FILE TRANSFER APPLICATION
ELEN4017 Project Report

Kayla-Jade Butkow (714227) and Jared Ping (704447)

School of Electrical & Information Engineering, University of the Witwatersrand, Private Bag 3, 2050, Johannesburg, South Africa

Abstract:

Key words:

1. INTRODUCTION

The File Transfer Protocol (FTP) is essential in the implementation of a File Transfer Application. The file transfer protocol allows for the transfer of files between two end systems [1].

2. SYSTEM OVERVIEW

2.1 FTP Server

Unimplemented Features:

2.2 FTP Client

The FTP client runs from the user's local host and allows the user to interact with the FTP server in order to transfer files. In order to improve user experience, a client with a graphical user interface (GUI) was implemented.

The client allows the user to specify the FTP server address that they wish to connect to, as well as the port that the server is running on. The user is also able to input their username and password for the FTP server they are connecting to.

Once the user has successfully connected to the FTP server, they are able to view their local file system as well as the remote file system within the client GUI. The user is also able to navigate both file systems. Once the required file is found, the user is able to upload the file to the remote server from the local file system, or download the file from the remote system to the local storage. When uploading a file, the file is saved to the currently selected directory on the server. If a directory has not been selected by the user, the file is saved to the home directory of the user's remote repository. Likewise, when the user is downloading a file, the file is saved to the current local folder, or if none is selected, to the user's home directory. On Mac OS X operating systems, this home directory is found at */Users/Username*.

The user also has the ability to delete files or folders, as well as to recursively remove a folder and all of its contents.

Once the client has finished using the FTP connection, they can disconnect from the server and connect to another server if they wish to.

Unimplemented Features: The feature to change the file structure from file to record or page was not implemented. This was not implemented since the implementation was complex and deemed unnecessary since any file can be transfered using the file structure. Furthermore, since the file structure is the default type, any server that the client wishes to interact with will be compatible with the file structure type [2]. The client also does not allow the user the opportunity to change the transmission mode from stream to block or compression. Once again, since stream mode is the default mode, any FTP server must accept stream mode, meaning that implementation of the other types in unnecessary [2]. The client also does not have implementation to allow the user to append data onto the end of an existing text file.

** Rename from and to if not implemented

3. COMMANDS AND REPLY CODES

4. IMPLEMENTATION DETAILS

The server and the client were both implemented using Python 3. On both systems, all communication sockets are created using the Python `socket` module. Interfacing with the operating system is performed using the `os` module. This module allows for the traversing of paths in the operating system, as well as for saving and opening files.

4.1 Server

4.2 Client

In order to connect to the server, once the user has supplied the server address and port, a TCP connection is created between the server and the client. This TCP connection acts as a control connection to transfer FTP commands and replies between the client and the server [1]. The user's username and password are then transferred over the control connection to authenticate the user on the server.

5. DIVISION OF WORK

6. RESULTS

7. CRITICAL ANALYSIS

7.1 Server

7.2 Client

Limitations: Since the append command is never sent from the client, if the user tries to upload a file with a name that already exists in the current directory, the preexisting file will be overwritten. This could result in the accidental loss of the user's data.

8. CODE STRUCTURE

9. CONCLUSION

REFERENCES

[1] J. Kurose and K. Ross. *Computer Networking. A Top-Down Approach*, p. 51. Pearson Education, sixth ed., 2013.

[2] J. Postel and J. Reynolds. "File Transfer Protocol (FTP)." *Network Working Group*, oct 1985.

Appendix