

VISVESVARAYA TECHNOLOGICAL UNIVERSITY
“JnanaSangama”, Belgaum -590014, Karnataka.



LAB REPORT
on
COMPUTER NETWORKS

Submitted by

IMADH AJAZ BANDAY (1BM20CS059)

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
October-2022 to Feb-2023

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



CERTIFICATE

This is to certify that the Lab work entitled "**COMPUTER NETWORKS**" carried out by **IMADH AJAZ BANDAY (1BM20CS059)**, 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 2022. The Lab report has been approved as it satisfies the academic requirements in respect of a **COMPUTER NETWORKS - (20CS5PCCON)** work prescribed for the said degree.

Dr. Nandini Vineeth
Assistant Professor
Department of CSE
BMSCE, Bengaluru

Dr. Jyothi S Nayak
Professor and Head
Department of CSE
BMSCE, Bengaluru

Index

Sl. No.	Date	Experiment Title	Page No.
Cycle - 1			
0.	3-11-22	My First PT Lab.	1-5
1.	10-11-22	Creating a topology and simulate sending a simple PDU from source to destination using hub and switch as connecting devices.	6-18
2.	17-11-22	Configuring IP address to Routers in Packet Tracer. Explore the following messages: Ping Responses, Destination unreachable, Request timed out, Reply	19-31
3.	24-11-22	Configuring default route to the Router	32-42
4.	1-12-22	Configuring DHCP within a LAN in a packet Tracer	43-51
5.	8-12-22	Configuring RIP Routing Protocol in Routers	52-60
6.	15-12-22	Demonstration of WEB server and DNS using Packet Tracer	61-67
Cycle - 2			
7.	29-12-22	Write a program for error detecting code using CRC-CCITT (16-bits).	68-72
8.	5-1-23	Write a program for distance vector algorithm to find suitable path for transmission.	73-75
9.	12-1-23	Implement Dijkstra's algorithm to compute the shortest path for a given topology.	76-78
10.	12-1-23	Write a program for congestion control using Leaky bucket algorithm.	79-81
11.	27-1-23	Using TCP/IP sockets, write a client-server program to make client sending the file name and the server to send back the contents of the requested file if present.	82-85
12.	27-1-23	Using UDP sockets, write a client-server program to make client sending the file name and the server to send back the contents of the requested file if present.	86-90

Lab: Week 5

(Cisco Packet Tracer)

↳ Interface Overview:

The ten main components of the main interface are as follows:

1. Menu Bar
2. Main Tool Bar
3. Common Tools Bar
4. Logical / Physical Workspace and Navigation Bar
5. Workspace
6. Realtime / Simulation Bar
7. Network Component Bar
8. Device - Type Selection Bar
9. Device - Specific Selection Bar
10. User Created Packet Window

The packet tracer has two workspaces (Logical and Physical) and two modes (Realtime and Simulation).

We can switch between the Physical Workspace and Logical Workspace with the tabs on this bar.

In the logical workspace, we can switch between various options like creating a New Cluster, New Object, Set Titled Background and Viewport.

While as in the physical workspace, the bar will allow us to navigate through various spaces and locations, like creating a new city, Home, Corporate office or a New Building. We can even move objects and set some background and go to the working closet.

The two modes available in Packet Tracer Software include the Realtime and Simulation.

We can toggle between real-time and simulation modes.

At the beginning, the Packet Tracer we work in real-time mode in which the networking protocols work in real-time.

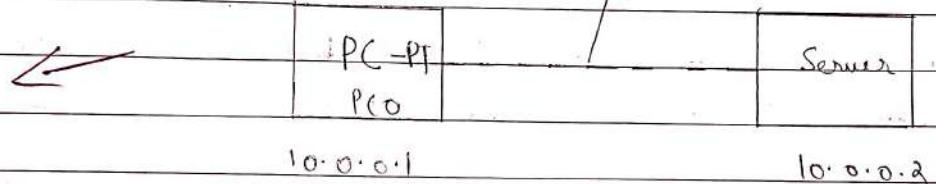
Using the simulation mode, we can see packets flowing from one node to another and can also click on a packet to see detailed information categorized by OSI layers. Using the real-time / simulation tabs we can switch from one mode to another.

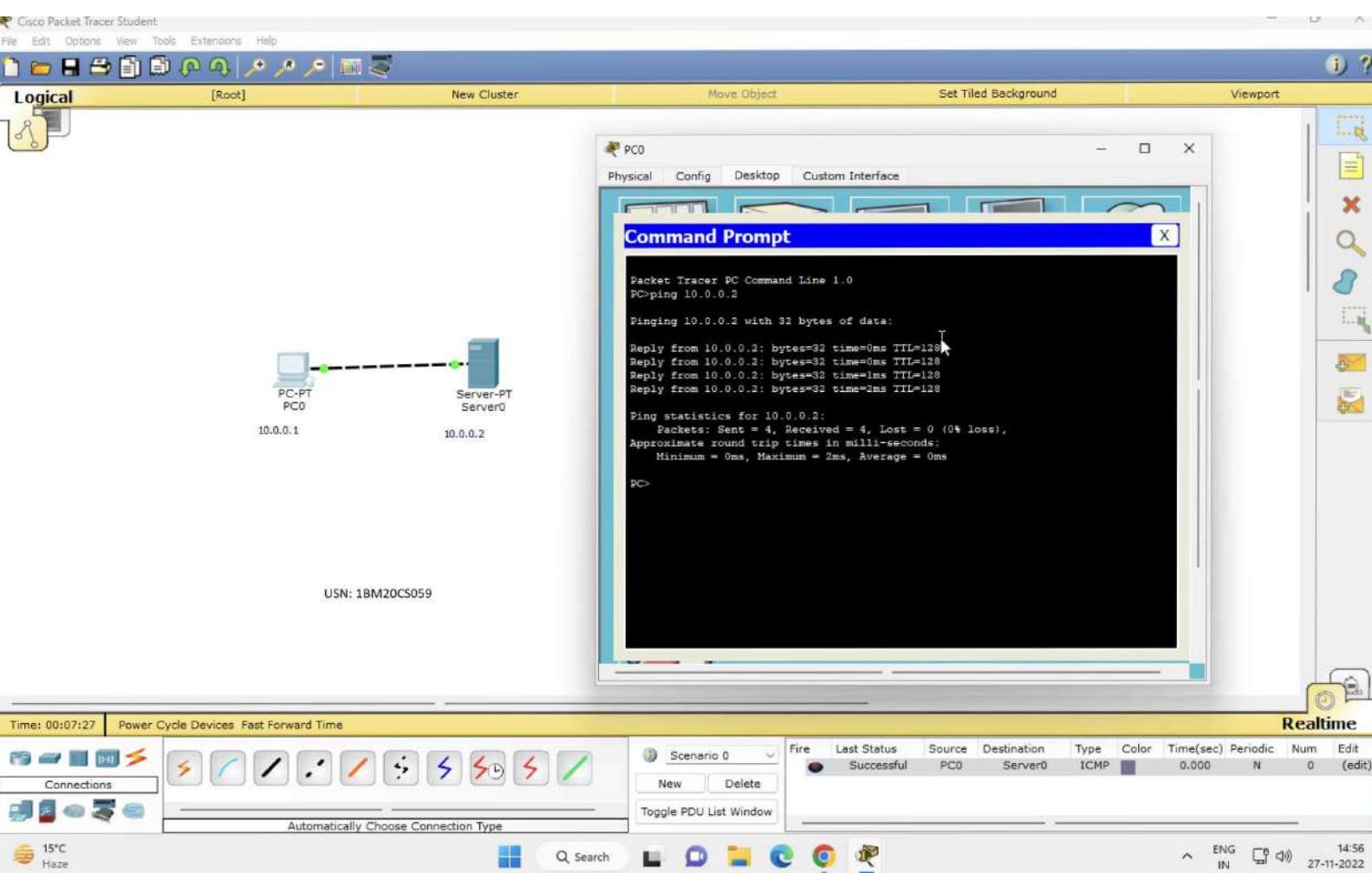
↳ My First PT lab:

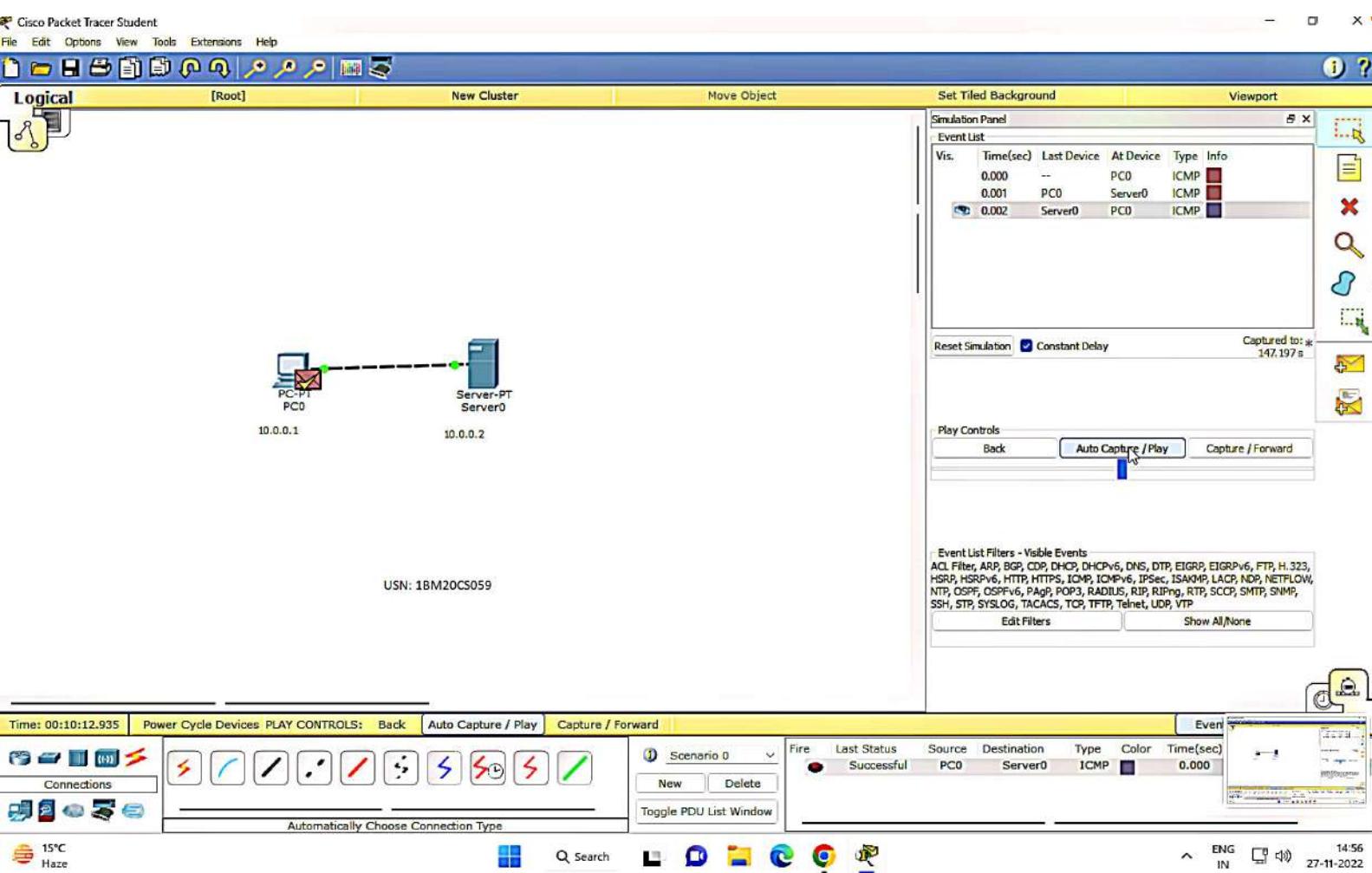
1. Launch Packet tracer.
2. Creating first network with the help of a generic PC and a generic server.
3. Under connection, select copper straight cable and connect PC & server.
4. Configure IP addresses.
5. Select simple PDU, and click on both devices.
6. Finally click on Auto Capture / Play & hence animation can be viewed of the Packet tracer in simulation mode.
7. In real time mode, open command-prompt and send ping using commands & destination IP address.

Topology:

Copper over fiber.







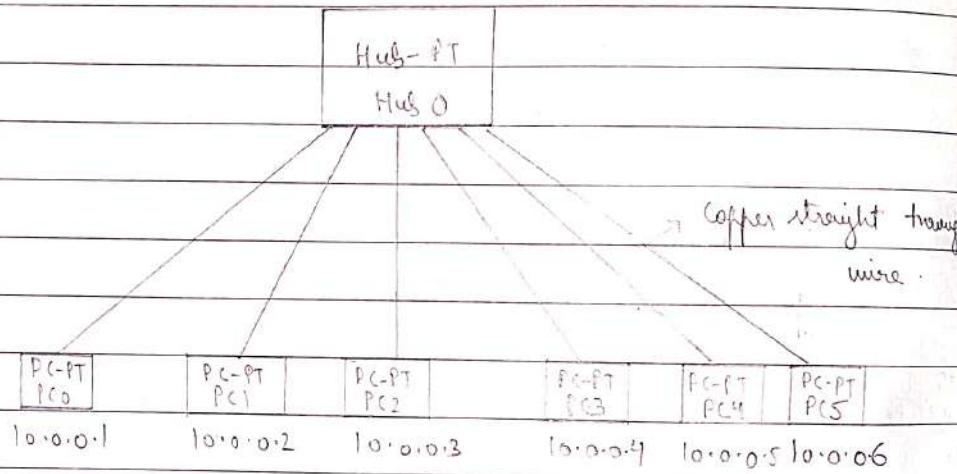
(Lab: Week 1)

(Hubs and switches)

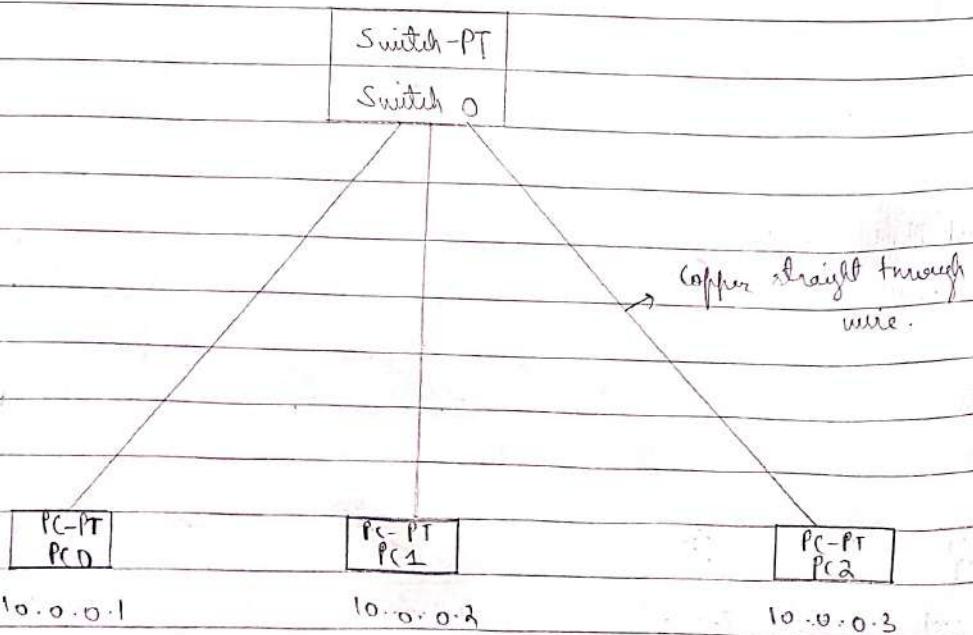
↳ Aim: Creating a topology and simulate sending a simple PDU from source to destination using simple hub and switch as connecting devices.

↳ Topology:

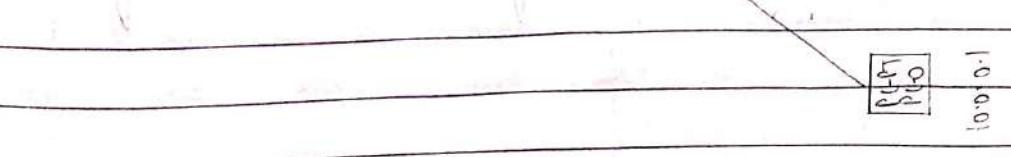
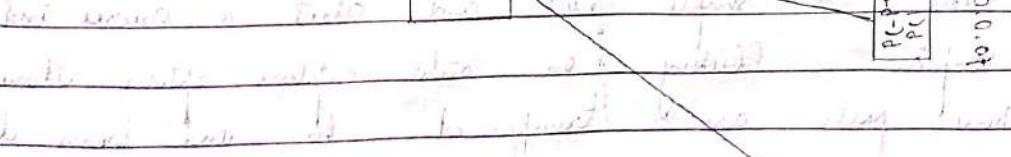
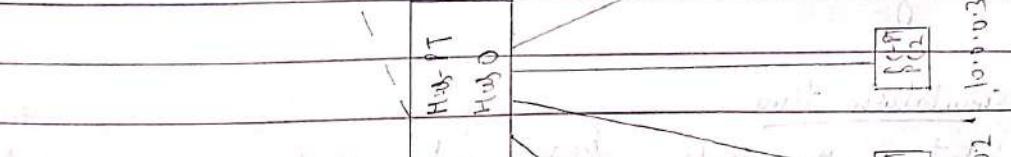
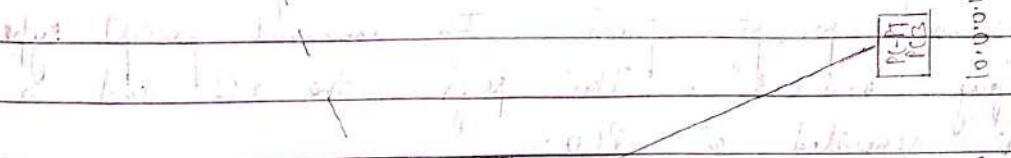
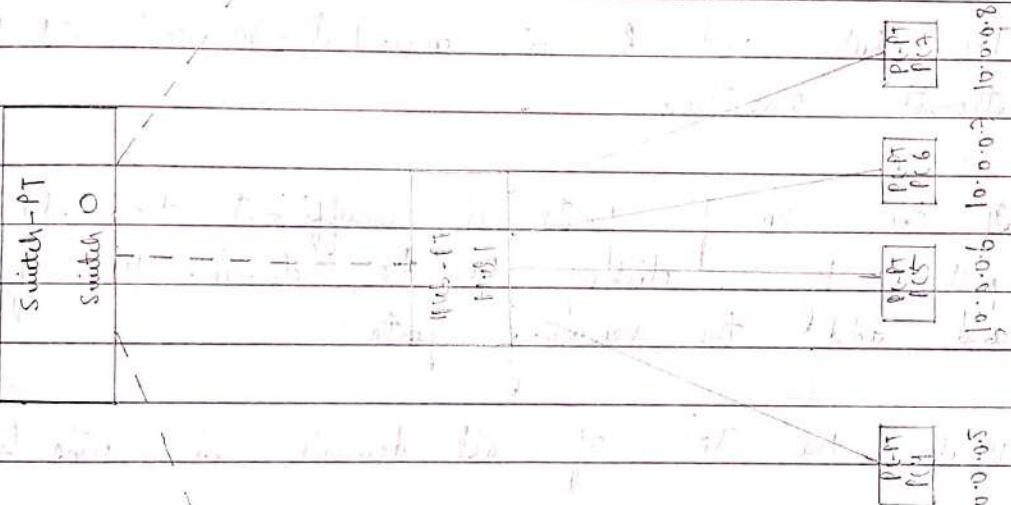
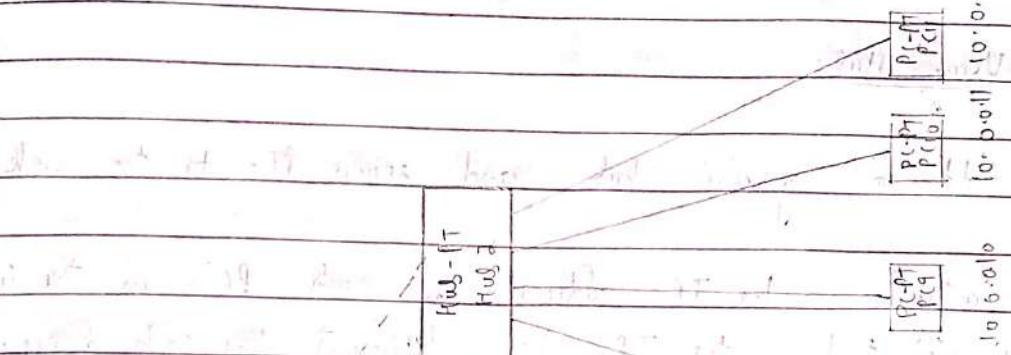
→ Using Hub:

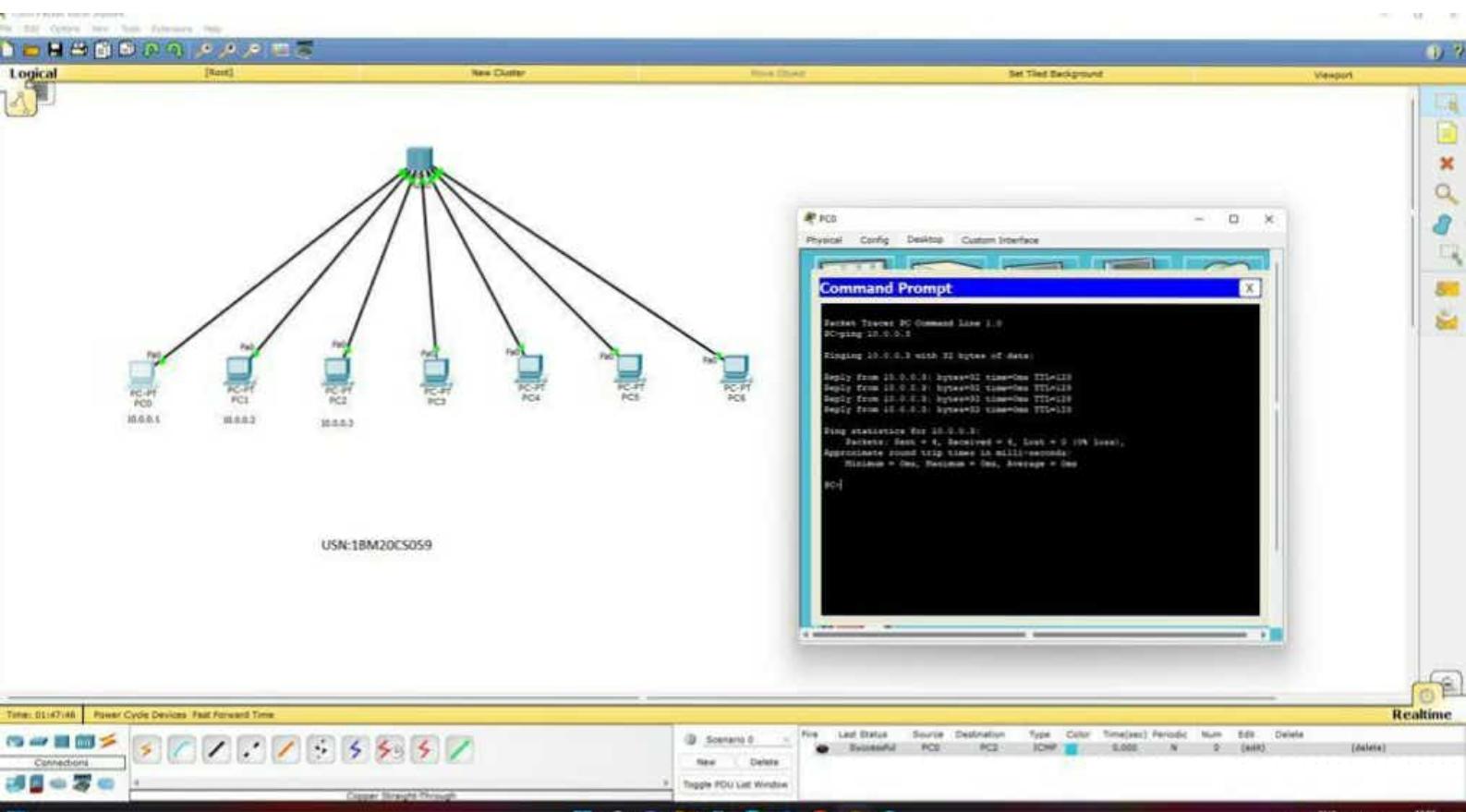


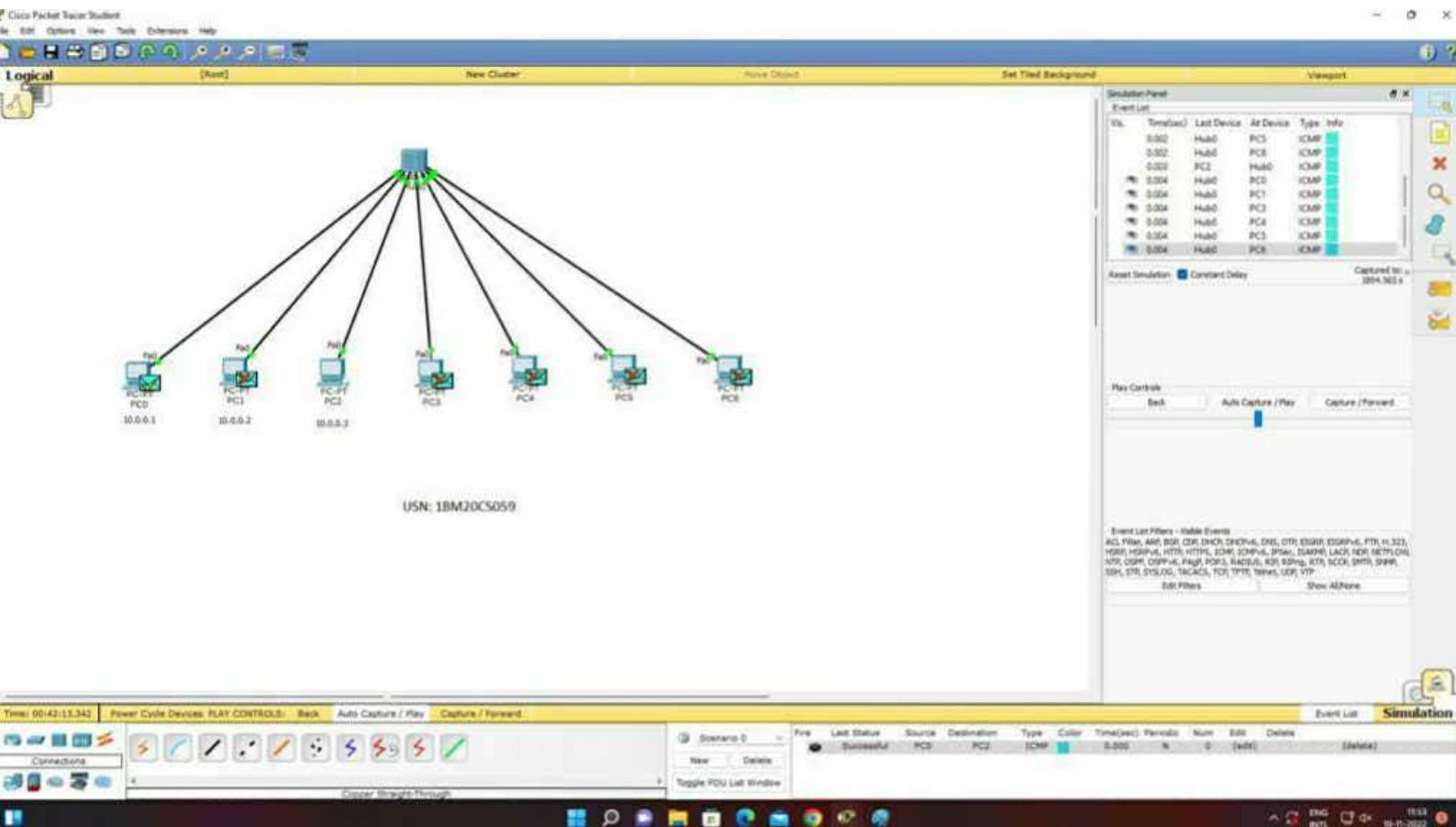
→ Using Switch

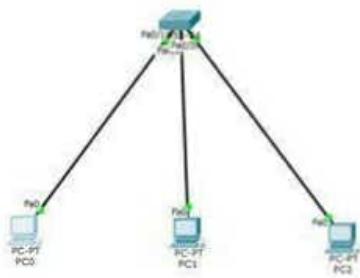


→ Hybrid (Using hubs and switches)

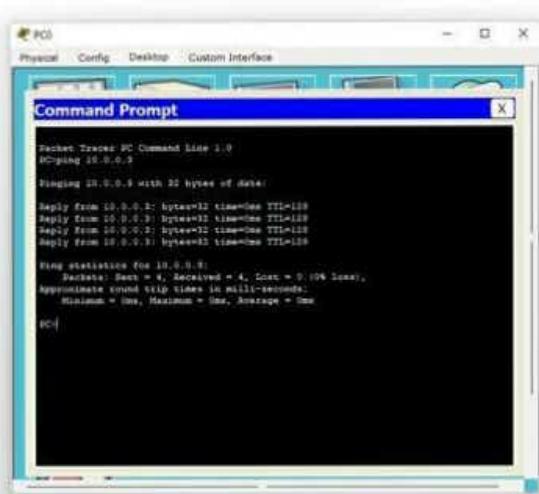


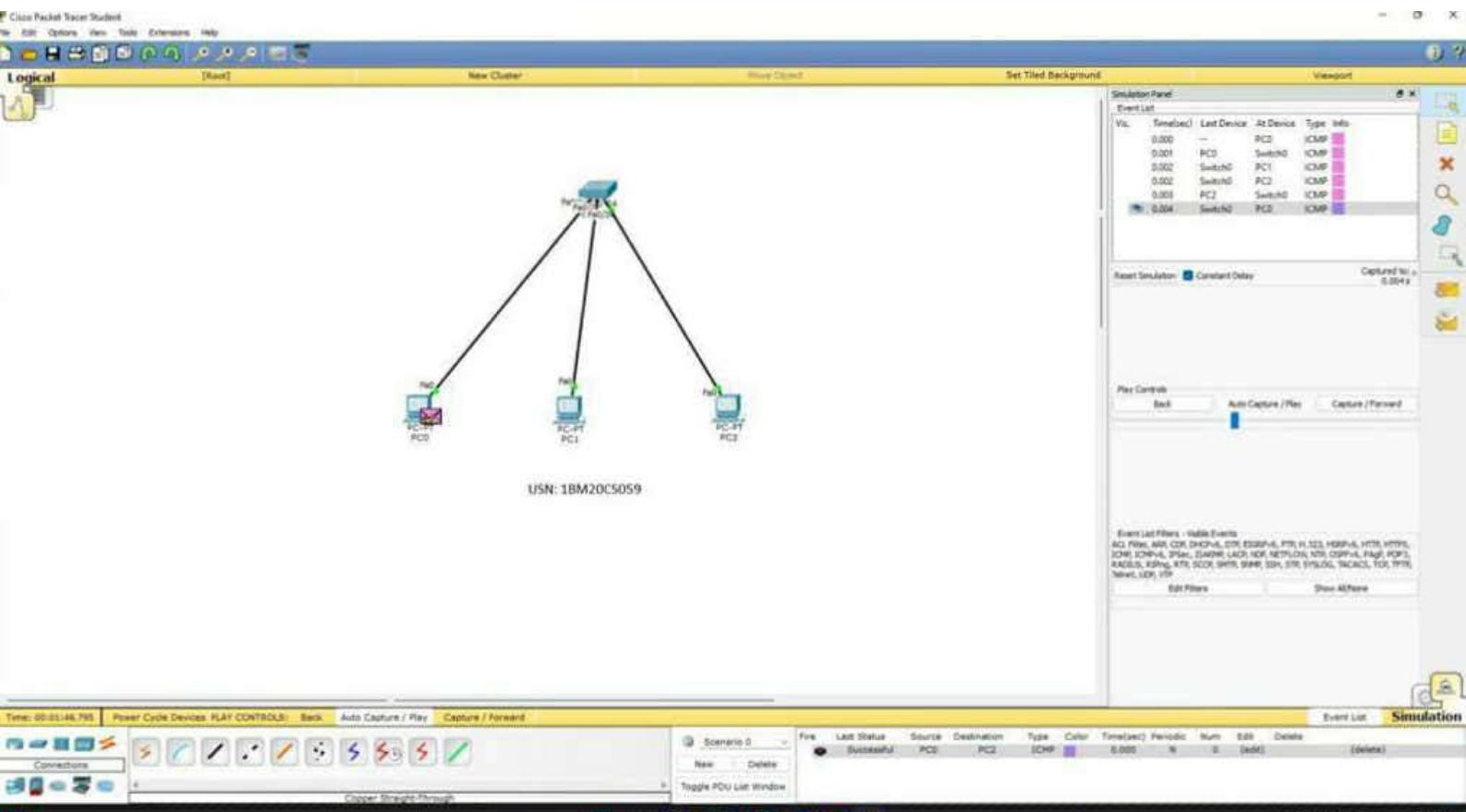


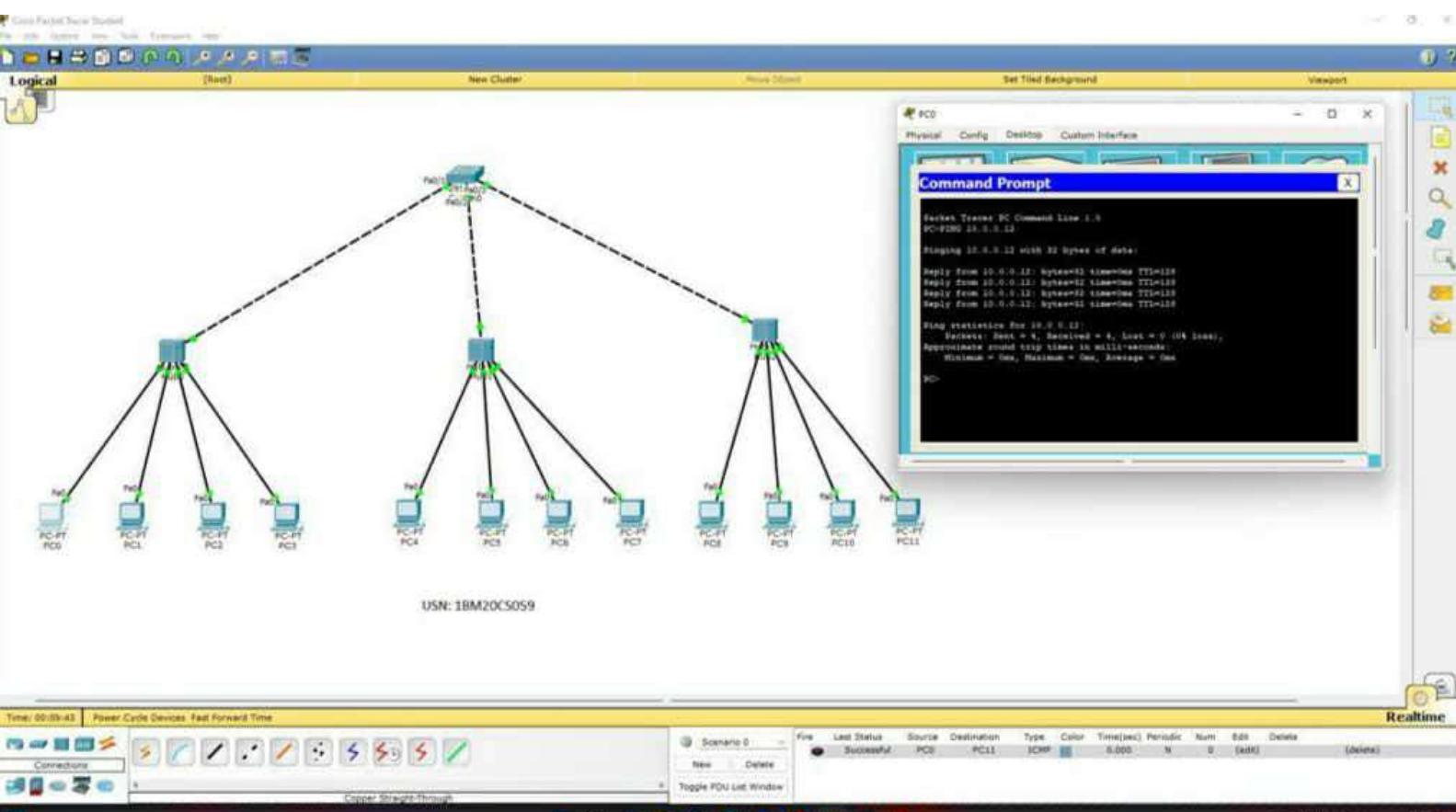


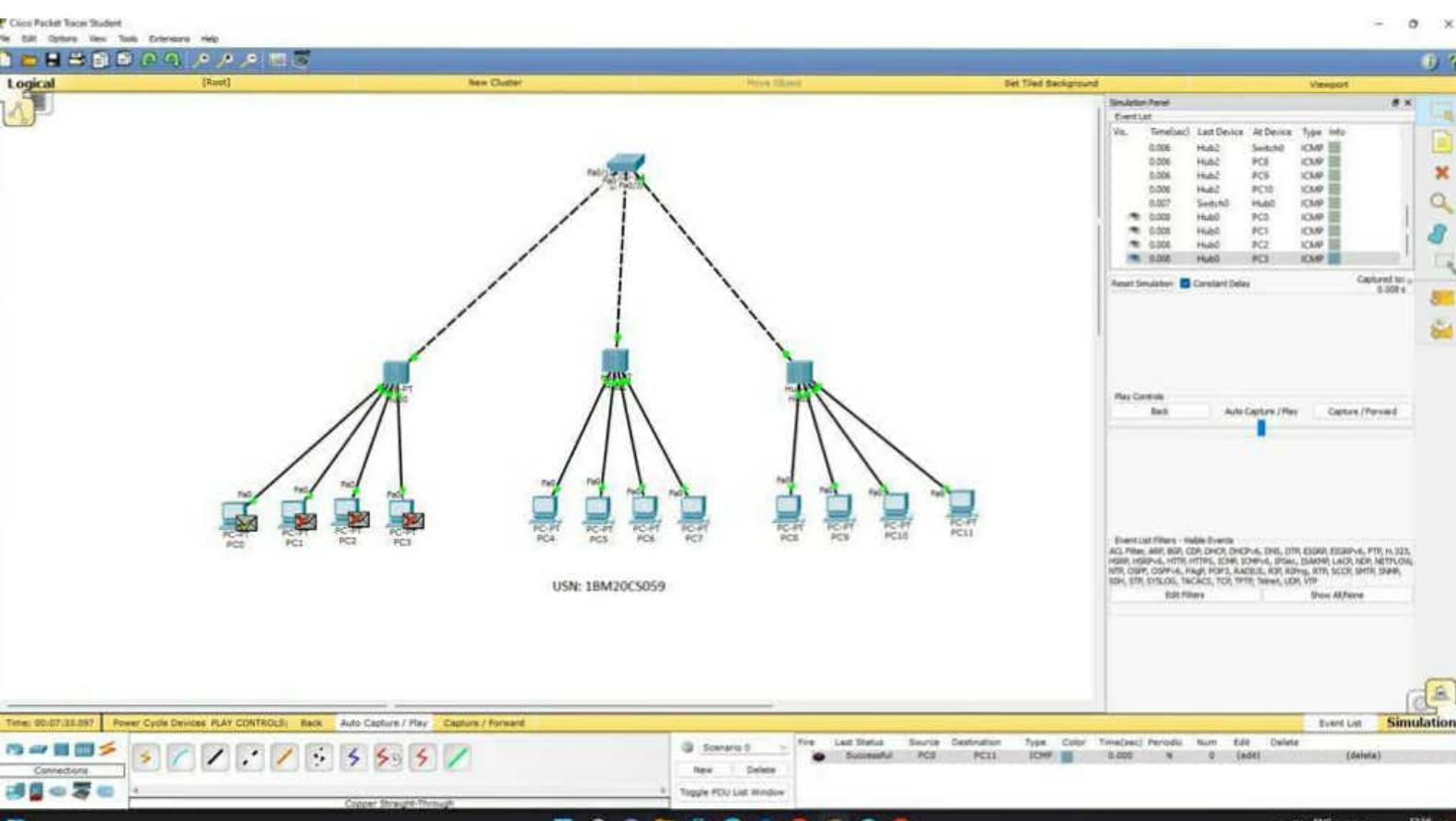


USN: 1BM20CS059









4 Procedure:

→ Using Hub:

- i) Add a generic hub and seven PCs to the workspace.
- ii) Configure the IP address of each PC's in the configurations. Ensure that the IP is different for each device.
- iii) Connect all PCs to hub using copper straight wire.
- iv) The hub and PC is connected to each other's fast ethernet connection.
- v) If the no. of ports is insufficient then add extra port by clicking on the device. Turn off the device and add the necessary ports.
- vi) Write the IP's of all devices in a note below the device.

Real time:

Select a source PC and in the desktop tab select the command prompt option. In command prompt type "ping 10.0.0.3". This pings the PC2 and a response is generated in PC0.

Simulation time:

Select a simple PDV and select a source and destination computer. Clicking on auto capture option allows us to see how ports are transferred to and from device.

→ Using Switch :

- i) Add a generic switch and three PCs to the workspace.
- ii) Configure the IP addresses of each PC's in the configuration tab. Ensure that IP is different for each device.
- iii) Connect all PC's to the switch using copper straight through.
- iv) If no. of ports are insufficient then add extra ports by clicking on device. Turn off the device and add the necessary ports.
- v) Write the IP's of all devices in a note below the device.

Real time:

Select a source PC and in the desktop tab select command prompt option. In command prompt option, ping the destination PC by specifying its IP.

Simulation time:

Select a simple PDU and select a source and destination computer. Clicking on auto capture option allows us to see how packets are transferred.

↳ Hybrid Mode:

- i) Add a switch, 3 hubs and 12 PCs to the workspace.

- ii) Connect the three hubs to the switch and 4 PCs to each of the hubs using copper cross over and copper straight through wires respectively.

- iii) Configure the IP of each of the PC in configure and add a note below each PC containing IP address.

Real-time mode:

Select the PC you want to send the packet from and open its command prompt. Specify the destination PC by specifying its IP address. A response is sent by the destination PC to the source PC.

Simulation mode:

Add a simple POU by selecting the pair of PC and click on auto capture from right panel.

Observation:

→ Hub :

Learning outcome:

- i) When a source sends a packet in the network the hub scans the packet and broadcast over the network, i.e., it sends data to all the end devices in the network and the node where it matches with the specified address accepts the packet and acknowledges it. Remaining nodes denied / ignore the message.
- ii) The communication between hub and end device is established through copper straight through wire as they belong to different layers.
- iii) The number of ports can be added if needed by clicking on the device and adding the necessary ports.

Result:

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=0ms

Ping statistics for 10.0.0.3

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

→ Switch:Learning options:

- When a source device sends a message to the switch once a connection is established. Which takes some time called as learning time, the switch receives the packet. It initially broadcasts the packet to all connected devices to locate the destination. Once the destination is located the message is sent only to that device.
- The connection between the switch and end device is established using copper straight through as they belong to different network layers.
- The number of ports can be added if needed by clicking on the device and adding the necessary ports.

Result:

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=0ms

Reply from 10.0.0.3 : bytes=32 time=0ms

Reply from 10.0.0.3 : bytes=32 time=0ms

Reply from 10.0.0.3 : bytes = 32 time = 0 ms

Ping statistics for 10.0.0.3:

Packets: Sent = 4, Received = 4, Lost = 0.

