CN ASSIGNMENT 1

(Group No 3)

Piyush Narula (2022354)

Aditya Aggrawal (2022028)

Question 1: UDP Pinger Program

Introduction

Implemented a UDP ping client to communicate with a provided ping server. The client sends data packets to the server, which processes them by capitalizing the messages and sending them back. We incorporated functionality to measure round-trip times for each packet sent. The server simulates packet loss, allowing us to handle and report on lost packets, thereby enhancing the client's ability to assess the reliability of the connection.

```python
import random
from socket import *

# Create a UDP socket
# Notice the use of SOCK_DGRAM for UDP packets
serverSocket = socket(AF_INET, SOCK_DGRAM)

# Assign IP address and port number to socket
serverSocket.bind(('', 12000))

while True:
    # Generate random number in the range of 1 to 10
    rand = random.randint(1, 10)
    #NOTE: Wrong implementation of randint in provided code.

    # Receive the client packet along with the address it is coming from
    message, address = serverSocket.recvfrom(1024)

    # Capitalize the message from the client
    message = message.upper()

    # If rand is less than 4, we consider the packet lost and do not respond
    if rand < 4:
        continue

    # Otherwise, the server responds
    serverSocket.sendto(message, address)
```

```python
from socket import *
import time
import random

address = ("127.0.0.1", 12000)

clientSocket = socket(AF_INET, SOCK_DGRAM)
clientSocket.settimeout(1)

rtt=[]
lost=0
num_pings = 10
for i in range(num_pings):
    send_time=time.time()
    data = f'Ping {i+1} {send_time}'

    try:
        clientSocket.sendto(data.encode(), address)
        response, server = clientSocket.recvfrom(1024)
        print(f'Response: {response.decode()}')

        #NOTE: Can introduce artificial delay for better demonstration of RTT
        # artificial_delay = random.uniform(0.1, 0.9)
        # time.sleep(artificial_delay)

        recieve_time = time.time()
        rt = recieve_time - send_time
        rtt.append(rt)
        print(f'Ping {i+1} RTT : {rt} seconds')
    except timeout:
        lost+=1
        print(f'Ping {i+1} Request Timeout')


if(len(rtt)>0):
    mini=min(rtt)
    maxi=max(rtt)
    avg=sum(rtt)/len(rtt)
    print(f'Avgerage RTT : {avg} seconds')
    print(f'Maximum RTT : {maxi} seconds')
    print(f'Minimum RTT : {mini} seconds')

total_loss=(lost/num_pings)*100
print(f'Packet Loss Rate: {total_loss}%')
```

Client Report

```
piyush@piyush:~/Downloads/CN A1$ python3 UDPPingerClient.py
Ping 1 Request Timeout
Ping 2 Request Timeout
Response: PING 3 1726854479.153243
Ping 3 RTT : 0.00034499168395996094 seconds
Response: PING 4 1726854479.1536133
Ping 4 RTT : 8.630752563476562e-05 seconds
Ping 5 Request Timeout
Ping 6 Request Timeout
Ping 7 Request Timeout
Response: PING 8 1726854482.1584687
Ping 8 RTT : 0.00033473968505859375 seconds
Response: PING 9 1726854482.1588304
Ping 9 RTT : 0.00011944770812988281 seconds
Ping 10 Request Timeout
Avgerage RTT : 0.00022137165069580078 seconds
Maximum RTT : 0.00034499168395996094 seconds
Minimum RTT : 8.630752563476562e-05 seconds
Packet Loss Rate: 60.0%
piyush@piyush:~/Downloads/CN A1$
```

- This UDP ping client sends ten ping requests to the server, measuring the round-trip time (RTT) for each response. The client records the time taken for each packet to be sent and received, calculating the RTT by subtracting the send time from the receive time.
- If a response is not received within the specified timeout (1 second), the client counts the packet as lost and increments the lost packet counter. After completing all ping attempts, the client calculates and prints the average, minimum, and maximum RTT values based on successful responses. It also calculates the packet loss rate as a percentage of total pings
- A lower average RTT indicates better performance, while a high packet loss rate suggests potential issues in the network, such as congestion or instability.

Question 2: UDP Heartbeat Program

Functionality

The provided UDP program facilitates communication between the server and the client. The client sends messages to the server, referred to as heartbeat. One message consists of a sequence number and the current timestamp. The server decodes the message, extracts the timestamp sent by the client, calculates the time difference between when the client sent the message and when the server received it, and then sends the calculated value back to the client. The client reports this time difference and simulates packet loss. The server drops some packets based on a randomly generated variable with a 30% probability. The client waits for 1 second before it assumes the packet was lost and times out. This introduces an artificial packet loss to emulate a real-life network.

Server Timeout Analysis

At each ping, there is a 30% probability that the packet is lost. This means that the probability of three consecutive packet losses is 2.7%. This is a considerably very low probability. Hence, theoretically speaking, there is a high probability that the number of heartbeats stops between 10s and 100s. In practice, it is seen that the program might terminate as early as 5 pings and may even cross 100 heartbeats before termination.

Server Report

```
Sequence Number: 1
Server timestamp: 171842.8353454
 Client timestamp: 171842.8341389

Sequence Number: 2
Server timestamp: 171842.8369356
 Client timestamp: 171842.8368793

Sequence Number: 3
Server timestamp: 171842.8371431
 Client timestamp: 171842.8370888

Sequence Number: 4
Server timestamp: 171842.8376842
 Client timestamp: 171842.8376042

Sequence Number: 5
Server timestamp: 171843.8428391
 Client timestamp: 171843.8427394

Sequence Number: 6
Server timestamp: 171843.8431396
 Client timestamp: 171843.8430959

Sequence Number: 7
Server timestamp: 171844.845894
 Client timestamp: 171844.8456898

Sequence Number: 8
Server timestamp: 171845.8517337
 Client timestamp: 171845.8515928

Sequence Number: 9
Server timestamp: 171845.8527086
 Client timestamp: 171845.8526264
```

```
Sequence Number: 27
Server timestamp: 171849.8860889
 Client timestamp: 171849.8860551

Sequence Number: 28
Server timestamp: 171850.8905536
 Client timestamp: 171850.8903493

Sequence Number: 29
Server timestamp: 171850.8911315
 Client timestamp: 171850.8910632

Sequence Number: 30
Server timestamp: 171850.8922153
 Client timestamp: 171850.8918156

Sequence Number: 31
Server timestamp: 171850.8924955
 Client timestamp: 171850.8924738

Sequence Number: 32
Server timestamp: 171850.8926658
 Client timestamp: 171850.8926407

Sequence Number: 33
Server timestamp: 171851.8959777
 Client timestamp: 171851.8958025

Sequence Number: 34
Server timestamp: 171851.8971552
 Client timestamp: 171851.8970809

Sequence Number: 35
Server timestamp: 171852.9012785
 Client timestamp: 171852.9010285

Sequence Number: 36
Server timestamp: 171853.9094337
 Client timestamp: 171853.9092974
```

Client Report

```
C:\Users\Aditya\Desktop\CN A1>python UDPHeartbeatClient.py
Time Difference: 0.0012064999900758266
Time Difference: 5.629999213851988e-05
Time Difference: 5.43000060133636e-05
Ping 4 Request Timeout
Time Difference: 9.970000246539712e-05
Ping 6 Request Timeout
Ping 7 Request Timeout
Time Difference: 0.00014089999604038894
Time Difference: 8.219998562708497e-05
Ping 10 Request Timeout
Time Difference: 0.00019049999536946416
Time Difference: 9.039998985826969e-05
Time Difference: 5.5599986808374524e-05
Time Difference: 0.00010999999358318746
Ping 15 Request Timeout
Time Difference: 0.00020300000323913991
Ping 17 Request Timeout
Ping 18 Request Timeout
Time Difference: 0.00017899999511428177
Time Difference: 8.440000237897038e-05
Time Difference: 0.00015969999367371202
Time Difference: 0.00011889997404068708
Time Difference: 4.549999721348286e-05
Time Difference: 3.9300008211284876e-05
Time Difference: 3.5200006095692515e-05
Time Difference: 4.390001413412392e-05
Ping 27 Request Timeout
Time Difference: 0.00020430000131379813
Time Difference: 6.829999620094895e-05
Time Difference: 0.00039969998761080205
Time Difference: 2.1700019715353847e-05
Ping 32 Request Timeout
Time Difference: 0.00017520002438686788
Ping 34 Request Timeout
Ping 35 Request Timeout
Ping 36 Request Timeout
Program terminated after 36 pings
Packet Loss Rate: 33.333333333333336%
```