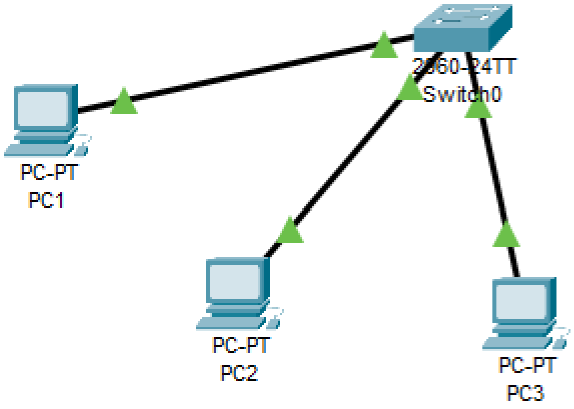
Dinesh Prabhu S
Thiagarajar College of Engineering

Module 3 and 4

Q1. Simulate a small network with switches and multiple devices. Use ping to generate traffic and observe the MAC address table of the switch. Capture packets using Wireshark to analyze Ethernet frames and MAC addressing.

1. Open **Cisco Packet Tracer**.
2. Drag and drop the following devices:
   - **1 Switch** (e.g., Cisco 2960)
   - **3 PCs**
3. Use **Copper Straight-through cables** to connect:
   - PC1 → FastEthernet 0/1 (Switch)
   - PC2 → FastEthernet 0/2 (Switch)
   - PC3 → FastEthernet 0/3 (Switch)

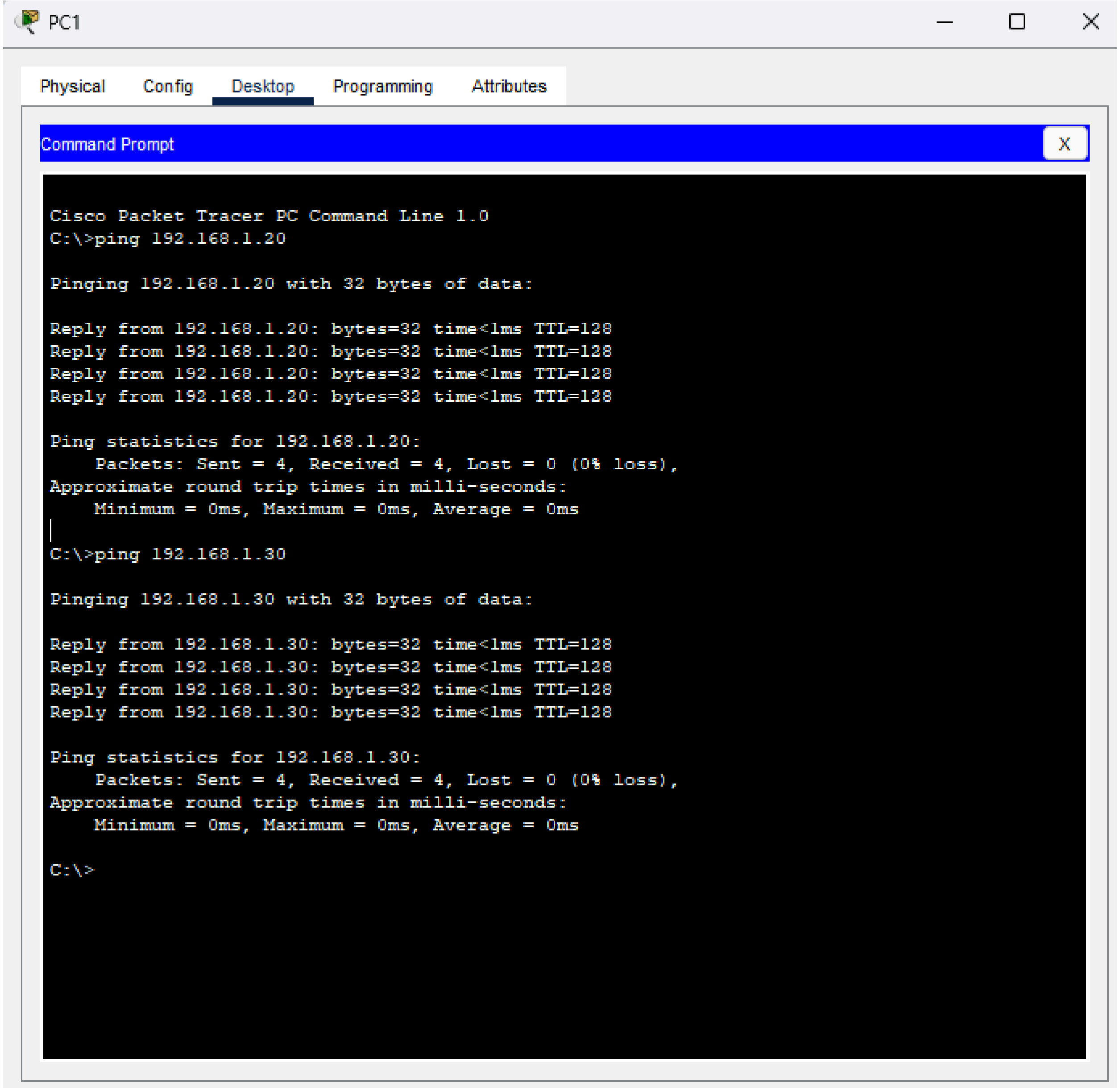


Click on each **PC**, go to **Desktop** → **IP Configuration**, and assign:

- **PC1:** `192.168.1.10 / 255.255.255.0`
- **PC2:** `192.168.1.20 / 255.255.255.0`
- **PC3:** `192.168.1.30 / 255.255.255.0`

Open the **Command Prompt** (`Desktop → Command Prompt`) on **PC1** and type:

```
ping 192.168.1.20
ping 192.168.1.30
```



