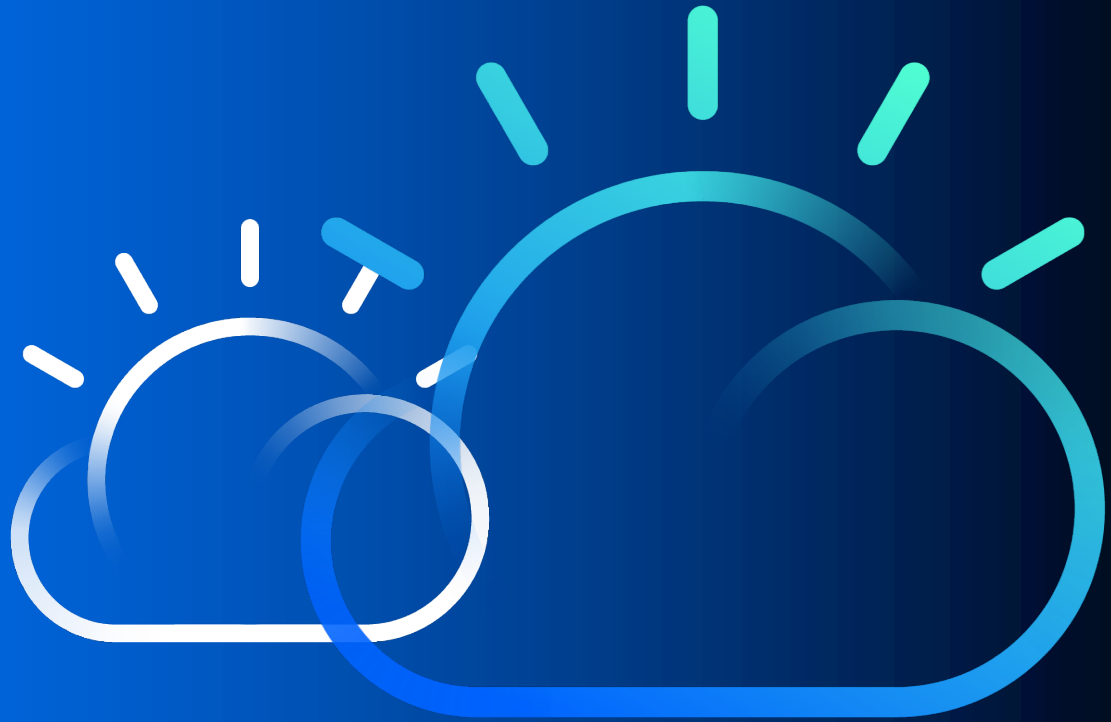


Containers and Docker

Concepts



What are containers?

A standard way to package an application and all its dependencies so that it can be moved between environments and run without change

Work by hiding the differences between applications inside the container so that everything outside the container can be standardized

Docker: provides a standard way to create images for Linux Containers



Linux Containers (LXC) details:

- An isolated user space within a running Linux OS
- Shared kernel across containers
- Direct device access
- All packages and data in an isolated runtime, saved as a filesystem
- Resource management implemented with control groups (cgroups)
- Resource isolation through namespaces

Why use containers?

Containers are a critical foundation for distributed apps in hybrid clouds

Ship more software

Accelerate development, CI and CD pipelines by eliminating headaches of setting up environments and dealing with differences between environments. On average, Docker users ship software more frequently.

Resource efficiency

Lightweight containers run on a single machine and share the same OS kernel while images are layered file systems sharing common files to make efficient use of RAM and disk and start instantly.

App portability

Isolated containers package the application, dependencies, and configurations together. These containers can then seamlessly move across environments and infrastructures.

The challenge

**Multiplicity
of Stacks**

Static website:

- Nginx
- OpenSSL
- Bootstrap 2
- ModSecurity

User DB:

- PostgreSQL
- pgv8
- v8

Web front end:

- Ruby
- Rails
- Sass
- Unicom

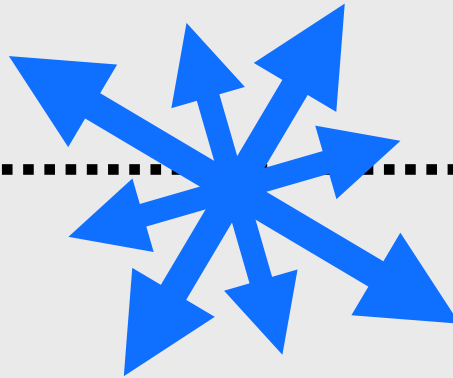
Queue:

- Redis
- Redis-sentinel

Analytics DB:

- Hadoop
- Hive
- Thrift
- OpenJDK

**Do services
and apps interact
appropriately?**



**Multiplicity of
hardware
environments**



Development
VM



QA server



Customer
Data Center



Public Cloud



Production
Cluster



Contributor's
laptop

**Can I migrate
smoothly and
quickly?**

Docker: A shipping container for code

**Multiplicity
of Stacks**

Static website

User DB

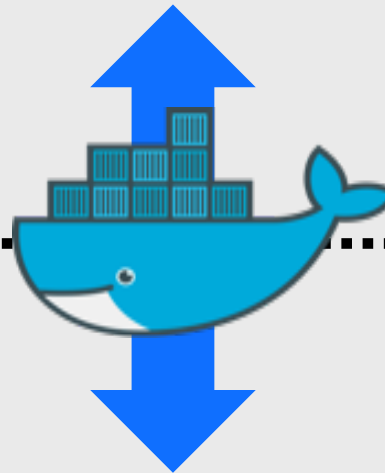
Web front end

Queue

Analytics DB

**Do services
and apps interact
appropriately?**

An engine that enables any
payload to be encapsulated
as a lightweight, portable,
self-sufficient container...



...that can be manipulated by
using standard operations, and
run consistently on virtually any
hardware platform.

**Multiplicity of
hardware
environments**



Development
VM



QA server



Customer
Data Center



Public Cloud



Production
Cluster



Contributor's
laptop

**Can I migrate
smoothly and
quickly?**

Benefits of using containers

Can run on many different platforms

Processes share OS resources, but remain segregated

Isolate the different requirements between the applications that run inside the container, and the operations that run outside the container

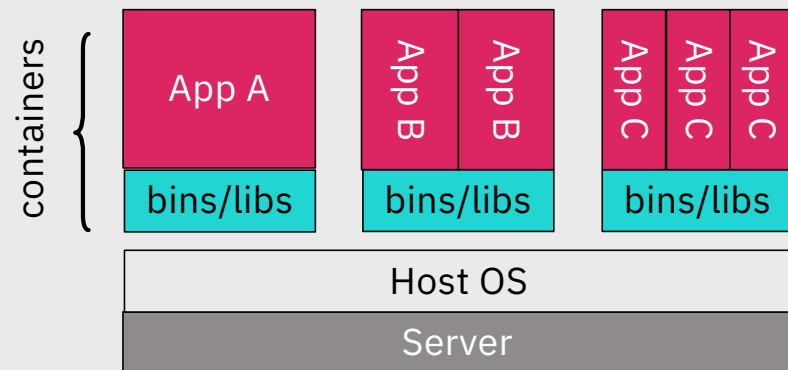
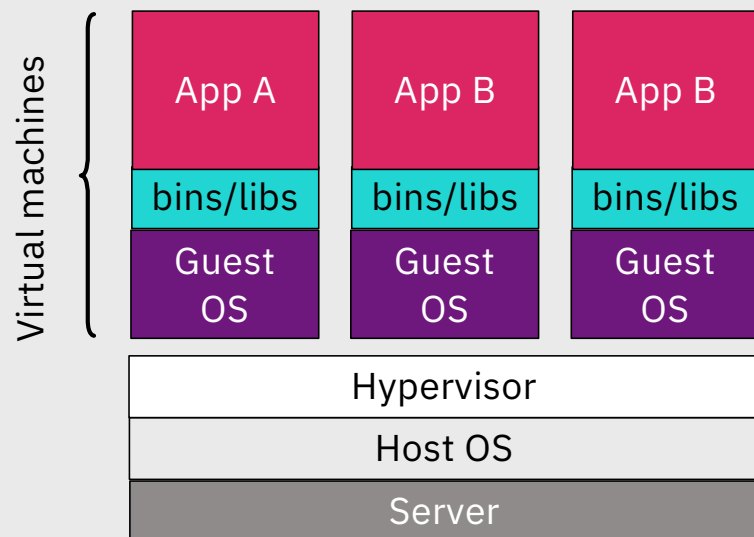
Quick and easy to create, delete, start, stop, download, and share

