

VISVESVARAYA TECHNOLOGICAL UNIVERSITY

“JnanaSangama”, Belgaum -590014, Karnataka.



LAB REPORT

on

COMPUTER NETWORKS

Submitted by

ADNAN ANWAR (1BM21CS008)

in partial fulfillment for the award of the degree of
BACHELOR OF ENGINEERING
in
COMPUTER SCIENCE AND ENGINEERING



B.M.S. COLLEGE OF ENGINEERING

(Autonomous Institution under VTU)

BENGALURU-560019 JUN-2023 to SEP-2023

**B. M. S. College of Engineering,
Bull Temple Road, Bangalore 560019**
(Affiliated To Visvesvaraya Technological University, Belgaum)
Department of Computer Science and Engineering



CERTIFICATE

This is to certify that the Lab work entitled “**COMPUTER NETWORKS**” carried out by **ADNAN ANWAR(1BM21CS008)**, who is bonafide student of **B. M. S. College of Engineering**. It is in partial fulfillment for the award of **Bachelor of Engineering in Computer Science and Engineering** of the Visvesvaraya Technological University, Belgaum during the year 2023. The Lab report has been approved as it satisfies the academic requirements in respect of a **Computer Networks - (22CS4PCCON)**work prescribed for the said degree.

Dr. Nandini Vineeth

Assistant Professor
Department of CSE
BMSCE, Bengaluru

Dr. Jyothi S Nayak

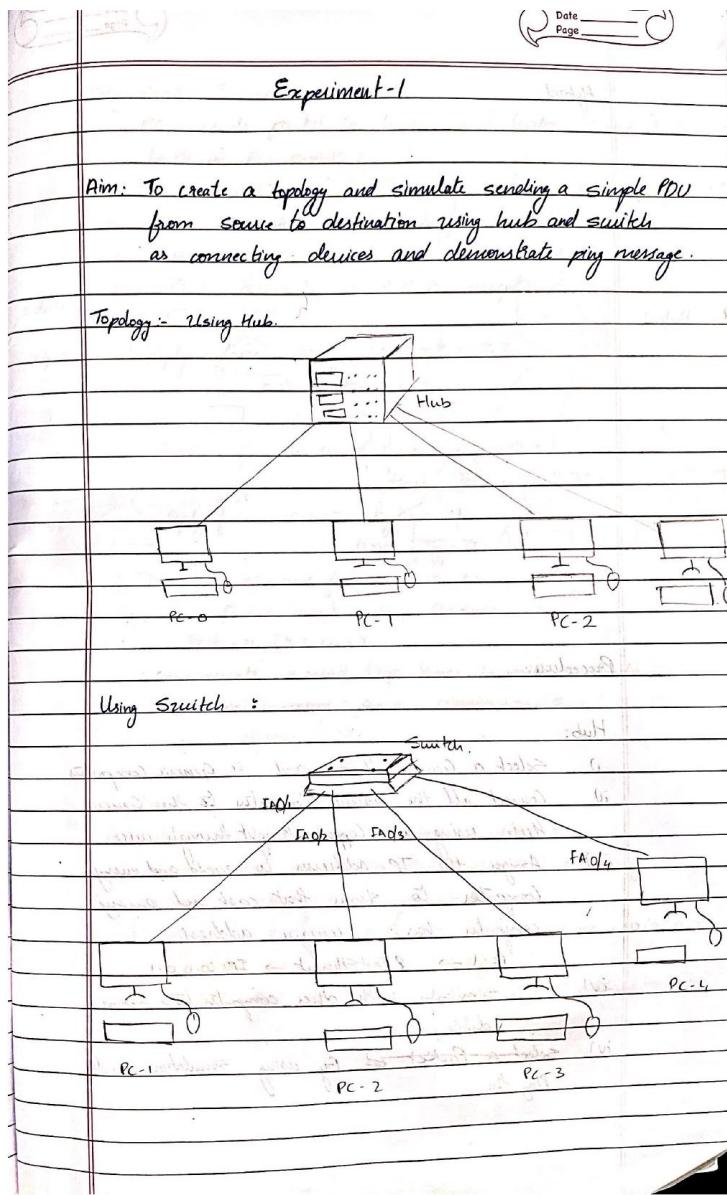
Professor and Head
Department of CSE
BMSCE, Bengaluru

Index

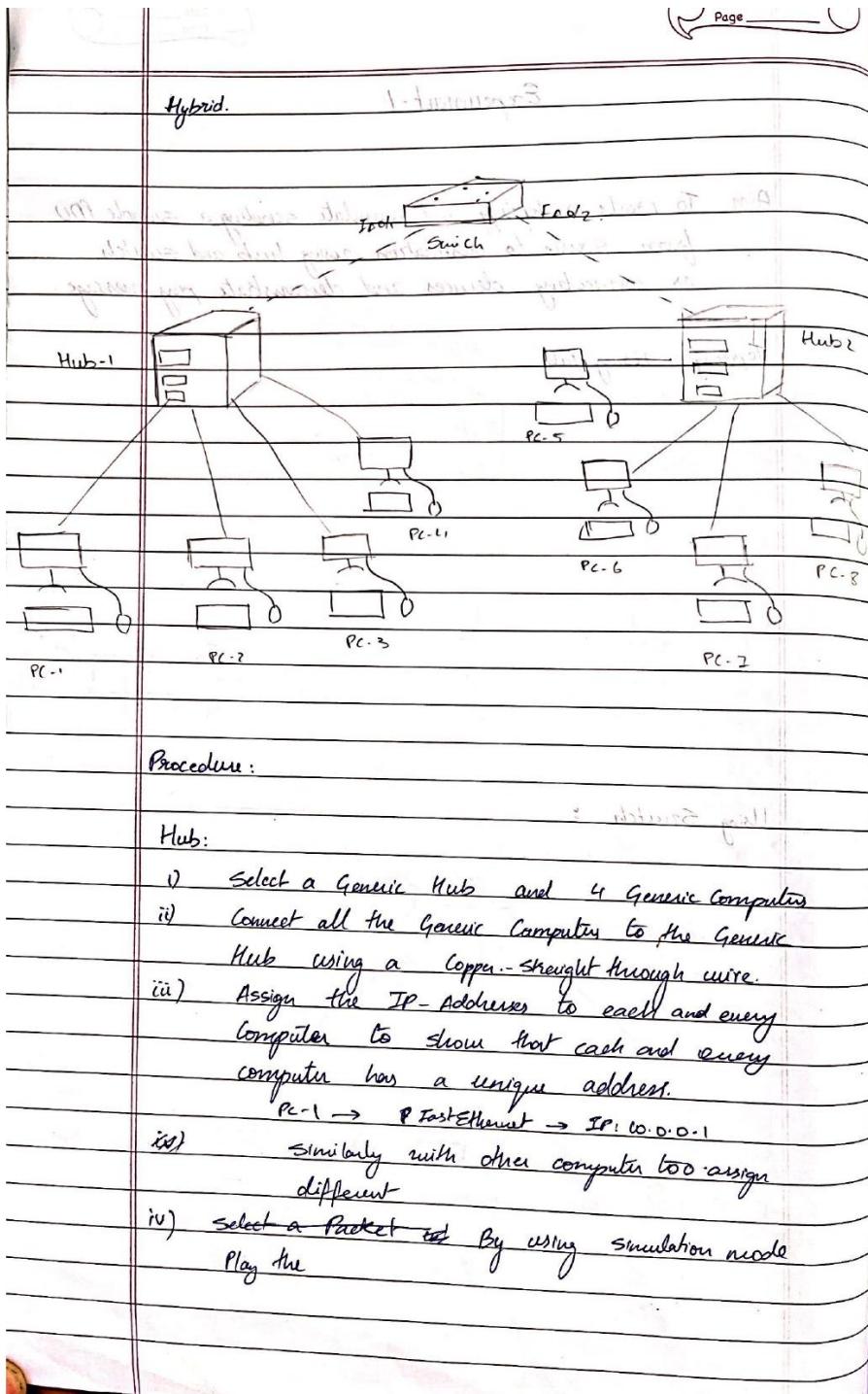
Sl. No.	Date	Experiment Title	Page No.
CYCLE 1			
1	15-06-23	Create a topology and simulate sending a simple PDU from source to destination using hub and switch as connecting devices and demonstrate ping message.	4
2	22-06-23	Configure IP address to routers in packet tracer. Explore the following messages: ping responses, destination unreachable, request timed out, reply	12
3	13-07-23	Configure default route, static route to the Router	19
4	13-07-23	Configure DHCP within a LAN and outside LAN	23
5	20-07-23	Configure Web Server, DNS within a LAN	30
6	20-07-23	Configure RIP routing Protocol in Routers	33
7	27-07-23	Configure OSPF routing protocol	37
8	03-08-23	To construct simple LAN and understand the concept and operation of Address Resolution Protocol (ARP)	42
9	10-08-23	To construct a VLAN and make the PC's communicate among a VLAN	45
10	10-08-23	Demonstrate the TTL/ Life of a Packet	48
11	10-08-23	To construct a WLAN and make the nodes communicate wirelessly	52
12	10-08-23	To understand the operation of TELNET by accessing the router in server room from a PC in IT office.	55
CYCLE 2			
13	17-08-23	Write a program for error detecting code using CRCCCITT (16-bits)	59
14	17-08-23	Write a program for congestion control using Leaky bucket algorithm	64
15	24-08-23	Using TCP/IP sockets, write a client-server program to make client sending the file name and the server to send back the contents of the requested file if present.	67
16	24-08-23	Using UDP sockets, write a client-server program to make client sending the file name and the server to send back the contents of the requested file if present	71
17	31-08-23	Tool Exploration -Wireshark	75

EXPERIMENT-1

Q) Create a topology and simulate sending a simple PDU from source to destination using hub and switch as connecting devices and demonstrate ping message.



Scanned with CamScanner



Scanned with CamScanner

classmate
Date _____
Page _____

Observation: In simulation mode.

- A PC sends packet to hub and hub sends it to both PC1 and PC2.
- PC1 dispatches the message when PC2 accepts it.
- PC2 sends the acknowledgement packet to hub.
- hub again sends it to PC0 and PC1.
- PC1 discards and PC0 accepts it.

Output:

- Reply from 10.0.0.2 bytes = 32 ms = 2ms
- RTT = 2ms
- Round trip time = 4ms
- Reply from 10.0.0.2 bytes = 32 ms = 2ms
- RTT = 2ms
- Round trip time = 4ms
- Reply from 10.0.0.2 bytes = 32 ms = 2ms
- RTT = 2ms
- Round trip time = 4ms

Ping statistics for 10.0.0.2.

Packet sent = 4, Received = 4, bytes = 32 ms = 2ms

Loss = 0% (0% loss)

Average round trip times in milli-seconds -

Minimum = 2ms, Maximum = 3ms, Average = 2.5ms

Switch configuration:

Procedure:

- Select a switch and 3 PC's.
- Connect the switch to the conditional ports using a copper straight through cable.
- Assign the IP addresses to the PC's - 10.0.0.4, 10.0.0.5, 10.0.0.6 respectively.
- Select the IP's and the source and destination to PC.

Scanned with CamScanner

Observation in Simulation:

- PC3 sends the packet to switch and it sends to both PC4 and PC5 in first round.
- PC4 rejects and PC5 accepts and sends acknowledgement packet to both PC3 and PC5
- PC4 discards it if PC3 accepts it
- Now because of switch PC5 denies packet only to PC5

Output: Reply from 10.0.0.5 bytes = 32 time = 0ms TTL = 125

Reply from 10.0.0.5 bytes = 32 time = 0ms TTL = 125

Reply from 10.0.0.5 bytes = 32 time = 0ms TTL = 125

Ping statistics from 10.0.0.5

Packets sent 4, received = 4, lost = 0 (0% loss)

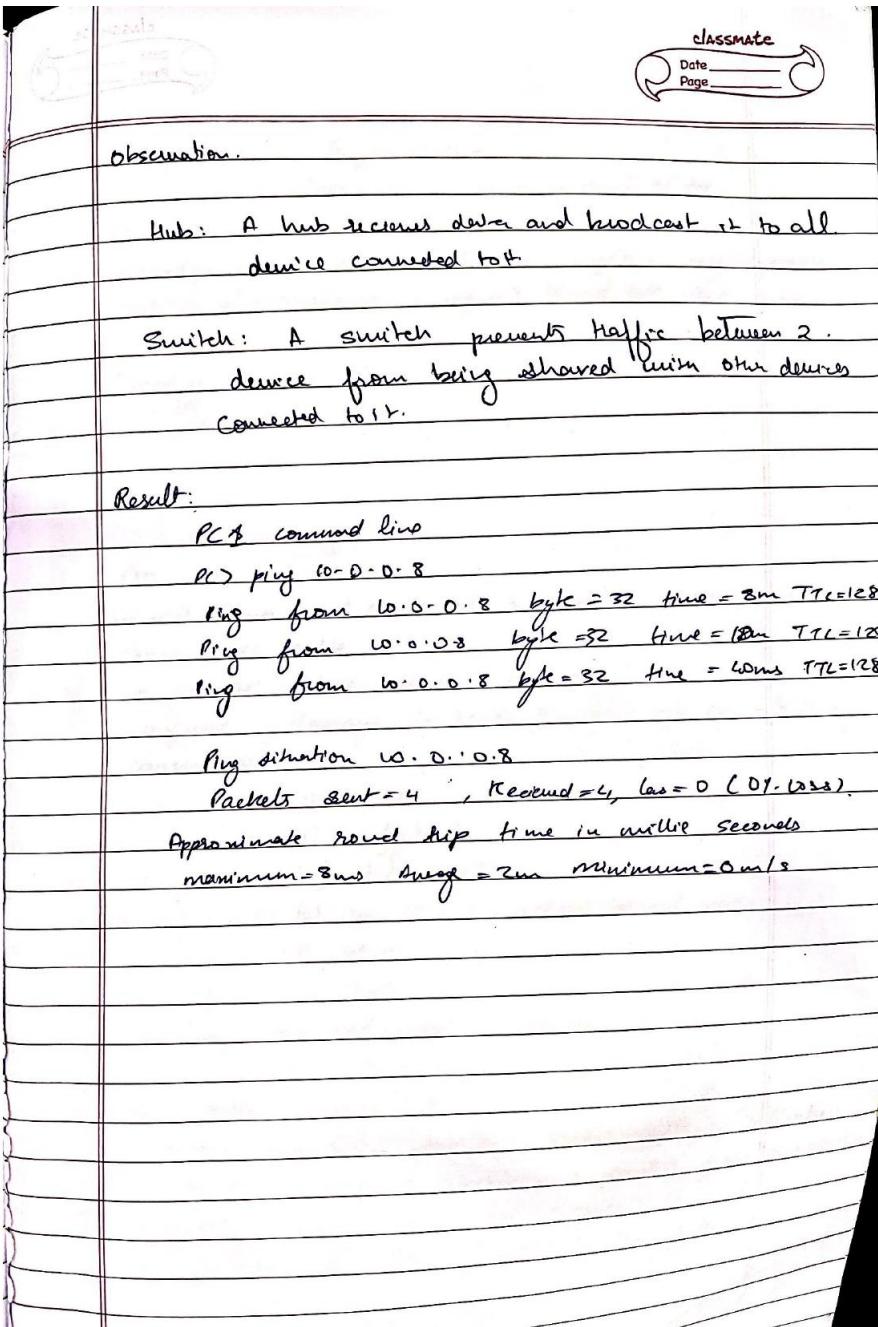
Approximate round trips in millisecond

minimum = 0ms maximum = 3ms, average ± 2ms

Hybrid

Procedure:

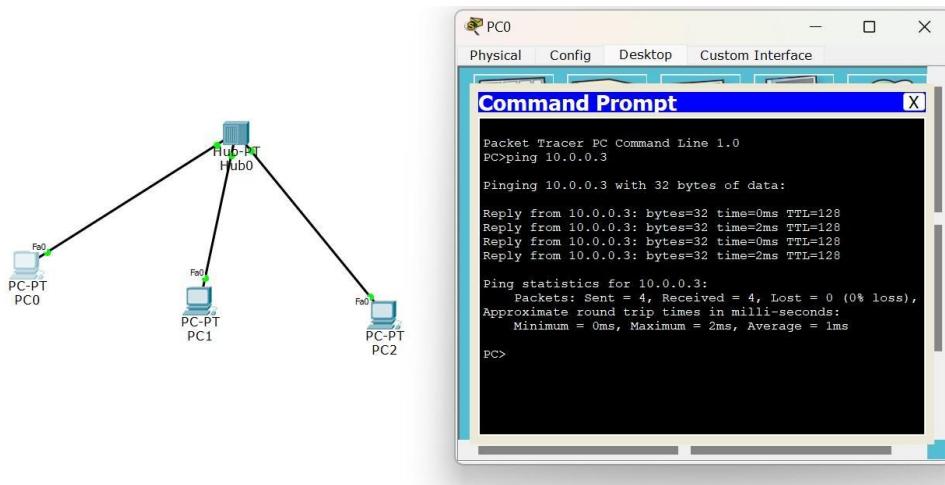
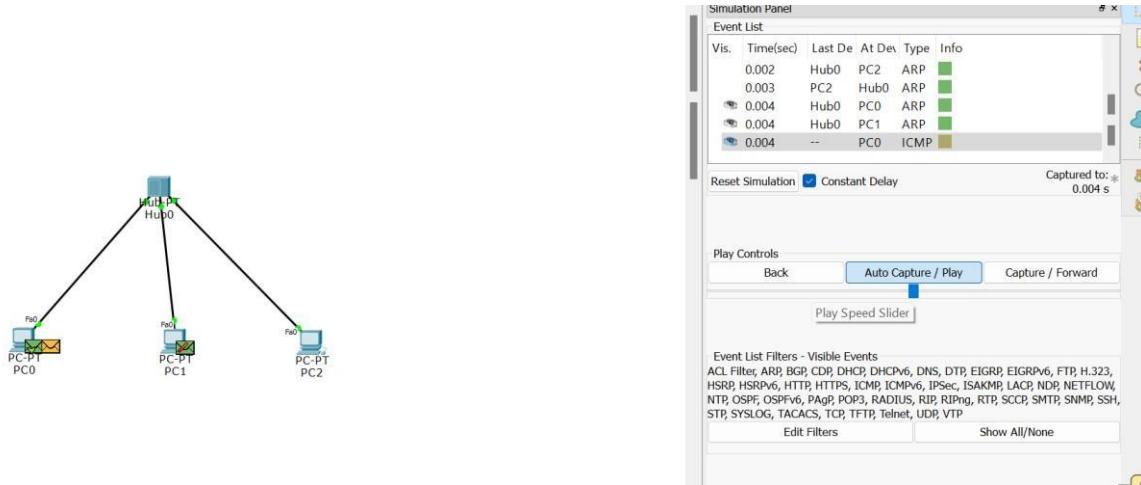
- 11 PC's - 2 generic hubs and 1 switch were placed on the worktop
- 4 PC's were connected to hub 0 via copper straight through cables. Remaining 7 pc's were connected to hub 1 via copper straight cable.
- All pc's were assigned address IP (10.0.0.1 to 10.0.0.11)
- Two hubs were connected to the switch via copper cross over cables which are used to connect devices on the same level.



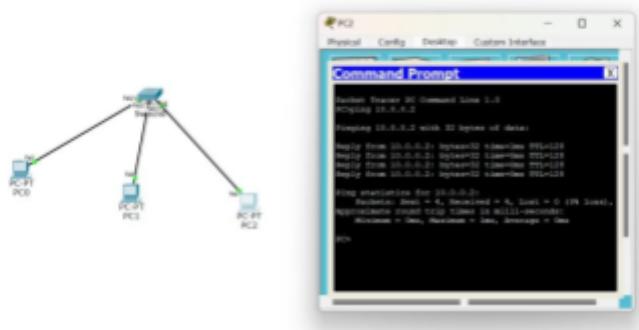
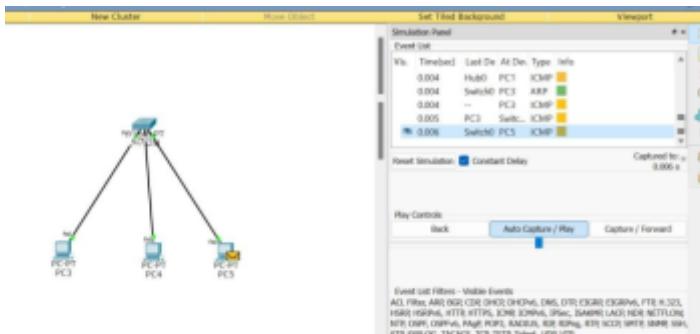
Scanned with CamScanner

TOPOLOGY & OUTPUT

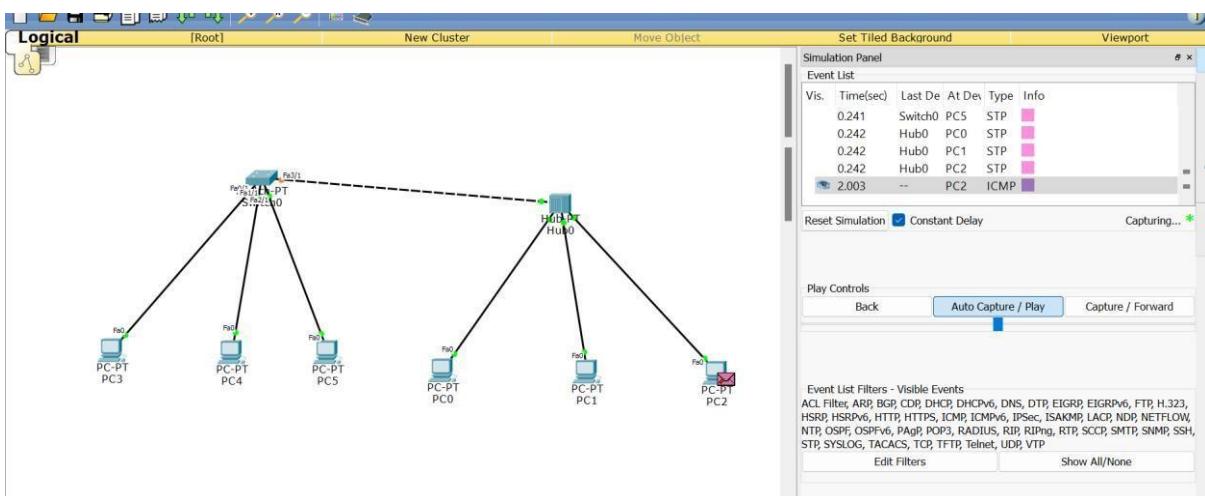
1. Hub and PCs

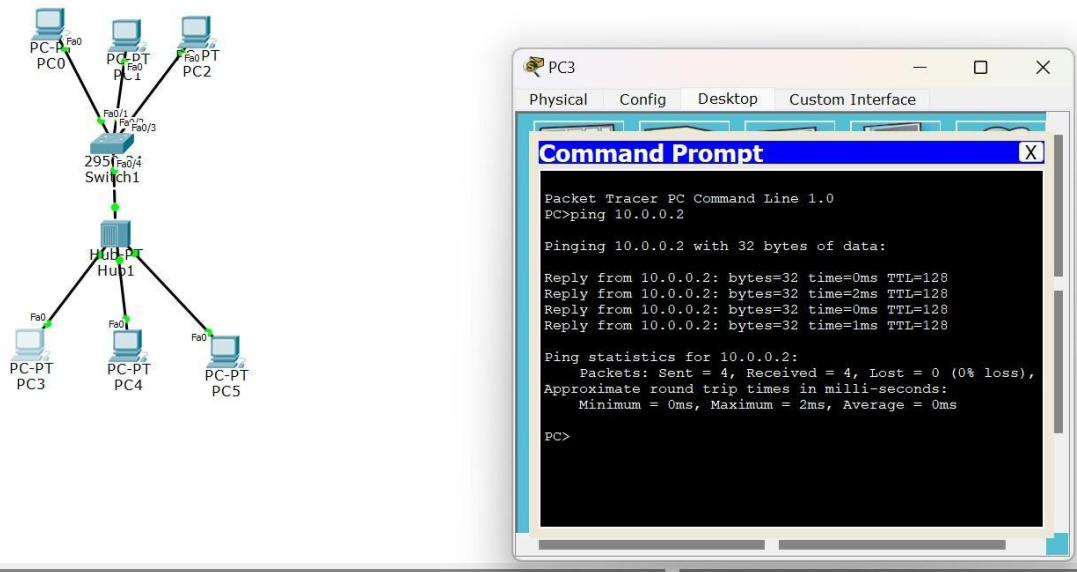


2. Switch and PCs



3. Hub, Switch and PCs





EXPERIMENT-2

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

PROGRAM 2.1

classmate
Date _____
Page _____

Experiment-2
Network connection using single router.

Aim:

Configuring IP addresses to router, explore ping responses, destination unreachable, request timed out and reply.

Topology:

Router (10.0.0.1)
PC-0 (10.0.0.1) --- Router --- PC-1 (20.0.0.1)

Procedure:

- Connect two end devices to a router through copper cross-over cable.
- Assign IP address to end devices.
- Configure gateways in router through CLI using following commands:
i) Enable
ii) config
iii) interface <port>
iv) ip address <ip> <subnet mask>
v) no shutdown
vi) exit
- Ping from one end node to another.

Observation:

Router is a device used to connect multiple networks.
Router is capable of transforming packets from one network.
End devices sends data packets to router. The destination IP address is noted by the router.

Scanned with CamScanner

For router OCLI

Router > enable

Router # config t

Router (config) # interface fast ethernet 0/0

Router (config-if) # ip address 20.0.0.10 255.0.0.0

Router (config-if) # no shutdown

exit

Router (config-if) # interface fast ethernet 1/0

Router (config-if) # ip address 20.0.0.11 255.0.0.0

Router (config-if) # no shutdown

exit

Result in case of switch being shut down

Pinging 20.0.0.11 with 32 bytes of data

Request timed out. loss at switch due to switch

Reply from 20.0.0.11 bytes=32 time=1ms TTL=127

Reply from 20.0.0.11 bytes=32 time=1ms TTL=127

Reply from 20.0.0.11 bytes=32 time=1ms TTL=127

Ping statistics from 20.0.0.11

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

Approximate round trip times in cumulative seconds.

Minimum = 0ms, maximum = 1ms, Avg = 0ms

Replies of 1ms loss 25% went well

Afterwards

selected algorithm turned off because switch is acting

otherwise it would always forward to all ports in switch

and hence at gateway does change switch ports

String will go down or number of ports will decrease

PROGRAM 2.2

classmate
Date _____
Page _____

Network connection with multiple Router.

Aim:
Configuring IP address of multiple routers, exploring ping responses, destination unreachable, request timed out and reply.

Topology:

End device **End device**
10.0.0.1 10.0.0.1

Procedure:

- Add two end devices and three routers to work space.
- Connect routers through serial DTE cable and end device to route through copper (cross-over) cable.
- Assign IP address to end devices and gateways.
- Configure gateway through CLI using following command
 - (i) enable mode at (config) level
 - (ii) config terminal (config) level
 - (iii) interface <port>
 - (iv) ip address <ip address> <subnet mask>
 - (v) no shut.
- exit

Using command Ip route<destination ip><routing ip> set path for each router.

Ping from one end device to another.

Scanned with CamScanner

Result: Max time taken was 12ms
 Ping 40.0.0.1 : with length 32 byte of data.
 Request time out from 40.0.0.1 byte = 32 time = 12ms TTL = 12
 Reply from 40.0.0.1 byte = 32 time = 12ms TTL = 12
 Reply from 40.0.0.1 byte = 32 time = 12ms TTL = 12
 Reply from 40.0.0.1 byte = 32 time = 12ms TTL = 12

Ping statistic from 40.0.0.1:
 Packet Sent = 4 received = 4, received = 3,
 Lost = 1 (25% loss)
 Approximate round trip in milliseconds
 Minimum = 12ms maximum = 14ms Avg = 12ms.

Configure the router by opening CC7
 Router 0:
 Router > enable
 Router > config terminal
 Router (config) # interface fastEthernet 0/0
 Router (config-if) # ip address 40.0.0.10 255.0.0.0
 Router (config-if) no shutdown
 Router (config) # interface serial 2/0
 Router (config-if) # ip address 20.0.0.10 255.0.0.0
 Router (config-if) # no shutdown
 exit
 Router > write memory
 exit
 Router > enable
 Router (config) # interface serial 2/0
 Router (config-if) # ip address 20.0.0.10 255.0.0.0
 Router (config-if) # no shutdown
 Router (config) # interface fastEthernet 0/0
 Router (config-if) # ip address 40.0.0.10 255.0.0.0
 Router (config-if) # no shutdown

Scanned with CamScanner

Router 1:

```
router >enable
Router # config t
Router (config) # interface serial 2/0
Router (config-if) # ip address 20.0.0.20 255.0.0.0
Router (config-if) # no shut
```

```
exit
```

```
Router (config) # interface serial 3/0
Router (config-if) # ip address 30.0.0.20 255.0.0.0
Router (config-if) # no shutdown
```

```
exit
```

```
Router (config) # interface serial 3/0
Router (config-if) # ip address 30.0.0.20 255.0.0.0
Router (config-if) # no shutdown
```

```
exit
```

```
Router (config) # exit
Router (config) # exit
```

```
Router 2: configuration is recommended to avoid inconsistency
```

```
Router >enable
```

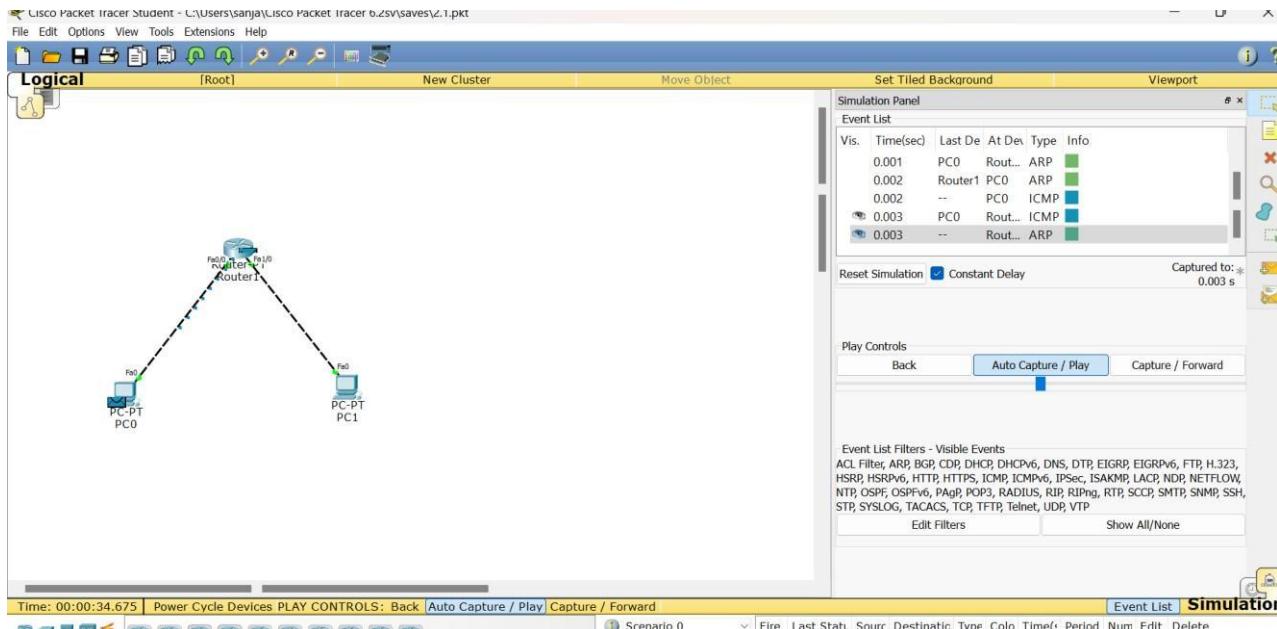
```
Router # config t
Router (config) # interface serial 2/0
Router (config-if) # ip address 30.0.0.20 255.0.0.0
Router (config-if) # no shutdown
exit
```

```
Router (config-if) # interface fast ethernet 0/10
```

```
Router (config-if) # ip address 40.0.0.10 255.0.0.0
Router (config-if) # no shutdown
exit
```

TOPOLOGY & OUTPUT

PROGRAM 2.1



Command Prompt

```
Packet Tracer PC Command Line 1.0
PC>PING 20.0.0.1

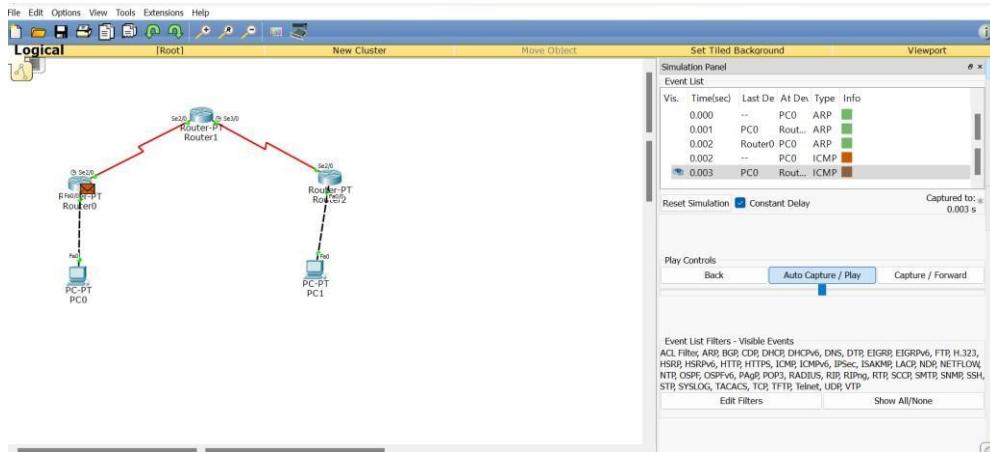
Pinging 20.0.0.1 with 32 bytes of data:

Reply from 20.0.0.1: bytes=32 time=0ms TTL=127
Reply from 20.0.0.1: bytes=32 time=0ms TTL=127
Reply from 20.0.0.1: bytes=32 time=0ms TTL=127
Reply from 20.0.0.1: bytes=32 time=2ms TTL=127

Ping statistics for 20.0.0.1:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 0ms, Maximum = 2ms, Average = 0ms

PC>
```

PROGRAM 2.2



```
Packet Tracer PC Command Line 1.0
PC>40.0.0.1
Invalid Command.

PC>PING 40.0.0.1
Pinging 40.0.0.1 with 32 bytes of data:
Reply from 40.0.0.1: bytes=32 time=11ms TTL=125
Reply from 40.0.0.1: bytes=32 time=6ms TTL=125
Reply from 40.0.0.1: bytes=32 time=8ms TTL=125
Reply from 40.0.0.1: bytes=32 time=2ms TTL=125

Ping statistics for 40.0.0.1:
  Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 2ms, Maximum = 11ms, Average = 6ms

PC>
```

EXPERIMENT-3

Q)Configure default route, static route to the Router

Experiment-3

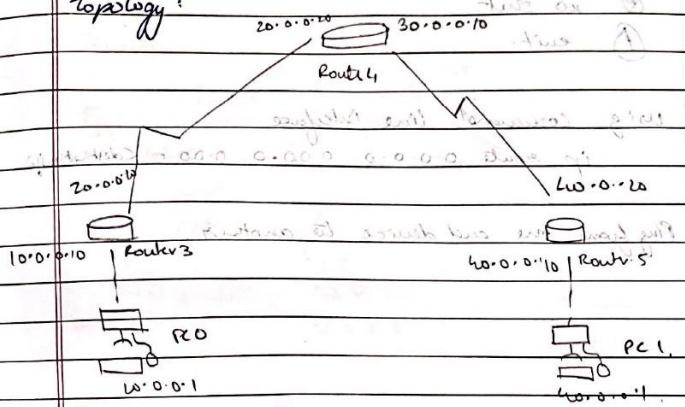
classmate
Date _____
Page _____

Configuration of default route

Aim:

Configure default route and static route to the route

Topology:



Procedure:

Add two end devices and three routers to workspace.

Connect routers through serial DTE cable and end devices through copper crossover cable. Assign IP addresses to the end devices.

End device 1 : 10.0.0.1

End device 2 : 40.0.0.1

Observation..

- Routers are seen to connect two different networks
- If router has only one port to transverse.
- Default routing to send packet of any destination.
- End devices sends the packet to the router which then redirect it to the appropriate destination.

SUBNET MASKS

Configuration through CLI command

(1) enable.

