# Container

A container is a set of isolated processes and resources. Linux achieves

this by using namespaces, which allows processes to access only resources

in that particular namespace, which allows having a process tree means set

of processes that is completely independent of the rest of the systems processes.

# Docker definition:

A container is a standard unit of software that packages

up code and all its dependencies so the application runs quickly and reliably

from one computing environment to another.

# Docker

Docker is one of the tools that used the idea of the isolated resources to

create a container that allows applications to be packaged with all the

dependencies installed and ran wherever we wanted.

Docker can only run on Linux machines this means I cant install Dokcer directly on Windows or any other OS.

If I want install Docker on windows then I need to run a Linux VM in windows on top that I need to run Docker.

# Virtualization (VM)

- VM is way of running virtual OS on top a host OS using a special software called Hyperviser.

- VM directly shares the harware of the host OS.

**VM**  vs **Containerisation**

1. Virtualization at hardware level 1. Virtualization at OS level

2. Heavyweight - consume more host 2. Lightweight

resources

3. VM useses hypervisor 3. containerisation tool is used

4. limited performace - Boot up time 4. Native performace - usualy boot

is more which is in minutes fast in seconds.

5. Cosumes more storage 5. Shres OS storage means only uses

required storage.

6. Supports all OS 6. Supports on Linux

# host machine

This is the machine in which docker is running

# Docker images

List images in a machine

docker images

# To pull / download docker image

docker pull <image\_name>:<tag\_name>

# Note:

if we wont specify tag the by default latest will be considered.

To delete docker image

docker rmi <image\_name>:<tag\_name>

(OR)

docker rmi <image\_id>

To delete all the images

docker rmi $(docker images -q)

To tag a docker image

docker tag <old\_image> <new\_image>

ex: docker tag ubuntu:latest my\_ubuntu:1.0

To delete dangling images / to delete all unwanted images / images which are not used

docker image prune

Docker containers

To create/run a container from image

docker run -it -d --name <container\_name> <image\_name>

-it - Interactive Terminal (tty)

-d - deatached mode (when ever we create a container it will auto login to avoid this

we can create a container in detached mode)

--name used to provide user defined conatainer name

To list the running containers

docker ps

(OR)

docker container ls

To delete a stop container

docker rm <conatainer\_name>

(OR)

docekr rm <conatainer\_id>

To delete a running container

Forceful deletion

docker rm -f <conatainer\_id>

Graceful deletion

docker rm $(docker stop <conatainer\_id>)

To list all the containers (running and stopped)

docker ps -a

To list all stopped conatainers

docker ps -a --filter status=exited

To delete all stopped container

docker rm $(docker ps -aq --filter status=exited)

To check the logs of conatainers

docker logs <container\_id>

To run a command inside a conatainer

docker exec -it <container\_id> <command>

To login / get inside a containre

docker attach <container\_id>

Assignment: work with docker commands

Try to create a jenkins container (jenkins/jenkins:lts)

# Custom Docker Image / Dockerfile

- Dockerfile is used to create custom image on top stock image or any other image as base image.

FROM

- FROM must be the first non-command instruction in the Dockerfile.

- FROM is used to specify the base image on top of this image all the next instructions will be executed.

FROM <image\_name>:<tag>

COPY and ADD

- Both copy and add instruction is used to Pcopy files and directories from host machine to the image.

- The source path to copy files should always be evaluted with reference to Dockerfile.

ADD supports extra source formats

- If the source is a compressed file add will automatically

uncompresses it to the destination.

- If the source is a link to a downloadable file it will download

to the destination.

CMD and ENTRYPOINT

shell format

CMD "ls -lrt"

ENTRYPOINT "ls -lrt"

EXEC Format

CMD ["ls","-l","-rt"]

ENTRYPOINT ["ls","-lrt"]

- Both CMD and ENTRYPOINT are used to define the execution command of the container which will be created

from this image.

- If we use multiple CMD or ENTRYPOINT in the same Dockerfile only the latest one will be considered

and all the other CMD or ENTRYPOINT will be ignored.

- If we use both CMD and ENTRYPOINT in the same Dockerfile, ENTRYPOINT will get the

higest priority and the command of CMD will become as argumetns to ENTRYPOINT

Difference

- CMD command can be overridden at the runtime.

- ENTRYPOINT can't be overridden at the runtime but the runtime command

will become parameters to ENTRYPOINT command.

Note: Q. Can we override ENTRYPOINT

Yes, after docker 1.6 version docker has given option to over

Entrypoint command at the runtime using --entrypoint

ENV

- This instruction is used to set the environment variable inside the container.

ENV <variable\_name> <value>

ENV <variable\_name>=<value>

multiple

ENV <variable\_name>=<value> <variable\_name>=<value> <variable\_name>=<value> ....

To create environment variables at run time

- using -e or --env option at the runtime we can create env variables

- For multiple variables use multiple -e

ex: docker run .... -e <variable\_name>=<value> -e <variable\_name>=<value> ....

The best way to load multiple env variable is using env file

using --env-file <file\_path> at the runtime (with docker run command) we can

load the env file containing n number variables.

ARG

Using ARG we can pass parameters to Dockerfile as user inputs

ARG <var\_name>=<default\_value>

To pass the value at build time

docker build --build-arg <var\_name>=<value> ....

WORKDIR

This is used to set the working directory for all the instructions that follows it

such as RUN, CMD, ENTRYPOINT, COPY, ADD

WORKDIR <path>

EXPOSE

- used to expose a port to the docker network

- All conatainers in the same newtwork will have access to exposed port.

EXPOSE <prot\_number>

# Docker Volumes

- As the layers inside the image are readonly which means once the image is created

we cannot change/edit so we cannot put the conatainer data in image.

- Container create a top most RW layer and all the runtime data is saved here.

- Container layer is temparary layer, If we loose the container we loose data. so

to retain/persist the container runtime data we need docker volumes.

Bind Mounts

- we can mount host machine filesystem (files and directories) to the container

docker run -v <host\_path>:<container\_path>

Docker Volumes

- These are docker managed filesystem and we use docker commands to manage these

volumes

- Volumes are easier to manage, backup or migrate than bind mounts.

- Volumes supports many drivers which means we can maunt many types of filesystem.

- Default location of docker volume is /var/lib/docker/volumes

docker run -v <volume\_name>:<container\_path>

To create volume

docker volume create <volume\_name>

To list volume

docker volume ls

To Delete volume

docker volume rm <volume\_name>

Assignment: How to save the content of container to a image.

Namespaces and cgroups

- Docker uses linux namespaces to provide isolated workspace for processes called

conatainer

- when a container is created, docker creates a set of namespaces for it and provides

a layer of isolation for container.

- Each container run in a different namespace

namespace (To list - lsns)

cgroups

- Linux OS uses cgroups to manage the availale hardware resoruces such as

cpu, RAM, Disk, I/O.

- we can control tje access and also we can apply the limitations

To list - lscgroup

pid - namespace for managing processes (process isolation)

user - namespace for non-root user on linux.

uts - namespace for unix timesharing system which isolates kernel and version identifiers,

time bonding of process.

ipc - (interprocess communication) namespace for managing the process communication.

mnt - namespace for managing filesystem mounts.

net - namespace for managing network interfaces.

# Docker networking

Publish

PUBLISH = Expose + outside world port mapping

- publics will bind the container port to the host port which we can access from

outside world using <host\_ip>:<port\_mapped>

- To publish a port

docker run -p <host\_port>:<container\_port> .....

-P publish\_all

It binds all the exposed ports of the container to host machine port

To map direct IP address to the host

port to port

ip:<host\_port>:<container\_port>

ip::<container\_port>

Range of ports

many to one

-p 8080-8090:8080

many to many

-p 8080-8085:8086-8091

- The total number of host ports in range should be same as the

total number of container ports range.

# Docker network types

1. Bridge

- This is a private internal network created by docker on the host machine

by name docker0

- This is the default network type for all the container which are created

without any network configurations.

- By default all the containers in the same bridge can communicate with

eachother without any extra configuration.

- We cannot use container name for communication only IP address is allowed in

default bridge.

Custom bridge

To create bridge network

docker network create --driver bridge my\_bride

- In custom bridge containers can communicate with eachother with container

name and also with IP address.

Assignment: 1) docker export vs save vs commit

2) How to have communication between containers which are in

two different bridge network.

3) Docker Architecture

2. Host

- This driver removes the network isolation between docker and the host.

- The containers are directly connected to host machine network without

extra layer of any docker network.

- Shares the same TCP.IP stack and same namespace of host machine.

- All the network interfaces which are there in host machine are

accessable by this container.

3. None

- Containers are not attached to any network by docker.

- All the required network configurations need to be done

manually.

- The host or any other containers won't be able to communicate

with this container untill a custom network is configured.