Use hardware resources more efficiently than virtual machines, and are more lightweight

Can be treated as unchangeable

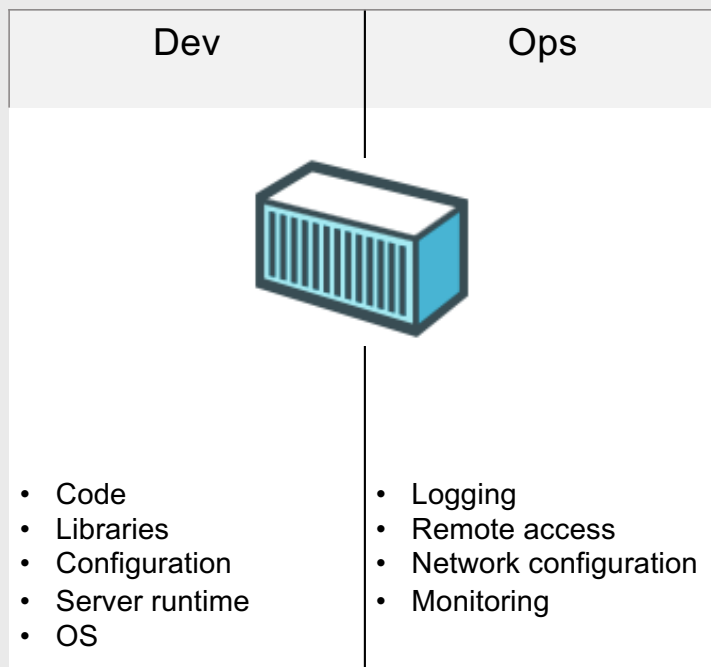


Virtual machines versus containers



Containers are isolated, but share OS and, where appropriate, bins/libraries

Dev versus Ops



Separation of concerns

- A container separates and bridges the **Dev** and **Ops** in DevOps
- **Dev** focuses on the application environment
- **Ops** focuses on the deployment environment

Container ecosystem

Docker

The most common standard, made Linux containers usable by the masses



© 2018 IBM Corporation

Rocket (rkt)

