Setting Up a Secure FTP Server on Ubuntu (Using vsftpd)

Written By: *Saikat Goswami*

---

1. Why FTP Still Matters (and When to Use It)

FTP (File Transfer Protocol) is one of the oldest ways to move files between machines. While tools like **SCP**, **SFTP**, and **cloud storage** are popular today, FTP is still widely used in:

- Legacy systems
- Shared hosting environments
- Automated file uploads/downloads
- Simple internal file distribution

That said, **plain FTP is insecure** because it sends usernames and passwords in clear text. That's why in this guide we'll focus on **securing FTP using TLS**, resulting in **FTPS**.

We'll use **vsftpd** (*Very Secure FTP Daemon*), which is:

- Fast
- Lightweight
- Actively maintained
- Designed with security in mind

---

2. What You'll Build by the End of This Guide

By the end, you'll have:

- A working FTP server on Ubuntu
- Local users restricted to their home directories
- Encrypted connections using TLS
- Firewall rules properly configured
- A setup suitable for production or lab use

---

3. Prerequisites

Before we start, make sure you have:

- Ubuntu 20.04 / 22.04 / 24.04
- A user with `sudo` privileges
- Basic command-line familiarity
- Port access (especially if using a cloud VM)

All commands below assume:

```
sudo -i
```

(or prefix commands with `sudo`)

---

4. Installing vsftpd

First, update your package index:

```
apt update
```

Now install vsftpd:

```
apt install vsftpd -y
```

Check that the service is running:

```
systemctl status vsftpd
```

You should see something like:

Active: active (running)

Enable it to start on boot:

```
systemctl enable vsftpd
```

---

5. Understanding the vsftpd Configuration File

The main configuration file is:

```
/etc/vsftpd.conf
```

Before editing, **always back it up**:

```
cp /etc/vsftpd.conf /etc/vsftpd.conf.bak
```

This single file controls authentication, permissions, encryption, and behavior.

---

6. Creating an FTP User

For security, avoid using `root` or admin accounts.

Create a dedicated FTP user:

```
adduser ftpuser
```

Set a password and follow the prompts.

This user's home directory will be:

```
/home/ftpuser
```

We'll later restrict this user so they **cannot escape this directory**.

---

7. Basic vsftpd Configuration (Secure Defaults)

Open the config file:

```
nano /etc/vsftpd.conf
```

Update or ensure the following settings exist:

```
# Disable anonymous FTP
anonymous_enable=NO

# Allow local users
local_enable=YES

# Enable file uploads and changes
write_enable=YES

# Set default permissions
local_umask=022

# Show messages when entering directories
dirmessage_enable=YES

# Log FTP activity
xferlog_enable=YES
```

.

What this does:

- Only authenticated users can log in
- Users can upload/download files
- Anonymous access is disabled

---

8. Chroot Users (Critical Security Step)

By default, an FTP user might browse the filesystem. This is dangerous.

Enable **chroot jail** (locks users into their home directory):

```
chroot_local_user=YES
```

vsftpd does not allow writing to a chroot directory by default.

To safely allow this, add:

```
allow_writeable_chroot=YES
```

Now users are **sandboxed** and can't explore your server. It means:

- **Isolation**: Users are confined to their home directory (the chroot jail).
- **Limited access**: They cannot move outside that directory to browse or modify other parts of the filesystem.
- **Safety**: Even if a user tries malicious commands, they are restricted to the sandbox and cannot harm the wider server.

---

9. Enabling FTPS (FTP over TLS)

This is the most important part.

9.1 Create a Self-Signed SSL Certificate

```
openssl req -x509 -nodes -days 365 \
  -newkey rsa:2048 \
  -keyout /etc/ssl/private/vsftpd.key \
  -out /etc/ssl/certs/vsftpd.crt
```

When prompted, enter your server details.

9.2 Configure TLS in vsftpd

Add these lines to `/etc/vsftpd.conf`:

```
# Enable TLS
ssl_enable=YES

# Certificate files
rsa_cert_file=/etc/ssl/certs/vsftpd.crt
rsa_private_key_file=/etc/ssl/private/vsftpd.key

# Enforce TLS
```

```
force_local_data_ssl=YES
force_local_logins_ssl=YES

# Disable insecure SSL
ssl_sslv2=NO
ssl_sslv3=NO

# Prefer strong ciphers
ssl_ciphers=HIGH
```

This ensures:

- Passwords are encrypted
- Data transfers are encrypted
- Weak SSL versions are disabled

---

10. Passive Mode Configuration (Very Important)

Most FTP clients use **passive mode**, especially behind NAT or firewalls.

Add this to your `vsftpd.conf` file :

```
pasv_enable=YES
pasv_min_port=40000
pasv_max_port=40100
```

This limits FTP data connections to a known port range.

