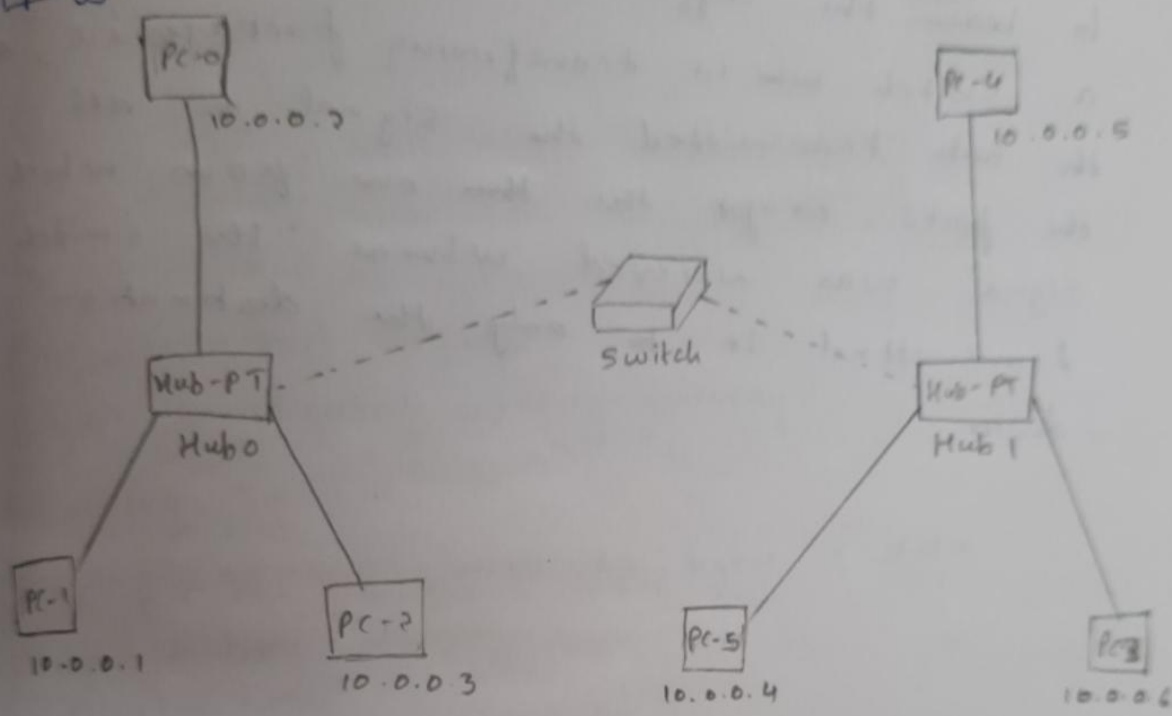
16-6-23

Experiment 1

1. Create a topology consisting of 3 or more devices connected with the help of a hub and switch as connecting devices and demonstrate ping message. Send PDU from source to destination

Topology:



Observation

The PC-0 sends the message to hub1 which recieves & then broadcoasts to PC1, PC2 and switch. The switch then sends it to Hub2 which recieves it & broadcoasts it to all the devices PC-3, PC-4, PC-5. The PC-4 which is the destination device & has the address match with the message will accept it.

Ping command: (from 10.0.0.2)

Ping 10.0.0.6

Reply from 10.0.0.6 byte=32 time=0ms TTL=128

Reply from 10.0.0.6 byte=32 time=0ms TTL=128

Reply from 10.0.0.6 byte=32 time=0ms TTL=128

Reply from 10.0.0.6 byte=32 time=0ms TTL=128

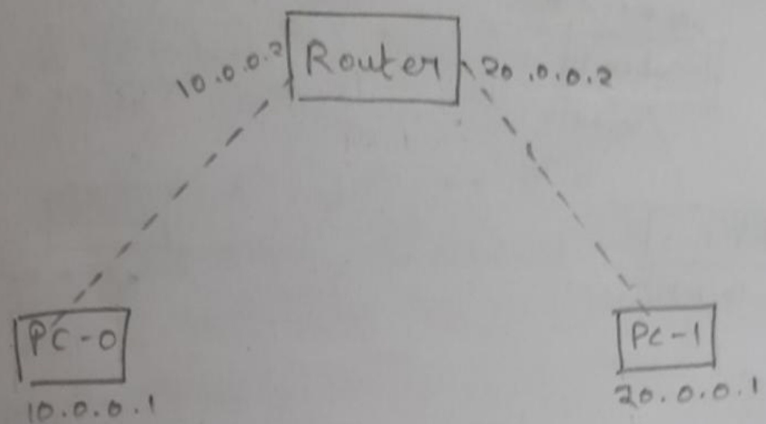
Reply from 10.0.0.6 byte = 32 time = 0ms TTL=128
Ping statistics for 10.0.0.6
packets sent = 4, recieved = 4, loss=0

<u>Outcome:</u>
Through this experiment we demonstrated the use of switch and hub. This enabled us to ~~Learn~~ notice the difference between a hub and a switch ~~wh~~ in transfering packages i.e., ~~a~~ the hub transmitted the signal to all the ports except the ~~the~~ one from which signal was recieved whereas the switch transmitted it to only the destination device.

23-6-23

Experiment 2

2. Create a topology consisting of 2 devices connected with the help of a router.



Ping command Output:

Before ~~congfiguring~~ entering gateway:

PC > ping 20.0.0.1

Pinging 20.0.0.1 with 32 bytes of data

Request timed out
Request timed out
Request timed out
Request timed out

Ping statistics for 20.0.0.1:
Packets : Sent = 4, Received = 0, Lost = 4 (100% loss)

After entering gateway:

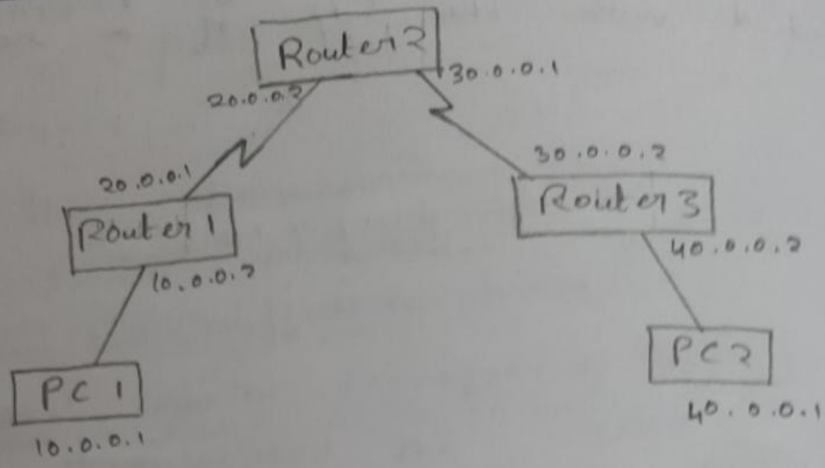
PC > ping 20.0.0.1

Pinging 20.0.0.1 with 32 bytes of data

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

Ping statistics for 20.0.0.1
Packets : Sent = 4, Received = 4, Lost = 0 (0% Loss)

Another Topology (Hybrid)



Observation:

When we ping from PC1, it fails to get a successfully get a reply before we ~~con~~ statically configure all the routers.

Ping only works successfully when we statically add all the networks.

Ping Command (RESULT):

~~Ping 20.0.0.2~~ Router > enable
Router # configure terminal
Router (config) # interface fastEthernet 0/0
Router (config-if) # ip address 10.0.0.2 255.0.0.0
Router (config-if) # no shutdown

PC>Ping 20.0.0.2

~~Request timed~~

Pinging 20.0.0.2 with 32 bytes of data

Request timed out
Request timed out
Request timed out
Request timed out

Ping statistics for 20.0.0.2
Packets Sent = 4, Received - 0, loss = 4

PC > ping 20.0.0.1

Pinging 20.0.0.1 with 32 bytes of data

Reply from 20.0.0.1 time=0ms TTL=128

Reply from 20.0.0.1 time=0ms TTL=128

Reply from 20.0.0.1 time=0ms TTL=128

Reply from 20.0.0.1 time=0ms TTL=128

ping statistics for 20.0.0.1

packets sent = 4, recieved = 4, loss = 0%.

PC > ping 30.0.0.1

Pinging 30.0.0.1 with 32 bytes of data

Reply from 20.0.0.2 Destination host unreachable

Reply from 20.0.0.2 Destination host unreachable

Reply from 20.0.0.2 Destination host unreachable

Reply from 20.0.0.2 Destination host unreachable

Ping statistics for 30.0.0.1

packets sent = 4, recieved = 0, loss = 4

For network 20.0.0.02, the ping command shows request timed out as message reaches host but host is unable to respond back.

For network 30.0.0.1, the ping command shows destination host unreachable as ~~this network~~ here the message is unable to reach the network 30.0.0.1 and hence it shows. destination host unreachable.

Outcome:

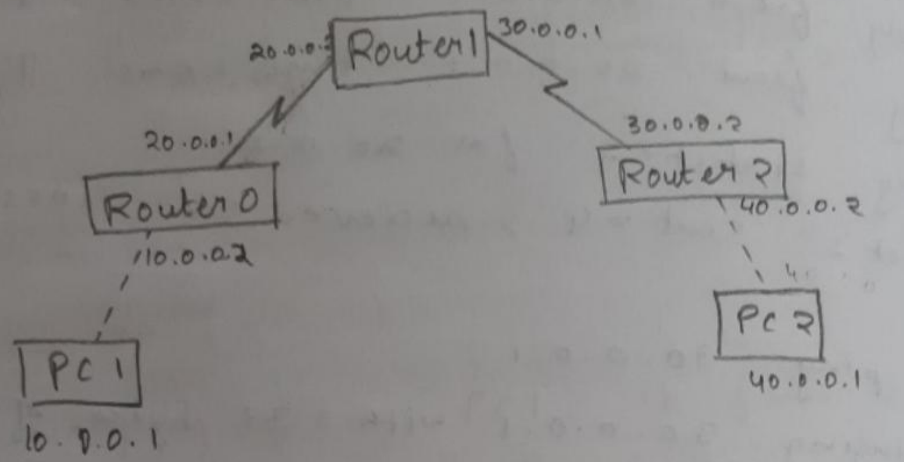
Through this experiment, we learnt how to configure IP address for the routers and also the possible ping responces like destination host unreachable, request timed out and successful replies along with the reason for getting such responces.

30-6-23

Experiment 3

Q. Send packets from one end device to another through multiple routers.

TOPOLOGY



By static routing:

Commands

Router >enable

Router # configure terminal

Router (config)# ip ~~address~~ route 30.0.0.0 255.0.0.0 20.0.0.2

Router (config) # ip ~~address~~ route 40.0.0.0 255.0.0.0 20.0.0.2

Router >show ip route

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

S - 30.0.0.0/8 [1/0] via 20.0.0.2

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

S - 40.0.0.0 [1/0] via 20.0.0.2

Output (Ping)

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 = 7ms TTL=125

reply from 30.0.0.2 bytes=32 time=2ms TTL=125

Reply from 30.0.0.2 bytes=32 time =2ms TTL=125

Reply from 30.0.0.2 bytes = 32 time = 7ms TTL = 125

ping statistics for 30.0.0.2

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

Approximate round trip in ms:

minimum = 2ms, maximum = 7ms, Average = 4ms

Observation:

Initially the package sent to router0 was not set to router 1 as it didn't have any knowledge of 30.0.0.0 or 40.0.0.0 network. It was statically entered into router 0. Similarly router 1 and 2 also didn't know about other networks. Once entered, the packet was successfully sent from PC1 to PC2 via the routers.

Using default routing:

Commands

Router > enable

Router # configure terminal

Router (config) # ip route 0.0.0.0 0.0.0.0 20.0.0.2

Router > show ip route

Gateway of last resort is 20.0.0.2 to network 0.0.0.0

S* 0.0.0.0 [1/0] via 20.0.0.2

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

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

[This is possible for router 0 and router 2. Router1 is configured using ~~defa~~ static routing as above]

Output (Ping)

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=8ms TTL=125

Reply from 40.0.0.1 bytes=32 time=8ms TTL=125

Reply from 40.0.0.1 bytes=32 time=8ms TTL=125

Reply from 40.0.0.1 bytes=32 time=8ms TTL=125

ping statistics for 40.0.0.1

packets sent =4, recieved=4, lost =0

Approximate round trip in milli-seconds

minimum=8ms, maximum=8ms, Average=8ms

Observation:

The packets reach destination successfully but this time default routing was used on routers connected to end devices. Default routing implies it can go to any of the connected networks as we specify the network address as 0.0.0.0.

Outcome:

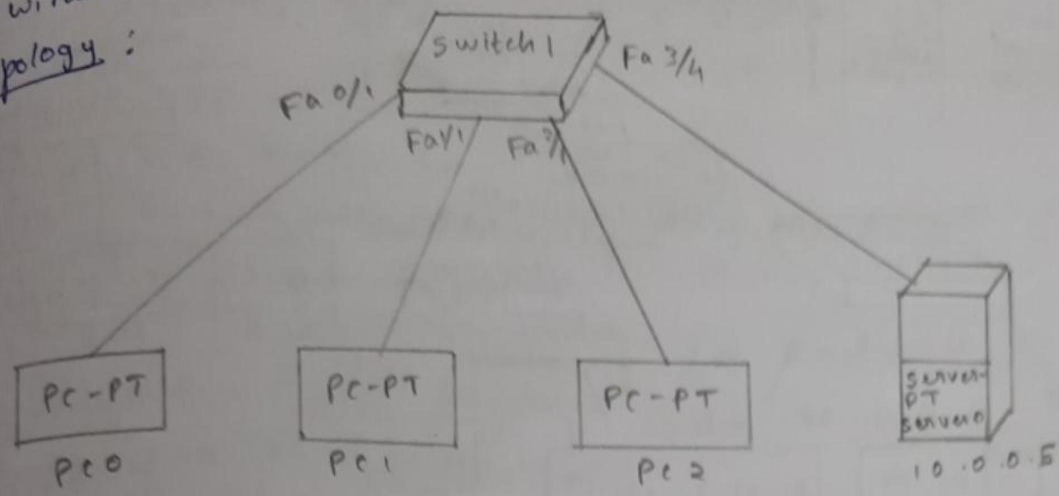
Through this experiment we learnt how to configure routers statically and by default method. This solved the problem which was faced earlier where the ping command yielded responces like request timed-out and destination host unreachable, as every router was aware of all other networks.

Experiment 4

1. Configure DHCP within a LAN & outside a LAN.

within a LAN:

i) Topology:



Procedure:

i) After creating the above topology, give IP address of server as 10.0.0.5

ii) Go to services → DHCP of services and add the pool with
poolname: serverpool1
start IP address: 10.0.0.10
Subnet Mask: 255.0.0.0

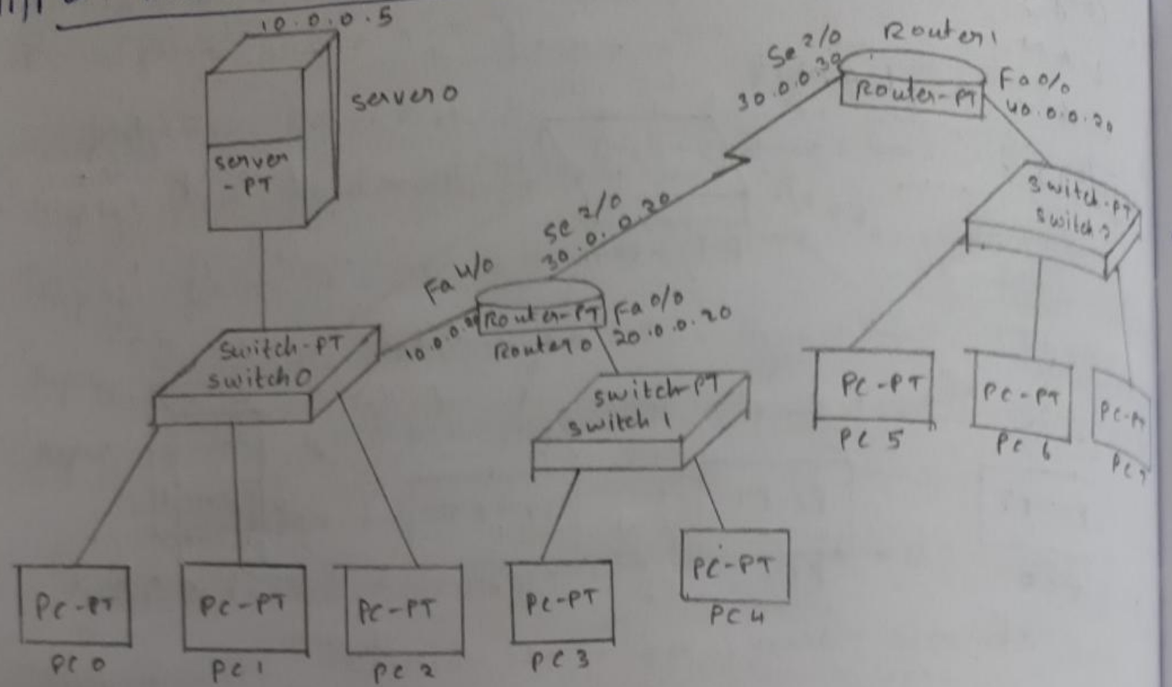
iii) Now go to the PCs and switch to DHCP in IP configuration.

iv) IP address will be assigned to the PCs by DHCP [PC0, PC1, PC2]

Output

For PC 0
DHCP request successful
IP address: 10.0.0.10
Subnet Mask: 255.0.0.0
Default Gateway: 0.0.0.0
DNS Server: 0.0.0.0

ii) For DHCP outside the lan



~~Proc~~

Procedure :

i) Repeat the procedure used for LAN (DHCP within a LAN)

ii) Now add a router, another set of PCs & Switch 1 (LAN 2)

iii) Configure the IP address of ports Fa 4/0 and Fa 0/0 of Router 0.

