VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"JnanaSangama", Belgaum -590014, Karnataka.

LAB RECORD

Computer Network Lab (22CS4PCCON)

Submitted by

MANDAAR ADARSH (1BM22CS358)

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 (22CS4PCCON)"** carried out by **MANDAAR ADARSH (1BM22CS358),** 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

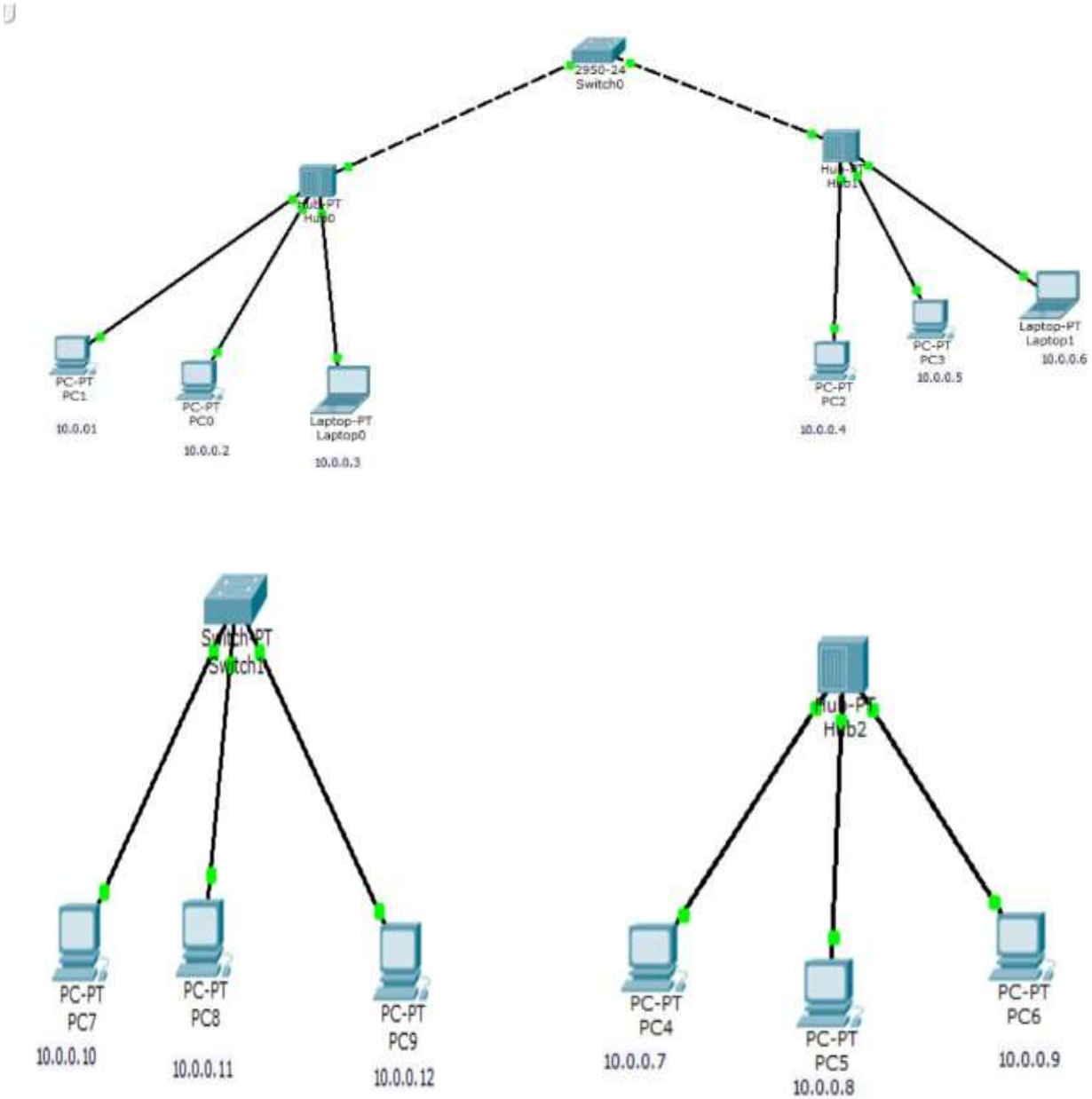
Index

Github Link : https://github.com/TanmayBj23/CN_LAB

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:

LAB-1:

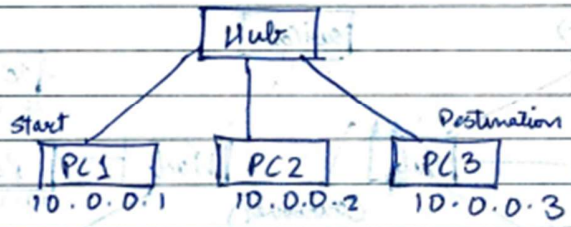
PAGE NO:
DATE: / /

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

→ AIM: Simulating the transmission of simple PDU using hub and switch as connecting devices.

→ Devices used: Hub, switch & end devices.

→ Topology 1: Hub & 3 end devices



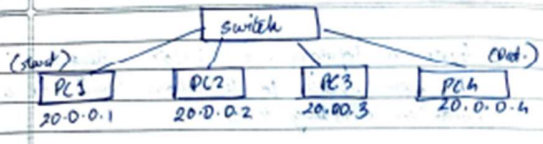
→ Procedure & Observation:

i) connect PC1, PC2, PC3 to hub through cables
ii) Assign IP address to each devices.
iii) Select a single PDU, PC1 as start & PC3 as destination

During simulation, PC3 will recive message and ~~PC1~~ PC1 acknowledge the same.

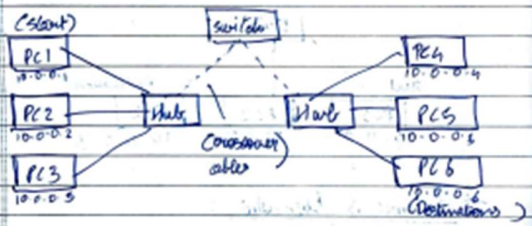
→ Topology 2: Switch and 4 devices

P.T.O.



i) Connect devices to switch with mentioned IP.
ii) Select simple PDU, PC1 as Start, PC4 as Dest.
iii) Connect with cables. The msg is sent from PC1 to PC4 and acknowledgment is sent.

Topology 3:- Switch, Hubs, and End Devices



i) Connect 3 end devices to hub1 with mentioned IP, further connect to switch.
ii) ~~Hub~~ Hub and switches connected through crossover cable
iii) Then connect switch to similar hub and end device config.
iv) Take 3 PCs as Start and any one of the other 3 as destination
v) Demonstrate simulation and analyse the flow of message and acknowledgement

→ Difference between hub, switch:-

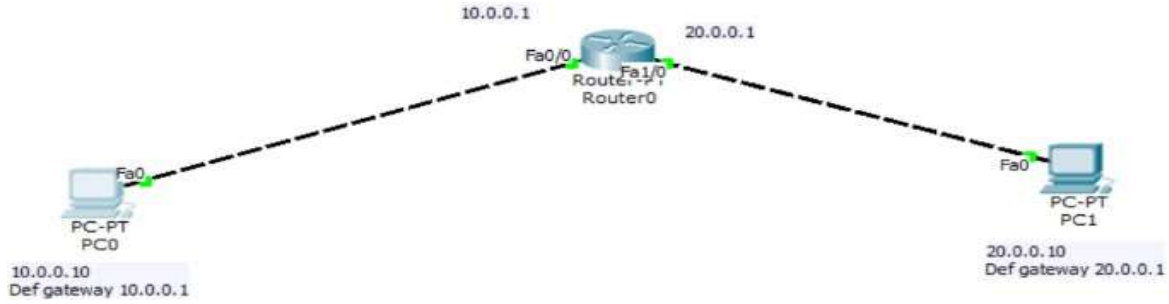
- Hub operates at physical layer of OSI model.
- Broadcasts packets to all connected devices regardless of intended recipients.
- Less efficient and supports lower speeds.

- Switch operates at data link layer
- Broadcasts to specific devices which data is intended
- More efficient and supports higher speeds.

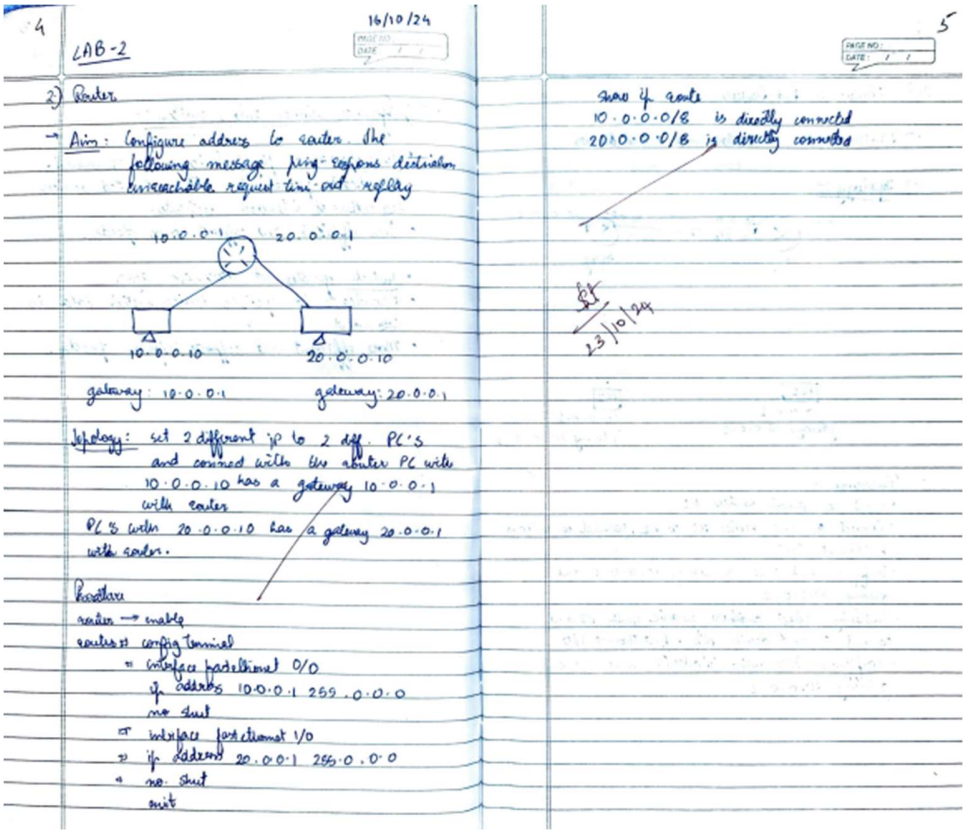
Program 2 :

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:



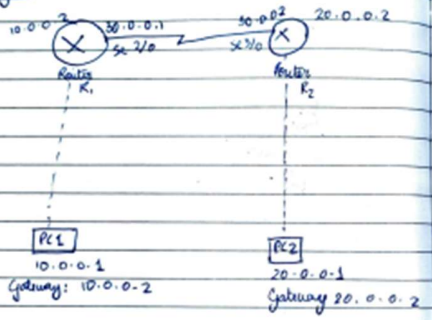
6

LAB-5

2.2) 2Routers & End Devices

→ Devices used: 2 routers and 2 end devices

→ Topology :-



→ Procedure :-

- Select a generic router R1
- Connect an end device PC1 to R1, parallel connection fastethernet 0/0
- Configure PC1 with IP address 10.0.0.1 and gateway 10.0.0.2
- Similarly select another generic router R2 and connect an end device PC2 - fastethernet 1/0
- Configure PC2 with IP address 20.0.0.1 and gateway 20.0.0.2

7

→ Go to R1 CLI

Router > enable
Router # config terminal
Router(config)# ipAddress 10.0.0.2 255.0.0.0
Router(config)# no shut
"Interface FastEthernet 0/0, changed state to up"

→ Similarly select router R2, goto CLI

Router > enable
Router # config terminal
Router(config)# interface FastEthernet 1/0
Router(config)# ip address 20.0.0.2 255.0.0.0
Router(config-if)# no shut
"Interface FastEthernet 1/0 changed to state up".

→ Connection is established between Router and end devices.

→ Connect router R1 with R2 using serial cable

- Select router R1 & go to CLI

Router(config)# interface serial 2/0
Router(config)# ip address 30.0.0.1 255.0.0.0
Router(config-if)# no shut

- Select router R2 and go to CLI

Router(config)# interface serial 3/0
Router(config)# ip address 30.0.0.1 255.0.0.0
Router(config-if)# no shut

"Interface serial 2/0 changed state to up".

