

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 ~~switch~~
- 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 ^{transmit} between a source & a destination

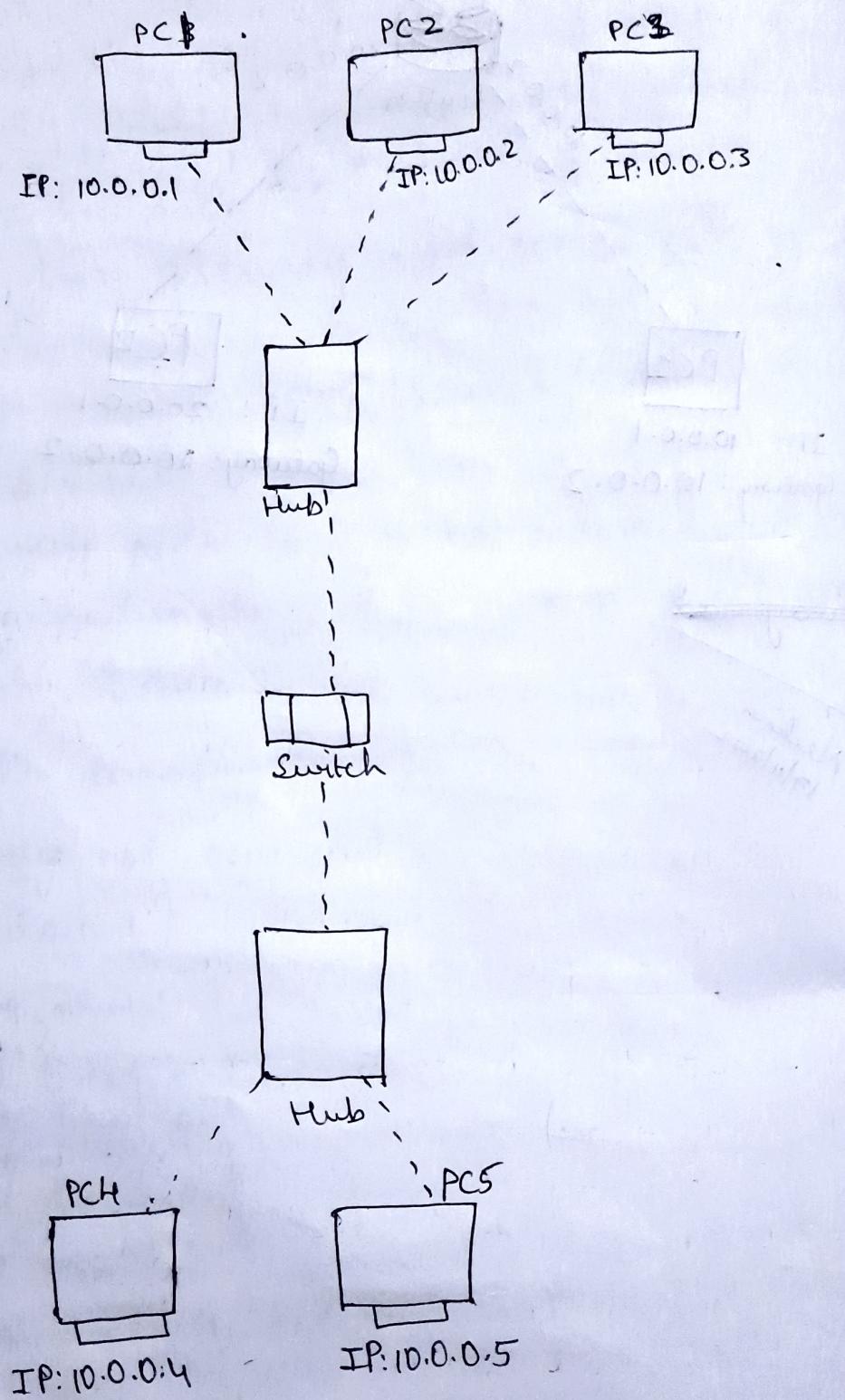
Result: The transmitting of PDUs were successful between the source & destination

Observations:

- Hubs broadcast a PDU to all the connected devices in a network, when a source transmits a PDU to the 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 ~~is~~ unicasting between the source & destination via the switch.
- If a receiver host isn't connected to a internetwork, ~~host~~ a message cannot be pinged & hence response will be timed out.

*Nitin's
extremely*

Topology (Expt 1):



Aim: To configure IP address to Router in Packet Tracer. Explore the following messages: Ping responses, ~~Destination unreachable~~, Request timed out, Reply.

Topology: Star topology

Procedure: • Two end devices are connected to a router

- The IP addresses of the end devices are set.
- IP address of a router can be set in the CLI window of the respective router, by executing the following commands:

```
>* enable
# config terminal
# interface Fa0/0 → interface network
# ip address 10.0.0.2 255.0.0.0
# no shutdown
# exit
```

- The gateway addresses of the end devices are mentioned.
- A message can be pinged to check if the connection is established between the router & end devices.

Result: A ping message is successfully sent & received from one end device to another end device.

Observations:

Observation:

- When a device pings the destination end device initially, the request is timed out as the gateway address is ~~not~~ for the interface is not specified / set.
- When the gateway address is set on both end devices, the router can successfully ping the message from & to both the end devices.
- A gateway path has to be established & checked for connection, to send a message from one end device to another.

Output: ~~PC → ping 20.0.0.1~~

1) Gateway not connected

From: 10.0.0.1

pc > ping 20.0.0.1

Request timed out

Request timed out

Packets =

Request timed out

Request timed out

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

2) Gateway connection established

From : 10.0.0.1

PC > ping 20.0.0.1

~~Reply from~~ Pinging 20.0.0.1 with 32 bytes of data:

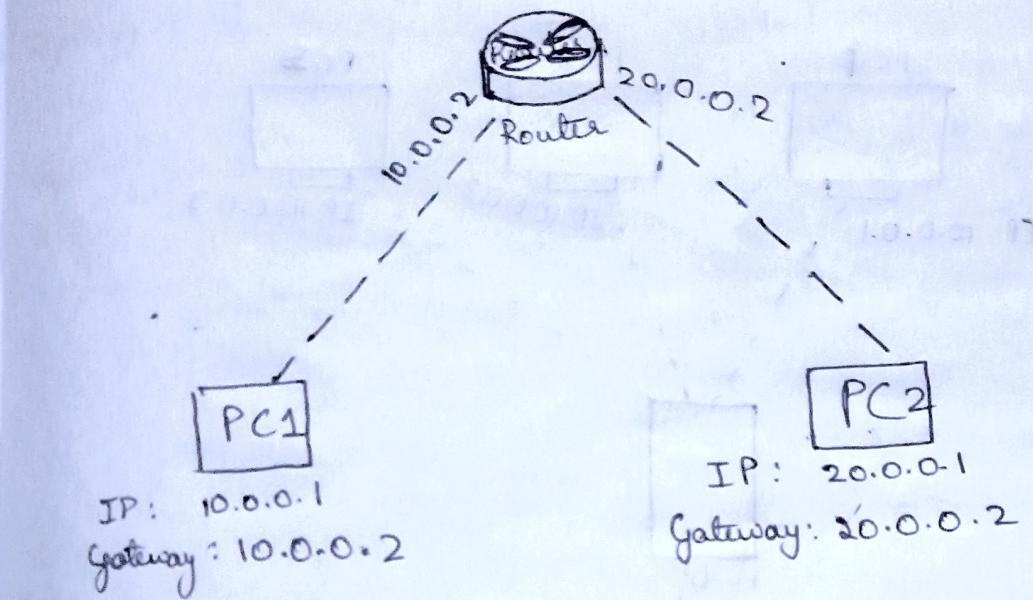
Reply from 20.0.0.1: bytes=32	time=0ms TTL=127
Reply from 20.0.0.1: bytes=32	time=0ms TTL=127
Reply from 20.0.0.1: bytes=32	time<0ms TTL=127
Reply from 20.0.0.1: bytes=32	time=0ms TTL=127

Topology (1)

end device
as the
ace is

on both
fully
the

& checked
one end



~~Gateway end~~

Neekina
17/1/2022

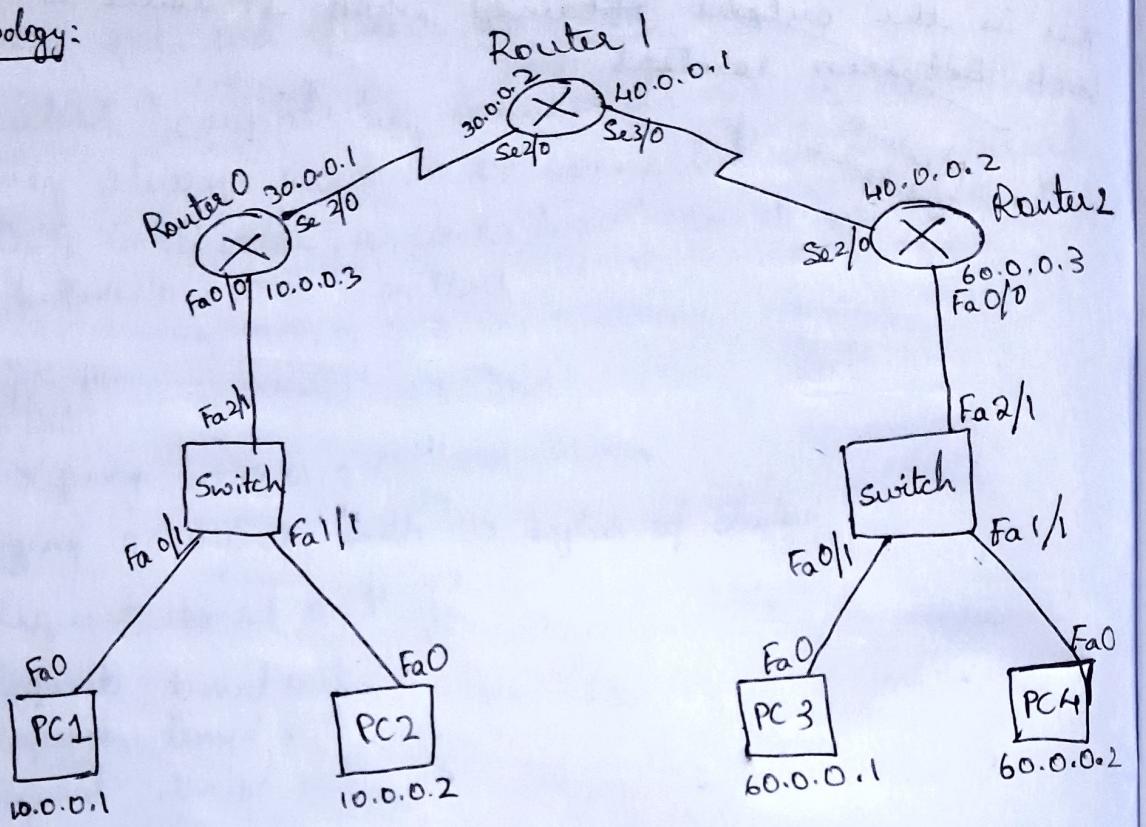
1. loss)

Data:

=127
127
127
127

Aim: To configure default route to the routers

Topology:



Procedure:

- Construct topology using end devices, switch & routers as drawn above
- Set the IP addresses of all end devices. All addresses of devices connected to a switch must have the same network id.
- The gateway addresses of the end devices are set
- Default route is set by executing the following commands:

Router # config terminal

Router(config)# ip route 0.0.0.0 0.0.0.0 30.0.0.1

Router(config)# ip route 0.0.0.0 0.0.0.0 40.0.0.2
any destination id any subnet mask next hop

The connections one end-device

Observations:

- The packets of the end devices timed out)
- When a default route is set), the router not recognize destination
- Only 50% packets go to a router from a switch end-device
- a default route other router to the next
- while pingin to 30.0.0.1 of pack because no destination Then the router to the echo

Output:

PC > ping 30.0.0.1

Pinging 30.0.0.1

Request timed out

Request timed out

Request timed out

Ping statistics for 30.0.0.1

- 12/22
- The connections are checked by pinging packets from one end-device to another via interfaces.
- Observations:
- The packets suffer a loss when the gateways of the end devices aren't set (results in request timed out)
 - When a default route isn't set (but gateway is set), the packets pinged from a PC will not recognize an interface router (results in destination host unreachable)
 - Only 50% part of the packets are transferred to a router. But all packets are pinged from a source end-device & destination end-device. The middle routers are designed to have a default route to send packets to two other routers. Hence 50% packets are transferred to the next interface & 50% are sent back.
 - While pinging to the destination end-device, ~~one~~ 1 of packets are sent first via the switch because ~~one~~ 1 packets are sent back by the destination to confirm the destination end-device. Then the rest of the packets are sent to the recognized destination end-device.

following

30.0.0.1
0.0.2
ext hop

Output:

PC > ping 30.0.0.2

Pinging 30.0.0.2 with 32 bytes of data:

Request timed out

Request timed out

Request timed out

Request timed out

Ping statistics for 30.0.0.2: Packets: Sent=4, Received=0, lost=4 (100% loss)

PC > ping 10.0.0.1.