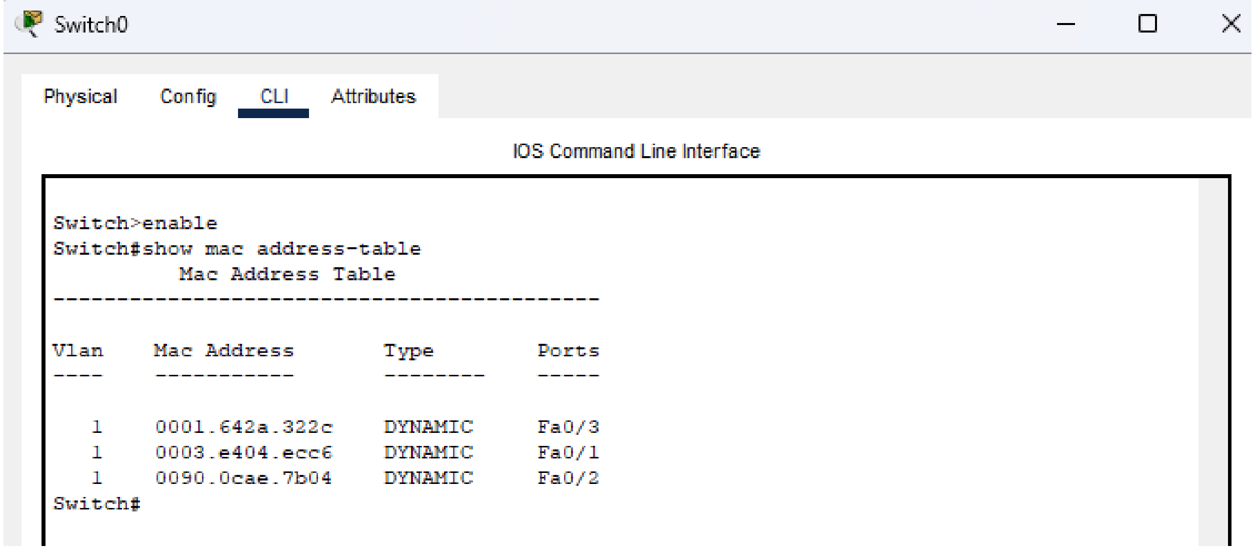
```
Cisco Packet Tracer PC Command Line 1.0
C:\>ping 192.168.1.20

Pinging 192.168.1.20 with 32 bytes of data:

Reply from 192.168.1.20: bytes=32 time<1ms TTL=128
Reply from 192.168.1.20: bytes=32 time<1ms TTL=128
Reply from 192.168.1.20: bytes=32 time<1ms TTL=128
Reply from 192.168.1.20: bytes=32 time<1ms TTL=128

Ping statistics for 192.168.1.20:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 0ms, Average = 0ms

C:\>ping 192.168.1.30

Pinging 192.168.1.30 with 32 bytes of data:

Reply from 192.168.1.30: bytes=32 time<1ms TTL=128
Reply from 192.168.1.30: bytes=32 time<1ms TTL=128
Reply from 192.168.1.30: bytes=32 time<1ms TTL=128
Reply from 192.168.1.30: bytes=32 time<1ms TTL=128

Ping statistics for 192.168.1.30:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 0ms, Average = 0ms

C:\>
```

Click on the **Switch**.

Go to **CLI** and enter:

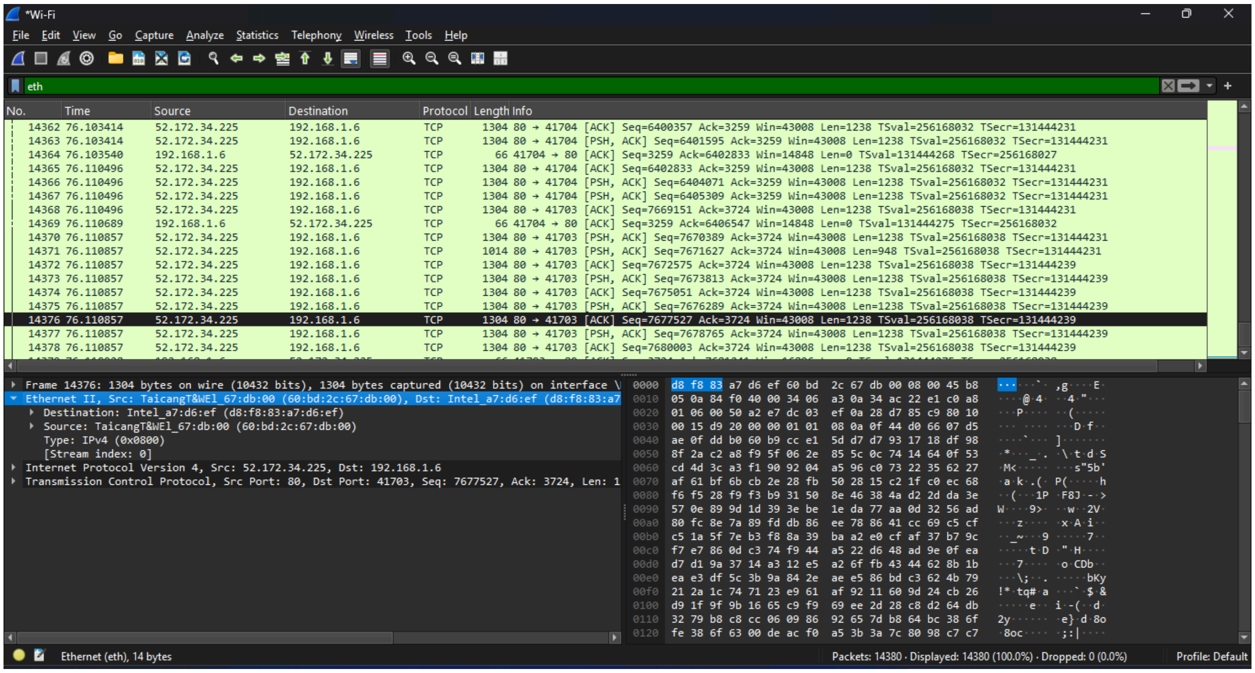
```
enable
show mac address-table
```

This will display the **MAC addresses** learned by the switch.



```
Switch>enable
Switch#show mac address-table
          Mac Address Table
-------------------------------------------

Vlan    Mac Address       Type        Ports
----    -----------       --------    -----

   1    0001.642a.322c    DYNAMIC     Fa0/3
   1    0003.e404.ecc6    DYNAMIC     Fa0/1
   1    0090.0cae.7b04    DYNAMIC     Fa0/2
Switch#
```