DATE : / /

Observations :-

- After setting up the mentioned topology Now try to ping PC2 with PC0
  Open CMD of PC1 and ping 20.0.0.1
  → Destination host unreachable
  Packets sent: 4 received 0 lost 4 Loss: 100%

→ Also PC1 is only ping with R1

Ping 30.0.0.1 → successful
Packets sent 4: received: 4 lost: 0 loss: 0%

Hence even though routers are connected serially and devices are unable to ping each other.

23/10/24

PC0

Physical | Config | Desktop | Custom Interface

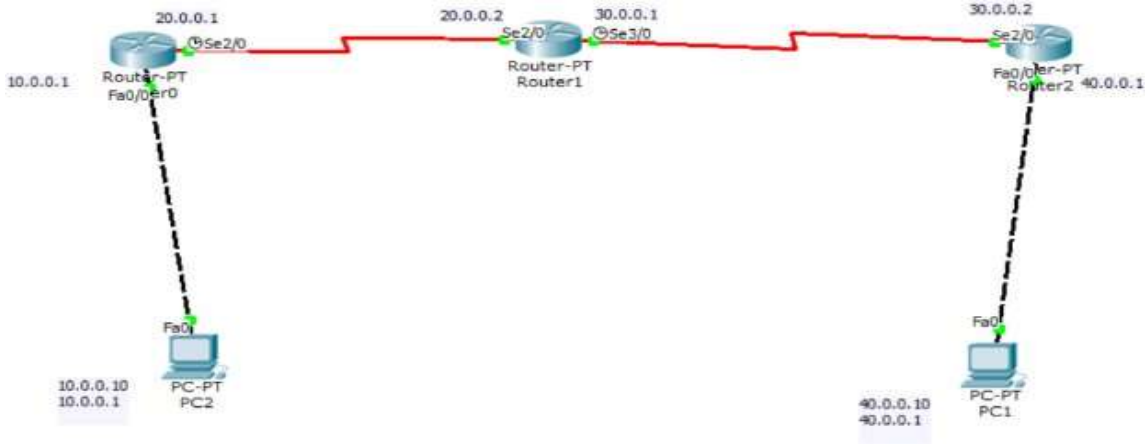
Command Prompt

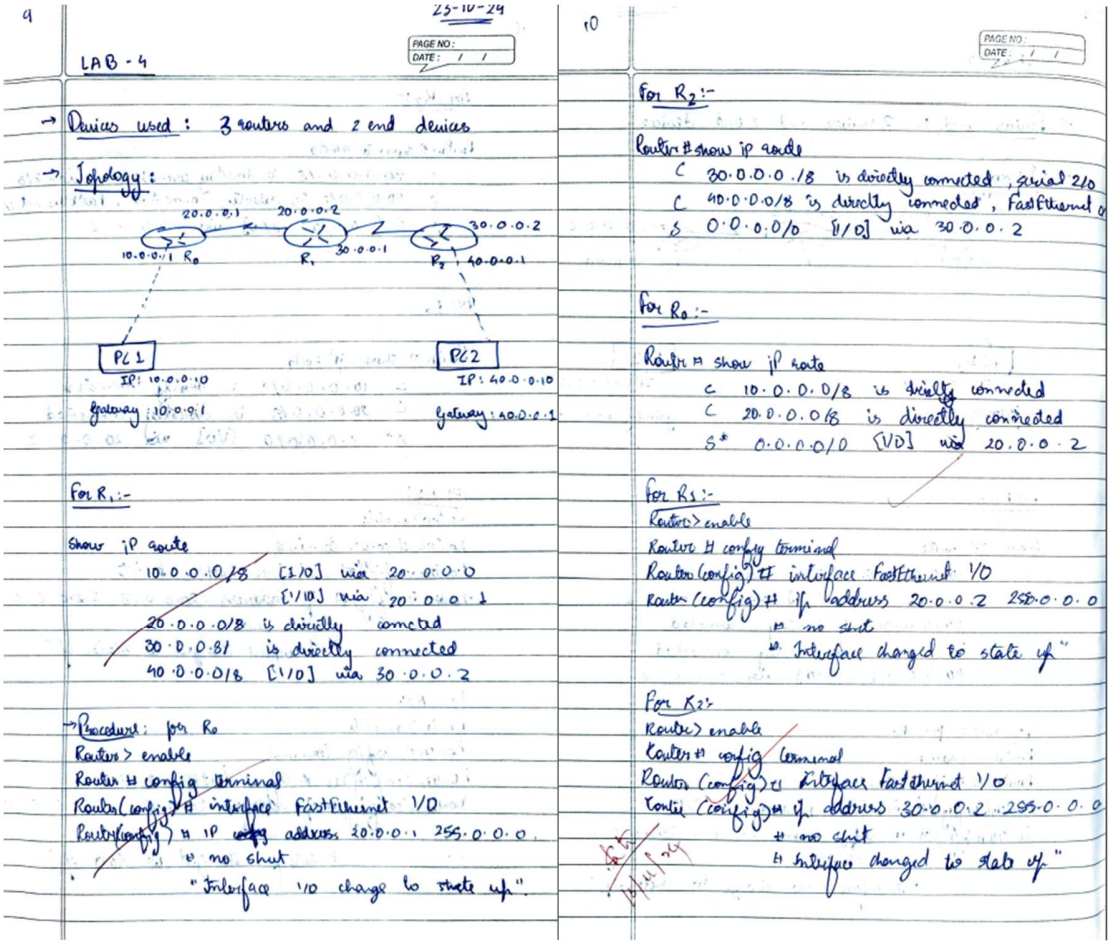
```
Pinging 20.0.0.10 with 32 bytes of data:

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 :





9

25-10-24

LAB - 4

→ Devices used : 3 routers and 2 end devices

→ Topology :

For R1 :-

show ip route

10.0.0.0/8 [1/0] via 20.0.0.0
[1/0] via 20.0.0.1
20.0.0.0/8 is directly connected
30.0.0.8/1 is directly connected
40.0.0.0/8 [1/0] via 30.0.0.2

→ Procedure : for R0

Router > enable
Router # config terminal
Router(config)# interface FastEthernet 1/0
Router(config)# ip config address 10.0.0.1 255.0.0.0
no shut
"Interface 1/0 change to state up".

10

For R2 :-

Router # show ip route
C 30.0.0.0/8 is directly connected, Serial 2/0
C 40.0.0.0/8 is directly connected, FastEthernet 0
S 0.0.0.0/0 [1/0] via 30.0.0.2

For R0 :-

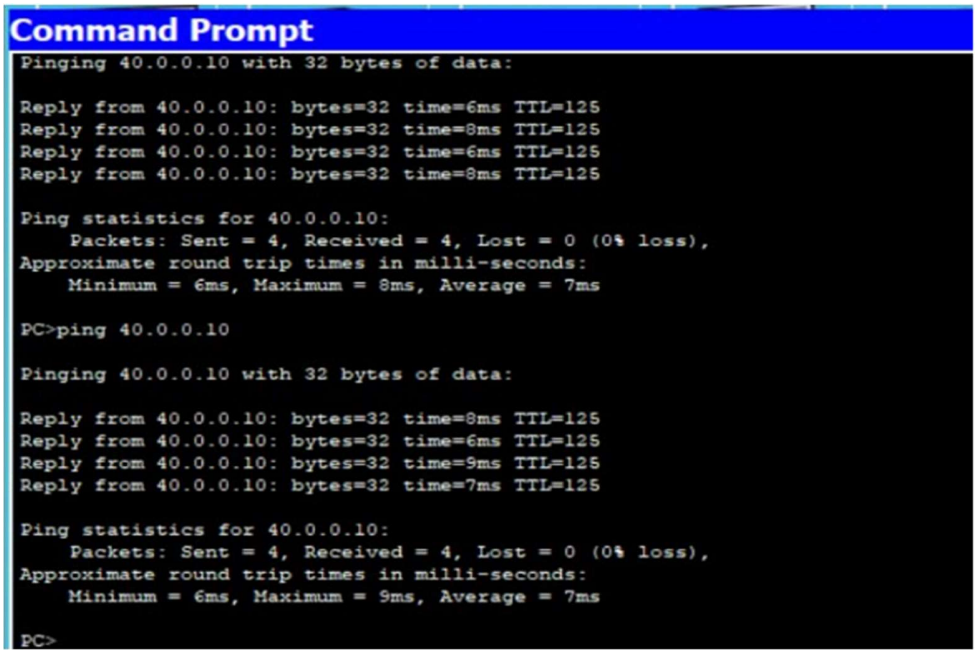
Router # show ip route
C 10.0.0.0/8 is directly connected
C 20.0.0.0/8 is directly connected
S* 0.0.0.0/0 [1/0] via 20.0.0.2

For R1 :-

Router > enable
Router # config terminal
Router(config)# interface FastEthernet 1/0
Router(config)# ip address 20.0.0.2 255.0.0.0
no shut
Interface changed to state up"

For R2 :-

Router > enable
Router # config terminal
Router(config)# interface FastEthernet 1/0
Router(config)# ip address 30.0.0.2 255.0.0.0
no shut
Interface changed to state up"

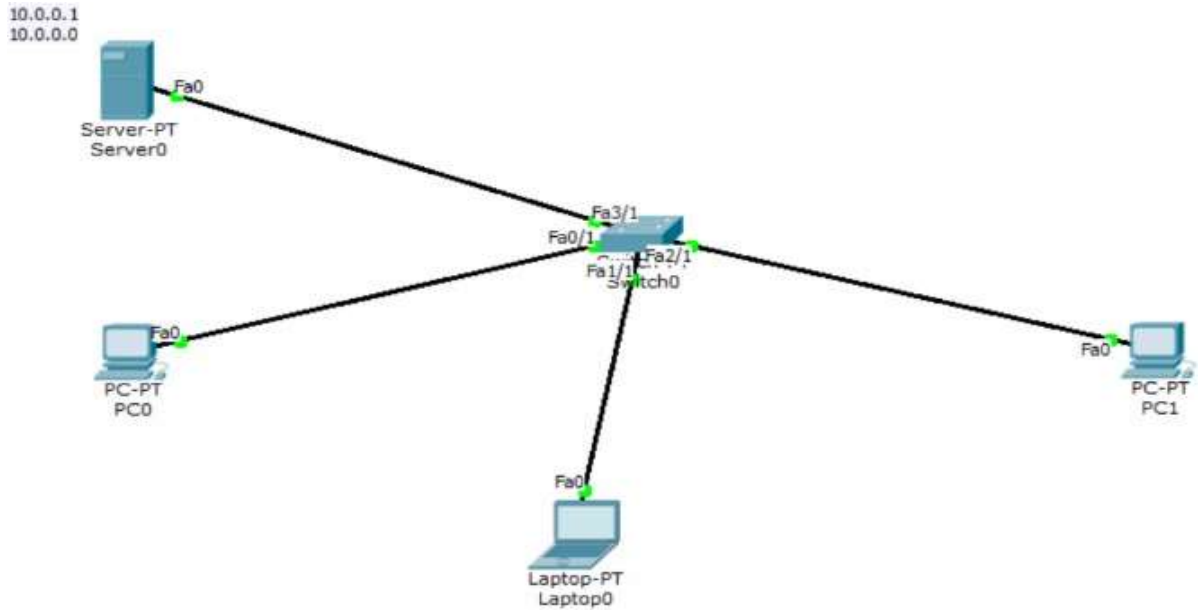

Command Prompt
Pinging 40.0.0.10 with 32 bytes of data:
Reply from 40.0.0.10: bytes=32 time=6ms TTL=125
Reply from 40.0.0.10: bytes=32 time=8ms TTL=125
Reply from 40.0.0.10: bytes=32 time=6ms TTL=125
Reply from 40.0.0.10: bytes=32 time=8ms TTL=125
Ping statistics for 40.0.0.10:
Packets: Sent = 4, Received = 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>

Program 4:

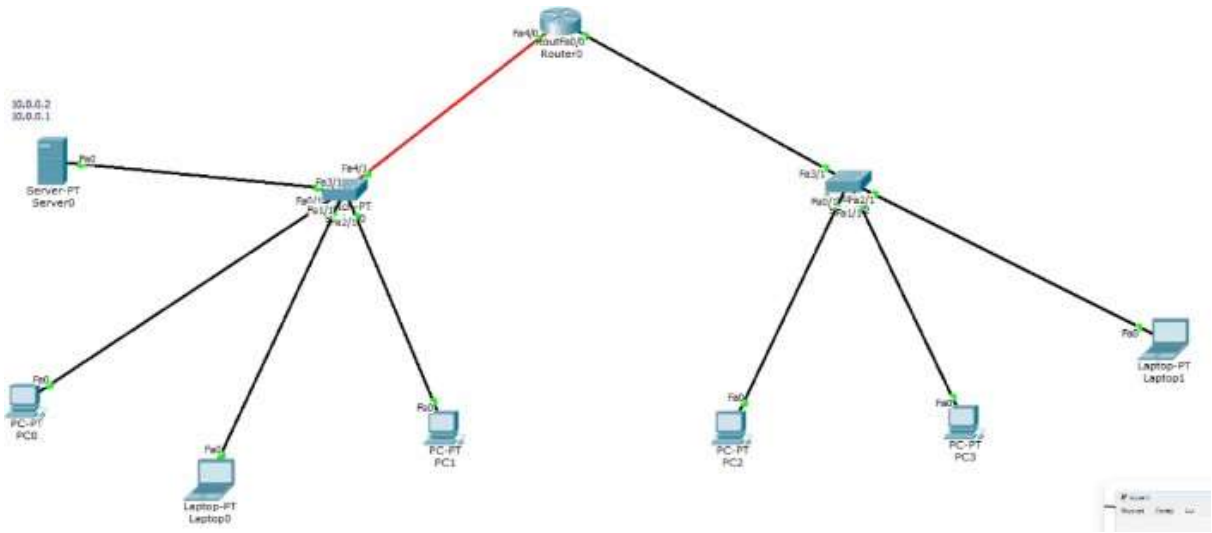
Aim: Configure DHCP within a LAN and outside LAN.

Topology:

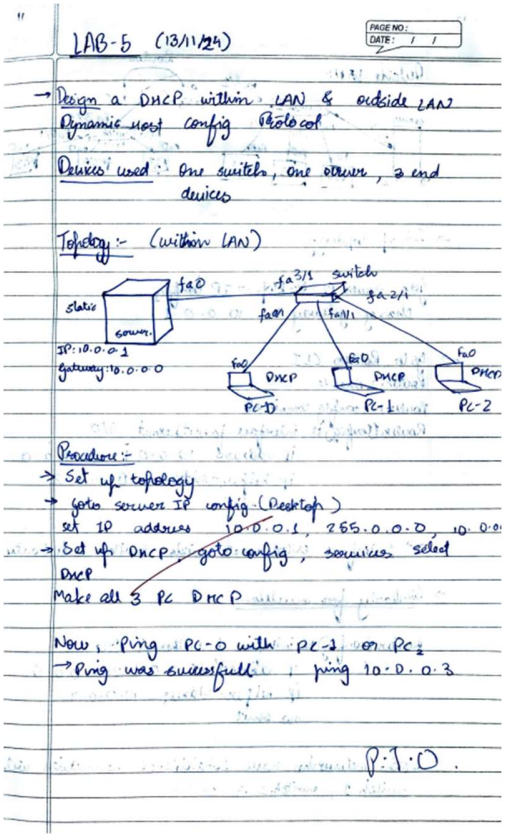
Within LAN



Outside LAN



Procedure and Observation:



LAB-5 (13/11/24)

