Docker and Kubernetes

Session 1

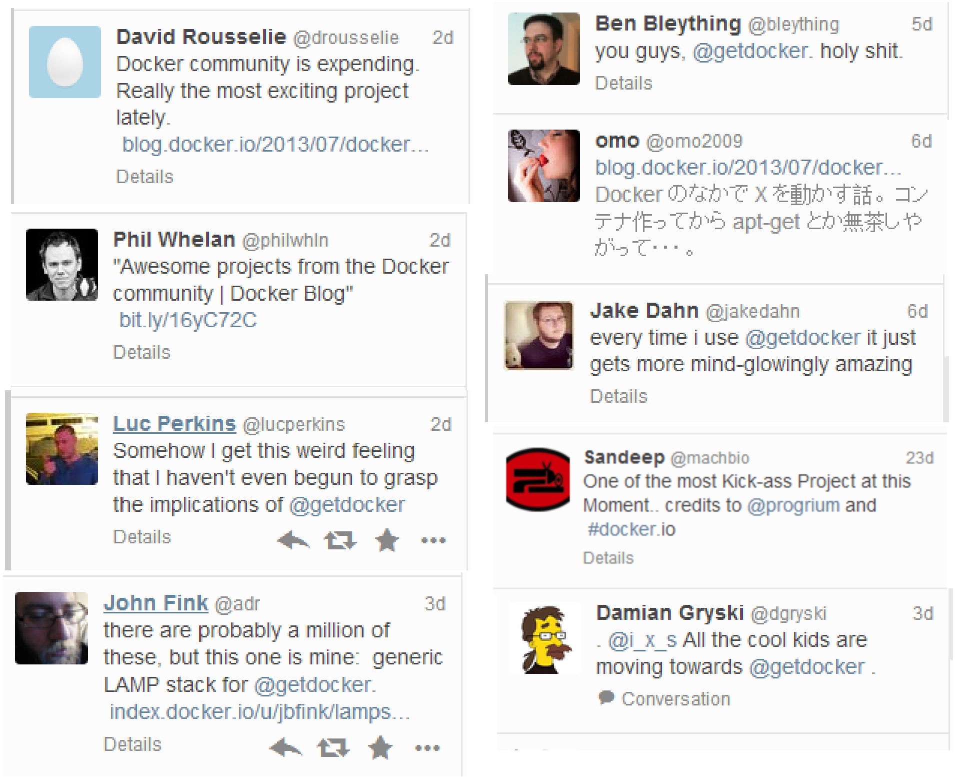
Getting started with Docker

Agenda

- Introduction
- Docker Overview
- Getting Started with Docker
- Basic commands

In the 8 months since we launched

- >200,000 pulls
- >7,500 github stars
- >200 significant contributors
- >200 projects built on top of docker
  - ✓ UIs, mini-PaaS, Remote Desktop….
- 1000's of Dockerized applications
  - ✓ Memcached, Redis, Node.js…and Hadoop
- Integration in Jenkins, Travis, Chef, Puppet, Vagrant and OpenStack
- Meetups arranged around the world…with organizations like Ebay, Cloudflare, Yandex, and Rackspace presenting on their use of Docker



David Rousselie @drousselie 2d
Docker community is expending. Really the most exciting project lately.
blog.docker.io/2013/07/docker…
Details

Phil Whelan @philwhln 2d
"Awesome projects from the Docker community | Docker Blog"
bit.ly/16yC72C
Details

Luc Perkins @lucperkins 2d
Somehow I get this weird feeling that I haven't even begun to grasp the implications of @getdocker
Details

John Fink @adr 3d
there are probably a million of these, but this one is mine: generic LAMP stack for @getdocker.
index.docker.io/u/jbfink/lamps…
Details

Ben Bleything @bleything 5d
you guys, @getdocker. holy shit.
Details

omo @omo2009 6d
blog.docker.io/2013/07/docker…
Docker のなかで X を動かす話。コンテナ作ってから apt-get とか無茶しやがって・・・。

Jake Dahn @jakedahn 6d
every time i use @getdocker it just gets more mind-glowingly amazing
Details

Sandeep @machbio 23d
One of the most Kick-ass Project at this Moment.. credits to @progrium and #docker.io
Details

Damian Gryski @dgryski 3d
. @i_x_s All the cool kids are moving towards @getdocker .
Conversation

WHY ALL THE EXCITEMENT?

The Challenge

Multiplicity of Stacks

Static website

nginx 1.5 + modsecurity + openssl + bootstrap 2

User DB

postgresql + pgv8 + v8

Queue

Redis + redis-sentinel

Analytics DB

hadoop + hive + thrift + OpenJDK

Background workers

Python 3.0 + celery + pyredis + libcurl + ffmpeg + libopencv + nodejs + phantomjs

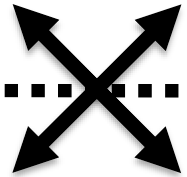
Web frontend

Ruby + Rails + sass + Unicorn

API endpoint

Python 2.7 + Flask + pyredis + celery + psycopg + postgresql-client

Do services and apps interact appropriately?



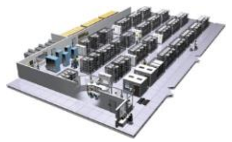
Multiplicity of hardware environments

Development VM

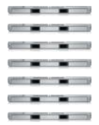
QA server

Customer Data Center

Public Cloud



Production Cluster



Disaster recovery

Production Servers

Contributor's laptop

Can I migrate smoothly and quickly?

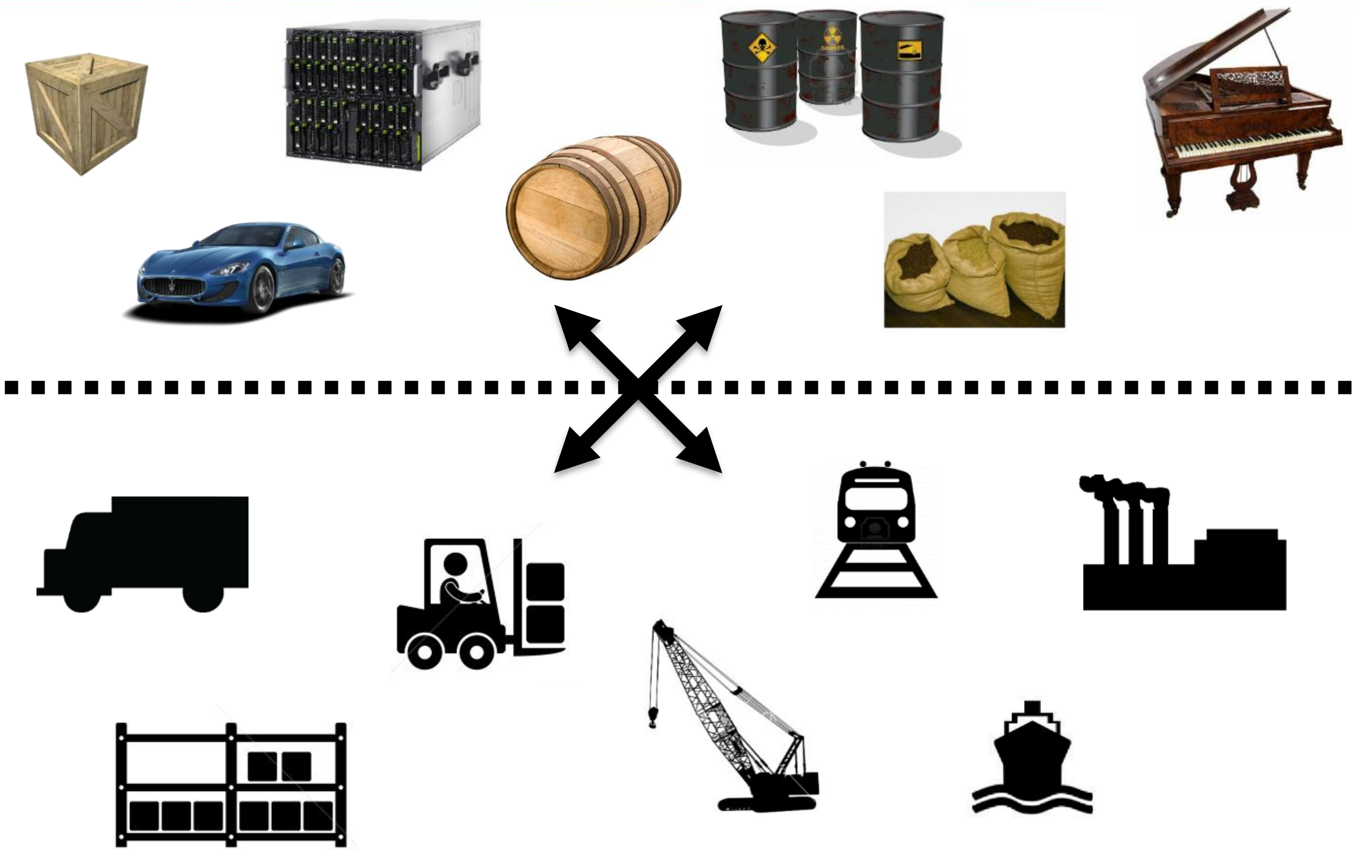
The Matrix From Hell

	Development VM	QA Server	Single Prod Server	Onsite Cluster	Public Cloud	Contributor’s laptop	Customer Servers
Static website	?	?	?	?	?	?	?
Web frontend	?	?	?	?	?	?	?
Background workers	?	?	?	?	?	?	?
User DB	?	?	?	?	?	?	?
Analytics DB	?	?	?	?	?	?	?
Queue	?	?	?	?	?	?	?

Cargo Transport Pre-1960

Multiplicity of Goods

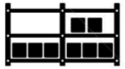
Do I worry about how goods interact (e.g. coffee beans next to spices)



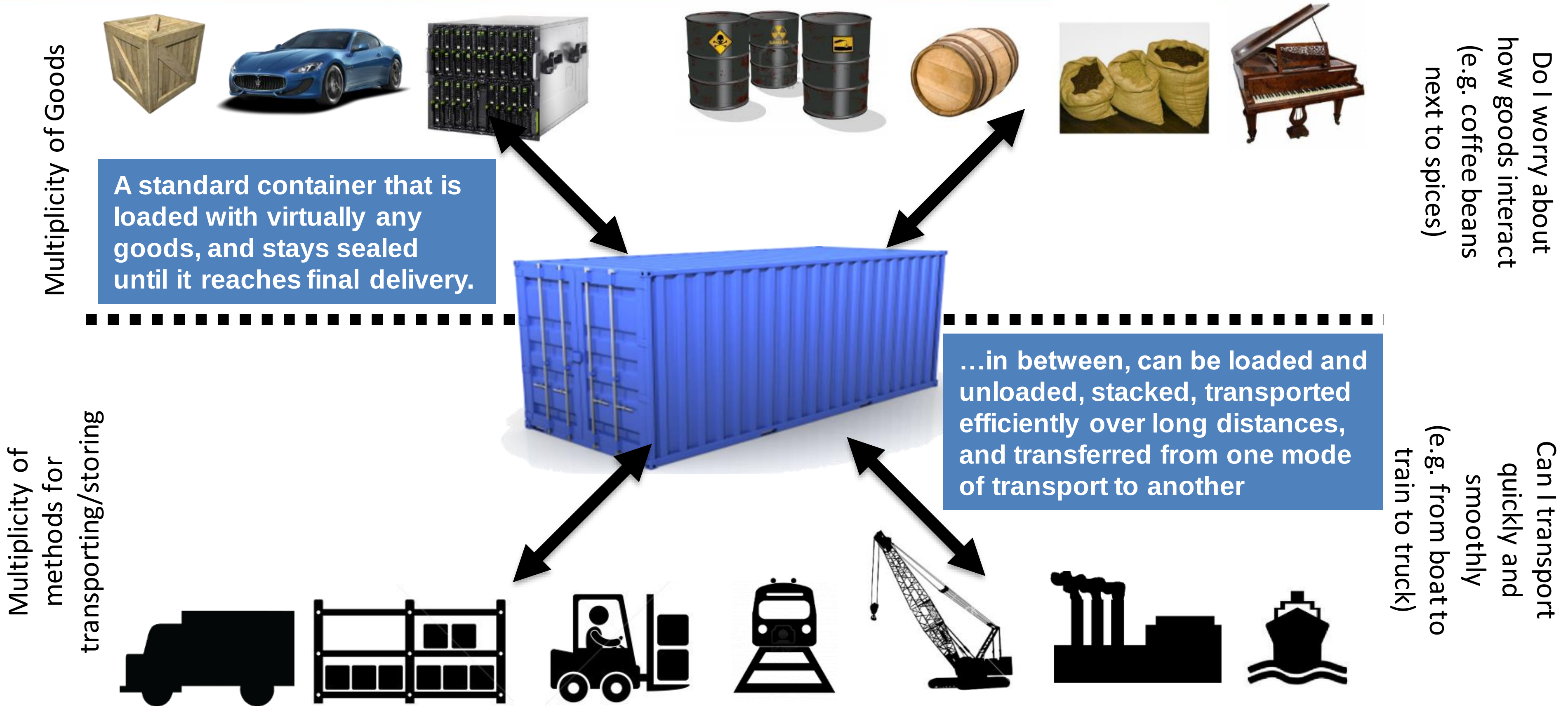
Multipilicity of methods for transporting/storing

Can I transport quickly and smoothly (e.g. from boat to train to truck)

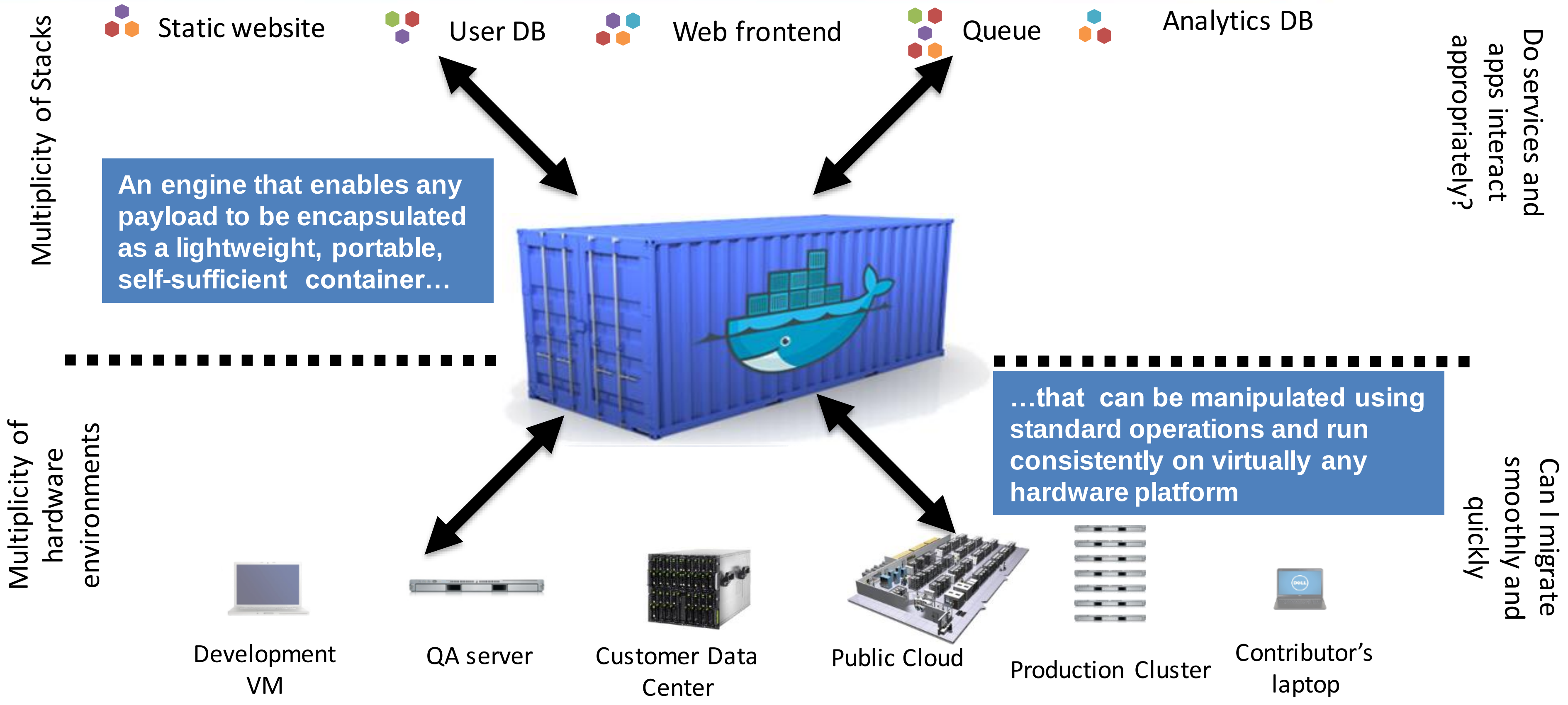
Also a matrix from hell

	?	?	?	?	?	?	?
	?	?	?	?	?	?	?
	?	?	?	?	?	?	?
	?	?	?	?	?	?	?
	?	?	?	?	?	?	?
	?	?	?	?	?	?	?
							

Solution: Intermodal Shipping Container


Multiplicity of Goods
A standard container that is loaded with virtually any goods, and stays sealed until it reaches final delivery.
Do I worry about how goods interact (e.g. coffee beans next to spices)
Multiplicity of methods for transporting/storing
…in between, can be loaded and unloaded, stacked, transported efficiently over long distances, and transferred from one mode of transport to another
Can I transport quickly and smoothly (e.g. from boat to train to truck)

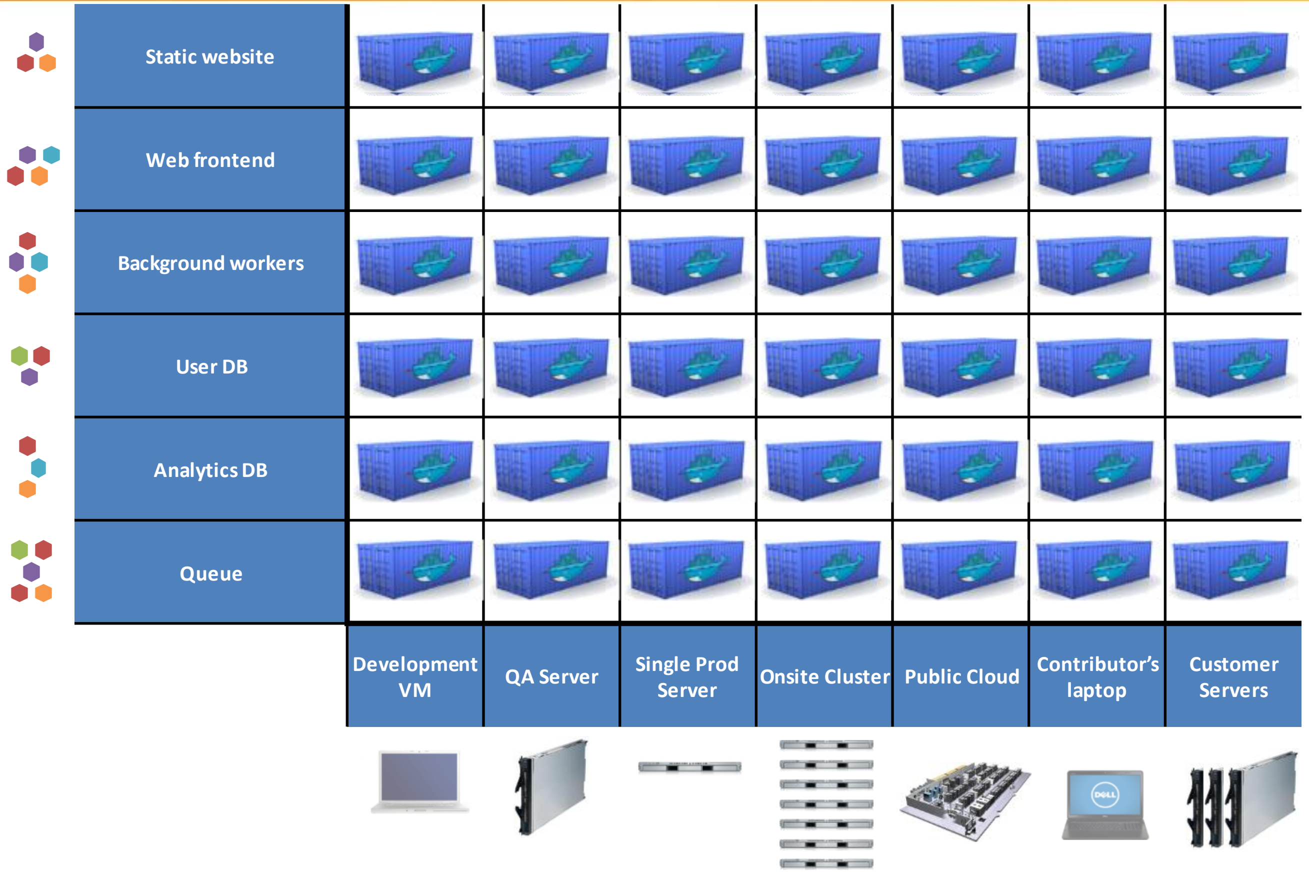
Docker is a shipping container system for code



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

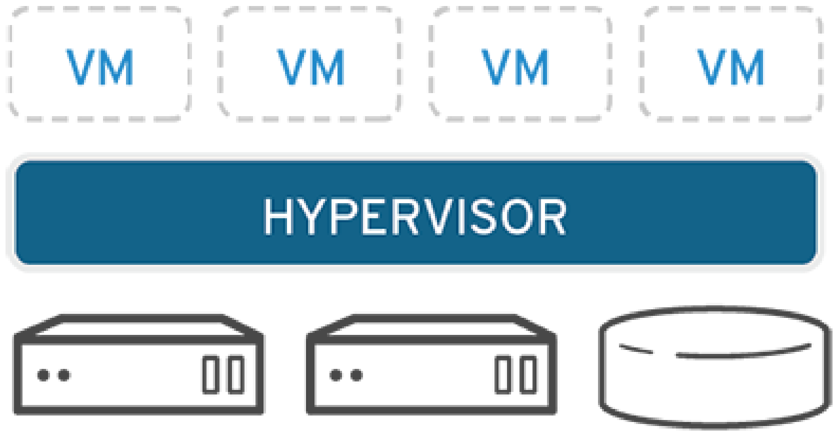
…that can be manipulated using standard operations and run consistently on virtually any hardware platform

Docker eliminates the matrix from Hell



Virtualization

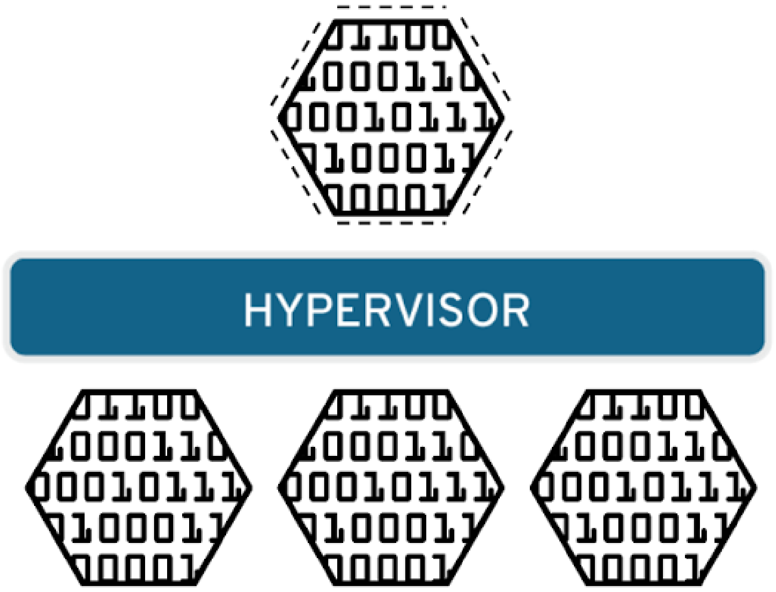
Virtualization is technology that lets you create useful IT services using resources that are traditionally bound to hardware. It allows you to use a physical machine’s full capacity by distributing its capabilities among many users or environments.



Types of virtualization

Data virtualization

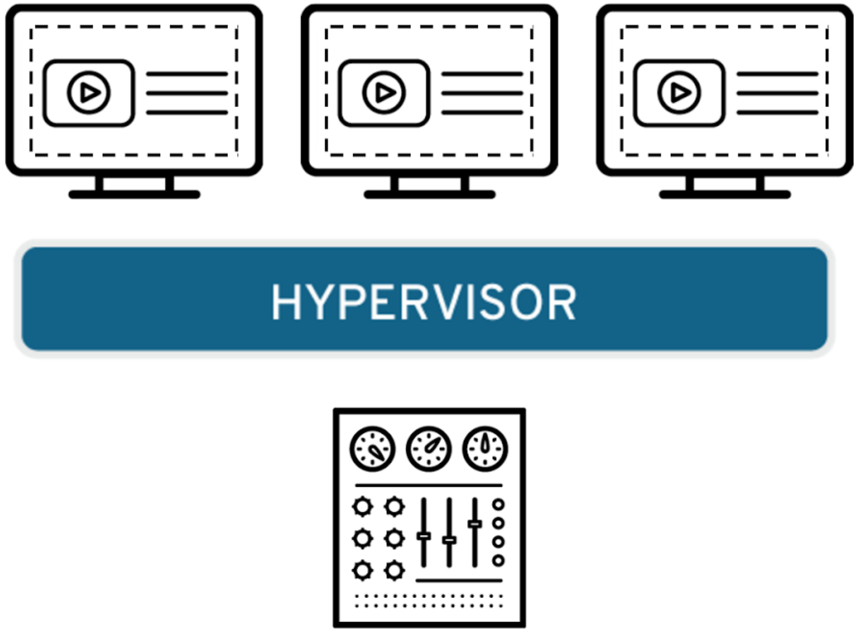
Data that's spread all over can be consolidated into a single source. Data virtualization allows companies to treat data as a dynamic supply—providing processing capabilities that can bring together data from multiple sources, easily accommodate new data sources, and transform data according to user needs



Types of virtualization

Desktop virtualization

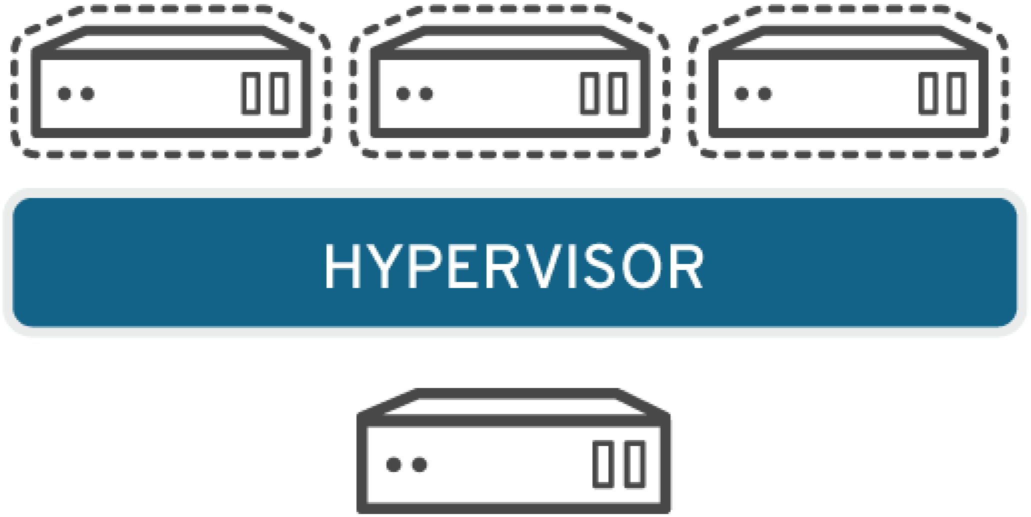
Easily confused with operating system virtualization—which allows you to deploy multiple operating systems on a single machine—desktop virtualization allows a central administrator (or automated administration tool) to deploy simulated desktop environments to hundreds of physical machines at once



Types of virtualization

Server virtualization

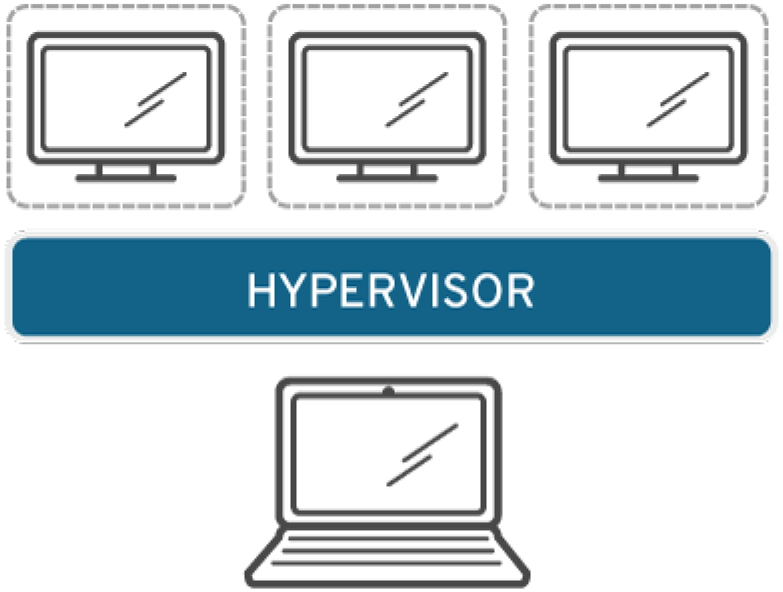
Servers are computers designed to process a high volume of specific tasks really well so other computers—like laptops and desktops—can do a variety of other tasks. Virtualizing a server lets it to do more of those specific functions and involves partitioning it so that the components can be used to serve multiple functions.



Types of virtualization

Operating system virtualization

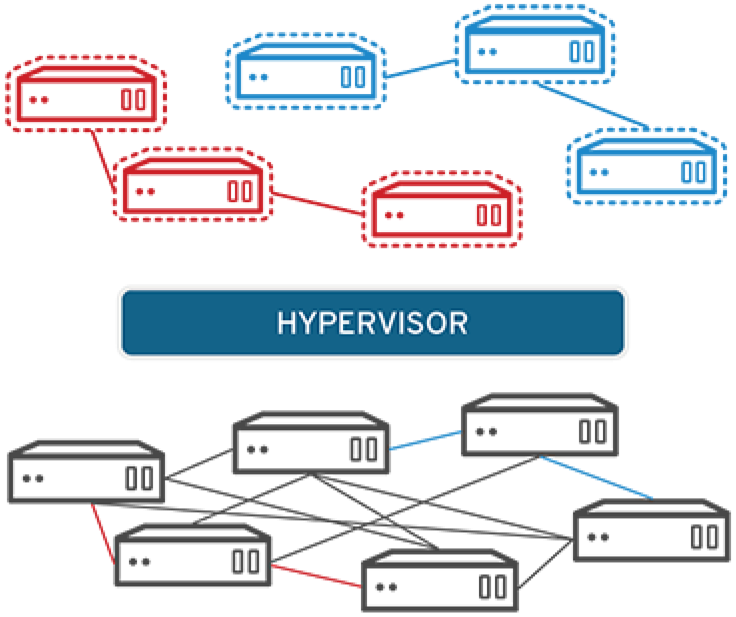
Operating system virtualization happens at the kernel—the central task managers of operating systems. It's a useful way to run Linux and Windows environments side-by-side



Types of virtualization

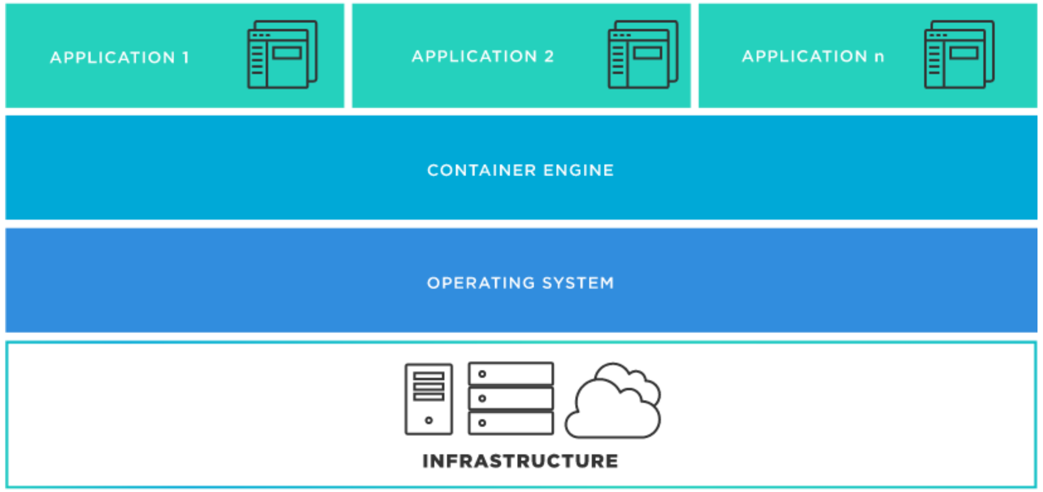
Network functions virtualization

Network functions virtualization (NFV) separates a network's key functions (like directory services, file sharing, and IP configuration) so they can be distributed among environments.

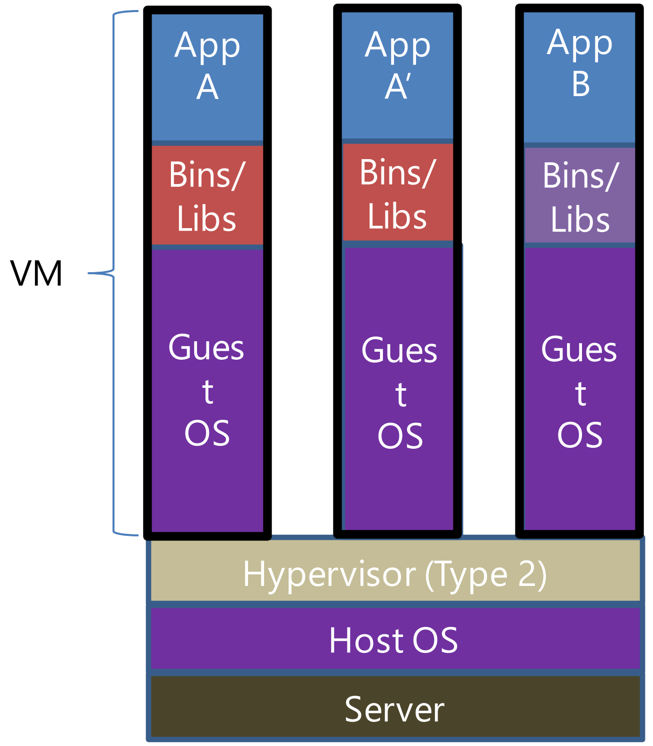


What is containerization?

Containerization is the packaging of software code with just the operating system (OS) libraries and dependencies required to run the code to create a single lightweight executable—called a container—that runs consistently on any infrastructure

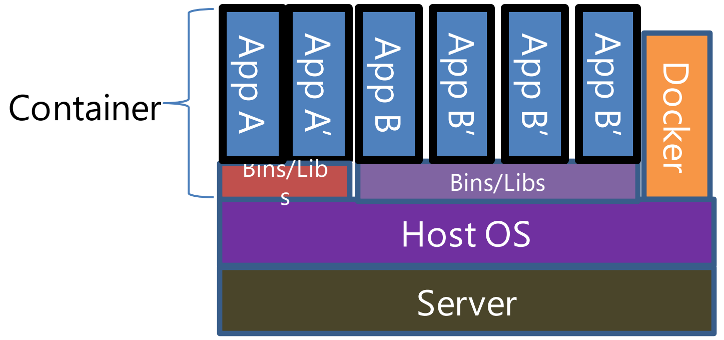


Containers vs. VMs

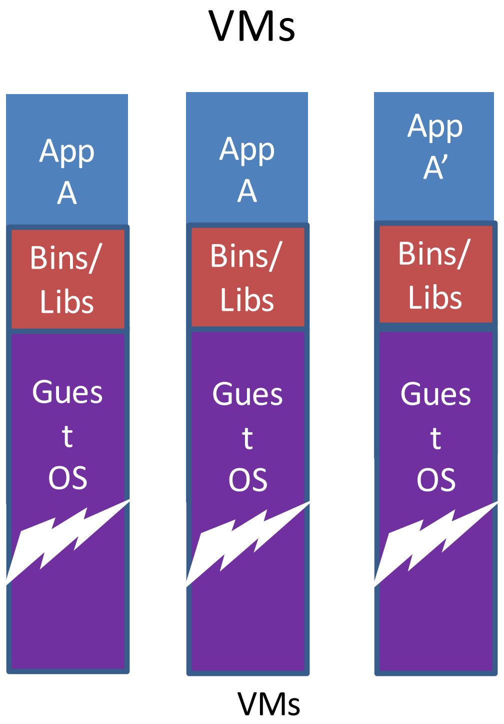


Containers are isolated, but share OS and, where appropriate, bins/libraries

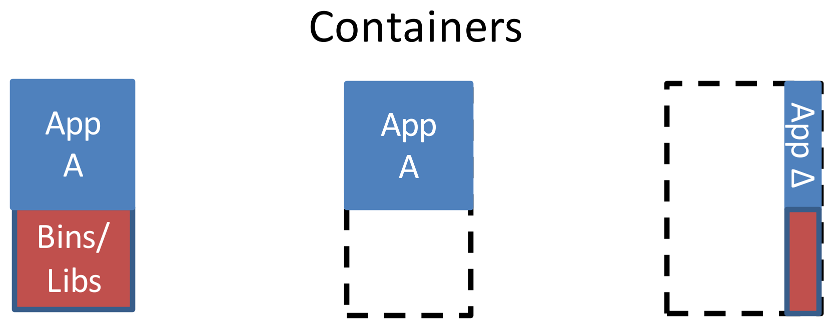
...result is significantly faster deployment, much less overhead, easier migration, faster restart



Why are Docker containers lightweight?



Every app, every copy of an app, and every slight modification of the app requires a new virtual server



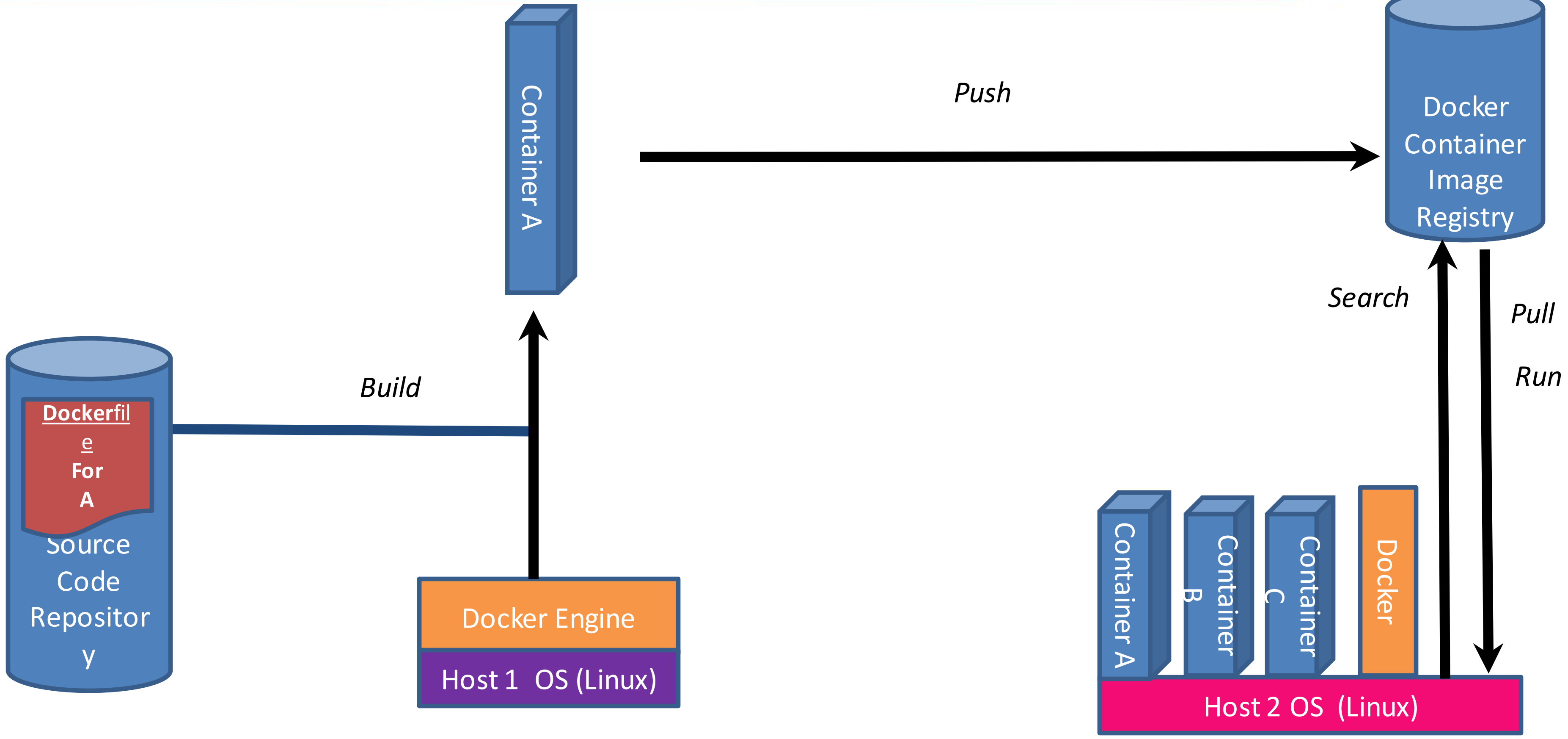
Original App
(No OS to take up space, resources, or require restart)

Copy of App
No OS. Can Share bins/libs

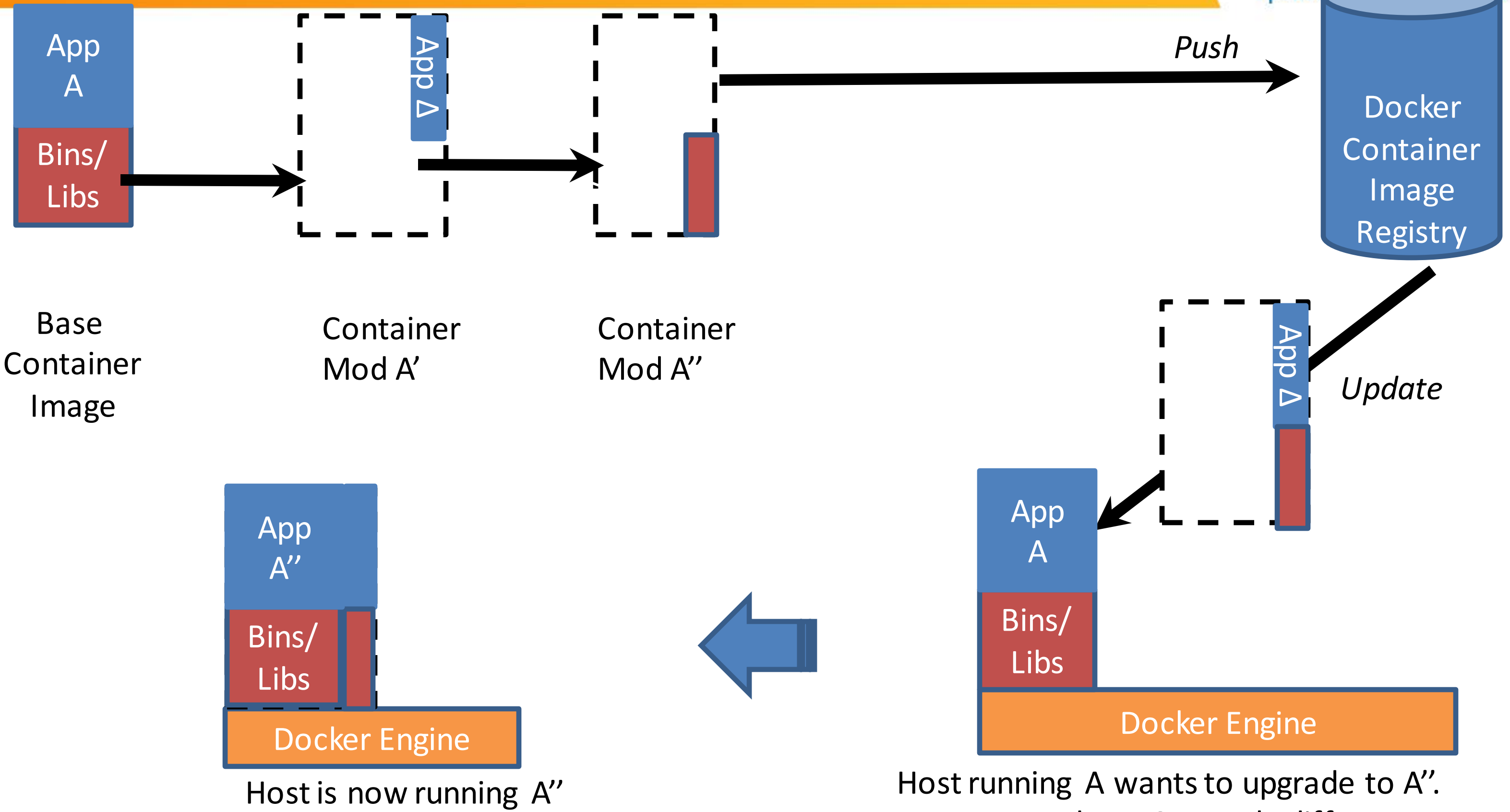
Modified App

Copy on write capabilities allow us to only save the diffs Between container A and container A'

What are the basics of the Docker system?



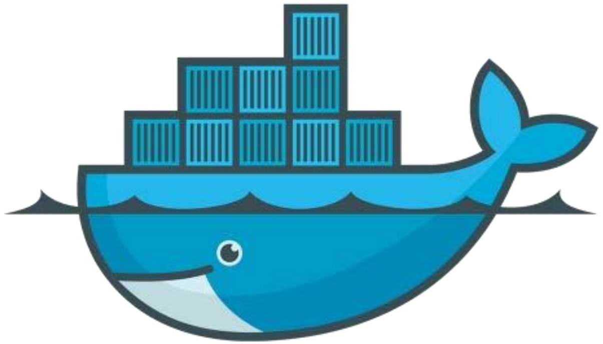
Changes and Updates



Host is now running A''

