VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"JnanaSangama", Belgaum -590014, Karnataka.

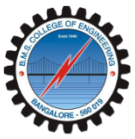
LAB REPORT
on

CN LAB REPORT

Submitted by

RACHANA A (1BM21CS154)

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
June-2023 to Sept-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 "CN LAB" carried out by **RACHANA A (1BM21CS154),** who is a bonafide student of **B. M. S. College of Engineering.** It is in partial fulfillment for the award of **Bachelor of Engineering in Computer Science and Engineering** of the Visvesvaraya Technological University, Belgaum during the year 2023. The Lab report has been approved as it satisfies the academic requirements in respect of a **COMPUTER NETWORKS - (22CS4PCCON)** work prescribed for the said degree.

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

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

`

Index

Experiment No -1

Lab - 01 [09/06/23]

LAN: A series of computers linked together to form a network in a circumscribed location.

WAN: A computer network that connects smaller netw-orks that is not tied to a single location.

Ethernet: A system for connecting a no. of computer systems to form a LAN with protocols to control the passing of information between systems.

Ip address: A unique string of characters that iden-tify each computer using the internet protocol to communicate over a network.

Hub: Hub is a node that broadcasts data to every computer or ethernet based device that is connected to it.

Switch: It connects devices in a network to each other enabling them to talk by exchanging data packets.

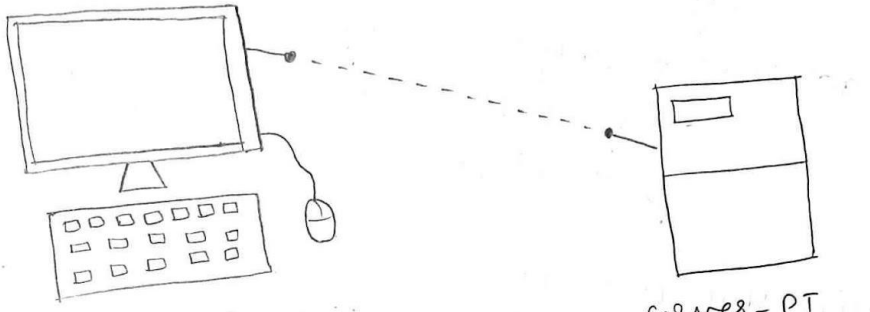
Server: It is a computer program or device that provides a service to another computer pro-gram and its user known as client.

End device: Are either the source or destination of data transmitted over the network.

Node: The connection point among network devices such as routers, printers or switches that can recieve and send data from one end point to another.

packet tracker:

1) Add PC and server from end devices.
2) connect them with copper cross over.
3) set PC ethernet Ip address as 10.0.0.1 and DNS server address as 10.0.0.2
4) set server ethernet Ip address as 10.0.0.2
5) services → DNS → Name : www.first.com
   Address : 10.0.0.2
   Add



observation/output:

click on pc in real time → desktop → command prompt

command:
ping 10.0.0.2

pinging 10.0.0.2 with 32 bytes of data:

Reply from 10.0.0.2: bytes = 32 time = 0ms TTL = 128
Reply from 10.0.0.2: bytes = 32 time = 0ms TTL = 128
Reply from 10.0.0.2: bytes = 32 time = 0ms TTL = 128
Reply from 10.0.0.2: bytes = 32 time = 0ms TTL = 128

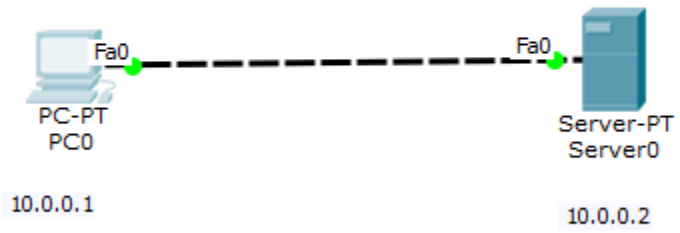
ping statistics for 10.0.0.2:
packets: sent = 4, Recieved = 4, Lost = 0
(0% loss)

Approximate round trip times in milliseconds:

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

16/6

1BM21CS154



Command Prompt

```
Packet Tracer PC Command Line 1.0
PC>PING 10.0.0.2

Pinging 10.0.0.2 with 32 bytes of data:

Reply from 10.0.0.2: bytes=32 time=42ms TTL=128
Reply from 10.0.0.2: bytes=32 time=0ms TTL=128
Reply from 10.0.0.2: bytes=32 time=0ms TTL=128
Reply from 10.0.0.2: bytes=32 time=0ms 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 = 0ms, Maximum = 42ms, Average = 10ms
```

Experiment No-2

Lab-02 16/06/23

Create a topology and simulate sending ~~hence~~ simulate a simple PDU from source to destination using simple hub and switch as connecting domains.

Steps involved:

Step 1: Drag and drop 3 generic PC's and a generic switch. Connect 3 PC's as peripherals to the switch after setting the IP addresses as 10.0.0.1, 10.0.0.2 and 10.0.0.3 for PC1, PC2 and PC3 respectively and connect them.

Step 2: Drag and drop 3 more generic PC's and a generic hub. Set the IP addresses of PC4, PC5 and PC6 as 10.0.0.4, 10.0.0.5 and 10.0.0.6 respectively. connect all the three 3 PC's to the hub.

Scenario 1:

Step 3: Turn on the switch and send a PDU from PC1 (10.0.0.1) to PC2 (10.0.0.2) via switch.

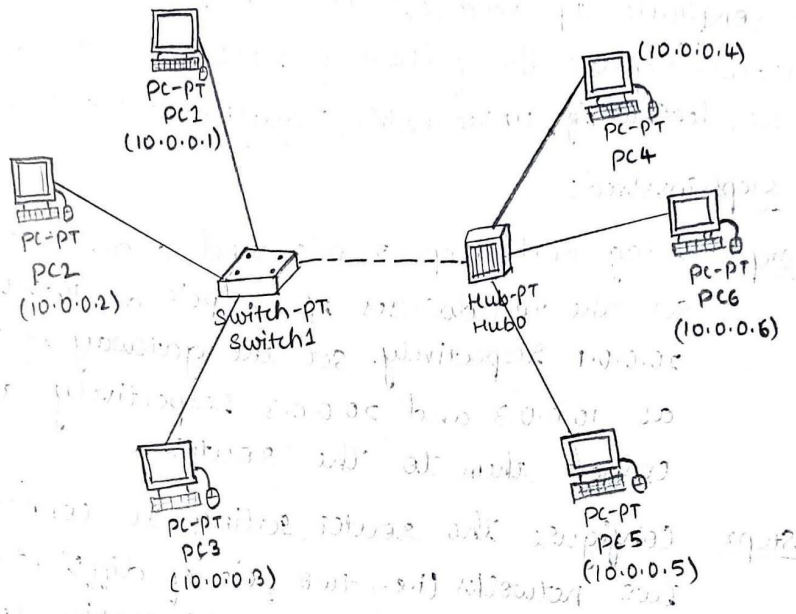
Scenario 2:

Step 4: Send a PDU from PC4 with ip address 10.0.0.4 to PC6 with IP address 10.0.0.6 via hub. Hub will send PDU to every PC connected to it. PC6 will acknowledge and recieve it.

Scenario 3:

Step 5: Connect switch and hub. Send a PDU from PC1 with IP address 10.0.01 to PC6

with Ip address 10.0.0.6 via switch and hub.



Command prompt:

PC1

PC > ping 10.0.0.6

pinging 10.0.0.6: with 32 bytes of data:

Reply from 10.0.0.6: bytes = 32 time = 6ms TTL = 128

Reply from 10.0.0.6: bytes = 32 time = 6ms TTL = 128

Reply from 10.0.0.6: bytes = 32 time = 6ms TTL = 128

Reply from 10.0.0.6: bytes = 32 time = 6ms TTL = 128

ping statistics for 10.0.0.6:

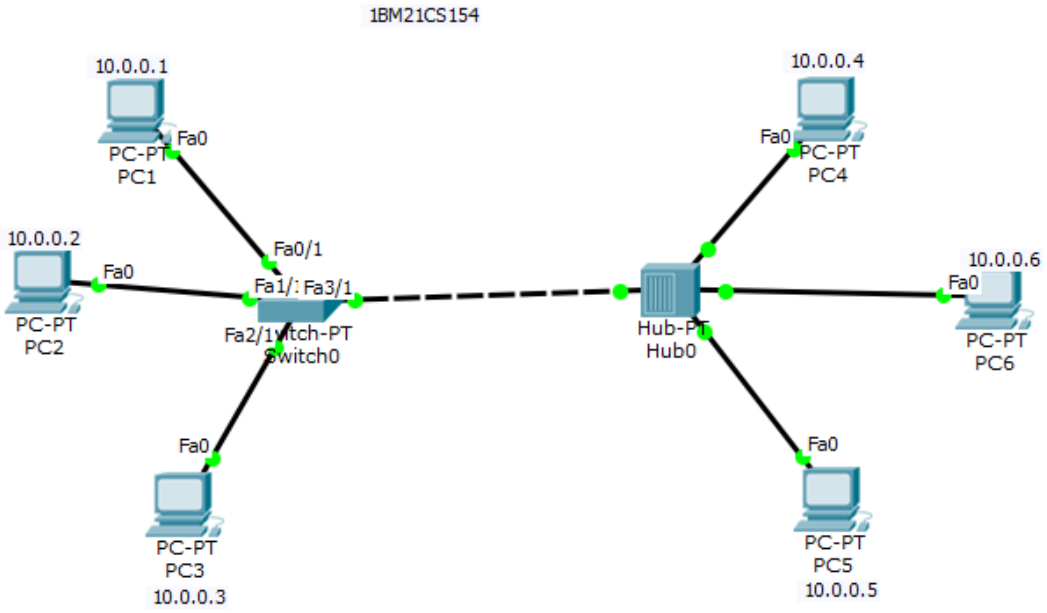
packets: Sent = 4, Received = 4, Lost = 0 (0% loss),

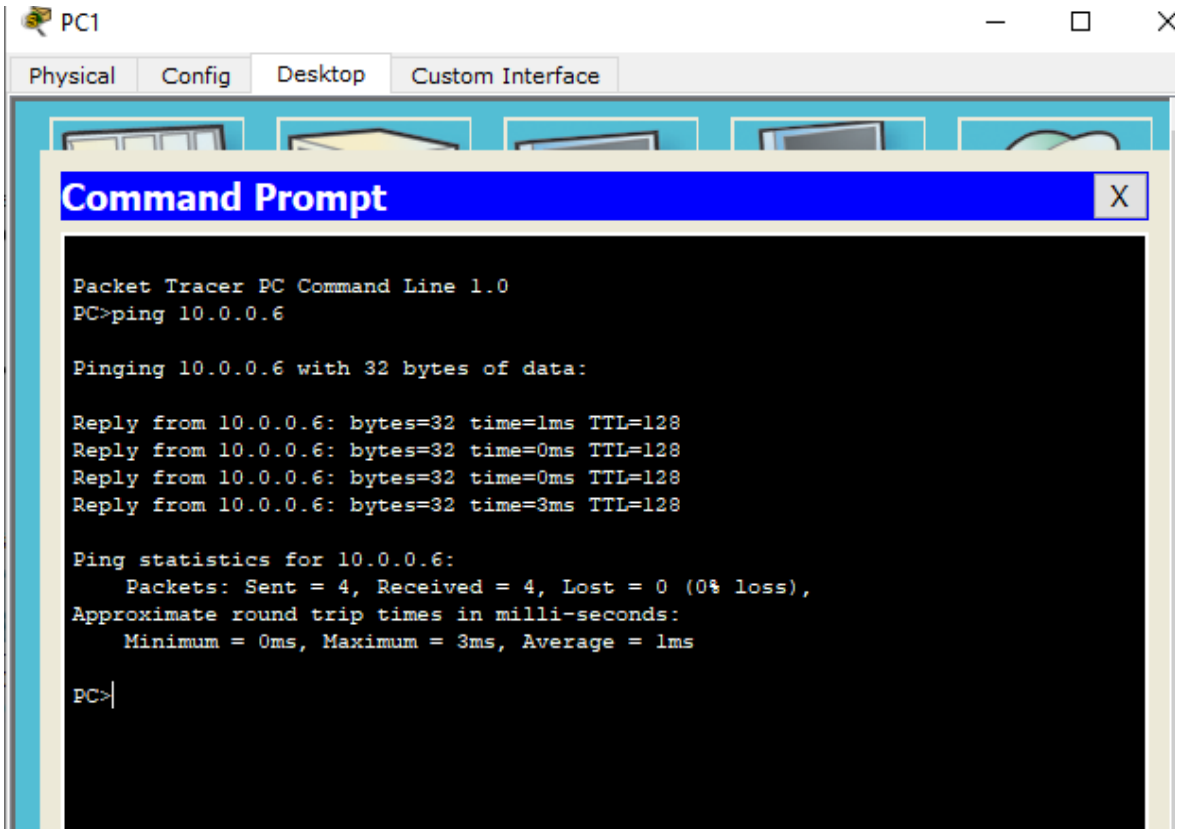
Approximate round trip times in milli-seconds:

Minimum = 6ms, Maximum = 6ms, Average = 6ms

10/10

16/6


1BM21CS154
10.0.0.1
PC-PT
PC1
Fa0
10.0.0.4
Fa0
PC-PT
PC4
10.0.0.2
Fa0
PC-PT
PC2
Fa0/1
Fa1/: Fa3/1
Fa2/1
Switch-PT
Switch0
Hub-PT
Hub0
10.0.0.6
Fa0
PC-PT
PC6
Fa0
PC-PT
PC3
10.0.0.3
Fa0
PC-PT
PC5
10.0.0.5


PC1
Physical
Config
Desktop
Custom Interface
Command Prompt
X
Packet Tracer PC Command Line 1.0
PC>ping 10.0.0.6
Pinging 10.0.0.6 with 32 bytes of data:
Reply from 10.0.0.6: bytes=32 time=1ms TTL=128
Reply from 10.0.0.6: bytes=32 time=0ms TTL=128
Reply from 10.0.0.6: bytes=32 time=0ms TTL=128
Reply from 10.0.0.6: bytes=32 time=3ms TTL=128
Ping statistics for 10.0.0.6:
Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
Minimum = 0ms, Maximum = 3ms, Average = 1ms
PC>

Experiment No-3

Lab-03 23/06/2023

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

Steps involved:

Step 1: Drag and drop 2 pc's and a generic router. Set the Ip addresses of 2 pc's as 10.0.0.1 and 20.0.0.1 respectively. Set the gateway of 2 pc's as 10.0.0.3 and 20.0.0.3 respectively and connect them to the router.

Step 2: Configure the router settings to connect the two networks (i.e., two pc's of different n/w) by using following steps after making the connections.

```
Router > enable
Router # config terminal
Router (config) # interface fastethernet 0/0
Router (config-if)# ip address 10.0.0.3 255.0.0.0
Router (config-if)# no shutdown
Router (config-if)# exit
Router (config) # interface fastEthernet 1/0
Router (config-if) # ip address 20.0.0.3 255.0.0.0
Router (config-if)# no shutdown
Router (config-if) # exit
Router (config) # exit
Router #
```

Step 3: Send a simple PDU from pc0 with ip address 10.0.0.1 to pc1 with ip address 20.0.0.1 and confirm how many packets sent by using ping command.

