

Computer Networks I: CS3530

**Programming Assignment I & II
Socket Programming**

Assignment Report

Sagar Jain - CS17BTECH11034

November 25, 2019

Contents

Assignment I	2
Question I	2
Feature I: Interactive Server & Persistent Client .	2
Points to note about the code	2
Illustration	3
Feature II: File Transfer to Server	3
Points to note about the code	3
Illustration	4
Question II	4
Main Problems & Solutions	4
Illustration	5
Assignment II	5
Main Problems & Solutions	5
Illustration	6
References	6

Assignment I

Question I

Feature I: Interactive Server & Persistent Client

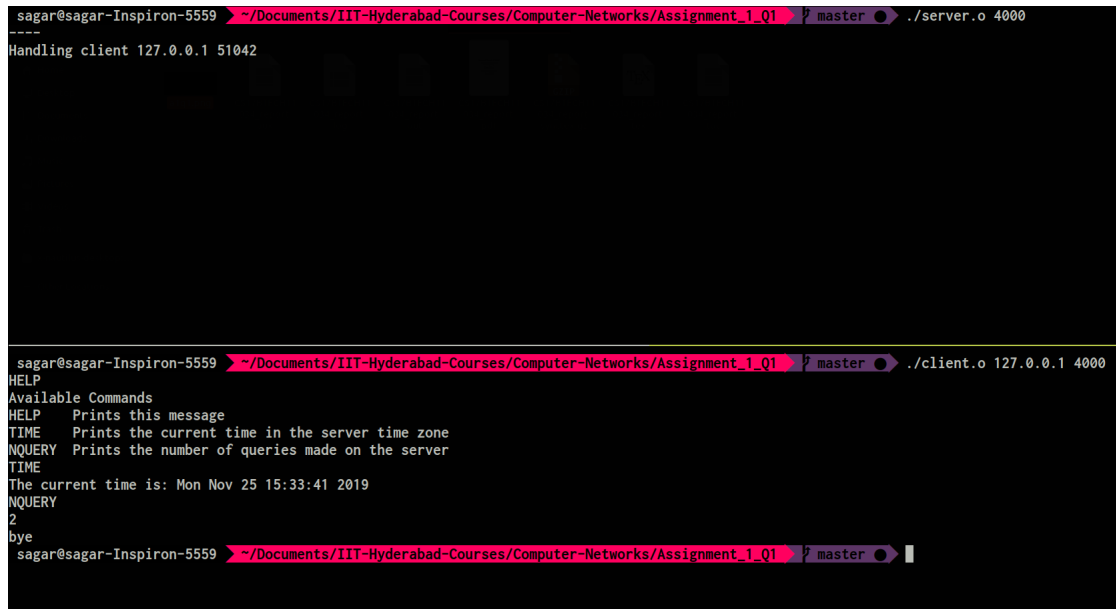
The following has been achieved:

1. The client does not close after sending a single message, the client can keep sending messages and finally quit using *bye*.
2. Clients can use HELP to know all available commands.
3. Clients are allowed to request for the time by typing in TIME.
4. Clients can query the total number of requests using NQUERY.
5. The methodology to create many more such commands is incorporated with just a simple string check and can be seamlessly extended according to the developer.

Points to note about the code

1. The client was made continuous by putting the receive and send in a while loop and clearing the receive and send buffers every iteration.
2. The commands are run on server using string comparisons, and if the receive length is zero it is assumed that the client has disconnected.

Illustration



```
sagar@sagar-Inspiron-5559 ~/Documents/IIT-Hyderabad-Courses/Computer-Networks/Assignment_1_Q1 master ● ./server.o 4000
-----
Handling client 127.0.0.1 51042

sagar@sagar-Inspiron-5559 ~/Documents/IIT-Hyderabad-Courses/Computer-Networks/Assignment_1_Q1 master ● ./client.o 127.0.0.1 4000
HELP
Available Commands
HELP Prints this message
TIME Prints the current time in the server time zone
NQUERY Prints the number of queries made on the server
TIME
The current time is: Mon Nov 25 15:33:41 2019
NQUERY
2
bye
sagar@sagar-Inspiron-5559 ~/Documents/IIT-Hyderabad-Courses/Computer-Networks/Assignment_1_Q1 master ●
```

Feature II: File Transfer to Server

The following has been achieved in this feature.

1. The client can select a file to send, instead of a string to send.
2. The file is received and stored on the server side.
3. New file received will not replace old files.

Points to note about the code

1. The file is sent by copying the file into a string buffer.
2. The file is received into a buffer piece by piece, and put into a file on the server piece by piece.
3. This can happen because sockets work similar to file pointers and move everytime you read from them. Finally when the entire buffer is read the close from client side is read as a 0 length message and the server starts waiting for a new connection.

Illustration

```
sagar@sagar-Inspiron-5559 ~/Documents/IIT-Hyderabad-Courses/Computer-Networks/Assignment_1_Q1 master ● ls | grep rec
✖ sagar@sagar-Inspiron-5559 ~/Documents/IIT-Hyderabad-Courses/Computer-Networks/Assignment_1_Q1 master ● ./fileServer.o 4000
-----
Handling client 127.0.0.1 52812
Receiving File..
Receiving File..
Receiving File..
Receive Complete!
-----
Handling client 127.0.0.1 52814
Receiving File..
Receiving File..
Receiving File..
Receive Complete!
^C
✖ sagar@sagar-Inspiron-5559 ~/Documents/IIT-Hyderabad-Courses/Computer-Networks/Assignment_1_Q1 master ● ls | grep rec
recFile1.txt
recFile2.txt
sagar@sagar-Inspiron-5559 ~/Documents/IIT-Hyderabad-Courses/Computer-Networks/Assignment_1_Q1 master ● |
sagar@sagar-Inspiron-5559 ~/Documents/IIT-Hyderabad-Courses/Computer-Networks/Assignment_1_Q1 master ● ./fileClient.o 127.0.0.1 4000 test.txt
sagar@sagar-Inspiron-5559 ~/Documents/IIT-Hyderabad-Courses/Computer-Networks/Assignment_1_Q1 master ● ./fileClient.o 127.0.0.1 4000 test.txt
sagar@sagar-Inspiron-5559 ~/Documents/IIT-Hyderabad-Courses/Computer-Networks/Assignment_1_Q1 master ●
```

Question II

Chosen mode is hard mode.

Main Problems & Solutions

1. **How to listen and send concurrently**, it is known that while waiting for a `recv` you cannot use a `send` in C, so how do you read a message which has come from another client while the program is expecting input from you or other way how do you give input when the program is actually waiting on a `recv` from the server?

Solution Implemented, the answer to both the above questions that I have implemented is to use `fork` to have one process listen to the server always and other process to take input and send to the server always. I have used **fork** for this task, I could have used **threads** too.

2. To have the **server work with multiple clients**. This was done in the following steps:
 - (a) Provide a maximum number of concurrent clients.
 - (b) Create a **fd_set** for all the clients.
 - (c) Use **select** to check if any of the sockets have any activity.

- (d) If there is activity at the master socket implies it is a new connection.
 - (e) If it is any of the other sockets implies they are either sending a message or have disconnected.
3. How to allow clients to communicate with each other. **A string parsing solution**, Clients can send messages in the format where a vertical line separates client ID and the message.

Illustration

<pre>sagar@sagar-Inspiron-5559 ~/Documents/IIT-Hyderabad-Courses/Computer-Network s/Assignment_1_Q2 master ./server.o 4001 5 New connection , socket fd is 4 , ip is : 127.0.0.1 , port : 38388 Adding to list of sockets as 0 send: Success New connection , socket fd is 5 , ip is : 127.0.0.1 , port : 38390 Adding to list of sockets as 1 send: Success New connection , socket fd is 6 , ip is : 127.0.0.1 , port : 38392 Adding to list of sockets as 2 send: Success 14 New message to 1 19 New message to 2 19 New message to 1</pre>	<pre>sagar@sagar-Inspiron-5559 ~/Documents/IIT-Hyderabad-Courses/Computer-Network s/Assignment_1_Q2 master ./client.o 127.0.0.1 4001 Welcome to CS178TECH11034's Server Your ID is: 0 The following clients are available 0 1 Hello 1 this is 0 Your message was sent</pre>
<pre>sagar@sagar-Inspiron-5559 ~/Documents/IIT-Hyderabad-Courses/Computer-Network s/Assignment_1_Q2 master ./client.o 127.0.0.1 4001 Welcome to CS178TECH11034's Server Your ID is: 1 The following clients are available 0 1 New message from Client: 2 Hi this is 2 2 Hello 2 this is 1 Your message was sent New message from Client: 0 Hello 1 this is 0</pre>	<pre>sagar@sagar-Inspiron-5559 ~/Documents/IIT-Hyderabad-Courses/Computer-Network s/Assignment_1_Q2 master ./client.o 127.0.0.1 4001 Welcome to CS178TECH11034's Server Your ID is: 2 The following clients are available 0 1 2 1 Hi this is 2 Your message was sent New message from Client: 1 Hello 2 this is 1</pre>

Assignment II

Main Problems & Solutions

1. **Get DNS working**, this is straightforward, the code from the pdf works fine to get us the **addrinfo** from which we can get the **sock_addr**.
2. Get server to work with both IPv4 and IPv6, this can be done by mapping all IPv4 addresses to IPv6.
3. Other than this we can just use the code for the basic TCP echo server and client.

Illustration

```
sagar@sagar-Inspiron-5559 ~/Documents/IIT-Hyderabad-Courses/Computer-Networks/Assignment_2 master ● ./server.o 4000
Client address is ::ffff:127.0.0.1
Client port is 52930
HelloIPv4
::ffff:127.0.0.1Client address is ::1
Client port is 38950
HelloIPv6

sagar@sagar-Inspiron-5559 ~/Documents/IIT-Hyderabad-Courses/Computer-Networks/Assignment_2 master ● ./client.o localhost 4000 HelloIPv4
Received: HelloIPv4
sagar@sagar-Inspiron-5559 ~/Documents/IIT-Hyderabad-Courses/Computer-Networks/Assignment_2 master ● ./client.o ip6-localhost 4000 HelloIPv6
Received: HelloIPv6
sagar@sagar-Inspiron-5559 ~/Documents/IIT-Hyderabad-Courses/Computer-Networks/Assignment_2 master ●
```

References

For All

1. <https://linux.die.net/man/>

For Assignment 1, Question 2

1. https://www.tutorialspoint.com/cprogramming/c_file_io.htm
2. <https://stackoverflow.com/questions/238603/how-can-i-get-a-files-size-in-c>
3. <https://www.geeksforgeeks.org/c-program-demonstrate-fork-and-pipe/>

For Assignment 2

1. https://www.ibm.com/support/knowledgecenter/ssw_ibm_i_72/rzab6/xacceptboth.htmCS5060_NP.pdf