VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"JnanaSangama", Belgaum -590014, Karnataka.

LAB REPORT
on

COMPUTER NETWORKS

Submitted by

MUKESH KUMAR N V (1BM20CS088)

in partial fulfillment for the award of the degree of
BACHELOR OF ENGINEERING
in
COMPUTER SCIENCE AND ENGINEERING

B.M.S. COLLEGE OF ENGINEERING
(Autonomous Institution under VTU)
BENGALURU-560019
October-2022 to Feb-2023

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

CERTIFICATE

This is to certify that the Lab work entitled "LAB COURSE **COMPUTER NETWORKS**" carried out by **MUKESH KUMAR N V (1BM20CS088),** who is bonafide student of **B. M. S. College of Engineering.** It is in partial fulfillment for the award of **Bachelor of Engineering in Computer Science and Engineering** of the Visvesvaraya Technological University, Belgaum during the year 2022. The Lab report has been approved as it satisfies the academic requirements in respect of a **Computer Networks - (20CS5PCCON)** work prescribed for the said degree.

Lohith JJ
Assistant Professor
Department of CSE
BMSCE, Bengaluru

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

`

Index

Cycle 1

Experiment No 1

Expt-1.

Aim: To create a topology & simulate sending a simple PDU from source to destination using hub & switch as connecting devices.

Topology: Star topology

Procedure: → End devices are connected to the hub

→ The hubs are interconnected via a switch.

→ IP addresses of the end devices are set.

→ Connections between all of them are checked if it is working.

→ They are checked by pinging a message between 2 end devices.

→ Once verified, a simple PDU is sent/transmitted b/w a source & a destination.

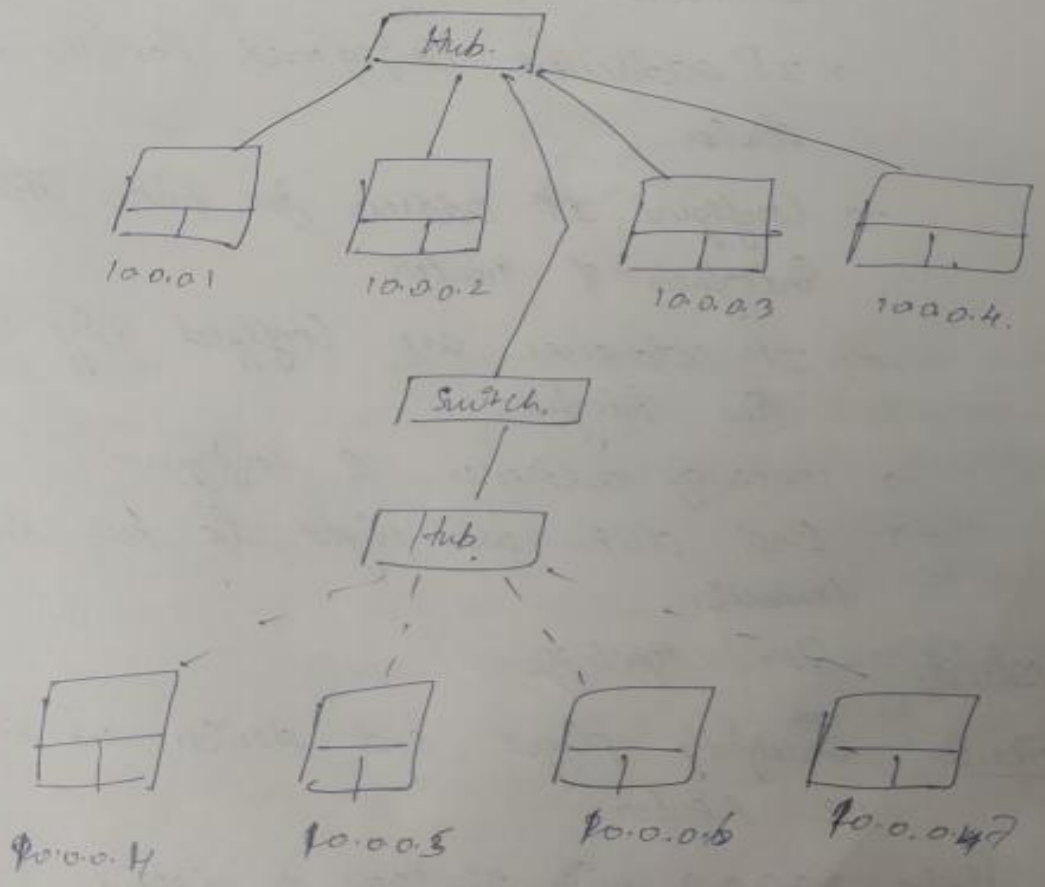
Result: The transmitting of PDU's were successful between the source & destination.

Observations: → Hubs broadcast a PDU to all the connected devices in a network, when a source transmit a PDU to hub.