Q2. Capture and analyze Ethernet frames using Wireshark. Inspect the structure of the frame, including destination and source MAC addresses, Ethertype, payload, and FCS.



```
eth
```

```
No.   Time      Source         Destination    Protocol Length Info
14362 76.103414 52.172.34.225  192.168.1.6    TCP      1304 80 → 41704 [ACK] Seq=6400357 Ack=3259 Win=43008 Len=1238 TSval=256168032 TSecr=131444231
14363 76.103414 52.172.34.225  192.168.1.6    TCP      1304 80 → 41704 [PSH, ACK] Seq=6401595 Ack=3259 Win=43008 Len=1238 TSval=256168032 TSecr=131444231
14364 76.103540 192.168.1.6    52.172.34.225  TCP        66 41704 → 80 [ACK] Seq=3259 Ack=6402833 Win=14848 Len=0 TSval=131444268 TSecr=256168027
14365 76.110496 52.172.34.225  192.168.1.6    TCP      1304 80 → 41704 [ACK] Seq=6402833 Ack=3259 Win=43008 Len=1238 TSval=256168032 TSecr=131444231
14366 76.110496 52.172.34.225  192.168.1.6    TCP      1304 80 → 41704 [PSH, ACK] Seq=6404071 Ack=3259 Win=43008 Len=1238 TSval=256168032 TSecr=131444231
14367 76.110496 52.172.34.225  192.168.1.6    TCP      1304 80 → 41704 [PSH, ACK] Seq=6405309 Ack=3259 Win=43008 Len=1238 TSval=256168032 TSecr=131444231
14368 76.110496 52.172.34.225  192.168.1.6    TCP      1304 80 → 41703 [ACK] Seq=7669151 Ack=3724 Win=43008 Len=1238 TSval=256168038 TSecr=131444231
14369 76.110689 192.168.1.6    52.172.34.225  TCP        66 41704 → 80 [ACK] Seq=3259 Ack=6406547 Win=14848 Len=0 TSval=131444275 TSecr=256168032
14370 76.110857 52.172.34.225  192.168.1.6    TCP      1304 80 → 41703 [PSH, ACK] Seq=7670389 Ack=3724 Win=43008 Len=1238 TSval=256168038 TSecr=131444231
14371 76.110857 52.172.34.225  192.168.1.6    TCP      1014 80 → 41703 [PSH, ACK] Seq=7671627 Ack=3724 Win=43008 Len=948 TSval=256168038 TSecr=131444231
14372 76.110857 52.172.34.225  192.168.1.6    TCP      1304 80 → 41703 [ACK] Seq=7672575 Ack=3724 Win=43008 Len=1238 TSval=256168038 TSecr=131444239
14373 76.110857 52.172.34.225  192.168.1.6    TCP      1304 80 → 41703 [PSH, ACK] Seq=7673813 Ack=3724 Win=43008 Len=1238 TSval=256168038 TSecr=131444239
14374 76.110857 52.172.34.225  192.168.1.6    TCP      1304 80 → 41703 [ACK] Seq=7675051 Ack=3724 Win=43008 Len=1238 TSval=256168038 TSecr=131444239
14375 76.110857 52.172.34.225  192.168.1.6    TCP      1304 80 → 41703 [PSH, ACK] Seq=7676289 Ack=3724 Win=43008 Len=1238 TSval=256168038 TSecr=131444239
14376 76.110857 52.172.34.225  192.168.1.6    TCP      1304 80 → 41703 [ACK] Seq=7677527 Ack=3724 Win=43008 Len=1238 TSval=256168038 TSecr=131444239
14377 76.110857 52.172.34.225  192.168.1.6    TCP      1304 80 → 41703 [PSH, ACK] Seq=7678765 Ack=3724 Win=43008 Len=1238 TSval=256168038 TSecr=131444239
14378 76.110857 52.172.34.225  192.168.1.6    TCP      1304 80 → 41703 [ACK] Seq=7680003 Ack=3724 Win=43008 Len=1238 TSval=256168038 TSecr=131444239
```

```
> Frame 14376: 1304 bytes on wire (10432 bits), 1304 bytes captured (10432 bits) on interface \
v Ethernet II, Src: TaicangT&WEl_67:db:00 (60:bd:2c:67:db:00), Dst: Intel_a7:d6:ef (d8:f8:83:a7
    > Destination: Intel_a7:d6:ef (d8:f8:83:a7:d6:ef)
    > Source: TaicangT&WEl_67:db:00 (60:bd:2c:67:db:00)
      Type: IPv4 (0x0800)
      [Stream index: 0]
> Internet Protocol Version 4, Src: 52.172.34.225, Dst: 192.168.1.6
> Transmission Control Protocol, Src Port: 80, Dst Port: 41703, Seq: 7677527, Ack: 3724, Len: 1
```

```
0000  d8 f8 83 a7 d6 ef 60 bd  2c 67 db 00 08 00 45 b8   ······`· ,g····E·
0010  05 0a 84 f0 40 00 34 06  a3 0a 34 ac 22 e1 c0 a8   ····@·4· ··4·"···
0020  01 06 00 50 a2 e7 dc 03  ef 0a 28 d7 85 c9 80 10   ···P···· ··(·····
0030  00 15 d9 20 00 00 01 01  08 0a 0f 44 d0 66 07 d5   ··· ···· ···D·f··
0040  ae 0f dd b0 60 b9 cc e1  5d d7 d7 93 17 18 df 98   ····`··· ]·······
0050  8f 2a c2 a8 f9 5f 06 2e  85 5c 0c 74 14 64 0f 53   ·*···_·. ·\·t·d·S
0060  cd 4d 3c a3 f1 90 92 04  a5 96 c0 73 22 35 62 27   ·M<····· ···s"5b'
0070  af 61 bf 6b cb 2e 28 fb  50 28 15 c2 1f c0 ec 68   ·a·k·.(· P(·····h
0080  f6 f5 28 f9 f3 b9 31 50  8e 46 38 4a d2 2d da 3e   ··(···1P ·F8J·-·>
0090  57 0e 89 9d 1d 39 3e be  1e da 77 aa 0d 32 56 ad   W····9>· ··w··2V·
00a0  80 fc 8e 7a 89 fd db 86  ee 78 86 41 cc 69 c5 cf   ···z···· ·x·A·i··
00b0  c5 1a 5f 7e b3 f8 8a 39  ba a2 e0 cf af 37 b7 9c   ··_~···9 ·····7··
00c0  f7 e7 86 0d c3 74 f9 44  a5 22 d6 48 ad 9e 0f ea   ·····t·D ·"·H····
00d0  d7 d1 9a 37 14 a3 12 e5  a2 6f fb 43 44 62 8b 1b   ···7···· ·o·CDb··
00e0  ea e3 df 5c 3b 9a 84 2e  ae e5 86 bd c3 62 4b 79   ···\;··. ·····bKy
00f0  21 2a 1c 74 71 23 e9 61  af 92 11 60 9d 24 cb 26   !*·tq#·a ···`·$·&
0100  d9 1f 9f 9b 16 65 c9 f9  69 ee 2d 28 c8 d2 64 db   ·····e·· i·-(··d·
0110  32 79 b8 c8 cc 06 09 86  92 65 7d b8 64 bc 38 6f   2y······ ·e}·d·8o
0120  fe 38 6f 63 00 de ac f0  a5 3b 3a 7c 80 98 c7 c7   ·8oc···· ·;:|····
```

Ethernet (eth), 14 bytes

Packets: 14380 · Displayed: 14380 (100.0%) · Dropped: 0 (0.0%)

