









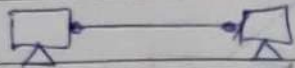


Sanjeet P. Paudyal
(18M22CS241)

Date / /
Page

	<u>Date</u>	<u>Title</u>	<u>Sign</u>
(1)	27/9	lab1	
(2)	4/10	lab2	
(3)	18/10	lab3	
(4)	25/10	lab4	
(5)	8/11	lab5	
(6)	15/11	lab6	 15/11
(7)	22/11	lab7	 22/11
(8)	29/11	lab8	 26/12
(9)	20/12	lab9	 26/12
(10)	20/12	lab10	 26/12

- (1) Aim: To explore functionality of network devices by comparing operations of switch and hub in simulated LAN using Cisco packet tracer.

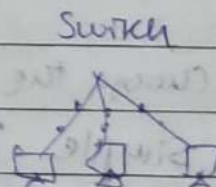
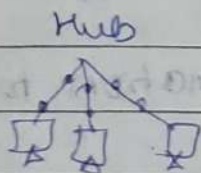


Implementation:

- select end device which was generic one, PC-PT-1 and PC-PT-2
- configure it with IP address 10.0.0.1 & 10.0.0.2
- It creates a subnet mask of 255.0.0.0
- Using Automatic connection, two end device connected
- Msg (Simple PDU) was passed from PC-PT-1 to PC-PT-2

Observation:

- Green dot at both ends → successful connection
- Status was successful & type of layer was L2MP



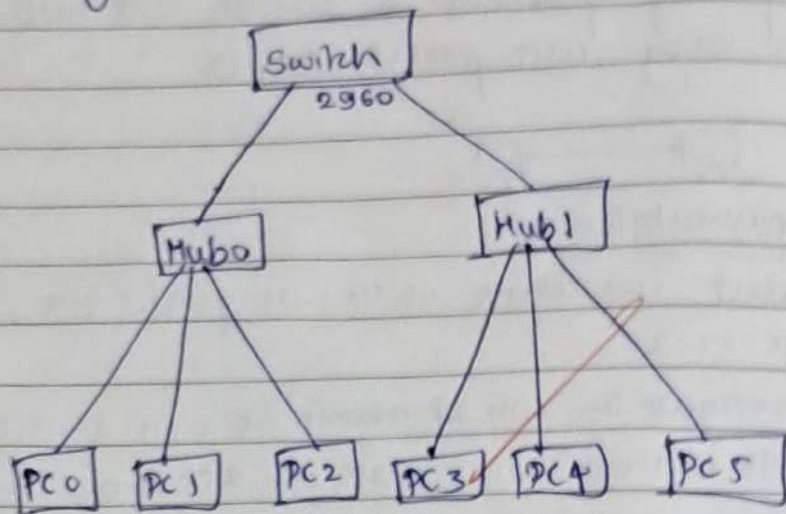
Basic diff b/w two is when received information (hub) it broadcast to all connected devices.

- Hub broadcasts to all ports
- Switch to specific port (MAC address-based)
- operates at layer 1 (Hub) → Physical layer
- operates at layer 2 (Switch) → Data Link layer

→ Outcome: Successfully created network topology for simulating simple PDU (Protocol Data Unit) to analyze communication.

27/9/24

(Ex 2) Aim: Create a topology using multiple hubs & a switch connecting them to simulate a PDU (Packet Distribution)



→ Configure PC0, PC1, PC2, PC3, PC4, PC5 with IP addresses as 10.0.0.1, 10.0.0.2, 10.0.0.3, 10.0.0.4, 10.0.0.5, 10.0.0.6

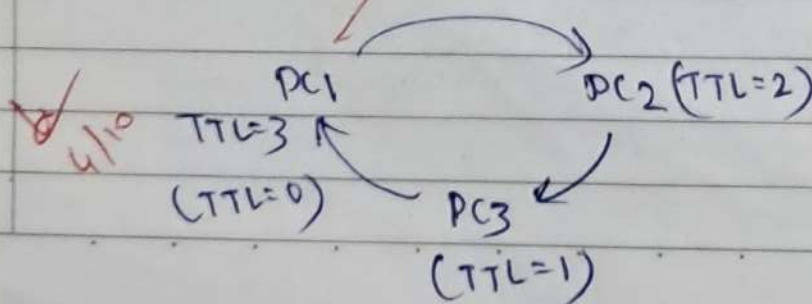
→ Establish the connection between switch with two hubs and then with devices using copper cross over

→ Choose the source and destination to send a simple PDU

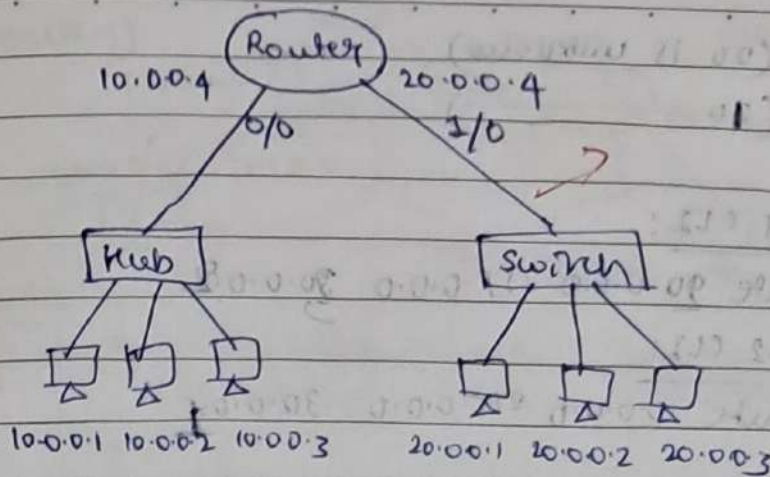
→ In command prompt type ping destination-address
eg ping 10.0.0.3

→ Status becomes successful displaying the source and destination device with ICMP type.

