CPSC 471-04 Project #2

Required Software

You are required to use Python 3.9.7 or newer for this project.

Project Submission

You are responsible for the content of your submitted file. Double-check your submission in Canvas to ensure that the submitted file is not corrupted, and it is readable, and it contains all required files and their contents are what you intend to submit. Submitting a wrong file (from another class for example) is not a valid excuse.

Submit **one zip file** using naming convention: **firstnamelastname_p2.zip** in Canvas.

The zip file shall contain the following files:

(1) PDF report file using naming convention **firstnamelastname_p2.pdf** (must be a pdf document, not Word).
(2) Python UDP client code file from Part 1 **xx1.py**
(3) Python UDP server code file from Part 2 **xx2.py**
(4) Python UDP server code file from Part 3 **xx3.py**
(5) Python Heartbeat client code file **xx4c.py**
(6) Python Heartbeat server code file **xx4s.py**

where xx = initials (the two characters representing the first character of your first and last name).

Part 1: UDP Pinger with No Delay and No Loss

In this project, you will learn the basics of socket programming for UDP in Python. You will learn how to send and receive datagram packets using UDP sockets and, how to set a proper socket timeout. Throughout the lab, you will gain familiarity with a Ping application and its usefulness in computing statistics such as packet loss rate.

You will first study a simple Internet ping server written in the Python and implement a corresponding client. The functionality provided by these programs is like the functionality provided by standard ping programs available in modern operating systems. However, these programs use a simpler protocol, UDP, rather than the standard Internet Control Message Protocol (ICMP) to communicate with each other. The ping protocol allows a client machine to send a packet of data to a remote machine, and have the remote machine return the data back to the client unchanged (an action referred to as echoing). Among other uses, the ping protocol allows hosts to determine round-trip times to other machines.

You are given the complete code for the Ping server in the next subsection. Your task is to write the UDP Ping client.

Server Code

You are provided with the following code which fully implements a ping server. You need to run this code before running your client program. Your client code in this portion of the project will be graded using this exact same server code (do not modify it for this portion of the project).

You should study this code carefully, as it will help you write your ping client code.

```
# udppingserver_no_loss.py
from socket import *
# Create a UDP socket
serverSocket = socket(AF_INET, SOCK_DGRAM)
# Assign IP address and port number to socket
serverSocket.bind(('', 12000))
while True:
    # Receive the client packet along with the address it is coming from
    message, address = serverSocket.recvfrom(1024)
    # The server responds
    serverSocket.sendto(message, address)
```

The server sits in an infinite loop listening for incoming UDP packets. When a packet comes in, the server simply sends it back to the client.

Client Code

You task is to implement the client program as explained below.

The client should send a specified number of pings to the server. Because UDP is an unreliable protocol, a packet sent from the client to the server may be lost in the network, or vice versa. For this reason, the client cannot wait indefinitely for a reply to a ping message. You should get the client wait up to one second for a reply; if no reply is received within one second, your client program should assume that the packet was lost during transmission across the network. You will need to look up the Python documentation to find out how to set the timeout value on a datagram socket.

Client Code Requirements

The client program should:

- send the ping message using UDP
- print the response message from server if any was received
- calculate and print the round-trip time (RTT), in milliseconds, of each packet if the server responses
- otherwise, print "Request timed out"
- provide a summary report at the end (of all pings) which includes:
    - minimum RTT in milliseconds,
    - maximum RTT in milliseconds,
    - average RTT in milliseconds,
    - percentage packet loss rate

You should run the udppingserver_no_loss.py on your machine and test your client by sending packets to the localhost.

Ping Message Format

The client ping message is a one line, consisting of ASCII characters and must be in the following format:

```
        firstname ping_number date_and_time
```

where:

firstname is your first name.

ping_number starts at 1 and progresses to total number of pings for each successive ping message sent by the client, and time is the time when the client sends the message.

For example: The following is a sample display from the client program for student Mary.

```
Mary 1: server reply: Mary 1 Fri Sep 23 09:00:15 2022, RTT = 3.99 ms
Mary 2: server reply: Mary 2 Fri Sep 23 09:00:15 2022, RTT = 0.00 ms
```

Notes: the blue text represents the response from the server upon receiving the ping message (which is the message the server received from the client). The other texts are from the client itself.

Refer to the Appendix section on the last page for samples and required data format.

What to Hand in

PDF file report

Create a section called **Part 1 – UDP Pinger with No Delay and No Loss**. Include the followings:

(1) Describe the operation of your UDP Pinger, for example how it works.
(2) Explain how to specify the timeout value for a datagram socket. Provide an example.

(3) Explain how to run your code, i.e., command line and any applicable parameter(s)
    a. Include run-time screen captures for a sequence consists of **10 pings**

    Refer to the Appendix section on the last page for samples and required data format.

(4) Include your Python code listing:
    a. Include as text the listing of your Python code.
    Please use consolas font size 10 or equivalent monospace font. The use of these monospace font is to clearly show indentations in your code.