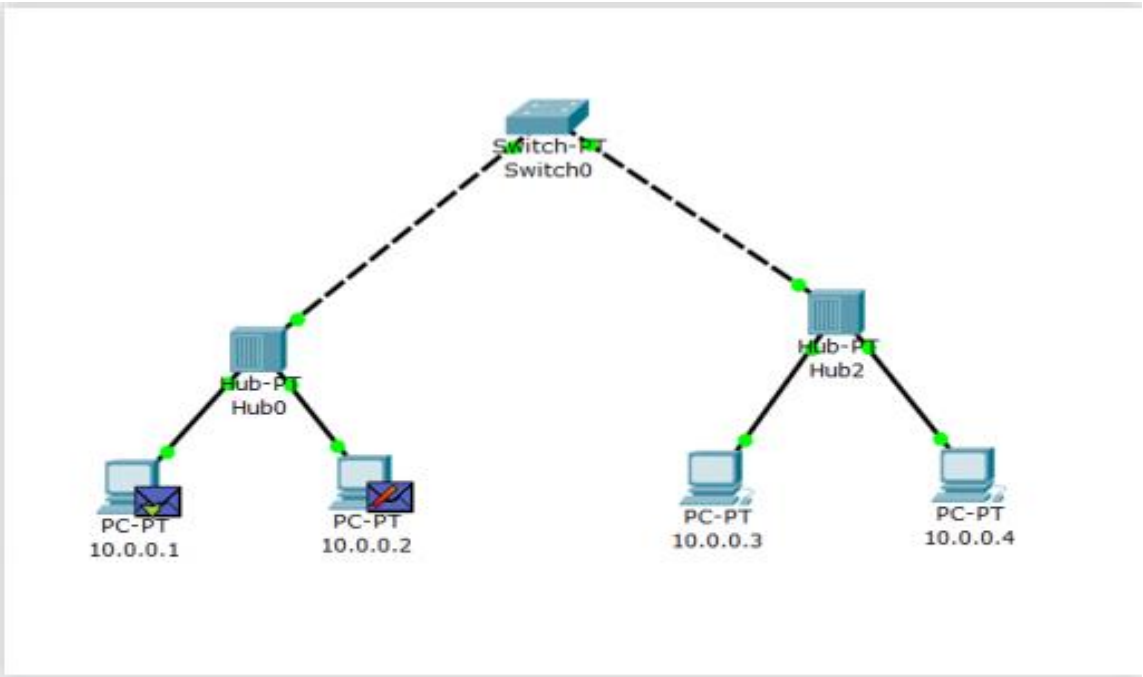
→ Switches initially broadcast a PDU to the remaining connected devices. The device destination, replies back with a message to confirm the destination MAC address

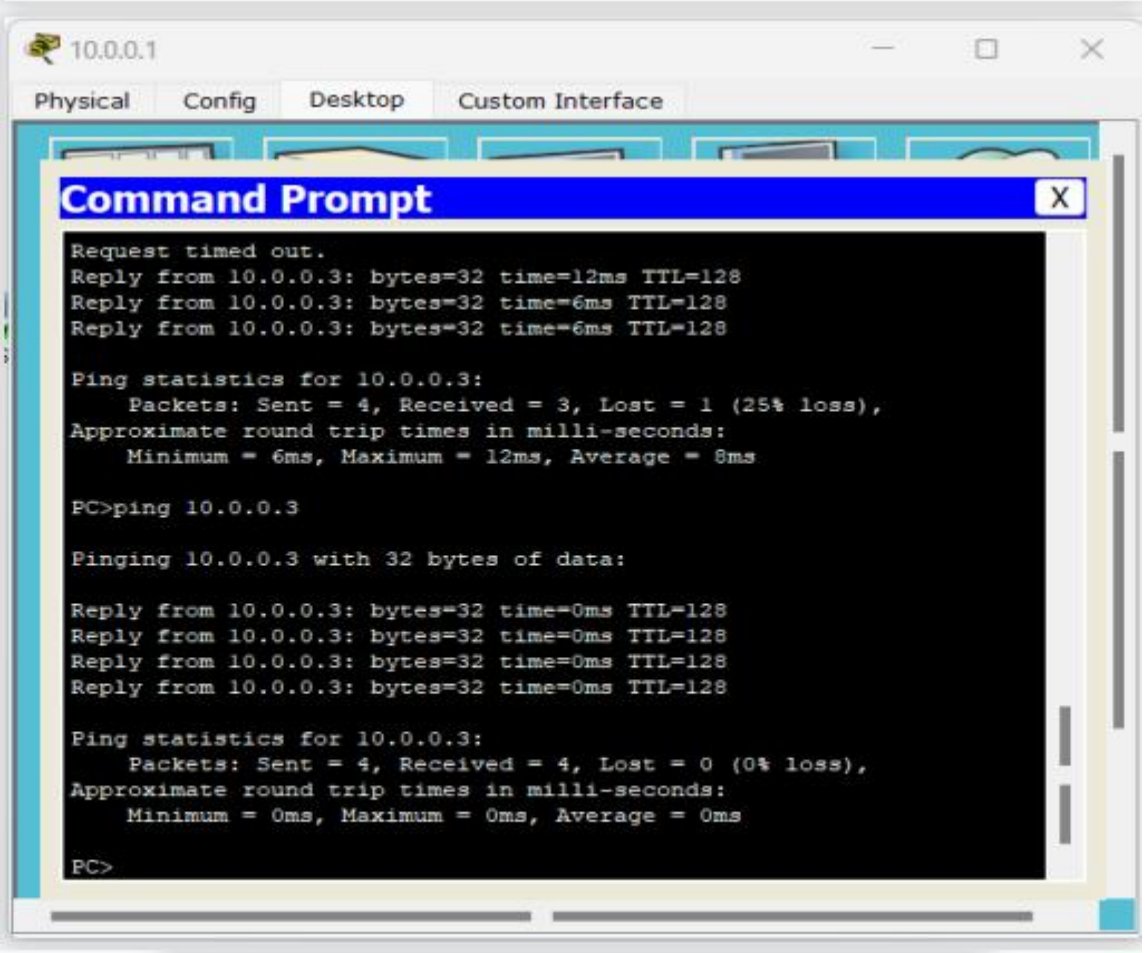
Once the connection is established, there is unicasting between the source and destination via the switch.

→ If a receiver host isn't connected to a internetwork a message cannot be pinged & hence response will be timed out



Snapshot of Output





Experiment No 2

Aim :- Configuring IP address to routers in packet tracer. Explore ping responses destination unreachable, reply, request, timed out.

Procedure: → End devices are PC's which are connected to a Router.

→ IP addresses configured for the end devices.

→ Configure IP address for both the interfaces of router.

→ IP addresses are configured using command line interface.