An emerging container standard from CoreOS, the company that developed etcd



Garden

Cloud Foundry component for creating and managing containers



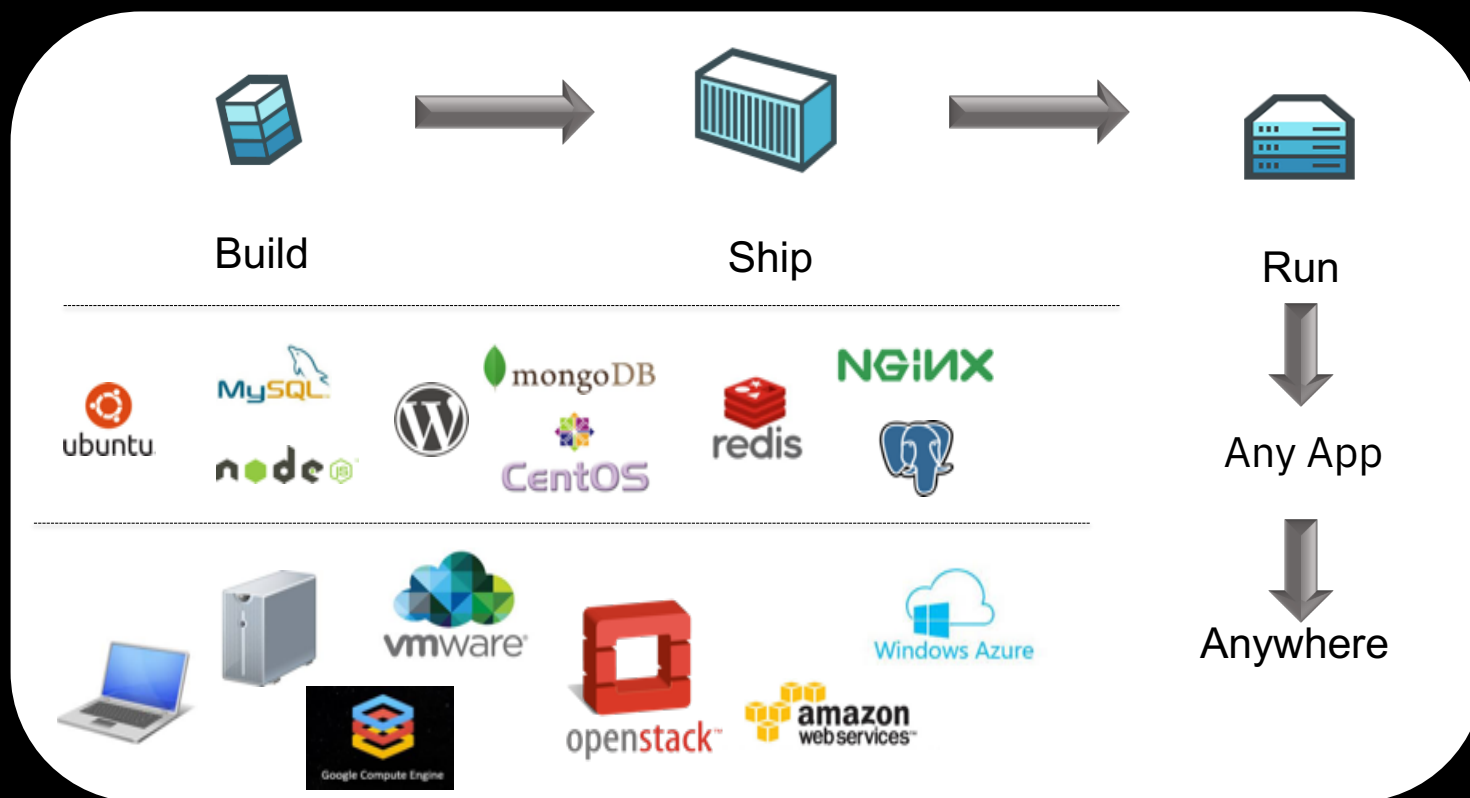
Open Container Initiative (OCI)

A Linux Foundation project that is developing a governed container standard



Docker mission

Docker is an **open platform** for building distributed applications for developers and system administrators.



Docker adoption

Enables application development efficiency, making deployment more efficient, and eliminating vendor lock-in with true portability

Open Contribution

2000+ contributors

#2 most popular project

185 community meet-up groups in 58 countries

Open Design

Contributors include IBM, Red Hat, Google, Microsoft, VMware, AWS, Rackspace, and others

Open Software

Launched March 2013

2.0+ billion downloads of Docker images

Open governance

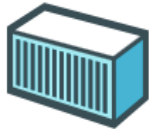
Docker, the Open Container Initiative (OCI), and the Cloud Native Computing Foundation (CNCF) are jointly developing container standards

Docker basic concepts

Image



A read-only snapshot of a container that is stored in a Docker registry and used as a template for building containers

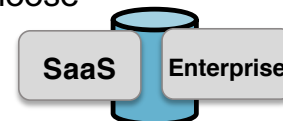


Container

The standard unit in which the application service resides or is transported

Registry

Available in SaaS or Enterprise to deploy anywhere you choose
Stores, distributes and shares container images