Step 4: Similarly, connect two more pc's with a router and configure by following above mentioned steps. Introduce one more router and connect it to the existing two routers of different network and configure it

Step 5: Now, if you ping from the pc with ip address 10.0.0.1 as > ping 40.0.0.1 the response will be destination unreachable. Although, it seemed there's a connection between these two pc's indirectly via routers, but every router may not have information regarding every network present in the topology so these pc's cannot communicate. To eliminate this, we should use static routing to teach every router manually.

Step 6: We can do static routing for router2 by the following steps:

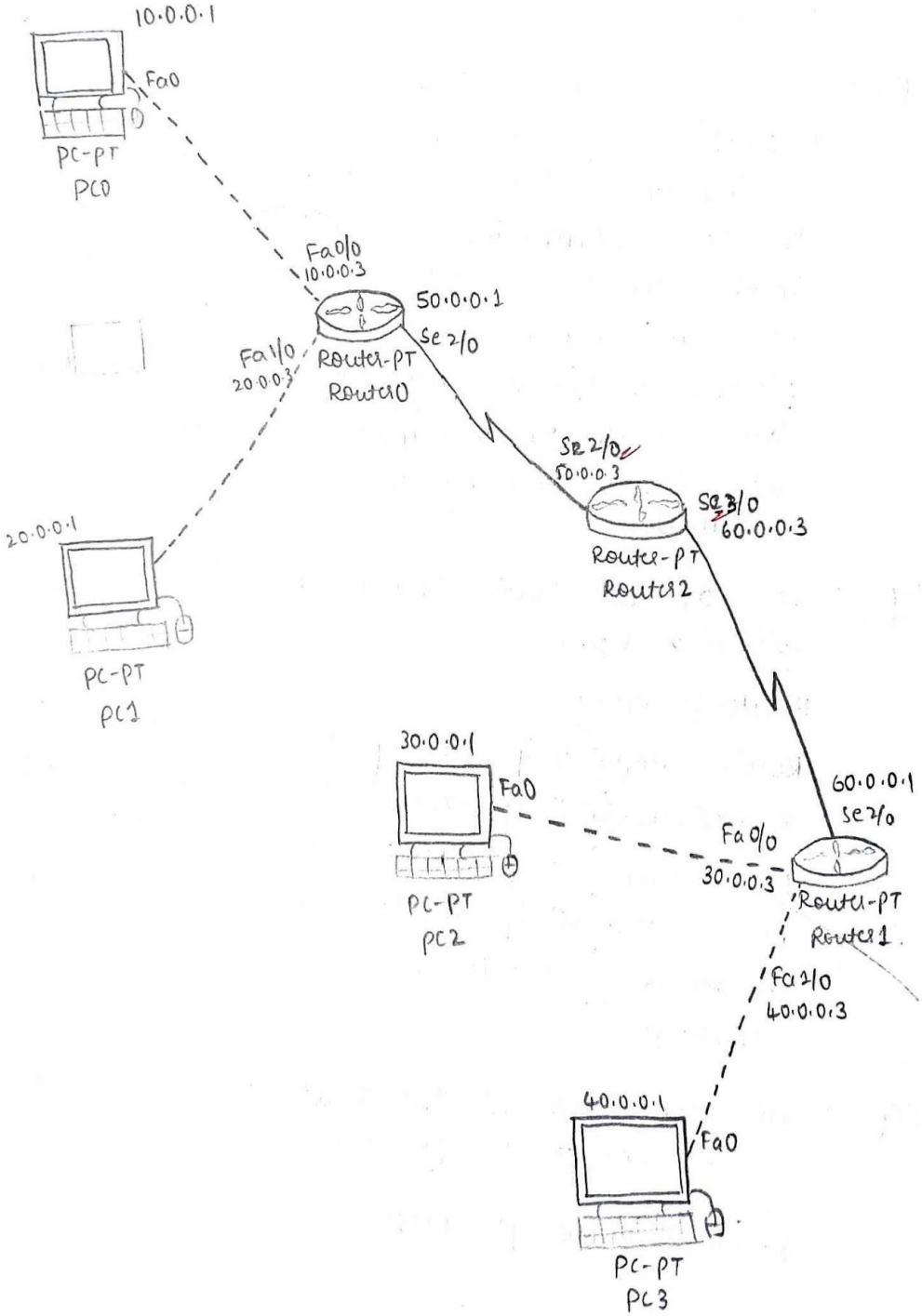
```
Router # config t
Router (config) # ip route 10.0.0.0 255.0.0.0 50.0.0.1
Router (config) # ip route 20.0.0.0 255.0.0.0 50.0.0.1
Router (config) # ip route 30.0.0.0 255.0.0.0 60.0.0.1
Router (config) # ip route 40.0.0.0 255.0.0.0 60.0.0.1
Router (config) # exit
Router #
```

Step 7: We can view all the ~~rout~~ networks connected to a router as follows:

Router # show ip route.

codes: C - connected , S - Static

S 10.0.0.0/8 [1/0] via 50.0.0.1
S 20.0.0.0/8 [1/0] via 50.0.0.1
S 30.0.0.0/8 [1/0] via 60.0.0.1
S 40.0.0.0/8 [1/0] via 60.0.0.1
C 50.0.0.0/8 is directly connected, Serial 2/0
C 60.0.0.0/8 is directly connected, Serial 2/0



Before making static route,
from PC2 ping 10.0.0.1

Command prompt:

PC > ping 10.0.0.1

Pinging 10.0.0.1 with 32 bytes of data:

Reply from 30.0.0.3 : Destination host unreachable
Reply from 30.0.0.3 : Destination host unreachable
Reply from 30.0.0.3 : Destination host unreachable
Request timed out.

ping statistics for 10.0.0.1:

Packets : Sent = 4, Recieved = 0, Lost = 4 (100% loss).

After static route,
from PC1 ping PC3
Command prompt:

PC > ping 40.0.0.1

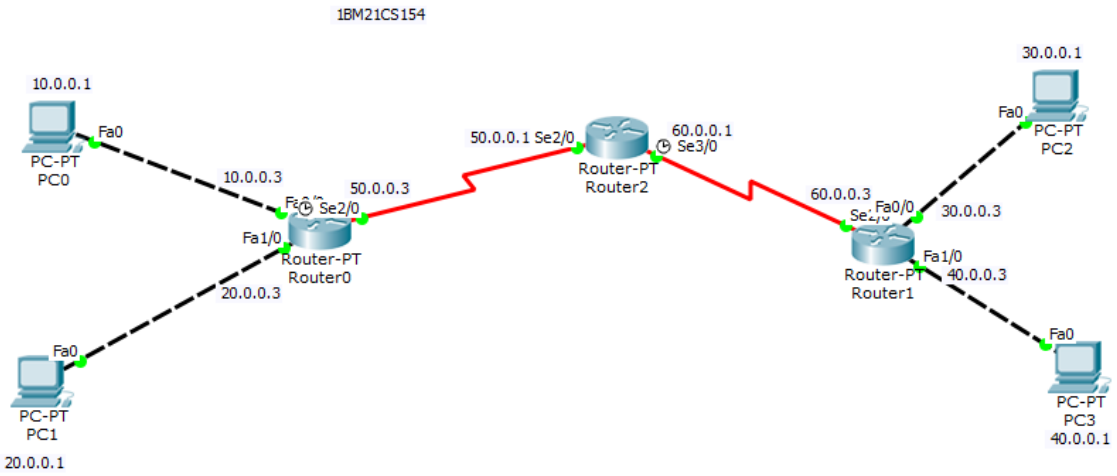
Pinging 40.0.0.1 with 32 bytes of data:

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

ping statistics for 40.0.0.1:

Packets : Sent = 4, Recieved = 3, Lost = 1 (25% Loss)

26/6/23
10/10



Command Prompt

```
    Minimum = 0ms, Maximum = 0ms, Average = 0ms

PC>ping 40.0.0.1

Pinging 40.0.0.1 with 32 bytes of data:

Reply from 10.0.0.3: Destination host unreachable.
Reply from 10.0.0.3: Destination host unreachable.
Reply from 10.0.0.3: Destination host unreachable.
Reply from 10.0.0.3: Destination host unreachable.

Ping statistics for 40.0.0.1:
    Packets: Sent = 4, Received = 0, Lost = 4 (100% loss),

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=2ms TTL=125
Reply from 40.0.0.1: bytes=32 time=20ms 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 = 3, Lost = 1 (25% loss),
Approximate round trip times in milli-seconds:
    Minimum = 2ms, Maximum = 20ms, Average = 8ms

PC>
```

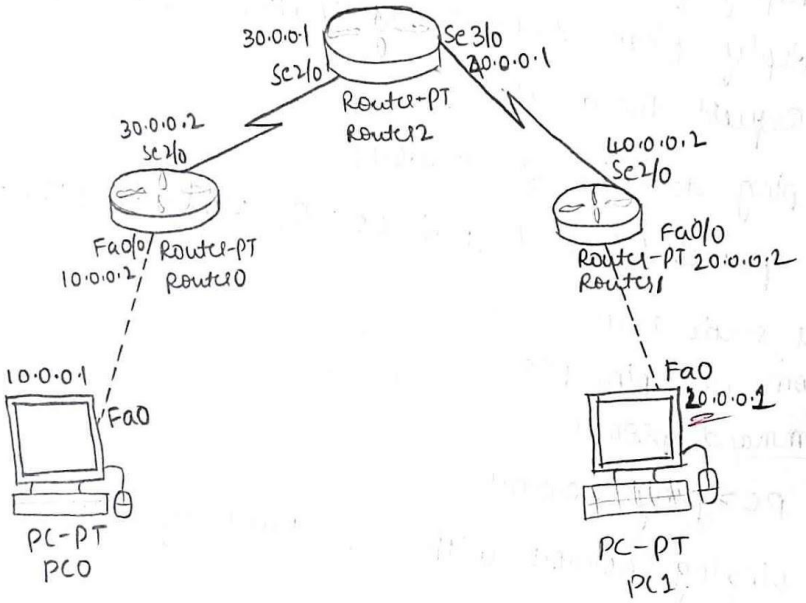
Experiment No-4

Lab-04 | 30/06/2023

Default Routing and Static Routing

Aim: To configure default route, static route to the router.

Topology:



Procedure:

Step 1: Drag and drop 2 PC's and 3 routers to the workspace and connect them as shown in the above figure to construct a topology.

Step 2: Set the ip address of 1st PC0 as 10.0.0.1 and 2nd PC1 as 20.0.0.1. Also, set the gateway of two PCs as 10.0.0.2 and 20.0.0.2 respectively.

Step 3: Place different n/w ip addresses as 30.0.0.1 and 40.0.0.1 to the left and right of Router2 and start configuring the router interfaces.

for Router0,

```
Router > enable
Router # config t
Router (config)# interface fastethernet 0/0
Router (config-if)# ip address 10.0.0.2 255.0.0.0
Router (config-if)# no shut
Router (config-if)# exit
Router (config) # interface serial
Router (config-if)# ip address
Router (config-if)# no shut
Router (config-if)# exit
Router (config)# exit
```

Similarly, configure Router1 and Router2

Step 4: Do the static routing for router2 and default routing for router0 and router1 as -

for Router0,

```
Router > enable
Router # config t
Router(config) # ip route 0.0.0.0 0.0.0.0 30.0.0.1
```

for Router1,

```
Router # config t
Router(config) # ip route 0.0.0.0 0.0.0.0 40.0.0.1
```

for Router2,

```
Router # config t
Router (config) # ip route 10.0.0.0 255.0.0.0 30.0.0.2
Router (config) # ip route 20.0.0.0 255.0.0.0 40.0.0.2
Router (config) # exit
Router #
```

Now, you can check the routing information as follows-

Router0,

Router# show ip route

c - connected s - static * - candidate default

Gateway of last resort is 30.0.0.1 to network 0.0.0.0

C 10.0.0.0/8 is directly connected, FastEthernet 0/0
C 30.0.0.0/8 is directly connected, Serial 2/0
S* 0.0.0.0/0 [1/0] via 30.0.0.1

Router2,

Router# show ip route

C - connected S - static

S 10.0.0.0/8 [1/0] via 30.0.0.2
S 20.0.0.0/8 [1/0] via 40.0.0.2
C 30.0.0.0/8 is directly connected, Serial 2/0
C 40.0.0.0/8 is directly connected, Serial 3/0

ping operations: (Result)

from pc0 (10.0.0.1) ping pc1 (20.0.0.1)

pc> ping 20.0.0.1

pinging 20.0.0.1 with 32 bytes of data:

Reply from 20.0.0.1 : bytes=32 time=2ms TTL=125
Reply from 20.0.0.1 : bytes=32 time=4ms TTL=125
Reply from 20.0.0.1 : bytes=32 time=17ms TTL=125
Reply from 20.0.0.1 : bytes=32 time=25ms TTL=125

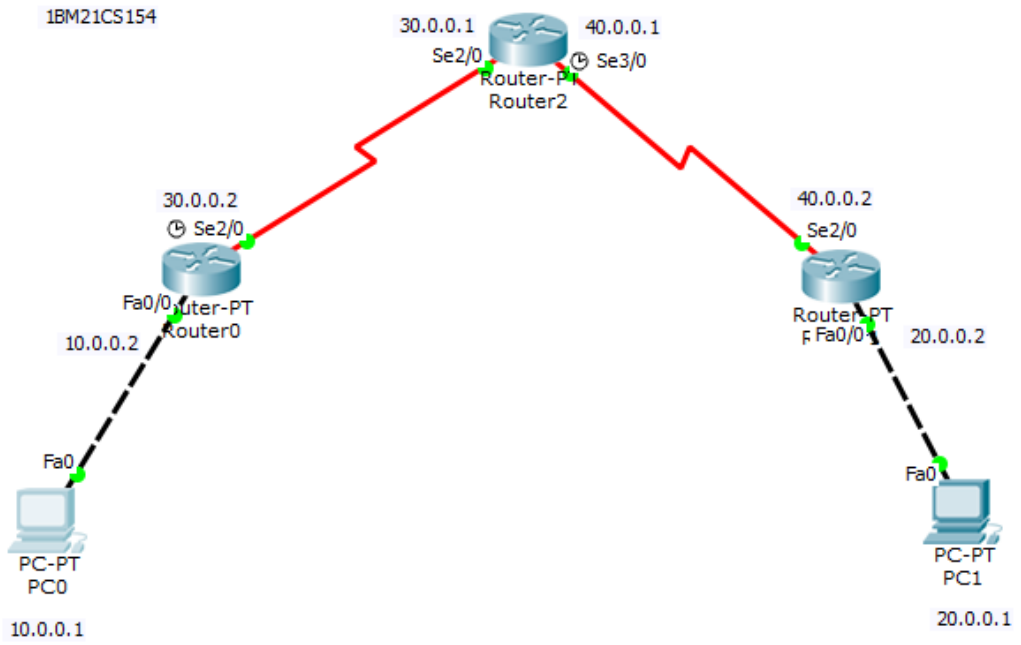
ping statistics for 20.0.0.1:

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

Approximate round trip times in milli-seconds:

Minimum=2ms, Maximum=25ms, Average=12ms

10/10



Command Prompt

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

Pinging 20.0.0.1 with 32 bytes of data:

Request timed out.
Reply from 20.0.0.1: bytes=32 time=13ms TTL=125
Reply from 20.0.0.1: bytes=32 time=17ms TTL=125
Reply from 20.0.0.1: bytes=32 time=2ms TTL=125

Ping statistics for 20.0.0.1:
    Packets: Sent = 4, Received = 3, Lost = 1 (25% loss),
Approximate round trip times in milli-seconds:
    Minimum = 2ms, Maximum = 17ms, Average = 10ms

