VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"JnanaSangama", Belgaum -590014, Karnataka.

LAB RECORD

COMPUTER NETWORKS

Submitted by

Avani.A(1BM22CS059)

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

Academic Year 2024-25

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 "**COMPUTER NETWORKS**" carried out by **AVANI.A(1BM22CS059)** , 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 2024-25.
The Lab report has been approved as it satisfies the academic requirements in respect of **Computer Networks Lab - (23CSSPCCON)** work prescribed for the said degree.

Dr. Nandini Vineeth Associate Professor Department of CSE, BMSCE	Dr. Kavitha Sooda Professor & HOD Department of CSE, BMSCE

<u>Index</u>
CYCLE-1

CYCLE-2

Github Link:
https://github.com/Avani-A/CN_LAB_CS059.git

EXPERIMENT-1

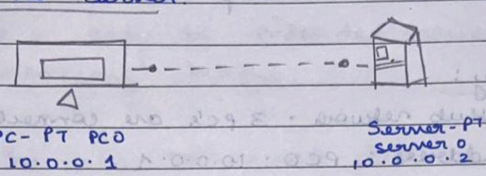
Aim: Create a topology and simulate sending a simple PDU from source to destination using hub and switch as connecting devices and demonstrate ping messages.

Topology , Procedure and Observation:

25/1 Page:

Experiment - 1

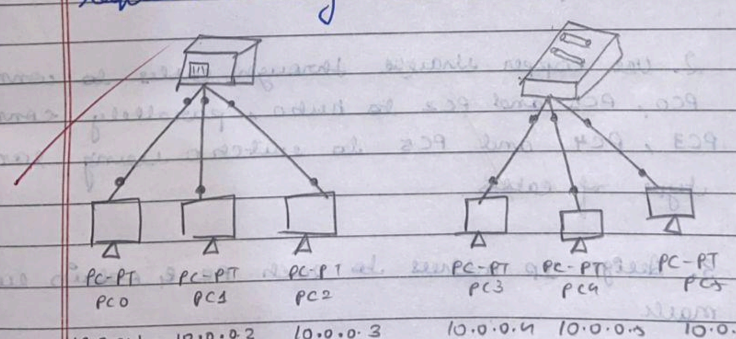
1 PC to server



Aim : To set up a point-to-point network b/w a PC & a server, facilitating direct communica-tion to observe data exchange.

Topology : A PC0 is connected to server using a crossover ethernet cable. IP address PC0 : 10.0.0.1 IP address of server 0 : 10.0.0.2

Observation : The direct connection allows PC0 to communicate w server 0, which is typical in small networks for tasks such as file sharing, servi requests / seeking server responses to client query



Q1) Create a topology and simulate sending a simple PDU from source to destination using hub and switch as connecting devices and demonstrate ping message

Aim : To create a simple network consisting of 3 PC's connected to a central Hub and another network with 3 PC's connected to a switch. This configuration will help observe.

Topology :

1. Hub network : 3 PC's are connected to switch

IP - address : PC0 : 10.0.0.1

PC1 : 10.0.0.2

PC2 : 10.0.0.3

2. Switch network : 3 PCs (PC3, PC4, PC5) are connected to a (switch 0) using straight - through ethernet cables

IP - address : PC3 = 10.0.0.4

PC4 = 10.0.0.5

PC5 = 10.0.0.6

Procedure :

1. Add 1 hub, 1 switch and 6PCs (PC0, PC1, PC2 for the hub); PC3, PC4, PC5 for the switch) to cisco packet tracer.

2. Use copper straight through cables to connect PC0, PC1 and PC2 to hub0, parallely connect PC3, PC4 and PC5 to switch 0 using same type of cables.

3. Assign IP address to each PC & obtain subnet mask

4. switch to simulation to observe data traffic behaviour when packets are sent b/w devices

Bafna Gold
Date: Page:

5. In the hub network, notice how the hub broadcasts packets to all devices, causing potential traffic overload. In the switch network, observe the switch forward packets only to intended recipient, reducing unnecessary traffic.

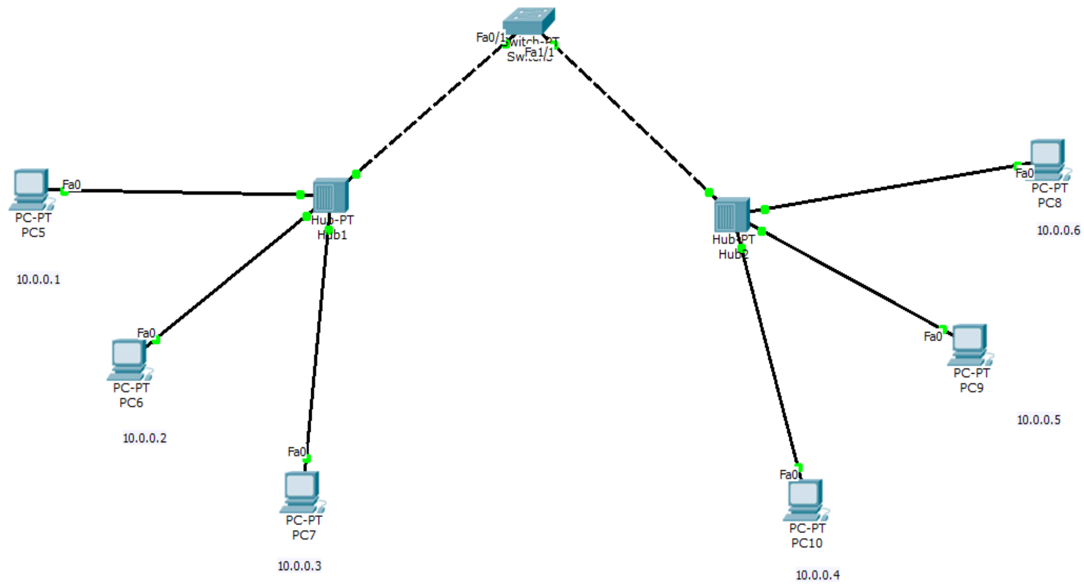
6. The hub broadcasts data to all connected devices leading to more network congestion while the switch efficiently sends data only to the correct device, optimizing performance.

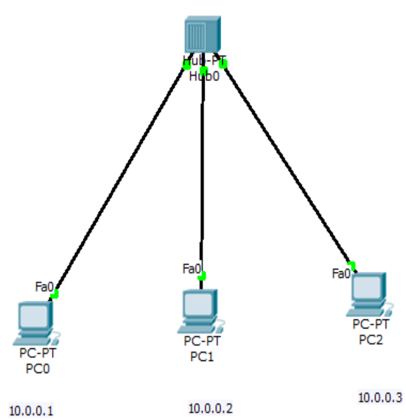
Observation:

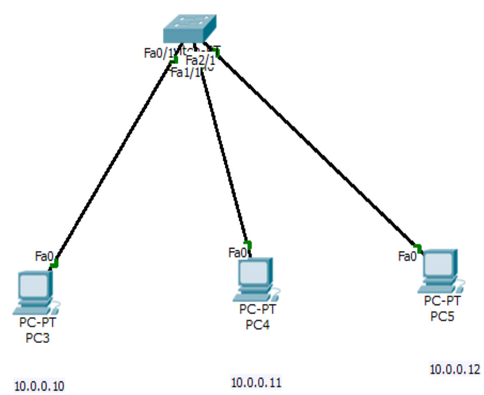
1. The hub broadcasts packets to all devices which may cause unnecessary traffic.
2. The switch forwards packets only to the appropriate device by learning new address, making it more efficient in reducing traffic.

3/1/25

Screen Shots:



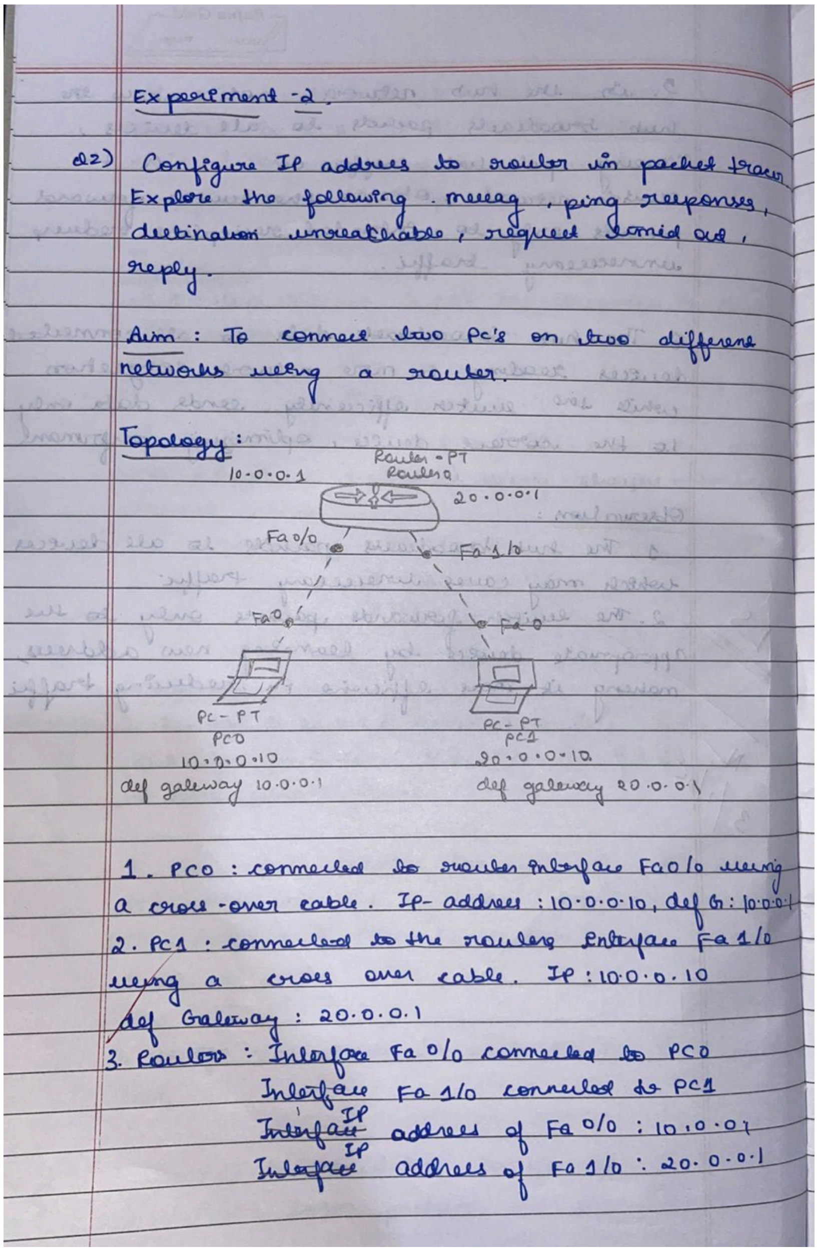




EXPERIMENT- 2A

Aim: Configure IP address to routers in packet tracer. Explore the following messages: ping responses, destination unreachable, request timed out, reply.

Topology , Procedure and Observation:



Experiment -2.

Q2) Configure IP address to router in packet tracer
Explore the following messag, ping responses, destination unreachable, request timed out, reply.

Aim : To connect two PC's on two different networks using a router.

Topology :

1. PC0 : connected to router interface Fa0/0 using a cross-over cable. IP-address : 10.0.0.10, def G : 10.0.0.1
2. PC1 : connected to the routers interface Fa1/0 using a cross over cable. IP : 10.0.0.10 def Gateway : 20.0.0.1
3. Router : Interface Fa0/0 connected to PC0
Interface Fa1/0 connected to PC1
~~Interface~~ IP address of Fa0/0 : 10.0.0.1
~~Interface~~ IP address of Fa1/0 : 20.0.0.1

Bafna Gold
Date: Page:

Procedure:

Open cisco Pkt Tracer and drag the following components onto the workspace

1) Router : place one router
2) PCs : place two PC's

2> use (straights) copper - straight - through to connect device
PC0 - Routers Fa0/0 interface
PC1 - Routers Fa1/0 interface

3> config the router:

Router > enable
Router # config t
Router (config) # ip address 10.0.0.1 255.0.0.0
Router (config-if) # no shutdown
Router (config) # interface fastethernet 1/0
Router (config-if) # ip address 20.0.0.1 255.0.0.0
Router (config) # no shutdown

4> config the PCs:
PC0 : Ip address : 10.0.0.1
Subnet mask : 255.0.0.0
def gateway : 10.0.0.1

PC1 : Ip address : 20.0.0.1
Subnet mask : 255.0.0.0
def gateway : 20.0.0.1

5> Test connectivity :
open command prompt on PC0 to PC1
• use ping command command
>> ping 10.0.0.10.

Observation :
1. If the configurations and cabling are correct, you will recieve successful ping replies between the two PC's

in router click and go to CLI.

Router > show ip route
codes : C - connected . . . . .

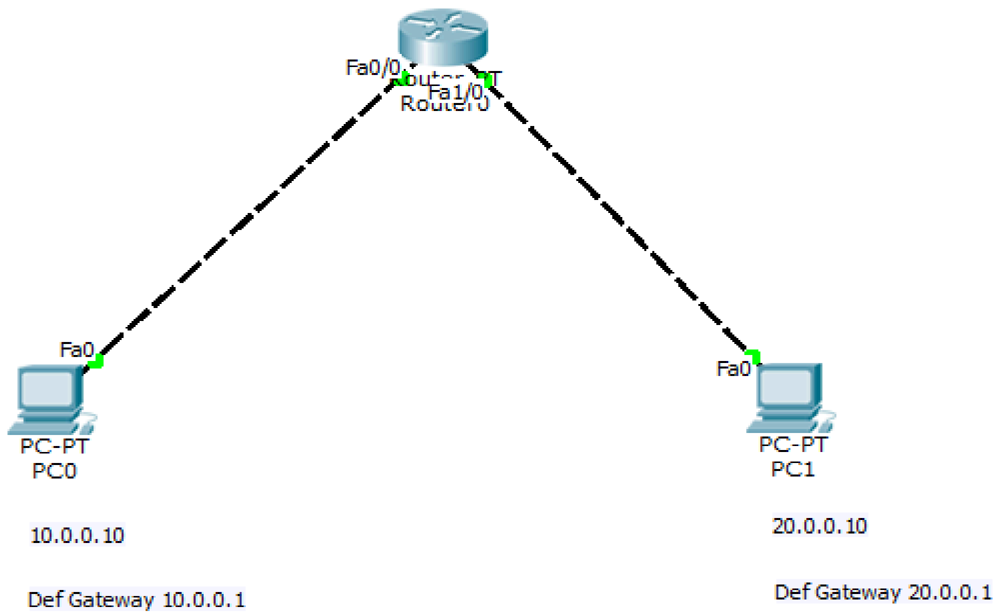
gateway of last resort is not set
C 10.0.0.0/8 is directly connected, fast ethernet 0/0
C 20.0.0.0/8 is directly connected, fast ethernet 1/0

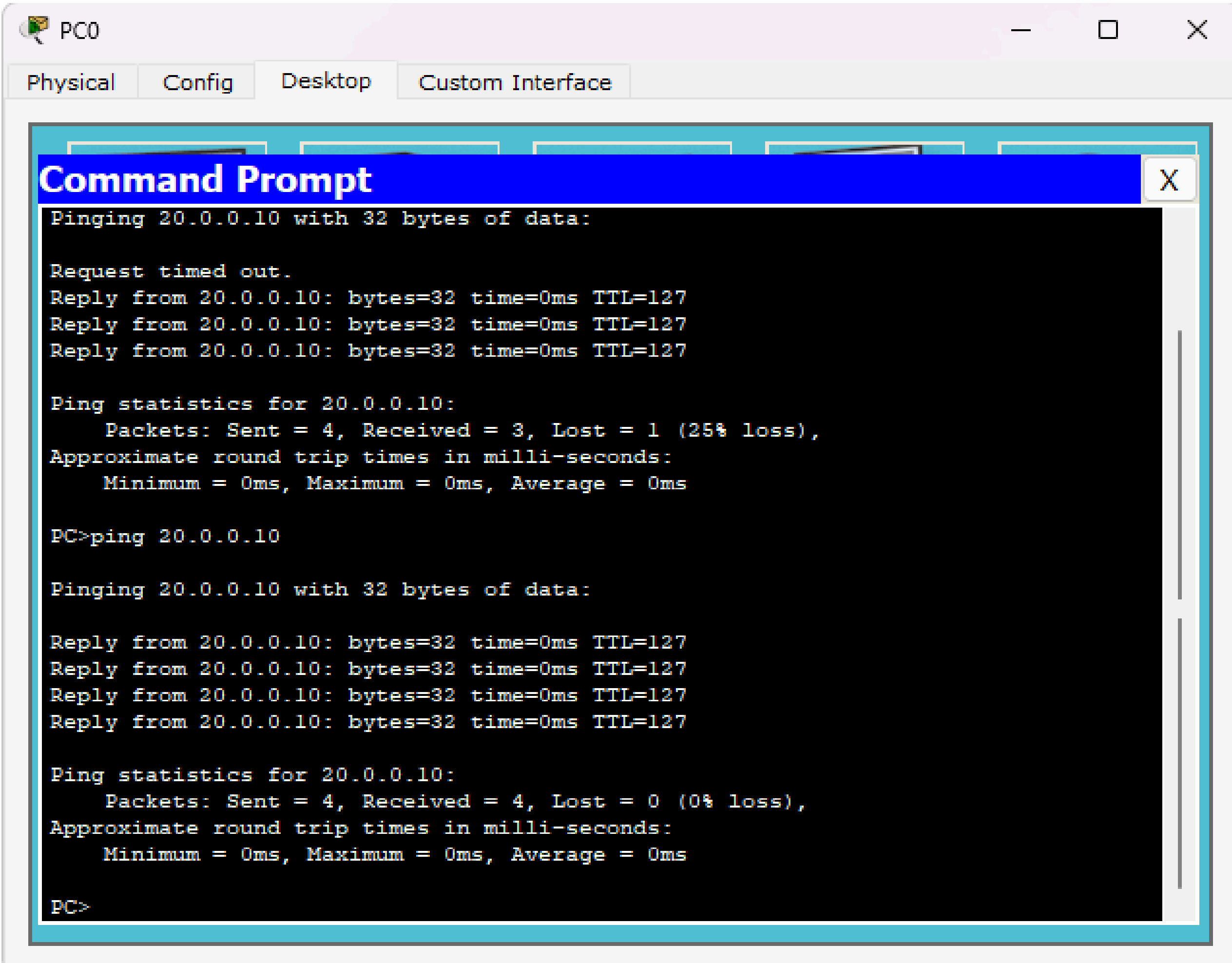
2. pc> PING 10.0.0.10
Pinging 10.0.0.10 with 32 bytes of data
Reply from 10.0.0.10 bytes = 32 time lal TTL = 128
Reply from 10.0.0.10 bytes = 32 time lal TTL = 128
:
:

Ping statistics for 10.0.0.10 :
pkts : sent = 4, Recieved = 4, loss = 0 (0% loss).

