# VISVESVARAYA TECHNOLOGICAL UNIVERSITY

**“JnanaSangama”, Belgaum -590014, Karnataka.**

**LAB REPORT**

**on**

# COMPUTER NETWORKS

***Submitted by***

**Dipesh Sah**

***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**
**Sep 2024-Jan 2025**

# B. M. S. College of Engineering,

**Bull Temple Road, Bangalore 560019**

(Affiliated To Visvesvaraya Technological University, Belgaum)

## Department of Computer Science and Engineering

**<u>CERTIFICATE</u>**

This is to certify that the Lab work entitled "**COMPUTER NETWORKS**" carried out by **Dipesh Sah (1BM22CS092),** who is bonafide student of **B. M. S. College of Engineering.** It is in partial fulfillment for the award of **Bachelor of Engineering in Computer Science and Engineering** of the Visvesvaraya Technological University, Belgaum during the year 2024-25. The Lab report has been approved as it satisfies the academic requirements in respect of **Computer Networks Lab - (23CS5PCCON)** work prescribed for the said degree.

**Sandhya A Kulkarni**

Assistant Professor,
Department of CSE,
BMSCE, Bengaluru

**Dr. Kavitha Sooda**

Professor and Head,
Department of CSE
BMSCE, Bengaluru

# INDEX

**CYCLE 1**

# INDEX

**CYCLE 2**

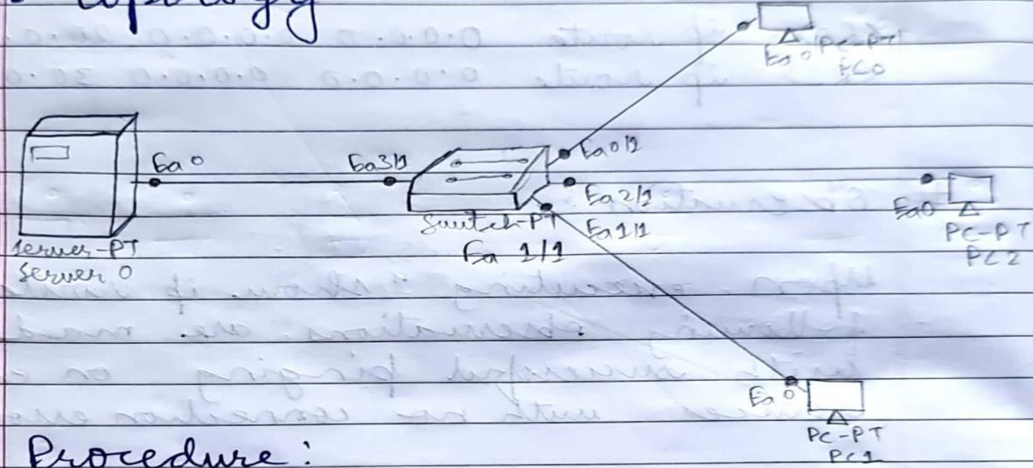
**Program 5**
**Aim:** Configure DHCP within a LAN and outside LAN.

**Topology , Procedure and Observation:**

Lab - 04

DHCP.
Aim :- Configure DHCP within a LAN and outside LAN.

① within the LAN.
• Topology




• Procedure:
1. Choose a generic server, a switch and 2PCs with a laptop and connect them to switch using auto cable.

2. Click on server → Desktop → IP config → Static. Set IP Address as 10.0.0.1 and default gateway 10.0.0.0

3. Again configure the server PT by: config → Services → DHCP
Service → ON, pool name; switch1.
Default gateway: 10.0.0.0
max. no. of users = 100
Start IP → 10.0.0.3 then click 'Add'.

4) For each PC -> go to config -> IP config
change static to DHCP.

# Observation :-

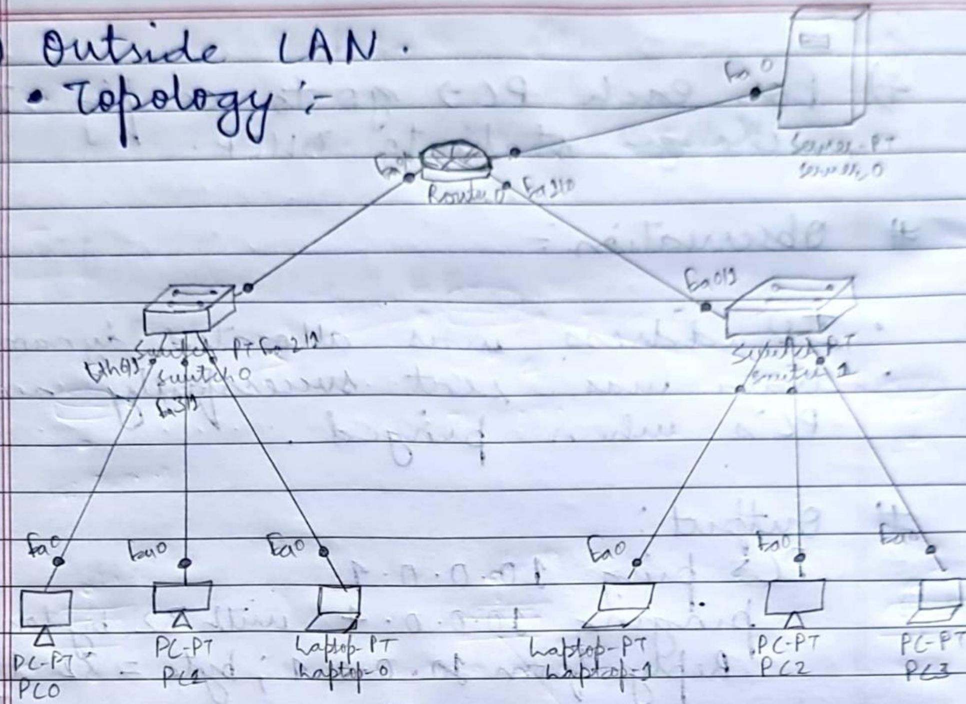
- IP address was allocated dynamically.
- Data was sent successfully among PC's when pinged.

# Output :
PC> ping 10.0.0.4
pinging 10.0.0.4 with 32 bytes of data:
Reply from 10.0.0.4 ; byte = 32 time =0ms TTL= 28
" "
" "
" "

Ping statistics for 10.0.0.4
Packets sent = 4, received = 4, lost = 0
(0% loss)
appropriate round trip time in milliseconds.
Minimum =0ms , Maximum = 0 ms ,

## (11) Outside LAN.

• Topology :-




# Procedure :

1. For the existing server, with switch & server, a laptop and 2 PCs, add a router and add second network switch.

2. In server IP config → static → IP addr: 10.0.0.2
   def gateway : 10.0.0.1

3. In server 2 → config → server → DHCP → modify the existing switch 1 → def gateway : 10.0.0.1
   in same : 10.0.0.0.

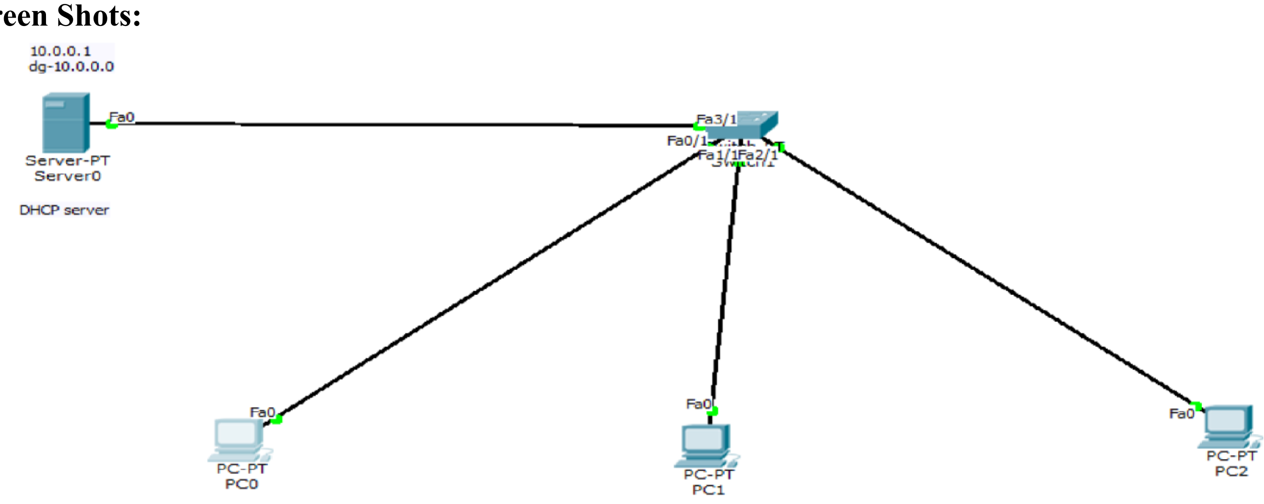
4. In server → DHCP → for switch 2,
   Poolname : switchtwo.
   def gateway :- 20.0.0.1
   Start IP : 20.0.0.3

