VISVESVARAYA TECHNOLOGICAL UNIVERSITY

“JnanaSangama”, Belgaum -590014, Karnataka.

LAB RECORD

Computer Network Lab (23CS5PCCON)

Submitted by

SUJAY PRASAD P V (1BM23CS422)

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
Academic Year 2024-25 (odd)

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 (23CS5PCCON)"** carried out by **SUJAY PRASAD P V (1BM23CS422),** who is a bonafide student of **B.M.S. College of Engineering.** It is in partial fulfilment for the award of **Bachelor of Engineering in Computer Science and Engineering** of the Visvesvaraya Technological University, Belgaum. The Lab report has been approved as it satisfies the academic requirements of the above-mentioned subject and the work prescribed for the said degree.

Prof. Srushti C S Assistant Professor Department of CSE, BMSCE	Dr. Kavitha Sooda Professor & HOD Department of CSE, BMSCE

Index

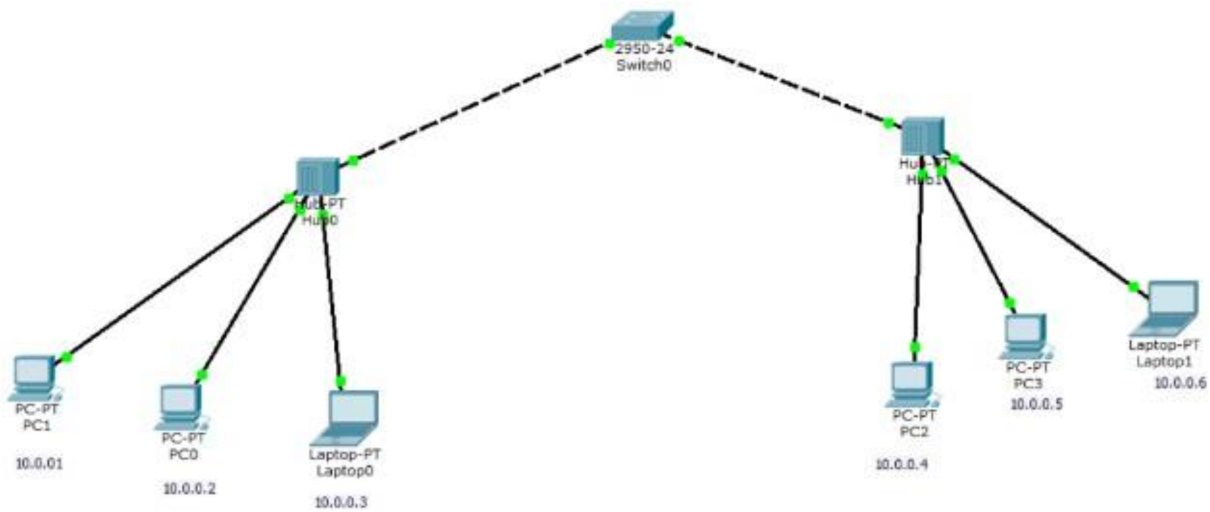
Sl. No.	Date	Experiment Title	Page No.
1	09/10/24	Create a topology and simulate sending a simple PDU from source to destination using hub and switch as connecting devices and demonstrate ping messages.	1 - 3
2	16/10/24	Configure IP address to routers in packet tracer. Explore the following messages: ping responses, destination unreachable, request timed out, reply.	4 - 7
3	23/10/24	Configure default route, static route to the Router.	8 - 10
4	13/11/24	Configure DHCP within a LAN and outside LAN.	11 - 14
5	20/11/24	Configure RIP routing Protocol in Routers .	15 - 16
6	20/11/24	Demonstrate the TTL/ Life of a Packet.	17 - 18
7	27/11/24	Configure OSPF routing protocol.	19 - 22
8	18/12/24	Configure Web Server, DNS within a LAN.	23 – 24
9	18/12/24	To construct a simple LAN and understand the concept and operation of Address Resolution Protocol (ARP).	25 – 27
10	18/12/24	To understand the operation of TELNET by accessing the router in the server room from a PC in the IT office.	28 – 30
11	18/12/24	To construct a VLAN and make the PC's communicate among a VLAN.	31 – 32
12	18/12/24	To construct a WLAN and make the nodes communicate wirelessly.	33 – 35
13	18/12/24	Write a program for error detecting code using CRC-CCITT (16-bits).	36 – 37
14	18/12/24	Write a program for congestion control using Leaky bucket algorithm.	38 – 41
15	18/12/24	Using TCP/IP 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.	42 – 44
16	18/12/24	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.	43 - 46

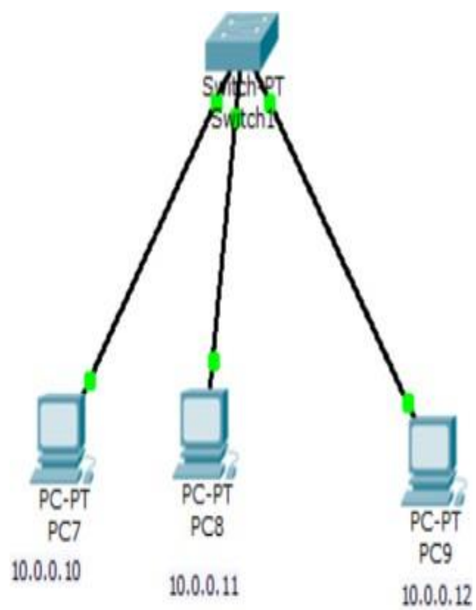
Github Link : https://github.com/SujayPrasadPV/CN

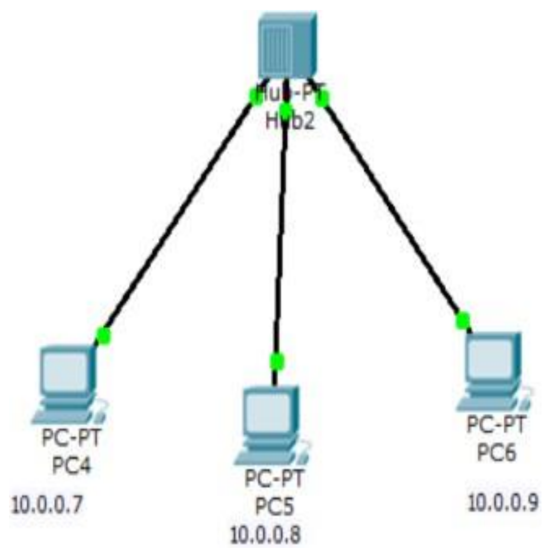
Program 1:

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

Topology:







Procedure and Observations:

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.

Aim of the Experiment:

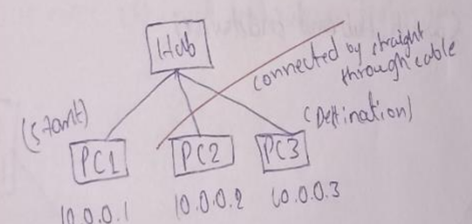
Simulating the transmission of simple PDU using Hub and Switch as connecting devices.

Devices Used:

Hub, Switch and End devices.

Topology 1:

Hub and 3 End devices



Procedure and Observations

i. connect end devices PC1, PC2, and PC3 to the hub through straight cable.

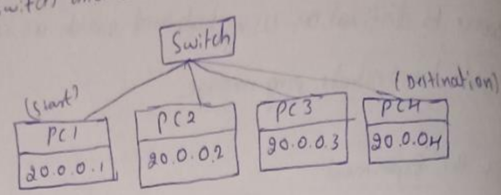
ii. Assign IP address to each of the end devices.

iii. Select a simple PDU. select PC1 as start node and PC3 as destination.

During simulation, the message will be recived by PC3 by PC2 and acknowledge the same.

Topology 2:

Switch and End devices



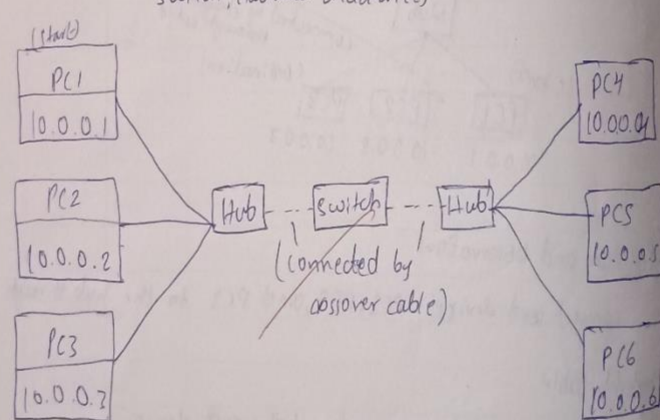
Connect 4 end devices PC1, PC2, PC3 PC4 to the switch with the mentioned IP addresses.

Select simple PDU, PC1 as start and PC4 as destination and simulate

Connection to be made through straight through cables. The message will be sent from PC1 to PC4 and in return the acknowledgement will be sent from PC4 to PC1.

Topology 3:

Switch, Hub and End devices



Connect the 3 end user devices PC1, PC2, PC3 with mentioned IP addresses to a Hub and further is connected to a switch. The connection between the Hub and switch is through a cross over cable.

Then connect switch to another hub with 3 end user devices with mentioned IP addresses.

Select a simple PDU and assign any one of the first three PCs as destination node.

Demonstrate the simulation and analyse the flow of message and acknowledgement from PC1 to PC6.

The succellfull ping mellage confirms the connectivity between the source and destination.

Difference between Hub and switch

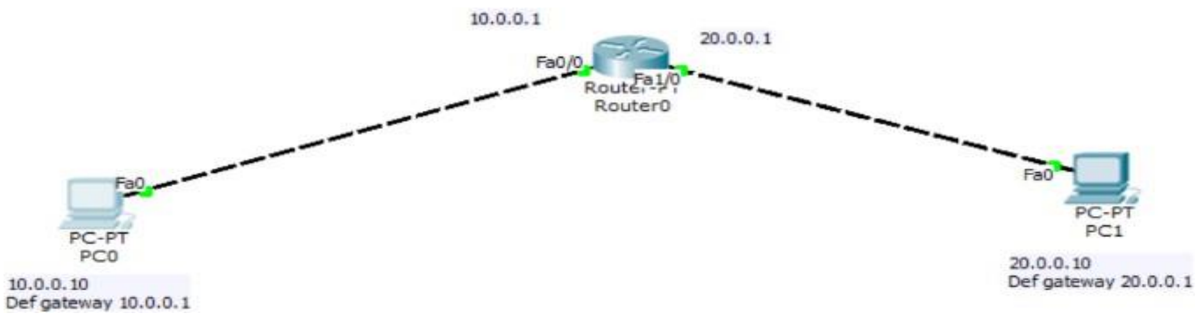
- Hub operates at the physical layer (layer1) of OSI model.
- It broadcasts data packets to all connected devices regardless of intended recipents.
- It is less efficient and supports lower speeds.

- Switch operates at data link layer (layer 2) of OSI model.
- It broadcast data packets only to specific device which data is intended.
- It is more efficiend and supports higher speed.

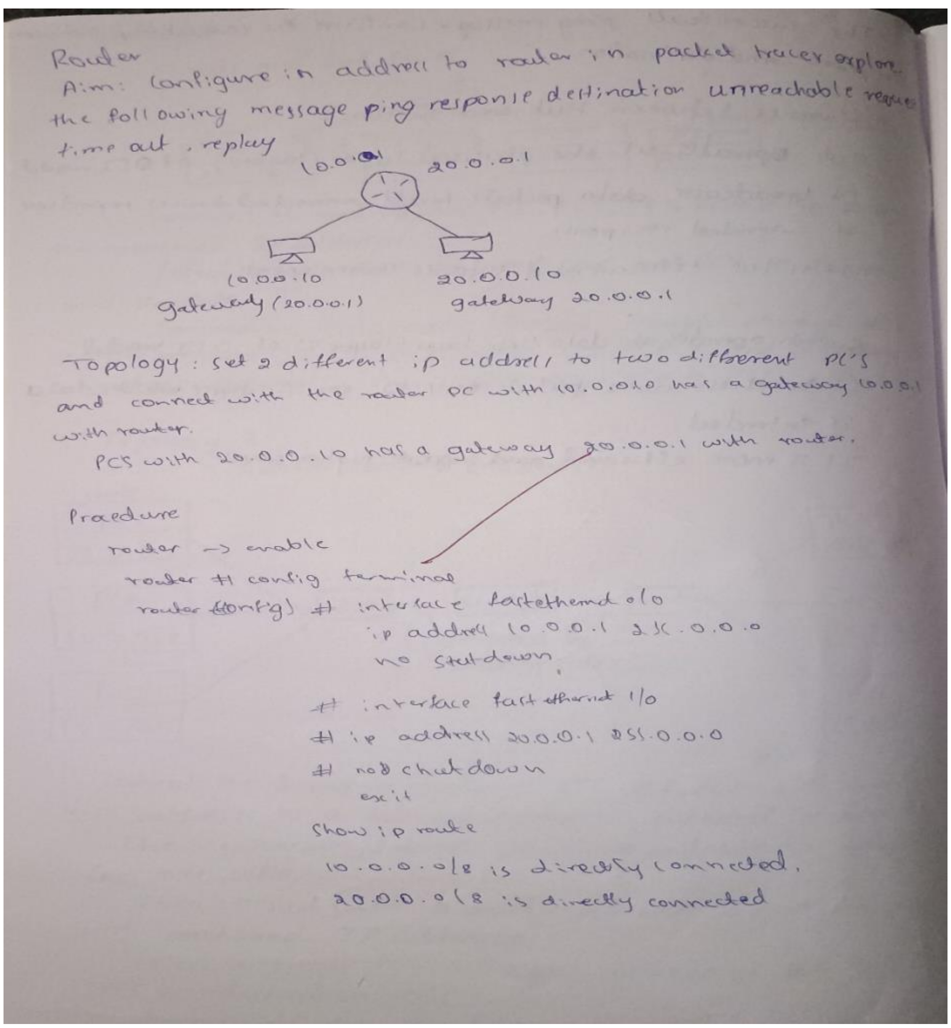
Program 2 :

Aim: Configure IP address to routers in packet tracer. Explore the following messages: ping responses, destination unreachable, request timed out, reply.

Topology:



Procedure and Observations:


Router
Aim: Configure in address to router in packet tracer explore the following message ping response destination unreachable request time out, replay

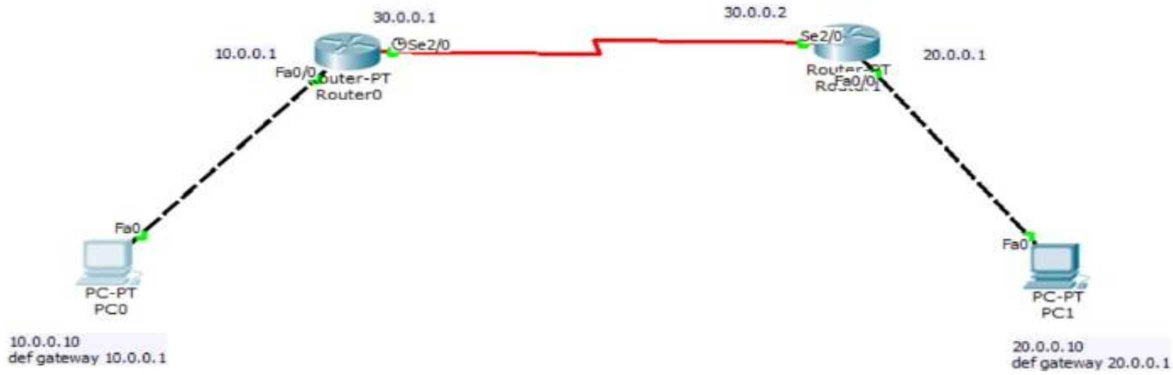
10.0.0.1 20.0.0.1

10.0.0.10 gateway (20.0.0.1)
20.0.0.10 gateway 20.0.0.1

Topology: set 2 different ip address to two different PC'S and connect with the router PC with 10.0.0.10 has a gateway 10.0.0.1 with router.
PCS with 20.0.0.10 has a gateway 20.0.0.1 with router.

