VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"JnanaSangama", Belgaum -590014, Karnataka.

LAB REPORT
on

Computer Networks

Submitted by

J S MEGHANA (1BM18CS039)

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 "LAB COURSE **Computer Networks**" carried out by **J S MEGHANA (1BM18CS039),** 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 Latha N.R.
Assistant Professor

Department of CSE
BMSCE, Bengaluru

Dr. Jyothi S Nayak
Professor and Head

Department of CSE
BMSCE, Bengaluru

`

Index

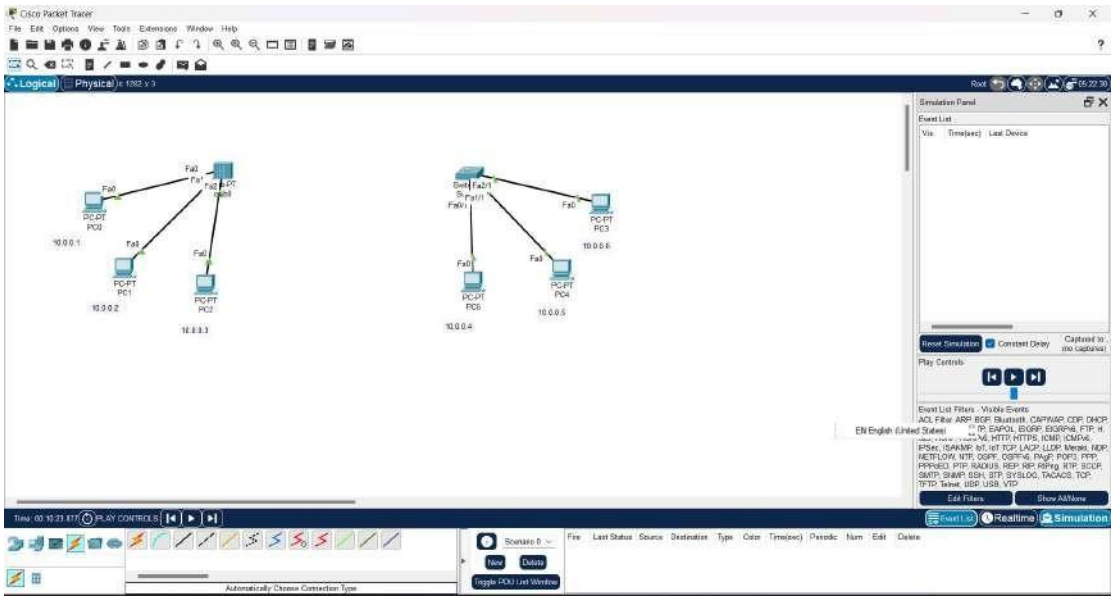
Experiment No. 1

Cycle 1

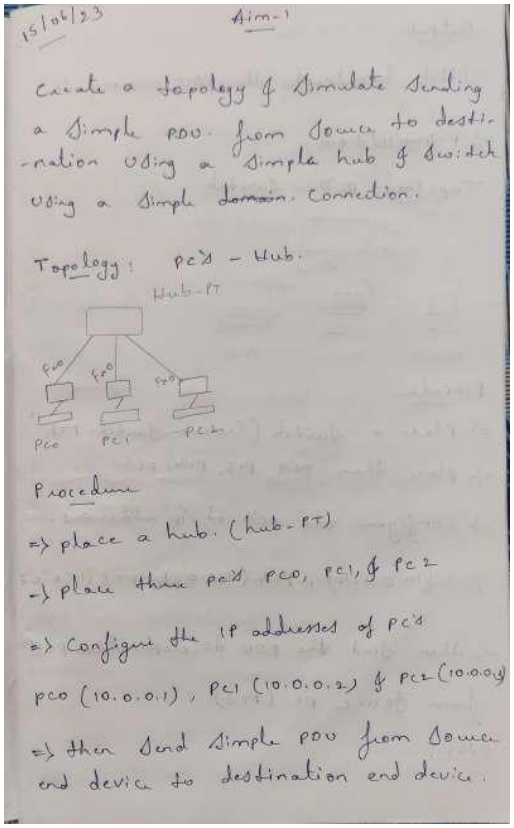
Aim-1

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

Topology:



Procedure for Hub:


15/06/23 Aim-1

Create a topology & simulate sending a simple PDU from source to destination using a simple hub & switch using a simple domain connection.

Topology: PC's - Hub.

Hub-PT

Fa0 Fa0 Fa0

PC0 PC1 PC2

Procedure

=> place a hub. (hub-PT)

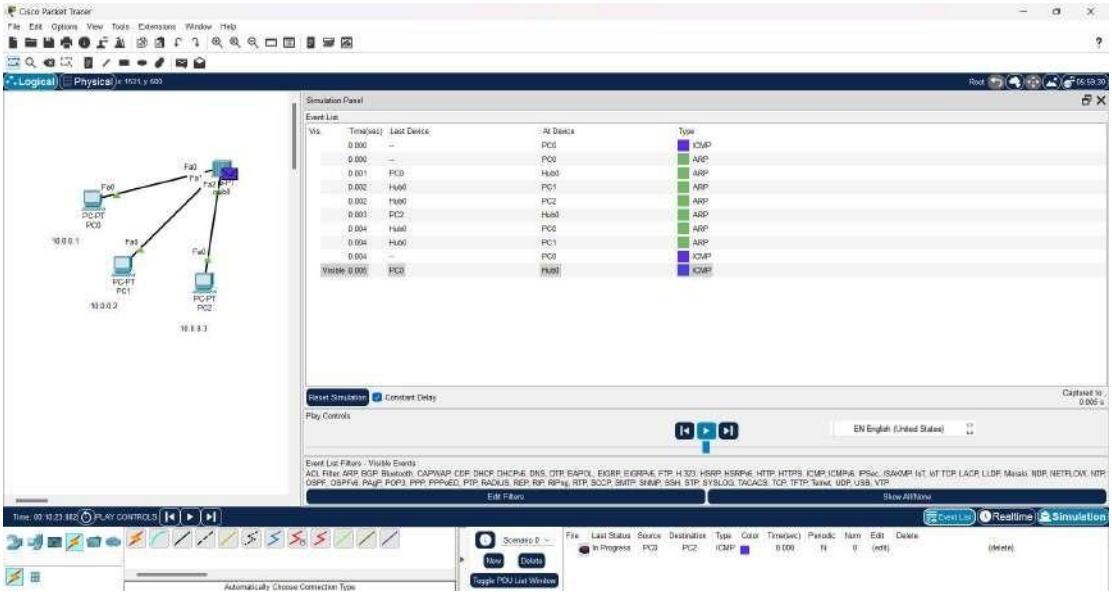
=> place three PC's PC0, PC1, & PC2

=> Configure the IP addresses of PC's

PC0 (10.0.0.1), PC1 (10.0.0.2) & PC2 (10.0.0.3)

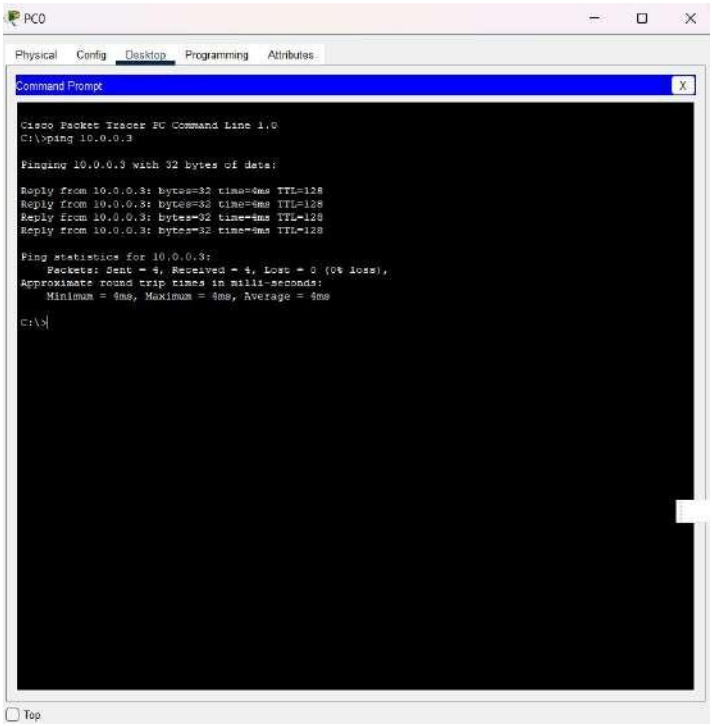
=> then send simple PDU from source end device to destination end device.

Simulation:



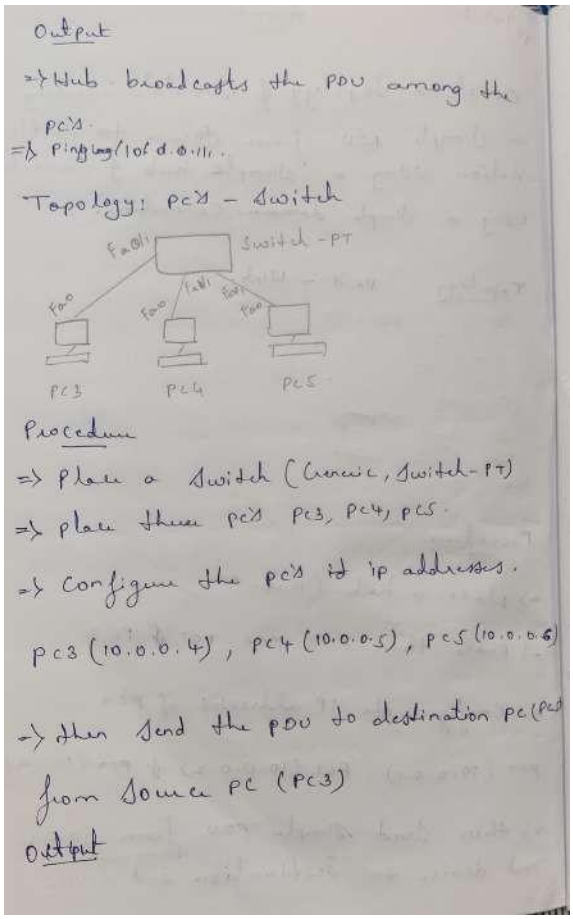
Sending PDU from source device (pc0) to destination device (pc2).

Ping Response:

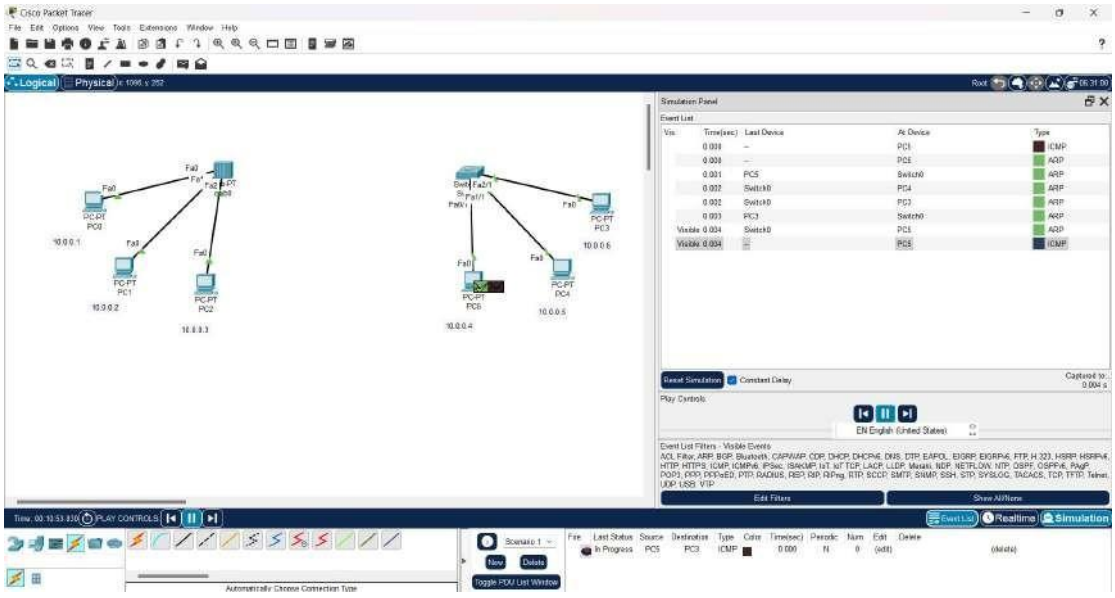


Ping response from pc0 to pc2.

Procedure for Switch:

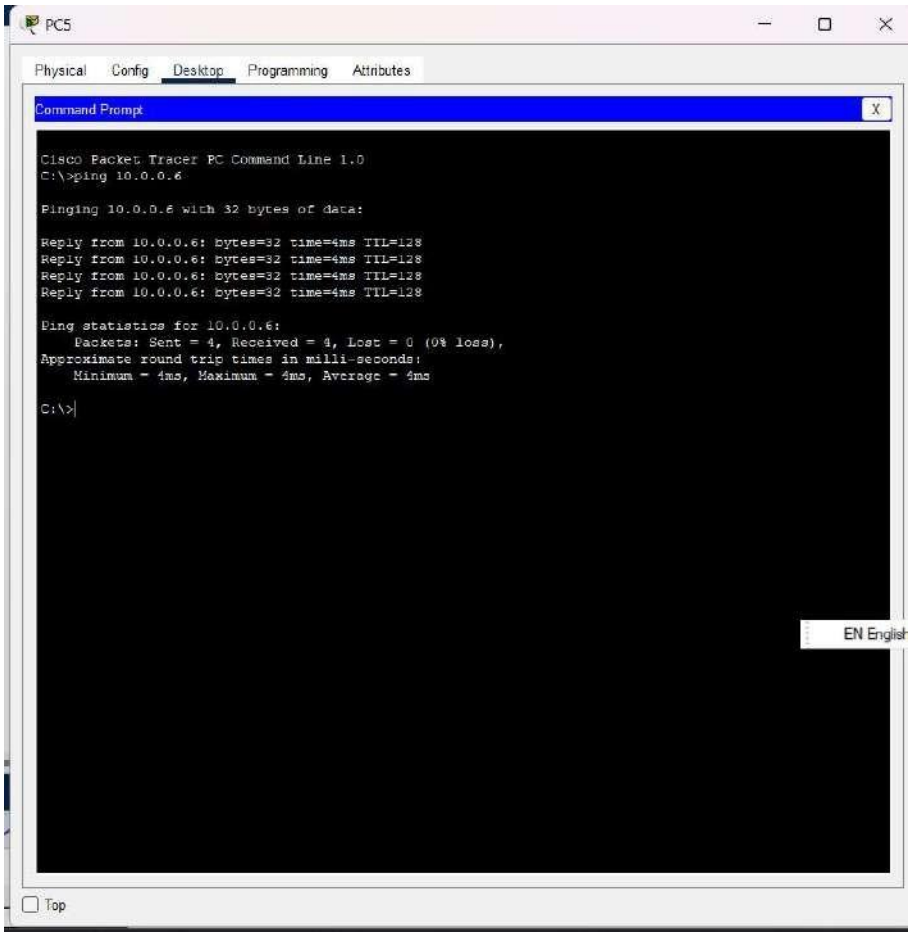


Simulation:



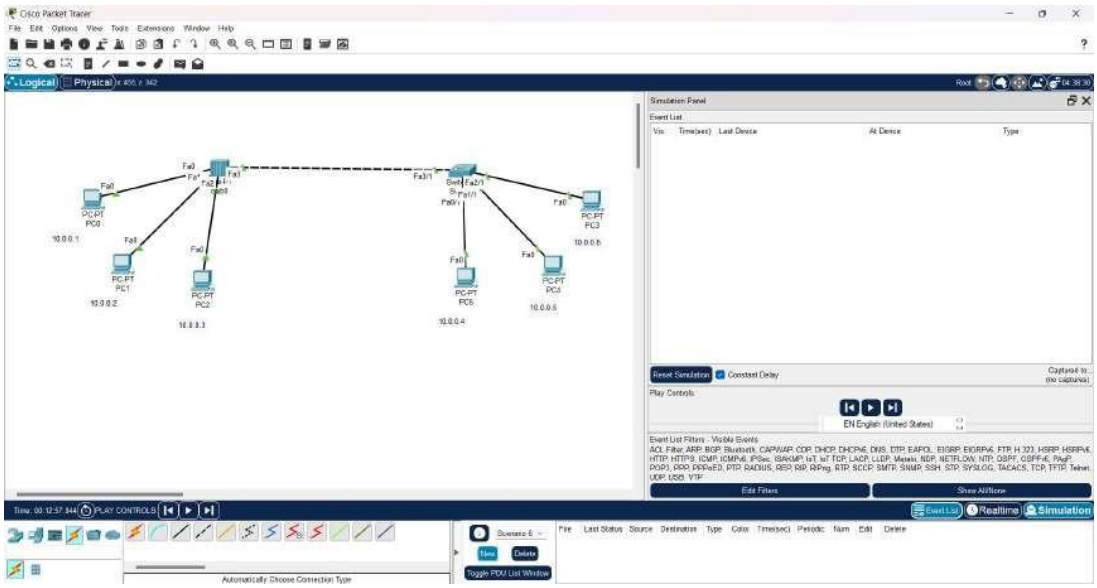
Sending PDU from source device (pc5) to destination device (pc3).

Ping Response:

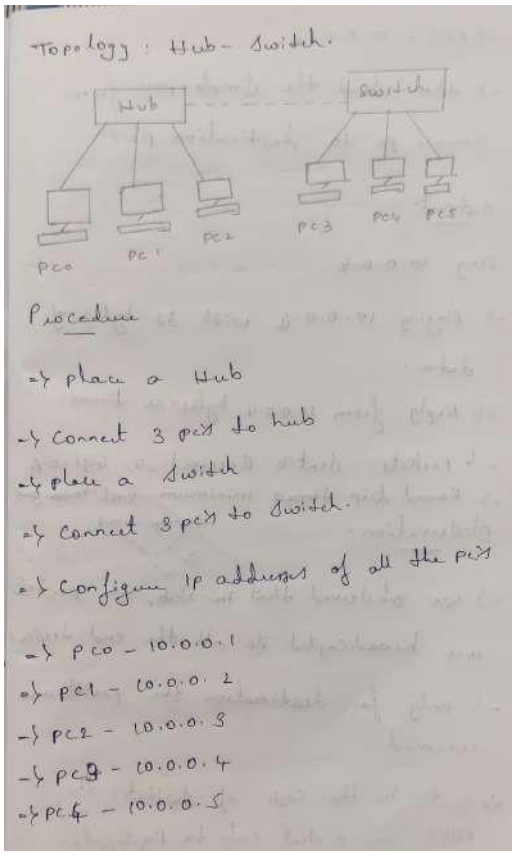


Ping response from pc5 to pc3.

Topology (Hub-Switch):



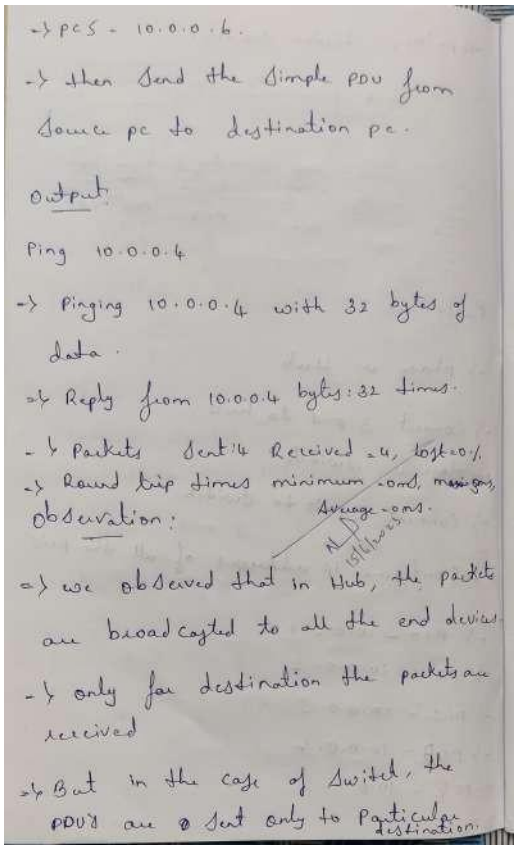
Procedure (Hub-Switch):



Topology : Hub - Switch.

Procedure

-> place a Hub
-> Connect 3 pcs to hub
-> place a Switch
-> Connect 3 pcs to Switch.
-> Configure IP addresses of all the pcs
-> pc0 - 10.0.0.1
-> pc1 - 10.0.0.2
-> pc2 - 10.0.0.3
-> pc3 - 10.0.0.4
-> pc4 - 10.0.0.5



-> pc5 - 10.0.0.6.
-> then Send the Simple PDU from Source pc to destination pc.

Output

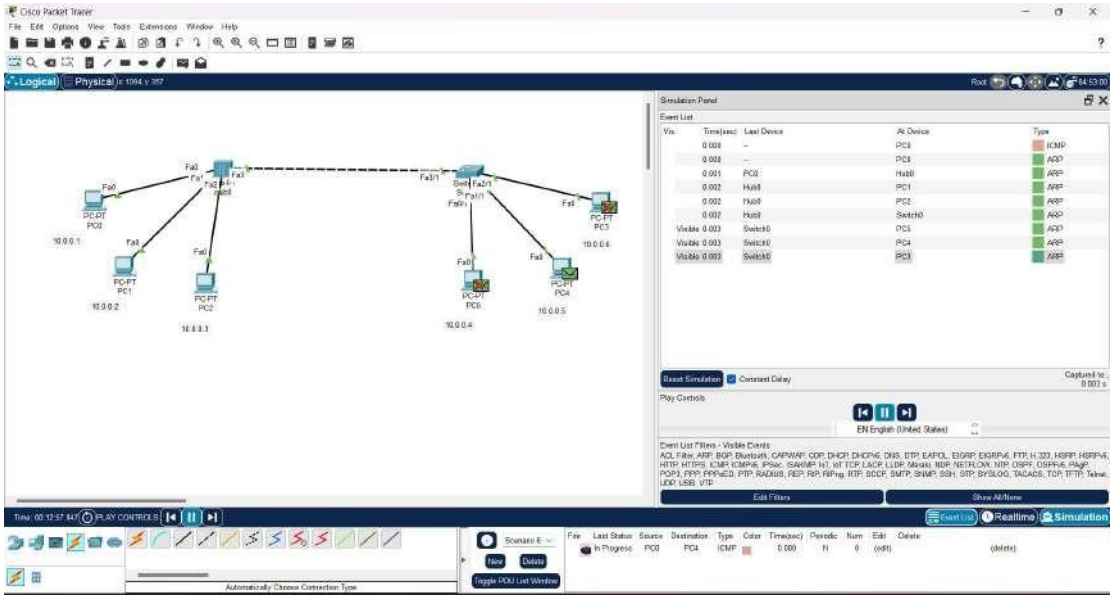
Ping 10.0.0.4

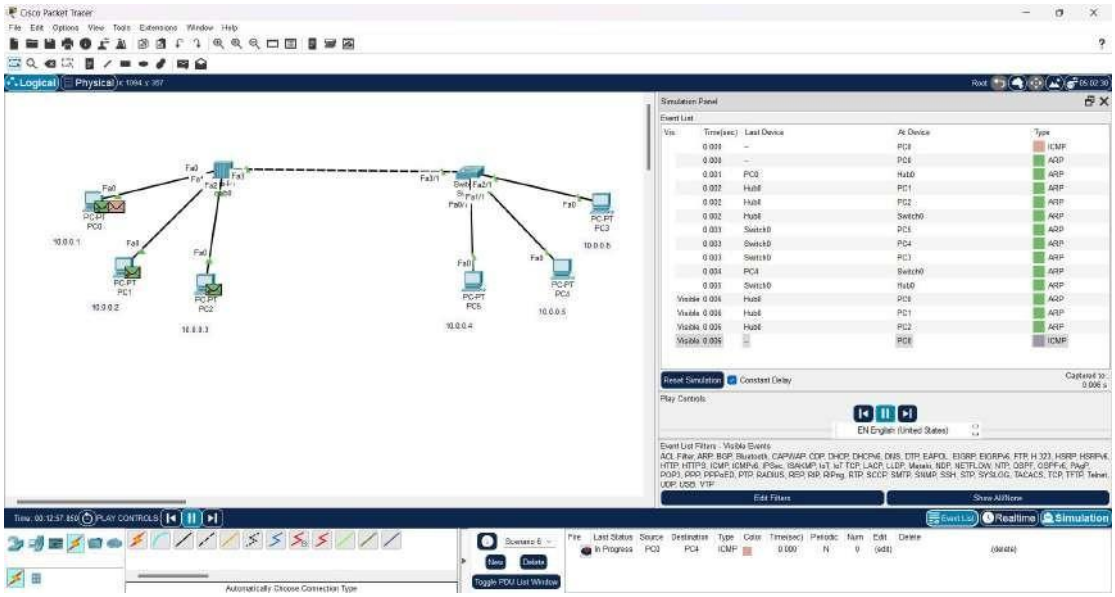
-> Pinging 10.0.0.4 with 32 bytes of data.
-> Reply from 10.0.0.4 bytes:32 times.
-> Packets Sent:4 Received=4, lost=0%
-> Round trip times minimum=0ms, maximum=0ms, Average=0ms.

Observation:

-> we observed that in Hub, the packets are broadcasted to all the end devices
-> only for destination the packets are received
-> But in the case of Switch, the PDUs are sent only to Particular destination.

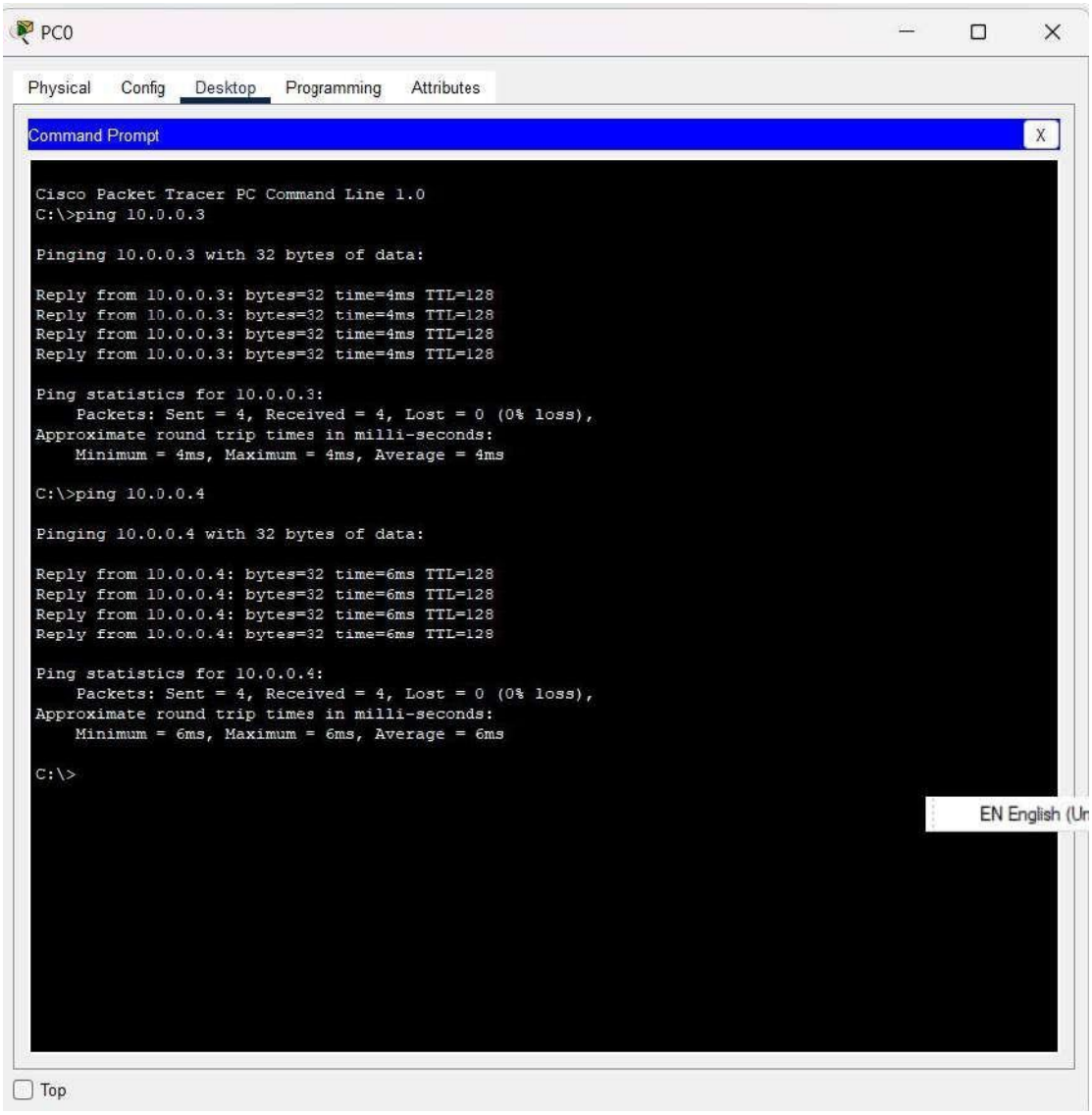
Simulation:





Sending PDU from source device (pc0) to destination device (pc4).

Ping Response:



Ping response from pc0 to pc5.

Experiment No. 2

Cycle 1

Aim-2

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

Topology:

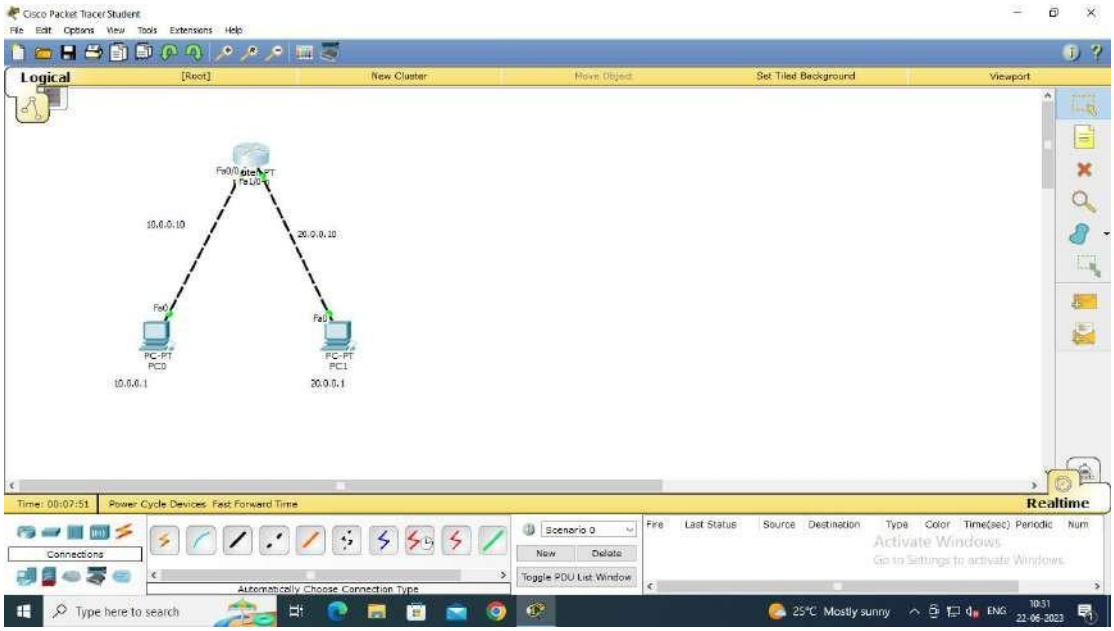


Fig 1: Topology

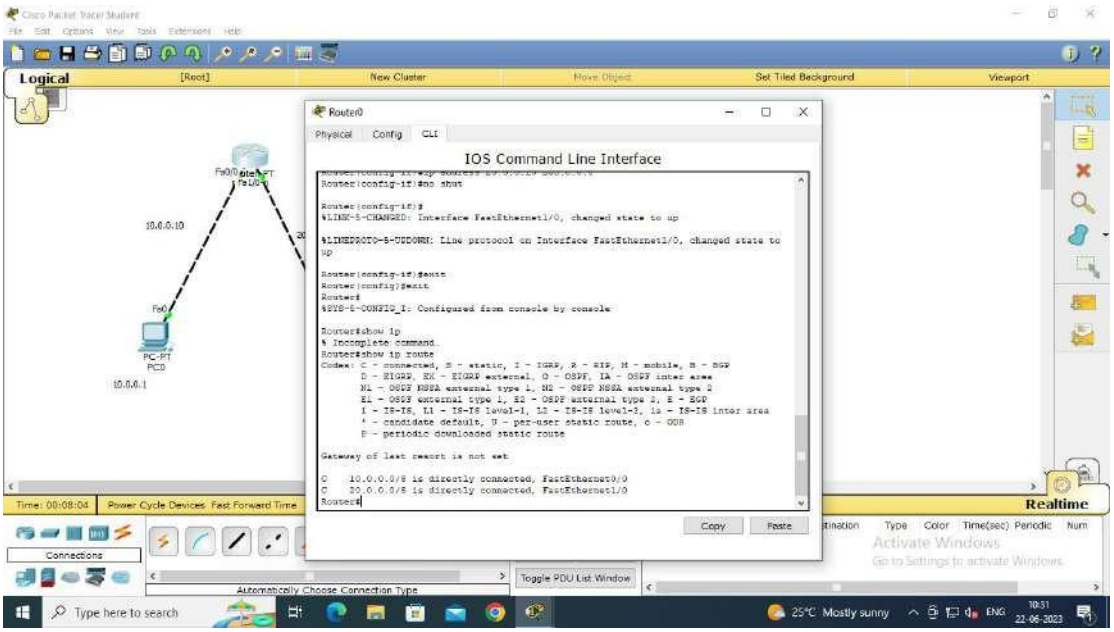


Fig 2: Router showing its connections after configuration

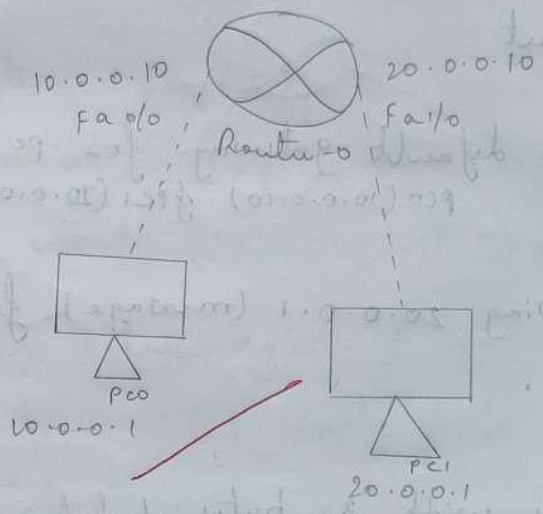
Procedure and Observation:

22/06/23

Aim - 2

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

Topology



Procedure:

=> Configuring the IP address of two PC's i.e; PC0 (10.0.0.1) & PC1 (20.0.0.1).

=> Configuring the 1st part of router

-> n

-> enable

-> config t
-> interface fastethernet 0/0
-> ip address 10.0.0.10 255.0.0.0
-> no shut
-> exit

=> Configuring the other port of router
-> interface fastethernet 1/0
-> ip address 20.0.0.10 255.0.0.0
-> no shut
-> exit

-> configuring default gateway for pc's i.e;
PC0 (10.0.0.10) & PC1 (20.0.0.10)

Output

-> Passing ping 20.0.0.1 (message) from pc0 to pc1.

Response
Pinging 20.0.0.1 with 32 bytes of data:
Request timed out

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

Ping Statistics for 20.0.0.1:
Packets: Sent = 4, Received = 3, Lost = 1 (25% loss),

Approximate round trip times in milli-seconds:

Minimum = 0ms, Maximum = 0ms, Average = 0ms.

-> If try pinging message without setting default gateway the reply would be "destination unreachable".

-> After completely configuring if you get & If tried pinging again the 1st reply would be "request timed out".

-> Then later again if tried will get the successful reply from source to destination pcs.

Rec
13/7/23

Output:

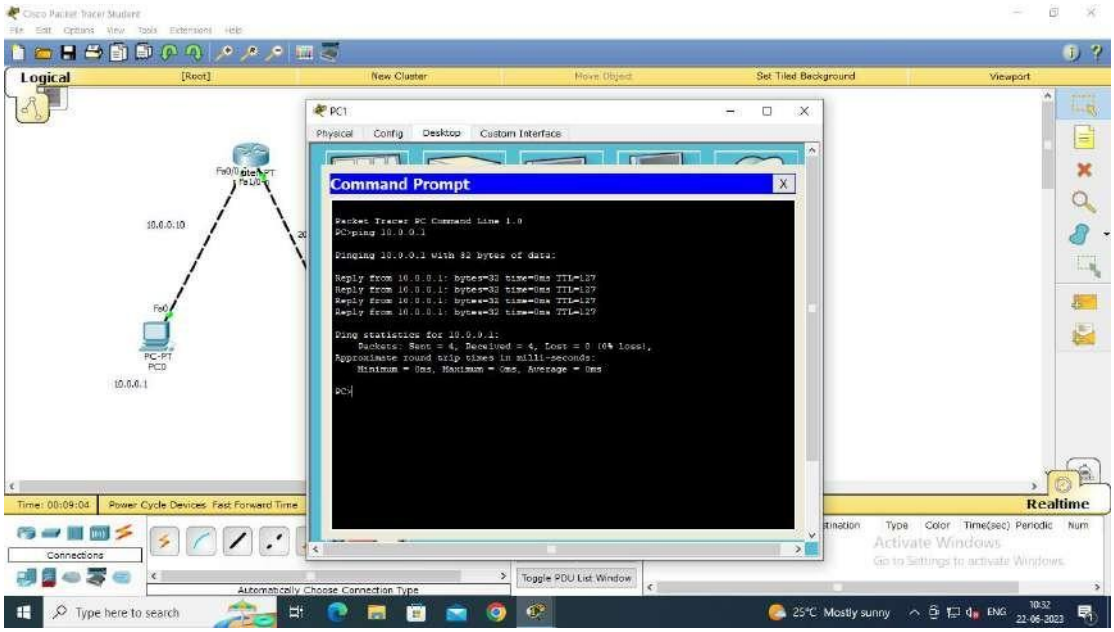


Fig 3: Pinging from pc1 to pco

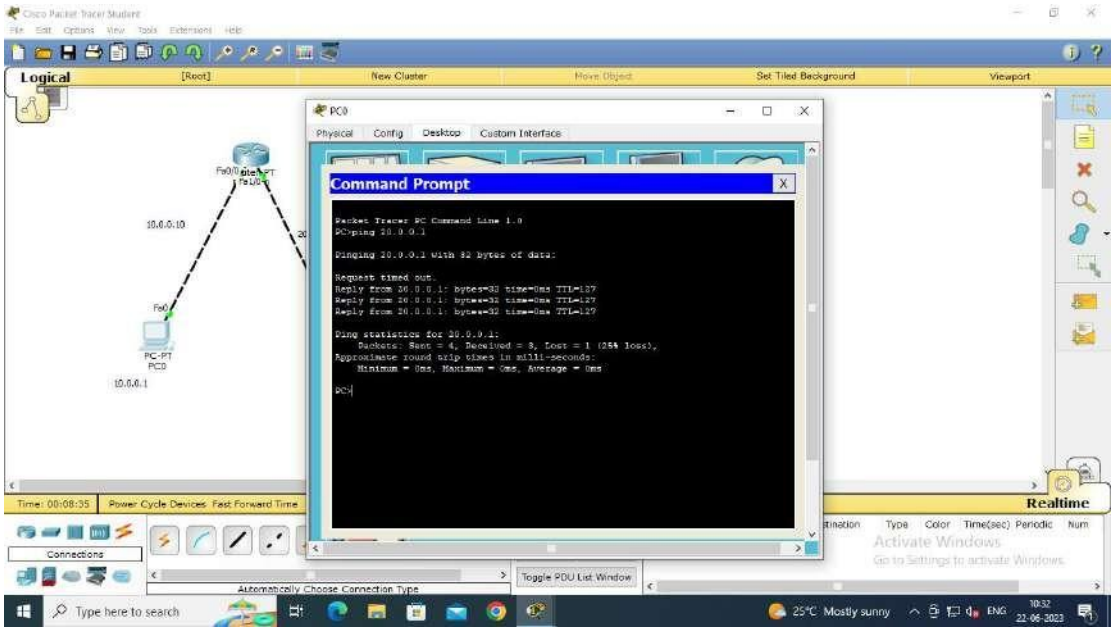


Fig 4: Pinging from pc0 to pc1

Experiment No. 3

Cycle 1

Aim-3

3. Configure default route, static route to the Router

Static Route:
Topology:

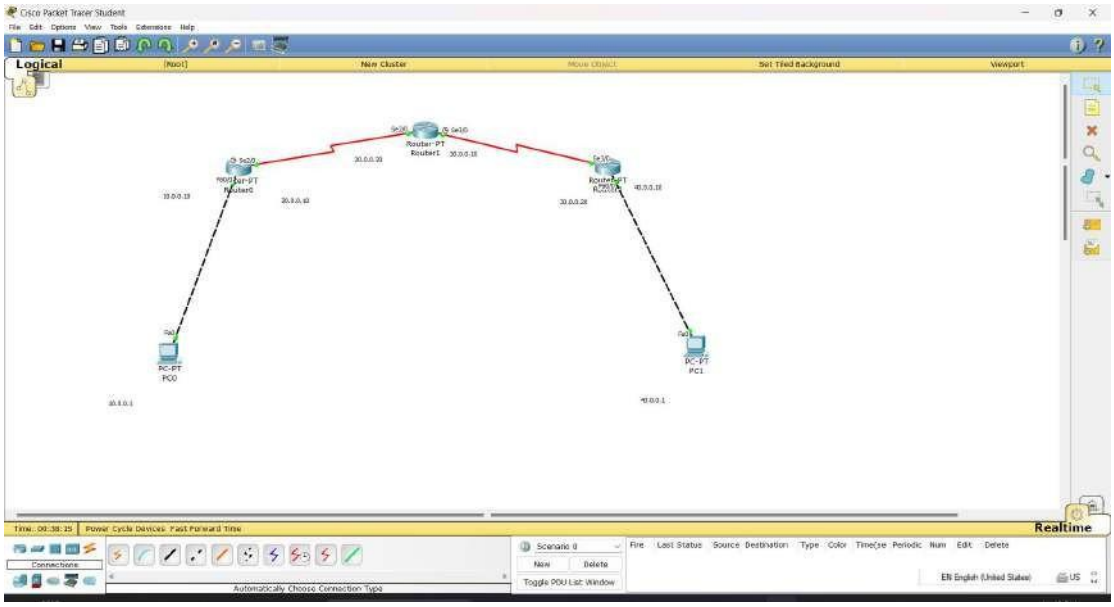


Fig 1: Topology

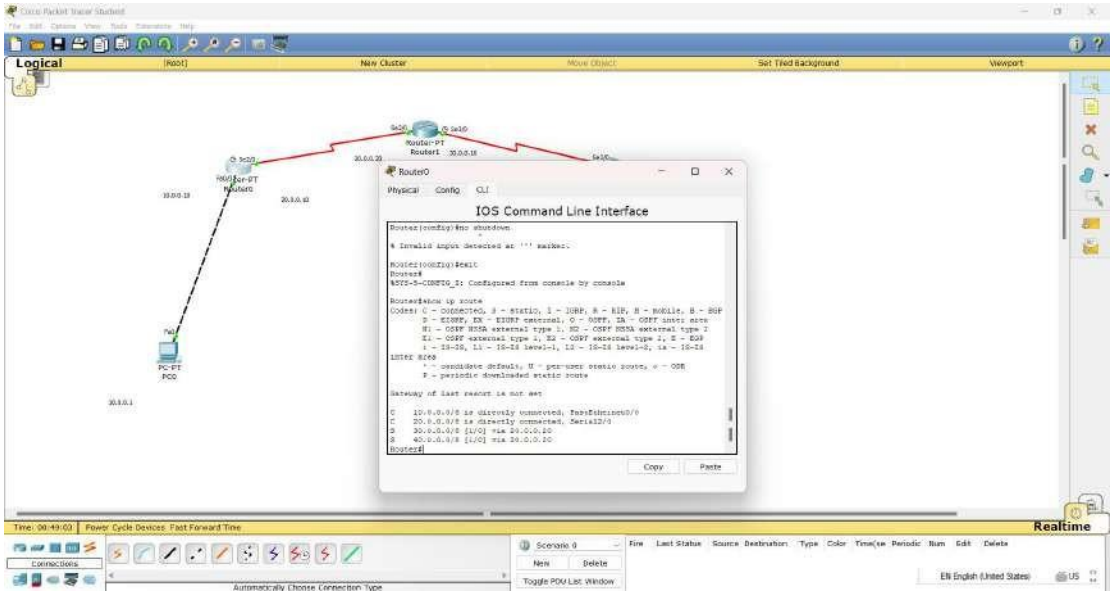


Fig 2: Router0 Connections

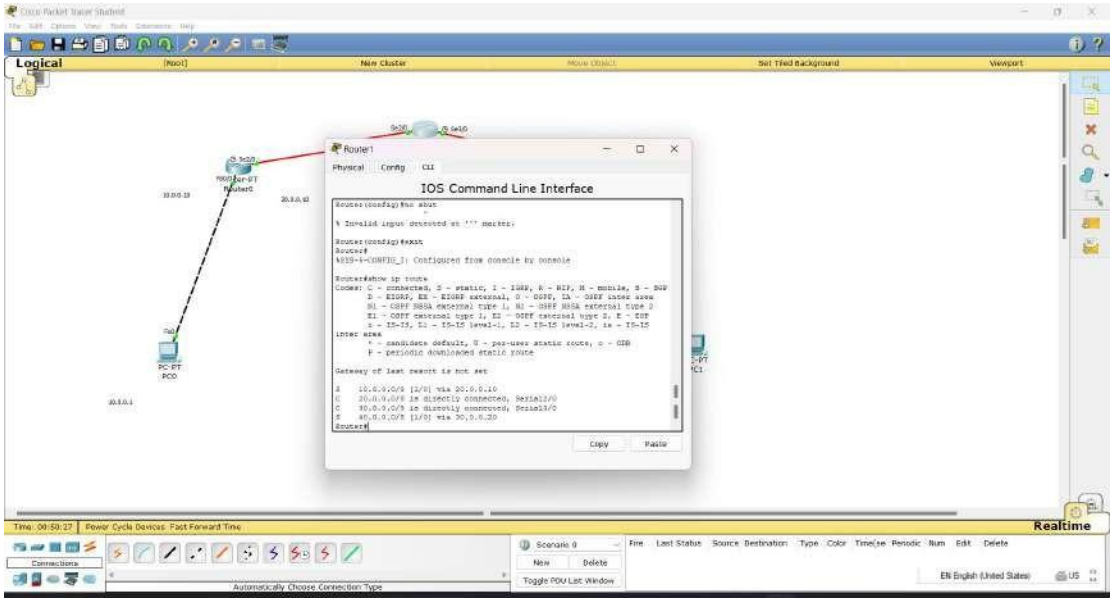


Fig 3: Router1 Connections

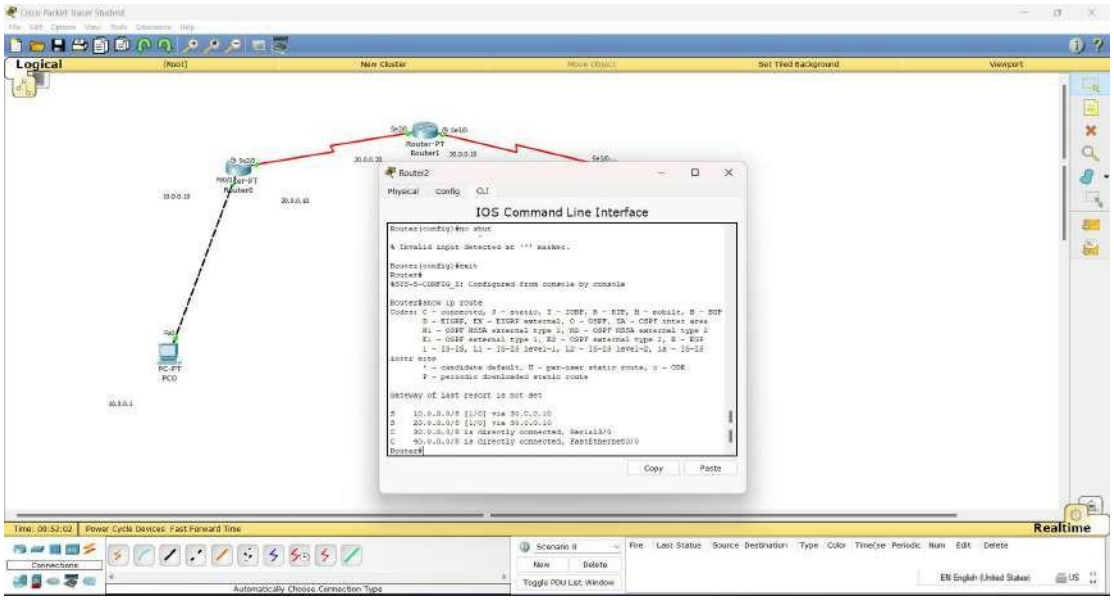


Fig 4: Router2 Connections

Procedure and Observation:

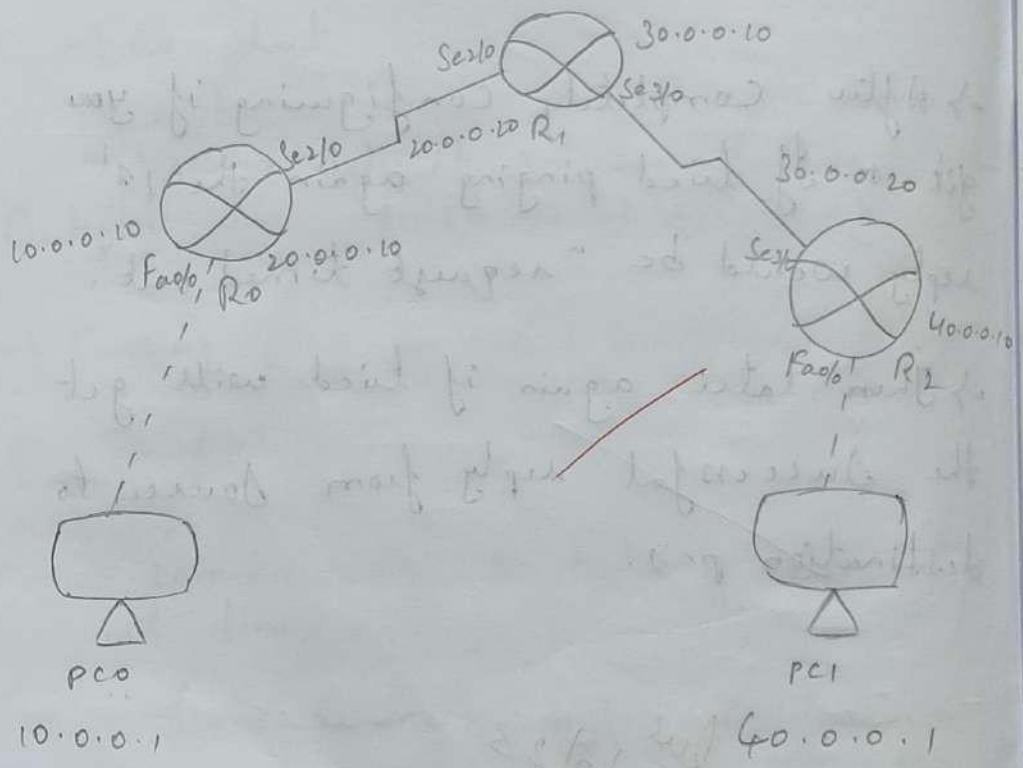
13/07/23

Aim-3

configure default route, static route to the router.

Static Route

Topology



Procedure

-> Configure the IP addresses of the PCs i.e; for PC0 (10.0.0.1) & for PC1 (40.0.0.1).

-> Now configuring the routers.

Router 0

-> n

-> enable

-> Config t

-> interface fastethernet 0/0

-> IP address 10.0.0.10 255.0.0.0

-> no shut

-> exit.

(for Serial).

-> interface Serial 2/0

-> IP address 20.0.0.10 255.0.0.0

-> no shut

-> exit

Router 1

-> n
-> enable
-> Config t
-> interface Serial 2/0
-> ip address 20.0.0.20 255.0.0.0
-> no shut down
-> exit
(for Serial 3 port).
-> interface Serial 3/0
-> ip address 30.0.0.10 255.0.0.0
-> no shut
-> exit.

Router 2

-> n
-> enable
-> Config t
-> ip address interface Serial 3/0

-> ip address 30.0.0.20 255.0.0.0
-> no shut
-> exit.
(for fastethernet).
-> interface fastethernet 0/0
-> ip address 40.0.0.10 255.0.0.0
-> no shut
-> exit.
-> Now check the connections and see if all of them are connected.

Router 0
-> exit
-> show ip route

C 10.0.0.0/8 is directly connected, FastEthernet 0/0
C 20.0.0.0/8 is directly connected, Serial 2/0

Router 1
-> exit
-> show ip route
C 20.0.0.0/8 is directly connected, Serial 2/0
C 30.0.0.0/8 is directly connected, Serial 3/0

Router 2

=> exit

=> show ip route

C 30.0.0.0/8 is directly connected, Serial3/0

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

=> Now configuring the static ip addresses of routers so that the communication can happen by passing through all the routers & networks present.

Router 0

=> config t

=> ip route 30.0.0.0 255.0.0.0 20.0.0.20

=> ip route 40.0.0.0 255.0.0.0 20.0.0.20

=> no shut => exit

Router 1

=> config t

=> ip route 10.0.0.0 255.0.0.0 20.0.0.10

-> ip route 40.0.0.0 255.0.0.0 30.0.0.20
=> no shut => exit

Router 2

=> Config t

=> ip route 10.0.0.0 255.0.0.0 30.0.0.10

=> ip route 20.0.0.0 255.0.0.0 30.0.0.10.
=> no shut => exit.
=> Now again check if connections are made properly.

Router 0

=> exit

=> show ip route

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

C 20.0.0.0/8 is directly connected, Serial 2/0

S 30.0.0.0/8 [1/0] via 20.0.0.20

S 40.0.0.0/8 [1/0] via 20.0.0.20

Router 1

=> exit

=> show ip route

S 10.0.0.0/8 [1/0] via 20.0.0.10

C 20.0.0.0/8 is directly connected, Serial 2/0

C 30.0.0.0/8 is directly connected, Serial 3/0

S 40.0.0.0/8 [1/0] via 30.0.0.20

Router 2

-> exit

-> show ip route

S 10.0.0.0/8 [1/0] via 30.0.0.10

S 20.0.0.0/8 [1/0] via 30.0.0.10

C 30.0.0.0/8 is directly connected, Serial 3/0

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

Output

=> Now passing ping message from PC0 to PC1

-> ping 40.0.0.1

Pinging 40.0.0.1 with 32 bytes of data:

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

Reply from 40.0.0.1: byts = 32 time = 18ms
TTL = 125

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

Reply from 40.0.0.1: byts = 32 time = 9ms
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 = 18ms,
Average = 10ms.

=> Now passing ping message from pc1 to pc0

=> Ping 10.0.0.1

Reply from 10.0.0.1: bytes = 32 time = 16 ms
TTL = 125

Reply from 10.0.0.1: bytes = 32 time = 4 ms
TTL = 125

Reply from 10.0.0.1: bytes = 32 time = 12 ms
TTL = 125

Reply from 10.0.0.1: bytes = 32 time = 4 ms
TTL = 125

Ping statistics for 10.0.0.1:

Packets: Sent = 4, Received = 4, Lost = 0
(0% loss)

Approximate round trip in milli-seconds:

Minimum = 4 ms, Maximum = 16 ms,

Average = 9 ms.

Output:

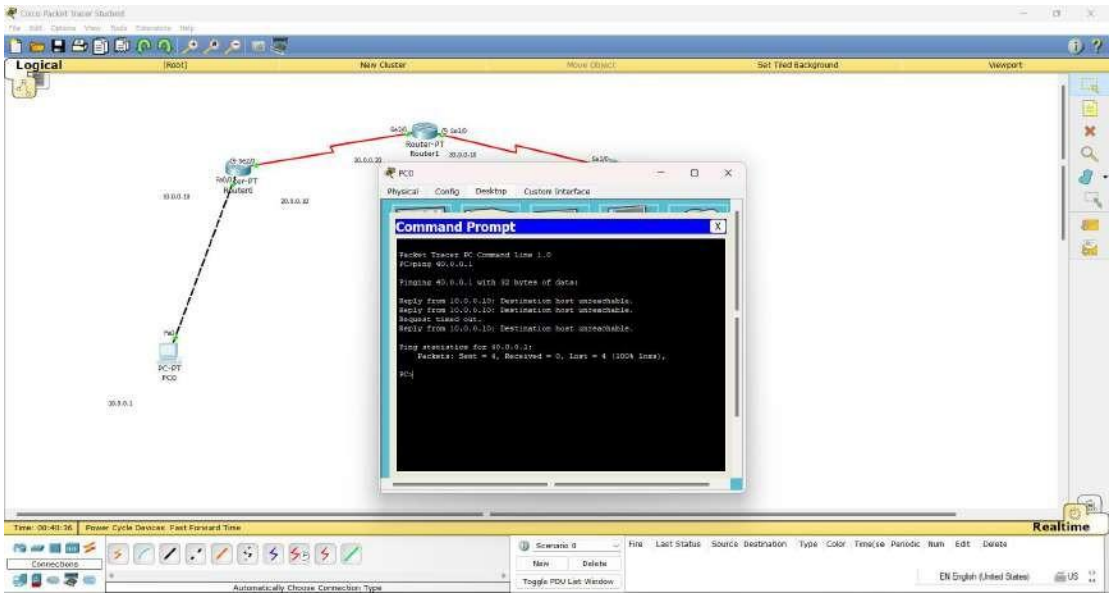


Fig 5: Pinging from pco to pc1 (with the messages destination host unreachable and request timed out)

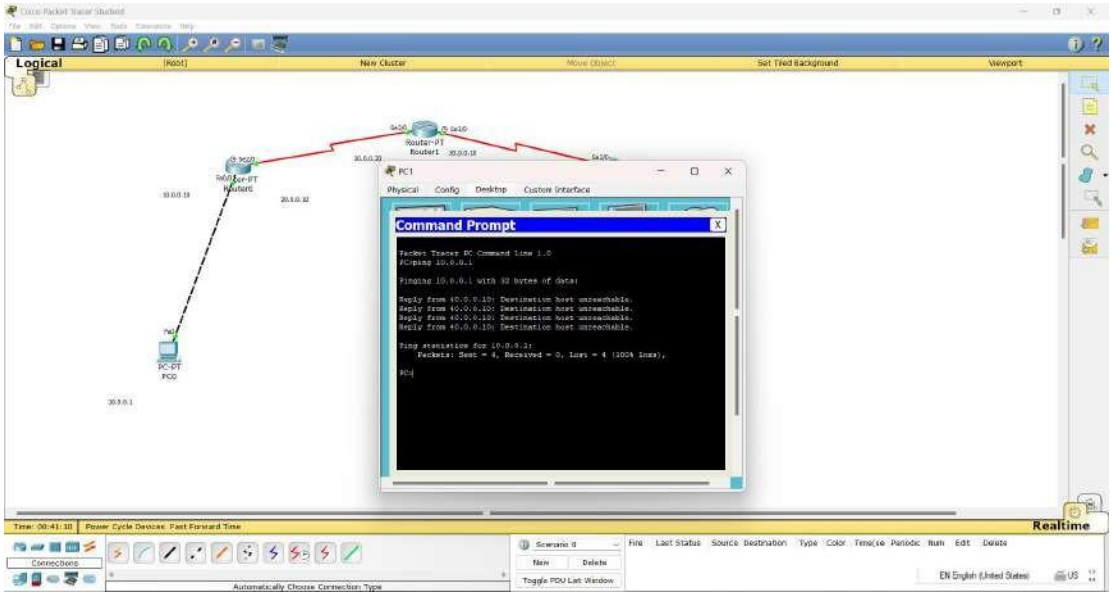


Fig 6: Pinging from pc1 to pc0 (with the messages destination host unreachable)

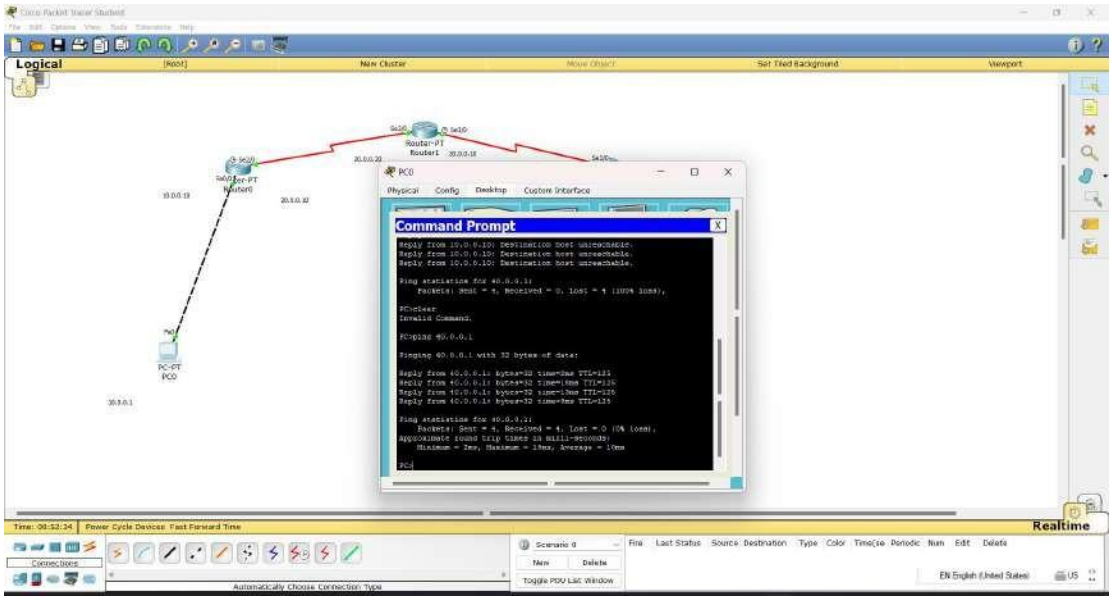


Fig 7: Pinging from pc0 to pc1 (message passing successful)

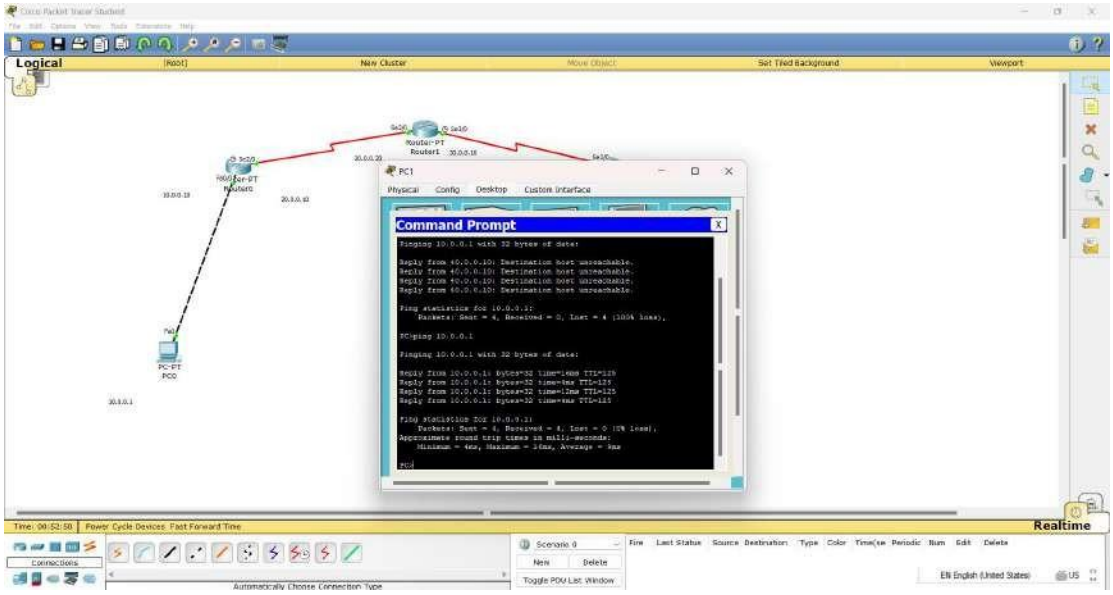


Fig 8: Pinging from pc1 to pc0 (message passing successful)

Default Route:

Topology:

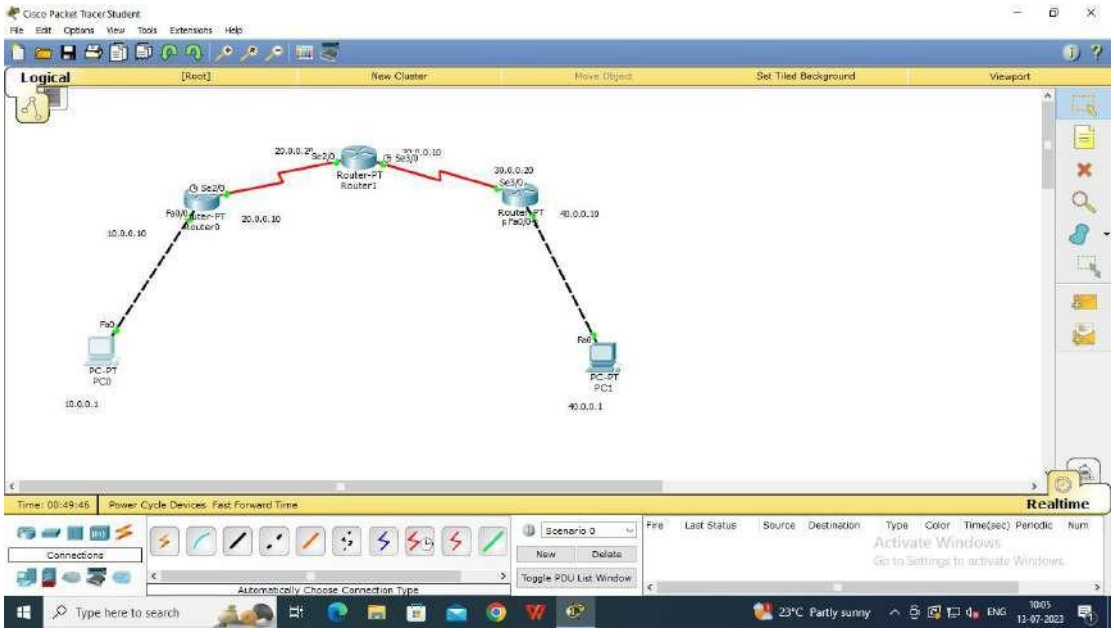


Fig 1: Topology

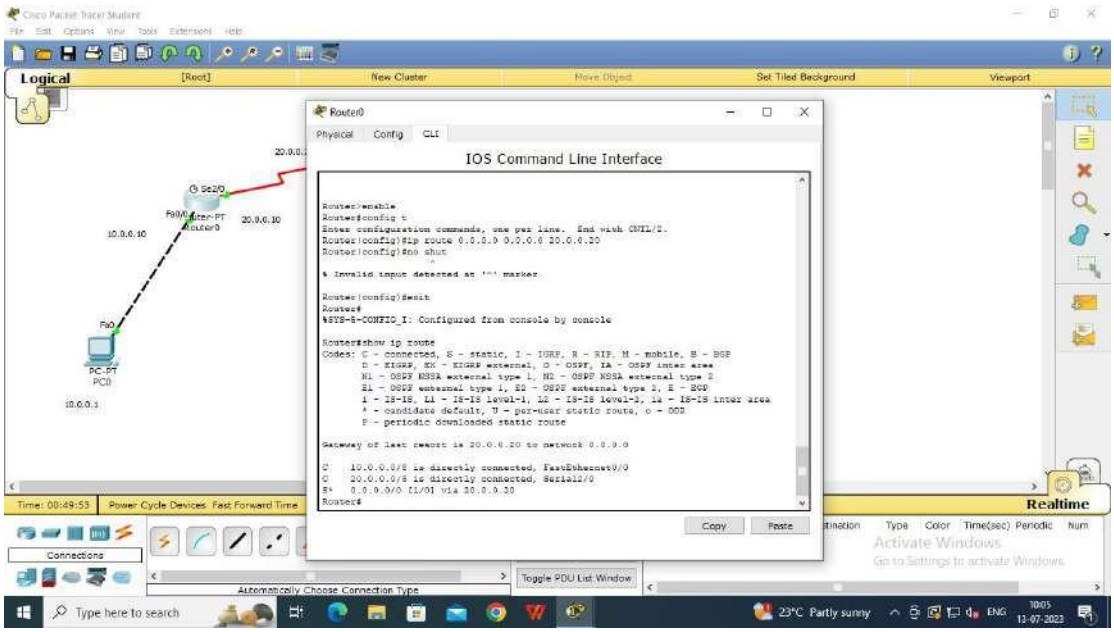


Fig 2: Router0 Connections

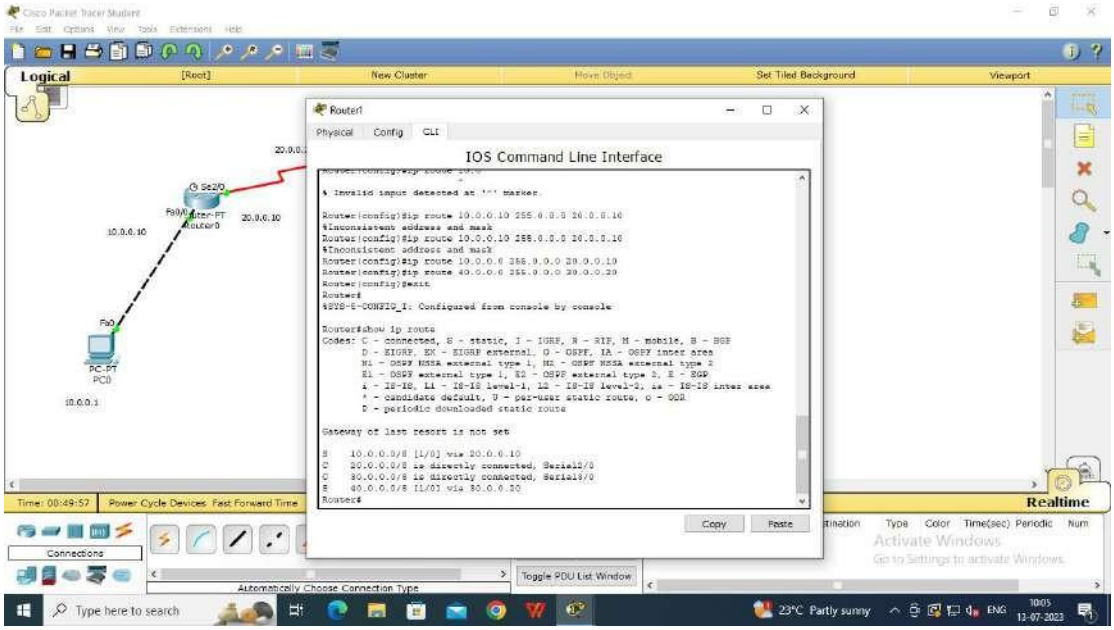


Fig 3: Router1 Connections

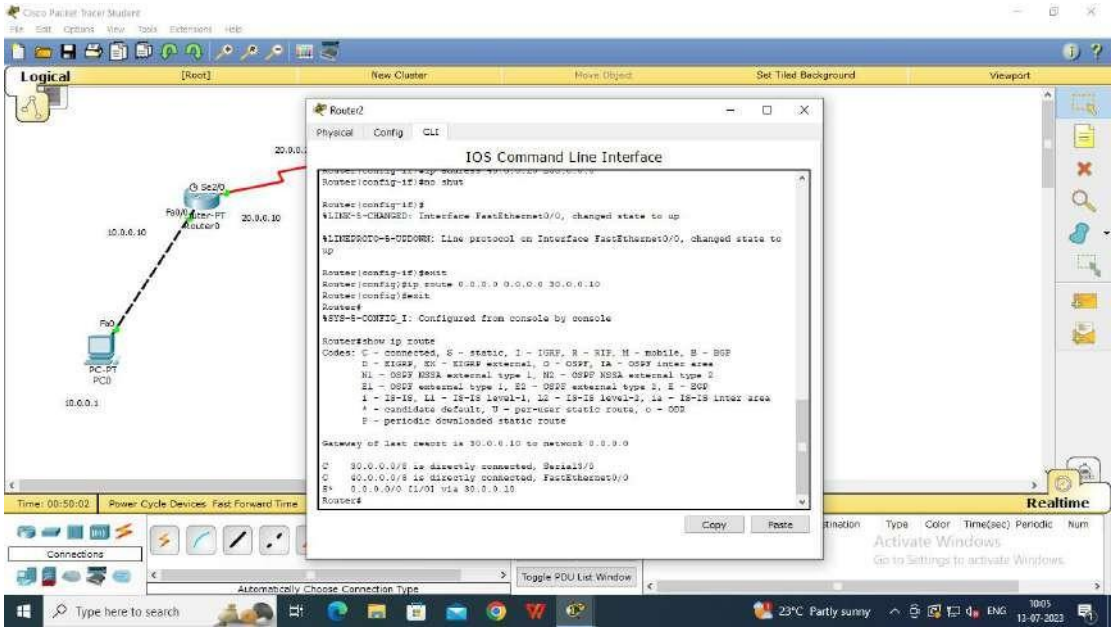
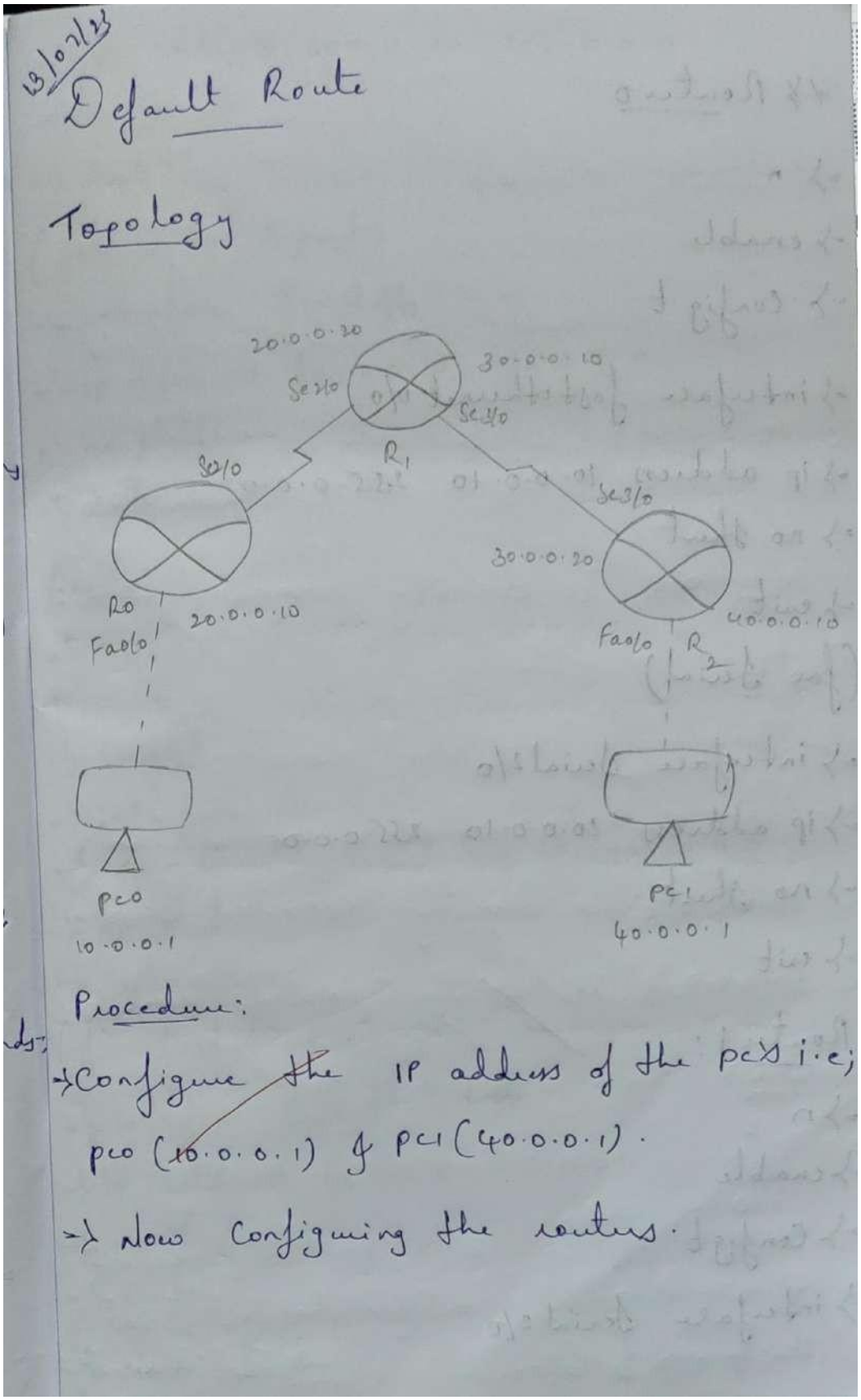


Fig 4: Router2 Connections

Procedure and Observation:



Router0

-> n
-> enable
-> config t
-> interface fastethernet 0/0
-> ip address 10.0.0.10 255.0.0.0
-> no shut
-> exit
(for Serial)
-> interface Serial 2/0
-> ip address 20.0.0.10 255.0.0.0
-> no shut
-> exit

Router1:

-> n
-> enable
-> Config t
-> interface Serial 2/0

-> ip address 20.0.0.20 255.0.0.0
-> no shut
-> exit
(for Serial 3 part)
-> interface Serial 3/0
-> ip address 30.0.0.10 255.0.0.0
-> no shut
-> exit

Router 2:

-> n
-> enable
-> config t
-> interface Serial 3/0
-> no ip address 30.0.0.20 255.0.0.0
-> no shut
-> exit
(for fastethernet)
-> interface fastethernet 0/0
-> ip address 40.0.0.10 255.0.0.0
-> no shut
-> exit.
-> Now check the connection & see if all

-> them are connected.

Router 0

-> exit

-> show ip route

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

C 20.0.0.0/8 is directly connected, Serial 2/0

Router 1

-> exit

-> show ip route

C 20.0.0.0/8 is directly connected, Serial 2/0

C 30.0.0.0/8 is directly connected, Serial 3/0

Router 2

-> exit

-> show ip route

C 30.0.0.0/8 is directly connected, Serial 3/0

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

=> Now configuring the default route to all the routers connected or present, so that the communication can happen by/through all the routers & networks present.

Router 0

-> config t

=> ip route 0.0.0.0 0.0.0.0 20.0.0.20

-> exit

Router 1

->config t

-> ip route 10.0.0.0 255.0.0.0 20.0.0.10

=> ip route 40.0.0.0 255.0.0.0 30.0.0.20

=> exit

Router 2

-> Configt

=> ip route 0.0.0.0 0.0.0.0 30.0.0.10

-> exit.

=> Now again check if connections the default route is proper.

Router 0

-> show ip route

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

C 20.0.0.0/8 is directly connected Serial 2/0

S* 0.0.0.0/0 [1/0] Via 20.0.0.20

Router 1

-> show ip route

S 10.0.0.0/8 [1/0] Via 20.0.0.10

C 20.0.0.0/8 is directly connected, Serial 2/0

C 30.0.0.0/8 is directly connected, Serial 3/0

S 40.0.0.0/8 [1/0] Via 30.0.0.20

Router 2

-> show ip route

C 30.0.0.0/8 is directly connected, Serial 3/0

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

S* 0.0.0.0/0 [1/0] Via 30.0.0.10

Output

=> Now passing ping message from PC0 to PC1.

M -> ping 40.0.0.1

Pinging 40.0.0.1 with 32 byts of data:

Request timed out

Reply from 40.0.0.1: byts = 32 time = 15ms
TTL = 125

Reply from 40.0.0.1: byts = 32 times = 2ms
TTL = 125

Reply from 40.0.0.1: byts = 32 times = 2ms
TTL = 125

Ping statistics for 40.0.0.1:

Packets: Sent = 4, Received = 3, Lost = 1
(25% loss)

Approximate round trip times in milli-seconds:

Minimum = 2ms, Maximum = 15ms, Average = 6ms

Now again Sending the ping message from pc0 to pc1.

