

# Docker containers

A comprehensive installation guide of the docker containers used for greiner.live and greinet.com.

- [Random notes](#)
  - [Registry garbage collect](#)
  - [Multiarch builds](#)
- [DrawIO - A flowchart maker](#)
- [Grafana - A analytics & monitoring solution](#)
- [Guacamole - Remote server manager](#)
- [Heimdall - Application dashboard](#)
- [Home Assistant - Central smart home service](#)
- [HrConvert - File conversion tool](#)
- [InfluxDB - A time series database](#)
- [Jenkins - Open Source Automation Server](#)
- [LeanTime - Open Source Project Management](#)
- [MeshCentral - Computer management portal](#)
- [NetCheck - Webservice monitoring](#)
- [Nextcloud - Self hosted cloud](#)
- [Node-RED - Smarthome simplified](#)
- [OpenHab - Home automation platform](#)
- [OpenVas - Cyber security scanner](#)
- [OS.js - Browser based OS](#)
- [Plex - Multimedia platform](#)
- [Portainer - Manage your containers](#)
- [SilverStrike - Finance manager](#)
- [SSHwifty - Web ssh and telnet client](#)
- [SVN Server - Software Versioning System](#)
- [Teamspeak Server for Raspberry Pi](#)
- [Traefik with OID Keycloak](#)

- [wbo - Collaborative whiteboard](#)
- [wger - Workout Manager](#)
- [XWiki - A extendable wiki service](#)
- [GMOD TTT - A trouble in terrorist town game server](#)
- [Minecraft Servers](#)
- [Authelia - SSO provider](#)
- [Kutt - A URL shortener](#)
- [IceCoder - A web-based IDE](#)
- [Neko - A browser in the browser](#)
- [Shiori - A bookmark manager](#)
- [Gitlab - A git server](#)
- [RoF/RoD - A ring of death/ring of fire webserver](#)
- [Matrix - A full matrix stack](#)
- [Actual - A budgeting software](#)
- [Akaunting - A budgeting software](#)
- [Immich - A self-hosted Google Photos alternative](#)

# Random notes

A collection of random notes regarding docker.

Random notes

# Registry garbage collect

In folder **/etc/docker/registry** execute **registry garbage-collect config.yml**

# Multiarch builds

Docker supports building images for multiple architectures like ARM or AMD from one Dockerfile on a single device.

## **Resources**

- <https://medium.com/@artur.klauser/building-multi-architecture-docker-images-with-buildx-27d80f7e2408>
- <https://github.com/TryGhost/node-sqlite3/pull/1362>

# DrawIO - A flowchart maker

## Summary

DrawIO is a flowchart maker able to create all kinds of different charts or diagrams. It supports multiple different export modes like PDF, Images or HTML.

## Installation via docker-compose behind traefik proxy

```
services:
  plantuml-server:
    image: jgraph/plantuml-server
    expose:
      - "8080"
    networks:
      - drawio_local
    volumes:
      - /PATH/TO/FOLDER/fonts:/usr/share/fonts/drawio
  image-export:
    image: jgraph/export-server
    expose:
      - "8000"
    networks:
      - drawio_local
    volumes:
      - /PATH/TO/FOLDER/fonts:/usr/share/fonts/drawio
    environment:
      - DRAWIO_SERVER_URL=https://drawio.domain.tld
  drawio:
    image: jgraph/drawio
    links:
      - plantuml-server:plantuml-server
      - image-export:image-export
    depends_on:
      - plantuml-server
      - image-export
    networks:
```

- drawio\_local

- traefik

labels:

- "traefik.enable=true"

- "traefik.http.routers.drawio.rule=Host(`drawio.domain.tld`)"

- "traefik.http.routers.drawio.entrypoints=websecure"

- "traefik.http.routers.drawio.tls=true"

- "traefik.http.services.drawio.loadBalancer.server.port=8080"

- "traefik.http.routers.drawio.middlewares=traefik-forward-auth"

- "traefik.docker.network=traefik"

environment:

- DRAWIO\_SELF\_CONTAINED=1

- PLANTUML\_URL=http://plantuml-server:8080/

- EXPORT\_URL=http://image-export:8000/

- DRAWIO\_BASE\_URL=https://drawio.domain.tld

# Grafana - A analytics & monitoring solution

## Summary

Grafana is a analytics and monitoring solution for every database available. It supports creating dashboards for your database data and also monitors the data, with options for alerting via various ways.

## Installation via docker-compose

To prevent file permission problems it is recommended to set the user id of your host linux user (run the command "*id -u*").

```
version: "2"
services:
  grafana:
    image: grafana/grafana:latest
    user: "1000"
    restart: always
    ports:
      - "3000:3000"
    volumes:
      - /PATH/TO/FOLDER/data:/var/lib/grafana
    environment:
      - GF_SECURITY_ADMIN_USER=USERNAME
      - GF_SECURITY_ADMIN_PASSWORD=PASSWORD
```



# Guacamole - Remote server manager

## Summary

The guacamole service enables users to access remote computer via services like ssh or rdp. This container is a all in one container being able to be run on raspberry pi.

## Installation via docker-compose behind traefik proxy

```
version: '3'

networks:
  traefik:
    name: traefik

volumes:
  remote_guacamole_config:
    external: true

services:
  guacamole:
    image: maxwaldorf/guacamole
    networks:
      - traefik
    volumes:
      - remote_guacamole_config:/config
    restart: always
    labels:
      - "traefik.enable=true"
      - "traefik.http.routers.guac.rule=Host(`guac.domain.com`)"
      - "traefik.http.routers.guac.entrypoints=websecure"
      - "traefik.http.routers.guac.tls=true"
      - "traefik.http.services.guac.loadBalancer.server.port=8080"
      - "traefik.docker.network=traefik"
  backup:
```

image: reg.greinet.com/jareware/docker-volume-backup:2.6.0

environment:

- BACKUP\_CRON\_EXPRESSION="@daily"

volumes:

- remote\_guacamole\_config:/backup/guacamole-config:ro
- /home/ubuntu/backups/guacamole:/archive

# Heimdall - Application dashboard

## Summary

Heimdall is a application dashboard and launcher, organising all your web services in a single application. The documentation canbe found at [linuxserver/Heimdall](https://linuxserver.io/Heimdall).

## Installation via docker-compose behind traefik proxy

```
heimdall:
  image: linuxserver/heimdall
  environment:
    - PUID=1000
    - PGID=1000
    - TZ=Europe/Berlin
  networks:
    - traefik
  volumes:
    - /path/to/config:/config
  labels:
    - "traefik.enable=true"
    - "traefik.http.routers.heimdall.rule=Host(`heimdall.domain.com`)"
    - "traefik.http.routers.heimdall.entrypoints=websecure"
    - "traefik.http.routers.heimdall.tls=true"
    - "traefik.http.services.heimdall.loadBalancer.server.port=80"
    - "traefik.docker.network=traefik"
```

# Home Assistant - Central smart home service

## Summary

Home Assistant is a central smart home service used to connect all your smart home utilities.

## Installation via docker-compose

```
version: '3'
services:
  homeassistant:
    container_name: homeassistant
    image: "ghcr.io/home-assistant/home-assistant:stable"
    volumes:
      - /home/ubuntu/docker/homeassistant/config:/config
      - /etc/localtime:/etc/localtime:ro
    restart: unless-stopped
    privileged: true
    network_mode: host
```

Use behind proxy

Add the http configuration to your configuration.yaml.

```
http:
  base_url: homeassistant.domain.tld
  use_x_forwarded_for: true
  trusted_proxies:
    - 192.1.1.1
```

# HrConvert - File conversion tool

## Summary

Online conversion tools for 75 different file types including audio or video files, documents, presentations, archives and much more.

## Installation via docker-compose.yml with traefik

```
convert:
  image: dwaaan/hrconvert2-docker:latest
  labels:
    - "traefik.enable=true"
    - "traefik.http.routers.convert.rule=Host(`conv.domain.com`)"
    - "traefik.http.routers.convert.entrypoints=websecure"
    - "traefik.http.routers.convert.tls=true"
    - "traefik.http.services.convert.loadBalancer.server.port=80"
    - "traefik.docker.network=traefik"
  networks:
    - traefik
  restart: always
  volumes:
    - "/path/to/folder/config.php:/var/www/html/HRProprietary/HRConvert2/config.php"
```

# InfluxDB - A time series database

## Summary

InfluxDB is a time series database mostly used for sensor data. It supports APIs for data storing, querying and processing.

## Installation

The easiest way to initialize influxdb is by running a docker run command with environment variables to initialize the folder structure. Don't forget to create the folder for the volumes first.

```
docker run -p 8086:8086 \
  -v /PATH/TO/FOLDER/influxdb/data:/var/lib/influxdb2 \
  -v /PATH/TO/FOLDER/config:/etc/influxdb2 \
  -e DOCKER_INFLUXDB_INIT_MODE=setup \
  -e DOCKER_INFLUXDB_INIT_USERNAME=USERNAME \
  -e DOCKER_INFLUXDB_INIT_PASSWORD=PASSWORD \
  -e DOCKER_INFLUXDB_INIT_ORG=ORGANISATION \
  -e DOCKER_INFLUXDB_INIT_BUCKET=BUCKET \
  influxdb:2.0
```

After the initialization you need to delete the docker container again. Then you can start the normal container via docker-compose.

```
version: "2"

services:
  influxdb:
    image: influxdb:2.0
    ports:
      - "8086:8086"
    restart: always
    volumes:
      - /PATH/TO/FOLDER/data:/var/lib/influxdb2
      - /PATH/TO/FOLDER/config:/etc/influxdb2
```

## Data deletion

If you inserted some data wrongfully or want to delete data for any other reason, you need to execute a command in the influxdb container. Either run it via docker exec or just connect into the container shell via portainer. Then you can delete measurements with the following command:

```
influx delete --org ORGANIZATION --bucket BUCKET --start 1970-01-01T00:00:00Z --stop $(date +"%Y-%m-%dT%H:%M:%SZ") --predicate '_measurement="MEASUREMENT"'
```

# Jenkins - Open Source Automation Server

## Summary

Jenkins is an open source automation server, acting as toolchain to build, test and deploy your software.

## Installation via docker-compose behind traefik proxy

```
jenkins:
  image: jenkins/jenkins:lts
  user: root
  privileged: true
  labels:
    - "traefik.enable=true"
    - "traefik.http.routers.jenkins.rule=Host(`jenkins.yourdomain.com`)"
    - "traefik.http.routers.jenkins.entrypoints=websecure"
    - "traefik.http.routers.jenkins.tls=true"
    - "traefik.http.services.jenkins.loadBalancer.server.port=8080"
    - "traefik.docker.network=traefik"
  networks:
    - traefik
  restart: always
  volumes:
    - /path/to/data:/var/jenkins_home
    - /var/run/docker.sock:/var/run/docker.sock
    - /usr/bin/docker:/usr/bin/docker
```



# LeanTime - Open Source Project Management

## Summary

LeanTime is a open source project management solution providing kan-boards, To-Do-Lists and big-sized projects.

## Installation via docker-compose behind traefik proxy

```
leantime_web:
  environment:
    - LEAN_DB_HOST=leantime_db
    - LEAN_DB_USER=leantime_db_user
    - LEAN_DB_PASSWORD=leantime_db_password
    - LEAN_DB_DATABASE=leantime_db_name
    - LEAN_APP_URL=https://leantime.yourdomain.com
  image: leantime/leantime:latest
  labels:
    - "traefik.enable=true"
    - "traefik.http.routers.leantime.rule=Host(`leantime.yourdomain.com`)"
    - "traefik.http.routers.leantime.entrypoints=websecure"
    - "traefik.http.routers.leantime.tls=true"
    - "traefik.http.services.leantime.loadBalancer.server.port=80"
    - "traefik.docker.network=traefik"
  networks:
    - traefik
    - cmas_local
  restart: always
  #user with uid and guid 82 needed on host for volumes
  volumes:
    - /path/to/data/userfiles:/var/www/html/userfiles
    - /path/to/data/userfiles-public:/var/www/html/public/userfiles
leantime_db:
  image: mysql
  command: --default-authentication-plugin=mysql_native_password
```

restart: always

volumes:

- /path/to/data/db:/var/lib/mysql

networks:

- cmas\_local

environment:

- MYSQL\_ROOT\_PASSWORD=db\_root\_password
- MYSQL\_DATABASE=leantime\_db\_name
- MYSQL\_USER=leantime\_db\_user
- MYSQL\_PASSWORD=leantime\_db\_password

# MeshCentral - Computer management portal

## Summary

MeshCentral is a web portal to manage multiple computers via agent and enables screen share and remote control.

## Installation via docker-compose behind traefik proxy

```
meshcentral:
  image: typhonragewind/meshcentral:2
  restart: always
  networks:
    - traefik
  environment:
    - ALLOW_NEW_ACCOUNTS=true
  volumes:
    - /path/to/folder:/opt/meshcentral/meshcentral-data
    - /path/to/folder2:/opt/meshcentral/meshcentral-files
  labels:
    - "traefik.enable=true"
    - "traefik.http.routers.mesh.rule=Host(`mesh.domain.com`)"
    - "traefik.http.routers.mesh.entrypoints=websecure"
    - "traefik.http.routers.mesh.tls=true"
    - "traefik.http.services.mesh.loadBalancer.server.port=4430"
    - "traefik.docker.network=traefik"
```

# NetCheck - Webservice monitoring

## Summary

Tool to monitor your website uptime and performance automatically.

## Installation via docker-compose behind traefik proxy

```
version: '2'

networks:
  traefik:
    external:
      name: traefik
  monitoring_local:
    external:
      name: monitoring_local

services:
  netcheck_db:
    image: postgres
    environment:
      - POSTGRES_USER=PGUSERNAME
      - POSTGRES_PASSWORD=PGPASSWORD
      - POSTGRES_DB=netcheck
    networks:
      - monitoring_local
    volumes:
      - /path/to/folder/:/var/lib/postgresql/data
    restart: always
  netcheck_api:
    image: memphisx/netcheck-api:latest
    environment:
      - POSTGRES_USER=
      - POSTGRES_USER=PGUSERNAME
```

- POSTGRES\_PASSWORD=PGPASSWORD
- POSTGRES\_DB=netcheck
- POSTGRES\_PORT=5432
- POSTGRES\_HOST=netcheck\_db
- SETTINGS\_NOTIFICATIONS\_PUSHOVER\_ENABLED=false
- SETTINGS\_NOTIFICATIONS\_PUSHOVER\_APITOKEN=
- SETTINGS\_NOTIFICATIONS\_PUSHOVER\_USERIDTOKEN=

depends\_on:

- netcheck\_db

labels:

- "traefik.enable=true"
- "traefik.http.routers.netcheck-api.rule=Host(`netcheck.domain.com`) &&

PathPrefix(`/api`,`/docs`,`/events`)"

- "traefik.http.routers.netcheck-api.entrypoints=websecure"
- "traefik.http.routers.netcheck-api.tls=true"
- "traefik.http.services.netcheck-api.loadBalancer.server.port=8080"
- "traefik.docker.network=traefik"

networks:

- traefik
- monitoring\_local

restart: always

netcheck\_web:

image: memphisx/netcheck-frontend:latest

depends\_on:

- netcheck\_api

networks:

- traefik
- monitoring\_local

labels:

- "traefik.enable=true"
- "traefik.http.routers.netcheck-web.rule=Host(`netcheck.domain.com`)"
- "traefik.http.routers.netcheck-web.entrypoints=websecure"
- "traefik.http.routers.netcheck-web.tls=true"
- "traefik.http.services.netcheck-web.loadBalancer.server.port=80"
- "traefik.http.routers.netcheck-web.middlewares=traefik-forward-auth"
- "traefik.docker.network=traefik"

restart: always

# Nextcloud - Self hosted cloud

## Summary

Nextcloud is a self hosted one stop cloud solution, offering support for file hosting, calendars, notes and much more.

## Installation via docker-compose behind traefik proxy

```
services:
  nextcloud_web:
    image: nextcloud
    restart: always
    networks:
      - traefik
    labels:
      - "traefik.enable=true"
      - "traefik.http.routers.nextcloud.rule=Host(`nextcloud.domain.tld`)"
      - "traefik.http.routers.nextcloud.entrypoints=websecure"
      - "traefik.http.routers.nextcloud.tls=true"
      - "traefik.http.services.nextcloud.loadBalancer.server.port=80"
      - "traefik.docker.network=traefik"
    volumes:
      - /PATH/TO/FOLDER/data:/var/www/html
    environment:
      - MYSQL_PASSWORD=PASSWORD
      - MYSQL_DATABASE=nextcloud
      - MYSQL_USER=nextcloud
      - MYSQL_HOST=nextcloud_db
  nextcloud_db:
    image: mariadb
    restart: always
    command: --transaction-isolation=READ-COMMITTED --binlog-format=ROW
    volumes:
      - /PATH/TO/FOLDER/db:/var/lib/mysql
```

environment:

- MYSQL\_ROOT\_PASSWORD=ROOT\_PASSWORD
- MYSQL\_PASSWORD=PASSWORD
- MYSQL\_DATABASE=nextcloud
- MYSQL\_USER=nextcloud

networks:

- traefik

# Node-RED - Smarthome simplified

## Summary

Node-RED is a drag'n'drop service to simplify your smart home automations.

## Installation via docker-compose

```
version: "3.7"

services:
  node-red:
    image: nodered/node-red:latest
    environment:
      - TZ=Europe/Berlin
    ports:
      - "1880:1880"
    volumes:
      - /home/ubuntu/nodered/data:/data
```



# OpenHab - Home automation platform

## Summary

OpenHab is a home automation platform integrating all your different smart home services. The documentation can be found at [openhab.org](https://openhab.org).

## Installation via docker-compose behind traefik proxy

```
openhhab:
  image: openhab/openhab
  labels:
    - "traefik.enable=true"
    - "traefik.http.routers.openhab.rule=Host(`openhab.domain.com`)"
    - "traefik.http.routers.openhab.entrypoints=websecure"
    - "traefik.http.routers.openhab.tls=true"
    - "traefik.http.services.openhab.loadBalancer.server.port=8080"
    - "traefik.docker.network=traefik"
  networks:
    - traefik
  restart: always
```

# OpenVas - Cyber security scanner

## Summary

OpenVas is a cyber security scanner, designed to perform penetration testing on internal and external systems. The documentation can be found at [immauss/openvas](https://immauss.github.io/openvas/).

## Installation via docker-compose behind traefik proxy

Found 2 configurations, not sure which is the working one :/

```
openhav:
  image: "immauss/openvas"
  restart: always
  labels:
    - "traefik.enable=true"
    - "traefik.http.routers.openvas.rule=Host(`openvas.domain.com`)"
    - "traefik.http.routers.openvas.entrypoints=websecure"
    - "traefik.http.routers.openvas.tls=true"
    - "traefik.http.services.openvas.loadBalancer.server.port=9392"
    - "traefik.docker.network=traefik"
  networks:
    - traefik
  environment:
    PASSWORD: "<PASSWORD>"
```

```
version: "2"
services:
  gvm:
    image: securecompliance/gvm
    volumes:
      - /PATH/TO/FOLDER/openvas/gvm:/var/lib/gvm
      - /PATH/TO/FOLDER/openvas/plugins:/var/lib/openvas/plugins
    environment:
      - USERNAME="USERNAME"
```

- PASSWORD="PASSWORD"

- AUTO\_SYNC=true

- DB\_PASSWORD="none"

ports:

- "9392:9392" # Web interface

restart: unless-stopped

version: "3"

services:

openvas:

ports:

- 8080:9392

environment:

- SKIPSYNC=true

volumes:

- /path/to/data:/data

image: immauss/openvas:22.4.32

# OS.js - Browser based OS

## Summary

OS.js is a virtual operating system written in JavaScript and living fully in your browser. The documentation can be found at [os-js/OS.js](https://os-js.github.io/OS.js).

## Installation via docker-compose behind traefik proxy

```
osjs:
  image: osjs/osjs:latest
  labels:
    - "traefik.enable=true"
    - "traefik.http.routers.osjs.rule=Host(`osjs.domain.com`)"
    - "traefik.http.routers.osjs.entrypoints=websecure"
    - "traefik.http.routers.osjs.tls=true"
    - "traefik.http.services.osjs.loadBalancer.server.port=8000"
    - "traefik.docker.network=traefik"
  networks:
    - traefik
  restart: always
```

# Plex - Multimedia platform

## Summary

Manage and view your movie and series collection.

## Installation via docker-compose

```
plex:
  image: ghcr.io/linuxserver/plex
  network_mode: host
  environment:
    - PUID=1000
    - PGID=1000
    - VERSION=docker
    - ADVERTISE_IP=http://192.168.178.188:32400/
  #- PLEX_CLAIM=claim-XXXXXXXXXXXXXXXXXX
  volumes:
    - /PATH/TO/FOLDER/config:/config
    - /PATH/TO/FOLDER/tvseries:/tv
    - /PATH/TO/FOLDER/movies:/movies
  restart: unless-stopped
```

Start the container once with the PLEX\_CLAIM environment variable to claim the plex server to your account.

# Portainer - Manage your containers

## Summary

Portainer is a container management service for docker containers.

## Installation via docker-compose

```
version: "3.3"

services:
  portainer:
    image: "portainer/portainer-ce"
    ports:
      - 9000:9000
    volumes:
      - "/var/run/docker.sock:/var/run/docker.sock:ro"
      - "/home/ubuntu/docker/portainer/data:/data"
    restart: always
```

To change to business edition, use image **portainer/portainer-ee**. You can [get up to 5 nodes for free](#).

## Setup

Create a administrator user, for example admin.

## ▼ New Portainer installation

Please create the initial administrator user.

Username

admin

Password


.....

Confirm password

.....



✓ The password must be at least 8 characters long

 Create user

☒ Allow collection of anonymous statistics. You can find more information about this in our [privacy policy](#).

Then select **Get Started** to use the local docker installation.

### Welcome to Portainer

---

We have connected your local environment of docker to Portainer.

Get started below with your local portainer or connect more container environments.



#### Get Started

Proceed using the  
local environment  
which Portainer is  
running in



#### Add Environments

Connect to other  
environments

# SilverStrike - Finance manager

## Summary

SilverStrike is a finance management tool. The documentation can be found at [silverstrike.org](https://silverstrike.org).

## Installation via docker-compose behind traefik proxy

The service is split up in the service itself and a postgres database.

```
silverstrike_web:
  environment:
    - ALLOWED_HOSTS='*'
    - DATABASE_URL=postgres://silverstrike:DATABASEPASSWORD@silverstrike_db/silverstrikedb
    - SECRET_KEY=SECRETKEY
  image: simhnna/silverstrike
  labels:
    - "traefik.enable=true"
    - "traefik.http.routers.silverstrike.rule=Host(`silverstrike.domain.com`)"
    - "traefik.http.routers.silverstrike.entrypoints=websecure"
    - "traefik.http.routers.silverstrike.tls=true"
    - "traefik.http.services.silverstrike.loadBalancer.server.port=8000"
    - "traefik.docker.network=traefik"
  networks:
    - traefik
    - personal_local
  restart: always
```

```
silverstrike_db:
  environment:
    - POSTGRES_DB=silverstrikedb
    - POSTGRES_USER=silverstrike
    - POSTGRES_PASSWORD=secretpass
  image: postgres:10.3
  volumes:
```



```
- /path/to/data:/var/lib/postgresql/data
```

networks:

```
- personal_local
```

```
restart: always
```

## Traefik V2

services:

silverstrike:

environment:

```
- ALLOWED_HOSTS='*'
```

```
- DATABASE_URL=postgres://silverstrike:secretpass@database/silverstrikedb
```

```
- SECRET_KEY=pass
```

image: simhanna/silverstrike

links:

```
- database:database
```

labels:

```
- "traefik.enable=true"
```

```
- "traefik.http.routers.silverstrike.rule=Host(`budget.domain.com`)"
```

```
- "traefik.http.routers.silverstrike.entrypoints=web-insecure"
```

```
- "traefik.http.routers.silverstrike.middlewares=redirect@file"
```

```
- "traefik.http.routers.silverstrike-secured.rule=Host(`budget.domain.com`)"
```

```
- "traefik.http.routers.silverstrike-secured.tls=true"
```

```
- "traefik.http.routers.silverstrike-secured.tls.certResolver=main"
```

```
- "traefik.http.routers.silverstrike-secured.entrypoints=web-secure"
```

```
- "traefik.docker.network=webv2"
```

```
- "traefik.http.services.silverstrike.loadBalancer.server.port=8000"
```

networks:

```
- webv2
```

database:

environment:

```
POSTGRES_DB: silverstrikedb
```

```
POSTGRES_USER: silverstrike
```

```
POSTGRES_PASSWORD: secretpass
```

image: postgres:10.3

volumes:

```
- /path/to/db:/var/lib/postgresql/data
```

networks:

```
- webv2
```

# SSHwifty - Web ssh and telnet client

## Summary

Web client to use ssh and telnet in your browser.

## Installation via docker-compose behind traefik proxy

```
sshwifty:
  image: niruix/sshwifty:latest
  labels:
    - "traefik.enable=true"
    - "traefik.http.routers.ssh.rule=Host(`ssh.domain.tld`)"
    - "traefik.http.routers.ssh.entrypoints=websecure"
    - "traefik.http.routers.ssh.tls=true"
    - "traefik.http.services.ssh.loadBalancer.server.port=8182"
    - "traefik.docker.network=traefik"
  networks:
    - traefik
  restart: always
```

# SVN Server - Software Versioning System

## Summary

The SVN Server is a software versioning system based on subversion used to maintain current and historical versions of software.

## Installation via docker-compose behind traefik proxy

```
svnserver:
  image: elleflorio/svn-server
  restart: always
  labels:
    - "traefik.enable=true"
    - "traefik.http.routers.svn.rule=Host(`svn.yourdomain.com`)"
    - "traefik.http.routers.svn.entrypoints=websecure"
    - "traefik.http.routers.svn.tls=true"
    - "traefik.http.services.svn.loadBalancer.server.port=80"
    - "traefik.docker.network=traefik"
  volumes:
    - /path/to/data:/home/svn
    - /path/to/data/passwd:/etc/subversion/passwd
  ports:
    - 3690:3690
  networks:
    - traefik
```

## Another install

```
svnserver:
  image: elleflorio/svn-server:issue-12
  labels:
    - "traefik.enable=true"
    - "traefik.http.routers.svn.rule=Host(`svn.domain.com`)"
    - "traefik.http.routers.svn.entrypoints=web-insecure"
```

- "traefik.http.routers.svn.middlewares=redirect@file"
- "traefik.http.middlewares.svn.replacepathregex.regex=^/\$"
- "traefik.http.middlewares.svn.replacepathregex.replacement=/svn"
- "traefik.http.routers.svn-secured.middlewares=svn@docker"
- "traefik.http.routers.svn-secured.rule=Host(`svn.domain.com`)"
- "traefik.http.routers.svn-secured.tls=true"
- "traefik.http.routers.svn-secured.tls.certResolver=main"
- "traefik.http.routers.svn-secured.entrypoints=web-secure"
- "traefik.docker.network=webv2"
- "traefik.http.services.svn.loadBalancer.server.port=80"

restart: always

ports:

- 3690:3690

volumes:

- "/path/to/svn:/home/svn"
- "/path/to/passwd:/etc/subversion/passwd"
- "/path/to/subversion-access-control:/etc/subversion/subversion-access-control"

networks:

- webv2

# Teamspeak Server for Raspberry Pi

## Summary

A teamspeak server to run on a raspberry pi.

## Installation via docker-compose behind traefik proxy

```
version: "3"
networks:
  social:
    name: social
volumes:
  social_teamspeak_data:
    external: true
services:
  ts3-server:
    image: ertagh/teamspeak3-server:latest-box
    restart: unless-stopped
    ports:
      - 9987:9987/udp
      - 10011:10011
      - 30033:30033
    environment:
      TS_UPDATE: 1
      TS_UPDATE_BACKUP: 1
      TIME_ZONE: Europe/Berlin
      UID: 1000
      GID: 1000
    volumes:
      - social_teamspeak_data:/teamspeak/save/
    networks:
      - social
```

# Traefik with OID Keycloak

```
version: '2'

networks:
  traefik:
    name: traefik

volumes:
  docker_networking_keycloak_postgresdata:
    external: true
  docker_networking_traefik_acme:
    external: true
  docker_networking_traefik_rules:
    external: true
  docker_networking_traefik_logs:
    external: true
  docker_networking_keycloak_postgresbackup:
    external: true

services:
  traefik:
    image: traefik
    restart: always
    networks:
      - traefik
    command:
      - --pilot.token=xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
      - --entrypoints.web.address=:80
      - --entrypoints.websecure.address=:443
      - --entryPoints.ws.address=:8081
      - --entryPoints.wss.address=:8083
      - --providers.docker
      - --providers.docker.exposedByDefault=false
      - --providers.file.directory=/config/
      - --api
      - --log.filePath=logs/log.txt
```

- --log.format=json
- --log.level=DEBUG
- --accesslog=true
- --accesslog.filepath=/logs/access.log
- --certificatesresolvers.leresolver.acme.email=xxxxxxxxxxxxxx@mail.com
- --certificatesresolvers.leresolver.acme.storage=/acme/acme.json
- --certificatesresolvers.leresolver.acme.dnschallenge=true
- --certificatesresolvers.leresolver.acme.dnschallenge.provider=namedotcom
- --certificatesresolvers.leresolver.acme.dnschallenge.resolvers=163.114.216.17

environment:

- [illegible]

```
ports:
```

- "80:80"
- "443:443"
- "8081:8081"
- "8083:8083"

volumes:

- "/var/run/docker.sock:/var/run/docker.sock:ro"
- docker\_networking\_traefik\_logs:/logs/
- docker\_networking\_traefik\_acme:/acme/
- docker\_networking\_traefik\_rules:/config/

labels:

## # Dashboard

- "traefik.enable=true"
- "traefik.http.routers.traefik.rule=Host(`traefik.domain.com`)"
- "traefik.http.routers.traefik.service=api@internal"
- "traefik.http.routers.traefik.entrypoints=websecure"
- "traefik.http.routers.traefik.middlewares=traefik-forward-auth"
- "traefik.http.routers.traefik.tls=true"

```
# global redirect to https
```

- ```
- "traefik.http.routers.http-catchall.rule=hostregexp(`{host:.+}`)"
- "traefik.http.routers.http-catchall.entrypoints=web"
- "traefik.http.routers.http-catchall.middlewares=redirect-to-https"
```

```
# middleware redirect
```

```
- "traefik.http.middlewares.redirect-to-https.redirectscheme.scheme=https"
```

```
# traefik network
```

```
- "traefik.docker.network=traefik"
```

```
# global wildcard certificates
```

```
- 'traefik.http.routers.wildcard-certs.tls.certresolver=leresolver'
```

```
- 'traefik.http.routers.wildcard-certs.tls.domains[0].main=domain.com'
```

```
- 'traefik.http.routers.wildcard-certs.tls.domains[0].sans=*.domain.com'
```

```
extra_hosts:
```

```
- host.docker.internal:172.1.1.1
```

```
keycloak:
```

```
image: mihaibob/keycloak:15.0.1
```

```
restart: always
```

```
labels:
```

```
- "traefik.enable=true"
```

```
- "traefik.http.routers.keycloak.rule=Host(`keycloak.domain.com`)"
```

```
- "traefik.http.routers.keycloak.entrypoints=websecure"
```

```
- "traefik.http.routers.keycloak.tls=true"
```

```
- "traefik.http.services.keycloak.loadBalancer.server.port=8080"
```

```
- "traefik.docker.network=traefik"
```

```
networks:
```

```
- traefik
```

```
environment:
```

```
- KEYCLOAK_USER=admin
```

```
- KEYCLOAK_PASSWORD=xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
```

```
- PROXY_ADDRESS_FORWARDING=true
```

```
- KEYCLOAK_HOSTNAME=keycloak.domain.com
```

```
- DB_VENDOR=POSTGRES
```

```
- DB_ADDR=postgres
```

```
- DB_DATABASE=keycloak
```

```
- DB_USER=keycloak
```

```
- DB_SCHEMA=public
```

```
- DB_PASSWORD=xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
```

```
depends_on:
```

```
- postgres
```

```
postgres:
```

```
user: "65534:100"
```

```
image: postgres:13.4
```

```
restart: unless-stopped
```



volumes:

- docker\_networking\_keycloak\_postgresdata:/var/lib/postgresql/data

environment:

- PGDATA=/var/lib/postgresql/data/keycloak
- POSTGRES\_DB=keycloak
- POSTGRES\_USER=keycloak
- POSTGRES\_PASSWORD=xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

networks:

- traefik

#-----Keycloak-Postgres-

Backup-----

pgbackups:

image: prodrigestivill/postgres-backup-local

restart: always

volumes:

- docker\_networking\_keycloak\_postgresbackup:/backups

links:

- postgres

depends\_on:

- postgres

environment:

- POSTGRES\_HOST=postgres
- POSTGRES\_DB=keycloak
- POSTGRES\_USER=keycloak
- POSTGRES\_PASSWORD=xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
- SCHEDULE=@daily
- BACKUP\_KEEP\_DAYS=7
- BACKUP\_KEEP\_WEEKS=4
- BACKUP\_KEEP\_MONTHS=6
- HEALTHCHECK\_PORT=8080

networks:

- traefik

healthcheck:

test: curl --fail http://localhost:8080 || exit 1

interval: 5m

retries: 5

start\_period: 20s

timeout: 10s

traefik-forward-auth:

image: thomseddon/traefik-forward-auth:2-arm64

restart: unless-stopped

command:

- "--default-provider=oidc"
- "--providers.oidc.issuer-url=https://keycloak.domain.com/auth/realms/master"
- "--providers.oidc.client-id=traefik-forward-auth"
- "--providers.oidc.client-secret=xxxxxxxxxxxxxxxxxxxxxxxxxxxx"
- "--secret=xxxxxxxxxxxxxxxxxxxxxxxxxxxx"
- "--insecure-cookie"
- "--cookie-domain=domain.com"
- "--auth-host=auth.domain.com"
- "--log-level=debug"

labels:

- "traefik.enable=true"
- "traefik.http.routers.traefik-forward-auth.rule=Host(`auth.domain.com`)"
- "traefik.http.services.traefik-forward-auth.loadbalancer.server.port=4181"
- "traefik.http.routers.traefik-forward-auth.entrypoints=websecure"
- "traefik.http.routers.traefik-forward-auth.tls=true"
- "traefik.docker.network=traefik"
- "traefik.http.routers.traefik-forward-auth.middlewares=traefik-forward-auth"
- "traefik.http.middlewares.traefik-forward-auth.forwardauth.address=http://traefik-forward-auth:4181"
- "traefik.http.middlewares.traefik-forward-auth.forwardauth.authResponseHeaders=X-Forwarded-User"
- "traefik.http.middlewares.traefik-forward-auth.forwardauth.trustForwardHeader=true"

networks:

- traefik

depends\_on:

- keycloak

# wbo - Collaborative whiteboard

## Summary

WBO is a collaborative whiteboard service. The documentation can be found at [lovasoa/whitebophr](https://lovasoa.github.io/whitebophr/).

## Installation via docker-compose behind traefik proxy

```
whiteboard:
  image: lovasoa/wbo:latest
  labels:
    - "traefik.enable=true"
    - "traefik.http.routers.whiteboard.rule=Host(`whiteboard.yourdomain.com`)"
    - "traefik.http.routers.whiteboard.entrypoints=websecure"
    - "traefik.http.routers.whiteboard.tls=true"
    - "traefik.http.services.whiteboard.loadBalancer.server.port=80"
    - "traefik.docker.network=traefik"
  networks:
    - traefik
  restart: always
  volumes:
    - "/path/to/data:/opt/app/server-data"
```

# wger - Workout Manager

## Summary

A workout manager to plan your training routines and track your eating habits.

## Installation via docker-compose behind traefik proxy

```
wger_web:
  image: wger/devel:2.0-dev
  volumes:
    - /path/to/folder/:/home/wger/static
    - /path/to/folder2/:/home/wger/media
  environment:
    - DJANGO_DB_ENGINE=django.db.backends.postgresql
    - DJANGO_DB_DATABASE=wger
    - DJANGO_DB_USER=wger
    - DJANGO_DB_PASSWORD=DBPASSWORD
    - DJANGO_DB_HOST=wger_db
    - DJANGO_DB_PORT=5432
    - DJANGO_CACHE_BACKEND=django_redis.cache.RedisCache
    - DJANGO_CACHE_LOCATION=redis://wger_cache:6379/1
    - DJANGO_CACHE_TIMEOUT=1300000
    - DJANGO_CACHE_CLIENT_CLASS=django_redis.client.DefaultClient
    - DJANGO_DEBUG=False
    - WGER_USE_GUNICORN=True
  depends_on:
    - wger_db
    - wger_cache
  networks:
    - personal_local
  restart: always

wger_nginx:
  image: nginx
  volumes:
    - /path/to/folder/:wger/static:ro
    - /path/to/folder2/:wger/media:ro
    - /path/to/folder3/nginx.conf:/etc/nginx/conf.d/default.conf
```

depends\_on:

- wger\_web

networks:

- traefik
- personal\_local

labels:

- "traefik.enable=true"
- "traefik.http.routers.wger.rule=Host(`wger.domain.com`)"
- "traefik.http.routers.wger.entrypoints=websecure"
- "traefik.http.routers.wger.tls=true"
- "traefik.http.services.wger.loadBalancer.server.port=80"
- "traefik.docker.network=traefik"

restart: always

wger\_db:

image: postgres:12.0-alpine

volumes:

- /home/agreiner/dockerData/personal/wger/db:/var/lib/postgresql/data/

environment:

- POSTGRES\_USER=wger
- POSTGRES\_PASSWORD=DBPASSWORD
- POSTGRES\_DB=wger

networks:

- personal\_local

restart: always

wger\_cache:

restart: always

image: redis:latest

networks:

- personal\_local

# XWiki - A extendable wiki service

## Summary

XWiki is a wiki service built on java, which offers a lot of extendability through java plugins.

## Installation via docker-compose behind traefik proxy

```
xwiki_service:
  image: "xwiki:lts-postgres-tomcat"
  restart: always
  labels:
    - "traefik.enable=true"
    - "traefik.http.routers.wiki.rule=Host(`wiki.domain.tld`)"
    - "traefik.http.routers.wiki.entrypoints=websecure"
    - "traefik.http.routers.wiki.tls=true"
    - "traefik.http.services.wiki.loadBalancer.server.port=8080"
    - "traefik.docker.network=traefik"
  environment:
    - DB_USER=USERNAME
    - DB_PASSWORD=PASSWORD
    - DB_DATABASE=DB
    - DB_HOST=xwiki_db
  volumes:
    - /PATH/TO/FOLDER/data:/usr/local/xwiki
  networks:
    - cmas_local
    - traefik

xwiki_db:
  image: "postgres:12-alpine"
  restart: always
  volumes:
    - /PATH/TO/FOLDER/db:/var/lib/postgresql/data
  environment:
    - POSTGRES_ROOT_PASSWORD=DB_ROOT_PASSWORD
```

- POSTGRES\_PASSWORD=PASSWORD
- POSTGRES\_USER=USERNAME
- POSTGRES\_DB=xwiki
- POSTGRES\_INITDB\_ARGS="--encoding=UTF8"

networks:

- cmas\_local

# GMOD TTT - A trouble in terrorist town game server

## Summary

Trouble in Terrorist Town is a gamemode for Garry's Mod involving innocent people and traitors. This is a game server for said mode.

## Installation via docker-compose

```
services:
  ttt:
    image: jusito/docker-ttt:gmod_ttt_debian
    ports:
      - "27015:27015/udp"
      - "27015:27015/tcp"
    environment:
      - SERVERPORT=27015
      - INSTALL_CSS=true
      - WORKSHOP_COLLECTION_ID=1957242851
      #- WORKSHOP_COLLECTION_ID=2267908973
      - SERVER_NAME=SERVERNAME
      - SERVER_DEFAULT_MAP=ttt_bb_teenroom_b2
      - SERVER_RCON_PASSWORD=RCONPASSWORD
    volumes:
      - "/PATH/TO/FOLDER/ttt:/home/steam/serverfiles"
```



# Minecraft Servers

## FTB Unlimited Reloaded

ftbunlimitedreloaded:

image: jusito/docker-ftb-alpine:FTBUltimateReloaded-1.9.0-1.12.2

environment:

- ADMIN\_NAME=<MCUSERNAME>

- motd=<MOTD>

- online\_mode=false

- JAVA\_PARAMETERS=-Xms4G -Xmx4G -XX:+UnlockExperimentalVMOptions -XX:+UseG1GC -

XX:+UseStringDeduplication -XX:+UseCGroupMemoryLimitForHeap -Dio.netty.leakDetection.level=advanced

ports:

- "25565:25565"

volumes:

- "/PATH/TO/FOLDER/data:/home/docker"

restart: always

## Vanilla

minecraft\_1\_19\_3:

image: itzg/minecraft-server:latest

ports:

- "25555:25555"

environment:

- EULA=TRUE

- MOTD="-"

- SERVER\_NAME=""

- TYPE=PAPER

- VERSION=1.19.3

- MEMORY=6G

- INIT\_MEMORY=1G

- MAX\_MEMORY=10G

- ENFORCE\_WHITELIST=true

volumes:

- "/PATH/TO/FOLDER/data:/data"

restart: unless-stopped

```
tty: true
stdin_open: true
```

## Curseforge Modded

```
minecraft_1_19_2_modded:
  image: itzg/minecraft-server:latest
  ports:
    - "25565:25565"
  environment:
    - EULA=TRUE
    - MOTD="-"
    - SERVER_NAME=""
    - TYPE=AUTO_CURSEFORGE
    - VERSION=1.19.2
    - MEMORY=6G
    - INIT_MEMORY=1G
    - MAX_MEMORY=10G
    - ENFORCE_WHITELIST=true
    - CF_SLUG=bbou-server
  volumes:
    - "/PATH/TO/FOLDER/data:/data"
  restart: unless-stopped
  tty: true
  stdin_open: true
```

## Automatic Backups

```
backup_minecraft_1_19_2_modded:
  image: itzg/mc-backup
  restart: unless-stopped
  environment:
    BACKUP_INTERVAL: "4h"
    PRUNE_BACKUPS_DAYS: "5"
  volumes:
    - /PATH/TO/MC/SERVER/data:/data:ro
    - /PATH/TO/BACKUPS:/backups
  network_mode: "service:mc_server"
```

# Authelia - SSO provider

## **docker-compose.yml**

```
version: '2'

networks:
  traefik:
    name: traefik
    external: true
volumes:
  networking_authelia_config:
    external: true
  authelia:
    image: authelia/authelia
    restart: unless-stopped
    networks:
      - traefik
  expose:
    - 9091
  volumes:
    - networking_authelia_config:/config
  environment:
    - TZ=Europe/Berlin
  labels:
    - 'traefik.enable=true'
    - 'traefik.http.routers.authelia.rule=Host(`auth.domain.com`)'
    - 'traefik.http.routers.authelia.entryPoints=websecure'
    - 'traefik.http.routers.authelia.tls=true'
    - 'traefik.http.services.authelia.loadBalancer.server.port=9091'
    -
'traefik.http.middlewares.authelia.forwardAuth.address=http://authelia:9091/api/verify?rd=https%3A%2F%2Fauth.domain.com%2F'
    - 'traefik.http.middlewares.authelia.forwardAuth.trustForwardHeader=true'
    - 'traefik.http.middlewares.authelia.forwardAuth.authResponseHeaders=Remote-User,Remote-Groups,Remote-Name,Remote-Email'
    - 'traefik.http.middlewares.authelia-basic.forwardAuth.address=http://authelia:9091/api/verify?auth=basic'
```

- 'traefik.http.middlewares.authelia-basic.forwardAuth.trustForwardHeader=true'
- 'traefik.http.middlewares.authelia-basic.forwardAuth.authResponseHeaders=Remote-User,Remote-Groups,Remote-Name,Remote-Email'

## Generate password

```
docker run authelia/authelia:latest authelia crypto hash generate argon2 --password '<PASSWORD>'
```

# Kutt - A URL shortener

## Usefull commands

```
psql -U user kutt
```

```
SELECT verification_token FROM users;
```

# IceCoder - A web-based IDE

## Installation via docker-compose behind traefik proxy

```
icecoder:
  image: greinet/httpd-php-icecoder
  labels:
    - "traefik.enable=true"
    - "traefik.http.routers.icecoder.rule=Host(`icecoder.domain.com`)"
    - "traefik.http.routers.icecoder.entrypoints=web-insecure"
    - "traefik.http.routers.icecoder.middlewares=redirect@file"
    - "traefik.http.routers.icecoder-secured.rule=Host(`icecoder.domain.com`)"
    - "traefik.http.routers.icecoder-secured.tls=true"
    - "traefik.http.routers.icecoder-secured.tls.certResolver=main"
    - "traefik.http.routers.icecoder-secured.entrypoints=web-secure"
    - "traefik.docker.network=webv2"
    - "traefik.http.services.icecoder.loadBalancer.server.port=80"
  networks:
    - webv2
  restart: always
```

# Neko - A browser in the browser

## Docker compose

```
neko:
  environment:
    DISPLAY: :99.0
    NEKO_PASSWORD: nekopw
    NEKO_PASSWORD_ADMIN: adminpw
    NEKO_BIND: :8080
    NEKO_EPR: 59000-59100
  image: nurdism/neko:firefox
  labels:
    - "traefik.enable=true"
    - "traefik.http.routers.neko.rule=Host(`neko.domain.com`)"
    - "traefik.http.routers.neko.entrypoints=websecure"
    - "traefik.http.routers.neko.tls=true"
    - "traefik.http.services.neko.loadBalancer.server.port=8080"
    - "traefik.docker.network=traefik"
  networks:
    - traefik
  ports:
    - "59000-59100:59000-59100/udp"
  restart: always
  shm_size: "1gb"
```

# Shiori - A bookmark manager

## Docker compose

```
shiori:
  image: radhifadlillah/shiori
  labels:
    - "traefik.enable=true"
    - "traefik.http.routers.shiori.rule=Host(`bookmarks.domain.com`)"
    - "traefik.http.routers.shiori.entrypoints=web-insecure"
    - "traefik.http.routers.shiori.middlewares=redirect@file"
    - "traefik.http.routers.shiori-secured.rule=Host(`bookmarks.domain.com`)"
    - "traefik.http.routers.shiori-secured.tls=true"
    - "traefik.http.routers.shiori-secured.tls.certResolver=main"
    - "traefik.http.routers.shiori-secured.entrypoints=web-secure"
    - "traefik.docker.network=webv2"
    - "traefik.http.services.shiori.loadBalancer.server.port=8080"
  volumes:
    - /home/agreiner/dockerData/shiori:/srv/shiori
  networks:
    - webv2
```



# Gitlab - A git server

## Docker compose

services:

gitlab:

image: 'gitlab/gitlab-ce:latest'

restart: always

hostname: 'gitlab'

labels:

- "traefik.enable=true"
- "traefik.docker.network=webv2"
- "traefik.http.routers.gitlab.rule=Host(`gitlab.domain.com`)"
- "traefik.http.routers.gitlab.entrypoints=web-insecure"
- "traefik.http.routers.gitlab.middlewares=redirect@file"
- "traefik.http.routers.gitlab-secured.rule=Host(`gitlab.domain.com`)"
- "traefik.http.routers.gitlab-secured.tls=true"
- "traefik.http.routers.gitlab-secured.tls.certResolver=main"
- "traefik.http.routers.gitlab-secured.entrypoints=web-secure"
- "traefik.http.services.gitlab.loadBalancer.server.port=80"
- "traefik.http.routers.gitlab.service=gitlab"
- "traefik.http.routers.gitlab-secured.service=gitlab"
- "traefik.http.routers.gitprom.rule=Host(`gitprom.domain.com`)"
- "traefik.http.routers.gitprom.entrypoints=web-insecure"
- "traefik.http.routers.gitprom.middlewares=redirect@file"
- "traefik.http.routers.gitprom-secured.rule=Host(`gitprom.domain.com`)"
- "traefik.http.routers.gitprom-secured.tls=true"
- "traefik.http.routers.gitprom-secured.tls.certResolver=main"
- "traefik.http.routers.gitprom-secured.entrypoints=web-secure"
- "traefik.http.services.gitprom.loadBalancer.server.port=9090"
- "traefik.http.routers.gitprom.service=gitprom"
- "traefik.http.routers.gitprom-secured.service=gitprom"

environment:

GITLAB\_OMNIBUS\_CONFIG: |

external\_url 'https://gitlab.domain.com'

nginx['listen\_https'] = false

nginx['listen\_port'] = 80

```
prometheus['listen_address'] = 'localhost:9090'
```

volumes:

- /path/to/config:/etc/gitlab
- /path/to/logs:/var/log/gitlab
- /path/to/data:/var/opt/gitlab

networks:

- webv2

version: "2"

# RoF/RoD - A ring of death/ring of fire webserver

## Docker compose

```
rodBackup:
  image: agreiner/rod:v4
  ports:
    - 8080:8080
    - 4848:4848
  networks:
    - webv2
rod:
  image: agreiner/rod:v7
  labels:
    - "traefik.enable=true"
    - "traefik.http.routers.rod.rule=Host(`rod.domain.com`)"
    - "traefik.http.routers.rod.entrypoints=web-insecure"
    - "traefik.http.routers.rod.middlewares=redirect@file"
    - "traefik.http.routers.rod-secured.rule=Host(`rod.domain.com`)"
    - "traefik.http.routers.rod-secured.tls=true"
    - "traefik.http.routers.rod-secured.tls.certResolver=main"
    - "traefik.http.routers.rod-secured.entrypoints=web-secure"
    - "traefik.docker.network=webv2"
    - "traefik.http.services.rod.loadBalancer.server.port=8080"
  networks:
    - webv2
rof:
  image: agreiner/rof:latest
  labels:
    - "traefik.enable=true"
    - "traefik.http.routers.rof.rule=Host(`rof.domain.com`)"
    - "traefik.http.routers.rof.entrypoints=web-insecure"
    - "traefik.http.routers.rof.middlewares=redirect@file"
    - "traefik.http.routers.rof-secured.rule=Host(`rof.domain.com`)"
```

- "traefik.http.routers.rof-secured.tls=true"
- "traefik.http.routers.rof-secured.tls.certResolver=main"
- "traefik.http.routers.rof-secured.entrypoints=web-secure"
- "traefik.docker.network=webv2"
- "traefik.http.services.rof.loadBalancer.server.port=8080"

networks:

- webv2

# Matrix - A full matrix stack

## Docker compose

conduit:

image: matrixconduit/matrix-conduit:latest

restart: unless-stopped

networks:

- traefik
- matrix\_default

volumes:

- /path/to/db:/var/lib/matrix-conduit/

environment:

- CONDUIT\_SERVER\_NAME=matrix-domain.t.com
- CONDUIT\_DATABASE\_PATH=/var/lib/matrix-conduit/
- CONDUIT\_DATABASE\_BACKEND=rocksdb
- CONDUIT\_PORT=6167
- CONDUIT\_MAX\_REQUEST\_SIZE=10000000 # in bytes, ~10 MB
- CONDUIT\_ALLOW\_REGISTRATION=false
- CONDUIT\_ALLOW\_FEDERATION=true
- CONDUIT\_MAX\_CONCURRENT\_REQUESTS=5
- CONDUIT\_ALLOW\_CHECK\_FOR\_UPDATES=true
- CONDUIT\_TRUSTED\_SERVERS=["matrix.org"]
- CONDUIT\_ADDRESS=0.0.0.0
- CONDUIT\_CONFIG="" # Ignore this

labels:

- "traefik.enable=true"
- "traefik.http.routers.matrix.rule=Host(`matrix-domain.t.com`)"
- "traefik.http.routers.matrix.entrypoints=websecure"
- "traefik.http.routers.matrix.tls=true"
- "traefik.http.routers.matrix.middlewares=cors-headers-matrix@docker"
- "traefik.http.services.matrix.loadBalancer.server.port=6167"
- "traefik.docker.network=traefik"
- "traefik.http.middlewares.cors-headers-matrix.headers.accessControlAllowOriginList=\*"
- "traefik.http.middlewares.cors-headers-matrix.headers.accessControlAllowHeaders=Origin, X-Requested-

With, Content-Type, Accept, Authorization"

- "traefik.http.middlewares.cors-headers-matrix.headers.accessControlAllowMethods=GET, POST, PUT,

## DELETE, OPTIONS"

conduit-well-known:

image: nginx:latest

restart: unless-stopped

networks:

- traefik

volumes:

- /path/to/matrix.conf:/etc/nginx/conf.d/matrix.conf # the config to serve the .well-known/matrix files
- /path/to/www:/var/www/ # location of the client and server .well-known-files

labels:

- "traefik.enable=true"
- "traefik.http.routers.matrix-wellknown.rule=Host(`matrix-domain.com`) && PathPrefix(`/well-known/matrix`)"

known/matrix`)"

- "traefik.http.routers.matrix-wellknown.entrypoints=websecure"
- "traefik.http.routers.matrix-wellknown.tls=true"
- "traefik.http.routers.matrix-wellknown.middlewares=cors-headers-matrix@docke"
- "traefik.http.services.matrix-wellknown.loadBalancer.server.port=80"
- "traefik.docker.network=traefik"
- "traefik.http.middlewares.cors-headers-matrix.headers.accessControlAllowOriginList=\*"
- "traefik.http.middlewares.cors-headers-matrix.headers.accessControlAllowHeaders=Origin, X-Requested-

With, Content-Type, Accept, Authorization"

- "traefik.http.middlewares.cors-headers-matrix.headers.accessControlAllowMethods=GET, POST, PUT,

