Docker

DevOps Training

Movement in the cloud

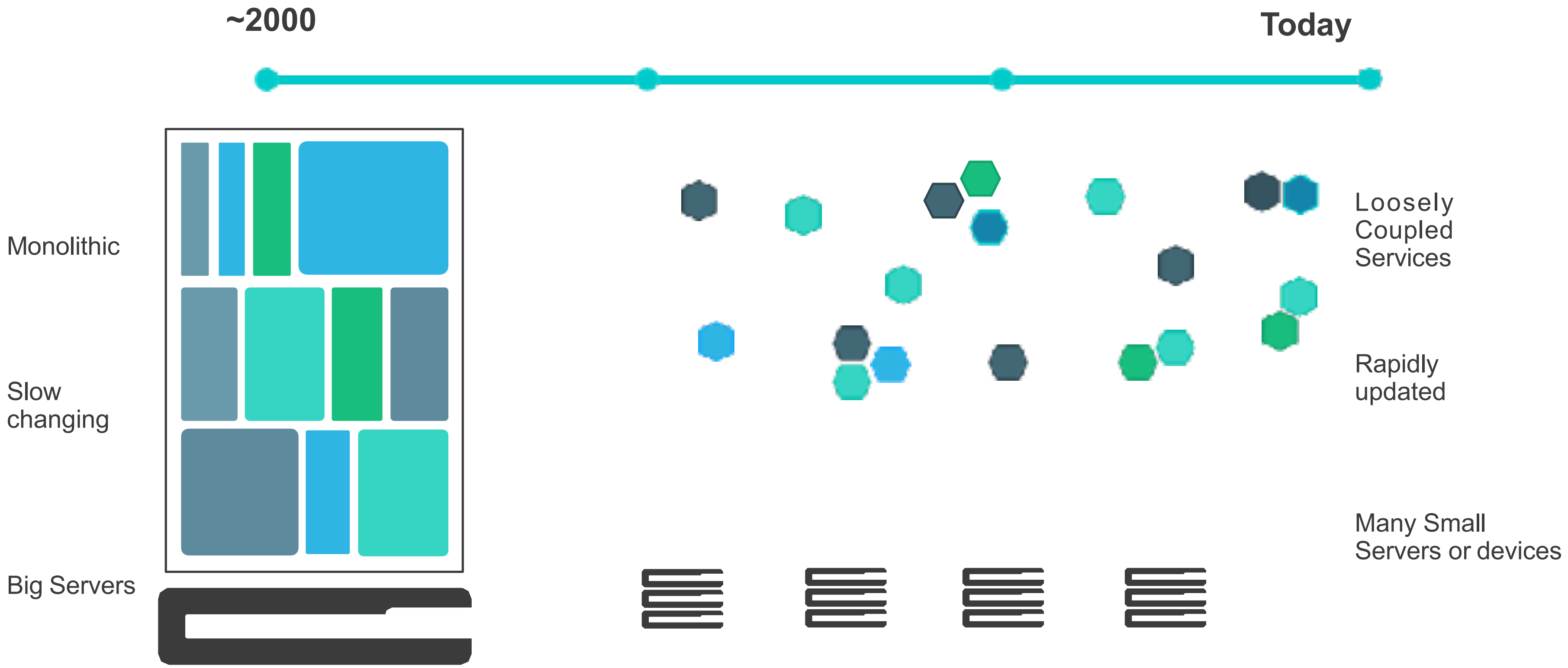
80%

Migrate workloads to cloud

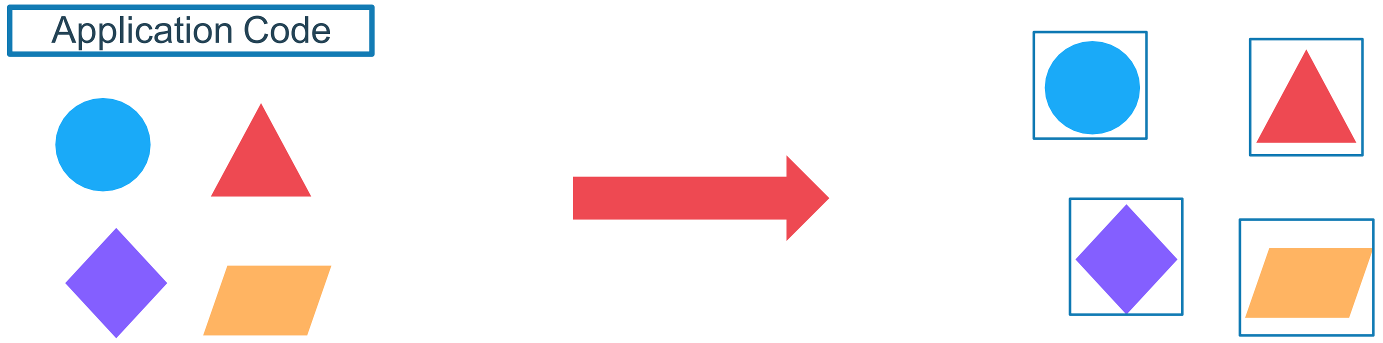
Portability across environments

Want to avoid cloud vendor lock-in

Applications are transforming



Application Modernization



Developer Issues:

- Minor code changes require full re-compile and re-test
- Application becomes single point of failure
- Application is difficult to scale

Microservices: Break application into separate operations

12-Factor Apps: Make the app independently scalable, stateless, highly available by design

Tug of War Between Developers and Ops

Developers

- Freedom to create and deploy apps fast
- Define and package application needs

IT Operations

- Quickly and flexibly respond to changing needs
- Standardize, secure, and manage

Organizations Must Deal with Diverse Technology

- Bare Metal
- On Premises
- Linux
- Traditional

+

- Virtual
- Cloud
- Windows
- Microservices

The Myth of Bi-Modal IT

	MICROSERVICES	TRADITIONAL APPS
Cloud or New Infrastructur e	You are either here..	▨
Old Infrastructure	▨	…or here

