VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"JnanaSangama", Belgaum -590014, Karnataka.

LAB REPORT
on

COMPUTER NETWORKS

Submitted by

MANIKANTHA GADA (1BM20CS194)

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

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

B. M. S. College of Engineering,

Bull Temple Road, Bangalore 560019

(Affiliated To Visvesvaraya Technological University, Belgaum)

Department of Computer Science and Engineering

CERTIFICATE

This is to certify that the Lab work entitled "LAB COURSE **COMPUTER NETWORKS**" carried out by **MANIKANTHA GADA (1BM20CS194),** who is bonafide student of **B. M. S. College of Engineering.** It is in partial fulfillment for the award of **Bachelor of Engineering in Computer Science and Engineering** of the Visvesvaraya Technological University, Belgaum during the year 2022. The Lab report has been approved as it satisfies the academic requirements in respect of a **Computer Networks - (20CS5PCCON)** work prescribed for the said degree.

Dr. Nandhini Vineeth
Assistant Professor
Department of CSE
BMSCE, Bengaluru

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

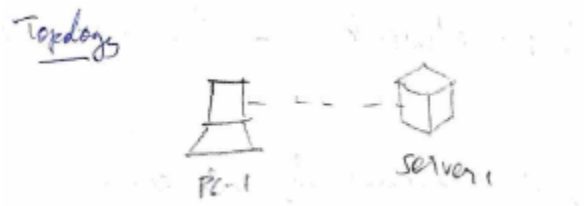
`

Index

Experiment No-1

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

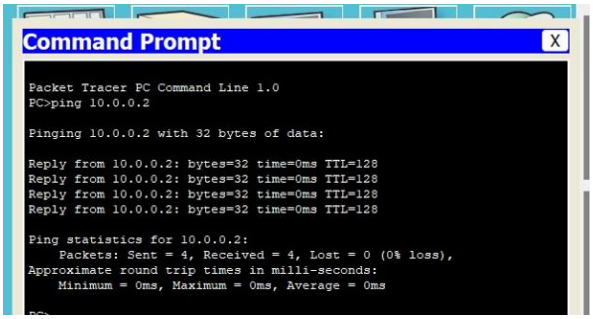
Topology



Procedure

- Launch Packet Tracer
- Create network with generic PC & generic Server
- Select copper Straight cable & connect PC & server
- Configure IP address
- Select simple PDU & click on both devices
- Finally click on auto capture & animation can be viewed of packet flow in simulation mode
- Real time mode, open command prompt & ping using commands & destination IP address

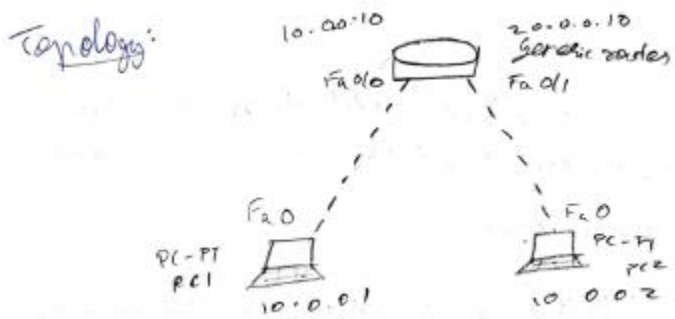
Snapshot of Output



Experiment No-2

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

Topology



Procedure

Procedure:

- Place a generic router & 2 generic PC's in the work space
- It can be found in the bottom left corner
- Connect the router & PCs using copper cross over
- Select it from connections
- Configure IP address for each PC & in the configuration tab of settings set gateway for both PCs to Routers
- Click on Generic Router & go to CCI Tab
- Enable following commands to set up a connection between PCs & generic router through gate way 10.0.0.10
- No
  enable
  configt
  interface Fastethernet 0/0
  ip address 10.00.10 255.0.0.0
  no shut
  exit
- Now to set up connection between PCs & the router through gateway 20.0.0.10
- Interface fastethernet 1/0
  IP address 20.0.0.10 255.0.0.0
  no shut
  exit
- The lights which were red until then will turn green now, indicating that the 2 devices are ready for communication

Simulation mode: Add a simple PDU by selecting the PC's & click on auto capture from right panel

Real time mode: Select the PC you want to send the packet & open its command prompt from desktop tab. Specify destination PC by specifying the dest^n address. A response is sent from destn PC to source PC.

