

VISVESVARAYA TECHNOLOGICAL UNIVERSITY

“JnanaSangama”, Belgaum -590014, Karnataka.



LAB REPORT on

Computer Network

Submitted by

Dhavan SK (1BM21CS054)

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 Network” carried out by **Dhavan SK (1BM21CS054)**, who is 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 Network - (22CS4PCCON)** work prescribed for the said degree.

Prof Swathi Sridharan
Assistant Professor
Department of CSE
BMSCE, Bengaluru

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

Index

Sl. No.	Experiment Title	Page No.
CYCLE 1		
1	Create a topology and simulate sending a simple PDU from source to destination using hub and switch as connecting devices and demonstrate ping message	4
2	Configure IP address to routers in packet tracer. Explore the following messages: ping responses, destination unreachable, request timed out, reply	13
3	Configure default route, static route to the Router	25
4	Configure DHCP within a LAN and outside LAN	30
5	Configure RIP routing Protocol in Routers	41
6	Configure OSPF routing protocol	45
7	Demonstrate the TTL/ Life of a Packet	51
8	Configure Web Server, DNS within a LAN	58
9	To construct simple LAN and understand the concept and operation of Address Resolution Protocol (ARP)	63
10	To understand the operation of TELNET by accessing the router in server room from a PC in IT office	69
11	To construct a VLAN and make the PCs communicate among a VLAN	74
12	To construct a WLAN and make the nodes communicate wirelessly	80
CYCLE 2		
13	Write a program for error detecting code using CRC CCITT (16-bits)	84
14	Write a program for congestion control using Leaky bucket algorithm	93
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	98
16	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	102
17	Tool Exploration -Wireshark	106

LAB-1

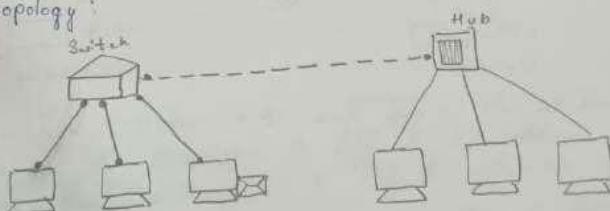
Create a topology and simulate sending a simple PDU from source to destination using hub and switch as connecting devices and

1) Create a topology & Simulate sending a simple PDU from Source to destination using Hub and Switch as connecting devices and demonstrate ping message.

Procedure

- * Place a hub and 3 PC and connect all the 3 PC's to the hub using copper straight through and set the 3 PC's with different IP address.
- * Take a switch and 3 PC's and connect all the 3 PC's to the switch using copper straight through and set the 3 PC's with different IP address.
- * Check if the packet is transferred to the different PC's
- * Then connect the switch and the hub
- * And the check if the packet from one PC [connected to switch] is transferred to another PC [connected to hub]

Topology :



Output :

Pinging 10.0.0.1 with 32 bytes of data:
Reply from 10.0.0.1 : bytes=32 time=8ms TTL=10
Reply from 10.0.0.1 : bytes=32 time=0ms TTL=10
Reply from 10.0.0.1 : bytes=32 time=4ms TTL=10
Reply from 10.0.0.1 : bytes=32 time=4ms TTL=10

Reply from 10.0.0.1? bytes=32 time: 4ms TTL=118

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 = 4ms, Average = 2ms

Switch

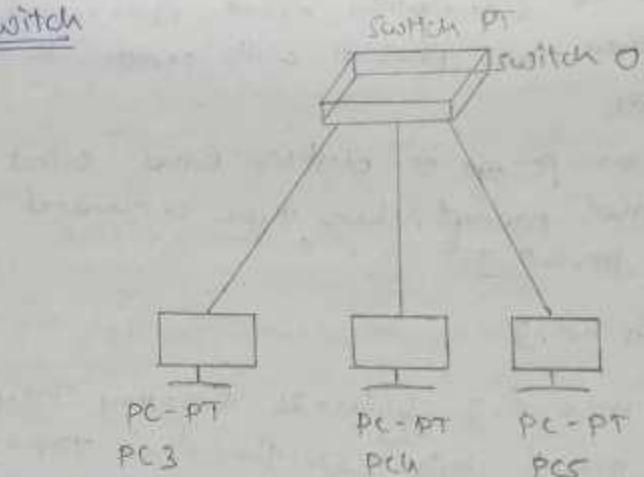
Port - 1
Fast Ethernet 0 IP address
10.0.0.4

Port - 2
Fast Ethernet 0 IP address
10.0.0.2/8

Port - 3
IP address
10.0.0.3/8

N
15/6/2023

Switch



procedure:-

Step 1:- Select switch and 3pc's.

Step 2:- Set IP address for all the pc's

10.0.0.4, 10.0.0.8, 10.0.0.6

PC → catalog → fast ethernet → IP address.

Step 3:- Connect pc's to the switch by setting copper straight-through.

Step 4:- Add simple PDU.

Select source and destination.

Step 5:- Go to simulation mode and click on Auto capture / play.

Step 6:- Click on PC's Desktop → command prompt

ping message

PC > ping 10.0.0.6

Pinging 10.0.0.6 with 32 bytes of data.

Reply from 10.0.0.6 bytes=32 time=4ms TTL=128

ping statistics for 10.0.0.6

packet sent = 4 Received = 4 Cost = 0 (0% lost)

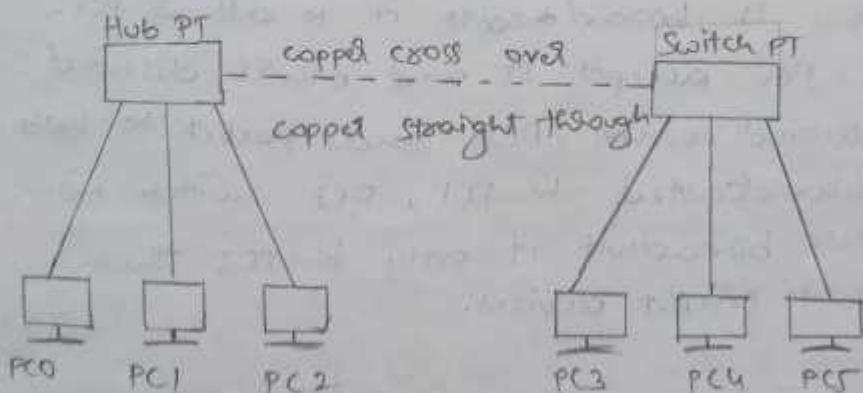
Appropriate round trip times in milliseconds

minimum = 4ms Maximum = 4ms Average = 4ms

Observation:-

When the first time the packet is sent the switch will distribute the packet with all the devices. Once it reaches around the IP address it will only send packet to the destination and send acknowledgement to the source.

Switch-Hub Connection



Step 1:- Previously drawn hub-topology and switch topology are connected through copper cross over. In hub port 3 is used in switch port ethernet 3/1 is used.

Step 2!- Add simple PDC from PC0 to PC3

ping 10.0.0.4.

Pinging 10.0.0.4 with 32 bytes of data

Reply from 10.0.0.4: bytes=32 time=1ms TTL=128

Reply from 10.0.0.4: bytes=32 time=1ms TTL=128

Reply from 10.0.0.4 : bytes = 32 time = 1ms TTL = 128
Reply from 10.0.0.4 : bytes = 32 time = 1ms TTL = 128
Ping satisfies to 10.0.0.4.
Packets : sent = 4 Received = 4 Lost = 0 (0% loss)

Appropriate round trip times in milliseconds
minimum = 1ms Maximum = 4ms Average = 1ms

Observation:-

In simulation mode PCD sends packet to hub sends it to PC1, PC2 & switch board with it to PC3, PC4 and PC5.

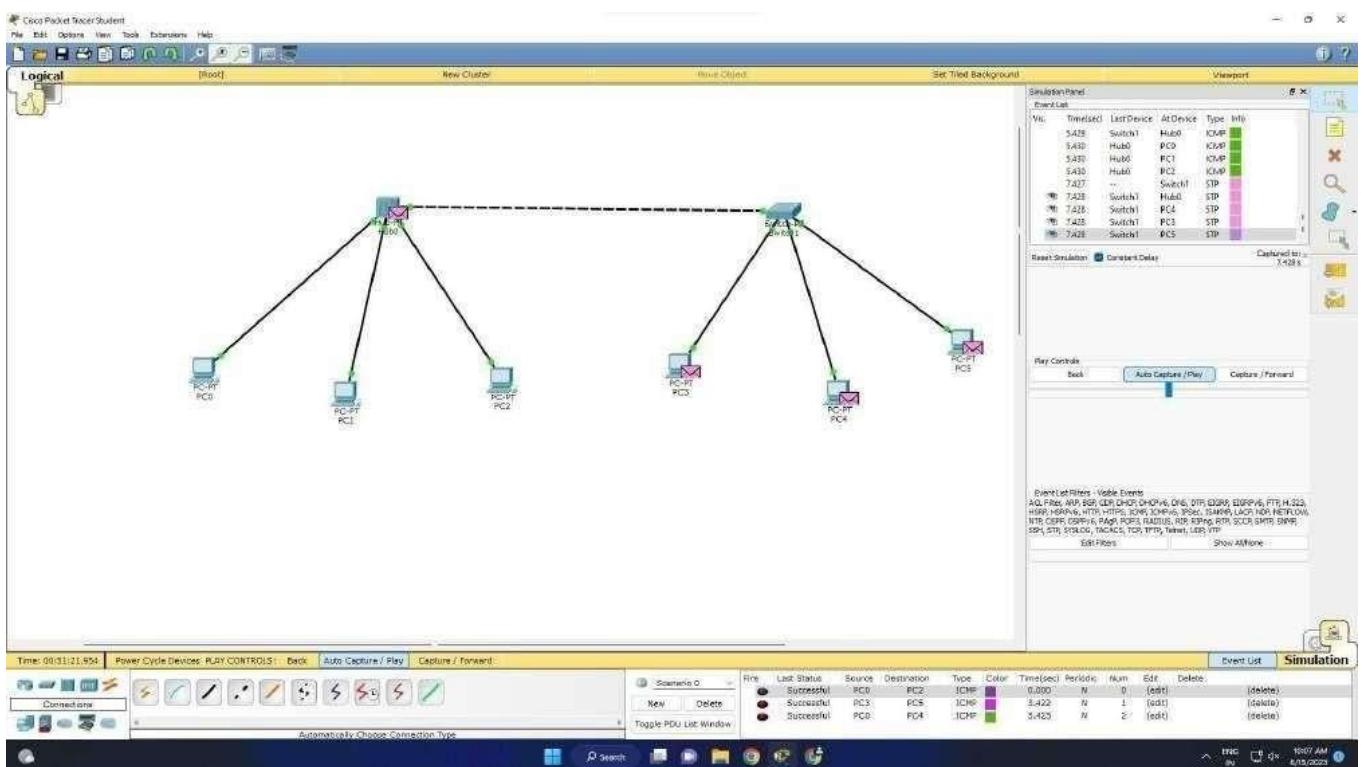
PC1, PC2, PC4 and PC5 discard them, PC3 accepts and sends acknowledgement to hub through switch.

Hub TS board = casts it to all 3 PCs.

Only PC0 accepts it and others discarded

In second round PC0 sends packet to hub & broadcasted to PC1, PC2 switch now switch broadcast it only to PC3, thus switch it smart devices.

OUTPUT:



```

PC0: C:\> ping 192.168.1.2

Pinging 192.168.1.2 with 32 bytes of data:
Reply from 192.168.1.2: bytes=32 time=1ms TTL=128

Ping statistics for 192.168.1.2:
    Packets: Sent = 4, Received = 4 (100% loss),
    Approximate round trip times in millisecond:
        Minimum = 0ms, Maximum = 0ms, Average = 0ms

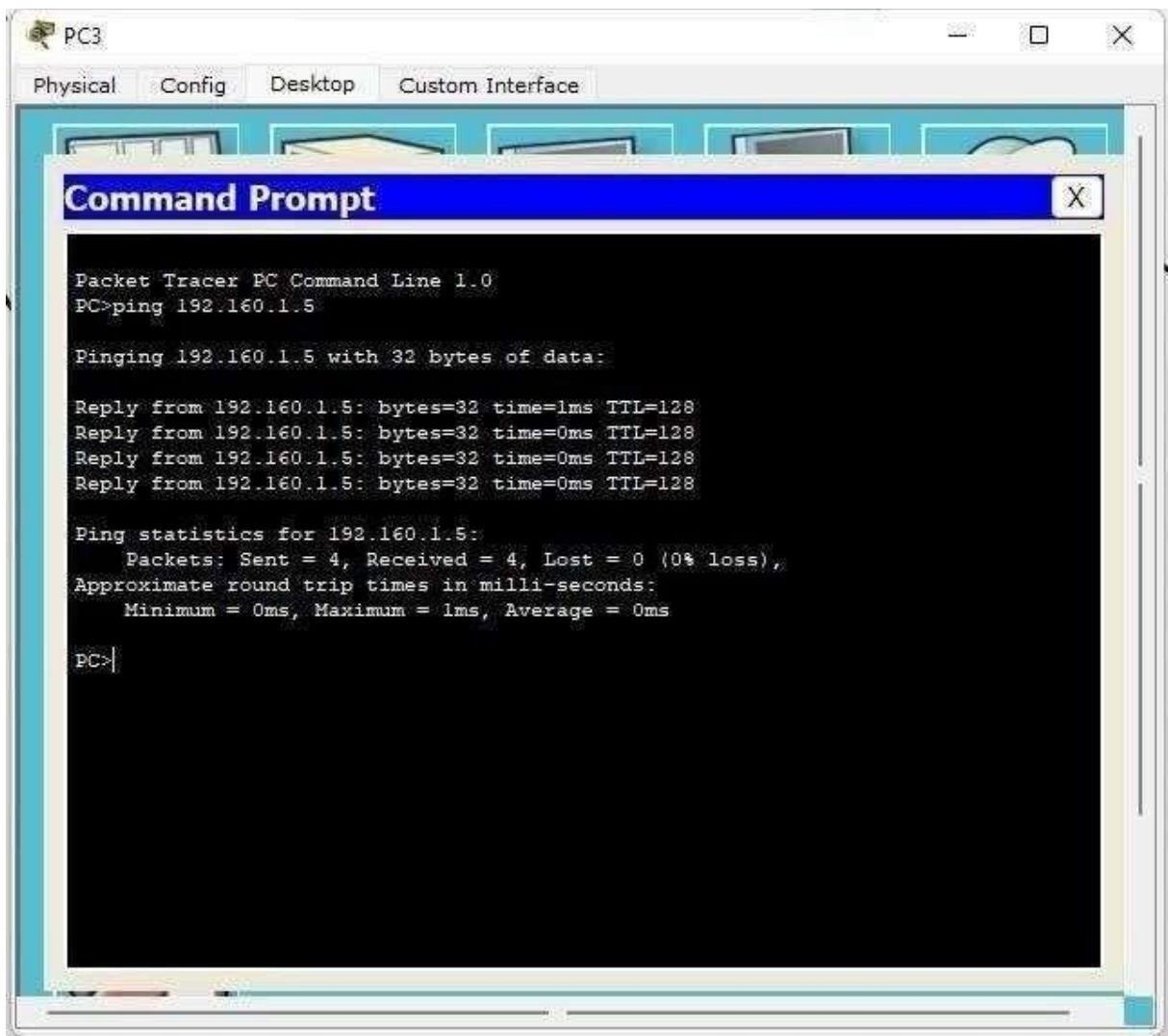
PC0: C:\> ping 192.168.1.3

Pinging 192.168.1.3 with 32 bytes of data:
Request timed out.
Request timed out.
Request timed out.
Request timed out.

Ping statistics for 192.168.1.3:
    Packets: Sent = 4, Received = 0, Lost = 4 (100% loss),
    Approximate round trip times in millisecond:
        Minimum = 0ms, Maximum = 0ms, Average = 0ms

PC0: C:\>

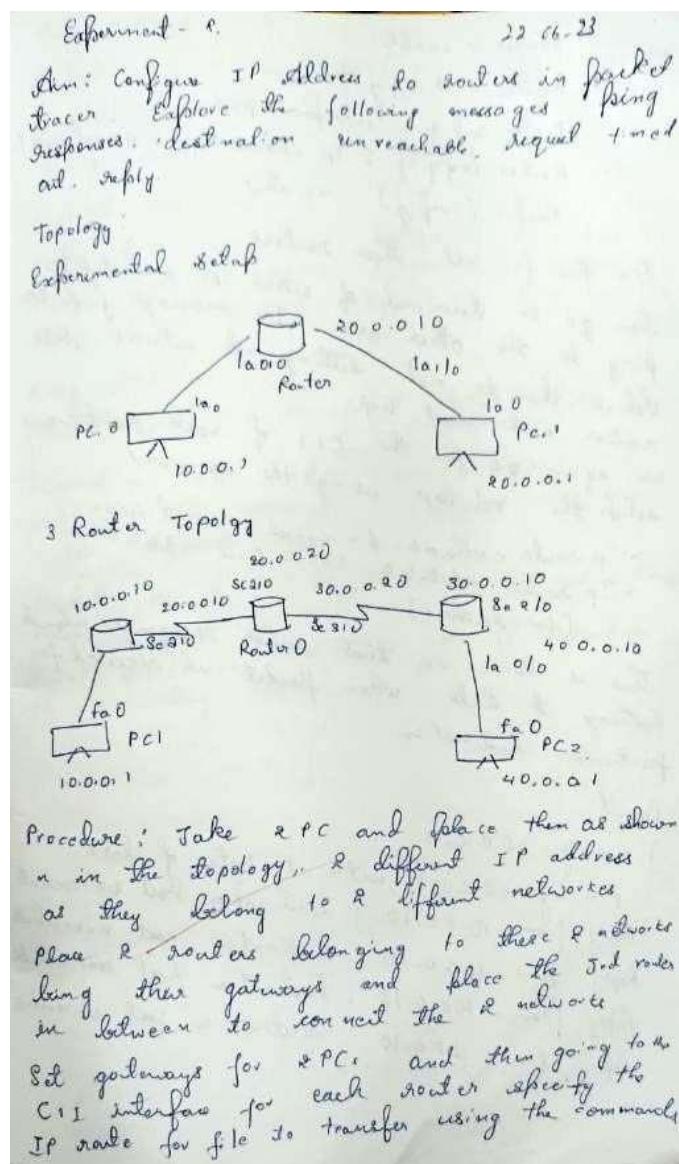
```



LAB 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:



Router > enable
Router > # config terminal
Router config : interface <port>
Router config if : Ip address <ip> <subnet mask>
Router (config) : no shutdown

Do this for all three routers

Then go to terminal of either PC and try to ping to the other one. The message fails to deliver due to not setting up network static router and next hop.

We again go for the CLI of each router and set up the "next hop" using the command.

```
> ip route <network-id> <mask> <next-hop>
> ip route 40.0.0.0 255.0.0.0 20.0.0.20
    (for router1)
```

This is done so that router recognizes which pathway to take when packet is received for particular destination

Result:

i) > ping 40.0.0.1

pinging 40.0.0.1 with 32 bytes of data

Reply from 10.0.0.10 : Destination host unreachable

Ping Statistic

packets sent = 4 ; Received = 0 ; loss = 4 (100% loss)

i) > ping 40.0.0.1

ping 40.0.0.1 with 32 bytes of data

Request timed out

Reply from 40.0.0.10 bytes = 32 time = 2ms TTL

Reply from 40.0.0.10 bytes = 32 time = 2ms TTL

Reply from 40.0.0.10 bytes = 32 time = 2ms TTL

Ping Statistic

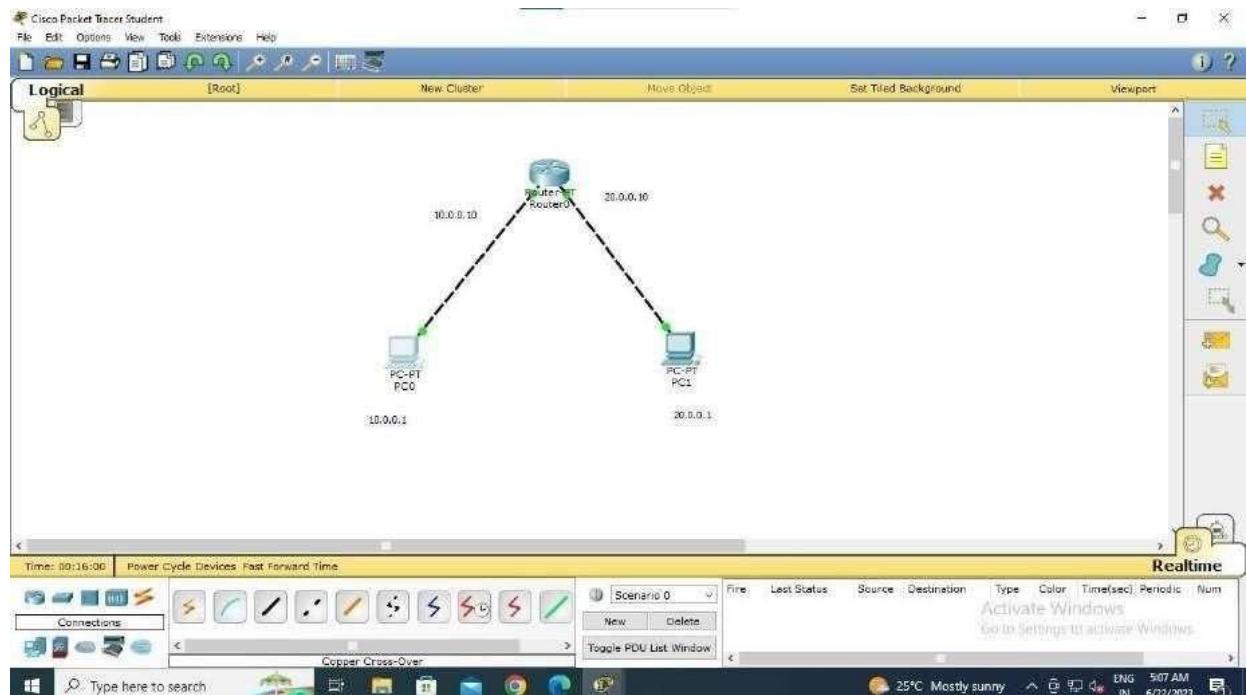
packets sent = 4 ; Received = 1 , loss = 1 (25% loss)

Observation: The router connects LAN to the internet. It connects different networks with different loss.

packets are forwarded to the destination through network hopping.

Serial ports are used to connect 2 routers
the connecting cable

Topology



The screenshot shows a 'Command Prompt' window for PC0. The window title is 'PC0'. The tabs at the top are Physical, Config, Desktop, and Custom Interface. The main area is a terminal window titled 'Command Prompt' with the following text:

```
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>
```

Default IP address LAB - 3

Topology Similar to that of Lab-2 that contains
5 routers & PC's connected as shown in that

Procedure:

- Select PC's and Router and configure them with suitable IP address.
- Make connections to all the devices using suitable connection links.
- for the routers to link between pc and other routers. use CL2 mode and start typing the commands
- Now enter Enable → Enter
- Config T enter
- En Interface fastethernet 0/0
- IP address 10.0.0.10 255.0.0.0
- Now shut

Repeat this step similarly to all the routers
In order to make default path

- Type . in config T
- IP route Destination Subnet Mask Intermediate device.
- IP router 0.0.0.0 0.0.0.0 20.0.0.10
Similarly perform this to all the routers
- Show IP route enter
 - 10.0.0.0/8 is directly connected to fastethernet 0/0
 - 20.0.0.0/8 is directly connected. via 0.0.0.10
0.0.0.10 [1/0] via 20.0.0.10

Similarly all the routers are connected.

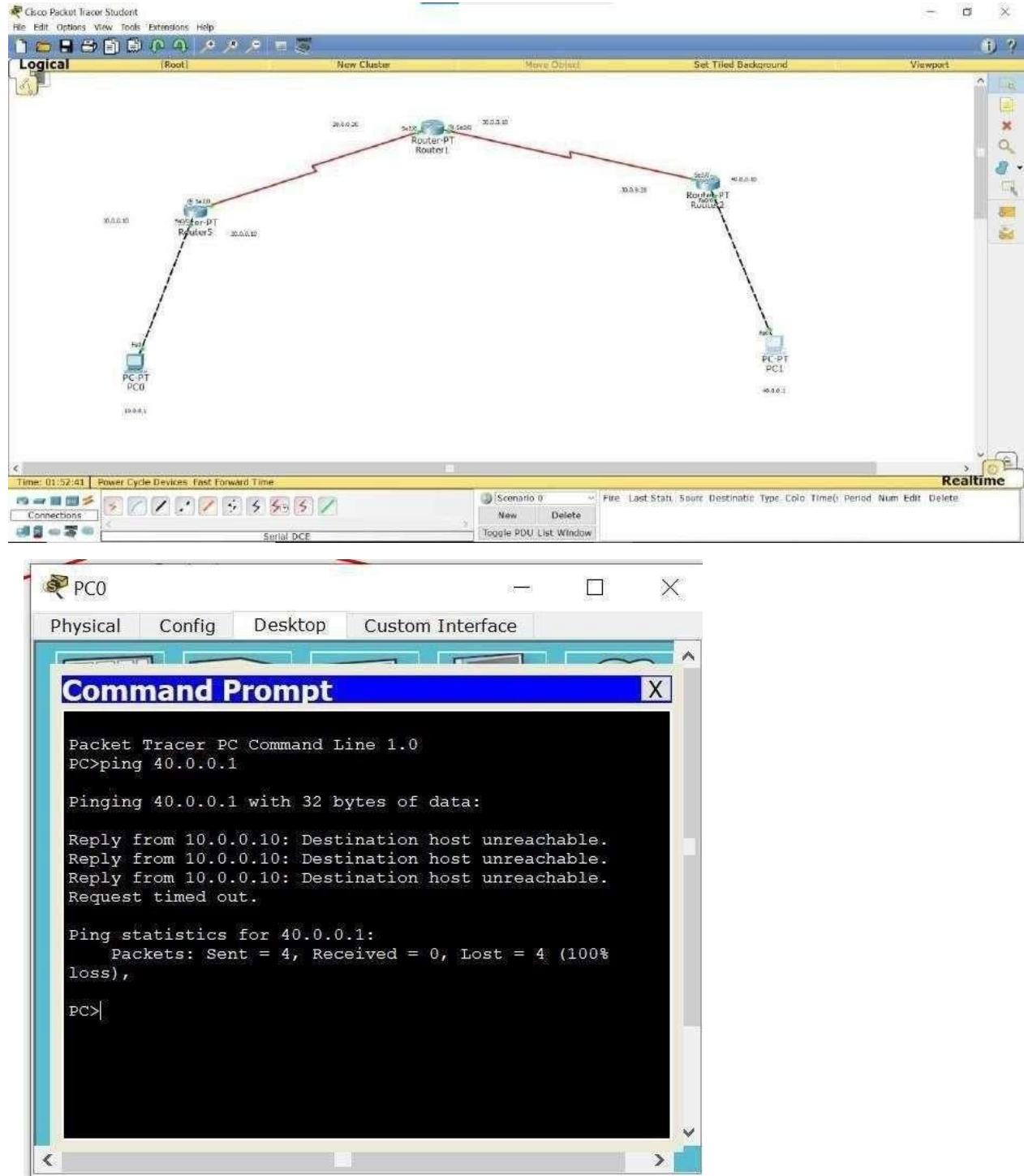
Observation:

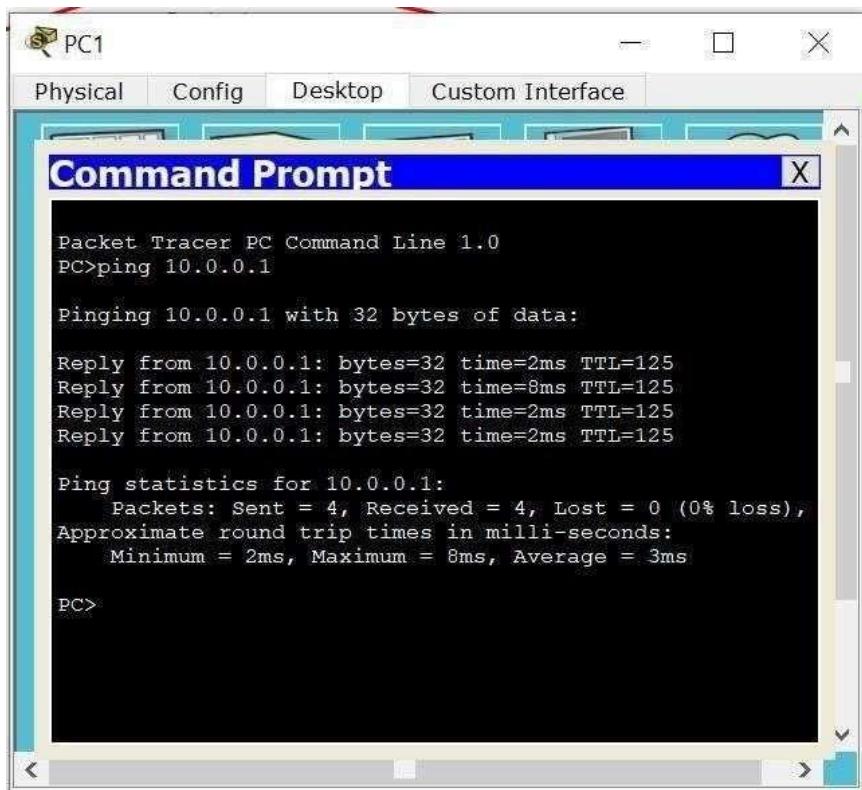
In the previous one, we have given the IP address to all the routers with default subnet mask and intermediate IP address of those particular device. But here in this experiment we use a default IP address i.e. 0.0.0.0 and Subnet Mask 0.0.0.0 so that it can create a pass through channel to all the packets that are sent will be transferred by the intermediate device. This is generally used in large no. of device connection.

Topology:

8/1/8

Topology:

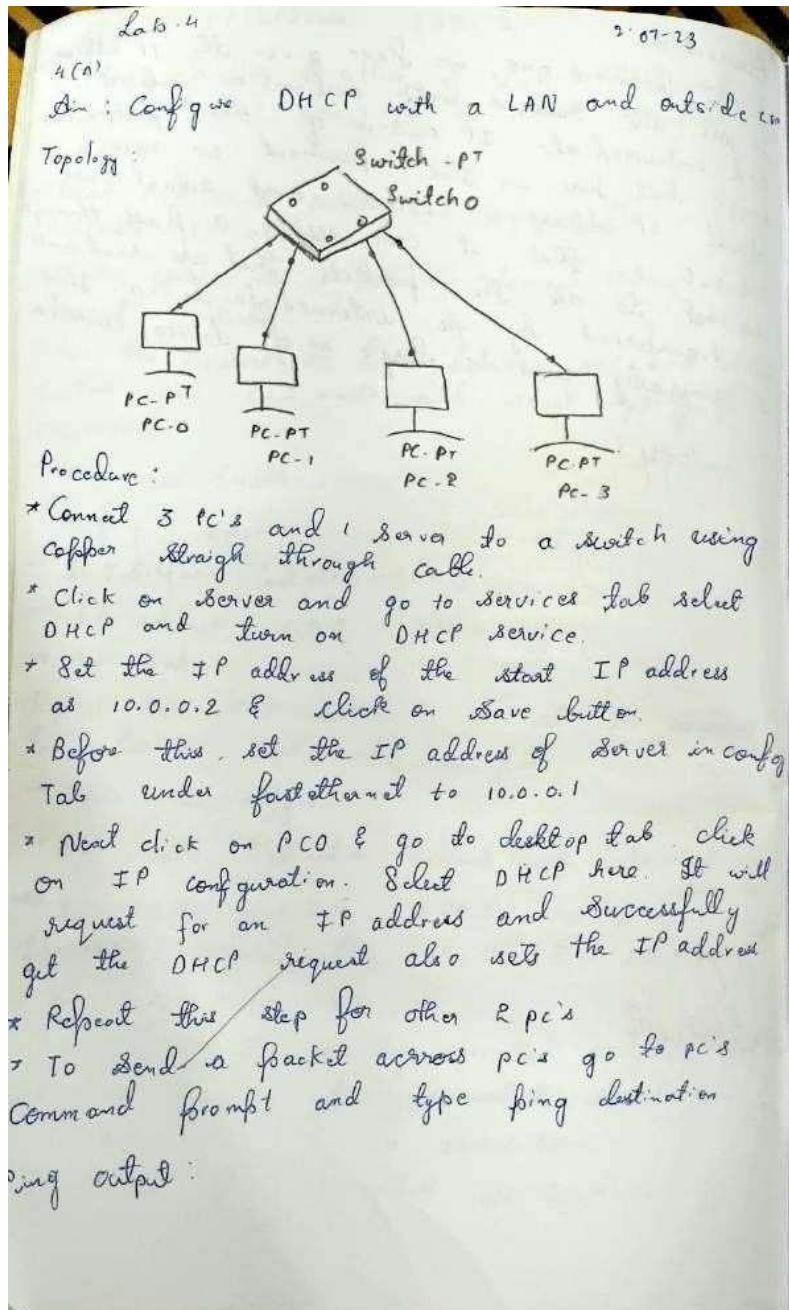




LAB 4

Configure DHCP within a LAN and outside

LAN.OBSERVATION:



Packet tracer PC command line 10

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

Ping statistics from 10.0.0.3

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

Approximate round trip times in ms

Minimum = 0ms Maximum = 1ms Average = 0ms

Observation:

* DHCP is used to dynamically assigns an IP address to any device or node

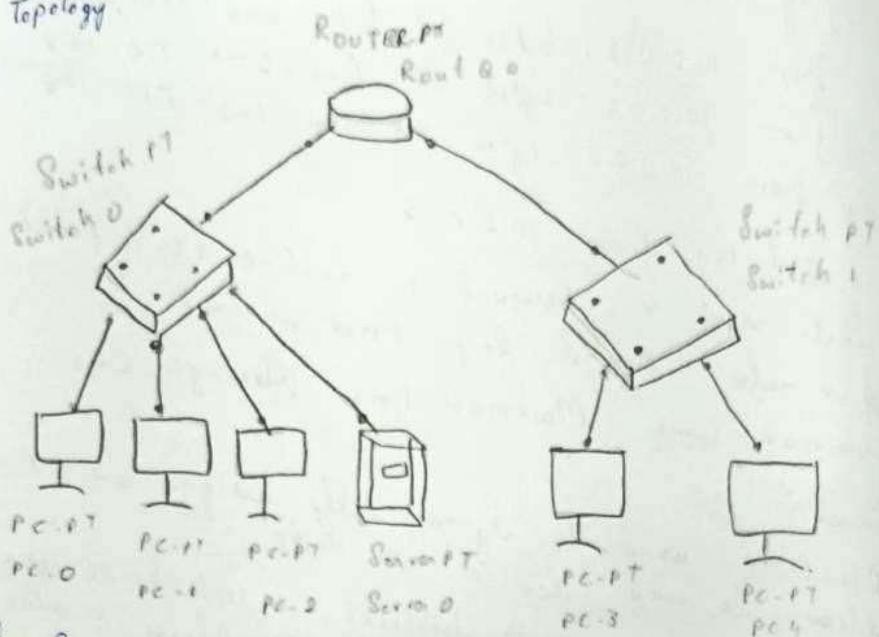
* It is client - server protocol in which servers manage a pool of unique IP address & also about client configuration parameters

* DHCP enabled clients sends a request to DHCP server when they want to connect to a network.

"(B)"

Aim : Configure DHCP with a LAN and outside LAN

Topology



Procedure :

Add a router, a switch and 8 PCs to 4(A)
program network & connect the router to both
switches

→ Set the server IP address of server & with the
helps of server set the first 3 PCs IP address
through DHCP

→ click on Server

→ go to desktop → IP configuration

→ Add IP address, Subnet Mask and gateway

IP address 10.0.0.1

Subnet Mask 255.0.0.0

GATE way 10.0.0.20

Crating wrong

Step 3 : Configure the Router

→ click on router go to CL 2

nable

```
router # config t  
router (config) # fastethernet 0/0  
router (config) # ip address 10.0.0.20 255.0.0.0  
router (config) # no shutdown  
Router (config-if) # exit  
Router (config) # interface fastethernet 1/0  
Router (config) # ip address 20.0.0.20 255.0.0.0  
Router (config-if) # no shutdown  
Router (config-if) # exit  
exit
```

Couting table

Router > show ip route
10.0.0.0/1 is directly connected.

Step 4: Go to server

- select services then go to DHCP
- set service on
- set start IP address from the service

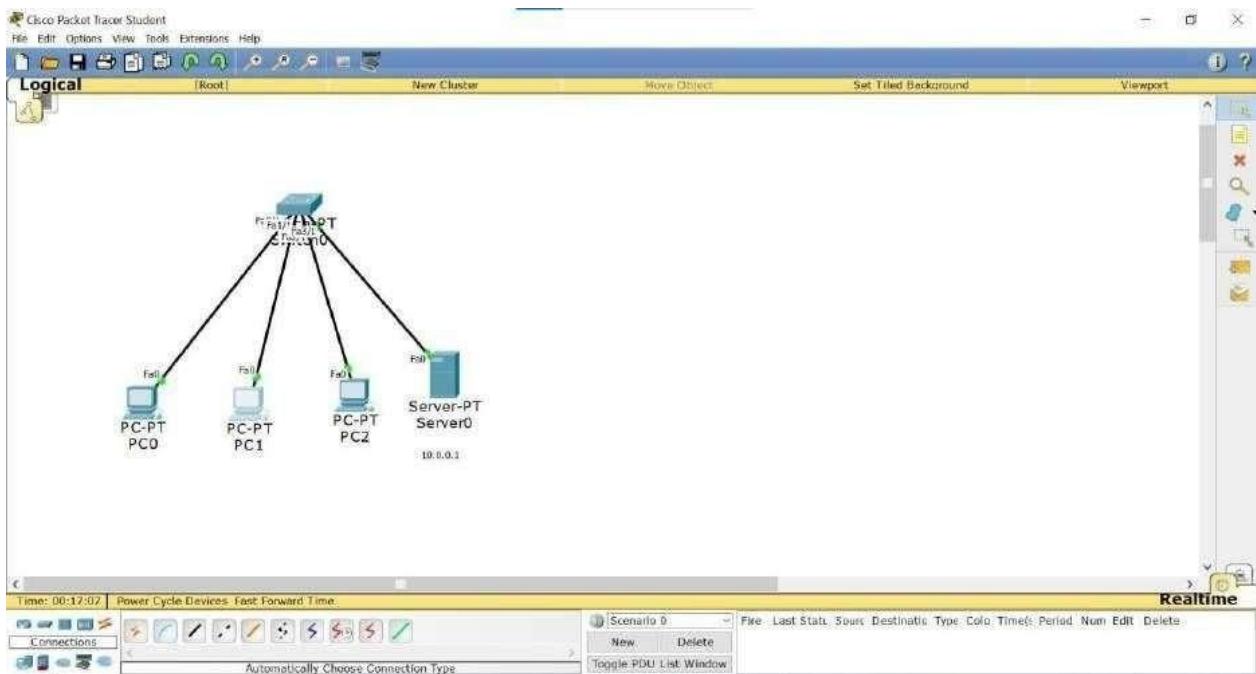
Step 5: Then configure the PCs

- Select a PC then desktop - go to IP configuration
- Select DHCP
- Repeat the same procedure for all other PCs

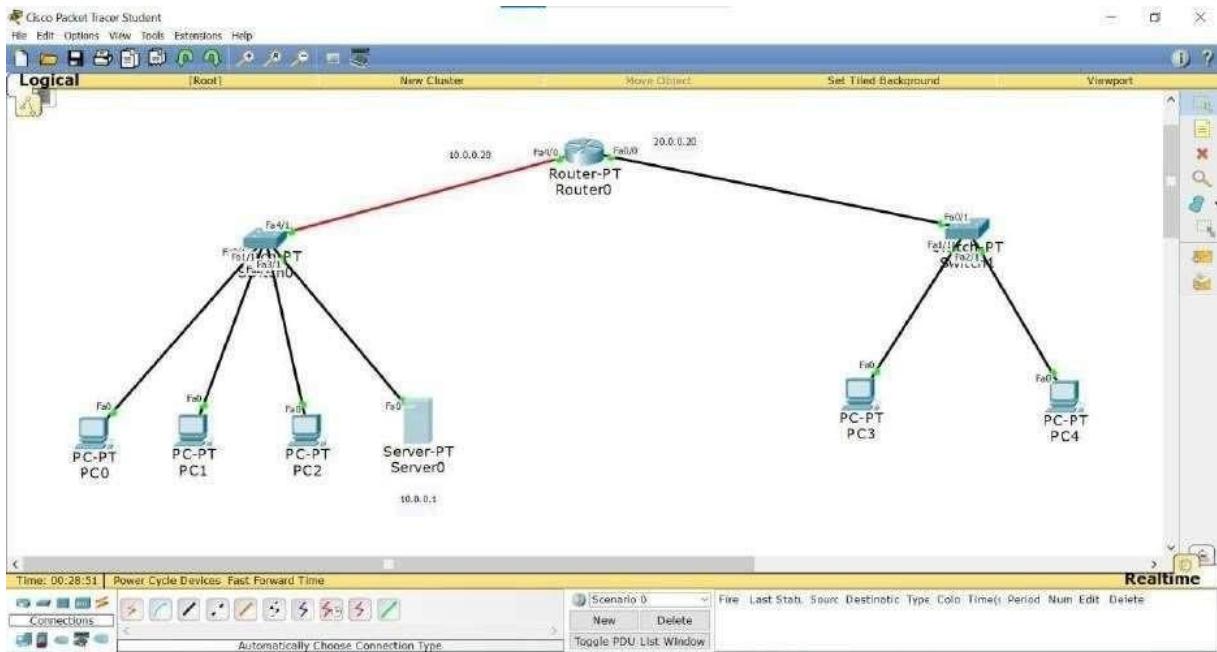
Observation:

- DHCP is used to assign IP address dynamically to different devices.
- To assign continuous IP address we create a server pool where we assign the starting IP address and a default gateway number.
- for PCs under different switches we create a different server pool again & start.

PROGRAM 4.1:

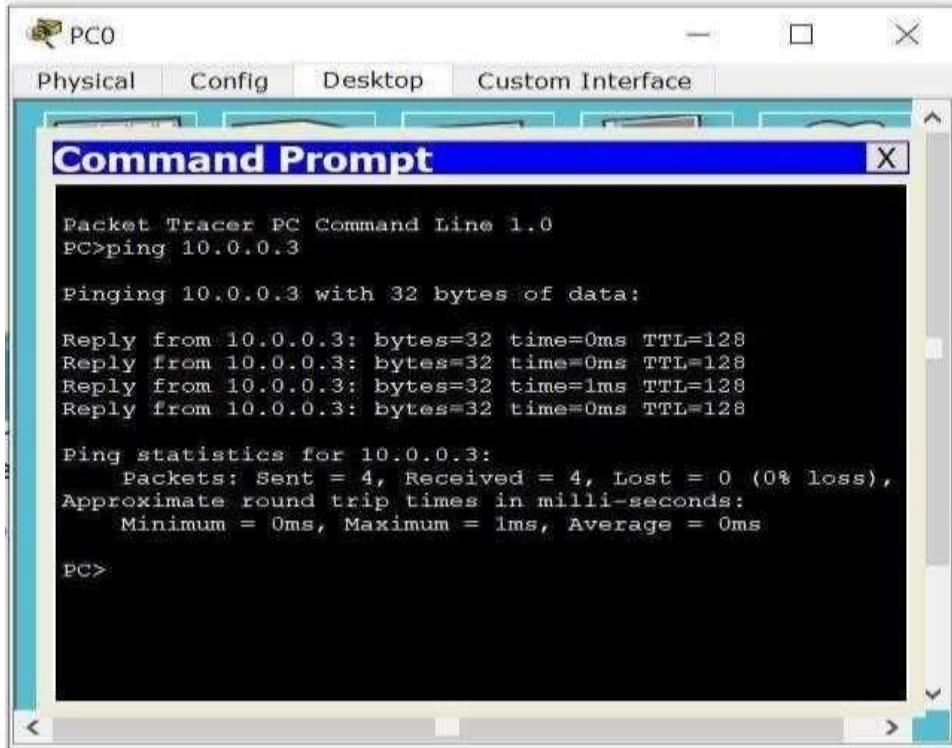


PROGRAM 4.2:



OUTPUT:

PROGRAM 4.1:



```
PC0
Physical Config Desktop Custom Interface

Command Prompt X

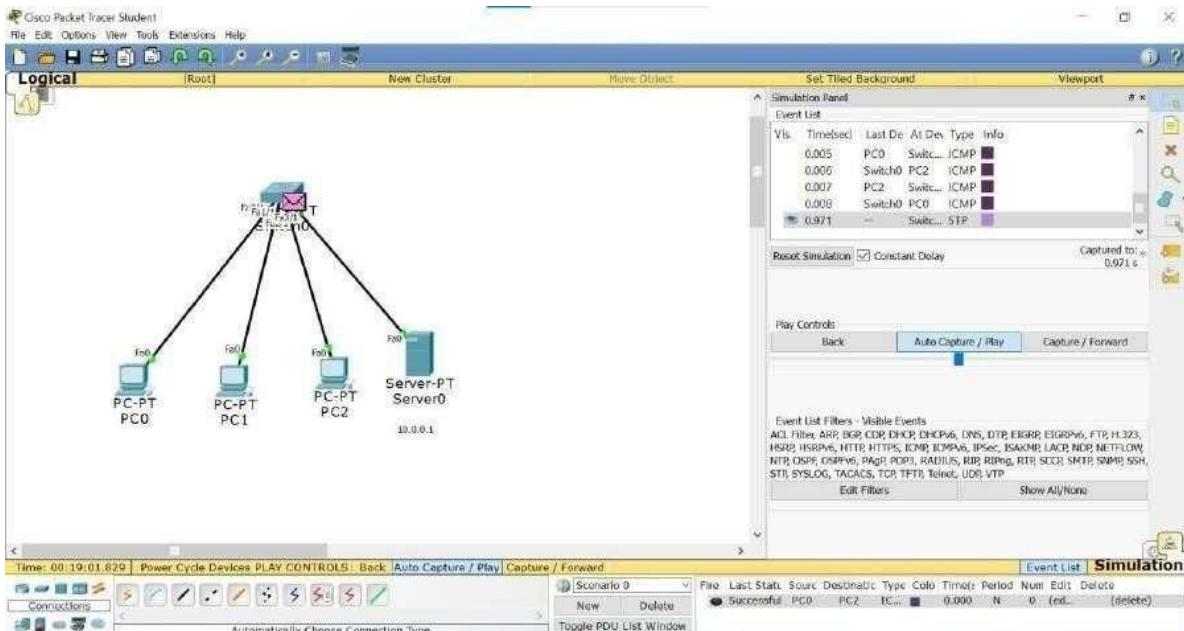
Packet Tracer PC Command Line 1.0
PC>ping 10.0.0.3

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>
```



PROGRAM 4.2:

PC0

Physical Config Desktop Custom Interface

Command Prompt

```
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

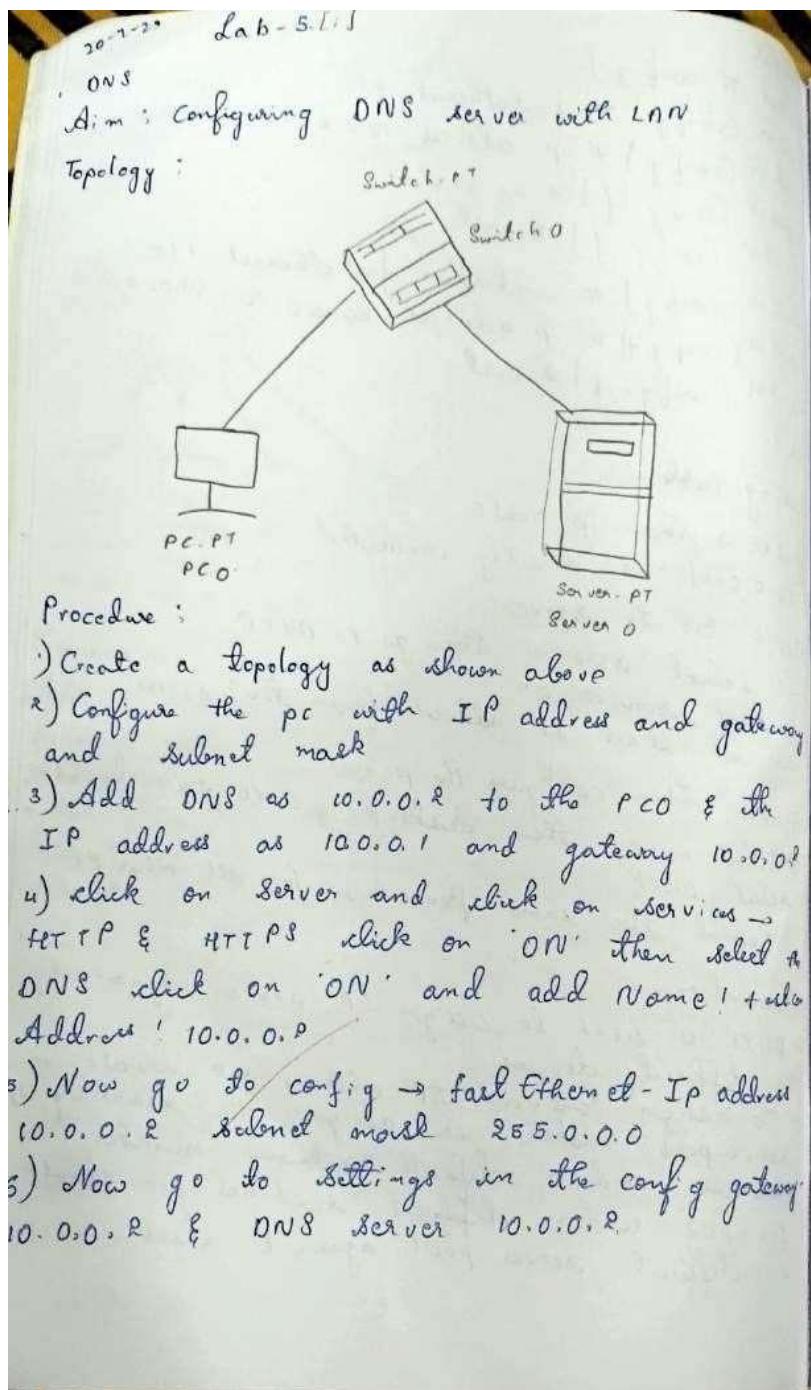
Ping statistics for 20.0.0.3:
    Packets: Sent = 4, Received = 3, Lost = 1 (25% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 0ms, Average = 0ms

PC>
```

LAB 5

Configure Web Server, DNS within a LAN.

