**Docker**

* **What is docker?**

Docker is a container management. Tool

IT is a set of platform as a service. Products that use OS-level virtualization to deliver software in packages called containers.

Containers are isolated from one another and bundle their own software, libraries and configuration files; they can communicate with each other through well-defined

* **What is virtualization?**

Virtualization means run multiple OS on single physical system.

* In virtualization every vm is independent
* To set up a vm hypervisor is required
* Every vm has different credentials
* For running an application we need a VM.
* If we setup multiple VM’s for an application. We need multiple isolation VM’s space is wasted. And high cost
* **What is container management?**

Container management is a thin virtualization (only application libbers can have remaining are taken from host OS) here containers means vm. Container has only one application, it will take the required files

* Host os, it will use the kernel of host os.
* There is no OS in container. Only liberies and binary files remaining taken from host os
* It will run only application related files.

**Docker images are stored in hub.docker.com**

* It does not have any os
* Fast performing
* Self-sustained (all necessary dependencies)
* It has layered architecture.
* All advantages of virtualization.
* Easy to scale.

**Note:** Vm is for multiple application because all are available.

* Container is for only one application

**VIRTUALIZATION DOCKER**

Isolation OS isolation container

Space is wasted space is not wasted

Every file is installed required bin &lib files only installed, remaining files shares from host OS.

Take time to boot take very less time to boot

Take time to restart vm take very less time to restart container.

* **How to setup these containers?**

Normal iOS----->vm

Vagrant box ------> vm

Aws AMI ----------> instance

Docker images -----> container

* **Docker install & configure:**

**Bourse**: download docker for Linux

Set up the repository commands

Docker Ce install commands

* **COMMANDS**

#Systemctl start docker

#systemctl status docker

**Docker images to download from**

#docker pull image name (Ubuntu)

#docker images

#docker run -ti --rm Ubuntu /bin/bash - to run the new container

Ti= terminal interact, rm=remove

#exit - exit from container

#docker ps =To check running containers

#docker ps -a = To display stopped containers

#docker start container id

#docker inspect container id = display properties this container used in host machine

#docker run -ti --name 'container1" --hostname "disney.com" -m 10m --memory-swap 20m ubuntu /bin/bash = (assign hostname and memory size)

#docker stop container ID

#docker pause container ID

#docker unpause container ID

#docker rm container ID

#docker stop $(docker ps -a -q) multiple dockers stop at a time

#docker rm $(docker ps -a -q) multiple dockers at a time delete

* **Docker images**
* A **Docker image** is a file, comprised of multiple layers, that is used to execute code in a **Docker container**. ...
* When the **Docker** user runs an **image**, it can become one or multiple instances of that **container**. **Docker** is an open source OS-level virtualization software platform primarily designed for Linux.
* **write docker images**

**#mkdir dir1**

**#cd dir1**

**#vi dockerfile**

From java: latest

MAINTAINER ramki info@google.com

LABEL env=production {label}

ENV apparea /data/app {environment variables}

Run mkdir-p $apparea {while start the image this directory is created}

ADD ./gitbucket.war $apparea {add the file to this directory}

WORKDIR $apparea

CMD ["java","-jar","gitbucket.war"] {run the gitbucket at the container start}

Save exit

#ls

**After create the docker file build that file as image**

#sudo docker build -t dir1/gitbucket .

#sudo docker images

To run the gitbucket image

#sudo docker run -d -p 80:8080 dir1/gitbucket

#sudo docker ps

#ip a

To login to the running container

#sudo docker exec -ti container id /bin/bash -- exec=to login to the running container

#ls

#ip a

#exit

**Write Jenkins images**

#mkdir dir1

#cd dir1

#vi dockerfile

From java: latest

MAINTAINER ramki info@google.com

LABEL env=production {label}

ENV apparea /data/app {environment variables}

Run mkdir-p $apparea {while start the image this directory is created}

ADD ./jenkins.war $apparea {add the file to this directory}

WORKDIR $apparea

CMD ["java","-jar","jenkins.war"] {run the gitbucket at the container start}

Save exit

#ls

**After create the docker file build that file as image**

#sudo docker build -t dir1/jenkins .

#sudo docker images

**To run the Jenkins image**

#sudo docker run -d -p 8080:8080 dir1/gitbucket

#sudo docker ps

#ip a

To login to the running container

#sudo docker exec -ti container id /bin/bash –exec = to login to the running container

#ls

#exit

* **volume mapping in docker**

Local system ---------> container data

If you delete the container the data will be stored in local system.

#sudo docker run -d -v local system (/home/ramki/gitbucket\_data/):/root/.gitbucket -p 8080:8080 dir1/gitbucket

**Configuration and installation**

$ sudo yum install -y yum-utils

$ sudo yum-config-manager \

--add-repo \

https://download.docker.com/linux/centos/docker-ce.repo

31 sudo yum-config-manager --enable docker-ce-nightly

32 sudo yum-config-manager --enable docker-ce-test

33 sudo yum-config-manager --disable docker-ce-nightly

34 sudo yum install docker-ce -y

35 systemctl start docker

36 systemctl status docker

37 docker images

38 docker pull ubuntu

39 docker images

40 history

66 docker run -ti ubuntu /bin/bash

67 docker ps

68 docker start

69 docker start ee0db03715a4

70 docker ps

71 docker ps -a

72 docker inspect ee0db03715a4

73 docker ps

74 docker run -ti --name "container1" --hostname "centos" ubuntu /bin/bash

75 docker ps

76 docker ps -a

77 docker inspect ce945c5cfdf7

78 docker inspect 5831cba01f09

79 docker rm 5831cba01f09

80 docker ps -a

create own docker images

mkdir my\_dockeriamge

cd my\_dockeriamge/

vi Dockerfile

From java:latest

MAINTAINER ramki info@google.com

LABEL env=production

ENV apparea /data/app

Run mkdir -p $apparea

ADD ./gitbucket.war $apparea

WORKDIR $apparea

CMD ["java","-jar","gitbucket.war"]

:wq!

ls

sudo docker build -t techmine/gitbucket .

sudo docker images