Submission zip file

include your Python client code file **xx1.py**

where xx = initials (the two characters representing the first character of your first and last name).

Part 2: UDP Pinger with Delays

Delays

Our experiment so far has been on a local host running both server and client programs, and therefore we saw zero delays. In this portion of the project, you are asked to modify the server code to simulate random **RTT delays ranging from 10ms to 20ms.**

Hint: Create a variable which holds a randomized integer to determine the delay amount.

What to Hand in

PDF file report

Create a section called **Part 2 – UDP Pinger No Loss, with Delays**. Include the followings:

(1) Describe the operation of your UDP Ping Server and explain how it simulates **10ms to 20ms RTT delays**.
(2) Explain how to run your code, i.e., command line and any applicable parameter(s)
    a. Include run-time screen captures for a sequence consists of **10 pings**

    Refer to the Appendix section on the last page for samples and required data format.

(3) Include your Python code listing of your UDP Ping Server:
    a. Include as text the listing of your Python code.
    Please use consolas font size 10 or equivalent monospace font. The use of these monospace font is to clearly show indentations in your code.

Submission zip file

include your Python UDP Ping Server with Delays code file **xx2.py** where xx = initials (the two characters representing the first character of your first and last name).

Part 3: UDP Pinger with Delays and Packet Losses

Packet Loss Injection

UDP provides applications with an unreliable transport service. Messages may get lost in the network due to router queue overflows, faulty hardware, or some other reasons. Because packet loss is rare or even non-existent in typical campus or home networks, you are asked to modify the server code in this portion of the project to inject artificial losses to simulate the effects of network packet loss.

Hint: Create a variable which holds a randomized integer to determine whether a particular incoming packet is lost or not.

What to Hand in

PDF file report

Create a section called **Part 3 – UDP Pinger with Delays and Packet Losses**. Include the followings:

(1) Describe the operation of your UDP Ping Server and explain how it simulates **delays between 10ms and 20ms, with up to 10% packet losses**.

(2) Explain how to run your code, i.e., command line and any applicable parameter(s)

   a. Include run-time screen captures for a sequence consists of **50 pings**

      Refer to the Appendix section on the last page for samples and required data format.

(3) Include your Python code listing of your UDP Ping Server:

   a. Include as text the listing of your Python code.
      Please use consolas font size 10 or equivalent monospace font. The use of these monospace font is to clearly show indentations in your code.

Submission zip file

include your Python UDP Ping Server with Delay and Losses code file **xx3.py**

where xx = initials (the two characters representing the first character of your first and last name).

Part 4: HeartBeat Monitor Using Python

Another similar application to the UDP Ping would be the UDP Heartbeat. The Heartbeat can be used to check if an application is up and running on the client side and to report one-way packet loss. The client continuously sends a message acting as a heartbeat in the UDP packet to the server, which is monitoring the heartbeat (i.e., the UDP packets) of the client. Upon receiving the packets, the server calculates the time difference. If the heartbeat packets are missing for some specified time interval, the server can assume that the client application has stopped working.

Implement the UDP Heartbeat (both client and server). You are asked to create both the server and client programs.

Use the following file naming convention:

- **xx4c.py** for client
- **xx4s.py** for server

where xx = initials (the two characters representing the first character of your first and last name).

The client program sends a ping message to the server using UDP every 2 to 25 seconds to simulate delays.

Run two clients and one server on your computer. Identify the clients as follow:

- "Yourfirstname client1" for client 1
- "Yourfirstname client2" for client 2

The heartbeat data format from the client **must show the date and time** and conform exactly as shown below.

Example: For a student named Mary the following messages are sent from the client to the server.

Message from client 1 to the server:

```
Mary client1 heartbeat at Fri Sep 23 11:09:13 2022
```

Message from client 2 to the server:

```
Mary client2 heartbeat at Fri Sep 23 11:09:28 2022
```

The server program monitors if any ping was received from any of the clients. **If there is no ping from any clients for 20 or more seconds, it prints the message "No heartbeat after 20 seconds. Server quits. Server stops."**.

Required data format at the server:

The server program prints out:

```
Server received: Mary client1 heartbeat at Fri Sep 23 11:11:56
2022 Last heartbeat received 2 seconds ago
```

```
Server received: Mary client2 heartbeat at Fri Sep 23 11:11:59
2022 Last heartbeat received 3 seconds ago
```

Refer to the Appendix section on the last page for samples and required data format.

What to Hand in

PDF file report

Create a section called **Part 4 – UDP Heartbeat Monitor**. This portion of your report includes:

(1) Instructions on how to run the code, i.e., command line and any applicable parameter(s) for the client and the server programs
(2) Run-time screen capture showing:
    a. the client sends heartbeat pings to the server.
    b. server prints the received heartbeat pings from the client, and the time interval.
    c. server detects absence of client heartbeat and quits.
(3) Python code listing:
    a. Include as text, the client program listing.
    b. Include as text, the server program listing.

