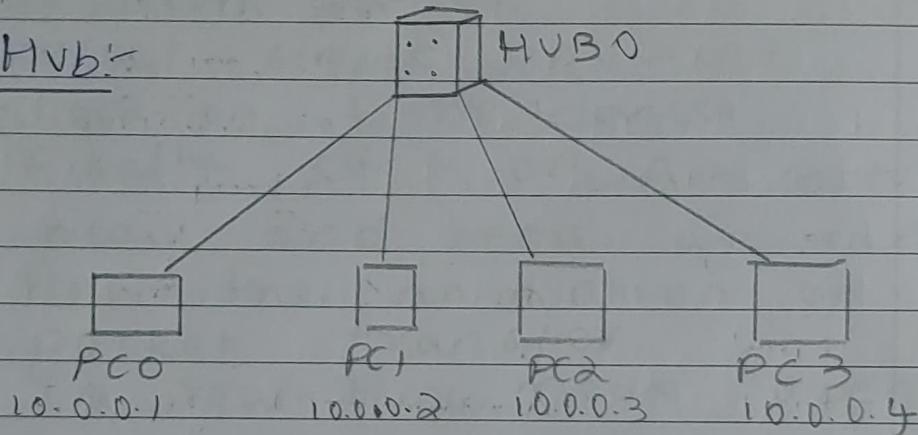
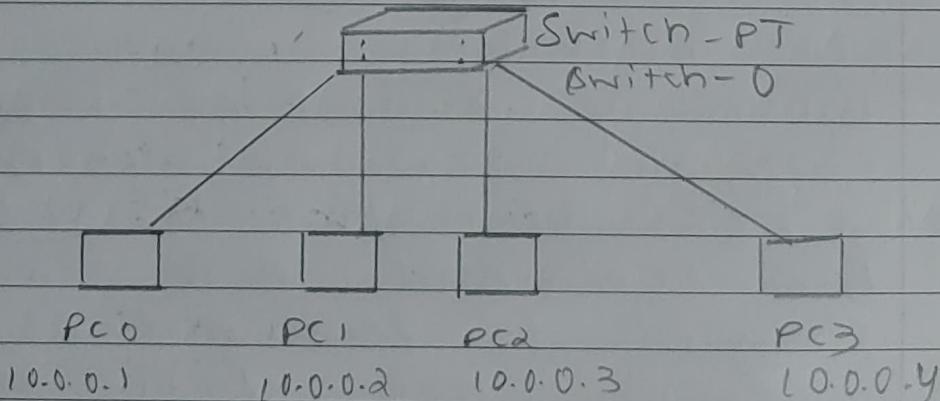


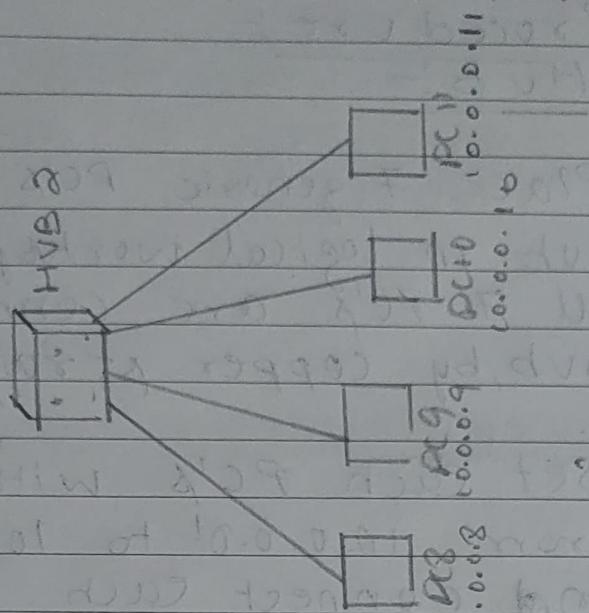
10/11/22

EXPERIMENT - I

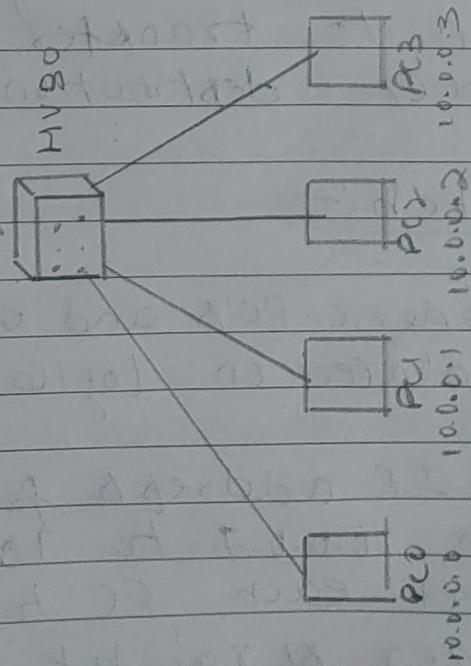
Aim: - Creating a topology and simulate sending a simple PDU from source to destination using hub and switch as connecting devices.

Topology:-Hub:-Switch:-

Hybrid:



Execution



Procedure :-

* Hub :-

- Place 7 generic PC's and 1 generic hub in logical workspace and all 7 PC's are connected to hub, by copper straight wire.
- Set each PC's with IP addresses from 10.0.0.0 to 10.0.0.6 respectively. And connect each PC to hub by copper straight wire.
- A simple PDU is placed on any 2 devices and message/packet passing can be seen in simulation mode, by clicking autocapture mode.
- In realtime mode a command prompt is opened for certain PC and following command is given to transfer message
PING destination-IP-address

* Switch :-

- 4 generic PC's and one generic switch is placed on logical workspace.
- set IP address for each PC from 10.0.0.7 to 10.0.0.10 and connect each PC to switch using copper straight wire.

- In simulation mode after placing simple PDU to any 2 PC's, click auto-capture and packet transfer can be seen.
- In real-time mode click on any PC and open command prompt and type 'PING dest.IP' to send message.
- * Hybrid :-
- 12 PC's, 3-hub, 1-switch all generic are placed onto logical workp-
- 3-generic hubs are connected to switch via using copper crossover wire and 12 PC's are connected to 3 hub, 4 PC each using copper straight wire after assigning IP address for each PC from 10.0.0.0 to 10.0.0.11 respectively.
- After selecting 2 PC's from different hub with simple-PDU and clicking on auto-capture, packet passing simulation can be seen in simulation mode.

→ In real-time mode open command prompt by clicking any PC → Services → command prompt and type 'PING dest. IP-address' to send packet.

Observations:-

* Hub :-

Learning outcome :-

- After source sends message to hub it's broadcasted to all end devices but only destination device reads and send response back to hub for source to get response.
- Hub establishes connection to end-devices quickly and signals by green-light.

Result :-

PING 10.0.0.3

PINGING 10.0.0.3 with 32 bytes of data

REPLY FROM 10.0.0.3 : bytes=32 time=0ms

PING STATISTICS FOR 10.0.0.3 :

"Details of how many packets sent and received."

* Switch :-

learning operation:-

- Unlike hub, switch does not give green signal immediately but takes some amount of time called learning time and the packets can be sent once green signal is generated.
- Initially switch also broadcasts for all end-devices and the next time the communication happens and message passing happens only between source and destination devices.

Result :-

PING 10.0.0.5

PINGING 10.0.0.5 with 32 bytes of data
:

PING STATISTICS FOR 10.0.0.3:

"Details of how many packets sent and received."

* Hybrid :-

Learning outcome:-

- Message sent by one PC of one hub to ~~all~~ ip sent to ~~all~~ hub destination switch on hub which broadcast to all devices of that hub and

only destined end-devices sends back response to source of other hub.

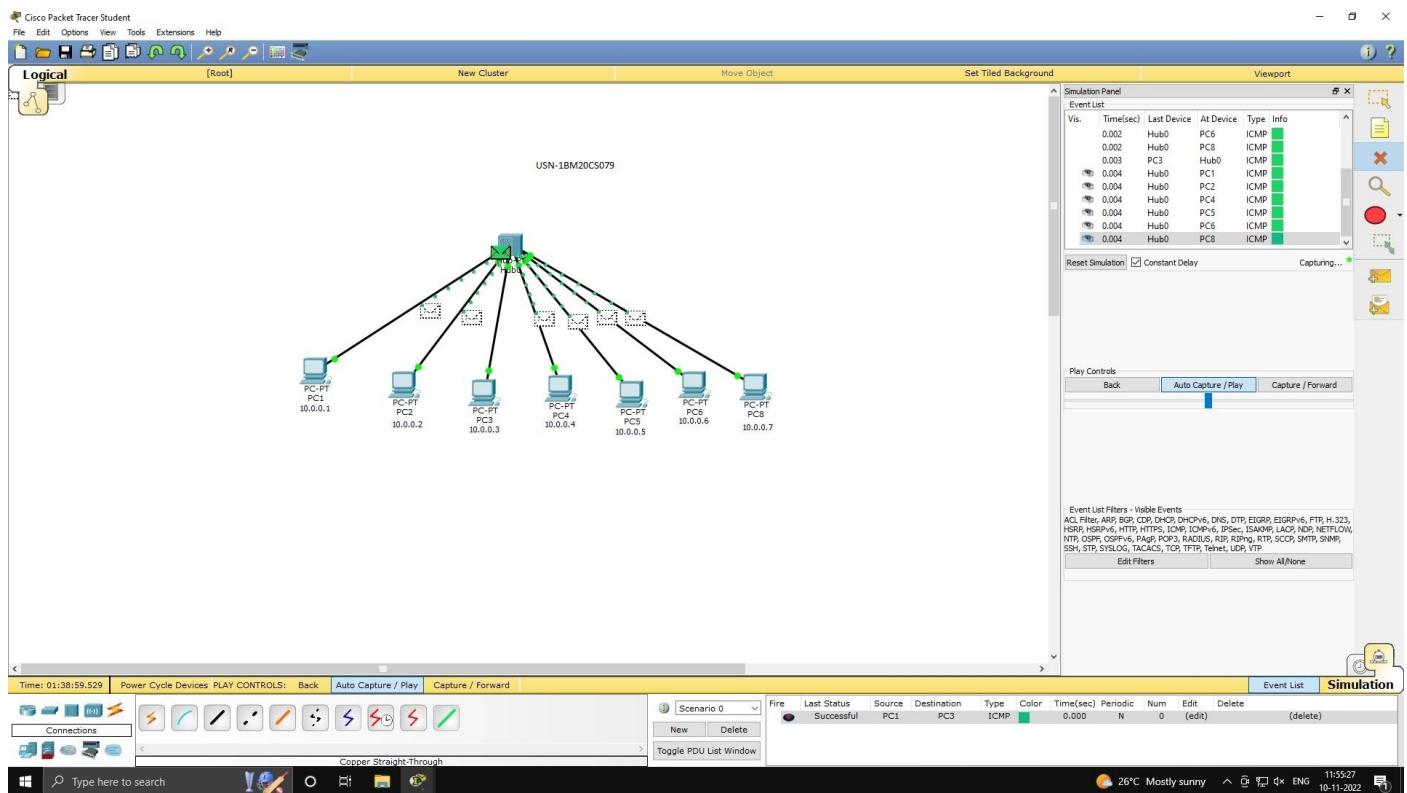
Result:

PING 10.0.0.4

PINGING 10.0.0.4 with 32 bytes of data
REPLY from 10.0.0.4: bytes = 32

PING STATISTICS FOR 10.0.0.4:

"Details of number of packets sent and received"



PC1

Physical Config Desktop Custom Interface

Command Prompt X

```
Packet Tracer PC Command Line 1.0
PC>PING 10.0.0.4

Pinging 10.0.0.4 with 32 bytes of data:

Reply from 10.0.0.4: bytes=32 time=0ms TTL=128

Ping statistics for 10.0.0.4:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 0ms, Maximum = 0ms, Average = 0ms

PC>|
```

USN-1BM20CS079

10.0.0.7
PC9

Physical Config Desktop Custom Interface

Command Prompt X

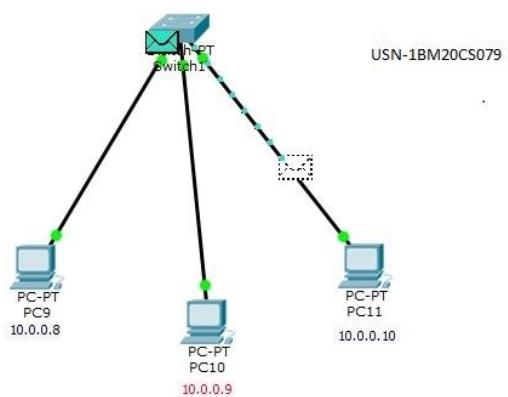
```
Packet Tracer PC Command Line 1.0
PC>PING 10.0.0.10

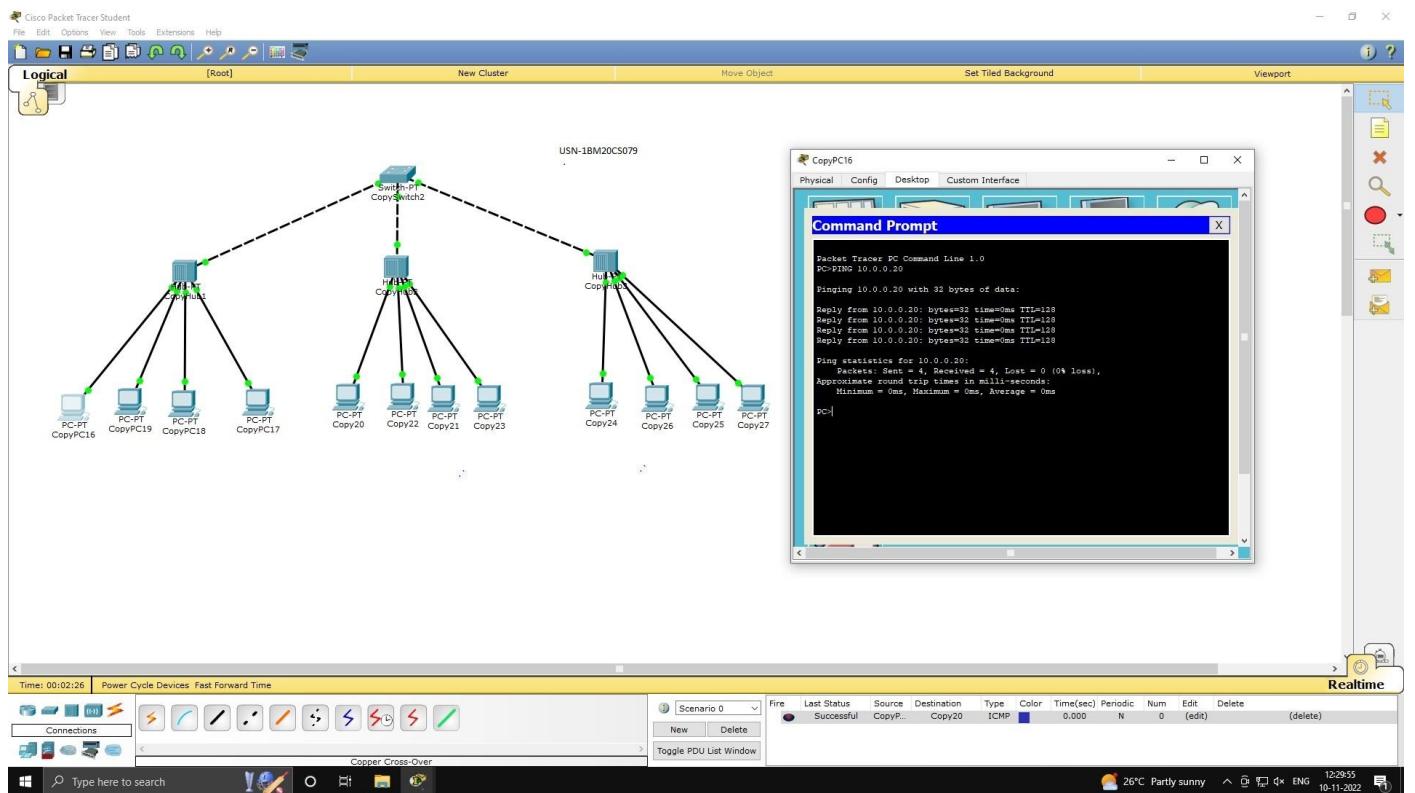
Pinging 10.0.0.10 with 32 bytes of data:
Reply from 10.0.0.10: bytes=32 time=0ms TTL=128
Reply from 10.0.0.10: bytes=32 time=0ms TTL=128
Reply from 10.0.0.10: bytes=32 time=0ms TTL=128
Reply from 10.0.0.10: bytes=32 time=4ms TTL=128

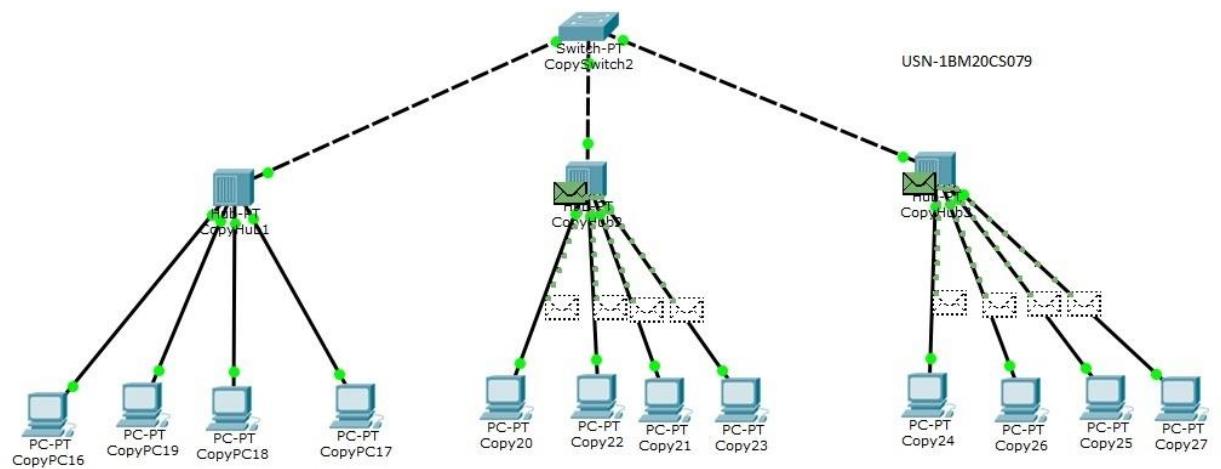
Ping statistics for 10.0.0.10:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 0ms, Maximum = 4ms, Average = 1ms

PC>
```

10 USN-1BM20CS079







17/11/22

Date / /

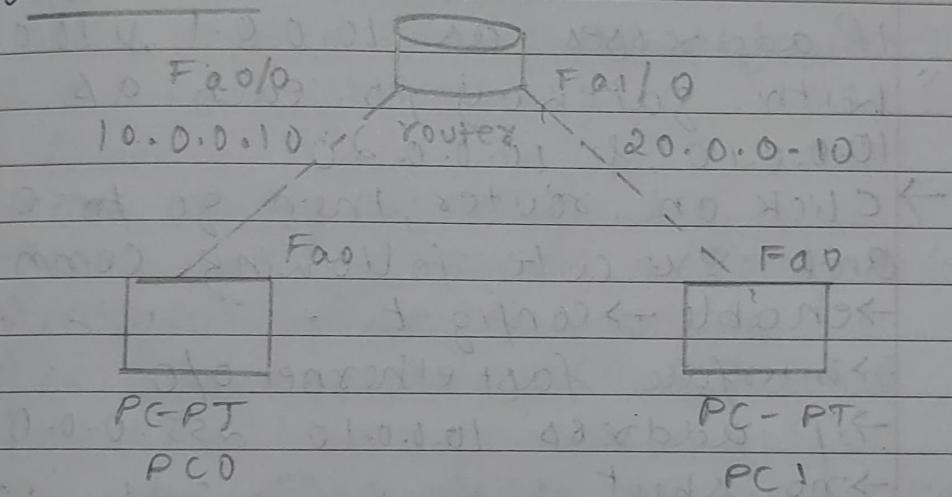
Page _____

SPLASH

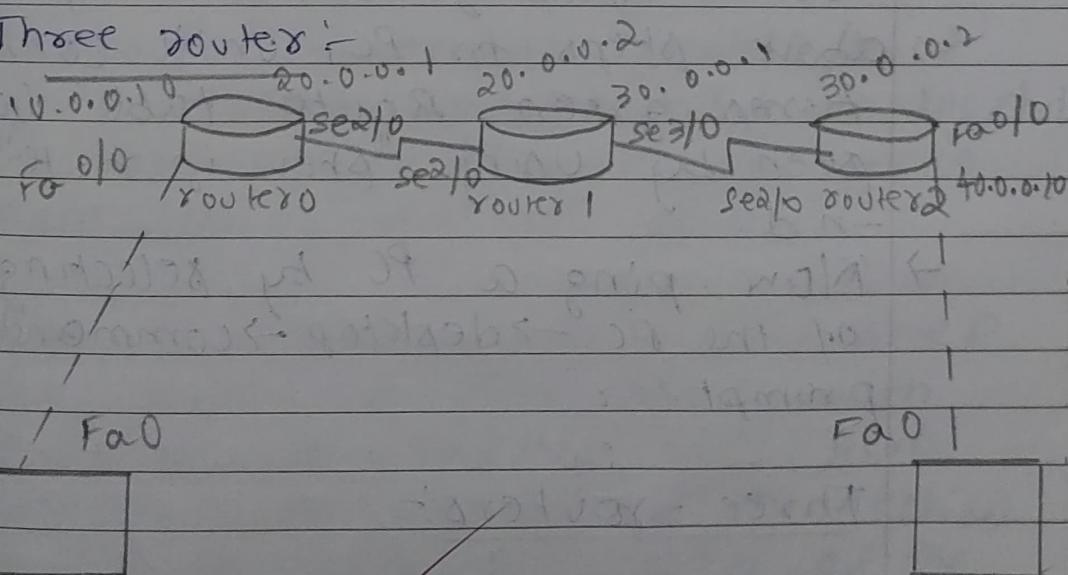
Aim: Configuring IP addresses to routers in packet tracer. Explore the following messages: Ping responses, destination unreachable, request timed out, reply.

Topology:

One-router:



Three routers:



Procedure:

One-router :-

- Place 2 generic PC, a generic router and notes with IP addresses and connect using copper cross wire.
- Click on both PC's and set IP addresses as 10.0.0.1 & 10.0.0.2 with gateway for each as 10.0.0.10 & 10.0.0.20
- Click on router, then go to CLI and execute following commands
 - enable → config t
 - interface fast ethernet 0/0
 - ip address 10.0.0.10 255.0.0.0
 - no shutdown

now connection between PC & router turns green and repeat above steps for PC, until connection turns green. Route table can be seen by using "show ip route" command.

- Now ping a PC by selecting one of the PC → desktop → command prompt.

Three routers:-

- Place 2 generic PC's, 3 routers such that 2 PC's are connected to 1st and 3rd router using copper

Computer and 3 routers are connected with sequence 1st ↔ 2nd ↔ 3rd using serial DCE cable.

- place notes to indicate IP addresses and ports of fastethernet & serial ports.
- set IP addresses, gateway, subnet mask for both the PCs.
- Now configure router 1:
 - click → no → enable → Config +
 - interface fastethernet 0/0
 - ip address 10.0.0.10 255.0.0.0
 - no shut

by this first PC and left half of router connection is established.

→ config + → interface serial 2/0 → ip address 20.0.0.1 255.0.0.0 → no shut
now right side connection is established for 1st router.
- Do the same above done step for router 2 and three with correct IP address and port selection to establish connection on both sides of each of the routers successfully.
- After all connections, green lights

are shown as a result of successful completion, now if we ping, we get "destination unreachable"

- If we ping to other routers, we get "request timed out" as routers are not trained for indirect connected LAN's
- Train router 1 by :-
ip route 30.0.0.0 255.0.0.0 20.0.0.2
ip route 40.0.0.0 255.0.0.0 20.0.0.2
- Train router 2 by :-
ip route 10.0.0.0 255.0.0.0 20.0.0.1
ip route 40.0.0.0 255.0.0.0 30.0.0.2
- Train router 3 by :-
ip route 10.0.0.0 255.0.0.0 30.0.0.1
ip route 20.0.0.0 255.0.0.0 30.0.0.1
- Now pinging from PC0 to PC1 is successful.

Observations:-

one-router :-

Pinging output for first time

ping 20.0.0.1

pinging 20.0.0.1 with 32 bytes of data:
request timed out

Reply from 20.0.0.1: bytes = 32

Reply from 20.0.0.1: bytes = 32

Reply from 20.0.0.1: bytes = 32

ping statistics for 20.0.0.1:

packets sent = 4, received = 3, lost = 1

But when PC0 pings PCI again
or if reverse ping happens, we
get reply all 4 times

3 routers

- * Output when PC0 ip pinged by PCI or vice versa before routers are trained of unknown IP's.

ping 40.0.0.1

pinging 40.0.0.1 with 32 bytes of data

reply from 10.0.0.10: destination host unreachable

pinging statistics 40.0.0.1

packets sent = 4 received = 0 lost = 4

ping 20.0.0.2

request timed out

request timed out

request timed out

request timed out

pinging statistics 20.0.0.2

packets sent = 4, received = 0, lost = 40

→ once routes are trained.

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 = 2 ms

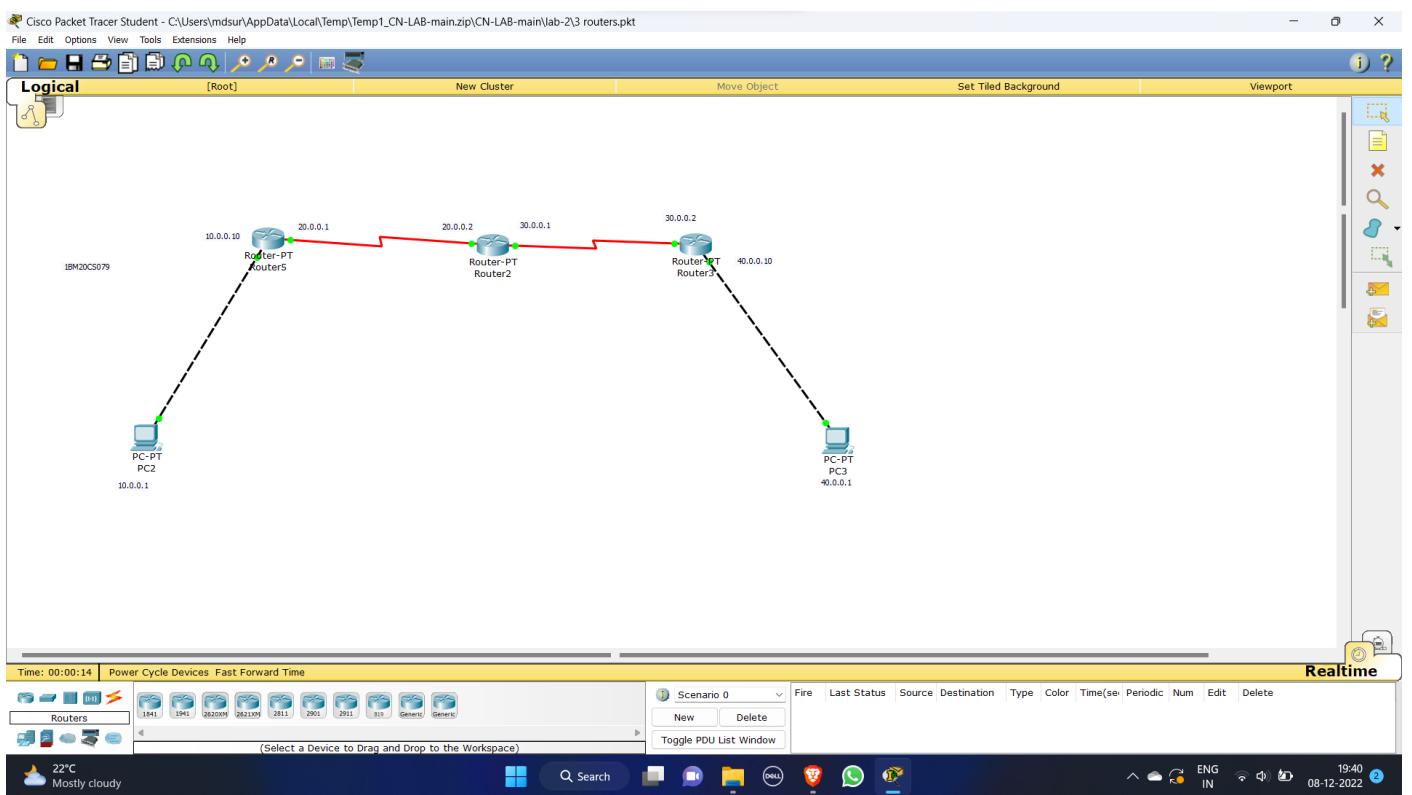
reply from 40.0.0.1: bytes = 32 time = 2 ms

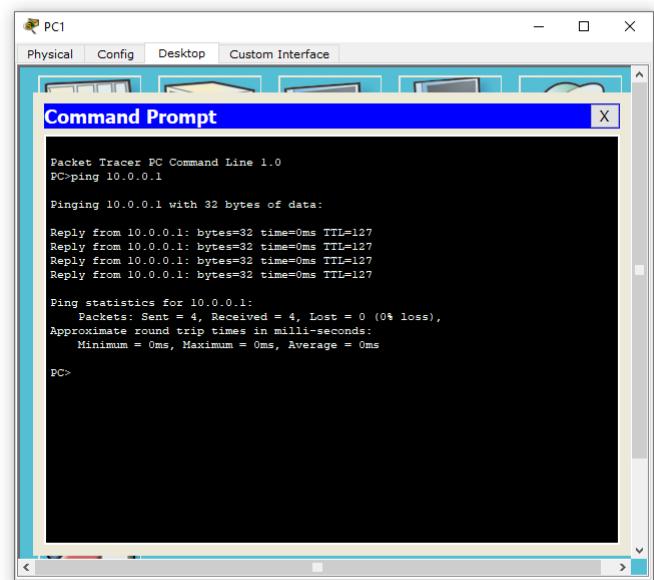
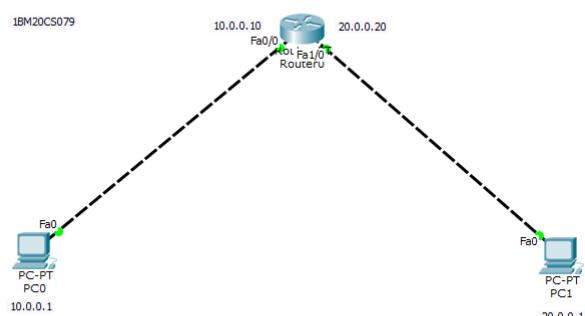
reply from 40.0.0.1: bytes = 32 time = 2 ms

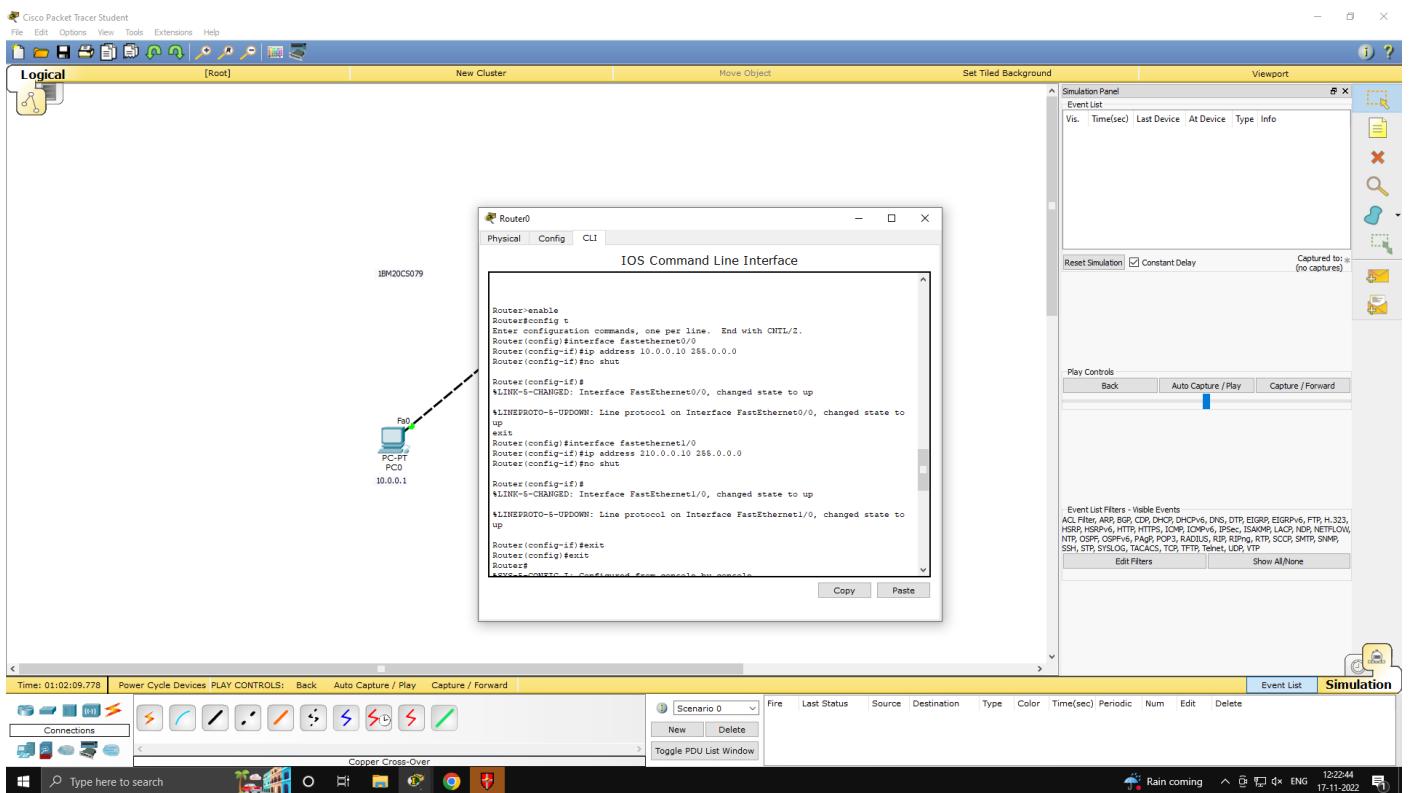
pinging statistics for 40.0.0.1

packets sent = 4, received = 3, lost = 1

✓
24/11/22







24/11/22

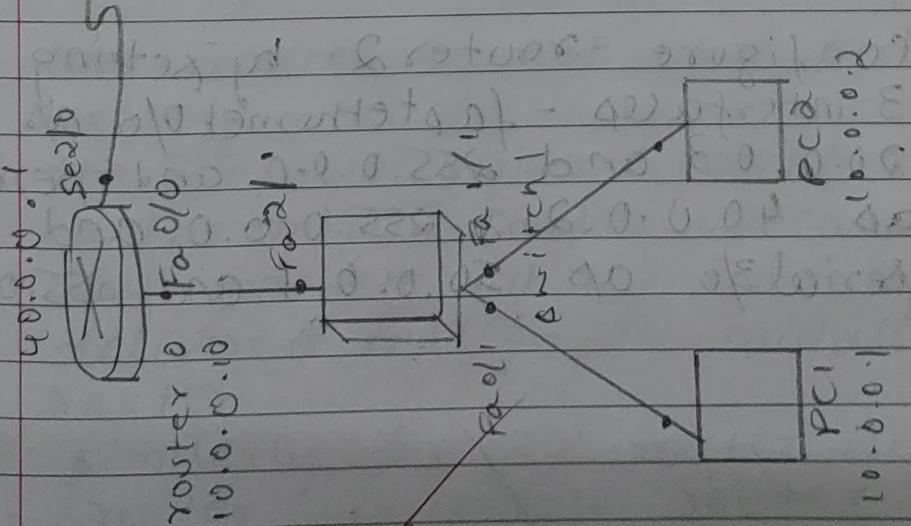
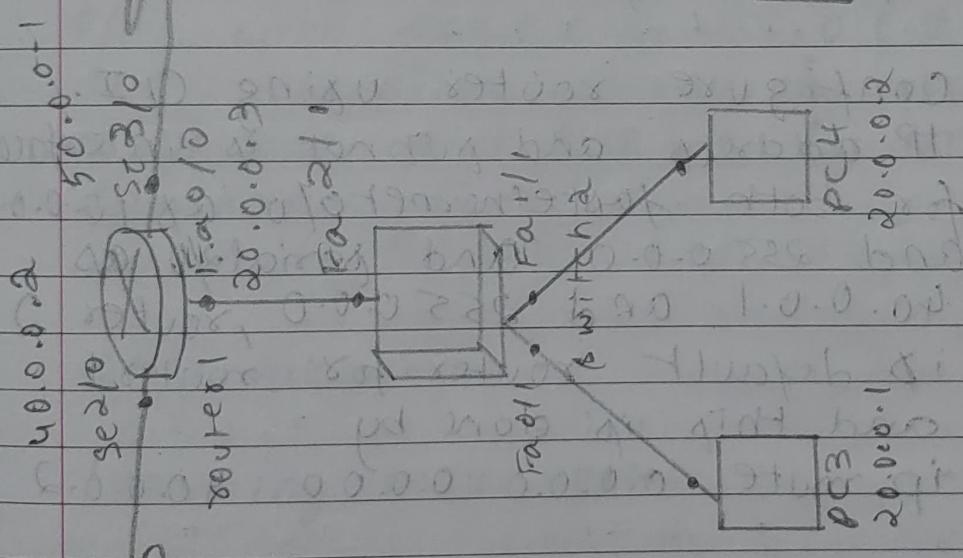
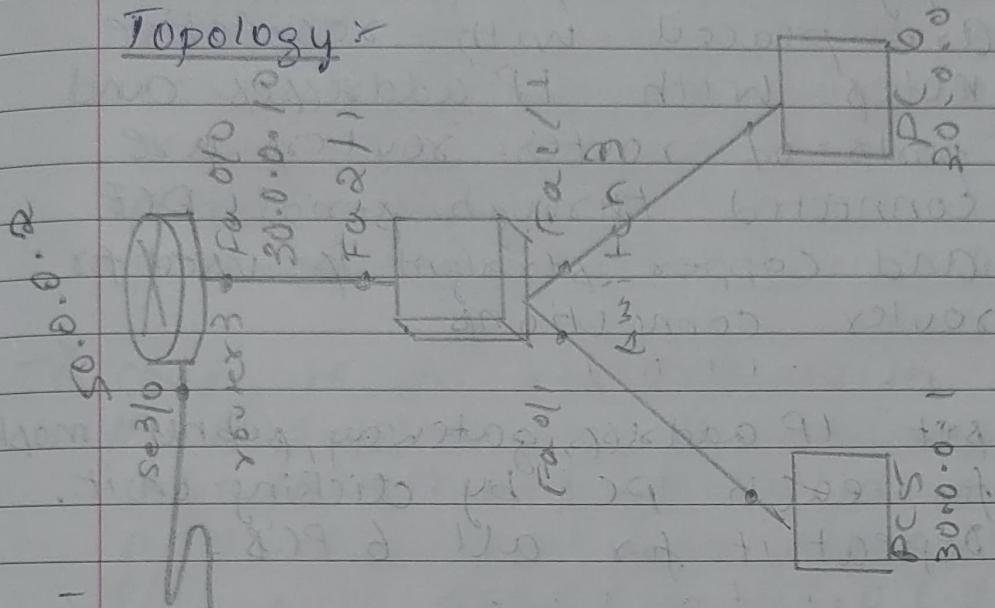
Date 11/11

Page

SPLASH

Aim: configuring default route to router

Topology:



Procedure:

- 6, generic PC's, 3 switches, 3 routers are placed with respective notes with IP addresses and ports of router. routers are connected through serial DCE and copper straight IP used for router connections.
- set IP address, gateway, subnet mask for each PC by clicking on it. Repeat it for all 6 PC's.
- configure router using CLI.
IP address and subnet ip interface for both fastethernet0/0 as 10.0.0.10 and 255.0.0.0 and serial2/0 as 40.0.0.1 and 255.0.0.0, router 2 ip default router for router1 and this ip done by ip route 0.0.0.0 0.0.0.0 40.0.0.2
- configure router2 by setting 3 interfaces - fastethernet0/0 as 20.0.0.3 and 255.0.0.0 and serial1 as 40.0.0.2 & 255.0.0.0 and serial3/0 as 50.0.0.1 and 255.0.0.1

Router 2 does not have any default routers and static routing is done for the network 10 and 40 by the following command,

`ip route 10.0.0.0 255.0.0.0 40.0.0.1`

`ip route 30.0.0.0 255.0.0.0 50.0.0.2`

→ Configure for Router 3 with IP addresses and subnet mask 0.0.0.0/0 with 50.0.0.10 and 255.0.0.0 and serial serial 2/0 with 50.0.0.2 & 255.0.0.0.

The default router for Router 3, Router 2 and this set by the command

`ip route 0.0.0.0 0.0.0.0 50.0.0.1`

Observation:

Learning outcomes:-

- * One router cannot have a default router.
- * The Default router for first router is the middle router because any packets which have to be delivered will go to middle router.
- * Similarly for 3rd router, default router is middle router.
- * No default router for middle

router because if one of the router ip made default then there is a chance that the packets which are to be sent to the switch are sent to the router.

Result :-

ping 120.0.0.1

pinging 120.0.0.1 with 32 bytes of data

request timed out

reply from 120.0.0.1: bytes=32, time=1ms,

reply from 120.0.0.1: bytes=32, time=2ms,

reply from 120.0.0.1: bytes=32, time=6ms,

ping 30.0.0.2

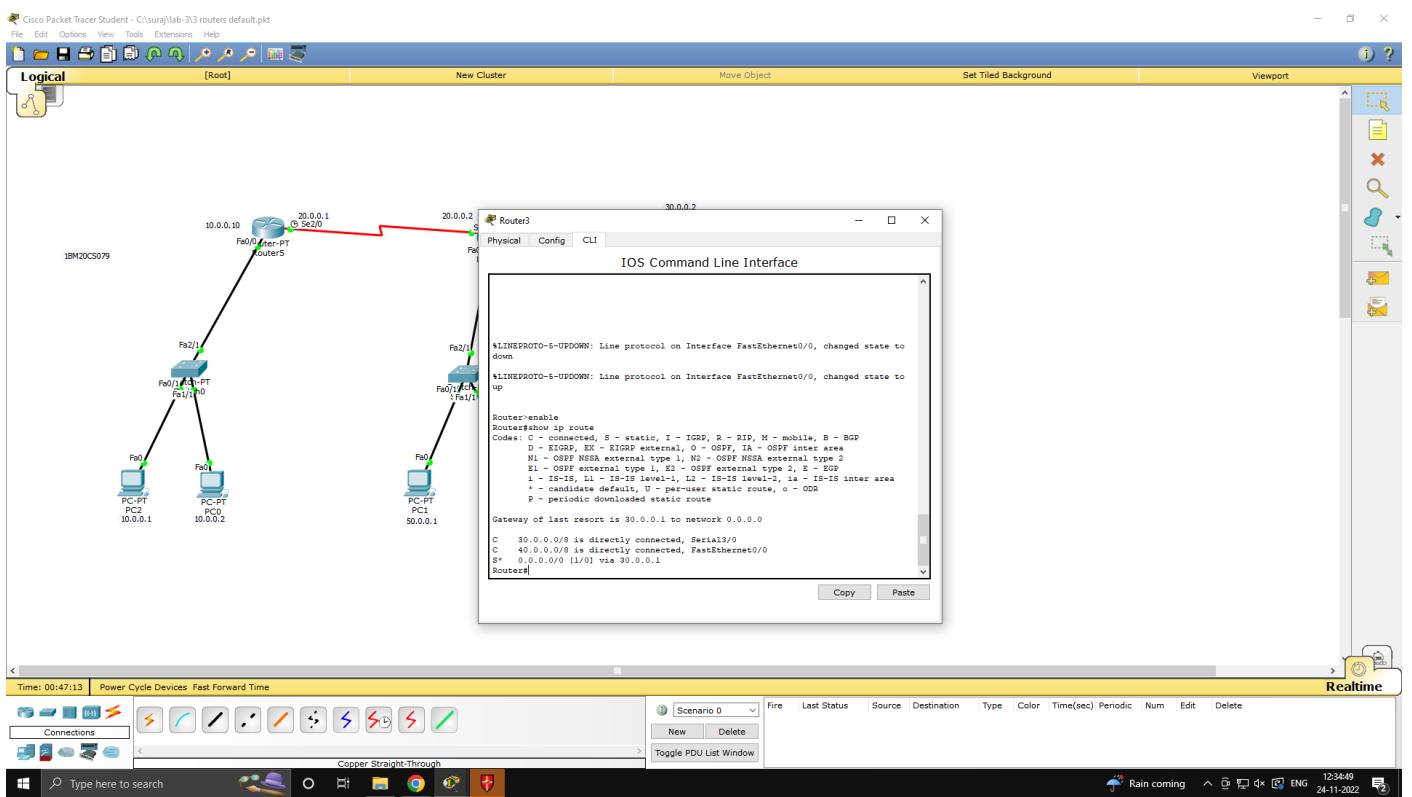
pinging 30.0.0.2 with 32 bytes of data
request timed out

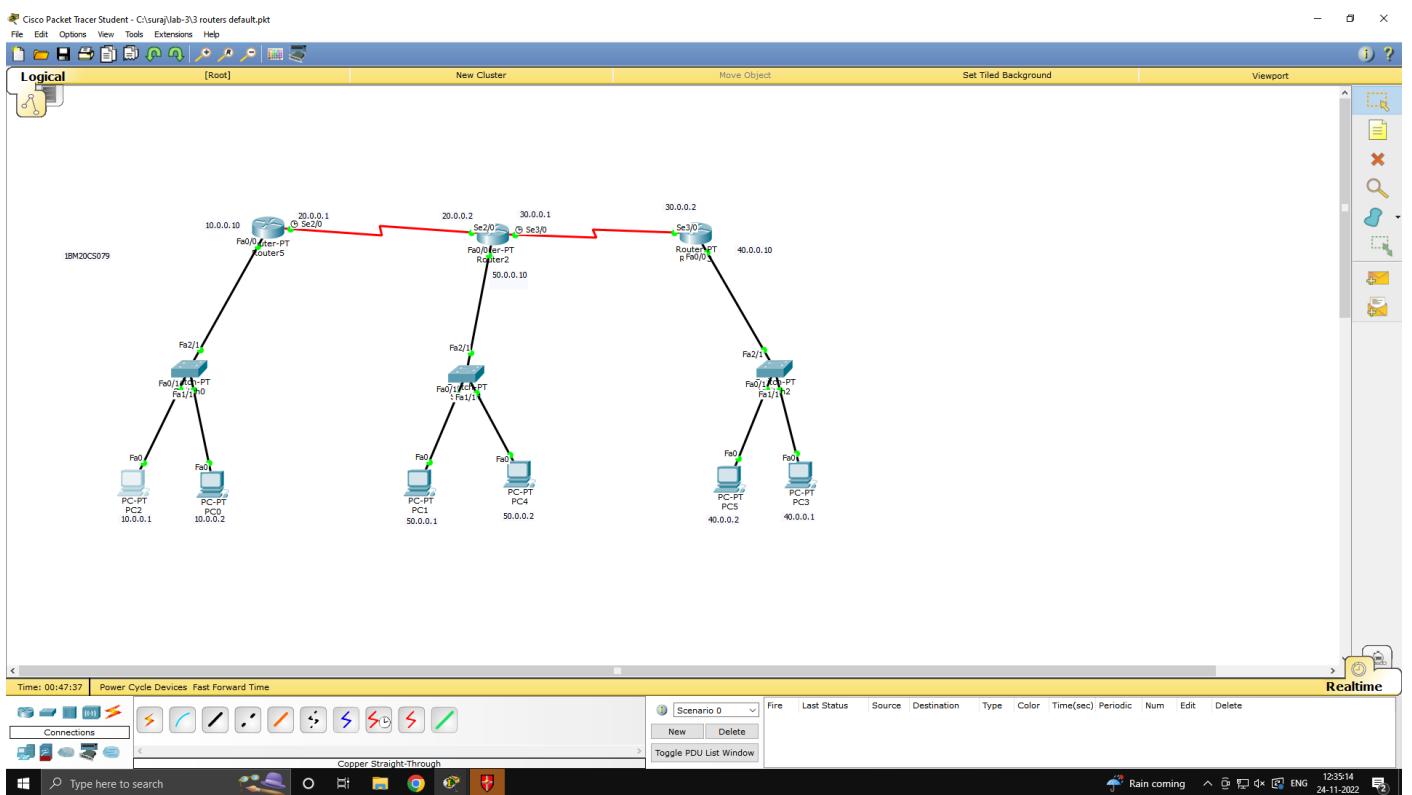
reply from 30.0.0.2: bytes=32, time=4ms

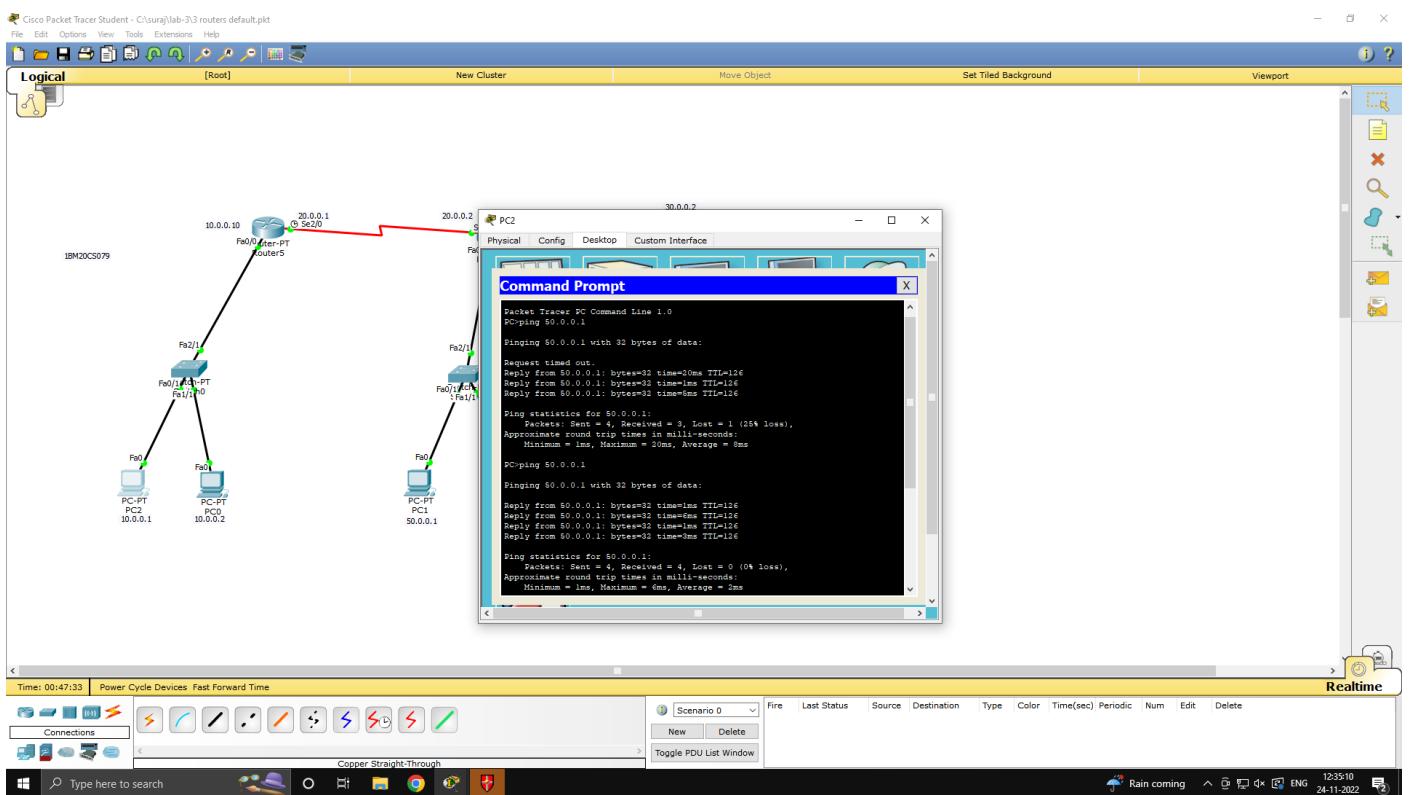
reply from 30.0.0.2: bytes=32, time=4ms

reply from 30.0.0.2: bytes=32, time=4ms

1/12/22





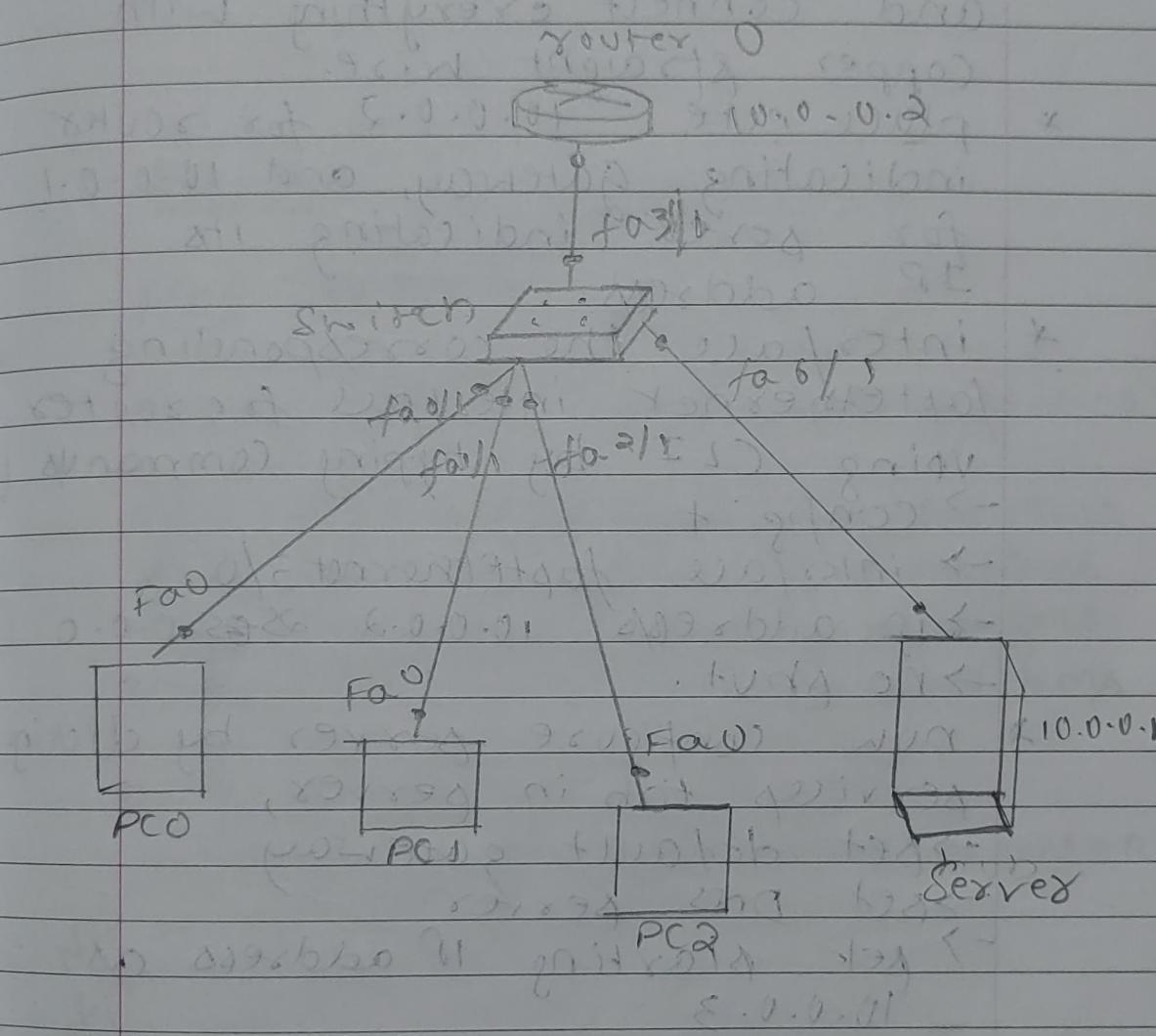


1/12/22

Date _____
Page _____
SPLASH

Aim:- Configuring DHCP within a LAN in a packet tracer

Topology:-



Importance of port

Port mapping in port

Procedure:

- * place 3 pc's , one router, one generic switch, one server and connect everything with copper straight wire.
- * place note 10.0.0.2 for router indicating gateway, and 10.0.0.1 for server indicating its IP address.
- * interface the corresponding fastethernet interface for router using CLI by typing commands
 - config +
 - interface fastethernet 3/0
 - ip address 10.0.0.2 255.0.0.0
 - no shut.
- * now configure server by clicking services tab in server,
 - set default gateway
 - set DNS server
 - set starting IP address as 10.0.0.3
- * Observation:

learning outcome:

- clicking on IP configuration and selecting DHCP mode automatically fetches

IP address, gateway from server for which we had configured already.

Result :-

Ping 10.0.0.4

Pinging 10.0.0.5 with 32 bytes of data

reply from 10.0.0.5: bytes = 32 time = 4ms

reply from 10.0.0.5: bytes = 32 time = 4ms

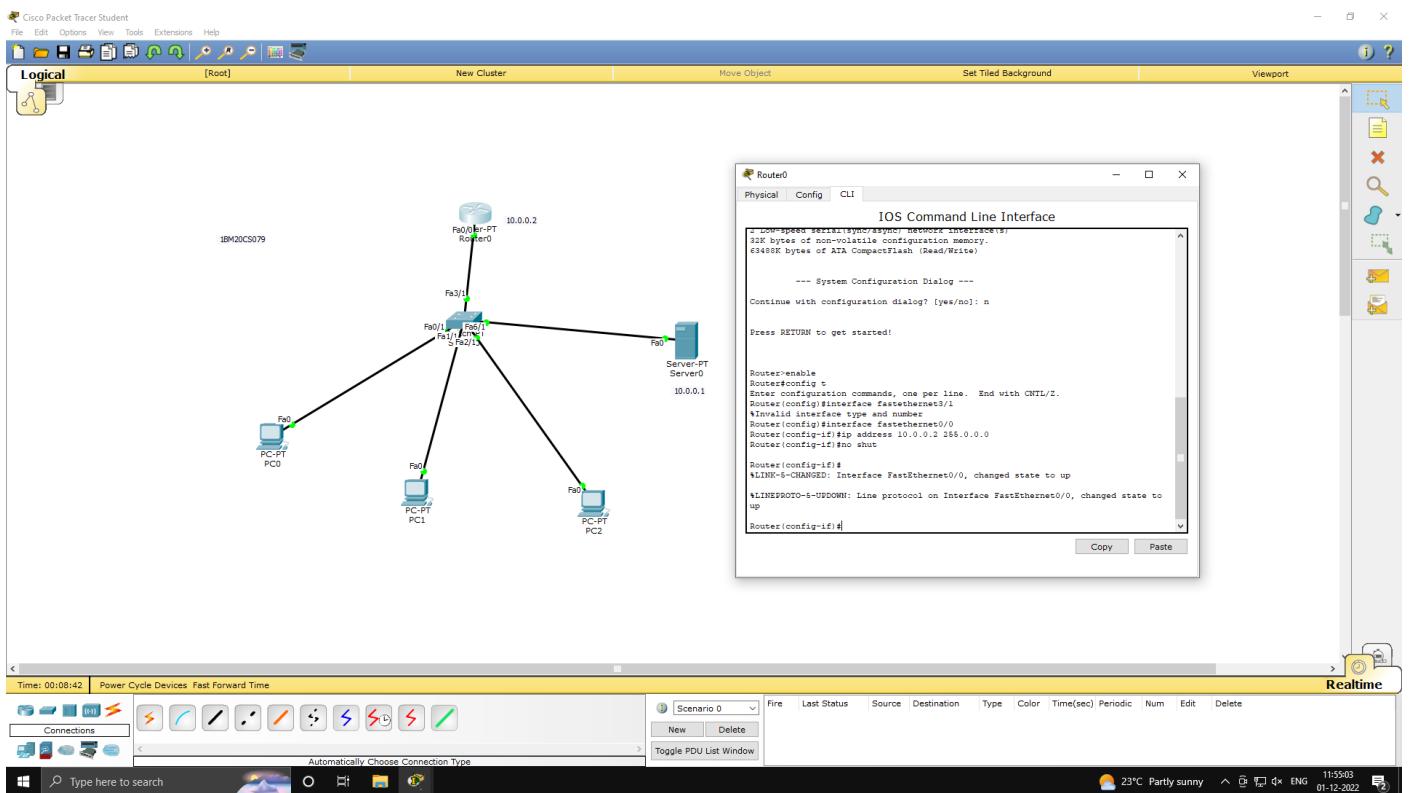
reply from 10.0.0.5: bytes = 32 time = 4ms

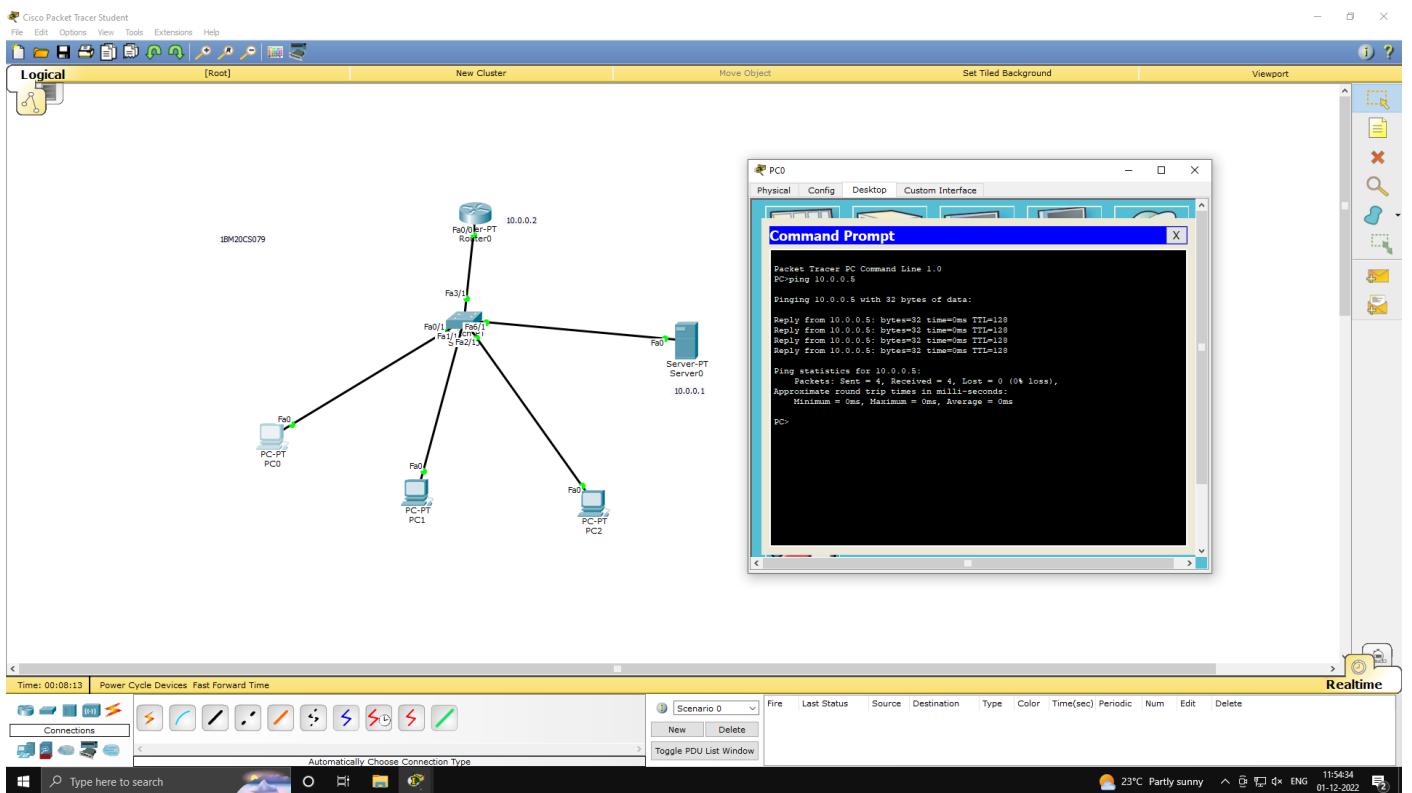
reply from 10.0.0.5: bytes = 32 time = 4ms

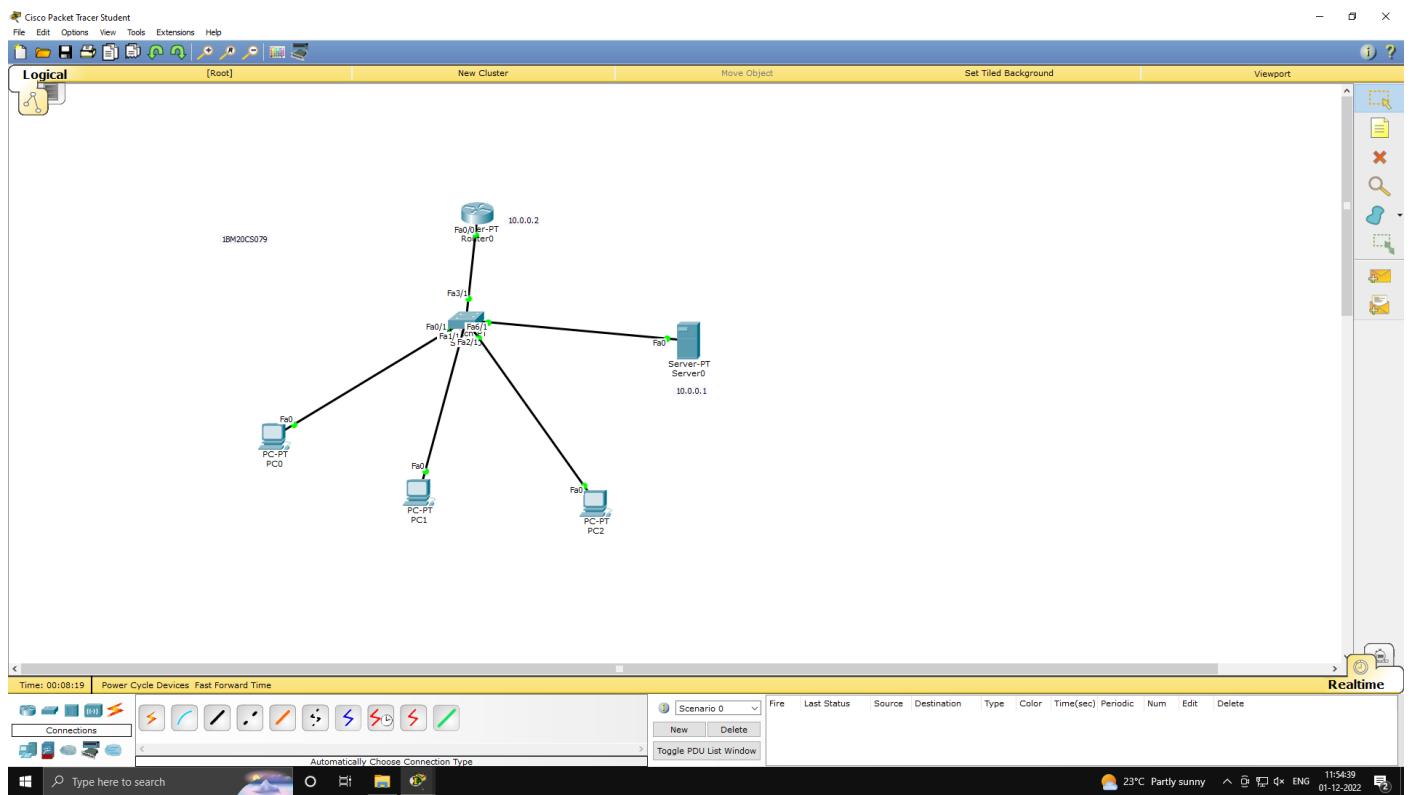
ping statistics for 10.0.0.5:

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

↖
8/11/2021



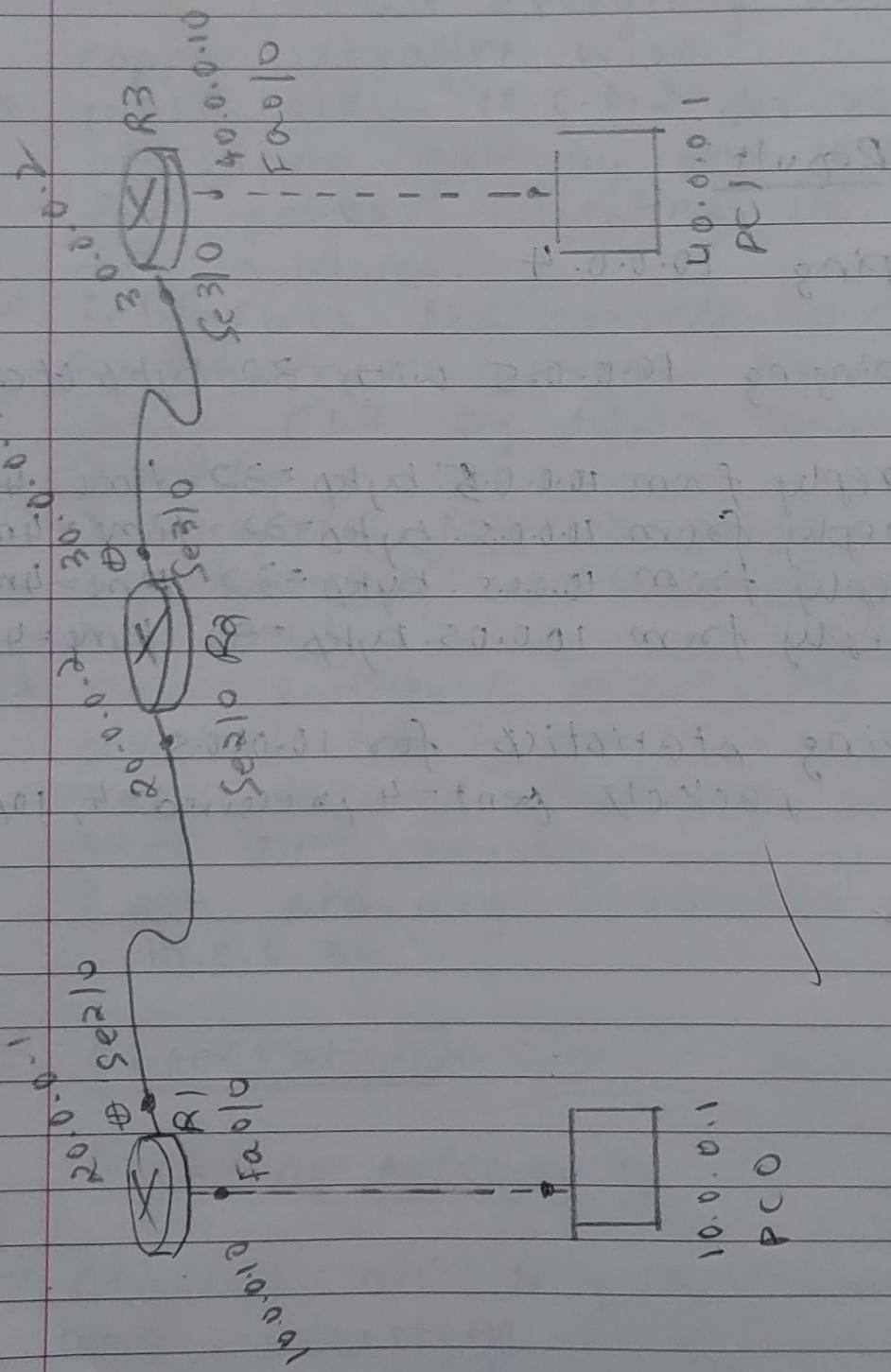




8/12/22

Aim:- configuring RIP routing protocol in routers

Topology :-



Procedures

- * RIP is routing information protocol which finds ^{shortest} best path between source and destination network.
→ It is a distance vector routing protocol.
- * Procedure:
 - place 3 generic routers, 2 generic PC and placed notes to indicate respective IP addresses.
 - use serial DCE cable to connect routers and open copper cross over cable to connect PC with router 1 and router 3
 - set IP address, gateway and subnet masks as 10.0.0.1, 10.0.0.10, 255.0.0.0 for PC0 and respective by set 40.0.0.1, 40.0.0.10, 255.0.0.0 for PC1
 - interface PC0 and router 1 using following commands
 - interface fastethernet 0/0
 - ip address 20.0.0.10 255.0.0.0
 - no shutdown
 - for interfacing serial 2/0 of router 1 open following commands
 - interface serial 2/0
 - ip address 20.0.0.10 255.0.0.0
 - encapsulation PPP

DCE

- clock rate 64000
- no phvt

* use above commands for interfacing router which has clock symbol in cable near to it and for other interfaces of routers use same above command except "clock rate 64000" command

* Once all the green lights are visible, follow the commands below for each router,

- router>rip
- network 10.0.0.0
- network 20.0.0.0
- exit

* repeat above commands for router 2 and router 3 with respective network addresses.

→ Observations:

* instead of using static IP routing for all routers, by using RIP, routing becomes easy when large number of routers are present.

→ Result :-

Pinging 10.0.0.1 with 32 bytes of data:

Reply from 10.0.0.1: bytes=32

Reply from 10.0.0.1: bytes=32

Reply from 10.0.0.1: bytes=32

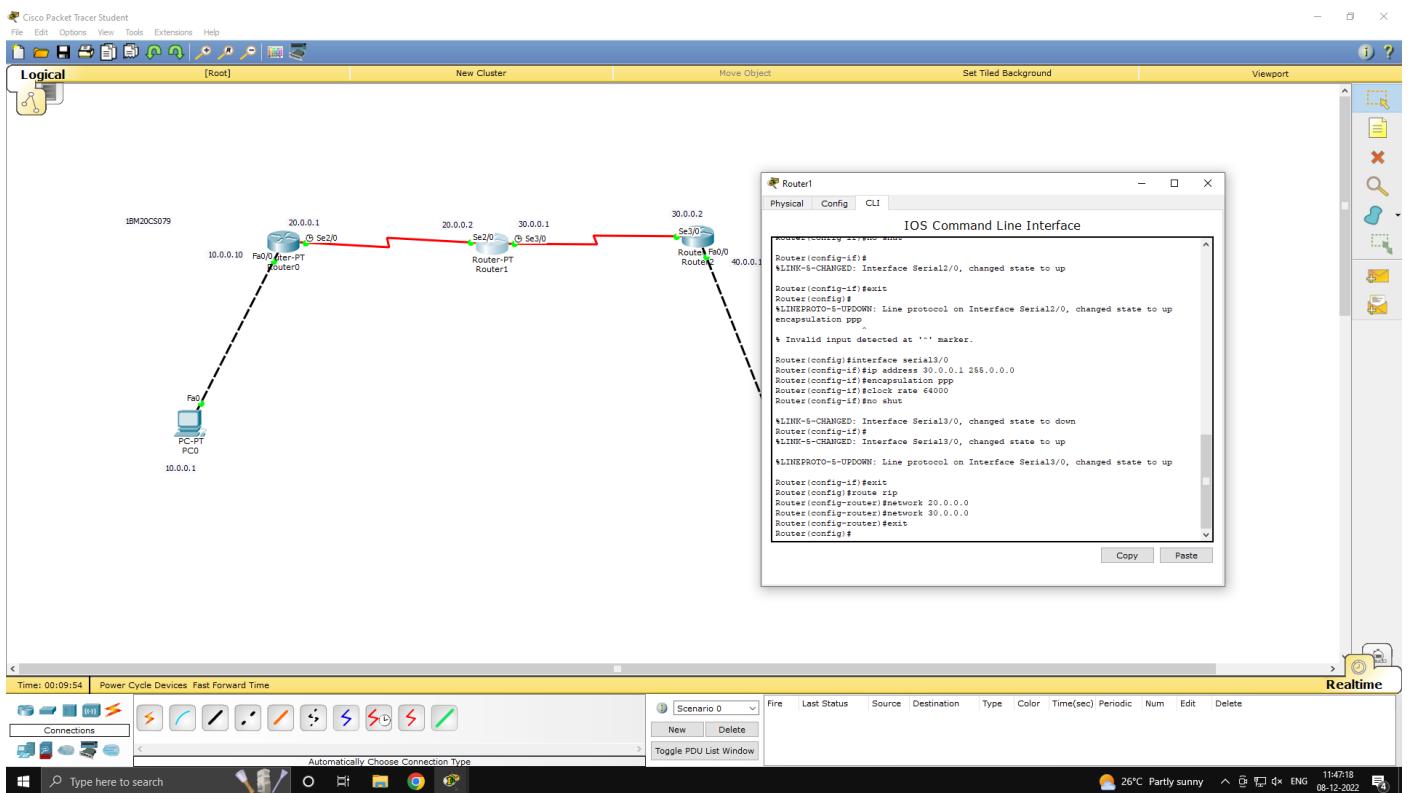
Reply from 10.0.0.1: bytes=32

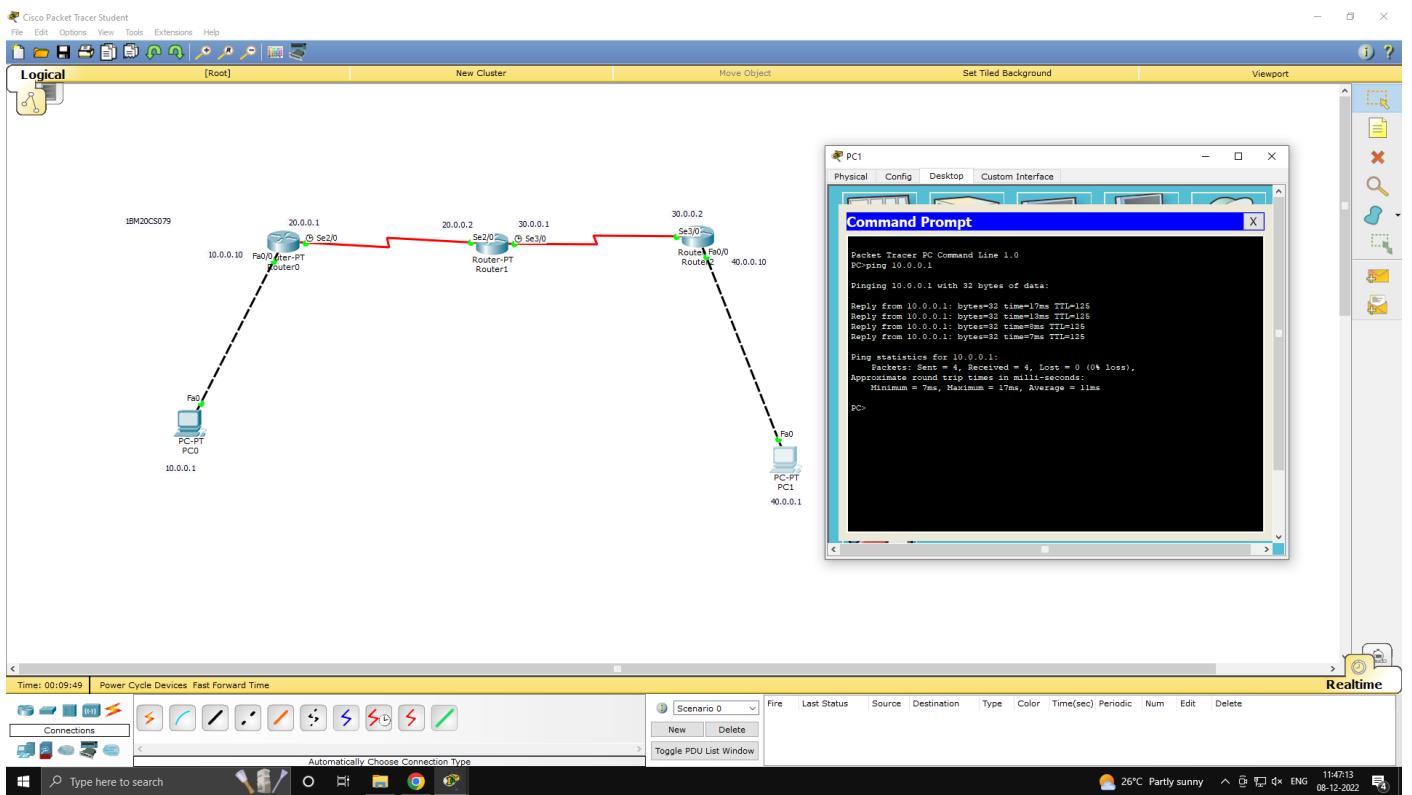
Ping statistics for 10.0.0.1:

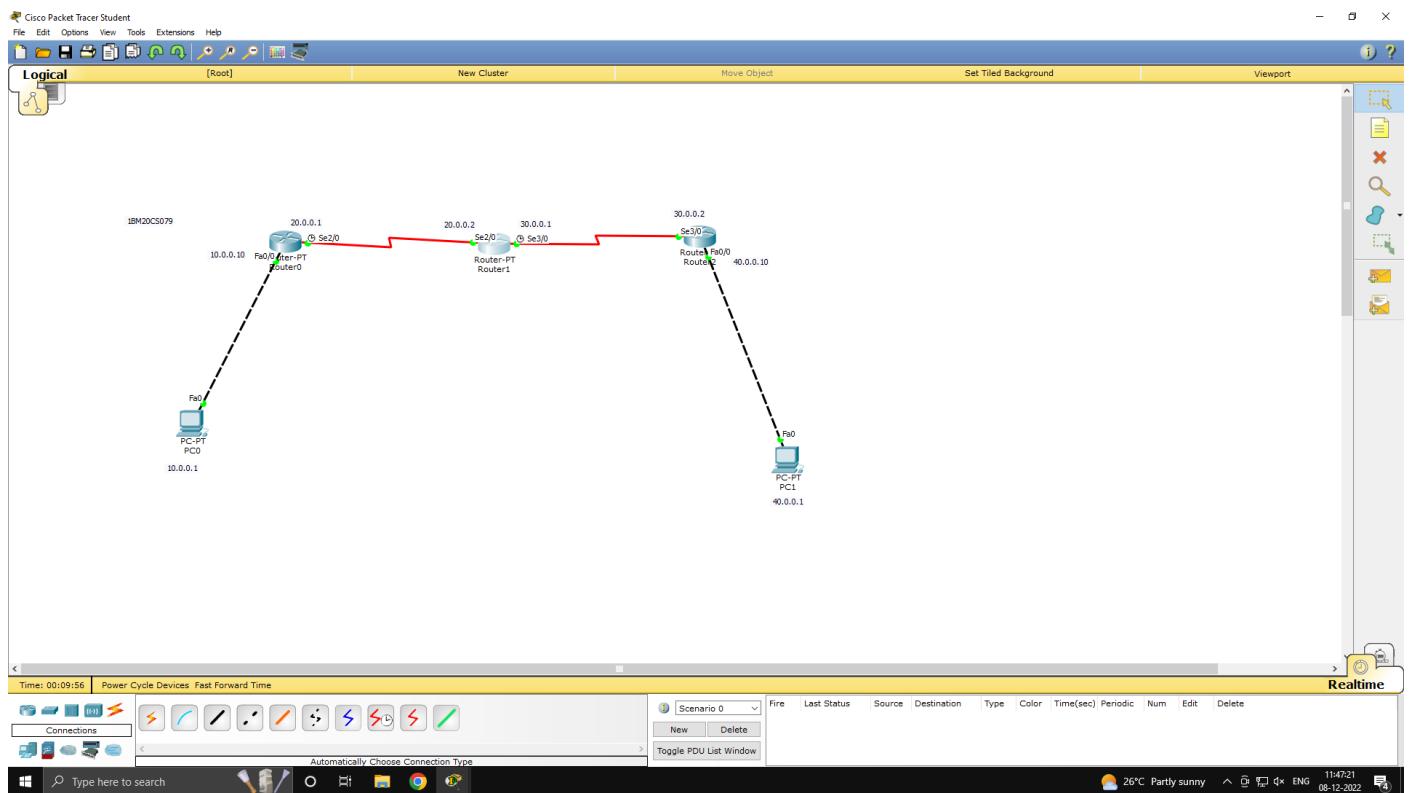
packets: sent=4, received=4, lost=0

N

8 | 12 | 22

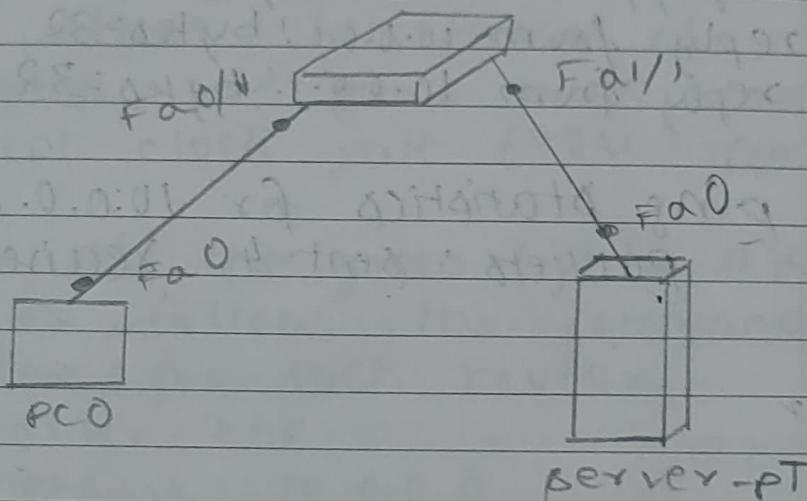






Aim:- Demonstration of WEB server and DNS using packet tracer

Topology



Procedure:

- Set the IP addresses of the PC and server respectively as 10.0.0.2 & 10.0.0.1.
- Open web-browser in desktop tab of PC and type IP of server "http://10.0.0.2"
- Default home page will be displayed.
- Now go to services tab of server enable HTTP and change contents of index.htm by clicking edit option.

→ check again in the browser of PCO to see the updated changes.

→ Activate DNS →

→ enable DNS on DNS service to activate it.

→ enter the domain name and IP addresses needed to be mapped.

↓ "on" → [10.0.0.2]

→ click add to add the new mapping.

→ Now give name in the web-browser to check if its displaying index.html

→ Custom page

→ Create a new page resume.html and save it in http service.

→ change hyperlink in index.html to link the above created file.

→ check the output in the webbrowser of PCO by clicking on hyper-link.

Operations

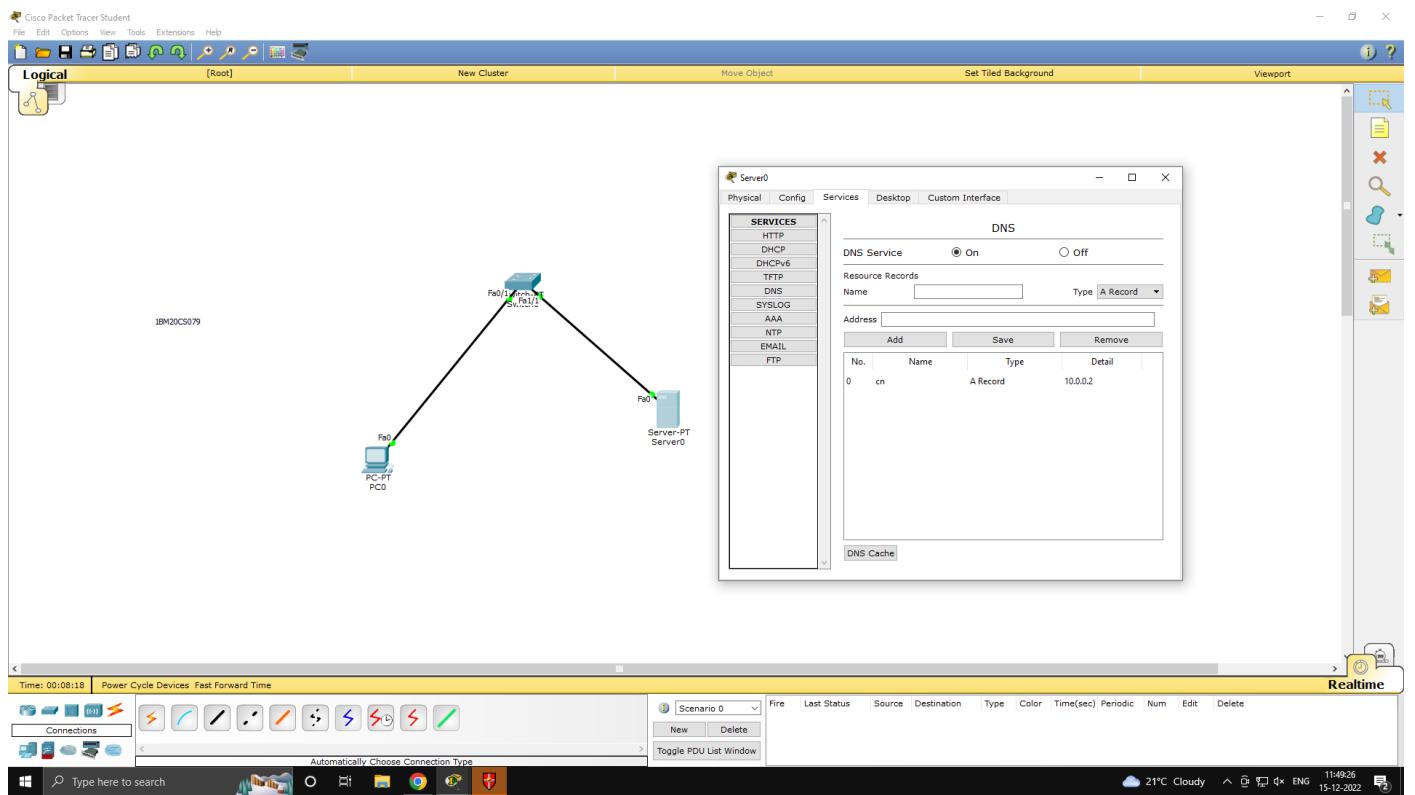
- We can view the webpage when we type "cn" in browser because 192.0.0.2 address is mapped to the name "cn", by domain name system (concept).
- ↳ Mapping is required because it's difficult for users to remember IP's, hence IP is mapped with a name.

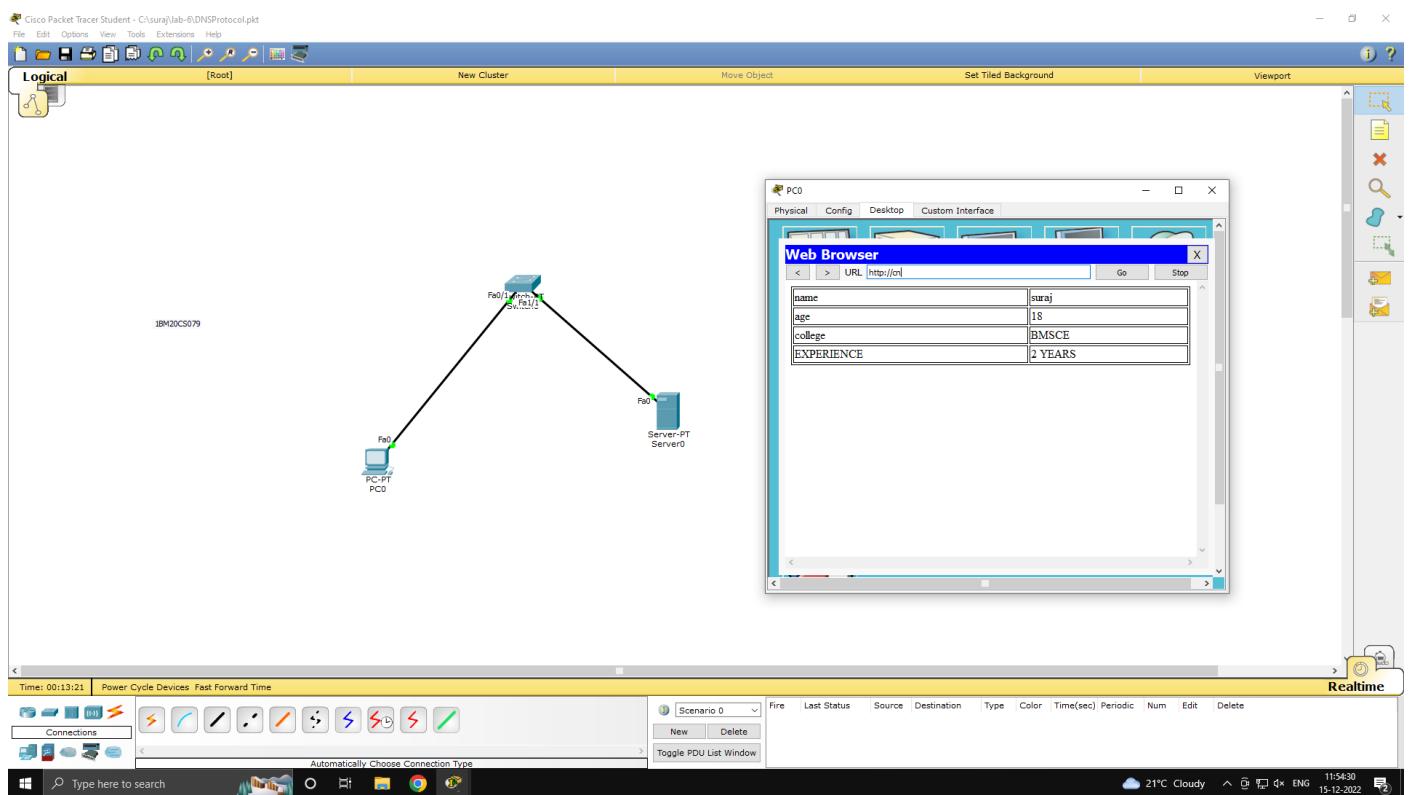
Result

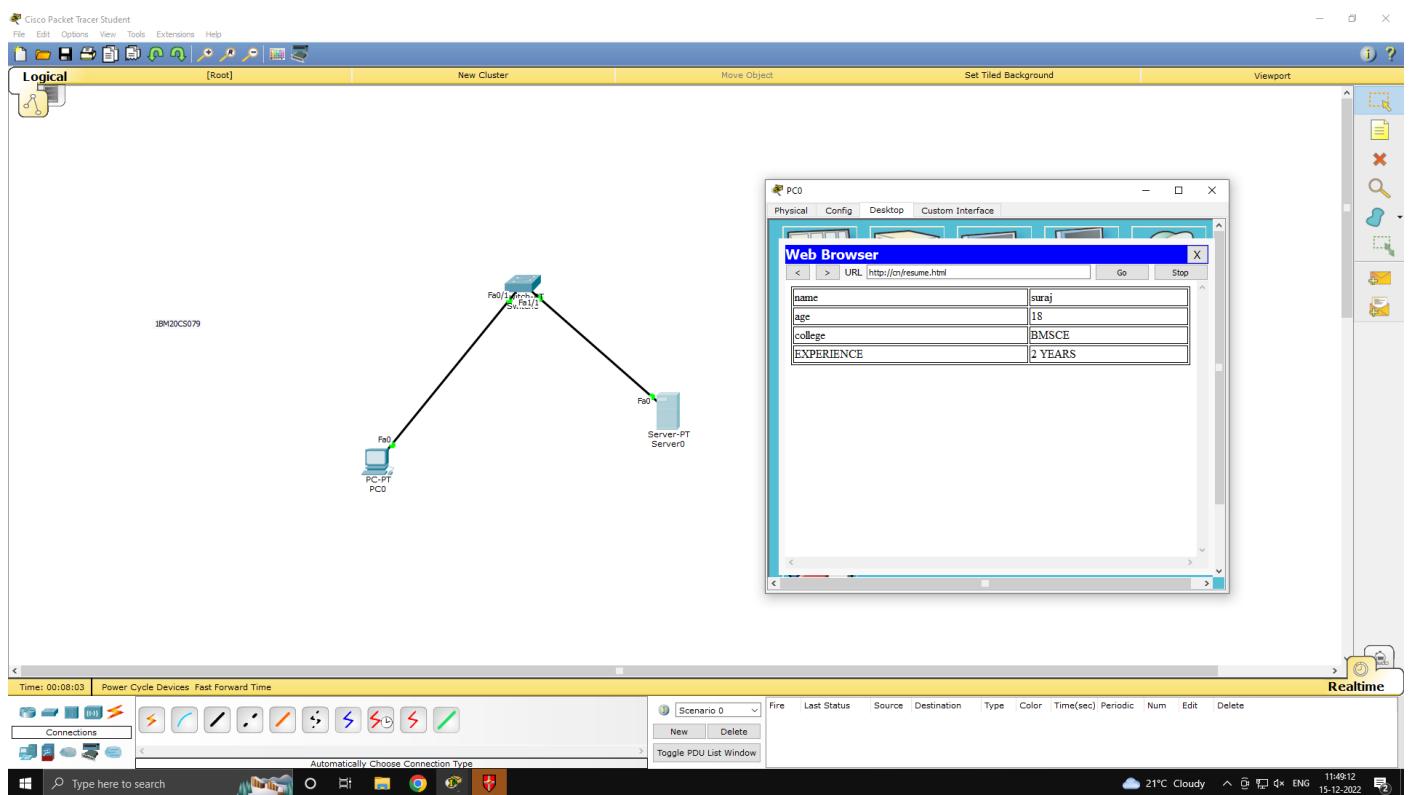
Web Browser 
↔ URL <http://cn/resume.html>

name	suraj
age	18
college	BMSCE
Experience	2 Years

✓ All address and
port mapped
29-12-2022







Aim: Write a program for
error detecting code using
ERC-CCTT

code:

```

def dir (code, poly):
    i = len (poly)
    temp = code[0:len(poly)]
    for j in range(0, len(code)-len(poly)):
        if (temp[j:i] != '0'):
            rep = xor (temp, poly, len(poly))
            temp = rep[1:] + code[i]
        else:
            temp = temp[1:] + code[i]
        i = i + 1
    print
    if (temp[0:i] != '0'):
        rep = xor (temp, poly, len(poly))
        send = rep[1:i]
    else:
        send = temp[1:]
    return send