Engine



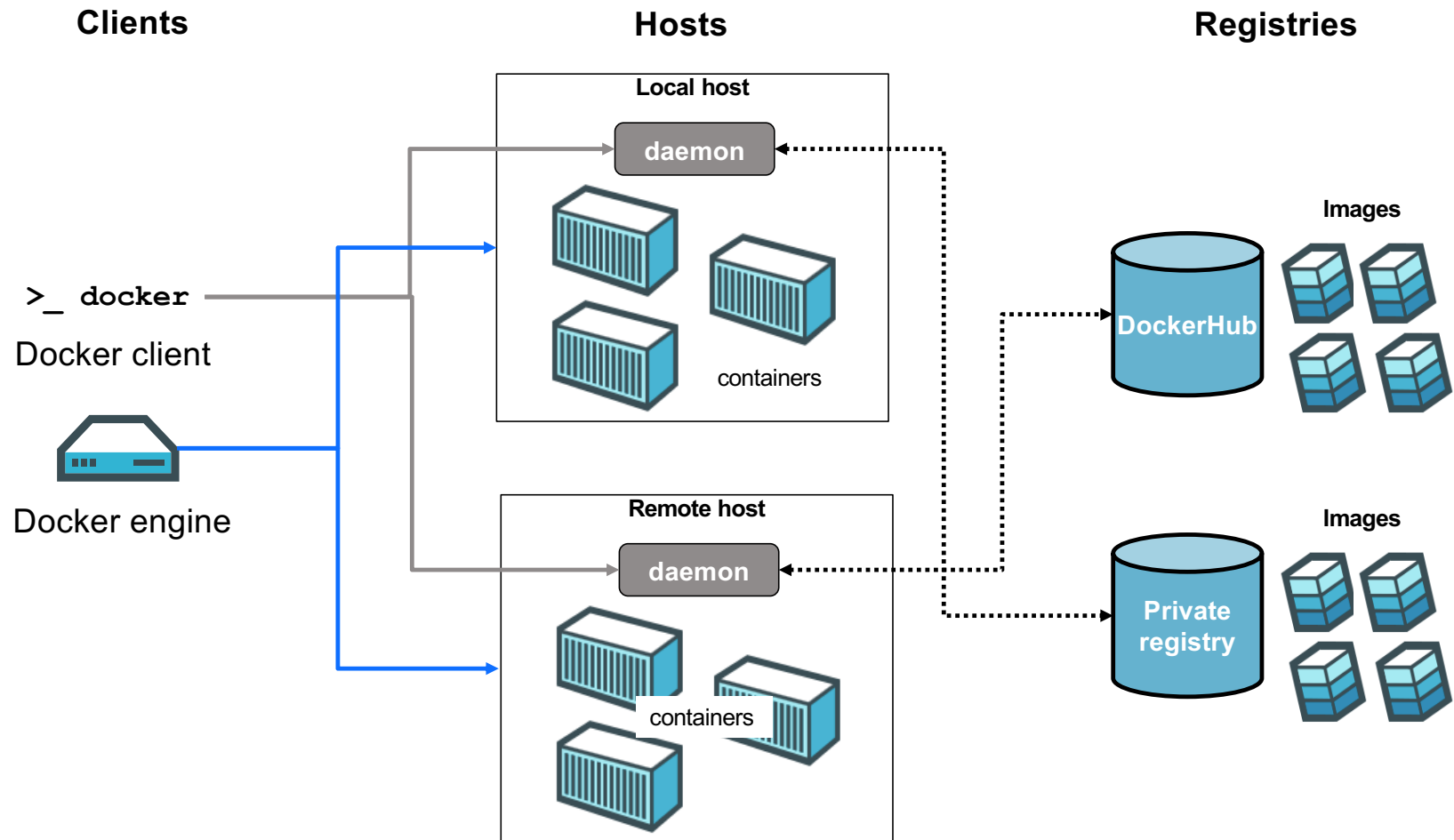
A program that creates, ships and runs application containers
Runs on any physical or virtual machine locally, in private, or public cloud

