SECURE FTP

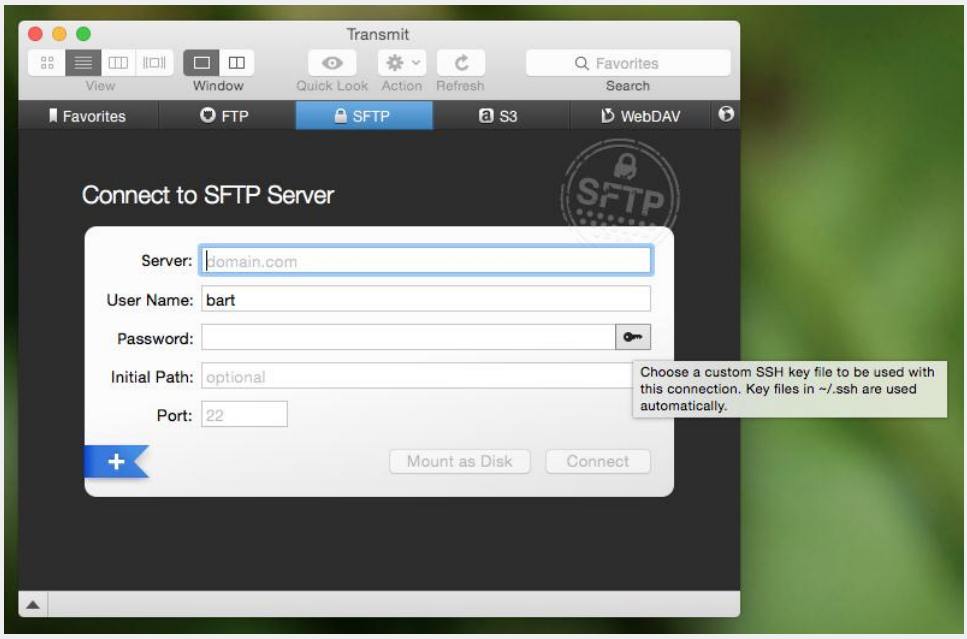
The final secure file transfer protocol we'll be looking at is SFTP, which is basically the old insecure FTP protocol re-implemented to use SSH as the communication channel. This protocol mostly used by GUI apps like Panic's Transmit rather than from the terminal. This is because, unlike **rsync** and **scp**, this command does not immediately do anything, it simply gives you a new command shell in which to enter FTP commands.

If you know the source and destination paths, I would recommend using **scp** or **rsync** over SFTP when working in the terminal. However, it can be useful if you need to explore the remote file system to find the file you want to transfer, or if you are already familiar with the FTP shell.

Like **scp** and **rsync**, SFTP can make use of SSH key-pairs to connect without the need to enter a password. This is also true when using SFTP through most SFTP GUI apps. Good GUI SFTP apps like Transmit will use SSH keys automatically, but some SFTP GUI apps make you manually specify that you wish to use a key, and/or specify the key to be used.

Transmit is the SFTP client I use each and every day, and I love it, but, they didn't make it at all obvious that they have SSH key support.

Users could be forgiven for not connecting the small key icon next to the password field with SSH key-pairs. If you hover over that icon you'll see that Transmit uses keys in the default location automatically, and that if you want to use a key in a different location, you need to click on the key icon to specify the path to the key file you'd like to use.



While the FTP shell is not difficult to use, I don't think it is worth spending too much time on it in this series. Personally, I never use it because I find that **scp** and **rsync** allow me to achieve my goals more easily. But, I would like to give you a flavour of it, and you can then decide whether or not you'd like to learn more.

Let's looks at how to initiate an SFTP session, at some of the most important FTP commands.

You can connect to the remote computer with the command:

> sftp **user@computer**

If you know the remote folder you want to copy files from, you can also specify that while connecting as follows:

> sftp **user@computer**:**remote_path**

When ever any command puts me into another shell, the first thing I want to know is how to get out! With SFTP you have two choices, the traditional FTP command **bye**, or the more memorable command **exit**.

Within a BASH shell you are used to the concept of a present working directory, but in an (S)FTP shell that concept is extended to two present working directories, a present local working directory, and a present remote working directory. The default local present working directory is the folder from which you issued the (S)FTP command, and the default remote present wording directory is the home directory of the user you connected as. You can see each of these two current paths with the commands **lpwd** (local present working directory) and **pwd** (remote present working directory).

You can change both of these paths at any time using the **lcd** (local change directory), and **cd** (remote change directory) commands.

You can also list the contents of both present working directories with the commands **lls** (local file listing), and **ls** (remote file listing).

Finally, there are the all important commands for uploading and downloading files. To download a file from the remote present working directory, to the local present working directory, you use the **get** command, which takes one or more arguments, the names of the files to download. Similarly, to upload a file from the local present working directory to the remote present working directory, you use the **put** command, which also takes file names as arguments.

Source: https://www.bartbusschots.ie/s/2015/04/04/taming-the-terminal-part-31-of-n-ssh-file-transfers/