

**COMPUTER NETWORK LAB RECORD**

**S.Jaswanth**

**NAME:**

**2021-2022**

**122010325012**

**REDG NO: YEAR:**

**CSE -CS**

**4TH SEM**

**S**

**B25**

**BRANCH: SEM: SEC:**

**CN LAB**

**19ECS232P**

**COURSE: COURSE CODE:**

**EXPERIMENT - 1**

**Objective: Write a report that includes a diagram showing the topology, type of connection devices, and speed of the wired and wireless LAN in your organization. Also find out the MAC and IP addresses and the subnet mask of your computer.**

1.Topology: The arrangement of a network that comprises nodes and connecting lines via sender and receiver is referred to as network topology. Topology simply means that how the end devices are connected to each other.

Types of Topologies:

**A) MESH TOPOLOGY:**

In a mesh topology, every device is connected to every another device via a particular channel. These channels are known as links. 



If N number of devices is connected with each other in a mesh topology, the total number of links that are required by each device is N-1. In the Figure, there are 5 devices connected to each other, hence the total number of links required by each device is 4. Total number of links required=N\*(N-1)/2.

* Suppose, N number of devices are connected with each other in a mesh topology, then the total number of dedicated links required to connect them is NC2 i.e., N(N-1)/2. In In the Figure, there are 5 devices connected to each other, hence the total number of links required is 5\*4/2 = 10.

**Advantages of this topology:**

* It is robust.
* The fault is diagnosed easily. Data is reliable because data is transferred among the devices through dedicated channels or links.
* Provides security and privacy.

**Problems with this topology:**

* Installation and configuration are difficult.
* The cost of cables is high as bulk wiring is required, hence suitable for less number of devices.
* The cost of maintenance is high.

**B) STAR TOPOLOGY:**

In star topology, all the devices are connected to a single hub through a cable. This hub is the central node and all other nodes are connected to the central node. The hub can be passive in nature i.e., not an intelligent hub such as broadcasting devices, at the same time the hub can be intelligent known as an active hub. Active hubs have repeaters in them.



**Advantages of this topology:**

* If N devices are connected to each other in a star topology, then the number of cables required to connect them is N. So, it is easy to set up.
* Each device requires only 1 port i.e. to connect to the hub, therefore the total number of ports required is N.

**Problems with this topology:**

* If the concentrator (hub) on which the whole topology relies fails, the whole system will crash down.
* The cost of installation is high.
* Performance is based on the single concentrator i.e. hub.

**C) BUS TOPOLOGY:**

Bus topology is a network type in which every computer and network device is connected to a single cable. It transmits the data from one end to another in a single direction. No bi-directional feature is in bus topology. It is a multi-point connection and a non-robust topology because if the backbone fails the topology crashes.



**Advantages of this topology:**

* If N devices are connected to each other in a bus topology, then the number of cables required to connect them is 1, which is known as backbone cable, and N drop lines are required.
* The cost of the cable is less as compared to other topologies, but it is used to build small networks.

**Problems with this topology:**

* If the common cable fails, then the whole system will crash down.
* If the network traffic is heavy, it increases collisions in the network. To avoid this, various protocols are used in the MAC layer known as Pure Aloha, Slotted Aloha, CSMA/CD, etc.
* Security is very low.

**D) RING TOPOLOGY:**

In this topology, the connected devices form a ring with its exactly two neighboring devices.

A number of repeaters are used for Ring topology with a large number of nodes, because if someone wants to send some data to the last node in the ring topology with 100 nodes, then the data will have to pass through 99 nodes to reach the 100th node. Hence to prevent data loss repeaters are used in the network.

The transmission is unidirectional, but it can be made bidirectional by having 2 connections between each Network Node, it is called Dual Ring Topology.



The following operations take place in ring topology are : 

1. One station is known as a **monitor** station which takes all the responsibility to perform the operations.
2. To transmit the data, the station has to hold the token. After the transmission is done, the token is to be released for other stations to use.
3. When no station is transmitting the data, then the token will circulate in the ring.
4. There are two types of token release techniques: **Early token release** releases the token just after transmitting the data and **Delay token release** releases the token after the acknowledgment is received from the receiver.

**Advantages of this topology:**

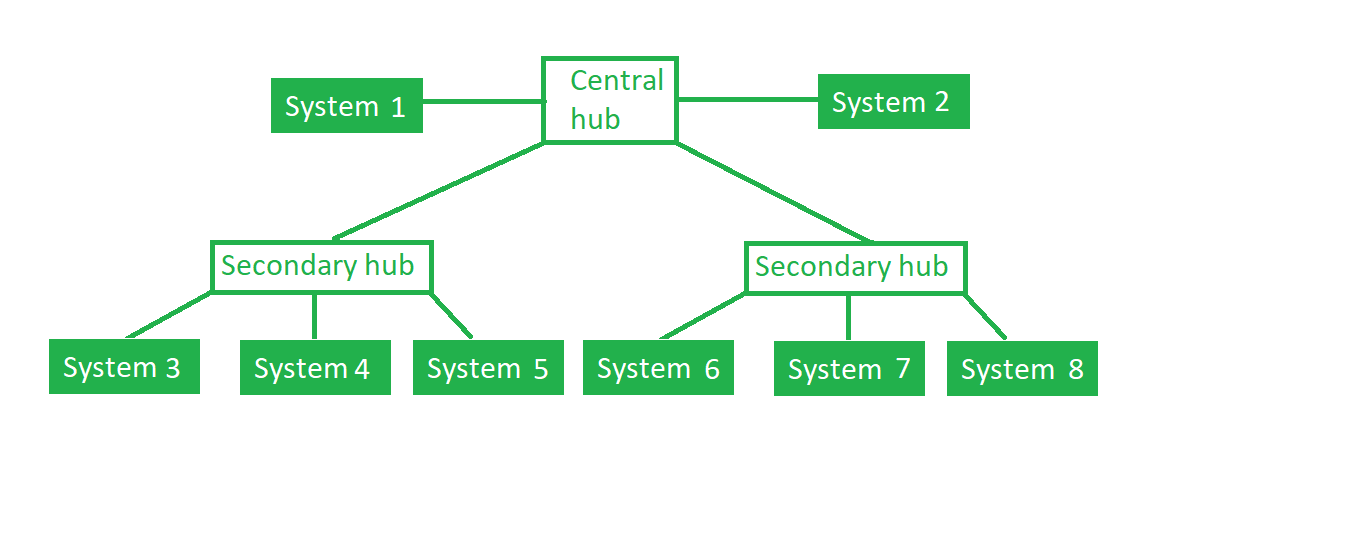
* The possibility of collision is minimum in this type of topology.
* Cheap to install and expand.

**Problems with this topology:**

* Troubleshooting is difficult in this topology.
* The addition of stations in between or removal of stations can disturb the whole topology.
* Less secure.

**E) TREE TOPOLOGY:**

This topology is the variation of Star topology. This topology has a hierarchical flow of data.



In this, the various secondary hubs are connected to the central hub which contains the repeater. In this data flow from top to bottom i.e. from the central hub to secondary and then to the devices or from bottom to top i.e. devices to the secondary hub and then to the central hub. It is a multi-point connection and a non-robust topology because if the backbone fails the topology crashes.

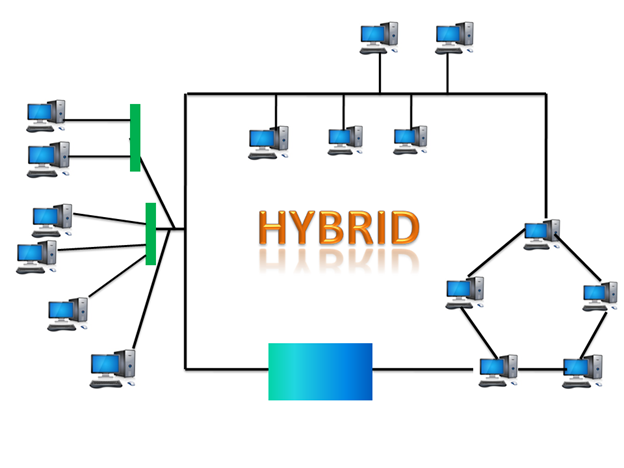
**Advantages of this topology:**

* It allows more devices to be attached to a single central hub thus it decreases the distance that is traveled by the signal to come to the devices.
* It allows the network to get isolate and also prioritize from different computers.

**Problems with this topology:**

* If the central hub gets fails the entire system fails.
* The cost is high because of cabling.

**F) HYBRID TOPOLOGY**



* The combination of various different topologies is known as **Hybrid topology**.
* A Hybrid topology is a connection between different links and nodes to transfer the data.
* When two or more different topologies are combined together is termed as Hybrid topology and if similar topologies are connected with each other will not result in Hybrid topology. For example, if there exists a ring topology in one branch of ICICI bank and bus topology in another branch of ICICI bank, connecting these two topologies will result in Hybrid topology.

### Advantages of Hybrid Topology

* **Reliable:** If a fault occurs in any part of the network will not affect the functioning of the rest of the network.
* **Scalable:** Size of the network can be easily expanded by adding new devices without affecting the functionality of the existing network.
* **Flexible:** This topology is very flexible as it can be designed according to the requirements of the organization.
* **Effective:** Hybrid topology is very effective as it can be designed in such a way that the strength of the network is maximized and weakness of the network is minimized.

### Disadvantages of Hybrid topology

* **Complex design:** The major drawback of the Hybrid topology is the design of the Hybrid network. It is very difficult to design the architecture of the Hybrid network.
* **Costly Hub:** The Hubs used in the Hybrid topology are very expensive as these hubs are different from usual Hubs used in other topologies.
* **Costly infrastructure:** The infrastructure cost is very high as a hybrid network requires a lot of cabling, network devices, etc.

**2.Connection Devices**

**Physical Layer Devices:**

**Repeater** – A repeater operates at the physical layer. Its job is to regenerate the signal over the same network before the signal becomes too weak or corrupted so as to extend the length to which the signal can be transmitted over the same network. An important point to be noted about repeaters is that they do not amplify the signal. When the signal becomes weak, they copy the signal bit by bit and regenerate it at the original strength. It is a 2-port device.

**Hub** – A hub is basically a multiport repeater. A hub connects multiple wires coming from different branches, for example, the connector in star topology which connects different stations. Hubs cannot filter data, so data packets are sent to all connected devices.  In other words, the collision domain of all hosts connected through Hub remains one.  Also, they do not have the intelligence to find out the best path for data packets which leads to inefficiencies and wastage.

**Link Layer Devices:**

**Bridge** – A bridge operates at the data link layer. A bridge is a repeater, with add on the functionality of filtering content by reading the MAC addresses of source and destination. It is also used for interconnecting two LANs working on the same protocol. It has a single input and single output port, thus making it a 2-port device.

**Switch** – A switch is a multiport bridge with a buffer and a design that can boost its efficiency (a large number of ports imply less traffic) and performance. A switch is a data link layer device. The switch can perform error checking before forwarding data, which makes it very efficient as it does not forward packets that have errors and forward good packets selectively to the correct port only.  In other words, the switch divides the collision domain of hosts, but broadcast domain remains the same.

**NIC** – NIC or network interface card is a network adapter that is used to connect the computer to the network. It is installed in the computer to establish a LAN.  It has a unique id that is written on the chip, and it has a connector to connect the cable to it. The cable acts as an interface between the computer and router or modem. NIC card is a layer 2 device which means that it works on both physical and data link layer of the network model. 

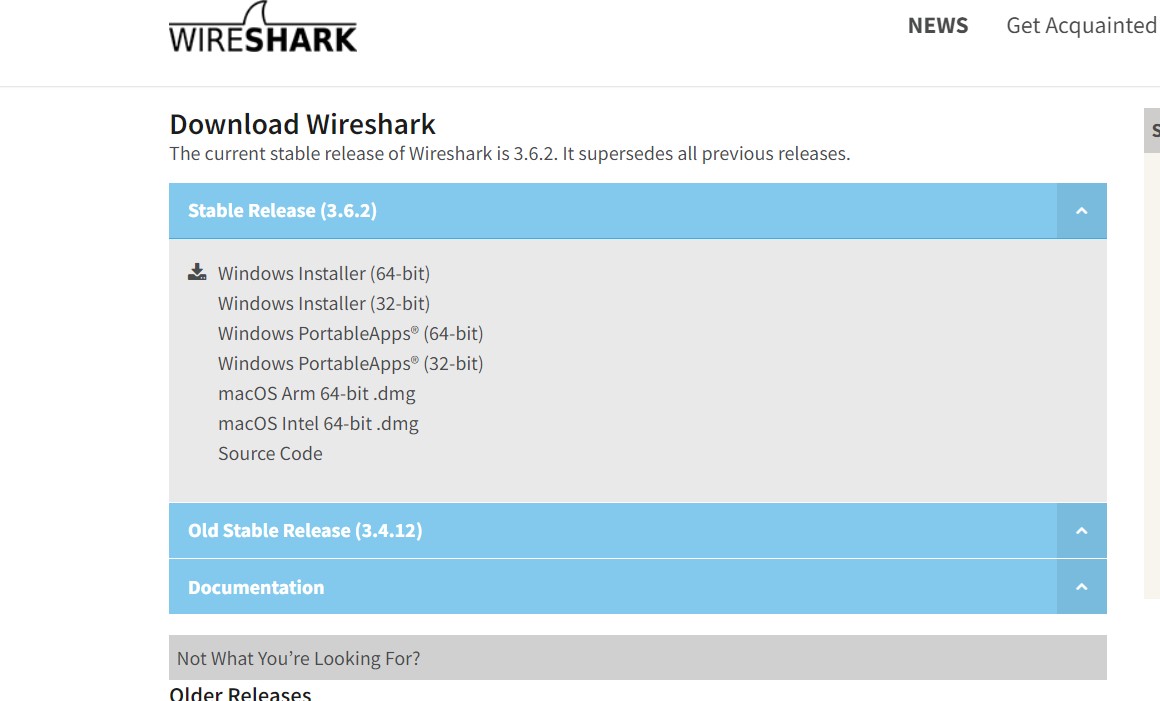
**Network layer Devices:**  
    
 **Routers** – A router is a device like a switch that routes data packets based on their IP addresses. The router is mainly a Network Layer device. Routers normally connect LANs and WANs together and have a dynamically updating routing table based on which they make decisions on routing the data packets. Router divide broadcast domains of hosts connected through it.

**Gateway** – A gateway, as the name suggests, is a passage to connect two networks together that may work upon different networking models. They basically work as the messenger agents that take data from one system, interpret it, and transfer it to another system. Gateways are also called protocol converters and can operate at any network layer. Gateways are generally more complex than switches or routers.

**EXPERIMENT - 2**

**Install and run a network diagnosis tool such as Tcp dump or Wireshark. Start capturing packets on an active interface, open a browser and type the address of your favourite search engine. Wait till the page loads and stop capture. List out the type and number of each type of packets captured.**

**INSTALLING WIRESHARK:**

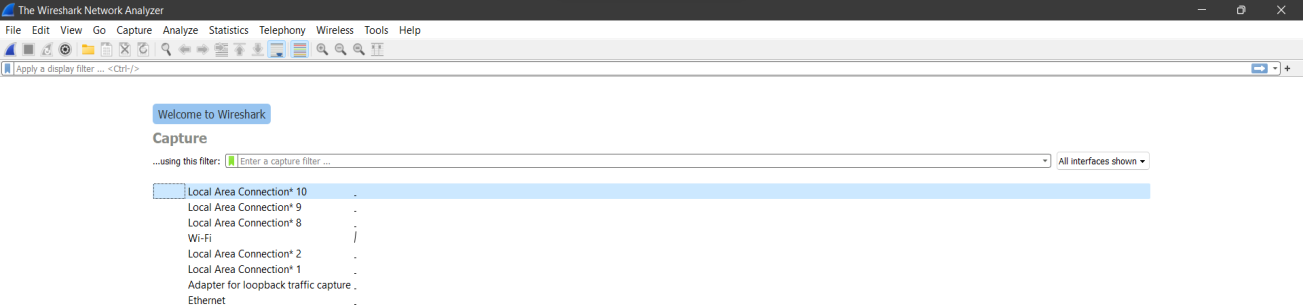


Go to: <https://www.wireshark.org/download.html>

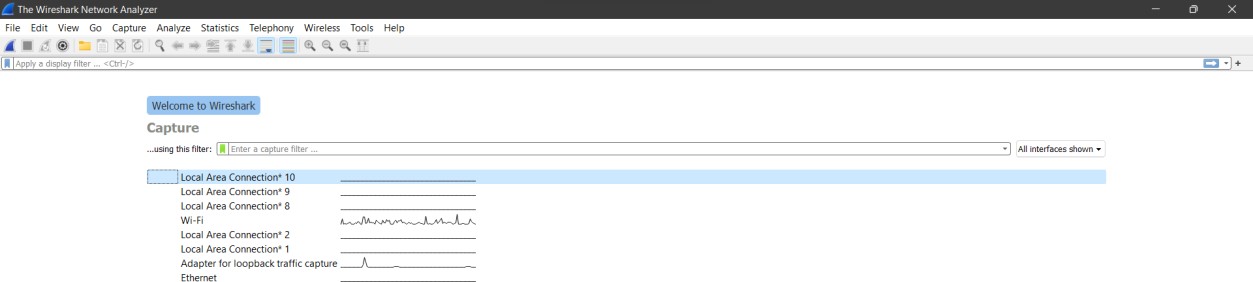
Open the Wireshark after installing it.

“WIFI”:

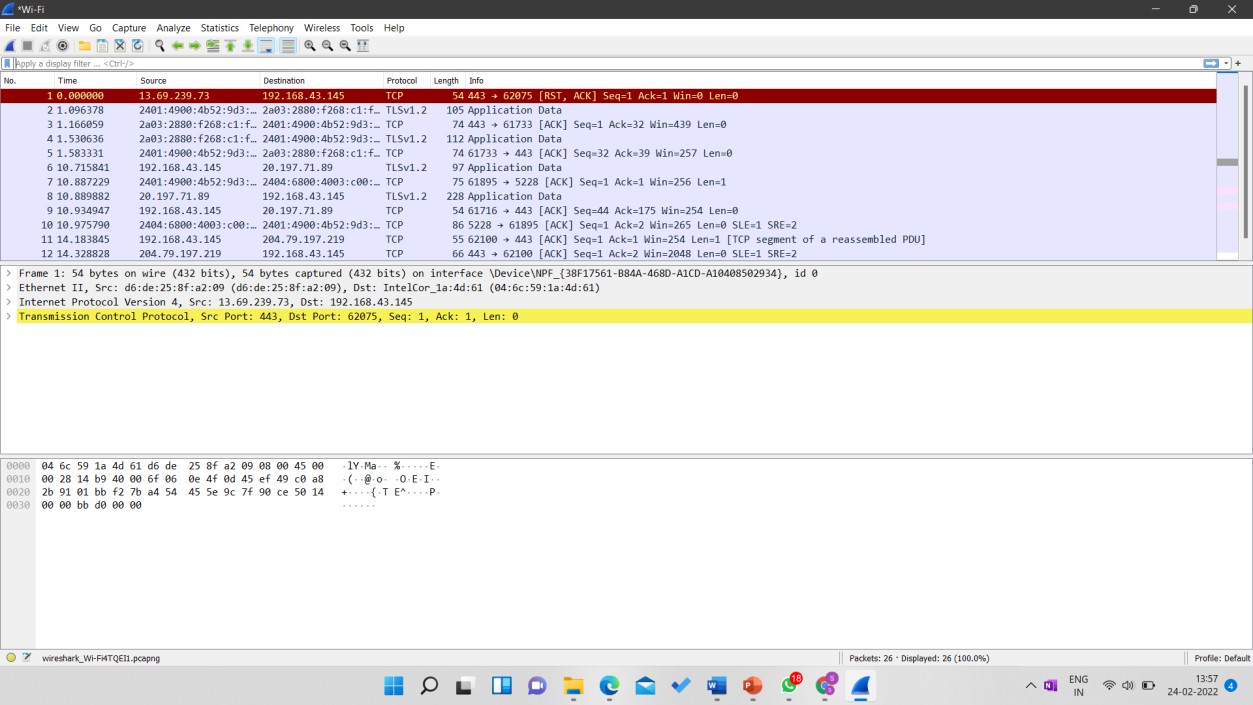
BEFORE CONNECTION:



AFTER CONNECTION:



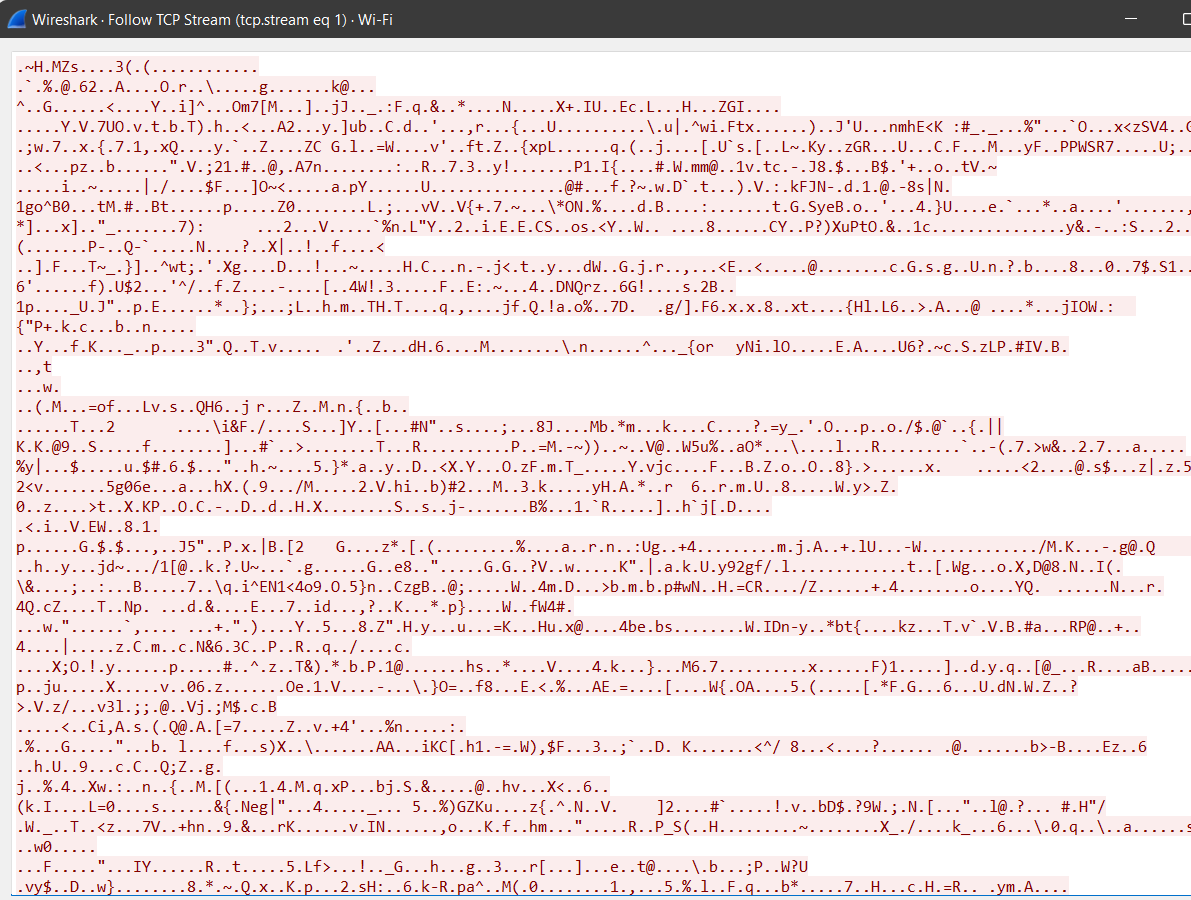
CAPTURING PACKETS:



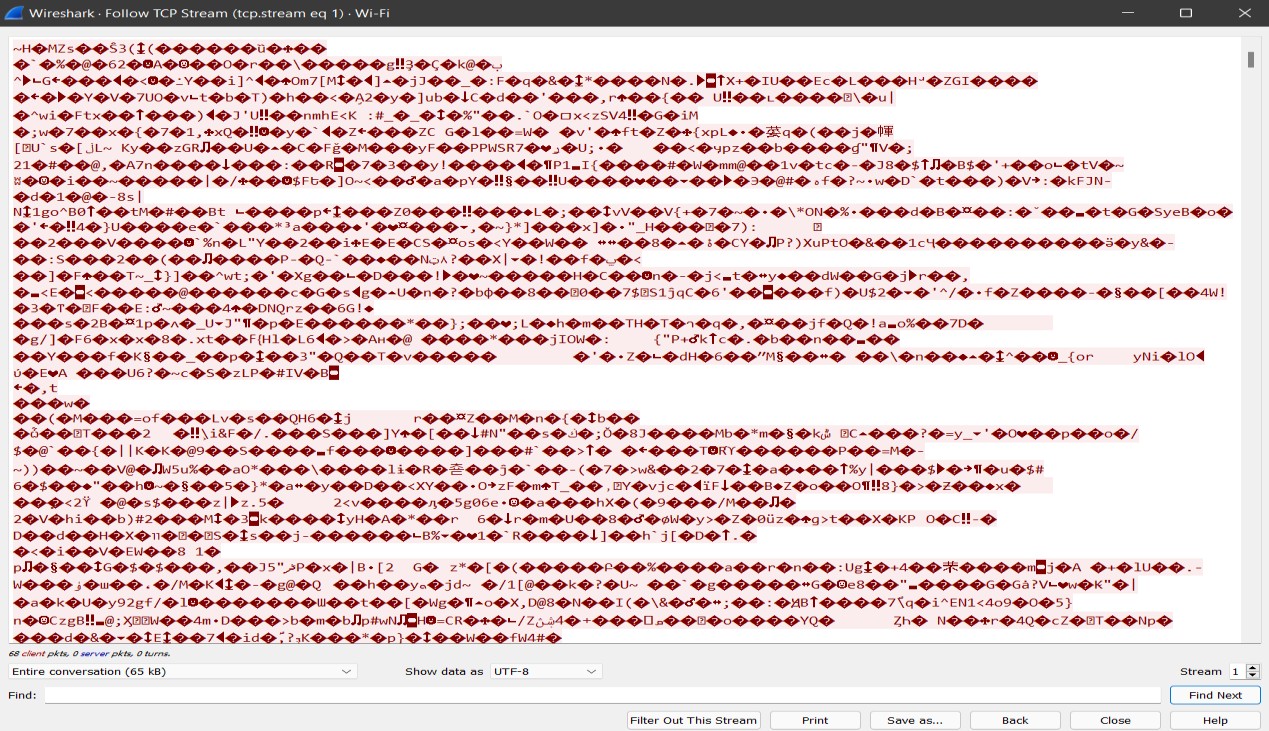
**Type of packets captured.**

1.TCP:

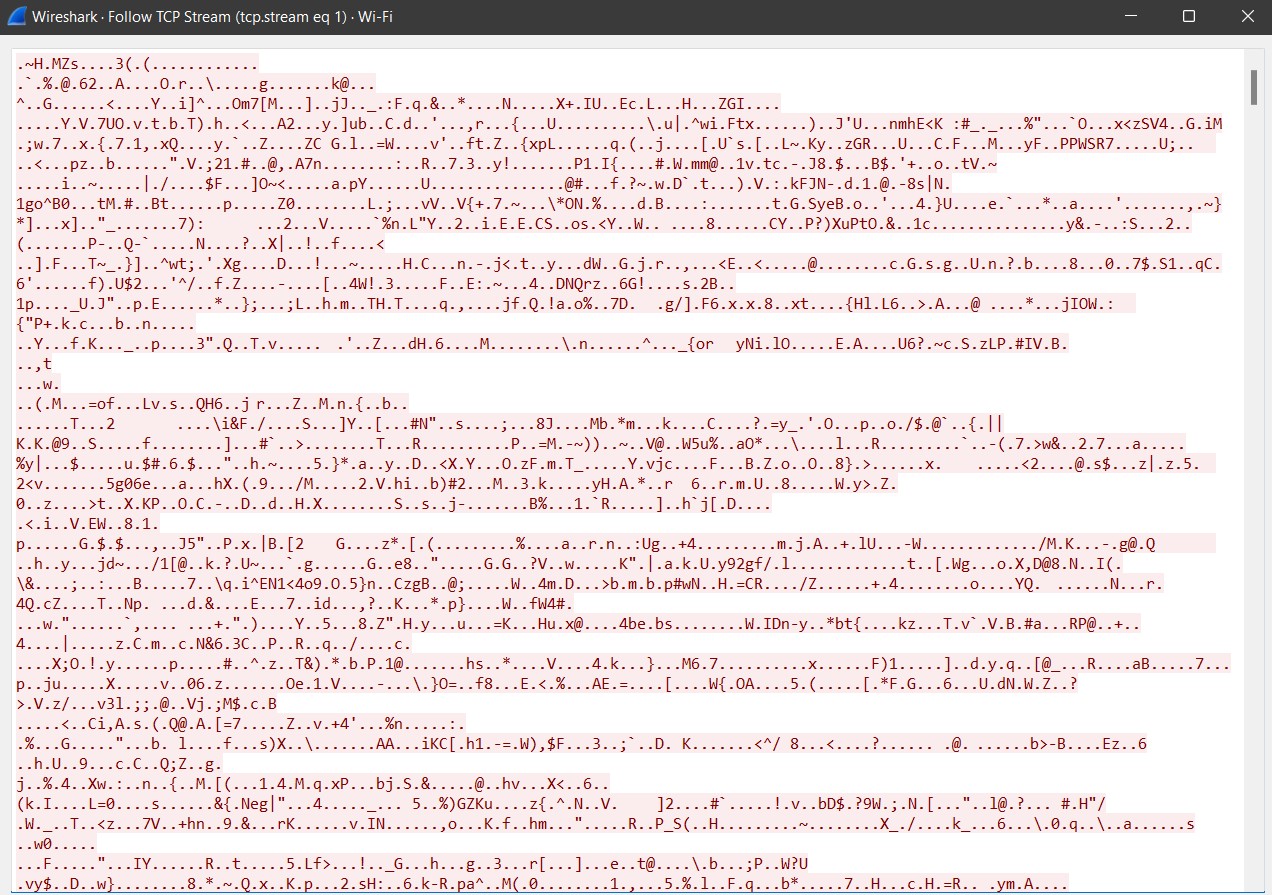
1. ASCII CODE:

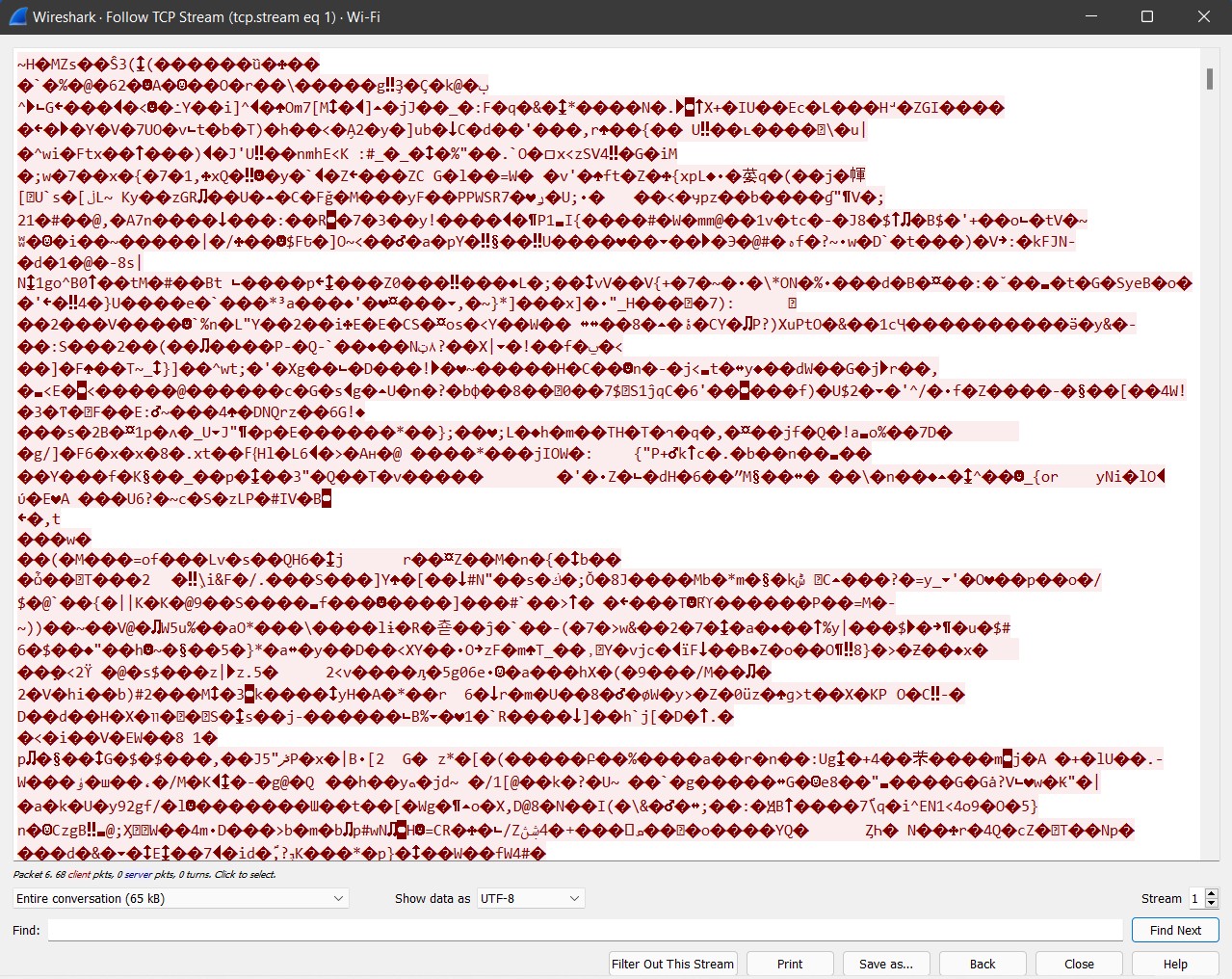


1. UTF-8:

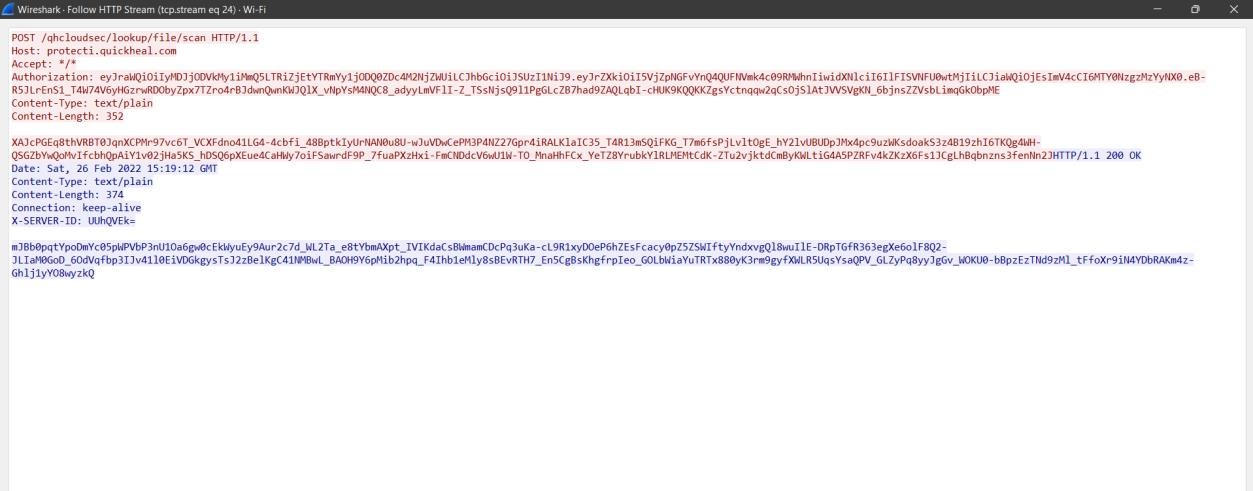


1. SSL (through TCP stream): ASCII:



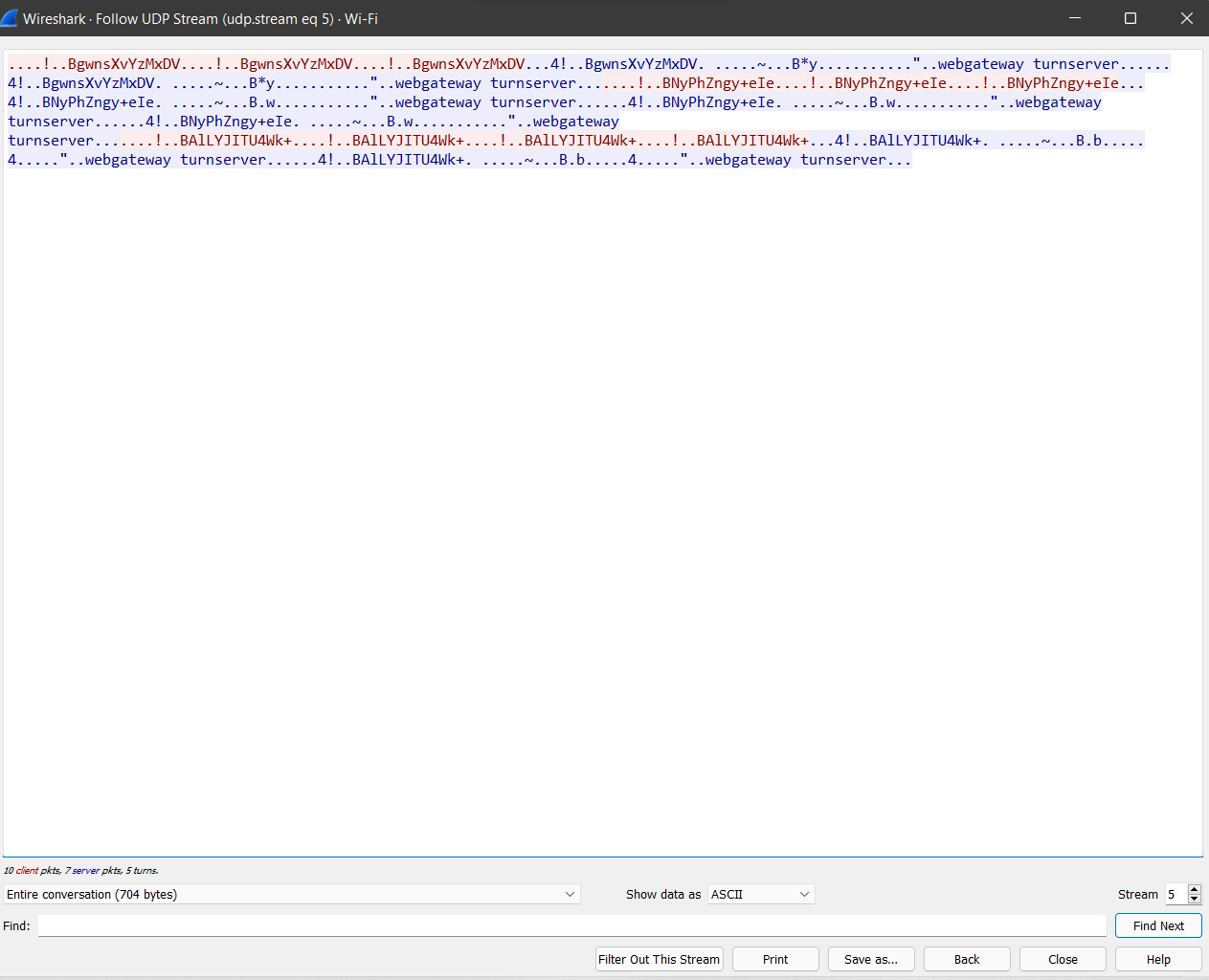


2.HTTP: (through HTTP stream) ASCII:

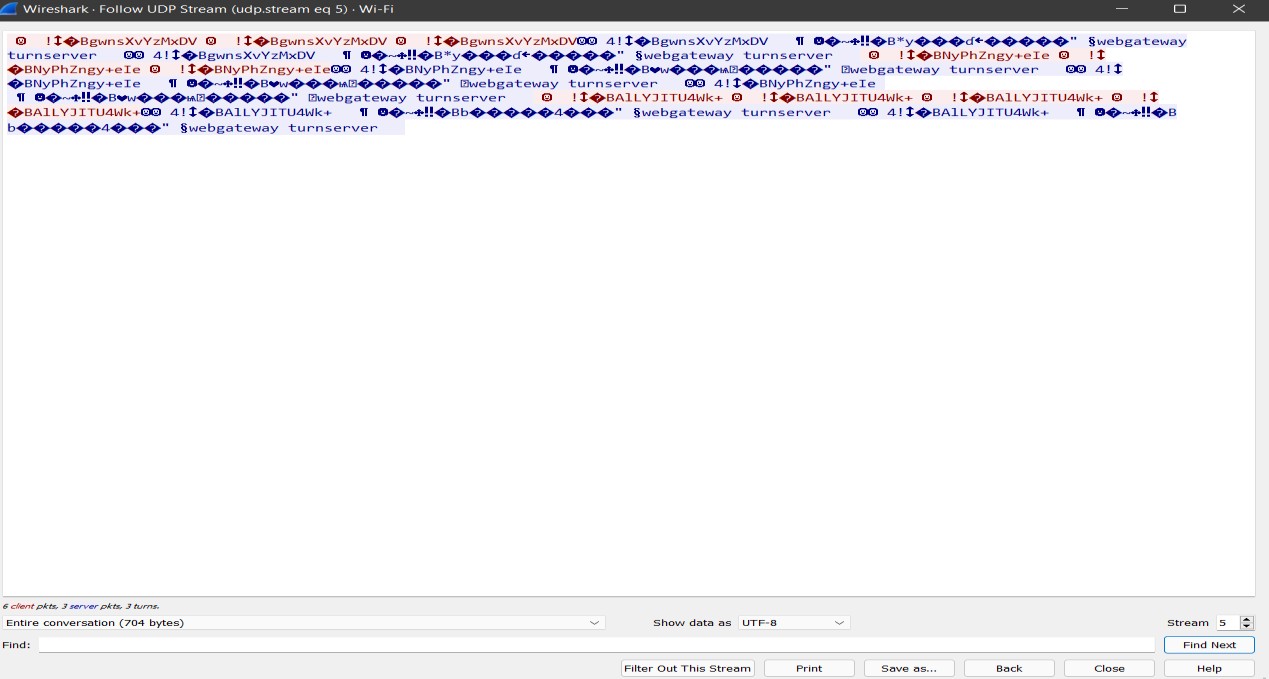


3. STUN: (through UDP Stream)

1. ASCII:



1. UTF-8:



**EXPERIMENT - 3**

**Write a program to create a server that listens to port 5003 using stream sockets. Write a simple client program to connect to the server. Send a simple text message “Hello” from the client to the server and the server to the client and close the connection.**

**Server code :**

print("This is experiment 3rd server code.")

print("Name: JASWANTH \nRoll No: 122010325012 \nBranch: CSE-CS")

import socket

IP = socket.gethostbyname(socket.gethostname())

PORT = 5004

ADDR = (IP, PORT)

SIZE = 1024

FORMAT = "utf-8"

def main():

    print("[STARTING] Server is starting.")

    server = socket.socket(socket.AF\_INET, socket.SOCK\_STREAM)

    server.bind(ADDR)

    server.listen(2)

    print("[LISTENING] Server is listening.")

    while True:

        conn, addr = server.accept()

        print(f"[NEW CONNECTION] {addr} connected.")

        while True:

            msg = conn.recv(SIZE).decode(FORMAT)

            print(msg)

            if msg == "Bye" or msg == "bye" :

                conn.close()

                print(f"[DISCONNECTED] {addr} disconnected.")

                break

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

    main()

**client code:**

print("This is experiment 3rd client code.")

print("Name: JASWANTH \nRoll No: 122010325012 \nBranch: CSE-CS")

import socket

IP = socket.gethostbyname(socket.gethostname())

PORT = 5004

ADDR = (IP, PORT)

FORMAT = "utf-8"

SIZE = 1024

def main():

    client = socket.socket(socket.AF\_INET, socket.SOCK\_STREAM)

    client.connect(ADDR)

    while True:

        msg = input("\nEnter the msg: ")

        client.send(msg.encode(FORMAT))

        if msg == "Bye" or msg == "bye" :

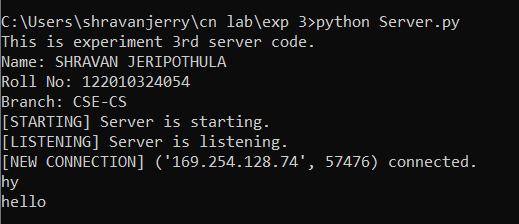
            break

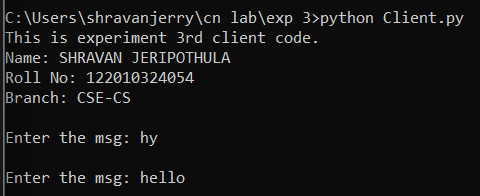
    client.close()

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

    main()

**OUTPUT:**

****

****

**EXPERIMENT - 4**

**Write a program to create a chat server that listens to port 5004 using stream sockets. Write a simple client program to connect to the server. Send multiple text messages from the client to the server and vice versa. When either party types “Bye”, close the connection.**

**Server code :**

print("This is experiment 4th server code.")

print("Name: JASWANTH \nRoll No: 122010325012 \nBranch: CSE-CS")

import socket

IP = socket.gethostbyname(socket.gethostname())

PORT = 5005

ADDR = (IP, PORT)

SIZE = 2048

FORMAT = "utf-8"

def main():

    print("[STARTING] Server is starting.")

    server = socket.socket(socket.AF\_INET, socket.SOCK\_STREAM)

    server.bind(ADDR)

    server.listen()

    print("[LISTENING] Server is listening.")

    while True:

        conn, addr = server.accept()

        print(f"[NEW CONNECTION] {addr} connected.")

        filename = conn.recv(SIZE).decode(FORMAT)

        print(f"[RECV] Receiving the filename.")

        file = open(filename, "w")

        conn.send("Filename received.".encode(FORMAT))

        data = conn.recv(SIZE).decode(FORMAT)

        print(f"[RECV] Receiving the file data.")

        file.write(data)

        conn.send("File data received".encode(FORMAT))

        file.close()

        conn.close()

        print(f"[DISCONNECTED] {addr} disconnected.")

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

    main()

**client code:**

print("This is experiment 4th client code.")

print("Name: JASWANTH \nRoll No: 122010325012 \nBranch: CSE-CS")

import socket

IP = socket.gethostbyname(socket.gethostname())

PORT = 5005

ADDR = (IP, PORT)

FORMAT = "utf-8"

SIZE = 2048

def main():

    client = socket.socket(socket.AF\_INET, socket.SOCK\_STREAM)

    client.connect(ADDR)

    file = open("data/textfile.txt", "r")

    data = file.read()

    client.send("yt.txt".encode(FORMAT))

    msg = client.recv(SIZE).decode(FORMAT)

    print(f"[SERVER]: {msg}")

    client.send(data.encode(FORMAT))

    msg = client.recv(SIZE).decode(FORMAT)

    print(f"[SERVER]: {msg}")

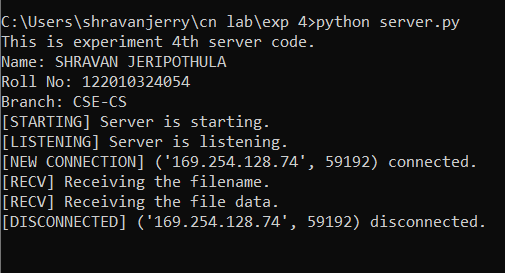
    file.close()

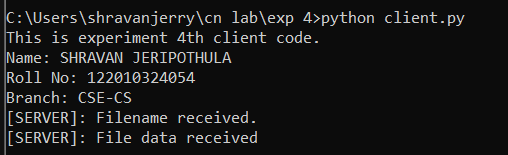
    client.close()

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

    main()

**OUTPUT:**

****

****

**EXPERIMENT - 5**

**Write a program to create a server that listens to port 5005 using stream sockets. Write a simple client program to connect to the server. The client should request for a text file and the server should return the file before terminating the connection.**

**Server code :**

print("This is experiment 5th server code.")

print("Name: JASWANTH \nRoll No: 122010325012 \nBranch: CSE-CS")

import socket

def Main():

    host = "127.0.0.1"

    port = 5000

    mySocket = socket.socket()

    mySocket.bind((host, port))

    mySocket.listen(1)

    conn, addr = mySocket.accept()

    print("Connection from: " + str(addr))

    while True:

        data = conn.recv(1024).decode()

        if not data:

            break

        print("from connected  user: " + str(data))

        data = str(data).upper()

        print("sending: " + str(data))

        conn.send(data.encode())

    conn.close()

if \_\_name\_\_ == '\_\_main\_\_':

    Main()

**client code:**

print("This is experiment 5th client code.")

print("Name: JASWANTH \nRoll No: 122010325012 \nBranch: CSE-CS")

import socket

def Main():

    host = '127.0.0.1'

    port = 5000

    mySocket = socket.socket()

    mySocket.connect((host, port))

    message = input(" -> ")

    while message != 'q':

        mySocket.send(message.encode())

        data = mySocket.recv(1024).decode()

        print('Received from server: ' + data)

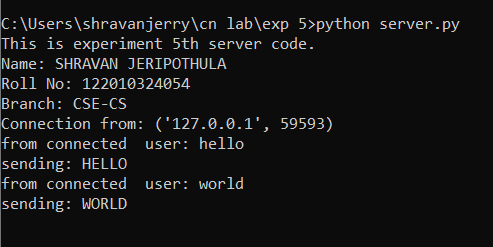
        message = input(" -> ")

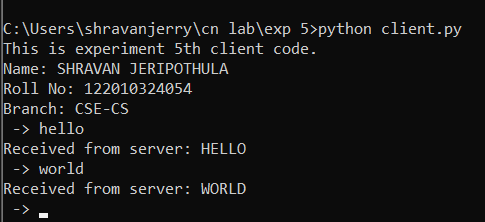
    mySocket.close()

if \_\_name\_\_ == '\_\_main\_\_':

    Main()

**OUTPUT:**



****

**EXPERIMENT - 6**

**Write a program to create a server that listens to port 5006 using stream sockets. Write a simple client program to connect to the server. Run multiple clients that request the server for binary files. The server should service each client one after the other before terminating the connection.**

**Server code :**

print("This is experiment 6th server code.")

print("Name: JASWANTH \nRoll No: 122010325012 \nBranch: CSE-CS")

import socket

server=socket.socket()

server.bind(( "localhost" ,5006))

server.listen(3)

print( "Server Activated" )

while True:

    client, address = server.accept()

    print(str(address)+ "is connected.")

    client\_request = client.recv(1024)

    print( 'Server received' ,client\_request)

    filename=client\_request

    f=open(filename,'rb')

    l=f.read(1024)

    client.send(l)

    f.close()

    print( "File Sent to "+ str(address)+ "server !")

server.close()

**client1 code:**

print("This is experiment 6th client1 code.")

print("Name: JASWANTH \nRoll No: 122010325012 \nBranch: CSE-CS")

import socket

client=socket.socket()

source=input( "Enter desired file: " )

client.connect(( "localhost" ,5006))

client.send(source.encode())

n=input( "Enter new file name: " )

with open(n, "wb") as f:

   print( 'File opened' )

   while True:

     data=client.recv(1024)

     if not data:

      break

     print( 'Receiving data...' )

     print('Data in New file:')

     print(data)

     print( "file recieved by client01" )

     print( "Connection Lost" )

     f.close()

client.close()

**client2 code:**

print("This is experiment 6th client2 code.")

print("Name: JASWANTH \nRoll No: 122010325012 \nBranch: CSE-CS")

import socket

client=socket.socket()

source=input( "Enter desired file: " )

client.connect(( "localhost" ,5006))

client.send(source.encode())

n=input( "Enter new file name: " )

with open(n, "wb") as f:

   print( 'File opened' )

   while True:

     data=client.recv(1024)

     if not data:

      break

     print( 'Receiving data...' )

     print('Data in New file:')

     print(data)

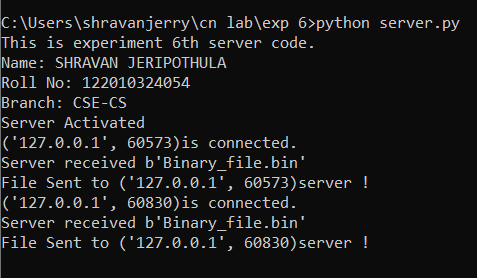
     print( "file recieved by client01" )

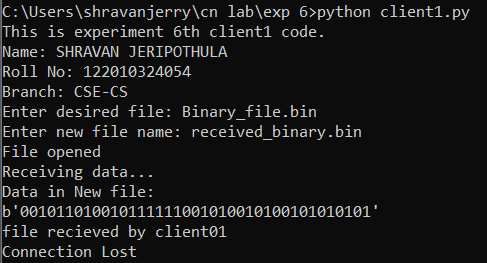
     print( "Connection Lost" )

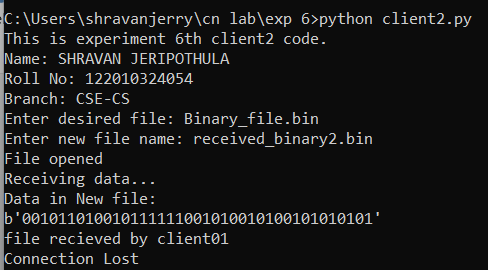
     f.close()

client.close()

**OUTPUT:**

****

****

****

**EXPERIMENT - 7**

**Write a program to create a server that listens to port 5007 using stream sockets. Write a simple client program to connect to the server. Run multiple clients that request the server for text files. The server should service all clients concurrently.**

**Server code :**

print("This is experiment 7th server code.")

print("Name: JASWANTH \nRoll No: 122010325012 \nBranch: CSE-CS")

import socket

port=5007

s=socket.socket()

host=socket.gethostname()

s.bind((host,port))

s.listen(5)

print('Server is listening....')

while True:

    conn,addr=s.accept()

    print('Got connection from',addr)

    data=conn.recv(1024)

    print('Server received',data)

    filename=data

    f=open(filename,'rb')

    l=f.read(1024)

    while(l):

        conn.send(l)

        print('Sent',repr(l))

        l=f.read(1024)

        f.close

    print('Done sending' )

    conn.close()

**client1 code:**

print("This is experiment 7th client1 code.")

print("Name: JASWANTH \nRoll No: 122010325012 \nBranch: CSE-CS")

import socket

s=socket.socket()

host=socket.gethostname()

port=5007

s.connect((host,port))

s.send('First\_File.txt'.encode())

with open('recieved\_file.txt','wb')as f:

    print('file opened')

    while True:

        data=s.recv(1024).decode()

        if not data:

            break

        print('recieving data...')

        print('data in recieved file:')

        print(data)

        f.write(data.encode())

        f.close()

        print('Successfully got the second file')

s.close()

print('connection closed')

**client2 code:**

print("This is experiment 7th client2 code.")

print("Name: JASWANTH \nRoll No: 122010325012 \nBranch: CSE-CS")

import socket

s=socket.socket()

host=socket.gethostname()

port=5007

s.connect((host,port))

s.send('Second\_File.txt'.encode())

with open('recieved\_file.txt','wb')as f:

    print('file opened')

    while True:

        data=s.recv(1024).decode()

        if not data:

            break

        print('recieving data...')

        print('data in recieved file:')

        print(data)

        f.write(data.encode())

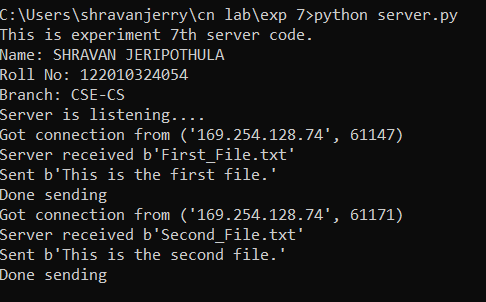
        f.close()

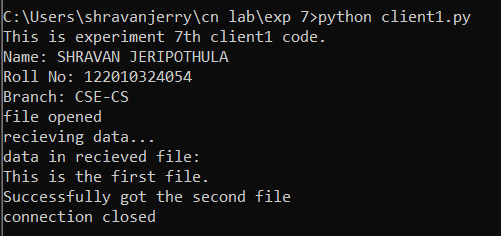
        print('Successfully got the second file')

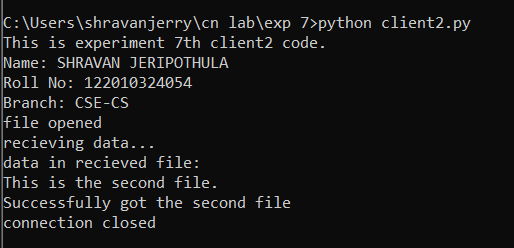
s.close()

print('connection closed')

**OUTPUT:**

****

****

****

**EXPERIMENT - 8**

**Write a program to create a server that listens to port 5009 using datagram sockets. Write a simple client program that requests the server for a binary file. The server should service multiple clients concurrently and send the requested files in response.**

**Server code :**

print("This is experiment 8 server code.")

print("Name: JASWANTH \nRoll No: 122010325012 \nBranch: CSE-CS")

from socket import \*

host= '127.0.0.1'

port=5009

server\_soc= socket(AF\_INET, SOCK\_DGRAM)

server\_soc.bind((host, port))

print( 'Server is ready to listen' )

n=int(input('Enter number of clients: '))

for i in range(n):

 file\_name, client\_adrs= server\_soc.recvfrom(1024)

 print('Client',i+1,'- ', file\_name.decode())

 try:

    f=open(file\_name.decode(),'rb')

    l=f.read()

    server\_soc.sendto(l, client\_adrs)

    print('The file has been sent to client',i+1)

    f.close()

 except:

    msg='The file is not present'

    server\_soc.sendto(msg.encode(), client\_adrs)

**client1 code:**

print("This is experiment 8th

client1 code.")

print("Name: JASWANTH \nRoll No: 122010325012 \nBranch: CSE-CS")

from socket import \*

host= '127.0.0.1'

port= 5009

client\_soc= socket(AF\_INET, SOCK\_DGRAM)

file\_name= input('Enter file name required: ')

client\_soc.sendto(file\_name.encode(),(host, port))

file, server\_adrs= client\_soc.recvfrom(1024)

**client2 code:**

print("This is experiment 8th client2 code.")

print("Name: JASWANTH \nRoll No: 122010325012 \nBranch: CSE-CS")

from socket import \*

host= '127.0.0.1'

port= 5009

client\_soc= socket(AF\_INET, SOCK\_DGRAM)

file\_name= input('Enter file name required: ')

client\_soc.sendto(file\_name.encode(),(host, port))

file, server\_adrs= client\_soc.recvfrom(1024)

**client3 code:**

print("This is experiment 8th client3 code.")

print("Name: JASWANTH \nRoll No: 122010325012 \nBranch: CSE-CS")

from socket import \*

host= '127.0.0.1'

port= 5009

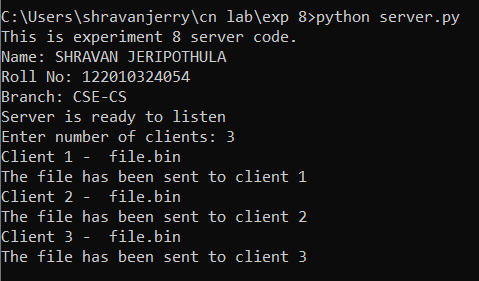
client\_soc= socket(AF\_INET, SOCK\_DGRAM)

file\_name= input('Enter file name required: ')

client\_soc.sendto(file\_name.encode(),(host, port))

file, server\_adrs= client\_soc.recvfrom(1024)

**OUTPUT:**

****

**THANK YOU**