Procedure
router -> enable
router # config terminal
router (config) # interface fastethernet 0/0
ip address 10.0.0.1 255.0.0.0
no shutdown

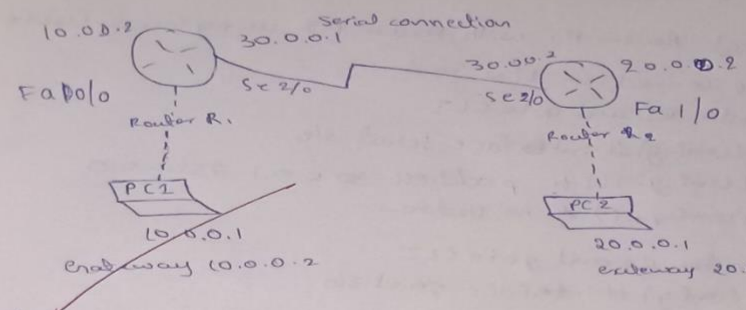
interface fast ethernet 1/0
ip address 20.0.0.1 255.0.0.0
no shutdown
exit
Show ip route
10.0.0.0/8 is directly connected.
20.0.0.0/8 is directly connected



2. Routers and End devices

Devices used : 2 routers and 2 end devices

Topology.



Procedure

- Select a generic router R1
- Connect an end device PC1 to router R1 through parallel connection fastethernet 0/0.
- Configure PC1 with ipaddress, router R2 and connect an end device PC2 fastetherenet 1/0.
- Configure PC2 with ipaddress 20.0.0.1 and gateway 20.0.0.2

Now select router R1 go to CLI and execute the following

Router > enable
Router# config terminal
Router(config)# interface fastethernet 0/0
Router(config-if)# ip address 10.0.0.2 255.0.0.0
Router (config-if)# no shutdown

" Interface fastethernet 0/0, changed state to up ".

Similarly select router R2 goto CLI and execute the same

Router> enable
Router# config terminal
Router(config)# interface fastethernet 1/0

Router (config-if) # ip address 20.0.0.2 255.0.0.0
Router (config-if) # no shutdown.
"Interface fastethernet 1/0, changed state to up".

Hence the connection b/w Router and end devices is established.

Now connect Router R1 with Router R2 using Serial Cable

To setup connection b/w routers,
Select router R1 and goto CLI.
Router (config) # interface serial 2/0.
Router (config-if) # ip address 30.0.0.1 255.0.0.0
Router (config-if) # no shutdown

Select router R2 and goto CLI
Router (config) # interface serial 3/0
Router (config-if)# ip address 30.0.0.2 255.0.0.0
Router (config-if) # no shutdown.
"Interface serial ~~2/0~~ 3/0 changed state to up".

Observations:

After setting up the ~~common~~ mentioned topology, try to ping PC1 to PC2

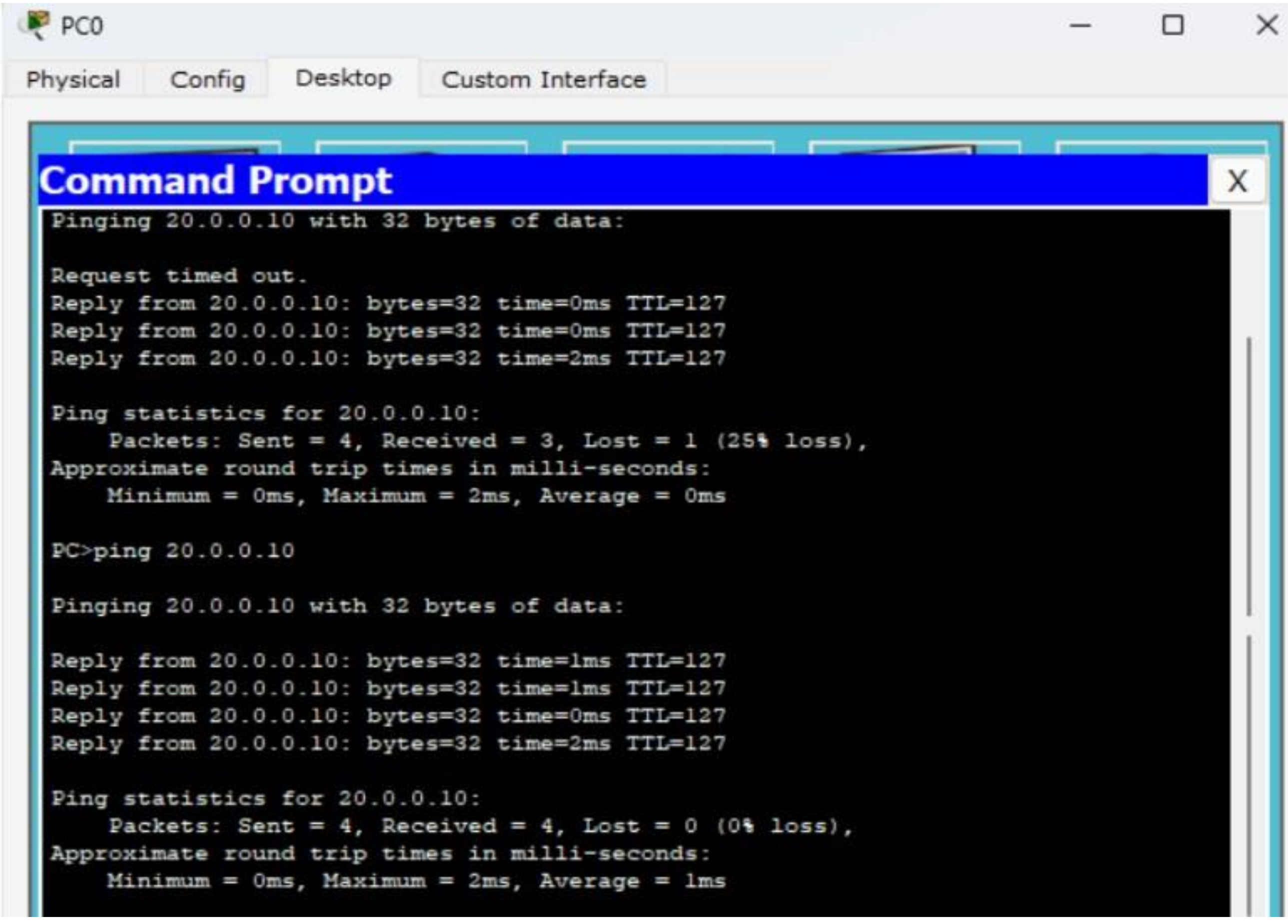
Open command prompt for PC1 type ping 20.0.0.1
Destination host unreachable
Packets sent 4: recieved: 0 lost : 4 Loss = 100%

It is also observed that the end system PC1 was only pinged with router R1 only.

~~ping~~ Ping 30.0.0.1 -> successfull.
Packets sent : 4 recieved : 4 lost : 0 loss = 0%.

23/10/24

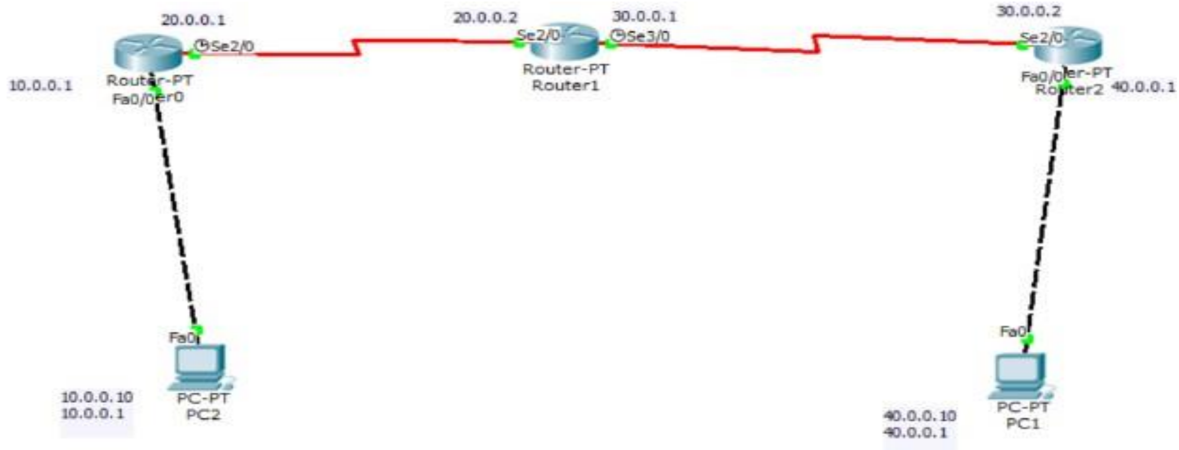
Hence although the routers were connected serialay the end devices were not able to ping each other.

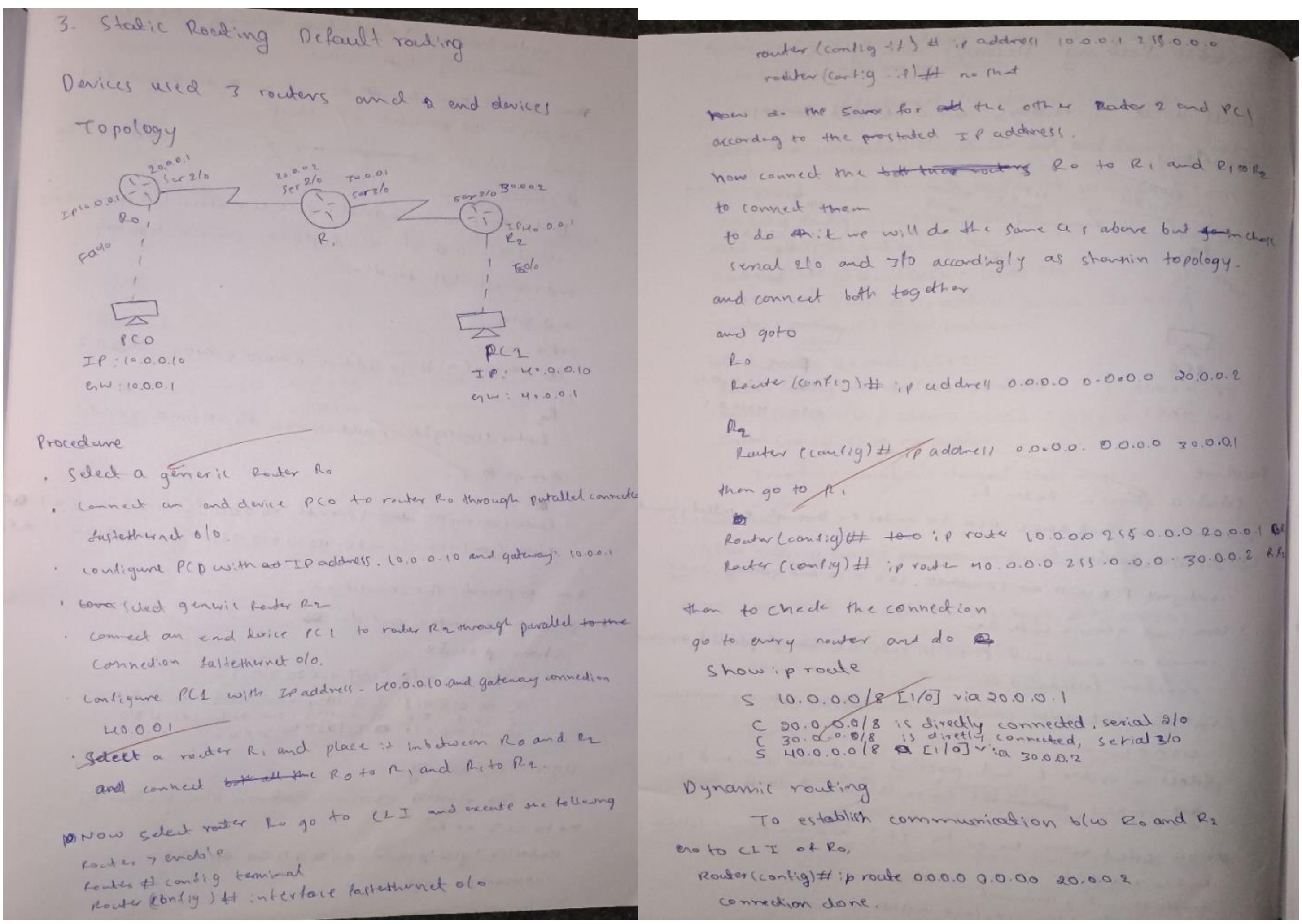

PC0
Physical
Config
Desktop
Custom Interface
Command Prompt
X
Pinging 20.0.0.10 with 32 bytes of data:
Request timed out.
Reply from 20.0.0.10: bytes=32 time=0ms TTL=127
Reply from 20.0.0.10: bytes=32 time=0ms TTL=127
Reply from 20.0.0.10: bytes=32 time=2ms TTL=127
Ping statistics for 20.0.0.10:
Packets: Sent = 4, Received = 3, Lost = 1 (25% loss),
Approximate round trip times in milli-seconds:
Minimum = 0ms, Maximum = 2ms, Average = 0ms
PC>ping 20.0.0.10
Pinging 20.0.0.10 with 32 bytes of data:
Reply from 20.0.0.10: bytes=32 time=1ms TTL=127
Reply from 20.0.0.10: bytes=32 time=1ms TTL=127
Reply from 20.0.0.10: bytes=32 time=0ms TTL=127
Reply from 20.0.0.10: bytes=32 time=2ms TTL=127
Ping statistics for 20.0.0.10:
Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
Minimum = 0ms, Maximum = 2ms, Average = 1ms

Program 3:

Aim: Configure default route, static route to the Router**.**

Topology, Procedure and Observations :





3. Static Routing Default routing

Devices used 3 routers and 2 end devices

Topology

20.0.0.1 Ser 2/0; 20.0.0.2 Ser 2/0; 30.0.0.1 Ser 3/0; Ser 2/0 30.0.0.2; IP 10.0.0.1 R_0; Fa0/0; R_1; IP 40.0.0.1 R_2; Fa0/0

PC0
IP : 10.0.0.10
GW : 10.0.0.1

PC1
IP : 40.0.0.10
GW : 40.0.0.1

Procedure

- Select a generic Router R_0
- Connect an end device PC0 to router R_0 through parallel connection fastethernet 0/0.
- configure PC0 with ~~at~~ IP address . 10.0.0.10 and gateway: 10.0.0.1
- ~~Conn~~ Select generic Router R_2
- connect an end device PC1 to router R_2 through parallel ~~to the~~ connection fastethernet 0/0.
- Configure PC1 with IP address - 40.0.0.10 and gateway connection 40.0.0.1
- Select a router R_1 and place it in between R_0 and R_2 and connect ~~both all the~~ R_0 to R_1 and R_1 to R_2.

Now select router R_0 go to CLI and execute the following

```
Router > enable
Router # config terminal
Router (config) # interface fastethernet 0/0
router (config-if) # ip address 10.0.0.1 255.0.0.0
router (config-if) # no shut
```

Now do the same for ~~all~~ the other Router 2 and PC1 according to the prestated IP address.

now connect the ~~both two routers~~ R_0 to R_1 and R_1 to R_2

to connect them

to do this we will do the same as above but ~~go~~ in chose serial 2/0 and 3/0 accordingly as shown in topology.

and connect both together

and goto

R_0

```
Router (config) # ip address 0.0.0.0 0.0.0.0 20.0.0.2
```

R_2

```
Router (config) # ip address 0.0.0.0. 0.0.0.0 30.0.0.1
```

then go to R_1

```
Router (config) # ~~to~~ 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
```

then to check the connection

go to every router and do

Show ip route

```
S  10.0.0.0/8 [1/0] via 20.0.0.1
C  20.0.0.0/8 is directly connected, serial 2/0
C  30.0.0.0/8 is directly connected, serial 3/0
S  40.0.0.0/8 [1/0] via 30.0.0.2
```

Dynamic routing

To establish communication b/w R_0 and R_2

go to CLI of R_0,

```
Router (config) # ip route 0.0.0.0 0.0.0.0 20.0.0.2
```

connection done.

