COMPUTER NETWORKS
USER MANUAL

SIMULATION APPLICATION

Version 1.0

Prepared by

Rishika V Menon: B200697CS
Sandhra GireeshKumar: B200701CS

Date: 15-04-2023

Introduction

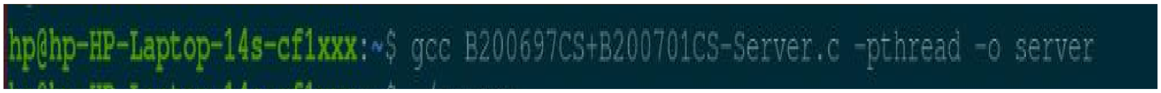
To design a simulation application that uses UDP sockets for the transmission and reception of packets between a sender and receiver. The sender will generate and transmit packets of two different types while the receiver will receive and process the packets. Sender packets contain a packet type (1 or 2), sequence number, byte array of 1024 bytes, and a checksum trailer for the entire packet. Four threads will be involved in the receiver process, each with a distinct function. First thread will check for errors in the received packets, the second and third threads will process packets of type1 and type2, respectively, and the fourth thread will periodically print out the total number of packets received.

Prerequisite

1. A linux operating system installed on your machine. The server and client code is designed and tested to work specifically on Linux.
2. A C compiler, such as GCC installed on your system to compile the server and client code.
3. The server and client code may use the pthread library. Ensure the pthread library is installed in your system.
4. The system needs to be configured with proper network settings.

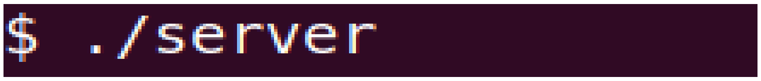
Instructions

1. Open a terminal window.
2. Navigate to the directory where the server and client code files are located.
3. Compile the server code using the command:



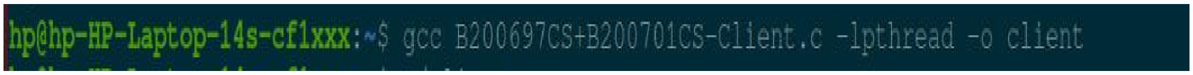
```
hp@hp-HP-Laptop-14s-cf1xxx:~$ gcc B200697CS+B200701CS-Server.c -pthread -o server
```

4. Run the server code using the command



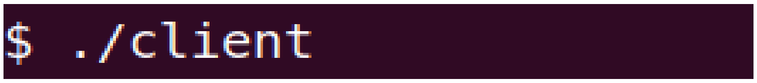
```
$ ./server
```

5. Open another terminal window.
6. Navigate to the directory where the server and client code files are located
7. Compile the client code using the command:



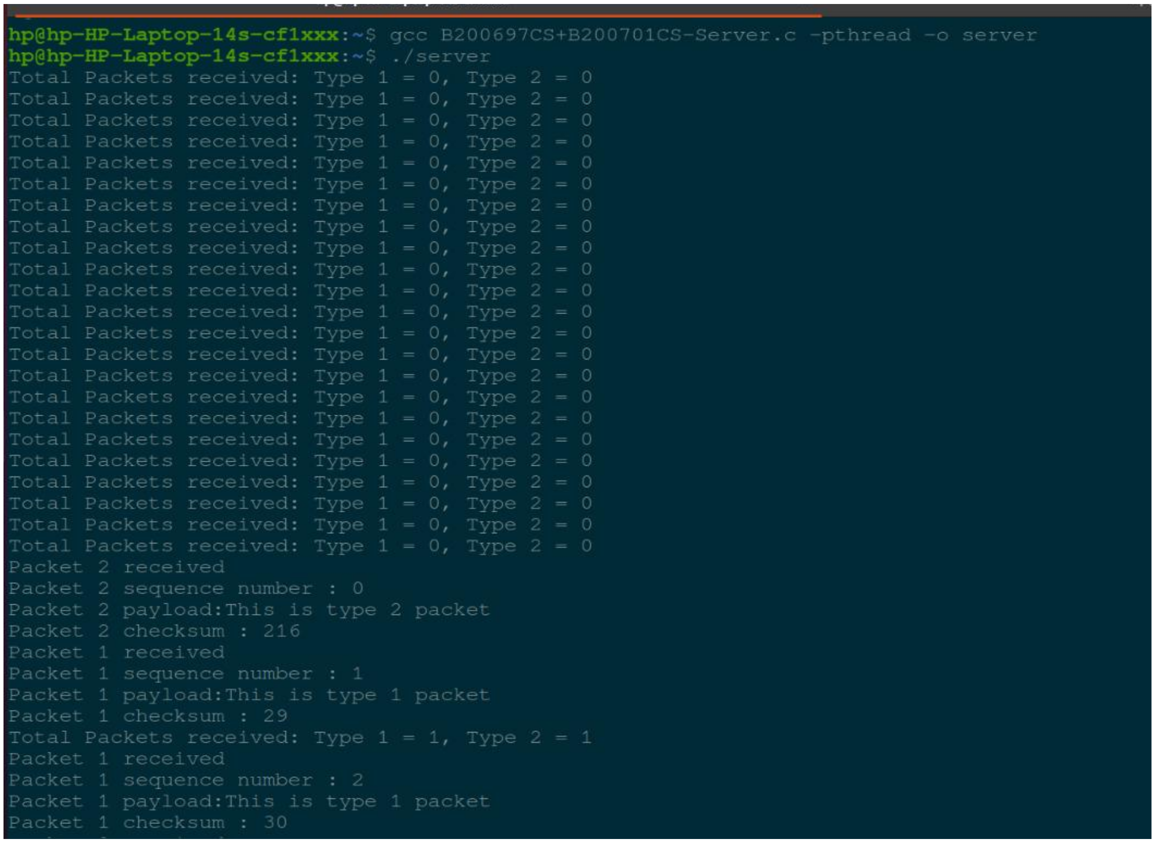
```
hp@hp-HP-Laptop-14s-cf1xxx:~$ gcc B200697CS+B200701CS-Client.c -lpthread -o client
```

