

# DevOps



**Caltech**

Center for Technology &  
Management Education

## Post Graduate Program in DevOps

# DevOps



**Caltech**

Center for Technology &  
Management Education

## CI/CD Pipeline with Jenkins

# Jenkins on Docker



# Learning Objectives

By the end of this lesson, you will be able to:

- List the similarities and differences between Docker and a virtual machine
- Install Docker on Windows, Unix, and macOS machines
- Install and access Jenkins on Docker
- Discuss Docker volume and its advantages



# Docker

# Introduction to Docker

Docker is a light weight, open-source software that is used to create, deploy, and manage virtualized application containers. These containers are managed on a common operating system.



Provides abstraction and automation of OS-level virtualization

Allows developers and admins to deploy applications within containers and run them on any Linux or Windows host

Supports packaging both software and dependencies within the Docker container in a standard process

# Introduction to Docker

Docker is operating system-independent. It works on Windows, Linux, and macOS-based platforms seamlessly.



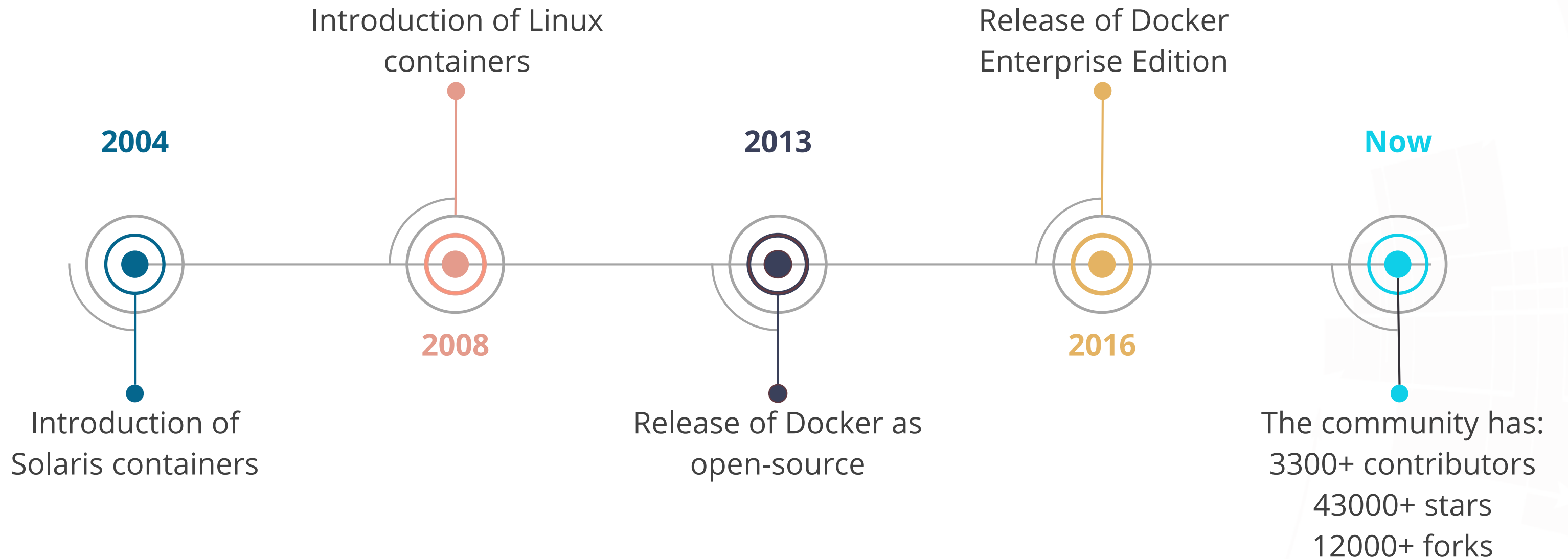
Is fast, reliable, and highly scalable

Helps resolve versioning problem

## *Note*

Docker containers run totally isolated from each other.