Similarly go to CLI of router 2.

Router(config)# ip route 0.0.0.0 0.0.0.0 30.0.0.1

Hence communication done.

Now go to desktop of PC2.

Ping 40.0.0.10 (ip address of PC2)

Packets sent = 4 recieved = 4 lost = 0 0% Loss

Hence static routing and default routing is achieved.

Pinging 40.0.0.10 with 32 bytes of data:

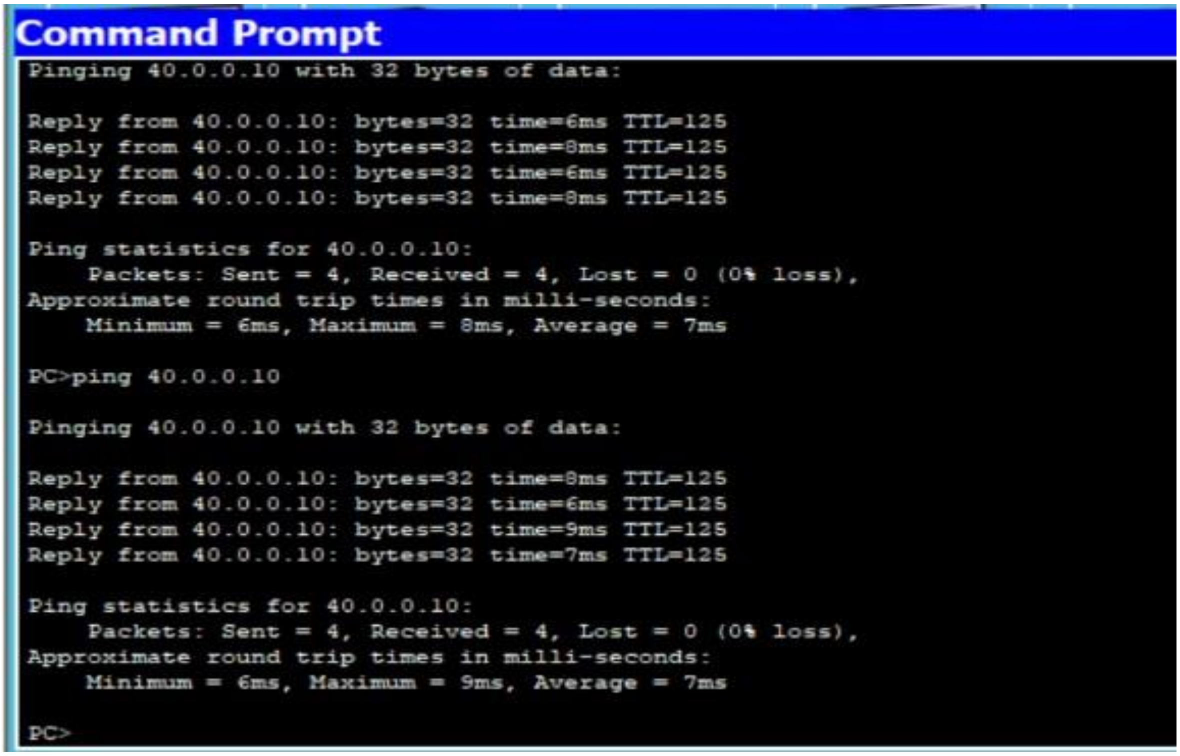
Reply from 40.0.0.10: bytes = 32 time = 7ms TTL = 125

Reply from 40.0.0.10 : bytes = 32 time = 6ms TTL = 125

Reply from 40.0.0.10 : bytes = 32 time = 9ms TTL = 125

Reply from 40.0.0.10 : bytes = 32 time = 6ms TTL = 125

13/11/24.



Command Prompt

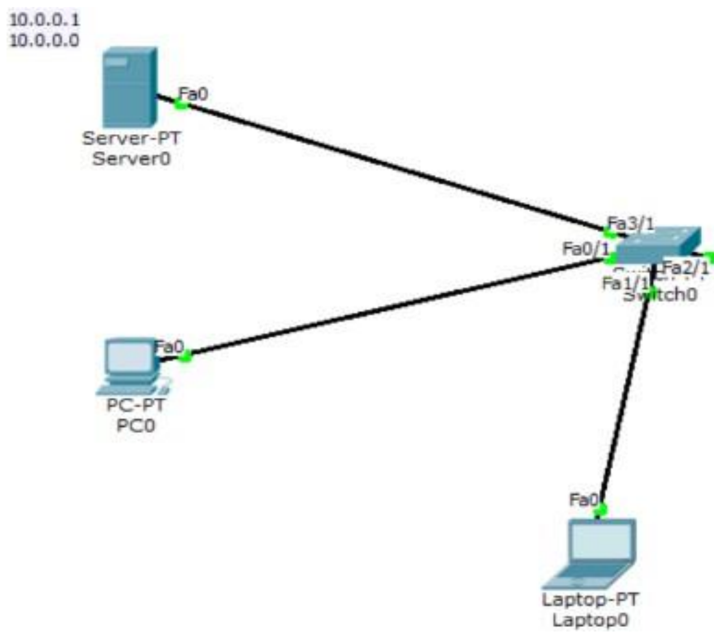
```
Pinging 40.0.0.10 with 32 bytes of data:

Reply from 40.0.0.10: bytes=32 time=6ms TTL=125
Reply from 40.0.0.10: bytes=32 time=8ms TTL=125
Reply from 40.0.0.10: bytes=32 time=6ms TTL=125
Reply from 40.0.0.10: bytes=32 time=8ms TTL=125

Ping statistics for 40.0.0.10:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 6ms, Maximum = 8ms, Average = 7ms

PC>ping 40.0.0.10

Pinging 40.0.0.10 with 32 bytes of data:

Reply from 40.0.0.10: bytes=32 time=8ms TTL=125
Reply from 40.0.0.10: bytes=32 time=6ms TTL=125
Reply from 40.0.0.10: bytes=32 time=9ms TTL=125
Reply from 40.0.0.10: bytes=32 time=7ms TTL=125

Ping statistics for 40.0.0.10:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 6ms, Maximum = 9ms, Average = 7ms

PC>
```

<u>Program 4:</u>

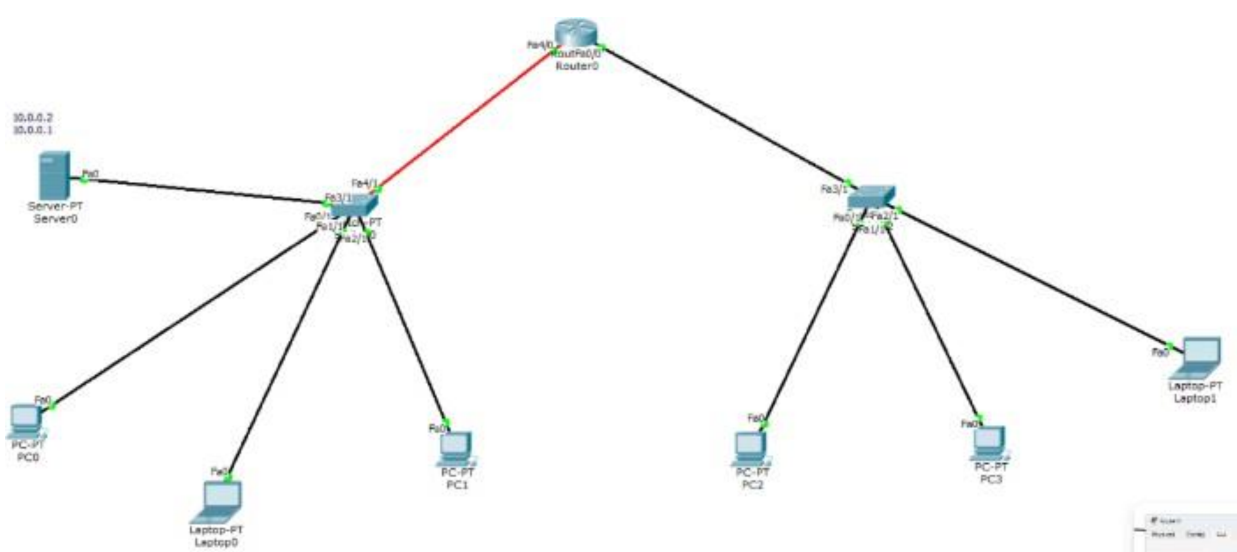
Aim: Configure DHCP within a LAN and outside LAN.

Topology:

Within LAN



Outside LAN

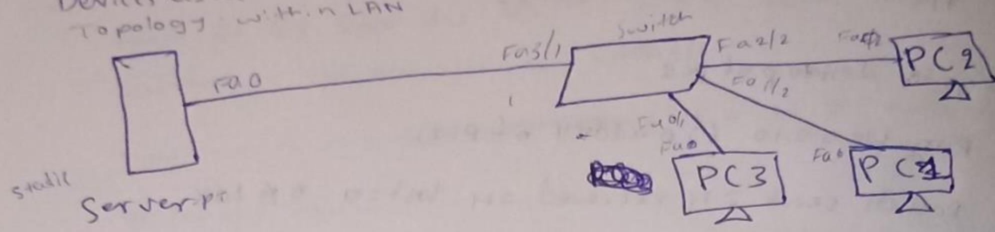


Procedure and Observation:

4 - Design a DHCP within LAN and Outside LAN

Devices used : One switch, one server, 3 end devices, ~~within~~

Topology within LAN



IP: 10.0.0.1

Gateway: 10.0.0.0

Procedure

Setup the topology as mentioned

Goto server IP configuration (Desktop)

Select IP address 10.0.0.1, 255.0.0.0, 10.0.0.0

Then to setup DHCP, goto config, server select DHCP

make DHCP to all three PC's.

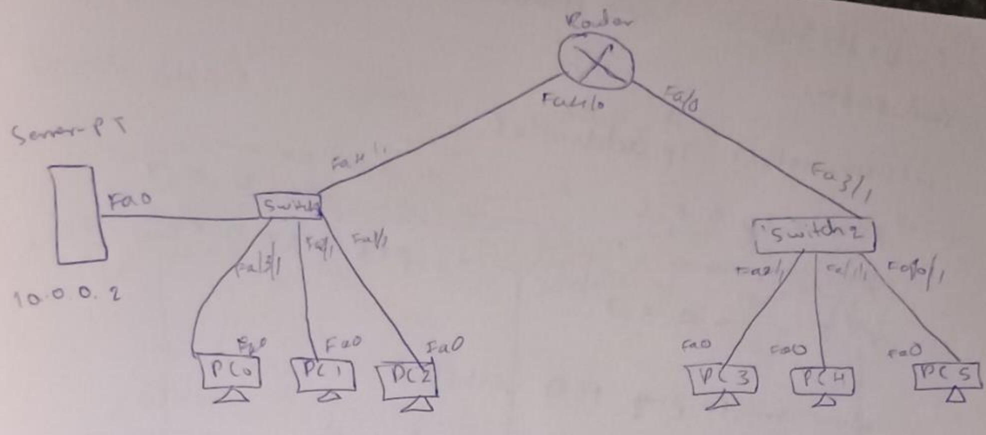
Dynamic IP address will be assigned.

Now ping PC-1 with PC-2 or PC-3

PC-1 had 10.0.0.2 so go to command prompt

Ping was successful ping 10.0.0.3

outside LAN



Setup the topology as mentioned

Go to server PT, Desktop, IP configuration
change default gateway to 10.0.0.1

Now go to Router CLI

Router > enable

Router # config terminal

Router(config) # interface fastethernet 4/0

Router (config-if) # ip address 10.0.0.1 255.0.0.0

Router (config-if) # ip helper-address 10.0.0.2

Router (config-if) # no shut

→ Interface established with switch 1 and Router

Similarly for switch 2

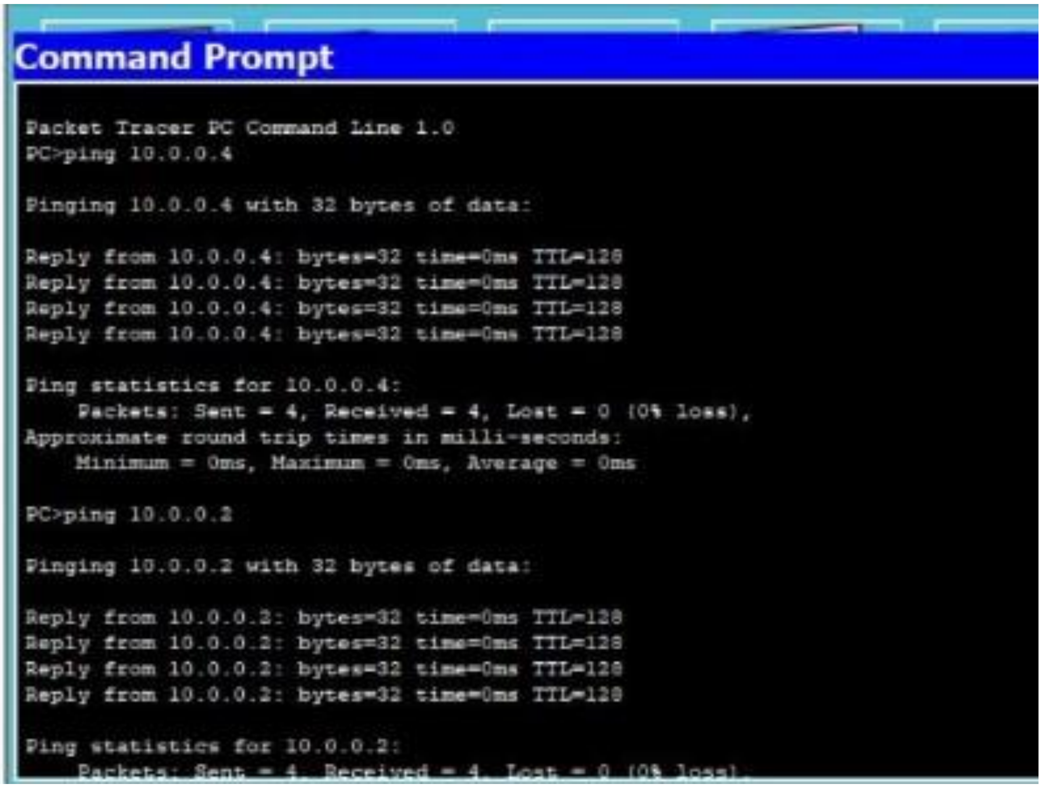
Router CLI, Router (config) # interface fastethernet 0/0