→ TTL (time to live)



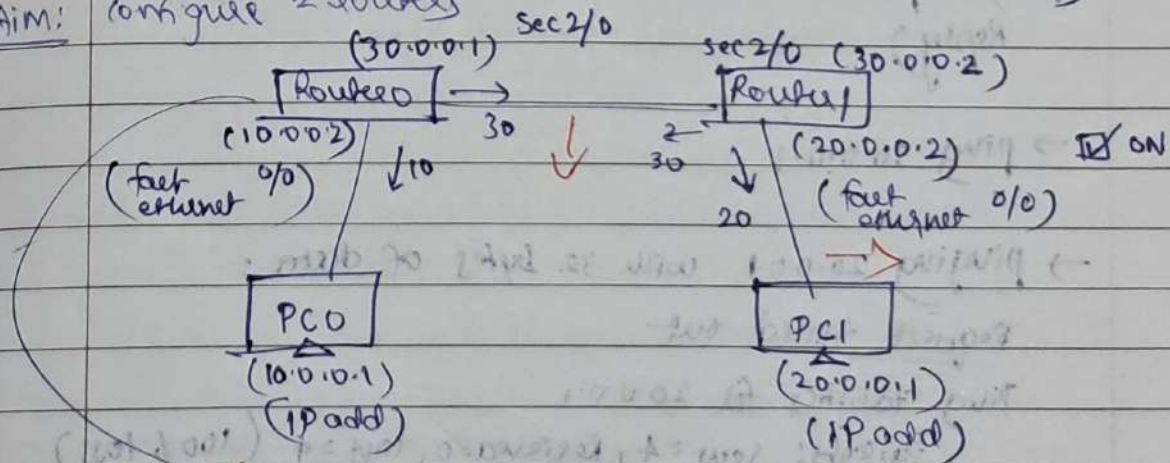
18/10

Date / /
Page 

- Click on Router → CLI → continue with [yes/no]: no configuration
- press Enter
- Router > enable
 - > config t / configure terminal
 - > interface fastEthernet 0/0 (or 1/0)
 - > ip address 10.0.0.4 (or 20.0.0.4) 255.0.0.0
 - > no shutdown
 - > exit

12) → Give the gateways for each PC (e.g. Hub → 10.0.0.4, Switch → 20.0.0.4)

Aim: configure 2 routers



CLI

> enable

> show ip route → 10.0.0.0/8
30.0.0.0/8 } Router0

→ 20.0.0.0/8
30.0.0.0/8 } Router1

→ Router 1 (20 is unknown)
—— 2 (10) ——>

→ in Router 1 CL2:

ip route 20.0.0.0 255.0.0.0 30.0.0.2

in Router 2 CL2:

ip route 10.0.0.0 255.0.0.0 30.0.0.1

→ simulate the packets

in PC0 → Desktop → cmd prompt → ping ip-address of PC1

Pinging 20.0.0.1 with 32 bytes of data
(2nd ip address)

(Request timed out/
Destination
unreachable) ⇒ (configure IP address
and gateway
properly)

(⊕) Configure Router and end devices and replace
ping responses, destination unreachable, Request timed out,
Reply)

→ ping 20.0.0.1

→ pinging 20.0.0.1 with 32 bytes of data:

Request timed out

Ping statistics for 20.0.0.1

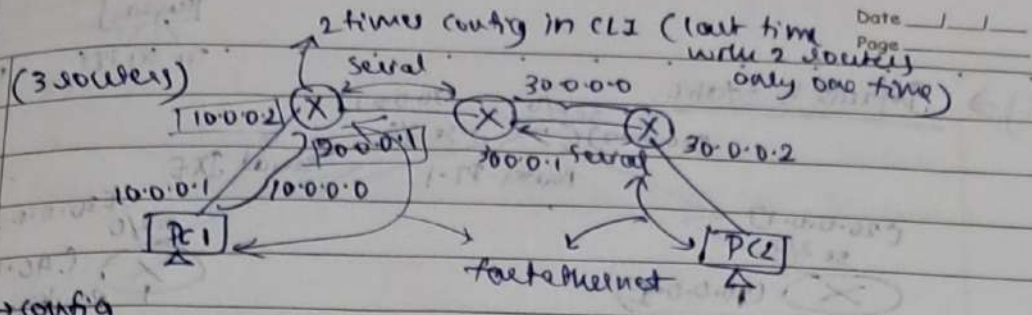
Packets: sent=4, received=0, lost=4 (100% loss)

→ Reply from 20.0.0.1, bytes=32 time=1ms TTL=126

Sent=4, Received=4, lost=0 (0% loss)

Approx round trip in ms:

min=1ms, max=8ms, Avg=5ms



→ config

IP Source	N/w address & unknown	Subnet mask	Interface Router
-----------	-----------------------	-------------	------------------

- Simulate tracer from PC1 to PC2
- ping 20.0.0.1
- o/p: Reply from 10.0.0.2: Destination host unreachable
- (Router unable to find another n/w)

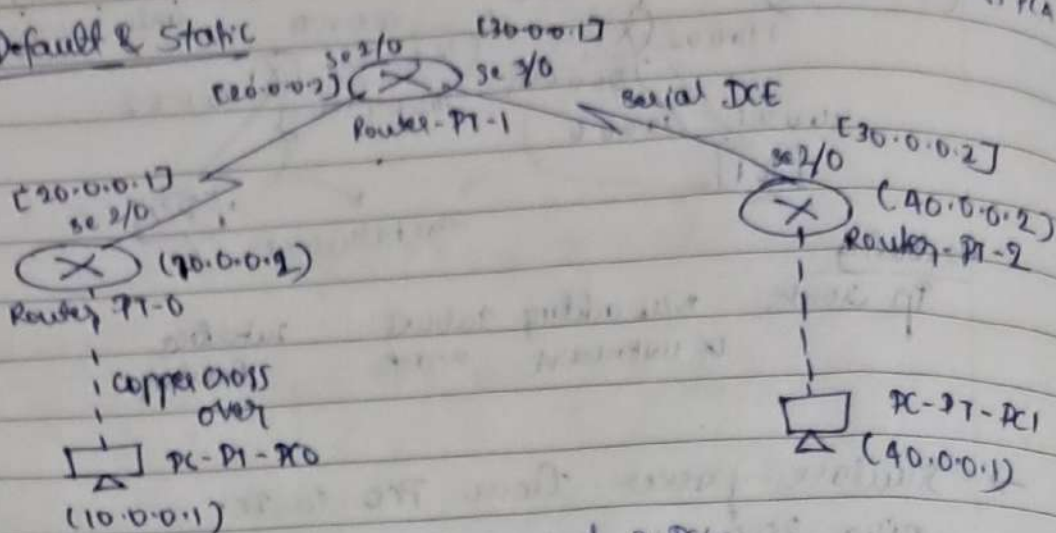
18/10

Router > Show ip route

Outcome: Configured IP address on routers, and explored various ping messages including successful, replies, request timed out, destination unreachable error.

(3) →

Default & Static



Aim: Configuring 3 routers and 2 PCs

Router-R0 → fast ethernet Fa 0/0 10.0.0.2

Router-R2 → fast ethernet Fa 0/0 40.0.0.2

PC0 (gateway) → 10.0.0.2

PC1 (gateway) → 40.0.0.2

> ping 40.0.0.1

pinging 40.0.0.1 with 32 bytes of data
Request timed out

Revised: 0, lost = 4 (100% loss)

> ping 40.0.0.1

Reply from 10.0.0.2: Destination host unreachable

6/8 7/8 ~~8/8~~ Router 0: (S* 0.0.0.0/0 [1/0] via 20.0.0.2)

show ip route

10.0.0.0/8 is directly connected, fast ethernet 0/0

20.0.0.0/8 is directly connected, serial 2/0

Router 1: &

20.0.0.0/8 30.0.0.0/8

(serial 2/0)

(serial 3/0)

Router 2: (S* 0.0.0.0/0 [1/0] via 30.0.0.2)

show ip route (40.0.0.0/8 is directly connected 0/0)

(30.0.0.0/8 is directly connected serial 2/0)

Router 0:

Router(config)# ip route 0.0.0.0 0.0.0.0 20.0.0.2
Router(config)# ip route 0.0.0.0 0.0.0.0 30.0.0.1

Router 1:

S 10.0.0.0/8 [1/0] ~~via~~ 20.0.0.1

S 40.0.0.0/8 [1/0] via 30.0.0.2

A

PC>ping 10.0.0.1

pinging 10.0.0.1 with 32 bytes of data

Reply from 40.0.0.1: bytes=32 time=15 ms TTL=155

ping statistics for 40.0.0.1:

packets: sent=4, received=4, lost=0 (0% lost)

Approximate round trip times in milliseconds:

minimum=2ms, maximum=15ms, average=8ms

8/11

1) Req. timed out

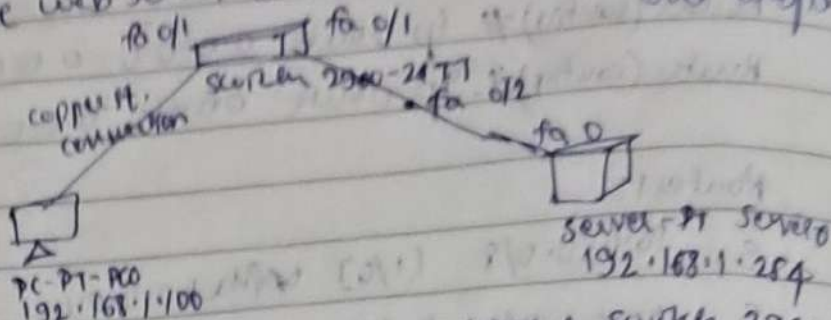
2) Destination un

Outcome: Successfully configured default and static routes on routers, enabling proper routing of packets between different network segments.

8/11/24
(4) = Q

Demonstrate web server & DNS

Aim: Testing connectivity and webpage



- create LAN using generic PC, generic server & switch 2960-24TT & configure ip address for end devices using copper straight through PC - 192.168.1.100 - fast ethernet 0/1 2960-24TT switch - 192.168.1.254 with subnet mask 255.255.255.0

- In server → global settings → gateway/pri/s → static
↳ config → fastEthernet → port status → ON
↳ static ip address → 192.168.1.254

PC → config → fastEthernet → static ip → 192.168.1.100

simulate packet from pc to server

pc> ping 192.168.1.254
pinging 192.168.1.254 with 32 bytes of data
Reply from 192.168.1.254 : bytes = 32 time = 123 ms TTL = 128

