UNIX Basics and Commands, PuTTY V1.0

Preface

This document will take you through UNIX architecture, directory structure and commands description.

Also it will give overview of PuTTY.

Table of Contents

The **File Transfer Protocol** (**FTP**) is one of the most common means of copying files between servers over the Internet. Most Web-based download sites use the built-in FTP capabilities of Web browsers, and, therefore, most server-oriented operating systems usually include an FTP server application as part of the software suite. Linux is no exception.

1. FTP OVERVIEW

FTP relies on a pair of TCP ports to get the job done. It operates using two connection channels:

1.1 FTP control channel, TCP Port 21 All commands you send, as well as

the FTP server's responses to those commands, go over the control connection, but any data sent back (such as ls directory lists or actual file data in either direction) will go over the data connection.

1.2 FTP data channel, TCP Port 20   This port is used for all subsequent

data transfers between the client and server.

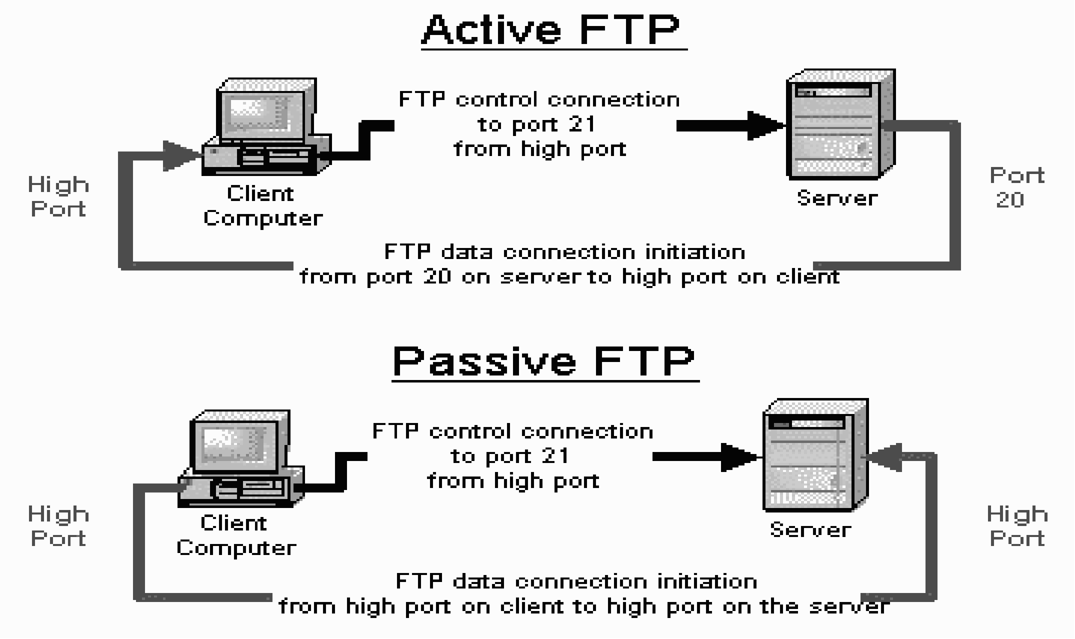
In addition to these channels, there are several varieties of FTP.

1.3 Types of FTP

From a networking perspective, the two main types of FTP are active and passive.

In **active FTP**, the FTP server initiates a data transfer connection back to the client. For **passive FTP,** the connection is initiated from the FTP client.

These are illustrated in the below Figure



From a user management perspective, there are two additional types of FTP: **regular FTP**, in which files are transferred using the username and password of a regular user FTP server, and **anonymous FTP,** in which general access is provided to the FTP server using a well known universal login method. Take a closer look at each type.

1.3.1 Active FTP

The sequence of events for active FTP is:

1. Your client connects to the FTP server by establishing an FTP control connection to port 21 of the server. Your commands such as ls and get are sent over this connection.

2. Whenever the client requests data over the control connection, the server initiates data transfer connections back to the client. The source port of these data transfer connections is always port 20 on the server, and the destination port is a high port (greater than 1024) on the client.

3. Thus the ls listing that you asked for comes back over the port 20 to high port connection, not the port 21 control connection. FTP active mode, therefore, transfers data in a counter intuitive way to the TCP standard, as it selects port 20 as its source port (not a random high port that's greater than 1024) and connects back to the client on a random high port that has been pre-negotiated on the port 21 control connection. Active FTP may fail in cases where the client is protected from the Internet via many to one NAT (masquerading), because the firewall will not know which of the many servers behind it should receive the return connection.

1.3.2 Passive FTP

Passive FTP works differently:

1. Your client connects to the FTP server by establishing an FTP control connection to port 21 of the server. Your commands such as ls and get are sent over that connection.

2. Whenever the client requests data over the control connection, the client initiates the data transfer connections to the server. The source port ofthese data transfer connections is always a high port on the client with a destination port of a high port on the server.