PC>
```

Experiment No-5

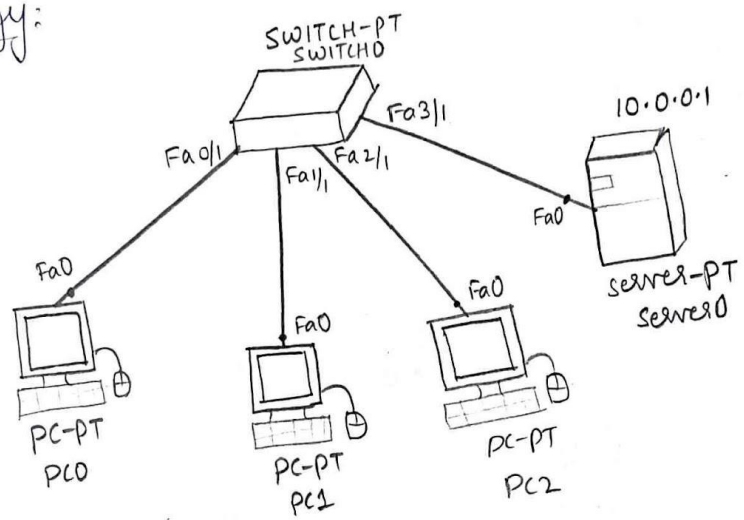
Lab - 05

14/07/2023

DHCP within LAN and Outside LAN

Aim: To configure DHCP within LAN.

Topology:



Procedure:

Step 1: Drag and drop 3 PCs, 1 server and 1 switch to the workspace and connect them as shown in the above topology.

Step 2: Set the ip address of server as 10.0.0.1 and in services tab, turn on the DHCP and create a serverpool ~~sta~~ with start address as 10.0.0.2 and save.

Step 3: In all the PC's go to desktop → Ip configuration and turn on DHCP, the ip address will be assigned as 10.0.0.2, 10.0.0.3 etc...) These ip addresses are provided by the server through DHCP protocol.

Aim: TO configure DHCP outside the LAN

Topology:

procedure:

Step 4: To the previous topology, connect 2 routers 2 switches and 4 pcs as shown in above topology and set the gateway of server as 10.0.0.25

Step 5: Configure the router0 by using following commands -

```
router > enable
router # config t
router (config) # interface fastethernet 4/0
router (config-if) # ip address 10.0.0.25 255.0.0.0
router (config-if) # no shut
router (config-if) # exit
router (config) # interface fastethernet 0/0
router (config-if) # ip address 20.0.0.1 255.0.0.0
router (config-if) # no shut
router (config-if) # exit
router (config) # interface serial 2/0
router (config-if) # ip address 30.0.0.25 255.0.0.0
router (config-if) # no shut
router (config) #
```

Step 6: Configure the router1 by using following commands -

```
router > enable
router # config t
router (config) # interface serial 2/0
router (config-if) # ip address 30.0.0.26 255.0.0.0
router (config-if) # no shut
router (config-if) # exit
router (config) # interface fastethernet 0/0
router (config-if) # ip address 40.0.0.25 255.0.0.0
router (config-if) # no shut
router (config-if) # exit
```

Step 7: Since router0 knows only 10.0.0.0, 20.0.0.0 and 30.0.0.0 networks, we have to perform static routing to connect with 40.0.0.0 network by using following commands.

router0

router > enable
router # config t
router (config) # ip route 40.0.0.0 255.0.0.0 30.0.0.2
router (config) # exit.

Step 8: For router1 perform default routing by using following commands.

router1

router > enable
router # config t
router (config) # ip route 0.0.0.0 0.0.0.0 30.0.0.25
router (config) # exit.

Step 9: For router0 and router1 ~~set~~ configure the ip helper address as ~~server's~~ address.

router0

router (config) # interface fastethernet 0/0
router (config-if) # ip helper-address 10.0.0.1
router (config-if) # exit

router1

router (config) # interface fastethernet 0/0
router (config-if) # ip helper-address 10.0.0.1
router (config-if) # exit.

Step 10: Create 2 more serverpools in the server with starting addresses 20.0.0.2 and 40.0.0.2 and gateway 10.0.0.25. And check all pc's ip configuration window by setting to DHCP. the dynamic ip's will be assigned.

Observation:

Ip addresses will be assigned to pcs dynamically by the server through DHCP protocol.

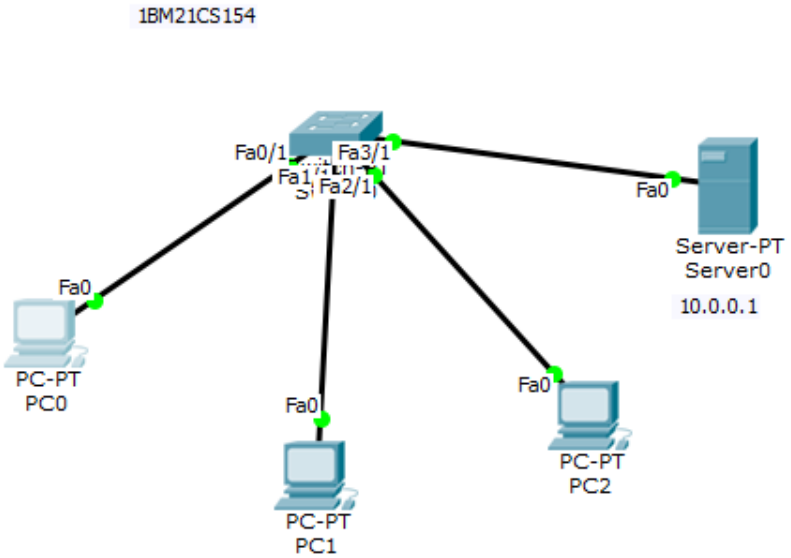
write about default gateway

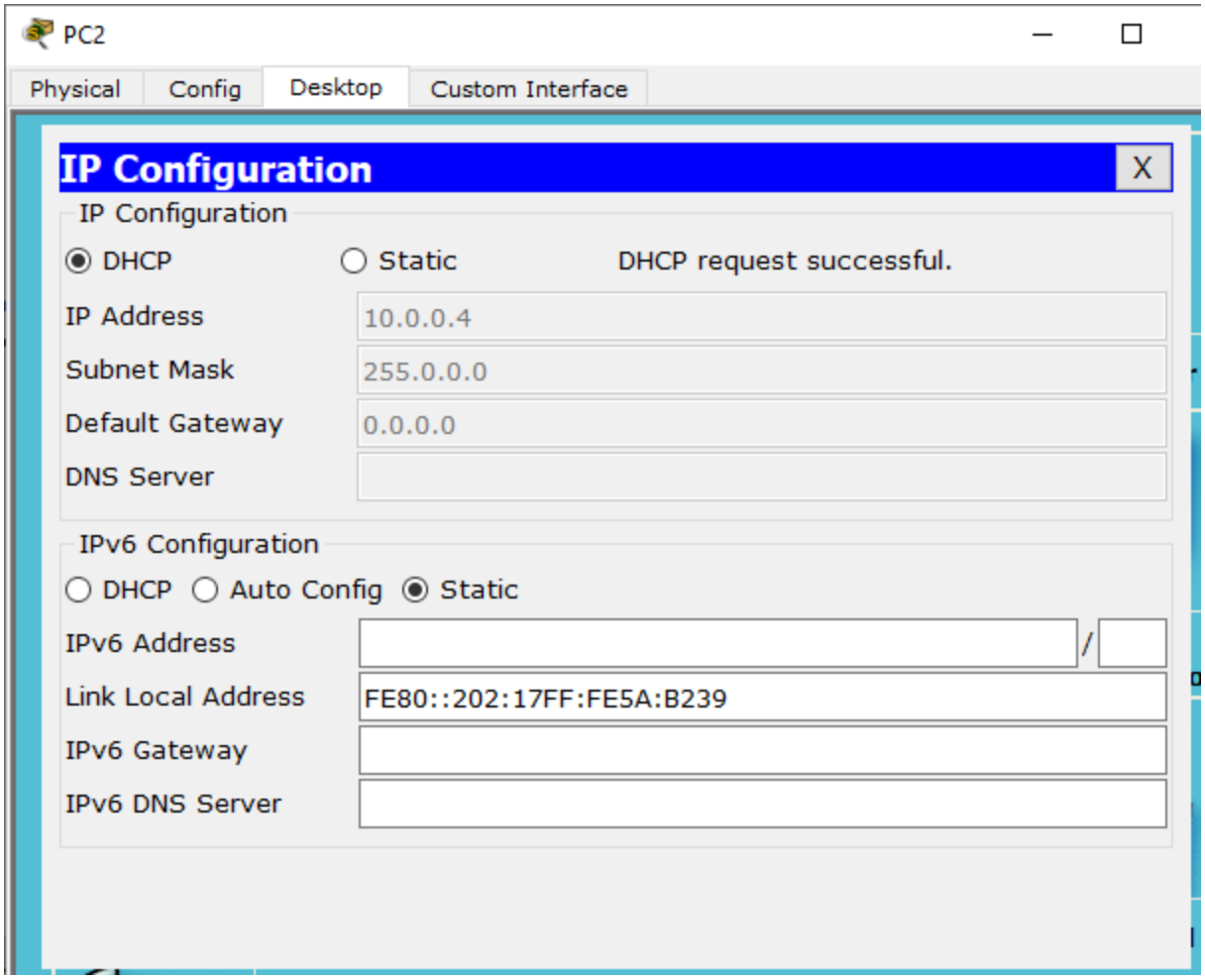
10/10

NL

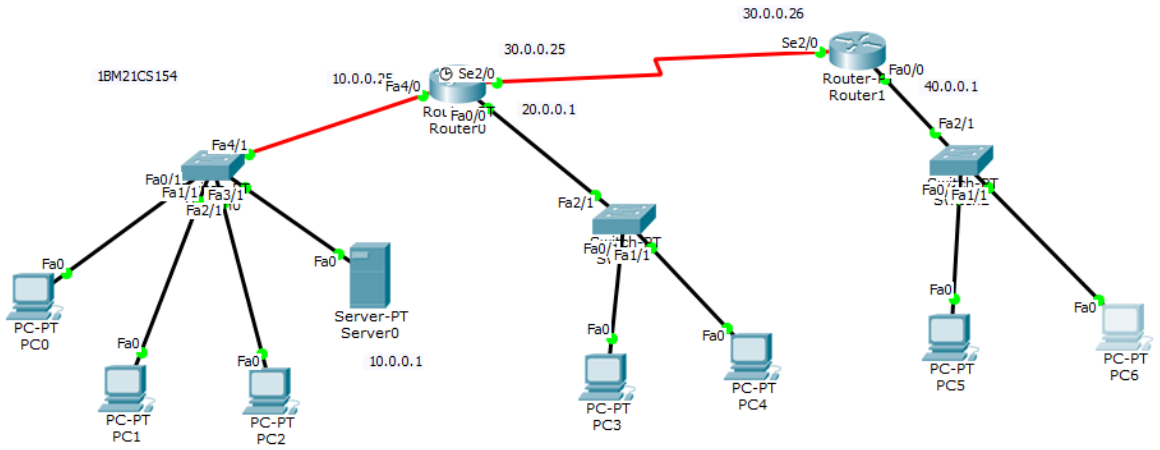
17/7/23

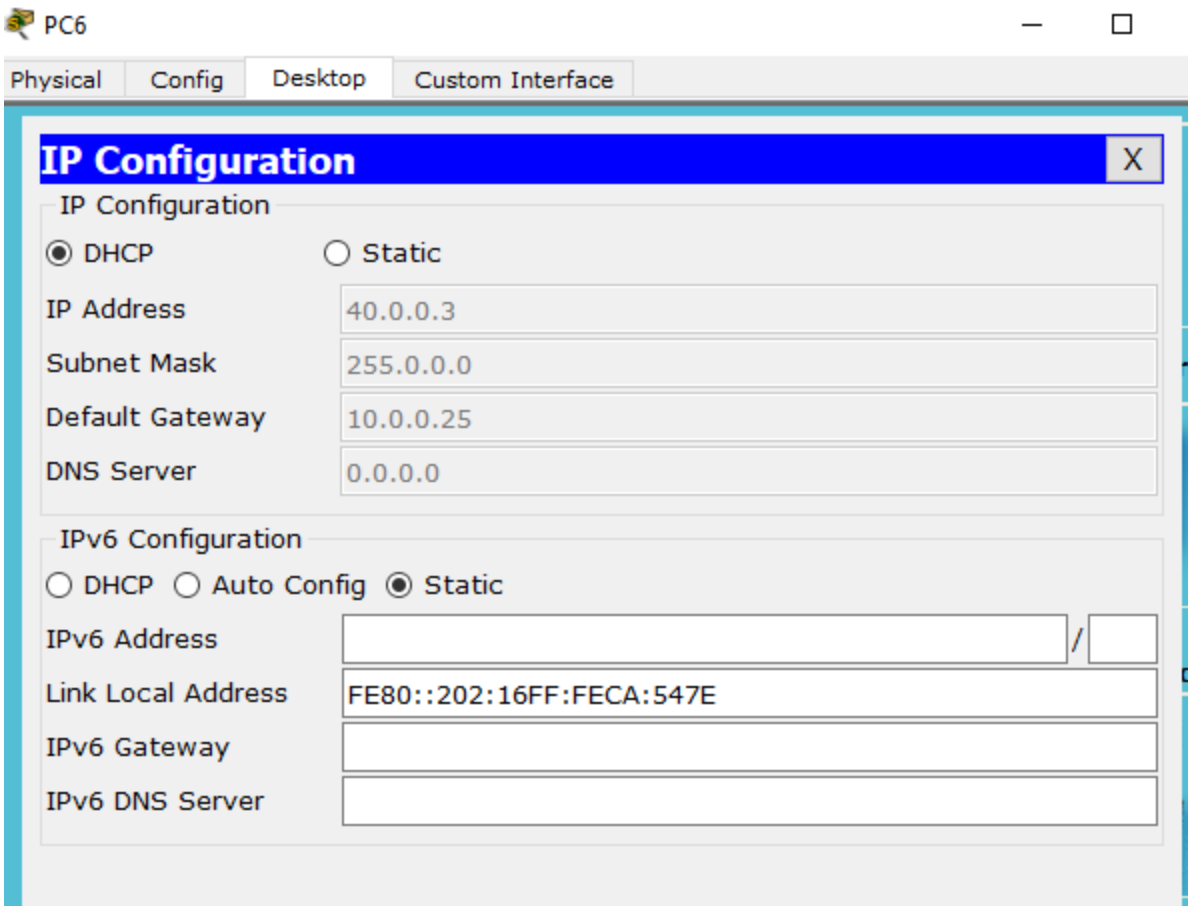
DHCP WITHIN LAN





DHCP OUTSIDE LAN




PC6
Physical
Config
Desktop
Custom Interface
IP Configuration
X
IP Configuration
DHCP
Static
IP Address
40.0.0.3
Subnet Mask
255.0.0.0
Default Gateway
10.0.0.25
DNS Server
0.0.0.0
IPv6 Configuration
DHCP
Auto Config
Static
IPv6 Address
/
Link Local Address
FE80::202:16FF:FECA:547E
IPv6 Gateway
IPv6 DNS Server

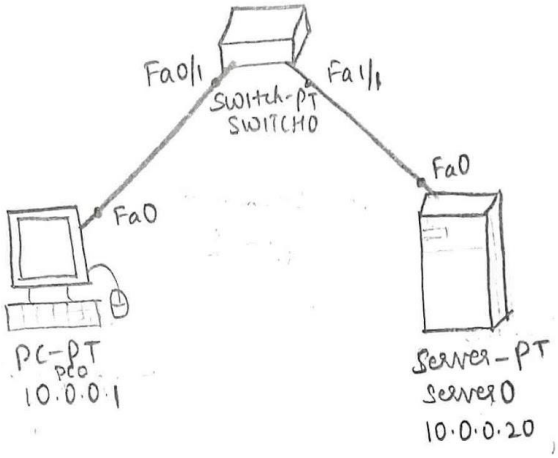
Experiment No-6 ,7

21/07/21

1) Configure web server, DNS within a LAN
2) Configure RIP routing protocol in Routers.

Aim: To configure web server, DNS within a LAN.

Topology:



Procedure:

Step1: Create a topology as shown above using a PC, Server and Switch.

Step2: Set the ip addresses as 10.0.0.1 and 10.0.0.20 for PC and Server respectively.

Step3: In the server, under DNS service create new to example.com website with url 10.0.0.20 and add. Under HTTP, modify the index.html file and add name and USN as

