Chapter 6: Using PSFTP to transfer files securely

PSFTP, the PuTTY SFTP client, is a tool for transferring files securely between computers using an SSH connection.

PSFTP differs from PSCP in the following ways:

- PSCP should work on virtually every SSH server. PSFTP uses the new SFTP protocol, which is a feature of SSH-2 only. (PSCP will also use this protocol if it can, but there is an SSH-1 equivalent it can fall back to if it cannot.)
- PSFTP allows you to run an interactive file transfer session, much like the Windows `ftp` program. You can list the contents of directories, browse around the file system, issue multiple `get` and `put` commands, and eventually log out. By contrast, PSCP is designed to do a single file transfer operation and immediately terminate.

6.1 Starting PSFTP

The usual way to start PSFTP is from a command prompt, much like PSCP. To do this, it will need either to be on your `PATH` or in your current directory. To add the directory containing PSFTP to your `PATH` environment variable, type into the console window:

```
set PATH=C:\path\to\putty\directory;%PATH%
```

Unlike PSCP, however, PSFTP has no complex command-line syntax; you just specify a host name and perhaps a user name:

```
psftp server.example.com
```

or perhaps

```
psftp fred@server.example.com
```

Alternatively, if you just type `psftp` on its own (or double-click the PSFTP icon in the Windows GUI), you will see the PSFTP prompt, and a message telling you PSFTP has not connected to any server:

```
C:\>psftp
psftp: no hostname specified; use "open host.name" to connect
psftp>
```

At this point you can type `open server.example.com` or `open fred@server.example.com` to start a session.

PSFTP accepts all the general command line options supported by the PuTTY tools, except the ones which make no sense in a file transfer utility. See section 3.8.3 for a description of these options. (The ones not supported by PSFTP are clearly marked.)

PSFTP also supports some of its own options. The following sections describe PSFTP's specific command-line options.

6.1.1 `-b`: specify a file containing batch commands

In normal operation, PSFTP is an interactive program which displays a command line and accepts commands from the keyboard.

If you need to do automated tasks with PSFTP, you would probably prefer to specify a set of commands in advance and have them executed automatically. The `-b` option allows you to do this. You use it with a file name containing batch commands. For example, you might create a file called `myscript.scr` containing lines like this:

```
cd /home/ftp/users/jeff
del jam-old.tar.gz
ren jam.tar.gz jam-old.tar.gz
put jam.tar.gz
chmod a+r jam.tar.gz
```

and then you could run the script by typing

```
psftp user@hostname -b myscript.scr
```

When you run a batch script in this way, PSFTP will abort the script if any command fails to complete successfully. To change this behaviour, you can add the `-be` option (section 6.1.3).

PSFTP will terminate after it finishes executing the batch script.

6.1.2 `-bc`: display batch commands as they are run

The `-bc` option alters what PSFTP displays while processing a batch script specified with `-b`. With the `-bc` option, PSFTP will display prompts and commands just as if the commands had been typed at the keyboard. So instead of seeing this:

```
C:\>psftp fred@hostname -b batchfile
Sent username "fred"
```

```
Remote working directory is /home/fred
Listing directory /home/fred/lib
drwxrwsr-x    4 fred     fred         1024 Sep  6 10:42 .
drwxr-sr-x   25 fred     fred         2048 Dec 14 09:36 ..
drwxrwsr-x    3 fred     fred         1024 Apr 17  2000 jed
lrwxrwxrwx    1 fred     fred           24 Apr 17  2000 timber
drwxrwsr-x    2 fred     fred         1024 Mar 13  2000 trn
```

you might see this:

```
C:\>psftp fred@hostname -bc -b batchfile
Sent username "fred"
Remote working directory is /home/fred
psftp> dir lib
Listing directory /home/fred/lib
drwxrwsr-x    4 fred     fred         1024 Sep  6 10:42 .
drwxr-sr-x   25 fred     fred         2048 Dec 14 09:36 ..
drwxrwsr-x    3 fred     fred         1024 Apr 17  2000 jed
lrwxrwxrwx    1 fred     fred           24 Apr 17  2000 timber
drwxrwsr-x    2 fred     fred         1024 Mar 13  2000 trn
psftp> quit
```

6.1.3 `-be`: continue batch processing on errors

When running a batch file, this additional option causes PSFTP to continue processing even if a command fails to complete successfully.

You might want this to happen if you wanted to delete a file and didn't care if it was already not present, for example.