Router > enable
Router (config) # interface Fa 4/0
Router (config-if) # ip address 10.0.0.20 255.0.0.0
Router (config-if) # ip helper-address 10.0.0.5
Router (config-if) # no shutdown
Router (config-if) # exit

Do similarly for Fa 0/0.

Here ip helper address is the ip-address of server0

iv) Go to PC3 and PC4 and switch to DHCP in System → IP configuration

vi) Connect Router 1 to Router 0. Router 1 is connected to switch 2 and 3PCs [PC5, PC6, PC7].

vii) Configure IP address of Router 1 for interface serial 2/0 and Fa 0/0 with 30.0.0.30 and 40.0.0.20 respectively.

viii) Configure IP address of Router 0 for serial 2/0 as 30.0.0.20

viii) Perform static routing for Router 0 =>
Router (config) # ip route 40.0.0.0 255.0.0.0 30.0.0.30

For Router 1 =>
Router (config) # ip route 10.0.0.0 255.0.0.0 30.0.0.20

ix) Go to server and create 2 more server pools with different names.

	Default Gateway	DNS server	Start IP address	Subnet Mask
Serverpool 1	10.0.0.20	10.0.0.5	10.0.0.10	255.0.0.0
Serverpool 2	10.0.0.20	10.0.0.5	20.0.0.20	255.0.0.0
Serverpool 3	10.0.0.20	10.0.0.5	40.0.0.10	255.0.0.0

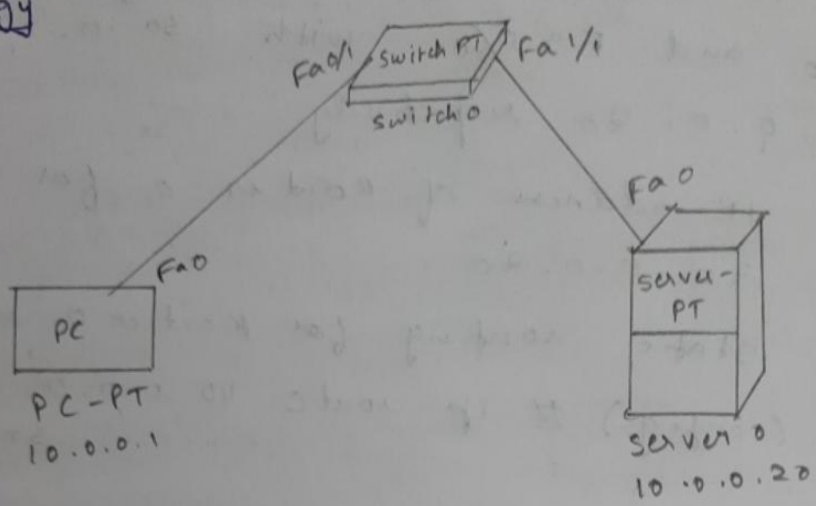
x) Go to PC5/PC6/PC7 & switch to DHCP in IP configuration. An IP address will be given to each of the PCs.

<u>Output</u> - for PC5
DHCP request successful
IP address : 40.0.0.11
Subnet Mask : 255.0.0.0
Default gateway : 10.0.0.20
DNS server : 10.0.0.5

Experiment 5

1) Configure web server, DNS within a LAN

Topology



Procedure:

→ Create a topology as shown above using a p
server and switch.

→ Set the ip addresses as 10.0.0.1 and 10.0.
for PC and server respectively.

→ In the server, under DNS service create new
Example.com website with url 10.0.0.2 an
add under HTTP, modify the index.html fil
and add name and USN as

<h1> S Charu Netra </h1>
<h1> 1BM21CS175 </h1>

→ In PC0, go to desktop ⇒ web browser an
type Example-com. You'll be able to see the
website with entered name and USN

Result

Web Browser

Url http://Example.com

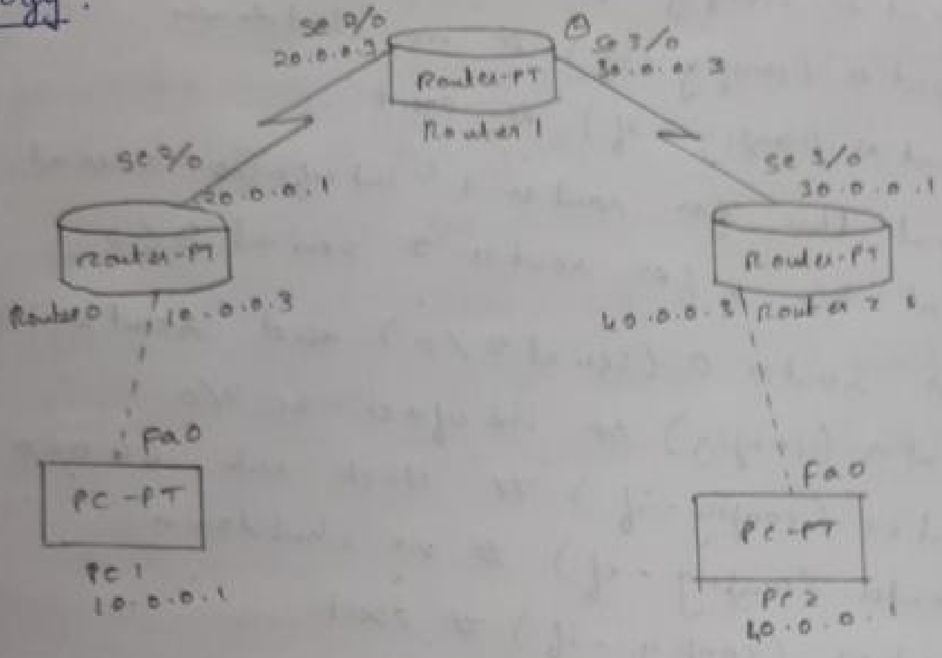
Cisco Packet Tracer

S Charu Netra

1BM21CS175

2) To configure RIP routing Protocol in Routers:

Topology:



Procedure:

→ Create a topology as shown above using 2PCs and 3 routers.

