VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"JnanaSangama", Belgaum -590014, Karnataka.

LAB REPORT
on

COMPUTER NETWORKS

Submitted by

OMAR ABDULLA SHERIEF (1BM20CS209)

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

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

B. M. S. College of Engineering,

Bull Temple Road, Bangalore 560019

(Affiliated To Visvesvaraya Technological University, Belgaum)

Department of Computer Science and Engineering

CERTIFICATE

This is to certify that the Lab work entitled "**COMPUTER NETWORKS**" carried out by **Omar Abdulla Sherief(1BM20CS209),** who is bonafide student of **B. M. S. College of Engineering.** It is in partial fulfillment for the award of **Bachelor of Engineering in Computer Science and Engineering** of the Visvesvaraya Technological University, Belgaum during the year 2022. The Lab report has been approved as it satisfies the academic requirements in respect of a **COMPUTER NETWORKS - (20CS5PCCON)** work prescribed for the said degree.

Dr.Shymala G
Assistant Professor
Department of CSE
BMSCE, Bengaluru

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

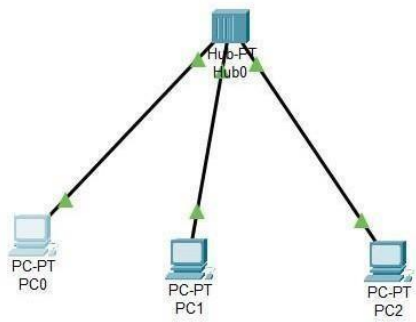
`

Index

Cycle-1 Experiment No 1 Aim of the program

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

Hub Topology



Procedure

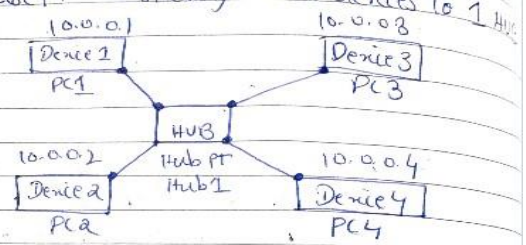
7/11/22 Experiment : 1

Aim :-

Q. Create a topology consisting of two or more devices using hub.

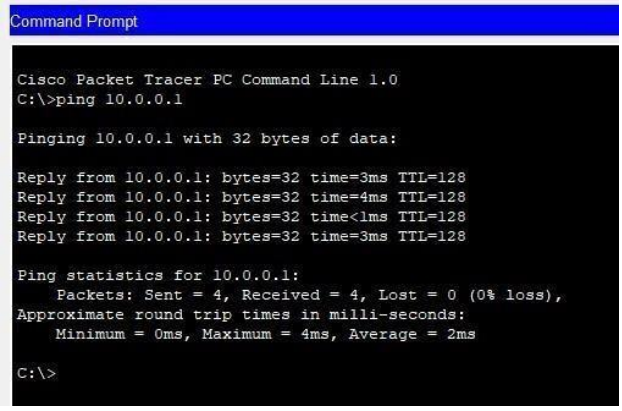
Procedure:

Ans: Case 1: Connecting All devices to 1 Hub

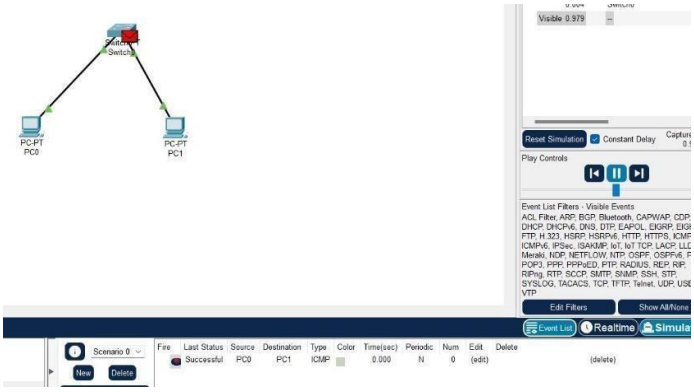


* HUB is a dumb device i.e it is not intellegent. So whatever data is sent from a sender to hub then the hub broadcasts the data to the rest of the devices.
* Time it takes to send from sender to receiver is about 0.002 seconds.
* A default type connection is used
* Device and Hub is connected using a copper straight through cable.
* It will broadcast even to devices which do not require the data

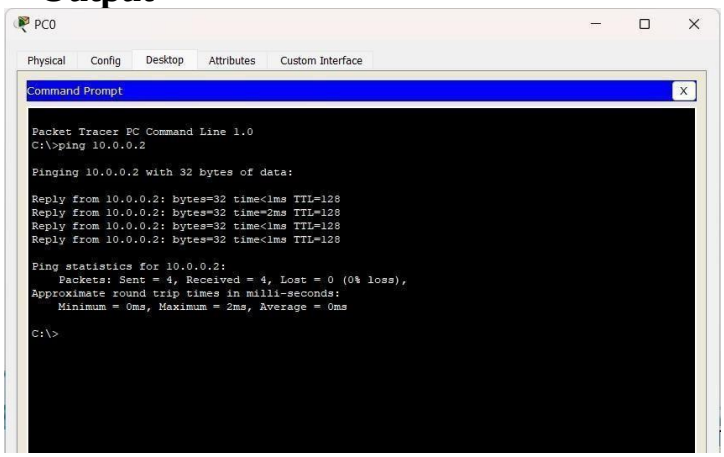
Output



Switch Topology



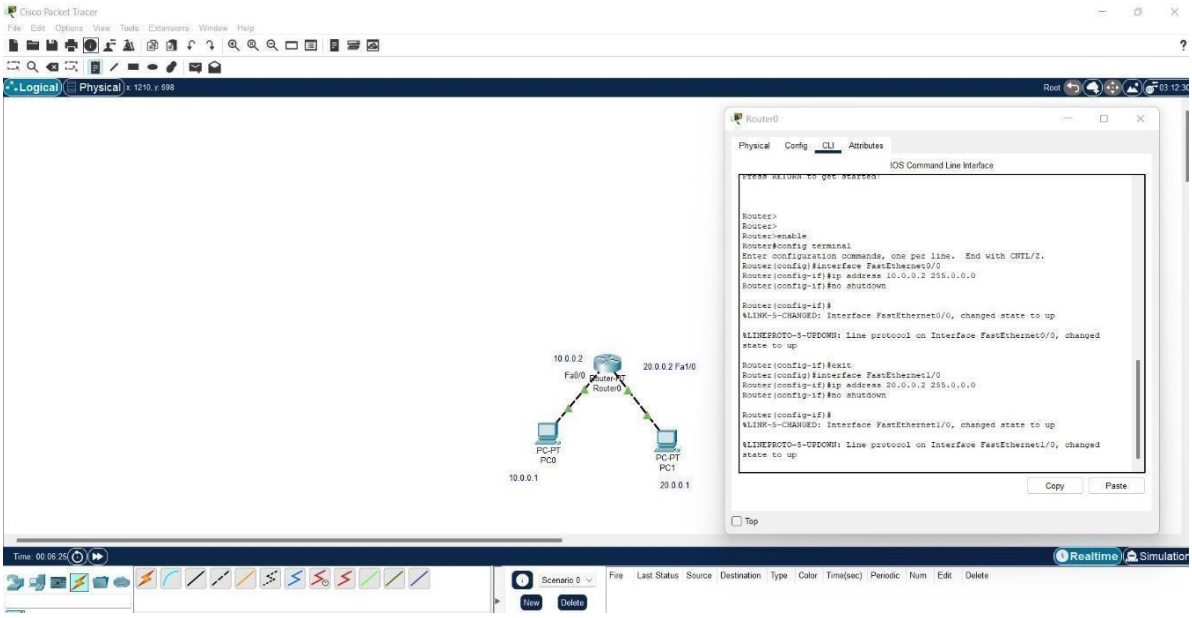
Output


PC0
Physical
Config
Desktop
Attributes
Custom Interface
Command Prompt
X
Packet Tracer PC Command Line 1.0
C:\>ping 10.0.0.2
Pinging 10.0.0.2 with 32 bytes of data:
Reply from 10.0.0.2: bytes=32 time<1ms TTL=128
Reply from 10.0.0.2: bytes=32 time=2ms TTL=128
Reply from 10.0.0.2: bytes=32 time<1ms TTL=128
Reply from 10.0.0.2: bytes=32 time<1ms TTL=128
Ping statistics for 10.0.0.2:
Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
Minimum = 0ms, Maximum = 2ms, Average = 0ms
C:\>

Experiment No 2 Aim of the program

Configuring IP address to Routers in Packet Tracer. Exploring the following messages: Ping Responses, Destination unreachable, Request timed out, Reply.

Topology



Procedure

14/11/22 LAB PROGRAM : 2

AIM :-

Q Configuring IP address to Routers in Packet Tracer. Explore the following messages: Ping responses, Destination unreachable, Request timed out, Reply.

Procedure:

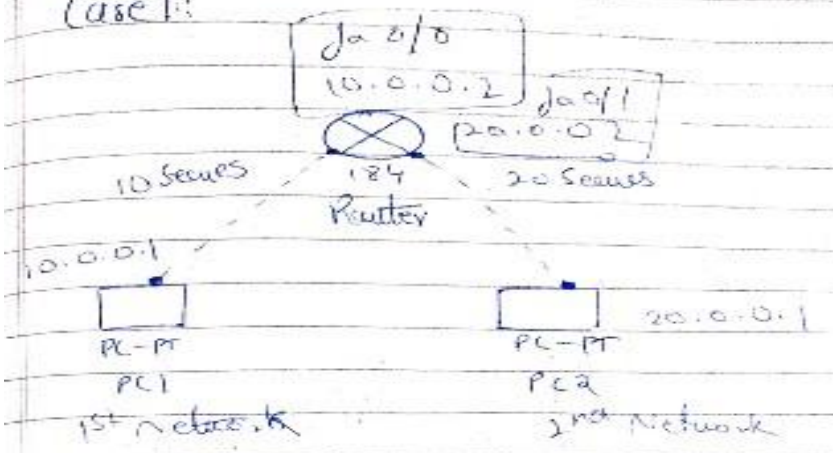
Continue with configuration dialog [Yes/no]

Router > enable
Router # Config terminal
Enter configuration mode
Router (config) # interface fa0/0
Router (config-if) # IP address
10.0.0.2 255.0.0.0
Router
Router (config-if) # no shutdown

up [Enter]

Router (config-if) # exit

Case 1:



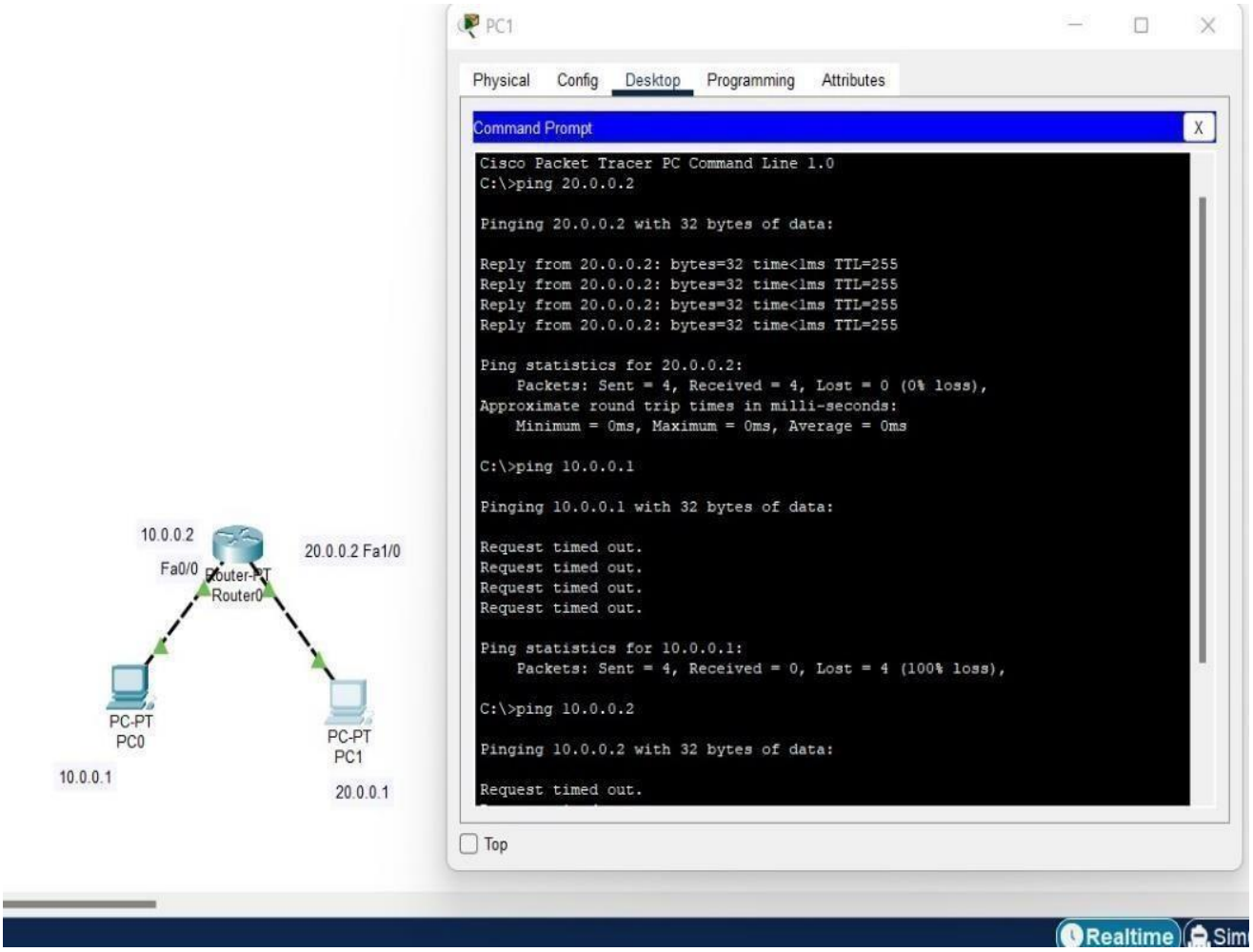
From 1st PC to router
PC> ping 10.0.0.2
Pinging 10.0.0.2 with 3
Reply from 10.0.0.2: bytes 32 TTL=255
Reply from 10.0.0.2: bytes 32 TTL:255
Reply from 10.0.0.2: bytes 32 TTL:255
Reply from 10.0.0.2: bytes 32 TTL:255