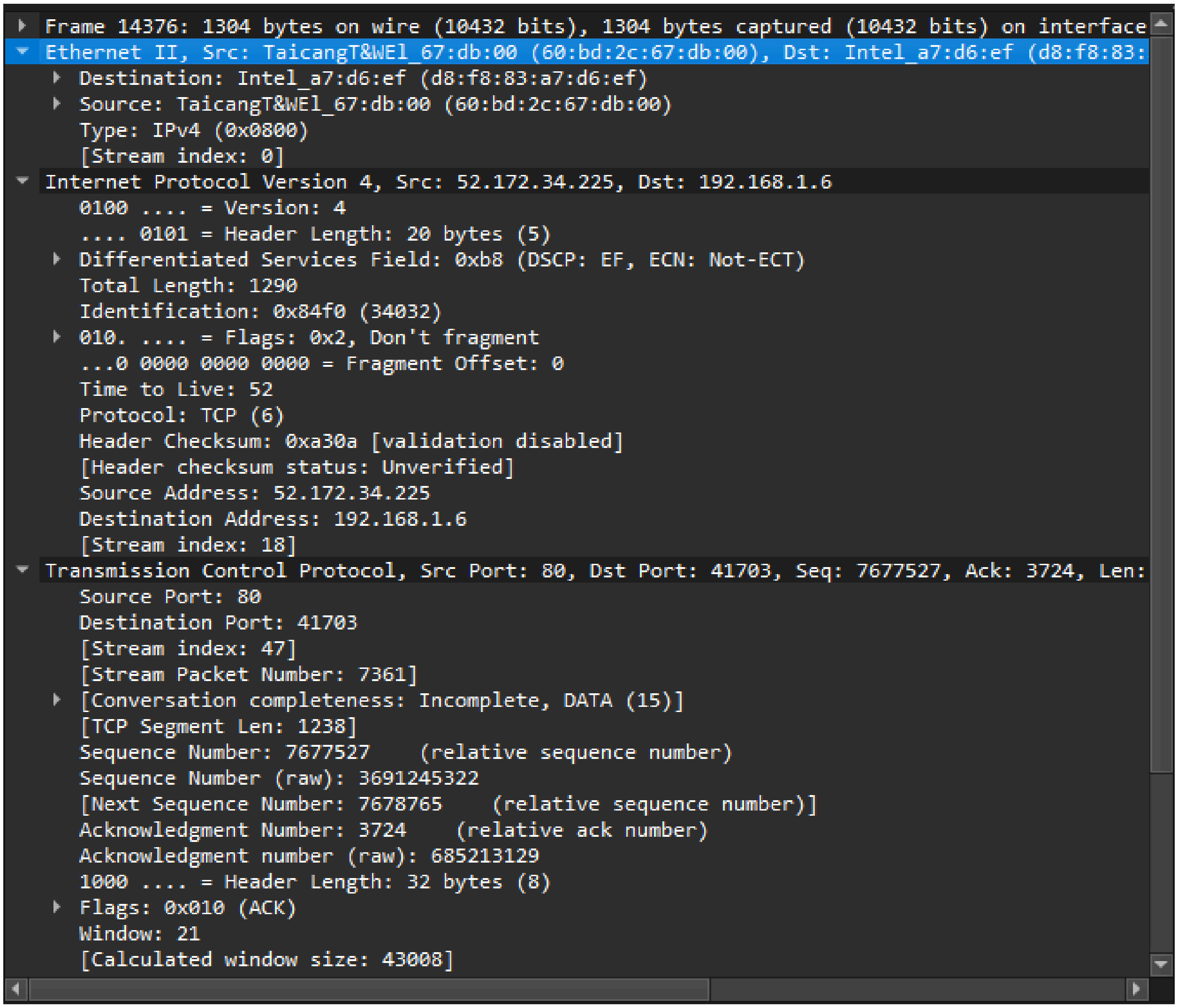
Profile: Default

In the **filter bar**, type:

```
eth
```

- This shows **all Ethernet frames**

- Click on a **frame** to expand its details.

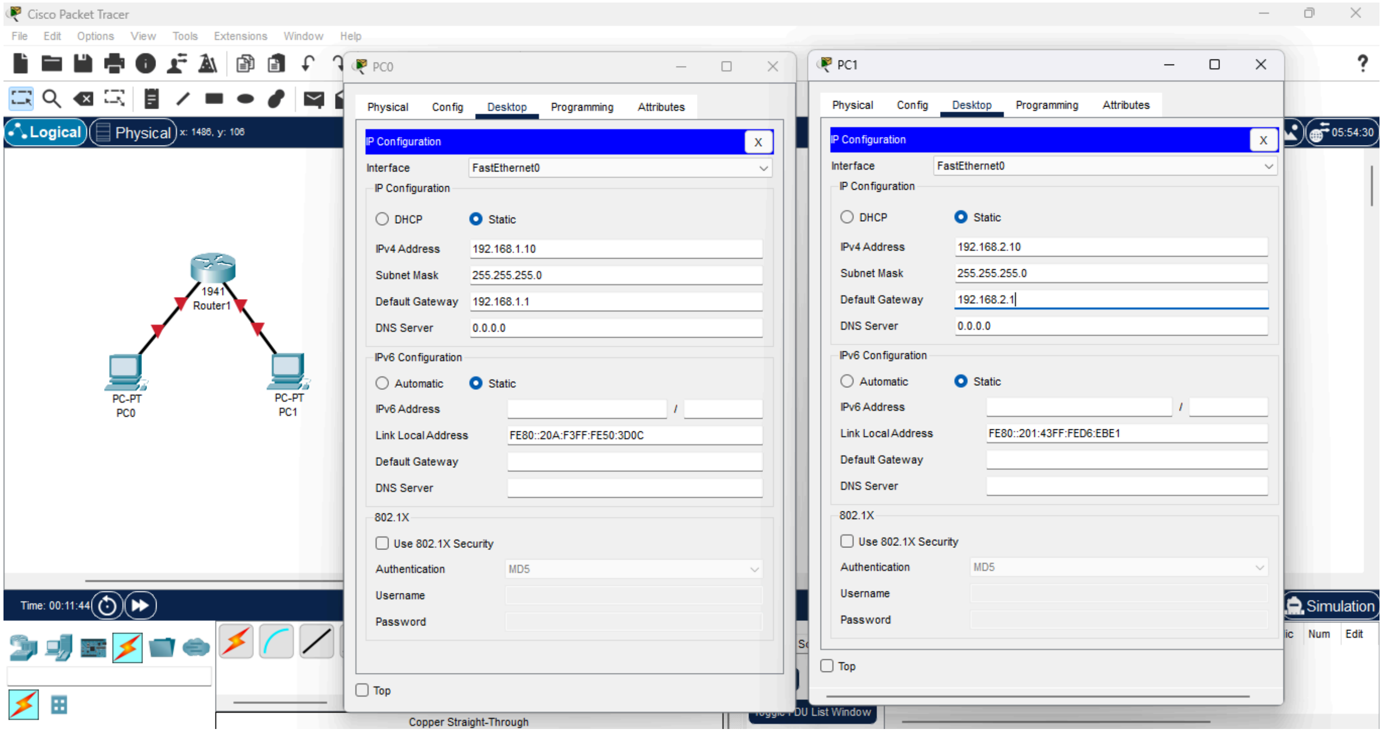


```
▸ Frame 14376: 1304 bytes on wire (10432 bits), 1304 bytes captured (10432 bits) on interface
▾ Ethernet II, Src: TaicangT&WEl_67:db:00 (60:bd:2c:67:db:00), Dst: Intel_a7:d6:ef (d8:f8:83:
  ▸ Destination: Intel_a7:d6:ef (d8:f8:83:a7:d6:ef)
  ▸ Source: TaicangT&WEl_67:db:00 (60:bd:2c:67:db:00)
    Type: IPv4 (0x0800)
    [Stream index: 0]
▾ Internet Protocol Version 4, Src: 52.172.34.225, Dst: 192.168.1.6
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  ▸ Differentiated Services Field: 0xb8 (DSCP: EF, ECN: Not-ECT)
    Total Length: 1290
    Identification: 0x84f0 (34032)
  ▸ 010. .... = Flags: 0x2, Don't fragment
    ...0 0000 0000 0000 = Fragment Offset: 0
    Time to Live: 52
    Protocol: TCP (6)
    Header Checksum: 0xa30a [validation disabled]
    [Header checksum status: Unverified]
    Source Address: 52.172.34.225
    Destination Address: 192.168.1.6
    [Stream index: 18]
▾ Transmission Control Protocol, Src Port: 80, Dst Port: 41703, Seq: 7677527, Ack: 3724, Len:
    Source Port: 80
    Destination Port: 41703
    [Stream index: 47]
    [Stream Packet Number: 7361]
  ▸ [Conversation completeness: Incomplete, DATA (15)]
    [TCP Segment Len: 1238]
    Sequence Number: 7677527    (relative sequence number)
    Sequence Number (raw): 3691245322
    [Next Sequence Number: 7678765    (relative sequence number)]
    Acknowledgment Number: 3724    (relative ack number)
    Acknowledgment number (raw): 685213129
    1000 .... = Header Length: 32 bytes (8)
  ▸ Flags: 0x010 (ACK)
    Window: 21
    [Calculated window size: 43008]
```