```
<h1> Raihana A </h1>
<h1> 1BM21CS154 </h1>
```

Step4: In PC0, go to destop → web browser and type example.com. You'll be able to see the website with entered name and USN.

Result:

Web Browser

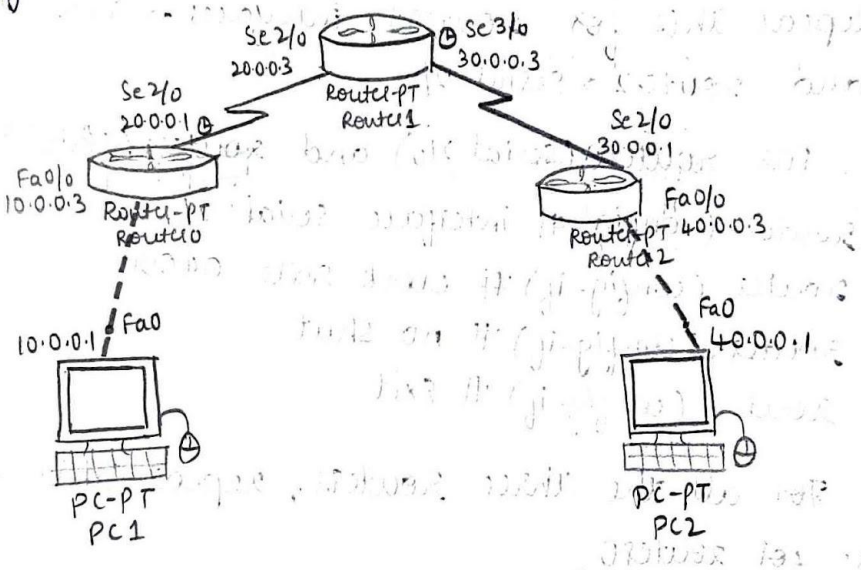
url | http://example.com

cisco packet Tracer

Rachana A
1BM21CS154

Aim: To configure RIP routing protocol in Routers.

Topology:



procedure:

Step 1: create a topology as shown above using 2 pcs and 3 routers.

Step 2: configure the ip addresses of 2 pcs as 10.0.0.1 and 40.0.0.1 for pc1 and pc2 respecti-vely and set the gateways as 10.0.0.3 & 40.0.0.3

Step 3: plan the ips to configure the routers.

for Router0,

```
Router > enable
Router# config t
Router(config)# interface fastethernet 0/0
router(config-if)# ip address 10.0.0.3 255.0.0.0
router(config-if)# no shut
```

router(config)# interface serial 2/0
router(config-if)# ip address 20.0.0.1 255.0.0.0
router(config-if)# no shut.

Similarly, configure the ports of router1 and router

Step 4: For router0,

router(config)# interface serial 2/0
router(config-if)# encapsulation ppp
router(config-if)# no shut
router(config-if)# exit

Repeat this for router1 interfaces => serial 2/0 & 3/0 and router2 => serial 2/0

Step 5: For router0 (serial 2/0) and router1 (serial 3/0),

router(config)# interface serial 2/0
router(config-if)# clock rate 64000
router(config-if)# no shut
router(config-if)# exit

Step 6: For all the three routers, repeat this step.

Ex: for router0,

router > enable
router# config t
router(config)# router rip
router(config-router)# network 10.0.0.0
router(config-router)# network 20.0.0.0

Similary, do this for router1 and router2

then, router# show ip route

This will result in saying that every router knows all the 4 networks in the topology. Now, you can ping from PC1 to PC2 (it works)

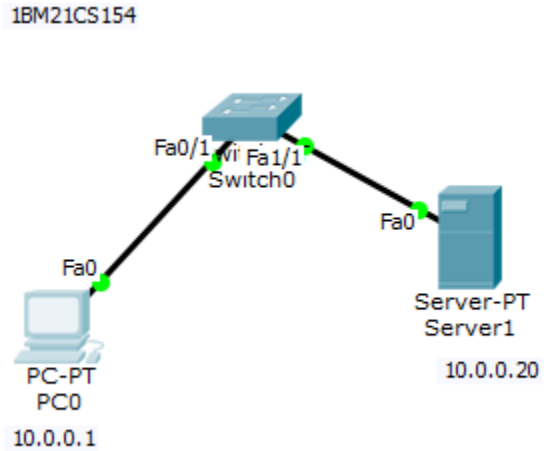
Result: In command prompt of PC1,

```
PC > ping 40.0.0.1

pinging 40.0.0.1 with 32 bytes of data:

Reply from 40.0.0.1 : bytes = 32 time = 12ms TTL = 125
Reply from 40.0.0.1 : bytes = 32 time = 6 ms TTL = 125
Reply from 40.0.0.1 : bytes = 32 time = 2ms TTL = 125
Reply from 40.0.0.1 : bytes = 32 time = 6ms TTL = 125

ping statistics for 40.0.0.1:
    packets : Sent = 4, Recieved = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 2ms, Maximum = 12ms, Average = 6 ms
```

10/10

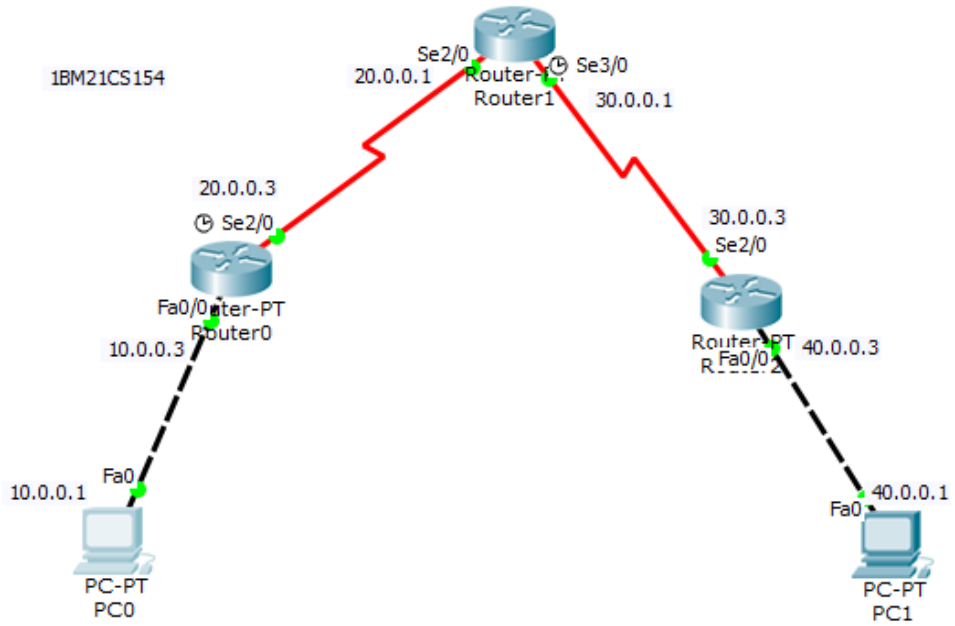
25/7/23

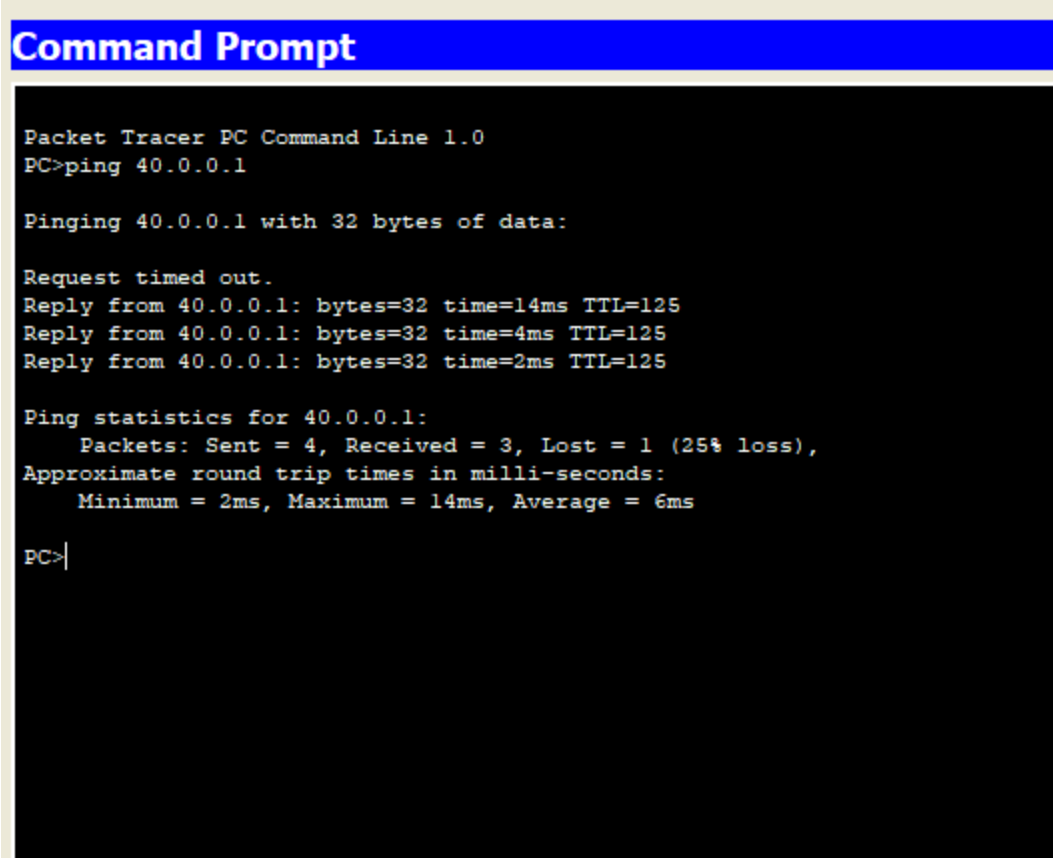
DNS




Web Browser
X
<
>
URL
http://example.com
Go
Stop
Cisco Packet Tracer
Welcome to Cisco Packet Tracer. Opening doors to new opportunities. Mind Wide Open.
Quick Links:
A small page
Copyrights
Image page
Image
RACHANA A
1BM21CS154

RIP

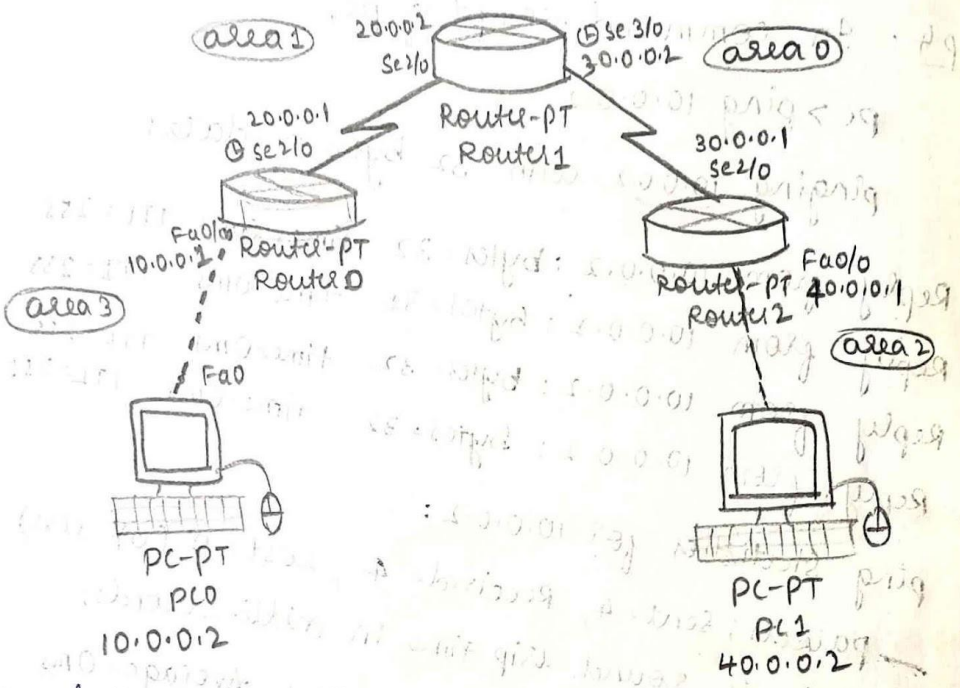

1BM21CS154
Se2/0
20.0.0.1
Router1
Se3/0
30.0.0.1
20.0.0.3
Se2/0
Fa0/0
Router0
10.0.0.3
30.0.0.3
Se2/0
Fa0/0
40.0.0.3
10.0.0.1
Fa0
PC-PT
PC0
40.0.0.1
Fa0
PC-PT
PC1


Command Prompt
Packet Tracer PC Command Line 1.0
PC>ping 40.0.0.1
Pinging 40.0.0.1 with 32 bytes of data:
Request timed out.
Reply from 40.0.0.1: bytes=32 time=14ms TTL=125
Reply from 40.0.0.1: bytes=32 time=4ms 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 = 3, Lost = 1 (25% loss),
Approximate round trip times in milli-seconds:
Minimum = 2ms, Maximum = 14ms, Average = 6ms
PC>

Experiment No-8

Aim: To Configure OSPF routing protocol. and connect areas.

Topology:



Procedure:

Step 1: Create a topology as shown above using 2 PCs and three routers.

Step 2: Configure ip address and gateways for PCs as 10.0.0.2 & 10.0.0.1 for PC0 and 40.0.0.2 & 40.0.0.1 for PC1 respectively.

Step 3: Configure ip address to all ~~int~~ router interfaces,

Router R0,

R0 (config)# interface fastethernet 0/0

R0 (config-if)# ip address 10.0.0.1 255.0.0.0

R0 (config-if)# no shut

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

Similarly, configure for R1 and R2.

Step 4 : Now, Enable ip routing by configuring ospf routing protocol in all routers.

Router R0,

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

Similarly, configure for R1 and R2.

Step 5 : Now check routing table of R0.

Router# show ip route.

C - connected
O - ospf
C 10.0.0.0/8 is directly connected, Fa 0/0
C 20.0.0.0/8 is directly connected, serial 2/0
O IA 40.0.0.0/8 via 20.0.0.2, 00:04:23, serial 2/0
O IA 30.0.0.0/8 via 20.0.0.2, 00:07:29, serial 2/0

Here R1 knows area 0. Network 20.0.0.0 connected to R1 from R0, so R0 learns networks through this network.

Router(config)# router ospf 1, 1 => process id (1-65535)

There must be one interface up to keep ospf process up. So it's better to configure loopback address to routers. It is a virtual interface never goes down once we configured.

R0(config-if)# interface loopback 0

R0(config-if)# ip add 172.16.1.252 255.255.0.0

R0(config-if)# no shut

Similarly, configure for R1 and R2.

Step 6: Now, check routing table of R3

R3# show ip route

codes: O-ospf C-connected

O IA 20.0.0.0/8 via 30.0.0.2, 00:18:58, serial 2/0

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

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

Here, R3 doesn't know about the area3 so we have to create virtual link between R0 and R1

Step 7: Create virtual link between R0, R1 by this we create a virtual link to connect area3 to area 0.

In R0,

R0(config)# router ospf 1

R0(config-router)# area 1 virtual-link 2.2.2.2

In R1, R1(config)# router ospf 1

R1(config-router)# area 1 virtual-link 1.1.1.1

Step8: R1 and R2 get updates about Area3. Now, check routing table of R2

R2 # show ip route

Codes: O-OSPF C-connected.

O IA 20.0.0.0/8 via 30.0.0.1, 00:01:56, serial 2/0

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

O IA 10.0.0.0/8 via 30.0.0.2, 00:01:56, serial 2/0

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

Step9: ping PC1 from PC0,

Result:

In PC0,

PC> ping 40.0.0.2

pinging 40.0.0.2 with 32 bytes of data:

Reply from 40.0.0.2: bytes=32 time=2ms TTL=125

Reply from 40.0.0.2: bytes=32 time=10ms TTL=125

Reply from 40.0.0.2: bytes=32 time=14ms TTL=125

Reply from 40.0.0.2: bytes=32 time=2ms TTL=125

ping statistics for 40.0.0.2:

packets: Sent=4, Recieved=4, Lost=0 (0% loss)

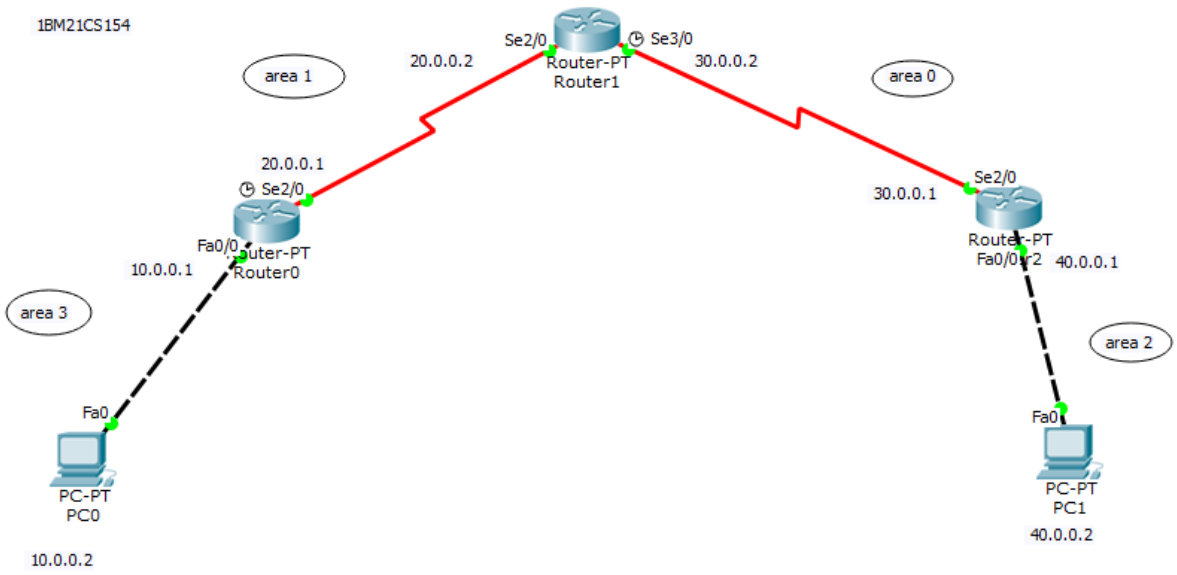
Approximate round trip times in milli-seconds:

minimum: 2ms, Maximum=14ms, Average=7ms.

10/10

28/8/23

