VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"JnanaSangama", Belgaum -590014, Karnataka.

LAB REPORT
on

COMPUTER NETWORKS

Submitted by

NIKHIL SRIKANTH (1BM20CS096)

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
Oct 2022-Feb 2023

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

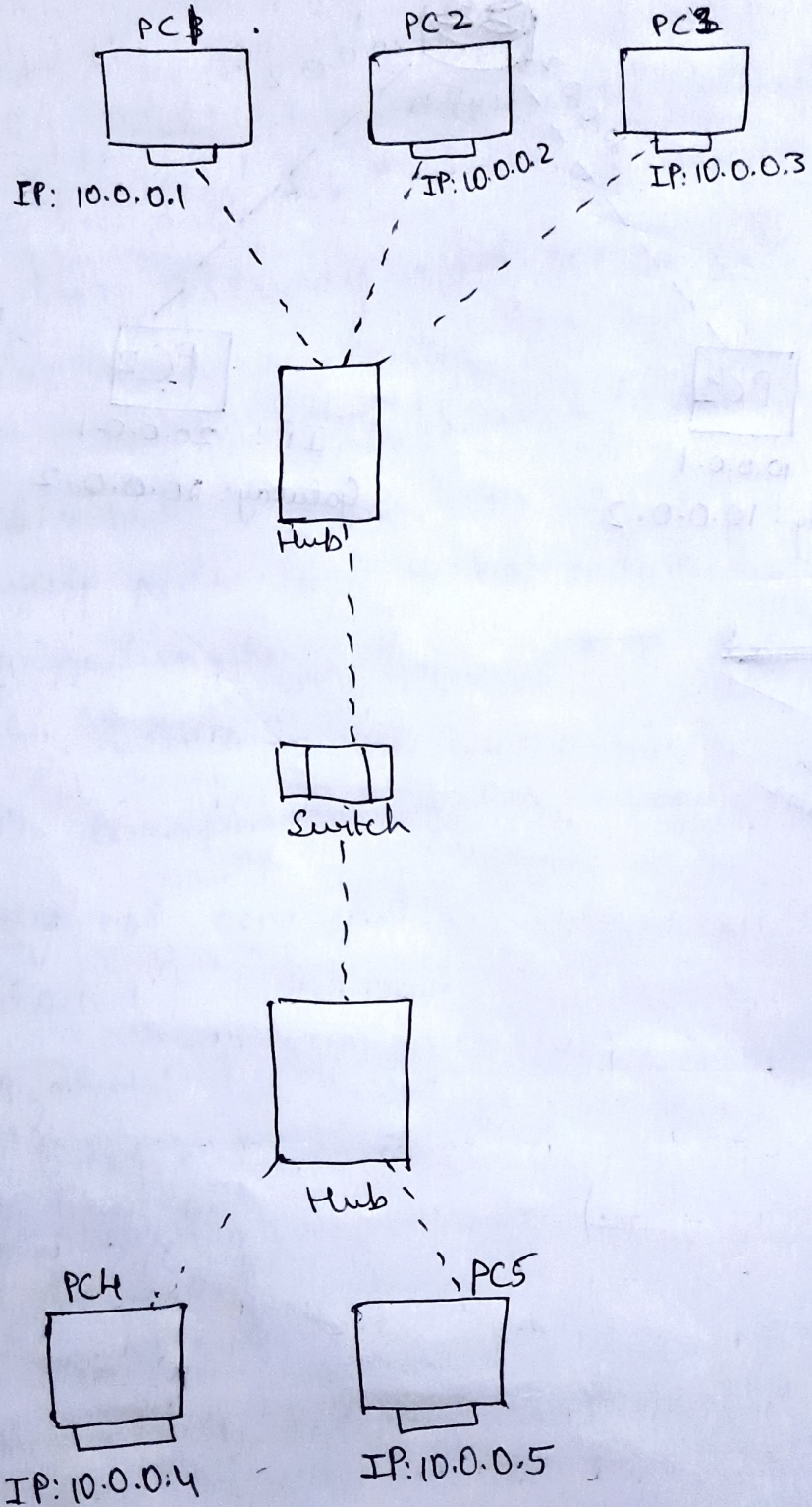
Procedure:
- End devices are connected to the hub ~~& switch~~
- The hubs are interconnected via a switch
- IP addresses of the end devices are set
- Connections between all of them are checked, if it is working
- They are checked by pinging a message between 2 end devices.
- Once verified, a simple PDU is sent 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 unicasting between the source & destination via the switch.
- If a receiver host isn't connected to a internetwork, ~~the~~ a message cannot be pinged & hence response will be timed out.

Nivedhini 13/10/2022

Topology (Expt 1):



Expt 2

17/11/22

Aim: TO configure IP address to Routers 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 succesfully 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~~ 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) 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