6.1.4 `-batch`: avoid interactive prompts

If you use the `-batch` option, PSFTP will never give an interactive prompt while establishing the connection. If the server's host key is invalid, for example (see section 2.2), then the connection will simply be abandoned instead of asking you what to do next.

This may help PSFTP's behaviour when it is used in automated scripts: using `-batch`, if something goes wrong at connection time, the batch job will fail rather than hang.

6.2 Running PSFTP

Once you have started your PSFTP session, you will see a `psftp>` prompt. You can now type commands to perform file-transfer functions. This section lists all the available commands.

6.2.1 General quoting rules for PSFTP commands

Most PSFTP commands are considered by the PSFTP command interpreter as a sequence of words, separated by spaces. For example, the command `ren oldfilename newfilename` splits up into three words: `ren` (the command name), `oldfilename` (the name of the file to be renamed), and `newfilename` (the new name to give the file).

Sometimes you will need to specify file names that *contain* spaces. In order to do this, you can surround the file name with double quotes. This works equally well for local file names and remote file names:

```
psftp> get "spacey file name.txt" "save it under this name.txt"
```

The double quotes themselves will not appear as part of the file names; they are removed by PSFTP and their only effect is to stop the spaces inside them from acting as word separators.

If you need to *use* a double quote (on some types of remote system, such as Unix, you are allowed to use double quotes in file names), you can do this by doubling it. This works both inside and outside double quotes. For example, this command

```
psftp> ren ""this"" "a file with ""quotes"" in it"
```

will take a file whose current name is `"this"` (with a double quote character at the beginning and the end) and rename it to a file whose name is `a file with "quotes" in it`.

(The one exception to the PSFTP quoting rules is the `!` command, which passes its command line straight to Windows without splitting it up into words at all. See section 6.2.19.)

6.2.2 Wildcards in PSFTP

Several commands in PSFTP support 'wildcards' to select multiple files.

For *local* file specifications (such as the first argument to `put`), wildcard rules for the local operating system are used. For instance, PSFTP running on Windows might require the use of `*.*` where PSFTP on Unix would need `*`.

For *remote* file specifications (such as the first argument to `get`), PSFTP uses a standard wildcard syntax (similar to POSIX wildcards):

- `*` matches any sequence of characters (including a zero-length sequence).
- `?` matches exactly one character.
- `[abc]` matches exactly one character which can be `a`, `b`, or `c`.

  `[a-z]` matches any character in the range `a` to `z`.

  `[^abc]` matches a single character that is *not* `a`, `b`, or `c`.

  Special cases: `[-a]` matches a literal hyphen (`-`) or `a`; `[^-a]` matches all other characters. `[a^]` matches a literal caret (`^`) or `a`.

- `\` (backslash) before any of the above characters (or itself) removes that character's special meaning.

A leading period (`.`) on a filename is not treated specially, unlike in some Unix contexts; `get *` will fetch all files, whether or not they start with a leading period.

6.2.3 The `open` command: start a session

If you started PSFTP by double-clicking in the GUI, or just by typing `psftp` at the command line, you will need to open a connection to an SFTP server before you can issue any other commands (except `help` and `quit`).

To create a connection, type `open host.name`, or if you need to specify a user name as well you can type `open user@host.name`.

Once you have issued this command, you will not be able to issue it again, *even* if the command fails (for example, if you mistype the host name or the connection times out). So if the connection is not opened successfully, PSFTP will terminate immediately.

6.2.4 The `quit` command: end your session

When you have finished your session, type the command `quit` to close the connection, terminate PSFTP and return to the command line (or just close the PSFTP console window if you started it from the GUI).

You can also use the `bye` and `exit` commands, which have exactly the same effect.

6.2.5 The `close` command: close your connection

If you just want to close the network connection but keep PSFTP running, you can use the `close` command. You can then use the `open` command to open a new connection.

6.2.6 The `help` command: get quick online help

If you type `help`, PSFTP will give a short list of the available commands.

If you type `help` with a command name - for example, `help get` - then PSFTP will give a short piece of help on that particular command.

6.2.7 The `cd` and `pwd` commands: changing the remote working directory

PSFTP maintains a notion of your ‘working directory’ on the server. This is the default directory that other commands will operate on. For example, if you type `get filename.dat` then PSFTP will look for `filename.dat` in your remote working directory on the server.

To change your remote working directory, use the `cd` command. If you don't provide an argument, `cd` will return you to your home directory on the server (more precisely, the remote directory you were in at the start of the connection).