→ Gateway addresses is configured.

→ End devices are pinged to test the connection.

Topology: Star topology

Result: Successfully pinged end devices ~~us~~ through a router.

Pinging 20.0.0.2 with 32 bytes of data:

Reply from 20.0.0.2 : bytes = 32 time = 0ms TTL = 255

Reply from 20.0.0.2 : bytes = 32 time = 0ms TTL = 255

Reply from 20.0.0.2 : bytes = 32 time = 0ms TTL = 255

Reply from 20.0.0.2 : bytes = 32 time = [illegible]

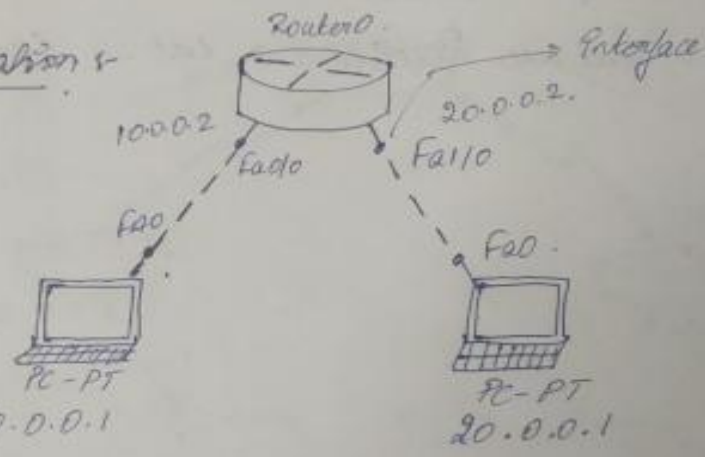
Ping statistics for 20.0.0.2:

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

Approximate round trip times in milli-seconds:

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

Observation :-



Observation.

Successfull

1) → when we configure both end devices and router with appropriate IP addresses and by configuring subnet mask of interface of router, as 255.0.0.0 and gateway of PC0 is set as 10.0.0.2. which is of Fa0/0 interface. ~~and~~ & followed by same for PC1, we could successfully ping the end device.

Timed out:

→ If IP address of end devices or gateway is not configured properly, then we get request timed out.

Pinging 20.0.0.3 with 32 bytes of data:

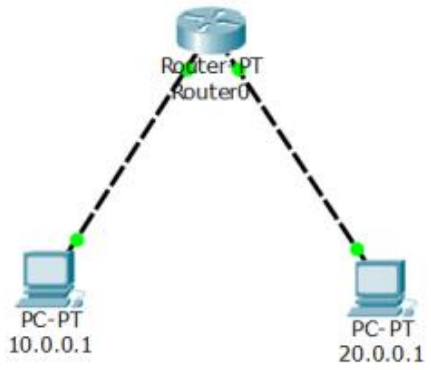
Request timed out

Ping statistics for 20.00.3 :

Packets : Sent = 4, Received = 0, Lost = 4 (100% Loss)

Neelima
17/10/2022

Snapshot of Output




```
PC>ping 20.0.0.1

Pinging 20.0.0.1 with 32 bytes of data:

Reply from 10.0.0.2: Destination host unreachable.
Reply from 10.0.0.2: Destination host unreachable.
Reply from 10.0.0.2: Destination host unreachable.
Reply from 10.0.0.2: Destination host unreachable.

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

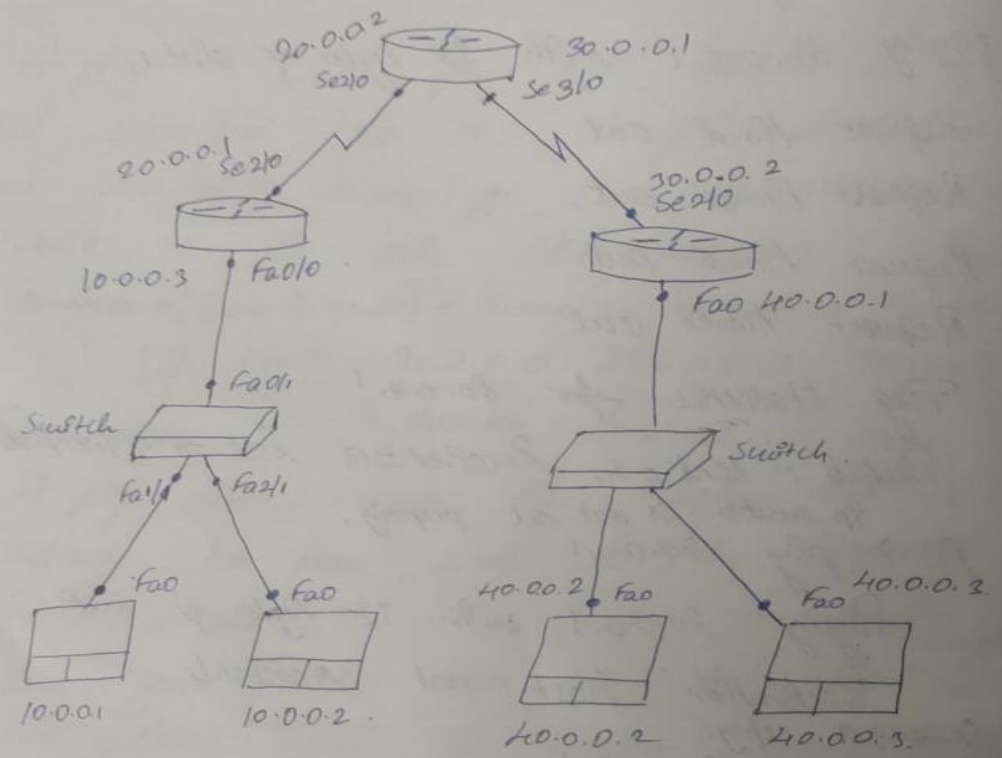
```
PC>ping 20.0.0.1