# History of Docker





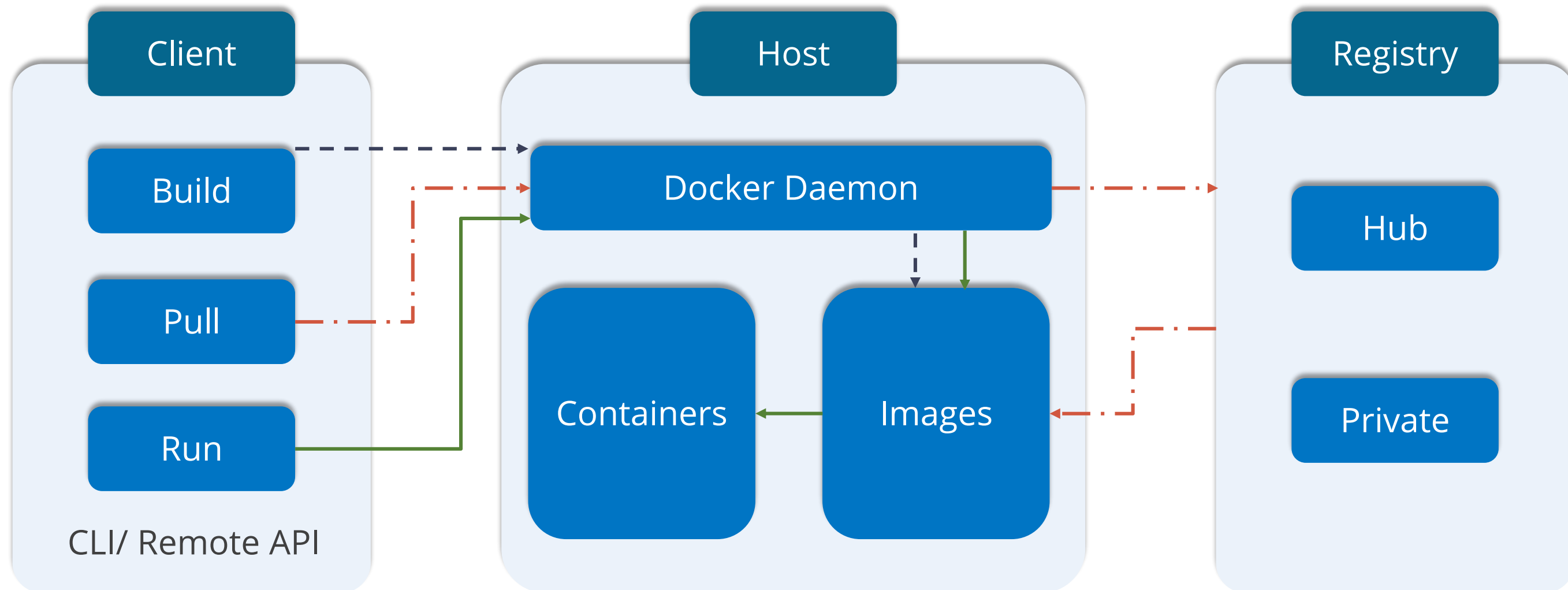
# Benefits of Docker

Some of the benefits of Docker are:

- Helps reduce infrastructure cost
- Ensures applications run in isolated environment
- Establishes standardization of applications across multiple environments
- Removes versioning and dependency issues related to source code
- Is built once and deployed to multiple environments and operating systems
- Helps in faster configurations and building applications

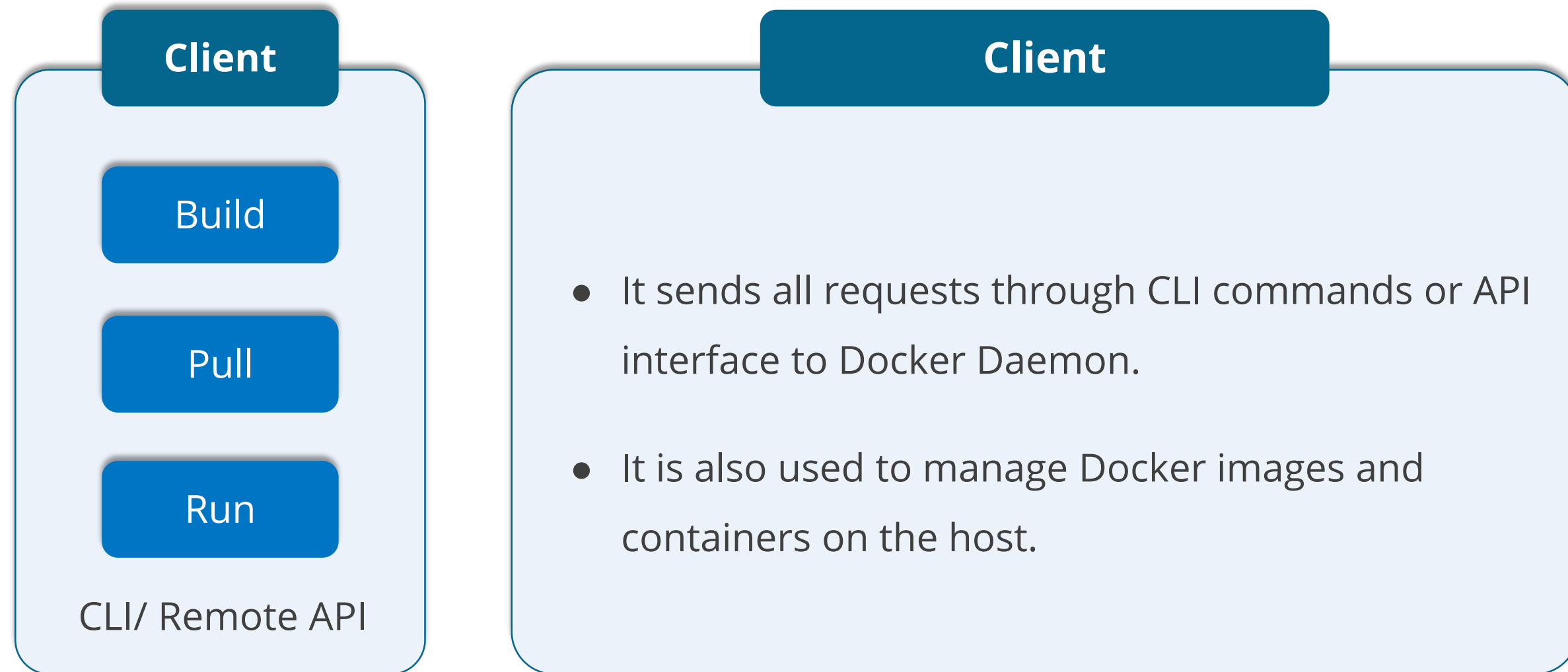
# Components of Docker

Docker uses a client-server architecture.  
The diagram below represents the architecture of Docker.



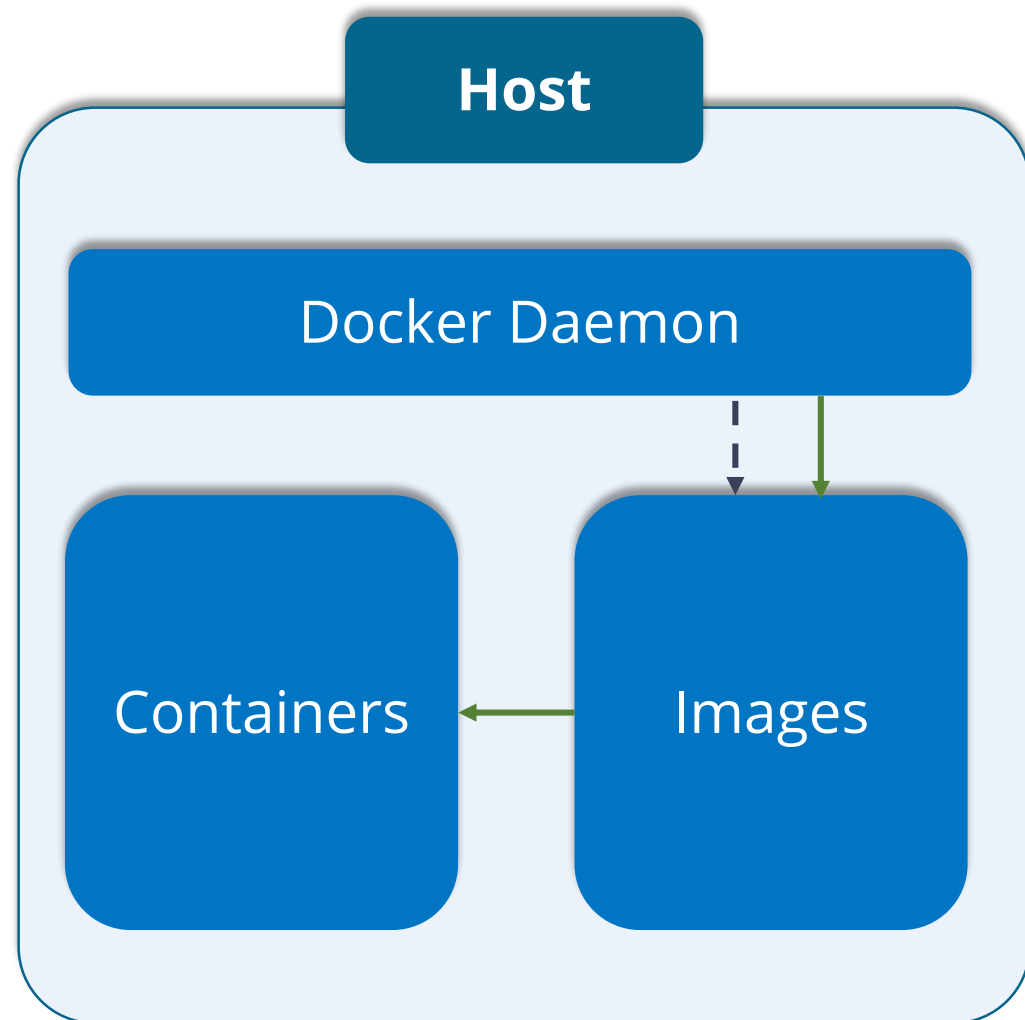
# Components of Docker

The Docker client is used to interact with a Docker Daemon.



# Components of Docker

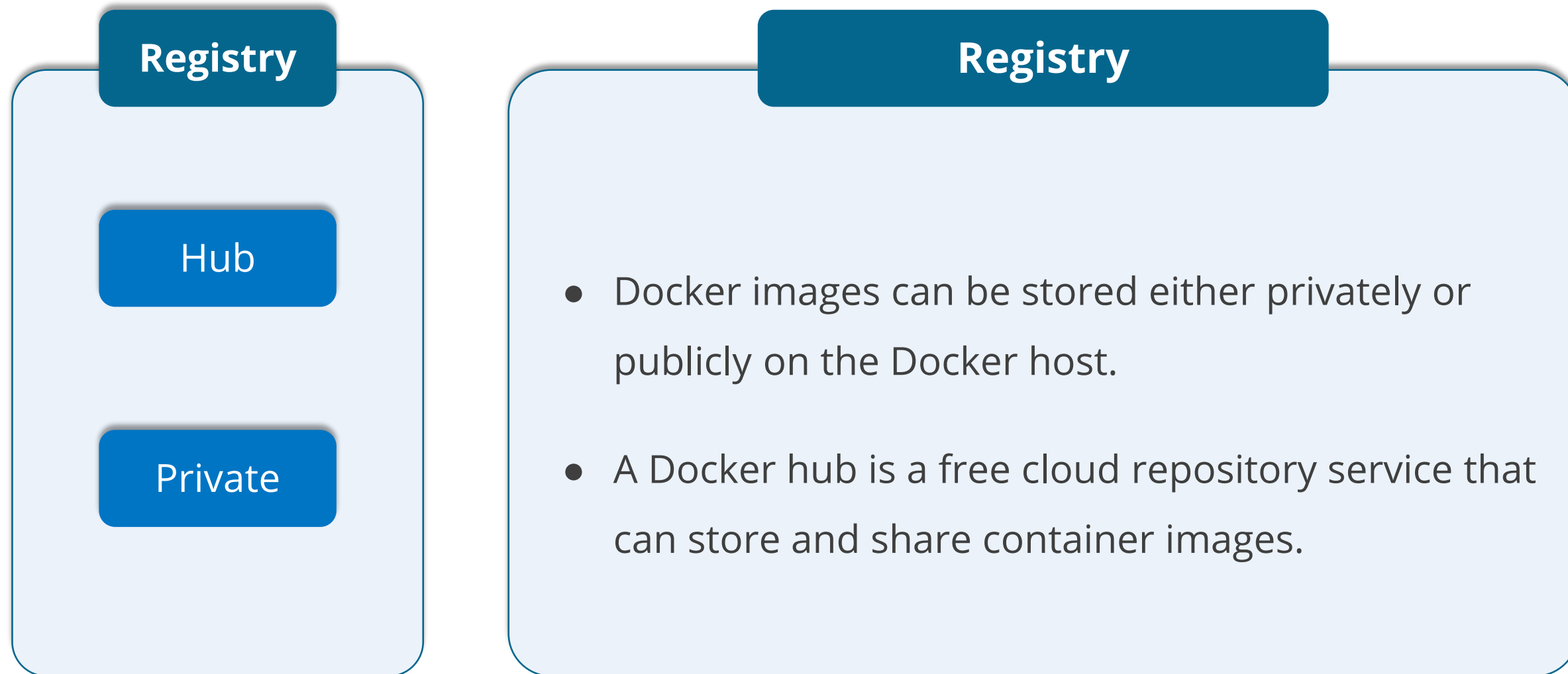
The Docker Host is a physical computer or a virtual machine on which the Docker daemon and the Docker host can run.



- The Docker Daemon (dockerd) accepts API from the client and performs various operations on a Docker host.
- A Docker image is a read-only template that has instructions for creating a Docker container.
- A Docker container is an instance of a Docker image.