→ Configure the ip addresses of 2 PCs as 10.0.0.1 and 40.0.0.1 for PC1 and PC2 respectively. Set the gateways as 10.0.0.3 and 40.0.0.3.

→ ~~Plan the~~ Configure the routers

For router 0,

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) # ip interface serial 2/0

Router (config-if) # ip address 20.0.0.1 255.0.0.0

Router (config-if) # no shutdown

Similarly, configure the ports of router 1 and 2.

→ For router 0,

router (config) # interface serial 2/0

router (config-if) # encapsulation ppp

router (config-if) # no shutdown

router (config-if) # exit

router (config-if) #

Repeat this for router 1 interfaces serial 2/0, serial 3/0, for router 2 serial 3/0

→ For router 0 (serial 2/0) and router 1 (serial

router (config) # interface se 2/0

router (config-if) # clock rate 64000

router (config-if) # no shutdown

router (config-if) # exit

→ For all the 3 routers, repeat this step.

eg: For router 0

router > enable

router # configure terminal

router (config) # router rip

router (config-router) # network 10.0.0.0

router (config-router) # network 20.0.0.0

Similarly do for router 1 and 2.

Then router # show ip route

This will result in an output showing that every router knows all the 4 networks in the topology. Now, we can ping from pc1 to PC 2 successfully.

<u>Result</u>: Ping from 10.0.0.1

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 = 6ms 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, received = 4, lost = 0
Approximate round trip time in milliseconds
minimum = 2ms, maximum = 12ms, average = 6ms

Outcome:

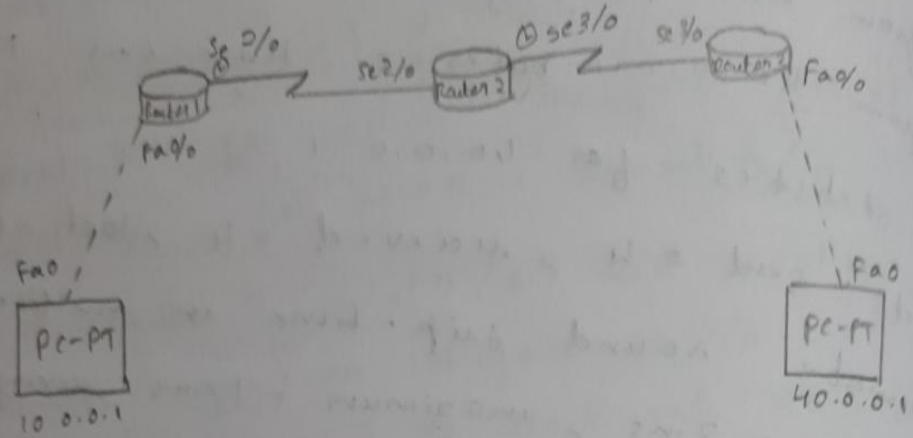
In DNS protocol configuring experiment, the procedure to configure a web server using DNS protocol was understood.

In RIP protocol configuring experiment, the procedure to configure routers using RIP protocol was demonstrated using the commands learnt.

Experiment 6

i) OSPF

Topology



Procedure: i) configure ip addresses to all interfaces.

For Router R1,

R1(config) # interface fa 0/0

R1(config-if) # ip address 10.0.0.2 255.0.0.0

R1(config-if)#no shutdown

R1(config) # interface se 2/0

R1(config-if) # ip address 20.0.0.1 . 255.0.0.0

R1(config-if)#encapsulation ppp

R1(config-if)#clock rate 64000

R1(config-if) # no shutdown

Similarly for R2 and R3.

~~ii) configure RIP to all routers.~~

~~For Router R1,~~

~~R1(config)# router rip~~

ii) Enable ip routing by configuring OSPF protocol.

For Router R1,

R1(config) # router ospf 1

R1(config-router)# router-id 1.1.1.1
R1(config-router)# network 10.0.0.0 0.255.255.255 area

R1(config-router)# network 20.0.0.0 0.255.255.255 area 1

Similarly give router id as 2.2.2.2, networks as 20.0.0.0 and 30.0.0.0 and area as 1 and 0 respectively (For router 2)

Give router id as 3.3.3.3 for router 3, networks as 30.0.0.0 and 40.0.0.0 and area as 0 and 2 respectively.

iv) Router# show ip route
Gateway of last resort is not set
C 10.0.0.0/8 is directly connected, Fa0/0
C 20.0.0.0/8 is directly connected, Se2/0
O IA 40.0.0.0/8 [110/129] via 20.0.0.2, 00:04:23, serial 2/0
O IA 30.0.0.0/8 [110/128] via 20.0.0.2, 00:07:29, serial 2/0

Configure loopback addresses to routers.
R1(config-if)# interface loopback 0
R1(config-if)# ip add 172.16.1.252 255.255.0.0
R1(config-if)# no shutdown

R2(config-if)# interface loopback 0
R2(config-if)# ip add 172.16.1.253 255.255.0.0
R2(config-if)# no shutdown

R3(config-if)# interface loopback 0
R3(config-if)# ip add 172.16.1.254 255.255.0.0
R3(config-if)# no shutdown

v) check routing table of R3.

R3# show ip route

Gateway of last resort is not set

O IA 20.0.0.0/8 [110/128] via 30.0.0.1
00:18:58, Se 2/0

C 40.0.0.0/8 is directly connected, Fe 2/0

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

For R3 to know about area 3 we should create virtual link b/w R1 and R2

vi) For Router R1,

R1(config)# router ospf 1

R1(config-router)# area 1 virtual-link 2.2.2.2

~~R1(config-router)#~~ R1(config-router)# exit

For Router R2,

R2(config-router)# area 1 virtual-link 1.1.1.1

R2(config-router)# exit

vii) R2 and R3 know about area 3 now.

