

DOCKER

[devops](#), [info](#)

related

- [docker \(altran\)](#) [devops](#), [cursos](#), [docker](#)
- [Docker: SecDevOps](#) [devops](#), [docker](#), [swarm](#), [libros](#), [tech](#)
- [Native Docker clustering with Swarm](#) [libros](#), [tech](#), [docker](#), [swarm](#)
- [MasterClass Docker](#) [docker](#)
- [docker \(first contact\)](#) [docker](#)
- [Taller pentesting en Docker](#) [docker](#)
- [docker swarm: official guide](#) [devops](#), [docker](#), [swarm](#)

+info

- [dockerd](#)
- [Dockerfile](#)
- [docker-compose](#)
- [docker volumes](#)
- [docker TLS \(OLD\)](#)
- [docker daemon TLS](#)
- [docker context](#)
- [docker scan](#)
- Dev Containers: <https://www.youtube.com/watch?v=DkKs29etRis>
- <https://tech.paulcz.net/blog/secure-docker-with-tls/>
- <https://gist.github.com/kekru/974e40bb1cd4b947a53cca5ba4b0bbe5>
- <https://blog.elhacker.net/2024/11/comandos-de-docker-basicos-y-avanzados.html>

take a look

- <https://docs.docker.com/engine/reference/commandline/create/>
- <https://docs.docker.com/storage/bind-mounts/>
- <https://docs.docker.com/compose/compose-file/#volumes>

casos de uso

- [docker-compose casos de uso](#)
- [docker run](#)

herramientas

- [Dockly:](#)

```
docker run -it --rm -v /var/run/docker.sock:/var/run/docker.sock  
lirantal/dockly
```

- [Oxker:](#)

```
docker run --rm -it -v /var/run/docker.sock:/var/run/docker.sock:ro --pull=always mrjackwills/oxker
```

- LazyDocker:

```
docker run --rm -it -v /var/run/docker.sock:/var/run/docker.sock -v /yourpath:/config/jesseduffield/lazydocker lazyteam/lazydocker
```

- Dry:

```
docker run --rm -it -v /var/run/docker.sock:/var/run/docker.sock -e DOCKER_HOST=$DOCKER_HOST moncho/dry
```

/via: <https://atareao.es/podcast/cinco-herramientas-para-gestionar-docker/>

documentación

- [docker-cookbook.pdf](#)

dockerhub ♥

- nextcloud : https://hub.docker.com/_/nextcloud/
- vsftpd : <https://dockerfile.readthedocs.io/en/latest/content/DockerImages/dockerfiles/vsftp.html#environment-variables>
- rtorrent + rutorrent : <https://hub.docker.com/r/diameter/rtorrent-rutorrent/>
- mariadb : https://hub.docker.com/_/mariadb/
- arm : <https://hub.docker.com/u/arm32v7/>
 - <https://medium.freecodecamp.org/the-easy-way-to-set-up-docker-on-a-raspberry-pi-7d24ced073ef>
- autossh : <https://www.everythingcli.org/ssh-tunnelling-for-fun-and-profit-autossh/>
 - <https://github.com/dreamhost/docker-autossh/blob/master/Dockerfile>
- VPN
 - <https://hub.docker.com/r/hwds12/ipsec-vpn-server/>
 - <https://hub.docker.com/r/kylemanna/openvpn/>
 - <https://www.digitalocean.com/community/tutorials/how-to-run-openvpn-in-a-docker-container-on-ubuntu-14-04>
 - https://www.reddit.com/r/docker/comments/7e0gid/docker_with_pia_vpn_for_transmission_and/

resumen comandos

instalación

- añadir usuario actual al grupo de docker y poder usarlo al instante: `sudo usermod -aG docker $(whoami) && newgrp docker`

informativos

- `docker info`
- `docker ps`

- -a : all
- -q : only ID
- -f <filter> : docker -ps -af «status=exited»
- docker search
- docker inspect <id> : vuela JSON con información
 - docker inspect <id> | grep IPAddress

contenedores

- estados de los contenedores
 - **CREATED**
 - → start → **RUNNING**
 - → pause → **PAUSED** → unpause **RUNNING**
 - → stop → **STOPPED** → restart → **RUNNING**
 - → kill → **KILLED**
- docker ps
 - -a
 - -q
- docker run <imagen> [<comando>]
 - -i : interactive
 - -t : tty
 - --rm : contenedor de un solo uso
 - -d : detach (el proceso no se queda «colgado» ejecutando el contenedor)... lanzar contenedor en background
 - -p 80:80: mapea el puerto host:contenedor, en todos los interfaces
 - -p 127.0.0.1:80:80: el contenedor solo será accesible desde 127.0.0.1
 - -P: mapea el puerto en el que está escuchando el contenedor a un puerto aleatorio del equipo
 - --name <nombre_contenedor>
 - docker run -it <container_id> : al salir del contenedor pasa a **STOPPED**
- docker start <contenedor> : arrancar contenedor parado
- docker pause <contenedor>
- docker unpause <contenedor>
- docker stop <contenedor>
- docker kill <contenedor>
- docker exec [-it] <container> [<comando>]
 - bash o sh serían comandos válidos si están instalados en el contenedor
- docker rm <container_id>
 - no se puede eliminar containers en ejecución
 - docker rm \$(docker ps -aq) : elimina todos los contenedores
- docker commit <id_contenedor> [REPOSITORY[:TAG]] : crea imagen de un contenedor

imágenes

- docker pull debian[:<tag>]
- docker images
 - -f o -filter
 - «dangling=true»: filtra lista imágenes no tagueadas
 - «label=<clave>»: filtra por labels (a nivel de imagen, se ven siempre)
 - «label=<clave>=<valor>»: filtra por el contenido de las claves
 - --format «{{.ID}}:{{.Repository}}»: formato de salida (escrito en Go, plantilla)
- docker rmi <imagen_id>: borrar una imagen
- docker tag <imagen_id> <nuevo_nombre> asignar un nombre a una imagen sin taguear, copiar si ya estaba tagueada
- docker save -o <destino> <imagen>:<tag> guarda en <destino> una copia «física» de la imagen

- NO recomendado!
- `docker load -i <imagen_disco>`: importa la imagen
- `docker history <imagen>`: muestra las capas de una imagen y el tamaño de cada una

```
# mostrar (y eliminar) imágenes huérfanas <none>:<none> sin usar
docker image list -q --filter "dangling=true"
docker rmi $(docker image list -q --filter "dangling=true")
docker image rm $(docker image list -q --filter "dangling=true") ??
```

errores

- 12/2022 - **docker: Error response from daemon: cgroups: cgroup mountpoint does not exist: unknown.:**

```
sudo mkdir /sys/fs/cgroup/systemd
sudo mount -t cgroup -o none,name=systemd cgroup /sys/fs/cgroup/systemd
```

- /via: <https://bigdata-etl.com/docker-cgroup-mountpoint-does-not-exist-unknown/>

From:
<https://miguelangel.torresegea.es/wiki/> - **miguel angel torres egea**

Permanent link:
<https://miguelangel.torresegea.es/wiki/tech:docker:start?rev=1734433816>

Last update: **17/12/2024 03:10**

