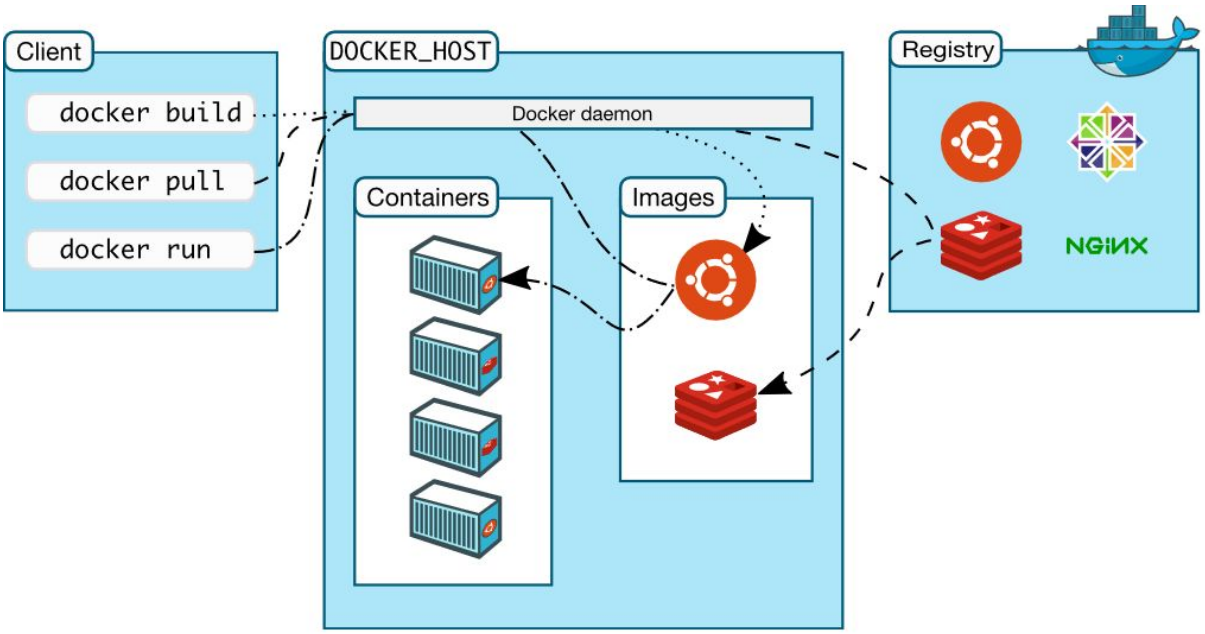
Docker Architecture



daemon: process that runs on a host machine (server)

client: primary Docker interface

image: read-only template (build component)

registry: public or private image repositories (distribution, ship component)

container: created form image, holds everything that is needed for an application to run (run component)

Underlying technologies

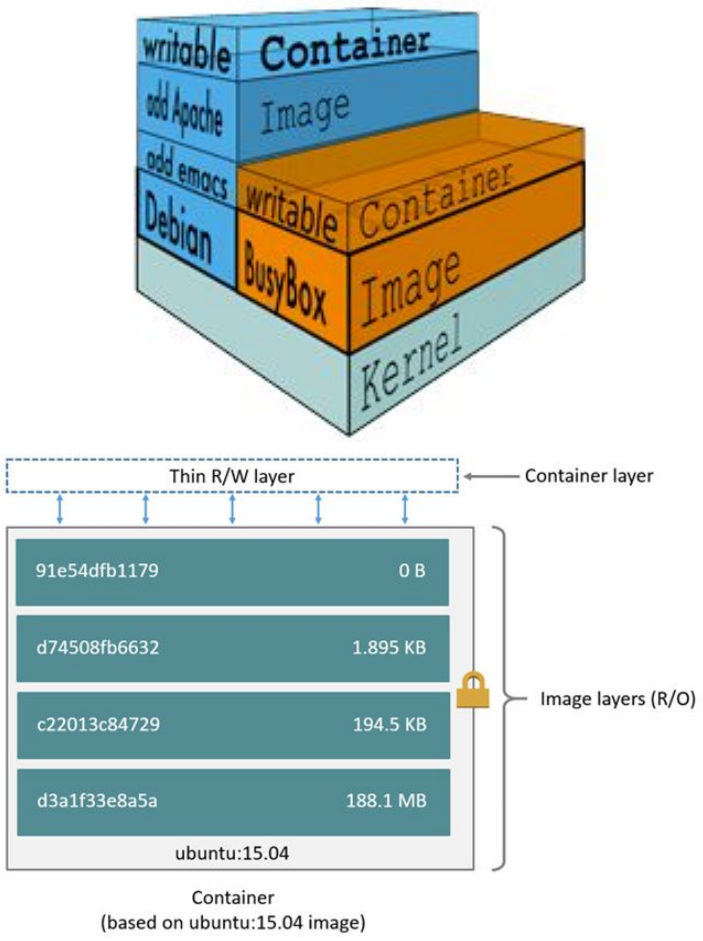
namespaces	Cgroups
pid net mnt ipc uts	memory CPU block I/O network

union file systems	container format
UnionFS AUFS btrfs vfs DeviceMapper	libcontainer LXC

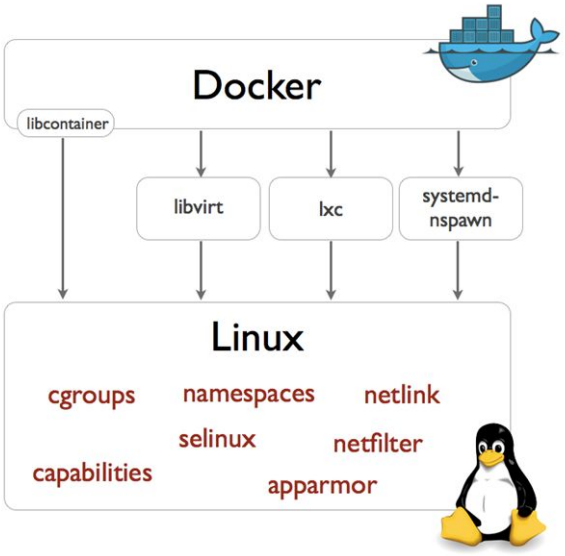
- **namespaces**
  - **pid** namespace: used for process isolation (Process ID)
  - **net** namespace: used for managing network interfaces
  - **mnt** namespace: used for managing mount-points
  - **ipc** namespace: used for managing access to IPC resources (InterProcess Communication)
  - **uts** namespace: used for isolating kernel and version identifiers (Unix Timesharing System)

- **control groups** (cgroups)
  - used for sharing available hardware resources
  - and setting up limits and constraints

- **union file system** (UnionFS)
  - file system that operate by creating layers
  - many layers are merged and visible as one consistent file system
  - many available file systems: **AUFS**, btrfs, vfs, DeviceMapper



- **container format**
  - two supported container formats: **libcontainer**, LXC



Install docker components

docker

```
curl -fsSL get.docker.com -o get-docker.sh
sudo sh get-docker.sh
docker --version
```