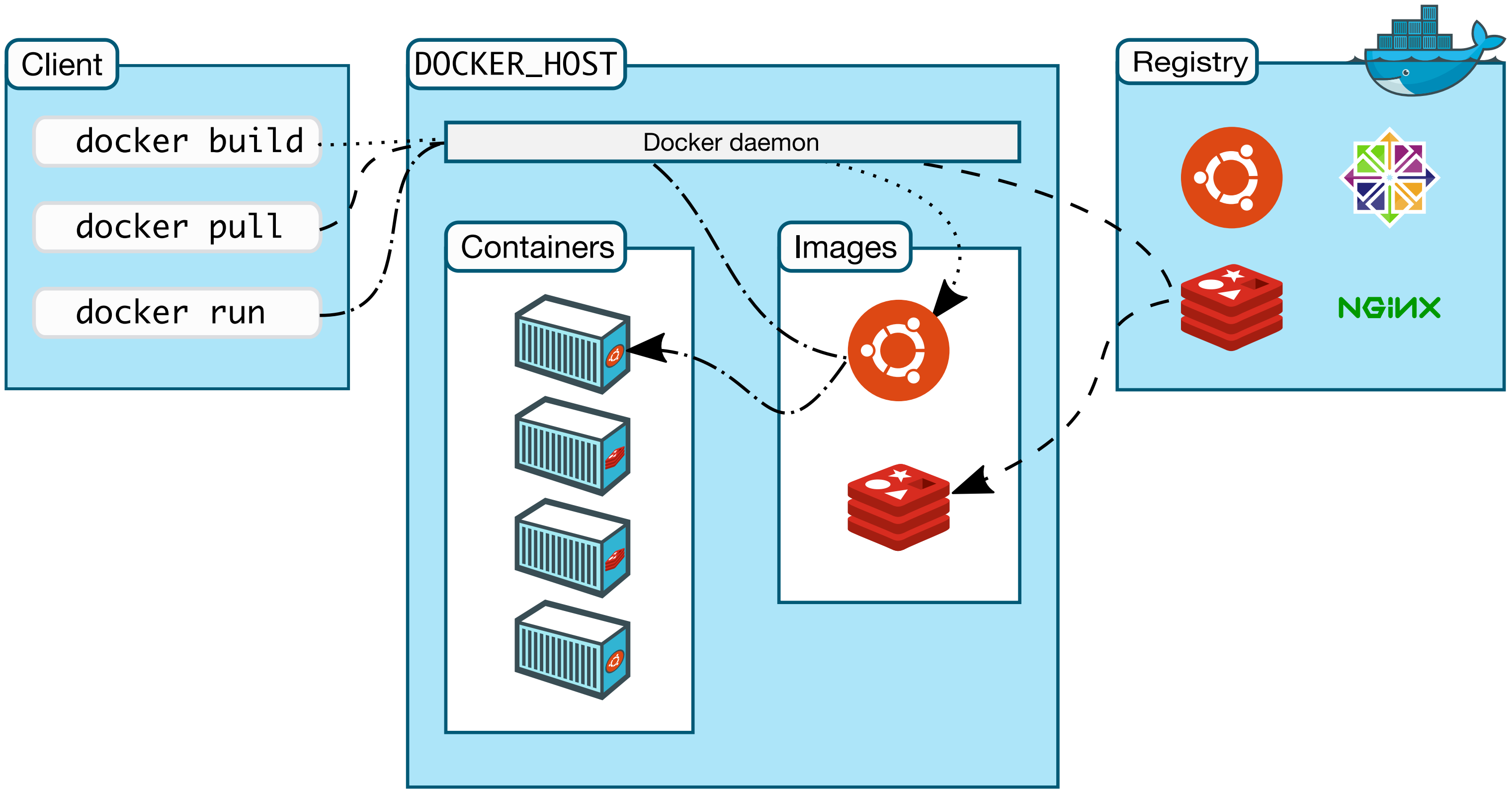
Host running A wants to upgrade to A''. Requests update. Gets only diffs

What Is Docker?

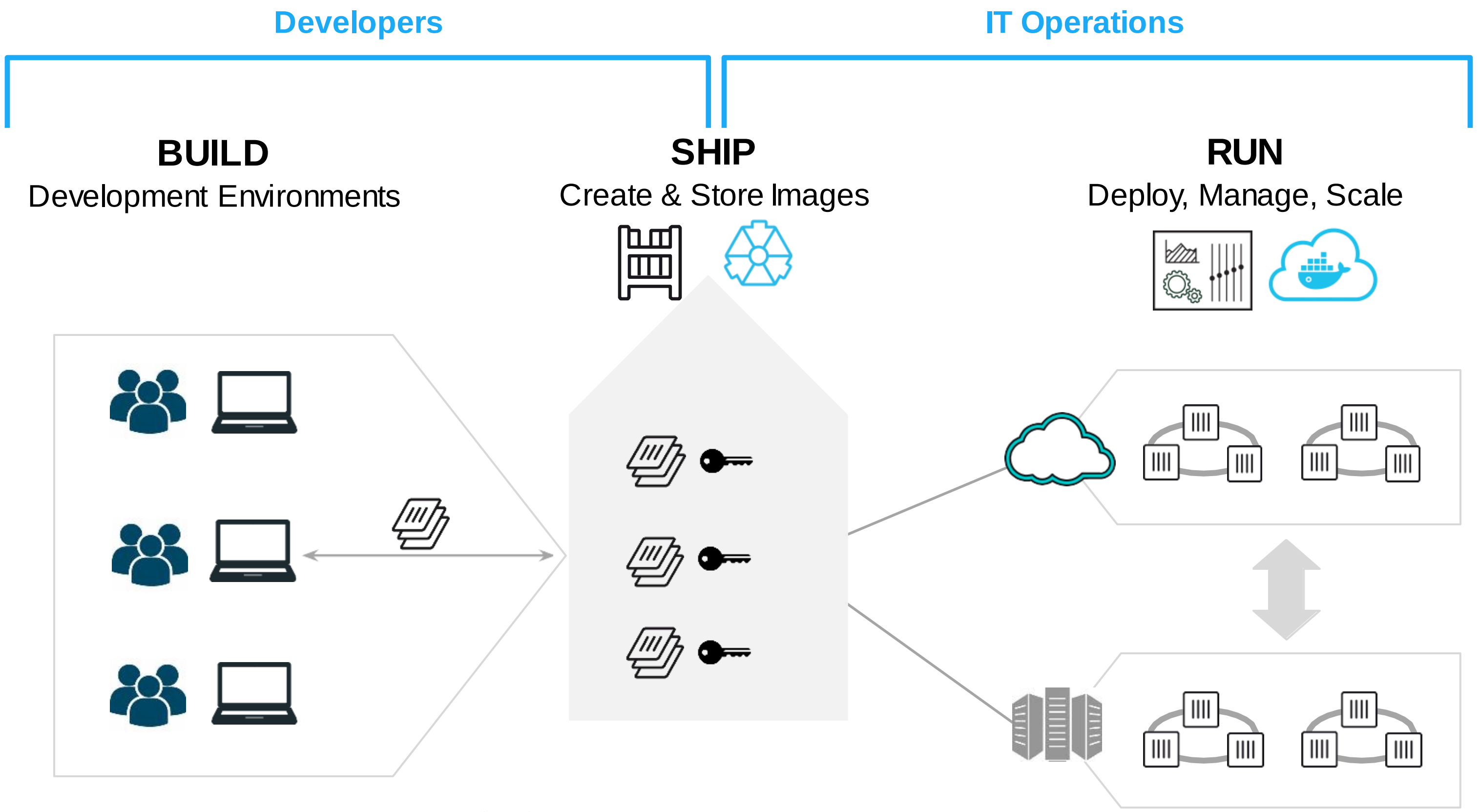


- **Lightweight, open, secure platform**
- **Simplify building, shipping, running apps**
- Runs natively on Linux or Windows Server
- Runs on Windows or Mac Development machines (with a virtual machine)
- Relies on "images" and "containers"

Docker architecture



Using Docker: Build, Ship, Run Workflow



Some Docker vocabulary

Docker Image

The basis of a Docker container. Represents a full application

Docker Container

The standard unit in which the application service resides and executes

Docker Engine

Creates, ships and runs Docker containers deployable on a physical or virtual, host locally, in a datacenter or cloud service provider

27

Registry Service (Docker Hub(Public) or Docker Trusted Registry(Private))

Cloud or server based storage and distribution service for your images

Basic Docker Commands

```
$ docker image pull node:latest

$ docker image ls
$ docker container run -d -p 5000:5000 --name node node:latest

$ docker container ps


$ docker stop <container id>

$ docker rm <container id>

$ docker image rmi (or <image id>)

$ docker build -t node:2.0 .

$ docker image push node:2.0

$ docker --help
```

Q&A