How To Use SFTP to Securely Transfer Files with a Remote Server

Authored by: ***ASPHostServer Administrator*** *[asphostserver@gmail.com]*
Saved From: *http://faq.asphosthelpdesk.com/article.php?id=178*

FTP, or "File Transfer Protocol" is a popular method of transferring files between two remote systems.

SFTP, which stands for SSH File Transfer Protocol, or Secure File Transfer Protocol, is a separate protocol packaged with SSH that works in a similar way over a secure connection. The advantage is the ability to leverage a secure connection to transfer files and traverse the filesystem on both the local and remote system.

In almost all cases, SFTP is preferable to FTP because of its underlying security features and ability to piggy-back on an SSH connection. FTP is an insecure protocol that should only be used in limited cases or on networks you trust.

Although SFTP is integrated into many graphical tools, this guide will demonstrate how to use it through its interactive command line interface.

How to Connect with SFTP

By default, SFTP uses the SSH protocol to authenticate and establish a secure connection. Because of this, the same authentication methods are available that are present in SSH.

Although passwords are easy to use and set up by default, we recommend you create SSH keys and transfer your public key to any system that you need to access. This is much more secure and can save you time in the long run.

Please see this guide to set up SSH keys in order to access your server if you have not done so already.

If you can connect to the machine using SSH, then you have completed all of the necessary requirements necessary to use SFTP to manage files. Test SSH access with the following command:

```
ssh username@remote_hostname_or_IP
```

If that works, exit back out by typing:

```
exit
```

We can establish an SSH connection and then open up an SFTP session using that connection by issuing the following command:

```
sftp username@remote_hostname_or_IP
```

You will connect the the remote system and your prompt will change to an SFTP prompt.

Getting Help in SFTP

The most useful command to learn first is the help command. This gives you access to a summary of the SFTP help. You can call it by typing either of these in the prompt:

```
help
```

```
?
```

This will a list of the available commands:

```
Available commands:
bye                                Quit sftp
cd path                            Change remote directory to 'path'
chgrp grp path                     Change group of file 'path' to 'grp'
chmod mode path                    Change permissions of file 'path' to 'mode'
chown own path                     Change owner of file 'path' to 'own'
df [-hi] [path]                    Display statistics for current directory or
                                   filesystem containing 'path'
exit                               Quit sftp
get [-Ppr] remote [local]          Download file
help                               Display this help text
lcd path                           Change local directory to 'path'
. . .
```

We will explore some of the commands you see in the following sections.

Navigating with SFTP

We can navigate through the remote system's file hierarchy using a number of commands that function similarly to their shell counterparts.

First, let's orient ourselves by finding out which directory we are in currently on the remote system. Just like in a typical shell session, we can type the following to get the current directory:

```
pwd
```

```
Remote working directory: /home/demouser
```

We can view the contents of the current directory of the remote system with another familiar command:

```
ls
```

```
Summary.txt     info.html       temp.txt        testDirectory
```

Note that the commands within the SFTP interface are not the normal shell commands and are not as feature-rich, but they do implement some of the more important optional flags:

```
ls -la
```

```
drwxr-xr-x    5 demouser   demouser       4096 Aug 13 15:11 .
drwxr-xr-x    3 root     root         4096 Aug 13 15:02 ..
-rw-------    1 demouser   demouser          5 Aug 13 15:04 .bash_history
-rw-r--r--    1 demouser   demouser        220 Aug 13 15:02 .bash_logout
-rw-r--r--    1 demouser   demouser       3486 Aug 13 15:02 .bashrc
drwx------    2 demouser   demouser       4096 Aug 13 15:04 .cache
-rw-r--r--    1 demouser   demouser        675 Aug 13 15:02 .profile
. . .
```

To get to another directory, we can issue this command:

```
cd testDirectory
```

We can now traverse the remote file system, but what if we need to access our local file system? We can direct commands towards the local file system by preceding them with an "l" for local.

All of the commands discussed so far have local equivalents. We can print the local working directory:

```
lpwd

Local working directory: /Users/demouser
```

We can list the contents of the current directory on the local machine:

```
lls

Desktop   local.txt  test.html
Documents  analysis.rtf  zebra.html
```

We can also change the directory we wish to interact with on the local system:

```
lcd Desktop
```

Transferring Files with SFTP

Navigating the remote and local filesystems is of limited usefulness without being able to transfer files between the two.

Transferring Remote Files to the Local System

If we would like download files from our remote host, we can do so by issuing the following command:

```
get remoteFile

Fetching /home/demouser/remoteFile to remoteFile
/home/demouser/remoteFile                     100%   37KB  36.8KB/s   00:01
```

As you can see, by default, the "get" command downloads a remote file to a file with the same name on the local file system.

We can copy the remote file to a different name by specifying the name afterwards:

```
get remoteFile localFile
```

The "get" command also takes some option flags. For instance, we can copy a directory and all of its contents by specifying the recursive option:

```
get -r someDirectory
```

We can tell SFTP to maintain the appropriate permissions and access times by using the "-P" or "-p" flag:

```
get -Pr someDirectory
```

Transferring Local Files to the Remote System

Transferring files to the remote system is just as easily accomplished by using the appropriately named "put" command:

```
put localFile

Uploading localFile to /home/demouser/localFile
localFile                                     100% 7607     7.4KB/s   00:00
```

The same flags that work with "get" apply to "put". So to copy an entire local directory, you can issue:

```
put -r localDirectory
```

One familiar tool that is useful when downloading and uploading files is the "df" command, which works similar to the command line version. Using this, you can check that you have enough space to complete the transfers you are interested in:

```
df -h

    Size     Used    Avail   (root)    %Capacity
  19.9GB   1016MB   17.9GB   18.9GB           4%
```

Please note, that there is no local variation of this command, but we can get around that by issuing the "!" command.

The "!" command drops us into a local shell, where we can run any command available on our local system. We can check disk usage by typing:

```
!
df -h
```

```
Filesystem      Size   Used  Avail Capacity  Mounted on
/dev/disk0s2   595Gi   52Gi  544Gi     9%    /
devfs          181Ki  181Ki    0Bi   100%    /dev
map -hosts       0Bi    0Bi    0Bi   100%    /net
map auto_home    0Bi    0Bi    0Bi   100%    /home
```

Any other local command will work as expected. To return to your SFTP session, type:

```
exit
```

You should now see the SFTP prompt return.

Simple File Manipulations with SFTP

SFTP allows you to perform the type of basic file maintenance that is useful when working with file hierarchies.

For instance, you can change the owner of a file on the remote system with:

```
chown userID file
```

Notice how, unlike the system "chmod" command, the SFTP command does not accept usernames, but instead uses UIDs. Unfortunately, there is no easy way to know the appropriate UID from within the SFTP interface.

An involved work around could be accomplished with:

```
get /etc/passwd
!less passwd

root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/bin/sh
bin:x:2:2:bin:/bin:/bin/sh
sys:x:3:3:sys:/dev:/bin/sh
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/bin/sh
man:x:6:12:man:/var/cache/man:/bin/sh
. . .
```

Notice how instead of giving the "!" command by itself, we've used used it as a prefix for a local shell command. This works to run any command available on our local machine and could have been used with the local "df" command earlier.

The UID will be in the third column of the file, as delineated by colon characters.

Similarly, we can change the group owner of a file with:

```
chgrp groupID file
```

Again, there is no easy way to get a listing of the remote system's groups. We can work around it with the following command:

```
get /etc/group
!less group

root:x:0:
daemon:x:1:
bin:x:2:
sys:x:3:
adm:x:4:
tty:x:5:
disk:x:6:
lp:x:7:
. . .
```

The third column holds the ID of the group associated with name in the first column. This is what we are looking for.

Thankfully, the "chmod" command works as expected on the remote file system:

```
chmod 777 publicFile

Changing mode on /home/demouser/publicFile
```

There is no command for manipulating local file permissions, but you can set the local umask, so that any files copied to the local system will have the appropriate permissions.

That can be done with the "lumask" command:

```
lumask 022

Local umask: 022
```

Now all regular files downloaded (as long as the "-p" flag is not used) will have 644 permissions.

SFTP allows you to create directories on both local and remote systems with "lmkdir" and "mkdir" respectively. These work as expected.

The rest of the file commands target only the remote filesystem:

```
ln
rm
rmdir
```

These commands replicate the basic behavior of the shell versions. If you need to perform these actions on the local file system, remember that you can drop into a shell by issuing this command:

```
!
```

Or execute a single command on the local system by prefacing the command with "!" like so:

```
!chmod 644 somefile
```

When you are finished with your SFTP session, use "exit" or "bye" to close the connection.

```
bye
```

Conclusion

Although SFTP is a simple tool, it is very useful for administrating servers and transferring files between them.

If you are used to using FTP or SCP to accomplish your transfers, SFTP is a good way to leverage the strengths of both. While it is not appropriate for every situation, it is a flexible tool to have in your repertoire.