# Components of Docker

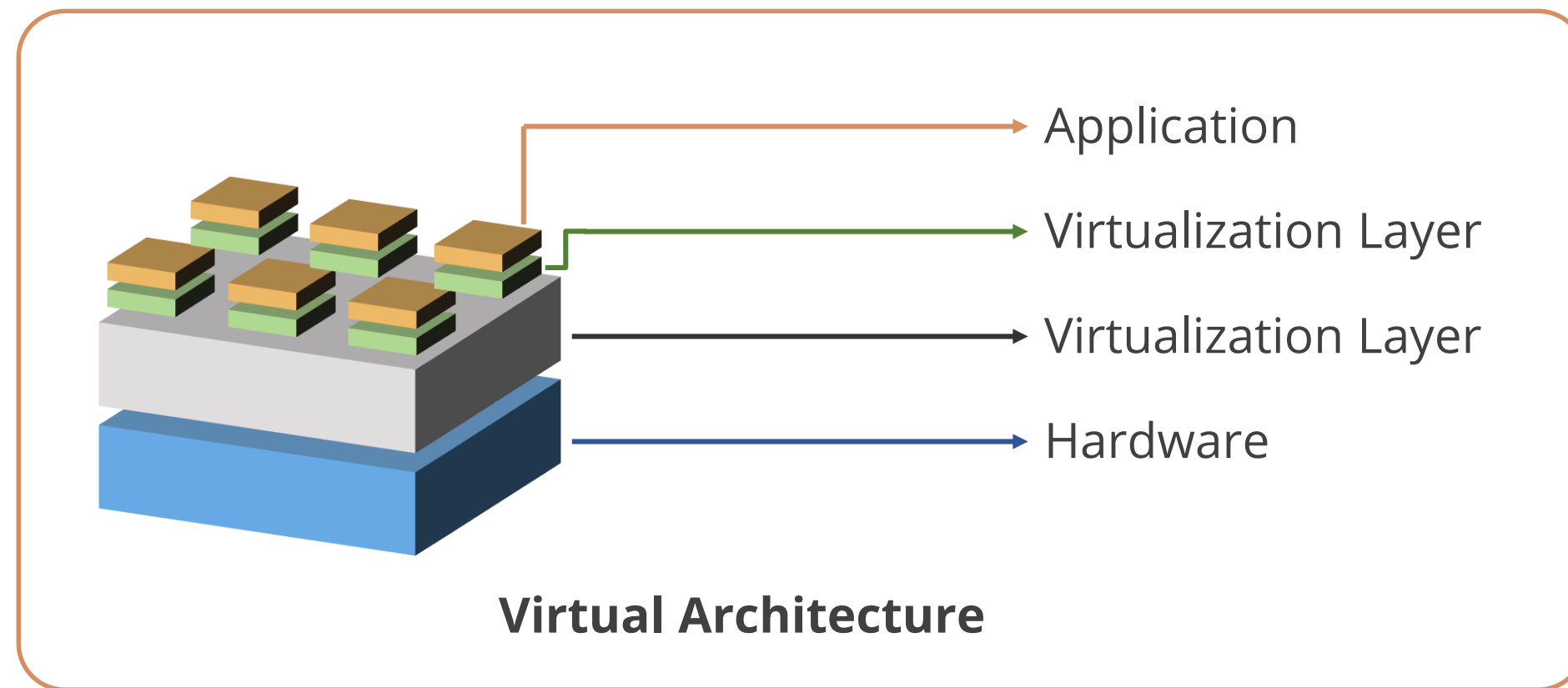
A Docker registry hosts Docker images.





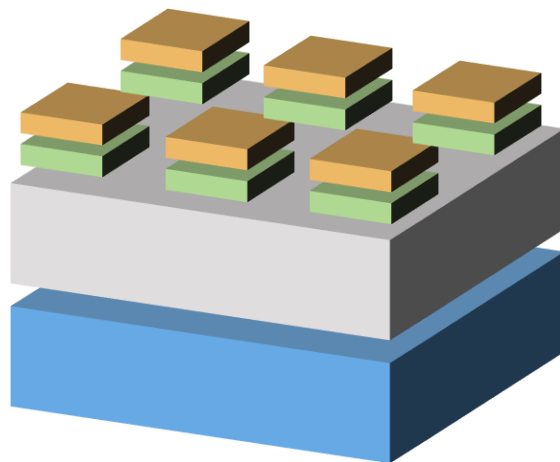
# Virtualization

Virtualization refers to creating virtual resources like server, CPU, storage, or network. This is done by creating an abstraction layer over the hardware devices. It helps in efficient utilization of the hardware.



# Virtualization

Virtualization leads to optimal utilization of physical resources, which, in turn, reduces expenses.



