Homework
Project Specifications - February

Computer Systems & Programming
Academic Year 2023-2024

Overview

The project shall be carried out ***individually***: it must constitute ***an original creation*** , therefore it is not possible to share parts of the code with other students/groups, or copy contents deriving from other sources.

The project shall be submitted:

- via email to the teacher's official email
- at least by midnight of D-6
- not earlier than one week before the deadline (see previous bullet)
- in a tar file including:
  - the source file
  - a Makefile or script for compilation
  - a README file to guide the installation, compilation and **test**

Project Specifications - I

- **Objective: implement a "simple FTP server", i.e. a server that lets clients read and write files from its own (remote server) disk**
- Connection should be stateful: the client connects to the server, sends a set of requests and closes the connection (hint: use stream sockets)
- The server should be able to manage an unlimited number of clients, with minimal delay to establish a connection (hint: use `fork()` to create a dedicated process server for each client)
- The shared portion of the server file system is one and the same for all clients

Project Specifications - II

- The client executable should accept the following option on the command line:
  - IP address or DNS name of the server
- When started, the client should immediately connect to the server on a well-known port
- Supported client commands: `ls`/`get`/`put` (same syntax as SFTP, no options – see `man sftp`) using only relative pathnames on the server and the client
- The client should wait for a new command from its standard input and send it immediately to the server
- The client should loop forever (i.e., till it reads an `exit` command)

Project Specifications - III

- The server should correctly manage file ownership and access rights, i.e.:
  - require client authentication with user and password, as defined in the server
    - password encryption not required
  - file access should be subject to client rights for all operations (read, write, create, delete)
    - hint: use `setuid()`
  - root access should not be allowed
  - define a “home” directory for each user

Project Specifications - IV

- The server executable should wait for input on a well-known port
- The server should shutdown when it receives a user-selected signal or a `quit` command from the command line.
- The server executable should accept the following options on the command line (default values should be stored in a configuration file):
  - root directory of the shared portion of file system
- **Extra bonus (I)**: implementing any of the following SFTP commands provides and extra bonus: `pwd` or `cd`/`lcd` pair or `mkdir`/`rmdir` pair
- **Extra bonus (II)**: provide automated tests (see slide on correctness)

Implementation & Discussion

The project consists of a C language program that satisfies the specified requirements, using the library calls ***that are part of the course program***. The use of other calls is not accepted. If in doubt, ask the teacher.

The project code must correctly compile and execute in the required software environment (compiler version, kernel version, clib version)

Once the project has been submitted and a sufficient grade has been obtained, it is mandatory to show up at the immediately following exam session to take the exam.

The oral exam consists of a discussion on design choices and software implementation of the project. Taking inspiration from the project work, questions may be asked on any topics that are part of the course program.

Correctness

The project shall successfully pass (at least) **ALL of the following test cases**:

1. Server startup (start the server only)
2. Client connect (after (1))
3. Client file get and put (after (1))
4. Second client file get and put (after (3), start another client that gets/puts other files from the server)
5. Client shutdown (after (4))
6. Server shutdown (after (5)).

Evaluation Criteria

Prerequisite: the code must correcly compile, link and start on Ubuntu 22.04.1 (gcc version 11.4.0). If it does not, **you cannot take the oral exam**.

Correctness of the code: main evaluation element that determines (alone!) the passing of the exam.

Error handling: it is an integral part of the correctness of the code!

Modularity and readability of the code: division into functions, comments, function and variable names (sic!), etc...

Quality of documentation: user manual, software architecture, README file, project report.