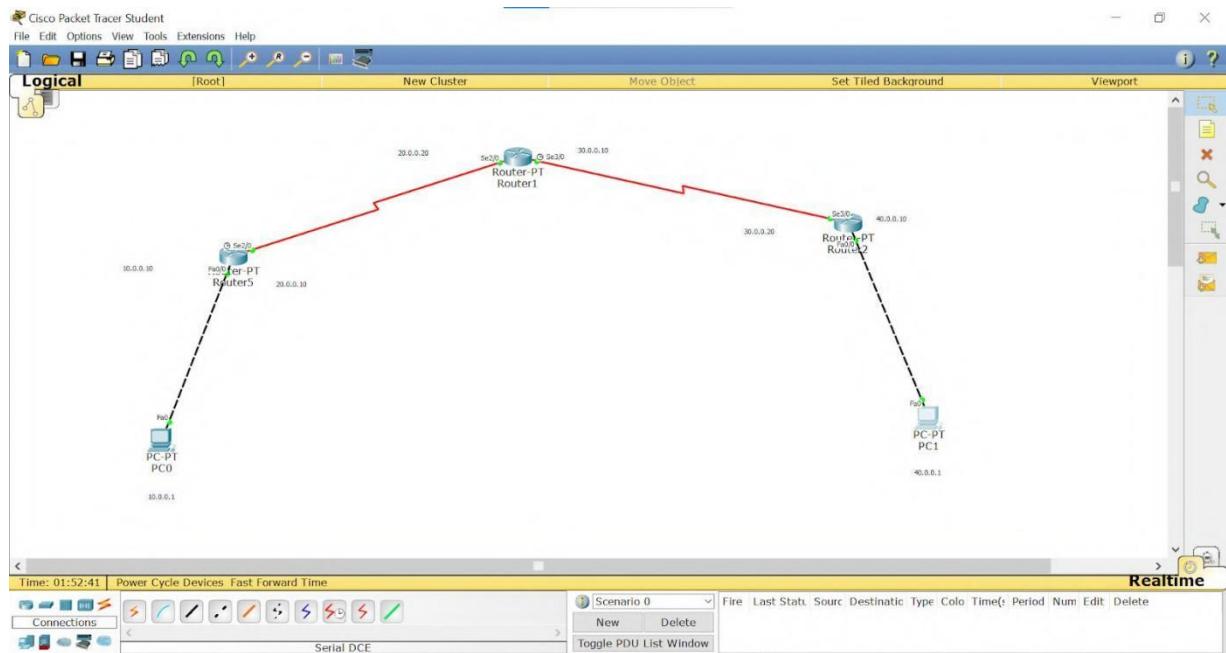


TOPOLOGY:

PROGRAM 2.1



PROGRAM 2.2



OUTPUT:

PROGRAM 2.1

The screenshot shows the Cisco Packet Tracer interface. At the top, there's a toolbar with icons for Physical, Config, Desktop, and Custom Interface. Below the toolbar is a menu bar with File, Edit, Options, View, Tools, Extensions, and Help. The main area displays a network diagram with three nodes: Router0 (IP 10.0.0.10), PC-PT PC0 (IP 10.0.0.1), and PC-PT PC1 (IP 20.0.0.10). Router0 is connected to both PCs via dashed lines. On the right side of the screen, there's a Simulation Panel with an Event List table and various controls like Back, Auto Capture / Play, and Capture / Forward. The Event List table shows several CDP entries. At the bottom, there's a taskbar with icons for Connections, Power Cycle Devices, PLAY CONTROLS (Back, Auto Capture / Play, Capture / Forward), Scenario 0 (New, Delete), and a Simulation panel showing a single ICMP entry. The desktop background is visible at the very bottom.

```

Packet Tracer PC Command Line 1.0
PC>ping 20.0.0.1

Pinging 20.0.0.1 with 32 bytes of data:

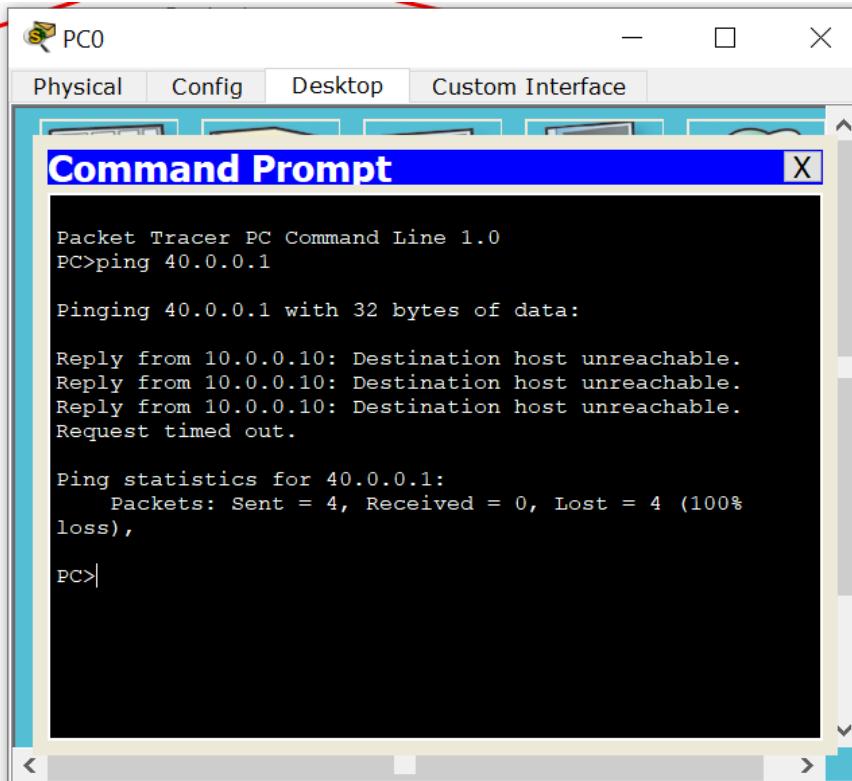
Request timed out.
Reply from 20.0.0.1: bytes=32 time=0ms TTL=127
Reply from 20.0.0.1: bytes=32 time=0ms TTL=127
Reply from 20.0.0.1: bytes=32 time=10ms TTL=127

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>

```

PROGRAM 2.2



PC0

Physical Config Desktop Custom Interface

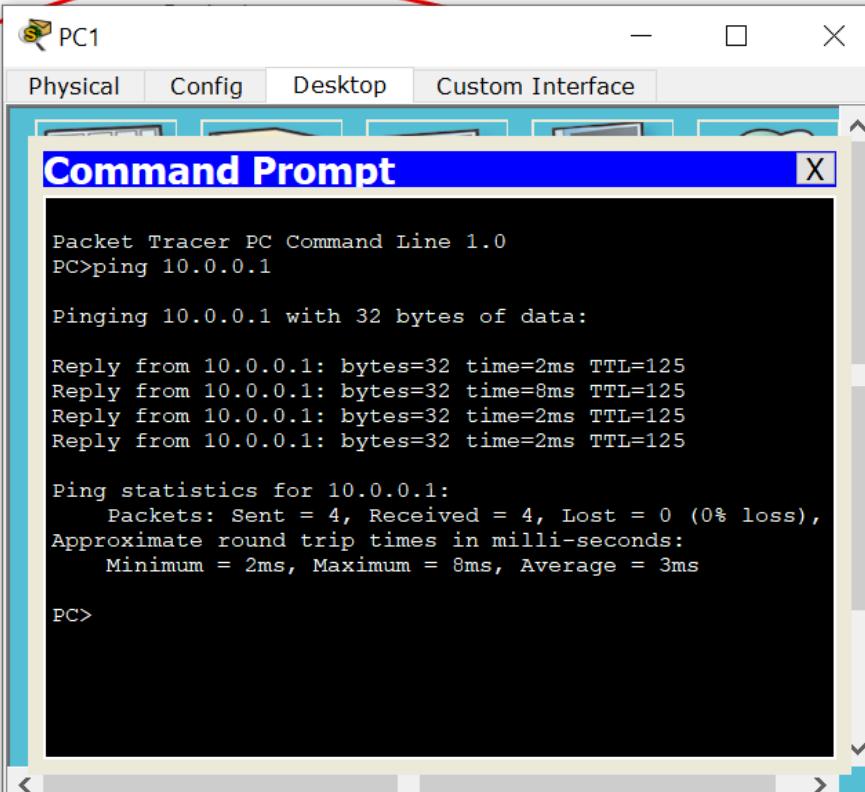
Command Prompt

```
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

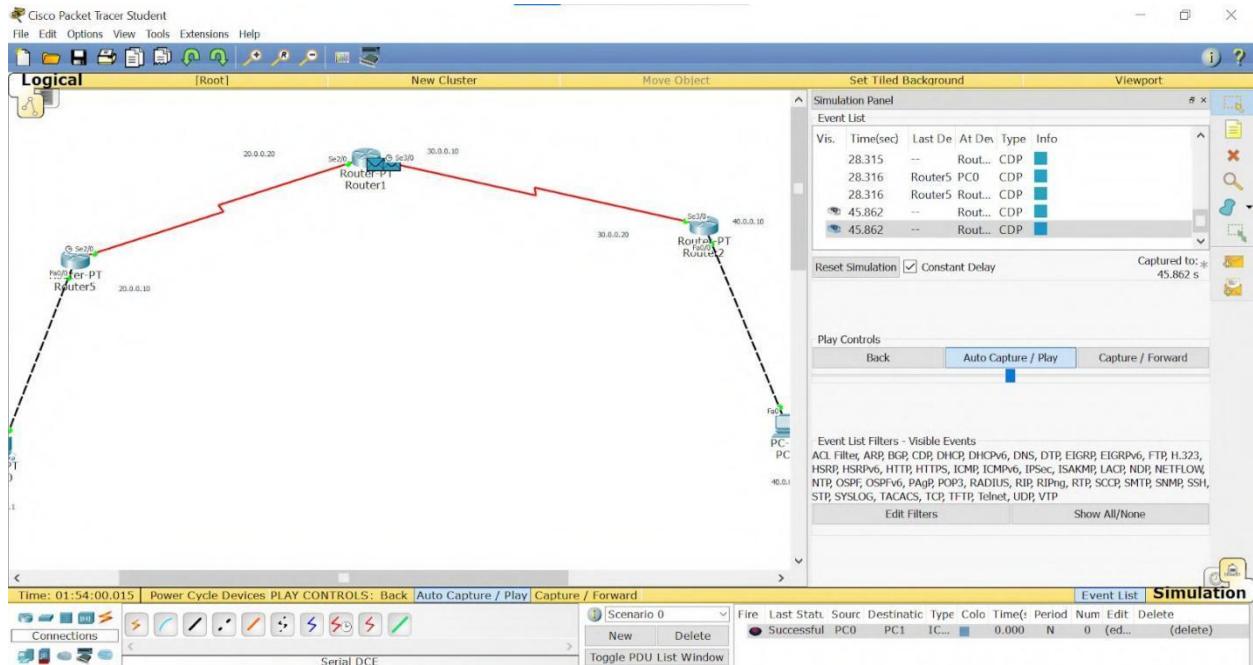
Command Prompt