R3# show ip route

Gateway of last resort is not set

O IA 20.0.0.0/8 [110/128] via 30.0.0.1, 00:01:58
Se 2/0

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

O IA 10.0.0.0/8 [110/129] via 30.0.0.1, 00:01:58
Se 2/0

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

viii) Output
ping 40.0.0.1 (from 10.0.0.1)

pinging 40.0.0.1 with 32 bytes of data:

Reply from 40.0.0.1 : Bytes=32 time=8ms
TTL=125

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

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

reply from 40.0.0.1 : Bytes = 32 time = 4ms TTL = 125

Ping statistics for 40.0.0.1

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

Approximate round trip time in milli seconds:

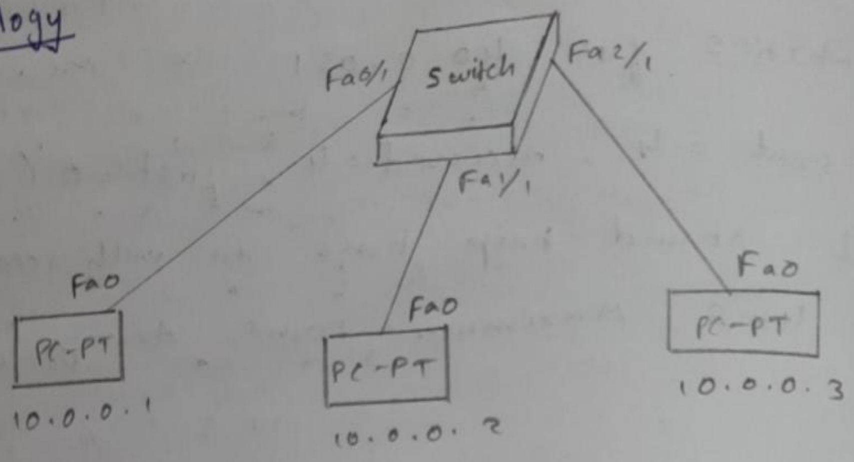
Minimum = 4ms, Maximum = 10ms, Average = 7ms

Outcome:

OSPF is an application layer protocol which is used to find the best route for packet transfer and involves creating a virtual link.

ii) ARP

Topology



Procedure:

→ Configure IP address

→ (CMD) for PC0

→ arp -a

→ ping 10.0.0.2

→ arp -a

→ arp -d

→ Output

* arp -a

No arp entries found

* ping 10.0.0.2

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

* arp -a

Internet address	physical address	Type
10.0.0.2	0.002.16C5.98a0	Dynamic

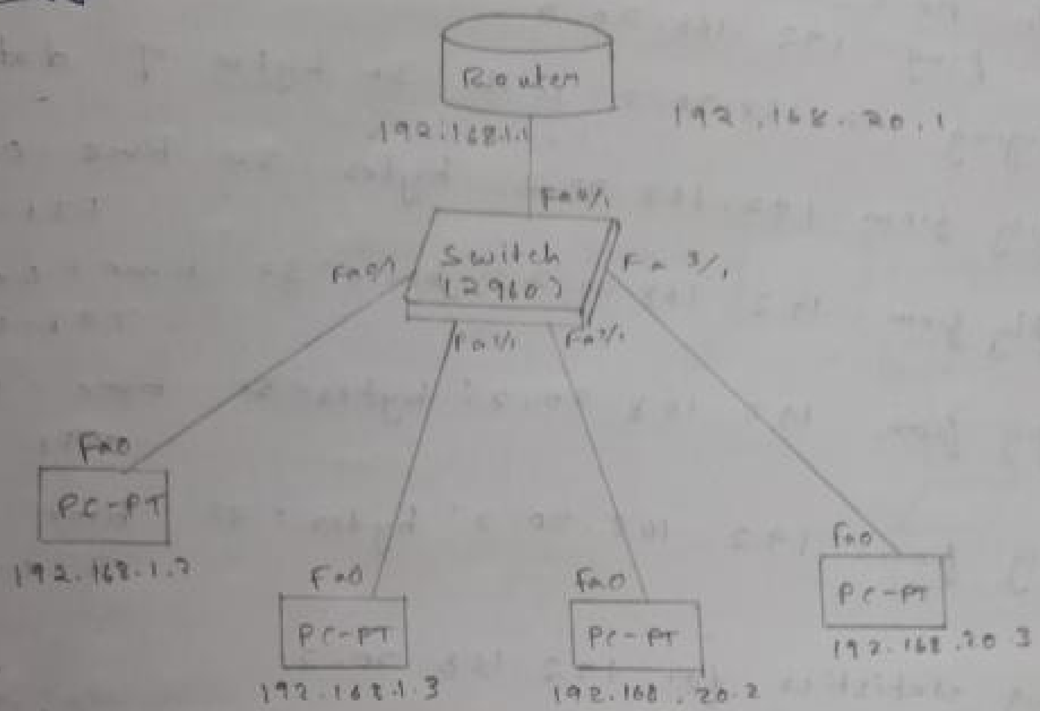
Outcome:

To learn about how the Address Resolution Protocol maps the IP address to the MAC address

Experiment 7

VLAN

Topology



Procedure:

-> In switch => VLAN Database => configure VLAN number and VLAN name

FastEthernet 0/5 -> In dropdown, select Trunk

FastEthernet 0/4 => configure VLAN

FastEthernet 0/3 => configure VLAN

-> In Router,

=> VLAN Database => configure VLAN number and VLAN name

CLI of router:

config t
interface fa 0/0.1
encapsulation dot 1q 20
ip address 192.168.20.1 255.255.255.0
no shutdown
exit

-> Ping the device.

~~Result~~

Output

In PC0,

ping 192.168.20.2

pinging 192.168.20.2 with 30 bytes of data.