→ Hybrid Mode

Learning Outcome:

- i) The switch and hub are connected through copper crows over as they belong to the same network layers but PC and hub are connected through copper straight through as they belong to the different network layers.
- ii) The message from the source PC to destination is sent through the hub which then sends the all its connected PC's and the switch. The switch then sends the message to the respective hub and the hub sends the message to all its connected PC. The destination PC acknowledges that it has received the message by sending a acknowledgement back to the source PC.
- iii) The number of ports can be added if needed by clicking on the device and adding the necessary ports.

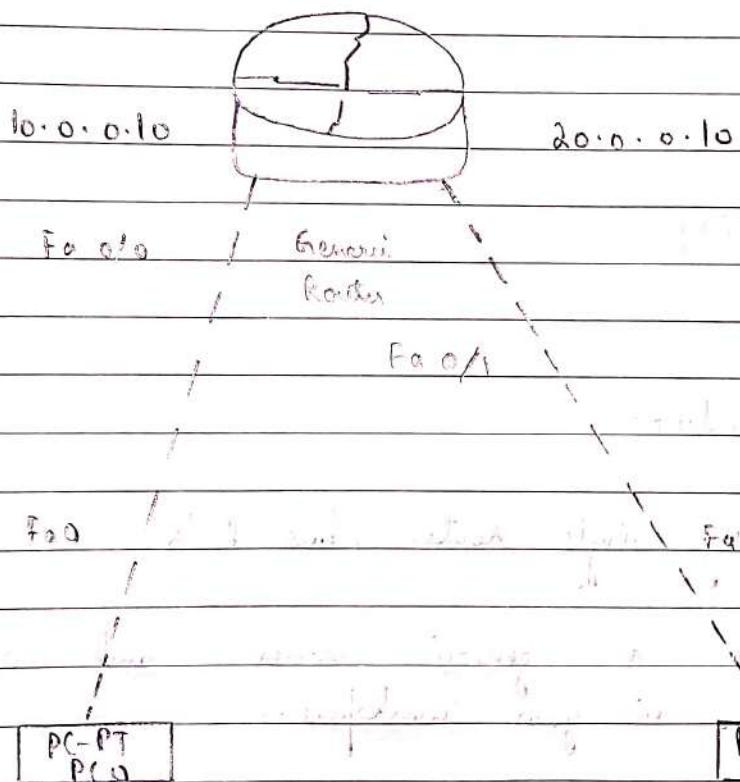
Lab: Week 2

Experiment using routers and PCs

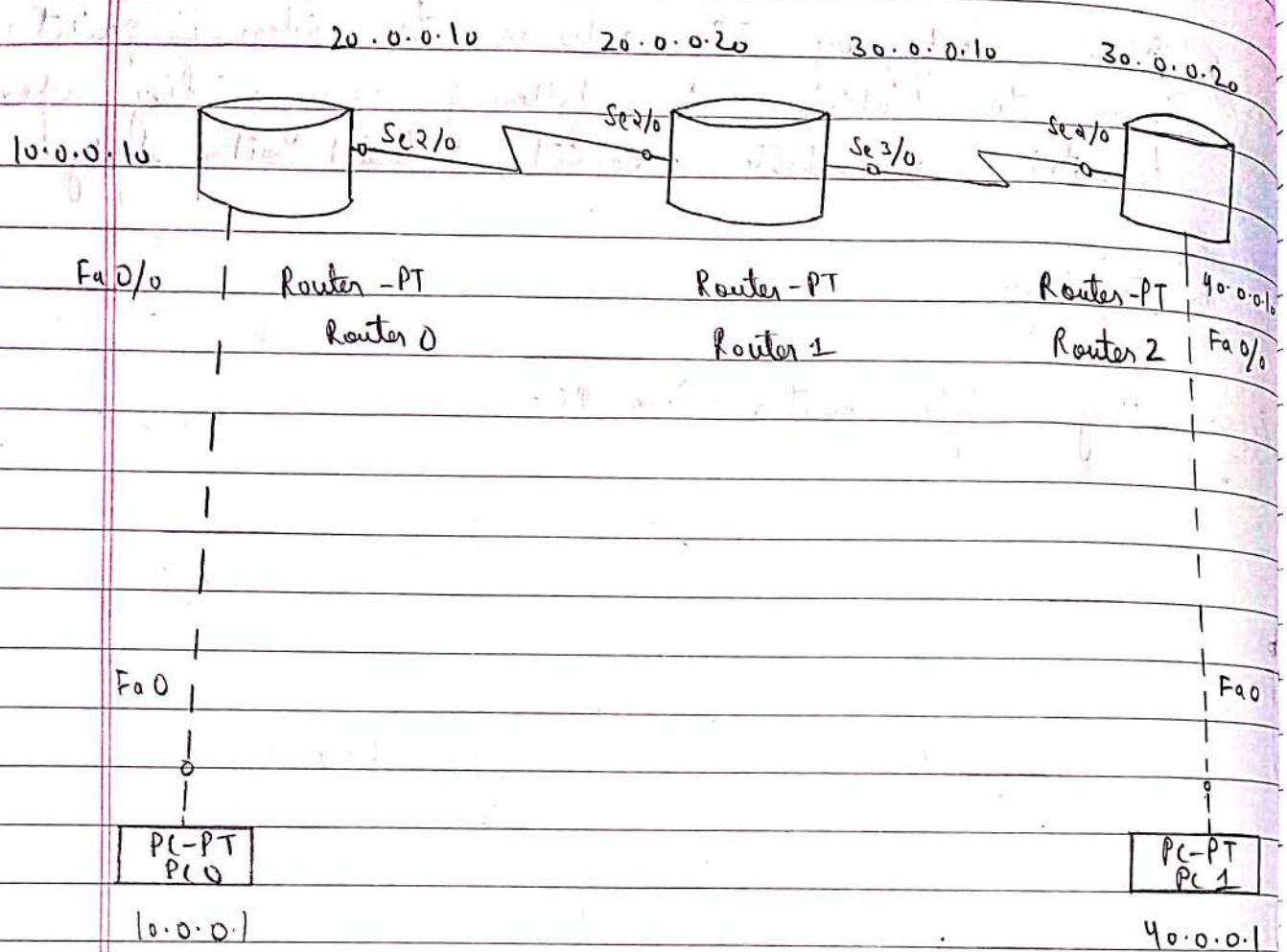
↳ Aim: Configuring IP addresses to routers in packet tracer to explore the following messages: Ping response, destination unreachable, Request timed out, reply!

↳ Topology:

→ Using single router, two PCs:



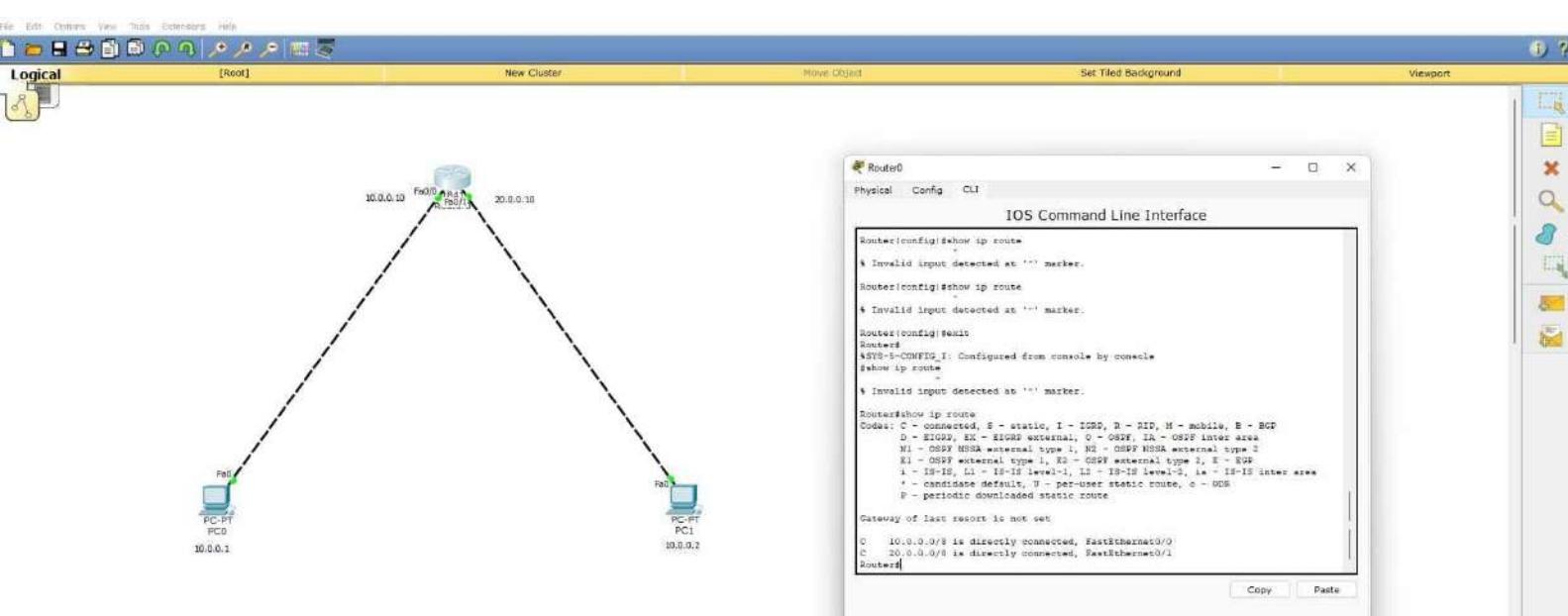
→ Using three routers, two PCs:



↳ Procedure:

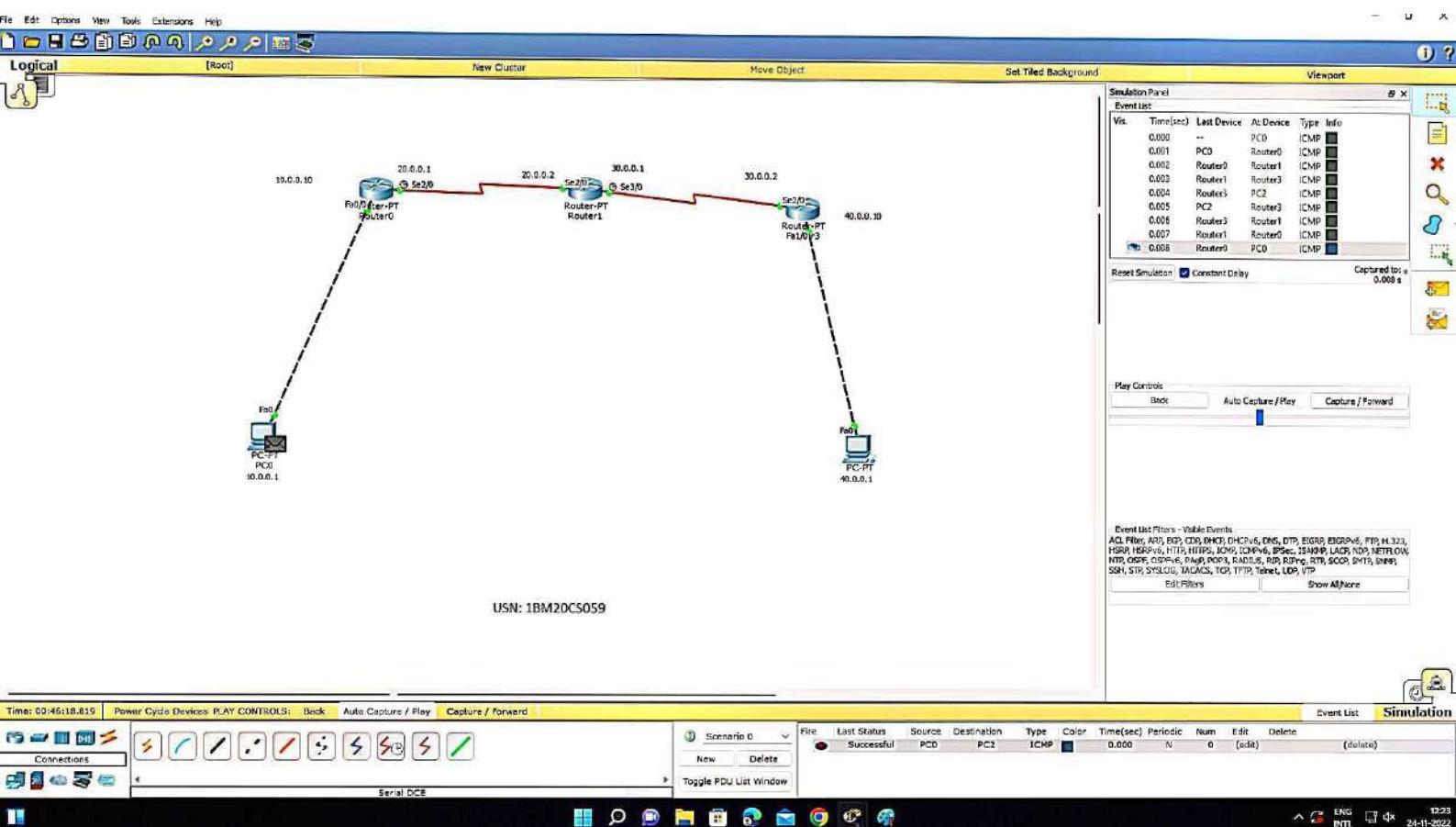
→ Using single router, two PCs

- Place a generic router and add the generic PCs in your workspace.
- Connect the router and PCs using copper wires.
- Configure IP address of each PC and in the configuration tab under settings, set gateways for both PCs to the router.
- Click on the generic router and go to the CLI



USN: 1BM20CS059





task : Enter the following commands to set up a connection between PCs (1) and generic router through gateway 10.0.0.10.

→ No

→ enable

config t

(config) # interface fastethernet 0/0

(config-if) # ip address 10.0.0.10 255.0.0.0

no shut

exit

Now to set up connection between PC and the router through gateway 10.0.0.10

interface fastethernet 1/0

ip address 10.0.0.10 255.0.0.0

no shut

exit

Once we enter no shut both times the amber light between the PC and router turns green indicating that the two devices are ready for use.

Simulation mode: Add a simple PDU by selecting the PCs and click on capture from right panel.

Real time mode: Select the PC you want to send the packet from which is PC0 in our case and open its command prompt from desktop tab. Specify the destination address. Response is sent from destination PC to source PC.

→ Using three routers, two PCs

- i) Place 3 generic routers and 2 generic PCs in the workspace.
- ii) Place a note for each device (PC and router) and specify the IP address.
- iii) Connect the routers and PCs using copper crossover.
- iv) Connect the routers using serial DCE.
- v) Click on each PC, go to the configure tab. Set the IP address and subnet mask in fastethernet.
- vi) Next click on settings in the config tab. Set the gateway as the IP address of the next router [e.g. 10.0.0.10]
- vii) IP address of PCs and its gateway address should belong to the same network.

For connecting two routers:

Click on Router #10. Go to CLI and enter the following commands.

- no enable
- enable
- config t
- interface serial 2/0
- ip address 10.0.0.10 255.0.0.0
- no shut

Click on router 1, open CLI and enter the following commands.

- no
- enable
- config t
- interface serial 2/0
- ip address 20.0.0.20 255.0.0.0
- no exit

After this procedure, the red lights between the two routers will now turn green [router 0 and router 1]. Indicating that they are now ready for communication.

For connecting two devices [1 PC and one router].

- i) Since IP address of the PC is already configured, go to router.
- ii) open CLI for router 0 and enter the following commands.

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

The red light turns green meaning that the router is ready for communication.

Teaching Router 0 of network 30:

- no
- enable
- config t
- interface serial 2/0
- ip route 30.0.0.0 255.0.0.0 20.0.0.20
- exit
- show ip route

Teaching Router 0 of network 40:

- no
- enable
- config t
- interface serial 2/0
- ip route 40.0.0.0 255.0.0.0 20.0.0.20
- exit
- show ip route

Similarly repeat this for router 1 and router 2.

Simulation mode: Add a simple PDU by selecting the PC 2 and click on auto capture from right panel.

Real time mode: Select the PC PC 0 and go to its command prompt and ping the router 0. Once the message has been sent successfully repeat this with routers 1 and 2 as well. Finally, ping PC 1.

↳ Observation:

learning outcome:

→ 1 router:

When PC 0 pings PC 1 for the first time, we get the first packet as request timed out.

Now, if we ping PC 1 again from PC 0 we get all 4 packets without any loss. Now reverse pinging of PC 0 from PC 1 will also not lead to any loss, all packets are acknowledged.

→ 3 routers:

Before training the routers, we get the results as destination host unreachable. After training the routers, we get clear statistics on the result.

↳ Result:

→ Using 1 router, three PCs.

c:\> Ping 20.0.0.1

Pinging 20.0.0.1 with 32 bytes of data.

Request timed out

Reply from 20.0.0.1: bytes = 32 time < 1ms TTL=127

Reply from 20.0.0.1: bytes = 32 time < 1ms TTL=127

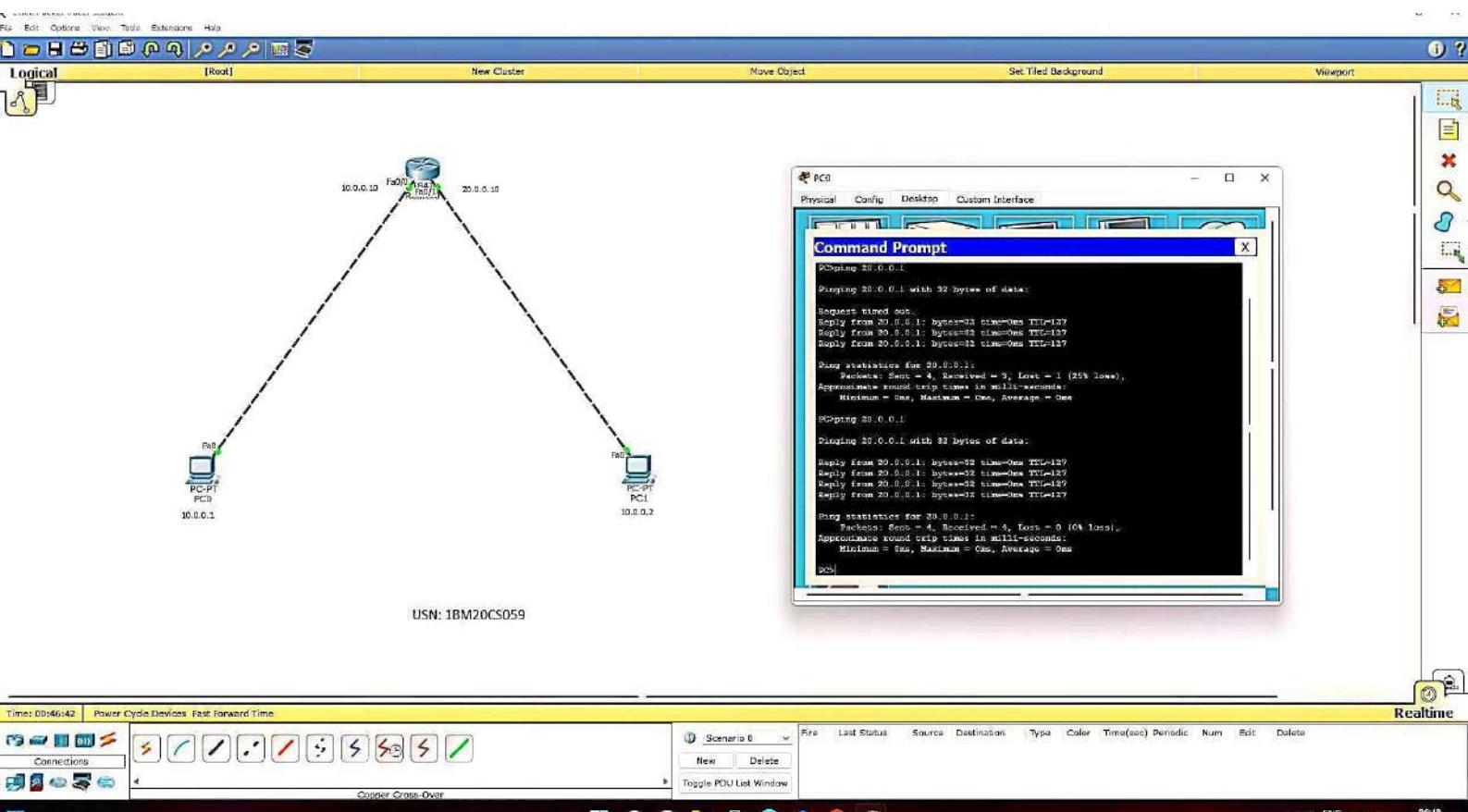
Reply from 20.0.0.1: bytes = 32 time < 1ms TTL=127

Ping statistics for 20.0.0.1

Pinging 20.0.0.1 with 32 bytes of data:

Reply from 20.0.0.1: bytes = 32 time < 1ms TTL=127

Reply from 20.0.0.1: bytes = 32 time < 1ms TTL=127



Reply from 20.0.0.1 : bytes = 32 time < 1ms TTL = 127.

Reply from 20.0.0.1 : bytes = 32 time < 1ms TTL = 127.

Ping statistics for 20.0.0.1:

packets : sent = 4 , received = 4 , lost = 0 (0% loss)

→ Using three routers, two PCs:

1. PC > Ping 40.0.0.1

Pinging 40.0.0.1 with 32 bytes of data

Reply from 10.0.0.10 : destination host unreachable.

Reply from 10.0.0.10 : destination host unreachable.

Reply from 10.0.0.10 : destination host unreachable

Reply from 10.0.0.10 : destination host unreachable

Ping statistics for 40.0.0.1

Packet : sent = 4, received = 0 , lost = 4 (100% loss)

2. PC > Ping 10.0.0.10 with 32 bytes of data

Reply from 10.0.0.10 : bytes = 32 time = 0ms TTL = 255

Reply from 10.0.0.10 : bytes = 32 time = 0 ms TTL = 255

Reply from 10.0.0.10 : bytes = 32 time = 0 ms TTL = 255

Reply from 10.0.0.10 : bytes = 32 time = 0 ms TTL = 255

Ping statistics for 10.0.0.10:

Packets : sent = 4, Received = 4, lost = 0 (0% lost)

3. PC > Ping 20.0.0.10 with 32 bytes of data

Reply from 20.0.0.10 : bytes = 32 time = 1ms TTL = 255

Reply from 20.0.0.10 : bytes = 32 time = 1ms TTL = 255

Reply from 20.0.0.10 : bytes = 32 time = 0 ms TTL = 255

Reply from 20.0.0.10 : bytes = 32 time = 0 ms TTL = 255

Ping statistics for 20.0.0.10:

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

4. PC > ping 20.0.0.10

Pinging 20.0.0.10 with 32 bytes of data

Reply from 20.0.0.10: bytes = 32 time = 1ms TTL = 254.