Passive FTP should be viewed as the server never making an active attempt to connect to the client for FTP data transfers. Because the client always initiates the required connections, passive FTP works better for clients protected by a firewall.

As Windows defaults to active FTP and Linux defaults to passive, you'll probably have to accommodate both forms when deciding upon a security policy for your FTP server.

1.3.3 Regular FTP

By default, the VSFTPD package allows regular Linux users to copy files to and from their home directories with an FTP client using their Linux usernames and passwords as their login credentials.

VSFTPD also has the option of allowing this type of access to only a group of Linux users, enabling you to restrict the addition of new files to your system to authorized personnel.

The disadvantage of regular FTP is that it isn't suitable for general download distribution of software as everyone either has to get a unique Linux user account or has to use a shared username and password. Anonymous FTP allows you to avoid this difficulty.

1.3.4 Anonymous FTP

Anonymous FTP is the choice of Web sites that need to exchange files with numerous unknown remote users. Common uses include downloading software updates and MP3s and uploading diagnostic information for a technical support engineers' attention. Unlike regular FTP where you login with a preconfigured Linux username and password, anonymous FTP requires only a username of anonymous and your e-mail address for the password. Once logged into a VSFTPD server, you automatically have access to only the default anonymous FTP directory (/var/ftp in the case of VSFTPD) and all its subdirectories.

2. HOW TO DOWNLOAD AND INSTALL VSFTPD

Most Red Hat and Fedora Linux software products are available in the RPM format. Downloading and installing RPMs isn't hard. It is best to use the latest version of VSFTPD.When searching for the file, remember that the VSFTPD RPM's filename usually starts with the word "vsftpd" followed by a version number, as in vsftpd-1.2.1-5.i386.rpm.

2.1 HOW TO GET VSFTPD STARTED

You can start, stop, or restart VSFTPD after booting using these commands:

```
[root@bigboy tmp]# service vsftpd start
[root@bigboy tmp]# service vsftpd stop
[root@bigboy tmp]# service vsftpd restart
```

To configure VSFTPD to start at boot, use the chkconfig command:

```
[root@bigboy tmp]# chkconfig vsftpd on
```

2.2 TESTING THE STATUS OF VSFTPD

You can always test whether the VSFTPD process is running by using the netstat -a command, which lists all the TCP and UDP ports on which the server is listening for traffic. This example shows the expected output:

```
[root@bigboy root]# netstat -a | grep ftp
tcp 0 0 *:ftp *:* LISTEN
[root@bigboy root]#
If VSFTPD wasn't running, there would be no output at all.
```

2.3 THE VSFTPD.CONF FILE

VSFTPD reads the contents of its vsftpd.conf configuration file only when it starts, so you'll have to restart VSFTPD each time you edit the file in order for the changes to take effect. This file uses a number of default settings you need to know about:

2.3.1 VSFTPD runs as an anonymous FTP server

Unless you want any remote user to log into to your default FTP directory using a username of anonymous and a password that's the same as their e-mail address, I suggest turning this off. You can set the configuration file's anonymous_enable directive to no to disable this feature. You'll also need to simultaneously enable local users to be able to log in by removing the comment symbol (#) before the local_enable instruction.

2.3.2 VSFTPD allows only anonymous FTP downloads to remote users, not uploads from them

You can change this by modifying the anon_upload_enable directive shown later.

2.3.3 VSFTPD doesn’t allow anonymous users to create directories on your FTP server

You can change this by modifying the anon_mkdir_write_enable directive.

2.3.4 VSFTPD logs FTP access to the /var/log/vsftpd.log log file

You can change this by modifying the xferlog_file directive.

2.3.5 VSFTPD expects files for anonymous FTP to be placed in the

/var/ftp directory: You can change this by modifying the anon_root directive. There is always the risk with anonymous FTP that users will discover a way to write files to your anonymous FTP directory. You run the risk of filling up your /var partition if you use the default setting. It is best to make the anonymous FTP directory reside in its own dedicated partition.

The configuration file is fairly straightforward as you can see in the snippet:

```
# Allow anonymous FTP?
anonymous_enable=YES
# Uncomment this to allow local users to log in.
local_enable=YES
# Uncomment this to enable any form of FTP write command.
# (Needed even if you want local users to be able to upload files)
write_enable=YES
# Uncomment to allow the anonymous FTP user to upload files. This only
# has an effect if global write enable is activated. Also, you will
# obviously need to create a directory writable by the FTP user.
#anon_upload_enable=YES
# Uncomment this if you want the anonymous FTP user to be able to
create
# new directories.
#anon_mkdir_write_enable=YES
# Activate logging of uploads/downloads.
xferlog_enable=YES
# You may override where the log file goes if you like.
# The default is shown# below.
#xferlog_file=/var/log/vsftpd.log
# The directory which vsftpd will try to change
# into after an anonymous login. (Default = /var/ftp)
#anon_root=/data/directory
```

To activate or deactivate a feature, remove or add the # at the beginning of the appropriate line.

2.4 Other *vsftpd.conf* Options

There are many other options you can add to this file:

Limiting the maximum number of client connections (max_clients)

Limiting the number of connections by source IP address (max_per_ip)

Setting the maximum rate of data transfer per anonymous login (anon_max_rate)

Setting the maximum rate of data transfer per non-anonymous login (local_max_rate)

Descriptions on this and more can be found in the vsftpd.conf man pages.

3. FTP SECURITY ISSUES

FTP has a number of security drawbacks, but you can overcome them in some cases. You can restrict an individual Linux user's access to non-anonymous FTP, and you can change the configuration to not display the FTP server's software version information, but unfortunately, though very convenient, FTP logins and data transfers are not encrypted.

3.1 The /etc/vsftpd.ftpusers File

For added security, you may restrict FTP access to certain users by adding them to the list of users in the /etc/vsftpd.ftpusers file. The VSFTPD package creates this file with a number of entries for privileged users that normally shouldn't have FTP access. As FTP doesn't encrypt passwords, thereby increasing the risk of data or passwords being compromised, it is a good idea to let these entries remain and add new entries for additional security.

3.2 Anonymous Upload

If you want remote users to write data to your FTP server, then you should create a write-only directory within /var/ftp/pub. This will allow your users to upload but not access other files uploaded by other users. The commands you need are:

```
[root@bigboy tmp]# mkdir /var/ftp/pub/upload
[root@bigboy tmp]# chmod 722 /var/ftp/pub/upload
```

4. FTP Greeting Banner

Change the default greeting banner in the vsftpd.conf file to make it harder for malicious users to determine the type of system you have. The directive in this file is:

```
ftpd_banner= New Banner Here
```

4.1 Using SCP as Secure Alternative to FTP

One of the disadvantages of FTP is that it does not encrypt your username and password. This could make your user account vulnerable to an unauthorized attack from a person eavesdropping on the network connection. **Secure Copy** (**SCP**) and **Secure FTP** (**SFTP**) provide encryption and could be considered as an alternative to FTP for trusted users. SCP does not support anonymous services, however, a feature that FTP does support.

5. Send and receive a file in FTP

To get files from the server onto your own computer use the get command as shown in the example below. In this example, you would get the file myfile.htm.

Tip: If you want to get more than one file use mget and wildcards, for example, if you wanted to get all files that end with .htm you could type mget *.htm. Finally, if you do not want to be prompted as each file is being sent make sure to type prompt to disable prompting.

5.1 get myfile.htm

To send a file from your computer to the computer you are connected to assuming you have the rights use the send command as shown in the example below. In this example, we are sending the myfile.htm to the directory we're currently in.

5.2 send myfile.htm

It is important to realize that the files being sent must be in your local working directory. In other words the directory you were in when you typed the FTP command. If you want to change to the directory that contains your files use the lcd command. For example, on Windows you'd type lcd c:\windows to set the local directory to the Windows directory.

6. FTP Commands

Depending upon the version of FTP and the operating system being used, each of the below commands may or may not work. Typing -help or a ? will list the commands available to you. Below is a general description of FTP commands available in the Windows command line FTP command

Command	**Information**
!	This command toggles back and forth between the operating system and ftp. Once back in the operating system, typing exit takes you back to the FTP command line.
?	Access the Help screen.
append	Append text to a local file.
ascii	Switch to ASCII transfer mode
bell	Turns bell mode on or off.
binary	Switches to binary transfer mode.
bye	Exits from FTP.

cd	Changes directory.
close	Exits from FTP.
delete	Deletes a file.
debug	Sets debugging on or off.
dir	Lists files if connected. dir -C = Will list the files in wide format. dir -1 = Lists the files in bare format in alphabetic order dir -r = Lists directory in reverse alphabetic order. dir -R = Lists all files in current directory and sub directories. dir -S = Lists files in bare format in alphabetic order.
disconnect	Exits from FTP.
get	Get file from the computer connected to.
glob	Sets globbing on or off. When turned off the file name in the put and get commands is taken literally and wildcards will not be looked at.
hash	Sets hash mark printing on or off. When turned on for each 1024 bytes of data received a hash-mark (#) is displayed.
help	Access the Help screen and displays information about command if command typed after help.
lcd	Displays local directory if typed alone or if path typed after lcd will change local directory.
literal	Sends a literal command to the connected computer with an expected one line response.
ls	Lists files of the remotely connected computer.
mdelete	Multiple delete.
mdir	Lists contents of multiple remote directories.
mget	Get multiple files.
mkdir	Make directory.
mls	Lists contents of multiple remote directories.