**Lu Lu**

**Program 1 Report**

**Build Script**

Build Server file: g++ -std=c++11 Server.cpp -o Server -lpthread

Build Client file: g++ -std=c++11 Client.cpp -o Client

(Make sure to run server first)

Run Server: ./Server (portnumber) exp(./Server 75118)

Run Client: ./Client (serverName)(portnumber)(20000)(nbufs)(bufsize) (type)

Exp( ./Client csslab3.uwb.edu 75118 20000 15 100 1)

**Client:**

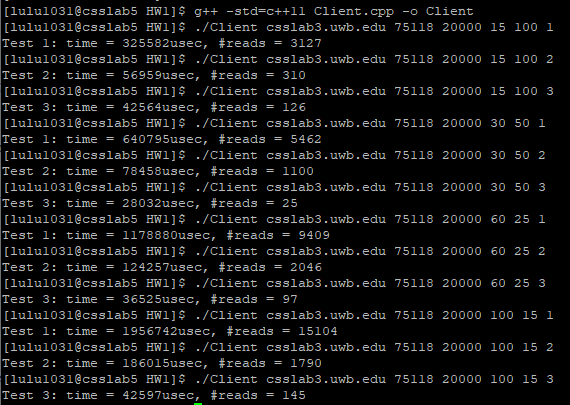
Client program is responsible to connect to the server, then based on the input send data to the server and get it back. After getting input from command prompt, the program will start prepare to buffer. It will connect to the server socket then and create also send data to the server. When the program gets response back from the server, it will print out the time that the data travel.

**Server:**

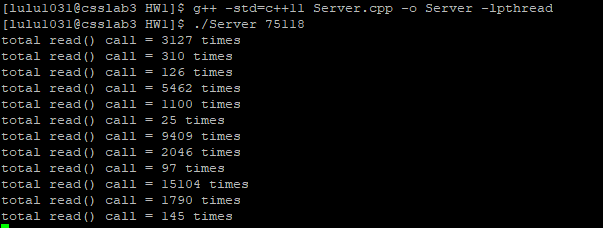
Server is responsible to create a socket that a client can connect to. The main task is to read data from the clients and return how many times that the read operation performed and data receiving time back to client. When the server gets input from the command prompt, it will connect to a socket, then it will start listening for connections. Now, when a client sends a request, it will create a thread to handle. It will read data sent from the client and count the time. After that, it will send the number of read operation and time back to the client.

**Execution output:**

Client side:

****

Server Side:



**Performance evaluation:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test# | type | Number Buffer | Buffer Size | Time | Read |
| 1 | 1 | 15 | 100 | 325582 | 3127 |
| 2 | 2 | 15 | 100 | 56959 | 310 |
| 3 | 3 | 15 | 100 | 42564 | 126 |
| 4 | 1 | 30 | 50 | 640795 | 5462 |
| 5 | 2 | 30 | 50 | 78458 | 1100 |
| 6 | 3 | 30 | 50 | 28032 | 25 |
| 7 | 1 | 60 | 25 | 1178880 | 9409 |
| 8 | 2 | 60 | 25 | 124257 | 2046 |
| 9 | 3 | 60 | 25 | 36525 | 97 |
| 10 | 1 | 100 | 15 | 1956742 | 15104 |
| 11 | 2 | 100 | 15 | 186015 | 1790 |
| 12 | 3 | 100 | 15 | 42597 | 145 |

**Discussion:**

I find that single writes are faster than writev and writev is faster than multiple writes. It gives me the same results too from my previous tests. No matter number of buffer and buffer size, single write is always the fastest.

When the data is split into smaller numbers and bigger buffer size like Test# 1~3, it gives us the fastest time. When the data is bigger and buffer size gets smaller, the time will be slower too.