docker-compose

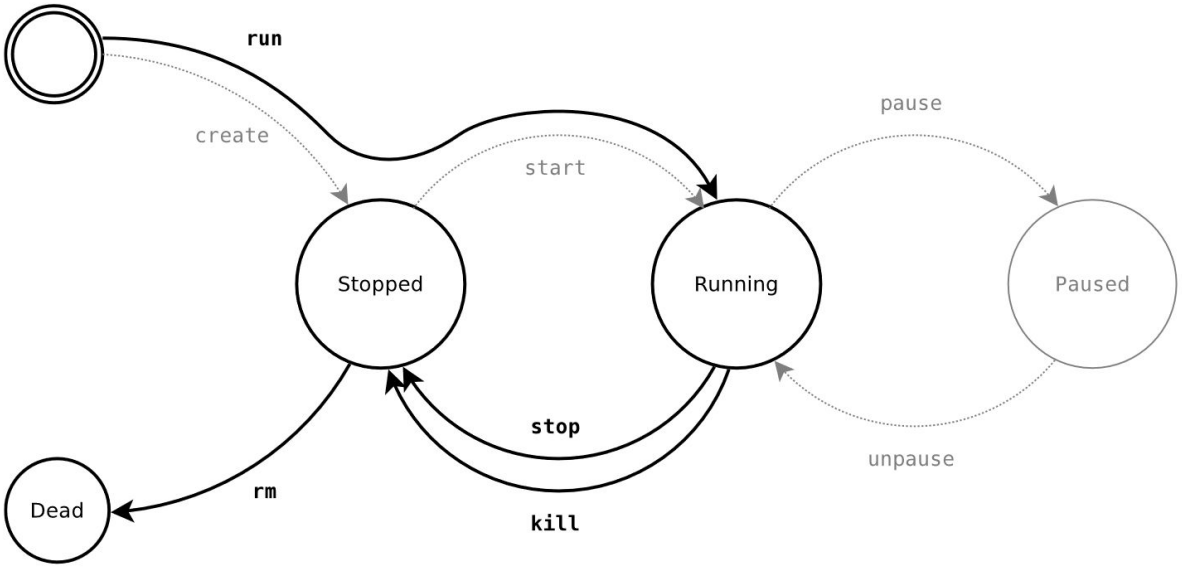
```
sudo curl -L
https://github.com/docker/compose/releases/download/$dockerComposeVersion/doc
ker-compose-`uname -s`-`uname -m` -o /usr/local/bin/docker-compose
sudo chmod +x /usr/local/bin/docker-compose
docker-compose --version
```

Change /$dockerComposeVersion to the newest docker-compose version.

docker-machine

```
curl -L
https://github.com/docker/machine/releases/download/v0.12.2/docker-machine-`u
name -s`-`uname -m` >/tmp/docker-machine &&
chmod +x /tmp/docker-machine &&
sudo cp /tmp/docker-machine /usr/local/bin/docker-machine
```

Docker container lifecycle



docker subcommands

docker attach Attach local standard input, output, and error streams to a running container

docker build Build an image from a Dockerfile

docker commit Create a new image from a container's changes

docker cp Copy files/folders between a container and the local filesystem

docker deploy Deploy a new stack or update an existing stack

docker diff Inspect changes to files or directories on a container's filesystem

docker events Get real time events from the server

docker exec Run a command in a running container

docker export Export a container's filesystem as a tar archive

docker history Show the history of an image

docker images List images

docker import Import the contents from a tarball to create a filesystem image

docker info Display system-wide information

docker inspect Return low-level information on Docker objects

docker kill Kill one or more running containers

docker load Load an image from a tar archive or STDIN

docker login Log in to a Docker registry

docker logout Log out from a Docker registry

docker logs Fetch the logs of a container

docker network Manage networks

docker node Manage Swarm nodes

docker port List port mappings or a specific mapping for the container

docker ps List containers

docker pull Pull an image or a repository from a registry

docker push Push an image or a repository to a registry

docker rename Rename a container

docker rm Remove one or more containers

docker rmi Remove one or more images

docker run Run a command in a new container

docker save Save one or more images to a tar archive (streamed to STDOUT by default)

docker search Search the Docker Hub for images

docker service Manage services

docker stack Manage Docker stacks

docker start Start one or more stopped containers

docker stats Display a live stream of container(s) resource usage statistics

docker stop Stop one or more running containers

docker swarm Manage Swarm

docker system Manage Docker

docker tag Create a tag TARGET_IMAGE that refers to SOURCE_IMAGE

docker top Display the running processes of a container

docker update Update configuration of one or more containers

docker version Show the Docker version information

docker volume Manage volumes

docker wait Block until one or more containers stop, then print their exit codes

Getting help

docker --help list all docker commands

docker command --help list subcommand options

docker run

foreground mode (default) - stdout and stderr are redirected to the terminal, docker run propagates the exit code of the main process

-d the container is run in detached mode

-t allocate pseudo terminal for the container (stdin closed immediately, terminal signals are not forwarded)

-i stdin open, terminal signals forwarded to the container

-itd open stdin, allocate terminal and run process in the background (required for docker attach)

-u user used to tun container

-w working directory

-e setting environment variables

-h container hostname

-l sets labels

-p host_port:container_port publish container port on the host

-P Publish all exposed ports to random ports (30000-32767)

-**v /path** creates random name volume and attach it to the container path

volume_name:/container_path create named volume and attach it to the container path

host_path:/container_path mounts host directory in container path

--entrypoint overwrite entrypoint defined in Dockerfile

--log-driver logging driver for the container

--log-opt log driver options

--name assign name to container (by default a random name is generated → adjective name)

--network connect a container to a network

--rm automatically removes container after exit

--network attach docker interface to the specified network (by default it connect container to the bridge network)

docker build

-f used for custom dockerfile names

-t tags image after build

--**build-arg** set build-time variables

--rm deletes intermediate containers after successful build

docker rm

-f force removal of a running container

-v remove the volumes associated with the container

docker network

connect	connect a container to a network
create	create a network
disconnect	disconnect a container from a network
inspect	display detailed information about the network
ls	list networks
prune	remove all unused networks
rm	remove one or more networks

docker volume

create	Create a volume
inspect	Display detailed information on one or more volumes
ls	List volumes
prune	Remove all unused volumes

rm	Remove one or more volumes

docker bulk commands

docker rm $(docker ps -q) - delete all running containers containers

docker rm $(docker ps -qa) - delete all containers

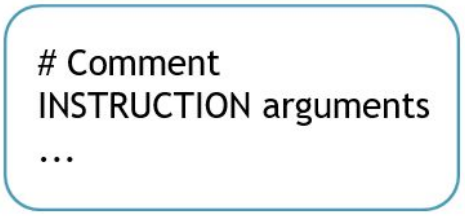
docker start/stop $(docker ps -qa) - start/stops all containers

docker kill $(docker ps -q) - kill all running containers

docker rmi $(docker images) - delete all docker images

Dockerfile instructions

Dockerfile syntax:



FROM – sets the base image (required)

COPY – copy files into containers

ADD – unpack tar archives, copy files and remote file URLs into containers

RUN – execute any command in containers and creates new fs layer

SHELL - sets shell in which commands are run

CMD – provides defaults for running containers

ENTRYPOINT - configure container that will run as executable

ENV – sets environment variables

EXPOSE – sets port on which containers will be listening

USER – sets the username

VOLUME – mounts volume from HostOS

MAINTAINER – sets Author of the image (deprecated)

LABEL – adds metadata to the image

ARG - defined variables passed during build-time

ONBUILD - adds trigger used when the image is used as the base image

WORKDIR - sets the working directory for any RUN, CMD, ENTRYPOINT, COPY and ADD instructions that follow it in the Dockerfile

STOPSIGNAL – sets the system call signal that will be sent to the container to exit

HEALTHCHECK – sets container healthchecks

docker-compose

build	Build or rebuild services
config	Validate and view the compose file
create	Create services
down	Stop and remove containers, networks, images, and volumes
exec	Execute a command in a running container
help	Get help on a command
kill	Kill containers
port	Print the public port for a port binding
ps	List containers
rm	Remove stopped containers
run	Run a one-off command
scale	Set number of containers for a service
start	Start services
stop	Stop services
up	Create and start containers
version	Show the Docker-Compose version information

docker-machine

active	Print which machine is active
config	Print the connection config for machine
create	Create a machine
env	Display the commands to set up the environment for the Docker client
inspect	Inspect information about a machine
ip	Get the IP address of a machine
kill	Kill a machine
ls	List machines
provision	Re-provision existing machines
regenerate-certs	Regenerate TLS Certificates for a machine
restart	Restart a machine
rm	Remove a machine
ssh	Log into or run a command on a machine with SSH.
scp	Copy files between machines
start	Start a machine
url	Get the URL of a machine
version	Show the Docker Machine version or a machine docker version
help	Shows a list of commands or help for one command

eval $(docker-machine env machine_name) activates machine (points docker client to docker machine)

docker-swarm

init	Initialize a swarm
join	Join a swarm as a node and/or manager
leave	Leave the swarm

docker node

demote	demote one or more nodes from manager in the swarm
inspect	display detailed information about the node
ls	list swarm nodes
promote	promote nodes in the swarm
ps	list tasks running on the nodes
rm	remove node from the swarm

docker service

create	create new service
inspect	display detailed information about the service
logs	fetch the logs of a service or task
ls	list services
ps	list tasks
rm	remove service
scale	scale one or multiple replicated services