VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"JnanaSangama", Belgaum -590014, Karnataka.

LAB REPORT

on

COMPUTER NETWORKS

Submitted by

Keerthi Reddy (1BM22CS094)

in partial fulfillment for the award of the degree of

BACHELOR OF ENGINEERING

in

COMPUTER SCIENCE AND ENGINEERING

B.M.S. COLLEGE OF ENGINEERING

(Autonomous Institution under VTU)

BENGALURU-560019

Sep 2024-Jan 2025

B. M. S. College of Engineering,

Bull Temple Road, Bangalore 560019

(Affiliated To Visvesvaraya Technological University, Belgaum)

Department of Computer Science and Engineering

CERTIFICATE

This is to certify that the Lab work entitled "**COMPUTER NETWORKS**" carried out by **Keerthi Reddy** **(1BM22CS094)** who is bonafide student of **B. M. S. College of Engineering.** It is in partial fulfillment for the award of **Bachelor of Engineering in Computer Science and Engineering** of the Visvesvaraya Technological University, Belgaum during the year 2024-25. The Lab report has been approved as it satisfies the academic requirements in respect of **Computer Networks Lab - (23CS5PCCON)** work prescribed for the said degree.

Dr. Nandhini Vineeth

Associate Professor,
Department of CSE,
BMSCE, Bengaluru

Dr. Latha NR

Professor and Head,
Department of CSE
BMSCE, Bengaluru

INDEX

CYCLE 1

12	18-12-2024	To construct a WLAN and make the nodes communicate wirelessly	33-35

INDEX

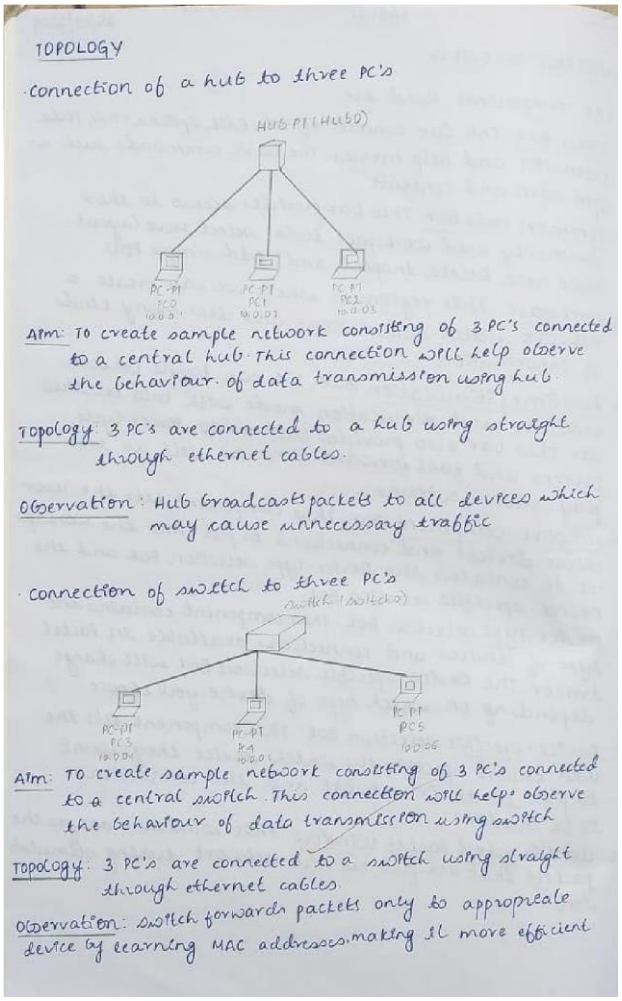
CYCLE 2

CYCLE-1

Question 1:

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

Observation:



TOPOLOGY

connection of a hub to three PC's

Aim: To create sample network consisting of 3 PC's connected to a central hub. This connection will help observe the behaviour of data transmission using hub.

Topology: 3 PC's are connected to a hub using straight through ethernet cables.

Observation: Hub broadcasts packets to all devices which may cause unnecessary traffic

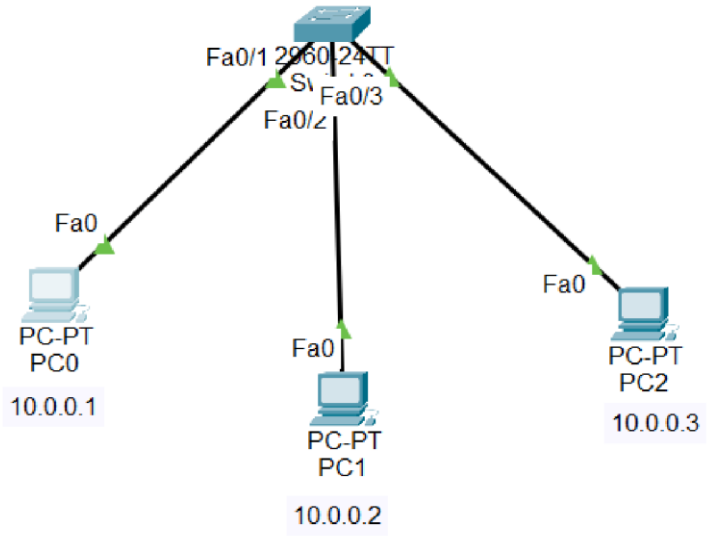
connection of switch to three PC's

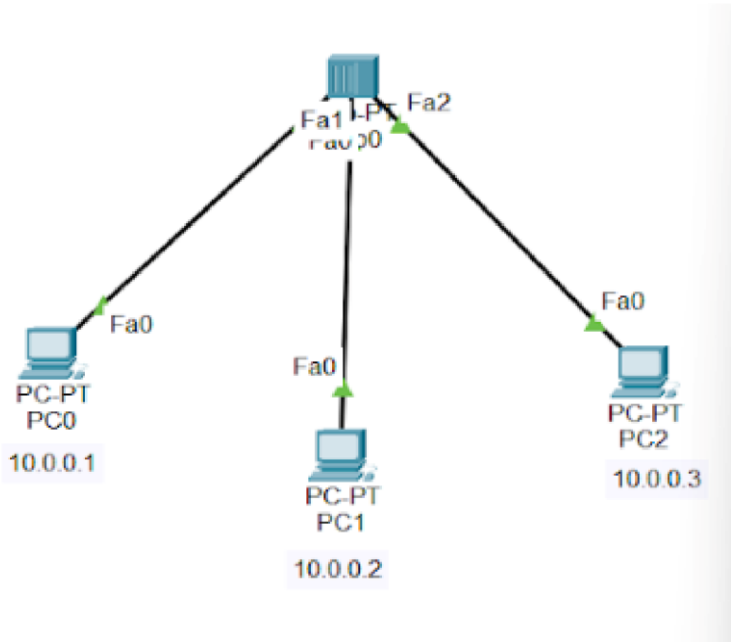
Aim: To create sample network consisting of 3 PC's connected to a central switch. This connection will help observe the behaviour of data transmission using switch

Topology: 3 PC's are connected to a switch using straight through ethernet cables.

Observation: Switch forwards packets only to appropreate device by learning MAC addresses, making it more efficient

Screenshot of the topology:





Screenshot of the output:

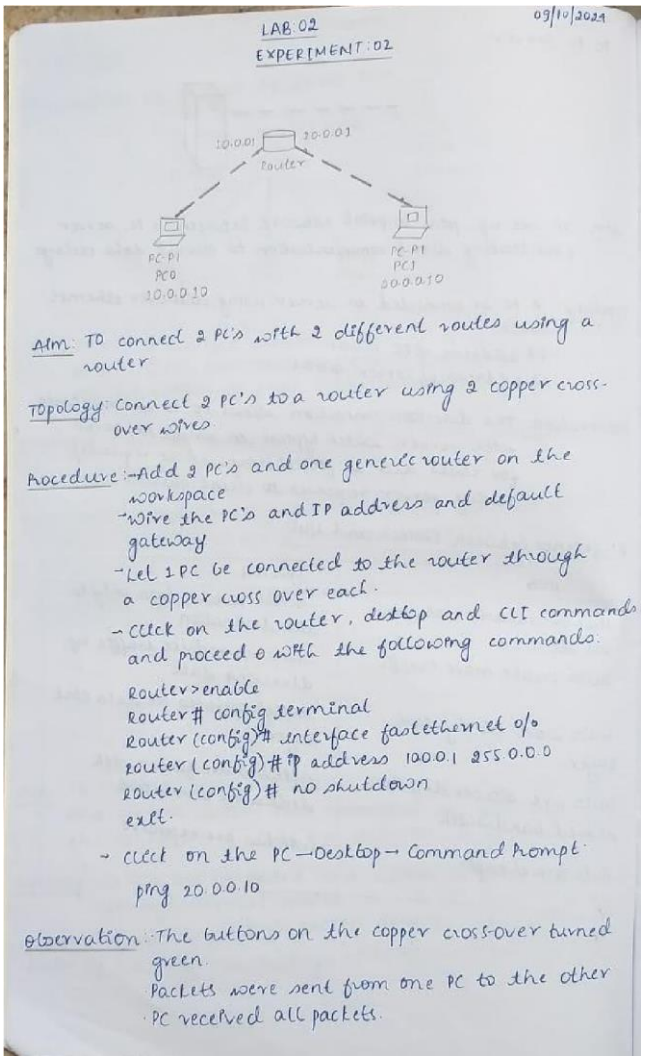
```
Cisco Packet Tracer PC Command Line 1.0
C:\>ping 10.0.0.1

Pinging 10.0.0.1 with 32 bytes of data:

Reply from 10.0.0.1: bytes=32 time=8ms TTL=128
Reply from 10.0.0.1: bytes=32 time=4ms TTL=128
Reply from 10.0.0.1: bytes=32 time=4ms TTL=128
Reply from 10.0.0.1: bytes=32 time=4ms 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 = 4ms, Maximum = 8ms, Average = 5ms

C:\>
```

```
Cisco Packet Tracer PC Command Line 1.0
C:\>ping 10.0.0.3

Pinging 10.0.0.3 with 32 bytes of data:

Reply from 10.0.0.3: bytes=32 time<1ms TTL=128
Reply from 10.0.0.3: bytes=32 time<1ms TTL=128
Reply from 10.0.0.3: bytes=32 time<1ms TTL=128
Reply from 10.0.0.3: bytes=32 time<1ms TTL=128

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

C:\>
```

Question 2:

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

Observation:



LAB 02 09/10/2024

EXPERIMENT: 02

10.0.0.1 20.0.0.1 Router

PC-PT PC0 10.0.0.10

PC-PT PC1 20.0.0.10

Aim: TO connect 2 PC's with 2 different routes using a router

Topology: Connect 2 PC's to a router using 2 copper cross-over wires

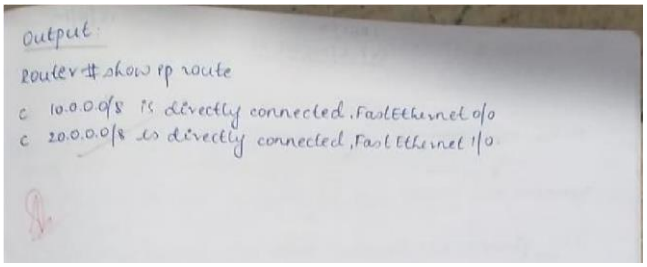
Procedure:
- Add 2 PC's and one generic router on the workspace
- Wire the PC's and IP address and default gateway
- Let 1 PC be connected to the router through a copper cross over each.
- Click on the router, desktop and CLI commands and proceed with the following commands:

```
Router>enable
Router# config terminal
Router(config)# interface fastethernet 0/0
Router(config)# ip address 100.0.1 255.0.0.0
Router(config)# no shutdown
exit
```

- Click on the PC → Desktop → Command Prompt

ping 20.0.0.10

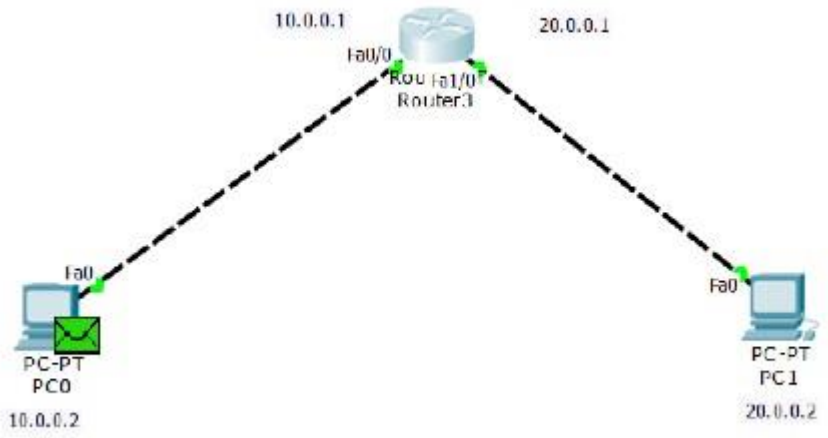
Observation: The buttons on the copper cross-over turned green.
Packets were sent from one PC to the other
PC received all packets.