Output

```
Cisco Packet Tracer PC Command Line 1.0
C:\>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 time<1ms TTL=127
Reply from 20.0.0.1: bytes=32 time<1ms TTL=127
Reply from 20.0.0.1: bytes=32 time<1ms TTL=127

Ping statistics for 20.0.0.1:
    Packets: Sent = 4, Received = 3, Lost = 1 (25% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 0ms, Average = 0ms

C:\>ping 20.0.0.1

Pinging 20.0.0.1 with 32 bytes of data:

Reply from 20.0.0.1: bytes=32 time<1ms TTL=127
Reply from 20.0.0.1: bytes=32 time<1ms TTL=127
Reply from 20.0.0.1: bytes=32 time<1ms TTL=127
Reply from 20.0.0.1: bytes=32 time<1ms TTL=127

Ping statistics for 20.0.0.1:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 0ms, Average = 0ms
```

```
PC>
PC>ping 20.0.0.10

Pinging 20.0.0.10 with 32 bytes of data:

Reply from 20.0.0.10: bytes=32 time=1ms TTL=255
Reply from 20.0.0.10: bytes=32 time=0ms TTL=255
Reply from 20.0.0.10: bytes=32 time=0ms TTL=255
Reply from 20.0.0.10: bytes=32 time=0ms TTL=255

Ping statistics for 20.0.0.10:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 1ms, Average = 0ms

PC>ping 30.0.0.10

Pinging 30.0.0.10 with 32 bytes of data:

Reply from 30.0.0.10: bytes=32 time=1ms TTL=254
Reply from 30.0.0.10: bytes=32 time=1ms TTL=254
Reply from 30.0.0.10: bytes=32 time=1ms TTL=254
Reply from 30.0.0.10: bytes=32 time=8ms TTL=254

Ping statistics for 30.0.0.10:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 1ms, Maximum = 8ms, Average = 2ms

PC>
```

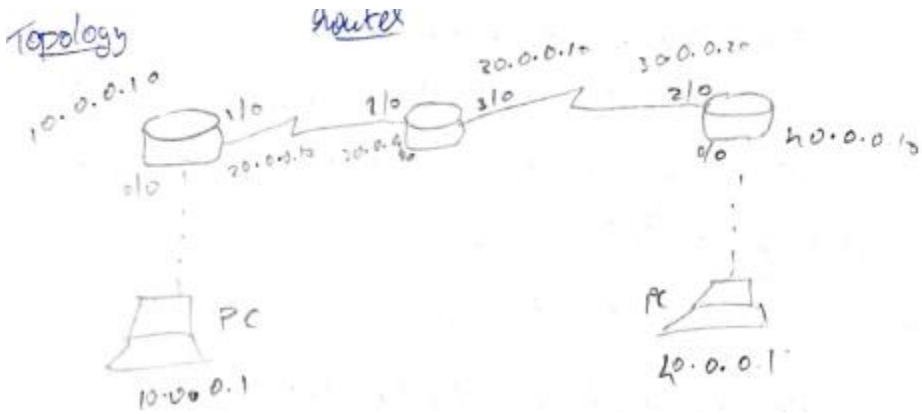
```
Packet Tracer PC Command Line 1.0
PC>ping 40.0.0.1

Pinging 40.0.0.1 with 32 bytes of data:

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

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

PC>
```

```
Request timed out.
Reply from 40.0.0.1: bytes=32 time=10ms TTL=125
Reply from 40.0.0.1: bytes=32 time=13ms TTL=125
Reply from 40.0.0.1: bytes=32 time=8ms TTL=125

Ping statistics for 40.0.0.1:
    Packets: Sent = 4, Received = 3, Lost = 1 (25% loss),
Approximate round trip times in milli-seconds:
    Minimum = 8ms, Maximum = 13ms, Average = 10ms

PC>ping 40.0.0.1
```

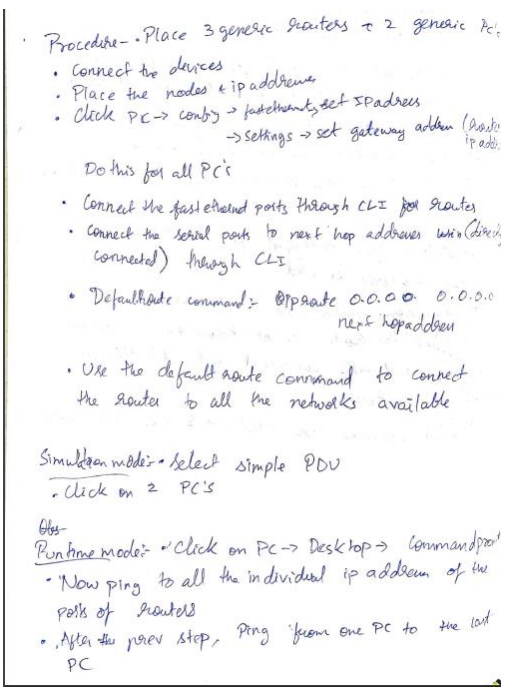
Experiment No-3

Aim: Configuring default route to the Router

Topology:



Procedure:



Procedure - • Place 3 generic routers + 2 generic PC's
• Connect the devices
• Place the nodes + ip addresses
• Click PC → config → fastethernet, set IP address
→ settings → set gateway address (router ip addr.)

Do this for all PC's

• Connect the fastethernet ports through CLI for routers
• Connect the serial ports to next hop addresses w/o (directly connected) through CLI
• Defaultroute command :- ip route 0.0.0.0 0.0.0.0 next hop address
• Use the default route command to connect the router to all the networks available

Simulation mode :- • Select simple PDU
• Click on 2 PC's

Obs-
Runtime mode :- • Click on PC → Desktop → Command prompt
• Now ping to all the individual ip addresses of the ports of routers
• After the prev step, ping from one PC to the last PC

Output:

```
IOS Command Line Interface
Router(config)#no shut
                ^
% Invalid input detected at '^' marker.

Router(config)#exit
Router#
%SYS-5-CONFIG_I: Configured from console by console
show ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, ia - IS-IS
inter area
       * - candidate default, U - per-user static route, o - ODR
       P - periodic downloaded static route

Gateway of last resort is 20.0.0.10 to network 0.0.0.0

C    20.0.0.0/8 is directly connected, Serial2/0
C    30.0.0.0/8 is directly connected, Serial3/0
C    50.0.0.0/8 is directly connected, FastEthernet0/0
S*   0.0.0.0/0 [1/0] via 20.0.0.10
              [1/0] via 30.0.0.20
Router#
```

Copy Paste

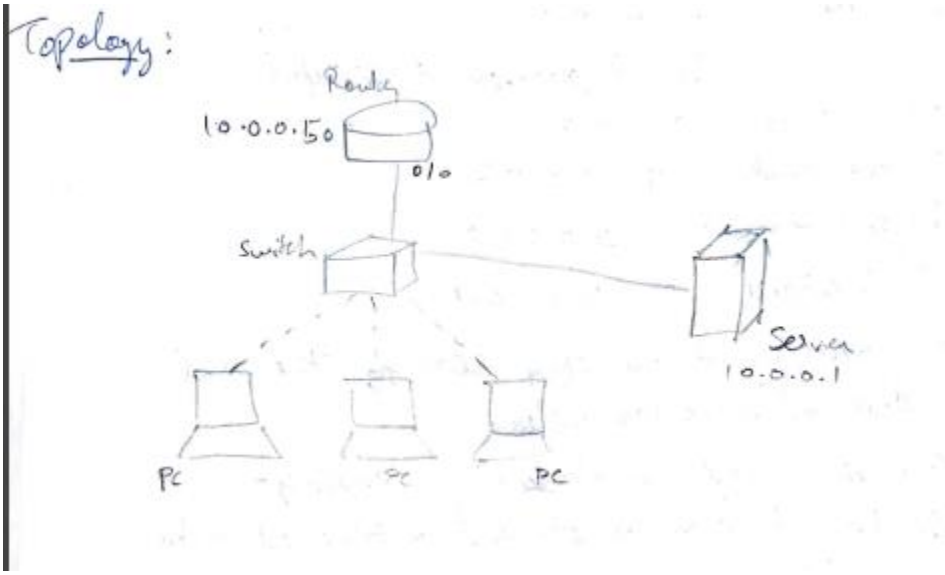
```
Packet Tracer PC Command Line 1.0
PC>ping 50.0.0.1

Pinging 50.0.0.1 with 32 bytes of data:

Request timed out.
Reply from 50.0.0.1: bytes=32 time=14ms TTL=126
Reply from 50.0.0.1: bytes=32 time=12ms TTL=124
Reply from 50.0.0.1: bytes=32 time=3ms TTL=124

Ping statistics for 50.0.0.1:
    Packets: Sent = 4, Received = 3, Lost = 1 (25% loss),
Approximate round trip times in milli-seconds:
    Minimum = 3ms, Maximum = 14ms, Average = 9ms

PC>ping 50.0.0.1

Pinging 50.0.0.1 with 32 bytes of data:

Reply from 50.0.0.1: bytes=32 time=2ms TTL=124
Reply from 50.0.0.1: bytes=32 time=2ms TTL=124
Reply from 50.0.0.1: bytes=32 time=11ms TTL=124
Reply from 50.0.0.1: bytes=32 time=2ms TTL=124

Ping statistics for 50.0.0.1:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 2ms, Maximum = 11ms, Average = 4ms
```

```
PC>ping 40.0.0.1

Pinging 40.0.0.1 with 32 bytes of data:

Request timed out.
Reply from 40.0.0.1: bytes=32 time=20ms TTL=125
Reply from 40.0.0.1: bytes=32 time=3ms TTL=125
Reply from 40.0.0.1: bytes=32 time=10ms TTL=125

Ping statistics for 40.0.0.1:
    Packets: Sent = 4, Received = 3, Lost = 1 (25% loss),
Approximate round trip times in milli-seconds:
    Minimum = 3ms, Maximum = 20ms, Average = 11ms

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=23ms TTL=125
Reply from 40.0.0.1: bytes=32 time=18ms TTL=125
Reply from 40.0.0.1: bytes=32 time=14ms TTL=125
Reply from 40.0.0.1: bytes=32 time=3ms TTL=125

Ping statistics for 40.0.0.1:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 3ms, Maximum = 23ms, Average = 14ms
```

Experiment No-4

Aim: Configuring DHCP within a LAN in a packet Tracer

Topology:



Procedure:

Procedure:- • Place 3 generic PC's, 1 generic Switch, 1 generic router & 1 generic server

• connect PC's to switch & to router, connect server to switch

• Click on router -> CLI -> Set the IP address of its port

• Click on server -> config -> set gateway address
-> Fastethernet -> Set IP address
-> Services -> DHCP

• In DHCP tab click ON, enter Pool name

-> set the gateway as router IP address & TFTP server with same IP address

-> Set DNS server as the IP address of the server

-> Enter Start IP address

-> Save

• Click on a PC -> Desktop -> IP configuration, select DHCP

• Ping PC to PC

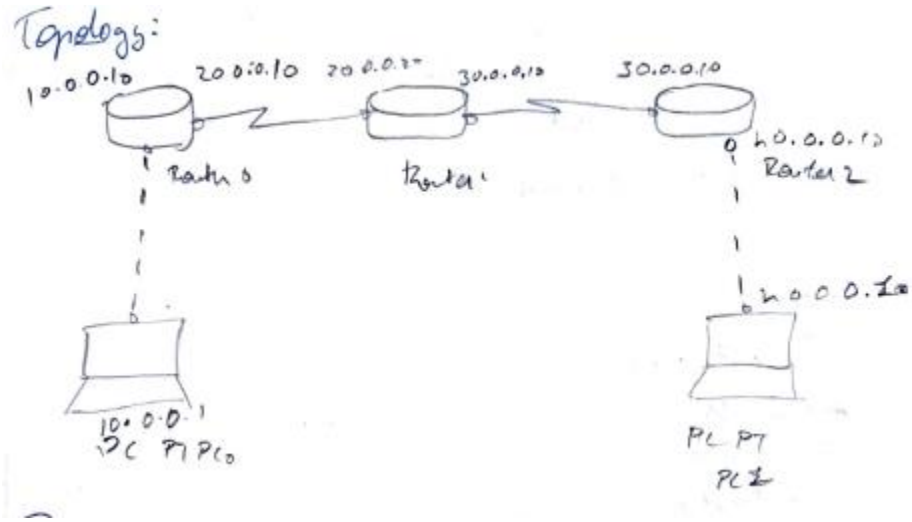
Output:

```
Packet Tracer PC Command Line 1.0
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=0ms TTL=128
Reply from 10.0.0.3: bytes=32 time=0ms TTL=128
Reply from 10.0.0.3: bytes=32 time=1ms TTL=128
Reply from 10.0.0.3: bytes=32 time=3ms TTL=128

Ping statistics for 10.0.0.3:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 3ms, Average = 1ms

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
Reply from 10.0.0.4: bytes=32 time=0ms TTL=128
Reply from 10.0.0.4: bytes=32 time=1ms TTL=128
Reply from 10.0.0.4: bytes=32 time=1ms 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 = 1ms, Average = 0ms
```

Experiment No-5

Aim: Configuring RIP Routing Protocol in Routers

Topology:



Procedure:

Procedure → place 3 generic routers & 2 generic PC's
* Connect routers to PC with copper straight
* Connect routers with serial DCE with clock symbol
* place notes near the devices
* set IP address for PC's & their subnet mask & default gateway
* CLI of router 0 → enable
→ config t
→ interface fastethernet 0/0
→ ip address 10.0.0.10 255.0.0.0
→ no shut
* Connection should turn green
* repeat for PC1 & router 2
* open CLI of router 0
→ enable
→ config t
→ interface serial 2/0
→ ip address 20.0.0.10 255.0.0.0
→ encapsulation PPP
→ clock rate 64000
→ no shut

* open CLI of router 1
→ enable
→ config t
→ interface serial 2/0
→ ip address 30.0.0.10 255.0.0.0
→ encapsulation PPP
→ clock rate 64000
→ no shut
→ exit
* open CLI of router 2
→ enable
→ config t
→ interface serial 2/0
→ ip address 30.0.0.20 255.0.0.0
→ encapsulation PPP
→ no shut
* open CLI of router 1
→ exit
Config # router RIP
Config router # network 10.0.0.0
Config - router # network 20.0.0.0
Config - router # exit

follow
* open CLI for router 1
→ enable
→ config t
→ interface serial 2/0
→ ip address 20.0.0.20 255.0.0.0
→ encapsulation PPP
→ no shut
* Connection turns green
* → exit
→ show ip route

→ show ip route
* Similarly repeat for router 1 & router 2 with network 20,30 & 30,40

Sim mode: Add simple PDU by selecting PC's & click auto capture

Real time mode: Select PC0 go to command prompt & select destn address [ping 10.0.0.10] then other addresses. finally ping PC2

Output:

```
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=255
Reply from 10.0.0.10: bytes=32 time=0ms TTL=255
Reply from 10.0.0.10: bytes=32 time=1ms TTL=255
Reply from 10.0.0.10: bytes=32 time=0ms TTL=255

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 = 1ms, Average = 0ms

PC>ping 20.0.0.10

Pinging 20.0.0.10 with 32 bytes of data:

Reply from 20.0.0.10: bytes=32 time=0ms TTL=255
Reply from 20.0.0.10: bytes=32 time=1ms TTL=255
Reply from 20.0.0.10: bytes=32 time=0ms TTL=255
Reply from 20.0.0.10: bytes=32 time=1ms TTL=255

Ping statistics for 20.0.0.10:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 1ms, Average = 0ms
```

