TITLE:

NAME:SAURABH MUKHERJEE

CLASS:BCSE 3

ROLL NO:001910501006

GROUP:A1

ASSIGNMENT NUMBER:2

SUBJECT: COMPUTER NETWORKING LAB

PROBLEM STATEMENT:

Implement three data link layer protocols, Stop and Wait, Go Back N Sliding Window and Selective Repeat Sliding Window for flow control. Several functions may be required which can be designed as and when required.

DESCRIPTION:

Stop-N-Wait Protocol:

Stop and wait means, whatever the data that sender wants to send, he sends the data to the receiver. After sending the data, he stops and waits until he receives the acknowledgment from the receiver. The stop and wait protocol is a flow control protocol where flow control is one of the services of the data link layer.It is a data-link layer protocol which is used for transmitting the data over the noiseless channels. It provides unidirectional data transmission which means that either sending or receiving of data will take place at a time. It provides flow-control mechanism but does not provide any error control mechanism.The idea behind the usage of this frame is that when the sender sends the frame then he waits for the acknowledgment before sending the next frame.

Go-Back-N Protocol:

GoBackN protocol is based on the window size.We have a predetermined window size .Once the window size is full , the system waits for acknowledgement and if the acknowledgement is not received than all the packets of the current window is sent again.In this way we reduce the time taken by the simple StopN Wait protocol .In Go-Back-N ARQ, **N** is the sender's window size. Suppose we say that Go-Back-3, which means that the three frames can be sent at a time before expecting the acknowledgment from the receiver.It uses the principle of protocol pipelining in which the multiple frames can be sent before receiving the acknowledgment of the first frame. If we have five frames and the concept is Go-Back-3, which means that the three frames can be sent, i.e., frame no 1, frame no 2, frame no 3 can be sent before expecting the acknowledgment of frame no 1.

In Go-Back-N ARQ, the frames are numbered sequentially as Go-Back-N ARQ sends the multiple frames at a time that requires the numbering approach to distinguish the frame from another frame, and these numbers are known as the sequential numbers.

Selective Repeat Protocol:

This protocol has the added advantage over the go back n that not all the packets are sent again for the current window rather those packets are sent for which the acknowledgement has not yet been received.

Selective Repeat protocol provides for sending multiple frames depending upon the availability of frames in the sending window, even if it does not receive acknowledgement for any frame in the interim. The maximum number of frames that can be sent depends upon the size of the sending window.

The receiver records the sequence number of the earliest incorrect or un-received frame. It then fills the receiving window with the subsequent frames that it has received. It sends the sequence number of the missing frame along with every acknowledgement frame.

The sender continues to send frames that are in its sending window. Once, it has sent all the frames in the window, it retransmits the frame whose sequence number is given by the acknowledgements. It then continues sending the other frames.

DESIGN:

The system has three major components :

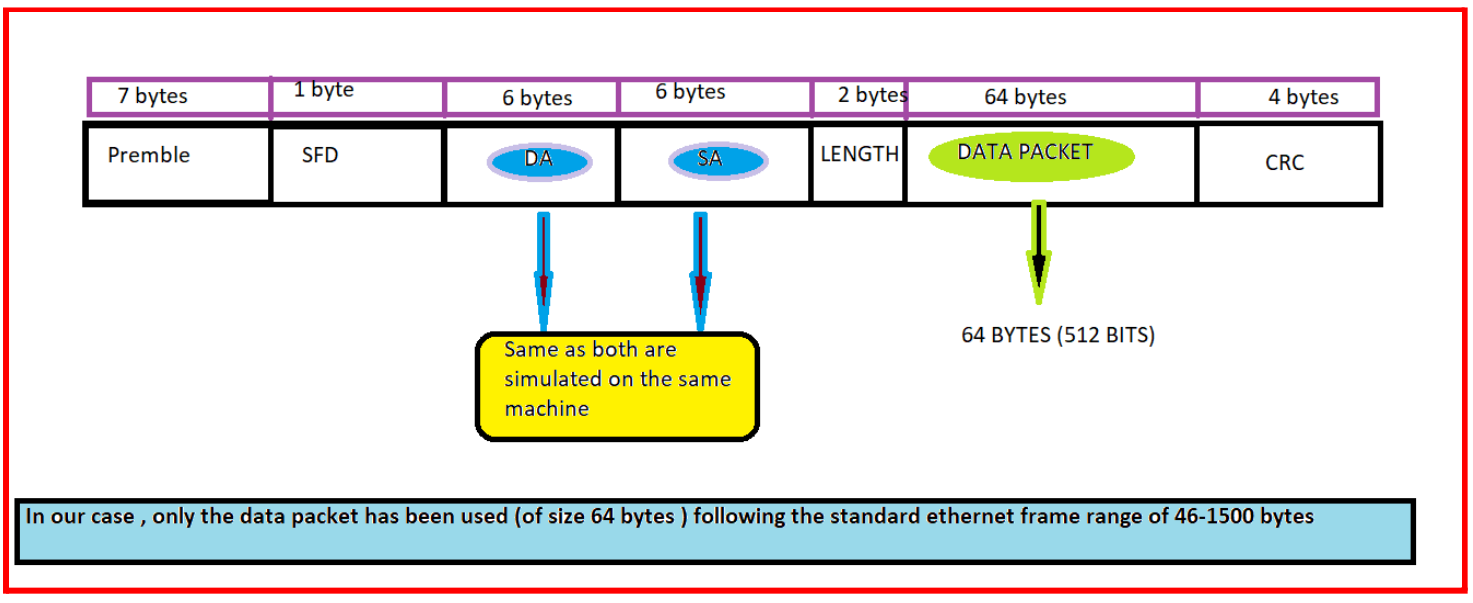
1.Sender (Client in our case)

2.Channel (Implemented using the injectError() methods)

3.Receiver (Server in our case)

All the three components has been designed using sockets and are interconnected for data transmission.The socket is first sending the date to the Channel.To simulate the real world scenario, we must have some delay between the sending time of the packets and the receipt of the acknowledgement from the receiver side.Since ,in our case the server as well as the client are on the same machines , so a delay() method is used on the channel which is inserting some random delay on the sending of data packets which takes it closer to the real world scenario.

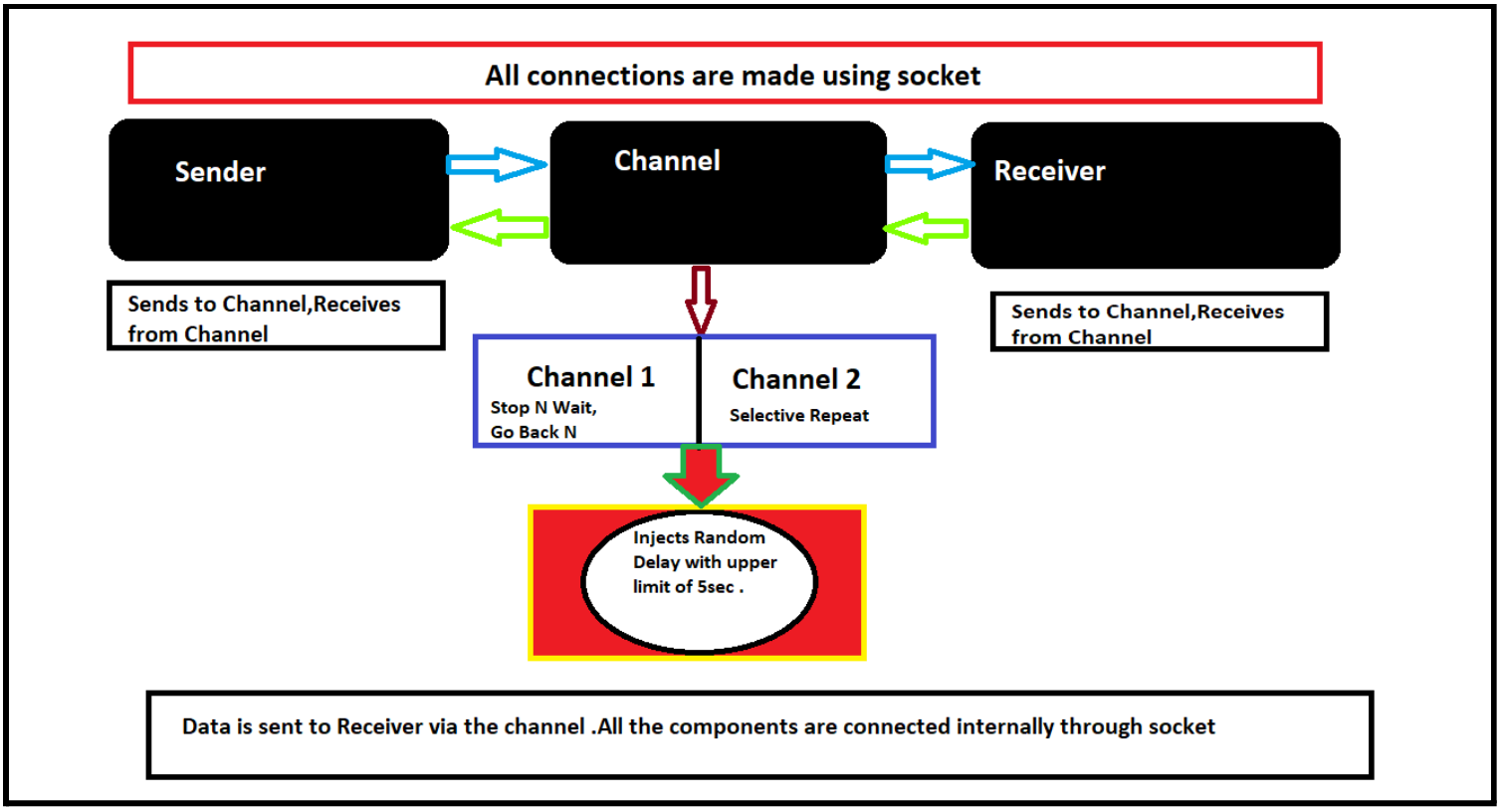
Frame Format Used

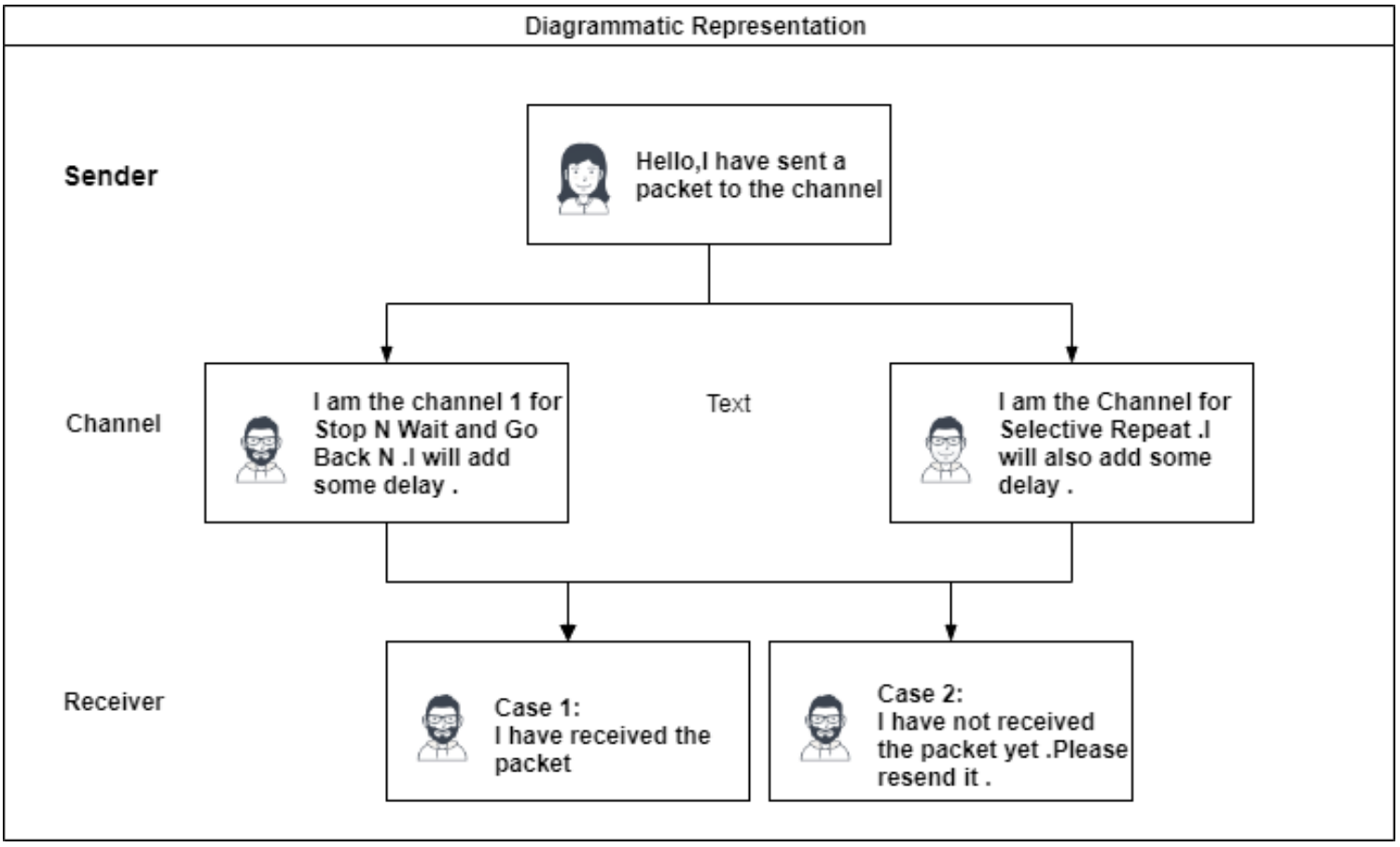


```
def StopNWait(data,count_of_packets):
    msg={} #now instead of this msg we have to send the data packets one by one

    counter,k=0,0
    round_trip_time={}
    while counter<count_of_packets:

        msg={"data":data[k:k+512]}
        s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
        s.connect((socket.gethostname(), 4000))
        currtime=datetime.datetime.now()
        s.sendall(pickle.dumps(msg))

        msg=s.recv(100)
        my_dict=pickle.loads(msg)
        if my_dict["ack"]==1:
            print("Acknowledgement for packet ",counter+1," is received..Sending next")
            round_trip_time[counter+1]=(datetime.datetime.now()-currtime).total_seconds()
            print("Round trip time in seconds:",(datetime.datetime.now()-currtime).total_seconds())
            counter+=1
            k+=512
        else:
            print("Negative Ack, Resending the packet :",counter+1)

        s.close()
```

```
import pickle,socket,datetime
def GoBackN(data,count_of_packets):
    # GO BACK N PROTOCOL
    number_of_frames=count_of_packets

    # frame_size=4
    frames_list=[]

    counter=0
    current_size,k=0,0
    while counter<number_of_frames :

        while current_size<4:
            frames_list.append(data[k:k+512])
            k+=512
            current_size+=1


        msg={"data":frames_list}
        s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
        s.connect((socket.gethostname(), 4000))
        currtime=datetime.datetime.now()
        s.sendall(pickle.dumps(msg))

        msg=s.recv(100)
        my_dict=pickle.loads(msg)
        ack_num=my_dict["ack"]

        pop_front_count=0
        if ack_num!=-1:
            while pop_front_count<ack_num:
                frames_list.pop(0)
                current_size-=1
                pop_front_count+=1


        if ack_num!=-1:
            print("Acknowledgement for frames till ",ack_num," is received..Sending next")
            print("Round trip time in seconds:",(datetime.datetime.now()-currtime).total_seconds())
            counter+=(ack_num+1)
        else:
            print("Ack Received:", ack_num ,"Frame Corrupted , Negative ack , resending the frames for this window :")
            k=(k-512*4)

        s.close()
```

```
import pickle,socket,datetime

def SelectiveRepeat(data,number_of_frames):
    # frame_size=4
    counter,k=0,0
    current_size=0
    frames_list=[]

    round_trip_time={}
    while True:
        while current_size<4 and k<len(data):
            frames_list.append({k//512:data[k:k+512]})
            k+=512
            current_size+=1

        if current_size==0:break

        msg={"data":frames_list}
        # msg=frames_list
        s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
        s.connect((socket.gethostname(), 4000))
        currtime=datetime.datetime.now()
        s.sendall(pickle.dumps(msg))


        msg=s.recv(4096)
        my_set=pickle.loads(msg)#set containing corrupted packets or negative acknowledgement here
        my_list=[]
        # here in my list we collect the keys present in the frame at this moment
        for i in range(len(frames_list)):
            dict=frames_list[i]
            for x in dict.keys():
                my_list.append(x)

        for x in my_list:
            if x not in my_set:
                print("Acknowledgement for frame ",x," has been received ..")
                round_trip_time[x]=(datetime.datetime.now()-currtime).total_seconds()
                print("Round trip time is " ,round_trip_time[x])
                counter+=1
            else:
                print("Negative Acknowledgement for frame sequence ", x ,"  received..needs to retransmit"
)

        newFramesList=[]
        for i in range(len(my_list)):
            if my_list[i] in my_set:
                newFramesList.append(frames_list[i])


        frames_list=newFramesList
        current_size=len(frames_list)

        s.close()

    print(round_trip_time)
```

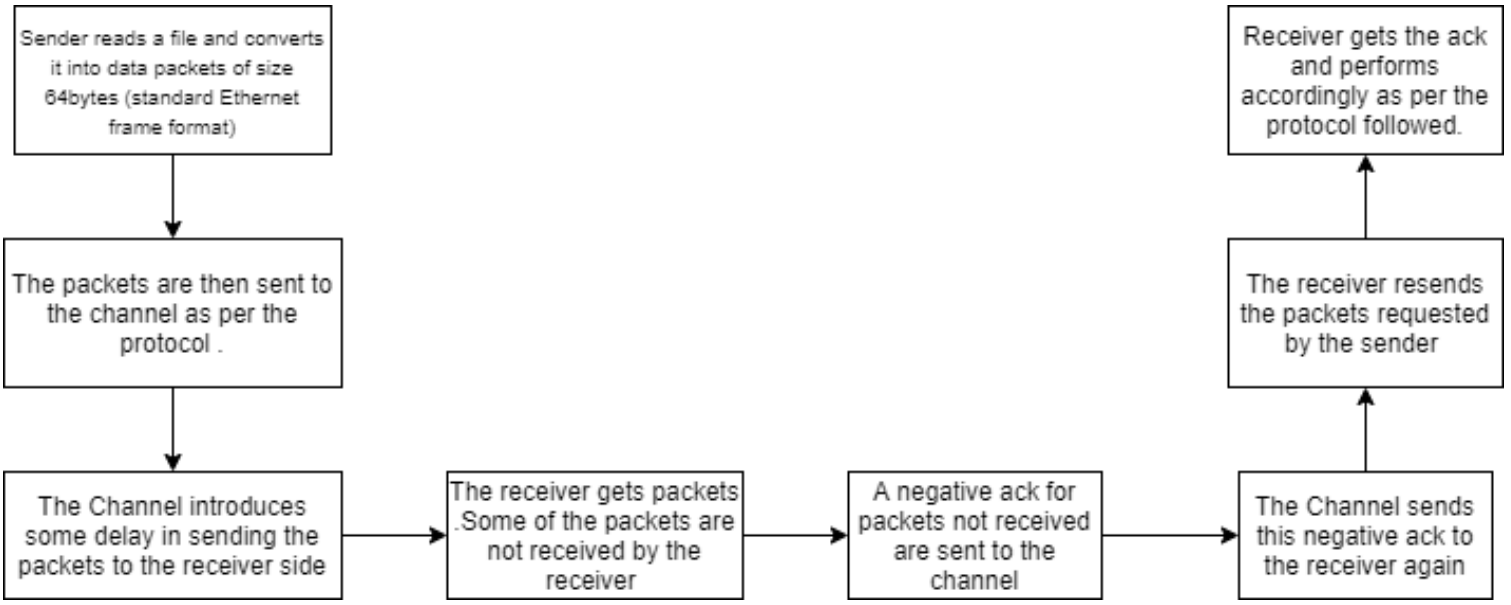




The sender is sending the packets to the channel following the protocol.In case of StopNWait and Go Back N protocols , we are using channel 1 to serve the purpose , but in the case of Selective Repeat Protocol , we have channel 2 . The channel has some methods for causing a delay in sending the packets to the receiver side.The channel then waits for the receiver to send the acknowledgement which it then sends to the sender .Instead of timeout , we have used the concept for flags to indicate the non-receipt of some packets or corrupted or damaged packets which needs to be resent by the sender to the receiver .The decision for packets

acknowledgement or the damage of packets is decided randomly by the receiver , with a probability of about 50 percent depending on the random function .

Flow of the packets from sender to receiver via Channel



For simulating the concept of timeout , the upper limit of the sleep() method has been fixed at 5 seconds , so that if the packet is received within that time , it is taken to be successfully sent depending on the acknowledgement received , else it is resent by the sender .It is ensured that the maximum time the packet would take to get acknowledged is this time and after that it is decided depending on the acknowledgement .

INPUT DATA

The data packet used for the test cases in our case :-

Hello , this is the test packet for the assignment in which is to be converted into frames of size 512 bits which is the standard ethernet packet size .Then that packet must be sent to the channel in which error is to be injected which is further sent to the server .The protocols used are Stop N Wait , Go Back N and Selective Repeat Protocol .These protocols are to be used for sending the data packets and according to the various algorithms we need to ensure the resending of the unacknowledged packet..

The input data is a text file which is taken as a command line input and then it is converted into binary data which is now treated as the data packet .This text file is converted into binary stream of data which is to be segregated into packet sizes of 64 bytes each following the standard ethernet frame format. The data packet length is first made a multiple of the frame size by appending proper number of bits are appended to the data packet so that it becomes a multiple of the standard data packet. Then the data is sent over the network , which in our case is the socket connection .

Note: All the functions for the sender , channel and receiver for the various protocols have been accumulated under a package name "saum" and the user is only able to see an abstraction of it .

IMPLEMENTATION

1.CLIENT SYSTEM (SENDER)

The data packets prepared are sent to the channel.The Channel then introduces some delay and then the data packet is sent to the receiver from the channel side.This whole process is using socket programming under the hood and connection is socket connection from Sender-Channel and Channel-Receiver for the data transmission purpose.

```
# divide the data into packets of size 512 bits each,the standard ethernet
packet_size=512
temp = 0
if (len(data) % packet_size != 0):
    temp = packet_size - (len(data) % packet_size)
data += ('0' * temp)
count_of_packets = len(data) // packet_size

if choice==1:
    StopNWait(data,count_of_packets)
elif choice==2:
    GoBackN(data,count_of_packets)
else:
    SelectiveRepeat(data,count_of_packets)
```

2.CHANNEL SYSTEM

The main purpose of the channel in real life is that it acts as the medium between the sender and receiver through which all the packets must pass.The channel may introduce some error and cause a delay in the transmission of data.The channel introduces some random delay by calling the sleep() method and then after waking up the data packet which is with the channel as sent by the sender is sent to the receiver.Thus in our case , channel servers the function of introducing delay in the data transmission mechanism.

```
# This channel method is to be used for Stop N Wait and Go Back N protocol
import socket,random,time
def Channel1():
    s=socket.socket(socket.AF_INET,socket.SOCK_STREAM)
    s.bind((socket.gethostname(),4000))
    s.listen(5)

    clientsocket,address=s.accept()
    print(f"Connection from {address} has been established.")

    recv_msg=clientsocket.recv(100)


    s2 = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    s2.connect((socket.gethostname(), 5000))

    # here we will make our channel sleep for few seconds selected randomly
    time.sleep(random.randint(0,5))
    s2.send(recv_msg)

    ackfromrec=s2.recv(100)#This is also in the form of a dictionary from the receiver
    clientsocket.send(ackfromrec) #sends the acknowl rec from the receiver to the sender a
gain
```

```
# This channel method is to be used for Selective Repeat Protocol
import socket
def Channel2():
    s=socket.socket(socket.AF_INET,socket.SOCK_STREAM)
    s.bind((socket.gethostname(),4000))
    s.listen(5)

    clientsocket,address=s.accept()
    print(f"Connection from {address} has been established.")


    recv_msg=clientsocket.recv(4096)

    s2 = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    s2.connect((socket.gethostname(), 5000))

    s2.send(recv_msg)

    ackfromrec=s2.recv(4096)#This is also in the form of a dictionary from the receiver
    clientsocket.send(ackfromrec) #sends the acknowl rec from the receiver to the sender a
gain
```

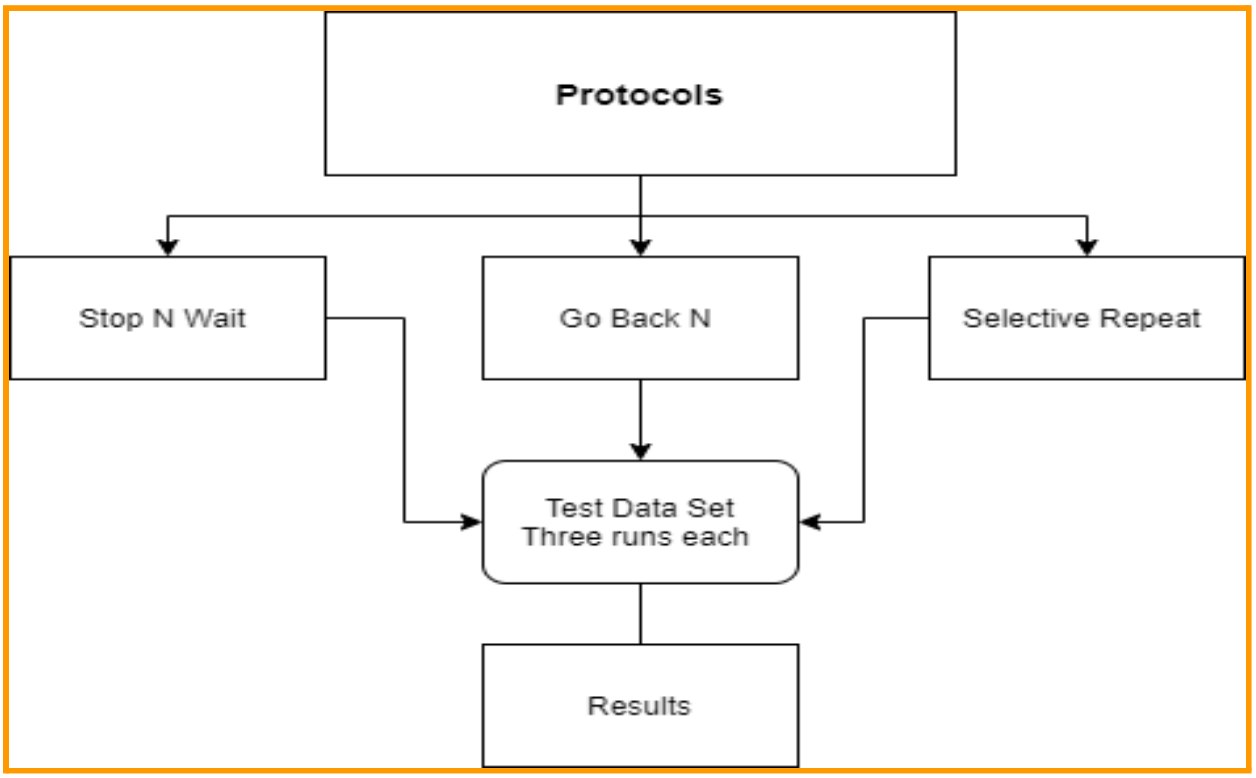
3.SERVER (RECEIVER)

The receiver receives the data and then prepares the acknowledgement randomly and then sends the acknowledgement to the sender via the channel depending on the protocol that is being followed.The positive and negative acknowledgement are decided at the runtime by the random function and as per the acknowledgement provided by the receiver , the receiver requests for the packet depending on it.

```
from saum.Receiver.receiverGoBackN import *


from saum.Receiver.receiverStopNWait import *
from saum.Receiver.receiverSelectiveRepeat import *

while True:
    # receiveStopNWait()
    receiverGoBackN()
    # receiverSelectiveRepeat()
```

RESULTS:

Preparation Strategy:



For a comparative study between the various protocols , following performance metrics have been used :

Performance Metrics Used:

1.Receiver Throughput

2.RTT

3.Bandwidth Delay Product

4.Utilization Percentage(Assuming the channel to be ideal), hence 100 in our case..

Case 1: Stop N Wait Protocol

Test Set 1 :

The test data is same but due to random injection of delay , the output is changing for every iteration

```
PS C:\Users\saura\Desktop\CS005\College\SEM5\ComputerNetworking\LAB\ASSIGN2\main> python driver.py saurabh.txt

-------User have the options for the following different protocols--------
1.STOP AND WAIT PROTOCOL
2.GO BACK N
3.SELECTIVE REPEAT PROTOCOL
Enter your choice among the above three options:1
Acknowledgement for packet  1  is received..Sending next
Round trip time in seconds: 0.046314
Acknowledgement for packet  2  is received..Sending next
Round trip time in seconds: 4.013733
Acknowledgement for packet  3  is received..Sending next
Round trip time in seconds: 2.015458
Negative Ack, Resending the packet : 4
Negative Ack, Resending the packet : 4
Negative Ack, Resending the packet : 4
Acknowledgement for packet  4  is received..Sending next
Round trip time in seconds: 0.00389
Negative Ack, Resending the packet : 5
Negative Ack, Resending the packet : 5
Negative Ack, Resending the packet : 5
Acknowledgement for packet  5  is received..Sending next
Round trip time in seconds: 4.011776
Negative Ack, Resending the packet : 6
Acknowledgement for packet  6  is received..Sending next
Round trip time in seconds: 1.004372
Acknowledgement for packet  7  is received..Sending next
Round trip time in seconds: 3.004918
PS C:\Users\saura\Desktop\CS005\College\SEM5\ComputerNetworking\LAB\ASSIGN2\main>
```

Test Set 2:

```
PS C:\Users\saura\Desktop\CS005\College\SEM5\ComputerNetworking\LAB\ASSIGN2\main>
-------User have the options for the following different protocols--------
1.STOP AND WAIT PROTOCOL
2.GO BACK N
3.SELECTIVE REPEAT PROTOCOL
Enter your choice among the above three options:1
Negative Ack, Resending the packet : 1
Acknowledgement for packet  1  is received..Sending next
Round trip time in seconds: 3.009899
Negative Ack, Resending the packet : 2
Negative Ack, Resending the packet : 2
Acknowledgement for packet  2  is received..Sending next
Round trip time in seconds: 5.007423
Acknowledgement for packet  3  is received..Sending next
Round trip time in seconds: 0.0
Acknowledgement for packet  4  is received..Sending next
Round trip time in seconds: 4.009865
Acknowledgement for packet  5  is received..Sending next
Round trip time in seconds: 3.010482
Acknowledgement for packet  6  is received..Sending next
Round trip time in seconds: 4.011791
Negative Ack, Resending the packet : 7
Acknowledgement for packet  7  is received..Sending next
Round trip time in seconds: 2.003655
```

Test Set 3:

```
PS C:\Users\saura\Desktop\CS005\College\SEM5\ComputerNetworking\LAB\ASSIGN2\main> python driver.py saurabh.txt

-------User have the options for the following different protocols--------
1.STOP AND WAIT PROTOCOL
2.GO BACK N
3.SELECTIVE REPEAT PROTOCOL
Enter your choice among the above three options:1
Negative Ack, Resending the packet : 1
Negative Ack, Resending the packet : 1
Acknowledgement for packet  1  is received..Sending next
Round trip time in seconds: 0.008218
Acknowledgement for packet  2  is received..Sending next
Round trip time in seconds: 1.003165
Acknowledgement for packet  3  is received..Sending next
Round trip time in seconds: 0.004002
Acknowledgement for packet  4  is received..Sending next
Round trip time in seconds: 4.01073
Negative Ack, Resending the packet : 5
Negative Ack, Resending the packet : 5
Negative Ack, Resending the packet : 5
Acknowledgement for packet  5  is received..Sending next
Round trip time in seconds: 1.016464
Acknowledgement for packet  6  is received..Sending next
Round trip time in seconds: 3.006839
Acknowledgement for packet  7  is received..Sending next
Round trip time in seconds: 4.012937
```

In our simulation , the RTT is dependent completely on the time for which the sleep method is called , which is purely decided at runtime.

All measurements are in seconds .

PACKET NUMBER	RTT(RUN 1)	RTT(RUN 2)	RTT(RUN 3)
1	0.046314	3.009899	0.008218
2	4.013733	5.007423	1.003165
3	2.015458	0.0000	0.004002
4	0.00389	4.009865	4.01073
5	4.011776	3.010982	1.016465
6	1.004372	4.011791	3.006839
7	3.004918	2.003655	4.012937
	Avg=2.0143	Avg=3.008	Avg=1.867

Average time taken for a packet for the successful acknowledgement in our case is approximately (over all runs)is 2.296 seconds.

Throughput (Packets/Time) = 0.4355

Throughput (Bits/second) = 222.996

Bandwidth-Delay Product = 511.99 bits or 64 bytes(approx)

Utilization percent = 99.99 (approx)

Case 2:GO-BACK-N Protocol

Test Set 1:

```
PS C:\Users\saura\Desktop\CS005\College\SEM5\ComputerNetworking\LAB\ASSIGN2\main> python driver.py saurabh.txt

-------User have the options for the following different protocols--------
1.STOP AND WAIT PROTOCOL
2.GO BACK N
3.SELECTIVE REPEAT PROTOCOL
Enter your choice among the above three options:2
Ack Received: -1 Frame Corrupted , Negative ack , resending the frames for this window :
Ack Received: -1 Frame Corrupted , Negative ack , resending the frames for this window :
Acknowledgement for frames till  3  is received..Sending next
Round trip time in seconds: 5.004452
Acknowledgement for frames till  1  is received..Sending next
Round trip time in seconds: 1.002889
Ack Received: -1 Frame Corrupted , Negative ack , resending the frames for this window :
Acknowledgement for frames till  1  is received..Sending next
Round trip time in seconds: 2.017847
PS C:\Users\saura\Desktop\CS005\College\SEM5\ComputerNetworking\LAB\ASSIGN2\main>
```

Test Set 2:

```
PS C:\Users\saura\Desktop\CS005\College\SEM5\ComputerNetworking\LAB\ASSIGN2\main>
-------User have the options for the following different protocols--------
1.STOP AND WAIT PROTOCOL
2.GO BACK N
3.SELECTIVE REPEAT PROTOCOL
Enter your choice among the above three options:2
Acknowledgement for frames till  0  is received..Sending next
Round trip time in seconds: 5.015597
Acknowledgement for frames till  2  is received..Sending next
Round trip time in seconds: 0.0
Ack Received: -1 Frame Corrupted , Negative ack , resending the frames for this window :
Acknowledgement for frames till  2  is received..Sending next
Round trip time in seconds: 2.010763
PS C:\Users\saura\Desktop\CS005\College\SEM5\ComputerNetworking\LAB\ASSIGN2\main>
```

Test Set 3:

```
PS C:\Users\saura\Desktop\CS005\College\SEM5\ComputerNetworking\LAB\ASSIGN2\main> python driver.py saurabh.txt

-------User have the options for the following different protocols--------
1.STOP AND WAIT PROTOCOL
2.GO BACK N
3.SELECTIVE REPEAT PROTOCOL
Enter your choice among the above three options:2
Acknowledgement for frames till  2  is received..Sending next
Round trip time in seconds: 3.022256
Acknowledgement for frames till  0  is received..Sending next
Round trip time in seconds: 1.017796
Ack Received: -1 Frame Corrupted , Negative ack , resending the frames for this window :
Acknowledgement for frames till  2  is received..Sending next
Round trip time in seconds: 0.0
```

PACKET NUMBER	RTT(RUN 1)	RTT(RUN 2)	RTT(RUN 3)
1	5.004452	5.015597	3.022256
2	5.004452	0.00	3.022256
3	5.004452	0.00	3.022256
4	5.004452	0.00	1.017796
5	1.002889	2.010763	0.00
6	2.017847	2.010763	0.00
7	2.017847	2.010763	0.00
	Avg=3.5798	Avg=1.5783	Avg=1.4406

Average time taken for all the packets to get successfully acknowledged (over all the runs) is approximately 2.1995

Throughput (Packets/Time) = 0.4546

Throughput (Bits/second) = 232.78

Bandwidth-Delay Product = 511.99 (512) bits or 64 bytes

Case 3:SELECTIVE REPEAT Protocol

Test Set 1:

```
-------User have the options for the following different protocols--------
1.STOP AND WAIT PROTOCOL
2.GO BACK N
3.SELECTIVE REPEAT PROTOCOL
Enter your choice among the above three options:3
Negative Acknowledgement for frame sequence  2   received..needs to retransmit
Negative Acknowledgement for frame sequence  3   received..needs to retransmit
Acknowledgement for frame  0  has been received ..
Acknowledgement for frame  2  has been received ..
Acknowledgement for frame  3  has been received ..
Negative Acknowledgement for frame sequence  4   received..needs to retransmit
Negative Acknowledgement for frame sequence  4   received..needs to retransmit
Negative Acknowledgement for frame sequence  5   received..needs to retransmit
Acknowledgement for frame  6  has been received ..
Acknowledgement for frame  7  has been received ..
Acknowledgement for frame  4  has been received ..
Acknowledgement for frame  5  has been received ..
{1: 0.0, 0: 0.0, 2: 0.0, 3: 0.0, 6: 0.0, 7: 0.0, 4: 0.008, 5: 0.008}
```

Test Set 2:

```
PS C:\Users\saura\Desktop\CS005\College\SEM5\ComputerNetworking\LAB\ASSIGN2\main> python driver.py saurabh.txt

-------User have the options for the following different protocols--------
1.STOP AND WAIT PROTOCOL
2.GO BACK N
3.SELECTIVE REPEAT PROTOCOL
Enter your choice among the above three options:3
Acknowledgement for frame  0  has been received ..
Negative Acknowledgement for frame sequence  1   received..needs to retransmit
Negative Acknowledgement for frame sequence  2   received..needs to retransmit
Negative Acknowledgement for frame sequence  3   received..needs to retransmit
Negative Acknowledgement for frame sequence  1   received..needs to retransmit
Negative Acknowledgement for frame sequence  2   received..needs to retransmit
Acknowledgement for frame  3  has been received ..
Acknowledgement for frame  4  has been received ..
Negative Acknowledgement for frame sequence  5   received..needs to retransmit
Negative Acknowledgement for frame sequence  6   received..needs to retransmit
Acknowledgement for frame  1  has been received ..
Negative Acknowledgement for frame sequence  2   received..needs to retransmit
Acknowledgement for frame  5  has been received ..
Negative Acknowledgement for frame sequence  6   received..needs to retransmit
Acknowledgement for frame  2  has been received ..
Acknowledgement for frame  6  has been received ..
Negative Acknowledgement for frame sequence  7   received..needs to retransmit
Negative Acknowledgement for frame sequence  7   received..needs to retransmit
Negative Acknowledgement for frame sequence  7   received..needs to retransmit
Acknowledgement for frame  7  has been received ..
{0: 0.0, 3: 0.0, 4: 0.0, 1: 0.008007, 5: 0.008007, 2: 0.0, 6: 0.0, 7: 0.0}
```

Test Set 3:

```
PS C:\Users\saura\Desktop\CS005\College\SEM5\ComputerNetworking\LAB\ASSIGN2\main> python driver.py saurabh.txt

-------User have the options for the following different protocols--------
1.STOP AND WAIT PROTOCOL
2.GO BACK N
3.SELECTIVE REPEAT PROTOCOL
Enter your choice among the above three options:3
Acknowledgement for frame  0  has been received ..
Negative Acknowledgement for frame sequence  1   received..needs to retransmit
Negative Acknowledgement for frame sequence  2   received..needs to retransmit
Negative Acknowledgement for frame sequence  3   received..needs to retransmit
Negative Acknowledgement for frame sequence  1   received..needs to retransmit
Negative Acknowledgement for frame sequence  2   received..needs to retransmit
Acknowledgement for frame  3  has been received ..
Acknowledgement for frame  4  has been received ..
Negative Acknowledgement for frame sequence  5   received..needs to retransmit
Negative Acknowledgement for frame sequence  6   received..needs to retransmit
Acknowledgement for frame  1  has been received ..
Negative Acknowledgement for frame sequence  2   received..needs to retransmit
Acknowledgement for frame  5  has been received ..
Negative Acknowledgement for frame sequence  6   received..needs to retransmit
Acknowledgement for frame  2  has been received ..
Acknowledgement for frame  6  has been received ..
Negative Acknowledgement for frame sequence  7   received..needs to retransmit
Negative Acknowledgement for frame sequence  7   received..needs to retransmit
Negative Acknowledgement for frame sequence  7   received..needs to retransmit
Acknowledgement for frame  7  has been received ..
{0: 0.0, 3: 0.0, 4: 0.0, 1: 0.008007, 5: 0.008007, 2: 0.0, 6: 0.0, 7: 0.0}
```

PACKET NUMBER	RTT(RUN 1)	RTT(RUN 2)	RTT(RUN 3)
0	0.00	0.00	0.00
1	0.00	0.008007	0.008007
2	0.00	0.00	0.00
3	0.00	0.00	0.00

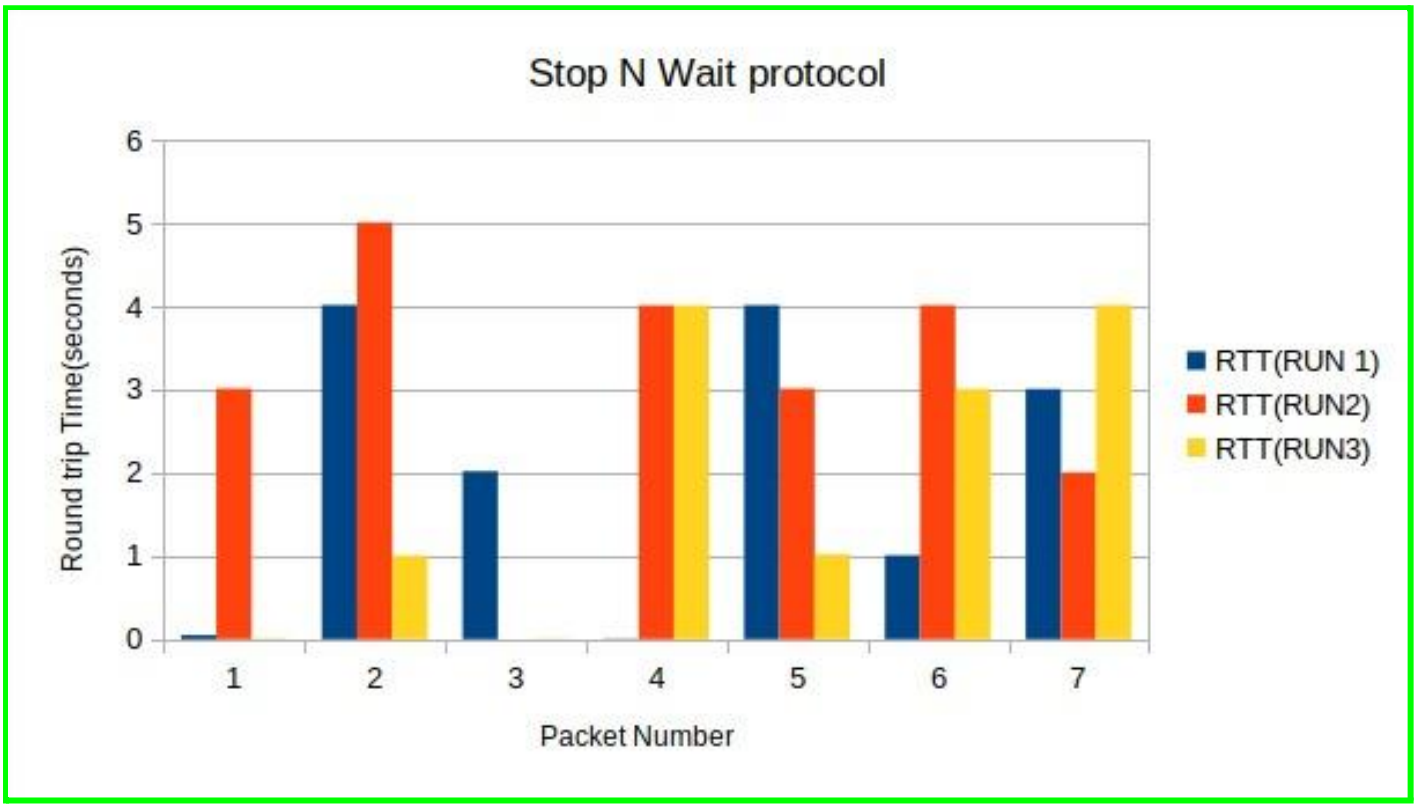
4	**0.008**	**0.00**	**0.00807**
5	**0.008**	**0.00807**	**0.00**
6	**0.00**	**0.00**	**0.00**
7	**0.00**	**0.00**	**0.00**
	Avg=0.001	**Avg=0.002**	**Avg=0.002**

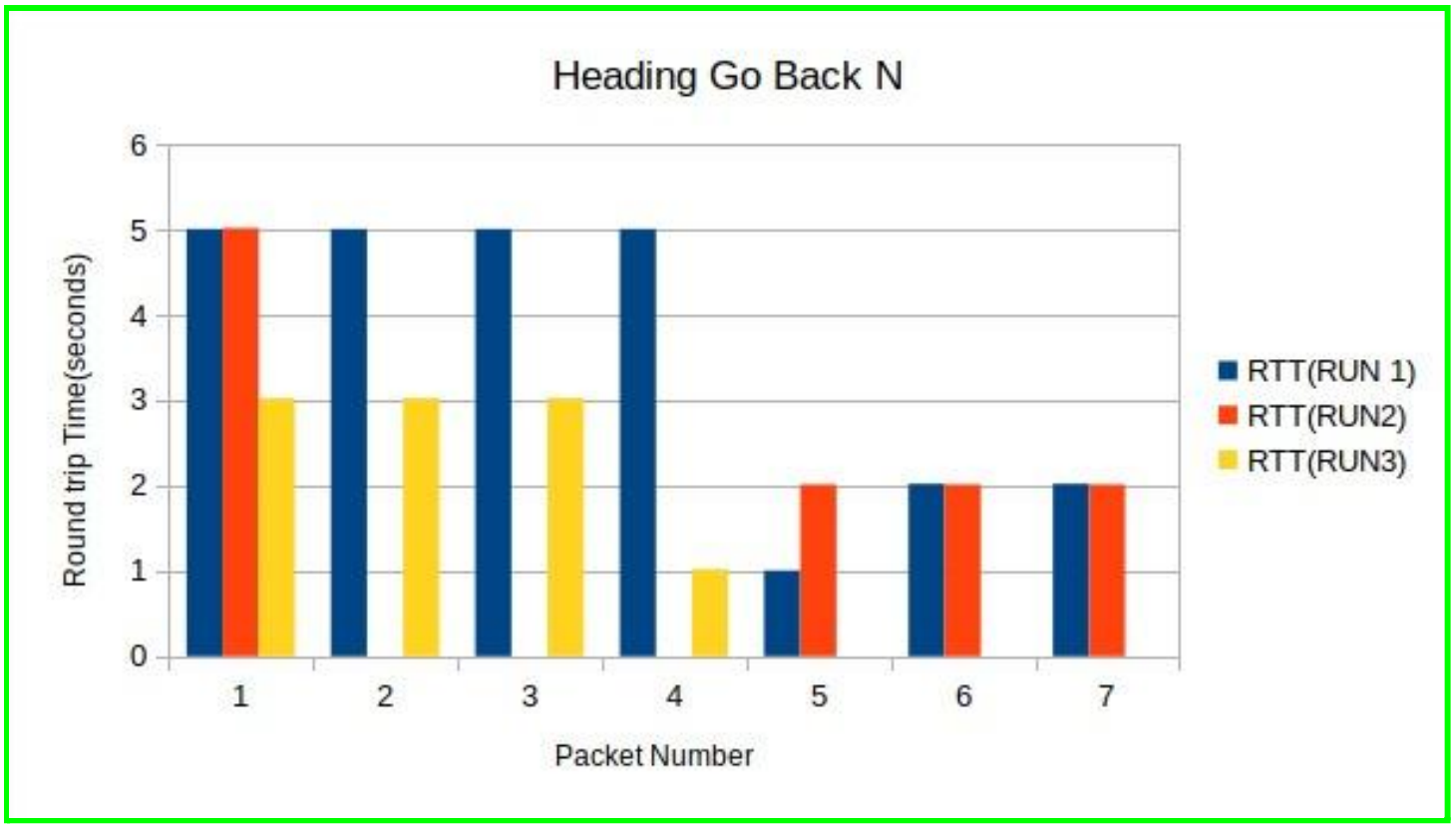
The average time taken for all the packets for getting acknowledged successfully by the receiver is approximately about 0.002 seconds.

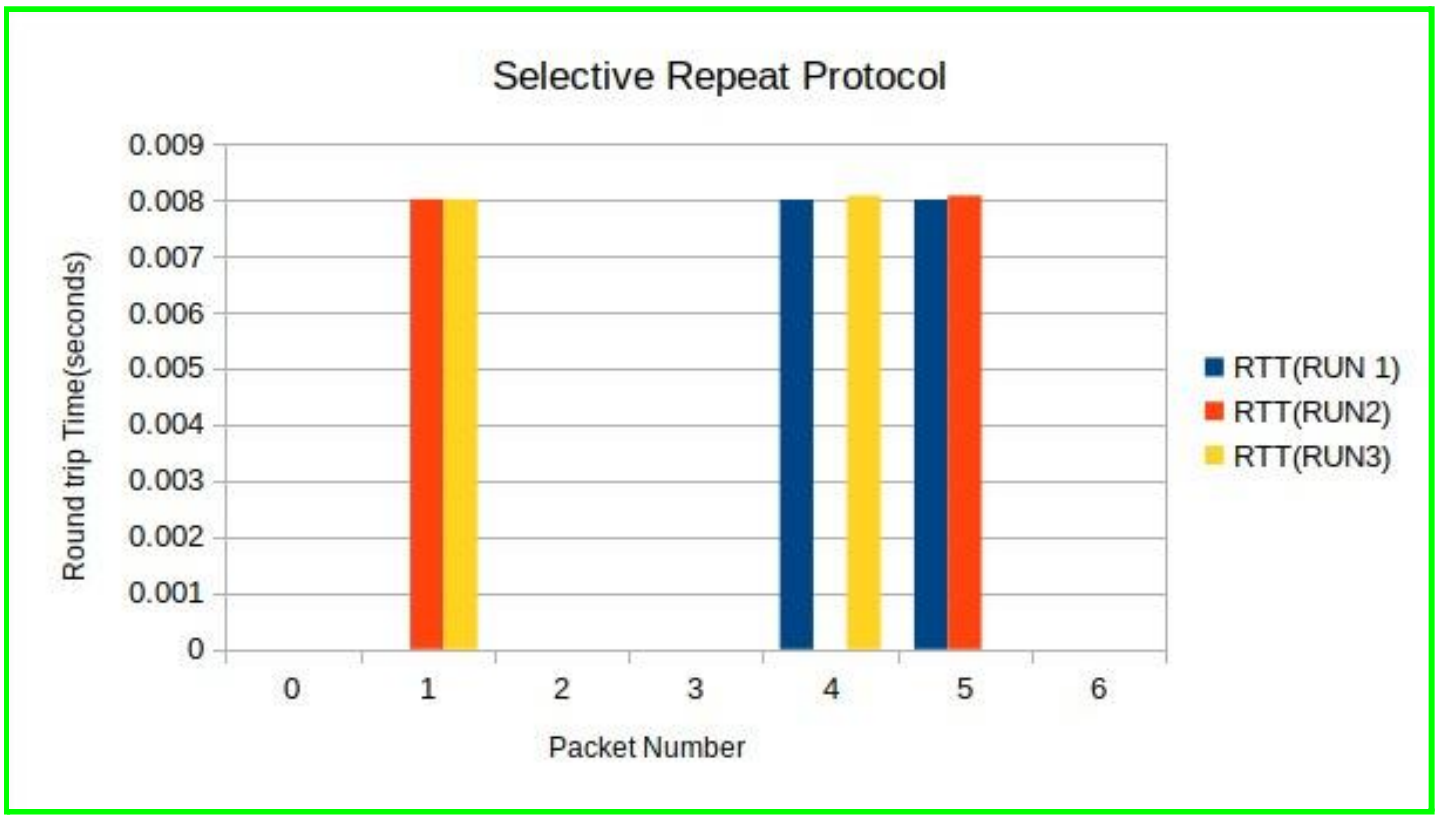
Throughput (Packets/Time) = 500

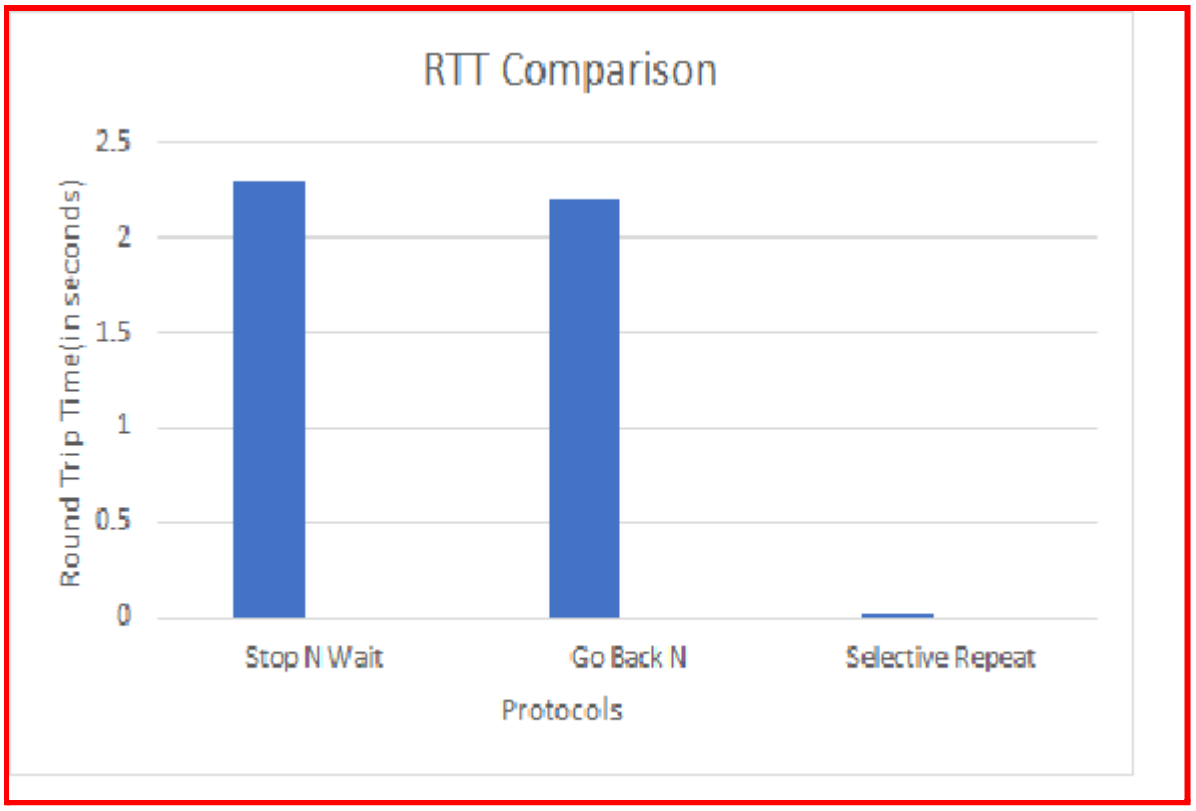
Throughput (Bits/second) = 256,000

Bandwidth-Delay Product = 511.99 (512) bits or 64 bytes

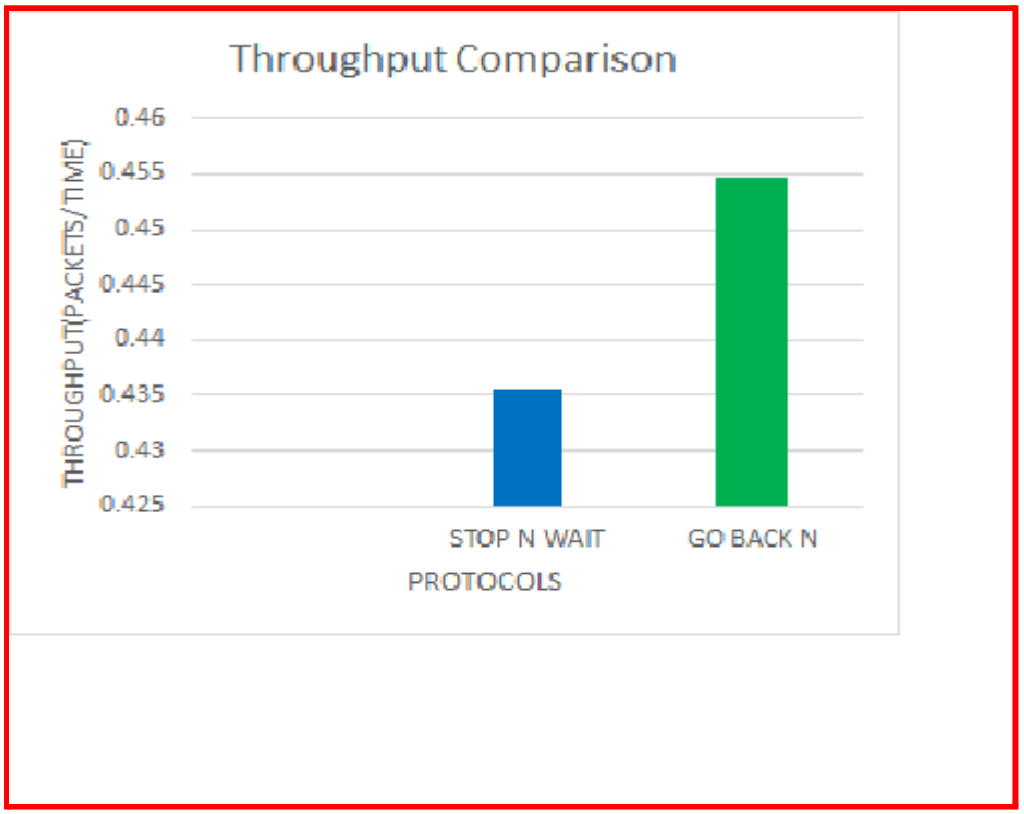



Heading Go Back N
Round trip Time(seconds)
0
1
2
3
4
5
6
1
2
3
4
5
6
7
Packet Number
RTT(RUN 1)
RTT(RUN2)
RTT(RUN3)


Selective Repeat Protocol
Round trip Time(seconds)
0
0.001
0.002
0.003
0.004
0.005
0.006
0.007
0.008
0.009
0
1
2
3
4
5
6
Packet Number
RTT(RUN 1)
RTT(RUN2)
RTT(RUN3)



The average Round-Trip-Time for all the packets over all the runs .



Throughput Comparison for Stop and Wait over others.The throughput for Selective Repeat in our case is much larger than the others ,that's why it is no shown in our case.