How To Install a FTP Server On Ubuntu Server 18.04

VsFTPD ” Very Secure FTP Daemon”

VsFTPD “very secure FTP daemon” is an open source FTP server for Linux systems, in this quick guide we will install VsFTPD on a Ubuntu server and secure the FTP server with SSL/TLS. Please visit the official website of VsFTPD if you need more information about the application.

Requirements

- Ubuntu Server 18.04
- User with sudo privileges.
- Static IP address
- Configured firewall
- Server connected to internet

For more information on how to create a sudo and configure a static IP please see the quick guides Create Sudo User , Set Static IP address and Configure Ubuntu Firewall.

Install VsFTPD

Vsftpd is available in Ubuntu 18.04 default repository and do not require any extra pre configuration.

Run the following command to install Vsftpd

```
sudo apt-get install vsftpd -y
```

Wait for the application to finish installing, start the Vsftpd service and enable it to start on boot.

```
sudo systemctl start vsftpd
sudo systemctl enable vsftpd
```

Verify that the VsFTPD is up and running.

```
sudo systemctl status vsftpd
```

```
root@iphone:~# sudo systemctl status vsftpd
● vsftpd.service - vsftpd FTP server
   Loaded: loaded (/lib/systemd/system/vsftpd.service; enabled; vendor preset:
enabled)
   Active: active (running) since Sat 2019-06-08 18:17:39 UTC; 2min 54s ago
 Main PID: 2311 (vsftpd)
```

```
     Tasks: 1 (limit: 2214)
    CGroup: /system.slice/vsftpd.service

Jun 08 18:17:39 iphone systemd[1]: Starting vsftpd FTP server...
Jun 08 18:17:39 iphone systemd[1]: Started vsftpd FTP server.
```

Configure The Firewall

We need to open port 20 and 21 for active FTP and ports 40000-50000 for passive FTP.

```
sudo ufw allow 20/tcp
```

```
sudo ufw allow 21/tcp
```

```
sudo ufw allow 40000:50000/tcp
```

Display the firewall rules.

```
sudo ufw status
```

```
root@iphone:~# sudo ufw status
Status: active

To                         Action      From
--                         ------      ----
22/tcp                     ALLOW       Anywhere
OpenSSH                    ALLOW       Anywhere
21/tcp                     ALLOW       Anywhere
40000:50000/tcp            ALLOW       Anywhere
```

```
22/tcp (v6)                ALLOW       Anywhere (v6)
OpenSSH (v6)               ALLOW       Anywhere (v6)
21/tcp (v6)                ALLOW       Anywhere (v6)
40000:50000/tcp (v6)       ALLOW       Anywhere (v6)

root@iphone:~#
```

Create FTP User

Create a low privileges user that can be used to access the FTP server.

When prompted enter password and user information for the user.

```
sudo adduser ftpuser
```

```
root@iphone:~# sudo adduser ftpuser
Adding user `ftpuser' ...
Adding new group `ftpuser' (1001) ...
Adding new user `ftpuser' (1001) with group `ftpuser' ...
Creating home directory `/home/ftpuser' ...
Copying files from `/etc/skel' ...
Enter new UNIX password:
Retype new UNIX password:
passwd: password updated successfully
Changing the user information for ftpuser
Enter the new value, or press ENTER for the default
        Full Name []:
        Room Number []:
        Work Phone []:
        Home Phone []:
        Other []:
Is the information correct? [Y/n] y
root@iphone:~#
```

Create a FTP Directory For The FTP User

First we want to create a FTP folder for the ftpuser.

```
sudo mkdir /home/ftpuser/ftp
```

Next we want to set the folder ownership.

```
sudo chown nobody:nogroup /home/ftpuser/ftp
```

Remove write permissions to the ftp folder.

```
sudo chmod a-w /home/ftpuser/ftp
```

Verify FTP folder permissions.

```
sudo ls -la /home/ftpuser/ftp
```

```
root@iphone:/home/ftpuser# sudo ls -la /home/ftpuser/ftp
total 8
dr-xr-xr-x 2 nobody  nogroup 4096 Jun  8 19:01 .
drwxr-xr-x 3 ftpuser ftpuser 4096 Jun  8 19:02 ..
root@iphone:/home/ftpuser#
```

Create a directory for file uploads and assign ownership to ftpuser.

```
sudo mkdir /home/ftpuser/ftp/files
sudo chown ftpuser:ftpuser /home/ftpuser/ftp/files
```