## DELETE, OPTIONS"

element-web:

image: vectorim/element-web:latest

restart: unless-stopped

volumes:

- /path/to/config.json:/app/config.json

networks:

- traefik

depends\_on:

- conduit

labels:

- "traefik.enable=true"
- "traefik.http.routers.element.rule=Host(`element.domain.com`)"
- "traefik.http.routers.element.entrypoints=websecure"
- "traefik.http.routers.element.tls=true"
- "traefik.http.services.element.loadBalancer.server.port=80"
- "traefik.docker.network=traefik"



# Actual - A budgeting software

## Installation via docker-compose behind traefik proxy

```
services:
  actual_server:
    image: actualbudget/actual-server:latest
    ports:
      - '5006:5006'
    environment:
      - ACTUAL_UPLOAD_FILE_SYNC_SIZE_LIMIT_MB=100
      - ACTUAL_UPLOAD_SYNC_ENCRYPTED_FILE_SYNC_SIZE_LIMIT_MB=200
      - ACTUAL_UPLOAD_FILE_SIZE_LIMIT_MB=100
    volumes:
      - ./actual-data:/data
    restart: unless-stopped
    labels:
      - "traefik.enable=true"
      - "traefik.http.routers.budget.rule=Host(`budget.domain.com`)"
      - "traefik.http.routers.budget.entrypoints=websecure"
      - "traefik.http.routers.budget.tls=true"
      - "traefik.http.services.budget.loadBalancer.server.port=5006"
      - "traefik.docker.network=traefik"
    networks:
      - traefik
```



# Akaunting - A budgeting software

## Installation via docker-compose behind traefik proxy

```
services:
  akaunting:
    image: docker.io/akaunting/akaunting:3.1.4
    labels:
      - "traefik.enable=true"
      - "traefik.http.routers.budget.rule=Host(`budget.domain.com`)"
      - "traefik.http.routers.budget.entrypoints=websecure"
      - "traefik.http.routers.budget.tls=true"
      - "traefik.http.services.budget.loadBalancer.server.port=80"
      - "traefik.docker.network=traefik"
    environment:
      # Use for setup
      - AKAUNTING_SETUP=false
      - APP_INSTALLED=true
      # Further variables
      - APP_URL=https://budget.domain.com
      - LOCALE=de-DE
      - DB_HOST=akaunting-db
      - DB_PORT=3306
      - DB_NAME=akaunting
      - DB_DATABASE=akaunting
      - DB_USERNAME=admin
      - DB_PASSWORD=db_pw
      - DB_PREFIX=domain_
      - COMPANY_NAME=Domain
      - COMPANY_EMAIL=dummy
      - ADMIN_EMAIL=admin@domain.com
      - ADMIN_PASSWORD=pw
      - APP_DEBUG=true
    ports:
```

- 1122:80

depends\_on:

- akaunting-db

#volumes:

# - akaunting-data:/var/www/html

networks:

# - traefik

- personal\_default

restart: unless-stopped

akaunting-db:

image: mariadb:11.1.3

#volumes:

# - akaunting-db:/var/lib/mysql

restart: unless-stopped

environment:

- MYSQL\_DATABASE=akaunting

- MYSQL\_USER=admin

- MYSQL\_PASSWORD=db\_pw

- MYSQL\_RANDOM\_ROOT\_PASSWORD=yes

networks:

- personal\_default

phpmyadmin:

image: phpmyadmin

restart: always

expose:

- "40001"

ports:

- "40001:80"

environment:

- PMA\_HOST=akaunting-db

- PMA\_PORT=3306

networks:

- personal\_default

# Immich - A self-hosted Google Photos alternative

## Installation via docker-compose behind traefik proxy

```
version: '2'

networks:
  traefik:
    external: true
  immich_default:
    external: true

volumes:
  immich_server_upload:
    external: true
  immich_postgres_data:
    external: true
  immich_ml_cache:
    external: true
  immich_redis_data:
    external: true

services:
  server:
    image: ghcr.io/immich-app/immich-server:${IMMICH_VERSION}
    command: [ "start.sh", "immich" ]
    volumes:
      - immich_server_upload:/usr/src/app/upload
      - /etc/localtime:/etc/localtime:ro
    environment:
      - DB_HOSTNAME=postgres
      - DB_USERNAME=postgres
      - DB_DATABASE_NAME=immich
      - REDIS_HOSTNAME=redis
```

- DB\_PASSWORD=\${DB\_PASSWORD}

depends\_on:

- redis
- postgres

networks:

- immich\_default
- traefik

restart: unless-stopped

labels:

- "traefik.enable=true"
- "traefik.http.routers.immich.rule=Host(`immich.domain.com`)"
- "traefik.http.routers.immich.entrypoints=websecure"
- "traefik.http.routers.immich.tls=true"
- "traefik.http.services.immich.loadBalancer.server.port=3001"
- "traefik.docker.network=traefik"
- # CORS for duplicate finder
- "traefik.http.routers.immich.middlewares=immich-cors"
- "traefik.http.middlewares.immich-cors.headers.accessControlAllowOriginList=\*"
- "traefik.http.middlewares.immich-cors.headers.accessControlAllowMethods=GET, PUT, POST, DELETE,

OPTIONS"

- "traefik.http.middlewares.immich-cors.headers.accessControlAllowHeaders=X-API-Key, User-Agent,

Content-Type"

- "traefik.http.middlewares.immich-cors.headers.accessControlMaxAge=1728000"

microservices:

image: ghcr.io/immich-app/immich-server:\${IMMICH\_VERSION}

command: [ "start.sh", "microservices" ]

networks:

- immich\_default

volumes:

- immich\_server\_upload:/usr/src/app/upload
- /etc/localtime:/etc/localtime:ro

environment:

- DB\_HOSTNAME=postgres
- DB\_USERNAME=postgres
- DB\_DATABASE\_NAME=immich
- REDIS\_HOSTNAME=redis
- DB\_PASSWORD=\${DB\_PASSWORD}

depends\_on:

- redis
- postgres

```

restart: unless-stopped
machine-learning:
  image: ghcr.io/immich-app/immich-machine-learning:${IMMICH_VERSION}
  volumes:
    - immich_ml_cache:/cache
  networks:
    - immich_default
  environment:
    - DB_HOSTNAME=postgres
    - DB_USERNAME=postgres
    - DB_DATABASE_NAME=immich
    - REDIS_HOSTNAME=redis
    - DB_PASSWORD=${DB_PASSWORD}
restart: unless-stopped
redis:
  image: redis:6.2-
alpine@sha256:c5a607fb6e1bb15d32bbcf14db22787d19e428d59e31a5da67511b49bb0f1ccc
restart: unless-stopped
networks:
  - immich_default
volumes:
  - immich_redis_data:/data
postgres:
  image: tensorchord/pgvecto-rs:pg14-
v0.1.11@sha256:0335a1a22f8c5dd1b697f14f079934f5152eaaa216c09b61e293be285491f8ee
environment:
  POSTGRES_PASSWORD: ${DB_PASSWORD}
  POSTGRES_USER: postgres
  POSTGRES_DB: immich
volumes:
  - immich_postgres_data:/var/lib/postgresql/data
networks:
  - immich_default
restart: unless-stopped

```

## Delete duplicates

- Create duplicates database

```

docker container run --rm --volume immich_server_upload:/upload:ro --volume "$PWD:/output/" \
  ghcr.io/agross/immich-duplicates-findimagedupes --prune \

```

```
--fingerprints /output/dupes.db --recurse --no-compare \  
--exclude '\.webp$' /upload/thumbs/<USERID>
```

- Group duplicates

```
docker container run --rm --volume "$PWD:/app/data/" ghcr.io/agross/immich-duplicates-grouper 5
```

- Launch duplicate browser

```
docker container run --env IMMICH_URL=https://immich.domain.com--rm --publish 2222:80  
ghcr.io/agross/immich-duplicates-browser
```