(2) config t

(3) interface <port> <switch-threshold> <mask>

(4) ip address <ip address> <subnet mask>

(5) no shut

(6) exit.

using command line interface -

ip route 0.0.0.0 0.0.0.0 0.0.0.0 <destination ip>

Ping from one end device to another

that will have much less cost

: address

so less TTL longer segment without turning

so less overhead more network

network has set of numbers

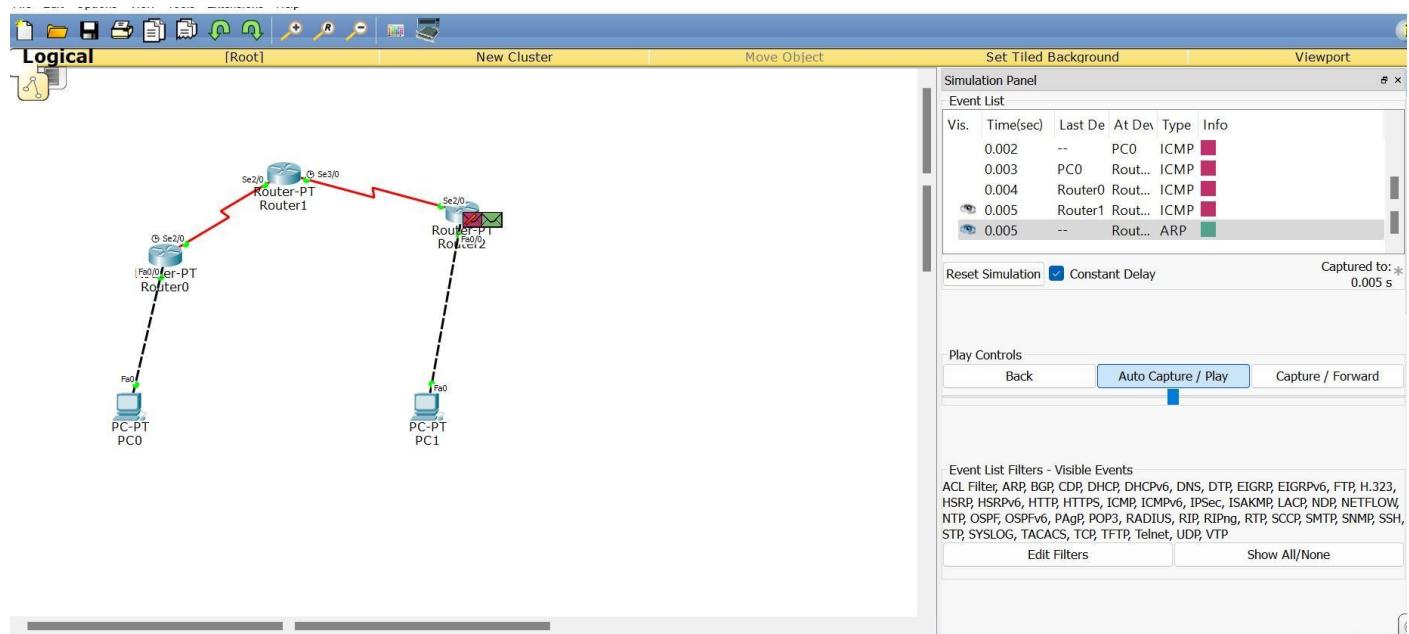
1.0.0.0 : 1 round trip

1.0.0.0.5 : 5 round trip

: address

so less overhead

TOPOLOGY & OUTPUT



Command Prompt

```
Packet Tracer PC Command Line 1.0
PC>ping 40.0.0.1

Pinging 40.0.0.1 with 32 bytes of data:

Reply from 40.0.0.1: bytes=32 time=12ms TTL=125
Reply from 40.0.0.1: bytes=32 time=13ms TTL=125
Reply from 40.0.0.1: bytes=32 time=7ms TTL=125
Reply from 40.0.0.1: bytes=32 time=8ms TTL=125

Ping statistics for 40.0.0.1:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 7ms, Maximum = 13ms, Average = 10ms

PC>|
```

EXPERIMENT-4

Q) Configure DHCP within a LAN and outside LAN

PROGRAM 4.1

Date _____
Page _____

Experiment-4 -> 3rd year
Configure DHCP within a LAN & outside LAN

D) Within a LAN

Topology:

Procedure:

- Create a LAN network (10.0.0.0) by selecting 3 ports as a server and connect them to a switch.
- Set the server IP address to 10.0.0.1 & set the default gateway to 10.0.0.20.
- Set the server to DHCP mode (Router mode).
- Put down the gateway & start IP address (10.0.0.2).

Result:

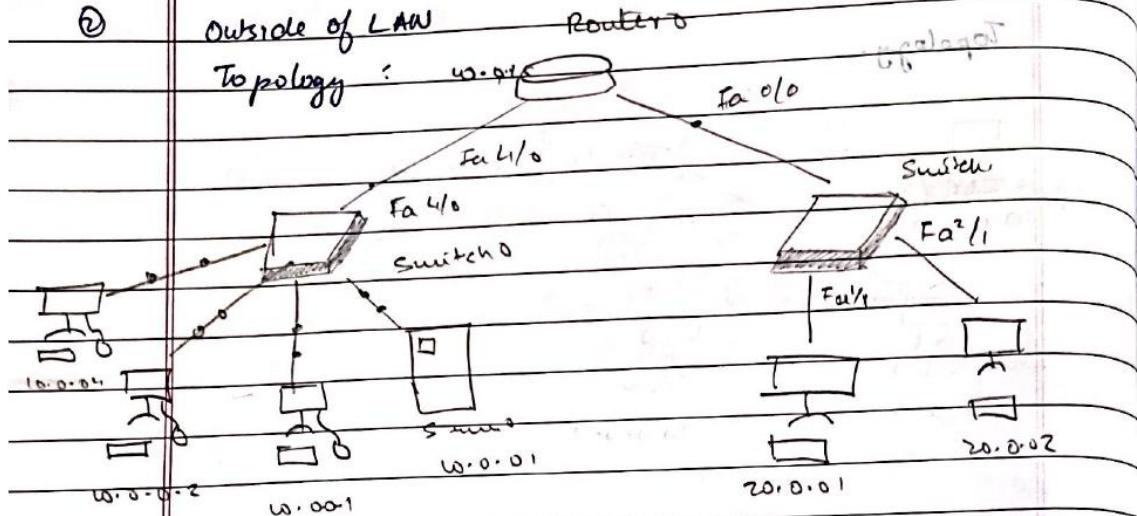
```
root> ping 10.0.0.4
Pinging 10.0.0.4 with 32 byte of data.
Reply from 10.0.0.4 bytes=32 time=1ms TTL=128
Gathering 10.0.0.0 results
    1 loss 0% loss
```

Scanned with CamScanner

Ping statistic for 10.0.0.4
 Packet sent = 4, Received = 4, Lost = 0 (0% loss)
 minimum = 0ms, maximum 1ms, Avg = 0ms

②

Outside of LAN



Procedure:

→ Follow the same steps as in case of creating a

LAN, by creating 10.0.0.0/24 network with its

IP address - 10.0.0.1 and the gateway

- 10.0.0.0/10. Then do the same for the

20.0.0.0/10 network with its IP address

Create network with 2 PCs and a switch and

connect the 2 networks using a Router.

→ enable

→ config t

→ interface 4/0/0

→ ip address 10.0.0.20 255.0.0.0

→ no shutdown

→ exit

→ interface fa 0/0/0

→ ip address 20.0.0.20 255.0.0.0

→ no shutdown

→ exit

PROGRAM 4.2

Go to the sources of service 0 and set another
DHCPO pool gateway (default) to 20.0.0.10.

The following are the 2 pools

Pool Name	Default gateway	DNS Server	Subnet mask
Service pool	10.0.0.10	0.0.0.0	10.0.0.2 255.0.0.0
Service pool	20.0.0.10	0.0.0.0	20.0.0.1 255.0.0.0

> config t

> interface fa0/0

> ip helper address (server ip-address)

> no shut

exit

Result

PC > ping 20.0.0.2.

Pinging 20.0.0.2 with 32 byte of data

Request timed out

Reply from 20.0.0.2: bytes=32 time=0ms TTL=127

Reply from 20.0.0.2: bytes=32 time=2ms TTL=127

Reply from 20.0.0.2: bytes=32 time=1ms TTL=127

Ping statistics for 20.0.0.2.

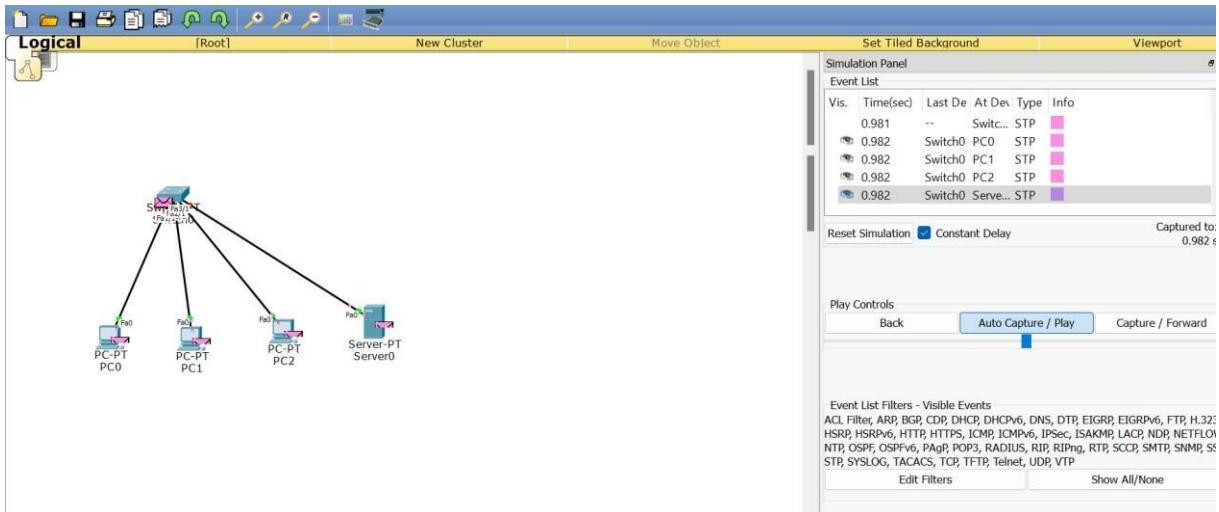
Packets: sent=4 received=3 lost=1 (25% loss)

Avg. time (in millisecond)

Minimum=0ms, maximum=4ms, Average= 1ms

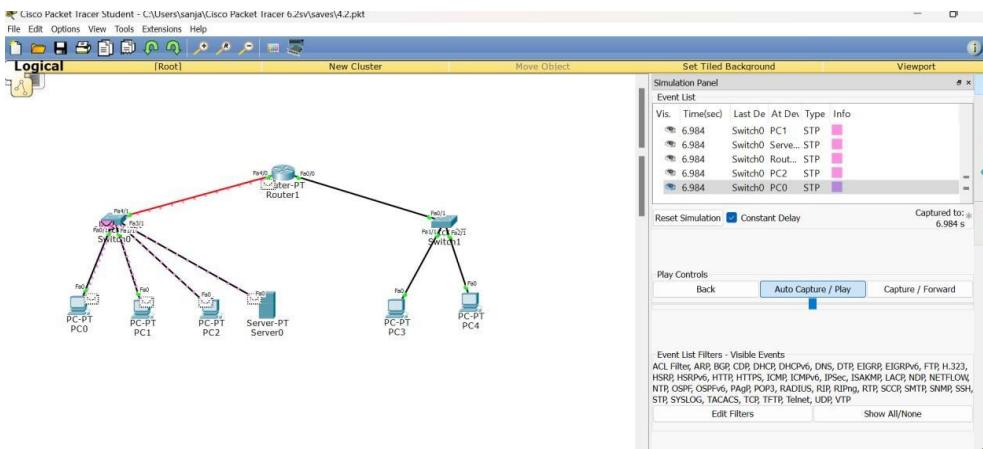
TOPOLOGY & OUTPUT

PROGRAM 4.1



```
Packets: Sent = 1, Received = 0, Lost = 1 (100% loss),  
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=1ms 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 4.2



Command Prompt

```
Packet Tracer PC Command Line 1.0
PC>ping 20.0.0.2

Pinging 20.0.0.2 with 32 bytes of data:

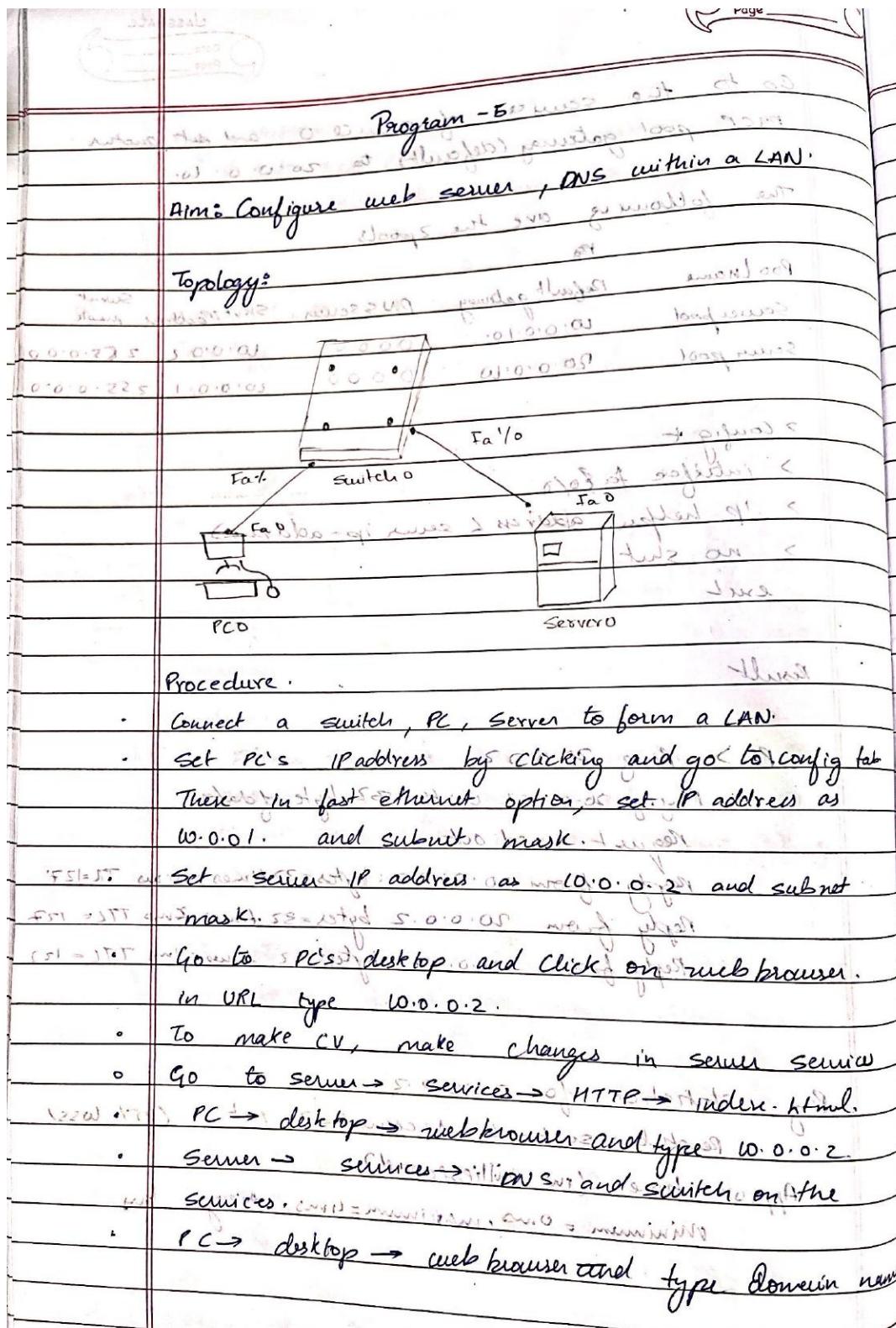
Request timed out.
Reply from 20.0.0.2: bytes=32 time=0ms TTL=127
Reply from 20.0.0.2: bytes=32 time=0ms TTL=127
Reply from 20.0.0.2: bytes=32 time=0ms TTL=127

Ping statistics for 20.0.0.2:
    Packets: Sent = 4, Received = 3, Lost = 1 (25% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 0ms, Average = 0ms