Verify the new folder permission.

```
sudo ls -la /home/ftpuser/ftp
```

```
root@iphone:/home/ftpuser/ftp/files# sudo ls -la /home/ftpuser/ftp
total 12
dr-xr-xr-x 3 nobody  nogroup 4096 Jun  8 19:08 .
drwxr-xr-x 3 ftpuser ftpuser 4096 Jun  8 19:02 ..
drwxr-xr-x 2 ftpuser ftpuser 4096 Jun  8 19:08 files
root@iphone:/home/ftpuser/ftp/files#
```

Create and add txt file to the files folder we created in the step above.

```
echo "Test create txt file" | sudo tee /home/ftpuser/ftp/files/txt01.txt
```

Configuring VsFTPD

Edit the VsFTPD configuration file vsftpd.conf

```
cd etc/
```

```
sudo nano vsftpd.conf
```

```
##
# Allow anonymous FTP? (Disabled by default).
anonymous_enable=NO
#
# Uncomment this to allow local users to log in.
local_enable=YES
##
```

Enable uploading to the FTP server by uncomment the write_enable parameter.

```
##
write_enable=YES
##
```

Prevent the FTP users from accessing files or to run commands outside there directory by uncomment the chroot_local_user=YES parameter.

```
##
chroot_local_user=YES
##
```

Scroll down to the bottom and add the the port range for passive FTP.

```
pasv_min_port=40000
pasv_max_port=50000
```

Previously we created a ftp/file directory and folder for the ftpsuser, now we need to configure VsFTPD to log the ftpuser to home ftp directory we created.

Add the line bellow.

```
user_sub_token=$USER
local_root=/home/$USER/ftp
```

Restart the daemon.

```
sudo systemctl restart vsftpd
```

Testing The FTP Access

You can use a ftp client like FileZilla or the command line to confirm that you can access the ftp server and that you can see the txt file you created in the ftpuser ftp directory.

I am using the command line on the FTP server in this example to confirm that i can access the FTP and that i can download the txt01.txt.

```
root@iphone:/# ftp 127.0.0.1
Connected to 127.0.0.1.
220 (vsFTPd 3.0.3)
Name (127.0.0.1:toor): ftpuser
331 Please specify the password.
Password:
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
```

```
ftp>
```

Lets confirm that we can change to the “files” directory.

```
ls
cd files
```

```
ftp> ls
200 PORT command successful. Consider using PASV.
150 Here comes the directory listing.
drwxr-xr-x    2 1001     1001         4096 Jun 08 19:17 files
226 Directory send OK.
ftp> cd files
250 Directory successfully changed.
ftp>
```

List the directory and use the get command to transfer the test file.

```
ls
get txt01.txt
```

```
ftp> ls
200 PORT command successful. Consider using PASV.
150 Here comes the directory listing.
-rw-r--r--    1 0        0              21 Jun 08 19:16 txt01.txt
226 Directory send OK.
ftp> get txt01.txt
local: txt01.txt remote: txt01.txt
200 PORT command successful. Consider using PASV.
150 Opening BINARY mode data connection for txt01.txt (21 bytes).
226 Transfer complete.
21 bytes received in 0.00 secs (259.5926 kB/s)
```

```
ftp>
```

Upload the file with a new name to test users write permissions. To upload a file we use the put command.

```
put txt01.txt txt01-upload.txt
```

```
ftp> put txt01.txt txt01-upload.txt
local: txt01.txt remote: txt01-upload.txt
200 PORT command successful. Consider using PASV.
150 Ok to send data.
226 Transfer complete.
21 bytes sent in 0.00 secs (1.0541 MB/s)
ftp>
```

Listing the files directory should show two files now.

```
ls
```

```
ftp> ls
200 PORT command successful. Consider using PASV.
150 Here comes the directory listing.
-rw-------    1 1001     1001           21 Jun 08 21:00 txt01-upload.txt
-rw-r--r--    1 0        0              21 Jun 08 19:16 txt01.txt
226 Directory send OK.
ftp>
```

(Optional) Secure The FTP Server With TLS

Lets start adding the firewall rule for TLS traffic, add port 990 to the firewall access list.

```
sudo ufw allow 990/tcp
```

```
root@iphone:/# sudo ufw allow 990/tcp
Rule added
Rule added (v6)
root@iphone:/#
```

