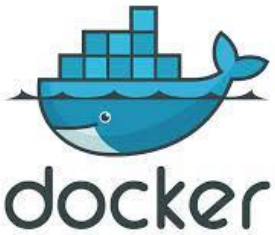

docker

## ❖ DOCKER:

- Docker is an open platform tool for developing, shipping, & running applications. Docker enables you to separate your applications from your infrastructure so you can deliver software quickly.
- Docker is a bit like a virtual machine. But unlike a virtual machine, rather than creating a whole virtual operating system.
- Docker provides a way to run applications securely isolated in a container, packaged with all its dependencies and libraries.
- It is designed to benefit both developers and system administrators, making it a part of many DevOps tool chains.

## ➢ VIRTUALIZATION (VT):

- **VT** is a software technology that makes computing environments independent of physical infrastructure.
- VT is a process of creating virtual apps, virtual servers, storage and network
- It is the single most effective way to reduce IT expenses while boosting efficiency & agility for all size businesses.

### VIRTUALIZATION BENEFITS:

- Reduced capital and operating costs.
- Minimized or eliminated downtime.
- Increased IT productivity, efficiency, agility and responsiveness.
- Faster provisioning of applications and resources.
- Greater business continuity and disaster recovery.

### VIRTUALIZATION TYPES:

- Server Virtualization
- Network Virtualization
- Desktop Virtualization
- Para-virtualization
- Hardware-level virtualization

## ➢ HYPERVISORS:

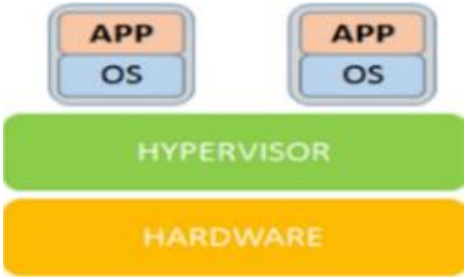
- A hypervisor is a hardware virtualization technique that **allows multiple guest operating systems** to run on a single host system at the same time.
- Guest OS shares hardware of the host computer, have its own processor, memory and other h/w resources.
- A hypervisor is also known as a **Virtual Machine Manager (VMM).**

## HYPERVISIOR TYPES:

### TYPE1:

- **Type1** is on bare metal. VM resources are scheduled directly to the hardware by the hypervisor.

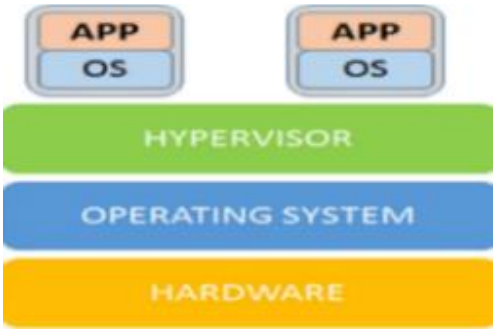
  **Eg:** VMware ESXI, Citrix XenServer, Microsoft Hyper-V, Linux KVM.




### TYPE2:

- **Type2** is hosted. VM resources are scheduled against a host operating system, which is then executed against the hardware.

  **Eg**: VMware workstation and Oracle virtual box.



## ➢ VIRTUAL MACHINE (VM):

- **A VM** is a virtual environment that functions as a virtual computer system with its own CPU, memory, network, and storage, created on a physical.
- Most enterprises use a combination of physical and virtual infrastructure to balance the corresponding advantages and disadvantages.

## KEY PROPERTIES OF VIRTUAL MACHINE:

### PARTITIONING:

- Run multiple operating systems on one physical machine.
- Divide system resources between virtual machines.

### ISOLATION:

- Provide fault and security isolation at the hardware level.
- Preserve performance with advanced resource controls.

### ENCAPSULATION:

- Save the entire state of a virtual machine to files.
- Move and copy virtual machines as easily as moving and copying files.