    Please use consolas font size 10 or equivalent monospace fonts. The use of these monospace font is to clearly show indentations in your code.

Submission zip file

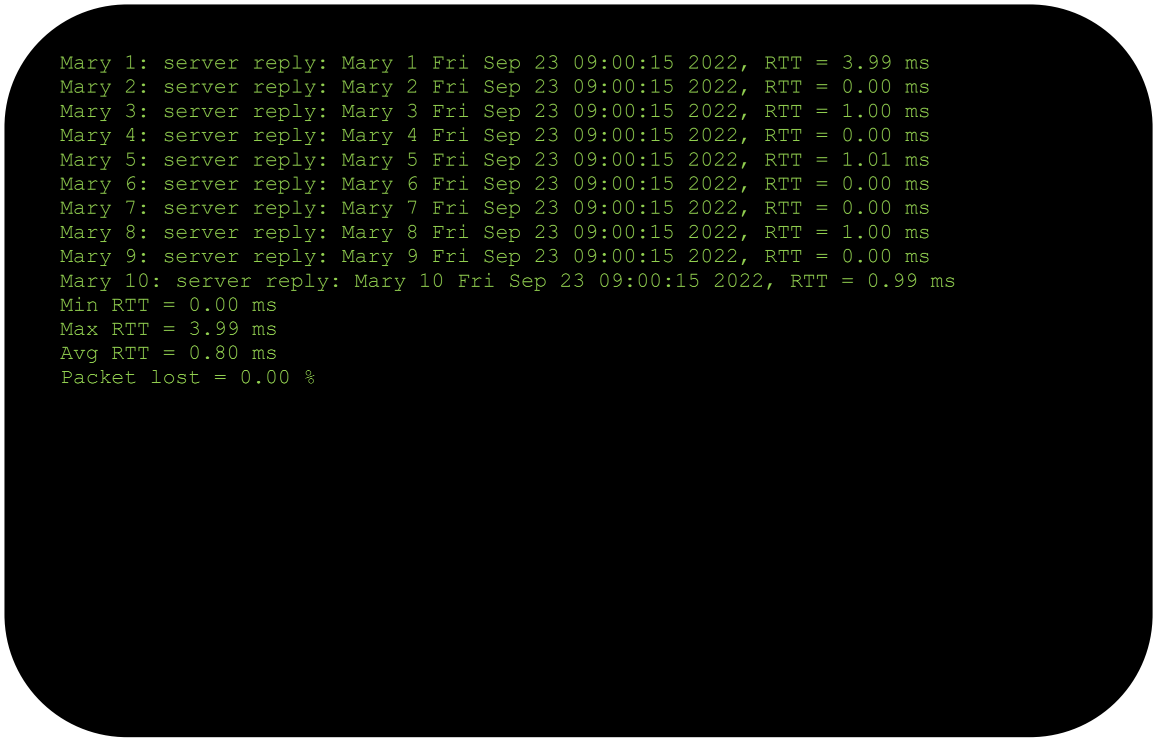
include both your Python client and server code files **xx4c.py** and **xx4s.py**

where xx = initials (the two characters representing the first character of your first and last name).

Appendix

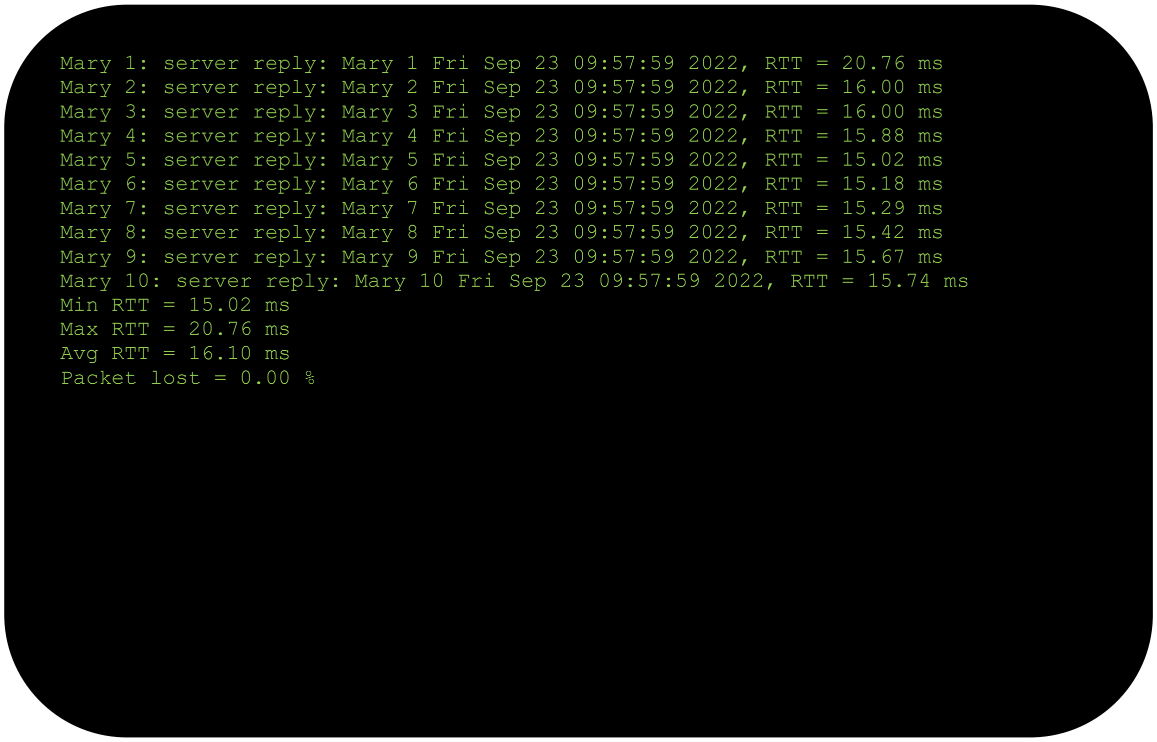
Note: All the examples in this section use an arbitrary student name Mary.