=> Ping 40.0.0.1

Pinging 40.0.0.1 with 32 bytes of data:

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

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

Reply from 40.0.0.1: bytes = 32 time = 2ms
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 = 2ms,

Average = 2ms.

=> Now pinging from PC1 to PC0.

-> Ping 10.0.0.1

Pinging from 10.0.0.1: bytes=32 time=3ms TTL=125

Pinging 10.0.0.1 with 32 bytes of data:

Reply from 10.0.0.1: bytes=32 time=3ms TTL=125

Reply from 10.0.0.1: bytes=32 time=9ms TTL=125

Reply from 10.0.0.1: bytes=32 time=3ms TTL=125

Reply from 10.0.0.1: bytes=32 time=3ms TTL=125

Ping statistics for 10.0.0.1:

Packets: Sent=4, Received=4, Lost=0(0%lo

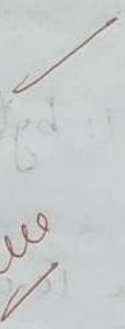
Approximate round trip times in milli-seco-nds:

Minimum = 3ms, Maximum = 9ms, Average = 4ms

observations

-> Configuring default ip route ensures that the packet passes through the default route when no other route is available for an ip destination address

-> The simulation of sending a simple PDU from source to destination (Here from pc0 to pc1) shows the route taken by the ICMP protocol.



Output:

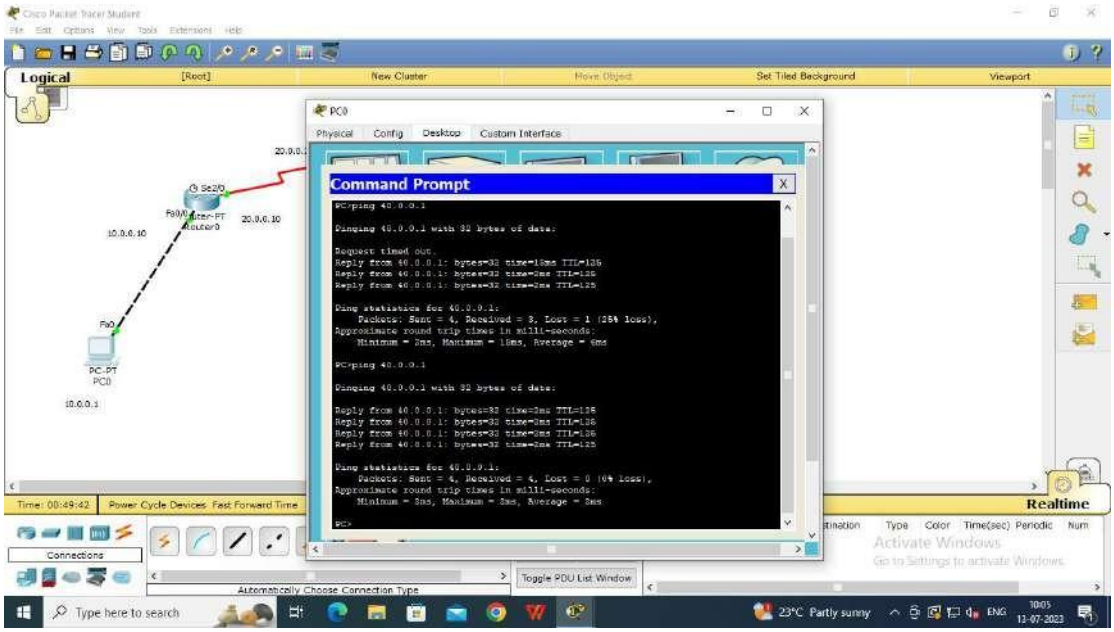


Fig 5: Pinging from pc0 to pc1

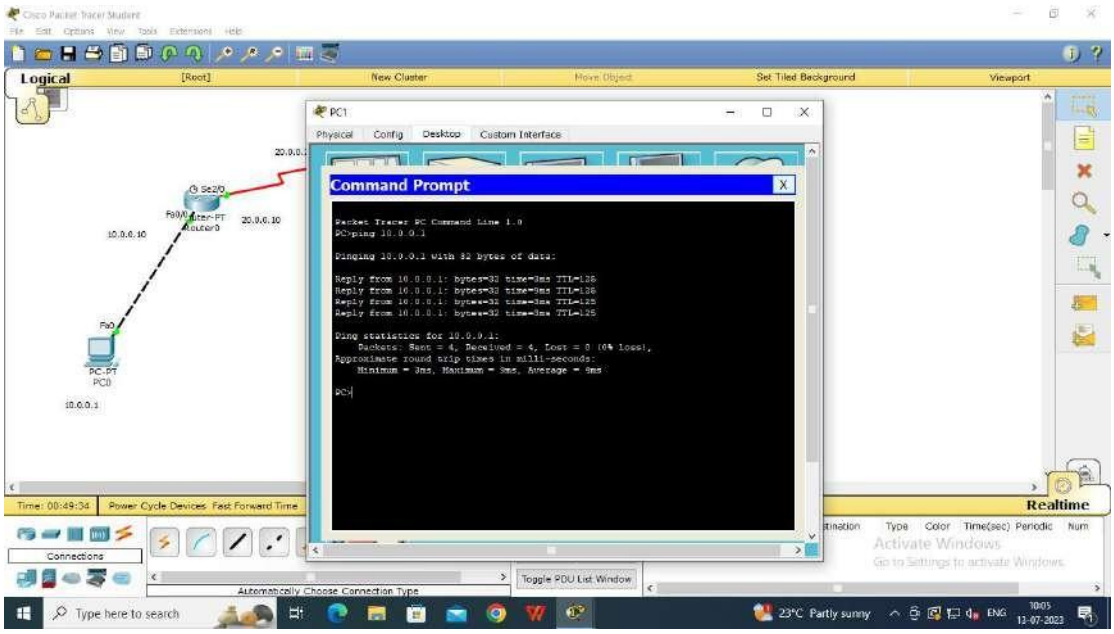


Fig 6: Pinging from pc0 to pc1

Experiment No. 4

Cycle 1

Aim-4

4. Configure DHCP within a LAN and outside LAN.

Topology:

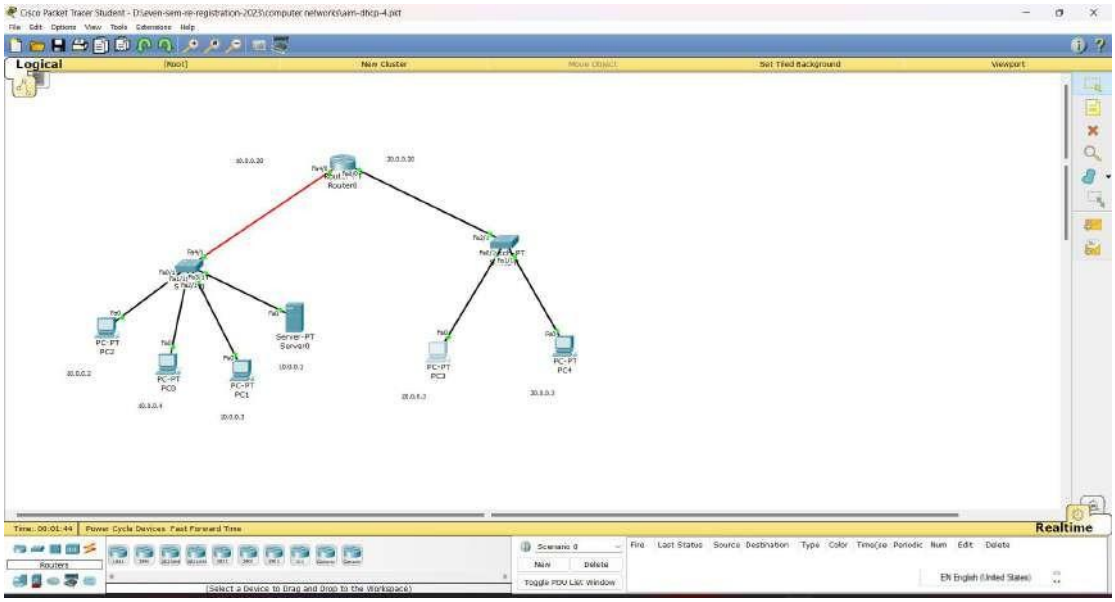


Fig 1: Topology

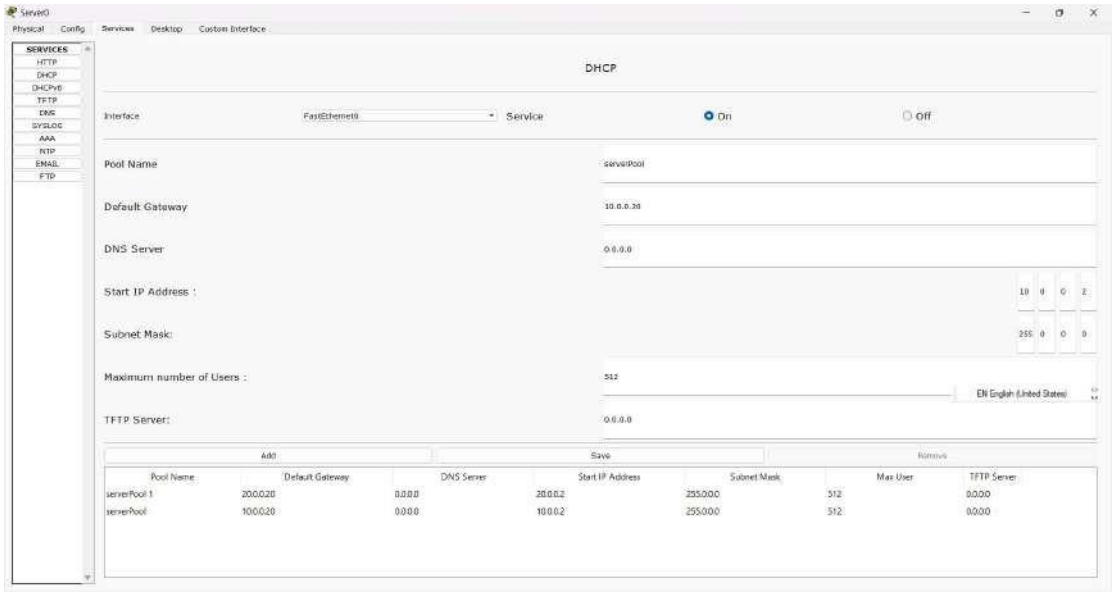


Fig 2: Configuring DHCP Settings in Server

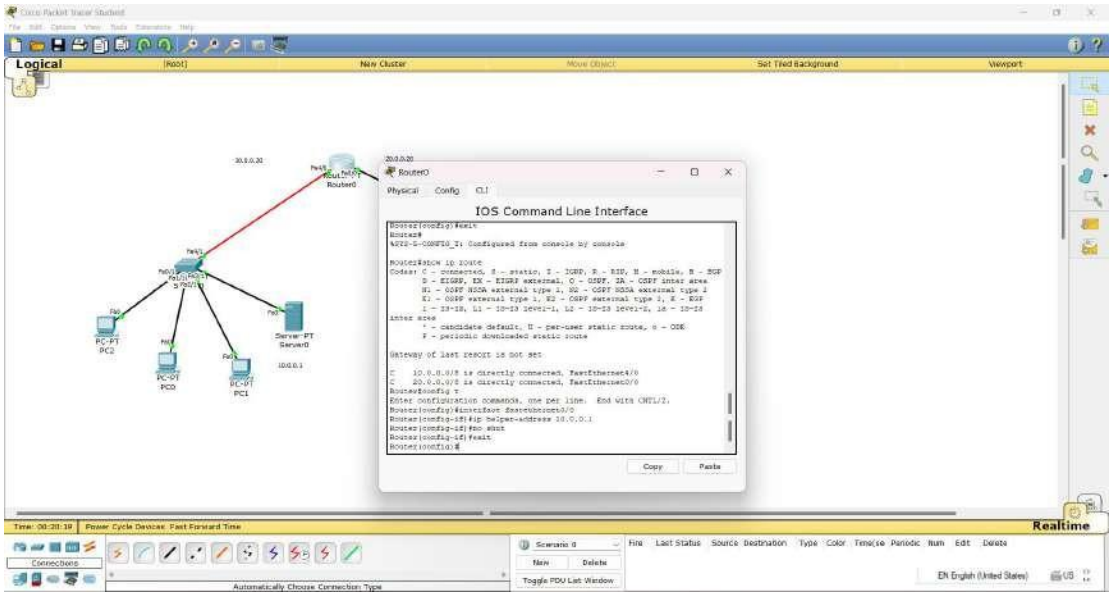


Fig 3: Router Configuring and Connections

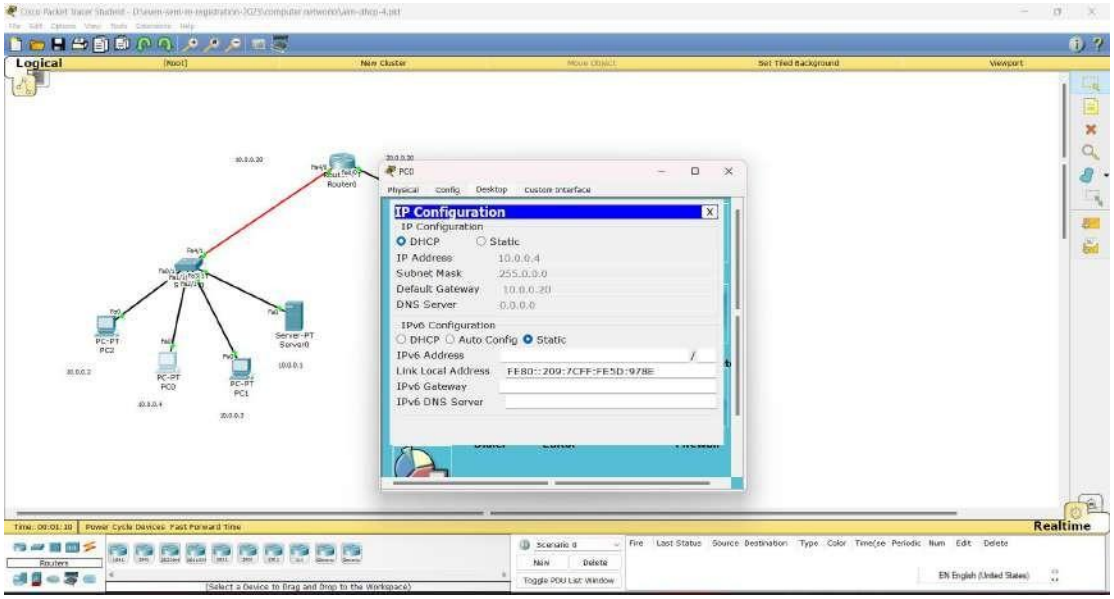


Fig 4: Pc0 Configurations

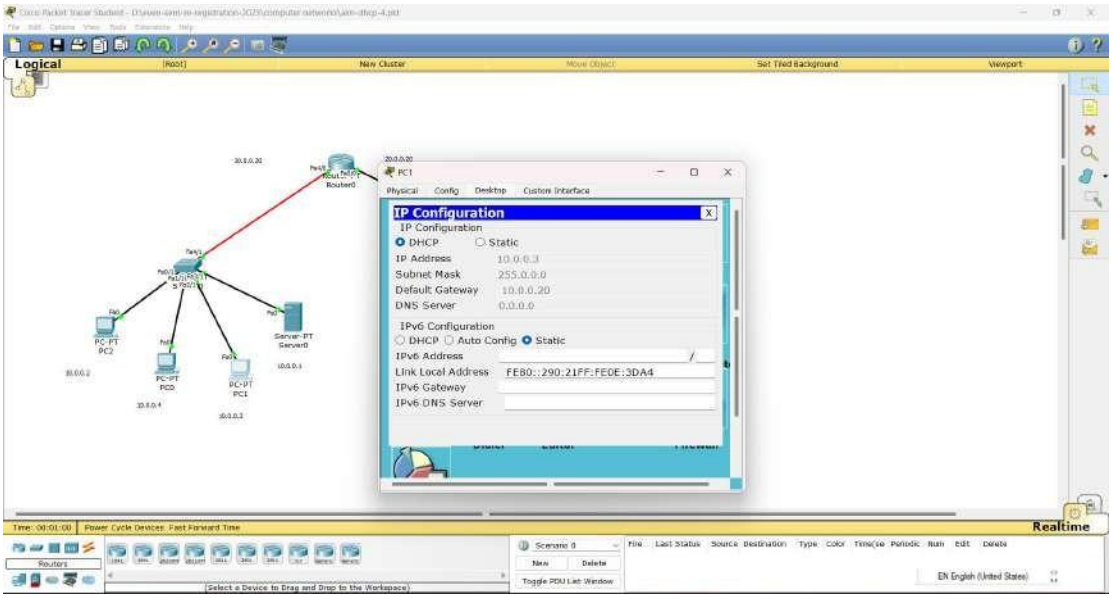


Fig 5: Pc1 Configurations

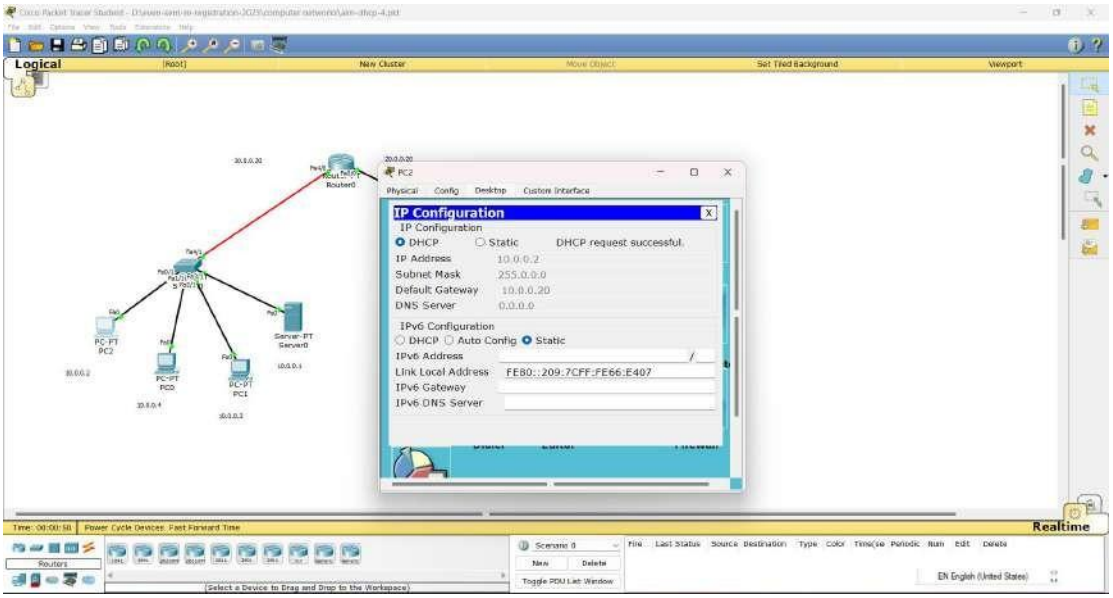


Fig 6: Pc2 Configurations

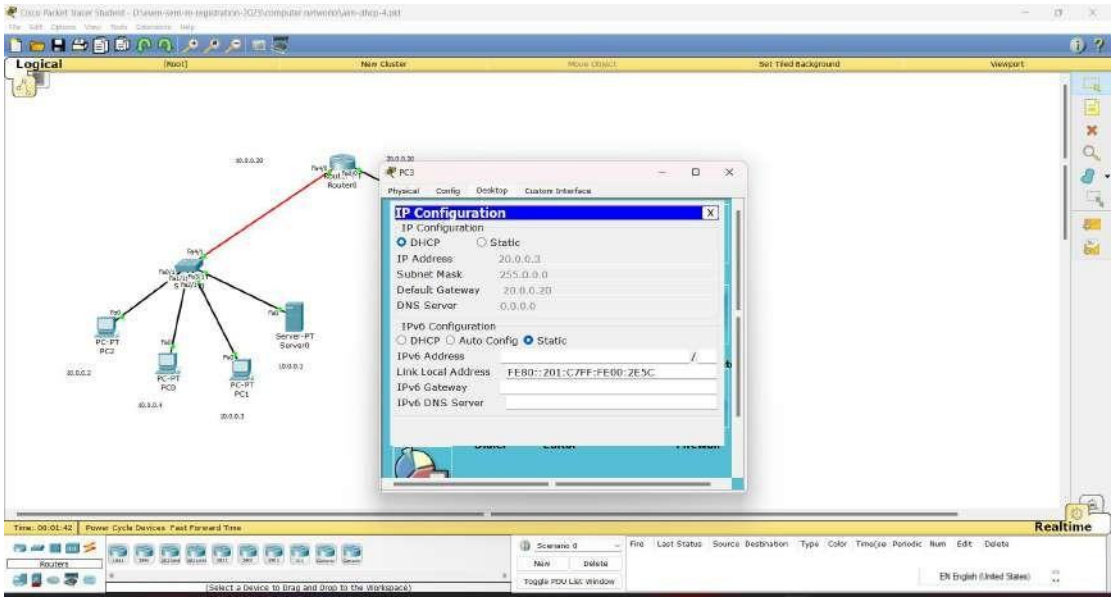


Fig 7: Pc3 Configurations

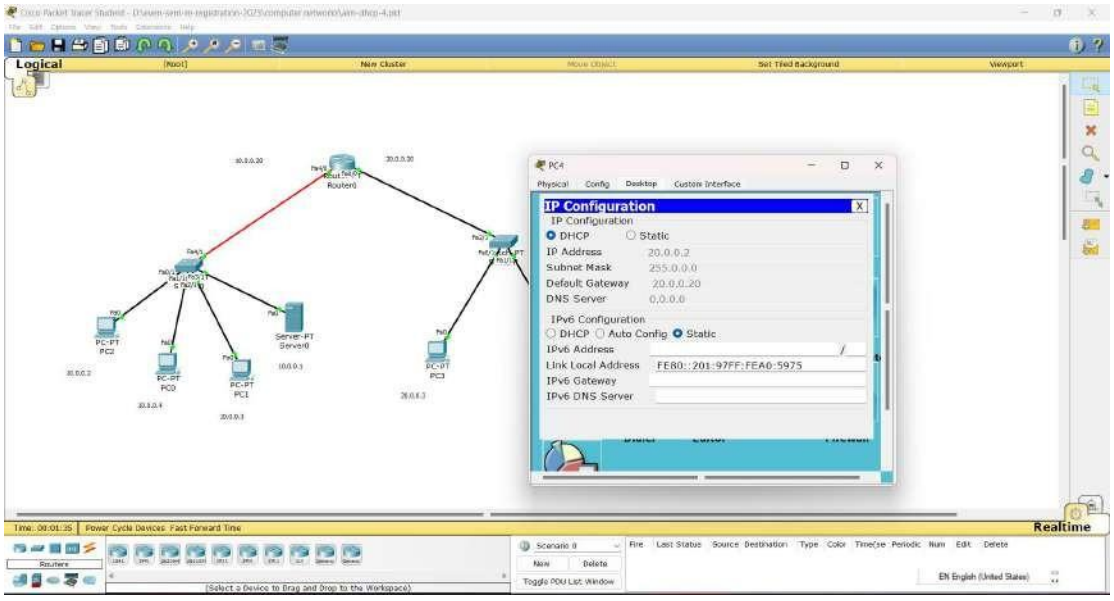


Fig 8: Pc4 Configurations

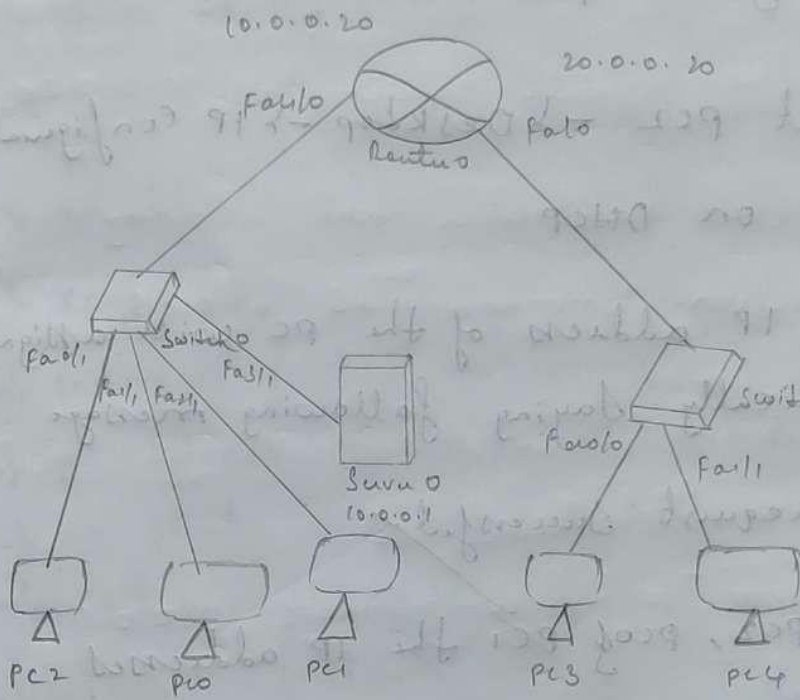
Procedure and Observations:

Aim-4

18/07/23

Configure DHCP within a LAN and outside LAN.

Topology



Procedure

=> Server Configuration

-> Server -> Services -> DHCP

-> Service "on" (toggle from off to on)

-> Set the Start IP address : 10.0.0.2 and Subnet mask as 255.0.0.0 then click on Save.

-> In Config tab of Server -> FastEthernet0

-> Set the default gateway as

-> Set the IP address as 10.0.0.1.

-> Now go to pcs.

-> Select PC2 -> Desktop -> IP Configuration

-> click on DHCP

-> the IP address of the PC will assigned automatically saying following message

DHCP request Successful

-> for PC2, PC0 & PC1 the IP addresses are 10.0.0.2, 10.0.0.4, 10.0.0.3 respectively along with Subnet mask as 255.0.0.0.

-> Like how 3 pcs of a Server is connected to a Switch, like wise 2 pcs can has

-> to be connected to a one more switch (Basically lan). (Say switch1 pc3 & pc4)

-> And switch0 & switch1 has to be connected to a router.

-> Now configuring the router

-> n

-> enable

-> config t

-> interface fastethernet 4/0

-> ip address 10.0.0.20 255.0.0.0

-> no shut

-> exit

-> interface fastethernet 0/0

-> ip address 20.0.0.20 255.0.0.0

-> no shut

-> exit

-> Now check if connections are proper.

-> show ip route

C 10.0.0.0/8 is directly connected,
FastEthernet 4/0

C 20.0.0.0/8 is directly connected
FastEthernet 0/0

=> Now click on Server -> Config tab

=> Set the default gateway as 10.0.0.20.

=> Now click on Router and following Configuration commands are as follows

=> Config t
=> interface fastethernet 0/0
=> ip address ip helper-address 10.0.0.1
=> no shut
=> exit.

=> Then go to Server click on Services tap.

=> In DHCP change the pool Name as Serverpool.

-> then set the start IP address as 20.0.0.2 and click on add and save.

-> Now click on pc4 which are connected to switch1.

-> pc -> desktop -> IP configuration -> DHCP.

-> the IP addresses & subnet mask will assigned automatically.

-> the IP address for pc3 & pc4 are 20.0.0.3 & 20.0.0.4 respectively and subnet mask as 255.0.0.0

=> In DHCP(server) for server pool seat default gateway as 10.0.0.20 & for server pool1 also same 20.0.0.29 and click on save each time the default gateway as been set.

observation

-> Dynamic Host Configuration Protocol (DHCP)

-> It works on DORA application

=> DORA is D(Discover), O(offer), R(Request) & A(Acknowledgement)

=> Basically the client broadcasts for network

-> Server offers the client

=> the client requests the Server

-> and Server sends the acknowledgement to clients and assigns the IP as provided the IP address to the client which has requested.

Output :

-> Passing Ping message from PC4 to PC1

Ping 10.0.0.2
Pinging 10.0.0.2 with 32 bytes of data:
Request timed out

Reply from 10.0.0.2 : bytes = 32 time = 0ms
TTL = 127

Reply from 10.0.0.2 : bytes = 32 time = 3 ms
TTL = 127

Reply from 10.0.0.2 : bytes = 32 time = 0ms
TTL = 127

Ping Statistics for 10.0.0.2:

Packets: Sent = 4, Received = 3, Lost = 1 (25% loss)

Approximate round trip times in milli-seconds:

Minimum = 0 ms, Maximum = 3 ms,

Average = 1 ms.

=> Passing Ping message from PC1 to PC3

Ping 20.0.0.3

Pinging 20.0.0.3 with 32 bytes of data:

Reply from 20.0.0.3: bytes=32 time=0 ms
TTL=127

Reply from 20.0.0.3: bytes=32 time=0 ms
TTL=127

Reply from 20.0.0.3: bytes=32 time=1 ms
TTL=127

Reply from 20.0.0.3: bytes=32 time=0 ms
TTL=127

Ping statistics for 10.0.0.3:

Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),

Approximate round trip time in milli-seconds:

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

18/7/23

Output:

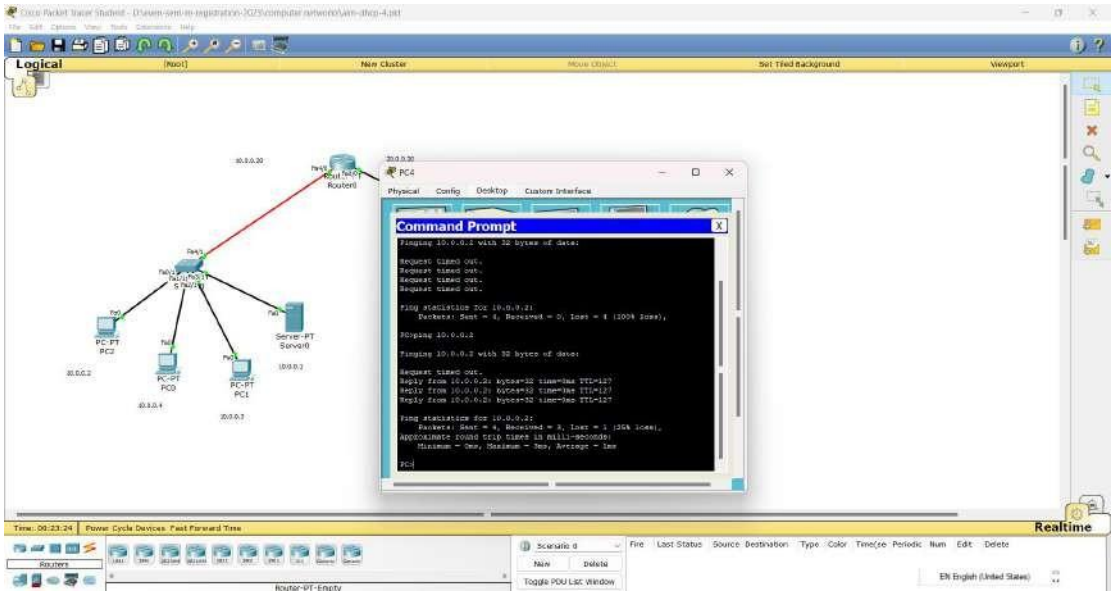


Fig 9: Pinging from pc4 to pc2

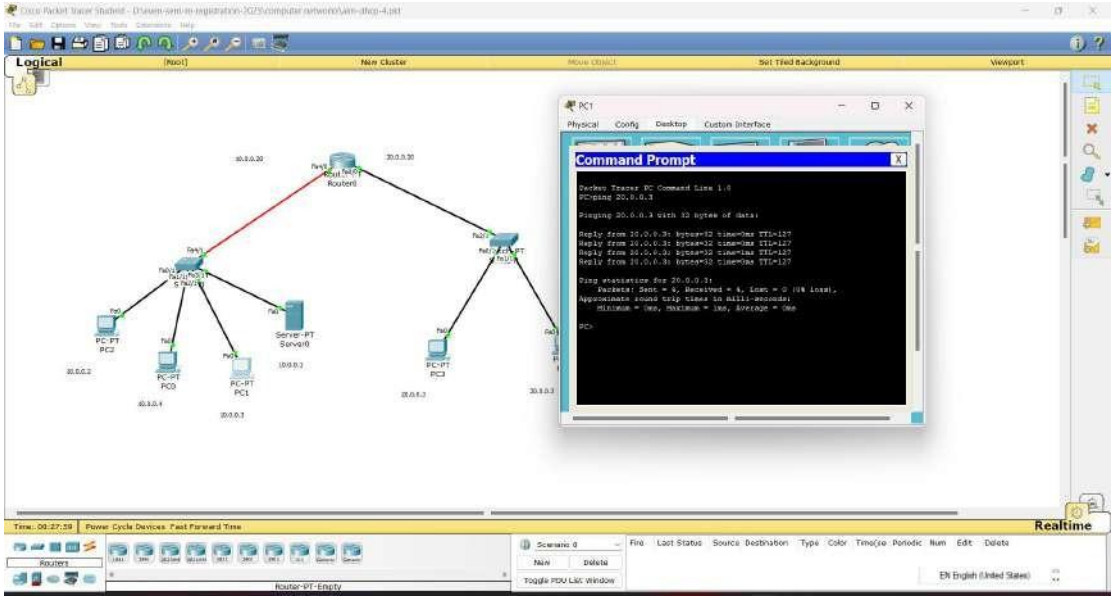


Fig 10: Pinging from pc1 to pc3

Experiment No. 5

Cycle 1

Aim-5

5. Configure RIP routing protocol in Routers

Topology:

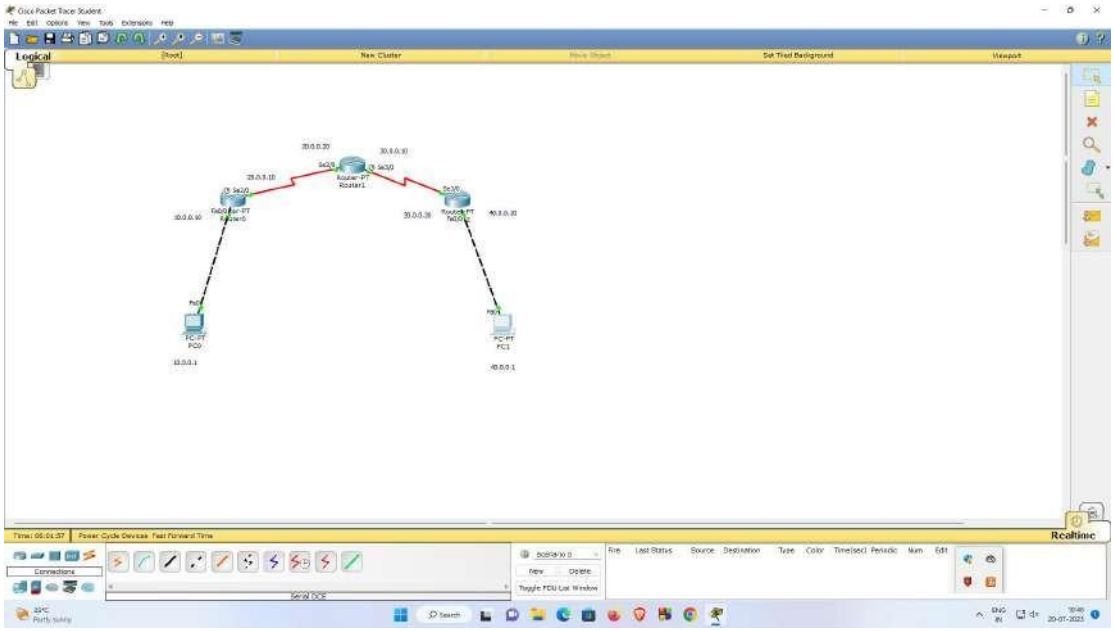


Fig 1: Topology

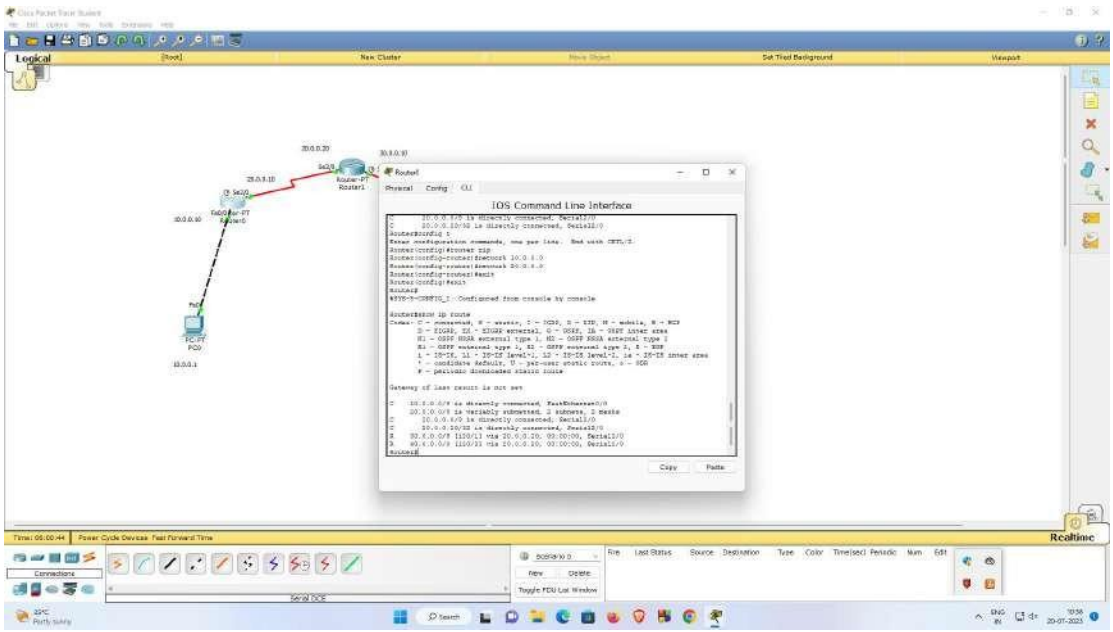


Fig 2: Router 0 networks and next hop ip addresses

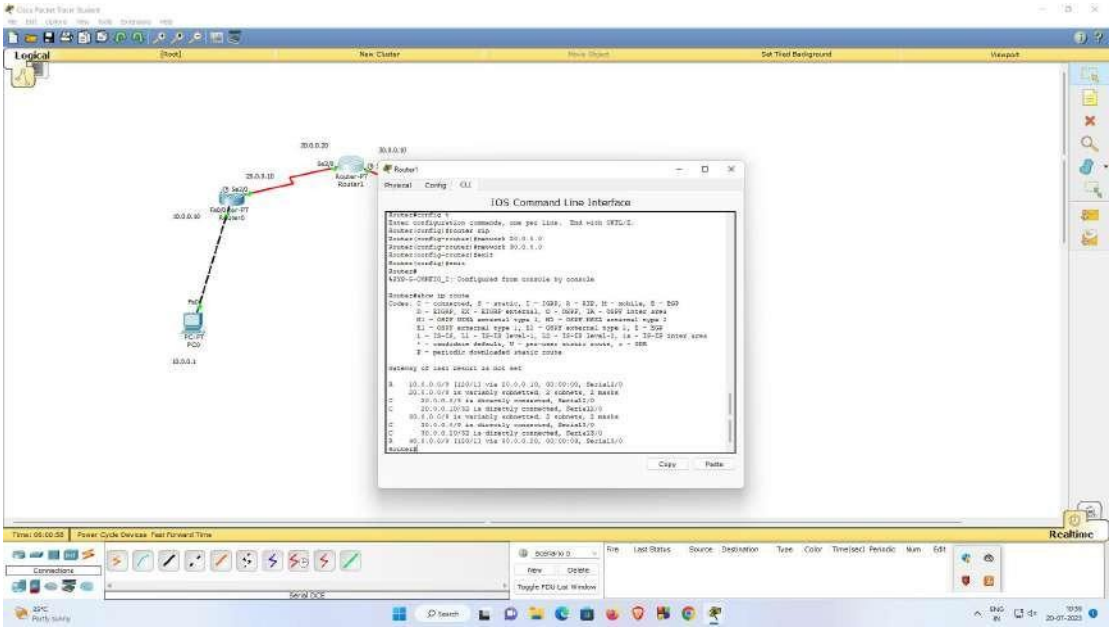


Fig 3: Router 1 networks and next hop ip addresses

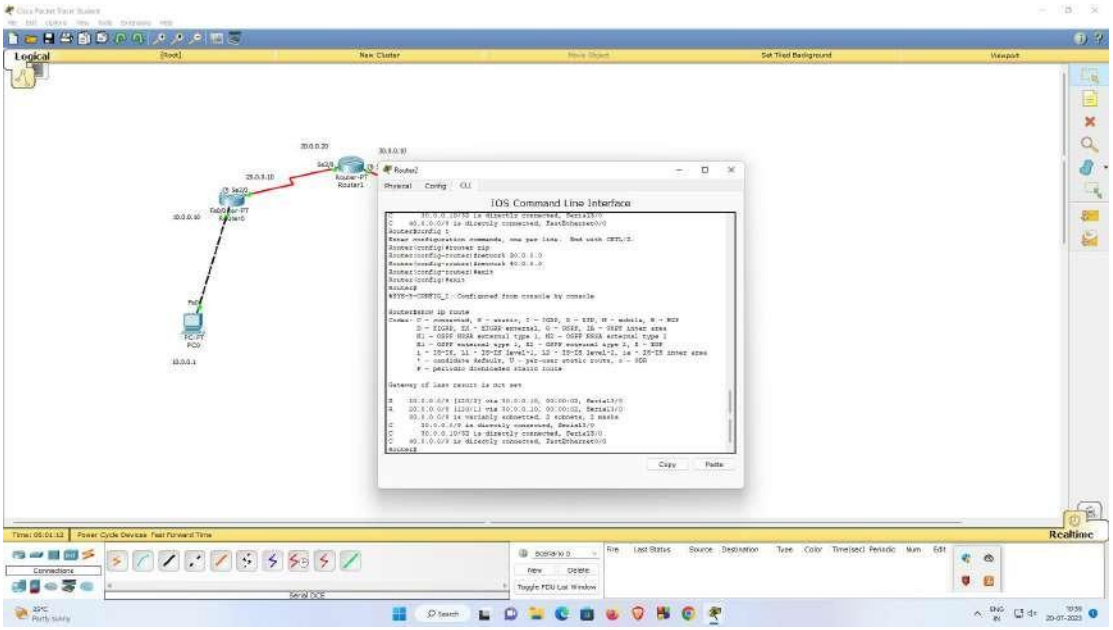


Fig 4: Router 2 networks and next hop ip addresses

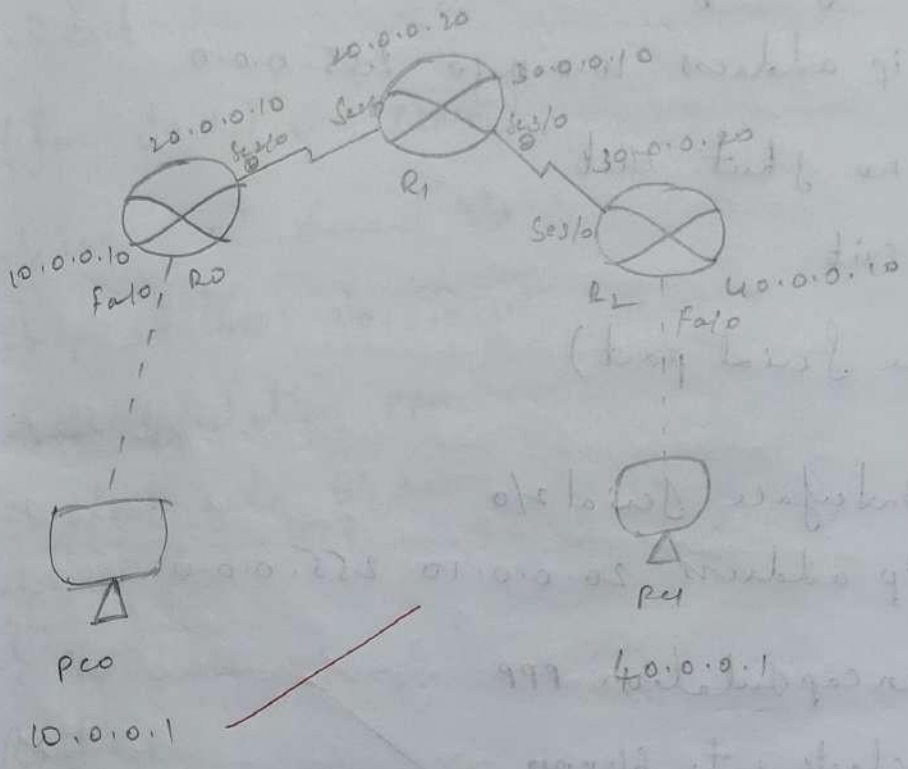
Procedure and Observation:

20/07/23

Aim-5

Configure RIP routing Protocol in Routers.