3/1/25

Screen Shots:

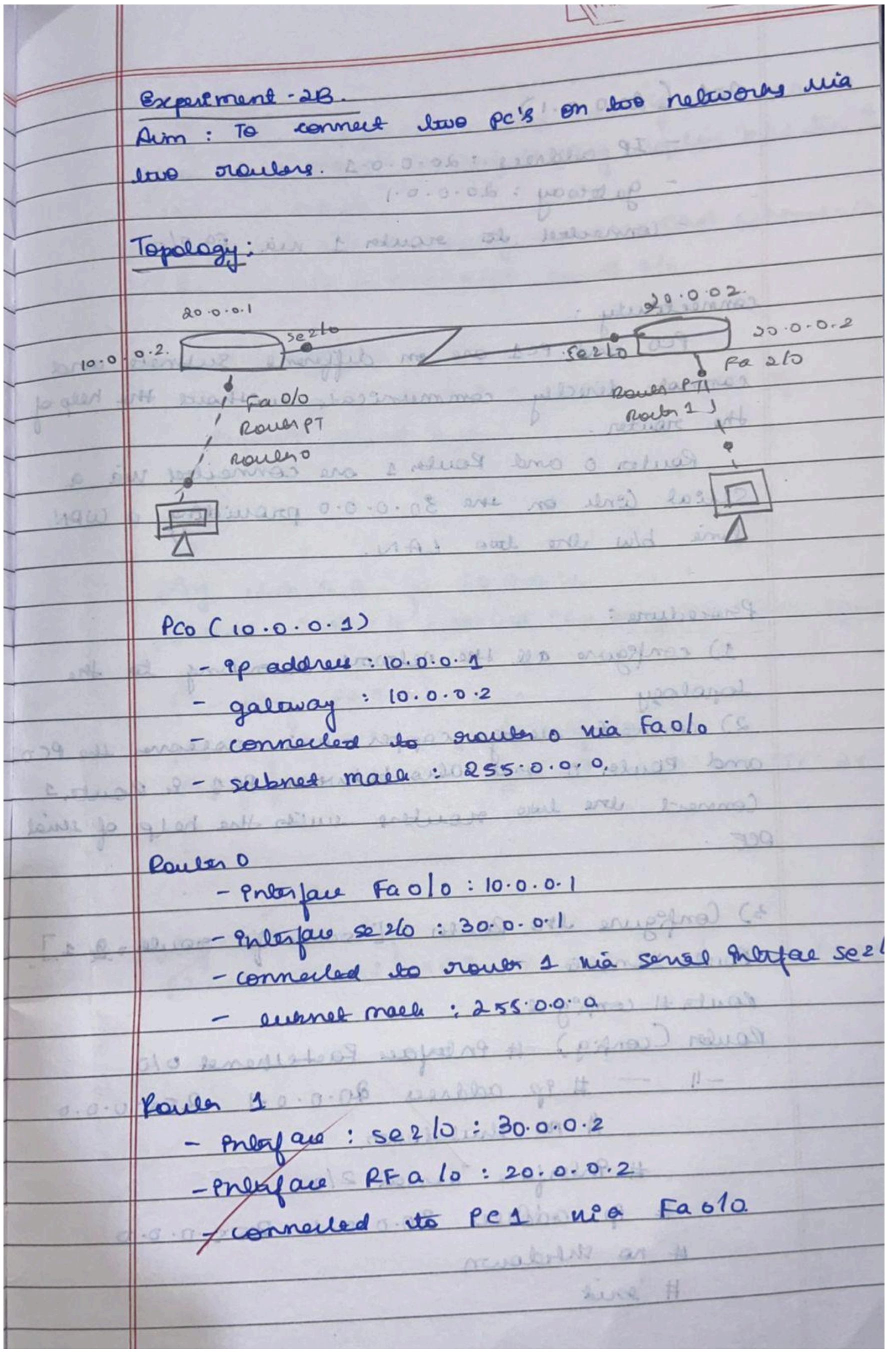



PC0
Physical
Config
Desktop
Custom Interface
Command Prompt
X
Pinging 20.0.0.10 with 32 bytes of data:
Request timed out.
Reply from 20.0.0.10: bytes=32 time=0ms TTL=127
Reply from 20.0.0.10: bytes=32 time=0ms TTL=127
Reply from 20.0.0.10: bytes=32 time=0ms TTL=127
Ping statistics for 20.0.0.10:
Packets: Sent = 4, Received = 3, Lost = 1 (25% loss),
Approximate round trip times in milli-seconds:
Minimum = 0ms, Maximum = 0ms, 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=127
Reply from 20.0.0.10: bytes=32 time=0ms TTL=127
Reply from 20.0.0.10: bytes=32 time=0ms TTL=127
Reply from 20.0.0.10: bytes=32 time=0ms TTL=127
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 = 0ms, Average = 0ms
PC>

EXPERIMENT-2B

Aim: To connect two PC's on two networks via two routers

Topology , Procedure and Observation:



Experiment - 2B.

Aim : To connect two pc's on two networks via two routers.

Topology:

PC0 (10.0.0.1)

- ip address : 10.0.0.1
- gateway : 10.0.0.2
- connected to router 0 via Fa0/0
- subnet mask : 255.0.0.0.

Router 0

- interface Fa0/0 : 10.0.0.1
- interface se2/0 : 30.0.0.1
- connected to router 1 via serial interface se2
- subnet mask : 255.0.0.0

Router 1

- interface : se2/0 : 30.0.0.2
- interface RFa/0 : 20.0.0.2
- connected to PC1 via Fa0/0

PC1 (20.0.0.1)

- IP address : 20.0.0.1
- gateway : 20.0.0.1
- connected to router 1 via Fa 0/0

connectivity :

PC0 and PC1 are on differene subnets and cannot directly communicate without the help of the router.

Router 0 and Router 1 are connected via a serial link on the 30.0.0.0 providing a WAN link b/w the two LAN.

Procedure :

1) configure all the network according to the topology.

2) Connect using copper-wire crossover the PC and Router 0 and also between PC1 & Router 1. Connect the two routers with the help of serial DCE.

3) Configure the Router : [same for router 0 & 1]

Router >enable
Router # config t
Router (config) # interface Fastethernet 0/0
-"- # ip address 20.0.0.1 255.0.0.0
no shutdown
interface serial 2/0
ip address 30.0.0.1 255.0.0.0
no shutdown
exit

Bafna Gold
Date: Page:

Observation :

If the configuration and cabling are carried you will receive successful ping replies b/w two PC

show ip route :

C 10.0.0.0/8 is directly connected Fast ethernet 0/0

C 30.0.0.0/8 is connected serial 2/0

Output : ping 20.0.0.1

pinging 20.0.0.1 with 32 bytes of

Request timed out

.
.
.

ping statistics for 20.0.0.1:

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

PC > Ping 20.0.0.1

pinging 20.0.0.1 bytes (32) of data :

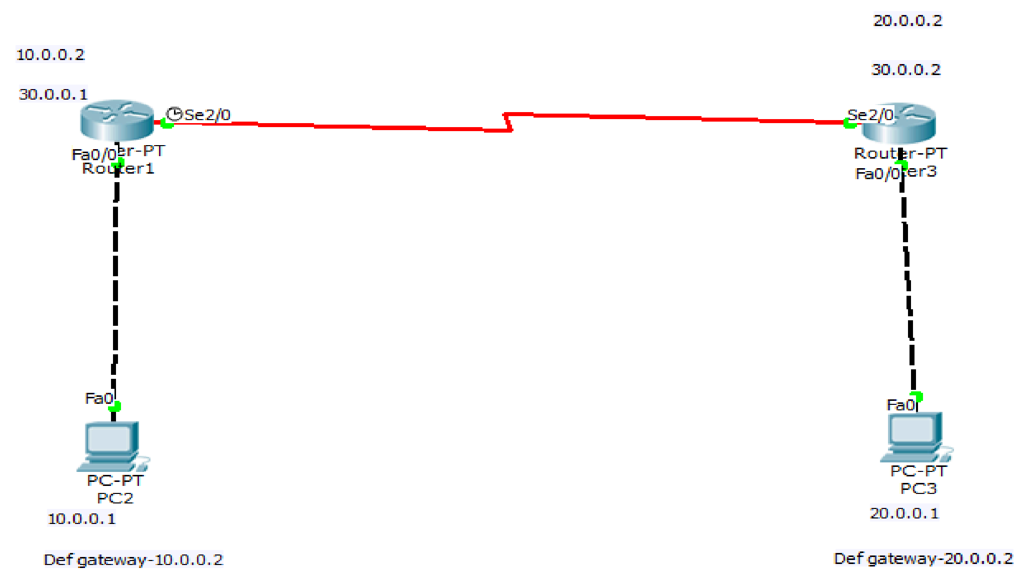
Reply from 20.0.01 : bytes = 32 time : 2ms TTL : 2r

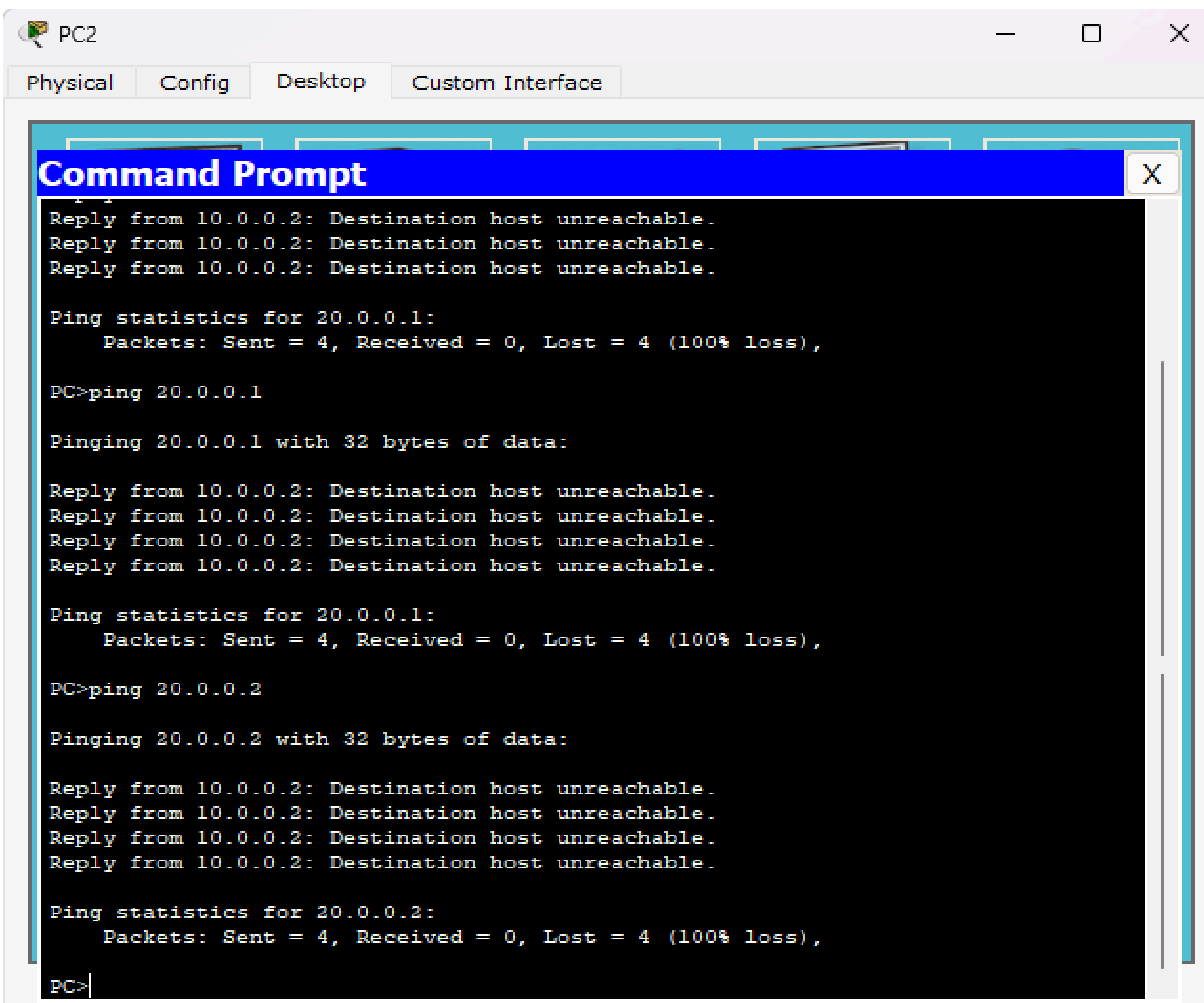
.
.
.

ping statistic for 20.0.0.1:

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

Screen Shots:





EXPERIMENT-3

Aim: Configure default route, static route to the Router.

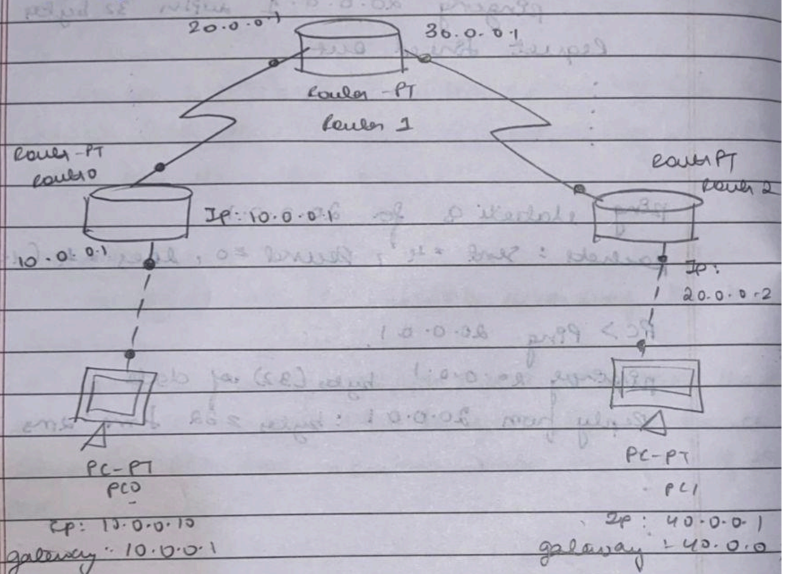
Topology , Procedure and Observation:

Experiment 3.

Q) Static and default routing:

Aim: demonstrate static and default routing using three routers.

Topology:



★ Two PCs:

PC:0 → IP: 10.0.0.10, gateway: 10.0.0.1

PC 1 : IP: 40.0.0.10 / gateway: 40.0.0.1

★ Three Router:

· Router 0: Interface Fa0/0 : 10.0.0.1

Interface Se 2/0 : 20.0.0.1

connected to PC0 and Router 1

* Router 1 : connects router 0 and router 2
  - interface Se 2/0 : 20.0.0.2
  - interface Se 3/0 : 30.0.0.1

* Router 2 : connected to PC1 and Router 1
- interface Se 3/0 : 30.0.0.2
- interface Fa 0/0 : 40.0.0.1

Procedure :

1> configure the PC0 & PC1
2> Configure Router interface.

Router > enable
Router # config t.
Router (config): interface fastethernet 0/0
Router (config) : ip address 10.0.0.1 255.0.0.0
Router (config) no shut
Router > exit

Router # config t
Router # interface serial Se 2/0
Router (config): ip address 20.0.0.1 255.0.0.0
Router (config): (ip address) no shut
Router > exit

Same configuration for
Router 2 & Router 1.

P.T.O.

Router 0 > show ip route.

C 10.0.0.0/8 is directly connected to fastethernet 0/0

C 20.0.0.0/8 is directly connected to Serial 2/0

S* 0.0.0.0/0 [1/0] via 20.0.0.0.

Router 2 > show ip route

C 30.0.0.0/8 is directly connected serial 3/0
C 40.0.0.0/8 is directly connected fastethernet 0/0
S* 0.0.0.0/0 [1/0] via 30.0.0.2

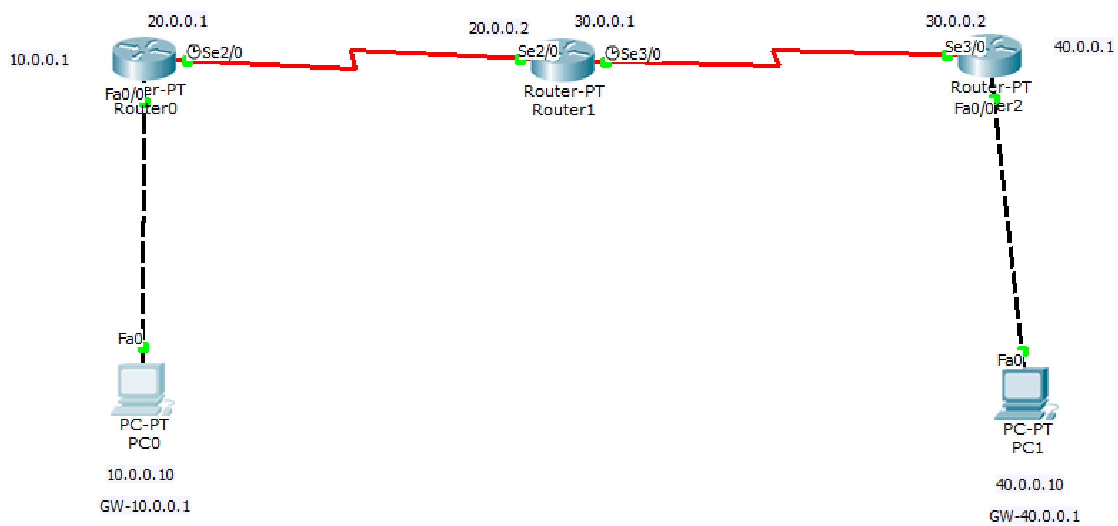
Router 1 > show ip route

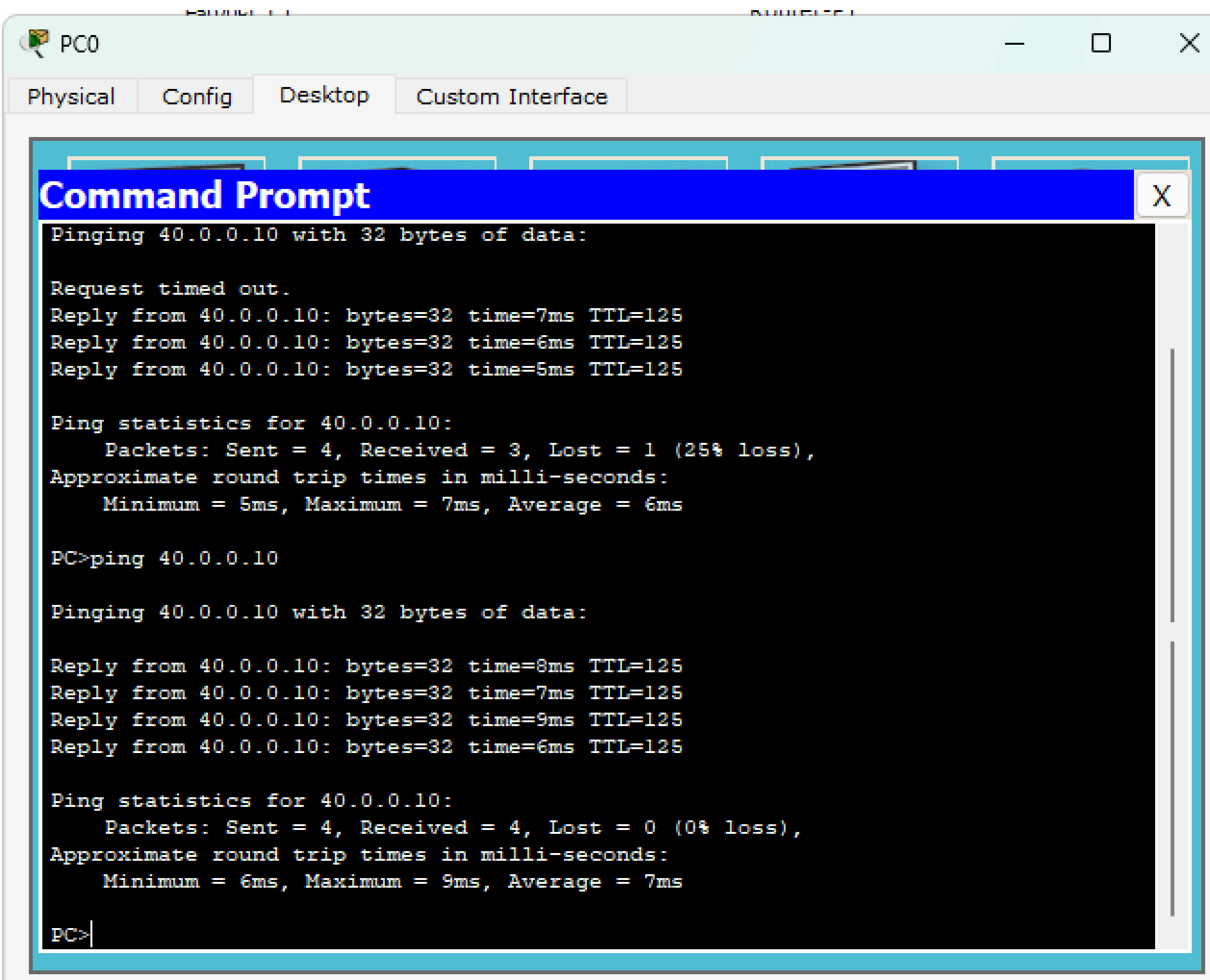
S 10.0.0.0/8 [1/0] via 20.0.0.1
C 20.0.0.0/8 is directly connected serial 2/0
C 30.0.0.0/8 is directly connected serial 3/0
S 40.0.0.0/8 [1/0] via 30.0.0.2

Observation:
if configuration and cabling are correct, you will receive successful ping replies b/w the two PC's

3/1/25

Screen Shots:




PC0
Physical
Config
Desktop
Custom Interface
Command Prompt
X
Pinging 40.0.0.10 with 32 bytes of data:
Request timed out.
Reply from 40.0.0.10: bytes=32 time=7ms TTL=125
Reply from 40.0.0.10: bytes=32 time=6ms TTL=125
Reply from 40.0.0.10: bytes=32 time=5ms TTL=125
Ping statistics for 40.0.0.10:
Packets: Sent = 4, Received = 3, Lost = 1 (25% loss),
Approximate round trip times in milli-seconds:
Minimum = 5ms, Maximum = 7ms, Average = 6ms
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=8ms TTL=125
Reply from 40.0.0.10: bytes=32 time=7ms TTL=125
Reply from 40.0.0.10: bytes=32 time=9ms TTL=125
Reply from 40.0.0.10: bytes=32 time=6ms TTL=125
Ping statistics for 40.0.0.10:
Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
Minimum = 6ms, Maximum = 9ms, Average = 7ms
PC>

EXPERIMENT-4

Aim: Configure DHCP within a LAN and outside LAN.

