CSS503
Jesse Leu
2020.06.04

Program 4
Socket Programming

Documentation

Client:

```
Int main(int argc, char* argv[]
{
Check parameters
Set up data and the buffer
Connect and get socket file descriptor
Send repetition to the server
Start the clock
for (int i = 0; i < reps; i++)
{
     Use different writing method depends on the test type to write to the server
}
Stop the clock
Get the read count from the server
Print out the result
Close socket
}
```

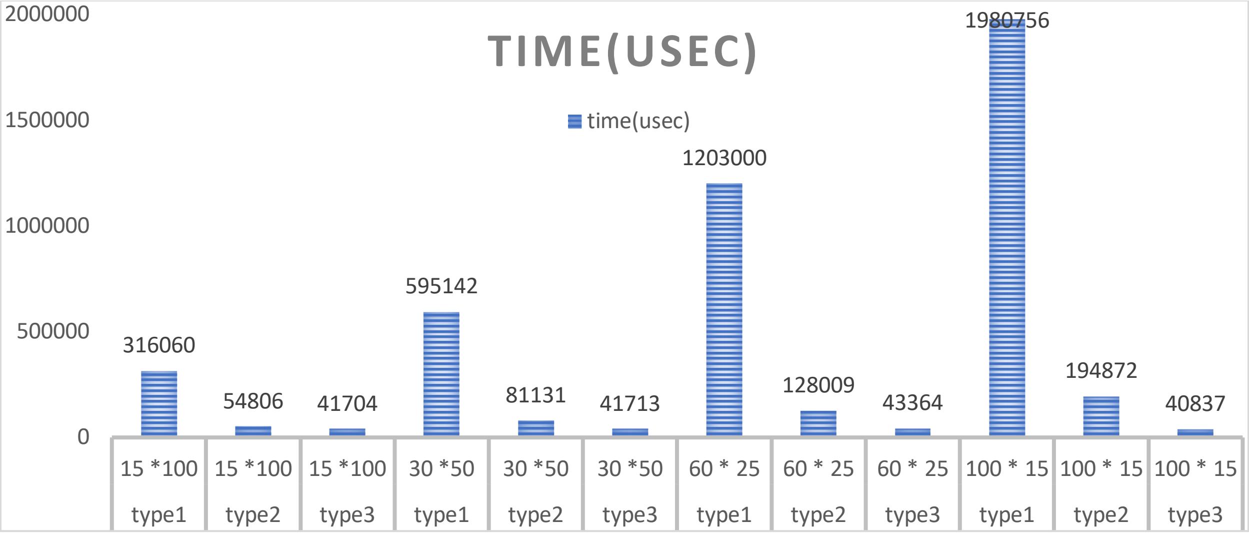
server

```
void* readClient(void* data) //this method is for thread
{

Set up buffer
 Read file descriptor from parameter
 get rep time
set count, totalRead to 0
while(total != BUFFSIZE * rep) // make sure to read the correct amount of data
   {
      Read from client
      Count++;
      totalRead += readByte
   }
    Write the count to client
   Close socket descriptor
   Pthread exit
}
```

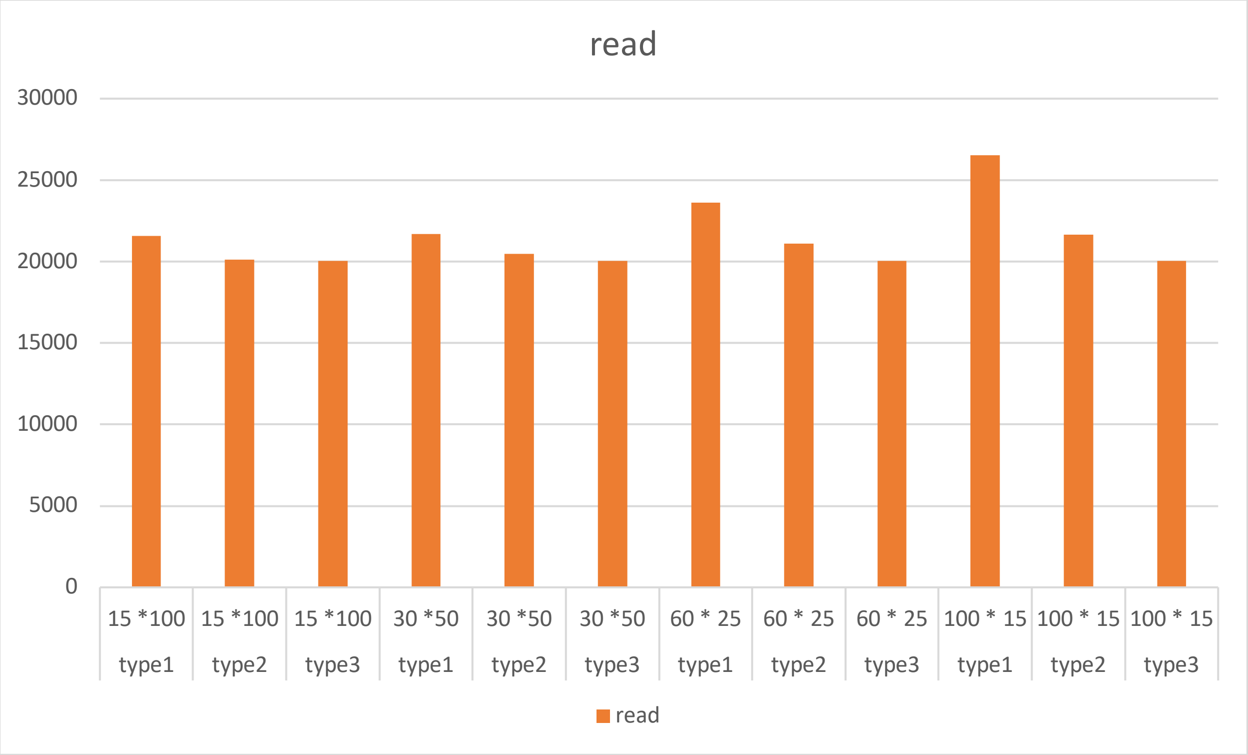
CSS503
Jesse Leu
2020.06.04

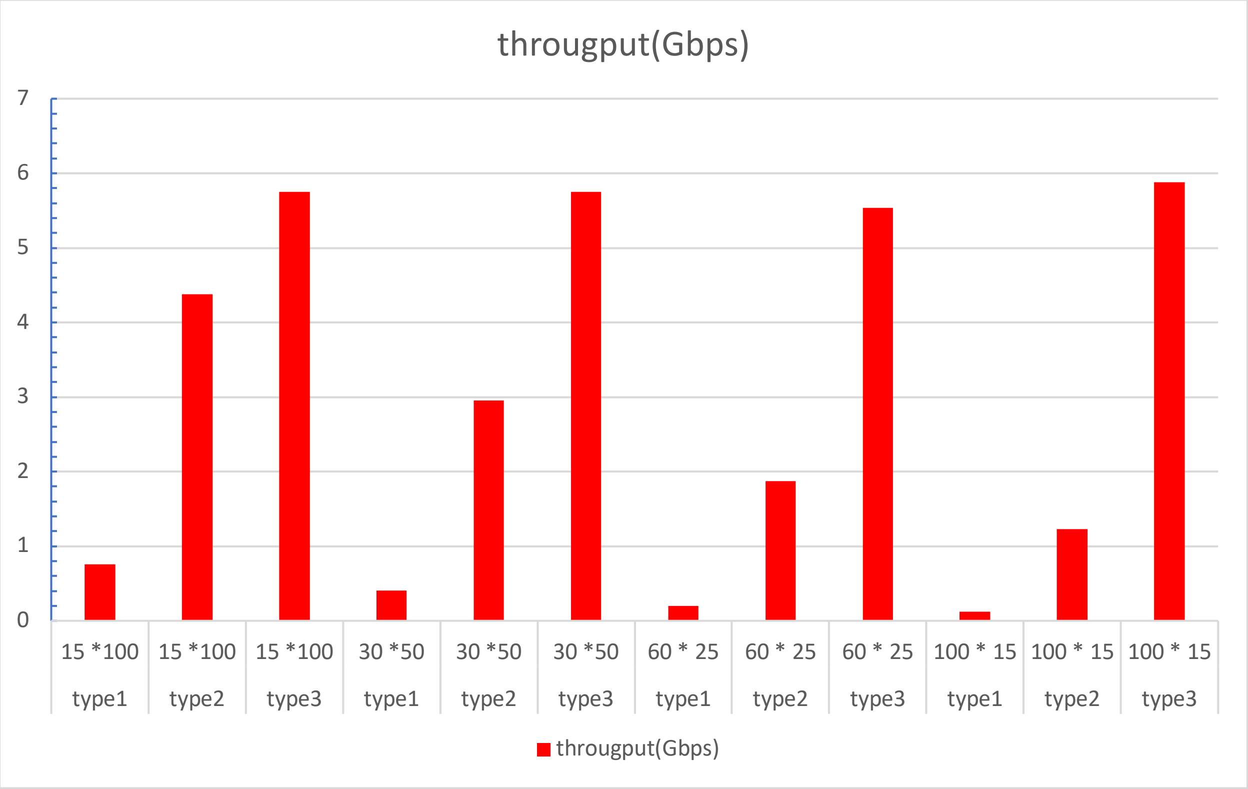
```
int main(int argc, char *argv[])
{
    Check the parameter and set up port



     Build address
     Open socket and bind
     listen and accept

    while(1)
    {
          wait and accept connection
          create a thread and run readClient(void* data)  for the new accept connection

    }

}
```

Performance evaluation



CSS503
Jesse Leu
2020.06.04



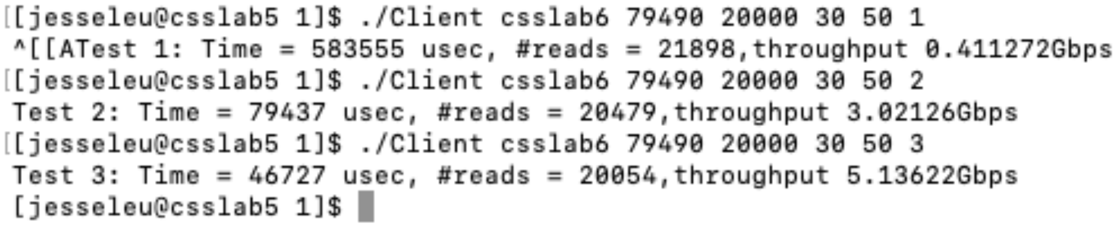


Discussion

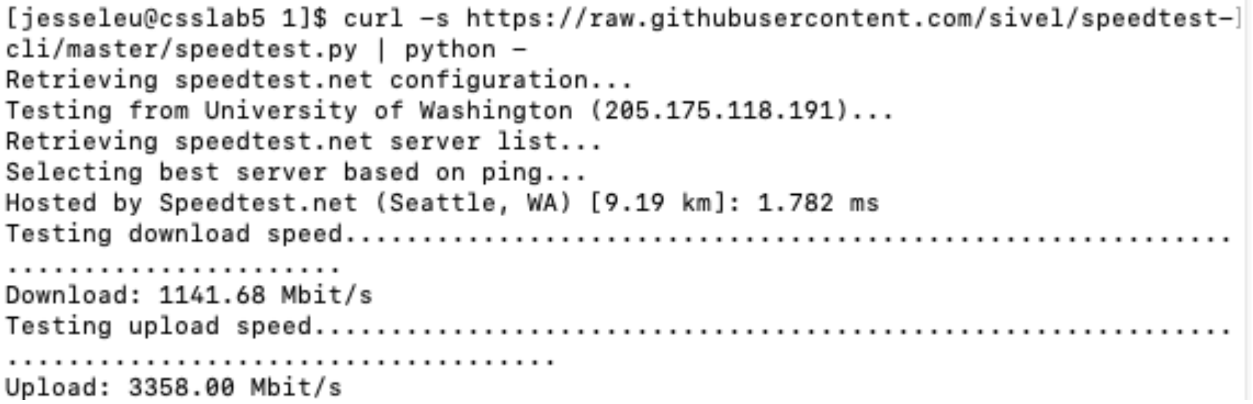
(1) comparing your actual throughputs to the underlying bandwidth

CSS503
Jesse Leu
2020.06.04

If the server and client are both generate at csslab, my actual throughputs can be faster than the speed test result.

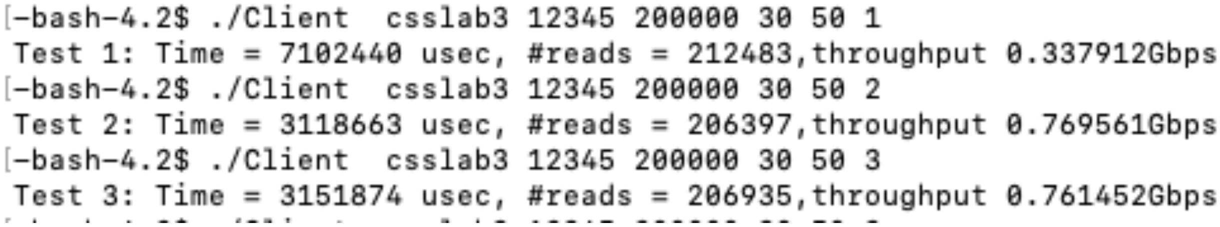


```
[[jesseleu@csslab5 1]$ ./Client csslab6 79490 20000 30 50 1
 ^[[ATest 1: Time = 583555 usec, #reads = 21898,throughput 0.411272Gbps
[[jesseleu@csslab5 1]$ ./Client csslab6 79490 20000 30 50 2
 Test 2: Time = 79437 usec, #reads = 20479,throughput 3.02126Gbps
[[jesseleu@csslab5 1]$ ./Client csslab6 79490 20000 30 50 3
 Test 3: Time = 46727 usec, #reads = 20054,throughput 5.13622Gbps
 [jesseleu@csslab5 1]$ 
```



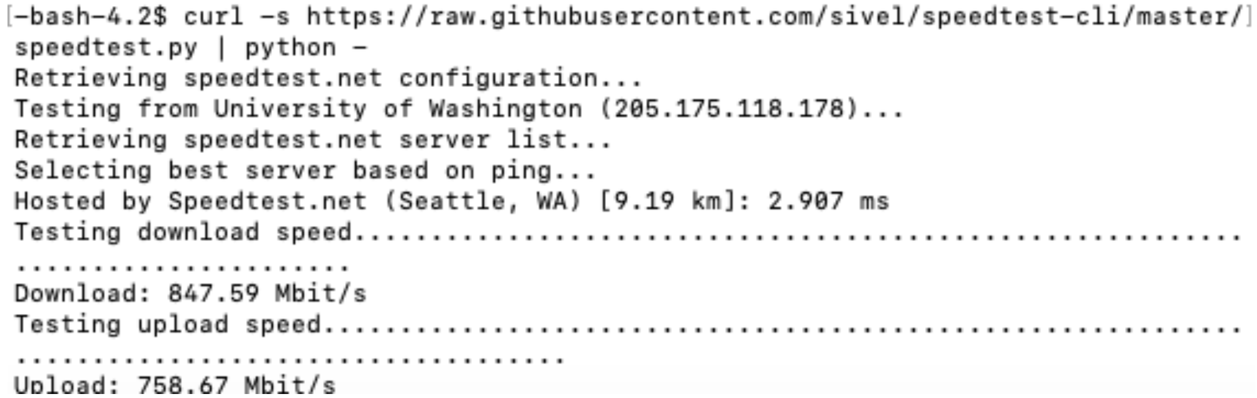
```
[jesseleu@csslab5 1]$ curl -s https://raw.githubusercontent.com/sivel/speedtest-
cli/master/speedtest.py | python -
Retrieving speedtest.net configuration...
Testing from University of Washington (205.175.118.191)...
Retrieving speedtest.net server list...
Selecting best server based on ping...
Hosted by Speedtest.net (Seattle, WA) [9.19 km]: 1.782 ms
Testing download speed................................................................
......................
Download: 1141.68 Mbit/s
Testing upload speed..................................................................
....................................
Upload: 3358.00 Mbit/s
```

However, if I generate my client.cpp at uw1-320-02p while generating my server.cpp at csslab, my actual throughputs are very closed to the speed test result of the uw1-320-02p lab.



```
[-bash-4.2$ ./Client  csslab3 12345 200000 30 50 1
 Test 1: Time = 7102440 usec, #reads = 212483,throughput 0.337912Gbps
[-bash-4.2$ ./Client  csslab3 12345 200000 30 50 2
 Test 2: Time = 3118663 usec, #reads = 206397,throughput 0.769561Gbps
[-bash-4.2$ ./Client  csslab3 12345 200000 30 50 3
 Test 3: Time = 3151874 usec, #reads = 206935,throughput 0.761452Gbps
```

CSS503
Jesse Leu
2020.06.04



```
[-bash-4.2$ curl -s https://raw.githubusercontent.com/sivel/speedtest-cli/master/]
speedtest.py | python -
Retrieving speedtest.net configuration...
Testing from University of Washington (205.175.118.178)...
Retrieving speedtest.net server list...
Selecting best server based on ping...
Hosted by Speedtest.net (Seattle, WA) [9.19 km]: 2.907 ms
Testing download speed..........................................................
......................
Download: 847.59 Mbit/s
Testing upload speed............................................................
....................................
Upload: 758.67 Mbit/s
```

(2) Comparisons of the performance of multi-writes, writev, and single-write performance

As the graphs in the Performance evaluation section show, if other conditions are the same, single-write(fastest) > writev > multi-write(slowest). The greater number of buffers(nbuff) there are, the more obvious the result is even though the total size (nbuf * bufsize) are the same.
single-write’s performances are consistent no matter the change of nbuf and bufsize.
Writev does not change as much as multi-write does when nbuf becomes greater.
Multi-write becomes very slow when the nbuf becomes big and bufsize becomes small.

(3) Comparison of the different buffer size / number buffers combinations.

As the graphs in the Performance evaluation section show, when the number of buffers (nbuf)becomes greater, and the buffer size becomes small, the speed of Writev and multi-write become slower. The result of single-write keep the same when buffer size / number buffers change. In general, we want the buffer size big and reduce the number of buffers, so we can reduce the times of server to read and improve the performance.