Reply from 20.0.0.10: bytes = 32 time = 1ms TTL = 254

Reply from 20.0.0.10: bytes = 32 time = 1ms TTL = 254

Reply from 20.0.0.10: bytes = 32 time = 8ms TTL = 254.

Ping statistics for 20.0.0.10:

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

5. PC > 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 = 10ms TTL = 125

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

Reply from 40.0.0.1: bytes = 32, time = 8ms TTL = 125.

Ping statistics for 40.0.0.1:

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

6. PC > ping 40.0.0.1

Pinging 40.0.0.1 with 32 bytes of data:

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

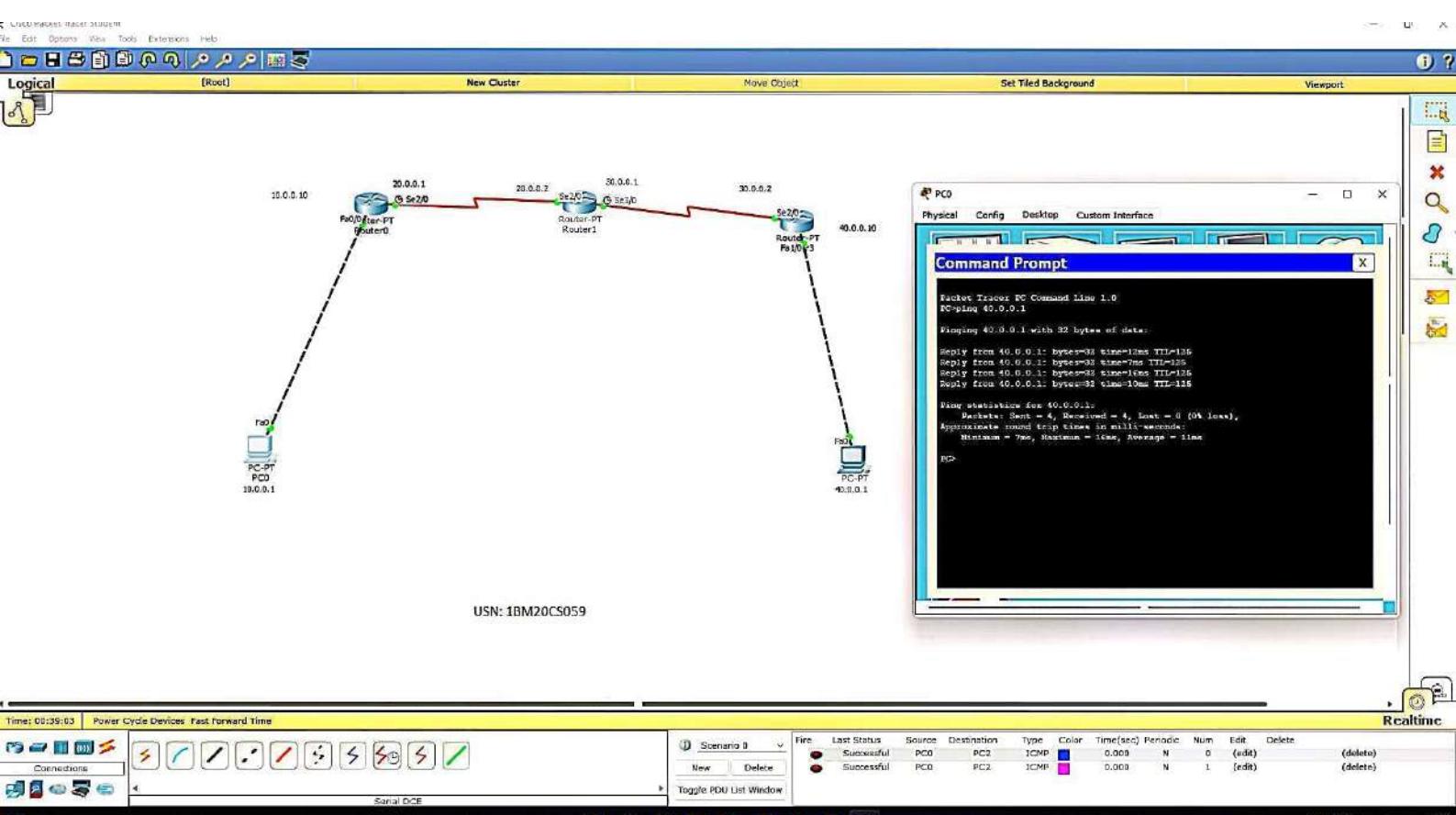
21/12 Reply from 40.0.0.1: bytes = 32 time = 24ms TTL = 125.

Reply from 40.0.0.1: bytes = 32 time = 9ms TTL = 125.

Reply from 40.0.0.1: bytes = 32 time = 9ms TTL = 125.

Ping statistics for 40.0.0.1:

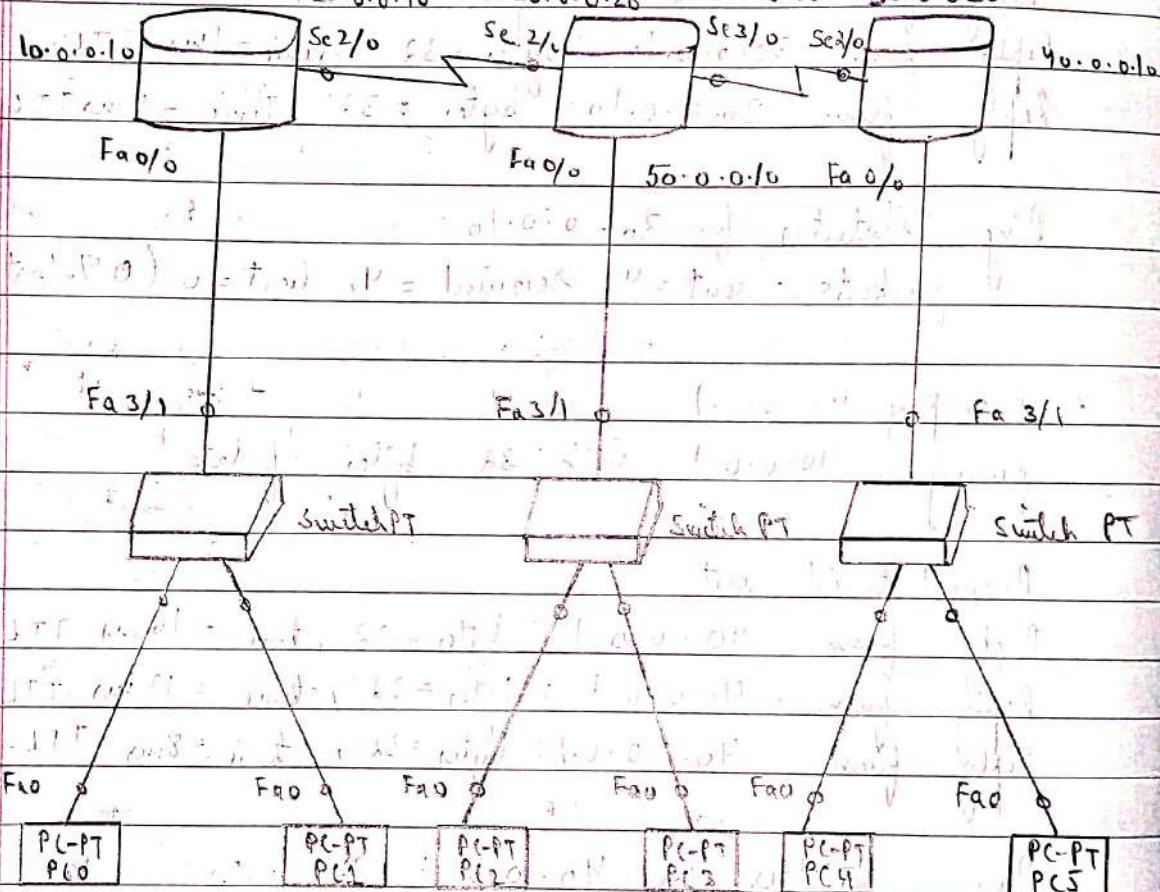
Packets: sent = 4, received = 4, lost = 0 (0% loss).



Lab: Week 3 Experiment using Routers & Switches.

- ↳ Aim: To configure default router to a router via switcher using minimum commands.

↳ Topology:



↳ Procedure:

- Place 3 generic routers, 3 generic switches and 6 generic PCs in the workspace.
- Connect the PCs to the switch using copper straight through wire.
- Connect the switches to routers also using copper straight through.

- iv) Connect the routers with one another using serial DCE.
- v) Set the IP address of each PC and subnet mask in fast ethernet 0.
- vi) Set the default gateway for each PC using settings.
- vii) Click on the router and enter the following commands to establish connection with the switch.

```

→ enable
→ config t
→ interface fast ethernet 0/0
→ ip address 10.0.0.10 255.0.0.0
→ no shut.

```

After some time the light which was amber for the switch will turn green indicating the switch and router are ready for communication.

Repeat the same for the other three routers.

Click on the router to now establish connection with the neighbouring router.

```

→ enable
→ config t
→ interface serial 2/0
→ ip address 20.0.0.10 255.0.0.0
→ no shut.

```

→ Click on router 1

→ enable

→ config +

→ interface serial 2/0

→ ip address 20.0.0.20 255.0.0.0

→ no shut

The red light between the two routers will turn green indicating they are ready for communication.

Teaching Router 0 about network 30, 40 & 50 :-

Click on router 0, open CLI :-

→ enable

→ config +

→ interface serial 2/0

→ ip route 0.0.0.0 0.0.0.0 20.0.0.20

→ exit

→ show ip route

It will show that networks 30, 40 & 50 are connected via gateway 20.0.0.20.

Teaching Router 1 of network 10 & 40 :-

→ enable

→ config +

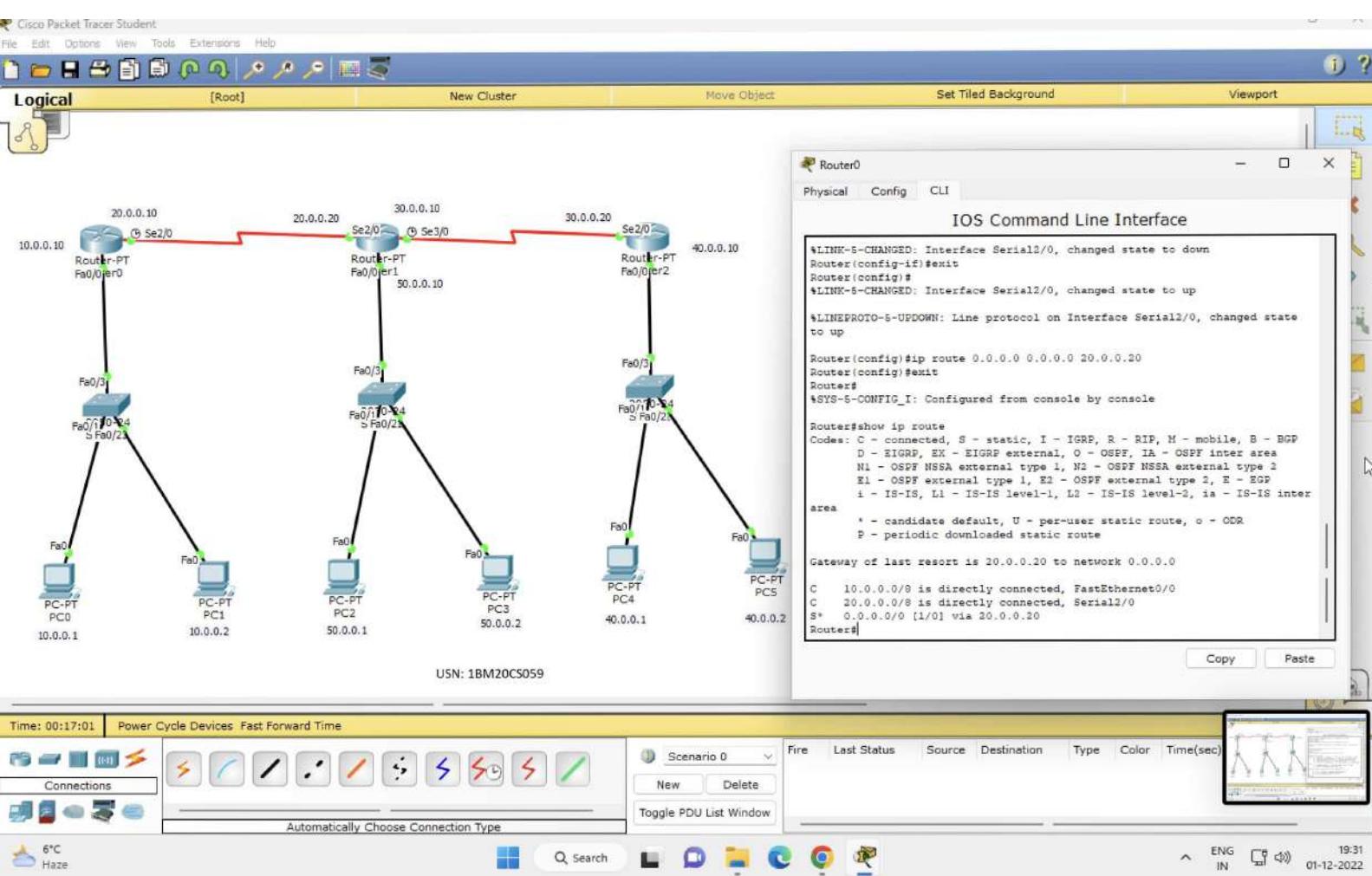
→ interface serial 2/0

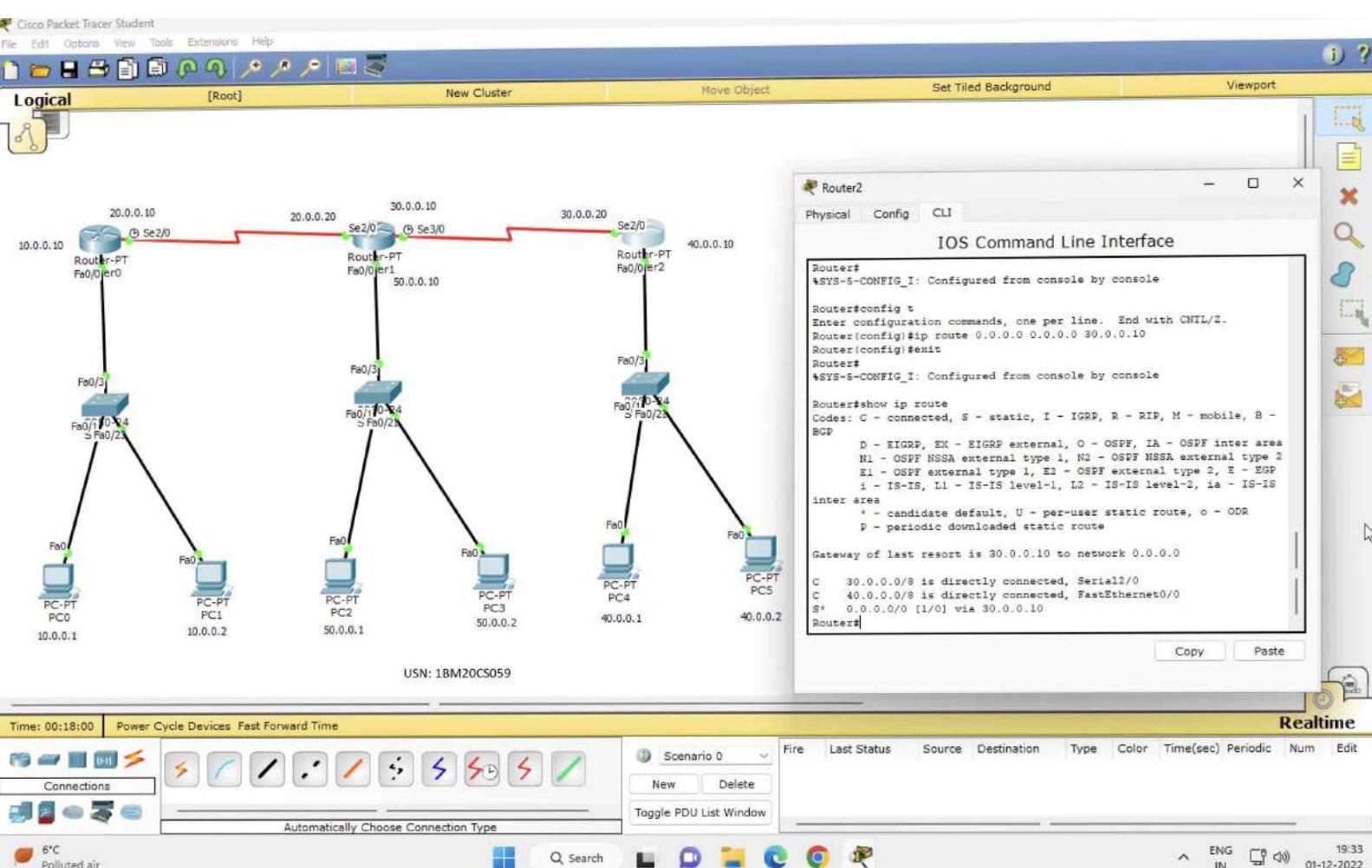
→ ip route 10.0.0.0 255.0.0.0 20.0.0.10

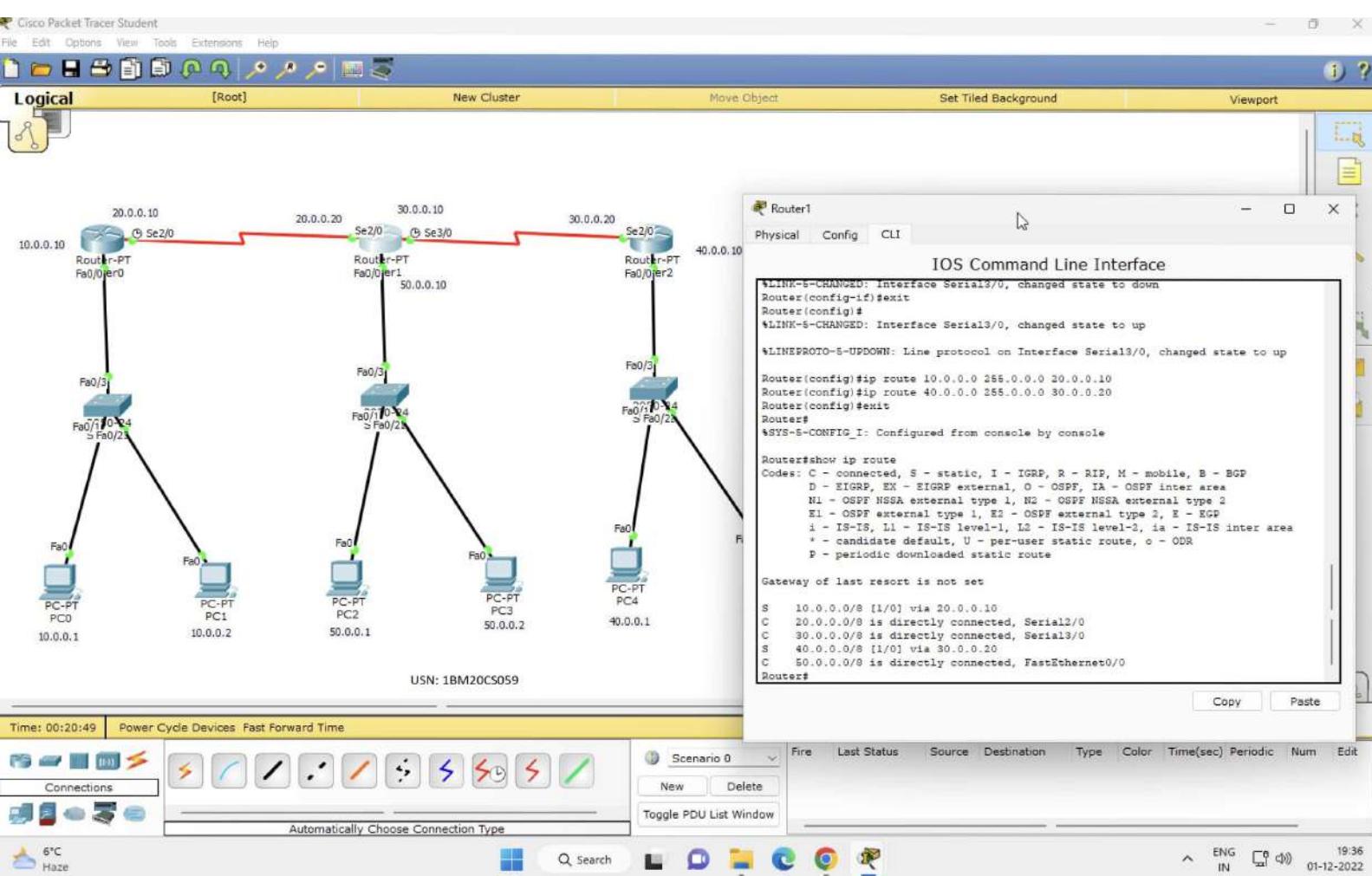
→ exit

→ interface serial 2/0

→ ip route 40.0.0.0 255.0.0.0 30.0.0.10







- exit
- show ip route

Teaching router 2 of network 10, 20 850:

- enable
- config t
- interface serial 2/0
- ip route 0.0.0.0 0.0.0.0 30.0.0.10
- exit
- show ip route