Topology



Procedure

=> Configure the PC's & PC1's IP address

i.e; PC0 (10.0.0.1) & PC1 (40.0.0.1)

=> Now configure the routers

Router 0

-> n

-> enable

-> config t

-> interface fastethernet 0/0

-> ip address 10.0.0.10 255.0.0.0

-> no shut exit

-> exit

(for Serial port)

-> interface Serial 2/0

-> ip address 20.0.0.10 255.0.0.0

-> encapsulation PPP

-> clock rate 64000

-> no shut

-> exit

Router 1

-> n

-> enable

-> config t

-> interface Serial 2/0
-> ip address 20.0.0.20 255.0.0.0
-> encapsulation PPP
-> no shut
-> exit

(for Serial 3/0 point)

-> interface Serial 3/0
-> ip address 30.0.0.10 255.0.0.0
-> encapsulation PPP
-> clock rate 64000
-> no shut
-> exit

Router 2

-> n
-> enable
-> configt
-> interface Serial 3/0
-> ip address 30.0.0.20 255.0.0.0
-> encapsulation PPP
-> no shut -> exit

(for fastethernet port)

=> interface fastethernet 0/0

-> ip address 40.0.0.10 255.0.0.0

-> no shut

-> exit

Note:

=> As here it is rip routing for all serial interfaces we use/include this extra command i.e; "encapsulation ppp".

-> And for serial interfaces ~~with~~ only which has clock symbol ~~use~~ use/include this command i.e; "clock rate 64000" (after encapsulation ppp command).

=> Now check the connected networks (the help)

Router 0

=> exit

=> show ip route

=> Now we've to make routers help to themselves know the about the networks present. For that we need to use following commands

Router

=> config t

=> router rip

=> network 10.0.0.0

=> network 20.0.0.0

=> exit

Router 1

=> Config t

=> router rip

-> network 20.0.0.0

-> network 30.0.0.0

-> exit

Router 2

-> Config t

-> router rip

=> network 30.0.0.0

-> network 40.0.0.0

-> exit

=> Now check the connected networks and the next hops of all the routers

Router 0

-> exit

-> show ip route

C 10.0.0.0/8 is directly connected FastEthernet 0/0

C 20.0.0.0/8 is directly connected Serial 2/0

20.0.0.0/8 is Variably Subnetted, 2 Subn-
-ets, 2 masks

C 20.0.0.20/32 is directly connected
Serial 2/0

R 30.0.0.0/8 [120/1] via 20.0.0.20, 00:00:00,
Serial 2/0

R 40.0.0.0/8 [120/1] via 20.0.0.20, 00:00:00,
Serial 2/0

Router1

=> exit

=> show ip route

R 10.0.0.0/8 [120/1] via 20.0.0.10, 00:00:00,
Serial 2/0

20.0.0.0/8 is directly connected Variably
Subnetted, 2 subnets, 2 masks

C 20.0.0.0/8 is directly connected, Serial2/0

C 20.0.0.10/32 is directly connected, Serial2/0

30.0.0.0/8 is Variably Subnetted, 2 Subn-
-ets, 2 masks

C 30.0.0.0/8 is directly connected,
Serial 3/0

C 30.0.0.20/32 is directly connected,
Serial 3/0

R 40.0.0.0/8 [120/1] via 30.0.0.20, 00:00:03
Serial 3/0.

Router 2

=> exit

=> show ip route

R 10.0.0.0/8 [120/2] via 30.0.0.10, 00:00:03
Serial 3/0

R 20.0.0.0/8 [120/1] via 30.0.0.10, 00:00:02
Serial 3/0

30.0.0.0/8 is variably subnetted, 2 subnets
2 masks

C 30.0.0.0/8 is directly connected, Serial 3/0

C 30.0.0.10/32 is directly connected, Serial 3/0

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

-> Set the default gateways of pcs.

i.e; pc0 (10.0.0.10) & pc1 (40.0.0.10).

Output

=> Pinging from pc0 to pc1

>ping 40.0.0.1

Pinging 40.0.0.1 with 32 bytes of data:

Request timed out

Reply from 40.0.0.1: bytes = 32 time = 6ms
TTL = 125

Reply from 40.0.0.1: bytes = 32 time = 11 ms
TTL = 125

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

Ping statistics for 40.0.0.1:

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

Approximate round trip times in milli-seconds

Minimum = 5ms, Maximum = 11 ms,
Average = 7 ms

=> Pinging from PC1 to PC0

>Ping 10.0.0.1 :

Pinging 10.0.0.1 with 32 bytes of data :

Reply from 10.0.0.1 : bytes = 32 time = 12ms
TTL = 125

Reply from 10.0.0.1 : bytes = 32 time = 7 ms
TTL = 125

Reply from 10.0.0.1 : bytes = 32 time = 12 ms
TTL = 125

Reply from 10.0.0.1 : bytes = 32 time = 9 ms
TTL = 125

Ping statistics for 10.0.0.1 :
Packets : Sent = 4, Received = 4, Lost = 0
(0% loss)

Approximate round trip times in milli-seconds

Minimum = 7 ms, Maximum = 12 ms,

Average = 10 ms.

Observation

=> RIP is (Routing Information Protocol)

=> It uses distance vector algorithm

=> when you use this particular protocol the routers themselves learn about the other networks present and the next hop to be communicated.

Output:

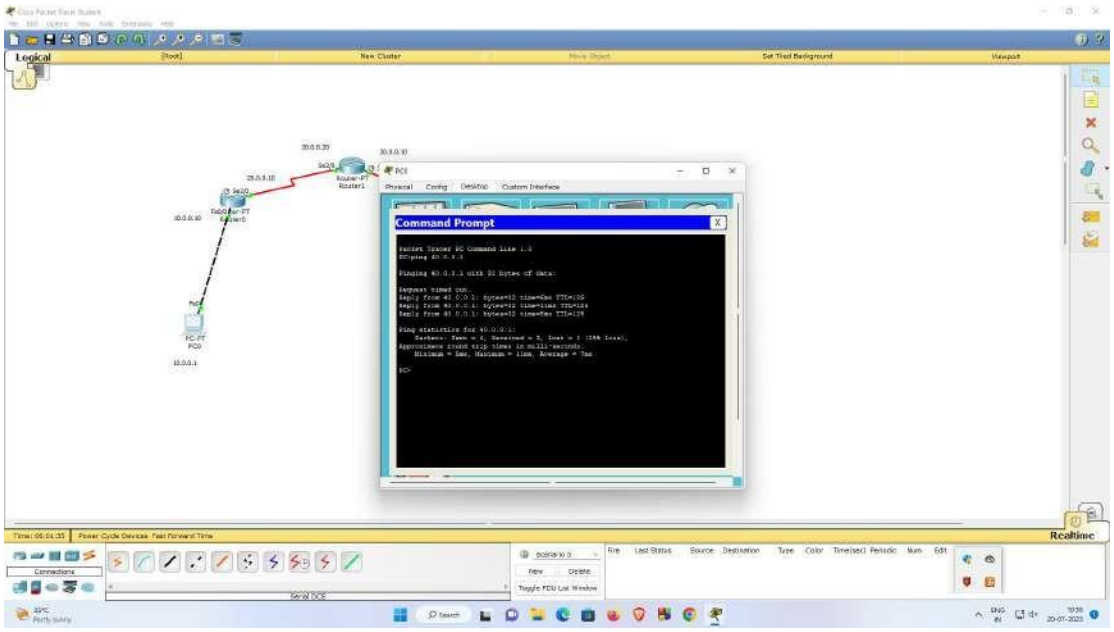


Fig 5: Pinging from pc0 to pc1

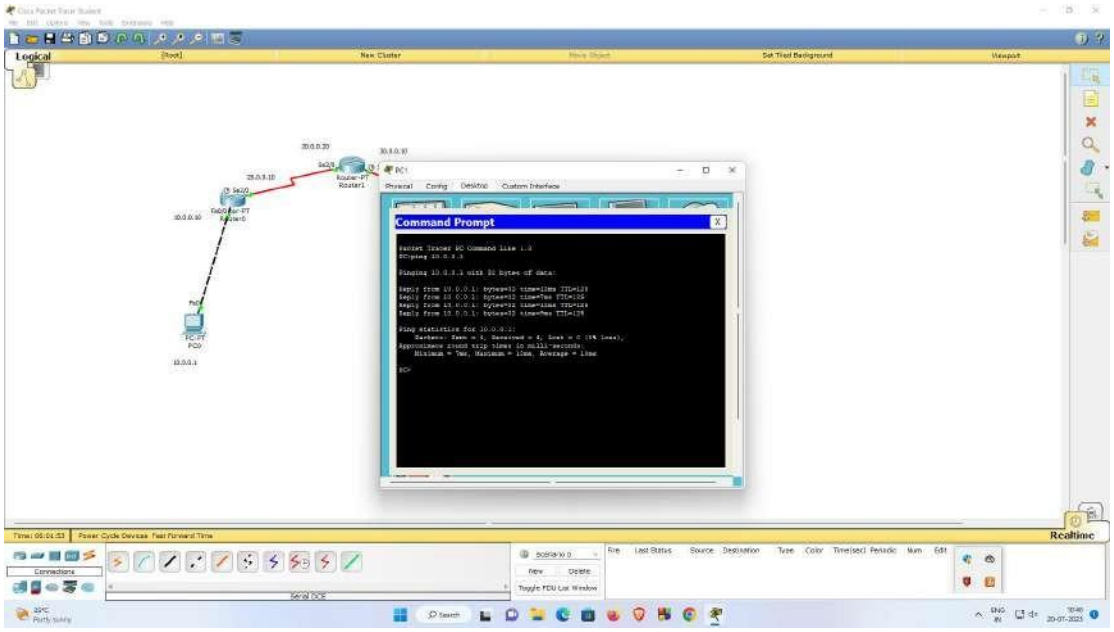


Fig 6: Pinging from pc1 to pc0

Experiment No. 6

Cycle 1

Aim-6

6. Configure Web Server, DNS within a LAN.

Topology:

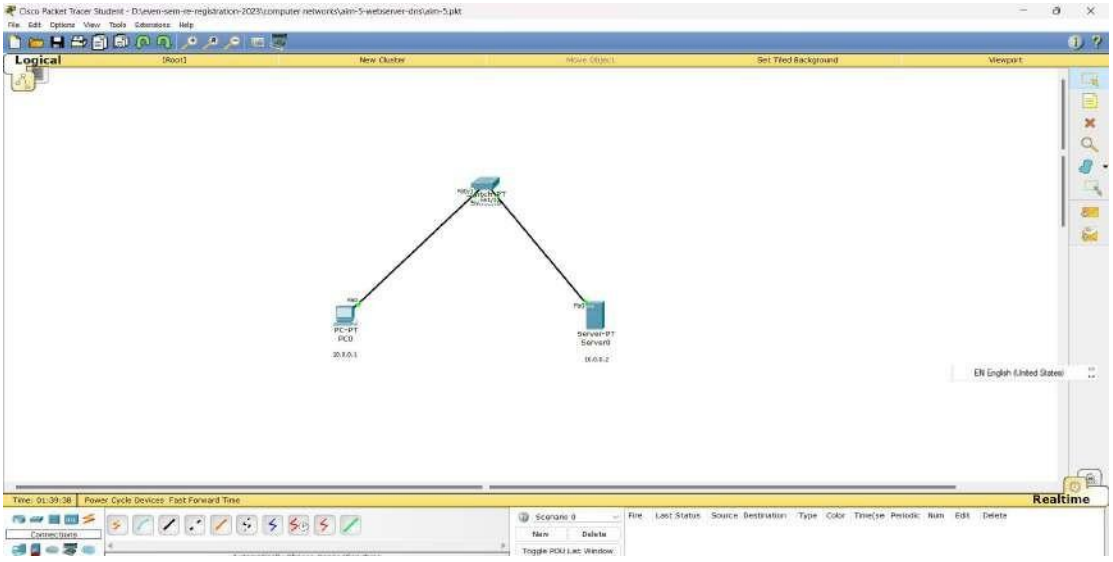


Fig 1: Topology

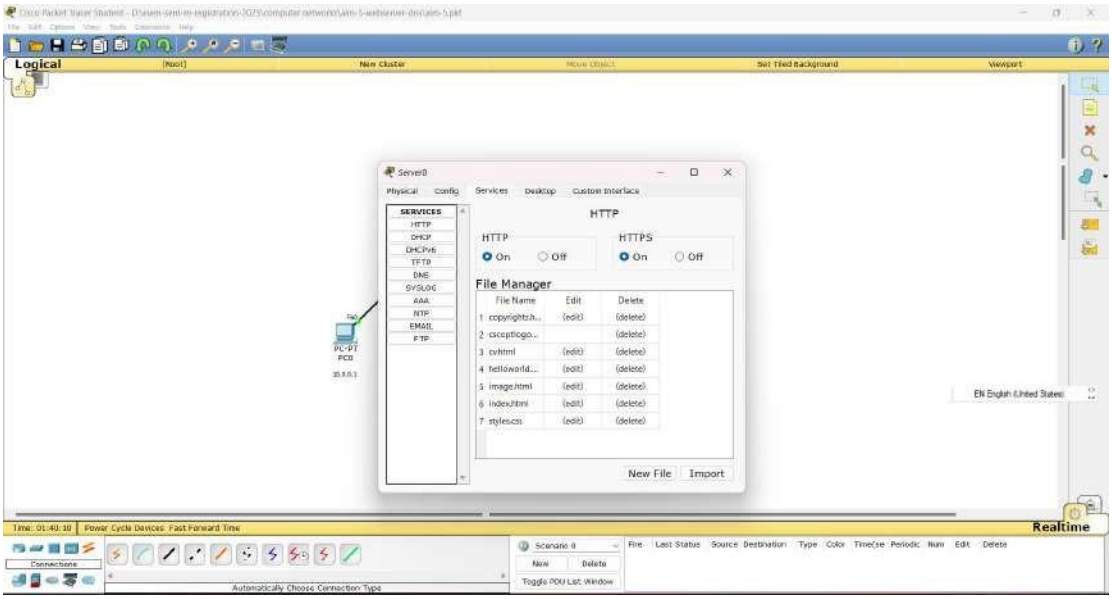


Fig 2: Adding new web page and integrating it into the existing one in Server's HTTP service tab

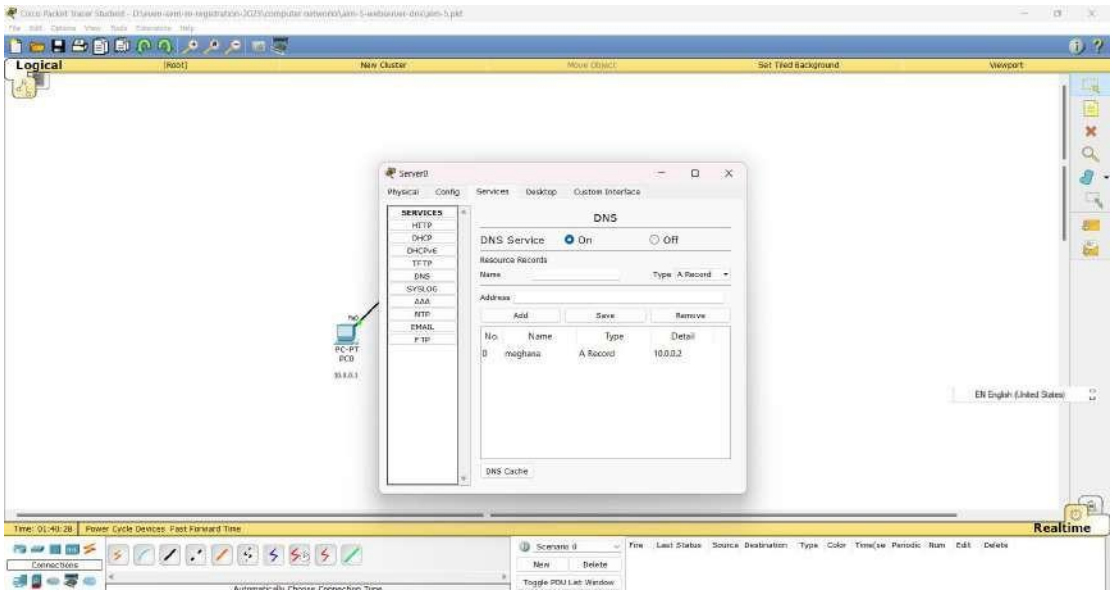
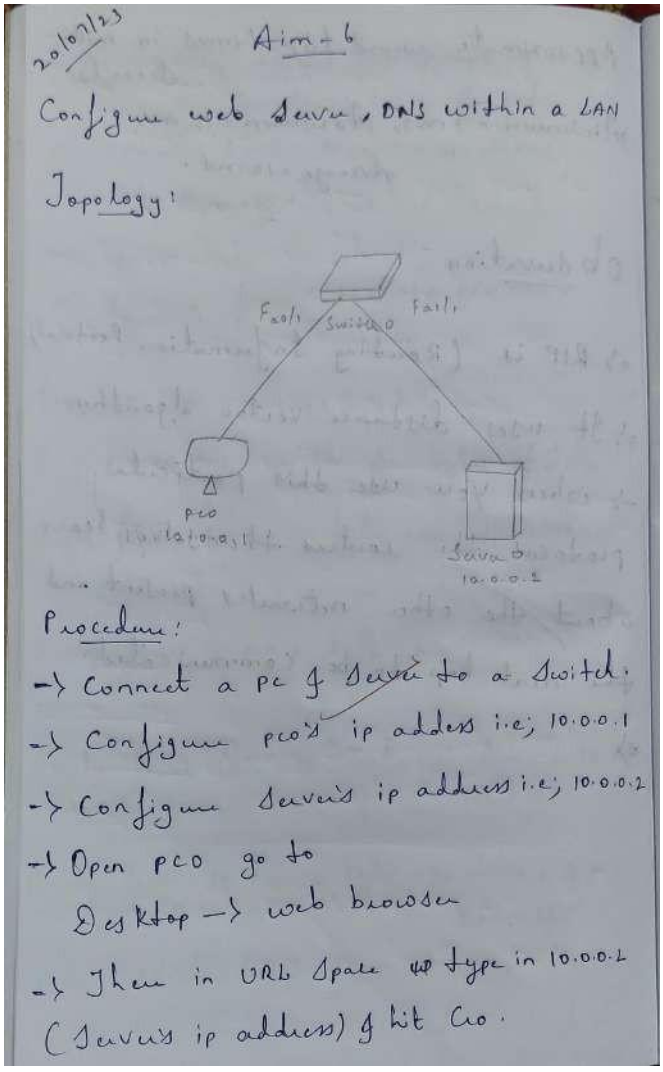


Fig 3: Configuring in Server's DNS service tab

Procedure and Observation:



=> A small web page of cisco packet tracer pops out.

=> Now go to Server -> Services tab -> HTTP

-> Keep HTTP "On"

=> Then in File manager, create a small simple CV (just for practice purpose to check if the web page is working after editing) file (Newfile/input).

=> Now go to "DNS" in Services tab of Server.

-> Keep DNS "On".

-> Name: Meghana (any name can be written).

-> Address: 10.0.0.2 (Server's ip address)

=> Then hit "Add".

=> Now go to pc0

-> pc0 -> web browser -> URL (meghana)

=> instead of Server's ip address mention the name which was done in Server.

=> A small web page pops out with the new addition i.e; "CV.html" file.

=> when you click on it you can see/view newly created CV file.

Output URL http://meghana c:\o stop

Cisco Packet Tracer

Welcome to CPT. Opening doors to new opportunities. Mind wide open.

Quick Links

A small page

Copyrights

Image Page

View my CV (Newly Created web page)

Image

=> Now click on "View my cv (link)" it so will open a small simple basic CV web page.

Meghana

Contact Information
Email: meghana@gmail.com
phone: 1234567890
website: www.linkedin.com/in/meghana

Education
Computer Science and Engineering, BE, BMS College of Engineering, 2023
High School, ABC School, 20XX

Skills and Programming Languages
HTML, CSS, JavaScript, Python, C Language, C++, Java, problem Solving

Observation

=> DNS (Domain name System)
=> It basically works on how map / associate ip address to domain name.
=> It redirects to the default Cisco packet tracer web page after assigning the ip address of the to the Server.

Tue 24/7/23

Output:

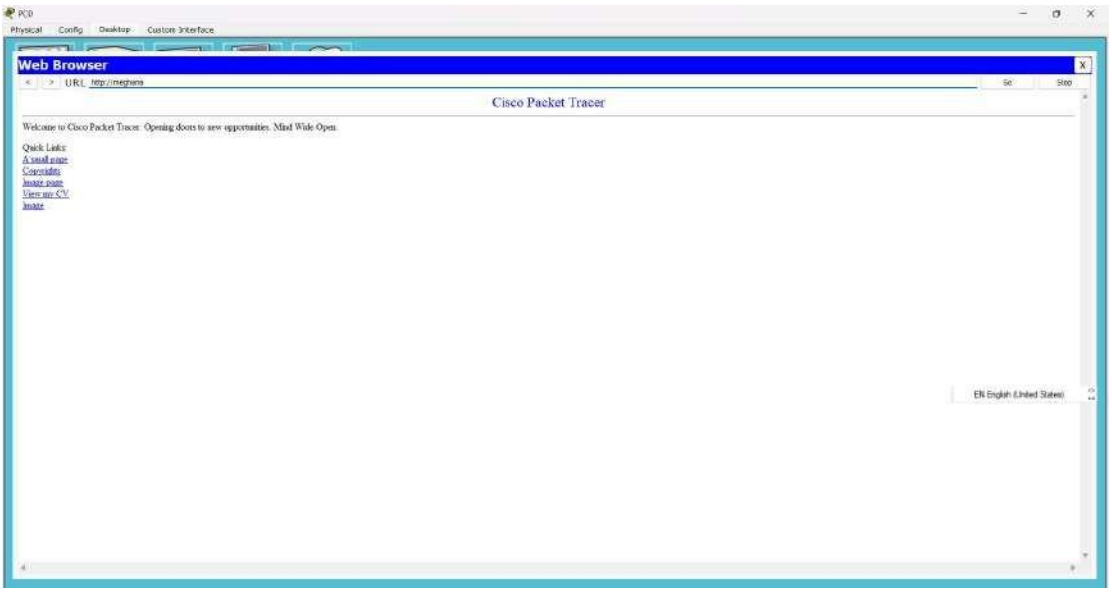


Fig 3: Web Browser of Pco

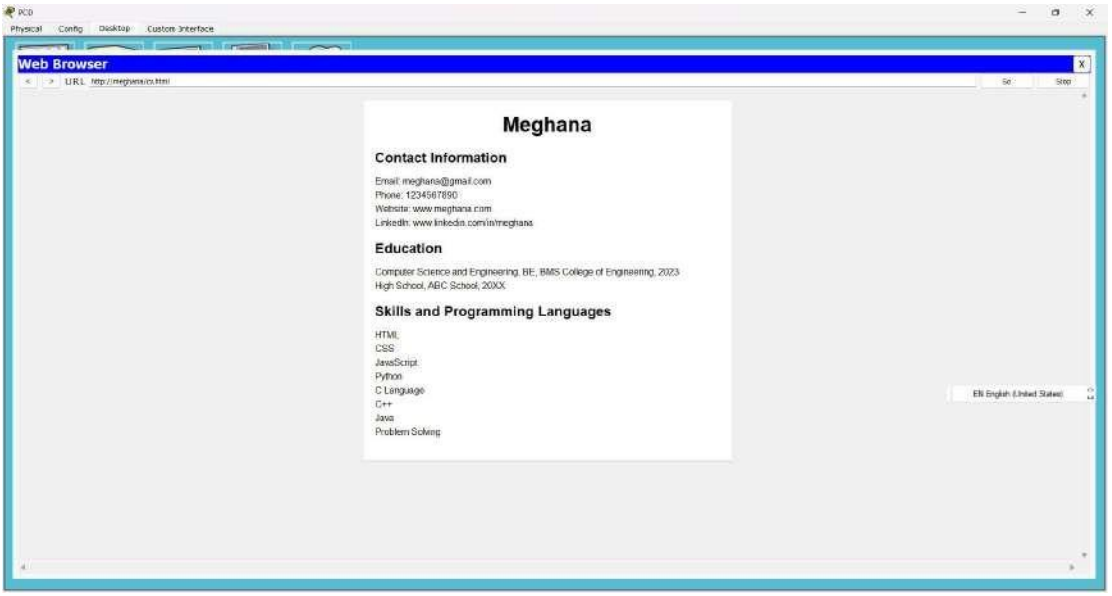


Fig 4: Newly added WebPage is being shown in Pc0's Web Browser

Experiment No. 7

Cycle 1

Aim-7

7. Configure OSPF routing protocol

Topology:

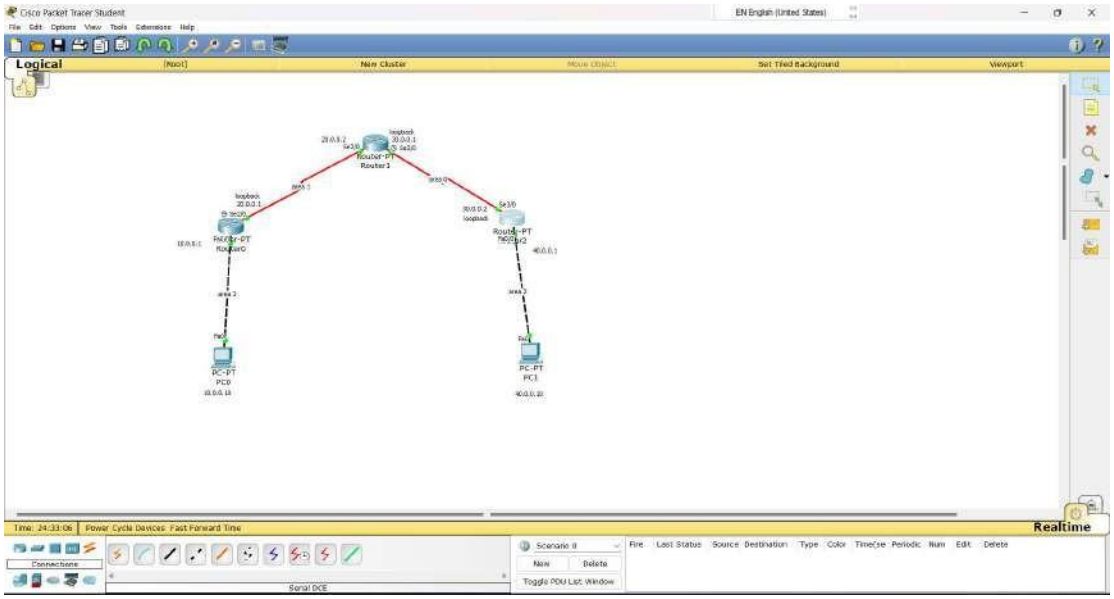


Fig 1: Topology

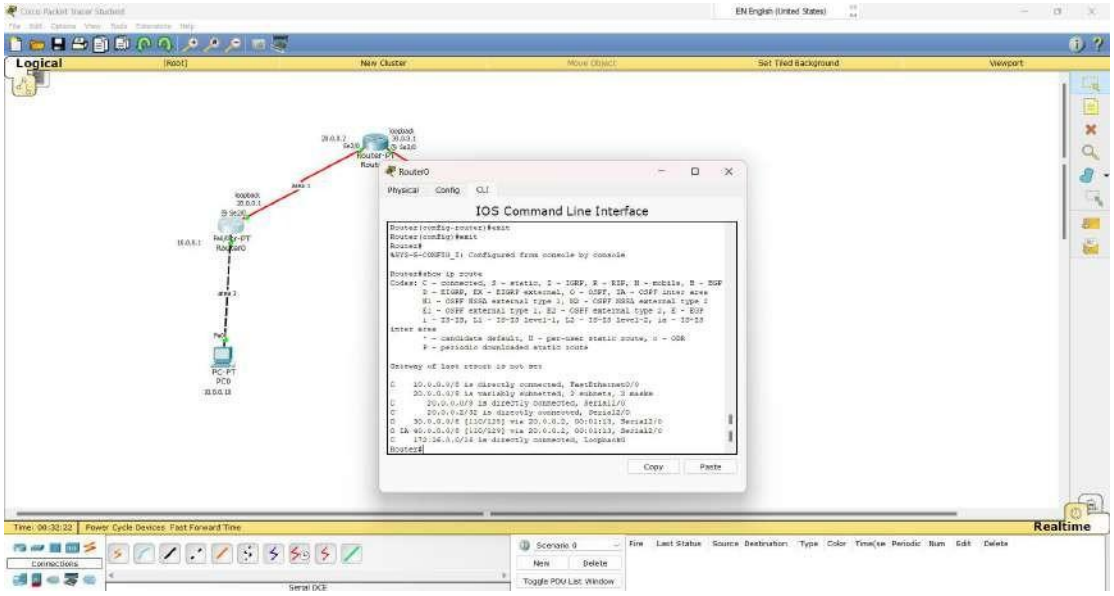


Fig 2: Routing table of Router 0

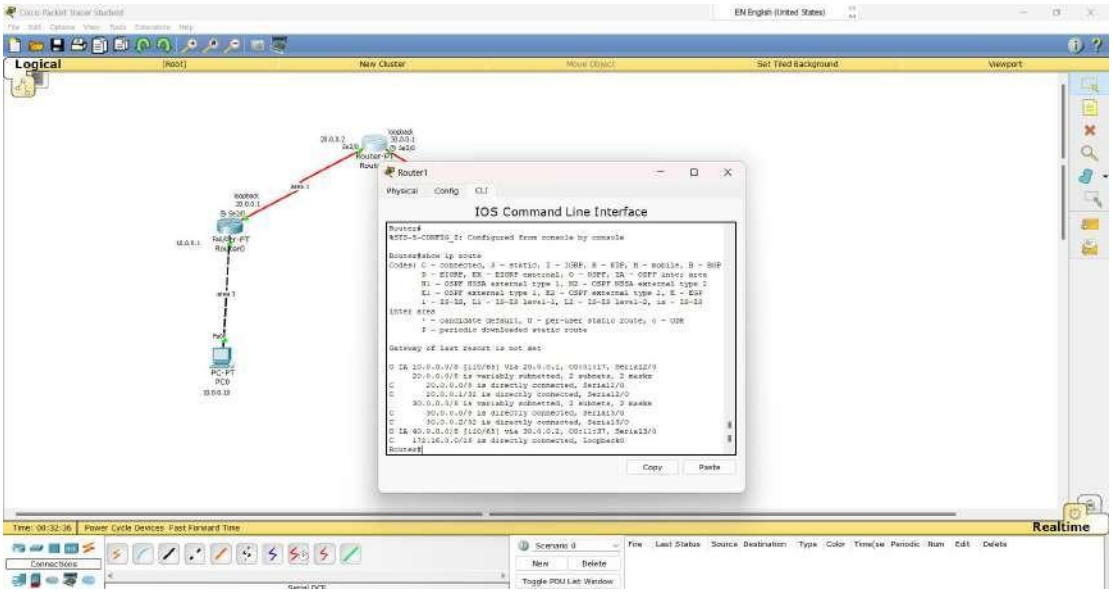


Fig 3: Routing table of Router 1

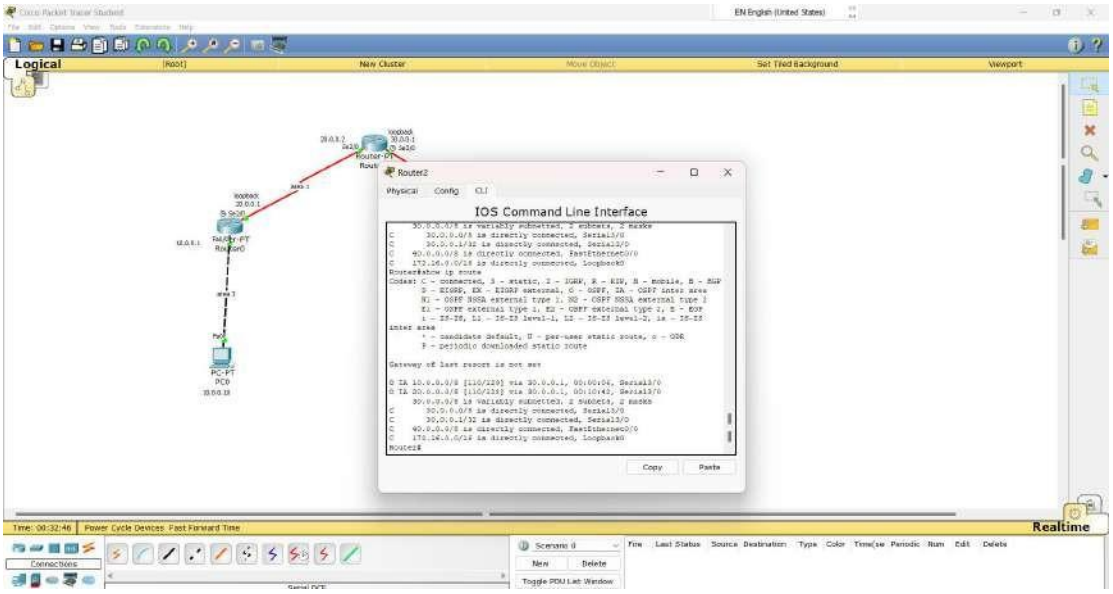


Fig 4: Routing table of Router 2

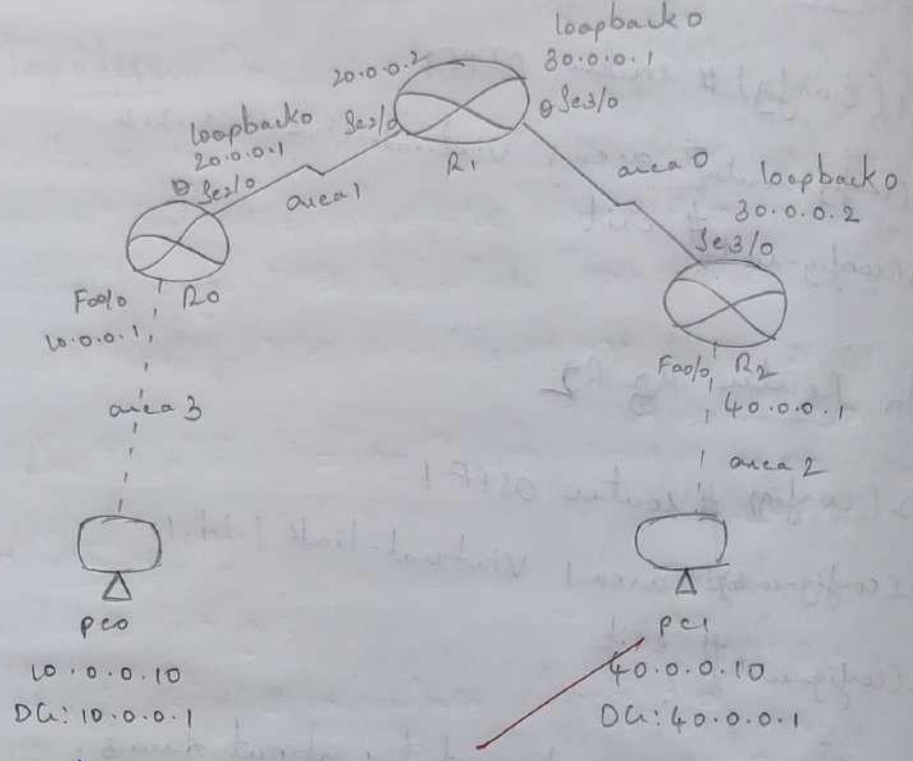
Procedure and Observations:

27/07/23

Aim-7

7. Configure OSPF routing Protocol.

Topology:



Procedure:

=> Configure PCs & the routers with IP address & default gateway according to the topology seen above.

=> Configure the routers as seen in the topology to be set as done in rip routing protocol experiment.

-> Now enable ip routing by configuring OSPF routing protocol in all routers.

In Router R0

R0(config)# router ospf 1
R0(config-router)# router-id 1.1.1.1
R0(config-router)# network 10.0.0.0 0.255.255.255 area 3
R0(config-router)# network 20.0.0.0 0.255.255.255 area 1
R0(config-router)# exit

In Router R1

R1(config)# router ospf 1
R1(config-router)# router-id 2.2.2.2
R1(config-router)# network 20.0.0.0 0.255.255.255 area 1
R1(config-router)# network 30.0.0.0 0.255.255.255 area 0
R1(config-router)# exit

In Router R2

R2(config)# router ospf 1
R2(config-router)# router-id 3.3.3.3
R2(config-router)# network 30.0.0.0 0.255.255.255 area 0

R2 (config-router)# network 40.0.0.0 0.255.255.255 area 2

R2 (config-router)# exit

=> Now Configuring the Virtual interface (loopback)

In Router R0

Router # config t
router (config)# interface Serial 2/0
router (config-if)# interface loopback 0
router (config-if)# ip address 172.16.1.252 255.255.0.0
router (config-if)# no shut

In Router R1

router # config t
router (config)# interface Serial 3/0
router (config-if)# interface loopback 0
router (config-if)# ip address 172.16.1.253 255.255.0.0
router (config-if)# no shut

In Router R2

router # config t
router (config)# interface Serial 3/0
router (config-if)# interface loopback 0

router (config-if) # ip address 172.16.1.254 255.255.0.0

router (config-if) # no shut

=> Create Virtual line between R1, R2 by this we create a virtual link to connect area 3 to area 0.

In Router R0

router # config t

router (config) # router OSPF 1

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

router (config-router) # exit

In Router R1

router # config t

router (config) # router OSPF 1

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

router (config-router) # exit

=> R1 & R2 get updates about Area 3. Now, check routing table of R0.

router

In Router R1

router (config) # exit

Router # show ip route

C 10.0.0.0/8 is directly Connected, FastEthernet 0/0

20.0.0.0/8 is Variably Subnetted, 2 subnets, 2 masks

C 20.0.0.0/8 is directly Connected, Serial 2/0

C 20.0.0.2/32 is directly Connected, Serial 2/0

O 30.0.0.0/8 (110/128) via 20.0.0.2, 00:01:13, Serial 2/0

O IA 40.0.0.0/8 (110/129) via 20.0.0.2, 00:01:13, Serial 2/0

C 172.16.0.0/16 is directly Connected, Loopback 0.

=> Now check routing table of R0 & R2.

In Router R0

router (config) # exit

router # show ip route

O IA 10.0.0.0/8 (110/65) via 20.0.0.1, 00:01:17, Serial 2/0

20.0.0.0/8 is Variably Subnetted, 2 subnets, 2 masks

C 20.0.0.0/8 is directly connected, Serial 2/0

C 20.0.0.1/32 is directly connected, Serial 2/0

30.0.0.0/8 is Variably Subnetted, 2 Subnets, 2 masks

C 30.0.0.0/8 is directly connected, Serial 3/0

C 30.0.0.2/32 is directly connected, Serial 3/0

O IA 40.0.0.0/8 (110/65) via 30.0.0.2, 00:11:37, Serial 3/0

C 172.16.0.0/16 is directly connected, Loopback0

In Router R2

router (config) # exit

router # show ip route

O IA 10.0.0.0/8 (110/129) via 30.0.0.1, 00:01:5, Serial 3/0

O IA 20.0.0.0/8 (110/128) via 30.0.0.1, 00:12:26, Serial 3/0

30.0.0.0/8 is Variably Subnetted, 2 Subnets, 2 masks

C 30.0.0.0/8 is directly connected, Serial

C 30.0.0.1/32 is directly connected, Serial 3

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

C 172.16.0.0/16 is directly connected, Loopback0

=> Pinging from PC0 to PC1

=> 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=2ms
TTL=125

Reply from 40.0.0.10: byte=32 time=7ms
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=3, Lost=1 (25% loss),

Approximate round trip times in milli-seconds:

Minimum = 2ms, Maximum = 7ms, Average = 5ms

=> Pinging from PC1 to PC0

=> Ping 10.0.0.10

Pinging 10.0.0.10 with 32 bytes of data:

10

Reply from 10.0.0.10: bytes = 32 time = 2ms
TTL = 125

Reply from 10.0.0.10: bytes = 32 time = 2ms
TTL = 125

Reply from 10.0.0.10: bytes = 32 time = 2ms
TTL = 125

Reply from 10.0.0.10: bytes = 32 time = 2ms
TTL = 125

Ping statistics for 10.0.0.10:

Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),

Approximate round trip times in milli-seconds

Minimum = 2ms, Maximum = 2ms, Average = 8 ms.

Observation:

-> OSPF (Open Shortest Path First)

-> It is a open standard routing protocol

-> link state routing protocol

-> Algorithm uses is "Dijkistra", to find shortest path.

Lee 21/7/23

Output:

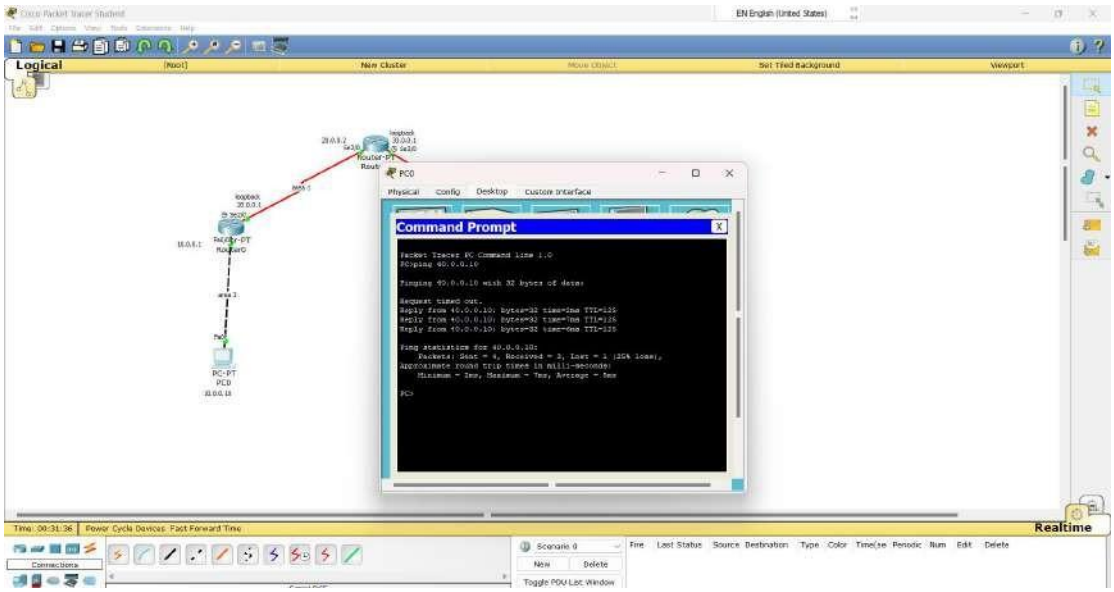


Fig 5: Pinging from Pc0 to Pc1

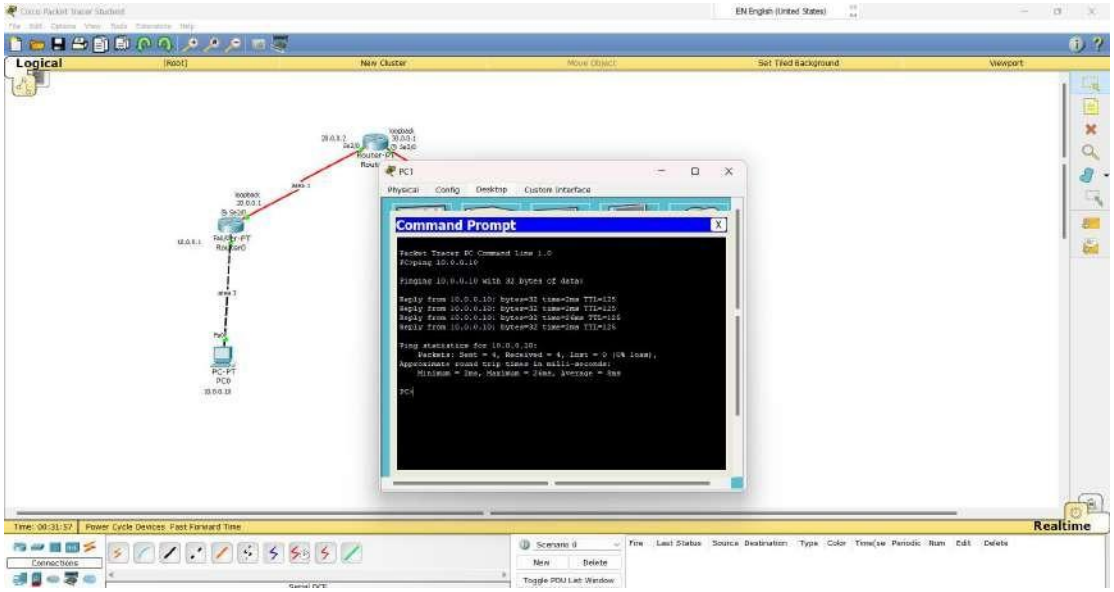


Fig 6: Pinging from Pc1 to Pc0

Experiment No. 8

Cycle 1

Aim-8

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

Topology:

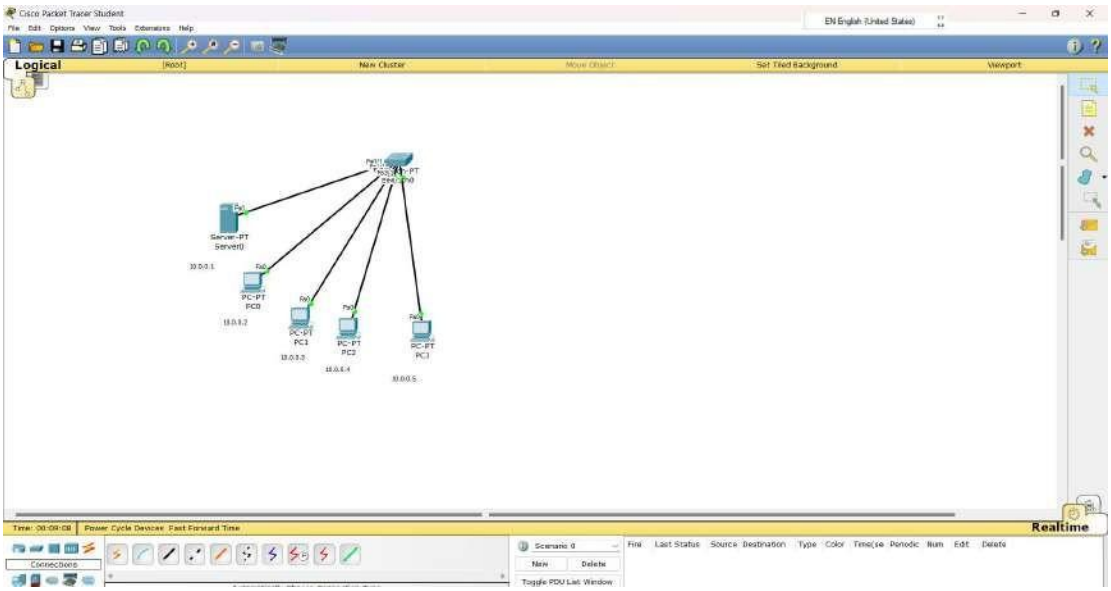


Fig 1: Topology

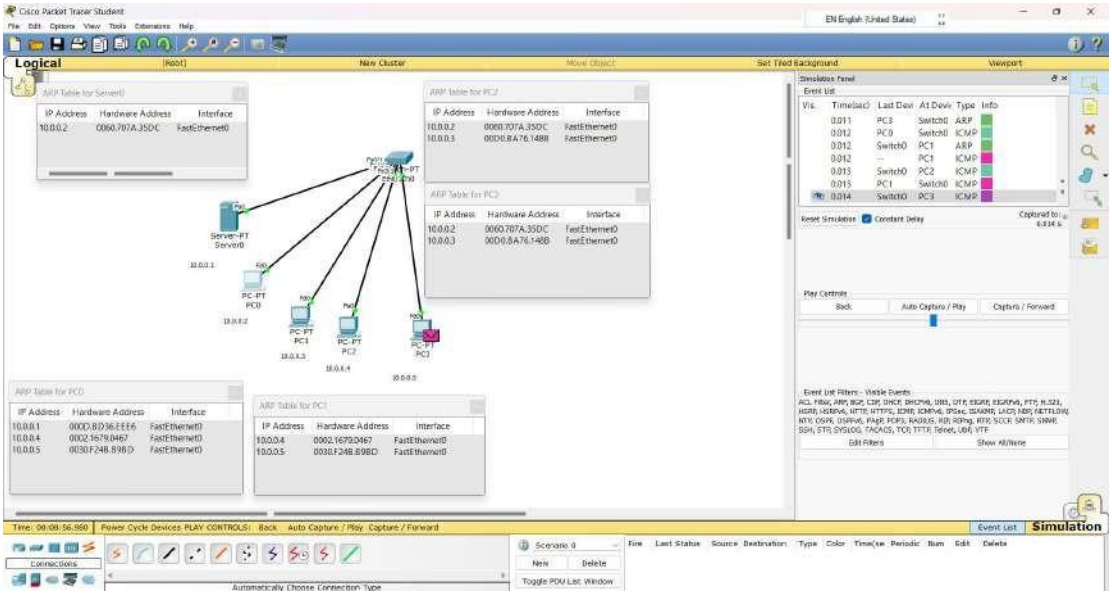


Fig 2: ARP tables of the devices connected in the topology

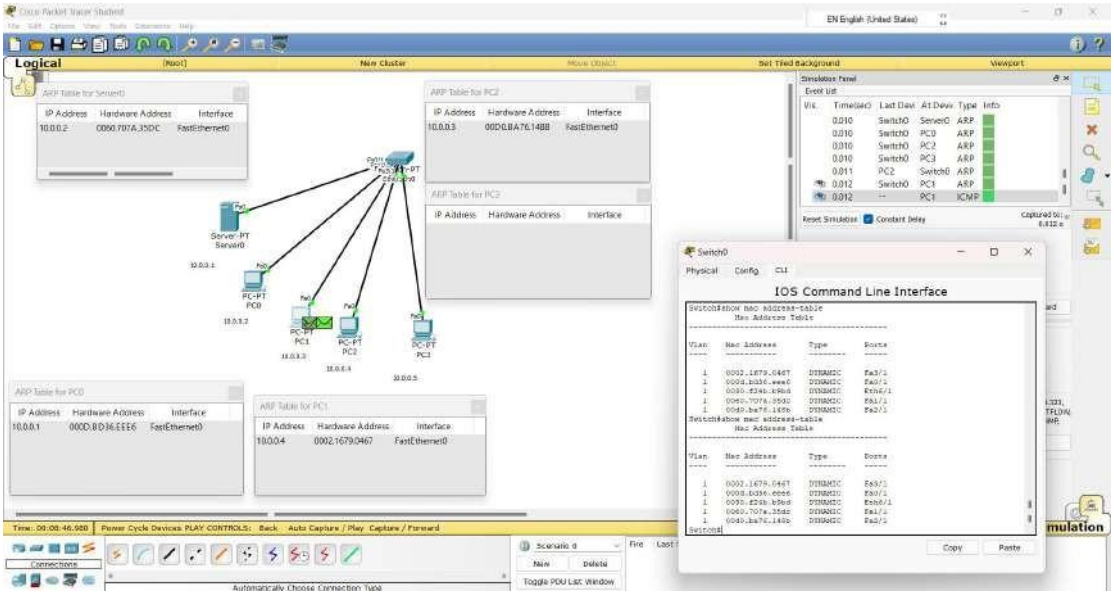


Fig 3: PDU packets are being sent/received with acknowledgement

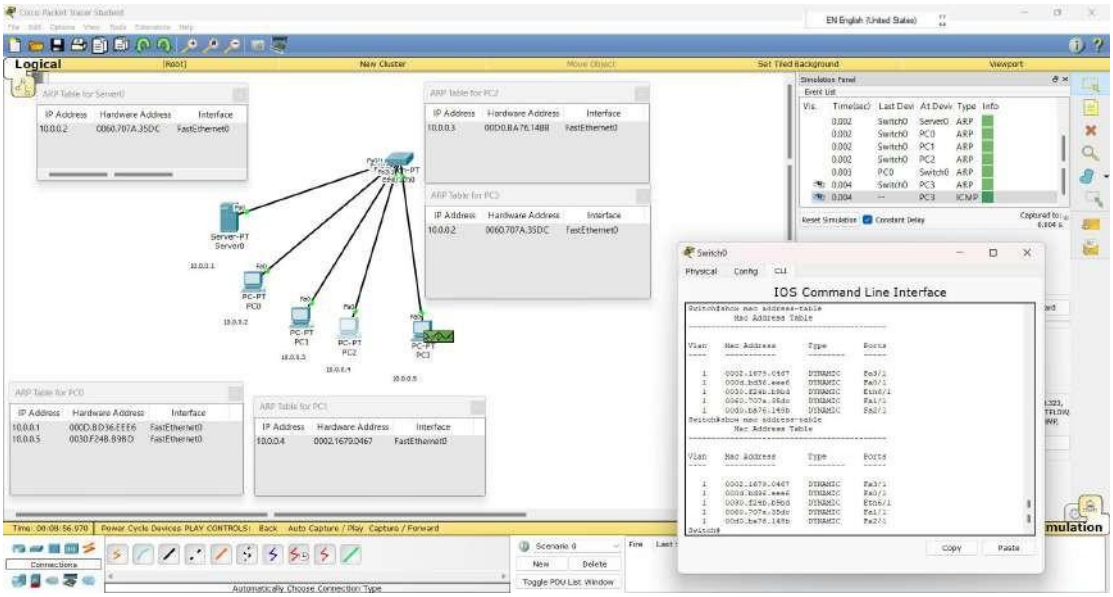
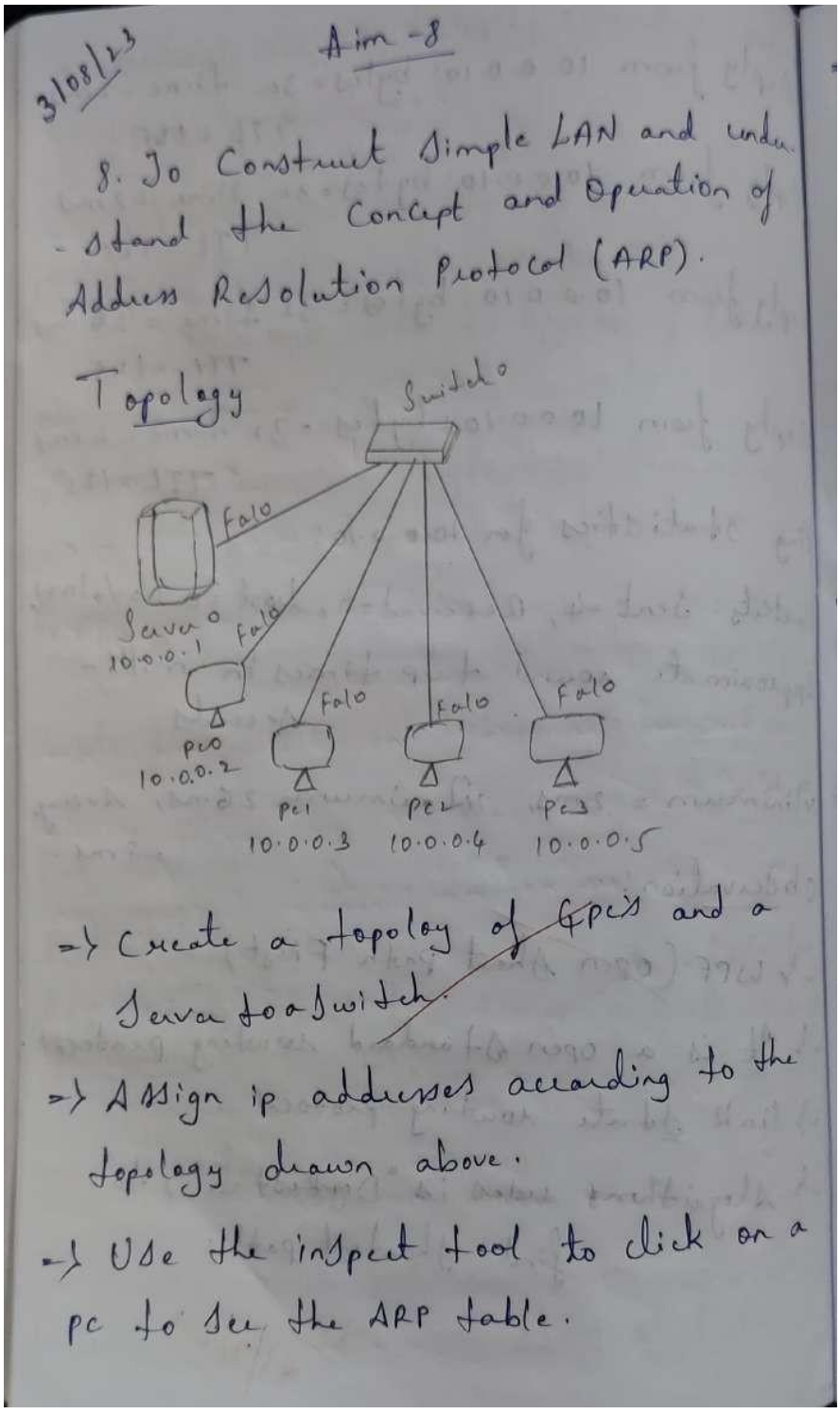


Fig 4: PDU packets are being sent/received with acknowledgement

Procedure and Observation:



-> Command in PKT CLI for the same is arp -a (this command is to be used in pc's command prompt which is situated in Desktop tab.

-> when the "arp -a" command is used initially you will get a reply saying ARP table is empty.

-> Also in CLI of switch, the command show mac address-table can be given on every transaction to see how the switch learns from transactions & build the address-table.

-> Use the capture/forward button in the simulation panel to go step by step so that the changes in ARP can be clearly noted.

Outcome

Output observation

-> when ping message is passed from pc0 to Server0 in Simulation mode.

Commands

* (pc0 - Server0)

-> arp -a

No ARP Entries Found.

-> Ping 10.0.0.1

-> then click on capture/forward

-> Yo After the message / ARP packets / packets are passed there an entry made in Server0's ARP table

"-> 10.0.0.2 0060.2072.35DC FastEthernet

-> then there an entry made in pc0's ARP table.

"-> 10.0.0.1 000D.BD36.EEE6 FastEthernet0

-> Like wise the packets / messages have been passed from PC1 - PC2, PC3 to PC0, PC2 - PC0 & PC1 - PC3.

-> the entries in respective PCs ARP table will be made as Switch learns their MAC addresses.

Output

-> Now check the MAC address table in Switch

-> check the MAC address table in Switch after each transaction is being made.

Switch> enable

Switch# show mac address-table

Mac Address Table

Vlan	Mac Address	Type	Ports
1	0002.1697.0467	DYNAMIC	Fa3/1
1	0090.bd36.eceb	DYNAMIC	Fa0/1
1	0030.f24b.69bd	DYNAMIC	Eth6/1
1	0060.707a.35dc	DYNAMIC	Fa4/1
1	00d0.ba7b.1486	DYNAMIC	Fa2/1

Output:

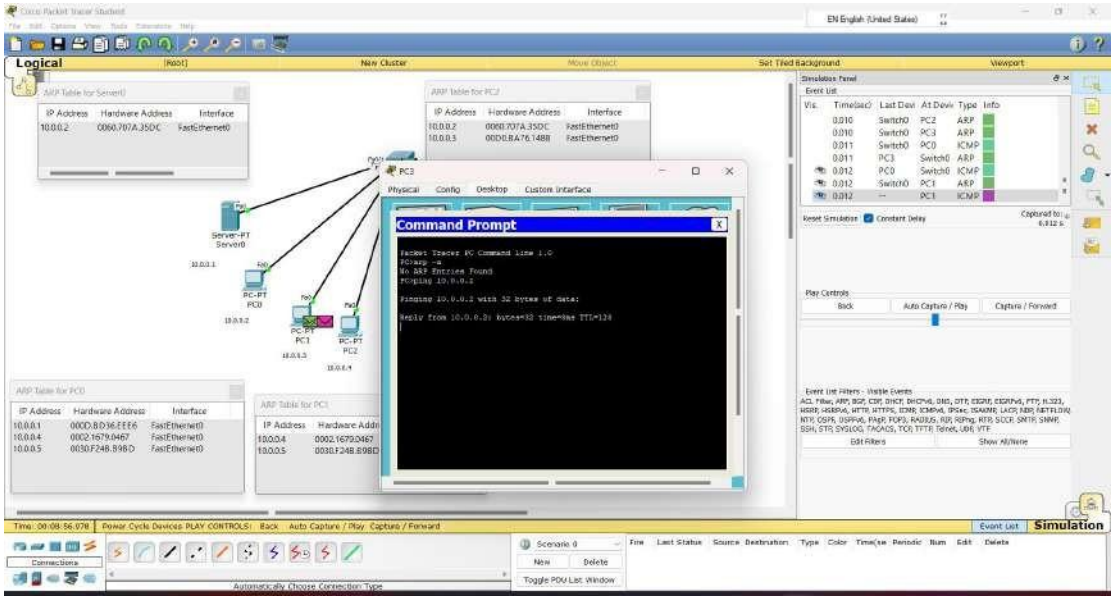


Fig 5: Initially you will not find any entries in ARP table

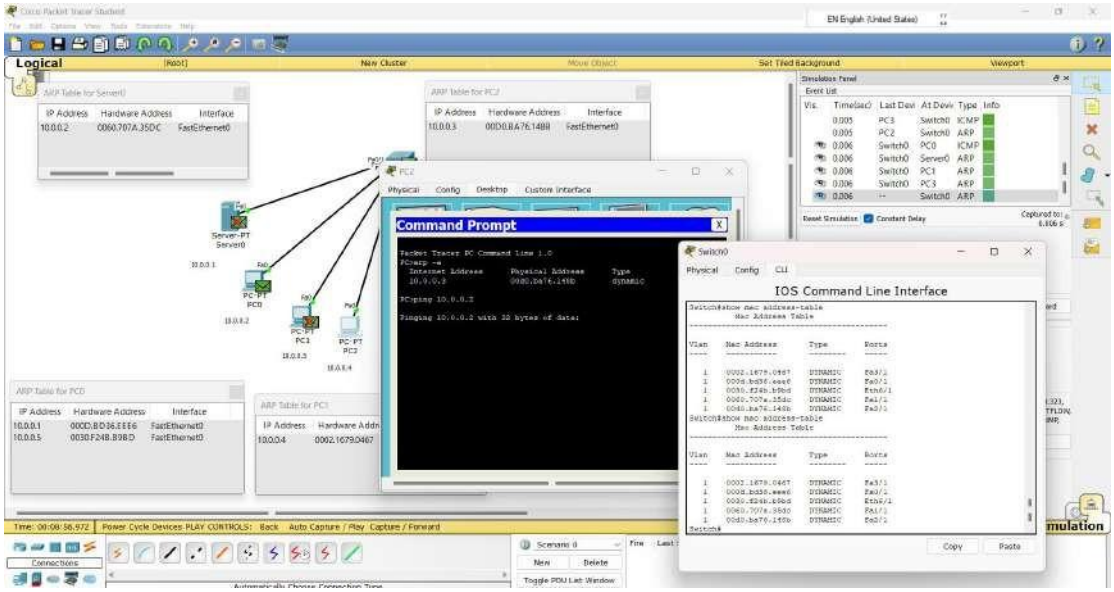


Fig 6: showing the contents of the ARP table

Fig 7: Showing mac address-table for every transaction

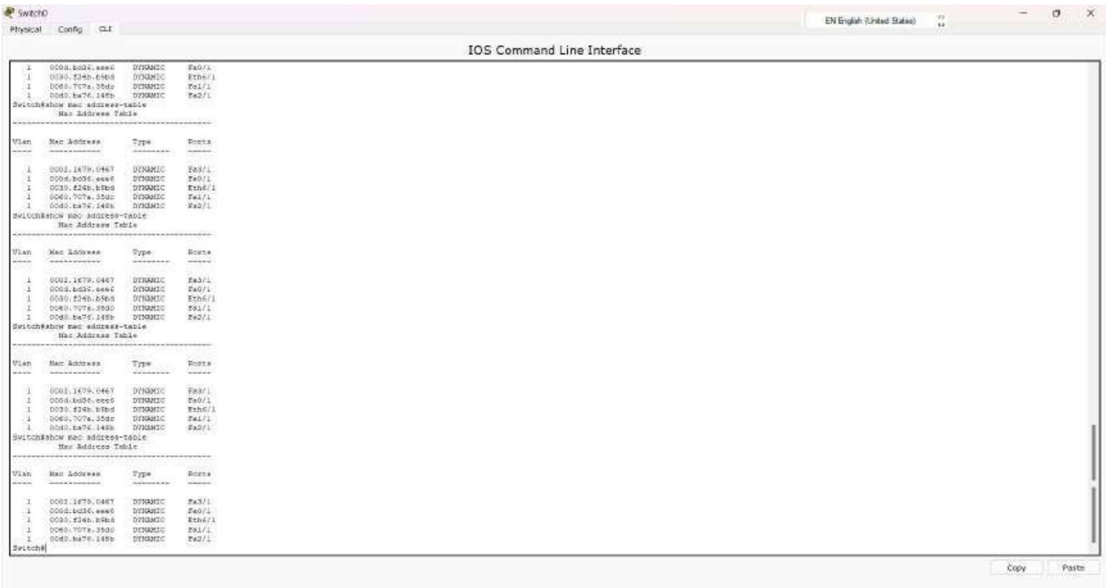


Fig 8: Showing mac address-table for every transaction

Experiment No. 9

Cycle 1

Aim-9

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

Topology:

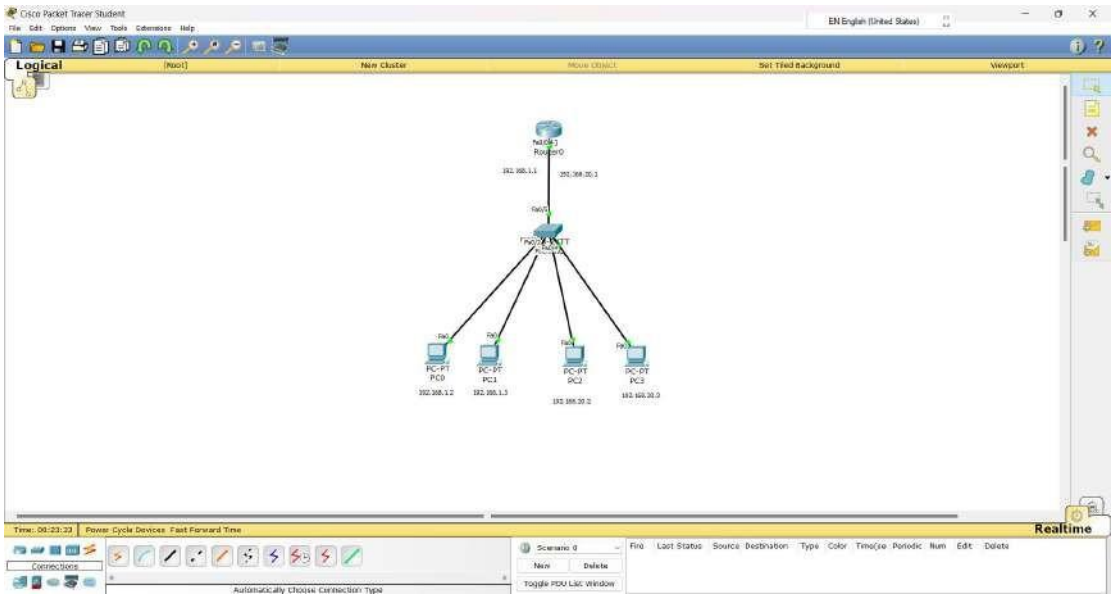


Fig 1: Topology

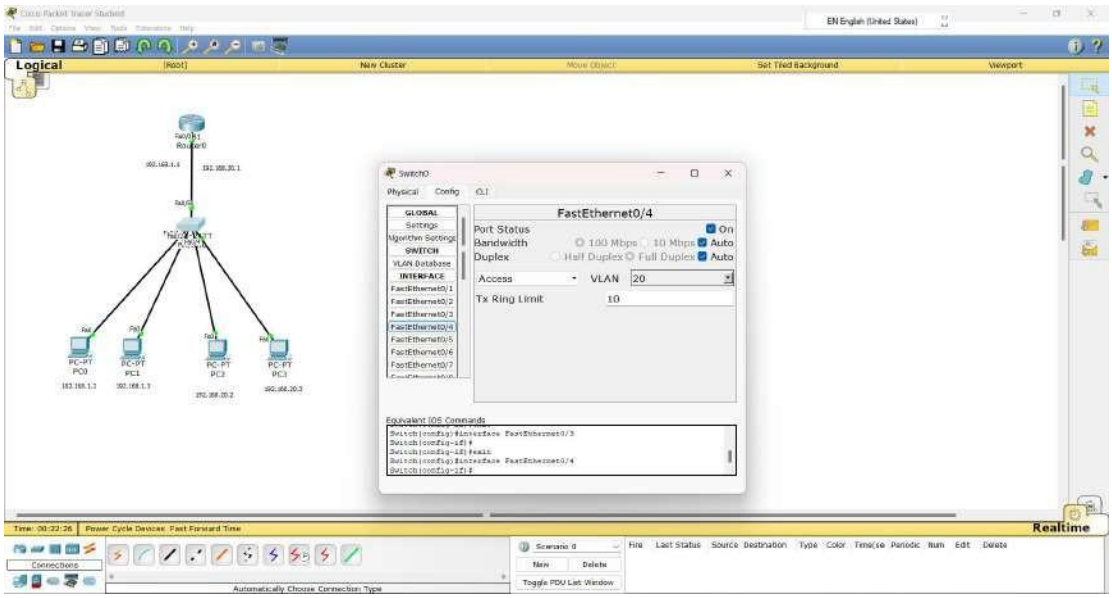


Fig 2: Fastethernet port configuration

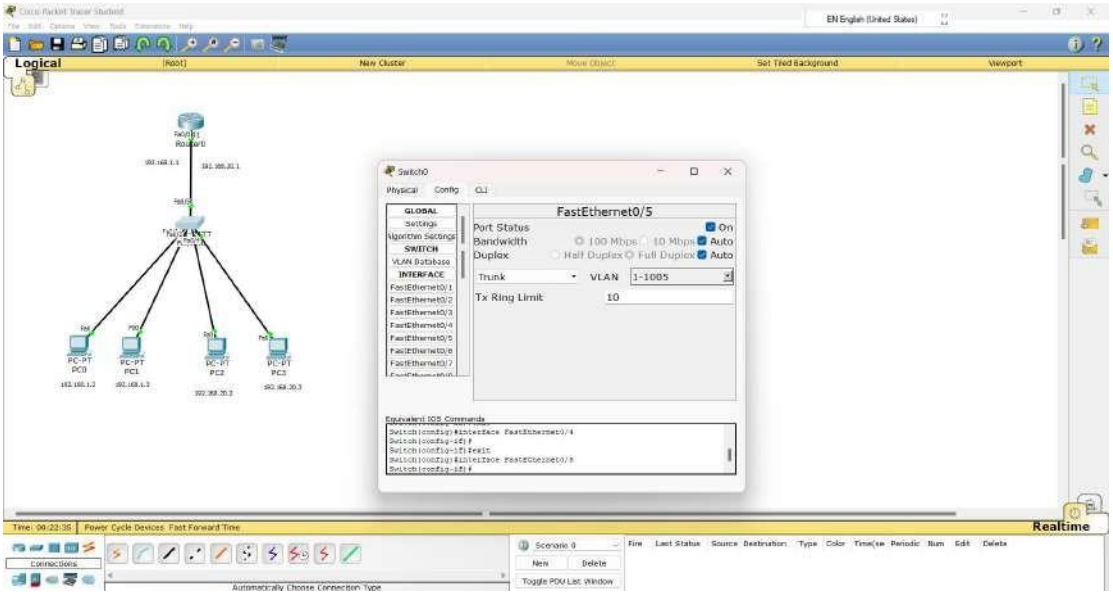


Fig 3: Port Configuration

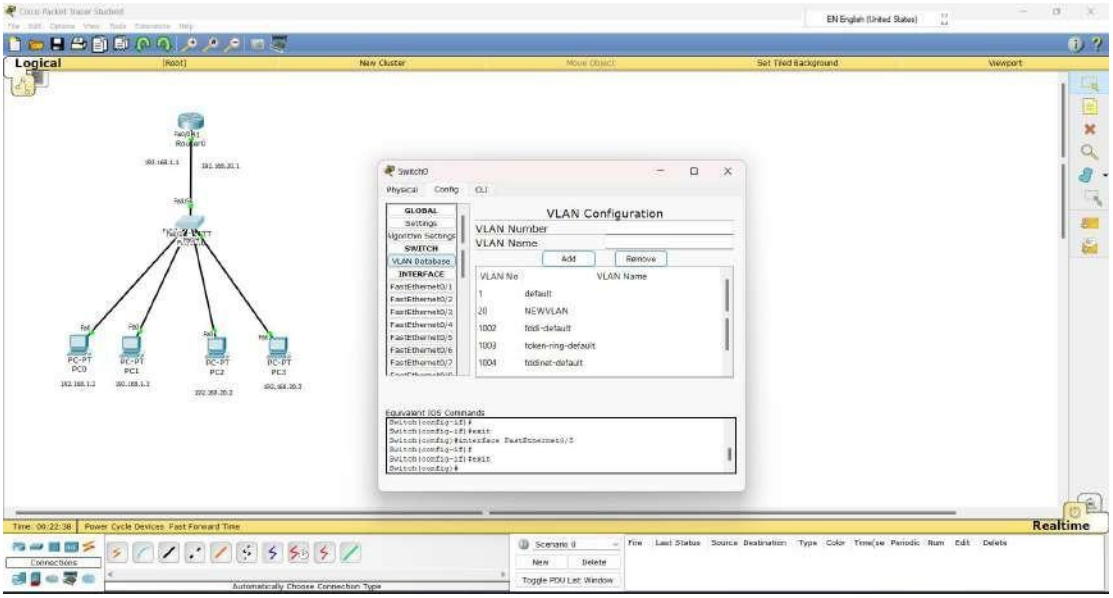


Fig 4: VLAN configuration

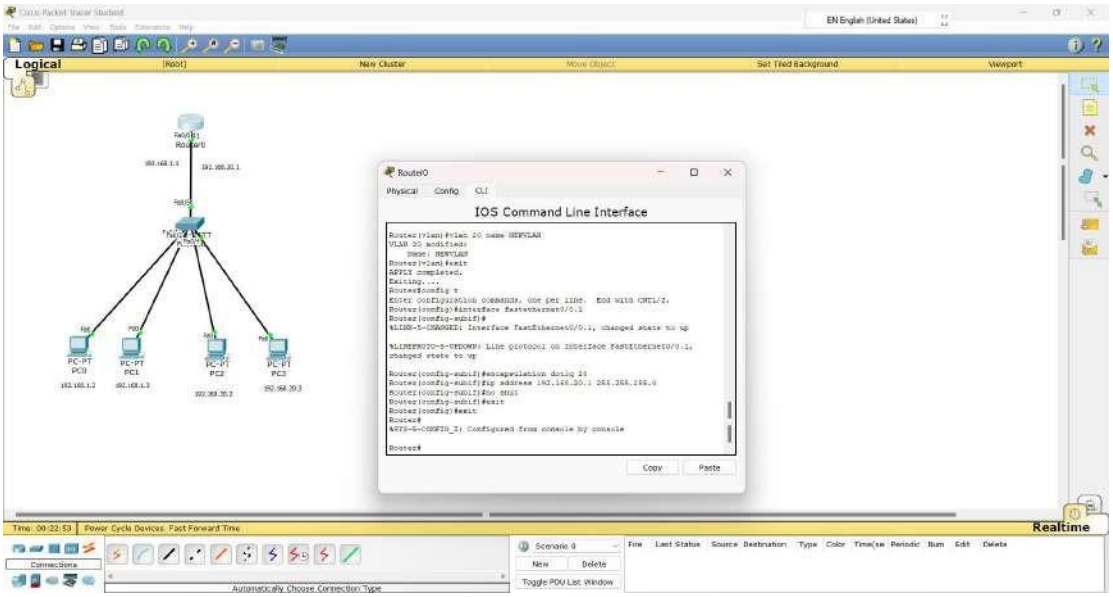


Fig 5: Virtual port configuration

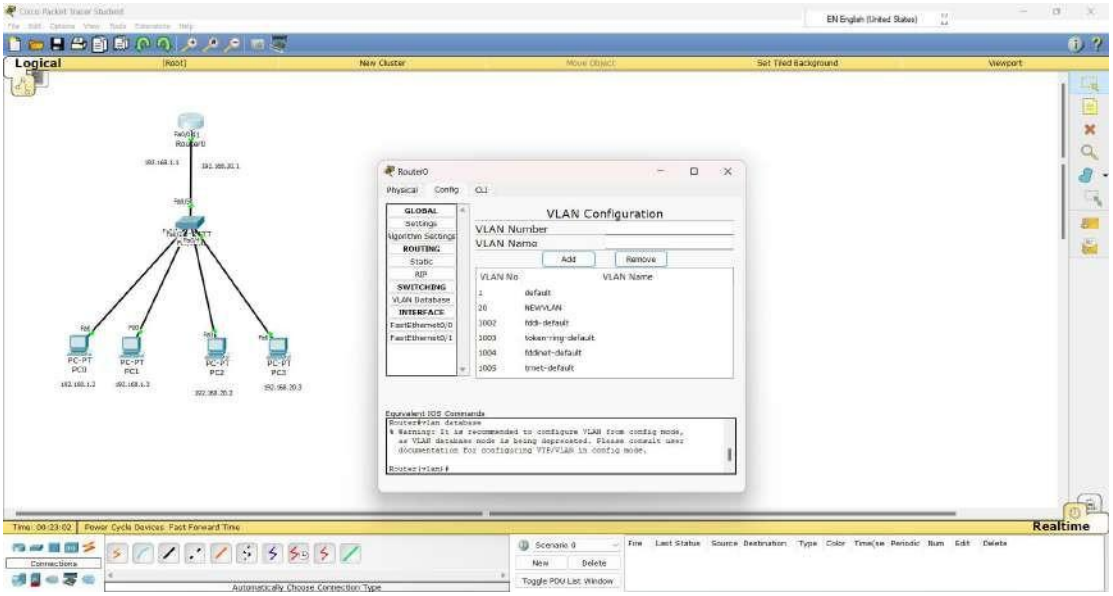


Fig 6: Newly created VLAN

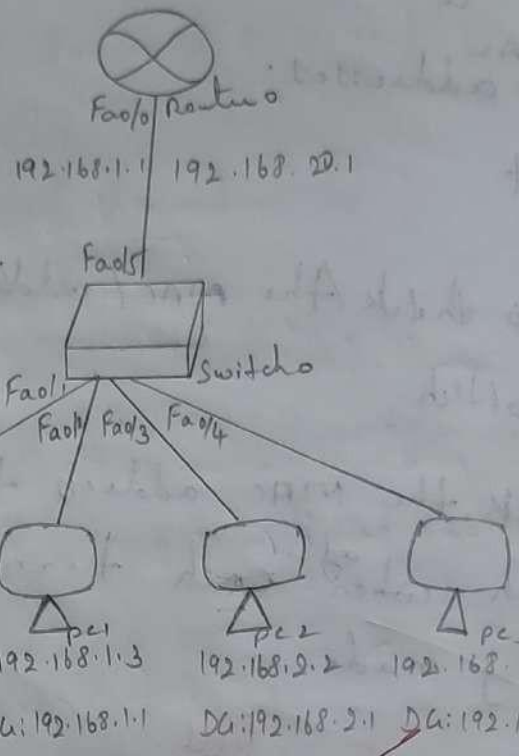
Procedure and Observation:

Aim-9

6/08/23

9. To Construct a VLAN and make the pc's communicate among a VLAN.

Topology:



Procedure:

-> Create a topology of one router (1841), a Switch and 4 PC's

-> Assign the IP addresses of the PC's as shown above.