OBSERVATION:



7) Now go to pc click on desktop → web browser and type the url.com' in URL and click on go

8) You will see the html video that has written the HTTP button.

Observation:

DNS is the Domain Name System. DNS is linked to the internet and focused on system using Internet protocol. DNS servers are required for working of DNS. The IP address is calculated with the aid of a look up table.

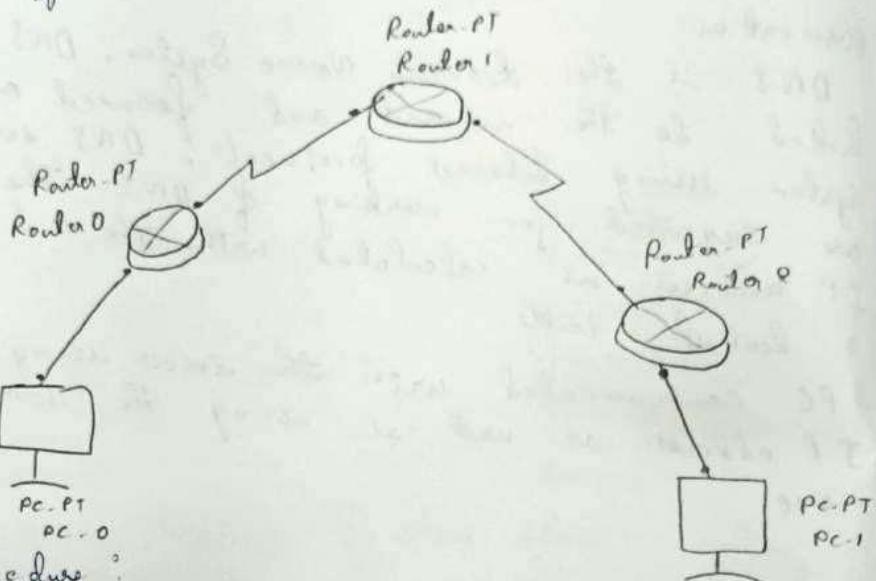
→ PC communicated with the server using the IP address as well as using the domain name.

Lab - 6[i]

RIP Routing Protocol

Aim : Configuring RIP Routing Protocol in Router

Topology :



Procedure :

- a) Three routers and 2 PC's are connected as shown in topology.
- b) Configure the PC's with proper IP address and gateway address.
- c) Similarly, configure the Router's with the proper IP address in CLI mode.
 - N. Enable
 - Config T
 - Interface fastethernet 0/0
 - IP address 10.0.0.1 255.0.0.
 - encapsulation ppp
 - clockrate 6400 0
 - no shut

Note: The encapsulation PPP should be done by all the routers and clock rate 64000 ^{comma} should be only given to the clock symbols sides of the router.

→ for making the routers to know about the other devices, in the previous experiments we used static and the other with dynamic addresses but here we use a routing protocol algorithm that itself makes the router to know other devices.

→ router s:p

→ network 20.0.0.0 of router 2

→ network 30.0.0.0

→ router s:p

→ network 30.0.0.0 of router 3

→ network 40.0.0.0

→ router s:p

→ network 10.0.0.0 of router 1

→ network 20.0.0.0

Ping output:

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:0ms TTL:128

ping statistics from 40.0.0.1

packets sent 4 Received 4 lost=0 (0% loss)

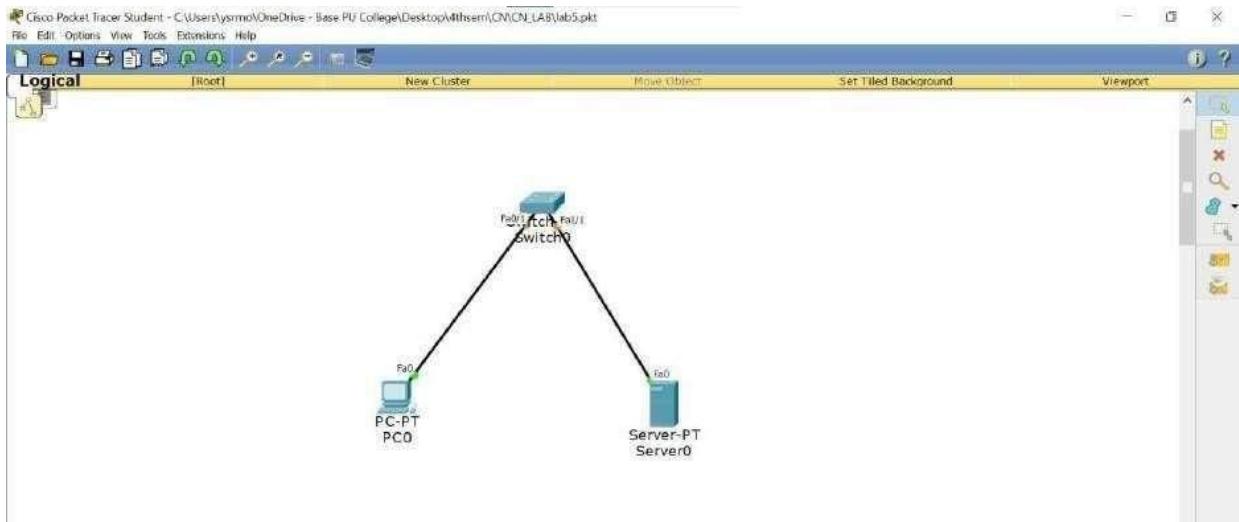
Approximate round trip times in ms

minimum: 0ms, maximum: 0ms Average:0ms

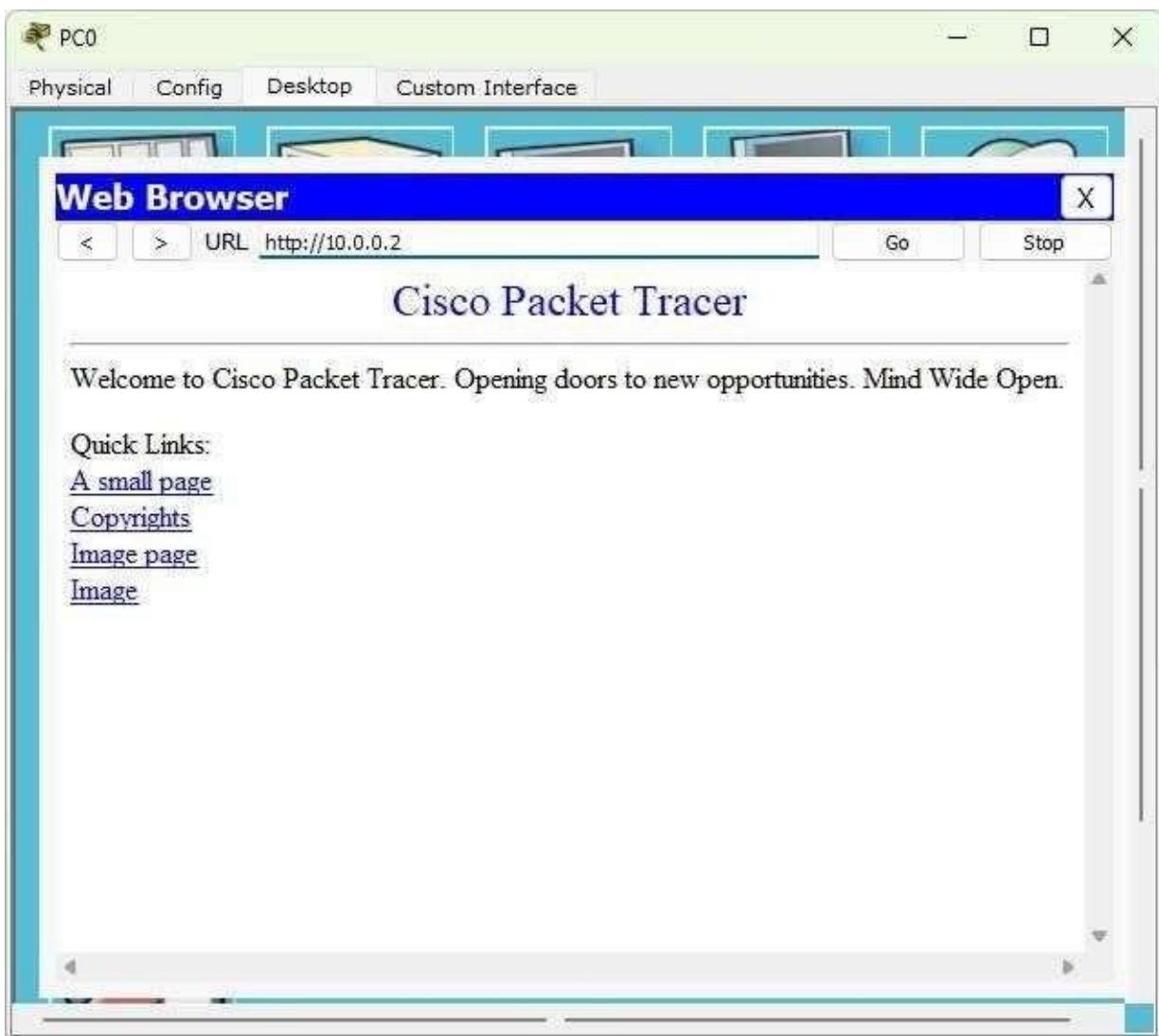
Observation: RIP is the Routing Information Protocol as a distance vector protocol that counts hop count as its primary metric. It is a link state routing protocol that routers should share information about moving traffic among an interconnected graph of LANs.

RIP protocol here used to connect the two routers to one other and PCs using RIP protocol and message is exchanged successfully.

TOPOLOGY:

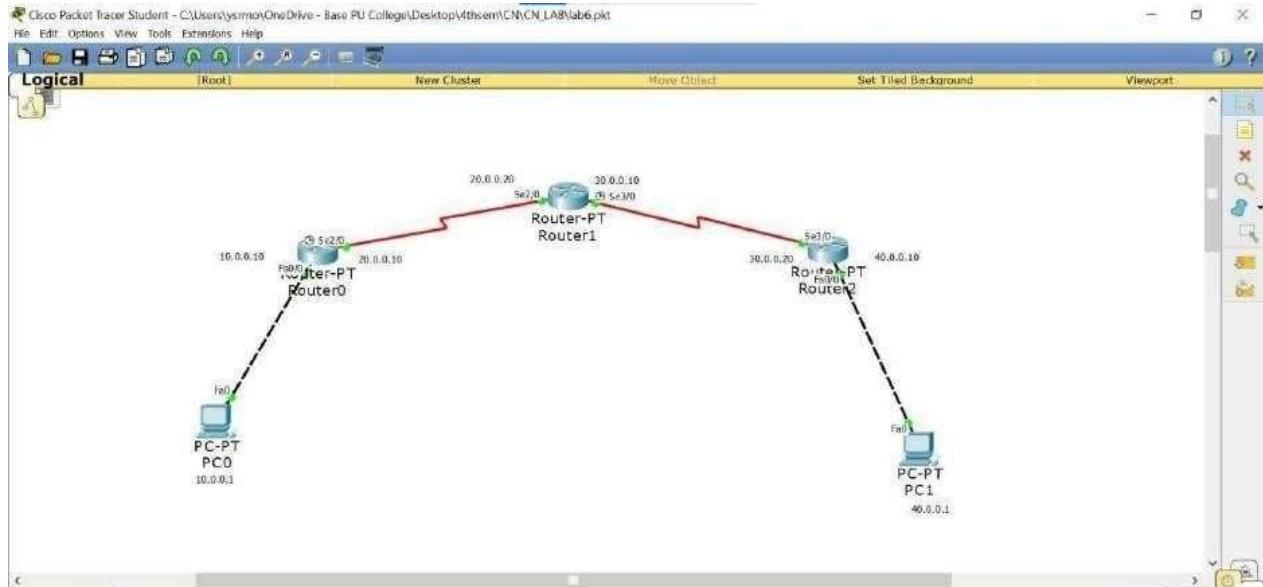


OUTPUT:



TOPOLOGY:

RIP Protocol



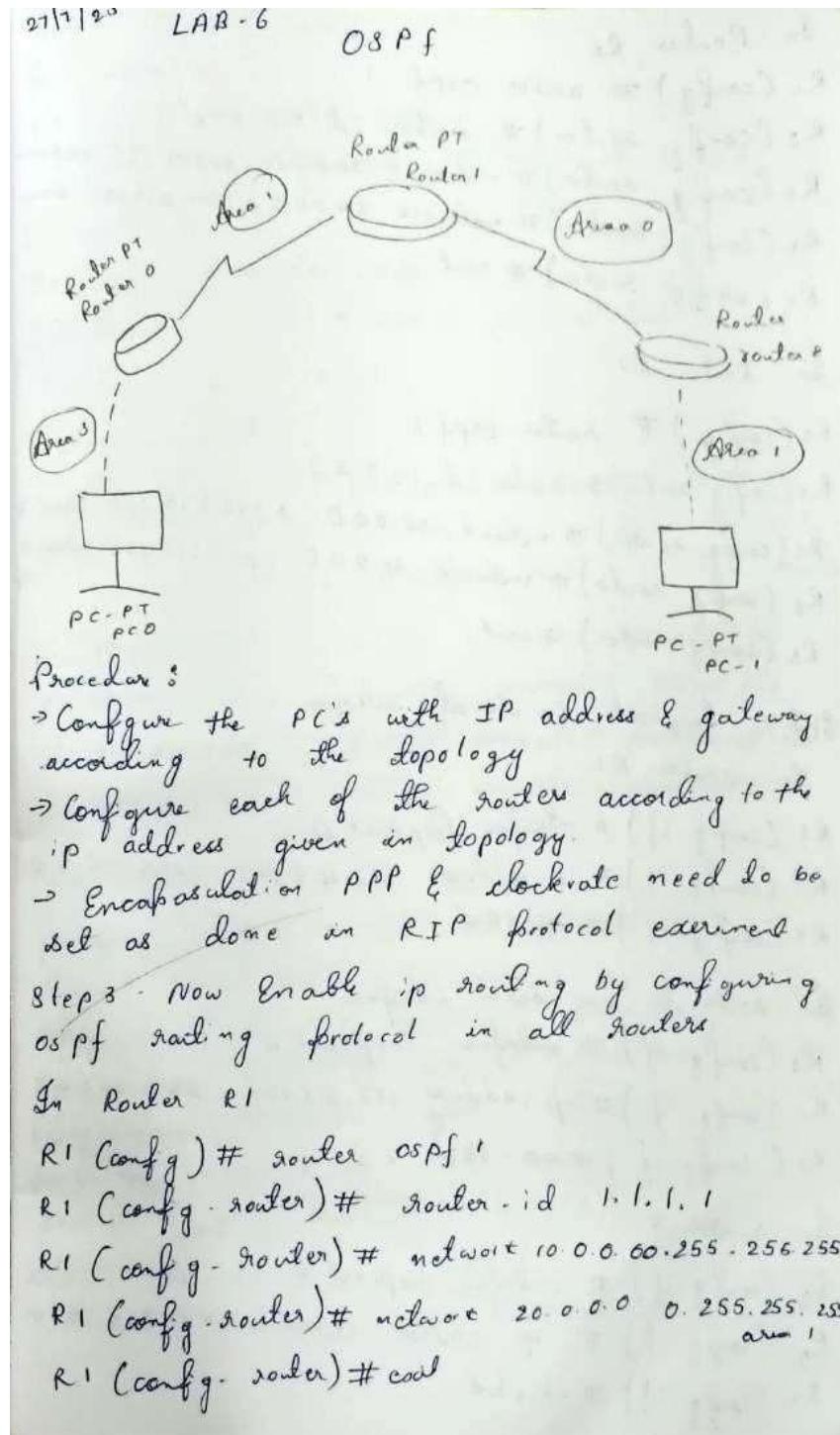
OUTPUT:

```
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=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>
```

Configure OSPF routing protocol.

Lab-6



In Router R₂.

```
R2(config)# router ospf 1  
R2(config-router)# router-id 2.2.2.2  
R2(config-router)# network 200.0.0.0 255.255.255.255  
R2(config-router)# network 300.0.0.0 255.255.255.255  
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  
R3(config-router)# network 40.0.0.0 0.255.255.255  
R3(config-router)# exit
```

Step 4: Loopback in Serial interface

In Router R₁

```
R1(config-if)# interface loopback 0  
R1(config-if)# ip address 172.16.1.252 255.255.0  
R1(config-if)# no shutdown
```

In Router R₂ in Serial interface

```
R2(config-if)# interface loopback 0  
R2(config-if)# ip address 172.16.1.253 255.255.0  
R2(config-if)# no shutdown
```

In Router R₃

```
R3(config-if)# interface loopback 0  
R3(config-if)# ip address 172.16.1.254 255.255.0  
R3(config-if)# no shutdown
```

Step 5 - Virtual link.

In router R1

R1 (config) # router ospf 1

R1 (config-router) # area 1 virtual-link 2 2 2 2

R1 (config-router) # exit

In router R2

R2 (config) # router ospf 1

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

R2 (config-router) # exit

→ show ip route

0 0 0 0 0 0 18 [110/129] via 30.0.0.1 serial 3/0

0 0 0 0 0 0 18 [110/128] via 30.0.0.1 serial 3/0

30.0.0.0 18 is locally subnetted, 2 subnets, 2 masks

c 30.0.0.0 18 is directly connected, serial 3/0

c 30.0.0.1/32 is directly connected, serial 3/0

c 40.0.0.0 18 is directly connected fastethernet 0/0

c 172.16.0.0 16 is directly connected, loopback

Ping output

pinging 40.0.0.10 with 82 bytes of data

Request timed out

Reply from 40.0.0.10 bytes=82 time=2ms TTL=125

Reply from 40.0.0.10 bytes=82 time=9ms TTL=125

Reply from 40.0.0.10 bytes=82 time=10ms TTL=125

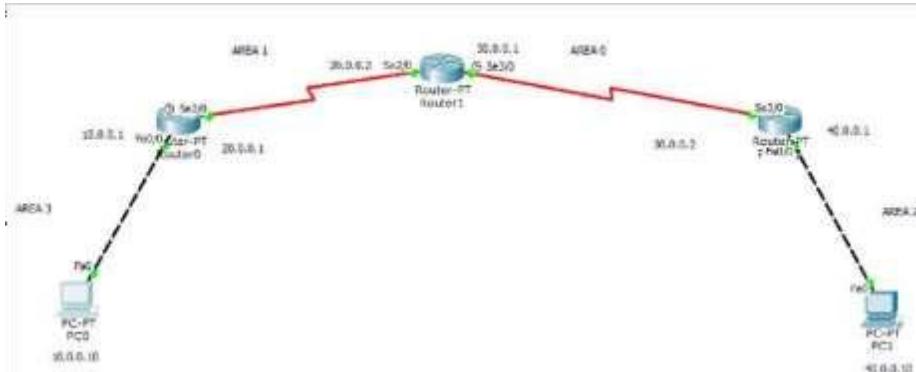
pinging statistics for 40.0.0.10

packets: sent: 4, received: 3, lost: 1

Approx round trip in ms

min: 2ms, max: 10ms, Average: 7ms

TOPOLOGY:



OUTPUT:

PC> PC0

Physical Config Desktop Custom Interface

Command Prompt X

```
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

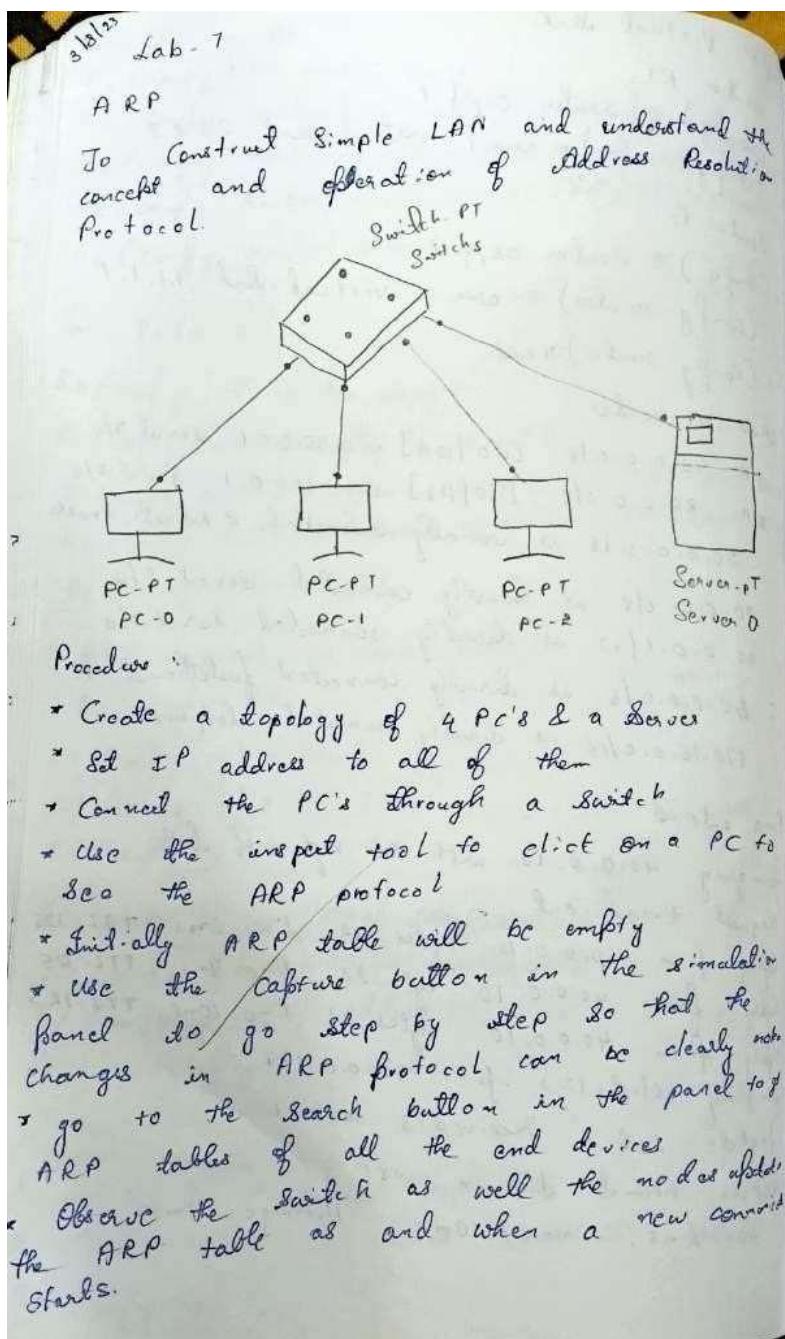
Pinging 40.0.0.10 with 32 bytes of data:

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>
```

LAB 7

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



Ping output

PC > arp -a

PC > ping 10.0.0.2

PC > ping 10.0.0.03

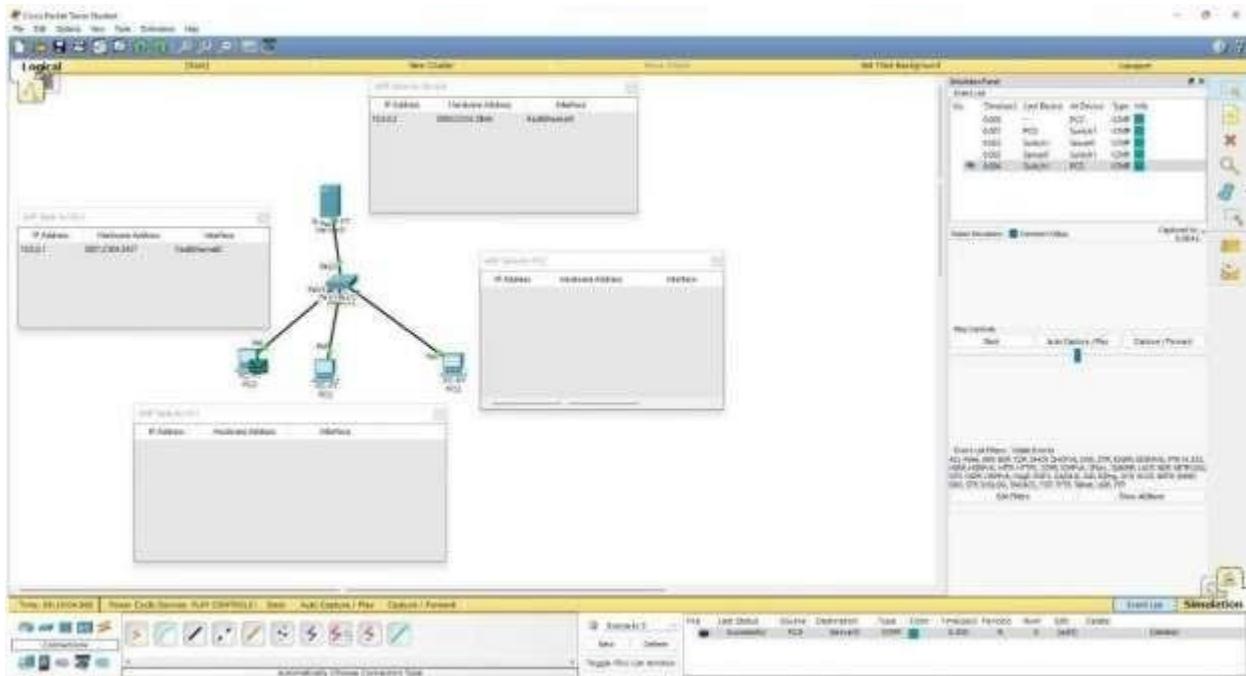
PC > ping 10.0.0.4.

Internal address	Physical address	Type
10.0.0.2	00cd.bd15.8a18	dynamic
10.0.0.3	0000.0657.b9e8	dynamic
10.0.0.4	00d0.bC57.aa38	dynamic

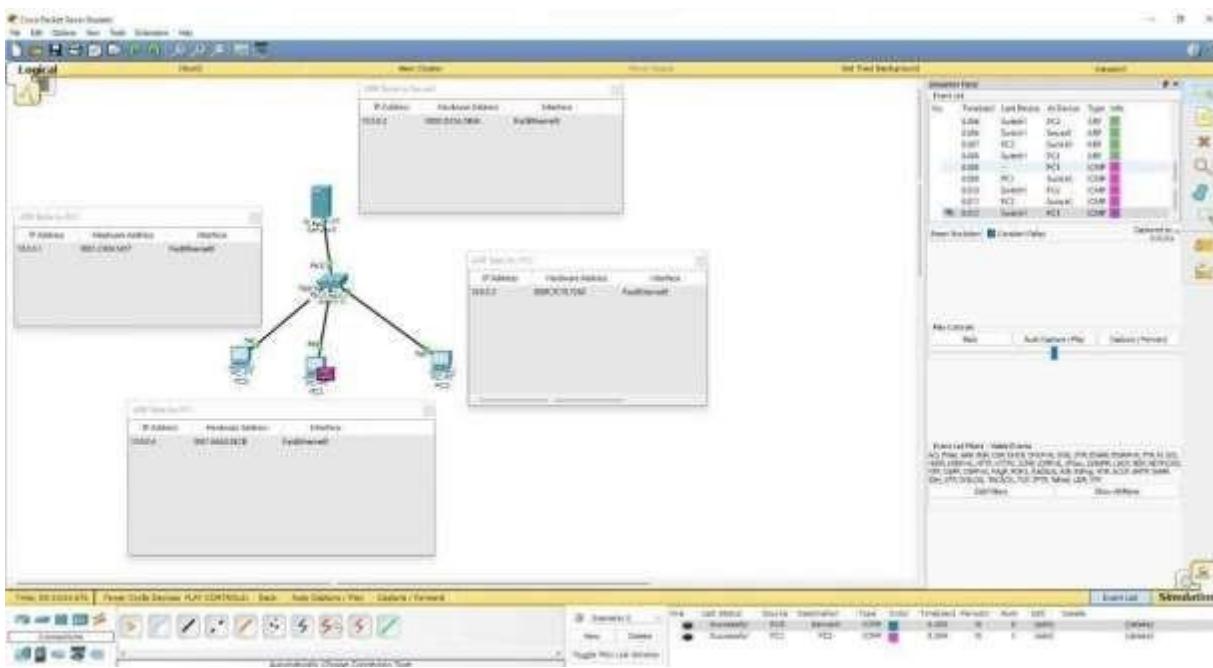
Observation

We observe the changes when we do every time when we do in all the PC's ARP table will have values empty at the starting & values will be formed whenever we ping each pc in the ping

TOPOLOGY:



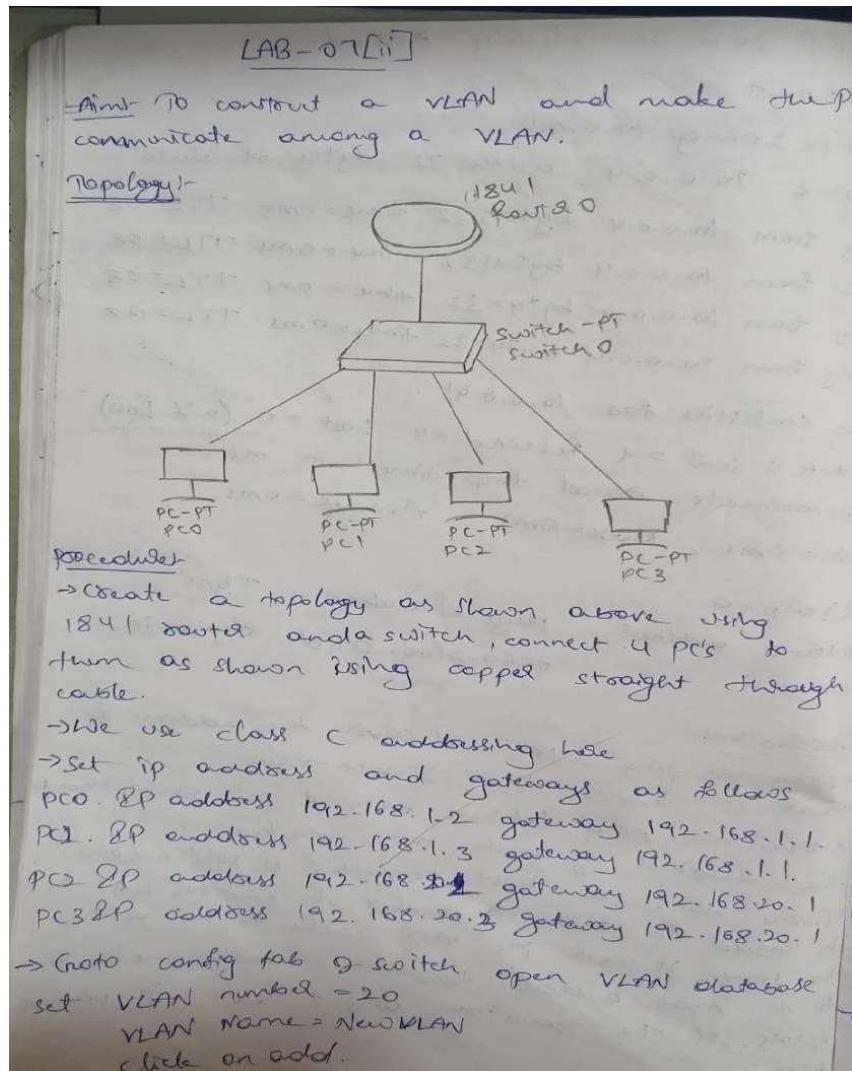
OUTPUT:



LAB 8

To construct a VLAN and make a pc communicate among VLAN

Observation:



- In switch go to fastethernet 0/0 and connect it to router and configure it.
Select Trunk and choose 20, NewVLAN.
→ For fast0/3 and port 0/1 in select 10 : NewVLAN
and keep access as it is.
→ Open config mode in router, goto VLAN database Add VLAN no. 10

→ In router, go to CLI mode.

for 0/0

Rout# ip address 192.168.1.1 255.255.255.0

Rout# no shut.

Rout# interface fastethernet 0/0.1

Rout# encapsulation dot1q 20

Rout# ip address 192.168.20.1 255.255.255.0

Rout# no shut

Rout# exit

Ping output

PC> ping 192.168.20.2

Pinging 192.168.20.2 with 32 bytes of data.

Request timed out.

Reply from 192.168.20.2 bytes=32 time=0ms TTL=127

Reply from 192.168.20.2 bytes=32 time=2ms TTL=127

Reply from 192.168.20.2 bytes=32 time=1ms TTL=127

Ping statistics for 192.168.20.2

Packets sent=4 Received=3 Lost=1 (25% loss)

Approximate round trip in ms:

min=0ms Max=2ms Average=0ms

Observation

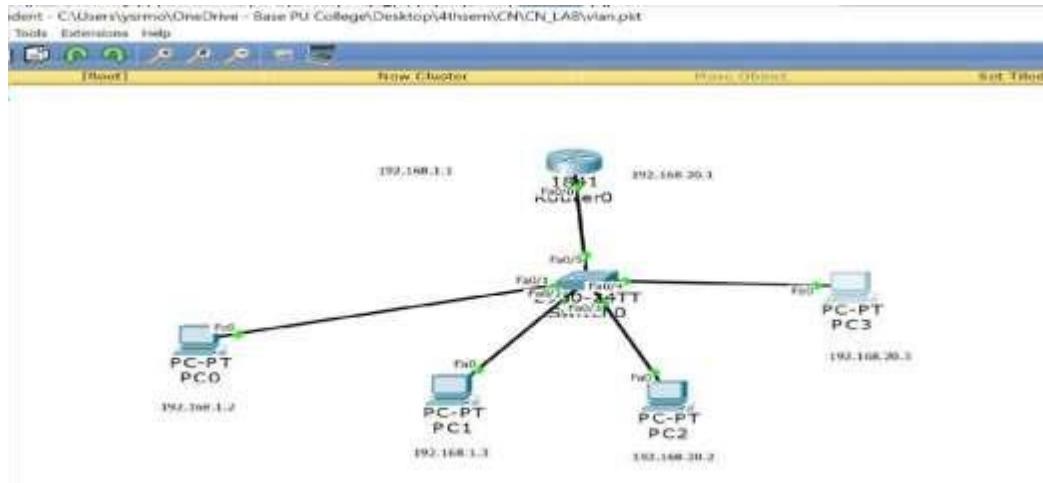
→ We can observe that after VLAN is configured
we can successfully ping pc2 (192.168.20.2)

from pc0 (192.168.1.2)

• pc2 and pc3 are grouped together and
communication among them is done via VLAN.

• 192.168.20.1 is a sub interface o/p. I created

TOPOLOGY:



OUTPUT:

PC0

Physical Config Desktop Custom Interface

Command Prompt

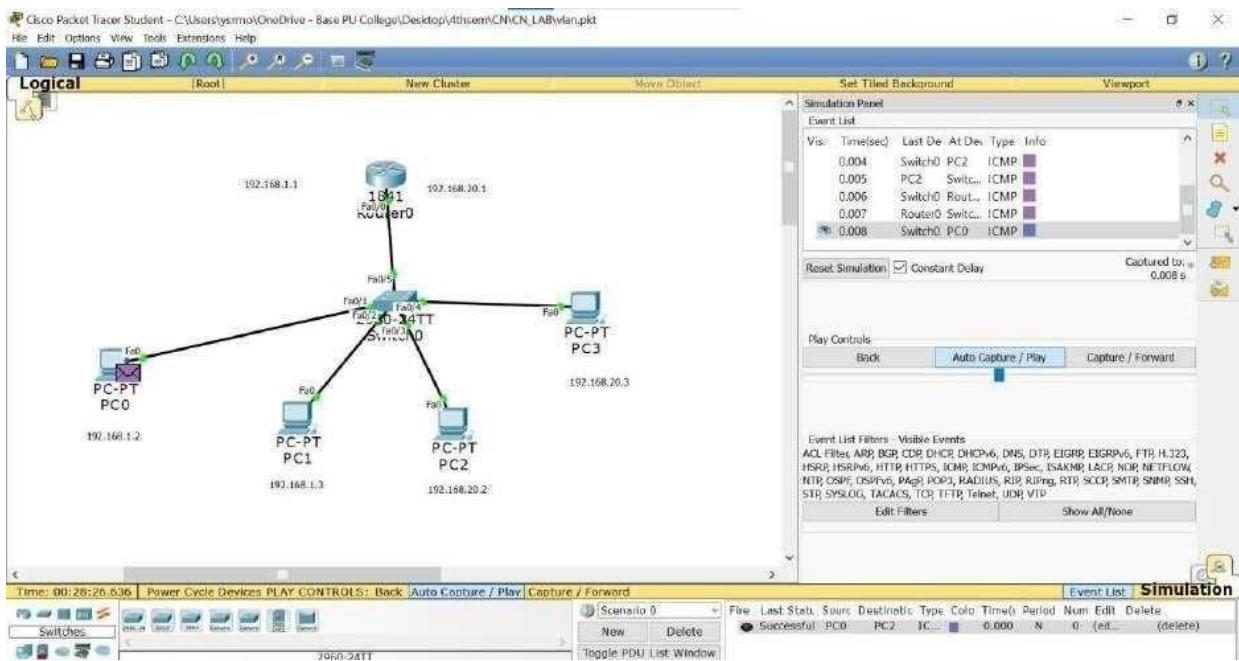
```
Packet Tracer PC Command Line 1.0
PC>ping 192.168.20.3

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

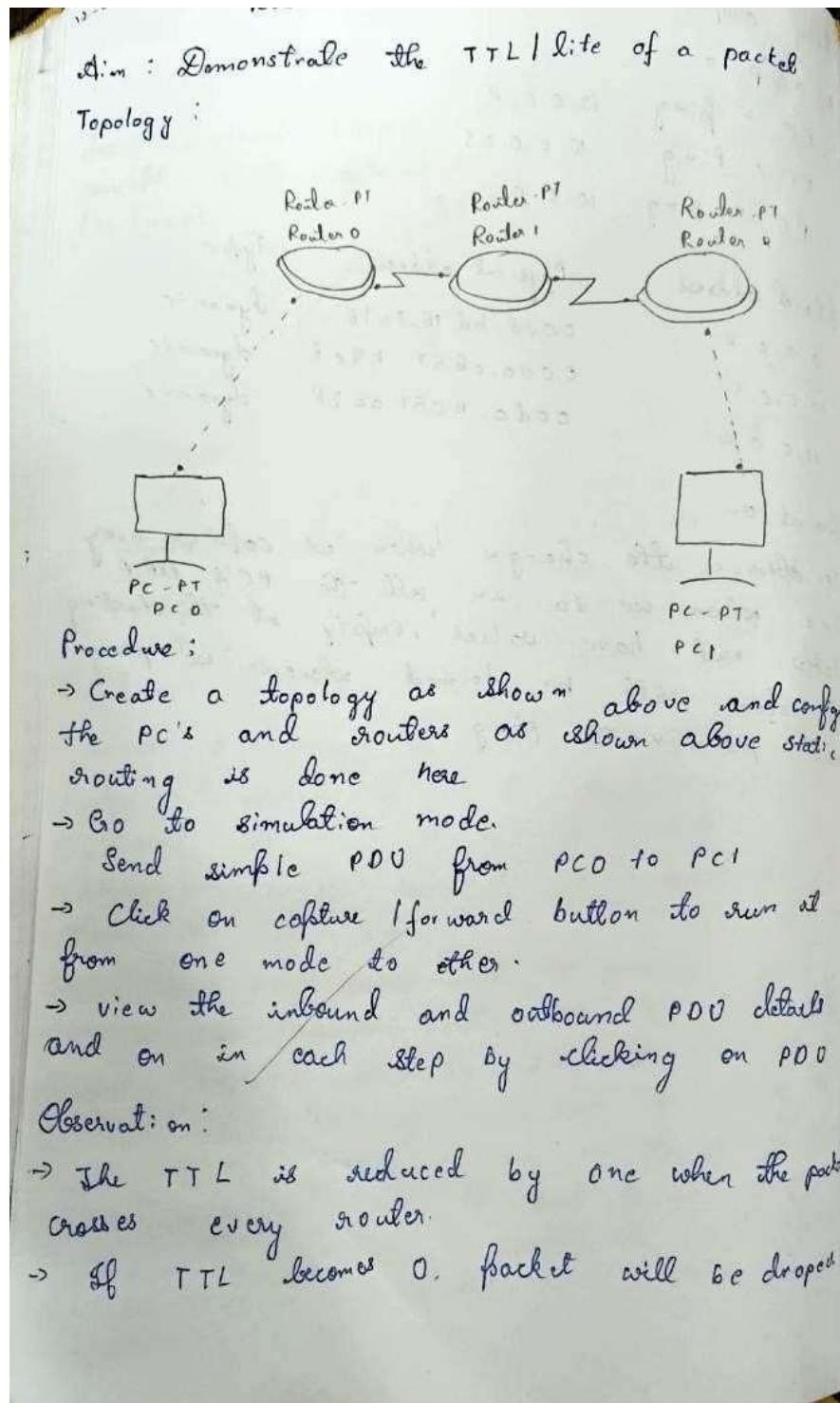
Ping statistics for 192.168.20.3:
    Packets: Sent = 4, Received = 3, Lost = 1 (25% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 0ms, Maximum = 5ms, Average = 1ms

PC>
```



LAB 9

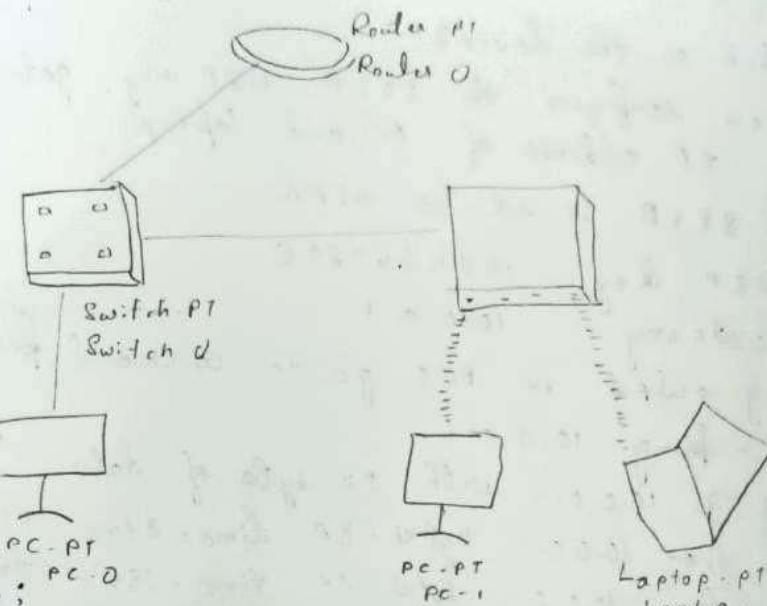
Demonstrate the TTL/ Life of a Packet



Lab - 08 [ii]

Ques : To construct a WLAN and make the nodes communicate wirelessly

Topology :



Procedure :

- Create the topology as shown above.
- Configure the PCs as normally we do config
PC 3
- IP address : 0.0.0.2
- Configure router O
 - Set the IP address as 10.0.0.1 for fa 0/0
- Configure the access point PT in config of
port 1.1
- 8310 - WLAN

Select WEP and give any 10-digit number
as password 1234567890

- To configure PC 4 and Laptop with wireless
standard
- Switch off the device

- * Drag the existing P.T. MOB1-NM-1-NM to do place it on its mentioned name.
- * Drag the WMP300N wireless interface to expand port switch on the device.
- * Now configure the SSID, WEP key, gateway and IP address of PC and laptop.

The SSID is set to WLAN.

WEP key : 1234567890

Gateway : 10.0.0.1

Ping output in PC0 go to command prompt

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=24ms TTL=11

Reply from 10.0.0.4 bytes=32 time=15ms TTL=11

Reply from 10.0.0.4 bytes=32 time=5ms TTL=11

Reply from 10.0.0.4 bytes=32 time=18ms TTL=11

Reply from 10.0.0.4 bytes=32 time=18ms TTL=11

- Ping statistics for 10.0.0.4

- Packets: Sent = 4, Received = 4 Lost = 0

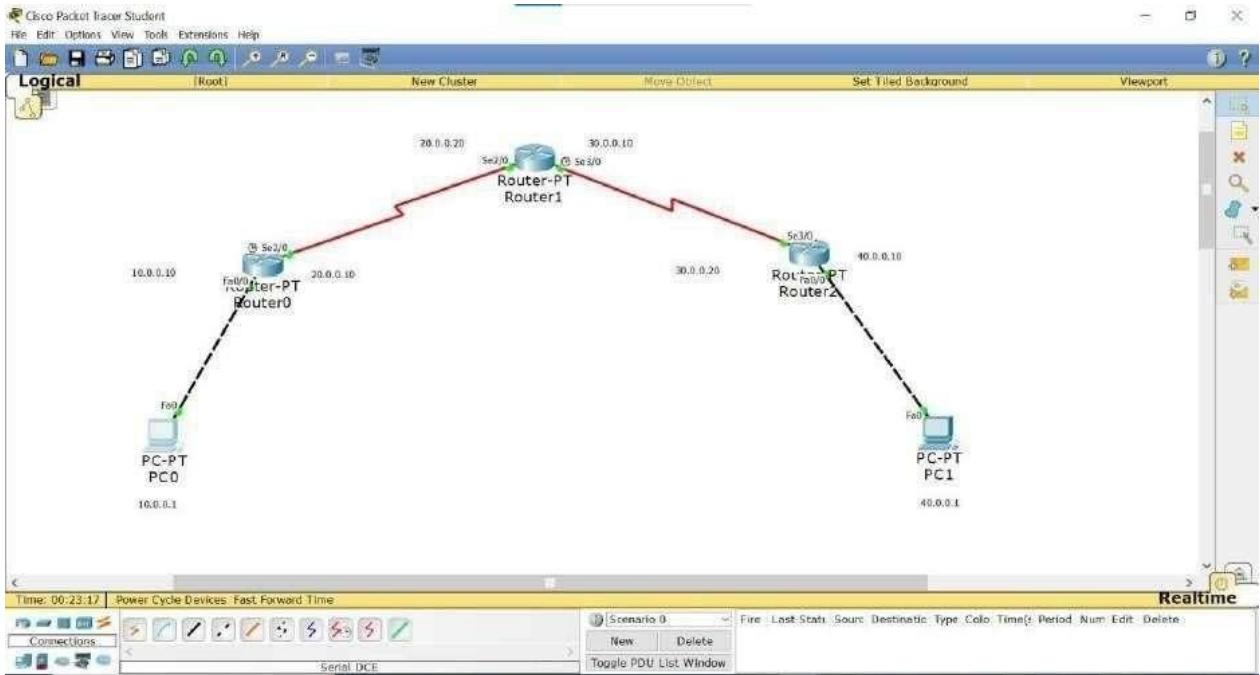
- Approximate round trip time in ms

- min = 5ms max = 24ms average = 15

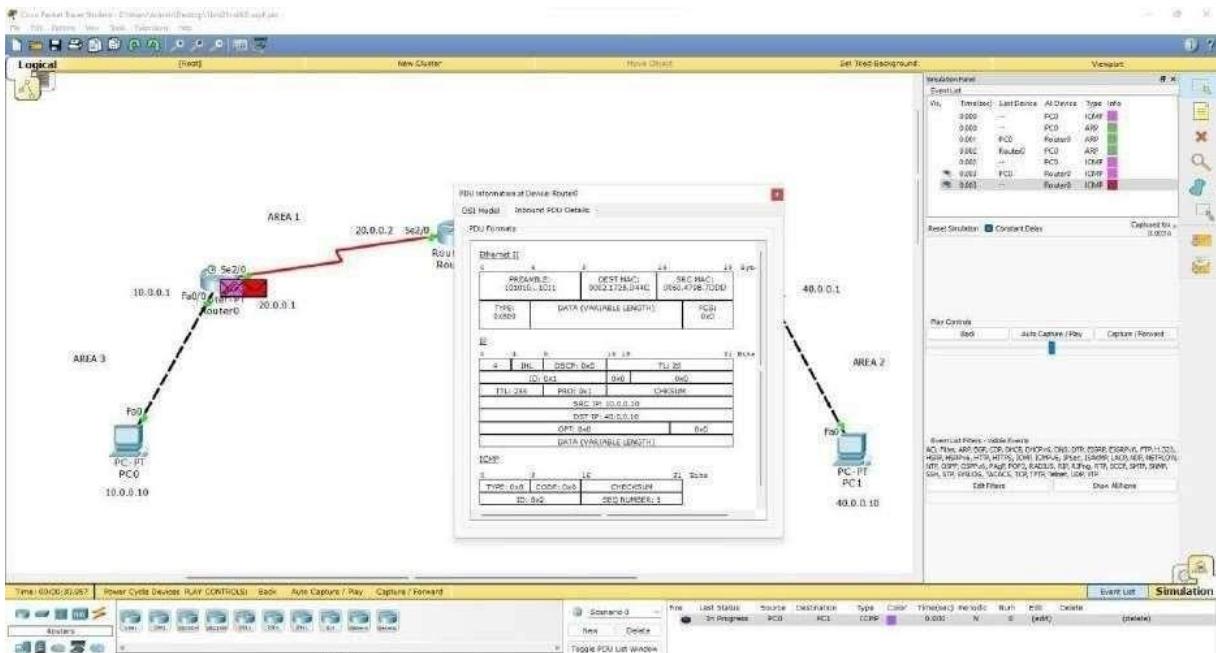
- Observation : We can ping each and every node in the network. So we can observe that wireless connection is done successfully.

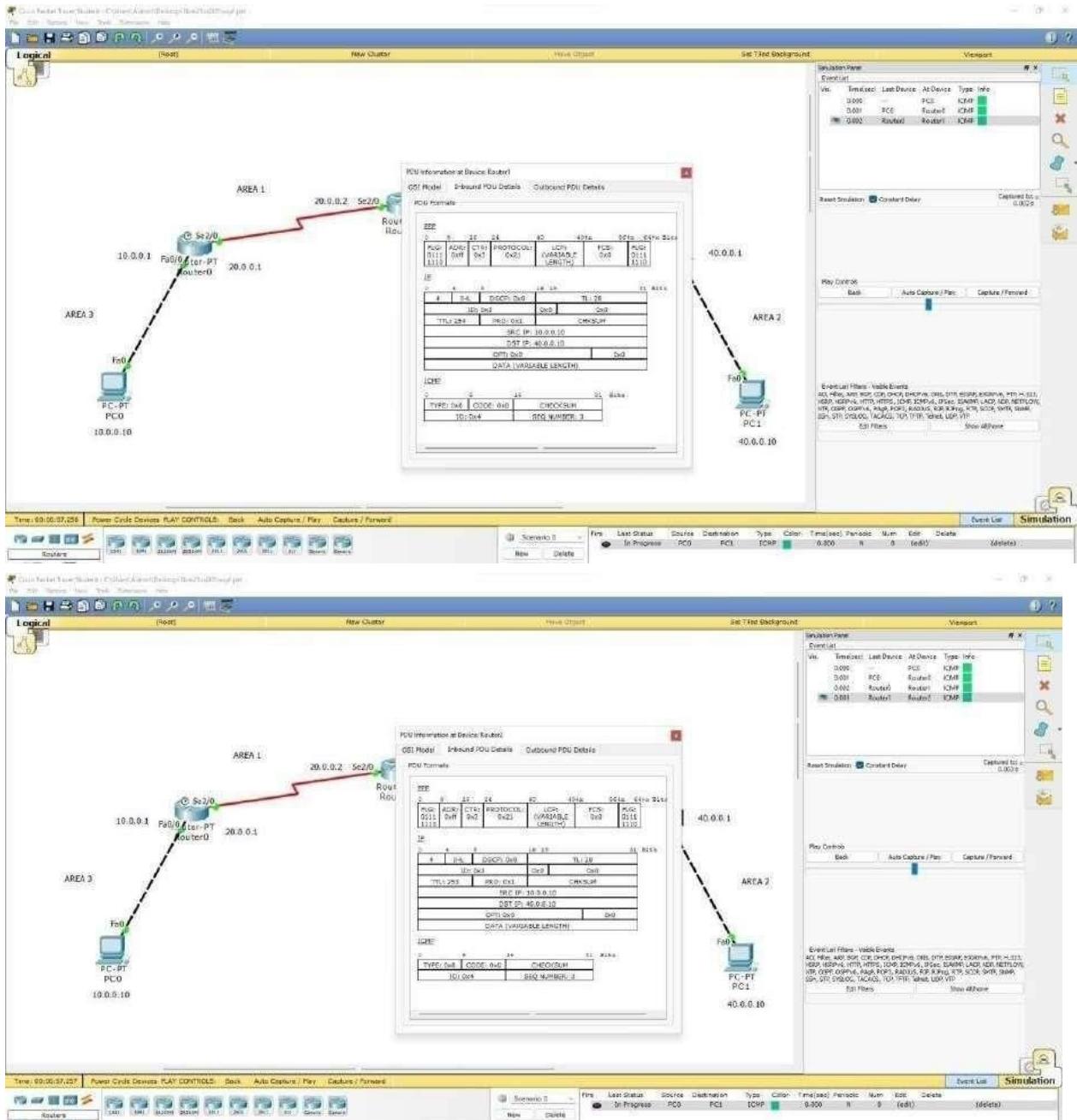
→ When connection is established there is wireless connection between two devices. connecting access points and end devices.

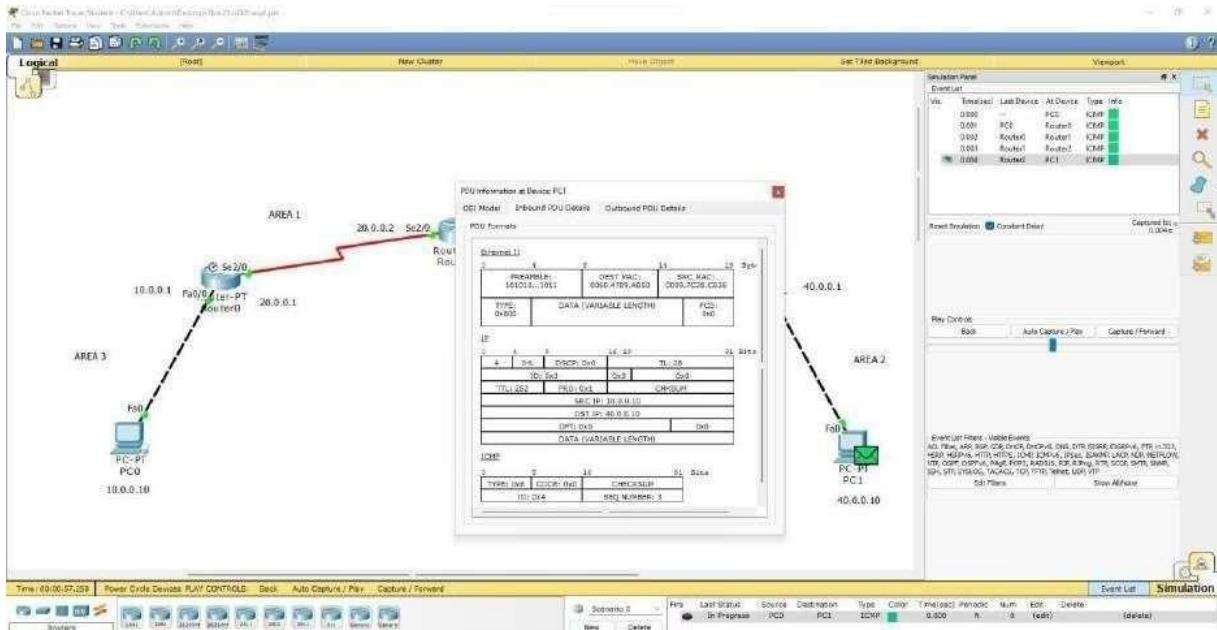
TOPOLOGY:



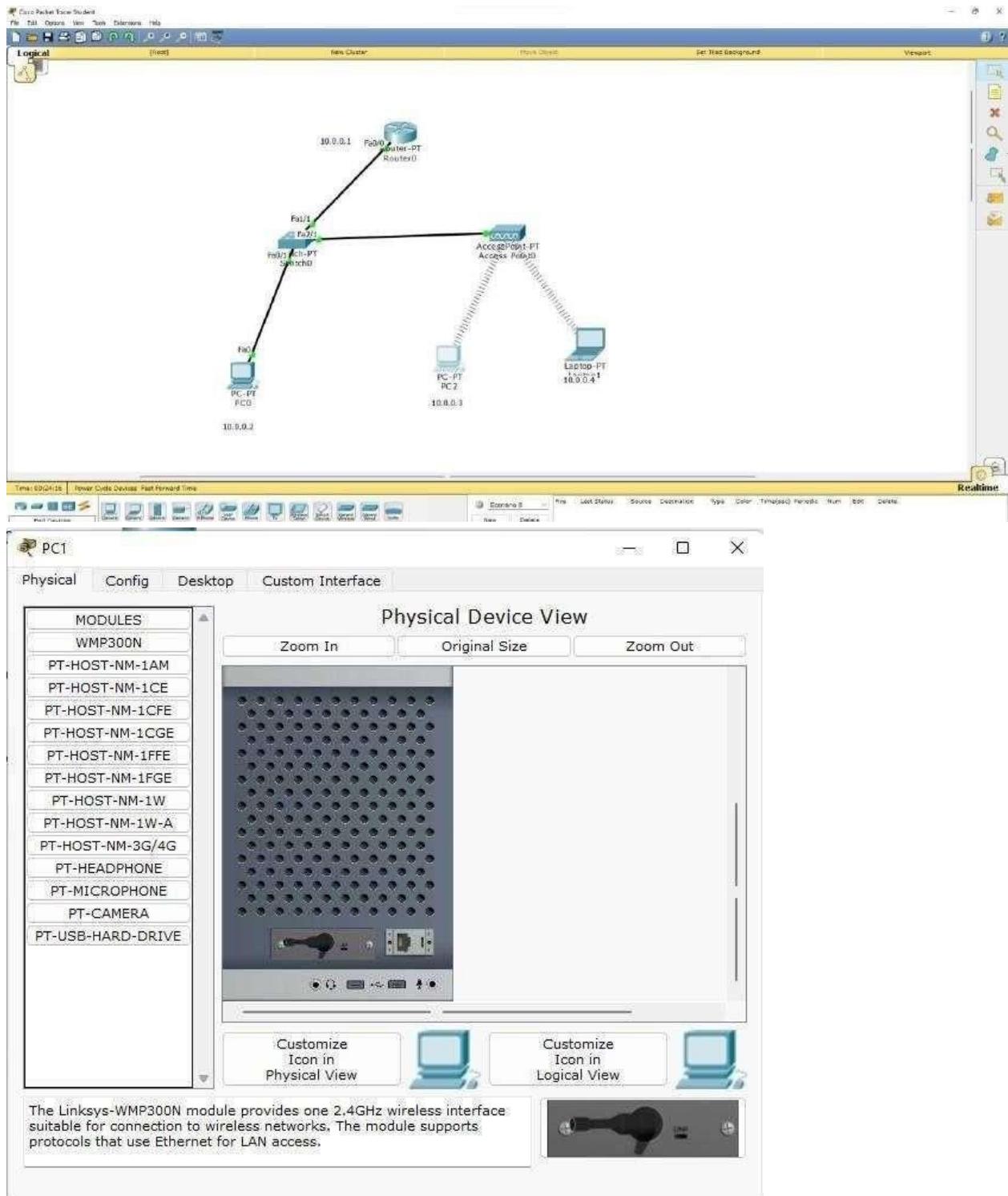
OUTPUT:







TOPOLOGY: WLAN





OUTPUT:

```

PC0
Physical Config Desktop Custom Interface

Command Prompt
X

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

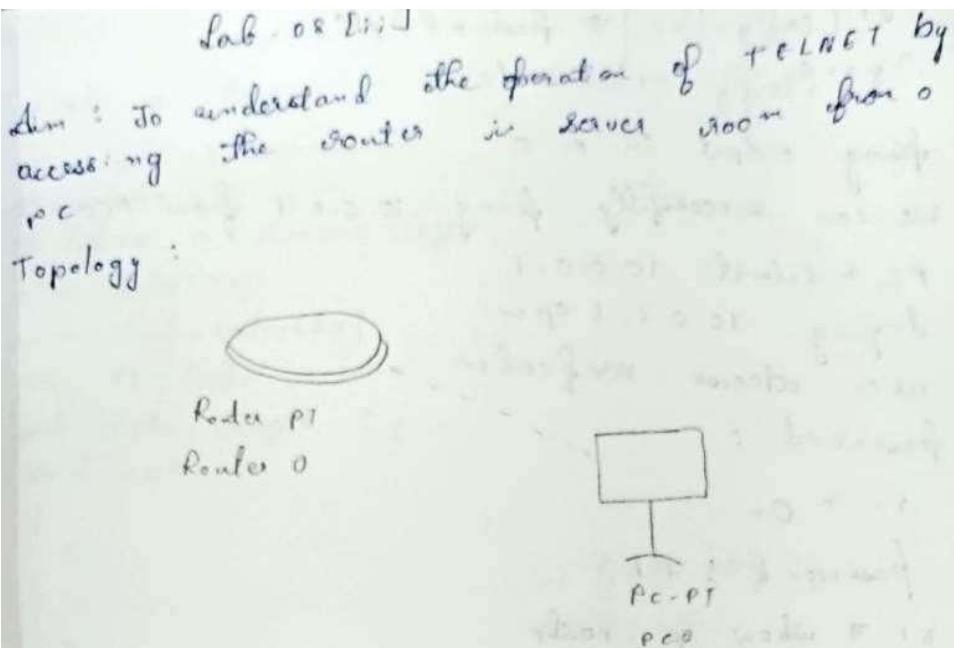
Pinging 10.0.0.3 with 32 bytes of data:

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>

```

To understand the operation of TELNET by accessing the router in server room from a PC in IT office. **Lab-10**



Procedure :

- Create the topology as shown above.
- Connect the devices using copper cross-over
- Configure the PC
 - IP address : 10.0.0.2
 - Gateway : 10.0.0.1
- Go to CLI mode in Router O

Router > on

```
Router# conf t
Router(Config)# hostname r1
r1(Config)# enable secret pass1.
r1(Config)# interface fa 0/0
r1(Config)# ip address 10.0.0.1 255.0.0.0
r1(Config-if)# no shut
r1(Config-if)# line vty 0-5
r1(Config-if)# login.
```

`R1 (config-line) # password po
R1 (config-line) # exit`

pinging output in PC 0

We can successfully ping 10.0.0.1 from PC0

PC > telnet 10.0.0.1

dryping 10.0.0.1 open

User access verification

password :

`R1 > On`

password : po

`R1 # show ip route`

c 10.0.0.0/8 is directly connected. fa 0/0

Observations:

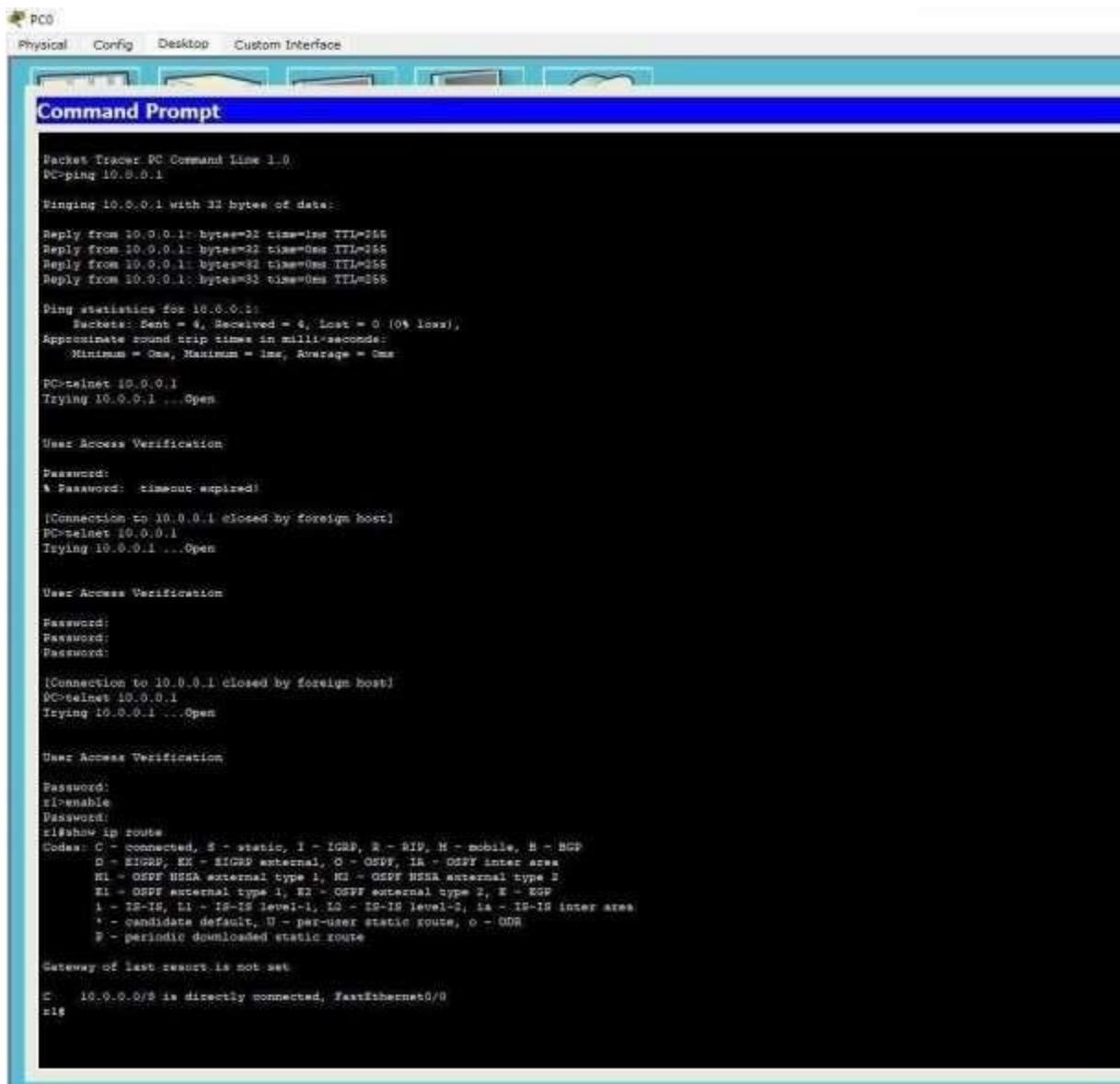
→ We can observe that the admin in PC is run commands as run in Router CLI and the result from the PC

→ So with the help of TELNET, we can access the Router in server room from a PC.

TOPOLOGY:



OUTPUT:



The screenshot shows a software interface titled "PCO" with tabs for "Physical", "Config", "Desktop", and "Custom Interface". A "Command Prompt" window is open, displaying the following terminal session:

```
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:
*Password
Password:
*Password

[Connection to 10.0.0.1 closed by foreign host]
PC>show ip route
Codes: C - connected, S - static, R - RIP, M - mobile, B - BGP
      D - EIGRP, EX - EIGRP external, O - OSPF, I - OSPF inter area
      E1 - OSPF NSSA external type 1, E2 - OSPF NSSA external type 2
      E - OSPF external type 3, E0 - OSPF NSSA external type 3
      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

S* 10.0.0.0/8 is directly connected, FastEthernet0/0
c1g
```

LAB 13 Program 1

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

CODE:

```
#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); crc();
for(i=0;(i<N-1) && (check_value[i]!='1');i++){

    if(i<N-1)
        printf("\nError detected\n\n");
    else
        printf("\nNo error 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];
    check_value[j]=data[i++];
}
```

```

}while(i<=data_length+N-1);
}

int main()
{ printf("\nEnter data to be transmitted: ");
scanf("%s",data); printf("\n Enter the
Generating polynomial: ");
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("\nCRC or Check 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 data to be sent : %s",data);
receiver();
return 0;
}

```

OUTPUT:

```

Enter data to be transmitted: 10001000000100001
Enter the Generating polynomial: 1011
Data padded with n-1 zeros : 10001000000100001000
CRC or Check value is : 100
Final data to be sent : 10001000000100001100
Enter the received data: 10001000000100001100
Data received: 10001000000100001100
No error detected

```

```

Enter data to be transmitted: 10001000000100001
Enter the Generating polynomial: 1011
Data padded with n-1 zeros : 10001000000100001000
CRC or Check value is : 100
Final data to be sent : 10001000000100001100
Enter the received data: 10010000000100001100
Data received: 10010000000100001100
Error detected

```

Program 2

Write a program for congestion control using Leaky bucket algorithm.

CODE:

```

#include<stdio.h>
void main()
{ int b_size,d_rate,in_d_rate,rem_b_size;
printf("Enter the bucket size:\n");
scanf("%d",&b_size);
rem_b_size=b_size; printf("Enter the
outgoing data rate:\n");
scanf("%d",&d_rate); while(1) {
printf("Enter the size of incoming packet\n");
scanf("%d",&in_d_rate);
if(in_d_rate<=b_size)
{ if(in_d_rate<=rem_b_size)
{ rem_b_size=rem_b_size-in_d_rate;
rem_b_size=rem_b_size+d_rate; printf("Data packet is
accepted\n"); printf("Remaining space in bucket is.....
%d\n",rem_b_size); printf("\n");
} else{ printf("Data packet is dropped because the bucket size is less than
the packet
size\n");
printf("\n");
}
}
}

```

}

OUTPUT:

```
Enter the bucket size:  
5000  
Enter the outgoing data rate:  
200  
Enter the size of incoming packet  
3000  
Data packet is accepted  
Remaining space in bucket is.... 2200  
  
Enter the size of incoming packet  
2500  
Data packet is dropped because the bucket size is less than the packet size  
  
Enter the size of incoming packet
```

Lab - q 6

Write a program for error detecting code using CRC

```

#include <stdio.h>
#include <string.h>
#define N strlen(Poly)
char data[30];
char check_value[30];
char Poly[10];
int data_length, i, j;
void xor
{
    for (i = 0; j < N; j++)
        check_value[j] = Poly[j] ^ data[i];
}
void receiver()
{
    printf("Enter the received data : ");
    scanf("%s", data);
    printf("Data received : %s", data);
    crc();
    for (i = 0; i < N - 1 && (check_value[i] == '1'); i++)
        if (N - 1)
            printf("Error detected");
        else
            printf("No error detected");
}
void crc()
{
    for (i = 0; i < N; i++)
        check_value[i] = data[i];
    do {

```

```

if (check_value[i] == '1')
    XOR();
for (j = 0; j < N-1; j++)
    check_value[j] = check_value[j+1];
    check_value[j] = data[j+1];
} while (i < data.length+N-1);

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; i++)
        data[i] = 0;
    printf ("Data padded with %u zeros : %s", data_length-N, data);
    crc();
    printf ("CRC value is %s", check_value);
    for (i = data_length; i < data_length+N-1; i++)
        data[i] = check_value[i-data_length];
    printf ("final datword to be sent : %s", data);
    receiver();
    return 0;
}

```

Output :

Enter data to be transmitted : 101010

Enter the divisor polynomial : 1011

Data Padded with zeros : 101010000

CRC value is : 001

Final code word to be sent : 101010001

Enter the received data : 100010000

Error detected.

✓
AJ
28/12/23

Lab. 09 [ii]

Write a program for Congestion control using
Leaky Bucket algorithm

```
#include <stdio.h>
int main()
{
    int incoming, outgoing, buck_size, n, store = 0;
    printf ("Enter the bucket size : ");
    scanf ("%d", &outgoing);
    printf ("Enter the no of inflows : ");
    scanf ("%d", &n);
    while (n != 0)
    {
        printf ("Enter the incoming bucket size : ");
        scanf ("%d", &incoming);
        if (incoming <= (buck_size, store))
        {
            store = incoming;
            printf ("Bucket Buffer size %d. d. out of %d\n"
                   "bucket size");
        }
        else
        {
            printf ("Dropped %d no of packets 1m");
            incoming = (buck_size - store);
            printf ("Bucket Buffer Size %d of %d \n. store"
                   "buck_size");
            store = buck_size;
        }
    }
}
```

Store : store.outgoing ;
print ("After outgoing $\frac{1}{2}$ d packets left out of
 $\frac{1}{2}$ d in buffer in , store, buck size);

8

8

Output :

Enter bucket size : 5000

Enter outgoing rate : 2000

Enter number of packets in : 8

Enter the incoming packet size : 3000

Enter the incoming packet size : 3000

Bucket Buffer size 3000 out of 5000

After outgoing 1000 packets left out of 5000 in

Buffer

Enter the incoming packet size : 1000

Bucket Buffer size 2000 out of 5000

Bucket Buffer size 2000 out of 5000

After outgoing 0 packets left out of 5000 in

Buffer.

NP
2/9/2023

LAB 14 Program 1

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.

Observation:

22/8/23 LAB-14

Q1] Using TCP/IP Sockets, write a client server program to make client sending the file name & the server to send back the contents of the requested file if present.

Soln,
ClientTCP.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(6524).decode()
print('From 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 True:
    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(l.encode())
    print('Insert contents of ' + sentence)
    file.close()
    connectionSocket.close()
```

Output

Server Side

The server is ready to receive

Client Side

Edit file name: servetCP.py

The contents of file servetTCP is displayed
here.

Server Side-

Sent contents of servetTCP.py.

SOLUTION:

```
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:

Client:

```
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/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,"r")
    l = file.read(1024)
    connectionSocket.send(l.encode())
    print('\nsent contents of'+sentence)
    file.close()
    connectionSocket.close()
```

```
>>>
= RESTART: C:/Users/Admin/AppData/Local/Programs/Python/Python310/clientTCP.py =
Enter file name:aab.py

From Server:

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.

class Node:
    def __init__(self,data):
        self.data=data
        self.left=None
        self.right=None
        self.height=1

class AVL_Tree:
    def getHeight(self,root):
        if not root:
            return 0
        return root.height

    def getBalance(self,root):
        if not root:
            return 0
        return self.getHeight(root.left)-self.getHeight(root.right)

    def rightRotate(self,z):
        y=z.left
        T3=y.right

        y.right=z
        z.left=T3

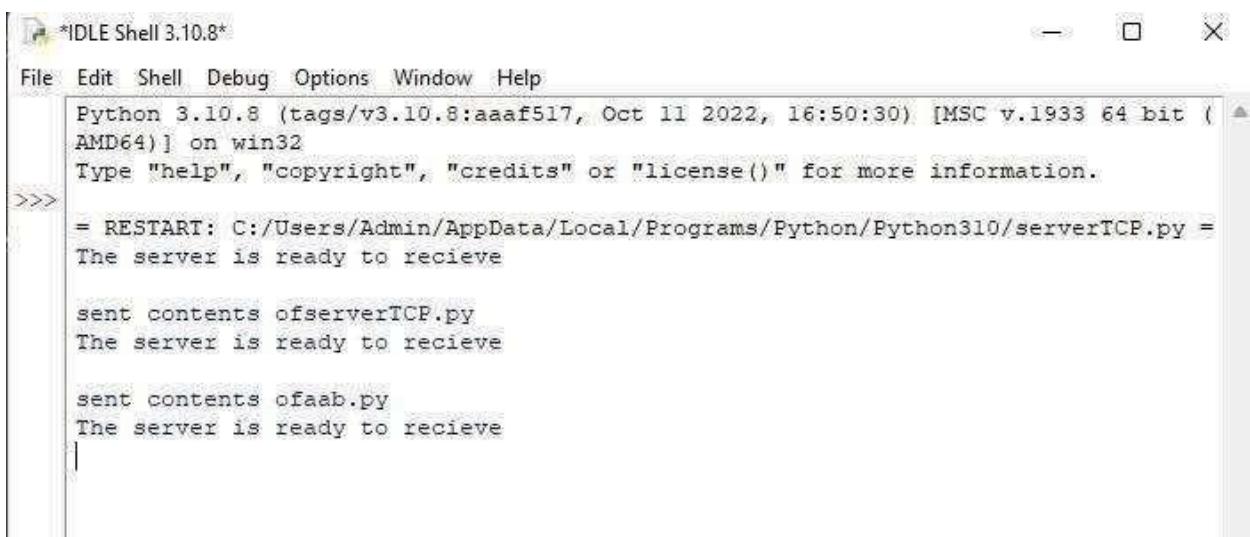
        z.height=1+max(self.getHeight(z.left),self.getHeight(z.right))
        y.height=1+max(self.getHeight(y.left),self.getHeight(y.right))

        return y

    def insert(self,root,data):
        if not root:
            return Node(data)
        if data < root.data:
            root.left=self.insert(root.left,data)
        else:
            root.right=self.insert(root.right,data)

        root.height=1+max(self.getHeight(root.left),self.getHeight(root.right))
```

Server: >>>



The screenshot shows a window titled "IDLE Shell 3.10.8*". The menu bar includes File, Edit, Shell, Debug, Options, Window, and Help. The main area displays the following Python session:

```
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/serverTCP.py =
The server is ready to receive

sent contents of serverTCP.py
The server is ready to receive

sent contents of aab.py
The server is ready to receive
```

Program 2

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.

LAB - 10/20

Q2] Using UDP socket, write a client-server program to make client sending filename, the server to send back the contents of the requested file if present.

Soln

ClientUDP.py

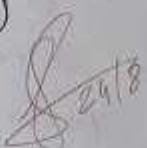
```
from socket import *
ServerName = "127.0.0.1"
ServerPort = 12000
clientSocket = socket(AF_INET, SOCK_DGRAM)
sentence = input("In Enter file name:")
clientSocket.sendto(bytes(sentence, "utf-8"), (ServerName, ServerPort))
filecontents, serverAddress = clientSocket.recvfrom(2048)
print("\nReply from server:\n")
print(filecontents.decode("utf-8"))
for i in filecontents:
    if i == '\n':
        print()
    else:
        print(i, end=' ')
clientSocket.close()
```

ServerUDP.py

```
from socket import *
ServerPort = 12000
```

Observation:

```
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("Sent contents of ", end="")
    print(sentence)
    # for i in sentence:
    #     print(str(i), end="")
    file.close()
```

Output 
second side

The server is ready to receive.

client side

file name : ServerUDP.py.

The contents of the file ServerUDP are displayed here.

server side

Sent contents of ServerUDP.py.

SOLUTION:

```
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:

Client:

```
= RESTART: C:/Users/Admin/AppData/Local/Programs/Python/Python310/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()  
  
>>>
```

Server:

```
>>>  
= RESTART: C:/Users/Admin/AppData/Local/Programs/Python/Python310/serverUDP.py =  
The server is ready to receive  
  
Sent contents of serverUDP.py
```

Lab 15

Observation

WireShark

1) It is a tool for network traffic analysis. WireShark is an open source application which is used for education and development of communication protocols and networks.

2) Troubleshooting: it is used so that each one has its specific needs. It is compatible with various network protocols and security problems. It is also used by network administrators to examine security problems. It allows them to use applications to analyze the data being transmitted over the network.

3) It is used by network engineers to monitor latency and malicious activities.

4) It can analyze dropped frames.

5) It helps us to know all the devices in local network.

Features of WireShark available for Linux and windows

- Capture wire packet data from a network interface
- Open files containing packet data captured with `tcpdump` / `winDump`, `wire shark` and many other packet capture programs
- Display packets with very detailed protocol information.
- Save packet dates captures.
- Filter packets on many criteria
- Create various statistics.

XLF
21/12/2023

