CS558 Assignment 3
Due: 11:59pm April 10 (Monday)

This assignment is done individually. You can use C/C++/Java/Python in this assignment.

You will implement a simple secure file transfer protocol (sftp) using the **Secure Socket Layer (SSL)**. SSL is used to establish a secure connection between the server and the client.

Specifications

Your will implement an sftp server and an sftp client. The sftp server is an iterative server (i.e., the server will serve one client at a time).

The sftp server has at least one argument <server_port>, which specifies the port number at which the server accepts the connection request from the client. The port number should be a user-defined number between 1024 and 65535. As multiple students will test their programs on remote.cs, please use a unique port number (e.g., the last 4 digits of your B number) so that it is different from other students' port numbers.

The sftp client has at least two arguments: <server_domain> and <server_port>. <server_domain> is the domain name of the machine on which the server is running. There are seven remote.cs machines: remote01-07. cs.binghamton.edu. For example, If the server runs on remote01.cs.binghamton.edu, then the domain name of the server is remote01.cs.binghamton.edu. When we test your program, we may execute your server and your client on the same or different remote.cs machines. <server_port> is the port number of the server. You can add other arguments to the client and the server if needed. If you use C/C++, your program needs to convert the domain name to the corresponding 32-bit IP address using **gethostbyname** function (https://man7.org/linux/man-pages/man3/gethostbyname.3.html).

If you use C, your Makefile should generate an two executables **sftpserv** and **sftpcli**. If you use Java, your Makefile should generate files **Sftpserv.**class and **Sftpcli.class**. If you use python, your python files should have name **sftpserv.py** and **sftpcli.py**.

For example, if you use C/C++, the sftp server can be invoked as:

./sftpserv ***3479***

The sftp client can be invoked as (if the sftp server is running on remote01):

./sftpcli ***remote01.cs.binghamton.edu 3479***

Upon connection, the sftp client prompts the user to enter his/her ID and password. The server maintains a file ***password*** which has the following content and format:

alice 12345
bob 67890

Where alice and bob are IDs, and 12345 and 67890 are their passwords. The ID and the password are separated by a space. In this assignment, you do not have to encrypt/hash the password (in practice, the password file should be protected using encryption, hashing and access control mechanisms). After the server receives the ID and the password, the server verifies the ID and the password. If both the ID and the password are correct, then the server sends “correct ID and password” to the client. Otherwise, the server sends “incorrect ID and password” to the client and the client prompts the user to enter the ID and password again.

Next, the client prints “sftp **>”**, which allows the user to execute the put, lls, and exit

commands. **sftp > put <filename>** // transfer a **text file** <filename> from the client to the

server. **sftp > lls** // list files and sub-directories of the directory on the client side

sftp > exit // terminates the sftp session (to make it easier for the TA to grade the assignment, please terminate both the client and the server in this assignment)

If a user enters a command other than the above three commands, then the client prints “Invalid Command” and “sftp >”, prompting the user to enter a different command.

You need to generate a **public key certificate for the sftp server** to establish the ssl connection. You do not have to generate a certificate for the client. You can use commands or programs to generate the certificate. The certificate generated should be stored as a file, which will be included in the assignment submission. For example, if you use C, you can use the following command to generate the certificate file “cert.pem”.

```
openssl req -x509 -newkey rsa:4096 -keyout key.pem -out cert.pem -days
```

365 If you use java, you can use keytool to generate the certificate.

To compile your C program, please use the following commands:

```
gcc -Wall -w -o sslcli sslcli.c -I/usr/local/ssl/include/ -L/usr/local/ssl/lib -lssl -lcrypto
gcc -Wall -w -o sslserv sslserv.c -I/usr/local/ssl/include/ -L/usr/local/ssl/lib -lssl
-lcrypto
```

Please note that, you can use any code available on the web for SSL connection. However, you must write your own code to implement put, lls, and exit. You should also generate the certificate for the server by yourself. To make it easier for the TA to grade your program, please do not use passphrase when generating the certificate.

Grading guideline

- Readme: 4'
- Makefile (C/C++/Java): 8'
- SSL connection (C/C++/Java): 43’
- SSL connection (python): 51

- put: 30'
- lls: 10’
- exit: 5'

Submission guideline

- Create a directory with a unique name (e.g. p3-[userid]), which contains
  - A makefile,
  - A README file
  - The file “password”, which contains the following two lines

    alice 12345
    bob 67890
  - a subdirectory “server”, which contains the source code of the server and the server’s certificate
  - a subdirectory “client”, which contains the source code of the client

- The TA will use a program to parse the README (text file, please do not submit a .doc file). Please use the following format for your README:
  - On the first line, your name (do not include a prefix like "Name:")
  - On the second line, your email (do not include a prefix like "Email:")
  - On the third line, the language, either "c", "c++", "java", "python2", or "python3" (do not include a prefix like "Language:")
  - On the fourth line, an empty line
  - On the subsequent lines:
    - Whether your code was tested on remote.cs.binghamton.edu.
    - How to build the program.
    - How to execute the program.
    - If any software is needed to be installed on remote.cs to execute your program, please give the commands for installing the software on remote.cs
    - (Optional) Briefly describe anything special about your submission that the TA should take note of.

-Tar the contents of this directory using the following command.

```
tar –cvf p3-[userid].tar p3-[userid]
```

E.g. tar -cvf p3-pyang.tar p3-pyang/

- Upload the tared file you create above to brightspace.

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 codes and solutions. Discussing solutions to the problem is NOT acceptable. Copying an assignment from another student or allowing other students to copy your work may lead to a 0 in the assignment or an F in the 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.