Part 1: Sequence of 10 Pings with No Delay and No Packet Loss



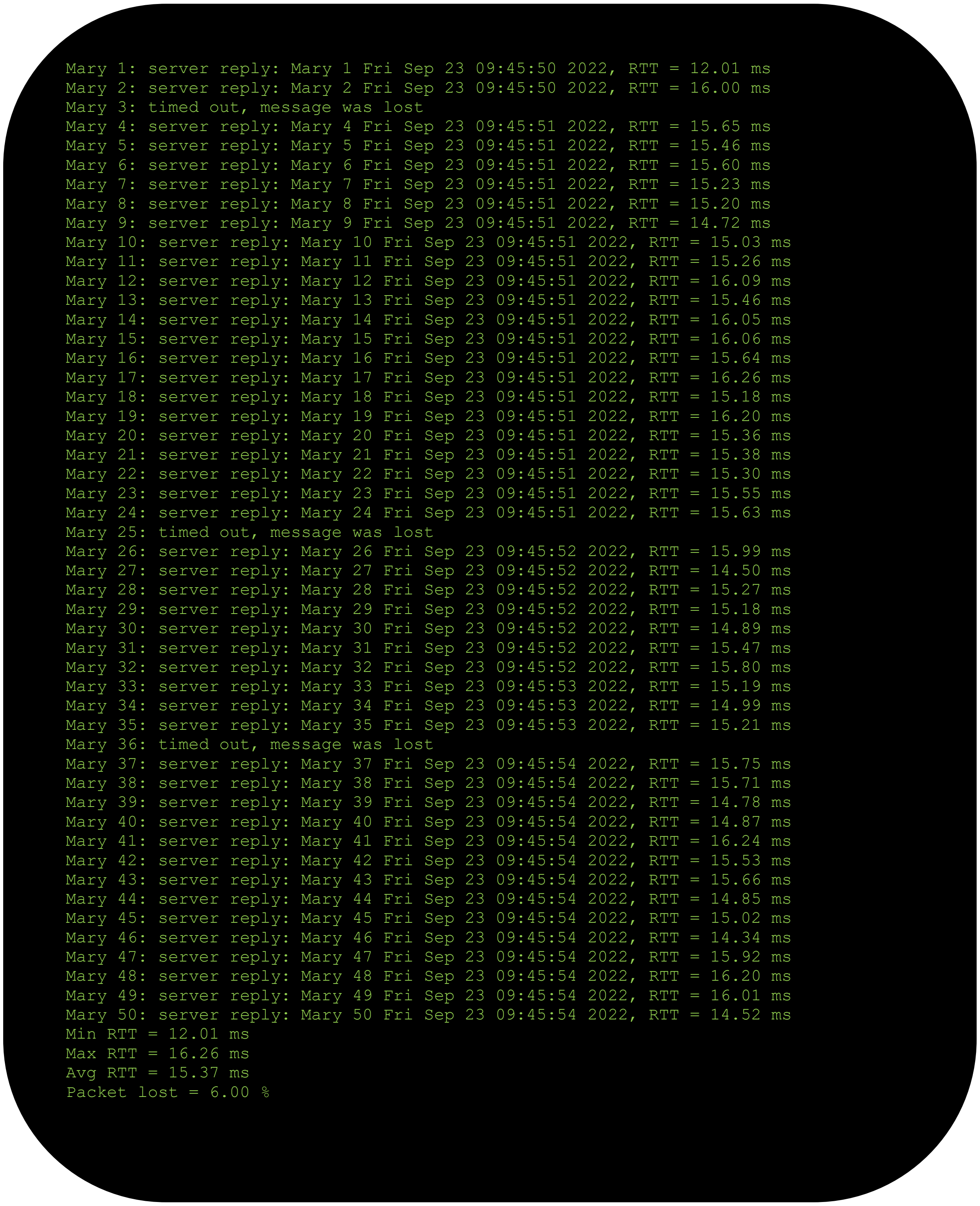
```
Mary 1: server reply: Mary 1 Fri Sep 23 09:00:15 2022, RTT = 3.99 ms
Mary 2: server reply: Mary 2 Fri Sep 23 09:00:15 2022, RTT = 0.00 ms
Mary 3: server reply: Mary 3 Fri Sep 23 09:00:15 2022, RTT = 1.00 ms
Mary 4: server reply: Mary 4 Fri Sep 23 09:00:15 2022, RTT = 0.00 ms
Mary 5: server reply: Mary 5 Fri Sep 23 09:00:15 2022, RTT = 1.01 ms
Mary 6: server reply: Mary 6 Fri Sep 23 09:00:15 2022, RTT = 0.00 ms
Mary 7: server reply: Mary 7 Fri Sep 23 09:00:15 2022, RTT = 0.00 ms
Mary 8: server reply: Mary 8 Fri Sep 23 09:00:15 2022, RTT = 1.00 ms
Mary 9: server reply: Mary 9 Fri Sep 23 09:00:15 2022, RTT = 0.00 ms
Mary 10: server reply: Mary 10 Fri Sep 23 09:00:15 2022, RTT = 0.99 ms
Min RTT = 0.00 ms
Max RTT = 3.99 ms
Avg RTT = 0.80 ms
Packet lost = 0.00 %
```