8. The server starts to listen for the incoming connections on the specified port.
9. Run the client code using the command



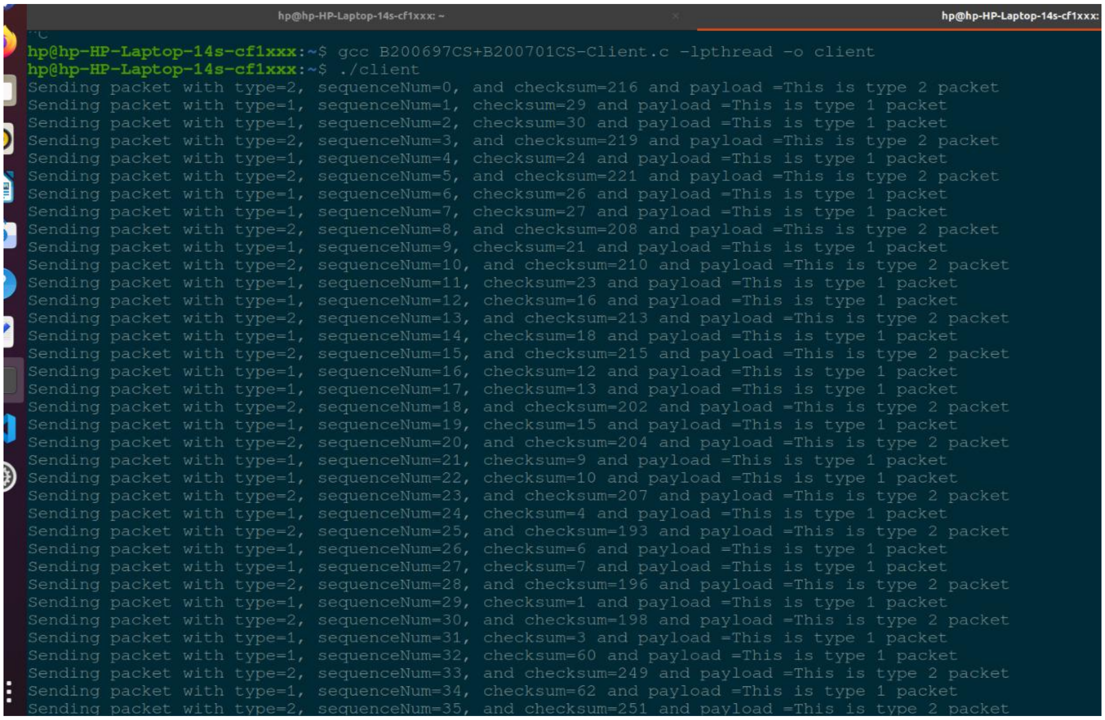
```
$ ./client
```

10. Client process will get connected to the server and starts sending packets to the server
11. Information about sent/received packets, errors and sequence number will be displayed on the console.



```
hp@hp-HP-Laptop-14s-cf1xxx:~$ gcc B200697CS+B200701CS-Server.c -pthread -o server
hp@hp-HP-Laptop-14s-cf1xxx:~$ ./server
Total Packets received: Type 1 = 0, Type 2 = 0
Total Packets received: Type 1 = 0, Type 2 = 0
Total Packets received: Type 1 = 0, Type 2 = 0
Total Packets received: Type 1 = 0, Type 2 = 0
Total Packets received: Type 1 = 0, Type 2 = 0
Total Packets received: Type 1 = 0, Type 2 = 0
Total Packets received: Type 1 = 0, Type 2 = 0
Total Packets received: Type 1 = 0, Type 2 = 0
Total Packets received: Type 1 = 0, Type 2 = 0
Total Packets received: Type 1 = 0, Type 2 = 0
Total Packets received: Type 1 = 0, Type 2 = 0
Total Packets received: Type 1 = 0, Type 2 = 0
Total Packets received: Type 1 = 0, Type 2 = 0
Total Packets received: Type 1 = 0, Type 2 = 0
Total Packets received: Type 1 = 0, Type 2 = 0
Total Packets received: Type 1 = 0, Type 2 = 0
Total Packets received: Type 1 = 0, Type 2 = 0
Total Packets received: Type 1 = 0, Type 2 = 0
Total Packets received: Type 1 = 0, Type 2 = 0
Total Packets received: Type 1 = 0, Type 2 = 0
Total Packets received: Type 1 = 0, Type 2 = 0
Total Packets received: Type 1 = 0, Type 2 = 0
Total Packets received: Type 1 = 0, Type 2 = 0
Packet 2 received
Packet 2 sequence number : 0
Packet 2 payload:This is type 2 packet
Packet 2 checksum : 216
Packet 1 received
Packet 1 sequence number : 1
Packet 1 payload:This is type 1 packet
Packet 1 checksum : 29
Total Packets received: Type 1 = 1, Type 2 = 1
Packet 1 received
Packet 1 sequence number : 2
Packet 1 payload:This is type 1 packet
Packet 1 checksum : 30
```

Server console



```
^C
hp@hp-HP-Laptop-14s-cf1xxx:~$ gcc B200697CS+B200701CS-Client.c -lpthread -o client
hp@hp-HP-Laptop-14s-cf1xxx:~$ ./client
Sending packet with type=2, sequenceNum=0, and checksum=216 and payload =This is type 2 packet
Sending packet with type=1, sequenceNum=1, checksum=29 and payload =This is type 1 packet
Sending packet with type=1, sequenceNum=2, checksum=30 and payload =This is type 1 packet
Sending packet with type=2, sequenceNum=3, and checksum=219 and payload =This is type 2 packet
Sending packet with type=1, sequenceNum=4, checksum=24 and payload =This is type 1 packet
Sending packet with type=2, sequenceNum=5, and checksum=221 and payload =This is type 2 packet
Sending packet with type=1, sequenceNum=6, checksum=26 and payload =This is type 1 packet
Sending packet with type=1, sequenceNum=7, checksum=27 and payload =This is type 1 packet
Sending packet with type=2, sequenceNum=8, and checksum=208 and payload =This is type 2 packet
Sending packet with type=1, sequenceNum=9, checksum=21 and payload =This is type 1 packet
Sending packet with type=2, sequenceNum=10, and checksum=210 and payload =This is type 2 packet
Sending packet with type=1, sequenceNum=11, checksum=23 and payload =This is type 1 packet
Sending packet with type=1, sequenceNum=12, checksum=16 and payload =This is type 1 packet
Sending packet with type=2, sequenceNum=13, and checksum=213 and payload =This is type 2 packet
Sending packet with type=1, sequenceNum=14, checksum=18 and payload =This is type 1 packet
Sending packet with type=2, sequenceNum=15, and checksum=215 and payload =This is type 2 packet
Sending packet with type=1, sequenceNum=16, checksum=12 and payload =This is type 1 packet
Sending packet with type=1, sequenceNum=17, checksum=13 and payload =This is type 1 packet
Sending packet with type=2, sequenceNum=18, and checksum=202 and payload =This is type 2 packet
Sending packet with type=1, sequenceNum=19, checksum=15 and payload =This is type 1 packet
Sending packet with type=2, sequenceNum=20, and checksum=204 and payload =This is type 2 packet
Sending packet with type=1, sequenceNum=21, checksum=9 and payload =This is type 1 packet
Sending packet with type=1, sequenceNum=22, checksum=10 and payload =This is type 1 packet
Sending packet with type=2, sequenceNum=23, and checksum=207 and payload =This is type 2 packet
Sending packet with type=1, sequenceNum=24, checksum=4 and payload =This is type 1 packet
Sending packet with type=2, sequenceNum=25, and checksum=193 and payload =This is type 2 packet
Sending packet with type=1, sequenceNum=26, checksum=6 and payload =This is type 1 packet
Sending packet with type=1, sequenceNum=27, checksum=7 and payload =This is type 1 packet
Sending packet with type=2, sequenceNum=28, and checksum=196 and payload =This is type 2 packet
Sending packet with type=1, sequenceNum=29, checksum=1 and payload =This is type 1 packet
Sending packet with type=2, sequenceNum=30, and checksum=198 and payload =This is type 2 packet
Sending packet with type=1, sequenceNum=31, checksum=3 and payload =This is type 1 packet
Sending packet with type=1, sequenceNum=32, checksum=60 and payload =This is type 1 packet
Sending packet with type=2, sequenceNum=33, and checksum=249 and payload =This is type 2 packet
Sending packet with type=1, sequenceNum=34, checksum=62 and payload =This is type 1 packet
Sending packet with type=2, sequenceNum=35, and checksum=251 and payload =This is type 2 packet
```

Client console

Note

1. Ensure that all the prerequisites are met before running the server and client code. Taking this measure will help avoid any issues or errors during the simulation application.
2. In order to work efficiently during runtime, both the server and client code should include mechanisms for handling errors.