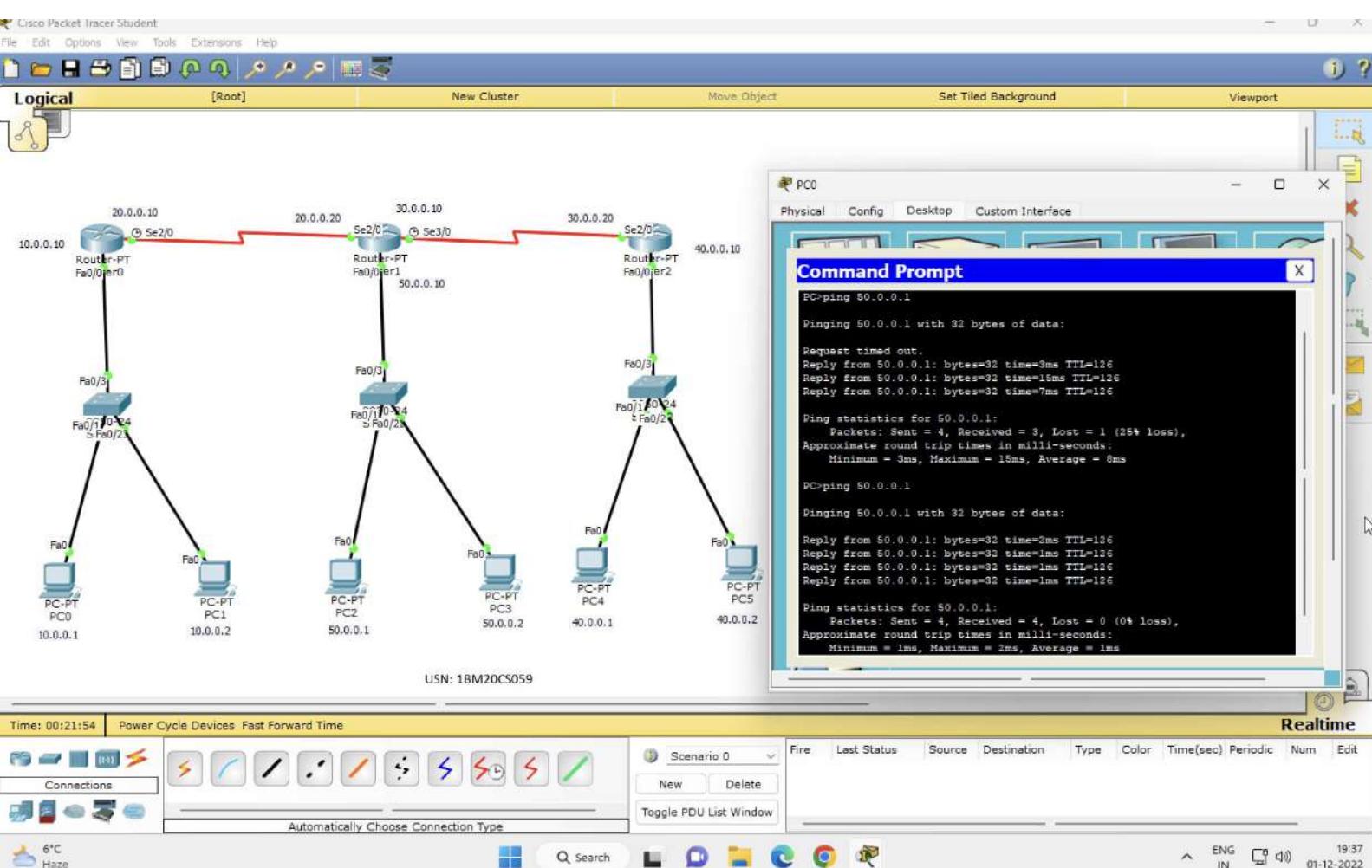
Simulation mode: Add a simple PDU by selecting the PCs and click on the "auto capture" from right panel.

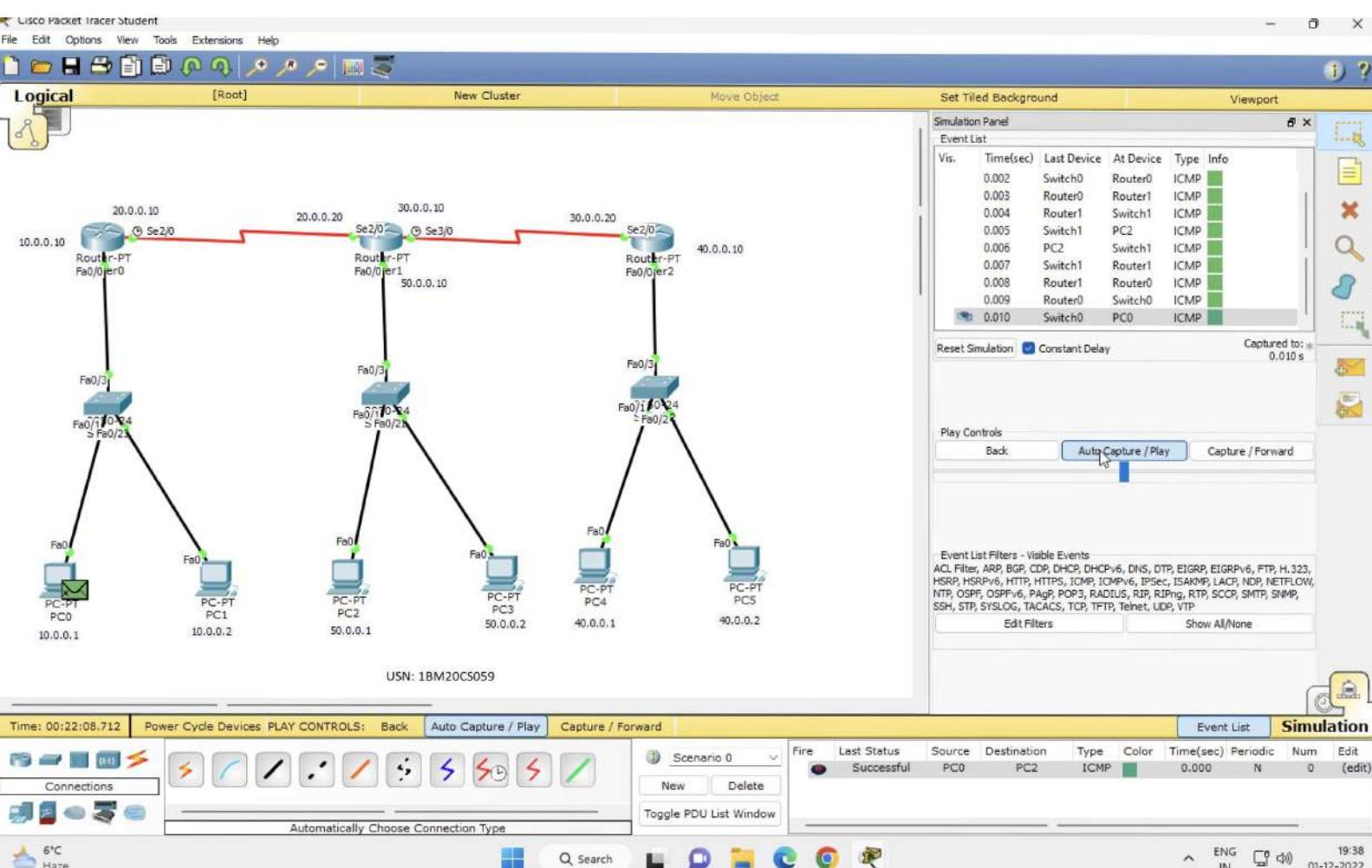
Real-time mode: Select the PC Pw and give the "tr" command at prompt and ping a PC in network 50.

At first it will show request timed out & 1 packet will be lost during transmission. But on executing the command one more, the PC will now have learnt the network and the message will be successfully sent to the PC in network 50 without any losses. Finally ping a PC in network 40 and repeat the same. We will observe that the message is sent successfully.

4 Observation :

Learning outcome: In this network router R₁ does not have a default router, because R₀ and R₂ cannot become a default router simultaneously and if any one of R₀ and R₁ is default then the packets that are supposed to enter R₁ can go to R₂/R₀ as they are default.





↳ Result:

1. PC > Ping 50.0.0.1

Pinging 50.0.0.1 with 32 bytes of data:

Request timed out.

Reply from 50.0.0.1 : bytes = 32 time = 14 ms TTL = 128

Reply from 50.0.0.1 : bytes = 32 time = 12 ms TTL = 127

Reply from 50.0.0.1 : bytes = 32 time = 3 ms TTL = 124

Ping statistics for 50.0.0.1:

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

2. PC > Ping 50.0.0.1

Pinging 50.0.0.1 with 32 bytes of data:

Reply from 50.0.0.1 : bytes = 32 time = 2 ms TTL = 124

Reply from 50.0.0.1 : bytes = 32 time = 2 ms TTL = 124

Reply from 50.0.0.1 : bytes = 32 time = 11 ms TTL = 124

Reply from 50.0.0.1 : bytes = 32 time = 2 ms TTL = 124

Ping statistics for 50.0.0.1:

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

3. PC > 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 = 20 ms TTL = 125

Reply from 40.0.0.1 : bytes = 32 time = 3 ms TTL = 125

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

Ping statistics for 40.0.0.1:

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

4. PC > Ping 40.0.0.1
Pinging 40.0.0.1 with 32 bytes of data.

Reply from 40.0.0.1 : bytes = 32 time = 23ms TTL = 125.

Reply from 40.0.0.1 : bytes = 32 time = 18ms TTL = 125.

Reply from 40.0.0.1 : bytes = 32 time = 14ms TTL = 125.

Reply from 40.0.0.1 : bytes = 32 time = 3ms TTL = 125.

Ping statistics for 40.0.0.1:

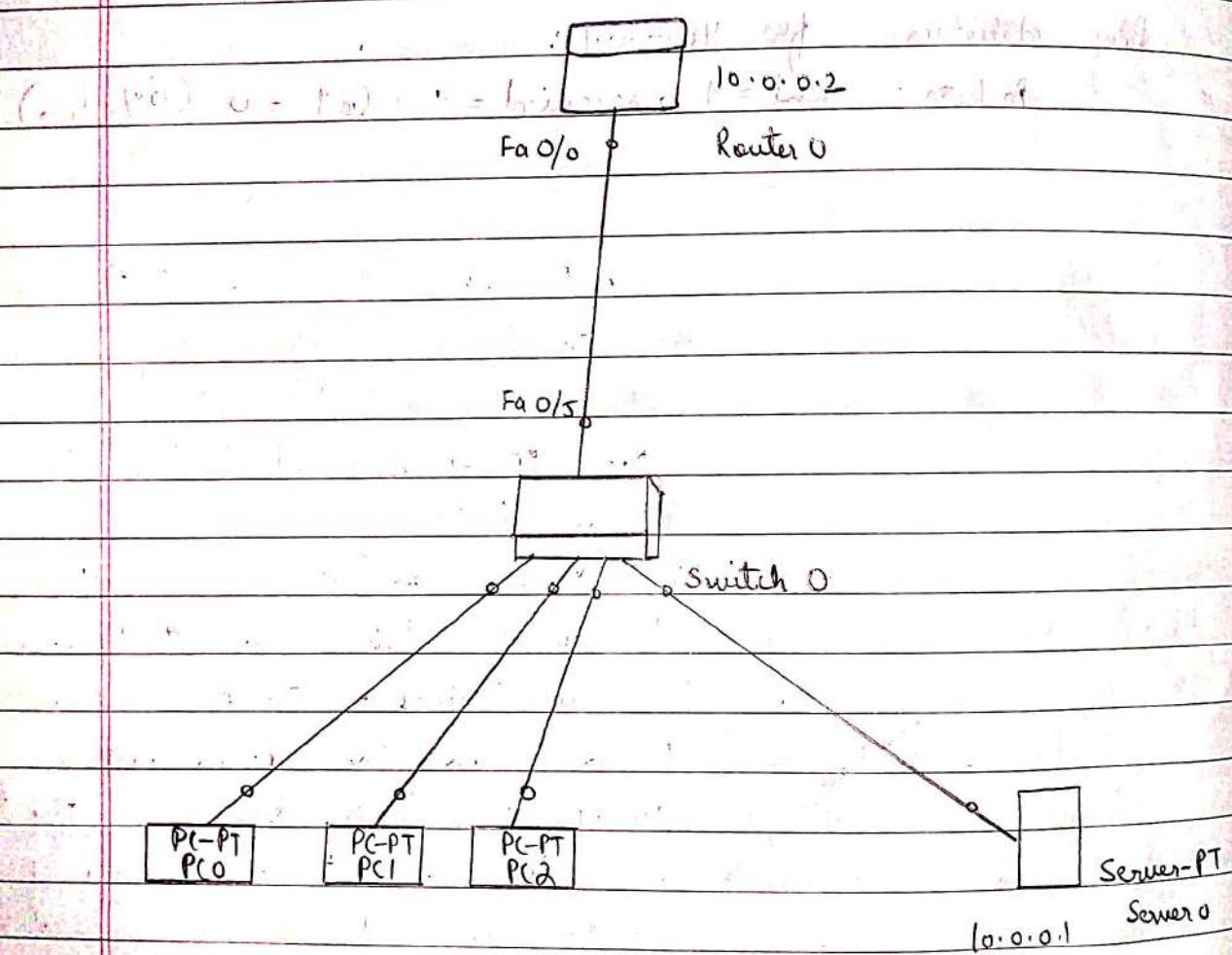
packets: sent = 4, received = 4, lost = 0 (0% loss).

Lab: Week 4

DHCP configuration

- ↳ Aim: Configuring DHCP within a LAN in a packet bauer.

- ↳ Topology:



- ↳ Procedure:

- Place a generic router, a generic switch, 3 generic PCs, and a generic server in the workspace as shown in the given topology.
- Connect the PCs to the switch through copper straight through.

- iii) Connect the server to the switch and switch to the router using copper straight through
- (iv) Place a note below the server and write the ip address as 10.0.0.1
- v) Configure the ip address of the server as 10.0.0.1 and configure the gateway as 10.0.0.2.
- vi) Open the CLI of the router by clicking the router and enter the following commands

```

→ enable
→ config t
→ interface fastethernet 0/0
→ ip address 10.0.0.2 255.0.0.0
→ no shut

```

The light will turn green for router and amber for the switch.

After some time, the amber color also changes to green.

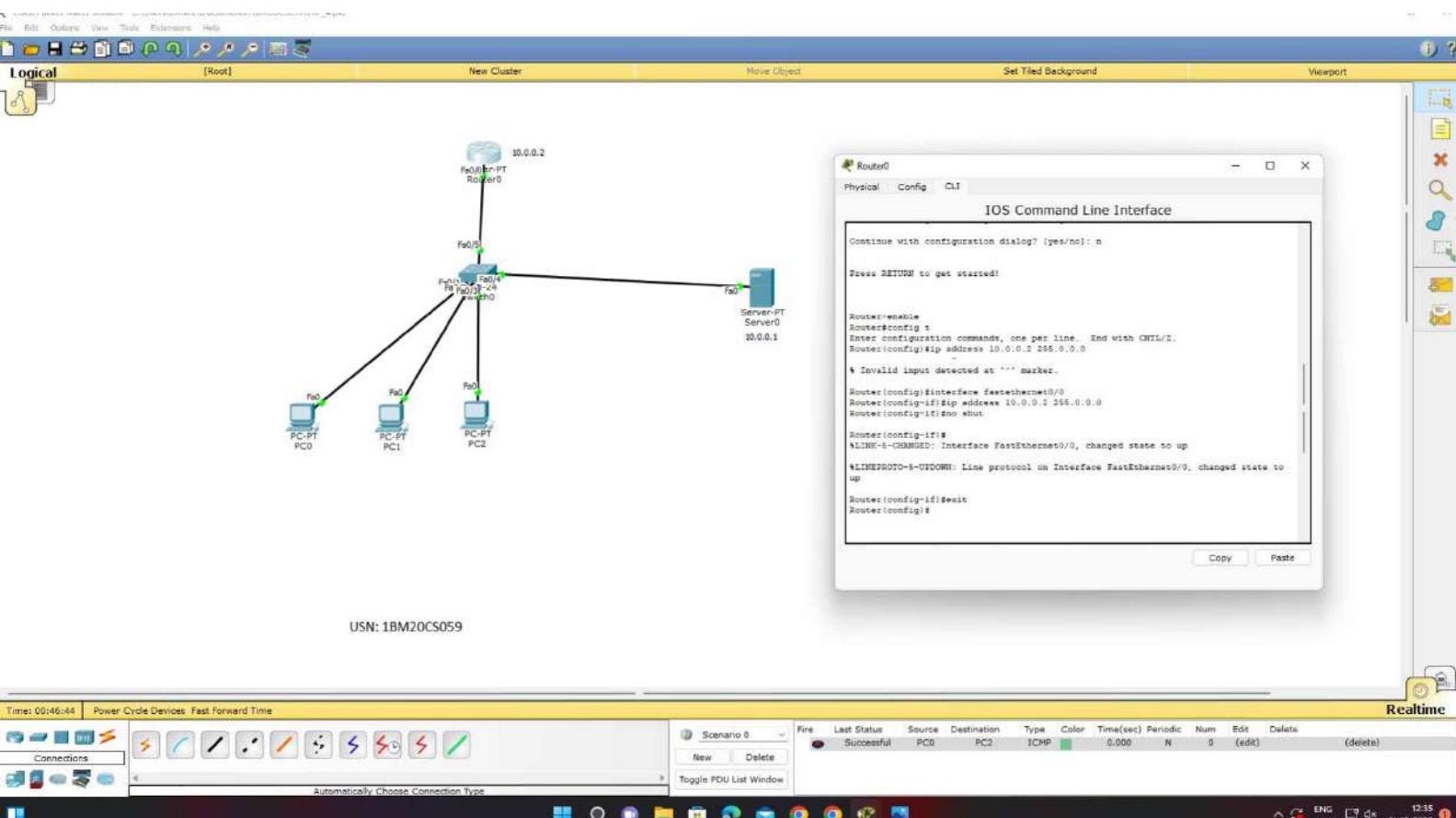
Now, click on the server

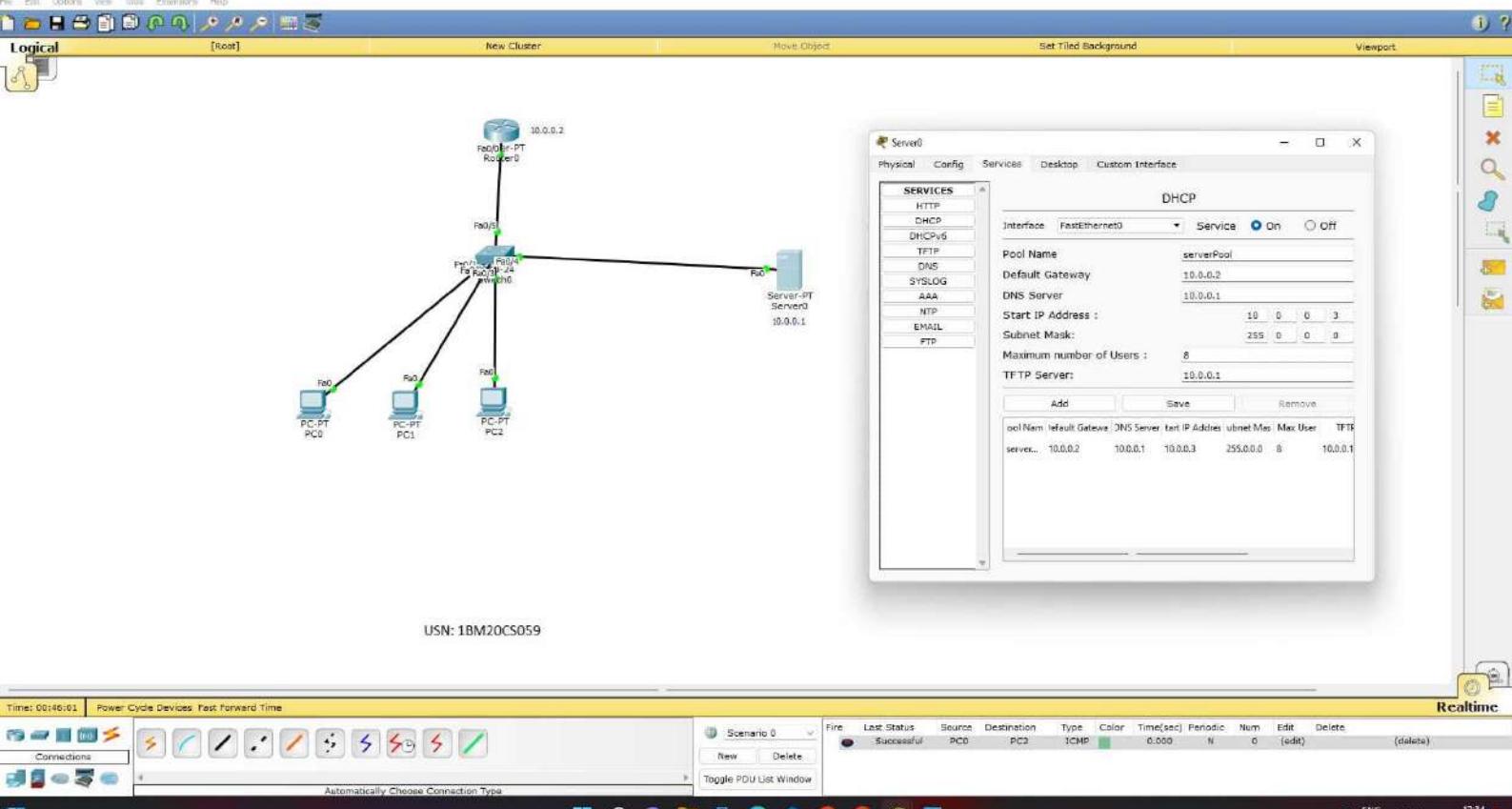
```

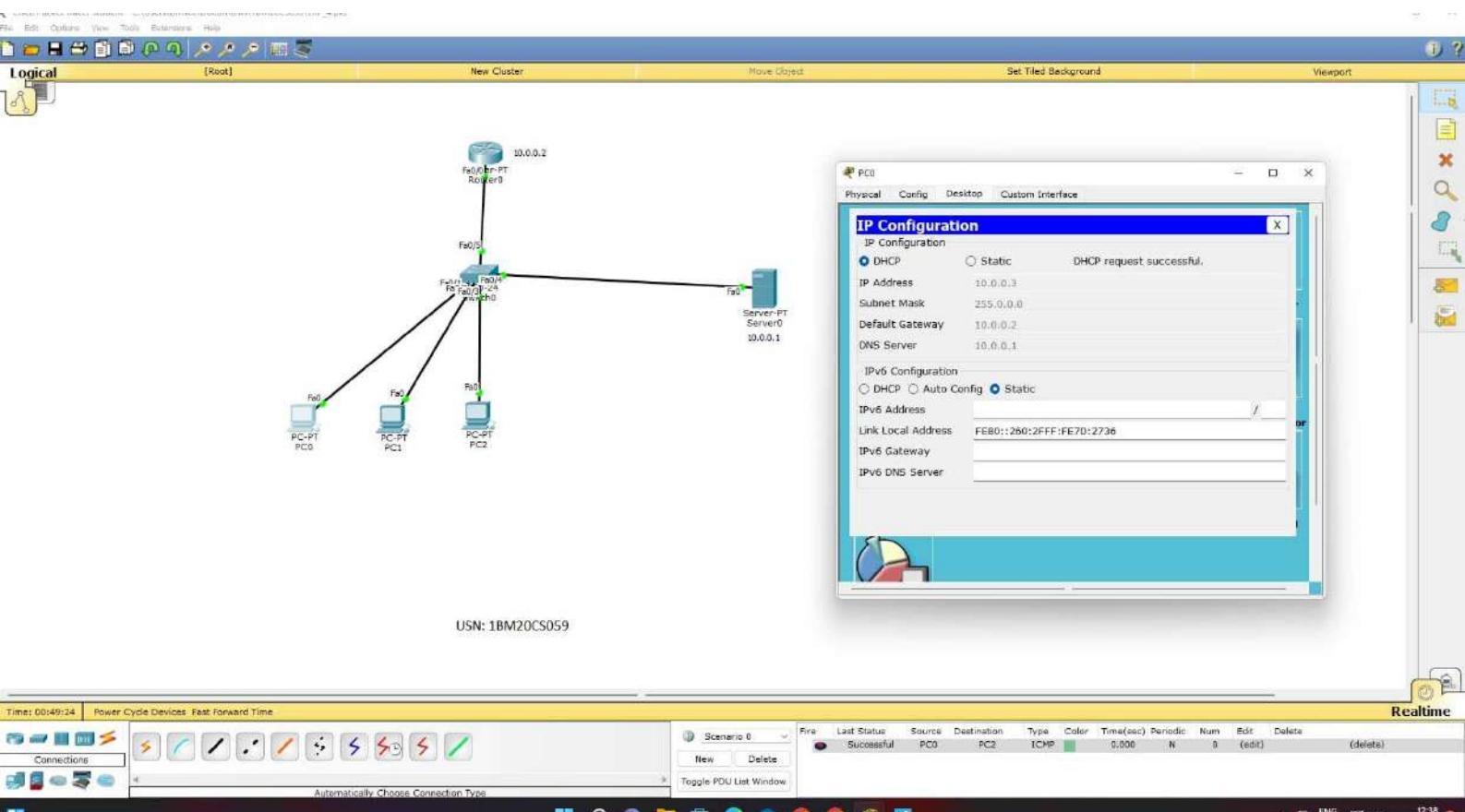
→ open the server tab
→ Click on DHCP
→ Turn the switch ON
→ set default gateway as 10.0.0.2
→ DNS server as 10.0.0.1 (same as IP address)
→ TFTP server as 10.0.0.1 (same as IP address)
→ keep the maximum number of users 8.
→ keep the start IP address as 10.0.0.3 & subnet mask as 255.0.0.0