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),
```

```
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=6ms TTL=127
Reply from 20.0.0.1: bytes=32 time=0ms TTL=127
Reply from 20.0.0.1: bytes=32 time=1ms 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),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 6ms, Average = 1ms

PC>
```

Experiment No 3

Aim: Configuring Default route to router.

Topology: Star topology.



Procedure:

→ End devices are connected to the switch. Switches are then connected to routers.

→ Configure end devices ~~& eno~~ and interfaces of routers

→ Configure IP address between routers as serial interface.

→ To configure IP address following commands are executed

Config terminal
Interface Se2/0
ip address 10.0.0.2 255.0.0.0
no shutdown
exit

→ Configure gateway address of the end device with connected interface of the router.

→ Inorder to establish default route across the routers, ip route is configured using the command

ip route 0.0.0.0 0.0.0.0 20.0.0.2

Observation

→ A ping doesnt cross the interface until a gateway has been sent to the connected interface/router.

→ Once gateway has been set, the ping will not ~~wor~~ cross over to another router as the router are not connected to other networks and they won't know which route to take are where the next hop of the signal is to be done

→ Default route is configured between the routers, where ip route and subnet mask is not specified only via interface of connected router.

→ Later pinged connections between all the routers and end device.

Result:

A successfull ping message has been sent over the end devices that are connected to different router/ network.

Output:

i) Gateway not configured.

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

Ping statistics for 20.0.0.1

Packets: sent = 4, Received = 0, lost = 4 (100% loss)

ii) IP route not configured

PC > ping 30.0.0.1

Pinging 30.0.0.1 with 32 bytes of data

Destination host unreachable

iii) Successfull reply

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

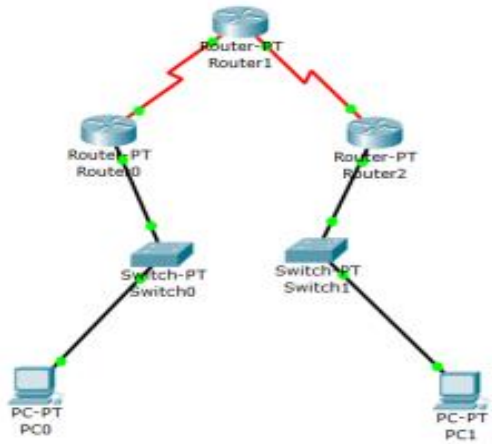
Reply from 40.0.0.1, bytes = 32, time = 8ms, TTL = 125

Ping statistics from 40.0.0.1

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

Network Layer

Snapshot of Output




```
PC>ping 30.0.0.1

Pinging 30.0.0.1 with 32 bytes of data:

Request timed out.
Request timed out.
Request timed out.
Request timed out.

Ping statistics for 30.0.0.1:
    Packets: Sent = 4, Received = 0, Lost = 4 (100% loss),

PC>
```

```
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 40.0.0.1: Destination host unreachable.
Request timed out.
Reply from 40.0.0.1: Destination host unreachable.
Reply from 40.0.0.1: Destination host unreachable.

Ping statistics for 10.0.0.1:
    Packets: Sent = 4, Received = 0, Lost = 4 (100% loss),

PC>
```

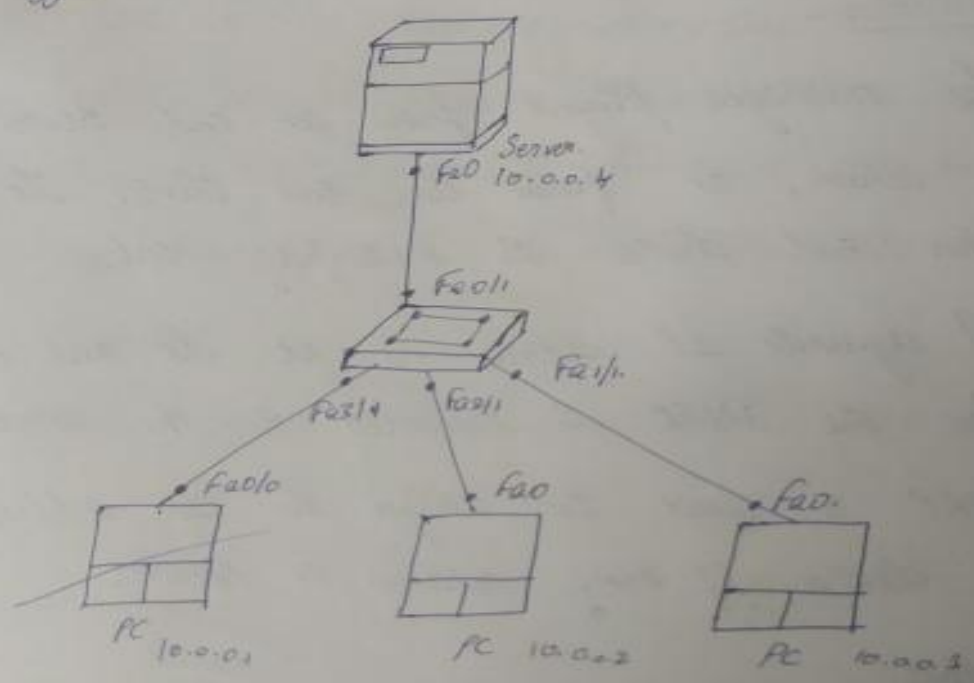
```
PC>ping 40.0.0.2

Pinging 40.0.0.2 with 32 bytes of data:

Reply from 40.0.0.2: bytes=32 time=14ms TTL=125
Reply from 40.0.0.2: bytes=32 time=11ms TTL=125
Reply from 40.0.0.2: bytes=32 time=11ms TTL=125
Reply from 40.0.0.2: bytes=32 time=11ms 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 = 11ms, Maximum = 14ms, Average = 11ms