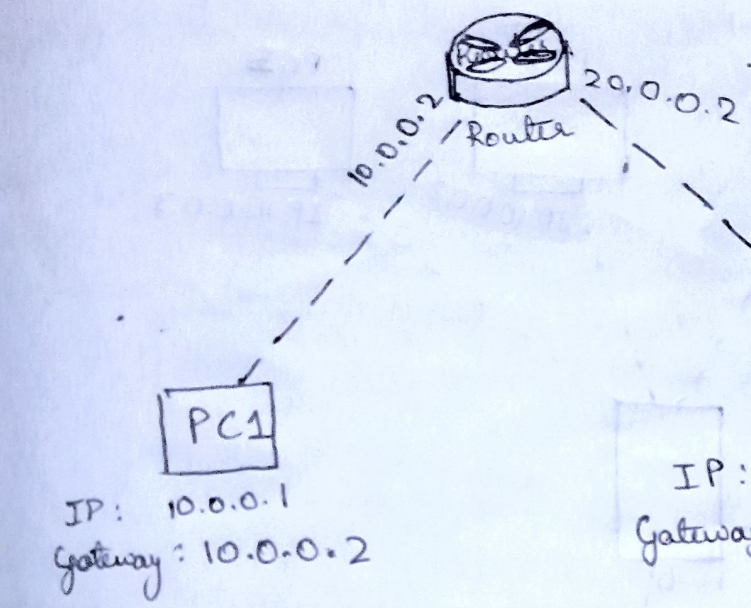
From: 10.0.0.1

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:



~~Gateway and~~

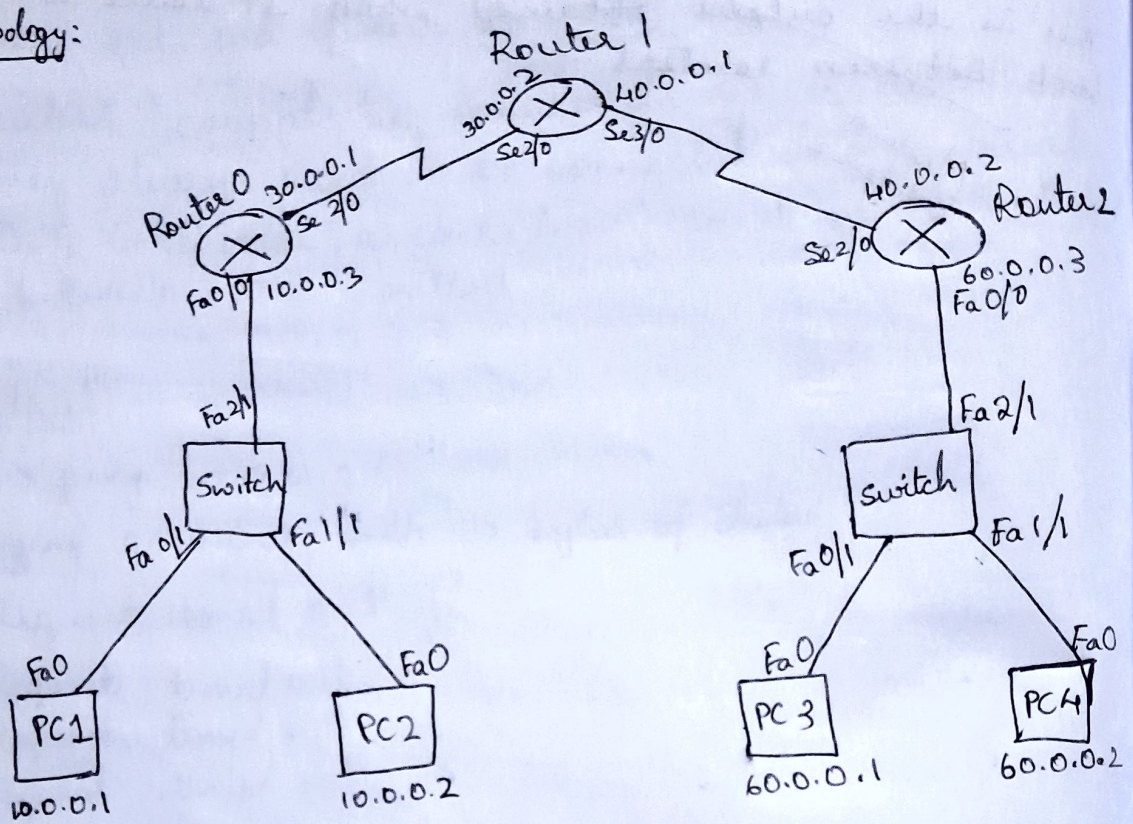
Neelima
19/11/2022

Lab-4

1/12/22

Aim: To configure default route to the routers

Topology:



Procedure:

- Construct topology using enddevices, 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
```

0.0.0.0 — any destination id; 0.0.0.0 — any subnet mask; 40.0.0.2 — next hop

- The connections
one end-device

Observations:

- The packets
of the end
timed out)
- When a def
is set), the
not recogniz
destination
- Only 50% pa
to a router
from a s
end-device
- a default
other route
to the nex
- While pingi
1 of pack
because
destination
Then the r
to the reco

Output:

```
PC > ping 30.0
Pinging 30.0.0.2
Request timed
Request timed
Request timed
Request timed
```

• 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% ~~pact~~ 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 a default route to send packets to two other routers. Hence 50% packets are transfers to the next interface & 50% are sent back.

• While pinging to the destination end-device, ~~[redacted]~~ of packets are sent first via the switch because ~~[redacted]~~ packets are sent back, to the switch by the destination to confirm the destination end-device. Then the rest of the packets are sent to the recognized destination end-device.

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 40.0.0.

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

PC > ping 40.0.0.1
Pinging 40.0.0.1 with 32 bytes of data:
Reply from 10.0.0.3: Destination host unreachable
Reply from 10.0.0.3: Destination host unreachable
Reply from 10.0.0.3: Destination host unreachable
Reply from 10.0.0.3: Destination host unreachable

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

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

PC > ping 40.0.0.2
Pinging 40.0.0.2 with 32 bytes of data:
Request timed out
Reply from 40.0.0.2: bytes=32 time=11ms TTL=253
Request timed out
Reply from 40.0.0.2: bytes=32 time=2ms TTL=253

Ping statistics for 40.0.0.2:
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

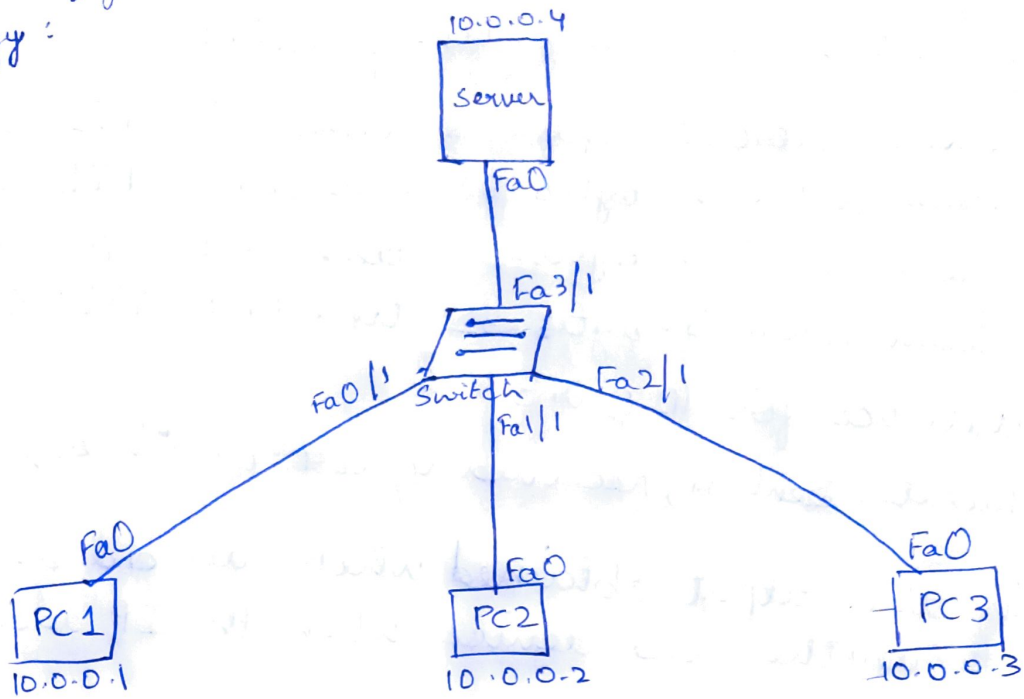
1/12

Lab 6

15/12/22

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 to end devices when the DHCP is initiated in the server
- RARP is used to assign the IP address of a device if 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)
```

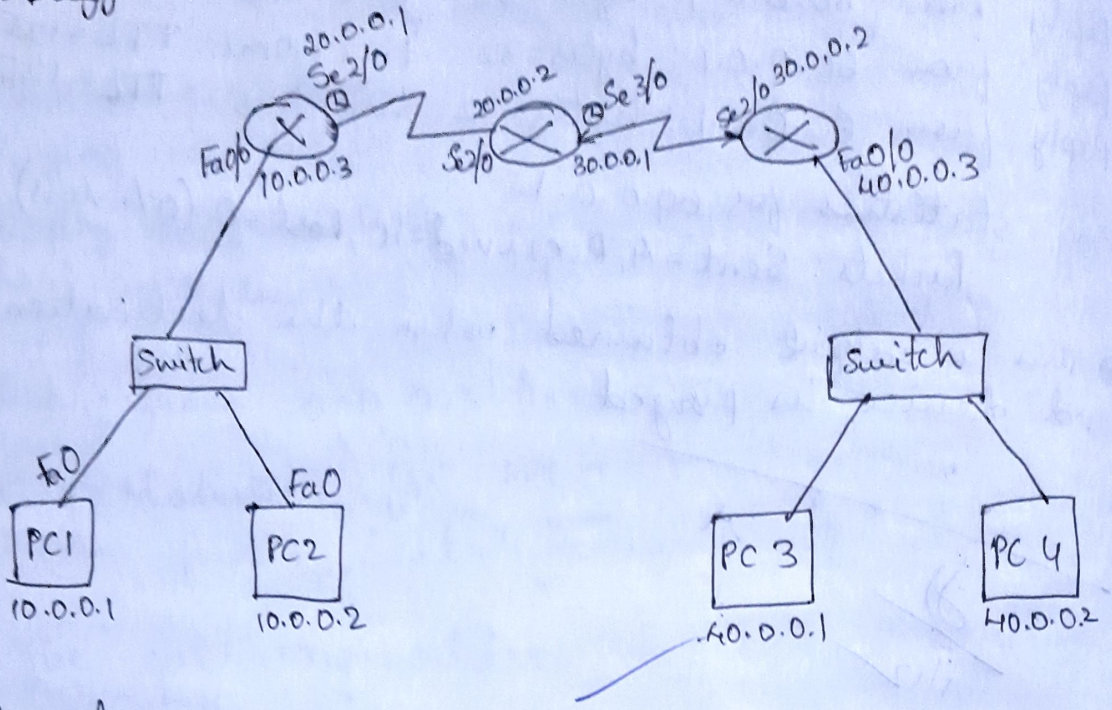
This is the output obtained when an end device pings to another end device when the IP address is set via DHCP

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

Lab-5

Aim: Configure RIP (Routing Information Protocol) in Routers

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 addresses of the routers are set)

ip address — subnet mask

```
Router (config-if) # ip address 20.0.0.1 255.0.0.0
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 } connected
Router (config-route) # network 20.0.0.0 } 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 neither ip route nor ~~nor~~ 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
Reply from 10.0.0.3: Destination host unreachable
Reply from 10.0.0.3: Destination host unreachable
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)
```

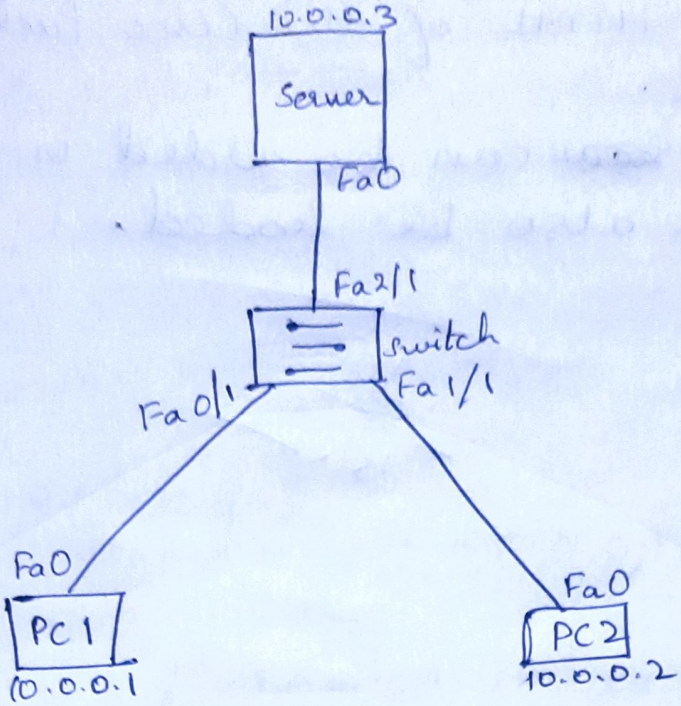
Note: While pinging via a switch, the first packet is sent to confirm the destination end device. Hence the first packet results in a timed out situation. 4 packets are sent & 3 packets are received.

8/12

Lab-6

Aim: How to configure WEB server & DNS server

Topology:



Procedure:

- The above topology is contructed. 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 records is added & saved.
- The web browser of an end device is opened & searched for the ~~url~~ created url

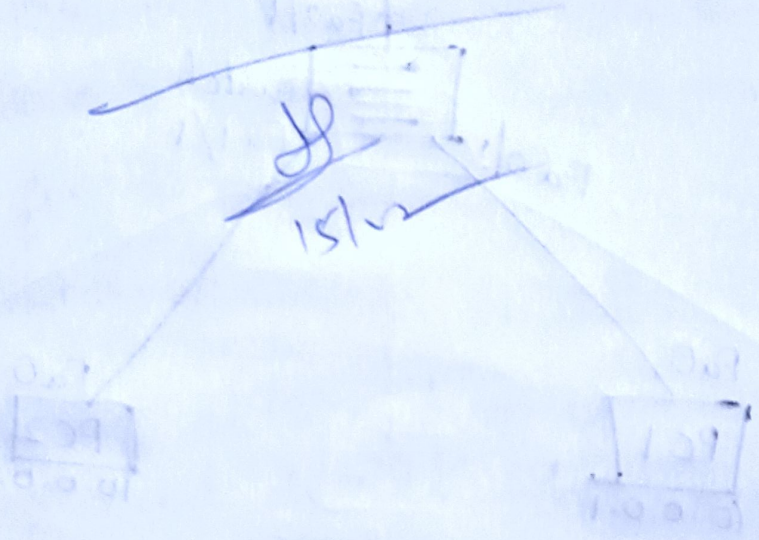
Observations:

- 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 be added in the HTTP window & also be loaded.



Lab-7

29-12-22

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[] = {1,0,0,0,1,0,0,0,0,0,0,1,0,0,0,0,1};
  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
1

1110110000011000111
0010000000100000000

12/12/2023

Lab 8 5/1/23

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
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

29/11/2023

Lab 9 12-1-23

Bellman-Ford algorithm

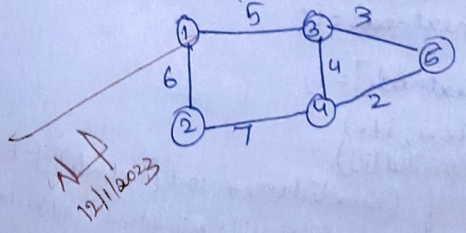
```
# include <~~stdio.h~~ iostream>
# include <~~stdlib.h~~ bits/stdc++.h>
# define MAX 10
~~int Bellman(int G[20][20], int V, int E, int edge[20][20])~~
using namespace std;
typedef struct edge
 { int src,dest,wt;
 }edge;

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 << "\t" << dis[i];
}
```

```
int main ()
{ int nv, ne, src;
  edge e[MAX];
  cout<< "Enter the 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
```

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



12/1/2023

Lab 10

Djikstra's algorithm

```
void dijkstra(int G[MAX][MAX], int n, int startnode)
{ int cost[MAX][MAX], distance[MAX], pred[MAX];
  int visited[MAX], count, 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
             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 = pred[j]
      printf("<-%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 7 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("Enter file name: ")
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, addr = 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, addr = 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-server 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)
sentance = input("Enter file name: ")
clientsocket.sendto(bytes(sentence, "utf-8"), (serverName, serverPort))
file, serverAddress = clientsocket.recvfrom(2048)
print(filecontents.decode("utf-8"))

clientsocket.close()
clientsocket.close()
```

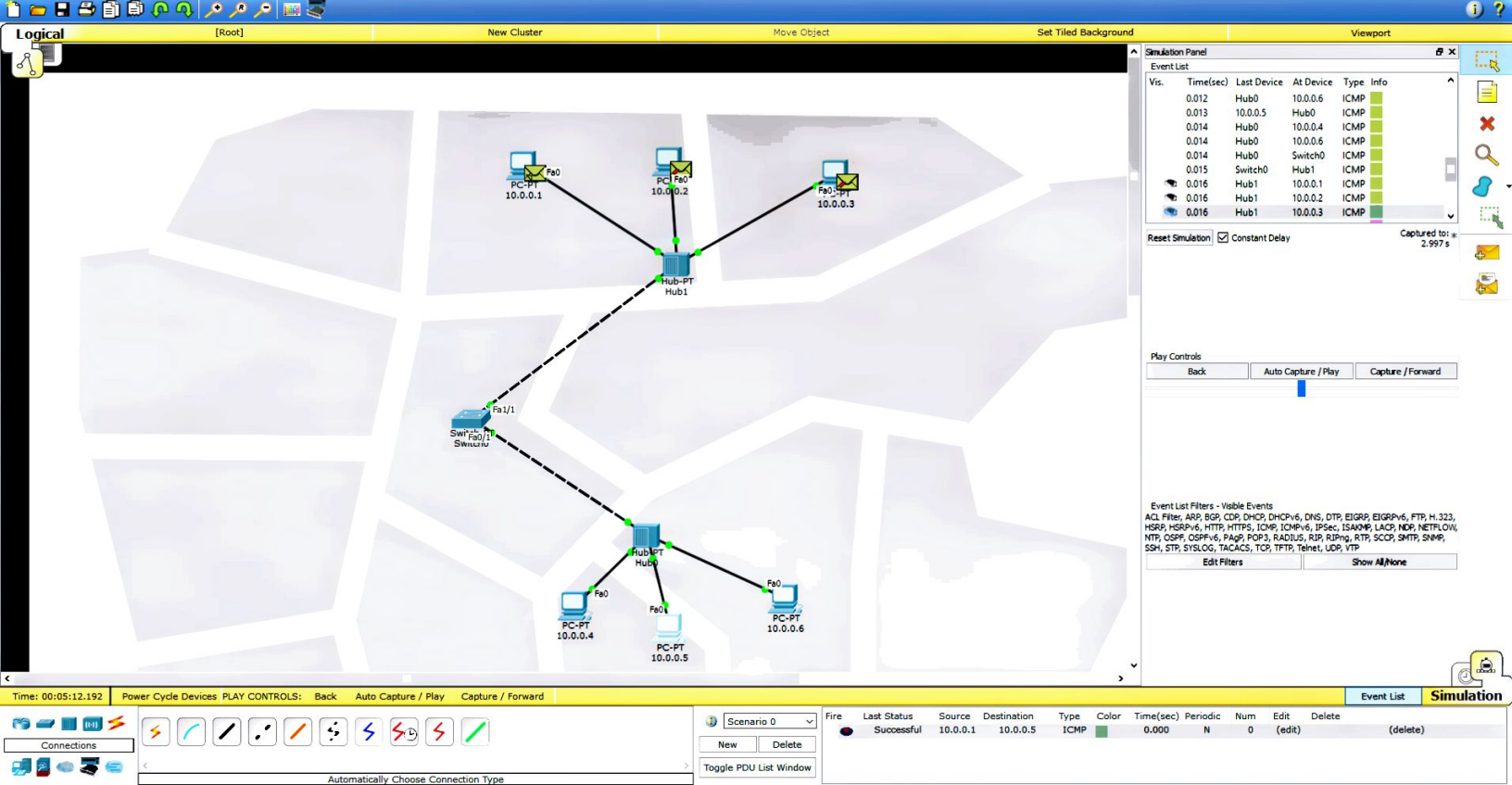
Server

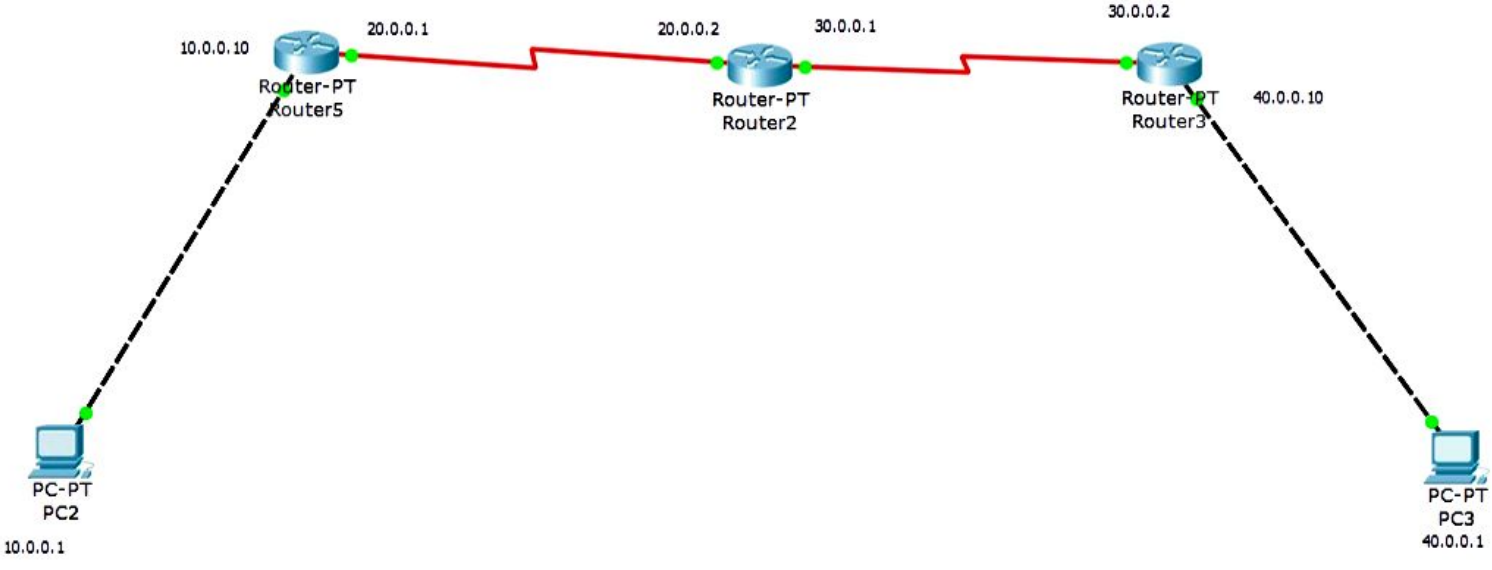
```
from socket import *
serverName = '127.0.0.1'
serverPort = 12000
serversocket = socket(AF_INET, SOCK_DGRAM)
serversocket.bind(("127.0.0.1", serverPort))
while (1):
    sentence, clientAddress = serversocket.recvfrom(2048)
    sentence = sentence.decode("utf-8")
    file = open(sentence, "r")
   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

    file.close()
```

Output

Enter file name: serverudp.py

```
from socket import *
serverPort = 12000
serversocket = socket(AF_INET, DGRAM)
serversocket.bind(("127.0.0.1", serverPort))
while (1):
    sentance, clientAddress = serversocket.recvfrom(2048)
    sentence = serversocket.decode("utf-8")
    file = open(sentance, "r")
    l = file.read(2048)
    serversocket.sendto(bytes(l, "utf-8"), clientAddress))
    print(sentence)
    file.close().
```

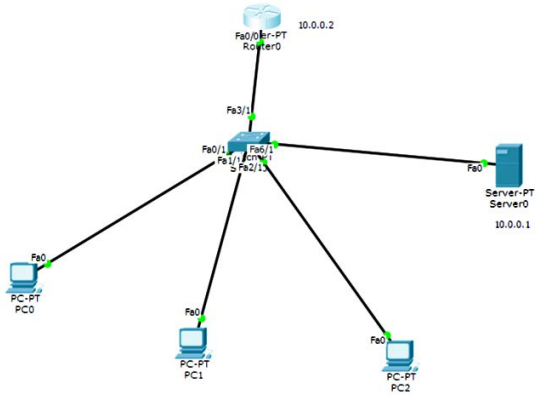

Logical
[Root]
New Cluster
Move Object
Set Tiled Background
Viewport
PC-PT
10.0.0.1
Fa0
PC-PT
10.0.0.2
Fa0
PC-PT
10.0.0.3
Fa0
Hub-PT
Hub1
Fa1/1
Switch0
Fa0/1
Hub-PT
Hub0
Fa0
PC-PT
10.0.0.4
Fa0
PC-PT
10.0.0.5
Fa0
PC-PT
10.0.0.6
Simulation Panel
Event List
Vis.
Time(sec)
Last Device
At Device
Type
Info
0.012 Hub0 10.0.0.6 ICMP
0.013 10.0.0.5 Hub0 ICMP
0.014 Hub0 10.0.0.4 ICMP
0.014 Hub0 10.0.0.6 ICMP
0.014 Hub0 Switch0 ICMP
0.015 Switch0 Hub1 ICMP
0.016 Hub1 10.0.0.1 ICMP
0.016 Hub1 10.0.0.2 ICMP
0.016 Hub1 10.0.0.3 ICMP
Reset Simulation
Constant Delay
Captured to: 2.997 s
Play Controls
Back
Auto Capture / Play
Capture / Forward
Event List Filters - Visible Events
ACL Filter, ARP, BGP, CDP, DHCP, DHCPv6, DNS, DTP, EIGRP, EIGRPv6, FTP, H.323, HSRP, HSRPv6, HTTP, HTTPS, ICMP, ICMPv6, IPSec, ISAKMP, LACP, NDP, NETFLOW, NTP, OSPF, OSPFv6, PAgP, POP3, RADIUS, RIP, RIPng, RTP, SCCP, SMTP, SNMP, SSH, STP, SYSLOG, TACACS, TCP, TFTP, Telnet, UDP, VTP
Edit Filters
Show All/None
Time: 00:05:12.192
Power Cycle Devices
PLAY CONTROLS:
Back
Auto Capture / Play
Capture / Forward
Event List
Simulation
Connections
Automatically Choose Connection Type
Scenario 0
New
Delete
Toggle PDU List Window
Fire
Last Status
Source
Destination
Type
Color
Time(sec)
Periodic
Num
Edit
Delete
Successful 10.0.0.1 10.0.0.5 ICMP 0.000 N 0 (edit) (delete)


10.0.0.10
20.0.0.1
20.0.0.2
30.0.0.1
30.0.0.2
Router-PT
Router5
Router-PT
Router2
Router-PT
Router3
40.0.0.10
PC-PT
PC2
10.0.0.1
PC-PT
PC3
40.0.0.1

Cisco Packet Tracer Student

File Edit Options View Tools Extensions Help

Logical | [Root] | New Cluster | Move Object | Set Tiled Background | Viewport



Router0

Physical | Config | CLI

IOS Command Line Interface

```
2 Low-speed serial(sync/async) network interface(s)
32K bytes of non-volatile configuration memory.
63488K bytes of ATA CompactFlash (Read/Write)


         --- System Configuration Dialog ---

Continue with configuration dialog? [yes/no]: n


Press RETURN to get started!



Router>enable
Router#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)#interface fastethernet3/1
%Invalid interface type and number
Router(config)#interface fastethernet0/0
Router(config-if)#ip address 10.0.0.2 255.0.0.0
Router(config-if)#no shut

Router(config-if)#
%LINK-5-CHANGED: Interface FastEthernet0/0, changed state to up

%LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet0/0, changed state to
up

Router(config-if)#
```

Copy | Paste

Time: 00:08:42 | Power Cycle Devices | Fast Forward Time | Realtime

Connections

Automatically Choose Connection Type

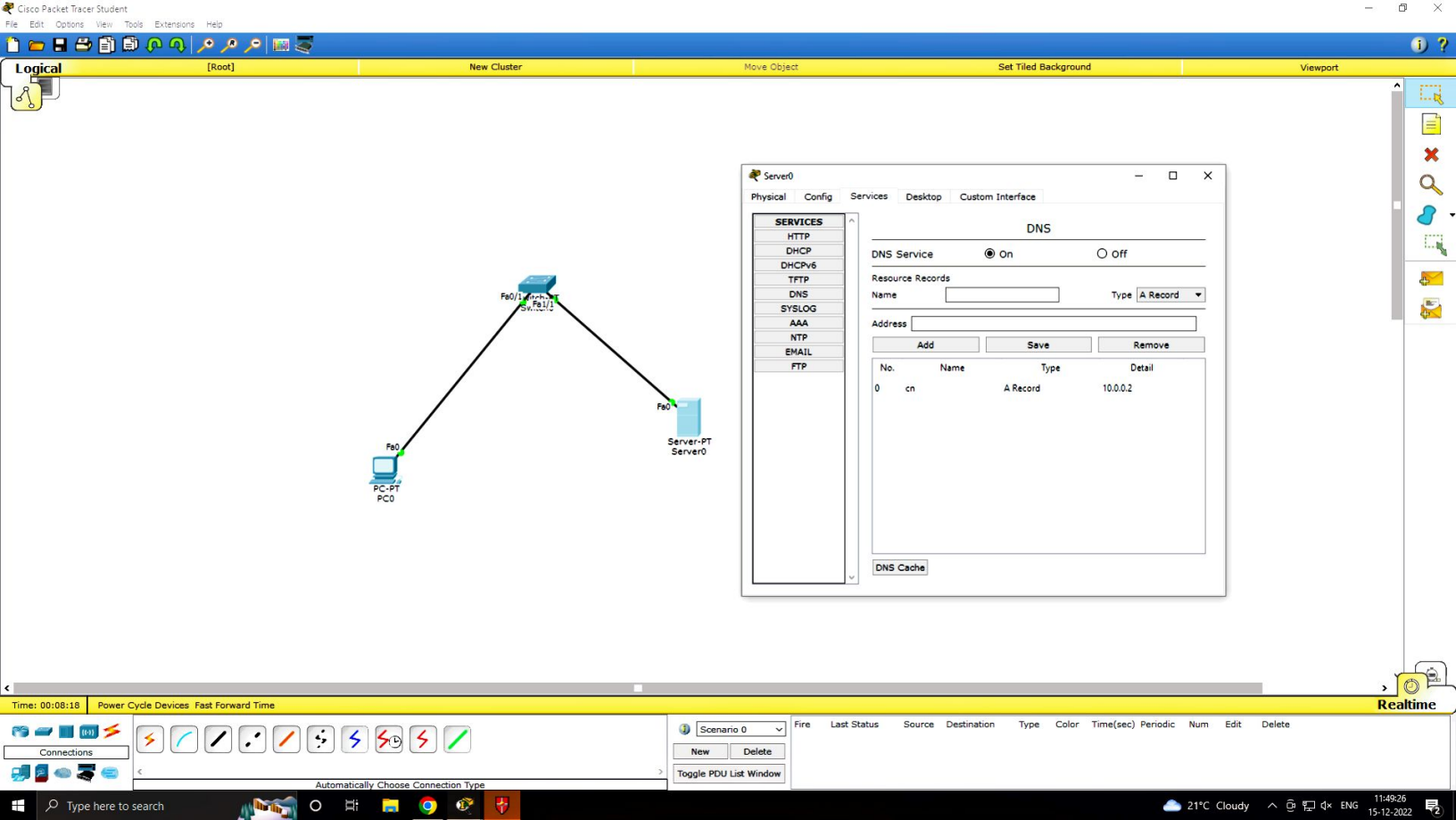
Scenario 0 | New | Delete | Toggle PDU List Window

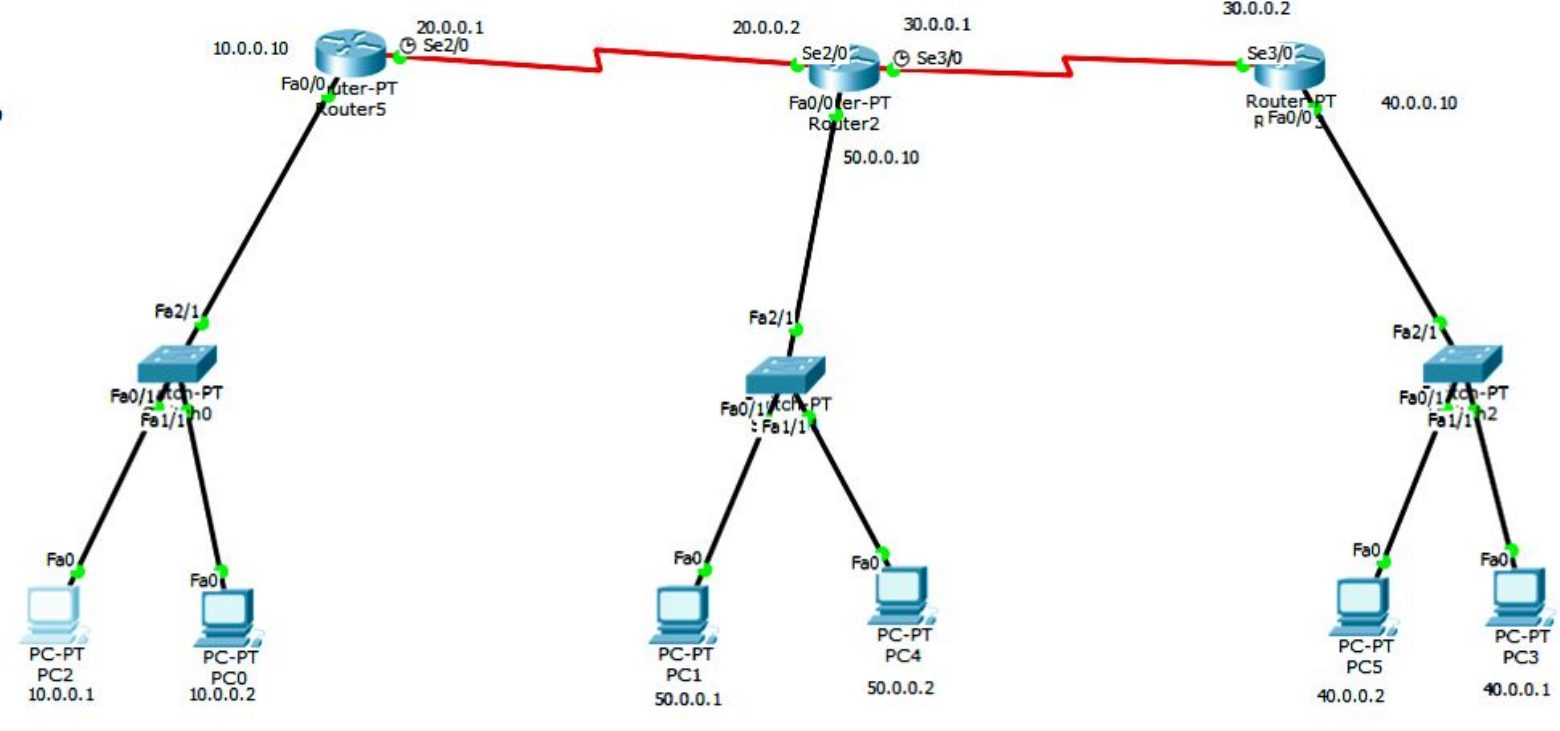
Fire | Last Status | Source | Destination | Type | Color | Time(sec) | Periodic | Num | Edit | Delete

Type here to search

23°C Partly sunny | ENG | 11:55:03 | 01-12-2022


20.0.0.1
Se2/0
20.0.0.2
Se2/0
30.0.0.1
Se3/0
30.0.0.2
Se3/0
10.0.0.10
Fa0/0
Router-PT
Router0
Router-PT
Router1
Router2
Fa0/0
40.0.0.10
Fa0
PC-PT
PC0
Fa0
PC-PT
PC1
40.0.0.1


Cisco Packet Tracer Student
File Edit Options View Tools Extensions Help
Logical
[Root]
New Cluster
Move Object
Set Tiled Background
Viewport
Fa0/1
Fa1/1
Fa0
Server-PT
Server0
Fa0
PC-PT
PC0
Server0
Physical
Config
Services
Desktop
Custom Interface
SERVICES
HTTP
DHCP
DHCPv6
TFTP
DNS
SYSLOG
AAA
NTP
EMAIL
FTP
DNS
DNS Service
On
Off
Resource Records
Name
Type
A Record
Address
Add
Save
Remove
No.
Name
Type
Detail
0
cn
A Record
10.0.0.2
DNS Cache
Time: 00:08:18
Power Cycle Devices
Fast Forward Time
Realtime
Connections
Automatically Choose Connection Type
Scenario 0
New
Delete
Toggle PDU List Window
Fire
Last Status
Source
Destination
Type
Color
Time(sec)
Periodic
Num
Edit
Delete
Type here to search
21°C Cloudy
ENG
11:49:26
15-12-2022


10.0.0.10
20.0.0.1
Se2/0
Fa0/0
Router-PT
Router5
20.0.0.2
Se2/0
30.0.0.1
Se3/0
Fa0/0
Router-PT
Router2
30.0.0.2
Se3/0
Router-PT
Fa0/0
40.0.0.10
50.0.0.10
Fa2/1
Fa0/1
Fa1/1
Fa0
PC-PT
PC2
10.0.0.1
PC-PT
PC0
10.0.0.2
PC-PT
PC1
50.0.0.1
PC-PT
PC4
50.0.0.2
PC-PT
PC5
40.0.0.2
PC-PT
PC3
40.0.0.1

```
Enter the data:1000010000010001
Enter the key:10101
CRC=1101
Dataword=10000100000100011101
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
```