Output:

```
Router# show ip route
C  10.0.0.0/8 is directly connected, FastEthernet 0/0
C  20.0.0.0/8 is directly connected, FastEthernet 1/0
```

Screenshot of the topology:



Screenshot of the output:

```
C    10.0.0.0/8 is directly connected, FastEthernet0/0
C    20.0.0.0/8 is directly connected, FastEthernet1/0
```

```
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=4ms TTL=127
Reply from 10.0.0.2: bytes=32 time=4ms TTL=127
Reply from 10.0.0.2: bytes=32 time=4ms TTL=127
Reply from 10.0.0.2: bytes=32 time=4ms TTL=127

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

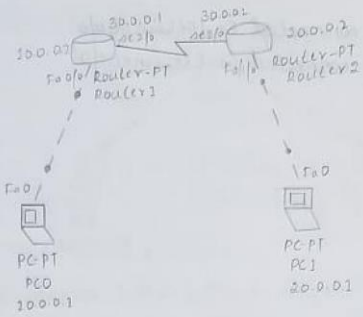
Question 3:

Configure default route, static route to the Router

Observation:

LAB: 03 16/10/24

EXPERIMENT 02



Aim: Configuration of 2 routers

Topology: Connect 2 routers and 1 PC to each of the 2 routers

Procedure:
1. Add 2 routers and 2 PC's, one connecting to 1 router and the other PC to the other routers using a copper cross-over cable
2. Connect both the routers using a serial DTE
3. Configure the IP addresses, subnet mask and Gateway for both the PC's
4. Configure the ip addresses and connection with PC by clicking on the router → CLI and manually type the following commands

```
Router>enable
Router# config terminal
Router(config)#interface fastethernet/serial
[fastethernet → PC, serial → router]
Router(config-if)# ip address 10.0.0.2 255.0.0.0
Router(config-if)# no shutdown
exit
```

5. This completes the connection between 2 routers and router to PC.

6.

Observation: The PC's are not communicating even when they are connected through the routers.
- On pinging PC0 to Fa0/0 port of the router the message is pinged
- On pinging PC0 to the other network, the message isn't reachable.

EXPERIMENT:03 (a)

Aim: TO connect the default, static route to the router

Procedure

1. By continuing the perious procedure, to connect the PC's so they can communicate with each other, go to router, CLI and continue with the following commands

ip route 20.0.0.0 255.0.0.0 30.0.0.2

(ip address of neighbour network (router))

2. This should be repeated for the other router, which completes the static routing.

Observation: The PC's are now communicated with each other

on pinging.

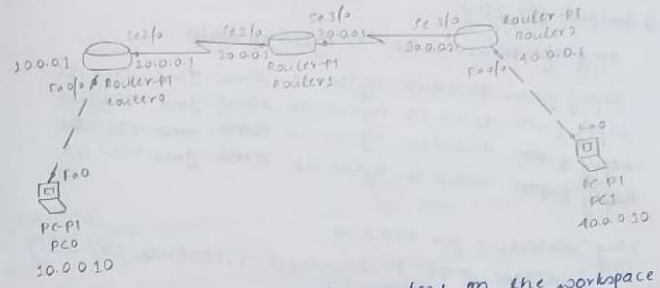
LAB: 04

EXPERIMENT:04 (b) 23/10/21

Question: Configure default route and static route to the Router

Aim: TO configure default routing

Topology:



Procedure: 1) Place 2 PC's and 3 routers on the workspace

2) setup the IP addresses and Gateway of the PC's

3) setup the IP addresses and connection of the routers in its corresponding CLI

4) Go to CLI of Router1 and follow the below commands

Router# config terminal

Router(config)# ip route 10.0.0.0 255.0.0.0 20.0.0.1

Router(config)# ip route 40.0.0.0 255.0.0.0 30.0.0.2

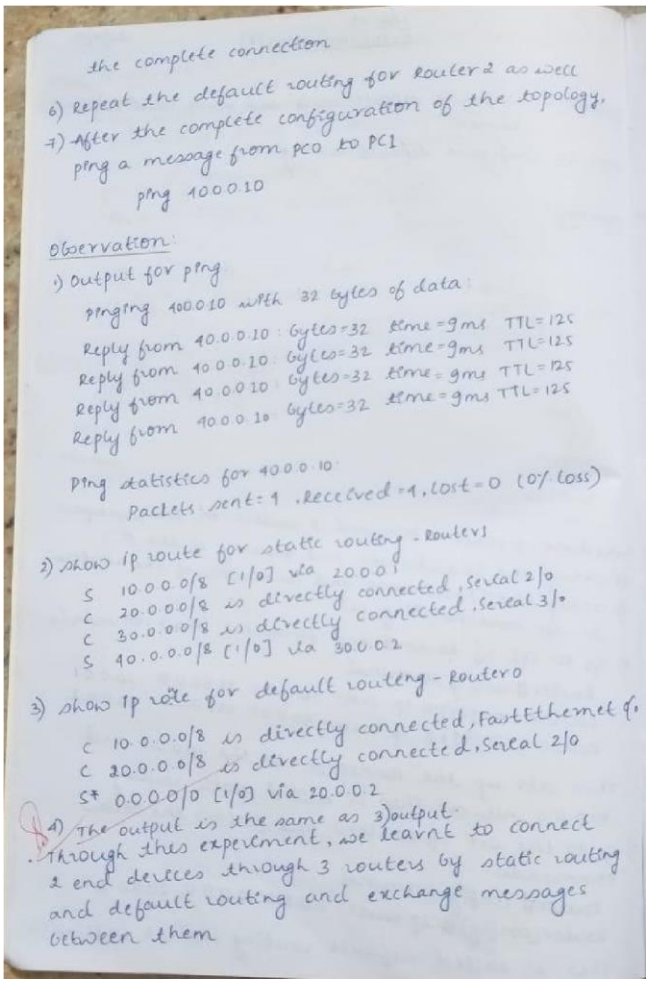
This sets up the connection to the 10.0.0.0 and 40.0.0.0 network. This is called static routing

5) Go to the CLI of Router0 and follow the below commands:

Router# config terminal

Router(config)# ip route 0.0.0.0 0.0.0.0 20.0.0.2

This is called default routing and it helps with



the complete connection

6) Repeat the default routing for Router2 as well

7) After the complete configuration of the topology, ping a message from pc0 to PC1

ping 100.0.10

Observation

1) Output for ping

pinging 100.0.10 with 32 bytes of data

Reply from 40.0.0.10 bytes=32 time=9ms TTL=125
Reply from 40.0.0.10 bytes=32 time=9ms TTL=125
Reply from 40.0.0.10 bytes=32 time=9ms TTL=125
Reply from 40.0.0.10 bytes=32 time=9ms TTL=125

ping statistics for 40.0.0.10

Packets sent=4, Received=4, Lost=0 (0% loss)

2) show ip route for static routing - Router1

S 10.0.0.0/8 [1/0] via 20.0.0.1
C 20.0.0.0/8 is directly connected, Serial 2/0
C 30.0.0.0/8 is directly connected, Serial 3/0
S 40.0.0.0/8 [1/0] via 30.0.0.2

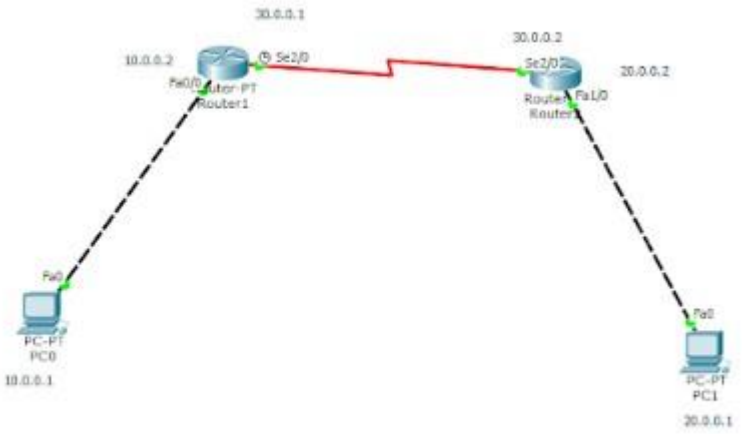
3) show ip route for default routing - Router0

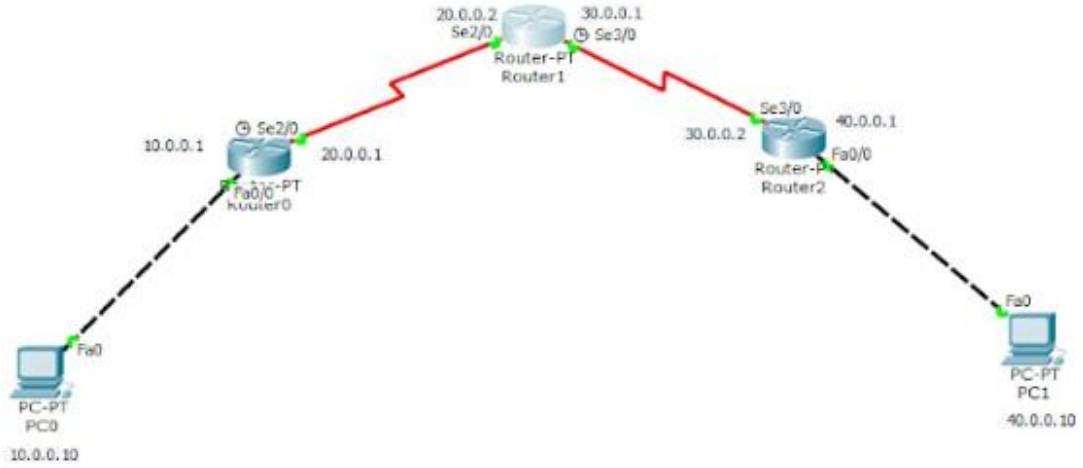
C 10.0.0.0/8 is directly connected, FastEthernet 0/
C 20.0.0.0/8 is directly connected, Serial 2/0
S* 0.0.0.0/0 [1/0] via 20.0.0.2

4) The output is the same as 3) output

Through this experiment, we learnt to connect 2 end devices through 3 routers by static routing and default routing and exchange messages between them

Screenshot of the topology:





Screenshot of the output:

```
S    10.0.0.0/8 [1/0] via 20.0.0.1
C    20.0.0.0/8 is directly connected, Serial2/0
C    30.0.0.0/8 is directly connected, Serial3/0
S    40.0.0.0/8 [1/0] via 30.0.0.2
```

```
C    10.0.0.0/8 is directly connected, FastEthernet0/0
C    20.0.0.0/8 is directly connected, Serial2/0
S*   0.0.0.0/0 [1/0] via 20.0.0.2
```

Question 4:

Configure DHCP within a LAN and outside LAN.