The gateway addresses are not set. Hence there is no way pinging to the routers

PC > ping 10.0.0.1.

Pinging 10.0.0.1 with 32 bytes of data:

Reply from 10.0.0.3: Destination host unreachable

Ping statistics for 10.0.0.1:

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

The gateway address is set & default route between the routers is not set.

PC > ping 10.0.0.2

Pinging 10.0.0.2 with 32 bytes of data:

Request timed out

Reply from 10.0.0.2: bytes=32 time=11ms TTL=253

Request timed out

Reply from 10.0.0.2: bytes=32 time=2ms TTL=253

Ping statistics for 10.0.0.2:

 Packets: Sent=4, Received=2, Lost=2 (50% loss)

The default route is set. This output is obtained after pinging the routers.

PC > Ping 60.0.0.1
Pinging 60.0.0.1 with 32 bytes of data:
Reply from 60.0.0.1: bytes=32 time=13ms TTL=125
Reply from 60.0.0.1: bytes=32 time=3ms TTL=125
Reply from 60.0.0.1: bytes=32 time=10ms TTL=125
Reply from 60.0.0.1: bytes=32 time=3ms TTL=125

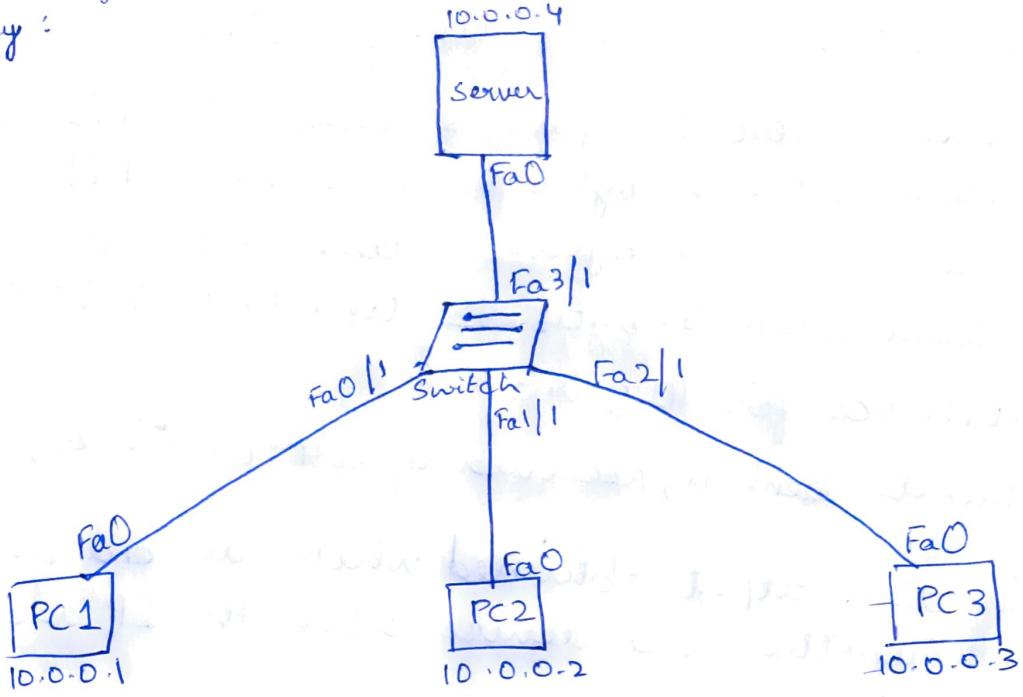
Ping statistics for 60.0.0.1:
Packets: Sent = 4, Received = 4, Lost = 0 (0% loss)

This is output obtained when the destination end device is pinged

~~D~~
~~1/12~~

Aim: Configure DHCP to server

Topology:



Procedure:

- Construct topology as above
- Configure an IP address only to the server
The network IDs of the server & end devices must be the same
- Under the DHCP menu in Services window of the server, switch on the service button for the interface, give a pool name & set the start IP address. Then save the changes.
- This initiates the DHCP
- Now change the IP configuration of all end devices to DHCP from static. The IP addresses are dynamically allocated to each end device

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 ~~given~~ to end devices when the DHCP is initiated in the server
- RARP is used to assign the IP address of a device if ~~a~~ 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
Reply from 10.0.0.3: bytes=32 time=0ms TTL=128
Reply from 10.0.0.3: bytes=32 time=0ms TTL=128

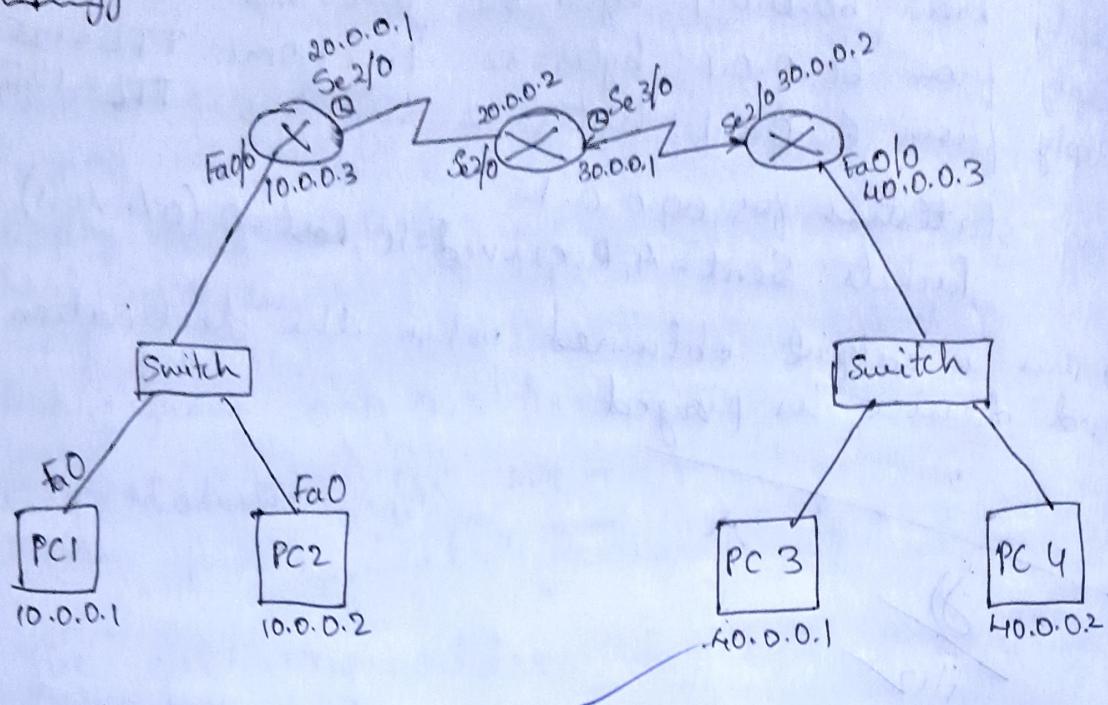
Ping statistics for 10.0.0.3:

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

This is the output obtained when an end device pings to another end device when the IP address is set via DHCP

The same output is obtained when a server pings an end device.

Lab - 5
Aim: Configure RIP (Routing Information Protocol) in Router
Topology:



Procedure:

- Construct a topology as above.
- End devices, ^{IP addresses} are configured. Gateway addresses are set.
- The routers are connected with a serial clock connection.
- Point-to-point protocol (PPP) is used to encapsulate all network connections within routers. (after the IP address of the routers are set)

Router(config-if) # ip address	<u>20.0.0.1</u>	<u>255.0.0.0</u>
Router(config-if) # encapsulation	PPP	
Router(config-if) # clock rate	64000	
Router(config-if) # no shutdown		

- Clock rate is set to only one of the routers in a pair, where serial connection is established
- IP route need not be set between the routers
- RIP is used to connect a router to the remaining routers ^{dynamically}. Network ID is mentioned so that a connection is established

Router(config) # router rip]
Router(config-router) # network 10.0.0.0	
Router(config-router) # network 20.0.0.0	

connected networks.

After RIP is established, a message can be pinged from one end device to another end device without setting the IP route

Observation:

- End device cannot ping an end device of different network id without either ip route nor ~~RIP~~ RIP, with PPP
- Clock rate can be set only to DCE interfaces.
- IP route is not necessary when RIP is established to ping two end devices of different network ids.
- Ping routes from end devices without setting a route will result in destination host unreachable.

Output:

i) When no gateway or RIP is set:

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)

ii) When only gateway is set & RIP isn't set

PC > ping 30.0.0.1

Pinging 30.0.0.1 with 32 bytes of data:

Reply from 10.0.0.3: Destination host unreachable

Ping statistics for 30.0.0.1:

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

iii) when both gateway & RIP is set

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=9ms TTL=125

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

Reply from 40.0.0.1: bytes=32 time=16ms TTL=125

Reply from 40.0.0.1: bytes=32 time=9ms TTL=125

Ping statistics for 40.0.0.1:

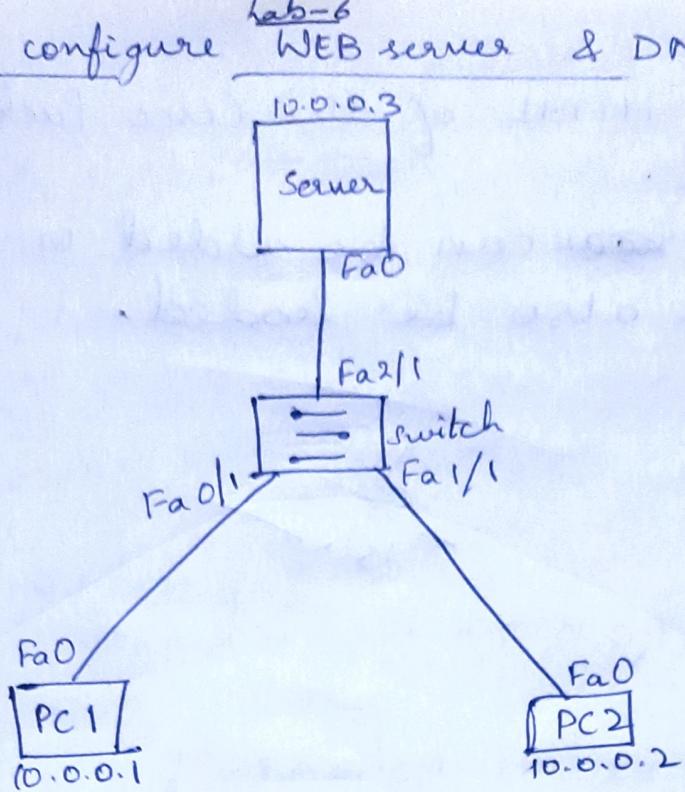
Packets: Sent=4, Received=4, Lost=0 (0% loss)

Note: while pinging via a switch, the first packet is sent to confirm the destination and device. Hence the first packet results in a timed out situation. 4 packets are sent & 3 packets are received.

~~8/12~~

Aim : How to configure WEB server & DNS server

Topology:



Procedure:-

- The above topology is contruted. The IP address of the server is set.
- The DHCP is initiated to the server. The IP address of the end devices are dynamically set.
- The HTTP service is set to on.
- The DNS service is set to on. A new url is mentioned by giving the address of the server. The seconde is added & saved.
- The web browser of an end device is opened & searched for the url created url

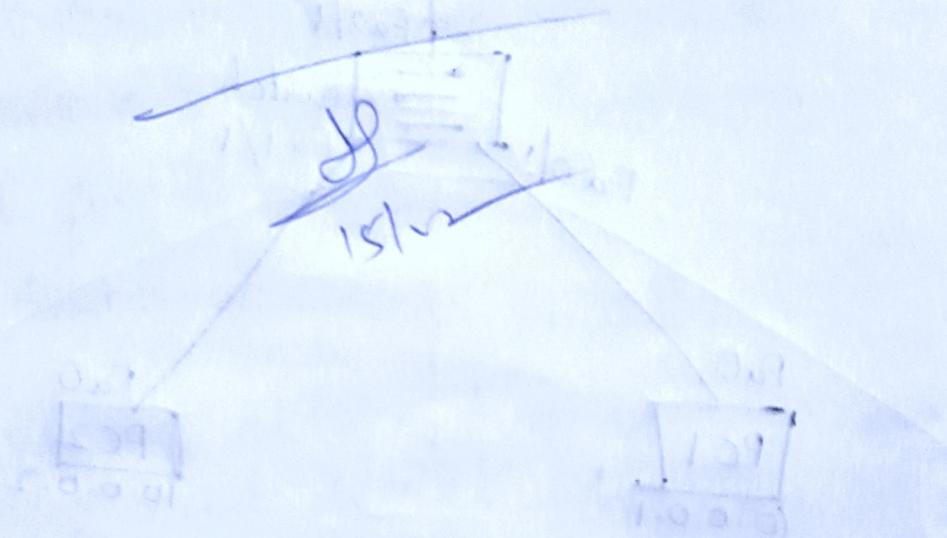
Observations:-

- If the default gateway & DNS server is not set in the DHCP window, the created url is not loaded.
- The url is functioning if ip address is set statically & using DHCP
- The url results in a request timed out if the address of url is not same as server address

Output:

The basic HTML of the Cisco Packet Tracer is loaded.

The files ~~can~~ can be added in the HTTP window & also be loaded.



Lab-7

Write a program for error detection using CRC 16-bit

```

#include <iostream>
using namespace std;
int XOR ( int a, int b)
{
    return (a ^ b);
}

int main()
{
    int i= 0, j=0 ; n;
    cout << "Enter no. of data ";
    cin >> n;
    int a[n+16], rem[n];
    int g[] = {1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1};
    int m=n+16;
    int s = sizeof(a)/sizeof(a[0]);
    for (i=0; i<m; i++)
        if (i>=n)
            a[i]=0;
        else
            cin >>a[i];
    if (i<n)
        rem[i]=a[i];
    for (i=0; i<m; i++)
        cout << a[i];
    for (i=0; i<n; i++)
        if (a[i] != 1)
            continue;
    for (j=0; j<17; j++)
        a[i+j] = a[i+j]^g[j];
    for (i=0; i<n; i++)
        a[i]=rem[i];
    cout << "\n" << "----" << "\n";
    for (int i=0; i<m; i++)
        cout << a[i];
}

```

```
for (i=0; i < n; i++)  
    if (a[i] != 1)  
        continue;  
    for (j=0; j < 17; j++)  
        a[i+j] = a[i+j] ^ g[j];  
  
for (i=0; i < m; i++)  
    cout << a[i];  
  
return 0;
```

{

Output:

Enter the data to be inserted

2
1
1
{111011000001100011
001000000010000000N
12/10/2023

Leaky Bucket Algorithm

```

#include <iostream>
#include <bits/stdc++.h>
using namespace std;

int main()
{
    int cap, out, n, ch;
    int buc = 0;
    cout << "Enter bucket capacity: ";
    cin >> cap;
    cout << "Enter outflow rate: ";
    cin >> out;
    while (1)
    {
        cout << "1. Insert \n 2. Exit";
        cout << "Enter choice: ";
        cin >> ch;
        switch (ch)
        {
            case 1: cout << "Enter packet size: ";
                cin >> n;
                if ((buc + n) <= cap)
                {
                    buc = buc + n;
                    cout << buc << "\n";
                    buc = buc - out;
                    if (buc < 0)
                        buc = 0;
                    cout << "After outflow: " << buc;
                }
                else if ((buc + n) > cap)
                    cout << "Bucket overflow";
                break;
            case 2: cout << "No inputs. Program exited";
                break;
        }
    }
}

```

Output:

Enter bucket capacity: 50

Enter outflow rate: 10

1. Insert

2. Exit

Enter your choice: 1

Enter the packet size: 40

After outflow: 30

Enter your choice: 1

Enter the packet size: 5

35

After outflow: 25

Enter your choice: 1

Enter the packet size: 30

Bucket overflow

N
Rajiv

Bellman-Ford algorithm

```

# include < iostream >
# include < bits/stdc++.h >
# define MAX 10
int Bellman(int G[20][20], int V, int E, int edge[20][20])
using namespace std;
typedef struct edge
{ int src, dest, wt;
} edge;
void bellman (int nv, edge e[], int src, int ne)
{
    int u, v, weight, i, j = 0;
    int dis[MAX];
    for (i = 0; i < nv; i++)
    {
        dis[i] = 999;
    }
    dis[src] = 0;
    for (i = 0; i < nv - 1; i++)
    {
        for (j = 0; j < ne; j++)
        {
            u = e[j].src;
            v = e[j].dest;
            weight = e[j].wt;
            if (dis[u] != 999 & dis[u] + weight < dis[v])
                dis[v] = dis[u] + weight;
        }
    }
    for (j = 0; j < ne; j++)
    {
        u = e[j].src;
        v = e[j].dest;
        weight = e[j].wt;
        if (dis[u] + weight < dis[v])
            cout << "Negative Cycle Present";
        return;
    }
    cout << "Vertex" << " Dist from source";
    for (i = 1; i <= nv; i++)
    {
        cout << i << " " << dis[i];
    }
}

```

```

int main()
{
    int nv, ne, src;
    edge e[MAX];
    cout << "Enter no. of vertices";
    cin >> nv;
    cout << "Enter the source vertex of graph";
    cin >> src;
    cout << "Enter no. of edges";
    cin >> ne;
    for (int i=0; i<ne; i++)
    {
        cout << "For edge " << i+1 << "=";
        cout << "Enter source vertex ";
        cin >> e[i].src;
        cout << "Enter destination vertex ";
        cin >> e[i].dest;
        cout << "Enter weight ";
        cin >> e[i].wt;
    }
    bellman(nv, e, src, ne);
    return 0;
}

```

Output:

Enter the no. of vertices: 5
 Enter the source vertex of graph: 1
 Enter no. of edges: 6

For edge 1 =>
 Enter source vertex : 1
 Enter destination vertex : 2
 Enter weight : 6

For edge 2 =>
 Enter source vertex : 1
 Enter destination vertex : 3
 Enter weight : 5

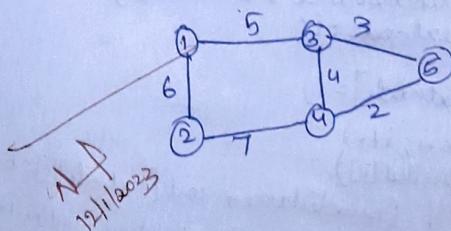
For edge 3 =>
 Enter source vertex: 2
 Enter destination vertex: 4
 Enter weight: 7

For edge 4 =>
 Enter source vertex: 3
 Enter destination vertex: 4
 Enter weight: 4

For edge 5 =>
 Enter source vertex: 3
 Enter destination vertex: 5
 Enter weight: 3

For edge 6 =>
 Enter source vertex: 4
 Enter destination vertex: 5
 Enter weight: 2

Vertices	Distance from source
1	0
2	6
3	5
4	9
5	8



Lab 10

Dijkstra's algorithm

```

void dijkstra(int G[MAX][MAX], int n, int startnode)
{
    int cost[MAX][MAX], distance[MAX];
    int visited[MAX], mindistance, nextnode, i, j;
    for (i=0; i<n; i++)
        for (j=0; j<n; j++)
            if (G[i][j] == 0)
                cost[i][j] = INFINITY;
            else
                cost[i][j] = G[i][j];
    for (i=0; i<n; i++)
    {
        distance[i] = cost[startnode][i];
        pred[i] = startnode;
        visited[i] = 0;
    }
    distance[startnode] = 0;
    visited[startnode] = 1;
    count = 1;
    while (count < n-1)
    {
        mindistance = INFINITY;
        for (i=0; i<n; i++)
            if (distance[i] < mindistance && !visited[i])
            {
                mindistance = distance[i];
                nextnode = i;
            }
        visited[nextnode] = 1;
        for (i=0; i<n; i++)
            if (!visited[i])
                if (mindistance + cost[nextnode][i] < distance[i])
                    distance[i] = mindistance + cost[nextnode][i];
                    pred[i] = nextnode;
        count++;
    }
    for (i=0; i<n; i++)
        if (i != startnode)
        {
            printf ("Distance of %d : %d", i, distance[i]);
            printf ("Path = %d", i);
        }
    }
}

```

```

j=1;
int count=1;
while (j != startnode)
    j = pred[j];
printf ("%d -> %d", j);
count++;

```

```

void main()
{
    int G[MAX][MAX], i, j, n, u;
    printf ("Enter the no. of vertices: ");
    scanf ("%d", &n);
    printf ("Enter adjacency matrix: ");
    for (i=0; i<n; i++)
        for (j=0; j<n; j++)
            scanf ("%d", &G[i][j]);
    printf ("Enter starting node: ");
    scanf ("%d", &u);
    dijkstra(G, n, u);
}

```

Output:

Enter no. of vertices: 5
 Enter adjacency matrix

```

0 3 1 0 0
3 0 7 5 1
1 7 0 2 0
0 5 2 0 7
0 1 0 9 0

```

Enter starting node: 0

Distance of 1 = 3

Path = 1 ← 0

Distance of 2 = 1

Path = 2 ← 0

Distance of 3 = 3

Path = 3 ← 2 ← 0

Distance of 4 = 4

Path = 4 ← 1 ← 0

Lab - 11

Using TCP-IP, write a client - server program where client sends file name & server sends back contents of requested file

Client

```
from socket import *
serverName = '127.0.0.1'
serverPort = 12000
clientSocket = socket (AF_INET, SOCK_STREAM)
clientSocket.connect ((serverName, serverPort))
sentence = input ("Entrez le nom de fichier : ")
clientSocket.send (sentence.encode ())
file = clientSocket.recv (1024).decode ()
print (file)
clientSocket.close ()
```

Server

```
from socket import *
serverName = '127.0.0.1'
serverPort = 12000
serverSocket = socket (AF_INET, SOCK_STREAM)
serverSocket.bind ((serverName, serverPort))
serverSocket.listen (1)
while (1):
    connectionSocket, address = serverSocket.accept ()
    sentence = connectionSocket.recv (1024).decode ()
    file = open (sentence, "r")
    l = file.read (1024)
    connectionSocket.send (l.encode ())
    file.close ()
    connectionSocket.close ()
```

Output:

Enter file name: serverTCP.py

From Server:

```
from socket import *
serverName = '127.0.0.1'
serverPort = 12000
serverSocket = socket (AF_INET, SOCK_STREAM)
serverSocket.bind ((serverName, serverPort))
serverSocket.listen (1)
while (1):
    connectionSocket, address = serverSocket.accept ()
    sentence = connectionSocket.recv (1024).decode ()
    file = open (sentence, "r")
    l = file.read (1024)
    connectionSocket.send (l.encode ())
    file.close ()
    connectionSocket.close ()
```

Lab 12

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

Client

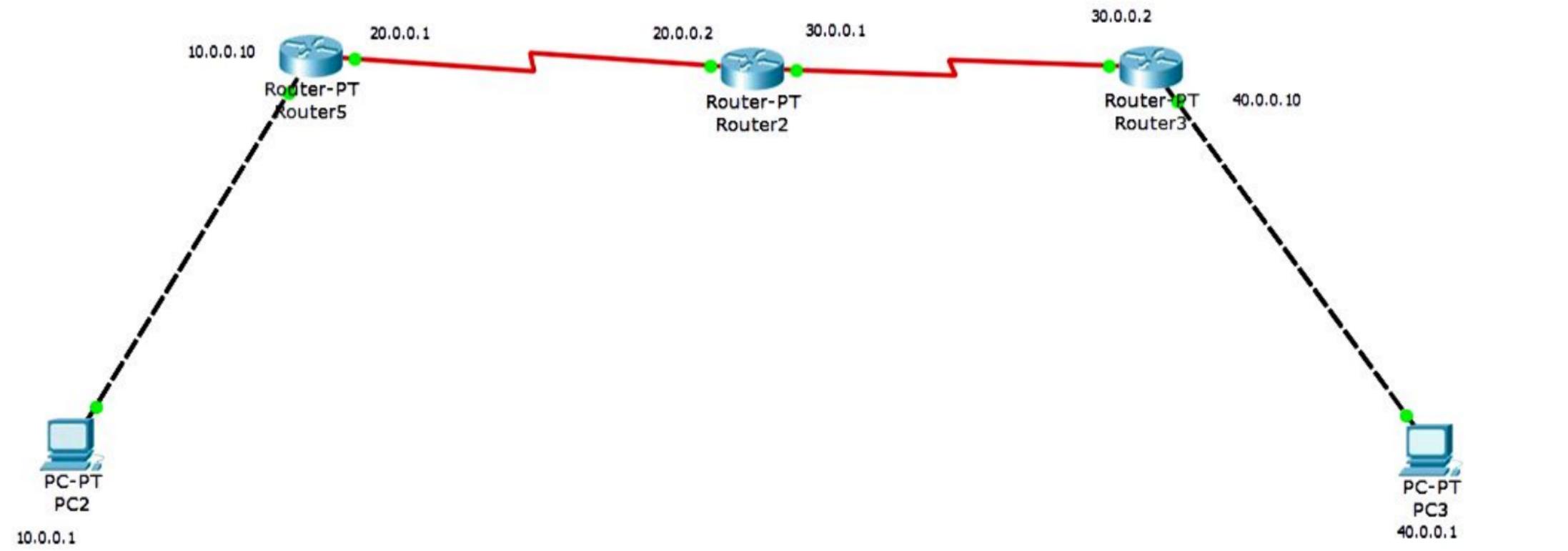
```
from socket import *
serverName = '127.0.0.1'
serverPort = 12000
clientSocket = socket(AF_INET, SOCK_DGRAM)
sentence = input("Enter file name: ")
clientSocket.sendto(sentence.encode("utf-8"), (serverName, serverPort))
file, serverAddress = clientSocket.recvfrom(2048)
print(file.decode("utf-8"))
clientSocket.close()
clientSocket.close()
```

