What is a container

- Traditionally software applications are deployed following a monolithic architecture approach where each components are dependent or locked each other. The major drawback of this approach is lack of reliability , scailability and high availability. Also traditional software deployment will be in the life cycle of test,dev,production environment which is a time consuming task.

- Containers are a new generation of virtual machines. That brings software development to a new level . Containers are an isolated set of different rules and resources inside a single operating system. This means containers can provide benefits as virtual machines but use less CPU,memory and storage. There are several popular containers like LXC,Rocket,Docker.

Container features and advantages

Some of the major advantages of containers are as follows:

- **Efficient hardware resource consumption** - if we are using normal virtualisation method to host our applications in a VM environment we may need OS to be installed in each VM to host different applications and sometime we can observed that major part of system resources are utilised by OS where the applications used to face lac of CPU and memory which will impact the performance. But in container technology we have OS installed in only in one system and different containers are installed on top of that .

- **Application and service isolation -** imagine if we have 10 applications hosted in a single server ,each applications has number of dependencies (like packages,libraries etc). If we need to update the packages we need to update all of it's dependencies which may impact another applications also. In container technology each application is isolated to each other and application update is an easy process.

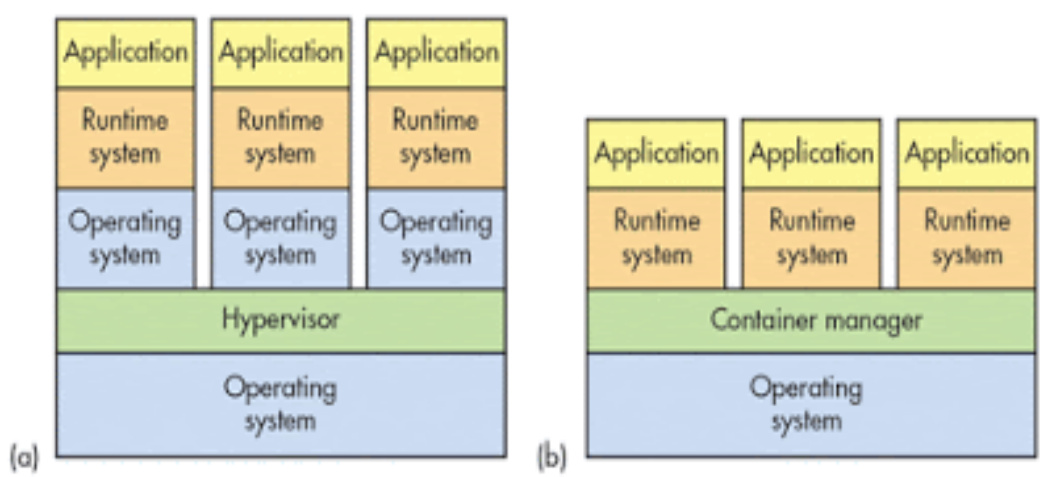
- **Faster deployment -** Using container images we can speed up the deployment . To restart applications in containers will take only some seconds where restarting the VM's will take minutes.The main reason is container does not need to restart an OS while restarting containers.

- **Micro services architecture -** Containers bring the application deployment to a new level called micro services architecture where our applications are parted to different pieces and installing n different containers. For example if our application contains a webserver and a database server, usually in traditional level we will install it in different VM's . But in container we will install it in different containers on top of a single OS and allows them to communicate each other.

- **The stateless nature of containers** - Containers are stainless which means you can create destroy bring up and down the containers at any time and it will not affect the application performance. This is one of the greatest features of the container.

Difference between containers and VM's

VM'S Containers



Each virtualized application includes not only the application binaries, but the OS also which may consume GB of space and memory

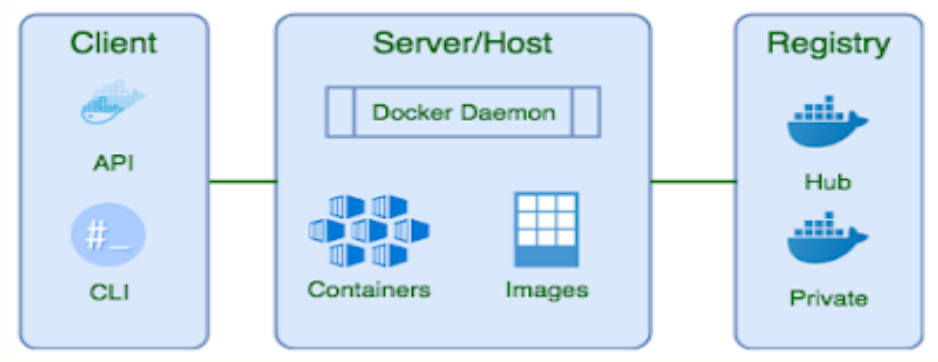
Where the container only contains application binaries and libraries which consumes MB of spaces and very less memory.

To create a VM it might take minutes where the container creation takes only seconds

The Average booting time of a VM is 600 seconds (depends up on the OS) where the container will be up with in seconds

A VM snapshot might take minutes where the container snapshot will take only seconds

Docker container architecture



Docker s one of the most popular application container technology now a days. This is an opensource technology and docker made partnership with industry reputed companies like Redhat,google. It allows creating sharing and running applications inside docker containers in an efficient manner

Docker uses client server type architecture

Docker server: - This is a service running in an operating system. This service is responsible for downloading,building and running containers

Docker Client: - This is a CLI tool which is responsible for communicating with docker server using REST API

Docker main components

Docker containers : - Isolated user space environments running the same or different applications and sharing the same host OS. Containers are created from docker mages

Docker Images:- Docker templates or docker images are the bundle of application libraries and respective applications. These mages used to create containers and we can bring up the containers immediately .

Docker registries :- This is an image store and these registries can be public or private which means we can download the images from internet or we can create our own store.

Linux Containers

Below are the main features of the Linux kernel which is used to help docker functioning as a container

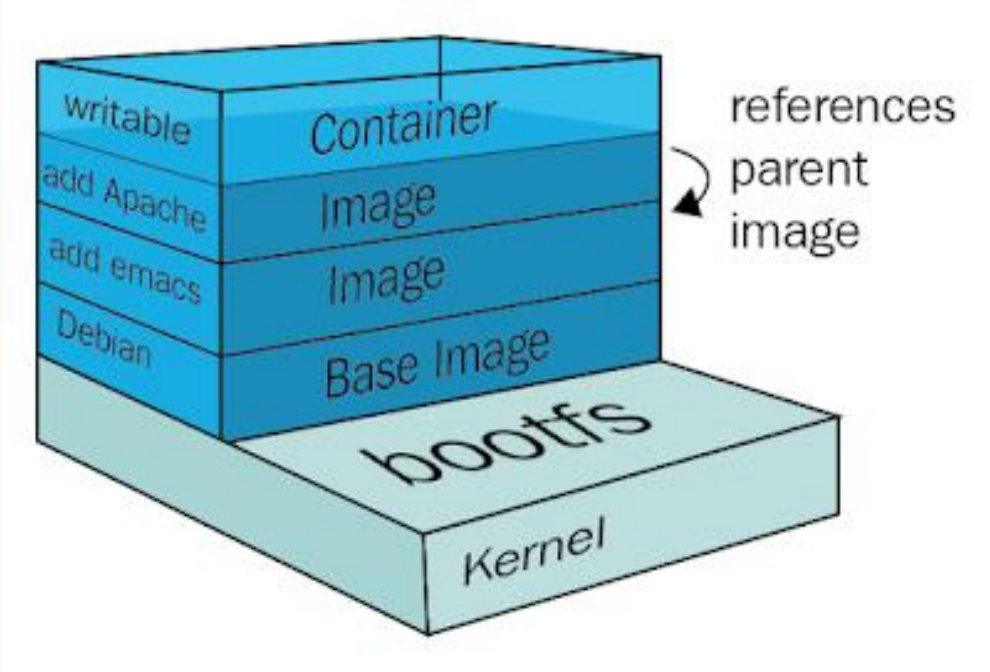
Linux name spaces : - This is a feature of the linux kernel to isolate applications each other.This allows one set of linux processes to see one group of resources while allowing another set of processes. Examples of linux name spaces are Mount (mnt), Process ID (PID), Network, User ID, Inter Process Communication (IPC). One of the best example of name spaces is two processes in two different mounted namespaces may have different views of what the mounted root file system is. Each container has specific name spaces and same will be used inside respective containers only.

Control groups (C-Groups) : - This feature can be use to control the system resources effectively .
C-groups allows containers to control resource utilisation on container wise.

SE-Linux : - Security Enhanced linux is a mandatory access control used for isolated system access. Docker is using the SE-Linux to protect the hosts and isolate the containers each other.

Docker Image file systems

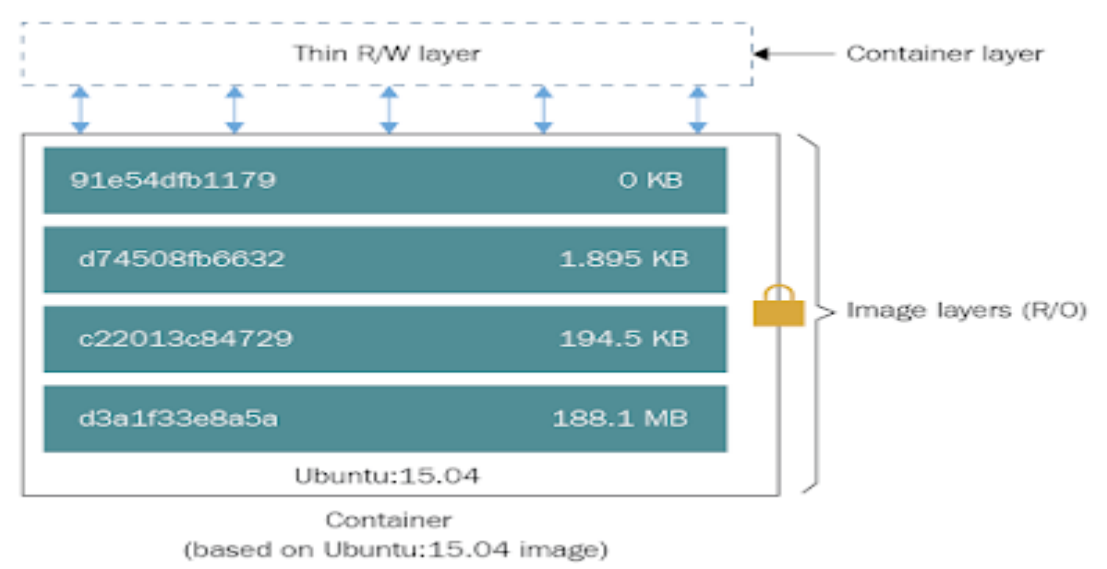
Docker image s a read-only template to build containers. An image consist of number of layers which can be accessed as a single virtual file system by the docker. Simply saying docker images are read only but we can add extra layers and create new images .



The container filesystem, used for every Docker image, is represented as a list of read-only
layers stacked on top of each other. These layers eventually form a base root filesystem for a
container. In order to make it happen, different storage drivers are being used. All the changes to the filesystem of a running container are done to the top level image layer of a
container. This layer is called a Container layer. What it basically means is that several containers may share access to the same underlying level of a Docker image, but write the
changes locally and uniquely to each other

Container Image layers

Docker image contains number of layers that are combined to a single file system using storage driver.The layers (images) are created when commands are executed during image build process. Each layer except the last one is read only so what ever the changes are doing in the images will get reflected to the top layer only.



Docker Registries

Docker registries are the method to share the image public or private mode which is also called docker store. This is highly scalable server side applications which you can use to store download docker images. As it is open source project users can download the images and create new docer mages with necessary changes . Docker registries are 2 type

Public registry

You can start a container from an image stored in a public registry. By default, the Docker
daemon looks for and downloads Docker images from Docker Hub, which is a public registry provided by Docker. However, many vendors add their own public registries to the Docker configuration at installation time

Private Registry

Organisations which do not want some specific images which they don't want to share public will create their own registries called private registry . So it have access only from with in your internal network.

You can easily install a private Docker registry by running a Docker container from a public
registry image. The private Docker registry installation process is no different from running
a regular Docker container with additional options.

Docker registries can be accessed from docker client through http method using REST API.