PC>
```

Experiment No 4

LAB-6

Aim: Configure DHCP Server.

Topology:



Procedure:

→ Construct the following topology. That is Server is connected to a switch, and further is connected to PC's (end devices)

→ Configuring IP address of the server as 10.0.0.4,

→ Add/save IP address of the end devices on. Start IP address of DHCP, After switching on.

→ ~~Later~~ the DHCP server.

→ After assigning addresses, change the IP address of the end devices from static to DHCP.

→ Ping the IP addresses across all the end devices.

Observation:-

→ The messages pinged from an end device to the server, or from an end device to another end device is successfully sent.

→ A dynamic IP address is set to end device when the DHCP is initiated on the server.

→ RARP is used to assign the IP address of a device of MAC address is known.

Output:

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=1ms TTL=128

Reply from 10.0.0.3 bytes=32 time=0ms 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 = 4ms, Average = 1ms.

2) SERVER > ping 10.0.0.1

Pinging 10.0.0.1 with 32 bytes of data:

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

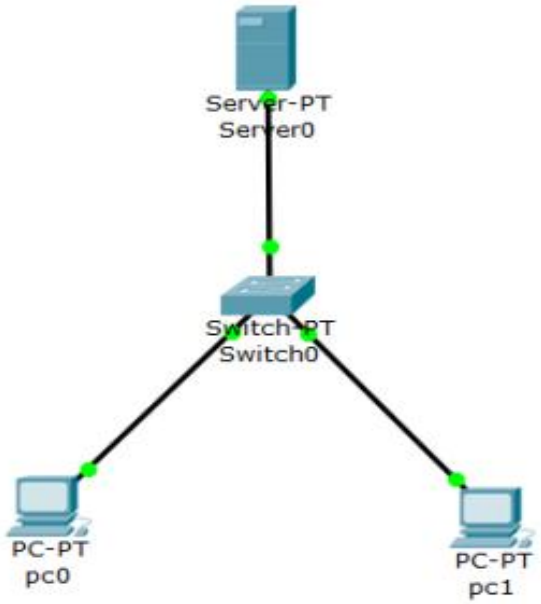
Reply from 10.0.0.1 : bytes = 32 time = 0ms TTL = 128.

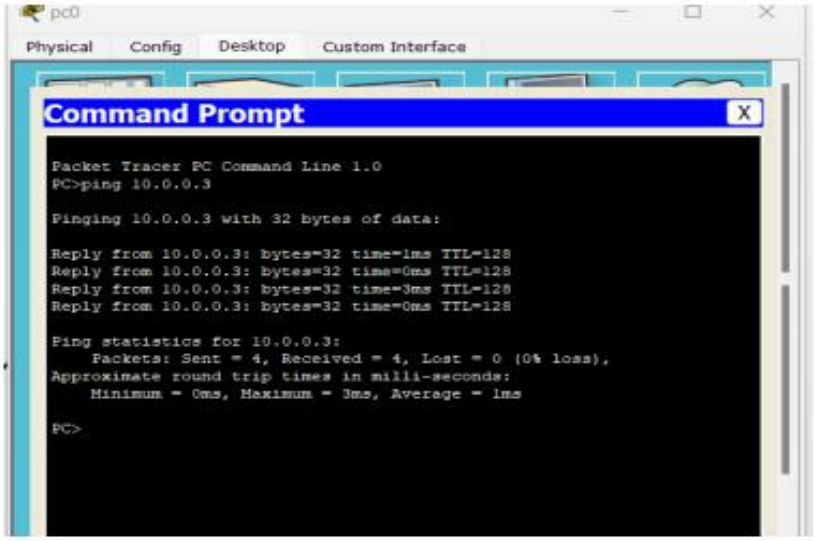
Ping statistics

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

15/12

Snapshot of Output



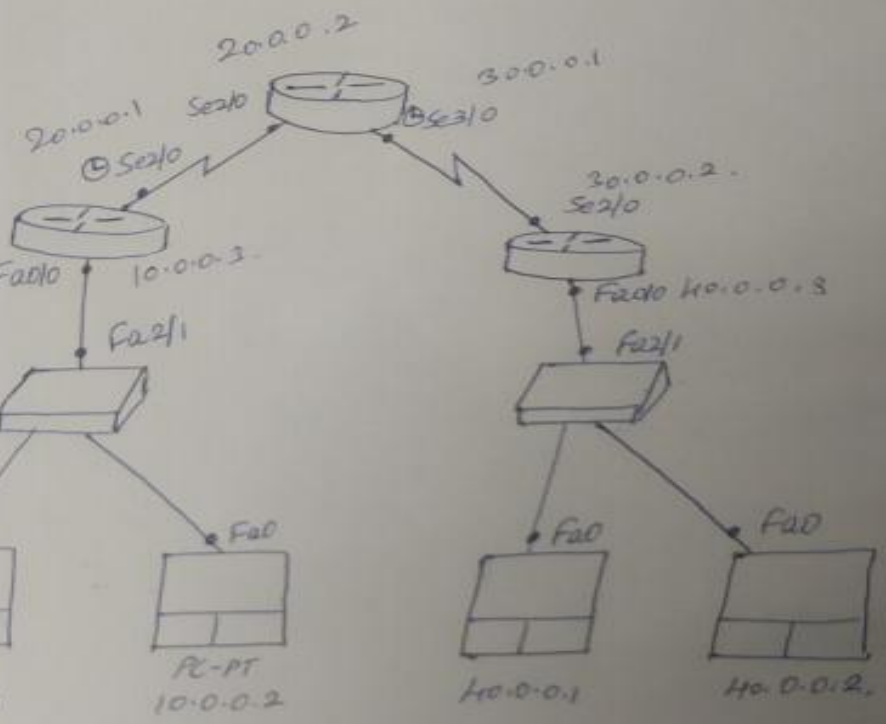


Experiment No 5

LAB-5

Aim:- Configuring RIP routing protocol on routers

Topology:



Procedure

→ Three router are connected to 2 switches which are then connected to ~~routerend~~ device.

→ Configure end device and interface of router.

→ Configure IP address following commands are executed

enable

Config terminal

Interface se2/0

IP address 10.0.0.2 255.0.0.0

no shutdown

Cont.

→ Configure gateway address of end device with connected interface of router.

→ Inorder to configure RIP protocol among the router. Serial DCE connection, we run following commands.

#router rip

#network 20.0.0.0

#network 30.0.0.0.

→ For every serial DCE connection, to configure RIP, with defined clock rate.

encapsulation PPP

clock rate 64,000.

→ A ping has been sent from one end device to other network end device.

Observation:

> ping 40.0.0.1

Reply from 40.0.0.1 with 32 bytes of data

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

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

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

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

Ping statistics for 10.0.0.1:

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

Approximate round trip in milliseconds

Minimum = 2ms, Maximum = 40ms.

Average = 10ms.

Since RIP protocol has been established, IP route does not have to be set for each router.

Before RIP was set. ~~for each~~

Ping 10.0.0.1 → 40.0.0.1 : Destination host unreachable.

Before RIP

Ping 10.x → 20.x Request timed out.

Only on correctly configuring gateways and protocol, to reply received property.

Result:

(Routing Information Protocol) RIP is established in network correctly

Note:

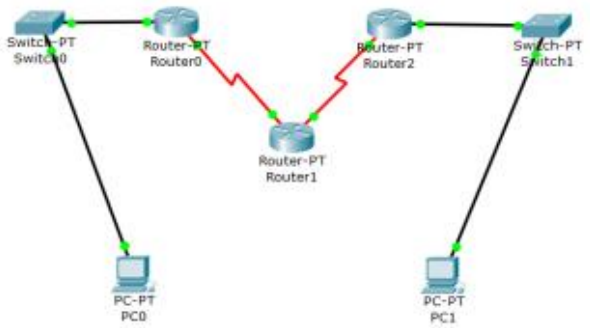
Even on proper connection and configuration. The first packet of first inter network ping to time out as switches have not learnt network yet.

RIP- Routing Information Protocol.-

Is dynamic routing protocol that uses hop count as a meter to find best path. between source and destination.

8/12

Snapshot of Output




