

# Recetas de Docker Compose usando Traefik

Archivos para instalación y configuración de aplicaciones contenerizadas usando Docker Compose y Traefik

- [Traefik](#)
- [Bookstack + Traefik](#)
- [Readeck + Traefik](#)
- [NocoDB + Traefik](#)
- [Nginx + Traefik](#)
- [Wordpress + Traefik](#)
- [Nextcloud + Traefik](#)
- [Linkding + Traefik](#)

# Traefik

## Requerimientos previos

- Tener [instalado Docker](#)

## Pasos

1. Crear una red en Docker para Traefik
2. Crear una contraseña para Tablero de control de Traefik `[[`
3. Crear archivos de configuración:
  1. Archivo de Docker Compose: `compose.yaml`
  2. Configuración principal de Traefik: `traefik.yaml`
  3. Configuración dinámica (para middlewares y otras configuraciones): `dynamic.yaml`
4. Levantar contenedor
5. Acceder al tablero de control

## Configuración previa

### Crear una red para Traefik

```
sudo docker network create nombre_de_la_red
```

## Archivos de configuración

### .env

```
MAIN_DOMAIN=ejemplo.com  
VOLUME_PATH=/var/www/dashboard.ejemplo.com
```

### compose.yaml

```
services:  
  traefik:
```

```
image: traefik
container_name: traefik
ports:
  - "80:80"
  - "443:443"
  - "8082:8082"
volumes:
  - "/var/run/docker.sock:/var/run/docker.sock:ro"
  - ./traefik.yaml:/etc/traefik/traefik.yaml:ro
  - ./dynamic.yaml:/etc/traefik/dynamic.yaml:ro
  - traefik-certs:/certs
networks:
  - traefik
restart: always
labels:
  - "traefik.enable=true"
  - "traefik.http.middlewares.traefik-auth.basicauth.removeheader=true"
  - "traefik.http.middlewares.traefik-auth.basicauth.users=user:password"
  - "traefik.http.routers.traefik.rule=Host(`dashboard.${MAIN_DOMAIN}`)"
  - "traefik.http.routers.traefik.service=api@internal"
  - "traefik.http.routers.traefik.tls=true"
  - "traefik.http.routers.traefik.tls.certresolver=tls"
  - "traefik.http.routers.traefik.entrypoints=https"
  - "traefik.http.routers.traefik.middlewares=traefik-
auth,secHeaders@file,autodetectContenttype@file"
healthcheck:
  test: ["CMD", "traefik", "healthcheck", "--ping"]
  interval: 60s
  timeout: 5s
  retries: 3
  start_period: 5s
volumes:
  traefik-certs:
    name: traefik-certs
networks:
  traefik:
    external: true
```

## traefik.yaml

```
providers:
  docker:
    exposedByDefault: false
    endpoint: "unix:///var/run/docker.sock"
    network: traefik
  file:
    filename: /etc/traefik/dynamic.yaml
    watch: true
api:
  dashboard: true
entryPoints:
  http:
    address: :80
  https:
    address: :443
metrics:
  address: :8082
  transport: # Opcional
    respondingTimeouts:
      readTimeout: 60m # Amplia el tiempo antes de desconectar una conexión. Permite la
subida de archivos grandes al servidor.
ping:
  entryPoint: metrics
accessLog:
  filePath: ./logs/access.log
  bufferingSize: 50
certificatesResolvers:
  tls:
    acme:
      email: correo@electronico.com
      storage: /certs/acme.json
      tlsChallenge: {}
      httpChallenge:
        entryPoint: http
```

## dynamic.yaml

```
tls:
  options:
    default:
```

```
minVersion: VersionTLS12
cipherSuites:
  - TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256
  - TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384
  - TLS_ECDHE_RSA_WITH_CHACHA20_POLY1305
  - TLS_AES_128_GCM_SHA256
  - TLS_AES_256_GCM_SHA384
  - TLS_CHACHA20_POLY1305_SHA256
curvePreferences:
  - CurveP521
  - CurveP384
```

```
http:
```

```
  middlewares:
```

```
    # agregar middlewares aquí
```

## Middlewares de ejemplo

Los middlewares se pueden configurar en el segundo archivo de configuración, `dynamic.yaml`, que permite actualizar la configuración en tiempo de ejecución, sin reiniciar el contenedor.

Más información en <https://doc.traefik.io/traefik/v2.5/middlewares/http/headers/>

## Encabezados seguros (secHeaders)

```
http:
```

```
  middlewares:
```

```
    secHeaders:
```

```
      headers:
```

```
        browserXssFilter: true
```

```
        contentTypeNosniff: true
```

```
        frameDeny: true
```

```
        stsIncludeSubdomains: true
```

```
        stsPreload: true
```

```
        stsSeconds: 31536000
```

```
        customFrameOptionsValue: "SAMEORIGIN"
```

```
        customResponseHeaders:
```

```
          server: ""
```

```
          x-powered-by: ""
```

## Política de permisos (permissionsPolicy)

```
http:
  middlewares:
    permissionsPolicy:
      headers:
        permissionsPolicy: "autoplay=(self), fullscreen=(self), picture-in-picture=(self),
screen-wake-lock=(self), sync-xhr=(self), web-share=(self)"
```

## Política de referidos (referrerPolicy)

```
http:
  middlewares:
    referrerPolicy:
      headers:
        referrerPolicy: "no-referrer-when-downgrade"
```

## Autodetección de content-type (autoDetectContentType)

```
http:
  middlewares:
    autoDetectContentType:
      contentType: {}
```

# Bookstack + Traefik

## Requerimientos previos

- Docker instalado
- Instancia de Traefik instalada y corriendo
- Red de Docker creada para integrar contenedor con Traefik
- Requiere 2 [middlewares configurados en Traefik](#):
  - secHeaders@file
  - autoDetectContentType@file

## Pasos

1. Preparar dominio/subdominio de la instancia a configurar
2. Crear secretos para credenciales de base de datos y correo
3. Crear archivos de configuración
4. Cambiar valores de acuerdo a las necesidades de la instancia
5. Levantar el contenedor
6. Visitar la aplicación y comprobar que esté funcionando correctamente

## Configuración previa

### Crear secretos en Docker para acceso a base de datos

```
# Para usuario de base de datos (db_password)
printf "palabra secreta" | docker secret create db_password -

# Para usuario root (db_root_password)
printf "palabra secreta" | docker secret create db_root_password -
```

### Crear secretos en Docker para acceso a SMTP

```
# Para usuario SMTP (mail_password)
printf "palabra secreta" | docker secret create mail_password -
```

# Archivos de configuración

## .env

```
APP_KEY=base64:123456
APP_URL=https://ejemplo.com
DB_DATABASE=bd_ejemplo
DB_HOST=localhost
DB_PORT=7076
INSTANCE_URL=ejemplo.com
MAIL_DRIVER=sendmail
MAIL_FROM_NAME=(Remitente de correo)
MAIL_FROM=usuario@ejemplo.com
MAIL_HOST=smtp.ejemplo.com
MAIL_PORT=465
MAIL_USERNAME=mailer@ejemplo.com
MAIL_ENCRYPTION=ssl
PGID=1000
PUID=1000
TZ=America/Guatemala
VOLUME_PATH=/var/www/ejemplo.com
```

## compose.yaml

En la configuración de `entrypoints` y `certresolver` cambiar de acuerdo a los valores configurados en Traefik.

```
services:
  bookstack_db:
    image: mariadb
    container_name: bookstack_db
    environment:
      - MYSQL_DATABASE=${DB_DATABASE}
      - MYSQL_USER=/run/secrets/db_user
      - MYSQL_PASSWORD=/run/secrets/db_password
      - MYSQL_ROOT_PASSWORD=/run/secrets/db_root_password
    volumes:
      - ${VOLUME_PATH}/bookstack-db:/var/lib/mysql
      - ${VOLUME_PATH}/entrypoint-initdb:/docker-entrypoint-initdb.d:ro
```

```
- ./my.cnf:/etc/mysql/my.cnf
```

```
networks:
```

```
- traefik_network
```

```
restart: unless-stopped
```

```
bookstack_app:
```

```
image: lscr.io/linuxserver/bookstack:latest
```

```
container_name: bookstack_app
```

```
depends_on:
```

```
- bookstack_db
```

```
environment:
```

```
- PUID=1001
```

```
- PGID=1001
```

```
- TZ=${TZ}
```

```
- APP_URL=${APP_URL}
```

```
- APP_KEY=${APP_KEY}
```

```
- APP_DEBUG=false
```

```
- APP_THEME=comunidad
```

```
- DB_HOST=bookstack-db
```

```
- DB_DATABASE=${DB_DATABASE}
```

```
- DB_USERNAME=/run/secrets/db_user
```

```
- DB_PASSWORD=/run/secrets/db_password
```

```
- MAIL_DRIVER=${MAIL_DRIVER}
```

```
- MAIL_FROM_NAME=${MAIL_FROM_NAME}
```

```
- MAIL_FROM=${MAIL_FROM}
```

```
- MAIL_HOST=${MAIL_HOST}
```

```
- MAIL_PORT=${MAIL_PORT}
```

```
- MAIL_USERNAME=${MAIL_USERNAME}
```

```
- MAIL_PASSWORD=/run/secrets/mail_password
```

```
- MAIL_ENCRYPTION=${MAIL_ENCRYPTION}
```

```
labels:
```

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

```
- "traefik.http.routers.nombre_interno_instancia.rule=Host(`${INSTANCE_URL}`)"
```

```
- "traefik.http.routers.nombre_interno_instancia.entrypoints=https"
```

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

```
- "traefik.http.routers.nombre_interno_instancia.tls.certresolver=tls"
```

```
-
```

```
"traefik.http.routers.nombre_interno_instancia.middlewares=secHeaders@file,autodetectContenttype@file"
```

```
- "traefik.http.services.nombre_interno_instancia.loadbalancer.server.port=80"
```

```
- "traefik.http.services.nombre_interno_instancia.loadbalancer.passHostHeader=true"
```

```
volumes:
  - ${VOLUME_PATH}/bookstack-files:/config
restart: unless-stopped
ports:
  - 7075:80
networks:
  - traefik_network
networks:
  traefik_network:
    external: true
```

## my.conf (opcional)

Configuración adicional para reducir el uso de RAM de parte de la base de datos MariaDB

```
[client-server]
socket = /run/mysqld/mysqld.sock

!includedir /etc/mysql/mariadb.conf.d/
!includedir /etc/mysql/conf.d/

max_connections          = 10
connect_timeout         = 5
wait_timeout            = 600
max_allowed_packet      = 16M
thread_cache_size       = 4M
sort_buffer_size        = 32K
bulk_insert_buffer_size = 0
tmp_table_size          = 1K
max_heap_table_size     = 16K
query_cache_limit       = 128K
query_cache_size        = 512K
innodb_buffer_pool_size = 10M
innodb_log_buffer_size  = 512K
```

# Readeck + Traefik

## Requerimientos previos

- Docker instalado
- Instancia de Traefik instalada y corriendo
- Red de Docker creada para conectar contenedor con Traefik
- Opcionalmente, usa un [middleware configurado en Traefik](#):
  - autoDetectContentType@file

## Pasos

1. Preparar dominio/subdominio de la instancia a configurar
2. Crear archivos de configuración
3. Cambiar valores de acuerdo a las necesidades de la instancia
4. Levantar contenedor (docker composer up)
5. Visitar aplicación y comprobar que esté funcionando correctamente

## Archivos de configuración

### .env

```
INSTANCE_PORT=8000
INSTANCE_DOMAIN=ejemplo.com
VOLUME_PATH=/var/www/ejemplo.com
MAIL_HOST=smtp.ejemplo.com
MAIL_PORT=587
MAIL_USERNAME=ejemplo
MAIL_PASSWORD=xyz123
MAIL_ENCRYPTION=TLS
MAIL_FROM=ejemplo@ejemplo.com
MAIL_FROMNOREPLY=ejemplo@ejemplo.com
```

### compose.yaml



En la configuración de `entrypoints` y `certresolver` cambiar de acuerdo a los valores configurados en Traefik.

```
services:
  readeck:
    image: codeberg.org/readeck/readeck:latest
    container_name: readeck
    restart: unless-stopped
    environment:
      - READECK_SERVER_HOST=0.0.0.0
      - READECK_SERVER_PORT=${INSTANCE_PORT}
      - READECK_SERVER_PREFIX=/
      - READECK_ALLOWED_HOSTS=${INSTANCE_DOMAIN}
      - READECK_PUBLIC_SHARE_TTL=2160
      - READECK_MAIL_HOST=${MAIL_HOST}
      - READECK_MAIL_PORT=${MAIL_PORT}
      - READECK_MAIL_USERNAME=${MAIL_USERNAME}
      - READECK_MAIL_PASSWORD=${MAIL_PASSWORD}
      - READECK_MAIL_ENCRYPTION=${MAIL_ENCRYPTION}
      - READECK_MAIL_FROM=${MAIL_FROM}
      - READECK_MAIL_FROMNOREPLY=${MAIL_FROMNOREPLY}
    volumes:
      - ${VOLUME_PATH}/data:/readeck
    networks:
      - traefik_network
    labels:
      - "traefik.enable=true"
      - "traefik.http.routers.readeck.rule=Host(`${INSTANCE_DOMAIN}`)"
      - "traefik.http.routers.readeck.entrypoints=https"
      - "traefik.http.routers.readeck.tls=true"
      - "traefik.http.routers.readeck.tls.certresolver=tls"
      - "traefik.http.routers.readeck.middlewares=autodetectContenttype@file"
      - "traefik.http.services.readeck.loadbalancer.server.port=${INSTANCE_PORT}"
      - "traefik.http.services.readeck.loadbalancer.passHostHeader=true"
    healthcheck:
      test: ["CMD", "/bin/readeck", "healthcheck", "-config", "config.toml"]
      interval: 60s
      timeout: 2s
      retries: 3
  networks:
```

```
traefik_network:  
  external: true
```

# NocoDB + Traefik

## Requerimientos previos

- Docker instalado
- Instancia de Traefik instalada y corriendo
- Red de Docker creada para conectar contenedor con Traefik

## Pasos

1. Preparar dominio/subdominio de la instancia a configurar
2. Crear archivos de configuración
3. Cambiar valores de acuerdo a las necesidades de la instancia
4. Levantar contenedor (docker composer up)
5. Visitar aplicación y comprobar que esté funcionando correctamente

## Archivos de configuración

### .env

```
INSTANCE_URL=ejemplo.com

# Database
DATABASE_NAME=nocodb_bd
DATABASE_USER=nocodb_bd_user
DATABASE_PW=abc123
VOLUME_PATH=/var/www/ejemplo.com
```

### compose.yaml

En la configuración de `entrypoints` y `certresolver` cambiar de acuerdo a los valores configurados en Traefik.

```
services:
  nocodb_app:
```

```
container_name: nocodb_app
depends_on:
  - nocodb_db
environment:
  - "NC_DB=pg://nocodb_db:5432?u=${DATABASE_USER}&p=${DATABASE_PW}&d=${DATABASE_NAME}"
  - "NC_PUBLIC_URL=https://${INSTANCE_URL}"
  - "NC_DISABLE_TELE=true"
image: "nocodb/nocodb:latest"
labels:
  - "traefik.enable=true"
  - "traefik.http.services.nocodb.loadbalancer.server.port=8080"
  - "traefik.http.routers.nocodb.entrypoints=https"
  - "traefik.http.routers.nocodb.tls=true"
  - "traefik.http.routers.nocodb.tls.certresolver=tls"
  - "traefik.http.routers.nocodb.rule=Host(`${INSTANCE_URL}`)"
networks:
  - traefik
restart: always
volumes:
  - "${VOLUME_PATH}/data:/usr/app/data"
nocodb_db:
  container_name: nocodb_db
  environment:
    POSTGRES_DB: "${DATABASE_NAME}"
    POSTGRES_PASSWORD: "${DATABASE_PW}"
    POSTGRES_USER: "${DATABASE_USER}"
  healthcheck:
    interval: 60s
    retries: 10
    test: "pg_isready -U ${DATABASE_USER} -d ${DATABASE_NAME}"
    timeout: 2s
  image: "postgres:12.17-alpine"
  networks:
    - traefik
  restart: always
  volumes:
    - "${VOLUME_PATH}/db:/var/lib/postgresql/data"
networks:
  traefik:
    external: true
```



