

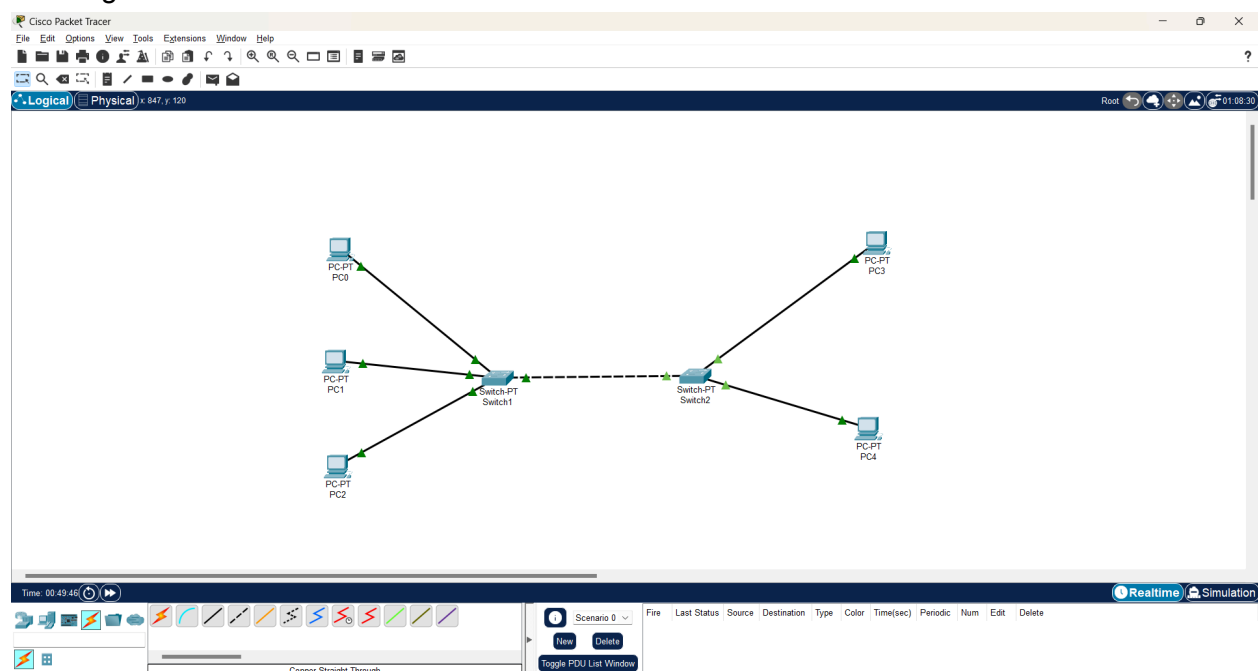


Networking Assessment 2

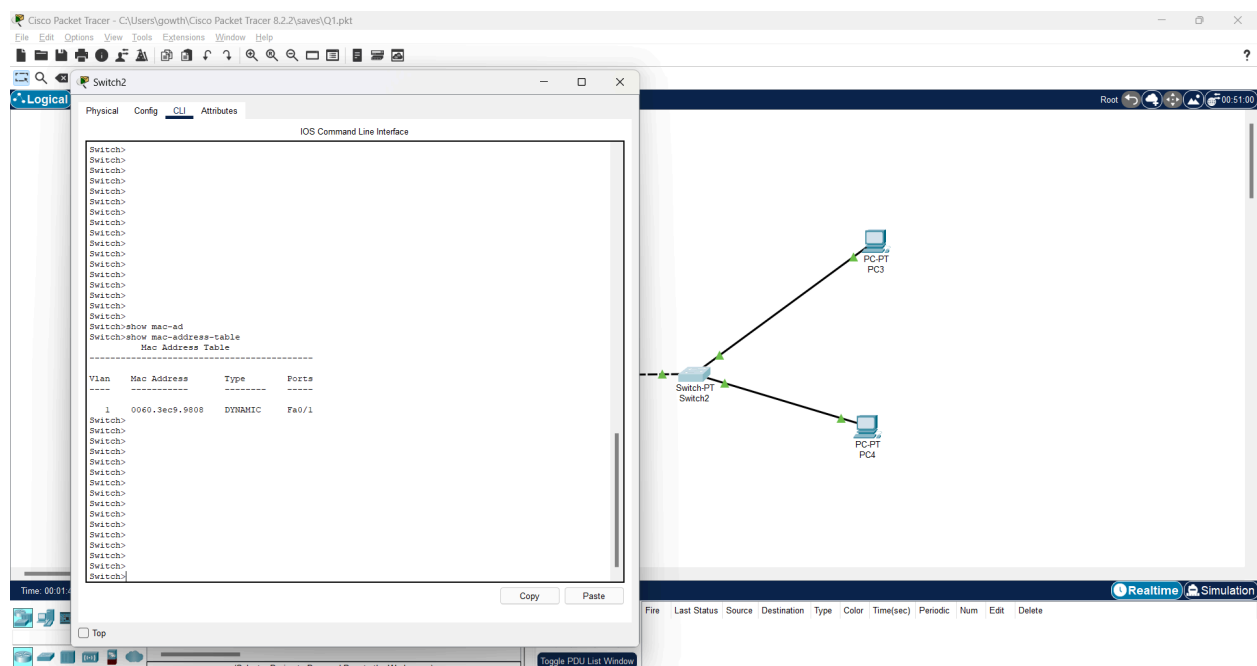
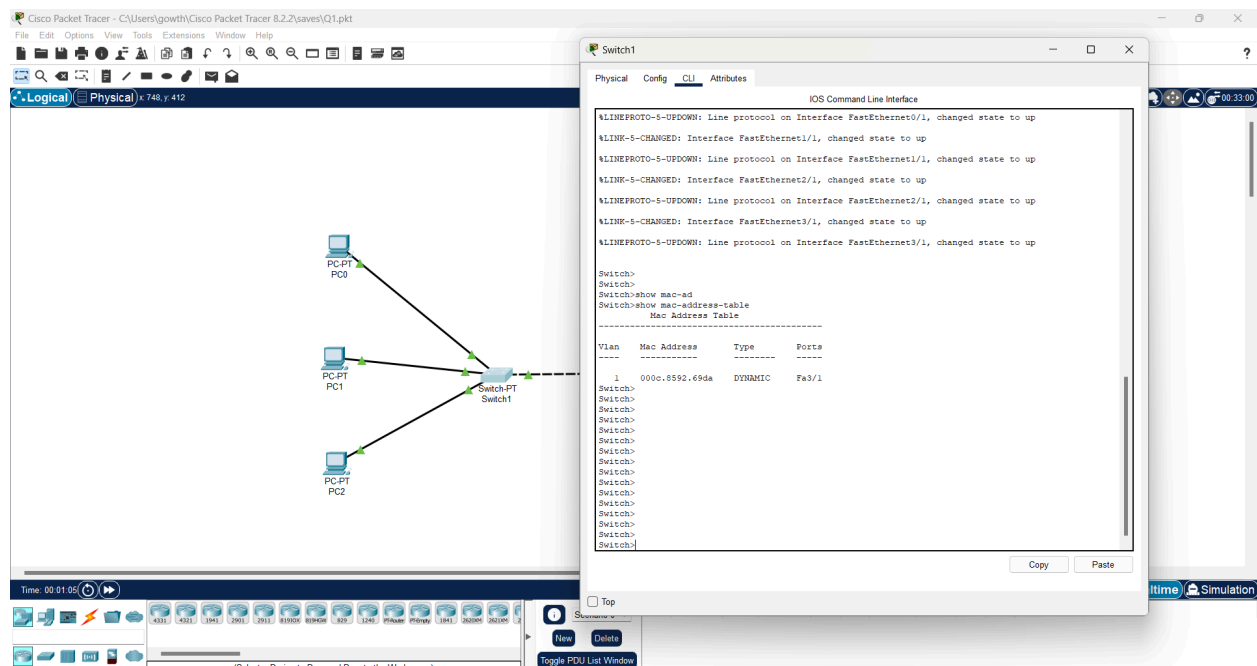
Name	T K Gowtham
Email ID	gowthamkamalasekar@gmail.com
College	VIT Chennai

1. 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.

Creating a network :



Current Mac Address Before Pinging on both routers :



The screenshot displays the Cisco Packet Tracer interface. On the left, a network topology is shown with three PCs (PC-PT 001, PC-PT 002, PC-PT 003) connected to a central Switch (Switch-PT 001). The PCs are connected to the switch via Ethernet cables. The interface includes a top menu bar with File, Edit, Options, View, Tools, Extensions, Windows, and Help. Below the menu is a toolbar with various icons for network components and actions. The main workspace shows the network diagram. At the bottom, there is a status bar with a timer (Time 00:02:41) and a toolbar with icons for simulation and other functions.