→ Design a DHCP within LAN & outside LAN
Dynamic host config Protocol

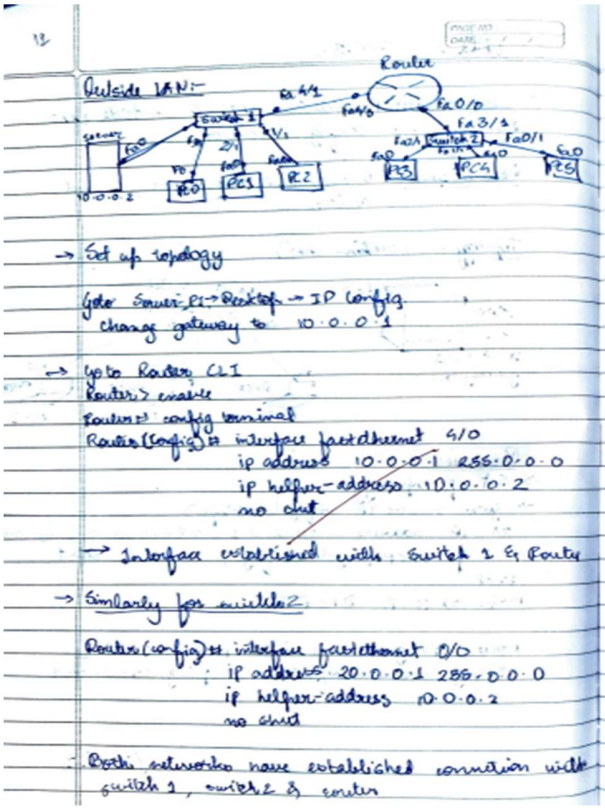
Devices used: One switch, one server, 2 end devices

Topology:- (within LAN)

Procedure:-
→ Set up topology
→ goto server IP config (Desktop)
set IP address 10.0.0.1, 255.0.0.0, 10.0.0.
→ Set up DHCP, goto config, services select DHCP
Make all 3 PC DHCP

Now, Ping PC-0 with PC-1 or PC-2
→ Ping was successfull : ping 10.0.0.3

P.T.O.



Router

Outside LAN:-

→ Set up topology

goto Server PC → Desktop → IP Config
change gateway to 10.0.0.1

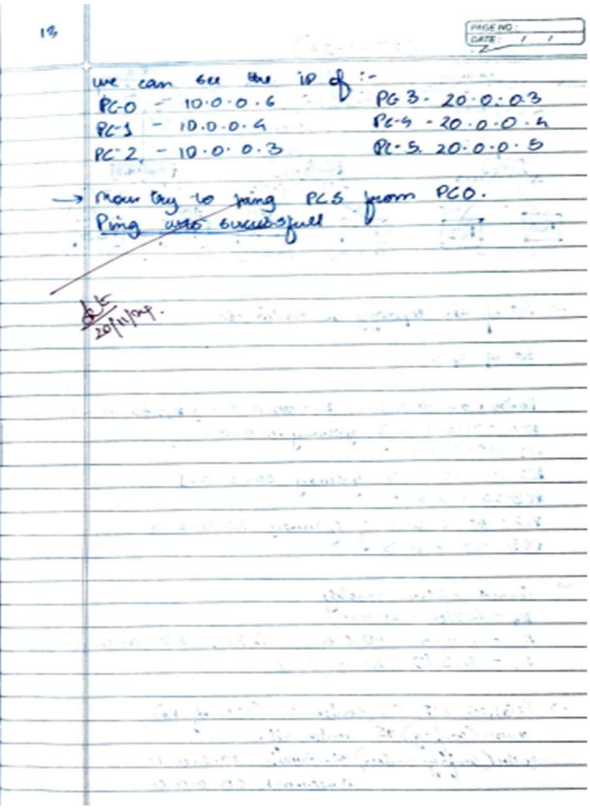
→ goto Router CLI
Router> enable
Router# config terminal
Router(config)# interface fastethernet 4/0
ip address 10.0.0.1 255.0.0.0
ip helper-address 10.0.0.2
no shut

→ Interface established with Switch 1 & Router

→ Similarly for switch 2

Router(config)# interface fastethernet 0/0
ip address 20.0.0.1 255.0.0.0
ip helper-address 10.0.0.2
no shut

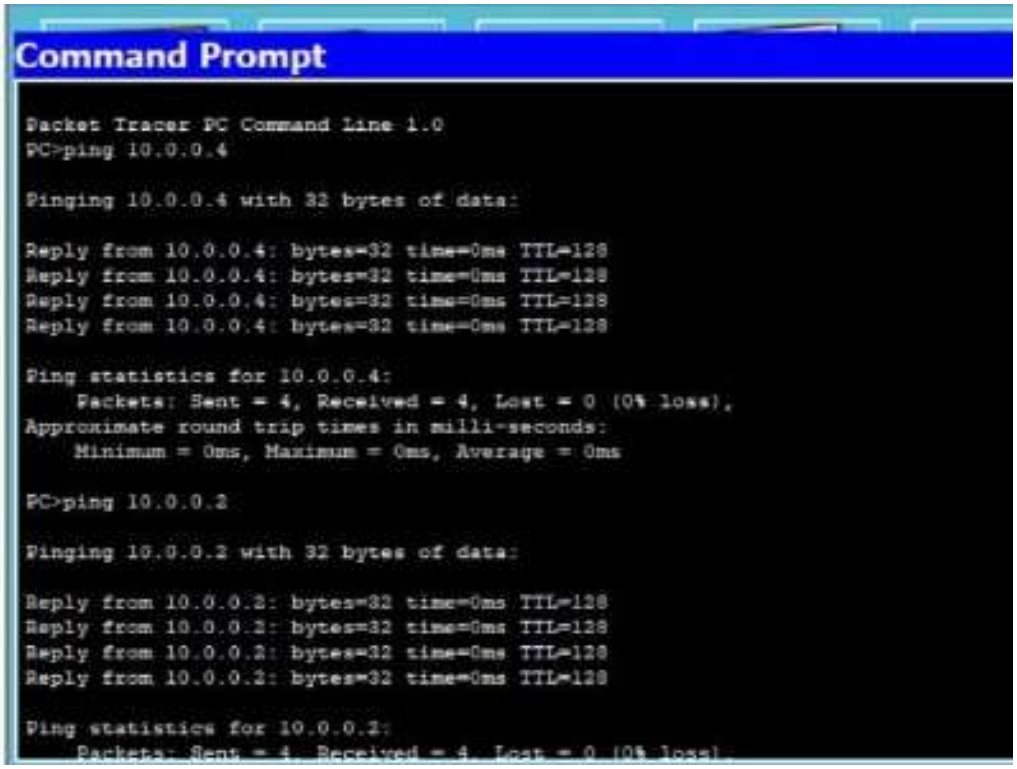
Both networks have established connection with switch 1, switch 2 & router



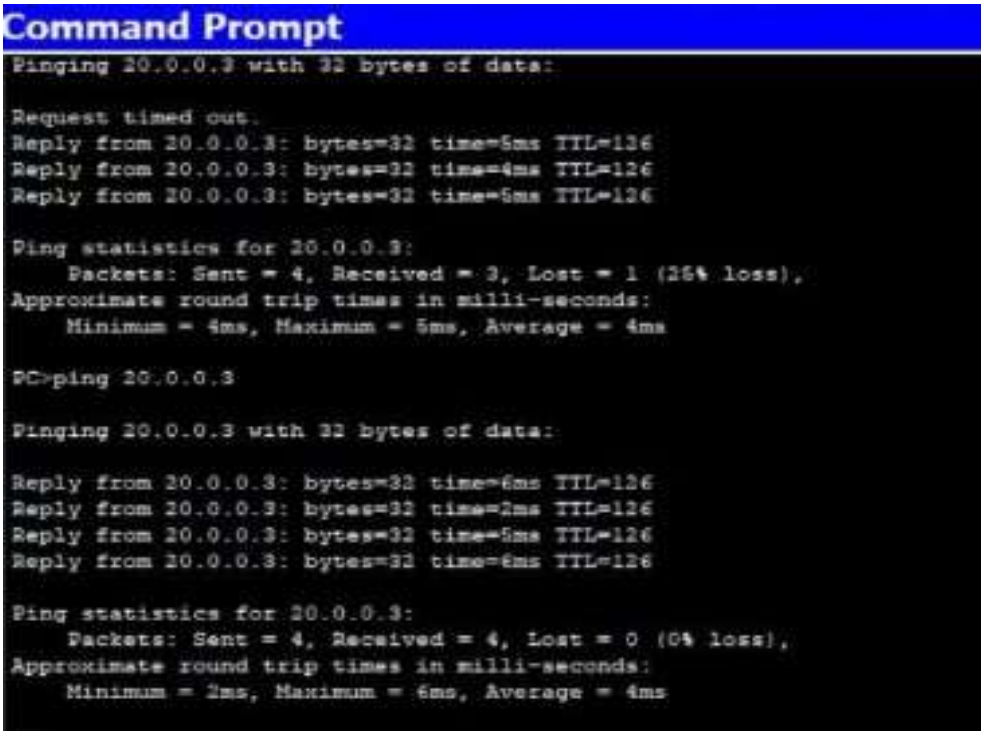
We can see the ip of :-

PC-0 - 10.0.0.6
PC-1 - 10.0.0.4
PC-2 - 10.0.0.3
PC-3 - 20.0.0.3
PC-4 - 20.0.0.4
PC-5 - 20.0.0.5

→ Now try to ping PC5 from PC0.
Ping was successfull



Within LAN

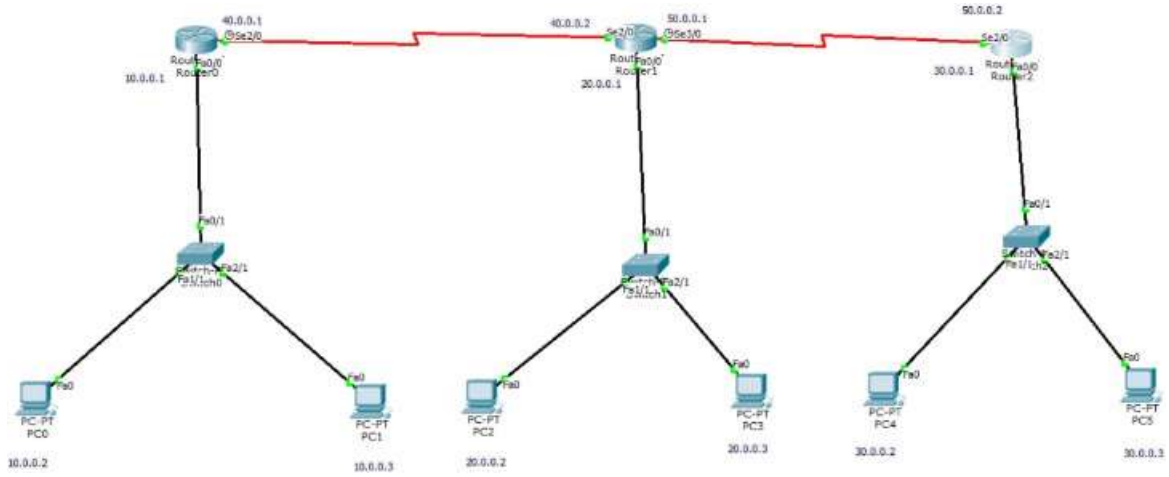


Outside LAN

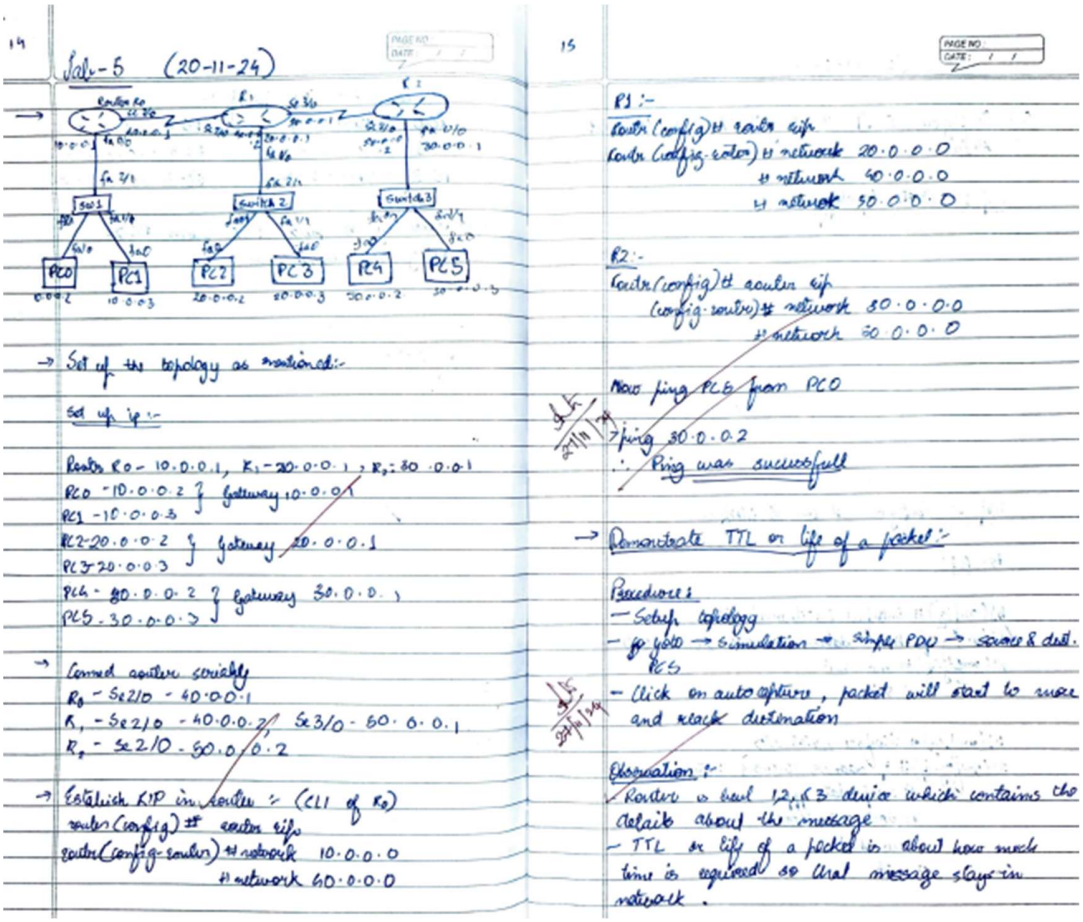
Program 5:

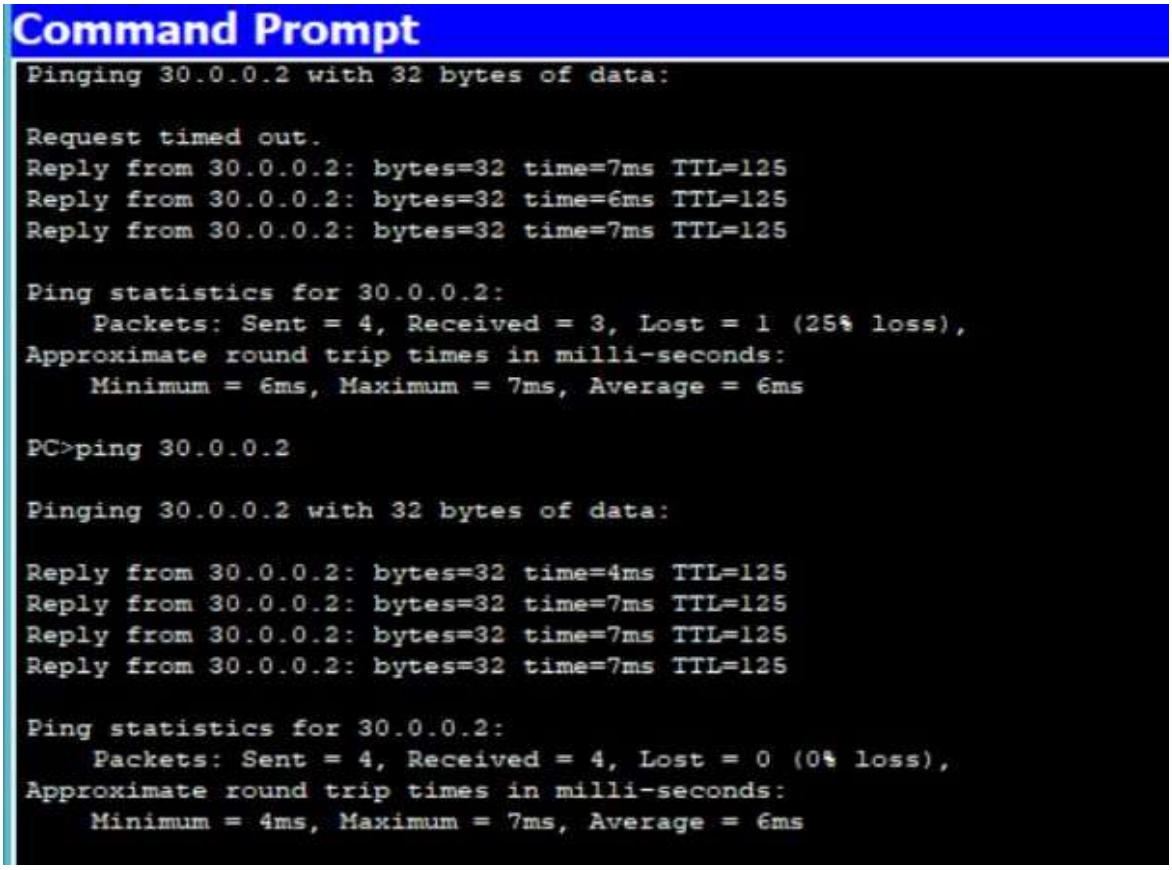
Aim: Configure RIP routing Protocol in Routers.

Topology:



Procedure and Observation:

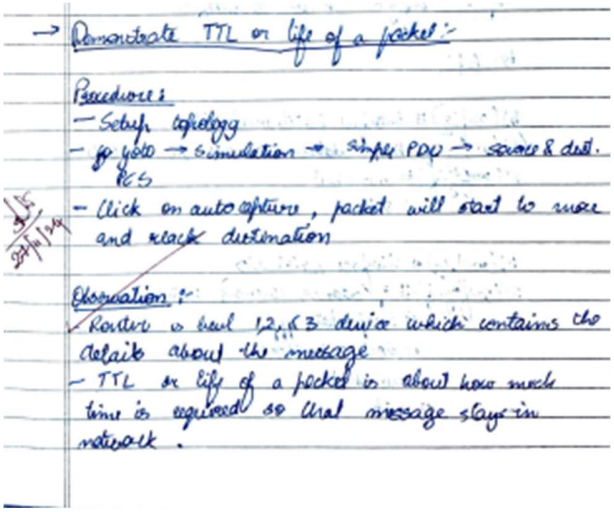


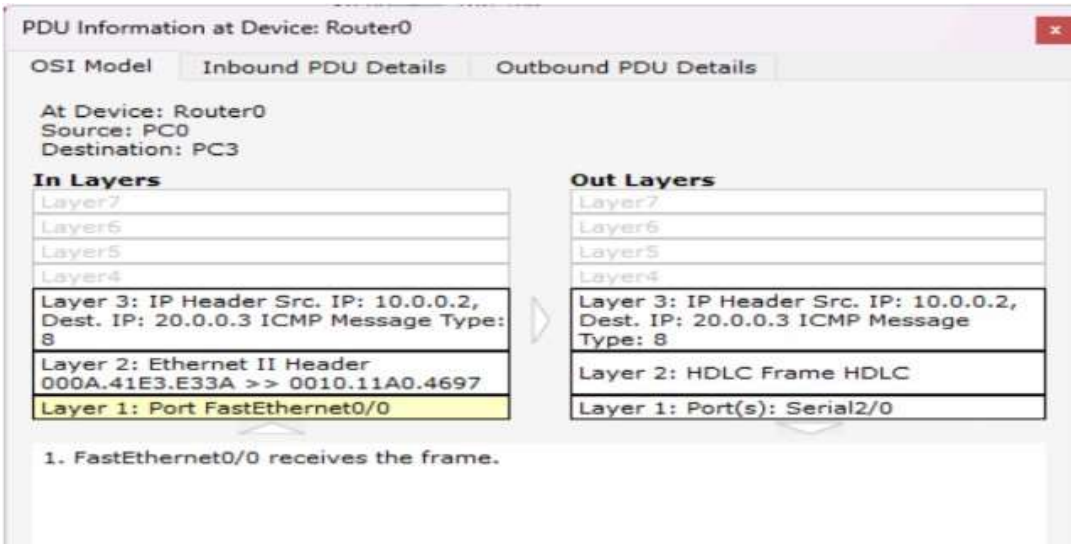

Command Prompt
Pinging 30.0.0.2 with 32 bytes of data:
Request timed out.
Reply from 30.0.0.2: bytes=32 time=7ms TTL=125
Reply from 30.0.0.2: bytes=32 time=6ms TTL=125
Reply from 30.0.0.2: bytes=32 time=7ms TTL=125
Ping statistics for 30.0.0.2:
Packets: Sent = 4, Received = 3, Lost = 1 (25% loss),
Approximate 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

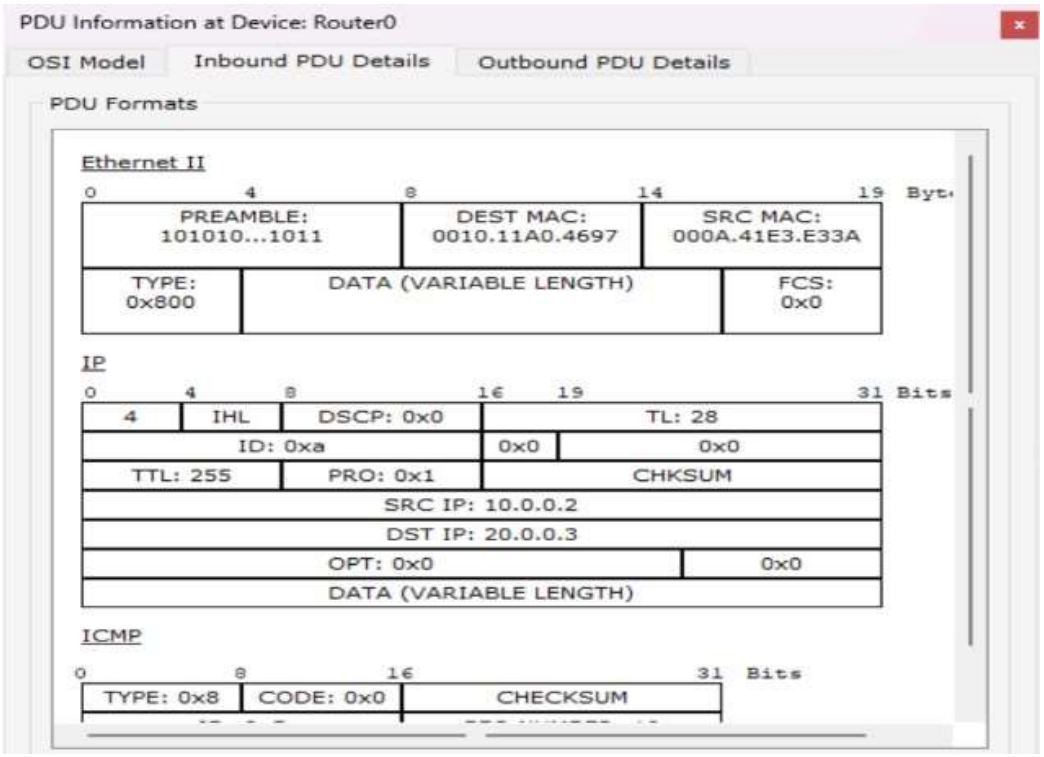
Program 6:

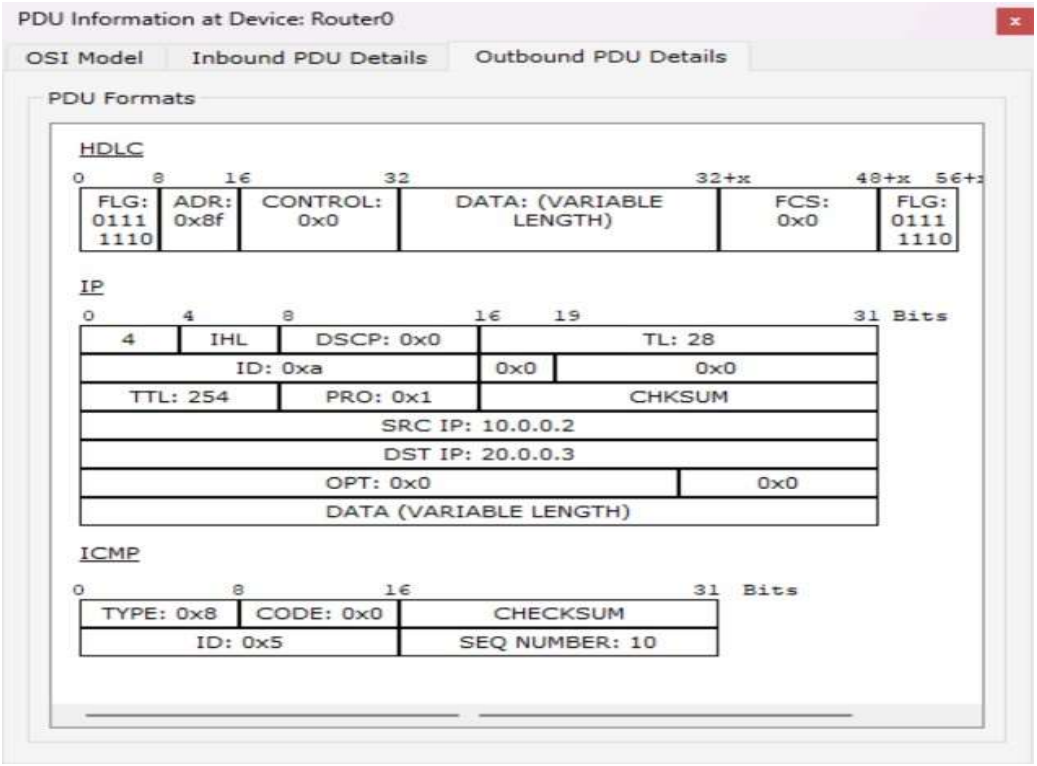
Aim: Demonstrate the TTL/ Life of a Packet.

Procedure and Observation:





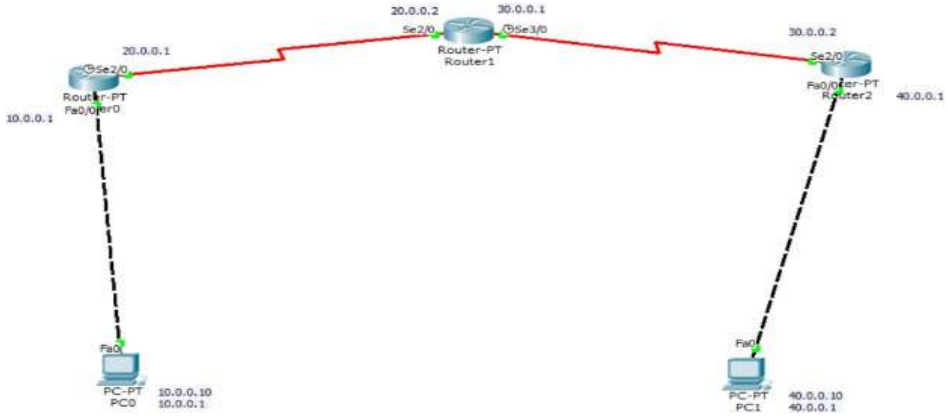

PDU Information at Device: Router0
OSI Model
Inbound PDU Details
Outbound PDU Details
PDU Formats
Ethernet II
0 4 8 14 19 Byt
PREAMBLE: 101010...1011
DEST MAC: 0010.11A0.4697
SRC MAC: 000A.41E3.E33A
TYPE: 0x800
DATA (VARIABLE LENGTH)
FCS: 0x0
IP
0 4 8 16 19 31 Bits
4
IHL
DSCP: 0x0
TL: 28
ID: 0xa
0x0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