-> Now only configure the interface which near router (i.e; Fa0/0) just assign the ip address & Subnet mask (i.e; 192.168.1.1 and ~~255.0.0~~ 255.255.255.0 respectively).

-> In the Switch, go to config tab and Select VLAN Database.

-> In place of VLAN Number give "20" and in place of VLAN Name give any name i.e; for ex (NEWVLAN) and click on "Add".

-> Select the interface which near the Switch from router (i.e; Fa0/5) in the Switch's Config tab.

-> Select "trunk" by clicking on drop down (left side).

-> Select the interfaces which are near the Switch from the PCs i.e; Fa0/3 & Fa0/4 in the Switch's config tab.

-> click on VLAN drop down and Select the or check tick on the 20: NEWVLAN.

-> Now go to router config tab click on VLAN Database and enter the number and name of VLAN created i.e; here we gave (20, NEWVLAN).

-> Now Goto router CLI and give the below commands.

Router(VLAN) # exit
Router # config t
Router (config) # interface fastethernet 0/0.1
Router (config-subif) # encapsulation dot1q 20
Router (config-subif) # ip address 192.168.20.1 255.255.255.0
Router (config-subif) # no shut
Router (config-subif) # exit
Router (config) # exit

Output

→ Passing ping commands from pc0 to pc3 and from pc3 to pc0.

"pc0 to pc3"

pc> ping 192.168.20.3

Pinging 192.168.20.3 with 32 bytes of data:

Request timed out

Reply from 192.168.20.3: bytes=32 time=1ms 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

Ping statistics for 192.168.20.3:

Packets: Sent =4, Received =3, Lost =1 (25% loss),

Approximate round trip times in milli-seconds:

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

"pc8 to pc0"

pc8 Ping 192.168.1.2

Pinging 192.168.1.2 with 32 bytes of data:

Reply from 192.168.1.2: bytes=32 time=0ms TTL=127

Reply from 192.168.1.2: bytes=32 time=0 ms TTL=127

Reply from 192.168.1.2: bytes=32 time=1ms TTL=127

Reply from 192.168.1.2: bytes=32 time=0 ms TTL=127

Ping Statistics for 192.168.1.2:

Packets: Sent=4, Received=4, Lost=0 (0% loss),

Approximate round trip times in milli-seconds:

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

"pc1 to pc2"

pc> ping 192.168.20.2

Pinging 192.168.20.2 with 32 bytes of data:

Request timed out.

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

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

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

Ping Statistics for 192.168.20.2:

Packets: Sent=4, Received=3, Lost=1 (25% loss),

Approximate round trip times in milli-seconds:

Minimum=0ms, Maximum=1ms, Average=0ms.

"pc2 to pc1"

pc2 Ping 192.168.1.3

Pinging 192.168.1.3 with 32 bytes of data:

Reply from 192.168.1.3: bytes=32 time=0 ms
TTL=127

Reply from 192.168.1.3: bytes=32 time=0 ms
TTL=127

Reply from 192.168.1.3: bytes=32 time=0 ms
TTL=127

Reply from 192.168.1.3: bytes=32 time=0 ms
TTL=127

Ping statistics for 192.168.1.3:

Packets: Sent=4, Received=4, Lost=0 (0% loss)

Approximate round trip times in milli-seconds:

Minimum=0ms, Maximum=0ms,
Average=0ms.

observation:

- Virtual LAN (VLAN).
- Here we use a concept called VLAN trunking allows switches to forwards frames diffuent VLANS over a single link called trunk.
- This is done by adding an additional header information called tag to the Ethernet frame. The process of adding this small header is called VLAN tagging.

lww
10/8/23

Output:

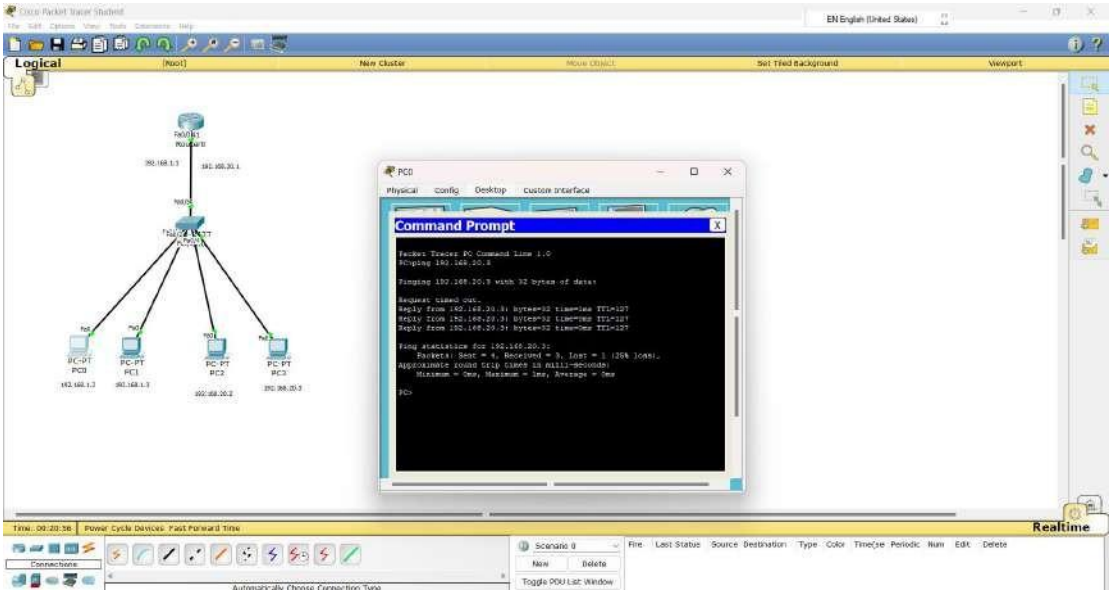


Fig 7: pinging from pc0 to pc3

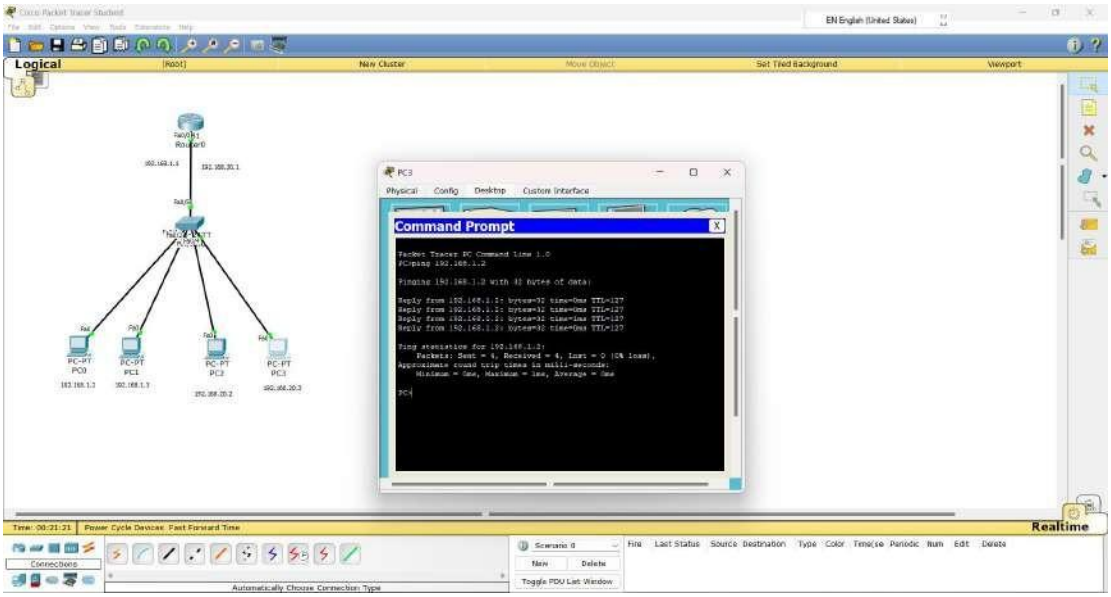


Fig 8: pinging from pc3 to pc0

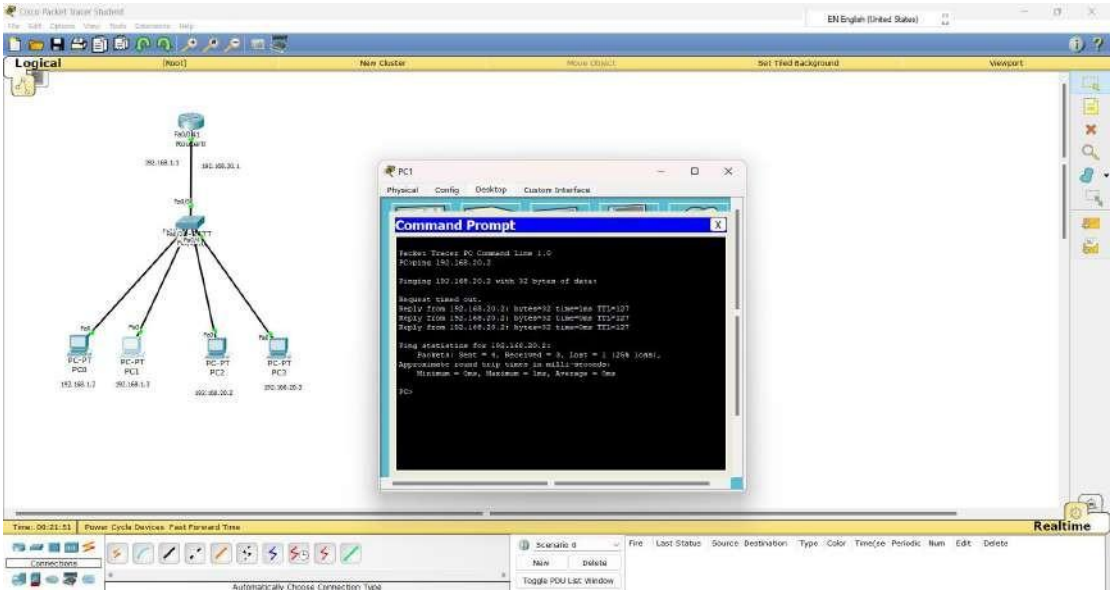


Fig 9: pinging from pc1 to pc2

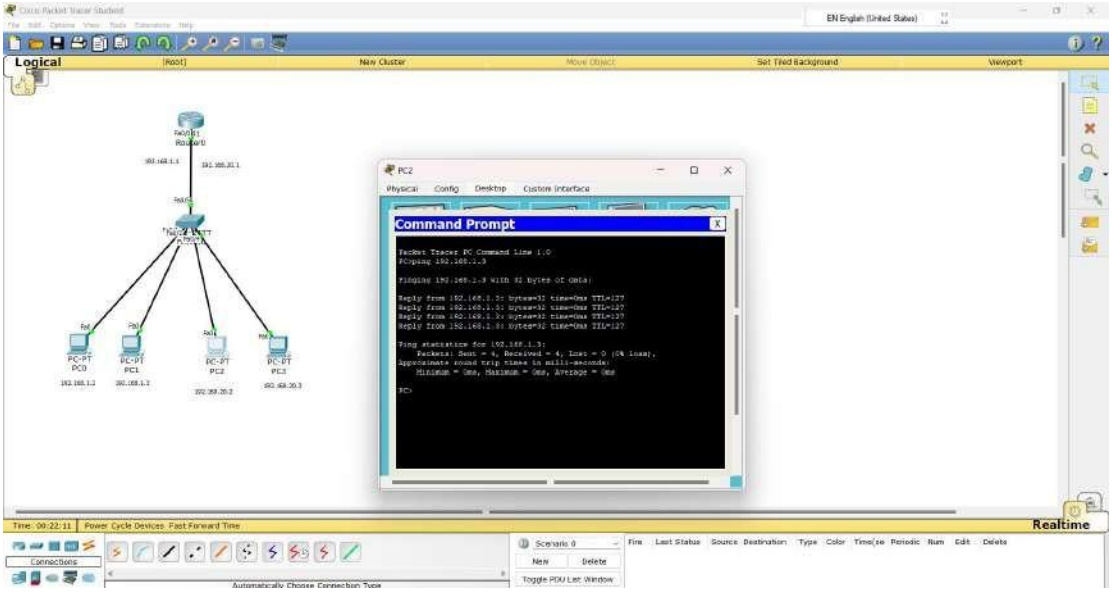


Fig 10: pinging from pc2 to pc1

Experiment No. 10
Cycle 1

Aim-10

10. Demonstrate the TTL/ Life of a Packet

Topology:

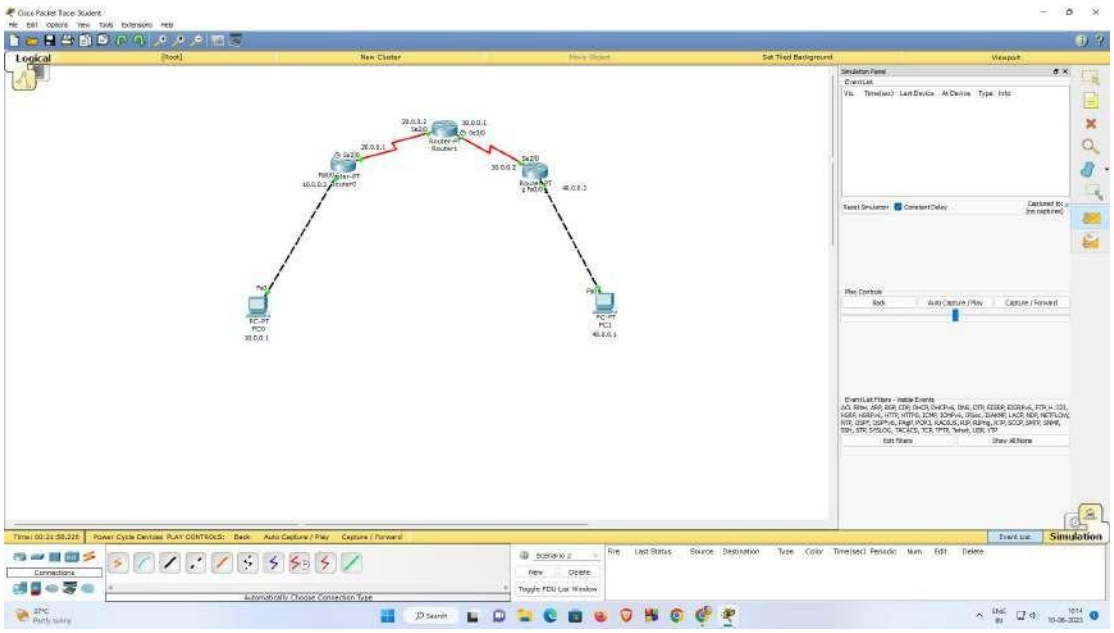


Fig 1: Topology

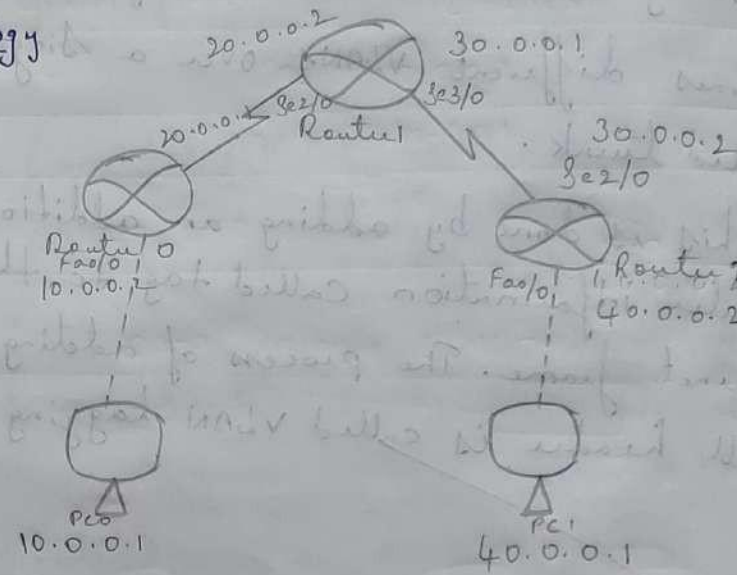
Procedure and Observation:

10/08/23

Aim-10

10) Demonstrate the TTL / Life of a packet.

Topology



Procedure

=> Create a topology of 2 pc's and 3 routers.

=> Configure the IP addresses and default gateway of the pc's.

=> Configure the devices as per static/ default / dynamic routing.

=> In the Simulation mode, send a simple PDU from one pc to another.

=> Use capture button to capture every transfer.

=> click on the PDU during every transfer to see the Inbound and Outbound PDU details observe that there is a difference of 1 in TTL when it crosses every router.

Output

11/9/2023

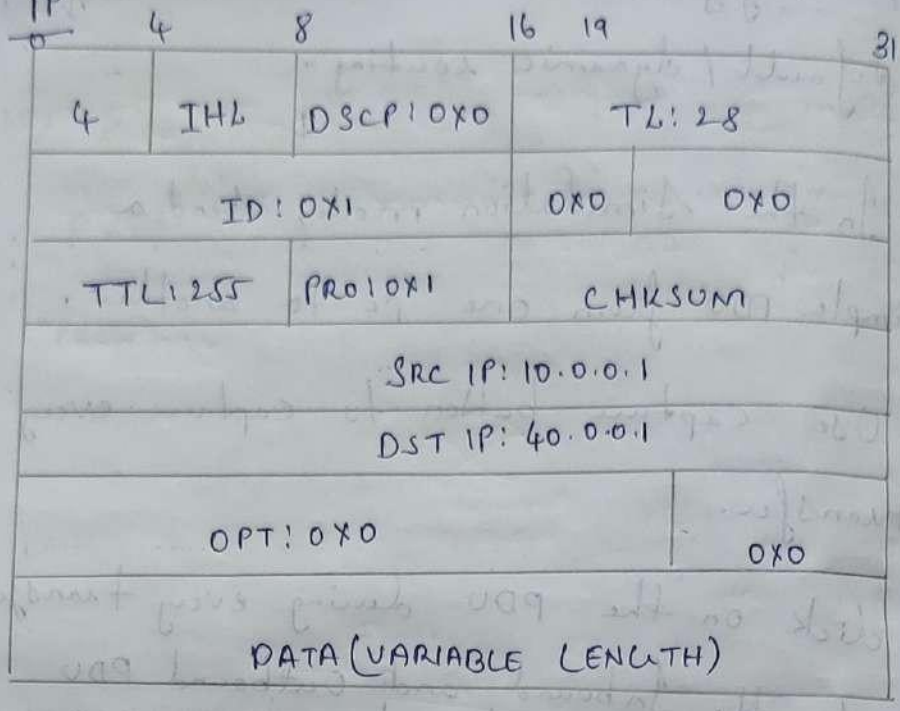
=> Initially when a packet (PDU) is sent from pc1 to pc0 (when it is at pc0 with the acknowledgement).

The PDU information at Device pc0

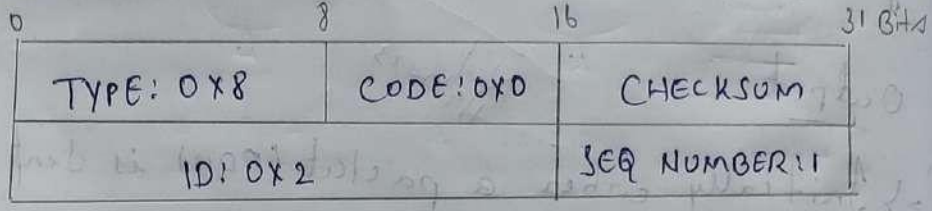
Ethernet II (header)

0	4	8	14	Bytes 19
PREAMBLE 10101010....1011		DEST MAC: 000C:8586:DA04	SRC MAC: 000A:F3C3:7764	
TYPE: 0X800	DATA(VARIABLE LENGTH)			FCS 0X0

IP

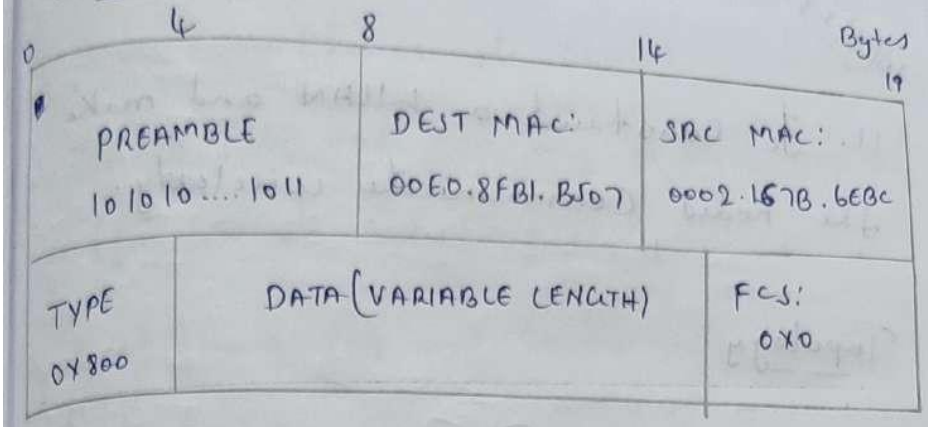


ICMP

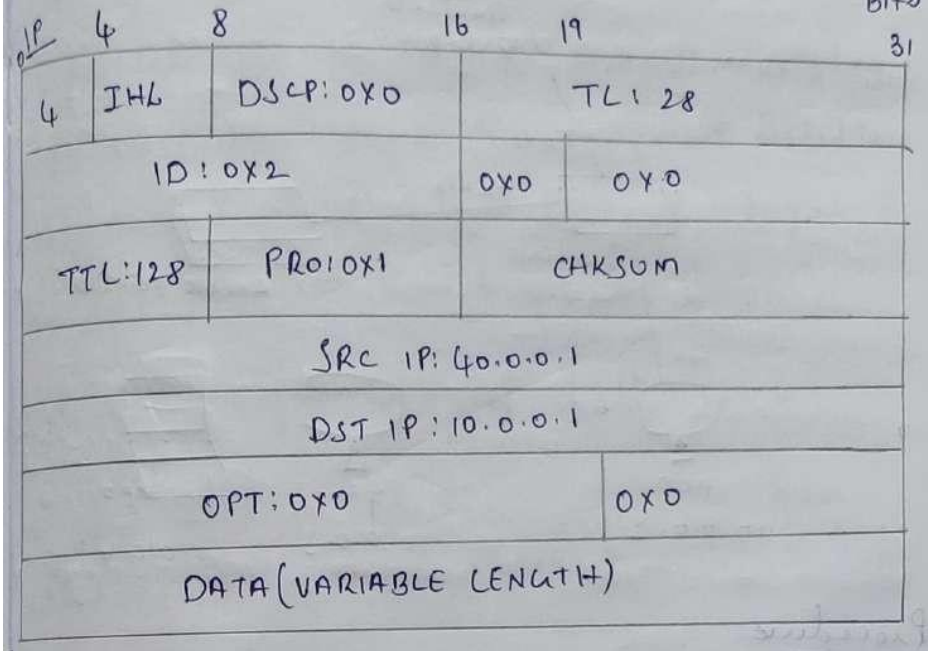


→ When the PDU has been sent reached successfully PC1 ~~with~~ the PDU information at Device PC1 is as follows.

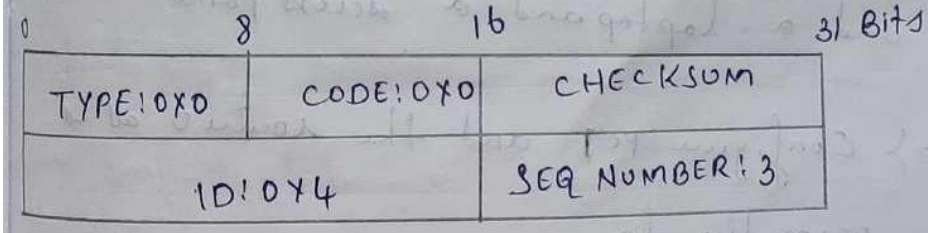
Ethernet II



IP



ICMP



Output:

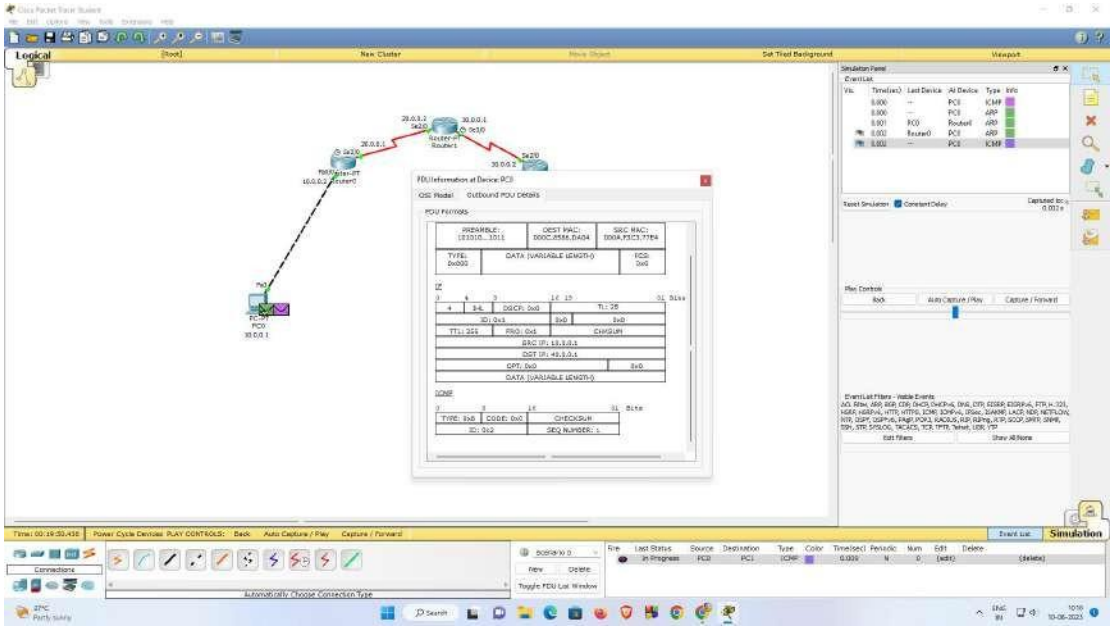


Fig 2: PDU information at device pco

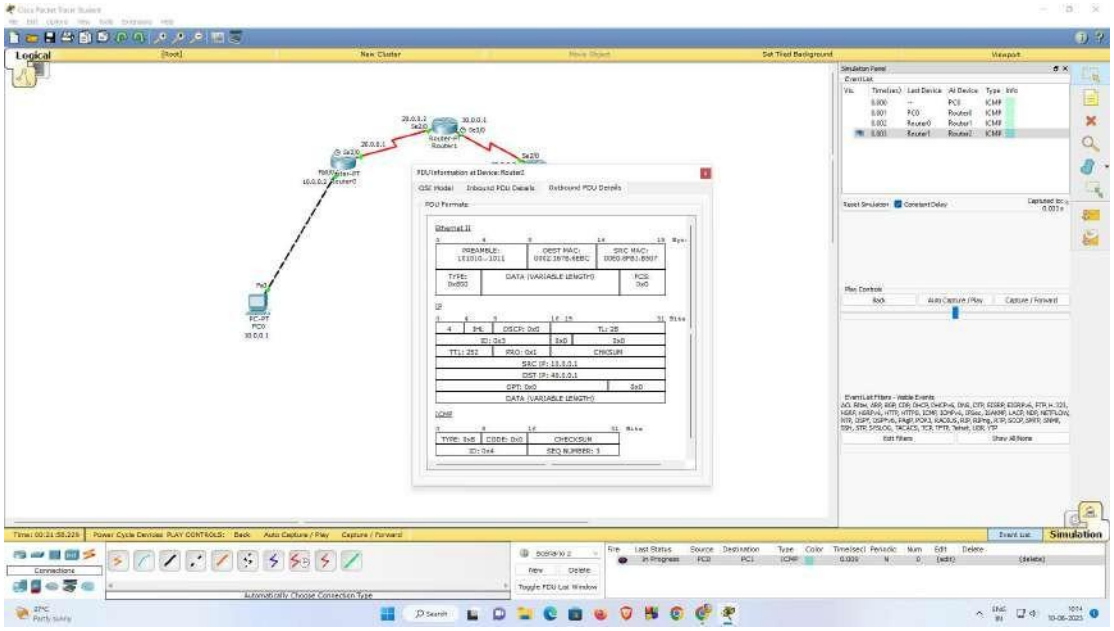


Fig 3: PDU information at device router2

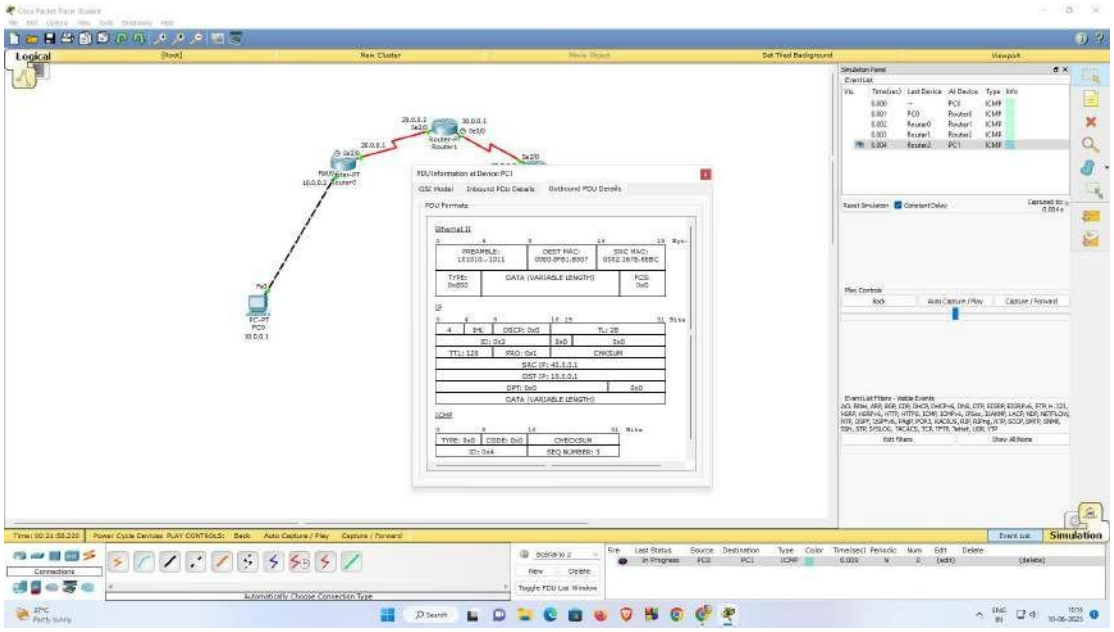


Fig 4: PDU information at device pc1

Experiment No. 11

Cycle 1

Aim-11

11. To construct a WLAN and make the nodes communicate wirelessly

Topology:

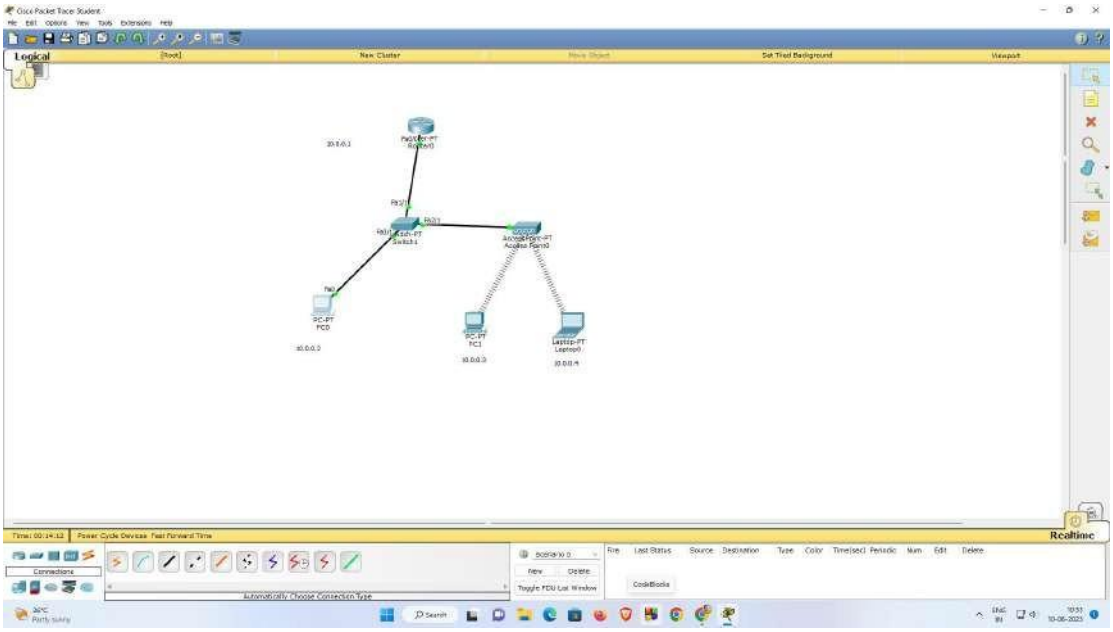


Fig 1: Topology

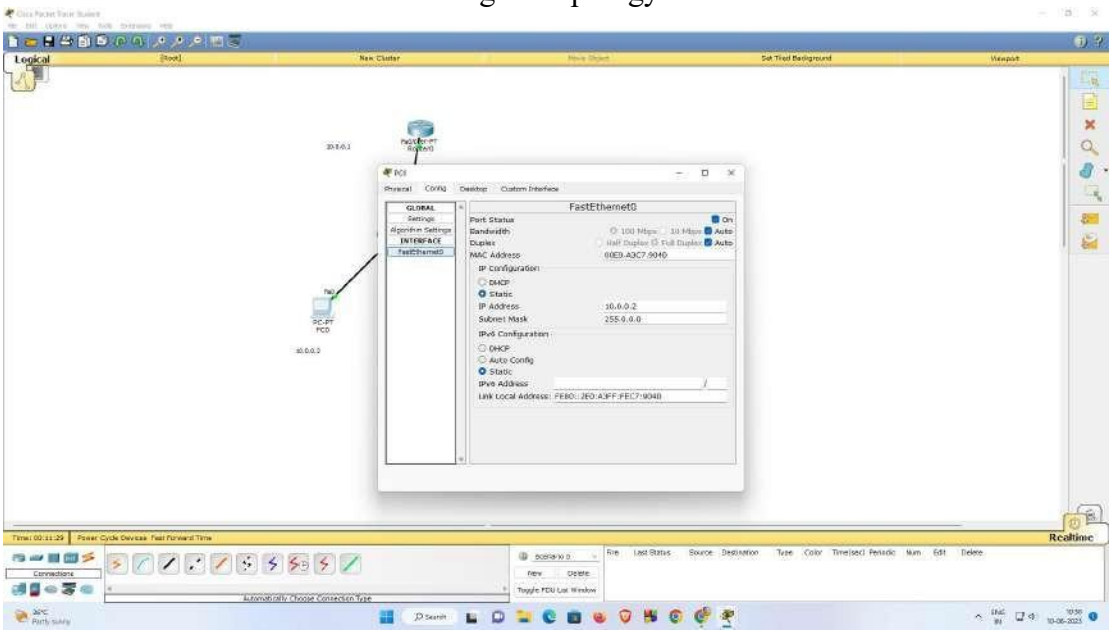


Fig 2: Pco configuration

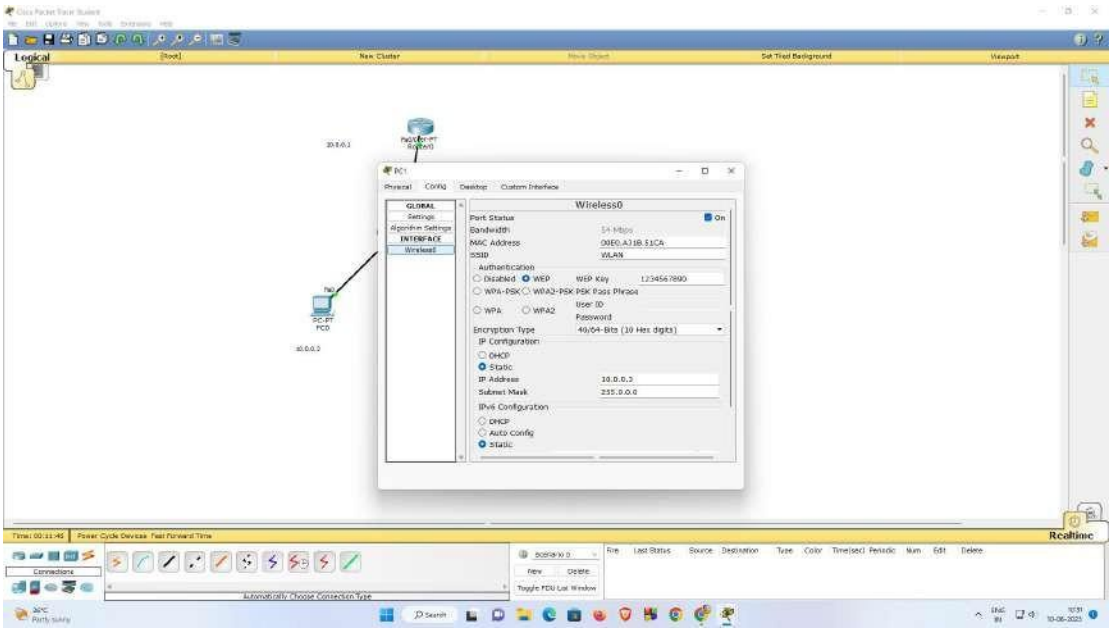


Fig 3: Pc1 configuration

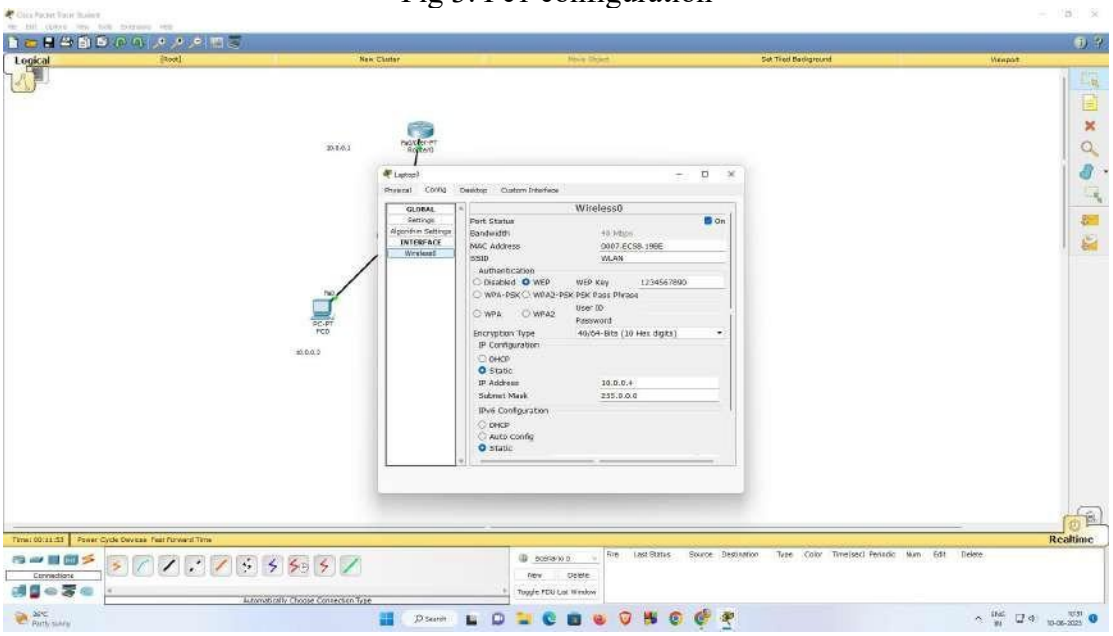


Fig 4: Laptop0 Configuration

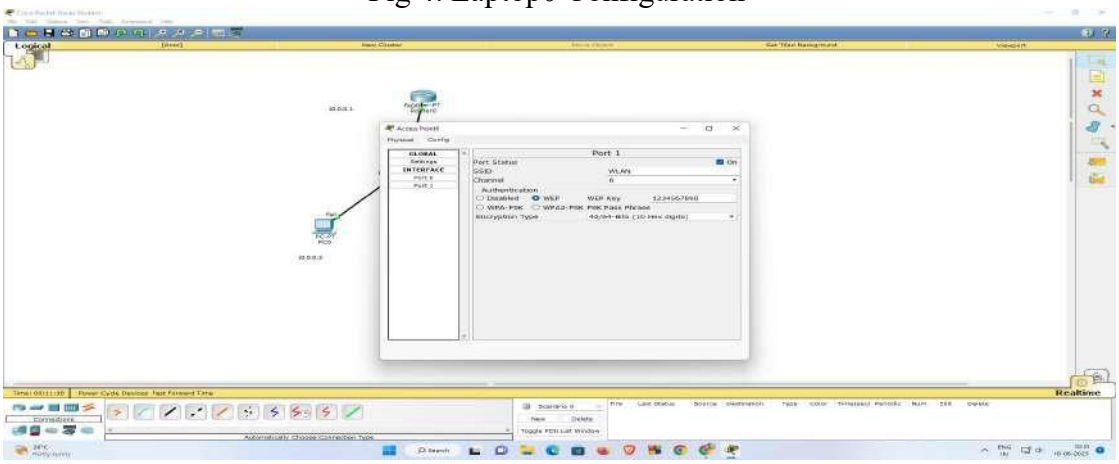


Fig 4: Access point0 Configuration

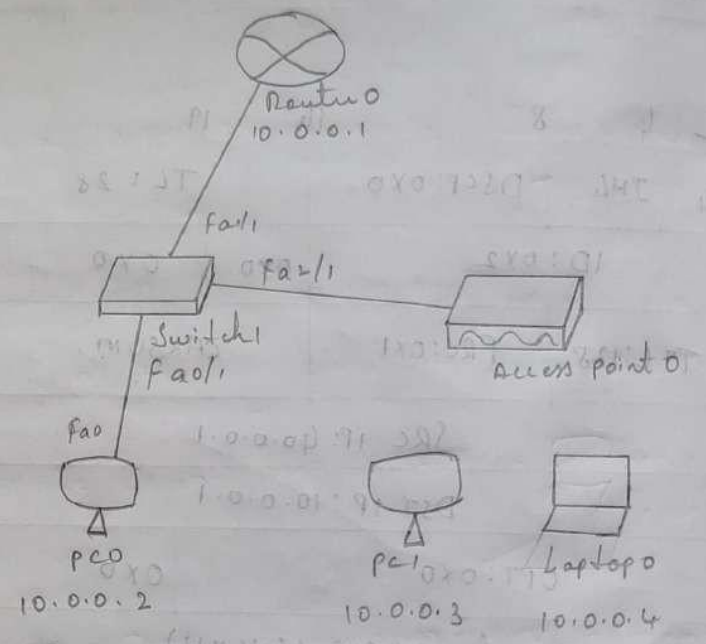
Procedure and Observation:

10/8/23

AIM-11

11. To construct a WLAN and make the nodes communicate wirelessly.

Topology



Procedure

=> Create a topology of a router 2 PCs and a laptop and a access point.

=> Configure PCs and the router0 as is normally done.

-> Configure Access point 1 - port 1 -> SSID Name - any name (for example WLAN).

-> Select WEP and give any 10 digit hex key - (for example 1234567890).

-> Configuring PC0 and laptop with wireless Handards.

-> Switch off the device. Drag the existing PT-HOST-NM-1AM to the Component listed in the LHS. Drag WMP300N wireless interface to the empty port. Switch on the device.

-> In the Config tab of both pc1 and laptop a new wireless interface would've been added. Now configure SSID, WEP, WEP Key, IP address and Gateway (as normally done) to the device. (same values as access points).

Output

Pinging from PC0 to PC1

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=8ms
TTL=128

Reply from 10.0.0.3: bytes=32 time=11ms
TTL=128

Reply from 10.0.0.3: bytes=32 time=7ms
TTL=128

Reply from 10.0.0.3: bytes=32 time=8ms
TTL=128

Ping statistics for 10.0.0.3:
Packets: Sent=4, Received=4, Lost=0 (0% loss),
Approximate round trip times in milli-seconds:

Minimum = 7ms, Maximum = 11ms, Average = 8ms

-> Pinging from pc1 to laptop

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=9ms
TTL=128

-> Reply from 10.0.0.4: bytes=32 time=11ms TTL=128

Reply from 10.0.0.4: bytes=32 time=8ms 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=11ms, Average=8ms.

-> Now pinging from laptop0 to pc1

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=33ms TTL=128

Reply from 10.0.0.3: bytes=32 time=21ms TTL=128

Reply from 10.0.0.3 bytes=32 time=12ms TTL=128

Reply from 10.0.0.3 bytes=32 time=12ms 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 = 12 ms, Maximum = 33 ms, Average = 19 ms

=> Pinging from laptop 0 to PC0

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=23 ms TTL=128

Reply from 10.0.0.2: bytes=32 time=9 ms TTL=128

Reply from 10.0.0.2: bytes=32 time=12 ms TTL=128

Reply from 10.0.0.2: bytes=32 time=14 ms TTL=128

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

Approximate round trip times in milli-seconds:

Minimum = 9 ms, Maximum = 23 ms, Average = 14 ms

=> Now Pinging from pc1 to laptop 4

pc1 ping 10.0.0.4

Pinging 10.0.0.4 with 32 bytes of data:

Reply from 10.0.0.4: bytes=32 time=16 ms TTL=128

Reply from 10.0.0.4: bytes=32 time=18 ms TTL=128

Reply from 10.0.0.4: bytes=32 time=13 ms TTL=128

Reply from 10.0.0.4: bytes=32 time=14 ms 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 = 13 ms, Maximum = 18 ms, Average = 15 ms

=> Now ping from pc1 to pc0

pc1 ping 10.0.0.2

Pinging from 10.0.0.2 with 32 bytes of data

Reply from 10.0.0.2: bytes=32 time=27 ms TTL=128

Reply from 10.0.0.2: bytes=32 time=13 ms
TTL=128

Reply from 10.0.0.2: bytes=32 time=8 ms
TTL=128

Reply from 10.0.0.2: bytes=32 time=8 ms
TTL=128

Ping Statistics for 10.0.0.2:

Packets: Sent=4, Received=4, Lost=0 (0% loss),

Approximate round trip times in milli-seconds:

Minimum=0 ms, Maximum=27 ms, Average=
6 ms.

NLP
11/9/2023

Output

Using the Concept of VLAN we've
setup a wireless communication with
the help of access point device.

Output:

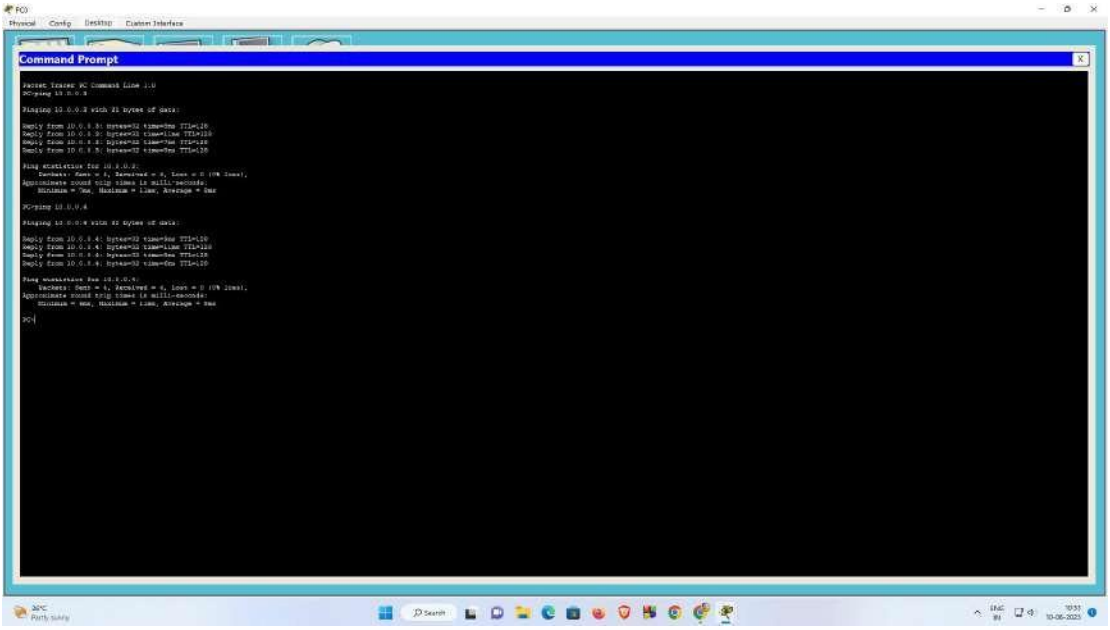


Fig 5: Ping responses from pc1 and laptop0

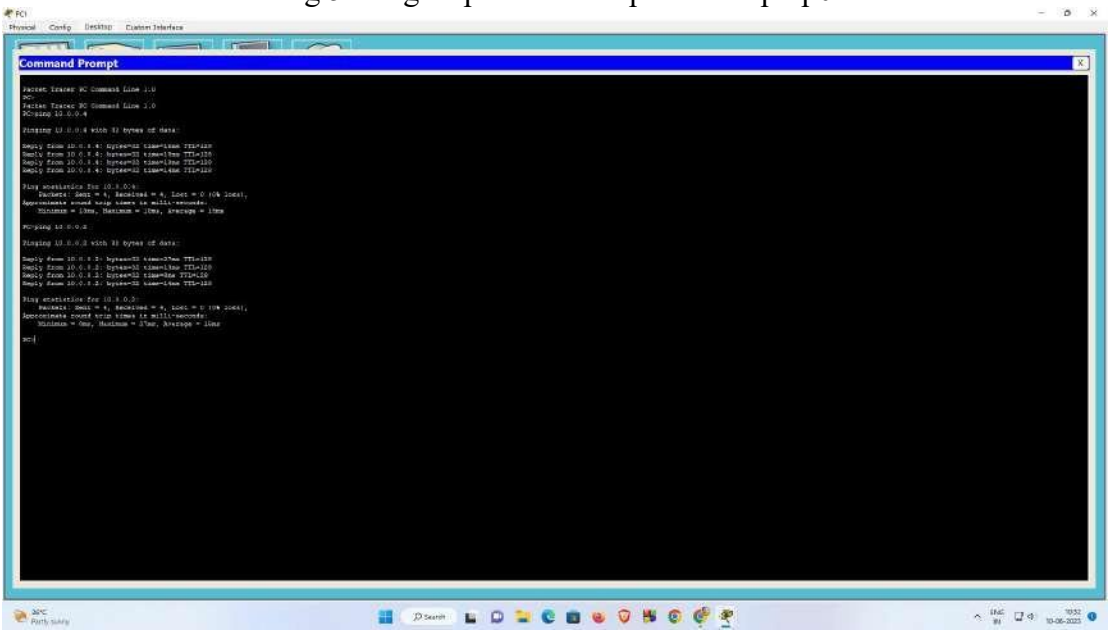


Fig 6: Ping responses from pc0 and laptop0

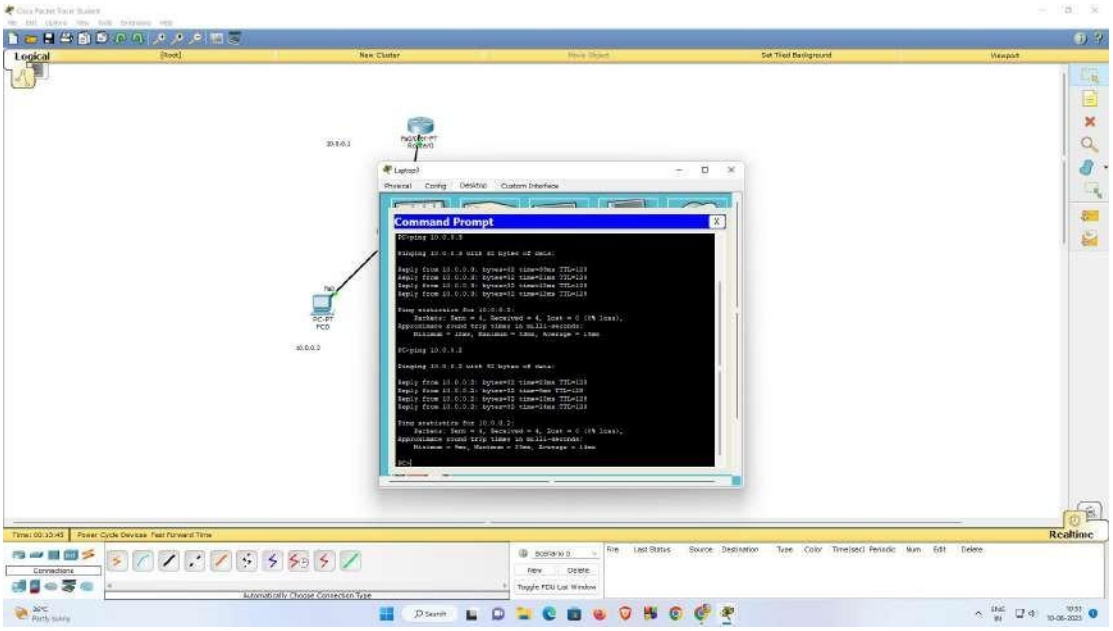


Fig 7: Ping responses from pc0 and pc1

Experiment No. 12

Cycle 1

Aim-12

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

Topology:

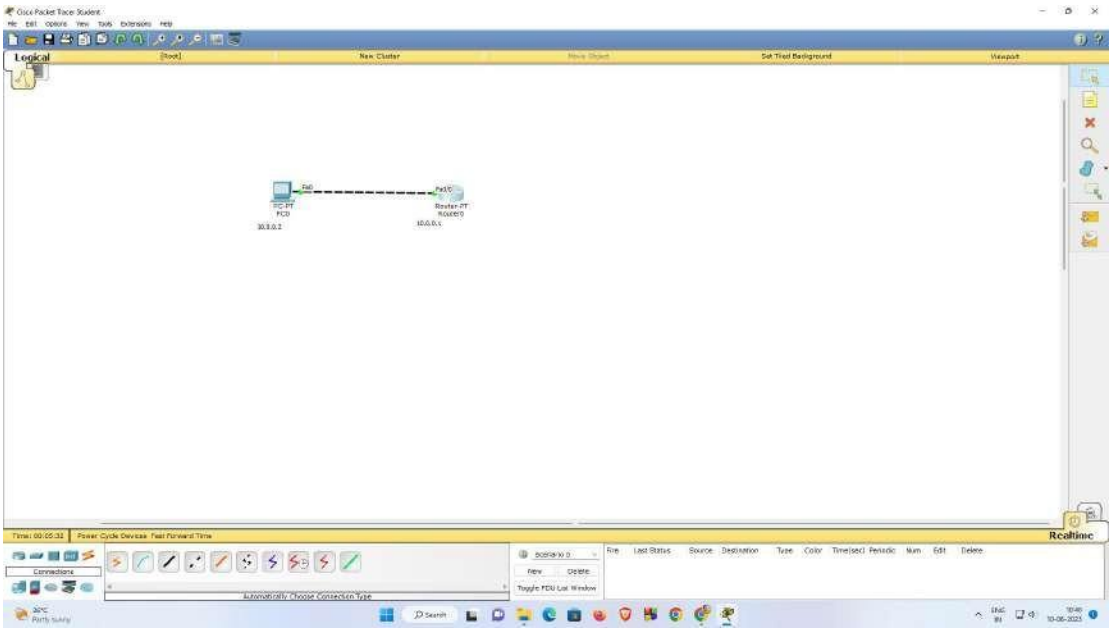


Fig 1: Topology

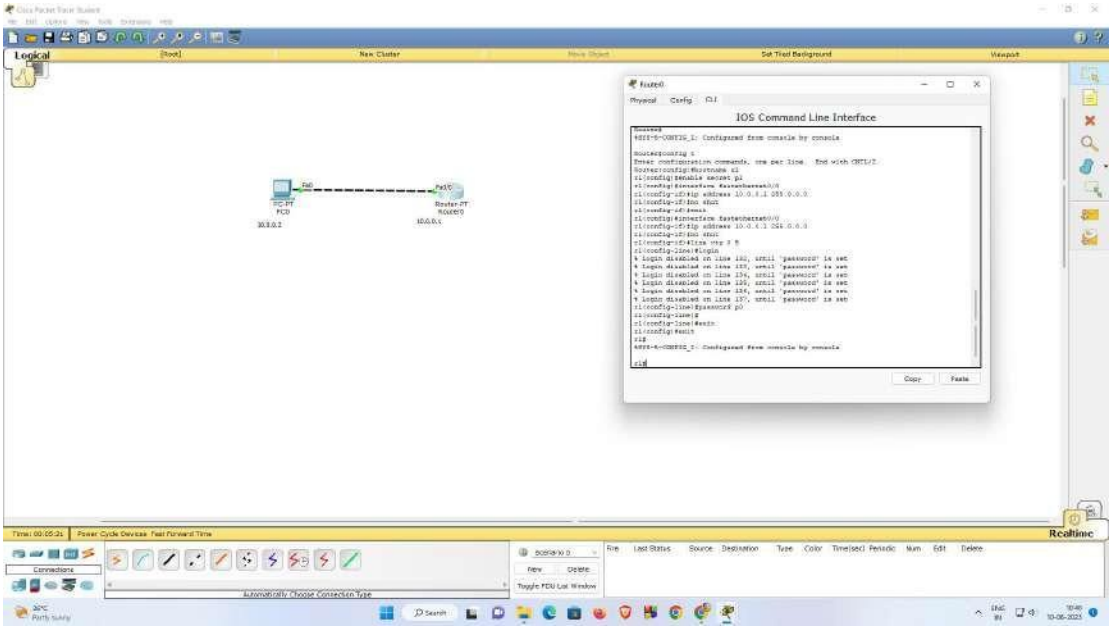


Fig 2: Router0 Configuration

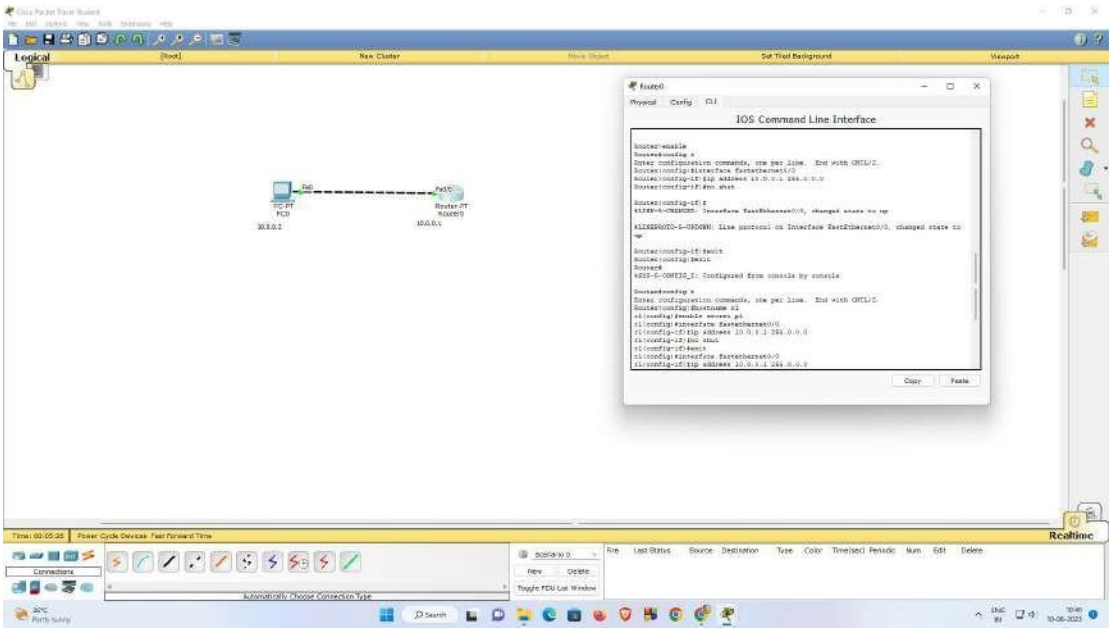
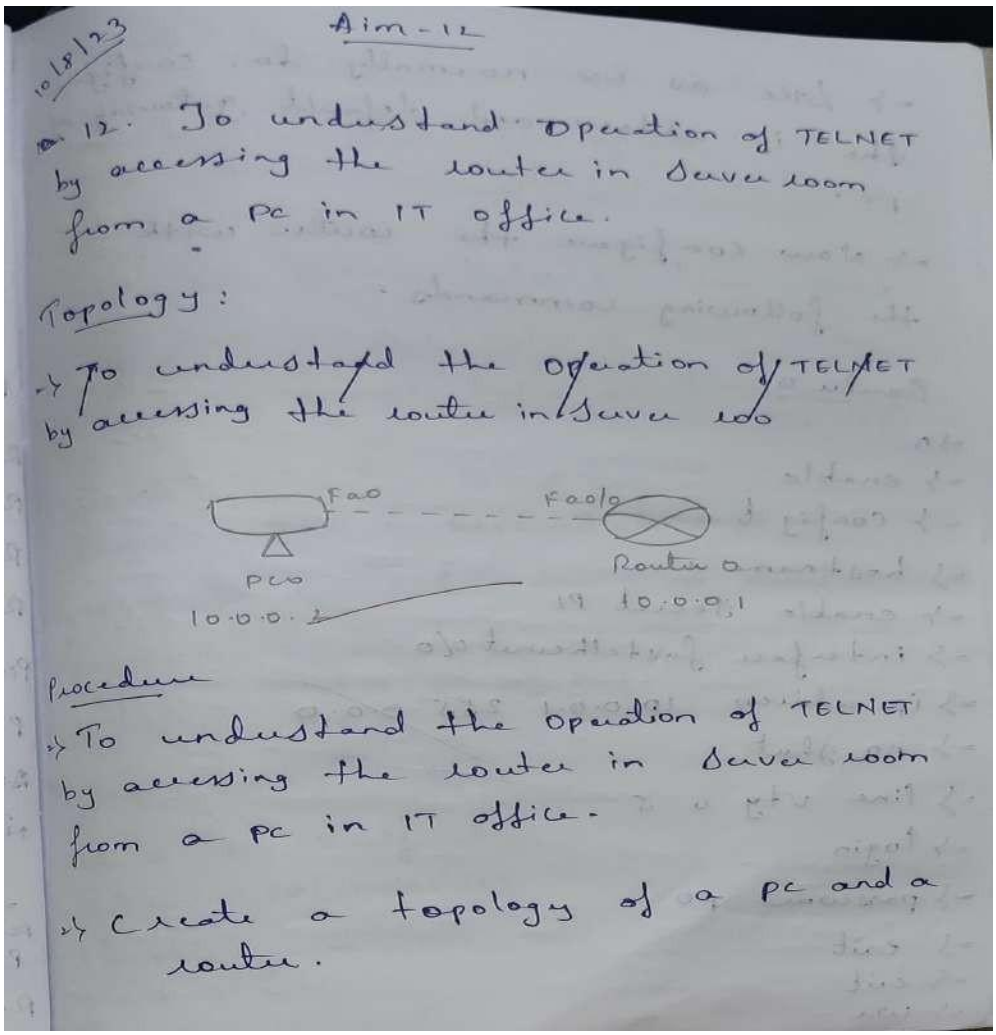


Fig 3: Router0 Configuration

Procedure and Observation:



-> Like as we normally do, configure the ip address and default gateway of pc.

-> Now configure the router with the following commands.

Router 0:

->n
-> enable
-> config t
-> hostname r1
-> enable secret p1
-> interface fastethernet 0/0
-> ip address 10.0.0.1 255.0.0.0
-> no shut
-> line vty 0 5
-> login
-> password p0
-> exit
-> exit
-> wr

Output

=> Pinging from PC0 to router

PC> Ping 10.0.0.1

Pinging 10.0.0.1 with 32 bytes of data:

Pinging from 10.0.0.1: bytes=32 time=0ms TTL=255

Pinging from 10.0.0.1: bytes=32 time=0ms TTL=255

Pinging from 10.0.0.1: bytes=32 time=0ms TTL=255

Pinging 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 =0 ms, Maximum=0ms, Average=0ms

(Command prompt of PC)

PC> telnet 10.0.0.1

Trying 10.0.0.1 .... Open

User Access Verification

Password: PC (invisible)

R1> enable

Password: P1 (invisible)

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

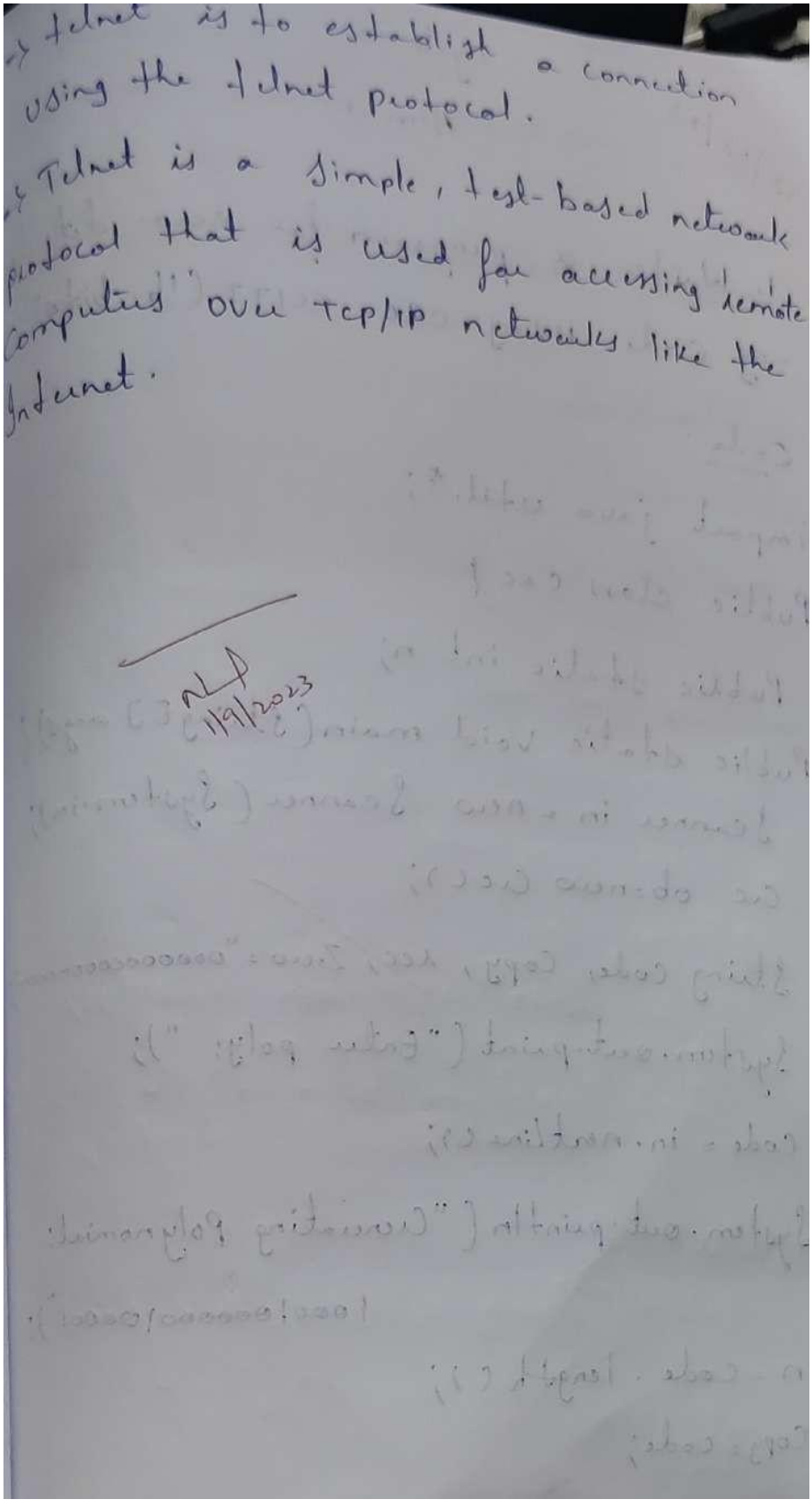
P - periodic download static route

Gateway of last resort is not set:

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

Observation

=> Telnet stands for Teletype Network, but it can also be used as a verb; to


-> Telnet is to establish a connection using the Telnet protocol.
-> Telnet is a simple, text-based network protocol that is used for accessing remote computers over TCP/IP networks like the Internet.
NLP
1/9/2023

Output:

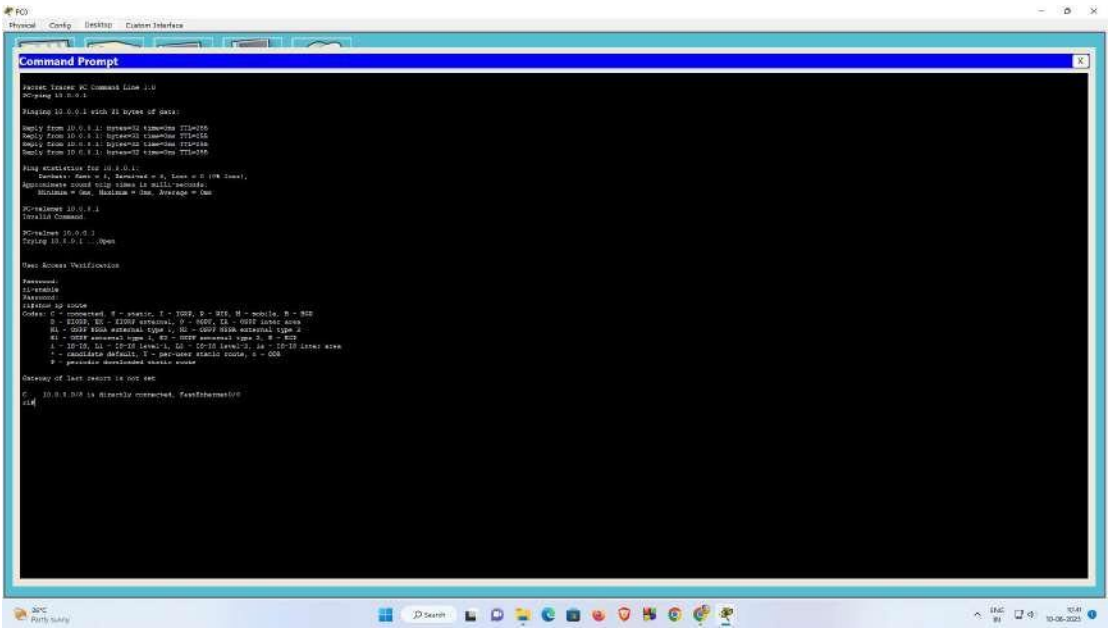


Fig 4: ping response and trying to access the device remotely

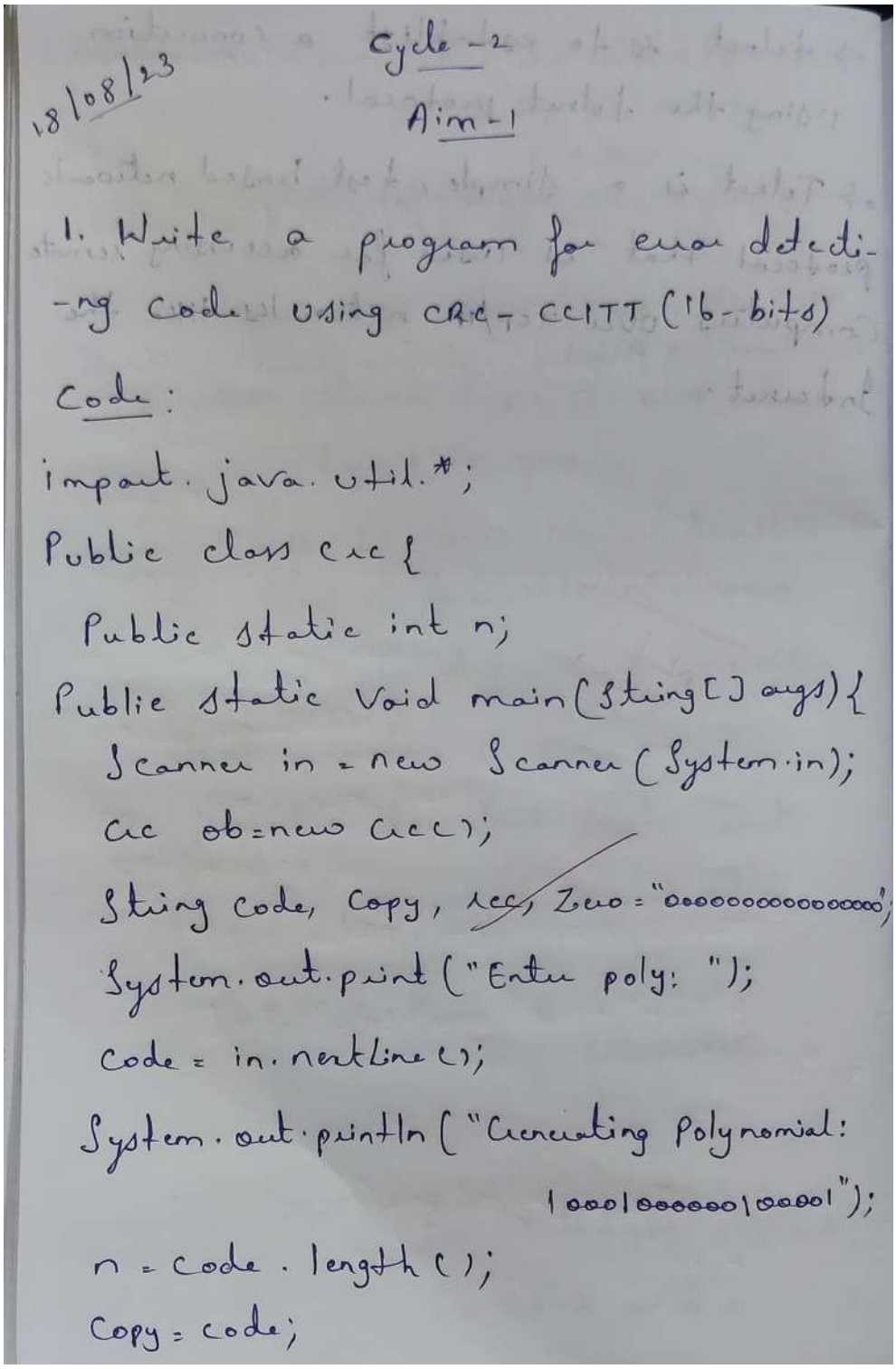
Experiment No. 13

Cycle 2

Aim-13

13. Write a program for error detecting code using CRCCCITT (16-bits).

Observation:



18/08/23

Cycle-2

Aim-1

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

Code:

```
impart.java.util.*;
Public class crc{
  Public static int n;
Public static Void main(Sting[] args){
   Scanner in = new Scanner(System.in);
   crc ob=new crc();
   Sting code, copy, recv, Zero="0000000000000000";
   System.out.print("Entu poly: ");
   code = in.nextLine();
   System.out.println("Generating Polynomial:
                                  10001000000100001");
   n = code.length();
   copy = code;
```

```
code+ = Zero;
System.out.println("Modified poly: " + code);
code = ob.divide(code);
System.out.println("checkSum: " + code.Su
                    bstring(n1));
copy = copy.Substring(0, n) + code.Substring(n);
System.out.println("Final Codeword: " + copy);
System.out.println("Test Error detection 0
                    (yes) 1(no)? : ");
int choice = in.nextInt();
if(choice == 0){
System.out.print("Enter position on error: ");
int errPos = in.nextInt();
if(copy.charAt(errPos) == '1')
  Copy = copy.Substring(0, errPos) + "0" + copy.
                    Substring(errPos+1);

Copy = copy.Substring(0, errPos) + "1" + copy.
                    Substring(errPos+1);
System.out.println("Erroneous data: " + copy);
```

```
System.out.println("Error detected");
}
else
System.out.println("No error detected");
}
Public string divide(string s){
int i,j;
char x;
String div="10001000000100001";
for(i=0; i<n; i++){
x=s.charAt(i);
for(j=0; j<7; j++){
if(x=='1'){
if(s.charAT(i+j)!=div.charAT(j))
g=s.substring(0,i+j)+"1"+s.substr
-ng(i+j+1);
else
g=s.substring(0,i+j)+"0"+s.substring
(i+j+1);
}
}
return 0;
}
```

Output 1:

Enter poly : 1000|0001|0001|0001

Generating polynomial : 1000|0000 0001|00001

Modified Poly : 1000|0001|0001|0000000000000000

CheckSum = 1000001|0101|0001

Final Codeword : 1000|0001|0001|0001 0000010|0|011
0001

Is it Error detection 0(yes) 1(No)? : 0

Enter position on error : 0

Erroneous data: 0000|0001|0001|0001|00000|0|011
0001

Error detected.

Output 2:

Enter poly: 0|0|0|0|0|0|0|0|0|0

Generating polynomial : 1000|000000|00001

Modified poly: 0|0|0|0|0|0|0|0|0|0|0000000000000
00

CheckSum : 1111|0110|0011|0|0

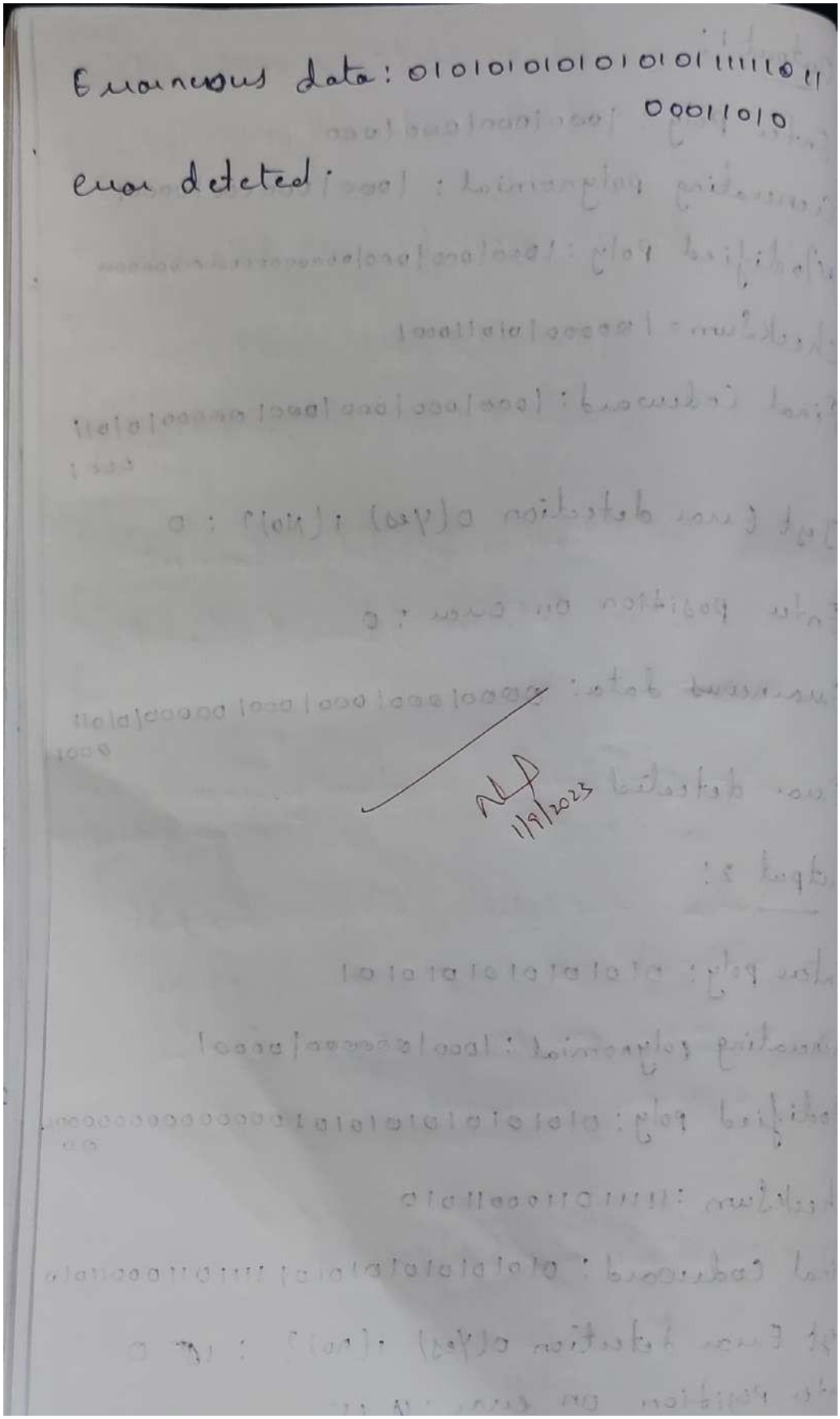
Final Codeword : 0|0|0|0|0|0|0|0|0|0|1111|0110|0011|0|0

Is it Error detection 0(Yes) 1(no)? : 1 0

Enter Position on error : 8 15

Erroneous data: 01010101010101011111011
00011010

error detected:



Code:

```
import java.util.*;
public class crc{
   public static int n;
   public static void main(String[] args){
      Scanner in=new Scanner(System.in);
      crc ob=new crc();
      String code, copy, rec,zero="0000000000000000";
      System.out.print("Enter poly: ");
      code=in.nextLine();
      System.out.println("Generating polynomial: 10001000000100001");
      n=code.length();
      copy=code;
      code+=zero;
      System.out.println("Modified poly: "+code);
      code=ob.divide(code);
      System.out.println("CheckSum: "+code.substring(n));
      copy=copy.substring(0,n)+code.substring(n);
      System.out.println("Final Codeword: "+copy);
      System.out.print("Test Error detection 0(yes) 1(no)? : ");
      int choice = in.nextInt();
      if(choice == 0){
         System.out.print("Enter position on error: ");
         int errorPos = in.nextInt();
         if(copy.charAt(errorPos) == '1')
            copy = copy.substring(0,errorPos) + "0" + copy.substring(errorPos+1);
         else
            copy = copy.substring(0,errorPos) + "1" + copy.substring(errorPos+1);
         System.out.println("Errorneous data: "+copy);
         System.out.println("Error detected");
      }
      else
```

```
            System.out.println("No Error detection");
        }
        public String divide(String s){
            int i,j;
            char x;
            String div="10001000000100001";
            for(i=0;i<n;i++){
                x=s.charAt(i);
                for(j=0;j<17;j++){
                    if(x=='1'){
                        if(s.charAt(i+j)!=div.charAt(j))
                            s=s.substring(0,i+j)+"1"+s.substring(i+j+1);
                        else
                            s=s.substring(0,i+j)+"0"+s.substring(i+j+1);
                    }
                }
            }
            return s;
        }
}
```

Output:

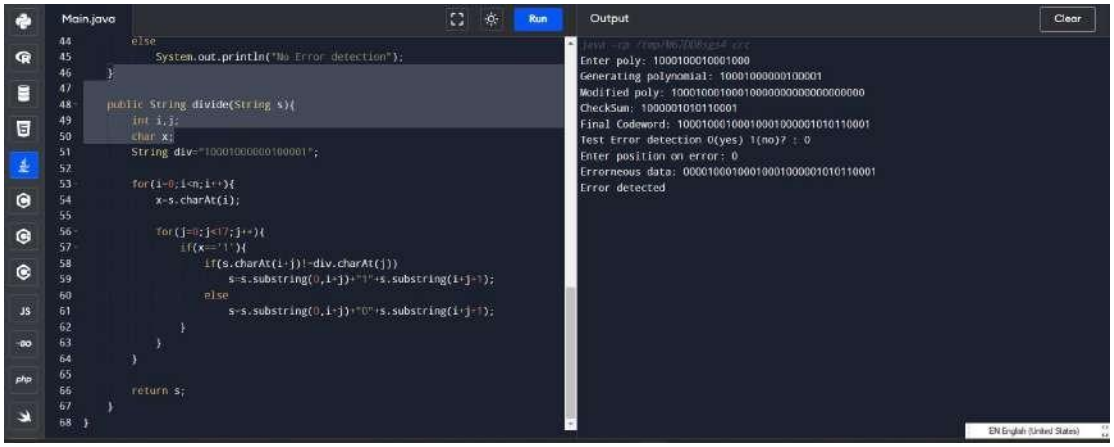


Fig: Output 1 (poly 1)

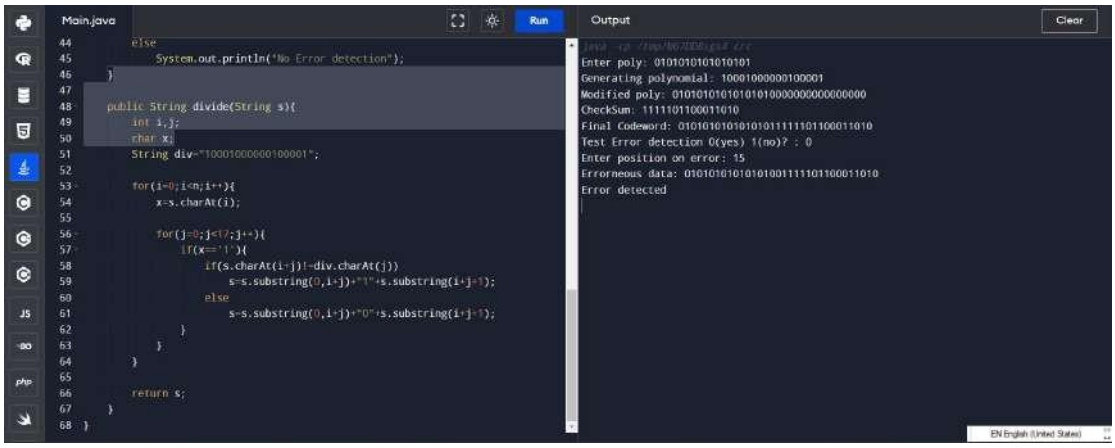


Fig: Output 2 (poly 2)

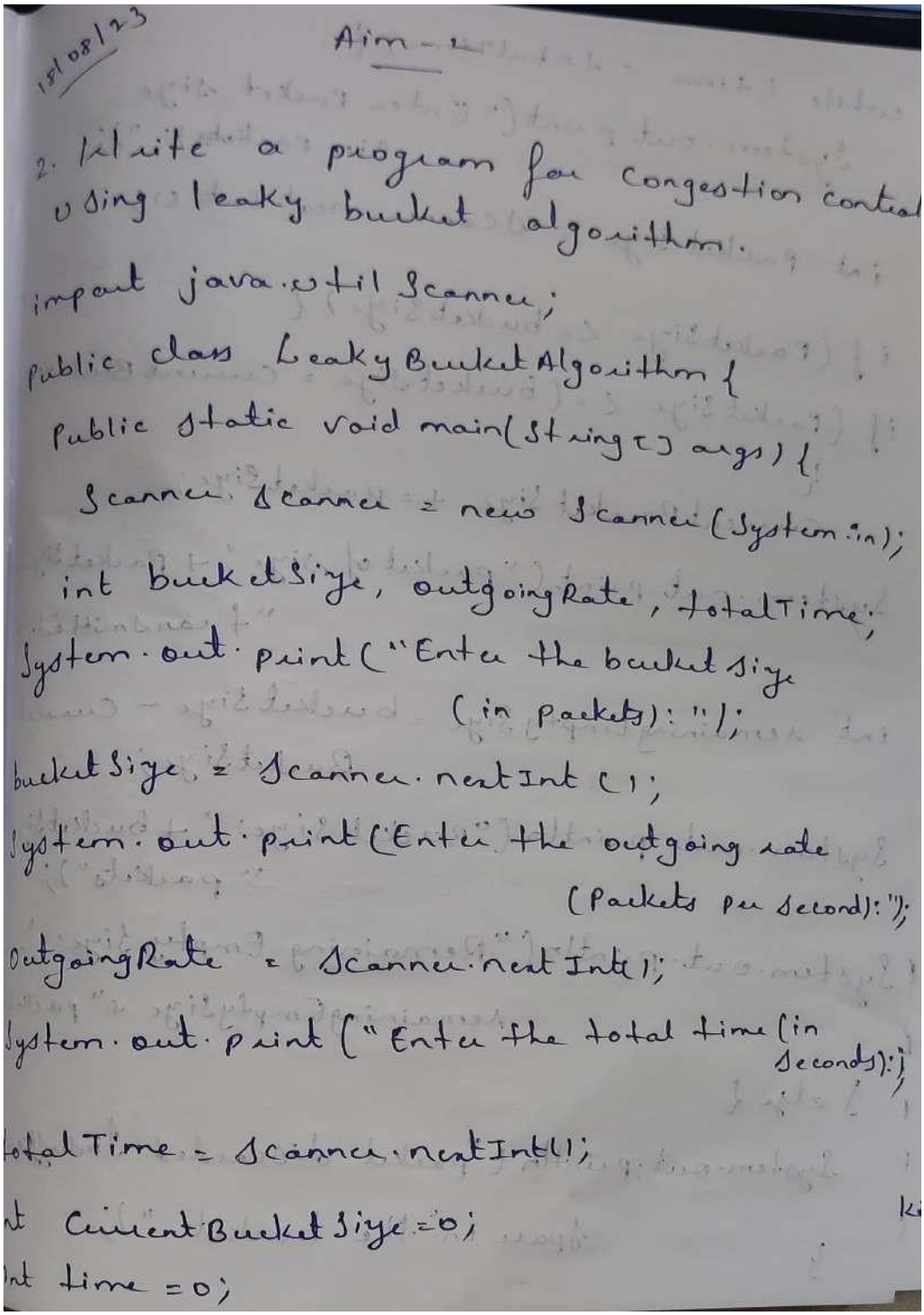
Experiment No. 14

Cycle 2

Aim-14

14. Write a program for congestion control using Leaky bucket algorithm.

Observation:



```
while (time < totalTime) {
    System.out.print("Enter Packet size
                        (in Packets): ");
    int packetSize = Scanner.nextInt();

    if (PacketSize <= bucketSize) {
    if (PacketSize <= (bucketSize - currentBucketSize))
        {
        currentBucketSize += packetSize;
    System.out.print("packet of size " + PacketSize+
                        " transmitted.");
    int remainingEmptySize = bucketSize - Current
                        BucketSize;
    System.out.println("Bucket Size: " + BucketSize+
                        " packets");
    System.out.println("Remaining Empty Size: "+
                        remainingEmptySize +" packets");
    } else {
        System.out.println("packet discarded: Insufficient
                        Space in the bucket.");
    }
```

```
} else {
    System.out.println("Packet discarded: Exceeds
                        the bucket size.");
}
time++;
}
scanner.close();
}
}
```

Output 1:

11/9/2023

```
Enter the bucket size (in packets): 5
Enter the outgoing rate (Packets per second): 2
Enter the total time (in seconds): 5
Enter the packet size (in packets): 3
Packet of size 3 transmitted. Bucket size: 5 packets
Remaining Empty size: 2 packets
Enter packet size (in packets): 10
Packet discarded: Exceeds the bucket size.
Enter packet size (in packets): 3
Packet discarded: Insufficient space in the bucket
```

Enter packet size (in packets): 2

packets of size 2 transmitted.

Bucket size: 5 packets

Remaining Empty size = 0 packets.

Output 2:

Enter the bucket size (in packets): 10

Enter the outgoing rate (packets per second): 2

Enter the total time (in seconds): 5

Enter packet size (in packets): 3

Packet of size 3 transmitted

Bucket size: 10 packets

Remaining Empty size: 7 packets

Enter packet size (in packets): 5

Packet of size 5 transmitted

Bucket size: 10 packets

Remaining Empty size: 2 packets

Enter packet size (in packets): 4

packet discarded: Insufficient space in the bucket

Enter packet size (in packets): 12
packet discarded: Exceeds the bucket size
Enter packet size (in packets): 2
Packet of size 2 transmitted.
Bucket size = 10 packets
Remaining Empty size: 0 packets.

11/9/2023

Code:

```
import java.util.Scanner;
public class LeakyBucketAlgorithm {
public static void main(String[] args) {
  Scanner scanner = new Scanner(System.in);
  int bucketSize, outgoingRate, totalTime;
  System.out.print("Enter the bucket size (in packets): ");
  bucketSize = scanner.nextInt();
  System.out.print("Enter the outgoing rate (packets per second): ");
  outgoingRate = scanner.nextInt();
  System.out.print("Enter the total time (in seconds): ");
  totalTime = scanner.nextInt();
  int currentBucketSize = 0;
  int time = 0;
  while (time < totalTime) {
     System.out.print("Enter packet size (in packets): ");
     int packetSize = scanner.nextInt();
              if (packetSize <= bucketSize) {
        if (packetSize <= (bucketSize - currentBucketSize)) {
         currentBucketSize += packetSize;
        System.out.println("Packet of size " + packetSize + "transmitted.");
         int remainingEmptySize = bucketSize - currentBucketSize;
            System.out.println("Bucket Size: " + bucketSize +“packets");
         System.out.println("Remaining Empty Size: "+remainingEmptySize + " packets");
         } else {
            System.out.println("Packet discarded: Insufficient space in the bucket.");
         }
     } else {
        System.out.println("Packet discarded: Exceeds the bucket size.");
     }

   time++;
  }

  scanner.close();
 }
}
```

Output:

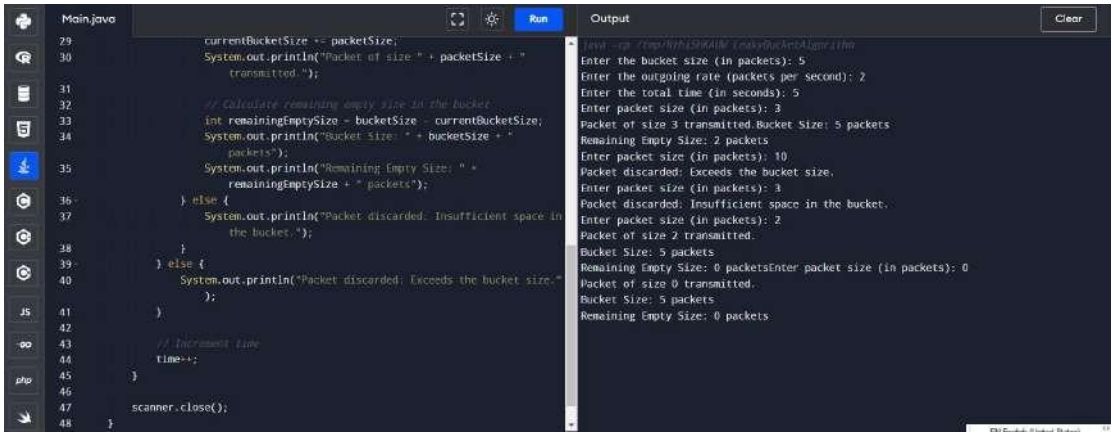


Fig1: Output 1

Fig 2: Output 2

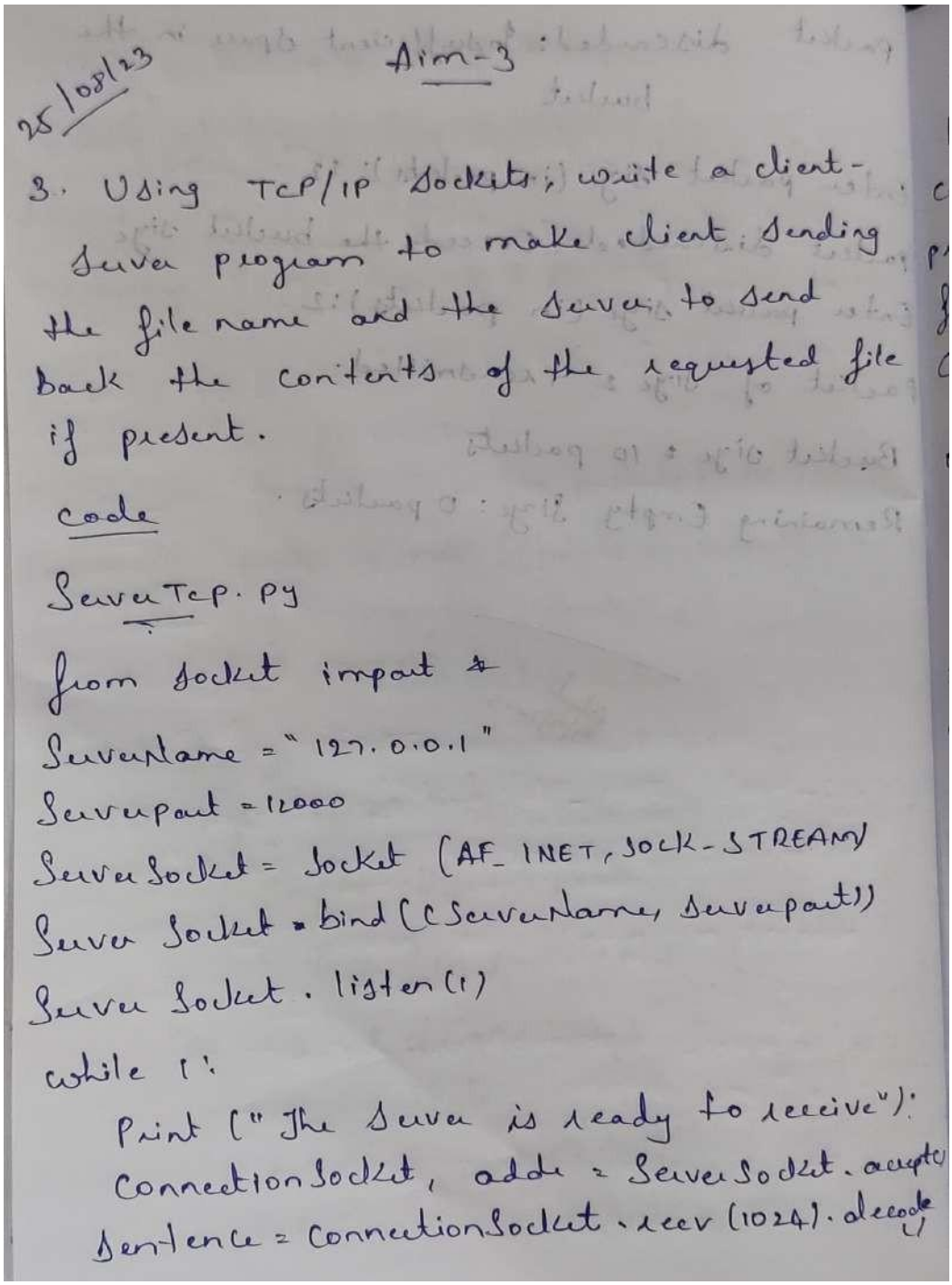
Experiment No. 15

Cycle 2

Aim-15

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

Observation:



25/08/23

Aim-3

3. 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.

code

ServerTcp.py

```
from socket import *
Servername = "127.0.0.1"
Serverport = 12000
ServerSocket = socket (AF_INET, SOCK_STREAM)
Server Socket = bind ((Servername, Serverport))
Server Socket . 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

```
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 Serv:\n')
print(filecontents)
clientSocket.close()
```

11/9/2021

Code:

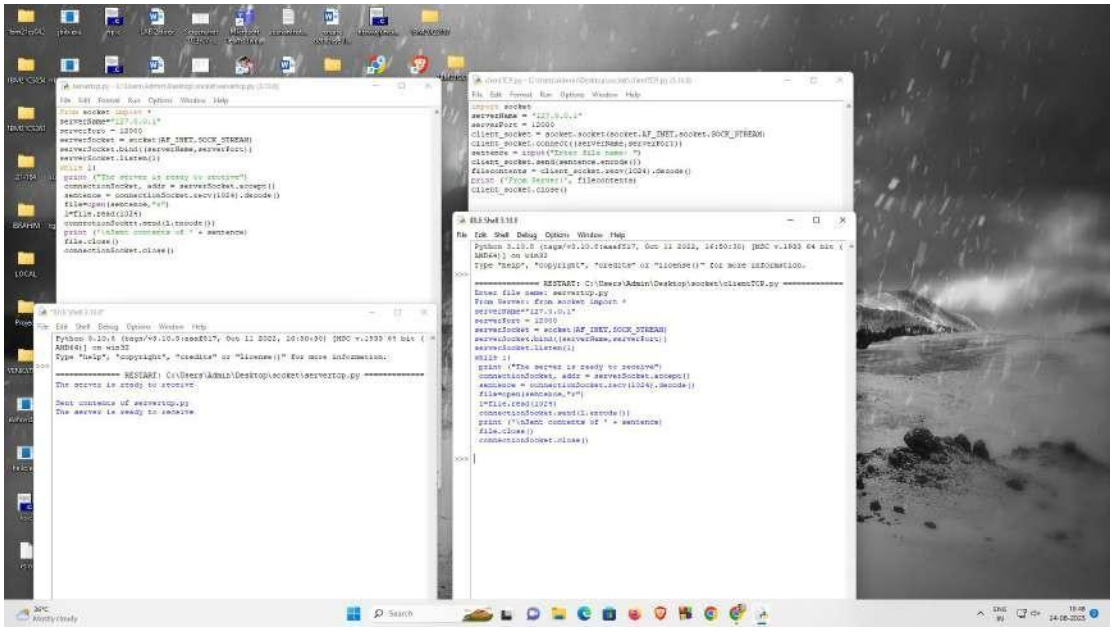
Server Tcp:

```
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()
```

Client Tcp:

```
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()
```

Output:



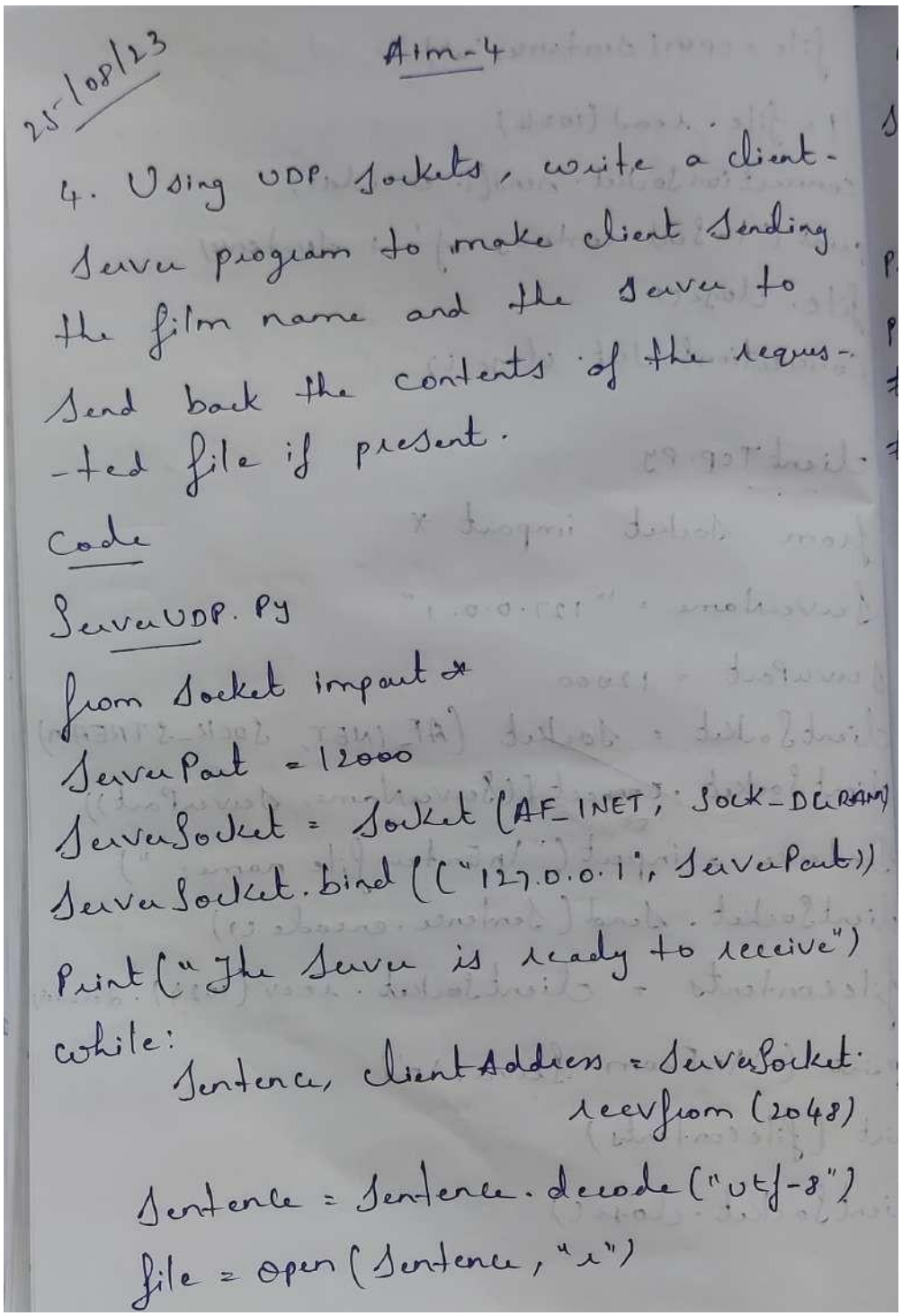
Experiment No. 16

Cycle 2

Aim-16

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

Observation:



25/10/23

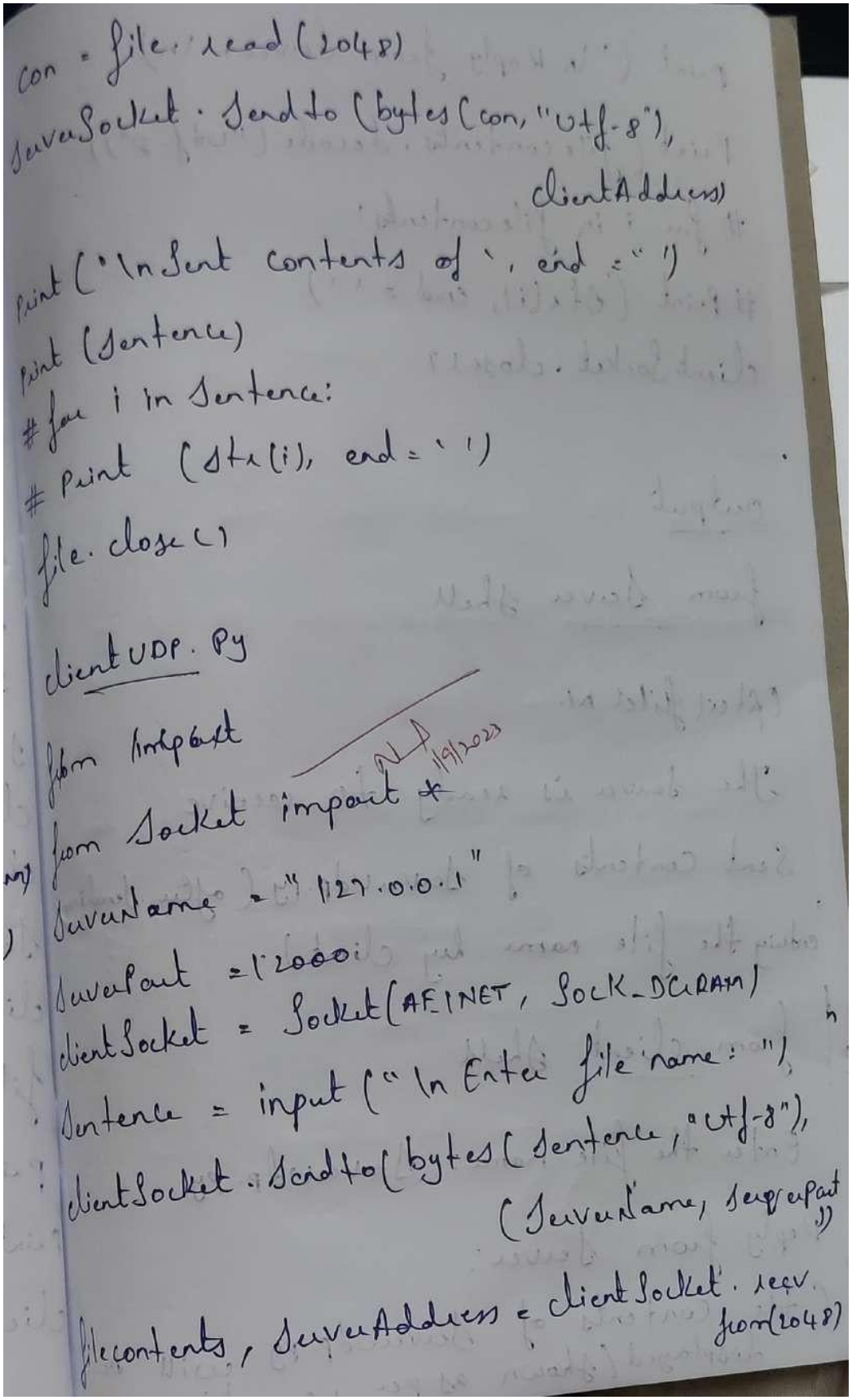
Aim-4

4. 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:
    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()
```

clientUDP.py

```
from tkinter
from socket import *
```

19/2023

```
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.recv.
                              from(2048)
```

```
print ('\n Reply from Server : \n')
Print (filecontents.decode ("utf-8"))
# for i in filecontents:
# Print (str(i), end = '')
clientSocket.close()
```

Output

from Server Shell

Enter file name

The Server is ready to receive

Sent contents of ServerUdp.py (after sending entering the file name by client).

from client shell

Enter the file name: ServerUdp.py

Reply from Server:

(the contents of ServerUdp.py will be displayed / shown as per request).

Code:

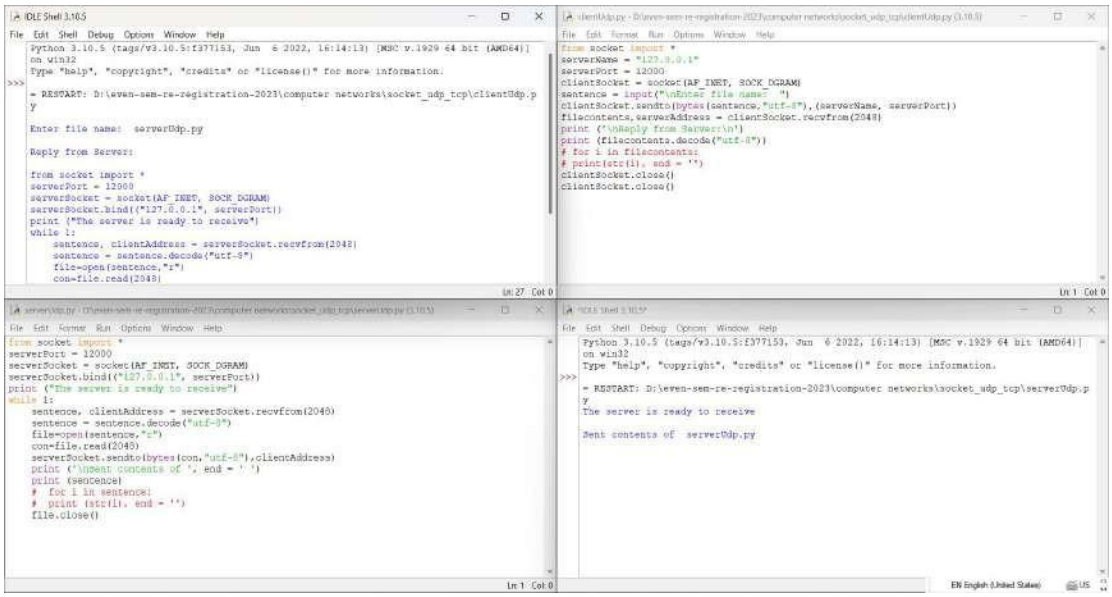
Server Udp:

```
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()
```

Client Udp:

```
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()
```

Output:

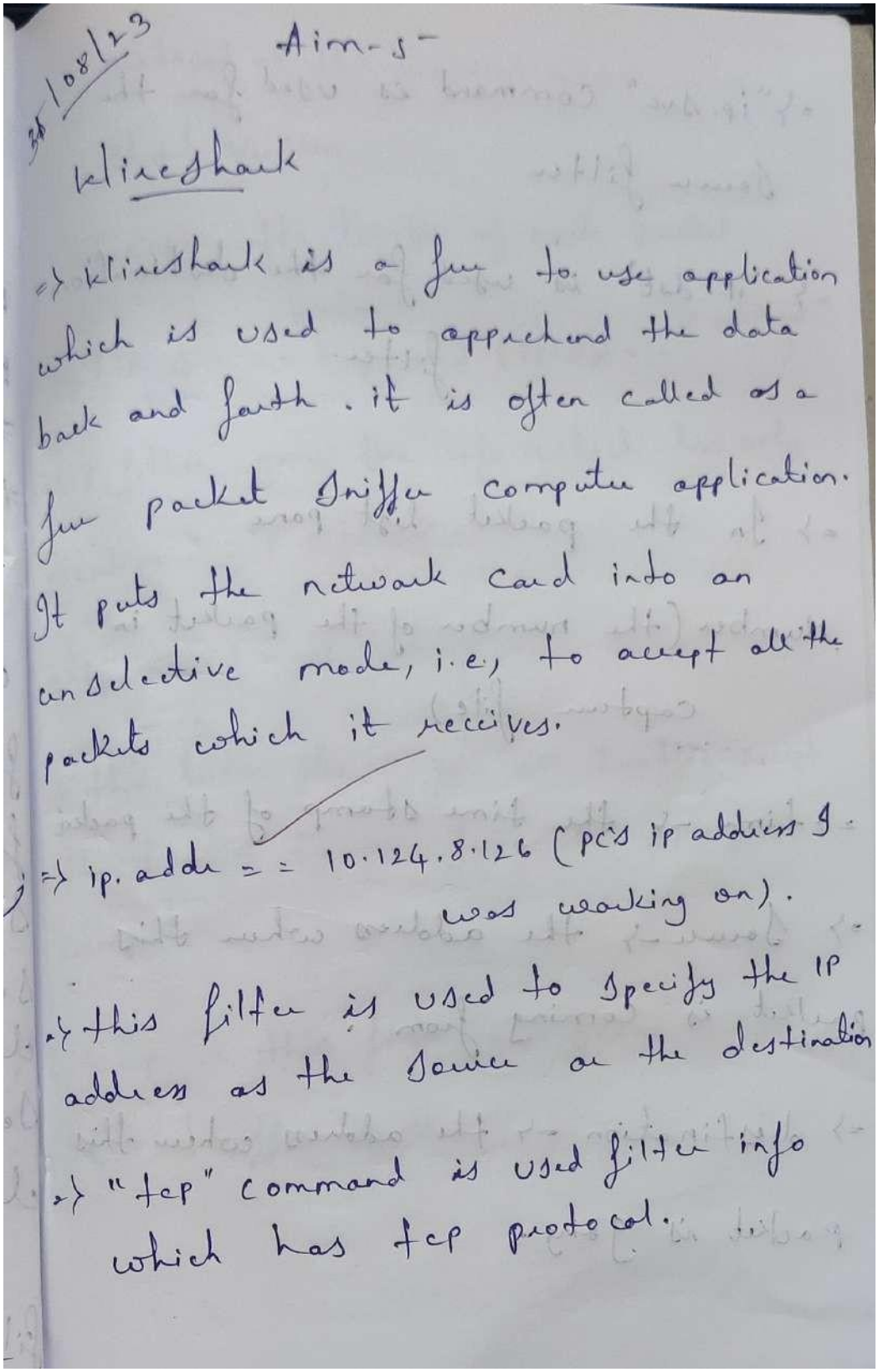

IDLE Shell 3.10.5
File Edit Shell Debug Options Window Help
Python 3.10.5 (tags/v3.10.5:f377153, Jun 6 2022, 16:14:13) [MSC v.1929 64 bit (AMD64)]
on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: D:\even-sem-re-registration-2023\computer networks\socket_udp_tcp\clientUdp.py
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)
Ln: 27 Col: 0
clientUdp.py - D:\even-sem-re-registration-2023\computer networks\socket_udp_tcp\clientUdp.py (3.10.5)
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()
Ln: 1 Col: 0
server\Udp.py - D:\even-sem-re-registration-2023\computer networks\socket_udp_tcp\server.Udp.py (3.10.5)
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()
Ln: 1 Col: 0
IDLE Shell 3.10.5
File Edit Shell Debug Options Window Help
Python 3.10.5 (tags/v3.10.5:f377153, Jun 6 2022, 16:14:13) [MSC v.1929 64 bit (AMD64)]
on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: D:\even-sem-re-registration-2023\computer networks\socket_udp_tcp\serverUdp.py
The server is ready to receive
Sent contents of serverUdp.py
EN English (United States)
US

Experiment No. 17

Cycle 2

Aim-17

17. Wireshark Report



=> "ip.src" command is used for the Source filter

=> "ip.dst" is used for the destination filter.

=> In the packet list pane

Number (the number of the packet in capture file)

=> time -> the time stamp of the packet

=> Source -> the address where this packet is coming from.

=> destination -> the address where this packet is going.

-> protocol -> the protocol name in a short version.

=> length - the length of each packet

=> (ip.src == ipaddress) and (ip.dst == ip address)

this filter gives the info which has only mentioned ip. addresses.

-> ip.addr != ip address

NLP 11/9/2023

=> this filter shows you all traffic except the traffic to or from the specified computer.

=> icmp -> this filter will show you only icmp traffic in the capture.