Router (config-if) # ip address 20.0.0.1 255.0.0.0

Router (config-if) # ip helper-address 10.0.0.2

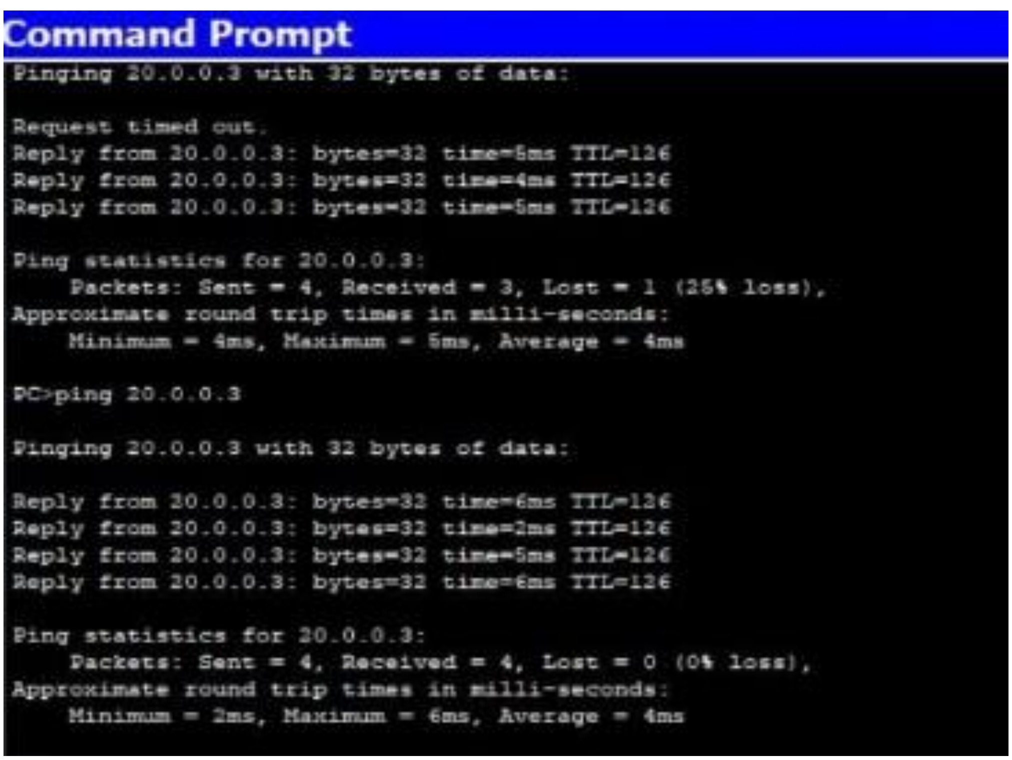
Router (config-if) # no shut

Proceed
20/11/24



```
Command Prompt

Packet Tracer PC Command Line 1.0
PC>ping 10.0.0.4

Pinging 10.0.0.4 with 32 bytes of data:

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

Ping statistics for 10.0.0.4:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 0ms, Average = 0ms

PC>ping 10.0.0.2

Pinging 10.0.0.2 with 32 bytes of data:

Reply from 10.0.0.2: bytes=32 time=0ms TTL=128
Reply from 10.0.0.2: bytes=32 time=0ms TTL=128
Reply from 10.0.0.2: bytes=32 time=0ms TTL=128
Reply from 10.0.0.2: bytes=32 time=0ms TTL=128

Ping statistics for 10.0.0.2:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
```

Within LAN



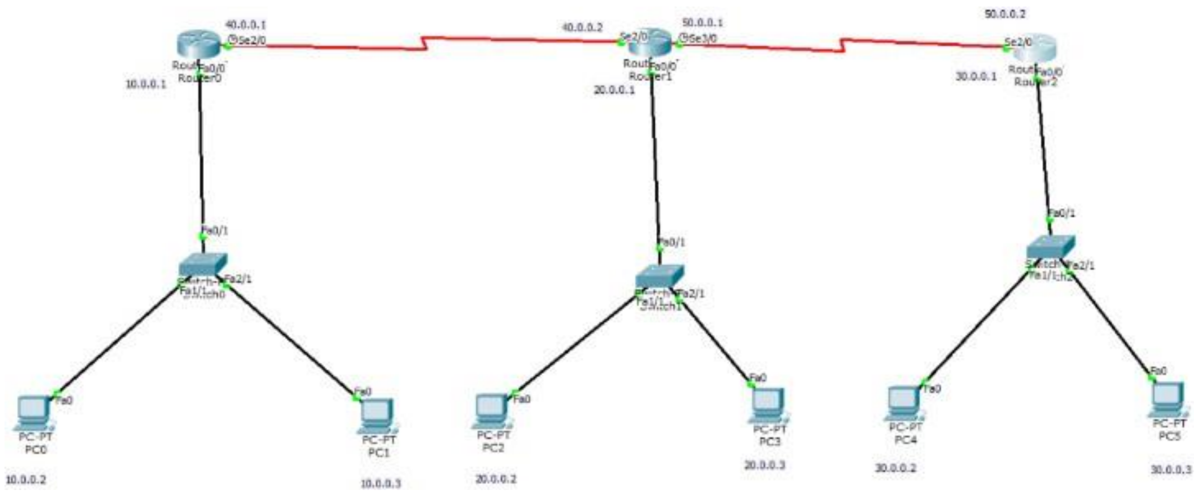
```
Command Prompt
Pinging 20.0.0.3 with 32 bytes of data:

Request timed out.
Reply from 20.0.0.3: bytes=32 time=5ms TTL=126
Reply from 20.0.0.3: bytes=32 time=4ms TTL=126
Reply from 20.0.0.3: bytes=32 time=5ms TTL=126

Ping statistics for 20.0.0.3:
    Packets: Sent = 4, Received = 3, Lost = 1 (25% loss),
Approximate round trip times in milli-seconds:
    Minimum = 4ms, Maximum = 5ms, Average = 4ms

PC>ping 20.0.0.3

Pinging 20.0.0.3 with 32 bytes of data:

Reply from 20.0.0.3: bytes=32 time=6ms TTL=126
Reply from 20.0.0.3: bytes=32 time=2ms TTL=126
Reply from 20.0.0.3: bytes=32 time=5ms TTL=126
Reply from 20.0.0.3: bytes=32 time=6ms TTL=126

Ping statistics for 20.0.0.3:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 2ms, Maximum = 6ms, Average = 4ms
```

Outside LAN

<u>Program 5:</u>

Aim: Configure RIP routing Protocol in Routers.

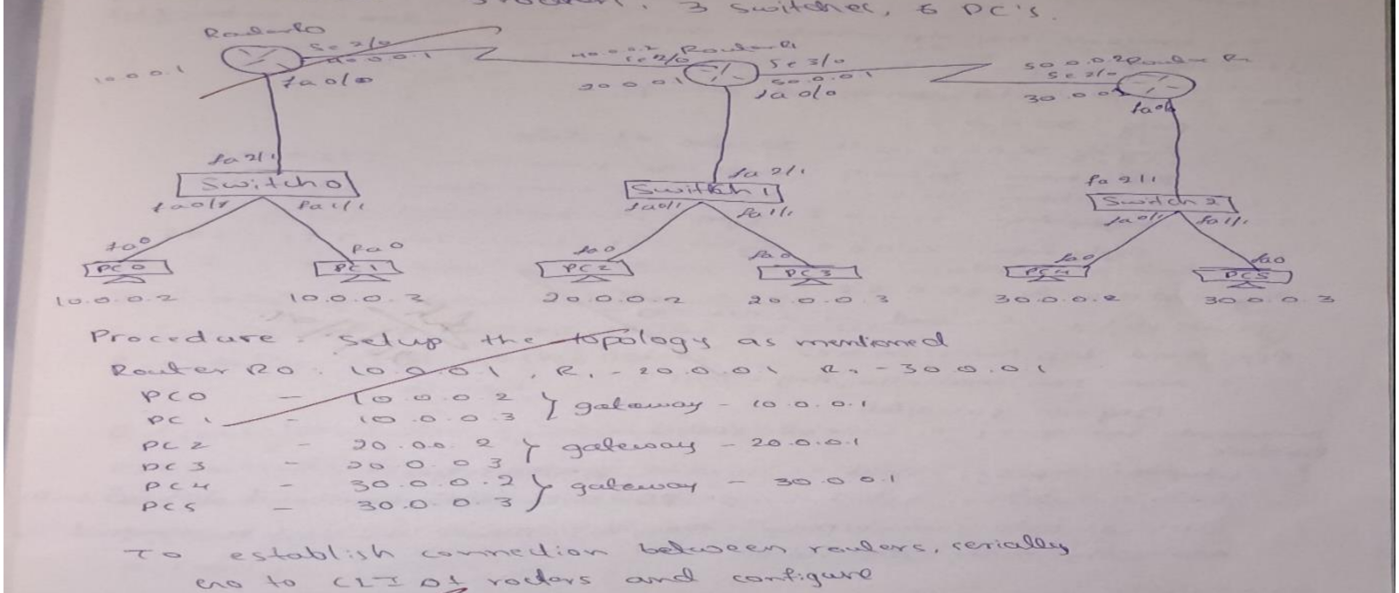
Topology:



Procedure and Observation:

5 Configure routing information protocol in routers

Devices used : 3 routers, 3 switches, 6 PC's.



Procedure : Setup the topology as mentioned

Router R0 : 10.0.0.1 , R1 - 20.0.0.1 R2 - 30.0.0.1

PC0 — 10.0.0.2
PC1 — 10.0.0.3 } gateway - 10.0.0.1

PC2 — 20.0.0.2
PC3 — 20.0.0.3 } gateway - 20.0.0.1

PC4 — 30.0.0.2
PC5 — 30.0.0.3 } gateway — 30.0.0.1

To establish connection between routers, serially go to CLI of routers and configure

R0 - Se 2/0 — 40.0.0.1
R1 - Se 2/0 - 40.0.0.2 , Se 3/0 - 50.0.0.1
R2 - Se 2/0 - 50.0.0.2

To establish RIP in routers, goto CLI of router 0.

Router (config)# router rip
Router (config-router) # network 10.0.0.0
network 40.0.0.0

Router 1
Router (config) router rip
Router (config-router) # network 20.0.0.0
network 40.0.0.0
network 50.0.0.0

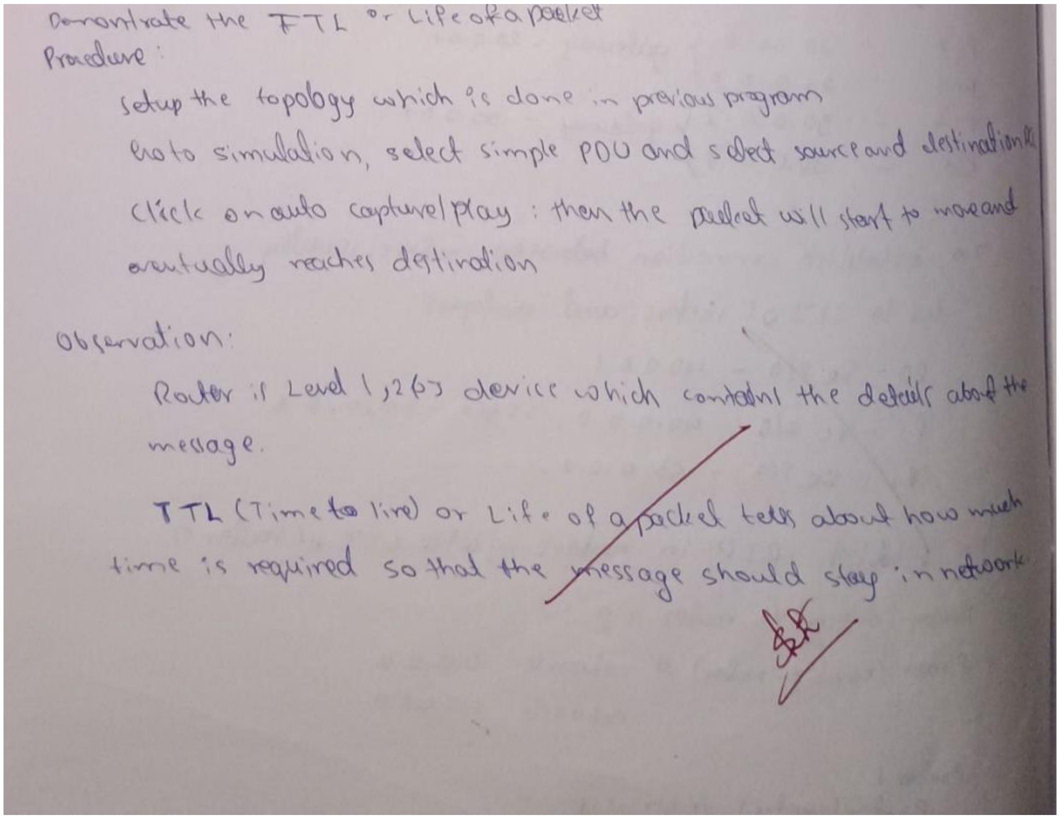
Command Prompt

```
Pinging 30.0.0.2 with 32 bytes of data:

Request timed out.
Reply from 30.0.0.2: bytes=32 time=7ms TTL=125
Reply from 30.0.0.2: bytes=32 time=6ms TTL=125
Reply from 30.0.0.2: bytes=32 time=7ms TTL=125

Ping statistics for 30.0.0.2:
    Packets: Sent = 4, Received = 3, Lost = 1 (25% loss),
Approximate round trip times in milli-seconds:
    Minimum = 6ms, Maximum = 7ms, Average = 6ms

PC>ping 30.0.0.2

Pinging 30.0.0.2 with 32 bytes of data:

Reply from 30.0.0.2: bytes=32 time=4ms TTL=125
Reply from 30.0.0.2: bytes=32 time=7ms TTL=125
Reply from 30.0.0.2: bytes=32 time=7ms TTL=125
Reply from 30.0.0.2: bytes=32 time=7ms TTL=125

Ping statistics for 30.0.0.2:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 4ms, Maximum = 7ms, Average = 6ms
```

Program 6:

Aim: Demonstrate the TTL/ Life of a Packet.

Procedure and Observation:



Demonstrate the TTL or Life of a packet

Procedure:

Setup the topology which is done in previous program

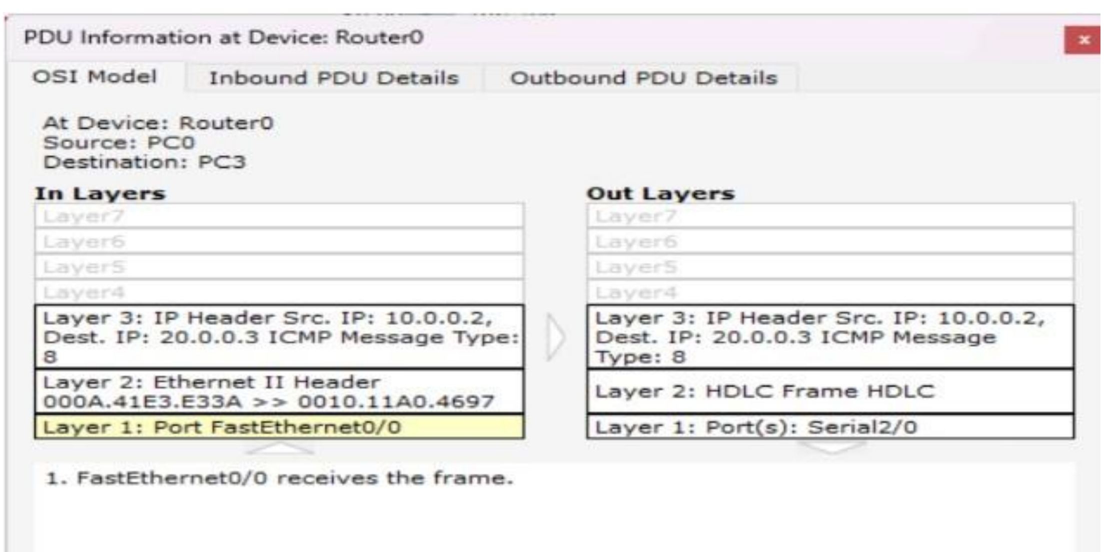
Goto simulation, select simple PDU and select source and destination

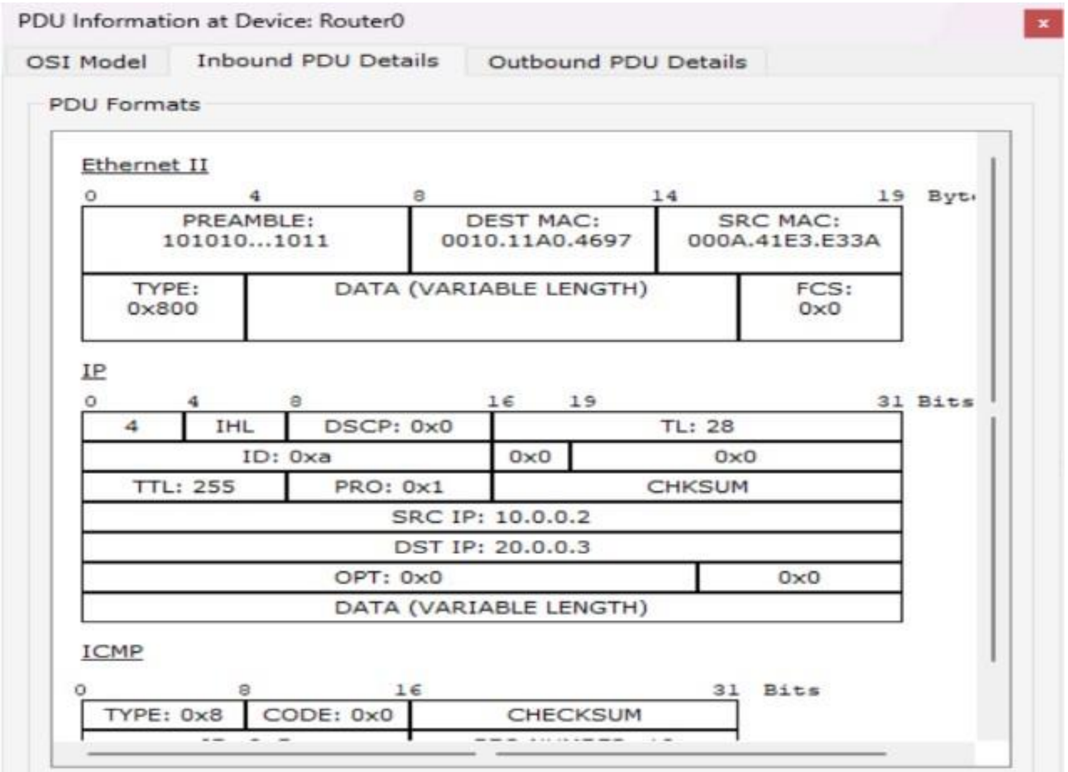
Click on auto capture/play : then the packet will start to move and eventually reaches destination

