CS 371 –Project

Deadline: 04/17/2019

Design of a network private remote folder

Maximum points 100

General information

The project is an INDIVIDUAL effort, groups can be allowed only by explicit authorization of the instructor.

The project report and the code should be submitted on Canvas as a single Zip file by 04/17.

The project should be discussed with the TA during the week of the 04/20 – 04/24. Contact the TA to set up an appointment. The discussion will occur through Zoom.

NOTE: Compared to the previous version of the project main differences are:

1. The project is an individual effort.
2. The required part should handle only a single client and thus no concurrent operations.
3. Optional extensions are available for those that want to extend the project with those functionalities.
4. The evaluation on upload/download speed can be run on a single computer. No need to compare wireless vs. wired.

Project overview

Remote folders are common and useful applications enabled by the cloud computing paradigm. In these applications, a user can store his/her files in a folder that resides on a server. The user has access to common functionalities such as download, upload, delete, and see the whole content of the folder.

The objective of this project is to realize a client-server application for a private remote folder. The project will also include an evaluation of upload/download performance in different settings.

Project design

The project should be realized through socket programming, thread programming, and use of semaphores to handle concurrent operations.

CLIENT

The client will run on a shell through an interpreter able to execute the following operations:

CONNECT server_IP_address server_port: This function opens a connection with the server on the specified port.

UPLOAD filename: the client uploads on the server the prespecified file.

DOWNLOAD filename: the client downloads the specified file on a local folder. If the file does not exist or the file is uploading or deleting, the server returns an error.

DELETE filename: the specified file name is deleted.

DIR: returns the content of the shared folder, i.e., file name, size, upload date and time, number of downloads.

The client should also collect important statistics for the evaluation, such as the upload and download data rate (e.g., MB/sec). This information should be shown in real time (e.g., by updating it every few seconds) in textual or graphical form. It should also be saved on a text file to realize graphs to be shown in the report.

SERVER

The server will be realized through a process accepting an incoming connection and handling the commands coming from the client. Handling only a single client is required.

Evaluation

The evaluation is structured in two parts. The first part is meant to show the correct implementation of the client functionalities, the second part is devoted to the network performance.

Client functionalities

The evaluation should show, through appropriate screenshots, all basic operations listed above. As an example, the client should be able to execute upload a file. Screenshots should show the command line on the client side, and the file correctly uploaded on the server side.

Network performance

The second part of the evaluation is devoted to study the network performance, i.e. upload/download speed.

You should produce two graphs. Each of this graph has x-axis time and y-axis speed (e.g., Mb/sec). The graphs are the following 1) network download speed, 2) network upload speed. Consider sufficiently large files (at least several hundred of megabytes or more) to make sure you have enough data for 1min of upload/download or longer.

This part can be evaluated by running client and server on the same computer if two are not available.

The results should be discussed, and meaningful explanations should be given. Do not just say in words what the graph shows but discuss “why” the results look that way.

Report

A report should be prepared. It must include at least the following sections:

- Abstract: Summary the overall report
- Introduction and motivation: Overview of shared folders and project design

- Project design and Implementation: Pseudo-code, description and rationale of the proposed solution, description of the methodology used for implementing the proposed solution (e.g. relevant functions and data structures, structure of the program, etc.).
- Experiments: Description of the experiments performed and obtained results. Results should be represented in a graphical form (snapshots, graphs) as well as discussed in the writing.
- Conclusions: Summary of the work and final considerations.

Optional part

The extensions to the project described below are optional. If an optional part is addressed, it should be discussed in the project report and it should be run during the meeting with the TA.

Each optional part is worth 10 points.

1) Graphical interface. The clients can operate through a standard graphical interface of a folder and execute the operations using the mouse or keyboard.

2) Subfolders. Clients will have the possibility of creating, deleting, and navigating, subfolders on the server. Opportune functionalities should be added on the client side.

3) Authentication. The server will enforce an authentication procedure through username/password to allow clients to log in and access/modify the content of the folder.

4) Shared folder: The server should be realized as a process running multiple threads. This will allow accepting new connections and handling multiple concurrent clients. The server also needs to ensure that concurrent operations are handled correctly. As an example, more than one client may try to upload the same file (same name) in the folder. The Server needs to be able to handle such conflicts.

Resources that may helpful:

https://www.cs.rpi.edu/~moorthy/Courses/os98/Pgms/socket.html