```
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

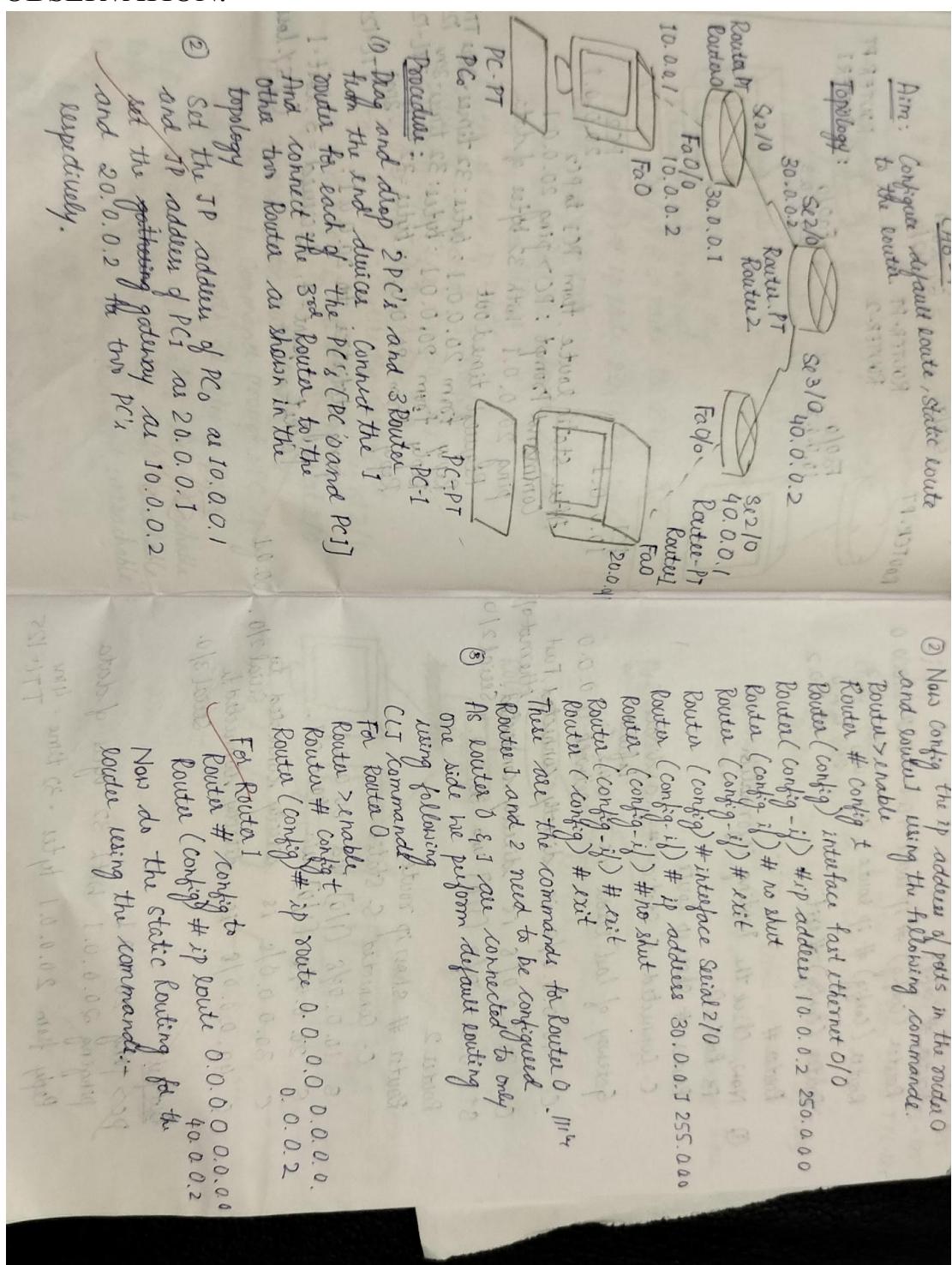
Ping statistics for 10.0.0.1:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 2ms, Maximum = 8ms, Average = 3ms
PC>
```



WEEK 3

Configure default route, static route to the Router.

OBSERVATION:



- ② Now config the IP address of ports in the router and switch using the following command.

Router>enable
Router# config t
Router(Config-if)# ip address 10.0.0.1 255.0.0.0

Router(Config-if)# no shutdown
Router(Config-if)# exit

Router(Config)# interface serial 2/0
Router(Config-if)# ip address 30.0.0.1 255.0.0.0

Router(Config-if)# no shutdown
Router(Config-if)# exit

Router(Config)# interface fast ethernet 0/0
Router(Config-if)# ip address 10.0.0.2 255.0.0.0

Router(Config-if)# no shutdown
Router(Config-if)# exit

Router# config t
Router(Config)# ip route 0.0.0.0 0.0.0.0 10.0.0.1
Router(Config)# exit

These are the commands for Router 0.
Router 1 and 2 need to be configured similarly. These are connected to only one side we perform default routing using following CLI commands.

CLI Commands:
Router# config t
Router(Config)# interface fast ethernet 0/0
Router(Config-if)# ip address 10.0.0.2 255.0.0.0
Router(Config-if)# no shutdown
Router(Config-if)# exit

Router# config t
Router(Config-if)# ip address 40.0.0.1 255.0.0.0
Router(Config-if)# no shutdown
Router(Config-if)# exit

Router# config t
Router(Config-if)# ip address 40.0.0.2 255.0.0.0
Router(Config-if)# no shutdown
Router(Config-if)# exit

For Router 1

Router# config t
Router(Config)# interface fast ethernet 0/0
Router(Config-if)# ip address 10.0.0.1 255.0.0.0
Router(Config-if)# no shutdown
Router(Config-if)# exit

Now do the static routing for the Router using the command.

Router# config t
Router(Config)# ip route 0.0.0.0 0.0.0.0 10.0.0.1
Router(Config)# exit

Router# config t
Router(Config)# ip route 0.0.0.0 0.0.0.0 10.0.0.2
Router(Config)# exit

Router # config
 Router (config) # ip route 10.0.0.0 255.0.0.0
 Router (config) # ip route 20.0.0.0 30.0.0.2
 255.0.0.0
 40.0.0.2
 Router (config) # exit

Router #

Now, check the routing information

Router #

sho ip route

C- Connected S-Static *- Candidate

Gateway of last resort is 30.0.0.2 to network 0.0.0.0

C 10.0.0.0/8 is directly connected, Fast

C 30.0.0.0/8 is directly connected, Serial 2/0
S* 0.0.0.0/0 [1/0] via 30.0.0.1 Serial 2/0

Router 2

Router # show ip route

C- Connected S-Static *- Candidate

Gateway of last resort is 30.0.0.2 to network 0.0.0.0

C 10.0.0.0/8 [1/0] via 30.0.0.1

S* 20.0.0.0/8 [1/0] via 40.0.0.1.

C 30.0.0.0/8 is directly connected to Serial 2/0.

0.0.0.0/0 via 40.0.0.1 is directly connected to Serial 3/0.

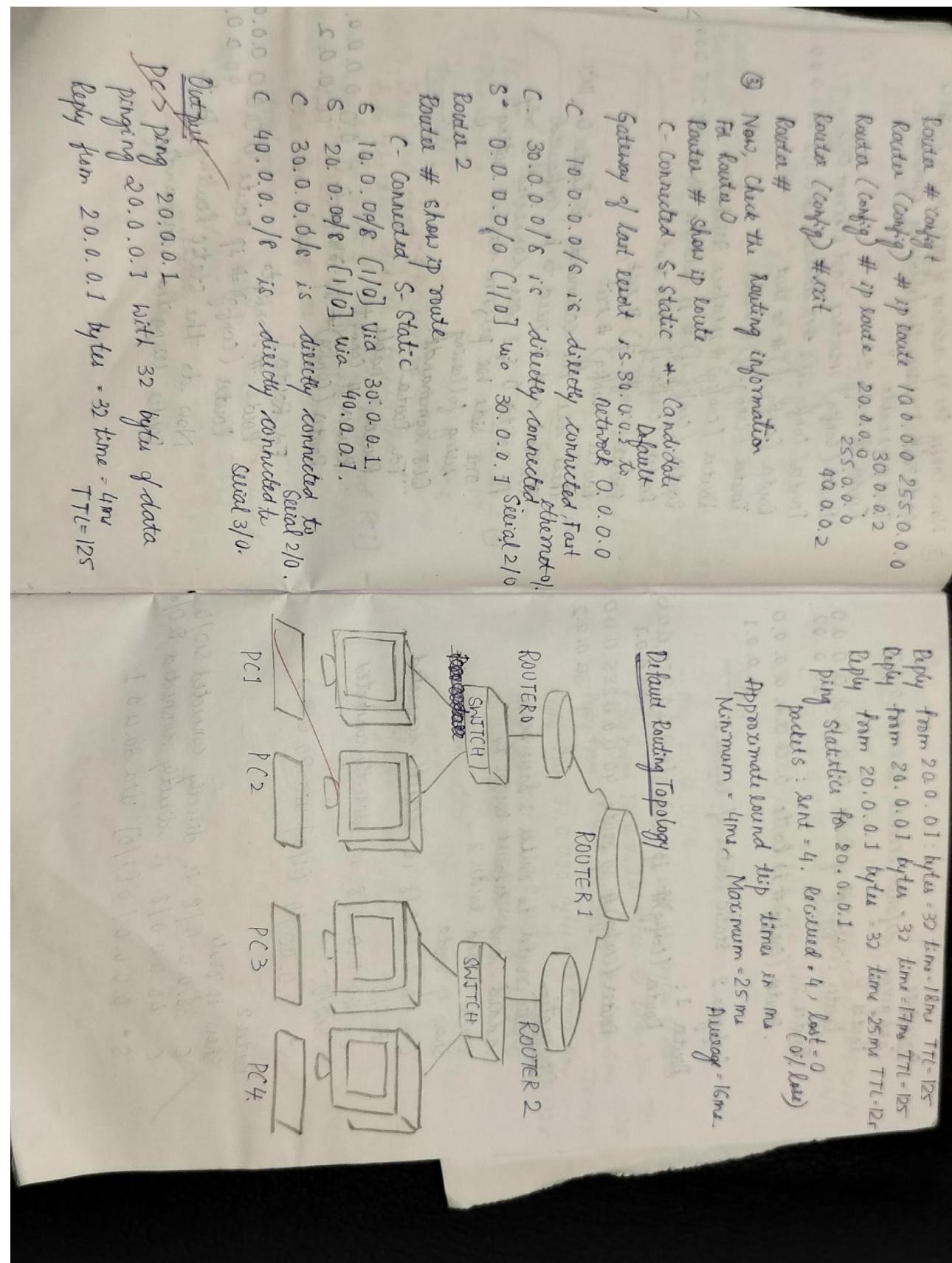
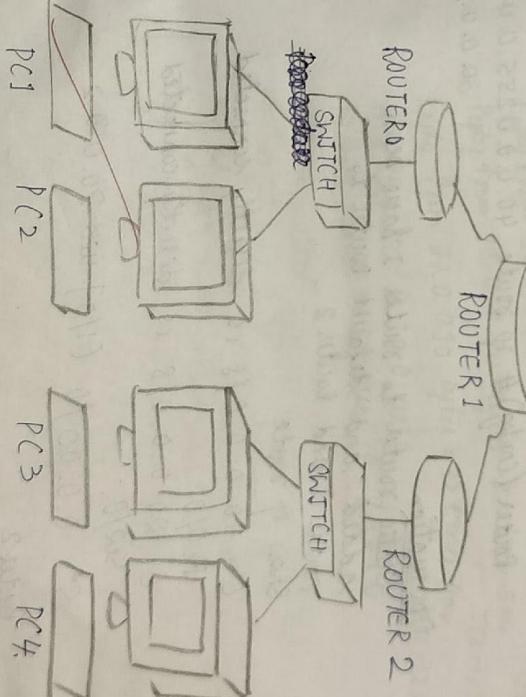
Output

PING 20.0.0.1 with 32 bytes of data
 Pinging 20.0.0.1 with 32 bytes of data
 Reply from 20.0.0.1 bytes = 32 time = 4ms
 TTL=125

Reply from 20.0.0.1 bytes = 32 time = 18ms TTL=125
 Reply from 20.0.0.1 bytes = 32 time = 17ms TTL=125
 Reply from 20.0.0.1 bytes = 32 time = 25ms TTL=125
 ping statistics for 20.0.0.1:
 packets: sent = 4, received = 4, lost = 0 (0% loss)

Approximate round trip time is ms.
 Minimum = 4ms, Maximum = 25ms
 Average = 16ms

Default Routing Topology



Procedure default route to route 0 & 2

Configuring

Route 1 ip route 0.0.0.0 0.0.0.0

Route 2 ip route 0.0.0.0 0.0.0.0

Route 3 ip route 0.0.0.0 0.0.0.0

Route 4 ip route 0.0.0.0 0.0.0.0

Route 5 ip route 0.0.0.0 0.0.0.0

Route 6 ip route 0.0.0.0 0.0.0.0

Route 7 ip route 0.0.0.0 0.0.0.0

Route 8 ip route 0.0.0.0 0.0.0.0

Route 9 ip route 0.0.0.0 0.0.0.0

Route 10 ip route 0.0.0.0 0.0.0.0

Route 11 ip route 0.0.0.0 0.0.0.0

Route 12 ip route 0.0.0.0 0.0.0.0

Route 13 ip route 0.0.0.0 0.0.0.0

Route 14 ip route 0.0.0.0 0.0.0.0

Route 15 ip route 0.0.0.0 0.0.0.0

Route 16 ip route 0.0.0.0 0.0.0.0

Route 17 ip route 0.0.0.0 0.0.0.0

Route 18 ip route 0.0.0.0 0.0.0.0

Route 19 ip route 0.0.0.0 0.0.0.0

Route 20 ip route 0.0.0.0 0.0.0.0

Route 21 ip route 0.0.0.0 0.0.0.0

Route 22 ip route 0.0.0.0 0.0.0.0

Route 23 ip route 0.0.0.0 0.0.0.0

Route 24 ip route 0.0.0.0 0.0.0.0

Route 25 ip route 0.0.0.0 0.0.0.0

Route 26 ip route 0.0.0.0 0.0.0.0

Route 27 ip route 0.0.0.0 0.0.0.0

Route 28 ip route 0.0.0.0 0.0.0.0

Route 29 ip route 0.0.0.0 0.0.0.0

Route 30 ip route 0.0.0.0 0.0.0.0

Route 31 ip route 0.0.0.0 0.0.0.0

Route 32 ip route 0.0.0.0 0.0.0.0

Route 33 ip route 0.0.0.0 0.0.0.0

Route 34 ip route 0.0.0.0 0.0.0.0

Route 35 ip route 0.0.0.0 0.0.0.0

Route 36 ip route 0.0.0.0 0.0.0.0

Route 37 ip route 0.0.0.0 0.0.0.0

Route 38 ip route 0.0.0.0 0.0.0.0

Route 39 ip route 0.0.0.0 0.0.0.0

Route 40 ip route 0.0.0.0 0.0.0.0

Route 41 ip route 0.0.0.0 0.0.0.0

Route 42 ip route 0.0.0.0 0.0.0.0

Route 43 ip route 0.0.0.0 0.0.0.0

Route 44 ip route 0.0.0.0 0.0.0.0

Route 45 ip route 0.0.0.0 0.0.0.0

Route 46 ip route 0.0.0.0 0.0.0.0

Route 47 ip route 0.0.0.0 0.0.0.0

Route 48 ip route 0.0.0.0 0.0.0.0

Route 49 ip route 0.0.0.0 0.0.0.0

Route 50 ip route 0.0.0.0 0.0.0.0

Route 51 ip route 0.0.0.0 0.0.0.0

Route 52 ip route 0.0.0.0 0.0.0.0

Route 53 ip route 0.0.0.0 0.0.0.0

Route 54 ip route 0.0.0.0 0.0.0.0

Route 55 ip route 0.0.0.0 0.0.0.0

Route 56 ip route 0.0.0.0 0.0.0.0

Route 57 ip route 0.0.0.0 0.0.0.0

Route 58 ip route 0.0.0.0 0.0.0.0

Route 59 ip route 0.0.0.0 0.0.0.0

Route 60 ip route 0.0.0.0 0.0.0.0

Route 61 ip route 0.0.0.0 0.0.0.0

Route 62 ip route 0.0.0.0 0.0.0.0

Route 63 ip route 0.0.0.0 0.0.0.0

Route 64 ip route 0.0.0.0 0.0.0.0

Route 65 ip route 0.0.0.0 0.0.0.0

Route 66 ip route 0.0.0.0 0.0.0.0

Route 67 ip route 0.0.0.0 0.0.0.0

Route 68 ip route 0.0.0.0 0.0.0.0

Route 69 ip route 0.0.0.0 0.0.0.0

Route 70 ip route 0.0.0.0 0.0.0.0

Route 71 ip route 0.0.0.0 0.0.0.0

Route 72 ip route 0.0.0.0 0.0.0.0

Output

Request from PC4

IP ping

10.0.0.10

from 10.0.0.10

with 32 bytes of data.

Reply from 10.0.0.10

bytes = 32 time = 5ms

TTL = 125

Reply from 10.0.0.10

bytes = 32 time = 4ms

TTL = 125

Ping statistics for 10.0.0.10

Packets sent = 4

Packets lost = 0

Approximate round trip times in ms:

Min = 4ms

Max = 5ms

Avg = 4.5ms

StdDev = 0.5ms

Range = 1ms

Loss = 0.0%

Round trip times in ms:

Min = 4ms

Max = 5ms

Avg = 4.5ms

StdDev = 0.5ms

Range = 1ms

Loss = 0.0%

Round trip times in ms:

Min = 4ms

Max = 5ms

Avg = 4.5ms

StdDev = 0.5ms

Range = 1ms

Loss = 0.0%

Procedure

default route to router 0 & 2

Configuring

Router 0

Router 1

Router (config)# ip route 0.0.0.0 0.0.0.0

20.0.0.2

Router 2

Router (config)# ip route 0.0.0.0 0.0.0.0

30.0.0.1

Configur 2. static routes of Router 1

Router 1

Router (config)# ip route 10.0.0.0 255.0.0.0

20.0.0.1

Router (config)# ip route 40.0.0.0 255.0.0.0

30.0.0.2

Observation
Static route to Router 1 have been
added and default route to
Router 0 and Router 2.

Show ip route

C 10.0.0.0/8 is directly connected

Fo 0/0

C 20.0.0.0/8 is directly connected

Se 2/0

S* 0.0.0.0/0 [1/0] via 20.0.0.2

Router 2

Show ip route

C 30.0.0.0/8 is directly connected Fo 0/0

C 40.0.0.0/8 is directly connected Se 2/0

S* 0.0.0.0/0 [1/0] via 30.0.0.1

Output
Ping request from PC4

1000 Ping to 10.0.0.10

Transmitting ping 10.0.0.10 with 32 bytes of data

Time = 9ms TTL = 125

Reply from 10.0.0.10 bytes = 32 time = 5ms TTL = 125

Reply from 10.0.0.10 bytes = 32 time = 4ms TTL = 125

Reply from 10.0.0.10 bytes = 32 time = 4ms TTL = 125

Ping statistics for 10.0.0.10

bytes = 4 lost = 0

Packets : sent = 4 received = 4 lost = 0

Router 1

Show ip route

S 10.0.0.0/8 [1/0] via 20.0.0.1

S 20.0.0.0/8 is directly connected Fo 0/0

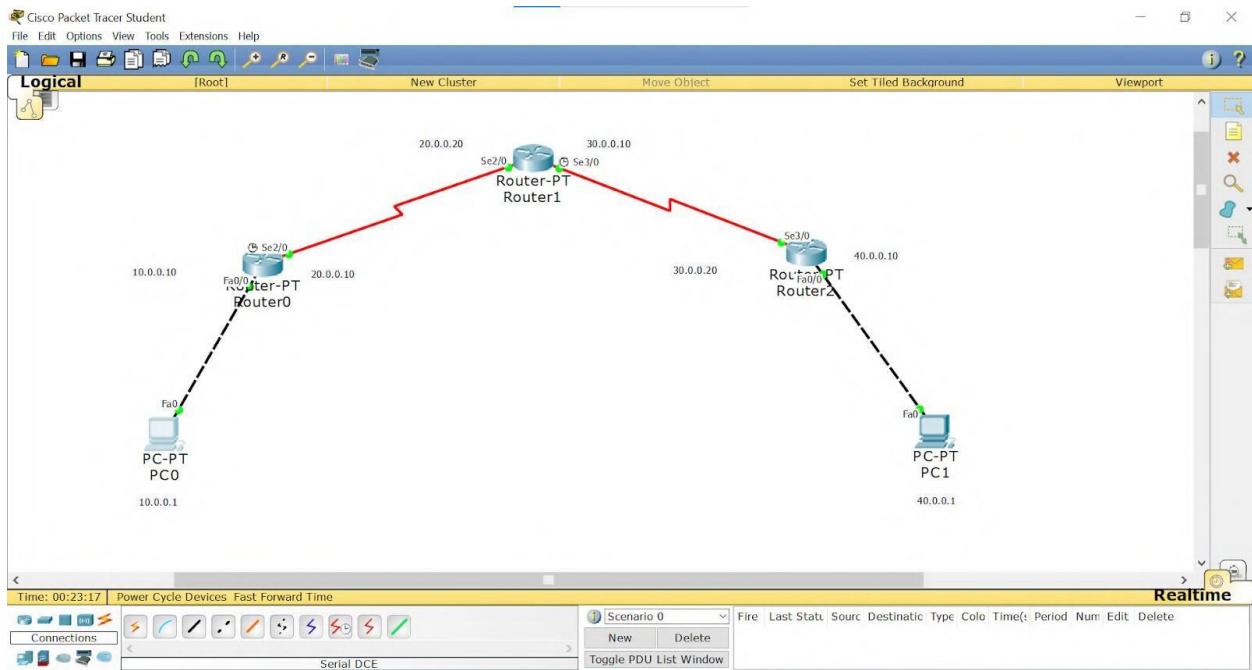
C 30.0.0.0/8 is directly connected Se 2/0

C 40.0.0.0/8 is directly connected Fo 0/0

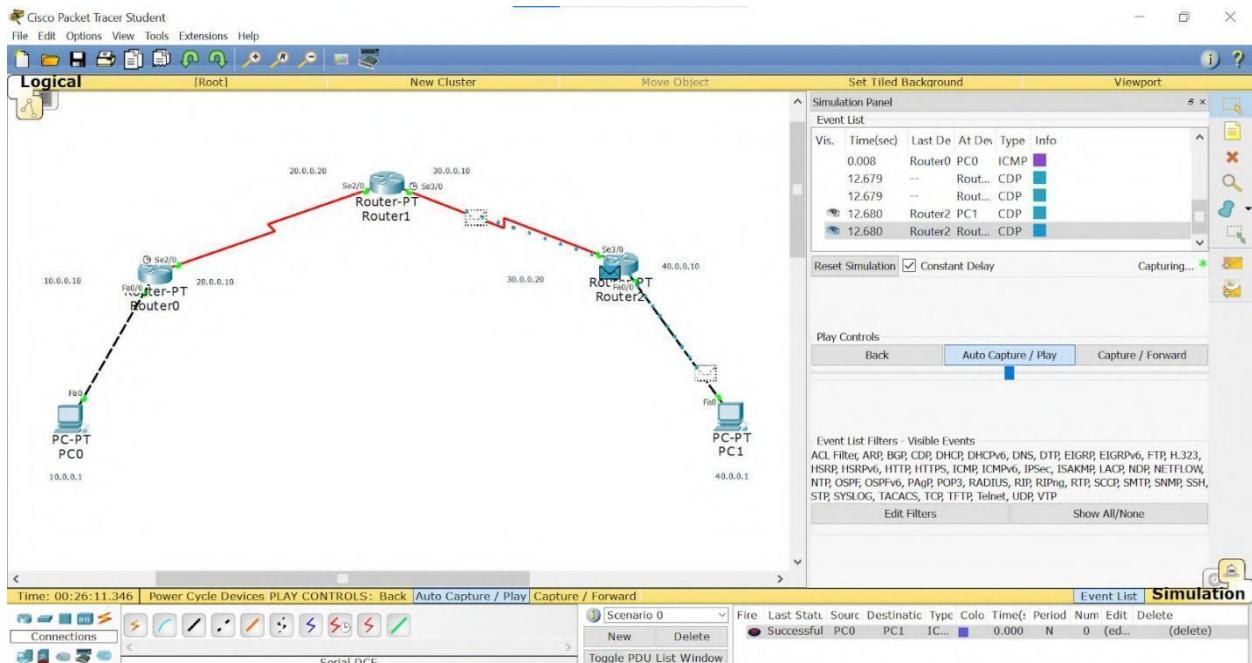
S* 0.0.0.0/0 [1/0] via 30.0.0.1

history 92H1 about www.kitp

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:

LAB-5.

DHCP within LAN and outside LAN

Aim: To configure DHCP within LAN

Topology: Switch - PT - Switch - O

PC-PT
PC0 PC-PT
PC1 PC-PT
PC2
10.0.0.2 10.0.0.3 10.0.0.4

Procedure:

- (1) Drag and drop 3 end device and switch and 1 server to the workspace and connect them as shown in the figure.
- (2) Set the ip address of the server as 10.0.0.1 and in service tab, turn on the DHCP and create a server pool with start address as 10.0.0.2 and save.
- (3) In all the PC's go to desktop → IP Configuration and turn on DHCP ip address will be assigned as 10.0.0.2, 10.0.0.3
These ip addresses are provided by the server through DHCP protocol

Observation

As soon as server is turned on, the IP address 10.0.0.1 is assigned to PC0. The other two PCs receive IP addresses 10.0.0.2 and 10.0.0.3 respectively.

Aim:

10.0.0.1
10.0.0.2
10.0.0.3
PC0

Procedure :

- (1) Drag and drop the end devices
Connect them as shown in the figure
Click on the router - CLI
 → enable
 → configure terminal
 → interface Fa0/0
 → ip address 10.0.0.4 255.0.0.0
 → no shutdown
 → exit
 → interface Fa1/0
 → ip address 20.0.0.1 255.0.0.0
 → no shutdown
 → exit
 → interface Fa1/0
 → ip helper address 10.0.0.1
 → exit

(2) Click on the server → services → DHCP
server pool → Gateway 10.0.0.4

Starting address 10.0.0.2

Maximum no of device = 10

Click on save

Server pool new → gateway 20.0.0.1

Starting address 20.0.0.2

Maximum no of devices = 100

Click on add.

(3) For each PC set fast ethernet
O/I to DHCP

(4) All PC's will have their IP
address automatically allotted

(5) Ping PC's to test connectivity

Observation

For a server
allotted
A gateway
for server
the server
and
Once the
server
be connected

Result:

Ping
Reply
Reply
Reply
Reply
Reply

Set
Set
21/1/23

N

10.0.0.1
Server PT
Server O

Observation

As soon as the server is assigned a server pool with the appropriate starting IP address all devices within that server pool will have an IP address starting from that starting address.

Result:- 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 = 125

Reply from 10.0.0.3 bytes = 32 time = 0ms TTL = 125

Reply from 10.0.0.3 bytes = 32 time = 0ms TTL = 125

Reply from 10.0.0.3 bytes = 32 time = 0ms TTL = 125

Reply from 10.0.0.3 bytes = 32 time = 0ms TTL = 125

Ping Statistics for 10.0.0.3

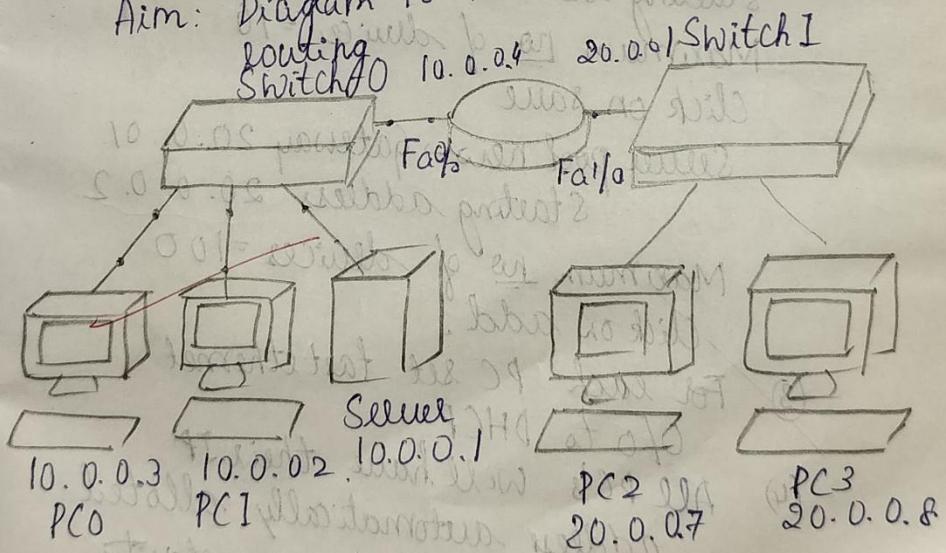
Packets sent = 4 Received = 4 lost = 0

(approximate) (loss)

Approximate = 0ms Maximum = 0ms

Average = 0ms

Aim: Diagram for internal LAN DHCP



d
otocol

Observation

For a server, IP address must be allotted statically.

A gateway is assigned to the server for server pool now, the PC's outside the server LAN will have 2 random address 169.254.0.1

Once the gateway is assigned to the server the PCs will automatically be assigned the gateway.

Result:

Ping 20.0.0.7 (from 10.0.0.2)

Reply from 20.0.0.7 byte = 32 time = 1ms TTL = 127

Reply from 20.0.0.7 byte = 32 time = 0ms TTL = 127

Reply from 20.0.0.7 byte = 32 time = 1ms TTL = 127

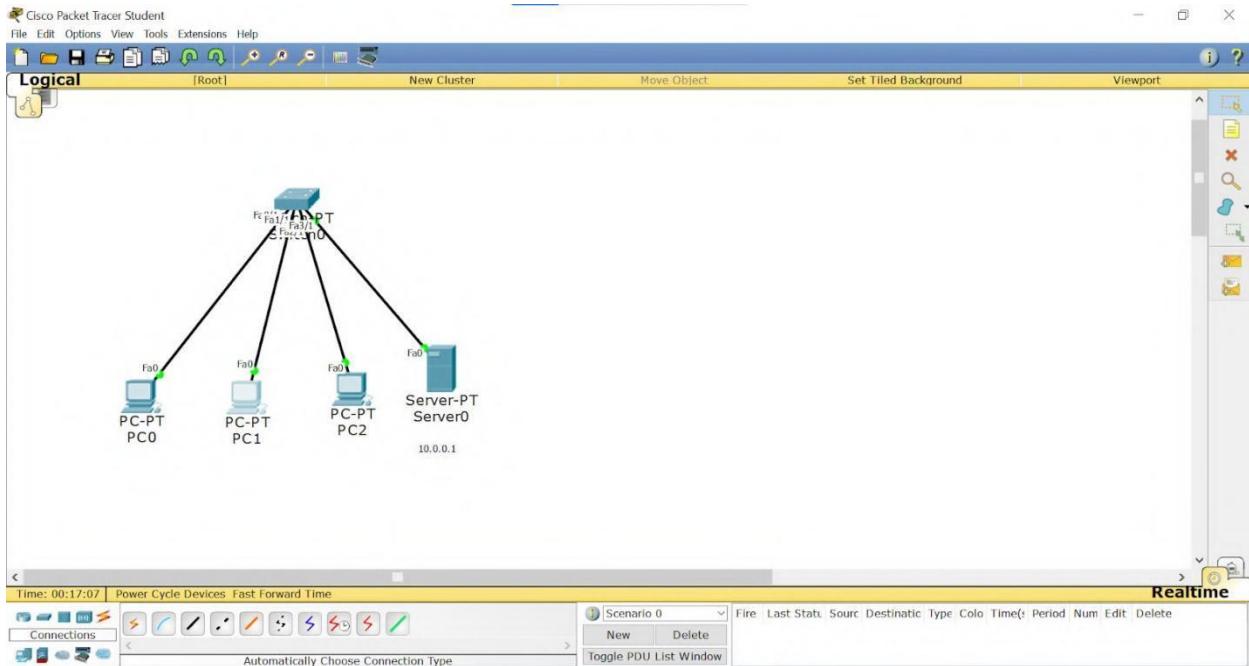
Reply from 20.0.0.7 byte = 32 time = 0ms TTL = 127

Reply from 20.0.0.7 byte = 32 time = 0ms TTL = 127

ST 0.0.0.1
2/11/23

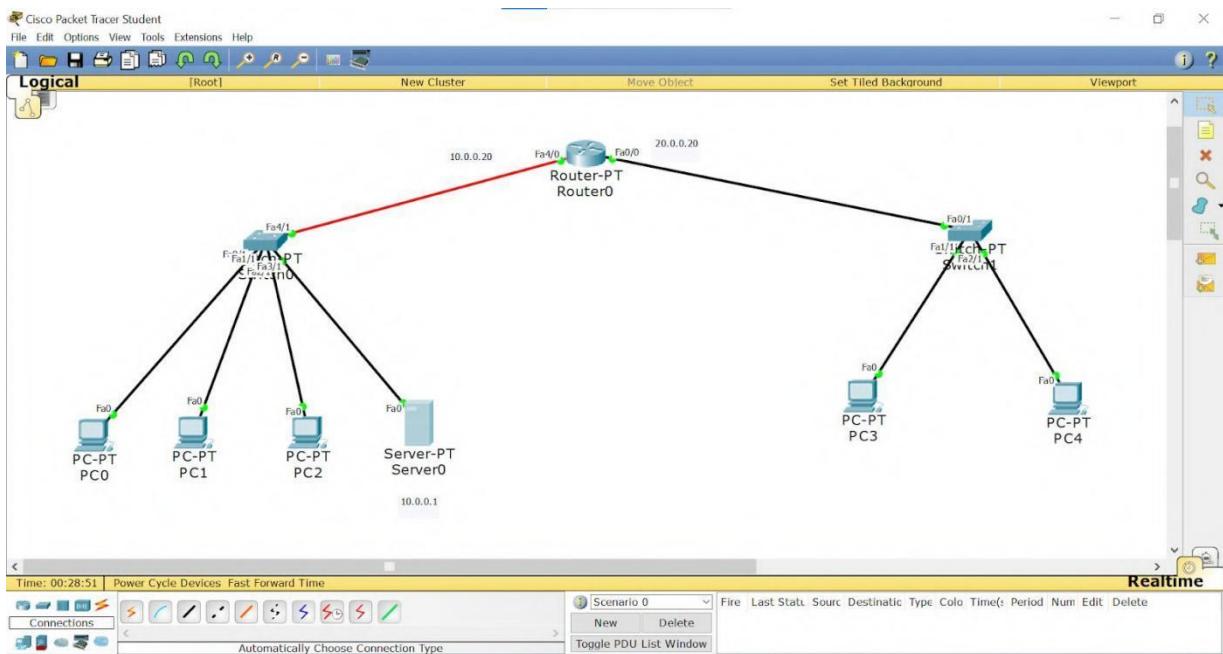
TOPOLOGY:

PROGRAM



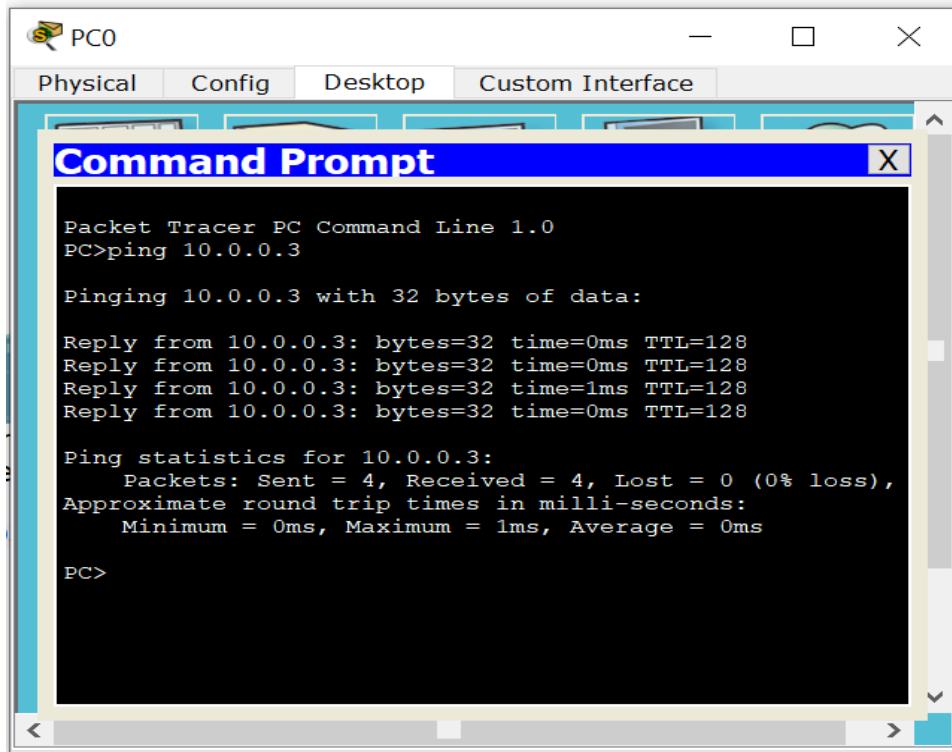
4.1:

PROGRAM 4.2:



OUTPUT:

PROGRAM



