SANKARI BALASUBRAMANIYAN
SR1

TREE FTP REPORT

TP-1 SR-1

OBJECTIVE

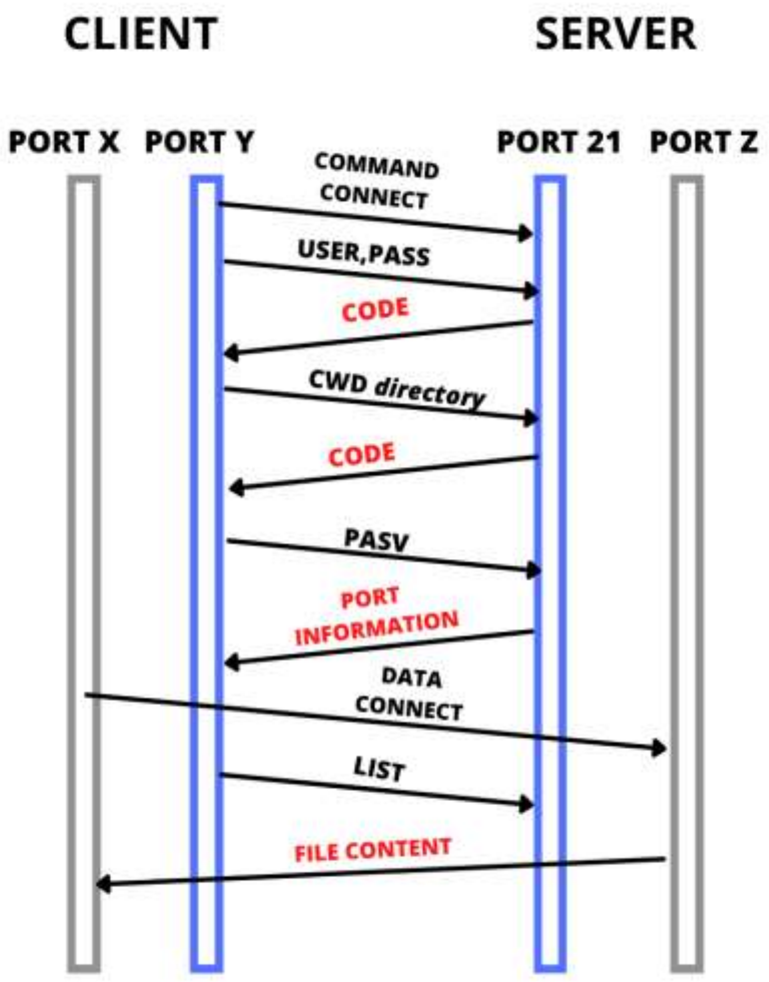
List the directories and files present in the given path of an FTP server

- Establish Client, server connection via telnet/TCP protocol
- Communicate commands, status and data between Client and Server.

THEORY

- FTP RFC 959 is a protocol used to transfer files between hosts. And it is based on client-server architecture.
- Default port the server listens to 21, which is utilised in this project.
- There are two separate channels – control and data channels
    1. **Control channel** : Is used to communicate ASCII based commands and status codes, phrases, establish authorisation, browsing directory listing and so forth, and it is usually kept open till the end of the session.
    2. **Data channel** : supports transfer of files between host and it is closed after every transfer.
- FTP protocol helps in maintaining the state of program, hence the user can browse or list the current director without the entire path of the directory as and when the session proceeds.
- FTP supports two different modes of operation ACTIVE and PASSIVE modes, Passive mode is used for the purpose of this project (since active mode work with firewalls or NAT traversal).

CODE EXPLANATION



- Initially the client (or the user) establishes a data connection with the server using the server's IP and default port (21) which the server listens to.
- The client requests for authorisation with username and password using the ASCII command USER and PASS, for the server responds back with a code and phrase - the status of verification.
- If the authorisation process is smooth, the data connection is ready for the session and the client asks for a change in directory which it wants to browse by sending CWD command along with the directory.
- After the directory is changed, the client request to connect in passive mode for data connection using the command PASV.
- The server responds back with the IP and Port number it will be listening to.
- Client establishes a data connection with the given server details

- With the control connection the client requests to list the contents of the current directory.
- The server responds back with the contents via data connection and the data socket is closed.

PROGRAM STRUCTURE

The project has the following classes and methods

- **Main Class** Launches the main function
    1. **main** function : gets the arguments (address, port, directory) from the user and calls to establish control channel. Closes the sockets and pipelines by the end of the program
    2. **getClient** : returns the Client_ServerFTP object, purpose is to maintain a single class object for the entire session.
- **Client_ServerFTP Class :** establish the client to FTP server connection
    1. **controlConnect** : gets the address, port, username and password as the argument and connects to FTPserver to maintain a control channel for communicating ASCII commands and status codes
    2. **dataConnect** : gets the name/path of the directory the user needs to be listed. Establishes a connection to the FTP server via passive mode and maintains a data channel to receive the list of files/directory sent by the server.
    3. **dataAddress** : returns the address of the server which client uses to establish data connection.
    4. **dataPort** : returns the port of server which the server listens to.
    5. **closeSocket** : close control socket
    6. **closeCommand** : close command and data pipelines.
- **FTP_Tree class** : print directories in the form of tree
    1. **listTree** : finds the type and name of given directory and calls necessary function to print.

2. **directory** : prints the given directory name in colour blue and call dataConnect function to get the contents in the directory.
3. **link** : prints link name maintaining indentation in colour cyan.
4. **file** : prints file name maintaining identation in colour red.

EXECUTION

- The project is build with maven hence it's a compulsory prerequisite
- **PARAMETERS**
    1. **ADDRESS** : Address of the server the user wishes to connect
    2. **PORT NUMBER** : Port number which the server listens to
    3. **DIRECTORY** : Path of the directory the user wishes to tree of.

  By default the parameters are set as
    1. **ADDRESS** : ftp.ubuntu.com
    2. **PORT NUMBER** : 21
    3. **DIRECTORY** : /

  The project also runs without any parameters, hence none of the parameters are mandatory. (The Main function has to be modified just a little bit to get username and password from the user).

Execution examples

1. java -cp ./target/FTPTree-0.0.1-SNAPSHOT.jar org.bisp.mavenproject.FTPTree.Main
2. java -cp ./target/FTPTree-0.0.1-SNAPSHOT.jar org.bisp.mavenproject.FTPTree.Main ftp.free.fr 21 /cdimage/focal

KEY DEVELOPMENT POINTS

- The program complies with common project guidelines.
- The program works with FTP servers like ftp.ubuntu.com, ftp.free.fr etc.
- The supports an added parameter which gets the directory path from the user and displays its tree. So any directory's tree can be displayed.
- The program adheres to the colour code of ubuntu tree command.