

My First Packet Tracer Lab :

Packet tracer is a cross-platform visual simulation tool designed by Cisco Systems that allows users to create network topologies and simulate modern computer networks.

The software allows users to create network topologies and simulate the configuration of Cisco routers and switches using a simulated command line interface.

Steps to create a Network :-

- ① Usually, we are supposed to do it in Logical with Realtime and not in Physical, simulation.
- ② Start creating a network by first selecting "end devices" which is located in the left most bottom of the screen.
- ③ Add a "generic PC" (1st option in beside row) and add another "generic PC". (Press on icon - drag - place)
- ④ Under "connections" located in the bottom left corner (zigzag icon) select "Automatically choose connection type" (default) and connect the devices.
- ⑤ Click on the end device (2 devices connected with wire) choose "config" then click "Fast Ethernet" then enter "IP address" (e.g.: 10.0.0.1) then click on "subnet mask" automatically the value occurs as (255.0.0.0) then exit. Do the same thing for the first generic device with diff. "IP address" (e.g.: 10.0.0.2) then exit. Now click on "simple PDU" located right side (envelope icon) place on both the devices. In the bottom right it shows as successfull.
- ⑥ Now, click on the <sup>left</sup> end generic device - go to "Desktop" click "Command Prompt" - type "ping" space IP address of the generic device which we entered want to send and press enter. (e.g.: 10.0.0.1)

→ HUB: • At least single network is required to connect.  
• It's a physical layer device i.e; layer 1.  
• works on the basis of broadcasting.  
• multipoint repeater in which a signal introduced at the I/P of any port appears at the O/P of all available ports.

→ SWITCH: • At least single network is required to connect.

• It's a Data link Layer device i.e; layer 2.  
• works on the basis of MAC address.  
• It's a telecommunication device which receives message from any device connected to it and transmits message only to device for which message is intended.

→ ROUTER: • At least 2 networks are required to connect.

• Router is a network layer device i.e; layer 3.  
• works on basis of IP address.  
• reads the header of incoming packet & forward it to port & which its intended there by determining the route.

10/11/22 . LAB-2

CYCLE - 1

**EXPERIMENT - 1**

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

Q) Using hub as connecting device.

Hub is an un-intelligent device. It operates in physical layer. No signal processing / regeneration occurs.

Procedure:

- We open Cisco packet tracer in logical mode. At the left hand side button corner we select end devices from device-type selection box.
- We select 4 generic end devices and enter the following IP addresses:- 10.0.0.1, 10.0.0.2, 10.0.0.3, 10.0.0.4. They have a common subnet mask of 255.0.0.0.
- We select a generic hub and make connections to the end devices using copper - straight - through connections. We add a PDV to source end device (IP: 10.0.0.1) & destination end device (IP: 10.0.0.4).
- We switch to simulation mode and select auto capture/play.
- Message moves from Device (10.0.0.1) to Hub.
- The Hub transmits the message to remaining devices.
- Only Device (10.0.0.4) receives it correctly!  
The other 2 devices reject it.

Event List:

Time	Lost Device	At device
0.000	---	PC10
0.001	PC0	Hub0
0.002	Hub0	PC1
0.002	Hub0	PC2
0.002	Hub0	PC3
0.003	PC3	Hub0

## Real Time (Event list)

First host address	source	Destination	Time (sec)	Periodic	Num
Successful	PC0	PC3	0.000	N	0

• Ping PC > Ping 10.0.0.4

Pinging 10.0.0.4 with 32 byte of data.

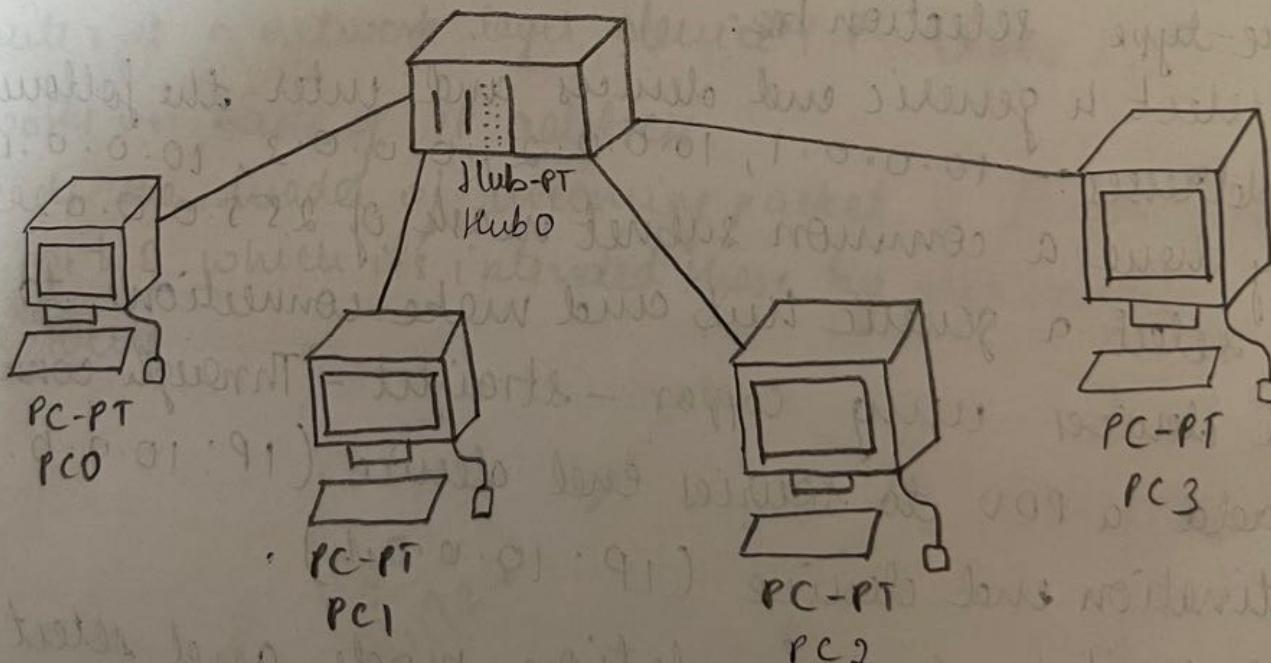
Reply from 10.0.0.4 ; bytes = 32 time = 0ms TTL = 128  
Statistics

Packet: sent = 1, receives = 1, lost = 0 (0% loss)

round trip times:

Minimum = 0ms, Maximum = 0ms, Average = 0ms

## TOPOLOGY:



## b) Using switch as connecting Device:

Switch is a point-to-point communication device. It operates at data link layer. It uses switching table to obtain correct address.

### Procedure :

- We select end devices from the Device-type selection box. We enter 10.0.0.5, 10.0.0.6, 10.0.0.7, 10.0.0.8 as their IP address respectively. They have common subnet mask 255.0.0.0.
- We select a generic switch and make connections to the end devices using copper-straight-through connect's.
- We add a PDU to source End device (IP: 10.0.0.5), and destination End device (IP: 10.0.0.8)
- We enter simulation mode and select auto capture/play.
- Message moves from end device (10.0.0.5) to switch.
- The switch upon receiving the message sends it to destination end device (10.0.0.8) without broadcasting the message to other devices. Point-to-point communication is present.
- Real time (Event list)

Time	lost status	source	Destinet <sup>n</sup>	Travel(sec)	Periodic	Num	edit
	successful	PC4	PC7	0.000	N	0	✓

### Simulation Model (Event list)

Time (sec)	lost Device	At Device
0.000	--	PC4
0.001	PC4	switch0
0.002	switch0	PC7
0.003	PC7	switch0
0.004	switch0	PC4

ping PC > ping 10.0.0.8

Ping to 10.0.0.8 with 32 bytes of data:

Reply from 10.0.0.8: bytes = 32 time = 11ms TTL = 128

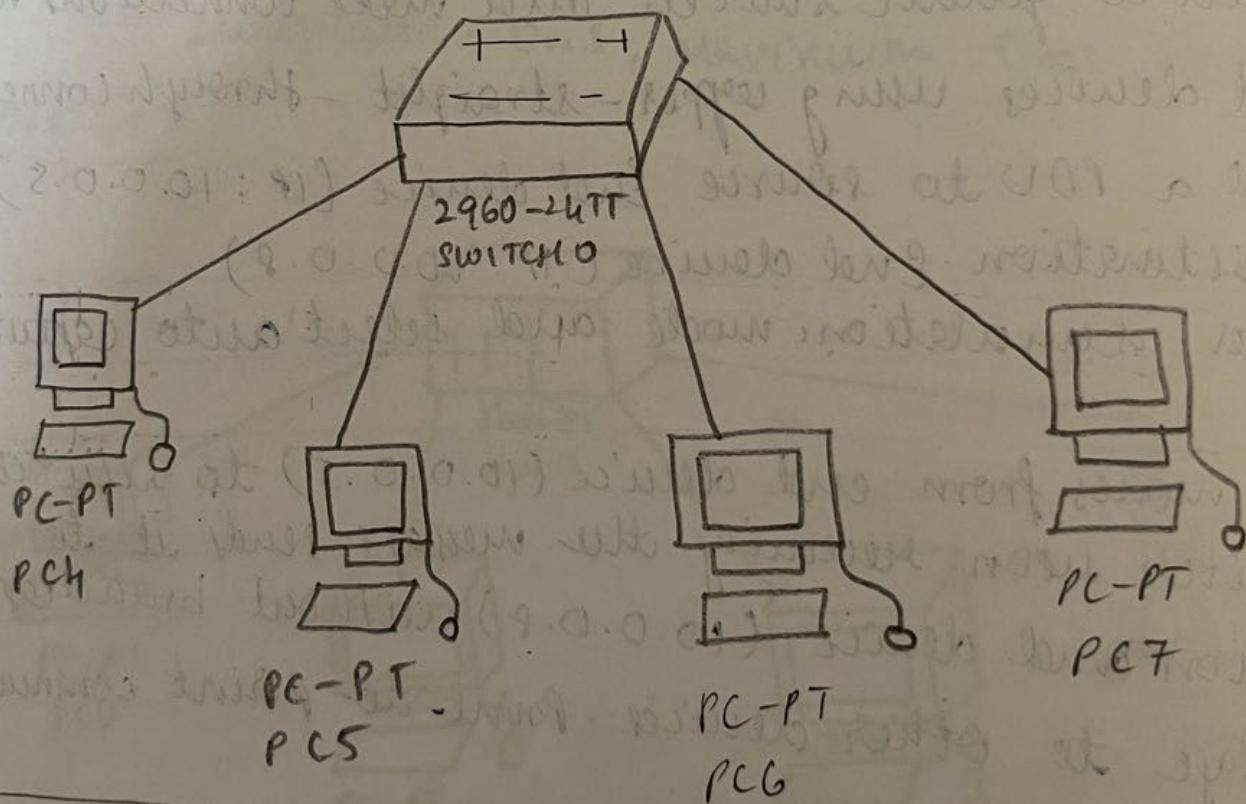
Statistics for 10.0.0.8

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

Approximate round trip times in milli-seconds:

Minimum = 0ms, Maximum = 11ms, Avg = 4 ms.

### Topology:



c) Using both hub and switch as connecting devices:

Procedure:

- We form an interconnected LAN by making a connection b/w Hub & switch established previously using a copper-cross-over connection.
- We add a PDU to end device-source (IP: 10.0.0.1) & to end device-destination (IP: 10.0.0.8).  
Source connected to Hub & Destination connected to switch.
- We enter simulation mode & select auto capture/ play.

real time (Event list)

Time	lost status	Source	Destination	Time(sec)	Periodic	Num
	successful	PC0	PC7	0.00	N	0

- Message moves from source end device (IP: 10.0.0.1) to Hub.
- Hub broadcasts the message to devices (IP: 10.0.0.2, IP: 10.0.0.3, IP: 10.0.0.4) and to the switch.
- The end devices reject message.
- The switch receives the message and sends it to destination end device (IP: 10.0.0.8) directly & not to any other device.
- In the next cycle, message sent from device (IP: 10.0.0.8) goes to switch and then to hub directly.
- Hub broadcasts it to devices (IP: 10.0.0.1, 10.0.0.2, 10.0.0.3, 10.0.0.4), source device (IP: 10.0.0.1) receives message.

## • Simulation Model (Event list)

Time (sec)	Lost Device	At Device
0.000	---	PC0 Hub0 Switch0 PC7 Switch0 Hub0
0.001	PC0	
0.002	Hub0	
0.003	Switch0	
0.004	PC7	
0.005	Switch0	
0.006	Hub0	
0.006	Hub0	PC4
0.006	Hub0	PC3
0.006	Hub0	PC2
0.006	Hub0	PC1

Ping PC > ping 10.0.0.8

Pinging 10.0.0.8 with 32 bytes of data.

Reply from 10.0.0.8 : bytes=32 time<1ms TTL=128

Reply from 10.0.0.8 : bytes=32 time<1ms TTL=128

Reply from 10.0.0.8 : bytes=32 time<1ms TTL=128

Reply from 10.0.0.8 : bytes=32 time=1ms TTL=128

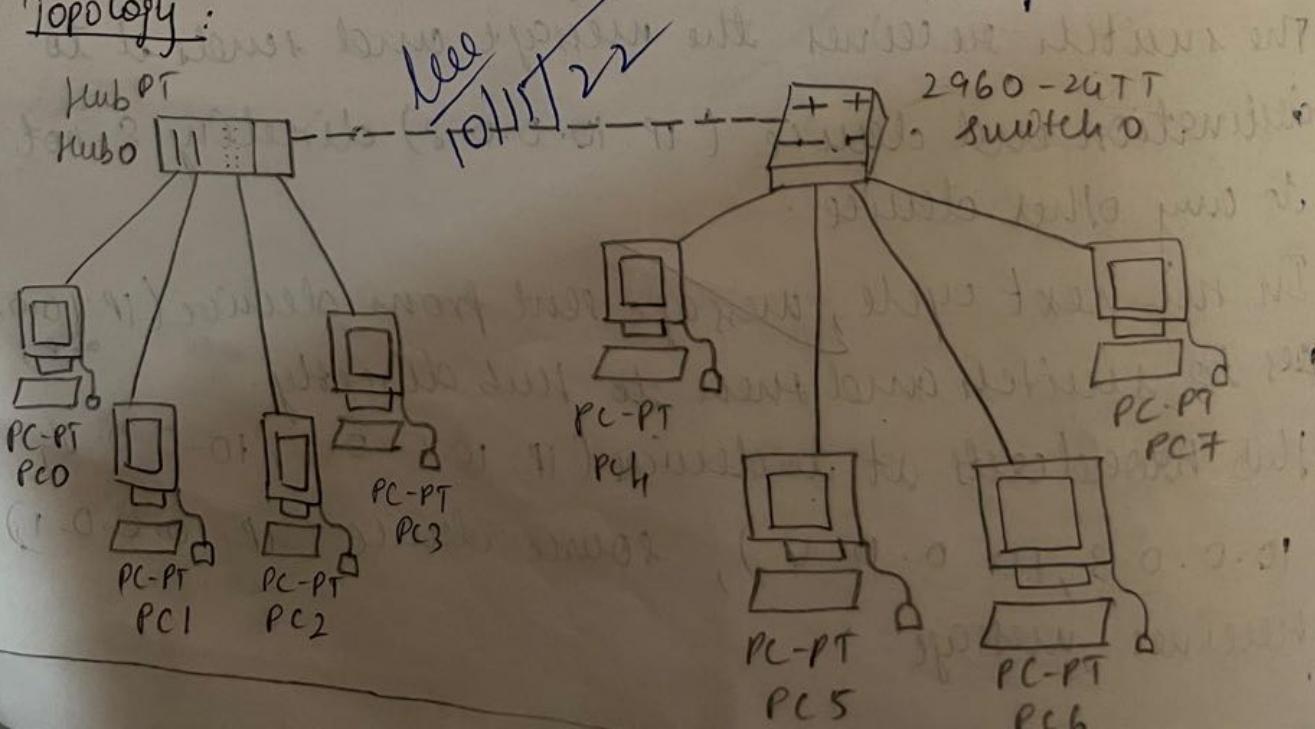
Ping statistics for 10.0.0.8

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

Approximate round trip times in (mili)-seconds

Minimum = 0ms, Maximum=1ms, Avg=0ms.

Topology:



17/11/22  
configuring  
following  
request

• Routers  
at N etr  
which P  
proceed

- We s
- selection
- at 10.0
- subnet
- We s
- end
- We s
- denot
- We S
- The
- on
- con

1<sup>st</sup> sl

Route

configuring IP address +2 routers in Pocket Tracer. Explore the following messages: Ping Response, Destination unreachable, Request timed out, Reply.

Routers are sophisticated multi-port devices. They operate at Network layer and use a routing table to determine which path from source to destination should be selected.