Q3. Configure static IP addresses, modify MAC addresses, and verify network connectivity using ping and ifconfig commands.

Static IP :

Click the PC and go to the desktop menu and inside that click the ip configuration to assign a static IP, default gateway, subnet mask to the end devices.



PC0 — Physical | Config | Desktop | Programming | Attributes

IP Configuration

Interface: FastEthernet0

IP Configuration

○ DHCP ● Static

IPv4 Address: 192.168.1.10

Subnet Mask: 255.255.255.0

Default Gateway: 192.168.1.1

DNS Server: 0.0.0.0

IPv6 Configuration

○ Automatic ● Static

IPv6 Address: /

Link Local Address: FE80::20A:F3FF:FE50:3D0C

Default Gateway:

DNS Server:

802.1X

□ Use 802.1X Security

Authentication: MD5

Username:

Password:

□ Top

PC1 — Physical | Config | Desktop | Programming | Attributes

IP Configuration

Interface: FastEthernet0

IP Configuration

○ DHCP ● Static

IPv4 Address: 192.168.2.10

Subnet Mask: 255.255.255.0

Default Gateway: 192.168.2.1

DNS Server: 0.0.0.0

IPv6 Configuration

○ Automatic ● Static

IPv6 Address: /

Link Local Address: FE80::201:43FF:FED6:EBE1

Default Gateway:

DNS Server:

802.1X

□ Use 802.1X Security

Authentication: MD5

Username:

Password:

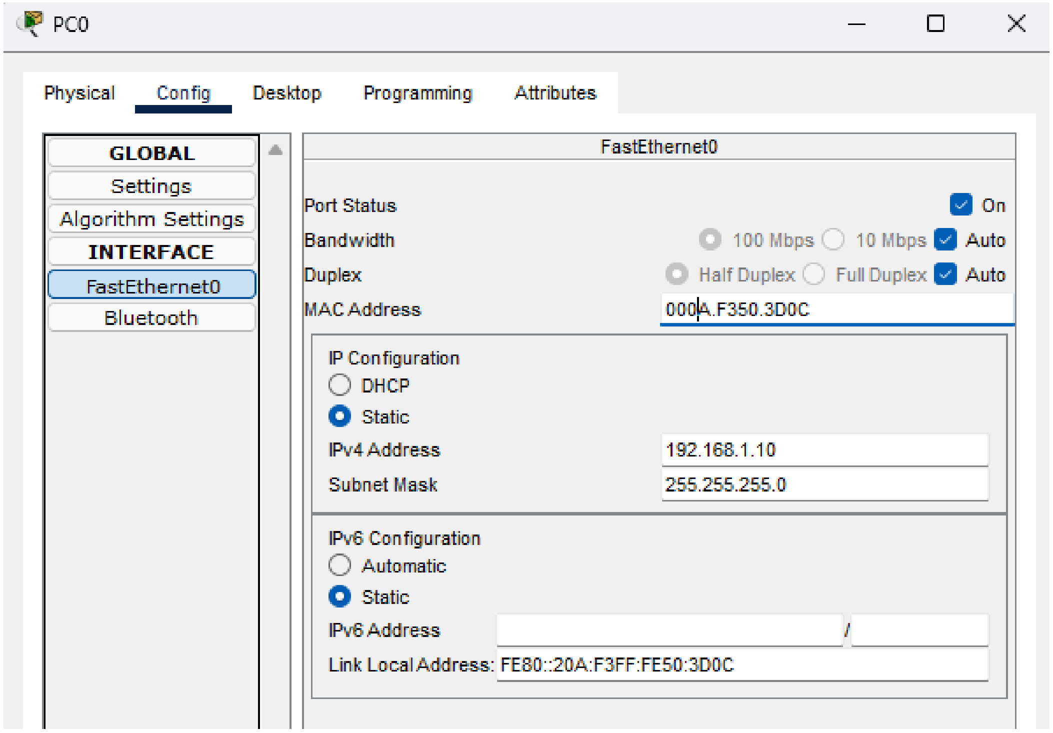
□ Top

Modify MAC address :