PC 1 → Router is one configuration
PC 2 → Router is second configuration

For PC1 gateway will the IP address of router 10.0.0.2

For PC2 gateway will be IP address of router of 20 series i.e 20.0.0.2

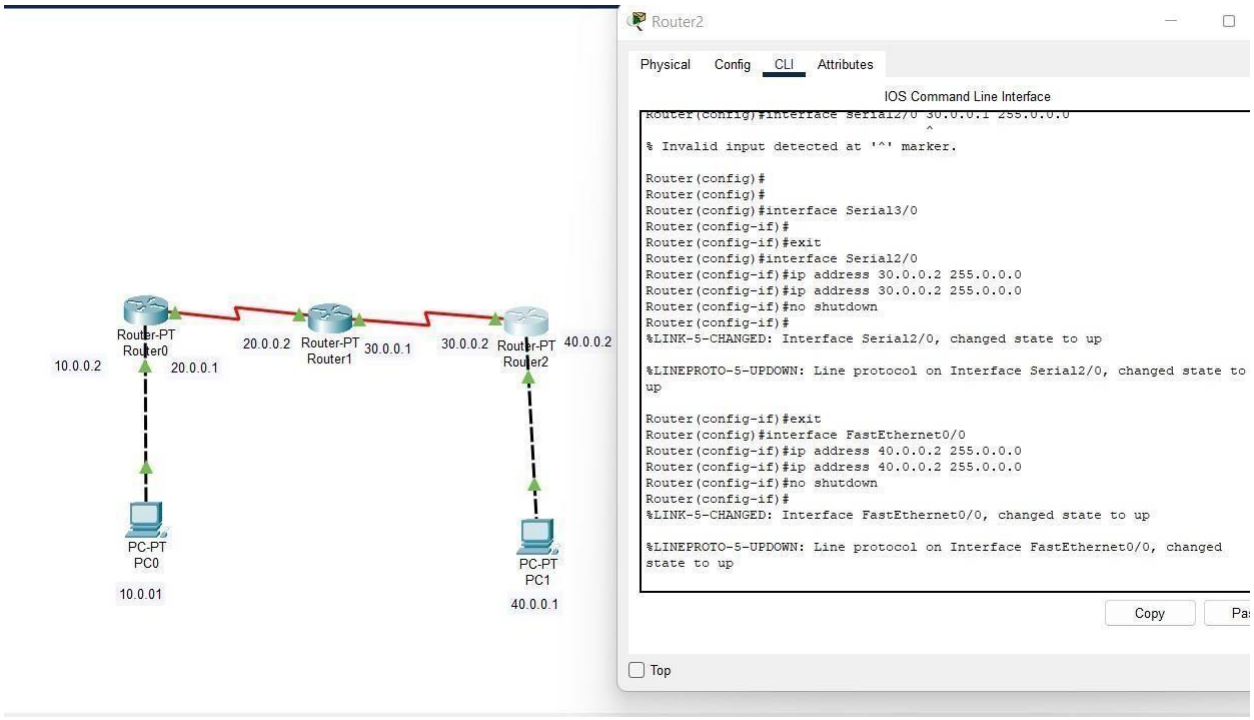
Output


10.0.0.2
Fa0/0
Router-PT
Router0
20.0.0.2 Fa1/0
PC-PT
PC0
10.0.0.1
PC-PT
PC1
20.0.0.1
PC1
Physical
Config
Desktop
Programming
Attributes
Command Prompt
Cisco Packet Tracer PC Command Line 1.0
C:\>ping 20.0.0.2
Pinging 20.0.0.2 with 32 bytes of data:
Reply from 20.0.0.2: bytes=32 time<1ms TTL=255
Reply from 20.0.0.2: bytes=32 time<1ms TTL=255
Reply from 20.0.0.2: bytes=32 time<1ms TTL=255
Reply from 20.0.0.2: bytes=32 time<1ms TTL=255
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
C:\>ping 10.0.0.1
Pinging 10.0.0.1 with 32 bytes of data:
Request timed out.
Request timed out.
Request timed out.
Request timed out.
Ping statistics for 10.0.0.1:
Packets: Sent = 4, Received = 0, Lost = 4 (100% loss),
C:\>ping 10.0.0.2
Pinging 10.0.0.2 with 32 bytes of data:
Request timed out.
Top
Realtime

Experiment No 3 Aim of the program

Configuring default route to the Router

Topology



Procedure

(iii) For Router 3

R3> IP router 0.0.0.0 0.0.0.0 30.0.0.1
R3> IP Router 0.0.0.0 0.0.0.0 20.0.0.1

outcome

PC> Ping 10.0.0.1

Pinging 10.0.0.1 with 32 bytes of data

Reply from 10.0.0.1: bytes = 32, time = 1ms TTL = 125
Reply from 10.0.0.1 bytes = 32, time = 20ms TTL = 125
Reply from 10.0.0.1: bytes = 32, time = 20ms TTL = 125
Reply from 10.0.0.1: bytes = 32, time = 20ms TTL = 125

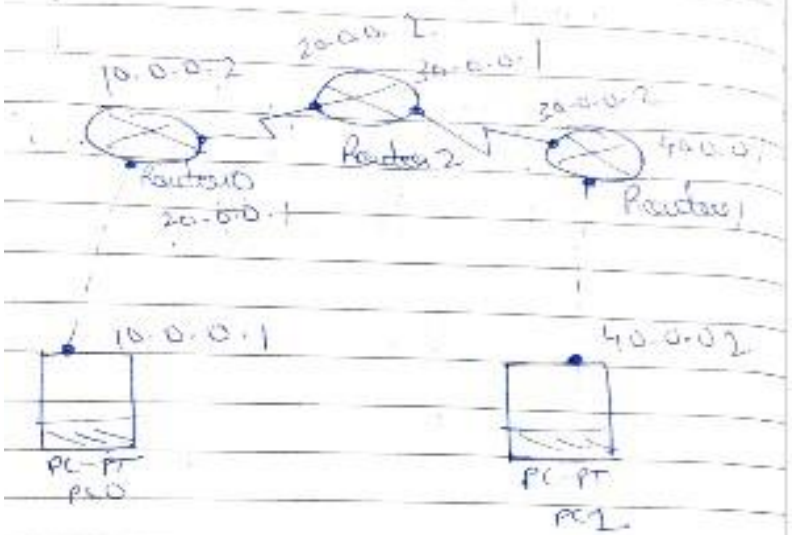
Packets sent = 4, Recieved = 4, Lost = 0
(0% loss)

Approximate round trip times in milli-seconds:
Minimum = 20ms, Maximum = 25ms,
Average = 10ms

∴ Default Routing can be configured even in with multiple routers.//

Aim: Configure default route to the router

Procedure:



To configure Router to route using default routing:-

For Router '0'

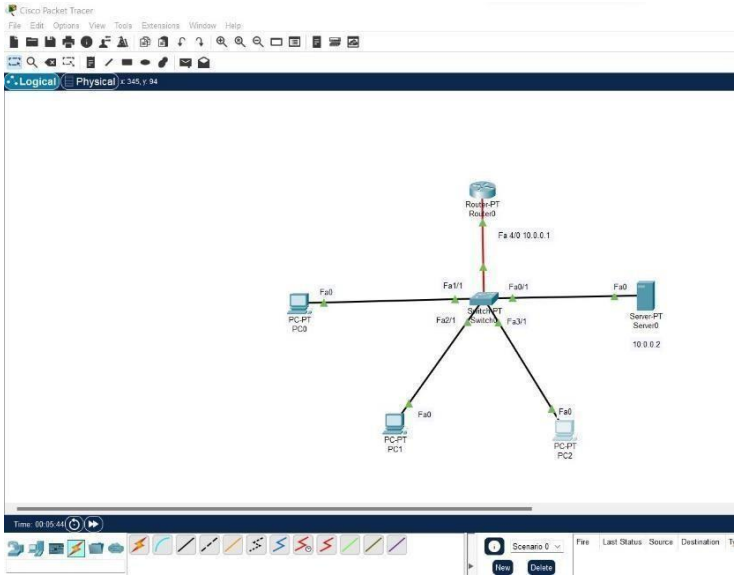
Output

```
Packet Tracer PC Command Line 1.0
C:\>ping 40.0.0.1

Pinging 40.0.0.1 with 32 bytes of data:

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

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

C:\>ping 40.0.0.1

Pinging 40.0.0.1 with 32 bytes of data:

Reply from 10.0.0.10: Destination host unreachable.
Reply from 10.0.0.10: Destination host unreachable.
Reply from 10.0.0.10: Destination host unreachable.
Reply from 10.0.0.10: Destination host unreachable.

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

C:\>ping 40.0.0.1

Pinging 40.0.0.1 with 32 bytes of data:

Request timed out.
Reply from 40.0.0.1: bytes=32 time=10ms TTL=125
Reply from 40.0.0.1: bytes=32 time=10ms TTL=125
Reply from 40.0.0.1: bytes=32 time=10ms TTL=125

Ping statistics for 40.0.0.1:
    Packets: Sent = 4, Received = 3, Lost = 1 (25% loss),
Approximate round trip times in milli-seconds:
    Minimum = 10ms, Maximum = 10ms, Average = 10ms

C:\>
```

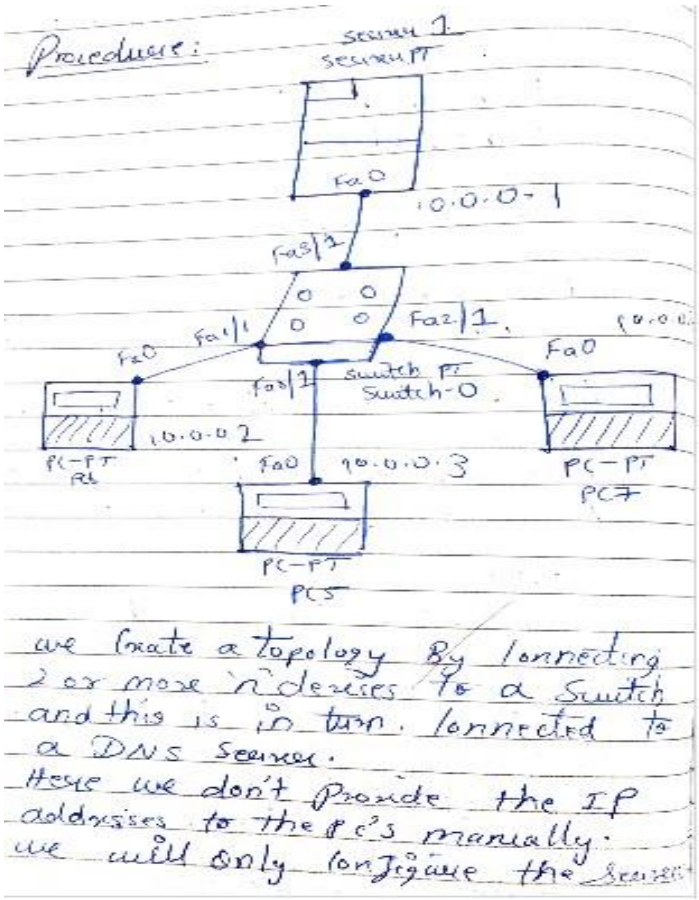
Experiment No 4 Aim of the program

Configuring DHCP within a LAN in a packet Tracer

Topology



Procedure


Procedure:
Server 1
Server PT
Fa0
10.0.0.1
Fa3/1
Fa1/1
Fa2/1
Fa0
Fa0
Switch PT
Switch-0
Fa0/1
10.0.0.2
PC-PT
PC6
Fa0
10.0.0.3
PC-PT
PC7
PC-PT
PC5
we create a topology By connecting
2 or more devices to a Switch
and this is in turn. connected to
a DNS Server.
Here we don't Provide the IP
addresses to the pc's manually.
we will only configure the Server

and the PC's will automatically raise their IP addresses.

