Contents

Mark as Complete

Melissa Anderson

Subscribe Share

How To Set Up vsftpd for a User's Directory on Ubuntu 16.04

50

Updated February 20, 2018 447.1k LINUX BASICS SECURITY UBUNTU UBUNTU 16.04

Introduction

FTP, short for File Transfer Protocol, is a network protocol that was once widely used for moving files between a client and server. It has since been replaced by faster, more secure, and more convenient ways of delivering files. Many casual Internet users expect to download directly from their web browser with `https`, and command-line users are more likely to use secure protocols such as the `scp` or sFTP.

FTP is still used to support legacy applications and workflows with very specific needs. If you have a choice of what protocol to use, consider exploring the more modern options. When you do need FTP, however, vsftpd is an excellent choice. Optimized for security, performance, and stability, vsftpd offers strong protection against many security problems found in other FTP servers and is the default for many Linux distributions.

In this tutorial, we'll show you how to configure vsftpd to allow a user to upload files to his or her home directory using FTP with login credentials secured by SSL/TLS.

Prerequisites

To follow along with this tutorial you will need:

- **An Ubuntu 16.04 server with a non-root user with `sudo` privileges**: You can learn more about how to set up a user with these privileges in our Initial Server Setup with Ubuntu 16.04 guide.

Once you have an Ubuntu server in place, you're ready to begin.

Step 1 — Installing vsftpd

We'll start by updating our package list and installing the vsftpd daemon:

```
$ sudo apt-get update
$ sudo apt-get install vsftpd
```

When the installation is complete, we'll copy the configuration file so we can start with a blank configuration, saving the original as a backup.

```
$ sudo cp /etc/vsftpd.conf /etc/vsftpd.conf.orig
```

With a backup of the configuration in place, we're ready to configure the firewall.

Step 2 — Opening the Firewall

We'll check the firewall status to see if it's enabled. If so, we'll ensure that FTP traffic is permitted so you won't run into firewall rules blocking you when it comes time to test.

```
$ sudo ufw status
```

SCROLL TO TOP

Mark as Complete

In this case, only SSH is allowed through:

```
Output
Status: active

To Action  From
-- ------  ----
OpenSSH ALLOW   Anywhere
OpenSSH (v6)   ALLOW   Anywhere (v6)
```

You may have other rules in place or no firewall rules at all. Since only `ssh` traffic is permitted in this case, we'll need to add rules for FTP traffic.

We'll need to open ports 20 and 21 for FTP, port 990 for later when we enable TLS, and ports 40000-50000 for the range of passive ports we plan to set in the configuration file:

```
$ sudo ufw allow 20/tcp
$ sudo ufw allow 21/tcp
$ sudo ufw allow 990/tcp
$ sudo ufw allow 40000:50000/tcp
$ sudo ufw status
```

Now our firewall rules looks like:

```
Output
Status: active

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

With `vsftpd` installed and the necessary ports open, we're ready to proceed to the next step.

Step 3 — Preparing the User Directory

For this tutorial, we're going to create a user, but you may already have a user in need of FTP access. We'll take care to preserve an existing user's access to their data in the instructions that follow. Even so, we recommend you start with a new user until you've configured and tested your setup.

First, we'll add a test user:

```
$ sudo adduser sammy
```

Assign a password when prompted and feel free to press "ENTER" through the other prompts.

FTP is generally more secure when users are restricted to a specific directory. `vsftpd` accomplishes this with `chroot` jails. When `chroot` is enabled for local users, they are restricted to their home directory by default. However, because of the way `vsftpd` secures the directory, it must not be writable by the user. This is fine for a new user who should only connect via FTP, but an existing user may need to write to their ome folder if they also shell access.

SCROLL TO TOP

In this example, rather than removing write privileges from the home directory, we're will create an `ftp` directory to serve as the `chroot` and a writable `files` directory to hold the actual files.

Mark as Complete

Create the `ftp` folder, set its ownership, and be sure to remove write permissions with the following commands:

```
$ sudo mkdir /home/sammy/ftp
$ sudo chown nobody:nogroup /home/sammy/ftp
$ sudo chmod a-w /home/sammy/ftp
```

Let's verify the permissions:

```
$ sudo ls -la /home/sammy/ftp
```

```
Output
total 8
4 dr-xr-xr-x  2 nobody nogroup 4096 Aug 24 21:29 .
4 drwxr-xr-x 3 sammy  sammy   4096 Aug 24 21:29 ..
```

Next, we'll create the directory where files can be uploaded and assign ownership to the user:

```
$ sudo mkdir /home/sammy/ftp/files
$ sudo chown sammy:sammy /home/sammy/ftp/files
```

A permissions check on the `files` directory should return the following:

```
$ sudo ls -la /home/sammy/ftp
```

```
Output
total 12
dr-xr-xr-x 3 nobody nogroup 4096 Aug 26 14:01 .
drwxr-xr-x 3 sammy  sammy   4096 Aug 26 13:59 ..
drwxr-xr-x 2 sammy  sammy   4096 Aug 26 14:01 files
```

Finally, we'll add a `test.txt` file to use when we test later on:

```
$ echo "vsftpd test file" | sudo tee /home/sammy/ftp/files/test.txt
```

Now that we've secured the `ftp` directory and allowed the user access to the `files` directory, we'll turn our attention to configuration.

Step 4 — Configuring FTP Access

We're planning to allow a single user with a local shell account to connect with FTP. The two key settings for this are already set in `vsftpd.conf`. Start by opening the config file to verify that the settings in your configuration match those below:

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

/etc/vsftpd.conf

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

SCROLL TO TOP

```
local_enable=YES
. . .
```

Next we'll need to change some values in the file. In order to allow the user to upload files, we'll uncomment the `write_enable` setting so that we have:

/etc/vsftpd.conf

```
. . .
write_enable=YES
. . .
```

We'll also uncomment the chroot to prevent the FTP-connected user from accessing any files or commands outside the directory tree.

/etc/vsftpd.conf

```
. . .
chroot_local_user=YES
. . .
```

We'll add a `user_sub_token` in order to insert the username in our `local_root directory` path so our configuration will work for this user and any future users that might be added.

/etc/vsftpd.conf

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

We'll limit the range of ports that can be used for passive FTP to make sure enough connections are available:

/etc/vsftpd.conf

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

Note: We pre-opened the ports that we set here for the passive port range. If you change the values, be sure to update your firewall settings.

We will also add a directive telling `vsftpd` to listen on a particular port for incoming FTP connections:

/etc/vsftpd.conf

```
listen_port=45000
```

Since we're only planning to allow FTP access on a case-by-case basis, we'll set up the configuration so that access is given to a user only when they are explicitly added to a list rather than by default:

/etc/vsftpd.conf

```
userlist_enable=YES
userlist_file=/etc/vsftpd.userlist
userlist_deny=NO
```

`userlist_deny` toggles the logic. When it is set to "YES", users on the list are denied FTP access. When it is set to "NO", only users on the list are allowed access. When you're done making the change, save and exit the file.

Finally, we'll create and add our user to the file. We'll use the `-a` flag to append to file:

SCROLL TO TOP

```
$ echo "sammy" | sudo tee -a /etc/vsftpd.userlist
```

Contents

Mark as Complete

Double-check that it was added as you expected:

```
cat /etc/vsftpd.userlist
```

```
Output
sammy
```

Restart the daemon to load the configuration changes:

```
$ sudo systemctl restart vsftpd
```

Now we're ready for testing.

Step 5 — Testing FTP Access

We've configured the server to allow only the user `sammy` to connect via FTP. Let's make sure that's the case.

Anonymous users should fail to connect: We disabled anonymous access. Here we'll test that by trying to connect anonymously. If we've done it properly, anonymous users should be denied permission:

```
$ ftp -p 203.0.113.0
```

```
Output
Connected to 203.0.113.0.
220 (vsFTPd 3.0.3)
Name (203.0.113.0:default): anonymous
530 Permission denied.
ftp: Login failed.
ftp>
```

Close the connection:

```
ftp> bye
```

Users other than `sammy` should fail to connect: Next, we'll try connecting as our `sudo` user. They, too, should be denied access, and it should happen before they're allowed to enter their password.

```
$ ftp -p 203.0.113.0
```

```
Output
Connected to 203.0.113.0.
220 (vsFTPd 3.0.3)
Name (203.0.113.0:default): sudo_user
530 Permission denied.
ftp: Login failed.
ftp>
```

Close the connection:

```
ftp> bye
```

SCROLL TO TOP

`sammy` should be able to connect, as well as read and write files: Here, we'll make sure that our designated user *can*connect:

```
$ ftp -p 203.0.113.0
```

```
Output
Connected to 203.0.113.0.
220 (vsFTPd 3.0.3)
Name (203.0.113.0:default): sammy
331 Please specify the password.
Password: your_user's_password
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp>
```

We'll change into the `files` directory, then use the `get` command to transfer the test file we created earlier to our local machine:

```
ftp> cd files
ftp> get test.txt
```

```
Output
227 Entering Passive Mode (203,0,113,0,169,12).
150 Opening BINARY mode data connection for test.txt (16 bytes).
226 Transfer complete.
16 bytes received in 0.0101 seconds (1588 bytes/s)
ftp>
```

We'll turn right back around and try to upload the file with a new name to test write permissions:

```
ftp> put test.txt upload.txt
```

```
Output
227 Entering Passive Mode (203,0,113,0,164,71).
150 Ok to send data.
226 Transfer complete.
16 bytes sent in 0.000894 seconds (17897 bytes/s)
```

Close the connection:

```
ftp> bye
```

Now that we've tested our configuration, we'll take steps to further secure our server.

Step 6 — Securing Transactions

Since FTP does *not* encrypt any data in transit, including user credentials, we'll enable TTL/SSL to provide that encryption. The first step is to create the SSL certificates for use with vsftpd.

We'll use `openssl` to create a new certificate and use the `-days` flag to make it valid for one year. In the same command, we'll add a private 2048-bit RSA key. Then by setting both the `-keyout` and `-out` flags to the same value, the private key and the certificate will be located in the same file.

We'll do this with the following command:

Contents

Mark as Complete

SCROLL TO TOP

```
$ sudo openssl req -x509 -nodes -days 365 -newkey rsa:2048 -keyout /etc/ssl/private/vsftpd.pem
```

Mark as Complete

You'll be prompted to provide address information for your certificate. Substitute your own information for the questions below:

```
Output
Generating a 2048 bit RSA private key
............................................................................+++
..........+++
writing new private key to '/etc/ssl/private/vsftpd.pem'
-----
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:US
State or Province Name (full name) [Some-State]:NY
Locality Name (eg, city) []:New York City
Organization Name (eg, company) [Internet Widgits Pty Ltd]:DigitalOcean
Organizational Unit Name (eg, section) []:
Common Name (e.g. server FQDN or YOUR name) []: your_IP_address
Email Address []:
```

For more detailed information about the certificate flags, see OpenSSL Essentials: Working with SSL Certificates, Private Keys and CSRs

Once you've created the certificates, open the `vsftpd` configuration file again:

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

Toward the bottom of the file, you should two lines that begin with `rsa_`. Comment them out so they look like:

/etc/vsftpd.conf

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

Below them, add the following lines which point to the certificate and private key we just created:

/etc/vsftpd.conf

```
rsa_cert_file=/etc/ssl/private/vsftpd.pem
rsa_private_key_file=/etc/ssl/private/vsftpd.pem
```

After that, we will force the use of SSL, which will prevent clients that can't deal with TLS from connecting. This is necessary in order to ensure all traffic is encrypted but may force your FTP user to change clients. Change `ssl_enable` to `YES`:

/etc/vsftpd.conf

```
ssl_enable=YES
```

After that, add the following lines to explicitly deny anonymous connections over SSL and to require SSL for both data transfer and logins:

/etc/vsftpd.conf

SCROLL TO TOP

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

Mark as Complete

After this we'll configure the server to use TLS, the preferred successor to SSL by adding the following lines:

/etc/vsftpd.conf

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

Finally, we will add two more options. First, we will not require SSL reuse because it can break many FTP clients. We will require "high" encryption cipher suites, which currently means key lengths equal to or greater than 128 bits:

/etc/vsftpd.conf

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

When you're done, save and close the file.

Now, we need to restart the server for the changes to take effect:

```
$ sudo systemctl restart vsftpd
```

At this point, we will no longer be able to connect with an insecure command-line client. If we tried, we'd see something like:

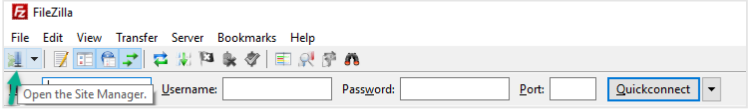
```
$ ftp -p 203.0.113.0
$ Connected to 203.0.113.0.
$ 220 (vsFTPd 3.0.3)
$ Name (203.0.113.0:default): sammy
$ 530 Non-anonymous sessions must use encryption.
$ ftp: Login failed.
$ 421 Service not available, remote server has closed connection
$ ftp>
```

Next, we'll verify that we can connect using a client that supports TLS.

Step 7 — Testing TLS with FileZilla

Most modern FTP clients can be configured to use TLS encryption. We will demonstrate how to connect using FileZilla because of its cross platform support. Consult the documentation for other clients.

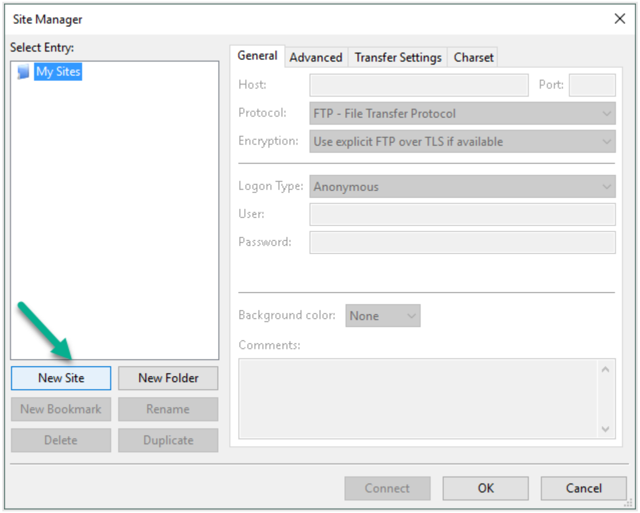
When you first open FileZilla, find the Site Manager icon just below the word File, the left-most icon on the top row. Click it:



A new window will open. Click the "New Site" button in the bottom right corner:

SCROLL TO TOP

Mark as Complete



Protocol: FTP - File Transfer Protocol

Encryption: Use explicit FTP over TLS if available

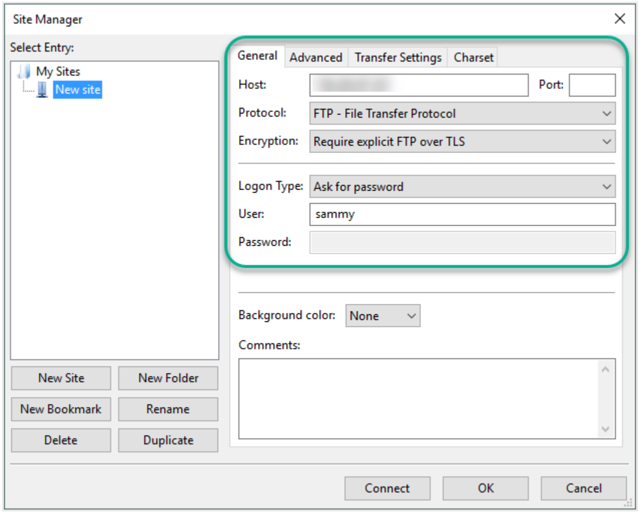
Logon Type: Anonymous

Background color: None

Under "My Sites" a new icon with the words "New site" will appear. You can name it now or return later and use the Rename button.

You must fill out the "Host" field with the name or IP address. Under the "Encryption" drop down menu, select "Require explicit FTP over TLS". You also want to specify that Filezilla should use port 45000 by filling out the “Port” field.

For "Logon Type", select "Ask for password". Fill in the FTP user you created in the "User" field:



Host: [redacted]

Protocol: FTP - File Transfer Protocol

Encryption: Require explicit FTP over TLS

Logon Type: Ask for password

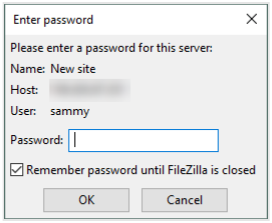
User: sammy

Background color: None

Click "Connect" at the bottom of the interface. You will be asked for the user's password:

SCROLL TO TOP

Mark as Complete

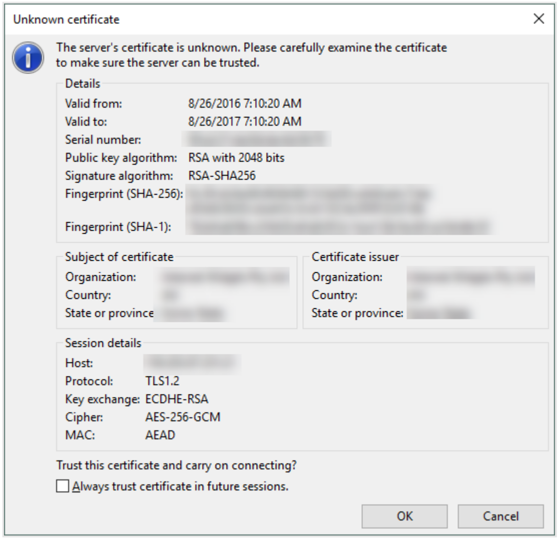


Enter password

Please enter a password for this server:
Name: New site
Host: [redacted]
User: sammy
Password:

☑ Remember password until FileZilla is closed

Click "OK" to connect. You should now be connected with your server with TLS/SSL encryption.



Unknown certificate