Topology , Procedure and Observation:

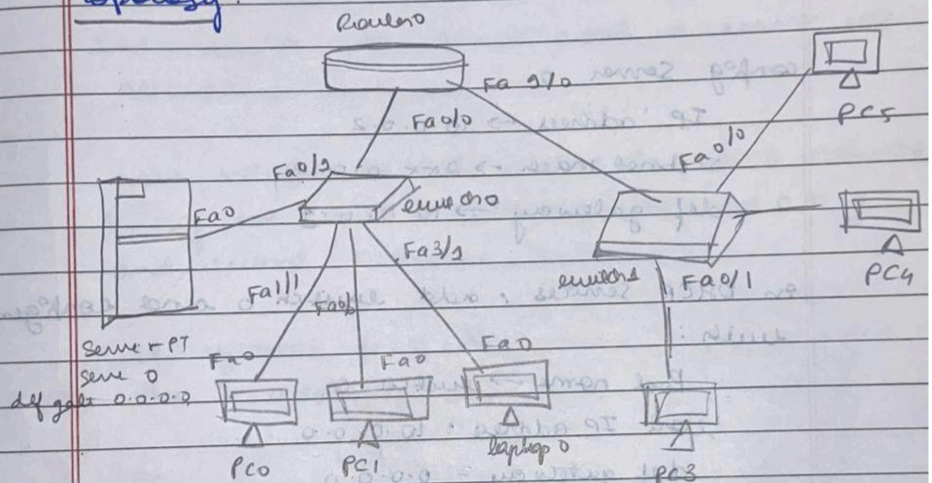
Date: Page:

Experiment -4.

a) Configure DHCP within a LAN & outside LAN

Aim: Demonstrate DHCP within LAN and outside a LAN

Topology:



1. switch 0 connected router0 interface fa0/0 using copper-straight-through cable from

2. PC0, PC1, PC2 are connected to switch 0 via copper straight cable with IP address 10.0.0.2, 10.0.0.3 and 10.0.0.4 respectively

3. Server 0 is connected to switch 0 with IP address 10.0.0.2

4. PC3, PC4, PC5 are connected to switch 1 with IP address → PC3: 20.0.0.3
PC4: 20.0.0.4
PC5: 20.0.0.5.

Procedure:

- Router: Place the router in the middle
- Switch: Connect two switches to router
- PC: take 3 PC's & connect it to switch 0 and another 3 PC's to switch 1.
- Server: place & connect it to the switch 1 via cable.

config server 0.

IP address → 10.0.0.2

subnet mask → 255.0.0.0

def gateway → 10.0.0.1

In DHCP services, add switch 0 and configure with:

Pool name → switch 0

start IP address: 10.0.0.0

def gateway - 0.0.0.0

subnet mask - 255.0.0.0

In DHCP services, add switch 1 configure with poolname - switch 1

start - IP address - 10.0.0.3

def gateway - 80.0.0.1

subnet mask - 255.0.0.0

→ set IP configurations of all PC's PC0 to PC1 to DHCP due to which each is alone, its IP address, subnet mask, default gateway

Date: Page:

Observation:

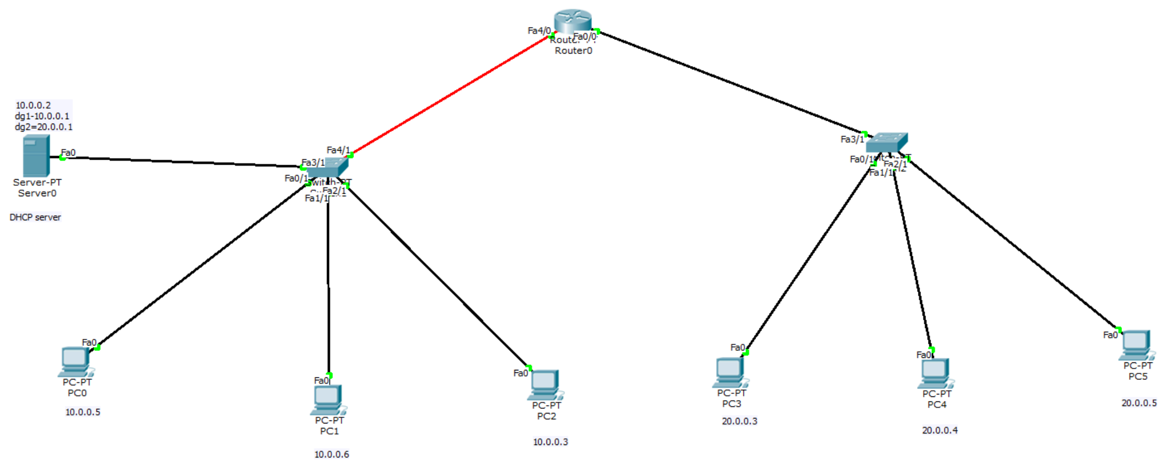
when we have to dynamically assign IP address to another network we will do it using a router and a server when we configure and connect router 1 to switch 0 to switch 1 using, so that we will be using ip-helper add run 10.0.0.2, 2nd network can access pool in server of network 1.

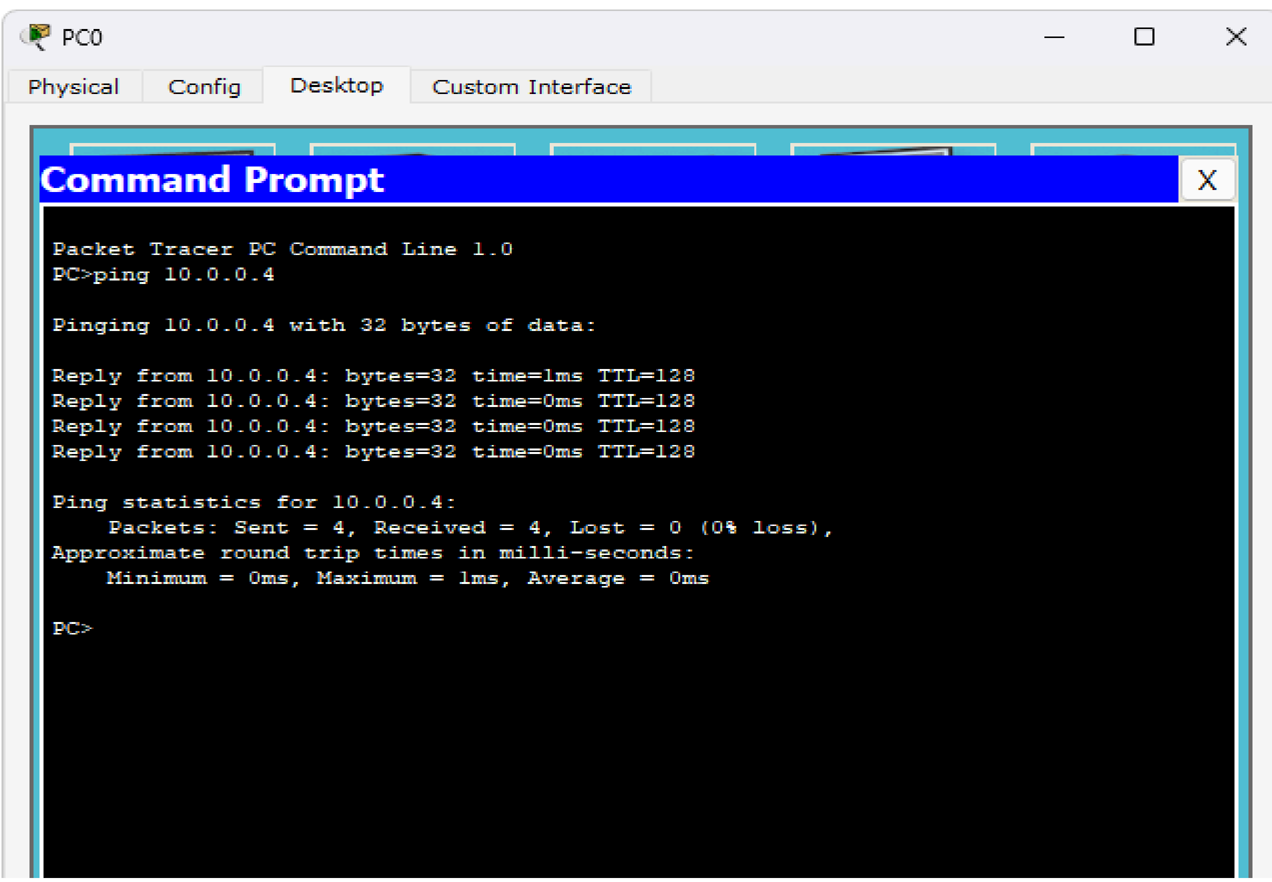
now we go to pc's and select DHCP network 1 if we will be starting from 10.0.0.8 and second network 20.0.0.3.

all systems will be dynamically assigned the IP address.

3/1/25

Screen Shots:

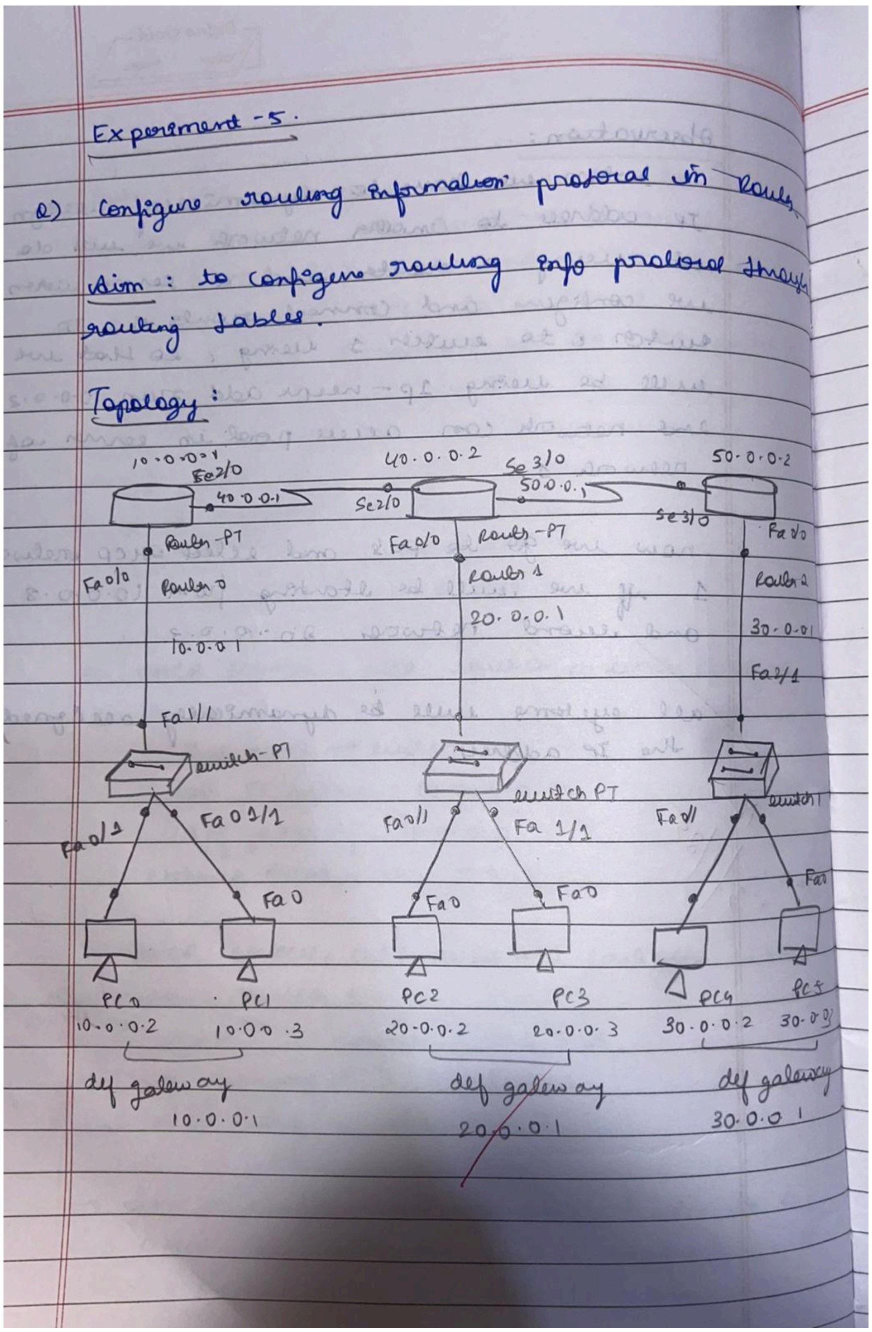




EXPERIMENT-5

Aim: Configure RIP routing Protocol in Routers .

Topology , Procedure and Observation:



Bafna Gold
Date: Page:

1. 3 routers, 3 switches and 6 - PC's.

Router → ip → 10.0.0.1
switch 1 → Fa 0/1 → PC0 → 10.0.0.2.
subnet : 255.0.0.0

Fa 1/1 → PC1 : ip : 10.0.0.3
subnet : 255.0.0.0
gateway for PC0 and PC1 is 10.0.0.1

2 - Router 2 — ip 1 → 40.0.0.2
ip 2 → 50.0.0.2
switch 2 → Fa 0/0 → 20.0.0.1
Fa 0/1 → PC2 : ip : 20.0.0.2, subnet : 255.0.0.0
Fa 1/1 → PC3 → ip: 20.0.0.3, subnet : 255.0.0.0
gateway for PC2 and PC3 is 20.0.0.1

3- Router 3 → ip - 50.0.0.2
switch 3 → Fa 0/0 → 30.0.0.1
Fa 0/1 → PC4 — ip 30.0.0.2, subnet : 255.0.0.0
Fa 1/1 → PC5 → ip: 30.0.0.3, subnet : 255.0.0.0

gateway for PC4 and PC5 is 30.0.0.1.

Procedure :
1. connect the end devices to the switches through copper - straight through wire

2. connect the router to the other router to through serial dce.

3. Router - switches [copper - straight - through]

→ Configure all the 8 routers :

Router 0

Router > enable

config t

interface serial 2/0.

ip address 40.0.0.1 255.0.0.0.

no shut

exit

interface fastethernet 0/0

ip address 10.0.0.1 255.0.0.0.

no shutdown

exit

Router 1

Router >enable

config t

interface serial 2/0

ip address 40.0.0.1

no shutdown

exit

interface Se 3/0

ip address 80.0.0.1 255.0.0.0

no shutdown.

P.T.O.

Router 2.

Router (config)# interface Fa 0/0
ip address 30.0.0.1 255.0.0.0
interface Se 2/0.
ip address 50.0.0.2 255.0.0.0

→ Configure the route.

Router 0:

Router (config)# router rip
network 10.0.0.0
network 40.0.0.0

Router 1:

Router (config)# router rip
network 40.0.0.0
network 20.0.0.0
network 50.0.0.0

Router 3:

router (config)# router rip
network 30.0.0.0
network 50.0.0.0

Test connectivity:

ping 30.0.0.1

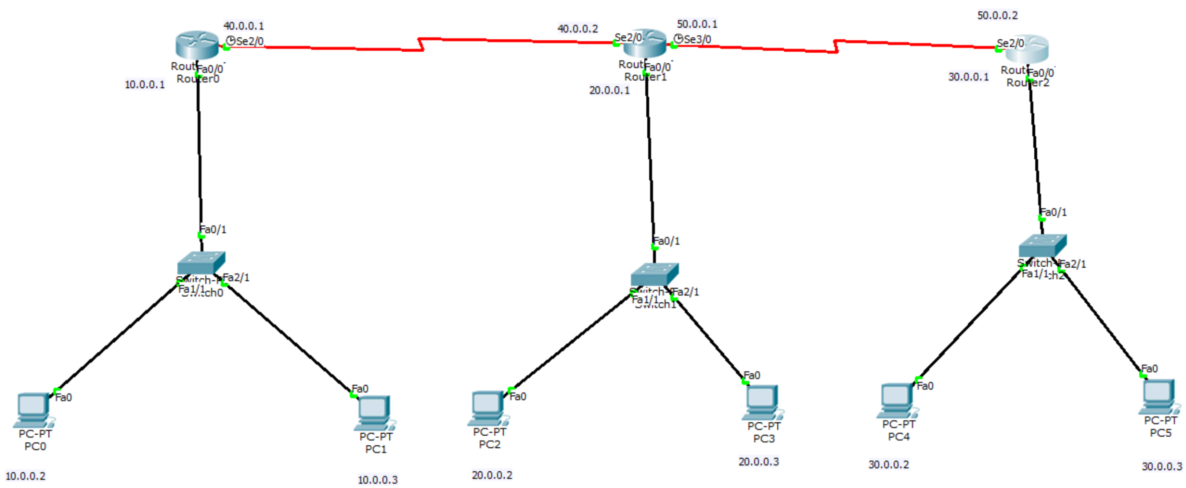
sent =4 loss =0 100%

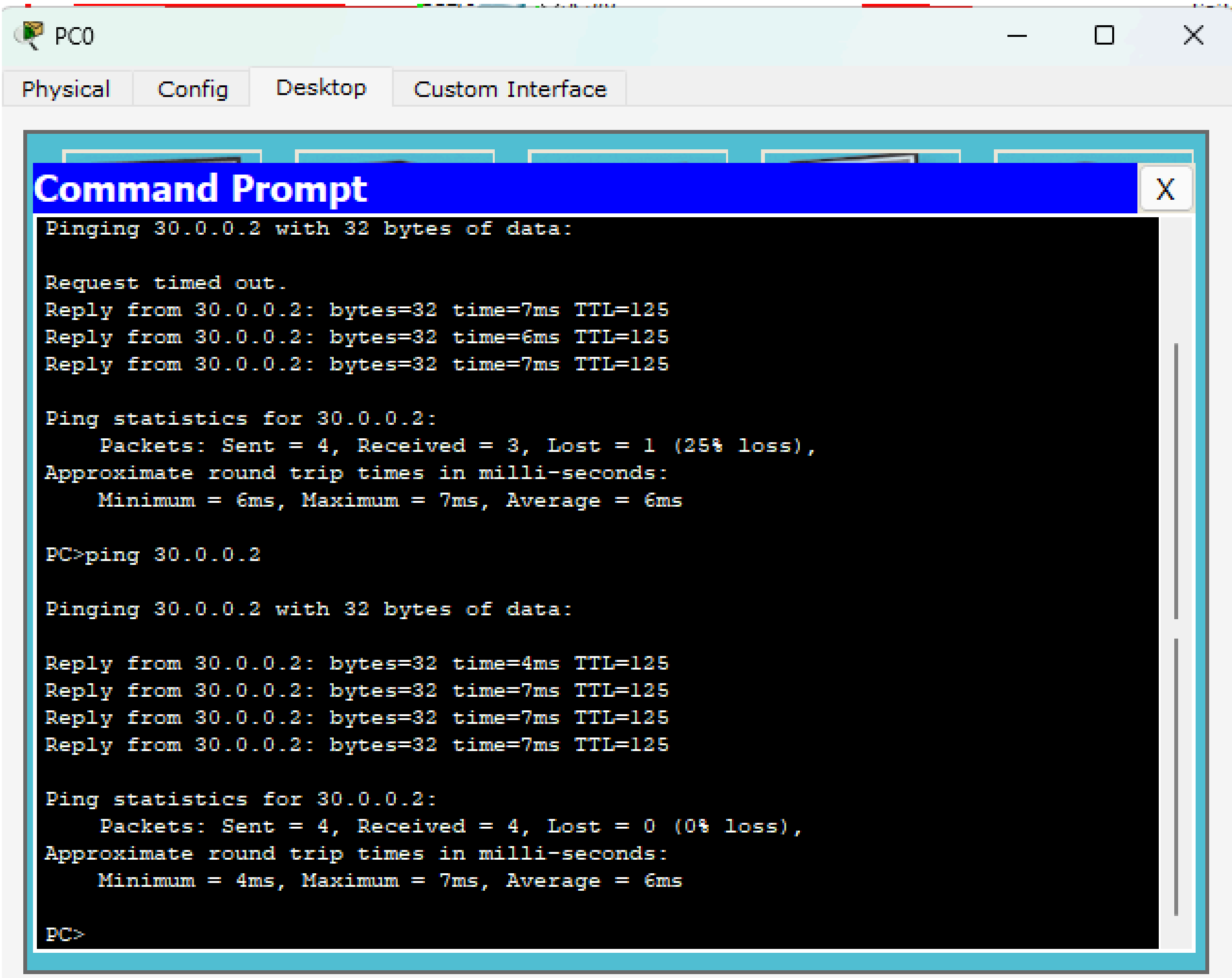
Observation :

The routers communicate with each other, share a common routing table, once RIP is installed / activated in routers. Every router shares its routing protocol with its neighbors. Hence after along every router will know about all info that their neighbors are connected to.