### HARDWARE INDEPENDENCE:

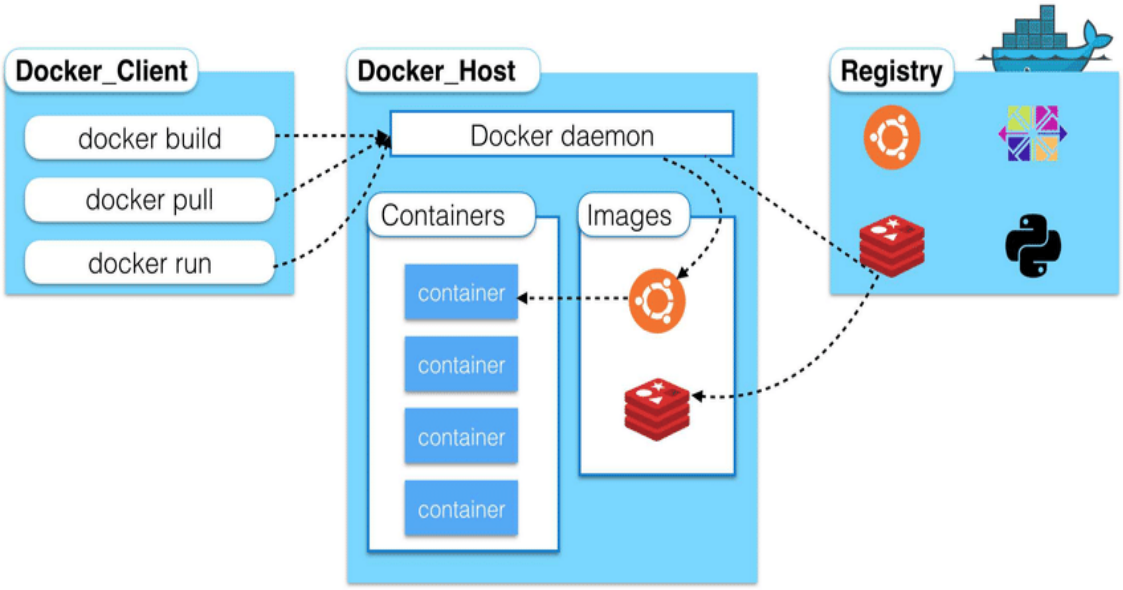
- Provision or migrate any virtual machine to any physical server.

## ➢ VIRTUALIZATION vs. CLOUD COMPUTING:

- Virtualization is software that makes computing environments independent of physical infrastructure.
- Cloud computing is a service that delivers shared computing resources (software and/or data) on demand via the Internet.
- As complementary solutions, organizations can begin by virtualizing their servers and then moving to cloud computing for even greater agility and self-service.

## ➢ DOCKER ARCHITECTURE:

- Docker uses a **client-server** architecture. The Docker **client** talks to the **Docker daemon**, which does the heavy lifting of building, running, and distributing your Docker containers.
- The Docker client and daemon can run on the same system, or you can connect a Docker client to a remote **Docker daemon**.




### DOCKER DAEMON:

- The Docker daemon (dockerd) listens for Docker API requests and manages Docker objects such as images, containers, networks, and volumes.

### DOCKER CLIENT:

- The Docker client (docker) is the primary way that many Docker users interact with Docker.

### DOCKER REGISTRIES:

- A Docker *registry* stores Docker images.
- Docker Hub is a public registry that anyone can use, and Docker is configured to look for images on Docker Hub by default.

## ➢ DOCKER OBJECTS:

### IMAGES:

- An *image* is a read-only template with instructions for creating a Docker container.
- A container is launched by running an image. An **image** is an executable package that includes everything needed to run an application–the code, a runtime, libraries, environment variables, and configuration files.