```

```

def xor (a, b, n):
    ans = ""
    for i in range(n):
        if (a[i] == b[i]):
            ans += "0"
        else:
            ans += "1"
    return ans

```

```

code = input("enter codeword\n")
poly = "10001000000100001"
zeroes = "0000000000000000"
mod_code = code + zeroes
sender = division(mod_code, poly)
receiver = code + sender
ans = division(receiver, poly)
if ans == "0000000000000000":
    print("no error in data")
else:
    print("error in data")

```

OUTPUT~~Enter codeword~~~~100100~~~~no error~~~~Enter codeword~~~~1001000~~~~error when 100100000 is sent~~~~check for error~~

```
PS C:\Users\mdsur\Documents\COLLEGE\1BM20CS079-CN-LAB\7-CRC> python -u "c:\Users\mdsur\Documents\COLLEGE\1BM20CS079-CN-LAB\7-CRC\crc.py"
ENTER CODEWORD
10001000000100001
sender data is 110
receiver data is 10001000000100001110
no error
PS C:\Users\mdsur\Documents\COLLEGE\1BM20CS079-CN-LAB\7-CRC>
```

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

```
#include <bits/stdc++.h>
using namespace std;
#define MAX 10
int n;
class Router {
    char adj-new[MAX], adj-old[MAX];
    int table-new[MAX], table-old[MAX];
public:
    Router() {
        for (int i = 0; i < MAX; i++) table-old[i] = table-new[i] = 99;
    }
    int eval() {
        for (int i = 0; i < n; i++) {
            if (table-old[i] != table-new[i] || adj-new[i] != adj-old[i]) return 0;
        }
        return 1;
    }
    void input(int j) {
        cout << "Enter 1 for corresponding\nRouter id adjacent to Router ";
        cout << (char)('A' + j) << " else 99." << endl;
        cout << "Enter matrix";
```

```
for (int i=0; i<n; i++) {  
    if (i==j)  
        table_new[i]=0;  
    else  
        cin>>table_new[i];  
    adj_new[i] = (char)('A'+1);  
}  
cout<<endl;  
}  
  
void display(){  
    cout<<"In debt matrix:";  
    for (int i=0; i<n; i++)  
        cout<<"\n out line";  
    for (int i=0; i<n; i++) cout<<adj[i];  
    cout<<"\n np count";  
    for (int i=0; i<n; i++) cout<<table[i];  
}  
  
void build_tab6(){  
    int i=j=0;  
    while (i!=n){  
        for (i=j; i<n; i++) {  
            r[i].copy();  
            r[i].build(i);  
        }  
        for (i=0; i<n; i++)  
            if (!r[i].equal()) {  
                j=i;  
                break;  
            }  
    }  
}
```

```
int main() {  
    cout << "enter no of routers (<" << MAX << ")";  
    cin >> n;  
    for (int i=0; i<n; i++) r[i].input(i);  
    build();  
    for (i=0; i<n; i++) {  
        cout << " routers table entries for  
        router " << (char)(A+i) << ":" - /",  
        r[i].display();  
        cout << endl << endl;  
    }  
}
```

OUTPUT:-

enter number of routers (<10): 5
enter 1 if router is adjacent to A else 99:
B C D E

enter matrix: 1 1 99 99

enter 1 if router is adjacent to B else 99:
A C D E

A B C E

enter matrix: 99 99 1 99

enter 1 if router is adjacent to E else 99:
A B C D

enter matrix: 99 99 1 99

router table entries for A:-

Dept router : ABCDE

outgoing line : ABCDE

Hop count : 0 1 1 99 99

router table entries for B:-

dept router : ABCDEF

outgoing line : ABCDEF

Hop count : 1 0 99 99 99

router table entries for router C:-

dept router : ABCDE

out line : ABCDEF

Hop count : 1 99 0 11

router table entries for router D:-

dept router : ABCDEF

out line : ABCDEF

Hop count : 99 99 1 0 99

router table entries for router E:-

dept router : ABCDEF

out line : ABCDEF

Hop count : 99 99 199 0

```
PS C:\Users\mdsur\Documents\COLLEGE-LAB> cd "c:\Users\mdsur\Desktop\DISTANCE_VECTOR\" ; if ($?) { g++ dv.cpp -o dv } ; if (?) { .\dv }

Enter the number the routers(<10): 5
Enter 1 if the corresponding router is adjacent to routerA else enter 99:
B C D E
Enter matrix:1 1 99 99

Enter 1 if the corresponding router is adjacent to routerB else enter 99:
A C D E
A B C E
Enter matrix:99 99 1 99

Enter 1 if the corresponding router is adjacent to routerE else enter 99:
A B C D
Enter matrix:99 99 1 99

Router Table entries for router A:-
Destination Router: A B C D E
Outgoing Line: A B C D E
Hop Count: 0 1 1 99 99

Router Table entries for router B:-
Destination Router: A B C D E
Outgoing Line: A B C D E
Hop Count: 1 0 99 99 99

Router Table entries for router C:-
Destination Router: A B C D E
Outgoing Line: A B C D E
Hop Count: 1 99 0 1 1

Router Table entries for router D:-
Destination Router: A B C D E
Outgoing Line: A B C D E
Hop Count: 99 99 1 0 99

Router Table entries for router E:-
Destination Router: A B C D E
Outgoing Line: A B C D E
Hop Count: 99 99 1 99 0
```

Aim: Implement Dijkstra's algorithm to compute shortest path for given topology

Code:

```
#include <bits/stdc++.h>
#include <limits.h>
#include <iostream.h>
using namespace std;
#define V 4
int minDist(int d[3], bool s[])
{
    int min = INTMAX, m_i;
    for (int v=0; v<V; v++)
        if (s[v] == false && d[v] < min)
            min = d[v], m_i = v;
    return m_i;
}
```

```
void printSolution(int dist[])
{
}
```

```
printf("vertex dist from source");
for (int i=0; i<V; i++)
    printf("%d %d \n", i, dist[i]);
}
```

```
void dijkstra(int g[V][V], int p)
{
}
```

```
int d[V]; bool s[V];
for (int i=0; i<V; i++)
    d[i] = INTMAX, s[i] = false;
```

$d[\text{src}] = 0$

```
for (int c = 0; c < r - 1; c++) {
    int u = minD(d[r], s);
    s[u] = true;
    for (int v = 0; v < r; v++)
        if (!d[v] && g[u][v] && d[u] + INT_MAX <= d[v] + g[u][v] < d[v])
            d[v] = d[u] + g[u][v];
}
```

} print solution(dip+);

```
int main()
```

```
{ int g[r][r];
cout << "enter graph" << endl;
for (int i = 0; i < r; i++)
    for (int j = 0; j < r; j++)
        cin >> g[i][j];
}
```

dijkstra(graph, 0);

return 0;

}

OUTPUT:

enter graph

0 9 2 5

9 0 6 8

2 6 0 0

5 8 0 0

vertex distance (from source)

0 0

1 8

2 2

3 5

```
PS C:\Users\mdsur\Documents\COLLEGE-LAB> cd "c:\Users\mdsur\Documents\COLLEGE-LAB\" ; if ($?) { g++ djikstras.cpp -o djikstras } ; if ($) { .\djikstras }
Enter the graph
0 9 2 5
9 0 6 8
2 6 0 0
5 8 0 0
Vertex      Distance from Source
0            0
1            8
2            2
3            5
PS C:\Users\mdsur\Documents\COLLEGE-LAB> []
```

Aim :- WAP for congestion control
using leaky bucket algorithm

code:-

```
b-c = input("enter bucket capacity")
o-r = input("enter output rate = ")
b-cont = 0
while (1):
    p-s = input("enter frame size")
    if (p-s > b-c):
        print("exceeded")
        continue
    if (int(p-s)+b-cont>int(b-c)):
        print("full")
        b-cont += b-cont - int(o-r)
    else:
        b-cont += int(p-s)
        b-cont -= int(o-r)
    print("b-cont = "+str(b-cont))
```

OUTPUT :-

enter bucket capacity = 500
 enter output rate = 5
~~enter frame size = 250~~
~~b-cont = 245~~
 enter frame size = 260
~~exceeded~~
~~b-cont = 240~~

WAP
1/23

```
enter bucket capacity = 500
enter output rate = 5
enter packet size = 250
current bucket content = 245
enter packet size = 260
bucket is already full
240
enter packet size = 220
current bucket content = 455
enter packet size = 45
current bucket content = 495
enter packet size = |
```

```
C:\Program Files\WindowsAp <--> + | ^  
enter bucket capacity = 500  
enter output rate = 5  
enter packet size = 600  
packet size exceeded  
current bucket content = 0  
enter packet size = |
```

Aim:- Using TCP / IP sockets, write client-server program to make client sending the filename and server to send back contents of requested file if present

ClientTCP.py :-

```
from socket import *
serverName = '127.0.0.1'
serverPort = 12000
clientSock = socket(AF_INET, SOCK_STREAM)
clientSock.connect((serverName, serverPort))
msg = input("Enter filename: ")
clientSock.send(msg.encode())
fileCont = clientSock.recv(1024).decode()
print("From Server")
print(fileCont)
clientSock.close()
```

ServerTCP.py :-

```
from socket import *
sName = '127.0.0.1'
sPort = 12000
sock = socket(AF_INET, SOCK_STREAM)
sock.bind((sName, sPort))
sock.listen(1)
while 1:
    print("ready to receive")
    cSock, addrs = sock.accept()
```

```
msg = cSock.recv(1024).decode()
file = open(msg, "r")
l = file.read(1024)
cSock.send(l.encode())
print('In sent contents of ' + msg)
file.close()
cSock.close()
```

client.py - C:/Users/mdsur/Desktop/client.py (3.10.9)

```
File Edit Format Run Options Window Help
from socket import *
serverName = '127.0.0.1'
serverPort = 12000
2
clientSocket = socket(AF_INET, SOCK_STREAM)
clientSocket.connect((serverName,serverPort))
sentence = input("\nEnter file name: ")
clientSocket.send(sentence.encode())
filecontents = clientSocket.recv(1024).decode()
print ("\nFrom Server:\n")
print(filecontents)
clientSocket.close()
```

IDLE Shell 3.10.9

```
File Edit Shell Debug Options Window Help
Python 3.10.9 (tags/v3.10.9:1dd9be6, Dec 6 2022, 20:01:21) [MSC v.1934 64 bit (AMD64)]
on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> ===== RESTART: C:/Users/mdsur/Desktop/client.py =====
Enter file name: server.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 ('Sent contents of ' + sentence)
    file.close()
    connectionSocket.close()
>>>
```

Ln: 14 Col: 0 Ln: 6 Col: 0

28°C Sunny ENG IN 14:36 29-01-2023

The screenshot shows a Windows desktop environment with two windows open. On the left is a code editor window titled "server.py - C:/Users/mdsur/Desktop/server.py (3.10.9)". It contains Python code for a socket server. On the right is an "IDLE Shell 3.10.9" window, which is a terminal window for Python. The terminal shows the Python interpreter running and executing the server code. The taskbar at the bottom displays various icons and system status.

```
server.py - C:/Users/mdsur/Desktop/server.py (3.10.9)
File Edit Format Run Options Window Help
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()

*IDLE Shell 3.10.9*
File Edit Shell Debug Options Window Help
Python 3.10.9 (tags/v3.10.9:1dd9be6, Dec 6 2022, 20:01:21) [MSC v.1934 64 bit (AMD64)]
on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> ===== RESTART: C:/Users/mdsur/Desktop/server.py =====
The server is ready to receive
Sent contents of server.py
The server is ready to receive

Ln: 1 Col: 1 Ln: 5 Col: 0
28°C Sunny 14:36
ENG IN 29-01-2023
```

The screenshot shows a Windows desktop environment with two windows open. On the left is a code editor window titled "server.py - C:/Users/mdsur/Desktop/server.py (3.10.9)". It contains Python code for a socket server. On the right is an "IDLE Shell 3.10.9" window, which is a terminal window for the Python interpreter. The terminal output shows the server is ready to receive data.

```
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()
```

```
*IDLE Shell 3.10.9*
File Edit Shell Debug Options Window Help
Python 3.10.9 (tags/v3.10.9:1dd9be6, Dec 6 2022, 20:01:21) [MSC v.1934 64 bit (AMD64)]
on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
=====
RESTART: C:/Users/mdsur/Desktop/server.py =====
The server is ready to receive
```

AIM:- Using UDP sockets, write a client server program to make client sending the file name and the server to send back the contents of requested file if present

client.py

```
from socket import *
sname = '127.0.0.1'; sport = 12000
csock = socket(AF_INET, SOCK_DGRAM)
msg = input("Enter file name")
csock.sendto(msg.encode("utf-8"), (sname, sport))
```

```
fout, eadd = csock.recvfrom(2048)
print(file.decode("utf-8"))
csock.close()
```

server.py

```
from socket import *
sname = '127.0.0.1'; sport = 12000
psock = socket(AF_INET, SOCK_DGRAM)
psock.bind(("127.0.0.1", sport))
print("Ready to Listen")
while 1:
    msg, eadd = psock.recvfrom(2048)
    msg = msg.decode("utf-8")
    file = open(msg, "r")
    l = file.read(2048)
    psock.sendto(l.encode("utf-8"), eadd)
    print(msg)
    file.close()
```



```
C:\Users\mdsur\Desktop\UDP>python -u serverUDP.py
The server is ready to receive
Sent contents of  serverUDP.py
|  
  
C:\Users\mdsur\Desktop\UDP>python -u clientUDP.py
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()  
  
C:\Users\mdsur\Desktop\UDP>
```