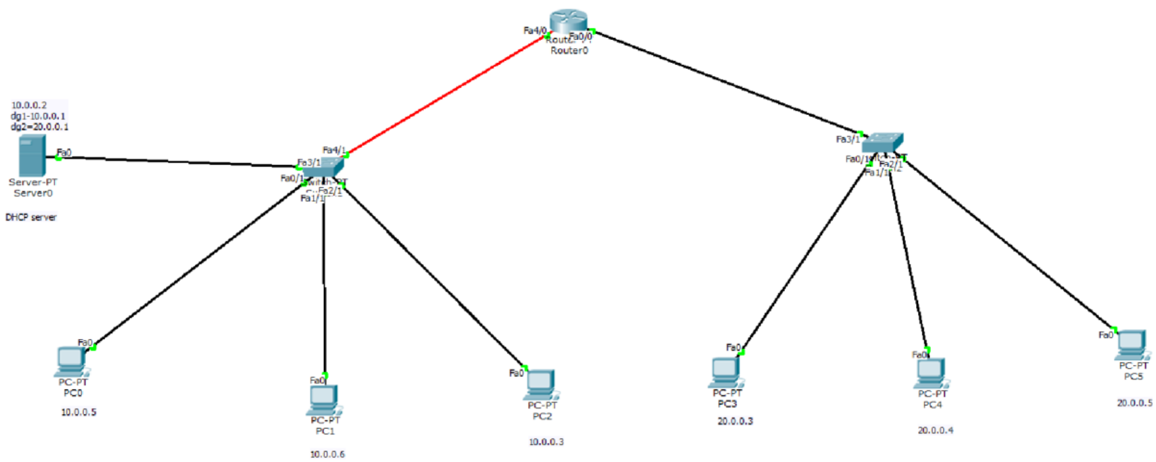
Bafna Gold
Date: Page 21

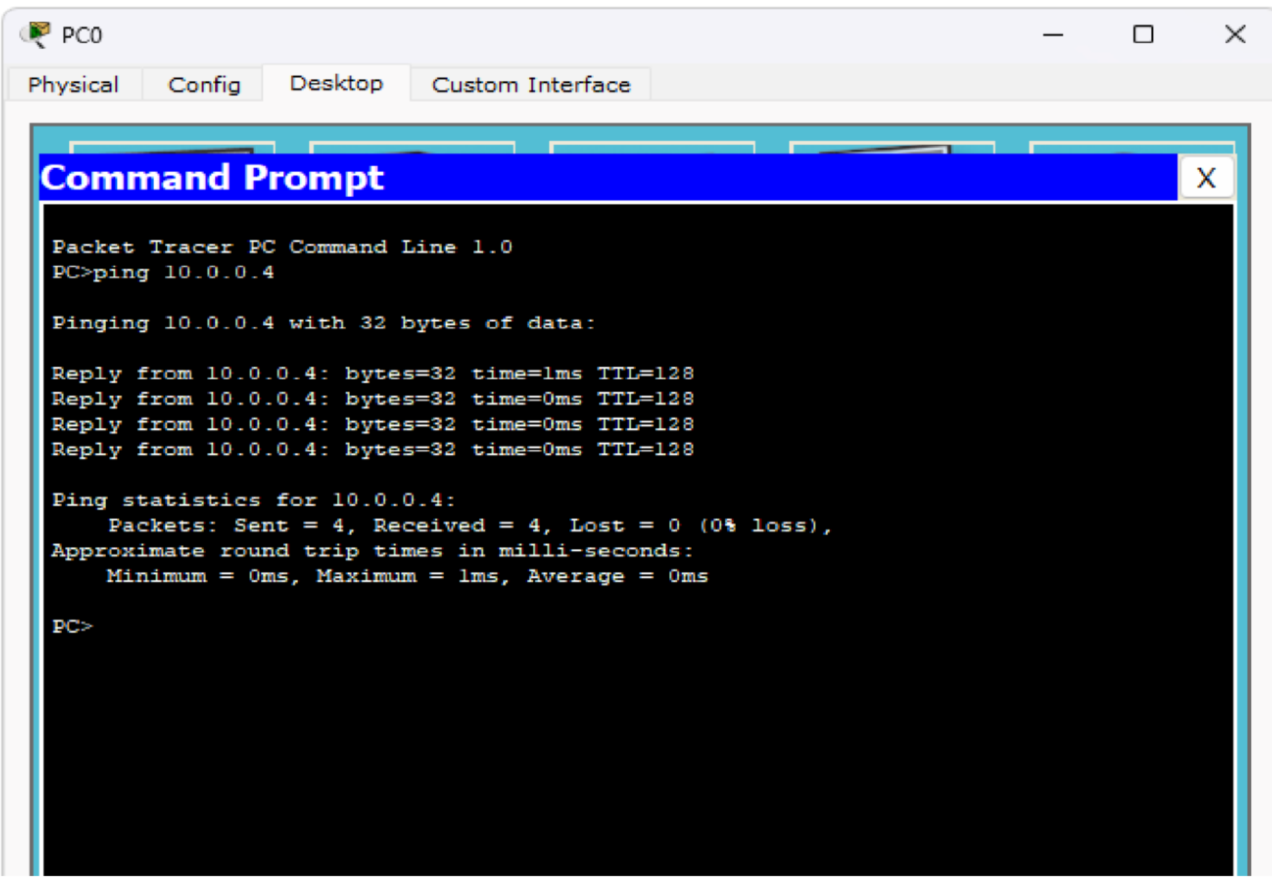
5. Do the router configuration :
enable ,
config terminal
interface fastethernet 4/0
ip address 10.0.0.1 255.0.0.0
ip helper address 10.0.0.2
no shut
exit

6. Now same for 2nd network,
interface fastethernet 0/0
ip address 20.0.0.1 255.0.0.0
ip helper address 10.0.0.2
no shut
exit.

**Screen Shots:**




10.0.0.2
dg1-10.0.0.1
dg2=20.0.0.1
Server-PT
Server0
DHCP server
Router0
PC-PT
PC0
10.0.0.5
PC-PT
PC1
10.0.0.6
PC-PT
PC2
10.0.0.3
PC-PT
PC3
20.0.0.3
PC-PT
PC4
20.0.0.4
PC-PT
PC5
20.0.0.5


PC0
Physical
Config
Desktop
Custom Interface
Command Prompt
X
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=1ms 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 = 1ms, Average = 0ms
PC>

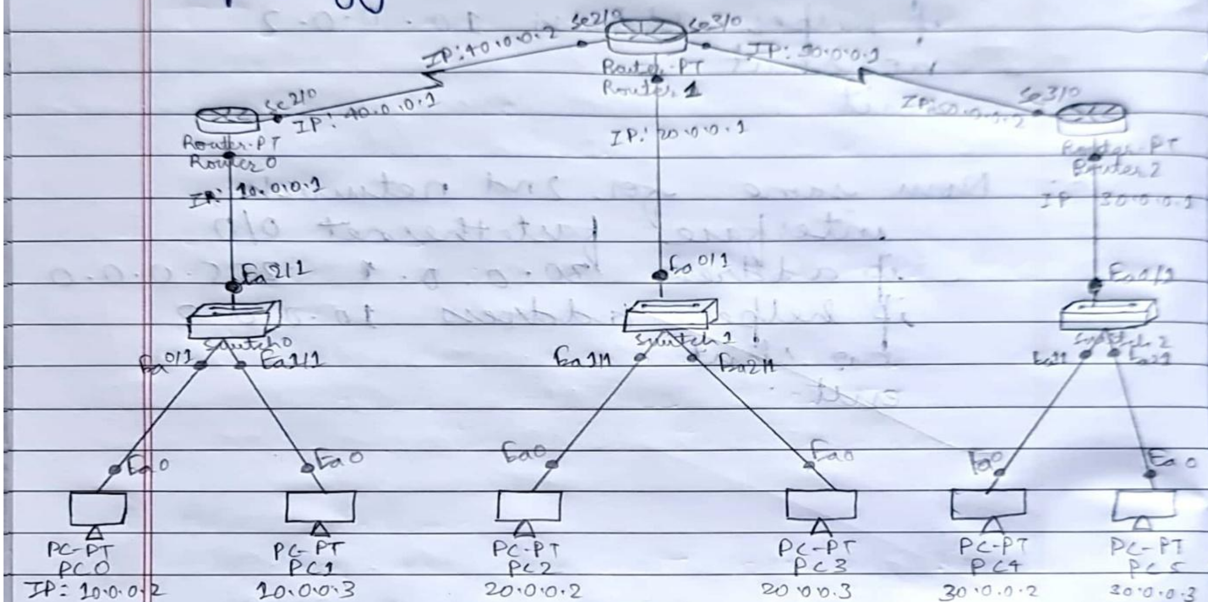
**Program 6**

**Aim:**Configure RIP routing Protocol in Routers .

**Topology , Procedure and Observation:**

Lab :- 6 (Experiment-5 (a))

- Aim : configure routing information protocol in routes (R.I.P).
- Topology :-




# Procedure :

1. Place 3 routers (generic), 3 generic switches and 6 PCs.
2. Connect the routers to the correspon-ding switches, Then connect 2 PCs to one switch.
3. Configure the end devices & define the gateways.
4. Configure the routers using CLI and check again for green lights for all the connections.

5) Configure routing information protocol to 3 routers.

In Router - 0;
(config) # router rip
(config - router) # network 10.0.0.0
(config - router) # network 40.0.0.0

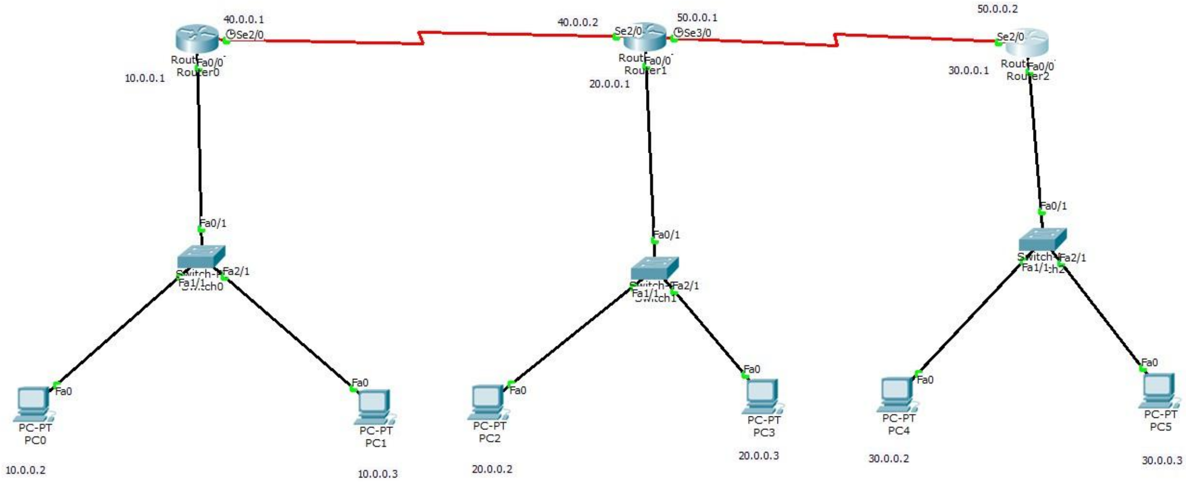
In Router - 1:
(config) # router rip
(config - router) # network 40.0.0.0
(config - router) # network 50.0.0.0
(config - router) # network 20.0.0.0

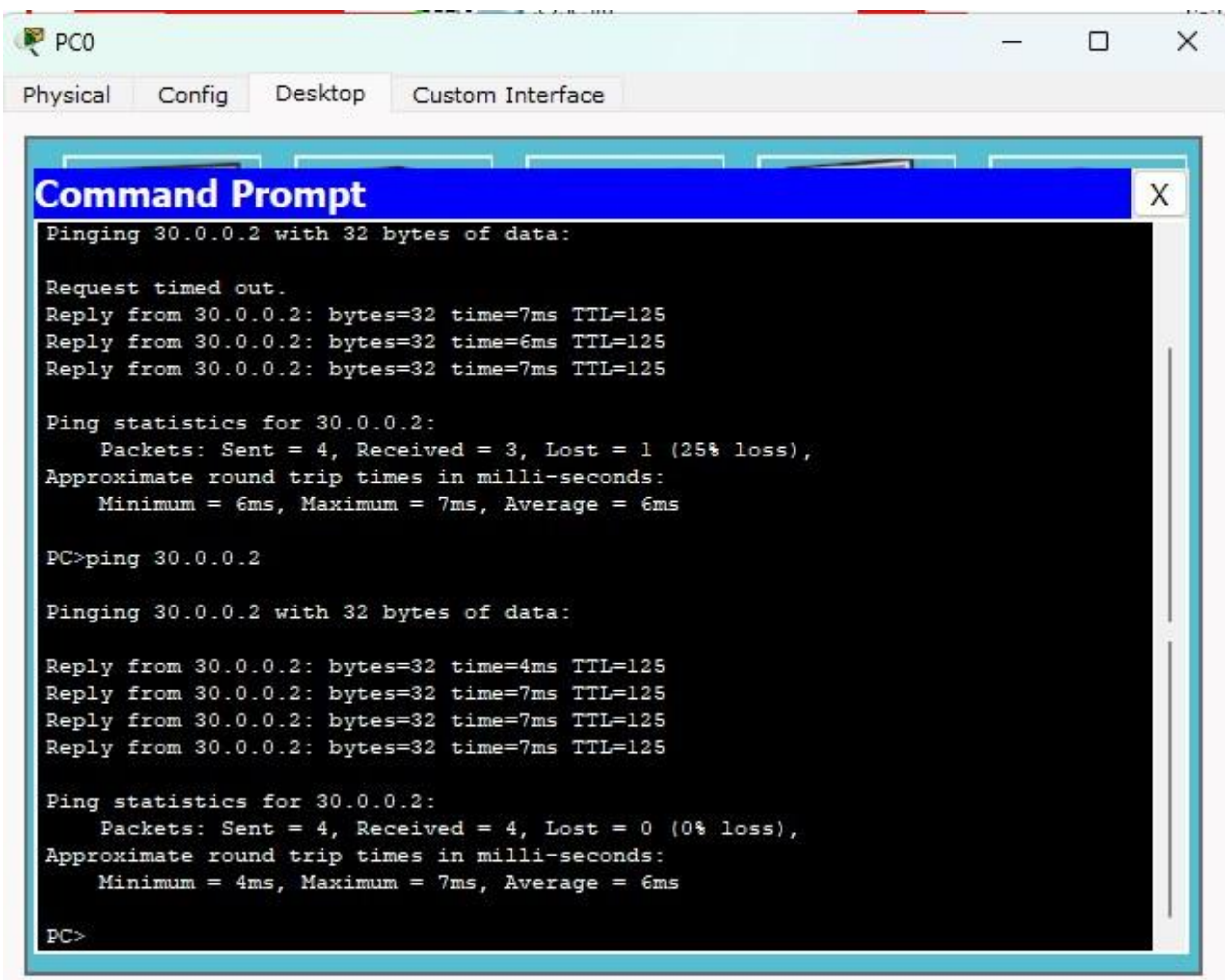
In Router - 2:
(config) # router rip
(config - router) # network 50.0.0.0
(config - router) # network 30.0.0.0

# Observation:
- Before doing RIP when pinging it showed timed out.
- After applying RIP when 30.0.0.3 was pinged from 10.0.0.1 was pinged successfully.