```
PC>ping 40.0.0.10

Pinging 40.0.0.10 with 32 bytes of data:

Reply from 40.0.0.10: bytes=32 time=9ms TTL=125
Reply from 40.0.0.10: bytes=32 time=9ms TTL=125
Reply from 40.0.0.10: bytes=32 time=5ms TTL=125
Reply from 40.0.0.10: bytes=32 time=11ms TTL=125

Ping statistics for 40.0.0.10:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
```

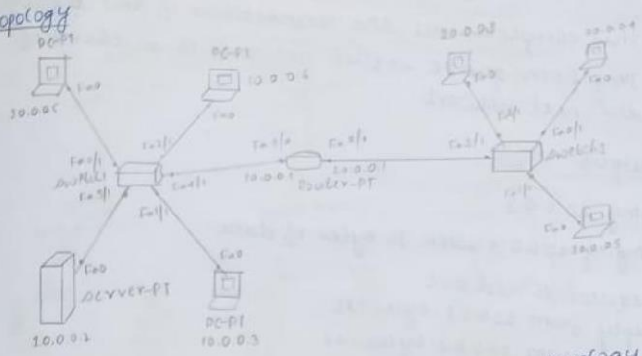
Observation:

LAB 05
EXPERIMENT 04
13/11/2024

Question: Configure DHCP within a LAN and outside LAN

Aim: TO configure DHCP

Topology



Procedure:

1) setup the devices as shown in the above topology

2) TO configure server:
Go to Desktop → IP configuration and give the IP address = 10.0.0.2 and default Gateway 10.0.0.1 10.0.0.1 is the IP address of the router for the first network
Services → DHCP → service → on
Change default gateway, start ip address for the 2 switches on both networks and add those.

3) configure the router:
Router → CLI and following commands
- enable
- config terminal
- interface fastethernet 1/0
- ip address 10.0.0.1 255.0.0.0
- ip helper-address 10.0.0.2

do write repeat
Follow the commands for both the sides of router.

4 configure the PC's:
PC → Desktop → DHCP
DHCP is successfully set.
Do as above for all the PC's in the topology

5 This completes all the connections of the topology

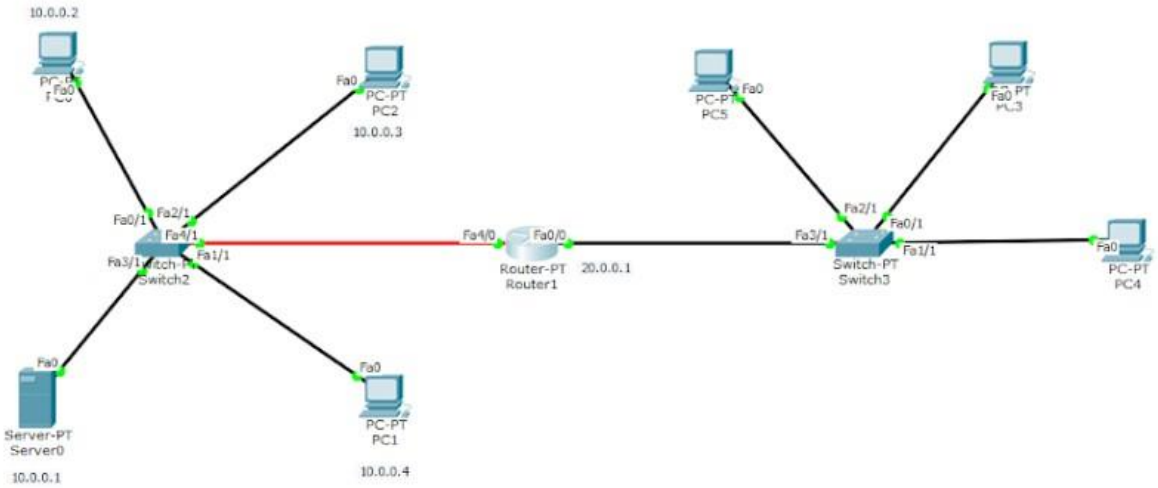
6 Ping from one PC of first network to another PC of the next network.

Output:

ping 20.0.0.3
Pinging 20.0.0.3 with 32 bytes of data:
Request timed out
Reply from 20.0.0.3: bytes=32
Reply from 20.0.0.3: bytes=32
Reply from 20.0.0.3: bytes=32
Packets: sent=4 Received=3 Lost=1 (25% loss)

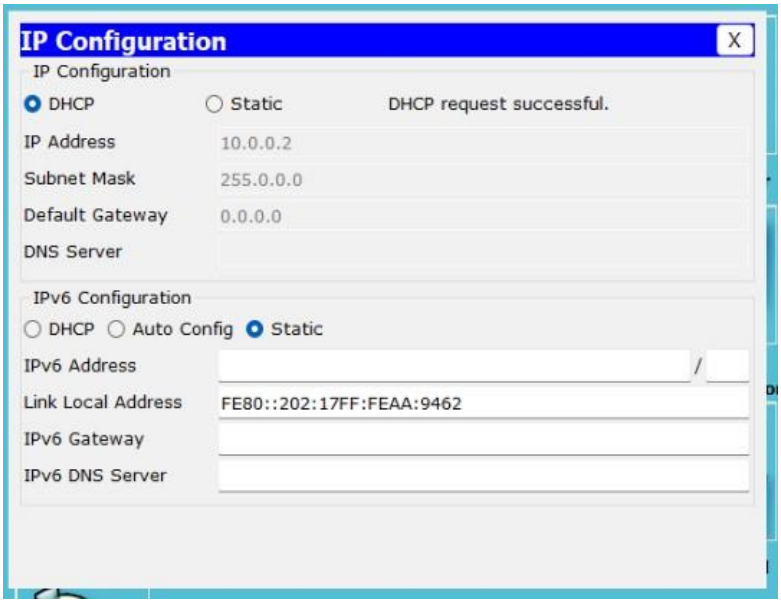
21/11/24

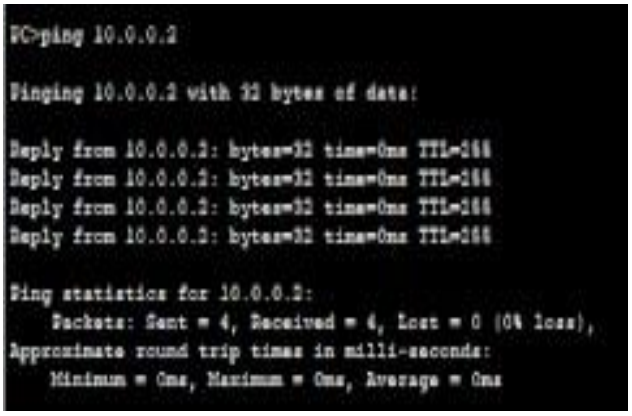
Screenshot of the topology:


10.0.0.2
Fa0
PC-PT
PC2
10.0.0.3
Fa0
PC-PT
PC5
Fa0
Fa0/1
Fa2/1
Fa4/1
Fa3/1
Fa1/1
Switch-PT
Switch2
Fa4/0
Fa0/0
Router-PT
Router1
20.0.0.1
Fa2/1
Fa0/1
Fa2/1
Fa1/1
Switch-PT
Switch3
Fa0
PC-PT
PC4
Fa0
Server-PT
Server0
10.0.0.1
Fa0
PC-PT
PC1
10.0.0.4

Question

output:



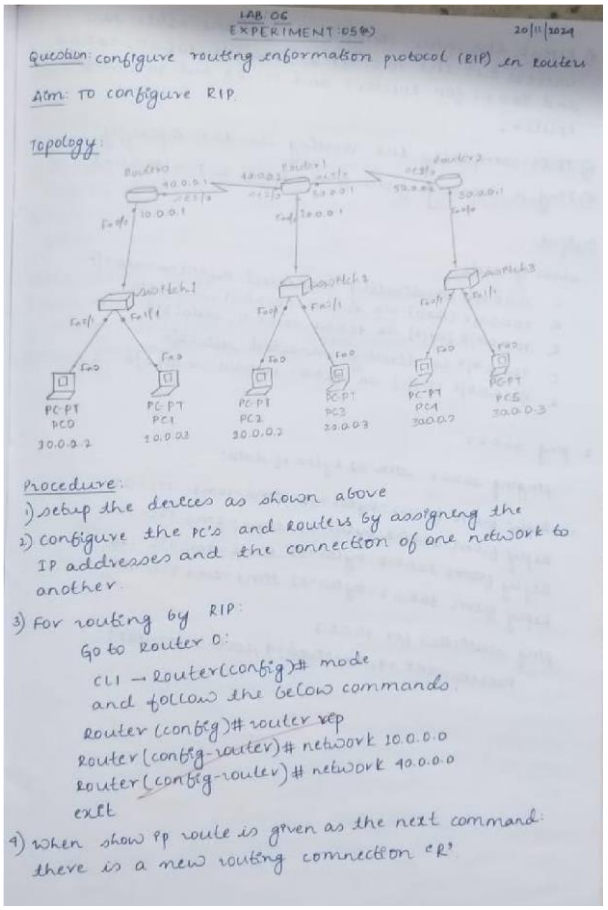


5:

Configure RIP routing Protocol in Routers.

Observation:

Screenshot of the



LAB 06
EXPERIMENT 05(a) 20/11/2024

Question: configure routing information protocol (RIP) in routers

Aim: TO configure RIP

Topology:

Procedure:

1) setup the devices as shown above
2) configure the PC's and Routers by assigning the IP addresses and the connection of one network to another
3) For routing by RIP:
   Go to Router 0:
   CLI → Router(config)# mode
   and follow the below commands
   Router(config)# router rip
   Router(config-router)# network 10.0.0.0
   Router(config-router)# network 40.0.0.0
   exit
4) When show ip route is given as the next command: there is a new routing connection 'R'

5) Repeat the same commands for the other two routers but the network address is 40.0.0.0, 20.0.0.0 and 50.0.0.0 for Router1 and 50.0.0.0 and 30.0.0.0 for Router2
6) This completes the routing in the topology
7) Ping a message from PC0 (10.0.0.2) to PC4 (30.0.0.2)

Output:

1. show ip route

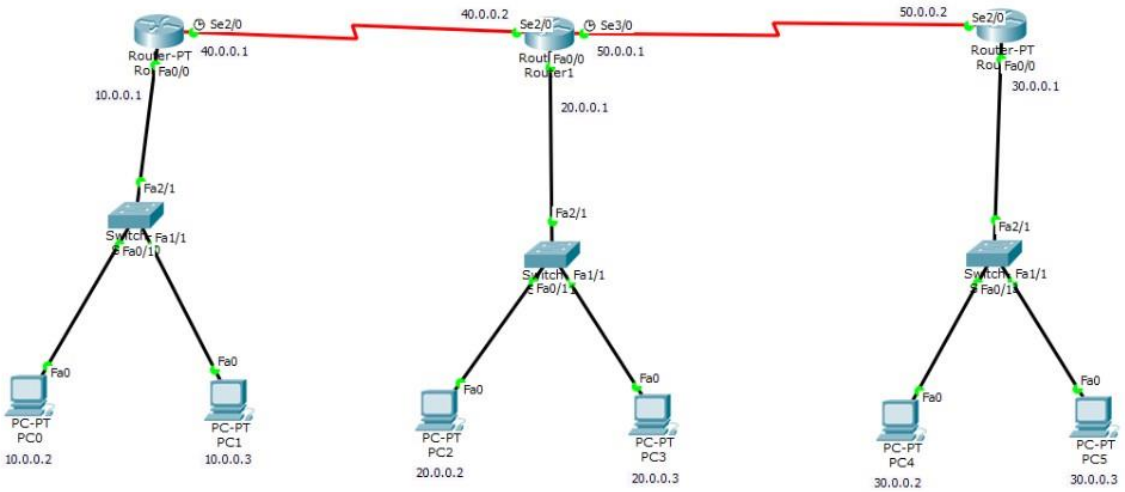
```
C   10.0.0.0/8 is directly connected, FastEthernet0/0
R   20.0.0.0/8 [120/1] via 40.0.0.2, 00:00:11, Serial2/0
R   30.0.0.0/8 [120/2] via 40.0.0.2, 00:00:11, Serial2/0
C   40.0.0.0/8 is directly connected, Serial2/0
R   50.0.0.0/8 [120/1] via 40.0.0.2, 00:00:11, Serial2/0
```

2. ping 30.0.0.2

```
Pinging 30.0.0.2 with 32 bytes of data:
Reply from 30.0.0.2: bytes=32 time=10ms TTL=125
Reply from 30.0.0.2: bytes=32 time=2ms TTL=125
Reply from 30.0.0.2: bytes=32 time=12ms TTL=125
Reply from 30.0.0.2: bytes=32 time=2ms TTL=125
ping statistics for 30.0.0.2:
    Packets: Sent=4, Received=4, Lost=0 (0% loss)
```

topology:

Question



Screenshot of the output:

```
R    10.0.0.0/8 [120/2] via 50.0.0.1, 00:00:07, Serial2/0
R    20.0.0.0/8 [120/1] via 50.0.0.1, 00:00:07, Serial2/0
C    30.0.0.0/8 is directly connected, FastEthernet0/0
R    40.0.0.0/8 [120/1] via 50.0.0.1, 00:00:07, Serial2/0
C    50.0.0.0/8 is directly connected, Serial2/0
```

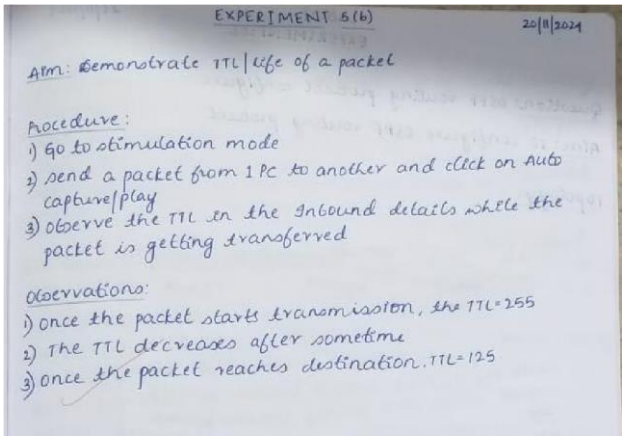
```
PC>ping 30.0.0.2

Pinging 30.0.0.2 with 32 bytes of data:

Reply from 30.0.0.2: bytes=32 time=10ms TTL=125
Reply from 30.0.0.2: bytes=32 time=2ms TTL=125
Reply from 30.0.0.2: bytes=32 time=12ms TTL=125
Reply from 30.0.0.2: bytes=32 time=2ms TTL=125

Ping statistics for 30.0.0.2:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 2ms, Maximum = 12ms, Average = 6ms
```

6:

Demonstrate the TTL/ Life of a Packet

Observation writeup images:

Screenshot of the



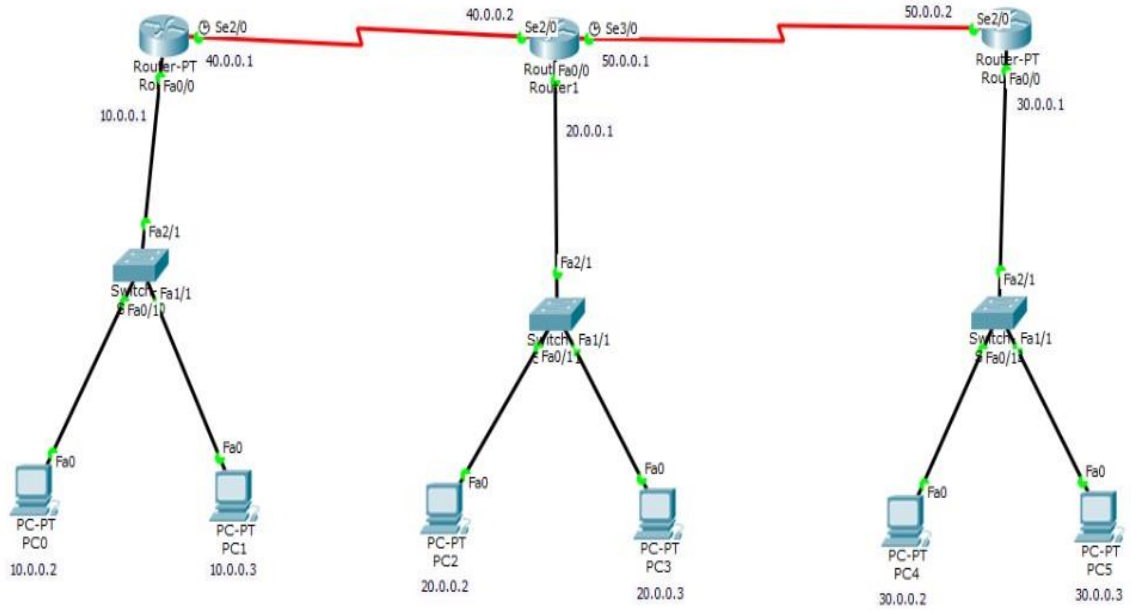
EXPERIMENT 5(b) 20/11/2024

Aim: Demonstrate TTL life of a packet

Procedure:
1) Go to simulation mode
2) Send a packet from 1 PC to another and click on Auto capture/play
3) Observe the TTL in the Inbound details while the packet is getting transferred

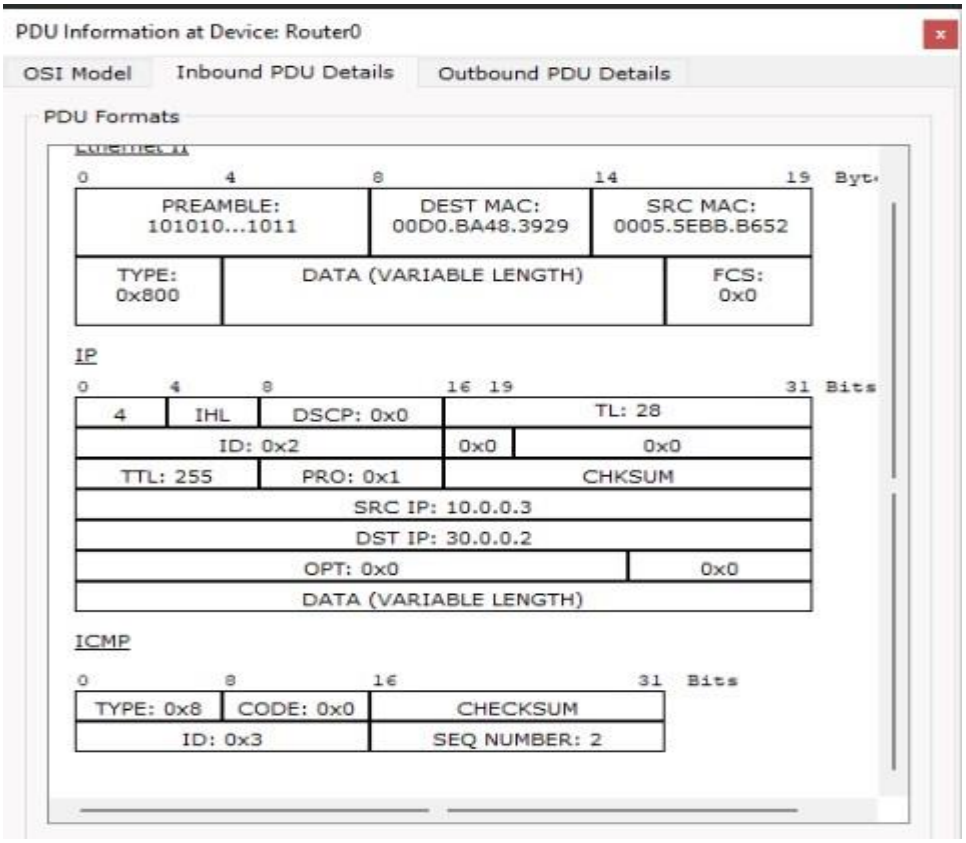
Observations:
1) Once the packet starts transmission, the TTL=255
2) The TTL decreases after sometime
3) Once the packet reaches destination. TTL=125

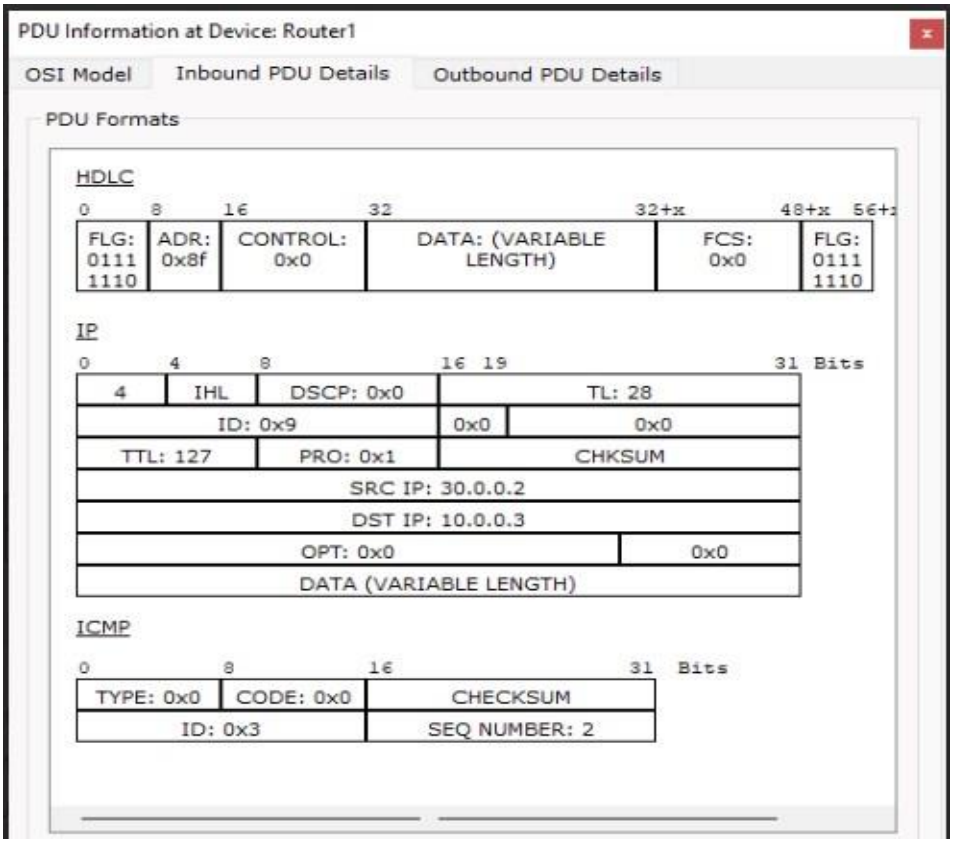
Screenshot of the topology:



output:

Question


PDU Information at Device: Router0
OSI Model
Inbound PDU Details
Outbound PDU Details
PDU Formats
0 4 8 14 19 Byt
PREAMBLE: 101010...1011
DEST MAC: 00D0.BA48.3929
SRC MAC: 0005.5EBB.B652
TYPE: 0x800
DATA (VARIABLE LENGTH)
FCS: 0x0
IP
0 4 8 16 19 31 Bits
4
IHL
DSCP: 0x0
TL: 28
ID: 0x2
0x0
0x0
TTL: 255
PRO: 0x1
CHKSUM
SRC IP: 10.0.0.3
DST IP: 30.0.0.2
OPT: 0x0
0x0
DATA (VARIABLE LENGTH)
ICMP
0 8 16 31 Bits
TYPE: 0x8
CODE: 0x0
CHECKSUM
ID: 0x3
SEQ NUMBER: 2

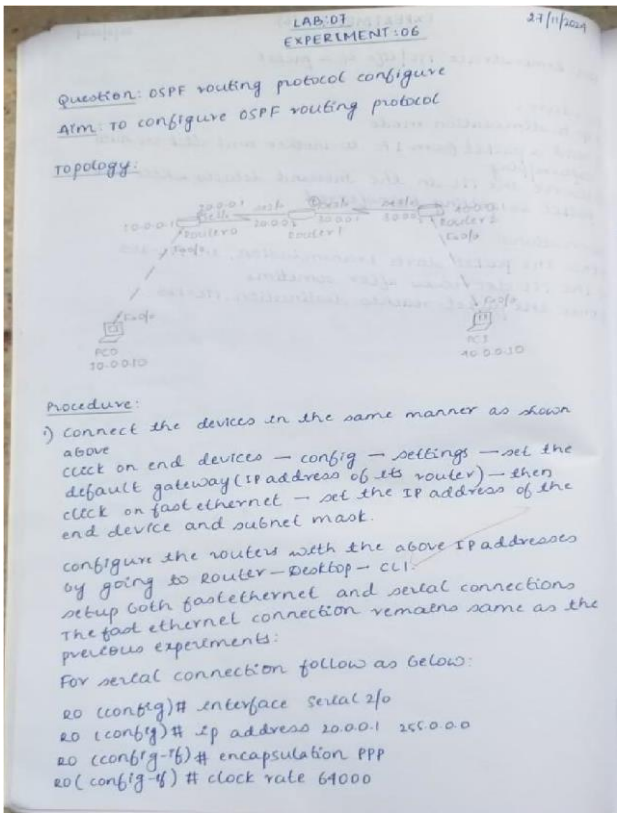

PDU Information at Device: Router1
OSI Model
Inbound PDU Details
Outbound PDU Details
PDU Formats
HDLC
0 8 16 32 32+x 48+x 56+
FLG: 0111 1110
ADR: 0x8f
CONTROL: 0x0
DATA: (VARIABLE LENGTH)
FCS: 0x0
FLG: 0111 1110
IP
0 4 8 16 19 31 Bits
4
IHL
DSCP: 0x0
TL: 28
ID: 0x9
0x0
0x0
TTL: 127
PRO: 0x1
CHKSUM
SRC IP: 30.0.0.2
DST IP: 10.0.0.3
OPT: 0x0
0x0
DATA (VARIABLE LENGTH)
ICMP
0 8 16 31 Bits
TYPE: 0x0
CODE: 0x0
CHECKSUM
ID: 0x3
SEQ NUMBER: 2

7:

Screenshot of the

Configure OSPF routing protocol

Observation :



LAB:07 EXPERIMENT:06 27/11/2024

Question: OSPF routing protocol configure

Aim: TO configure OSPF routing protocol

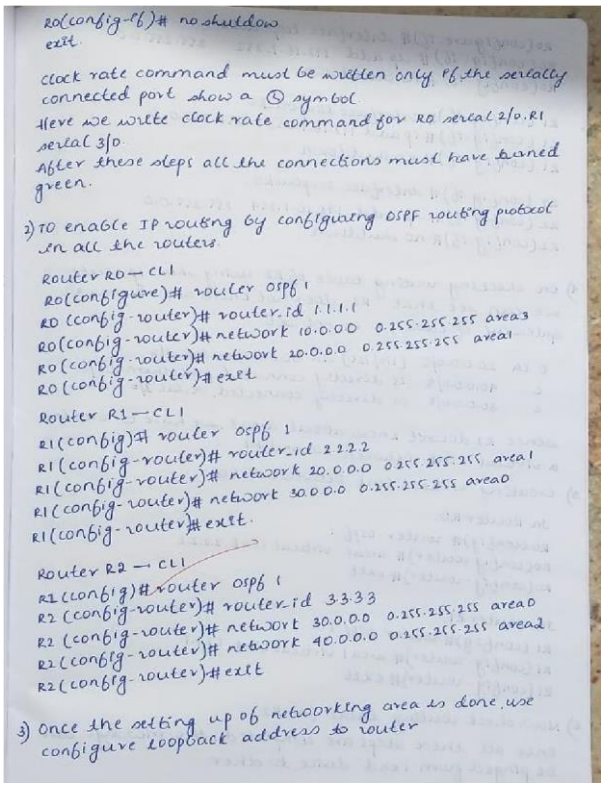
Topology:

Procedure:

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

configure the routers with the above IP addresses by going to Router → Desktop → CLI.
setup both fastethernet and serial connections
The fast ethernet connection remains same as the previous experiments:
For serial connection follow as below:

```
R0 (config)# interface serial 2/0
R0 (config)# ip address 20.0.0.1 255.0.0.0
R0 (config-if)# encapsulation PPP
R0(config-if)# clock rate 64000
```



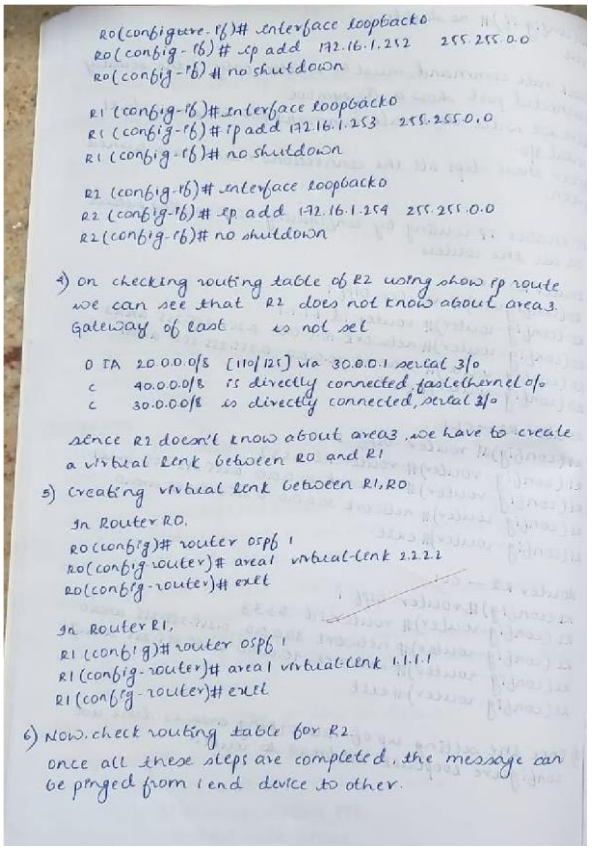
```
R0(config-if)# no shutdown
exit
```

clock rate command must be written only if the serially connected port show a Ⓢ symbol
Here we write clock rate command for R0 serial 2/0, R1 serial 3/0.
After these steps all the connections must have turned green.

2) TO enable IP routing by configuring OSPF routing protocol in all the routers

```
Router R0 — CLI
R0(configure)# router ospf 1
R0 (config-router)# router-id 1.1.1.1
R0(config-router)# network 10.0.0.0 0.255.255.255 area3
R0(config-router)# network 20.0.0.0 0.255.255.255 area1
R0(config-router)# exit

Router R1 — CLI
R1(config)# router ospf 1
R1(config-router)# router-id 2.2.2.2
R1(config-router)# network 20.0.0.0 0.255.255.255 area1
R1(config-router)# network 30.0.0.0 0.255.255.255 area0
R1(config-router)# exit

Router R2 — CLI
R2(config)# router ospf 1
R2 (config-router)# router-id 3.3.3.3
R2 (config-router)# network 30.0.0.0 0.255.255.255 area0
R2 (config-router)# network 40.0.0.0 0.255.255.255 area2
R2(config-router)# exit
```

3) Once the setting up of networking area is done, use configure loopback address to router



R0(configure-rb)# interface loopback0
R0(config-rb)# ip add 172.16.1.252 255.255.0.0
R0(config-rb)# no shutdown

R1(config-rb)# interface loopback0
R1(config-rb)# ip add 172.16.1.253 255.255.0.0
R1(config-rb)# no shutdown

R2(config-rb)# interface loopback0
R2(config-rb)# ip add 172.16.1.254 255.255.0.0
R2(config-rb)# no shutdown

4) On checking routing table of R2 using show ip route we can see that R2 does not know about area3. Gateway of last is not set

O IA 20.0.0.0/8 [110/125] via 30.0.0.1 serial 3/0
C 40.0.0.0/8 is directly connected fastethernet 0/0
C 30.0.0.0/8 is directly connected, serial 3/0

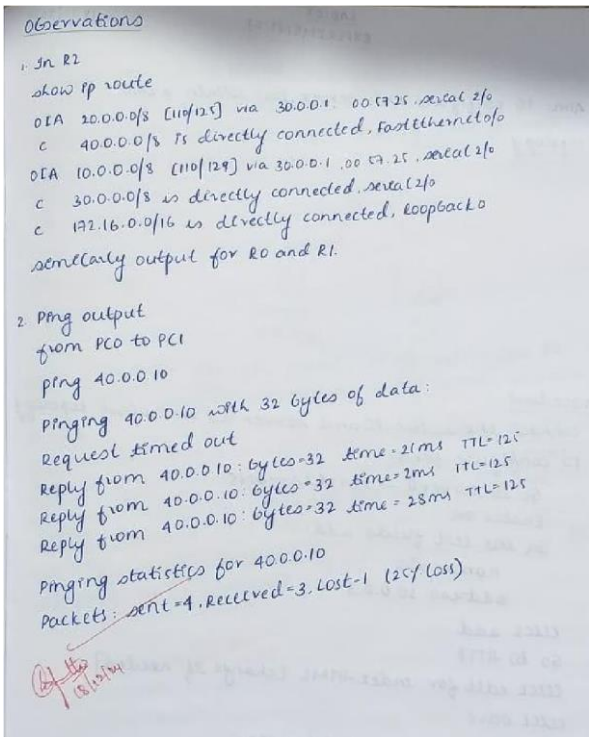
since R2 doesn't know about area3, we have to create a virtual link between R0 and R1

5) Creating virtual link between R1, R0

In Router R0.
R0(config)# router ospf 1
R0(config-router)# area1 virtual-link 2.2.2.2
R0(config-router)# exit

In Router R1,
R1(config)# router ospf 1
R1(config-router)# area1 virtual-link 1.1.1.1
R1(config-router)# exit

6) Now, check routing table for R2.
once all these steps are completed, the message can be pinged from end device to other.

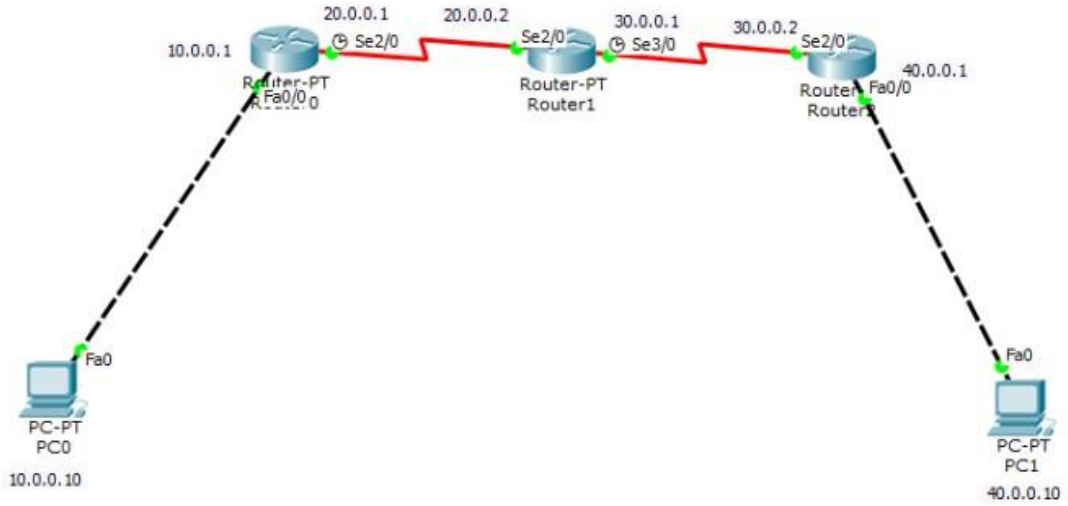


Observations

1. In R2
show ip route
O IA 20.0.0.0/8 [110/125] via 30.0.0.1 00:57:25, serial 2/0
C 40.0.0.0/8 is directly connected, FastEthernet0/0
O IA 10.0.0.0/8 [110/129] via 30.0.0.1, 00:57:25, serial 2/0
C 30.0.0.0/8 is directly connected, serial 2/0
C 172.16.0.0/16 is directly connected, loopback0
similarly output for R0 and R1.

2. Ping output
from PC0 to PC1
ping 40.0.0.10
Pinging 40.0.0.10 with 32 bytes of data:
Request timed out
Reply from 40.0.0.10: bytes=32 time=21ms TTL=125
Reply from 40.0.0.10: bytes=32 time=2ms TTL=125
Reply from 40.0.0.10: bytes=32 time=28ms TTL=125
Pinging statistics for 40.0.0.10
Packets: sent=4, received=3, Lost=1 (25% loss)

Question



Screenshot of the output:

```
C    10.0.0.0/8 is directly connected, FastEthernet0/0
     20.0.0.0/8 is variably subnetted, 2 subnets, 2 masks
C       20.0.0.0/8 is directly connected, Serial2/0
C       20.0.0.2/32 is directly connected, Serial2/0
O    30.0.0.0/8 [110/128] via 20.0.0.2, 00:05:09, Serial2/0
O IA 40.0.0.0/8 [110/129] via 20.0.0.2, 00:05:09, Serial2/0
C    172.16.0.0/16 is directly connected, Loopback0
```

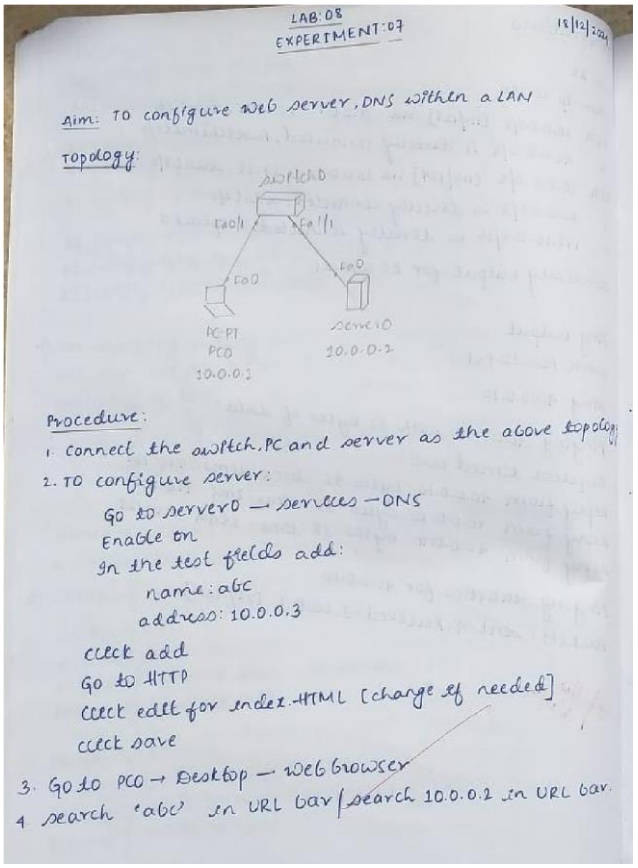
```
PC>ping 40.0.0.10