---

11. Firewall Configuration (UFW)

If UFW is enabled, allow FTP traffic:

```
ufw allow 21/tcp
ufw allow 40000:40100/tcp
```

Reload firewall rules:

```
ufw reload
```

Check status:

```
ufw status
```

---

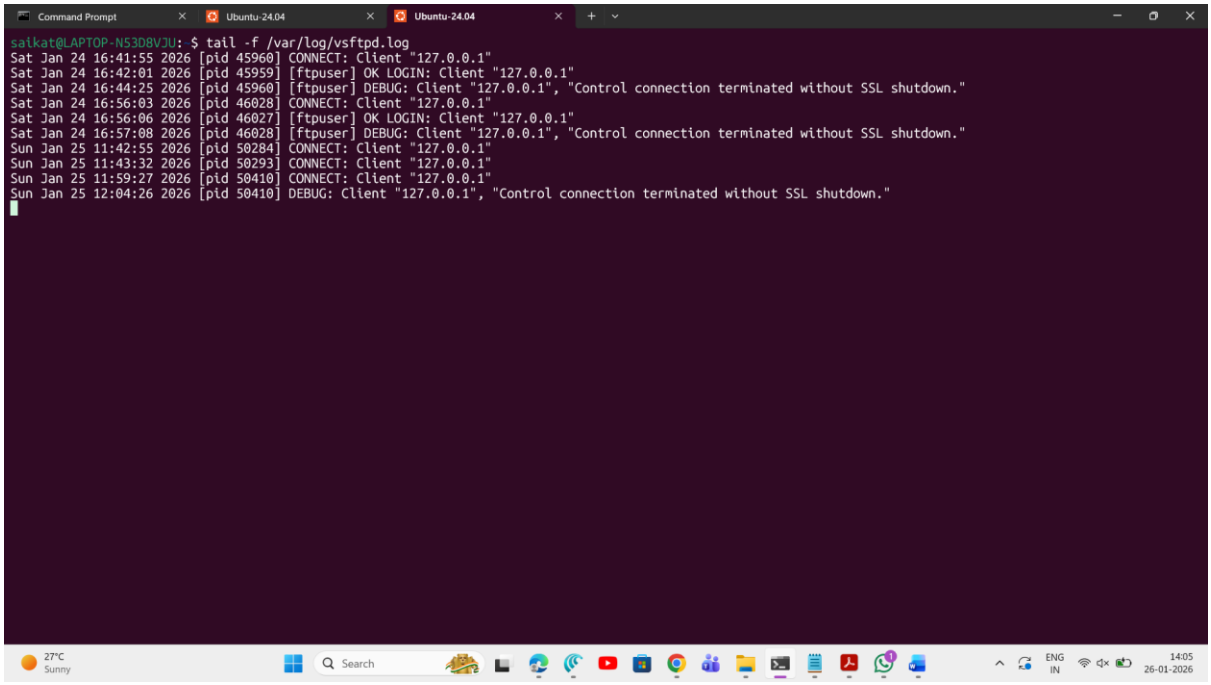
12. Restart vsftpd

Apply all changes:

```
sudo systemctl restart vsftpd
```

Check logs if something goes wrong:

```
tail -f /var/log/vsftpd.log
```



---

13. Testing the FTP Server

13.1 Using an FTP Client (Recommended)

Use FileZilla.

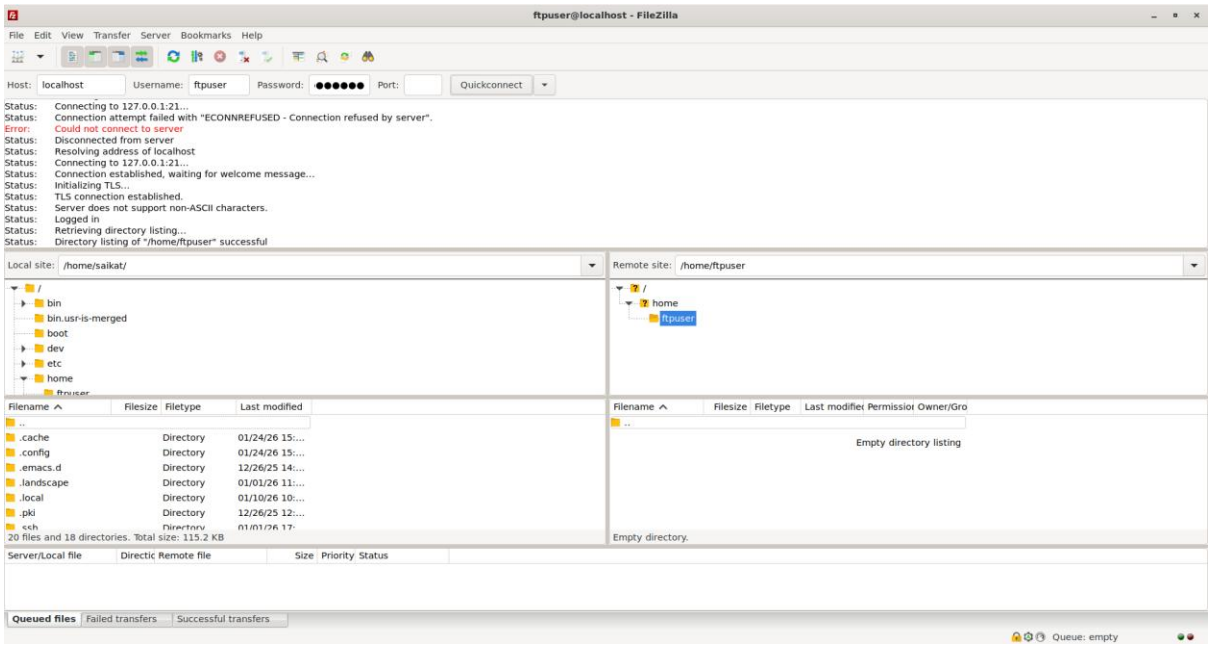
Install FileZilla with the following commands:

```
sudo apt update
```

```
sudo apt install filezilla -y
```

- Protocol: FTP
- Encryption: Require explicit FTP over TLS
- Host: server IP / hostname
- Username: ftpuser
- Password: (your password)
- Port: 21

Accept the certificate warning (expected for self-signed certs).

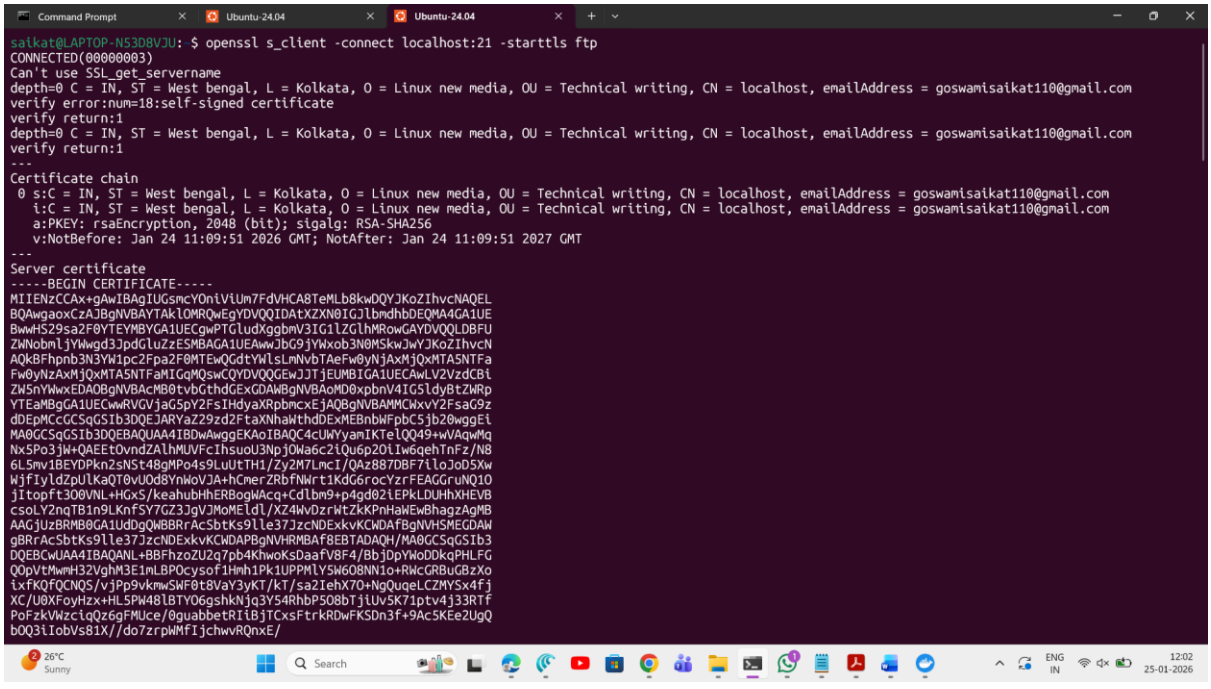


13.2 Testing from the Command Line

```
ftp localhost
```

For FTPS testing:

```
openssl s_client -connect localhost:21 -starttls ftp
```



14. Common Problems and Fixes

Login Fails

- Check `/etc/vsftpd.conf`
- Verify user exists
- Check `/var/log/vsftpd.log`

Passive Mode Not Working

- Firewall ports not open
- Wrong port range
- NAT not forwarding ports

Permission Denied on Upload

```
chown ftpuser:ftpuser /home/ftpuser
chmod 755 /home/ftpuser
```

---

15. Best Practices for Production

- Use **Let's Encrypt** instead of self-signed certs
- Disable FTP entirely if SFTP is enough
- Use separate users per application
- Monitor logs regularly
- Consider fail2ban for brute-force protection

---

16. FTP vs FTPS vs SFTP (Quick Comparison)

Protocol	Encryption	Port	Notes
FTP	No	21	Insecure
FTPS	TLS	21	Secure FTP
SFTP	SSH	22	Preferred modern choice

---

17. Final Thoughts

vsftpd lives up to its name: **very secure**, when configured correctly.

If you need compatibility with legacy systems or shared environments, FTPS with vsftpd is still a solid choice. For modern systems, consider **SFTP**, but it's always valuable to understand FTP deeply.

If you want, I can:

- Convert this setup to **SFTP-only**

- Add **virtual users**
- Harden it further for **cloud deployments (AWS/GCP/Azure)**