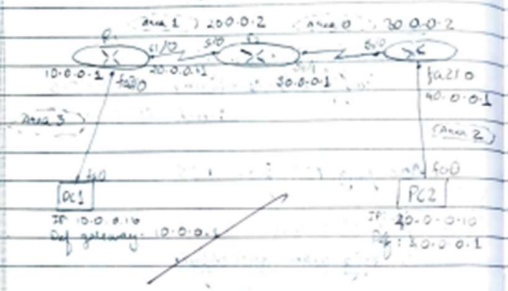


Procedure and Observation:

Lab-6 (27-11-24)

Q.) Configure OSPF Routing Protocol & connect Areas

Step 1: Create topology



Step 2:- Configure IP for all interfaces

for R1:-

R1(config)# interface fastethernet 2/0
R1(config-if)# ip address 10.0.0.1 255.0.0.0
R1(config-if)# no shut
exit

R1(config)# interface serial 2/0
R1(config-if)# ip address 20.0.0.1 255.0.0.0
encapsulate ppp
clock rate 64000
no shut
exit

In R2)

R2(config)# interface serial 2/0
R2(config-if)# ip address 20.0.0.2 255.0.0.0
encapsulation PPP
clock rate 64000
no shut
exit

R2(config)# interface serial 1/1
R2(config-if)# ip address 30.0.0.1 255.0.0.0
encapsulation ppp
clock rate 64000
no shut
exit

In R3:-

R3(config)# interface serial 3/0
R3(config-if)# ip address 30.0.0.2 255.0.0.0
encapsulation ppp
no shut
exit

R3(config)# interface fastethernet 2/0
R3(config-if)# ip address 40.0.0.1 255.0.0.0
no shut
exit

P-T-O

Step 3:- Configure OSPF routing protocol.

For R1:-

R1(config)# router ospf 1
R1(config-router)# router-id 1.1.1.1
network 10.0.0.0 0.255.255.255 area 3
network 20.0.0.0 0.255.255.255 area 1
exit

For R2:-

R2(config)# router ospf 1
R(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

For 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
exit

→ You have to configure router id when we configure OSPF. It is used to identify the router

Step 4:- Check routing table of R1,

Router# show ip route
C 10.0.0.0/8 directly connected, FastEthernet 2/0
C 20.0.0.0/8 directly connected, Serial 1/0
O IA 40.0.0.0/8 via 20.0.0.2
O IA 30.0.0.0/8 via 20.0.0.2

Here R2 knows area 0. Network 20.0.0.0 connected to R2 from R1

R3 - here, 1 is process ID. It initializes ospf process

There must be one process interface to keep up the OSPF process. Therefore we configure loopback address to all router

R1(config-if)# interface loopback 0
ip add 172.16.1.252 255.255.0.0
no shut

R2(config-if)# interface loopback 0
ip add 172.16.1.253 255.255.0.0
no shut

R3(config-if)# interface loopback 0
ip add 172.16.1.254 255.255.0.0
no shut

Step 5:- Check ip route of R3

R3# show ip route
O IA 20.0.0.0/8 via 30.0.0.1
C 40.0.0.0/8 directly connected, fastEthernet 2/0
C 30.0.0.0/8 directly connected, Serial 1/0

Here R3 doesn't know about area 3, so we create a virtual link between R1 and R2

Step 6:- Create virtual link between R1 and R2 to connect area 3 to area 0.

In R1:-

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

In R2:-