Pinging 40.0.0.10 with 32 bytes of data:

Request timed out.
Reply from 40.0.0.10: bytes=32 time=6ms TTL=125
Reply from 40.0.0.10: bytes=32 time=7ms TTL=125
Reply from 40.0.0.10: bytes=32 time=9ms TTL=125

Ping statistics for 40.0.0.10:
    Packets: Sent = 4, Received = 3, Lost = 1 (25% loss),
Approximate round trip times in milli-seconds:
    Minimum = 6ms, Maximum = 9ms, Average = 7ms
```

Screenshot of the topology:

8:

Configure Web Server, DNS within a LAN.

Observation:



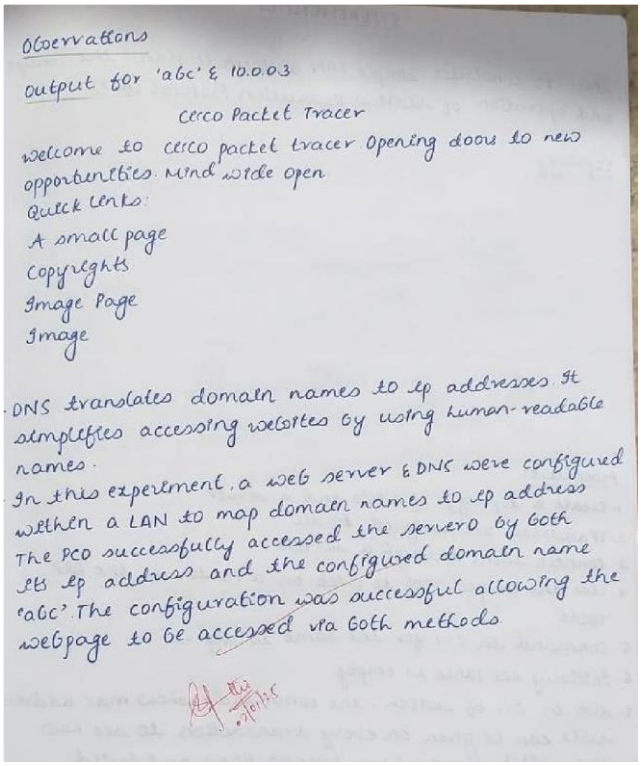
LAB: 08
EXPERIMENT: 07

Aim: To configure Web server, DNS within a LAN

Topology:

Procedure:

1. Connect the switch, PC and server as the above topology
2. To configure server:
   - Go to server0 → services → DNS
   - Enable on
   - In the text fields add:
     - name: abc
     - address: 10.0.0.3
   - click add
   - Go to HTTP
   - click edit for index.HTML [change if needed]
   - click save
3. Go to PC0 → Desktop → Web browser
4. search 'abc' in URL bar / search 10.0.0.2 in URL bar.



Observations

Output for 'abc' & 10.0.0.3

Cisco Packet Tracer

Welcome to cisco packet tracer Opening doors to new opportunities. Mind wide open

Quick Links:

A small page

Copyrights

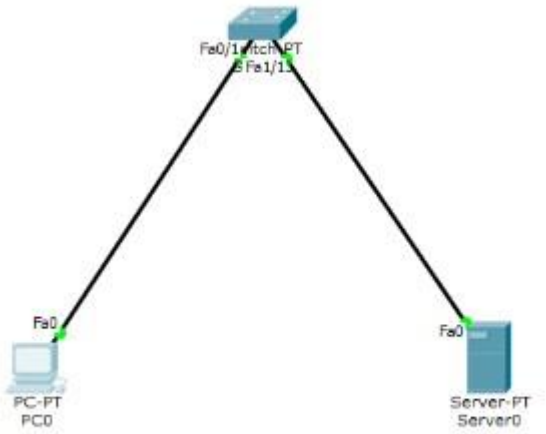
Image Page

Image

DNS translates domain names to ip addresses. It simplifies accessing websites by using human-readable names.

In this experiment, a web server & DNS were configured within a LAN to map domain names to ip address. The PC0 successfully accessed the server0 by both its ip address and the configured domain name 'abc'. The configuration was successful allowing the webpage to be accessed via both methods.

Question



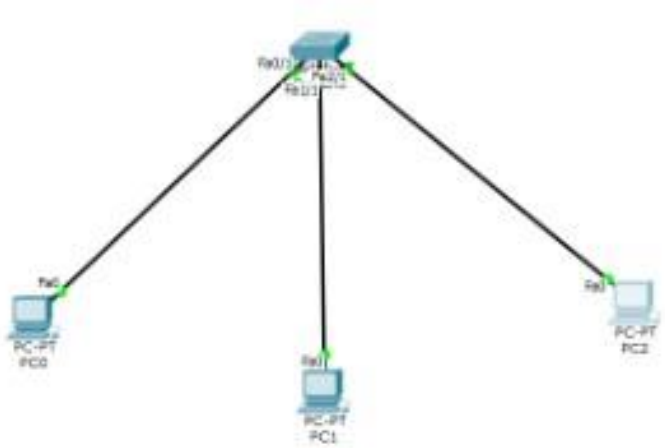
Screenshot of the output:

Screenshot of the topology:

9:

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

Observation:



Screenshot of the output:

```
Switch>show mac address-table
          Mac Address Table
-------------------------------------------

Vlan    Mac Address       Type        Ports
----    -----------       --------    -----

   1    000b.bed0.714b    DYNAMIC     Fa2/1
   1    0060.3e41.e693    DYNAMIC     Fa1/1
   1    0090.2ba4.59e4    DYNAMIC     Fa0/1
```

10:

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

Question
To
Observation:

Screenshot of the output:

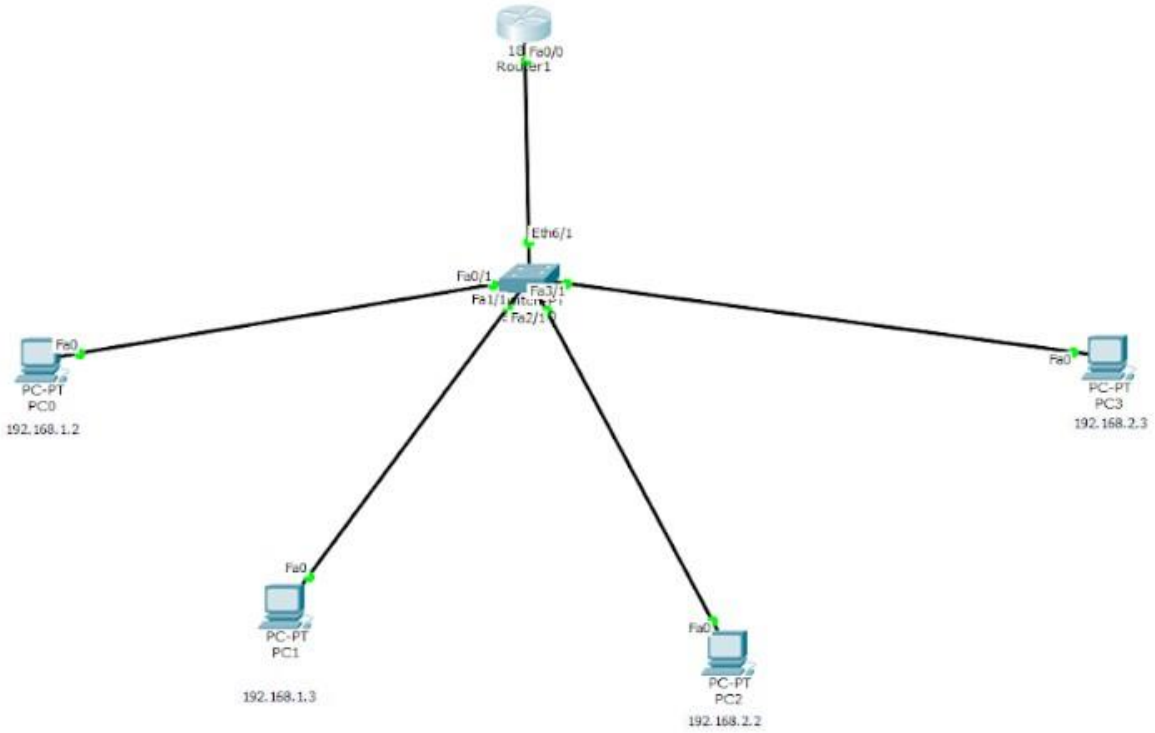
```
PC>telnet 10.0.0.10
Trying 10.0.0.10 ...Open


User Access Verification

Password:
Password:
R1>pl
Translating "pl"...domain server (255.255.255.255)
% Unknown command or computer name, or unable to find computer address