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

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

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

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

ping statistics for 192.168.20.2

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

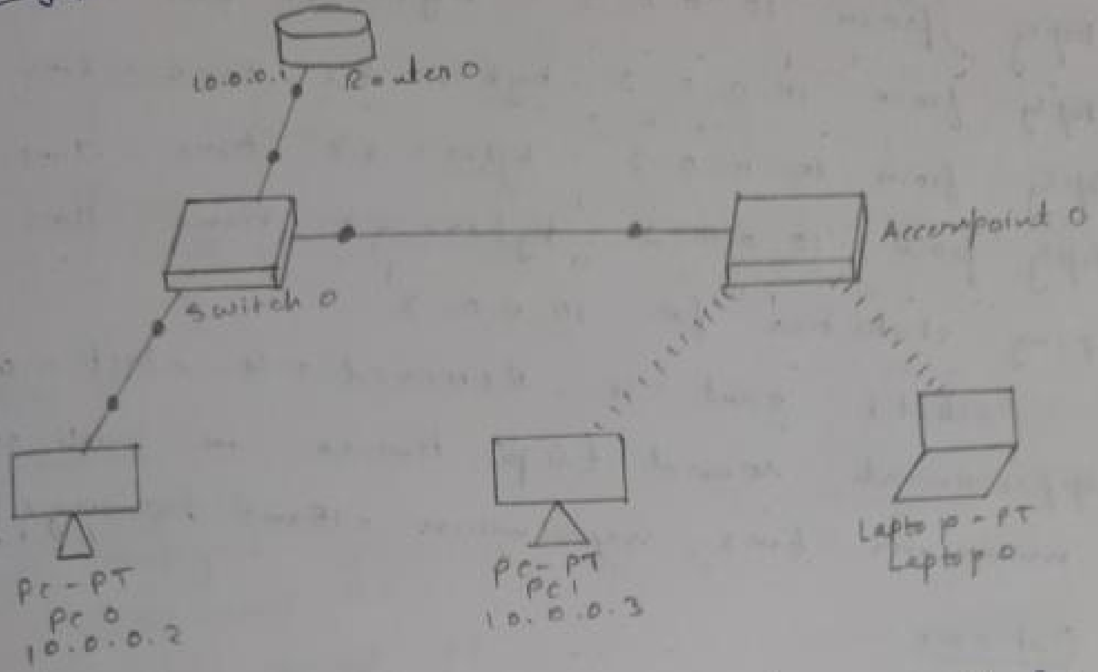
Outcome:

A virtual LAN is a virtualised connection that connects multiple devices and network modes from multiple LANs in 1 logical network.

Experiment 8

a) WLAN

Topology



Procedure:

→ Set the ip address and gateway of PCs.

→ For access point 0,

⇒ In port 1,

SSID ⇒ WLAN

WEN key ⇒ 1234 567890

→ For PC1 the existing PT-HOST-NM-Switch gf. Drag the component list and drag IAM to the empty port. Then switch on WMP300N to the device.

In wireless 0,

SSID ⇒ WLAN

WEN KEY ⇒ 1234567890

IP address ⇒ 10.0.0.3

Gateway ⇒ 10.0.0.1

→ Repeat the same for laptop.

→ Ping the device.

Result

Ping 10.0.0.3

pinging 10.0.0.3 with 32 bytes of data:

Reply from 10.0.0.3 : bytes=32 time=15ms TTL=127

Reply from 10.0.0.3 : bytes=32 time=6ms TTL=127

Reply from 10.0.0.3 : bytes=32 time=9ms TTL=127

Reply from 10.0.0.3 : bytes=32 time=11ms TTL=127

ping statistics for 10.0.0.3

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

Approximate round trip times in milli-seconds:

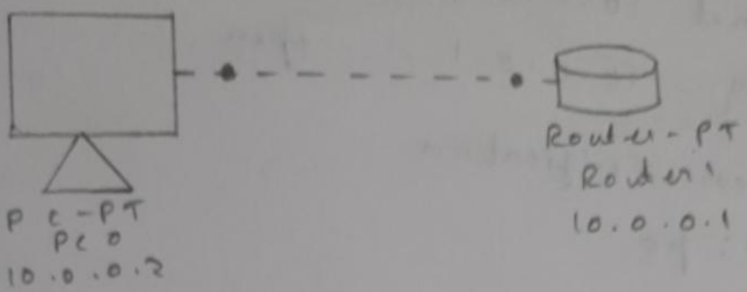
minimum = 6ms, maximum = 15ms, average = 10ms

Outcome:

WLAN is a group of drivers that form a network based on radio transmissions rather than wired connections and how we can configure it in a topology.

5) Telnet

Topology



Procedure:

In router 1,

```
enable
config-t
hostname r1
enable secret p1
interface fa0/0
ip address 10.0.0.1   255.0.0.0
no shutdown
line vty 0 5
login
password p0
exit
exit
wr
```

In PC0,

```
ping 10.0.0.1
```

Output

pinging 10.0.0.1 with 32 bytes of data:

Reply from 10.0.0.1 : bytes = 32 time = 17ms TTL = 255

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

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

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

ping statistics for 10.0.0.1;

packets : sent = 4 recieved = 4 , lost = 0 (0% loss)
Approximate round trip times in milli-seconds:
minimum = 0ms , maximum = 17ms , average = 4ms