- Creates multiple environments of a single physical system

- Helps host multiple operating systems isolated from each other

- Drives cloud computing

- Enables cloud users to scale resources with increased workload

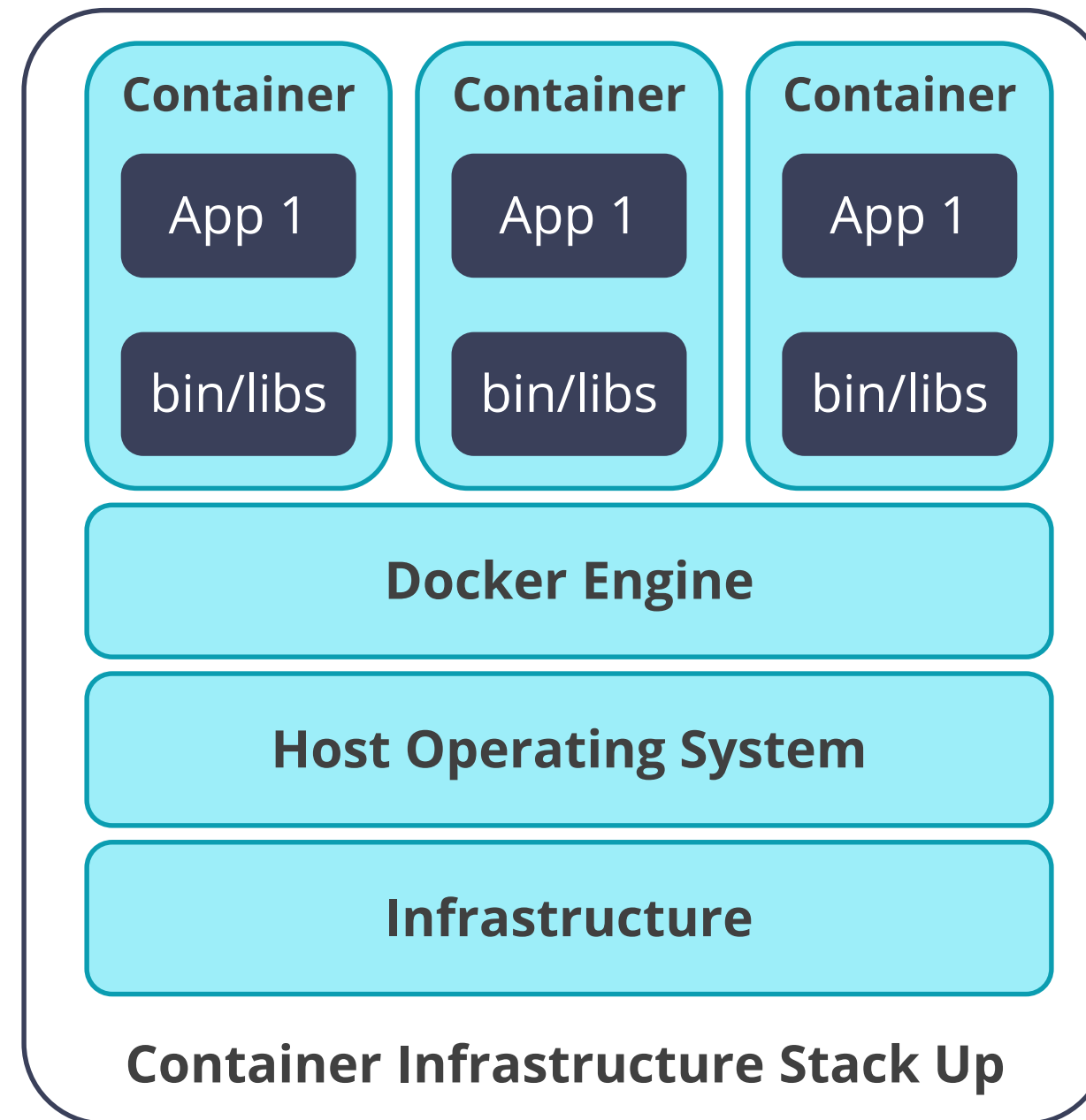
# Docker vs. Virtualization

The following table presents the similarities and differences between Docker and a virtual machine:

	Docker	Virtual Machine
Operating System	Supports basic OS-based images	Supports any operating system
Startup Time	Is 1-2s	Is 1-2 min
Disk Usage	Is low	Is high
Memory Usage	Is low	Is high
Configuration Setup	Is simple and easy	Is complex
Hardware Requirement	Is less	Is more
Security	Is not matured	Is matured and offers better security

# Docker and Kernel Integration

Docker uses Linux kernel and inherits all features of kernel including Cgroups and namespaces. These features enable Docker to segregate processes and run them independently.



# Docker Installation

Docker can be installed across different platforms and environments. Docker provides Desktop and Server editions for the following platforms:

## Desktop Edition

- Docker Desktop for Mac
- Docker Desktop for Windows

## Server Edition

- Docker for CentOS
- Docker for Ubuntu
- Docker for RedHat
- Docker for Fedora
- Docker for Debian



# Docker Installation on Unix Flavors

Docker can be installed on **Ubuntu** and **Debian** using the **apt package installer**.

To install Docker, the following set of commands are to be keyed in:

Demo-1

```
apt update  
apt install Docker
```

Docker can be installed on **CentOS**, **Fedora**, and **RedHat** using the **yum package installer**.

To install Docker, the following set of commands are to be keyed in:

Demo-1

```
yum install epel-release  
yum install docker-io
```

# Docker Installation on macOS

Docker supports desktop installations on mac platforms. To install a stable version of Docker on a mac host, follow the steps given below:

1

Download the **Docker dmg** file from the following URL:  
<https://download.docker.com/mac/stable/Docker.dmg>

2

Follow the steps in the installation page to complete the installation.

## Note

Docker installation requires the mac machine to have a 4GB RAM (minimum) and a xhyve hypervisor.

# Docker Installation on Windows

Docker supports desktop installations on Windows platforms. To install a stable version of Docker on a Windows host, follow the steps given below:

1

Download the **Docker exe** file from the following URL:  
<https://download.docker.com/win/stable/Docker%20Desktop%20Installer.exe>

2

Upon downloading the installer, follow the steps to complete the installation.

## Note

Docker installation requires the Windows machine to have a 4GB RAM (minimum) and a Hyper-V.

# Assisted Practice

Installing Docker on Windows and Linux.

Duration: 10 min

## Problem Statement:

Install Docker on Windows and Linux machines.

# Assisted Practice: Guidelines

---

## Steps to install Docker on Windows and Linux machines:

1. Download the Docker for Windows .exe file.
2. Configure Docker according to users' requirements and complete the installation.
3. Execute apt update and apt install docker.io commands to install Docker on ubuntu Operating system.
4. Run docker version and service docker status commands to validate the installation.



# Jenkins Installation on Docker

# Deploying Jenkins on Docker

Jenkins supports deployment of the Jenkins CI tool on Docker and Kubernetes. To deploy Jenkins on Docker, Docker recommends the official Docker image (jenkins/jenkins) available on the Docker Hub.



Jenkins Docker Image

- Is equivalent to the latest long-term support (LTS) release of Jenkins

- Is not bundled with Blue Ocean plugin and features

- Is published every time a new LTS version of Jenkins is rolled out

# Initialize Jenkins in a Docker Container

To install Jenkins inside a Docker container on the Docker host, follow the steps given below:

Step 1

Login to the Linux VM using the terminal.

Step 2

Create a new Docker network with the bridge driver using the Docker network create command.

Step 3

Deploy the Jenkins container using the newly created network with the Docker run command.

Demo-1

```
# Step 2:  
Docker network create jenkins  
  
# Step 3:  
Docker run -d --name jenkins --  
network=jenkins jenkins/jenkins:lts
```

# Start and Stop Management of Jenkins Container

To start and stop a Jenkins container, the container ID or the Docker container name must be obtained first.

To start and stop a Jenkins container follow the steps mentioned below:

**Step 1:** Get the container name using **Docker ps** command.

Demo-1

```
#Get the Container ID  
Docker ps
```

# Start and Stop Management of Jenkins Container

**Step 2:** Run **Docker start** command to start a Docker container and Jenkins application.

Demo-1

```
#Start Jenkins application along with Docker container  
Docker start <container_id>
```

**Step 3:** Run **Docker stop** command to stop the Docker container and Jenkins application from running.

Demo-1

```
#Stop Jenkins application along with Docker container  
Docker stop <container_id>
```



# Access Jenkins Home inside Docker Container

The **exec** command can be used to access files deployed inside the Docker container.

This command can be used to connect to the container and fetch the required files.

The Jenkins image is a custom-prepared Docker image in which **/var/Jenkins\_home** is configured as Jenkins Home Path.

# Access Jenkins Home inside Docker Container

Once the connection with the Docker container is established, users will be directly logged into the home directory inside the container.

Run the following commands to go inside the Docker container:

Demo-1

```
Docker ps  
Docker exec -it <CONTAINER ID or CONTAINER NAME> bash
```

# Access Jenkins Application Using VM Public DNS

Docker containers can connect to the outside world.

The Docker containers must be exposed to the outside world in order to access applications that are deployed inside the containers.

To **expose all ports** built in a Docker image, use the flag **-p** with the **Docker run** command.

# Access Jenkins Application Using VM Public DNS

To **expose only a specific port**, use the following command:

Demo-1

```
Docker run -d -name Jenkins -network = Jenkins -p 8080:8080 jenkins/jenkins:lts
```

Once the container is deployed, users can **access** Jenkins using the following syntax:

Demo-1

```
<PUBLIC IP>:8080
```

# Assisted Practice

## Deploying Jenkins Container on Docker

Duration: 10 min

### Problem Statement:

Demonstrate the deployment of Jenkins container on Docker.