# show ip route for router 2:
R 10.0.0.0/8 [120/2] via 50.0.0.1, serial 2/0
R 20.0.0.0/8 [120/1] via 50.0.0.1, serial 2/0
R 50.0.0.0/8 is directly connected, serial 2/0.

**Screen Shots:**



**Program 7**
**Aim:**Demonstrate the TTL/ Life of a Packet .

**Topology , Procedure and Observation:**

Experiment : 6(b)

- Aim:- Demonstrate TTL / Life of a Packet.

TTL means 'time to leave' for a packet. It tells that for how many time units the packet will be there in the network.

- Procedure:-
  - Send a simple PDU from PC1 to PC3.
  - Click on Auto capture the event list then observe the TTF of each router in PDU Information.

- Observation:-

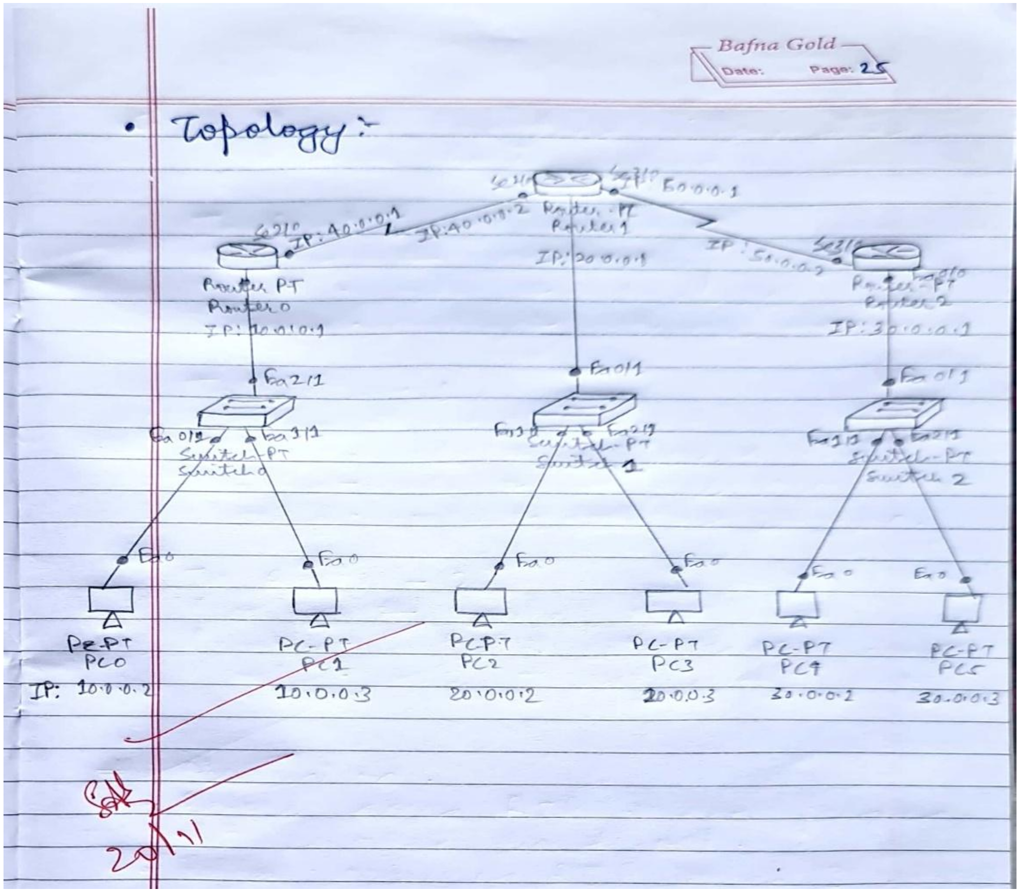
(i) When the packet passes Router 0,
inbound - TTL = 255 ms
outbound - TTL = 254 ms.

(ii) when the packet passes to Router 1,
inbound - TTL = 254 ms.
outbound - TTL = 253 ms.

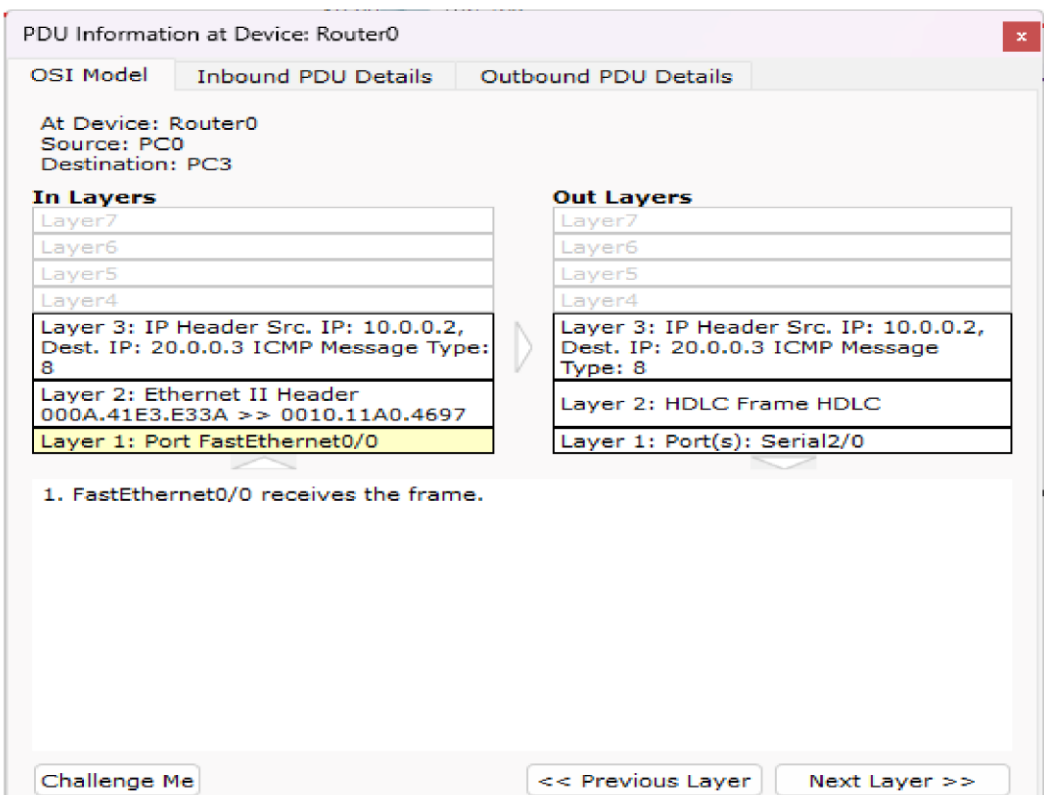
(iii) When the packet passes across Router 2,
inbound - TTL = 253 ms.
outbound - TTL = 252 ms.

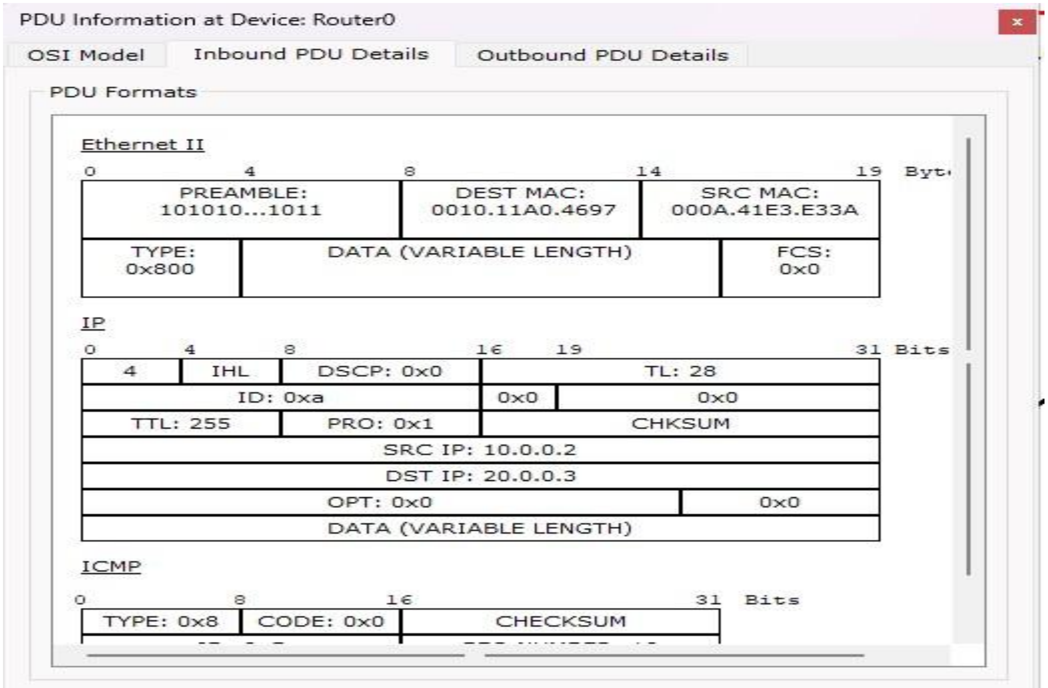
Thus, we conclude that there will be a decrement in TTL for 1ms when it passes across a Router.