PC>
```

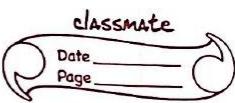
EXPERIMENT-5

Q) Configure Web Server, DNS within a LAN



5

etc
etc



output:

web browser.

[<] URL http://adnan [Go] [Stop]

CW

Adnan.

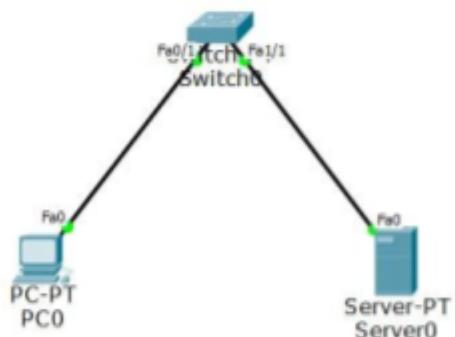
USIN: IBM21CS008

Languages: C/C#/Java

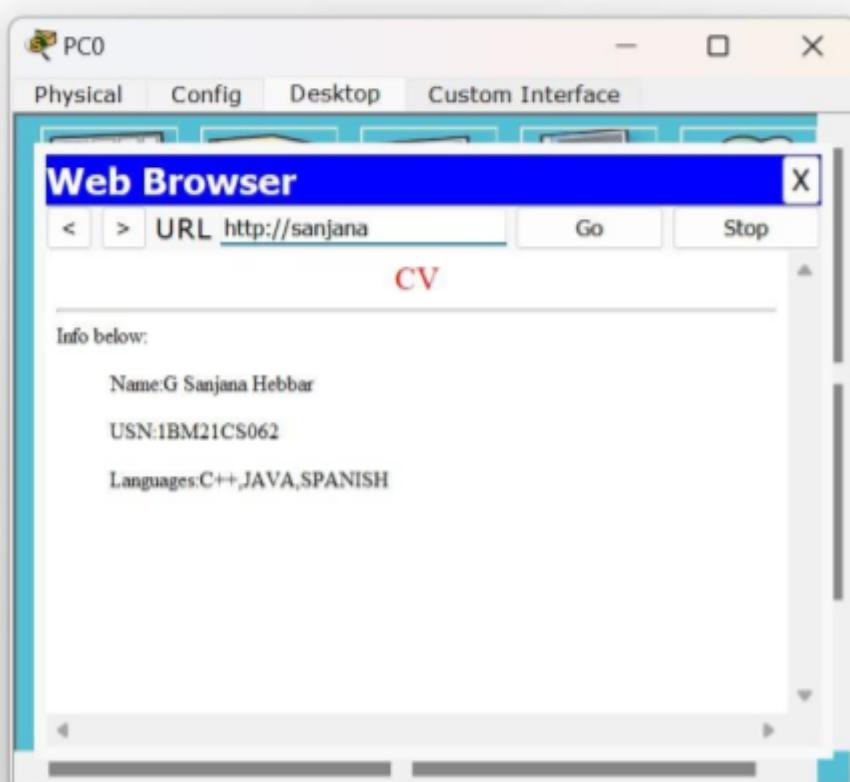
Observation:

- When you want to go to a certain website, you open web browser & type domain name of that website or you can also type IP address.
- Since the IP address cannot be remembered, we resolve it to find the domain and complete communication.

TOPOLOGY & OUTPUT

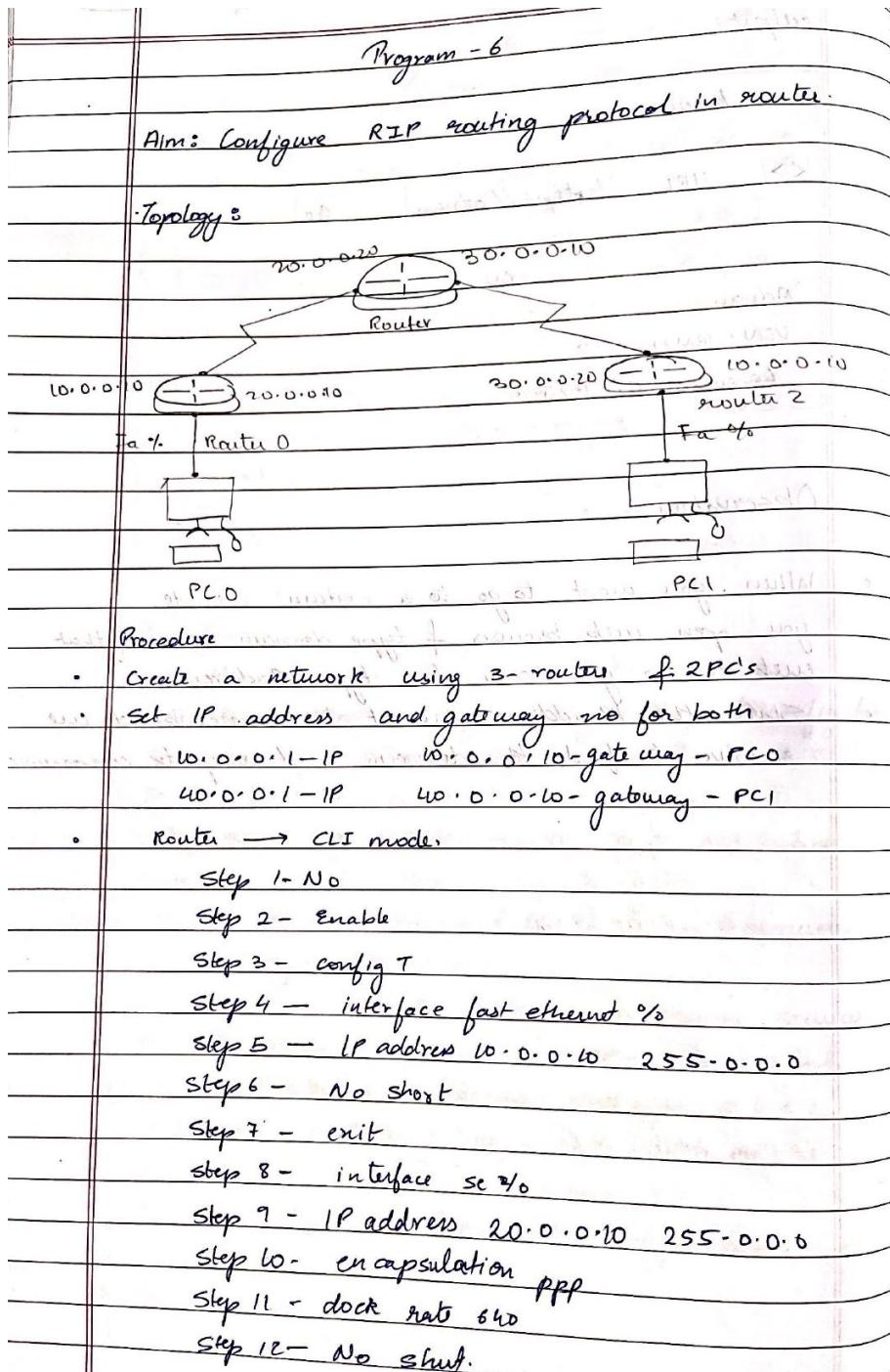


New Cluster Move Object



EXPERIMENT-6

Q) Configure RIP routing Protocol in Routers



Scanned with CamScanner

- here for routers with fast ethernet execute till step 9.
and type no shut.
 - Only for router to router connection execute all steps
also execute step 11.
 - again to Router 0 \rightarrow CLI mode :
- Step 1: config t
Step 2: router rip
Step 3: Network 10.0.0.0
Step 4: Network 20.0.0.0
Step 5: exit
- Repeat these steps for all routers
 - Now go to each router and type show ip route
 - Go to PC0 and Ping a msg to PC1 using
ping destination IP address command.

Ping output:

Packet tracer PC command line 1.0

PC > Ping 40.0.0.1.

Pinging 40.0.0.1 with 32 byte data.

Request time out

Reply from 40.0.0.1 : bytes = 32 Time = 8ms TTL = 125

Reply from 40.0.0.1 : bytes = 32 Time = 5ms TTL = 125

Reply from 40.0.0.1 : bytes = 32 Time = 10ms TTL = 125

Statistics

Packets sent = 4 ; received = 3, lost = 1 (25% loss)

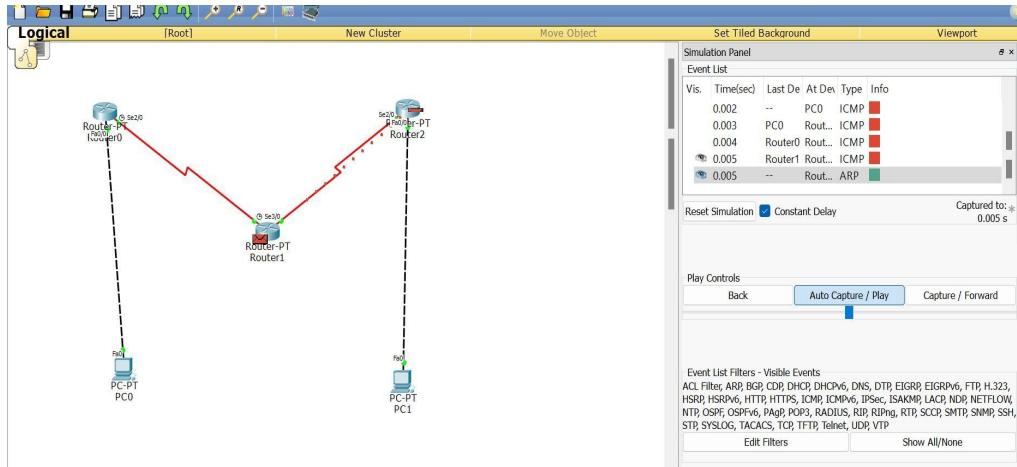
Approx round trip in ms:

min = 5ms, max = 10ms, Avg = 7ms.

Observation

- RIP is used to keep track of a routing matrix to find best path. It is a distance - vector routing
- updates of the network are exchanged periodically
- updates of routing information are broadcasted
- full routing information are broadcasted in updates
- router always trust routing information received from neighbour routers

TOPOLOGY & OUTPUT



Command Prompt

```
Packet Tracer PC Command Line 1.0
PC>ping 40.0.0.1

Pinging 40.0.0.1 with 32 bytes of data:

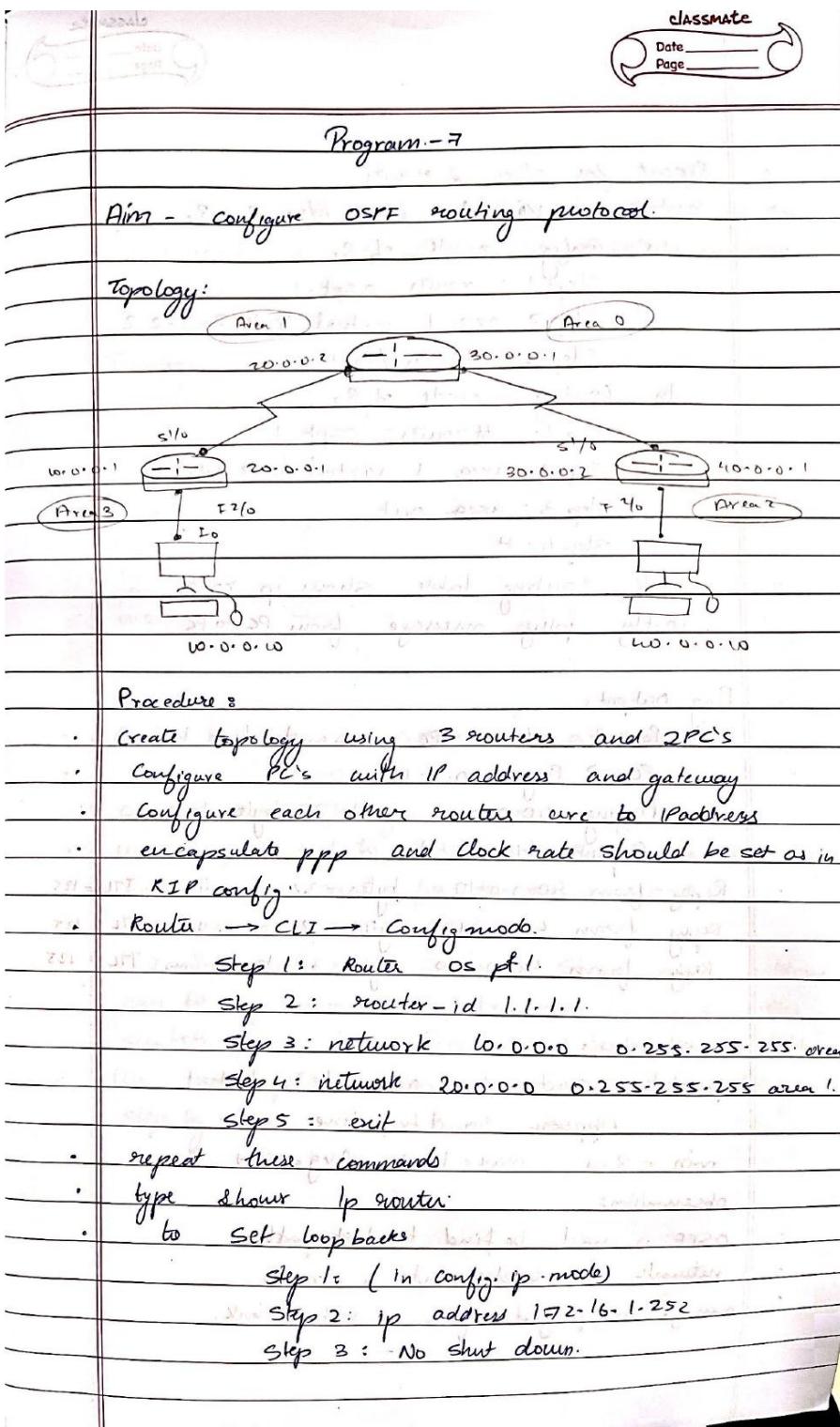
Reply from 40.0.0.1: bytes=32 time=13ms TTL=125
Reply from 40.0.0.1: bytes=32 time=2ms TTL=125
Reply from 40.0.0.1: bytes=32 time=9ms TTL=125
Reply from 40.0.0.1: bytes=32 time=12ms TTL=125

Ping statistics for 40.0.0.1:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 2ms, Maximum = 13ms, Average = 9ms

