Instrucciones de instalación de programas para servidor local (Docker y Docker Compose) por gouforit.com

ⓘ Información importante

Estas son las notas que hemos tomado en gouforit.com para poder instalar todos estos programas en nuestro Mini PC con Linux Mint. No es una guía exhaustiva de instalación. Debes revisar rutas y otros parámetros de los archivos docker-compose.yml y .env para ajustarlos a tu uso particular. Si tienes dudas, déjanos un comentario en ¿Qué programas instalar en tu servidor local?.

Instalar Docker

https://docs.docker.com/engine/install/ubuntu/

```
# Add Docker's official GPG key:
sudo apt-get update
sudo apt-get install ca-certificates curl
sudo install -m 0755 -d /etc/apt/keyrings
sudo curl -fsSL
https://download.docker.com/linux/ubuntu/gpg -o
/etc/apt/keyrings/docker.asc
sudo chmod a+r /etc/apt/keyrings/docker.asc

# Add the repository to Apt sources:
echo \
  "deb [arch=$(dpkg --print-architecture) signed-
```

```
by=/etc/apt/keyrings/docker.asc]
https://download.docker.com/linux/ubuntu \
  $(. /etc/os-release && echo
"${UBUNTU_CODENAME:-$VERSION_CODENAME}") stable" | \
  sudo tee /etc/apt/sources.list.d/docker.list >
/dev/null
sudo apt-get update
sudo apt-get install docker-ce docker-ce-cli
containerd.io docker-buildx-plugin docker-compose-plugin
sudo systemctl status docker
sudo docker run hello-world
```

Instalamos Portainer (GUI):

```
sudo docker run -d -p 9000:9000 --name portainer --
restart always -v
/var/run/docker.sock:/var/run/docker.sock
portainer/portainer-ce
```

> ⚠ **Ojo**
>
> Importante hacer backup Portainer antes de actualizar desde la GUI.

Actualizamos: https://docs.portainer.io/start/upgrade/docker

```
sudo docker stop portainer
sudo docker rm portainer
docker pull portainer/portainer-ce:lts

docker run -d -p 9000:9000 -p 9443:9443 --name=portainer
```

```
--restart=always -v
/var/run/docker.sock:/var/run/docker.sock -v
portainer_data:/data portainer/portainer-ce:lts
```

Y accedes:

```
http://localhost:9000
```

Instalamos Docker Compose (si no se ha hecho antes):

```
sudo apt-get update
sudo apt-get install docker-compose-plugin
```

Actualizaciones de apps en Docker

En general, actualizamos las apps de Docker con:

```
sudo docker compose pull
sudo docker compose up -d

sudo docker image prune
```

Fundamental

Para que todos los contenedores se reinicien en caso de reinicio fortuito del PC: sudo docker update --restart unless-stopped NOMBRE_DEL_CONTENEDOR

Pasos generales comunes a todas las apps

1. Vas a la carpeta de tu usuario principal en Linux y creas una carpeta llamada docker.
2. Dentro creas una carpeta para cada servicio que quieras instalar.
3. Y dentro de cada carpeta metes un archivo docker-compose.yml (indicado en cada apartado) con la configuración de cada app. Algunas pueden tener también un archivo .env.
4. Desde la terminal, en cada carpeta de cada servicio, tienes que poner el siguiente comando: **docker-compose up -d**
5. Tu servicio empezará a funcionar en tu red local con una url tipo: http://IP_Local:puerto (accedes desde el navegador).

> **Peligro**
>
> Tanto en los archivos docker-compose.yml como en los .env, deberás cambiar algunas cosas como rutas a tus datos, zona horaria, usuario y contraseña... Depende del servicio. Suele ser habitual tener que poner las rutas a tus discos duros para que la app dentro de Docker pueda acceder a ellos.

Navidrome

https://www.navidrome.org/docs/installation/docker/

```
services:
  navidrome:
    image: deluan/navidrome:latest
    user: 1000:1000 # should be owner of volumes
```

```
    ports:
      - "4533:4533"
    restart: unless-stopped
    environment:
      # Optional: put your config options customization
here. Examples:
      ND_LOGLEVEL: info
    volumes:
      - "/path/to/data:/data"
      - "/path/to/your/music/folder:/music:ro"
```

Inicias sesión en Navidrome y creas el usuario. Tienes que cambiar los paths: el primero al disco SSD, el segundo al disco duro externo con música.

Immich

Instalamos Immich (fotos) con https://immich.app/docs/install/portainer y https://immich.app/docs/guides/external-library

Descargamos archivos:

```
wget -O docker-compose.yml https://github.com/immich-
app/immich/releases/latest/download/docker-compose.yml
wget -O .env https://github.com/immich-
app/immich/releases/latest/download/example.env
```

Cambiamos los parámetros de .env:

```
# You can find documentation for all the supported env
variables at
https://docs.immich.app/install/environment-variables

# The location where your uploaded files are stored
UPLOAD_LOCATION=./library

# The location where your database files are stored.
Network shares are not supported for the database
DB_DATA_LOCATION=./postgres

# To set a timezone, uncomment the next line and change
Etc/UTC to a TZ identifier from this list:
https://en.wikipedia.org/wiki/List_of_tz_database_time_z
ones#List
# TZ=Etc/UTC

# The Immich version to use. You can pin this to a
specific version like "v2.1.0"
IMMICH_VERSION=v2

# Connection secret for postgres. You should change it
to a random password
# Please use only the characters `A-Za-z0-9`, without
special characters or spaces
DB_PASSWORD=postgres

# The values below this line do not need to be changed
###################################################
############################
DB_USERNAME=postgres
DB_DATABASE_NAME=immich
```

- Completa UPLOAD_LOCATION con la ubicación preferida para almacenar los activos de copia de seguridad. Debe ser un nuevo directorio en el servidor con suficiente espacio libre.
- Considera cambiar DB_PASSWORD a un valor personalizado. Postgres no está expuesto públicamente, por lo que esta contraseña solo se utiliza para la autenticación local. Para evitar problemas con el análisis de este valor por parte de Docker, es mejor utilizar solo los caracteres A-Za-z0-9.
- Configura tu zona horaria descomentando la línea TZ=.
- Rellena la información de la base de datos personalizada si es necesario.

Backup: https://immich.app/docs/administration/backup-and-restore/

```
docker exec -t immich_postgres pg_dumpall --clean --if-exists --username=<DB_USERNAME> | gzip > "/path/to/backup/dump.sql.gz"
```

Jellyfin

https://jellyfin.org/docs/general/installation/container?method=docker-compose

```
services:
  jellyfin:
    image: jellyfin/jellyfin
    container_name: jellyfin
    user: uid:gid
    network_mode: 'host'
    volumes:
      - /path/to/config:/config
```

```
      - /path/to/cache:/cache
      - type: bind
        source: /path/to/media
        target: /media
      - type: bind
        source: /path/to/media2
        target: /media2
        read_only: true
      # Optional - extra fonts to be used during
transcoding with subtitle burn-in
      - type: bind
        source: /path/to/fonts
        target: /usr/local/share/fonts/custom
        read_only: true
    restart: 'unless-stopped'
    # Optional - alternative address used for
autodiscovery
    environment:
      - JELLYFIN_PublishedServerUrl=http://example.com
    # Optional - may be necessary for docker healthcheck
to pass if running in host network mode
    extra_hosts:
      - 'host.docker.internal:host-gateway'
```

Calibre web

https://hub.docker.com/r/linuxserver/calibre-web

```
---
services:
  calibre-web:
    image: lscr.io/linuxserver/calibre-web:latest
    container_name: calibre-web
```

```
    environment:
      - PUID=1000
      - PGID=1000
      - TZ=Etc/UTC
      - DOCKER_MODS=linuxserver/mods:universal-calibre
#optional
      - OAUTHLIB_RELAX_TOKEN_SCOPE=1 #optional
    volumes:
      - /path/to/calibre-web/data:/config
      - /path/to/calibre/library:/books
    ports:
      - 8083:8083
    restart: unless-stopped
```

Importante

Tienes la ruta mapeada en en /books

-Usuario y password por defecto: admin - admin123

https://docs.linuxserver.io/images/docker-calibre-web/#version-tags

Para cambiar contraseña:

```
sudo docker exec -it calibre-web python3 /app/calibre-
web/cps.py -p /config/app.db -s <user>:<pass>
```

Freshrss

https://docs.linuxserver.io/images/docker-freshrss/#supported-architectures

```
---
services:
  freshrss:
    image: lscr.io/linuxserver/freshrss:latest
    container_name: freshrss
    environment:
      - PUID=1000
      - PGID=1000
      - TZ=Etc/UTC
    volumes:
      - /path/to/freshrss/config:/config
    ports:
      - 80:80
    restart: unless-stopped
```

Note

Para app de iphone: activas la api enadministración, le das una contraseña en perfil de usuario y luego pones contraseña api y url http://tu-ip-local/api/greader.php

Diun

https://crazymax.dev/diun/install/docker/

```
name: diun

services:
  diun:
    image: crazymax/diun:latest
    command: serve
    volumes:
      - "./data:/data"
      - "/var/run/docker.sock:/var/run/docker.sock"
    environment:
      - "TZ=Europe/Paris"
      - "DIUN_WATCH_WORKERS=20"
      - "DIUN_WATCH_SCHEDULE=0 */6 * * *"
      - "DIUN_WATCH_JITTER=30s"
      - "DIUN_PROVIDERS_DOCKER=true"
    labels:
      - "diun.enable=true"
    restart: always
```

Linkwarden

https://docs.linkwarden.app/self-hosting/installation

Creamos directorio y descargamos archivos de configuración.

```
mkdir linkwarden && cd linkwarden
curl -O
https://raw.githubusercontent.com/linkwarden/linkwarden/
refs/heads/main/docker-compose.yml
curl -L
```

```
https://raw.githubusercontent.com/linkwarden/linkwarden/
refs/heads/main/.env.sample -o ".env"
```

Editamos las variables de entorno con nano .env:

```
NEXTAUTH_URL=http://localhost:3000/api/v1/auth
NEXTAUTH_SECRET=VERY_SENSITIVE_SECRET
MEILI_MASTER_KEY=VERY_SENSITIVE_MEILI_MASTER_KEY
POSTGRES_PASSWORD=CUSTOM_POSTGRES_PASSWORD
```

Lo único que DEBES cambiar aquí es NEXTAUTH_SECRET, POSTGRES_PASSWORD y MEILI_MASTER_KEY, todas ellas deben ser frases secretas diferentes. La frase debe estar entre comillas simples o dobles si se utilizan caracteres especiales.