On the right, a terminal window titled "PC0" is open, showing the output of a series of ping commands. The terminal output is as follows:

```

C:\>ping 192.168.1.11

Pinging 192.168.1.11 with 32 bytes of data:

Reply from 192.168.1.11: bytes=32 time=1ms TTL=128
Reply from 192.168.1.11: bytes=32 time=1ms TTL=128
Reply from 192.168.1.11: bytes=32 time=1ms TTL=128
Reply from 192.168.1.11: bytes=32 time=1ms TTL=128

Ping statistics for 192.168.1.11:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 0ms, Maximum = 0ms, Average = 0ms

C:\>ping 192.168.1.12

Pinging 192.168.1.12 with 32 bytes of data:

Reply from 192.168.1.12: bytes=32 time=1ms TTL=128
Reply from 192.168.1.12: bytes=32 time=1ms TTL=128
Reply from 192.168.1.12: bytes=32 time=1ms TTL=128
Reply from 192.168.1.12: bytes=32 time=1ms TTL=128

Ping statistics for 192.168.1.12:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 0ms, Maximum = 0ms, Average = 0ms

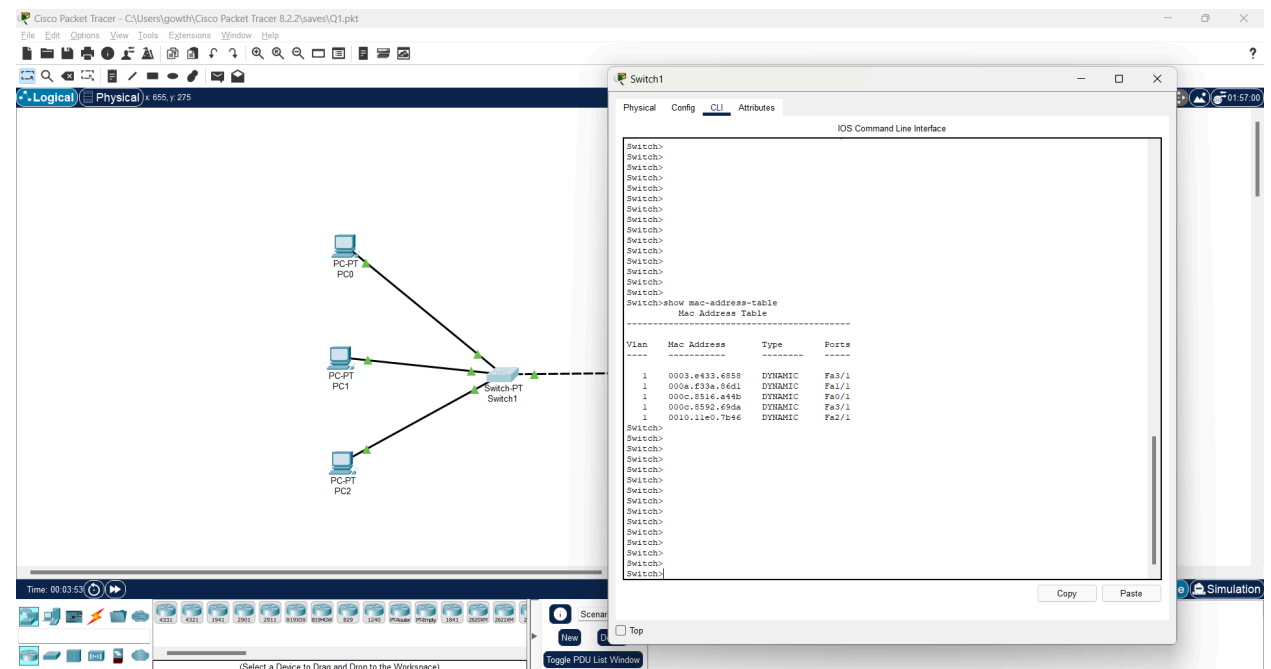
C:\>ping 192.168.1.13

Pinging 192.168.1.13 with 32 bytes of data:

Reply from 192.168.1.13: bytes=32 time=1ms TTL=128
Reply from 192.168.1.13: bytes=32 time=1ms TTL=128
Reply from 192.168.1.13: bytes=32 time=1ms TTL=128
Reply from 192.168.1.13: bytes=32 time=1ms TTL=128

Ping statistics for 192.168.1.13:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 0ms, Maximum = 0ms, Average = 0ms

C:\>
  
```



Using GNS3, with wireshark to capture packets :

The screenshot shows the GNS3 interface with a network topology. A central switch, Layer2Switch-1, is connected to two PCs, PC1 and PC2. The console window displays the following output:

```
PC1> ping 192.168.1.11
84 bytes from 192.168.1.11 icmp_seq=1 ttl=64 time=14.086 ms
84 bytes from 192.168.1.11 icmp_seq=2 ttl=64 time=2.032 ms
84 bytes from 192.168.1.11 icmp_seq=3 ttl=64 time=4.803 ms
84 bytes from 192.168.1.11 icmp_seq=4 ttl=64 time=1.676 ms
84 bytes from 192.168.1.11 icmp_seq=5 ttl=64 time=3.264 ms
PC1>
```

The screenshot shows the Wireshark interface with a packet capture of an ICMP Echo (ping) request and reply. The packet list shows the following details:

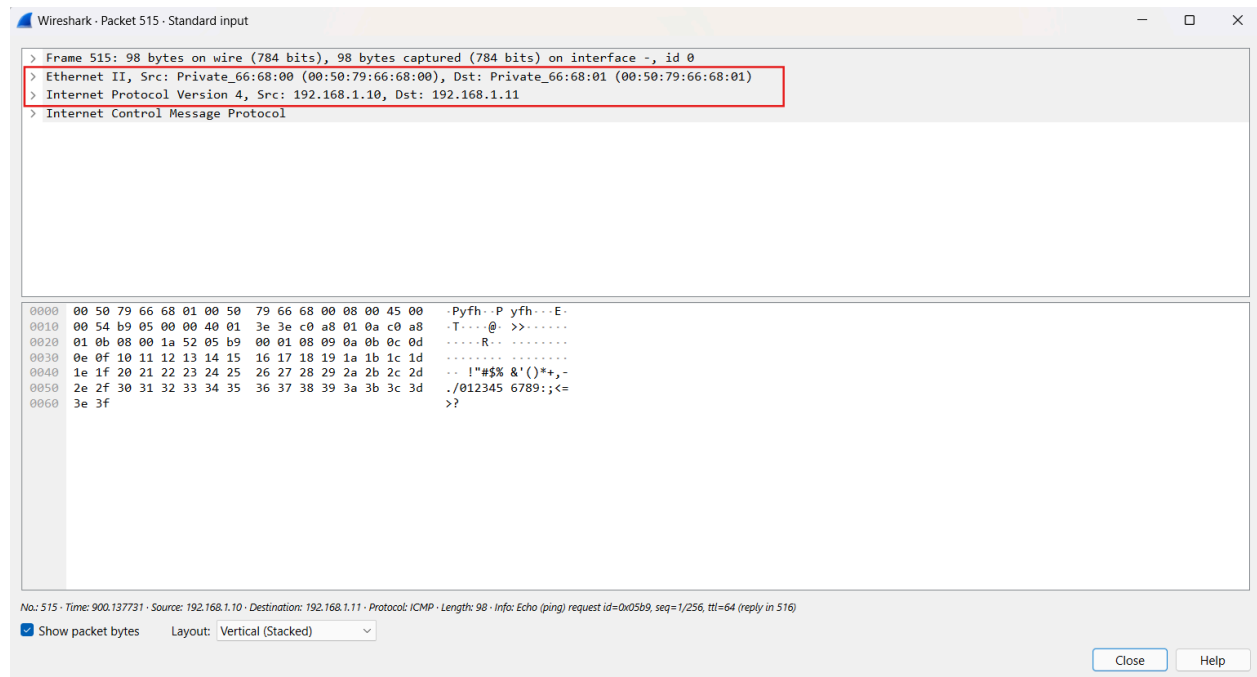
No.	Time	Source	Destination	Protocol	Length	Info
512	909.080271	0c:b6:cf:1b:00:00	Spanning-tree (for...	STP	60	Conf. Root = 32768/1/0c:b6:cf:1b:00:00 Cost = 0 Port = 0x8001
513	900.108197	Private_66:68:00	Broadcast	ARP	64	Who has 192.168.1.11? Tell 192.168.1.10
514	900.126224	Private_66:68:01	Private_66:68:00	ARP	64	192.168.1.11 is at 00:50:79:66:68:01
515	900.137731	192.168.1.10	192.168.1.11	ICMP	98	Echo (ping) request id=0x05b9, seq=1/256, ttl=64 (reply in 516)
516	900.151528	192.168.1.11	192.168.1.10	ICMP	98	Echo (ping) reply id=0x05b9, seq=1/256, ttl=64 (request in 515)
517	900.355076	0c:b6:cf:1b:00:00	0c:b6:cf:1b:00:00	LOOP	60	Reply
518	901.165623	192.168.1.10	192.168.1.11	ICMP	98	Echo (ping) request id=0x06b9, seq=2/512, ttl=64 (reply in 519)
519	901.167259	192.168.1.11	192.168.1.10	ICMP	98	Echo (ping) reply id=0x06b9, seq=2/512, ttl=64 (request in 518)
520	901.679416	0c:b6:cf:1b:00:00	Spanning-tree (for...	STP	60	Conf. Root = 32768/1/0c:b6:cf:1b:00:00 Cost = 0 Port = 0x8001
521	902.191374	192.168.1.10	192.168.1.11	ICMP	98	Echo (ping) request id=0x07b9, seq=3/768, ttl=64 (reply in 522)
522	902.195063	192.168.1.11	192.168.1.10	ICMP	98	Echo (ping) reply id=0x07b9, seq=3/768, ttl=64 (request in 521)
523	903.220649	192.168.1.10	192.168.1.11	ICMP	98	Echo (ping) request id=0x08b9, seq=4/1024, ttl=64 (reply in 524)
524	903.222054	192.168.1.11	192.168.1.10	ICMP	98	Echo (ping) reply id=0x08b9, seq=4/1024, ttl=64 (request in 523)
525	903.793714	0c:b6:cf:1b:00:00	Spanning-tree (for...	STP	60	Conf. Root = 32768/1/0c:b6:cf:1b:00:00 Cost = 0 Port = 0x8001
526	904.246581	192.168.1.10	192.168.1.11	ICMP	98	Echo (ping) request id=0x09b9, seq=5/1280, ttl=64 (reply in 527)
527	904.249541	192.168.1.11	192.168.1.10	ICMP	98	Echo (ping) reply id=0x09b9, seq=5/1280, ttl=64 (request in 526)
528	905.876549	0c:b6:cf:1b:00:00	Spanning-tree (for...	STP	60	Conf. Root = 32768/1/0c:b6:cf:1b:00:00 Cost = 0 Port = 0x8001

The packet details pane shows the following information:

- Frame 515: 98 bytes on wire (784 bits), 98 bytes captured (784 bits) on interface -, id 0
- Ethernet II, Src: Private_66:68:00 (00:50:79:66:68:00), Dst: Private_66:68:01 (00:50:79:66:68:01)
- Internet Protocol Version 4, Src: 192.168.1.10, Dst: 192.168.1.11
- Internet Control Message Protocol

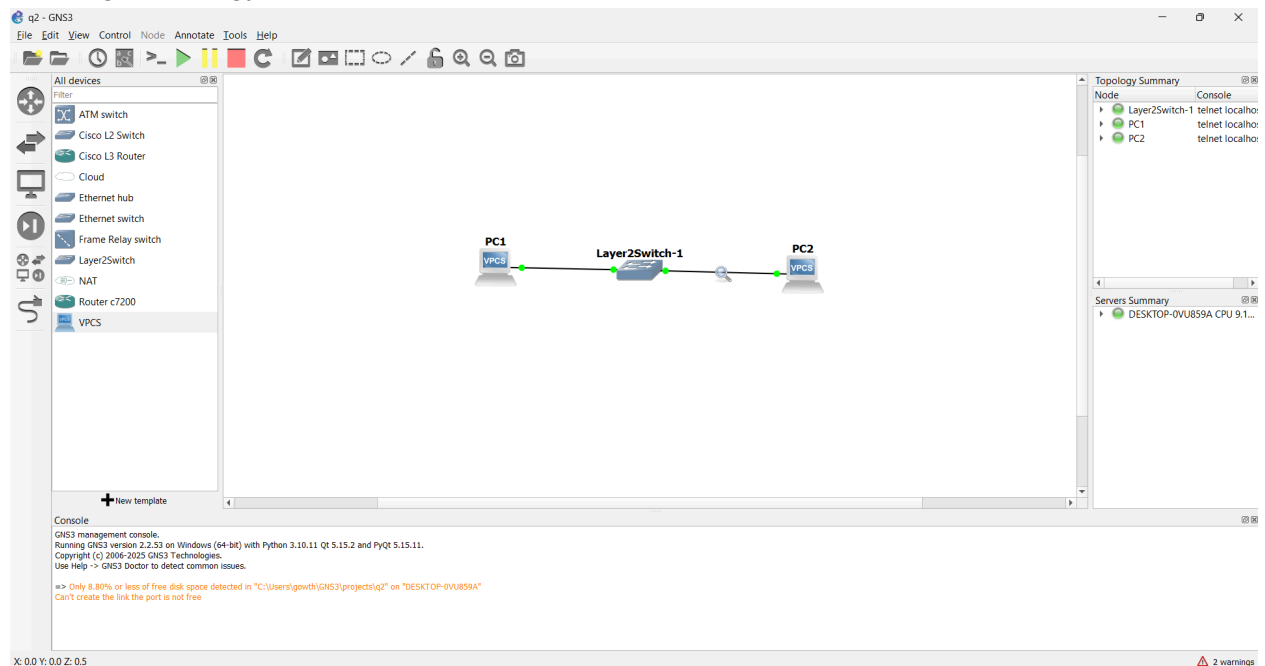
The packet bytes pane shows the raw data of the ICMP Echo request and reply.

We can see the Ethernet Frame and MAC Addressing as we analyse the packet :

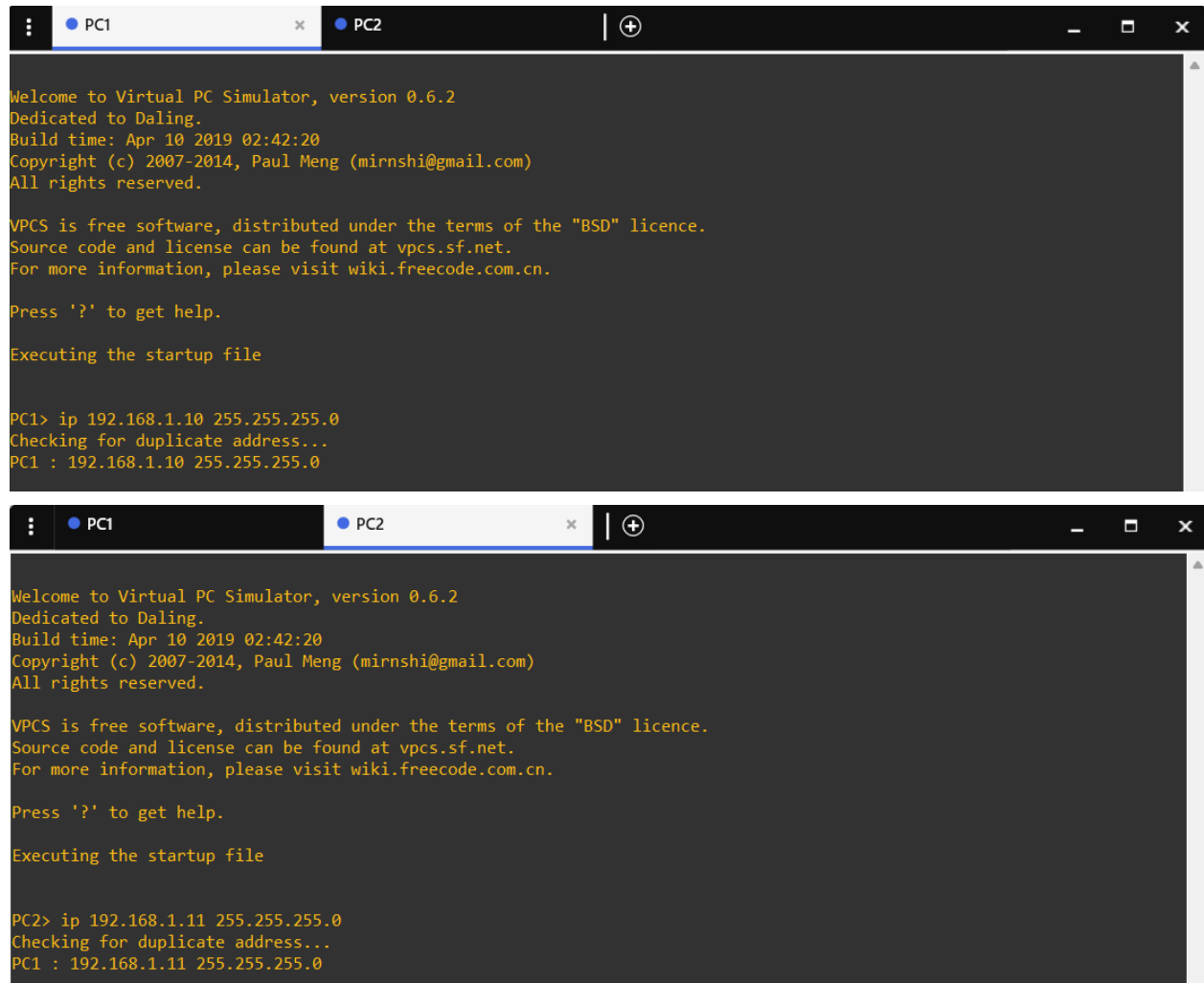


2. Capture and analyze Ethernet frames using Wireshark. Inspect the structure of the frame, including destination and source MAC addresses, Ethertype, payload, and FCS. Use GNS3 or Packet Tracer to simulate network traffic.

Creating a topology in GNS3 :



Setting IP address for PC1 and PC2 :



```
Welcome to Virtual PC Simulator, version 0.6.2
Dedicated to Daling.
Build time: Apr 10 2019 02:42:20
Copyright (c) 2007-2014, Paul Meng (mirnshi@gmail.com)
All rights reserved.

VPCS is free software, distributed under the terms of the "BSD" licence.
Source code and license can be found at vpcs.sf.net.
For more information, please visit wiki.freecode.com.cn.

Press '?' to get help.

Executing the startup file

PC1> ip 192.168.1.10 255.255.255.0
Checking for duplicate address...
PC1 : 192.168.1.10 255.255.255.0

Welcome to Virtual PC Simulator, version 0.6.2
Dedicated to Daling.
Build time: Apr 10 2019 02:42:20
Copyright (c) 2007-2014, Paul Meng (mirnshi@gmail.com)
All rights reserved.

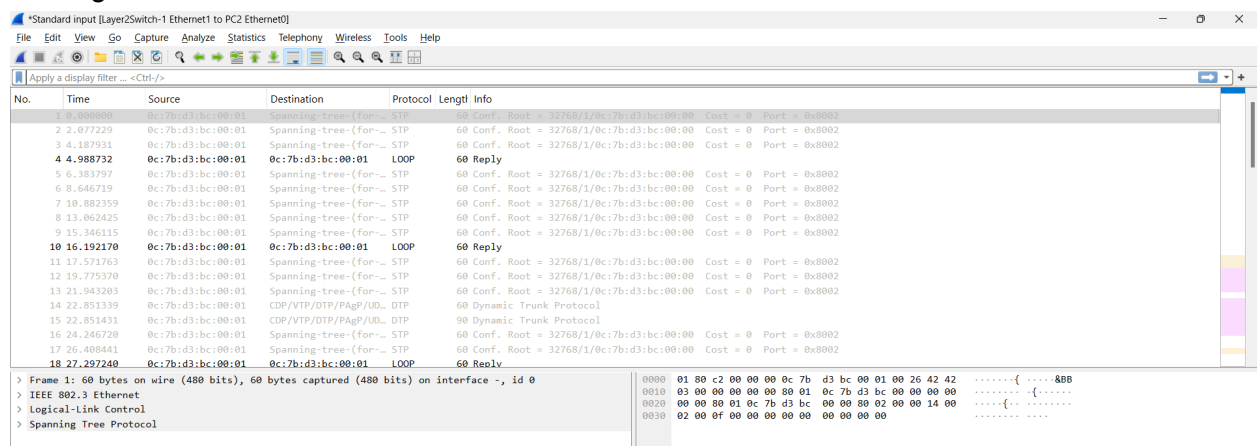
VPCS is free software, distributed under the terms of the "BSD" licence.
Source code and license can be found at vpcs.sf.net.
For more information, please visit wiki.freecode.com.cn.

Press '?' to get help.

Executing the startup file

PC2> ip 192.168.1.11 255.255.255.0
Checking for duplicate address...
PC1 : 192.168.1.11 255.255.255.0
```