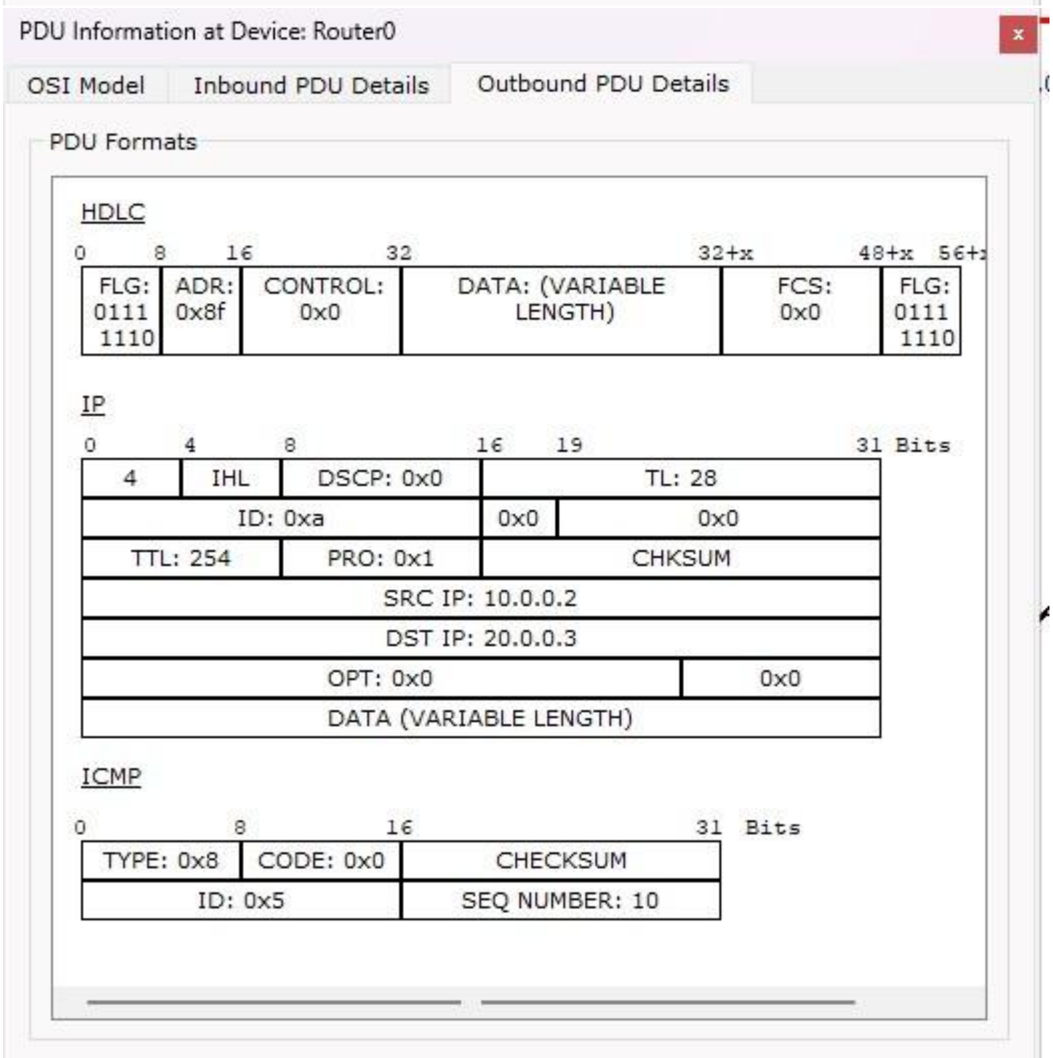
• Topology :-




**Screen Shots:**




PDU Information at Device: Router0
OSI Model
Inbound PDU Details
Outbound PDU Details
PDU Formats
Ethernet II
0 4 8 14 19
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
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 8**

**Aim:** Configure OSPF routing protocol .

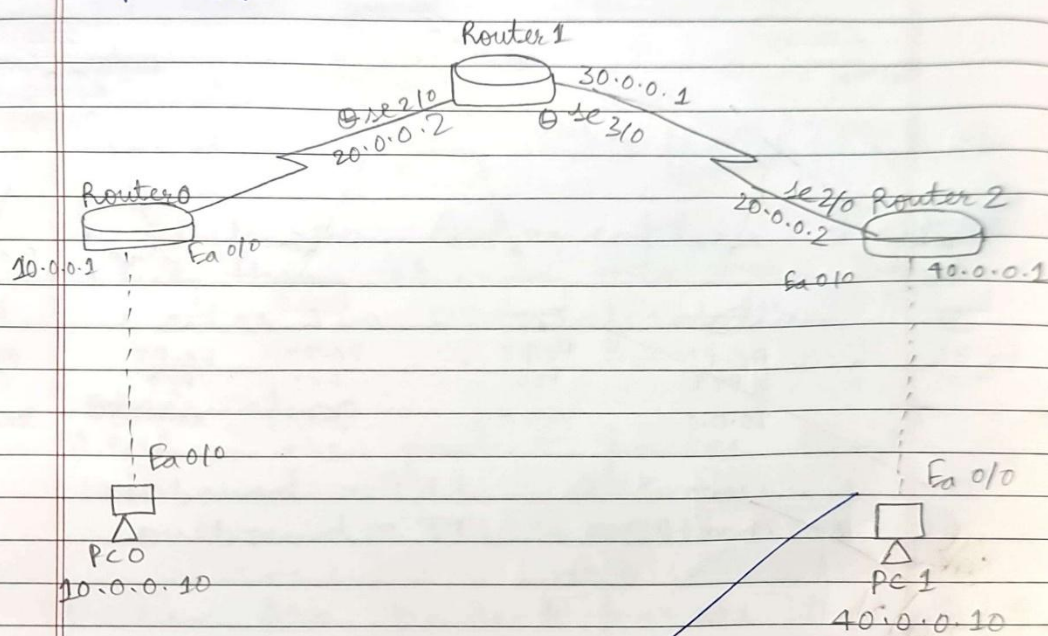
**Topology , Procedure and Observation:**

Experiment - 7

\# Question :> OSPF routing protocol configuration.

\# Aim :> To configure OSPF routing protocol.

\# Topology :>




\# Procedure :-

1) Connect the devices in the same manner as shown above.
Click on end devices → config → settings → set the default gateway (IP address of it's router) → then click on fast ethernet () → set the IP address of the end device & subnet mask.

Click on Router:
for Router 0 → CLI
(setting up fast ethernet)
R0 (config)# interface fastethernet 0/0
R0 (config-if)# ip address 10.0.0.1 255.0.0.0
R0 (config-if)# no shutdown.
R0 (config-if)# exit

(setting up serial connection)
R0 (config)# interface serial 2/0
R0(config)# ip address 20.0.0.1 255.0.0.0
R0 (config-if)# encapsulation PPP
R0 (config-if)# clock rate 64000
R0 (config-if)# no shut down
R0 (config-if)# exit

Similarly, we set up the IP's of R1 and R2 while the setup of fast ethernet remains the same, the setting up of serial connections has 2 extra lines (encapsulation PPP, clock rate 64000).
Clock rate 64000 must only be written if the serially connected port shows a ⓒ symbol.

Thus, we write the clock rate command for R0 serial 2/0, R1 serial 3/0.

After this step, all the connections must have been turned to green.

2.) To enable IP routing by configuring OSPF routing protocol in all routes.

Router R0 → CLI

R0(config)# router ospf 1
R0(config-router)# router-id 1.1.1.1
R0(config-router)# network 10.0.0.0

R0(config-router)# network 20.0.0.0

R0(config-router)# exit.

Similarly do the same for R1 and R2 and specify the area numbers and ip addresses with subnet masks in CLI.

3.) Once the setting up of networking area is done, we configure loopback address to the routers:-

R0(config-if)# interface loopback 0
R0 (config-if)# ip add 172.16.1.252 255.255.0.0
R0 (config-if)# no shutdown

R1 (config-if)# interface loopback 0
R1 (config-if)# ip add 172.16.1.253 255.255.0.0
R1 (config-if)# no shutdown

R2 (config-if)# interface loopback 0
R2 (config-if)# ip add 172.16.1.254 255.255.0.0
R2 (config-if)# no shutdown.

4) On checking routing table of R2 using:- show ip route, we can see that R2 doesn't know about area 3.

Gateway of last resort is not set.
O IA 20.0.0.0/8[110/128] via 30.0.0.1
serial 1/0.
C 40.0.0.0/8 is directly connected,
fastethernet 0/0
C 30.0.0.0/8 is directly connected,
serial 2/0.

Since, R2 doesn't know about area 3, we have to create a virtual link between R0 and R1.

5. Creating virtual link between R1, R0
In Router R0,
R0(config)# router ospf 1
R0(config.router)# area 1 virtual link 2.2.2.2
R0(config.router)# exit.

In Router R1
R1(config)# router ospf 1
R1(config.router)# area 1 virtual link 1.1.1.1
R1(config.router)# exit.

6) Now, check routing table of R2.
Once all these steps are completed, the messages can be pinged across the devices.

# Observations :

In Router R2 :

Router # show ip route

O IA 20.0.0.0/8[110/128] via 30.0.0.1,
00:57:25, serial 2/0

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

O IA 10.0.0.0/8 [110/129] via 30.0.0.1
00:57:25, serial 2/0

C 30.0.0.0/8 is directly connected,
serial 2/0

C 172.16.0.0/16 is directly connected,
loopback0.

Similarly the output is shown for
Router 0 and 1.

Ping output :
(from PC0 to PC1)

PC0 → command prompt

C:\\> ping 40.0.0.10

Pinging 40.0.0.10 with 32 bytes of data

Request timed out

Reply from 40.0.0.10: bytes =32, time=21 ms TTL=225

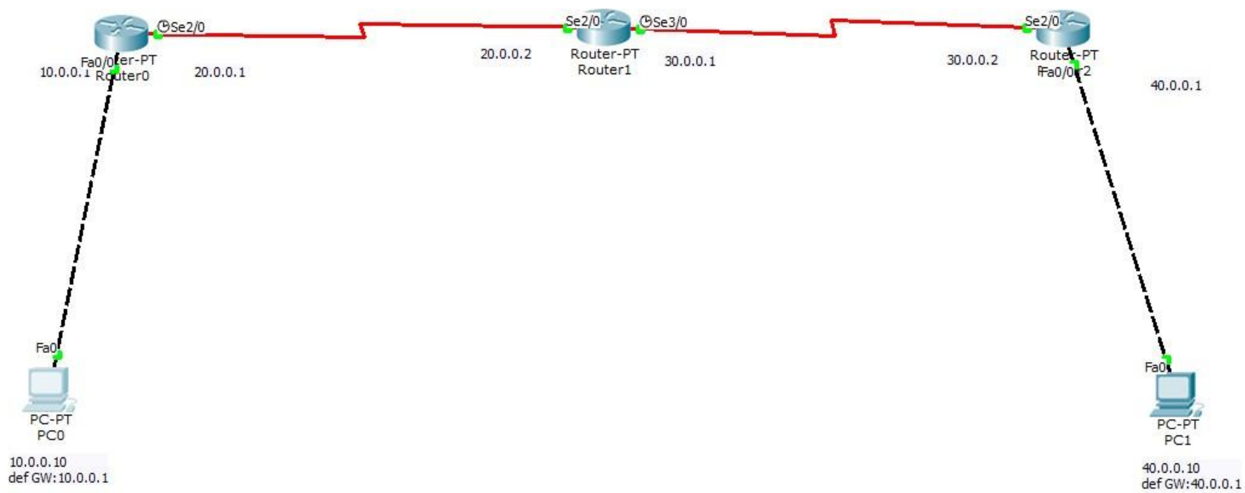
Reply from 40.0.0.10: bytes =32, time =2 ms TTL=225

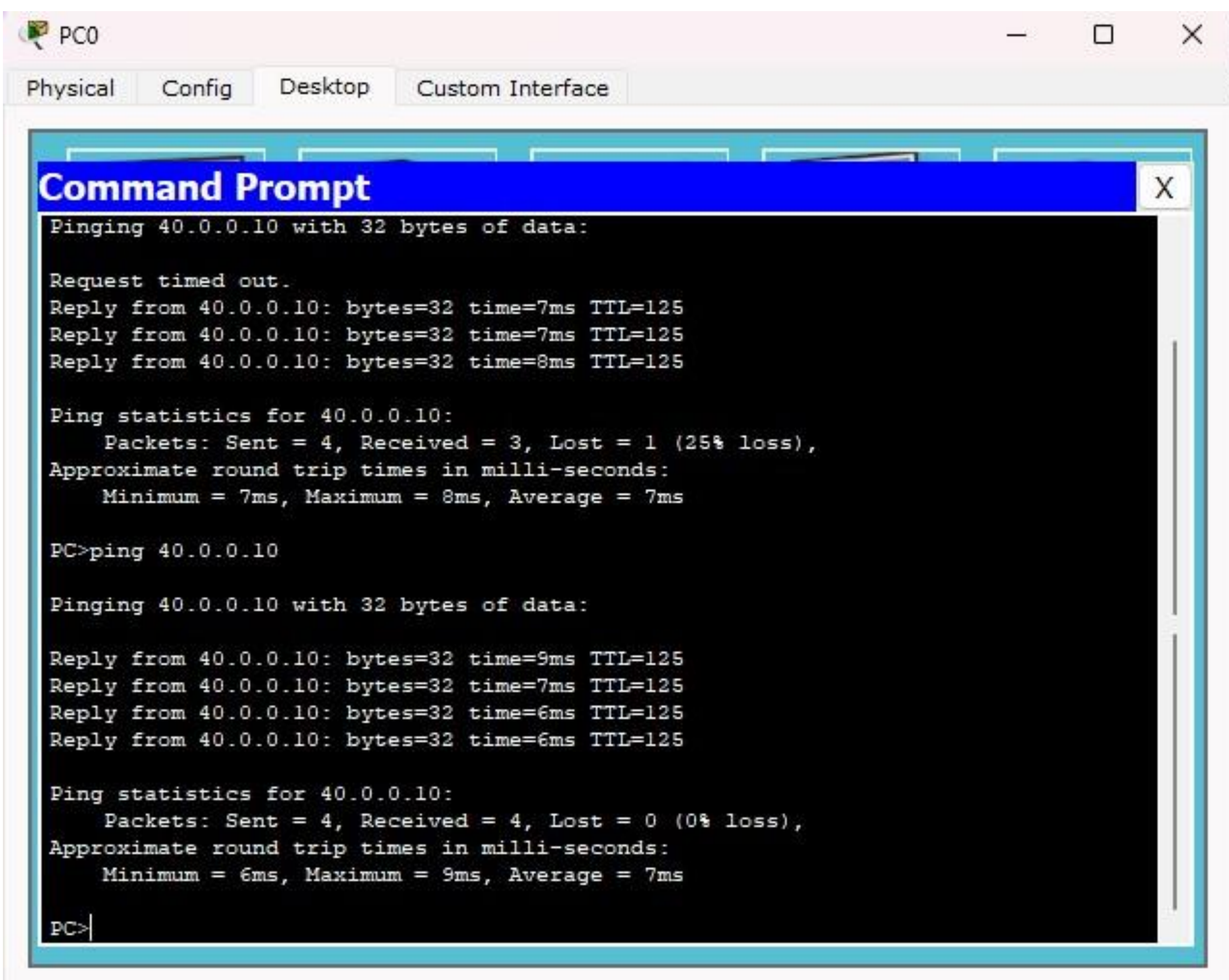
Reply from 40.0.0.10: bytes = 32, time =28ms TTL=128