```
PC>ping 40.0.0.2

Pinging 40.0.0.2 with 32 bytes of data:

Request timed out.
Reply from 40.0.0.2: bytes=32 time=2ms TTL=125
Reply from 40.0.0.2: bytes=32 time=2ms TTL=125
Reply from 40.0.0.2: bytes=32 time=2ms TTL=125

Ping statistics for 40.0.0.2:
    Packets: Sent = 4, Received = 3, Lost = 1 (25% loss),
Approximate round trip times in milli-seconds:
    Minimum = 2ms, Maximum = 2ms, Average = 2ms

PC>ping 40.0.0.2

Pinging 40.0.0.2 with 32 bytes of data:

Reply from 40.0.0.2: bytes=32 time=2ms TTL=125
Reply from 40.0.0.2: bytes=32 time=10ms TTL=125
Reply from 40.0.0.2: bytes=32 time=14ms TTL=125
Reply from 40.0.0.2: bytes=32 time=2ms TTL=125

Ping statistics for 40.0.0.2:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 2ms, Maximum = 14ms, Average = 7ms
```

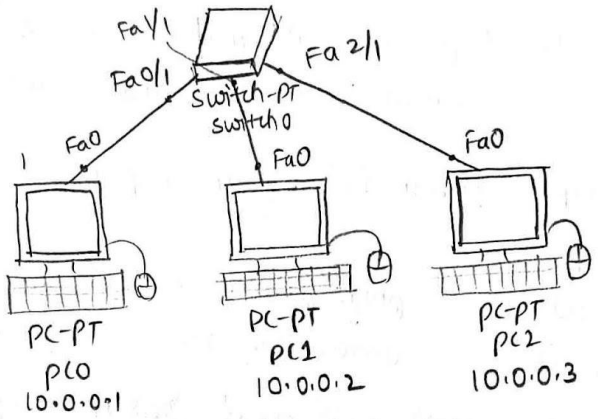


Experiment No-9

2)

Aim: To construct simple LAN and understand the concept of Address resolution protocol (ARP)

Topology:



Procedure:

Step 1: Drag and drop 3 PCs and 1 Switch to the workspace and connect them according to the topology as shown above.

Step 2: Configure the IP addresses for the PC's as 10.0.0.1, 10.0.0.2 and 10.0.0.3 for PC0, PC1 and PC2 respectively.

Step 3: Now, in the command prompt of PC0, if we run the command "arp -a" Initially arp table will be empty.

Step 4: Also in CLI of switch, the command -show mac address-table can be given on every transaction to see how the switch learns from the transactions and build the address-table. Intially all tables are empty.

Step 5: Now ping 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=0ms TTL=128

Reply from 10.0.0.3 : bytes=32 time=0ms TTL=128
Reply from 10.0.0.3 : bytes=32 time=0ms TTL=128
Reply from 10.0.0.3 : bytes=32 time=0ms TTL=128

ping Statistics for 10.0.0.3:

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

Approximate round-trip times in milli-seconds

minimum = 0ms, Maximum = 0ms, Average = 0 ms

Step6 : Run "arp-a" command again, in pc0

pc > arp -a

Internet address	physical address	Type
10.0.0.3	0090.2176.1580	dynamic

Similary ping pc2 from pc0 and rerun arp-a comma -nd,

pc > arp -a

Internet address	physical address	Type
10.0.0.3	0090.2176.1580	dynamic
10.0.0.2	0060.5C26.935d	dynamic

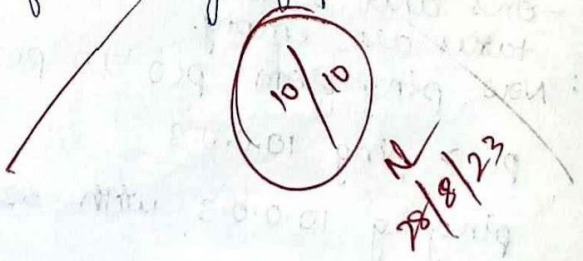
pc > arp -d

pc > arp -a

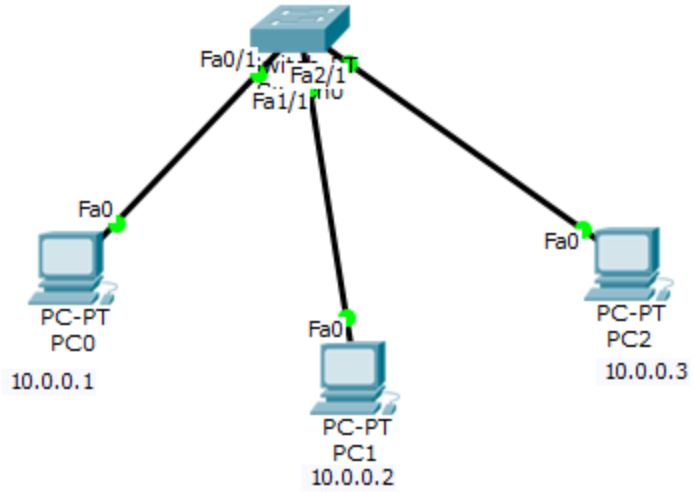
No ARP Enteries found.

"arp -d" command is used to clear the table.

Observation : By using ARP protocol, physical/MAC address of each device will get stored in the table, whenever there is a new transaction. with the help of this dynamically table, swith peforms forwarding of packet.



1BM21CS154



Command Prompt X

```
Packet Tracer PC Command Line 1.0
PC>ping 10.0.0.1

Pinging 10.0.0.1 with 32 bytes of data:

Reply from 10.0.0.1: bytes=32 time=1ms TTL=128
Reply from 10.0.0.1: bytes=32 time=0ms TTL=128
Reply from 10.0.0.1: bytes=32 time=0ms TTL=128
Reply from 10.0.0.1: bytes=32 time=0ms TTL=128

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

PC>arp -a
  Internet Address      Physical Address      Type
  10.0.0.1              00d0.bale.cb8d        dynamic

PC>arp -d
PC>arp -a
No ARP Entries Found
PC>
```

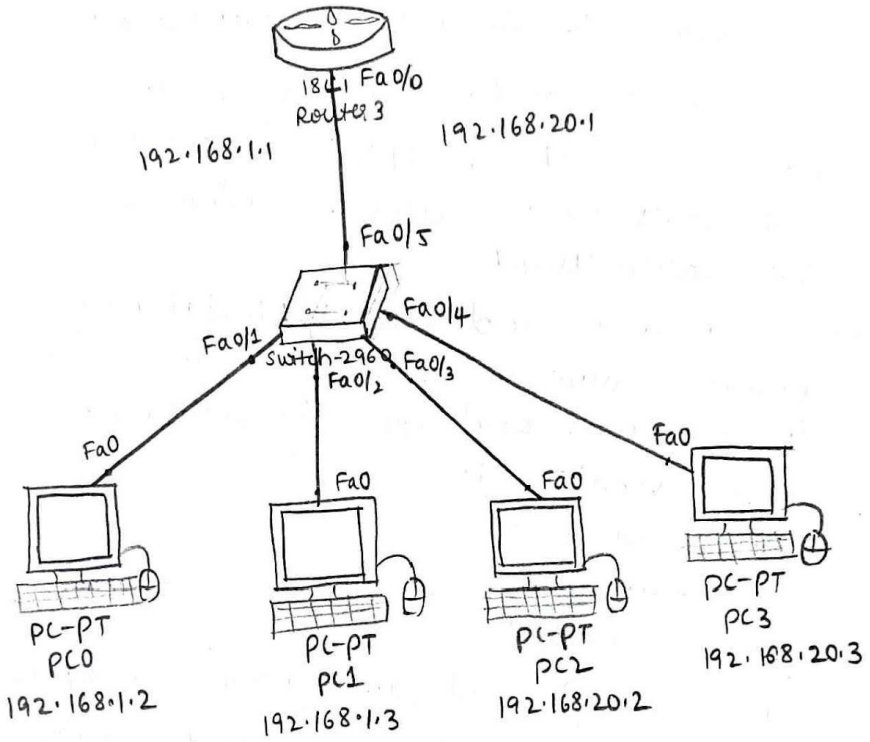
Experiment No-10

11/8/23

Lab-08

Aim: To construct a VLAN and make the PCs communi-cate among a VLAN

Topology:



procedure:

Step 1: Create a topology as given above using 4 PCs, 1 switch (switch-2960) and a router (router-1841)

Step 2: Configure the Ip addresses for pc's as 192.168.1.2 & 192.168.1.3 for PC0 and PC1 and 192.168.20.2 & 192.168.20.3 for PC2 and PC3 respe-ctively.

Step 3: Configure the ip address for router using following commands

```
Router > enable
Router# config t
Router(config)# interface fastethernet 0/0
```

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

Step 4: Set the gateway as 192.168.1.1 for PC0 and PC1 and 192.168.20.1 for PC2 and PC3 respectively.

Step 5: In the switch, go to VLAN database and create/add new VLAN database by giving a name ex: newVlan (2)

Step 6: Now, go to Interface fa 0/5 in the switch and make it trunk, in VLAN everything need to be selected. This allows different VLAN's over single link called trunk.

Step 7: Go to router and select VLAN database. Enter the number & name of VLAN created before, goto CLI in the router and give the following commands

Router (VLAN) # exit
APPLY completed
Exiting.....
Router # config t
Router (config) # interface fastethernet 0/0.1
Router (config-subif) # encapsulation dot1q 2
Router (config-subif) # ip address 192.168.20.1 255.255.255.0
Router (config-subif) # no shut
Router (config-subif) # exit

Step 8: In the switch, make for fa 0/3 and fa 0/4 ~~as~~ select VLAN and number as no given for VLAN while creating (here 2)

Now, ping from PC0 to PC3,

PC0: (In command prompt)

PC> ping 192.168.20.2

pinging 192.168.20.2 with 32 bytes of data:

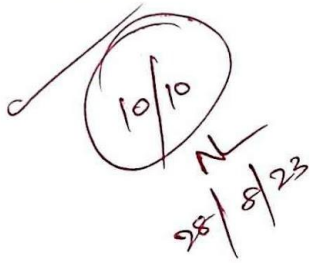
Reply from 192.168.20.2 : bytes = 32 time = 4ms TTL=127
Reply from 192.168.20.2 : bytes = 32 time = 0ms TTL=127
Reply from 192.168.20.2 : bytes = 32 time = 3ms TTL=127
Reply from 192.168.20.2 : bytes = 32 time = 1ms TTL=127

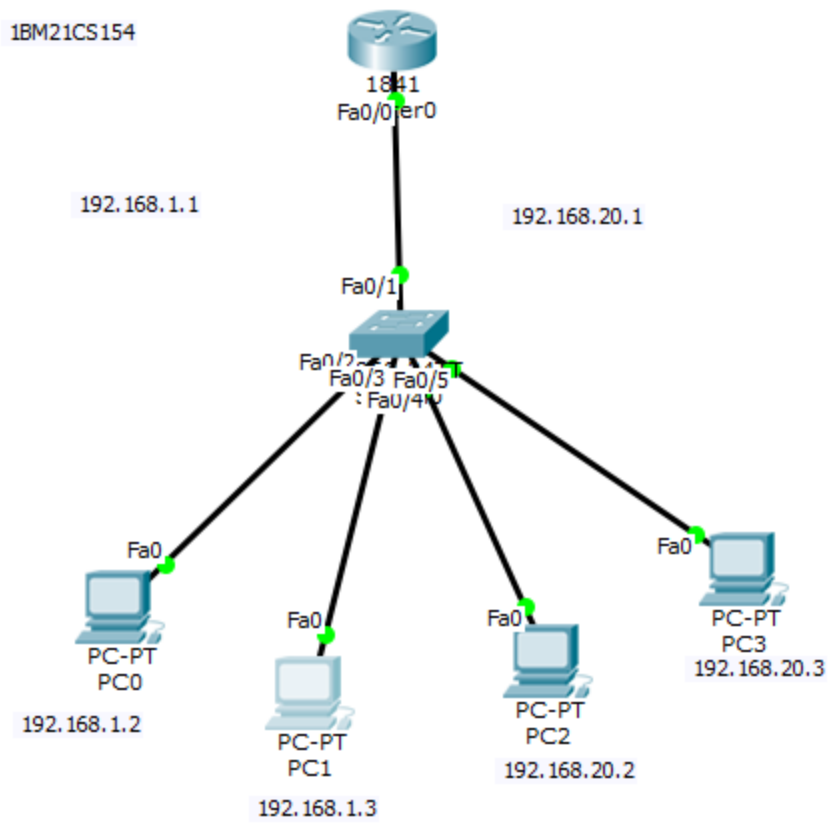
ping statistics for 192.168.20.2:

packets : sent = 4, Recieved = 4, Lost = 0 (0% loss)

Approximate round trip times in milli-seconds;

Minimum = 0ms, Maximum = 3ms Avg = 2ms





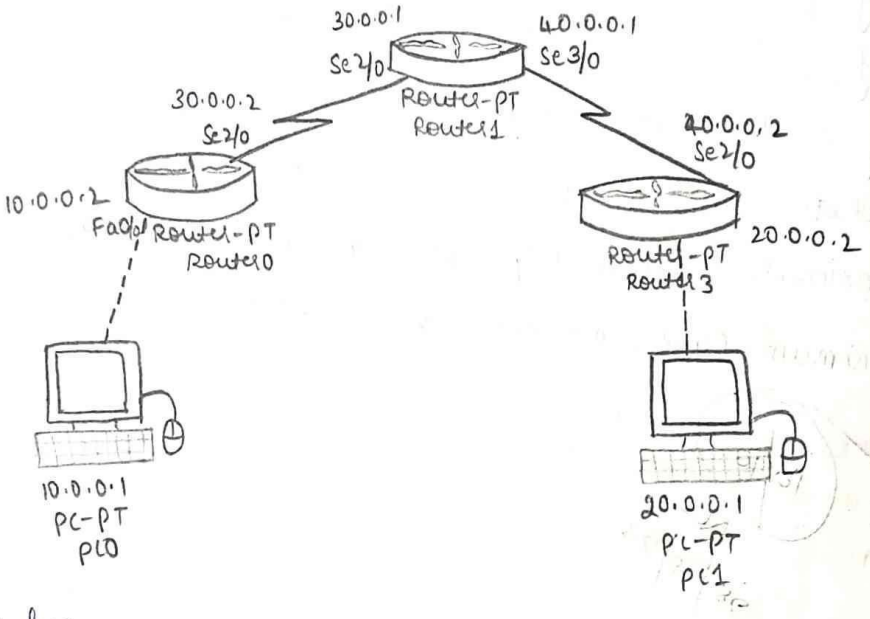
Command Prompt

```
Packet Tracer PC Command Line 1.0
PC>ping 192.168.20.3

Pinging 192.168.20.3 with 32 bytes of data:

Reply from 192.168.20.3: bytes=32 time=0ms TTL=127
Reply from 192.168.20.3: bytes=32 time=0ms TTL=127
Reply from 192.168.20.3: bytes=32 time=0ms TTL=127
Reply from 192.168.20.3: bytes=32 time=1ms TTL=127

Ping statistics for 192.168.20.3:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 1ms, Average = 0ms

PC>
```

Experiment No-11

Aim: To demonstrate the TTL/Life of a packet

Topology:



Procedure:

Step1: Create a topology with 2 pcs and 3 routers as shown above.

Step2: Configure the Ip addresses as 10.0.0.1 and 20.0.0.1 for pc0 and pc1 respectively.

Step 3: Configure the Ip addresses for routers and static default route

Router0;

Router # config t
Router (config) # interface fastethernet 0/0
Router (config-if) # ip address 10.0.0.2 255.0.0.0
Router (config-if) # ~~exit~~ no shut
Router (config-if) # exit
Router (config) # interface serial 2/0
Router (config-if) # ip address 30.0.0.2 255.0.0.0
Router (config-if) # no shut
Router (config-if) # exit

Router(config) # ip route 0.0.0.0 0.0.0.0 30.0.0.1
Router(config) # exit

Similarly, configure for router1 and router2

Step4: In simulation mode, send a simple PDU from one pc to another.

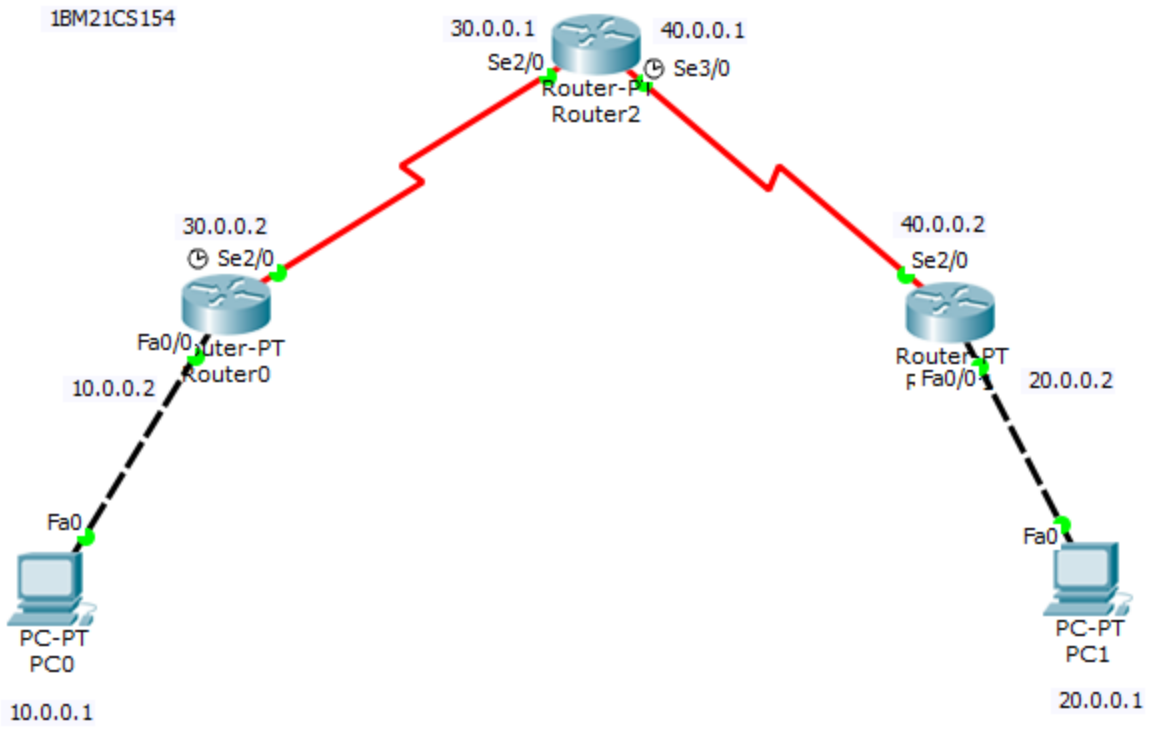
Step5: click on PDU during every transfer to see the Inbound & outbound PDU details, use capture button to capture every transfer.

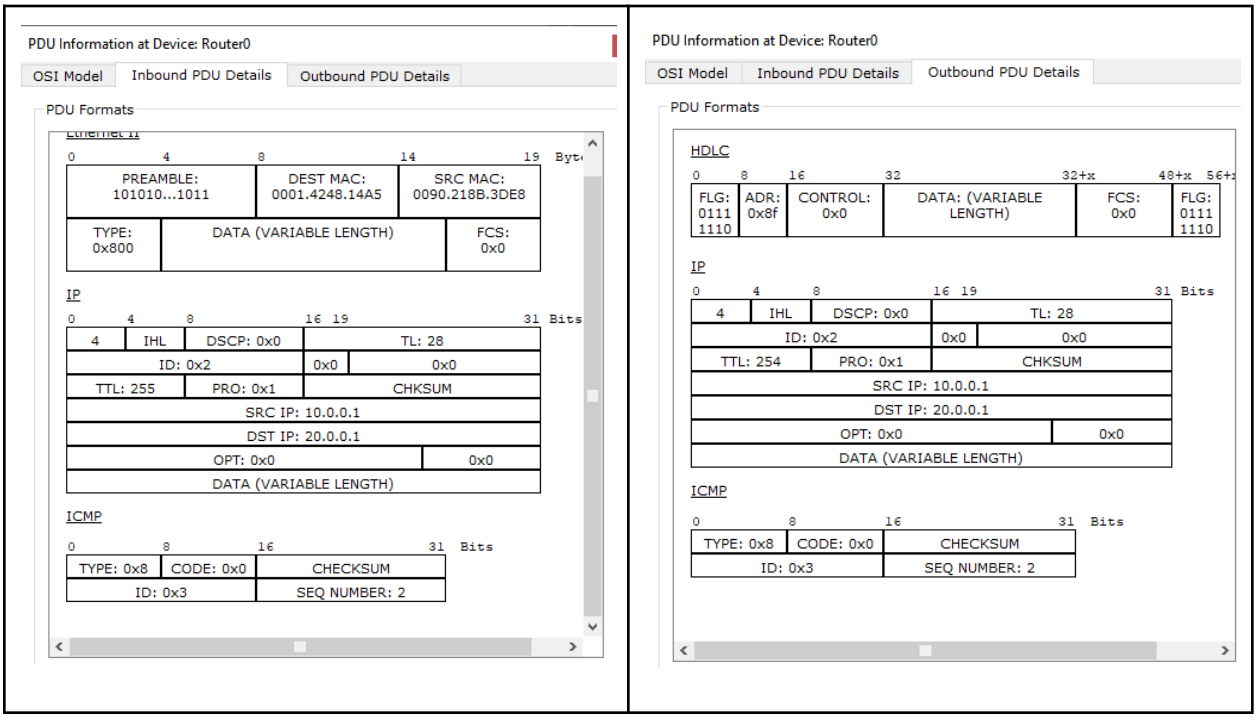
Observation ?

9/10

N

28/8/23

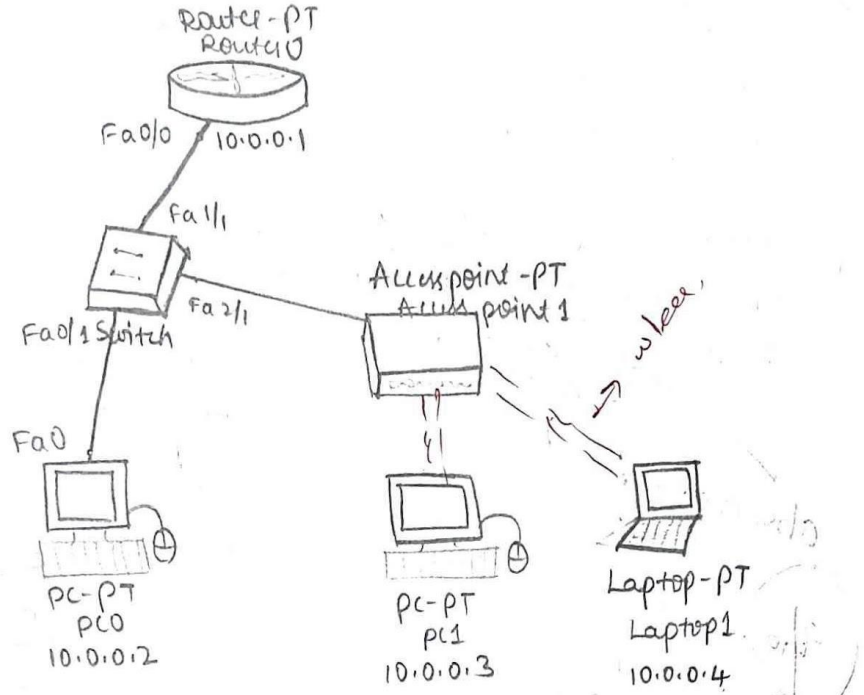

1BM21CS154
30.0.0.1
Se2/0
40.0.0.1
Se3/0
Router-PT
Router2
30.0.0.2
Se2/0
40.0.0.2
Se2/0
Fa0/0
Router-PT
Router0
10.0.0.2
Router-PT
Fa0/0
20.0.0.2
Fa0
Fa0
PC-PT
PC0
10.0.0.1
PC-PT
PC1
20.0.0.1


PDU Information at Device: Router0
OSI Model
Inbound PDU Details
Outbound PDU Details
PDU Formats
PREAMBLE: 101010...1011
DEST MAC: 0001.4248.14A5
SRC MAC: 0090.218B.3DE8
TYPE: 0x800
DATA (VARIABLE LENGTH)
FCS: 0x0
IP
4
IHL
DSCP: 0x0
TL: 28
ID: 0x2
TTL: 255
PRO: 0x1
CHKSUM
SRC IP: 10.0.0.1
DST IP: 20.0.0.1
OPT: 0x0
DATA (VARIABLE LENGTH)
ICMP
TYPE: 0x8
CODE: 0x0
CHECKSUM
ID: 0x3
SEQ NUMBER: 2
PDU Information at Device: Router0
OSI Model
Inbound PDU Details
Outbound PDU Details
PDU Formats
HDLC
FLG: 0111 1110
ADR: 0x8f
CONTROL: 0x0
DATA: (VARIABLE LENGTH)
FCS: 0x0
FLG: 0111 1110
IP
4
IHL
DSCP: 0x0
TL: 28
ID: 0x2
TTL: 254
PRO: 0x1
CHKSUM
SRC IP: 10.0.0.1
DST IP: 20.0.0.1
OPT: 0x0
DATA (VARIABLE LENGTH)
ICMP
TYPE: 0x8
CODE: 0x0
CHECKSUM
ID: 0x3
SEQ NUMBER: 2

Experiment No-12

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

Topology:



procedure:

Step1: Create the topology as shown above with PCs, switch, router, Access point and laptop.

Step2: Configure PC0 and Router as normally done.

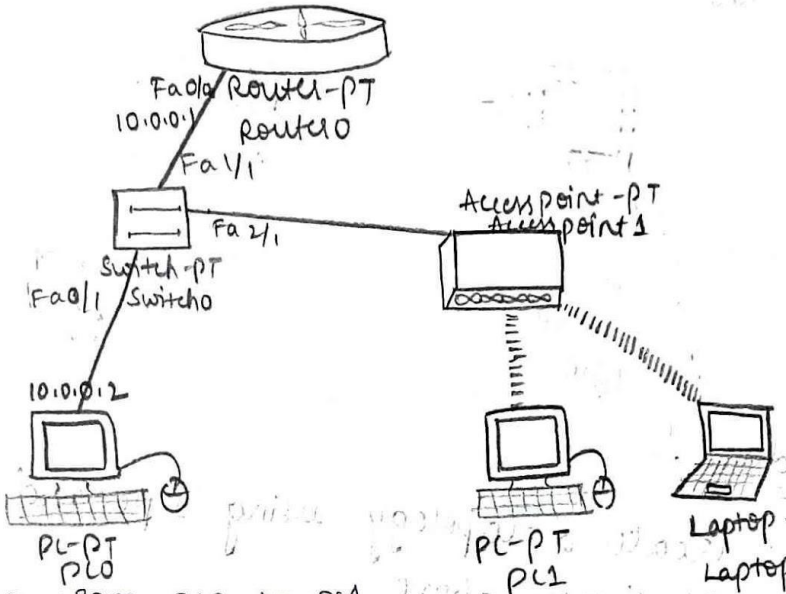
Step3: Configure the Accesspoint1, go to port1 and give SSIDname - (any name)

Step4: Select WEP and give any 10 digit Hex key. (1234567890 here). Configure PC1 and laptop with wireless standards.

Step5: 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.

Step 6: In the config tab a new wireless interface would have been added. Now configure, SSID, WEP, WEP key, IP address and Gateway (as normally done) to the device.

Final topology on screen:



Now, ping from PC0 to PC1

In PC0 command prompt,

PC> ping 10.0.0.3

Pinging 10.0.0.3 with 32 bytes of data:

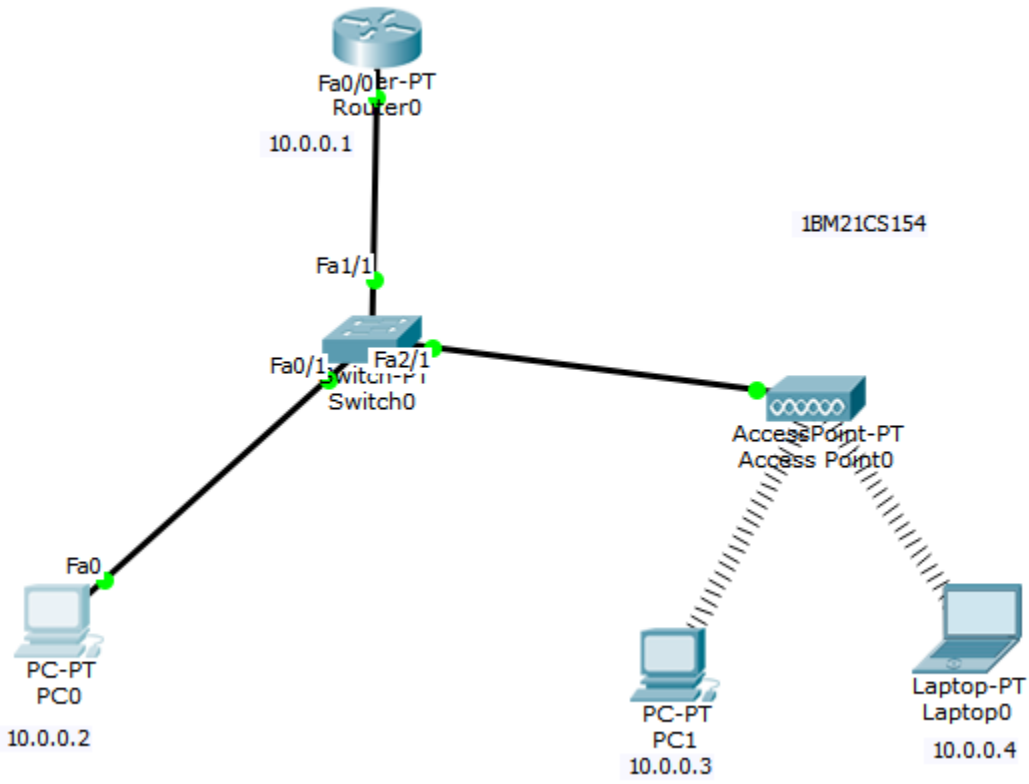
Reply from 10.0.0.3: bytes=32 time=47ms TTL=128
Reply from 10.0.0.3: bytes=32 time=32ms TTL=128
Reply from 10.0.0.3: bytes=32 time=35ms TTL=128
Reply from 10.0.0.3: bytes=32 time=3ms TTL=128

ping statistics for 10.0.0.3:

packets: Sent = 4, Recieved = 4, Lost = 0 (0% loss)

Approximate round-trip time in milli-seconds:

minimum = 3ms, Maximum = 47ms, Average = 29 ms