Confirm firewall status

```
sudo ufw status
```

```
root@iphone:/# sudo ufw status
Status: active

To                         Action      From
--                         ------      ----
22/tcp                     ALLOW       Anywhere
OpenSSH                    ALLOW       Anywhere
21/tcp                     ALLOW       Anywhere
40000:50000/tcp            ALLOW       Anywhere
990/tcp                    ALLOW       Anywhere
22/tcp (v6)                ALLOW       Anywhere (v6)
OpenSSH (v6)               ALLOW       Anywhere (v6)
21/tcp (v6)                ALLOW       Anywhere (v6)
40000:50000/tcp (v6)       ALLOW       Anywhere (v6)
990/tcp (v6)               ALLOW       Anywhere (v6)
```

```
root@iphone:/#
```

Create a OpenSSL certificate

Create a OpenSSL certificate for TLS/SSL encryption, first make a directory where you can save the certificate.

```
sudo mkdir /etc/ftpcert
```

Now we will create a new certificate, use the -days flag to make it valid for two years, 730 days. Next set the bit value of the RSA key, i am running with a 2048-bit RSA key.

Type in the -keyout and the -out flag, the flags will set the key values for the private key and the certificate.

NOTE: Setting both flags with the same value will create both the private key and the certificate in the same file.

You will be asked to enter details like country, state, etc. You don’t have to fill in the information. Just keep pressing ENTER for defaults.

```
sudo openssl req -x509 -nodes -days 730 -newkey rsa:2048 -keyout
/etc/ftpcert/vsftpd.pem -out /etc/ftpcert/vsftpd.pem
```

Confirm that the private key and the certificate is the ftpscert directory.

```
root@iphone:/# cd /etc/ftpcert/
root@iphone:/etc/ftpcert# ls
```

```
vsftpd.pem
root@iphone:/etc/ftpcert#
```

Next we need to configure vsftpd to allow TLS/SSL traffic and point out the directory of the private key and the certificate , open the vsftpd configuration file with a editor.

```
cd etc/
sudo nano vsftpd.conf
```

Scroll down until you find the rsa parameters, Comment them out and replace them with new lines that points out the privet key and the certificate we created.

```
##
# rsa_cert_file=/etc/ssl/certs/ssl-cert-snakeoil.pem
# rsa_private_key_file=/etc/ssl/private/ssl-cert-snakeoil.key
##

rsa_cert_file=/etc/ftpcert/vsftpd.pem
rsa_private_key_file=/etc/ftpcert/vsftpd.pem
```

Configure FTP connections to use use SSL/TLS, change the ssl_enable=NO parameter to YES.

```
##
ssl_enable=YES
##
```

Now add the following lines to deny anonymous connections over SSL and to require SSL for logging and transferring data.

```
##
allow_anon_ssl=NO
force_local_data_ssl=YES
force_local_logins_ssl=YES
##
```

Configure the server to use the TLS protocol

```
##
ssl_tlsv1=YES
ssl_sslv2=NO
ssl_sslv3=NO
##
```

Last configure SSL reuse parameter to NO due that it can have conflicts with FTP clients, next we need to use high encryption cipher suite, which means that the key lengths is equal to or greater than 128 bits.

Paste thee lines below.

```
##
require_ssl_reuse=NO
ssl_ciphers=HIGH
##
```

The configuration should have the below entry’s configured.

```
#
rsa_cert_file=/etc/ftpcert/vsftpd.pem
rsa_private_key_file=/etc/ftpcert/vsftpd.pem
#
```

```
#
ssl_enable=YES
#
allow_anon_ssl=NO
force_local_data_ssl=YES
force_local_logins_ssl=YES
##
ssl_tlsv1=YES
ssl_sslv2=NO
ssl_sslv3=NO
#
require_ssl_reuse=NO
ssl_ciphers=HIGH
#
```

Restart the VsFTPD to load the new configuration.

```
sudo systemctl restart vsftpd
```

Confirm FTP TLS Configuration

Download a FTP client like FileZilla, you grab the FileZilla client from the official site https://filezilla-project.org/

Run and install the FTP client, when connecting to the FTP server use “Require explicit FTP over TLS”. If everything is configured correct you should be grated with a pop up windows that displays the server certificate we created.

If you try to connect to the FTP server with just plain FTP protocol, you will get an error and you wont be able to connect to the server.

```
Status: Connection established, waiting for welcome message...
Response:       220 (vsFTPd 3.0.3)
Command:        USER ftpuser
Response:       530 Non-anonymous sessions must use encryption.
```

```
Error:  Could not connect to server
```

Conclusion

In this quick guide we have installed a FTP server on Ubuntu18.04.02, generated a certificate with OpenSSL and secured the server connectivity with TLS.