

# docker

## install

### ubuntu

```
apt-get install -y ca-certificates curl gnupg lsb-release
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --
dearmor -o /usr/share/keyrings/docker-archive-keyring.gpg
echo "deb [arch=$(dpkg --print-architecture) signed-
by=/usr/share/keyrings/docker-archive-keyring.gpg]
https://download.docker.com/linux/ubuntu \
$(lsb_release -cs) stable" | sudo tee
/etc/apt/sources.list.d/docker.list > /dev/null
apt update
apt-get install -y docker-ce docker-ce-cli containerd.io
mkdir /usr/local/lib/docker/cli-plugins -p
curl -SL
https://github.com/docker/compose/releases/download/v2.2.2/docker-compo
se-linux-x86_64 -o /usr/local/lib/docker/cli-plugins/docker-compose
chmod +x /usr/local/lib/docker/cli-plugins/docker-compose
```

run docker as user

```
usermod -aG docker hass
```

/etc/docker/daemon.json

### download

```
{
  "default-address-pools":
  [
    {"base": "172.16.0.0/21", "size": 26}
  ]
}
```

## reset to factory and restart

### download

```
# if swarm
docker swarm leave --force
```

```
systemctl stop docker
rm -fR /var/lib/docker/
ip l delete docker_gwbridge
ip l delete docker0
# and other bridges
systemctl start docker
```

## deploy

get a docker-compose.yml file and edit it

create and run in background

```
docker-compose up -d
```

check

```
docker ps
docker stop <id>
```

autostart

```
docker update --restart unless-stopped <id>
```

apply changes on containers if change docker-compose.yml

```
docker-compose up -d --no-deps --build
```

## docker backup example

```
#!/bin/sh

CONTAINER=52e59999a02b
TARGET=/backup/
SOURCE=/var/lib/docker/volumes/geonode_3x1t-backup-restore/_data/

set -x
rm $SOURCE/2*
docker exec -it $CONTAINER python manage.py backup -f --
config=/usr/src/geonode_3x1t/geonode_3x1t/br/settings_docker.ini --backup-
dir=/backup_restore/
rsync -av $SOURCE $TARGET
```

## create image

- <https://dockerlabs.collabnix.com/beginners/building-your-first-alpine-container.html>

start from a base image, alpine for example

```
docker pull alpine

# create a new container on alpine image
docker run -dit alpine sh

# get container id
docker ps
```

| CONTAINER ID | IMAGE  | COMMAND | CREATED       | STATUS       | PORTS |
|--------------|--------|---------|---------------|--------------|-------|
| 3fdbad8aa816 | alpine | "sh"    | 4 minutes ago | Up 4 minutes |       |
| keen_black   |        |         |               |              |       |

attach to container and copy directory from host to container

```
docker attach 3f
apk update
apk add nss
mkdir /opt
exit

docker cp /opt/sslvpn-plus 3f:/opt # this op stop container
docker start 3f
```

commit container 3f into image

```
docker commit 3f
```

tag image f8

```
docker tag f8 sslvpn-plus-leonardo
```

create container vpn-leonardo from image sslvpn-plus-leonardo

```
docker run --name vpn-leonardo --cap-add=NET_ADMIN --device=/dev/net/tun -
dit sslvpn-plus-leonardo bash
```

## clone source

```
docker pull dryseawind/wildfly14jdk8
```

create Dockerfile

```
FROM dryseawind/wildfly14jdk8
MAINTAINER Stefano Scipioni
RUN /opt/jboss/wildfly-14.0.1.Final/bin/add-user.sh admin admin#password --
silent
CMD ["/opt/jboss/wildfly-14.0.1.Final/bin/standalone.sh", "-c",
"standalone.xml", "-b", "0.0.0.0", "-bmanagement", "0.0.0.0", "--debug"]
```

```
docker build -t "soes/wildfly:14" .
```

## rename volume

```
docker volume create --name <new_volume>
docker run --rm -it -v <old_volume>:/from -v <new_volume>:/to alpine ash -c
"cd /from ; cp -av . /to"
docker volume rm <old_volume>
```

## multi platform build

Multi platform build need a docker registry, local or remote (see below)

create a docker builder or use existing one

```
NAME=builder1
docker buildx use $NAME || docker buildx create --use --name $NAME --driver-
opt network=host
```

build from a Dockerfile and push to local registry

```
REGISTRY=localhost:5000
docker buildx build --platform=linux/amd64,linux/arm64 -t
$REGISTRY/yololab/helloworld --push .
```

## docker registry (private)

create local image and tag with remote registry url

```
# get image from official docker registry
docker pull docker.io/hello-world

# tag
docker tag hello-world docker.csgalileo.org/hello-world
```

create credentials in ~/.docker/config.json

```
docker login docker.csgalileo.org
```

push

```
docker push docker.csgalileo.org/hello-world
```

get remote info with aur/reg arch program

```
reg ls docker.csgalileo.org  
reg digest docker.csgalileo.org/charta/httprest
```

docker registry on localhost

[docker-compose.yml](#)

```
services:  
  registry:  
    restart: no  
    image: registry:2  
    ports:  
      - 5000:5000  
    volumes:  
      - ./data:/var/lib/registry
```

## docker registry (public)

retag local build image to remote image

[download](#)

```
docker tag galileo/trac:1.4.3 scipioit/trac:1.4.3
```

push to docker.io

[download](#)

```
docker push scipioit/trac:1.4.3
```

## remote display via ssh

scenario: connect to remote docker host via ssh and run X11 GUI on client ssh

on docker host sshd edit /etc/ssh/sshd\_config

```
X11Forwarding yes
X11DisplayOffset 10
X11UseLocalhost no
```

on docker host create this helper script

[| set-display.sh](#)

```
XAUTH=/tmp/.Xauthority
HOSTIP=$(ip -o route get to 8.8.8.8 | sed -n 's/.*src
\([0-9.]\+\)\.*/\1/p')

AUTH_COOKIE=$(xauth list ${DISPLAY} | awk '{print $3}')
DISPLAY_NUMBER=$(echo $DISPLAY | cut -d. -f1 | cut -d: -f2)
export DISPLAY=${HOSTIP}:${DISPLAY_NUMBER}
touch $XAUTH
xauth -f $XAUTH add ${DISPLAY} MIT-MAGIC-COOKIE-1 ${AUTH_COOKIE}
```

session from client

```
ssh -X <hostip>
source set-display.sh
docker run --rm -e DISPLAY=$DISPLAY -v $XAUTH:/root/.Xauthority -t
beezu/xeyes
```

## docker swarm

- [docker routing mesh, SNAT](#)

/etc/docker/daemon.json

```
{
  "default-address-pools":
  [
    {"base": "172.16.0.0/21", "size": 26}
  ]
}
```

init

```
docker swarm init --advertise-addr 10.244.0.5 --default-addr-pool
172.16.8.0/21 --default-addr-pool-mask-length 26
```

convert web network scoped "local" to swarm scope

```
docker network rm web
docker network create -d overlay web
```

deploy compose as swarm

```
cd ~/traefik
docker stack deploy -c docker-compose.yml hosting3
```

add worker

```
# on manager get token
docker swarm join-token worker

# on worker add to manager
docker swarm join --token xxxx 172.30.18.94:2377

# on manager check nodes
docker node ls
```

assign label to nodes

```
docker node ls
```

| ID                          | HOSTNAME   | STATUS | AVAILABILITY | MANAGER |
|-----------------------------|------------|--------|--------------|---------|
| n4z6nb6c8xi6el63a6b0vflyv * | ujitsi-dev | Ready  | Active       | Leader  |
| jen361j71yr6k96zrhx7x7b6a   | ujitsireg3 | Ready  | Active       |         |

```
docker node update --label-add type=jibri jen361j71yr6k96zrhx7x7b6a
docker node update --label-add type=jvb n4z6nb6c8xi6el63a6b0vflyv
```

spread the deployment across nodes based on the value of "type" label: one node on type=jvb, one on type=jibri

```
docker service update --placement-pref-add 'spread=node.labels.type'
meet2_jibri
```

check spread

```
docker service ps meet2_jibri |grep Running
```

|              |                     |                             |            |
|--------------|---------------------|-----------------------------|------------|
| rbv77rkxpq1f | meet2_jibri.1       | galileo/jibri:stable-6726-1 | ujitsireg3 |
| Running      | Running 5 hours ago |                             |            |
| jan7bs8ko2c0 | meet2_jibri.2       | galileo/jibri:stable-6726-1 | ujitsi-dev |
| Running      | Running 5 hours ago |                             |            |

add a constraint to other services (meet2jvb, meet2jicofo, ...) to run on specific node

```
docker service update --constraint-add 'node.labels.type == jvb' meet2_jvb
```

show docker services across nodes

```
docker stack ps meet2 |grep Runn
```

show node labels

```
docker node ls -q | xargs docker node inspect -f '{{ .ID }} [{{  
.Description.Hostname }}]: {{ range $k, $v := .Spec.Labels }}{{ $k }}={{ $v  
}} {{end}}'
```

show service status across nodes

```
alias lsd='docker stack ls --format "{{.Name}}" | xargs -n1 docker stack ps  
--format "{{.Node}}\t{{.CurrentState}}\t{{.Name}}\t\t{{.Error}}" -f  
"desired-state=running"'
```

rebalance services across nodes after a node outage

```
alias rebalance='docker service ls --format "{{.ID}}" | xargs -n1 docker  
service update --force'
```

drain a node from containers and migrate to others

```
docker node update --availability drain <node>  
  
# re-enable node after maintenance  
docker node update --availability active <node>
```

get services using a particular network 'web'

```
docker network inspect --verbose web | jq '.[].Services | keys[]'
```

get ip allocation in docker networks

```
docker run -it --rm -v /var/run/docker.sock:/var/run/docker.sock docker/ip-  
util-check
```

From:

<https://wiki.csgalileo.org/> - **Galileo Labs**

Permanent link:

<https://wiki.csgalileo.org/projects/zibaldone/linux/docker>

Last update: **2023/04/19 08:08**