3/1/25

Screen Shots:




PC0
Physical
Config
Desktop
Custom Interface
Command Prompt
X
Pinging 30.0.0.2 with 32 bytes of data:
Request timed out.
Reply from 30.0.0.2: bytes=32 time=7ms TTL=125
Reply from 30.0.0.2: bytes=32 time=6ms TTL=125
Reply from 30.0.0.2: bytes=32 time=7ms TTL=125
Ping statistics for 30.0.0.2:
Packets: Sent = 4, Received = 3, Lost = 1 (25% loss),
Approximate round trip times in milli-seconds:
Minimum = 6ms, Maximum = 7ms, Average = 6ms
PC>ping 30.0.0.2
Pinging 30.0.0.2 with 32 bytes of data:
Reply from 30.0.0.2: bytes=32 time=4ms TTL=125
Reply from 30.0.0.2: bytes=32 time=7ms TTL=125
Reply from 30.0.0.2: bytes=32 time=7ms TTL=125
Reply from 30.0.0.2: bytes=32 time=7ms TTL=125
Ping statistics for 30.0.0.2:
Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
Minimum = 4ms, Maximum = 7ms, Average = 6ms
PC>

<u>**EXPERIMENT-6**</u>

Aim: Demonstrate the TTL/ Life of a Packet .

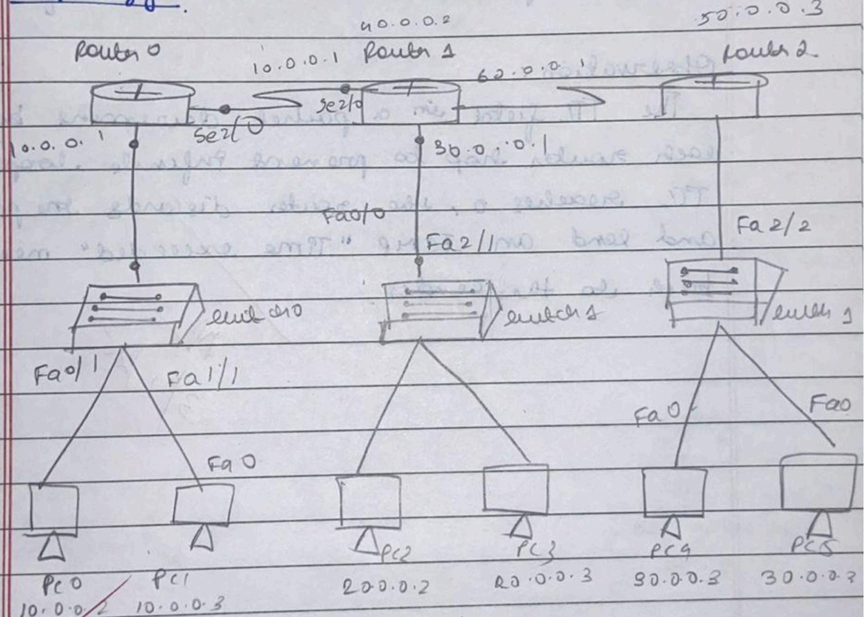
Topology , Procedure and Observation:

Experiment - 6.

Q) demonstrate the TTL or life of a packet.

Aim: To demonstrate how TTL (time to live) works including the decrement of TTL at each hop and its handling of expired packets

Topology:



1. Connect Router 0 to Router 1 and Router 1 to Router 2 using a serial-DCE cable
2. connect Router 0 to switch0 using copper straight cable IIIly repeat for router 1 to switch1 and router 2 to switch2.
3. connect 6PC's (2 each) to each of the switches using copper-straight cable and assign ip addresses.

Procedure:

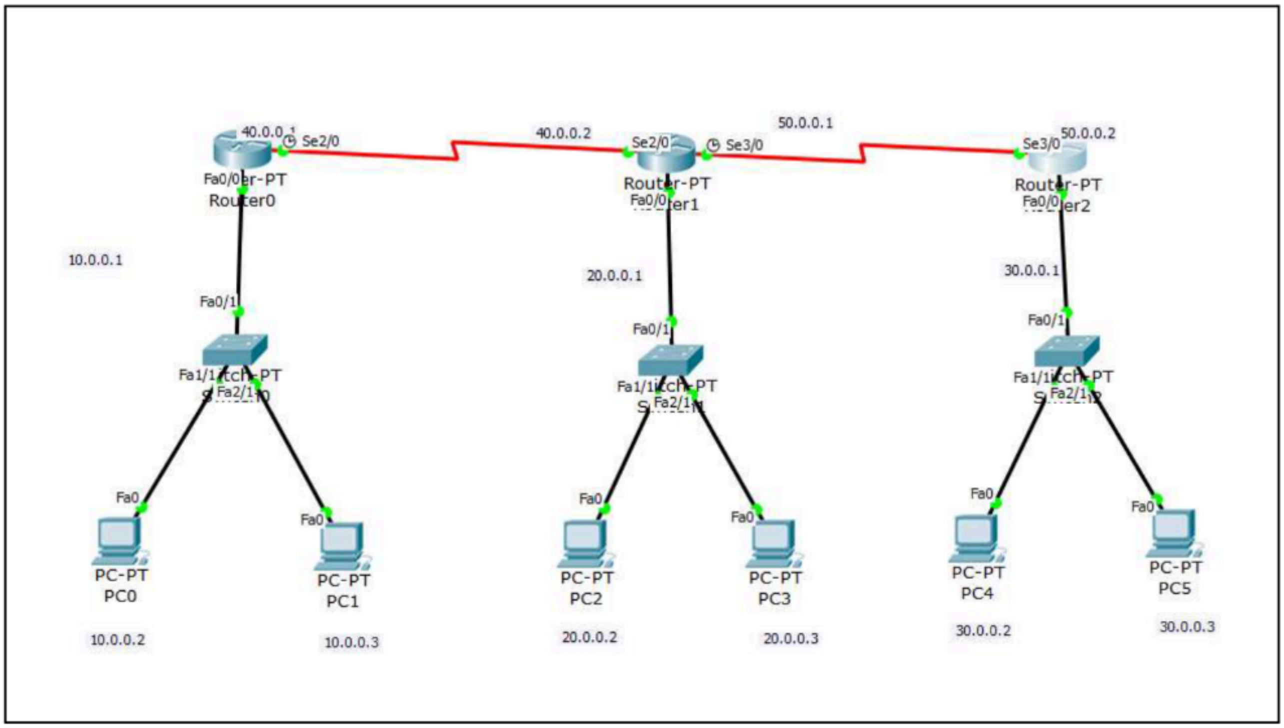
1. open CPT and establish the topology
2. select the simple PDU icon and select source and destination
3. Include the PC0 and PC5 as part of the communication that is to take place, then switch to simulation mode.
4. Start the simulation by clicking on auto capture / play button & observe the TTL of a packet.

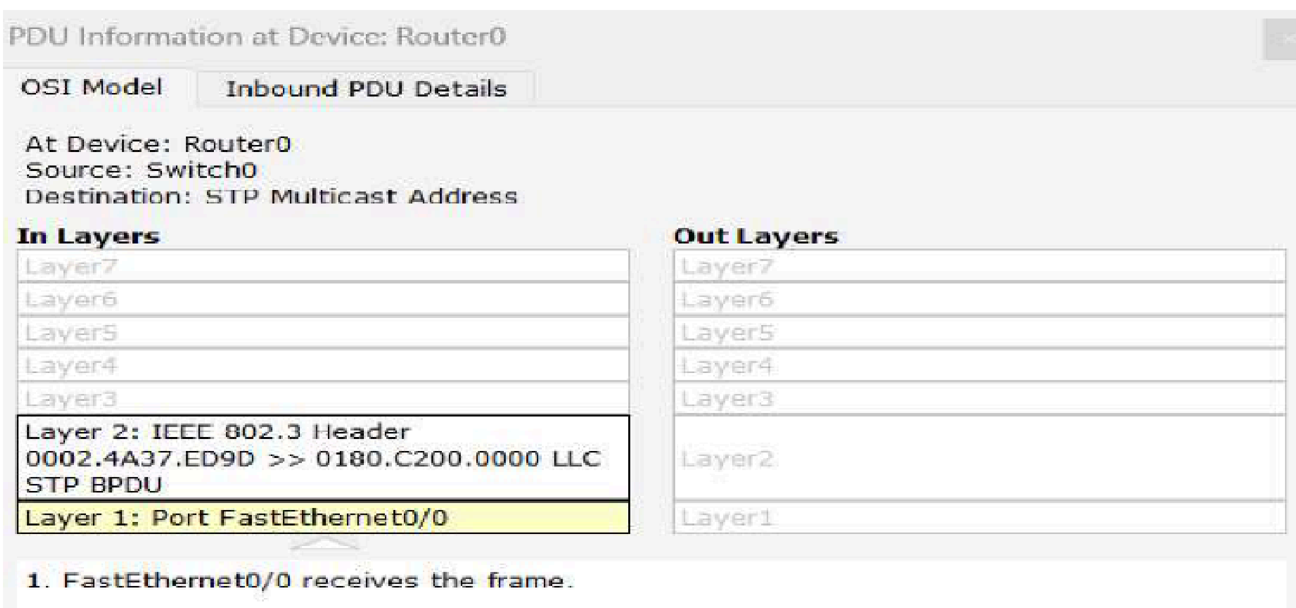
Observation:

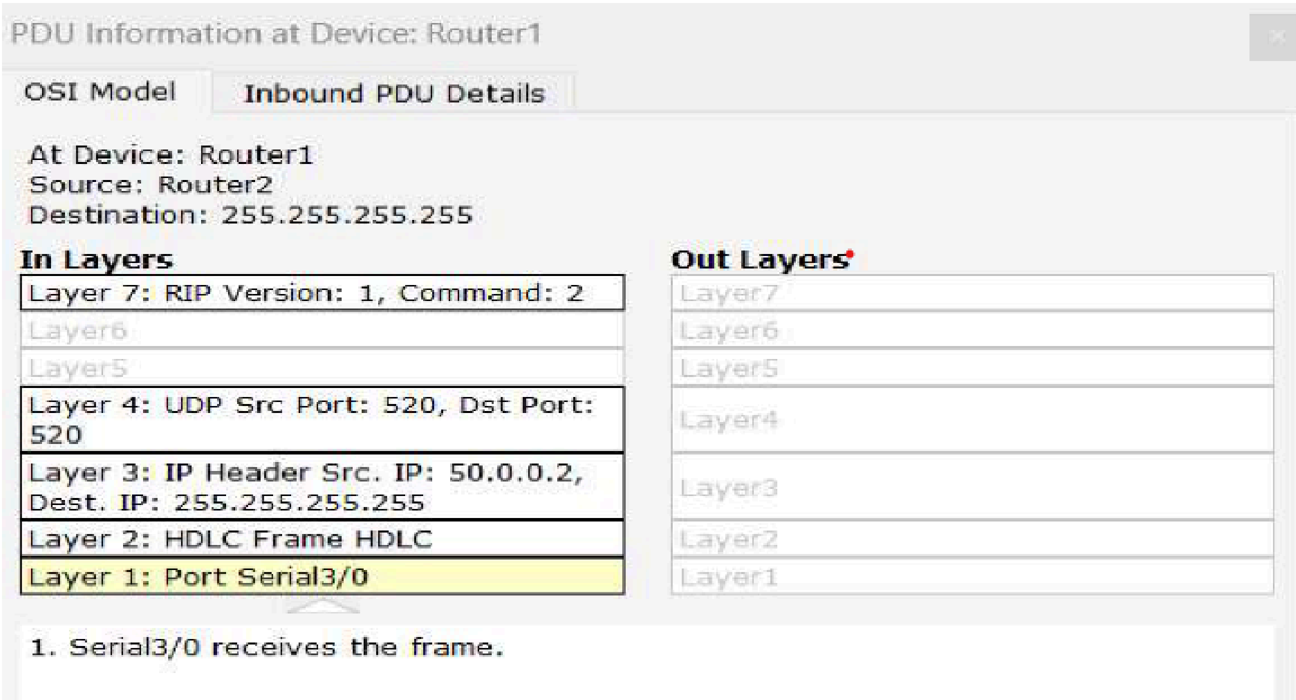
The TTL field in a packet decrements by 1 at each router hop to prevent infinite loops. If the TTL reaches 0, the router discards the packets and send an ICMP "Time exceeded" message is back to the sender.

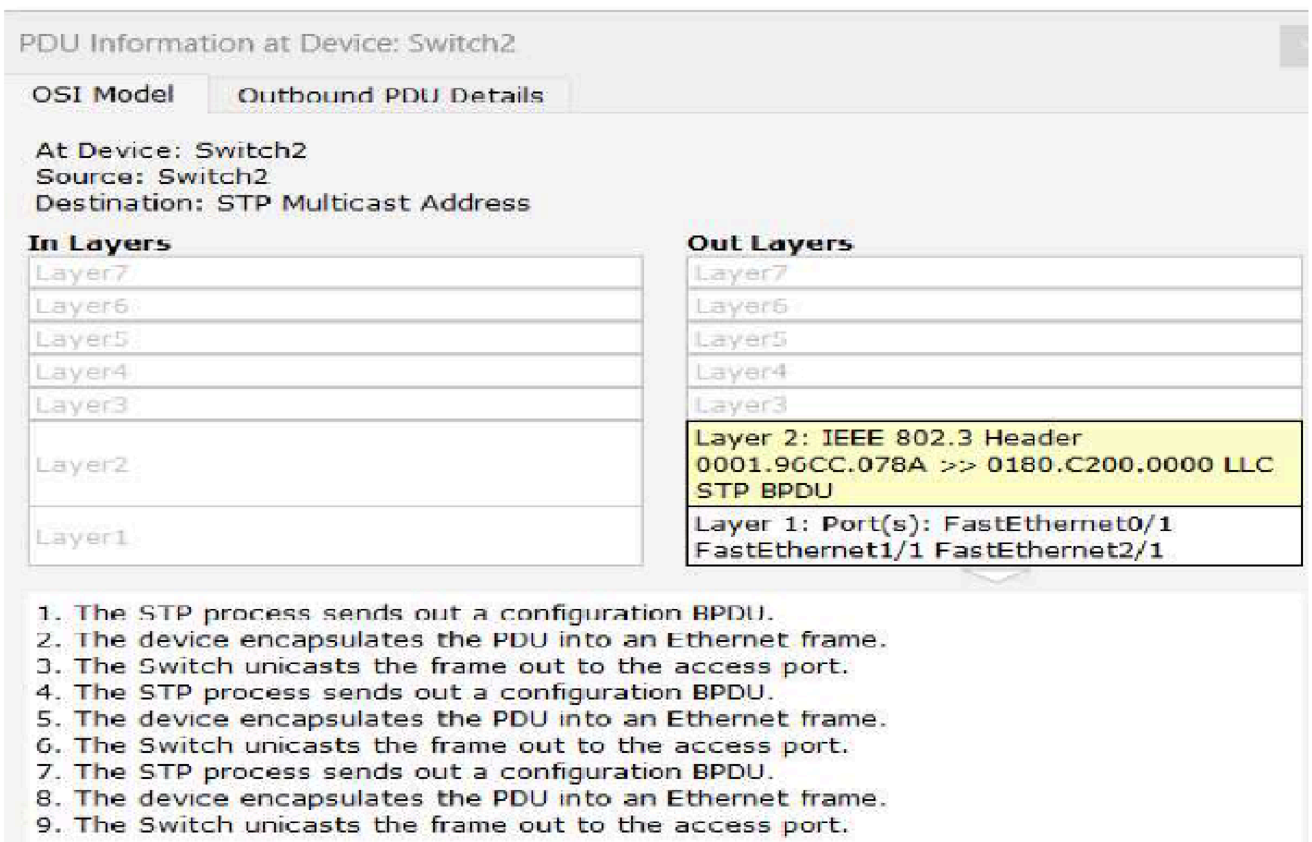
3/1/25

Screen Shots:

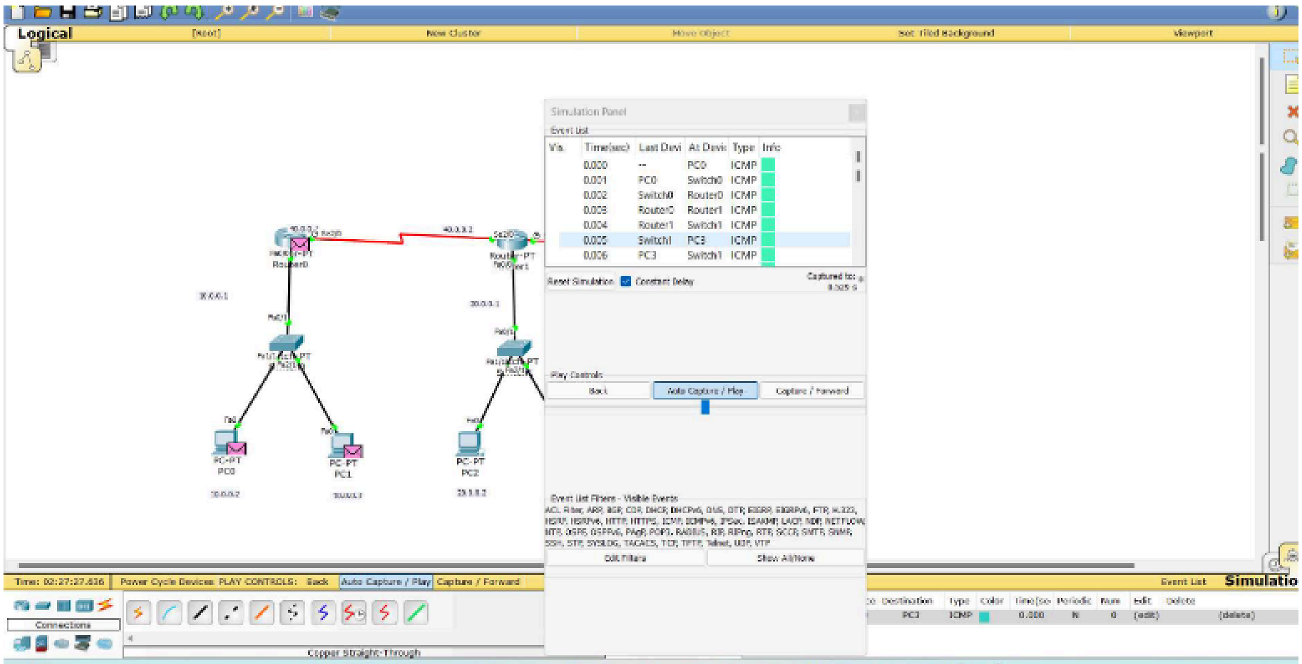


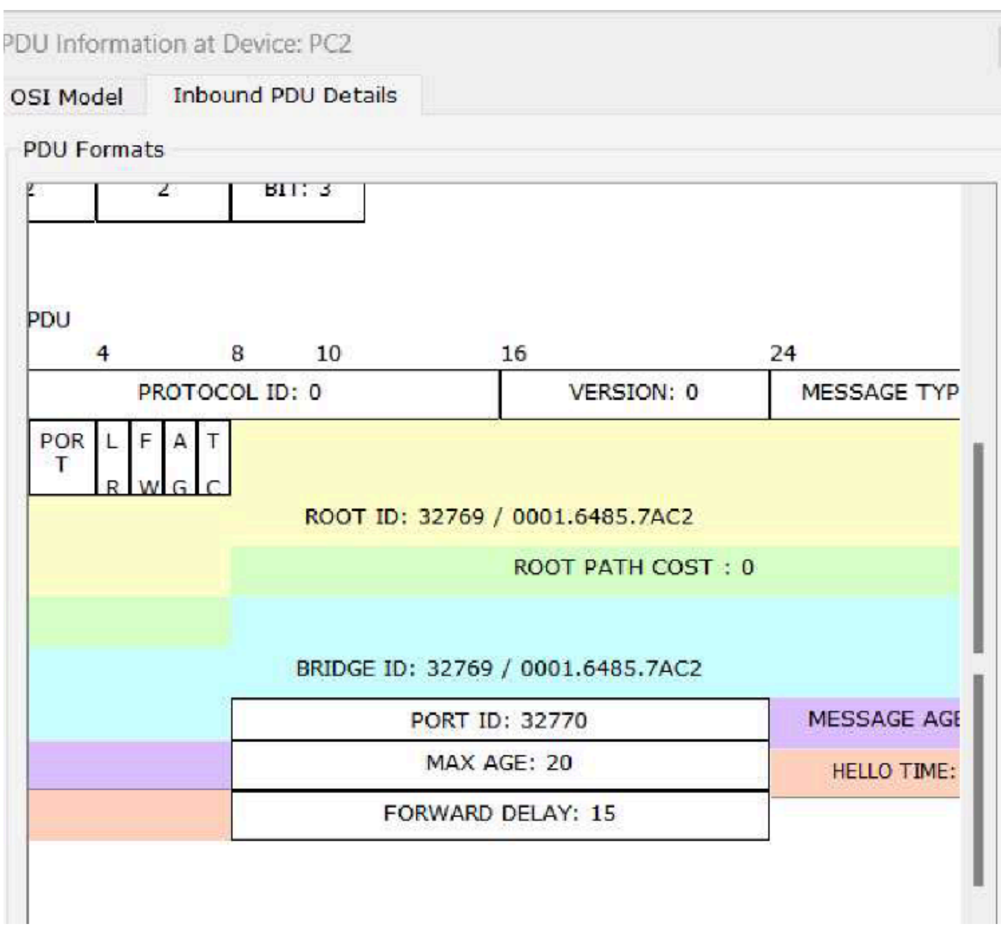






Fire	Last Status	Source	Destination	Type	Color	Time(se	Periodic	Num	Edit	Delete
●	Successful	PC0	PC3	ICMP		0.000	N	0	(edit)	(delete)

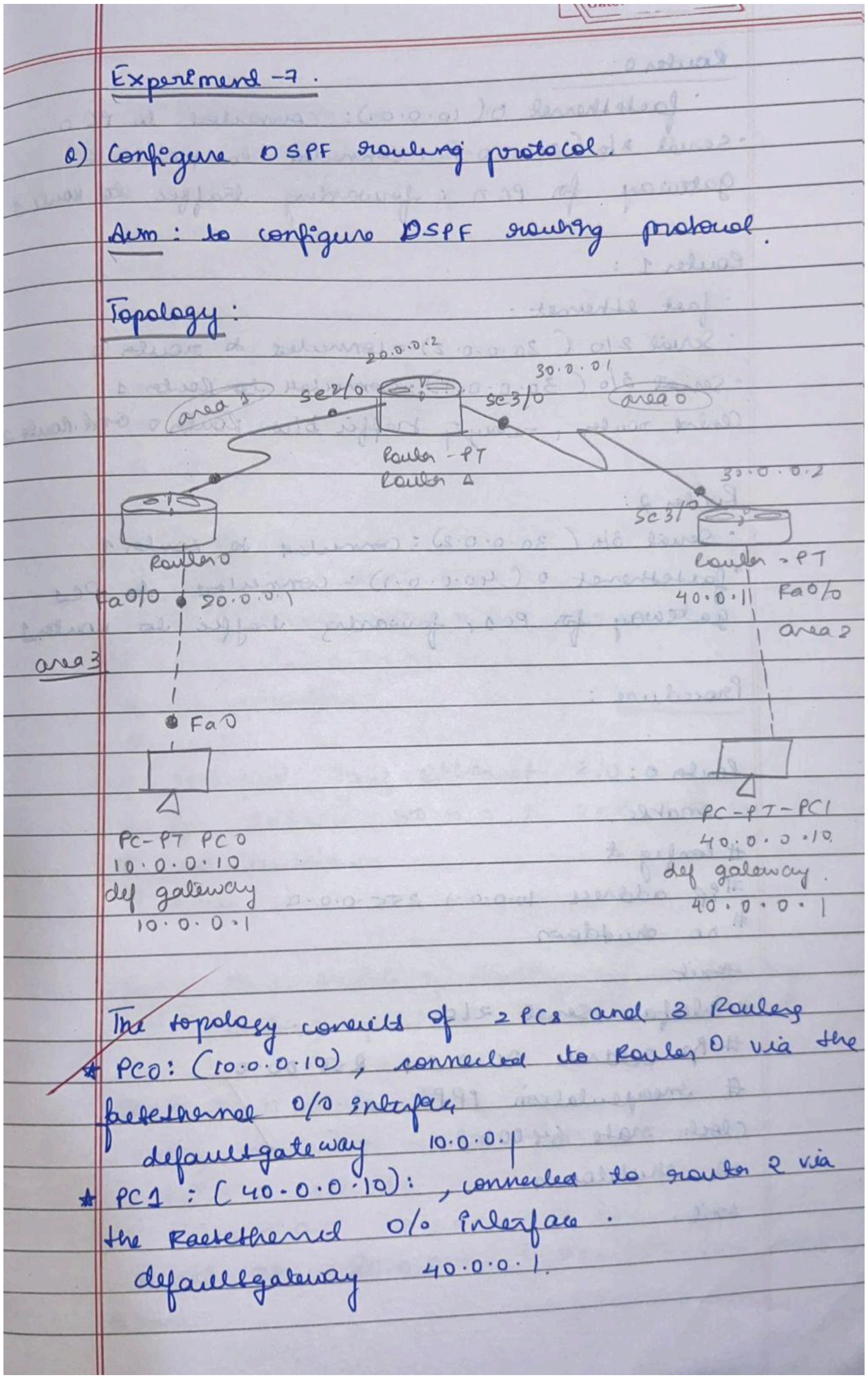

Logical
Simulation Panel
Event List
Vis. Time(sec) Last Devi At Devi Type Info
0.000 -- PC0 ICMP
0.001 PC0 Switch0 ICMP
0.002 Switch0 Router0 ICMP
0.003 Router0 Router1 ICMP
0.004 Router1 Switch1 ICMP
0.005 Switch1 PC3 ICMP
0.006 PC3 Switch1 ICMP
Play Controls
Back
Auto Capture / Play
Capture / Forward
Simulatio


PDU Information at Device: PC2
OSI Model
Inbound PDU Details
PDU Formats
BIT: 3
PDU
4
8
10
16
24
PROTOCOL ID: 0
VERSION: 0
MESSAGE TYP
POR T
L R
F W
A G
T C
ROOT ID: 32769 / 0001.6485.7AC2
ROOT PATH COST : 0
BRIDGE ID: 32769 / 0001.6485.7AC2
PORT ID: 32770
MESSAGE AGE
MAX AGE: 20
HELLO TIME:
FORWARD DELAY: 15

EXPERIMENT-7

Aim: Configure OSPF routing protocol .

Topology , Procedure and Observation:



Experiment -7 .

Q) Configure OSPF routing protocol.

Aim : to configure OSPF routing protocol.

Topology :

The topology consists of 2 PCs and 3 Routers

* PC0: (10.0.0.10), connected to Router 0 via the fastethernet 0/0 interface
  defaultgateway 10.0.0.1
* PC1 : (40.0.0.10): , connected to router 2 via the fastethernet 0/0 interface.
  defaultgateway 40.0.0.1.

Router 0:

· fastethernet 0 (10.0.0.1): connected to PC 0

· serial 2/0 (20.0.0.1): connected to Router 1

gateway for PC 0, forwarding traffic to Router 1

Router 1:

· fast ethernet ·

· Serial 2/0 (20.0.0.2): connected to router 0

· Serial 3/0 (30.0.0.1): connected to Router 1

Central router, relaying traffic b/w Router 0 and Router 2

Router 2:

· Serial 3/0 (30.0.0.2): connected to Router 1

· fastethernet 0 (40.0.0.1): connected to PC1

gateway for PC1, forwarding traffic to Router 1

Procedure:

Router 0:

enable

\# config t

\# ip address 10.0.0.1 255.0.0.0

\# no shutdown

\# exit

\# interface serial 2/0

\# ip address 20.0.0.1 255.0.0.0

\# encapsulation PPP

clock rate 64000

no shutdown

exit.

In Router 1 :

interface serial 2/0

ip address 20.0.0.2 255.0.0.0

encapsulation ppp

no shutdown

exit

interface serial 3/0

ip address 30.0.0.1 255.0.0.0

encapsulation ppp

no shutdown

exit

In Router 2 :

interface serial 3/0

ip address 30.0.0.2 255.0.0.0

encapsulation ppp.

no shutdown.

exit

interface fast ethernet 2/0.

ip address 40.0.0.1 255.0.0.0

no shutdown

exit .

step 3 : Now, enable ip routing by configuring ospf routing protocol in all routers.

Router 0 : # router ospf 1

(config-router) # router -id- 1.1.1.1

Router . # network

10.0.0.0 0.255.255.255 area 3.

network 20.0.0.0 0.255.255.255 area 1

exit

Router 1 :

router ospf 1.
router -id 2.2.2.2
network 20.0.0.0 0.255.255.255 area 1
~~exit~~ # network
30.0.0.0 0.255.255.255 area 2

exit

Router 2:

router ospf 1.
router - id : 3:3.3.3.
network 30.0.0.0 0.255.255.255 area 2
exit

step 4:- now check routing table of R1:
show ip route.
C 10.0.0.0/8 is directly connected, Fa2/0
C 20.0.0.0/8 is directly connected, serial 2/0
O IA 40.0.0.0/8 [40/129] via 20.0.0.2,
00:04:23, serial 2/0
O IA 30.0.0.0/8 [110/129] via 20.0.0.2,
00:07:29, serial 3/0

Configure loopback:
Router 0:
Router # interface loopback 0
ip add 172.10.1.252 255.255.0.0

Router 1 :
interface loopback 0
ip add 172.10.1.253 255.255.0.0
no shutdown

Date: Page:

Router 2:

interface loopback 0

ip add 172.16.1.25 4.255.255.0

no shutdown.

step 5: now, Check Routing table of R3

→ show ip route

O IA 90.0.0.0/8 [100/128] via 30.0.0.1, 00:18, serial 3/0,

C 40.0.0.8/8 is directly connected, Fa2/0

C 30.0.0.8/8 is directly connected, serial 3/0

step 6: create Virtual link between R1, R2,, by this we create a virtual link to connect area 3 to area 0.

Router 0:

router ospf 1

area 1 virtual-link 2.2.2.2

Router # 4 Feb 10 - - - -

from LOADING to full, loading done

Router 1:

Router (Router-config)# router ospf 1

area 1 virtual-link 1.1.1.1

exit

step 7: R2 and R8 get updates about Area 3

Router 2: show ip route

O IA 20.0.0.0/8 [110/128] via 30.0.0.1, 00:00:00 ; 56, serial 2/0

O IA 10.0.0.0/8 [110/128] via 30.0.0.1, 00:01:56, serial 1/0

C 30.0.0.0/8 is directly connected, serial3/0

Step 8: check connectivity between PC0
40.0.0.10 to 40.0.0.10

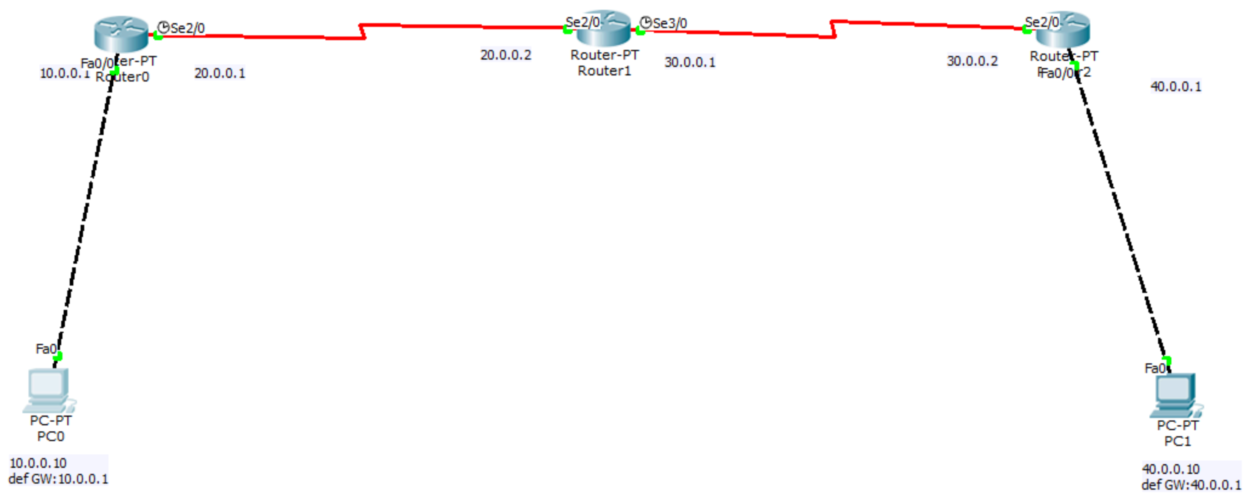
PC0 command prompt
PC> ping 40.0.0.10
pinging 40.0.0.10 with 32 bytes of data
Reply from 40.0.0.10: bytes=32 time=3ms
TTL=125.
.
;
Reply from 40.0.0.10.

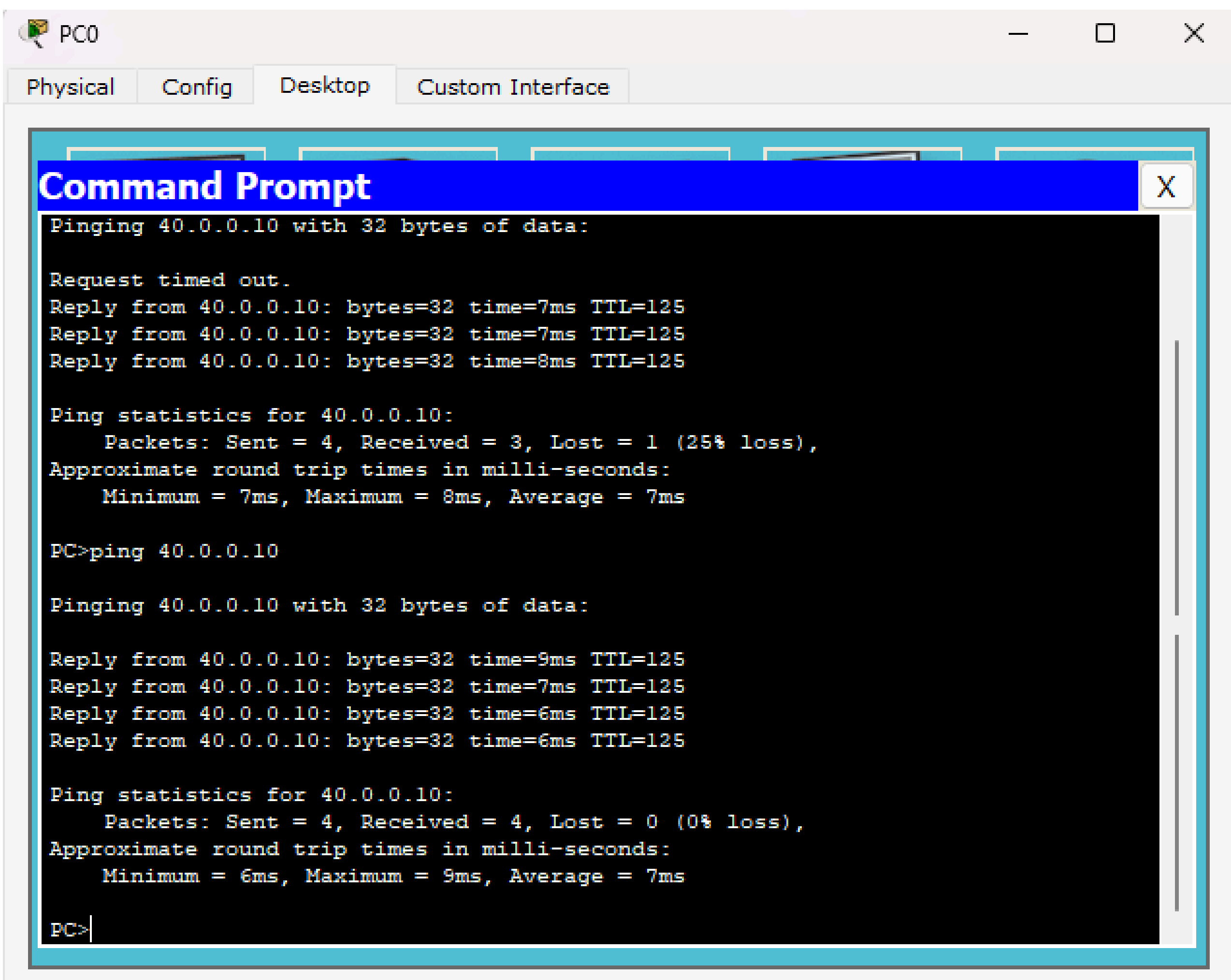
Ping statistics for 40.0.0.10.
Packets: sent=4, Received=4, Loss=0 (0% loss)

Observation:
The expt demonstrates how OSPF dynamically learns & advertises routes, enabling efficient and scalable routing across multiple areas

→ Routing tables on all routers must display networks from all areas with O IA indicating inter-area routes

Screen Shots:





EXPERIMENT-8

Aim: Configure Web Server, DNS within a LAN.