Pinging statistics for 40.0.0.10

Packets: sent = 4, Received= 3, Loss = 1 (25% loss)

**Screen Shots:**



**Program 9**
**Aim:** Configure Web Server, DNS within a LAN.

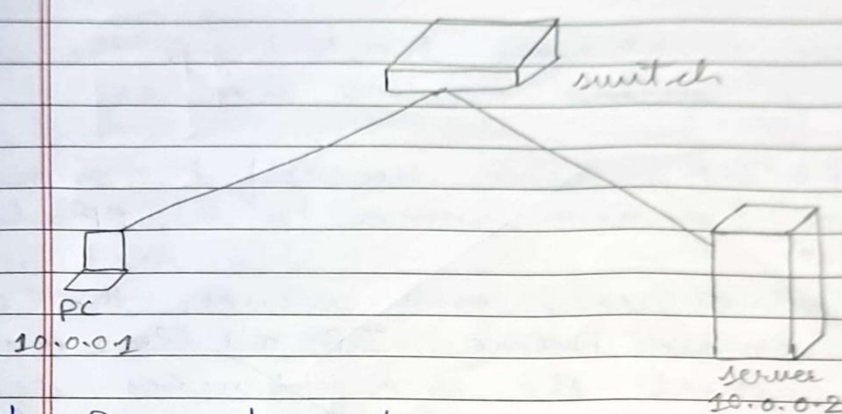
**Topology , Procedure and Observation:**

18 Dec
Page 32

Experiment -10
DNS

# Aim :- Configure Web server, DNS within a LAN.

# Topology :-




# Procedure :-

1.) ~~Set up the LAN as per the topology mentioned above and configure the devices accordingly.~~

2.) Go to Server ⇒ Services ⇒ DNS :
Name : bmsce [Domain name]
Address :- 10.0.0.2
Add the mapping of domain name to the address.

3.) Go to PC → Config → Global → setting → DNS Server : 10.0.0.2

[the server that provides the DNS mapping].

4.) Go to PC → Desktop → Web Browser
Type the URL: http://bmsce

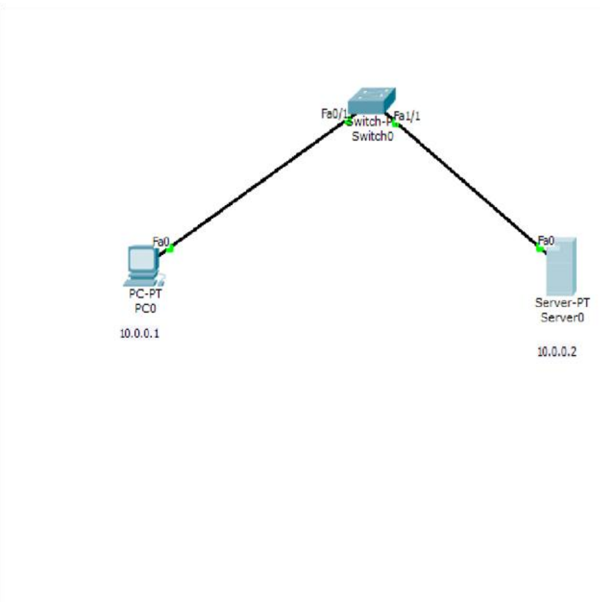
# Observations :-

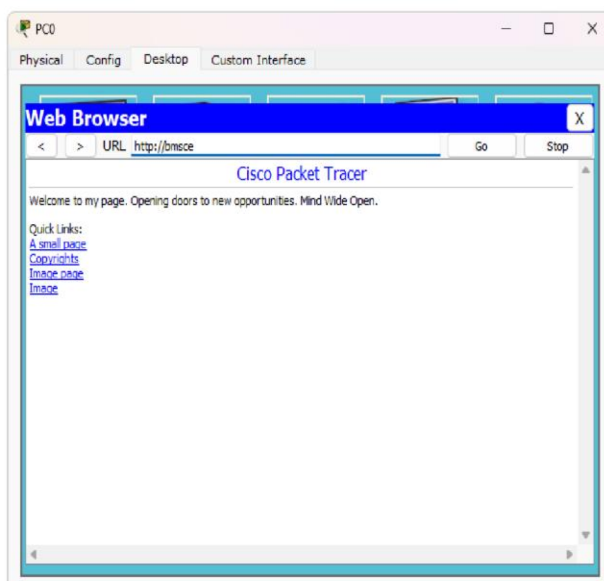
1.) The webpages hosted by the server were visible on the browser.

2.) The DNS was successful in mapping the domain name to the IP address.

3.) DNS server is a server that contains a Domain Name : IP address mapping to which the end devices send requests to map the Name to IP addresses.

**Screen Shots:**



**Program 10**

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

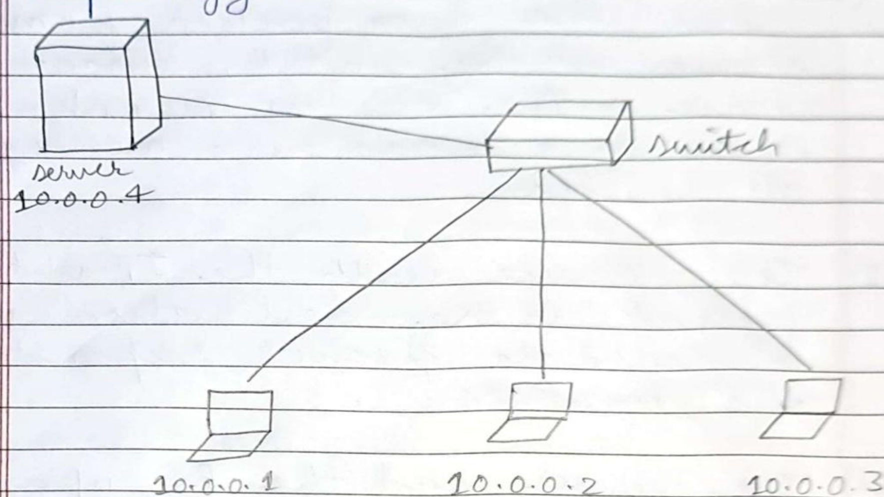
**Topology , Procedure and Observation:**

18 Dec Page 35

Experiment - 9
ARP

# Aim :- To construct a simple LAN & understand the concept and operation of Address Resolution Protocol (ARP).

# Topology :-




# Procedure :

1.) Create the topology as shown above.
2.) Configure the PC's and the server.
3) Click on the Inspect Mode (Q), then click on the end devices and open ARP tables.
4.) Send a data packet from any end device say server to other end device say 10.0.0.3 PC.

5.) Open simulation mode to capture each step of data transfer.

# Observations :-

1.) The ARP tables of all end devices are initially empty.

2.) When the data packet from server arrives at the switch, since the source MAC address is unknown, it sends a broadcast message to all devices.

3.) The device with the IP address present in the destination address of the data packet responds to the messages.

4.) The server and the PC updates their ARP tables matching the IP addresses to MAC addresses.

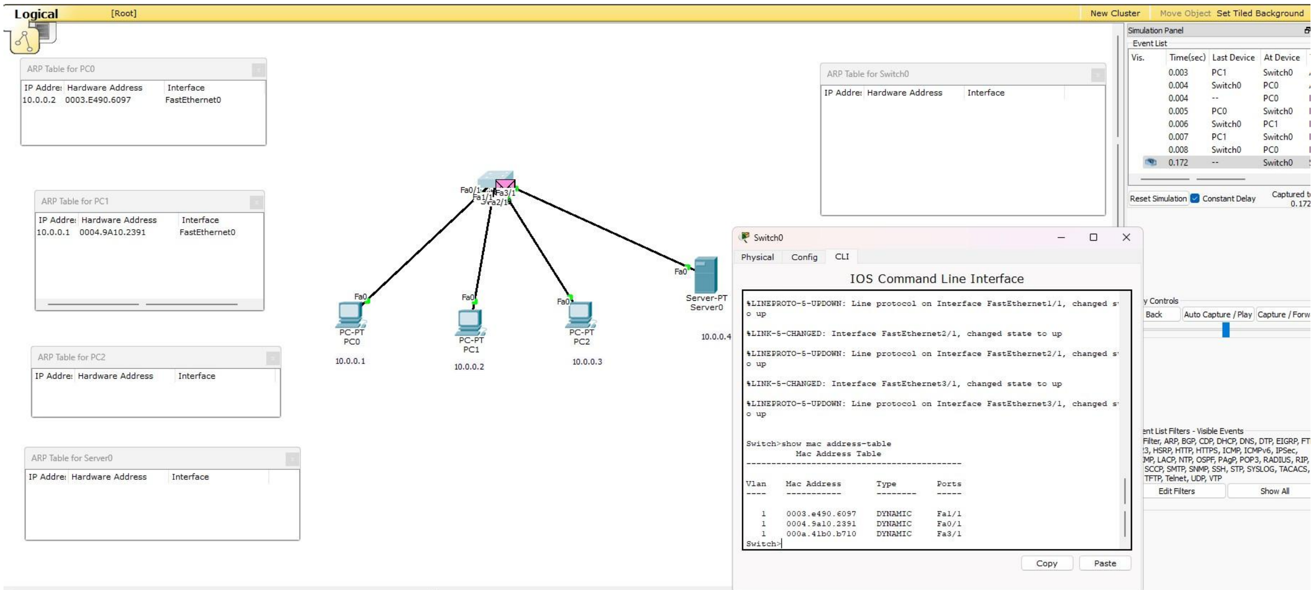
5.) Over time, the ARP tables grows as data packets are sent.

