lab - 1

Aim : To create topology and simulate a single PDU from source to destination using a hub and a switch as connecting devices

Topology : star topology

Procedure

1. create a generic hub and switch
2. Add generic PC's, connect 4 to hub 4 to switch
3. Connect them using copper straight cables.
4. Places nodes with IP address assigned

Result : Message transmission b/w every devices is successful

Observation :-

1. PDU is first sent to the
2. hub and hub will broadcast to all the devices connected to it, if any of the receiving device is destination it will read message otherwise discard it.
3. Initially switch will broadcast to all the ports. Fill the details of IP address & ports in a table and later on this table is used to broadcast a message to particular port.

lab - 2

Aim: Configuring IP addresses to routers in packet ~~devices~~ tracer, Explore ping response destination, unreachable, reply request time out

Procedure :- 1) End devices are connected to router of IP address configured to end devices

2) Config IP addresses & subnet mask using command line interface.
   set interface 10.0.0.2 255.0.0.0

3) Gateway is configured for end devices

4) End devices and interfaces are pinged to check connection

Topology : star topology

Result :- Successfully pinged

Ping 20.0.0.1

Pinging 20.0.0.1 with 32 bytes of data

Request time out

Request time out

Ping statistics for 20.0.0.1

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

After connecting networks

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 = 125
Reply from 20.0.0.1 : bytes = 32 time = 21ms
TTL = 125
Reply from 20.0.0.1 bytes = 32 time = 21ms
TTL = 125
Reply from 20.0.0.1 bytes = 32 time = 21ms
TTL = 125

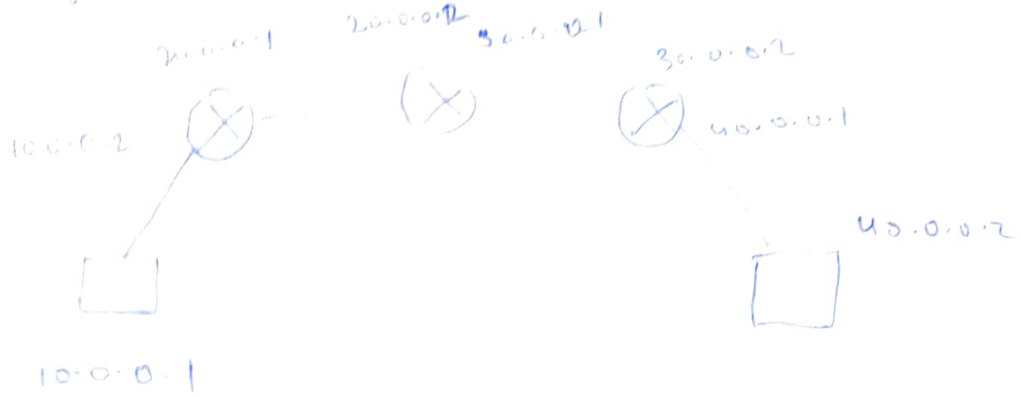
Ping statistics for 20.0.0.1
Packets : sent = 4, received = 4, lost = 0
Minimum = 2ms, Maximum = 21ms
Average = 11ms

Result. Successful ping only on connecting networks manually

Lab-3

Aim: To configure static ip address to the router

Configuration



Procedure
- Configure ip address of PCs
- Configure router interface
- Set gateway address of PCs
- for each router, check network it is directly connected
- Connect to other network in config mode.

Observation

Befor manually connecting networks

Ping 40.0.0.1

Pinging 40.0.0.1 with 32 bytes of data

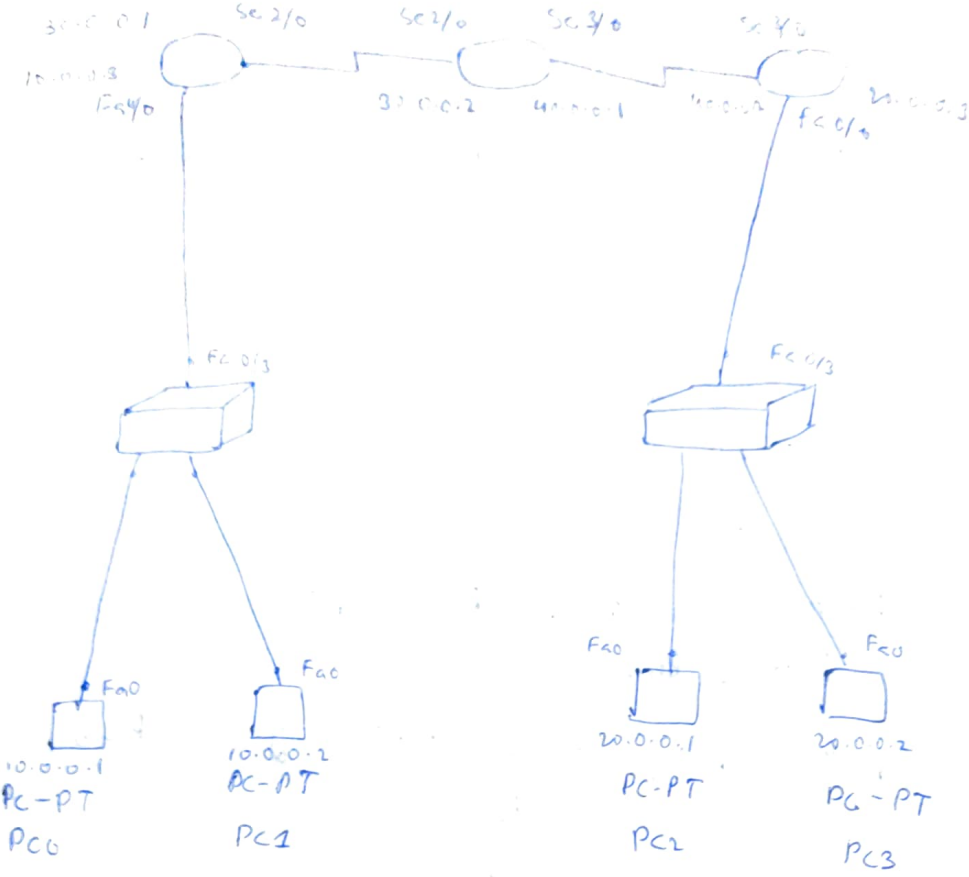
Reply from 10.0.0.2 destination host unreachable

PC > Ping 20.0.0.1

Pinging 20.0.0.1 with 32 bytes of data

Lab-4

Configuration



Aim: Configuring DHCP within a LAN in a packet tracer

Procedure: 1) Configure IP address of all four PC. Keep network ID of first two same and last two same

2) Connect PC's to switches using automatic wire connection

3) Take 3 routers config their IP address keeping network ID of two adjacent as same.

4) Connect routers to each other using serial DCE

5) Connect router to switch using fast ethernet connection.

observation: If we try pinging router 1 from PC 1

Ping 30.0.0.1

Pinging 30.0.0.1 with 32 bytes of data:

Request timed out
Request timed out
Request timed out
Request timed out

Ping statistics for 30.0.0.1

Packets: sent = 4, Received = 0, lost = 4 (100% loss)

If we ping end device from PC0

Ping 20.0.0.1

Pinging 20.0.0.1 with 32 bytes of data

Reply from ~~20.0.0.3~~:
Destination host unreachable
Reply from 10.0.0.3
Destination host unreachable
Reply from 10.0.0.3
Destination host unreachable

If we configure router with command Ip route 0.0.0.0 0.0.0.0 Destination address and then Ping destination host:

Ping 20.0.0.1

Reply from 20.0.0.1, bytes = 32
time = 2ms TTL = 125

Ping statistics for 10.0.0.1

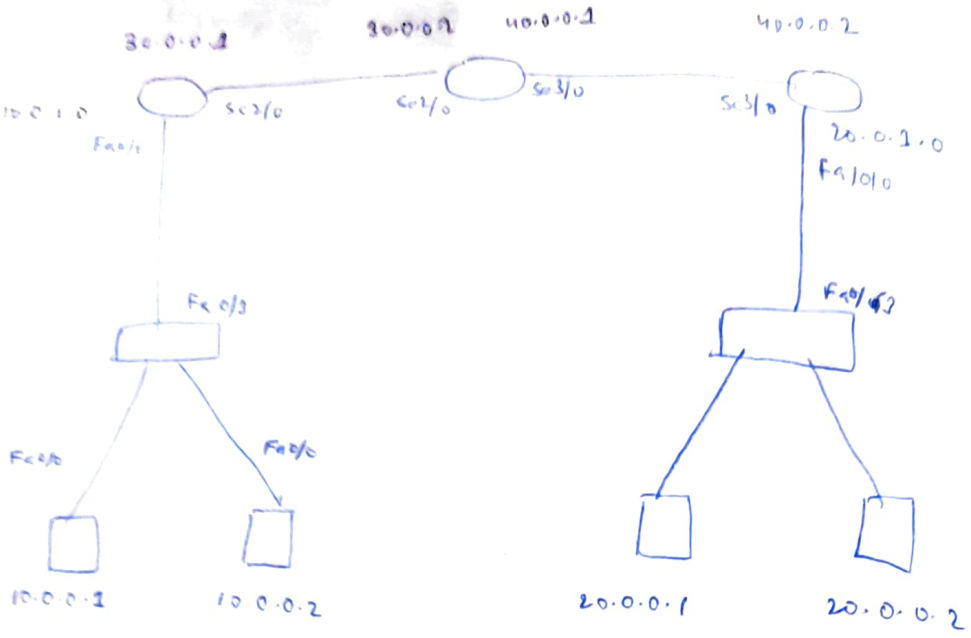
Packets: sent = 4

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

Neelima
1/12/2022

Lab - 5

Aim: Configure RIP (Routing Information protocol) using packet tracer



Procedure :- ① Configure IP address of all the end devices.

② Connect end devices to switches using fast ethernet and router to switch using fast ethernet

③ Connection b/w routers is made using serial DCE

④ Configure IP address of all the routers for 1st router config ip address.

⑤ Now IP address of all routers is configured

⑥ Now we need to encapsulate Point to Point Protocol (PPP). That is done only for the connection b/w two routers
encapsulation PPP.
clock rate 64000.

⑦ Perform sixth step for all routers.

⑧ Now for each router go to CLI type router rip and then

network Id should be different for all routers

observation Ping 20.0.0.1.

Pinging 20.0.0.1 with 32 bytes of data:

Reply from 20.0.0.1 bytes = 32 time = 15 ms TTL = 125

Reply from 20.0.0.1 bytes = 32 time = 1 ms TTL = 125

Reply from 20.0.0.1 bytes = 32 time = 2 ms TTL = 125

Reply from 20.0.0.1 bytes = 32 time = 2 ms TTL = 125

Ping statistics for 20.0.0.1:

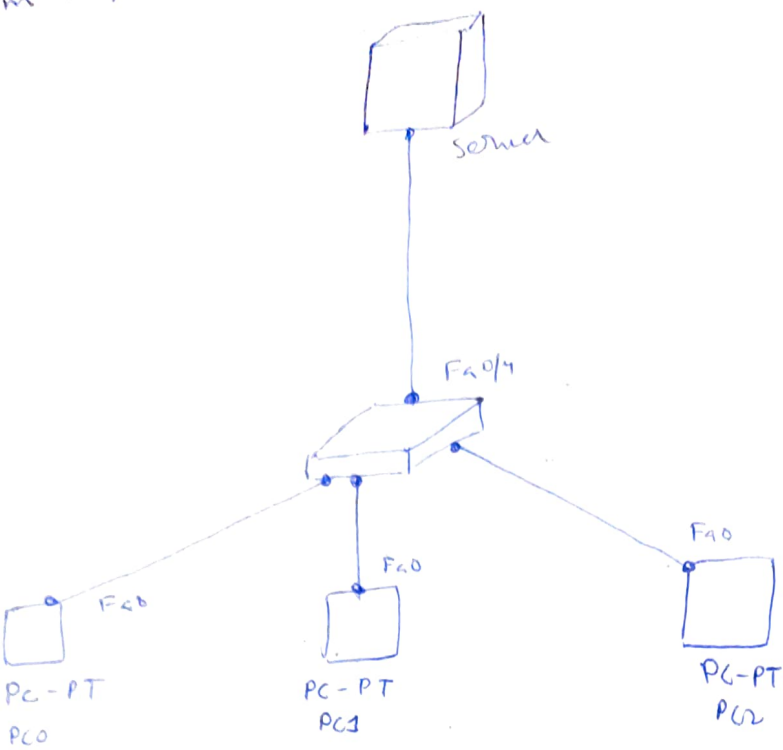
Packets: Sent = 4, Received = 4, Lost = 0 (0% lost)

Approximate round trip times in milli-seconds

Minimum = 2 ms Max = 15 ms Avg = 7 ms.

Ndelima
8/12/2021

Aim : How to configure DHCP.



Topology : Star

Procedure : ① Make connections among all the devices using automatic connection type.

② Configure IP address of server with any IP address.

③ Now go to services and on left side click on DHCP and assign a start IP address and turn ON service.

④ Now click on individual devices and go to config interface set IP configuration to DHCP from static.

⑤ Now ping other end devices from any one end device.

Observation: 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 = 1 ms
TTL = 128

Reply from 10.0.0.3 bytes = 32 time = 6 ms
TTL = 128

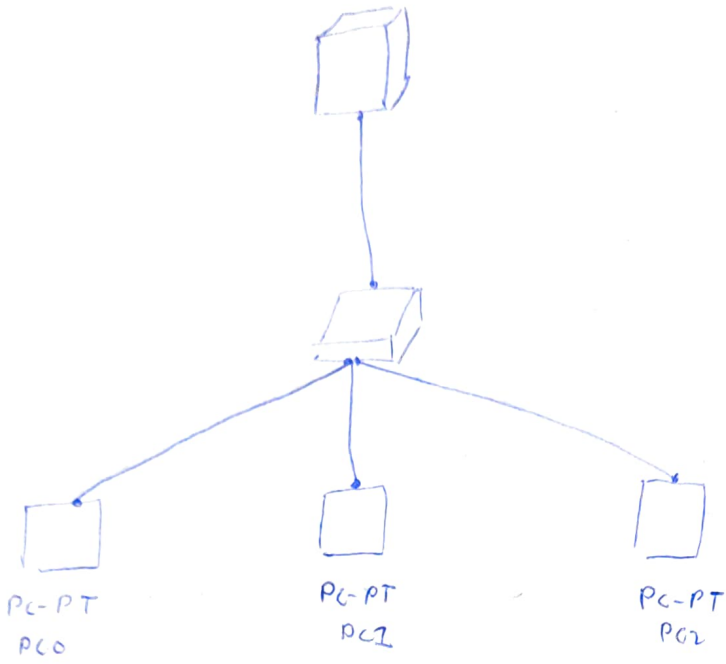
Reply from 10.0.0.3 bytes = 32 time = 4 ms
TTL = 128

Reply from 10.0.0.3 bytes = 32 time = 0 ms
TTL = 128

Ping statistics for 10.0.0.3:
Packets: sent = 4, Received = 4, lost = 0 (0% loss)
Approximate round trip in milli-seconds
minimum = 0 ms, maximum = 6 ms, Average = 2 ms

Neeling
15/12/2022

Aim: How to configure Web server and DNS server



Topology: Star

Procedure: ① Configure IP address of server

② Go to DNS and turn ON DNS service

③ Now give a name ~~to~~ like www.bms.com and just ~~that~~ give an IP address.

④ Click on add

⑤ Now Open any end device go to desktop and click on Web browser

⑥ Enter URL you just save and click on go.

⑦ Now you will see the default webpage rendered.

Observation:

URL | www.bmsce.com | Go | Stop

CISCO PACKET TRACER

Welcome to cisco Packet tracer. opening doors to new opportunities. Mind wide open.

Neelema
15/12/2022

```
def xor(a,b):
    result = []
    For i in range(1, len(b)):
        if a[i] == b[i]:
            result.append('0')
        else:
            result.append('1')
    return ''.join(result)

def mod2div(dividend, divisor):
    pick = len(divisor)
    tmp = dividend[0 : pick]
    While pick < len(dividend):
        if tmp[0] == '1':
            tmp = xor(divisor, tmp) +
                dividend[Pick]
        else:
            tmp = xor('0'*pick, tmp) + dividend
                [pick]
        pick += 1
    if tmp[0] == '1':
        tmp = xor(divisor, tmp)
    else:
        tmp = xor('0'*pick, tmp)
    checkboard = tmp
    return checkboard

def encodeData(data, key):
        l_key = len(key)
        append_data = data + '0'*(l_key-1)
```

```
    remainder = mode2div(appended_data, key)
    codeword = data + remainder
    print("Remainder : ", remainder)
    print("Encoded data (data + Remainder) :",
          codeword)

data = "100100"
key = "1000100000000100001"
encoded_data(data, key)
```

02/01/2023

8 Program for distance vector algorithm to find suitable path for transmission.

```
#include <stdio.h>
struct node
{   unsigned dist[20];
    unsigned From[20];
} rt[10];
int main()
{   int costmat[20][20];
    int nodes, i, j, k, count=0;
  printf("Enter no. of nodes:");
  scanf("%d", &nodes)
  printf("Enter the cost matrix");
 For(i=0; i<nodes; i++)
 {
     For(j=0; j<nodes; j++)
     {  scanf("%d", &costmat[i][j]);
        costmat[i][i]=0;
       rt[i].dist[j] = costmat[i][j];
       rt[i].From[j] =j;
     }
 }
   do
     {   count=0;
      For(i=0; i<nodes; i++)
      For(j=0; j<nodes; j++)
      For(k=0; k<nodes; k++)
          if (rt[i].dist[j] > costmat[i][k]
                  +rt[k].dist[j])
          {
```

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

06/01/2023

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

```
#include<stdio.h>
void dijstra();
int c[10][10], n, src;

void main()
{ int i,j;
  printf("Enter no. of vertices");
  scanf("%d", &n);
  printf("Enter the cost matrix");
  For (i=1; i<=n; i++)
    {
      For (j=1; j<=n; j++)
      { scanf ("%d", &c[i][j]);
      }
    }
printf("Enter the source node");
 scanf ("%d", &src);
  dijstra();
   getch()
}

void dijstra()
{ int vis[20], dist[10], u, j, count, min;
    For (j=1; j<=n; j++)
      {   dist[j] = c[src][j];
      }
      For (j=1; j<=n; j++)
        {   vis[j]=0;
        }
```

```
                dist[ ] = 0;
                vis[ ] = 1;
                count = 1;
            while ( count != n )
                min = 9999;
                for ( dist[ ] != 0 && vis[ ] == 0 )
                    if ( min > dist[j] )
                        min = dist[j];
                        u = j;
                    }
            }
            vis[u] = 1;
            count++;
        for ( j = 1; j <= n; j++ )
            {
                if ( min + c[u][j] < dist[j] && vis[j] == 0 )
                {
                    dist[j] = min + c[u][j];
                }
            }
        }
    printf("The shortest distance is:\n");
    for ( j = 1; j <= n; j++ )
        {
            printf("%d -> %d = %d", s, j, dist[j]);
        }
    }
```

10/01/2023

Implement leaky bucket algorithm.

```
#include<stdio.h>
#include<stdlib.h>
void bucket(int send);
int bucketsize=30, bucketmax=60;
int bucketrate=3;
int main()
{
    int i=0;
    while(i<10)
    {
        printf("1. send packet 2. Nothing to send 3. exit 4. Enter your choice");
        int ch;
        scanf("%d", &ch);
        switch(ch)
        {
            case 1:
                printf("Enter packet size required to be sent");
                int send;
                scanf("%d", &send);
                if(send<bucketmax-bucketsize)
                {
                    bucket(send);
                    printf("Packet sent successfully");
                }
                else
                {
                    Printf("error");
                    bucketoverflow(send);
                }
                printf("Bucket size = %d\n", bucketsize);
                break;
            Case 2:
```

```
            Printf("No packet sent");
            bucket size -= 3;
            printf(" Bucket size = %d", bucket size);
            break;
        case 3:
                exit(0);
                break;
        default:
                printf("Invalid option");
        }
        i += 1;
    }
    return 0;
}
void bucket(int send)
{
    bucket size += send;
    bucket size -= 3;
}
void bucket overFlow(int send)
{
    bucket size -= 3;
}
```

TCP/IP Socket

client.py

```
from socket import *
servername = "Desktop-0077SAC"
serverPort = 12004
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()
```

server.py

```
From socket import *
servername = "Desktop-0077SAC"
serverport = 12004
serverSocket = socked(AF_INET, SOCKET_STREAM)
ServerSocket.bind((servername, serverport))
Server.Socket.list(1)
print("this server is ready to receive")
While 1:
    connectionSocked, addr = ServerSaket.accept()
    sentence = connectionSocket.recv(1024).decode()
    File.open(Sentence, 'r')
    l = File.read(1024)
    print("Received File from client:", l)
connectionSocket.send(l.encode())
file.close()
connectionSocket.close()
```

UDP Sockets

client.py

```
From socket import *
serverName = "127.0.0.1"
serverPort = 12000
clientSocket = socket(AF_INET, SOCK_DGRAM)
sentence = input("Enter file name")
clientSocket.sendto(bytes(sentence,
        (serverName, serverPort))
FileContents, serverAddress = clientSocket.
                recvFrom(2048)
print("Reply from Server")
print(FileContents.decode("utf-8"))

clientSocket.close()
clientSocket.close()
```

Server.py

```
From socket import *
serverPort = 12000
serverSocket = socket(AF_INET, SOCK_DGRAM)
serverSocket.bind("127.0.0.1", serverPort)
print("server ready to receive")
while 1:
    sentence, clientAddress = serverSocket.
                recvfrom(1024)
    sentence = sentence.decode("utf-8")
    file = open(sentence, "r")
    l = file.read(2048)
```

```
server.socket.sendto(bytes, 'UTF-8',
                     client_address)
print("sent contents:")
file.close()
```