PC>
```

EXPERIMENT-7

Q) Configure OSPF routing protocol



Scanned with CamScanner

- Repeat for other 2 routers
- Create a virtual link b/w R₁, R₂
- In config mode of R₁
 - Step 1: router ospf 1
 - Step 2: area 1 virtual link 2.2.2.2
 - Step 3: # enter / exit
- In config mode of R₂
 - Step 1: #router ospf 1
 - Step 2: area 1 virtual link 1.1.1.1
 - Step 3: area exit
 - Step 4: #
- Check routing table, show ip route
- Lastly ping message from PC to PC.

Ping output:

Packet tracer PC command - line 1.0

PC > Ping 40.0.0.10

Pinging 40.0.0.10 with 32 byte data:

Request timed out.

Reply from 40.0.0.10 bytes = 32 time = 11ms TTL = 125

Reply from 40.0.0.10 bytes = 32 time = 11ms TTL = 125

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

Statistics:

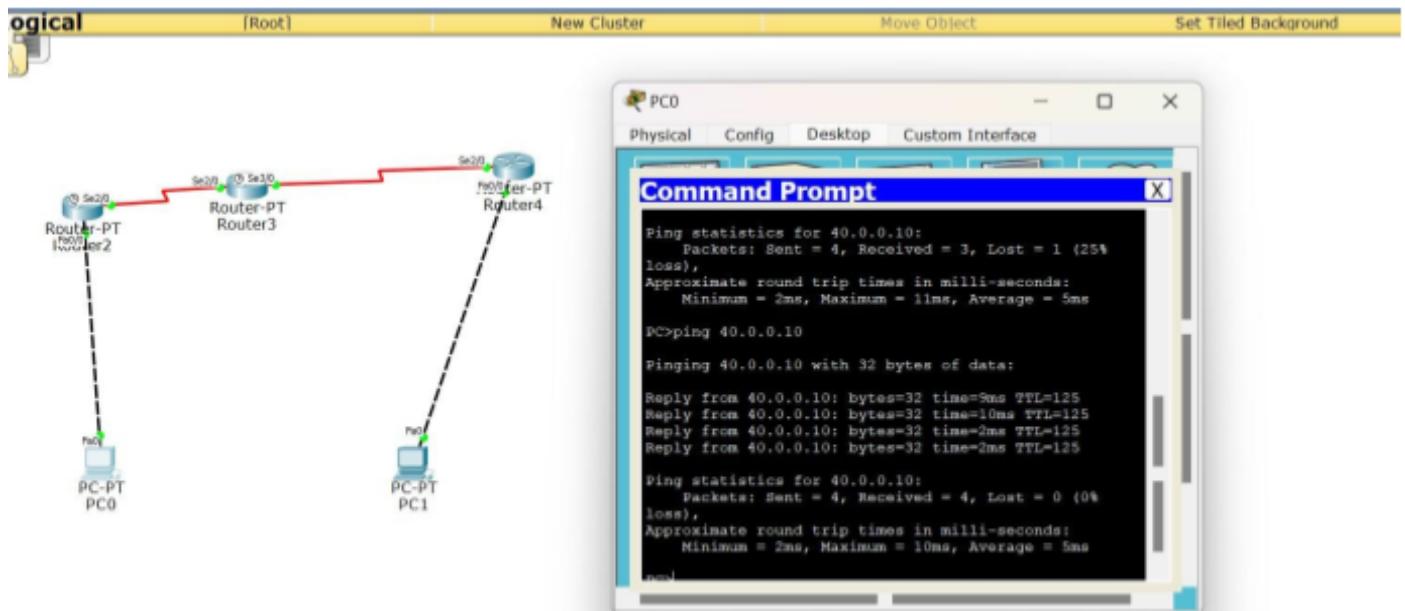
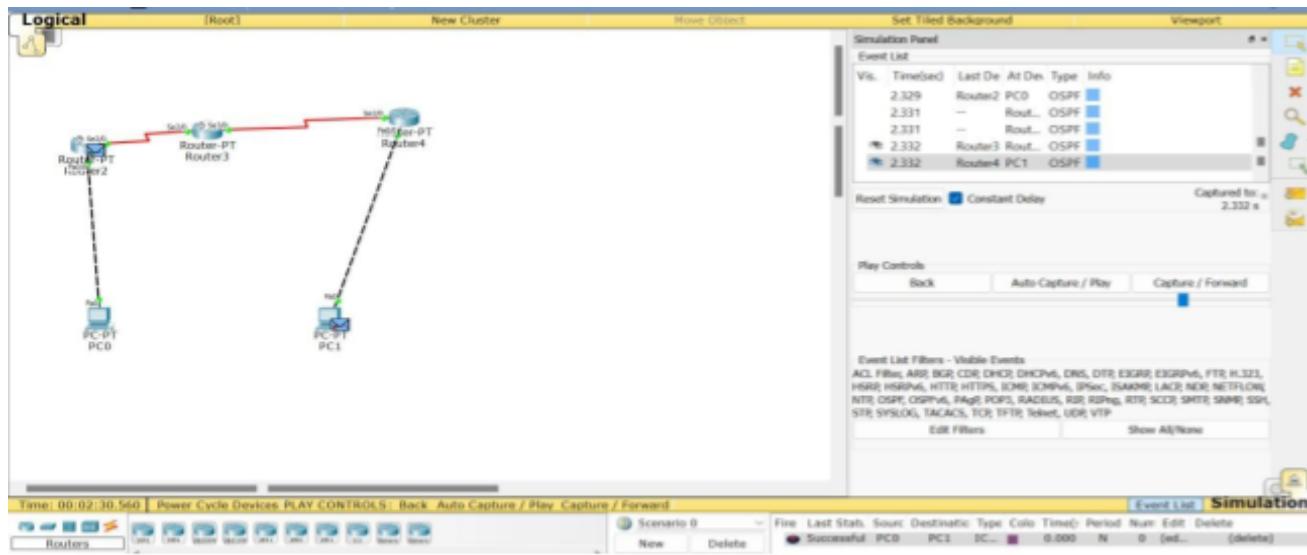
packets sent = 4, received = 3, lost = 1

Avg round trip time
min = 8ms, max = 11ms, Avg = 10ms

Observation:

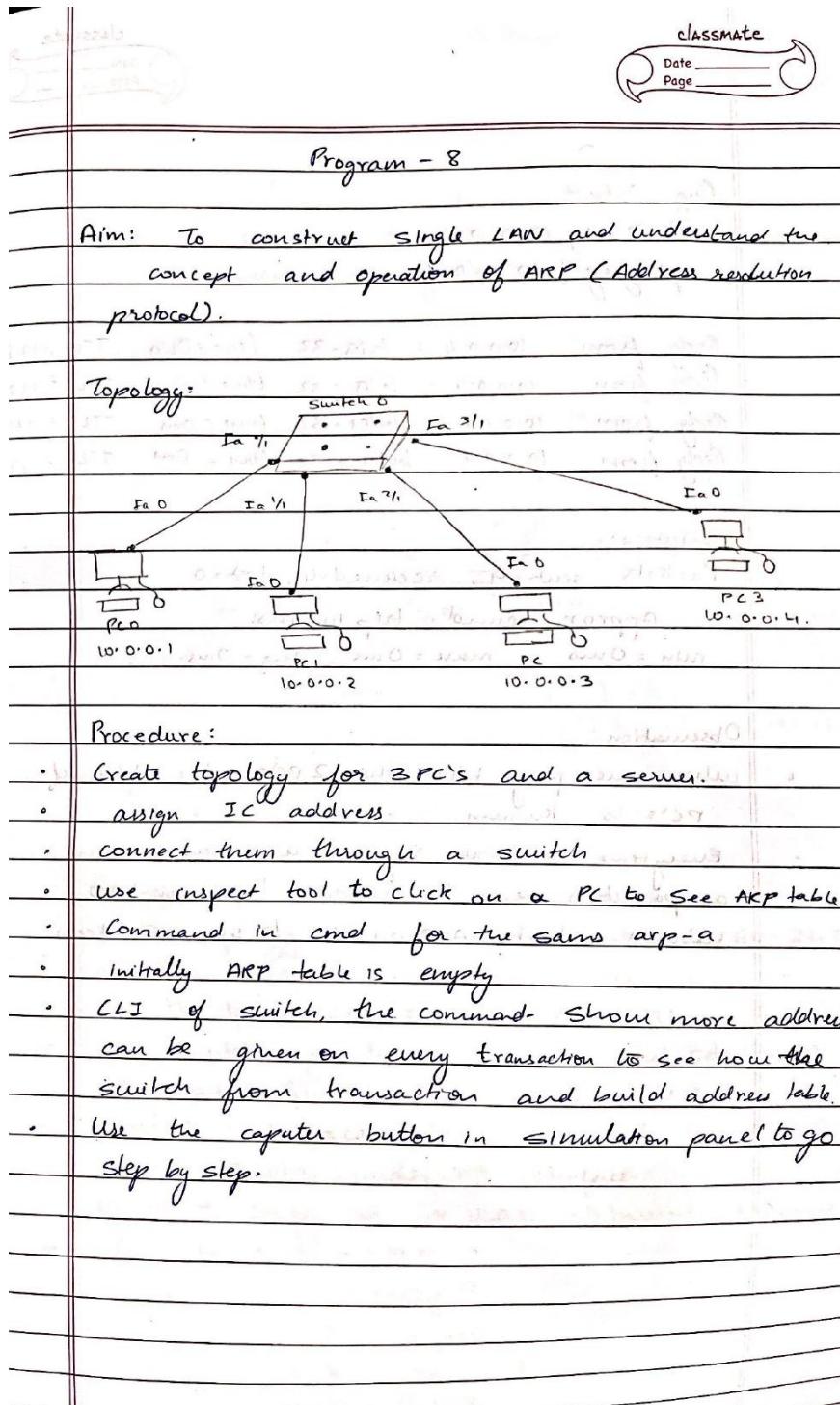
- OSPF is used to find the best path
- network is divided into 4 areas
- message is pinged after a virtual link.

TOPOLOGY & OUTPUT



EXPERIMENT-8

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



Ping Output:

PC > ping 10.0.0.4

pinging 10.0.0.4

Reply from 10.0.0.4 : bytes = 32 time = 0ms TTL = 12
Reply from 10.0.0.4 : bytes = 32 time = 0ms TTL = 12
Reply from 10.0.0.4 : bytes = 32 time = 0ms TTL = 12
Reply from 10.0.0.4 : bytes = 32 time = 0ms TTL = 12

Statistics:

Packets sent = 4, received = 4, lost = 0

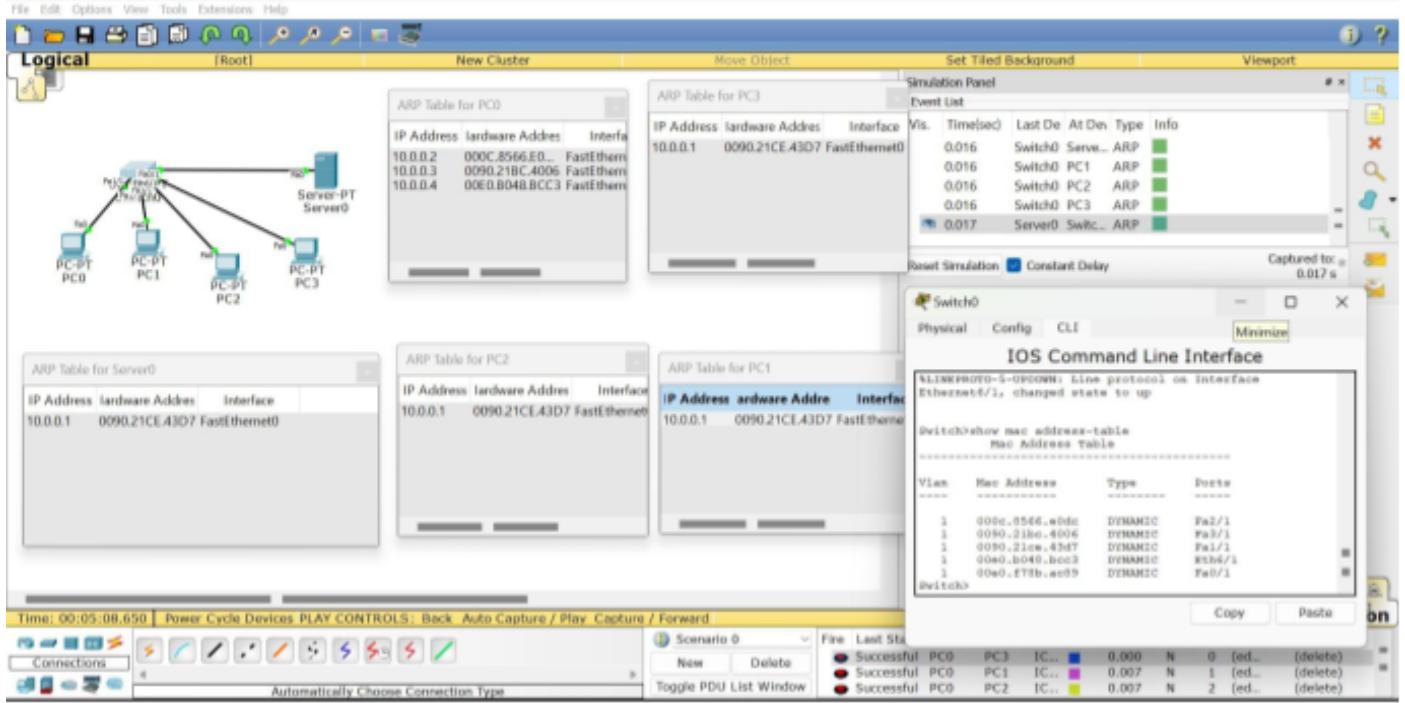
Avg approx round trip in ms:

min = 0ms, max = 0ms, Avg = 0ms.

Observation:

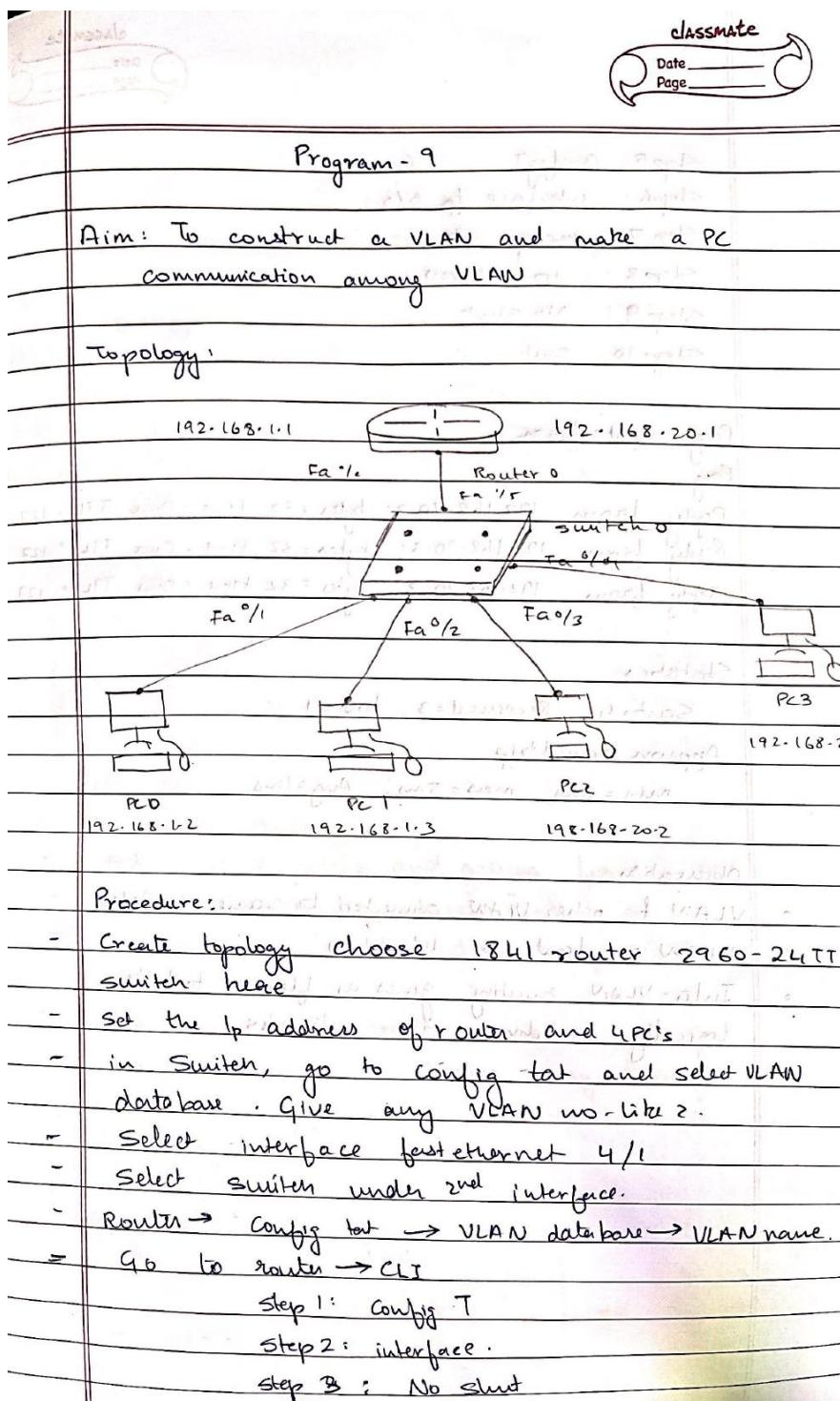
- when we ping b/w other 2 PCs, the address of PCs is known.
- Everytime a host request a mac address, a packet is sent to another host in LAN, which checks ARP cache to see IP address

TOPOLOGY & OUTPUT



EXPERIMENT-9

Q) To construct a VLAN and make the PC's communicate among a VLAN



Scanned with CamScanner

Step 5: config T

Step 6: interface fa 0/0-1

Step 7: encapsulate

Step 8: ip address

Step 9: No shut

Step 10: exit

Ping output = from

Ping

Reply from 192.168.20.3: bytes = 32 time = 0ms TTL = 127

Reply from 192.168.20.3: bytes = 32 time = 0ms TTL = 127

Reply from 192.168.20.3: bytes = 32 time = 0ms TTL = 127

Statistics:

Sent = 4, Received = 3, Lost = 1

Approx round trip

min = 0ms, max = 5ms, Avg = 1ms.

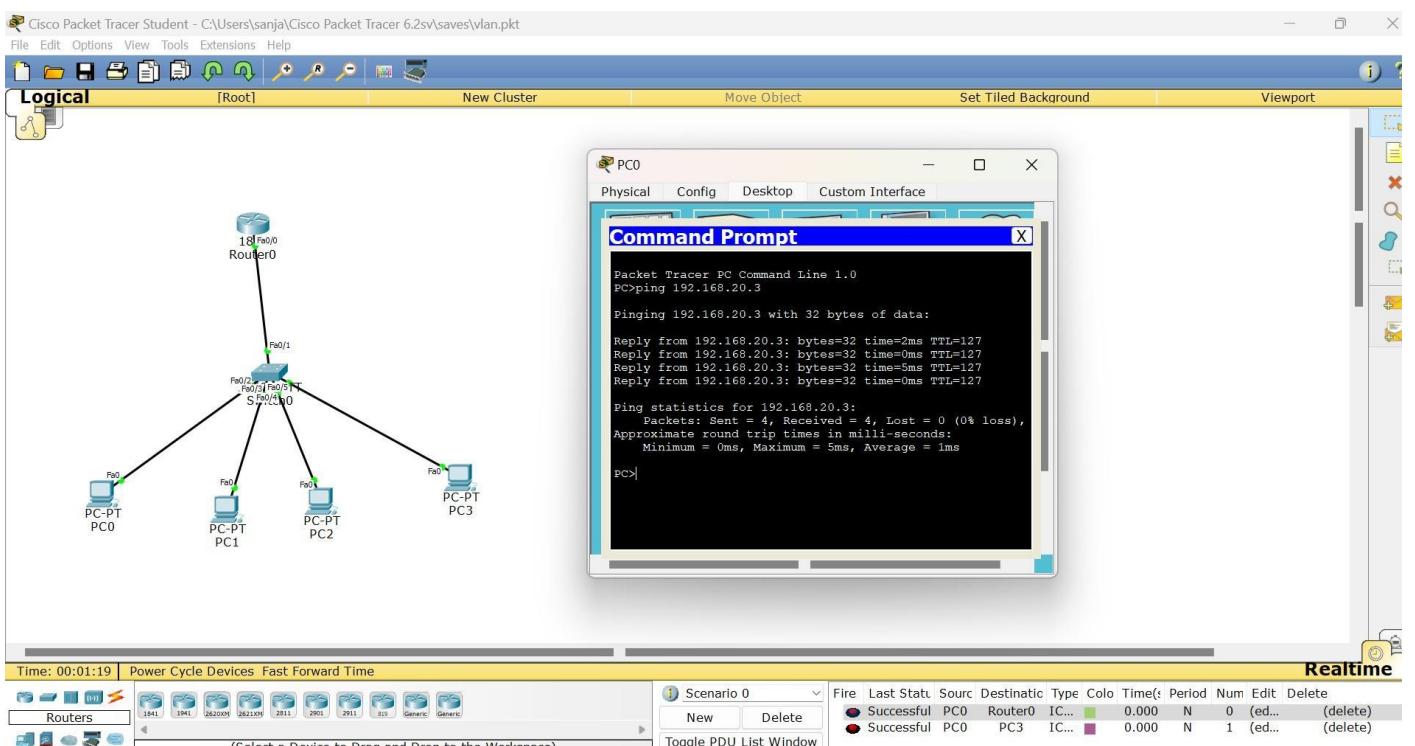
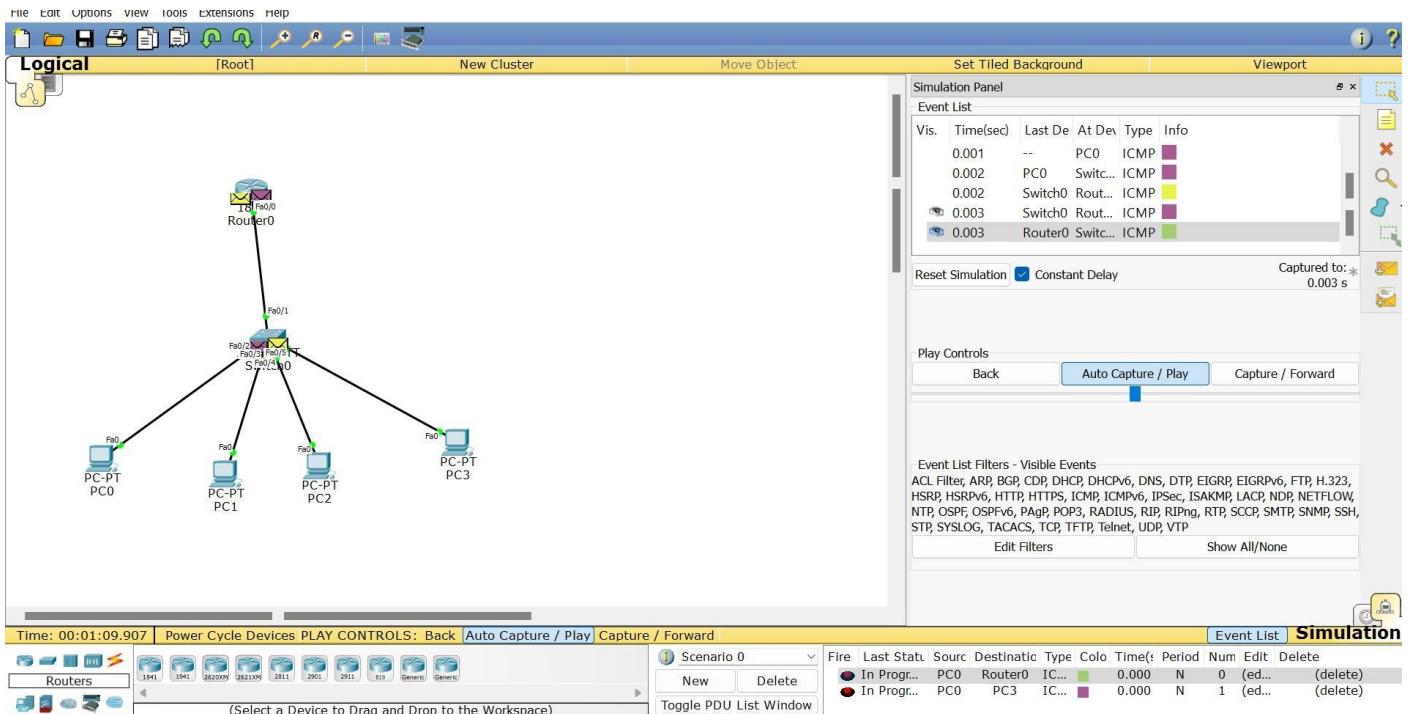
Observation

VLAN & other VLAN connected to same switch.

VLAN's don't use IP address

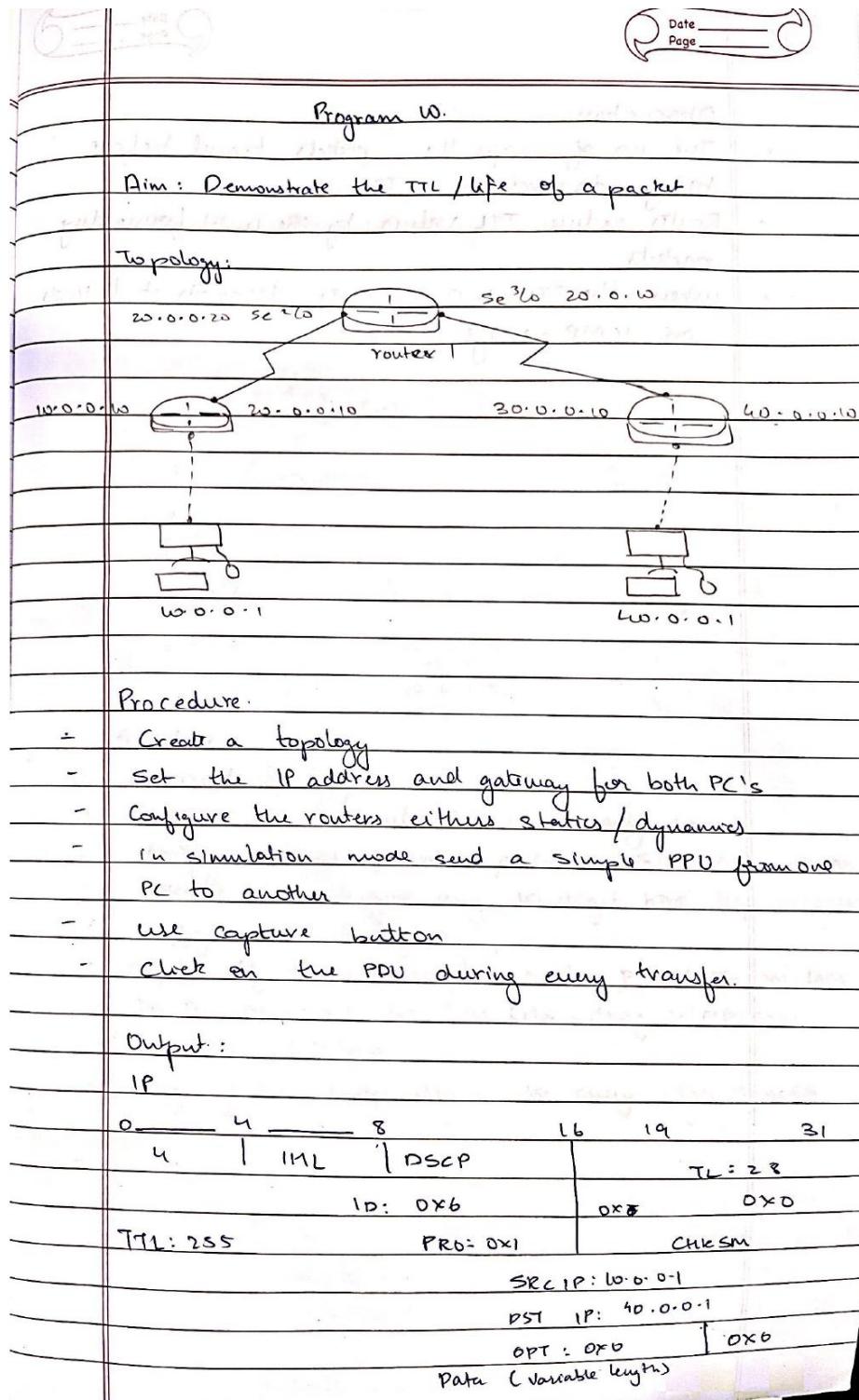
Inter-VLAN routing gives a flexible tool to logically subdivide their networks.

TOPOLOGY & OUTPUT



EXPERIMENT-10

Q) Demonstrate the TTL / Life of a Packet

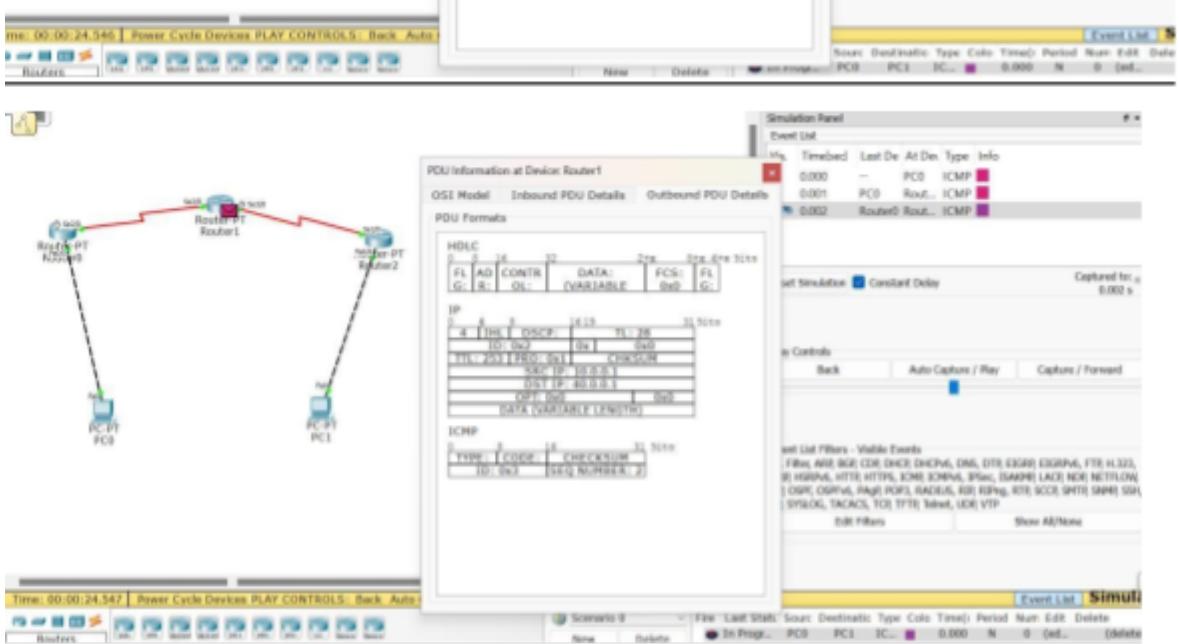
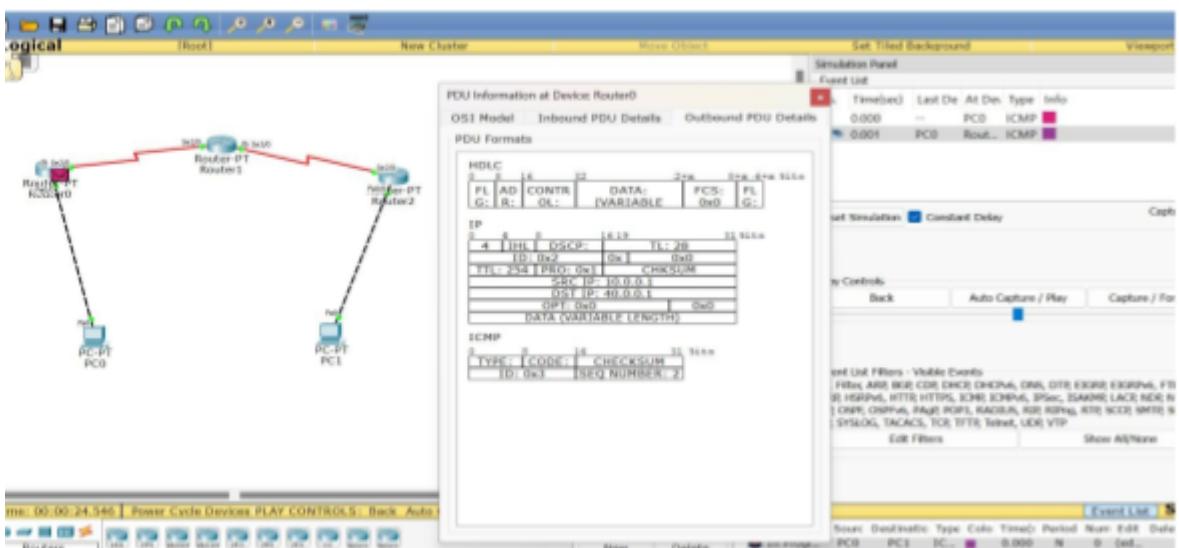
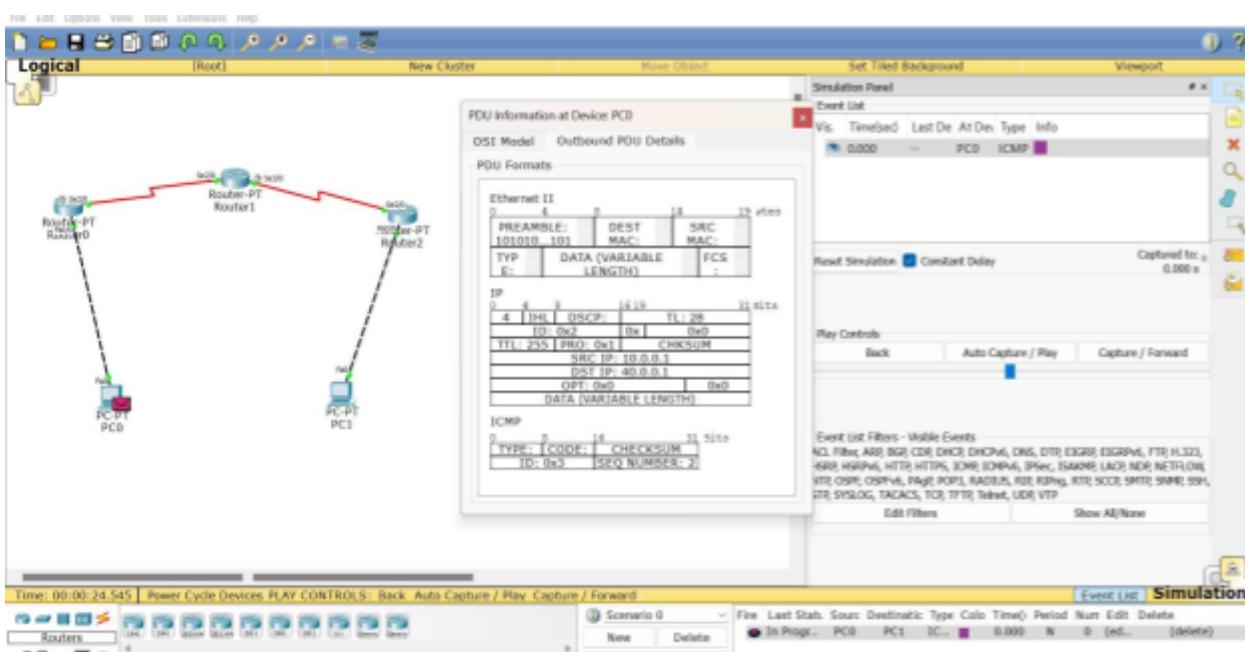


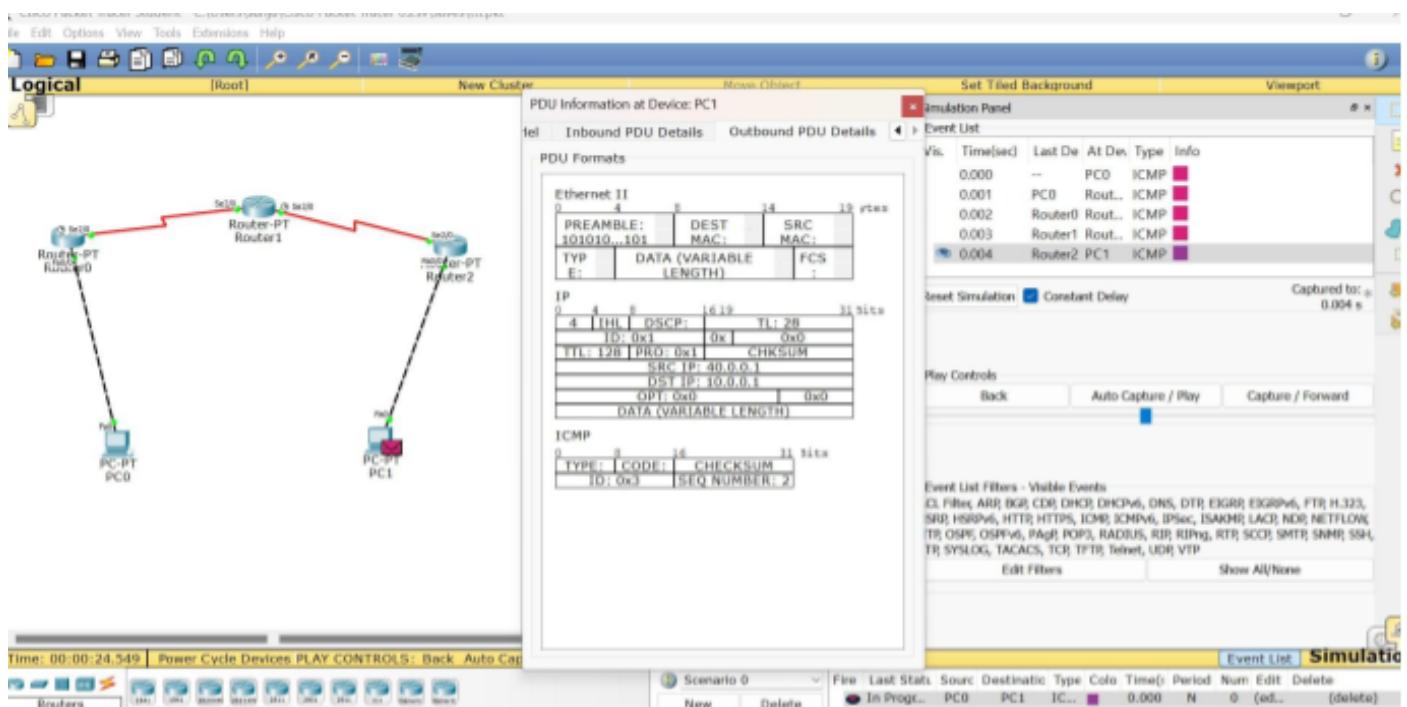
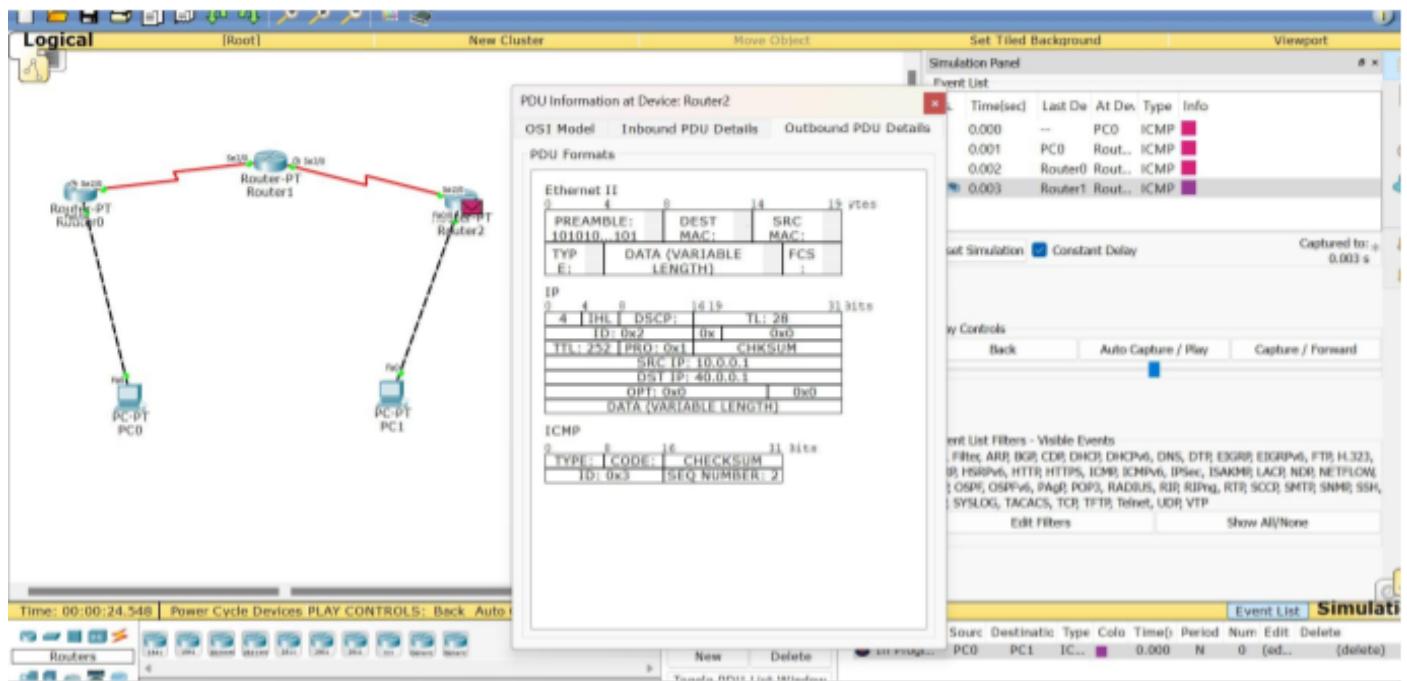
Scanned with CamScanner

Observation:

- The no of hops the packets travel before being discarded is as TTL
- Router reduce TTL values by one until forwards packets
- when the TTL is 0, router discards it & an ICMP message.

TOPOLOGY & OUTPUT