R1>enable
Password:
Password:
R1#
```

11:

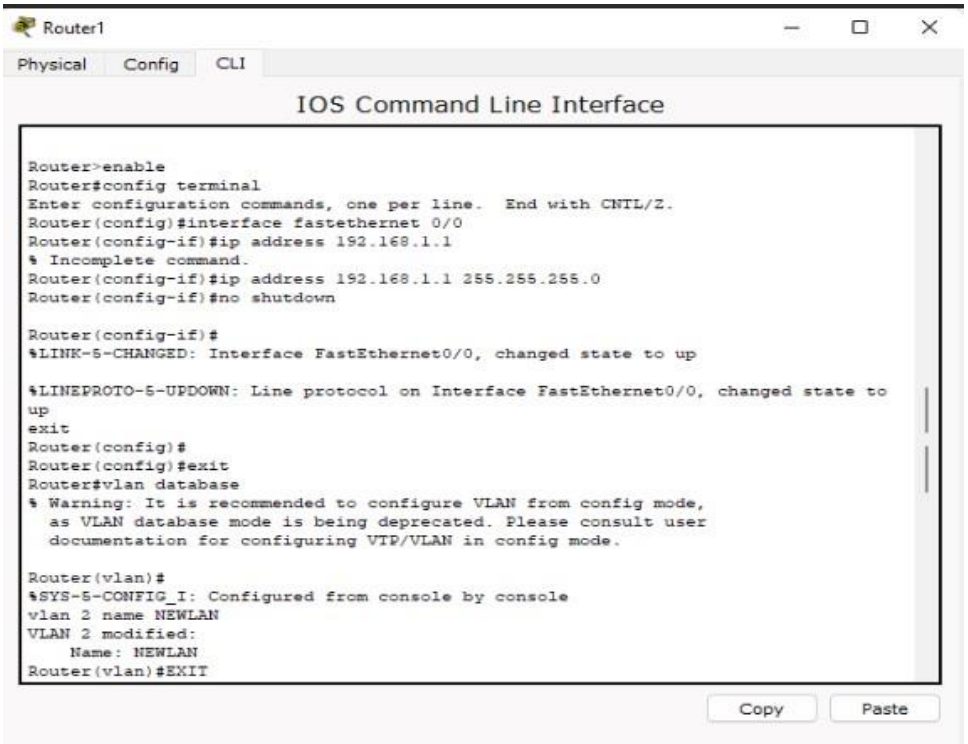
construct a VLAN and make the PC's communicate among a VLAN

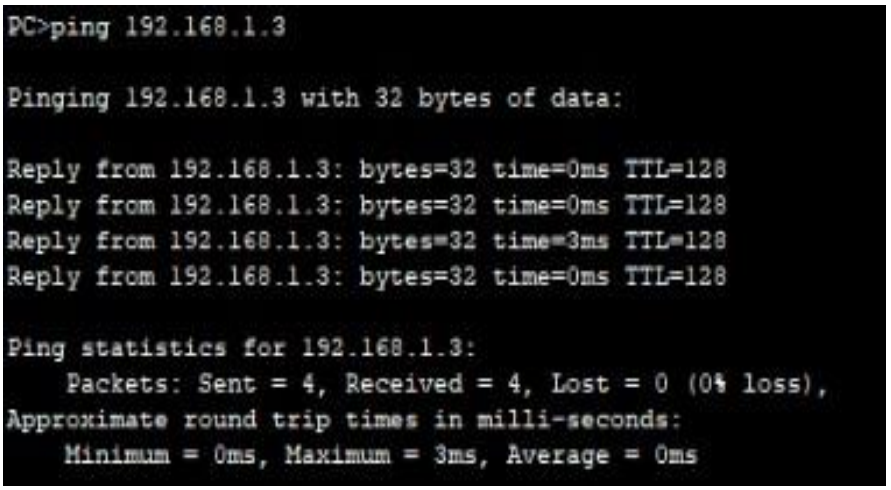
Observation:

Screenshot of the topology:



Screenshot of the output:



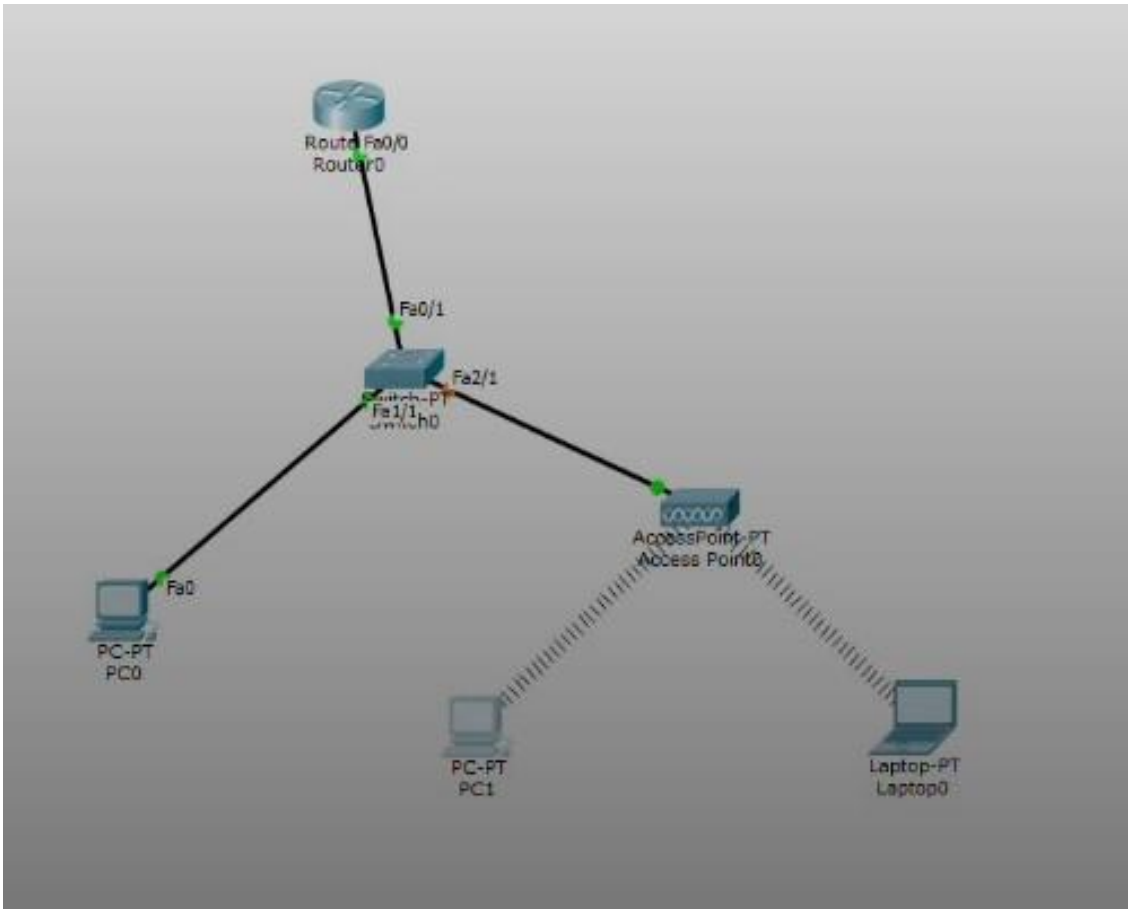

PC>ping 192.168.1.3
Pinging 192.168.1.3 with 32 bytes of data:
Reply from 192.168.1.3: bytes=32 time=0ms TTL=128
Reply from 192.168.1.3: bytes=32 time=0ms TTL=128
Reply from 192.168.1.3: bytes=32 time=3ms TTL=128
Reply from 192.168.1.3: bytes=32 time=0ms TTL=128
Ping statistics for 192.168.1.3:
Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
Minimum = 0ms, Maximum = 3ms, Average = 0ms

Question 12:

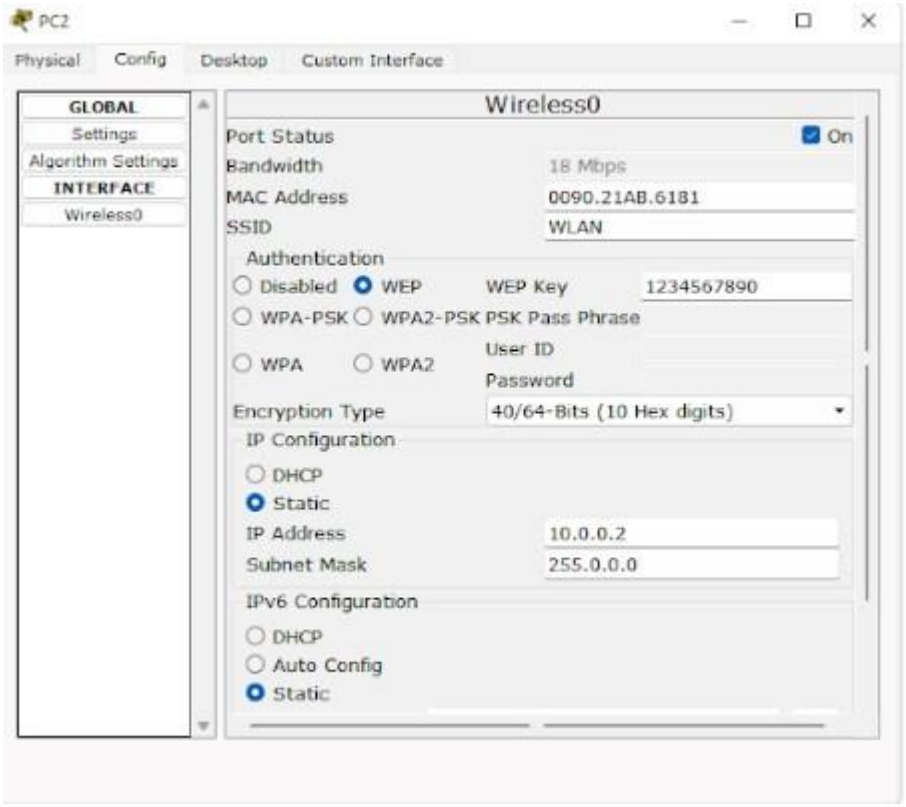
To construct a WLAN and make the nodes communicate wirelessly

Observation :

Screenshot of the topology:



Screenshot of the output:



CYCLE-2

Question 1:

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

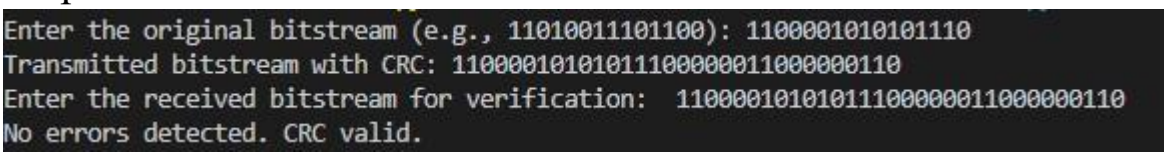
Code:

```
def crc_ccitt_16_bitstream(bitstream: str, poly: int = 0x1021, init_crc: int = 0xFFFF) -> int:
    crc = init_crc for
    bit in bitstream:
```

```
        crc ^= int(bit) << 15 # Align the bit with CRC's uppermost bit
        for _ in range(8): # Process each bit if crc & 0x8000: # Check
        if the leftmost bit is set crc = (crc << 1) ^ poly
            else:
                crc <<= 1 crc &= 0xFFFF # Ensure CRC
            remains 16-bit
    return crc


def append_crc_to_bitstream(bitstream: str) -> str:
    """
    Append the calculated 16-bit CRC to the given bitstream.
    """ crc = crc_ccitt_16_bitstream(bitstream) crc_bits =
    f"{crc:016b}" # Convert CRC to a 16-bit binary string return
    bitstream + crc_bits


def verify_crc_bitstream(bitstream_with_crc: str) -> bool:
    """
    Verify the CRC of the given bitstream with CRC appended.
    """ if len(bitstream_with_crc) <
    16:
        return False # Not enough bits to contain CRC data, received_crc =
    bitstream_with_crc[:-16], bitstream_with_crc[-16:] calculated_crc =
    crc_ccitt_16_bitstream(data) return calculated_crc == int(received_crc,
    2)
```

```
# Main Program if __name__ == "__main__": # User input for original bitstream
message_bits = input("Enter the original bitstream (e.g., 11010011101100): ").strip()
    # Validate input if not all(bit in "01" for bit
    in message_bits):
        print("Invalid input. Please enter a binary bitstream (e.g., 11010011101100).")
    else:
        # Calculate and append CRC bitstream_with_crc =
        append_crc_to_bitstream(message_bits) print(f"Transmitted
        bitstream with CRC: {bitstream_with_crc}")

        # User input for received bitstream user_bitstream = input("Enter the
        received bitstream for verification: ").strip()

        # Validate received input if not all(bit in "01"
        for bit in user_bitstream):
            print("Invalid input. Please enter a valid binary bitstream.")
        elif len(user_bitstream) < 16:
            print("Invalid input. Received bitstream must include at least 16 bits for CRC.")
        else:
            # Verify CRC
            is_valid = verify_crc_bitstream(user_bitstream)
            if is_valid:
                print("No errors detected. CRC valid.")
            else: print("Error detected! CRC
                invalid.")
```

Output:



Question 2:

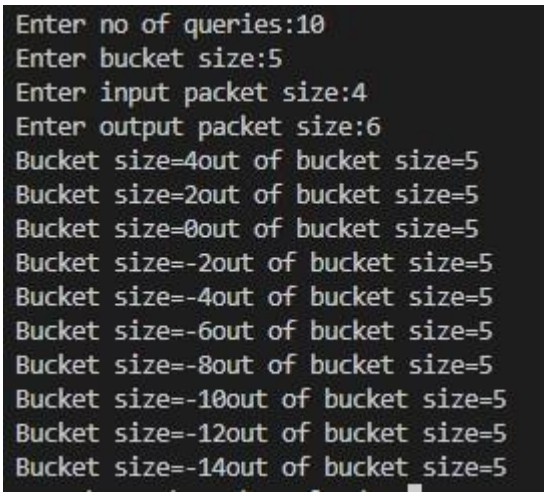
Write a program for congestion control using Leaky bucket algorithm.

Program:

Code:

```
storage=0 noofqueries=int(input("Enter no of

queries:")) bucketsize=int(input("Enter bucket

size:")) inputpktsize=int(input("Enter input packet

size:")) outputpktsize=int(input("Enter output packet

size:")) for i in range(0,noofqueries):

sizeleft=bucketsize-storage if

inputpktsize<=sizeleft: storage+=inputpktsize

    else:

        print("Packet loss=", inputpktsize) print(f"Bucket

    size={storage}out of bucket size={bucketsize}") storage-

    =outputpktsize
```

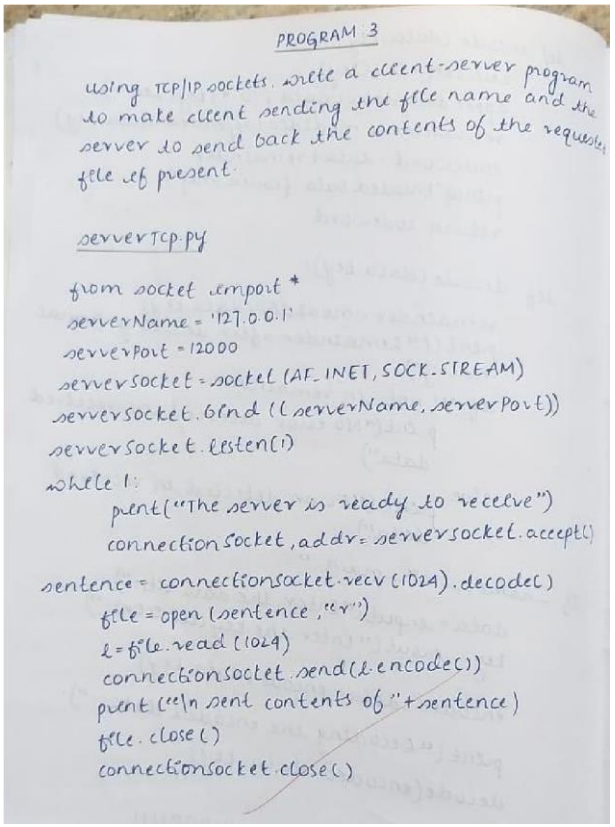
Output:


Enter no of queries:10
Enter bucket size:5
Enter input packet size:4
Enter output packet size:6
Bucket size=4out of bucket size=5
Bucket size=2out of bucket size=5
Bucket size=0out of bucket size=5
Bucket size=-2out of bucket size=5
Bucket size=-4out of bucket size=5
Bucket size=-6out of bucket size=5
Bucket size=-8out of bucket size=5
Bucket size=-10out of bucket size=5
Bucket size=-12out of bucket size=5
Bucket size=-14out of bucket size=5

Question 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.

Program:

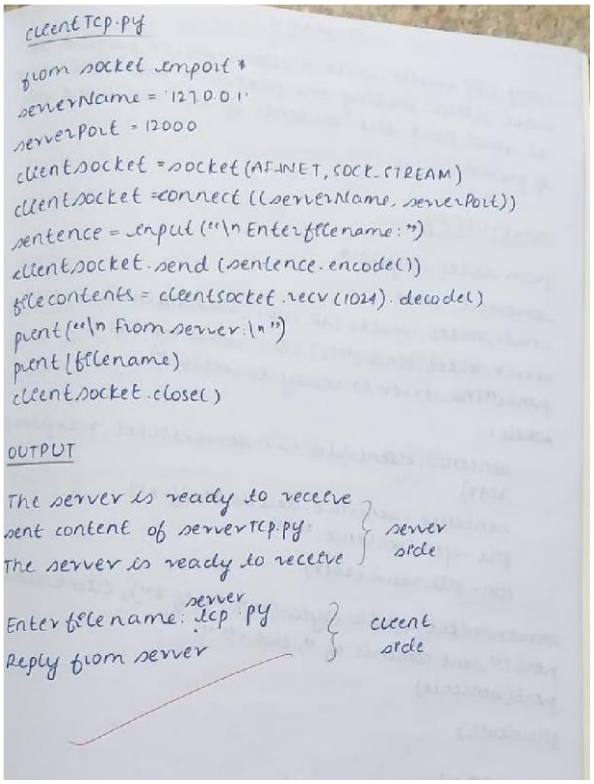


PROGRAM 3

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.

serverTcp.py

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



Code:

tcpserver.py: from socket import *

serverName="127.0.0.1" serverPort = 14000

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()

tcpclient.py: from socket import * serverName

= '127.0.0.1' serverPort = 14000 clientSocket =

socket(AF_INET, SOCK_STREAM)

clientSocket.connect((serverName,serverPort))

sentence = input("\nEnter file name: ")

clientSocket.send(sentence.encode()) filecontents =

clientSocket.recv(1024).decode() print ('\nFrom

Server:\n') print(filecontents) clientSocket.close()

Output:

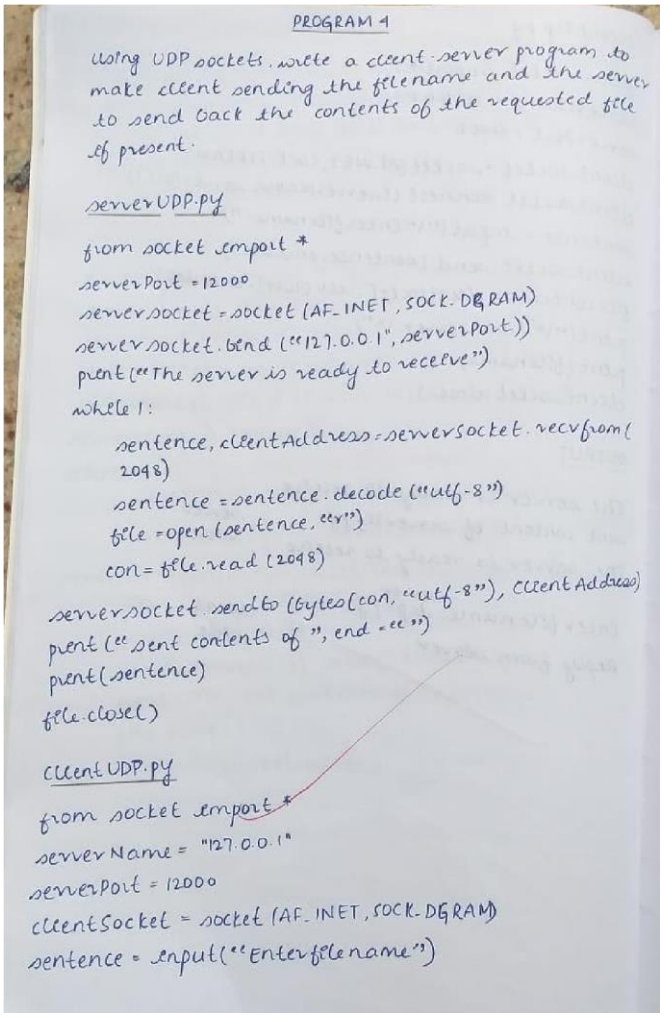
```
The server is ready to receive

Sent contents of tcpserver.py
The server is ready to receive
```

```
Enter file name: tcpserver.py

From Server:

from socket import *
serverName="127.0.0.1"
serverPort = 14000
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()
```

Question 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.

Program:



PROGRAM 1

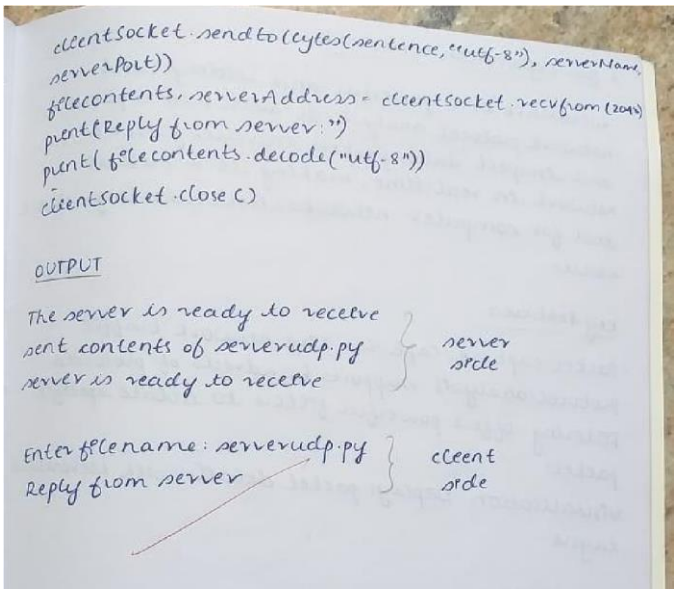
Using UDP sockets, write a client-server program to make client sending the filename and the server to send back the contents of the requested file if present.

serverUDP.py

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

clientUDP.py

```
from socket import *
serverName = "127.0.0.1"
serverPort = 12000
clientSocket = socket (AF_INET, SOCK_DGRAM)
sentence = input("Enter filename")
```



```
clientSocket.sendto(cytes(sentence, "utf-8"), serverName,
serverPort))
filecontents, serverAddress = clientSocket.recvfrom (2048)
print("Reply from server:")
print( filecontents.decode("utf-8"))
clientSocket.close()
```

OUTPUT

```
The server is ready to receive
sent contents of serverudp.py
server is ready to receive
```

server side

```
Enter filename: serverudp.py
Reply from server
```

client side

Code:

udpserver.py: from socket import * serverPort =

12000 serverSocket = socket(AF_INET,

SOCK_DGRAM)

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

udpclient.py:

```
from socket import * serverName = "127.0.0.1"
serverPort = 12000 clientSocket = socket(AF_INET,
SOCK_DGRAM) sentence = input("\nEnter file name: ")
clientSocket.sendto(bytes(sentence,"utf-8"),(serverName,
serverPort)) filecontents,serverAddress =
clientSocket.recvfrom(2048)
print ('\nReply from Server:\n')
print (filecontents.decode("utf-8"))
# for i in filecontents:
   # print(str(i), end = '')
clientSocket.close()
clientSocket.close()
```

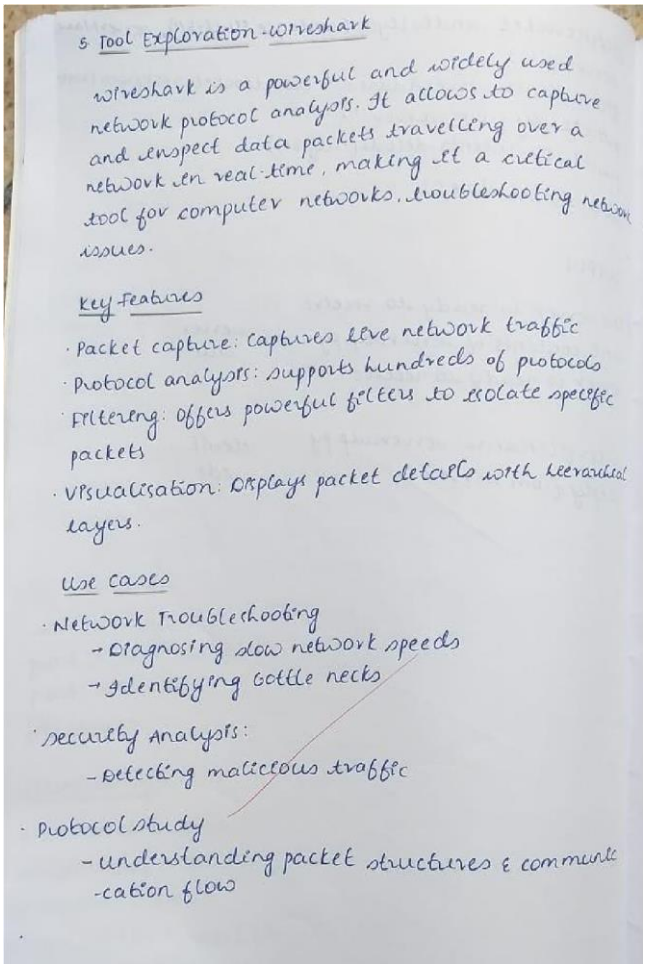
Output:

```
The server is ready to receive

Sent contents of  udpserver.py
```

```
Enter file name:  udpserver.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)
```

Question 5:

Tool Exploration -Wireshark



5 Tool Exploration - Wireshark

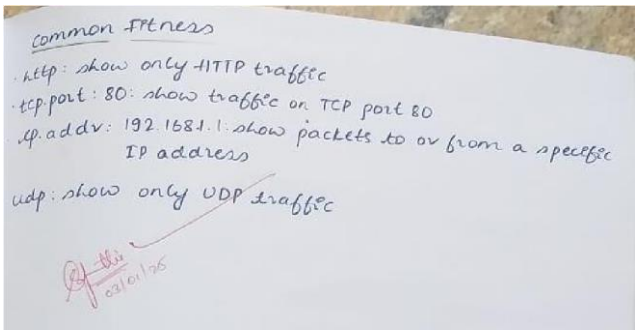
Wireshark is a powerful and widely used network protocol analysis. It allows to capture and inspect data packets travelling over a network in real-time, making it a critical tool for computer networks, troubleshooting network issues.

Key Features

- Packet capture: Captures live network traffic
- Protocol analysis: Supports hundreds of protocols
- Filtering: Offers powerful filters to isolate specific packets
- Visualisation: Displays packet details with hierarchical layers.

Use cases

- Network Troubleshooting
  - Diagnosing slow network speeds
  - Identifying bottle necks
- Security Analysis:
  - Detecting malicious traffic
- Protocol study
  - understanding packet structures & communication flow



Common Filters

- http: show only HTTP traffic
- tcp.port: 80: show traffic on TCP port 80
- ip.addr: 192.168.1.1: show packets to or from a specific IP address
- udp: show only UDP traffic

03/01/26