Part 2: Sequence of 10 Pings with 10ms-20ms Delays and No Packet Loss



```
Mary 1: server reply: Mary 1 Fri Sep 23 09:57:59 2022, RTT = 20.76 ms
Mary 2: server reply: Mary 2 Fri Sep 23 09:57:59 2022, RTT = 16.00 ms
Mary 3: server reply: Mary 3 Fri Sep 23 09:57:59 2022, RTT = 16.00 ms
Mary 4: server reply: Mary 4 Fri Sep 23 09:57:59 2022, RTT = 15.88 ms
Mary 5: server reply: Mary 5 Fri Sep 23 09:57:59 2022, RTT = 15.02 ms
Mary 6: server reply: Mary 6 Fri Sep 23 09:57:59 2022, RTT = 15.18 ms
Mary 7: server reply: Mary 7 Fri Sep 23 09:57:59 2022, RTT = 15.29 ms
Mary 8: server reply: Mary 8 Fri Sep 23 09:57:59 2022, RTT = 15.42 ms
Mary 9: server reply: Mary 9 Fri Sep 23 09:57:59 2022, RTT = 15.67 ms
Mary 10: server reply: Mary 10 Fri Sep 23 09:57:59 2022, RTT = 15.74 ms
Min RTT = 15.02 ms
Max RTT = 20.76 ms
Avg RTT = 16.10 ms
Packet lost = 0.00 %
```

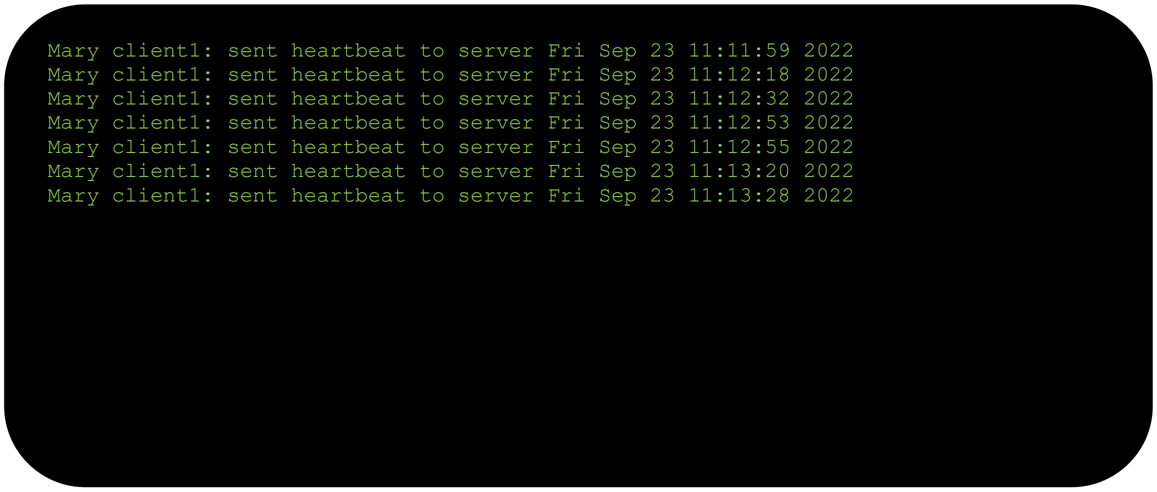
Part 3: Sequence of 50 Pings with 10ms-20ms Delays and up to 10% Packet Loss



```
Mary 1: server reply: Mary 1 Fri Sep 23 09:45:50 2022, RTT = 12.01 ms
Mary 2: server reply: Mary 2 Fri Sep 23 09:45:50 2022, RTT = 16.00 ms
Mary 3: timed out, message was lost
Mary 4: server reply: Mary 4 Fri Sep 23 09:45:51 2022, RTT = 15.65 ms
Mary 5: server reply: Mary 5 Fri Sep 23 09:45:51 2022, RTT = 15.46 ms
Mary 6: server reply: Mary 6 Fri Sep 23 09:45:51 2022, RTT = 15.60 ms
Mary 7: server reply: Mary 7 Fri Sep 23 09:45:51 2022, RTT = 15.23 ms
Mary 8: server reply: Mary 8 Fri Sep 23 09:45:51 2022, RTT = 15.20 ms
Mary 9: server reply: Mary 9 Fri Sep 23 09:45:51 2022, RTT = 14.72 ms
Mary 10: server reply: Mary 10 Fri Sep 23 09:45:51 2022, RTT = 15.03 ms
Mary 11: server reply: Mary 11 Fri Sep 23 09:45:51 2022, RTT = 15.26 ms
Mary 12: server reply: Mary 12 Fri Sep 23 09:45:51 2022, RTT = 16.09 ms
Mary 13: server reply: Mary 13 Fri Sep 23 09:45:51 2022, RTT = 15.46 ms
Mary 14: server reply: Mary 14 Fri Sep 23 09:45:51 2022, RTT = 16.05 ms
Mary 15: server reply: Mary 15 Fri Sep 23 09:45:51 2022, RTT = 16.06 ms
Mary 16: server reply: Mary 16 Fri Sep 23 09:45:51 2022, RTT = 15.64 ms
Mary 17: server reply: Mary 17 Fri Sep 23 09:45:51 2022, RTT = 16.26 ms
Mary 18: server reply: Mary 18 Fri Sep 23 09:45:51 2022, RTT = 15.18 ms
Mary 19: server reply: Mary 19 Fri Sep 23 09:45:51 2022, RTT = 16.20 ms
Mary 20: server reply: Mary 20 Fri Sep 23 09:45:51 2022, RTT = 15.36 ms
Mary 21: server reply: Mary 21 Fri Sep 23 09:45:51 2022, RTT = 15.38 ms
Mary 22: server reply: Mary 22 Fri Sep 23 09:45:51 2022, RTT = 15.30 ms
Mary 23: server reply: Mary 23 Fri Sep 23 09:45:51 2022, RTT = 15.55 ms
Mary 24: server reply: Mary 24 Fri Sep 23 09:45:51 2022, RTT = 15.63 ms
Mary 25: timed out, message was lost
Mary 26: server reply: Mary 26 Fri Sep 23 09:45:52 2022, RTT = 15.99 ms
Mary 27: server reply: Mary 27 Fri Sep 23 09:45:52 2022, RTT = 14.50 ms
Mary 28: server reply: Mary 28 Fri Sep 23 09:45:52 2022, RTT = 15.27 ms
Mary 29: server reply: Mary 29 Fri Sep 23 09:45:52 2022, RTT = 15.18 ms
Mary 30: server reply: Mary 30 Fri Sep 23 09:45:52 2022, RTT = 14.89 ms
Mary 31: server reply: Mary 31 Fri Sep 23 09:45:52 2022, RTT = 15.47 ms
Mary 32: server reply: Mary 32 Fri Sep 23 09:45:52 2022, RTT = 15.80 ms
Mary 33: server reply: Mary 33 Fri Sep 23 09:45:53 2022, RTT = 15.19 ms
Mary 34: server reply: Mary 34 Fri Sep 23 09:45:53 2022, RTT = 14.99 ms
Mary 35: server reply: Mary 35 Fri Sep 23 09:45:53 2022, RTT = 15.21 ms
Mary 36: timed out, message was lost
Mary 37: server reply: Mary 37 Fri Sep 23 09:45:54 2022, RTT = 15.75 ms
Mary 38: server reply: Mary 38 Fri Sep 23 09:45:54 2022, RTT = 15.71 ms
Mary 39: server reply: Mary 39 Fri Sep 23 09:45:54 2022, RTT = 14.78 ms
Mary 40: server reply: Mary 40 Fri Sep 23 09:45:54 2022, RTT = 14.87 ms
Mary 41: server reply: Mary 41 Fri Sep 23 09:45:54 2022, RTT = 16.24 ms
Mary 42: server reply: Mary 42 Fri Sep 23 09:45:54 2022, RTT = 15.53 ms
Mary 43: server reply: Mary 43 Fri Sep 23 09:45:54 2022, RTT = 15.66 ms
Mary 44: server reply: Mary 44 Fri Sep 23 09:45:54 2022, RTT = 14.85 ms
Mary 45: server reply: Mary 45 Fri Sep 23 09:45:54 2022, RTT = 15.02 ms
Mary 46: server reply: Mary 46 Fri Sep 23 09:45:54 2022, RTT = 14.34 ms
Mary 47: server reply: Mary 47 Fri Sep 23 09:45:54 2022, RTT = 15.92 ms
Mary 48: server reply: Mary 48 Fri Sep 23 09:45:54 2022, RTT = 16.20 ms
Mary 49: server reply: Mary 49 Fri Sep 23 09:45:54 2022, RTT = 16.01 ms
Mary 50: server reply: Mary 50 Fri Sep 23 09:45:54 2022, RTT = 14.52 ms
Min RTT = 12.01 ms
Max RTT = 16.26 ms
Avg RTT = 15.37 ms
Packet lost = 6.00 %
```