EXPERIMENT-11

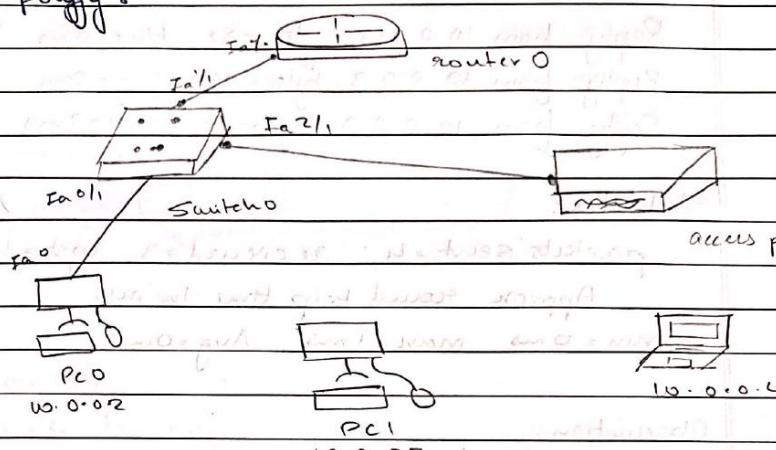
Q) To construct a WLAN and make the nodes communicate wirelessly

Date _____
Page _____

Program-11

Aim: To construct a WLAN and make nodes communication wirelessly.

Topology:



Procedure:

- Construct topology
- Configure Po 0 of router 0 as normally done
- Configure access point 1 - port 1 → SSID Name - select LOEP & give any 4 digit hex key - 01
- Configure PC1 & laptop
- Switch off device, drag the existing PT - Host - NM - to the component list in LMS - drag WMP300N wireless interface
- Ping from every device to every other device

Ping Output:

Packet tracer PC Command Line 1.0

PC > Ping 10.0.0.3

Pinging 10.0.0.3 with 32 bytes of data

Request time out;

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

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

Reply from 10.0.0.3: bytes = 32 time = 2ms TTL =

Statistics:

packets sent = 1, received = 3, lost = 1

Approx round trip time in ms:

min = 0ms, max = 1ms, Avg = 0ms

Observation:

- A WLAN is a group of connected devices
- Data sent in packets contain layers with 10
- The access point in the base station seen as a hub to which other stations are c

Received 1 packet in 1.01 seconds

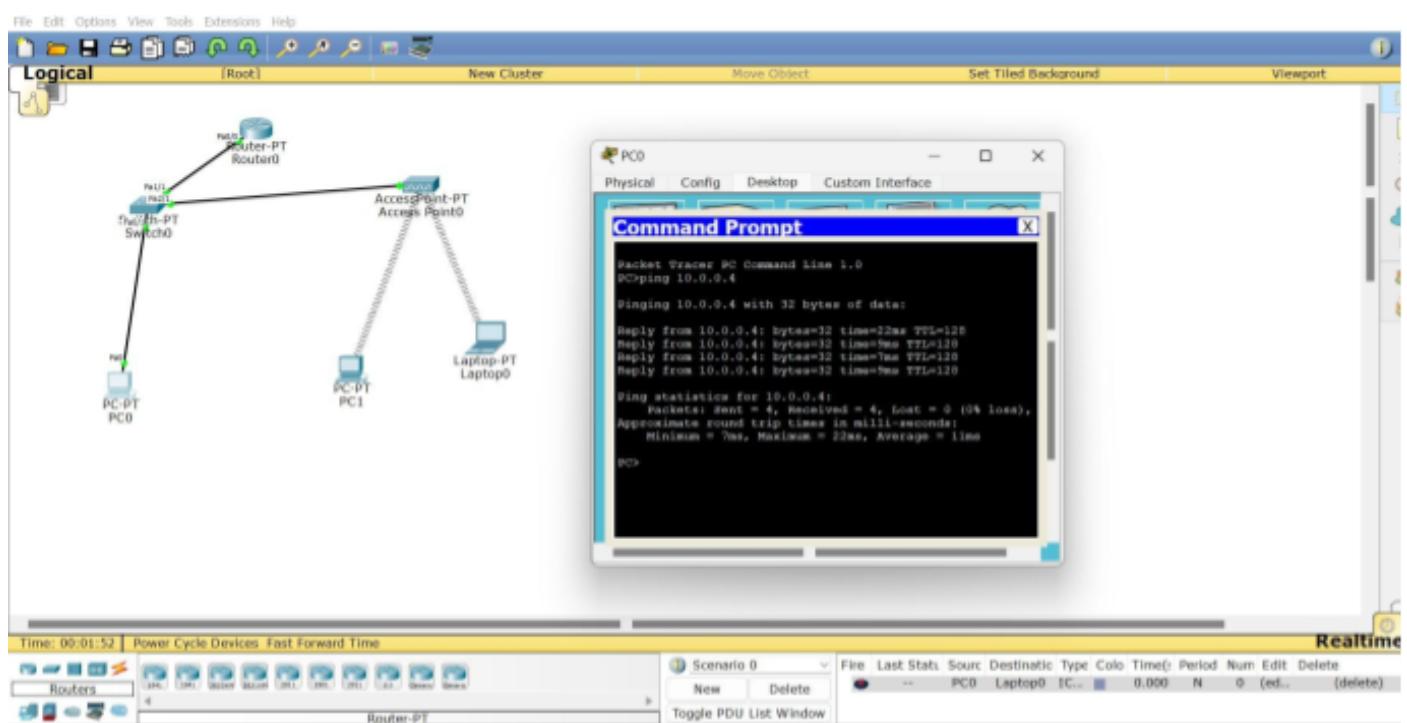
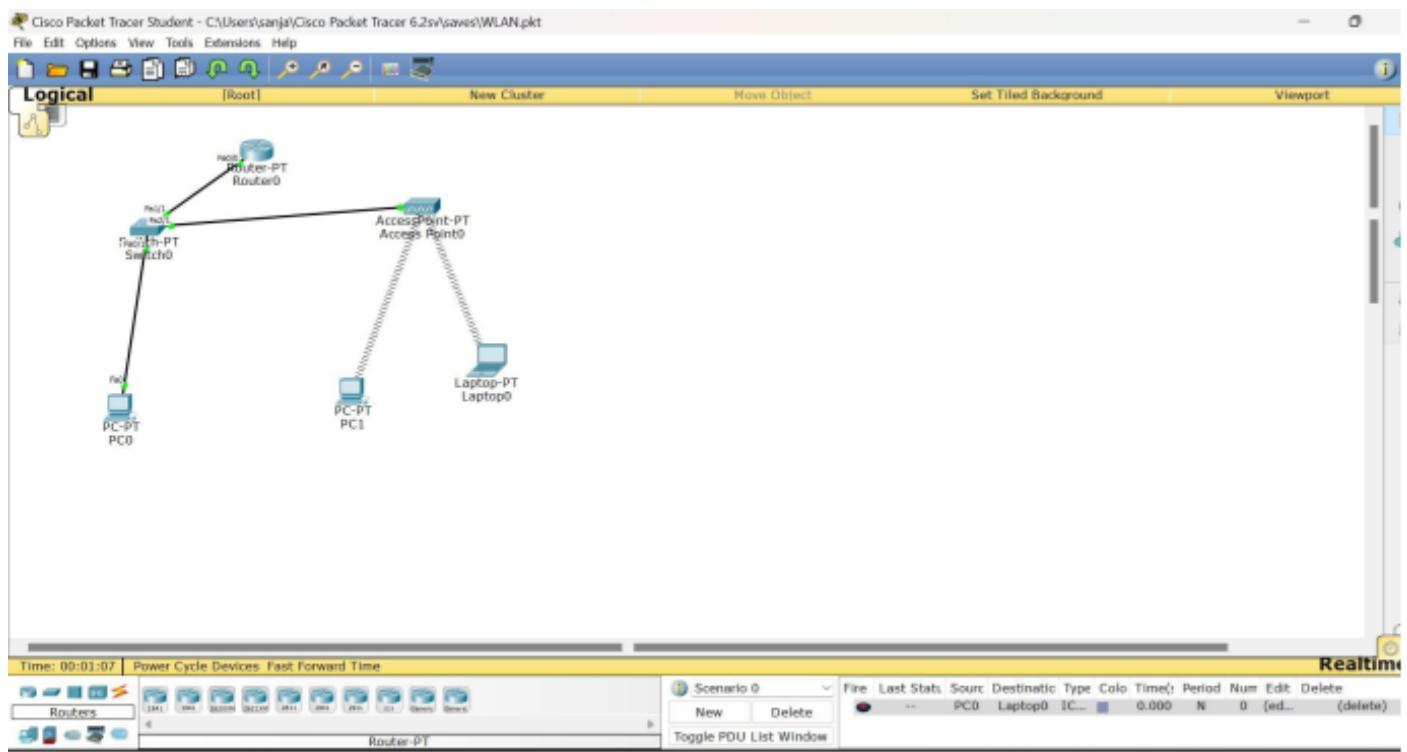
Max length = 1000 bytes, min. 63 bytes

Received 1 packet in 0.01 seconds with 0 bytes

Received 1 packet in 0.01 seconds with 0 bytes

Received 1 packet in 0.01 seconds with 0 bytes

TOPOLOGY & OUTPUT



EXPERIMENT-12

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

classmate
Date _____
Page _____

Program - 12

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

Topology:

The diagram illustrates a simple network topology. A computer (PC) is connected to a router via two ports: 'PC-PT' and 'PC0'. The router has three ports: 'Fa 0', 'Fa 1%', and 'S0/0'. The 'Fa 0' port is connected to the 'PC-PT' port. The 'Fa 1%' port is connected to the 'PC0' port. The 'S0/0' port is connected to a dashed line representing the server room.

Procedure.

- Create topology
- Configure the IP address & gateway for PC0
- Configure router at server room

Step 1: enable

Step 2: configt

Step 3: host name r1

Step 4: Enable secret p1

Step 5: interface fastethernet %0

Step 6: ip address 10.0.0.1 255.0.0.0

Step 7: no shut

Step 8: line vty 0 15

Step 9: login

Step 10: password p0

Step 11: exit, exit

Step 12: wr

Scanned with CamScanner

Ping message to router
 Password for user Access verification is Po
 Password for enable is Pl
 Accessing router CLI from PC
 show ip route

Ping Output

Packet tracer PC command line 1.0

PC > Ping 10.0.0.1

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

Statistics:

packets sent = 4, received = 4, lost = 0%

Approx round trip time in ms

min = 0ms, max = 0ms, Avg = 0ms

PC > telnet 10.0.0.1 port 23

Typing 10.0.0.1 > open

user Access verification is Po

Password: Po

Pl > enable

Password: Pl

rl # show ip route

Routers

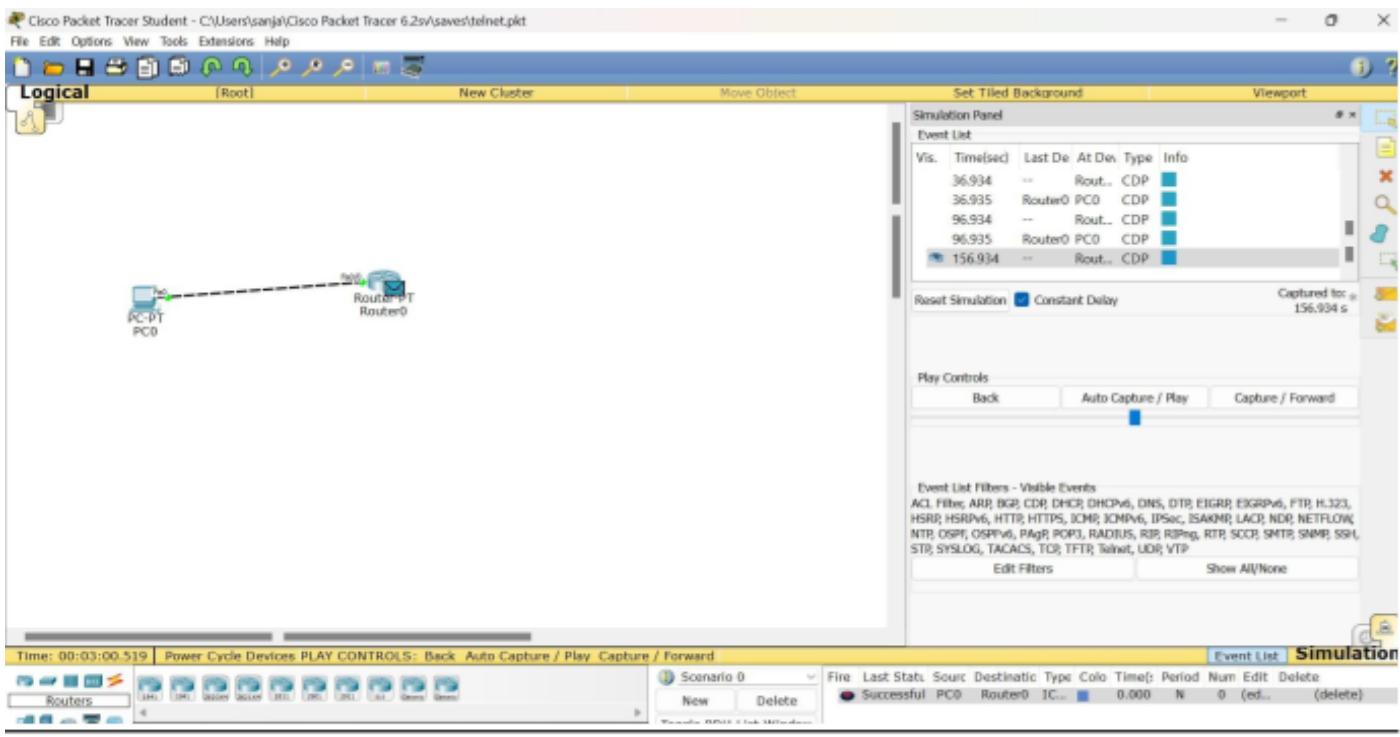
10.0.0.1 is the default gateway

Distance: 1, Metric: 1

Interface: FastEthernet0/0

IP address: 10.0.0.1

TOPOLOGY & OUTPUT



The screenshot shows a "Command Prompt" window titled "PC0" with the following output:

```
Packet Tracer PC Command Line 1.0
PCping 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

Ping statistics for 10.0.0.1:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 0ms, Maximum = 1ms, Average = 0ms

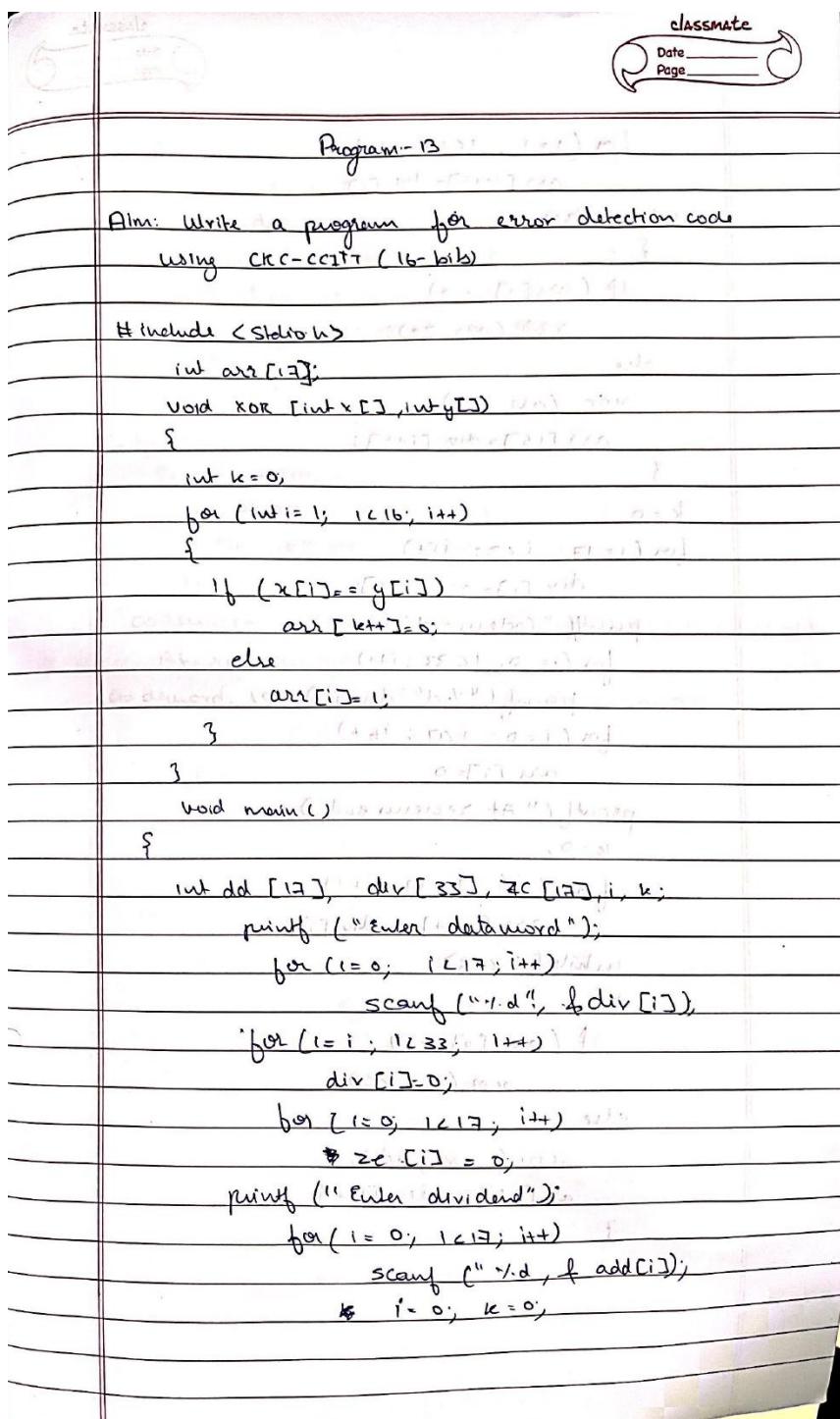
PCDtelnet 10.0.0.1
Trying 10.0.0.1 ...Open

User Access Verification
Password:
```

The command prompt shows the user attempting to ping and telnet to the router at 10.0.0.1. The ping is successful, and the telnet connection is established.

EXPERIMENT-13

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



Scanned with CamScanner

```

for (i=1; i<17; i++)
    arr[k++] = div[i];
while (i < 33)
{
    if (arr[0] == -o)
        xor (arr, ze);
    else
        xor (arr, dd);
    arr[16] = div[i++];
}
k = 0;
for (i=17; i < 33; i++)
    div[i] = arr[k++];
printf("Code word");
for (i=0; i < 33; i++)
    printf("%d", div[i]);
for (i=0; i < 17; i++)
    arr[i] = 0;
printf("At receiver end");
k = 0;
for (i=i; i < 17; i++)
    arr[k++] = div[i];
while (i < 33)
{
    if (arr[0] == -o)
        xor (arr, ze);
    else
        xor (arr, dd);
    arr[16] = div[i++];
}

```

```

K=0;
for (i=17; i<33; i++)
    div[i]=arr[k+i];
printf ("codeword ");
for (i=0; i<33; i++)
    printf ("%d", div[i]);
3

```

Output =

Enter datamword
10 11 00 1111 00 10111

Enter divisor
10 0010 0000 01 00011

codeword: 10 11 00.1111 00 10 1110 00 00 00 00 11 011

At receiver end
codeword: 10 11 00 11 11 00 10 11 10 00 0 00 0 00 0 00 0 00

CODE-

```

import java.util.Scanner;
import java.util.Arrays;

class Program {

    static String Xor(String a, String b) {
        String result = "";
        int n = b.length();
        for (int i = 1; i < n; i++) {
            result=(a.charAt(i) == b.charAt(i))?0:1;
        }
        return result;
    }

    static String Div(String data, String key) {
        int pick = key.length();
        String tmp = data.substring(0, pick);

```

```

int n = data.length();
while (pick < n) {
    if (tmp.charAt(0) == '1')
        tmp = Xor(data, tmp) + data.charAt(pick);
    else
        tmp = Xor(new String(new char[pick]).replace("\0", "0"), tmp) + data.charAt(pick);
    pick += 1;
}
if (tmp.charAt(0) == '1')
    tmp = Xor(divisor, tmp);
else
    tmp = Xor(new String(new char[pick]).replace("\0", "0"), tmp);
return tmp;
}

