CS558 Assignment 1 (C/C++/Java)

Due 11:59pm Feb. 28th (Sunday)

This assignment is done individually.

Goal :

1. Learn how to write and use makefile

   http://www.delorie.com/djgpp/doc/ug/larger/makefiles.html (C)

   http://www.cs.swarthmore.edu/~newhall/unixhelp/javamakefiles.html (Java)

2. Implement an ftp client and an **iterative** server using TCP sockets. Following commands should be supported: ls, lls, cd, pwd, mkdir, get and quit.

3. Implement the Autokey cipher.

Specifications

The ftp server is invoked as

ftpserv <server_port> (C/C++)

java FtpServ <server_port> (Java)

<server_port> specifies the port at which ftp server accepts connection requests.

The sftp client is invoked as

ftpcli <server_domain> <server_port> (C/C++)

java FtpCli <server_domain> <server_port> (Java)

<server_domain> is the domain name of the server **(i.e., bingsuns.binghamton.edu)**. If you use C/C++, you can use **gethostbyname()** (http://retran.com/beej/gethostbynameman.html) to convert the domain name into the corresponding IP address.

<***server_port***> refers to the port number on which server is listening.

Upon connecting to the server, the client prints out **ftp >**, which allows the user to execute the following commands.

ftp > ls //lists contents of the directory where the executable server code is

ftp> lls //lists contents of the directory where the executable client code is

ftp > mkdir <dir> //create a directory <**dir**>

ftp > cd <dir-relative-path> //change the current working directory to a path that is relative to //the current directory. E.g. cd path

ftp > cd .. //move up one folder

ftp > pwd //print the working directory of the server

ftp> get <filename> //transfer <filename> from the server to the client

ftp> quit

Implement the Auto key cipher **(key: security)**. When the client requests to transfer the file <filename> from the server using **get <filename>**, the server encrypts **<filename>** using

Autokey cipher and stores the encrypted file in **<filename>_se**. The server then transfers the encrypted file to the client. After the client receives the file, the client saves the received file in **<filename>_ce** and then decrypts the file. The decrypted file is stored in **<filename>_cd.**

Note:

1. If you use C, you can use cli.c and ser.c provided on the blackboard for this assignment. If you use Java, you can use TCPServer.java and TCPClient.java provided on the blackboard for this assignment. The instruction for compiling and executing these two files are given in cs558-01.pdf.

2. Since multiple students will be testing their servers, it is possible that multiple students end up using the same port number. To prevent this, use a unique port number such as last 4 digits of your ID.

3. No error handling is required, e.g. you dont need to check whether a command is valid or whether a file exists.

Submission guideline

- You need to hand in your **source code** and a **Makefile** electronically (**do not submit .o or executable code**). You must make sure that this code compiles and runs correctly on bingsuns.binghamton.edu. If you use C/C++, the Makefile **must** give the executable server code the name **ftpserv** and the executable client code the name **ftpcli**. If you use Java, the Makefile **must** generate files **FtpServ.class** and **FtpCli.class**.
- Write a **README** file (text file, do not submit a .doc file) which contains
  - You name and email address.
  - Whether your code was tested on bingsuns.
  - How to execute your program.
  - (Optional) Briefly describe your algorithm or anything special about your submission that the TA should take note of.
- Place all your files under one directory with a unique name (such as p1-[userid] for assignment 1, e.g. p1-pyang).
- Tar the contents of this directory using the following command.
  tar –cvf [directory_name].tar [directory_name]
  E.g. tar -cvf p1-pyang.tar p1-pyang/
- Use the Blackboard to upload the tared file you created above.

Grading guideline

- ls, lls, quit: 20’
- cd, mkdir, pwd: 30’
- File transfer & autokey cipher: 36’
- Readme, correct format of execution (correct executable names, the # of arguments): 6’
- Correct Makefile: 8’

Academic Honesty:

All students should follow Student Academic Honesty Code **(if you have not already read it, please read it carefully).** All forms of cheating will be treated with utmost seriousness. You may discuss the problems with other students, however, you must write your OWN program. Discussing algorithms and code is NOT acceptable. Copying an assignment from another student or allowing another student to copy your work __http://www.delorie.com/djgpp/doc/ug/larger/makefiles.html (C)

http://www.cs.swarthmore.edu/~newhall/unixhelp/javamakefiles.html (Java)
may lead to the following:

- **Report to the department and school**
- **0 in the assignment or F in this course.**

Moss will be used to detect plagiarism in programming assignments. You need ensure that your code and documentation are protected and not accessible to other students. Use **chmod 700** command to change the permissions of your working directories before you start working on the assignments. If you have any questions about whether an act of collaboration may be treated as academic dishonesty, please consult the instructor before you collaborate.