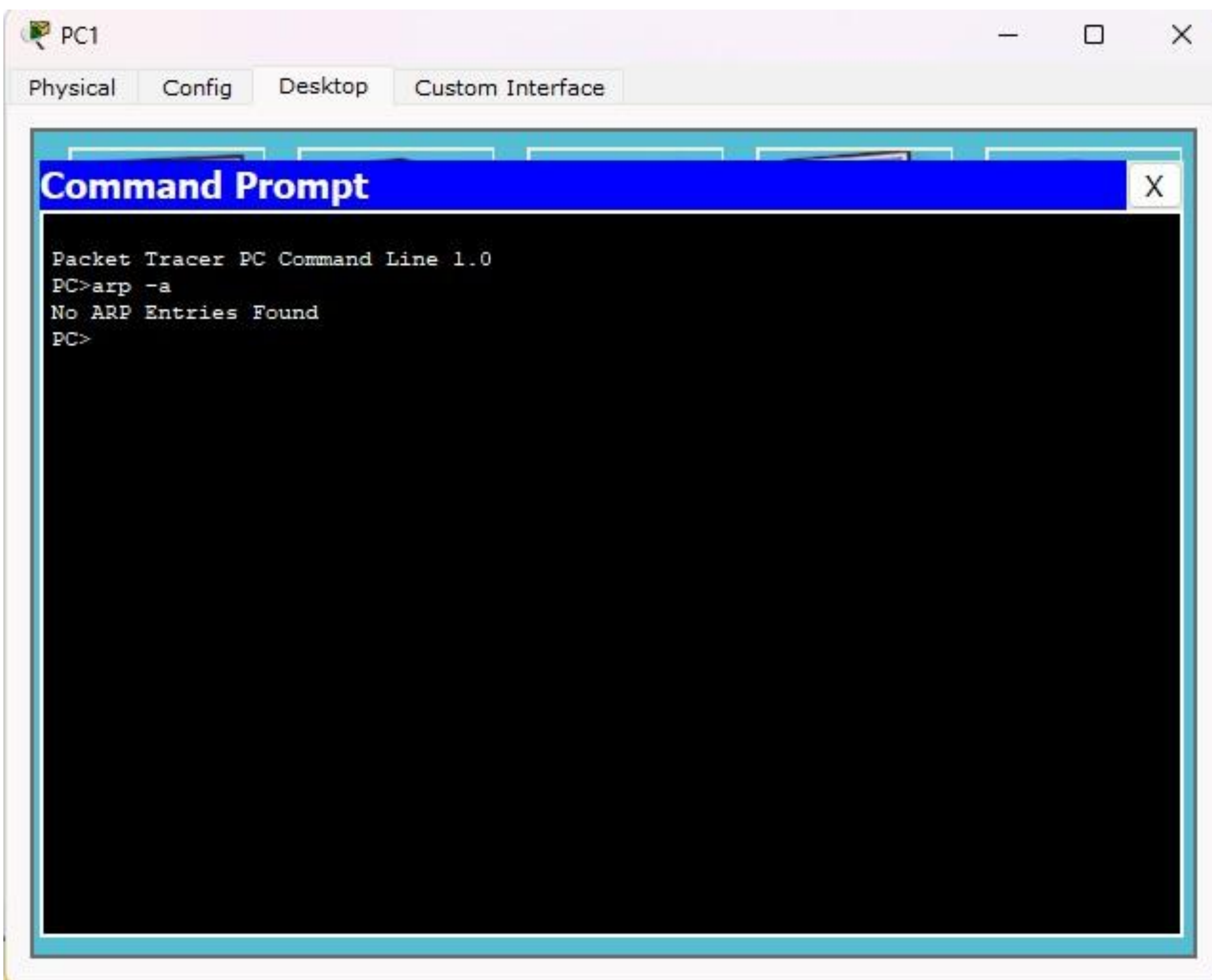
6.) The MAC table of the switch which was initially empty updates its MAC table gradually too.

ARP Table for 10.0.0.4

| IP Address | Hardware Address | Interface |
|---|---|---|
| 10.0.0.3 | 0001.C726.97E5 | FastEthernet0 |

**Screen Shots:**



**Program 11**

**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 Observation:**

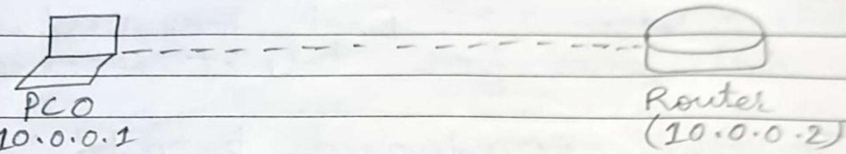
Date 10 Dec Page 39

Experiment - 11

TELNET

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

# Topology :-




# Procedure :-

1.) Create the topology as given above and configure the devices accordingly

2.) Commands in Router :-

Router > enable
Router # config terminal
Router (config)# hostname R1
R1 (config)# enable secret 1234
(enable password)
R1 (config)# interface fastethernet 0/0
R1 (config-if)# ip address 10.0.0.2 255.0.0.0
R1 (config-if)# no shut.

R1 (config-if)# line vty 0 3
R1 (config-line)# login.

% login disabled on line 194, untill
'password' is set.
R1(config-line)# password 4321
R1(config-line)# exit

user access verification password

R1(config)# exit
R1 # wr -> to perform write operation
Building configuration.
[OK]

Note: vty 0 3: First for virtual terminal lines for telnet access.

3.) In PC: command prompt:
- First try pinging to see if the devices are connected.

PC> telnet 10.0.0.2
Trying 10.0.0.2 open

User Access Verification:

Password : 4321
Password : 4321
R1> enable
Password : 1234

R1# show ip route.

C: 10.0.0.0/8 is directly connected, Fast, Ethernet 0/0.
R1#

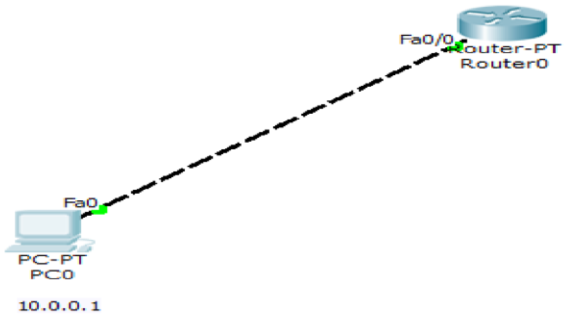
# Observations :

1.) The admin in PC is able to run commands as run in router CLI and see the results from PC.

2.) Telnet allows users to establish a remote session with another device like router, over a TCP/IP network.

3.) Using Telnet, we can access and control the remote device's CLI as if you were physically connected to it.

**Screen Shots:**




**Command Prompt**

```
Packet Tracer PC Command Line 1.0
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=255
Reply from 10.0.0.2: bytes=32 time=0ms TTL=255
Reply from 10.0.0.2: bytes=32 time=0ms TTL=255
Reply from 10.0.0.2: bytes=32 time=0ms TTL=255

Ping statistics for 10.0.0.2:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 0ms, Average = 0ms

PC>telnet 10.0.0.2
Trying 10.0.0.2 ...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 12**
**Aim:** To construct a VLAN and make the PC's communicate among a VLAN .

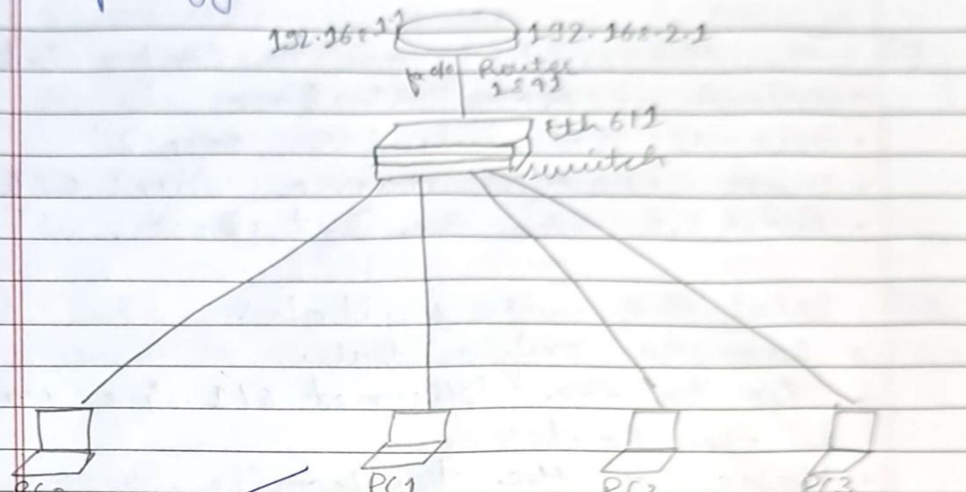
**Topology , Procedure and Observation:**



Experiment - 8
VLAN

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

# Topology :-




| | PC0 | PC1 | PC2 | PC3 |
|---|---|---|---|---|
| IP: | 192.168.1.2 | 192.168.1.3 | 192.168.2.2 | 192.168.2.3 |
| Gateway: | 192.168.1.1 | 192.168.1.1 | 192.168.2.1 | 192.168.2.1 |

# Procedure :-

1.) Place a 1841 Router, a switch and 4 PCs.

2.) Connect the four PCs to the switch via fast ethernet.

3.) Since only 4 fastethernet ports are available in the switch, we have to add an ethernet port.

4.) To add an ethernet port:
- Switch off the Power button of the switch.
- Add the ethernet port to the switch.
- Switch on the power button.
- Connect the router to the switch via ethernet 6/1.

5.) In the switch, go to Config Tab and:-
- select VLAN Database.
- Give VLAN number say '2'
- Give VLAN name say 'CSEISE'.
- Add it to the Database.

6.) Select the the switch:
- Go to config tab.
- Go to the Ethernet 6/1 i.e. connected to the Router.
- Make it the trunk.

7.) Configure the PCs as shown in the topology.

8.) Select switch:
- Go to config tab.
- Go to FastEthernet 2/1
- Set VLAN number as '2' i.e. 'cseise'
- Similarly set VLAN 2 for fastethernet 3/1 interface too.

9.) Configure the Router:-
Router (config)# interface FastEthernet 0/0

Router (config-if)# ip address 192.168.1.1
255.255.255.0
Router (config-if)# no shut
Router (config-if)# exit.

Now, to configure the router's VLAN interface :-
Router (config)# interface fastethernet 0/0.1
Router (config-subif)# encapsulation dot1q 2
Router (config-subif)# ip address 192.168.1.1
255.255.255.0
Router (config-subif)# no shut.
Router (config-subif) # exit.

10.) Ping devices within the same VLAN and to the devices of different VLAN.

# Observations :-

1.) When devices are pinged within same VLAN:
- Pinging 192.168.1.3 from 192.168.1.2
  - the data packet doesn't go to the router.
  - the switch forwards the packet without the need of the router.

2.) When a device pings a device of another VLAN:
- Pinging 192.168.2.3 from 192.168.1.2

- The data packet's journey is as follows:

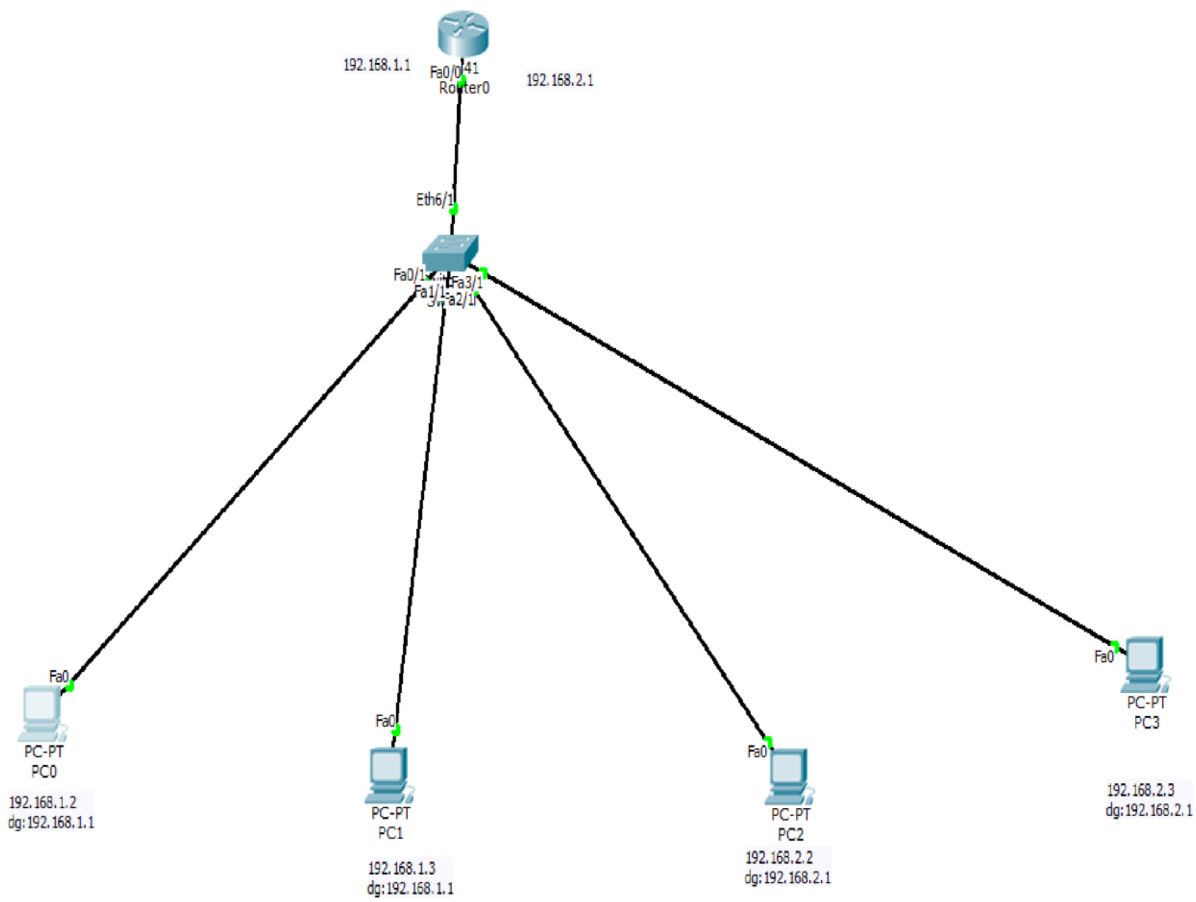
192.168.1.2 → switch → Router → Switch → 192.168.2.3

3.) VLANs divide a single switch into multiple logical switches.
- Devices in one VLAN cannot directly communicate with devices in another VLAN without a Router.

4.) Traffic Isolation:
- Each VLAN maintains its own broadcast domain.
- broadcasts sent by devices in one VLAN do not reach devices in another VLAN.

5.) VLAN trunking allowed switches to forward frames from different VLANs over a switch link called the trunk.
- This is done by adding an additional header information called tag to the ethernet frame - VLAN tagging.

**Screen Shots:**



**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=0ms TTL=127
Reply from 192.168.2.2: bytes=32 time=4ms 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 = 4ms, Average = 1ms

PC>ping 192.168.2.2

Pinging 192.168.2.2 with 32 bytes of data:

Reply from 192.168.2.2: bytes=32 time=0ms TTL=127
Reply from 192.168.2.2: bytes=32 time=0ms TTL=127
Reply from 192.168.2.2: bytes=32 time=2ms 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 = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 2ms, Average = 0ms

PC>ping 192.168.2.3

Pinging 192.168.2.3 with 32 bytes of data:

Request timed out.
Reply from 192.168.2.3: bytes=32 time=3ms TTL=127
Reply from 192.168.2.3: bytes=32 time=2ms TTL=127
Reply from 192.168.2.3: bytes=32 time=1ms TTL=127

Ping statistics for 192.168.2.3:
    Packets: Sent = 4, Received = 3, Lost = 1 (25% loss),
Approximate round trip times in milli-seconds:
    Minimum = 1ms, Maximum = 3ms, Average = 2ms

PC>ping 192.168.2.3

Pinging 192.168.2.3 with 32 bytes of data:

Reply from 192.168.2.3: bytes=32 time=0ms TTL=127
Reply from 192.168.2.3: bytes=32 time=0ms TTL=127
Reply from 192.168.2.3: bytes=32 time=2ms TTL=127
Reply from 192.168.2.3: bytes=32 time=0ms TTL=127

Ping statistics for 192.168.2.3:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 2ms, Average = 0ms

PC>
```

**Program 13**

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

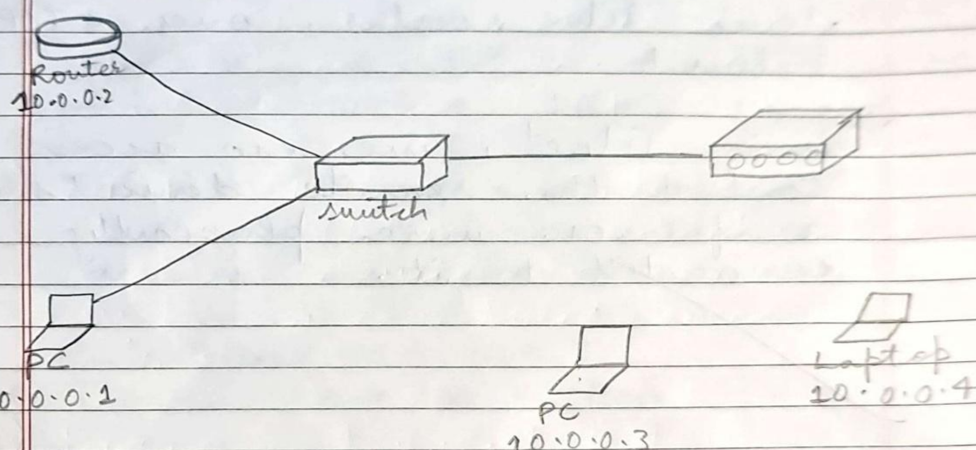
**Topology , Procedure and Observation:**

18-Dec-2023

Experiment - 12
WLAN

# Aim :- To construct a Wireless LAN and make the nodes communicate wirelessly.

# Initial Topology :-




# Procedure :-

1.) create the topology as given above and configure the devices.

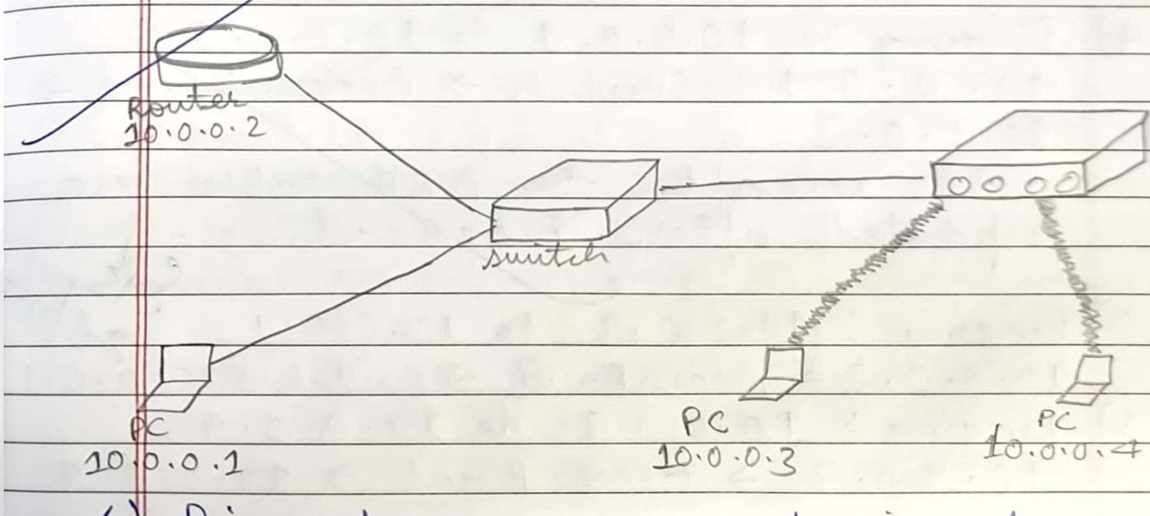
2.) configure Access Point :
click Access Point → config → Port 1 :
SSID : bmsce
Select ① WEP
Set key : 1234567890

3.) Configure PC and laptops with Wireless standards:
- Switch off device.
- Drag the existing PT-HOST-NM-LAN to the component listed in the LHS of Physical.
- Drag WMP300N Wireless interface to the empty port.
- Switch ~~of~~ on the device.

4.) In the config tab, a new wireless interface was added.

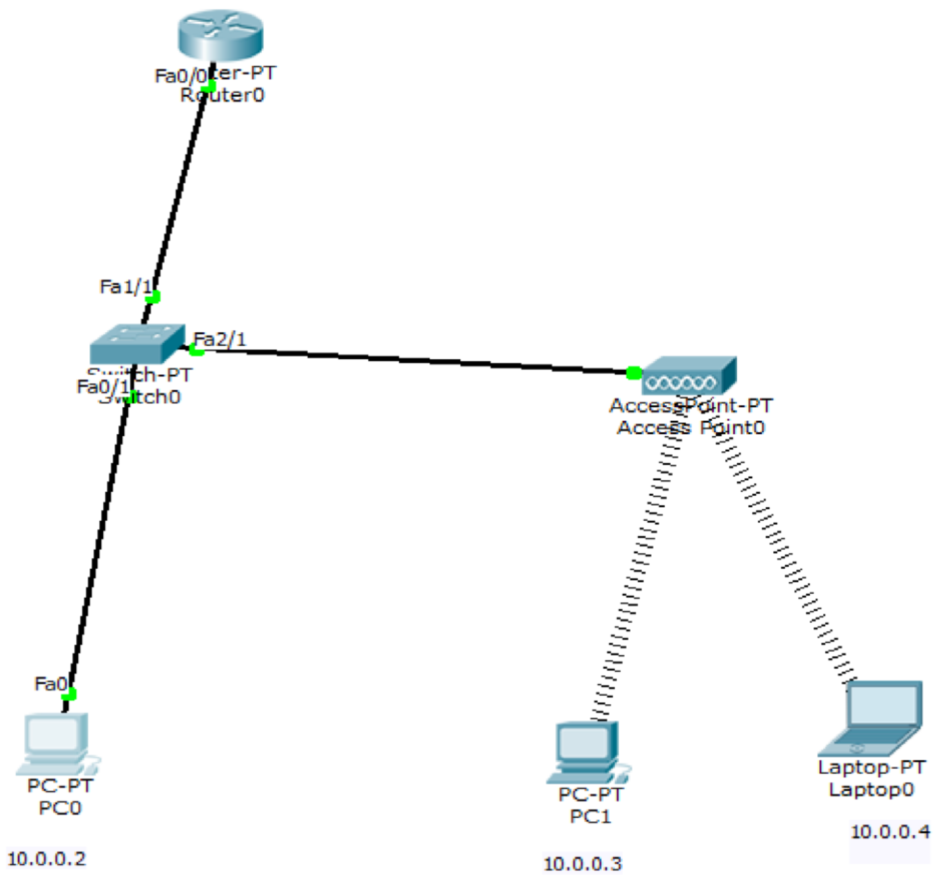
5.) Configure the device by entering SSID, WEP, WEP key, IP addresses and Gateway.