```
PC>ping 40.0.0.2

Pinging 40.0.0.2 with 32 bytes of data:

Reply from 10.0.0.2: Destination host unreachable.
Reply from 10.0.0.2: Destination host unreachable.
Reply from 10.0.0.2: Destination host unreachable.
Request timed out.

Ping statistics for 40.0.0.2:
    Packets: Sent = 4, Received = 0, Lost = 4 (100% loss),
```

```
PC>ping 20.0.0.2

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, Lost = 4 (100% loss),
```

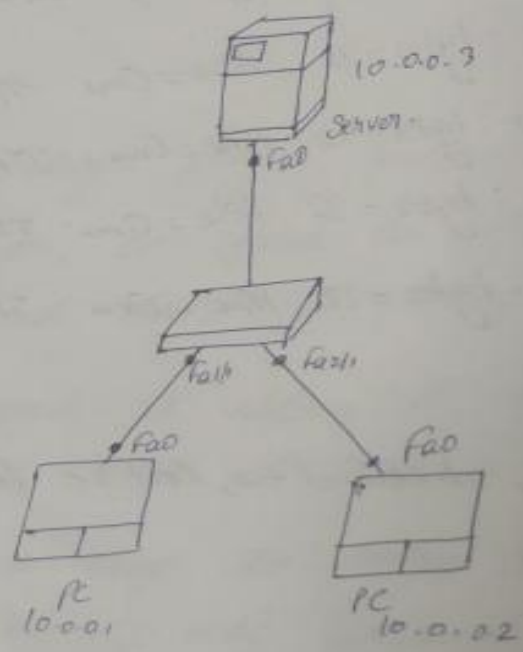
```
PC>ping 40.0.0.2

