Docker

What is Docker:

Docker is a software development tool and a virtualization technology that makes it easy to develop, deploy, and manage applications by using containers.

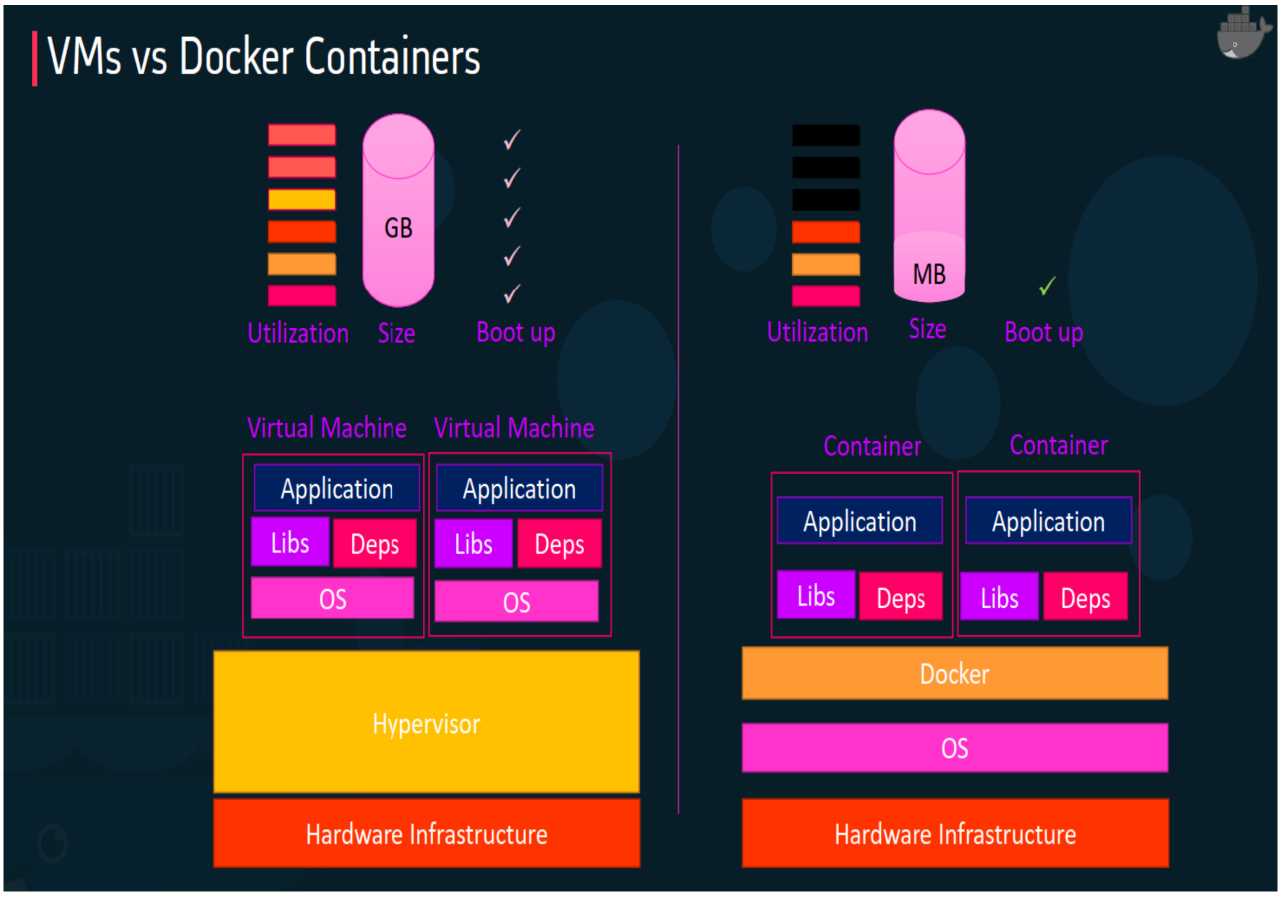
Container:

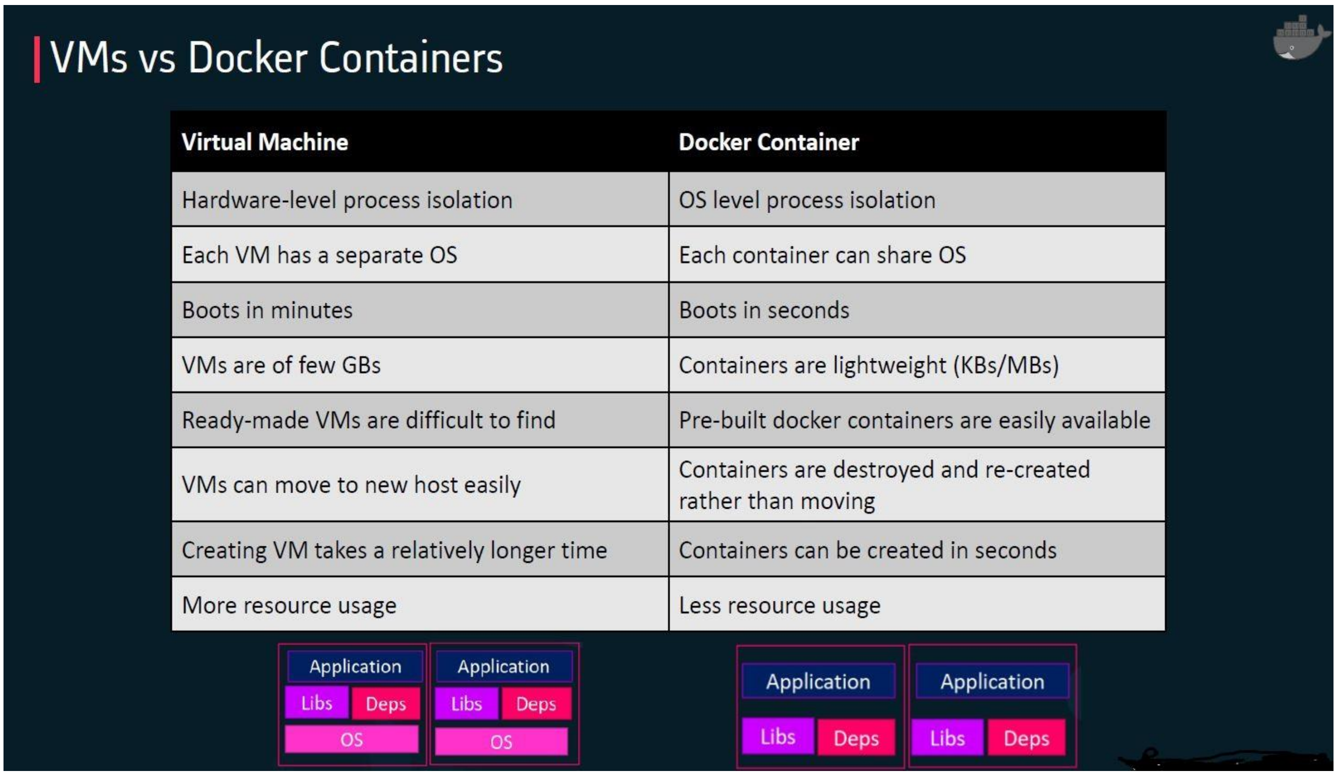
Container refers to a lightweight, stand-alone, executable package of a piece of software that contains all the libraries, configuration files, dependencies, and other necessary parts to operate the application.

What is Virtualization:

Virtualization is technology that lets you create useful IT services using resources that are traditionally bound to hardware. It allows you to use a physical machine's full capacity by distributing its capabilities among many users or environments.

Let's Make Difference on Container & Virtualization:

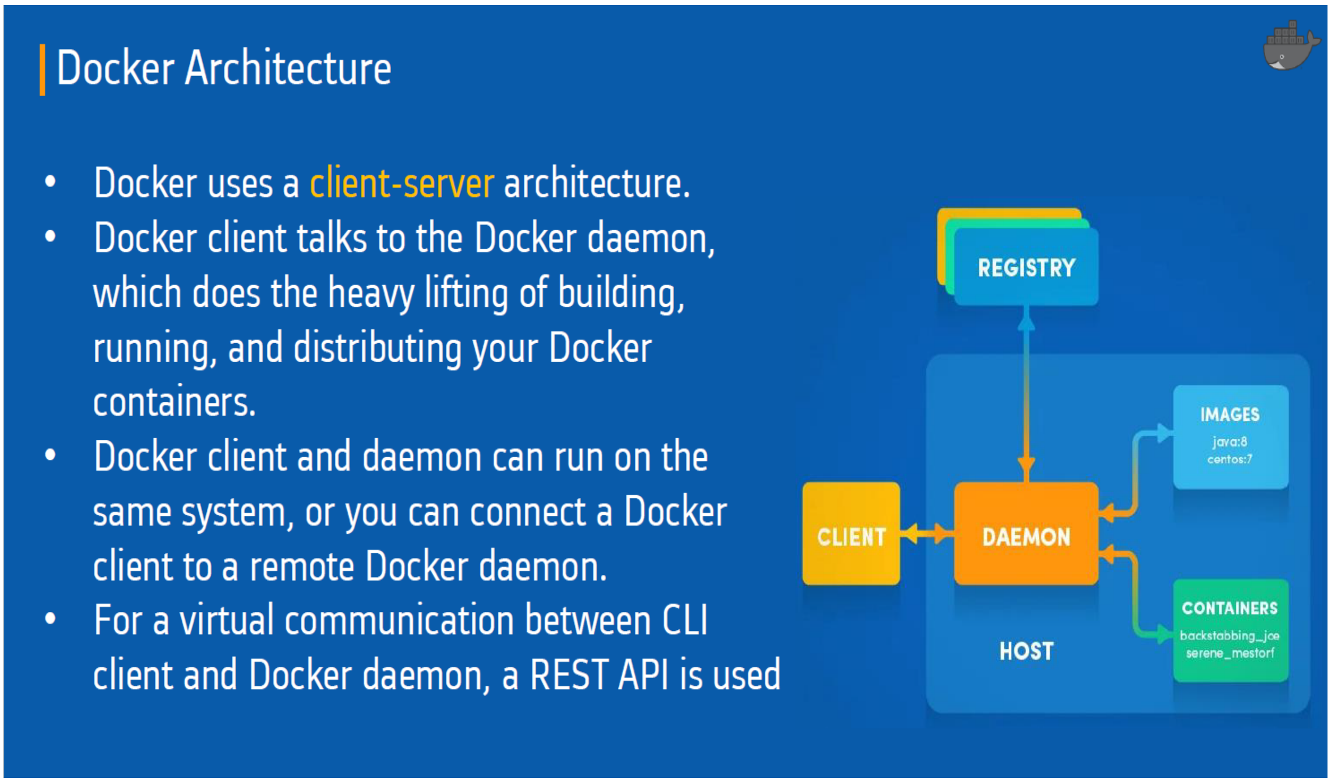




VMs vs Docker Containers

Virtual Machine	Docker Container
Hardware-level process isolation	OS level process isolation
Each VM has a separate OS	Each container can share OS
Boots in minutes	Boots in seconds
VMs are of few GBs	Containers are lightweight (KBs/MBs)
Ready-made VMs are difficult to find	Pre-built docker containers are easily available
VMs can move to new host easily	Containers are destroyed and re-created rather than moving
Creating VM takes a relatively longer time	Containers can be created in seconds
More resource usage	Less resource usage

Docker Architecture:



Docker Architecture

- Docker uses a client-server architecture.
- Docker client talks to the Docker daemon, which does the heavy lifting of building, running, and distributing your Docker containers.
- Docker client and daemon can run on the same system, or you can connect a Docker client to a remote Docker daemon.
- For a virtual communication between CLI client and Docker daemon, a REST API is used

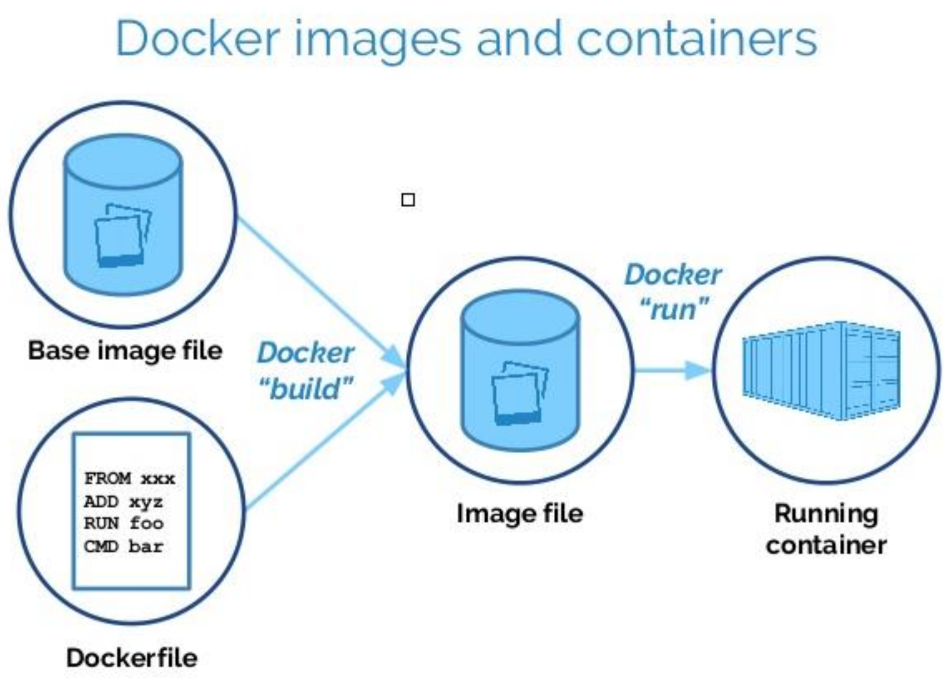
Docker Installation:

```
 # yum update -y
 # yum install yum-utils device-mapper-persistent-data lvm2 wget telnet vim -y
 # wget https://download.docker.com/linux/centos/docker-ce.repo
 # yum install docker-ce docker-ce-cli containerd.io
 # systemctl start docker
 # systemctl status docker
 # systemctl enable docker
 # systemctl is-enabled docker

# docker info
```

Docker Image:

A Docker image is a read-only template that contains a set of instructions for creating a container that can run on the Docker platform. It provides a convenient way to package up applications and preconfigured server environments, which you can use for your own private use or share publicly with other Docker users.



```
# docker image list
# docker pull centos:7
# docker image list
```

Docker Command:

```
Commands:

  attach      Attach local standard input, output, and error streams to a running container
  build       Build an image from a Dockerfile
  commit      Create a new image from a container's changes
  cp          Copy files/folders between a container and the local filesystem
  create      Create a new container
  diff        Inspect changes to files or directories on a container's filesystem
  events      Get real time events from the server
  exec        Run a command in a running container
  export      Export a container's filesystem as a tar archive
  history     Show the history of an image
  images      List images
  import      Import the contents from a tarball to create a filesystem image
  info        Display system-wide information
  inspect     Return low-level information on Docker objects
  kill        Kill one or more running containers
  load        Load an image from a tar archive or STDIN
  login       Log in to a Docker registry
  logout      Log out from a Docker registry
  logs        Fetch the logs of a container
  pause       Pause all processes within one or more containers
  port        List port mappings or a specific mapping for the container
  ps          List containers
  pull        Pull an image or a repository from a registry
  push        Push an image or a repository to a registry
  rename      Rename a container
  restart     Restart one or more containers
  rm          Remove one or more containers
  rmi         Remove one or more images
  run         Run a command in a new container
  save        Save one or more images to a tar archive (streamed to STDOUT by default)
  search      Search the Docker Hub for images
  start       Start one or more stopped containers
  stats       Display a live stream of container(s) resource usage statistics
  stop        Stop one or more running containers
  tag         Create a tag TARGET_IMAGE that refers to SOURCE_IMAGE
  top         Display the running processes of a container
  unpause     Unpause all processes within one or more containers
  update      Update configuration of one or more containers
  version     Show the Docker version information
  wait        Block until one or more containers stop, then print their exit codes
```

Deploy a Nginx Container:

```
# docker container run -it --name page-web --privileged=true -p 8080:80 -d centos:7
/usr/sbin/init
```

Login to a Container:

```
# docker container exec -it <container-id> bash
```

Install nginx on the Container:

```
# yum update -y
# yum install vim net-tools wget telnet*
# ifconfig
# vim /etc/yum.repos.d/nginx.repo
        [nginx]
        name=nginx repo
        baseurl=http://nginx.org/packages/mainline/centos/7/$basearch/
        gpgcheck=0
        enabled=1
:x

# yum install nginx -y
# systemctl start nginx
# systemctl enable nginx
# systemctl status nginx
```

Image creation from a running container:

```
 # docker commit -m "Commit a Nginx Container" 49fd278803d2 pagenginx
 # docker images
```

Docker Host Configuration:

Make the host enable port forwarding

```
# vim /etc/sysctl.conf
  net.ipv4.conf.all.forwarding=1
 :x
```

Disabled Selinux on Docker Host:

```
# vim/etc/selinux/config

 disabled
 :x

# init 6
```