PC > telnet 10.0.0.1
Trying 10.0.0.1 .... open

User Access verification

Password : p0

r1 > enable

password : p1

r1 # show ip route

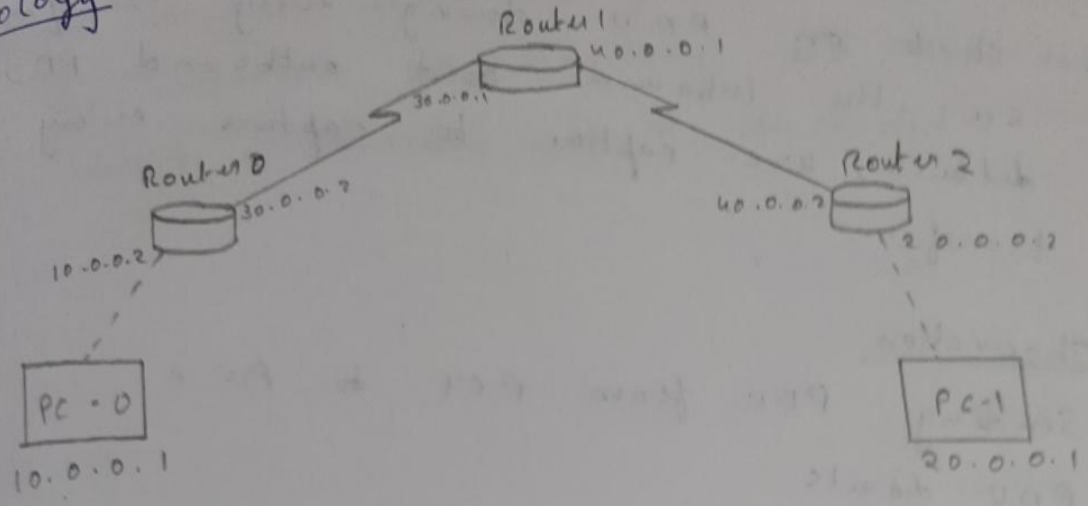
C 10.0.0.0/8 is directly connected, Fa0/0

Outcome

To understand how TELNET works in a LAN and how password is set and access is done.

c) TTL

Topology



Procedure:

1. Create a topology as shown above.
2. Configure the IP addresses as 10.0.0.1 & 20.0.0.1 for PC0 and PC1 respectively.
3. Configure the IP address for routers using static / default routing.

Router 0:

router > enable
router # config t
router (config) # interface fa0/0
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 se2/0
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 30.0.0.2
router (config) # exit

Similarly configure for router 1 and 2.

4. In simulation mode, send a simple PDU from one PC to another.

5. Check on PDU during every transfer to see the inbound and outbound PDU details, use capture to capture every transfer.

Observation

1) Sending PDU from PC1 to PC2

PDU details

At device 10.0.0.1

TTL = 255

At Router 0

TTL = 254

At Router 1

TTL = 253

At Router 2

TTL = 252

2) Sending PDU ~~from~~ back from PC2 to PC1

At device 20.0.0.1

TTL = 128

At router-2

TTL = 127

At Router-1

TTL = 126

At Router-0

TTL = 125

At PC-1

TTL = 125

Outcome :

TTL value in IP packet tells a network router when the packets have been in the network for too long and once this time limit is reached, it is discarded.

Cycle - 2 Programs

Q. Write a program for congestion control using Leaky Bucket.

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

Output

Enter bucket size, outgoing rate & no. of inputs

20 10 2

Enter the incoming packet size = 30
incoming packet size = 30
~~dropped buffer size~~
Dropped 10 no. of packets
Bucket buffer size 0 out of 20
After outgoing 10 packets left out of 20 in buffer

Enter incoming packet size = 10
incoming packet size = 10
Bucket buffer size 10 out of 20
After outgoing 10 packets left out of 20 in buffer.

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

ClientTCP.py

```
from socket import *
serverName ='127.0.0.1'
serverPort = 12000
clientSocket = socket(AF_INET, SOCK-STREAM)
clientSocket.connect(serverName, serverPort)
sentence = input("Enter file name")
clientSocket.send(sentence.encode())
print("In From Server")
print(filecontents)
clientSocket.close()
```

ServerTCP.py

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

Output

Server -

Server is ready to recieve

Client -

Enter file name : ServerTCP.py

From server :

- - - -

- - . .

code in ServerTCP.py

- - . .

- - - .

Server -

Sent contents of ServerTCP.py

The server is ready to recieve

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

Client UDP.py

```
from socket import *
serverName = "127.0.0.1"
serverPort = 12000
clientSocket = socket(AF_INET, SOCK_DGRAM)
sentence = input("Enter file name")
clientSocket.sendto(bytes(sentence, "utf-8"), (
                        serverName, serverPort))

filecontents, serverAddress = clientSocket.recvfrom(2048)
print("Reply from server")
~~#~~ clientSocket.close()
    clientSocket.close()
```

ServerUDP.py

```
from socket import *
serverPort = 12000
serverSocket = socket(AF_INET, SOCK_DGRAM)
serverSocket.bind(("127.0.0.1", serverPort))
print("The server is ready to 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("Sent contents of", end='')
    print(sentence)
```

file.close()

Output

Server:

The server is ready to recieve →

Sent contents of ServerUDP

Client

Enter file name: ServerUDP.py

Reply from Server:

- - - -

- - -

code from ServerUDP.py

- - - - -

Q. Implement CRC - CCITT (16-bits).

```
#include <stdio.h>
 char m[50], g[50], r[50], q[50], temp[50];

 int main()
{  printf("Enter frame bits");
   while((ch==getc(stdin))!='\n')
    m[i++]=ch;
    n=i;
   for(i=0;i<16;i++)
   m[n++]='0';
 printf("Message after appending 16 zeroes :%s");
  for(i=0;i<=16;i++)
   g[i]='0';
  g[0]=g[4]=g[11]=g[16]='1';
  printf("generator %s \n",g);
   crc(n);
 printf("quotient %s",q);
  caltrans(n);
 printf("~~Tran~~ Enter transmitted frame");
  scanf("%s",m);
 printf("CRC checking");
  crc(n);
 printf("last remainder %s",r);
 if(~~flag~~ r[i]!='0')
  flag=1;
 if(flag==1)
 printf("Error during transmission");
 else
 printf("Received frame is correct");
```

```
}
void crc(int n)
{ for(i=0;i<n;i++)
   temp[i]=m[i];
  for(i=0;i<n-16;i++)
  { if(r[0]=='1')
     { q[i]='1'
       calram();
     }
    else
    {
    q[i]='0';
    shiftl();
  }
void calram()
{ for(i=1;i<=16;i++)
    r[i-1]=((int)temp[i]-48)^((int)g[i]-48)
                                          +48;
}
void shiftl()
{   for(i=1;i<=16;i++)
    r[i-1]=r[i];
 }
void caltrans(int n)
{ 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 : 1011 1011 0001 0110 1011

Enter transmitted frame

1011 1011 0001 0110 1011

last remainder 0000 0000 0000 0000

Recieved frame is correct