The server's certificate is unknown. Please carefully examine the certificate to make sure the server can be trusted.

Details
Valid from: 8/26/2016 7:10:20 AM
Valid to: 8/26/2017 7:10:20 AM
Serial number: [redacted]
Public key algorithm: RSA with 2048 bits
Signature algorithm: RSA-SHA256
Fingerprint (SHA-256): [redacted]
Fingerprint (SHA-1): [redacted]

Subject of certificate
Organization: [redacted]
Country: [redacted]
State or province: [redacted]

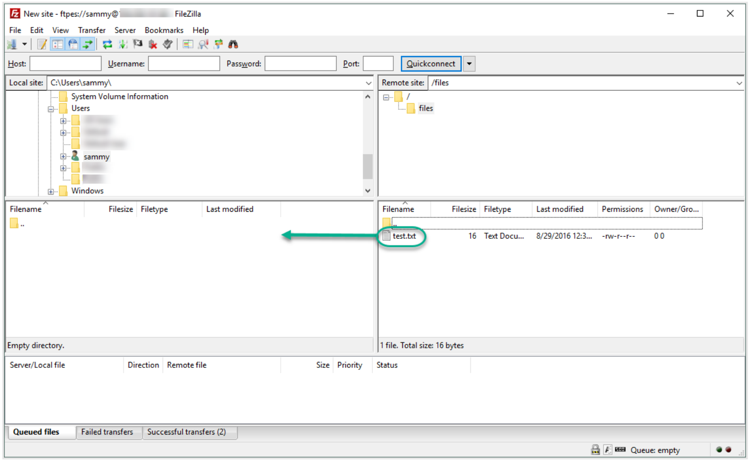
Certificate issuer
Organization: [redacted]
Country: [redacted]
State or province: [redacted]

Session details
Host: [redacted]
Protocol: TLS1.2
Key exchange: ECDHE-RSA
Cipher: AES-256-GCM
MAC: AEAD

Trust this certificate and carry on connecting?

□ Always trust certificate in future sessions.

When you've accepted the certificate, double-click the `files` folder and drag upload.txt to the left to confirm that you're able to download files.



New site - ftpes://sammy@[redacted] FileZilla

Local site: C:\Users\sammy\

Remote site: /files

Filename	Filesize	Filetype	Last modified	Permissions	Owner/Gro...
..					
test.txt	16	Text Docu...	8/29/2016 12:3...	-rw-r--r--	0 0

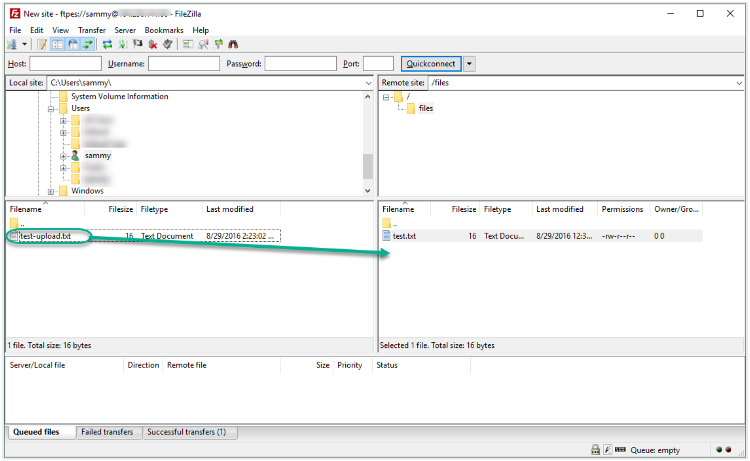
Empty directory.

1 file. Total size: 16 bytes

When you've done that, right-click on the local copy, rename it to upload-tls.txt` and drag it back to the erver to confirm that you can upload files.

SCROLL TO TOP

Mark as Complete



New site - ftpes://sammy@[redacted] - FileZilla

Filename	Filesize	Filetype	Last modified
..			
test-upload.txt	16	Text Document	8/29/2016 2:23:02 ...

1 file. Total size: 16 bytes

Filename	Filesize	Filetype	Last modified	Permissions	Owner/Gro...
..					
test.txt	16	Text Docu...	8/29/2016 12:3...	-rw-r--r--	0 0

Selected 1 file. Total size: 16 bytes

You've now confirmed that you can securely and successfully transfer files with SSL/TLS enabled.

Step 8 — Disabling Shell Access (Optional)

If you're unable to use TLS because of client requirements, you can gain some security by disabling the FTP user's ability to log in any other way. One relatively straightforward way to prevent it is by creating a custom shell. This will not provide any encryption, but it will limit the access of a compromised account to files accessible by FTP.

First, open a file called `ftponly` in the bin directory:

```
$ sudo nano /bin/ftponly
```

We'll add a message telling the user why they are unable to log in. Paste in the following:

```
#!/bin/sh
echo "This account is limited to FTP access only."
```

Change the permissions to make the file executable:

```
$ sudo chmod a+x /bin/ftponly
```

Open the list of valid shells:

```
$ sudo nano /etc/shells
```

At the bottom, add:

/etc/shells

```
. . .
/bin/ftponly
```

Update the user's shell with the following command:

```
$ sudo usermod sammy -s /bin/ftponly
```

SCROLL TO TOP

Now try logging in as sammy:

```
$ ssh sammy@203.0.113.0
```

You should see something like:

```
Output
This account is limited to FTP access only.
Connection to 203.0.113.0 closed.
```

This confirms that the user can no longer `ssh` to the server and is limited to FTP access only.

Mark as Complete

Conclusion

In this tutorial we covered setting up FTP for users with a local account. If you need to use an external authentication source, you might want to look into vsftpd's support of virtual users. This offers a rich set of options through the use of PAM, the Pluggable Authentication Modules, and is a good choice if you manage users in another system such as LDAP or Kerberos.

Melissa Anderson

♡ Upvote (50) Subscribe Share

Related Tutorials

How to Add and Delete Users on Ubuntu 16.04

How To Download Software and Content onto your Linux VPS

An Introduction to Useful Bash Aliases and Functions

An Introduction To Regular Expressions

How To Use Bash History Commands and Expansions on a Linux VPS

73 Comments

Leave a comment...

SCROLL TO TOP

Mark as Complete

Log In to Comment

remzi *September 7, 2016*

Thats good tutorials.

However I want to learn why dont use nologin instead of /bin/ftponly - "Step 8 — Disabling Shell Access (Optional) "?

MelissaAnderson MOD *September 7, 2016*

Privileged system accounts use the nologin shell, so I was reluctant to add it to the list of valid shells. I also wanted an explicit message provided to a user who tried to log in. If nologin were used for both types of account, I didn't see a way to clearly communicate the ftp-only limitation to the user.

uksitebuilder *September 15, 2016*

This works for Ubuntu 14.04 LTS also ?

MelissaAnderson MOD *September 15, 2016*

It should work as written except when you restart the daemon. Instead of using `systemctl`, you'll need to use `service`.

So, when you see:

```
sudo systemctl restart vsftpd
```

use this instead:

```
sudo service vsftpd restart
```

If you give it a try, let me know how it goes!

astienback *September 18, 2016*

I disable shell access but it prevents my ftp user from logging:
530 login incorrect
I restore **/bin/bash** as my ftp user's shell and all returns OK.
The likely explanation is within the ftponly shell which makes the user non local, even if /bin/ftponly was registered as a valid shell.
Any suggestion?

MelissaAnderson MOD *September 19, 2016*

I've tested the instructions for disabling shell access on Ubuntu 14.04 and 16.04 and can verify that they work on both distros, although output from 14.04 is a little different. In both cases, the user can log in with FileZilla using TLS and cannot log in from the shell.

Let me know which distro are you using and what FTP client are you connecting with and I'll see if I can lend a hand.

astienback *September 20, 2016*

Thanks for the testing Melissa. My ftp server is on a **ubuntu 16.04 server** very last version. My ftp client is the stock ftp client, right from the command line on another Ubuntu desktop. That works

SCROLL TO TOP

plainly with any external ftp servers. I'm used to get all ftp transfers via mc UI, successfully. My main purpose is to automate the upload of some files, using a script. That's why log in from the shell is a must.
Regards.

Mark as Complete

MelissaAnderson MOD *September 20, 2016*

As long as a shell in `/etc/shells` matches the user's shell in `/etc/passwd`, I can log in as my ftp user. The shell doesn't even have to exist.

You can use debugging flags from the ftp client when you log in to get more information about the failure:

```
$ ftp -dv server_address_or_ip
```

I also ran across this article that suggests looking for `pam_service_name=vsftpd` in the `/etc/vsftpd.conf` file and changing it to `pam_service_name=ftp`, then restarting daemon, which might be worth a try. Either setting works with my configuration, so I'm not sure it would make a difference.

Another alternative which I haven't configured before is configuring vsftpd virtual users, described here: https://help.ubuntu.com/community/vsftpd.

astienback *September 20, 2016*

Fixed it. Thanks for the tip about **pam***service***name** setting. The link was pretty helpful. I didn't run across that tuto when googling.
Regards.

dercampus *October 27, 2016*

Great tutorial, Melissa!

But how am I able to setup FTP to get access to my Wordpress files?

wimar *September 22, 2017*

I'm doing the same thing for my girlfriend's website currently. She is called Laura so let's assume the user with ftp access to the Wordpress files is called laura. If you follow along with the tutorial you should end up with an ftp folder in the home directory of laura (or sammy as this tutorial named the user). Within the ftp folder you than have a files folder:

```
/home/laura/ftp/
4 dr-xr-xr-x 3 nobody nogroup 4096 Sep 21 18:22 ftp
```

```
/home/laura/ftp/files
4 drwxr-xr-x 2 laura laura 4096 Sep 22 17:33 files
```

Now place the entire Wordpress installation in the files folder

`/home/laura/ftp/files/wordpress`

and make laura the owner and group of the Wordpress installation like so

`chown -R laura:laura /home/laura/ftp/files/wordpress`

At this point you will have an ftp directory that can be reached by your intended user (laura in my case). You will have a files folder in that ftp folder that the intended user (laura) can edit and upload stuff to and in there is the Wordpress installation.

All that is left than is to point the webserver at the folder that the Wordpress installation resides in. I am using Nginx so I would point the configuration file of my girlfriends website at

```
server {
  listen 80;
```

SCROLL TO TOP

```
  listen [::]:80;

  root /home/laura/ftp/files/wordpress;
...
```

Mark as Complete

crmvogel *November 4, 2016*

Great Tutorial, but in get **connection timed out** and **failed to retrieve directory listing** in FileZilla.
Can you give me some advice how to fix this please

thx

dylanh724 *September 30, 2017*

me too! Gah

davut *November 12, 2016*

Hi Melissa, thanks for this great tutorial. However, I want to disable access of ftp user to the other folders. Is it possible?

golabs *December 30, 2016*

Thanks for this tutorial - very helpful for a Linux noob like me :)

Is it perhaps an idea to add an additional section to the tutorial in which you explain how the added SFTP user "sammy" can also access the /var/www/html directory via SFTP?

I found a post about how to get this working and am testing it at the moment on my LAMP server:

http://www.ducea.com/2006/07/27/allowing-ftp-access-to-files-outside-the-home-directory-chroot/

I used the following commands:

```
mkdir /home/sammy/ftp/www_html
mount --bind /var/www/html/ /home/sammy/ftp/www_html
sudo nano /etc/fstab
```

and then added the following line to that fstab:

```
/var/www/html   /home/sammy/ftp/www_html    none    bind    0       0
```

So far I haven't been able to transfer files so if you have any suggestions, feel free to give me a pointer :D

sturi *May 4, 2017*

I had the same problem. I solved this!

```
mkdir /home/sammy/ftp/www_html
chown sammy:sammy /home/sammy/ftp/www_html
chown -R sammy:sammy /var/www/html/
mount --bind /var/www/html/ /home/sammy/ftp/www_html
```

But I don't know if is better solution.
Perhaps you can add sammy to www-data group, I hope someone more skilled than me can help.

jcarlos17 *June 20, 2017*

SCROLL TO TOP

I did it, but I can't see the `www_html` folder in the FTP client.
There is something missing?

Mark as Complete

Luova *July 17, 2017*

After doing this, I was unable to view any files but then I changed the ownership of the folder I needed the user to have access to and now when I FTP, I get unable to connect to the server and GnuTLS error -15: An unexpected TLS packet was received.

gerngeschehen *January 31, 2017*

Thanks for the great tutorial! I managed to set up my FTP server step by step following this tutorial.

However, there's one thing that's bothering me. When I tried to connect using explicit FTP over TLS, the client couldn't proceed and was stuck at "AUTH TLS". The certificate confirmation window never popped up. But on my server itself, I can connect properly with the same configuration on the client side, with 127.0.0.1 as the server address. And it also worked fine when I tried to connect to another computer within the same router.

So I guess it's because something was being naughty on my router or so? Could anyone give me some heads-up, as I am new to Linux administration and don't know where to look at. Thanks.

steffex *January 31, 2017*

Please note that the current method described at step 8 is not actually disabling ssh access for that user. It still logs the user in and shows your current motd, with could show information (like ip addresses) you don't want displayed, if you haven't changed it!

You can disable this message on a user-basis, by adding an empty file called `.hushlogin` in the users home dir.

Also if you don't want to show a custom message to the user when logging in using ssh, you can use `/bin/false` instead of `/bin/ftponly`

jarreau2001 *March 3, 2017*

This tutorial is awesome! Works great!
Thanks

jarreau2001 *March 7, 2017*

Followed this tutorial for my Ubuntu 14.04.1 LTS machine. I'm able to get FTP access across my LAN perfectly. When I attempt to connect from outside my LAN, I get a message that says, "the machine actively refused the connection". To me that indicates a router config issue but it's stumping me so I wanted to post here to see if it could be anything system related.

jarreau2001 *March 13, 2017*

According to this article `http://www.linuxquestions.org/questions/slackware-14/vsftpd-server-sent-passive-reply-with-unroutable-address-using-server-address-instead-4175575363/` FTP and NAT don't jive well so if you've followed this tutorial and require NAT to open up your server to public traffic, please be aware. Hopefully you won't waste a week trying to figure it out like I did.

jarreau2001 *March 13, 2017*

Just to sum up my experience here, since I've posted several comments:
The tutorial is pretty easy to follow, none of the steps threw errors, I was able to log into FTP via LAN. I was not able to log in from outside my LAN. I initially thought it was my router. After checking it and re-configuring the router settings 5 times, I paid a router specialist to verify the settings. The router was forwarding the port successfully NAT. After further research I found there are major issues with NAT and FTP based on how the system accepts requests and establishes sessions. There are some work arounds and hacks available to force it to work but they all seem really shady duct tape and bubble gum style patches. I've been exploring them all and each one just generates a different error message during failed login attempts. I am resolved to wipe out VSFTP all together and find another way to have FTP with users

SCROLL TO TOP

who can only upload / download from one specific directory. 2 weeks wasted on this and am starting from the drawing board all over!

Contents

Mark as Complete

wimar *September 22, 2017*

This does not sound like your issue is in any way related to this tutorial. As far as I can tell the FTP protocol itself does not jive well with NAT. Maybe you could try other means of letting your users gain access to files on your server. Or perhaps you could setup a dedicated server in your network (Raspberry Pi?) that is demilitarized (DMZ) and run the FTP server on there.

I understand your pain, as for example Wordpress still forces you to use FTP(S) if you would like a secure installation or if you don't want to mess with permissions on the server, but at the same time I don't think running an FTP server behind a routing device is a good idea. For you say so yourself, it does not work well.

My advice would be to either move the FTP server out from behind the NAT (if at all possible) or to use other means of letting your users gain access to their files (like SFTP). Hope that helps.

pokepud3 *March 24, 2017*

So I'm on ubuntu 16 LTS and when I get to this step:

```
ftp> get test.txt
local: test.txt remote: test.txt
local: test.txt: Permission denied
ftp> ls
150 Here comes the directory listing.
-rw-r--r-- 1 0 0 17 Mar 24 16:40 test.txt
```

It denies me from going further. I followed the steps, pretty much step by step and can't get past this one step. Any ideas?

vijfathurahman *October 23, 2017*

I get 550 failed to open file

```
ftp> get test.txt
local: test.txt remote: test.txt
227 Entering Passive Mode (139,59,237,73,186,78).
550 Failed to open file.
```

do you solve that already?

nicholasjohn16 *May 9, 2017*

This was working great until I started to receive this error message in Firefox:

Error: GnuTLS error -15: An unexpected TLS packet was received.

Not sure what caused it or why it starts and now I'm stuck.

Luova *July 17, 2017*

Did you figure this one out?

fischernick *July 23, 2017*

I've figured out the solution. In my case, the root of the problem was the wrong chroot settings. I've just commented out the following lines:

```
#chroot_list_enable=YES
#chroot_list_file=/etc/vsftpd.chroot_list
```

But ensure that the following is still enabled:

SCROLL TO TOP

```
chroot_local_user=YES
```

Contents

Mark as Complete

0 NorCalTechSupport *May 19, 2017*

Ubuntu 16.04.2 And everything works up until the last step in section 5: ftp>put test.txt upload.txt
227 Entering Passive Mode (10.1.1.183,169,207).
553 Could not create file.

This is for the Sammy user. Any ideas would be helpful. Thanks in advance.

0 iangrainger *October 9, 2017*

Me too :(Step 5, can't create a file.

I wonder if it's because i'm running as root - this being my first ever droplet?

0 rakeshmali *May 20, 2017*

I have followed all steps. My FTP setup using Filzilla is successfully established but now, when I am accessing my server using IP address though browser, it is not working at all And I am using apache server on it.
Could you please let me know why this stopped to work after setup of vsftpd?

1 thienhaxanh2405 *May 21, 2017*

Great tutorial.

In some cases, you need add "allow*writeable*chroot" to vsftpd.conf.

```
allow_writeable_chroot=YES
```

Hope it helpful.

0 SurfBlue714 *May 22, 2017*

Thanks Ms. Anderson, easily one of the more detailed vsftpd tutorials I've been able to find.

Although I am curious. How would one go a step farther and set it up to be browser accessible via a sub domain akin to this?

0 almarazrodolfo *June 20, 2017*

This tutorial us amazing everything worked, but I have a question How can I setup an extradrive to save files in other location, right now everything is storing in SSD of low capacity, I want that the ftp users use the other extradrive.

regards

Rod

Load More Comments

SCROLL TO TOP

Contents

Mark as Complete

Community Tutorials Questions Projects Tags Newsletter RSS

Distros & One-Click Apps Terms, Privacy, & Copyright Security Report a Bug Write for DOnations Shop

SCROLL TO TOP