To display your current remote working directory, type `pwd`.

6.2.8 The `lcd` and `lpwd` commands: changing the local working directory

As well as having a working directory on the remote server, PSFTP also has a working directory on your local machine (just like any other Windows process). This is the default local directory that other commands will operate on. For example, if you type `get filename.dat` then PSFTP will save the resulting file as `filename.dat` in your local working directory.

To change your local working directory, use the `lcd` command. To display your current local working directory, type `lpwd`.

6.2.9 The `get` command: fetch a file from the server

To download a file from the server and store it on your local PC, you use the `get` command.

In its simplest form, you just use this with a file name:

```
get myfile.dat
```

If you want to store the file locally under a different name, specify the local file name after the remote one:

```
get myfile.dat newname.dat
```

This will fetch the file on the server called `myfile.dat`, but will save it to your local machine under the name `newname.dat`.

To fetch an entire directory recursively, you can use the `-r` option:

```
get -r mydir
get -r mydir newname
```

(If you want to fetch a file whose name starts with a hyphen, you may have to use the `--` special argument, which stops `get` from interpreting anything as a switch after it. For example, '`get -- -silly-name-`'.)

6.2.10 The `put` command: send a file to the server

To upload a file to the server from your local PC, you use the `put` command.

In its simplest form, you just use this with a file name:

```
put myfile.dat
```

If you want to store the file remotely under a different name, specify the remote file name after the local one:

```
put myfile.dat newname.dat
```

This will send the local file called `myfile.dat`, but will store it on the server under the name `newname.dat`.

To send an entire directory recursively, you can use the `-r` option:

```
put -r mydir
put -r mydir newname
```

(If you want to send a file whose name starts with a hyphen, you may have to use the `--` special argument, which stops `put` from interpreting anything as a switch after it. For example, '`put -- -silly-name-`'.)

6.2.11 The `mget` and `mput` commands: fetch or send multiple files

`mget` works almost exactly like `get`, except that it allows you to specify more than one file to fetch at once. You can do this in two ways:

- by giving two or more explicit file names ('`mget file1.txt file2.txt`')
- by using a wildcard ('`mget *.txt`').

Every argument to `mget` is treated as the name of a file to fetch (unlike `get`, which will interpret at most one argument like that, and a second argument will be treated as an alternative name under which to store the retrieved file), or a wildcard expression matching more than one file.

The `-r` and `--` options from `get` are also available with `mget`.

`mput` is similar to `put`, with the same differences.

6.2.12 The `reget` and `reput` commands: resuming file transfers

If a file transfer fails half way through, and you end up with half the file stored on your disk, you can resume the file transfer using the `reget` and `reput` commands. These work exactly like the `get` and `put` commands, but they check for the presence of the half-written destination file and start transferring from where the last attempt left off.

The syntax of `reget` and `reput` is exactly the same as the syntax of `get` and `put`:

```
reget myfile.dat
reget myfile.dat newname.dat
reget -r mydir
```

These commands are intended mainly for resuming interrupted transfers. They assume that the remote file or directory structure has not changed in any way; if there have been changes, you may end up with corrupted files. In particular, the `-r` option will not pick up changes to files or directories already transferred in full.

6.2.13 The `dir` command: list remote files

To list the files in your remote working directory, just type `dir`.

You can also list the contents of a different directory by typing `dir` followed by the directory name:

```
dir /home/fred
dir sources
```

And you can list a subset of the contents of a directory by providing a wildcard:

```
dir /home/fred/*.txt
dir sources/*.c
```

The `ls` command works exactly the same way as `dir`.

6.2.14 The `chmod` command: change permissions on remote files

PSFTP allows you to modify the file permissions on files and directories on the server. You do this using the `chmod` command, which works very much like the Unix `chmod` command.

The basic syntax is `chmod modes file`, where `modes` represents a modification to the file permissions, and `file` is the filename to modify. You can specify multiple files or wildcards. For example:

```
chmod go-rwx,u+w privatefile
chmod a+r public*
chmod 640 groupfile1 groupfile2
```

The `modes` parameter can be a set of octal digits in the Unix style. (If you don't know what this means, you probably don't want to be using it!) Alternatively, it can be a list of permission modifications, separated by commas. Each modification consists of:

- The people affected by the modification. This can be `u` (the owning user), `g` (members of the owning group), or `o` (everybody else - 'others'), or some combination of those. It can also be `a` ('all') to affect everybody at once.
- A `+` or `-` sign, indicating whether permissions are to be added or removed.
- The actual permissions being added or removed. These can be `r` (permission to read the file), `w` (permission to write to the file), and `x` (permission to execute the file, or in the case of a directory, permission to access files within the directory).