```
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=34ms TTL=128
Reply from 10.0.0.4: bytes=32 time=22ms TTL=128
Reply from 10.0.0.4: bytes=32 time=12ms TTL=128
Reply from 10.0.0.4: bytes=32 time=9ms TTL=128

Ping statistics for 10.0.0.4:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 9ms, Maximum = 34ms, Average = 19ms

PC>
```

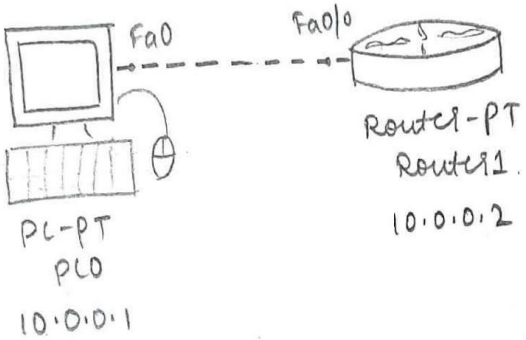
Experiment No-13

18/8/23

Lab-09

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

Topology :



procedure :

Step 1: Create a topology using 1 pc and 1 router as shown above.

Step 2: Set the ip address and gateway as 10.0.0.1 and 10.0.0.2 for the PC

Step 3: In the router, go to CLI

```
Router > enable
Router # config t
Router (config) # hostname r1
r1 (config) # enable secret p1
r1 (config) # interface fastethernet 0/0
r1 (config-if) # ip address 10.0.0.2 255.0.0.0
r1 (config-if) # no shut.
r1 (config-if) # line vty 0 5
r1 (config-line) # login
```

% Login disabled on line 132, until 'password' is set

r1(config-line)# password p0

r1(config-line)# exit

r1(config)# exit

r1# wr

Step 4: In command prompt of PC,

PC> ping 10.0.0.2

pinging 10.0.0.2 with 32 bytes of data:

Reply from 10.0.0.2 : bytes = 32 time = 0ms TTL = 255

Reply from 10.0.0.2 : bytes = 32 time = 0ms TTL = 255

Reply from 10.0.0.2 : bytes = 32 time = 0ms TTL = 255

Reply from 10.0.0.2 : bytes = 32 time = 0ms TTL = 255

ping statistics for 10.0.0.2:

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

Approximate round trip times in milli-seconds:

minimum = 0ms, maximum = 0ms, Average = 0ms

PC> telnet 10.0.0.2

Trying 10.0.0.2 .... open

User Access verification

password: (p0)

r1> enable

password: (p1)

r1# show ip route

codes: C - connected

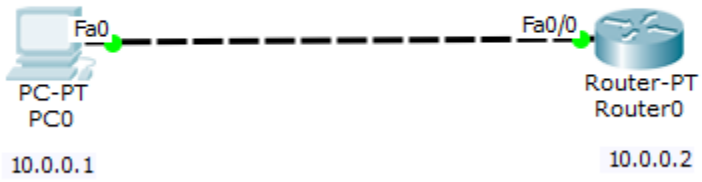
C 10.0.0.0/8 is directly connected, FastEthernet0/

Obs: Using telnet protocol, we can access the router from the PC (which is connected to it)

10/10

28/8/22

1BM21CS154



Command Prompt

```
Packet Tracer PC Command Line 1.0
PC>ping 10.0.0.2

Pinging 10.0.0.2 with 32 bytes of data:

Reply from 10.0.0.2: bytes=32 time=55ms TTL=255
Reply from 10.0.0.2: bytes=32 time=0ms TTL=255
Reply from 10.0.0.2: bytes=32 time=0ms TTL=255
Reply from 10.0.0.2: bytes=32 time=0ms TTL=255

Ping statistics for 10.0.0.2:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 55ms, Average = 13ms

PC>telnet 10.0.0.2
Trying 10.0.0.2 ...Open


User Access Verification

Password:
r1>enable
Password:
r1#show ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, ia - IS-IS inter area
       * - candidate default, U - per-user static route, o - ODR
       P - periodic downloaded static route

Gateway of last resort is not set

C    10.0.0.0/8 is directly connected, FastEthernet0/0
r1#
```

CYCLE 2

Experiment No-14

CYCLE-2

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

program:

```
#include<stdio.h>
char m[50], g[50], r[50], q[50], temp[50];
void caltrans(int);
void crc(int);
void calram();
void shift1();
void main()
{
  int n, i=0;
  char ch, flag=0;
  printf("Enter the frame bits:");
  while((ch=getc(stdin))!='\n')
  m[i++]=ch;
  n=i;
  for(i=0; i<16; i++)
  m[n++]='0';
  m[n]='\0';
  printf("message after appending 16 zeros: %s", m);
  for(i=0; i<=16; i++)
  g[i]='0';
  g[0]=g[4]=g[11]=g[16]='1'; g[17]='\0';
  printf("\n generator: %s\n", g);
  crc(n);
  printf("\n quotient: %s", q);
  caltrans(n);
  printf("\n transmitted frame: %s", m);
  printf("\n Enter transmitted frame:");
```

```
scanf("%s", m);
printf("CRC checking \n");
crc(n);
printf("\n\n last remainder: %s", r);
for (i=0; i<16; i++)
 if (r[i] != '0')
 flag = '1';
 else
  continue;
 if (flag == '1')
  printf("error during transmission");
 else
  printf("\n Recieved frame is correct");
}

void crc (int n)
{
 int i,j;
 for (i=0; i<n; i++)
  temp[i] = m[i];
 for (i=0; i<16; i++)
  r[i] = m[i];
  for (i=0; i<n-16; i++)
  {
   if(r[0] == '1')
   {
    q[i] = '1';
    caleam();
   }
  else
   q[i] = '0';
   shiftl();
  }
  r[16] = m[17+i];
  r[17] = '\0';
```

```
    for (j=0; j<=17 ;j++)
      temp[j]= r[j];
  }
  q[n-16]= '\0';
}
void calram()
{
  int i,j;
  for (i=1; j<=16; i++)
  r[i-1] = ((int)temp[i]-48) ^ ((int)q[i]-48)+48;
}

void shiftl()
{
  int i;
  for (i=1; i<=16;i++)
  r[i-1]=r[i];
}
void caltrans(int n)
{
  int i,k=0;
  for (i=n-16; i<n;i++)
  m[i] = ((int)m[i]-48)^((int) r[k++]-48)+48;
   m[i]='\0';
}
```

output:

Enter frame bits: 1011

message after appending 16 zeros: 1011 0000 0000 0000 0000

generator: 10001000000100001

quotient : 1011

Transmitted frame: 1011 1011 0001 0110 1011

Enter Transmitted frame: 1011 1011 0001 0110 1011

last remainder: 0000 0000 0000 0000

Recieved frame is correct.

OUTPUT:

```
Enter the frame bits:1011
Message after appending 16 zeros:1011000000000000000
generator:10001000000100001


quotient:1011
transmitted frame:10111011000101101011
Enter transmitted frame:10111011000101101011
CRC checking


last remainder:0000000000000000

Received frame is correct
Process returned 0 (0x0)   execution time : 14.468 s
Press any key to continue.
```

```
Enter the frame bits:1001
Message after appending 16 zeros:1001000000000000000
generator:10001000000100001


quotient:1001
transmitted frame:10011001000100101001
Enter transmitted frame:10011001000000101001
CRC checking


last remainder:0000000100000000Error during transmission
Process returned 0 (0x0)   execution time : 19.597 s
Press any key to continue.
```

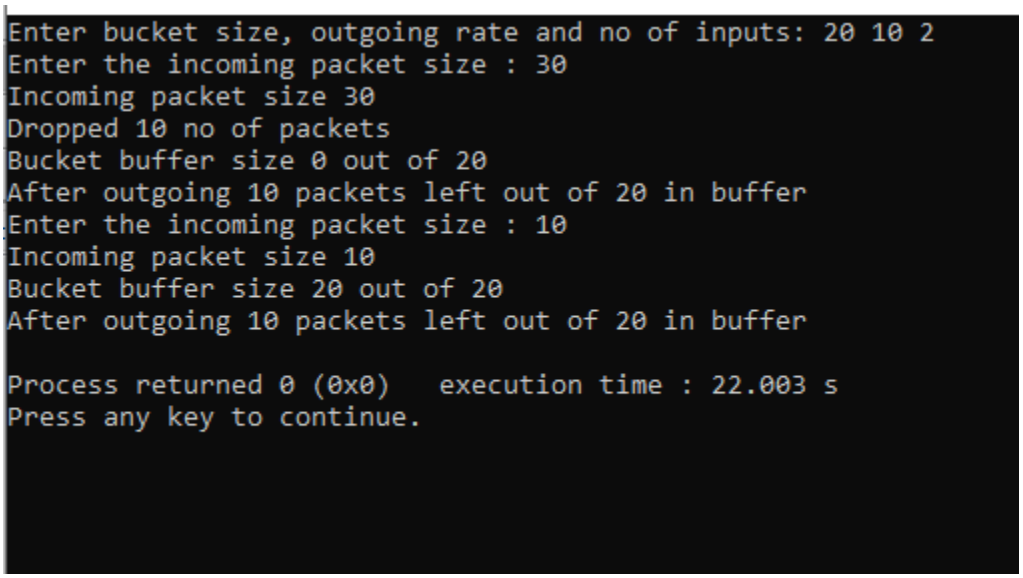
Experiment No-15

2) Write a program for congestion control using leaky bucket algorithm.

program :

```
#include <stdio.h>
int main(){
int incoming, outgoing, bucket_size, n, store=0;
printf("Enter bucket size, outgoing rate and no.
  of inputs: ");
scanf("%d %d %d", &bucket_size, &outgoing, &n);
while(n!=0){
printf("Enter the incoming packet size:");
scanf("%d", &incoming);
printf("Incoming packet size %d\n", incoming);
if (incoming <= (bucket_size - store)){
   store += incoming;
printf("Bucket buffer size %d out of %d\n",
  store, bucket_size);
}
else{
printf("Dropped %d no. of packets\n", incoming-
  (bucket_size - store));
printf("Bucket buffer size %d out of %d\n",
        store, bucket_size);
store = bucket_size;
}
store = store - outgoing;
printf("After outgoing %d packets left out %d
        in buffer\n", store, bucket_size);
  n--;
}return 0;
}
```

OUTPUT:


Enter bucket size, outgoing rate and no of inputs: 20 10 2
Enter the incoming packet size : 30
Incoming packet size 30
Dropped 10 no of packets
Bucket buffer size 0 out of 20
After outgoing 10 packets left out of 20 in buffer
Enter the incoming packet size : 10
Incoming packet size 10
Bucket buffer size 20 out of 20
After outgoing 10 packets left out of 20 in buffer
Process returned 0 (0x0) execution time : 22.003 s
Press any key to continue.

Experiment No-16

11/09/23 [Lab-10]

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.

Server:

- A server has a bind() method which binds it to a specific IP and port so that it can listen to incoming requests on that IP and port.
- A server has a listen() method which puts the server into listening mode. This allows the server to listen to incoming connections.
- And Server has an accept() and close() method. The accept method initiates a connection with the client and the close method closes the connection with the client.

Step 1: We import socket which is necessary for client-server program.
Then we made a socket objects and reserved a port on our pc.

Step 2: Bind the server to specified port. passing an empty string means that server can listen to incoming connections from other computers as well. Here we pass 127.0.0.1 so, it will listen to only those calls made within the local computer.

Step 3: We put the server into listening mode. At last, we make while loop and start to accept all incoming connection and close connections after thank you message.

=> open a new file in a IDLE and write the following code and save as "server.py"