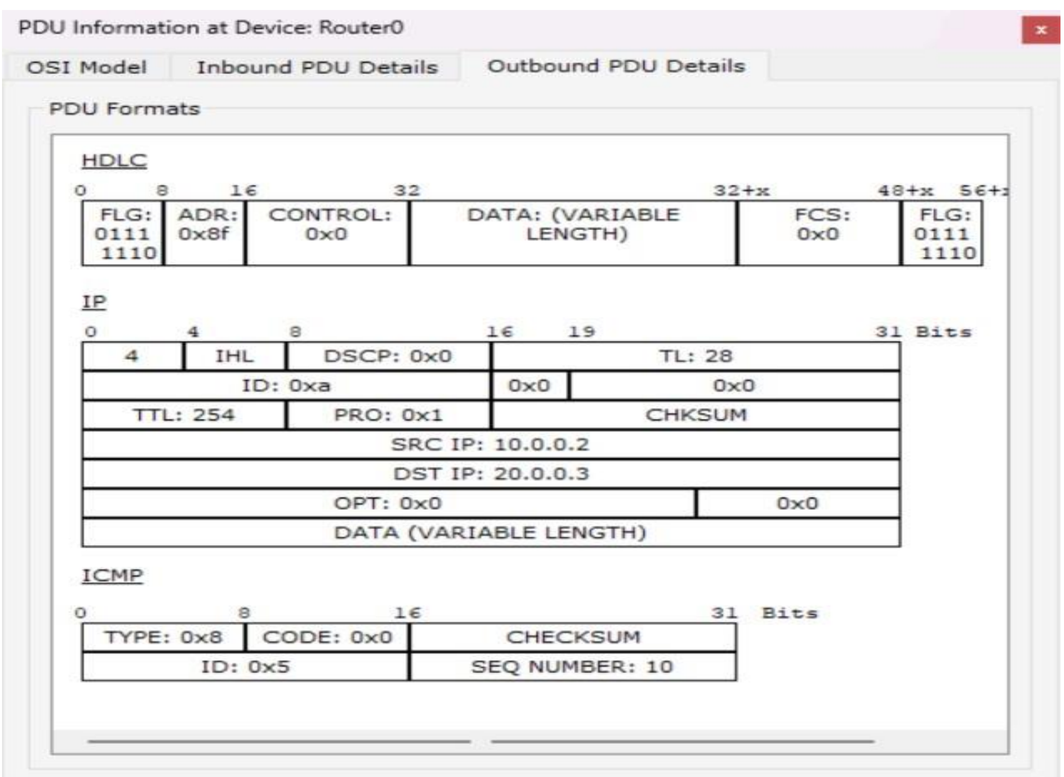
Observation:

Router is Level 1,2 & 3 device which contains the details about the message.

TTL (Time to live) or Life of a packet tells about how much time is required so that the message should stay in network.



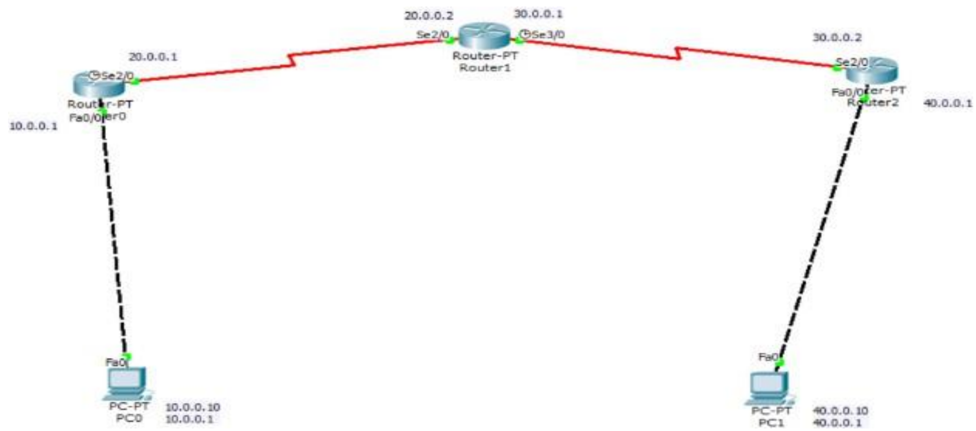

PDU Information at Device: Router0
OSI Model
Inbound PDU Details
Outbound PDU Details
PDU Formats
Ethernet II
0 4 8 14 19 Byt
PREAMBLE: 101010...1011
DEST MAC: 0010.11A0.4697
SRC MAC: 000A.41E3.E33A
TYPE: 0x800
DATA (VARIABLE LENGTH)
FCS: 0x0
IP
0 4 8 16 19 31 Bits
4
IHL
DSCP: 0x0
TL: 28
ID: 0xa
0x0
0x0
TTL: 255
PRO: 0x1
CHKSUM
SRC IP: 10.0.0.2
DST IP: 20.0.0.3
OPT: 0x0
0x0
DATA (VARIABLE LENGTH)
ICMP
0 8 16 31 Bits
TYPE: 0x8
CODE: 0x0
CHECKSUM


PDU Information at Device: Router0
OSI Model
Inbound PDU Details
Outbound PDU Details
PDU Formats
HDLC
0 8 16 32 32+x 48+x 56+
FLG: 0111 1110
ADR: 0x8f
CONTROL: 0x0
DATA: (VARIABLE LENGTH)
FCS: 0x0
FLG: 0111 1110
IP
0 4 8 16 19 31 Bits
4
IHL
DSCP: 0x0
TL: 28
ID: 0xa
0x0
0x0
TTL: 254
PRO: 0x1
CHKSUM
SRC IP: 10.0.0.2
DST IP: 20.0.0.3
OPT: 0x0
0x0
DATA (VARIABLE LENGTH)
ICMP
0 8 16 31 Bits
TYPE: 0x8
CODE: 0x0
CHECKSUM
ID: 0x5
SEQ NUMBER: 10

Program 7:

Aim: Configure OSPF routing protocol**.**

Topology:

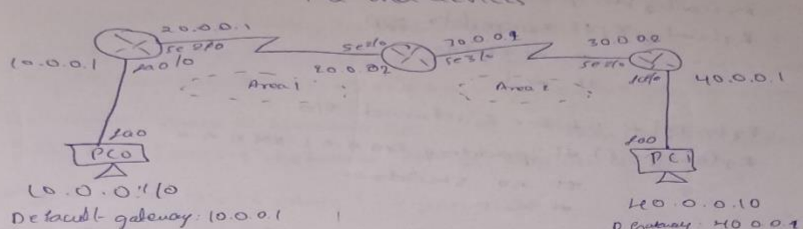


Procedure and Observation:

6. Configure OSPF Routing Protocol and Connect Areas

Devices used: 3 routers, 2 end devices

Step 1:



Step 2: configure ip address to all interface in Router R1

```
R1(config)# interface fastethernet 0/0
R1(config-if)# ip address 10.0.0.1 255.0.0.0
R1(config-if)# no shutdown
           # exit

R1(config)# interface serial 1/0
R1(config-if)# ip address 20.0.0.1 255.0.0.0
R1(config-if)# encapsulation ppp
R1(config-if)# clock rate 64000
R1(config-if)# exit
```

In Router R2

```
R2(config)# interface serial 1/0
R2(config-if)# ip address 20.0.0.2 255.0.0.0
R2(config-if)# encapsulation PPP
R2(config-if)# no shutdown
           # exit

R2(config)# interface serial 1/01
R2(config-if)# ip address 30.0.0.1 255.0.0.0
           # encapsulation PPP
           # clock rate 64000
           # no shutdown
           # exit
```

In Router R3

R3 (config) # interface serial 1/0

R3 (config) # ip address 30.0.0.2 255.0.0.0

R3 (config-if) # encapsulation ppp

no shutdown

exit

R3 (config) # interface fastethernet 2/0

R3 (config-if) # ipaddress 40.0.0.1 255.0.0.0

no shutdown

exit

Step 3: Now, enable ip routing by configuring ospf routing protocol in all routers

Router R1

~~Rout~~(config) # router ospf 1

R1 (config-router) # router-id 1.1.0.1

R1 (config-router) # network 10.0.0.0 0.255.255.255 area 0

network 20.0.0.0 0.255.255.255 area 1

exit

Router R2

R2 (config) # router ospf 1

R2 (config-router) # router-id 2.2.2.2

network 20.0.0.0 0.255.255.255 area 1

network 30.0.0.0 0.255.255.255 area 0

exit

Router R3

R3 (config) # router ospf 1

R3 (config-router) # router-id 3.3.3.3

network 30.0.0.0 0.255.255.255 area 0

network 40.0.0.0 0.255.255.255 area 2

we have to configure router-id when we configure ospf. it is used to identify the router.

Step 4: Now check routing table of R_1

Router # show ip route

C 10.0.0.0/8 is directly connected, FastEthernet 2/0

C 20.0.0.0/8 is directly connected, Serial 1/0

O IA 40.0.0.0/8 [110/129] via 20.0.0.2; 00:04:23 Serial 1/0

O IA 30.0.0.0/8 [110/128] via 20.0.0.2, 00:07:29, Serial 1/0

Here, R_2 knows Area 0 Network 20.0.0.0. Connected to R_2 from R_1; SO R_1 learns networks through this network.

R_3 (config) # router ospf 1, Here 1 is process ID.

There must be one interface up to keep ospf process up. So it's better to configure loophole address to routers. It is a virtual interface never goes down once we configured.

R_1 (config) # interface loopback 0

R_1 (config-if) # ip address 172.16.1.252 255.255.0.0

no shut down

R_2 (config) # interface loopback 0

R_2 (config-if) # ip address 172.16.1.253 255.255.0.0

no shutdown

R_3 (config) # interface loopback 0

R_3 (config-if) # ip address 172.16.1.254 255.255.0.0

no shutdown

Step 5: Now, check Routing table of R_3

R_3 # show ip route

O IA 20.0.0.0/8 [110/128] via 30.0.0.1, 00:18:38 Serial 1/0

C 40.0.0.0/8 is directly connected, FastEthernet 2/0

C 30.0.0.0/8 is directly connected, Serial 1/0

Here, R_3 doesn't know about the area 3, so we have to create virtual link between R_1 and R_2

Step 6: Create virtual link between R1, R2 by this we creat a virtual link to connect area 3 to area 0

In Router R1

R1(config) # router ospf 1

R1(config-router) # area 1 virtual-link 2.2.2.2

R2(config) # router ospf 1

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

exit

Step 7: R2 and R3 get updates about Area 3. Now, check routing table of R3

R3# show ip route

O IA 20.0.0.0/8 [110/128] via 30.0.0.1, 00:01:56, serial 1/0

C 40.0.0.0/8 is directly connected, Fastethernet 2/0

O IA 10.0.0.0/8 [110/129] via 30.0.0.1, 00:01:56 serial 1/0

C 30.0.0.0/8 is directly connected, serial 1/0

Step 8: Check connectivity between host 10.0.0.10 to 40.0.0.10

ping 40.0.0.010

Now, if we get the reply without loss then the connection is established.

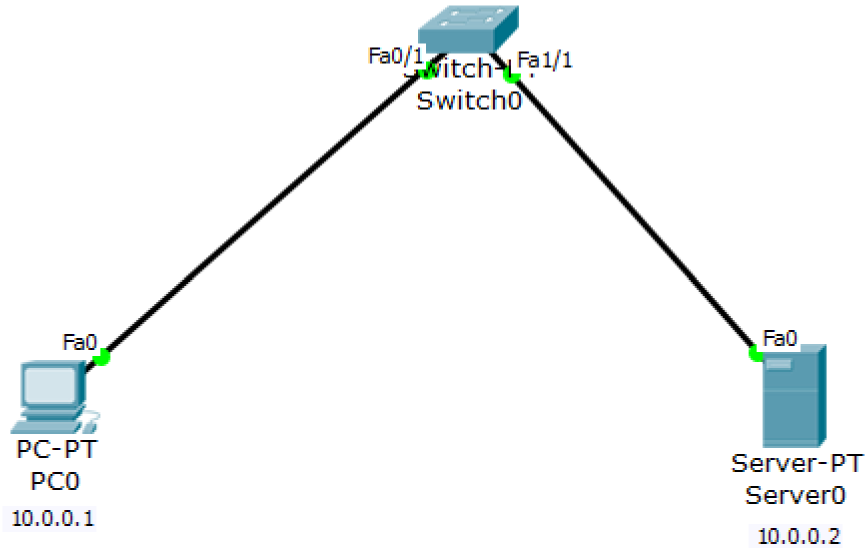
30/12/24

```
PC>ping 40.0.0.10

Pinging 40.0.0.10 with 32 bytes of data:

Reply from 40.0.0.10: bytes=32 time=7ms TTL=125
Reply from 40.0.0.10: bytes=32 time=7ms TTL=125
Reply from 40.0.0.10: bytes=32 time=6ms TTL=125
Reply from 40.0.0.10: bytes=32 time=6ms TTL=125

Ping statistics for 40.0.0.10:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 6ms, Maximum = 7ms, Average = 6ms
```

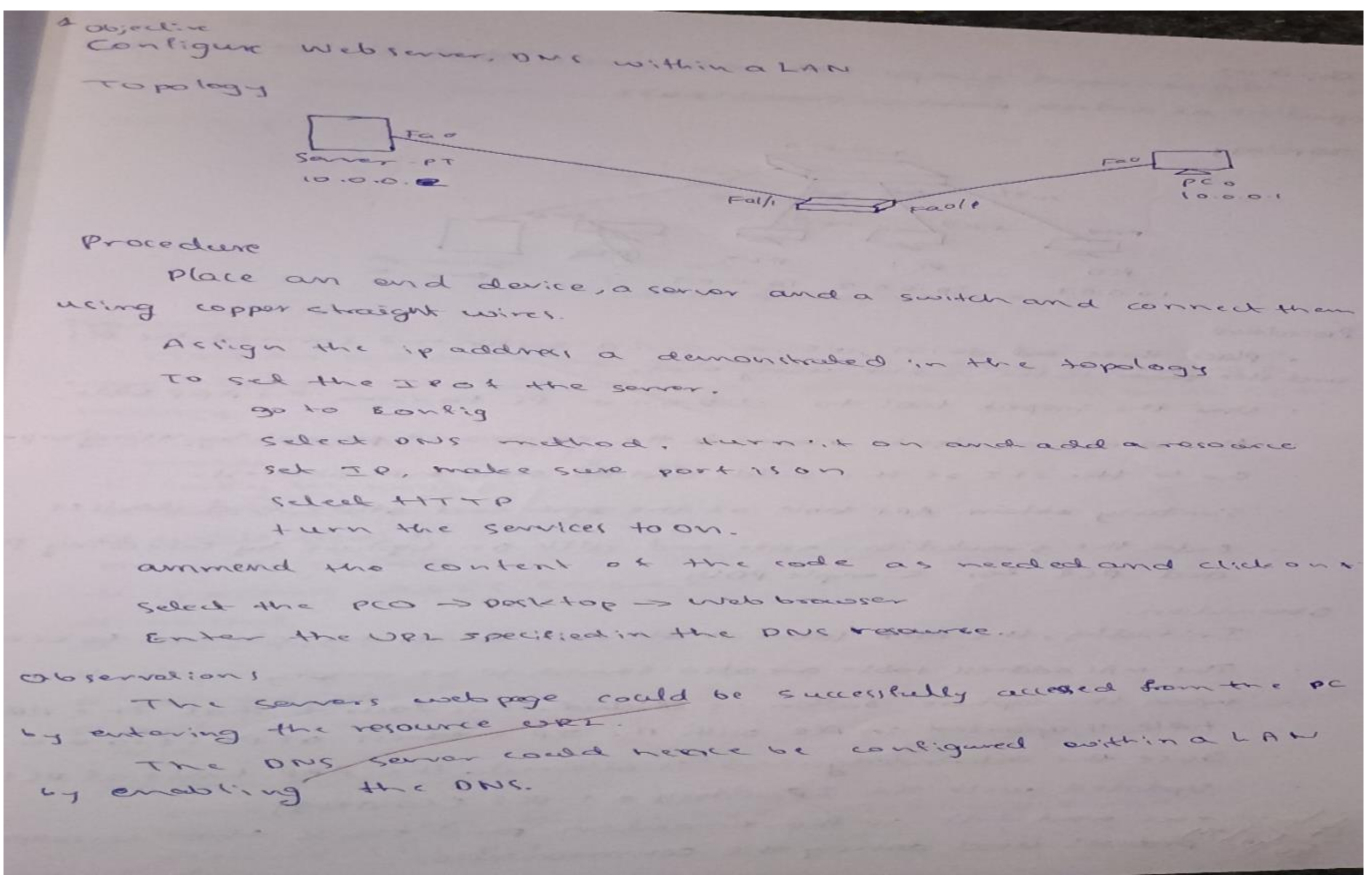
Program 8:

Aim: Configure Web Server, DNS within a LAN.