```
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 via 30.0.0.1 , Serial 1/0
    C     40.0.0.0/8 directly connected, FastEthernet 2/0
    O IA  10.0.0.0/8 via 30.0.0.1, Serial 1/0
    C     30.0.0.0/8 directly connected, Serial 1/0
```

Step 8:- Check connectivity between host 10.0.0.10 & 40.0.0.10

→ PING 40.0.0.10

32 bytes from 40.0.0.10 : seq=5 ttl=61 time=96.107ms

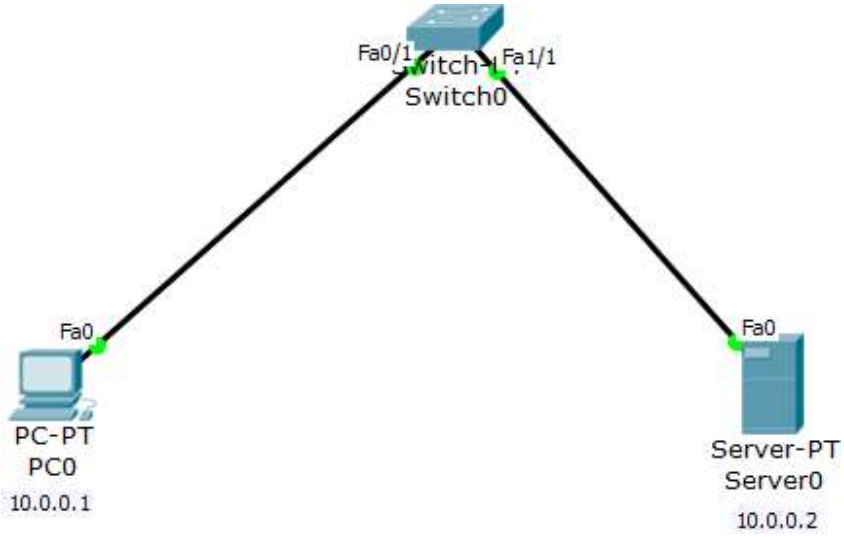
Ping was successful.

```
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
```

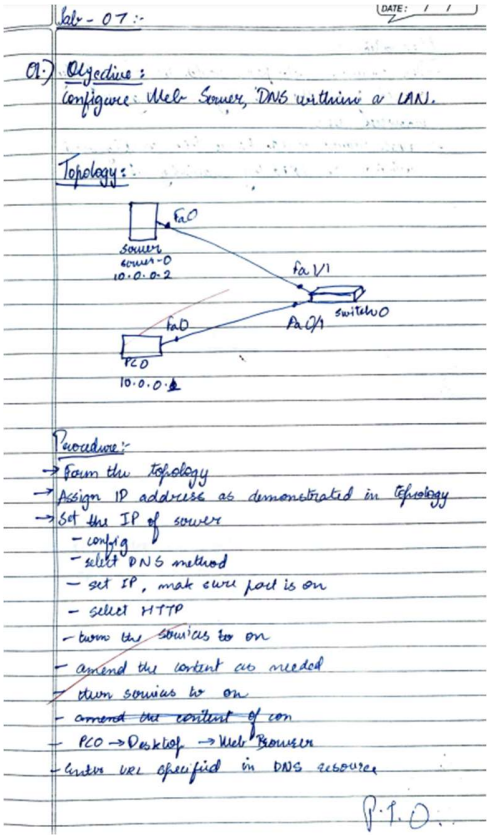
Program 8:

Aim: Configure Web Server, DNS within a LAN**.**

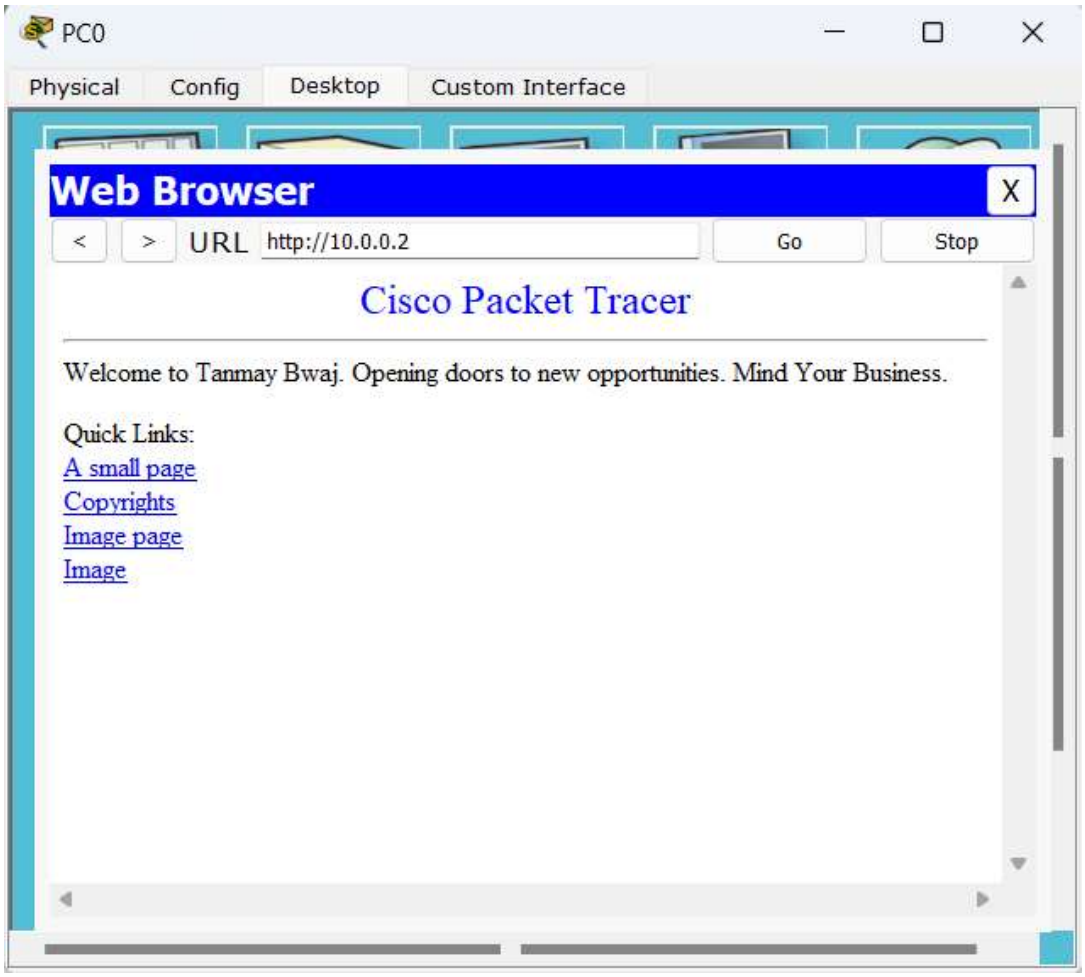
Topology:



Procedure and Observations:



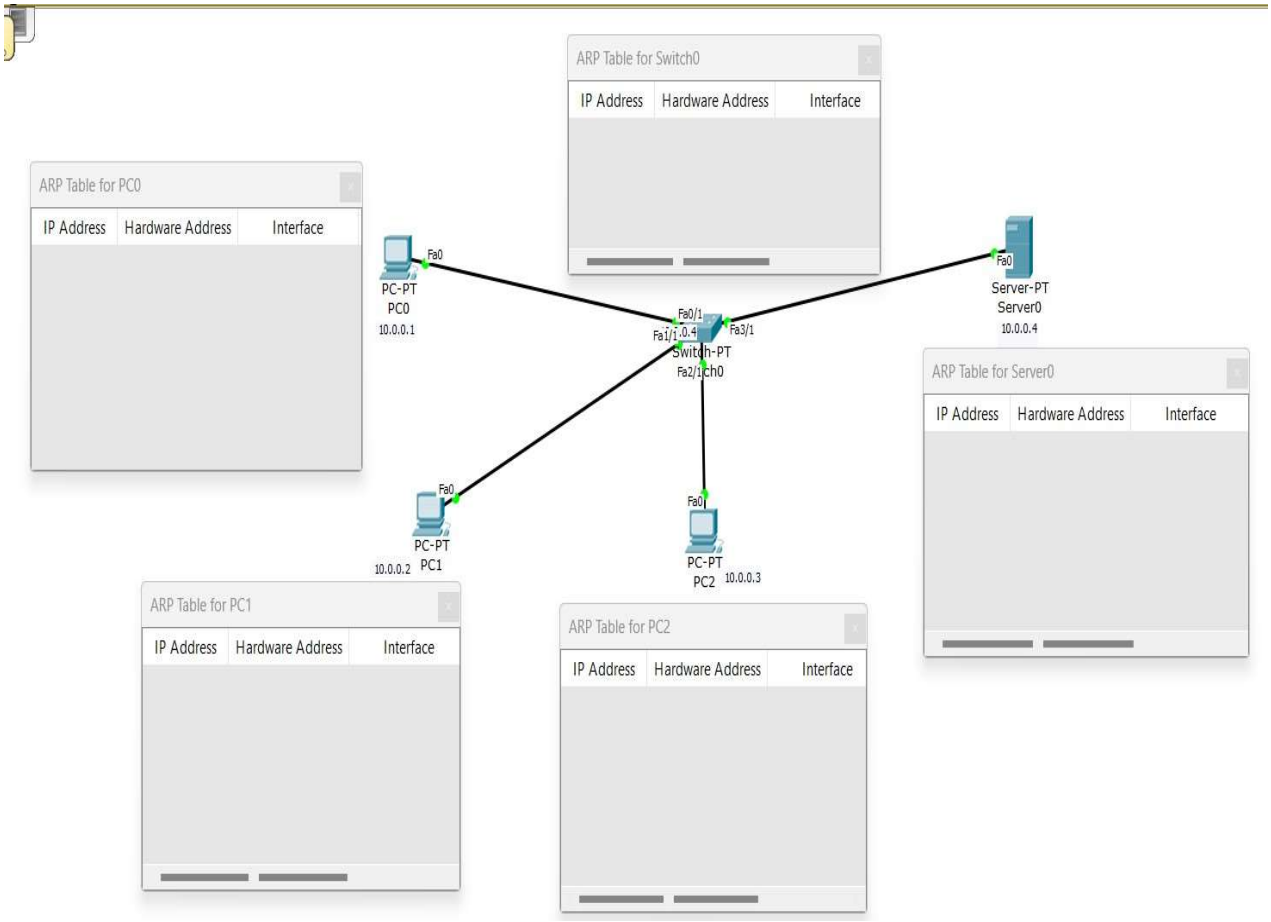
Observation :-
- the server's web page could be successfully accessed from the PC by entering the respective URL
- DNS server could hence be configured within an IAN by enabling the DNS.



Program 9:

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

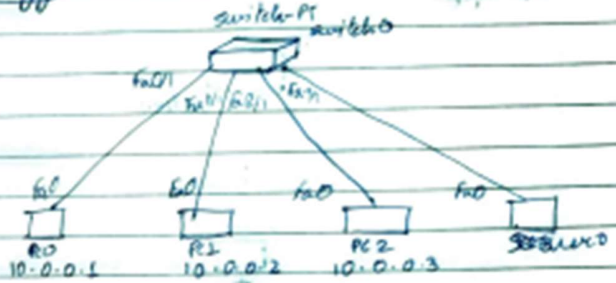
Topology:



Procedure and Observations:

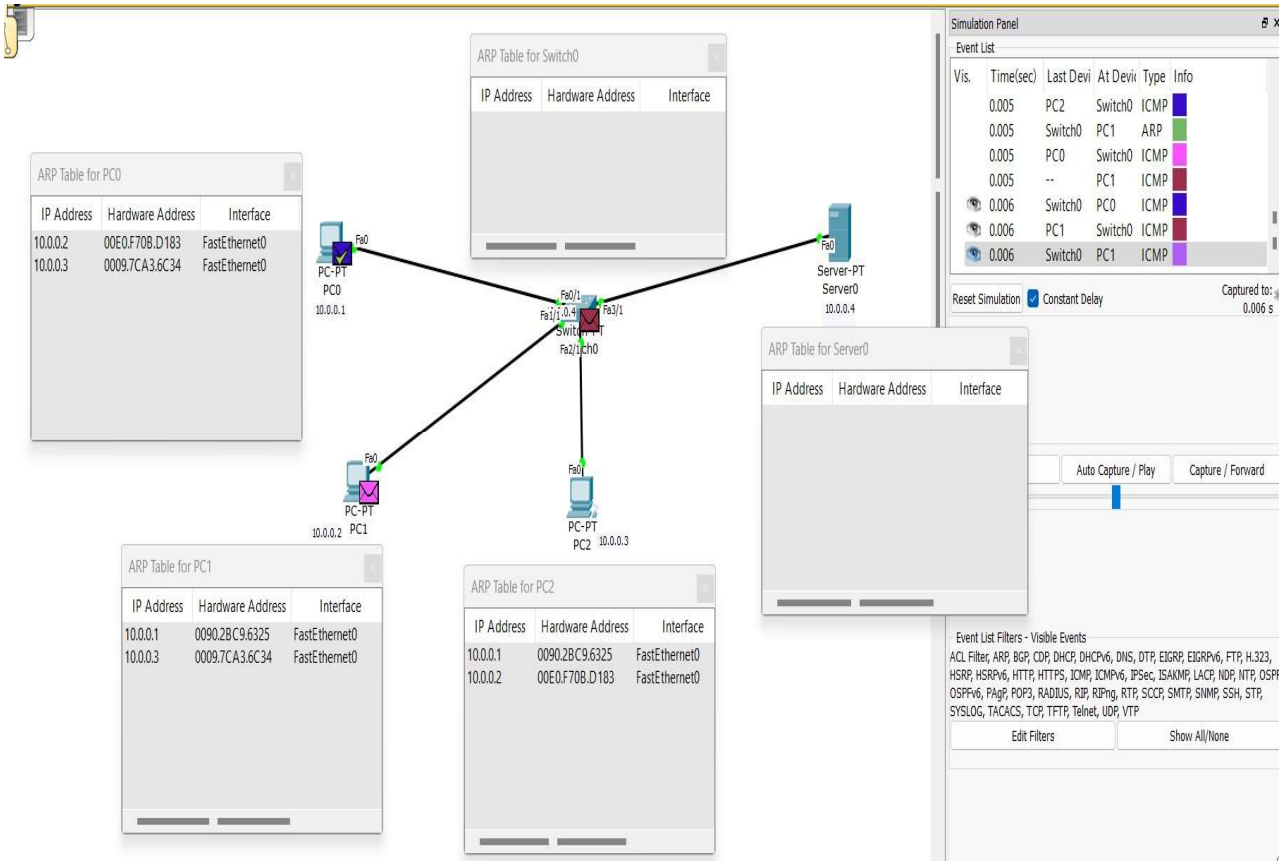
02) Objective :-
To construct simple LAN and understand the concept and operation of address Resolution Protocol ARP

Topology :-



Procedure :-
→ Form the above topology
→ Use inspect tool to view ARP table
→ Go to CLI of switch & do show mac address
→ Similarly obtain ARP table of server and other end devices.
→ Enter the simulation mode & click on 'capture' by selecting PC1 & PC2 for simple PDUs

Observation :-
- Initially, the ARP table of all end devices are observed to be empty
→ The MAC address table is also found to be empty
→ When the 'capture' is clicked ARP table is updated
→ Once the acknowledgment is obtained, the ARP table of PC1 is updated with IP of PC2
→ The event list in the simulation panel shows the corresponding protocol.




```
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:

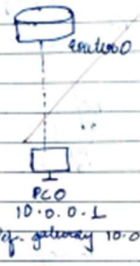
Aim: To understand the operation of TELNET by accessing the router in the server room from a PC in the IT office.

Topology :

Procedure and Observations:

Q3) Objective :-
To understand the operation of TELNET by accessing the router in server room from a PC in IT office.

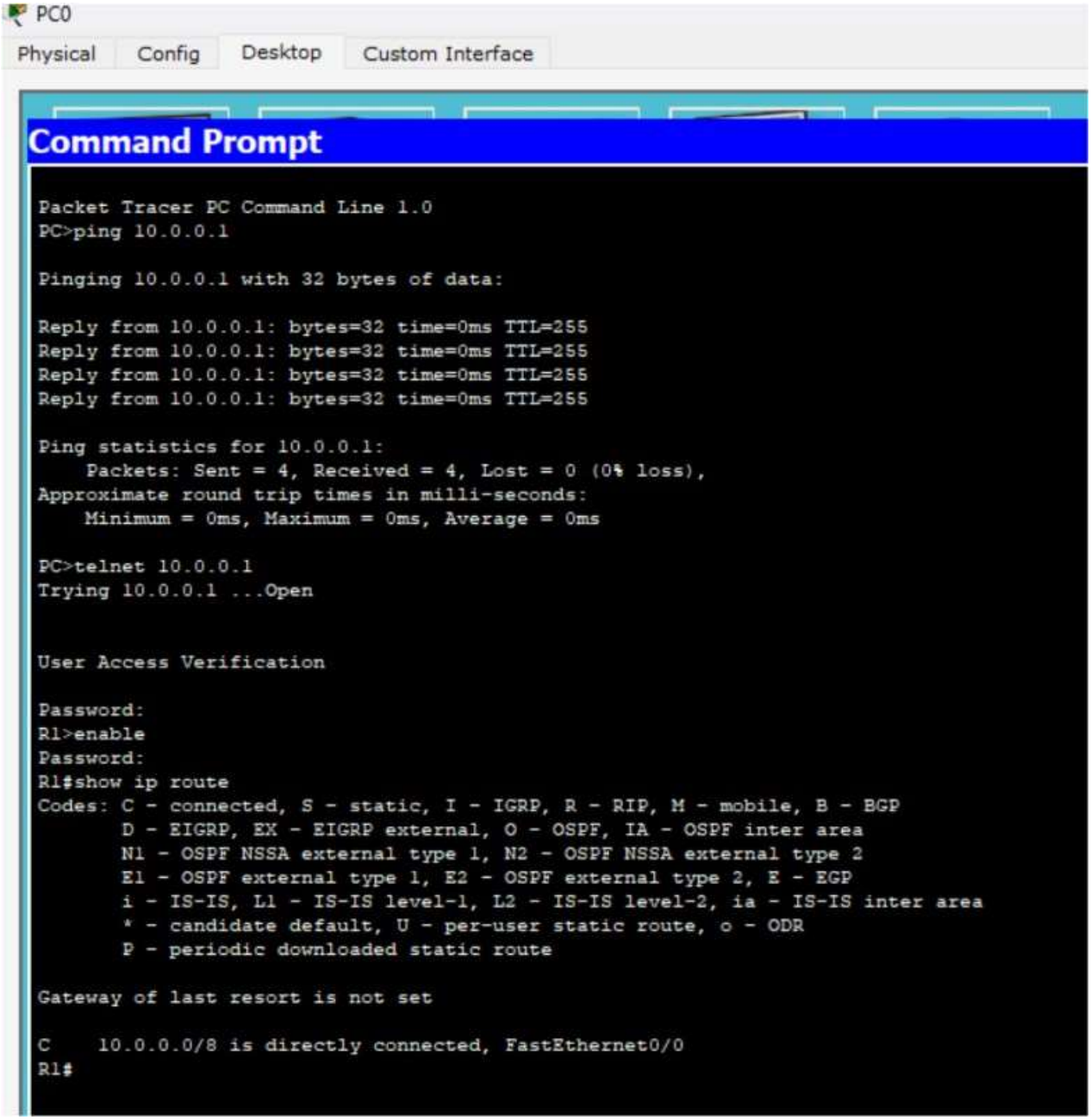
Topology:



Procedure:
→ Place the above topology with IP.
→ In router0, do :-
>enable
config terminal
hostname R1
enable secret p1
interface fa 0/0
ip address 10.0.0.1 255.0.0.0
no shut
line vty 0 5
login
password p0
exit
>exit

→ In PC0, do :
ping 10.0.0.1
telnet 10.0.0.2
User Access verification
Password : p0
>enable
Password : p1
show ip route

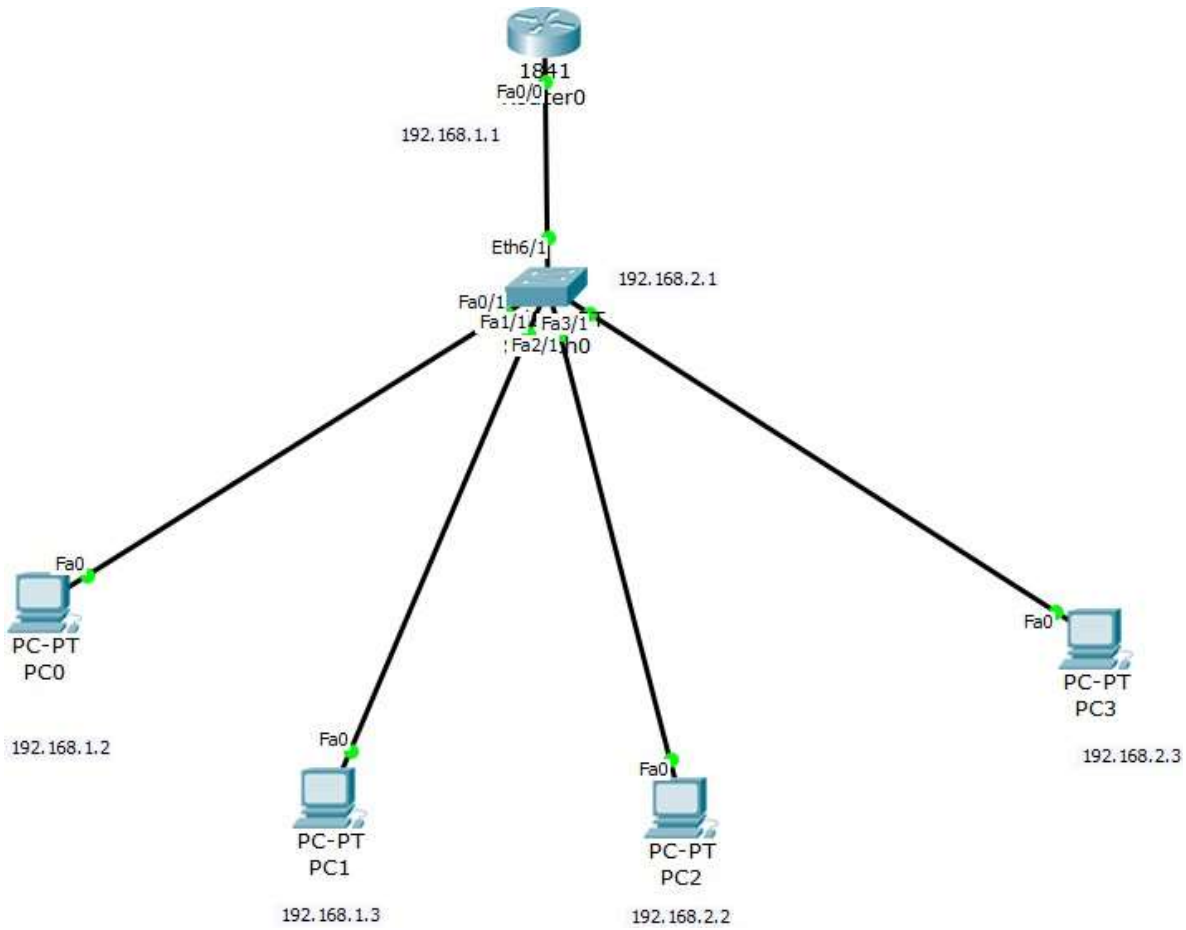
→ Observation :
→ Two password are given while configuring the router, one being the secret key for the router and other being the password for login and correspondingly access.
→ The passwords entered in the router are used in the inverse order there, i.e., the user has to login first to verify access via p0 and then obtain router a
→ Hence, the admin in PC is able to run commands as run in router CLI and see the results from PC.


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#

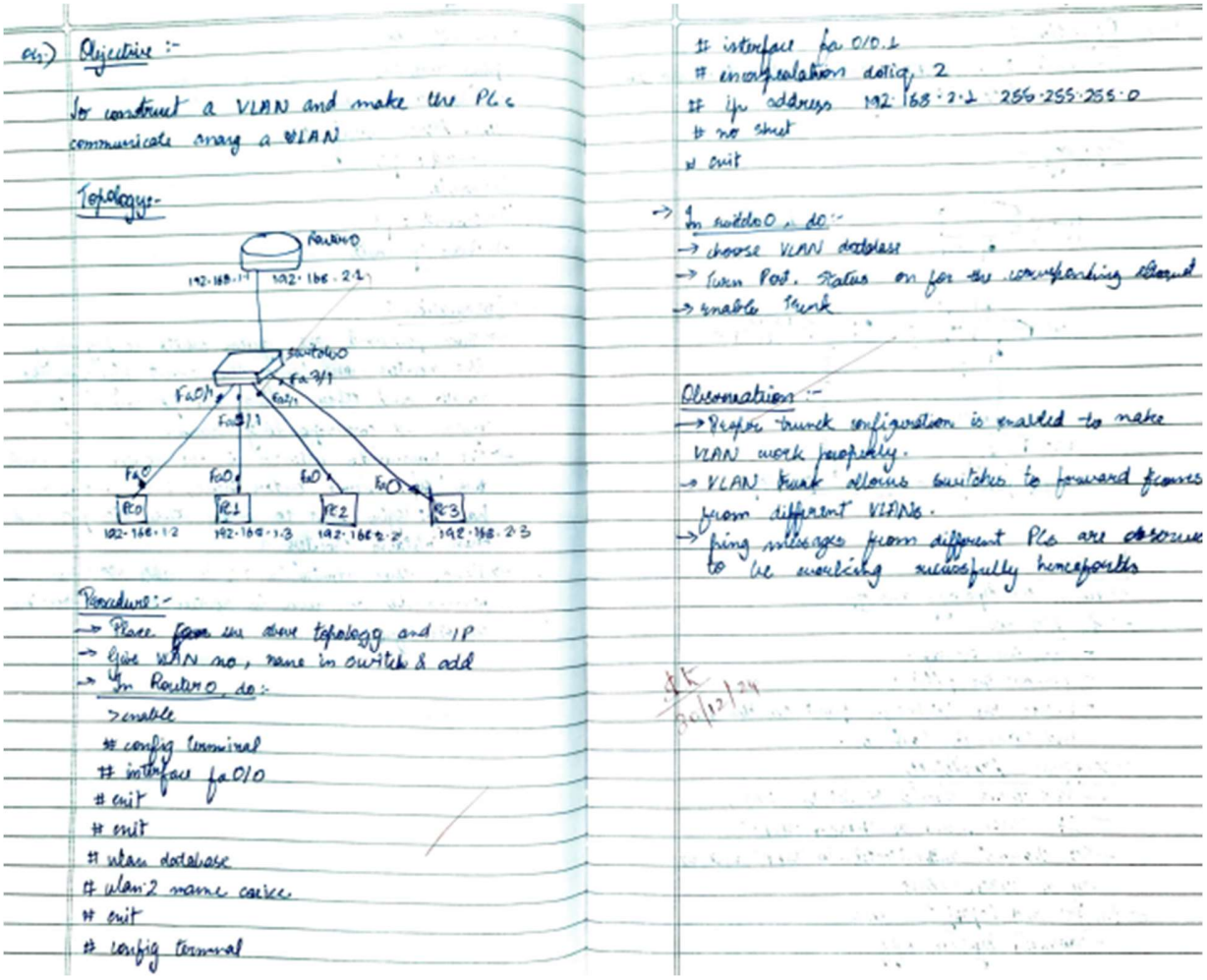
Program 11:

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

Topology:



Procedure and Observations:



Ex:) Objective :-

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

Topology:-

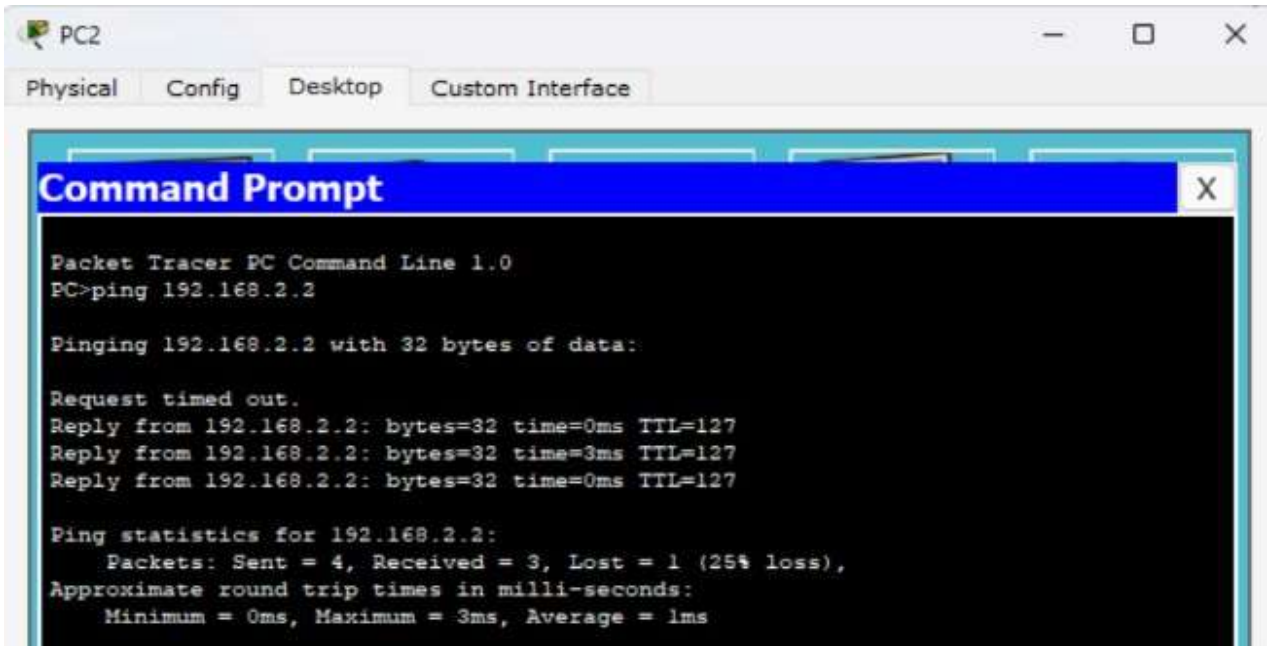
Procedure:-

→ Place the above topology and IP
→ Give VLAN no, name in switch & add
→ In Router0 do:
>enable
config terminal
interface fa0/0
exit
exit
vlan database
vlan 2 name cse cc.
exit
config terminal
interface fa 0/0.2
encapsulation dotiq 2
ip address 192.168.2.1 255.255.255.0
no shut
exit

→ In switch0 do:-
→ choose VLAN database
→ Turn Port. Status on for the corresponding Vlan
→ enable Trunk

Observation:-

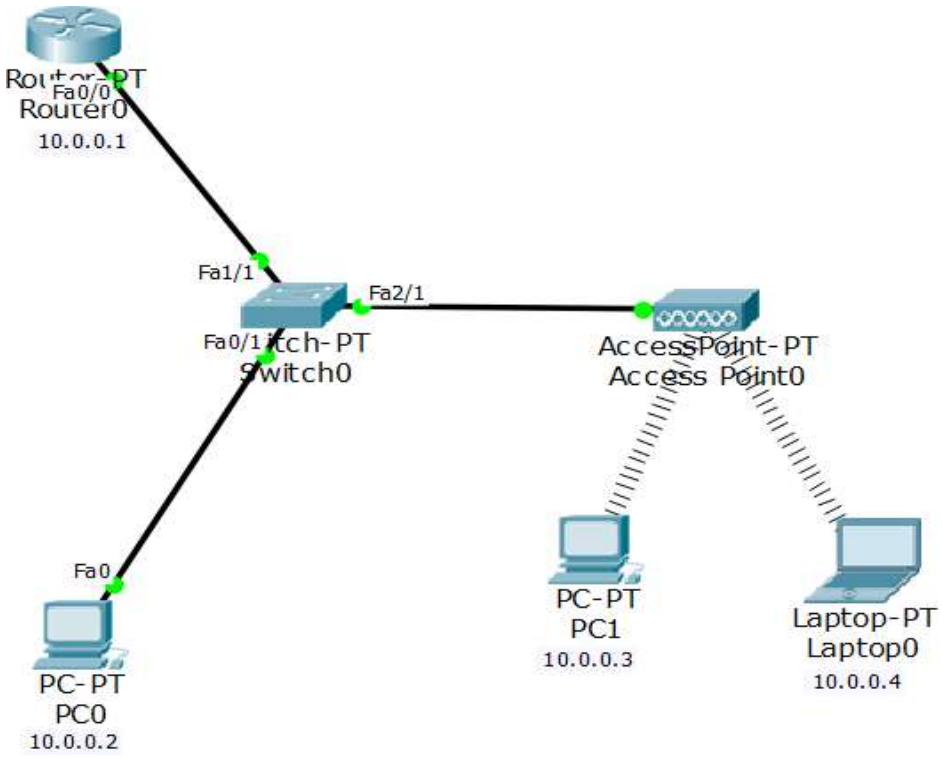
→ Proper trunk configuration is enabled to make VLAN work properly.
→ VLAN trunk allows switches to forward frames from different VLANs.
→ ping messages from different PCs are observed to be working successfully henceforth



Program 12:

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

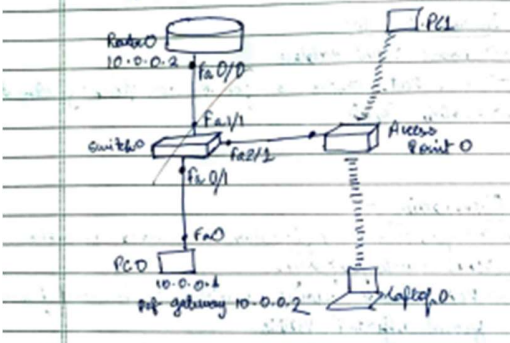
Topology:



Procedure and Observations:

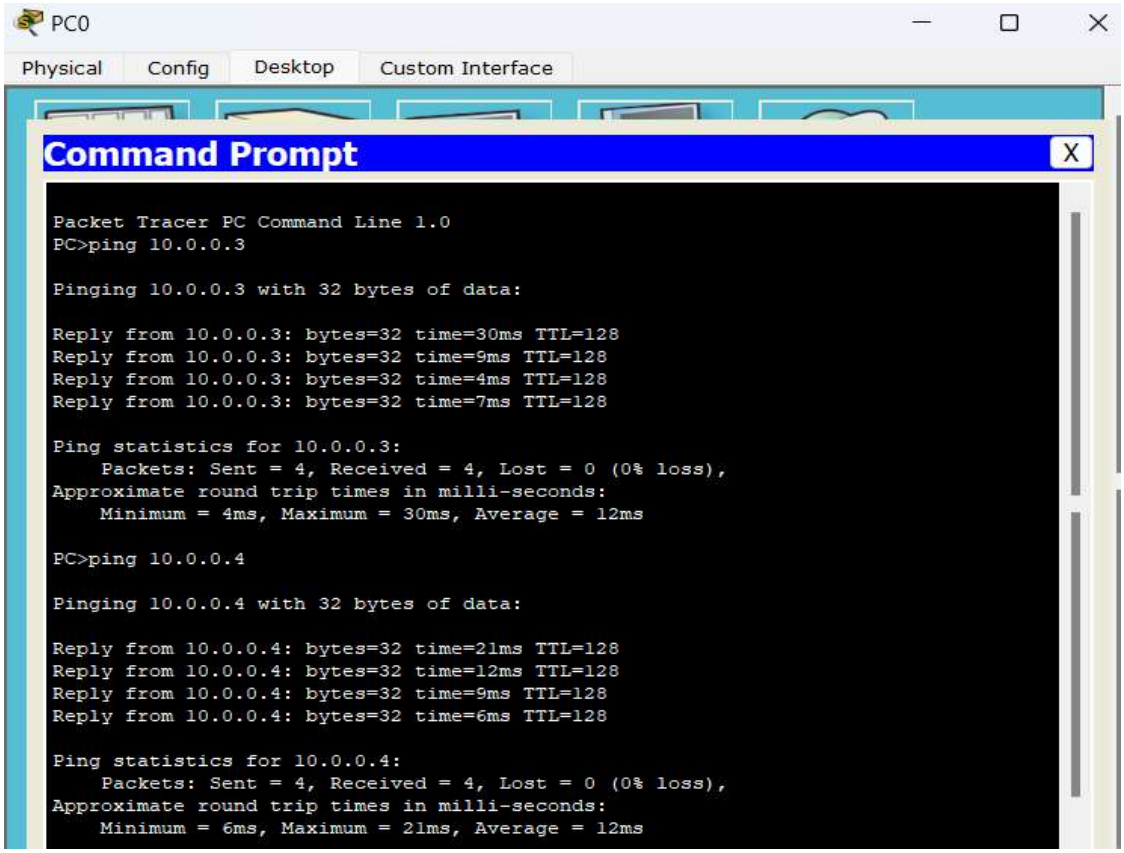
05) Objective :-
To construct a WLAN and make the nodes communicate wirelessly.

Topology :-



Procedure :-
→ Form the topology and IP
→ On PC0 do:-
- Turn the PC off.
- Remove the port.
- Place the Linksys port to the PC and turn it back on.
→ Configure AccessPoint0
- Port status should be set to 'ON'.
- Set SSID name as 'BMSCE5ECN'
- Set channel authentication to 'WEP' and set key as '1234567890'
→ On PC1 and Laptop 0, do:-
- Turn the system off
- Remove the port
- Place the wireless port and turn it back on.
→ On config, do:-
- set the same SSID
- set authentication to WEP and enter same key
→ Ping from different devices and observe the transmissions.

Observations :
- After the setup of PC1 and Laptop 0, wireless connections with dashed line were observed in connection with AccessPoint, indicating
- Devices could connect to WLAN since they were in the network range.
→ Signal strength decreases with increase in distance.


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

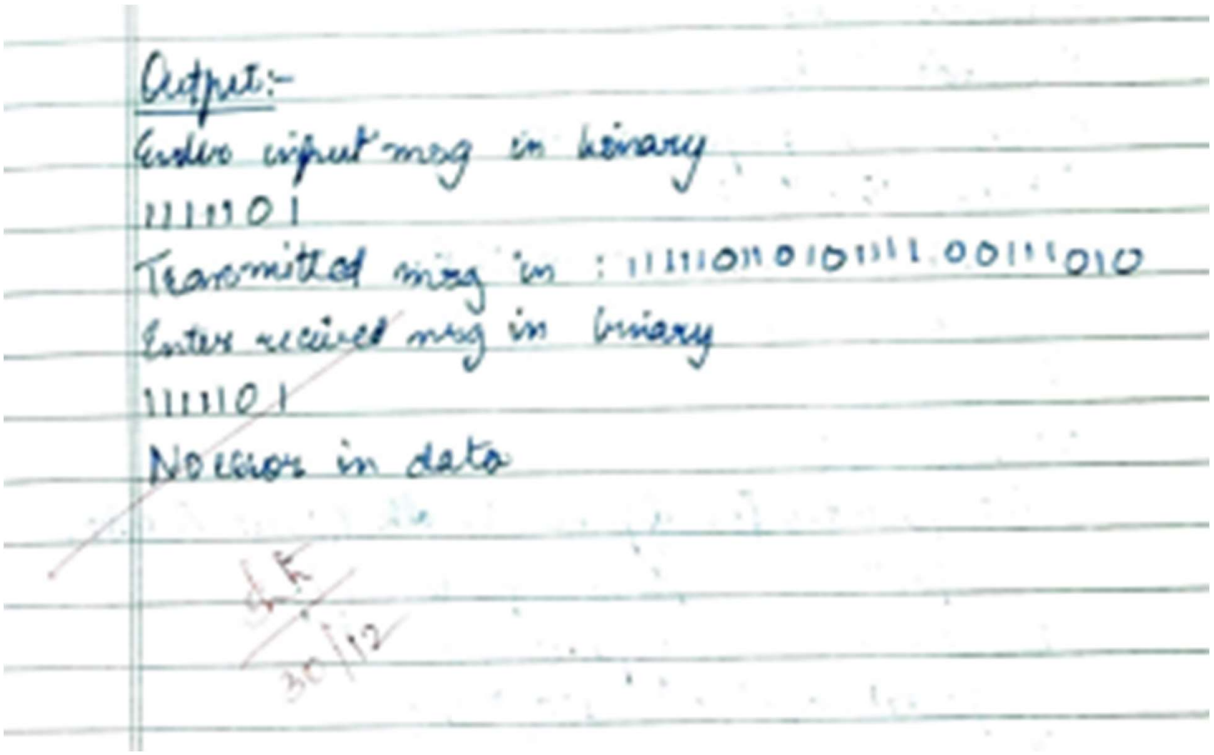
CYCLE - 2

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:-
Enter input msg in binary
1111101
Transmitted msg is : 11111011010111100111010
Enter received msg in binary
1111101
No error in data

Program 14:

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: 10
```

Time 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!!

Program 15:

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
```