### Procedure:

- We select 2 generic end devices from the Device-type selection box. We give the source device IP address as 10.0.0.1 and 20.0.0.1 to the other end device subnet mask 255.0.0.0.
- We select a generic Router-PT and connect it to the end devices using copper cross-over connections.
- We see interface between end device and router denoted by a red dot (Network not yet functional).
- We have to configure the interfaces.
- The following commands are executed by clicking on the router and selecting CLI.
- Continue with configuration dialog? [yes/no]: no

### 1<sup>st</sup> side configuration:

```
Router > enable
Router# config terminal
Router(config)# interface fastEthernet 0/0
Router(config-if)# IP address 10.0.0.10 255.0.0.0
Router(config-if)# no shutdown
Router(config-if)# exit
```

### 2<sup>nd</sup> side configuration:

```
Router(config)# interface fastEthernet 0/0
Router(config-if)# IP address 20.0.0.10 255.0.0.0
Router(config-if)# no shutdown
Router(config)# exit
```

- the interfaces (represented by red dot) turn green. This indicates that the network is functional.

- We add PDU's to the end devices.
- We ~~for~~ select the source end devices and go to the command prompt option in Desktop panel.
- Enter command:

PC > ping 20.0.0.1

Pinging 20.0.0.1 with 32 bytes of data.

~~Ref Request timed out~~

Request timed out

Request timed out

Request timed out

- Gateway address has to be added for end devices to know where to send PDU when Router is present.
- For source end-device (IP: 10.0.0.1) enter gateway of interface IP address: 10.0.0.10.
- For classification end-device (IP: 20.0.0.2) enter gateway as interface IP address: 20.0.0.10.
- Select source end device and go to desktop panel, choose command prompt option.
- Enter the command:

PC > ping 20.0.0.1

Pinging 20.0.0.1 with 32 bytes of data.

Reply from 20.0.0.1: bytes = 32 time = 0ms TTL = 127

Reply from 20.0.0.1: bytes = 32 time = 0ms TTL = 127

Reply from 20.0.0.1: bytes = 32 time = 0ms TTL = 127

Reply from 20.0.0.1: bytes = 32 time = 0ms TTL = 127

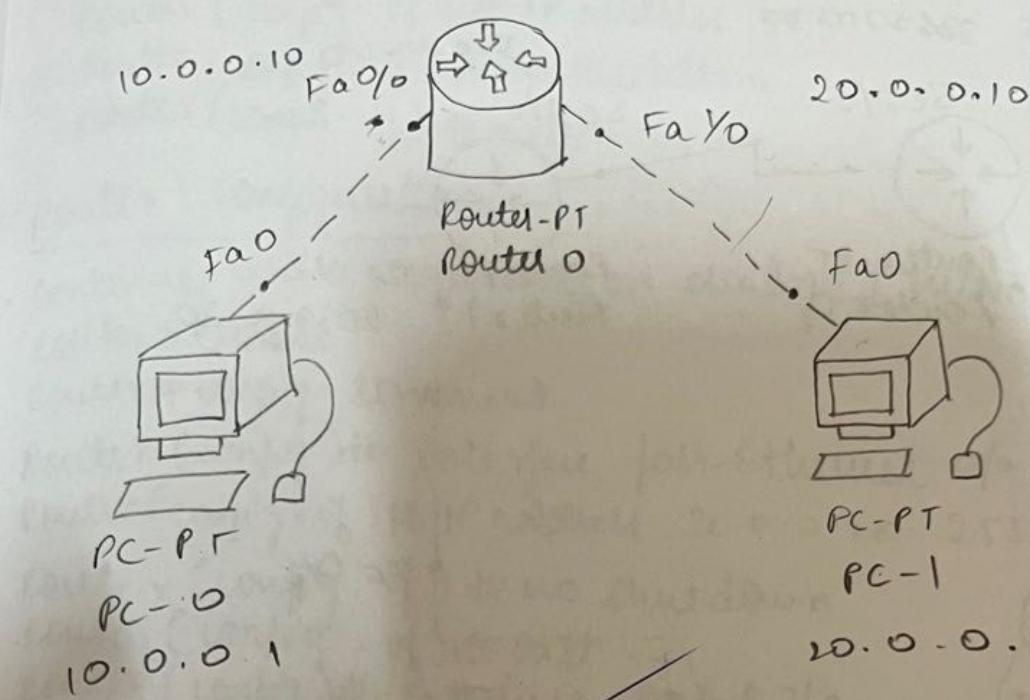
Ping statistics for 20.0.0.1

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

Approximate round trip times in milliseconds:

Minimum = 0ms, Maximum = 0ms, Average = 0ms.

## TOPOLOGY



(0.0.0.0) < 0.0.0.0 ping < 0.0.0.0  
widnows wan interface

: 0.0.0.0 ref method ping

(ref. 0.0) 0 = 100, 0 = leisure, n - task : 100%

(1 values at 0 rates) 0 < 0.0.0.0 ping

(ref. 1.0) 0 = 100, n = leisure, n - task value

no steaming 0 when

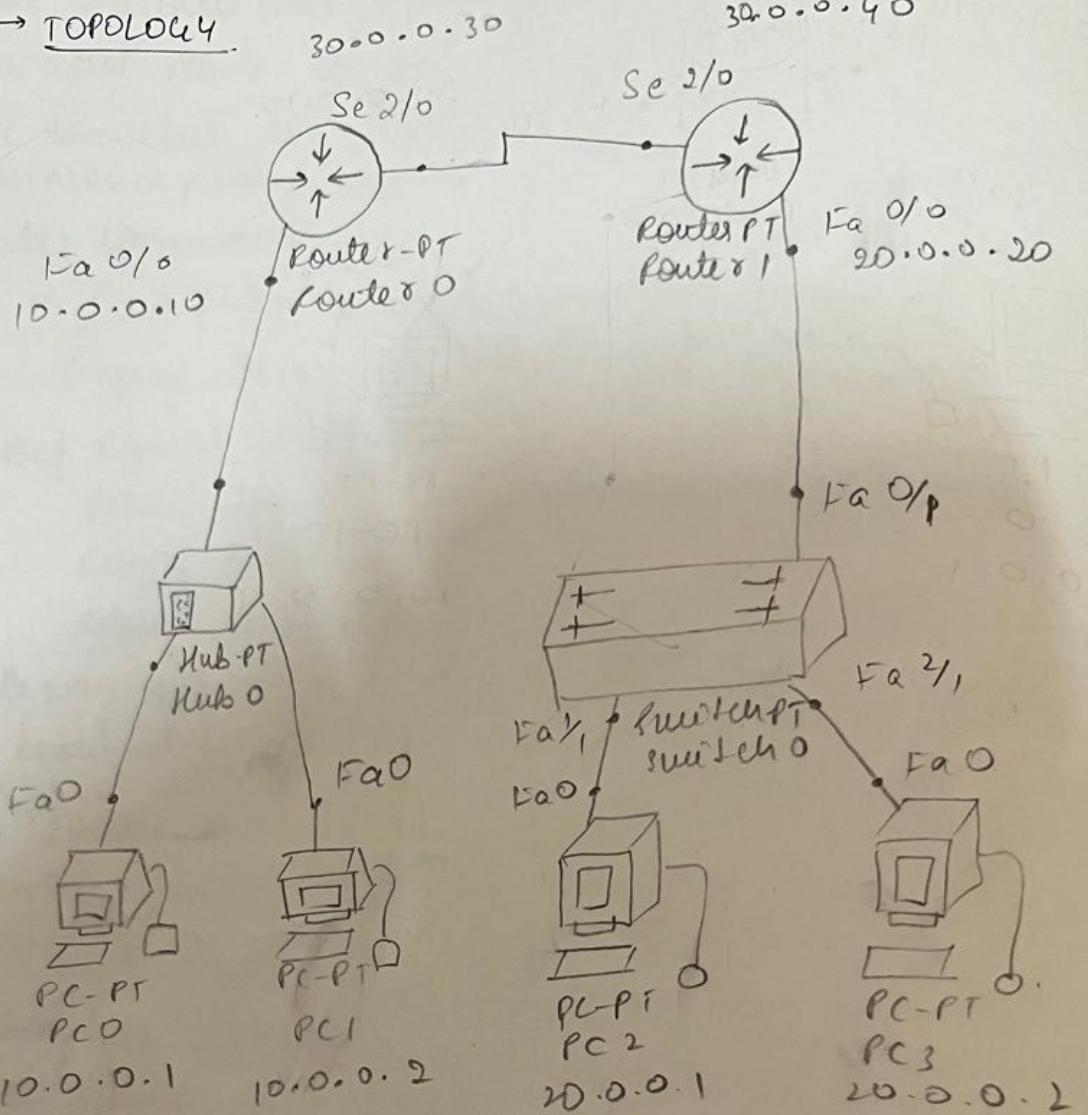
our count up ? packets retransmits this requires

when < (task)

lowest ping it related

## Part-2

→ TOPOLOGY



PC > ping 20.0.0.2 (from PC0)

Destination Not Unreachable.