-Click the PC and go to the config tab and choose the interface

-Modify the mac address of the PCs in the field

-We can view the MAC address using ipconfig /all command



FastEthernet0

Port Status ☑ On

Bandwidth ○ 100 Mbps ○ 10 Mbps ☑ Auto

Duplex ○ Half Duplex ○ Full Duplex ☑ Auto

MAC Address: 000A.F350.3D0C

IP Configuration

○ DHCP

● Static

IPv4 Address: 192.168.1.10

Subnet Mask: 255.255.255.0

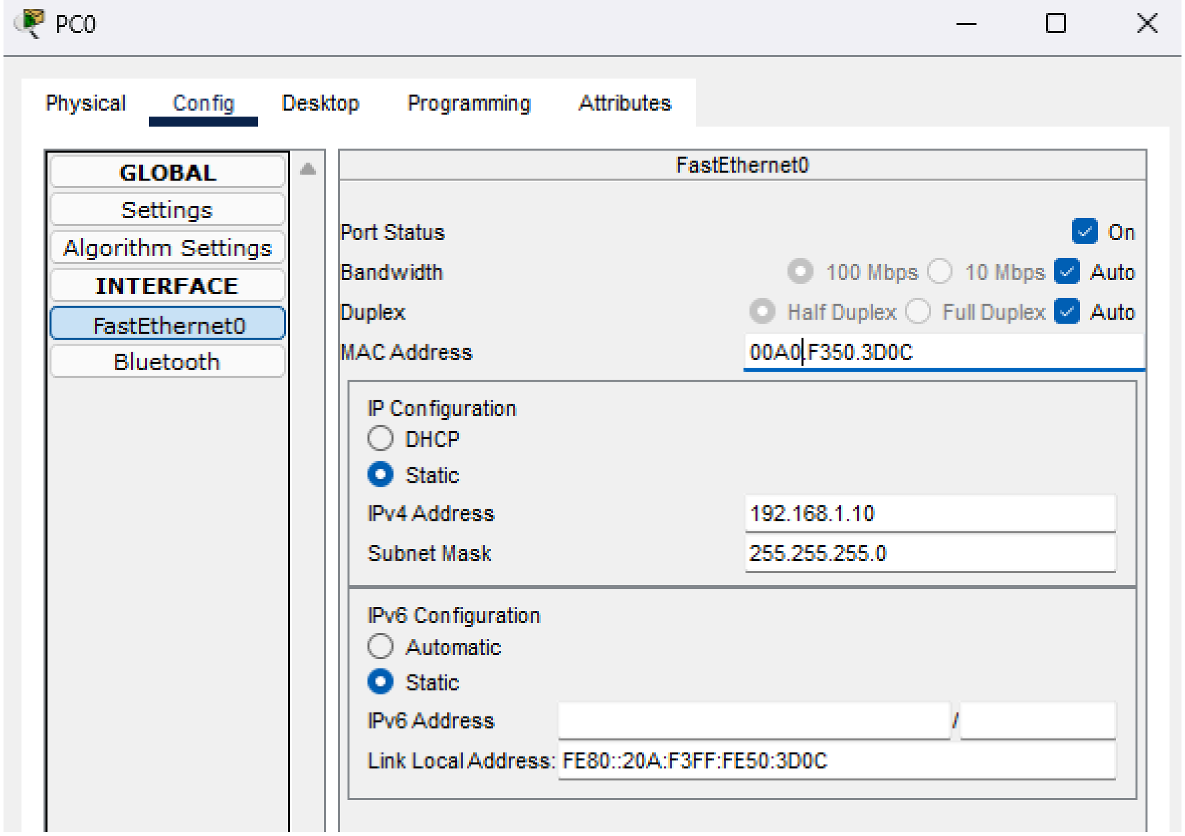
IPv6 Configuration

○ Automatic

● Static

IPv6 Address: /

Link Local Address: FE80::20A:F3FF:FE50:3D0C



FastEthernet0

Port Status ☑ On

Bandwidth ● 100 Mbps ○ 10 Mbps ☑ Auto

Duplex ● Half Duplex ○ Full Duplex ☑ Auto

MAC Address 00A0.F350.3D0C

IP Configuration

○ DHCP

● Static

IPv4 Address 192.168.1.10

Subnet Mask 255.255.255.0

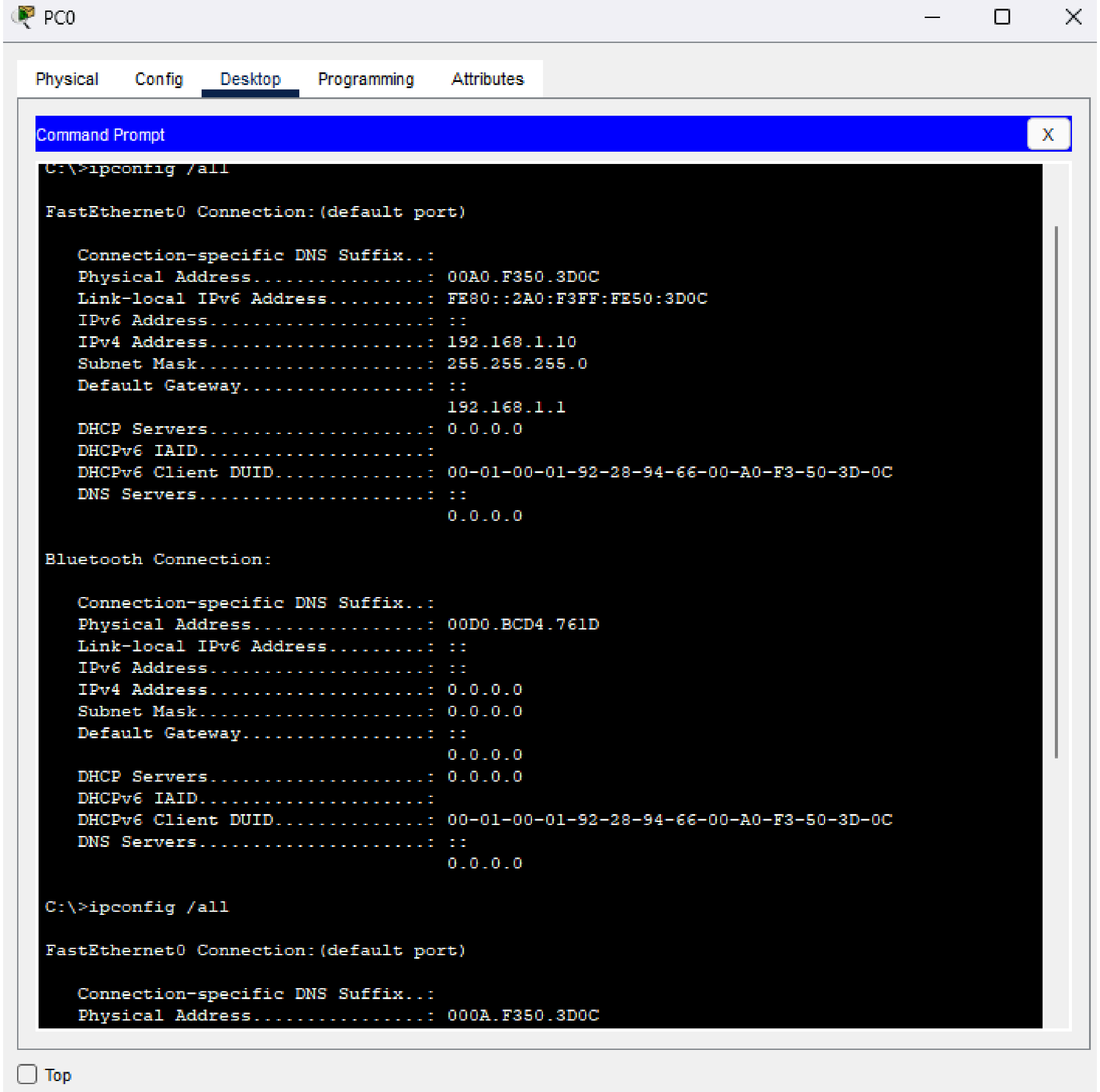
IPv6 Configuration

○ Automatic

● Static

IPv6 Address /

Link Local Address: FE80::20A:F3FF:FE50:3D0C

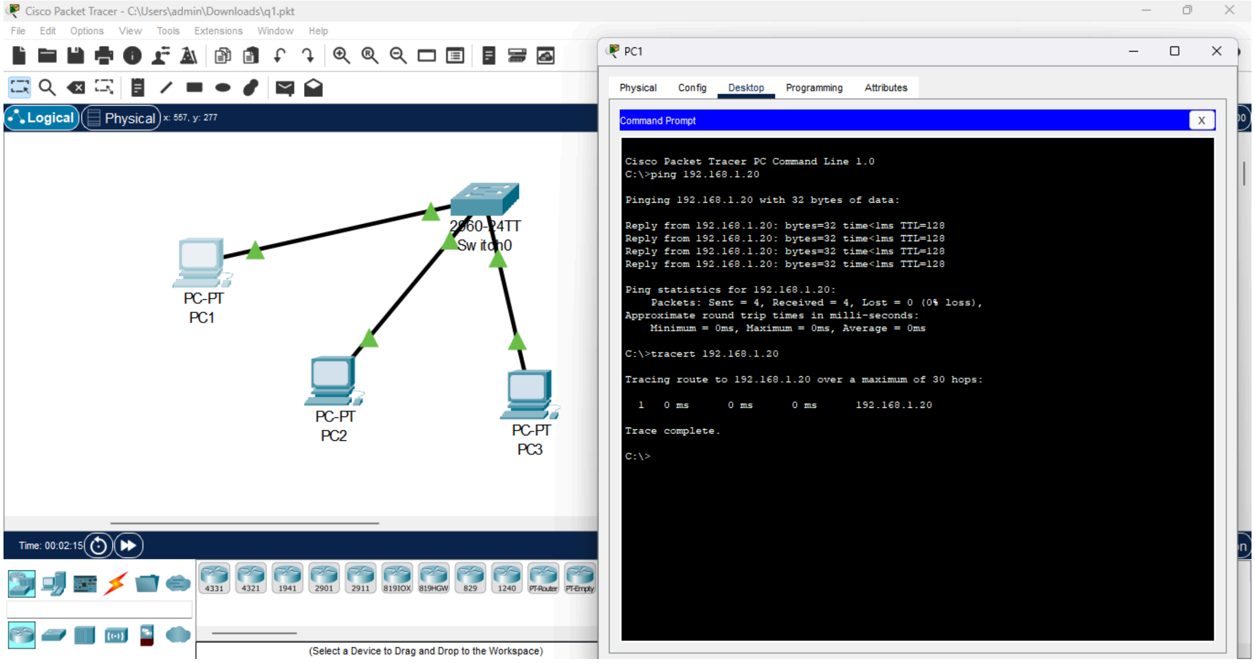


```
C:\>ipconfig /all

FastEthernet0 Connection:(default port)

   Connection-specific DNS Suffix..:
   Physical Address................: 00A0.F350.3D0C
   Link-local IPv6 Address.........: FE80::2A0:F3FF:FE50:3D0C
   IPv6 Address....................: ::
   IPv4 Address....................: 192.168.1.10
   Subnet Mask.....................: 255.255.255.0
   Default Gateway.................: ::
                                     192.168.1.1