Topology:



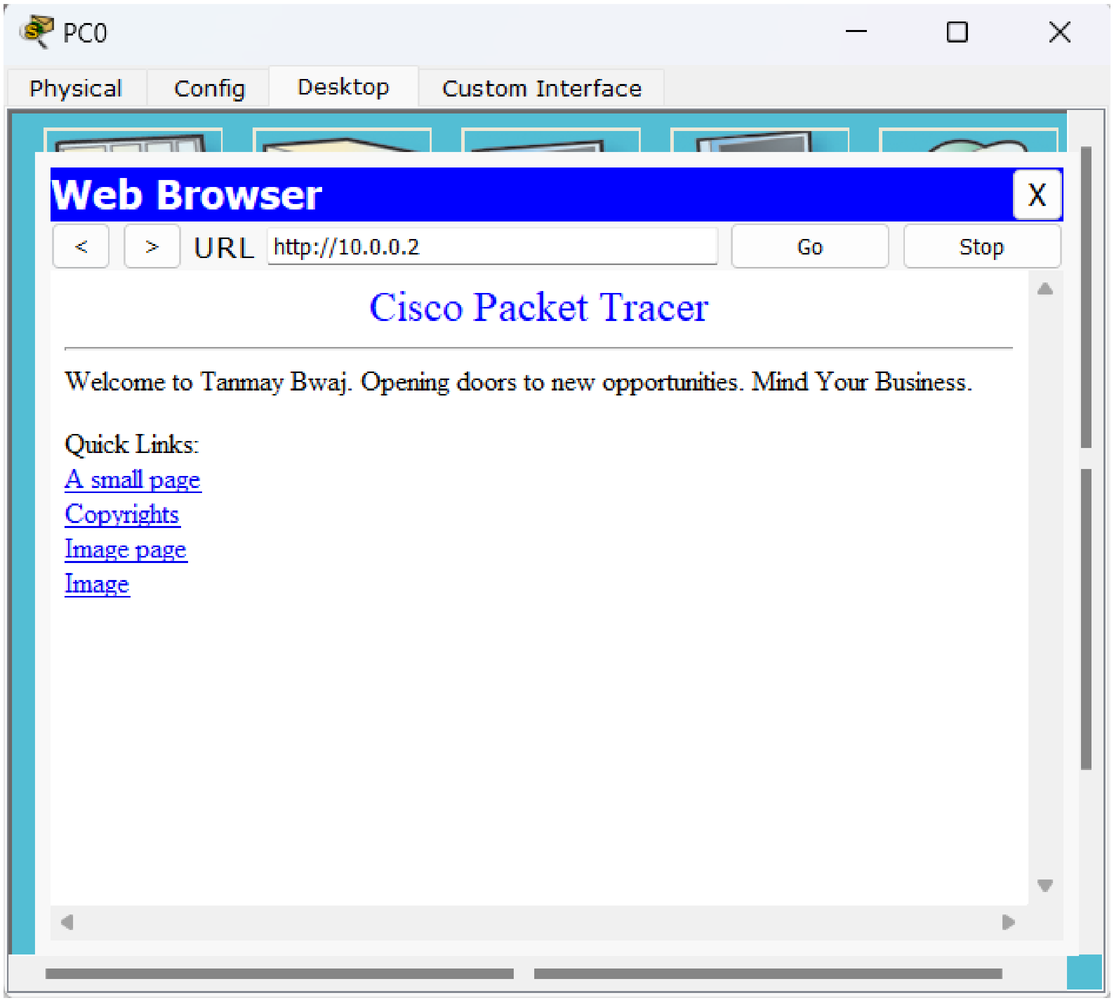
Procedure and Observations:



Objective
Configure Web server, DNS within a LAN
Topology

Procedure
Place an end device, a server and a switch and connect them using copper straight wires.
Assign the ip address a demonstrated in the topology
To set the IP of the server.
go to config
Select DNS method, turn it on and add a resource
set IP, make sure port is on
Select HTTP
turn the services to on.
ammend the content of the code as needed and click on s
Select the PC0 → Desktop → Web browser
Enter the URL specified in the DNS resource.

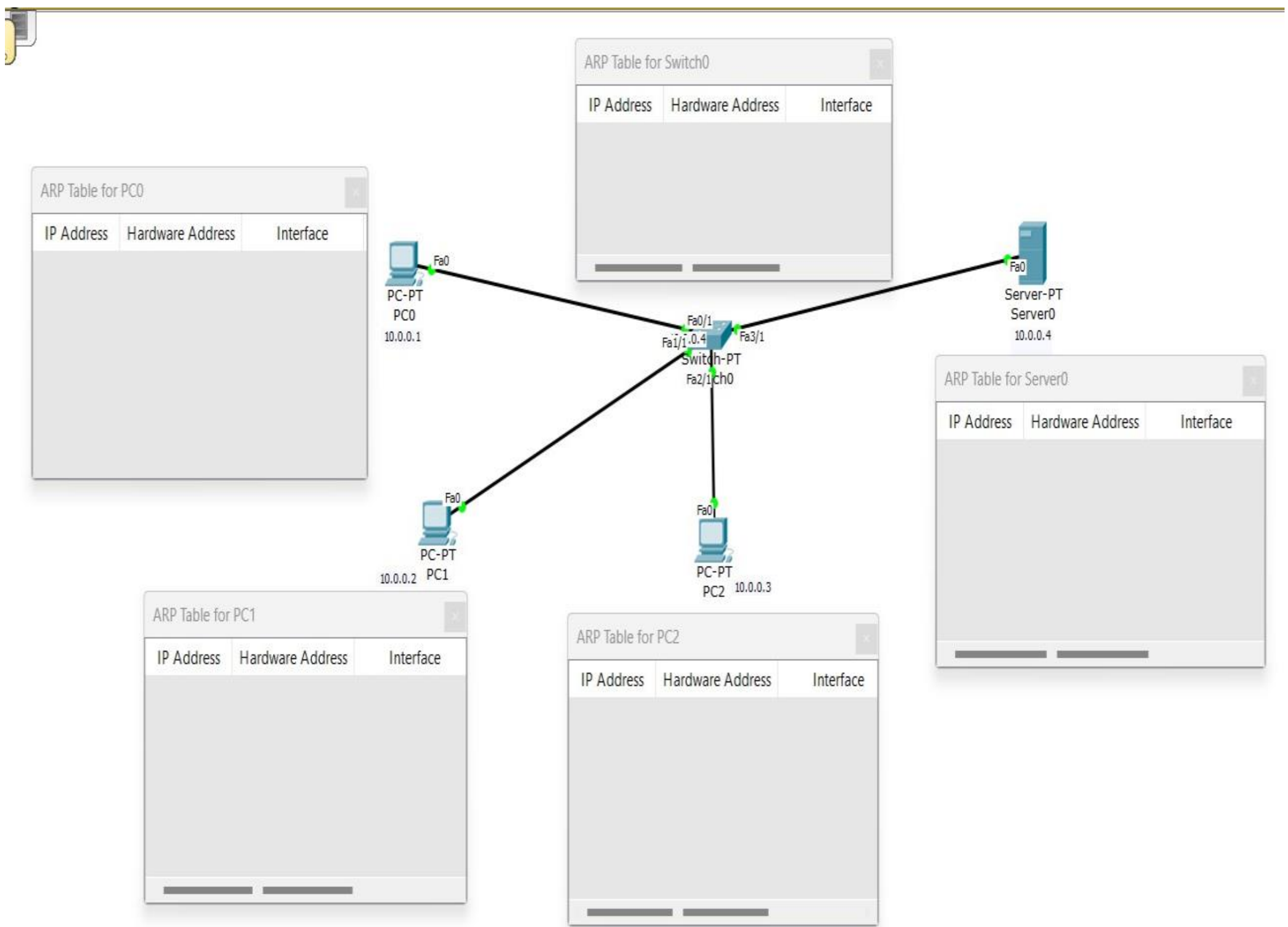
Observations
The servers web page could be successfully accessed from the PC by entering the resource URL.
The DNS server could hence be configured within a LAN by enabling the DNS.


PC0
Physical
Config
Desktop
Custom Interface
Web Browser
X
<
>
URL
http://10.0.0.2
Go
Stop
Cisco Packet Tracer
Welcome to Tanmay Bwaj. Opening doors to new opportunities. Mind Your Business.
Quick Links:
A small page
Copyrights
Image page
Image

<u>Program 9:</u>

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

Topology:

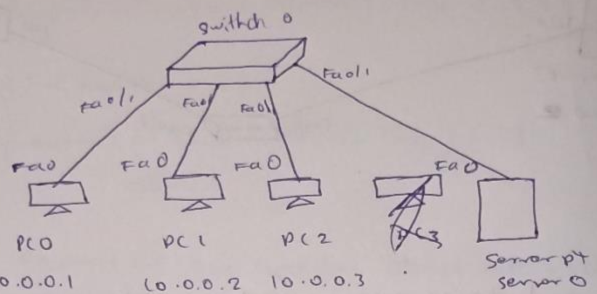


Procedure and Observations:

2 Objective

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

Topology



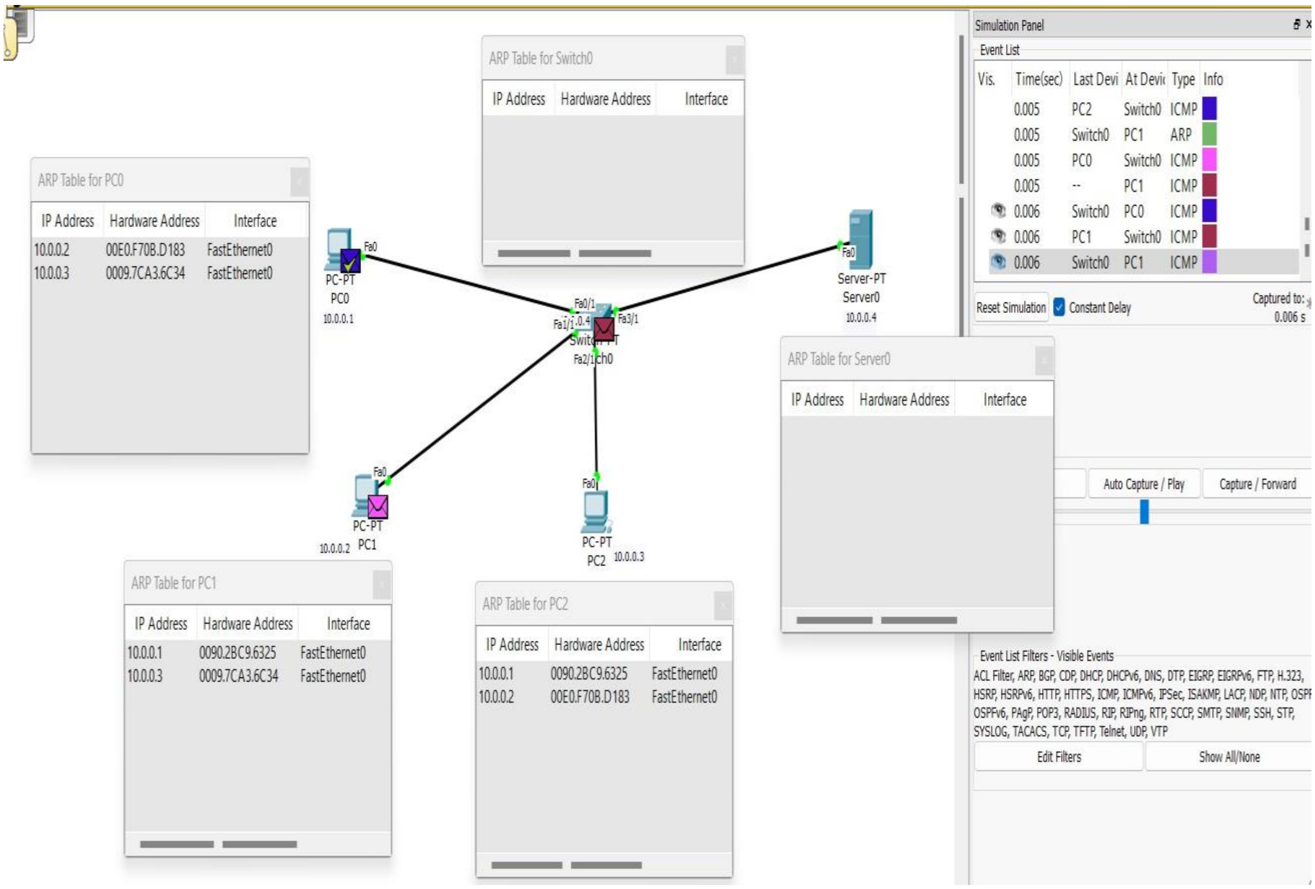
Procedure

- Place three end devices, a server and a switch and connect the PCs and the server to the switch using wires.
- User the inspect tool to click on a PC to view the ARP table.
- the same can also be viewed in the command prompt by using 'arp -a'
- Go to the CLI of the switch and do show MAC address.
- Similarly obtain ARP table of the server and other end devices.
- Enter the simulation mode and click on capture by selecting PC1 and PC2 for simple PDUs.

Observation.

- Initially, the ARP tables of all end devices are observed to be empty
- The MAC address table are also found to be empty
- when the capture button is clicked. it is found that the ARP table is updated in PC2 with the IP address of PC1 (10.0.0.2)
- Once the acknowledgement is obtained, the ARP table of PC1 is updated with the IP address of PC2 (10.0.0.3)
- The event list in the simulation panel shows the corresponding protocol used during the communication.

