CSE 6324

**Advanced Topics in Software Engineering**

Project 1: Project Report

Fall 2016

Team 5

Mrunalini Vijay Aghor

Tanvi Prakash Nachane

Tirumala Prasad

**Contents**

**1. Introduction**

1.1 File transfer protocol

1.2 Data and control connection in FTP

1.3 Active and passive FTP

**2. High level design**

**3. Minimum requirement implementation**

**1. Introduction**

**1.1 File transfer protocol (FTP)**

File transfer protocol is the commonly used protocol for exchanging files over the internet. FTP often uses client server architecture. FTP promotes sharing of files via remote computers with reliable and efficient data transfer.

**1.2 Data and control connection in FTP**

FTP uses two ports Data and control port.

1.2.1 Control connection

The first communication part in client and server takes place through Control connection. The client sends the first request to server through command channel.

1.2.2. Data connection

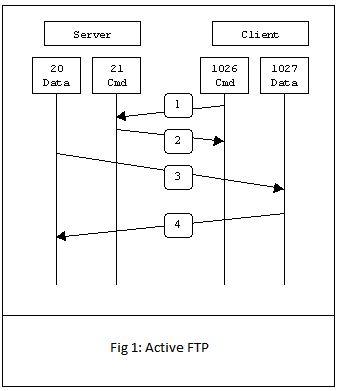
This port is used to transfer any data between client and server. The data transferred may be a part of files an entire file or number of files between that particular server and client.

**1.3 Active and Passive FTP**

There are two modes in FTP, Active FTP and passive FTP.

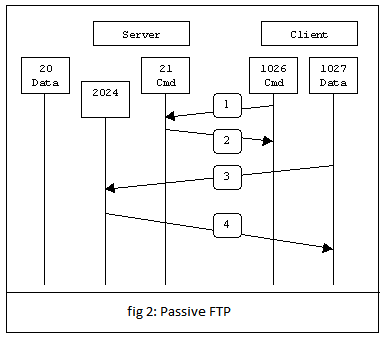
1.3.1 Active FTP

In Active mode client connects from a random port (N) greater than 1023 to server’s command port number 21. Then the client starts listening to port N+1 to the FTP server. The server will then connect back to the clients specified data port from its local data port which is port 20.



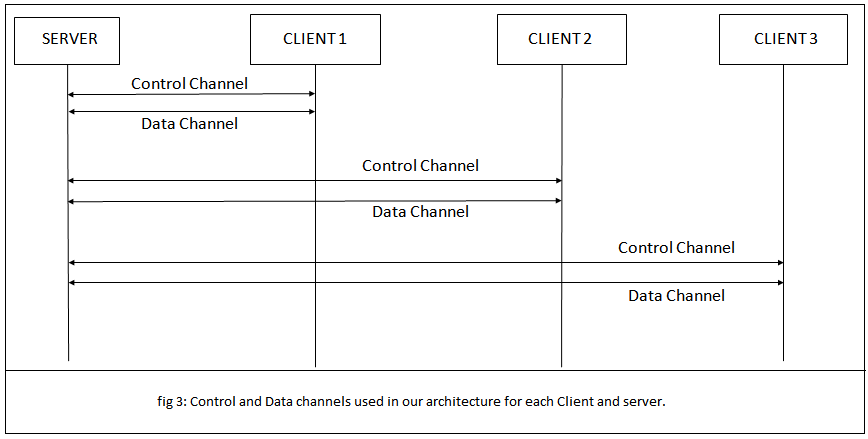
1.3.2 Passive FTP

In passive mode FTP the client initiates both connections to the server. When opening an FTP connection, the client opens two random unprivileged ports locally (N > 1023 and N+1). The first port contacts the server on port 21, but instead of then issuing a PORT command and allowing the server to connect back to its data port, the client will issue the PASV command. The result of this is that the server then opens a random unprivileged port (P > 1023) and sends P back to the client in response to the PASV command. The client then initiates the connection from port N+1 to port P on the server to transfer data.

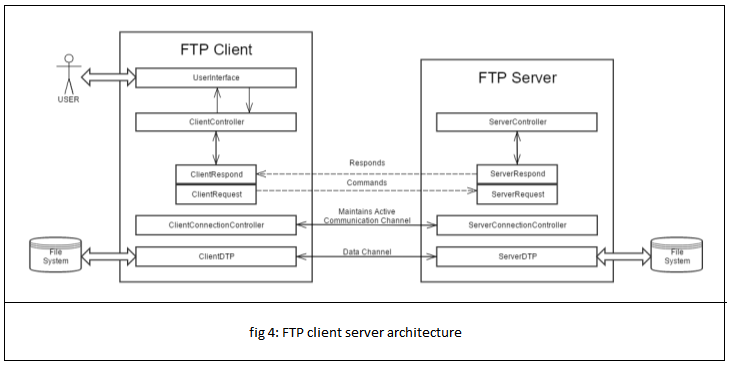


**2. High level design**

The Architecture of our system is as follows,



Architecture Diagram of our system is as follows,



**3. Minimum requirement implementation**

In order to make FTP workable without needless error messages, the following minimum implementation is required for all servers.

3.1 NOOP

This command does not affect any parameters or previously entered commands. It specifies no action other than that the server send an OK reply.

3.2 STORE (STOR)

This command causes a server to accept the data transfer via data connection and to store the data as a file at the server site. If the file exists on server site then its content shall be replaced by the data being transferred. A new file is created at the server site if the file specified in the pathname does not already exist.

3.3 STORE UNIQUE (STOU)

The resultant file is to be created in the current directory under a name unique to that directory.

3.4 RETRIEVE (RETR)

The file on the server is transferred or copied to the client side on the specified pathname. The status and contents of the file at the server site shall be unaffected.

3.5 FILE STRUCTURE (STRU)

The default structure used is file.

3.6 TRANSFER MODE (MODE)

The default transfer mode we use is Stream.

3.7 REPRESENTATION TYPE (TYPE)

3.8 DATA PORT (PORT)

3.9 LOGOUT (QUIT)

This command terminates a USER and if file transfer is not in progress, the server closes the control connection. If file transfer is in progress, the connection will remain open for result response and the server will then close it.

3.10 USER NAME (USER)

The argument field is a Telnet string identifying the user. This will be the first command when control connections are made. Client needs to enter the credentials to connect to the server. A new folder is created for each client.