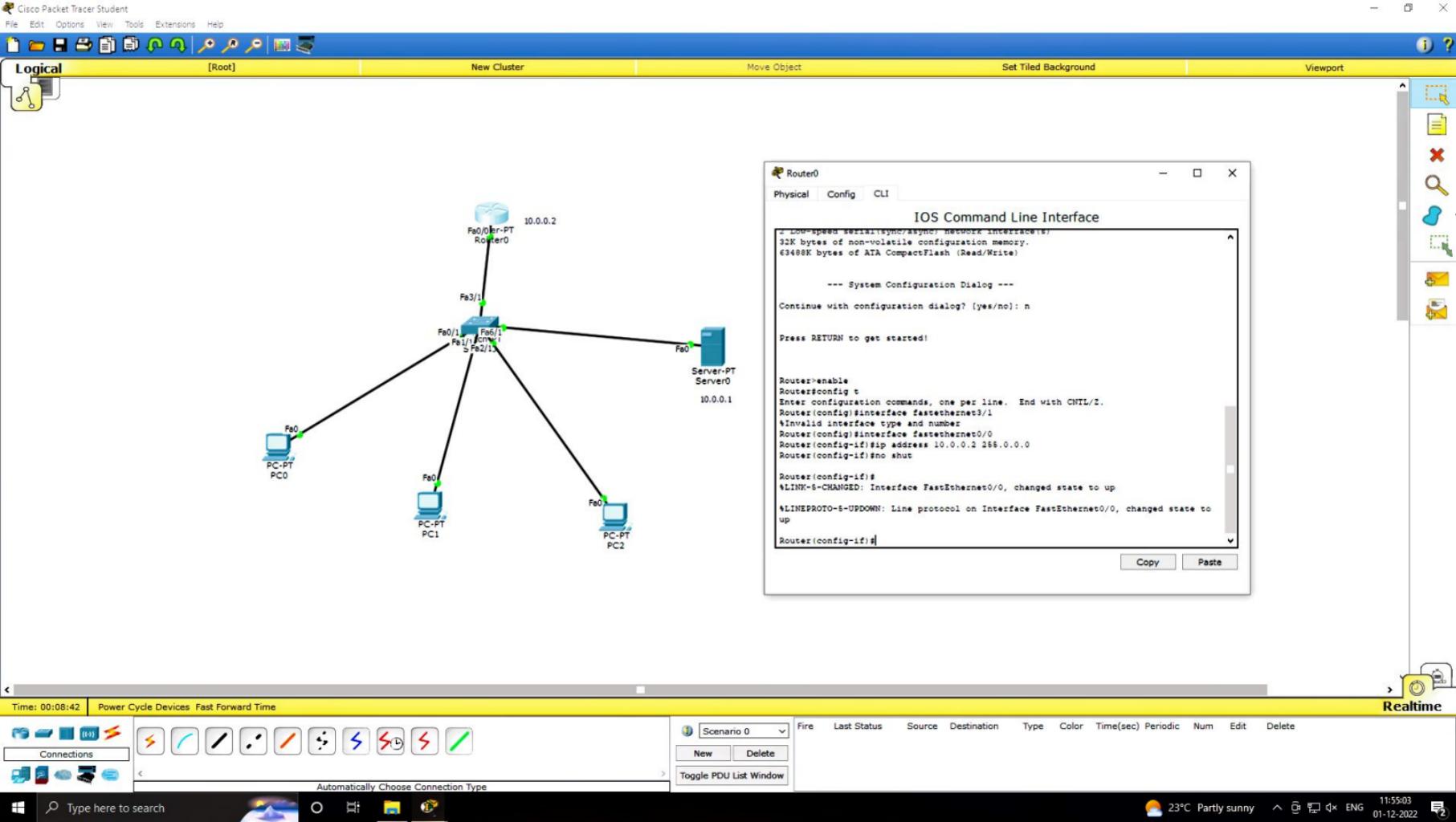
Server

```
from socket import *
serverName = '127.0.0.1'
serverPort = 12000
serverSocket = socket(AF_INET, SOCK_DGRAM)
serverSocket.bind(("127.0.0.1", serverPort))
while (1):
    sentence, clientAddress = serverSocket.recvfrom(2048)
    sentence = sentence.decode("utf-8")
    file = open(sentence, "r")
    l = file.read(2048)
    serverSocket.sendto(l.encode("utf-8"), clientAddress)
    print(sentence)
    file.close()
```

Output

```
# Enter file name: scannerdp.py
from socket import *
serverPort = 12000
serverSocket = socket(AF_INET, SOCK_DGRAM)
serverSocket.bind(("127.0.0.1", serverPort))
while (1):
    sentence, clientAddress = serverSocket.recvfrom(2048)
    fileSentence = serverSocket.decode("utf-8")
    file = open(sentence, "r")
    l = file.read(2048)
    serverSocket.sendto(l.encode("utf-8"), clientAddress)
    print(sentence)
    file.close()
```







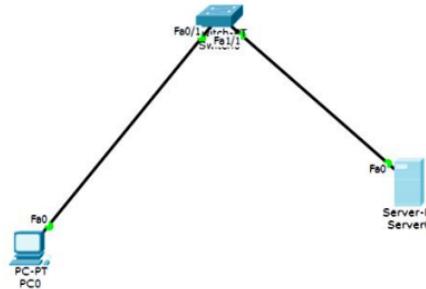
[Root]

New Cluster

Move Object

Set Tiled Background

Viewport



The screenshot shows the 'DNS' configuration page under the 'Custom Interface' tab. On the left, a sidebar lists various services: SERVICES, HTTP, DHCP, DHCPv6, TFTP, DNS, SYSLOG, AAA, NTP, EMAIL, and FTP. The main area has a title 'DNS' and a status bar indicating 'DNS Service' is 'On'. Below this are sections for 'Resource Records' (Name: cn, Type: A Record) and 'Address' (cn). At the bottom are buttons for 'Add', 'Save', and 'Remove', and a table showing the current record:

No.	Name	Type	Detail
0	cn	A Record	10.0.0.2

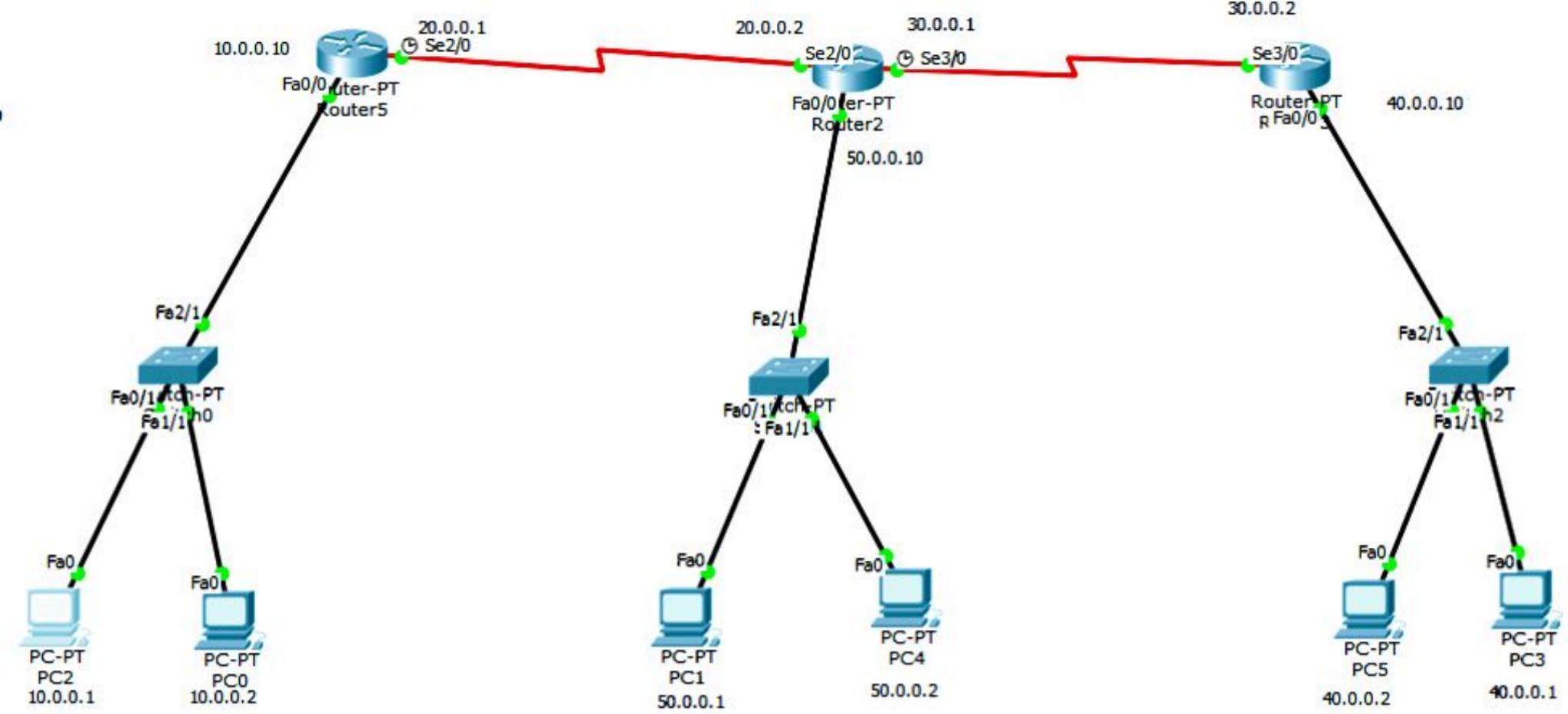
At the very bottom is a button labeled 'DNS Cache'.

Time: 00:08:18 Power Cycle Devices Fast Forward Time

	Scenario 0	Fire	Last Status	Source	Destination	Type	Color	Time(sec)	Periodic	Num	Edit	Delete
--	------------	------	-------------	--------	-------------	------	-------	-----------	----------	-----	------	--------

[New](#) [Delete](#)





Enter the data:10000100000010001

Enter the key:10101

CRC=1101

Dataword=100001000000100011101

Process returned 0 (0x0) execution time : 37.143 s

Press any key to continue.

Enter the bucket capacity: 50

Enter the outflow rate: 10

1.Insert

2.Exit

Enter choice: 1

Enter the packet size: 40

40

After outflow: 30

1.Insert

2.Exit

Enter choice: 1

Enter the packet size: 5

35

After outflow: 25

1.Insert

2.Exit

Enter choice: 1

Enter the packet size: 30

Bucket overflow1.Insert

2.Exit

Enter choice: 2

No more inputs. Program exited

Enter the number of vertices: 4

Enter the source vertex of the graph: 1

Enter no. of edges: 5

For edge 1=>

Enter source vertex :1

Enter destination vertex :2

Enter weight :4

For edge 2=>

Enter source vertex :1

Enter destination vertex :3

Enter weight :5

For edge 3=>

Enter source vertex :3

Enter destination vertex :2

Enter weight :7

For edge 4=>

Enter source vertex :2

Enter destination vertex :4

Enter weight :7

For edge 5=>

Enter source vertex :4

Enter destination vertex :3

Enter weight :-15

NEGATIVE CYCLE PRESENT..!!

Enter the no. of vertices: 5

Enter the adjacency matrix:

0 3 1 0 0

3 0 7 5 1

1 7 0 2 0

0 5 2 0 7

0 1 0 7 0

Enter the starting node: 0

Distance of 1 = 3

Path = 1 <-0

Distance of 2 = 1

Path = 2 <-0

Distance of 3 = 3

Path = 3 <-2 <-0

Distance of 4 = 4

Path = 4 <-1 <-0

C:\Users\Srikanth\PycharmProjects\pythonProject1\Scripts\python.exe C:\U

Enter file name: *ServerTCP.py*

From Server:

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

Process finished with exit code 0

C:\Users\Srikanth\PycharmProjects\pythonProject1\Scripts\python.exe

Enter file name: serverudp.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")
    l=file.read(2048)
    serverSocket.sendto(bytes(l,"utf-8"),clientAddress)
    print ('\nSent contents of ', end = ' ')
    print (sentence)
# for i in sentence:
#     print (str(i), end = '')
    file.close()
```

Process finished with exit code 0