Perig statistics for 20.0.0.2:

Metrics: Sent = 4, Received = 0, Cost = 4 (100% loss)

Ping 30.0.0.40 (from router 0 to router 1)

packets sent = 4, received = 1, lost = 0

Router 0 Configuration:

- continue with configuration dialog? [ Yes / No ] : no.  
Router > enable  
Router # config terminal.  
Router (config) # interface fastethernet 0/0.  
Router (config-if) # IP address 10.0.0.10 255.0.0.0  
Router (config-if) # no shutdown.

Router (config-if)# exit.  
Router (config)# interface serial 2/0.  
Router (config-if)# IP address 30.0.0.30 255.0.0.0.  
Router (config-if)# no shutdown  
Router (config-if)# exit.

### Router | configuration:

- continue with configuration dialog? [Yes/No]: No

Router > enable

Router# config terminal.

Router (config)# interface fastEthernet 0/0.

Router (config-if)# IP address 20.0.0.20 255.0.0.0

Router (config-if)# no shutdown

Router (config-if)# exit

Router (config)# interface serial 3/0.

Router (config-if)# IP address 30.0.0.40 255.0.0.0

Router (config-if)# no shutdown

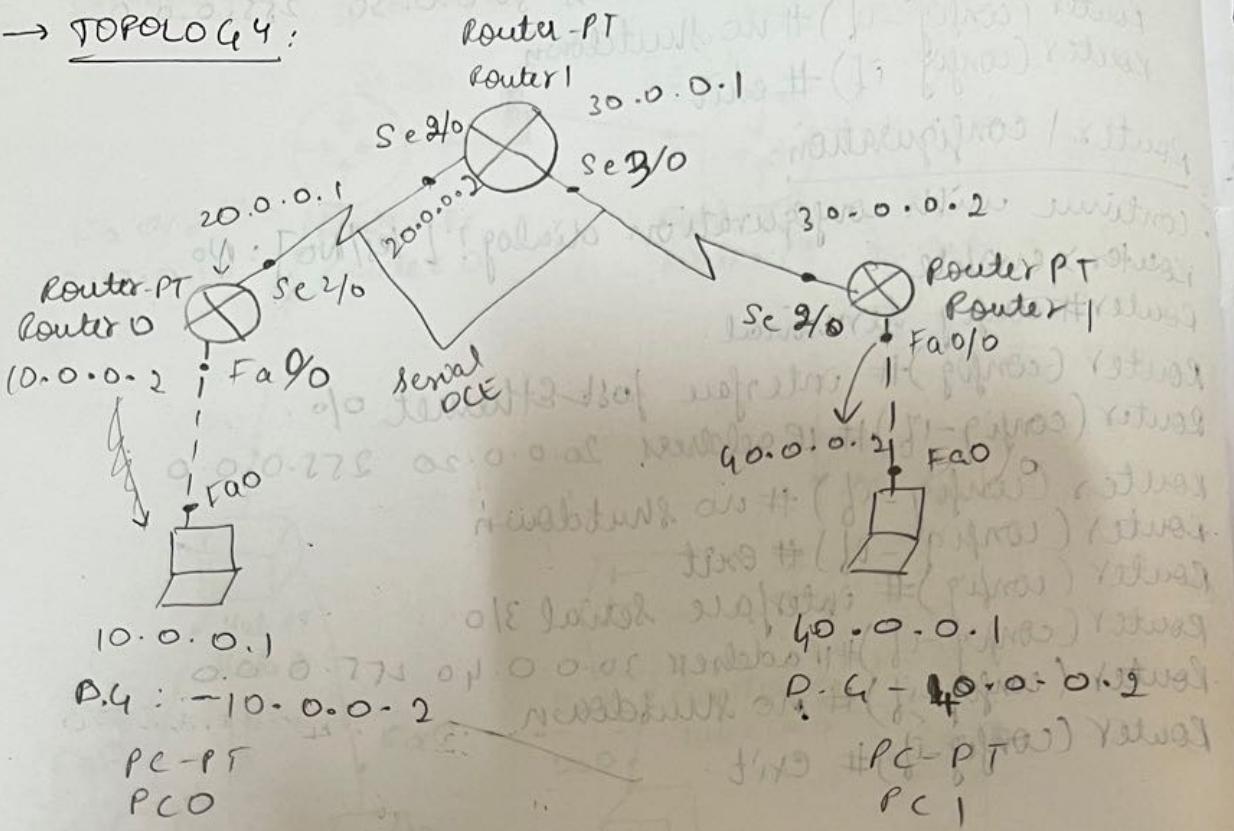
Router (config-if)# exit.

111  
17/11/22

24/11/22

LAB-4

PROGRAM - 3

→ TOPOLOGY:Router-0 configuration:

- Continue with configuration dialog [Yes/No]? No.

Router &gt; enable

Router # config terminal

Router (config)# interface fastEthernet 0/0

Router (config-if)# ip address 10.0.0.2 255.0.0.0

Router (config-if)# no shutdown

Router (config-if)# exit

Router (config)# interface serial 2/0

Router (config-if)# ip address 20.0.0.1 255.0.0.0

Router (config-if)# no shutdown

Router (config-if)# exit

ip route 30.0.0.0 255.0.0.0 20.0.0.2

ip route 40.0.0.0 255.0.0.0 20.0.0.2

Router-1 configuration:

- Continue with configuration dialog [Yes/No]? No.

Router &gt; enable

Router # config terminal

Router (config)# interface fastEthernet 1/0 serial 2/0

Router (config-if)# ip address 20.0.0.2 255.0.0.0

 Router (config)  
 Router (config)  
 Router (config)  
 Router (config)  
 Router (config)  
 Router (config)  
 Router (config)
Router-2 configuration:

continue with

Router &gt; enable

Router # config

Router (config)

Router(config-if) # no shutdown

Router(config-if) # exit

Router(config) # interface serial 3/0

Router(config-if) # IP address 30.0.0.1 255.0.0.0

Router(config-if) # no shutdown

Router(config-if) # exit

Router-2 configuration:

continue with configuration dialog [yes/no]? no

Router> enable

Router# config terminal

Router(config) # interface serial 2/0

Router(config-if) # IP address 30.0.0.2 255.0.0.0

Router(config-if) # no shutdown

Router(config-if) # exit

Router(config) # interface ~~serial 2/0~~ fastEthernet 0/0

Router(config-if) # IP address 30.0.0.2 255.0.0.0

Router(config-if) # no shutdown

Router(config-if) # exit

Router-1

• ip route 10.0.0.0 255.0.0.0 30.0.0.1

ip route 20.0.0.0 255.0.0.0 30.0.0.1

Router-2

ip route 10.0.0.0 255.0.0.0 20.0.0.1

ip route 40.0.0.0 255.0.0.0 30.0.0.2

PC > ping 40.0.0.1

Pinging 40.0.0.1 with 32 bytes of data:

Reply from 40.0.0.1: bytes=32 time=9ms TTL=125

Reply from 40.0.0.1: bytes=32 time=6ms TTL=125

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

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

251 JTT 2MSI = wait SE = wfpd : 1.0.0.0 work plq3

84 JTT 2MSI = wait SE = wfpd : 1.0.0.0 work plq3

251 JTT 2MSI = wait SE = wfpd : 1.0.0.0 work plq3

### Ping statistics:

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

Approximate round trip times in milli-seconds.

Minimum = 2ms, Maximum = 9ms, Average = 4ms.

• PDU from source end device (10.0.0.1)

• Sent successfully to destination end device (40.0.0.1)

### Router 1:

Router > show ip route

C 10.0.0.0/8 is directly connected, FastEthernet 0/0

C 20.0.0.0/8 is directly connected, serial 2/0

Router > enable

Router# config terminal

Router(config)# ip route 30.0.0.0 255.0.0.0 20.0.0.2

Router(config)# ip route 40.0.0.0 255.0.0.0 20.0.0.2

exit

Router > show ip route

C 10.0.0.0/8 is directly connected, FastEthernet 0/0.