Starting Wireshark :



No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	0c:7b:d3:bc:00:01	Spanning-tree-for-...	STP	60	Conf. Root = 32768/1/0c:7b:d3:bc:00:00 Cost = 0 Port = 0x8002
2	2.077229	0c:7b:d3:bc:00:01	Spanning-tree-for-...	STP	60	Conf. Root = 32768/1/0c:7b:d3:bc:00:00 Cost = 0 Port = 0x8002
3	4.187931	0c:7b:d3:bc:00:01	Spanning-tree-for-...	STP	60	Conf. Root = 32768/1/0c:7b:d3:bc:00:00 Cost = 0 Port = 0x8002
4	4.988732	0c:7b:d3:bc:00:01	0c:7b:d3:bc:00:01	LOOP	60	Reply
5	6.383797	0c:7b:d3:bc:00:01	Spanning-tree-for-...	STP	60	Conf. Root = 32768/1/0c:7b:d3:bc:00:00 Cost = 0 Port = 0x8002
6	8.646719	0c:7b:d3:bc:00:01	Spanning-tree-for-...	STP	60	Conf. Root = 32768/1/0c:7b:d3:bc:00:00 Cost = 0 Port = 0x8002
7	10.882359	0c:7b:d3:bc:00:01	Spanning-tree-for-...	STP	60	Conf. Root = 32768/1/0c:7b:d3:bc:00:00 Cost = 0 Port = 0x8002
8	13.062425	0c:7b:d3:bc:00:01	Spanning-tree-for-...	STP	60	Conf. Root = 32768/1/0c:7b:d3:bc:00:00 Cost = 0 Port = 0x8002
9	15.346115	0c:7b:d3:bc:00:01	Spanning-tree-for-...	STP	60	Conf. Root = 32768/1/0c:7b:d3:bc:00:00 Cost = 0 Port = 0x8002
10	16.192170	0c:7b:d3:bc:00:01	0c:7b:d3:bc:00:01	LOOP	60	Reply
11	17.571763	0c:7b:d3:bc:00:01	Spanning-tree-for-...	STP	60	Conf. Root = 32768/1/0c:7b:d3:bc:00:00 Cost = 0 Port = 0x8002
12	19.775370	0c:7b:d3:bc:00:01	Spanning-tree-for-...	STP	60	Conf. Root = 32768/1/0c:7b:d3:bc:00:00 Cost = 0 Port = 0x8002
13	21.943203	0c:7b:d3:bc:00:01	Spanning-tree-for-...	STP	60	Conf. Root = 32768/1/0c:7b:d3:bc:00:00 Cost = 0 Port = 0x8002
14	22.851339	0c:7b:d3:bc:00:01	CDP/VTP/DTP/PAGP/UDL	DTP	60	Dynamic Trunk Protocol
15	22.851431	0c:7b:d3:bc:00:01	CDP/VTP/DTP/PAGP/UDL	DTP	90	Dynamic Trunk Protocol
16	24.246720	0c:7b:d3:bc:00:01	Spanning-tree-for-...	STP	60	Conf. Root = 32768/1/0c:7b:d3:bc:00:00 Cost = 0 Port = 0x8002
17	25.808441	0c:7b:d3:bc:00:01	Spanning-tree-for-...	STP	60	Conf. Root = 32768/1/0c:7b:d3:bc:00:00 Cost = 0 Port = 0x8002
18	27.297240	0c:7b:d3:bc:00:01	0c:7b:d3:bc:00:01	LOOP	60	Reply

> Frame 1: 60 bytes on wire (480 bits), 60 bytes captured (480 bits) on interface -, id 0

> IEEE 802.3 Ethernet

> Logical-Link Control

> Spanning Tree Protocol

```
0000 01 80 c2 00 00 0c 7b d3 bc 00 01 00 26 42 42 .....8BB
0010 03 00 00 00 00 80 01 0c 7b d3 bc 00 00 00 00 .....-{}.....
0020 00 00 80 01 0c 7b d3 bc 00 00 02 00 00 14 00 .....{.....
0030 02 00 0f 00 00 00 00 00 00 00 00 00 00 00 00 .....

```

Pinging from PC1 to PC2 :

```

Welcome to Virtual PC Simulator, version 0.6.2
Dedicated to Daling.
Build time: Apr 10 2019 02:42:20
Copyright (c) 2007-2014, Paul Meng (mirnshi@gmail.com)
All rights reserved.

VPCS is free software, distributed under the terms of the "BSD" licence.
Source code and license can be found at vpcs.sf.net.
For more information, please visit wiki.freecode.com.cn.

Press '?' to get help.

Executing the startup file

PC1> ip 192.168.1.10 255.255.255.0
Checking for duplicate address...
PC1 : 192.168.1.10 255.255.255.0

PC1> ping 192.168.1.11
84 bytes from 192.168.1.11 icmp_seq=1 ttl=64 time=2.299 ms
84 bytes from 192.168.1.11 icmp_seq=2 ttl=64 time=4.616 ms
84 bytes from 192.168.1.11 icmp_seq=3 ttl=64 time=12.974 ms
84 bytes from 192.168.1.11 icmp_seq=4 ttl=64 time=1.969 ms
84 bytes from 192.168.1.11 icmp_seq=5 ttl=64 time=10.806 ms

PC1>
```

Capturing Packet :

The image shows a Wireshark packet capture window titled "*Standard input [Layer2Switch-1 Ethernet1 to PC2 Ethernet0]". The interface includes a menu bar (File, Edit, View, Go, Capture, Analyze, Statistics, Telephony, Wireless, Tools, Help) and a toolbar. A display filter is set to "<Ctrl>".

No.	Time	Source	Destination	Protocol	Length	Info
26	41.862930	0c:7b:d3:bc:00:01	Spanning-tree (for...	STP	60	Conf. Root = 32768/1/0c:7b:d3:bc:00:00 Cost = 0 Port = 0x8002
27	44.999427	0c:7b:d3:bc:00:01	Spanning-tree (for...	STP	60	Conf. Root = 32768/1/0c:7b:d3:bc:00:00 Cost = 0 Port = 0x8002
28	44.383766	Private_66:68:00	Broadcast	ARP	64	Who has 192.168.1.11? Tell 192.168.1.10
29	44.384991	Private_66:68:01	Private_66:68:00	ARP	64	192.168.1.11 is at 00:50:79:66:68:01
30	44.399844	192.168.1.10	192.168.1.11	ICMP	98	Echo (ping) request id=0x2df9, seq=1/256, ttl=64 (reply in 31)
31	44.399556	192.168.1.11	192.168.1.10	ICMP	98	Echo (ping) reply id=0x2df9, seq=1/256, ttl=64 (request in 30)
32	45.425387	192.168.1.10	192.168.1.11	ICMP	98	Echo (ping) request id=0x2ef9, seq=2/512, ttl=64 (reply in 33)
33	45.425788	192.168.1.11	192.168.1.10	ICMP	98	Echo (ping) reply id=0x2ef9, seq=2/512, ttl=64 (request in 32)
34	46.311536	0c:7b:d3:bc:00:01	Spanning-tree (for...	STP	60	Conf. Root = 32768/1/0c:7b:d3:bc:00:00 Cost = 0 Port = 0x8002
35	46.457264	192.168.1.10	192.168.1.11	ICMP	98	Echo (ping) request id=0x2ff9, seq=3/768, ttl=64 (reply in 36)
36	46.457695	192.168.1.11	192.168.1.10	ICMP	98	Echo (ping) reply id=0x2ff9, seq=3/768, ttl=64 (request in 35)
37	47.479332	192.168.1.10	192.168.1.11	ICMP	98	Echo (ping) request id=0x30f9, seq=4/1024, ttl=64 (reply in 38)
38	47.479682	192.168.1.11	192.168.1.10	ICMP	98	Echo (ping) reply id=0x30f9, seq=4/1024, ttl=64 (request in 37)
39	48.511323	192.168.1.10	192.168.1.11	ICMP	98	Echo (ping) request id=0x31f9, seq=5/1280, ttl=64 (reply in 40)
40	48.511607	192.168.1.11	192.168.1.10	ICMP	98	Echo (ping) reply id=0x31f9, seq=5/1280, ttl=64 (request in 39)
41	48.534549	0c:7b:d3:bc:00:01	Spanning-tree (for...	STP	60	Conf. Root = 32768/1/0c:7b:d3:bc:00:00 Cost = 0 Port = 0x8002
42	49.394846	0c:7b:d3:bc:00:01	0c:7b:d3:bc:00:01	LOOP	60	Reply
43	49.810997	0c:7b:d3:bc:00:01	CDP/NTP/DTP/PAeP/UD...	CDP	428	Device ID: v105-L2-01 Port ID: GigabitEthernet0/1

Below the packet list, the details pane shows the structure of the selected packet (No. 30):

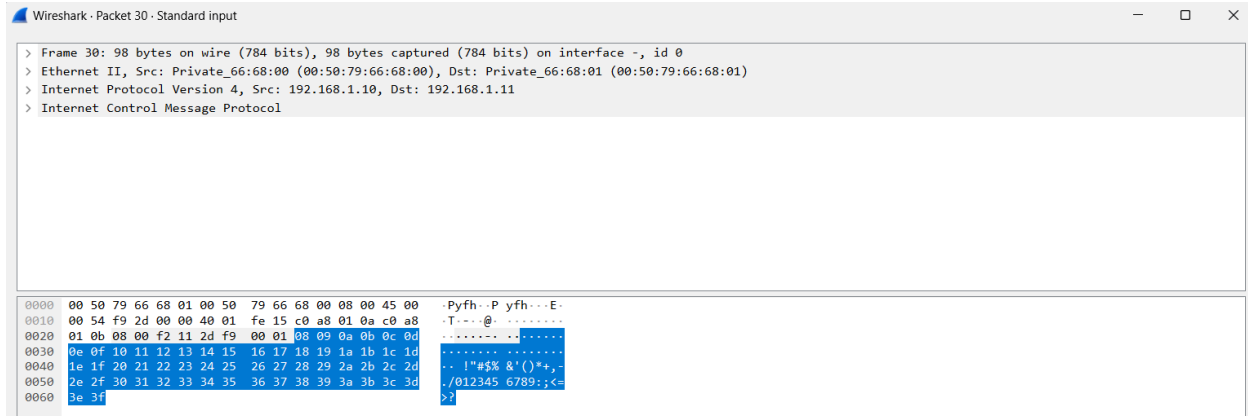
- > Frame 30: 98 bytes on wire (784 bits), 98 bytes captured (784 bits) on interface "...", id 0
- > Ethernet II, Src: Private_66:68:00 (00:50:79:66:68:00), Dst: Private_66:68:01 (00:50:79:66:68:01)
- > Internet Protocol Version 4, Src: 192.168.1.10, Dst: 192.168.1.11
- > Internet Control Message Protocol

The packet bytes pane shows the raw data in hexadecimal and ASCII:

```

0000  00 50 79 66 68 01 00 50 79 66 68 00 08 00 45 00  :Pyfh..P yfh...E:
0010  00 54 f9 2d 00 00 40 01 fe 15 c0 a8 01 0a c0 a8  :.T...@.....
0020  01 0b 08 00 f2 11 2d f9 00 01 08 09 0a 0b 0c 0d  :.....
0030  0e 0f 10 11 12 13 14 15 16 17 18 19 1a 1b 1c 1d  :.....
0040  1e 1f 20 21 22 23 24 25 26 27 28 29 2a 2b 2c 2d  :...!"%&'()*+,-./012345 6789;<=
0050  2e 2f 30 31 32 33 34 35 36 37 38 39 3a 3b 3c 3d  :...
0060  3e 3f                                     :>
```

Now let's inspect the structure :



Frame : It states the whole packet, 30 is the frame number in the captured sequence. The total size of the frame is 98 bytes long. ID 0 is the interface id. It contains interface id, encapsulation type which is ethernet, arrival time (also UTC and epoch)

Ethertype :

- ✓ Ethernet II, Src: Private_66:68:00 (00:50:79:66:68:00), Dst: Private_66:68:01 (00:50:79:66:68:01)
 - ✓ Destination: Private_66:68:01 (00:50:79:66:68:01)
 -0. = LG bit: Globally unique address (factory default)
 -0. = IG bit: Individual address (unicast)
 - ✓ Source: Private_66:68:00 (00:50:79:66:68:00)
 -0. = LG bit: Globally unique address (factory default)
 -0. = IG bit: Individual address (unicast)
- Type: IPv4 (0x0800)
- [Stream index: 4]

Consist of the source and destination MAC address. For each of it, it also specifies the Globally unique address set by the Manufacturer like Cisco or Juniper and the Individual address which is unique and either set automatically or by the network administrator.

IPv4 :

- ✓ Internet Protocol Version 4, Src: 192.168.1.10, Dst: 192.168.1.11
 - 0100 = Version: 4
 - 0101 = Header Length: 20 bytes (5)
 - > Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
 - Total Length: 84
 - Identification: 0xf92d (63789)
 - > 000. = Flags: 0x0
 - ...0 0000 0000 0000 = Fragment Offset: 0
 - Time to Live: 64
 - Protocol: ICMP (1)
 - Header Checksum: 0xfe15 [validation disabled]
 - [Header checksum status: Unverified]
 - Source Address: 192.168.1.10

Consist of the details needed for the networking layer such as the source and destination IP address. It also states IP version, header length, flags, fragmentation offset, TTL, protocol used, Checksum.

ICMP :

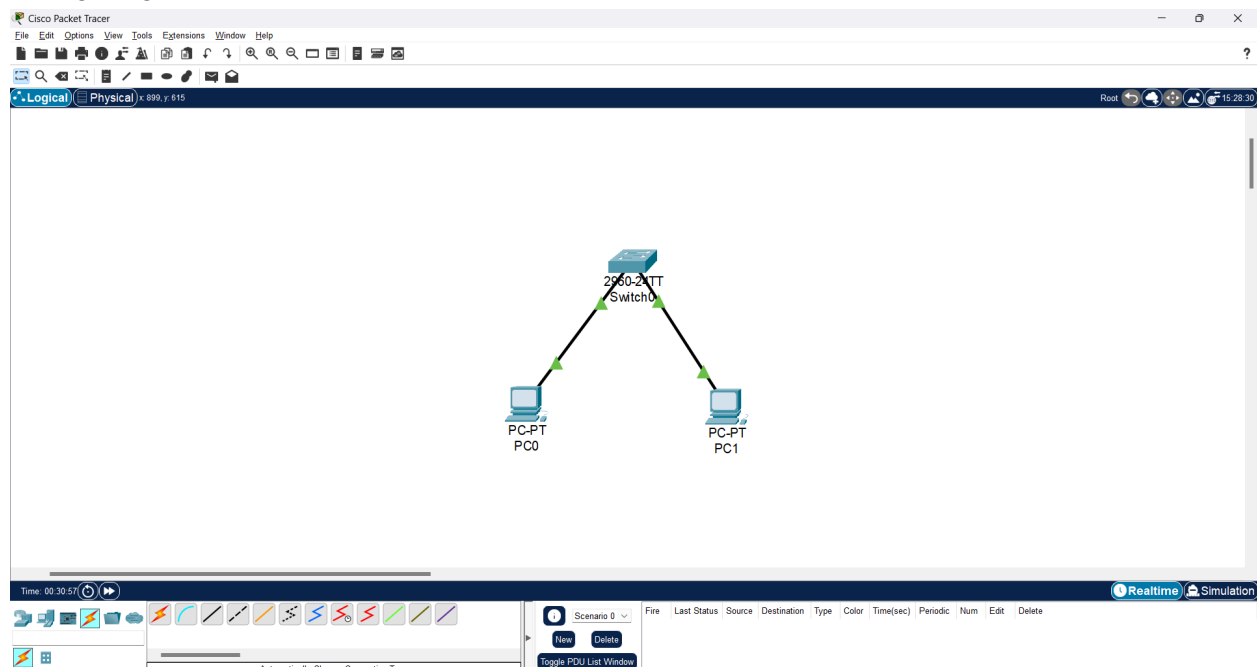
```
▼ Internet Control Message Protocol
  Type: 8 (Echo (ping) request)
  Code: 0
  Checksum: 0xf211 [correct]
  [Checksum Status: Good]
  Identifier (BE): 11769 (0x2df9)
  Identifier (LE): 63789 (0xf92d)
  Sequence Number (BE): 1 (0x0001)
  Sequence Number (LE): 256 (0x0100)
  [Response frame: 31]
  > Data (56 bytes)
```

It consists of the data Payload, checksum, identifier and sequence number for it.

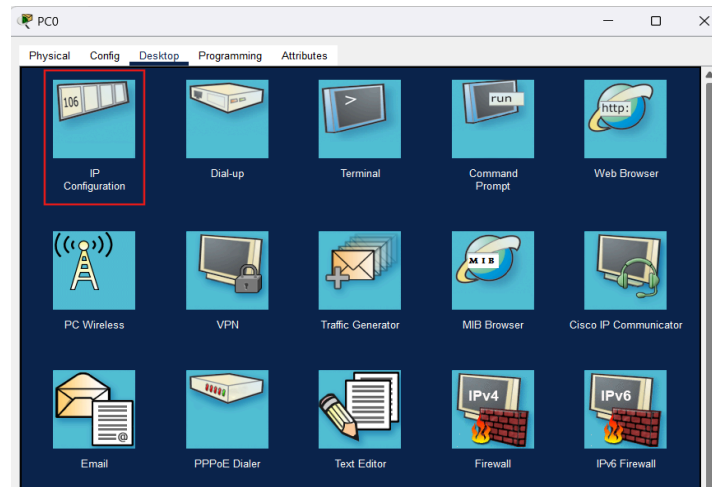
FCS : Frame Check Sequence won't be visible on Wireshark but it is checked by Wireshark. It is used for error detection in ethernet frames. It is basically the checksum which is appended to the end of the frame. It is important for data reliability and data integrity.

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

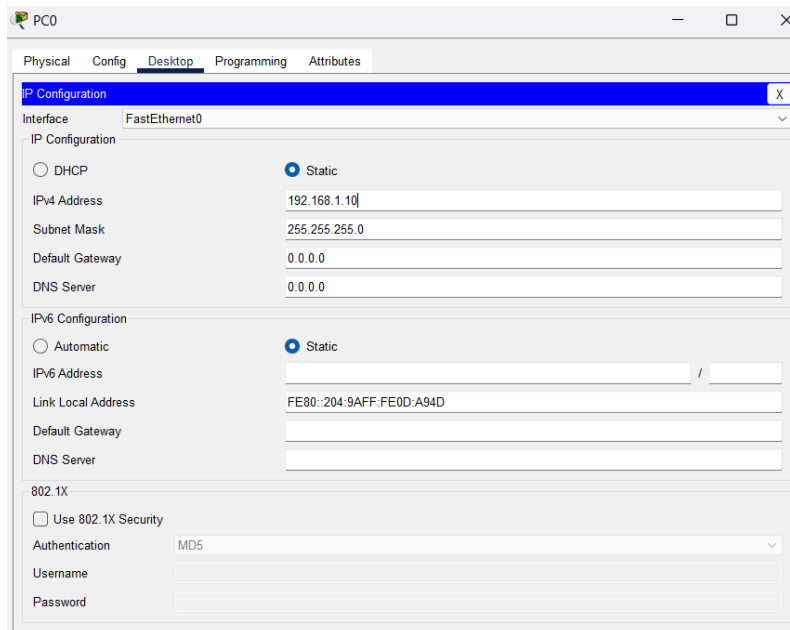
Configuring a network in cisco packet tracer with two PC.



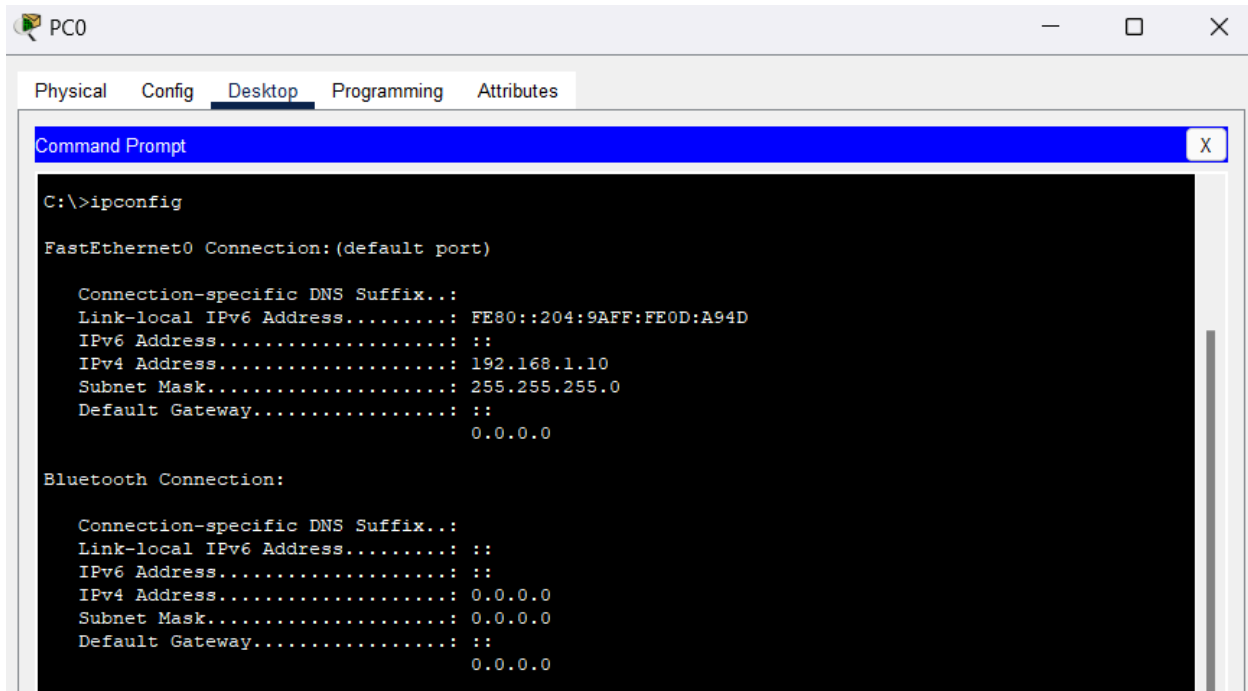
To configure static IP address, go to desktop, and click IP Configuration



Enter Static IP address and network mask



And configure for other PC also with 192.168.1.11. Now check IP using ipconfig



The screenshot shows a window titled 'PC0' with tabs for Physical, Config, Desktop, Programming, and Attributes. The 'Desktop' tab is active, displaying a 'Command Prompt' window. The command prompt shows the output of the 'ipconfig' command. It details the configuration for 'FastEthernet0 Connection:(default port)' and 'Bluetooth Connection'. The FastEthernet0 configuration includes a Link-local IPv6 Address (FE80::204:9AFF:FE0D:A94D), an IPv4 Address (192.168.1.10), a Subnet Mask (255.255.255.0), and a Default Gateway (0.0.0.0). The Bluetooth connection shows all fields as empty or 0.0.0.0.

```
C:\>ipconfig

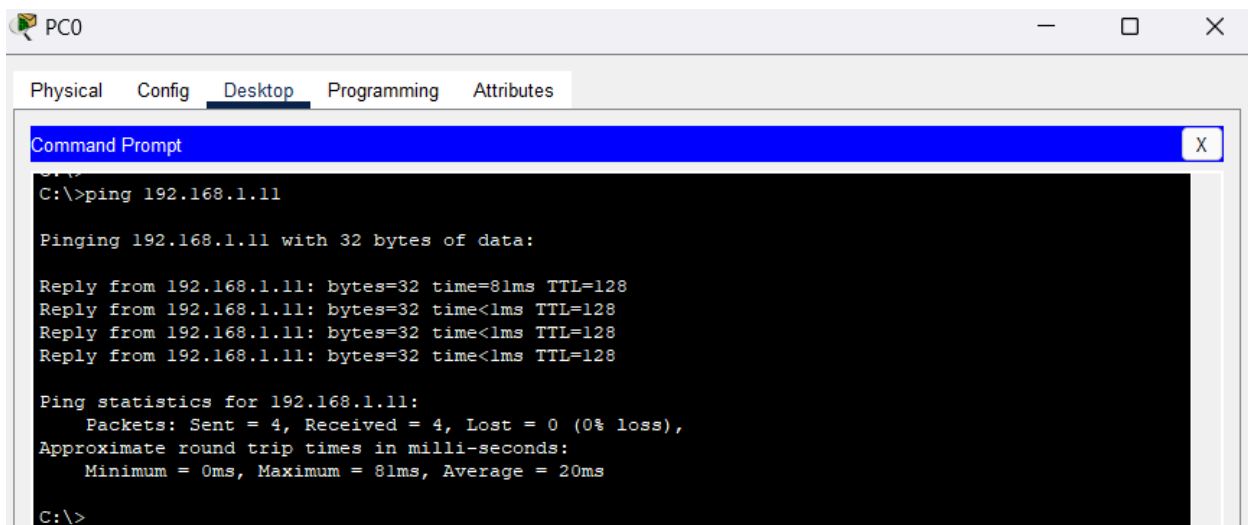
FastEthernet0 Connection:(default port)

    Connection-specific DNS Suffix...:
    Link-local IPv6 Address . . . . .: FE80::204:9AFF:FE0D:A94D
    IPv6 Address . . . . .: ::
    IPv4 Address . . . . .: 192.168.1.10
    Subnet Mask . . . . .: 255.255.255.0
    Default Gateway . . . . .: ::
                                   0.0.0.0

Bluetooth Connection:

    Connection-specific DNS Suffix...:
    Link-local IPv6 Address . . . . .: ::
    IPv6 Address . . . . .: ::
    IPv4 Address . . . . .: 0.0.0.0
    Subnet Mask . . . . .: 0.0.0.0
    Default Gateway . . . . .: ::
                                   0.0.0.0
```

Now lets ping from PC0 to PC1 and verify



The screenshot shows the same 'PC0' window with the 'Command Prompt' displaying the output of a 'ping 192.168.1.11' command. It shows four successful replies from 192.168.1.11 with 32 bytes of data, times less than 1ms, and TTL=128. Ping statistics show 4 packets sent, 4 received, 0% loss, and average round trip times of 20ms.

```
C:\>ping 192.168.1.11

Pinging 192.168.1.11 with 32 bytes of data:

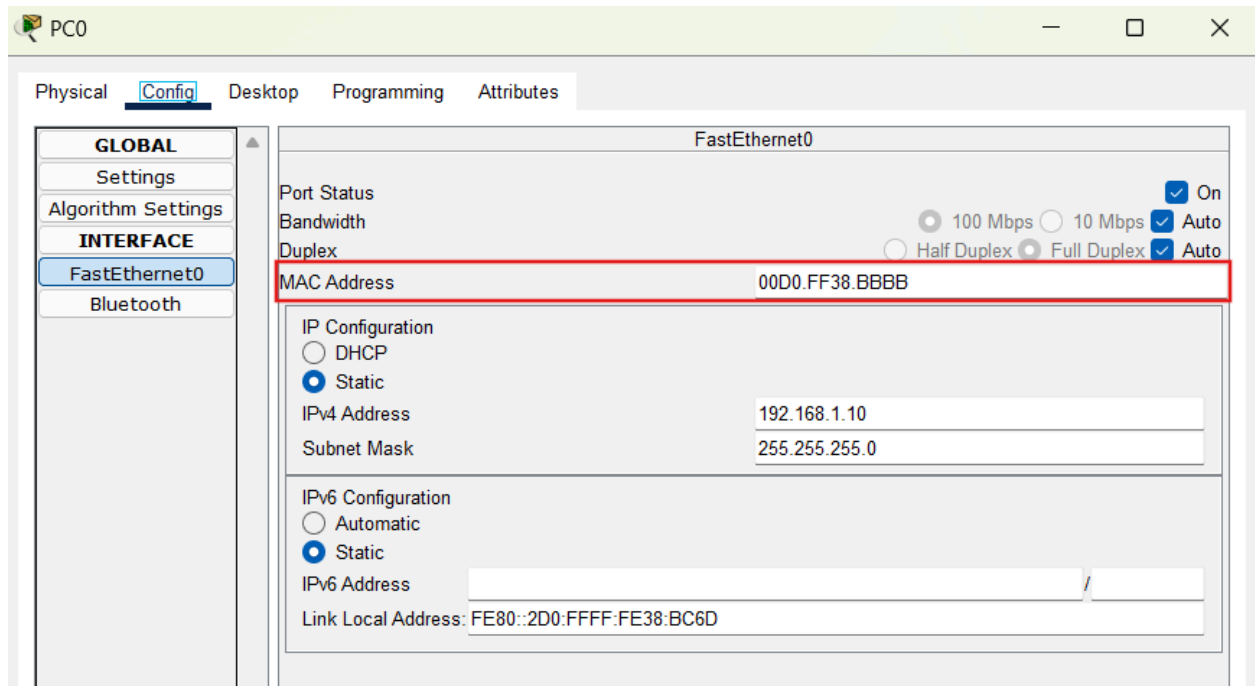
Reply from 192.168.1.11: bytes=32 time=81ms TTL=128
Reply from 192.168.1.11: bytes=32 time<1ms TTL=128
Reply from 192.168.1.11: bytes=32 time<1ms TTL=128
Reply from 192.168.1.11: bytes=32 time<1ms TTL=128

Ping statistics for 192.168.1.11:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 0ms, Maximum = 81ms, Average = 20ms

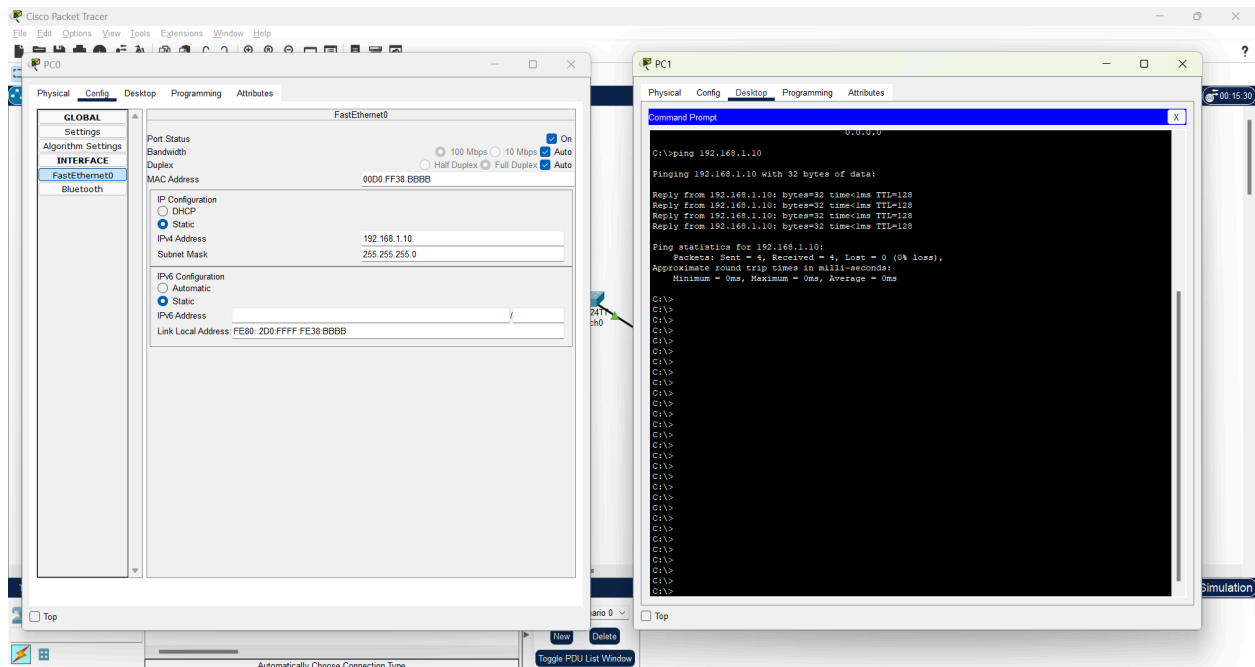
C:\>
```

To Modify the MAC Address :

We can change the MAC address by going to the config section of the PC and change the address. After changing lets check it with ping command also.



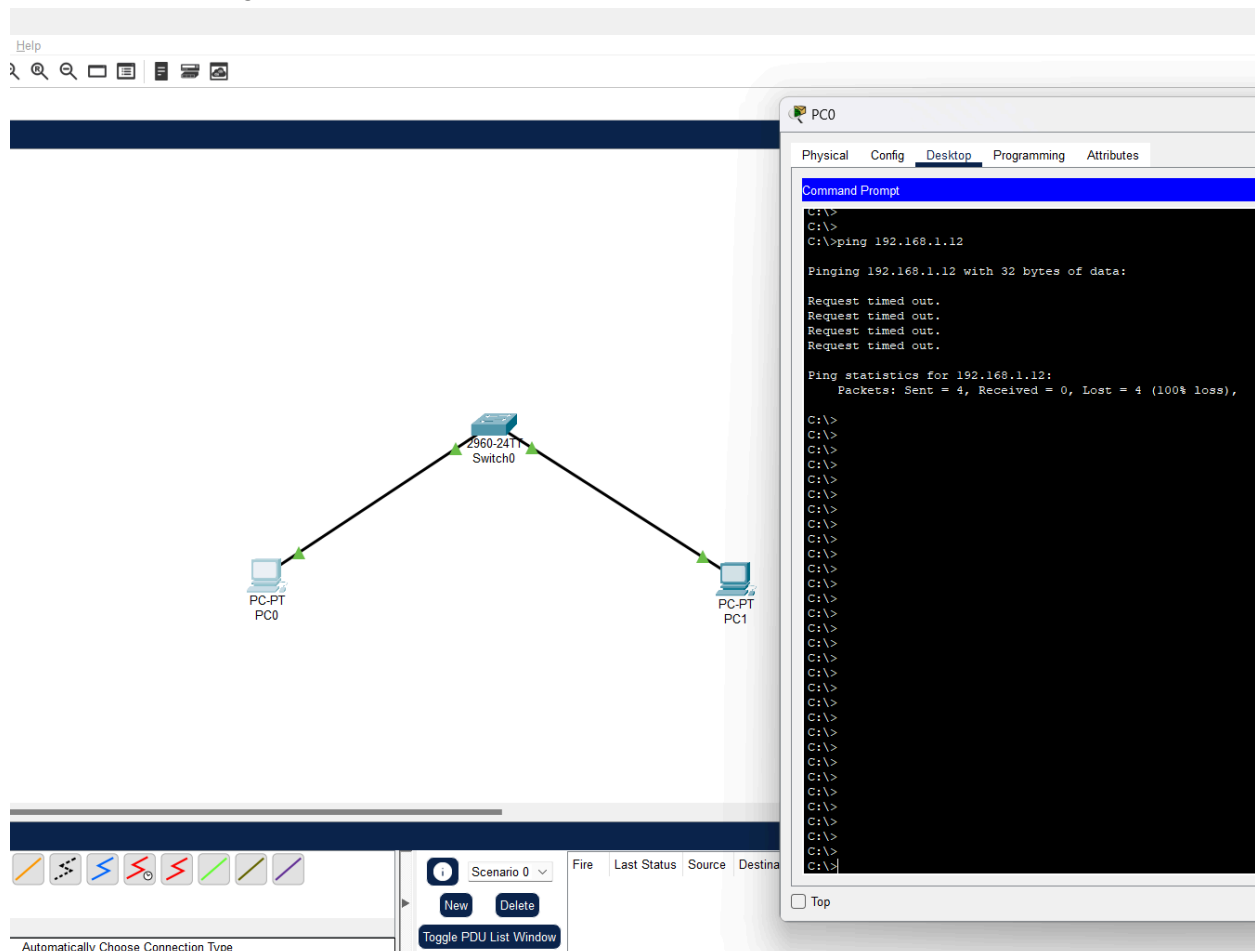
And we see that, it is working after changing the MAC address also :

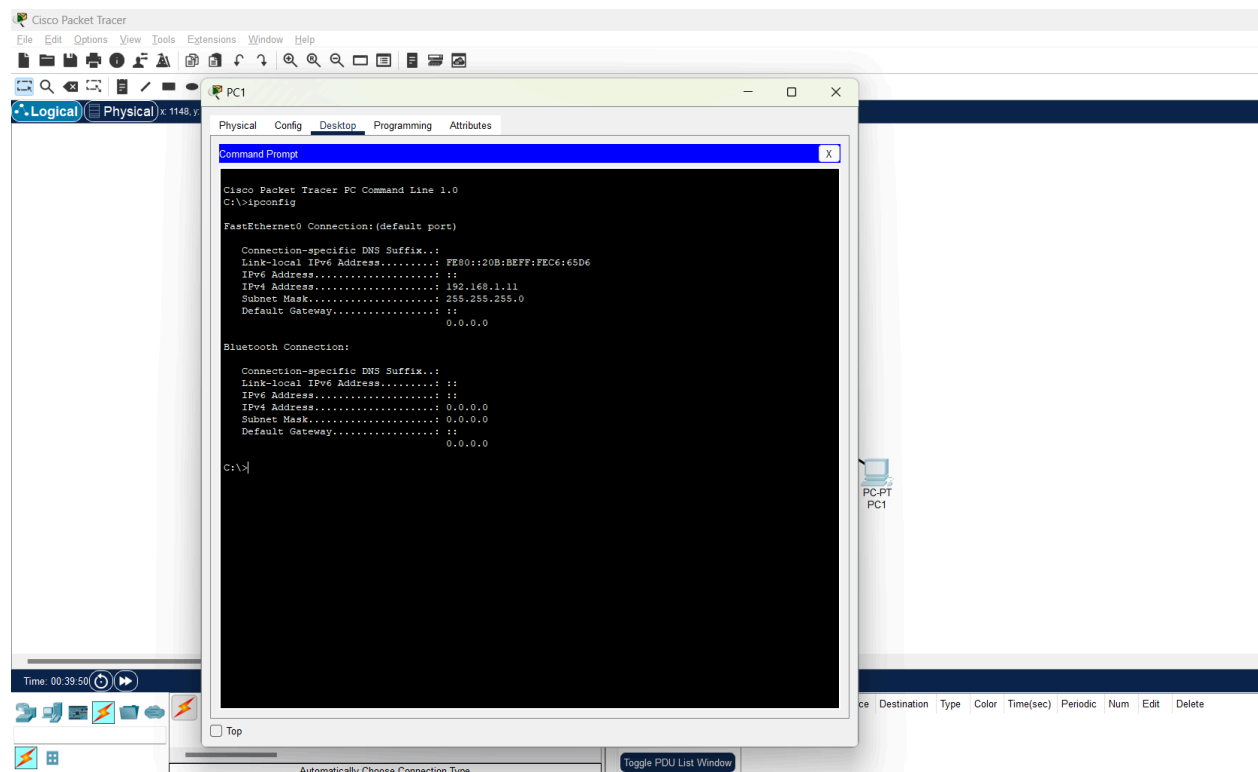
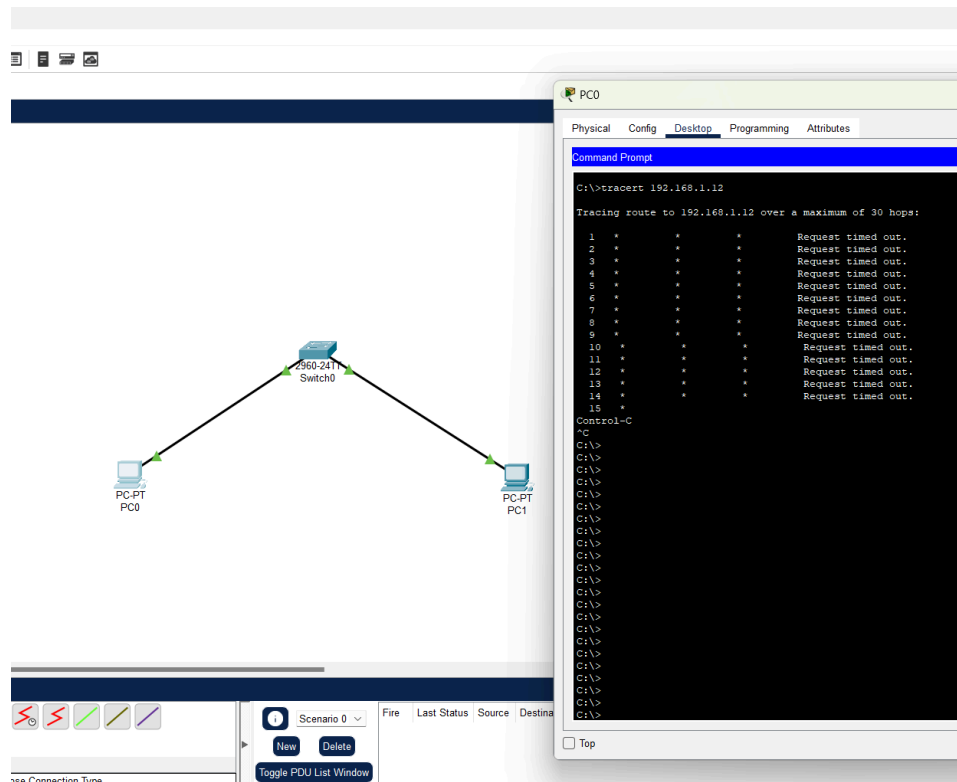


4. Troubleshoot Ethernet Communication with ping and traceroute -> Using cisco packet tracer:

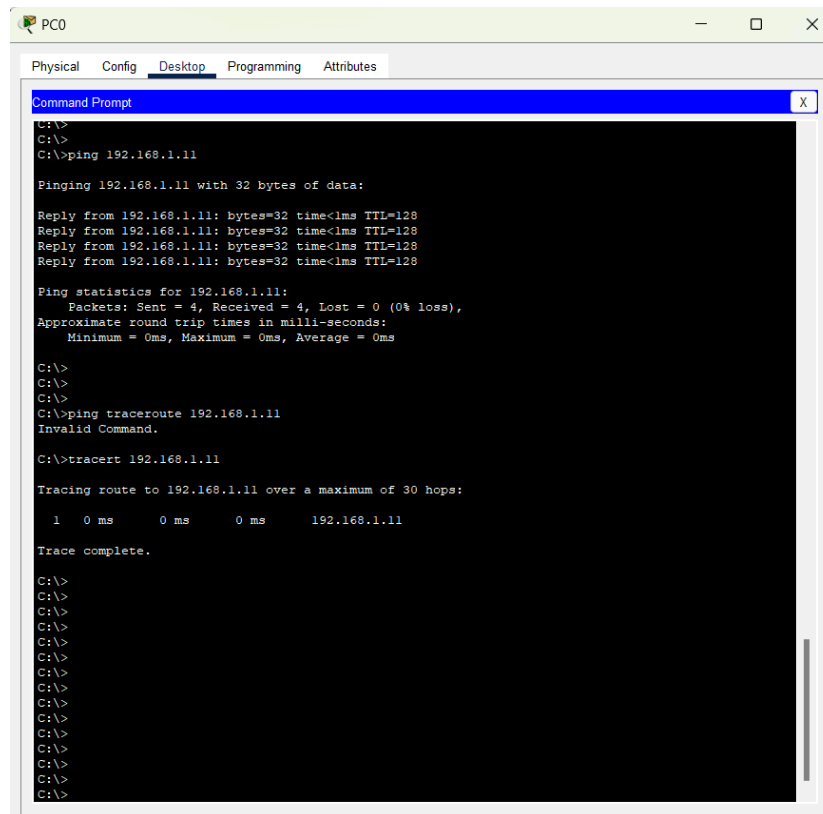
- Create a simple LAN setup with two Linux machines connected via a switch. Ping from one machine to the other.
- If it fails, use ifconfig to ensure the IP addresses are configured correctly.
- Use traceroute to identify where the packets are being dropped if the ping fails.

Create a simple LAN with two Machines connected via switch and trying to ping to PC1 but it is not working.





Now lets again try ping and traceroute with 192.168.1.11 and we see that it is working.



The screenshot shows a PC0 desktop environment with a window titled 'PC0'. Inside the window, there are tabs for 'Physical', 'Config', 'Desktop', 'Programming', and 'Attributes'. The 'Desktop' tab is active, displaying a 'Command Prompt' window. The Command Prompt shows the following commands and output:

```
C:\>
C:\>
C:\>ping 192.168.1.11

Pinging 192.168.1.11 with 32 bytes of data:

Reply from 192.168.1.11: bytes=32 time<1ms TTL=128
Reply from 192.168.1.11: bytes=32 time<1ms TTL=128
Reply from 192.168.1.11: bytes=32 time<1ms TTL=128
Reply from 192.168.1.11: bytes=32 time<1ms TTL=128

Ping statistics for 192.168.1.11:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 0ms, Maximum = 0ms, Average = 0ms

C:\>
C:\>
C:\>
C:\>ping traceroute 192.168.1.11
Invalid Command.

C:\>tracert 192.168.1.11

Tracing route to 192.168.1.11 over a maximum of 30 hops:

  1  0 ms    0 ms    0 ms    192.168.1.11

Trace complete.

C:\>
C:\>
C:\>
C:\>
C:\>
C:\>
C:\>
C:\>
C:\>
C:\>
C:\>
C:\>
```

Hence we have troubleshooted the network with ping and traceroute command.

5. 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).

In the linux kernel, the ethernet devices and the network interface are handled by the netdev which is a crucial component of managing network interfaces and all network related operations.

It consists of functionalities such as device drivers for NIC, management of network interfaces, and implementation of network protocols.

Network interface objects are software abstractions that represent network interfaces in the Linux kernel.

These items serve as an essential bridge connecting a system's various network hardware components and the networking stack of the kernel.

They allow the kernel to efficiently control network traffic by storing crucial data about each interface, including its MAC address, IP address, and operational status.

The kernel can receive incoming data from the network and route outgoing data to the appropriate interface for transmission by interacting with these objects.

Administrators may easily interact with and control these network interface objects by using the `ip` command, which enables them to monitor network traffic, configure IP addresses, and bring interfaces up or down.

Every Ethernet interface in Linux is controlled by a driver and is shown as a network device object in the kernel. Important details like the MAC and IP addresses are contained in this item.

These objects are used by the kernel's network stack to manage Ethernet frame transmission and reception, guaranteeing effective network connection. These interfaces can be managed and configured by users using the ip command.

Reference :

https://web.git.kernel.org/pub/scm/linux/kernel/git/torvalds/linux.git/tree/net/l2tp/l2tp_eth.c?h=v6.14-rc5

<https://docs.kernel.org/process/maintainer-netdev.html#netdev-faq>

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

To configure a LAN interface using the ip command :

Interface names can be eth0, eth1, en0 or anything similar to that.

Followed by which you can add or delete an LAN interface or a network interface using the IP command with the following commands :

```
sudo ip addr add 192.168.1.100/24 dev eth0
```

To delete an interface :

```
sudo ip route delete 10.0.0.0/24 via 192.168.1.1 dev eth0
```

To modify the default gateway :

```
sudo ip route add default via 192.168.1.254 dev eth0
```

7. 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.

To view the MAC address of the machine use :

```
tkgowtham@tkgowtham-VirtualBox:~$ ip link show enp0s3
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER UP> mtu 1500 qdisc fq_codel state UP mode DEFAULT group default qlen 1000
    link/ether 08:00:27:d8:25:4b brd ff:ff:ff:ff:ff:ff
tkgowtham@tkgowtham-VirtualBox:~$
```

To view the MAC table we will use the neighbour option with ip command :

```
tkgowtham@tkgowtham-VirtualBox:~$ ip neigh show
192.168.1.1 dev enp0s3 lladdr f8:c4:f3:c3:f3:d0 DELAY
fe80::1 dev enp0s3 lladdr f8:c4:f3:c3:f3:d0 router DELAY
tkgowtham@tkgowtham-VirtualBox:~$
```

Using bridge command to view the MAC table :

```
tkgowtham@tkgowtham-VirtualBox:~$ bridge fdb show dev enp0s3
01:00:5e:00:00:01 self permanent
33:33:00:00:00:01 self permanent
33:33:ff:d6:33:7f self permanent
01:00:5e:00:00:fb self permanent
33:33:ff:cd:80:1f self permanent
33:33:00:00:00:fb self permanent
33:33:ff:f7:83:e1 self permanent
tkgowtham@tkgowtham-VirtualBox:~$
```

Some switch operations that can be performed using ip command are :

Add a bridge : `sudo ip link add name <name> type bridge`

Add port to bridge : `sudo ip link set dev enp0s3 master <name>`

Delete a bridge : `sudo ip link delete <name>`

View a bridge : `sudo ip link show <name>`