

# HOW TO INSTALL A FULL ARR-STACK ON SYNOLOGY NAS RUNNING DSM 7.2

Mathias Furenes

# Table of Contents

<b>BEFORE WE BEGIN.....</b>	<b>4</b>
INFO ABOUT TL;DR .....	4
LEGAL DISCLAIMER .....	4
INTRODUCTION .....	4
WHAT ARE THE *ARRS?.....	5
<i>From the wiki itself:</i> .....	5
<i>In Simple terms:</i> .....	5
Prowlarr: .....	5
Radarr.....	5
Sonarr.....	5
Lidarr .....	5
Readarr.....	5
Whisparr .....	6
Bazarr .....	6
Flaresolverr .....	6
Overseerr .....	6
Requestrr.....	6
Arr-stack: .....	6
ALL THE DIFFERENT *ARR APPS (AND RELATED ONES THAT I KNOW OF) .....	7
THE APPS I WILL COVER IN THIS GUIDE .....	9
HARDWARE SPECS .....	10
<i>Minimum requirements</i> .....	10
CPU .....	10
RAM .....	10
Storage .....	10
<i>Recommended requirements</i> .....	11
CPU .....	11
RAM .....	11
Storage .....	11
RECOMMENDED NAS MODELS FOR THIS SETUP .....	12
<i>Minimum</i> .....	12
<i>Recommended</i> .....	12
<i>Models to avoid</i> .....	13
Low-End ARM-Based NAS Models.....	13
Why not:.....	13
Models with Less Than 2 GB RAM.....	13
Why not:.....	13
Older/Legacy Models .....	14
Why not:.....	14
Single-Bay NAS Models .....	14
Why not:.....	14
Models without integrated graphics .....	14
Why not:.....	14
What you need instead .....	15
TL;DR .....	16
<b>PREPARATIONS .....</b>	<b>22</b>
FOLDER STRUCTURE.....	22
<i>The folder structure should look like this:</i> .....	23

Media.....	23
Docker .....	24
SETTING PERMISSIONS .....	25
<b>THE INSTALLATION .....</b>	<b>26</b>
SETTING UP THE START UP SCRIPT FOR GLUETUN.....	26
FIREWALL RULES (IF YOU HAVE FIREWALL SET UP).....	29
WIREGUARD KERNEL MODULE .....	32
CREATING A SYNOLOGY BRIDGE NETWORK .....	34
DOCKER PROJECT .....	35
<i>Required</i> .....	35
<i>GlueTUN:</i> .....	36
Ports .....	37
Volumes.....	37
Environment.....	37
VPN Service provider .....	37
Finding WireGuard details for Mullvad.....	38
Proxy .....	39
Timezone.....	39
Firewall Outbound Subnets .....	39
Network_mode.....	40
security_opt: .....	40
Restart: always.....	40
Full GlueTUN docker-compose.yml.....	41
<i>Download client</i> .....	43
network_mode .....	43
Environment.....	43
PUID and PGID .....	43
UMASK.....	44
Volumes.....	44
Full docker-compose.yml for qbittorrent .....	45
<i>Sonarr</i> .....	46
Something important:.....	46
Full docker-compose.yml file .....	47
<i>Radarr</i> .....	48
<i>Lidarr</i> .....	49
<i>Prowlarr</i> .....	50
<i>Flaresolverr</i> .....	51
<i>Overseerr</i> .....	52
<i>Requestrr</i> .....	53
<i>Tautulli</i> .....	54
<i>Putting it all together</i> .....	55
<i>Common Errors</i> .....	60
<b>CONFIGURATION OF THE APPS .....</b>	<b>62</b>
QBITTORRENT.....	62
<i>Login</i> .....	62
<i>Change username and password</i> .....	63
<i>Change downloads path</i> .....	64
RADARR .....	65
<i>Adding Authentication method</i> .....	65
<i>Adding Root Folder</i> .....	66

<i>Changing Movie Naming Scheme</i> .....	67
<i>Quality Settings (File Size)</i> .....	69
<i>Quality profiles</i> .....	70
SONARR.....	71
<i>Adding Root Folder(s)</i> .....	71
<i>Changing Naming Scheme(s)</i> .....	73
<i>Quality Settings (File Size)</i> .....	74
<i>Quality Profiles</i> .....	75
<i>Quality Profiles (Anime)</i> .....	76
LIDARR .....	77
<i>Adding Root Folder(s)</i> .....	77
<i>Quality Settings</i> .....	79
CONNECTING DOWNLOAD CLIENT TO RADARR, SONARR AND LIDARR .....	80
PROWLARR .....	83
<i>Adding the apps to Prowlarr</i> .....	83
<i>How do I get the API key?</i> .....	86
<i>Connecting Flaresolverr</i> .....	87
<i>Adding indexers</i> .....	89
OVERSEERR .....	92
<i>Configuring Overseerr</i> .....	92
<i>Adding Radarr and Sonarr</i> .....	94
Radarr.....	95
Sonarr.....	97
<i>Requestr</i> .....	100
Configuring Requestr .....	100
How to get Discord Application Id and Bot Token .....	100
How to invite the bot to our Discord server .....	105
Connecting Requestr to Overseerr (Or Radarr, Sonarr, Ombi) .....	106
<b>SOURCES.....</b>	<b>109</b>

# Before we begin

## Info about TL;DR

The [TL;DR](#) comes before the main guide. For the best experience, skip this section and go to the [full guide](#). The [TL;DR](#) only covers the installation, and NOT the [configs](#) as they are too complex to put in a TL;DR. It also requires some knowledge on docker beforehand so you can edit the docker-compose to match your system.

## Legal Disclaimer

For legal reasons I must state that I do not condone illegal pirating of copyrighted material. This is made for educational purposes only. I expect that everyone who follows this guide will only use this for legal purposes. Please never ever use this to illegally download copyrighted material such as, but not limited to, movies and TV Shows.

## Introduction

Before we start let's figure out your needs. Do you want to download movies, tv shows, music, e-books, comics or adult videos? Most likely you want a combination of a lot of them. We also need to figure out whether you want it to be connected to a VPN. Here is a break-down of all the apps and their use-cases:

## What are the \*arrs?

From the wiki itself:

"Lidarr, Prowlarr, Radarr, Readarr, Sonarr, and Whisparr are collectively referred to as "\*arr" or "\*arrs". They are designed to automatically grab, sort, organize, and monitor your Music, Movie, E-Book, or TV Show collections for Lidarr, Radarr, Readarr, Sonarr, and Whisparr; and to manage your indexers and keep them in sync with the aforementioned apps for Prowlarr."

In Simple terms:

### *Prowlarr:*

An index manager. This just means it searches for files to download on websites you assign it.

### *Radarr:*

When Prowlarr finds a movie file, it gives it to Radarr. Radarr then sends it over to a download client, like QBitTorrent. After it's done downloading, Radarr takes it away from QBitTorrent again and rename the file appropriately before it puts it inside your Media library.

### *Sonarr:*

Same as Radarr, but for TV shows.

### *Lidarr:*

Same as Radarr, but for music

### *Readarr:*

Same as Radarr, but for e-books

*Whisparr:*

Same as Radarr, but for adult videos

*Bazarr:*

Connects with Radarr and Sonarr to download subtitles for your movies and TV shows.

*Flaresolverr:*

Bypasses cloudfare solver

*Overseerr:*

A requesting application where you browse or search for movies and TV shows, kinda like Netflix, and with a click of a button they start downloading to your own media collection!

*Requestr:*

Goes together with Overseerr or Radarr and Sonarr to allow for requesting through discord chat.

*Arr-stack:*

The arr-stack just refers to a collection of these applications bundled and working together.

## All the different \*arr apps (and related ones that I know of)

**Prowlarr** – Index manager. It searches the torrent sites for downloads (Recommended for everyone)

**Flaresolverr** – Some indexers require you to solve a Cloudflare captcha. Flaresolverr can do this.

**Jackett** – Alternative to Prowlarr. Most find Prowlarr to be both easier to set up and better to use.

**Sonarr** – TV Shows/Anime Shows downloader.

**Radarr** – Movie downloader.

**Lidarr** – Music downloader.

**Readarr** – E-book downloader

**Mylar** – Comics downloader.

**Bazarr** – Subtitle downloader for your Movies and TV Shows.

**Lazy Librarian** – A program to follow authors and grab metadata for digital reading.

**Whisparr** – Adult videos downloader.

**GlueTUN** – Required for use of VPN

**Plex** – A frontend to your media server. It's where you access all your media, in the style of something like Netflix. It also has a very good Spotify-like app called plexamp.

**Jellyfin** – Also a frontend to your media server. Jellyfin is, in contrast to Plex, open-source. This means that all of its features are and always will be free, but it also means that it doesn't have the same funding and therefore might not have all of the features Plex has.

**Overseerr** – Allows for easy requesting of movies and TV shows to add to Plex.

**Jellyseerr** – Same as Overseer but for Jellyfin.

**Ombi** – App to request Movies and TV Shows for plex or Emby.

**Requestrr** – Allows for requests for Sonarr and Radarr via chat, like Discord. It can also be integrated with Ombi and Overseer.

**qBitTorrent** – Torrent download client.

**NZBGet** – NZB download client.

**Tautulli** – Plex media server statistics

There are even more, but I have not gotten into these myself. These are the ones I have atleast some knowledge about.

## The apps I will cover in this guide

I would like to cover as many as possible, but I have not used or tried some of them myself. I host on Plex, so I use Overseer, and have not tried neither Jellyseer or Ombi. Even though I'm pretty sure Jellyseer is the exact same just for Jellyfin. But you should always read the official docs yourself. Anyway, the apps I will go over is:

- GlueTUN
- Prowlarr
- Flaresolverr
- Sonarr
- Radarr
- Lidarr
- Overseer
- Requestrr
- qBitTorrent
- Tautulli

## Hardware specs

I am running all this on a DS423+, with an extra 16GB memory stick and 512GB SSD cache. I have not tested it myself on any other devices, but I have made a few google searches and asked ChatGPT for some help to determine the systems requirements. Therefore, I ask you to take these number with a grain of salt.

## Minimum requirements

### *CPU*

A quad-core 64bit CPU with x86 architecture. (Docker can only run on x86, and not any ARM CPUs natively. You might be able to still try this out, but you will have to do some workarounds.) An Intel Celeron J4105 or Intel Celeron J4125 should be sufficient for basic use.

### *RAM*

4GB RAM (reported by ChatGPT). I think it might be able to run on 2GB for low, basic use. But don't expect the best performance

### *Storage*

These apps don't take up more than 2GB-5GB for database and configs.

A 2-hour movie in 480p will take 700MB-2GB space, and 3GB-6GB in 720p. A 12-episode TV show with 45 minutes long episodes will take 4GB-10GB in 480p and 12GB-27GB in 720p

## Recommended requirements

### *CPU*

Intel Celeron J4125 or higher. For best performance, Intel Core i3/i5 or AMD equivalent. But since most will probably run this on a Synology, the J4125 or better is sufficient.

### *RAM*

8GB RAM. For best performance, an upgrade to 16GB will make it a lot smoother.

### *Storage*

A 2-hour long movie in 1080p will take 8GB-15GB storage space, and in 4k it will be about 20GB-50GB. A 12-episode show with 45 minutes long episodes will take up 35GB-60GB in 1080p and 90GB-225GB in 4k.

## Recommended NAS models for this setup

Since I assume most people will use a Synology NAS, as this is what this guide was meant for, I will list some recommendations.

### Minimum

- **Synology DS220+**
  - CPU: Intel Celeron J4025 (dual-core)
  - RAM: 2GB (expandable to 6GB)
  - Suitable for running a few applications simultaneously with small to medium libraries.
- **Synology DS720+**
  - CPU: Intel Celeron J4125 (quad-core)
  - RAM: 2GB (expandable to 6GB)
  - Great for small to mid-size media libraries, running Docker containers, and handling multiple tasks at once.

### Recommended

- **Synology DS920+**
  - CPU: Intel Celeron J4125 (quad-core)
  - RAM: 4GB (expandable to 8GB)
  - Supports SSD caching, making it a solid choice for heavier workloads like streaming, transcoding, and multiple apps running concurrently.
- **Synology DS423+**
  - CPU: Intel Celeron J4125 (quad-core)
  - 2GB RAM (Recommended to upgrade to at least 4GB or 6GB)
  - 4-bay NAS with support for 2 NVME drives
- **Synology DS1821+**
  - CPU: AMD Ryzen V1500B (quad-core)
  - RAM: 4GB (expandable to 32GB)
  - 8-bay NAS with strong processing power for large libraries and heavy multitasking.

## Models to avoid

### *Low-End ARM-Based NAS Models*

- **Synology DS216j** (Marvell Armada 385, dual-core 1.0GHz, 512MB)
- **Synology DS218j** (Marvell Armada 385, Dual-Core 1.3 GHz, 512 MB RAM)
- **Synology DS219j** (Marvell Armada 3720, Dual-core 800 MHz, 256 MB RAM)
- **Synology DS220j** (Realtek RTD1296, Quad-Core 1.4 GHz, 512 MB RAM)

### Why not:

- **CPU:** These models come with weak, low-power ARM processors that are not suited for running multiple Docker containers or handling tasks like torrenting and media management.
- **RAM:** Most of these devices have **512 MB RAM or less**, which is far too little for running multiple services in Docker.
- **Docker Support:** ARM-based models may not fully support Docker, especially for complex workloads, and will struggle with performance under even light to moderate use.

### *Models with Less Than 2 GB RAM*

- **Synology DS220j** (512 MB RAM)
- **Synology DS218play** (1 GB RAM)
- **Synology DS218** (2 GB RAM)
- **Synology DS118** (1 GB RAM)

### Why not:

- **Insufficient RAM:** Apps like **Overseerr** and **qBittorrent** require more memory, and these models would quickly run out of resources. With less than 2 GB, you'll experience poor performance, constant swapping to disk, or the inability to run all your containers simultaneously.
- **No Upgrade Path:** Many of these models do not allow you to upgrade the RAM, so you're stuck with what they offer.

### *Older/Legacy Models*

- **Synology DS214+** (Dual-core 1.33 GHz, 1 GB RAM)
- **Synology DS415play** (Intel Atom CE5335, Dual-core 1.6 GHz, 1 GB RAM)
- **Synology DS216play** (ARM Cortex-A9, Dual-core 1.5 GHz, 1 GB RAM)

### *Why not:*

- **Outdated CPU architecture:** These older CPUs lack the power and modern architecture needed for virtualization and handling Docker workloads.
- **RAM limitations:** Even if some of these have x86 architecture, 1 GB or even 2 GB of RAM is not sufficient for your use case.

### *Single-Bay NAS Models*

- **Synology DS118** (Realtek RTD1296, Quad-core 1.4 GHz, 1 GB RAM)
- **Synology DS119j** (Marvell Armada 3700, Dual-core 800 MHz, 256 MB RAM)

### *Why not:*

- **Low performance:** These single-bay models come with very basic hardware, meaning you'll struggle to run Docker and multiple applications.
- **No redundancy:** With only one drive, there's no data redundancy (no RAID), which is a concern when managing large amounts of media files.

### *Models without integrated graphics*

- Synology DS923+ (AMD Ryzen R1600, Quad-core 2.6GHz, 4GB RAM)
- Synology DS 1522+ (AMD Ryzen R1600, Quad-core 2.6GHz, 8GB RAM)

### *Why not:*

- **No integrated graphics:** Although these can work just fine, they are not ideal as you won't be able to do hardware transcoding. For native playback of h264 files these can be perfect.

### *What you need instead*

To run this full arr-stack smoothly on Docker, aim for:

- **x86-64 architecture** (with embedded graphics)
- **Minimum 4 GB RAM**, ideally **8 GB or more**.
- **Expandable RAM** for future growth.
- **At least dual-bay** for RAID redundancy.

# TL;DR

If you don't want to read all that, just do this:

1. Create these folder inside /volume1/docker/arr-stack:
  - gluetun
  - lidarr > config
  - overseer > config
  - prowlarr > config
  - qbittorrent > config, downloads
  - radarr > config
  - requestrr > config
  - sonar > config
2. SSH into your NAS and type these commands

```
- sudo chmod -R 777 /volume1/Media  
- sudo chown -R < UID>:< GID> \volume1\Media  
- id
```

Take note of GID and UID output from id command.

3. Go to Task Scheduler and create a trigger task on start-up to run this script:

```
4. #!/bin/sh -e  
5.  
6. insmod /lib/modules/tun.ko
```

1. Create a firewall rule to allow port 1194 and 1195
2. Go to think link: <https://www.blackvoid.club/wireguard-spk-for-your-synology-nas/>. Find the correct version for your system and manually install the .spk file in package cener. Don't run when finished. Reboot afterwards
3. SSH into your device and type this command:  
4. /var/packages/WireGuard/scripts/start
5. Find out how to configure GlueTUN for your VPN provider:  
<https://github.com/qdm12/gluetun-wiki/tree/main/setup/providers>
6. Create a new network inside container manager:  
Configure the network like this:

- Subnet: 172.20.0.0/16
- IP range: 172.20.0.2/25
- Gateway: 172.20.0.1
- IPv6: Disabled
- IP Masquerade: enabled (Leave the “disable” option unticked)

7. Create a new docker project. Path should be /volume1/docker/arr-stack. Paste this docker-compose.yml and make necessary changes:

```

8. version: "3"
9. services:
10.   gluetun:
11.     image: qmcgaw/gluetun
12.     container_name: gluetun
13.     hostname: gluetun
14.     cap_add:
15.       - NET_ADMIN
16.     devices:
17.       - /dev/net/tun:/dev/net/tun
18.     ports:
19.       - 6881:6881
20.       - 6881:6881/udp
21.       - 8085:8085 # qbittorrent
22.       - 8989:8989 # Sonarr
23.       - 9696:9696 # Prowlarr
24.       - 7878:7878 # Radarr
25.       - 8686:8686 #Lidarr
26.       - 8191:8191 #FlareSolverr
27.       - 5055:5055 #Overseerr
28.       - 4545:4545 #Requestrr
29.
30.     volumes:
31.       - \volume1\docker\arr-stack\gluetun:/gluetun
32.     environment:
33.       - VPN_SERVICE_PROVIDER=mullvad
34.       - VPN_TYPE=wireguard
35.       - VPN_DISABLE_IPV6=true
36.     # OpenVPN:
37.     # - OPENVPN_USER=
38.     # - OPENVPN_PASSWORD=
39.     # Wireguard:
40.     - WIREGUARD_PRIVATE_KEY= iExxD5V5kkXnh+40dyo/PmCL1aus8eNBdHQMWergYFWo=
41.     - WIREGUARD_ADDRESSES=10.72.171.113/32
42.     - DNS=10.64.0.

```

```

43.      - SERVER_HOSTNAMES=se-sto-wg-001,se-sto-wg-002,se-sto-wg-003,se-sto-
44.      - SERVER_CITIES=stockholm
45.      - HTTPPROXY=off #change to on if you wish to enable
46.      - SHADOWSOCKS=off #change to on if you wish to enable
47.      # Timezone for accurate log times
48.      - TZ=Europe/Oslo
49.      # Server list updater
50.      # See https://github.com/qdm12/gluetun-
51.      - UPDATER_PERIOD=24h
52.      - FIREWALL_OUTBOUND_SUBNETS=172.20.0.0/192.168.0.0/24 #change this in
53.      line with your subnet see note on guide
54.      - FIREWALL_VPN_INPUT_PORTS=12345 #uncomment or remove this line
55.      based on the notes below
56.      network_mode: synobridge
57.      labels:
58.      - com.centurylinklabs.watchtower.enable=false
59.      security_opt:
60.      qbittorrent:
61.      image: lscr.io/linuxserver/qbittorrent
62.      container_name: qbittorrent
63.      network_mode: "service:gluetun"
64.      environment:
65.      - PUID=1026
66.      - PGID=100
67.      - TZ=Europe/Oslo
68.      - WEBUI_PORT=8085
69.      - UMASK=022
70.      volumes:
71.      - /volume1/docker/arr-stack/qbittorrent/config:/config
72.      - /volume1/Media/Torrents:/Media/Torrents
73.      depends_on:
74.      gluetun:
75.      condition: service_healthy
76.      security_opt:
77.      - no-new-privileges:true
78.      restart: always
79.      sonarr:
80.      image: lscr.io/linuxserver/sonarr:latest
81.      container_name: sonarr
82.      network_mode: "service:gluetun"
83.      environment:
84.      - PUID=1026

```

```

85.      - PGID=100
86.      - TZ=Europe/Oslo
87.  volumes:
88.    - /volume1/docker/arr-stack/sonarr/config:/config
89.    - /volume1/Media:/Media
90.  depends_on:
91.    gluetun:
92.      condition: service_healthy
93.  restart: unless-stopped
94.
95.  prowlarr:
96.    image: lscr.io/linuxserver/prowlarr:latest
97.    container_name: prowlarr
98.    network_mode: "service:gluetun"
99.    environment:
100.      - PUID=1026
101.      - PGID=100
102.      - TZ=Europe/Oslo
103.    volumes:
104.      - /volume1/docker/arr-stack/prowlarr/config:/config
105.    depends_on:
106.      gluetun:
107.        condition: service_healthy
108.      restart: unless-stopped
109.
110.  radarr:
111.    image: lscr.io/linuxserver/radarr:latest
112.    container_name: radarr
113.    network_mode: "service:gluetun"
114.    environment:
115.      - PUID=1026
116.      - PGID=100
117.      - TZ=Europe/Oslo
118.    volumes:
119.      - /volume1/docker/arr-stack/radarr/config:/config
120.      - /volume1/Media:/Media
121.    depends_on:
122.      gluetun:
123.        condition: service_healthy
124.      restart: unless-stopped
125.
126.  lidarr:
127.    image: lscr.io/linuxserver/lidarr:latest
128.    container_name: lidarr
129.    network_mode: "service:gluetun"
130.    environment:

```

```

131.      - PUID=1026
132.      - PGID=100
133.      - TZ=Europe/Oslo
134.  volumes:
135.    - /volume1/docker/arr-stack/lidarr/config:/config
136.    - /volume1/Media:/Media
137.  depends_on:
138.    gluetun:
139.      condition: service_healthy
140.    restart: unless-stopped
141.
142.  flaresolverr:
143.    image: ghcr.io/flaresolverr/flaresolverr:latest
144.    container_name: flaresolverr
145.    network_mode: "service:gluetun"
146.    environment:
147.      - TZ=Europe/Oslo
148.    depends_on:
149.      gluetun:
150.        condition: service_healthy
151.    security_opt:
152.      - no-new-privileges:true
153.    restart: unless-stopped
154.
155.  overseerr:
156.    image: sctx/overseerr:latest
157.    container_name: overseerr
158.    network_mode: "service:gluetun"
159.    environment:
160.      - LOG_LEVEL=debug
161.      - TZ=Europe/Oslo
162.    volumes:
163.      - /volume1/docker/arr-stack/overseer/config:/app/config
164.    depends_on:
165.      gluetun:
166.        condition: service_healthy
167.    restart: unless-stopped
168.
169.  requestrr:
170.    image: darkalfx/requestrr
171.    container_name: requestrr
172.    network_mode: "service:gluetun"
173.    volumes:
174.      - /volume1/docker/arr-stack/requestrr/config:/root/config
175.    depends_on:
176.      gluetun:

```

```
177.      condition: service_healthy  
178.      restart: unless-stopped
```

11. For config of apps, I can't really put it in a TL;DR. Chek out the [full guide](#), or find out yourself on the official docs.

# Preparations

## Folder Structure

I assume you have installed container manager (docker) on your Synology system already, but if not do that now. Then we need to go into our “docker” shared folder and make some new folder.

Inside the “docker” shared folder, create a folder and call it something like “arr-stack”

Inside “arr-stack”, create these folders:

- gluetun
- lidarr
- overseer
- prowlarr
- qbittorrent
- radarr
- requestrr
- sonar
- tautulli
- nzbget (Optional if you want to use usenet. NOTE: It costs money to join a usenet site)

If you have any other apps beside these, create a folder for them too as described in the official docs.

Create a folder called “config” inside lidarr, overseer, prowlarr, qbittorrent, radar, requestrr, sonarr and tautulli.

Create a new shared folder for housing your media. You can call this Data or Media. Within this shared library, you need another folder where you put the actual media. You can call this Media. In “/volume1/Media/Media” you create a folder for movies, tv shows, anime, music and whatever else you would like. Go back to “/volume1/Media” and create a folder called “torrents.” This is where we download our torrent files with

qBitTorrent. Inside it create “Incomplete” and “Complete”. If you want to use Usenet, you can create folder called “usenet” with “Complete” and “Incomplete” inside it.

The folder structure should look like this:

*Media*

```
|── Torrents
|   ├── Incomplete
|   └── Complete
|
|── Usenet
|   ├── Incomplete
|   └── Complete
|
|   ├── Movies
|   ├── Music
|   ├── Anime TV Shows
|   └── TV Shows
|
└── Media
    ├── Movies
    ├── Music
    ├── Anime TV Shows
    └── TV Shows
```

## *Docker*

```
└── arr-stack
    ├── bazarr
    │   └── config
    ├── gluetun
    ├── lidarr
    │   └── config
    ├── nzbget (Optional)
    │   └── config
    ├── overseer
    │   └── config
    ├── prowlarr
    │   └── config
    ├── qbittorrent
    │   └── config
    ├── radarr
    │   └── config
    ├── requestrr
    │   └── config
    ├── sonarr
    │   └── config
    └── tautulli
        └── config
```

## Setting permissions

To make sure that all the apps have the right permissions to read and write the relevant folder, we need to ssh into the NAS. I prefer PuTTY, but you can use powershell or anything else that you like.

1. SSH into your NAS
2. Type the following 2 commands:
  - `sudo chmod -R 777 /volume1/Media/Media`
  - `sudo chmod -R 777 /volume1/Media/Torrents`
3. Find your UID and GID
  - Type: `id`
4. You should get an output like this:

```
uid=1026(Mathias) gid=100(users)
```

As you can see, we get an output with uid=XXXX and gid=YYY.

5. Type the following 2 commands:
  - `sudo chown -R <UID>:<GID> /volume1/Media/Media`
  - `sudo chown -R <UID>:<GID> /volume1/Media/Torrents`

**NOTE:** If you have your Media library at another location than “/volume1/Media”, then put your path in the 2 relevant commands.

These commands ensure that we have the correct permissions to read and write the files of the directories and their subdirectories.

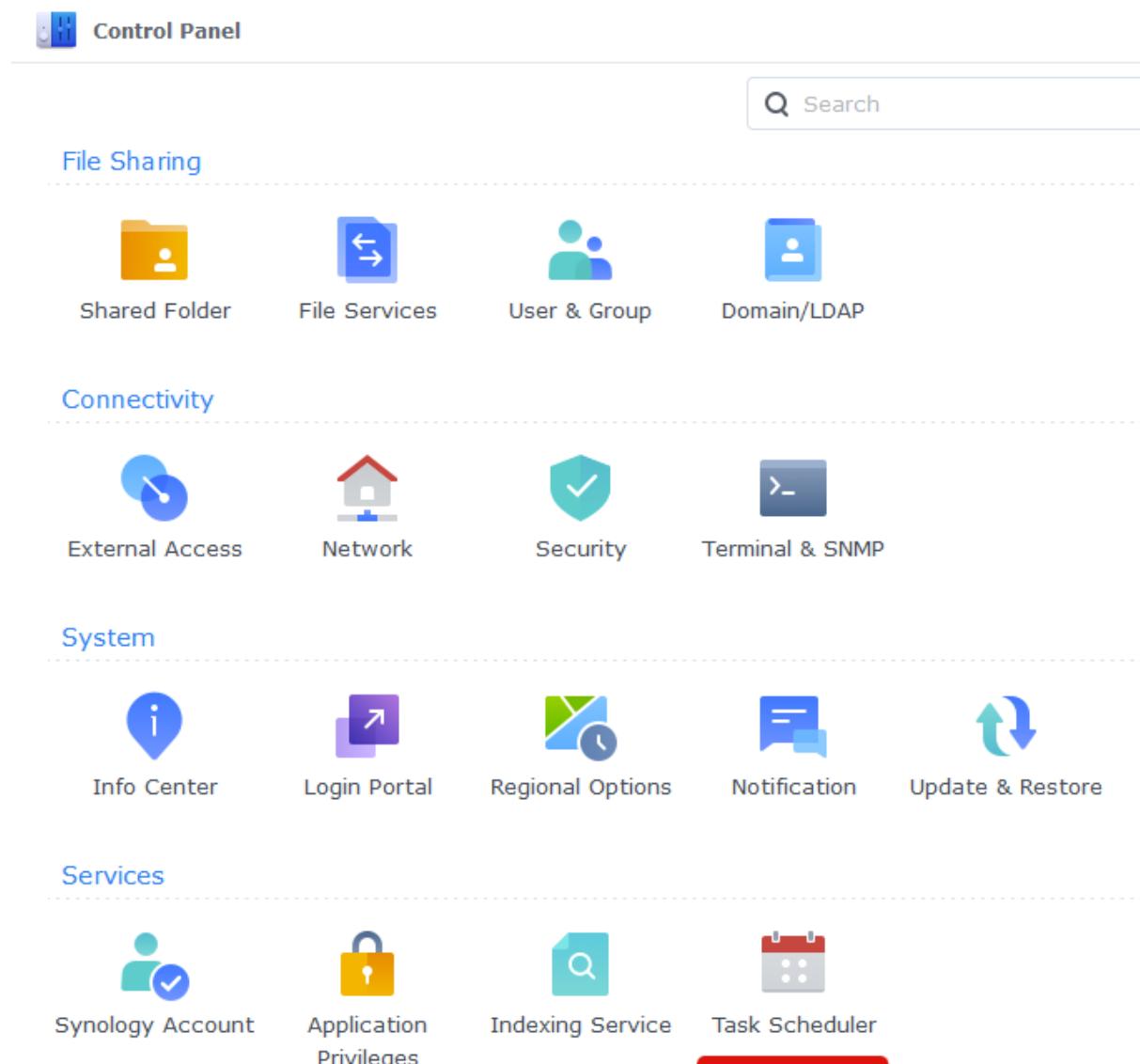
# The installation

Now that we have our folder structure ready, let's begin with the actual installation.

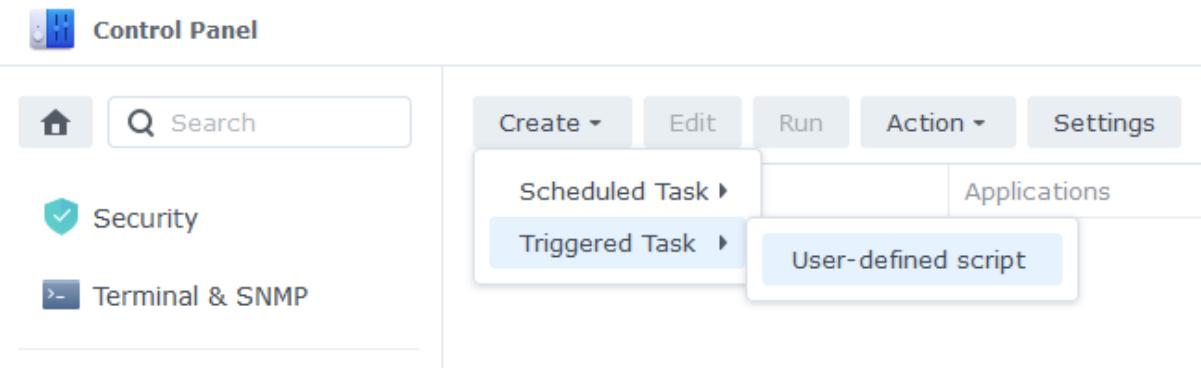
## Setting up the start up script for glueTUN

For glueTUN to start up automatically, we need to create a task on a schedule.

1. Open up control panel, then click on Task Scheduler.



2. Next click on Create, Triggered Task then User Defined Script.



3. Now enter a name for the script. It doesn't matter what you choose. The user **must** be 'root' and 'Boot-up' for the Event. Don't click OK yet.

Create task X

**General** **Task Settings**

**General Settings**

Task:	VPNTUN
User:	root
Event:	Boot-up
Pre-task:	

Enabled

Cancel OK

4. On the Task Settings tab copy and paste the code below in the 'User-Defined script' section. It will look like screenshot

On the Task Settings tab copy and paste the code below in the 'User-Defined script' section:

```
#!/bin/sh -e  
insmod /lib/modules/tun.ko
```

Create task X

General **Task Settings**

### Notification

Send run details by email

Email:

Send run details only when the script terminates abnormally

### Run command

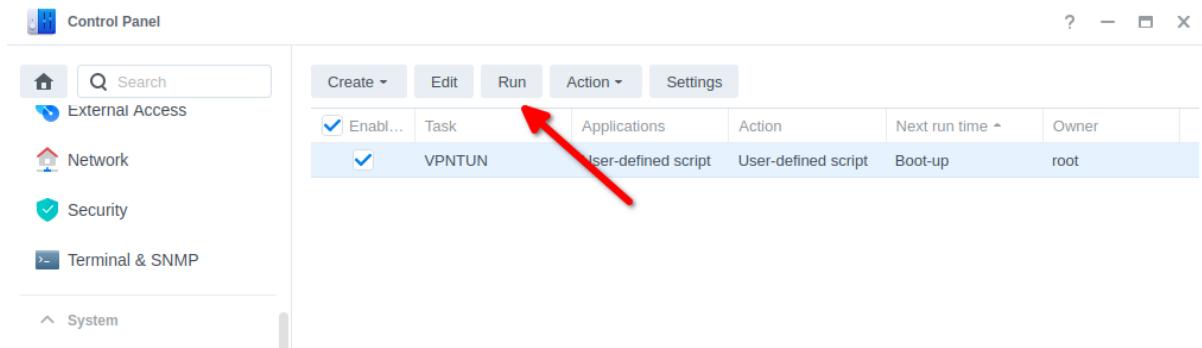
User-defined script i

```
#!/bin/sh -e  
insmod /lib/modules/tun.ko
```

Cancel

OK

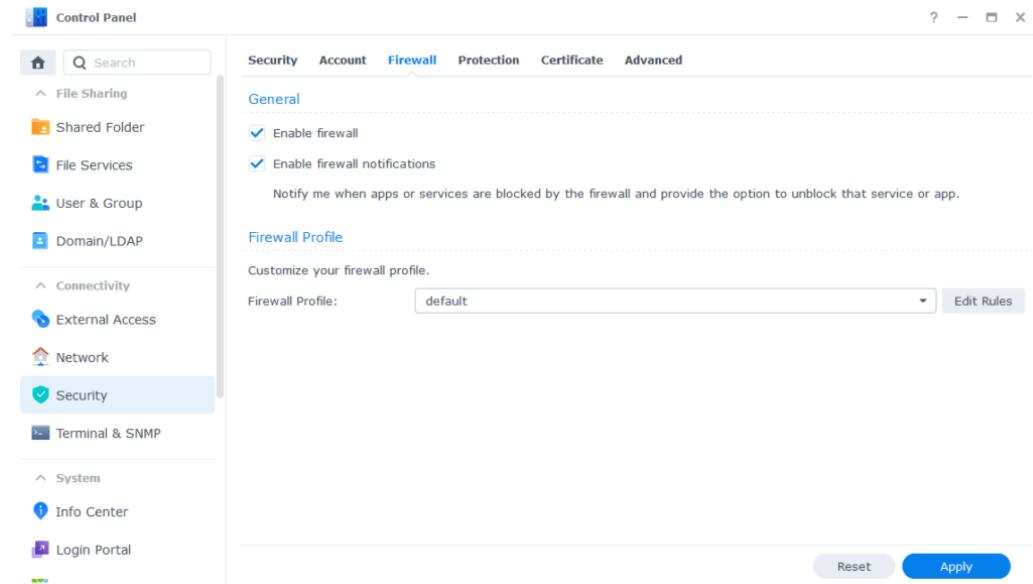
5. You can now press OK and agree to the warning message. Next run the script which will enable the TUN device.



## Firewall rules (if you have firewall set up)

If you have firewall rules set up on your synology to block all outgoing connections, we need to make some exception rules.

Go into Control Panel > Security > Firewall



Click on Edit Rules and then click on “Create”

The screenshot shows the 'Edit Profile "default"' dialog. It displays a table for 'Firewall Rules' with columns: Enabled, Ports, Protocol, Source IP, and Action. A note below states 'No items'. At the bottom, there are 'Cancel' and 'OK' buttons.

**Note:** If no rules in "All interfaces" are matched, rules in each interface will be matched.

**Note:** You can drag and drop the rules to rearrange the order. Rules at the top have higher priorities.

On the “Ports” section, select “Custom”

Create Firewall Rules X

---

**Ports**

All Select

Select from a list of built-in applications Select

Custom Custom

---

**Source IP**

All Select

Specific IP Select

Location Select

---

**Action**

Allow  Deny

Enabled

---

Cancel OK

On the screen that appears select the Type as “Destination Port” and Protocol as “All”. In this example I am going to open up both 1194 and 1195 as some providers use UDP and some TCP and these are the most commonly used ports.

## Create Firewall Rules

X

### Ports

All

Custom X

Type: Destination port ▾

Protocol: All ▾

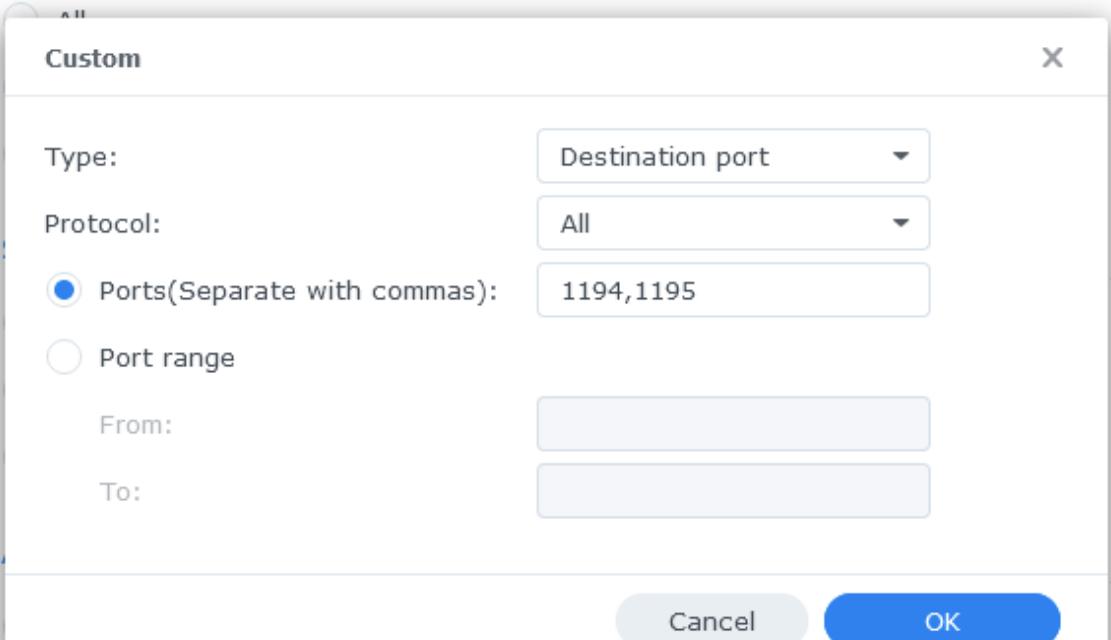
Ports(Separate with commas): 1194,1195

Port range

From:

To:

Cancel OK



Enabled

Cancel

OK

Click on OK. Leave the “Source IP” as “All” and “Action” as “Allow”, then “OK” again to apply.

## WireGuard Kernel Module

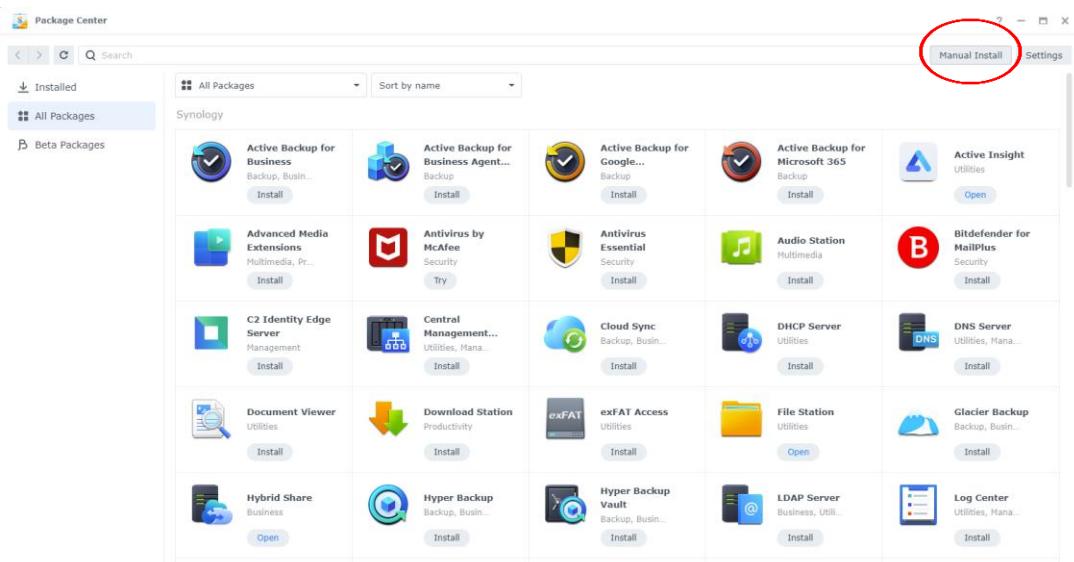
The WireGuard kernel module is not necessary, but it does lower the CPU usage a little bit. This in turn allows for better performance, better efficiency and lower electricity bills.

“The default Gluetun Wireguard setup uses a ‘Userspace’ implementation of Wireguard which normally should not use much from a CPU resource perspective. However, on Synology it tends to require high CPU utilisation. For example a 40MiB download via qBittorrent uses up to 176% in CPU (1.7 Cores) on my 1821+.”

(DrFrankenstein’s Tech Stuff)

[BlackVoid.club](https://www.blackvoid.club) have put together a Kernel Module for Synology which allows Gluetun to use the lower level Kernel to perform Wireguard duties.

1. Open this link: <https://www.blackvoid.club/wireguard-spk-for-your-synology-nas/>
2. Find your model of NAS under the correct DSM version section (If you are following this guide it will be 7.2) and download the pre compiled .spk file
3. Head into Package Center and click ‘Manual Install’ on the top right and install the .spk file and **untick** the box to run after install



4. Reboot
5. SSH Into your NAS using PuTTY, powershell or any other SSH client and elevate yourself to root by typing “sudo -l” and entering your password

6. Enter this command and press enter to start up the module

```
/var/packages/WireGuard/scripts/start
```

## Creating a Synology bridge network

As default, there is already a bridge network in container manager for Synology. The problem with the default one is that the IPs it assigns are not static, and therefore may change. This is fine for single containers that don't communicate with each other, but when connecting multiple containers with IPs the IP address need to always stay the same. To remedy this we will create our own bridge network.

1. Open up container manager and click on the network tab to the left.
2. Click "Add" at the top
3. Configure the network like this:
  - Subnet: 172.20.0.0/16
  - IP range: 172.20.0.2/25
  - Gateway: 172.20.0.1
  - IPv6: Disabled
  - IP Masquerade: enabled (Leave the "disable" option unticked)

## Docker project

Now we are ready to create the project. A docker project is just a collection of multiple docker containers.

1. Open “container manager” and head to the “Project” tab.
2. Click create
3. Give the project a name. I chose “arr-stack”
4. Set the path to be “docker > arr-stack” (should say “\docker\arr-stack”)
5. As source, select “Create docker-compose.yml”

Now this next step will be just a little bit different for everyone. What you will put in the docker-compose, will depend on what apps you plan to use and what VPN provider you have if you will use glueTUN. But don’t worry, as you can always come back to the project and edit the docker-compose to add more apps, or to fix any potential problems that may occur. I will try my best to explain what each section does, and if it is relevant to you or not.

### Required

Everyone will need to start the docker-compose out like this:

```
version: "3"
services:
```

Then under services we will add all of our apps and configs to each app.

## GlueTUN:

```
gluetun:
  image: qmcgaw/gluetun
  container_name: gluetun
  hostname: gluetun
  cap_add:
    - NET_ADMIN
  devices:
    - /dev/net/tun:/dev/net/tun
  ports:
    - 6881:6881
    - 6881:6881/udp
#      - Add all other ports required by the different apps here
  volumes:
    - <path\to\your\gluetun>:/gluetun
  environment:
    - VPN_SERVICE_PROVIDER=<your provider>
    - VPN_TYPE=<wireguard or openvpn>
    - VPN_DISABLE_IPV6=true
  # OpenVPN:
  #   - OPENVPN_USER=
  #   - OPENVPN_PASSWORD=
  # Wireguard:
  - WIREGUARD_PRIVATE_KEY=<your-private-key>
  - WIREGUARD_ADDRESSES=<your-wireguard-adress>
  - DNS=<your-wireguard-dns>
  - SERVER_HOSTNAMES=<your-hostnames>
  - SERVER_CITIES=<your-cities>
  - HTTPPROXY=off #change to on if you wish to enable
  - SHADOWSOCKS=off #change to on if you wish to enable
  # Timezone for accurate log times
  - TZ=<your-timezone>
  # Server list updater
  # See https://github.com/qdm12/gluetun-
wiki/blob/main/setup/servers.md#update-the-vpn-servers-list
    - UPDATER_PERIOD=24h
    - FIREWALL_OUTBOUND_SUBNETS=<synobridge-subnet>/16,<host-machine-subnet>/24
#change this in line with your subnet see note on guide
#      - FIREWALL_VPN_INPUT_PORTS=12345 #uncomment or remove this line based on
the notes below
  network_mode: synobridge
  labels:
```

```
- com.centurylinklabs.watchtower.enable=false  
security_opt:  
  - no-new-privileges:true  
restart: always
```

We have a lot to unpack here.

### *Ports*

It is in the “ports:” section we put in all the ports we are going to use. When we look the official docker-compose.yml to all the apps, they have their ports listed under their own service. However, since we are going to use a VPN we need it to be in the glueTUN network.

### *Volumes*

Here we put the path to the folder we made earlier. If you only have one volume on your synology and followed the same naming scheme as me, it should be “\volume1\docker\arr-stack\gluetun”. Then we mount it as “\gluetun” by adding a “:”. So the full volume mapping should be “\volume1\docker\arr-stack\gluetun:\gluetun”.

### *Environment*

It’s in the environment we put in all the config settings.

### **VPN Service provider**

In the “VPN\_SERVICE\_PROVIDER” you fill in your provider.

Here you can see a list of all the supported providers, as well as how to configure them:  
<https://github.com/qdm12/gluetun-wiki/tree/main/setup/providers>

As I am using Mullvad, that is what I will put in for my docker-compose.

You can also choose whether to use the WireGuard or OpenVPN protocol. In this example, we are using WireGuard as that is what I have found to work best.

If you don't use Mullvad, you have to find out yourself how to configure GlueTUN for your provider, as well as figure out how to find that information to put it. But if you are using Mullvad, you can follow my guide:

## Finding WireGuard details for Mullvad

1. Go to this link and put in your account number:  
<https://mullvad.net/no/account/wireguard-config>
2. Click on “Generate Key” **Note:** The displayed key is NOT the private key.
3. Choose a country, city and then the servers you wish to use. I have selected Sweden, Stockholm and all servers.
4. Click “Download zip-archive”

0

The screenshot shows the configuration interface for Mullvad. At the top, there are three dropdown menus: 'Sweden' (selected), 'Stockholm' (selected), and 'Alle servere' (selected). Below these is a section titled 'Avanserte innstillingar' (Advanced settings) with a dropdown arrow. The next section is 'Konfigurerer innholdsblokkering' (Configure content blocking). It asks 'Veg hvilken type innhold som skal blokkeres' (Select which type of content should be blocked) and provides two buttons: 'Velg alle' (Select all) and 'Ingen' (None). Below these are six checkboxes: 'Annonser' (Selected), 'Sporing' (Unselected), 'Skadelig programvare' (Unselected), 'Vokseninnhold' (Unselected), 'Pengespill' (Unselected), and 'Sosiale medier' (Unselected). At the bottom, there is a button labeled 'Last ned zip-arkiv' (Download zip archive).

5. Inside the zip file, you will find a bunch on .conf files. (Might be .json files) Open these in notepad.
6. You only need one of them, as the relevant information is the same in every single one. You need to look at your [Interface] section and copy your “PrivateKey” as well as “Address” and “DNS”.

```

[Interface]
# Device: Hip Prawn
PrivateKey = iExD5V5kkXnh+40dyo/PmCL1aus8eNBdHQMWergYFWo=
Address = 10.72.171.113/32,fc00:bbbb:bbbb:bb01::9:ab70/128
DNS = 10.64.0.1

[Peer]
PublicKey = MkP/Jytkg51/Y/EostONjIN6YaFRpsAYiNKMx27/CAY=
AllowedIPs = 0.0.0.0/0,::0/0
Endpoint = 185.195.233.76:51820

```

So for this example I would have to use:

“iExD5V5kkXnh+40dyo/PmCL1aus8eNBdHQMWergYFWo=” as my private key,  
 “10.72.171.113/32” as my address and  
 “10.64.0.1” as my DNS.

7. Also take note of the filenames. They should be something like “se-sto-wg-001.conf”, where “se” is the country, “sto” the city and “wg” the protocol. So for me it’s Sweden, Stockholm, WireGuard.

## Proxy

If you are using a proxy, you can enable httpproxy and shadowsocks. I have not used it, and will therefor not go over that now.

## Timezone

In the timezone you just put in your own timezone. To find your TZ format, find your region in this list:

[https://en.wikipedia.org/wiki/List\\_of\\_tz\\_database\\_time\\_zones](https://en.wikipedia.org/wiki/List_of_tz_database_time_zones)

## *Firewall Outbound Subnets*

In this line:

“FIREWALL\_OUTBOUND\_SUBNETS=<your-bridge-subnet>/,<your-host-subnet>”

We need to change the first IP to the one we just made for our synobridge. If you followed me, it will be 172.20.0.0/16. Then we will need to fill in our host machines subnet,

We have a few choices on how to identify it. If you are connected to the same network on your PC or phone, you can just figure out your IP address on your device of choice. So for PC, open “cmd” and type “ipconfig”. Look for IPv4 section.

If you have a iPhone, you can open settings >Wi-Fi > i (next to the Wi-Fi named) then scroll down to find “IPv4 address.” There should be a “IP Adress” field there.

The IPv4 address will probably be something like this “192.168.X.X”. Take the first 3 digits, e.g. “192.168.0” or whatever you have, then replace the last digit with a 0. So if your IP address on your phone is “192.168.0.53” the subnet would be “192.168.0.0”. Then we just add the network mask at the end. If you don’t know what that means, it’s probably /24 at the end, so the full subnet is “192.168.0.0/24”.

So my line would look like this:

“FIREWALL\_OUTBOUND\_SUBNETS=172.20.0.0/16,192.168.0.0/24”

#### *Network\_mode*

We just need to specify it to use the network we created earlier. Leave it as is.

#### *security\_opt:*

This line just makes it so Watchtower doesn’t update it automatically if you use that. That is because it will break the whole arr-stack each time. So best practice would be to just leave it in.

#### *Restart: always*

This line tells the container to restart if it shuts down unexpectedly.

## *Full GlueTUN docker-compose.yml*

With all of the info above, I now need you to make the appropriate edits for your docker-compose. This will be mine for this example:

```
gluetun:
  image: qmcgaw/gluetun
  container_name: gluetun
  hostname: gluetun
  cap_add:
    - NET_ADMIN
  devices:
    - /dev/net/tun:/dev/net/tun
  ports:
    - 6881:6881
    - 6881:6881/udp
  #     - 8888:8888/tcp # HTTP proxy
  #     - 8388:8388/tcp # Shadowsocks
  #     - 8388:8388/udp # Shadowsocks
    - 8085:8085 # qbittorrent
    - 8989:8989 # Sonarr
    - 9696:9696 # Prowlarr
    - 7878:7878 # Radarr
    - 8686:8686 #Lidarr
    - 8191:8191 #FlareSolverr
    - 5055:5055 #Overseer
    - 4545:4545 #Requestrr

  volumes:
    - \volume1\docker\arr-stack\gluetun:/gluetun
  environment:
    - VPN_SERVICE_PROVIDER=mullvad
    - VPN_TYPE=wireguard
    - VPN_DISABLE_IPV6=true
  # OpenVPN:
  #   - OPENVPN_USER=
  #   - OPENVPN_PASSWORD=
  # Wireguard:
    - WIREGUARD_PRIVATE_KEY=iExD5V5kkXnh+40dyo/PmCL1aus8eNBdHQMWergYFwo=
    - WIREGUARD_ADDRESSES=10.72.171.113/32
    - DNS=10.64.0.
    - SERVER_HOSTNAMES=se-sto-wg-001,se-sto-wg-002,se-sto-wg-003,se-sto-wg-004
    - SERVER_CITIES=stockholm
    - HTTPPROXY=off #change to on if you wish to enable
```

```
- SHADOWSOCKS=off #change to on if you wish to enable
# Timezone for accurate log times
- TZ=Europe/Oslo
# Server list updater
# See https://github.com/qdm12/gluetun-
wiki/blob/main/setup/servers.md#update-the-vpn-servers-list
- UPDATER_PERIOD=24h
- FIREWALL_OUTBOUND_SUBNETS=172.20.0.0/192.168.0.0/24 #change this in line
with your subnet see note on guide
#     - FIREWALL_VPN_INPUT_PORTS=12345 #uncomment or remove this line based on
the notes below
network_mode: synobridge
labels:
- com.centurylinklabs.watchtower.enable=false
security_opt:
```

## Download client

The best download client in my opinion is qBitTorrent and is therefore what I will use today. Here is the docker-compose:

```
qbittorrent:
  image: lscr.io/linuxserver/qbittorrent
  container_name: qbittorrent
  network_mode: "service:gluetun"
  environment:
    - PUID=<your-UID>
    - PGID=<your-GID>
    - TZ=<your-timezone>
    - WEBUI_PORT=8085
    - UMASK=022
  volumes:
    - \path\to\your\config:/config
    - \path\to\your\media\torrents:/Media/Torrents
  depends_on:
    - gluetun:
        condition: service_healthy
  security_opt:
    - no-new-privileges:true
  restart: always
```

### *network\_mode*

If you also are using a VPN, and therefore by extension gluetun, you will need to set *network\_mode* to “service:gluetun”. If you don’t use VPN, you can just do “*network\_mode*: bridge” or any other network you have set up.

### *Environment*

#### PUID and PGID

For PUID and PGID you need to find the ID for your user. To do this, SSH into your synology, log in to your own user, then type “id”. The output should include UID (user ID) and GID (Group ID).

## UMASK

I simply don't have enough information on this to tell you what it is or how it works. If you are interested, you can read here: <https://en.wikipedia.org/wiki/Umask>

## Volumes

If you followed me in the creation of the directories, it should be:

```
\volume1\docker\arr-stack\qbittorrent\config:/config
```

```
\volume1\Media/Torrents:/Media/Torrents
```

**NOTE:** It is very important that we have torrents in the same shared folder as our media library. This is because of how hardlinks work. If we do not have it in the same folder, we will then copy the files. So for every 10GB file, you would have to use 20GB of your storage. This adds up really quickly to many TB wasted storage.

### *Full docker-compose.yml for qbittorrent*

The full docker-compose could look something like this:

```
qbittorrent:
  image: lscr.io/linuxserver/qbittorrent
  container_name: qbittorrent
  network_mode: "service:gluetun"
  environment:
    - PUID=1026
    - PGID=100
    - TZ=Europe/Oslo
    - WEBUI_PORT=8085
    - UMASK=022
  volumes:
    - /volume1/docker/arr-stack/qbittorrent/config:/config
    - /volume1/Media/Torrents:/media/Torrents
  depends_on:
    - gluetun:
        condition: service_healthy
  security_opt:
    - no-new-privileges:true
  restart: always
```

## Sonarr

This is the docker-compose.yml

```
sonarr:
  image: lscr.io/linuxserver/sonarr:latest
  container_name: sonarr
  network_mode: "service:gluetun"
  environment:
    - PUID=1026
    - PGID=100
    - TZ=Europe/Oslo
  volumes:
    - /path\to\your\config:/config
    - /path\to\your\media:/Media
  depends_on:
    gluetun:
      condition: service_healthy
  restart: unless-stopped
```

We have already covered most of the components in the docker-compose file, so I won't repeat myself. The only thing I want to mention is the fact that if you separate your TV shows and Anime shows in your Media folder, like me, you need to mount both of them here.

### *Something important:*

The path to your media **NEED** to be in the same shared folder as the torrents, for the reason we talked about earlier.

*Full docker-compose.yml file*

```
sonarr:
  image: lscr.io/linuxserver/sonarr:latest
  container_name: sonarr
  network_mode: "service:gluetun"
  environment:
    - PUID=1026
    - PGID=100
    - TZ=Europe/Oslo
  volumes:
    - /volume1/docker/arr-stack/sonarr/config:/config
    - /volume1/Media:/Media
  depends_on:
    gluetun:
      condition: service_healthy
  restart: unless-stopped
```

## Radarr

Here we have radarr's docker-compose:

```
radarr:
  image: lscr.io/linuxserver/radarr:latest
  container_name: radarr
  network_mode: "service:gluetun"
  environment:
    - PUID=1026
    - PGID=100
    - TZ=Europe/Oslo
  volumes:
    - /volume1/docker/arr-stack/radarr/config:/config
    - /volume1/Media:/Media
  depends_on:
    gluetun:
      condition: service_healthy
  restart: unless-stopped
```

So, as you can see, it's identical to Sonarr's, except for the image and name.

## Lidarr

Docker-compose:

```
lidarr:
  image: lscr.io/linuxserver/lidarr:latest
  container_name: lidarr
  network_mode: "service:gluetun"
  environment:
    - PUID=1026
    - PGID=100
    - TZ=Europe/Oslo
  volumes:
    - /volume1/docker/arr-stack/lidarr/config:/config
    - /volume1/Media:/Media
  depends_on:
    gluetun:
      condition: service_healthy
  restart: unless-stopped
```

Again, we have a pretty familiar looking docker-compose file. Edit the appropriate settings just like you did for radar.

## Prowlarr

Prowlarr is what we use to search for the files we are going to download. Here is the docker-compose:

```
prowlarr:
  image: lscr.io/linuxserver/prowlarr:latest
  container_name: prowlarr
  network_mode: "service:gluetun"
  environment:
    - PUID=1026
    - PGID=100
    - TZ=Europe/Oslo
  volumes:
    - path\to\your\config:/config
  depends_on:
    gluetun:
      condition: service_healthy
  restart: unless-stopped
```

Obviously change the PUID and PGID. Other than that we only have to change the config path. It should be “\volume1\docker\arr-stack\prowlarr\config”.

## Flaresolverr

Docker-compose:

```
flaresolverr:
    image: ghcr.io/flaresolverr/flaresolverr:latest
    container_name: flaresolverr
    network_mode: "service:gluetun"
    environment:
        - TZ=Europe/Oslo
    depends_on:
        gluetun:
            condition: service_healthy
    security_opt:
        - no-new-privileges:true
    restart: unless-stopped
```

You don't need to change anything here. Just paste it in. Only exception is if you don't use VPN, then change "network\_mode" and delete "depends\_on".

## Overseerr

Docker-compose:

```
overseerr:
  image: sctx/overseerr:latest
  container_name: overseerr
  network_mode: "service:gluetun"
  environment:
    - LOG_LEVEL=debug
    - TZ=Europe/Oslo
  volumes:
    - \path\to\your\config:/app/config
  depends_on:
    gluetun:
      condition: service_healthy
  restart: unless-stopped
```

The only thing you need to change here is the path to your config. It should be “\volume1\docker\arr-stack\overseerr\config”

## Requestr

Docker-compose:

```
requestr:
  image: darkalfx/requestr
  container_name: requestr
  network_mode: "service:gluetun"
  volumes:
  - \path\to\your\config:/root/config
  depends_on:
    gluetun:
      condition: service_healthy
  restart: unless-stopped
```

Again, only change the config path. Should be “\volume1\docker\arr-stack\requestr\config”

## Tautulli

Docker-compose:

```
tautulli:
  image: ghcr.io/tautulli/tautulli
  container_name: tautulli
  network_mode: service:gluetun #Replace this bridge if you don't use GlueTUN
  restart: unless-stopped
  volumes:
    - \path\to\your\config:/config
  environment:
    - PUID=<your UID>
    - PGID=<your GID>
    - TZ=Europe/Oslo
  depends_on: #use this if you use GlueTUN for VPN
    gluetun:
      condition: service_healthy
```

For me it would look like this:

```
tautulli:
  image: ghcr.io/tautulli/tautulli
  container_name: tautulli
  network_mode: service:gluetun
  restart: unless-stopped
  volumes:
    - /volume1/docker/arr-stack/tautulli/config:/config
  environment:
    - PUID=1026
    - PGID=100
    - TZ=Europe/Oslo
  depends_on:
    gluetun:
      condition: service_healthy
```

## Putting it all together

Now we can put all our relevant docker-compose files together inside one, big file. Remember, if you want to add more -arr apps or other download-clients, you can just add them in this docker-compose just like we did for all the other apps. Just remember to open up the ports in GlueTUN as well. For me the docker-compose.yml is like this:

```
version: "3"
services:
  gluetun:
    image: qmcgaw/gluetun
    container_name: gluetun
    hostname: gluetun
    cap_add:
      - NET_ADMIN
    devices:
      - /dev/net/tun:/dev/net/tun
    ports:
      - 6881:6881
      - 6881:6881/udp
      - 8085:8085 # qbittorrent
      - 8989:8989 # Sonarr
      - 9696:9696 # Prowlarr
      - 7878:7878 # Radarr
      - 8686:8686 #Lidarr
      - 8191:8191 #FlareSolverr
      - 5055:5055 #Overseerr
      - 4545:4545 #Requestr
    volumes:
      - \volume1\docker\arr-stack\gluetun:/gluetun
  environment:
    - VPN_SERVICE_PROVIDER=mullvad
    - VPN_TYPE=wireguard
    - VPN_DISABLE_IPV6=true
    # OpenVPN:
    # - OPENVPN_USER=
    # - OPENVPN_PASSWORD=
    # Wireguard:
    - WIREGUARD_PRIVATE_KEY= iExD5V5kkXnh+40dyo/PmCL1aus8eNBdHQMWergYFWo=
    - WIREGUARD_ADDRESSES=10.72.171.113/32
    - DNS=10.64.0.
    - SERVER_HOSTNAMES=se-sto-wg-001,se-sto-wg-002,se-sto-wg-003,se-sto-wg-004
```

```

- SERVER_CITIES=stockholm
- HTTPPROXY=off #change to on if you wish to enable
- SHADOWSOCKS=off #change to on if you wish to enable
# Timezone for accurate log times
- TZ=Europe/Oslo
# Server list updater
# See https://github.com/qdm12/gluetun-
wiki/blob/main/setup/servers.md#update-the-vpn-servers-list
- UPDATER_PERIOD=24h
- FIREWALL_OUTBOUND_SUBNETS=172.20.0.0/192.168.0.0/24 #change this in line
with your subnet see note on guide
# - FIREWALL_VPN_INPUT_PORTS=12345 #uncomment or remove this line based on
the notes below
network_mode: synobridge
labels:
- com.centurylinklabs.watchtower.enable=false
security_opt:

qbittorrent:
image: lscr.io/linuxserver/qbittorrent
container_name: qbittorrent
network_mode: "service:gluetun"
environment:
- PUID=1026
- PGID=100
- TZ=Europe/Oslo
- WEBUI_PORT=8085
- UMASK=022
volumes:
- /volume1/docker/arr-stack/qbittorrent/config:/config
- /volume1/Media/Torrents:/Media/Torrents
depends_on:
gluetun:
condition: service_healthy
security_opt:
- no-new-privileges:true
restart: always

sonarr:
image: lscr.io/linuxserver/sonarr:latest
container_name: sonarr
network_mode: "service:gluetun"
environment:
- PUID=1026
- PGID=100
- TZ=Europe/Oslo

```

```

volumes:
  - /volume1/docker/arr-stack/sonarr/config:/config
  - /volume1/Media:/Media
depends_on:
  gluetun:
    condition: service_healthy
restart: unless-stopped

prowlarr:
  image: lscr.io/linuxserver/prowlarr:latest
  container_name: prowlarr
  network_mode: "service:gluetun"
  environment:
    - PUID=1026
    - PGID=100
    - TZ=Europe/Oslo
  volumes:
    - /volume1/docker/arr-stack/prowlarr/config:/config
depends_on:
  gluetun:
    condition: service_healthy
restart: unless-stopped

radarr:
  image: lscr.io/linuxserver/radarr:latest
  container_name: radarr
  network_mode: "service:gluetun"
  environment:
    - PUID=1026
    - PGID=100
    - TZ=Europe/Oslo
  volumes:
    - /volume1/docker/arr-stack/radarr/config:/config
    - /volume1/Media:/Media
depends_on:
  gluetun:
    condition: service_healthy
restart: unless-stopped

lidarr:
  image: lscr.io/linuxserver/lidarr:latest
  container_name: lidarr
  network_mode: "service:gluetun"
  environment:
    - PUID=1026
    - PGID=100

```

```

    - TZ=Europe/Oslo
volumes:
    - /volume1/docker/arr-stack/lidarr/config:/config
    - /volume1/Media:/Media
depends_on:
    gluetun:
        condition: service_healthy
restart: unless-stopped

flaresolverr:
    image: ghcr.io/flaresolverr/flaresolverr:latest
    container_name: flaresolverr
    network_mode: "service:gluetun"
    environment:
        - TZ=Europe/Oslo
depends_on:
    gluetun:
        condition: service_healthy
security_opt:
    - no-new-privileges:true
restart: unless-stopped

overseerr:
    image: sctx/overseerr:latest
    container_name: overseerr
    network_mode: "service:gluetun"
    environment:
        - LOG_LEVEL=debug
        - TZ=Europe/Oslo
volumes:
    - /volume1/docker/arr-stack/overseer/config:/app/config
depends_on:
    gluetun:
        condition: service_healthy
restart: unless-stopped

requestrr:
    image: darkalfx/requestrr
    container_name: requestrr
    network_mode: "service:gluetun"
    volumes:
    - /volume1/docker/arr-stack/requestrr/config:/root/config
depends_on:
    gluetun:
        condition: service_healthy
restart: unless-stopped

```

```
tautulli:
  image: ghcr.io/tautulli/tautulli
  container_name: tautulli
  network_mode: service:gluetun
  restart: unless-stopped
  volumes:
    - /volume1/docker/arr-stack/tautulli/config:/config
  environment:
    - PUID=1026
    - PGID=100
    - TZ=Europe/Oslo
  depends_on:
    gluetun:
      condition: service_healthy
```

## Common Errors

The most common error to get now is “gluetun is unhealthy” If you get this, it is likely an error in the config file. Usually, it relates to the provider specific elements. If you check the logs for the GlueTUN container it will tell you why it couldn’t connect. My best guess would be incorrect private key, or something similar. If you can’t figure it out, please drop a comment or DM me somehow with your logs, and I’ll look.

Also please, please, please! Double check the volume mounts. If they are not correctly set up, you will lose **HALF** your storage space to waste. It should be as follows:

### Media

```
└── Torrents
    ├── Incomplete
    └── Complete
        ├── Movies
        ├── Music
        ├── Anime TV Shows
        └── TV Shows

    └── Usenet
        ├── Incomplete
        └── Complete
            ├── Movies
            ├── Music
            ├── Anime TV Shows
            └── TV Shows

    └── Media
        ├── Movies
        ├── Music
        ├── Anime TV Shows
        └── TV Shows
```



# Configuration of the apps

## qBitTorrent

### Login

The first thing we should do is configure qBitTorrent. Open up a web browser /on your computer) then type in your NAS IP address followed by port 8085. For this example it would look like this: 192.168.0.2:8085

You will then get to the login page for the qBitTorrent WebUI. The username is always admin. The password could be adminadmin, as this is the default. But most likely you will find a temporary password in the logs for the qBitTorrent container inside container manager.

Time	Log
11/21/2023 21:00	crond[146]: USER root pid 189 cmd run-parts /etc/periodic/hourly
11/21/2023 21:00	crond[146]: USER root pid 188 cmd run-parts /etc/periodic/15min
11/21/2023 20:58	[ls.io-init] done.
11/21/2023 20:58	Connection to localhost (127.0.0.1) 8090 port [tcp/*] succeeded!
11/21/2023 20:58	You should set your own password in program preferences.
11/21/2023 20:58	The WebUI administrator password was not set. A temporary password is provided for this session: eGszghhqe
11/21/2023 20:58	The WebUI administrator username is: admin
11/21/2023 20:58	

## Change username and password

The first thing to do now that you are logged in is to change your username and password. Click the cog icon at the top of the page, then go to the Web UI tab. Put in your new details then click save at the bottom of the page.

**NOTE:** Please change both your username and password for maximum security.

The screenshot shows the 'Options' window with the 'Web UI' tab selected. The 'Authentication' section is highlighted with a red box, specifically around the 'Username' and 'Password' fields which contain 'changeme' and '.....' respectively. Other visible settings include port 8090, UPnP/NAT-PMP options, and session timeout configurations.

Options

Behavior Downloads Connection Speed BitTorrent RSS Web UI Advanced

Web User Interface (Remote control)

IP address: \* Port: 8090

Use UPnP / NAT-PMP to forward the port from my router

Use HTTPS instead of HTTP

Certificate: [ ]

Key: [ ]

Information about certificates

Authentication

Username: changeme

Password: .....

Bypass authentication for clients on localhost

Bypass authentication for clients in whitelisted IP subnets

Example: 172.17.32.0/24, fdff:ffff:c8::/40

Ban client after consecutive failures: 20

ban for: 3600 seconds

Session timeout: 3600 seconds

Use alternative Web UI

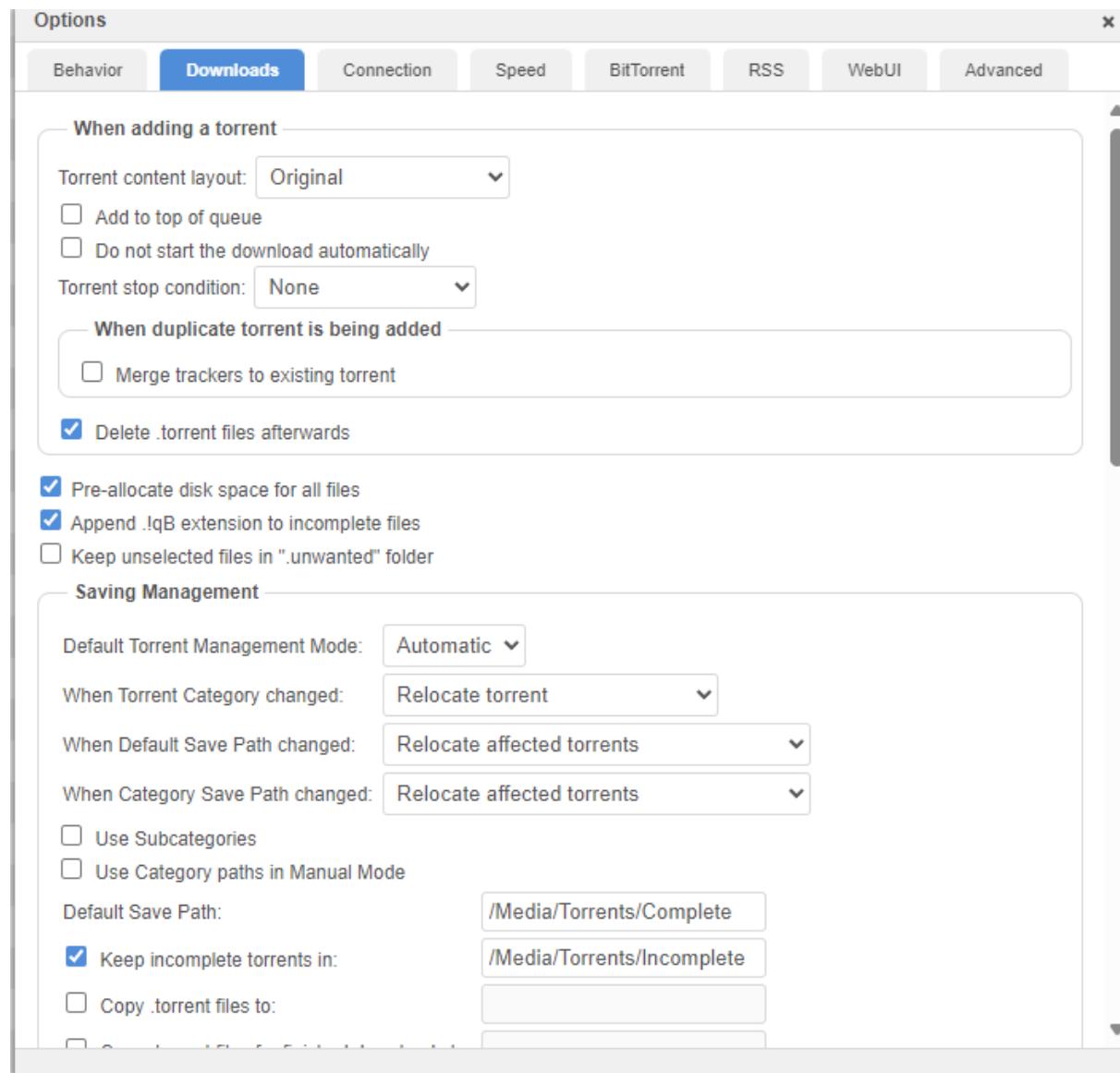
Files location: [ ]

Security

## Change downloads path

Go back to settings, then go to the “Downloads” tab.

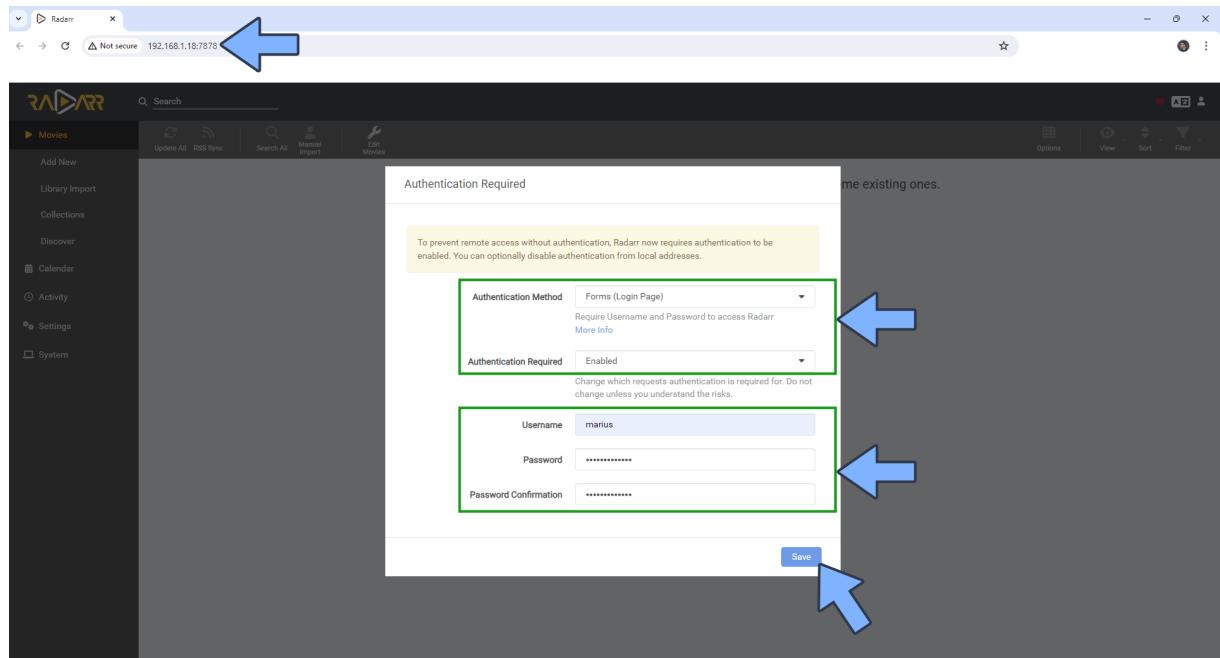
Change the default save path to “/Media/Torrents/Complete” and tick the box for “Keep incomplete torrents in:”. Select “/Media/Torrents/Incomplete”



# Radarr

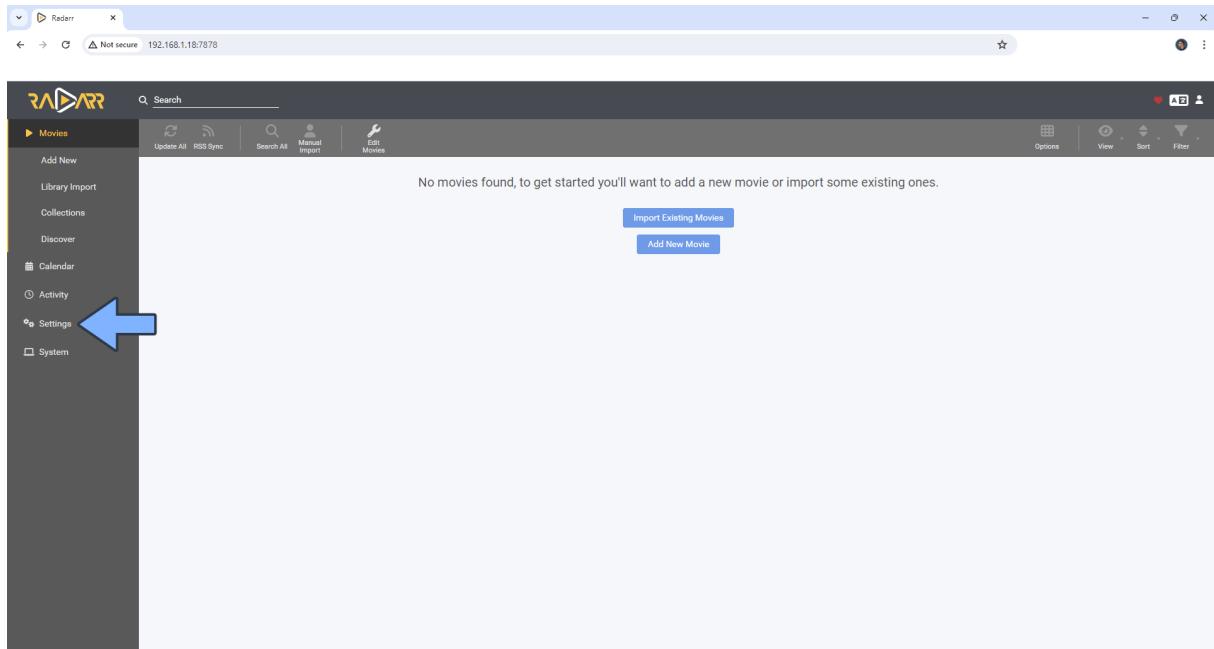
## Adding Authentication method

Open a web browser on your PC, then type in the IP of your NAS followed by port 7878. So e.g. 192.168.0.2:7878. The first time you open this, you will be prompted to add an authentication method. I recommend to use forms and have it enabled. Set username to something else than admin or administrator as this is easy to guess.

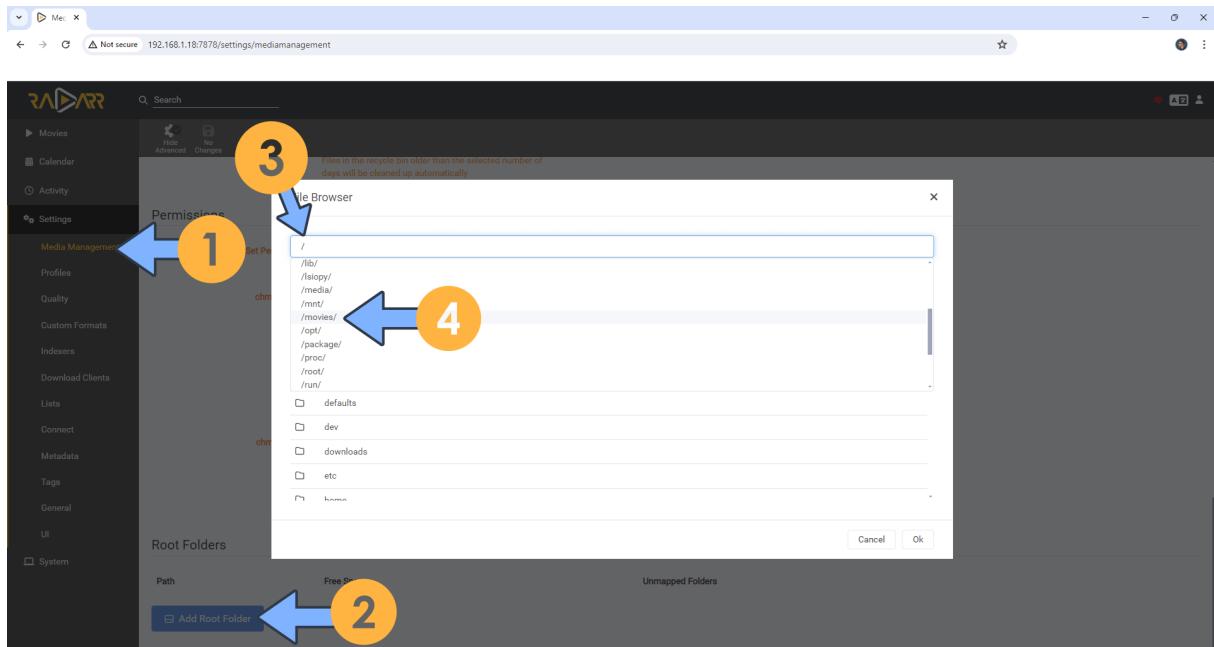


## Adding Root Folder

Now go to settings on the left-hand side menu.

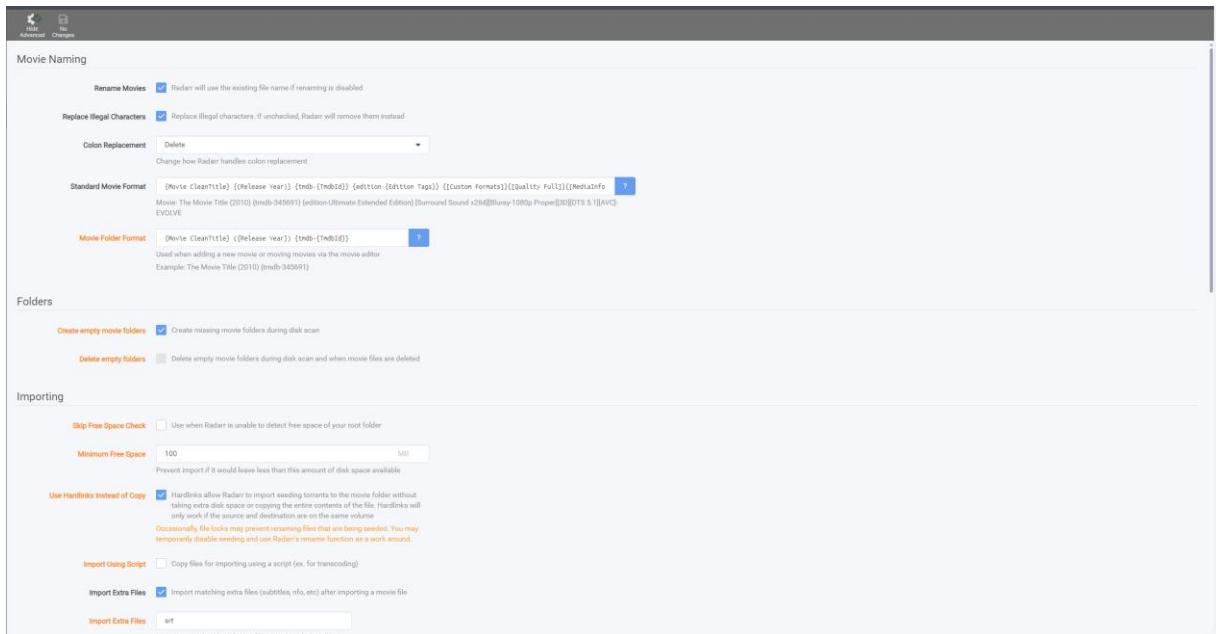


Now go to media management, scroll down and click on “Add Root Folder”. Type /Media/Media/Movies and select it. Click OK.

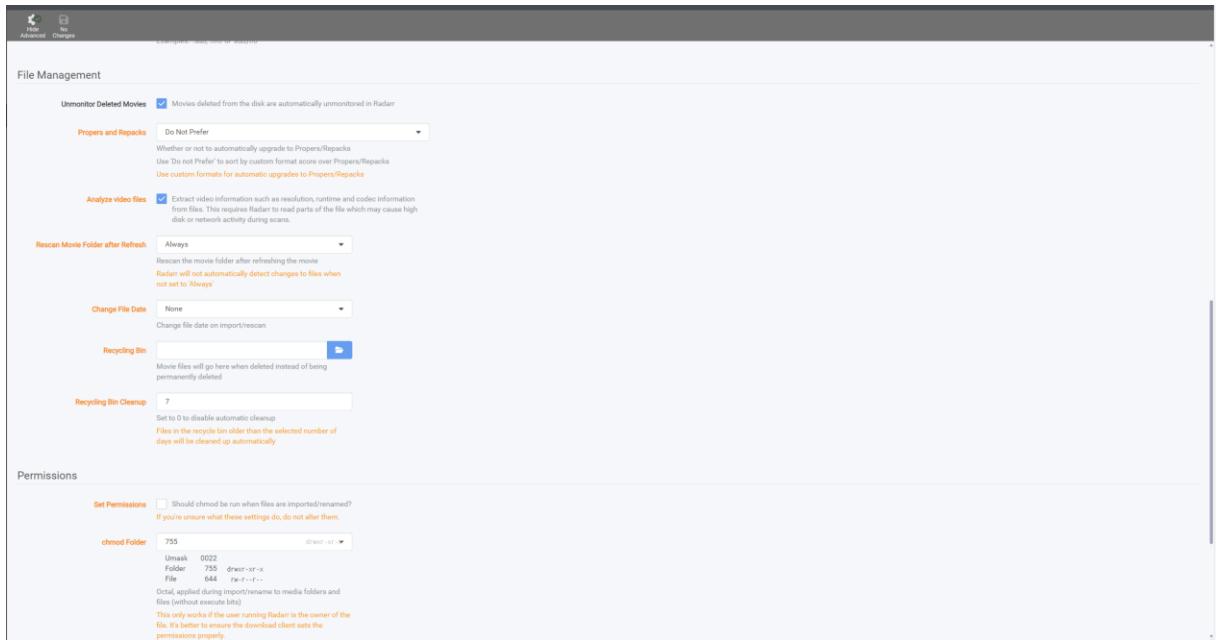


## Changing Movie Naming Scheme

1. Go to settings, then choose media management. Turn on advanced settings (right under the search bar).
2. Tick the box next to “Rename Movies” and “Replace illegal characters.”
3. Change the “Standard Movie Format” to the one that fits you the best on TRaSH’s website. For Plex, I recommend to stick with Plex (TVDB). You can read more about recommended naming schemes here: <https://trash-guides.info/Radarr/Radarr-recommended-naming-scheme/>
4. For Movie Folder Format I would go for “[Movie CleanTitle] ({Release Year}) {tmdb-{TmdbId}}”, but again you can read more on TRaSH Guides.



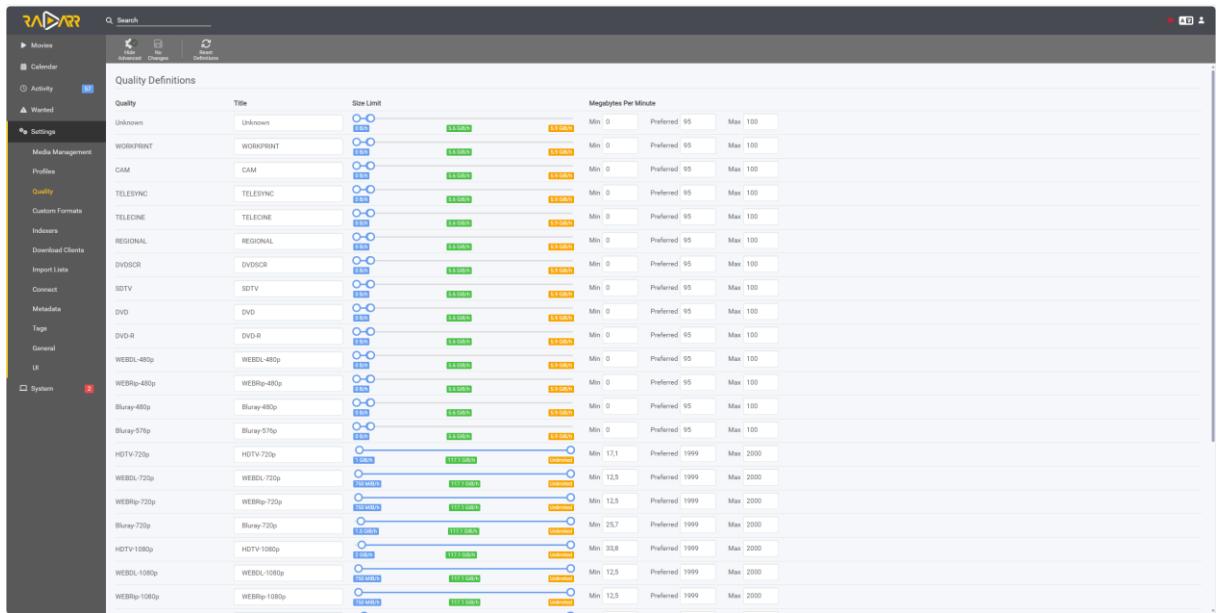
5. Make sure “Use Hardlinks instead of Copy” under “Importing” is enabled.
6. Scroll down until you get to “File Management” and select “Do Not Prefer for” “Props and Repacks”. This is important for our quality profiles later.



Now be sure to click save, next to the “Show advanced” switch under the search bar.

## Quality Settings (File Size)

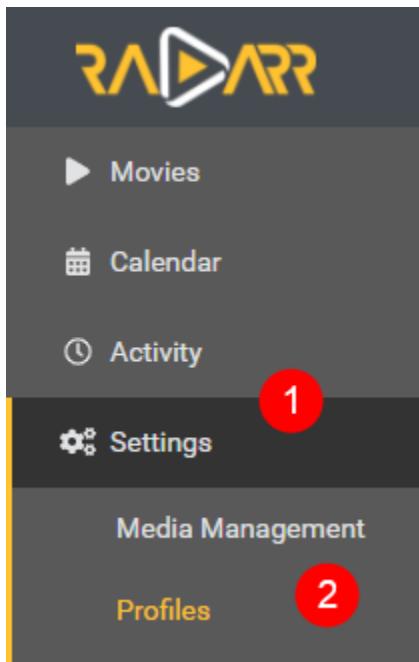
1. Go to settings, then select “Quality” from the menu. Make sure to enable “Show Advanced” by clicking the cog under the search bar.
2. Go to TRaSH Guids and change the appropriate settings as described.  
<https://trash-guides.info/Radarr/Radarr-Quality-Settings-File-Size/>



The screenshot shows the Radarr application interface with the 'Quality' section selected in the sidebar. The main area displays a table titled 'Quality Definitions' with columns for 'Quality', 'Title', 'Size Limit', and 'Megabytes Per Minute'. The 'Quality' column lists various media formats like Unknown, WORKPRINT, CAM, TELESYNC, TELECINE, REGIONAL, DVDSCR, SDTV, DVD, DVD-R, WEBDL-480p, WEBRip-480p, Blu-ray-480p, Blu-ray-576p, Blu-ray-720p, HDTV-720p, WEBDL-720p, WEBRip-720p, Blu-ray-720p, HDTV-1080p, WEBDL-1080p, and WEBRip-1080p. The 'Title' column provides specific names for each quality level. The 'Size Limit' column contains numerical values such as 1.00GB, 2.00GB, 4.00GB, etc. The 'Megabytes Per Minute' column includes dropdown menus for 'Min', 'Preferred', and 'Max' values, with some entries like 'HDTV-720p' having higher maximum values (e.g., 2000 MB/min).

## Quality profiles

1. Go to settings, Profiles.



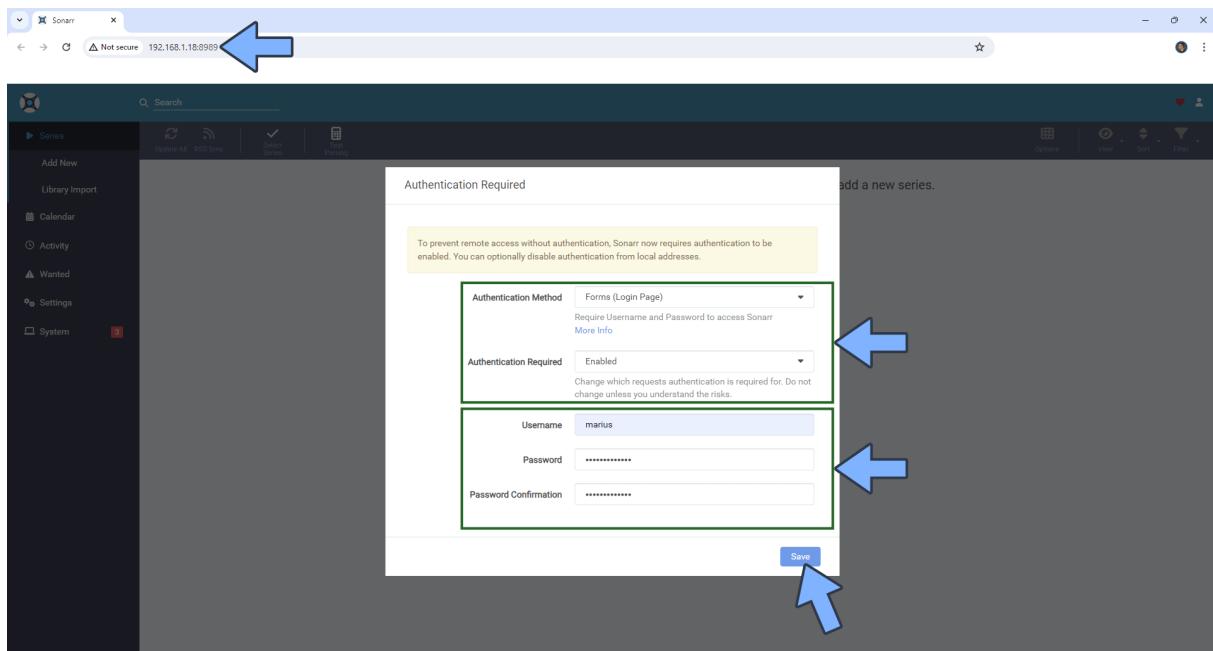
2. Delete all of the default ones.
3. Create a new one
4. Go to TRaSH Guide's and follow their instructions on how to setup quality profiles, aswell as to set custom formats. Add all the profiles you need for you setup, needs and wants.

<https://trash-guides.info/Radarr/radarr-setup-quality-profiles/>

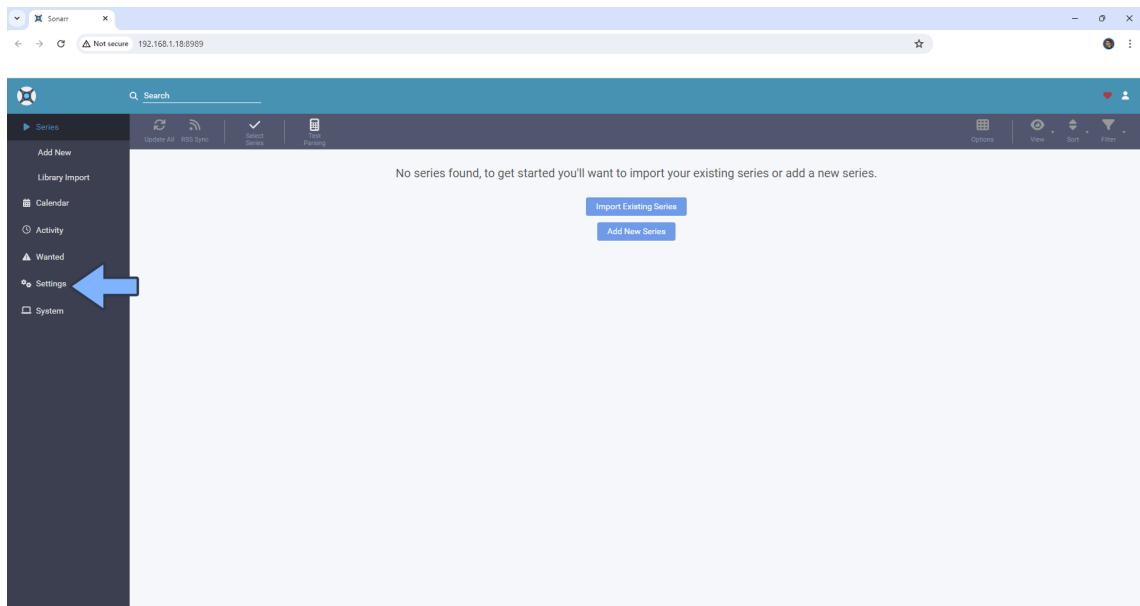
# Sonarr

## Adding Root Folder(s)

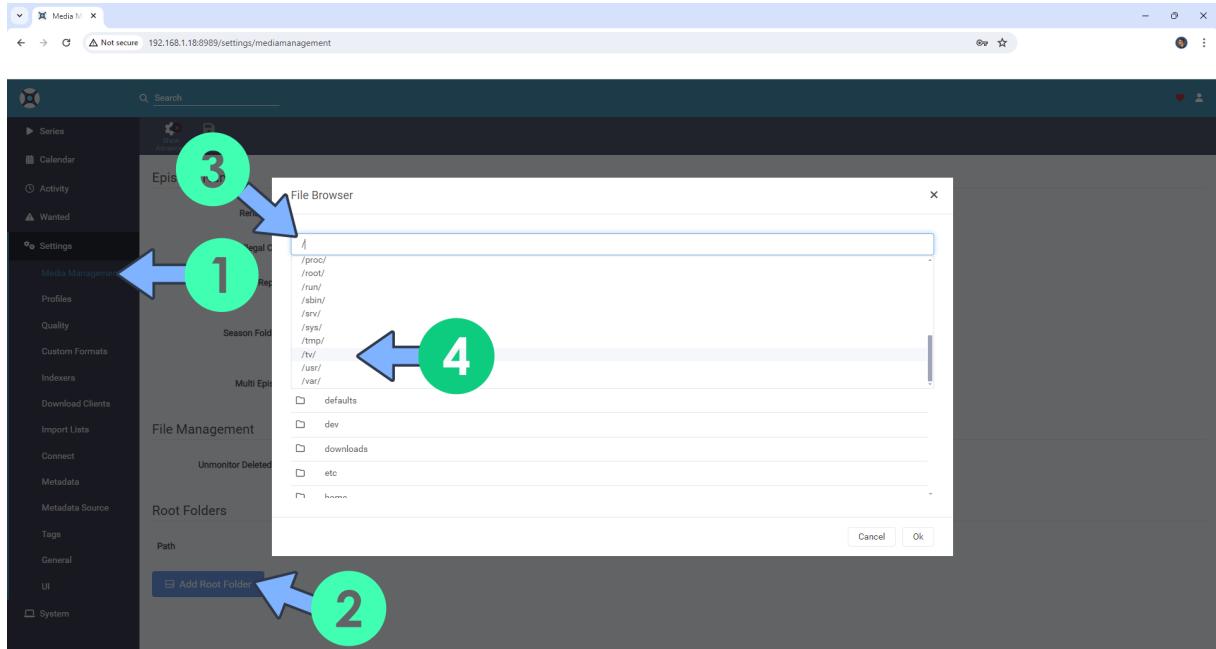
1. Open a web browser on your PC and type in the IP of your NAS followed by port 8989. So e.g. 192.168.0.2:8989.
2. Here you will be prompted to set up an authentication method. I recommend selecting “Forms” and having it enabled. Choose a username that is not admin or administrator, then select a password.



3. Go to settings

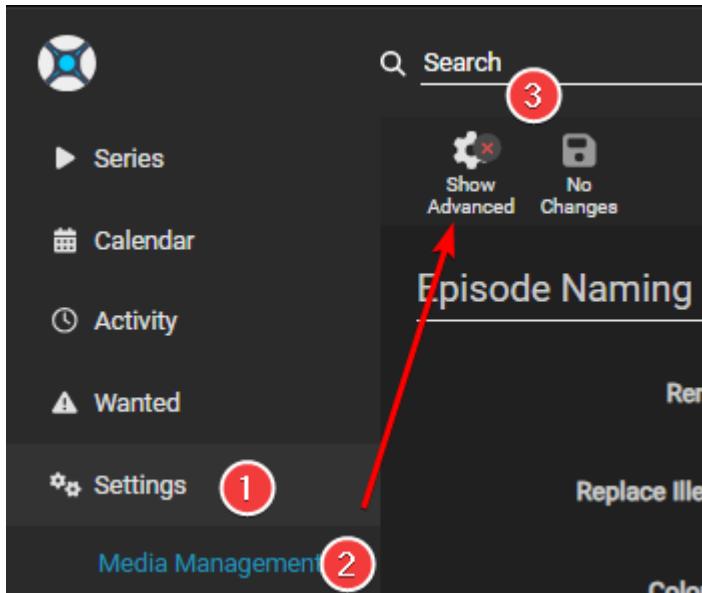


4. Select “Media Management”, scroll down and click on “Add Root Folder”. Type in “/Media/Media/TV Shows/” and select it before you click OK. If you have a separate folder for anime, add another root folder and search for “/Media/Media/Anime TV Shows” and select it. Click OK.

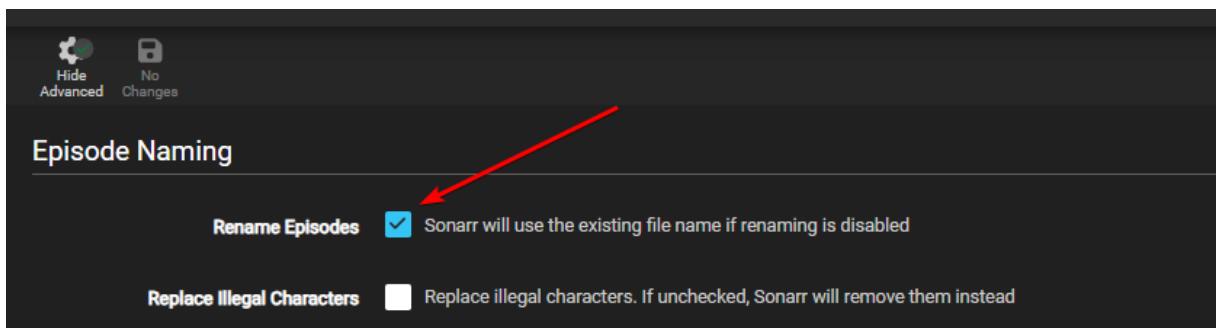


## Changing Naming Scheme(s)

1. Go to settings > Media Management and click on Show Advanced.



2. Enable “Rename Episodes”. I recommend to also enable “Replace illegal characters”



3. Go to TRaSH Guide’s and select the settings they recommend.

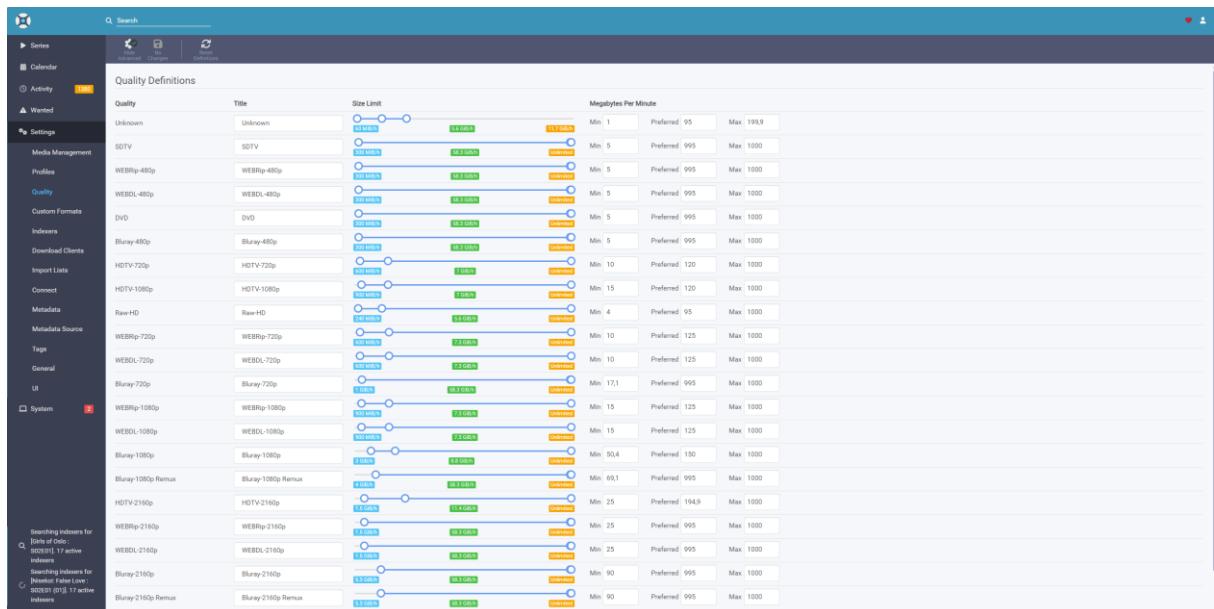
<https://trash-guides.info/Sonarr/Sonarr-recommended-naming-scheme/#standard>

4. Scroll down to “File Management” and select “Do not Prefer” for “Props and Repacks”. This is important for our quality profiles later.
5. Under “Importing” make sure “Use Hardlinks instead of Copy” is enabled.

## Quality Settings (File Size)

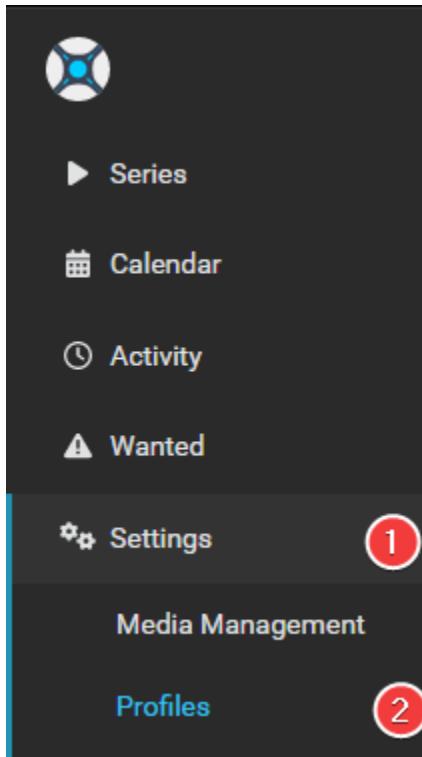
1. Go to settings and select “Quality”
2. Enabled “Show Advanced” by clicking the cog at the top.
3. Go to TRaSH Guides and edit the appropriate settings

<https://trash-guides.info/Sonarr/Sonarr-Quality-Settings-File-Size/>



## Quality Profiles

1. Go to settings and select “Profiles”

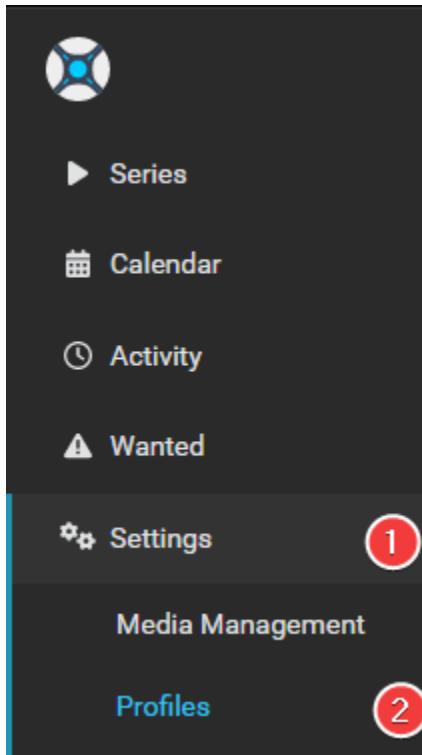


2. Delete all the default ones.
3. Go to TRaSH Guides and set up profiles as described there. Import the relevant custom formats that fits your needs and wants.

<https://trash-guides.info/Sonarr/sonarr-setup-quality-profiles/>

## Quality Profiles (Anime)

1. Go to settings and select “Profiles”



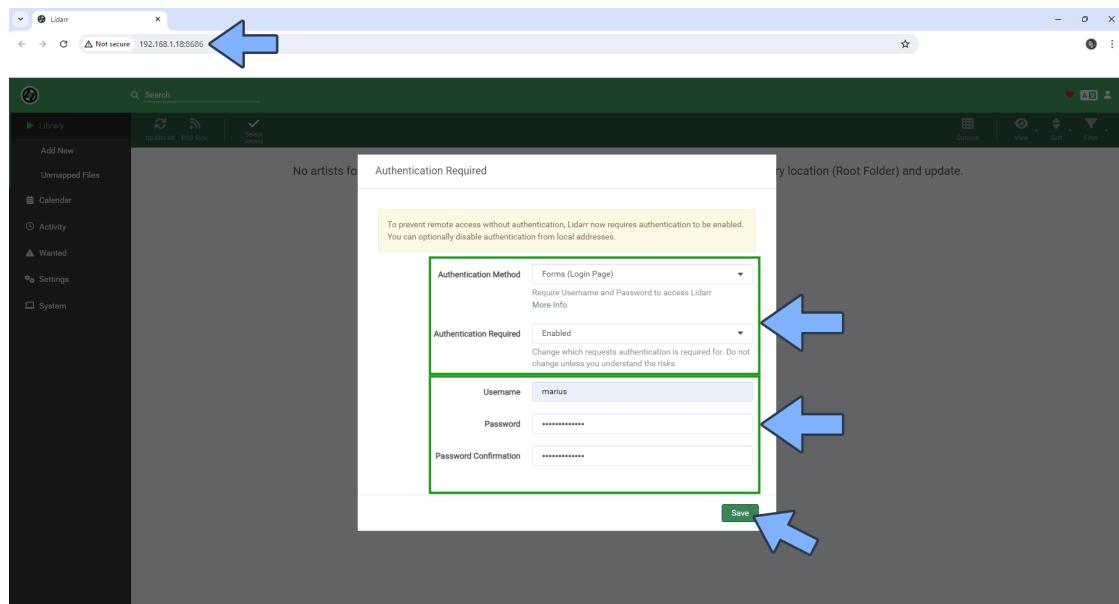
2. Delete all the default ones.
3. Go to TRaSH Guides and set up profiles as described there. Import the relevant custom formats that fits your needs and wants.

<https://trash-guides.info/Sonarr/sonarr-setup-quality-profiles-anime/>

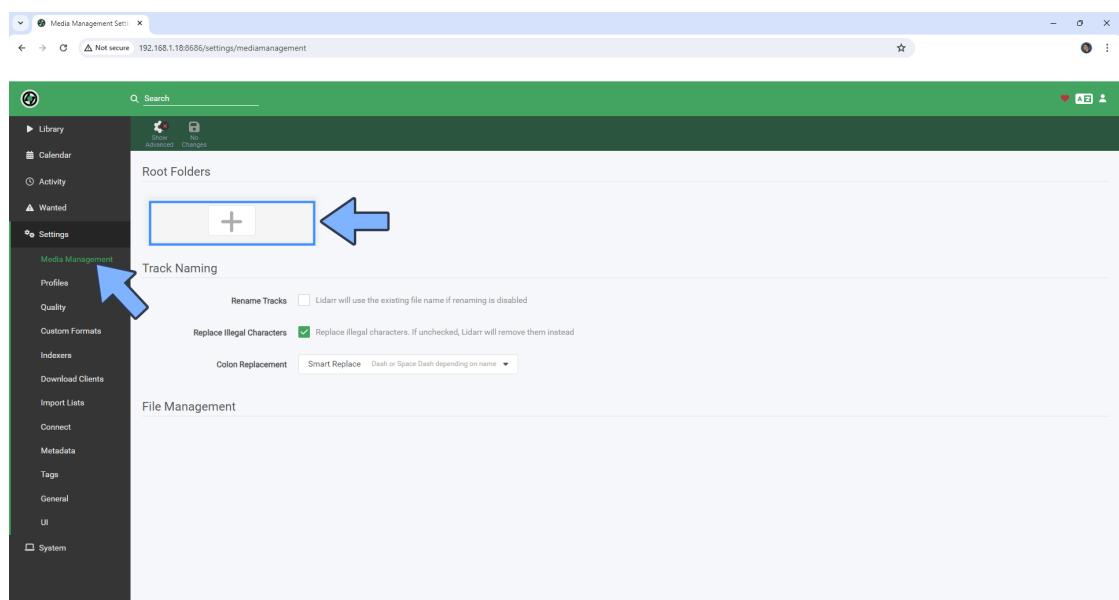
# Lidarr

## Adding Root Folder(s)

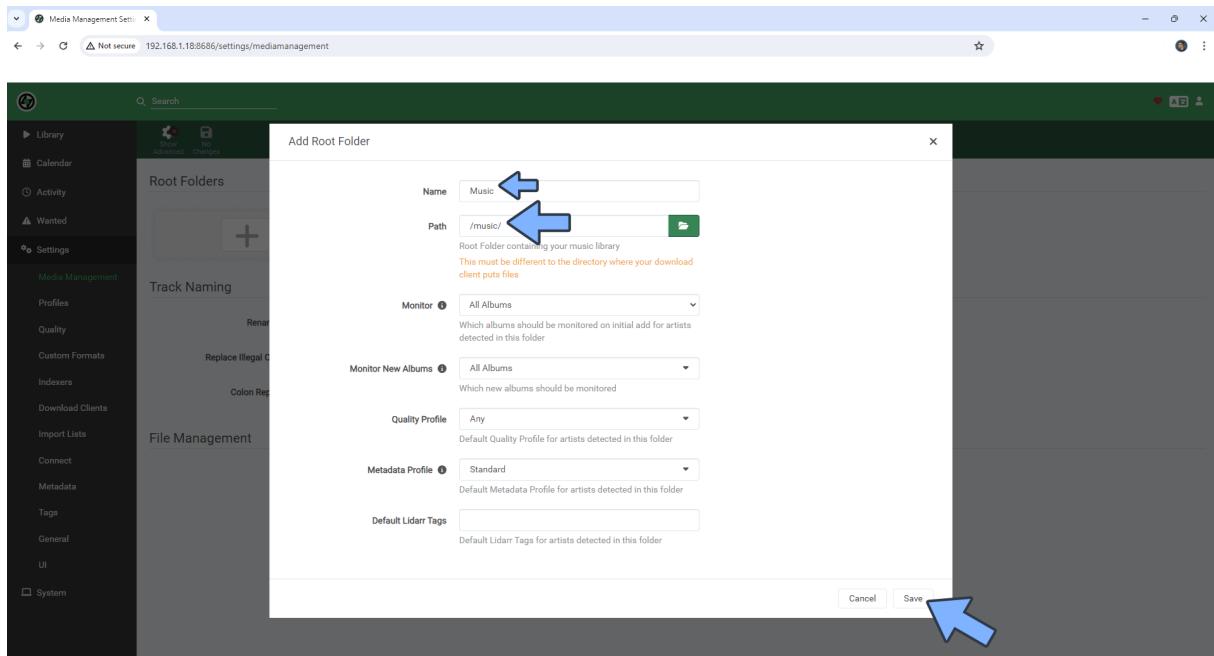
1. Open a web browser on your PC and type in the IP of your NAS followed by port 8686. So e.g. 192.168.0.2:8686.
2. Here you will be prompted to set up an authentication method. I recommend selecting “Forms” and having it enabled. Choose a username that is not admin or administrator, then select a password.



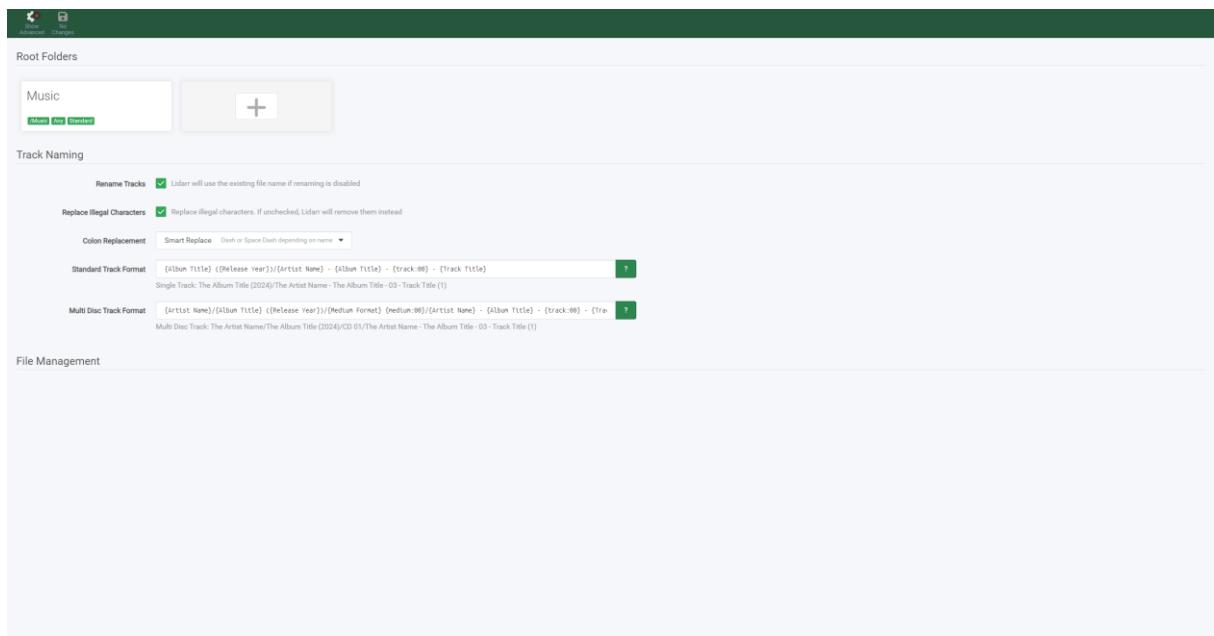
3. Go to settings, then select “Media Management”.
4. Click on the big “+” sign under “Add Root Folder”



5. Search for “/Media/Media/Music” select it then click on OK



6. Make sure “Rename Tracks” and “Replace illegal characters” is enabled.



“Standard Track Format” should be:

{Album Title} ({Release Year})/{Artist Name} - {Album Title} - {track:00} - {Track Title}

And “Multi Track Format” should be:

{Artist Name}/{Album Title} ({Release Year})/{Medium Format} {medium:00}/{Artist Name} - {Album Title} - {track:00} - {Track Title}

## Quality Settings

TRaSH Guides have no guides on lidarr. I have tried to search the internet but have not found any guides on quality settings or profiles for lidarr. I have only set up one profile myself on Lidarr, which is a pretty basic one without any scoring system with custom formats. Neither have I changed the size for the different Quality Settings.

The screenshot shows the 'Edit Quality Profile' dialog in Lidarr. The profile is named 'Standard'. Under 'Upgrades Allowed', there is a checked checkbox for 'If disabled qualities will not be upgraded'. The 'Upgrade Until' dropdown is set to 'Lossless', with a note below stating: 'Once this quality is reached Lidarr will no longer download albums'. Below this, there is a link to 'Want more control over which downloads are preferred? Add a Custom Format'. The main area displays a list of quality settings categorized by preference level:

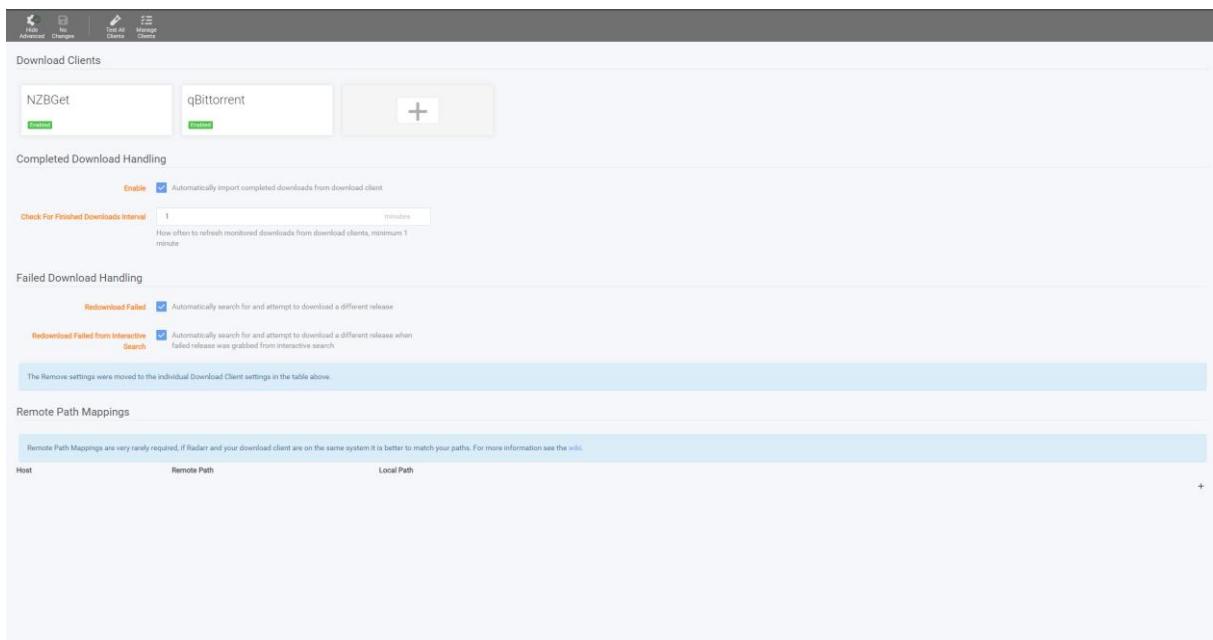
- Qualities:** Qualities higher in the list are more preferred even if not checked. Qualities within the same group are equal. Only checked qualities are wanted.
- Lossless:** ALAC 24bit, FLAC 24bit, WavPack, APE, ALAC, FLAC
- High Quality Lossy:** OGG Vorbis Q10, AAC-320, OGG Vorbis Q9, MP3-320, AAC-VBR, MP3-VBR-V0
- Mid Quality Lossy:** AAC-256, OGG Vorbis Q8, MP3-256, MP3-VBR-V2, OGG Vorbis Q7
- Low Quality Lossy:** MP3-224, WMA, AAC-192, OGG Vorbis Q6, MP3-192
- Poor Quality Lossy:** MP3-160, OGG Vorbis Q5, MP3-128, MP3-112, MP3-96
- WAV:**
- Trash Quality Lossy:** MP3-80, MP3-64, MP3-56, MP3-48, MP3-40, MP3-32, MP3-24, MP3-16, MP3-8
- Unknown:**

At the bottom left is a red 'Delete' button, and at the bottom right are 'Cancel' and 'Save' buttons.

## Connecting download client to Radarr, Sonarr and Lidarr

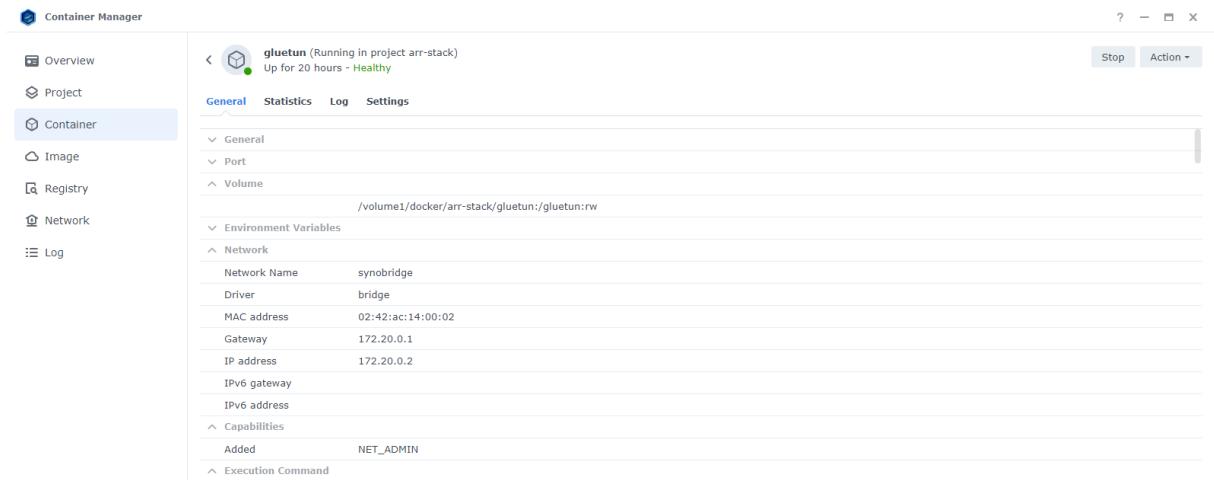
Now that we have configured all of our main arr-apps, we need to actually give them something to send the downloads to. For this guide we are using qBitTorrent. The setup will be identical for all of the apps, so just repeat the steps for all of them.

1. Go to settings, then select “Download Clients”
2. Make sure “Automatically import completed downloads from download client” is enabled.
3. Click on the big “+” icon.



4. Select qBitTorrent
5. Fill out the details
  - i Name: qBitTorrent
  - ii Enable
  - iii Host: 170.20.0.2

**NOTE:** This must be the IP of the **GlueTUN** container, and **NOT** the NAS itself. So instead of 192.168.0.2, we but 172.20.0.2. To confirm this is the correct IP, we can go back to Synology container manager > container and click on gluetun. Scroll down until you find “Network Settings” and look for where it says “IP Address”



iv Port: 8085

v Username: <username on qBitTorrent>

vi Password: <password on qBitTorrent>

vii Category: <the appropriate category. music/movies/tv/anime>

**NOTE:** For Sonarr, you will need to setup 2 download clients if you want to separate anime to a separate folder then tv. Both can be the same client, with identical setup. Only thing that has to change is the “Category” must be “anime” for anime and “tv” or “series” for normal TV Series.

6. Click “Test”. If it becomes green for a second, then press “Add” If it becomes red, review and double check your settings.

## Edit Download Client - qBittorrent

X

Name

Enable

Host

Port

Use SSL  Use a secure connection. See Options -> Web UI -> 'Use HTTPS instead of HTTP' in qBittorrent.

URL Base

Adds a prefix to the qBittorrent url, such as http://[host]:  
[port]/[urlBase]/api

Username

Password

Category

Adding a category specific to Radarr avoids conflicts with unrelated non-Radarr downloads. Using a category is optional, but strongly recommended.

Post-Import Category

Category for Radarr to set after it has imported the download. Radarr will not remove torrents in that category even if seeding finished. Leave blank to keep same category.

Recent Priority

Priority to use when grabbing movies that aired within the last 21 days

Older Priority

Priority to use when grabbing movies that aired over 21 days ago

Initial State

Initial state for torrents added to qBittorrent. Note that

[Delete](#)



[Test](#)

[Cancel](#)

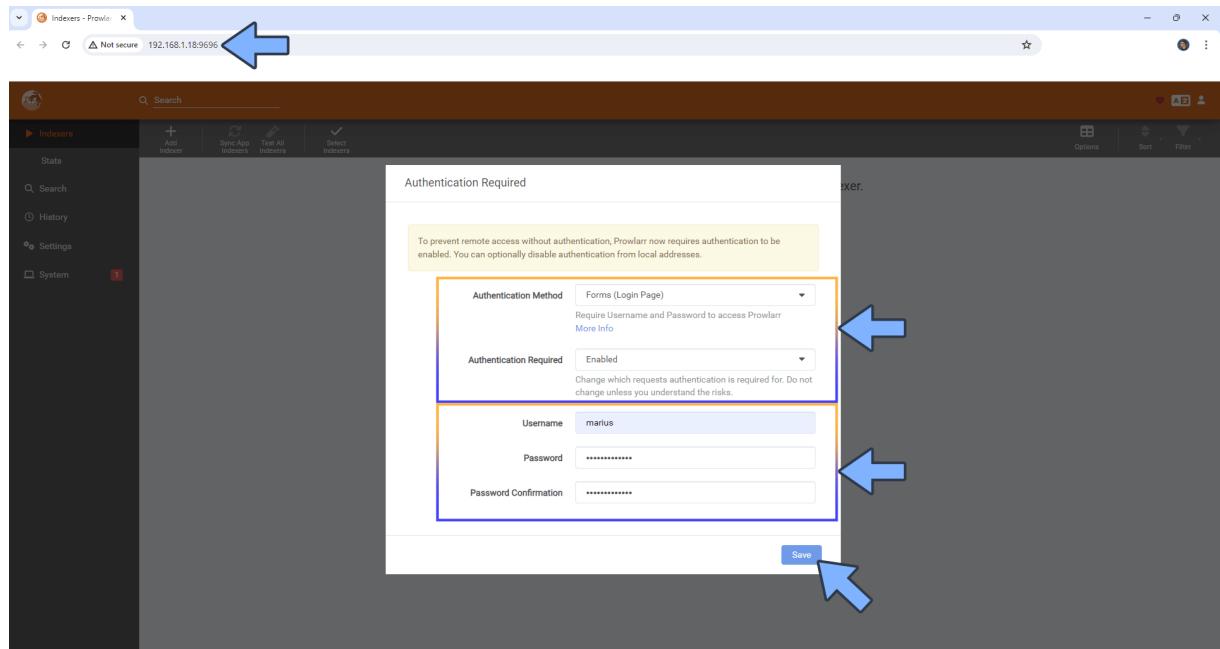
[Save](#)

# Prowlarr

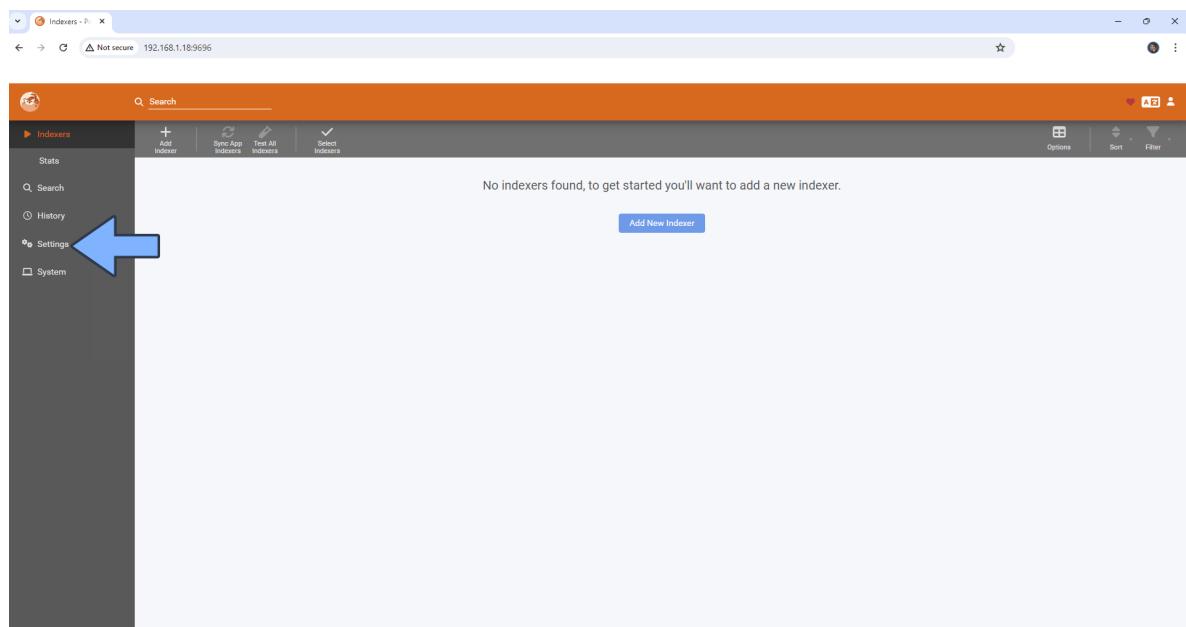
## Adding the apps to Prowlarr

Now that we have set up all our downloaders, we need something to actually search for the files we are going to download. For that, we will use Prowlarr.

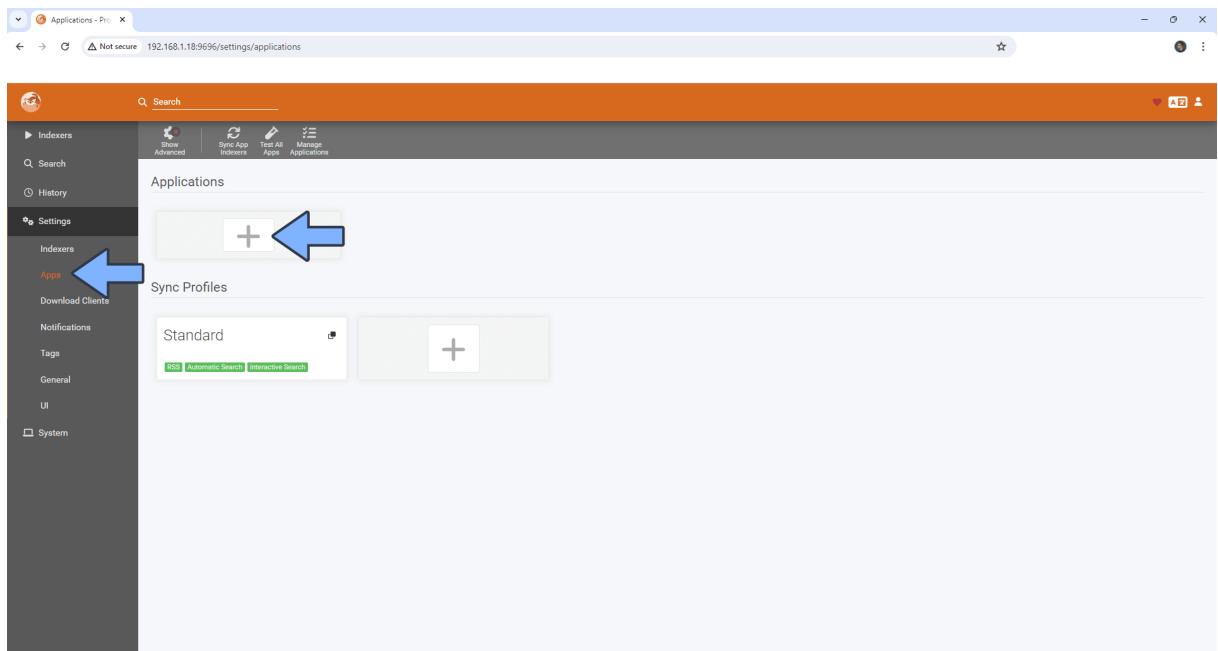
1. Go to <NAS-IP>:9696 e.g. 192.168.0.2:9696
2. Fill out the form just like earlier.



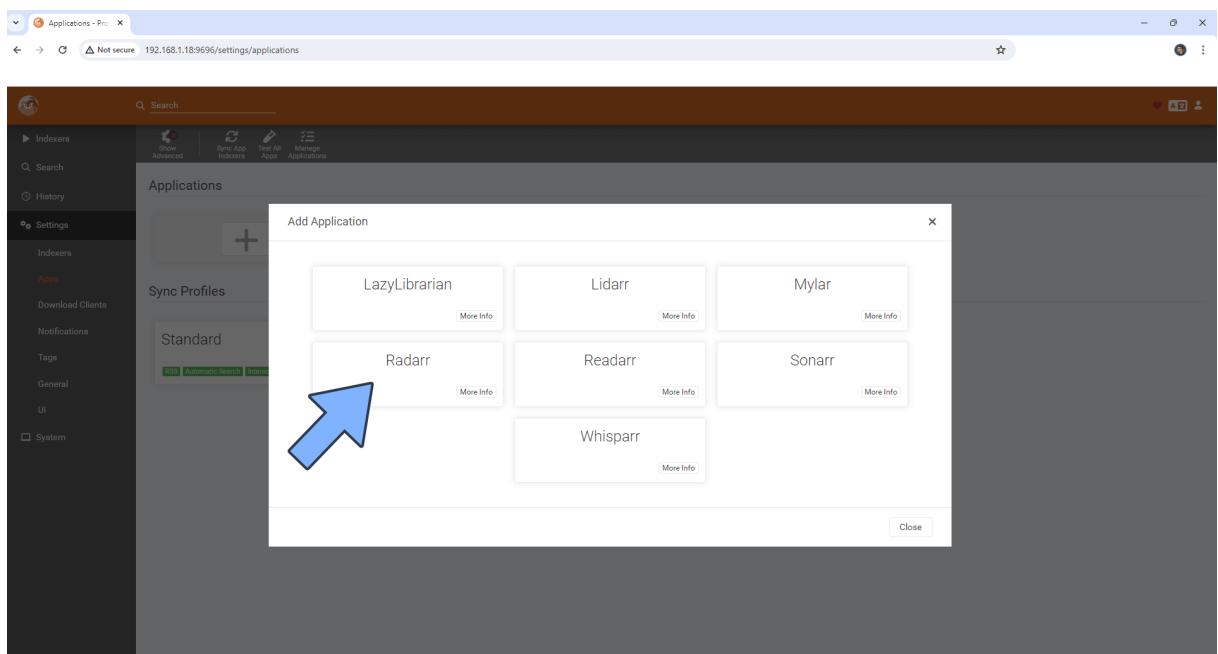
3. Go to settings on the left-hand side menu.



4. Click on apps, then the big “+” under applications



5. Click on the app you want. We can start with Radarr.



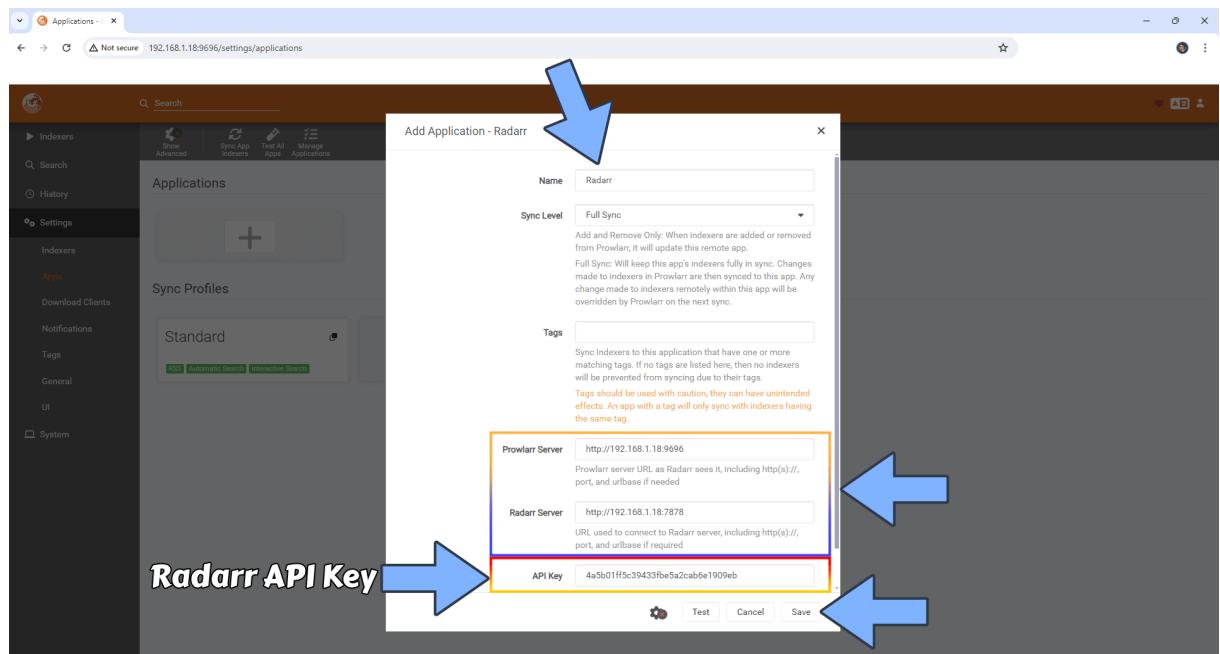
6. Fill out the form:

- Name can be default.
- Sync Level: Full Sync
- Tags: Leave blank
- Prowlarr server: <http://172.20.0.2:9696>
- Radarr server: <http://172.20.0.2:7878>

**NOTE:** If you chose another IP for your “synobridge” network, then put that in instead.

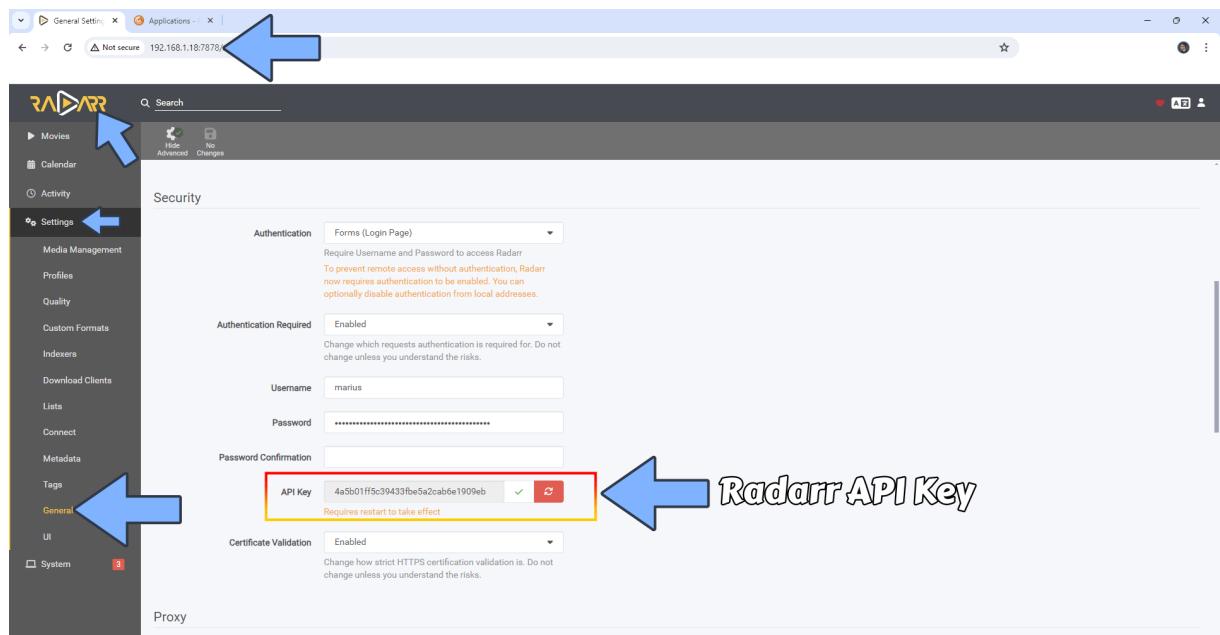
- API Key: Get this from the relevant app. In this case, Radarr. Scroll down to find out how.

7. Repeat this step for all the apps you want; LazyLibrarian, Lidarr, Mylar, Radarr, Readarr, Sonarr and Whisparr.



## How do I get the API key?

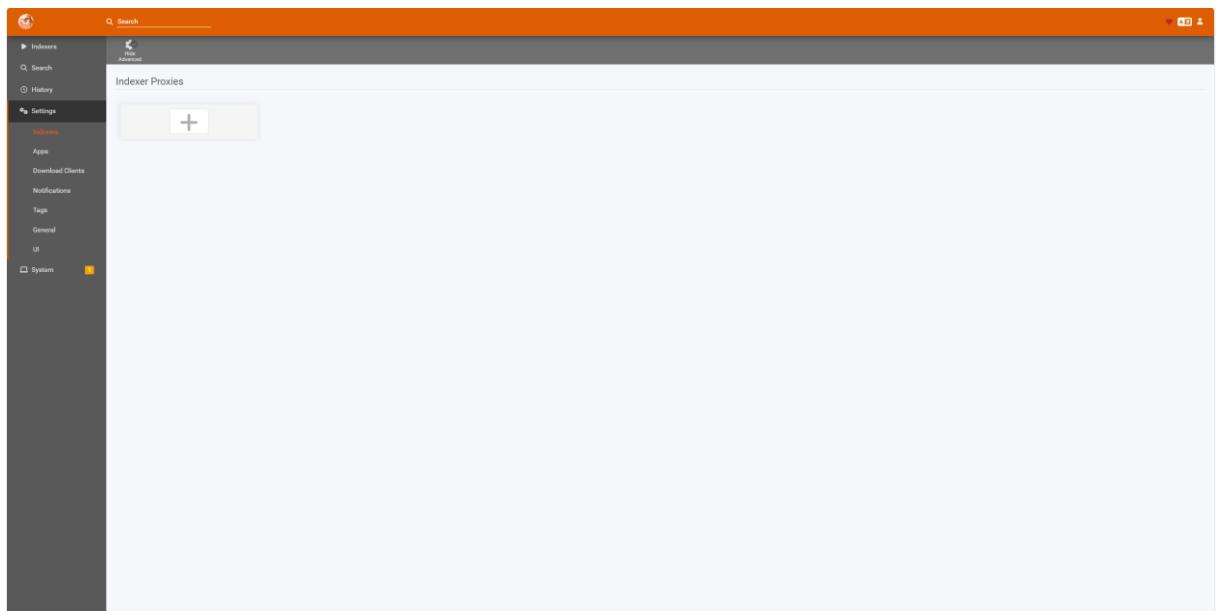
1. Open your relevant application in the browser.
2. Go to settings, then click on “General”
3. The API Key should be right there. Just click the copy button on the right, and paste that into Prowlarr.



## Connecting Flaresolverr

Some indexers, like 1337x which is one of the best free ones, require you to solve a Cloudflare verification before you get access. To make prowlarr do this, you have to use flaresolverr. To do this:

1. Go to settings, then click on indexers
2. Click the big + icon.



3. Edit the “Host” field to be “http://172.20.0.2:8191”

---

Add Indexer Proxy - FlareSolverr X

Name

Tags   
Applies to indexers with at least one matching tag

Host

Request Timeout  seconds  
FlareSolverr maxTimeout Request Parameter

---

⚙️ Test Cancel Save

4. Click on “Test” and if it becomes green for a second it works, and you can click “Save”. If it becomes red, then double check your Host IP. It should be the same as your “synobridge” network in docker.



## Adding indexers

Indexers are the websites prowlarr search for files. Sadly, a lot of the good ones are private. This means that to use them you will most likely need to pay to get access, so I will focus on the free ones.

1. Click on Indexers, then “Add Indexer”



2. Here can filter by protocol, Language, Privacy and Category. All indexers that use the “nzb” protocol are private. So you could just set the privacy to public to get a list of all the free ones. The 2 ones I would recommend to get at least are 1337x and TheRARBG for general TV and movies, and Nyaa for anime.
3. We can now search for our desired indexers, then click on the result
4. Fill out the settings. Base URL should be 1337x.to and I would also recommend to turn non advanced settings, then choose “minimum seeders: 1”.

Add Indexer - Cardigan (1337x) x

---

Name	<input type="text" value="1337x"/>
Enable	<input checked="" type="checkbox"/>
Redirect	<input type="checkbox"/> Redirect incoming download request for indexer and pass the grab directly instead of proxying the request via Prowlarr
Sync Profile	<input type="button" value="Standard"/> App profiles are used to control RSS, Automatic Search and Interactive Search settings on application sync
Base Url	<input type="text" value="https://1337x.to/"/> Select which base url Prowlarr will use for requests to the site
Query Limit	<input type="text"/>
The number of max queries as specified by the respective unit that Prowlarr will allow to the site	
Grab Limit	<input type="text"/>
The number of max grabs as specified by the respective unit that Prowlarr will allow to the site	
Limits Unit	<input type="button" value="Day"/> (0) <span style="font-size: small;">The unit of time for counting limits per indexer</span>
Apps Minimum Seeders	<input type="text" value="1"/> Minimum seeders required by the Applications for the indexer to grab, empty is Sync profile's default
Seed Ratio	<input type="text"/>
The ratio a torrent should reach before stopping, empty uses the download client's default. Ratio should be at least 1.0 and follow the indexers rules	
Seed Time	<input type="text"/> minutes <span style="font-size: small;">The time a torrent should be seeded before stopping, empty uses the download client's default</span>

---

⚙️ Test Cancel Save

If we scroll down, we can see that it says it requires flaresolverr. Now we can click “Test” to see if it works, and if it does we can click “Save”. Do this for all your wanted indexers.

Add Indexer - Cardigann (1337x)

follow the indexers rules

**Seed Time** minutes  
The time a torrent should be seeded before stopping, empty uses the download client's default

**Pack Seed Time** minutes  
The time a pack (season or discography) torrent should be seeded before stopping, empty is app's default

**FlareSolverr Info**  
This site may use Cloudflare DDoS Protection, therefore Prowlarr requires FlareSolverr to access it.

**Download link** iTorrents.org

**Download link (fallback)** magnet

**About the Download links**  
As the iTorrents .torrent download link on this site is known to fail from time to time, we suggest using the magnet link as a fallback. The BTCache and Torrage services are not supported because they require additional user interaction (a captcha for BTCache and a download button on Torrage.)

**Sort requested from site** created

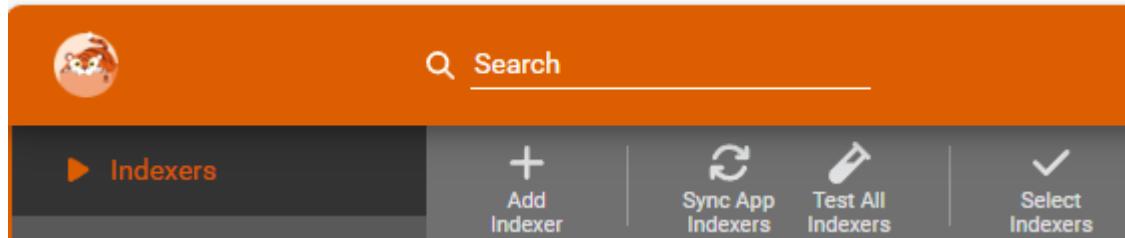
**Order requested from site** desc

**Indexer Priority** 25  
Indexer Priority from 1 (Highest) to 50 (Lowest). Default: 25.

**Tags**  
Use tags to specify Indexer Proxies or which apps the indexer is synced to.  
Tags should be used with caution, they can have unintended effects. An indexer with a tag will only sync to apps with the same tag.

Test Cancel Save

5. After you have added all your desired indexers, click on “Sync App Indexers” to push them to all your apps.



6. Let's wait 2-5 minutes for it to finish up, then we can head into our apps, like Radarr or Sonarr, and click on settings > indexers to see our selected indexers with a (prowlarr) at the end to indicate where it comes from.

Nyaa.si (Prowlarr)

RSS Automatic Search Interactive Search

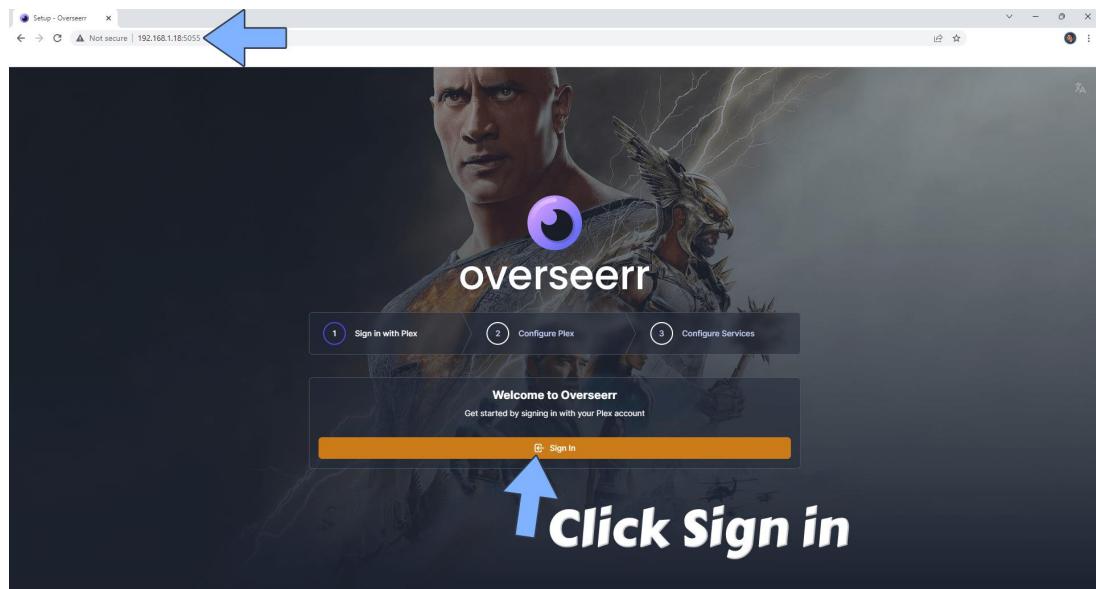
Priority: 50

## Overseerr

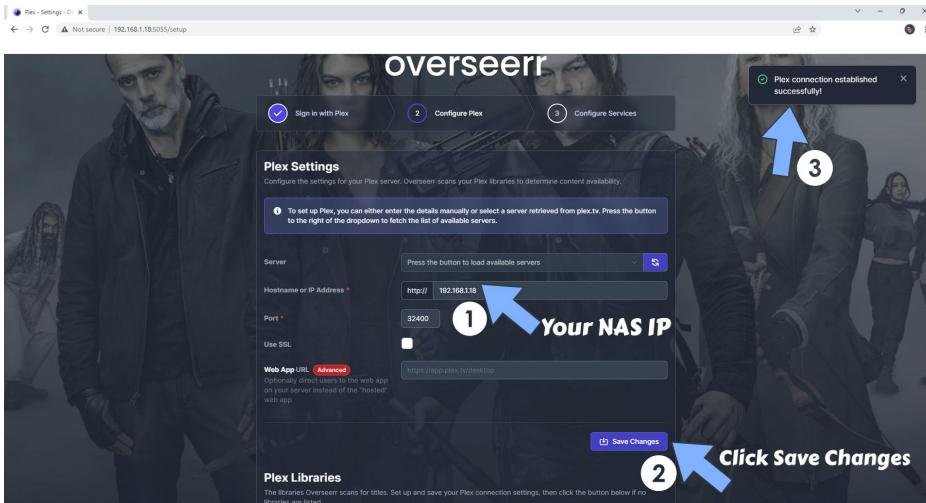
Now it's time to take the automation to the next level. With overseer, we get a huge catalogue of movies and shows at our fingertips, and can download them with just one click.

### Configuring Overseerr

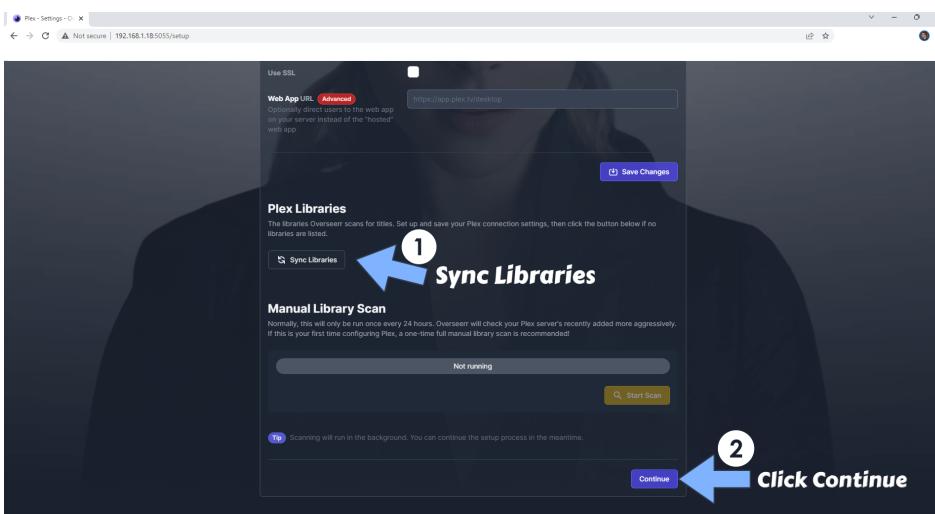
1. Go to <synolog-ip>:5055.
2. Click "Sign In", and follow the sign in instructions



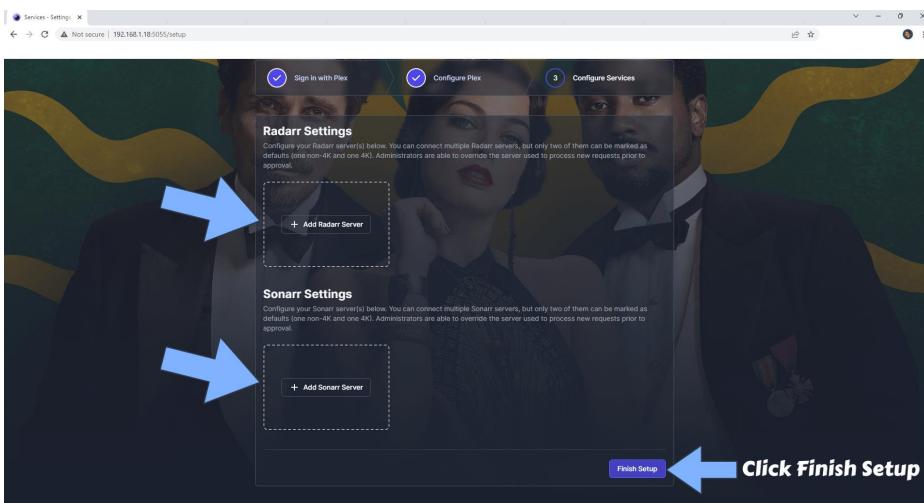
3. On the setup screen, **DON'T** use your NAS IP, but your synobridge IP. So for me it's 172.20.0.2. If you haven't changed the plex port, it can stay as default. Click "Save Changes". You should now see a green checkmark at the top.



4. Scroll down a bit and click “Sync Libraries”. Afterwards you can click continue. If it’s not done syncing, it will continue in the background.



5. Add your Radarr and Sonarr by clicking the buttons as shown in the image below. Scroll down to continue.



## Adding Radarr and Sonarr

On the first-time setup, you will get a prompt to add Radarr and Sonarr. If you manage to miss it, fret not as you can do it in settings later. Just head to settings > services then click on add.

The screenshot shows the overseerr application interface. On the left is a sidebar with icons for Discover, Movies, Series, Requests, Issues, Users, and Settings. The Settings icon is highlighted with a purple background. At the top right is a search bar labeled "Search Movies & TV". Below the search bar is a navigation menu with tabs: General, Users, Plex, Services (which is selected and highlighted in blue), Notifications, Logs, Jobs & Cache, and About. The main content area is divided into two sections: "Radarr Settings" and "Sonarr Settings".

**Radarr Settings**  
Configure your Radarr server(s) below. You can connect multiple Radarr servers, but only two of them can be marked as defaults (one non-4K and one 4K). Administrators are able to override the server used to process new requests prior to approval.

radarr	Default
Address	http://172.20.0.2:7878
Active Profile	HD Blu-ray + WEB

[Edit](#) [Delete](#)

[+ Add Radarr Server](#)

**Sonarr Settings**  
Configure your Sonarr server(s) below. You can connect multiple Sonarr servers, but only two of them can be marked as defaults (one non-4K and one 4K). Administrators are able to override the server used to process new requests prior to approval.

sonarr	Default
Address	http://172.20.0.2:8989
Active Profile	Blu-ray and WEB-DL ...

[Edit](#) [Delete](#)

[+ Add Sonarr Server](#)

## Radarr

1. When you click on “Add Radarr Server” you should see something like this:

The screenshot shows the 'Add New Radarr Server' form. It includes fields for 'Default Server' (checkbox), '4K Server' (checkbox), 'Server Name' (text input: 'radarr'), 'Hostname or IP Address' (text input: 'http:// 170.20.0.2'), 'Port' (text input: '7878'), 'Use SSL' (checkbox), 'API Key' (text input with a copy icon), 'URL Base' (text input), 'Quality Profile' (dropdown: 'Test connection to load quality profiles'), 'Root Folder' (dropdown: 'Test connection to load root folders'), 'Minimum Availability' (dropdown: 'Released'), 'Tags' (text input with a dropdown arrow), 'External URL' (text input), 'Enable Scan' (checkbox), 'Enable Automatic Search' (checkbox checked), 'Tag Requests' (checkbox), and a note about automatically adding a tag with user ID and display name. At the bottom are 'Cancel', 'Test' (yellow button), and 'Add Server' buttons.

2. Fill out server name, hostname, port and API Key as follows:
  - Server Name radar (or anything you like)
  - Hostname: 170.20.0.2 (your synobridge IP)
  - Port: Default. 7878 for Radar
  - API Key: Your relevant API Key. Don't know how to get it? Check the table of contents.
3. Now click on “Test”. If it’s successful, you will now get to select the rest of the settings.
4. Now configure the rest of the settings:
  - Quality Profile: Your desired default quality profile. This can be changed every time you request in Overseer, but not in Requestr
  - Root folder: “/Media/Media/Movies”
  - Minimum availability: Whatever you prefer. I want it to be released.
  - Tick the options for “Enable scan” and “Enable Automatic Search”

## Edit Radarr Server

Default Server

4K Server

Server Name \*

Hostname or IP Address \*

Port \*

Use SSL

API Key \*  

URL Base

Quality Profile \*

Root Folder \*

Minimum Availability \*

Tags

External URL

Enable Scan

Enable Automatic Search

Tag Requests   
Automatically add an additional tag with the requester's user ID & display name

5. Now we can save our changes.

## Sonarr

1. After you click “Add Sonarr”, you should see something like this:

**Add New Sonarr Server**

Default Server

4K Server

Server Name \*

Hostname or IP Address \*  http://

Port \*  8989

Use SSL

API Key \*  

URL Base

Quality Profile \*  Test connection to load quality profiles

Root Folder \*  Test connection to load root folders

Language Profile \*  Test connection to load language profiles

Tags  Test connection to load tags

Anime Quality Profile  Test connection to load quality profiles

Anime Root Folder  Test connection to load root folders

Anime Language Profile  Test connection to load language profiles

Anime Tags  Test connection to load tags

Season Folders

External URL

Enable Scan

Enable Automatic Search

Tag Requests   
Automatically add an additional tag with the requester's user ID & display name

2. Fill out the settings like this:

- Servername: sonar (or anything you like)
- Hostname: 170.20.0.2 (Your synobridge IP)
- Port: Default. 8989 for Sonarr
- API Key: Your API Key. Don't know where to get it? Check the table of contents.

3. Now we can click "Test" to test out connection. If it's successful, we can now move on to edit some more settings.

**Add New Sonarr Server**

Default Server

4K Server

Server Name \*

Hostname or IP Address \*

Port \*

Use SSL

API Key \*  

URL Base

Quality Profile \*

Root Folder \*

Language Profile \*

Tags

Anime Quality Profile

Anime Root Folder

Anime Language Profile

Anime Tags

Season Folders

External URL

Enable Scan

Enable Automatic Search

Tag Requests   
Automatically add an additional tag with the requester's user ID & display name

4. Now make these changes to the settings:

- Quality Profile: Your desired default quality profile. This can be changed every time you request in Overseer, but not in Requestr
- Root Folder: “/Media/Media/TV Shows”
- Language Profile: Deprecated
- Tags: series
- Anime Quality profile: Your desired default quality profile for anime shows. This can be changed every time you request in Overseer, but not in Requestr
- Anime Root Folder: “/Media/Media/Anime TV Shows”
- Language Profile: Deprecated
- Anime Tags: anime
- Be sure to tick the boxes for “Enable Scan” and “Enable Automatic Search”

5. Click “Save”

**Edit Sonarr Server**

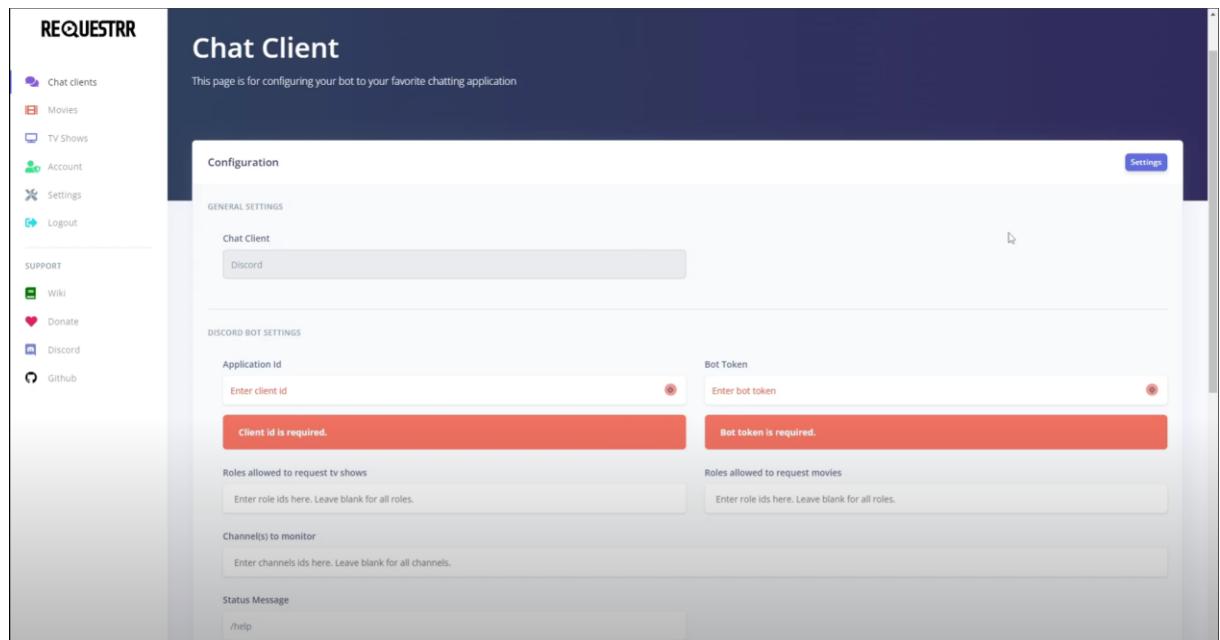
Default Server	<input checked="" type="checkbox"/>
4K Server	<input type="checkbox"/>
Server Name *	sonarr
Hostname or IP Address *	http:// 172.20.0.2
Port *	8989
Use SSL	<input type="checkbox"/>
API Key *	..... <input type="button" value=""/>
URL Base	
Quality Profile *	Blueray and WEB-DL (1080p)
Root Folder *	/Media/Media/TV Shows
Language Profile *	Deprecated
Tags	series <input type="button" value="x"/> <input type="button" value=" "/> <input type="button" value="▼"/>
Anime Quality Profile	Remux-1080p-Anime
Anime Root Folder	/Media/Media/Anime TV Shows
Anime Language Profile	Deprecated
Anime Tags	anime <input type="button" value="x"/> <input type="button" value=" "/> <input type="button" value="▼"/>
Season Folders	<input checked="" type="checkbox"/>
External URL	
Enable Scan	<input checked="" type="checkbox"/>
Enable Automatic Search	<input checked="" type="checkbox"/>
Tag Requests	Automatically add an additional tag with the requester's user ID & display name
<input type="button" value="Cancel"/> <input type="button" value="Test"/> <input type="button" value="Save Changes"/>	

## Requestrr

Now we can configure Requestrr. It allows us, and our users, to request movies and TV shows through chat in discord. This is a good alternative if you don't know how or don't want to port forward your Overseer. Or if you plan to share your Plex with strangers and don't want them to know your public IP-address

### Configuring Requestrr

1. Open Requestrr in a web browser by typing <NAS-IP>:4045
2. The first time you log in, you might get a pop-up to create an authentication method just like all the other apps. Fill out the form like earlier then.
3. When we open up Requestrr, we should see our Chat clients. If not, navigate to it on the left-hand side menu.



4. As you can see, we need an Application Id and Bot Token for Discord. Here is how you can get it:

### How to get Discord Application Id and Bot Token

1. Go to Discord Developer Portal by clicking this link:  
<https://discordapp.com/developers/applications>
2. Click on New Application

The screenshot shows the 'Applications' section of the Discord Developer Portal. On the left, there's a sidebar with 'DEVELOPER PORTAL' at the top, followed by 'Applications' (which is highlighted in blue), 'Teams', and 'Documentation'. The main area has a dark background with a central graphic of a computer monitor displaying a game. At the top right is a blue 'New Application' button with a red box around it. Below the button, text encourages creating a new application for the Game SDK and Server Commerce. There are sorting options ('Sort By: Date Created') and size selection buttons ('Small' and 'Large'). The section is titled 'My Applications' and includes a message: 'You don't have any applications yet, sad face.' with a link to 'Click New Application above to get started.'

3. Give it a name and a profile picture if you so wish. The important thing is to copy the Application Id.

The screenshot shows the 'General Information' step of creating a new application. It includes fields for 'APP ICON' (with a placeholder image and a 'Remove' button), 'NAME' (set to 'Rumbi'), and 'DESCRIPTION (MAXIMUM 400 CHARACTERS)' (with a note about appearing in the bot's profile). At the bottom, the 'APPLICATION ID' field contains the value '587482631235546784', which is also enclosed in a red box. Below this field is a blue 'Copy' button.

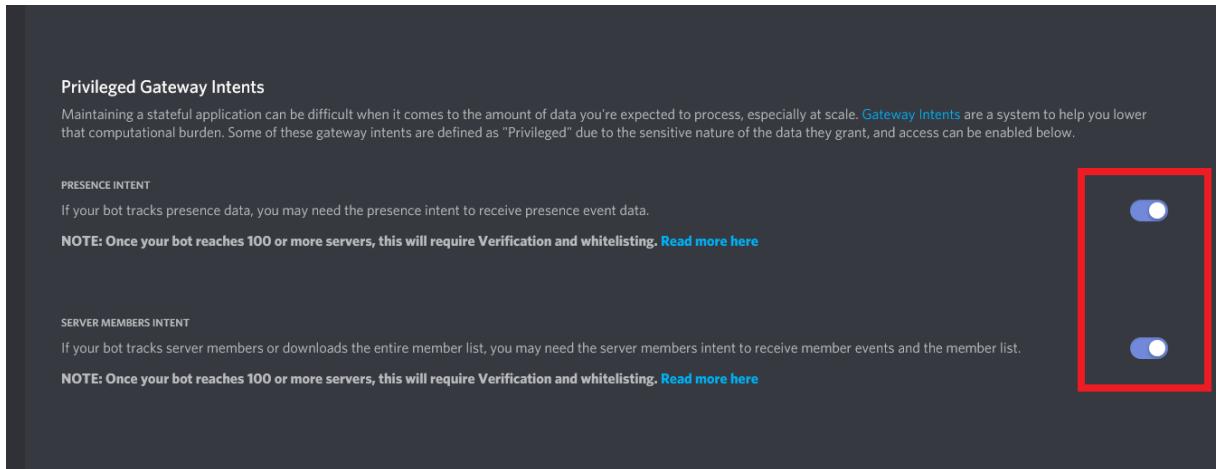
4. Paste it into Requestr

The screenshot shows the configuration interface for a 'Chat Client'. Under the 'DISCORD' section, there is a red box highlighting the 'Application Id' input field, which contains the placeholder 'Enter application id'. Below it, another red box highlights the 'Roles allowed to request tv shows' input field, which contains the placeholder 'Enter roles here. Leave blank for all roles.'.

5. Go back into Discord Developer Portal and create a bot for your newly created application.

The screenshot shows the 'Bot' section of the Discord Developer Portal for the 'Requestr' application. On the left, under 'SETTINGS', the 'Bot' option is highlighted with a red box. On the right, the 'Build-A-Bot' section is shown, featuring a large robot icon and a blue 'Add Bot' button, which is also highlighted with a red box.

6. Make sure these 2 settings are enabled:



## 7. Now copy the Bot Token

**Bot**

Bring your app to life on Discord with a Bot user. Be a part of chat in your users' servers.

[Learn more about bot users](#)

A wild bot has appeared!

**Build-A-Bot**

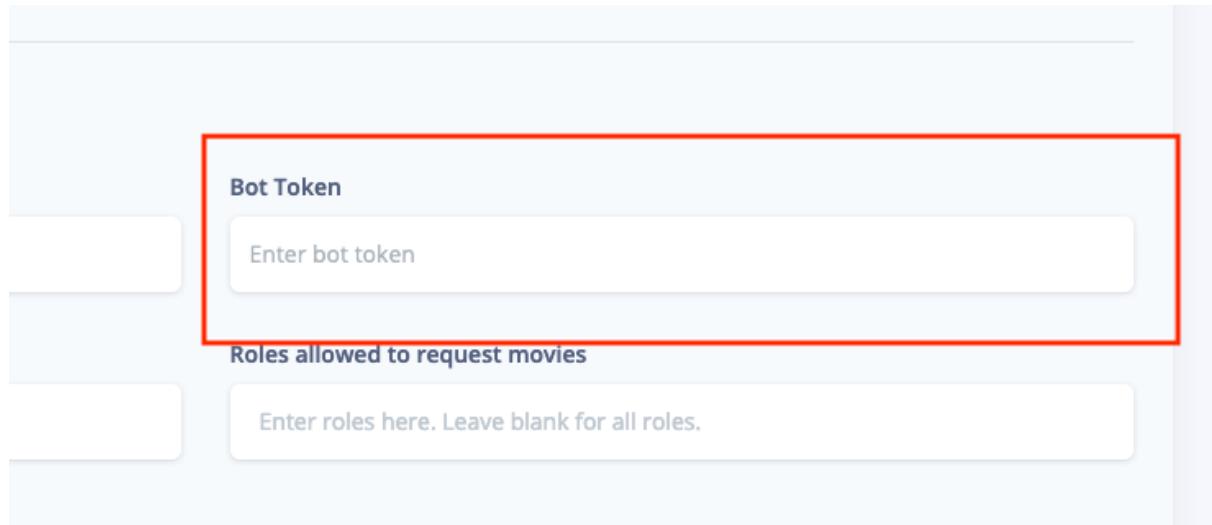
Bring your app to life by adding a bot user. This action is irreversible (because robots are too cool to delete).

ICON	USERNAME	TOKEN
	Requestrr	<a href="#">Click to Reveal Token</a> <button>Copy</button> <button>Regenerate</button>

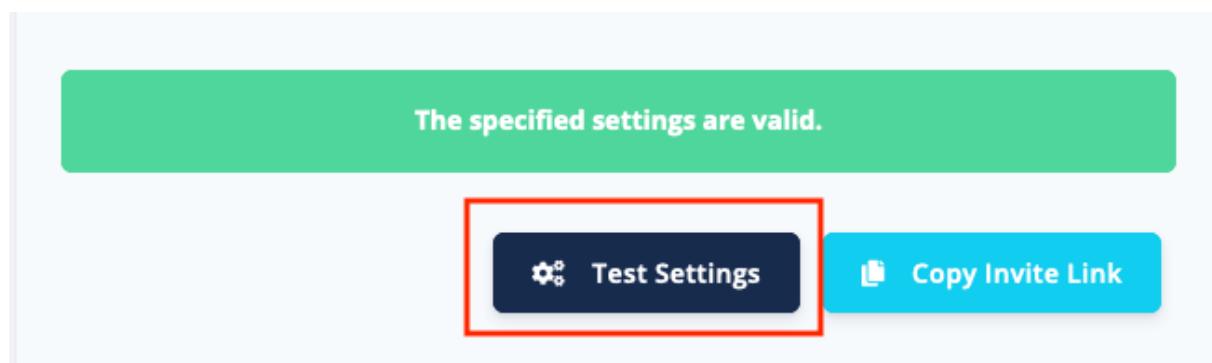
**Authorization Flow**

These settings control how OAuth2 authorizations are restricted for your bot (who can add your bot and what permissions they have).

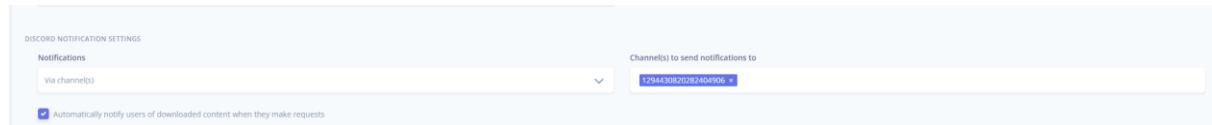
## 8. Paste the Bot Token into Requestrr



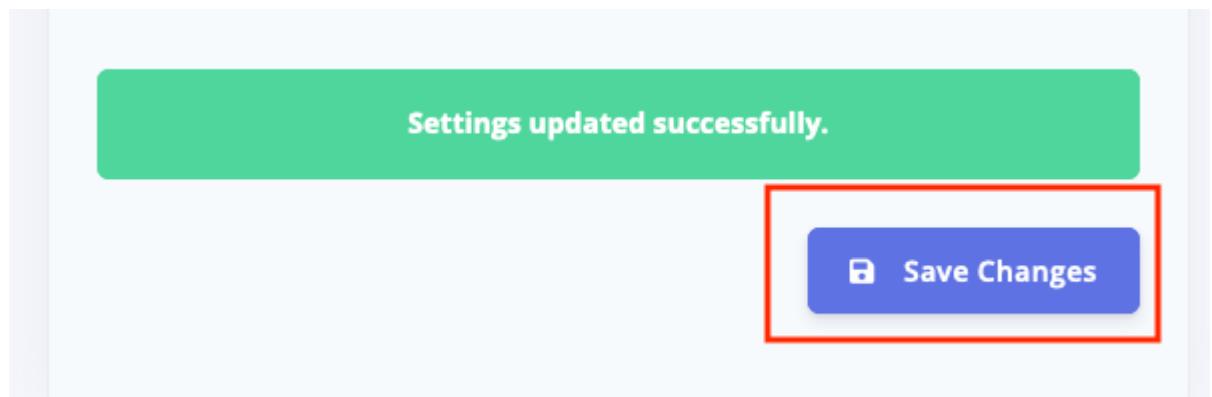
9. Click on “Test Settings” to check if the connection is good



10. Go to your discord server, right click on the channel you want your bot to be in and copy the channel ID. Paste it into “Channel(s) to send notifications to”

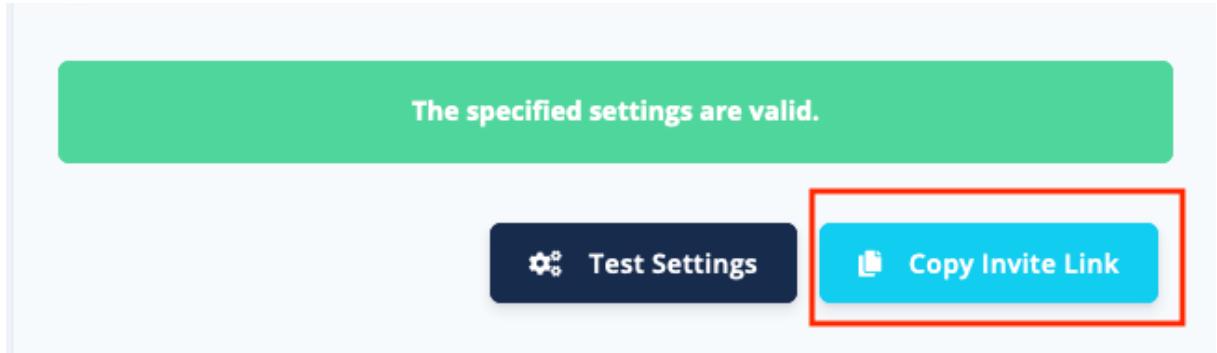


11. Finally, we can save our changes

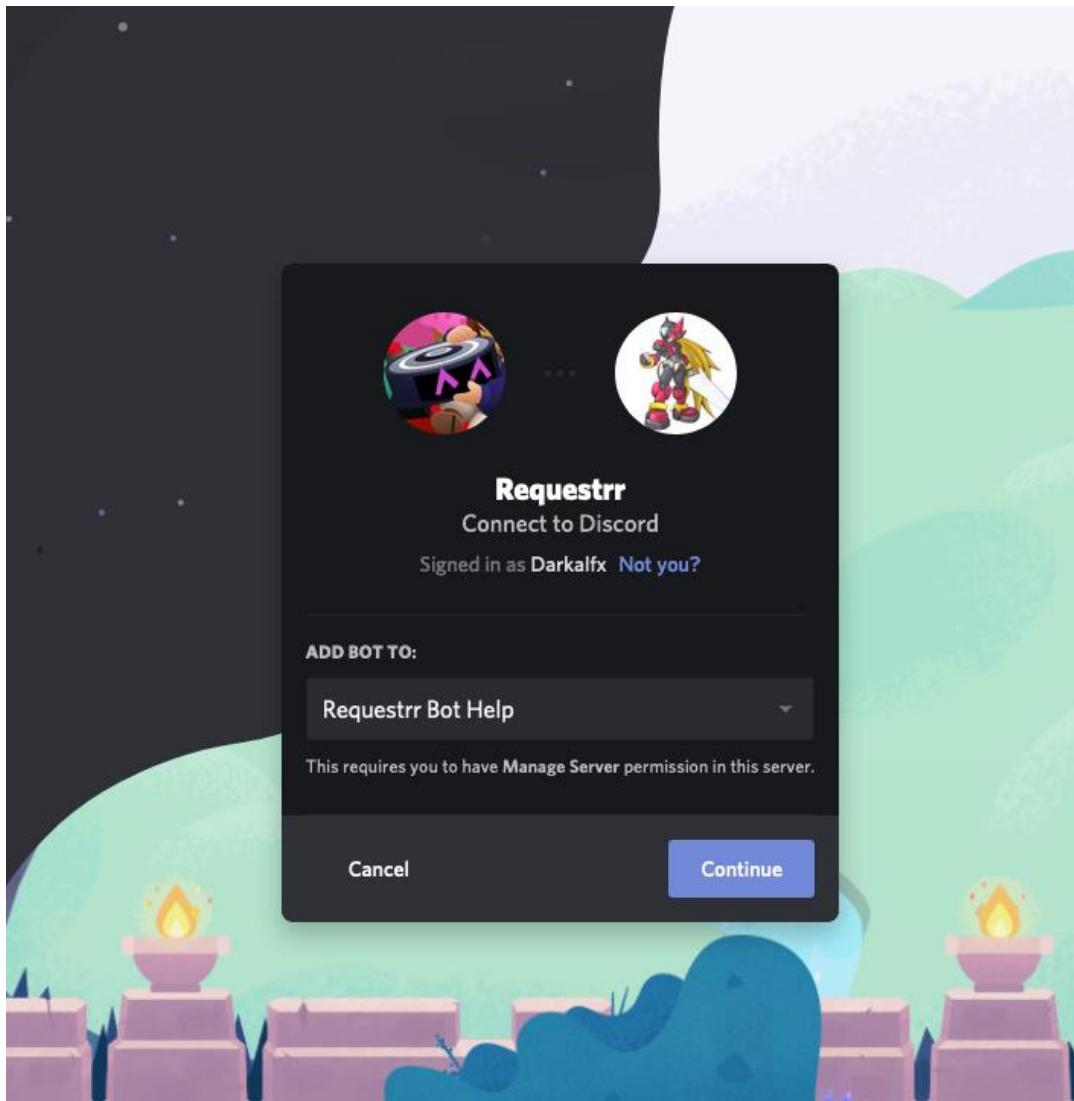


*How to invite the bot to our Discord server*

179. Be sure you have created a Discord server, or are admin in an existing one
180. Click “Copy Invite Link”



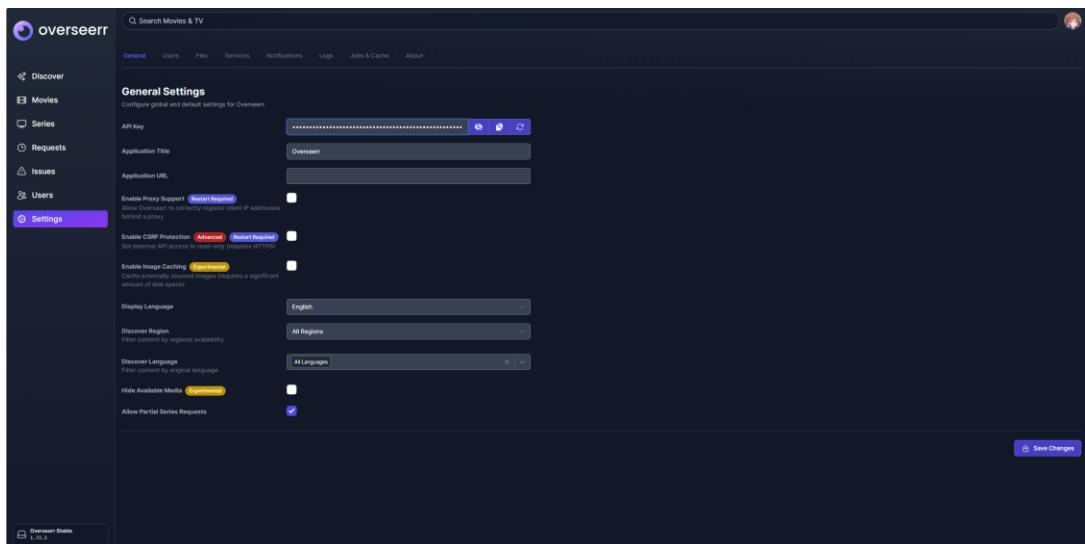
181. Choose the Server you whish to add the Bot to



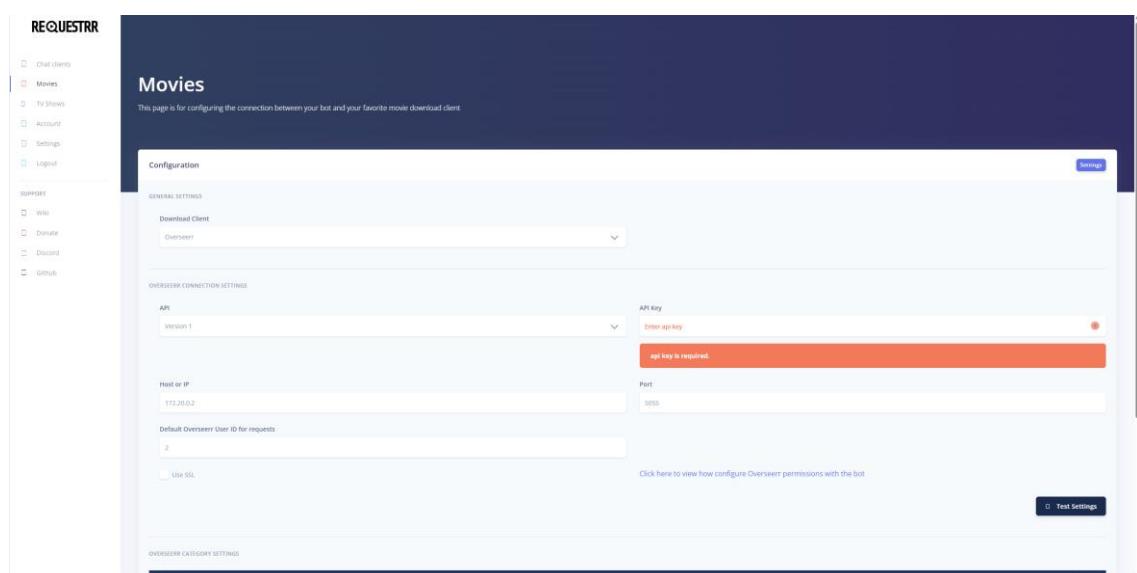
## Connecting Requestrr to Overseerr (Or Radarr, Sonarr, Ombi)

Now we have to set up Requestrr to actually, well, request. The best way would be to use Overseer or Ombi, but it also works directly with Radarr and Sonarr.

1. Go to Overseer on your web-browser. <NAS-IP>:5055
2. Go to settings (On the left-hand side menu) and make sure “Enable CSRF Protection” is disabled (unticked)
3. While you’re here, also copy your Overseer API key.



4. Go to Requestrr on your web-browser. <NAS-IP>:4045
5. Go to Movies (On the left-hand side menu)



6. Choose Overseer as Download Client. Paste in your API Key. Host should be 170.20.0.2 (your synobridge) and port 5055. Default Overseer user is which user the requests should come from. To find out your user ID to do as follows:
  - i Go back to overseer
  - ii Select “Users” from the menu
  - iii Then you should see a list of all your users. I made a custom Requestr user.



Click on the desired user. Now you should see your user id

**Joined October 12, 2024 | User ID: 2**

7. Click “Test Settings”, then scroll down and click on “Save Changes”
8. Now go to “TV Shows” from the menu, and do the exact same as you did for movies. Most of it should already be automatically filled out by your movies section when you select “Overseerr” as client.
9. Now it works! You can go to your server and type /help in the channel you copied the ID for earlier, and it will list all available commands.



## Sources

Marius Hosting have a lot of great guides on how to setup different things on Synology:

<https://mariushosting.com/>

DrFrankenstein's Tech Stuff also has a lot of useful information on a bunch of subjects within the synology ecosystem:

<https://drfrankenstein.co.uk/>

TRaSH Guides. The BEST source for configuring Radarr and Sonarr:

<https://trash-guides.info/>

Radarrs official docs:

<https://github.com/Radarr/Radarr>

Sonarrs official docs:

<https://github.com/sonarr/sonarr>

Lidarrs official docs:

<https://github.com/lidarr/lidarr>

Overseerrs official docs:

<https://github.com/sct/overseerr>

Requestrrs official docs:

<https://github.com/darkalfx/requestrr>

GlueTUNs official docs:

<https://github.com/qdm12/gluetun-wiki>

Awesome -arrs wiki:

<https://github.com/Ravencentric/awesome-arr>