FTP mini-HOWTO

Table of Contents

FTP mini-HOWTO

Matthew Borowski (mkb@yahoo.com) (http://tarp.worldserve.net/)

v0.2, 9 January 2000

How to use ftp clients and servers

1. Preamble

This document was written by Matthew Borowski for the Linux Documentation Project. I have not yet finished adding everything I want to add. There will be a section with information on operating an FTP server.

1.1 Contacting Me

Please contact me if you have any additions or changes for this document. Please do not ask for technical support -- you may purchase Unix consulting and support through my company, **WorldServe**, or you can ask for help on newsgroups such as **comp.os.linux.***. Please also keep in mind that I can only understand English and Farsi.

1.2 Legalities and Distribution

1.3 History of this document

- 991203: FTP HOWTO first written. Still a work-in-progress.

2. About FTP

FTP (File Transfer Protocol) is a client/server protocol that allows a user to transfer files to and from a remote network site. It works with TCP and is most commonly used on the Internet, although it can also be used on a LAN.

An FTP site is a computer that is running FTP server software (also known an FTP daemon, or `ftpd`). A public ftp site can usually be accessed by anybody by logging in as `anonymous` or `ftp`. There are many excellent public ftp sites that make repositories of free Unix software available. By learning how to use FTP, you give yourself access to an indespensible resource.

Private FTP sites require a user name or password. If you have a shell account with your ISP, you may be able to access your files via FTP (contact your system administrator to check on this).

An FTP client is the userland application that provides access to FTP servers. There are many FTP clients available. Some are graphical, and some are text-based.

FTP was first developed by the University of California, Berkeley for inclusion in 4.2BSD (Berkeley Unix). The RFC (Request for Comments) is available at ftp://nic.merit.edu/documents/rfc/rfc0959.txt.

3. Beginner's guide to using ftp

A quick guide to using ftp.

The standard ftp program is the original ftp client. It comes standard with most Linux distributions. It first appeared in 4.2BSD, which was developed by the University of California, Berkeley.

3.1 Running the `ftp` program

It's easy to use ftp. Let's say you want to connect to the anonymous ftp site metalab.unc.edu, to download the latest Linux kernel source.

At the command line, type:

```
$ ftp metalab.unc.edu
```

The ftp program will attempt to connect to metalab.unc.edu. Another way to do this is to run `ftp` from the command line with no parameters, and use the `open` command, with the site name as an argument:

```
$ ftp
ftp> open metalab.unc.edu
```

3.2 Logging into an FTP server

When you connect to an FTP site, it will ask you for a login (pressing enter will log in as your local user name, in this case, `foo`: We log in as `anonymous` or `ftp`, to get to the public archive.

```
220 helios.oit.unc.edu FTP server (Version wu-2.6.0(2) Wed Nov 17 14:44:12
EST 1999) ready.
Name (metalab.unc.edu:foo):
```

Now, we enter a complete e-mail address as the password (this is what most public FTP sites request).

```
331 Guest login ok, send your complete e-mail address as password.
Password:
```

After a successful login, the following information is given to us:

```
Remote system type is UNIX.
Using binary mode to transfer files.
ftp>
```

3.3 File transfer types

After you log in to an ftp site, `ftp` will print out the file transfer type. In our case, it is binary. Binary mode transfers the files, bit by bit, as they are on the FTP server. Ascii mode, however, will download the text directly. You can type `ascii` or `binary` to switch between the types.

You want to download the kernel source, so you leave the file transfer type at binary. The binary type is also what you would use for any non-text files -- such as graphic images, zip/gzip archives, executable programs, etc. If in doubt, use binary mode.

3.4 Navigating and listing directories

You do an `ls` to see a list of the files. The `ls` command on ftp servers is executed on the remote server, so the command line options that you can use with it vary from server to server. The most common options are generally available, check the manpage for `ls` for details.

```
ftp> ls
200 PORT command successful.
150 Opening ASCII mode data connection for /bin/ls.
total 33590
-r--r--r--   1 root     other    34348506 Dec 03 03:53 IAFA-LISTINGS
lrwxrwxrwx   1 root     other           7 Jul 15  1997 README -> WELCOME
-rw-r--r--   1 root     other         890 Nov 15 13:11 WELCOME
dr-xr-xr-x   2 root     other         512 Jul 15  1997 bin
dr-xr-xr-x   2 root     other         512 Jul 15  1997 dev
dr-xr-xr-x   2 root     other         512 Jul 18  1997 etc
drwxrwxrwx  11 ftp      20           4608 Nov 28 16:00 incoming
lrwxrwxrwx   1 root     other          13 Jun 04  1998 ls-lR -> IAFA-LISTINGS
dr-xr-xr-x  17 root     root          512 Jun 08 11:43 pub
dr-xr-xr-x   3 root     other         512 Jul 15  1997 unc
dr-xr-xr-x   5 root     other         512 Jul 15  1997 usr
226 Transfer complete.
```

If the `ls` command lists so many files that they scroll off the top of the screen, you can use Shift-PageUp to scroll up. This works in Linux console mode as well as in xterm or rxvt.

On public FTP archives, the downloadable resources are usually held in the `/pub` directory. In this example, you already know that the kernel sources are in the directory `/pub/Linux/kernel`, so you type the following to get into that directory:

```
ftp> cd pub/Linux/kernel
250-README for kernel
250-
250-What you'll find here: kernel sources and patches
250-
250-
250 CWD command successful.
```

The messages you see, which begin with "250", are information messages sent by the server. In this case, the ftp server is configured to automatically send you the README file when you cd into the directory.

3.5 Downloading and uploading files

Now, after doing another `ls`, you see that you want to `cd` into the v2.2 directory. You do yet another `ls`, and find the file you want to download. It is `linux-2.2.13.tar.gz`. So you type this:

```
ftp> get linux-2.2.13.tar.gz
local: linux-2.2.13.tar.gz remote: linux-2.2.13.tar.gz
200 PORT command successful.
150 Opening BINARY mode data connection for linux-2.2.13.tar.gz (15079540
bytes).
```

The ftp program has started saving the remote file `linux-2.2.13.tar.gz` as the local file `linux-2.2.13.tar.gz`.

If you wanted to save it as the local file `foo.tar.gz`, you could have specified it like this:

```
ftp> get linux-2.2.13.tar.gz foo.tar.gz
local: foo.tar.gz remote: linux-2.2.13.tar.gz
200 PORT command successful.
150 Opening BINARY mode data connection for linux-2.2.13.tar.gz (15079540
bytes).
```

If you want to download more than one file at a time, you'll have to use the `mget` (multiple get) command. You can use `mget` together with a space-delimited list of filenames you want to download, or you can use wildcards with the `mget` command. For example:

```
ftp> mget linux*
```

Would get all files starting with the string "linux". Normally, `mget` will prompt you for each file before it downloads it. You can toggle this by using the `prompt` command.

Now let's say you've written a piece of software, and you want to upload it to MetaLab to be included in their Linux software archive. First, you'd change to the /incoming directory (most public FTP servers have a directory, usually called incoming or uploads, where files can be uploaded), then you'd use the put command:

```
ftp> cd /incoming
ftp> put foo.tar.gz
local: foo.tar.gz remote: foo.tar.gz
200 PORT command successful.
150 Opening BINARY mode data connection for foo.tar.gz.
226 Transfer complete.
10257 bytes sent in 0.00316 secs (3.2e+03 Kbytes/sec)
```

The `put` command works the same way as the `get` command, so you can use `mput` to upload multiple files at the same time. You can also upload a local file with a different filename on the server by specifying the remote filename and/or pathname as an argument.

What if the file `foo.tar.gz` is not in your current local directory when you try to upload it? You can switch local directories by using the `lcd` (local change directory) command:

```
ftp> lcd /home/foo/
Local directory now /home/foo
```

3.6 Running shell commands

The ftp client supports using the bang (!) to run local commands. For example, to get a listing of files in your current local directory, do this:

```
ftp> !ls
```

The way this works is that ftp calls the shell (specified in the `$SHELL` environment variable), and it is the shell which runs `ls`. Thus, you can run any command-line which works with your shell simply by prepending "!" to it (the default shell in most Linux distributions is bash, the Bourne Again SHell). Please note that `!cd` does not work as you would expect, this is why the `lcd` command exists.

3.7 Hash marks and tick

Wouldn't it be nice if you could watch the progress while you're downloading a file with ftp? You can use the hash command to print out hash marks as you download a file:

```
ftp> hash
Hash mark printing on (1024 bytes/hash mark).
```

As you can tell, ftp will print a hash mark for every 1024 bytes of data you download.

There is also a `tick` option.

```
ftp> tick
Tick counter printing on (10240 bytes/tick increment).

This will print something to this effect as you download a file:

Bytes transferred: 11680
```

3.8 Other ftp commands

There are many other ftp commands. If you have the permissions to do so (which you should, if you are connected to your own private shell account), you can make a directory on the remote server using the `mkdir` command. You can remove a file on the remote server using the `delete` command, or `rmdir` to remove a directory. You can also change file permissions using the `chmod` command.

For more elaborate information on using ftp, please see the online help in the ftp program (accessible by typing help with no arguments for a list of commands, or help <commandname> for specific help on a command). You can also read the Unix man page for ftp by typing `man ftp` at your command prompt.

4. Console FTP clients

The original ftp program was the original ftp client, and it is a good investment to learn it. It's the only ftp client that you can be certain is available on most systems (even Win32 comes with the ftp command, albeit an archaic, braindead version of it).

There are many other console-mode (text-only) ftp clients available. The listing here is by no means comprehensive, but includes the most popular ones. Search at FreshMeat to find more.

4.1 NcFTP

NcFTP is the all-time favorite ftp client of many Unix users. It comes bundled with most Linux distributions, and offers many advanced features such as tab completion and bookmarks. Version 2 of NcFTP had a curses based full-screen mode. This was done away with in Version 3 (now in beta).

It's not 100% compatible with the commands that standard ftp uses. For example, get and put in NcFTP act like mget and mput do in standard ftp. So if you want to save a remote file as a different local filename, you'd have to do get -z remotename localname. Thankfully, NcFTP has a nice online help system to assist you in learning the commands.

You can get the latest version of NcFTP at http://www.ncftp.com.

4.2 lukemftp

A port of the NetBSD FTP client to other systems, lukemftp derives its name from the author of most of the enhanced features, which include: command-line editing, command-line fetches of FTP and HTTP URLs (including via proxies), context-sensitive word completion, dynamic progress bar, IPv6 support, modification time preservation, paging of local and remote files, passive mode support (with fallback to active mode), SOCKS support, TIS FWTK gate-ftp server support, and transfer rate throttling.

I highly recommend lukemftp to users who don't want to change to anything drastically different from the standard ftp client, but want more advanced features.

You can get the latest version of lukemftp at ftp://ftp.netbsd.org/pub/NetBSD/misc/lukemftp/.

4.3 lftp

lftp is a sophisticated command line based FTP client. Like bash, it has job control. It uses the GNU readline library for input, so you have command line completion and editing. lftp also has bookmarks, mirroring support, and can transfer several files in parellel.

You can get the latest version of lftp at http://ftp.yars.free.net/projects/lftp/.

Debian packages are available at ftp://ftp.freshmeat.net/pub/debs/lftp/.

4.4 cftp

Comfortable FTP (cftp) is a full screen mode client. What it lacks in features, it makes up for in ease of use. You browse through the directories using the arrow keys and enter.

You should be able to get the latest version of cftp at http://ftp.giga.or.at/pub/nih/cftp/.

4.5 yafc

Yafc is a very nice ftp client, with features including directory cache, remote filename completion, aliases, colorized ls, recursive get/put/ls/rm, nohup mode transfers, tagging (queueing), multiple connections, proxy support and more. It has support for Kerberos4 authentication.

You can get the latest version of yafc from http://www.stacken.kth.se/~mhe/yafc/.

Debian packages are available at http://members.home.com/decklin/experimental/.

Redhat packages are available at http://lz.freeservers.com/linux/yafc.html.

5. X Window FTP clients

There are several graphical FTP clients designed to run on the X Window system. These clients offer ease of use for users who are used to graphical environments, and sometimes offer versatile options that would be hard to implement in a text-based ftp client.

5.1 gFTP

gFTP is an FTP client for X Windows written using Gtk. The interface has some similarities to the popular WS_FTP software commonly used on a certain unstable operating system.

gFTP features simultaneous downloads, resuming of interrupted file transfers, file transfer queues, downloading of entire directories, ftp proxy support, remote directory caching, passive and non-passive file transfers, drag-n-drop support, a very nice connection manager and more.

If you are running Red Hat Linux and have the GNOME desktop installed, then you probably already have gFTP. If not, you can download gFTP from its homepage at http://gftp.seul.org/.

5.2 WXftp

WXftp is an FTP client for the X Window System designed to be used mainly on Linux workstations. It is written using the WXWindows toolkit, so it can be compiled to use either Motif or GTK+

It includes an intuitive user interface (much like WS_FTP), a session manager, on-line help, a progress bar, and more

Check out WXftp's homepage at http://www.wxftp.seul.org.

5.3 LLNL XDIR and XFTP

LLNL XFTP was one of the first graphical FTP clients for Linux. It supports FXP (file transfer between two remote hosts), and has a Motif based interface.

More information is available at http://www.llnl.gov/ia/xdir_xftp/.

5.4 guiftp

Guiftp is a simple ftp client written with the GTK+ toolkit. It's good if you don't need many features and want a simple, clean look.

Guiftp's homepage is at http://www.altern.org/ldufresne/guiftp/.

6. FTP Servers

6.1 How an FTP Server works

A traditional FTP server is executed from inetd (the internet superserver daemon). The standard FTP port is port 21. When a user tries to log in, the FTP server uses a standard system call to check the user name and password against the entries in the system password file, or the NIS tables if you are using NIS. If the login is correct, the user is given access to the system.

Anonymous FTP works differently. The user logs in with either the anonymous or the ftp username (this can be defined in the config file). He is then given access to a directory tree that has been chroot()'ed. This ensures that the user can not gain access to directory trees he is not authorized for. The chrooted directory tree usually contains a mock filesystem, with bin/, etc/, and lib/ directories. The files for download are usually put in the pub/ directory.

The reason for a mock filesystem in an anonymous FTP tree is that the FTP daemon runs external commands for ls requests. You can also place additional programs in the bin directory, and a user can run them with the SITE command in his ftp client. For example, Red Hat's FTP includes the RPM command (for users to query RPM packages on the site).

Some FTP servers work differently. For example, some will allow user accounts to be set up independant of the system-wide password file (FTP-only accounts). Some servers (ProFTPD and NcFTPd for instance) have built-in ls commands and do not need a special directory tree within the chroot structure. Other ftp servers stray altogether from the standard ftp concept. FTP4ALL, for example, does not use system passwords at all. It uses it's own user and group file, and has features such as upload/download ratio and customizable server messages.

6.2 Help with FTP Servers

WU-FTPD

WU-FTPD is the ftp daemon included with many Linux distributions, including Red Hat and Caldera. You can learn more about WU-FTPD at http://www.wu-ftpd.org.

The WU-FTPD FAQ can be found on the web at http://www.cetis.hvu.nl/~koos/wu-ftpd-faq.html.

ProFTPD

ProFTPD is a powerful FTP server that includes Apache-style configuration, extensive support for virtual hosts, and internal ls.

A complete command reference and downloads can be found at http://www.proftpd.org