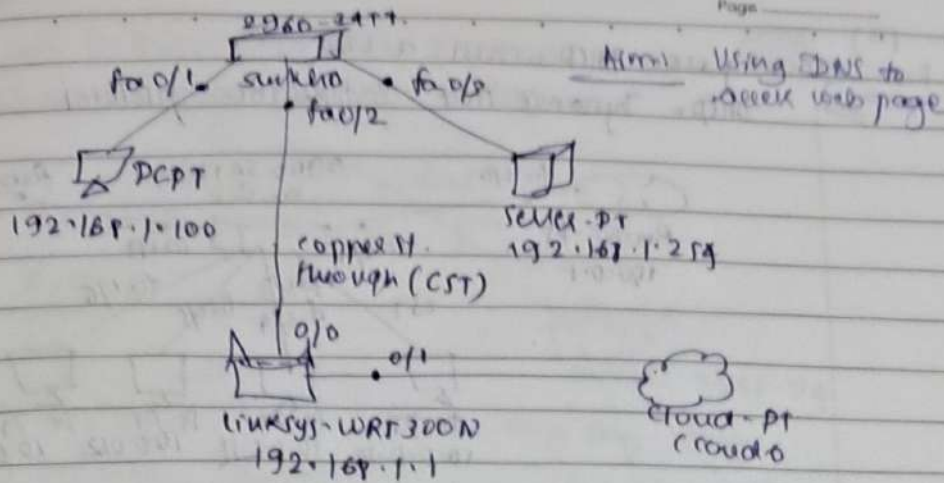
ping statistics for 192.168.1.254

packets: sent = 4, received = 4, lost = 0 (0% loss)

approx round trip times in ms:

min = 4 ms, Max = 178 ms, Avg = 48 ms

server → config → services → HTTP → HTML codes
PC → desktop → web browser → URL: 192.168.1.254
⇒ webpage appears



→ create a LAN using generic PC, server, 2960-24TT switch, WRT300N wireless device & cloud using copper st. through config PC → 192.168.1.100 - fastEthernet 0/1

Server → 192.168.1.254 - fastEthernet 0/3

WRT300N → 192.168.1.1 - fastEthernet 0/2

→ switch to WRT300N → ethernet 1
WRT300N to cloud → ethernet 2

→ server → ~~config~~ gateway → 192.168.1.1

↳ web server → http → circo packet tracer - html code

PC → gateway → 192.168.1.1

→ server → dns → dns service → ON

↳ Resource Records → Name - superyahoo.com

→ address - 192.168.1.254

→ add

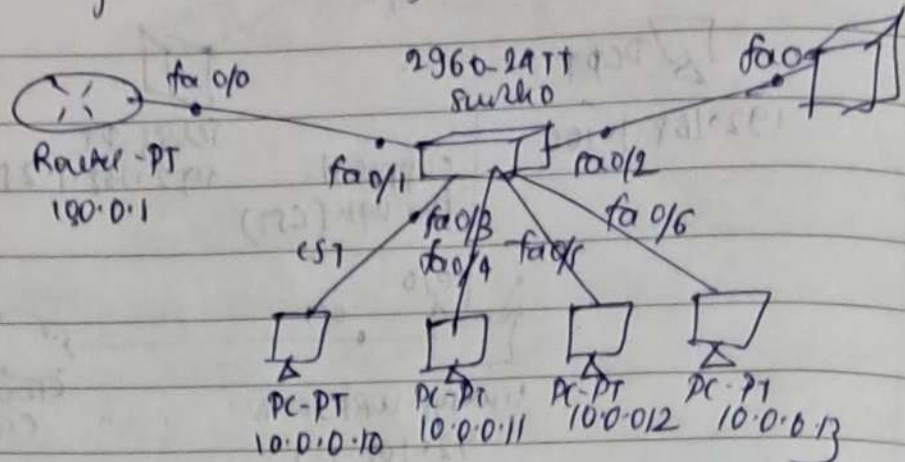
PC → Desktop → IP configuration → DNS server - 192.168.1.254

↳ web browser - URL: superyahoo.com

↳ circo packet tracer shows web page - displays

Outcome: Configured web server and DNS within LAN, allowing user to access website and resolve domain name to IP address

(3) Configure DHCP within a LAN.
DHCP - Dynamic Host Configuration Protocol



→ Create LAN → Router → 2960 24TT switch, generic, PC-PT 4 addresses

→ Configure router:

Router>> enable

Router# config t

Router(config)# interface fastEthernet 0/0

Router(config-if)# ip address 10.0.0.1 255.0.0.0

Router(config-if)# no shutdown

Router(config-if)# exit

→ server → gateway - 10.0.0.1

ip address → 10.0.0.2

services → DHCP → default - gateway - 10.0.0.1

on → DNS server - 10.0.0.2

→ start ip address - 10.0.0.10

→ subnet mask - 255.0.0.0

→ max no. of user - 500

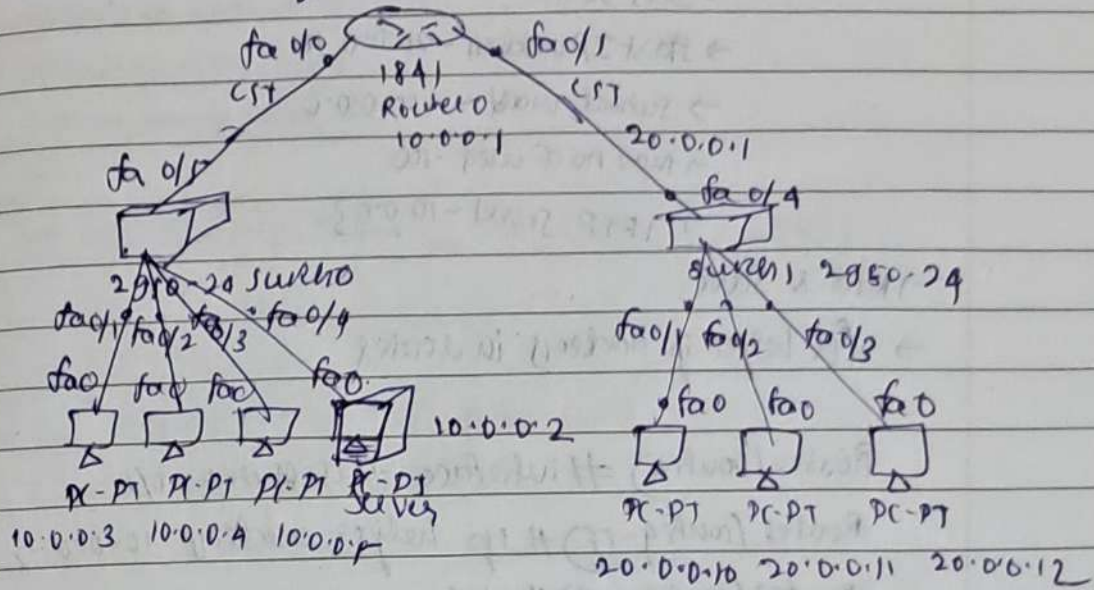
→ TFTP server - 10.0.0.2

→ save

→ PC → IP configuration → DHCP

↳ IP will be configured automatically

Aim: To get IP from DHCP that is present in some other network using IP Helper Address



Router configuration for fa0/0 and fa0/1

Router>enable

Router# config terminal

Router(config)# interface fa0/0

Router(config-if)# ip address 10.0.0.1 255.0.0.0

Router(config-if)# no shutdown

Router(config-if)# exit

Similarly, Router → fa0/1 → 20.0.0.1 255.0.0.0

Server → config → gateway →

↳ fa0/0 → ip address → 10.0.0.2

↳ DHCP-server on

PC → IP configuration → DHCP → IP address is automatically assigned to all the PC

To get IP address from DHCP in other network without server

Outcome: Configured DHCP both inside and outside the LAN, allowing automatic IP address assignment to devices.

server → config → DHCP → Default gateway - 20.0.0.10

↳ Pool Name - network

→ DNS server - 10.0.0.2

→ Start IP address - 20.0.0.10

→ Subnet mask - 255.0.0.0

→ Max no. of users - 100

→ TFTP server - 10.0.0.3

→ Add & Save

→ For helper ip address in router

Router(config)# interface fastethernet 0/1

Router(config-if)# ip helper-address 10.0.0.2

Router(config-if)# exit

PC → IP configuration - DHCP → ip address is automatically assigned by DHCP server.

— x — x — x — x —

15/11

29

Write a program for Error Detection using CRC-CCITT (16 bits)

Code:

```
#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 msg with 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 error, 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[100], op[100], poly[50];
```

```
    char poly[] = "1000100000100001";
```

```
    cout << "Enter input message in binary" << endl;
```

```
    cin >> ip;
```

```
    crc(ip, op, poly, 1);
```

```
    cout << "The transmitted message is: " << ip << op + strlen(ip)
```

```
    << endl;
```



```
cout << "Enter received message in binary" << endl;
cin >> decv;
```

```
if (uc(decv, op, poly, 0)) cout << "No error in data" << endl;
else cout << "Error in data transmission has occurred" << endl;
return 0;
```

Output:

→ Enter the input message in binary
1010

The transmitted message is: 10101010000101001010

Enter the received message in binary: 1010101010000101001010
No error in data

→ Enter the input message in binary
1010

The transmitted message is: 10101010000101001010

Enter the received message in binary: 10101010000101001011
Error in data transmission has occurred.

Leaky Bucket Algorithm

```
#include <iostream>
```

```
#include <bits/stdc++.h>
```

```
using namespace std;
```

```
int main()
```

```
{
    int no-of-queries, storage, output-pkt-size;
```

```
    int input-pkt-size, bucket-size, size-left;
```

```
    storage = 0;
```

```
    no-of-queries = 4;
```

```
    bucket-size = 10;
```

```
    input-pkt-size = 4;
```

```
    output-pkt-size = 4;
```

```

while (i=0; i < no. of queries; i++) {
    size-left = bucket-size - storage;
    if (input-pkt-size <= size-left) {
        storage += input-pkt-size;
    }
    else {
        printf("Packet loss = %d\n", input-pkt-size);
    }
    printf("Buffer size = %d out of bucket size = %d\n",
        storage, bucket-size);
    storage -= output-pkt-size;
}
returning;
}

```

Output:

Buffer size = 4 out of bucket size = 10

Buffer size = 7 out of bucket size = 10

Buffer size = 10 out of bucket size = 10

Packet loss = 4

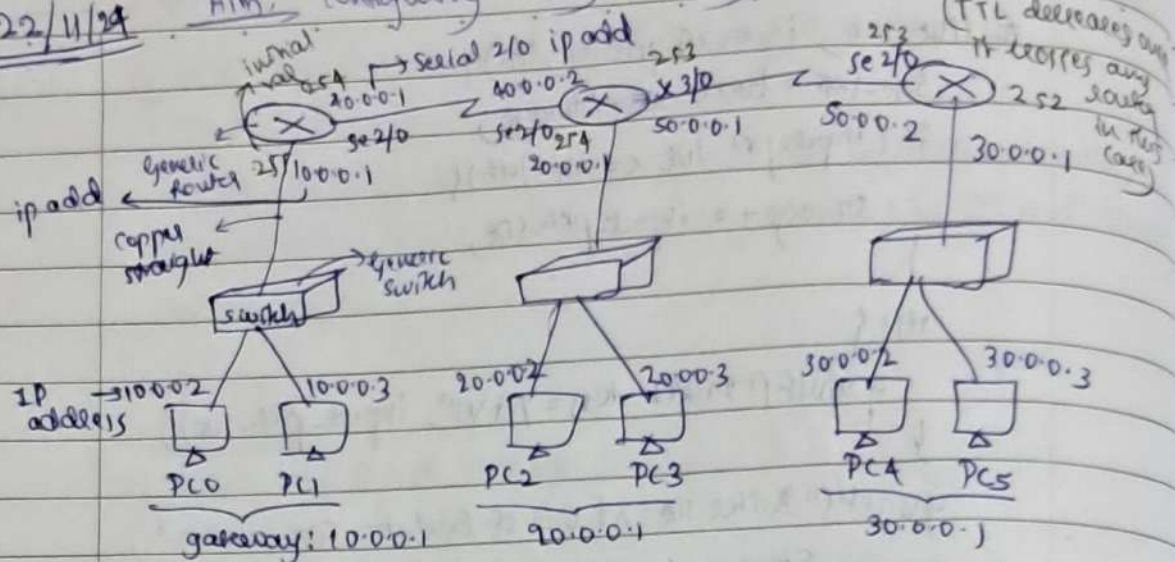
Buffer size = 9 out of bucket size = 10

(CRC) Outcome: Successfully implemented a program to detect transmission errors in data (Cyclic Redundancy Check)

(Leaky bucket) Outcome: Successfully implement program for congestion control, simulating packet flow and self-limiting to avoid network congestion.

22/11/24

Aim: Configuring using Routing Information Protocol



for each router

if in (PC0)

selection

ping 20.0.0.3

→ Destination host unreachable

eg (1) click on Router 0

(2) click on CLI

(3) Type enable

(4) Type config-terminal

(5) Router# router rip

(6) (For R0) #network 10.0.0.0
#network 40.0.0.0
#network

(For R1) 40.0.0.0
20.0.0.0
50.0.0.0

(For R2)

50.0.0.0

30.0.0.0

(7) #exit

(8) Router# show ip route

Now

in (PC0) (desktop → cmd prompt)

ping 20.0.0.3

Request timed out

Reply from 20.0.0.3: bytes=32 time=3ms TTL=126

Ping statistics for 20.0.0.3

Packets: Sent=4, Received=3, Lost=1 (25% loss)

Approximate round trip times in milliseconds:
Minimum = 3ms, Maximum = 4ms, Average = 3ms

PC> ping 20.0.0.3

pinging 20.0.0.3 with 32 bytes of data:

Reply from 20.0.0.3: bytes=32 time=2ms TTL=126
Reply from 20.0.0.3: bytes=32 time=2ms TTL=126
Reply from 20.0.0.3: bytes=32 time=2ms TTL=126
Reply from 20.0.0.3: bytes=32 time=2ms TTL=126
Ping statistics for 20.0.0.3:
Packets: sent=4, received=4, lost=0 (0% loss)

Minimum = 2ms, Maximum = 11ms, Average = 4ms

⇒ on sending packet from lets say PC0 to PC3
→ Inbound PDV Details (Simulation mode ⇒ Auto capture/Play
TTL: 255
(Switch to Router) ⇒ click on event list (any type of protocol ⇒ STP, ICMP) etc)

Outbound
TTL: 254

→ (Router 0 to Router 1)

Inbound: TTL-254

Outbound: TTL-253

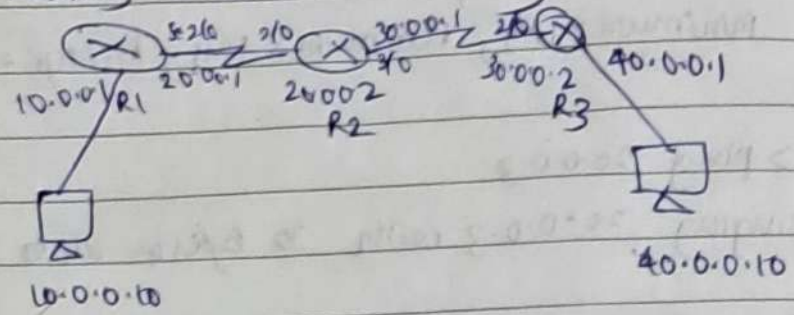
→ (Router 1 to Router 2)

Inbound: TTL-253

Outbound: TTL-252

✓
2.11 Outcome: Configured RIP on routers, allowing automatic routing updates and dynamic routing within the network

29/4 Aim OSPF Routing Protocol & connecting vceag



R1

- configure ip address to all interfaces (fa 0/0 → 10.0.0.1)
- for serial s2/0 ip address 20.0.0.1 255.0.0.0
- encapsulation ppp
- clock rate 64000
- no shutdown
- exit

R2 (fa 0/0 → 30.0.0.1)

- for serial s2/0
- encapsulation ppp
- clock rate 64000
- no shutdown
- exit

R3 (fa 0/0 → 40.0.0.1)

- configure fa 0/0 → 40.0.0.1
- for serial interface:
- config ip 30.0.0.2, encapsulation ppp, no clock rate
- Router R2: configure serial s0/0 → ip 20.0.0.2
- configure serial s2/0 → ip 30.0.0.1,
- encapsulation ppp
- clock rate 64000

Configuring OSPF Routing Protocol:

Router R1

```
R1(config)# router ospf 1
R1(config-router)# router-id 1.1.1.1
R1(config-router)# network 10.0.0.0 0.255.255.255 area 3
R1(config-router)# network 20.0.0.0 0.255.255.255 area 1
```

Router R2

```
(config)# router ospf 1
(config-router)# router-id 2.2.2.2
# network 20.0.0.0 0.255.255.255 area 1
# network 30.0.0.0 0.255.255.255 area 0
```

Router R3

```
(config)# router ospf 1
Router-id 3.3.3.3
network 30.0.0.0 0.255.255.255 area 0
network 40.0.0.0 0.255.255.255 area 2
```