Scrutiny

https://github.com/AnalogJ/scrutiny?tab=readme-ov-file#docker

```
version: '3.5'

services:
  scrutiny:
    container_name: scrutiny
    image: ghcr.io/analogj/scrutiny:master-omnibus
    cap_add:
      - SYS_RAWIO
    ports:
      - "8093:8080" # webapp (cambiado del 8080 al 8081)
      - "8094:8086" # influxDB admin (cambiado del 8086
al 8087)
    volumes:
```

```
      - /run/udev:/run/udev:ro
      - /ruta/scrutiny/config:/opt/scrutiny/config
      - /ruta/scrutiny/influxdb:/opt/scrutiny/influxdb

    devices:
      - "/dev/disk/by-uuid/68BF-EE26"  # Immich
      - "/dev/disk/by-uuid/68B3-1E00"  # Music
    environment:
      - COLLECTOR_CRON_SCHEDULE="0 0 * * *" # Ejecutar
cada noche a medianoche
```

Ejecutar ahora:

```
docker exec scrutiny /opt/scrutiny/bin/scrutiny-
collector-metrics run
```

Searnxg

https://github.com/searxng/searxng-docker

Obtenemos Searxng-docker y sus archivos de configuración con este comando:

```
cd /usr/local
git clone https://github.com/searxng/searxng-docker.git
cd searxng-docker
```

Este es el archivo docker-compose.yaml:

```
version: "3.7"
```

```
services:
  caddy:
    container_name: caddy
    image: docker.io/library/caddy:2-alpine
    network_mode: host
    restart: unless-stopped
    volumes:
      - ./Caddyfile:/etc/caddy/Caddyfile:ro
      - caddy-data:/data:rw
      - caddy-config:/config:rw
    environment:
      - SEARXNG_HOSTNAME=${SEARXNG_HOSTNAME:-
http://localhost}
      - SEARXNG_TLS=${LETSENCRYPT_EMAIL:-internal}
    logging:
      driver: "json-file"
      options:
        max-size: "1m"
        max-file: "1"

  redis:
    container_name: redis
    image: docker.io/valkey/valkey:8-alpine
    command: valkey-server --save 30 1 --loglevel
warning
    restart: unless-stopped
    networks:
      - searxng
    volumes:
      - valkey-data2:/data
    logging:
      driver: "json-file"
      options:
```

```
        max-size: "1m"
        max-file: "1"

  searxng:
    container_name: searxng
    image: docker.io/searxng/searxng:latest
    restart: unless-stopped
    networks:
      - searxng
    ports:
      - "192.168.50.181:8888:8080"
    volumes:
      - ./searxng:/etc/searxng:rw
      - searxng-data:/var/cache/searxng:rw
    environment:
      - SEARXNG_BASE_URL=https://${SEARXNG_HOSTNAME:-
localhost}/
    logging:
      driver: "json-file"
      options:
        max-size: "1m"
        max-file: "1"

networks:
  searxng:

volumes:
  caddy-data:
  caddy-config:
  valkey-data2:
  searxng-data:
```

Generamos secret key `sed -i "s|ultrasecretkey|$(openssl rand -hex 32)|g" searxng/settings.yml`

Editamos searxng/settings.yml

Stirling PDF

https://docs.stirlingpdf.com/Installation/Docker%20Install/

```
version: '3.3'
services:
  stirling-pdf:
    image:
docker.stirlingpdf.com/stirlingtools/stirling-pdf:latest
    ports:
      - '9090:8080'
    volumes:
      - /ruta/a/trainingData:/usr/share/tessdata
      - /ruta/a/extraConfigs:/configs
      - /ruta/a/customFiles:/customFiles/
      - /ruta/a/logs:/logs/
      - /ruta/a/pipeline:/pipeline/
    environment:
      - DISABLE_ADDITIONAL_FEATURES=false
      - LANGS=en_GB
```

Netdata

Instalado directamente Linux mint con:
https://learn.netdata.cloud/docs/netdata-agent/installation/linux

```
wget -O /tmp/netdata-kickstart.sh
https://get.netdata.cloud/kickstart.sh && sh
```

```
/tmp/netdata-kickstart.sh --stable-channel --disable-
telemetry
```

¿Actualizaciones? Con esta orden:

```
wget -O /tmp/netdata-kickstart.sh
https://get.netdata.cloud/kickstart.sh && sh
/tmp/netdata-kickstart.sh
```