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=12ms TTL=125
Reply from 40.0.0.2: bytes=32 time=2ms TTL=125
Reply from 40.0.0.2: bytes=32 time=12ms 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 = 12ms, Average = 7ms
```

Experiment No 6

LAB-7.

Aim: Configuring webserver and DNS server.

Topology:



Procedure:

→ Construct the following topology, where Server is connected to end devices through switches.

→ Configure IP address of the server and the end devices.

→ Set the HTTP and DNS server to on state.

→ Set the Server Domain Name and address of H 9f as same as server. and save the following

→ From one of the end devices, check if the Webserver is reachable from the end devices by entering URL.

Observation:

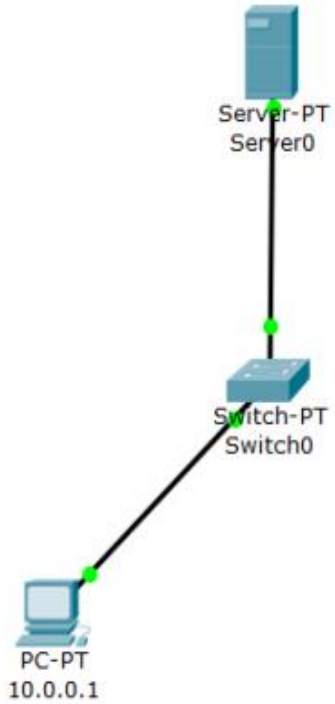
→ Web browser module is opened in the end device and the set domain name (www.dmsce.com) is entered.

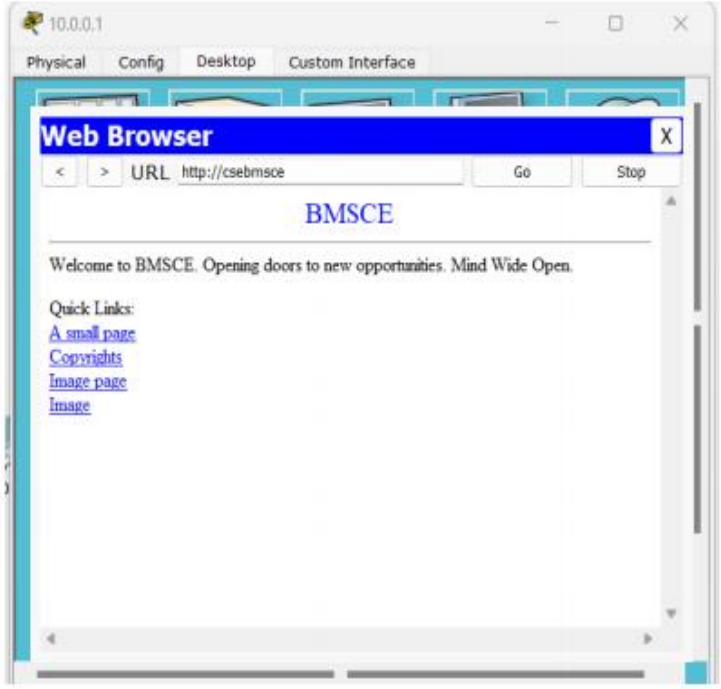
→ If the system/server hasn't been configured properly, i.e., set DNS server and default gateway, the 'Host Unresolved' is shown.

→ If configured properly, the page of Cisco Packet Tracer is opened.

15/12

Snapshot of Output





Cycle 2

Experiment No 7

1) Write a program, for error detection using CRC 16-bit.

Ans

```
import java.util.*;
class CRC-algo {
    public static void main(String args[]){
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter no.");
          int n = sc.nextInt();
        int[] data = new int[n+16];
        int[] rem = new int[n];
    for(int i=0; i<n+16; i++){
        data[i] = (i>n)?0 : sc.nextInt();
          if (i<n)
              rem[i] = data[i]
        }
    for (int i= 0; i<n; i++){
        if (data[i]==1)
            continue;
        for (int j=0; j<17; j++){
            data[i+j] = data[i+j] ^ divisor[j];
        }
    }
```

```
for (int i=0; i< rem.length;
        data[i] = rem[i];
System.out.println();
for (int i=0; i< data.length; i++)
        System.out.println (data[i] + " ");

data[10] =1;
for (int i=0; i<n; i++) {
        if (data[i]==1)
                continue;
        for (int j= 0; j<17; j++)
                data[i+j] = data[i+j] ^ divisor[j];

    }
System.out.println();
for (int i=0; i< data.length; i++)
        System.out.print (data[i] + " ");

}
}
```

Output.

Enter message to be parced : 3.

1 1

1 1 1 0 1 1 0 0 0 0 0 1 1 0 0 0 1 1

0 0 1 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0.

Output – Screen shot

```
Enter the length of Data Frame:
3
Enter the Message:
1 0 1
Data to be transmitted:
1 0 1 0 1 0 1 0 0 0 0 1 0 1 0 0 1 0 1
Enter the Reveived Data:
1 0 1 0 1 0 0 1 0 0 1 0 1 0 0 0 0 0 0
error in data
PS C:\Users\mknv7\OneDrive\Desktop\5th Sem\Computer Network>
```

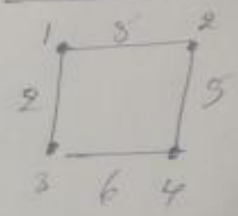
Experiment No 8

Write a program for distance vector routing algorithm to find suitable path for transmission

```
class DVR {
    int graph[];
    public static void main(String args[]){
        System.out.println("Enter number of edges");
          e = sc.nextInt();
        System.out.println("Enter number of vertices");
          v = sc.nextInt();
        for (int i=0; i<v; i++)
            for (int j=0; j<v; j++){
                if (i==j)
                    graph[i][j] = 0;
                else
                    graph[i][j] = 9999;
        }

    static void update_table (int source){
        for (int i=0; i<v; i++){
            if (graph[source][i] != 9999){
                int dist = graph[source][i];
                for (int j=0; j<v; j++){
                    inter_dist = rt[i][j];
                    if (via[i][j] = source)
                        inter_dist = 9999;
                    if (dist + inter_dist < rt[source][j]){
```

```
            rt[source][j] = dist + rtes[dist];
            via[source][j] = i;
        }
    }

static void update_table(){
    int k = 0;
    for (int i=0; i<4*v; i++){
        update_table(k);
        k++;
        if (k==v)
            k=0
    }
void print_table(){
    for (int i=0; i<v; i++){
        for (int j=0; j<v; j++){
            System.out.printf("DIST + rt[i][j]+"  ");
        }
        System.out.println();
    }
}
```

Output:



Routing table are

0	5	2	8
5	0	7	5
2	7	0	6
8	5	6	6

Output – Screen shot

```
Please enter the number of Vertices:
5
Please enter the number of Edges:
6
Please enter data for Edge 1:
Source: 1
Destination: 2
Cost: 3
Please enter data for Edge 2:
Source: 2
Destination: 4
Cost: 6
Please enter data for Edge 3:
Source: 3
Destination: 5
Cost: 4
Please enter data for Edge 4:
Source: 1
Destination: 4
Cost: 8
Please enter data for Edge 5:
Source: 2
Destination: 5
Cost: 5
Please enter data for Edge 6:
Source: 4
```

```
Source: 2
Destination: 5
Cost: 5
Please enter data for Edge 6:
Source: 4
Destination: 5
Cost: 2

