FTP

an overview

B. P. Blackshaw

Enterprise Distributed Technologies
London, UK

Table of Contents

1. Introduction

FTP (File Transfer Protocol) is a well established Internet protocol designed to transfer files (and information about files) across networks using TCP (Transmission Control Protocol).

FTP is defined in the Request For Comments 959 document (RFC 959), which can be obtained from the Internet Engineering Task Force (www.ietf.org).

FTP requires a client program (FTP client) and a server program (FTP server). The client can fetch files and file details from the server, and also upload files to the server. The server is generally password protected.

FTP commands are initiated by the client, which opens a TCP connection called the control connection to the server. This control connection is used for the entire duration of a session between the client and server. A session typically begins when the client logs in, and ends when the quit command is sent to the server. The control connection is used exclusively for sending FTP commands and reading server replies - it is never used to transfer files.

Transient TCP connections called data connections are set up whenever data (normally a file's contents) is to be transferred. For example, the client issues a command to retrieve a file from the server via the control channel. A data connection is then established, and the file's contents transferred to the client across it. Once the transfer is complete, the data connection is closed. Meanwhile, the control connection is maintained.

2. Active and passive modes

Data connections may be set up in two different ways, active and passive. Note that active and passive refer to the operation of the FTP server, not the client.

1. Passive mode

In passive mode, the client sends a PASV command to the server. This tells the server to listen for a connection attempt from the client, hence the server is passively waiting. The server replies to PASV with the host and port address that the server is listening on. The client deciphers this reply and when a data connection is required, attempts to initiate the connection to the server at this address.

2. Active mode

In active mode, the server actively connects to the client. To set up active mode, the client sends a PORT command to the server, specify the address and port number the

client is listening on. When a data connection is required, the server initiates a connection to the client at this address.

Generally the server is responsible for closing data connections.

1) FTP Commands

FTP commands sent across the control connection consist of simple text strings (and follow the Telnet protocol - see RFC 854). For example, to retrieve a file, the client sends "RETR filename" on the control connection to the FTP server. To transfer a file, the client sends "STOR filename".

The FTP server acknowledges each command with an FTP reply, which consists of a three digit number followed by human-readable text. The first digit indicates if the response is good, bad, or incomplete. If an error occurred, the second digit may be used to indicate what type of error occurred. Similarly, the third digit can indicate more details of the error.

The first digit is the most important, and the five possible values are described below:

1yz	Positive Preliminary reply. The request action has been initiated, but another reply is to be expected before the client issues another command
2yz	Positive Completion reply. The requested action has successfully completed, and the client may issue another command
3yz	Positive Intermediate reply. The command has been accepted, but more information is required. The client should send another command in reply.
4yz	Transient Negative reply. The command failed, but it can be retried
5yz	Permanent Negative Completion reply. The command failed, and should not be repeated.

2) Sample scenarios

Example 1

For example, to change directory the client sends:

```
CWD dirname
```

The server responds with:

```
250 CWD command successful
```

As the reply begins with a ‘2’, we know the command sequence is completed.

However if we attempt to change directory to one that does not exist:

```
CWD nonexistentdir
```

The server responds with:

```
550 nonexistentdir: The system cannot find the file specified.
```

As the reply begins with a ‘5’ we know that the command failed, and that it will fail again if repeated (unless the missing directory is created on the server).

Example 2

To transfer a text file, we issue a ‘RETR’ command to the server. However to transfer the file we require a data connection to be set up. This can be done using active or passive mode.

As discussed previously, in active mode, the client sends a ‘PORT’ command, indicating what address and port number the server should connect to:

```
PORT 192,168,10,1,4,93
```

The first four digits are the IP address, and the last two encode the port number (in two 8-bit fields, the first being the high order bits of the 16-bit port number).

The server responds with:

```
200 PORT command successful.
```

This indicates that the data connection has been established.

Next, the ‘RETR’ command is issued:

```
RETR abc.txt
```

The server responds with:

```
150 Opening ASCII mode data connection for abc.txt(70776 bytes).
```

The reply begins with a ‘1’, so we know that the command was successful, but the server will be sending another reply – the client cannot issue another command until this is received.

Eventually, the server sends:

```
226 Transfer complete.
```

The command sequence is complete, the file has been transferred, and the client can issue another command.

See RFC 959 for details about the second digit, and more extensive examples.

Note that most standard command-line FTP clients support debug mode, which displays the FTP commands that are being sent to the server, and the reply strings that are received back. Typing “debug” at the prompt will usually put the client into debug mode.

3) Data types

The two most common data types in usage are ASCII and binary.

ASCII is the default data type, and is intended for the transfer of text files. A line of text is followed by <CRLF>. Note that different operating systems use different end of line terminators.

Binary type (known as IMAGE in FTP) is used to transfer binary files. A byte-by-byte copy is made of the source file. Graphical images, video and executable files are all binary files. If they are transferred as ASCII, they will be corrupted.

4) Session commands

To begin an FTP session, the USER command is sent to the server:

```
USER javaftp
```

The server responds with:

```
331 Password required for javaftp.
```

The client must respond with the password:

```
PASS mypassword
```

The server responds with:

```
230 User javaftp logged in.
```

The session is now established, and the user can begin issuing various file and directory-related commands.

To end the session, the client sends:

```
QUIT
```

The server responds with:

```
221
```

The session is now closed, and any further attempt to send commands to the server will fail.

5) File commands

FTP supports numerous file-related commands.

Files can be deleted (DELE) and renamed (RNFR,RNTO) as well as stored (STOR) and retrieved (RETR). When a file is stored, it can be written over or appended to (APPE).

See the Sample scenarios examples for more details.

6) Directory commands

FTP supports a variety of directory-related commands.

Directories can be created (MKD), removed (RMD), or changed into (CWD, CDUP).

Two types of directory listings can be made with FTP.

The LIST method obtains a list of files (and possibly directories). If a directory is specified, the server returns a list of files in the directory, together with system specific information about the files. The file information sent will vary depending on the server system. The data type should be set to ASCII for this file name list. If no directory is specified, details of the current working directory listing are sent.

The NAME LIST (NLST) method is similar to LIST, but only file names are returned. No other information about the files is sent. Again, the data type should be set to ASCII.

7) References

RFC 959. File Transfer Protocol. J.Postel. J Reynolds. 1985.
http://www.ietf.org/rfc/rfc0959.txt