So the above examples would do:

- The first example: `go-rwx` removes read, write and execute permissions for members of the owning group and everybody else (so the only permissions left are the ones for the file owner). `u+w` adds write permission for the file owner.
- The second example: `a+r` adds read permission for everybody to all files and directories starting with 'public'.

In addition to all this, there are a few extra special cases for Unix systems. On non-Unix systems these are unlikely to be useful:

- You can specify `u+s` and `u-s` to add or remove the Unix set-user-ID bit. This is typically only useful for special purposes; refer to your Unix documentation if you're not sure about it.
- You can specify `g+s` and `g-s` to add or remove the Unix set-group-ID bit. On a file, this works similarly to the set-user-ID bit (see your Unix documentation again); on a directory it ensures that files created in the directory are accessible by members of the group that owns the directory.
- You can specify `+t` and `-t` to add or remove the Unix ‘sticky bit’. When applied to a directory, this means that the owner of a file in that directory can delete the file (whereas normally only the owner of the *directory* would be allowed to).

6.2.15 The `del` command: delete remote files

To delete a file on the server, type `del` and then the filename or filenames:

```
del oldfile.dat
del file1.txt file2.txt
del *.o
```

Files will be deleted without further prompting, even if multiple files are specified.

`del` will only delete files. You cannot use it to delete directories; use `rmdir` for that.

The `rm` command works exactly the same way as `del`.

6.2.16 The `mkdir` command: create remote directories

To create a directory on the server, type `mkdir` and then the directory name:

```
mkdir newstuff
```

You can specify multiple directories to create at once:

```
mkdir dir1 dir2 dir3
```

6.2.17 The `rmdir` command: remove remote directories

To remove a directory on the server, type `rmdir` and then the directory name or names:

```
rmdir oldstuff
rmdir *.old ancient
```

Directories will be deleted without further prompting, even if multiple directories are specified.

Most SFTP servers will probably refuse to remove a directory if the directory has anything in it, so you will need to delete the contents first.

6.2.18 The `mv` command: move and rename remote files

To rename a single file on the server, type `mv`, then the current file name, and then the new file name:

```
mv oldfile newname
```

You can also move the file into a different directory and change the name:

```
mv oldfile dir/newname
```

To move one or more files into an existing subdirectory, specify the files (using wildcards if desired), and then the destination directory:

```
mv file dir
mv file1 dir1/file2 dir2
mv *.c *.h ..
```

The `rename` and `ren` commands work exactly the same way as `mv`.

6.2.19 The `!` command: run a local Windows command

You can run local Windows commands using the `!` command. This is the only PSFTP command that is not subject to the command quoting rules given in section 6.2.1. If any command line begins with the `!` character, then the rest of the line will be passed straight to Windows without further translation.

For example, if you want to move an existing copy of a file out of the way before downloading an updated version, you might type:

```
psftp> !ren myfile.dat myfile.bak
psftp> get myfile.dat
```

using the Windows `ren` command to rename files on your local PC.

6.3 Using public key authentication with PSFTP

Like PuTTY, PSFTP can authenticate using a public key instead of a password. There are three ways you can do this.

Firstly, PSFTP can use PuTTY saved sessions in place of hostnames. So you might do this:

- Run PuTTY, and create a PuTTY saved session (see section 4.1.2) which specifies your private key file (see section 4.20.7). You will probably also want to specify a username to log in as (see section 4.14.1).
- In PSFTP, you can now use the name of the session instead of a hostname: type `psftp sessionname`, where `sessionname` is replaced by the name of your saved session.

Secondly, you can supply the name of a private key file on the command line, with the `-i` option. See section 3.8.3.18 for more information.

Thirdly, PSFTP will attempt to authenticate using Pageant if Pageant is running (see chapter 9). So you would do this:

- Ensure Pageant is running, and has your private key stored in it.
- Specify a user and host name to PSFTP as normal. PSFTP will automatically detect Pageant and try to use the keys within it.

For more general information on public-key authentication, see chapter 8.

If you want to provide feedback on this manual or on the PuTTY tools themselves, see the Feedback page.

[PuTTY release 0.60]