The initial Routing Tables are:
Dist: 0    Dist: 3    Dist: 12    Dist: 8    Dist: 8
Dist: 3    Dist: 0    Dist: 9    Dist: 6    Dist: 5
Dist: 12    Dist: 9    Dist: 0    Dist: 6    Dist: 4
Dist: 8    Dist: 6    Dist: 6    Dist: 0    Dist: 2
Dist: 8    Dist: 5    Dist: 4    Dist: 2    Dist: 0

Please enter the Source Node for the edge whose cost has changed: 2
Please enter the Destination Node for the edge whose cost has changed: 4
Please enter the new cost: 7

The new Routing Tables are:
Dist: 0    Dist: 3    Dist: 12    Dist: 8    Dist: 8
Dist: 3    Dist: 0    Dist: 9    Dist: 7    Dist: 5
Dist: 12    Dist: 9    Dist: 0    Dist: 6    Dist: 4
Dist: 8    Dist: 7    Dist: 6    Dist: 0    Dist: 2
Dist: 8    Dist: 5    Dist: 4    Dist: 2    Dist: 0
```

Experiment No 9

Implement dijkstra's algorithm for a given topology. Shortest path for a given topology.

```
#Include (stdio.h)
Void dijkstra();
Int c[10][10]; n, ch[9];
Void main(){
  Int i,j;
  for (i=1; i<=n; i++) {
     for (j=1; j<=n; j++) {
        scanf("%d", &c[i][j]);
        if c[i][j]==0
           c[i][j] = 9999;
     }
  }
Void dijkstra(){
  for (j=1; j<=n; j++)
     dist[j] = c[ch][j];
  for (j=1; j<=n; j++)
     vis[j] = 0;
  while (count!=n){
     min = 9999.
     for (j=1; j<=n; j++)
        if (dist[j] < min && vis[j] = 1)
           min = dist[j];
```

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

for (j=1; j<=n; j++)
    printf("%d->%d=%d", src, j, dist[j]);
```

<u>Output:</u>

```
Enter number of vertices: 3
Enter cost matrix:
0 4 1
4 0 2
1 2 0

Enter source node: 1
1->1=0
1->2=3
1->3=1.
```

Output – Screen shot

```
Vertex          Distance from Source
0               0
1               4
2               12
3               19
4               21
5               11
6               9
7               8
8               14
PS C:\Users\mknv7\OneDrive\Desktop\5th Sem\Computer Network>
```

Experiment No 10

LAB-9

9) Leaky - Bucket

```
import java.util.*;
class leaky-bucket {
    public static void main (String args[]) {
        Scanner sc = new Scanner (System.in);
        System.out.println ("Enter bucket capacity");
        int capacity = sc.nextInt();
        System.out.println ("Enter output rate");
        int rate = sc.nextInt();
        int curr = 0;
while (true) {
    System.out.println ("Enter the input rate");
    int input = sc.nextInt();
    if (curr + input > capacity)
        System.out.println ("Bucket Overflow");
    else {
        curr += input;
        curr -= rate;
        curr = Math.max (0, curr);
        System.out.println ("Bucket Capacity = "+curr
    }
```

```
System.out.println("Do you want to continue, 2 to exit, 1 to continue");
    int choose = sc.nextInt();
    if (choose == 2)
        break;
    }
}
}
```

Output

```
Enter bucket Capacity.
500
Enter Output rate.
200
Enter Input rate
300
Bucket Capacity is : 100.
Do you want to continue, 2 to exit, 1 to continue
2.
```

5/1/2023

Output – Screen shot

```
Enter the bucket capacity :
500
Enter output rate
200
Enter the input rate :
300
Bucket Capacity is 100
Do you want to continue, 2 to exit ,1 to continue
1
Enter the input rate :
300
Bucket Capacity is 200
Do you want to continue, 2 to exit ,1 to continue
1
Enter the input rate :
400
Bucket Overflow
Do you want to continue, 2 to exit ,1 to continue
```

Experiment No 11

(11) Using TCP/IP Socket, write a client-server program to make client sending the filename and the server send back the contents of the requested file if present.

Server TCP.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()
```

clientTCP.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("From server: \n")
print(fileContents)
clientSocket.close()
```

Output:

Serverside

The server is ready to receive

Sent contents of dummy.txt.

The server is ready to receive.

Client

Enter file name: dummy.txt

From server:

Received dummy.txt file from server

Output – Screen shot

```
The server is ready to receive

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

```
Enter file name: serverTCP.py

From Server:
 connectionSocket, addr = serverSocket.accept()
 sentence = connectionSocket.recv(1024).decode()

 file=open(sentence,"r")
 l=file.read(1024)

 connectionSocket.send(l.encode())
 print ('\nSent contents of ' + sentence)
 file.close()
 connectionSocket.close()
```

Experiment No 12

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.

a. Server_UDP.py

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

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))
```

```
file contents, serverAddress = clientSocket.recvfrom(2048)
print('Reply from server')
print(file contents.decode("utf-8"))
clientSocket.close()
clientSocket.close
```

Output.

Server side

Server is ready to receive

Sent contents of demo.txt

Client side

Enter file name : demo.txt

Reply from Server:

Hi hello this message is transported via UDP.

Output – Screen shot

```
C:\Windows\System32\cmd.e X

C:\Users\mknv7\OneDrive\Desktop\5th Sem\Computer Network\Lab\UDP>python -u s
erverUDP.py
The server is ready to receive

Sent contents of  dummy.txt
```

```
C:\Windows\System32\cmd.e X

C:\Users\mknv7\OneDrive\Desktop\5th Sem\Computer Network\Lab\UDP>python -u c
lientUDP.py

Enter file name: dummy.txt

Reply from Server:

Hi hello this message has been transported across client to server using UDP
.

C:\Users\mknv7\OneDrive\Desktop\5th Sem\Computer Network\Lab\UDP>
```