# Nginx + Traefik

## Requerimientos previos

- Docker instalado
- Instancia de Traefik instalada y corriendo
- Red de Docker creada para conectar contenedor con Traefik
- Opcionalmente, usa un [middleware configurado en Traefik](#):
  - autoDetectContentType@file

## Pasos

1. Preparar dominio/subdominio de la instancia a configurar
2. Crear archivos de configuración
3. Cambiar valores de acuerdo a las necesidades de la instancia
4. Levantar contenedor (docker composer up)
5. Visitar aplicación y comprobar que esté funcionando correctamente

## Archivos de configuración

### compose.yaml

En la configuración de `entrypoints` y `certresolver` cambiar de acuerdo a los valores configurados en Traefik.

```
services:
  web:
    container_name: web
    image: nginx:latest
    volumes:
      - ./files:/usr/share/nginx
      - ./nginx_conf:/etc/nginx/conf.d
    restart: unless-stopped
    networks:
      - traefik
```

labels:

- "traefik.enable=true"
- "traefik.http.routers.web.rule=Host(`ejemplo.com`) || Host(`subdominio.ejemplo.com`)"
- "traefik.http.routers.web.entrypoints=https"
- "traefik.http.routers.web.tls=true"
- "traefik.http.routers.web.tls.certresolver=tls"
- "traefik.http.routers.web.middlewares=autodetectContenttype@file"
- "traefik.http.services.web.loadbalancer.server.port=80"
- "traefik.http.services.web.loadbalancer.passHostHeader=true"

networks:

traefik:

external: true

# Wordpress + Traefik

## Requerimientos previos

- Docker instalado
- Instancia de Traefik instalada y corriendo
- Red de Docker creada para conectar contenedor con Traefik
- Opcionalmente, usa un [middleware configurado en Traefik](#):
  - autoDetectContentType@file
  - permissionsPolicy@file
  - referrerPolicy@file
  - cspWordpress@file
  - secHeaders@file

## Pasos

1. Preparar dominio/subdominio de la instancia a configurar
2. Crear archivos de configuración
3. Cambiar valores de acuerdo a las necesidades de la instancia
4. Levantar contenedor (docker composer up)
5. Visitar aplicación y comprobar que esté funcionando correctamente

## Configuración previa

### Crear secretos en Docker para acceso a base de datos

```
# Creación de secretos para credenciales de base de datos
printf "palabra secreta 1" | docker secret create db_root_password -
printf "palabra secreta 2" | docker secret create db_user -
printf "palabra secreta 3" | docker secret create db_password -
```

## Archivos de configuración

.env

```
INSTANCE_URL=ejemplo.com
DB_NAME=nombre_base_de_datos
DB_TABLE_PREFIX=prefijo_
UID=1000
GID=1000
GID_FILES=33
VOLUME_PATH=./web
```

## compose.yaml

En la configuración de `entrypoints` y `certresolver` cambiar de acuerdo a los valores configurados en Traefik.

```
services:
  bd:
    image: mariadb:latest
    container_name: bd
    volumes:
      - ${VOLUME_PATH}/wordpress-db:/var/lib/mysql
      - ./entrypoint-initdb:/docker-entrypoint-initdb.d:ro
    environment:
      - UID=${UID}
      - GID=${GID}
      - MYSQL_ROOT_PASSWORD=/run/secrets/db_root_password
      - MYSQL_USER=/run/secrets/db_user
      - MYSQL_PASSWORD=/run/secrets/db_password
      - MYSQL_DATABASE=${DB_NAME}
    restart: unless-stopped
    networks:
      - traefik
  wp:
    image: wordpress:latest
    container_name: wp
    volumes:
      - ${VOLUME_PATH}/wordpress-files:/var/www/html
      - ./uploads.ini:/usr/local/etc/php/conf.d/uploads.ini
    environment:
      - UID=${UID}
      - GID=${GID_FILES}
```

```

- WORDPRESS_DB_HOST=bd
- WORDPRESS_DB_NAME=${DB_NAME}
- WORDPRESS_TABLE_PREFIX=${DB_TABLE_PREFIX}
- WORDPRESS_DB_USER=/run/secrets/db_user
- WORDPRESS_DB_PASSWORD=/run/secrets/db_password
depends_on:
  - bd
restart: unless-stopped
labels:
  - "traefik.enable=true"
  - "traefik.http.routers.wordpress-http.rule=Host(`${INSTANCE_URL}`)"
  - "traefik.http.routers.wordpress-http.entrypoints=http"
  - "traefik.http.routers.wordpress-http.middlewares=http-to-https"
  - "traefik.http.middlewares.http-to-https.redirectscheme.scheme=https"
  - "traefik.http.routers.wordpress.rule=Host(`${INSTANCE_URL}`)"
  - "traefik.http.routers.wordpress.entrypoints=https"
  - "traefik.http.routers.wordpress.tls=true"
  -
"traefik.http.routers.wordpress.middlewares=permissionsPolicy@file,referrerPolicy@file,cspWord
press@file,secHeaders@file,autoDetectContentType@file"
  - "traefik.http.routers.wordpress.tls.certresolver=tls"
networks:
  - traefik
networks:
  traefik:
    external: true

```

## uploads.ini (opcional)

Configuración especial para PHP, para permitir subir archivos de hasta 100MB

```

file_uploads = On
memory_limit = 256M
upload_max_filesize = 100M
post_max_size = 100M
max_execution_time = 600

```

## Agregar Redis a la configuración

Para agregar Redis, como objeto de cache, para optimizar la instancia de Wordpress se debe añadir lo siguiente a los archivos de configuración:

## .env

```
REDIS_HOST_PASSWORD=ContraseñaDeRedisAquí
```

## compose.yaml

A la configuración del servicio de WordPress se debe agregar:

```
environment:
  - WORDPRESS_CONFIG_EXTRA=
    define('WP_REDIS_HOST', 'nombre_del_servicio_de_redis');
    define('WP_REDIS_PASSWORD', '${REDIS_HOST_PASSWORD}');
```

y la configuración del servicio de redis sería la siguiente:

```
redis:
  image: redis:alpine
  container_name: redis
  restart: unless-stopped
  ports:
    - '6379:6379'
  command: redis-server --requirepass ${REDIS_HOST_PASSWORD} --maxmemory 128mb --maxmemory-policy allkeys-lru
  volumes:
    - ${VOLUME_PATH}/redis:/var/lib/redis
    - ${VOLUME_PATH}/redis-data:/data
  networks:
    - traefik
```

# Nextcloud + Traefik

## Requerimientos previos

- Docker instalado
- Instancia de Traefik instalada y corriendo
- Red de Docker creada para conectar contenedor con Traefik
- Opcionalmente, usa un [middleware configurado en Traefik](#):
  - autoDetectContentType@file

## Pasos

1. Preparar dominio/subdominio de la instancia a configurar
2. Crear archivos de configuración
  1. Archivo de variables de entorno: `.env`
  2. Archivo para Docker Compose: `compose.yaml`
  3. Archivo de configuración personalizada de MariaDB/MySQL: `my.cnf`
  4. Archivo de configuración personalizada de PHP: `php.ini`
3. Cambiar valores de acuerdo a las necesidades de la instancia
4. Levantar contenedor (`docker composer up`)
5. Visitar aplicación y comprobar que esté funcionando correctamente

## Configuración previa

Por documentar

## Archivos de configuración

### .env

```
INSTANCE_URL=ejemplo.com
APP_CONTAINER_NAME=nextcloud-app
APP_CONTAINER_USER=www-data
MYSQL_ROOT_PASSWORD=abc123456
REDIS_HOST_PASSWORD=abc123456
```

```
MYSQL_DATABASE=nextcloud_db
MYSQL_USER=nextcloud_db_admin
MYSQL_PASSWORD=abc123456
VOLUME_PATH=./data
```

## compose.yaml

En la configuración de `entrypoints` y `certresolver` cambiar de acuerdo a los valores configurados en Traefik.

```
services:
  nextcloud-db:
    image: mariadb:lts
    container_name: nextcloud-db
    command: --transaction-isolation=READ-COMMITTED --log-bin=ROW --skip-innodb-read-only-compressed
    restart: unless-stopped
    volumes:
      - ${VOLUME_PATH}/db:/var/lib/mysql
      - ./my.cnf:/etc/mysql/my.cnf
    environment:
      - MYSQL_ROOT_PASSWORD=${MYSQL_ROOT_PASSWORD}
      - MYSQL_DATABASE=${MYSQL_DATABASE}
      - MYSQL_USER=${MYSQL_USER}
      - MYSQL_PASSWORD=${MYSQL_PASSWORD}
    networks:
      - traefik
  nextcloud-redis:
    image: redis:alpine
    container_name: nextcloud-redis
    hostname: nextcloud-redis
    restart: unless-stopped
    command: redis-server --requirepass ${REDIS_HOST_PASSWORD}
    networks:
      - traefik
  nextcloud-app:
    image: nextcloud
    container_name: nextcloud-app
    restart: unless-stopped
    depends_on:
```

- nextcloud-db
- nextcloud-redis

environment:

- REDIS\_HOST=nextcloud-redis
- REDIS\_HOST\_PASSWORD=\${REDIS\_HOST\_PASSWORD}
- MYSQL\_HOST=nextcloud-db
- MYSQL\_DATABASE=\${MYSQL\_DATABASE}
- MYSQL\_USER=\${MYSQL\_USER}
- MYSQL\_PASSWORD=\${MYSQL\_PASSWORD}
- OVERWRITEPROTOCOL=https
- OVERWRITEHOST=\${INSTANCE\_URL}
- TRUSTED\_PROXIES=127.0.0.1

volumes:

- \${VOLUME\_PATH}/files:/var/www/html
- ./nextcloud.ini:/usr/local/etc/php/conf.d/nextcloud.ini

labels:

- "traefik.enable=true"
- "traefik.http.middlewares.nc-repl.redirectregex.permanent=true"
- "traefik.http.middlewares.nc-repl.redirectregex.regex=https://(.\*)/.well-

known/(? :card|cal) dav"

- "traefik.http.middlewares.nc-repl.redirectregex.replacement=https://\${1}/remote.php/dav"
- "traefik.http.routers.nextcloud-secure.rule=Host(`\${INSTANCE\_URL}`)"
- "traefik.http.routers.nextcloud-secure.entrypoints=https"
- "traefik.http.routers.nextcloud-secure.middlewares=nc-repl@docker,secHeaders@file"
- "traefik.http.routers.nextcloud-secure.tls=true"
- "traefik.http.routers.nextcloud-secure.tls.certresolver=tls"
- "traefik.http.services.nextcloud-secure.loadbalancer.server.port=80"
- "traefik.http.services.nextcloud-secure.loadbalancer.passHostHeader=true"

networks:

- traefik

nextcloud-cron:

container\_name: nextcloud-cron

image: nextcloud

restart: unless-stopped

volumes:

- \${VOLUME\_PATH}/files:/var/www/html

entrypoint: /cron.sh

networks:

- traefik

```
networks:
  traefik:
    external: true
```

## my.cnf

Configuración para reducir el consumo de memoria de la base de datos. Es opcional. Se debe ajustar a las necesidades de la instancia.

```
# MariaDB configuración personalizada
# para instancia pequeña, de uso personal
# y principalmente almacenamiento de archivos

[mysqld]
bulk_insert_buffer_size = 0
connect_timeout         = 10
innodb_buffer_pool_size = 75M
innodb_log_buffer_size = 1M
max_allowed_packet     = 25M
max_connections        = 100
max_heap_table_size    = 16K
query_cache_limit      = 128K
query_cache_size       = 512K
sort_buffer_size       = 10M
thread_cache_size      = 4M
tmp_table_size         = 1K
wait_timeout           = 600
```

## nextcloud.ini

Configuración para ampliar la capacidad de subida de archivos. Es opcional. Se debe ajustar a las necesidades de la instancia.

```
memory_limit=1G
upload_max_filesize=1G
post_max_size=1G
max_input_time=3600
max_execution_time=3600
```

# Scripts útiles

Colocar en la misma carpeta que el archivo compose.yaml y el archivo .env

## occ.sh

```
#!/bin/bash
#
# Ejemplo de uso:
# ./occ.sh app:update --all
#
source .env
sudo docker exec -ti --user $APP_CONTAINER_USER $APP_CONTAINER_NAME /var/www/html/occ $@
```

## fix-permission.sh

```
#!/bin/bash
source .env
sudo docker exec -u root $APP_CONTAINER_NAME sh -c 'chmod 755 /var/www/html/config'
sudo docker exec -u root $APP_CONTAINER_NAME sh -c 'chown -R www-data:www-data
/var/www/html/config'
sudo docker exec -u root $APP_CONTAINER_NAME sh -c 'ls -lsa /var/www/html'
```

# Linkding + Traefik

## Requerimientos previos

- Docker instalado
- Instancia de Traefik instalada y corriendo
- Red de Docker creada para conectar contenedor con Traefik
- Opcionalmente, usa un [middleware configurado en Traefik](#):
  - autoDetectContentType@file

## Pasos

1. Preparar dominio/subdominio de la instancia a configurar
2. Crear archivos de configuración
3. Cambiar valores de acuerdo a las necesidades de la instancia
4. Levantar contenedor (docker composer up)
5. Visitar aplicación y comprobar que esté funcionando correctamente

## Archivos de configuración

### .env

```
INSTANCE_DOMAIN=ejemplo.com
LD_SUPERUSER_NAME=admin
LD_SUPERUSER_PASSWORD=abc1234567
LD_DISABLE_BACKGROUND_TASKS=False
LD_DISABLE_URL_VALIDATION=False
LD_ENABLE_AUTH_PROXY=False
LD_LOG_X_FORWARDED_FOR=true
LD_FAVICON_PROVIDER=https://icons.duckduckgo.com/ip3/{domain}.ico
VOLUME_PATH=./files
```

### compose.yaml

En la configuración de `entrypoints` y `certresolver` cambiar de acuerdo a los valores configurados en Traefik.

```
services:
  linkding:
    container_name: linkding
    image: sissbruecker/linkding:latest
    restart: unless-stopped
    volumes:
      - ${VOLUME_PATH}/data:/etc/linkding/data
    env_file:
      - .env
    networks:
      - traefik
    labels:
      - "traefik.enable=true"
      - "traefik.http.routers.linkding.rule=Host(`ejemplo.com`)"
      - "traefik.http.routers.linkding.entrypoints=https"
      - "traefik.http.routers.linkding.tls=true"
      - "traefik.http.routers.linkding.tls.certresolver=tls"
      - "traefik.http.routers.linkding.middlewares=autodetectContenttype@file"
      - "traefik.http.services.linkding.loadbalancer.server.port=9090"
      - "traefik.http.services.linkding.loadbalancer.passHostHeader=true"
networks:
  traefik:
    external: true
```