Assignment 3: FTP Server

CIS 457 Data Communications - Winter 2024

Posted on April 3, due on April 17 (11:59 pm)

1 Objectives

Most network applications rely on the file transfer protocol (FTP) in one form or the other. For instance, the HTTP protocol used on the Web is a generic file transfer protocol. In this assignment, you will work with your teammates (up to four members per team) and implement an FTP client program and an FTP server program for a simple file transfer. The server could handle one or more file transfers to the client(s) at any given time. The implemented FTP application supports text file transfer. The client program presents a command line interface that allows a user to:

- Connect to a server.
- List files stored at the server.
- Download (retrieve) a file from the server.
- Upload (store) a file from the client to the server.
- Terminate the connection to the server.

The server program binds to a port and listens for requests from a client. After a client connects to the server, the server waits for commands. When the client sends a terminate message (quit), the server terminates the connection and waits for the next connection.

2 Implementation

You can implement your project using Python as two independent programs, an FTP client called **ftp_client** and an FTP server called **ftp_server**. The ftp_client program presents a command line interface. The communication between the client and the server needs to be implemented using TCP sockets. The client should be able to send FTP commands to the server. Upon receiving a command, the server should parse the command and perform the appropriate action. The format of the commands is as follows:

1. `CONNECT <server name/IP address> <server port>`: This command allows a client to connect to a server. The arguments are the IP address of the server and the port number on which the server is listening for connections.
2. `LIST`: When this command is sent to the server, the server returns a list of the files in the current directory on which it is executing. The client should get the list and display it on the screen.
3. `RETRIEVE <filename>`: This command allows a client to get a file specified by its filename from the server.

4. `STORE <filename>`: This command allows a client to send a file specified by its filename to the server.

5. `QUIT`: This command allows a client to terminate the control connection. Upon receiving this command, the client should send it to the server and terminate the connection. When the ftp_server receives the quit command, it should close its end of the connection.

3 Deliverables

Only one copy of the deliverables is required from each team. The designated person responsible for submission should upload to Blackboard the following:

- The complete code.
- A report that describes what problems you have encountered and how you solved them.
- Screenshots demonstrating how each service (connect, list, retrieve, store, and quit) provided by the FTP server works as required.
- A `README.TXT` file that briefly describes each file, how to compile the file(s), and how to run the file.
- These files should be placed in a directory called `<team name>-asgmt3`. Create a unique team name that helps you stand out from the rest.
- Use tar command to place all the files in a single file called `<team name>-asgmt3.tar`. Assuming you are in the directory `<team name>-asgmt3` do the following:
  - Goto the parent directory: `cd ..`
  - tar the files:
    `tar -cvf <team name>-asgmt3.tar ./<team name>-asgmt3`
  - Verify the files have been placed in a tar file:
    `tar -tvf <team name>-asgmt3.tar`
  - Compress the files using gzip: `gzip <team name>-asgmt3.tar`
  - Verify that the gzipped file exists: `ls <team name>-asgmt3.tar.gz`