```
PC0 Physical Config Desktop Custom Interface
Command Prompt X
Packet Tracer PC Command Line 1.0
PC>ping 10.0.0.3

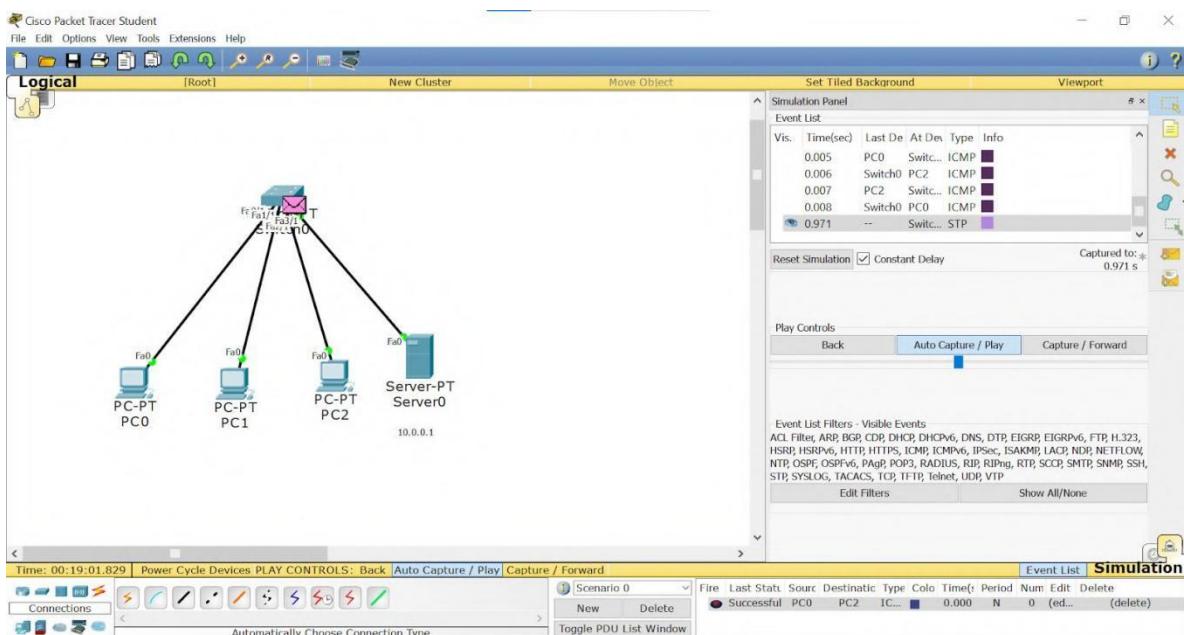
Pinging 10.0.0.3 with 32 bytes of data:

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

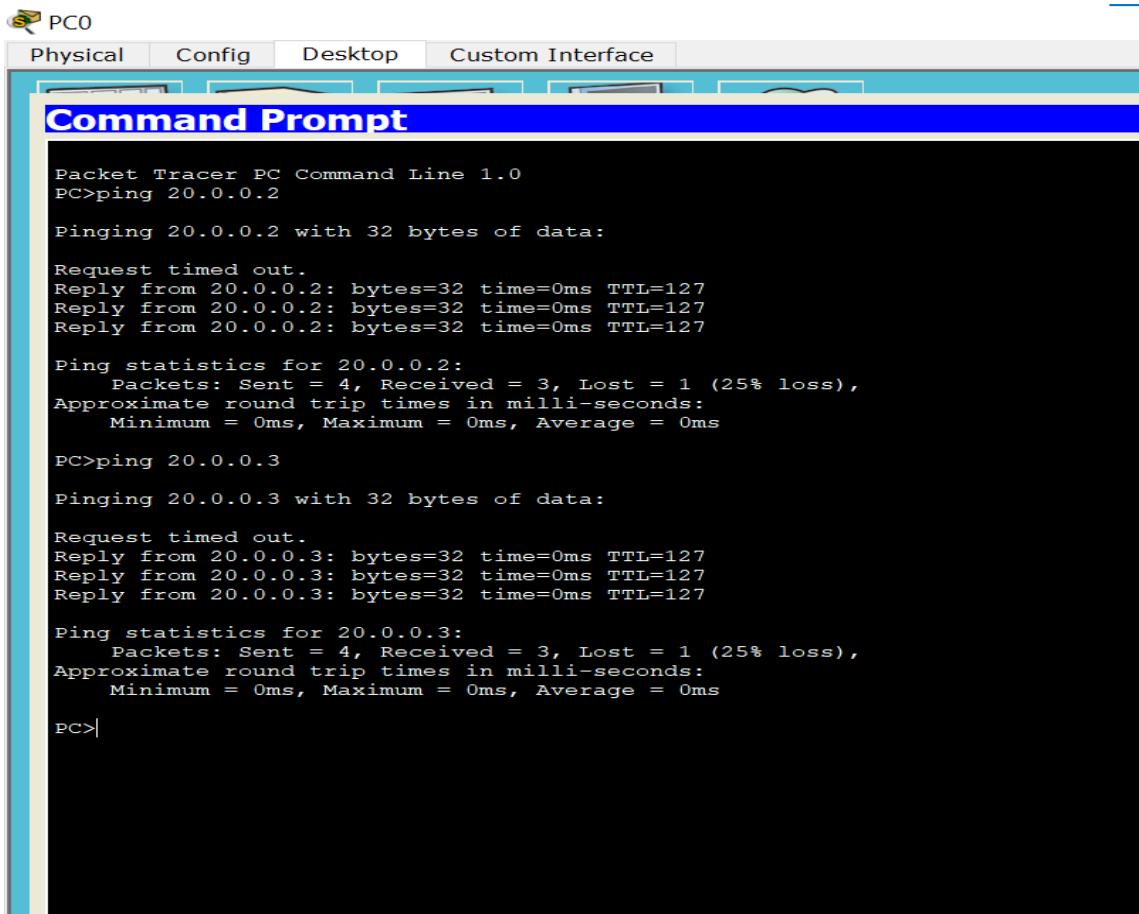
Ping statistics for 10.0.0.3:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 0ms, Maximum = 1ms, Average = 0ms

PC>
```

4.1:



PROGRAM 4.2:



The screenshot shows a Cisco Packet Tracer interface titled "PC0". The top menu bar includes "Physical", "Config", "Desktop", and "Custom Interface". A toolbar below the menu contains icons for various functions. The main window is titled "Command Prompt" in blue. The terminal window displays the following command-line session:

```
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

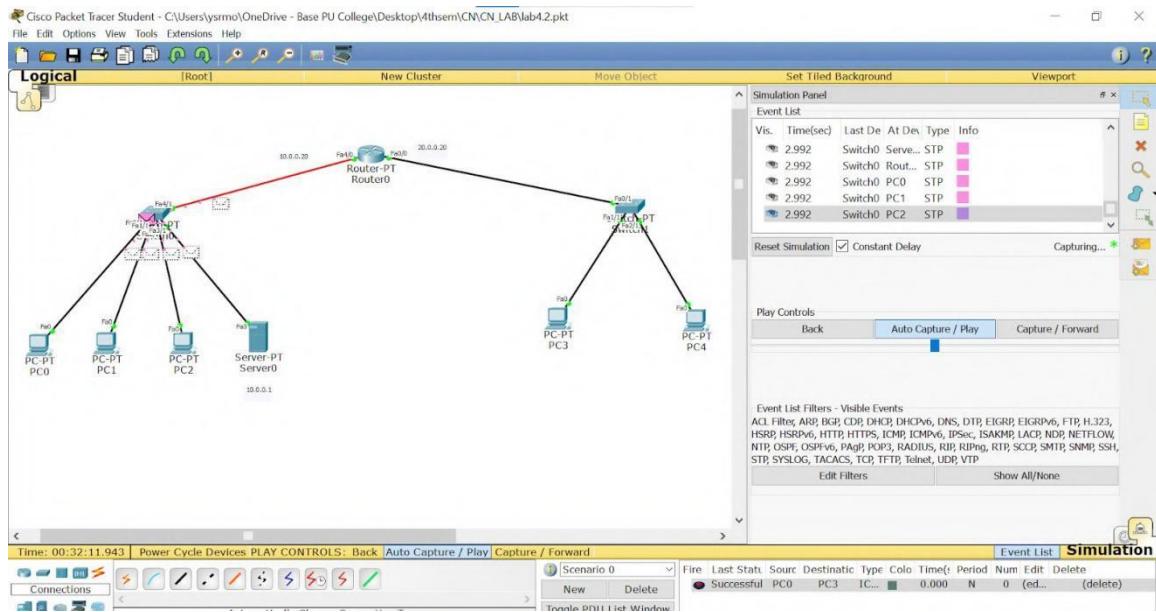
PC>ping 20.0.0.3

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:

APM :- Config Web Server, DNS within LAN

Procedure :

- (1) Create a topology by placing a PC, a server and a switch on the workspace
- (2) Configure PC and server (IP address + gateway)
- (3) Open web browser of PC to set IP address of server
- (4) Configure DNS of server with Name (server name) and URL (ip address)
- (5) Edit index.html to display USN and Name

Topology :

```
graph LR; PC[PC-PT] --- SW[Switch-PT]; SW --- Server[Server 0<br/>10.0.0.2]
```

Output

Output :-

Web Browser
URL http://hello123.com

USN: IBM21CS265
Name: Peithuvi

Route Interface

Topology : 20
10.0.0.2 Router

PC-PT
PC.O
10.0.0.1

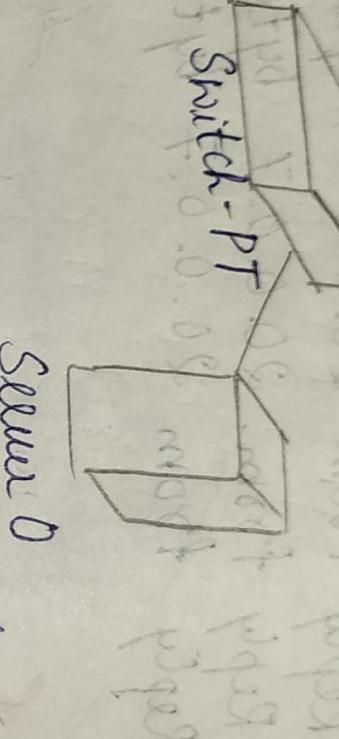
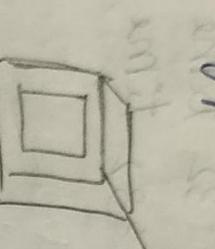
Procedure :

- Create a
- Configure gateway
- Configure

for
ena
conf
etc
IP
in
encap
serial
64000
pers
0-64000
unit
unit
unit
unit

Topology :

Instant
 $K_3 = \frac{4+1}{2} = 3$



Server 0
10.0.0.2.0

Output
PC-PT
PC8a 01

Output :-

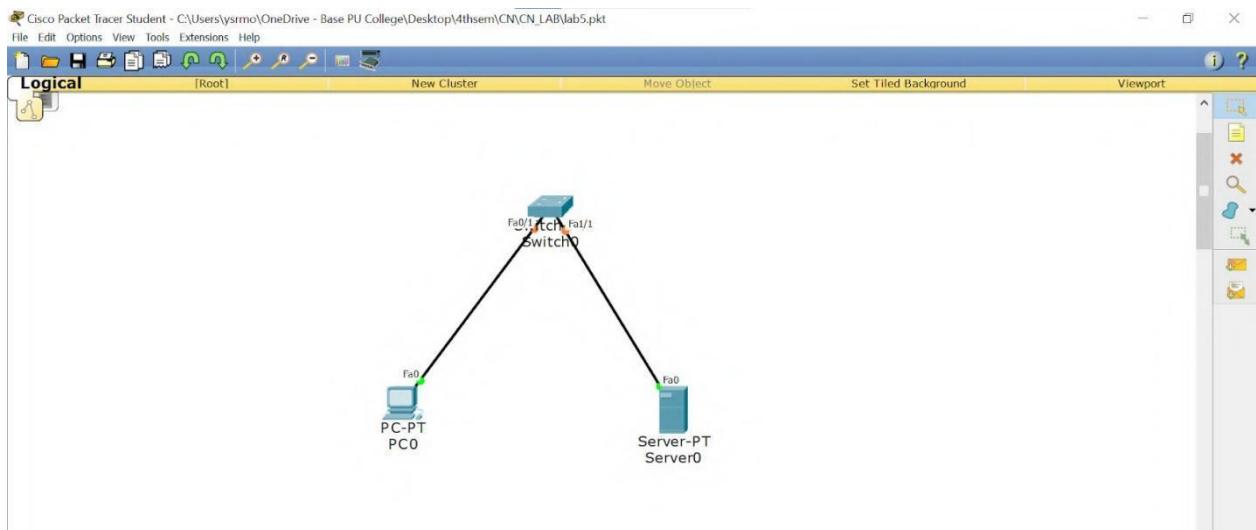
Web Browser

URL $http://\text{hello123.169}$

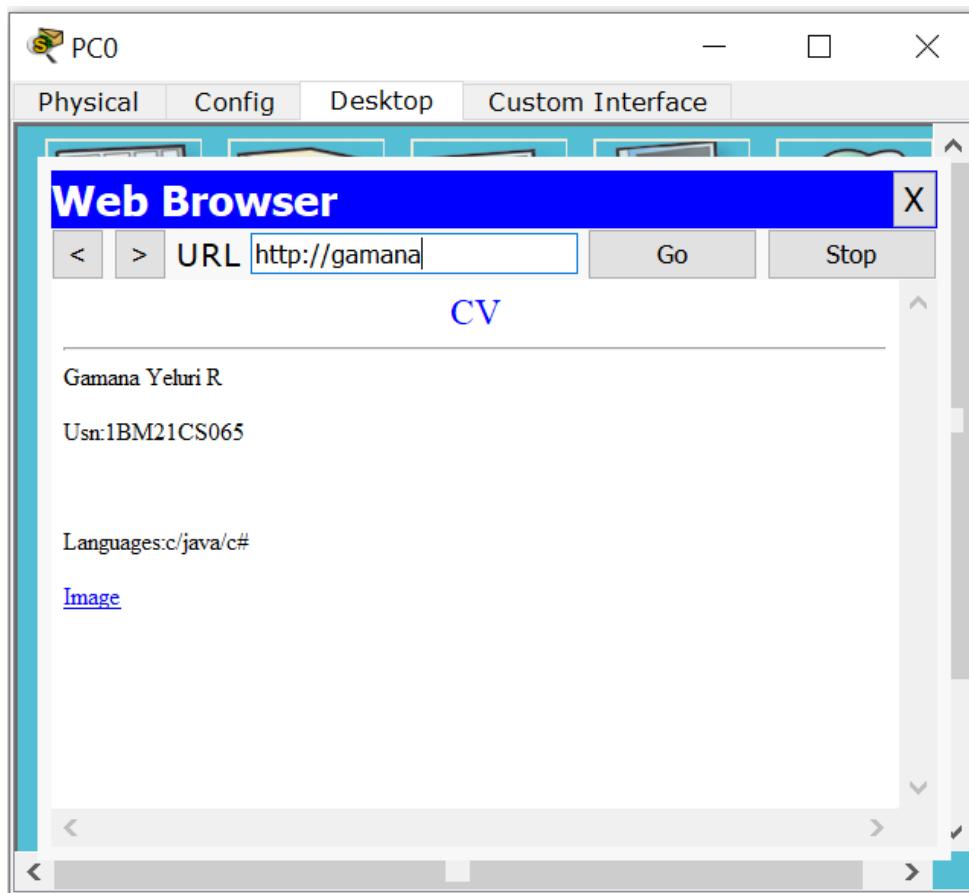
[60] Stop

OSN: IBM21CS265
Name: Prithvi

TOPOLOGY:



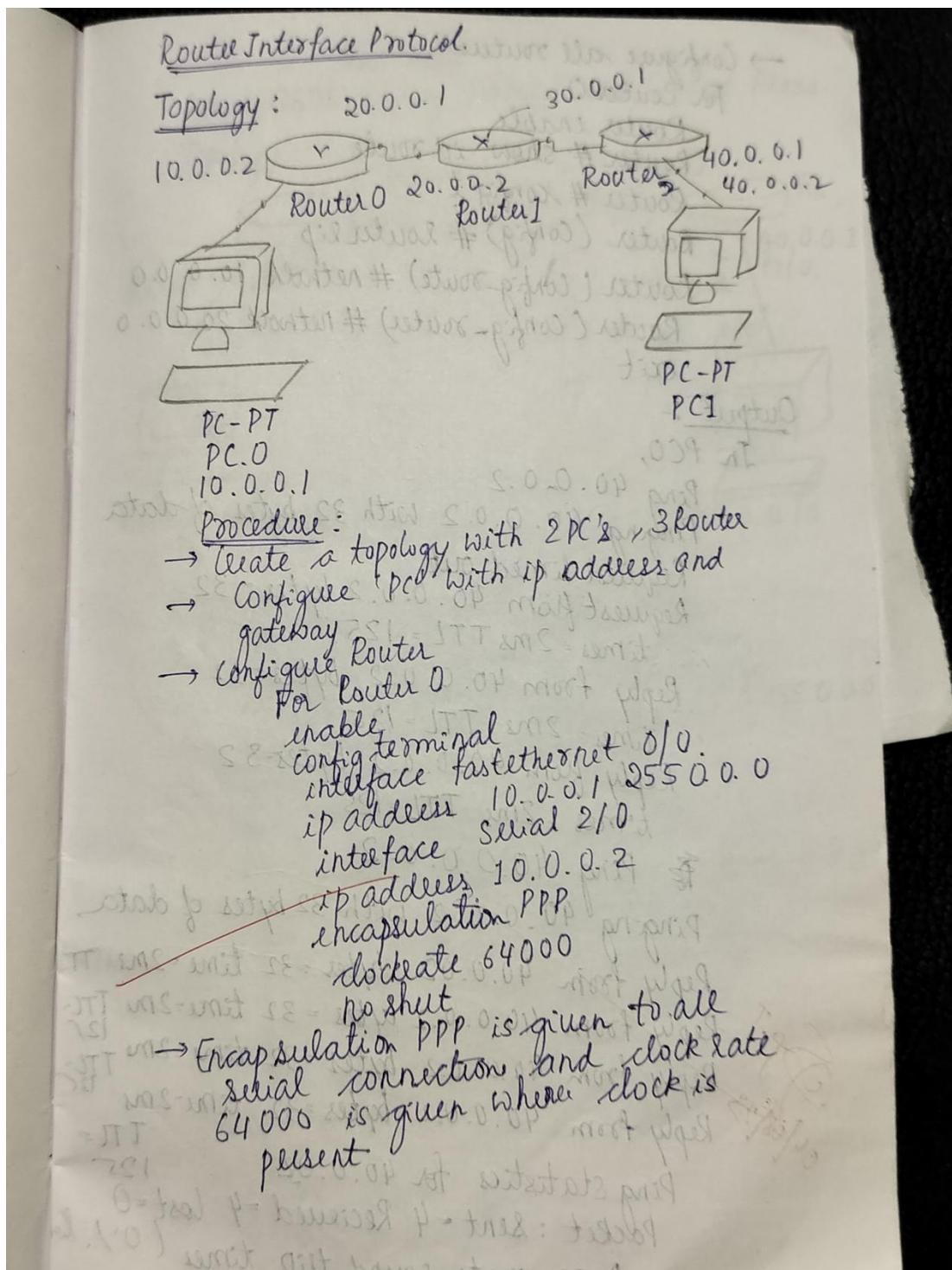
OUTPUT:



WEEK 6

Configure RIP routing Protocol in Routers.

OBSERVATION:



→ Configure all routers

For Router 0

Router enable

Router # show ip route

Router # config t

Router (config) # router rip

Router (config-route) # network 10.0.0.0

Router (config-route) # network 20.0.0.0

exit

Output :-

In PC0,

Ping 40.0.0.2

Pinging 40.0.0.2 with 32 bytes of data

Request timed out

Request from 40.0.0.2 bytes = 32

times = 2ms TTL = 125

Reply from 40.0.0.2 bytes = 32

times = 2ms TTL = 125

Reply from 40.0.0.2 bytes = 32

times = 2ms TTL = 125

Ping 40.0.0.2

Pinging 40.0.0.2 with 32 bytes of data

Reply from 40.0.0.2 bytes = 32 time = 2ms TTL = 125

Reply from 40.0.0.2 bytes = 32 time = 2ms TTL = 125

Reply from 40.0.0.2 bytes = 32 time = 2ms TTL = 125

Reply from 40.0.0.2 bytes = 32 time = 2ms TTL = 125

Ping statistics for 40.0.0.2

Packet : sent = 4 Received = 4 lost = 0 (0% loss)

Approximate round trip time

in milli-seconds

minimum = 2ms, Maximum = 13ms

Aim : C

Topology



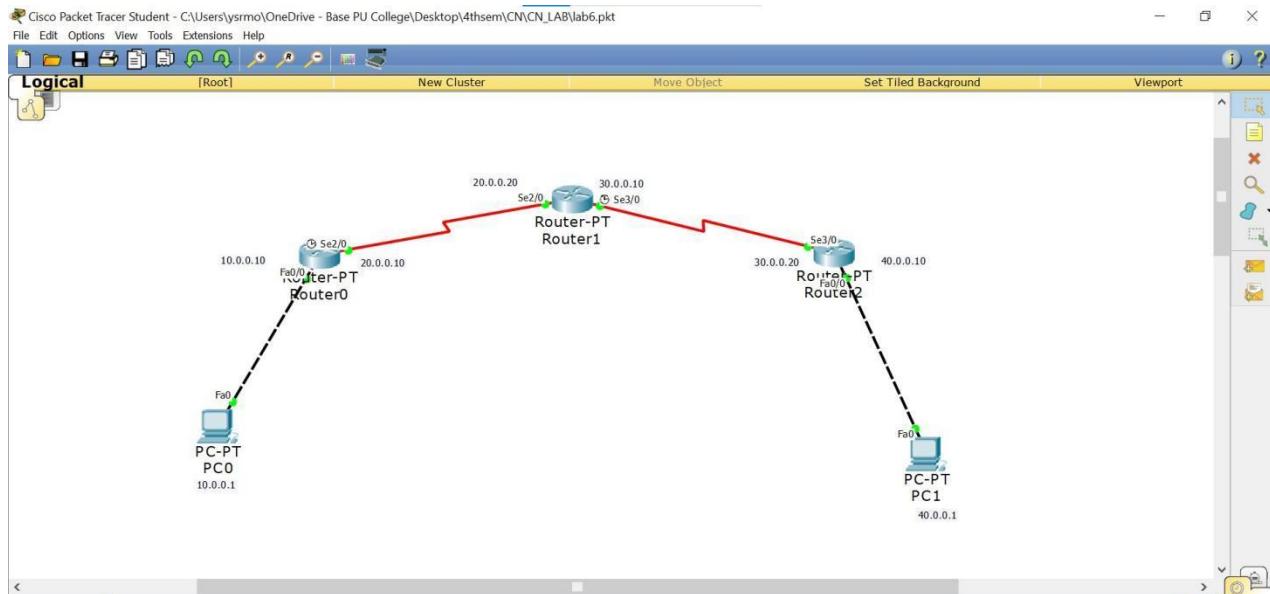
Procedure

(1) Clear

(2) Con

R

TOPOLOGY:



OUTPUT:

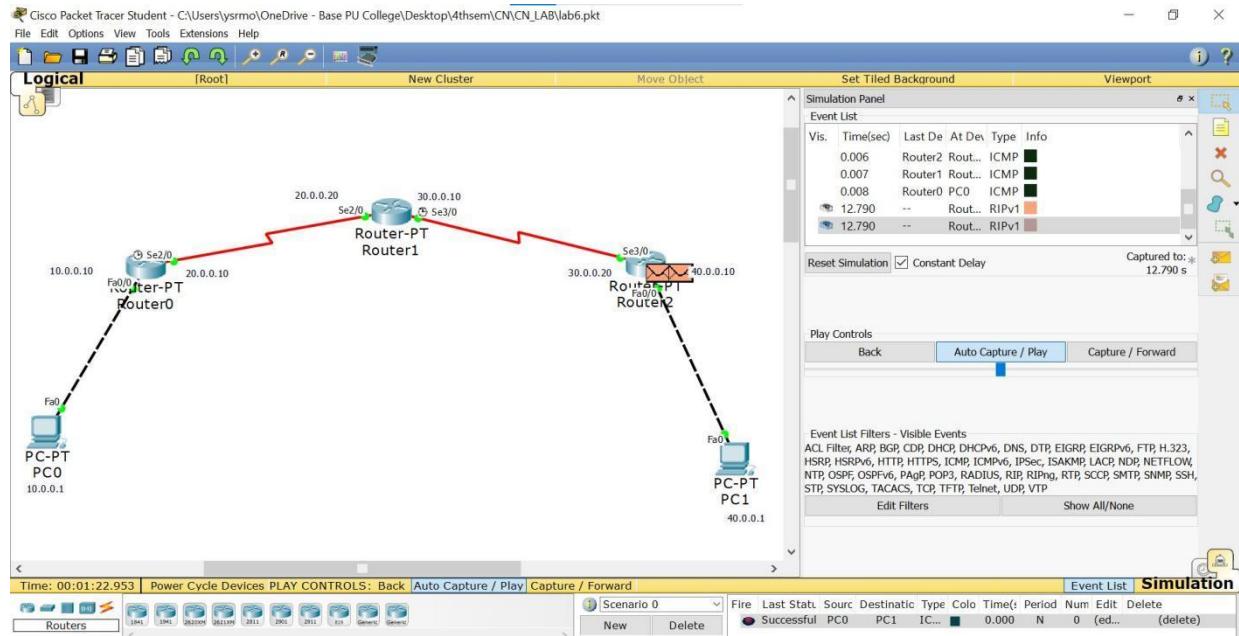
```
PC>ping 40.0.0.1

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

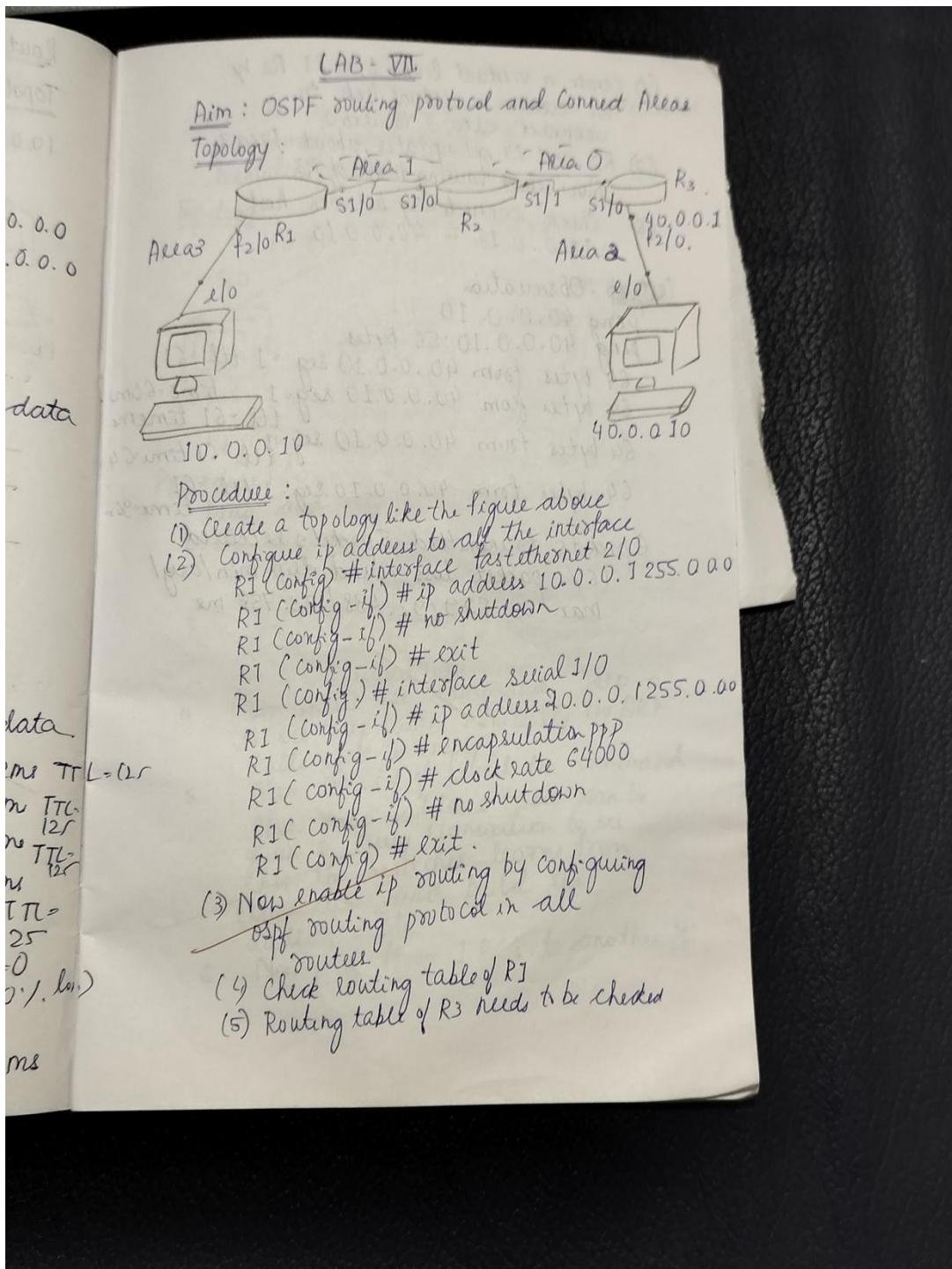
PC>
```



WEEK 7

Configure OSPF routing protocol.

OBSERVATION:



- (6) Create a virtual link b/w RI, R2 by
we create a virtual link to
connect area 3 to area 0.
- (7) R2 and R3 get update about Area 3,
Now check routing table of R3
- (8) Check connectivity between host
10.0.0.10 to 40.0.0.10

Output : Observation

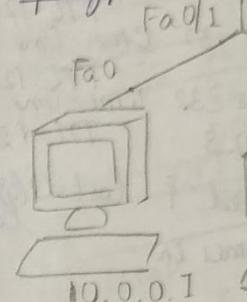
ping 40.0.0.10
 ping 40.0.0.10: 56 bytes
 64 bytes from 40.0.0.10 seg = 1 ttl = 61
 64 bytes from 40.0.0.10 seg = 1 time = 60ms
 64 bytes from 40.0.0.10 seg = 1 ttl = 61 time = 61ms
 64 bytes from 40.0.0.10 seg = 1 ttl = 61 time = 61ms
 64 bytes from 40.0.0.10 seg = 1 ttl = 61 time = 61ms

6 packets transmitted, 5 packets received
 16% packet loss round trip min/avg/
 max 60.829/92.438/173.758 ms

LAB

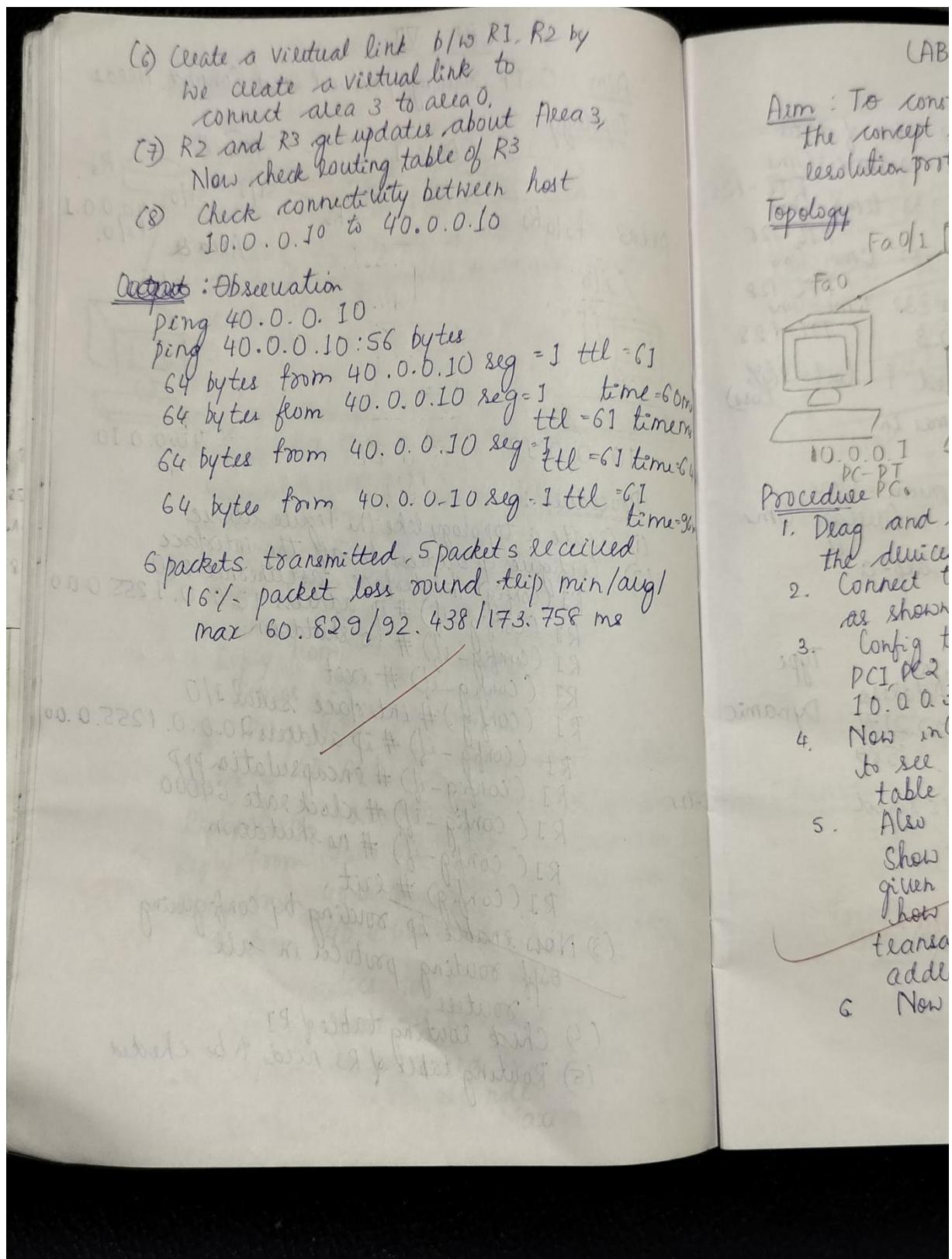
Aim : To cons
the concept
resolution pr

Topology

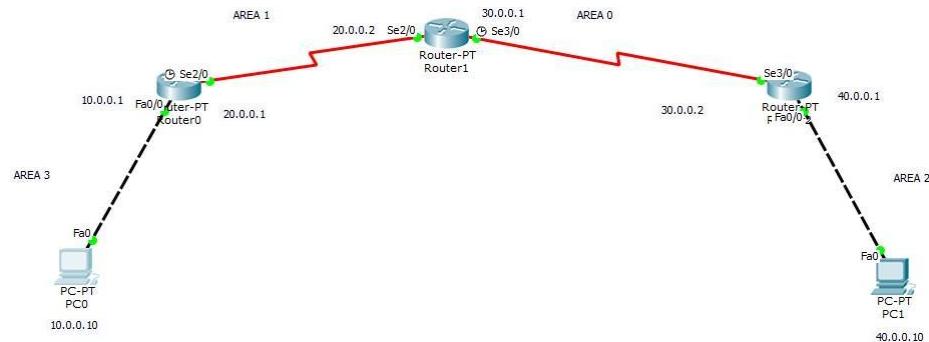


Procedure

1. Drag and
the device
2. Connect t
as shown
3. Config t
PCI, R2
10.0.0.1
4. Now in
to see
table
5. Also
Show
given
how
teamed
addl
6. Now



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),

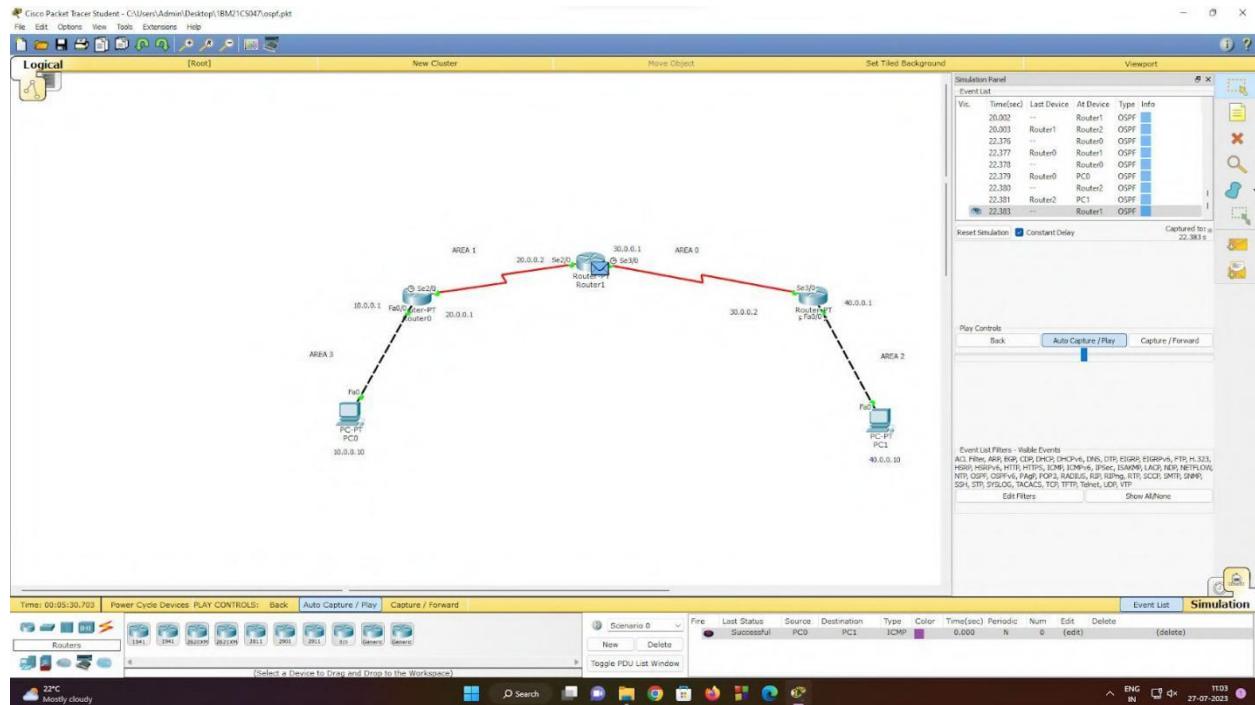
PC>ping 40.0.0.10

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

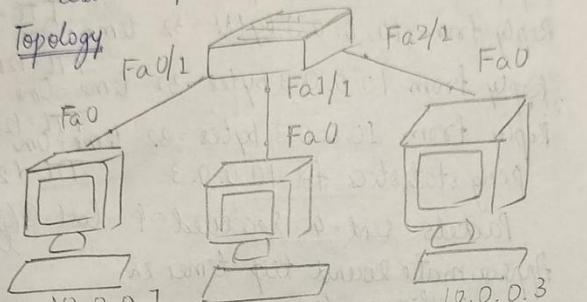
To construct a simple LAN and understand the concept and operation of Address Resolution Protocol (ARP).

OBSERVATION:

LAB - VIII

Aim : To construct simple LAN and understand the concept and operation of address resolution protocol (ARP)

Topology



ttl = 6
time = 60m
= 61 timer
l = 61 time < 4,
tl = 61
time = 96.
ived
in/aug/
me

Procedure

1. Drag and drop 3 PC's & 1 switch from the devices.
2. Connect the devices in the topology as shown above
3. Config the IP address for the PC's PC1, PC2 as, 10.0.0.1, 10.0.0.2, 10.0.0.3 respectively
4. Now in CLI use command "arp -a" to see arp table. Initially the ARP table will be empty.
5. Also in CLI of switch the command "show mac address table" can be given or busy transaction to see how the switch learns from transaction and build the addressee table.
6. Now ping from 1 PC's to another PC.

l	length	19
TTL	TTL	28
OXO	OXO	
CHKSUM	CHKSUM	
10.0.0.1	10.0.0.1	
10.0.0.3	10.0.0.3	

w 10.0.0.3
 10.0.0.3 with 32 bytes of data
 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 = 0ms TTL = 128
 Reply from 10.0.0.3 bytes = 32 time = 0ms TTL = 128
 ping statistics for 10.0.0.3 TTL = 128
 Packets: sent = 4 received = 4 lost = 0% (loss)
 Approximate round trip times in
 milliseconds
 most Minimum = 0ms Maximum = 0ms Average = 0ms

⑦ Again check with the arp-a command

Internet Address	Physical Address	Type
10.0.0.3	0090.217C.1589	Dynamic

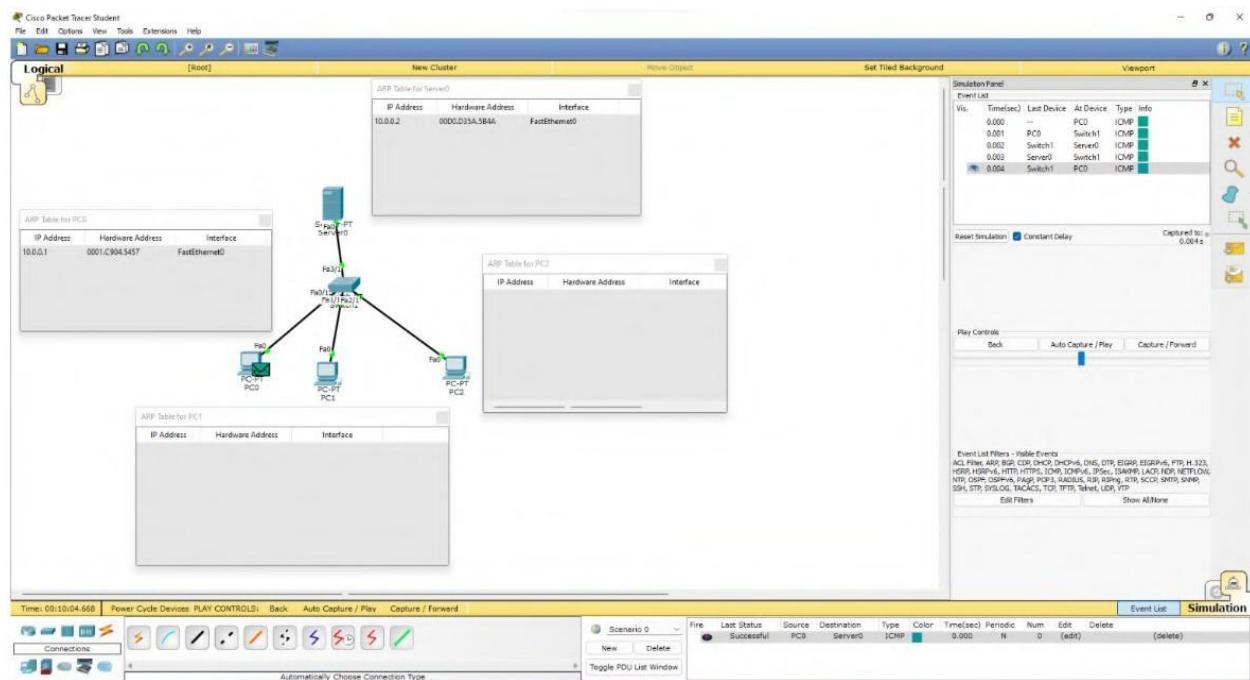
⑧ arp-a command is used to clear the table