C 20.0.0.0/8 is directly connected, Serial 2/0.

S 30.0.0.0/8 [10] via 20.0.0.2

S 40.0.0.0/8 [10] via 20.0.0.2

### Router 2:

Router(config)# ip route 10.0.0.0 255.0.0.0 20.0.0.1

Router(config)# ip route 40.0.0.0 255.0.0.0 30.0.0.2

### Router 3:

Router(config)# ip route 20.0.0.0 255.0.0.0 30.0.0.1

Router(config)# ip route 10.0.0.0 255.0.0.0 30.0.0.1

Router(config)# ip route Destination /n/w subset mask

PC > ping 10.0.0.1

net hop address.

Ping from 10.0.0.1 with 32 bytes of data

Reply from 10.0.0.1: bytes = 32 time = 12ms TTL = 125

Reply from 10.0.0.1: bytes = 32 time = 12ms TTL = 125

Reply from 10.0.0.1: bytes = 32 time = 20ms TTL = 125

ping statistics for 10.0.0.1  
packets: sent = 4, Received = 4 lost = 0 (0% loss)  
Approximate round trip times in milliseconds:  
minimum = 2ms, Maximum = 20ms, Average = 11ms.

Lee  
24/11/22

Less network delay present at dropper's door : 19ms  
less network delay

less network delay at New (19) than present (20)  
less network delay at present (20)

Configuration will take less time  
and minimize less stop time between the jets

network delay reduced

less time at each jet will be taken by the new configuration

less time at each jet will be taken by the new configuration

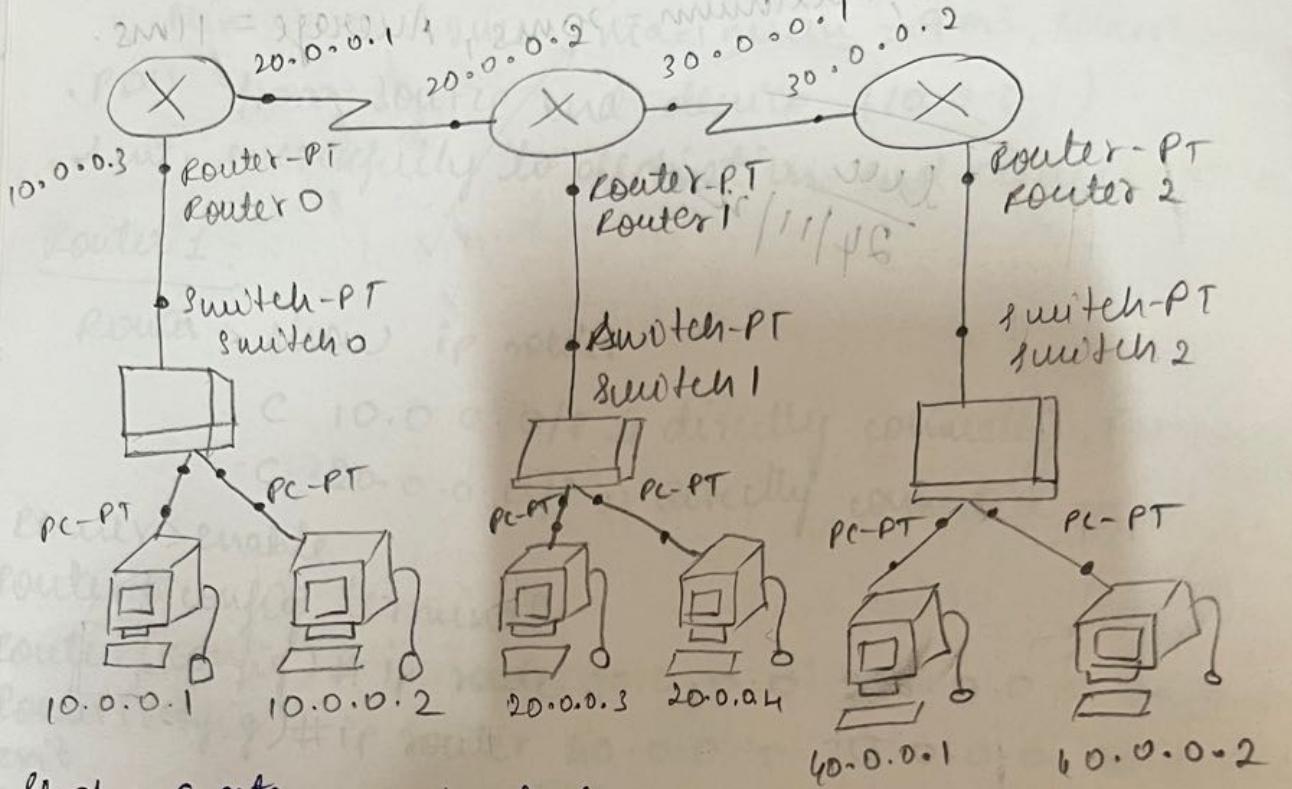
new times at each jet will be taken by the new configuration

1/12/22

[LAB-5]

[PROGRAM - 4]

→ TOPOLOGY:



Step 1: Create a network topology with Router and Switch and PC's.

Step 2: Configuring hosts (PC's) with IP addresses and Default gateway using IP addressing.

Step 3: Configuring the Interfaces (Routers) with IP addresses and Default gateways and assigning the default routes.

Router 0 configuration:

- > Now we will add the IP addr & its subnet mask.
- > We will add the IP addr of the interface FastEthernet 0/0 and its subnet mask.
- > Add IP addr of the next hope to connect with another LAN.

```
Router(config)# interface serial 2/0  
Router(config)# ip address 20.0.0.1 255.0.0.0  
Router(config)# no shutdown  
Router(config)# interface fastethernet 0/0  
Router(config)# ip address 10.0.0.3 255.0.0.0  
Router(config-if)# no shutdown  
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 20.0.0.2
```

### Router 1 configuration.

```
Router(config)# interface serial 2/0  
Router(config)# ip address 20.0.0.2 255.0.0.0  
Router(config-if)# no shutdown  
Router(config-if)# exit  
Router(config)# interface fastethernet 0/0  
Router(config)# ip address 30.0.0.1 255.0.0.0  
Router(config-if)# no shutdown  
Router(config-if)# exit  
Router(config)# ip route 0.0.0.0 0.0.0.0 20.0.0.1  
Router(config)# ip route 0.0.0.0 0.0.0.0 30.0.0.2
```

### Router 2 configuration.

```
Router(config)# interface serial 3/0  
Router(config)# ip address 30.0.0.2 255.0.0.0  
Router(config-if)# no shutdown  
Router(config-if)# exit  
Router(config)# interface fastethernet 1/0  
Router(config)# ip address 40.0.0.3 255.0.0.0  
Router(config)# exit  
Router(config)# ip route 0.0.0.0 30.0.0.1  
Router(config)# ip route 0.0.0.0 30.0.0.1
```

8/12/22

Ping 10.0.0.1

Pong from 10.0.0.1 with 32 bytes of data

Reply from 10.0.0.1: bytes=32 time=3ms TTL=255

Reply from 10.0.0.1: bytes=32 time=3ms TTL=255

Reply from 10.0.0.1: bytes=32 time=6ms TTL=255

Reply from 10.0.0.1: bytes=32 time=2ms TTL=255

Packets

Sent=4, Received=4, Lost=0

Minimum=2ms, Maximum=6ms Average=3ms

Pong 40.0.0.1

Pong from 40.0.0.1 with 32 bytes of data

Request Timed Out

Request Timed Out

Request Timed Out

Request Timed Out

Ping 40.0.0.1

Pong 40.0.0.1 with 32 bytes of data

Reply from 40.0.0.1: bytes=32 time=1ms TTL=255

Reply from 40.0.0.1: bytes=32 time=6ms TTL=255

Reply from 40.0.0.1: bytes=32 time=2ms TTL=255

Reply from 40.0.0.1: bytes=32 time=2ms TTL=255

Ping Statistics

Packets sent=4, Received=4, Lost=0

Approximate round trip time in milliseconds:

Minimum=2ms Maximum=9ms Average=4ms

→ Topo

→ Rout

Router>

Router#

Router(

Router(

Router(

Router(

Router(

Configure

config

service

→ Defa

DNS

Star

Sub

Max

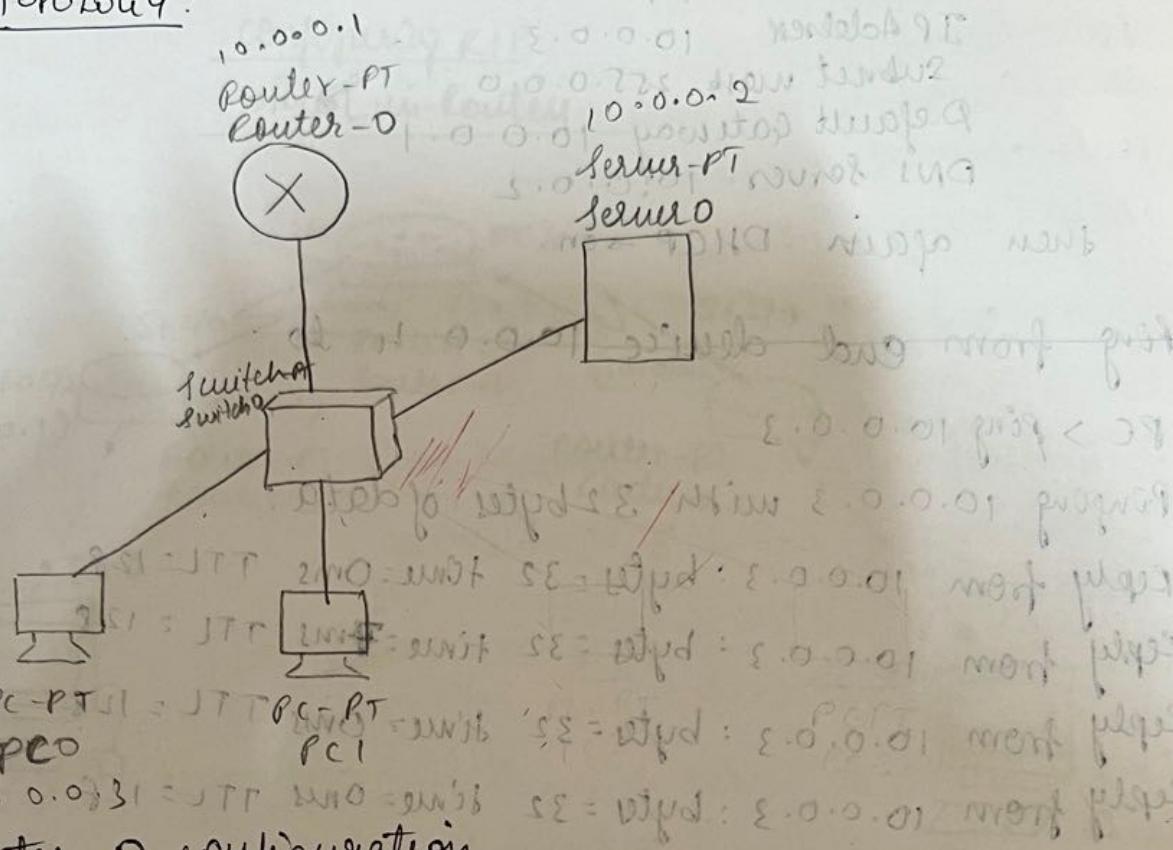
TFS

The

8/12/22

LAB-6

PROGRAM-5

DHCP (Dynamic Host Configuration Protocol)→ TOPOLOGY:→ Router 0 configuration:

Router&gt;enable

Router#config terminal

Router(config)#interface fastEthernet 0/0

Router(config-if)#IP address 10.0.0.1 255.0.0.0

Router(config-if)#no shutdown

exit.

Configure server ip address as 10.0.0.2

config → fastethernet → ip address 10.0.0.2.

services → DHCP (switch on)

→ Default gateway as 10.0.0.1 (Router add\*)

DNS server as 10.0.0.2 (Server ipaddr\*)

Start IP add\* as 10.0.0.3

Subnet mask as 255.0.0.0

Max. no. of users as ~~10.0.0.2~~ 512

TFTP server as 10.0.0.2

Then save.

<u>Server</u>
endgame
services
DHCP
10.0.0.1
10.0.0.2
10.0.0.3
10.0.0.3
512
10.0.0.2
Same

→ For an end device  
Desktop → IP configuration → DHCP

IP Address 10.0.0.3  
Subnet mask 255.0.0.0  
Default gateway 10.0.0.1  
DNS server 10.0.0.2

Then again DHCP → on.

Ping from end device 10.0.0.11 to

PC > ping 10.0.0.3

Pingpong 10.0.0.3 with 32 bytes of data:

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

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

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

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

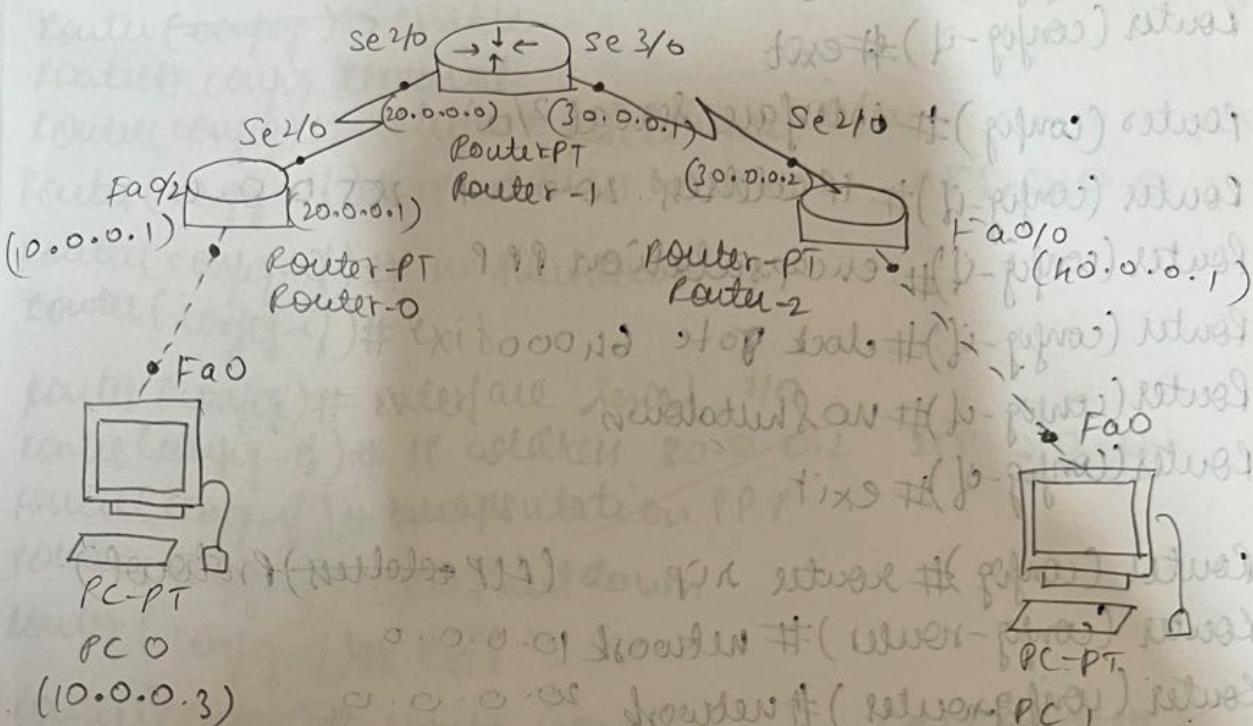
Ping statistics for 10.0.0.3:

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

Approximate round trip times in milli-seconds;

minimum = 0ms, Maximum = 7ms, Average = 1ms

8/12/22

TOPOLOGY:Configuring RIP Protocol in Routers

- RIP: Router Information Protocol
  - DVA: Distance Vector Algorithm
  - Finds optimal path. Also known as ~~Router Selection~~ protocol.
  - Constant / Periodic update relayed throughout network
  - Information about neighbours passed to all and trusted.
- PROCEDURE:
- Select 2 end devices & set their IP addresses as 10.0.0.3 & 40.0.0.3 with subnet mask 255.0.0.0 respectively.
  - Select 3 generic routers & make connections b/w routers and end devices.

## route -o -Configuration:

Router > enable

Router#config terminal

```
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

Router(config)# interface serial 2/0

```
Router(config-if)# ip address 20.0.0.1 255.0.0.0
```

Router (config-if) # encapsulation ppp

```
Router(config-if)#clock rate 64000
```

Router(config-if)# no shutdown

Router(Config-of)# exit

Router (config)# router rip (RIP address) protocol

Router (config-router) # network 10.0.0.0

Router (config-router) # network 20.0.0.0

```
router(config-router) # network exit
```

## Outer - I configuration:

~~Router > enable~~

Router # config terminal

```
Router(config)# interface serial 2/0
```

Router(config-if)# IP address 20.0.0.2 255.0.0.0

```
Router (config-if)# encapsulation ppp
```

Router (config-if)# no shutdown

Router (config-if)# exit

Router # config terminal

Routee (config) # interface serial 3/0

Router (config-if)# IP address 30.0.0.1

Router(config-if)#encapsulation ppp

Router(config-if)# clock 200 6:00

Counter (Config-of) no shutdown

center (cont'd-of) # exit

Router (config) # router rip  
Router (config-router) # network 20.0.0.0  
Router (config-router) # network 30.0.0.0  
Router (config-router) # exit

(RIP ~~distance~~  
Protocol)

### Router-2 configuration:

Router (config) > enable  
Router # config terminal  
Router (config) # interface fast Ethernet 0/0  
Router (config-if) # IP address 40.0.0.1 255.0.0.0  
Router (config-if) # no shutdown  
Router (config-if) # exit  
Router (config) # interface serial 1/0  
Router (config-if) # IP address 30.0.0.2 255.0.0.0  
Router (config-if) # encapsulation PPP  
Router (config-if) # no shutdown  
Router (config-if) # exit

Router (config) # router rip  
Router (config-router) # network 30.0.0.0  
Router (config-router) # network 40.0.0.0  
Router (config-router) # exit.

(RIP ~~address~~  
Protocol)

Assign 10.0.0.1 as gateway for End Device (10.0.0.3)  
Assign 40.0.0.1 as gateway for End Device (40.0.0.3)

### Ping Statistics:

PC> Ping 40.0.0.3

Pinging 40.0.0.3 with 32 bytes of data:

Reply from 40.0.0.3 bytes = 32 time = 2ms TTL = 128

Reply from 40.0.0.3 bytes = 32 time = 2ms TTL = 128

Reply from 40.0.0.3 bytes = 32 time = 2ms TTL = 125

Reply from 40.0.0.3 bytes = 32 time = 2ms TTL = 125

Reply from 40.0.0.3 bytes = 32 time = 2ms TTL = 125

Ring statistics for 40.0.0.3:

Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),  
Approximate round trip times in milli-seconds:

Minimum = 3ms, Maximum = 12ms, Average = 4ms

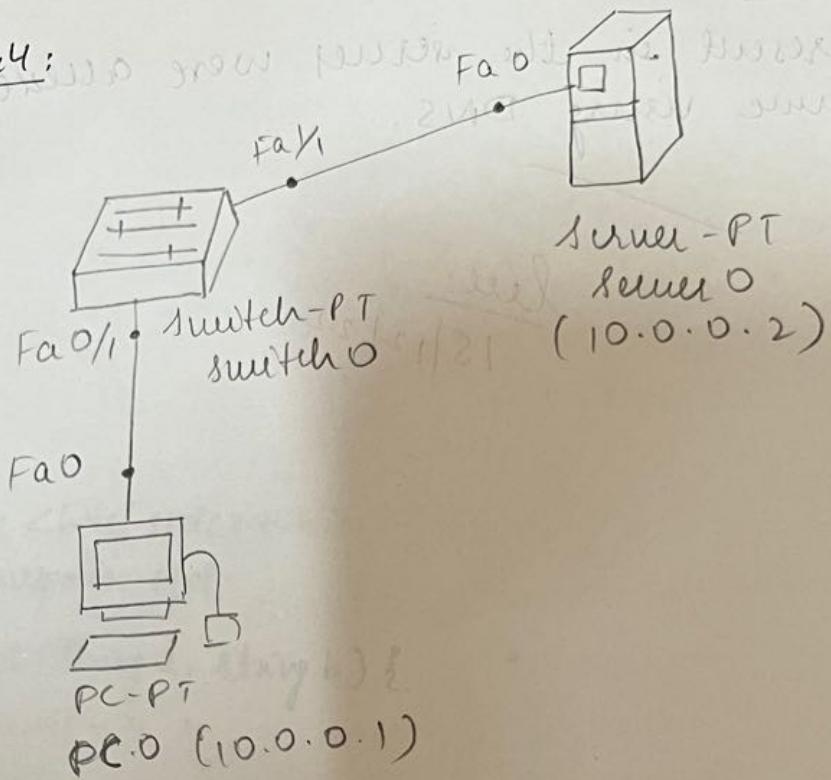
### OBSERVATION:

RIP protocol enabled the sharing of routing table information throughout the network.  
RIP was successfully sent from C end device to another device.

15/12/22

## Demonstration of WEB server and DNS using Packet Tracer

### TOPOLOGY:



### PROCEDURE:

- Select and end device and ~~configure~~ it with an IP address of 10.0.0.1 & subnet mask 255.0.0.0.
- Connect it to a switch and then make a connection b/w switch and a server.
- Server configuration to make it have ~~to~~ an IP address of 10.0.0.2 & subnet mask 255.0.0.0.
- Click on the server and go to services menu. Ensure that HTTP service is switched on. Go to DNS service, switch it on. Under resource records enter name of the desired website (e.g: www.google.com)
- Enter the IP address of the server { then add the records. Select the TFTP service & switch it on.
- Select the end device and go to desktop.
- Select web browser icon and in the URL space enter the website name / address along with filename.

in HTTP service. eg)  $\text{http://10.0.0.2/helloworld.html}$

• file is displayed along with content.

### OBSERVATION:

- Files present in the server were accessed by end device using DNS.

Lee  
15/12/22

29/12/22 [LAB-8]

[Program-7]

Write a program to detect errors using CRC 16-bit.

$$(x^{16} + x^{12} + x^5 + 1)$$

 $(1\ 0\ 1\ 1\ 1\ 0\ 1) \Rightarrow \text{data given.}$ 

$$\begin{array}{ccccccccccccccccc} x^{16} & x^{15} & x^{14} & x^{13} & x^{12} & x^{11} & x^{10} & x^9 & x^8 & x^7 & x^6 & x^5 & x^4 & x^3 & x^2 & x^1 & \\ \hline 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \end{array}$$
 $(n-1) \Rightarrow (16-1) \Rightarrow \underline{15} \quad \text{key:}$ 

(Add 15 0's to the given data.)

Code:

```
#include <bits/stdc++.h>
using namespace std;

string xor1(string a, string b) {
    string result = "";
    int n = b.length();
    for (int i = 1; i < n; i++) {
        if (a[i] == b[i])
            result += "0";
        else
            result += "1";
    }
    return result;
}

string mod2div(string dividend, string divisor) {
    int pick = divisor.length();
    string temp = dividend.substr(0, pick);
    int n = dividend.length();
    while (pick < n) {
        if (temp[0] == '1') {
            temp = xor1(divisor, temp) + dividend[pick];
        } else
            temp = xor1(std::string(pick, '0'), temp) +
                dividend[pick];
        pick++;
    }
    return temp;
}
```

```
pick += 1;
} if (tup[0] == '1')
    tup = xorI(olimior, tup);
else
    tup = xorI(std::string(pick, '0'), tup);
return tup;
}

void encodeData(string data, string key)
{
    int l_key = key.length();
    string appended_data = (data + std::string(l_key - 1, '0'));
    string remainder = mod2div(appended_data, key);
    string codeword = data + remainder;
    cout << "Remainder:" << remainder << "\n";
    cout << "Encoded Data (Data + Remainder):" << codeword << "\n";
}

void receiver(string data, string key)
{
    string curxor = mod2div(data.substr(0, key.size()), key);
    int curr = key.size();
    while (curr != data.size()) {
        if (curxor.size() != key.size()) {
            curxor.push_back(data[curr++]);
        } else {
            curxor = mod2div(curnxor, key);
        }
        if (curxor.size() == key.size()) {
            curxor = mod2div(curnxor, key);
        }
        if (curxor.find('1') != std::string::npos)
```

```

    { cout << "there is some error in data" << endl;
} else {
    cout << "correct message received" << endl;
}
}

int main()
{
    string data = "101110100000000000000000";
    string key = "1000100000100001";
    encodeData(data, key);
    return 0;
}

```

### Output:

remainder: 100001100101111

Encoded Data (Data+Remainder): 10110100000000000

correct message received

5/1/23

LAB - 9

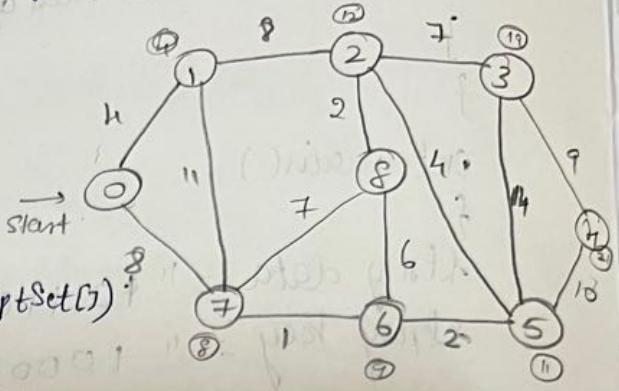
Program - 8

## Dijkstra's Algorithm

