COURSEWORK TWO

AE1CSF – Computer Fundamentals

Dr Eugene Ch'ng

CONTACT INFORMATION

Dr Eugene Ch'ng
Email: eugene.chng@nottingham.edu.cn
Room: SEB417

Coursework Two Information 35%:

Issue Date: 15 November 2016
Completion Date: 5pm 2 December 2016

A Simple Client-Server Information Retrieval

You are to implement a client-server information retrieval system using the Linux Berkeley Sockets Application Programming Interface (API). The server provides two-part information that the client can request from: 1) contents of a text file, 2) server information. The client connects to the server, gives commands based on the server protocol, and retrieves necessary information.

The Server (Total: 25%)

- We shall assume that the server can only accept one client connection at any one time
- The server should be able to read from a text file and return the contents by line number, with the protocol (10%):
  - FILE #
  - To return line 1 of the file for example: FILE 1
- The server should be able to return specific commands given by the client with the protocol (15%):
  - To return the current date and time of the server: "GET TIME"
  - To return the IP address of the server: "GET IP"
  - To return the hard disk space: "GET SPACE"
  - To return the memory information of the server "GET MEMORY"
  - To return the OS/System information (OS type, versions, etc.) of the server "GET SYS"
  - Plus two more pieces of information

Notes:

- The name of the text file in the server, which the client requests with the FILE command is up to you to specify.
- You may use either Linux system commands or C APIs to get the information required.
- The IP address must not be hardcoded, you must use certain API, or commands to acquire the IP address, or any other pieces of server information. Hardcoded text will get 0 marks.

The Client (Total 5%)

- The client shall connect to a server with "./clientYourStudentID [IP address]
- The client should be able to provide commands to the server with "FILE [line number]" and "GET [type]"
- The client should be able to disconnect from the server with "QUIT"

Additional Requirements (Total 5%):

- Every lines of code should be properly commented.
- Your client should have the naming convention client[NameAbbrev][Student ID] for both the source and executable
  - Example: clientMDL6547289.c and clientMDL6547289
- Your server should have the naming convention serv[NameAbbrev][Student ID] for both the source and executable
  - Example: servMDL6547289.c and servMDL6547289
- You should also submit a A4 PDF report of 500 words describing how you implemented the code on both the client and the server side.
- The server text file must also be submitted
- All files above should be zipped into an archive named clientserverMDL6547289.zip

Typical client screen when connected to the server:

```
~]$ ./client 10.178.1.73
zxxxxx@cs-linux ~]$ -- You are connected to the server!—
zxxxxx@cs-linux ~]$ [SERVER MESSAGE]: "Welcome to doubi server!"
zxxxxx@cs-linux ~]$ [Your command]: FILE 2
zxxxxx@cs-linux ~]$ [SERVER RESPONSE]: line 2 – "I think therefore I am…"
zxxxxx@cs-linux ~]$ [Your command]: GET TIME
```

```
zxxxxx@cs-linux ~]$ [SERVER RESPONSE]: The date is "Mon Nov 23 09:33:04 CST 2015"
zxxxxx@cs-linux ~]$ [Your command]: QUIT
zxxxxx@cs-linux ~]$ -- you have now disconnected from the server --
```

Note that this is the simplest form of communication between the client and the server, you may add your own textual response, and perhaps even emoticons. You may add aesthetics and format the server textual response so that the information is clearer. You may add ASCII characters too:

Welcome to my Home Server! __̴ı̴̴̡̡̡ ̡͌l̡̡̡ ̡͌l̡*̡̡ ̴̡ı̴̴̡ ̡̡͡|̲̲̲͡͡͡ ̲▫̲͡ ̲̲̲͡͡π̲̲͡͡ ̲̲͡▫̲̲͡͡ ̲|̡̡̡ ̡ ̴̡ı̴̡̡ ̡͌l̡̡̡̡.___

```
OoOoOoOoOoOoOoOoOoOoOoOoOoOoOoOoOoOoOoOoOoOoOoOoOoOoOoOoOoOoOoOoOoOoOoOoOoOoOoOoO
8                                                                              8 O
8                           WELCOME TO MY SERVER!                              8 O
8                                                                              8 O
OoOoOoOoOoOoOoOoOoOoOoOoOoOoOoOoOoOoOoOoOoOoOoOoOoOoOoOoOoOoOoOoOoOoOoOoOoOoOoOoO
```

DETAILS OF SUBMISSION

- Standard university penalty applies for late submission – 5% deduction from coursework mark for <u>every</u> working day the coursework is late
- Format of deliverable:
    - Coursework Two – to submit a report in PDF (A4 size) detailing your applications (at least 500 words excluding code explanation) together with your code (C source files) and Linux executable (client and server with folders clearly marked as first and second part).
- Submission is online through Moodle
- Feedback within 21 days of hand-in date