Client

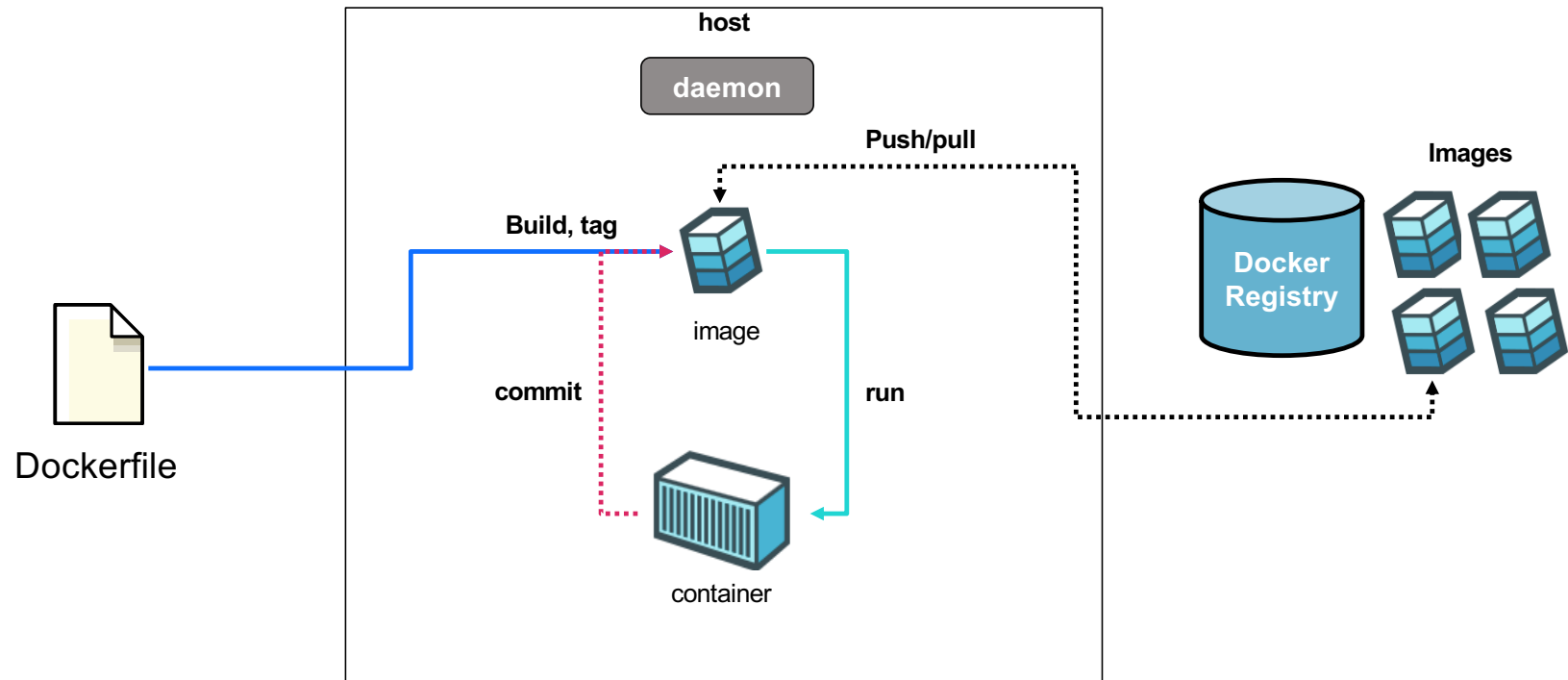
Communicates with engine to execute commands

```
>_ docker
```

Docker Architecture



Typical Workflow



Docker shared and layered file systems technology

Docker uses a copy-on-write (union) file system

New files and edits are only visible to current and above layers

Saves disk space and allows images to build faster

Maintains filesystem integrity by isolating the contents