```

After this procedure, save the file.







Now, click on PC 0 and under the desktop tab.

Go to IP configuration and click on DHCP.

If there are no errors, it will show

'DHCP request successful' and the IP address will be 10.0.0.3.

Similarly, repeat this procedure for the rest other PCs.

Simulation mode: Add a simple PDC by selecting the PCs and then click on auto capture from right panel.

Realtime mode: Select the PC PC0 and ping the PC in the command prompt. Once the ping statistics are successfully displayed, we can repeat this with PC 1 as well.

Observation:

Learning outcome: The server automatically sets the IP address and subnet and gateway to all the PCs and IP address is allocated serially in DHCP protocol.

Result:

1. PC > Ping 10.0.0.5

Pinging 10.0.0.5 with 32 bytes of data:

Reply from 10.0.0.5: bytes = 32 time = 20ms TTL = 128

Reply from 10.0.0.5: bytes = 32 time = 20ms TTL = 128

Reply from 10.0.0.5: bytes = 32 time = 0ms TTL = 128

Reply from 10.0.0.5: bytes = 32 time = 20ms TTL = 128

Ping statistics for 10.0.0.5:

Packets : sent = 4, Received = 4, lost = 0 (0% loss)

Approximate round-trip times in milli-seconds :

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

2. 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 = 0 ms TTL = 128

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

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

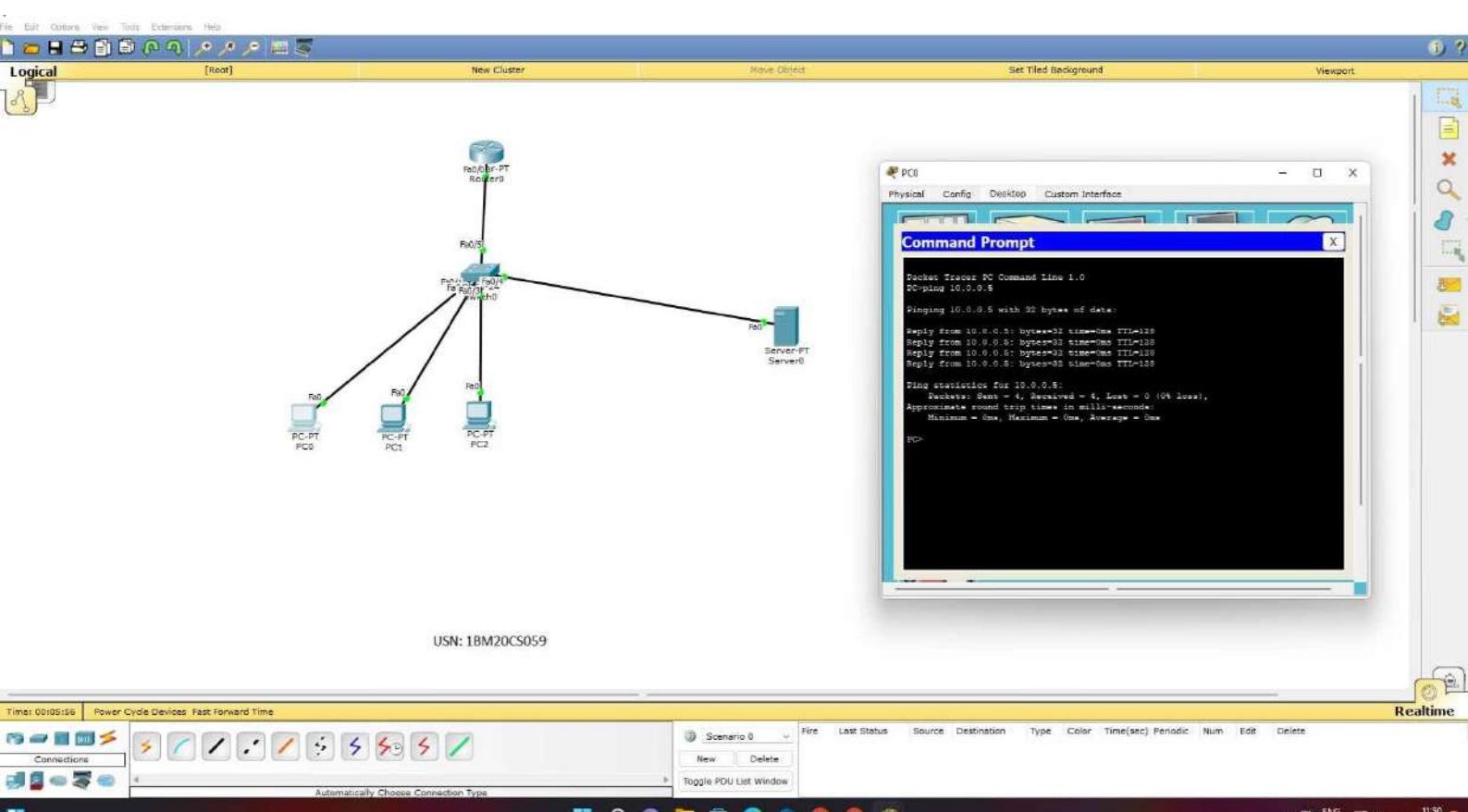
Reply from 10.0.0.4 : bytes = 32 time = 5 ms TTL = 128 .

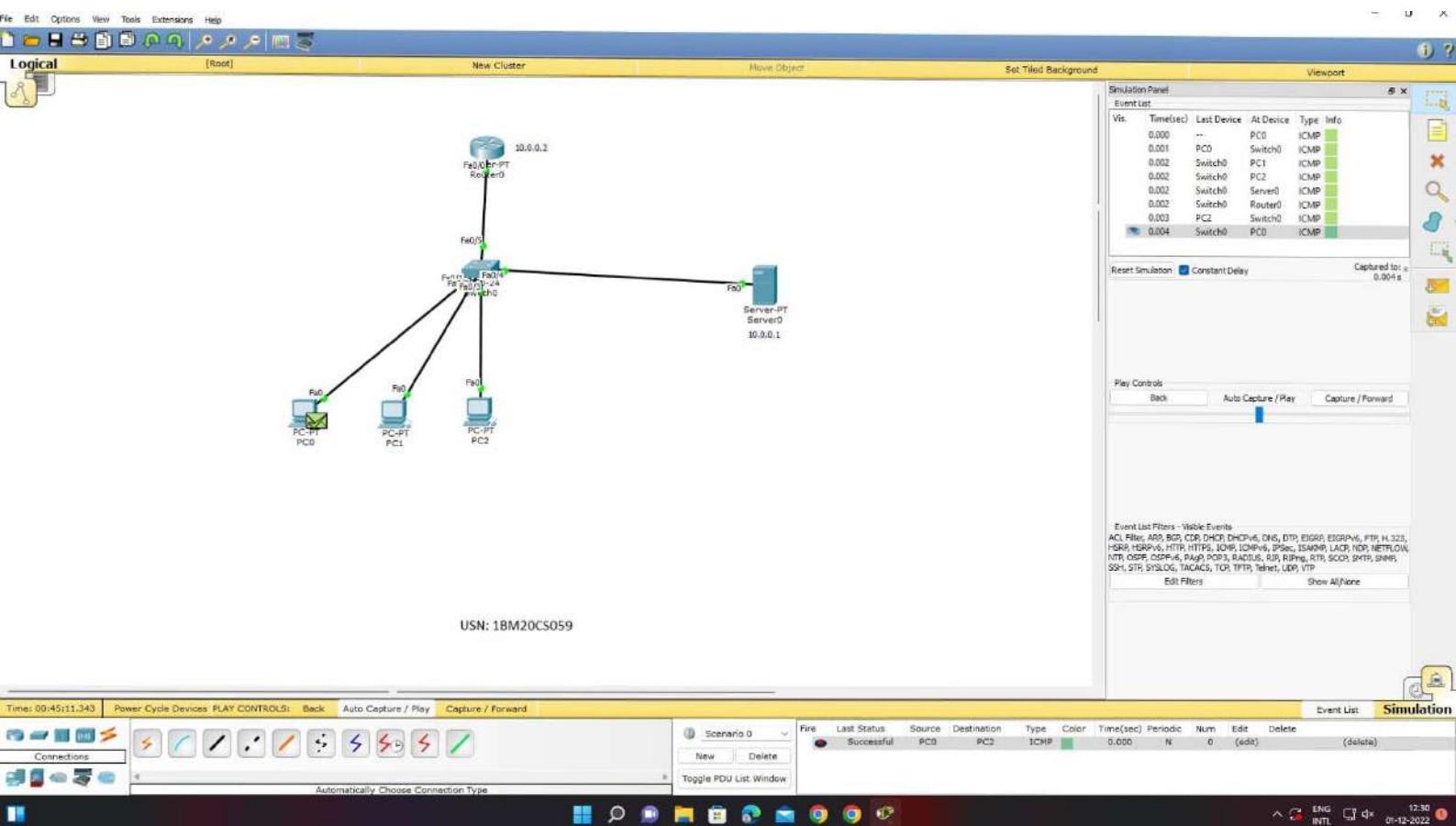
Ping statistics for 10.0.0.4:

Packets : sent = 4, Received = 4, lost = 0 (0% loss),

Approximate round-trip times in milli-seconds :

Minimum = 0 ms, Maximum = 5 ms, Average = 1 ms.

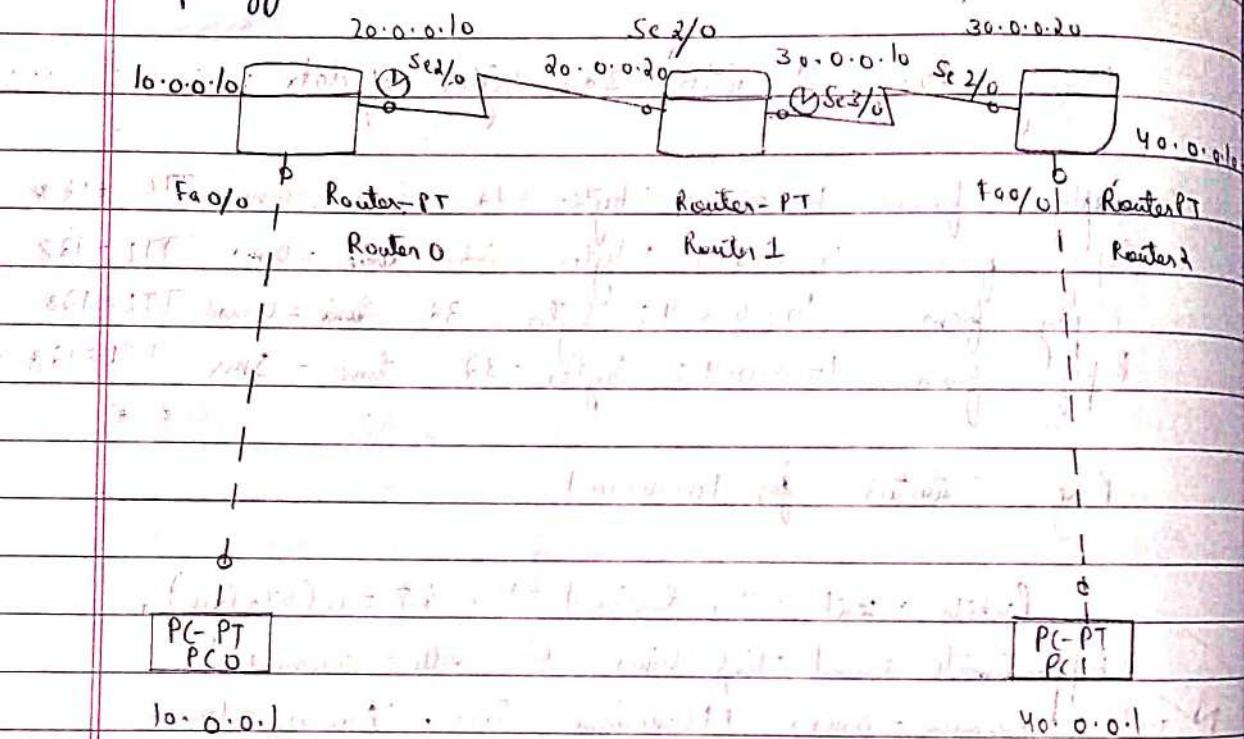




Lab: Week 5 - Routing Information Protocol

b) Aim: Configuring RIP routing protocol in routers.

c) Topology:



d) Procedure:

- i) Place 3 generic routers and 2 generic PCs in the workspace.
- ii) Connect the routers and the two PCs using copper cross over.
- iii) Connect the routers using serial DCE with clock symbol.
- iv) Place notes near the PCs and routers.
- v) Select the IP address of PC 0 and PC 1 as well as their subnet mask and default gateway.
- vi) Go to the CLI of router 0 and enter the following:

commands:

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

The connection should turn green. Repeat for PC1 and router 2 as well.

Now, Open CLI of router 0 and enter the following commands:

- enable
- config t
- interface serial 2/0
- ip address 20.0.0.10 255.0.0.0
- encapsulation PPP
- clock rate 64000
- no shut

Open CLI of router 1:

- enable
- config t
- interface serial 1/0
- ip address 20.0.0.20 255.0.0.0
- encapsulation PPP
- no shut

The connection will turn green.

Open CLI of router 1 again:

- enable
- config t
- interface serial 3/0
- ip address 30.0.0.10 255.0.0.0

→ encapsulation PPP

→ clock rate 64000

→ no shut.

Open CLI of router 2:

→ enable

→ config +

→ interface serial 2/0

→ ip address 30.0.0.20 255.0.0.0

→ encapsulation PPP

→ no shut.

Now,

Open CLI of router 0 and enter the following commands:

(Config) # router ip

config - router # network 10.0.0.0

config - router # network 20.0.0.0

config - router # exit

→ show ip route.

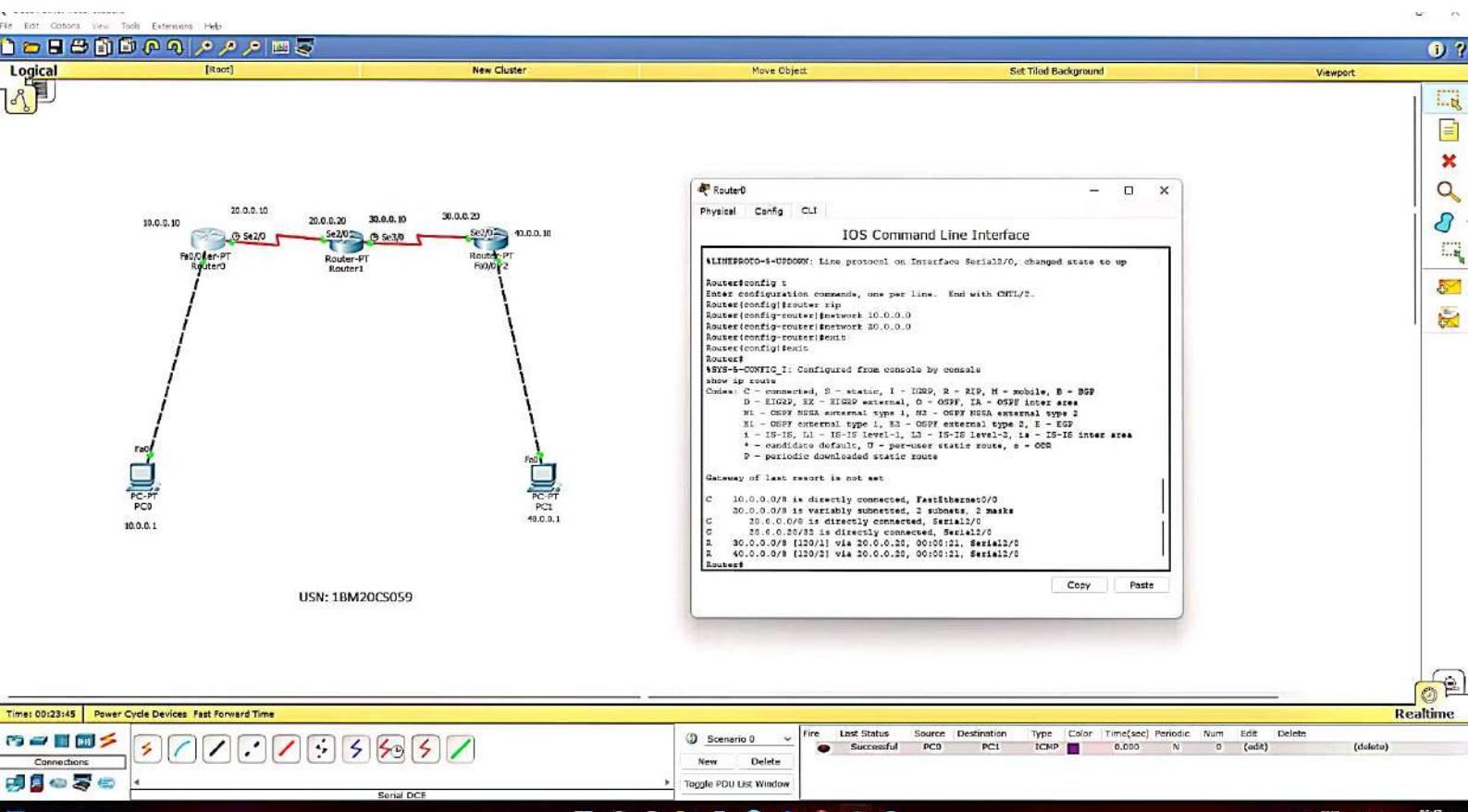
Similarly repeat for router 1 and router 2 with networks 20 & 30 and 30 & 40.

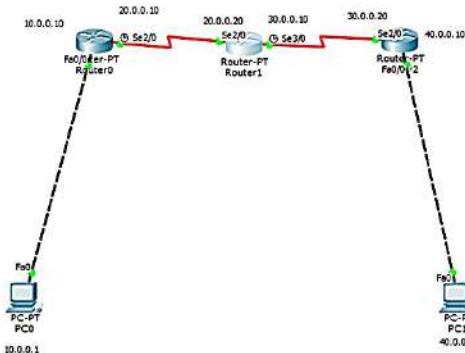
Simulation mode: Add a simple PDU by selecting the PCs and click on auto capture from right panel.

Real time mode: Select PC to go to command prompt and select the destination address 40.0.0.1.

4 Observation:

Learning outcome: Routing information protocol is a protocol that routers use to exchange information of topology among the network.





```
Router1
Physical Config CLI
IOS Command Line Interface

% Invalid input detected at '^' marker.

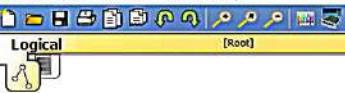
Router>enable
Router#config t
Enter configuration commands, one per line. End with CNTL/Z.
Router(config)#exit
Router#
%SYS-4-CONFIG_I: Configured from console by console
show ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
      D - EIGRP, EX - EIGRP external, O - OSPF, 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, is - IS-IS inter area
      * - candidate default, o - per-user static route, o - ODR
      ? - dynamically downloaded static route

Gateway of last resort is not set

R  0.0.0.0/0 [110/1] via 00.0.0.10, 00:00:02, Serial1/0
  20.0.0.0/8 is directly connected, 1 subnet, 2 masks
C   30.0.0.0/8 is directly connected, Serial1/0
  30.0.0.0/8 is directly connected, Serial1/0
C   30.0.0.0/8 is variably subnetted, 1 subnets, 2 masks
  30.0.0.0/6 is directly connected, Serial1/0
C   30.0.0.20/32 is directly connected, Serial1/0
R  40.0.0.0/8 [110/1] via 00.0.0.20, 00:00:16, Serial1/0
Routers]

```

USN: 1BM20CS059

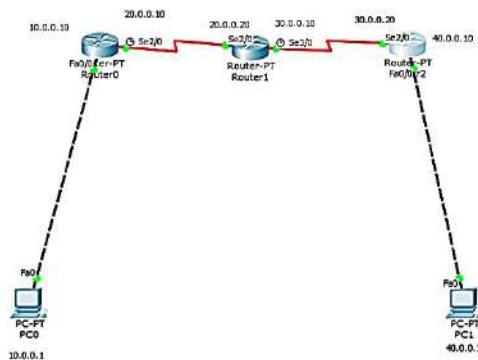


New Cluster

Move Object

Set Tiled Background

Viewport



Router2

Physical Config CLI

IOS Command Line Interface

```

Router2#enable
Router2#config t
Enter configuration commands, one per line. End with CTRL/Z.
Router2#configure|exit
Router2#
#SYS-5-CONFIG_I: Configured from console by console
show ip route
Codes: * - directly connected, S - static, I - OSPF, 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 - EGD
      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 - CDR
      P - periodic downloaded static route

Gateway of last resort is not set
R  10.0.0.0/8 [120/1] via 30.0.0.10, 00:00:17, Serial1/0
R  20.0.0.0/8 [120/1] via 30.0.0.10, 00:00:17, Serial1/0
S  30.0.0.0/16 [1/0] via 30.0.0.10, 00:00:17, Serial1/0
C   30.0.0.0/0 is directly connected, Serial1/0
C   30.0.0.16/32 is directly connected, Serial1/0
C   40.0.0.0/8 is directly connected, FastEthernet0/0
Router2#

```

Copy Paste

USN: IBM20CS059

Timer 00:08:16 Power Cycle Devices Fast Forward Time



Scenario 0	File	Last Status	Source	Destination	Type	Color	Time(sec)	Periodic	Num	Edit	Delete
		Successful	PC0	PC1	ICMP	purple	0.000	N	0	(edit)	(delete)

Toggle PDU List Window

^ ENG INTEL 00:12:2022 06:51

It is used when in place of static IP routing because with the help of rip protocol routing becomes easy when large scale of network is present.

↳ Result:

1. $\text{Pc} \rightarrow \text{ping } 40.0.0.1$

Request timed out

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

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

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

Ping statistics for 40.0.0.1:

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

2. $\text{Pc} \rightarrow \text{ping } 40.0.0.1$

Pinging 40.0.0.1 with 32 bytes of data:

Reply from 40.0.0.1 : bytes = 32 time = 19 ms TTL = 125

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

Reply from 40.0.0.1 : bytes = 32 time = 19 ms TTL = 125

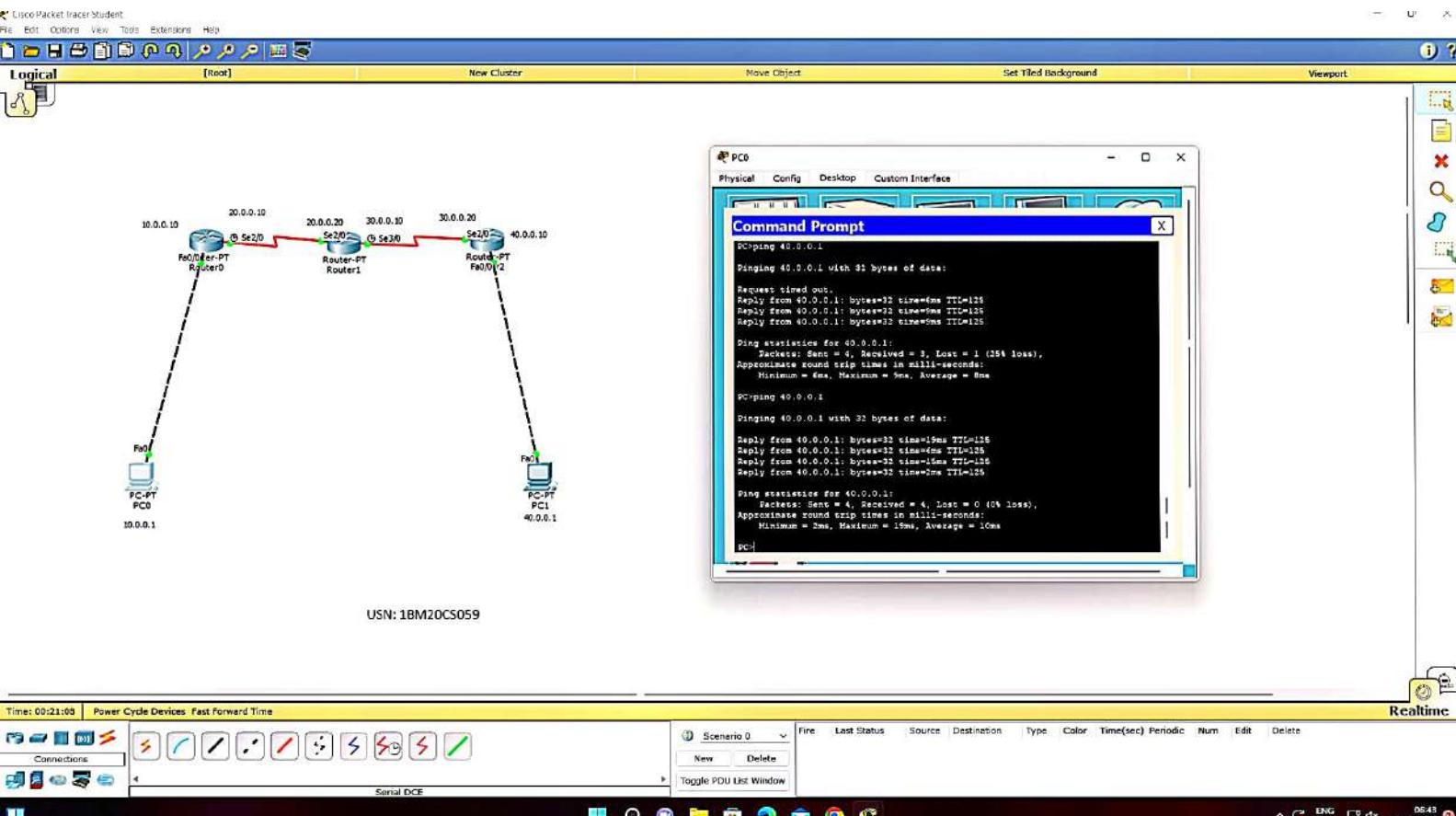
Reply from 40.0.0.1 : bytes = 32 time = 2 ms TTL = 125

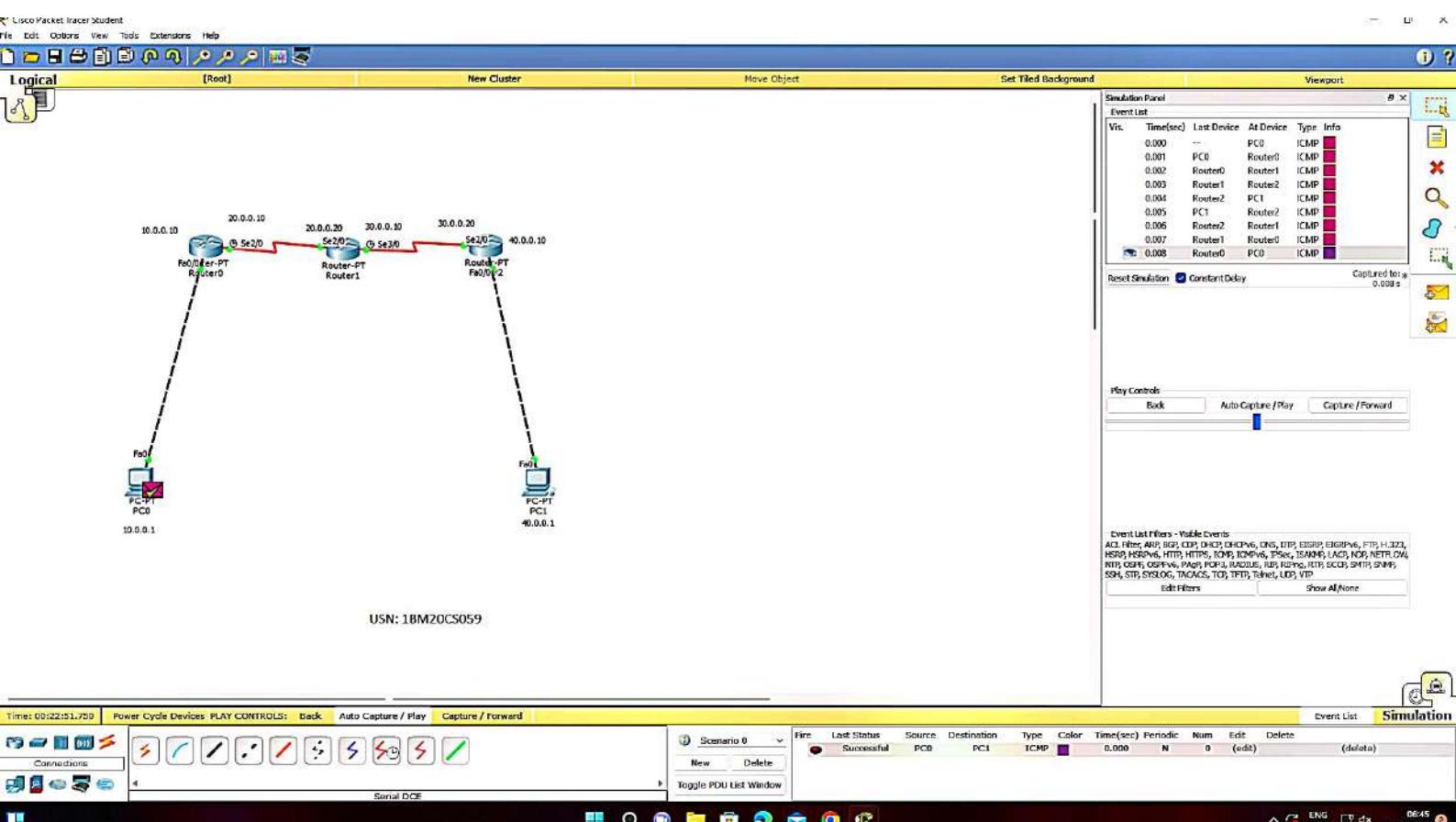
Ping statistics for 40.0.0.1:

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

Approximate round trip times in milliseconds:

Minimum = 2 ms, Maximum = 19 ms, Average = 10 ms.

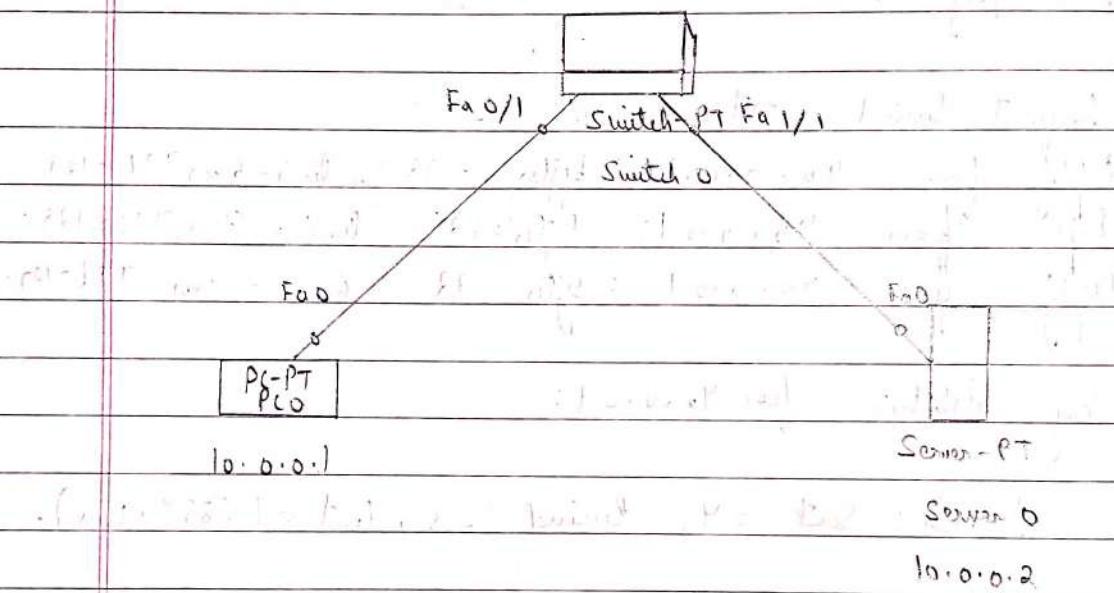




Lab: Week 6 Domain Name Service

↳ Aim: Demonstration of web server and DNS using packet tracer.

↳ Topology:



↳ Procedure:

- (i) Place a generic PC, switch, and server in the workspace.
 - (ii) Place the note of ip address under PC as 10.0.0.1 and under server as 10.0.0.2.
 - (iii) Connect the PC to switch and switch to server with copper straight through.
 - (iv) Set ip address of PC as 10.0.0.1 and server as 10.0.0.2.
- Click on PC and click on desktop tab

Open the web browser and enter the IP address 10.0.0.9.

- Now, click on server.
- Go to services
- Under HTTP go to index.html. Click on edit and make some changes. Click on save and then 'yes overwirte'.
- Now go to PC and under web browser :-
 - (i) In the url give the ip address as 10.0.0.2. Click on Go. It will display the changes made in the index.html.
 - Now, go to the server again.
 - (ii) Under services, go to DNS.
 - Configure name as : csebsce
 - Address : 10.0.0.2
 - Click on Add. Click Save.
- Now, go to PC and under web browser :-
 - (i) In the url enter : csebsce
 - If will display the index.html page.

→ Click on server.

HTTP

Click on new file [bottom right corner].

Enter file name as cv.html.

<html>

<head> <h1> CS STUDENT DETAILS </h1> </head>

<table>

<tr>

<th> Name </th>

<th> Branch </th>

<th> Class </th>

`<th> USN</th>`

`<th> DOB </th>`

`<th>`

`<th>`

`<td> 1 Imadh Ajog </td>`

`<td> S2 </td>`

`<td> S-B </td>`

`<td> IBM20 (S059) </td>`

`<td> 11-08-2001 </td>`

`<th>`

`<td>`

→ Now, go to PC and open the web browser:

- In the url bar, enter `10.0.0.2:csebmsce`. It will show a hyperlink in the index.html page as `cse.html`. Click on `cse.html` and it will display the "Student Details" page.

↳ Observation :

Learning outcomes:

- We can view the webpage when we type "csebmsce" in browser because 10.0.0.2 address is mapped to the name "csebmsce".
- Mapping is required because it's difficult to remember IP's, hence IP is mapped with a name.

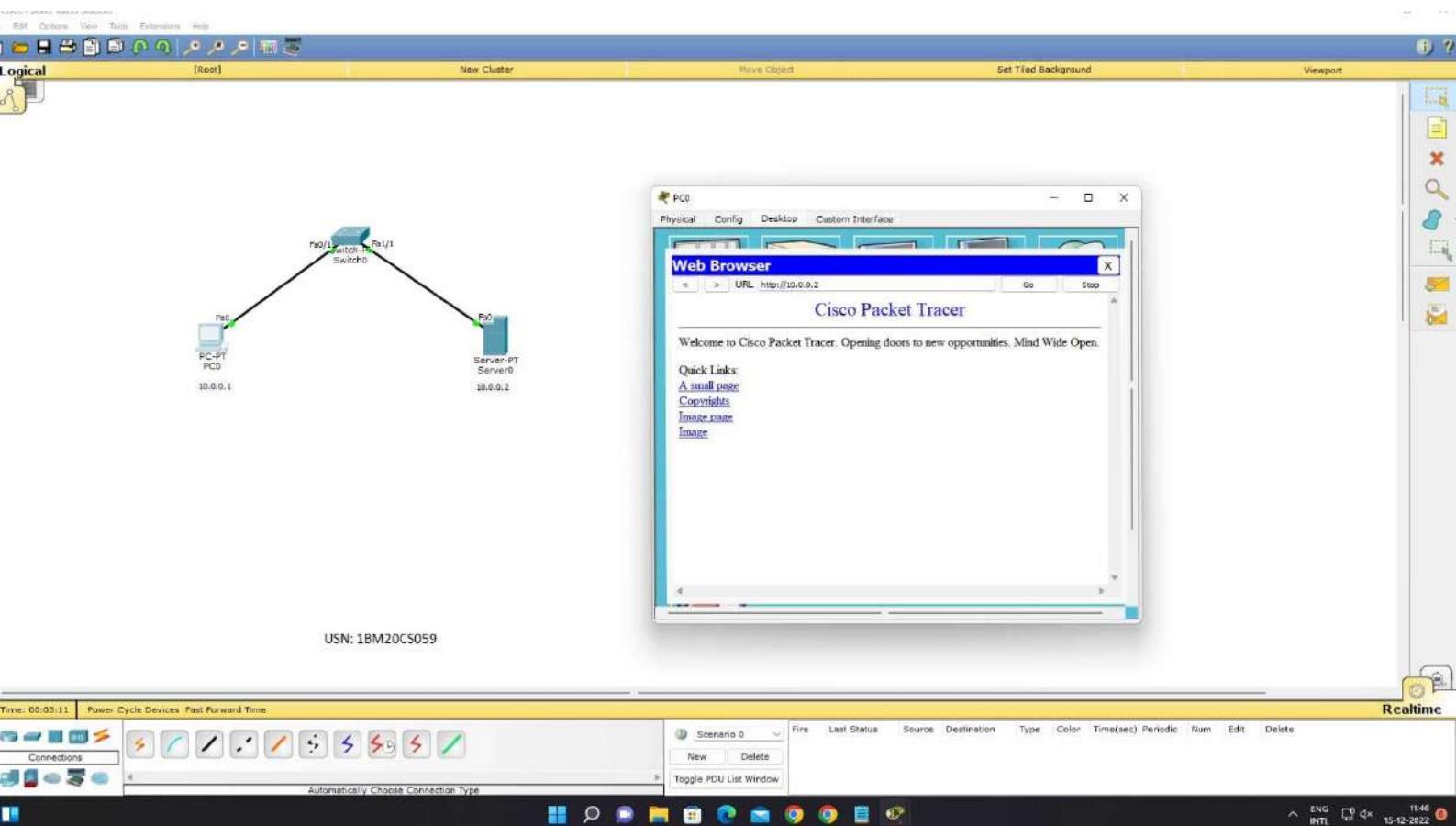
↳ Result :

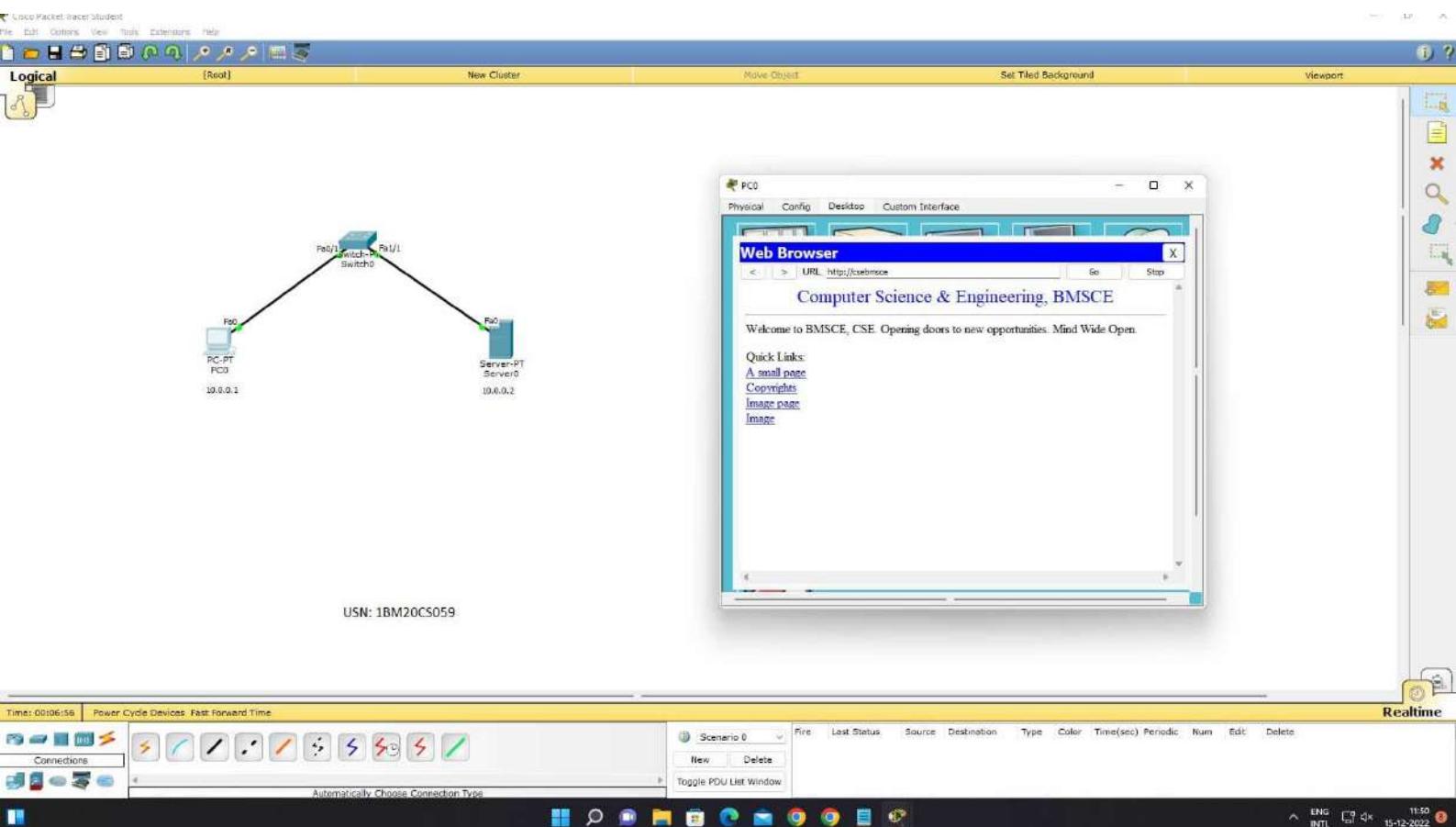
Web Browser

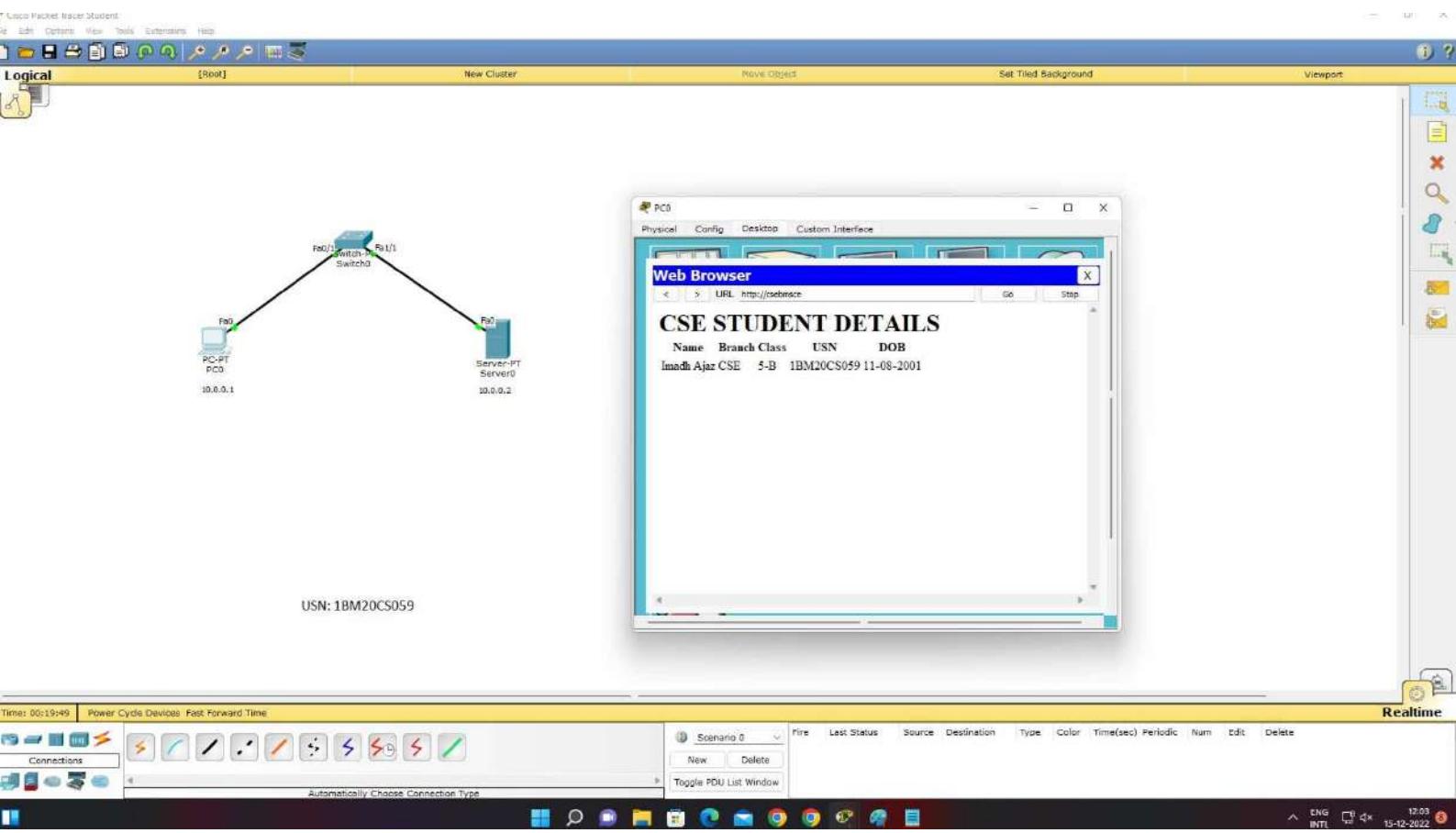
< > URL http://csebmsice/cv.html

Name	Branch	Class	USN	DOB
Jinali Ajay	CSE	5-B	IBM2015059	11-08-2001

✓
N
15/12/22CYCLE I
Completed







Cycle 2

Lab: Week 7

CRC

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

Program:

```
#include <stdio.h>
#include <string.h>
#define N str len (gen-poly)
char data [30];
char check-value [30];
char gen-poly [10];
int data-length, i, j;
void XOR() {
    for (j=1; j<N; j++)
        check-value [j] = ((check-value [j] == gen-poly[j])
            ? '0' : '1');
}
void OR() {
    for (i=0; i<N; i++)
        check-value [i] = data [i];
    do {
        if (check-value [0] == '1')
            XOR();
        for (j=0; j<N; j++)
            check-value [j] = check-value [j+1];
        check-value [0] = data [i+1];
    } while (i <= data-length+N-1);
}
void receiver() {
    printf ("Enter the data received at receiver site : ");
    scanf ("%s", data);
    printf ("Data received : %s", data);
}
```

```

on();
for (i=0; i<N-1) && (check_value[i] != '1'); i++);
    if (i < N-1)
        printf ("\n CRC at receiver site is: %s", check_value);
        printf ("\n Error detected!\n\n");
    else
        printf ("\n CRC at receiver site is: %s",
            check_value);
        printf ("\n No error detected\n\n");
}

```

```

int main ()
{
    printf ("Enter data to be transmitted :");
    scanf ("%s", data);
    printf ("Enter the generating polynomial :");
    scanf ("%s", gen_poly);
    data_length = strlen (data);
    for (i = data_length; i < data_length + N - 1; i++)
        data [i] = '0';
    printf ("\n Padded Data : %s", data);
    on();
    printf ("\n CRC at sender site is: %s", check_value);
    for (i = data_length; i < data_length + N - 1; i++)
        data [i] = check_value [i - data_length];
    printf ("\n Final data to be sent from sender site:
    %s\n", data);
    receiver ();
    return 0;
}

```

$$\begin{array}{cccc} 1 & 0 & 1 & 1 \\ \times & x^3 & +x^2 & +1 \end{array}$$

OUTPUT:

Enter data to be transmitted: 1011010101

Enter the generating Polynomial: 1010 GP

Padded Data: 10110101010000

CRC at sender site: 000

Final data to be sent from sender site: 10110101010000

Enter the data received at receiver site: 10110101010000

Data received: 10110101010000

CRC at receiver site is: 000

No error detected

Enter data to be transmitted: 1011010101

Enter the generating Polynomial: 1010

Padded Data: 10110101010000

CRC at sender site is: 000

Final data to be sent from sender site: 10110101010000

Enter the data received at receiver site: 11110101010000

Data received: 11110101010000

CRC at receiver site is: 010

Error detected!

```
      5 input

Enter data to be transmitted: 1011010101
Enter the Generating polynomial: 1010
Padded Data: 1011010101000
CRC at sender site is: 000
Final data to be sent from sender site: 1011010101000
Enter the date received at receiver site: 1011010101000
Data received: 1011010101000
CRC at receiver site is: 000
No error detected

...Program finished with exit code 0
Press ENTER to exit console.
```

```
Enter data to be transmitted: 1011010101  
Enter the Generating polynomial: 1010  
Padded Data: 1011010101000  
CRC at sender site is: 000  
Final data to be sent from sender site: 1011010101000  
Enter the date received at receiver site: 1111010101000  
Data received: 1111010101000  
CRC at receiver site is: 010  
Error detected!  
  
...Program finished with exit code 0  
Press ENTER to exit console.
```

Lab: week 8

Leaky Bucket

Ques 2: Write a program for congestion control using Leaky Bucket Algorithm.

Program:

```
#include <stdio.h>
int main () {
    int capacity = 0, packet = 0, bsize = 0, rate = 0;
    char ans = 'y';
    printf ("Enter the bucket size:");
    scanf ("%d", &capacity);
    printf ("Enter the leaking rate:");
    scanf ("%d", &rate);
    while (ans == 'y') {
        printf ("nEnter the packet size:");
        scanf ("%d", &packet);
        if ((bsize + packet) > capacity) {
            printf ("n buffer full at the moment");
        } else if ((bsize + packet) <= capacity) {
            bsize += packet;
            bsize -= rate;
            printf ("remaining bucket capacity is %d", bsize);
            printf ("n do you wish to keep adding packets? y/n : ");
            scanf ("%s", &ans);
        }
    }
}
```

OUTPUT:

Enter the bucket size : 500

Enter the leaking rate : 5

Enter the packet size : 250

Remaining bucket capacity is : 245

do you wish to keep adding packets? y/n : y

Enter the packet size : 410

Remaining bucket capacity is 180

do you wish to keep adding packets? y/n : y

Enter the packet size : 300

buffer full at the moment

remaining bucket capacity is 475

do you wish to keep adding packets? y/n : y

Enter the packet size : 500

buffer full at the moment

remaining bucket capacity is 470

do you wish to keep adding packets? y/n : y

Enter the packet size : 550

buffer full at the moment

remaining bucket capacity is 465

do you wish to keep adding packets? y/n : n

input

```
enter the bucket capacity: 500
enter the leaking rate: 5
enter the packet size: 250
remaining bucket capacity is 245
do you wish to keep adding packets? y/n: y
enter the packet size: 240
remaining bucket capacity is 480
do you wish to keep adding packets? y/n: y
enter the packet size: 300
buffer full at the moment
remaining bucket capacity is 475
do you wish to keep adding packets? y/n: y
enter the packet size: 500
buffer full at the moment
remaining bucket capacity is 470
do you wish to keep adding packets? y/n: y
enter the packet size: 550
buffer full at the moment
remaining bucket capacity is 465
do you wish to keep adding packets? y/n: n
...Program finished with exit code 0
Press ENTER to exit console.
```

68: Week 9

Bellman Ford Algorithm & Dijkstra's Algorithm

Q3: Write a program for distance vector algorithm to find suitable path for transmission.

Program:

```
#include <stdio.h>
struct node
{
    unsigned dist [20];
    unsigned from [20];
} st [10];
int main ()
{
    int contmat [20][20];
    int nodes, i, j, k, count = 0;
    printf ("n Enter the number of nodes :");
    scanf ("%d", &nodes);
    printf ("n Enter the cost matrix :n");
    for (i = 0; i < nodes; i++)
    {
        for (j = 0; j < nodes; j++)
        {
            scanf ("%d", &contmat [i] [j]);
            contmat [i] [j] = 20;
            st [i].dist [j] = contmat [i] [j];
            st [i].from [j] = j;
        }
    }
    do
    {
        count = 0;
        for (i = 0; i < nodes; i++)
        {
            for (j = 0; j < nodes; j++)
            {
                if (st [i].dist [j] > contmat [i] [j])
                {
                    st [i].dist [j] = contmat [i] [j];
                    st [i].from [j] = i;
                    count++;
                }
            }
        }
    } while (count != 0);
}
```

```

for (i=0; i < nodes; i++)
for (j=0; j < nodes; j++)
for (k=0; k < nodes; k++)
    if (xt[i].dist[j] > cost_mat[i][k] + xt[k].dist[j])
        {
            xt[i].dist[j] = xt[i].dist[k] + xt[k].dist[j];
            xt[i].from[j] = k;
            count++;
        }
while (count != 0);
for (i=0; i < nodes; i++)
{
    printf("%d\n" for outer loop \n", i+1);
    for (j=0; j < nodes; j++)
        printf("%d\n" for inner loop \n", j+1, xt[i].from[j]+1,
               xt[i].dist[j]);
}
printf("\n");

```

OUTPUT:

Enter the number of nodes : 3

Enter the cost matrix :

0	2	7
2	0	1
7	1	0

For router 1

node 1 via 1 Distance 0

node 2 via 2 Distance 2

node 3 via 2 Distance 3.

For router 2

node 1 via 1 Distance 2

node 2 via 2 Distance 0

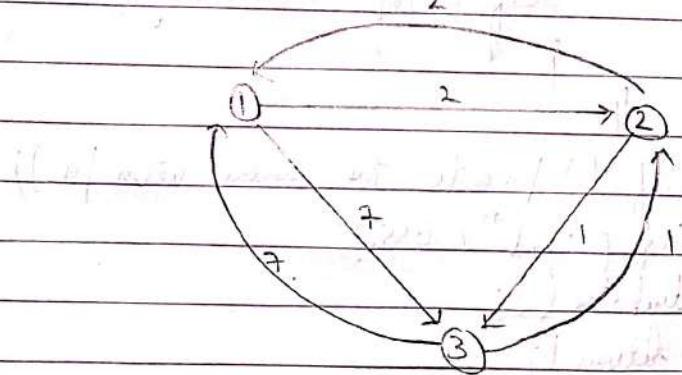
node 3 via 3 Distance 1

For router 3

node 1 via 2 Distance 3

node 2 via 2 Distance 1

node 3 via 3 Distance 0.



Q4: Implement Dijkstra's algorithm to compute the shortest path for a given topology.

Program:

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

int c[10][10], n, sr;
void dijkstra();
int main()
{
    printf("\n enter the number of vertex\n");
    scanf("%d", &n);
    printf("\n enter the cost matrix\n");
    for (int i=1; i<=n; i++)
    {
        for (int j=1; j<=n; j++)
        {
            scanf("%d", &c[i][j]);
        }
    }
    printf("\n enter the source vertex\n");
    scanf("%d", &sr);
    dijkstra();
    return 1;
}

void dijkstra()
{
    int dist[10], vis[10], j, count, min, u;
    for (j=1; j<=n; j++)
    {
        dist[j] = c[sr][j];
    }
}
```

```
for (j=1; j <= n; j++)
```

```
    vis[j] = 0;
```

```
    dist[su] = 0;
```

```
    vis[su] = 1;
```

```
    count = 1;
```

```
    while (count != n)
```

```
        min = 9999;
```

```
        for (j=1; j <= n; j++)
```

```
            if (dist[j] < min && vis[j] != 1)
```

```
                min = dist[j];
```

```
                u = j;
```

```
    vis[u] = 1;
```

```
    count++;
```

```
    for (j=1; j <= n; j++)
```

```
        if (min + c[u][j] < dist[j] && vis[j] != 1)
```

```
            dist[j] = min + c[u][j];
```

```
print("Shortest distance is ");
```

```
for (j=1; j <= n; j++)
```

```
    print("n " + j + " - - -> " + j + " = " + d[n][src[j]][dest[j]]);
```

OUTPUT:

Enter the number of vertices
5

Enter the cost matrix

9999 3 9999 7 9999

3 9999 4 2 9999

9999 4 9999 5 6

7 2 5 9999 4

9999 9999 6 4 9999

Enter the source vertex

1

Shortest distance is

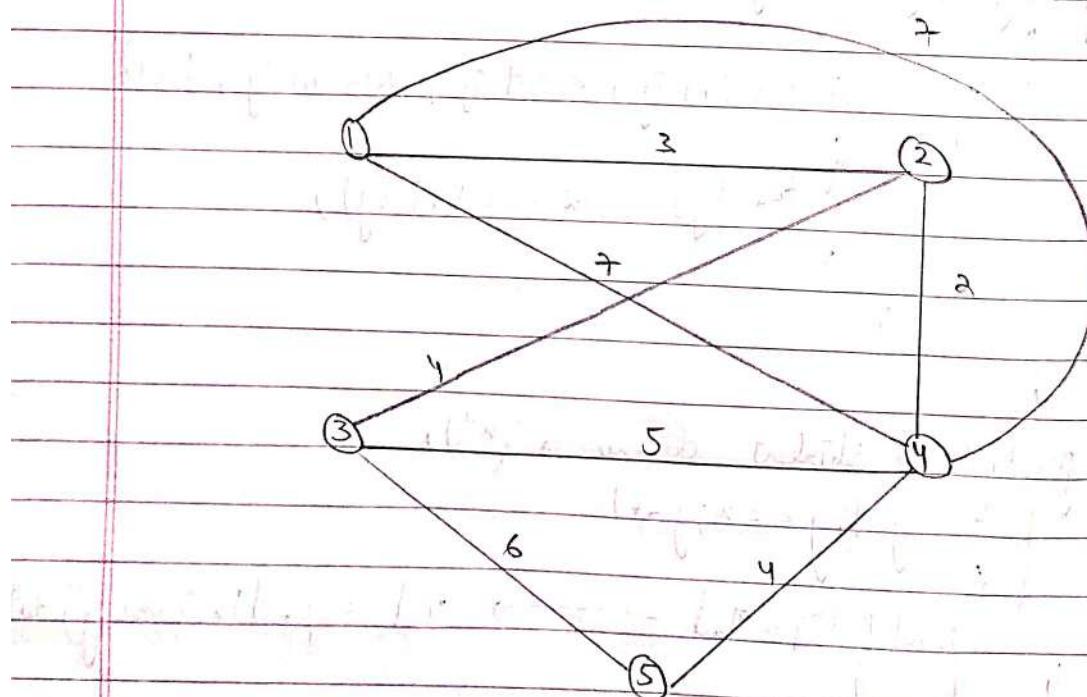
1 - - - - - > 1 = 0

1 - - - - - > 2 = 3

1 - - - - - > 3 = 7

1 - - - - - > 4 = 5

1 - - - - - > 5 = 9.



WEEK 10

TCP / IP Socket:

Exp 5:

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

Code : Client :

```
from socket import *
server_name = 'DESKTOP-9CJ0B77'
server_port = 12530
client_socket = socket(AF_INET, SOCK_STREAM)
Client_Socket = connect((server_name, server_port))
sentence = input("Enter the name")
Client_Socket.send(sentence.encode())
print("From server", file_contents)
Client_Socket.close()
```

Server:

```
from socket import
server_name = 'DESKTOP-9CJ0B77'
server_port = 12530
server_socket = socket(AF_INET, SOCK_STREAM)
server_socket.bind((server_name, server_port))
server_socket.listen(1)
print("The server is ready to receive")
```

while (1):

```
connection_socket, addrs = server_socket.accept()
sentence = connection_socket.recv(1024).decode()
file = open(sentence, "r")
```

```

e = file.read(1024)
connectionSocket.send(b'Hello')
file.close()
connectionSocket.close()

```

OUTPUT:

create a file a.txt
>> Hello I am Imothi

Running Server

server -1) running

MAINT2 -> 30 On client side Today is today work

((both send receive work)) transfer & transfer work

1) user Enter file name a.txt : a.txt

((client send)) Hello I am Imothi

((server receive work)) file

((client send)) transfer & transfer work

((server receive work)) transfer & transfer work

Before transfer work

'FFU OCS PLS WORKS' = server work

'OKOK' = Day work

((MAINT2 -> 30), Today is today work

((both send receive work)) transfer & transfer work

((client send)) transfer & transfer work

((server receive work)) transfer & transfer work

Time & date name = date & Date & Date work

ab((60)) = Date work

((a)) work? Date & Date

```
Enter file name: serverTCP.py
```

```
From Server:
```

```
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()
```

The server is ready to receive

Sent contents of serverTCP.py

The server is ready to receive

█

Exp 6: UDP - File Transfer

Aim: Using UDP's socket. Write a client server program to make client sending the filename and the server to send back. The contents of the requested file if present.

Code:

```

* server udp.py
from socket import *
server Port = 12000
server Socket = socket(AF_INET, SOCK_DGRAM)
server Socket.bind(("127.0.0.1", 12000))
print("The server is ready to receive")
while 1:
    sentence, client Address = server Socket.recvfrom(1024)
    sentence = sentence.decode("UTF-8")
    file = open(sentence, "r")
    fileContent = file.read(1024)
    server Socket.sendto(fileContent, client Address)
    print("File contents of " + sentence)
    file.close()

```

Client :

```

from socket import *
Server Name = "127.0.0.1"
Server Port = 12000

```

```

clientSocket = socket(AF_INET, SOCK_DGRAM)
sentence = input("Enter file name")
clientSocket.sendto(sentence.encode("UTF-8"),
                     (serverName, serverPort))

```

file contents, server Address = client socket.

recvfrom(1048)

```

print("Reply from server")
print("File contents - decode (""UTF-8""))
clientSocket.close()

```

clientSocket.close()

OUTPUT: where is your file

server udp.py

File server is ready to receive file

Sent contents of server udp.py

(server) ready to receive file

(client) ready to receive file

(client) file received

(client) file

(client) file received

(client) file

(client) file

file

→ Prefix before name

→ file is sent name

→ output is host name

```
PS C:\Users\Locesh R\Desktop\socket> & "C:/Program Files/Python311/python.exe" "c:/Users/Lokesh R/Desktop/socket/serverudp.py"
The server is ready to receive
Sent contents of  serverudp.py
```

The server is ready to receive

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")
    l=file.read(2048)

    serverSocket.sendto(bytes(l,"utf-8"),clientAddress)
    print ('\nSent contents of ', end = ' ')
    print (sentence)
    # for i in sentence:
    #     print (str(i), end = '')
    file.close()
```