Computer Networks 159.334

File transfer exercise using FTP

A. L. C. Barczak/2013

Part 1

This exercise makes use of the FTP protocol directly from a command line application. You have probably used FTP before, but in a transparent way (perhaps via a browser).
FTP is an acronym used with different meanings, each with its set of commands:
FTP *protocol* commands (e.g., USER, PASV, PORT etc)
FTP *application* commands, associated with a FTP client (e.g., get, put, dir etc).
When using different Operating Systems one might find that the FTP *application* works in different ways, even though they may be using the same FTP protocol. The commands described below should work in most OS. However different FTP clients may have incomplete implementations or use non-standard FTP application commands.

To start the exercise:

1) Open a terminal
2) Issue the command: `ftp servername`
   Use any ***public FTP server*** you might know, for example: `ftp.aarnet.edu.au`
3) The server will ask for a username and a password. As the FTP is a public one, there might be a *guest* user that general public can use (usually the word **`anonymous`**). The password could be anything. It is common practice to fill this field with your email address (this is not mandatory, but it is a contribution to the administrator's log)
   ```
   anonymous
   your.email@address.domain (or some random chars)
   ```

4) When the connection is established, it is time to see what files the server has to offer. The command to list files and directories is: `ls` (`dir` might also work...)
5) To change directories: `cd <subdirectory>`
6) To download a file: `get <filename>`
7) To upload a file (many servers will *not* allow that!) `put <filename>`
8) To download all the files with extension .txt: `nget <*.txt>`
9) To see other commands (implementation dependent), type: `help`
10) In order to choose between *binary* and *ascii* files one needs to issue either :
    ```
    bin
    asc
    ```

If you want to upload files, you need to connect to a server where you have permission to do so.
(**Warning**: if ftp.massey.ac.nz works, your *user* and *password* are sent unencrypted...)

Part 2

We have seen some examples using the application called *telnet* to connect to servers listening to certain ports (see some examples in Chapter 2). Does that work with FTP?

Try downloading a file connecting via telnet: `telnet ftp.servername 21`

Key questions:
1) Can you connect?
2) Can you download a file successfully? Why or why not?

Part 3

Understanding the difference between FTP "*protocol commands*" and "*application commands*":

We have looked at protocol commands, such as: RETR, USER, PASS, STOR etc.

Also, there are application commands with an FTP client: get, put, ls etc

How do they relate? Use the debug mode to find out.
1) Open a terminal
2) Issue the command: `ftp servername`
Use any public FTP server you might know, for example: *ftp.aarnet.edu.au*
3) Issue the command (after the ftp prompt): `debug`
This is going to change the ftp client's mode to debug mode. Now you can see what the client is sending to the server, what is the raw text message. For example:

```
ftp> ls
ftp: setsockopt (ignored): Permission denied
---> PORT 130,123,246,212,167,215
200 PORT command successful. Consider using PASV.
---> LIST
150 Here comes the directory listing.
-rw-r--r--    1 ftp      ftp           528 Nov 01  2007 README
-rw-r--r--    1 ftp      ftp           560 Sep 28  2007 index.html
drwxr-xr-x   42 ftp      ftp          4096 Jul 09 01:56 pub
226 Directory send OK.
Ftp>
```

You can see that the application command 'ls' generated the following FTP protocol commands:
PORT
LIST
and that the server responded with the following number codes:
200
150
226

4) Find out what are the protocol commands generated by the following application commands:
dir ___________________________
get ___________________________
help ___________________________
cd ___________________________
put ___________________________

5) Also, list all the number codes that the server sent to you when issuing the previous commands.
dir ___________________________
get ___________________________
help ___________________________
cd ___________________________
put ___________________________

Note: the application commands and the response codes may vary depending on the circumstances. This is going to be explored in another exercise.