Routing table: R1

Router# show ip route

Gateway of last resort is not set

e 10.0.0.0/8 is directly connected, fast ethernet 0/0
 20.0.0.0/8 is variably connected, 2 subnets, 2 masks
 O 2A 30.0.0.0/8 [110/128] via 20.0.0.2,
 00:01:18, Serial 1/0

Configure loopback address

R1:

```
Router(config)# interface loopback 0
Router(config-if)# ip add 192.168.1.2/24 255.255.0.0
no shutdown
```


RIP \rightarrow Hop Count (Dijkstra)

OSPF \rightarrow Link State

Date / /
Page

R2:

```
Router(config)#interface loopback 0
ip add 172.16.1.253 255.255.0.0
no shutdown
```

R3:

```
interface loopback 0
ip add 172.16.1.254 255.255.0.0
no shutdown
```

R1:

show ip route

o 10.0.0.0/8 is directly connected, FastEthernet 0/0
o 30.0.0.0/8 [910/20] via 20.0.0.2, 00:11:11 Serial 2/0

R2:

Router> show ip route

o RA 10.0.0.0/8 [110/65] via 20.0.0.1, 00:25:25, Serial 2/0
o 20.0.0.0/8 is directly connected, Serial 2/0
o RA 40.0.0.0/8 [110/65] via 30.0.0.2, 01:27:47, Serial 3/0
172.16.0.0/24 is subnetted, 1 subnet

R3:

Router> show ip route

o RA 20.0.0.0/8 [110/21] via 30.0.0.2, 00:48:20, Serial 2/0
o 172.16.1.0 is directly connected, loopback 0
o RA 10.0.0.0/8 [110/128] via 30.0.0.1, 00:00:09, Serial 2/0

R1:

```
Router(config)#router ospf 1
```

```
Router(config-router)#area 1 virtual-link 2.2.2.2
```

R2:

```
Router(config)#router ospf 1
Router(config-router)#area 1 virtual-link 1.1.1.1
```

→ ping from 10.0.0.10 to 40.0.0.10
ping 40.0.0.10

pinging 40.0.0.10 with 32 bytes of data

ping statistics for 40.0.0.10:

Packets: sent 4, Received = 3, lost = 1 (25% loss)

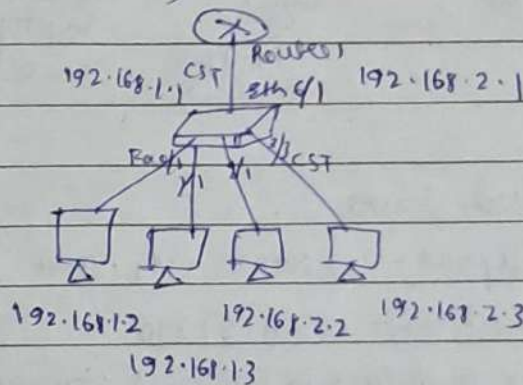
Approximate round trip in ms:

Min = 2ms, Max = 16ms, Avg = 8ms

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

Outcome: Configured your switch path first (OSPF) enabling link-state routing

Aim VLAN and make PCs communicate among VLAN
(choose 1841 router)



→ In switch, go to config tab & VLAN database

→ give VLAN no 2, select interface em 6/1 and make it trunk

→ VLAN trunking allows switches to forward frames over link called trunk

→ CLI

Router(Nbr)# exit

Router#config

Router(config)# interface fastEthernet 0/0.1

Router(config-subif)# encapsulation dot1q 2

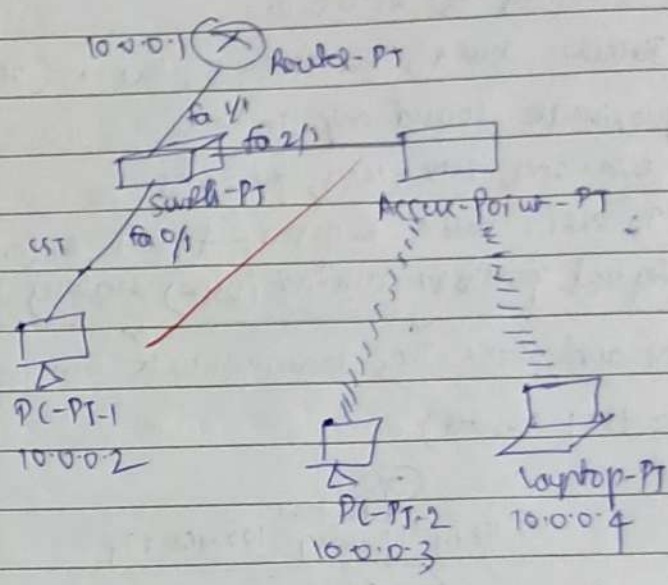
ip address 192.168.2.1 255.255.255.0

no shutdown

end

→ Dot1q (802.1Q) is networking standard that supports Virtual LANs (VLAN) on an IEEE Ethernet network

Q Aim:
WLAN and make nodes communicate wirelessly

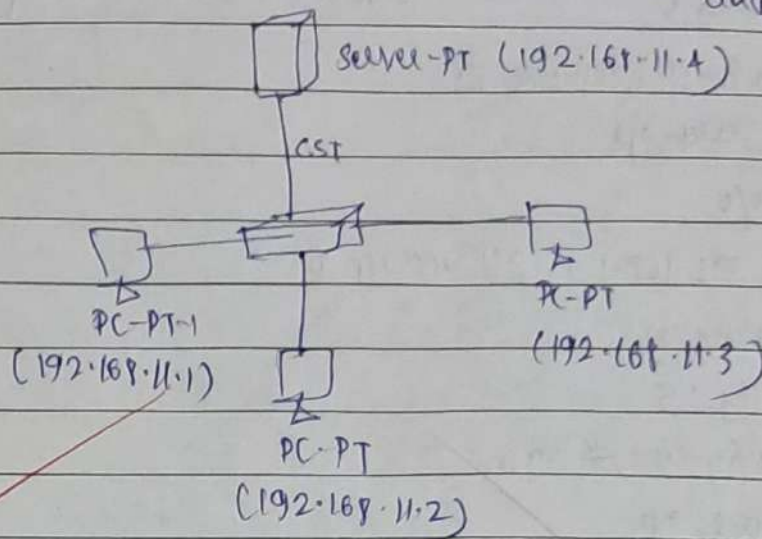


- configure PC1 and router
- configure Access point - ssid name - any name (WLAN here)
- select WEP and key - 1234567890
- configure PC-2 and laptop wireless standards
- switch off device, drag existing PT-HOST-NM-LRM to LMS and drag WMP300N to empty port, switch on
- in config, ssid, WEP, key, ip address and gateway (as normally done) to the device
- ping to every other device and ~~check~~ the result

>ping 10.0.0.3
 pingung 10.0.0.3 with 32 bytes of data
 reply from 10.0.0.3: bytes=32 time=16ms TTL=128
 sent=4, received=4, lost=0 (0% loss)

- before connection: Destination Host Unreachable
- ping 10.0.0.10 : Request timed out
- outcome: Constructed VLAN and communication among PCs isolating traffic from other hosts and built a WLAN enabling wireless communication

① ARP (Address Resolution Protocol) Aim: To construct simple LAN and understand concept of ARP



→ Simulation panel → Inspect → PC0 (Right click)
↳ ARP table (empty)
↳ Server-ARP table (empty)

→ Ping from PC-PT to server

> arp-a

no entries found

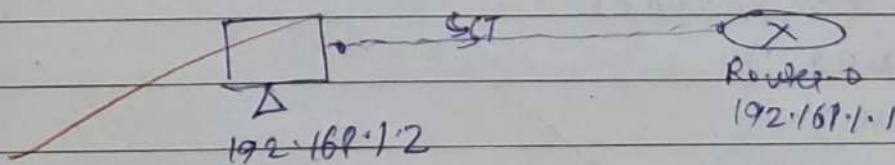
> ping 192.168.11.4

Reply from 192.168.11.4: bytes=32, time=8ms, TTL=128

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

Minimum = 4ms, Maximum = 8ms, Average = 6ms

→ ARP table shows the IP address of PC1 and then PC1 receives the acknowledgement, its ARP table shows IP address of server



Outcome: Built a simple LAN and demonstrated ARP operation, devices map IP addresses to MAC addresses for communication

CLI

Aim: To understand concept and operation of TELNET by accessing router in server room

```

Router> enable
Router# config t
Router(config)# hostname R1
#enable secret 3p
#int Fa 0/0
#ip add 192.168.1.1 255.255.255.0
# no shutdown
# line vty 0 5
Router(config-line)# login
# password tp
# exit
configured from console by console
R1# wr
Building configuration...
[OK]

```

```

> ping 192.168.1.1
Reply from 192.168.1.1 : bytes=32 time=1ms TTL=255
Ping statistics for 192.168.1.1 :
    Packets: Sent=4, Received=4, loss=0 (0% loss)
    Minimum=0ms, Maximum=1ms, Average=0ms

```

```

PC> telnet 192.168.1.1
Trying 192.168.1.1... open
Use ASCII characters for login now
Password: tp
Router> en
Password: tp

```

Outcome: Successfully accessed the router in server room using TELNET from PC demonstrating remote management

Q Socket Programming Aim: Build client/server applications that communicate using sockets

TCP: (Using TCP/IP, write client/server program for server and client sending file name) (Transmission Control Protocol Internet Protocol)

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())

print("from server")

clientSocket.close()

serverTCP.py

from socket import *

serverName = '127.0.0.1'

serverPort = 12000

serverSocket = socket(AF_INET, SOCK_STREAM)

serverSocket.listen(1)

while 1:

 print("Server is ready to receive")

 file = open(sentence, "r")

 data = file.read(1024)

 connectionSocket.send(data.encode())

 file.close()

 connectionSocket.close()

Output:

Server is ready to receive

Sent contents of server TCP.py

Server is ready to receive

Page _____
Aim: VDP (Using VDP socket, write client server program to make client sending file name and server to send back contents of requested file if present) (Use Datagram protocol)

Server VDP.py

SERVERPORT = 12000

SERVERSOCKET = socket(AF_INET, SOCK_DGRAM)

sentence = input("Enter file name: ")

fileContents, serverAddress = SERVERSOCKET.recvfrom(2048)

print(fileContents.decode("UTF-8"))

fileContents.close()

SERVERSOCKET.close()

Server VDP.py

from socket import *

SERVERPORT = 12000

SERVERSOCKET = socket(AF_INET, SOCK_DGRAM)

while 1:

sentence, clientAddress = SERVERSOCKET.recvfrom(2048)

sentence = sentence.decode("UTF-8")

file = open(sentence, "r")

con = file.read(2048)

SERVERSOCKET.sendto(bytes(con, "UTF-8"), clientAddress)

print("Sent contents of", end=" ")

print(sentence)

file.close()

Output: Server is ready to receive

Sent contents of server VDP.py

Server is ready to receive

Outcome:

(TCP): Implemented TCP client-server program where client requests file by name and server sends back the file contents if it exists demonstrating reliable file transfer.

(UDP): Implemented UDP client-server program where client sends file name and server sends file content showcasing connectionless communication

~~26/12~~