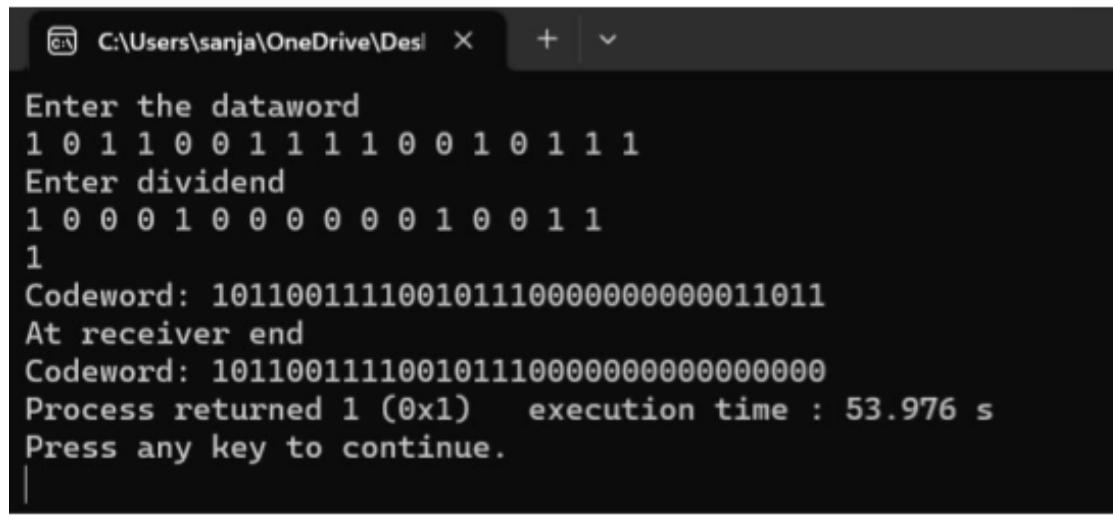
static void Encode(String data, String key) {
    int lkey = key.length();
    String appended_data = (data + new String(new char[lkey - 1]).replace("\0", "0"));
    String remainder = Mod2Div(appended_data, key);
    String codeword = data + remainder;
    System.out.println("Remainder : " + remainder);
    System.out.println("Encoded Data (Data + Remainder) :" + codeword + "\n");
}

public static void main(String[] args) {
    Scanner s = new Scanner(System.in);
    System.out.println("enter dataword and key");
    String data = s.next();
}

```

```
String key = s.next();  
  
EncodeData(data, key);  
}  
}
```

OUTPUT



The screenshot shows a terminal window with the following output:

```
C:\Users\sanja\OneDrive\Desktop> Enter the dataword  
1 0 1 1 0 0 1 1 1 1 0 0 1 0 1 1 1  
Enter dividend  
1 0 0 0 1 0 0 0 0 0 1 0 0 1 1  
1  
Codeword: 1011001111001011100000000000011011  
At receiver end  
Codeword: 10110011110010111000000000000000  
Process returned 1 (0x1) execution time : 53.976 s  
Press any key to continue.
```

EXPERIMENT-14

Q) Write a program for congestion control using Leaky bucket algorithm

Program 14.

Aim: Write a program for congestion control using leaky bucket algorithm

```
#include < stdio.h >
#include < stdlib.h >

int main()
{
    int bucket, outlet, k = 1, num, remaining;
    printf("Enter bucket size and outstream size");
    scanf("%d %d", &bucket, &outlets);
    remaining = bucket;
    while (k)
    {
        num = rand() % 1000;
        if (num < remaining)
        {
            remaining = remaining - num;
            printf("Packets of %d bytes are accepted", num);
        }
        else
        {
            printf("Packets of %d bytes is discarded", num);
        }
        if (buckets - remaining > outlets)
        {
            remaining += outlets;
        }
        else
        {
            remaining = buckets;
        }
    }
}
```

Scanned with CamScanner

while (remaining < buckets).
{

if (buckets - remaining > outlets)
{

 remaining += outlets;
}

else {

 remaining = buckets;

 printf ("Remaining bytes : %d", remaining);
}

}

Output:

Enter bucket size and outstream size

0 1000 200

~ Packets of 411 bytes are accepted

Remaining bytes : 1000

If you want to stop press 0, otherwise 1.

packets of 467 bytes are accepted

Remaining bytes 733

If you want to stop press 0, otherwise 1.

packets of 334 bytes are accepted

Remaining bytes = 599

CODE-

```
import java.util.*;

class Leakybucket {

    public static void main(String[] args)

    {

        int rem;

        Scanner sc=new Scanner(System.in);

        int s= 0;

        System.out.println("enter no of queries,buffer size,input and output packet size ");

        int q=sc.nextInt();

        int bs=sc.nextInt();

        int ip=sc.nextInt();

        int op=sc.nextInt();

        for (int i = 0; i < q; i++) {

            rem=bs-s;

            if (ip <= (rem)) {

                System.out.println("packet is accepted");

                s+=ip;

            }

            else {

                System.out.println("Packet not accepted ");

            }

            System.out

                .println("remaining space="+(bs-s));

            s -= op;

        }

    }

}
```

OUTPUT

```
PS C:\Users\sanja> cd C:\Users\sanja\OneDrive\Documents
PS C:\Users\sanja\OneDrive\Documents> javac Leakybucket.java
PS C:\Users\sanja\OneDrive\Documents> java Leakybucket
enter no of queries,buffer size,input and output packet size
4 10
6
1
packet is accepted
remaining space=4
Packet not accepted
remaining space=5
packet is accepted
remaining space=0
Packet not accepted
remaining space=1
PS C:\Users\sanja\OneDrive\Documents> █
```

EXPERIMENT-15

Q) 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.

Date _____
Page _____

Programs - 15

Aim: Using TCP/IP sockets, write a client-server program to make circuit sending

client TCP.py

```
from socket import *
server_name = '127.0.0.1'
server_port = 12000
client_socket = socket(AF_INET, SOCK_STREAM)
client_socket.connect((server_name, server_port))
sentence = input("In enter file name: ")
```

Client socket.send(sentence.encode())
file_content = clientsocket.recv(1024).decode()
print("In from server: \n")
print(file_content)
clientsocket.close()

Server TCP.py

```
from socket import *
serverport = 12000
servername = "127.0.0.1"
serversocket = socket(AF_INET, SOCK_STREAM)
serversocket.bind((servername, serverport))
serversocket.listen(1)

while 1:
    print("The server is ready to receive")
    connectionsocket, addr = serversocket.accept()
    sentence = connectionsocket.recv(1024).decode()
```

Scanned with CamScanner

Ques

```
file = open("sentence", "r")
l = file.read(1024)
connectionSocket.send(str.encode(l))
print("In sent contents of "+ sentence)
file.close()
connectionSocket.close()
```

CODE-

ClientTCP.py

```
from socket import *
serverName = "127.0.0.1"
serverPort = 12000
clientSocket = socket(AF_INET, SOCK_STREAM)
clientSocket.connect((serverName, serverPort))
sentence = input("\nEnter file name:")

clientSocket.send(sentence.encode())
filecontents = clientSocket.recv(1024).decode()
print ("\nFrom Server:\n")
print(filecontents)
clientSocket.close()
```

ServerTCP.py

```
from socket import *
serverName='127.0.0.1'
serverPort = 12000
serverSocket = socket(AF_INET,SOCK_STREAM)
serverSocket.bind((serverName,serverPort))
serverSocket.listen(1)
while 1:
    print("The server is ready to receive")
    connectionSocket, addr = serverSocket.accept()
    sentence = connectionSocket.recv(1024).decode()

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

    connectionSocket.send(l.encode())
    print ("\nSent contents of " + sentence)
    file.close()
    connectionSocket.close()
```

OUTPUT

The screenshot shows two windows side-by-side. The left window is titled "ServerTCP.py - C:/Users/sanja/OneDrive/Documents/ServerTCP.py (3.9.13)" and contains the Python code for a TCP server. The right window is titled "ClientTCP.py - C:/Users/sanja/OneDrive/Documents/ClientTCP.py (3.9.13)" and contains the Python code for a TCP client.

```
ServerTCP.py code:
from socket import *
serverName='127.0.0.1'
serverPort = 12000
serverSocket = socket(AF_INET,SOCK_STREAM)
serverSocket.bind((serverName,serverPort))
serverSocket.listen(1)
while 1:
    print("The server is ready to receive")
    connectionSocket, addr = serverSocket.accept()
    sentence = connectionSocket.recv(1024).decode()
    file=open(sentence,"r")
    l=file.read(1024)

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

ClientTCP.py code:
from socket import *
serverName = "127.0.0.1"
serverPort = 12000
clientSocket = socket(AF_INET, SOCK_STREAM)
clientSocket.connect((serverName,serverPort))
sentence = input("\nEnter file name:")

clientSocket.send(sentence.encode())
filecontents = clientSocket.recv(1024).decode()
print ("\nFrom Server:\n")
print(filecontents)
clientSocket.close()
```

The screenshot shows two windows of the IDLE shell. The left window is titled "*IDLE Shell 3.9.13*" and the right is titled "IDLE Shell 3.9.13". Both show the execution of the respective TCP scripts.

```
IDLE Shell 3.9.13 output (Server):
File Edit Shell Debug Options Window Help
Python 3.9.13 (tags/v3.9.13:6de2ca5, May 17 2022, 16:36:42) [MSC v.1929 64 b
it (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
=====
RESTART: C:/Users/sanja/OneDrive/Documents/ServerTCP.py =====
The server is ready to receive
The server is ready to receive
The server is ready to receive
=====
RESTART: C:/Users/sanja/OneDrive/Documents/ServerTCP.py =====
===
The server is ready to receive
Sent contents of ServerTCP.py
The server is ready to receive

IDLE Shell 3.9.13 output (Client):
File Edit Shell Debug Options Window Help
the target machine actively refused it
>>>
=====
RESTART: C:/Users/sanja/OneDrive/Documents/ClientTCP.py =====
Enter file name:ServerTCP.py
=====
RESTART: C:/Users/sanja/OneDrive/Documents/ClientTCP.py =====
Enter file name:ServerTCP.py
=====
RESTART: C:/Users/sanja/OneDrive/Documents/ClientTCP.py =====
Enter file name:
===== RESTART: C:/Users/sanja/OneDrive/Documents/ClientTCP.py =====
Enter file name:ServerTCP.py
From Server:
from socket import *
serverName='127.0.0.1'
serverPort = 12000
serverSocket = socket(AF_INET,SOCK_STREAM)
serverSocket.bind((serverName,serverPort))
serverSocket.listen(1)
while 1:
    print("The server is ready to receive")
    connectionSocket, addr = serverSocket.accept()
    sentence = connectionSocket.recv(1024).decode()
    file=open(sentence,"r")
    l=file.read(1024)

    connectionSocket.send(l.encode())
    print ("\nSent contents of " + sentence)
    file.close()
    connectionSocket.close()
```

EXPERIMENT-16

Q) 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

QUESTION

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 required file if present

```
client UDP.py
from socket import *
server_name = "127.0.0.1"
server_port = 12000
client_socket = socket(AF_INET, SOCK_DGRAM)
sentence = input("\n Enter file name ")
client_socket.sendto(sentence.encode("utf-8"), (server_name, server_port))
file_contents, server_address = client_socket.recvfrom(2048)
print(f"\n Reply from server, {file_contents.decode('utf-8')}")
for i in file_contents:
    print(str(i), end="")
client_socket.close()
```

Server UDP.py

```
from socket import *
server_port = 12000
server_socket = socket(AF_INET, SOCK_DGRAM)
server_socket.bind(("127.0.0.1", server_port))
print("The server is ready to receive")
```

Scanned with CamScanner

while (1)

sentence, client address = sever socket.recvfrom(2048)

sentence = sentence.decode("Utf-8")

file = open(sentence, "r")

con = file.read(2048)

sever socket.sendto(byts(con, "Utf-8"), client address)

print('In sent content of ', end = '')

print(sentence)

#for(i in sentence)

#print(str(i), end = '')

file.close()

server socket.recvfrom(2048) client address

client address

(2)

server socket.recvfrom(2048) client address

client address

if request == "GET":

if request == "POST":

if request == "PUT":

if request == "DELETE":

if request == "PATCH":

if request == "OPTIONS":

if request == "HEAD":

if request == "TRACE":

if request == "CONNECT":

if request == "PROPFIND":

if request == "MKCOL":

if request == "COPY":

if request == "MOVE":

if request == "DELETE":

if request == "PATCH":

if request == "PUT":

if request == "POST":

CODE-

ClientUDP.py

```
from socket import *
serverName = "127.0.0.1"
serverPort = 12000
clientSocket = socket(AF_INET, SOCK_DGRAM)
sentence = input("\nEnter file name:")

clientSocket.sendto(bytes(sentence,"utf-8"),(serverName, serverPort))
filecontents,serverAddress = clientSocket.recvfrom(2048)
print ("\nReply from Server:\n")
print (filecontents.decode("utf-8"))
#for i in filecontents:
#    print(str(i), end = "")
clientSocket.close()
clientSocket.close()
```

ServerUDP.py

```
from socket import *
serverPort = 12000
serverSocket = socket(AF_INET, SOCK_DGRAM)
serverSocket.bind(("127.0.0.1", serverPort))
print ("The server is ready to receive")
while 1:
    sentence, clientAddress = serverSocket.recvfrom(2048)
    sentence = sentence.decode("utf-8")
    file=open(sentence,"r")
    con=file.read(2048)
    serverSocket.sendto(bytes(con,"utf-8"),clientAddress)
    print ("\nSent contents of ", end = "")
    print (sentence)
    # for i in sentence:
    #     print (str(i), end = "")
    file.close()
```

OUTPUT

The image shows four windows from the IDLE Python shell version 3.9.13. The top-left window contains the code for ClientUDP.py, which reads a file and sends its contents to a server at 127.0.0.1 port 12000. The top-right window contains the code for ServerUDP.py, which listens for files and returns their contents. The bottom-left window shows the execution of ClientUDP.py, where it prompts for a file name and then prints the reply from the server. The bottom-right window shows the execution of ServerUDP.py, which prints "The server is ready to receive" and "nSent contents of ServerUDP.py".

```
ClientUDP.py - C:/Users/sanja/OneDrive/Documents/ClientUDP.py [3.9.13]
File Edit Format Run Options Window Help
from socket import *
serverName = "127.0.0.1"

serverPort = 12000
clientSocket = socket(AF_INET, SOCK_DGRAM)
sentence = input("\nEnter file name:")
clientSocket.sendto(bytes(sentence,"utf-8"),(serverName, serverPort))
filecontents,serverAddress = clientSocket.recvfrom(2048)
print ("\nReply from Server:\n")
print ([filecontents.decode("utf-8")])
#for i in filecontents:
#    print(str(i), end = "")
clientSocket.close()
clientSocket.close()

ServerUDP.py - C:/Users/sanja/OneDrive/Documents/ServerUDP.py [3.9.13]
File Edit Format Run Options Window Help
from socket import *
serverPort = 12000
serverSocket = socket(AF_INET, SOCK_DGRAM)
serverSocket.bind(("127.0.0.1", serverPort))
print ("The server is ready to receive")
while 1:
    sentence, clientAddress = serverSocket.recvfrom(2048)
    sentence = sentence.decode("utf-8")
    file=open(sentence,"r")
    con=file.read(2048)
    serverSocket.sendto(bytes(con,"utf-8"),clientAddress)
    print ("nSent contents of ", end = "")
    print (sentence)
    # for i in sentence:
    #     print (str(i), end = "")
    file.close()

IDLE Shell 3.9.13
File Edit Shell Debug Options Window Help
Python 3.9.13 (tags/v3.9.13:6de2ca5, May 17 2022, 16:36:42) [MSC v.1929 64 b
it (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> RESTART: C:/Users/sanja/OneDrive/Documents/ClientUDP.py
-----
Enter file name:
----- RESTART: C:/Users/sanja/OneDrive/Documents/ClientUDP.py
-----
Enter file name:ServerUDP.py
Reply from Server:

from socket import *
serverPort = 12000
serverSocket = socket(AF_INET, SOCK_DGRAM)
serverSocket.bind(("127.0.0.1", serverPort))
print ("The server is ready to receive")
while 1:
    sentence, clientAddress = serverSocket.recvfrom(2048)
    sentence = sentence.decode("utf-8")
    file=open(sentence,"r")
    con=file.read(2048)
    serverSocket.sendto(bytes(con,"utf-8"),clientAddress)
    print ("nSent contents of ", end = "")
    print (sentence)
    # for i in sentence:
    #     print (str(i), end = "")
    file.close()

>>>

IDLE Shell 3.9.13
File Edit Shell Debug Options Window Help
Python 3.9.13 (tags/v3.9.13:6de2ca5, May 17 2022, 16:36:42) [MSC v.1929 64 b
it (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> RESTART: C:/Users/sanja/OneDrive/Documents/ServerUDP.py
-----
The server is ready to receive
nSent contents of ServerUDP.py
```

EXPERIMENT-17

Q) Tool Exploration - Wireshark

11/17/

Tool Exploration - Wireshark

Wireshark is an open source packet analyzer which is used for education, analysis, software development, telecommunications, protocol development and network troubleshooting. It is used to track packets so that each one is filtered to meet our specific needs. It is commonly called as a sniffer, network protocol analyzer, network analyzer. It is also used by network security engineers to examine security problems.

Wireshark is a free application used to comprehend data (both end-to-end). It is also called as free packet sniffer computer application, ports, network card and an interface mode i.e. it accepts all packets which it receives.

Uses -

- 1) It is used by network security engineer to examine security problems.
- 2) It is used by network engineer to troubleshoot network issues.
- 3) It is also used to analyze dropped packets.
- 4) It helps us to troubleshoot latency issues between machines on the network.
- 5) It helps us to know all the devices like laptop, mobile phones, desktop, switch, etc.

Date / /
Page / /

Waller, commandet un o den. netwerk
or the rest of the world

Functionality of Wireshark.

It is used as similar to a TCP
dump in networking. It has a graphic
and non-graphic and filtering functions.
It also monitors the unicast traffic
which is not sent to network's MAC
address interface. The port monitoring is
a method to monitor network traffic. When
it is enabled switch sends copies of
all network packets present at one port to
another.

Features of Wireshark.

- It is a multi-platform software i.e. it can run the
same on Linux, Windows OS, FreeBSD,
NetBSD, etc.
- It is a standard free open source software.
- It performs deep inspection of chart of
protocols.
- It even has sort and filter option
which makes easier to user to know
the data.
- It can capture raw - loss traffic.
- Useful in IP analysis.
- Also enables the analysis & e from
different types of network like ethernet, sockets
etc through which we can read the file.