Jellyfin From Scratch: How to Build a Complete Media Server on Linux Mint (Docker, Nginx, and Let’s Encrypt)

Start from a clean OS and end with your own secure, self-hosted streaming platform — complete with HTTPS access, remote streaming, and full media organization.

Questions unlock answers.

□□ Introduction

Jellyfin is a free, open-source media server that lets you organize and stream your movies, shows, and music to any device. Think of it as your personal Netflix — except you control everything.

In this guide, you’ll start from a clean **Linux Mint** installation and finish with a **fully working Jellyfin server** running inside **Docker**, served securely through **Nginx** with **Let’s Encrypt SSL certificates**, protected by **Fail2ban**, and accessible anywhere through a **DuckDNS domain**.

This tutorial uses **CPU-only transcoding** (no GPU acceleration) and follows current **Docker Compose v2** standards. This means, if want to use your GPU to do all the heave lifting, I ain't gunna show you that here but, you can still use this tutorial to get your server up and running then, configure your GPU later.

□□ Prerequisites

- A clean install of **Linux Mint** (based on Ubuntu 20.04 or later)
- Sudo privileges
- Install SSH (if you want to connect remotely)
- Vim - Text editor (or which ever you prefer)
- An active internet connection
- A domain name - I am using a free one from DuckDNS.org (e.g., `myserver.duckdns.org`)
- Some media files stored locally or on a mounted drive. I do NOT go over how to mount a drive!
- Router: port forwarding
  - Ports 22, 80 and 443

Doing some of the Prereqs:

- Install your OS. I ain't helping with that part.
- If you even want to have remote access, Check this one out to do some sudo (super user) and some SSH (remote login) stuff.
- **Install VIM text editor:**

```
sudo apt install vim
```

- **You will need a domain name** for this. You can use your own custom one but, if you don't one, get one. duckdns.org is a good free source for that. Ill be using mint1.duckdns.org for this example.
- **Where is your media files right now?** If you haven't already, to but them where you are going to be using them at. /media is a good spot or, if you are mounting a drive, /mnt is also a good spot. i'll have a mounted RAID drive/s and will be mounting it in /mnt/jellyfin/media
  I ain't helppen you with this neither so... if need be, now is a good time to pause and come back once you know where your stuff will be stored.

  If you need help with a RAID, *Set up a RAID here* - Not hardware RAID. It is a Digital RAID.
- **Your Router:**
  (optional) If you want to have remote access by SSH from other computer, you will need to open port 22 for this. You can use the link above or go here if you don't know how.
  Either way, you will need to open up ports 80 and 443. This will expose your computer (and network) to the world, through those ports. But, that's OK cause, we want it to for now.

Once you are able to complete those few tasks, we can then continue with getting a Jellyfin server up and running.

□□ Step 1 — System Preparation

Start by updating your system and installing a few required packages.

```
sudo apt update && sudo apt upgrade -y
```

If this takes less than 10 seconds... read the error and fix it. If not...

...give it a minute to do its thang.

```
sudo apt install -y ca-certificates curl gnupg lsb-release ufw fail2ban
```

.....This might take another quick minute. If you have any prompts, say "yes" or "y".

Enable the firewall and allow only essential services: (run each line separately)

```
sudo ufw default deny incoming
sudo ufw default allow outgoing
sudo ufw allow OpenSSH #or "SSH" or "22"
sudo ufw allow 80/tcp
sudo ufw allow 443/tcp
sudo ufw enable
```

Check status:

```
sudo ufw status
```

You should see **SSH**, **HTTP**, and **HTTPS** allowed. Did it work?

□□ Step 2 — Install Docker and Docker Compose

Install Docker Engine and the Compose plugin using official repositories: These are to separate programs that work in tandem. "Docker" and "Docker Compose"

```
sudo install -m 0755 -d /etc/apt/keyrings
```

```
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o
/etc/apt/keyrings/docker.gpg
```

```
echo \
"deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/docker.gpg] \
https://download.docker.com/linux/ubuntu noble stable" | \
sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
```

```
sudo apt update
```

```
sudo apt install -y docker-ce docker-ce-cli containerd.io docker-compose-plugin
```

Enable and start Docker:

```
sudo systemctl enable docker
```

```
sudo systemctl start docker
```

Verify installation:

```
docker --version
```

```
docker compose version
```

They should both tell you what version of Docker & Docker Compose you have.

Be in the directory where your .yml file is located. (step 4)

```
docker compose up -d
```

Check containers:

```
docker ps
```

Can you see it running?

Visit your server: Your IP address plus the port number might look something like 192.168.2.196:8000

enter that into your web browser and see if Jellyfin comes up.

□□ Step 3 — Create Directory Structure

Where do you want your Jellyfin to live? For this, we'll keep Jellyfin and related configs in `/opt/jellyfin`.

```
sudo chown -R $USER:$USER /opt/jellyfin
```

Use which ever user you need it to be.

```
sudo mkdir -p /opt/jellyfin/{config,cache,media,certbot}
```

Where is your media stored at?
It needs to be in your local drive and ready to use.
If it's not being stored in your local computer, mount it.

Mount your media drive to `/opt/jellyfin/media`. Or, if you already have a local address, you can put it there.
We are not going over how to mount a drive in this tutorial so, if you don't know how to do that part, go learn now, and come back. This example may not make sense to you.
Example:

```
sudo mkdir /mnt/media
sudo mount /dev/sdX1 /mnt/media
sudo ln -s /mnt/media /opt/jellyfin/media
```

Or, you could mount directly to the directory:

```
sudo mount /dev/sdX1 /opt/jellyfin/media
```

You need this permanent so, go do that in fstab.

⚙ Step 4 — Create the Docker Compose File

Create the file:
This is a .yml file. Don't know the language? That's OK. I gotchew.

```
sudo vim /opt/jellyfin/docker-compose.yml
```

Paste this configuration:
above, we made a bunch of directories. config, cache, media, nginx, certbot
Make sure that the directories in here match the ones you made.

Press "i" to edit the text. Paste this in:

```
services:
  jellyfin:
    image: jellyfin/jellyfin:latest
    container_name: jellyfin
    environment:
      - PUID=1001
      - PGID=1001
      - TZ=America/Los_Angeles
    volumes:
      - /home/user/jellyfin/config:/config
      - /home/user/jellyfin/cache:/cache
      - /home/user/jellyfin/media:/media
    ports:
      - 8098:8096
    restart: unless-stopped
```

Save and exit
`Esc`
`:`
`wq`
`enter`

- Find your PUID:

  ```
  tail /etc/passwd
  ```

  Use that ID in the below.

- To check your timezone options, enter this:

  ```
  timedatectl list-timezones
  ```

  then use the same schema as the above in the yaml file.

□□ Step 5 — Configure Nginx Reverse Proxy

Install Nginx.

```
sudo apt install nginx -y
```

Now locate the config file so we can make a virtual reverse proxy.
Will probably be at /etc/nginx/sites-available/
Edit the file "default"
Paste the following:

```
server {
        server_name Your.customesite.com;
            location / {
                proxy_set_header Host              $host;
                proxy_set_header X-Real-IP         $remote_addr;
                proxy_set_header X-Forwarded-For   $proxy_add_x_forwarded_for;
                proxy_set_header X-Forwarded-Proto $scheme;
                proxy_http_version 1.1;
                proxy_set_header   Upgrade    $http_upgrade;
                proxy_set_header   Connection "upgrade";
                proxy_pass http://localhost:8096; #add your own port number if its not this
        }

    #place holder for certpot later. It will maintain your SSL cert for you.
    # you can delete this line and the one above it once you get the certbot running.
```

```
}
```

□□ Replace:

- `Your.customesite.com` with your actual DuckDNS (or other custom) domain.
- `localhost:8096` with your current port, if you have changed it.

Save and exit
`Esc`
`:`
`wq`
`enter`

Start/Test it

```
sudo systemctl start nginx
```

```
sudo systemctl enable nginx
```

Go to your domain.

□□ Step 6 — Get Your SSL Certificate with your domain and Certbot

First, make sure your **DuckDNS domain** is pointed to your public IP. If you haven't already, go to your domain dashboard and make sure.

Go here to see how to get a an automated SSL cert.

Inside that page, go through the "Certbot : Auto SSL certifications." section. Then come back to continue.

Visit your server:

`https://yourdomain.duckdns.org` Or whatever your domain name is. Go see it!

You should see the Jellyfin setup wizard

Revision #52
Created 2025-10-30 05:06:19 UTC by Danicus
Updated 2025-11-11 07:46:25 UTC by Danicus