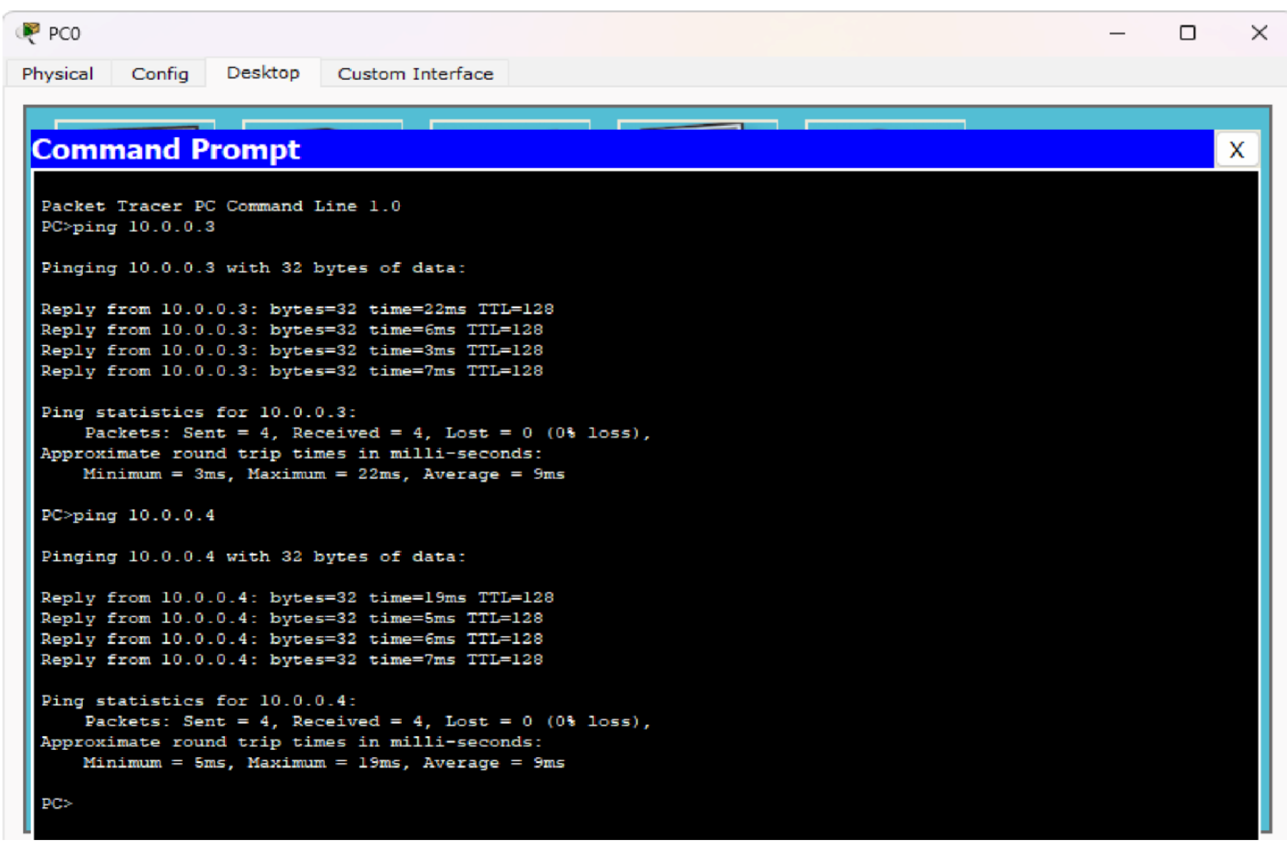
Topology after Wireless Configuration:




6.) Ping from every device to every other device to check for connection.

**Screen Shots:**



# Cycle-B

**Program 1**

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

**Code and Output:**

01-Jan-2025

Experiment - 14

```
# AIM: Implementation of CRC -
# Code:

def XOR (a, b):
    result = []
    for i in range (1, len(b));
        if a[i] == b[i];
            resut . append ('0');
        else :
            resut . append('1'):

    return '.'. join ('result')

def moddiv (dividend, divisor):
    pick = len (divisor)
    temp = dividend [0 : pick]
    while pick < len (dividend);
        if temp[0] == '1';
            temp = XOR (divisor, temp)+
            dividend [ pick ]
        else :
            temp = XOR ('0' * pick, temp)
            + dividend [pick ]
        pick += 1

    if temp [0] == '1';
        temp = XOR (divisor, temp)
    else ;
        temp = XOR ('0' * pick, temp)
```

```
        checkword = temp
        return checkword

def encode Data (data, key):
    l_key = len(key)
    append_data = data + '0'*(l_key-1)
    remainder = mod2div (append_data, key)
    codeword = data + remainder
    print ("Remainder ", remainder)
    print (" Encode Data (Data +Remainder)",
      "codeword)

data = "100100"
key = "1101"
encode Data (data, key)
```

# Output :

Sender site :---- ..
Remainder : 001

Encode Data (Data + Remainder) : 100100001

Receiver side :
correct message received.

**Output Screenshot:**

```
PS C:\Users\HP\OneDrive\Desktop\cn lab> python crc-ccitt.py
Enter the original bitstream (e.g., 11010011101100): 11010011101100
Transmitted bitstream with CRC: 1101001110110010100110000100100
Enter the received bitstream for verification: 1101000000000010100001
Error detected! CRC invalid.
```

Program 2

Write a program for congestion control using Leaky bucket algorithm.

**Code and Output:**

Cycle-2

Experiment -13

Leaky Bucket Algorithm.

In the network layer, before the network can make quality of service guarantees, it must know what traffic is being guaranteed, one of the main causes of congestion is that traffic is often

There are two types of traffic shaping :-

1.) Leaky Bucket
2.) Token Bucket.

For Example, let n = 1000
packet = 200 700 500 450 400 200
since n > size of the packet at the head of the queue i.e. n > 200.

Therefore, n = 1000 − 200 = 800
packet size of 200 is sent into the networks : 200 700 500 450 400

Now again, n > size of the packet of the head of the queue i.e. n > 400
Therefore, n = 800 − 400 = 400

# Code :

```
#include <stdio.h>
int main (){
    int incoming, outgoing, bucket size,
     n, store =0;
    printf ("Enter the bucket size,
     outgoing node, and the no. of
     stores:");
    scanf ("%d, %d %d", & bucket-size,
     & outgoing, & n);

    while (n! = 0){
        printf ("Enter the incoming
          packet size:");
        scanf ("%d", & incoming);
        printf ("Incoming packet size %d
          \n", incoming);

     if incoming in (bucket-size - store){
        store += incoming;
        printf ("Bucket buffer size %d
          out of %d \n", store, buck-size);
     } else {
        printf ("Dropped %d no of packets
          \n", incoming - (buck-size) - store));
        printf ("Bucket Buffer size %d
          out of %d \n", store, buck-size);
        store = buck_size;
     }
     store = store - outgoing;
     printf ("After outgoing %d bytes
      left out of %d in buffer \n", store,
```

```
        buck-size);
        m--;
    }
}
```

# Output:

Enter bucket size, outgoing rate and no. of inputs : 100 20 3
Enter the incoming packet size : 30
Incoming packet size 30

Bucket suffix size 30 out of 100
After outgoing 10 bytes of left out of 100 in buffer.

Enter the incoming packet size : 50
Incoming packet size : 50
Bucket suffix size 60 out of 100
After outgoing 40 bytes left out of 10 in buffer.

Enter the incoming packet size : 80
Incoming packet size : 80
Dropped 20 no. of packets.

Bucket Buffer size 40 out of 100
After outgoing 80 bytes of left out of 100 in buffer.

**Output Screenshot:**

```
PS C:\Users\Dell\OneDrive\Desktop\code> python leakybucketalgorithm.py
Enter no of queries:10
Enter bucket size:5
Enter input packet size:4
Enter output packet size:6
Bucket size=4out of bucket size=5
Bucket size=2out of bucket size=5
Bucket size=0out of bucket size=5
Bucket size=-2out of bucket size=5
Bucket size=-4out of bucket size=5
Bucket size=-6out of bucket size=5
Bucket size=-14out of bucket size=5
```

**Program 3**

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.

**Code and Output:**

01-Jan-2025

Experiment - 15

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

⇒ Code

ServerTCP.py

```
from socket import *
serverName = '127.0.0.1'
serverPort = 12000

serverSocket = socket(AF_INET, SOCK_
  STREAM)
serverSocket.bind((serverName, server
 Port))
server.Socket.listen(1)

while 1:
    print ("Server ready to receive")
    connectionSocket, addr = serverSocket.
     accept()

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

  file = open(sentence, "r")
  l = file.read(1024)
```

```
connection Socket . send (l. encode ())
print ('\n Sent contents of ' + sentence)
file. close ()
connection Socket. close ()
```

# ClientTCP.py

```
from socket import *

server Name = '127.0.0.1'
server Port = 14000

client Socket = socket (AF_INET, SOCK_STREAM)
client Socket . connect ((server Name,
 server Port)

sentence = input ("\n Enter the file
 name: ")

client Socket. send (sentence . encode ())
filecontents = client socket . recv(1024).
 decode ()

print ('\n From Server : \n')
print (filecontents)
client Socket . close ()
```

⇒ Output :-
The server is ready to receive.
Send contents of Server TCP . py.
Reply from server:

**Output Screenshots:**

```
PS C:\Users\Dell\OneDrive\Desktop\code> python tcpserver.py
The server is ready to receive

Sent contents of example.txt
The server is ready to receive
```

```
PS C:\Users\Dell\OneDrive\Desktop\code> python tcpclient.py

Enter file name: example.txt

From Server:

Hello, this is a sample file.
It is used for testing the TCP server.
```

**Program 4**

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

**Code and Output:**

02-Jan-2025

Experiment - 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:

ServerUDP.py

```
from socket import *
serverPort = 12000
serverSocket = socket(AF_INET, SOCK_DGRAM)
serverSocket.bind(("127.0.0.1", serverPort))

print ("The server is ready to receive")

while 1:
    sentence, clientAddress = serverSocket.
     recvfrom (2048)
    sentence = sentence.decode ("utf-8")
    file = open (sentence, "r")
    con = file.read (2048)

    serverSocket.sendto (bytes (con, "utf-8"),
      clientAddress)

    print ('\n Sent contents of', end = ' ')
    print (sentence)

    file.close()
```

# clientUdp.py

```
from socket import *

serverName = "127.0.0.1")
serverPort = 12000
clientSocket = socket(AF_INET, SOCK_DGRAM)

sentence = input("\n Enter file name: ")

clientSocket.sendto(bytes(sentence, "utf-8"),
 (serverName, serverPort))

filecontents, serverAddress = clientsocket.
 recvfrom(2048)
print('\n Reply from server:\n')
print (filecontents.decode("utf-8"))

client.Socket.close()
```

# Output :-

The server is ready to receive.
Sent contents of server UDP.py
Server is ready to receive } server side

Enter file name: serverUDP.py
Reply from server: } client side

**Output Screenshot:**

```
PS C:\Users\Dell\OneDrive\Desktop\code> python udpserver.py
The server is ready to receive

Sent contents of  example.txt
```

```
PS C:\Users\Dell\OneDrive\Desktop\code> python udpclient.py

Enter file name:  example.txt

Reply from Server:

Hello, this is a sample file.
It is used for testing the TCP server.
```

**Program 5**

Tool Exploration – WireShark

02-Jan-2025

Experiment - 17

Tool Exploration - Wireshark.

Wireshark is a powerful and widely used network protocol analyser. It allows us to capture and inspect data packets travelling over network in real-time, making it a crucial tool for studying computer networks, troubleshooting network issues and understanding the protocols.

Key Features :-

1.) Packet capture :- captures live network traffic from various interfaces (ex: ethernet, wi-fi)

2.) Protocol Analysis : supports hundreds of protocols (ex: TCP, UDP, HTTP, FTP)

3.) Filtering :- Offers powerful filters to isolate specific packets or traffic types.

4.) Visualization : Displays packets details with hierarchial layers (ethernet, IP, TCP/UDP).

# Use Cases of Wireshark.

1.) Network Troubleshooting.
Diagnosing slow network speeds
Identifying nerks a misconfigurations.

2.) Security Analysis:
Detecting malicious traffic / intrusions

3.) Protocol study.
Understanding packet structures and communication flows.

# common filters:

1.) http: shows only http traffic.

2.) tcp port == 8: shows traffic on TCP port 80

3.) ip addr == 192.168.1.1: shows packets to and from specific IP.

4.) UDP: shows only UDP traffic.