Configuring Server:

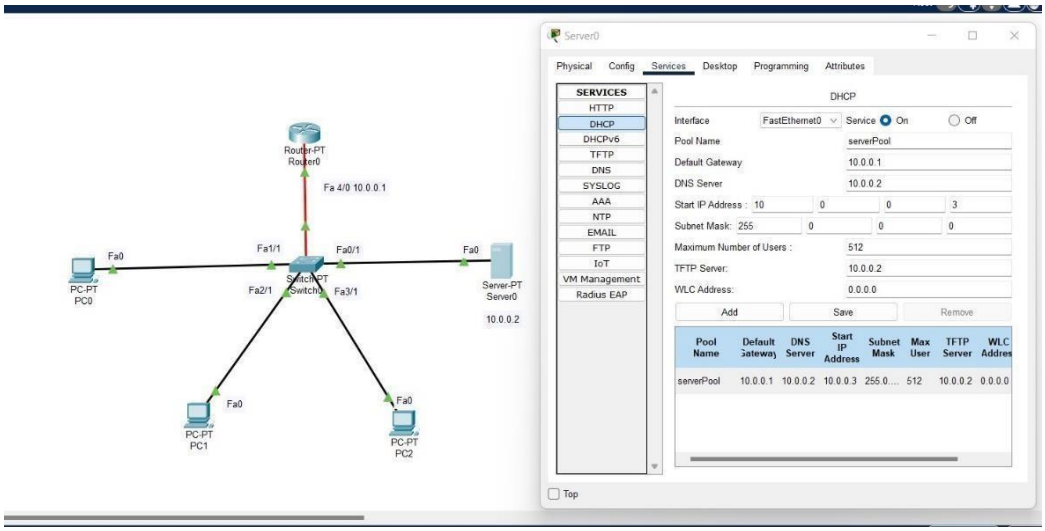
Services → DHCP → Service (ON) → DNS Server (10.0.0.1) → start IP address (10.0.0.2) → TFTP Server (10.0.0.1)

Here DNS Server address is equal to the TFTP Server address.
start IP address will be different.

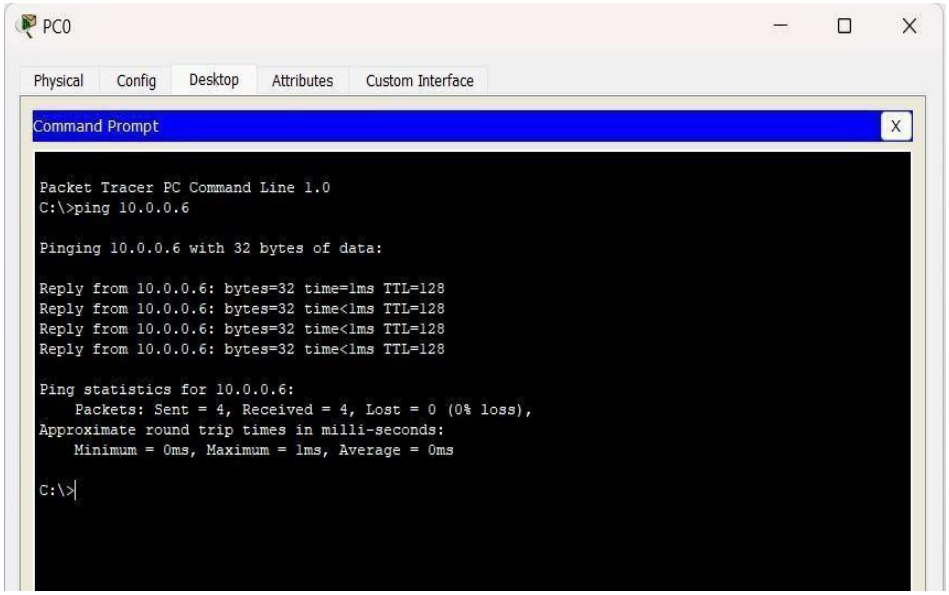
Once this configuration are observe that outcome:

PC6 gets an IP address 10.0.0.2
PC5 gets an IP address 10.0.0.3
PC7 gets an IP address 10.0.0.4

Result: Therefore, the PC's are


Server0
Physical
Config
Services
Desktop
Programming
Attributes
SERVICES
HTTP
DHCP
DHCPv6
TFTP
DNS
SYSLOG
AAA
NTP
EMAIL
FTP
IoT
VM Management
Radius EAP
DHCP
Interface
FastEthernet0
Service
On
Off
Pool Name
serverPool
Default Gateway
10.0.0.1
DNS Server
10.0.0.2
Start IP Address :
Subnet Mask:
Maximum Number of Users :
512
TFTP Server:
10.0.0.2
WLC Address:
0.0.0.0
Add
Save
Remove
Pool Name
Default Gateway
DNS Server
Start IP Address
Subnet Mask
Max User
TFTP Server
WLC Address
serverPool
10.0.0.1
10.0.0.2
10.0.0.3
255.0....
512
10.0.0.2
0.0.0.0
Top
Router-PT
Router0
Fa 4/0 10.0.0.1
Fa1/1
Fa0/1
Fa0
Fa0
PC-PT
PC0
Switch-PT
Switch
Fa2/1
Fa3/1
Server-PT
Server0
10.0.0.2
Fa0
Fa0
PC-PT
PC1
PC-PT
PC2

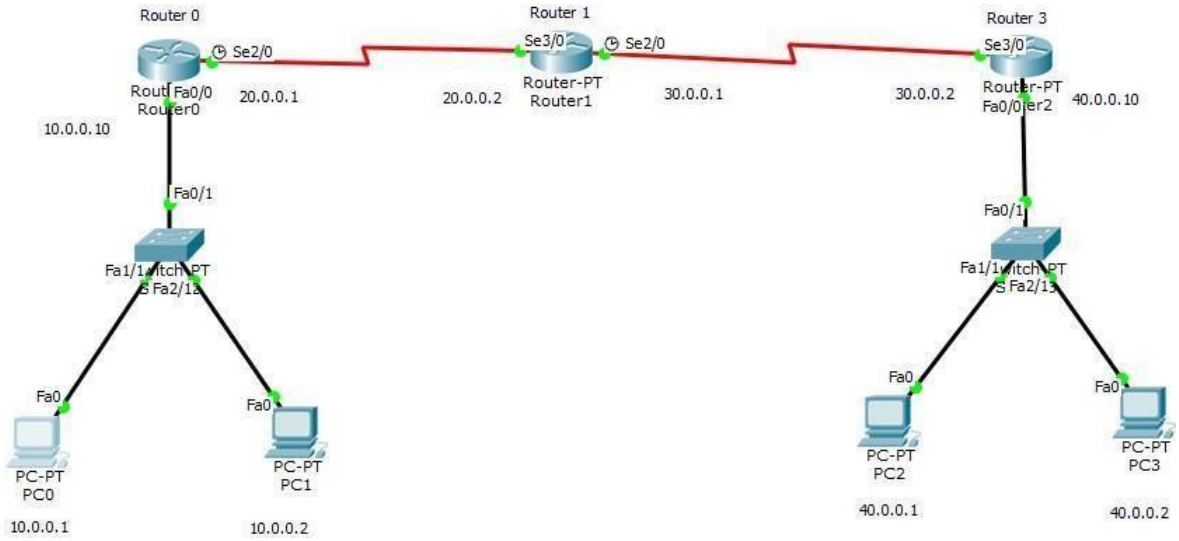
Output


PC0
Physical
Config
Desktop
Attributes
Custom Interface
Command Prompt
X
Packet Tracer PC Command Line 1.0
C:\>ping 10.0.0.6
Pinging 10.0.0.6 with 32 bytes of data:
Reply from 10.0.0.6: bytes=32 time=1ms TTL=128
Reply from 10.0.0.6: bytes=32 time<1ms TTL=128
Reply from 10.0.0.6: bytes=32 time<1ms TTL=128
Reply from 10.0.0.6: bytes=32 time<1ms TTL=128
Ping statistics for 10.0.0.6:
Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
Minimum = 0ms, Maximum = 1ms, Average = 0ms
C:\>