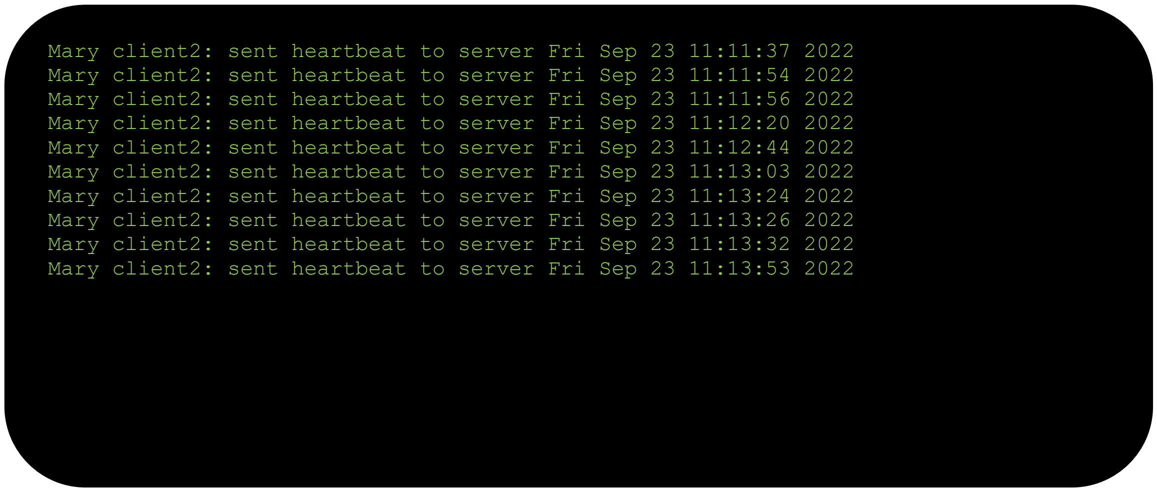
Part 4: Heartbeat Monitor Run-Time Samples

Client 1 Screen



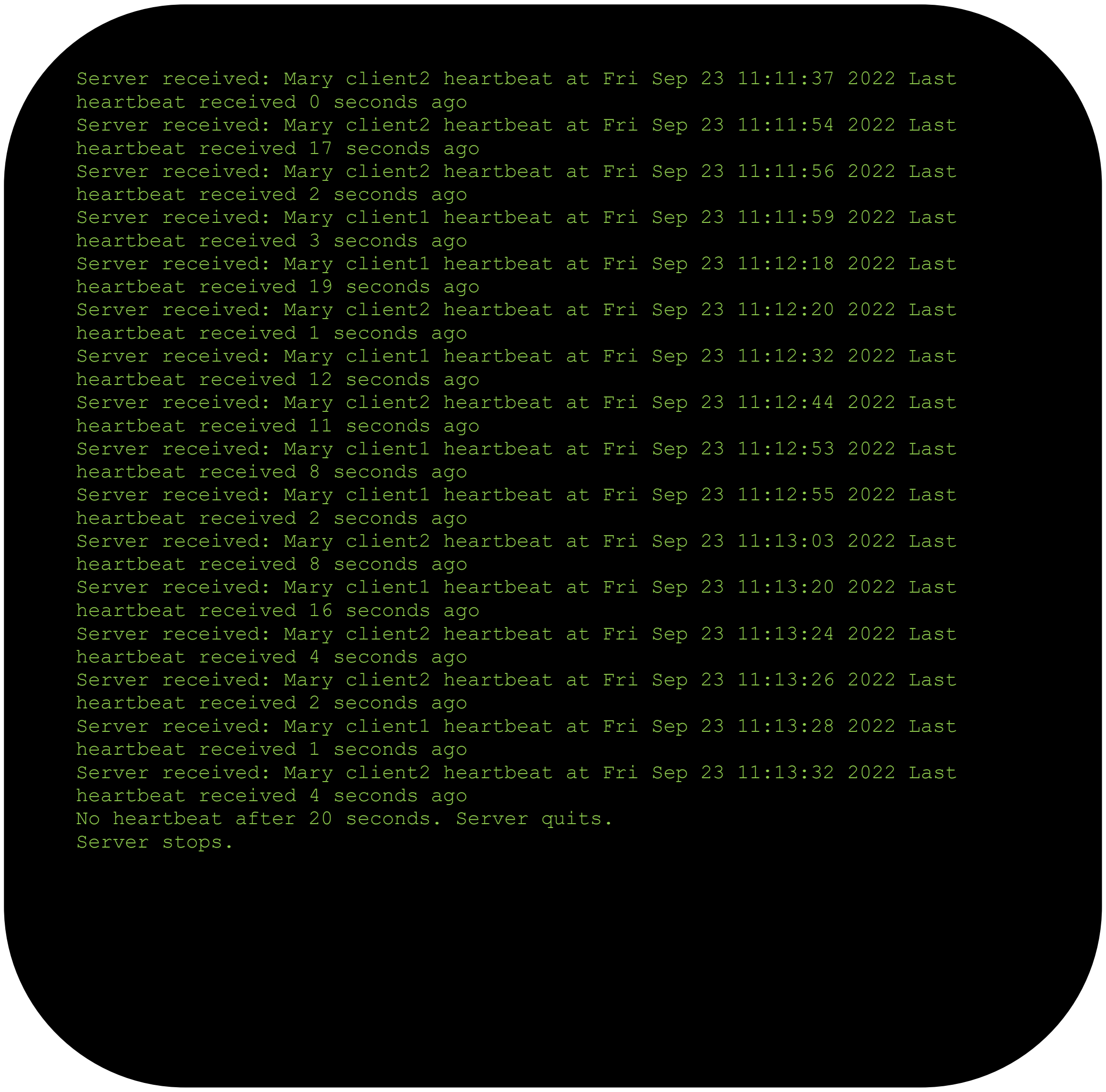
```
Mary client1: sent heartbeat to server Fri Sep 23 11:11:59 2022
Mary client1: sent heartbeat to server Fri Sep 23 11:12:18 2022
Mary client1: sent heartbeat to server Fri Sep 23 11:12:32 2022
Mary client1: sent heartbeat to server Fri Sep 23 11:12:53 2022
Mary client1: sent heartbeat to server Fri Sep 23 11:12:55 2022
Mary client1: sent heartbeat to server Fri Sep 23 11:13:20 2022
Mary client1: sent heartbeat to server Fri Sep 23 11:13:28 2022
```