```
from socket import *
serverName = "127.0.0.1"
serverPort = 12000
serverSocket = socket(AF_INET, SOCK_STREAM)
serverSocket.bind((serverName, serverPort))
serverSocket.listen(1)
while 1:
    print("The server is ready to receive");
    connectionSocket, addr = serverSocket.accept()
    sentence = connectionSocket.recv(1024).decode()
    file = open(sentence, "r")
    l = file.read(1024)
    connectionSocket.send(l.encode())
    print("\n Sent contents of ' + sentence)
    file.close()
    connectionSocket.close()
```

Run the file server.py

o/p => The server is ready to recieve.

This shows that our server is working.

Client:

step1: make a Socket object.

step2: Establish a connection with the server and lastly we will recieve data from the server and close the connection.

→ open a new file in a IDLE and write the following code and save as "client.py"

```
from socket import *
serverName = '127.0.0.1'
serverPort = 12000
clientSocket = socket(AF_INET, SOCK_STREAM)
clientSocket.connect((serverName, serverPort))
sentence = input("\n Enter file name: ")
clientSocket.send(sentence.encode())
fileContents = clientSocket.recv(1024).decode()
print("\n From Server:\n")
print(fileContents)
clientSocket.close()
```

Run the file client.py

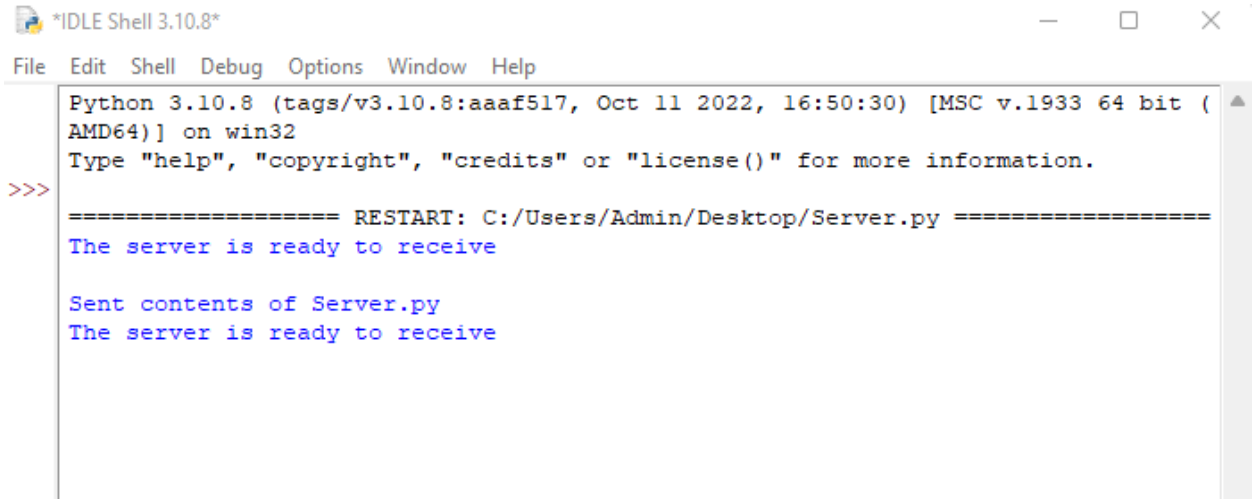
```
O/P => Enter file name : server.py
```

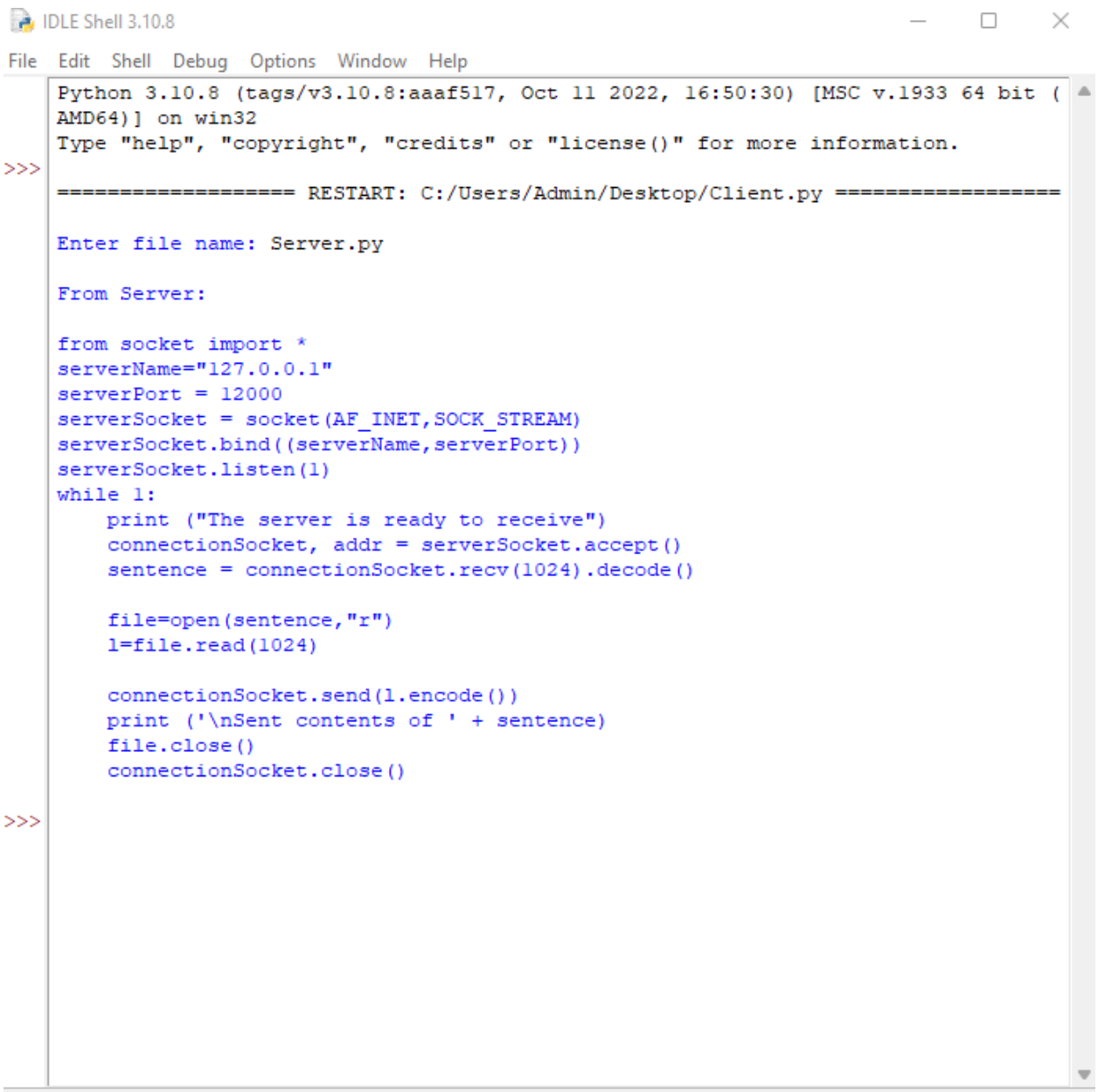
when the client requests for server.py the server will send a file which is requested and the filecontents will be stored in 'filecontents' file and in the output it will be printed.

server:

```
The server is ready to recieve
sent contents of server.py
The server is ready to recieve.
```

OUTPUT:


IDLE Shell 3.10.8
File Edit Shell Debug Options Window Help
Python 3.10.8 (tags/v3.10.8:aaaf517, Oct 11 2022, 16:50:30) [MSC v.1933 64 bit (
AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
========== RESTART: C:/Users/Admin/Desktop/Server.py ==========
The server is ready to receive
Sent contents of Server.py
The server is ready to receive


IDLE Shell 3.10.8
File Edit Shell Debug Options Window Help
Python 3.10.8 (tags/v3.10.8:aaaf517, Oct 11 2022, 16:50:30) [MSC v.1933 64 bit (
AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
========== RESTART: C:/Users/Admin/Desktop/Client.py ==========
Enter file name: Server.py
From Server:
from socket import *
serverName="127.0.0.1"
serverPort = 12000
serverSocket = socket(AF_INET,SOCK_STREAM)
serverSocket.bind((serverName,serverPort))
serverSocket.listen(1)
while 1:
print ("The server is ready to receive")
connectionSocket, addr = serverSocket.accept()
sentence = connectionSocket.recv(1024).decode()
file=open(sentence,"r")
l=file.read(1024)
connectionSocket.send(l.encode())
print ('\nSent contents of ' + sentence)
file.close()
connectionSocket.close()
>>>

Experiment No-17

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.

=> server:

Here, Like in TCP/IP we create socket object and bind it to specified port and server will be continuously listening when the client sends request it responds accordingly

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 recieve")
while 1:
    sentence, clientAddress = serverSocket.recvfrom(2048)
    sentence = sentence.decode ("utf-8")
    file = open (sentence, "r")
    con = file.read (2048)
    serverSocket.sendto (bytes (con, "utf-8"), clientAddress)
    print ('\n sent contents of ', end = ' ')
    print (sentence)
    file.close()
```

when you run the above file, the output window O/p => The server is ready to recieve.

clientUDP.py

```
from socket import *
serverName = "127.0.0.1"
serverPort = 12000
clientSocket = socket(AF_INET, SOCK_DGRAM)
sentence = input("\n Enter file name: ")
clientSocket.sendto(bytes(sentence, "utf-8"), (serverN
    -ame, serverPort))
fileContents, serverAddress = clientSocket.recvfrom
                                        (2048)
print("\nReply from server:\n")
print(fileContents.decode("utf-8"))
clientSocket.close()
```

o/p =>

```
enter file name: serverUDP.py
Reply from server:
```

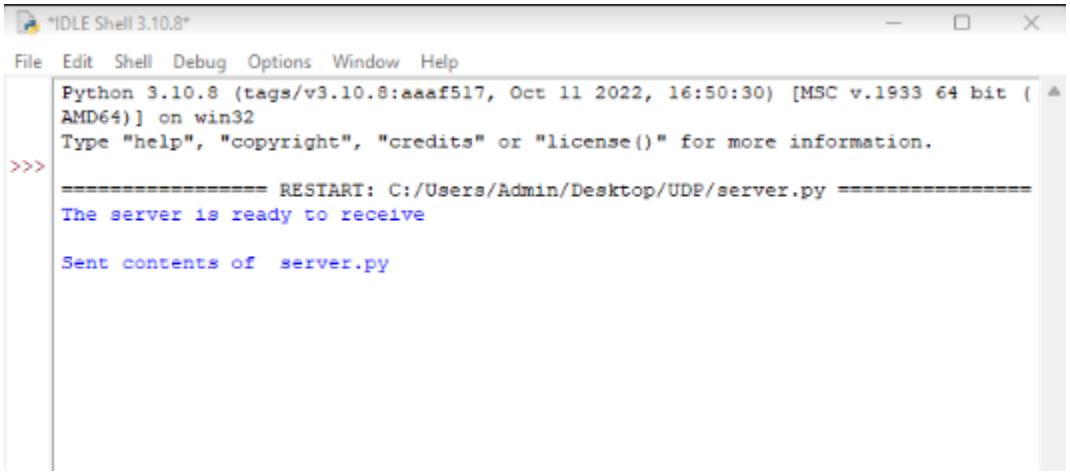
server:

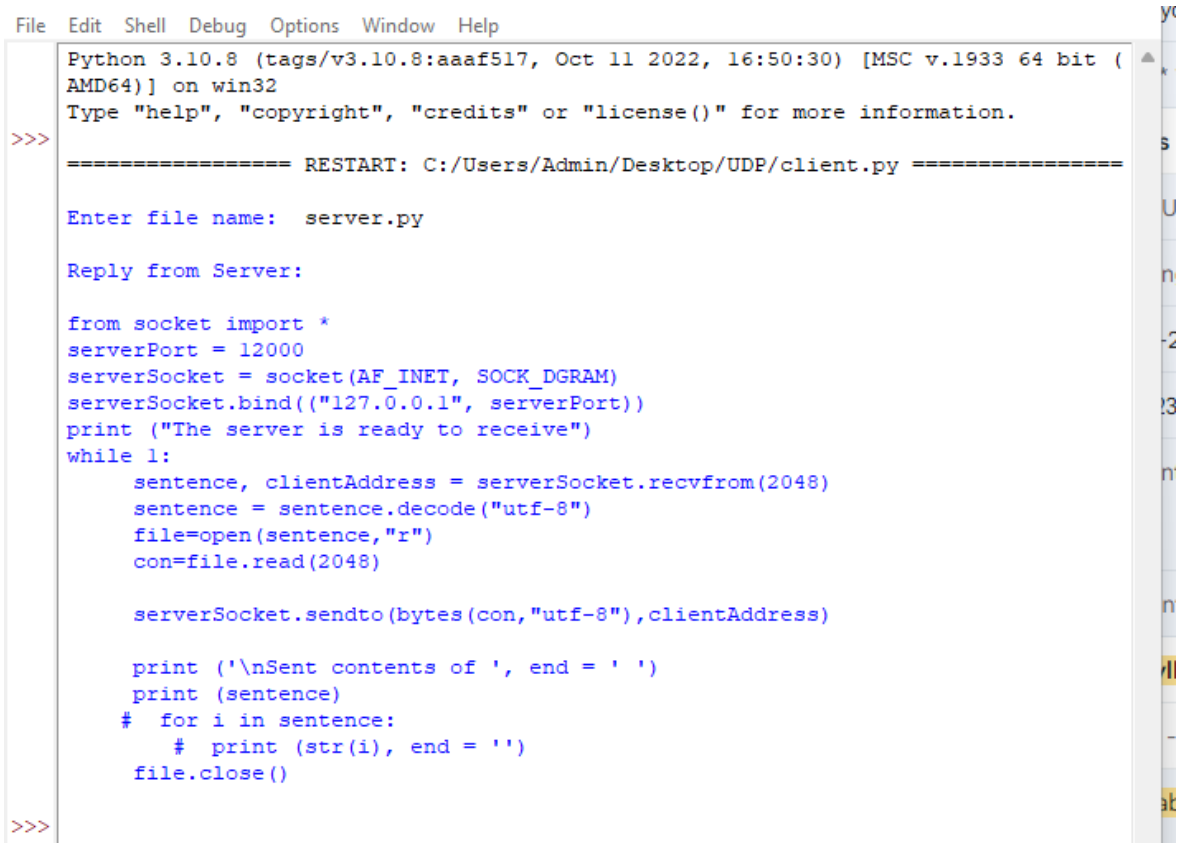
```
The server is ready to recieve
Sent contents of serverUDP.py
```

10/10

1/9/23

OUTPUT:


IDLE Shell 3.10.8
File Edit Shell Debug Options Window Help
Python 3.10.8 (tags/v3.10.8:aaaf517, Oct 11 2022, 16:50:30) [MSC v.1933 64 bit (
AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
========== RESTART: C:/Users/Admin/Desktop/UDP/server.py ==========
The server is ready to receive
Sent contents of server.py


File Edit Shell Debug Options Window Help
Python 3.10.8 (tags/v3.10.8:aaaf517, Oct 11 2022, 16:50:30) [MSC v.1933 64 bit (
AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
========== RESTART: C:/Users/Admin/Desktop/UDP/client.py ==========
Enter file name: server.py
Reply from Server:
from socket import *
serverPort = 12000
serverSocket = socket(AF_INET, SOCK_DGRAM)
serverSocket.bind(("127.0.0.1", serverPort))
print ("The server is ready to receive")
while 1:
sentence, clientAddress = serverSocket.recvfrom(2048)
sentence = sentence.decode("utf-8")
file=open(sentence,"r")
con=file.read(2048)
serverSocket.sendto(bytes(con,"utf-8"),clientAddress)
print ('\nSent contents of ', end = ' ')
print (sentence)
# for i in sentence:
# print (str(i), end = '')
file.close()
>>>