# Assisted Practice: Guidelines

---

## Steps to deploy Jenkins container on Docker:

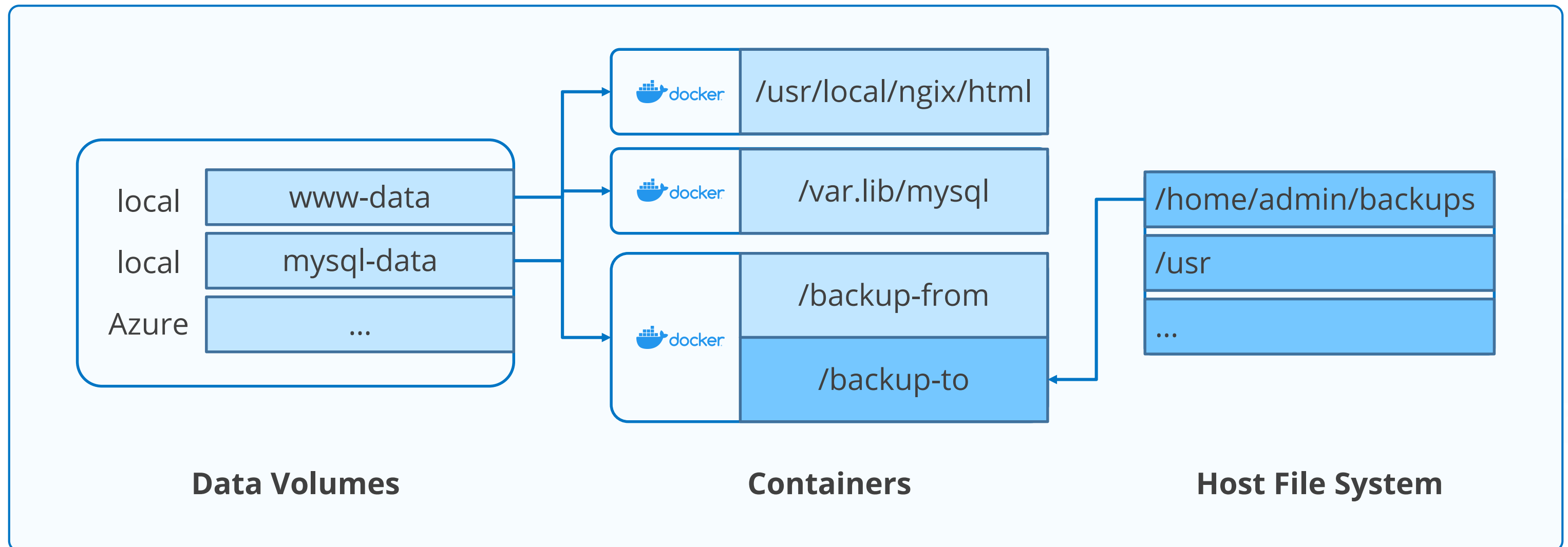
1. Login to the VM as root user. Check if Docker is up and running.
2. Pull Docker image.
3. Deploy the container using `docker container run -d -p 8080:8080 \, --name jenkins-docker, and jenkins/jenkins:its` commands.
4. Access Jenkins using Docker host IP address and 8080 ports.
5. Fetch the initial admin password using `docker container exec [CONTAINER ID or NAME] \ and sh -c "cat /var/jenkins_home/secrets/initialAdminPassword"` commands.

# Docker Volume with Jenkins



# Docker Volumes

Docker volume is a feature provided by Docker for persisting data generated and used by Docker containers. Docker volumes are nothing but file systems.



# Docker Volume

Docker volume is used to ensure data persistence while working on containers.



Volume

- Can be managed through API interface or Docker CLI commands

- Does not increase the size of the containers using it

- Allows data to exist independent of the lifecycle of a specific container

- Supports Docker containers to be attached for storing data inside the volume

# Make Jenkins Container Persistent Using Docker Volume

Data inside the Docker container can be made persistent using Docker volume. This results in data being available even when the container is lost.

Docker volume keeps data safe and secure outside the container.


To create and access Docker volume, run the following commands:

Demo-1

```
Docker volume ls  
Docker volume create <VOLUME NAME>  
Docker volume inspect <VOLUME NAME>
```

# Copy-on-Write Strategy

Copy-on-Write is a strategy for copying and sharing files. It is a resource management technique that delivers maximum efficiency. It is also called shadowing or implicit sharing.



A unit of data is copied; if not modified, then the copy acts as a reference to the original unit of data.

A unit of data is copied; when the copied data is modified, then a new copy is created.

## Note

If there is a file or directory within the image in a lower layer and another layer needs read access, it just uses the existing file.

# Mount a Volume on Jenkins Container

Volumes are used to ensure that there will be no data loss. To mount a volume on the Jenkins container while the container is created, use the **-v flag** with the **Docker run** command.

To create and access Docker volume, run the following commands:

## Demo-1

```
Docker container run -d \  
  -p [YOUR PORT]:8080 \  
  -v [YOUR VOLUME]:/var/jenkins_home \  
  --name jenkins \  
  jenkins/jenkins:lts
```

## Key Takeaways

- ❶ Docker is a light weight, open-source software that is used to create, deploy, and manage virtualized application containers.
- ❷ Virtualization refers to creating virtual resources like server, CPU, storage, or network.
- ❸ Docker can be installed across different platforms and environments.
- ❹ Jenkins supports deployment of the Jenkins CI tool on Docker and Kubernetes.
- ❺ Docker volume is a feature provided by Docker for persisting data generated and used by Docker containers.



## Lesson-End Project

## Making a Jenkins Container Persistent

### Problem Statement:

Perform the following:

- Integrate the Docker volume to make the Jenkins container persistent.

**Access:** Click on the **Labs** tab on the left side panel of the LMS. Copy or note the username and password that is generated. Click on the **Launch Lab** button. On the page that appears, enter the username and password in the respective fields, and click **Login**.







# Thank You