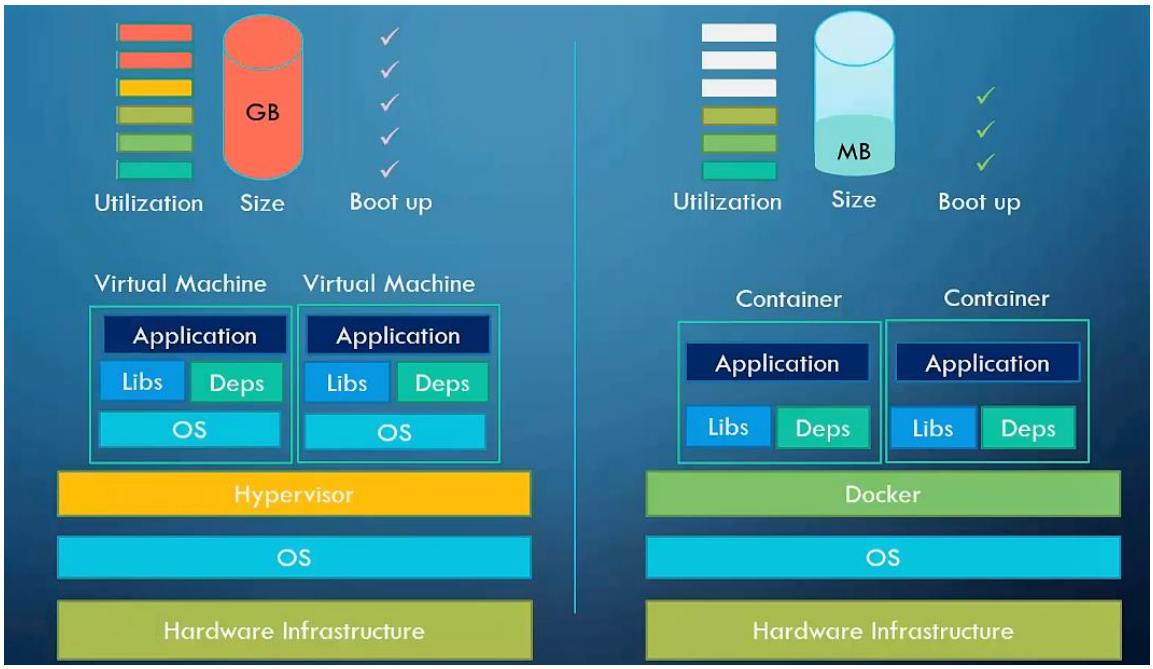
### CONTAINERS:

- A container is a runnable instance of an image. You can create, start, stop, move, or delete a container using the Docker API or CLI.
- Docker Containers are:
  - **Flexible** : The most complex applications can be containerized.
  - **Lightweight** : Containers leverage and share the host kernel.
  - **Interchangeable:** You can deploy updates and upgrades on-the-fly.
  - **Portable** : Build locally, deploy to the cloud, and run anywhere.
  - **Scalable** : Increase and automatically distribute container replicas.
  - **Stackable** : You can stack services vertically and on-the-fly.

## ➢ THE UNDERLYING TECHNOLOGY:

- Docker is written in the Go programming language and takes advantage of several features of the Linux kernel to deliver its functionality.
- Docker uses a technology called **namespaces** to provide the **isolated workspace** called the **container.**
- When you run a container, Docker creates a set of namespaces for that container.
- These namespaces provide a layer of isolation. Each aspect of a container runs in a separate namespace and its access is limited to that namespace.

## ➢ VIRTUAL MACHINES VS CONTAINERS:




### VIRTUAL MACHINES:

- A virtual machine (VM) is a virtual environment that functions as a virtual computer system with its own CPU, memory, network interface, and storage, created on a physical. In other words, creating a computer within a computer.
- Multiple virtual machines can run simultaneously on the same physical computer.

### CONTAINERS:

- A container is a running instance of an image.
- You can create, start, stop, move, or delete a container using the Docker API or CLI.