History of Docker

2004
Solaris Containers / Zones technology introduced

2008
Linux containers (LXC 1.0) introduced

2013
Solomon Hykes starts Docker as an internal project within dotCloud

Mar 2013
Docker released to open source

Feb 2016
Docker introduces first commercial product – now called Docker Enterprise Edition

Today
Open source community includes:
- 3,300+ contributors
- 43,000+ stars
- 12,000+ forks

Incredible adoption in just 4 years

14M
Docker Hosts

900K
Docker apps

77K%
Growth in Docker job listings

12B
Image pulls Over 390K% Growth

3300
Project Contributors

The Docker Family Tree

moby project

Open source **framework** for assembling core components that make a container platform

Intended for:
Open source contributors + ecosystem developers

docker
Enterprise Edition

Subscription-based, commercially supported **products** for delivering a secure software supply chain

Intended for:
Production deployments + Enterprise customers

docker
Community Edition

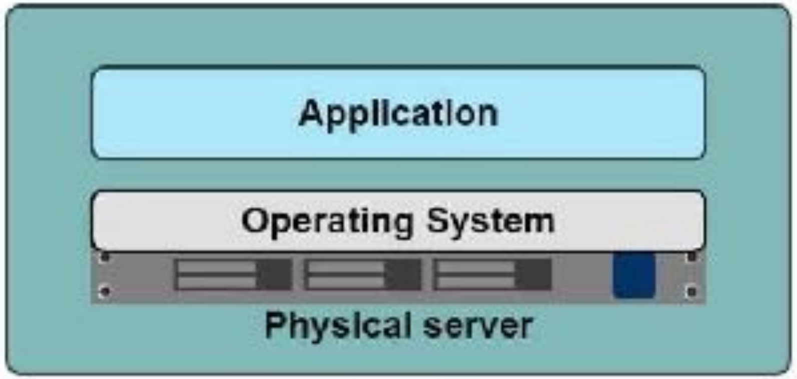
Free, community-supported **product** for delivering a container solution

Intended for:
Software dev & test

A History Lesson

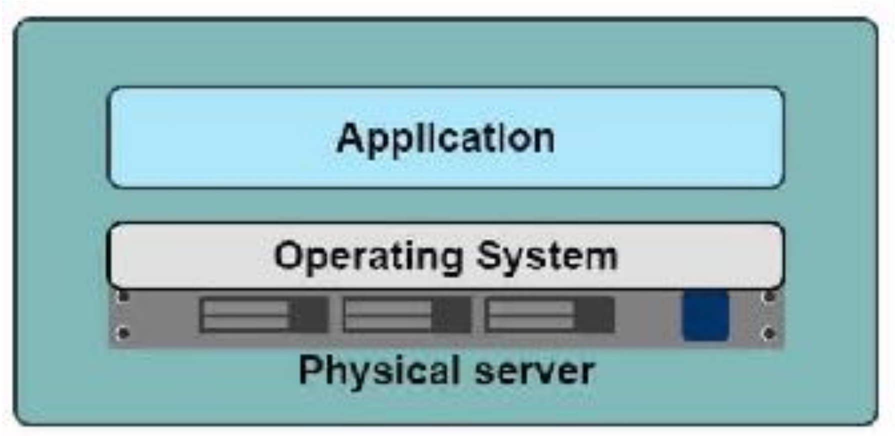
In the Dark Ages

One application on one physical server



Historical limitations of application deployment

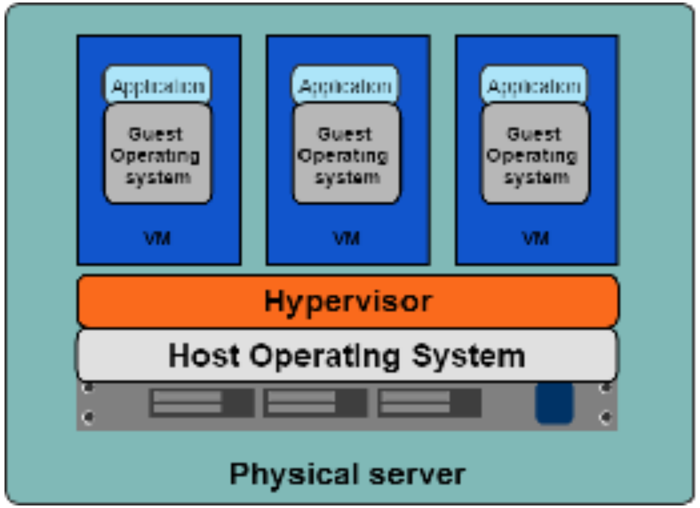
- Slow deployment times
- Huge costs
- Wasted resources
- Difficult to scale
- Difficult to migrate
- Vendor lock in



A History Lesson

Hypervisor-based Virtualization

- One physical server can contain multiple applications
- Each application runs in a virtual machine (VM)



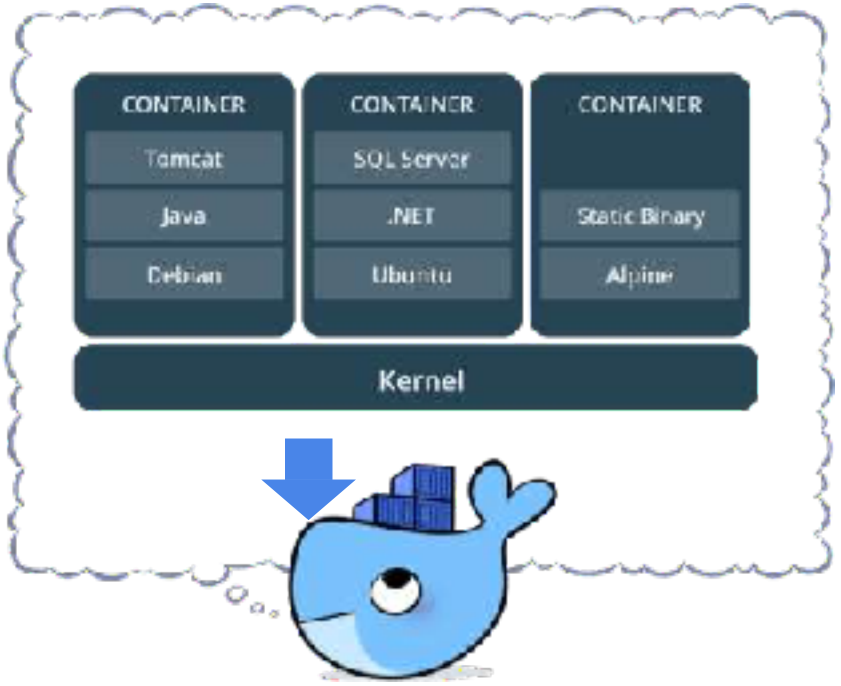
Benefits of VMs

- Better resource pooling
  - One physical machine divided into multiple virtual machines
- Easier to scale
- VMs in the cloud
  - Rapid elasticity
  - Pay as you go model

Limitations of VMs

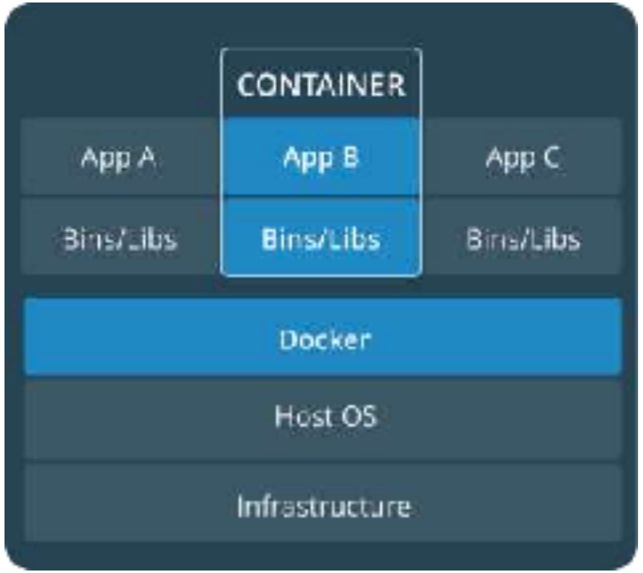
- Each VM stills requires
  - CPU allocation
  - Storage
  - RAM
  - An entire guest operating system
- The more VMs you run, the more resources you need
- Guest OS means wasted resources
- Application portability not guaranteed

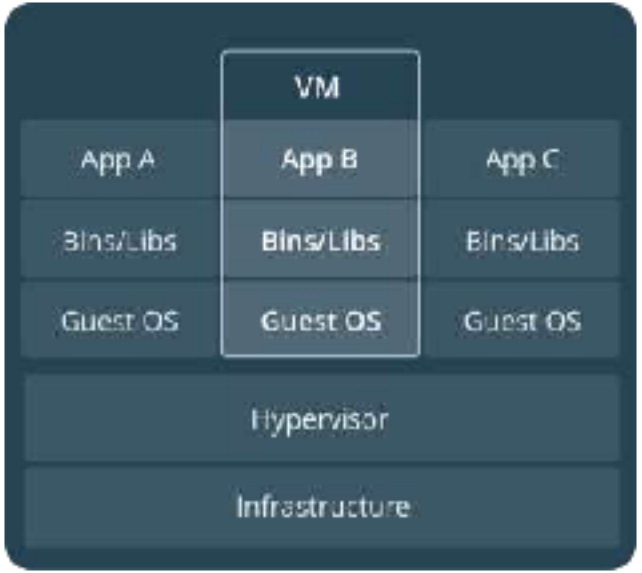
What is a container?



- Standardized packaging for software and dependencies
- Isolate apps from each other
- Share the same OS kernel
- Works with all major Linux and Windows Server

Comparing Containers and VMs

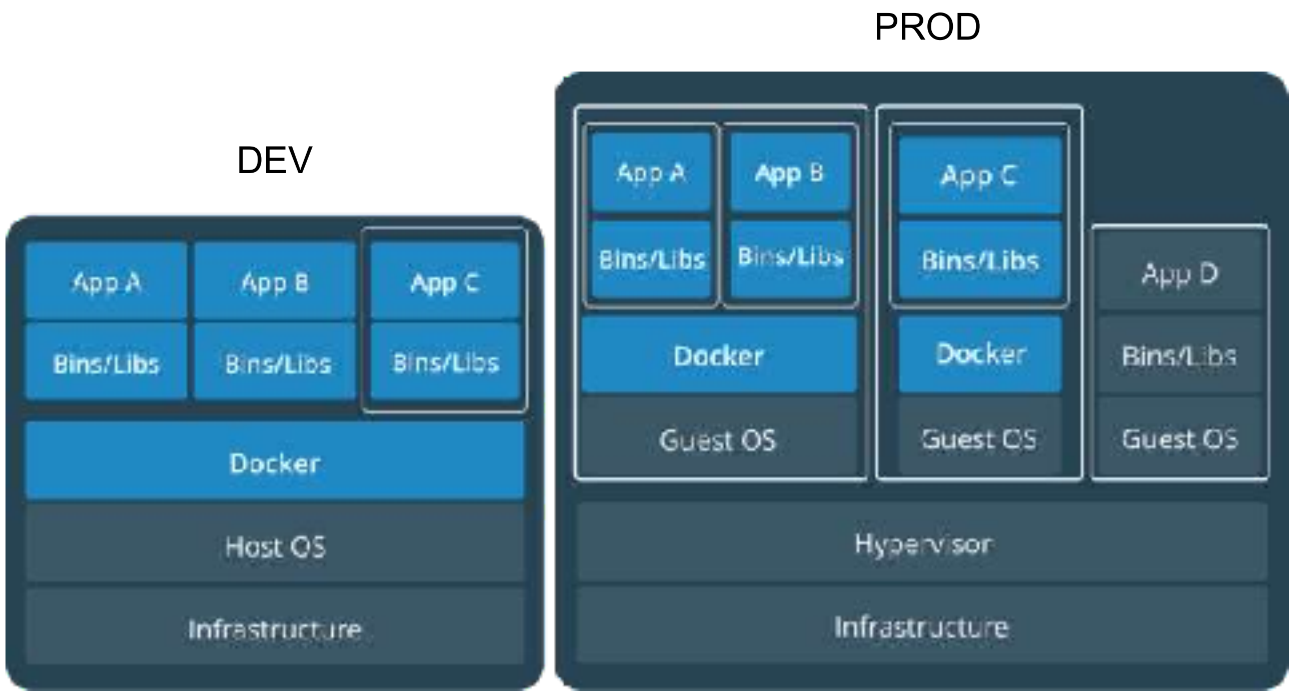




Containers are an app level construct

VMs are an infrastructure level construct to turn one machine into many servers

Containers and VMs together



Containers and VMs together provide a tremendous amount of flexibility for IT to optimally deploy and manage apps.

Key Benefits of Docker Containers

Speed

- No OS to boot = applications online in seconds

Portability

- Less dependencies between process layers = ability to move between infrastructure

Efficiency

- Less OS overhead
- Improved VM density

Docker Basics

Image

The basis of a Docker container. The content at rest.

Container

The image when it is 'running.' The standard unit for app service

Engine

The software that executes commands for containers. Networking and volumes are part of Engine. Can be clustered together.

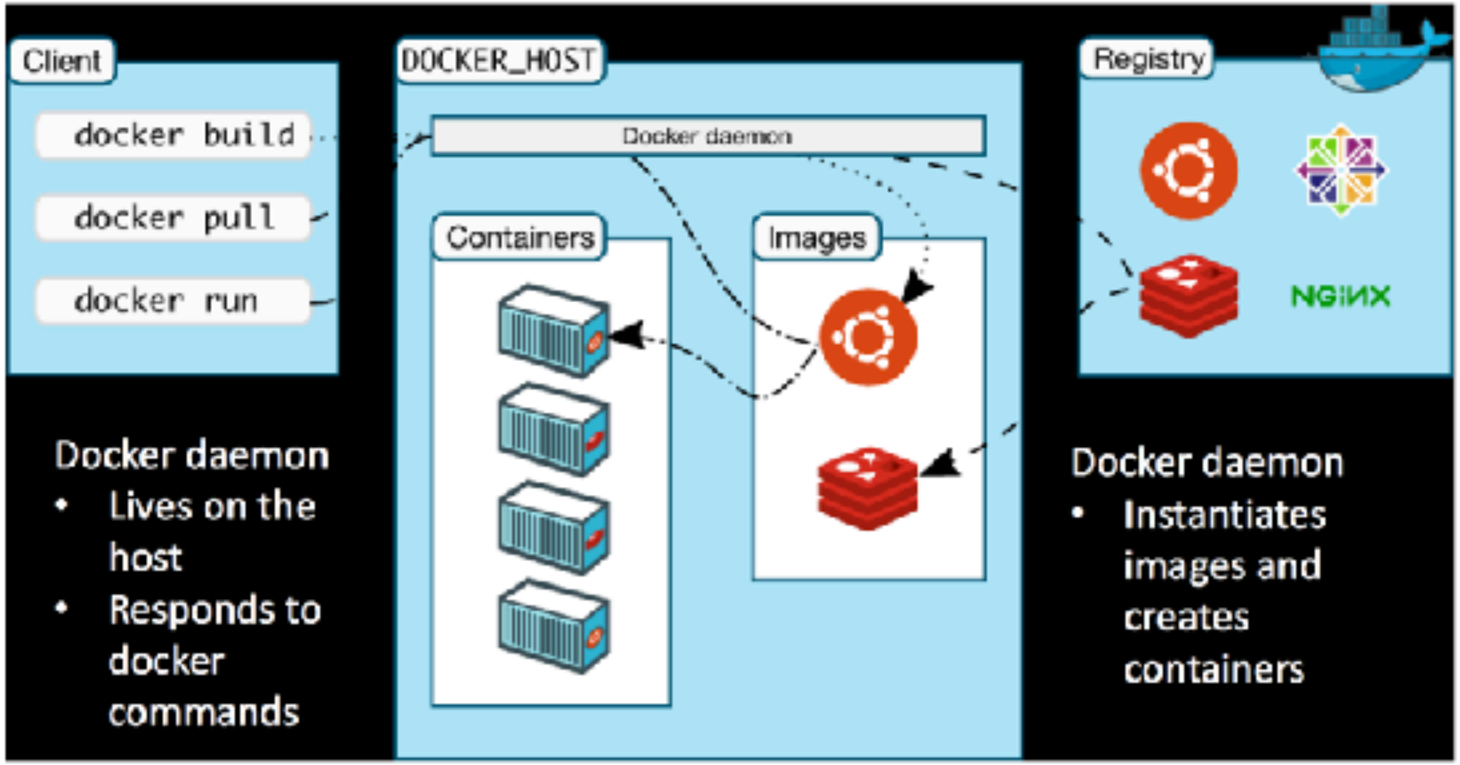
Registry

Stores, distributes and manages Docker images

Control Plane

Management plane for container and cluster orchestration

Docker Architecture



Docker daemon

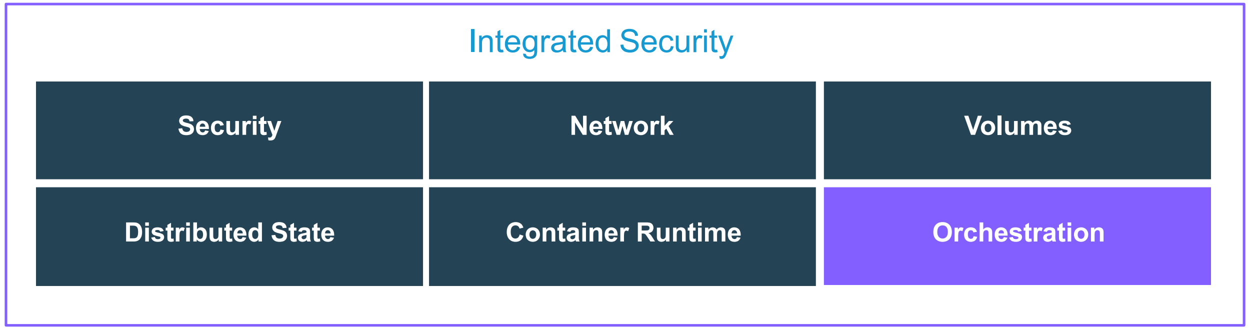
- Lives on the host
- Responds to docker commands

Docker daemon

- Instantiates images and creates containers

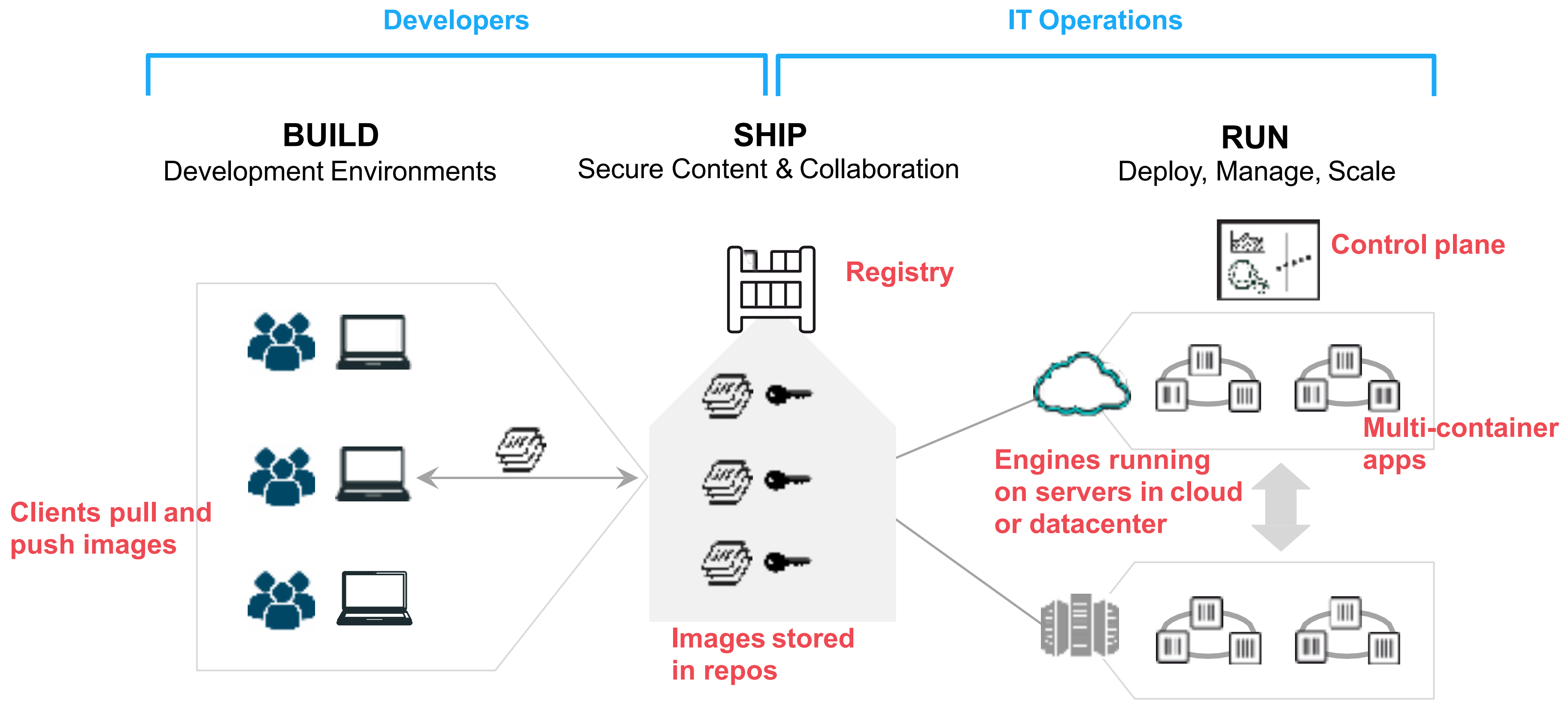
Image is instantiated to form container

Foundation: Docker Engine



Docker Engine

Containers as a Service



Docker Engine

- Docker Daemon
- Docker CLI

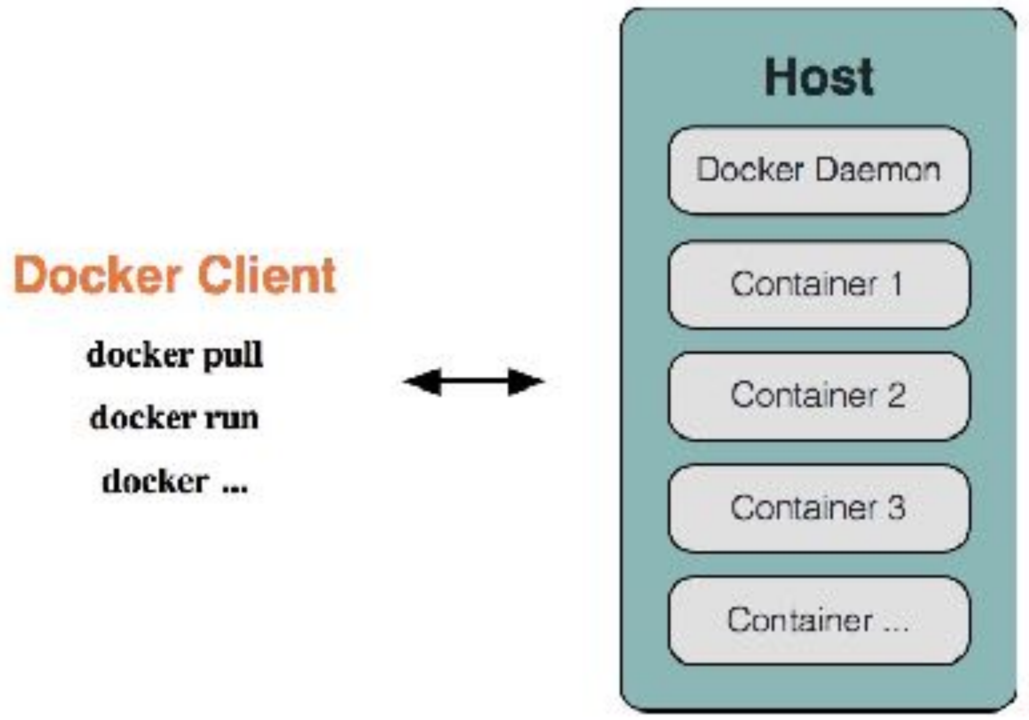
Docker Daemon

- Builds Images
- Runs and Manages Containers
- RESTful API

Docker CLI

docker build	# Build an image from a Dockerfile
docker images	# List all images on a Docker host
docker run	# Run an image
docker ps	# List all running and stopped instances
docker stop	# Stop a running instances
docker rm	# Remove an instance
docker rmi	# Remove an image

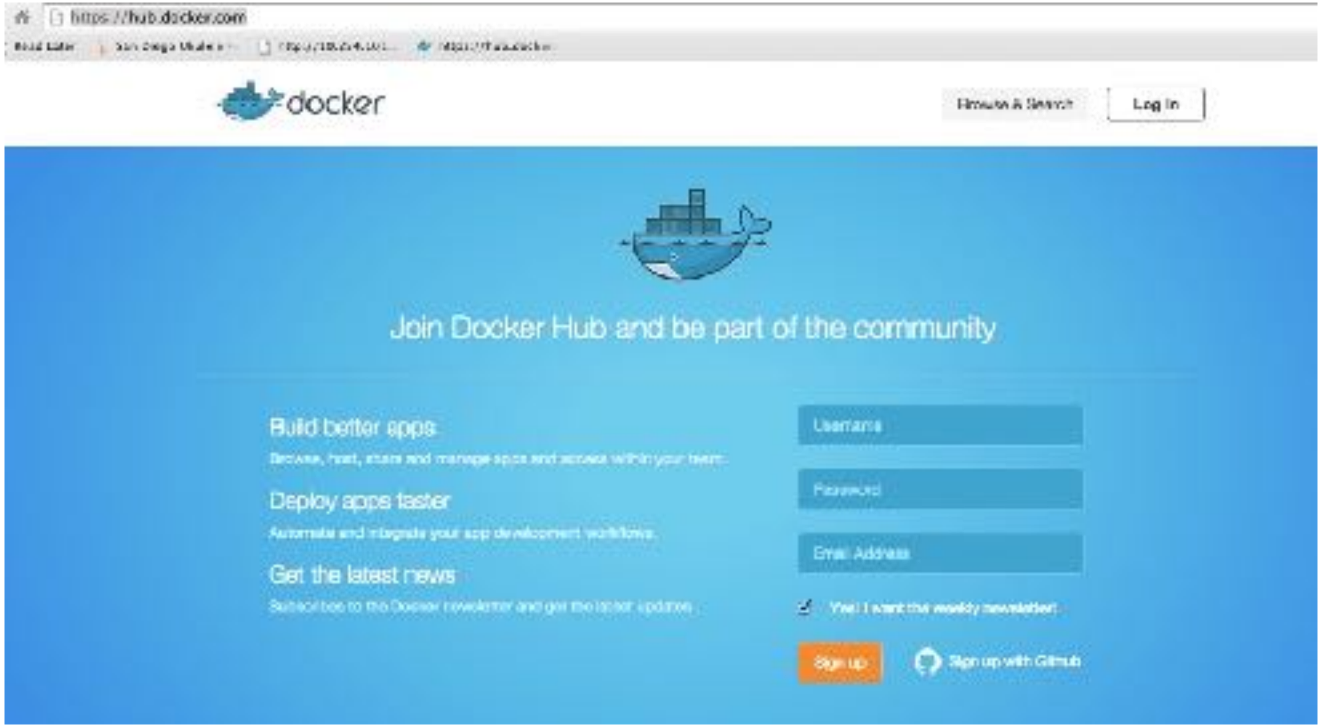
Docker Architecture



Docker Hub

- Provides Docker Services
- Library of public images
- Storage for your images
  - free for public images
  - cost for private images
- Automated builds(link github/bitbucket repo; trigger build on commit)

Docker Hub



https://hub.docker.com

Join Docker Hub and be part of the community

Build better apps

Browse, host, share and manage apps and services within your team.

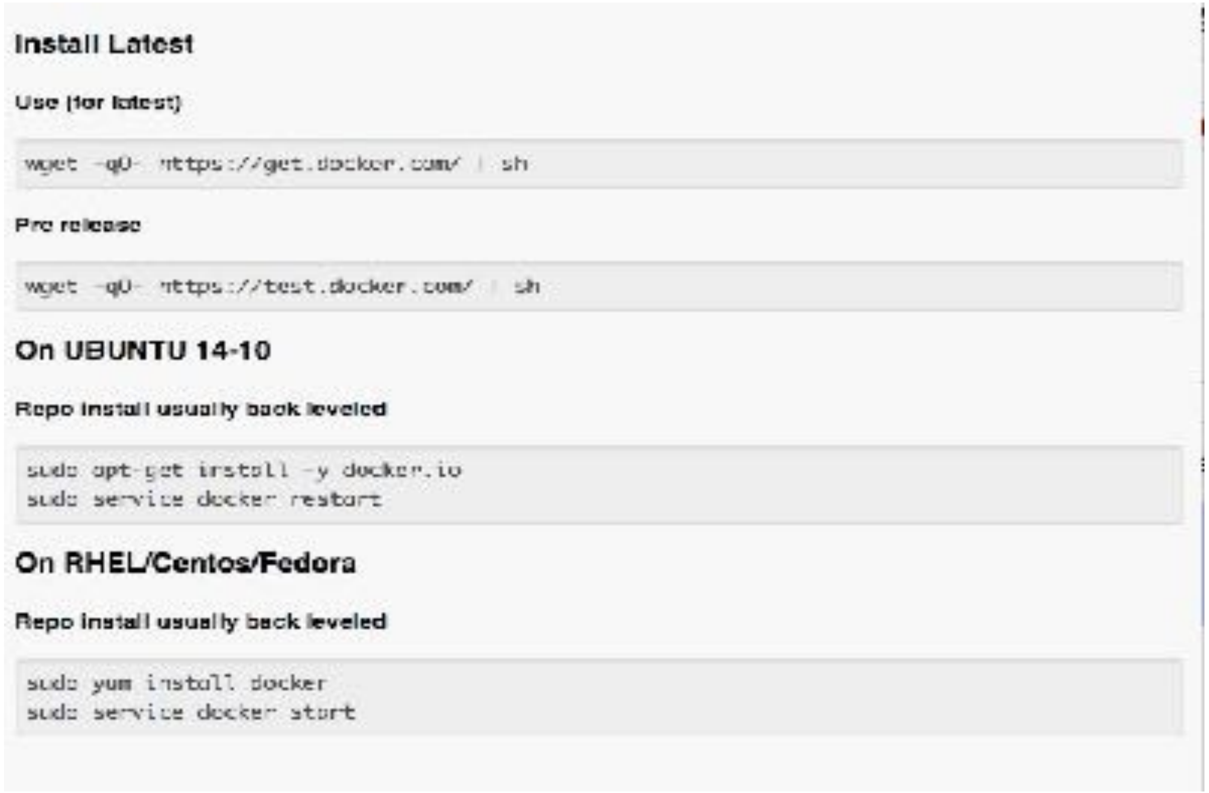
Deploy apps faster

Automate and integrate your app development workflows.

Get the latest news

Subscribe to the Docker newsletter and get the latest updates.

Docker Installation



Install Latest

Use (for latest)

```
wget -qO- https://get.docker.com/ | sh
```

Pre release

```
wget -qO- https://test.docker.com/ | sh
```

On UBUNTU 14-10

Repo install usually back leveled

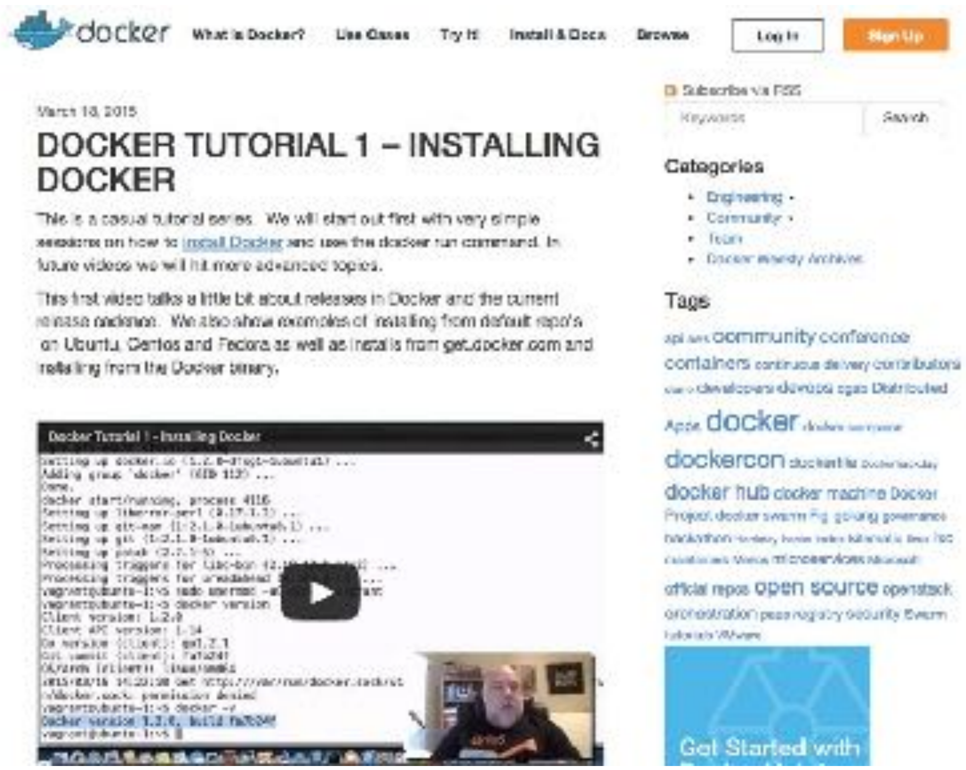
```
sudo apt-get install -y docker.io
sudo service docker restart
```

On RHEL/Centos/Fedora

Repo install usually back leveled

```
sudo yum install docker
sudo service docker start
```

Docker Installation



March 18, 2015

DOCKER TUTORIAL 1 – INSTALLING DOCKER

This is a casual tutorial series. We will start out first with very simple sessions on how to install Docker and use the docker run command. In future videos we will hit more advanced topics.

This first video talks a little bit about releases in Docker and the current release cadence. We also show examples of installing from default repo's on Ubuntu, Centos and Fedora as well as installs from get.docker.com and installing from the Docker binary.

Docker Tutorial 1 – Installing Docker

```
Setting up docker.io (1.2.0-0~trusty-0ubuntu1) ...
Adding group `docker' (GID 112) ...
Done.
docker start/running, process 4116
Setting up liberror-perl (0.17-1.1) ...
Setting up git-man (1:2.1.0-1ubuntu0.1) ...
Setting up git (1:2.1.0-1ubuntu0.1) ...
Setting up patch (2.7.1-5) ...
Processing triggers for libc-bin [illegible] ...
Processing triggers for ureadahead [illegible] ...
vagrant@ubuntu-1:~$ sudo usermod -a[illegible]rant
vagrant@ubuntu-1:~$ docker version
Client version: 1.2.0
Client API version: 1.14
Go version (client): go1.2.1
Git commit (client): fa7b24f
OS/Arch (client): linux/amd64
2015/03/16 14:23:58 Get http:///var/run/docker.sock/v1
.14/version: dial unix /var/run/docker.sock: permission denied
vagrant@ubuntu-1:~$ docker -v
Docker version 1.2.0, build fa7b24f
vagrant@ubuntu-1:~$
```

Categories

- Engineering
- Community
- Team
- Docker Weekly Archives

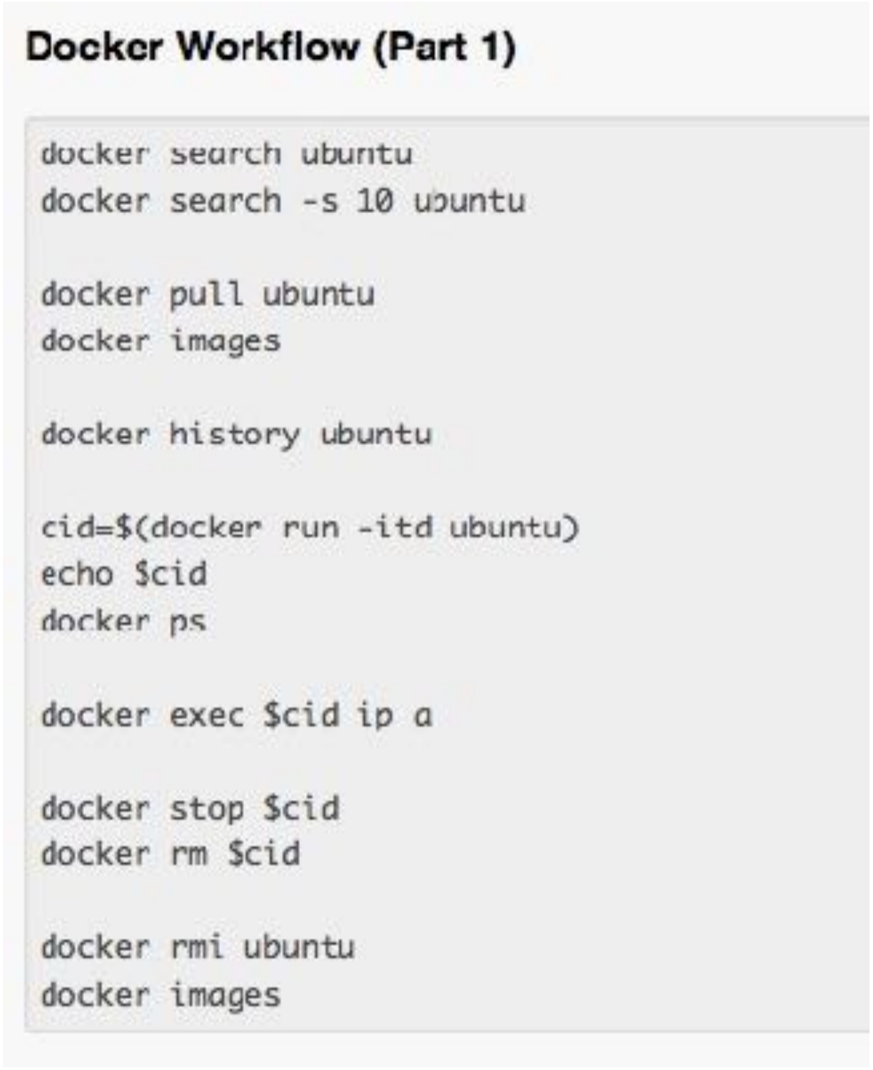
Tags

api aws community conference containers continuous delivery contributors [illegible] developers devops dgas Distributed Apps docker docker compose dockercon dockerfile dockersaturday docker hub docker machine Docker Project docker swarm Fig golang governance hackathon [illegible] microservices [illegible] official repos open source openstack orchestration paas registry security Swarm tutorials VMware

Get Started with

Docker Platform Workflow

- Find an Image on Docker Hub
- Pull an Image from Docker Hub
- Run an Image on Docker Host
- Stop an Instance
- Remove an Instance
- Remove an Image



Docker Workflow (Part 1)

```
docker search ubuntu
docker search -s 10 ubuntu

docker pull ubuntu
docker images

docker history ubuntu

cid=$(docker run -itd ubuntu)
echo $cid
docker ps

docker exec $cid ip a

docker stop $cid
docker rm $cid

docker rmi ubuntu
docker images
```

THANK YOU!

training@laksans.com