THADOMAL SHAHANI TSEC ENGINEERING COLLEGE

Experiment-1

CN-C32-2103164

Title:- Study of RJ45 & CAT6 calling & connection using crimping tool.

Theory:-

RJ45 & CAT6 calling are widely used in networking to connect various devices such as computer router, switches and accesspoint to create LAN and establish internet connectivity. The process of creating these connection involves the use of crimping tool to terminate the cable with RJ45 connector.

What is Straight-Through Cable?

Straight-through Cable is a type of CAT5 with RJ-45 connection at each end, and each has the same pin out. It is in accordance with either the same color code throughout the LAN. For consistency. This type of twisted-pair cable is used in LAN to connect a computer or a network hub such as a router. It is one of most common types of network cable

What is crossover cable?

The crossover cable is a type of CAT5 where one end is T568A configuration and other end as T568B configuration. In this type of

cable connection, pin 1 is crossed with Pin 3, and Pin 2 is crossed with Pin 6. The internal wiring of cross over cable reverses the transmission and receive signals. It is widely to connect two devices of the same type. eg two computer or two switches to each others.

RJ45 Connector (Registered Jack 45) :-

RJ45 is most common type of ethernet connector used for wired network connection. It is an 8P8C modular plug that can accomdate eight wire allowing for data transmission and communication between network devices. The connector name indicates that it has eight position & eight contacts.

CAT6 Cable (category 6 cable)

CAT6 is type of twisted ethernet cable which means it consist of pair of copper wires twisted together to reduce interference & crosstalk. It is an improvement over its predecessor CAT5e, offering higher data transfer speeds and improved performance.

Procedure of Experiment (Crimping Process)

1.) Strip the cable back 1 inch (25 mm) from the end.

2.) Untwist and straighten the wires inside of the cable

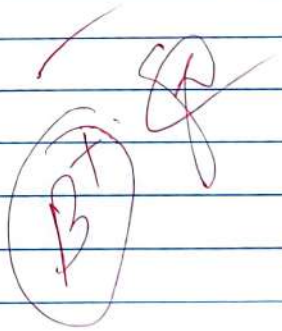
3) Arrange the wires into right order. The proper sequence is as follows from left to right.
Orange/white, Orange, Green/white, Blue, Blue/white, Green, Brown/White, Brown

4.) Cut the wires into even line 1/2 inch [13mm] from sheathing.

5.] Insert the wires into the RJ45 connector. If any small wires bend or don't fit into a groove correctly, take the cable out and straighten the wires with your finger before trying again.

6.] Stick the connector into the crimping part of the tool and squeeze twice.

7.] Remove the cable from tool and check that all of pins are down.



EXP - 2

CN - C32 - 2103164

AIM - Implementation of Hamming Code for error correction and detection

Theory :- The Hamming Code is a widely used for error correction and detection in digital communication. The theory behind the Hamming Code involves adding parity bits to data stream to detect and correct error. It works as follows :-

Data Encoding :- For input message of 'K' bits, 'r' parity bits are added at positions that are powers of 2 (1, 2, 4, 8... etc). The positions of these parity bits are calculated by using binary representation. Each parity bit is responsible for checking subset of data bits.

Parity Calculation :- Each parity bit calculates its value based on data bits that it covers. For eg, parity bit at position 1 checks all data bits that have a 1 in their least significant bit and so on. Parity bits are calculated using even parity, which means total numbers of 1's in group of bits should be even.

Error Detection :- During transmission, if a bit get flipped due to noise, the parity bits will no longer match their recalculated values. This discrepancy indicates an error

Error Correction :- If an error is detected, the erroneous bits position can be determined by looking at parity bits

that cover that position. By using the pattern of incorrect parity bits, the erroneous bits position ca identified and corrected.

Decoding:- The received data is compared to parity If any discrepancies are found, error detection correction are performed based on parity bits positi

Its limitations, is it cannot correct more tha two errors or detect error in parity bits the

Problem :- A 7 bit Hamming code is received as Assume Even parity & state whether the recei is correct or wrong, if wrong locate the bit in

Solution:- Received HC :-

D_7	D_6	D_5	P_4	D_3	P_2	P_1
1	0	1	1	0	1	1

Detecting errors :-

Step 1: Analyzing bits 1, 3, 5, 7

We have $P_1 D_3 D_5 D_7 \Rightarrow 1011$ { odd pa error

We put $P_1 = 1$

SN-C32-2103164

Step 2: analyzing bit 2, 3, 6 & 7

P_2 D_3 D_6 D_7

$\Rightarrow$ 1 0 0 1 { Even parity No Errors }

$\Rightarrow P_2 = 0$.

Step 3:- Analyzing bits 4, 5, 6 & 7

We have P_4 D_5 D_6 D_7

$\Rightarrow$ 1 1 0 1 { odd parity error Exists }

We put $P_4 = 1$

Correcting error

	P_4	P_2	P_1
error word E =	1	0	1

decimal value $\mathcal{E} = 5$ which shows that 5th bit is in error

We write the correct world by simply inverting the 5th bit

$\therefore$ Correct word = 1 0 0 1 0 1 1

27/08/23

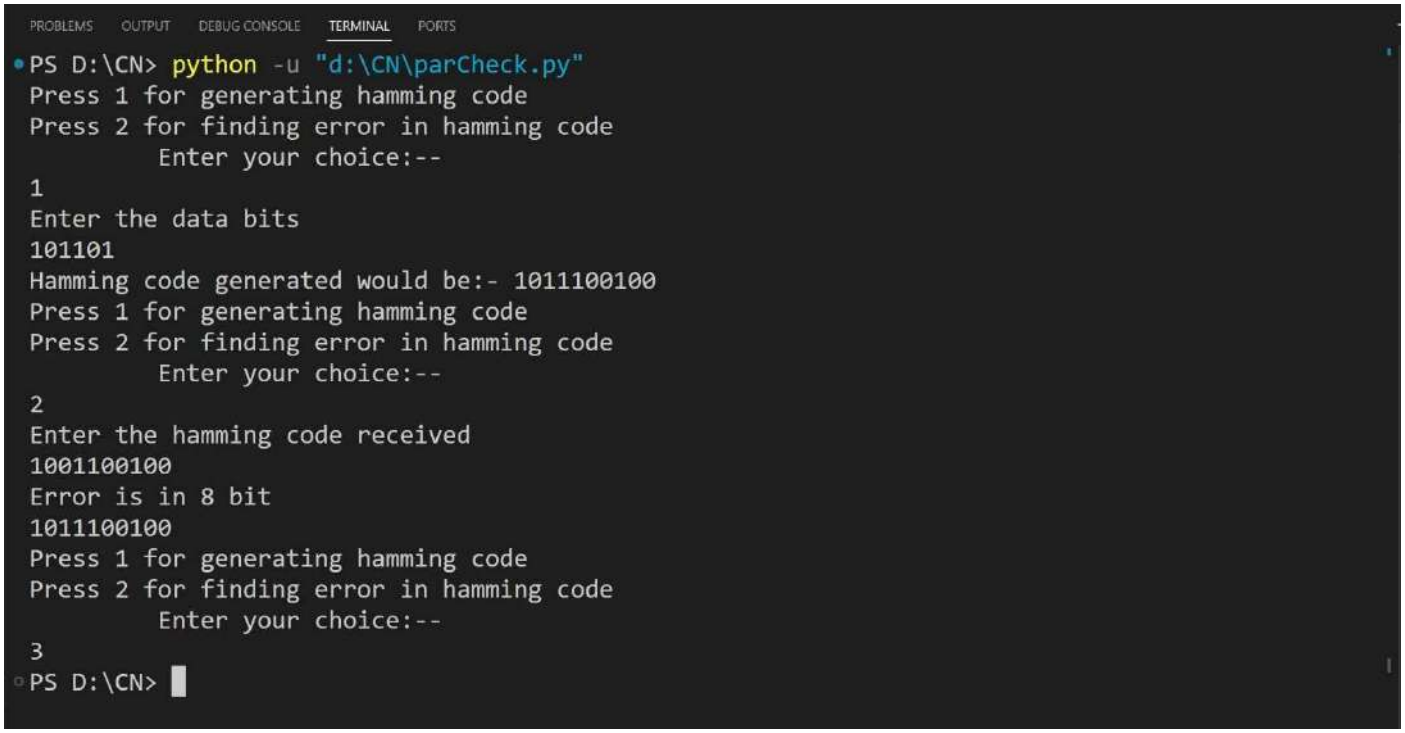
```
option=int(input('Press 1 for generating hamming code  \nPress 2 for finding error in hamming code\n\t Enter your choice:--\n'))
while(option>0 and option<3):  # GENERATE HAMMING CODE

   if(option == 1):
      print('Enter the data bits')
      d=input()
      data=list(d)
      data.reverse()
      c,ch,j,r,h=0,0,0,0,[]

      while ((len(d)+r+1)>(pow(2,r))):
         r=r+1

      for i in range(0,(r+len(data))):
         p=(2**c)

         if(p==(i+1)):
            h.append(0)
            c=c+1

         else:
            h.append(int(data[j]))
            j=j+1

      for parity in range(0,(len(h))):
         ph=(2**ch)
         if(ph==(parity+1)):
            startIndex=ph-1
            i=startIndex
            toXor=[]

            while(i<len(h)):
```

```
            block=h[i:i+ph]
            toXor.extend(block)
            i+=2*ph

        for z in range(1,len(toXor)):
            h[startIndex]=h[startIndex]^toXor[z]
        ch+=1

    h.reverse()
    print('Hamming code generated would be:- ', end="")
    print(int(''.join(map(str, h))))


elif(option==2): # DETECT ERROR IN RECEIVED HAMMING CODE
    print('Enter the hamming code received')
    d=input()
    data=list(d)
    data.reverse()
    c,ch,j,r,error,h,parity_list,h_copy=0,0,0,0,0,[],[],[]

    for k in range(0,len(data)):
        p=(2**c)
        h.append(int(data[k]))
        h_copy.append(data[k])
        if(p==(k+1)):
            c=c+1

    for parity in range(0,(len(h))):
        ph=(2**ch)
        if(ph==(parity+1)):

            startIndex=ph-1
```

```
        i=startIndex
        toXor=[]

        while(i<len(h)):
            block=h[i:i+ph]
            toXor.extend(block)
            i+=2*ph

        for z in range(1,len(toXor)):
            h[startIndex]=h[startIndex]^toXor[z]
        parity_list.append(h[parity])
        ch+=1
parity_list.reverse()
error=sum(int(parity_list) * (2 ** i) for i, parity_list in enumerate(parity_list[::-1]))

if((error)==0):
    print('There is no error in the hamming code received')

elif((error)>=len(h_copy)):
    print('Error cannot be detected')

else:
    print('Error is in',error,'bit')

    if(h_copy[error-1]=='0'):
        h_copy[error-1]='1'

    elif(h_copy[error-1]=='1'):
        h_copy[error-1]='0'
        print('After correction hamming code is:- ')
    h_copy.reverse()
    print(int(''.join(map(str, h_copy))))
```

```
    else:
        print('Option entered does not exist')


    option=int(input('Press 1 for generating hamming code  \nPress 2 for finding error in hamming code\n\t Enter your choice:--\n'))
```

OUTPUT



THADOMAL SHAHANI TSEC ENGINEERING COLLEGE

EXP-3

CN - C32 - 2103164

AIM - Implementation of CRC (Cyclic Redundancy Check)

- Theory :- CRC is method of detecting error in communication channel. Given K bit frame or message, the transmitter generates a n bit sequence known, as frame check sequence (FCS), so that resulting frame, consisting of (K+n) bits.

- Bits sequences can be written as polynomials with coefficient 0 and 1.

- frame with K bits is considered as polynomial of degree K-1.

- The most significant bit is coefficient of x^{K-1} The next bit is coefficient of x^{K-2}. Example : The bit sequence 10011010 corresponds to this polynomial :

- Sending and recieving message can be imagined as exchange of polynomials

$$M(x) = 1 * x^7 + 0 * x^6 + 0 * x^5 + 1 * x^4 + 1 * x^3 + 0 * x^2 + 1 * x^1 + 0 * x^0$$
$$= x^7 + x^4 + x^3 + x^1$$

- The Data link Layer Protocal specifies a generator polynomial C(x). Generator polynomial is available for both sender and reciever side.

- C(x) is a polynomial of degree K
- If eg
  $C(x) = x^3 + x^2 + x^0$
  $= 1101$, Thus $K = 3$
- Therefore, generator polynomial is degree 3.
- The degree of generator polynomial is equal to of bits minus one.
- If for a frame, the CRC need to be calculated, are appended to the frame.
- n corresponds to degree of generator polynomial

Generator polynomial	100110

- The generator polynomial has 6 digits
  - Therefore, five 0 bits are appended.

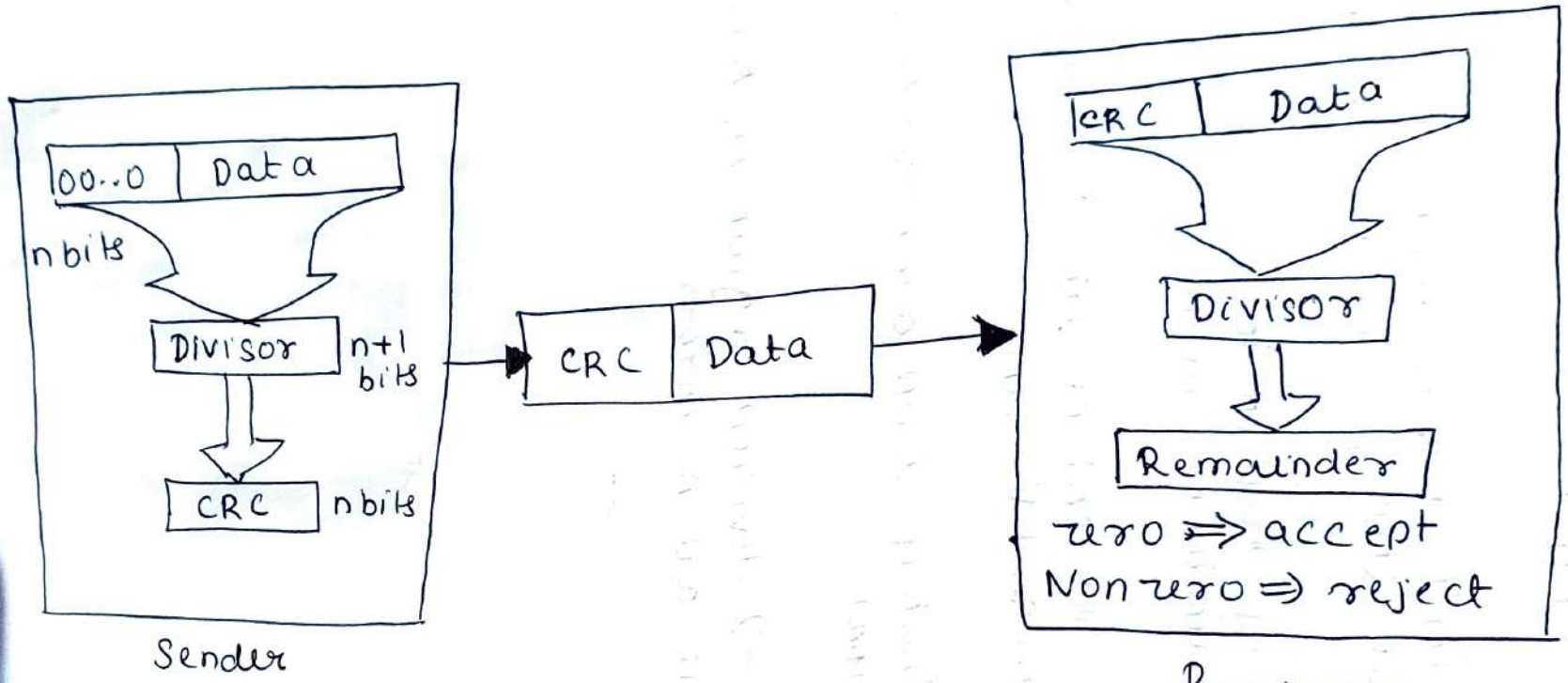
Frame (payload)	10101
Frame with appended 0 bit	101010000

Sender Side (Generation of encoded Data from Data and Generator Polynomial (or Key)):

- binary data is first argumented by adding n zeros in end of the data (n degree of generator polynomial)
- Use modulo-2 binary division to divide binary data by Key and store remainder of division.
- Append remainder at end of data to form the encoded data and send the same.

Reciever Side (Check if there are error introduced in transmission)

- Perform modulo-2 division again if the remainder is 0, then there are no errors

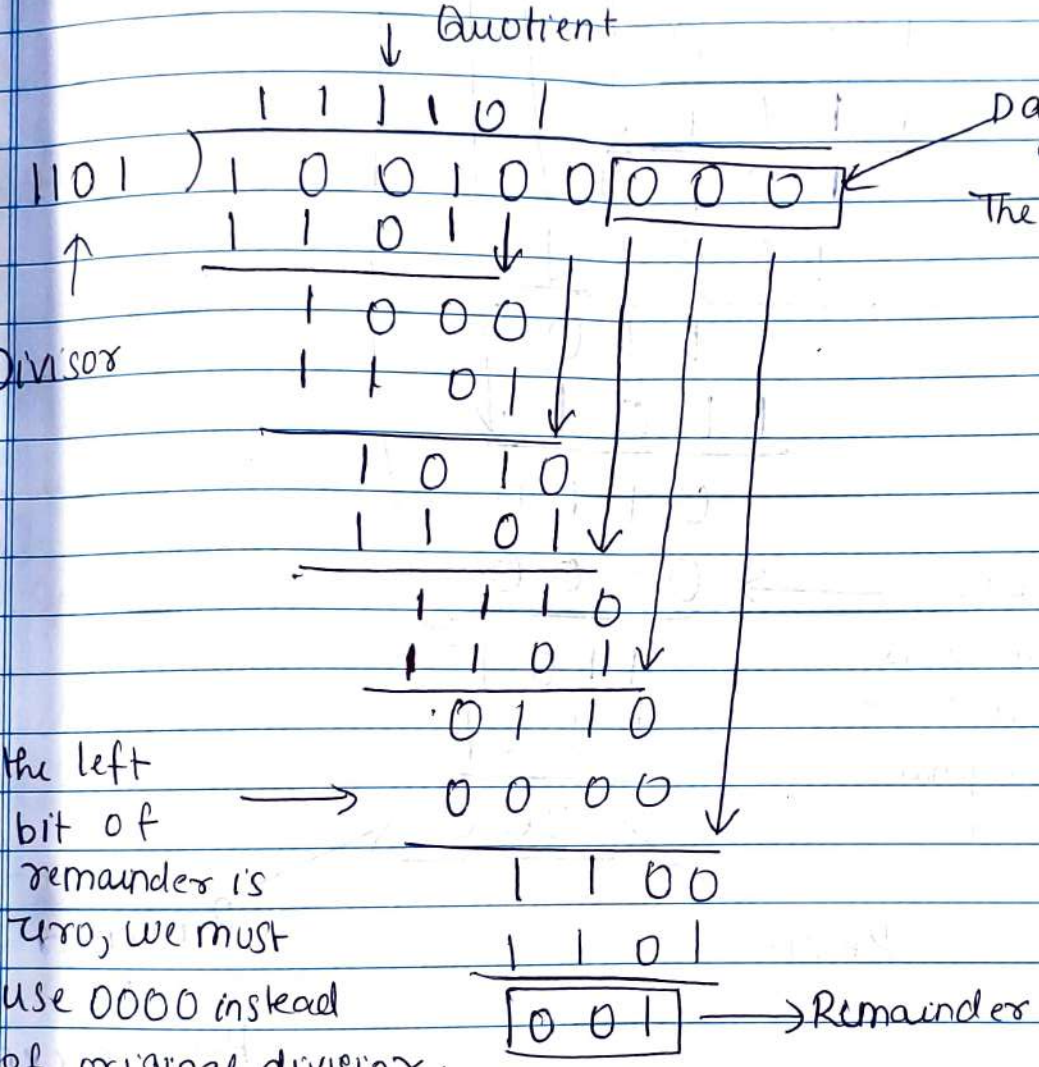


CRC (Cyclic Reducary

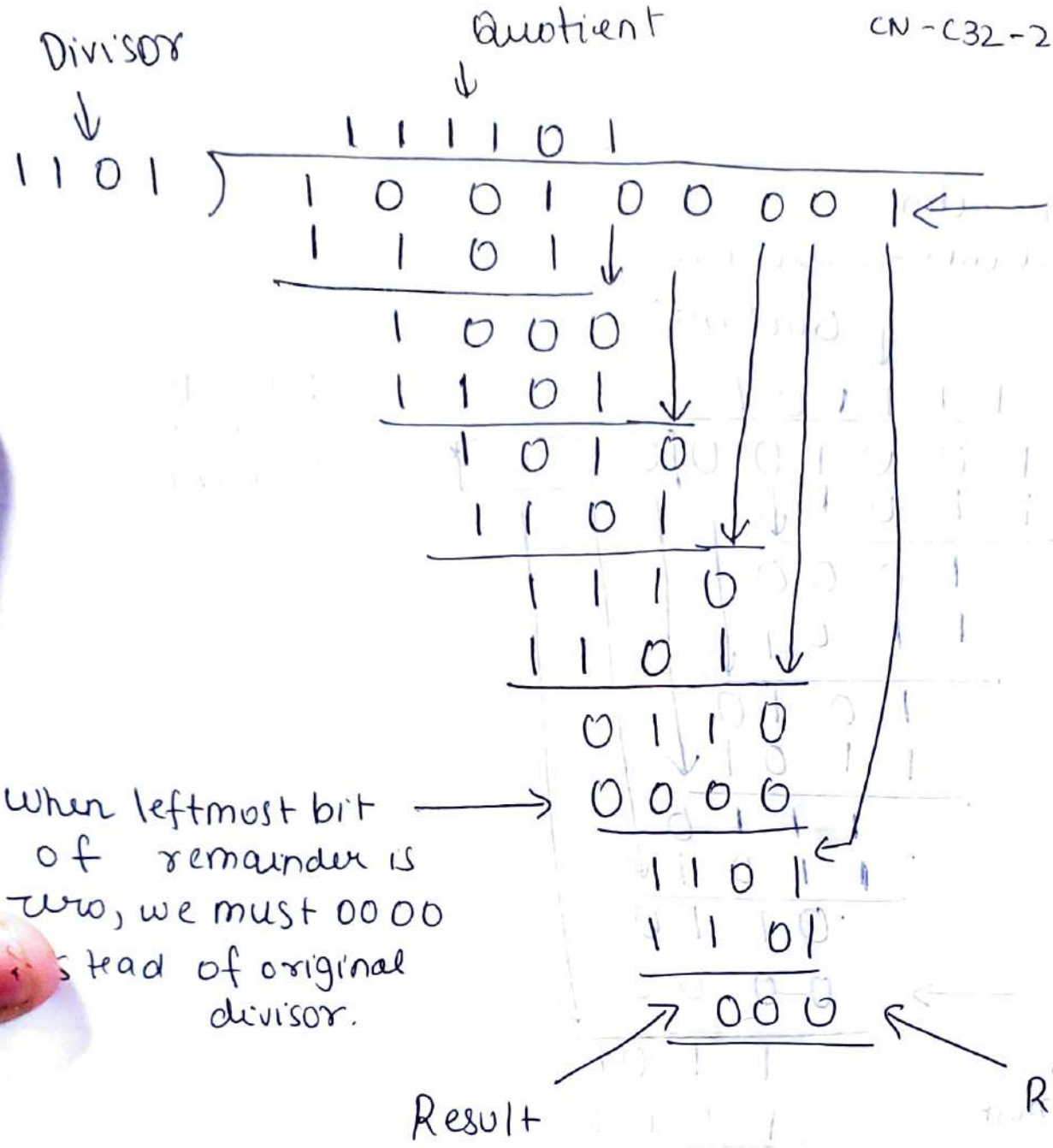
CN-C32-2103114

Eg:-

⇒ CRC generator uses modular-2 division



CN-C32-2103164



When leftmost bit of remainder is zero, we must 0000 instead of original divisor.

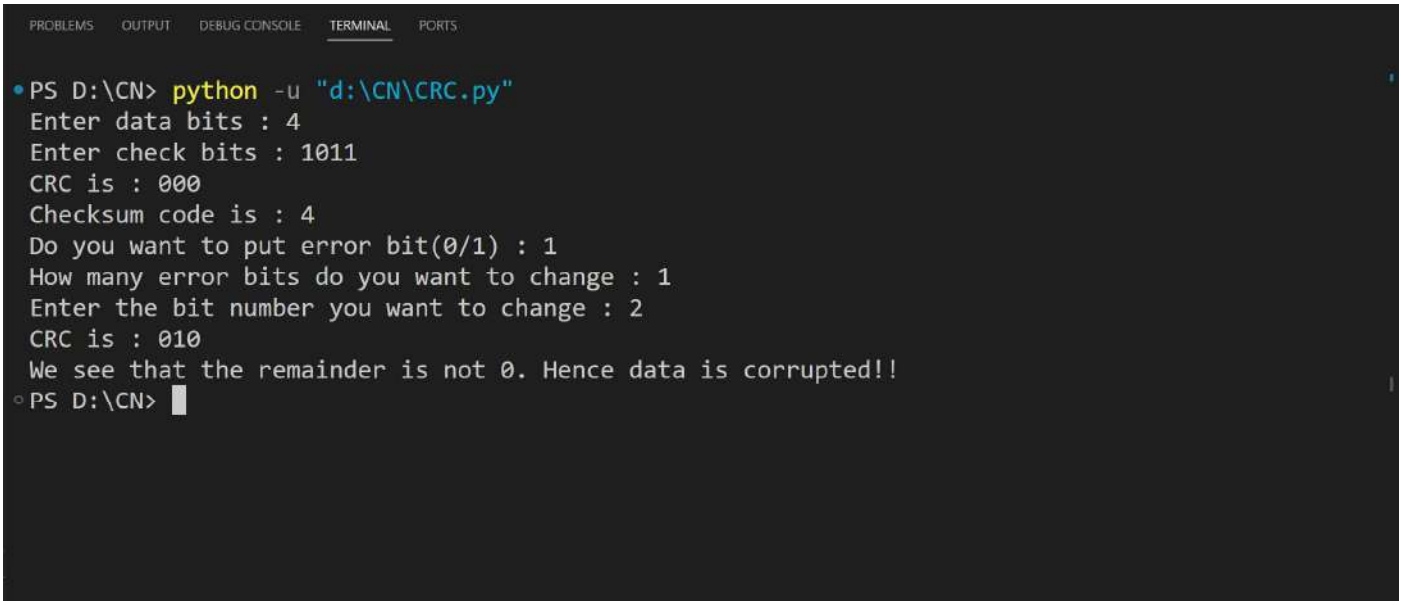
Hence acc

```
def XOR(x, y):
    if x == y:
        return 0
    return 1

def flip(x):
    if x == 0:
        return 1
    return 0

def moduloDivision(data, dividend, divisor):
    for i in range(len(data)):
        if dividend[i] == 1:
            for j in range(len(divisor)):
                dividend[i + j] = XOR(dividend[i + j], divisor[j])
    return dividend

def displayCRC(data, dividend):
    print("CRC is :", end=" ")
    for i in range(len(data), len(dividend)):
        print(dividend[i], end="")
    print()

def displayCheckSum(data, dividend):
    print("Checksum code is :", end=" ")
    for i in range(len(data)):
        dividend[i] = int(data[i])
        print(dividend[i], end="")
    print()

def main():
    print("Enter data bits : ", end="")
```

```
data = input()
print("Enter check bits : ", end="")
check = input()
dividend = [0] * (len(data) + len(check) - 1)
divisor = [0] * len(check)

for i in range(len(data)):
    dividend[i] = int(data[i])

for i in range(len(check)):
    divisor[i] = int(check[i])

# Calculating remainder (CRC)
dividend = moduloDivision(data, dividend, divisor)

# Display remainder
displayCRC(data, dividend)

# Display checksum
displayCheckSum(data, dividend)

# Asking for a change in checksum
print("Do you want to put error bit(0/1) : ", end="")
choice = int(input())

if choice == 1:
    print("How many error bits do you want to change : ", end="")
    select = int(input())
    print("Enter the bit number you want to change : ", end="")
    change = input()
    for i in range(select):
        dividend[int(change[i])] = flip(dividend[int(change[i])])
```

```
        dividend = moduloDivision(data, dividend, divisor)
        displayCRC(data, dividend)
        print("We see that the remainder is not 0. Hence data is corrupted!!")
    else:
        print("CRC obtained at the receiver side is zero")
        print("Data sent without corruption")

if __name__ == "__main__":
    main()
```

OUTPUT



THADOMAL SHAHANI
TSEC
ENGINEERING COLLEGE

EXP 4

CN-C32-2103164

Aim :- Implementation of Go back n sliding window.

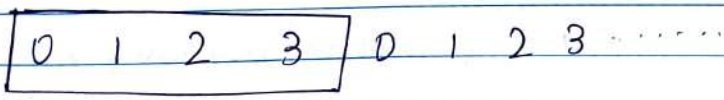
Theory :-

- Discard all subsequent frames following the damaged frames sending no ACKs.
- Eventually the sender times out and retransmits all the unacknowledged frames in order starting with the damaged or lost one.

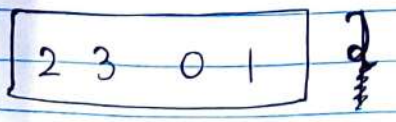
Operation :-

Assume no of bits in sequence no 2.
Window size :- 4 frames.

- Sending window



- Sender has sent 4 frames 0 to 3 and waits for an ACK.
- ACK 0 and ACK 1 received by the sender.



- Now sent 0 1

- Frame 2 lost not received but received 3
- Reciver discards 3 0 and 1

Note

A receiving station does not acknowledge each received frame explicitly. If a sending station recived an ack for frame j' and recieves an ack for frame K it assumes frams between j and K have been recieved correctly

→ Advantages :- Reduces no of acks and lessens n/w traffic.

→ What should be max sender's and recivers window size?

→ Assume K → no of bits in sequence no.

→ Frames are numbered from 0 to $2^K - 1$.

→ Window size cannot be larger than 2^K.

CN-C32-2103164

Case 1: Window size larger than 2^K

- Let K = 3
- Assume window size = 9
- sender window → [0 1 2 3 4 5 6 7 0] 1 2 3 ...
- Problem →
  Sender receives an ACK 0. Does not know if it was for first / last frame.