30/12/24

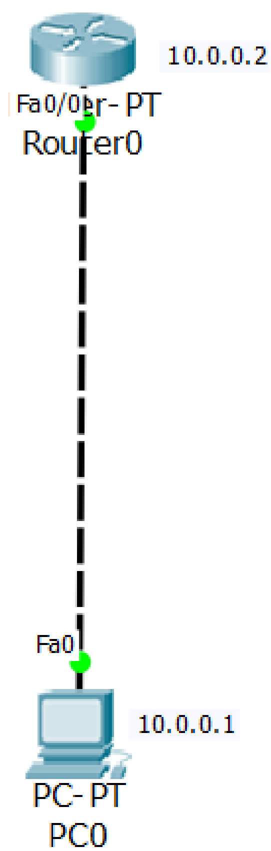



```
Switch>show mac address-table
          Mac Address Table
-------------------------------------------

Vlan    Mac Address       Type        Ports
----    -----------       --------    -----

   1    0009.7ca3.6c34    DYNAMIC     Fa2/1
   1    0090.2bc9.6325    DYNAMIC     Fa0/1
   1    00e0.f70b.d183    DYNAMIC     Fa1/1
Switch>
```

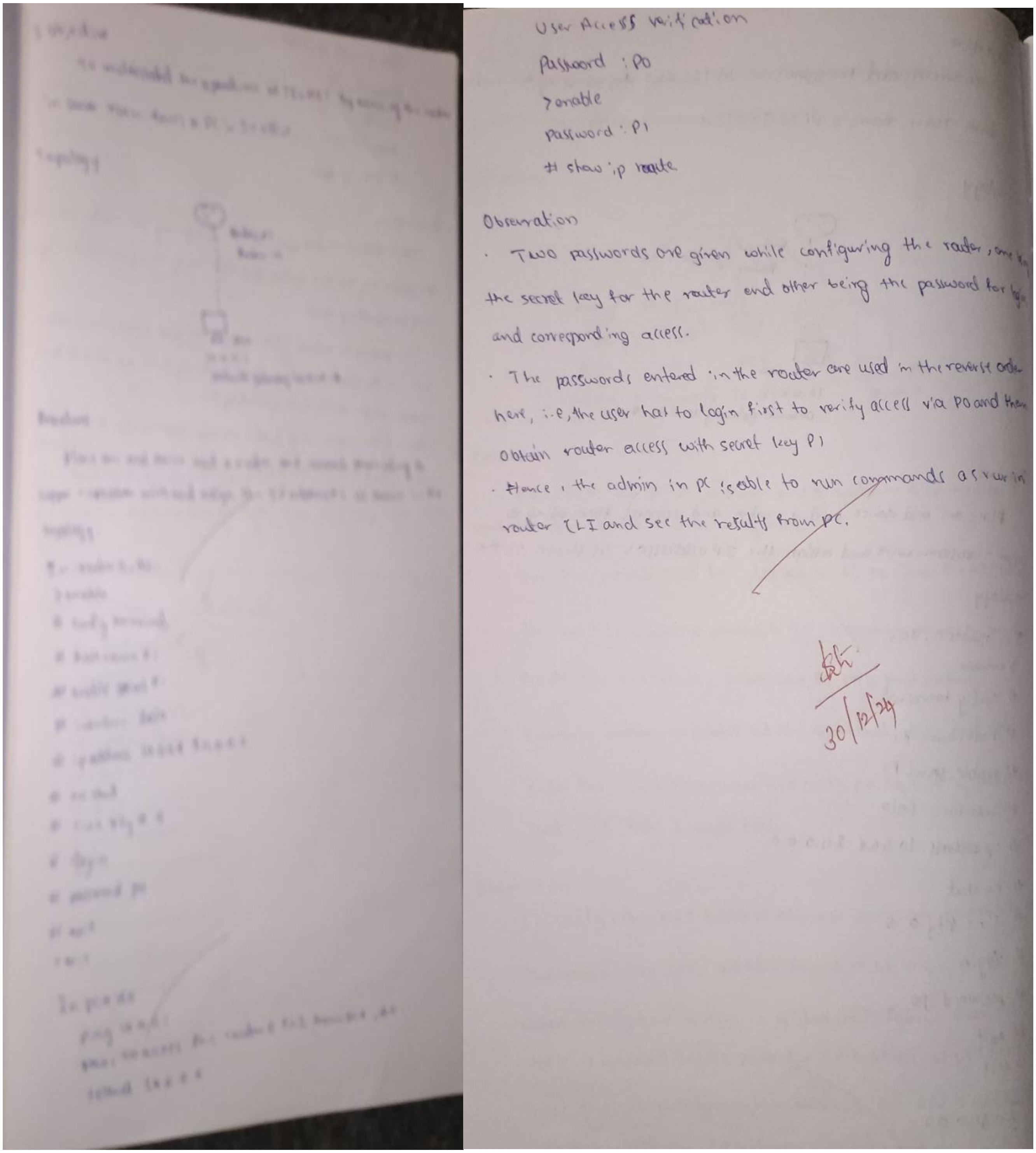
Program 10:

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

Topology :



Procedure and Observations:



User Access verification

```
Password : P0
>enable
password : P1
# show ip route
```

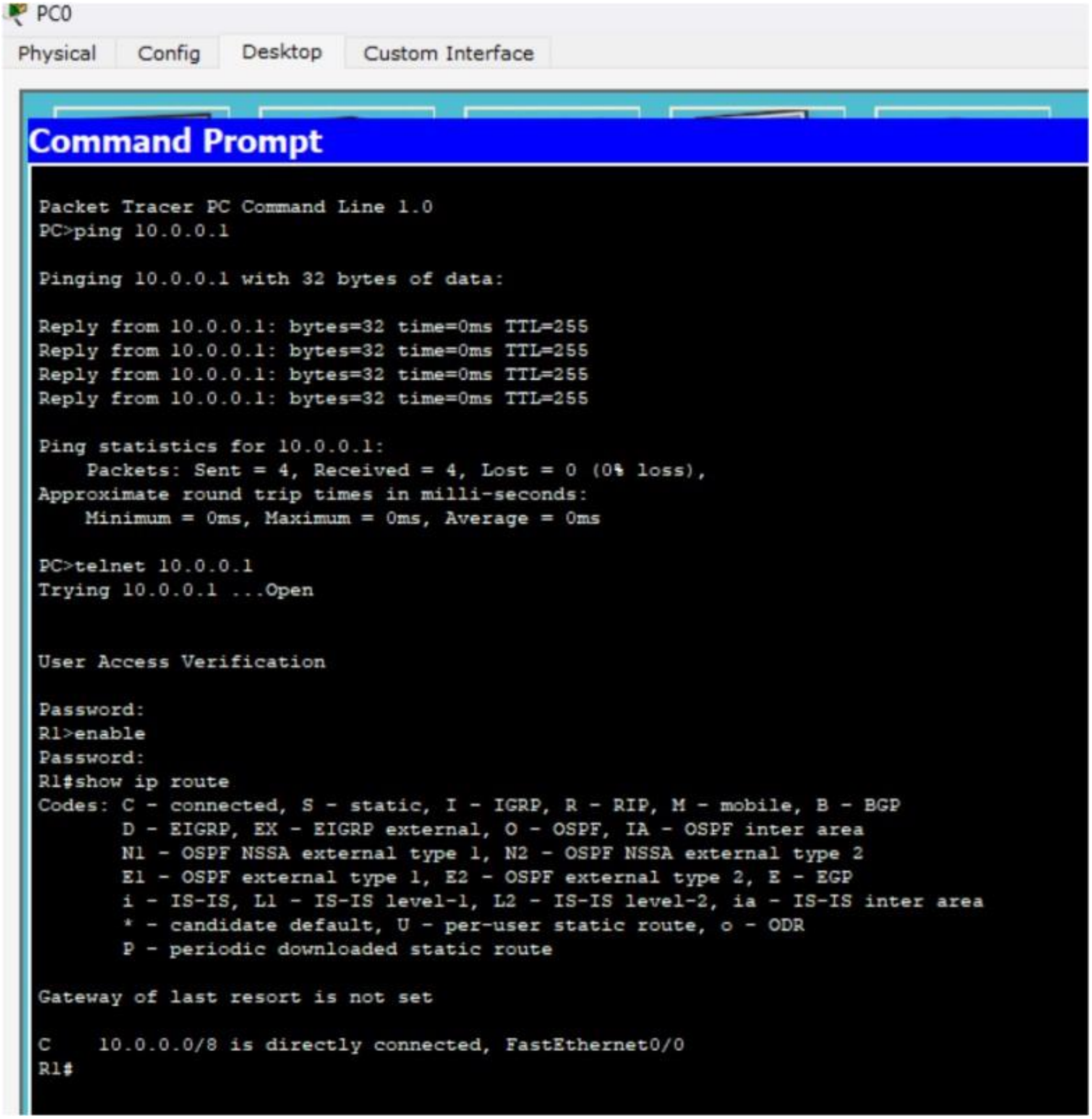
Observation

· Two passwords are given while configuring the router, one being the secret key for the router and other being the password for login and corresponding access.

· The passwords entered in the router are used in the reverse order here, i.e, the user has to login first to verify access via P0 and then obtain router access with secret key P1

· Hence, the admin in PC is able to run commands as run in router CLI and see the results from PC.

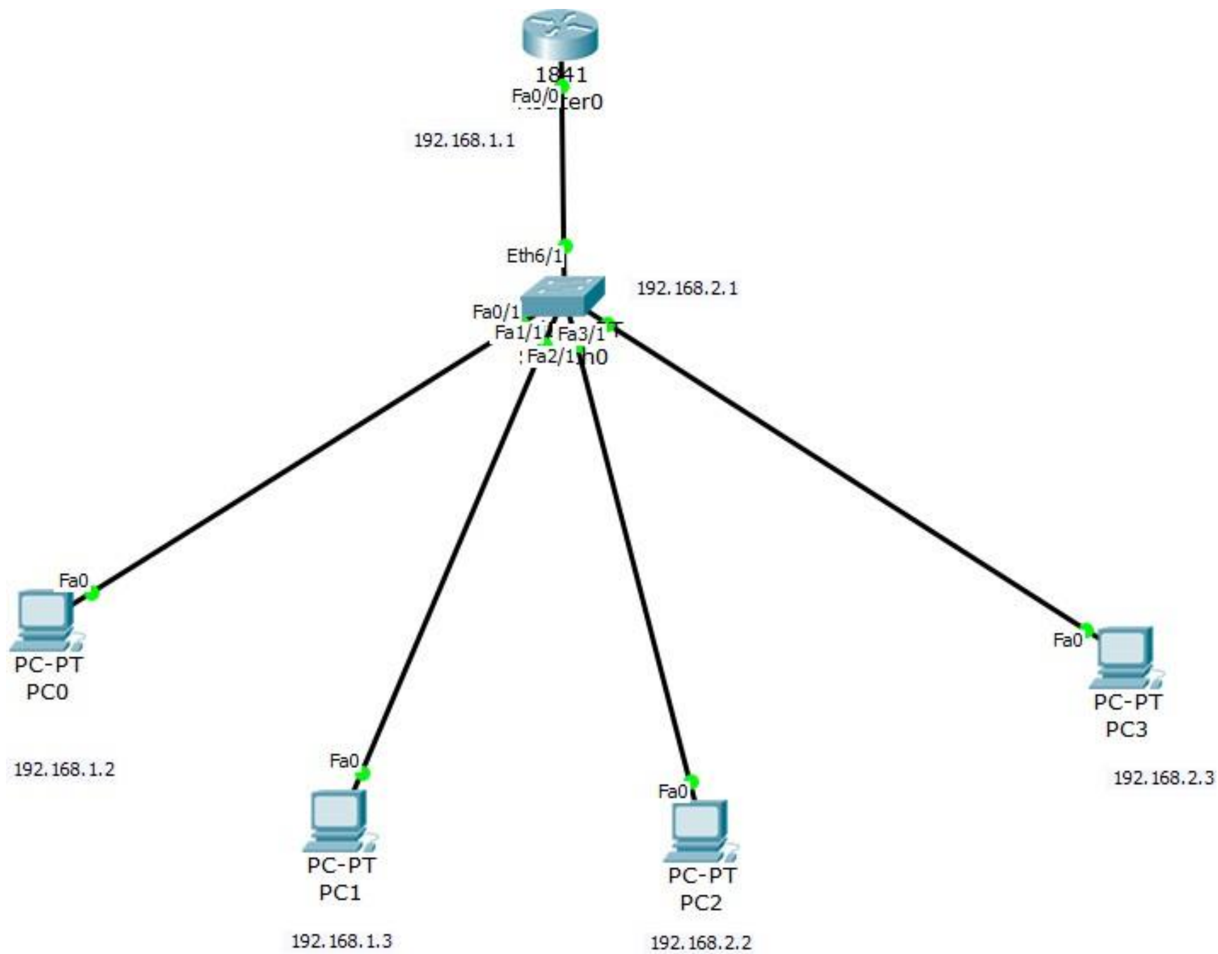
30/12/24


PC0
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=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
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 = 0ms, Average = 0ms
PC>telnet 10.0.0.1
Trying 10.0.0.1 ...Open
User Access Verification
Password:
R1>enable
Password:
R1#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
R1#

<u>Program 11</u>:

Aim : To construct a VLAN and make the PC's communicate among a VLAN.

Topology:

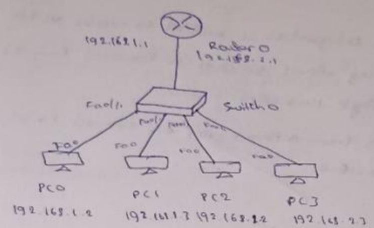


Procedure and Observations:

4 objective

To construct a VLAN and make the PCs communicate among a VLAN.

Topology



Procedure

Place four end devices, a switch and a router and connect the end devices to the switch and the switch to the router using copper straight wires.

Assign IP address to end devices as displayed in the topology. Give VLAN no, name in switch and add.

In router 0, do:

>enable
config terminal
interface fa 0/0
exit
exit
vlan database
vlan 2 name csc ise
exit
config terminal
interface fa 0/0.1
encapsulation dot 1q 2
ip address 192.168.2.1 255.255.255.0
no shut
exit

In switch, do:

Choose VLAN database

Turn port status on for the corresponding ethernet.

enable trunk.

Observation

- Proper trunk configuration is enabled to make VLAN work properly.
- VLAN trunking allows switches to forward frames from all VLAN over a single link called trunk.
- Ping messages from different PCs are observed to be working successfully henceforth.

30/12/24

PC2

Physical | Config | Desktop | Custom Interface

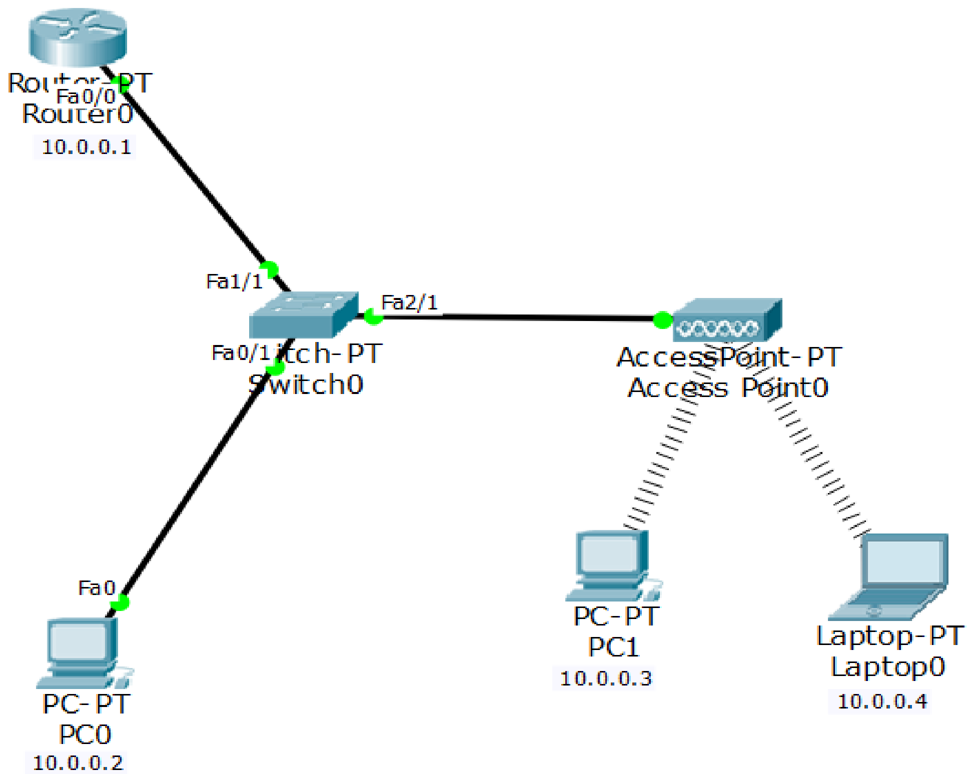
Command Prompt

```
Packet Tracer PC Command Line 1.0
PC>ping 192.168.2.2

Pinging 192.168.2.2 with 32 bytes of data:

Request timed out.
Reply from 192.168.2.2: bytes=32 time=0ms TTL=127
Reply from 192.168.2.2: bytes=32 time=3ms TTL=127
Reply from 192.168.2.2: bytes=32 time=0ms TTL=127

Ping statistics for 192.168.2.2:
    Packets: Sent = 4, Received = 3, Lost = 1 (25% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 3ms, Average = 1ms
```

Program 12:

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

Topology:

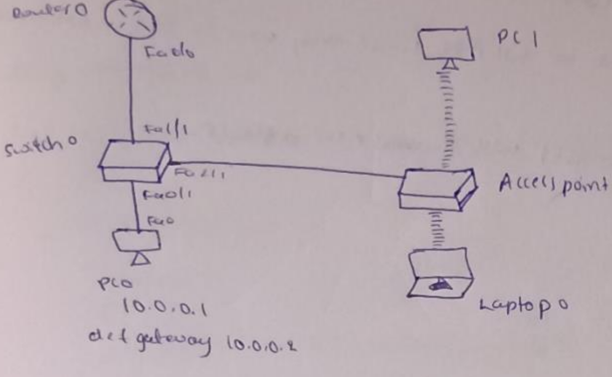


Procedure and Observations:

05 objective

To construct a WLAN and make the nodes communicate wirelessly

Topology:



Procedure

Place three end devices, a switch a router and an access point. connect the end device PC0. access point. connect the end device PC0. access point and the Router0 to switch0 using copper-straight wire.