S.J. 18/4/23

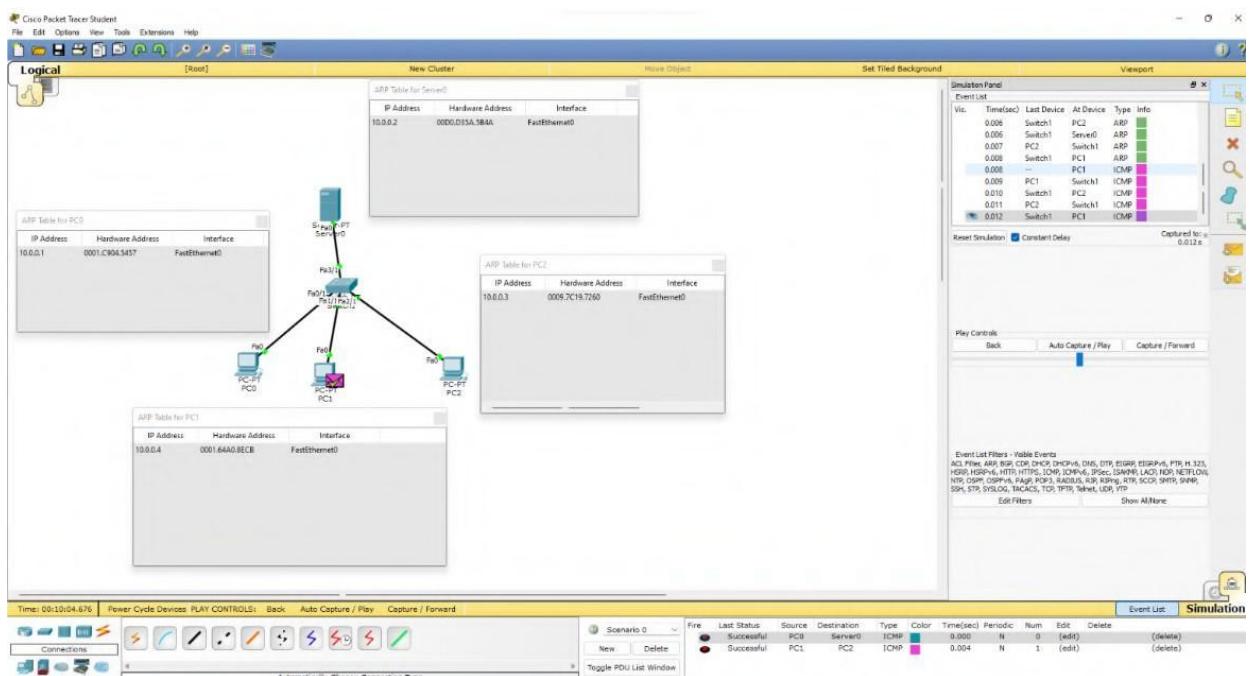
Dem
Aim: Demo
Topology
 ROUTER1

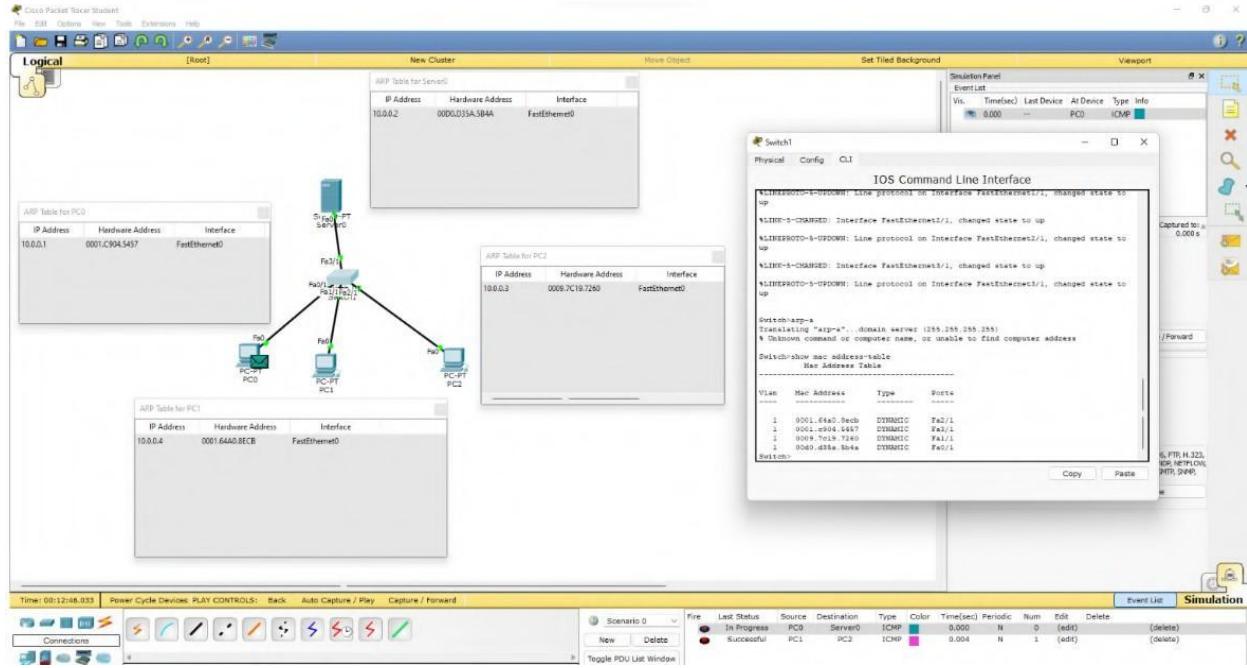
Procedure :
 (1) Con
 defo
 (2) In
 PDU
 (3) Use
 trans
 (4) Circ
 to S
 PDC
 (5) OI
 di
 ac
Observation
 Drf
 et
 Telnet
 Obse

TOPOLOGY:



OUTPUT:

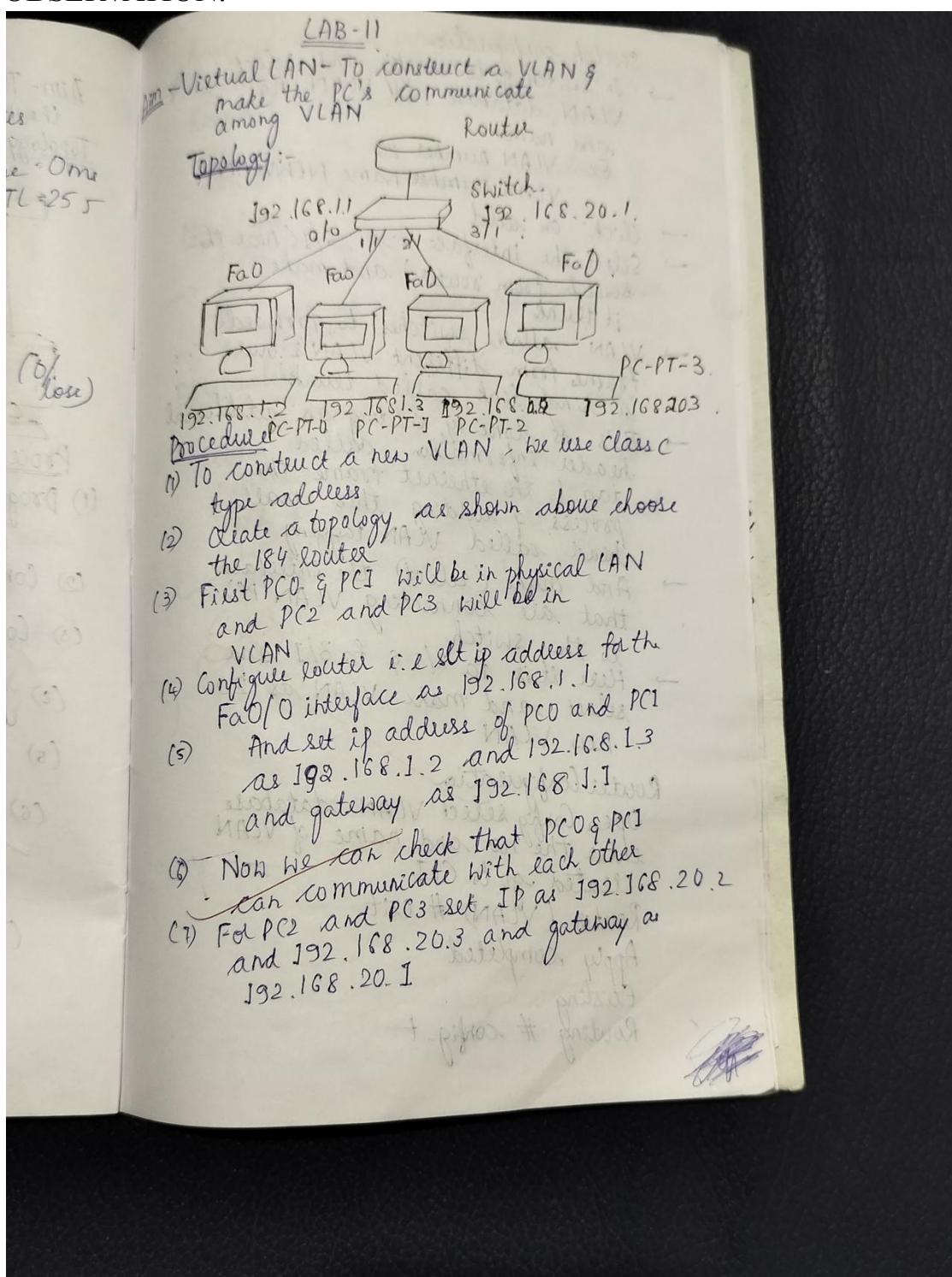




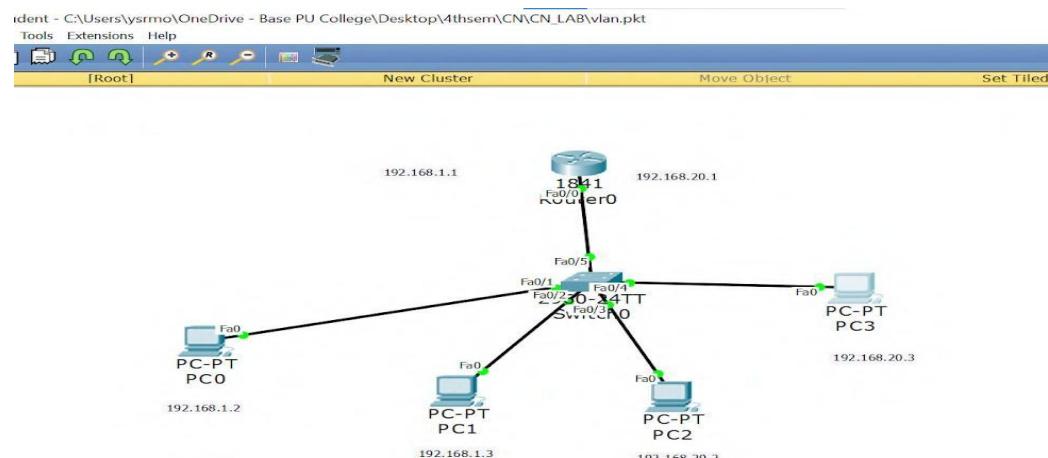
WEEK 9

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

OBSERVATION:



TOPOLOGY:



OUTPUT:

```

Packet Tracer PC Command Line 1.0
PC>ping 192.168.20.3

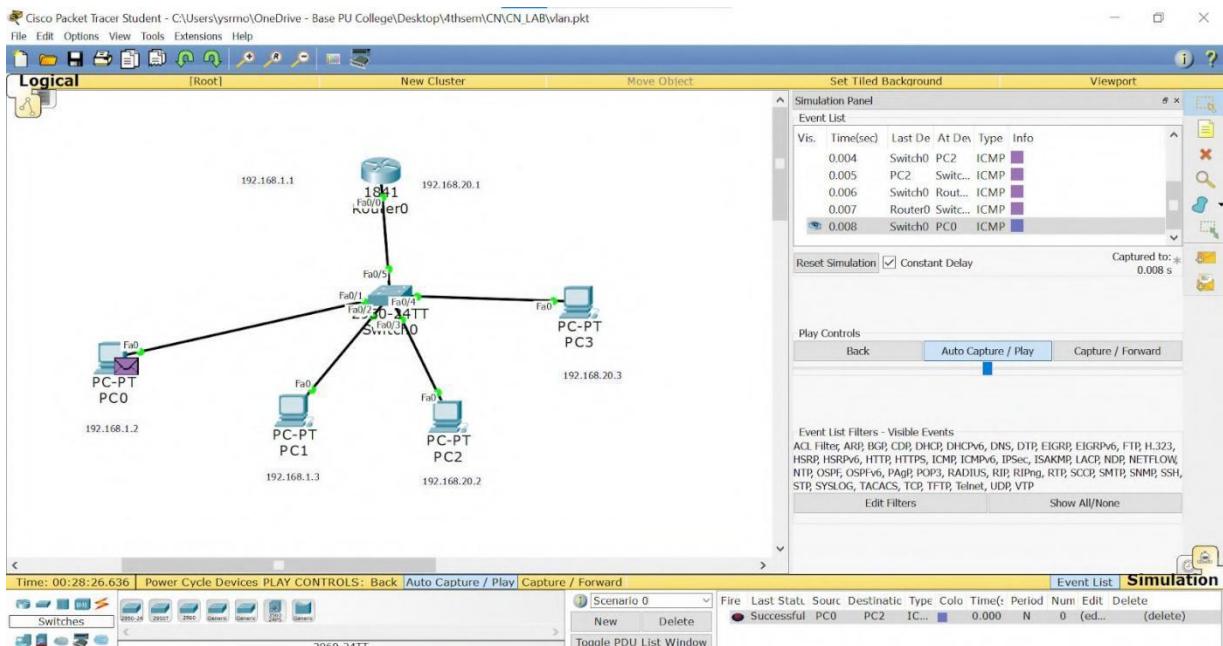
Pinging 192.168.20.3 with 32 bytes of data:

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:

LAB-IX
Demonstrate the TTL/ Life of a Packet

Aim: Demonstrate the TTL implementation

Topology

Procedure : (1) Create a topology as shown above
(2) Configure the device as per static default routing
(3) In the simulation mode send a simple PDU from one PC to another.
(4) Use capture button to capture every transfer.
(5) Click on the PDU during every transfer to SAP the inbound and outbound PDU details.
(6) Observe that there is a difference of 1 in it when it crosses across the router.

Observation and Result

~~Difference of 1 in TTL when the PDU crosses every routee.~~

Telnet Observation (Contd)

53

Observation

PDU information at Device : PCU

OSI model Outbound PDU details

PDU format

Ethernet II

0 4 8 14 19 Bytes

Preamble:	^	DEST ↑	SRC ^
101010...101	↓	MAC ↓	MAC ↗
Type F:	↓	DATA ↗	PCS ↓

0 4 8 16 19 21 31 Bytes
4 TTL DSCP TL: 28
ID: 0x7 OX0 OX0
TTL: 255 PRO: OX1 CHKSUM
SCR: IP: 10.0.0.1 DEST IP: 40.0.0.1
BUPT: OX0 OX0
DATA (Variable length)

ICMP

0 8 16 31 Bits

TYPE	CODE	CHECKSUM
ID: 0x8		SEQNUMBER: 7

PDU Information at Device Router I
OSI Model Inbound PDU details Outbound PDU

PDU Formats

Ethernet II

0 4 8 14 19 Bytes

Preamble:	^	DEST ↑	SRC ^
101010:	↓	MAC ↓	MAC ↗
TYPE:	↓	DATA (Variable length)	FCFS: ↗

Aim:

Topolog

ROUTER1



Procedure

(2)

(3)

(4)

(5)

(6)

Observation

Device

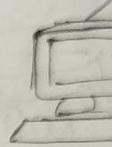
Tel 0

WNS OT SL 6
 10.0.0.3
 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 = 0ms TTL = 128
 Reply from 10.0.0.3 bytes = 32 time = 0ms TTL = 128
 Reply from 10.0.0.3 bytes = 32 time = 0ms TTL = 128
 Ping statistics for 10.0.0.3 TTL = 128
 Packets: sent = 4, received = 4 lost = 0% (loss)
 Approximate round trip times in
 milliseconds
 most Minimum = 0ms Maximum = 0ms
 Average = 0ms

⑦ Again check with the arp-a command
 PC > arp-a
 Internet Address Physical Type
 Address 10.0.0.3 0.090.217C. Dynamic
 1589

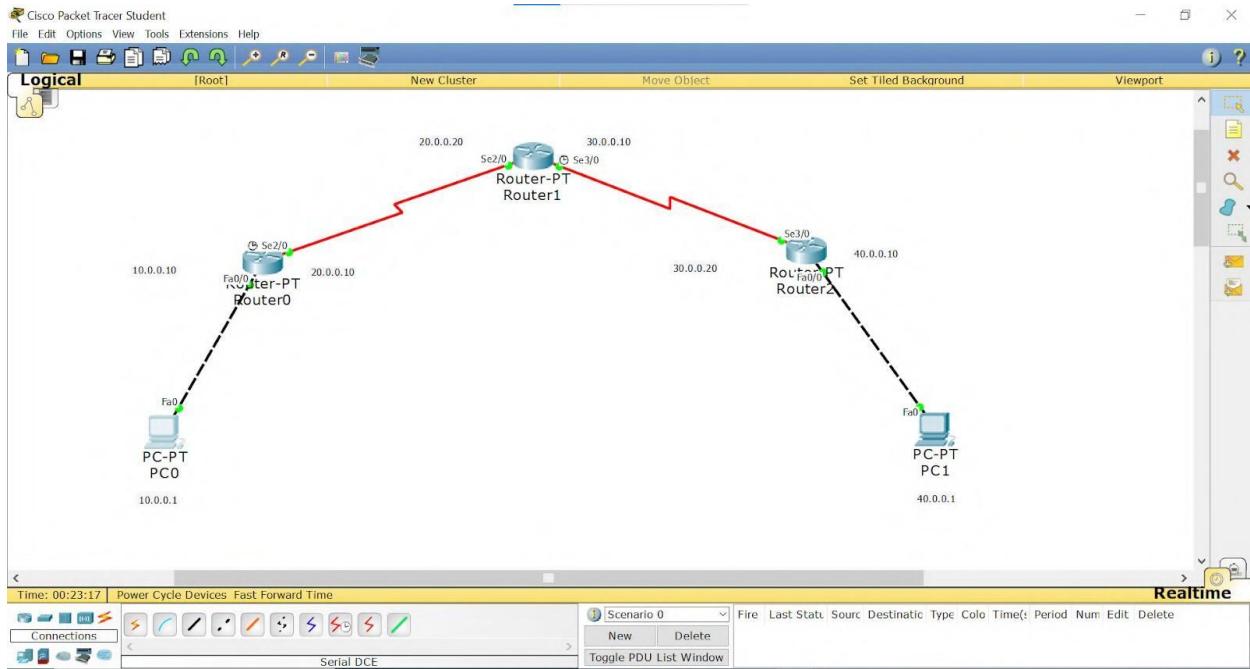
⑧ arp-a command is used
 to clear the table

SFT 18423

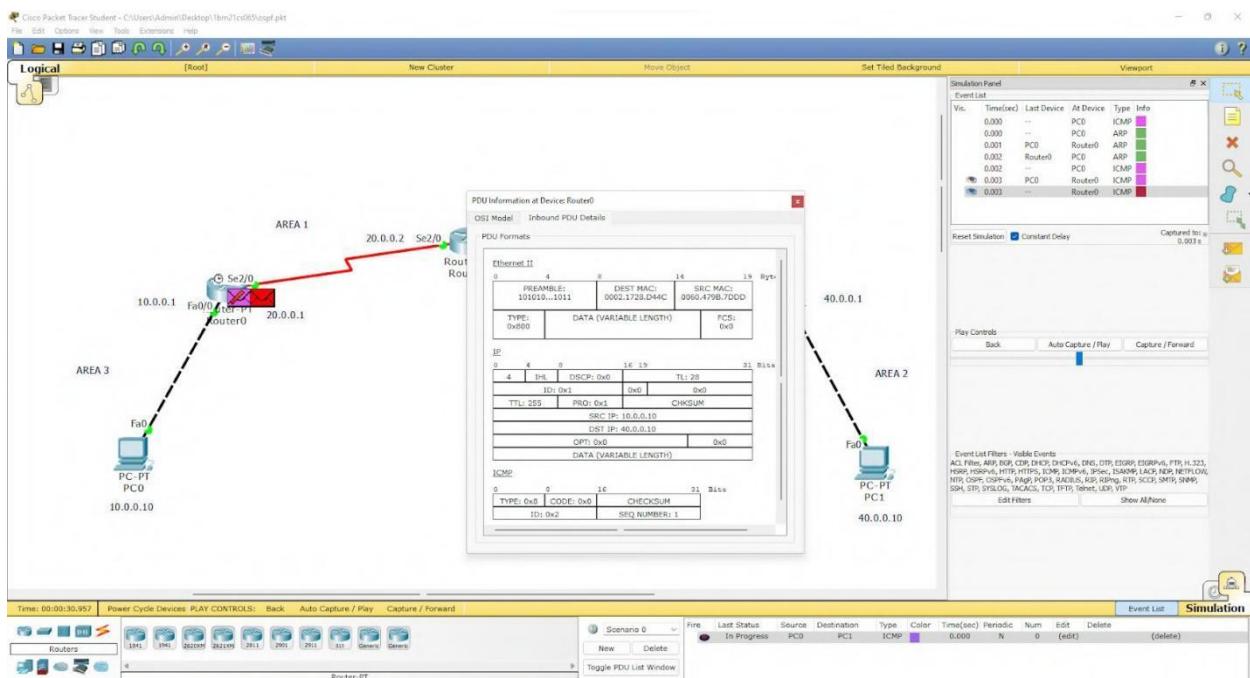
- Aim:
Topolog
ROUTER1

Procedure
 (2)
 (3)
 (4)
 (5)
 (6)

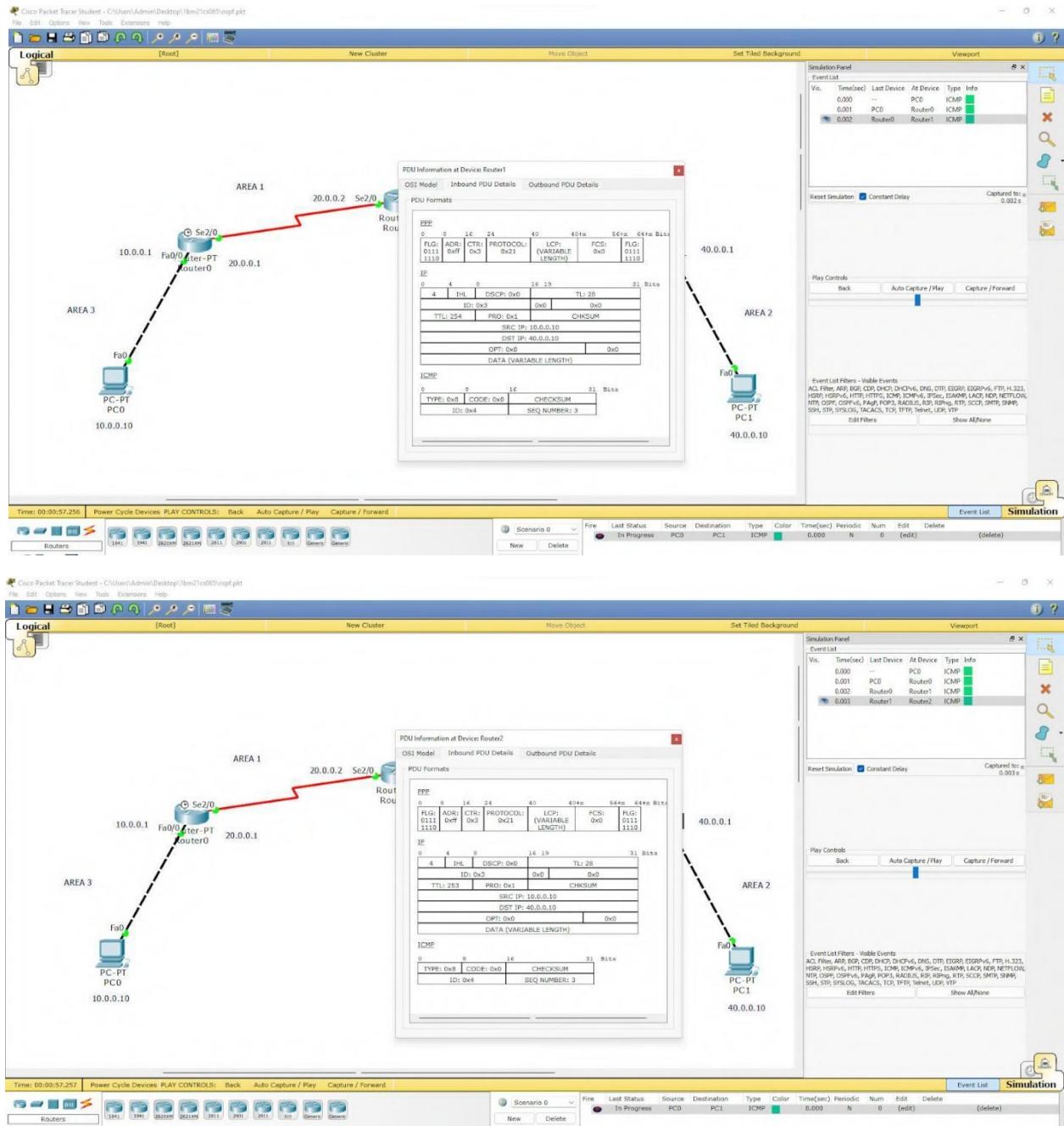
Observation
Te

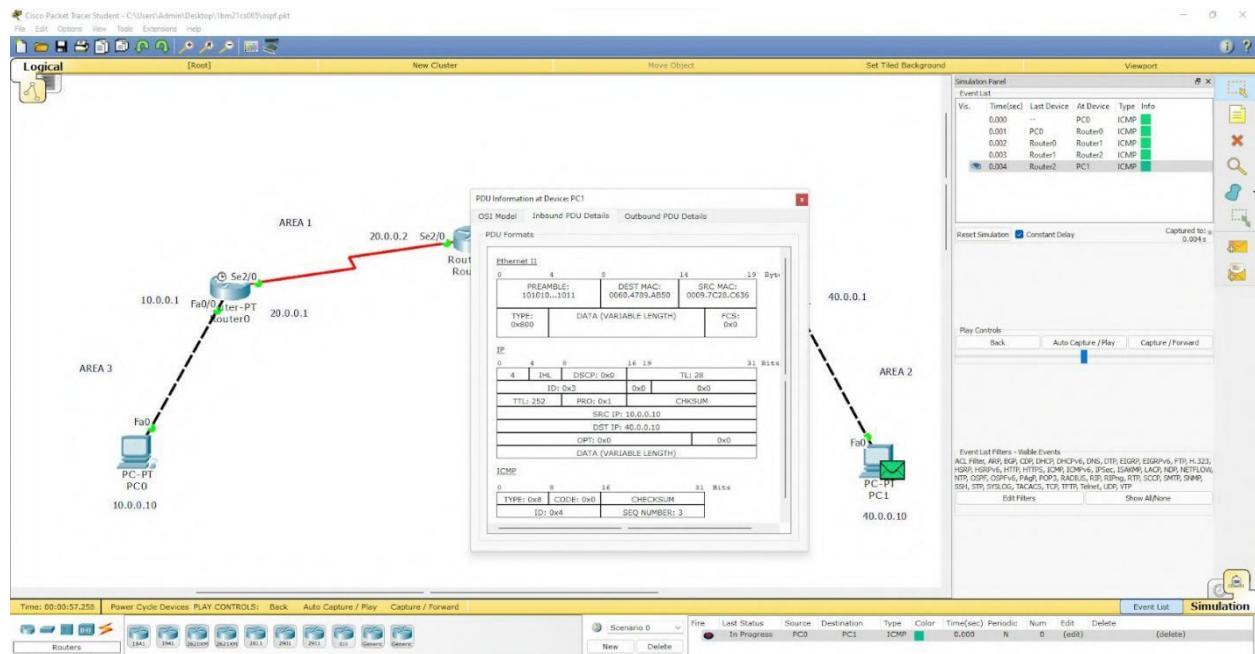
TOPOLOGY:



OUTPUT:



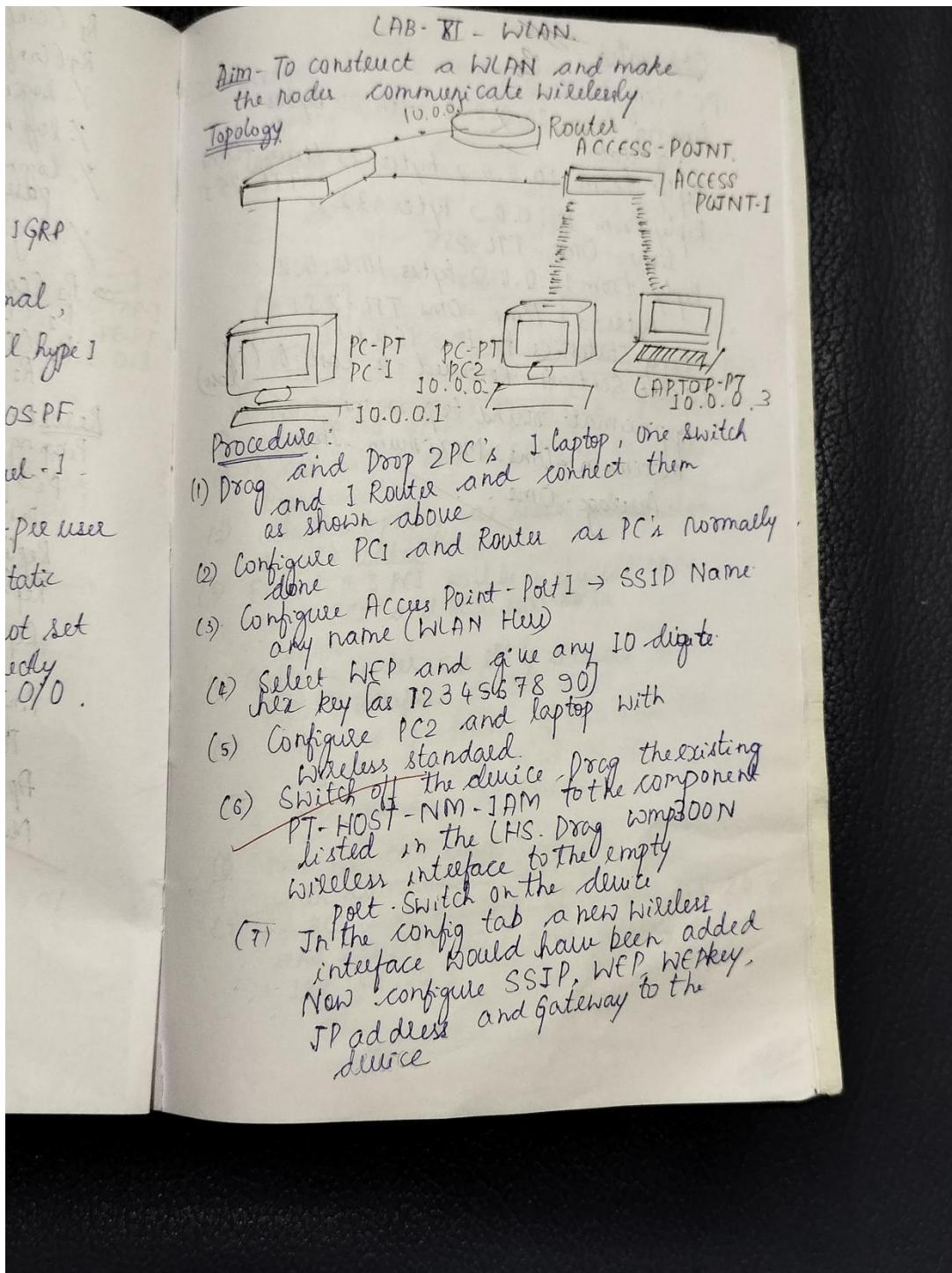




WEEK 11

To construct a WLAN and make the nodes communicate wirelessly

OBSERVATION:



Observation

PC > Ping 10.0.0.2

Pinging from 10.0.0.2 with 32 bytes
of data

Reply from 10.0.0.2 bytes = 32 time = 0ms

Reply from 10.0.0.2 bytes = 32 TTL = 255
time = 0ms TTL = 255

Reply from 10.0.0.2 bytes 10.0.0.2

bytes = 32 time = 0ms TTL = 255

Ping statistics for 192.168.1.1

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

Approximate round trip in ms

minimum = 0ms Maximum = 1ms

Average = 0ms.

Aim - Virtual LAN
make the
among

Topology :-

192.1



192.168.1.2

Procedure :-

(1) To consta

type a

(2) Create a

the 184

(3) First PC

and I

VLAN

(4) Configure

Fa0/C

(5) And

as

and

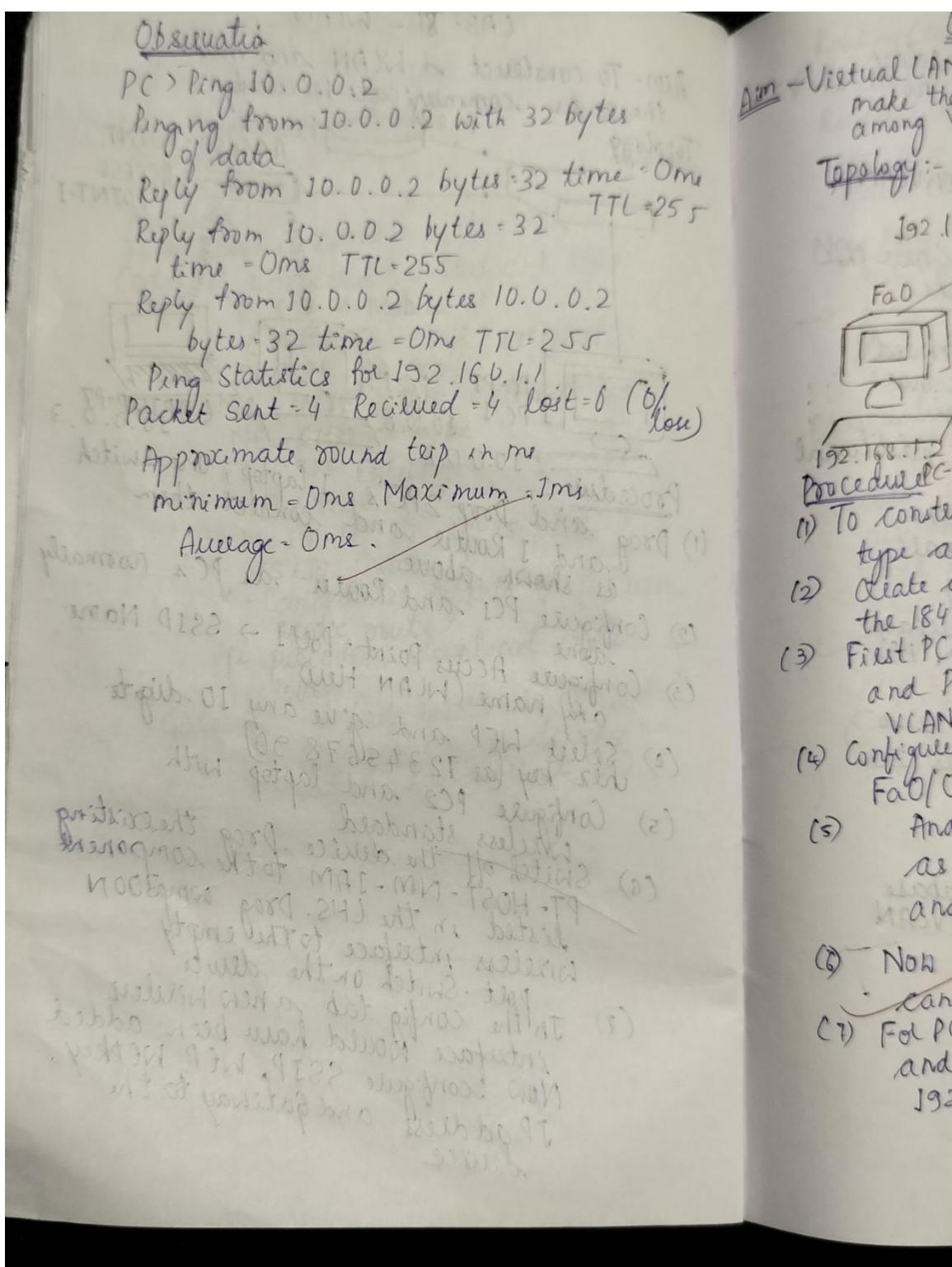
(6) Now

can

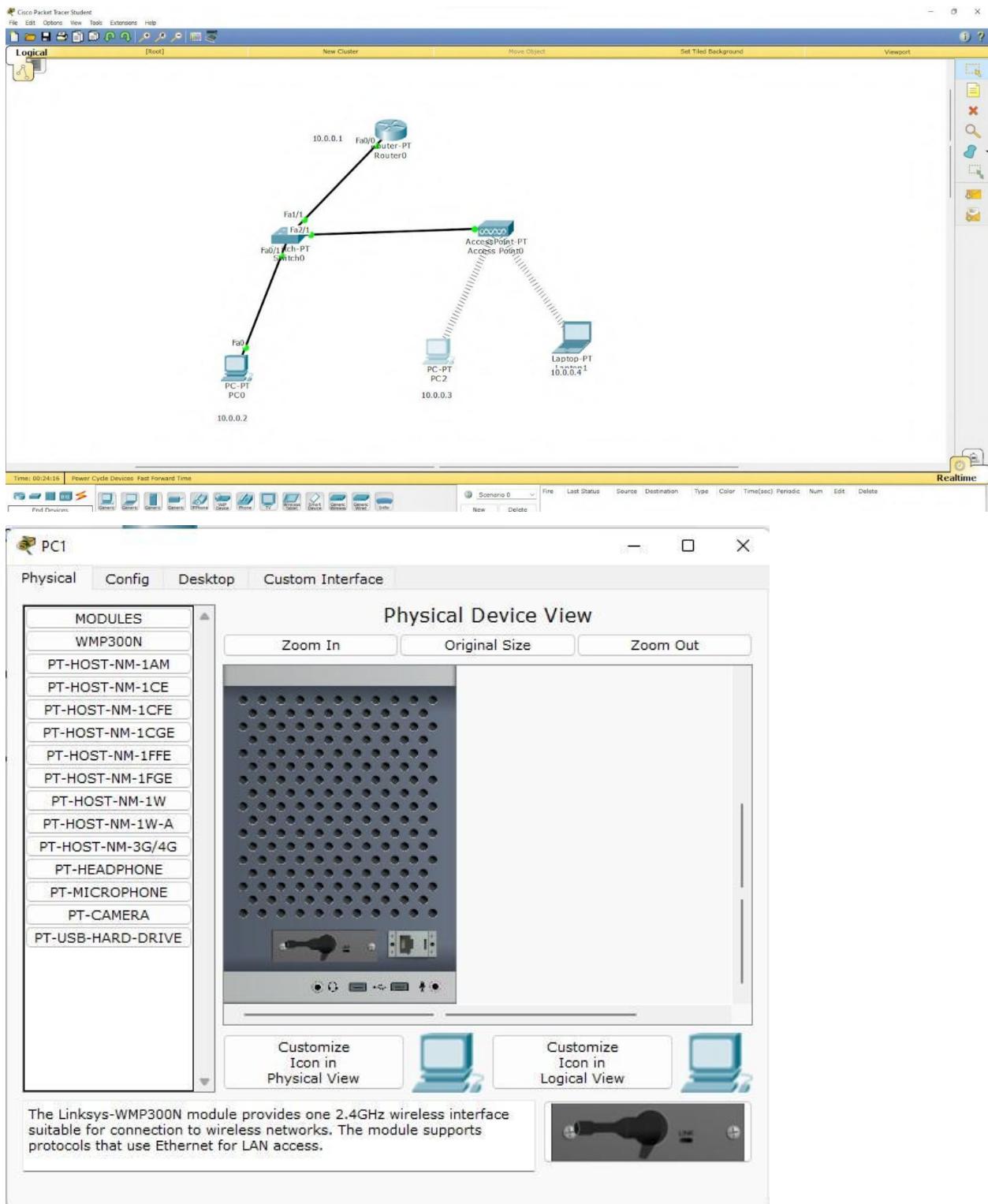
(7) For PC

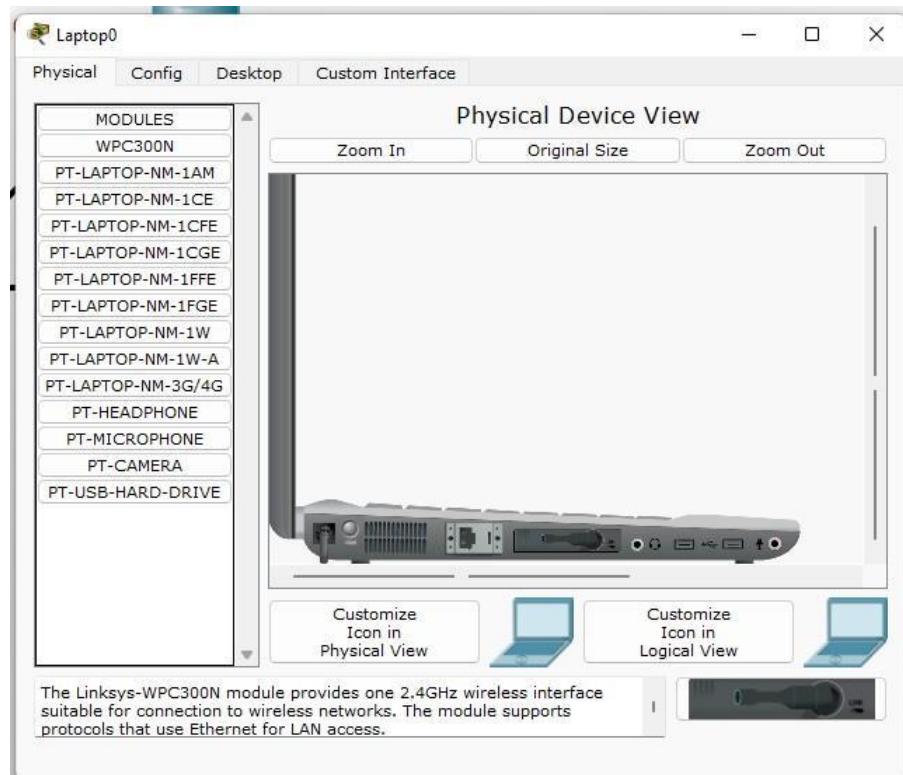
and

192



TOPOLOGY:





OUTPUT:

```

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>

```

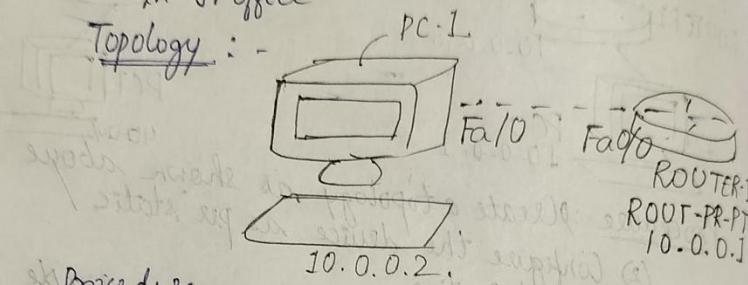
WEEK 12

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

OBSERVATION:

Aim : To understand the operation of TELNET by accessing the router is serve some from a PC in JTOffice

Topology :-



Procedure

1. Add a PC and a Router.

Connect PC to Router-1 using Cable across one wire from Fast Ethernet 0 port if PC to Fast Ethernet 0/0 port of Router-1

2. PC-1 Configuration.

Click on PC-1 → Config → Interface → FastEthernet 0.

Assign a static ip address [10.0.0.2] and subnet mask 255.0.0.0

3. Router - Configuration

Router > enable

Router # config

Router (config) # host name R1

R1 (config) # enable secret

R1 (config) # interface Fa0/0

R1 (config-if) # ip address 10.0.0.1

255.0.0.0

R1 (config-if) # no shutdown

R1 (config)
R1 (config)
% log
% log

② R1 (config)
R1 (config)
R1 (config)

Result

Ping
PC

P

Re

Re

Re

P

f

D

OUTER-1
Γ-PR-PT
[O.O.]

R1 (config-if) # line vty 0.5
R1 (Config-line) # login
% login disabled on line 132 until 'password' is set
% login disabled on line 132 until 'password' is set
% login disabled on line 132 until 'password' is set
% login disabled on line 132 until 'password' is set
% login disabled on line 132 until 'password' is set
R1 (Config-line) # password Ps
R1 (Config-line) # exit
R1 (Config) # exit
R1 # wr (to save changes in router)

Result: Router 1 through PC-1 in Command Prompt:

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=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%)
Approximate round trip times in ms
Minimum = 0ms, Maximum = 0ms
Average = 0ms

Aim - T
the
Topology

Proce

(1) Drag

(2) Con

(3) Co

(4) P

(5)

(6)

(7)

PC> telnet 10.0.0.1

Trying 10.0.0.1.. open

User Access Verification

Password:

Y is enable:

Password:

Y> # show ip route

Codes C-Connected S-Static L-IGRP

R-RIP M-Mobile

D-CIGRP EX-EIGRP external,

O-OSPF IA-OSPF

N1-OSPF NSSA external type 1

N2-external type

E1-OSPF External type 1 E2-OSPF

External type 2

IS-IS L1 IS-IS level 1

IS-IS L2

*-Candidate default V-Pee user

static route

P-periodic download static

route

Gateway of last resort is not set

C- 10.0.0.0 is directly

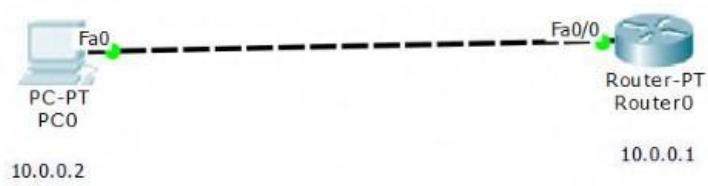
connected, Fast ethernet 0/0.

Serial port is selected

Serial port is selected

Serial port is selected

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).

CODE:

```
#include<stdio.h>
int arr[17];

void xor(int x[], int y[])
{
    int k=0;
    for(int i=1;i<16;i++)
    {
        if(x[i]==y[i])
            arr[k++]=0;
        else
            arr[i]=1;
    }
}

void main()
{
    int dd[17],div[33],ze[17],i,k;

    printf("Enter the dataword \n");
    for(i=0;i<17;i++)
        scanf("%d",&div[i]);

    for(i=i;i<33;i++)
        div[i]=0;

    for(i=0;i<17;i++)
        ze[i]=0;
    printf("Enter dividend \n");
```

```

for(i=0;i<17;i++)
    scanf("%d",&dd[i]);

i=0;
k=0;
for(i=i;i<17;i++)
    arr[k++]=div[i];
while(i<33)
{
    if(arr[0]==0)
        xor(arr,ze);
    else
        xor(arr,dd);

    arr[16]=div[i++];

}
k=0;
for(i=17;i<33;i++)
    div[i]=arr[k++];
printf("Codeword: ");
for(i=0;i<33;i++)
    printf("%d",div[i]);

for(i=0;i<17;i++)
    arr[i]=0;
printf("\nAt receiver end \n");

```

```

k=0;

for(i=i;i<17;i++)
    arr[k++]=div[i];
while(i<33)
{

```

```

if(arr[0]==0)
    xor(arr,ze);
else
    xor(arr,dd);

arr[16]=div[i++];

}

k=0;
for(i=17;i<33;i++)
    div[i]=arr[k++];

printf("Codeword: ");
for(i=0;i<33;i++)
    printf("%d",div[i]);
}

```

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: 1011001111001011100000000000000
Process returned 1 (0x1)   execution time : 49.507 s
Press any key to continue.

```

OBSERVATION:

LAB - 13

Write a program for error detecting code using CRC-CCITT

```

#include <stdio.h>
#include <string.h>
#define N string (poly)
char data[30], check_value[30];
poly[10];
int data[3], length, i, j;
void XOR
{
    for(j=1, j<1, j++)
        check_value[j] = { C, check_value[j] == poly[j], '0' };
}
void receiver()
{
    printf("Enter the received data: ");
    scanf("%d", data);
    printf("Data received : %s", data);
    CRC();
    for(i=0, (i<n-1) && (check_value[i]==i+1)
        if (N-1)
            printf("Error Detected.");
        else
            printf("No error \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[i] = check_value[j+1];
        check_value[j] = data[j+1];
    } while (i <= data.length - 1, N+1);
}

```

int main()
 { printf("Enter data: ");
 scanf("%s", &data);
 printf("Enter check value: ");
 scanf("%s", &check_value);
 data_length = strlen(data);
 for(i=0, i<data_length, i++)
 data[i];
 printf("Data: ");
 for(i=0, i<data_length, i++)
 printf("%c", data[i]);
 printf("Check Value: ");
 for(i=0, i<check_value.length, i++)
 check_value[i];
 printf("Sent Check Value: ");
 for(i=0, i<check_value.length, i++)
 printf("%c", check_value[i]);
 return 0;
 }

Output: Enter data:
 Data: 1234567890123456789012345678901234567890
 Check Value: 1234567890123456789012345678901234567890
 Sent Check Value: 1234567890123456789012345678901234567890


```

int main()
{
    printf("Enter data to be transmitted"),
    scanf("%s", &data);
    printf("Enter the divisor polynomial.");
    scanf("%s", &poly);
    data_length = strlen(data);
    for(i = data_length; i < data_length + N-1;
        data[i] = 0);
    printf("Data padded with n-1 zeros: %s", data);
    ccc();
    printf("CRC value is: %s) check_value",
    for(i = data_length; i < data_length + N-1;
        data[i] = check_value(i - data_length));
    printf("Final data word to be sent : %s", data);
    receiver();
    return(0);
}

```

Output: Enter the data to be transmitted:
Enter the divisor polynomial

101010

Data padded with 9's: 1010100

CRC value is: 001

Find code word to be sent: 10101001

Enter the received data: 10001000

Error detected

prompt: note=note
prompt: note=note
prompt: note=note
prompt: note=note
prompt: note=note

7

+1)

WEEK 14

Write a program for congestion control using Leaky bucket algorithm.

CODE:

```
#include <stdio.h>
#include <stdlib.h> // Include this for the rand() function
int main()
{
    int buckets, outlets, k = 1, num, remaining;
    printf("Enter Bucket size and outstream size\n");
    scanf("%d %d", &buckets, &outlets);
    remaining = buckets;
    while (k)
    {
        num = rand() % 1000; // Generate a random number between 0 and 999
        if (num < remaining)
        {
            remaining = remaining - num;
            printf("Packet of %d bytes accepted\n", num); // Added missing variable
        }
        else
        {
            printf("Packet of %d bytes is discarded\n", num);
        }
        if (buckets - remaining > outlets)
        {
            remaining += outlets; // Fixed the calculation
        }
        else
            remaining = buckets;
        printf("Remaining bytes: %d \n", remaining);
        printf("If you want to stop input, press 0, otherwise, press 1\n");
        scanf("%d", &k);
    }
}
```

```

}

while (remaining < buckets) // Fixed the condition
{
    if (buckets - remaining > outlets)
    {
        remaining += outlets; // Fixed the calculation
    }
    else
        remaining = buckets;
    printf("Remaining bytes: %d \n", remaining);
}
return 0; // Added a return statement to indicate successful completion
}

```

OUTPUT:

```

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> □
```

OBSERVATION:

Write a program for congestion control using leaky Bucket

```

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

```

Output

Enter the bucket
 Enter outgoing
 Enter Number
 Enter the size
 Bucket buffer
 in buffer
 Enter the
 Bucket buffer
 After outgoing
 of 5000

Output

Enter the bucket size = 5000

Enter outgoing rate = 2000

Enter number of input = 2

Enter the incoming packet size = 3000

Bucket buffer size 3000 out of 5000
in buffer.

Enter the incoming packet size = 1000

Bucket buffer size 2000 out of 5000

After outgoing a packet out
of 5000 in buffer

TEST 1 : 2000 - 1000 - 5000

initial buffer size = 5000, total buffer size = 5000

total of 5000 / 2 = 2500 - 1000 = 1500

packet size = 1000, total buffer size = 1500

total buffer size = 1500 - 1000 = 500

total buffer size = 500 - 1000 = -500

(-500) / 2 = -250 - 1000 = -1250

(-1250) / 2 = -625 - 1000 = -1625

packet size = 1000, total buffer size = -1625

total buffer size = -1625 - 1000 = -2625

total buffer size = -2625 - 1000 = -3625

packet size = 1000, total buffer size = -3625

total buffer size = -3625 - 1000 = -4625

packet size = 1000, total buffer size = -4625

total buffer size = -4625 - 1000 = -5625

packet size = 1000, total buffer size = -5625

total buffer size = -5625 - 1000 = -6625

packet size = 1000, total buffer size = -6625

total buffer size = -6625 - 1000 = -7625

packet size = 1000, total buffer size = -7625

total buffer size = -7625 - 1000 = -8625

WEEK 15

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.

CODE:

ClientTCP.py

```
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()
```

ServerTCP.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 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())
    )
```

```

print ("\nSent contents of " + sentence)
file.close()
connectionSocket.close()

```

OUTPUT:

The image shows two separate Python IDLE shells running simultaneously. Both windows have the title 'IDLE Shell 3.11.4'.

Left Window (ClientTCP.py):

```

File Edit Shell Debug Options Window Help
Python 3.11.4 (tags/v3.11.4:d2340ef, 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\lkm2lcs065\ClientTCP.py =====
Enter file name:ServerTCP.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,"z")
    l=file.read(1024)
    connectionSocket.send(l.encode())
    print('\nSent contents of' + sentence)
    file.close()
    connectionSocket.close()

>>>

```

Right Window (ServerTCP.py):

```

File Edit Shell Debug Options Window Help
Python 3.11.4 (tags/v3.11.4:d2340ef, 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\lkm2lcs065\ServerTCP.py =====
The server is ready to receive
Sent contents ofServerTCP.py
The server is ready to receive

```

OBSERVATION:

LAB - 15

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

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("Enter file name:")
clientSocket.send(sentence.encode())
filecontents = clientSocket.recv(1024)
print(filecontents.decode())
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 1:

print("The Server is ready to receive")

connectionSocket, address = serverSocket.accept()

Sentence = connectionSocket.recv(1024).decode()

file = open(sentence, "r")

l = file.read(1024)

connectionSocket.send(str(l).encode())
connectionSocket.close()

Output

Server Side:

The server is

Client side -

Enter file na

The contents

displayed

Server Side:

Sent conten

S. I
S. I

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.

CODE:

ClientUDP.py

```
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()
```

ServerUDP.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 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()

```

OUTPUT:

The image shows two separate Python IDLE shells running on Windows 10. Both shells are titled 'IDLE Shell 3.11.4'.

Left Shell (Server Side):

- File Edit Shell Debug Options Window Help
- Python 3.11.4 (tags/v3.11.4:d2340ef, 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\lmb21cs065\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()
- >>>

Right Shell (Client Side):

- File Edit Shell Debug Options Window Help
- Python 3.11.4 (tags/v3.11.4:d2340ef, 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\lmb21cs065\ServerUDP.py
- The server is ready to receive
- Sent contents of ServerUDP.py

OBSERVATION:

LAB-16 UDP socket

client UDP.PY

```

from socket import *
servername = '127.0.0.1'
serverport = 12000
clientsocket = socket(AF_INET, SOCK_DGRAM)
sentence = input("Enter file name: ")
clientsocket.sendto(sentence.encode(), (servername, serverport))
filecontents, serverAddress = clientsocket.recvfrom(4096)
print("In Reply from Server:\n")
print(filecontents.decode('utf-8'))
for i in filecontents:
    print(str(i), end=' ')
clientsocket.close()
clientsocket.close()

```

server P.D.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 1:
    sentence, clientAddress = serversocket.recvfrom(2048)
    sentence = sentence.decode('utf-8')
    file = open(sentence, "r")
    content = file.read(2048)
    serversocket.sendto(content.encode('utf-8'), clientAddress)
    print("Sent contents of file")
    for i in sentence:
        print(str(i), end=' ')
    file.close()

```

file.close()

output

Server side
The server is ready to receive
Enter file name:
The contents of file displayed

Client side
Sent contents

S.P.T

File name:
contents displayed
Server is ready to receive
Enter file name:
The contents of file displayed

socket
AF_INET, SOCK_DGRAM
(filename, "DGRAM")
byte sentence
port))
address = clientsocket
server : \n
ts decode('utf-8')
contents
d :)
e0
el)

(AF_INET,
SOCK_DGRAM)
127 0 0 1
(serverport))
ceme
err = serversocket
2048
decode('utf-8')
ce, "l")
2048
to (bytes (an,
utf-8))
contents of end-
end =)

file.close()
Output
Server side
The server is ready to receive
client side
Enter file name : server UDP.PY
The contents of the file server UDP
are displayed here
Server side
sent contents of server UDP.PY
Script
server -> client : so will. brief
will be lots of it see response
server takes message. will. get the message.
and for response will take msg. will ps
and. server will see msg. been
will be received. after that
will receive msg. so will. brief
will see msg. will
will ad will be msg
also if msg -> server
msg will be msg. will be msg. will
to server. q1 = socket. q1
will be msg.
will be msg. will be msg. will be msg. will be msg.
will be msg. will be msg. will be msg. will be msg. will be msg.
will be msg. will be msg. will be msg. will be msg. will be msg.
will be msg. will be msg. will be msg. will be msg. will be msg.

WEEK 17

Tool Exploration -Wireshark

OBSERVATION:

LAB-17

Tool Exploration - Wireshark

Wireshark

Wireshark is a network protocol analyser or an application that captures packets from a network interface such as from your computer to your home off a or the internet. Wireshark is the most often used packet sniffer in the world.

Open Wireshark

And click on capture → start. Now you can see the packets that are sent by the system and received by the system and the protocol being used.

And we can see the source and destination address of the packet.

If we click on a particular packet now you can see the ASCII code in the bottom.

To see ~~only~~ your system particular network in display filter type ip. address == ip address of the pc.
ipaddress - 10.124.7.1

Note: To know ip address of the system open cmd and type ipconfig.

You can now see the ip address of your system.

And we can see
the type of the pr
To colour packet
and stick on

✓ S.T.

- Wireshark

work protocol
calculator that
is neutral
as from
home office
is often used
today

→ starts
packets that
and received
protocol being

source and
is of the

attacker packet
→ ASCII

system participation
by filter type
→ address of

4.7.)
address of the
and type

→ the ip
system.

And we can even filter packets by
the type of the protocol

To color packet but go to view
and click on change packet test.

St.