Monday, 11 September 2017

EE 324, Programming Assignment #1

A simple client/server program

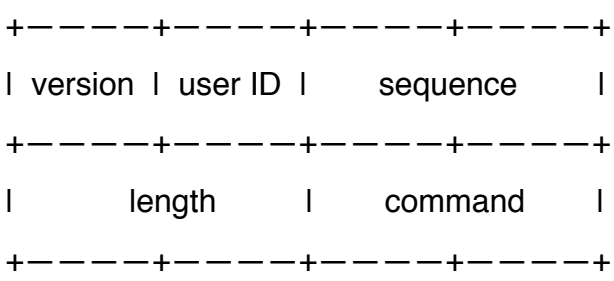
1. Overview

You need to design and implement a simple client and a server application. They are communicating with a custom protocol based on TCP/IP. The server waits a connection trial from the client with port 12345, and the client will ask the server to conduct several jobs.

NOTE: You don't need to employ a **non-blocking I/O**, which means that there is only one client for the server.

2. Protocol Specification

2.1 Header format



total = 8 bytes

- version: static value, it should be 0x04
- user ID: static value, it should be 0x08
- sequence: initially generated by a client, and the client should increment this value for the following packets in the same session
- length: total packet length (including the header)
- command: client command for the server job
  - 0x0001 - hello (client hello)
  - 0x0002 - hello (server hello)
  - 0x0003 - data delivery (server should receive the whole data from the client)
  - 0x0004 - data store (save the received data to a file, name should be explicitly specified)
  - 0x0005 - error (if there is any error)

0x0001, 0x0003, 0x0004 commands should be sent by the client

0x0002 command should be sent by the server

0x0005 command can be sent by any peer

- NOTE: The field "sequence", "length", and "command" should be transferred as network byte order (i.e. big endian).

3. Scenario

- Run your server.
- Run your client with the server's ip address, port, and file name (i.e. **"./client 127.0.0.1 12345 test.txt"**).
- After a TCP connection is set up between the client and the server, the client should send "client hello (0x0001)", and then the server should reply with "server hello (0x0002)".
- The client sends "data delivery (0x0003)" with the contents of the file.
- The server temporarily stores the contents into the memory (buffer).
- If the file transfer is complete, the client sends "data store (0x0004)" with the file name.
- Then, the server stores the contents of memory as a file.

4. More Requirements

- A client and a server should be different processes.
- You need to use "AF_INET" for connection (but you can test your program in a single machine).
- A client program should read a file and send to a server.
- A server should save a file with the received contents from the client.
- You don't need to consider a binary file transfer since TAs will test your program using **"*.txt"** files.
- If data delivery or connection set up fails, "error (0x0005)" should be delivered.
- Your program should be named as **"client"** for your client and **"server"** for your server.

5. Instruction for Submission

- You will be submitting one taball file to the KLMS website.

- Create a tarball (tar.gz) with all the source codes, README, and executables.
  - Tarball structure: (for program assignment 1)
    - Create a folder called “p1"
    - Put your source code in the folder named “p1/src"
    - Put your executable(s) in the folder named “p1/bin”
    - Put your make file “Makefile” in the root path - “p1/“
      - Make sure that your Makefile correctly complies your source code
    - readme.txt file goes to the root path - “p1/“
      - No README, No points; let us know how to run your program
    - install.sh goes to the root path as well - "p1/" (optional)
      - But, if your program needs a third party library, you should provide a script “install.sh”
    - Then, compress the entire "p1" folder into a single tarball; the name will be **“P1_yourIDnumber.tar.gz”**
  - Write the concise comments in the source code to understand your program.

! **IMPORTANT 1: Please strictly follow the above structure and name policy; if not, TAs will not evaluate your program.**

! **IMPORTANT 2: Please make sure that there is no compile and execution error. TAs will not give you any score in case of the problems.**

6. Test Case

1) TAs will check whether your program clearly sends/receives packets with the specified packet header (with Wireshark) - 50 points
   I. connection set up phase will be checked (i.e. client hello -> server hello) - 20 points
   II. all header values will be carefully checked - 30 points
      I. version & user ID - 5 points
      II. sequence - 5 points
      III. length - 5 points
      IV. command - 10 points
      V. network byte order - 5 points
2) TAs will check whether your client sends a file (**4MB**) to a server - 25 points

3) TAs will check whether your server stores a file at a server side (with the client command = 0x0004) - 25 points

7. Test Environment

- Language: C or C++
- Test O/S: **Ubuntu 14.04.5 LTS (Trusty Tahr) 64bit**
    - http://releases.ubuntu.com/14.04/
- If you need a VM, download using the following link.
    - http://nss.kaist.ac.kr/ee324.ova
    - username: ee324, password: ee324
    - Import the VM using VMware or VirtualBox
- **NOTE: We will not consider your compilation and execution problems due to the different OS versions.**

8. Due Date

- **11:59 PM, Sep. 29, 2017 (Friday)**
- The website automatically marks the time of the last submission/modification. (The website often becomes unavailable, so please make sure that you submit your assignment early)
- **10% late penalty per day**
- Hard deadline: **11:59 PM, Oct. 03. 2017 (Tuesday)**
    - We will not receive additional submissions after the hard deadline.
    - Exceptions: documented medical/personal emergency
- Please DO NOT e-mail Prof. Shin or TAs to submit your assignments.

9. Plagiarism

- You can discuss with your colleagues, but you should turn in your own programs

✓ Copy and Paste

- Will run plagiarism detection on source code
- “Copy and paste” codes will get severely penalized
- If detected, 0 point for all assignments (both providers and consumers)
- But you will have a chance to defend yourself

10. Questions?

- Please email TAs to ask any questions.
  - Junsik Seo (js0780@kaist.ac.kr)
  - Jinwoo Kim (jinwoo.kim@kaist.ac.kr)