Client 2 Screen



```
Mary client2: sent heartbeat to server Fri Sep 23 11:11:37 2022
Mary client2: sent heartbeat to server Fri Sep 23 11:11:54 2022
Mary client2: sent heartbeat to server Fri Sep 23 11:11:56 2022
Mary client2: sent heartbeat to server Fri Sep 23 11:12:20 2022
Mary client2: sent heartbeat to server Fri Sep 23 11:12:44 2022
Mary client2: sent heartbeat to server Fri Sep 23 11:13:03 2022
Mary client2: sent heartbeat to server Fri Sep 23 11:13:24 2022
Mary client2: sent heartbeat to server Fri Sep 23 11:13:26 2022
Mary client2: sent heartbeat to server Fri Sep 23 11:13:32 2022
Mary client2: sent heartbeat to server Fri Sep 23 11:13:53 2022
```

Server Screen



```
Server received: Mary client2 heartbeat at Fri Sep 23 11:11:37 2022 Last
heartbeat received 0 seconds ago
Server received: Mary client2 heartbeat at Fri Sep 23 11:11:54 2022 Last
heartbeat received 17 seconds ago
Server received: Mary client2 heartbeat at Fri Sep 23 11:11:56 2022 Last
heartbeat received 2 seconds ago
Server received: Mary client1 heartbeat at Fri Sep 23 11:11:59 2022 Last
heartbeat received 3 seconds ago
Server received: Mary client1 heartbeat at Fri Sep 23 11:12:18 2022 Last
heartbeat received 19 seconds ago
Server received: Mary client2 heartbeat at Fri Sep 23 11:12:20 2022 Last
heartbeat received 1 seconds ago
Server received: Mary client1 heartbeat at Fri Sep 23 11:12:32 2022 Last
heartbeat received 12 seconds ago
Server received: Mary client2 heartbeat at Fri Sep 23 11:12:44 2022 Last
heartbeat received 11 seconds ago
Server received: Mary client1 heartbeat at Fri Sep 23 11:12:53 2022 Last
heartbeat received 8 seconds ago
Server received: Mary client1 heartbeat at Fri Sep 23 11:12:55 2022 Last
heartbeat received 2 seconds ago
Server received: Mary client2 heartbeat at Fri Sep 23 11:13:03 2022 Last
heartbeat received 8 seconds ago
Server received: Mary client1 heartbeat at Fri Sep 23 11:13:20 2022 Last
heartbeat received 16 seconds ago
Server received: Mary client2 heartbeat at Fri Sep 23 11:13:24 2022 Last
heartbeat received 4 seconds ago
Server received: Mary client2 heartbeat at Fri Sep 23 11:13:26 2022 Last
heartbeat received 2 seconds ago
Server received: Mary client1 heartbeat at Fri Sep 23 11:13:28 2022 Last
heartbeat received 1 seconds ago
Server received: Mary client2 heartbeat at Fri Sep 23 11:13:32 2022 Last
heartbeat received 4 seconds ago
No heartbeat after 20 seconds. Server quits.
Server stops.
```