```

##include <iostream>
using namespace std;
#include <limits.h>
#define V 9
int mindistance(int dist[], bool sptSet[]);
int main = INT_MAX, min_index;
for (int v = 0; v < V; v++)
    if (sptSet[v] == false && dist[v] <= min)
        min = dist[v], min_index = v;
return min_index;
void printSolution(int dist[])
{
    cout << "Vertex \t Distance from Source" << endl;
    for (int i = 0; i < V; i++)
        cout << i << "\t\t\t\t" << dist[i] << endl;
}
void dijkstra(int graph[V][V], int src)
{
    int dist[V];
    bool sptSet[V];
    for (int i = 0; i < V; i++)
        dist[i] = INT_MAX, sptSet[i] = false;
    dist[src] = 0;
    for (int count = 0; count < V - 1; count++) {
        int u = mindistance(dist, sptSet);
        sptSet[u] = true;
        for (int v = 0; v < V; v++)
            if (!sptSet[v] && graph[u][v]
                && dist[u] != INT_MAX
                && dist[u] + graph[u][v] < dist[v])

```

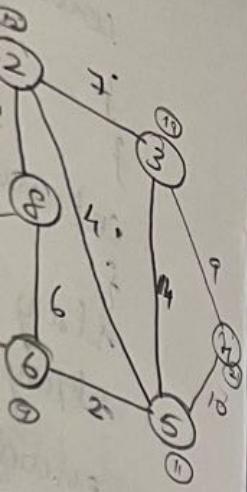


```

} printSol
} out man
{ out gro
dijk
seete
}

```

OUTPUT



$\text{dist}[v] = \text{dist}[u] + \text{graph}[u][v]$ ; [01-3A]

} printSolution( $\text{dist}$ ); [using recursive simulation]  
} int main()

{ out  $\text{graph}[v][v] = \{ \{ 0, 4, 0, 0, 0, 0, 0, 8, 0 \},$  → 0 to 0 → 0  
   ' { 4, 0, 8, 0, 0, 0, 0, 11, 0 },  
   ' { 0, 8, 0, 7, 0, 4, 0, 0, 2 },  
   ' { 0, 0, 7, 0, 9, 14, 0, 0, 0 },  
   ' { 0, 0, 0, 9, 0, 16, 0, 0, 0 },  
   ' { 0, 0, 4, 14, 10, 0, 2, 0, 0 },  
   ' { 0, 0, 0, 0, 0, 2, 0, 1, 6 },  
   ' { 8, 11, 0, 0, 0, 0, 1, 0, 7 },  
   ' { 0, 0, 2, 0, 0, 0, 6, 7, 0 } }.

dijkstra( $\text{graph}$ , 0);

return 0;

}

<u>OUTPUT:</u>	Vertex	Distance from Source
	0	0
	1	4
	2	12
	3	19
	4	21
	5	11
	6	9
	7	8
	8	14

12/11/23

LAB-10

Program-9

Distance vector algorithm [to find suitable path for transmission]

→ Distance-Vector Routing Algorithm for a Node

Distance-Vector-Routing()

{

// Initialize (create initial vectors for the node)

D [myself] = 0

for(y=1 to N)

{

if(y is a neighbor)

D[y] = c[myself][y]

else

D[y] =  $\infty$

}

send vector [D[1], D[2], ..., D[N]] to all neighbors.

Update (improve the vector with the vector received from a neighbor)

repeat (forever)

{

wait (for a vector D<sub>w</sub> from a neighbor w or any change in the link)

for(y=1 to N)

{

D[y] = min[D[y], (c[myself][w] + D<sub>w</sub>[y])] // Bellman-Ford eqn

if (any change in the vector)

send vector [D[1], D[2], ..., D[N]] to all neighbors

code:  

```
#include  
using  
#define  
10 int  
class r  
clear  
int ta  
table  
forl  
3vo  
for  
ad  
ta  
}}  
f  
5
```

code:

```
#include <bits/stdc++.h>
using namespace std;
#define MAX 10
int n;
class router {
    char adj_new[MAX], adj_old[MAX];
    int table_new[MAX];
    table_old[MAX]; public: router() {
        for(int i=0; i<MAX; i++) table_old[i] = table_new[i] = 99;
    }
    void copy() {
        for(int i=0; i<n; i++) {
            adj_old[i] = adj_new[i];
            table_old[i] = table_new[i];
        }
    }
    int equal() {
        for(int i=0; i<n; i++)
            if(table_old[i] != table_new[i] || adj_new[i] != adj_old[i])
                return 0;
        return 1;
    }
    void input(int j) {
        cout << "Enter 1 if the corresponding router is adjacent to router ";
        cout << (char)('A' + j) << " else enter 99 ";
        cout << endl << " ";
        for(int i=0; i<n; i++) if(i != j)
            cout << (char)('A' + i) << " ";
        cout << endl << " Enter matrix: ";
        for(int i=0; i<n; i++) {
            if(i == j) table_new[i] = 0;
            else cin >> table_new[i];
            adj_new[i] = (char)('A' + i);
        }
        cout << endl;
    }
    void display() {
        cout << "\nDestination Router: ";
        for(int i=0; i<n; i++)
            cout << (char)('A' + i) << " ";
        cout << "\nOutgoing Line: ";
        for(int i=0; i<n; i++)
            cout << (char)('A' + i) << " ";
        cout << "\nOutgoing Line: "
```

```

for (int i=0; i<n; i++)
    cout << adj_new[i] << " ";
    cout << "\n Hop count: ";
    for (int i=0; i<n; i++) cout << table_new[i] << " ";
}

void build (int j) {
    for (int i=0; i<n; i++)
        for (int k=0; k!=j && k<n; k++) if (table_old[i] == 99)
            if ((table_new[i] + table_new[k]) < table_new[k]),
    {
        table_new[k] = table_new[i] + table_new[k];
        adj_new[k] = (char) ('A' + i);
    }
} } > {MAX};

void build_table() {
    int i=0, j=0;
    while (i!=n)
    {
        for (i=j; i<n; i++)
            r[i].copy();
            r[i].build(i);
        for (i=0; i<n; i++) if (!r[i].equal())
        {
            j=i;
            break;
        }
    }
}

int main()
{
    cout << "Enter the number of routers (<" << MAX << ")";
    cin >> n; for (int i=0; i<n; i++) r[i].input(i);
    build_table(); for (int i=0; i<n; i++)
    cout << "Router Table entries for router "
    << (char) ('A' + i) << ":" -"; r[i].display();
}
}

```

OUTPUT

Enter  
Enter  
BCD  
Enter  
Enter  
@AC  
Enter  
Enter  
A  
B  
C  
E  
A  
C  
E

OUTPUT:  
Enter the number the routers (<10): 5  
Enter 1 if the corresponding router is adjacent to router A else enter 99:  
BCDE

Enter matrix: 1 1 99 99

Enter 1 of the corres. "

BCD E

Enter matrix: 1 99 99 99

Enter 1 " " " router C " "

AB D E

Enter matrix: 1 99 1 1

Enter 1 " " " router D " "

AB CE

Enter matrix: 99 99 1 99

Enter 1 " " " router E " "

ABCD

Enter matrix: 99 99 1 99

Router table entries for router A:-

Destination Router: ABCDE

Outgoing line: ABCDE

Hop count: 0 1 1 99 99

Router table entries for router B:-

DR: ABCDE

OL: ABCDE

HC: 1 0 99 99 99

Router C:-

DR: ABCDE

OL: ABCDE

HC: 1 99 0 1 1

Router D:-

DR: ABCDE

OL: ABCDE

HC: 99 99 1 0 99

Router E:-

DR: ABCDE

OL: ABCDE

HC: 99 99 1 99 0

28/11/23

## Lec Program

### Socket Programming using TCP

⇒ socket

⇒ Stream

⇒ Types of sockets

⇒ Client

⇒ Server

→ socket programming with UDP :-

⇒ Server & Client: