**=====================================================================**

**What is docker :**

Docker is a platform for developing shipping and running applications using container Virtualization technology . it creates lightweight vm’s

The platform consists of multiple and products/tools

* Docker engine
* Docker hub
* Docker machine
* Docker swarm
* Docker compose
* Kitematics

**=====================================================================**

**Containers** based virtualization uses the kernel on the host’s os to run multiple guest instances .

Each guest instances is called container .

Each container has it’s own root filesystem

Processes

Memory

Devices

Network ports

By using containers we can isolate the runtime environment for applications .

Docker engine is the program that enables containers to be built shipped run . it is also referred to as docker daemon .So we install the docker engine to host and docker engine uses linux kernel to create and manage containers .

It uses linux kernel features namespace and control groups . Namespaces gives us isolated workspace that we called a container .

**Creating a container**

=============================================================

.docker run [option] [image] [command] [args]

.docker run ubuntu :14.04 echo “hello world “

.docker run ubuntu:14.04 ps qx

docker run -i -t ubuntu:14.04 /bin/bash

docker run -d ubuntu:14.04 /bin/bash

.docker run -d -P tomcat:7

. docker run -it ubuntu:14.04 /bin/bash

Make changes install packages

* Like install curl
* Then commit and make new images

Docker commit [container ID] laxman/pack:1.0

This the one way that we can build new images by committing the changes in the container

**Dockerfile**

----------------------------------------------------------------------------------------------------------------------------

A configuration file that contains instructions for building an image.

More effective way then docker commit .

FROM ubuntu:14.04

RUN apt-get install vim

RUN qpt-get install curl

Each run instruction will execute the command on the top writable layer and perform a commit of the image . so if we will run 10 run instruction in dockerfile it will be 10 commit

RUN apt-get update && apt-get install -y \

curl \

vim\

openjdk-7-jdk

DOCKER BUILD

.syntax

. docker build [options] [path] ⇐ build context

. docker build -t laxman/myimage:1.0 [path]

CMD instruction

. it defines the default command to execute when a container is created

. it performs no action when image is build .

. can only be specified once in the dockerfile

.can be overridden over run time

Shell form

CMD ping 127.0.0.1 -c 30

Exec form

CMD [“ping” , “127.0.0.1”,”-c”,”50”]

ENTRYPOINT instructions

ENTRYPOINT [“ping”]

Here we have to provide arguments .

DOCKER STOP START

.docker stop containerID

.docker start container ID

GETTING TERMINAL ACCESS

Use docker exec command to start another process within a container

. docker exec -i -t [ containerID] /bin/bash

DELETING CONTAINERS

Can only delete containers that have been stopped

.docker rm

Docker push

. docker push [repo:tag]

TAGGING IMAGES

.docker tag containerID tagname

**VOLUMES**

It is a designed directory in a container , which is designed to persist data

Independent of the container lifecycle .Volumes are stored in the filesystem of the host running Docker.

Volume changes are excluded when updating an image

Persist when a container is deleted

Can be mapped to a host folder

Can be shared b/w containers

MOUNT A VOLUME

Volumes are mounted when creating and executing a container

Can be mapped to host dir

. docker run -d -P -v /myvolume nginx:1.7

. docker run -i -t -v /data/src : /test/src nginx:1.7

**Sharing volumes b/w containers**

.docker run -it --name john1 -v /john1 busybox

.docker run -it --name john2 --volumes-from john1 busybox

**Volumes in Dockerfile**

VOLUME instruction create a mount point

String example

VOLUME /myvolume

String example with multiple volume

VOLUME /www/website /www/website2

Json

VOLUME [“myvolume1” , “myvolume2”]

MAPPING PORTS

. p and P

Docker run -d -p 8080:80 ngnix:1.7

**Linking**

Is a communication method b/w containers which allows them securely transfer data from one to another .

CREATE THE SOURCE CONTAINER FIRST

Docker run -d --name dbms postgres

CREATE THE RECIPIENT CONTAINER AND USE THE --link option

Docker run -d -P --name website --link dbms :db ubuntu 14.04 bash

**Docker Operations**

Container troubleshooting

Docker logs <container name >

.docker logs -f <container name>

.docker logs -f -tail 1 <name>

.docker run -d -P -v /nginxlogs : /var/logs/ngnix ngnix

Here we can check the application logs

**Inspecting a container**

Display all the details of the specified containers

.docker inspect <container name>

.docker inspect <container name> | grep IPAddress

Starting and stopping docker demon

**DOCKER MACHINE**

Docker machine is a tool that automatically provisions docker hosts and install a docker engine on them .Machine creates the server , install Docker and configure the docker client .

**DOCKER COMPOSE**

Is a tool for creating and managing multi container applications Containers are defined in a single file called docker-compose.yml Each container run a particular component

/ service of your application

Ex :

* Web front end
* User authentication
* Payments
* database

**Some important commands**

.docker kill $(docker ps -q)

.docker rm $(docker ps -a)

.docker logs -h

.

Dockerfiles

----------------------------------

FROM ubuntu:14.04

RUN apt-get -y install apache2

CMD[“/usr/sbin/apache2ctl”, “-D” , “FOREGROUND” ]

-----------------------------------

.docker build --no-cache=true -f apache-dockerfile-ex1 -t apache-ex1 .

-----------------------------------

.FROM ubuntu:14.04

RUN \

Apt-get update && \

Apt-get -y install apache2

ADD index.html /var/www/html/index.html

EPOSE 80

------------------------------------