Topology , Procedure and Observation:

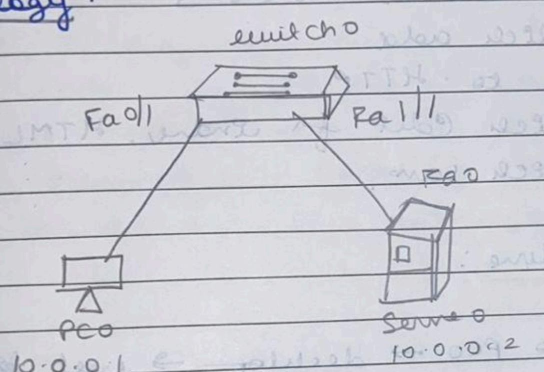
Date: Page:

Experiment -12.

Q. Configure web server & DNS within a LAN

Aim: To configure DNS server to demonstrate the mapping of IP addresses & domain name

Topology:



Connect a PC and a server to a switch, assign ip address as 10.0.0.1 & 10.0.0.2 resp.

Configuration:

open CPT & arrange as given in topology and configure the devices as given below

PC0:

IP address : 10.0.0.2

Server 0:

IP address : 10.0.0.3

Connect PC0 & Server 0 via a switch PT

PC0 connects to switch on interface FA0 & switch on Fa 0/1

Server connects to switch on interface FA0 & switch on Fa1/1

Server 0 :
Go to Server → Services → DNS
Enable on

In the text fields add :
name : abc.
address : 10.0.0. 3
click add
go to HTTP.
click edit for index. HTML (change if needed)
click save.

Procedure :

1. Go to PC0 → desktop → web browser
2. Search 'abc' in url bar (or)
3. Search 10.0.0. 2 in url bar

output : for both 'abc' & 10.0.0. 3.

C P T
welcome to CPT. Opening door to new opportunities. Mind wide open.

Quick Links :
A small page
copyrights
Image Page
Image

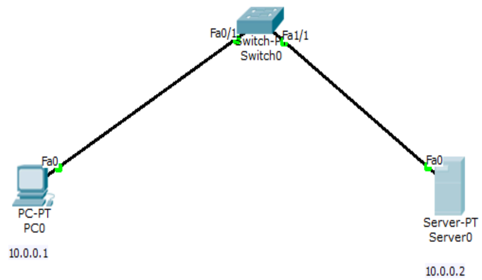
Bafna Gold
Date: Page:

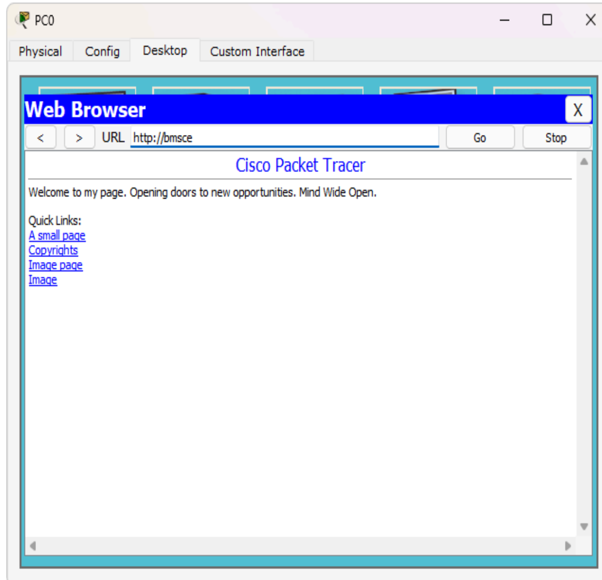
observations :

DNS translates domain names to ip addresses. It simplifies accessing websites by using human-readable names
In this experiment, a web server & DNS were configured within a LAN to map domain names to ip address. The PC0 successfully accessed the server 0. by both its ip address & the configured domain name 'abc'. The configuration was successful allowing the web page to be accessed via both methods.

3/1/25

Screen Shots:





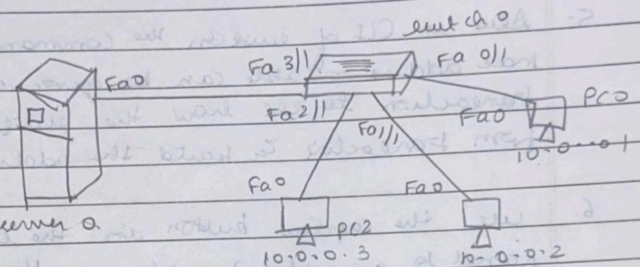
EXPERIMENT-9

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

Topology , Procedure and Observation:

Experiment - 8: To construct a simple LAN & understand concept & operation of ARP

Aim: construct a simple LAN simulate operation of address resolution protocol.



1. switch connected to 3 PCs and a server via three fastethernet interfaces & one ethernet interface respectively.

2. All connections made via copper straight-through cable

Procedure:

1. open CPT and drag the following switch and PCs place 3 PC's each connected to switch 0 and Server: place 1 server & connect it to switch 0

2. Assign an IP address & subnet mask to all the devices then connect them via a switch

3. Use the inspect tool ('a'), click on a PC to view

ARP table

3. display the ARP table of all devices

4. Initially ARP is empty for all

5. Also in CLI of switch, the command = show mac address - table can be given on every transaction to see how the switch learns from transaction to build the address table

6. Use the capture button in the simulation panel to go step by step so that changes in ARP can be clearly noted

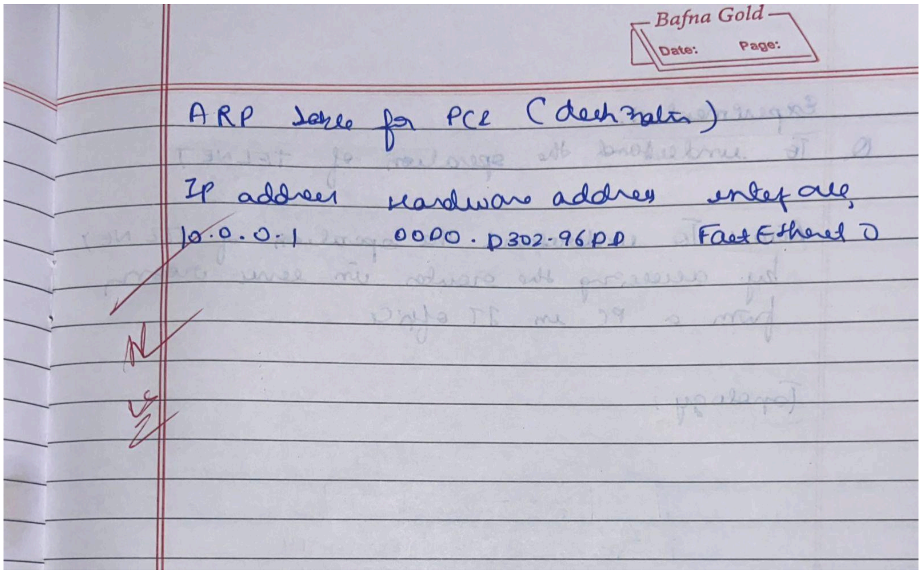
7. Observe the switch as well as nodes update the ARP table as and when new communication starts.

Observation:

As the message travels from one source host to its destination host the ARP table of all devices get updated
ARP maps an IP address to a MAC address. It ensures communication within a local network

ARP table for PC0 (source):

IP address	Hardware address	interface
10.0.0.3	0060.2F29.2CB8	FastEthernet0

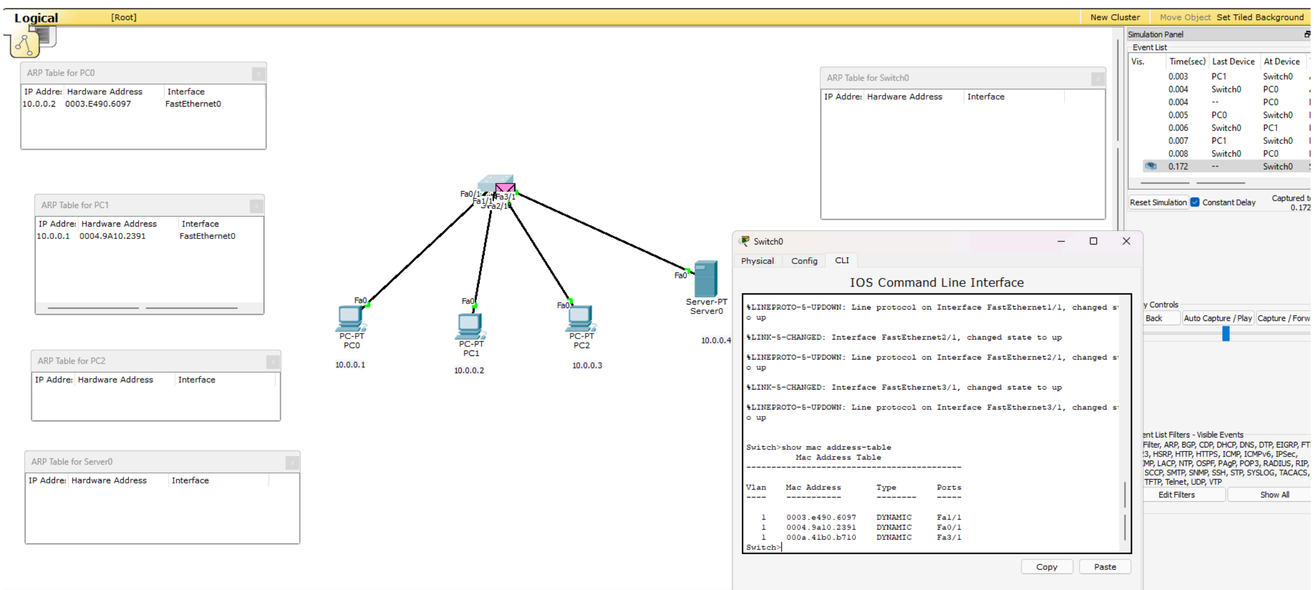


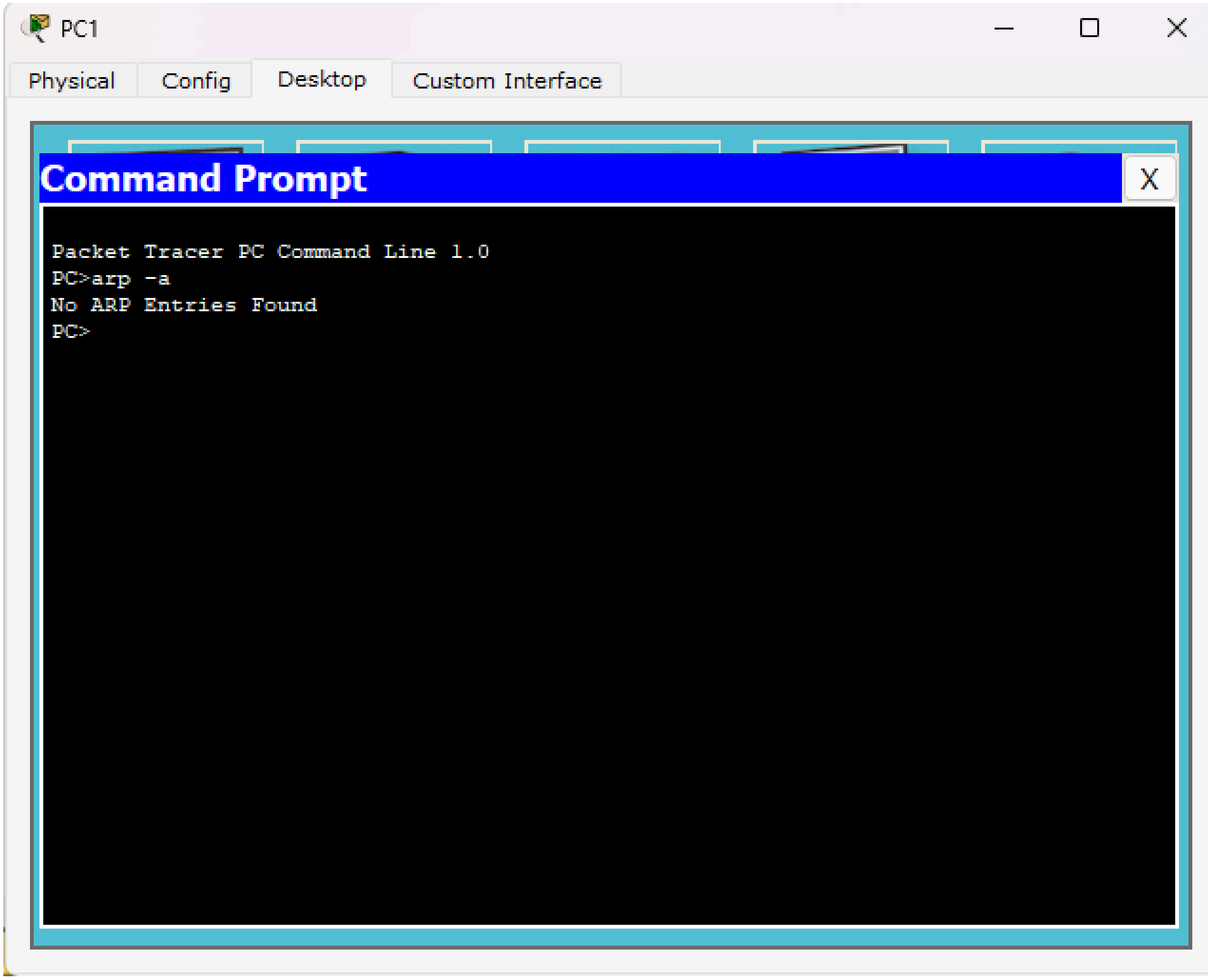
ARP Table for PC2 (destination)

IP address   Hardware address   interface

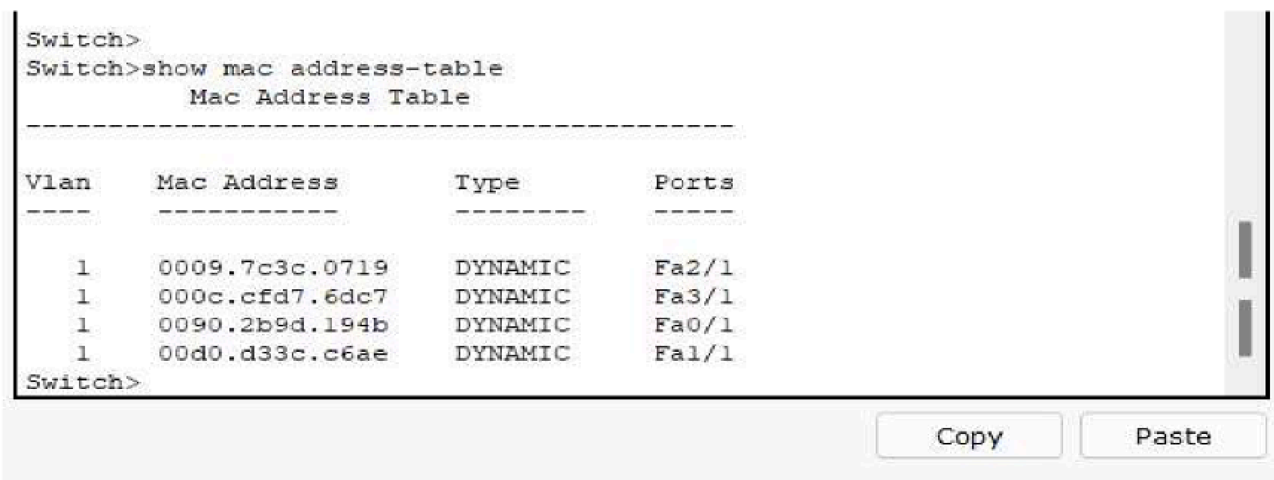
10.0.0.1   0000.D302.96DD   FastEthernet0

Screen Shots:




PC1
Physical
Config
Desktop
Custom Interface
Command Prompt
X
Packet Tracer PC Command Line 1.0
PC>arp -a
No ARP Entries Found
PC>

SWITCH:

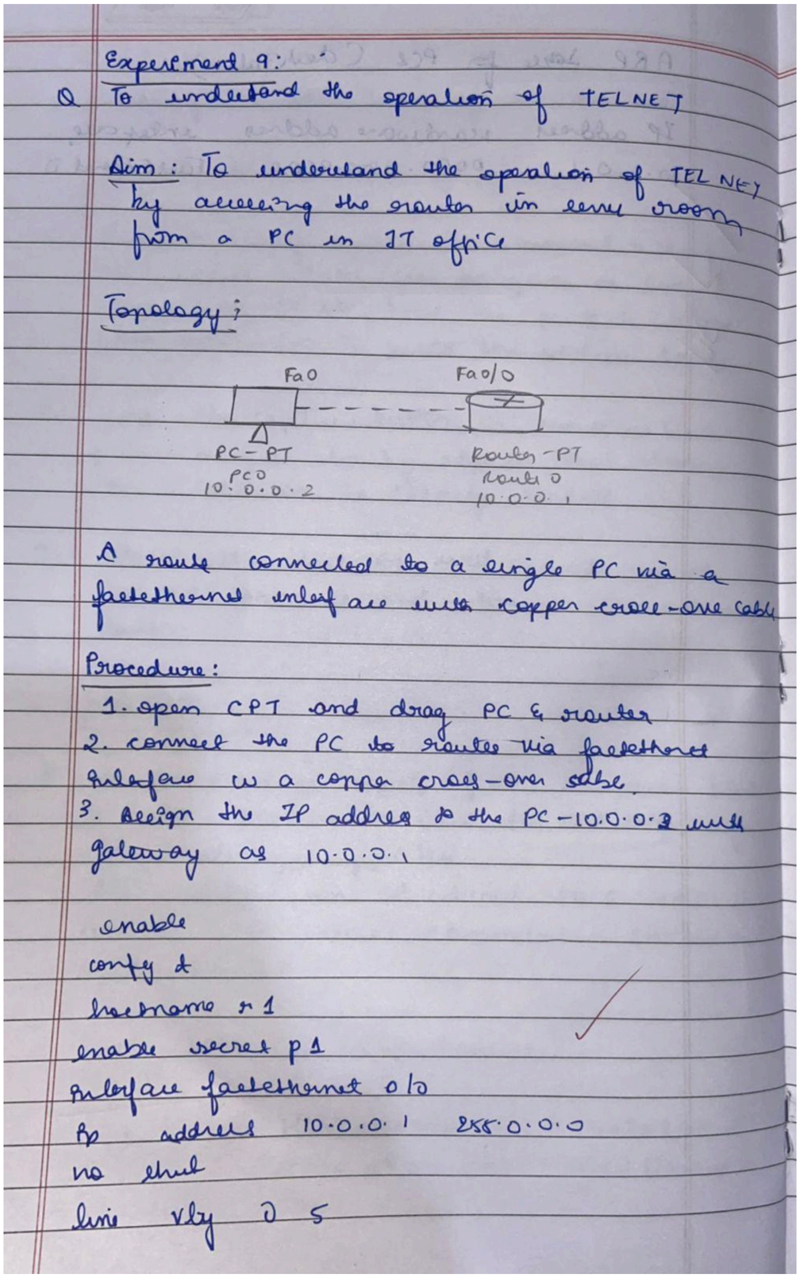

Switch>
Switch>show mac address-table
Mac Address Table

Vlan Mac Address Type Ports
---- ----------- -------- -----
1 0009.7c3c.0719 DYNAMIC Fa2/1
1 000c.cfd7.6dc7 DYNAMIC Fa3/1
1 0090.2b9d.194b DYNAMIC Fa0/1
1 00d0.d33c.c6ae DYNAMIC Fa1/1
Switch>
Copy
Paste

EXPERIMENT-10

Aim: To understand the operation of TELNET by accessing the router in the server room from a PC in the IT office.

Topology , Procedure and Observation:



Experiment 9:

Q To understand the operation of TELNET

Aim: To understand the operation of TELNET by accessing the router in server room from a PC in IT office

Topology:

A router connected to a single PC via a fastethernet interface with copper cross-over cable

Procedure:

1. open CPT and drag PC & router
2. connect the PC to router via fastethernet interface w a copper cross-over cable.
3. assign the IP address to the PC - 10.0.0.2 with gateway as 10.0.0.1

enable
config t
hostname r1
enable secret p1
interface fastethernet 0/0
ip address 10.0.0.1 255.0.0.0
no shut
line vty 0 5

login
password p0
exit
exit
exit.

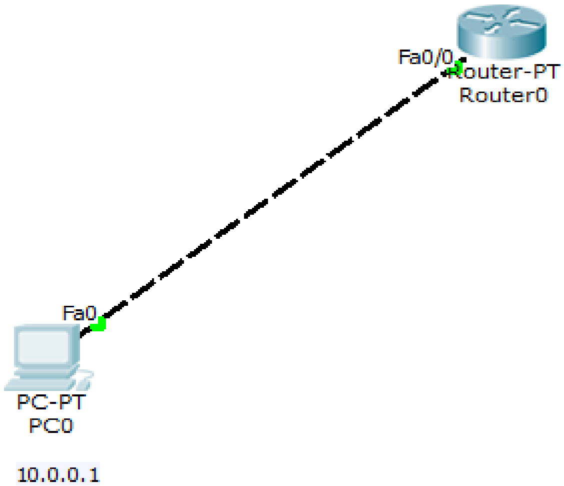
in cmd
ping 10.0.0.1
password for user authentication is p0
password for enable is p1

Observations:

Telnet is a protocol for remote access to servers. It allows cmd-line comm over a network

This pc is able to send the data to the router and indicates that the gateway is available & connected

Screen Shots:



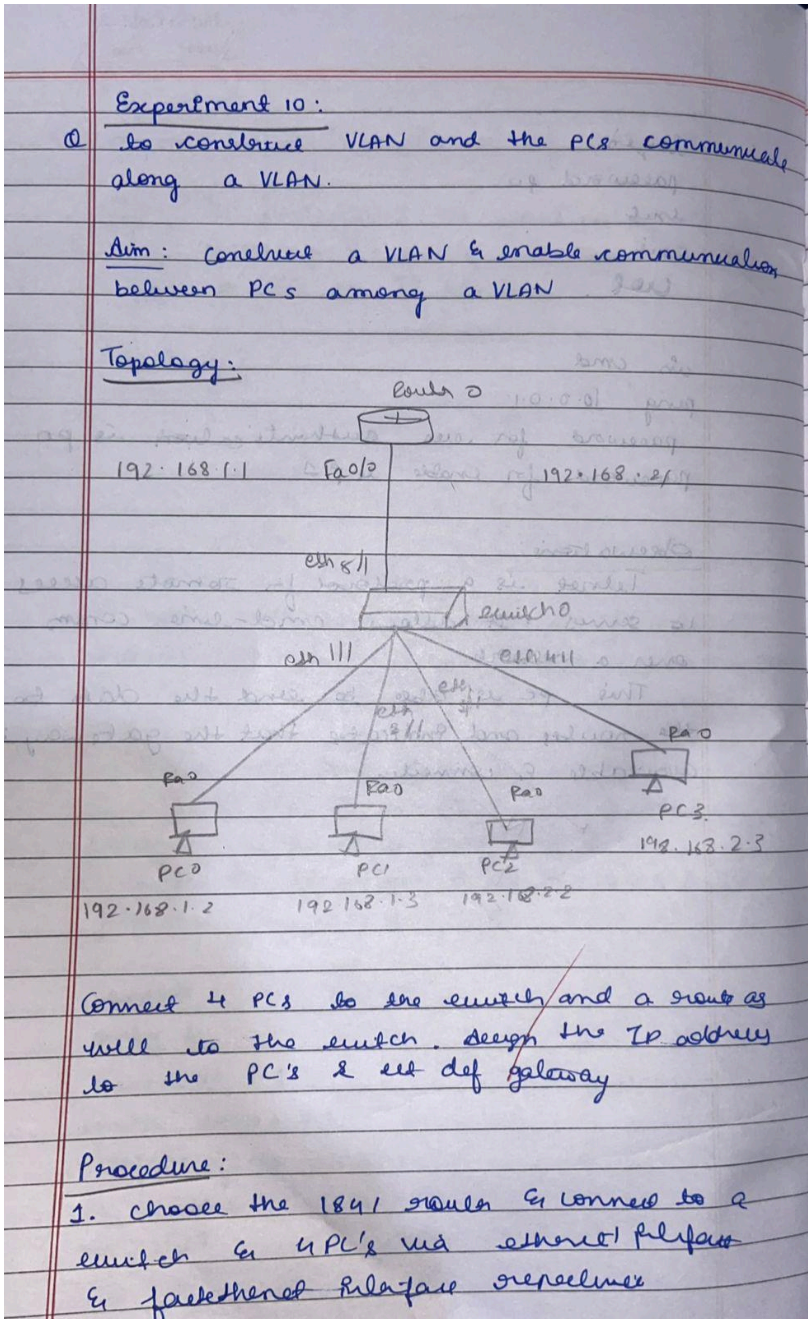
Command Prompt

```
Packet Tracer PC Command Line 1.0
PC>ping 10.0.0.2

Pinging 10.0.0.2 with 32 bytes of data:

Reply from 10.0.0.2: bytes=32 time=0ms TTL=255
Reply from 10.0.0.2: bytes=32 time=0ms TTL=255
Reply from 10.0.0.2: bytes=32 time=0ms TTL=255
Reply from 10.0.0.2: bytes=32 time=0ms TTL=255

Ping statistics for 10.0.0.2:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 0ms, Average = 0ms

PC>telnet 10.0.0.2
Trying 10.0.0.2 ...Open


User Access Verification

Password:
R1>enable
Password:
R1#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 not set

C    10.0.0.0/8 is directly connected, FastEthernet0/0
R1#
```

EXPERIMENT-11

Aim: To construct a VLAN and make the PC’s communicate among a VLAN .

Topology , Procedure and Observation:



Experiment 10:

Q to construct VLAN and the PCs communicate along a VLAN.

Aim: Construct a VLAN & enable communication between PCs among a VLAN

Topology:

Connect 4 PCs to the switch and a router as well to the switch. Assign the IP address to the PC's & set def gateway

Procedure:

1. Choose the 1841 router & connect to a switch & 4 PC's via ethernet interface & fastethernet interface respectively

Date: Page:

2. Set the IP address of the PCs & config the router 1 with IP address 192.168.1.1

Router > enable

\# config t

\# interface Fa 0/0

\# ip address 192.168.1.1 255.255.255.0

\# no shut

3. in the switch, go to config tab & select VLAN database

4. Set the VLAN number & VLAN name. Select the interface, i.e. fastethernet 5/1 & make it the trunk. VLAN trunking allows switches to forward frame from diff. VLAN over a single link called trunk

5. This is done by adding additional header info called tag to the ethernet frame

6. look into the interface of the switches with 2 new VLAN systems

Config tab of router select VLAN DATABASE enter number & name of VLAN created

Router (vlan) # exit

\# conf t

\# interface fastethernet 0/0.1 2.

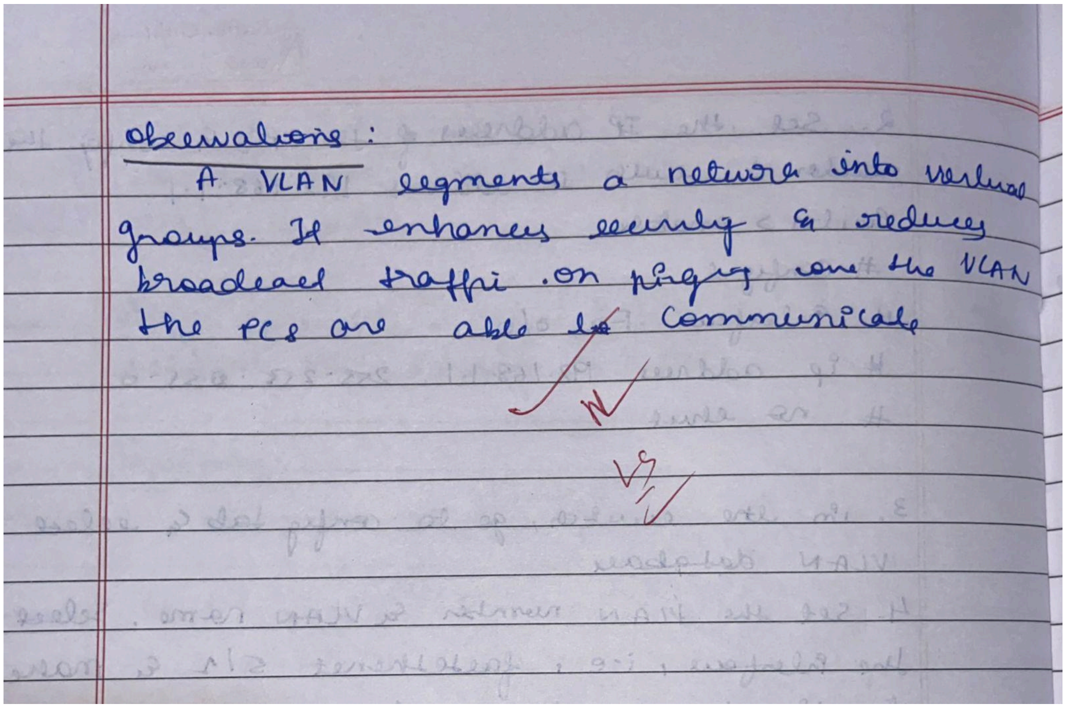
\# encapsulation dot1q

\# ip address 192.168.2.1 255.255.255.0

\# no shut

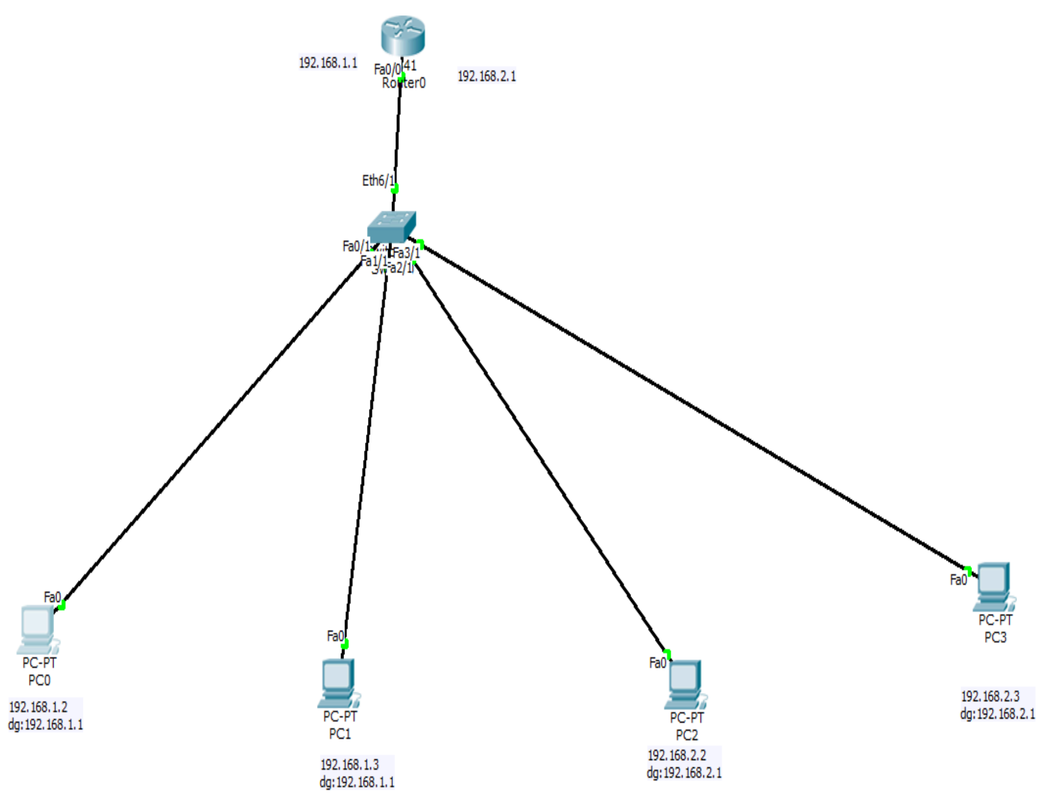
\# exit

\# exit



Observations:
A VLAN segments a network into virtual groups. It enhances security & reduces broadcast traffic. On pinging one the VLAN the PCs are able to communicate

Screen Shots:



Command Prompt

```
Packet Tracer PC Command Line 1.0
PC>ping 192.168.2.2

Pinging 192.168.2.2 with 32 bytes of data:

Request timed out.
Reply from 192.168.2.2: bytes=32 time=0ms TTL=127
Reply from 192.168.2.2: bytes=32 time=0ms TTL=127
Reply from 192.168.2.2: bytes=32 time=4ms TTL=127

Ping statistics for 192.168.2.2:
    Packets: Sent = 4, Received = 3, Lost = 1 (25% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 4ms, Average = 1ms

PC>ping 192.168.2.2

Pinging 192.168.2.2 with 32 bytes of data:

Reply from 192.168.2.2: bytes=32 time=0ms TTL=127
Reply from 192.168.2.2: bytes=32 time=0ms TTL=127
Reply from 192.168.2.2: bytes=32 time=2ms TTL=127
Reply from 192.168.2.2: bytes=32 time=0ms TTL=127

Ping statistics for 192.168.2.2:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 2ms, Average = 0ms

PC>ping 192.168.2.3

Pinging 192.168.2.3 with 32 bytes of data:

Request timed out.
Reply from 192.168.2.3: bytes=32 time=3ms TTL=127
Reply from 192.168.2.3: bytes=32 time=2ms TTL=127
Reply from 192.168.2.3: bytes=32 time=1ms TTL=127

Ping statistics for 192.168.2.3:
    Packets: Sent = 4, Received = 3, Lost = 1 (25% loss),
Approximate round trip times in milli-seconds:
    Minimum = 1ms, Maximum = 3ms, Average = 2ms

PC>ping 192.168.2.3

Pinging 192.168.2.3 with 32 bytes of data:

Reply from 192.168.2.3: bytes=32 time=0ms TTL=127
Reply from 192.168.2.3: bytes=32 time=0ms TTL=127
Reply from 192.168.2.3: bytes=32 time=2ms TTL=127
Reply from 192.168.2.3: bytes=32 time=0ms TTL=127

Ping statistics for 192.168.2.3:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 2ms, Average = 0ms

PC>
```

EXPERIMENT-12

Aim: To construct a WLAN and make the nodes communicate wirelessly.

Topology , Procedure and Observation:

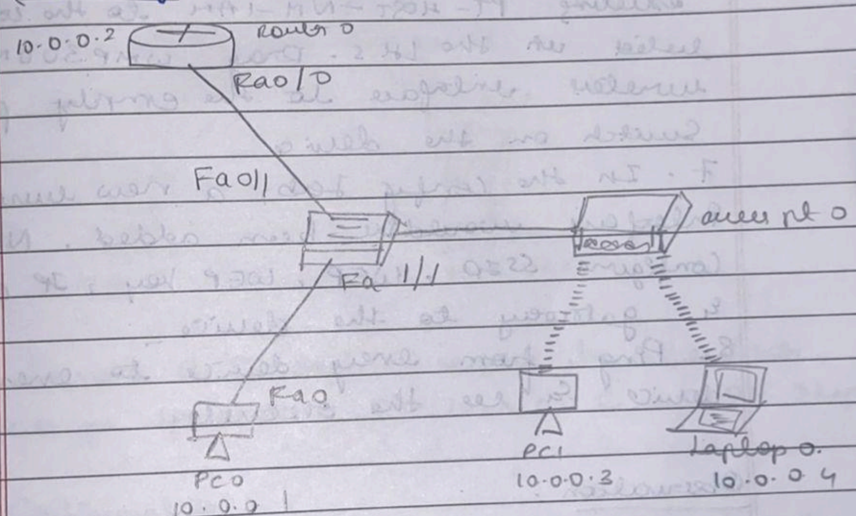
Date: Page:

Experiment 11 :

2. Construct a WLAN and make the nodes communicate wirelessly.

Aim : to construct WLAN and make nodes communicate wirelessly.

Topology :



Connect a router and access point to a switch through fast ethernet interface. connect a PC and set its ip address. Take a PC & a laptop & set either ip addresses

Procedure :

1. Drag a switch & connect it to a PC, router and an access point
2. Place a PC & laptop into any wireless connection

3. Configure PC0 w ip address 10.0.0.1 & router 0

4. Configure access point:

port 1 → SSID name → enter any name → select WEP & give any 10 digit hex key — 123456789D

5. Configure PC4 & laptop w wireless stds

6. Switch off the device. Drag the existing PT-HOST-NM-1AM to the component listed in the LHS. Drag WMP300N wireless interface to the empty port. Switch on the device

7. In the config tab, a new wireless interface would be been added. Now, configure SSID, WEP, WEP key, IP address & gateway to the device.

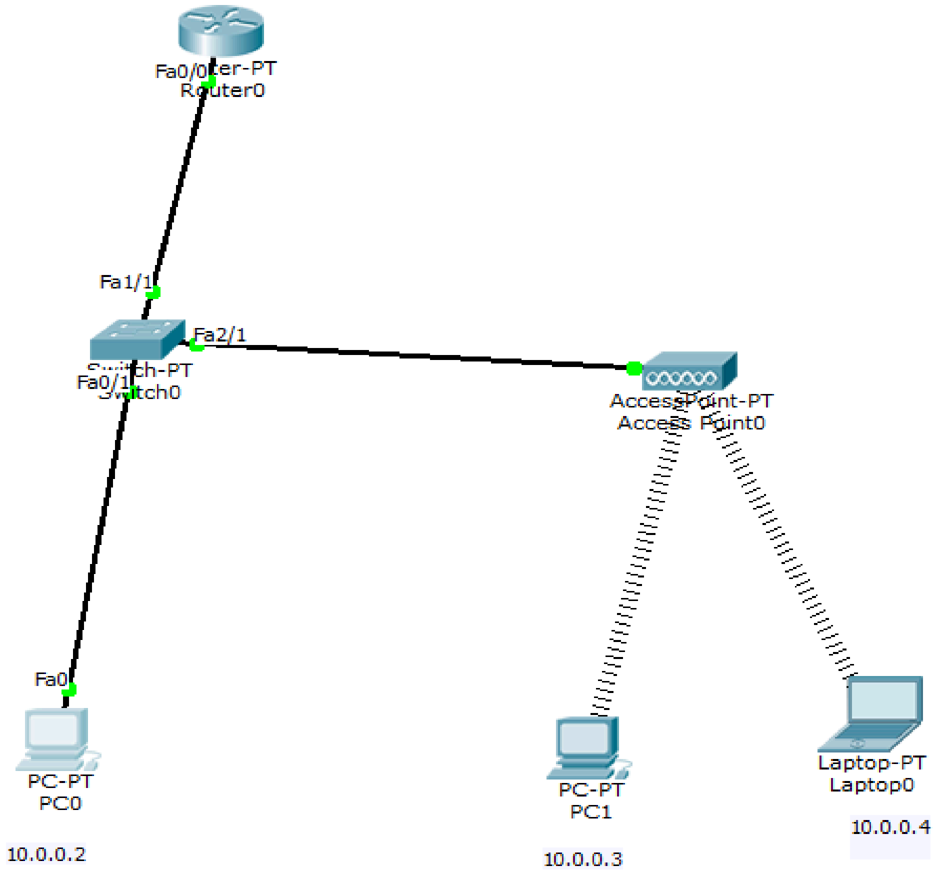
8. Ping from every device to every other device & see the results

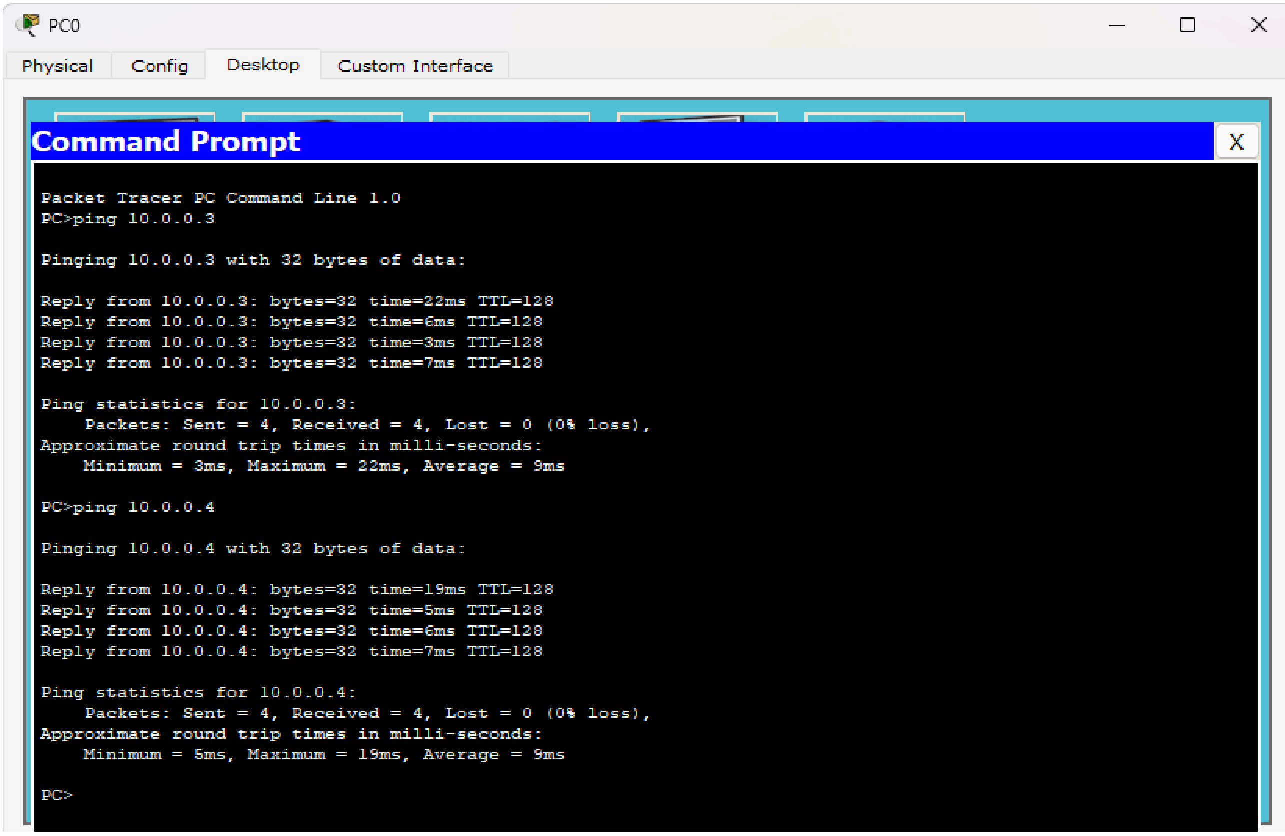
Observation:

WLAN enables wireless n/w comm. It uses radio waves for connectivity. WLAN connects devices wirelessly within a local area. It eliminates the need for physical cables.

3/4/25

Screen Shots:




PC0
Physical
Config
Desktop
Custom Interface
Command Prompt
X
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=22ms TTL=128
Reply from 10.0.0.3: bytes=32 time=6ms TTL=128
Reply from 10.0.0.3: bytes=32 time=3ms TTL=128
Reply from 10.0.0.3: bytes=32 time=7ms 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 = 3ms, Maximum = 22ms, Average = 9ms
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=19ms TTL=128
Reply from 10.0.0.4: bytes=32 time=5ms TTL=128
Reply from 10.0.0.4: bytes=32 time=6ms TTL=128
Reply from 10.0.0.4: bytes=32 time=7ms 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 = 5ms, Maximum = 19ms, Average = 9ms
PC>

CYCLE-2

EXPERIMENT-13

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

Code and Output:

Codes:

cycle-2.

Q2) Experiment-13.

Q2) i) write a program for error detection code using CRC-CCITT (16 bits)

code:
```
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 temp[0] == '1':
            tmp = xor(divisor, tmp) +
                dividend[pick]
        else:
            tmp = xor('0' pick, tmp)
                + dividend[pick)
        pick += 1
    if tem[0] == '1':
        tmp = xor(divisor, tmp)
    else
        tmp = xor('0' * pick, tmp)
    checkword = tmp
    return checkword
```

Bafna

Date:

```
def encode ( data , key ):
    l.key = len (key)
    appended - data = data + '0' + ( l.key - 1)
remainder - mod 2 div (append - data , key)
  codeward = data + remainder
    print (" Remainder :", remainder)
print (" encoded data [data + remainder codeword
return codeword

def decode data (encoded . data , key)
remainder = mod 2 div (encoded - data , key)
print (" Remainder after decoding : ", remainder )
  if '1' not in remainder
   print (" No error detected in received data" )
else print (" Error " detected in received data")

data = " 100100 1000 100100"
key = " 1101"
encoded - data = encodedData ( data , key )
decoded - data = encodeData ( data )
```

output :

Remainder = 11

encoded - Data (data + Remainder) =

100100 1000 100 10011

Remainder after decoding = 000

Received data

no error detected in received data

3/1/23

code:

```
#include <stdio.h>
#include <string.h>
#define N strlen(gen_poly)

int data[28],check[28],gen_poly[10];
int data_len,i,j;

void XOR(){
    for(j = 0; j<N;j++){
        if(check[j] == gen_poly[j]){
            check[j] = '0';
        } else{
            check[j] = '1';
        }
    }
}

void crc(){
    for(i = 0; i<N;i++){
        check[i] = data[i];
    }
    do{
        if(check[j] == '1'){
            XOR();
        }
        for(j = 0; j<N-1; i++){
            check[j] = check[j+1];
        }
        check[j] = data[i++];
    }
    while(i <= data_len + N - 1);
}

void reciever(){
    printf("\nData recieved: ");
    scanf("%s",data);

    crc();
    for(i = 0; i<N-1;i++){
        if (check[i] == '1'){
            break;
        }
    if (i < N-1){
        printf("\nERROR!");
        }else{
            printf("\nNO ERROR!");
        }
```

```
    }
}

int main(){
    printf("\nEnter data:");
    scanf("%s",data);
    printf("\nEnter generator:");
    scanf("%s",gen_poly);
    data_len = strlen(data);
    for(i = 0; i<data_len+N-1;i++){
        data[i] = '0';
    }
    printf("\nData with padded 0's: %s",data);
    crc();
    printf("\nCheck sum: %s",check);

    for(i = data_len; i<data_len+N-1; i++){
        data[i] = check[i-data_len];
    }
    reciever();
}
```

output:

```
Output                                                    Clear
Enter the data bits: 100100100100100
Enter the key (divisor): 10101
Encoded Data: 1001001001001000001

Decoding the encoded data...
Remainder after decoding: 0000
No error detected in received data

=== Code Execution Successful ===
```

EXPERIMENT-14

Write a program for congestion control using Leaky bucket algorithm.

Code and Output:

Experiment -4

a) ii) write a program for congestion control using leaky bucket algorithm

code:

```
#include <stdio.h>
# include <stdlib.h>
# include <unistd.h>
# define NOF_PACKETS 5
/*
int rand (int a)
{
    int rn = (random () % 10) % a;
    return rn == 0 ? 1 : rn
}
*/
/*
# include <stdlib.h>
long int random (void);
```

The random() function uses a nonlinear additive feedback random no generator employing a default table of size 31 long integers to return successive pseudo-random number in the range from 0 to RAND_MAX.

The period of this random number generator is very large, approx
16 * ((2 ^ 31) - 1).

```
int main()
{ int packet_sz[NOF PACKETS], i, clk, b_size,
o_rate, p_sz_rm = 0, p_sz, p_time, op;
for (int i = 0; i < NOF_PACKETS; ++i)
  packet_sz[i] = random() % 100;
```

Date: Page:

```
for (int i=0; i<NOF-PACKETS; ++i)
printf ("\n packets [%d]: %d bytes \t", i,
packet_sz [i]);
printf ("\n enter the output rate :");
scanf ("%d", &o_rate);
printf ("enter the bucket_size :");
scanf ("%d", &b_size);
for (i=0; i<NOF-PACKETS; ++i)
{
if (packet_sz [i] + p_sz_rm) > b_size)
if (packet_sz [i] > b_size)
printf ("\n\n Incoming packet size (%d bytes)
is greater than bucket capacity (%d bytes)
PACKET REJECTED", packet_sz [i], b_size);
else
printf ("\n\n bucket capacity exceeded. PACKET
REJECTED!");
else
{ p_sz_rm += packet_sz [i];
printf ("\n\n Incoming packet size : %d", packet_sz[i]);
printf ("\n Bytes remaining to transmit %d", p_sz_rm);
//p_time = random () * 10;
//printf ("\n time left for transmission: %d units", p_time);
//for (clk = 10 ; clk <= p_time ; clk += 10)
while (p_sz_rm > 0)
{ sleep (1);
if (p_sz_rm <= o_rate)
op = p_sz_rm, p_sz_rm = 0,
else
op = o_rate, p_sz_rm -= o_rate;
printf ("\n Packet of size %d transmitted", op);
printf ("___ bytes remaining to transmit: %d,
p_sz_rm);
}.
```

else
{
 printf ("\n no packets to transmit !! \n");
 }
}
}
}
}

output:
packet [0] : 83 bytes
packet [1] : 86 bytes
packet [2] : 77 bytes
packet [3] : 15 bytes
packet [4] : 93 bytes
enter the output rate = 30
enter the bucket size = 35

incoming packet size : 83
bytes remaining for transmit: 83
packets of size 30 transmitted....
bytes remaining to transmit : 53
packets of size 30 transmitted...
Bytes remaining to transmit : 23
packets of size 23 transmitted...
bytes remaining to transmit = 0

3/1/25

code:

```
capacity = 10
input_size = 4
out_size = 1
storage = 0
iterations = 4

for i in range(0,iterations):
    size_left = capacity - storage
    if input_size <= size_left:
        storage += input_size
    else:
        print("Packet lost: ",input_size)
    print(f"Buffer size = {storage} out of capacity {capacity}")
    storage -= 1
```

output:

Output Clear

```
packet[0]:83 bytes
packet[1]:86 bytes
packet[2]:77 bytes
packet[3]:15 bytes
packet[4]:93 bytes
Enter the Output rate:30
Enter the Bucket Size:85

Incoming Packet size: 83
Bytes remaining to Transmit: 83
Packet of size 30 Transmitted---Bytes Remaining to Transmit: 53
Packet of size 30 Transmitted---Bytes Remaining to Transmit: 23
Packet of size 23 Transmitted---Bytes Remaining to Transmit: 0

Incoming Packet size: 86
Incoming packet size (86bytes) is Greater than bucket capacity (85bytes)-PACKET REJECTED

Incoming Packet size: 77
Bytes remaining to Transmit: 77
Packet of size 30 Transmitted---Bytes Remaining to Transmit: 47
Packet of size 30 Transmitted---Bytes Remaining to Transmit: 17
Packet of size 17 Transmitted---Bytes Remaining to Transmit: 0

Incoming Packet size: 15
Bytes remaining to Transmit: 15
Packet of size 15 Transmitted---Bytes Remaining to Transmit: 0

Incoming Packet size: 93
Incoming packet size (93bytes) is Greater than bucket capacity (85bytes)-PACKET REJECTED
```

EXPERIMENT-15

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

Code and Output:

Date: Page:

Experiment-15

Q) using TCP/IP socket, 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

<u>clientTCP.py</u>

```
from socket import*
Server Name = '127.0.0.1'
Server Port = 12000
client socket = socket (AF_INET, SOCK_STREAM)
client socket = connect ((servername, server port))
sentence = input ("\n enter file name:")
client socket . send ( sentence . encode ())
filecontent = client socket. recv (1024). decode ()
print (filecontents)
client socket . close ()
```

<u>ServerTCP.py</u>

```
from socket import*
Server Name = "127.0.0.1"
server port = 12000
Server socket = socket (AF_INET, sock_STREAM)
Server socket. bind ((server name, server port))
server Socket. listen (1)
while 1:
    print ("The server is ready to receive")
connection socket, addr = server Socket. accept ()
sentence = connection socket. recv (1024). decode (
file = open ( sentence , "r")
file = open ( sentence , "r")
```

```
l = file . read (1024)
connection socket . send ( l.encode ())
print ('\n socket contents of ' + sentence)
file . close ()
connection socket . close ()
```

TCP. py [client]

```
from socket import *
Server Name = '127.0.0.1'
Server port = 12000
Server socket = socket (AF_INET , SOCK_STREAM)
server socket . bind ((Server Name , Serverport))
Server Socket . listen (1)
while 1:
    print (" The server is ready to receive ")
    connectionSocket, addr = Server Socket . accept ()
    sentence = connection Socket . recv(1024). decode ()
    file = open (sentence , "r")
    l = file. read (1024)
    connection Socket . send ( l. encode ())
    print ('\n sent contents of ' + sentence ) file
    file. close ()
    connection socket . close ()
```

TCP. py [server side]

The server is ready to receive

3/1/25

code:

clientTCP.py

```
from socket import *

serverIP = '127.0.0.1'
serverPort = 12000
clientSocket = socket(AF_INET,SOCK_STREAM)
clientSocket.connect((serverIP,serverPort))
sentence = input("File name:")
clientSocket.send(sentence.encode())
filecontents = clientSocket.recv(1024).decode()
print(filecontents)
clientSocket.close()
```

serverTCP.py

```
from socket import *

serverIP = '127.0.0.1'
serverPort = 12000
serverSocket = socket(AF_INET,SOCK_STREAM)
serverSocket.bind((serverIP,serverPort))
serverSocket.listen(1)

while(1):
    print("Server Ready!")
    connectionSocket, clientAddr = serverSocket.accept()
    sentence = connectionSocket.recv(1024).decode()
    file = open(sentence,'r')
    contents = file.read(1024)
    connectionSocket.send(contents.encode())
    file.close()
    serverSocket.close()
```

output:

```
PS C:\Users\I AM HP\CN> python ClientTCP.py

 Enter file name :ServerTCP.py

 From Server:

from socket import *
serverName="127.0.0.1"
serverPort=12000
serverSocket=socket(AF_INET,SOCK_STREAM)
serverSocket.bind((serverName,serverPort))
serverSocket.listen(1)
while 1:
    print("The server is ready to receive")
    connectionSocket,addr=serverSocket.accept()
    sentence=connectionSocket.recv(1024).decode()
    file=open(sentence,"r")
    l=file.read(1024)
    connectionSocket.send(l.encode())
    print("\n Sent contents of"+sentence)
    file.close()
    connectionSocket.close()
PS C:\Users\I AM HP\CN>
```

EXPERIMENT-16

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

Code and Output:

Data: Page:

experiment -16.

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

client UDP.py

```
from socket import *
Server name = "127.0.0.1"
Client Socket = Socket (AF_INET, SOCK_DGRAM)

sentence = input ("\n enter file name: ")
clientSocket. Sendto (bytes (sentence, "utf-8"),
        (serverName, server Port))
filecontents. serverAddress = client Socket. recvfrom (2048)
print (\n Reply from Server: \n)
print (file contents. decode ("utf-8"))
# for i in file contents:
# print (str(i), end = "")
client Socket . close ()
client Socket. close ()
```

Server UDP.py

```
from socket import *
Server Port = 12000
Server Socket = Socket (AF_INET, SOCK_DGRAM)
Server Socket. bind ("127.0.0.1", Server Port))
print ("The server is ready to receive")

while 1:
sentence, client Address = Server socket. recvfrom (2048)
sentence = sentence. decode ("UTF-8")
```

file = open (sentence . "r")
Con= file . read (2048)

Server Socket. sendto (bytes (con, "utf - 8"),
client Address)
print (" In sent Contents of ', end = '')
print (sentence)
for i in sentence :
print (str (i), end= '' ")
file . close ()

Server UDP. py
The server is ready to receive
sent contents of Server UDP. py
The Server is ready to receive

Client UDP. py
Enter file name : Server UDP. py
Reply from Server :
from socket import *
Server Port = 12000
Server Socket = Socket (AF_INET, SOCK_DGRAM)
Server Socket bind (("127-0.0.1", Server Port))

while 1 :
print (" The Server is ready to receive")
sentence, clientAddress = Server Socket. recvfrom (20
Sentence = sentence . decode (" UTF - 8")
file = open (sentence , "r")
'l' = file . read (2048)
server . socket . send to (bytes (l, "utf -8'), client ad
print (' In sent contents of ', end = '')
for i in sentence :
print (str(i), end = '')
file . close ()

73

CODE:

clientUPD.py

```
from socket import *

serverIP = '127.0.0.1'
serverPort = 12000
clientSocket = socket(AF_INET,SOCK_DGRAM)
sentence = input("File name")
clientSocket.sendto(bytes(sentence,'utf-8'),(serverIP,serverPort))
contents,serverAddress = clientSocket.recvfrom(2048)
print(contents.decode())
clientSocket.close()
```

serverUDP.py

```
from socket import *

serverIP = '127.0.0.1'
serverPort = 12000
serverSocket = socket(AF_INET,SOCK_DGRAM)
serverSocket.bind((serverIP,serverPort))

while(1):
   print("READY")
   sentence,clientAdd =  serverSocket.recvfrom(2048)
   sentence = sentence.decode()
   file = open(sentence,'r')
   cont = file.read(2048)
   serverSocket.sendto(bytes(cont,'utf-8'),clientAdd)
   file.close()
   serverSocket.close()
```

output:

```
DEBUG CONSOLE    PROBLEMS    TERMINAL    PORTS

PS C:\Users\I AM HP\CN> python ServerUDP.py
The server is ready to receive

 Sent contents of ServerUDP.py
The server is ready to receive
```

```
PS C:\Users\I AM HP\CN> python 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))
while 1:
    print("The server is ready to receive")
    sentence,clientAddress=serverSocket.recvfrom(2048)
    sentence=sentence.decode("utf-8")
    file=open(sentence,"r")
    con=file.read(2048)
    serverSocket.sendto(bytes(con,"utf-8"),clientAddre
ss)
    print("\n Sent contents of "+sentence)
    file.close()
PS C:\Users\I AM HP\CN>
```

EXPERIMENT-17

Tool exploration- wireshark

Experiement - 17

Q) Tool exploration - wireshark

Wireshark is a powerful and widely used network protocol analyzer. It allows you to capture and inspect data packets travelling over a network in real-time, making it a crucial tool for studying computer networks, troubleshooting network issues, and understanding protocols

Key features:

1. packet capture : captures live network traffic from various interfaces (eg. ethernet, Wi-Fi)
2. Protocol Analysis : supports hundreds of protocol (eg : TCP, UDP, HTTP, FTP)
3. filtering : offers powerful filters to isolate specific packets or traffic types
4. Visualization : displays packets details with hierarchical layers (ethernet, IP, TCP / UDP)

Use cases of wireshark :

1. network troubleshooting :
   - diagnosing slow network speeds
   - Identifying bottle necks or misconfigurations
2. Security analysis :
   - detecting malicious traffic or intrusions
3. Protocol Study :
   - understanding packets structures and communication flow.

Common Filters:

- http : show only HTTP traffic
- tcp.port ==80 : show traffic on TCP port 80
- ip.addr == 192.168.1.1 : show packets to or from a specific IP address
- udp : show only UDP traffic

3/1/25