Window size must be less than or equal to 2^K

Case 2: Window size = 2^K

) At time t1 A sends frames 0–7 to B
) B receives each one recently.
) At time t2, B sends ACK for the most recived frame ACK 7.
) ACK get lost
) Next frame expected by B is frame numbered 0.
) It does not receive ACK and hence retransmits frames 0 through 7 at time t3.
) At t4 B recieves frame 0. Sequence no matches with the one it is expecting. Hence B accepts it.

Protocal fails (since B has accepted a dulicipate and not a new frame)

Reason

- 2 consective windows contain the same sequence no.
- Solution reduce the sender's window size

Case 3 :- Window size less than 2^K.

Sender's window → [0 1 2 3 4 5 6] 0 1 2

1.) A sends frame 0. through 6 at t 1.
2.) B recives each one correctly.
3.) At time t2, B sends ACK for the most recent recieved frame ACK 6.
4.) ACK gets lost
5.) Nesct frame expected by B is frame nu

6.) A does not recive ACK and hence retra frame 0 through 6 at time t3

7.) B recives frame 0 through 6 at t4.
8.) B excepts frame 7.
9.) Hence ignores them.

10.) Eventually B and another ACK 6 whi A recives and A advances it window t include. 7. 0 1 2 3 4 5 and protoco continuous.

Operation performed

- Reciever sends a positive ACK if frame has arrived without error and in order (with excepted seq number)
- Receiver does not have to acknowledge each individual frame recieved correctly and in order.
- Reciever can send cumulative ACK for several frames (ACK 4 acknowledges frames (0, 1, 2, 3, 4))
- If frame is damaged or out of order, the receiver discards it and also discards all subsequent frames until it recieves the one excpted.
- In this case, no ACK will be transmitted.
- If the sender timer expires before recieving ACK, it will resend ALL frames beginning with one expired until last one set.

Drawback Gobackn :-

- Reciver discard all correct frames transmitted after bad one.
- Channel bandwidth wasted on retransmitted frames

Alternative Stragery :-

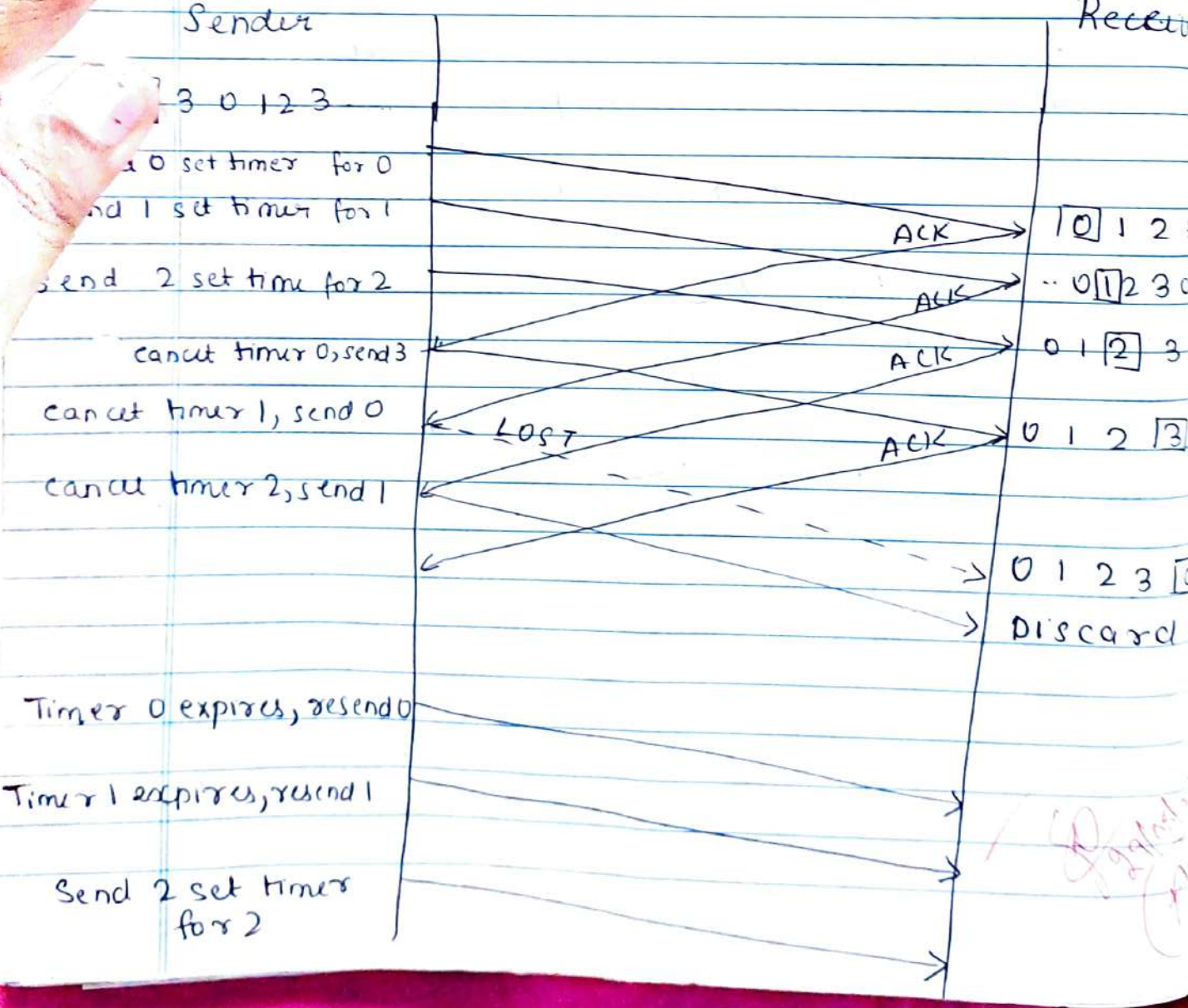
Allow the reciver to accept and buffer the frams following a damaged / lost one - Called Selective Repeat.

Summary

→ Sender window's size - strictly less than 2
(1 less than MAX SEQ)

→ What should be the receivers window s
ames are always received in order.

Hence Receiver's window size should not b
greater than 1.

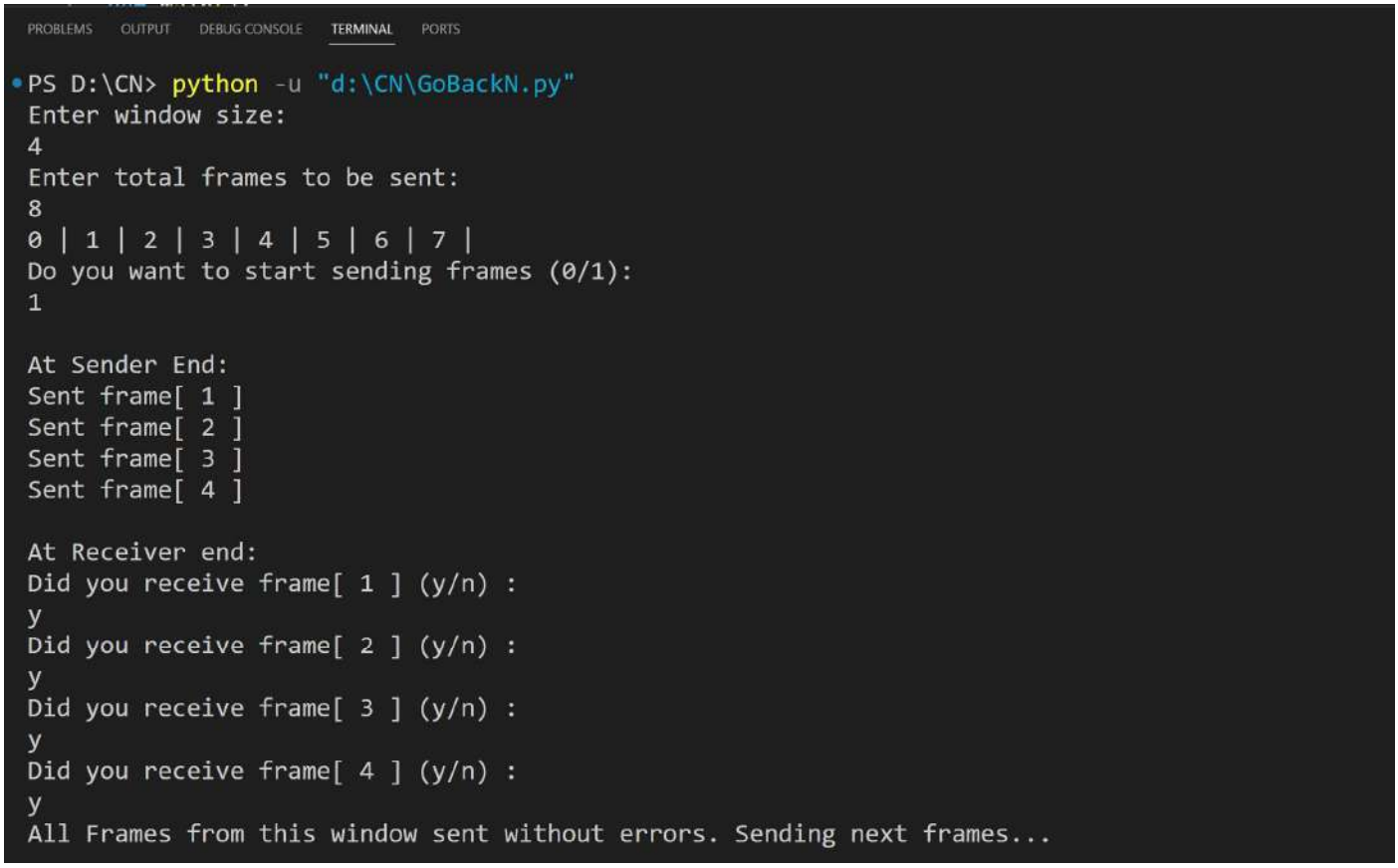


```
def main():
    print("Enter window size: ")
    window_size = int(input())
    print("Enter total frames to be sent: ")
    total_frames = int(input())

    # Initializing array with data frames
    sender_frames = [i for i in range(total_frames)]

    # Displaying data frames
    for frame in sender_frames:
        print(frame, "| ", end="")
    print()

    # Displaying sender window
    print("Do you want to start sending frames (0/1): ")
    choice = int(input())
    print()

    if choice == 1:
        ptr_on_window_left_sender = 0
        ptr_on_window_left_receiver = 0
        total_sent_frames = 0

        while ptr_on_window_left_sender < total_frames:
            # Sender side
            count = 0
            print("At Sender End:")
            for i in range(ptr_on_window_left_sender, total_frames):
                if count < window_size:
                    print("Sent frame[", i + 1, "]")
                    ptr_on_window_left_sender += 1
                    total_sent_frames += 1
                    count += 1
```

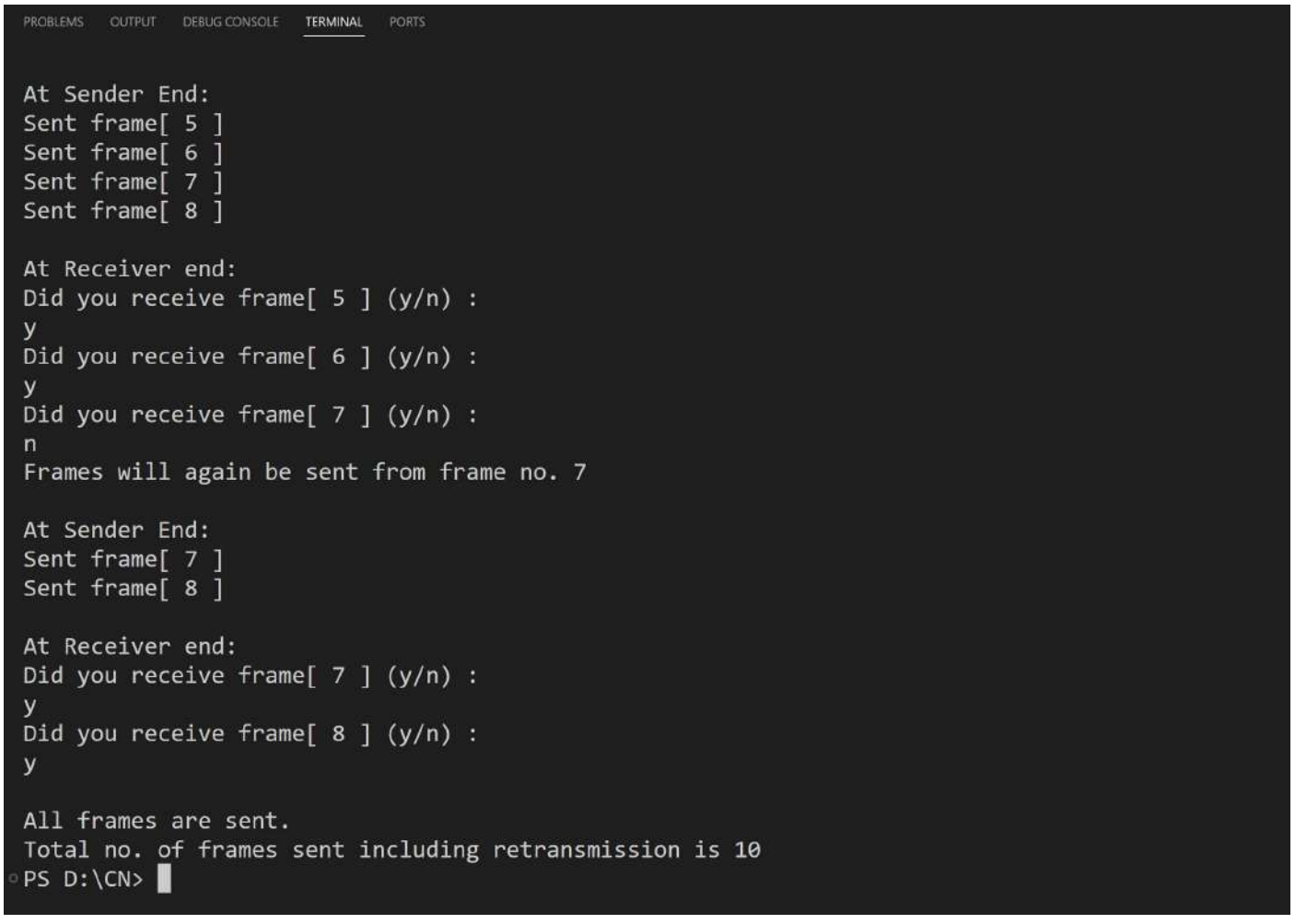
```
            else:
                break

        print()

        # Receiver side
        print("At Receiver end:")
        j = 0
        count = 0
        for i in range(ptr_on_window_left_receiver, total_frames):
            if count < window_size:
                print("Did you receive frame[", i + 1, "] (y/n) : ")
                y_n = input()
                if y_n == 'n':
                    print("Frames will again be sent from frame no.", i + 1)
                    print()
                    ptr_on_window_left_sender = i
                    break
                else:
                    j += 1
                    ptr_on_window_left_receiver += 1
                count += 1
            else:
                break

        if j == window_size:
            print("All Frames from this window sent without errors. Sending next frames...")
            print()

    print()
    print("All frames are sent.")
    print("Total no. of frames sent including retransmission is", total_sent_frames)

if __name__ == "__main__":
```

main()

OUTPUT



```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS D:\CN> python -u "d:\CN\GoBackN.py"
Enter window size:
4
Enter total frames to be sent:
8
0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
Do you want to start sending frames (0/1):
1

At Sender End:
Sent frame[ 1 ]
Sent frame[ 2 ]
Sent frame[ 3 ]
Sent frame[ 4 ]

At Receiver end:
Did you receive frame[ 1 ] (y/n) :
y
Did you receive frame[ 2 ] (y/n) :
y
Did you receive frame[ 3 ] (y/n) :
y
Did you receive frame[ 4 ] (y/n) :
y
All Frames from this window sent without errors. Sending next frames...
```



```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS


At Sender End:
Sent frame[ 5 ]
Sent frame[ 6 ]
Sent frame[ 7 ]
Sent frame[ 8 ]

At Receiver end:
Did you receive frame[ 5 ] (y/n) :
y
Did you receive frame[ 6 ] (y/n) :
y
Did you receive frame[ 7 ] (y/n) :
n
Frames will again be sent from frame no. 7

At Sender End:
Sent frame[ 7 ]
Sent frame[ 8 ]

At Receiver end:
Did you receive frame[ 7 ] (y/n) :
y
Did you receive frame[ 8 ] (y/n) :
y

All frames are sent.
Total no. of frames sent including retransmission is 10
PS D:\CN>
```

Suresh
C3 C32
2103164

EXPERIMENT NO - 5

CN - C32 - 2103164

Aim:- Build a simple topology and configure it for static routing protocol using packet tracer. Setup a network and configure IP addressing, subnetting, masking.

Theory:-

Network topology refers to physical or logical layout of devices and connection within a network. Different network topologies are suited. applications and have unique advantages and disadvantages. Here are a few different network topologies along with small examples of their application

Star Topology:-

Description: In star topology, all devices are connected to central hub or switch. Devices do not connect directly connect directly to each other.
Application:- Star topology is mainly used in home network where all devices connect to central router or switch. This topology is straightforward to set up and manage.

Bus Topology:-

Description:- In bus topology, all devices share a single communication, line (bus). Data bars -

milted down the bus and devices can "tap
to acss the data.

Application :- Bus topologies were historically used in early ethernet networks. While they are less common today, they can still be found in application like industrial control systems.

Ring Topology :-

Description :- In ring topology, each devices is conn to exactly two other devices, forming a closed loop. Data travels in one direction around the ring.

Application: Ring Topology are less common in modern LANs but are used in specific applica like fiber optic network. Token Ring Network used this topology in the past.

Mesh Topology :-

Description :- In mesh topology, every devices is connected to exactly two either two device, creating multiple paths for data to travel.

Application :- Mesh topologies are often used in critical application like data centers, where redundancy and fault tolerance are crucial. Each device can communicate directly with any other, enhancing reliability.

CN-CS2-2103164

Hybrid Topology :-

Description :- A hybrid topology combines two or more different topologies into a network. For example, a network might have star topology in one office and ring topology in another office connected through single network. router.

Application : Hybrid topologies allow organization to tailor their network to specific needs. They are commonly found in larger enterprises with diverse networking requirements.

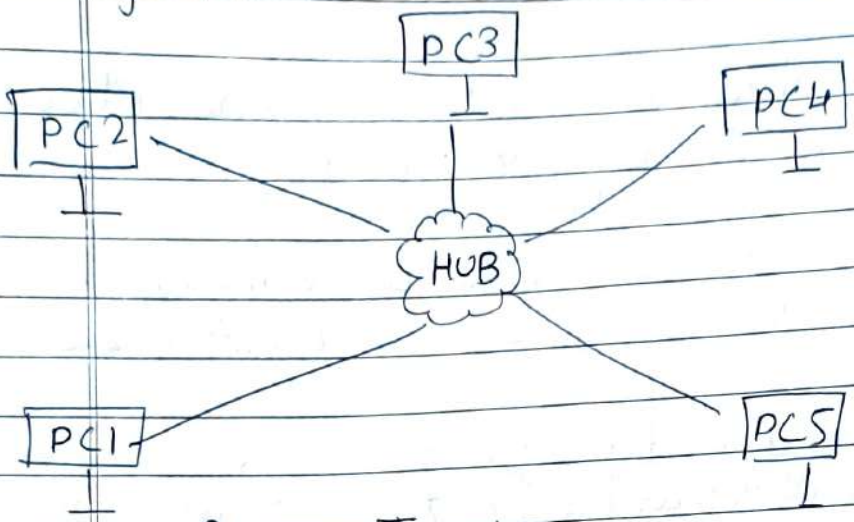
Tree Topology :-

Description :-

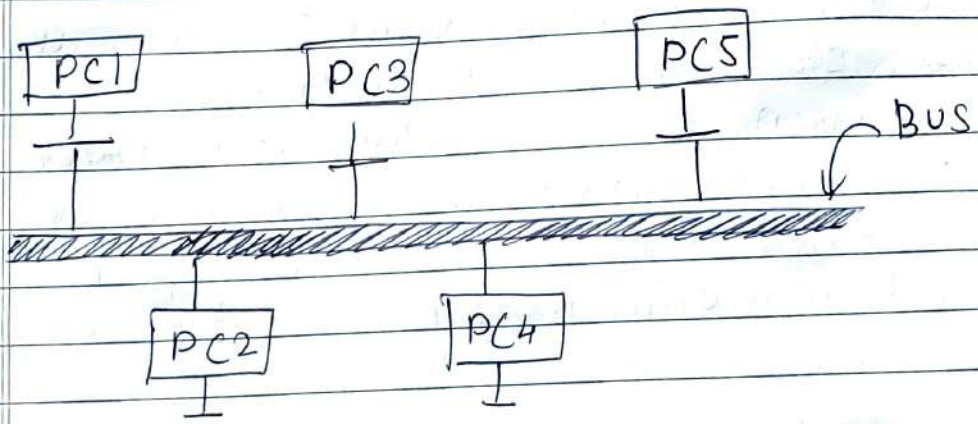
Also known as Hierachical Topology, there are star connected to central hub.

Application :- Tree topologies are often used in large scale network, such as cooporate networks with multiple branches. Each branch office may have star topology, they all connected to main office using a heirachical structure.

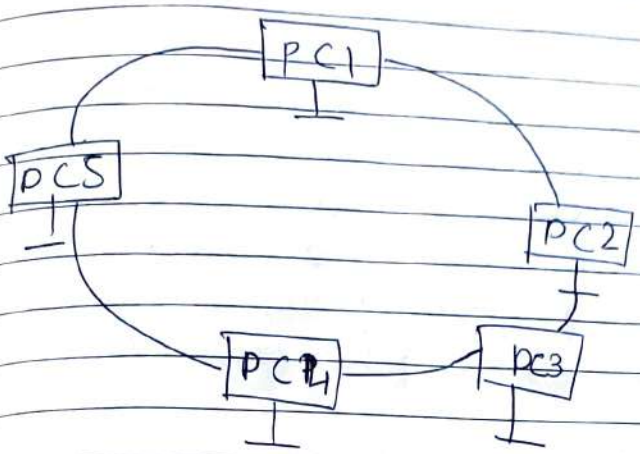
Diagrams :-



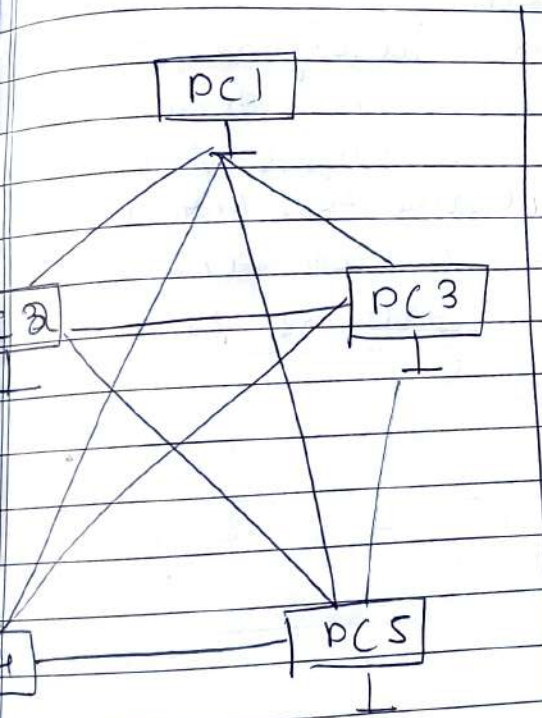
Star Topology



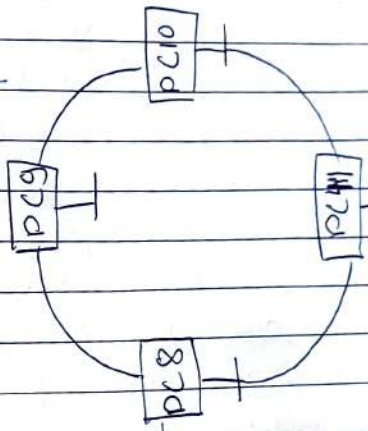
Bus Topology

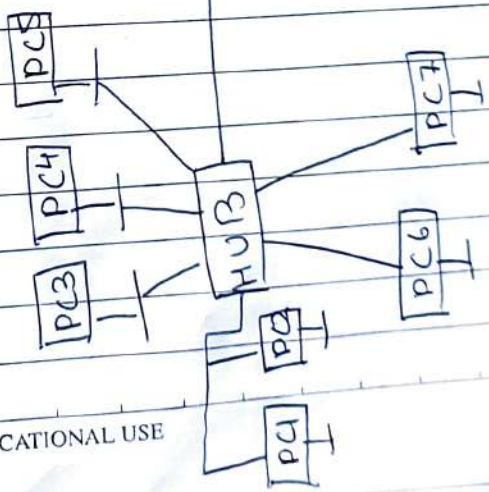


Ring Topology



Mesh Topology





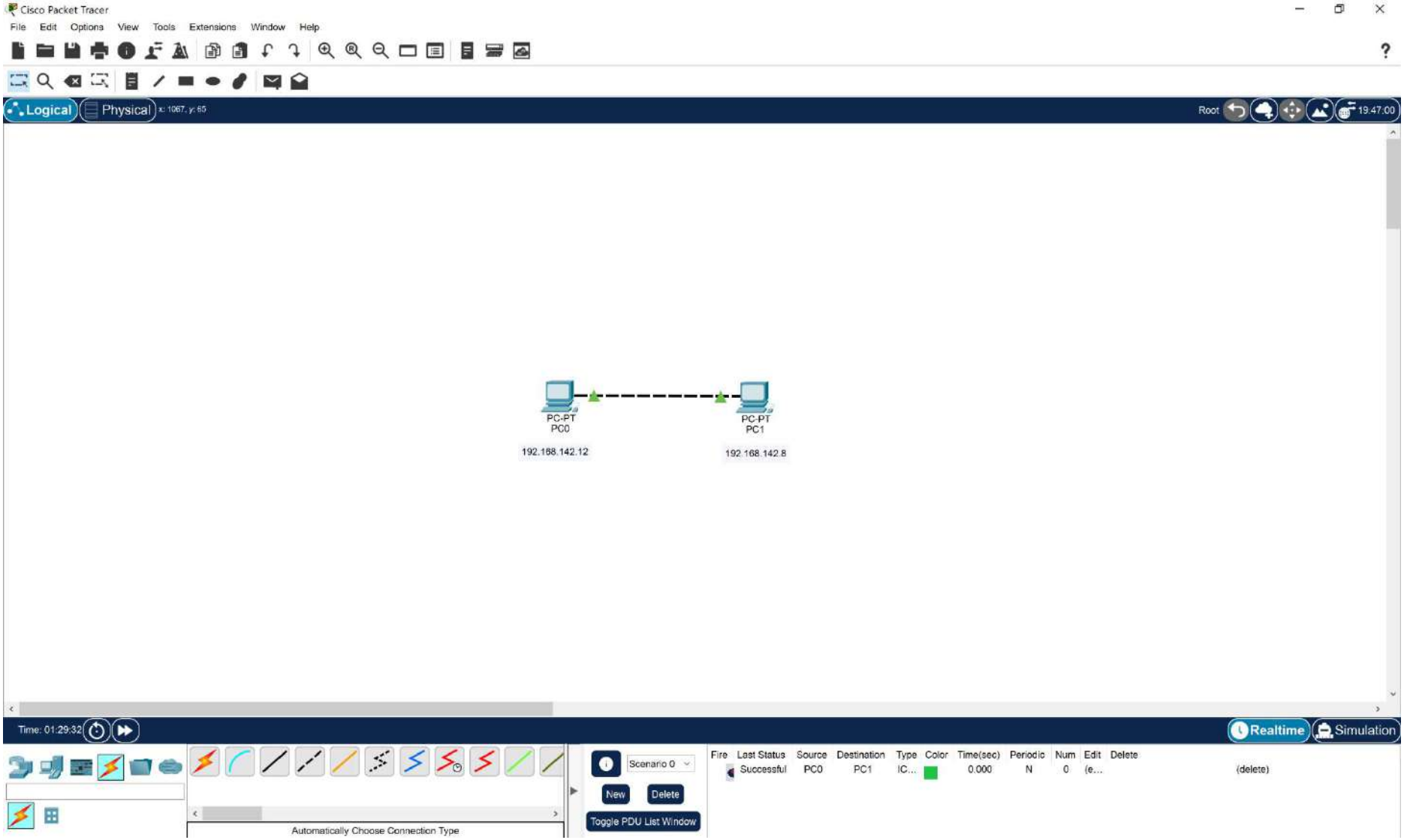
Hybrid Topology

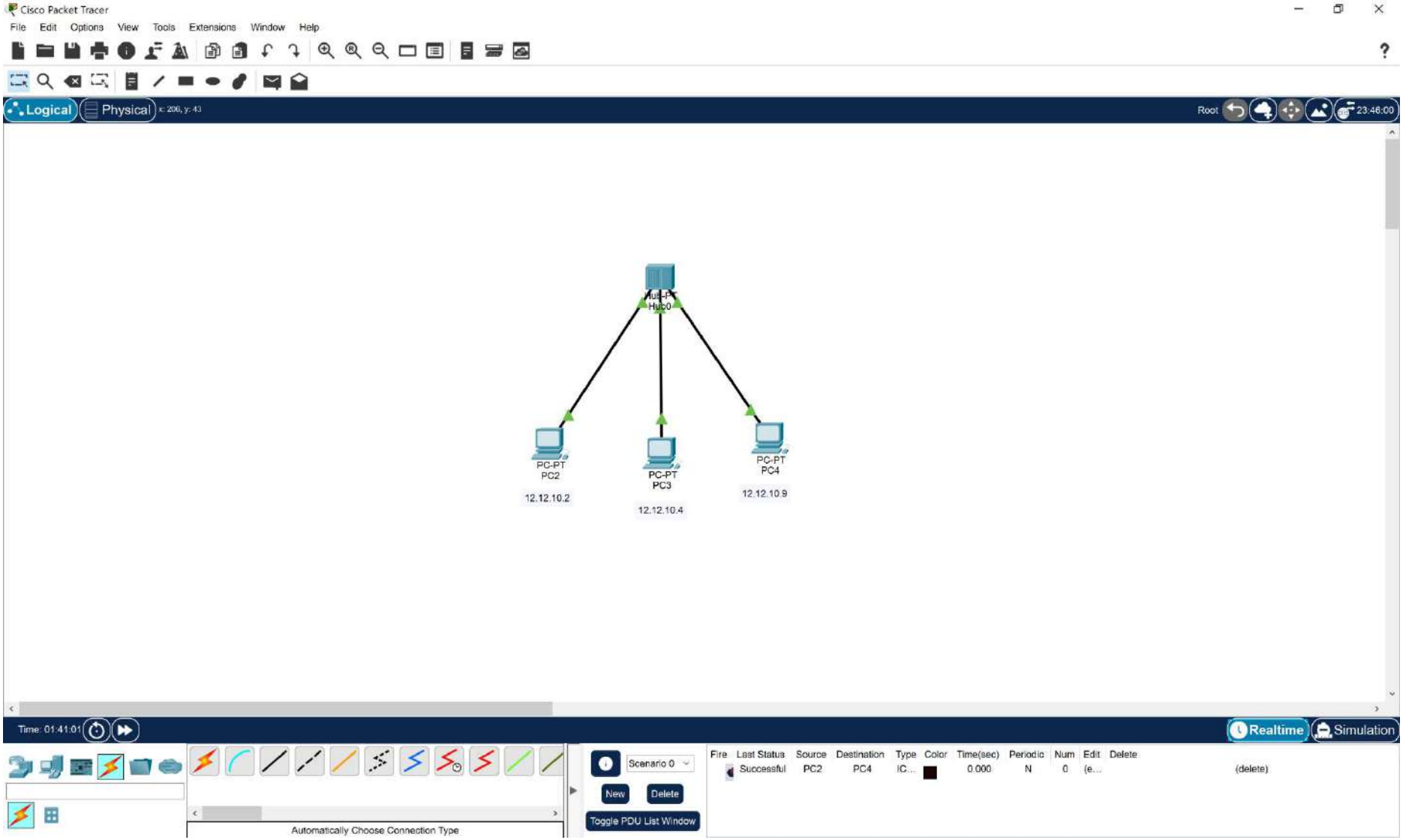
* Cisco packet Tracer :-

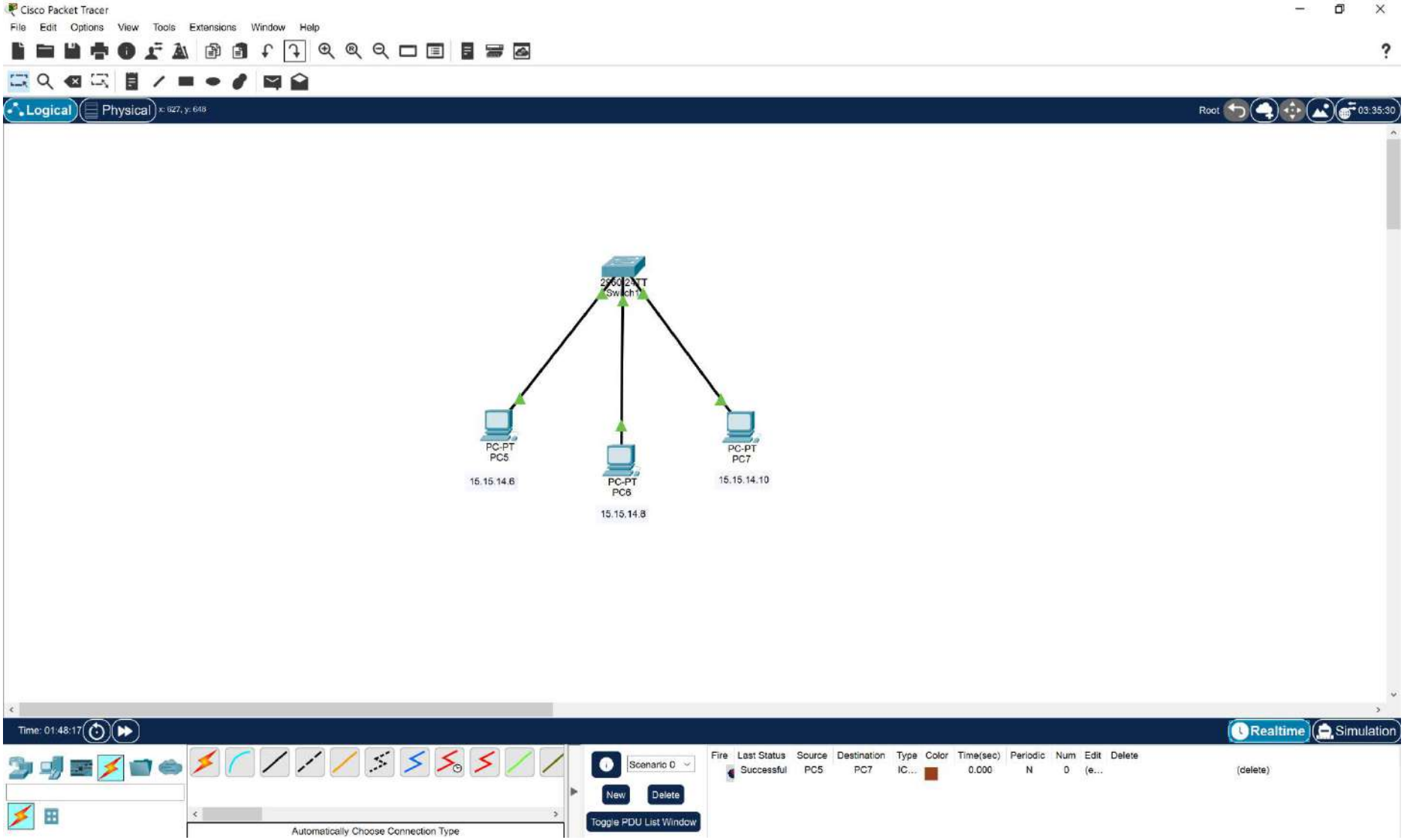
Cisco packet Tracer is a versatile networ simulation tool developed by Cisco Systems. I serves as a valuable resource for both networking professionals and learners, enablin them to design, configure, and test network setups in virtual environment.

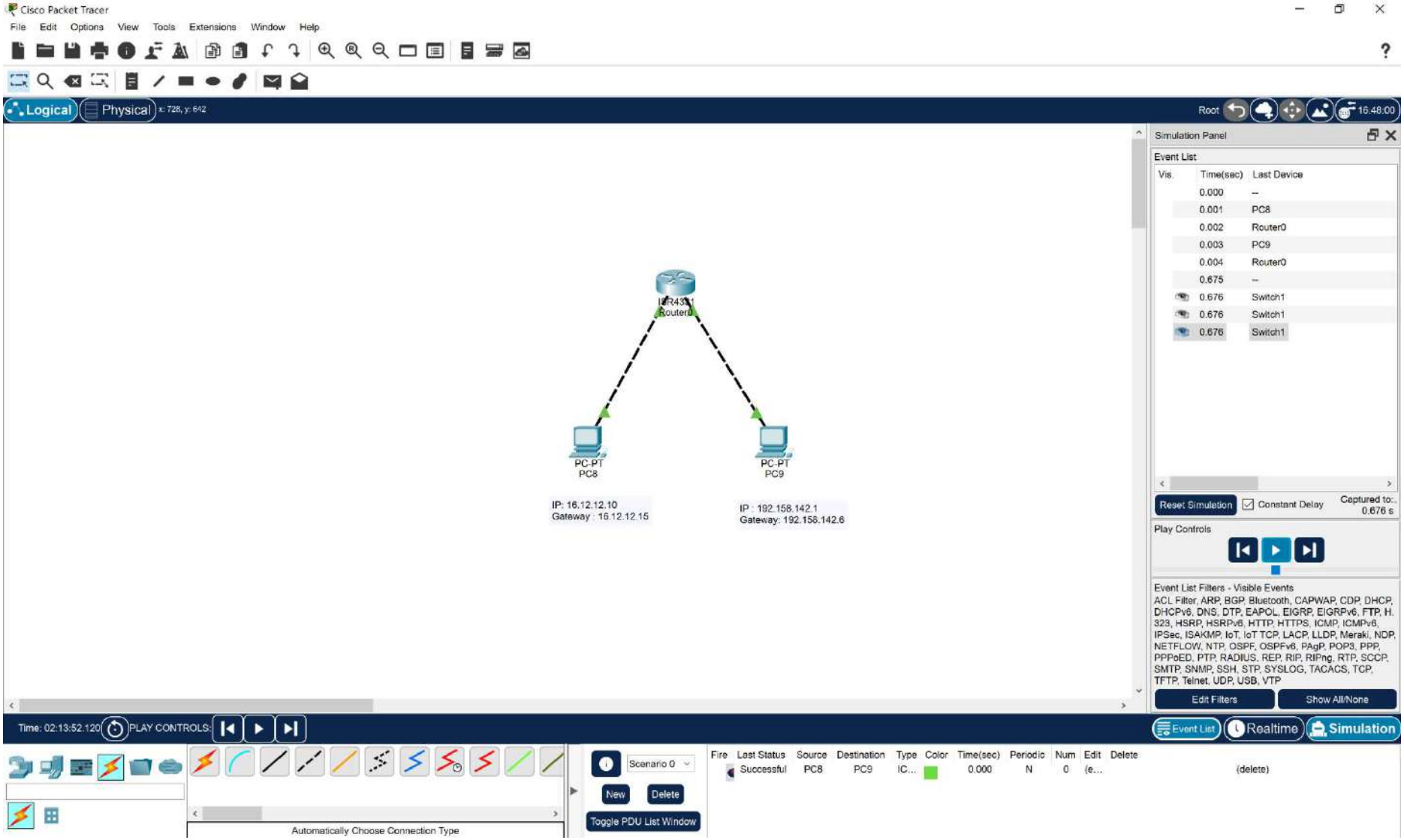
Packet Tracer offer a user friendly interfac and wide range of Cisco devices and networ components, allowing users to build and exper with various network topologies, troublesho issues and gain pratical experience without need of physical hardware. Its commonly used in educational settings to teach netwo concept and is valuable tool for gaining hand on experience in networking before working with real-world equipment.

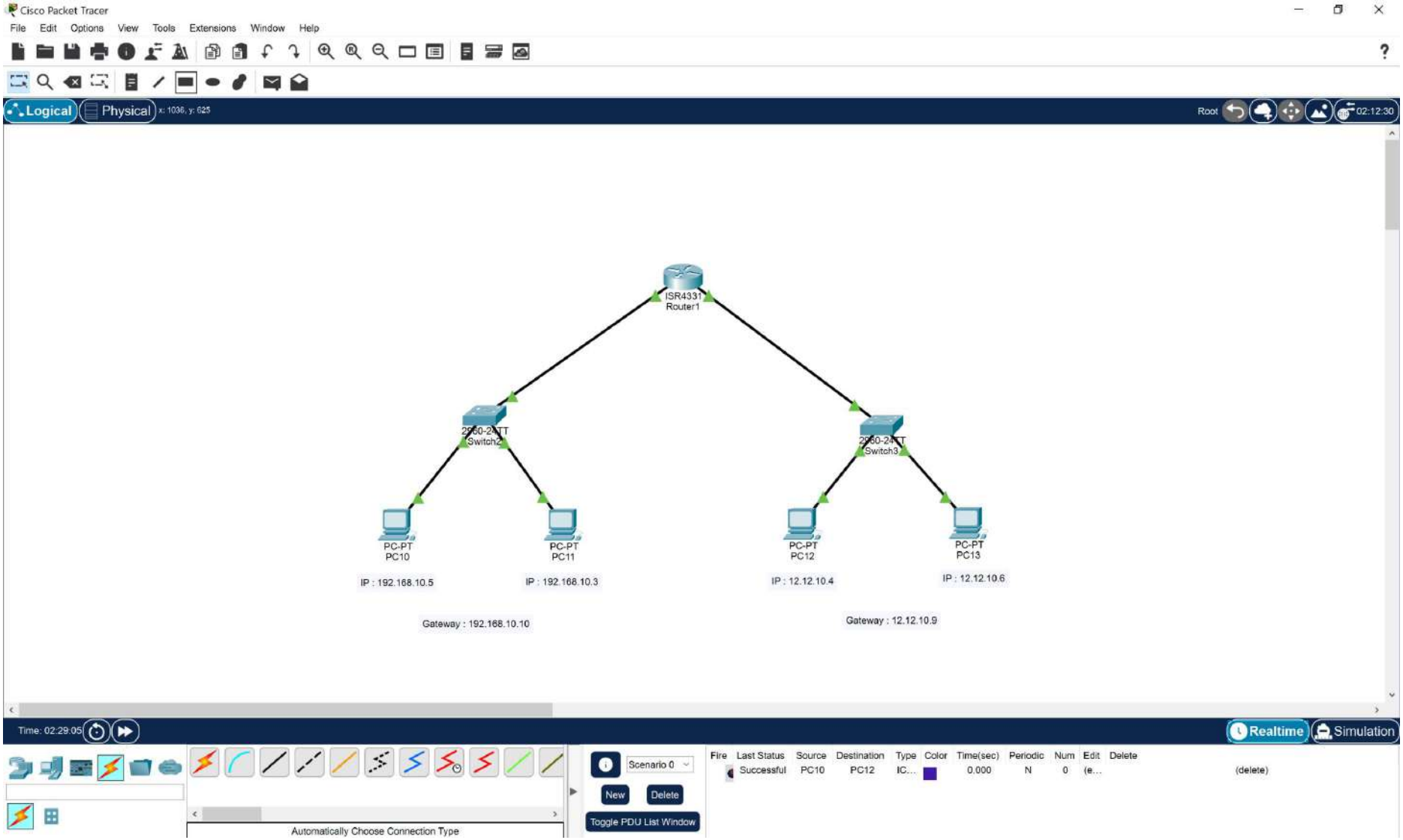
03/10/23


Cisco Packet Tracer
File Edit Options View Tools Extensions Window Help
Logical
Physical
Root
19:47:00
PC-PT
PC0
192.168.142.12
PC-PT
PC1
192.168.142.8
Time: 01:29:32
Realtime
Simulation
Automatically Choose Connection Type
Scenario 0
New
Delete
Toggle PDU List Window
Fire Last Status Source Destination Type Color Time(sec) Periodic Num Edit Delete
Successful PC0 PC1 IC... 0.000 N 0 (e... (delete)


Cisco Packet Tracer
File Edit Options View Tools Extensions Window Help
Logical
Physical
Root
23:46:00
Multi-PT Hub0
PC-PT PC2
12.12.10.2
PC-PT PC3
12.12.10.4
PC-PT PC4
12.12.10.9
Time: 01:41:01
Realtime
Simulation
Automatically Choose Connection Type
Scenario 0
New
Delete
Toggle PDU List Window
Fire Last Status Source Destination Type Color Time(sec) Periodic Num Edit Delete
Successful PC2 PC4 IC... 0.000 N 0 (e... (delete)


Cisco Packet Tracer
File Edit Options View Tools Extensions Window Help
Logical
Physical
Root
03:35:30
PC-PT PC5
15.15.14.6
PC-PT PC6
15.15.14.8
PC-PT PC7
15.15.14.10
Time: 01:48:17
Realtime
Simulation
Scenario 0
New
Delete
Toggle PDU List Window
Fire Last Status Source Destination Type Color Time(sec) Periodic Num Edit Delete
Successful PC5 PC7 IC... 0.000 N 0 (e... (delete)
Automatically Choose Connection Type


Cisco Packet Tracer
File Edit Options View Tools Extensions Window Help
Logical
Physical
Root
16:48:00
Router0
PC-PT
PC8
PC-PT
PC9
IP: 16.12.12.10
Gateway : 16.12.12.15
IP : 192.158.142.1
Gateway: 192.158.142.6
Simulation Panel
Event List
Vis. Time(sec) Last Device
0.000 --
0.001 PC8
0.002 Router0
0.003 PC9
0.004 Router0
0.675 --
0.676 Switch1
0.676 Switch1
0.676 Switch1
Reset Simulation
Constant Delay
Captured to:
0.676 s
Play Controls
Event List Filters - Visible Events
ACL Filter, ARP, BGP, Bluetooth, CAPWAP, CDP, DHCP, DHCPv6, DNS, DTP, EAPOL, EIGRP, EIGRPv6, FTP, H.323, HSRP, HSRPv6, HTTP, HTTPS, ICMP, ICMPv6, IPSec, ISAKMP, IoT, IoT TCP, LACP, LLDP, Meraki, NDP, NETFLOW, NTP, OSPF, OSPFv6, PAgP, POP3, PPP, PPPoED, PTP, RADIUS, REP, RIP, RIPng, RTP, SCCP, SMTP, SNMP, SSH, STP, SYSLOG, TACACS, TCP, TFTP, Telnet, UDP, USB, VTP
Edit Filters
Show All/None
Time: 02:13:52.120
PLAY CONTROLS:
Event List
Realtime
Simulation
Automatically Choose Connection Type
Scenario 0
New
Delete
Toggle PDU List Window
Fire Last Status Source Destination Type Color Time(sec) Periodic Num Edit Delete
Successful PC8 PC9 IC... 0.000 N 0 (e... (delete)


Cisco Packet Tracer
File Edit Options View Tools Extensions Window Help
Logical
Physical
Root
02:12:30
ISR4331
Router1
2960-24TT
Switch2
2960-24TT
Switch3
PC-PT
PC10
PC-PT
PC11
PC-PT
PC12
PC-PT
PC13
IP : 192.168.10.5
IP : 192.168.10.3
IP : 12.12.10.4
IP : 12.12.10.6
Gateway : 192.168.10.10
Gateway : 12.12.10.9
Time: 02:29:05
Realtime
Simulation
Scenario 0
New
Delete
Toggle PDU List Window
Fire
Last Status
Source
Destination
Type
Color
Time(sec)
Periodic
Num
Edit
Delete
Successful
PC10
PC12
0.000
N
0
(delete)
Automatically Choose Connection Type

THADOMAL SHAHANI TSEC ENGINEERING COLLEGE

Sherish
C3 C32
2103164

Experiment - 6

CN - C32 - 2103164

Aim:- Design VPN & configure RIP/OSPF using packet tracer.

Theory:-

A VPN or Virtual Private Network, is a technology that allows you to creat a secure and encrypted connection over a less secure network, such as the internet. VPNs are commonly used to provide privacy and security for data transmission over the internet. Here's how a VPN works in context of computer networking.

1.) Encryption and Tunneling:- When you connect to VPN, your computer or device creat a secure "tunnel" over the internet to remote server operated by VPN service provider. The tunnel is encrypted, which mean that all data travelling between your device and VPN server is encrypted and secure from eavedropping.

2.) Hide Your IP Address:- Your IP address like your online identity, and it can reveal your location and other information about you. When you connect a VPN, your real IP address is hidden and VPN address is used. This help protect your privacy & make it harder for websites and other online

services to track your location.

3) Secure Data Transmission :- Any data you send and receive while connected to VPN is encrypted. This includes web browsing, email, file transfer and any other online activities. This encryption makes it extremely difficult for hackers or malicious actors to intercept and read your data.

4) Access to Restricted Content :- VPNs can be used to bypass geo-restrictions. For example, if a website or streaming service is only available in certain countries, you connect to VPN server in that country to access the content of if you were physically located there.

5) Security and Privacy :- VPN provide an additional layer and privacy when using public Wi-Fi network, such as those in coffee shops or airports. They protect your data from physically located here.

6) Bussiness Use :- VPNs are commonly used by bussiness to provide secure remote access to their internal networks for employees working from home or while travelling. This ensures that sensitive company data remains secure even when accessed from outside the office.

7) Types Of VPNs :- There are several types of VPNs including remote access VPN, site to site VPN, client-to-site VPNs. Each type has its own use case & implementation.

5.) VPN service Providers - To use VPN you typically need to subscribe VPN service provider. These providers maintain a network of VPN server in various location around the world.

* RIP/OSPF

- RIP stands for Routing Information Protocol and OSPF stands for open Shortest path first are two different routing protocols used in computer networks to determine how data packets should be formacted from one network device to another. Both protocols are used to manage routing tables and facilitate efficient data transmission within a network, but they have significant difference in terms of operations and complexity

RIP

(1) Distance-vector Protocol: RIP is distance vector routing protocol. It determines the best path to distination by counting the number of hops (routers) between the source and destination. Each router periodically sends its routing table to its neighbours.

(2) Hop Count Metric :- RIP uses hop count as it metric for route selection. A route with fewer hops is considered better. However, RIP is limited to a maximum hop count of 15, which restricts its scalability.

(3) Convergence Time :- RIP has relatively slow converge times, when network topology changes occurs, it can take a noticeable amount of time for routers to update their routing table & converge to a stable state.

* OSPF

(1) Link-state protocol :- OSPF is a link state routing protocol. It builds and maintains a detailed database of the entire network topology, including information about the link & routers. This database is used to count the shortest path to reach any dest

(2) Cost metric :- OSPF uses a cost metric, typically based on bandwidth to determine the best path to destination. Lower costs represent better paths. This allow OSPF to take account factors like link bandwidth & latency when making routing decisions.

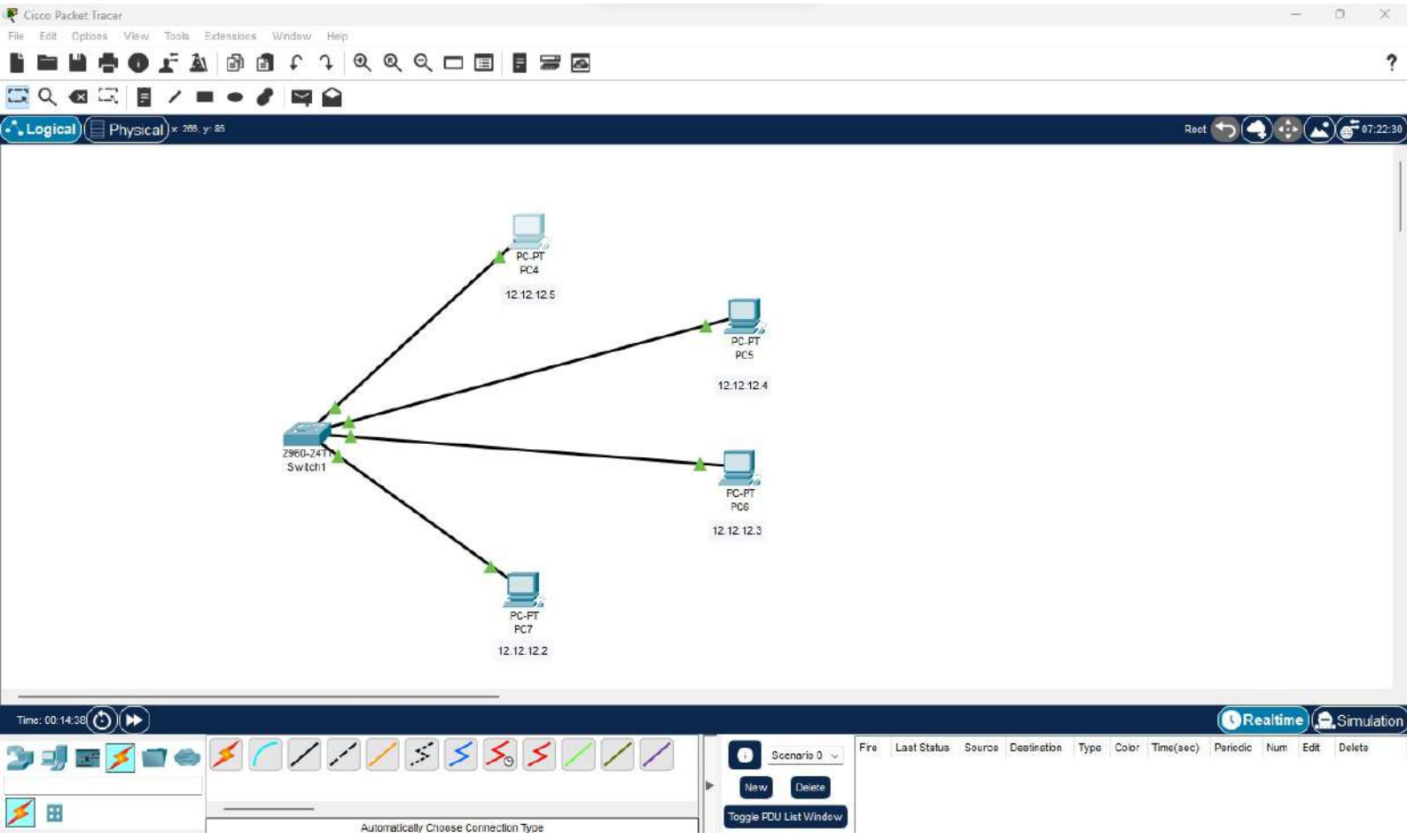
(3) Convergence Time :- OSPF generally has fast convergence times compared to RIP. When network changes occur, router quickly make it suitable for large and dynamic networks

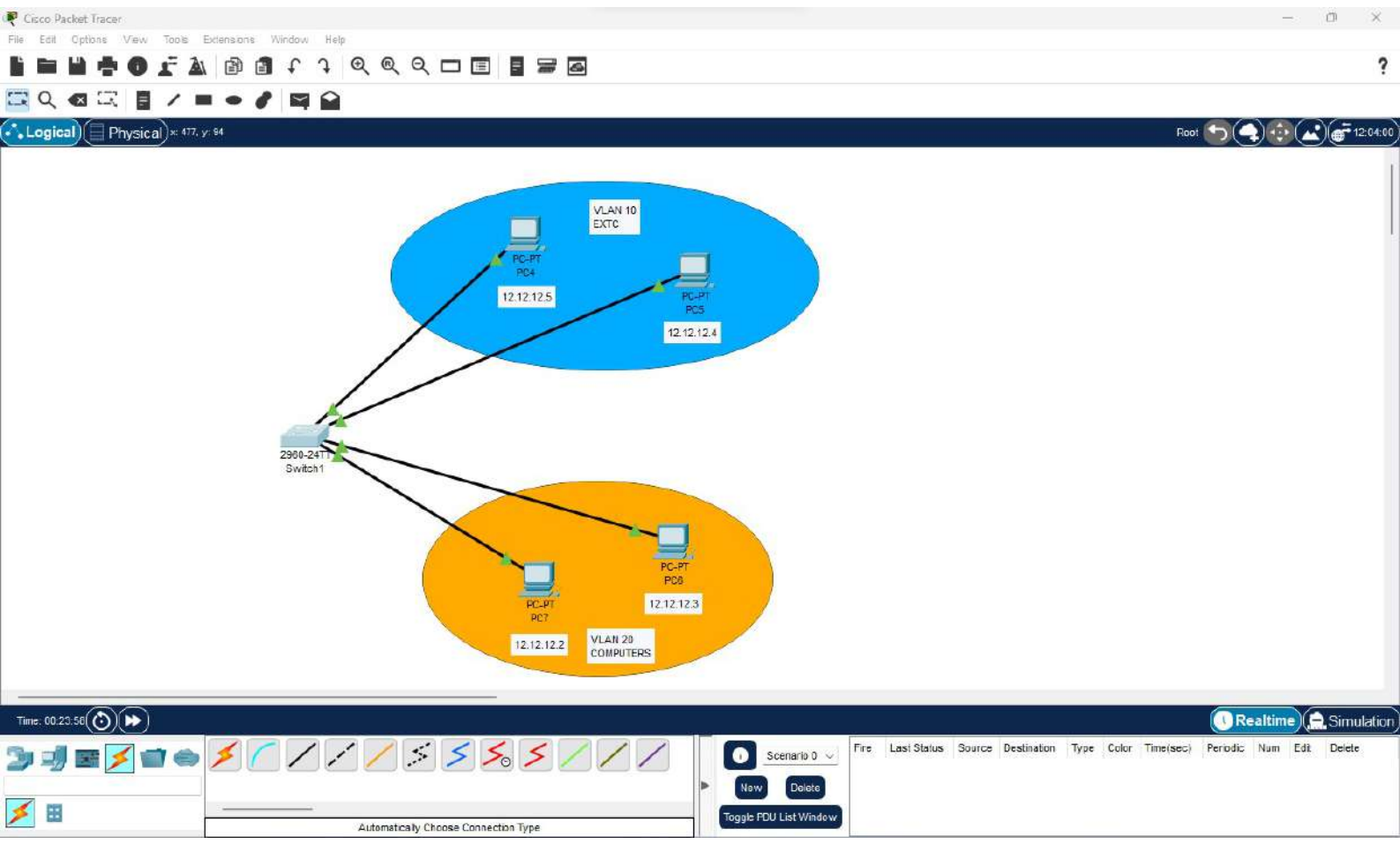
(4) Commonly used in Enterprise Network :- OSPF is commonly used in enterprise networks and service provider environment for its flexibility and robustness

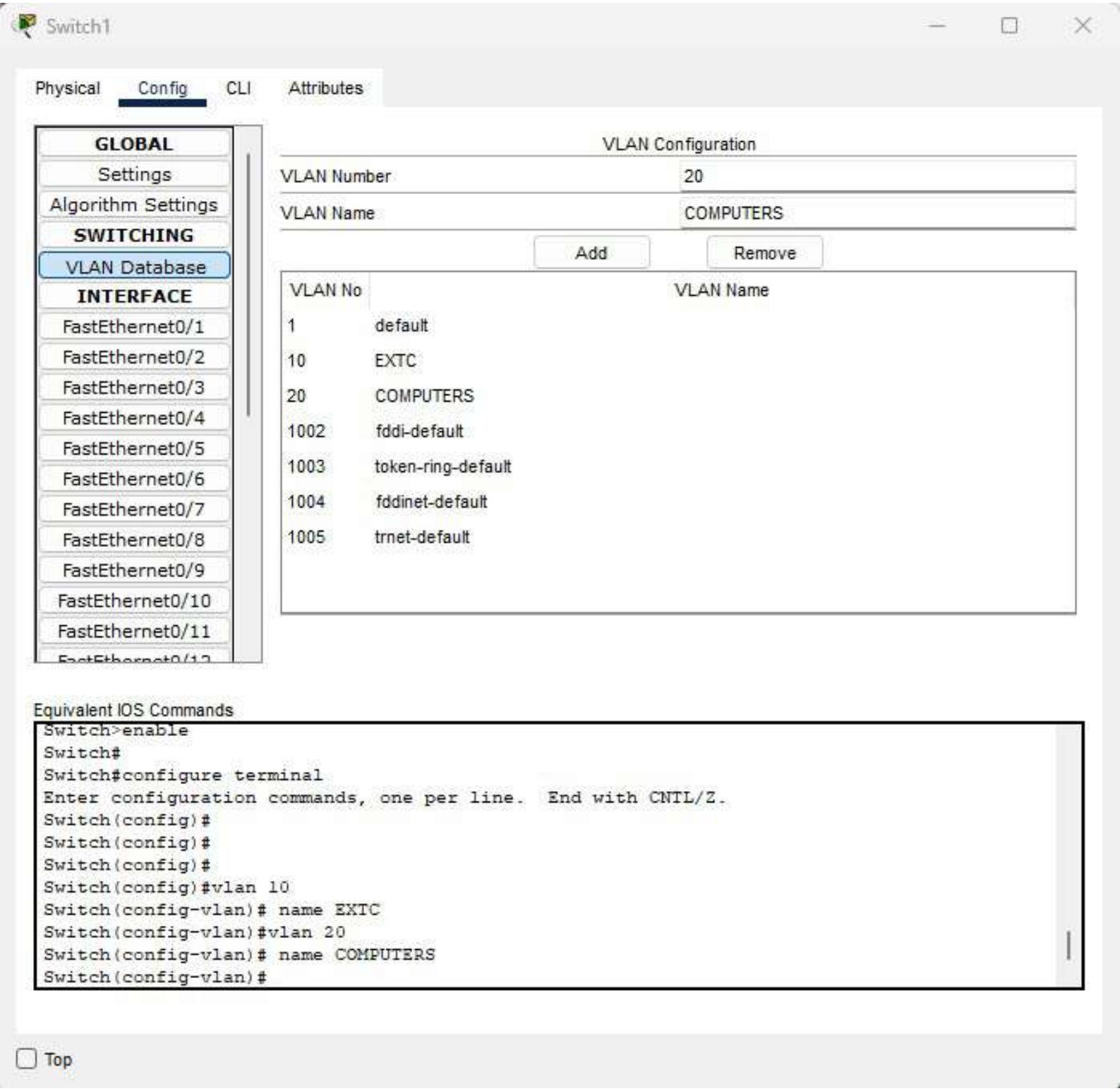
CN-C32-2103164

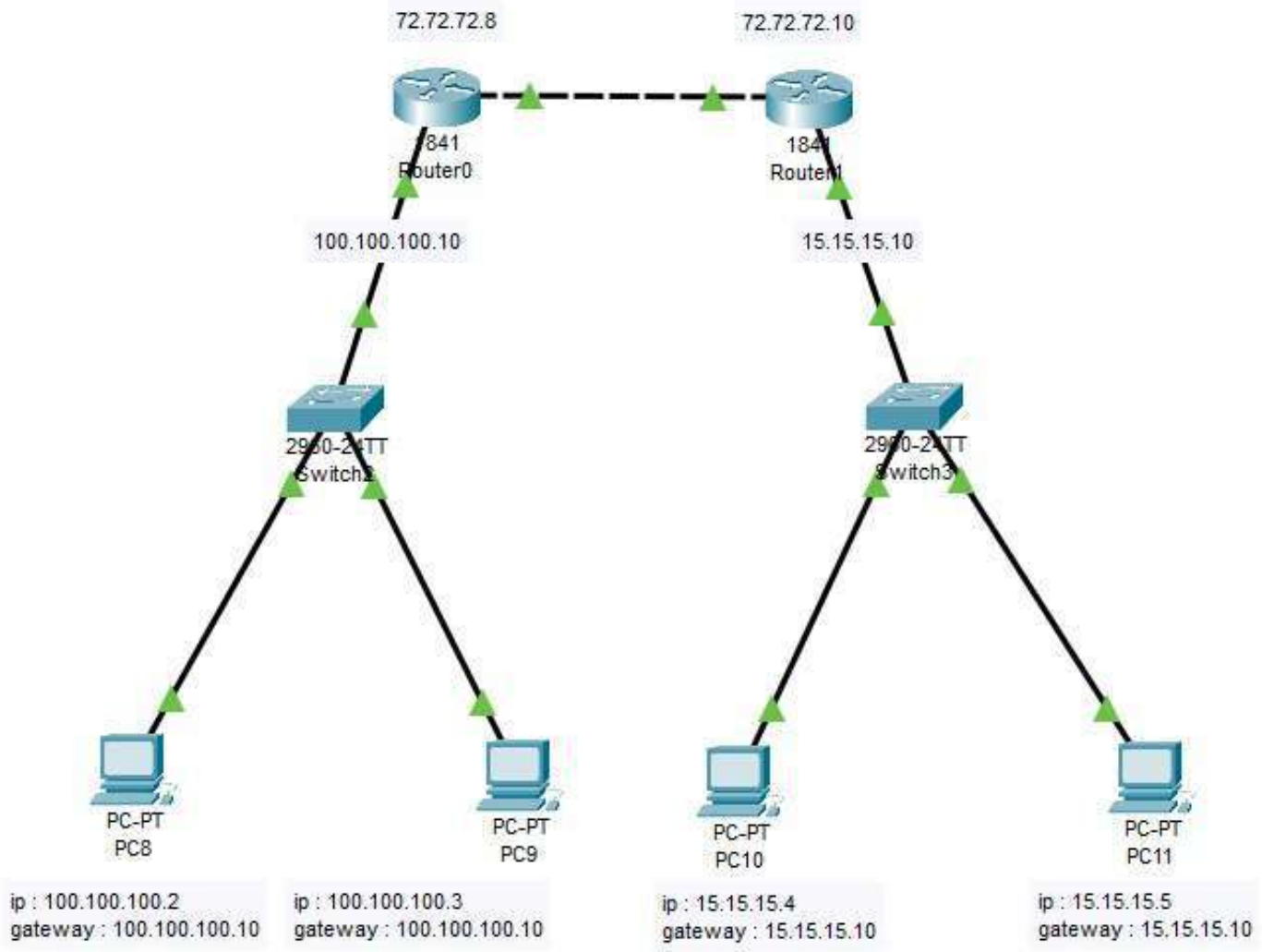
4) LSA (Link State Advertisment) - Routers exchange LSAs to share information about their link and network topology.

5) VLSM and CIDR :- OSPF is commonly used in enterprise networks and service provider environment for its flexibility and enchancing network scalability.

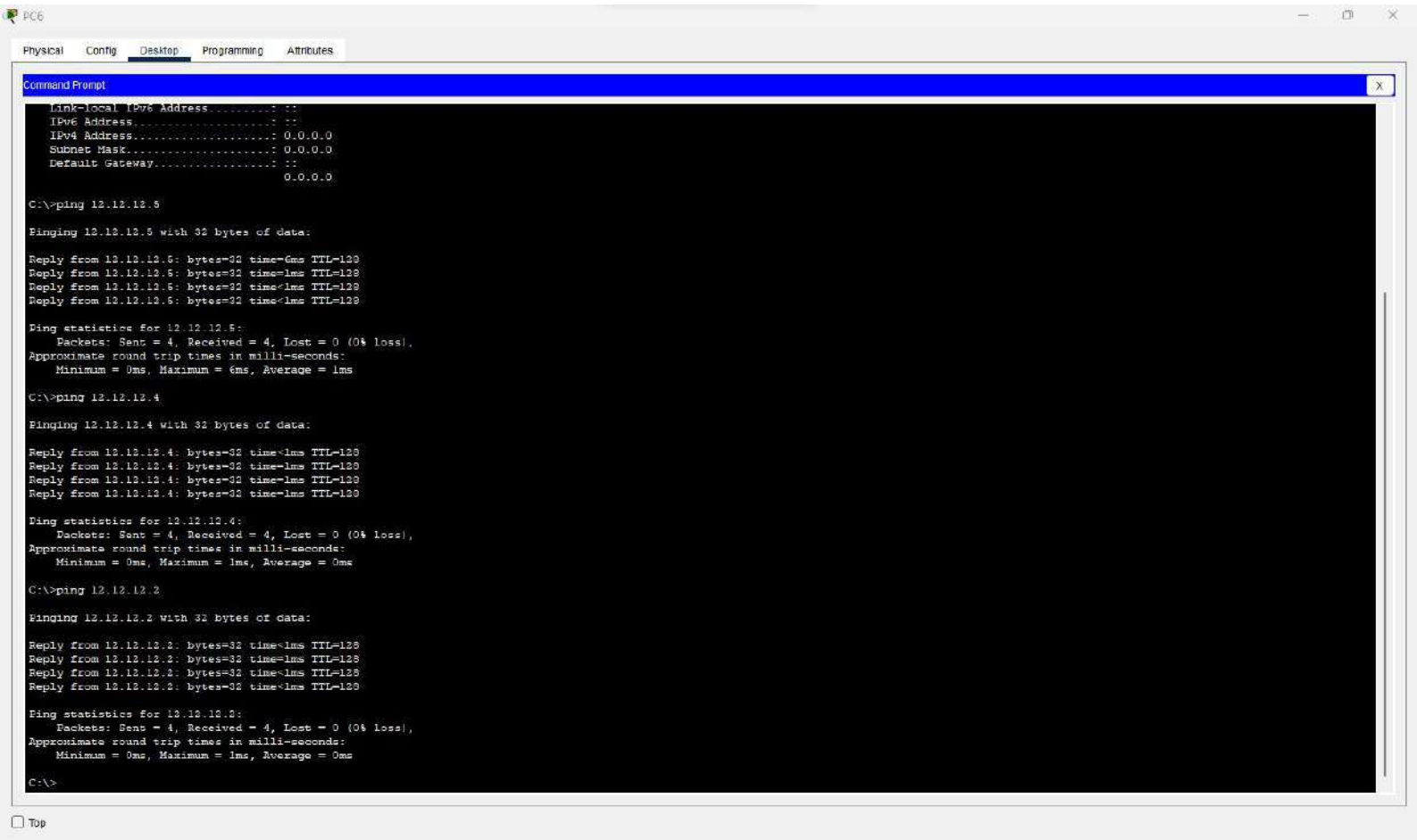

Cisco Packet Tracer
File Edit Options View Tools Extensions Window Help
Logical
Physical
Root
07:22:30
PC-PT
PC4
12.12.12.5
PC-PT
PC5
12.12.12.4
2960-24TT
Switch1
PC-PT
PC6
12.12.12.3
PC-PT
PC7
12.12.12.2
Time: 00:14:38
Realtime
Simulation
Scenario 0
New
Delete
Toggle PDU List Window
Fire
Last Status
Source
Destination
Type
Color
Time(sec)
Periodic
Num
Edit
Delete
Automatically Choose Connection Type


Cisco Packet Tracer
File Edit Options View Tools Extensions Window Help
Logical
Physical
x: 477, y: 94
Root
12:04:00
VLAN 10
EXTC
PC-PT
PC4
12.12.12.5
PC-PT
PC5
12.12.12.4
2960-24TT
Switch1
PC-PT
PC8
12.12.12.3
PC-PT
PC7
12.12.12.2
VLAN 20
COMPUTERS
Time: 00:23:58
Realtime
Simulation
Scenario 0
New
Delete
Toggle PDU List Window
Fire
Last Status
Source
Destination
Type
Color
Time(sec)
Periodic
Num
Edit
Delete
Automatically Choose Connection Type


Switch1
Physical
Config
CLI
Attributes
GLOBAL
Settings
Algorithm Settings
SWITCHING
VLAN Database
INTERFACE
FastEthernet0/1
FastEthernet0/2
FastEthernet0/3
FastEthernet0/4
FastEthernet0/5
FastEthernet0/6
FastEthernet0/7
FastEthernet0/8
FastEthernet0/9
FastEthernet0/10
FastEthernet0/11
VLAN Configuration
VLAN Number
20
VLAN Name
COMPUTERS
Add
Remove
VLAN No
VLAN Name
1 default
10 EXTC
20 COMPUTERS
1002 fddi-default
1003 token-ring-default
1004 fddinet-default
1005 trnet-default
Equivalent IOS Commands
Switch>enable
Switch#
Switch#configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
Switch(config)#
Switch(config)#
Switch(config)#
Switch(config)#vlan 10
Switch(config-vlan)# name EXTC
Switch(config-vlan)#vlan 20
Switch(config-vlan)# name COMPUTERS
Switch(config-vlan)#
Top


72.72.72.8
72.72.72.10
1841
Router0
Router1
100.100.100.10
15.15.15.10
Switch2
Switch3
PC-PT
PC8
PC-PT
PC9
PC-PT
PC10
PC-PT
PC11
ip : 100.100.100.2
gateway : 100.100.100.10
ip : 100.100.100.3
gateway : 100.100.100.10
ip : 15.15.15.4
gateway : 15.15.15.10
ip : 15.15.15.5
gateway : 15.15.15.10

```
C:\>ping 15.15.15.4

Pinging 15.15.15.4 with 32 bytes of data:

Reply from 100.100.100.10: Destination host unreachable.
Reply from 100.100.100.10: Destination host unreachable.
Reply from 100.100.100.10: Destination host unreachable.
Reply from 100.100.100.10: Destination host unreachable.

Ping statistics for 15.15.15.4:
    Packets: Sent = 4, Received = 0, Lost = 4 (100% loss),

C:\>ping 15.15.15.4
```


PC6
Physical
Config
Desktop
Programming
Attributes
Command Prompt
X
Link-local IPv6 Address.....: ::
IPv6 Address.....: ::
IPv4 Address.....: 0.0.0.0
Subnet Mask.....: 0.0.0.0
Default Gateway.....: ::
0.0.0.0
C:\>ping 12.12.12.5
Pinging 12.12.12.5 with 32 bytes of data:
Reply from 12.12.12.5: bytes=32 time=6ms TTL=128
Reply from 12.12.12.5: bytes=32 time=1ms TTL=128
Reply from 12.12.12.5: bytes=32 time<1ms TTL=128
Reply from 12.12.12.5: bytes=32 time<1ms TTL=128
Ping statistics for 12.12.12.5:
Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
Minimum = 0ms, Maximum = 6ms, Average = 1ms
C:\>ping 12.12.12.4
Pinging 12.12.12.4 with 32 bytes of data:
Reply from 12.12.12.4: bytes=32 time<1ms TTL=128
Reply from 12.12.12.4: bytes=32 time=1ms TTL=128
Reply from 12.12.12.4: bytes=32 time=1ms TTL=128
Reply from 12.12.12.4: bytes=32 time=1ms TTL=128
Ping statistics for 12.12.12.4:
Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
Minimum = 0ms, Maximum = 1ms, Average = 0ms
C:\>ping 12.12.12.2
Pinging 12.12.12.2 with 32 bytes of data:
Reply from 12.12.12.2: bytes=32 time<1ms TTL=128
Reply from 12.12.12.2: bytes=32 time=1ms TTL=128
Reply from 12.12.12.2: bytes=32 time<1ms TTL=128
Reply from 12.12.12.2: bytes=32 time<1ms TTL=128
Ping statistics for 12.12.12.2:
Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
Minimum = 0ms, Maximum = 1ms, Average = 0ms
C:\>
Top

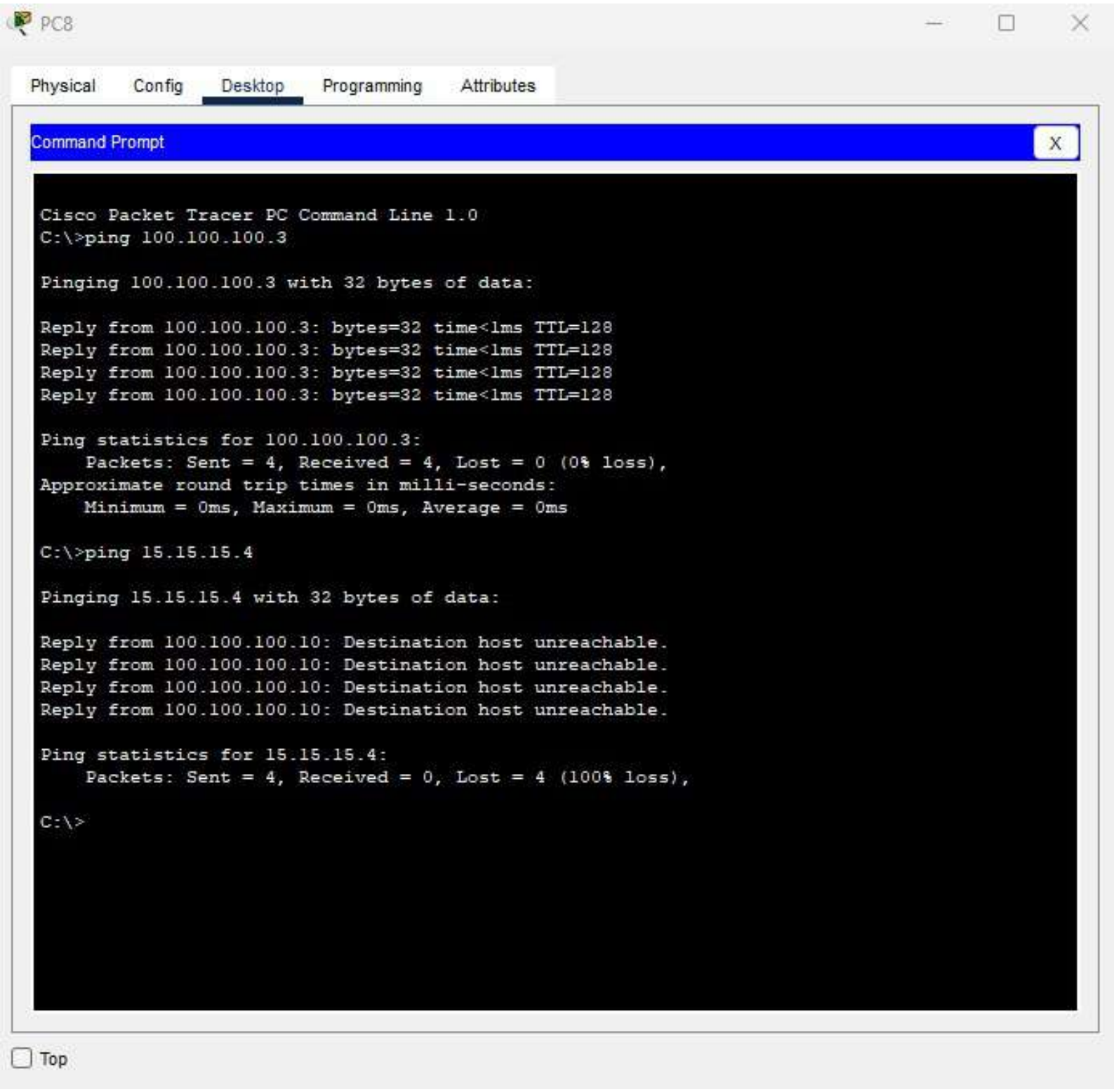
PC6

Physical | Config | Desktop | Programming | Attributes

Command Prompt

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

FastEthernet0 Connection:(default port)

   Connection-specific DNS Suffix..:
   Link-local IPv6 Address.....: FE80::202:4AFF:FE0A:7B1
   IPv6 Address.....: ::
   IPv4 Address.....: 12.12.12.3
   Subnet Mask.....: 255.0.0.0
   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

C:\>ping 12.12.12.5

Pinging 12.12.12.5 with 32 bytes of data:

Reply from 12.12.12.5: bytes=32 time=6ms TTL=128
Reply from 12.12.12.5: bytes=32 time=1ms TTL=128
Reply from 12.12.12.5: bytes=32 time<1ms TTL=128
Reply from 12.12.12.5: bytes=32 time<1ms TTL=128

Ping statistics for 12.12.12.5:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 6ms, Average = 1ms

C:\>ping 12.12.12.4

Pinging 12.12.12.4 with 32 bytes of data:

Reply from 12.12.12.4: bytes=32 time<1ms TTL=128
Reply from 12.12.12.4: bytes=32 time=1ms TTL=128
Reply from 12.12.12.4: bytes=32 time=1ms TTL=128
Reply from 12.12.12.4: bytes=32 time=1ms TTL=128

Ping statistics for 12.12.12.4:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
```

Top


PC8
Physical
Config
Desktop
Programming
Attributes
Command Prompt
X
Cisco Packet Tracer PC Command Line 1.0
C:\>ping 100.100.100.3
Pinging 100.100.100.3 with 32 bytes of data:
Reply from 100.100.100.3: bytes=32 time<1ms TTL=128
Reply from 100.100.100.3: bytes=32 time<1ms TTL=128
Reply from 100.100.100.3: bytes=32 time<1ms TTL=128
Reply from 100.100.100.3: bytes=32 time<1ms TTL=128
Ping statistics for 100.100.100.3:
Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
Minimum = 0ms, Maximum = 0ms, Average = 0ms
C:\>ping 15.15.15.4
Pinging 15.15.15.4 with 32 bytes of data:
Reply from 100.100.100.10: Destination host unreachable.
Reply from 100.100.100.10: Destination host unreachable.
Reply from 100.100.100.10: Destination host unreachable.
Reply from 100.100.100.10: Destination host unreachable.
Ping statistics for 15.15.15.4:
Packets: Sent = 4, Received = 0, Lost = 4 (100% loss),
C:\>
Top

Shirish
C3 C32
2103164

EXPERIMENT :- 7

CN-C32-2103164

Aim :- WAP to implement IPv4 addressing concept along with subnet masking

Theory :-

IPV4 addressing is a fundamental concept in networking in that involves assigning unique numerical label to devices on a network. These addresses are used for identification and communication between devices in the IP network. IPv4 addresses are represented as a series of four decimal numbers, each ranging from 0 to 255, seperated by periods. Each numbers is an octet representing 8 bits of 32 bit IPv4 address.

Here how IPv4 addressing works :-

32-bit Address :- IPv4 uses a 32-bit address space, allowing for approximately, 4.3 billon unique addresses.

Network and Host Portion :- An IPv4 addres is divided into two parts : the network portion and the host portion. The boundary between two pointers is determined by a subnet mask.

Subnet mask :- The subnet mask is 32 bit numb
that defines the boundary between the ne
and host portions of an IPV4 address. It
consists of series of contigous 1's followed by
of contigous 0s. The 1s indicate the network
bits, and 0s indicate the host bits.

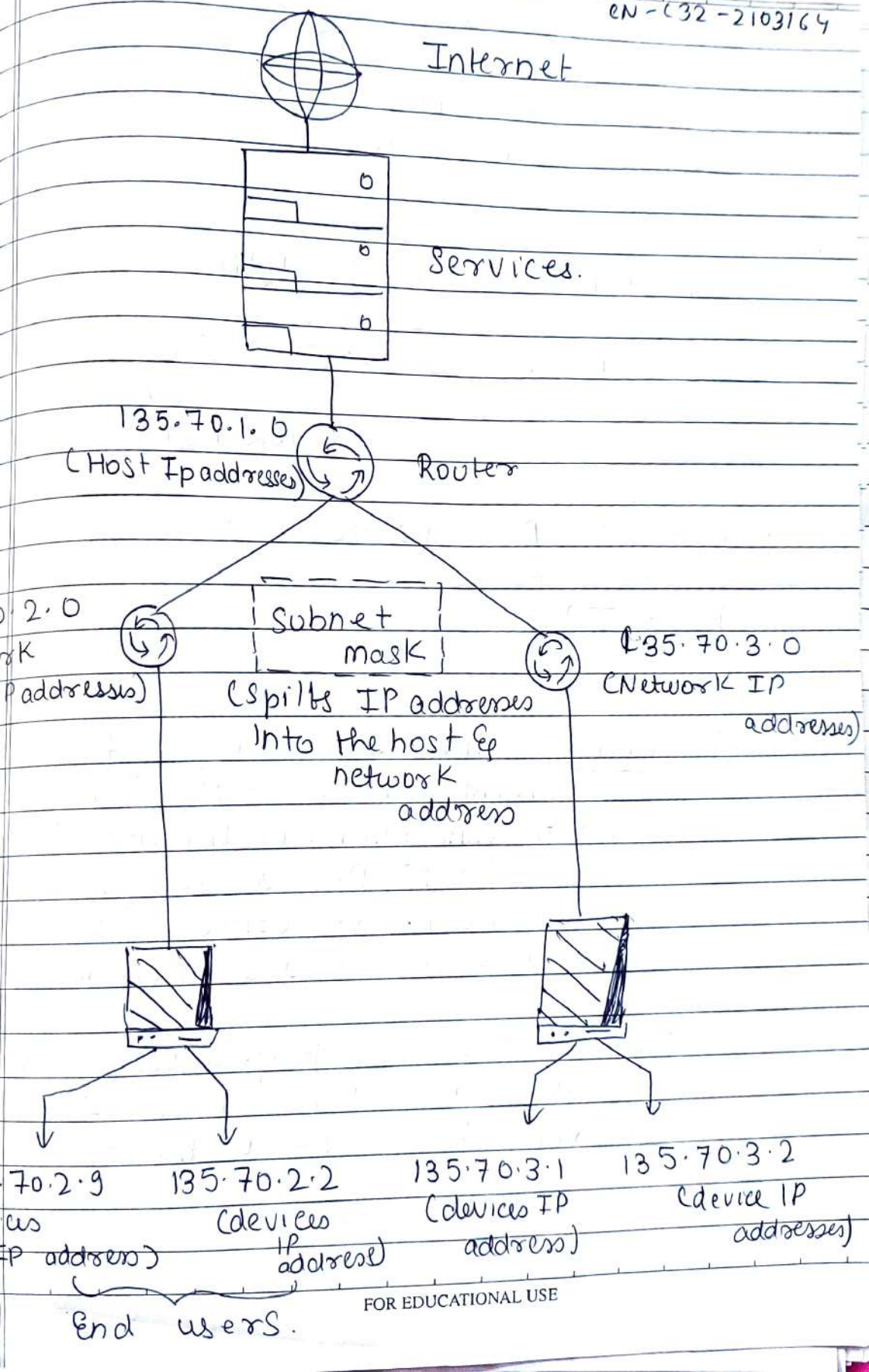
Network ID and Host ID :- The network portion
of address identifies the network to which
devices belongs, while the host portion id
specific devices within that network. Devic
within the same network share share the
same network ID.

Classes of IP addresses :- In past, IPV4 ad
were divided into five classes (A, B, C, D a
based on their first octet range. However
classful addressing has largely been rep
by classless InterDomain Routing (CIDR) which
allows for more flexible allocation of IP
addresses space.

Private and Public Addresses :- IPV4 addresses a
categorized into private and public addre
ranges. Private addresses are reserved f
use within private network and are not
routable on the public internet. Public ad
are globally unique and are used for devices
accessible from the internet.

Subnet Masking

EN-C32-2103164



Subnet Masking

* Example of IPV4 addresses with the associated subnet mask

- IPV4 Address : 192.168.1.10
- Subnet Mask : 255.255.255.0 (also represented as 124) (CIDR notation.)
- Network ID : 192.168.1.0
- Host ID : 10.

* Conclusion :- IPV4 addressing is fundamental concept in networking & is essential for the functioning of internet & local networks. Proper addresses allocation & subnetting are critical for efficient network, design and management.

This experiment helped us to shrunk & built it's logic which further enchanced the clarity of this concept

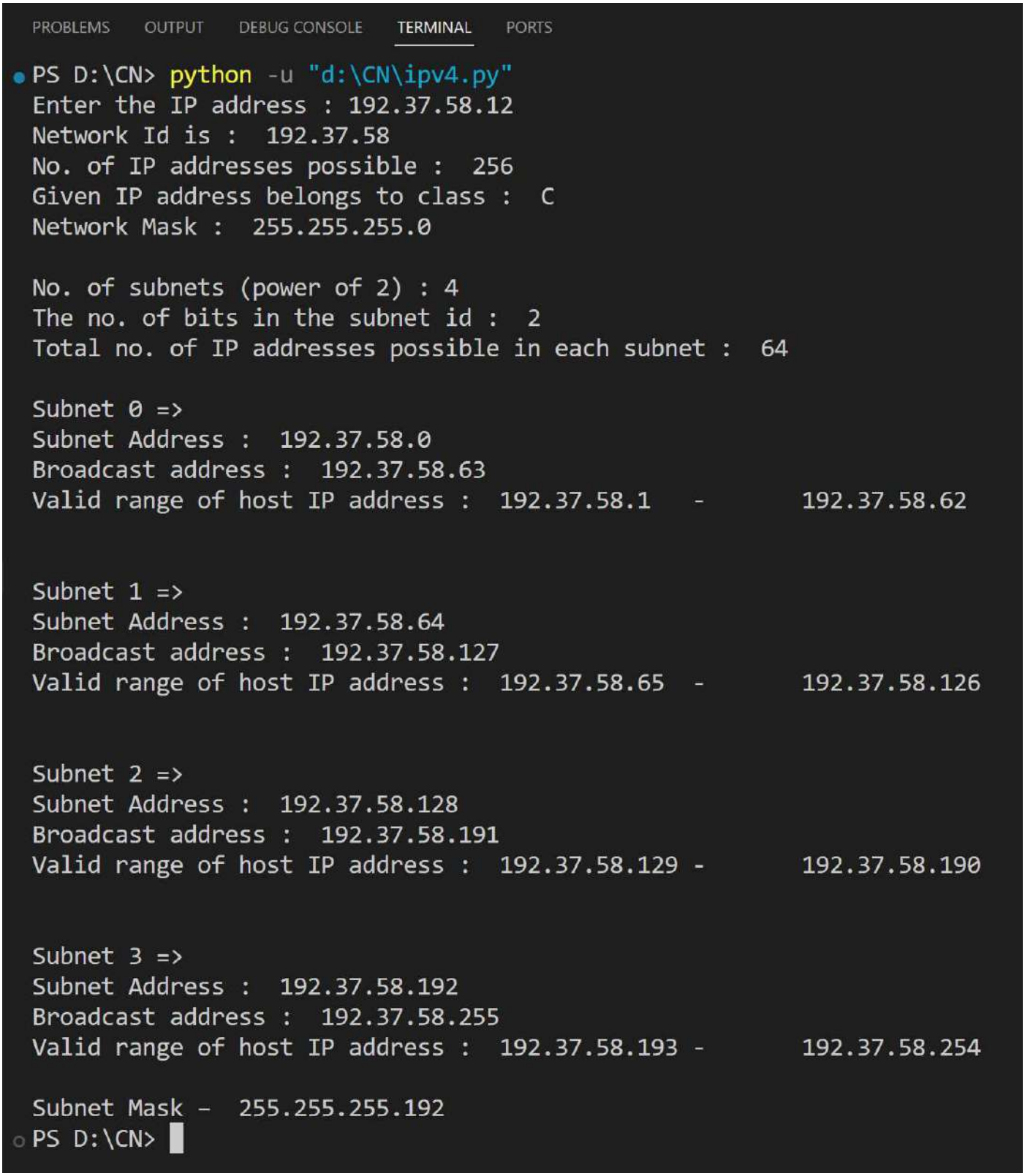
03/10/23

```
import math


def findClass(ip):
    if 0 <= ip[0] <= 127:
        print("Network Address is : ", ip[0])
        print('No. of IP addresses possible : ', 2 ** 24)
        return "A", '255.0.0.0'
    elif 128 <= ip[0] <= 191:
        ip = [str(i) for i in ip]
        print("Network Address is : ", ".".join(ip[0:2]))
        print('No. of IP addresses possible : ', 2 ** 16)
        return "B", '255.255.0.0'
    elif 192 <= ip[0] <= 223:
        ip = [str(i) for i in ip]
        print("Network Id is : ", ".".join(ip[0:3]))
        print('No. of IP addresses possible : ', 2 ** 8)
        return "C", '255.255.255.0'
    elif 224 <= ip[0] <= 239:
        print("In this Class, IP address is not divided into Network and Host ID")
        return "D"
    else:
        print("In this Class, IP address is not divided into Network and Host ID")
        return "E"


def Subnetting(ip, num, className, ip_addresses):
    temp = 0
    if className == "A":
        place2 = ip_addresses / (256 ** 2)
        for i in range(num):
```

```
            print(f"Subnet {i} => ")
            print(temp)
            print("Subnet Address : ", ip[0] + '.' + str(temp) + '.0' + '.0')
            temp += int(place2)
            print("Broadcast address : ", ip[0] + '.' + str(temp - 1) + '.255' + '.255')
            print("Valid range of host IP address : ", ip[0] + '.' + str(temp - int(place2)) + '.' + '0' +
'.1' + '\t-\t' + ip[0] + '.' + str(temp - 1) + '.254' + '.254')
            print()
    elif className == "B":
        place2 = ip_addresses / 256
        for i in range(num):
            print(f"\nSubnet {i} => ")
            print("Subnet Address : ", ".".join(ip[0:2]) + '.' + str(temp) + '.0')
            temp += int(place2)
            print("Broadcast address : ", ".".join(ip[0:2]) + '.' + str(temp - 1) + '.255')
            print("Valid range of host IP address : ",".".join(ip[0:2]) + '.' + str(temp - int(place2)) +
'.1\t-\t' + ".".join(ip[0:2]) + '.' + str(temp - 1) + '.254')
            print()
    elif className == "C":
        for i in range(num):
            print(f"\nSubnet {i} => ")
            print("Subnet Address : ", ".".join(ip[0:3]) + '.' + str(temp))
            temp += int(ip_addresses)
            print("Broadcast address : ", ".".join(ip[0:3]) + '.' + str(temp - 1))
            print("Valid range of host IP address : ",".".join(ip[0:3]) + '.' + str(temp -
int(ip_addresses) + 1) + '\t-\t' + ".".join(ip[0:3]) + '.' + str(temp - 2))
            print()
    else:
        print("In this Class, IP address is not divided into Network and Host ID")
```

```
def subnetmask(num, network_mask):
    var = '1' * int(math.log(num, 2))
    var1 = '0' * (8 - int(math.log(num, 2)))
    binary_num = var + var1
    network_mask = network_mask.split('.')
    network_mask = [i for i in network_mask if i != '0']
    network_mask.append (str(int(binary_num, 2)))
    while len(network_mask) < 5:
        network_mask.append('0')
    print('Subnet Mask – ', ".".join(network_mask[0:4]))

ip = input("Enter the IP address : ")
ip = ip.split(".")
ip = [int(i) for i in ip]
lst = findClass(ip)
networkClass = lst[0]
print("Given IP address belongs to class : ", networkClass)
ip = [str(i) for i in ip]
network_mask = lst[1]
print('Network Mask : ', network_mask)
num_subnet = int(input('\nNo. of subnets (power of 2) : '))
num_ip = int(2 ** (8 * (68 - ord(networkClass))) / num_subnet)
print('The no. of bits in the subnet id : ', int(math.log(num_subnet, 2)))
if ord(networkClass) < 68:
    print('Total no. of IP addresses possible in each subnet : ', num_ip)
Subnetting(ip, num_subnet, networkClass, num_ip)
subnetmask(num_subnet, network_mask)
```

OUTPUT:


PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS
PS D:\CN> python -u "d:\CN\ipv4.py"
Enter the IP address : 192.37.58.12
Network Id is : 192.37.58
No. of IP addresses possible : 256
Given IP address belongs to class : C
Network Mask : 255.255.255.0
No. of subnets (power of 2) : 4
The no. of bits in the subnet id : 2
Total no. of IP addresses possible in each subnet : 64
Subnet 0 =>
Subnet Address : 192.37.58.0
Broadcast address : 192.37.58.63
Valid range of host IP address : 192.37.58.1 - 192.37.58.62
Subnet 1 =>
Subnet Address : 192.37.58.64
Broadcast address : 192.37.58.127
Valid range of host IP address : 192.37.58.65 - 192.37.58.126
Subnet 2 =>
Subnet Address : 192.37.58.128
Broadcast address : 192.37.58.191
Valid range of host IP address : 192.37.58.129 - 192.37.58.190
Subnet 3 =>
Subnet Address : 192.37.58.192
Broadcast address : 192.37.58.255
Valid range of host IP address : 192.37.58.193 - 192.37.58.254
Subnet Mask - 255.255.255.192
PS D:\CN>

THADOMAL SHAHANI TSEC ENGINEERING COLLEGE

EXPERIMENT NO-8

CN-C32-2103164

Aim:- Use basic networking commands in Linux (ping, traceroute, nslookup, netstat, ARP, RARP, ipconfig, dig, route).

Theory:-

1) Ping:-

The 'ping' command is used to test network connectivity between two devices. It send ICMP Echo Request message to target host and wait for ICMP Echo Reply.

2) traceroute or tracert:-

'traceroute' is used to trace route taken by packet to destination. It shows the path packet follow through various routers and network to reach the target host.

Example:- 'traceroute google.com'

3) 'nslookup':-

'nslookup' is tool for quering DNS server to resolve domain name to corresponding IP address. It also provide information about DNS servers and domain record

Example:- 'ns lookup google.com'.

4.) 'netstat' :- 'netstat' is command for displaying network statics, it can show information about actuv network connection, listening port, routing tables and various network interfaces.
Example :- 'netstat -tuln' to display listening TCP/UDP ports.

5.) 'arp' :- The 'arp' command allows you to view and manipulate the Address Resolution protocol (ARP) cache. ARP maps IP address to MA addresses on local network.
Example "arp -a" to list ARP cache entries.

6.) "rarp" : Reverse Address Resolution is rarely used today and has been replaced by other method for IP address assignments.

7.) 'ip' :-

The 'ip' command is versatile tool for configuring and managing network, interfaces, routing tables and other networking parameters. You can use to assign IP address, set up routes, and more.
Example :- 'ip addres show' to display.

THADOMAL SHAHANI TSEC ENGINEERING COLLEGE

CN-C32-2103164

information about network interfaces.

'ifconfig' :- 'ifconfig' is used for configuring and displaying network interface while it is still present in many LINUX distribution, it is considered somewhat deprecated and replaced by the 'ip' command.
Eg :- 'ifconfig eth0' to show information about the "eth0" interface.

'dig' :-

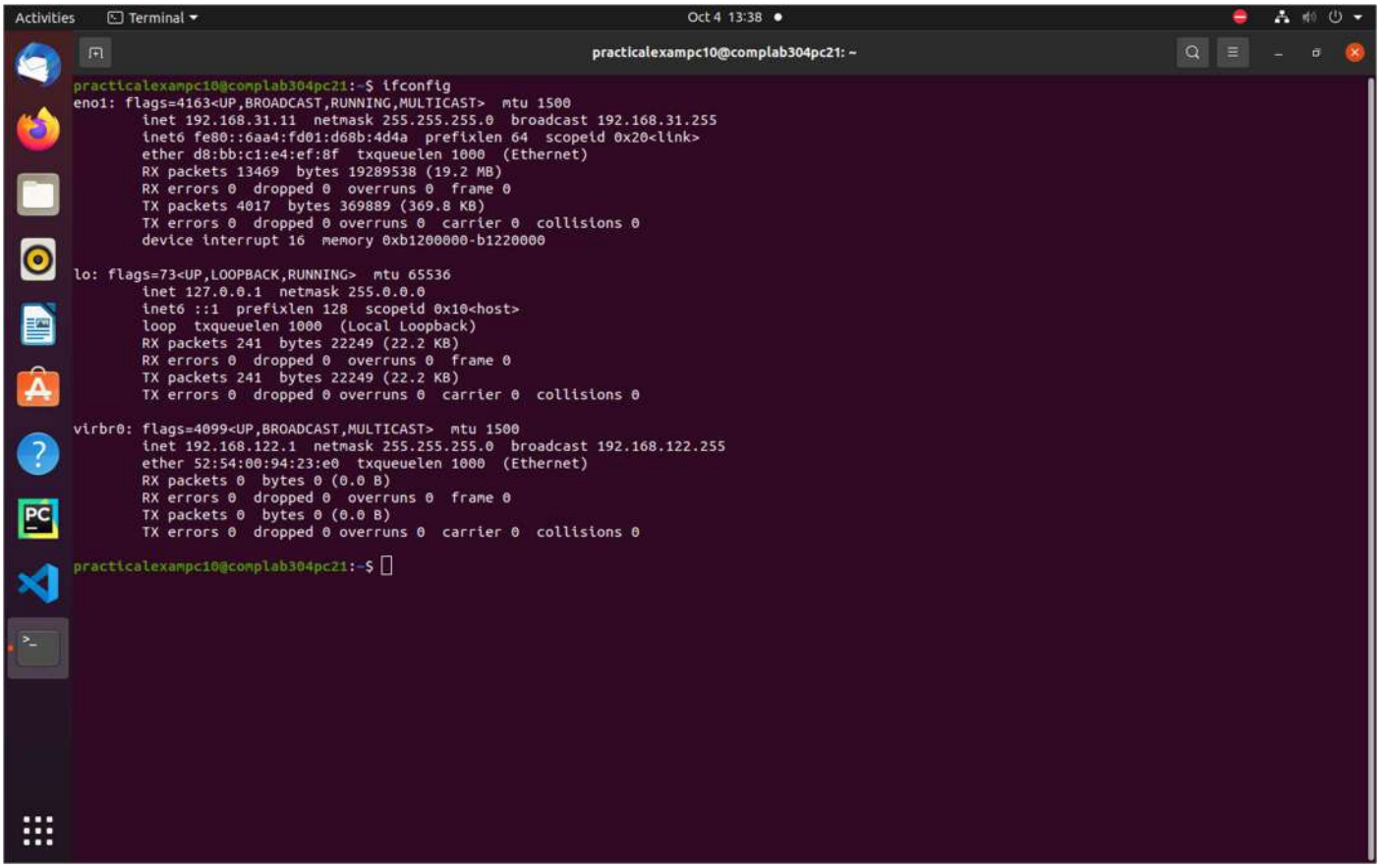
dig is powerful DNS query tool that allows you to retrieve detailed information from DNS servers. You can use it query DNS record, find authoritative name servers and more.
Example 'dig google.com'

'route'

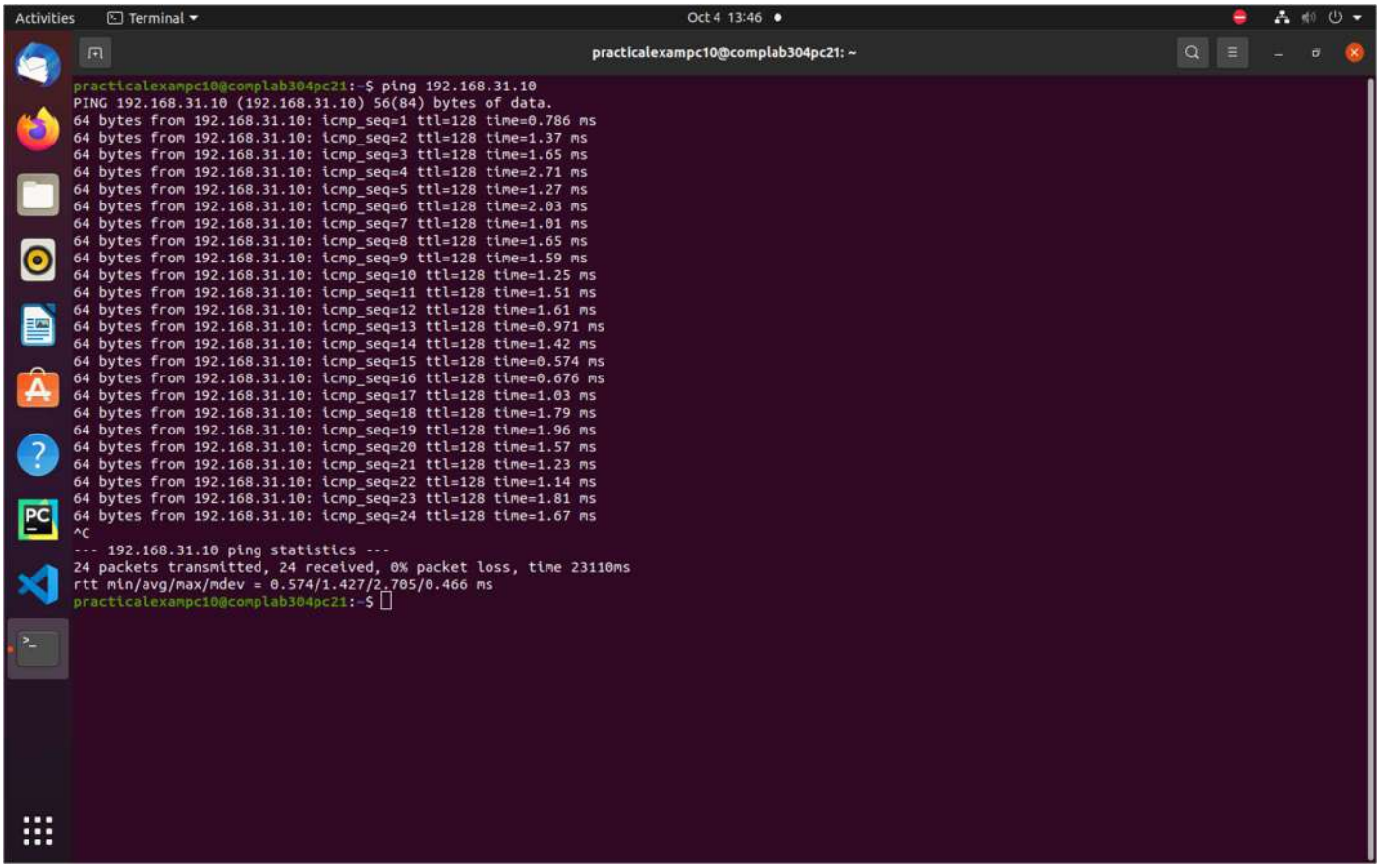
The route command help you view or manipulate the IP routing table. You can use it to add or remove routes, which determine how data packet are forwarded through a network.
Example :- 'route -n' to display the routing table in numeric format.

Conclusion :- These command are essential for Network diagnostics and configuration on LINUX system, helping administrators & users troubleshoot & manage network related issues.

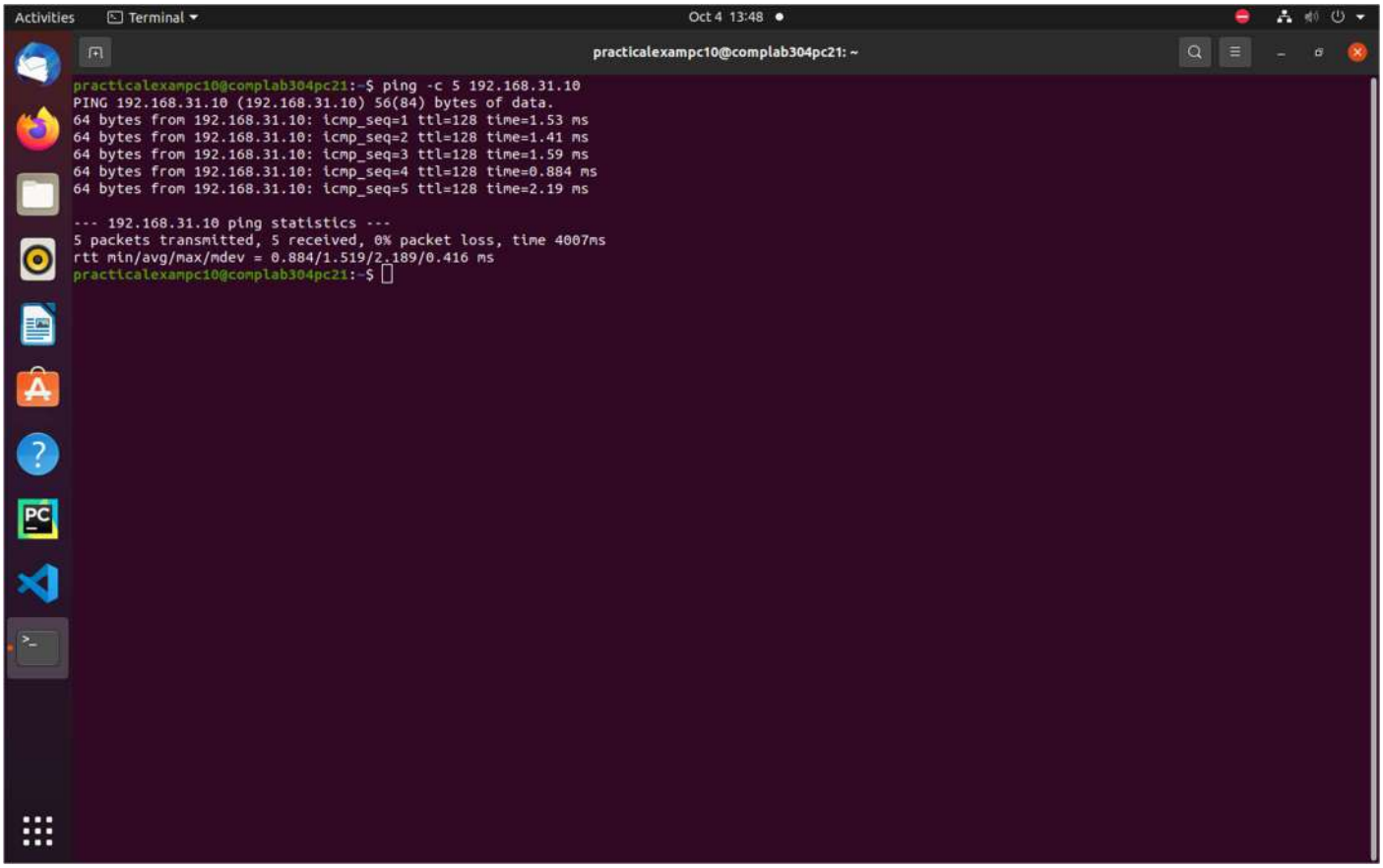
1. ifconfig



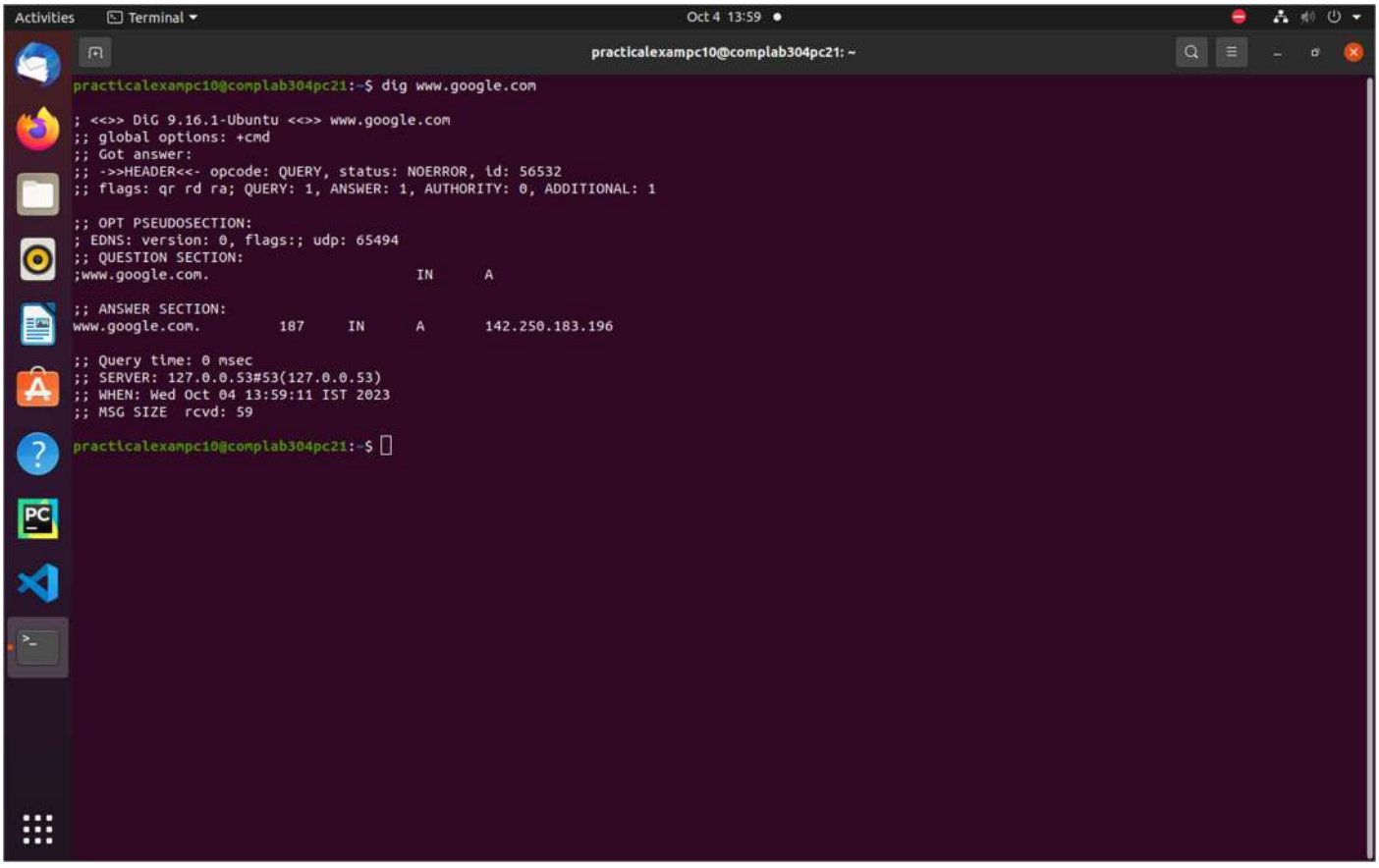
2. ping without count



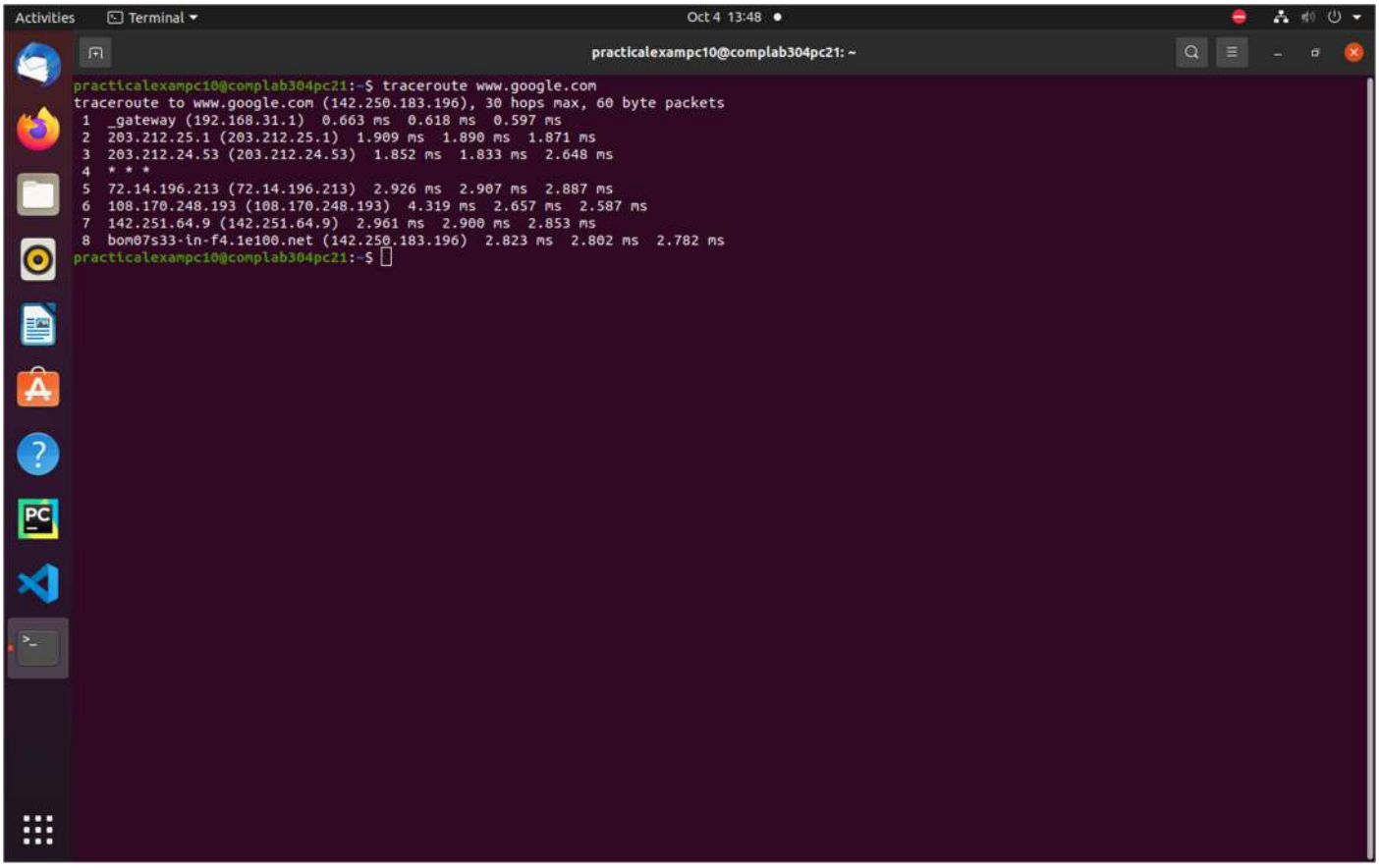
3. ping with count



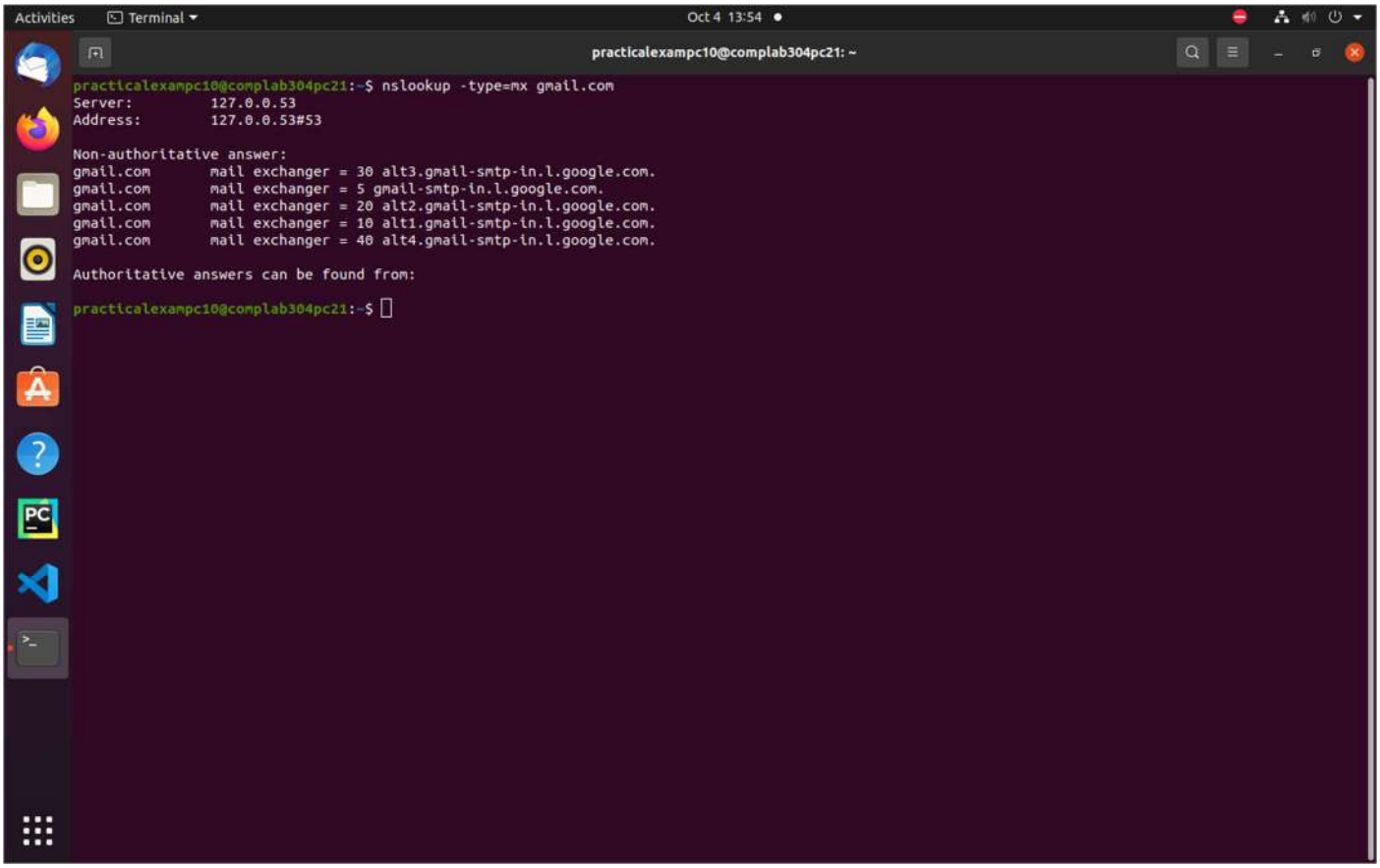
4. dig



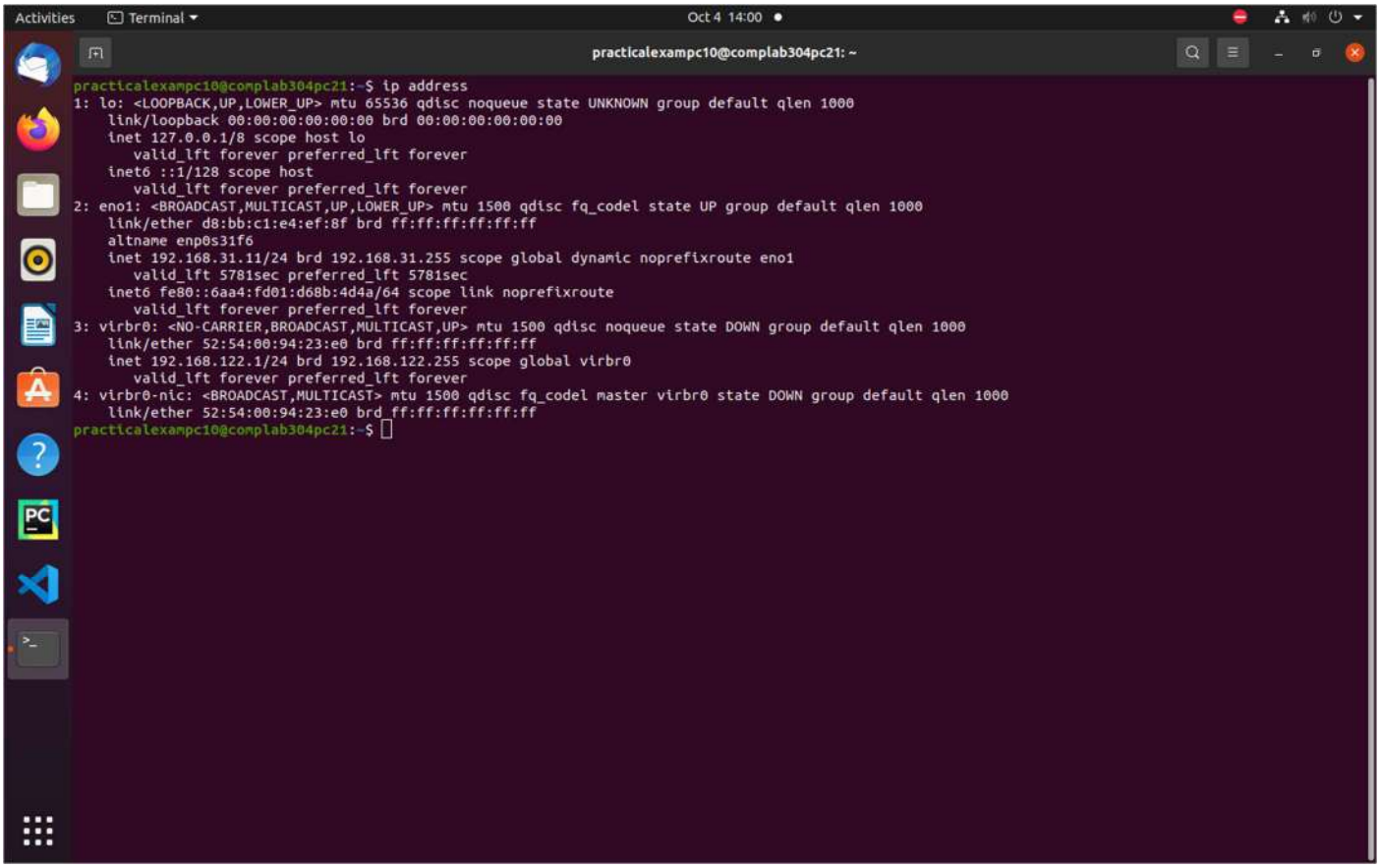
5. traceroute



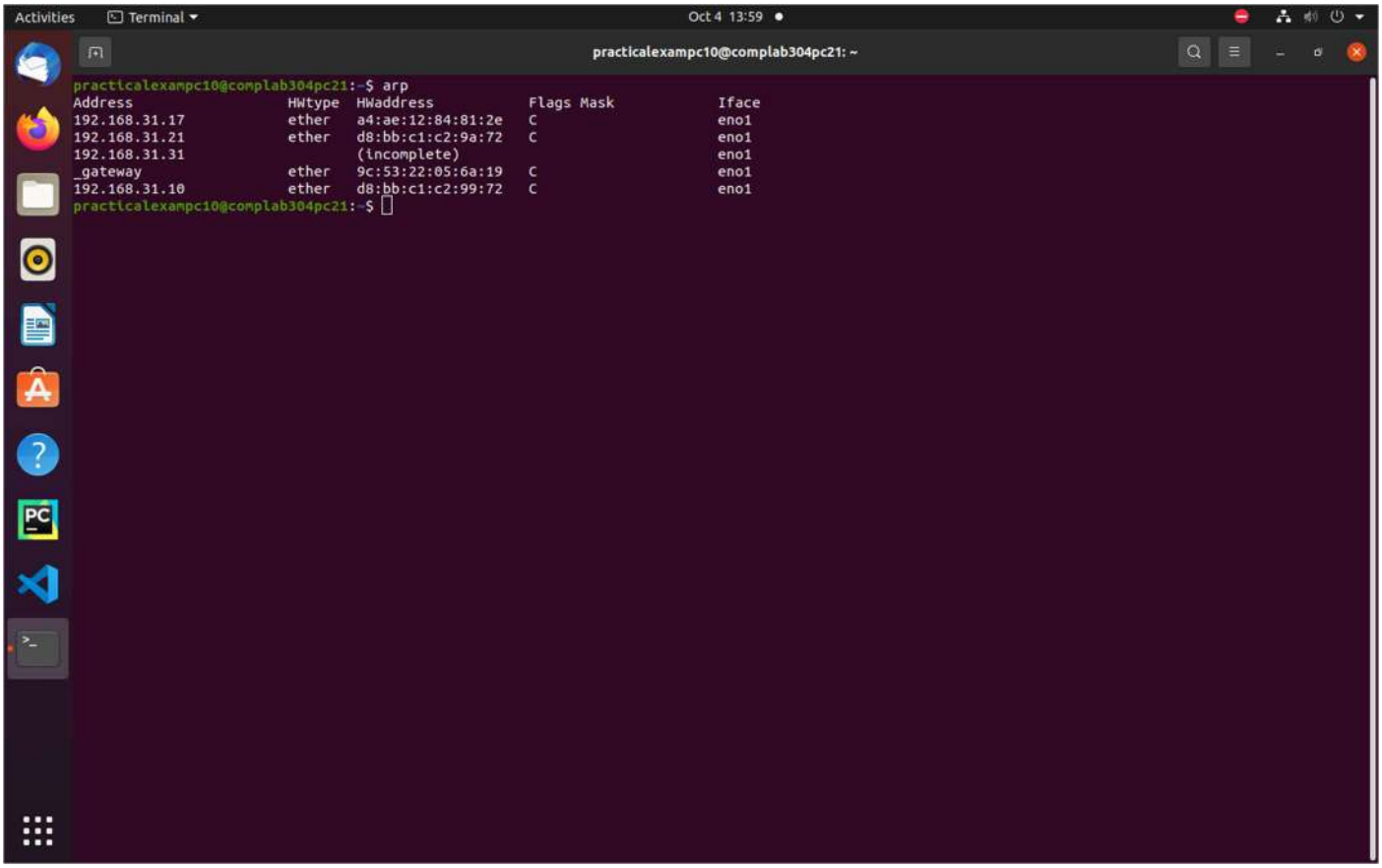
6. nslookup



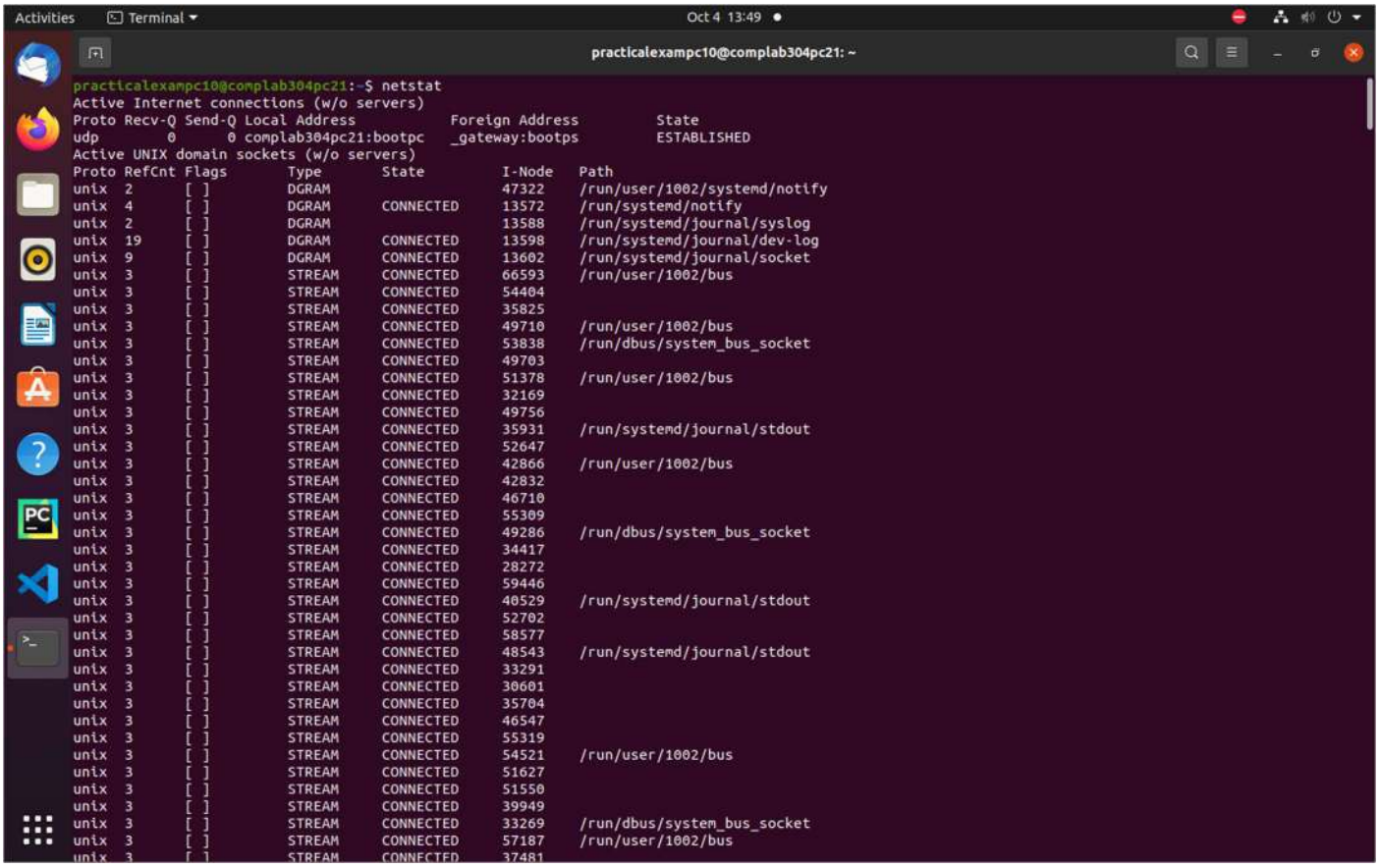
7. ip address



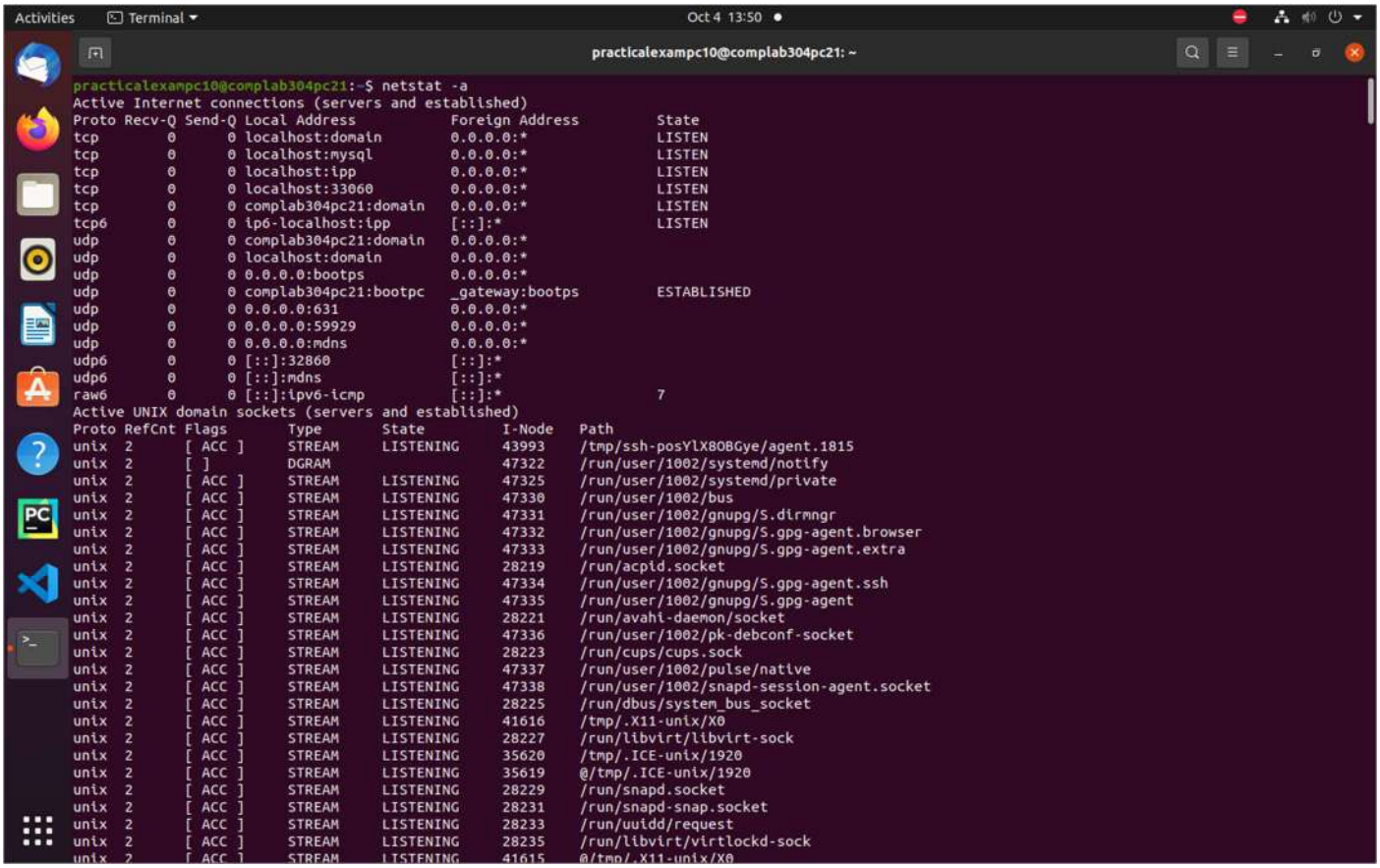
8. arp



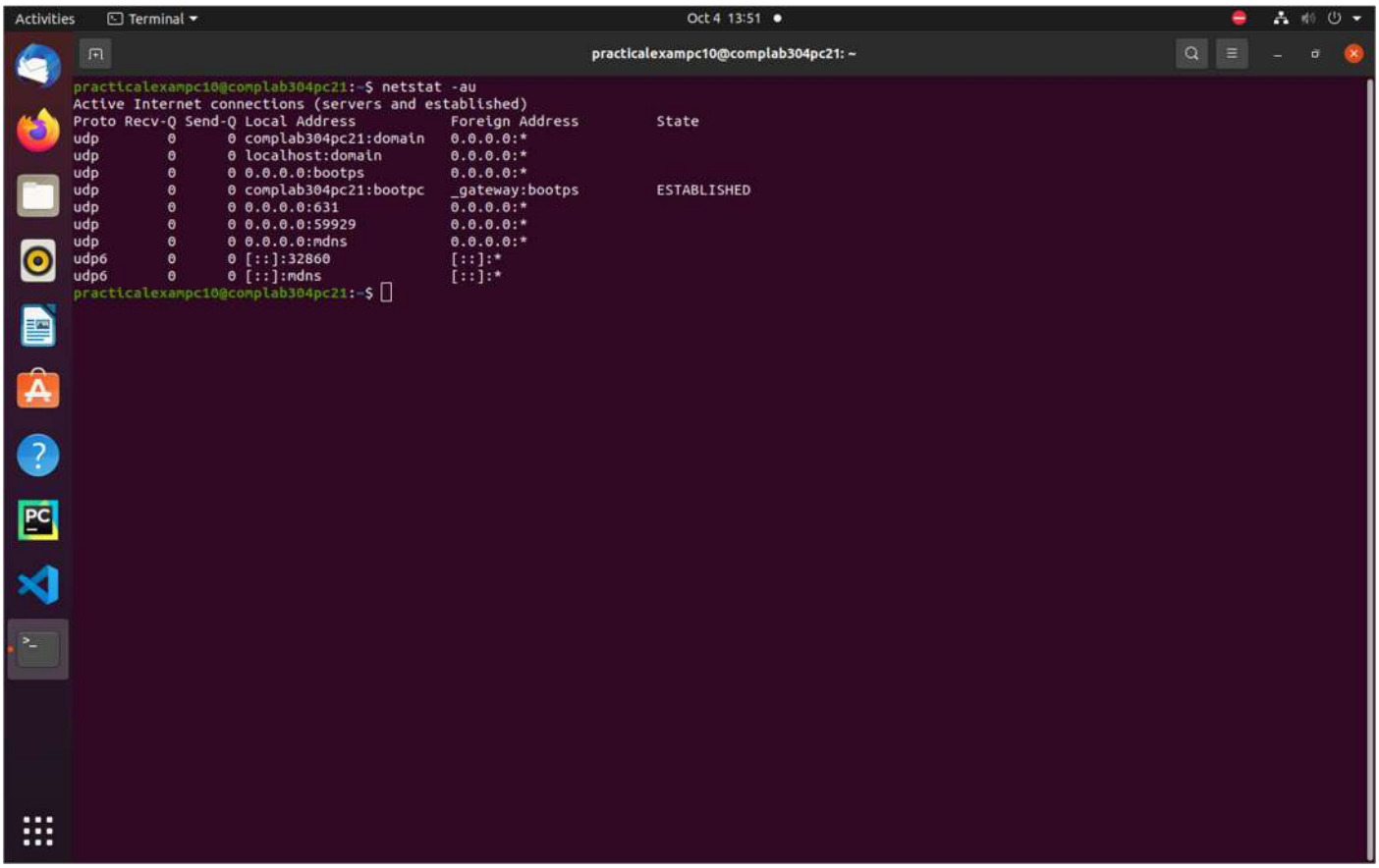
9. netstat



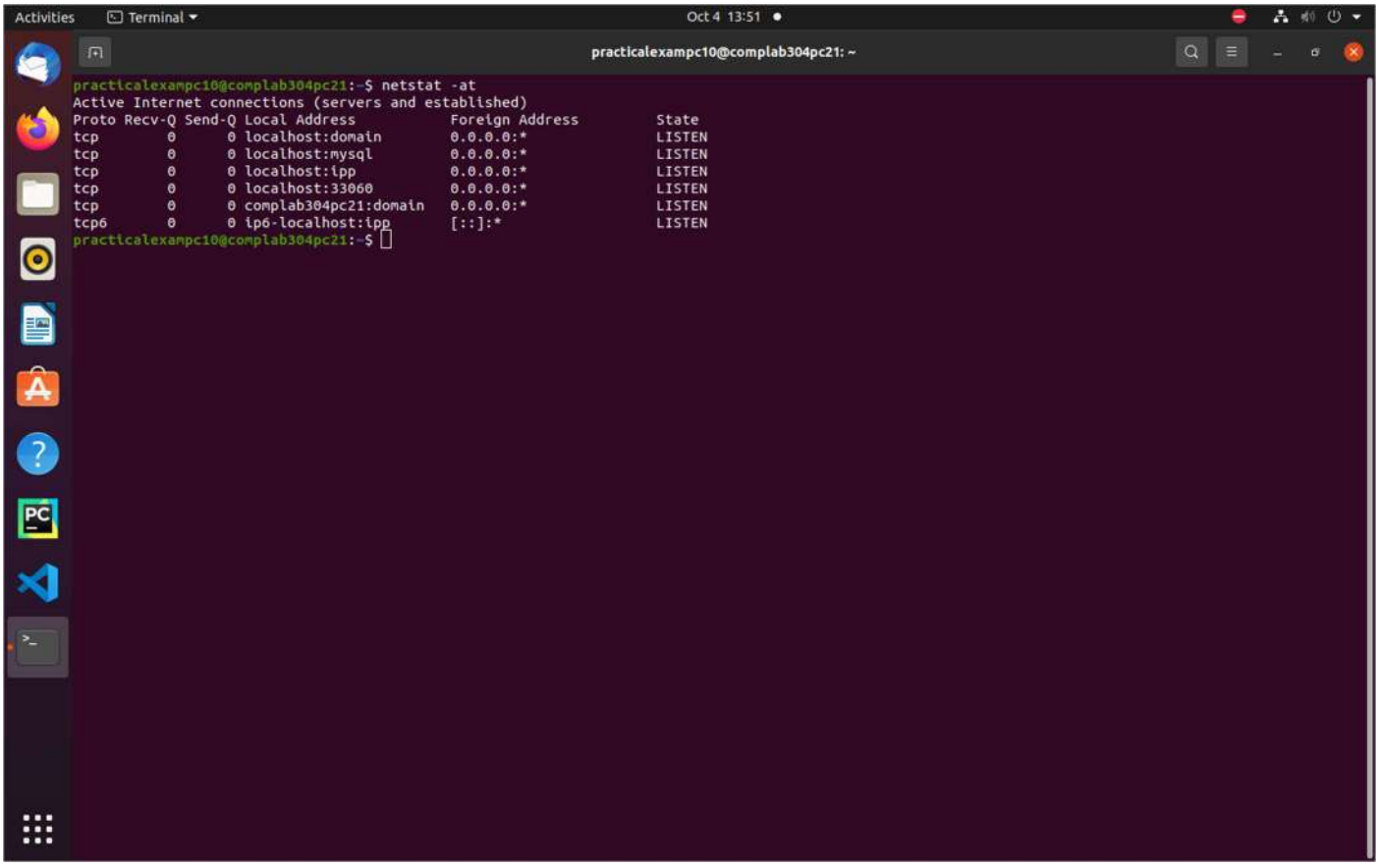
10. netstat -a



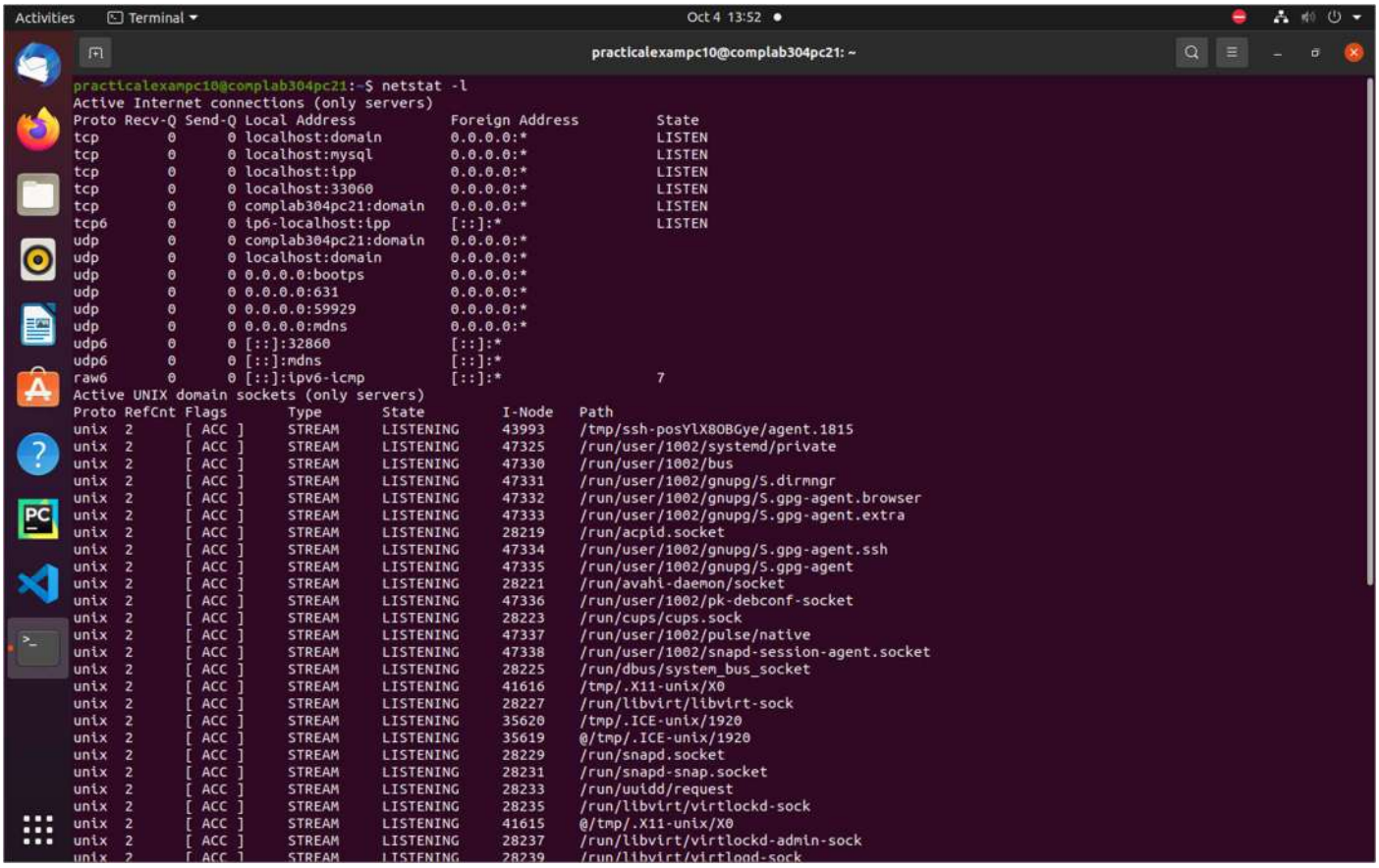
11. netstat -au



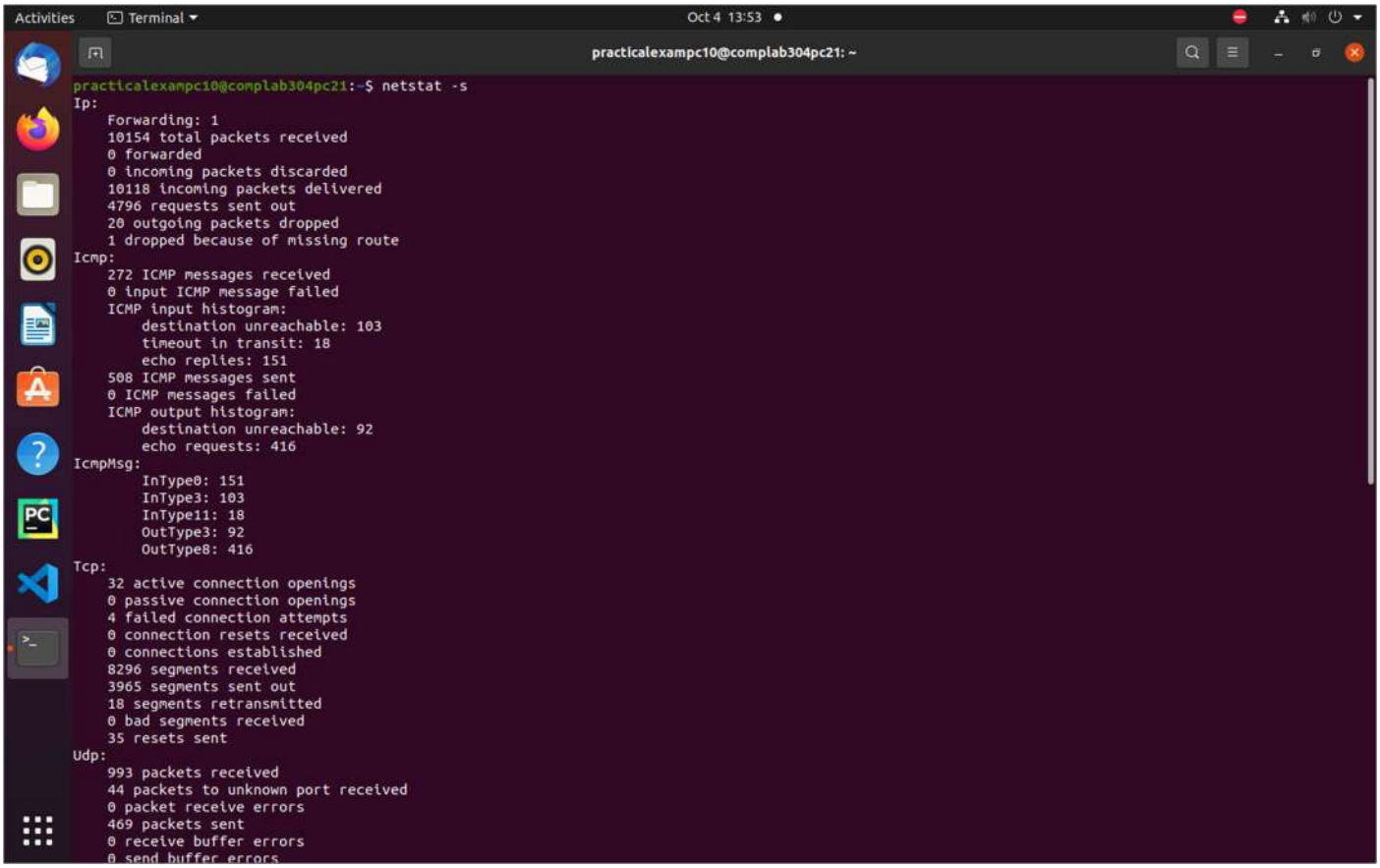
12. netstat -at



13. netstat -l



14. netstat -s



THADOMAL SHAHANI TSEC ENGINEERING COLLEGE

EXPERIMENT NO-9

CN-C32-2103164

Aim:-

Use wireshark to understand the operation of TCP/IP layers.

- Ethernet layers: Frame header/Frame sized.
- Data link Layer:- MAC address; ARP. (IP & MAC addressing)
- Network Layer: IP Packet.
- Transport layer: TCP Paset, TCP, handshake segments etc.

Application Layer:- DHCP, FTP, HTP. header format.

Theory:-

Wireshark is a popular, open source network protocol analysis used for capturing and inspecting packets on a network. It is a powerful tool for network administrators, security professionals and students.

The objective of this experiment is to use wireshark, a network protocol analysis, to capture and analyze network traffic to understand the operation of various layer of TCP/IP protocol suite.

CN-L82-2103164

1.) Ethernet layer.

This layer deals with frames, which are packets of data at the ethernet layer packets of data. one can observe the size of ethernet frames. one can observe the size of ethernet frames. One can observe the size of ethernet frames. Ethernet frames have headers containing information. like source & destination MAC address.

2.) Data link layer

The data link layer I deal with MAC addresses. Wireshark will you the MAC addresses of the addresses of the devices communicating in network. Users can capture ARP to packet to see how devices map, IP addresses to MAC addresses.

3.) Network Layer.

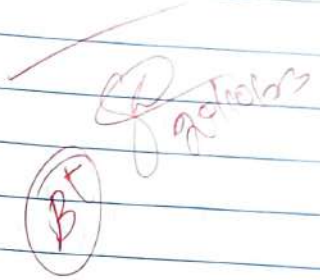
This layer involves a lot of process. It involves IP packets. Wireshark displays IP headers with information like source & destination IP address. ~~Source~~ ICMP packet can be captured to observe network troubleshooting, messages, like ping request & replies.

THADOMAL SHAHANI TSEC ENGINEERING COLLEGE

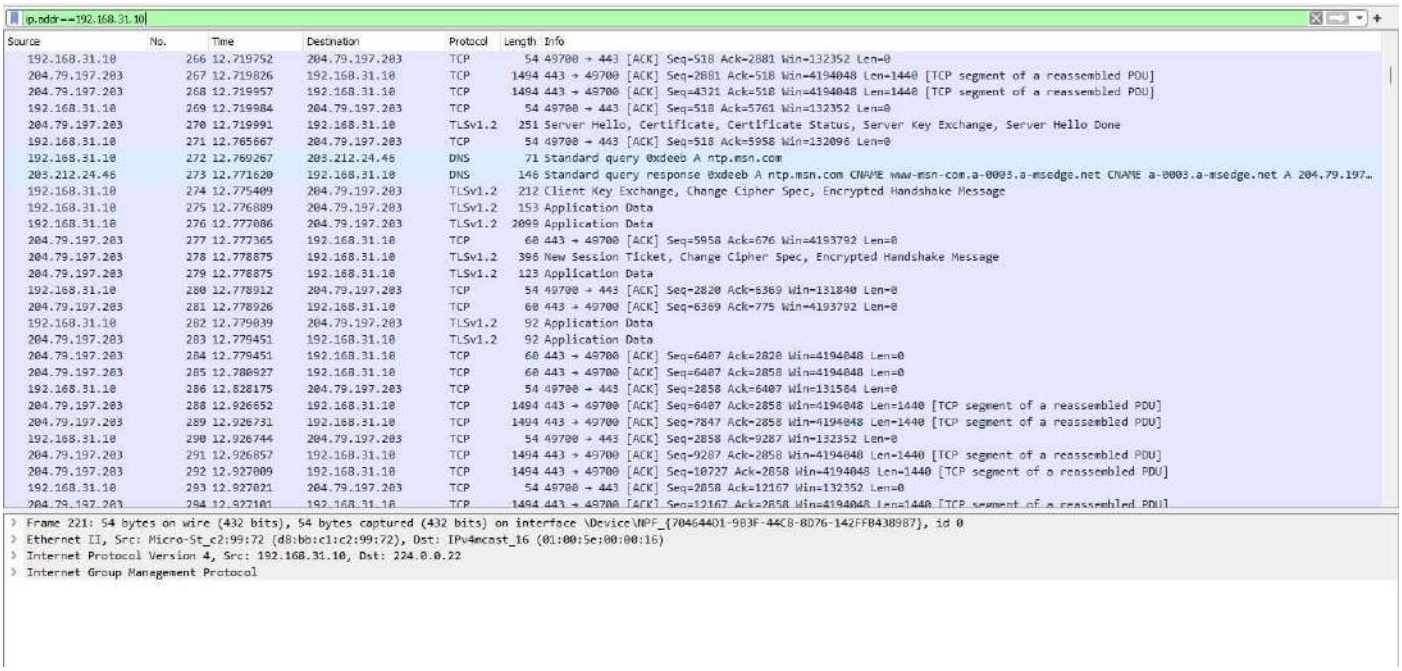
CN - C32 - 2103164

Transport layers :-

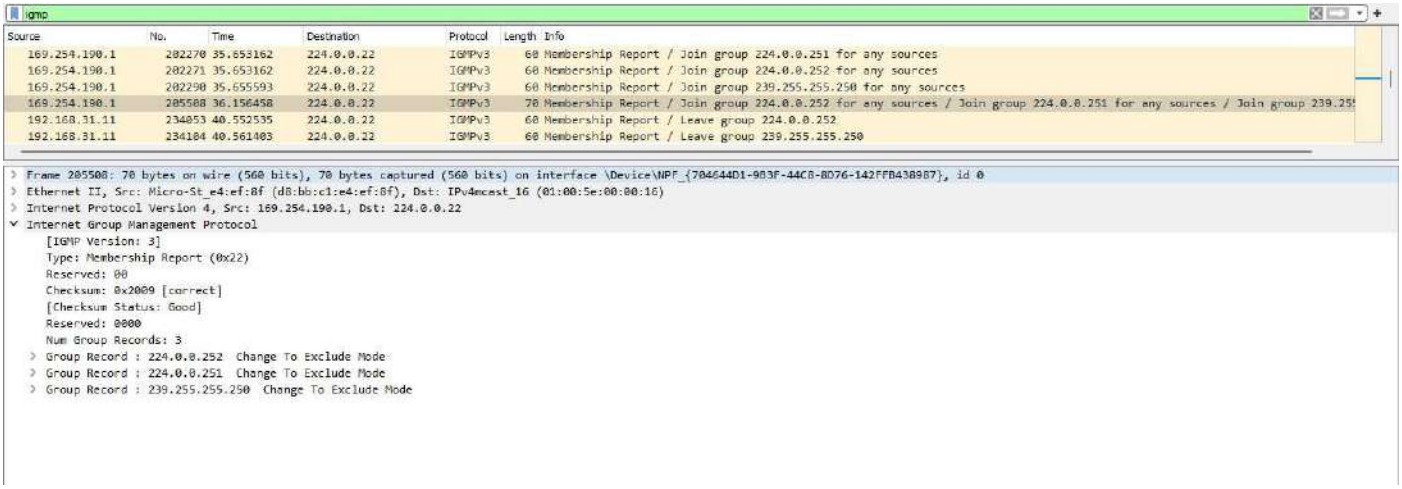
User can see which ports the used by application (eg:- web browsers, email clients) for communication. By capturing TCP traffic, you can see three-way handshake process, which is how two devices establish a connect.



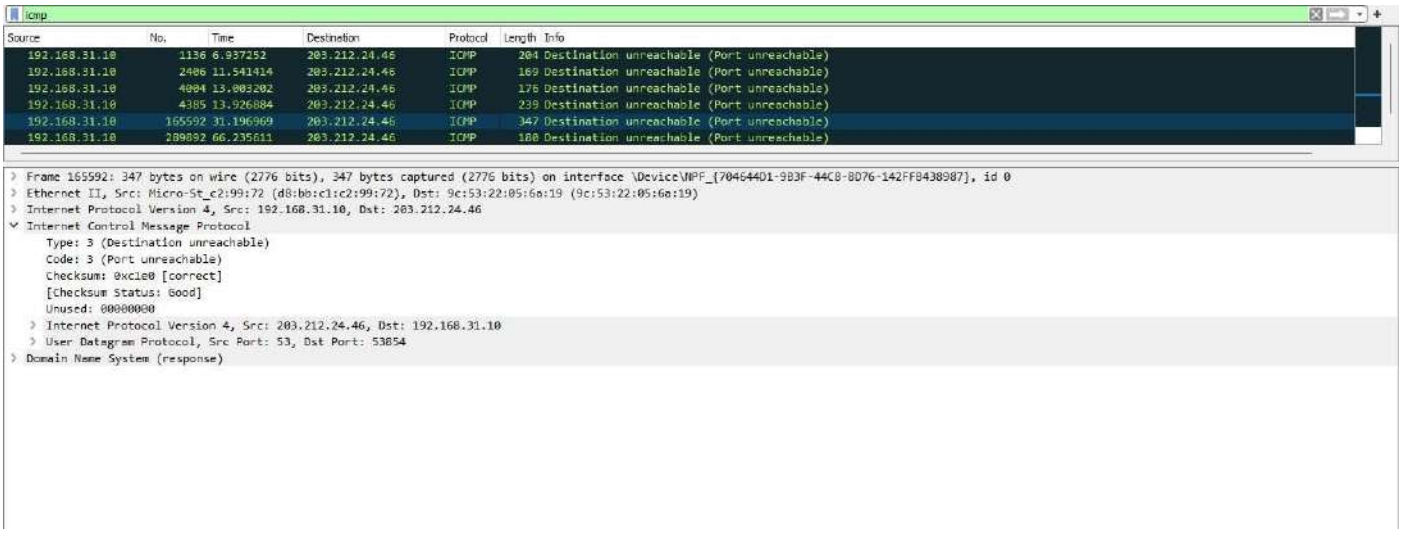
1. Internet Protocol (IP)



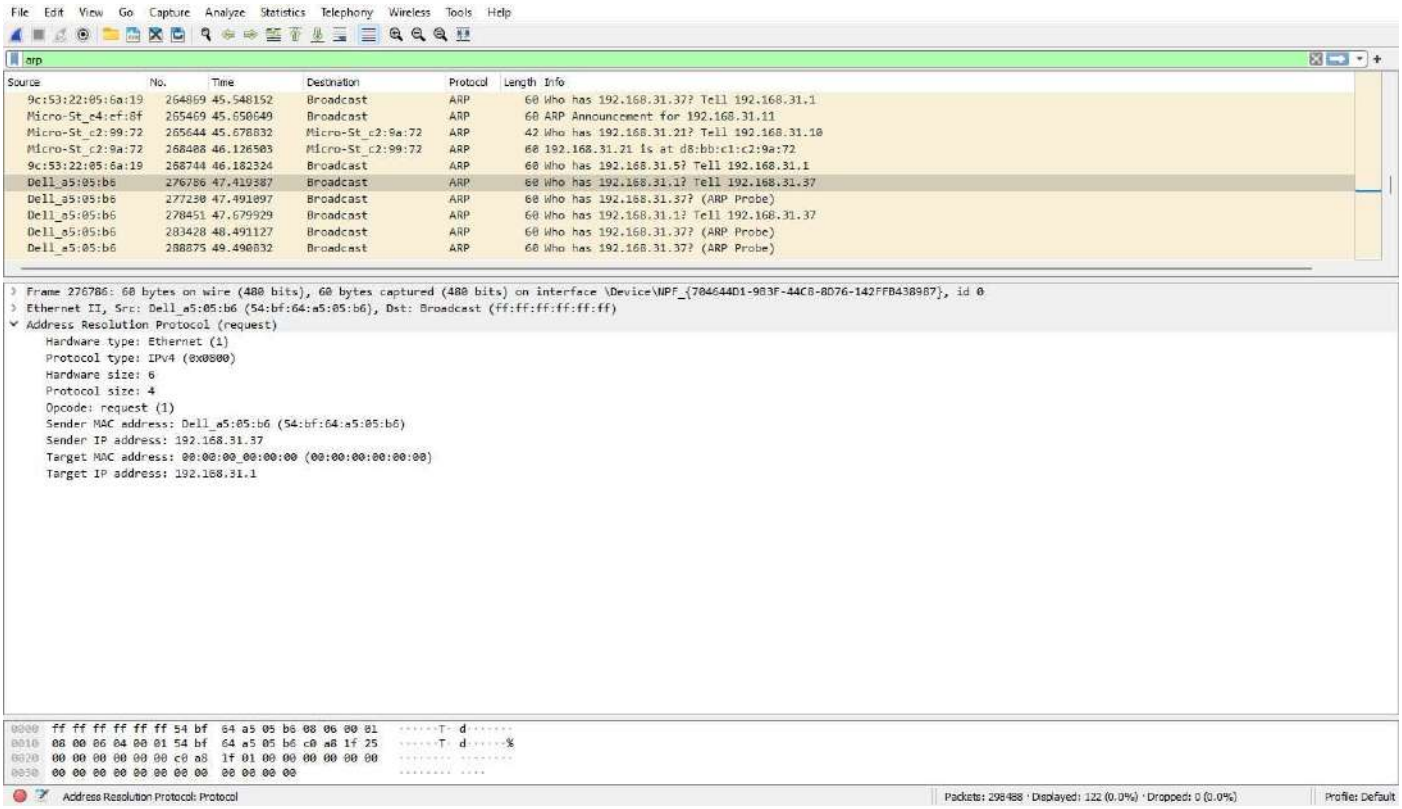
2. Internet Group Management Protocol (IGMP)



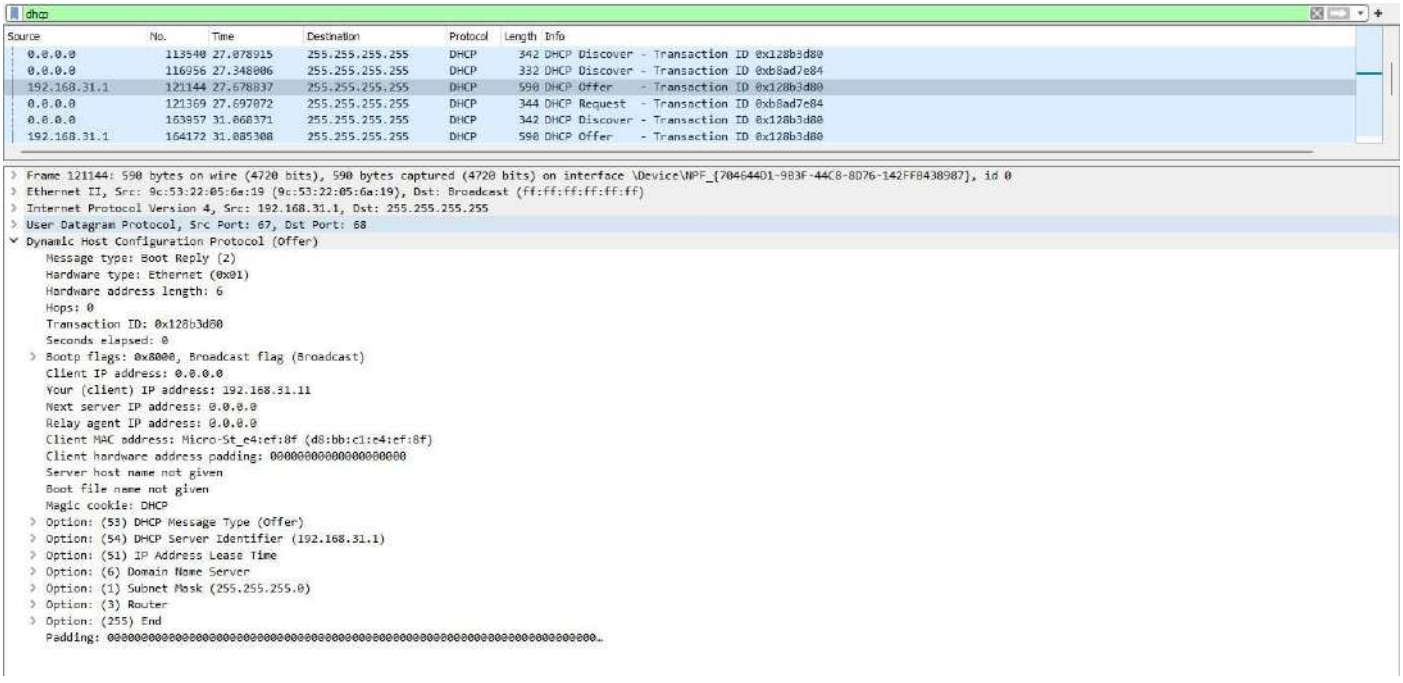
3. Internet Control Message Protocol (ICMP)



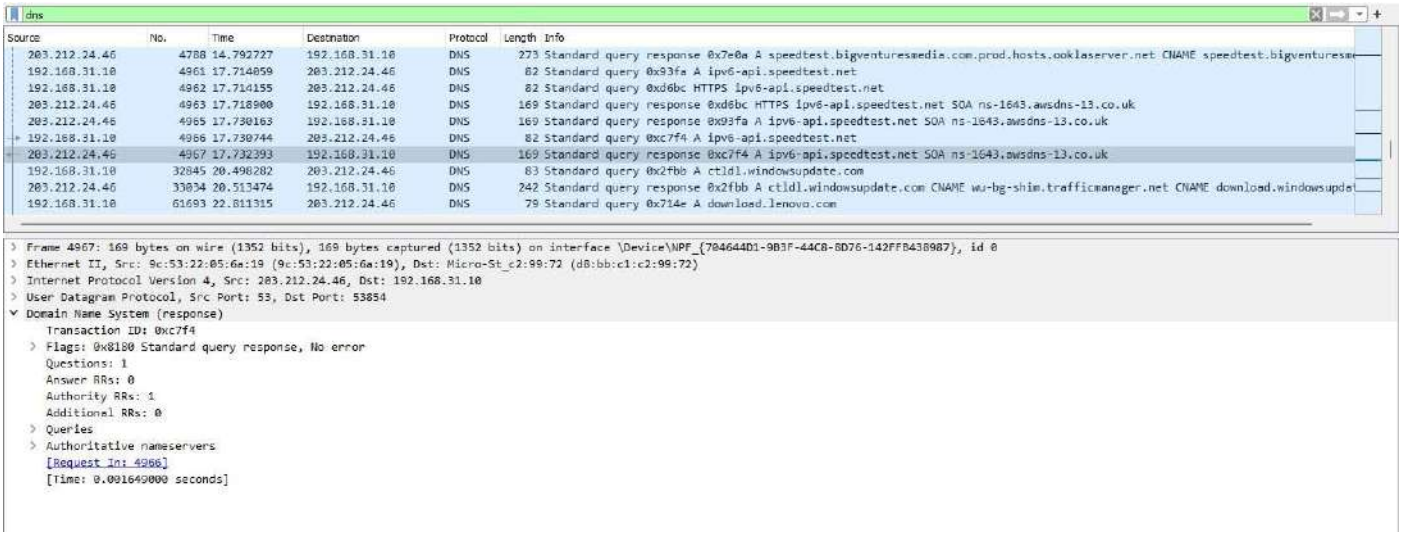
4. Address Resolution Protocol (ARP)



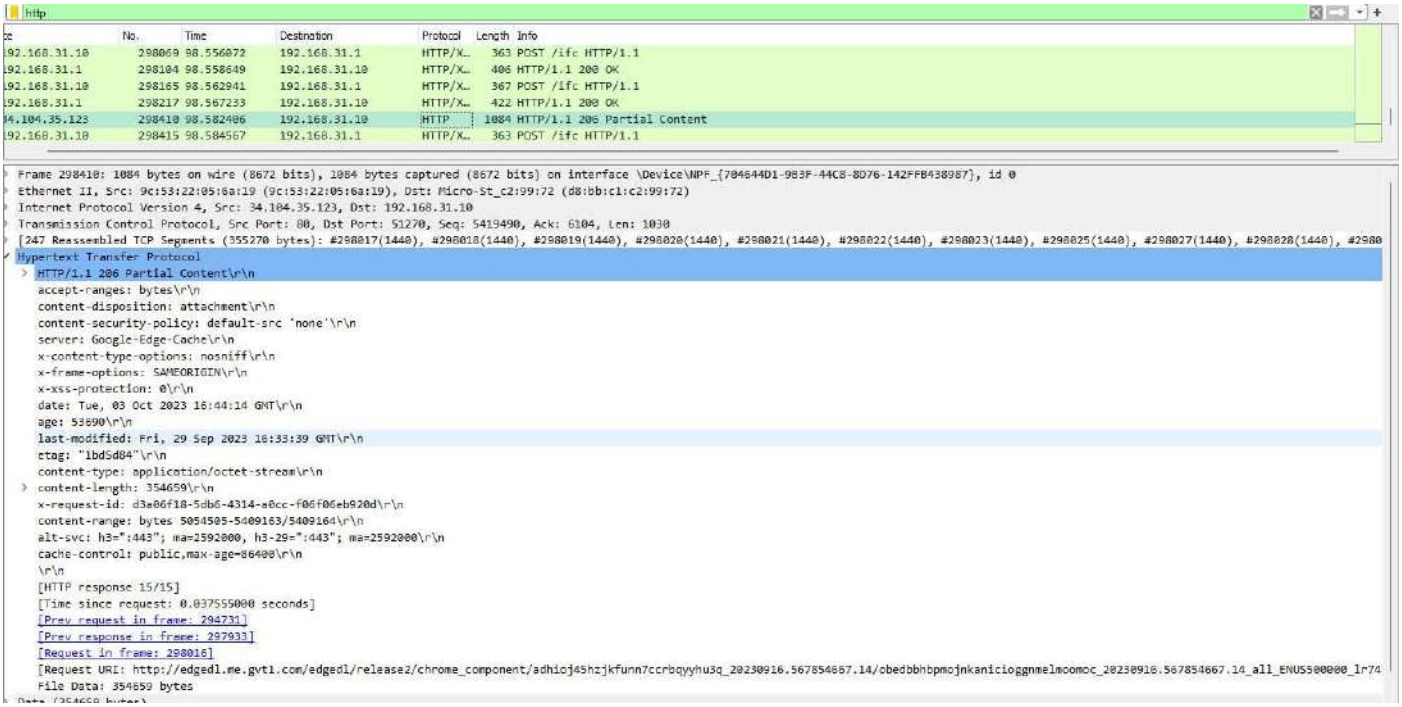
5. Dynamic Host Configuration Protocol (DHCP)



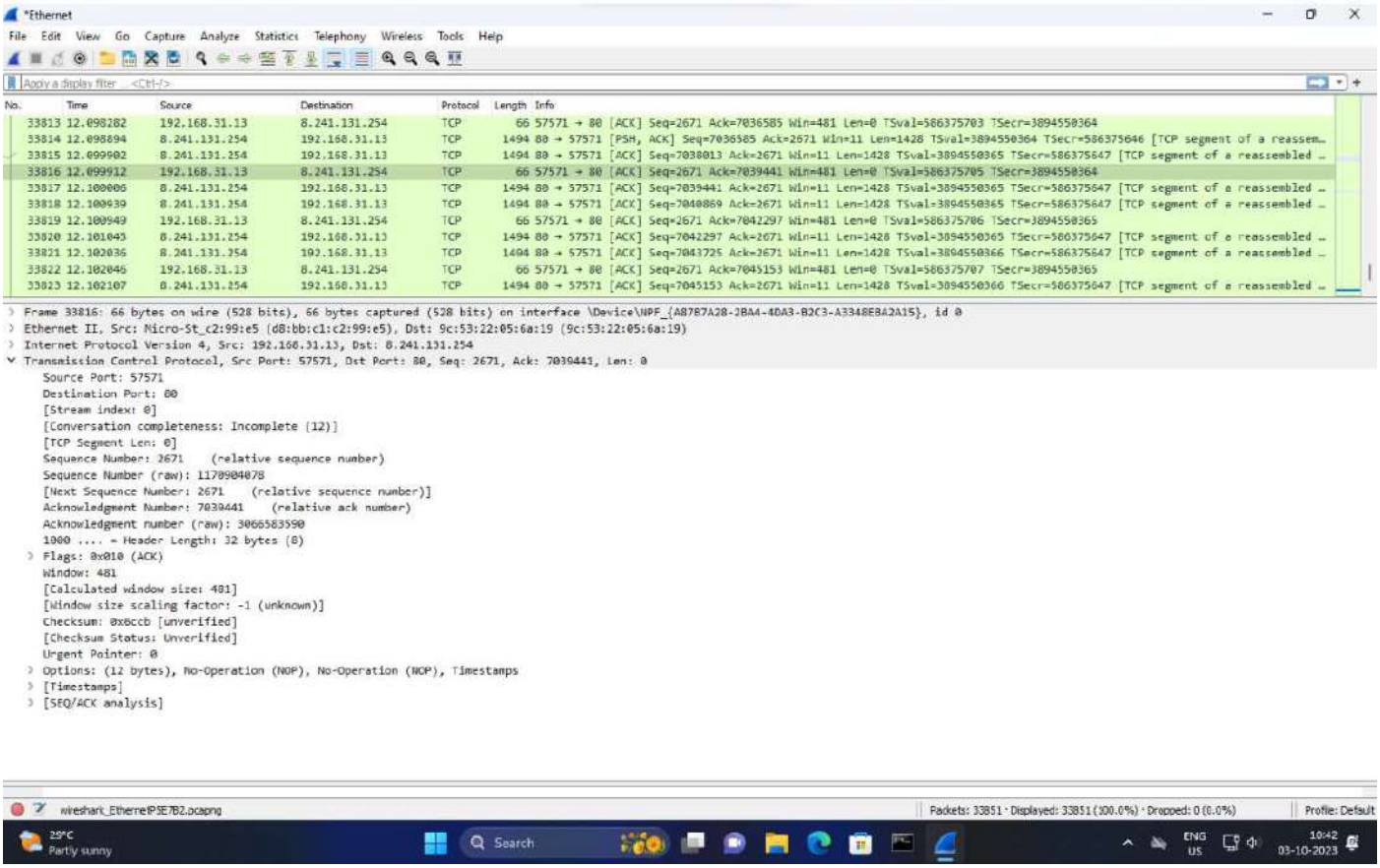
6. Domain Name System (DNS)



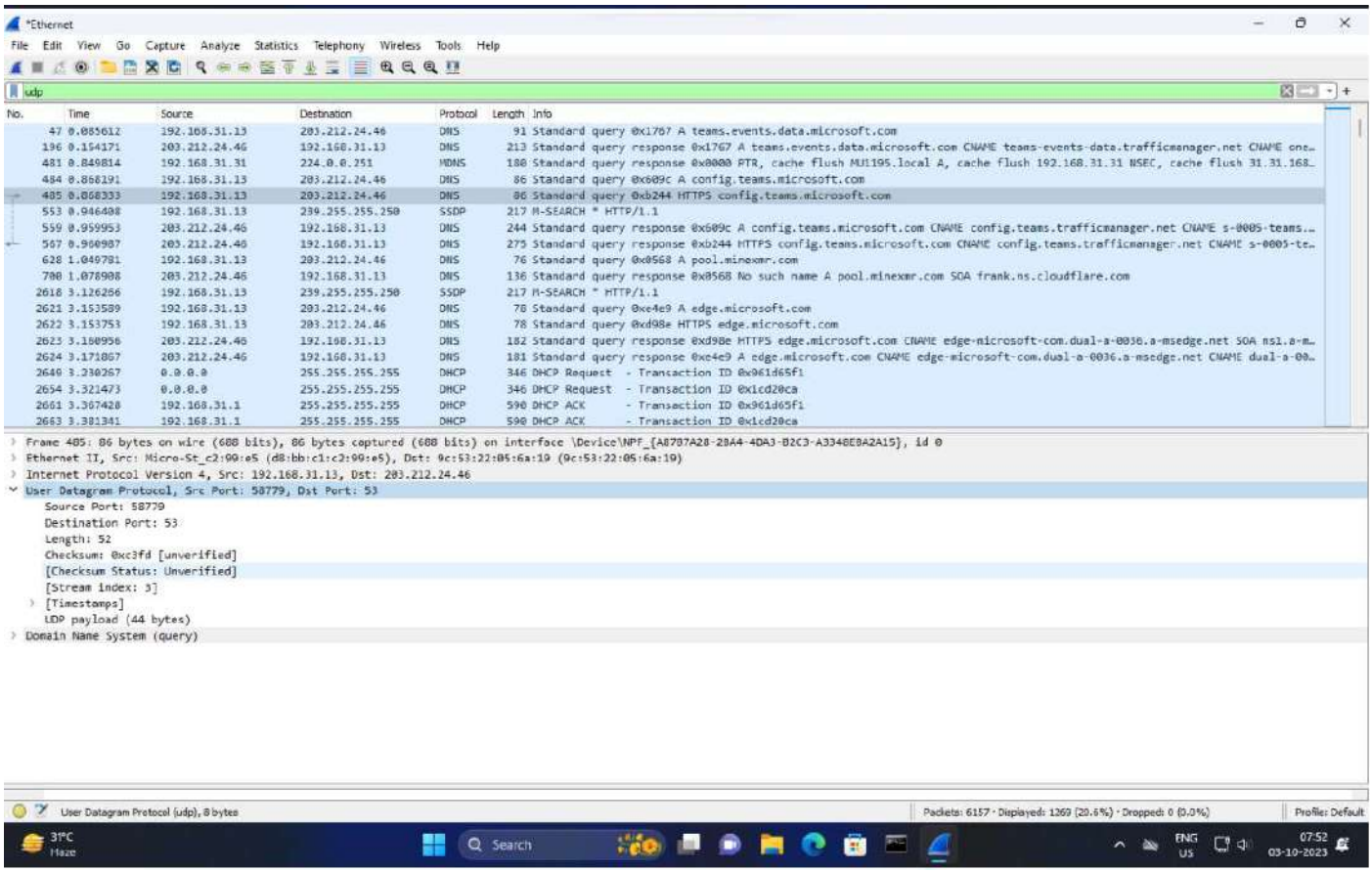
7. Hypertext Transfer Protocol (HTTP)



8. Transmission Control Protocol (TCP)



9. User Datagram Protocol (UDP)



Shirish
C3 C32
2103164

Experiment -10.

CN-C32-2103164

Aim:- Socket programming using TCP or UDP.

Theory:-

Socket programming is a crucial aspect of network communication, enabling data exchange between computers over a network. TCP(Transmission Control Protocol) and UDP(User Datagram Protocol) are two commonly used transport layer protocols for socket programming.

1. TCP

TCP ensures reliable ordered and error checked delivery of data between two devices. It establishes a connection before data exchange.

Data is sent as a stream of bytes and TCP manages the packetization and reassembly of data.

TCP uses acknowledgements to confirm the reciept of data packets and restransmits lost packet if necessary.

TCP is suitable for application where data integrity and order are crucial such as web browsing, email and file transfer.

2.) UDP

UDP dont establish a connection & doesn't guarantee delivery or order of data packets. Each UDP packet is treated as an idenpendent unit of data.
UDP doesn't use acknowledgement, so its faster but can result in lost or out-of order packets.

Socket connection consist of client side connection and server side connection.

Client-Side programming :-

To connect to another machine we need a socket connection. A socket connection means two machine have information about each other network location and TCP Port.
The java.net Socket class represents a socket. To open a socket,

Socket socket = new Socket("127.0.0.1", 5000)

The first agrument - IP address of local host where code will run on the single stand alone machine. The second argument TCP port (Just a number representing which application to run on a server. For example, HTTP runs on port 80. Port number can be from 0 to 65535.

To communicate over a socket connection, streams

used to both input and output the data.

Conclusion :- Learnt about socket programming and how to implement it in java using TCP or UDP.

03/10/23.

(B+)

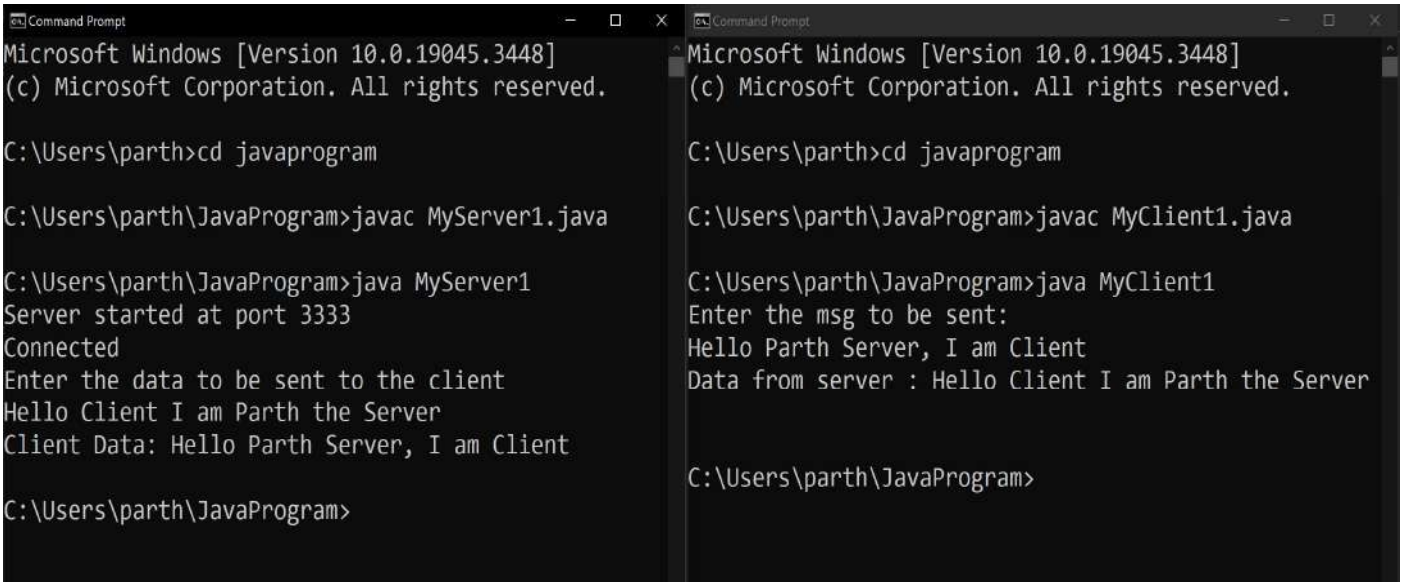
SERVER PROGRAM:

```
import java.io.*;
import java.net.*;


public class MyServer1
{
	public static void main(String args[])throws IOException
	{
		System.out.println("Server started at port 3333");
		ServerSocket ss = new ServerSocket(3333);
		Socket s = ss.accept(); //This will establish connection between client and server
		System.out.println("Connected");
		BufferedReader br1 = new BufferedReader(new InputStreamReader(System.in));
		System.out.println("Enter the data to be sent to the client");
		String nickname = br1.readLine();
		BufferedReader br = new BufferedReader(new InputStreamReader(s.getInputStream()));
		String msg = br.readLine();
		System.out.println("Client Data: "+msg);
		OutputStreamWriter os = new OutputStreamWriter(s.getOutputStream());
		PrintWriter out = new PrintWriter(os);
		out.println(nickname);
		out.flush();
		s.close();
		ss.close();
	}
}
```

CLIENT PROGRAM:

```
import java.net.*;
import java.io.*;
public class MyClient1
{
	public static void main(String args[])throws IOException
	{
		Socket s = new Socket("localhost",3333);
		System.out.println("Enter the msg to be sent: ");
		BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
		String str = br.readLine();
		OutputStreamWriter os = new OutputStreamWriter(s.getOutputStream());
		PrintWriter out = new PrintWriter(os);
		out.println(str);
		os.flush();
		BufferedReader br1 = new BufferedReader(new InputStreamReader(s.getInputStream()));
		String nickname = br1.readLine();
		System.out.println("Data from server : "+nickname);
		s.close();
	}
}
```

OUTPUT:



THADOMAL SHAHANI TSEC ENGINEERING COLLEGE

CN Assignment-1

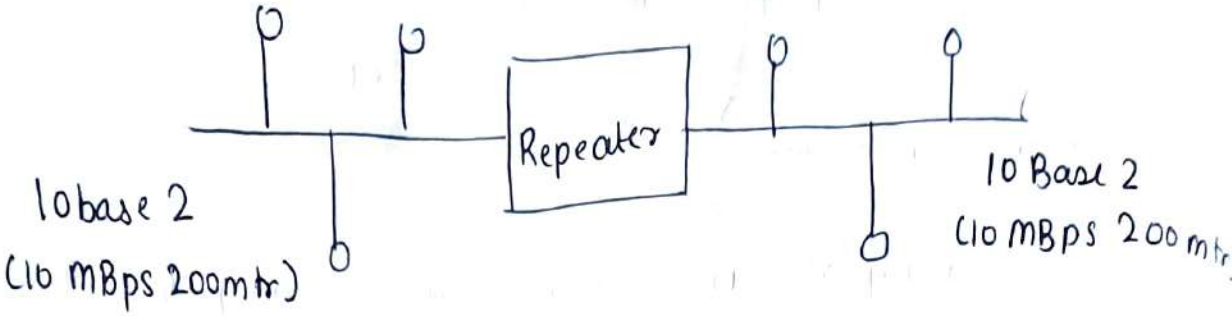
CN-C32-2103164

Write a short note on:-

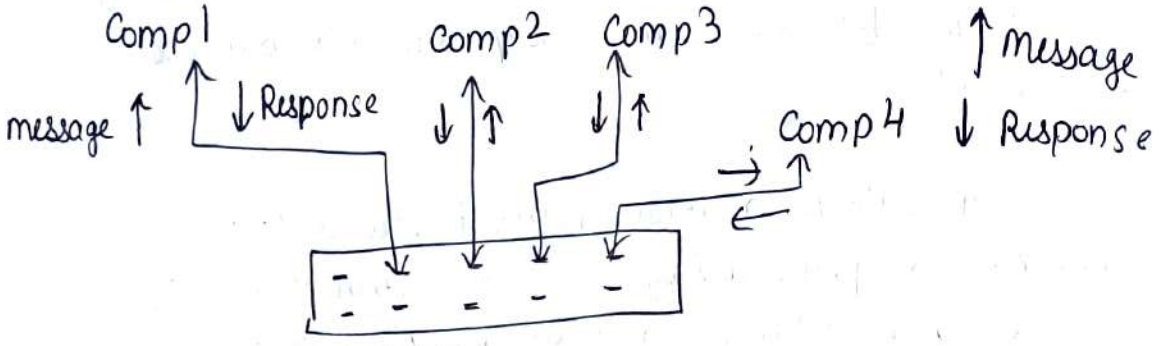
1) Repeater:- A repeater is a device used in computer networks to amplify and retransmit signals, extending the network's reach. It operates the physical layer, transparently regenerating signals without examining data content. However, advancement in networking technology have led to the decreased use of traditional repeaters in favour of more sophisticated devices like switches.

2) Hub:- A hub is a basic network device that operates at physical layer, used to connect multiple devices in a local area network (LAN). It receives incoming data from one device and broadcast it all connected devices. Hubs create single collision domain, leading to data collisions and reduced network efficiency. Due to their limitation and lack of intelligence, hubs have become outdated in modern networks and have been replaced by switches, which provide better performance and segmentation of collision domains for more efficient data transmission.

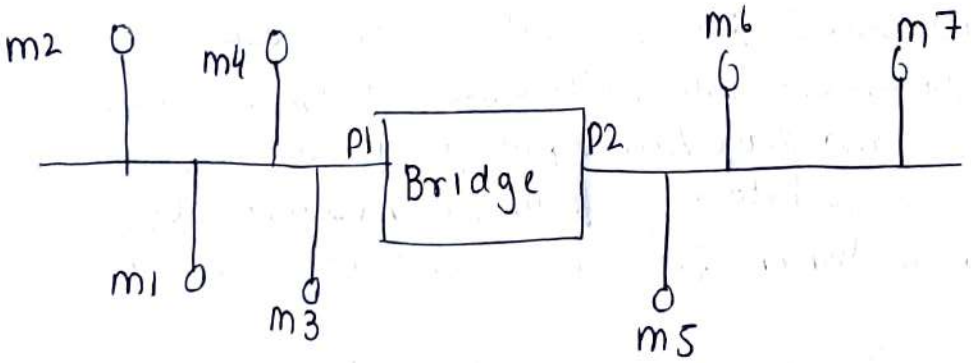
3) Bridges:- Bridges are network devices that operate at data link layer (Layer 2) of the OSI model, used to interconnect two or more LAN segments. They examine the MAC addresses of incoming data frames and maintain a table of MAC addresses and their associated segments. When a frame destination



1.) Repeater



2.) Hub



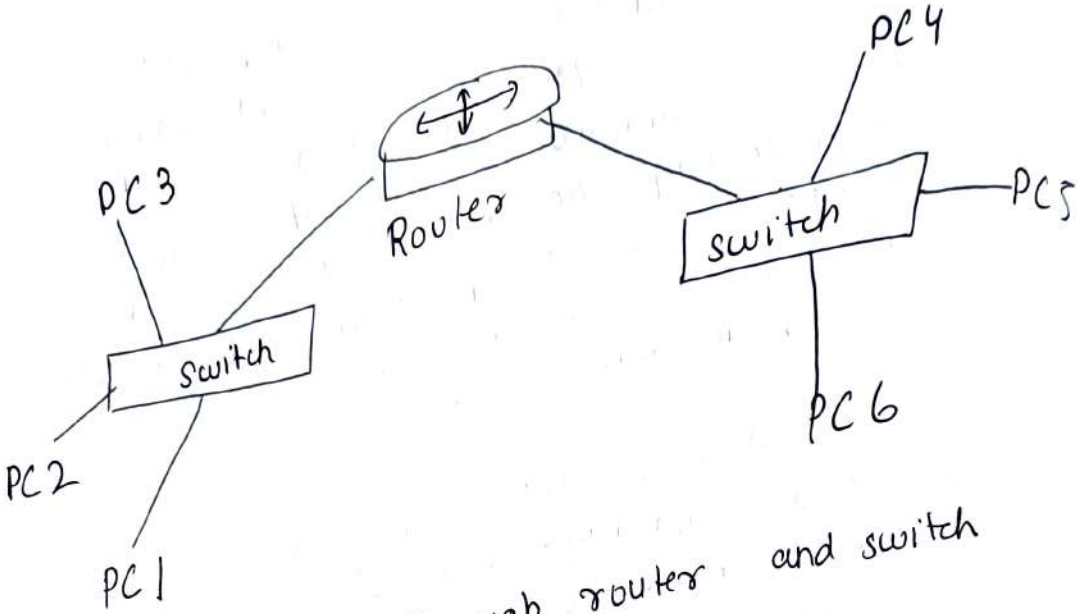
3.) Bridge

is on same segment as the source, the bridge filters the frame. If the distination is on different segment the bridge forwards the frame only to the relvant segment, reducing unnecessary network traffic. Bridges help in improving network traffic. Bridges help in improving network performance by dividing LANs into smaller collosion domains and are predecessors to modern switches.

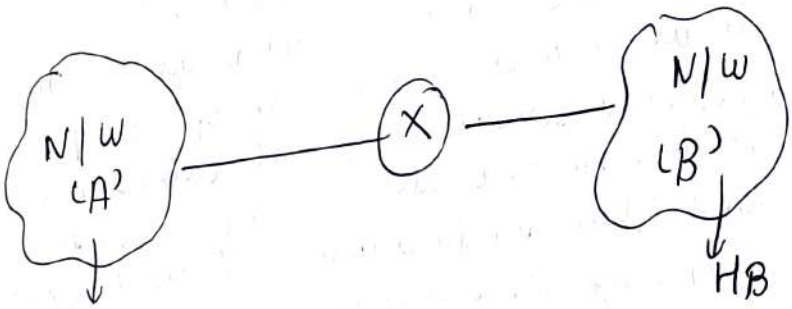
4) Switches :- Switches are integral network devices that operate at data link layer (Layer 2) of the OSI model, used to connect multiple devices within a local area network (LAN). They examine incoming data frames, make forwarding decisions based on the destination MAC address, and maintain a MAC address table to associate addresses with specific port. Unlike hubs, switches create separate collosion domain for each connected device, improving network efficiency. They are fundamental to modern networking providing efficient and reliable communication between devices in a LAN.

5) Router :- A Router is a vital network device that operates at network layer (Layer 3) of the OSI model. It connects multiple networks and make intelligent data forwarding decisions based on distination IP addresses. They maintain routing table to determine best path for data transmission. By doing so, router enable efficient data exchange, ensure proper network segmentation, provide security by acting

CN-C32-2103164



Connection through router and switch



Router

gateways

Router

Hub

Switch

Bridge

Repeater

as a gateway between networks. Routers play a crucial role in directing data traffic and enabling communication across complex & interconnected networks.

6.) Gateway :- A gateway is network device that act as a interface between different networks, protocols, or communication technologies. They play a vital role in connecting LANs to the internet, enabling access to external networks. Gateways can be hardware or software based and are essential for establishing communication between networks with distinct characteristics. In essence, gateway serve as bridges, allowing data to bomily between different network and ensuring interoperability in complex networking environments.

7.) Modem :- A modem, short for modulator-demodulator, is a network device used to convert digital data from computer or digital devices into analog signal suitable for transmission over analog communication channels. It also perform reverse process i.e converting incoming analog signal back to digital for receiving device. Modems are commonly used to provide internet access via dial-up connections and are essential for connecting to the internet over telephone lines. With the rise of broadband and digital communication technologies, traditional dial up modems have become less common, but their role in early networking history was crucial for establishing internet connectivity.

Shirish
C32

THADOMAL SHAHANI TSEC ENGINEERING COLLEGE

2103164
Shirish
C32

CN Assignment No-2

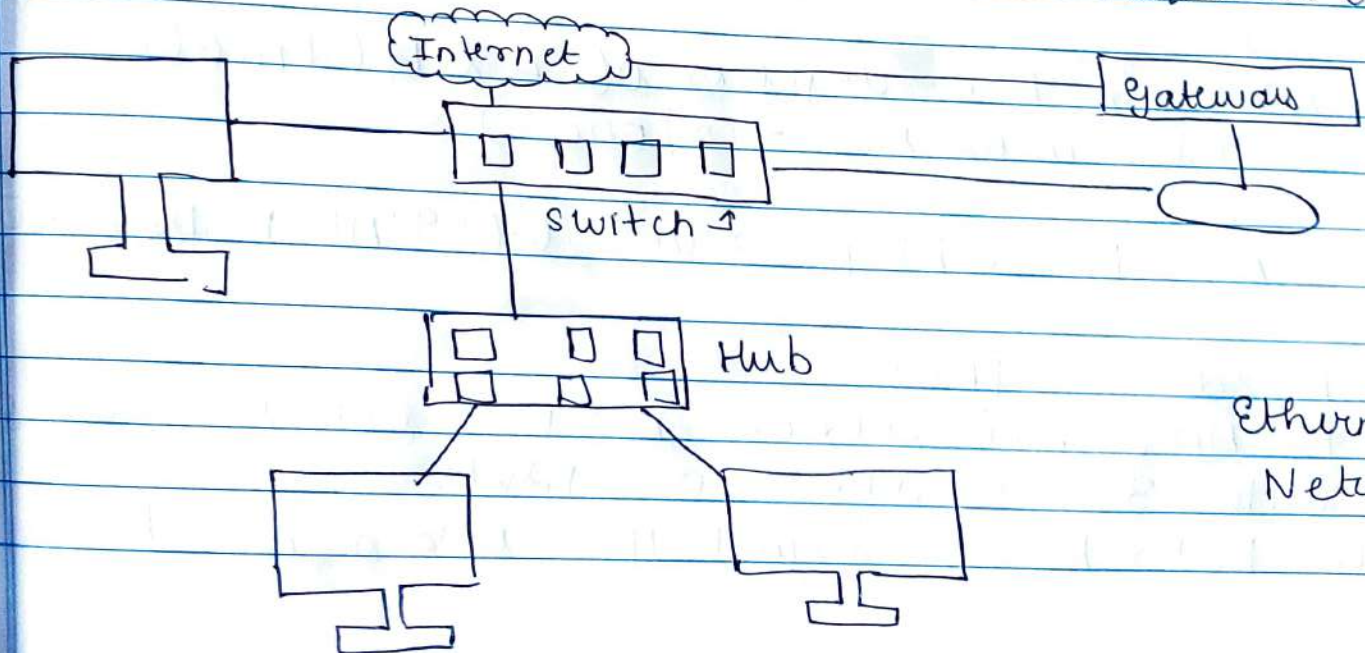
CN-C32-2103164

Write short note on

1) Ethernet

Ethernet is a type of communication protocol that connects computers on a network over a wired connection. It is all widely used LAN protocol, which is also known as Ayto Along Network. It connects computer within the local area network and wide area network. Numerous devices like printers & lap network. Can be connected by LAN & WAN within buildings, homes & even small neighbours.

The wireless networks replaced ethernet in many areas, however Ethernet is still more common for wired networking. Wi-Fi reduces the need for cabling as it allows users to connect smartphones or laptops to a network without the required cable.



Ethernet Network.

Advantages of Ethernet :-
i) It is costly, but still inexpensive than other options
ii) Provides high security.
iii) Quality of data is maintained.

Disadvantages of ethernet :-
i) Distance covered is less
ii) No acknowledgement from receivers side
iii) Can't determine which node is faulty.

② IPV6 :-
IPV6 was developed by Internet Engineering Task Force (IETF) to deal with the problem IPV4 exhaustion. IPV6 is a 128 bit address having an address space of 2^{128} which is way bigger than IPV4 IPV6 use Hexadecimal format separated by colons :)
Components of IPV6 address.
i) 8 - groups, each group represent 2 byte (16 bits.
ii) Each Hex - digit is of 4 bits (1 nibble)
iii) Delimeter used - colons :)

ABCD : EFD1 : 2345 : 6789 : ABCD : D201 • 548

Need of IPV6
i) An IPV6 address is 128 bits long compared with 32 bit address of IPV4.
ii) Better header format, IPV6 uses a new header

format in which options are separated from the base header & inserted when needed between the base header & the upper layer data. This simplifies i.e speed up the routing process.
(iii) IPV6 has new options to allow for functionalities
(iv) IPV6 allows extension of the protocol if required by new technologies.

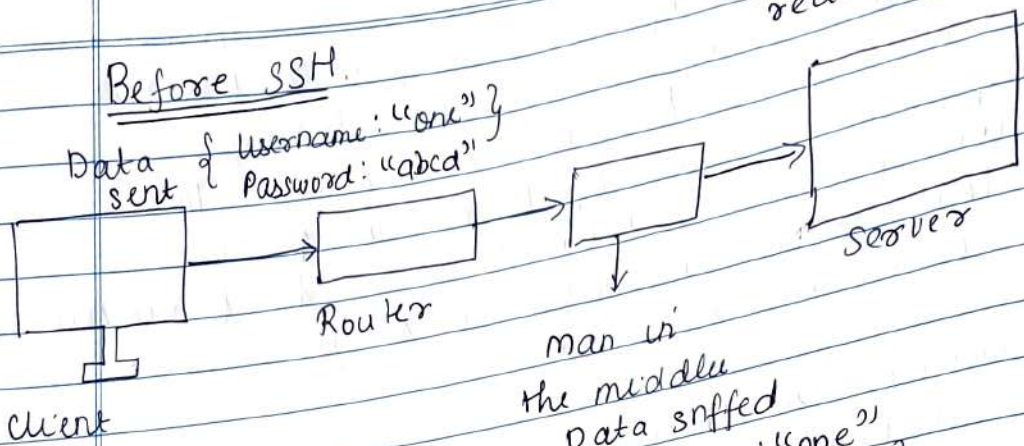
Advantages of IPV6.
i) Real time data transmission
ii) Supports authentication
iii) Performs encryption.
iv) Faster processing at router side.

) SSH.
SSH stand for secure shell or secure socket shell. It is cryptographic network protocol that allow two computers to communicate & share the data over an insecure network such as the internet. It is used to login to a Internet. It is used to login to a remote server to execute commands & data transfer from one machine to another machine.

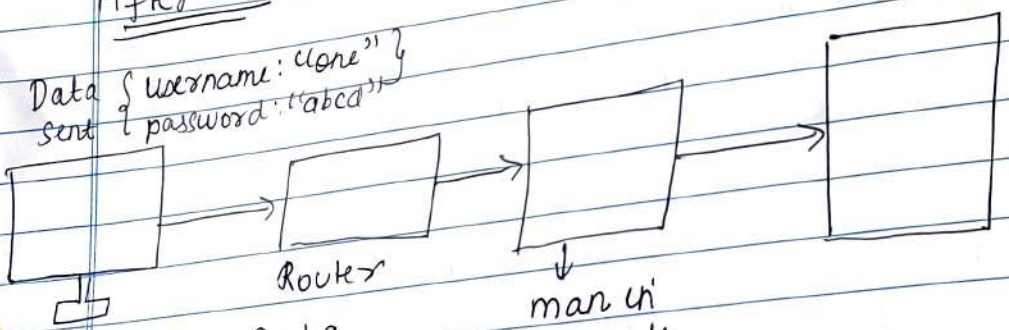
It provides a strong password encryption & password authentication communication with public key over insecure channel. It is used to replace unprotected remote login protocols such as telnet, rlogin, rsh, etc.

CN-C32 2203164

Before SSH



After SSH



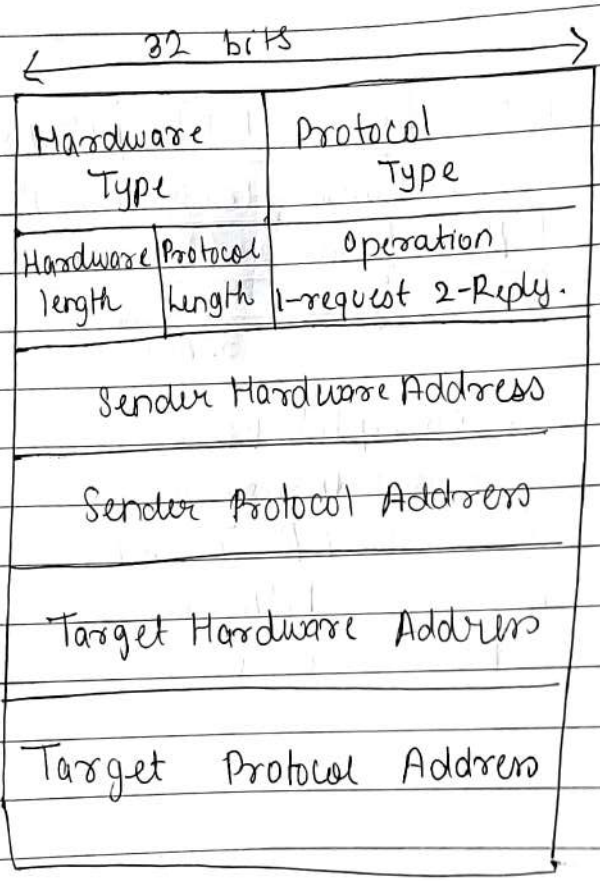
Advantages of SSH:
It prevent malicious activities like DNS spoofing, Data manipulation, eavesdropping or sniffing of transmitted data, IP address routing.

CN - CS 2103164

(iii) SSH enables port forwarding

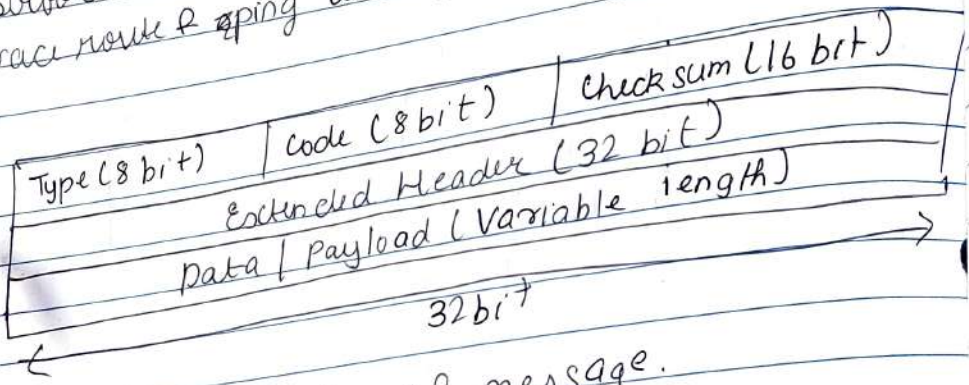
.) Explain purpose of following protocols with header format.
i) ARP ii) ICMP iii) DNS

) -i) ARP: The Address Resolution Protocol (ARP) is a fundamental networking protocol that plays a pivotal role in local network communication by mapping known IP address to their corresponding physical. Mac addresses. It enables devices on a network to discover & associate. Mac addresses with IP addresses.



ii) ICMP: ICMP is used for error reporting. It two devices connect over the internet & some error and occurs. So, the router send an ICMP error message to the source informing about the error. For example, whenever a device a device sends any message which is large enough for the reciever in that the reciever will drop the message & reply back ICMP message to the source.

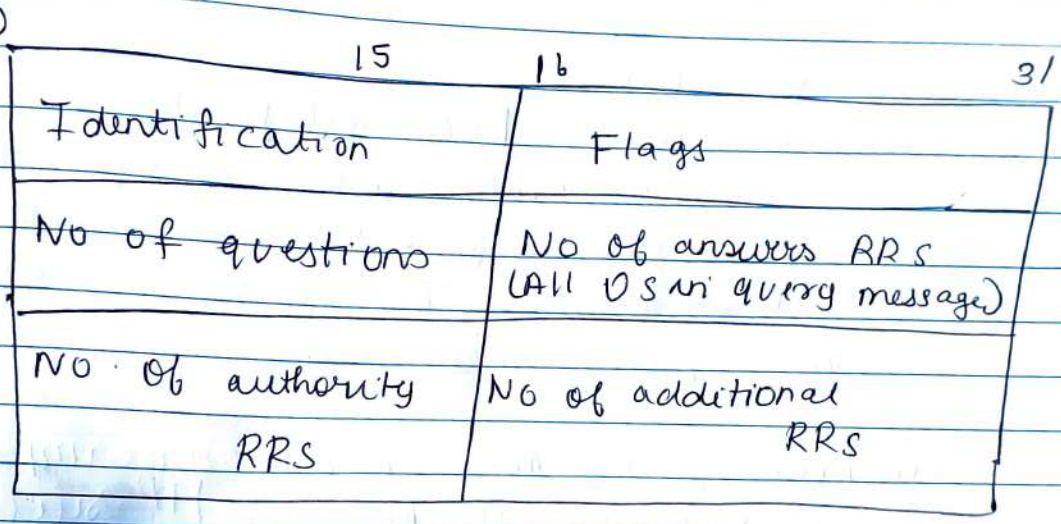
Another important use of ICMP protocol is used to perform network diagonous. by making use of trace route & ping utiling. We will discuss one by one.



Type : Basic description of message.
Code : Additional information
Checksum : Checksum of ICMP
Extended header :- Point out problem in IP messa
Data or Payload of variable length.

CN - CS2 - 2103164

1) DNS: The Domain Name System (DNS) serves vital role of translating user friendly domain names into numerical IP addresses, enabling seamless internet communication. It ensures user can access website & services using easily memorable names, simplyfying web browsing & resources location. It also provides data packet addressing & routing.



Indentification: Used to match the response with the request

Flags: 16 bit flags for indication of values.

No of Quations, number of authority RRs, number of answer RRs. & number of additional RPs are all explanatory.

CN-C32-2103164

Q.3) Dicuss the persistent & non persistent protocols used in transport & application layer of TCP/IP protocol su

Ans → Persistent & non persistent protocols II D are two different approches to handling connections in the Transport & Application layer of the TCP/ protocol suite. These protocols govern how data is exchanged between two devices over a network.

Persistent protocol :-

i) Connection oriented :- Persistent protocols are connection-oriented which means they establish a connection between sender & reciever before data exchange begins. This connection remains open for multiple transactions.

(ii) Examples :- The most common example of persistent protocol in the TCP/IP suite is the Transmission Control Protocol (TCP). TCP ensur reliable, ordered, error check delivery of data. It maintains a continous until explicitly closed by either the sender or reci

(iii) Use Cases : Persistent protocols are suitab for applications that requires reliable & ord data transfer such as web browsing, em file transfer & most client interactions. The are ideal for situations where data integrity correctness are crucial.

(iv) Overhead: They typically have more overhead due to the need for connection setup & maintaince which involves three-way hand hakes, acknowledgement message & error recovery mechanism.

Non-persistent protocol :-

(i) Connectionless :- Non persistent protocols are connectionless, meaning they donot establish a continous connection between sender & receiver. Instead a new connection is established for each data exchange & it is closed after that exchange is complete.

(ii) Examples: A common example is User Datagram Protocol (UDP)

(iii) Use cases :- Non persistent protocols are suitable for applications that prioritize speed & efficiency over reliability.
Commonly used in real time data sharing.

(iv) Overhead :- Lesser overhead as compared to persistent protocols because they skip connection setup.

Persistent protocols :-
Application : HTTP, FTP, SMTP
Transport : TCP

Non persistent protocols :-
Application Layer :- DNS, SNMP, DHCP
Transport :- UDP, SCTP