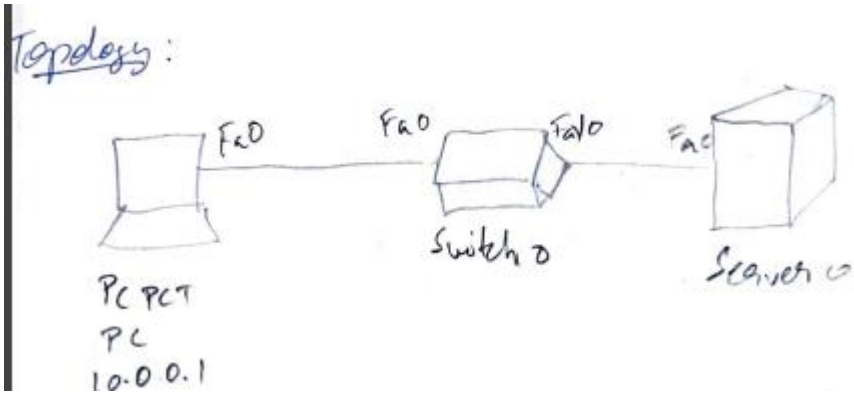
```
PC>ping 30.0.0.10

Pinging 30.0.0.10 with 32 bytes of data:

Reply from 30.0.0.10: bytes=32 time=9ms TTL=254
Reply from 30.0.0.10: bytes=32 time=11ms TTL=254
Reply from 30.0.0.10: bytes=32 time=6ms TTL=254
Reply from 30.0.0.10: bytes=32 time=6ms TTL=254

Ping statistics for 30.0.0.10:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 6ms, Maximum = 11ms, Average = 8ms

PC>ping 40.0.0.10

Pinging 40.0.0.10 with 32 bytes of data:

Reply from 40.0.0.10: bytes=32 time=2ms TTL=253
Reply from 40.0.0.10: bytes=32 time=19ms TTL=253
Reply from 40.0.0.10: bytes=32 time=2ms TTL=253
Reply from 40.0.0.10: bytes=32 time=8ms TTL=253

Ping statistics for 40.0.0.10:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 2ms, Maximum = 19ms, Average = 7ms
```

```
PC>ping 40.0.0.1

Pinging 40.0.0.1 with 32 bytes of data:

Request timed out.
Reply from 40.0.0.1: bytes=32 time=16ms TTL=125
Reply from 40.0.0.1: bytes=32 time=18ms TTL=125
Reply from 40.0.0.1: bytes=32 time=13ms TTL=125

Ping statistics for 40.0.0.1:
    Packets: Sent = 4, Received = 3, Lost = 1 (25% loss),
Approximate round trip times in milli-seconds:
    Minimum = 13ms, Maximum = 18ms, Average = 15ms

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=2ms TTL=125
Reply from 40.0.0.1: bytes=32 time=2ms TTL=125
Reply from 40.0.0.1: bytes=32 time=12ms TTL=125
Reply from 40.0.0.1: bytes=32 time=19ms TTL=125

Ping statistics for 40.0.0.1:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 2ms, Maximum = 19ms, Average = 8ms
```

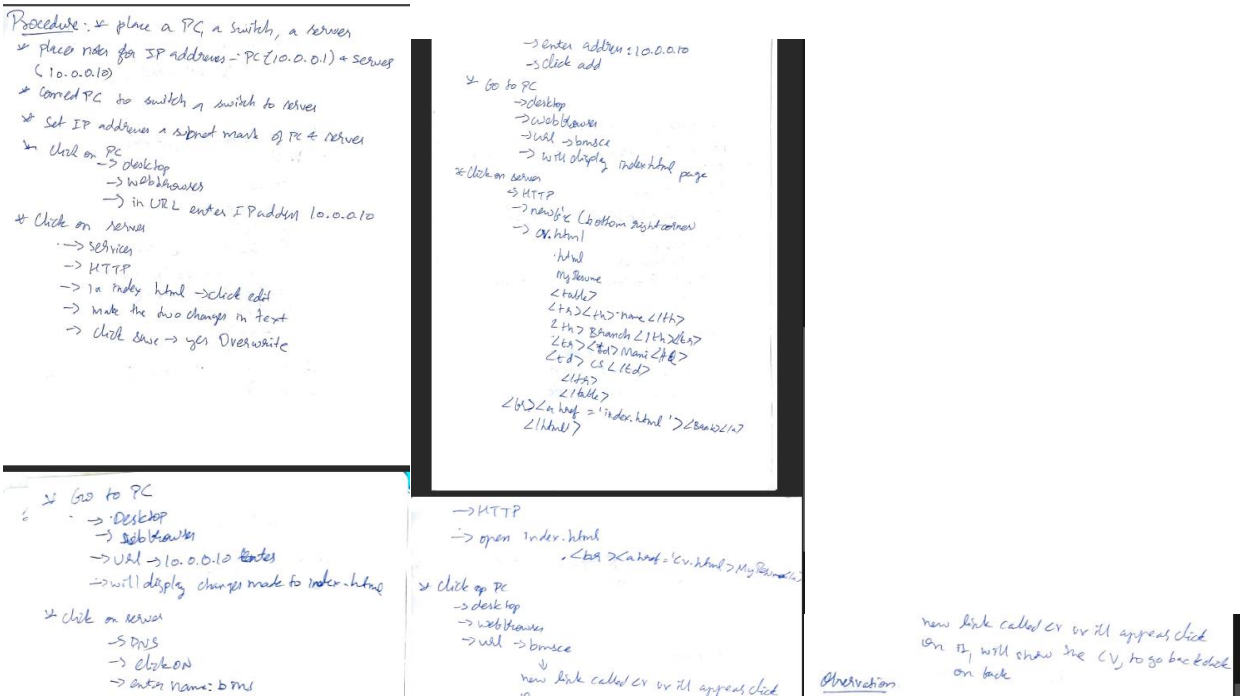
Experiment No-6

Aim: Demonstration of WEB server and DNS using Packet Tracer

Topology:

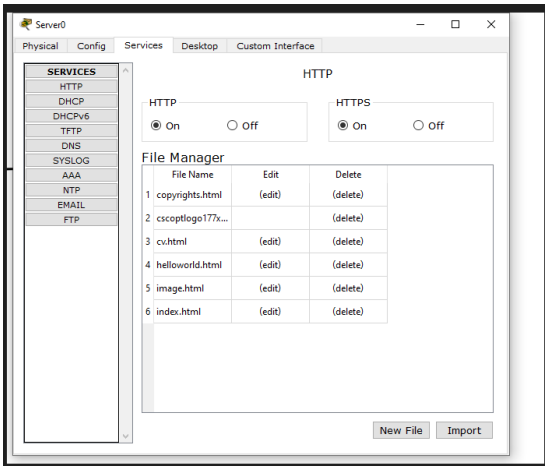


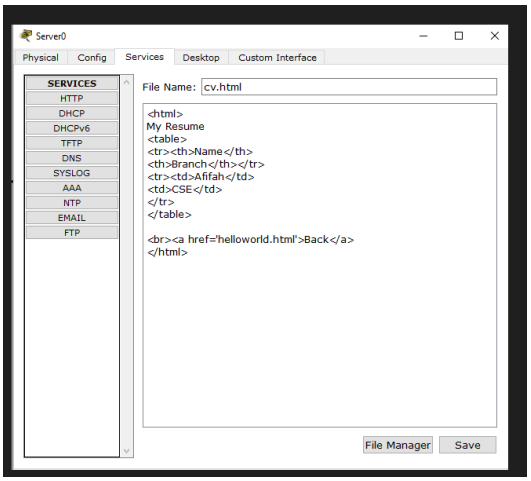
Procedure



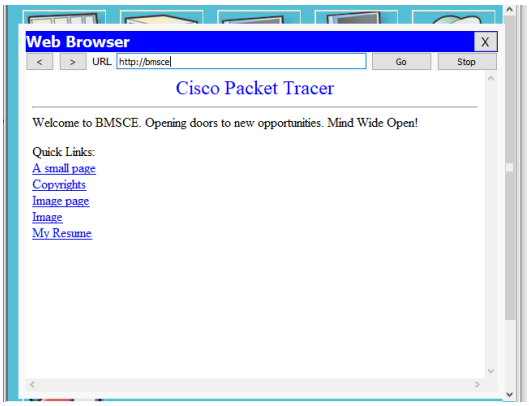
Output


Web Browser
URL http://10.0.0.10
Go
Stop
Cisco Packet Tracer
Welcome to BMSCE. Opening doors to new opportunities. Mind Wide Open!
Quick Links:
A small page
Copyrights
Image page
Image
My Resume


Server0
Physical
Config
Services
Desktop
Custom Interface
SERVICES
HTTP
DHCP
DHCPv6
TFTP
DNS
SYSLOG
AAA
NTP
EMAIL
FTP
HTTP
HTTP
On
Off
HTTPS
On
Off
File Manager
File Name
Edit
Delete
1 copyrights.html (edit) (delete)
2 cscoptlogo177x... (delete)
3 cv.html (edit) (delete)
4 helloworld.html (edit) (delete)
5 image.html (edit) (delete)
6 index.html (edit) (delete)
New File
Import


Server0
Physical
Config
Services
Desktop
Custom Interface
SERVICES
HTTP
DHCP
DHCPv6
TFTP
DNS
SYSLOG
AAA
NTP
EMAIL
FTP
File Name: cv.html
<html>
My Resume
<table>
<tr><th>Name</th>
<th>Branch</th></tr>
<tr><td>Afifah</td>
<td>CSE</td>
</tr>
</table>

Back
</html>
File Manager
Save


Web Browser
URL http://bmsce
Go
Stop
Cisco Packet Tracer
Welcome to BMSCE. Opening doors to new opportunities. Mind Wide Open!
Quick Links:
A small page
Copyrights
Image page
Image
My Resume

CYCLE 2

Experiment No-1

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

Code:
```
#include<stdio.h>
#include<stdlib.h>
#include<unistd.h>

#include <iostream>
#include <string.h>

using namespace std;

int crc(char *ip, char *op, char *poly, int mode)
{
strcpy(op, ip);
if (mode) {
for (int i = 1; i < strlen(poly); i++)
strcat(op, "0");
}
/* Perform XOR on the msg with the selected polynomial */
for (int i = 0; i < strlen(ip); i++) {
if (op[i] == '1') {
for (int j = 0; j < strlen(poly); j++) {
if (op[i + j] == poly[j])
op[i + j] = '0';
else
op[i + j] = '1';
}
}
}
/* check for errors. return 0 if error detected */
for (int i = 0; i < strlen(op); i++)
if (op[i] == '1')
return 0;
return 1;
}

int main()
{
char ip[50], op[50], recv[50];
char poly[] = "10001000000100001";

cout << "Enter the input message in binary"<< endl;
cin >> ip;
```

```
crc(ip, op, poly, 1);
cout << "The transmitted message is: " << ip << op + strlen(ip) << endl;
cout << "Enter the recevied message in binary" << endl;
cin >> recv;
if (crc(recv, op, poly, 0))
cout << "No error in data" << endl;
else
cout << "Error in data transmission has occurred" << endl;

return 0;
}
```

Output:

```
Output
/tmp/uztSwsRnax.o
Enter the input message in binary
11111
The transmitted message is: 111111110001111011110
Enter the recevied message in binary
11111
No error in data
```

Experiment No-2

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

Code:

```
#include <iostream>
#include <stdio.h>

using namespace std;

struct node {
    int dist[20];
    int from[20];
} route[10];

int main()
{
    int dm[20][20], no;

    cout << "Enter no of nodes." << endl;
    cin >> no;
    cout << "Enter the distance matrix:" << endl;
    for (int i = 0; i < no; i++) {
        for (int j = 0; j < no; j++) {
            cin >> dm[i][j];
            /*  Set distance from i to i as 0 */
            dm[i][i] = 0;
            route[i].dist[j] = dm[i][j];
            route[i].from[j] = j;
        }
    }

    int flag;
    do {
        flag = 0;
        for (int i = 0; i < no; i++) {
            for (int j = 0; j < no; j++) {
                for (int k = 0; k < no; k++) {
                    if ((route[i].dist[j]) > (route[i].dist[k] + route[k].dist[j])) {
                        route[i].dist[j] = route[i].dist[k] + route[k].dist[j];
                        route[i].from[j] = k;
                        flag = 1;
                    }
```

```
            }
        }
    }
  } while (flag);

  for (int i = 0; i < no; i++) {
    cout << "Router info for router: " << i + 1 << endl;
    cout << "Dest\tNext Hop\tDist" << endl;
    for (int j = 0; j < no; j++)
      printf("%d\t%d\t\t%d\n", j+1, route[i].from[j]+1, route[i].dist[j]);
  }
  return 0;
}
```

Output

```
Enter the number of nodes : 7

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


 For router 1

node 1 via 1 Distance 0
node 2 via 6 Distance 0
node 3 via 3 Distance 0
node 4 via 2 Distance 0
node 5 via 5 Distance 0
node 6 via 6 Distance 0
node 7 via 7 Distance 0

 For router 2

node 1 via 6 Distance 0
node 2 via 2 Distance 0
node 3 via 1 Distance 0
node 4 via 4 Distance 0
node 5 via 1 Distance 0
node 6 via 6 Distance 0
node 7 via 7 Distance 0
```

Experiment No-3

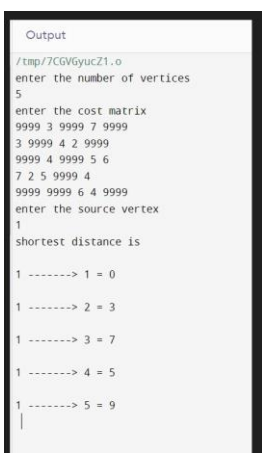
Aim: Implement Dijkstra's algorithm to compute the shortest path for a given topology

Code:

```
#include<stdio.h>
#include<conio.h>

int c[10][10],n,src;
void dijkistra();
int main()
{
   printf("\nenter the number of vertices\n");
   scanf("%d",&n);
   printf("\nenter the cost matrix \n");
   for(int i=1;i<=n;i++)
   {
      for(int j=1;j<=n;j++)
      {
         scanf("%d",&c[i][j]);
      }
   }
   printf("\nenter the source vertex\n");
   scanf("%d",&src);
   dijkistra();
   return 1;
}

void dijkistra()
{
   int dist[10],vis[10],j,count,min,u;
   for(j=1;j<=n;j++)
   {
      dist[j]=c[src][j];
   }
   for(j=1;j<=n;j++)
   {
      vis[j]=0;
   }
   dist[src]=0;
   vis[src]=1;
```

```
    count=1;
    while(count!=n)
    {
        min=9999;
        for(j=1;j<=n;j++)
        {
            if(dist[j]<min && vis[j]!=1)
            {
                min=dist[j];
                u=j;
            }
        }
        vis[u]=1;
        count++;
        for(j=1;j<=n;j++)
        {
            if(min+c[u][j]<dist[j] && vis[j]!=1)
            {
                dist[j]=min+c[u][j];
            }
        }
    }
    printf("\n shortest distance is \n");
    for(j=1;j<=n;j++)
    {
        printf("\n%d -------> %d = %d \n ",src,j,dist[j]);
    }
}
```

Output:



```
Output
/tmp/7CGVGyucZ1.o
enter the number of vertices
5
enter the cost matrix
9999 3 9999 7 9999
3 9999 4 2 9999
9999 4 9999 5 6
7 2 5 9999 4
9999 9999 6 4 9999
enter the source vertex
1
shortest distance is

1 -------> 1 = 0

1 -------> 2 = 3

1 -------> 3 = 7

1 -------> 4 = 5

1 -------> 5 = 9
```

Experiment No-4

Aim: Write a program for congestion control using Leaky bucket algorithm.

Code:
```
#include <iostream>
using namespace std;

int main() {
   int capacity=0,packet=0,bsize=0,rate=0;
   char ans='y';
   cout<<"enter the bucket size : ";
   cin>>capacity;
   cout<<"enter the leaking rate : ";
   cin>>rate;
   while(ans=='y')
   {
      cout<<"\nenter the packet size : ";
      cin>>packet;
      if((bsize+packet) > capacity)
      {
         cout<<"\n buffer full at the moment ";
      }

      else if((bsize+packet) <= capacity)
      {
         bsize+=packet;
      }
      bsize-=rate;
      cout<<"remaining bucket capacity is "<<bsize;
      cout<<"\ndo you wish to keep adding packets? y/n : ";
      cin>>ans;
   }

   return 0;
}
```

Output:

Output USN : 1BM20CS195

```
/tmp/ML8IKt2j4J.o
enter the bucket size : 70
enter the leaking rate : 2
enter the packet size : 20
remaining bucket capacity is 18
do you wish to keep adding packets? y/n : y
enter the packet size : 20
remaining bucket capacity is 36
do you wish to keep adding packets? y/n : y
enter the packet size : 20
remaining bucket capacity is 54
do you wish to keep adding packets? y/n : y
2enter the packet size : 0
remaining bucket capacity is 52
do you wish to keep adding packets? y/n : y
enter the packet size : 18
remaining bucket capacity is 68
do you wish to keep adding packets? y/n : y
enter the packet size : 4
buffer full at the moment remaining bucket capacity is 66
do you wish to keep adding packets? y/n : n
```

•

Experiment No-5

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

Code:

```
from socket import *
serverName='DESKTOP-9CJQB77'
serverPort=12530
clientSocket=socket(AF_INET,SOCK_STREAM)
clientSocket.connect((serverName,serverPort))
sentence=input("Enter file name")
clientSocket.send(sentence.encode())
filecontents=clientSocket.recv(1024).decode()
print('From Server:',filecontents)
clientSocket.close()
```

```
from socket import *
serverName='DESKTOP-9CJQB77'
serverPort=12530
serverSocket=socket(AF_INET,SOCK_STREAM)
serverSocket.bind((serverName,serverPort))
serverSocket.listen(1)
print("The server is ready to receive")
while(1):
    connectionSocket,addr=serverSocket.accept()
    sentence=connectionSocket.recv(1024).decode()
    file=open(sentence,"r")
    l=file.read(1024)
    connectionSocket.send(l.encode())
    file.close()
    connectionSocket.close()
```

Output:

```
IDLE Shell 3.11.0
File  Edit  Shell  Debug  Options  Window  Help
Python 3.11.0 (main, Oct 24 2022, 18:26:48) [MSC v.1933 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
============ RESTART: C:/Users/BMSCE/Desktop/1BM20CS184/clienttcp.py ===========
Enter file nameabc.txt
From Server: Hi Hello
>>>
```

```
*IDLE Shell 3.11.0*
File  Edit  Shell  Debug  Options  Window  Help
Python 3.11.0 (main, Oct 24 2022, 18:26:48) [MSC v.1933 64 b
it (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more
information.
>>> import socket
>>> socket.gethostname()
'DESKTOP-9CJQB77'
>>>
== RESTART: C:/Users/BMSCE/Desktop/1BM20CS184/servertcp.py =
The server is ready to receive
```

Experiment No-6

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 the requested file if present.

Code:

```
from socket import *
serverName = "127.0.0.1"
serverPort = 12000
clientSocket = socket(AF_INET, SOCK_DGRAM)
sentence = input("\nEnter file name: ")
clientSocket.sendto(bytes(sentence,"utf-8"),(serverName, serverPort))
filecontents,serverAddress = clientSocket.recvfrom(2048)
print ('\nReply from Server:\n')
print (filecontents.decode("utf-8"))
# for i in filecontents:
 # print(str(i), end = '')
clientSocket.close()
clientSocket.close()

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()
```

Output:

```
*IDLE Shell 3.11.1*
File  Edit  Shell  Debug  Options  Window  Help
Python 3.11.1 (tags/v3.11.1:a7a450f, Dec  6 2022, 19:58:39) [MSC v.1934 64 bit (
AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
==== RESTART: C:/Users/HP/AppData/Local/Programs/Python/Python311/server.py ====
The server is ready to receive

Sent contents of  hello.txt
```

```
IDLE Shell 3.11.1
File  Edit  Shell  Debug  Options  Window  Help
Python 3.11.1 (tags/v3.11.1:a7a450f, Dec  6 2022, 19:58:39) [MSC v.1934 64 bit (
AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
==== RESTART: C:/Users/HP/AppData/Local/Programs/Python/Python311/client.py ====

Enter file name: hello.txt

Reply from Server:

Hello world
>>>
```