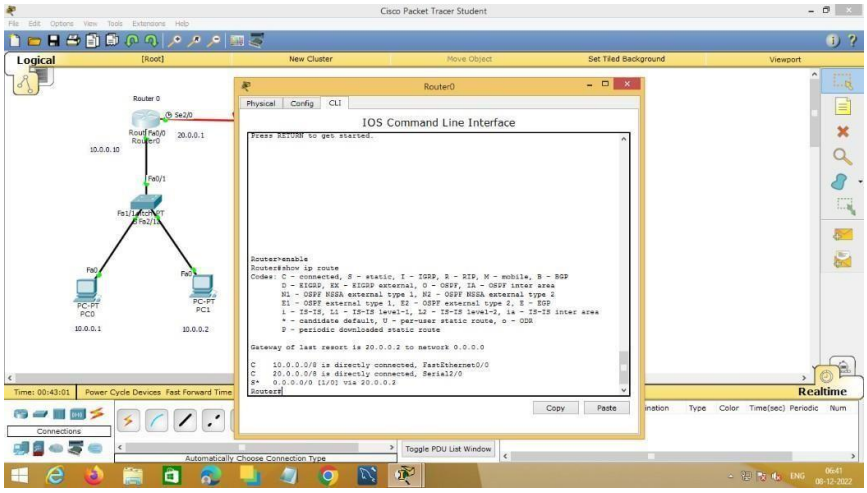
Experiment No 5 Aim of the program

Configuring default router to router

Topology



Procedure



(iii) For Router 3

R3> IP router 0.0.0.0 0.0.0.0 30.0.0.1
R3> IP Router 0.0.0.0 0.0.0.0 20.0.0.1

Outcome

PC> Ping 10.0.0.1

Pinging 10.0.0.1 with 32 bytes of data

Reply from 10.0.0.1 : bytes = 32, time = 1ms TTL = 125
Reply from 10.0.0.1 bytes = 32, time = 2ms TTL = 125
Reply from 10.0.0.1 : bytes = 32, time = 2ms TTL = 125
Reply from 10.0.0.1 : bytes = 32, time = 2ms TTL = 125

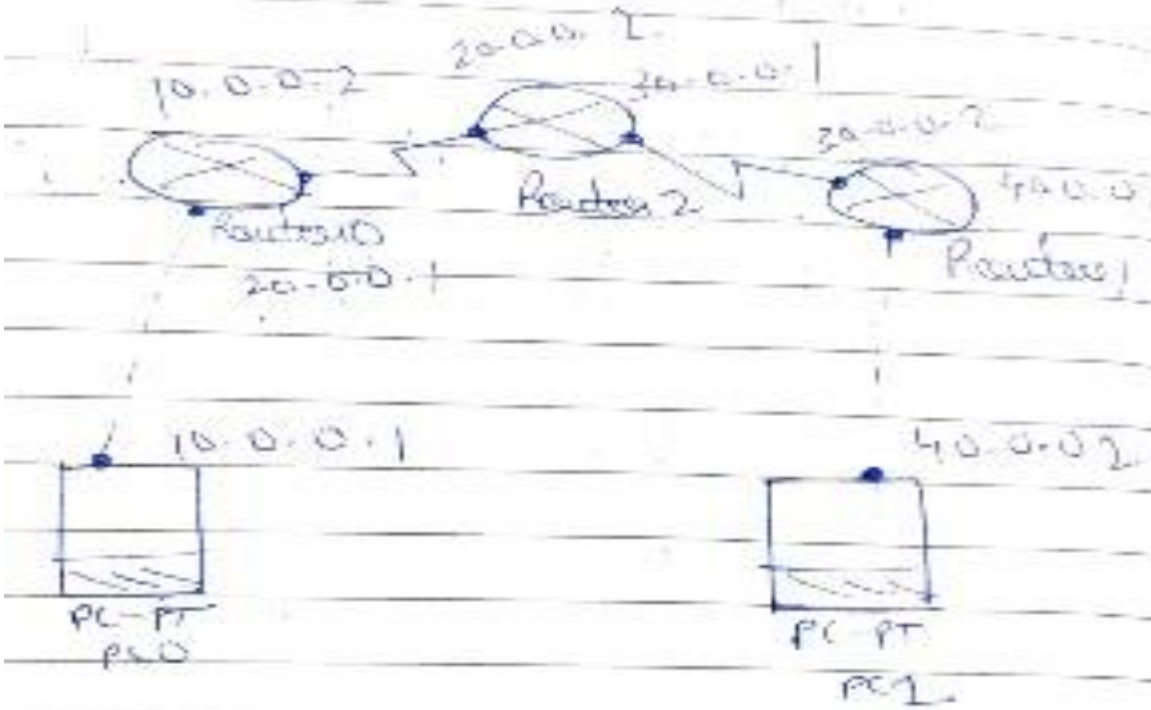
Packets sent = 4, Received = 4, Lost = 0
(0% loss)

Approximate round trip times in milli-seconds:
Minimum = 2ms, maximum = 2ms,
Average = 1ms

∴ Default Routing can be configured over an with multiple routers.//

Aim: Configure default route to the router

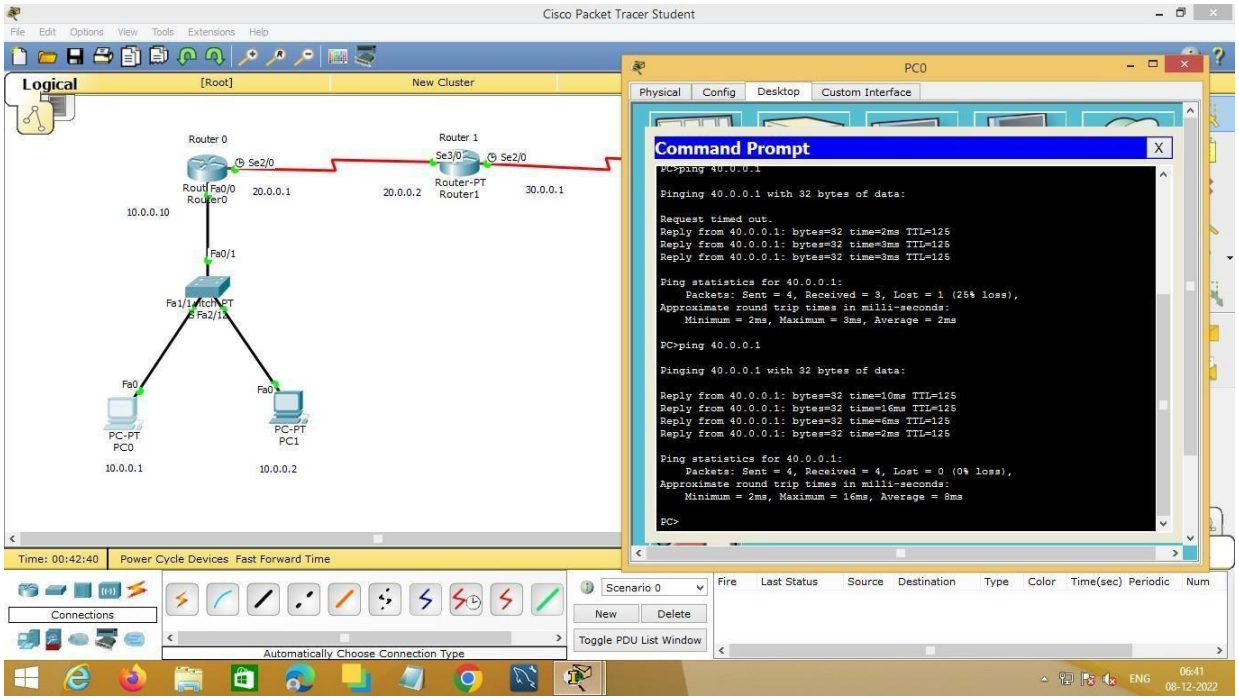
Procedure:



To configure Router to router using default routing:-

For Router '0'

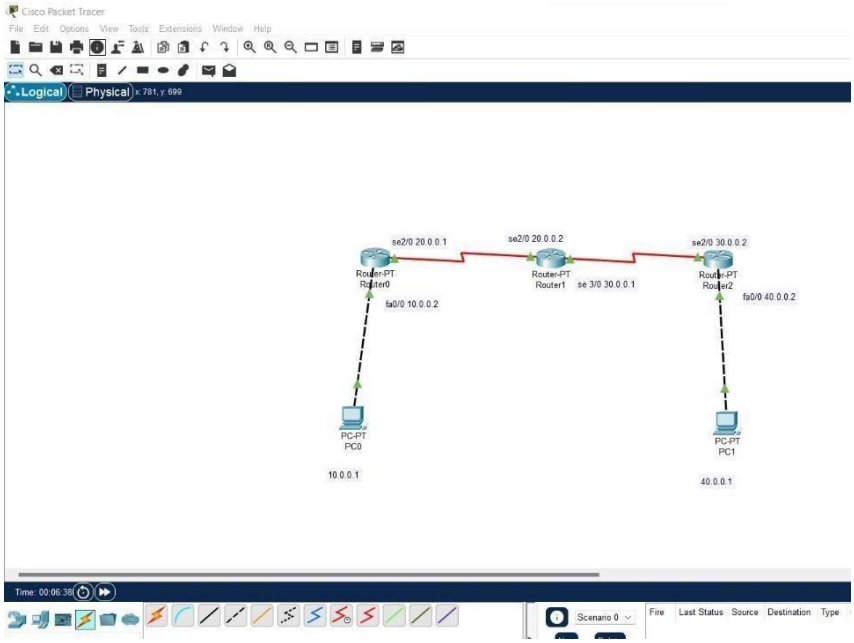
Output

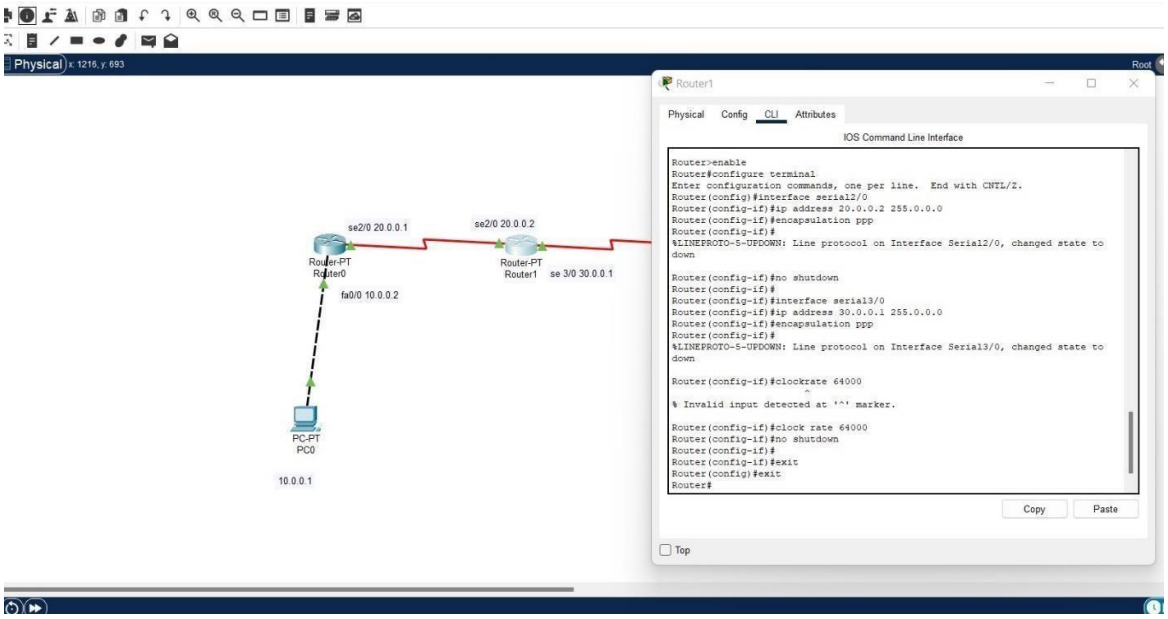

Cisco Packet Tracer Student
File Edit Options View Tools Extensions Help
Logical
[Root]
New Cluster
Router 0
Se2/0
Router 1
Se3/0
Se2/0
Fa0/0
Router0
20.0.0.1
20.0.0.2
Router-PT
Router1
30.0.0.1
10.0.0.10
Fa0/1
Fa0
Fa0
PC-PT
PC0
PC-PT
PC1
10.0.0.1
10.0.0.2
Time: 00:42:40
Power Cycle Devices Fast Forward Time
Connections
Automatically Choose Connection Type
Scenario 0
New
Delete
Toggle PDU List Window
Fire
Last Status
Source
Destination
Type
Color
Time(sec)
Periodic
Num
PC0
Physical
Config
Desktop
Custom Interface
Command Prompt
PC>ping 40.0.0.1
Pinging 40.0.0.1 with 32 bytes of data:
Request timed out.
Reply from 40.0.0.1: bytes=32 time=2ms TTL=125
Reply from 40.0.0.1: bytes=32 time=3ms TTL=125
Reply from 40.0.0.1: bytes=32 time=3ms TTL=125
Ping statistics for 40.0.0.1:
Packets: Sent = 4, Received = 3, Lost = 1 (25% loss),
Approximate round trip times in milli-seconds:
Minimum = 2ms, Maximum = 3ms, Average = 2ms
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=10ms TTL=125
Reply from 40.0.0.1: bytes=32 time=16ms 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
Ping statistics for 40.0.0.1:
Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
Minimum = 2ms, Maximum = 16ms, Average = 8ms
PC>
ENG
06:41
08-12-2022

Experiment No 6 Aim of the program

Configuring RIP Routing Protocol in Routers

Topology





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

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

* We have to give the following commands to look ping between router to router

At Sender's side:-

Router> enable
Config Terminal
interface se 1/0
IP address 20.0.0.1 255.0.0.0
encapsulation PPP
clock rate 64000
No shut down
exit

At reciever side:-

Router> enable
Config terminal
encapsulation PPP
no shutdown
∴ This establishes connection b/w

the routers.

Outcome

Making routers connect 10 and 20 series network

router rip
network 10.0.0.0
network 20.0.0.0

Command Prompt:-

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 = 1ms 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 = 14ms TTL = 125

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

Minimum = 2ms, maximum = 16ms, Avg = 9ms

Result :- Therefore the routers are configured using RIP protocol.

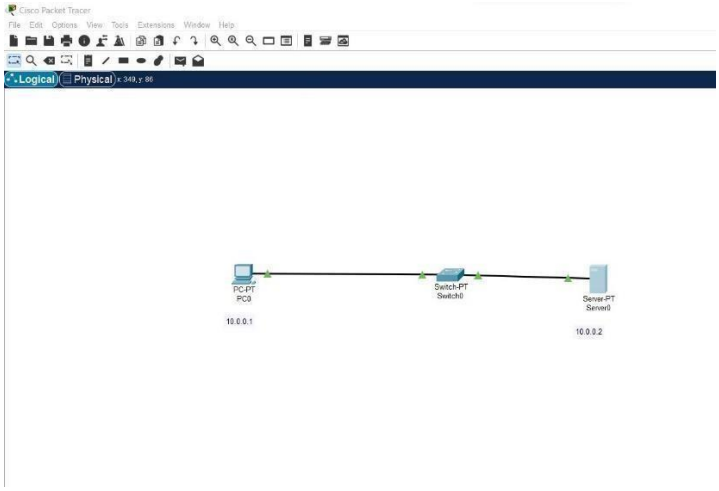
Output

```
C:\>ping 40.0.0.1

Pinging 40.0.0.1 with 32 bytes of data:

Request timed out.
Reply from 40.0.0.1: bytes=32 time=4ms TTL=125
Reply from 40.0.0.1: bytes=32 time=3ms TTL=125
Reply from 40.0.0.1: bytes=32 time=4ms TTL=125

Ping statistics for 40.0.0.1:
    Packets: Sent = 4, Received = 3, Lost = 1 (25% loss),
Approximate round trip times in milli-seconds:
    Minimum = 3ms, Maximum = 4ms, Average = 3ms

C:\>
```

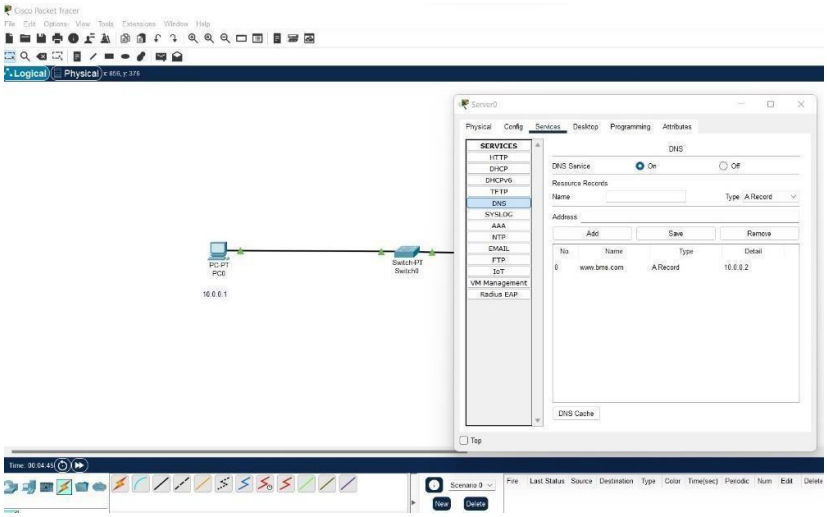
Experiment No 7 Aim of the program

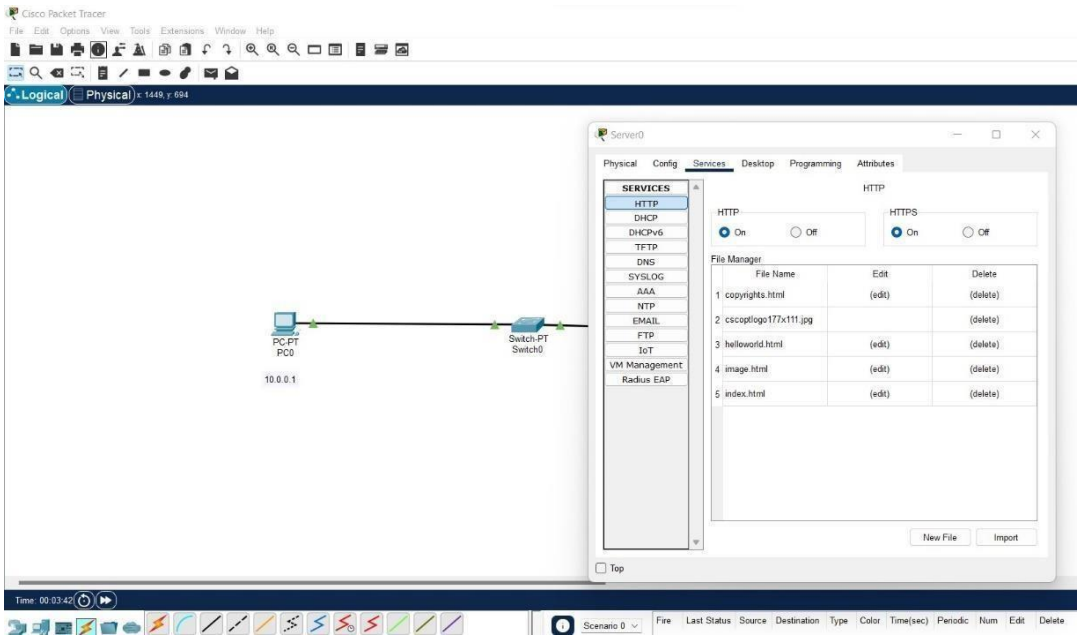
Demonstration of WEB server and DNS using Packet Tracer

Topology



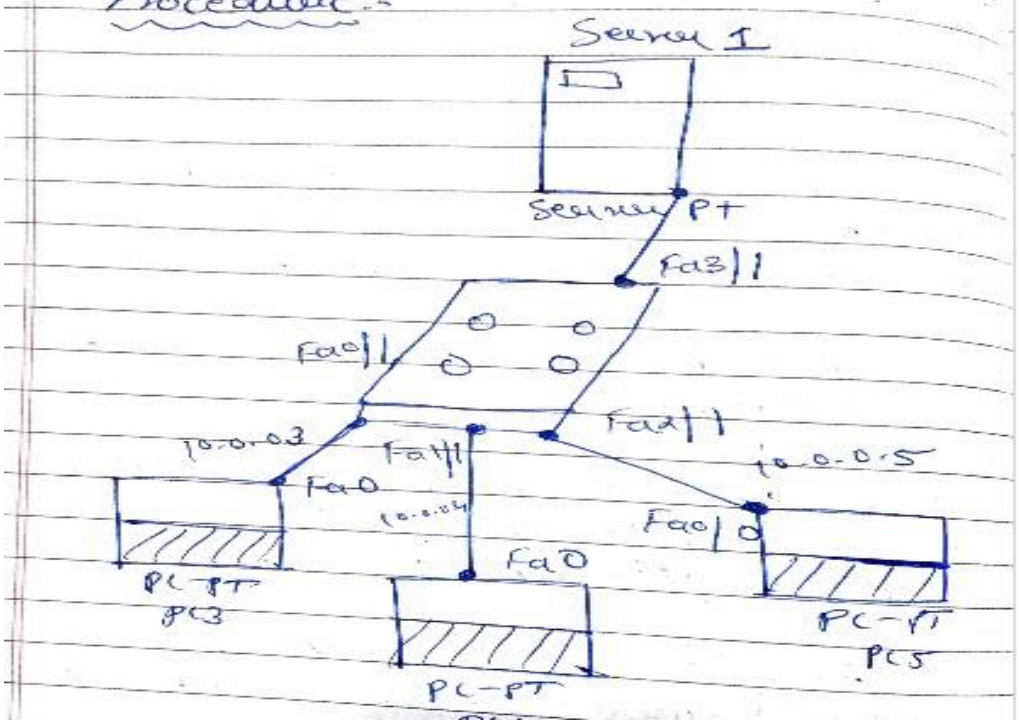
Procedure





Aim:- Demonstration of WEB Server and DNS using packet tracer

Procedure:-



HTTP :-

* Switch it on
* Edit in file manager

DHCP :-

* Pool name
* DNS Server address
* start IP address
* No TFTP IP address

Add

DNS :-

Name :- Random name (Eg : Omar)

Address :- 10.0.0.1

↳ www.omar.com.

↳ IP address of the Server.

Click "Add"

Go to one of the clients / PC :-

we change IP configuration to DHCP from static then we can get the pc's IP address automatically.

Desktop → web browser →

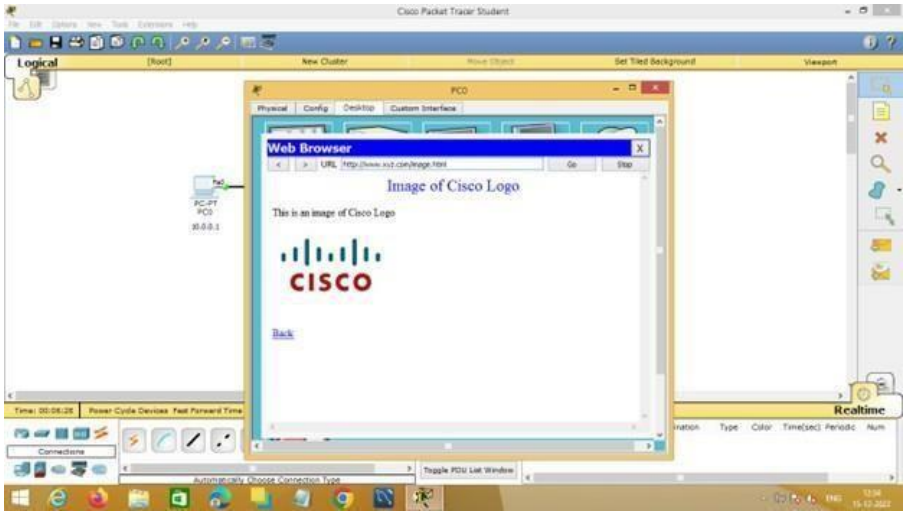
Syntax :

www.omar.com/helloword.html

File → helloword.

↳ Content's of helloword will

Output



Cycle-2 Experiment No 1 Aim of the Experiment

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

Code

```
#include<bits/stdc++.h> using
namespace std; void receiver(string
data, string key);

string xor1(string a, string b)
{

        string result = "";

        int n = b.length();

         for(int i = 1; i < n; i++)
        {
                if (a[i] == b[i])
                        result += "0";
else
                         result += "1";
        }
        return result;
}

string mod2div(string dividend, string divisor)
{
        int pick = divisor.length();
```

```
        string tmp = dividend.substr(0, pick);

        int n = dividend.length();

         while (pick < n)
        {
                if (tmp[0] == '1')                              tmp =
xor1(divisor, tmp) + dividend[pick];
                else
                        tmp = xor1(std::string(pick, '0'), tmp) +
                                dividend[pick];

                 pick += 1;
        }
        if (tmp[0] == '1')
tmp = xor1(divisor, tmp);
        else
                 tmp = xor1(std::string(pick, '0'), tmp);

        return tmp;
}

void encodeData(string data, string key)
{
         int l_key = key.length();

            string appended_data = (data +std::string(l_key - 1, '0'));

         string remainder = mod2div(appended_data, key);
```

```
        string codeword = data + remainder;
cout << "Remainder : "
                << remainder << "\n";
          cout << "Encoded Data (Data + Remainder) :"
                << codeword << "\n";
         receiver(codeword, key);
}
void receiver(string data, string key)
{       string currxor = mod2div(data.substr(0, key.size()), key);
int curr = key.size();      while (curr != data.size())
        {
                if (currxor.size() != key.size())
                {
                        currxor.push_back(data[curr++]);
                }
                else
                {
                        currxor = mod2div(currxor, key);
                }
        }
          if (currxor.size() == key.size())
        {
                currxor = mod2div(currxor, key);
        }
        if (currxor.find('1') != string::npos)
        {       cout << "there is some error in data" << endl;
        }
        else
```

```
        {
                cout << "correct message recieved" << endl;
        }
} int
main()
{

        string data = "1011101";
string key = "100010000001";

         encodeData(data, key);

        return 0;
}
```

```
Remainder : 10001011000
Encoded Data (Data + Remainder) :101110110001011000
correct message recieved


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

Experiment No 2 Aim of the Experiment

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

Code

```
#include<stdio.h>
                    struct node
                    {
                        unsigned dist[20];
                        unsigned from[20];
                    }rt[10];
                    int main()
                    {
                        int costmat[20][20];
                        int nodes,i,j,k,count=0;
                        printf("\nEnter the number of nodes : ");
                        scanf("%d",&nodes);//Enter the nodes
                        printf("\nEnter the cost matrix :\n");
                        for(i=0;i<nodes;i++)
                        {
                            for(j=0;j<nodes;j++)
                            {
                                scanf("%d",&costmat[i][j]);
                                costmat[i][i]=0;
                                rt[i].dist[j]=costmat[i][j];//initialise the distance equal
                    to cost matrix
                                rt[i].from[j]=j;
                            }
                        }
                            do
                            {
                                count=0;
                                for(i=0;i<nodes;i++)//We choose arbitary vertex k and we
                    calculate the direct distance from the node i to k using the cost
                    matrix
                                //and add the distance from k to node j
                                for(j=0;j<nodes;j++)
                                for(k=0;k<nodes;k++)
                                    if(rt[i].dist[j]>costmat[i][k]+rt[k].dist[j])
                                    {//We calculate the minimum distance
                                        rt[i].dist[j]=rt[i].dist[k]+rt[k].dist[j];
```

```
                        rt[i].from[j]=k;
                        count++;
                    }
            }while(count!=0);
            for(i=0;i<nodes;i++)
            {
                printf("\n\n For router %d\n",i+1);
                for(j=0;j<nodes;j++)
                {
                    printf("\t\nnode %d via %d Distance %d
",j+1,rt[i].from[j]+1,rt[i].dist[j]);
                }
            }
        printf("\n\n");
        getch();
    }
```

Output

```
Enter the cost matrix :
0 1 5 6
1 0 3 4
5 3 0 2
6 4 2 0


 For router 1

node 1 via 1 Distance 0
node 2 via 2 Distance 1
node 3 via 2 Distance 4
node 4 via 2 Distance 5

 For router 2

node 1 via 1 Distance 1
node 2 via 2 Distance 0
node 3 via 3 Distance 3
node 4 via 4 Distance 4

 For router 3

node 1 via 2 Distance 4
node 2 via 2 Distance 3
node 3 via 3 Distance 0
node 4 via 4 Distance 2

 For router 4

node 1 via 2 Distance 5
node 2 via 2 Distance 4
node 3 via 3 Distance 2
node 4 via 4 Distance 0



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

Experiment No 3 Aim of the Experiment

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

Code

```
#include<bits/stdc++.h>
#include <limits.h>
#include <stdio.h>
 using namespace std;

#define V 5


int minDistance(int dist[], bool Test[])
{

    int min = INT_MAX, min_index;

    for (int v = 0; v < V; v++)
        if ( Test[v] == false && dist[v] <= min)
            min = dist[v], min_index = v;

    return min_index;
}


void printSolution(int dist[])
{
    printf("Vertex \t\t Distance from Source\n");
    for (int i = 0; i < V; i++)
        printf("%d \t\t %d\n", i, dist[i]);
}


void dijkstra(int graph[V][V], int src)
{
    int dist[V];
    bool Test[V];
    for (int i = 0; i < V; i++)
        dist[i] = INT_MAX, Test[i] = false;
```

```
    dist[src] = 0;


    for (int count = 0; count < V - 1; count++) {

        int u = minDistance(dist, Test);


        Test[u] = true;


        for (int v = 0; v < V; v++)


            if (!Test[v] && graph[u][v] && dist[u] != INT_MAX
                && dist[u] + graph[u][v] < dist[v])
                dist[v] = dist[u] + graph[u][v];
    }


    printSolution(dist);
}


int main()
{

    int graph[V][V] ;
    cout<<"Enter the graph "<<endl;
    for(int i = 0; i<V; i++)
    {
        for(int j = 0; j<V; j++)
            cin>>graph[i][j];
    }

    dijkstra(graph, 0);

    return 0;
}
```

Output

```
Enter the graph
0 1 4 0 5
1 0 3 6 0
4 3 0 0 6
0 6 0 0 10
5 0 6 10 0
Vertex          Distance from Source
0               0
1               1
2               4
3               7
4               5
```

Experiment No 4 Aim of the Experiment

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.

Code

Server:

```
from socket import * serverName = '' serverPort =

12530 serverSocket = socket(AF_INET,SOCK_STREAM)

serverSocket.bind((serverName,serverPort))

serverSocket.listen(1) print("The server is ready to

receive") while 1:

   connectionSocket, addr = serverSocket.accept()

sentence = connectionSocket.recv(1024).decode()     try:

       file = open(sentence,"r")        l =

file.read(1024)

connectionSocket.send(l.encode())

file.close()     except Exception as e:

       message = "No such file exist"

connectionSocket.send(message.encode())      connectionSocket.close()
```

Client:

```
from socket import *

serverName = '192.168.1.104'

serverPort = 12530

clientSocket = socket(AF_INET, SOCK_STREAM)

clientSocket.connect((serverName,serverPort))

sentence = input("Enter file name")

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

clientSocket.recv(1024).decode() print ('From Server:',

filecontents) clientSocket.close()
```

OUTPUT:

```
C:\Windows\System32\cmd.exe - py server.py
Microsoft Windows [Version 10.0.19045.2486]
(c) Microsoft Corporation. All rights reserved.

D:\con054-main\CON_LAB\lab10>py server.py
The server is ready to receive
```

```
C:\Windows\System32\cmd.exe
icrosoft Windows [Version 10.0.19045.2486]
c) Microsoft Corporation. All rights reserved.

:\con054-main\CON_LAB\lab10>py client.py
nter file name: try.txt
rom Server: HELLO WORLD
```

Experiment No 5 Aim of the Experiment

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.

Code

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)


    file=open(sentence,"r")      l=file.read(2048)


    serverSocket.sendto(bytes(l,"utf-8"),clientAddress)

print("sent back to client",l)      file.close() Client:

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 ('From Server:',

filecontents)


clientSocket.close()
```

OUTPUT:

```
C:\Windows\System32\cmd.exe
Microsoft Windows [Version 10.0.19045.2486]
(c) Microsoft Corporation. All rights reserved.

D:\con054-main\CON_LAB\lab10>py uclient.py
Enter file name: try.txt
From Server: b'HELLO WORLD\n\n'

D:\con054-main\CON_LAB\lab10>
```

```
C:\Windows\System32\cmd.exe - py userver.py
Microsoft Windows [Version 10.0.19045.2486]
(c) Microsoft Corporation. All rights reserved.

D:\con054-main\CON_LAB\lab10>py userver.py
The server is ready to receive
sent back to client HELLO WORLD
```