Assign the IP address as shown in the topology In PC0, do:

- turn the PC off
- Remove the port
- Place the Linksys-WMP300N port to the PC and turn it back on.

Configure Access point 0:

- port status should be set to ON.
- set SSID name as "BMSCE&SECN".
- set channel authentication to 'WEP' and set key as '1234567890'

In PC1 and laptop0, do:

- turn the system off
- Remove the port
- Place the wireless port and turn it back on

In config, do:

- Set the same SSID
- Set authentication to WEP and enter same key

Ping from different devices and observe the transmissions.

Observation

- After the setup of PC1 and laptop 0, wireless connections with dashed lines were observed in connection with access point 0, indicating successfull wireless connections.
- Devices could connect to WLAN since they were in the network range.
- Signal strength decreases with increase in distance.

30/12/24.

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=30ms TTL=128
Reply from 10.0.0.3: bytes=32 time=9ms TTL=128
Reply from 10.0.0.3: bytes=32 time=4ms TTL=128
Reply from 10.0.0.3: bytes=32 time=7ms 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 = 4ms, Maximum = 30ms, Average = 12ms

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=21ms TTL=128
Reply from 10.0.0.4: bytes=32 time=12ms TTL=128
Reply from 10.0.0.4: bytes=32 time=9ms TTL=128
Reply from 10.0.0.4: bytes=32 time=6ms TTL=128

Ping statistics for 10.0.0.4:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 6ms, Maximum = 21ms, Average = 12ms
```

CYCLE - 2

Program 13:

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

```
#include <iostream>
#include <string.h>
using namespace std;

int crc(char *ip, char *op, char *poly, int mode)
{
strcpy(op, ip);
if (mode) {
   for (int i = 1; i < strlen(poly); i++)
        strcat(op, "0");
}
/* Perform XOR on the msg with the selected polynomial */
for (int i = 0; i < strlen(ip); i++) {
     if (op[i] == '1') {
          for (int j = 0; j < strlen(poly); j++) {
               if (op[i + j] == poly[j])
                        op[i + j] = '0';
    else
           op[i + j] = '1';
}}}
/* check for errors. return 0 if error detected */
for (int i = 0; i < strlen(op); i++)
       if (op[i] == '1') return 0;
return 1;
}

int main(){
     char ip[50], op[50], recv[50];
     /* x 16 + x12 + x5 + 1 */
      char poly[] = "10001000000100001";
      cout << "Enter the input message in binary"<< endl;
      cin >> ip;
      crc(ip, op, poly, 1);
      cout << "The transmitted message is: " << ip << op + strlen(ip) << endl;
      cout << "Enter the received message in binary" << endl;
      cin >> recv;
      if (crc(recv, op, poly, 0))
           cout << "No error in data" << endl;
     else
           cout << "Error in data transmission has occurred" << endl;
     return 0;
}
```

Observations:

Output 1

Enter the input message in binary

111101

The transmitted message is 1111101101010111100111010

Enter the recieved message in binary

111101

No error in data

Output 2

Enter the input

111101

The transmitted message is 11111001101010111100111010

Enter the recieved message in binary

1110

Error in data transmission has occured.

30/12/24

<u>Program 14:</u>

Aim: Write a program for congestion control using Leaky bucket algorithm.

Algorithm:
1. Start
2. Set the bucket size or the buffer size.
3. Set the output rate.
4. Transmit the packets such that there is no overflow.
5. Repeat the process of transmission until all packets are transmitted.
   (Reject packets whosesize is greater than the bucket size.)
6. Stop

Code:
```
#include <iostream>
#include <string.h>
using namespace std;

#include<stdio.h>
#include<stdlib.h>
#include<unistd.h>
#define NOF_PACKETS 10
int rand(int a){
      int rn = (random() % 10) % a;
      return rn == 0 ? 1 : rn;
}
int main() {
   int packet_sz[NOF_PACKETS], i, clk, b_size, o_rate, p_sz_rm=0, p_sz, p_time, op;
   for(i = 0; i<NOF_PACKETS; ++i)
         packet_sz[i] = rand(6) * 10;
   for(i = 0; i<NOF_PACKETS; ++i)
        printf("\npacket[%d]:%d bytes\t", i, packet_sz[i]);
   printf("\nEnter the Output rate:");
   scanf("%d", &o_rate);
   printf("Enter the Bucket Size:");
   scanf("%d", &b_size);
   for(i = 0; i<NOF_PACKETS; ++i){
        if( (packet_sz[i] + p_sz_rm) > b_size)
             if(packet_sz[i] > b_size)/*compare the packet size with bucket size*/
                  printf("\n\nIncoming packet size (%dbytes) is Greater than bucket capacity
                                  (%dbytes)-PACKET REJECTED", packet_sz[i], b_size);
             else
                  printf("\n\nBucket capacity exceeded-PACKETS REJECTED!!");
       else {
            p_sz_rm += packet_sz[i];
            printf("\n\nIncoming Packet size: %d", packet_sz[i]);
            printf("\nBytes remaining to Transmit: %d", p_sz_rm);
            p_time = rand(4) * 10;
```

```
 printf("\nTime left for transmission: %d units", p_time);
for(clk = 10; clk <= p_time; clk += 10) {
     sleep(1);
      if(p_sz_rm) {
            if(p_sz_rm <= o_rate)/*packet size remaining comparing with output rate*/
                 op = p_sz_rm, p_sz_rm = 0;
            else
                 op = o_rate, p_sz_rm -= o_rate;
             printf("\nPacket of size %d Transmitted", op);
             printf(" --- Bytes Remaining to Transmit: %d", p_sz_rm);
      }
      else {
         printf("\nTime left for transmission: %d units", p_time-clk);
         printf("\nNo packets to transmit!!");
}}}}
return 0;
}
```

OUTPUT:
packet[0]:30 bytes
packet[1]:10 bytes
packet[2]:10 bytes
packet[3]:50 bytes
packet[4]:30 bytes
packet[5]:50 bytes
packet[6]:10 bytes
packet[7]:20 bytes
packet[8]:30 bytes
packet[9]:10 bytes
Enter the Output rate:100
Enter the Bucket Size:50
Incoming Packet size: 30
Bytes remaining to Transmit: 30
Time left for transmission: 20 units
Packet of size 30 Transmitted --- Bytes Remaining to Transmit: 0
Time left for transmission: 0 units
No packets to transmit!!

Incoming Packet size: 10
Bytes remaining to Transmit: 10
Time left for transmission: 30 units
Packet of size 10 Transmitted --- Bytes Remaining to Transmit: 0
Time left for transmission: 10 units
No packets to transmit!!
Time left for transmission: 0 units
No packets to transmit!!
Incoming Packet size: 10
Bytes remaining to Transmit: 10Time left for transmission: 10 units
Packet of size 10 Transmitted --- Bytes Remaining to Transmit: 0

Incoming Packet size: 50
Bytes remaining to Transmit: 50
Time left for transmission: 10 units
Packet of size 50 Transmitted --- Bytes Remaining to Transmit: 0

Incoming Packet size: 30
Bytes remaining to Transmit: 30
Time left for transmission: 30 units
Packet of size 30 Transmitted --- Bytes Remaining to Transmit: 0
Time left for transmission: 10 units
No packets to transmit!!
Time left for transmission: 0 units
No packets to transmit!!

Incoming Packet size: 50

Bytes remaining to Transmit: 50
Time left for transmission: 20 units
Packet of size 50 Transmitted --- Bytes Remaining to Transmit: 0
Time left for transmission: 0 units
No packets to transmit!!

Incoming Packet size: 10
Bytes remaining to Transmit: 10
Time left for transmission: 10 units
Packet of size 10 Transmitted --- Bytes Remaining to Transmit: 0
Incoming Packet size: 20
Bytes remaining to Transmit: 20
Time left for transmission: 20 units
Packet of size 20 Transmitted --- Bytes Remaining to Transmit: 0
Time left for transmission: 0 units
No packets to transmit!!

Incoming Packet size: 30
Bytes remaining to Transmit: 30
Time left for transmission: 20 units
Packet of size 30 Transmitted --- Bytes Remaining to Transmit: 0
Time left for transmission: 0 units
No packets to transmit!!
Incoming Packet size: 10
Bytes remaining to Transmit: 10
Time left for transmission: 20 units
Packet of size 10 Transmitted --- Bytes Remaining to Transmit: 0
Time left for transmission: 0 units
No packets to transmit!!

<u>Program 15:</u>

Aim: 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.

Algorithm:
Client Side
1. Start.
2. Create a socket using the socket() system call.
3. Connect the socket to the server's address using the connect() system call.
4. Send the filename of the required file using the send() system call.
5. Read the contents of the file sent by the server using the recv() system call.
6. Stop.

Code:

```
#include <unistd.h>
int main()
{
    int soc, n;
    char buffer[1024], fname[50];
    struct sockaddr_in addr;
    /* socket creates an endpoint for communication and returns a file descriptor */
    soc = socket(PF_INET, SOCK_STREAM, 0);
    /*
    * sockaddr_in is used for ip manipulation
    * we define the port and IP for the connection.
    */
    addr.sin_family = AF_INET;
    addr.sin_port = htons(7891);
    addr.sin_addr.s_addr = inet_addr("127.0.0.1");
    /* keep trying to establish connection with server */
    while(connect(soc, (struct sockaddr *) &addr, sizeof(addr))) ;
        printf("\nClient is connected to Server");
    printf("\nEnter file name: ");
    scanf("%s", fname);
     /* send the filename to the server */
    send(soc, fname, sizeof(fname), 0);
    printf("\nRecieved response\n");0
     /* keep printing any data received from the server */
     while ((n = recv(soc, buffer, sizeof(buffer), 0)) > 0)
         printf("%s", buffer);
     return 0;
}
```

Algorithm:
Server Side
1. Start.
2. Create a socket using socket() system call.
3. Bind the socket to an address using bind() system call.
4. Listen to the connection using listen() system call.
5. accept connection using accept()
6. Receive filename and transfer contents of file with client.
7. Stop.

Code:
```
#include <stdio.h>
#include <arpa/inet.h>
#include <fcntl.h>
#include <unistd.h>
int main()
{
     int welcome, new_soc, fd, n;
     char buffer[1024], fname[50];
     struct sockaddr_in addr;
     welcome = socket(PF_INET, SOCK_STREAM, 0);
     addr.sin_family = AF_INET;
     addr.sin_port = htons(7891);
     addr.sin_addr.s_addr = inet_addr("127.0.0.1");
     bind(welcome, (struct sockaddr *) &addr, sizeof(addr));
     printf("\nServer is Online");
     /* listen for connections from the socket */
     listen(welcome, 5);
     /* accept a connection, we get a file descriptor */
    new_soc = accept(welcome, NULL, NULL);
    /* receive the filename */
    recv(new_soc, fname, 50, 0);
    printf("\nRequesting for file: %s\n", fname);
    /* open the file and send its contents */
    fd = open(fname, O_RDONLY);
    if (fd < 0)
         send(new_soc, "\nFile not found\n", 15, 0);
    else
        while ((n = read(fd, buffer, sizeof(buffer))) > 0)
   send(new_soc, buffer, n, 0);
   printf("\nRequest sent\n");
   close(fd);
   return 0;
}
```

OUTPUT:

Server is Online.
Requesting for file : test.txt
Request sent.

Client is connected to server
Enter file name : test.txt
Received Response
Hello World.

Program 16:

Aim: 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.

Code:

```
// server program for udp connection
#include <stdio.h>
#include <strings.h>
#include <sys/types.h>
#include <arpa/inet.h>
#include <sys/socket.h>
#include<netinet/in.h>
#define PORT 5000
#define MAXLINE 1000
// Driver code
int main()
{
    char buffer[100];
    char *message = "Hello Client";
    int listenfd, len;
    struct sockaddr_in servaddr, cliaddr;
    bzero(&servaddr, sizeof(servaddr));
    // Create a UDP Socket
    listenfd = socket(AF_INET, SOCK_DGRAM, 0);
    servaddr.sin_addr.s_addr = htonl(INADDR_ANY);
    servaddr.sin_port = htons(PORT);
    servaddr.sin_family = AF_INET;
    // bind server address to socket descriptor
    bind(listenfd, (struct sockaddr*)&servaddr, sizeof(servaddr));
    //receive the datagram
    len = sizeof(cliaddr);
    int n = recvfrom(listenfd, buffer, sizeof(buffer), 0, (struct sockaddr*)&cliaddr,&len);
    //receive message from server
    buffer[n] = '\0';
    puts(buffer);
    // send the response
    sendto(listenfd, message, MAXLINE, 0,(struct sockaddr*)&cliaddr, sizeof(cliaddr));
}


// udp client driver program
#include <stdio.h>
#include <strings.h>
#include <sys/types.h>
#include <arpa/inet.h>
#include <sys/socket.h>
#include<netinet/in.h>
```

```
#include<unistd.h>
#include<stdlib.h>
#define PORT 5000
#define MAXLINE 1000
// Driver code
int main()
{
    char buffer[100];
    char *message = "Hello Server";
    int sockfd, n;
    struct sockaddr_in servaddr;
    // clear servaddr
    bzero(&servaddr, sizeof(servaddr));
    servaddr.sin_addr.s_addr = inet_addr("127.0.0.1");
    servaddr.sin_port = htons(PORT);
    servaddr.sin_family = AF_INET;
    // create datagram socket
    sockfd = socket(AF_INET, SOCK_DGRAM, 0);
    // connect to server
   if(connect(sockfd, (struct sockaddr *)&servaddr, sizeof(servaddr)) < 0) {
        printf("\n Error : Connect Failed \n");
        exit(0);
    }
    // request to send datagram
    // no need to specify server address in sendto
    // connect stores the peers IP and port
    sendto(sockfd, message, MAXLINE, 0, (struct sockaddr*)NULL, sizeof(servaddr));
    // waiting for response
    recvfrom(sockfd, buffer, sizeof(buffer), 0, (struct sockaddr*)NULL, NULL);
    puts(buffer);
    // close the descriptor
    close(sockfd);
}
```

Output:

```
//Server output
Server is Online.
Hello Server

//Client Output
Hello Client
```