   DHCP Servers....................: 0.0.0.0
   DHCPv6 IAID.....................:
   DHCPv6 Client DUID..............: 00-01-00-01-92-28-94-66-00-A0-F3-50-3D-0C
   DNS Servers.....................: ::
                                     0.0.0.0

Bluetooth Connection:

   Connection-specific DNS Suffix..:
   Physical Address................: 00D0.BCD4.761D
   Link-local IPv6 Address.........: ::
   IPv6 Address....................: ::
   IPv4 Address....................: 0.0.0.0
   Subnet Mask.....................: 0.0.0.0
   Default Gateway.................: ::
                                     0.0.0.0
   DHCP Servers....................: 0.0.0.0
   DHCPv6 IAID.....................:
   DHCPv6 Client DUID..............: 00-01-00-01-92-28-94-66-00-A0-F3-50-3D-0C
   DNS Servers.....................: ::
                                     0.0.0.0

C:\>ipconfig /all

FastEthernet0 Connection:(default port)

   Connection-specific DNS Suffix..:
   Physical Address................: 000A.F350.3D0C
```

Network Connectivity:
-Ping the pc's ip address in the command prompt and and check the reply packets

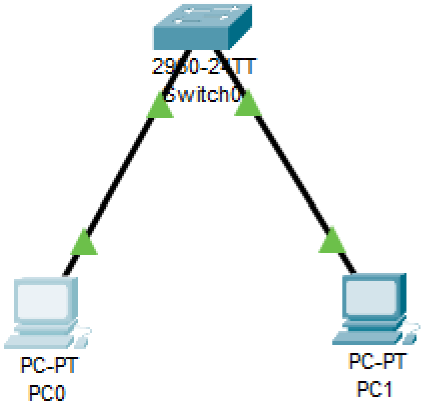
Q4. Troubleshoot Ethernet Communication with ping and traceroute -> Using cisco packet tracer:

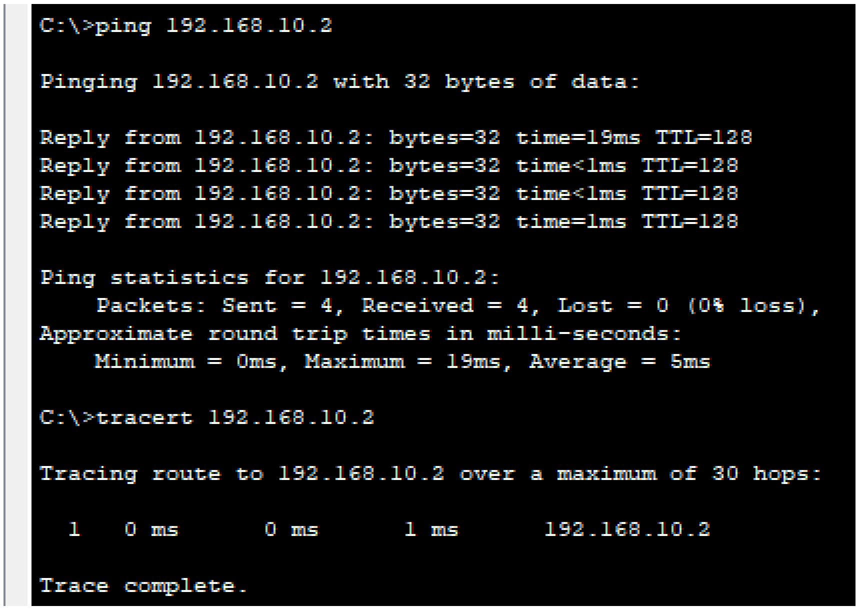
Checked IP configuration with ipconfig.
Tested connectivity using ping.
Traced packet routes with tracert.


Cisco Packet Tracer - C:\Users\admin\Downloads\q1.pkt
File Edit Options View Tools Extensions Window Help
Logical
Physical
x: 557, y: 277
2960-24TT
Switch0
PC-PT
PC1
PC-PT
PC2
PC-PT
PC3
Time: 00:02:15
4331 4321 1941 2901 2911 819IOX 819HGW 829 1240 PT-Router PT-Empty
(Select a Device to Drag and Drop to the Workspace)
PC1
Physical Config Desktop Programming Attributes
Command Prompt
X

```
Cisco Packet Tracer PC Command Line 1.0
C:\>ping 192.168.1.20

Pinging 192.168.1.20 with 32 bytes of data:

Reply from 192.168.1.20: bytes=32 time<1ms TTL=128
Reply from 192.168.1.20: bytes=32 time<1ms TTL=128
Reply from 192.168.1.20: bytes=32 time<1ms TTL=128
Reply from 192.168.1.20: bytes=32 time<1ms TTL=128

Ping statistics for 192.168.1.20:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 0ms, Average = 0ms

C:\>tracert 192.168.1.20

Tracing route to 192.168.1.20 over a maximum of 30 hops:

  1   0 ms      0 ms      0 ms      192.168.1.20

Trace complete.

C:\>
```

Q5. Create a simple LAN setup with two Linux machines connected via a switch. Ping from one machine to the other. If it fails, use ifconfig to ensure the IP addresses are configured correctly. Use traceroute to identify where the packets are being dropped if the ping fails.





```
C:\>ping 192.168.10.2

Pinging 192.168.10.2 with 32 bytes of data:

Reply from 192.168.10.2: bytes=32 time=19ms TTL=128
Reply from 192.168.10.2: bytes=32 time<1ms TTL=128
Reply from 192.168.10.2: bytes=32 time<1ms TTL=128
Reply from 192.168.10.2: bytes=32 time=1ms TTL=128

Ping statistics for 192.168.10.2:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 19ms, Average = 5ms

C:\>tracert 192.168.10.2

Tracing route to 192.168.10.2 over a maximum of 30 hops:

  1   0 ms     0 ms     1 ms     192.168.10.2

Trace complete.
```

Q6. Research the Linux kernel's handling of Ethernet devices and network interfaces. Write a short report on how the Linux kernel supports Ethernet communication (referencing kernel.org documentation).

The Linux kernel ensures smooth data transfer by working with network drivers, protocols, and system tools. Acting as a link between hardware and software, the kernel allows applications to send and receive data efficiently, ensuring a stable network connection.

It uses layered architecture :

1. **Physical Layer** – Responsible for transmitting raw data over physical cables and Network Interface Cards (NICs).
2. **Data Link Layer** – Manages data from the physical layer using device drivers and identifies the destination using MAC addresses.
3. **Network Layer** (IPv4/IPv6 Routing) – Routes data based on the MAC table and IP addresses. The ARP table assists in mapping IPs to MAC addresses.
4. **Transport Layer** – Uses TCP/UDP protocols for packet transport, handling flow control, congestion control, and error management. It also breaks requests into smaller packets for transmission.
5. **Application Layer** – Supports user programs that use system calls like send(), receive(), and socket() to transmit data over the network.

- When an application sends data, it goes through the socket API, where it is turned into packets following network rules.

- The Ethernet driver works with the Network Interface Card to send these packets over the network.
- When a packet arrives, the NIC detects it and passes it to the Ethernet driver, which sends it to the kernel.
- The kernel processes the packet, extracts the needed data, and delivers it to the right application using the socket API.
- This process ensures smooth and reliable network communication.

Q7. Describe how you would configure a basic LAN interface using the ip command in Linux (kernel.org).

1. **Check for interfaces:**

We need to check for all the available interfaces, using the command:

```
ip link show
```

2. **Check whether the interface is up or down:**

We can check whether the required interface like enpos3,eth0 etc. are up or down. If it is down we have to make that active, by giving the command

```
sudo ip link set enp0s3 up
```

3. **Assign IP statically:**

Next, I configure the network interface with a specific IP address. In this case, I assign 192.168.1.20 with a subnet mask of 255.255.255.0 using the following command:

```
ip addr add 192.168.1.20/24 dev enp0s3
```

4. **Default gateway:**

We can set the default gateway (e.g., 192.168.1.1) by using the command,

```
sudo ip route add default via 192.168.1.1
```

5. **Verification :**

To check if the IP address and routes are set correctly, I used the following commands,

```
ip addr show enp0s3
ip route show
```

6. **Connectivity:**

The connectivity can be checked using the ping command, which is:

```
ping 192.168.1.1
```

This setup ensures the LAN interface is properly configured for communication within the network

Q8. Use Linux to view the MAC address table of a switch (if using a Linux-based network switch). Use the bridge or ip link commands to inspect the MAC table and demonstrate a basic switch's operation.

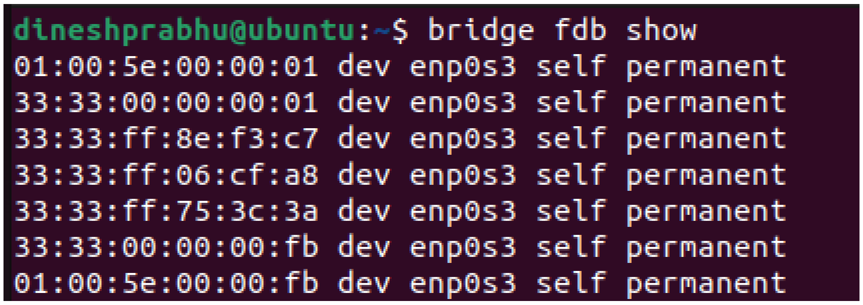
A MAC address table is used by network switches to store MAC addresses and their corresponding interfaces.

When a switch receives a frame, it checks the source MAC address and associates it with the incoming port. The switch adds or updates this entry in the MAC address table.

- If the destination MAC address is in the table, the switch forwards the frame to the correct port.
- If the MAC address is unknown, the switch sends the frame to all ports except the incoming one.

MAC table entries expire after a certain period if no traffic is received from that MAC address. This prevents stale entries and allows the table to update dynamically.

To display the MAC address table:



```
dineshprabhu@ubuntu:~$ bridge fdb show
01:00:5e:00:00:01 dev enp0s3 self permanent
33:33:00:00:00:01 dev enp0s3 self permanent
33:33:ff:8e:f3:c7 dev enp0s3 self permanent
33:33:ff:06:cf:a8 dev enp0s3 self permanent
33:33:ff:75:3c:3a dev enp0s3 self permanent
33:33:00:00:00:fb dev enp0s3 self permanent
01:00:5e:00:00:fb dev enp0s3 self permanent
```