Stop and Wait protocol

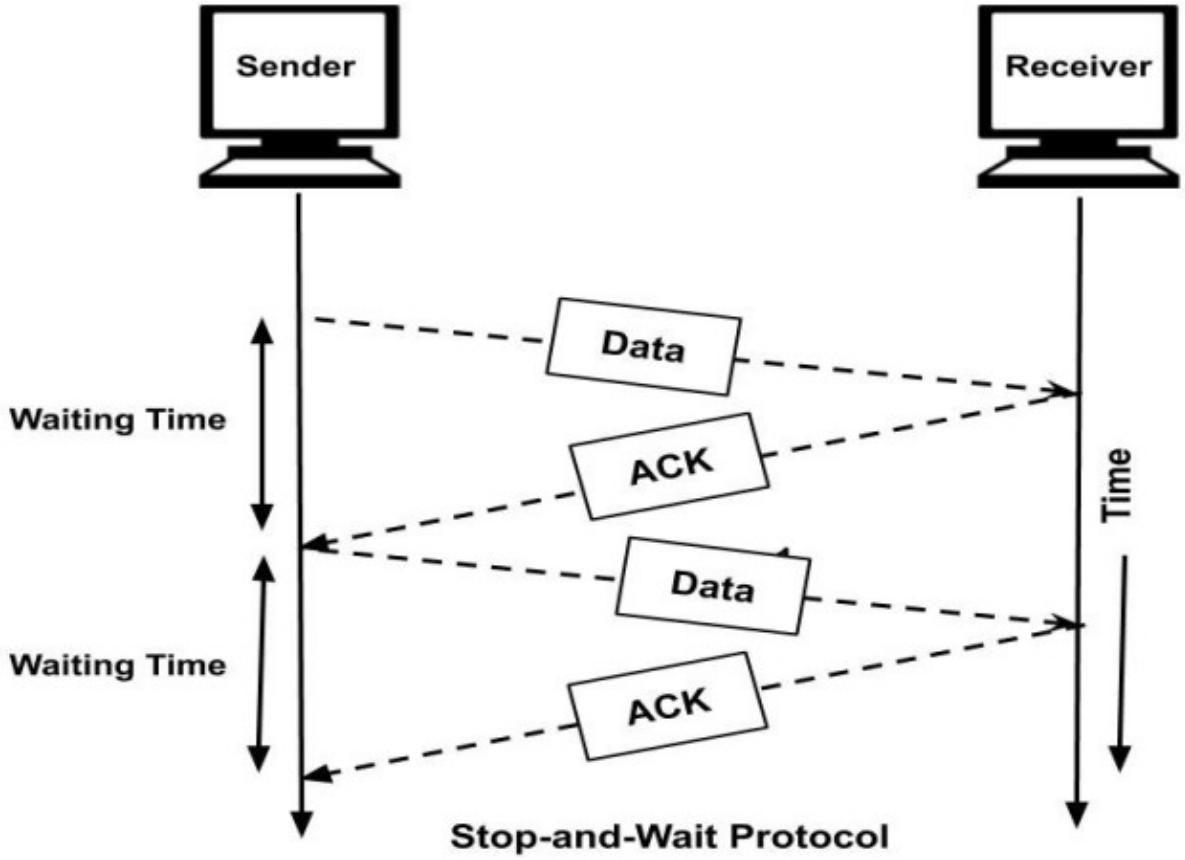
Anubhav Gupta :: IT :: 19124014

Aim :: To implement a Stop and Wait protocol depicting

(i) Normal operation (ii) Frame is lost (iii) Acknowledgement lost (iv) Delayed Acknowledgement

Theory::

It is the simplest flow control method. In this, the sender will send one frame at a time to the receiver. The sender will stop and wait for the acknowledgment from the receiver. This time(i.e. the time between message sending and acknowledgement receiving) is the waiting time for the sender and the sender is totally idle during this time. When the sender gets the acknowledgment(ACK), then it will send the next data packet to the receiver and wait for the acknowledgment again and this process will continue as long as the sender has the data to send. This can be understood by the diagram below:



This is achieved using the following Code with help of socket programming and Threading in Python 3.8

Sender Side

In [ ]:

```
import socket
from threading import *
import time

serversocket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
host = "localhost"
port = 4001

serversocket.bind((host, port))

class client(Thread):
    def __init__(self, socket, address):
        Thread.__init__(self)
        self.sock = socket
        self.addr = address
        self.start()

    def run(self):
        p = input('enter case \n 1 for simple \n 2 for Data Loss \n 3 for delay in a
        print(p)
        start_time = time.time()
        seconds = 5
        if p == '1':

            while True:
                current_time = time.time()
                elapsed = current_time - start_time
                if elapsed < seconds:
                    r=input("Send data -->")
                    r=r + p
                    clientsocket.send(r.encode())
                    print(clientsocket.recv(1024).decode())
                else:
                    break

        if p == '2':   # --The No acknowledgement Case --
            while True:
                current_time = time.time()
                elapsed = current_time - start_time
                if elapsed < seconds:
                    r=input("Send data -->")
                    r=r + p
                    clientsocket.send(r.encode())
                    if(not clientsocket.recv(1024).decode()):
                        print("no acknowledgement recieved Data Loss")
                    # print(clientsocket.recv(1024).decode())
                else:
                    break
            print("timeout :: No acknowledgement recieved")
        if p == '3':  # -- the Delay and timeout Case ---
            while True:
                current_time = time.time()
                elapsed = current_time - start_time
                if elapsed < seconds:
                    r=input("Send data -->")
                    r=r + p
                    clientsocket.send(r.encode())
                    print(clientsocket.recv(1024).decode())
                else:
                    break
            print("timeout :: No acknowledgement Late")
        if p == '4':   #-- the Frame Loss Case---
            while True:
                current_time = time.time()
                elapsed = current_time - start_time
```

```
                if elapsed < seconds:
                    r=input("Send data -->")
                    # r=r + p
                    # clientsocket.send(r.encode())
                    print(clientsocket.recv(1024).decode())
                else:
                    break
            print("timeout :: No acknowledgement recieved as No data was sent")



serversocket.listen(5)
print ('Sender ready and is listening')
while (True):

    #to accept all incoming connections
    clientsocket, address = serversocket.accept()
    print("Receiver "+str(address)+" connected")
    #create a different thread for every
    #incoming connection
    client(clientsocket, address)
```

```
Sender ready and is listening
```

The Reciever Side Code

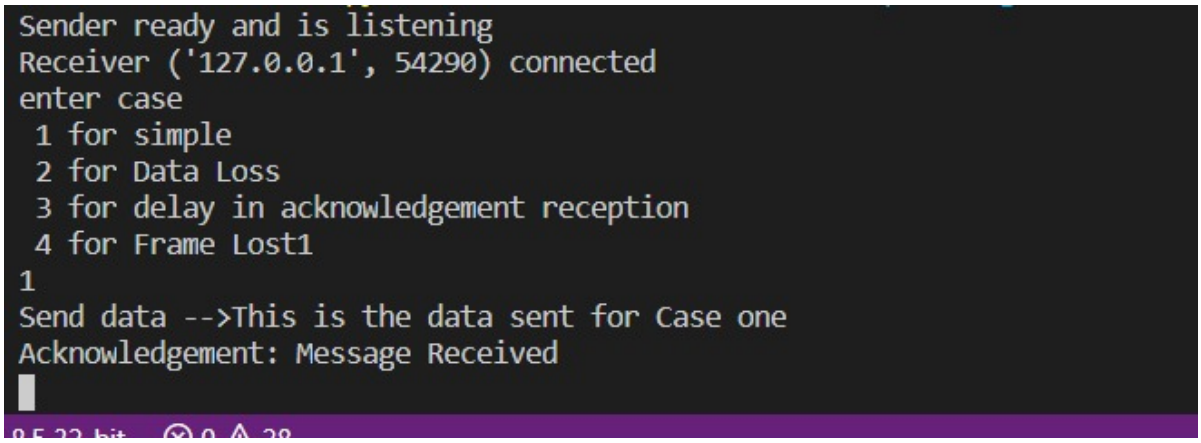
the connection is made by binding a socket to localhost at port 4001 and Localhost is used as a server for the communicaton

In [ ]:

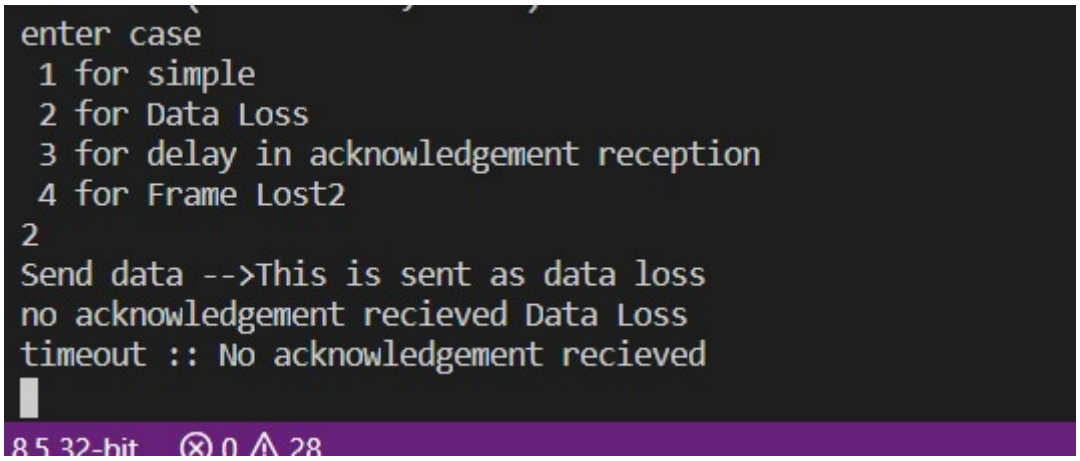
```
import socket
import time
s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
host ="localhost"
port =4001
s.connect((host,port))


start_time = time.time()
seconds = 5

while True:

    current_time = time.time()
    elapsed = current_time - start_time
    if elapsed < seconds:
        data=s.recv(1024).decode()
        case =  data[-1]
        print(case)
        if case == '1':
            print("Received --> "+data)
            str="Acknowledgement: Message Received"
            s.send(str.encode())
        if case == '2':
            s.close ()
            print('No acknowledgement will be sent')
        if case == '3':
            time.sleep(7)
            print("Received --> "+data)
            str="Acknowledgement: Message was sent with delay"
            s.send(str.encode())
    elif elapsed > seconds:
        str="Frame Loss No message recieved"
        s.send(str.encode())
```

```
s.close ()
```

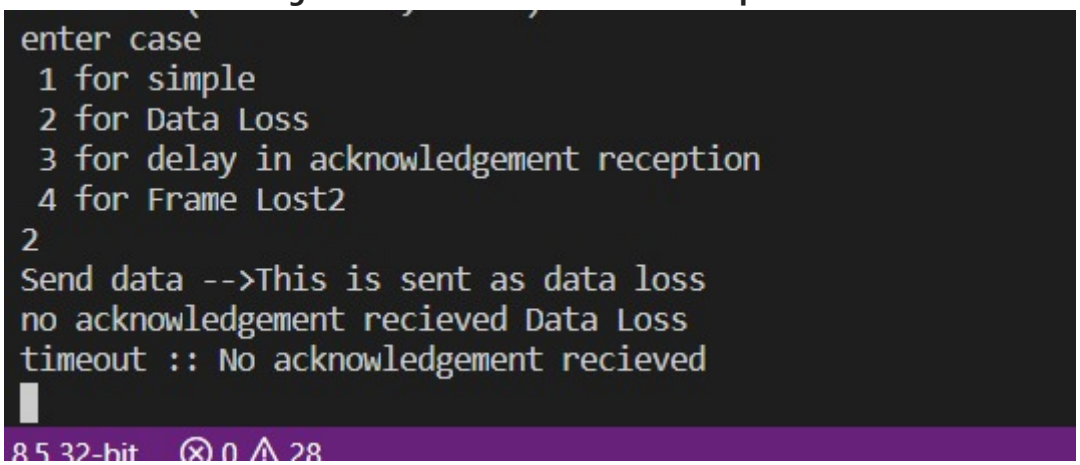
Case 1 : data is sent sucessfuly and is recieved sucessfully::



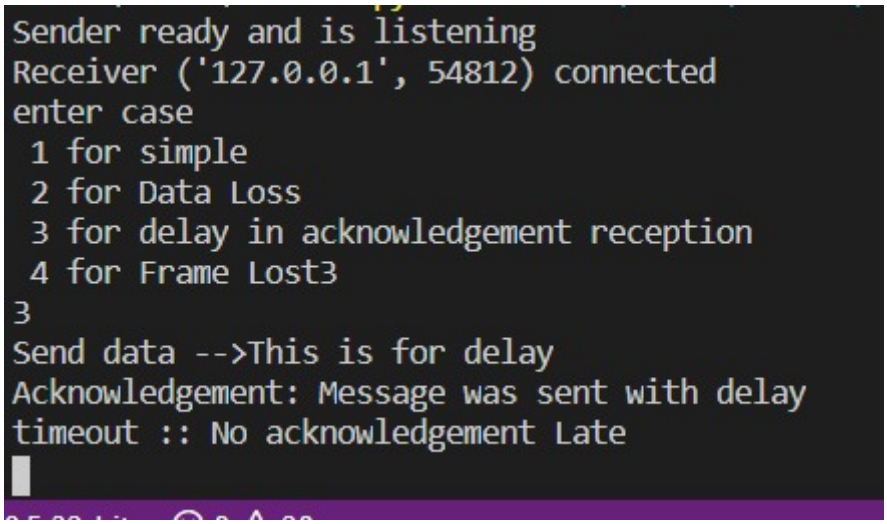
Case 2 No Acknowledgement recieved



Case 2 No Acknowledgement recieved due to frame drop

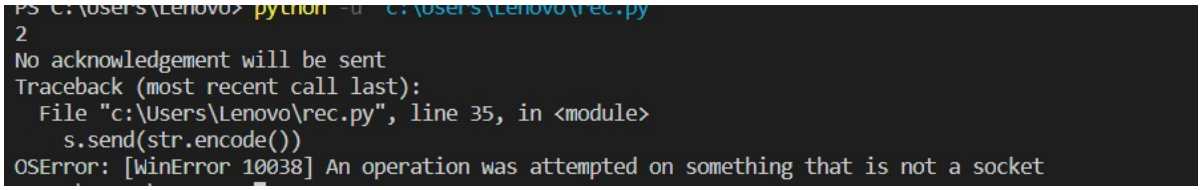


Case 4 recieved but delayed and timeout error

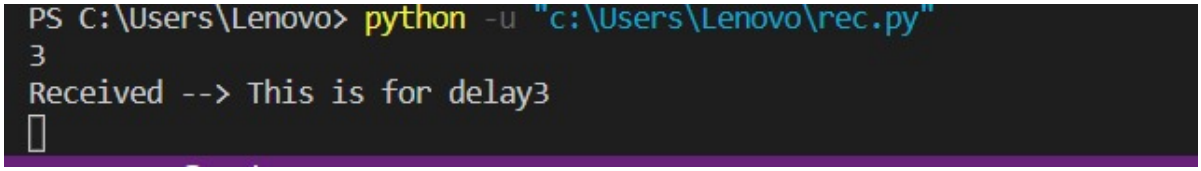


Terminal Snippets for reciever side

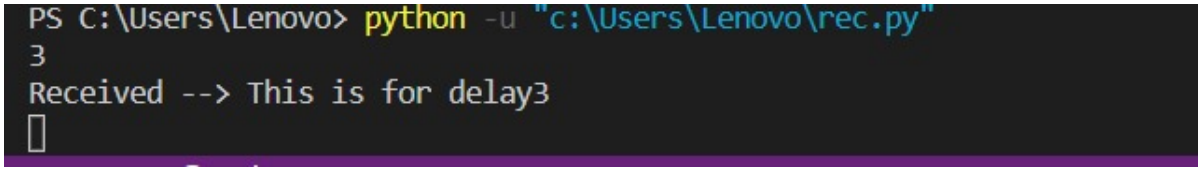
No Acknowledgement recieved



Acknowledgement recieved but late due to sleep called form time library



Acknowledgement Error



In [ ]:

In [ ]: