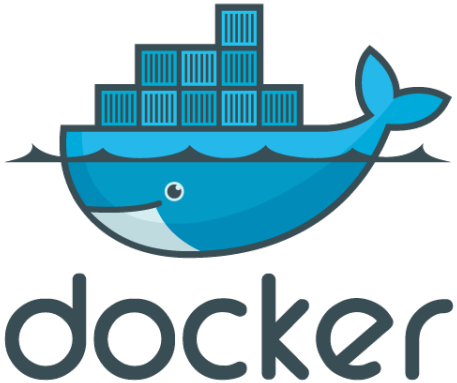


Docker Lisbon Meetup

Meetup 1 - Welcome to Docker - 6 February, 2014

8 months (November 2013) after launch

- >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

Why all the excitement?

The Challenge

Multiplicity of

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

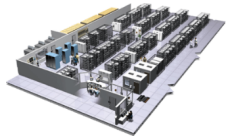
Customer Data Center

Public Cloud

Disaster recovery

Production Servers

Production Cluster



Contributor's laptop

Can I migrate smoothly and quickly?

docker

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	?	?	?	?	?	?	?

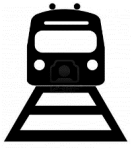
docker

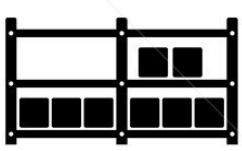
Cargo Transport Pre-1960

Multiplicity of

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

Multipilicity of methods for transporting/storin

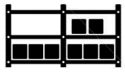




Can I transport quickly and smoothly (e.g. from boat to train to truck)

docker

Also a matrix from hell

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

Solution: Intermodal Shipping Container

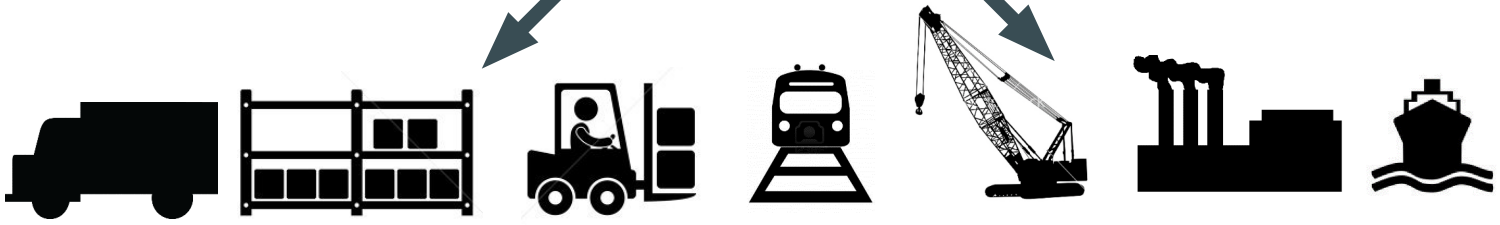
Multiplicity of Goods

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

A standard container that is loaded with virtually any goods, and stays sealed until it reaches final delivery.

…in between, can be loaded and unloaded, stacked, transported efficiently over long distances, and transferred from one mode of transport to another

Multiplicity of methods for transporting/storing



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

docker

Docker is a shipping container system for code

Multiplicity of Stacks

Static website

User DB

Web frontend

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 using standard operations and run consistently on virtually any hardware platform

Multiplicity of hardware environments

Development VM

QA server

Customer Data Center

Public Cloud

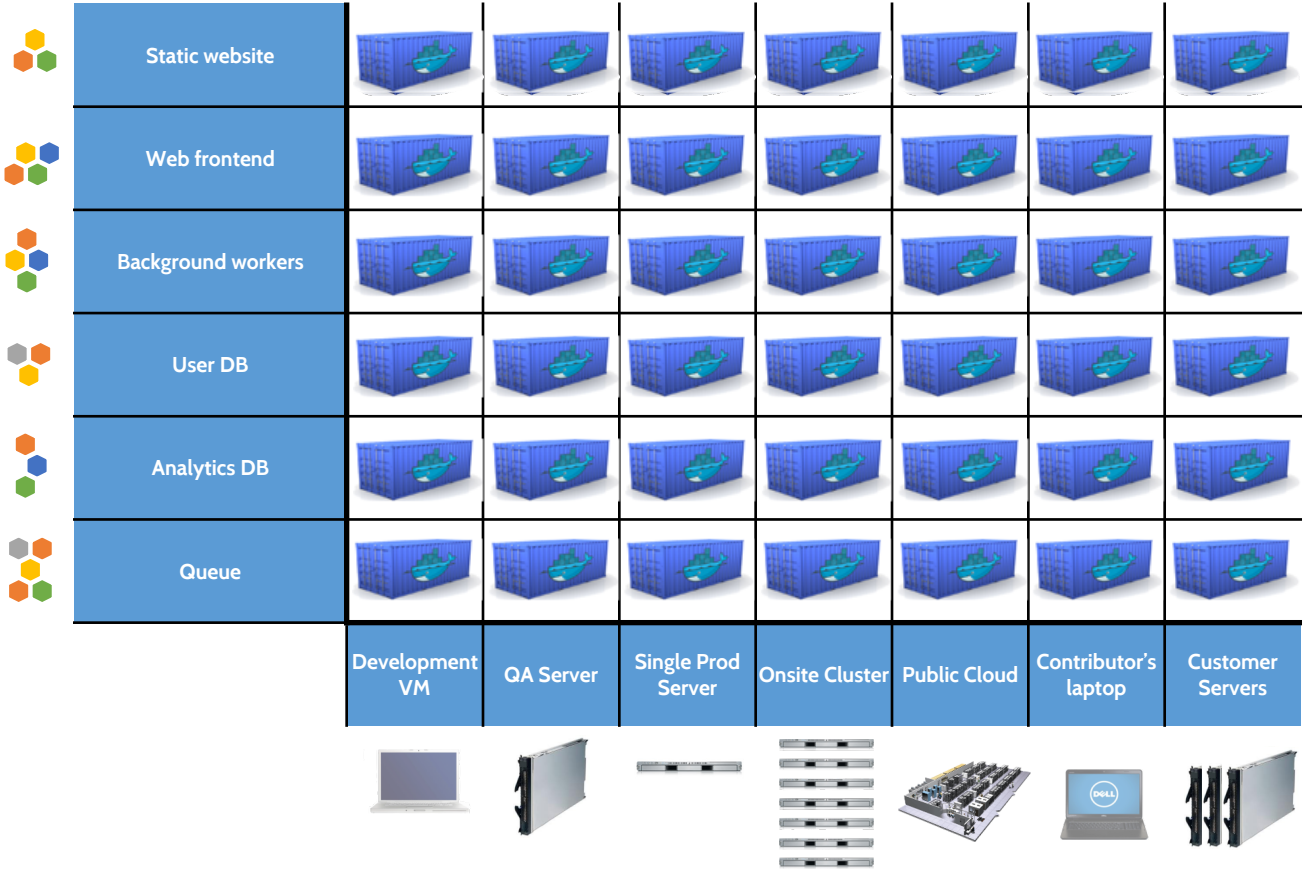
Production Cluster

Contributor's laptop

Can I migrate smoothly and quickly

docker

Docker eliminates the matrix from Hell



docker

Why Developers Care

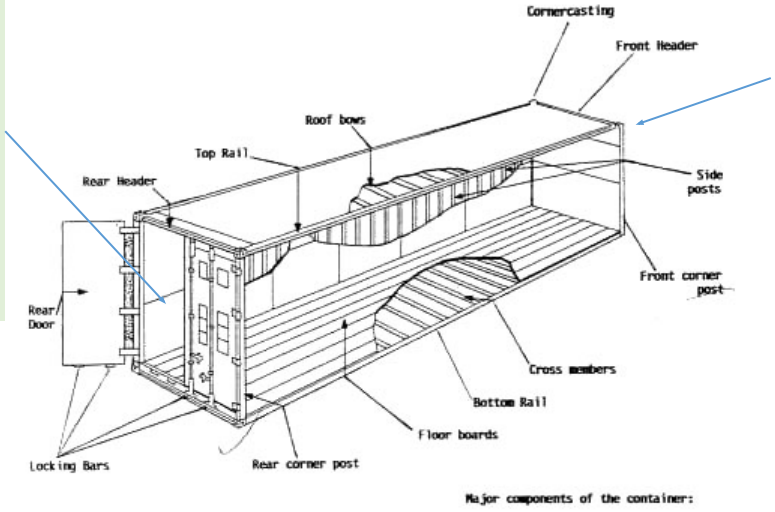
- Build once… run anywhere
  - A clean, safe, hygienic and portable runtime environment for your app.
  - No worries about missing dependencies, packages….
  - Run each app in its own isolated container, multiple parallel version.
  - Automate testing, integration, packaging… anything you can script
  - Reduce/eliminate concerns about platforms compatibility.
  - Cheap, zero-penalty containers to deploy services? A VM without the overhead of a VM? **Instant replay and reset** of image snapshots? That's the power of Docker

Why Devops Cares?

- Configure once…run anything
  - Make the entire lifecycle consistent, repeatable and efficient.
  - Increase the quality of code produced by developers.
  - Eliminate inconsistencies between development, test, production, and customer environments
  - Support segregation of duties
  - Significantly improves the speed and reliability of continuous deployment and continuous integration systems
  - Because the containers are so lightweight, address significant performance, costs, deployment, and portability issues normally associated with VMs

Why it works–separation of concerns

- Dan the Developer
  - Worries about what's "inside" the container
    - His code
    - His Libraries
    - His Package Manager
    - His Apps
    - His Data
  - All Linux servers look the same



- Oscar the Ops Guy
  - Worries about what's "outside" the container
    - Logging
    - Remote access
    - Monitoring
    - Network config
  - All containers start, stop, copy, attach, migrate, etc. the same way

docker

More technical explanation

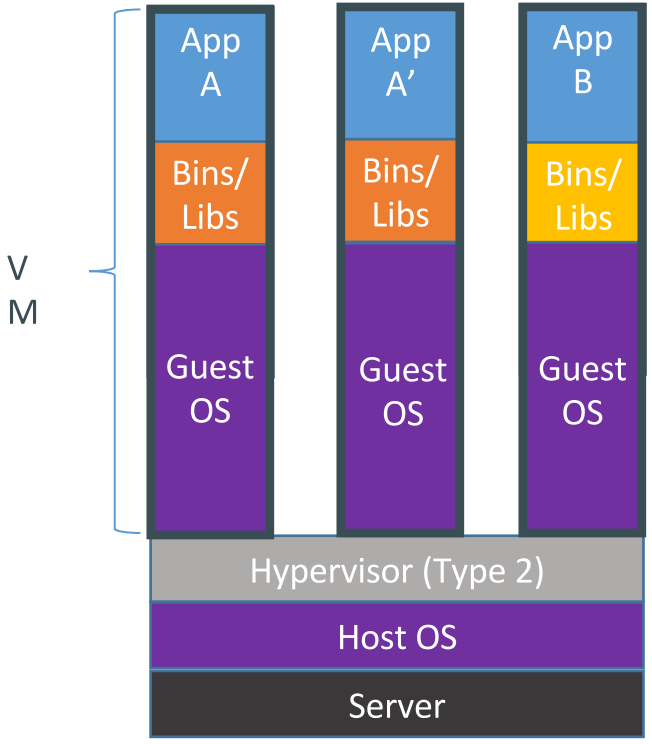
WHY

- Run everywhere
  - Kernel version 2.6.32+
  - Regardless of host distro
  - Physical or virtual, cloud or not
  - Container and host architecture must match
- Run anything
  - If it can run on the host, it can run in the container

WHA

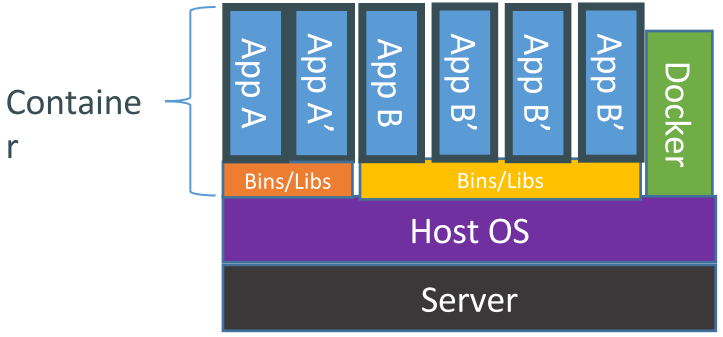
- High Level–It's a lightweight VM
  - Own process space
  - Own network interface
  - Can run stuff as root
  - Can have its own /sbin/init (different from host)
- Low Level–It's chroot on steroids
  - Can also *not* have its own /sbin/init
  - Container=isolated processes
  - Share kernel with host
  - No device emulation (neither HVM nor PV) from host)

Containers vs. VMs



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

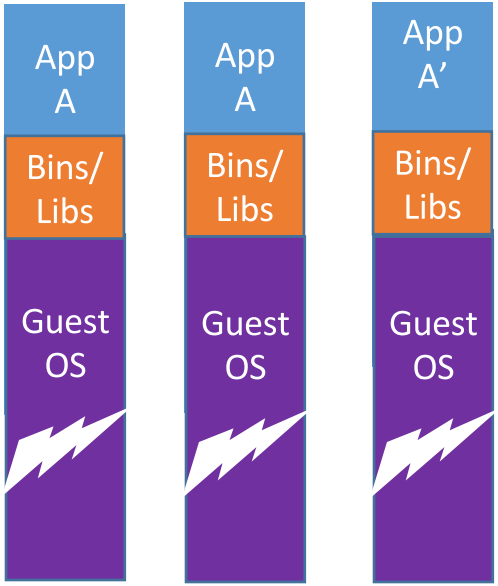
…result is significantly faster deployment, much less overhead, easier migration, faster restart



docker

Why are Docker containers lightweight?

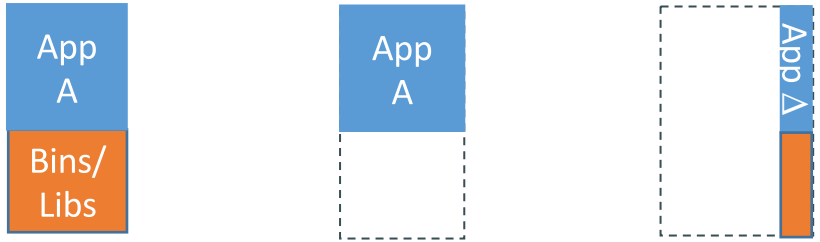
VMs



VMs

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

Containers



Original App

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

Copy of App

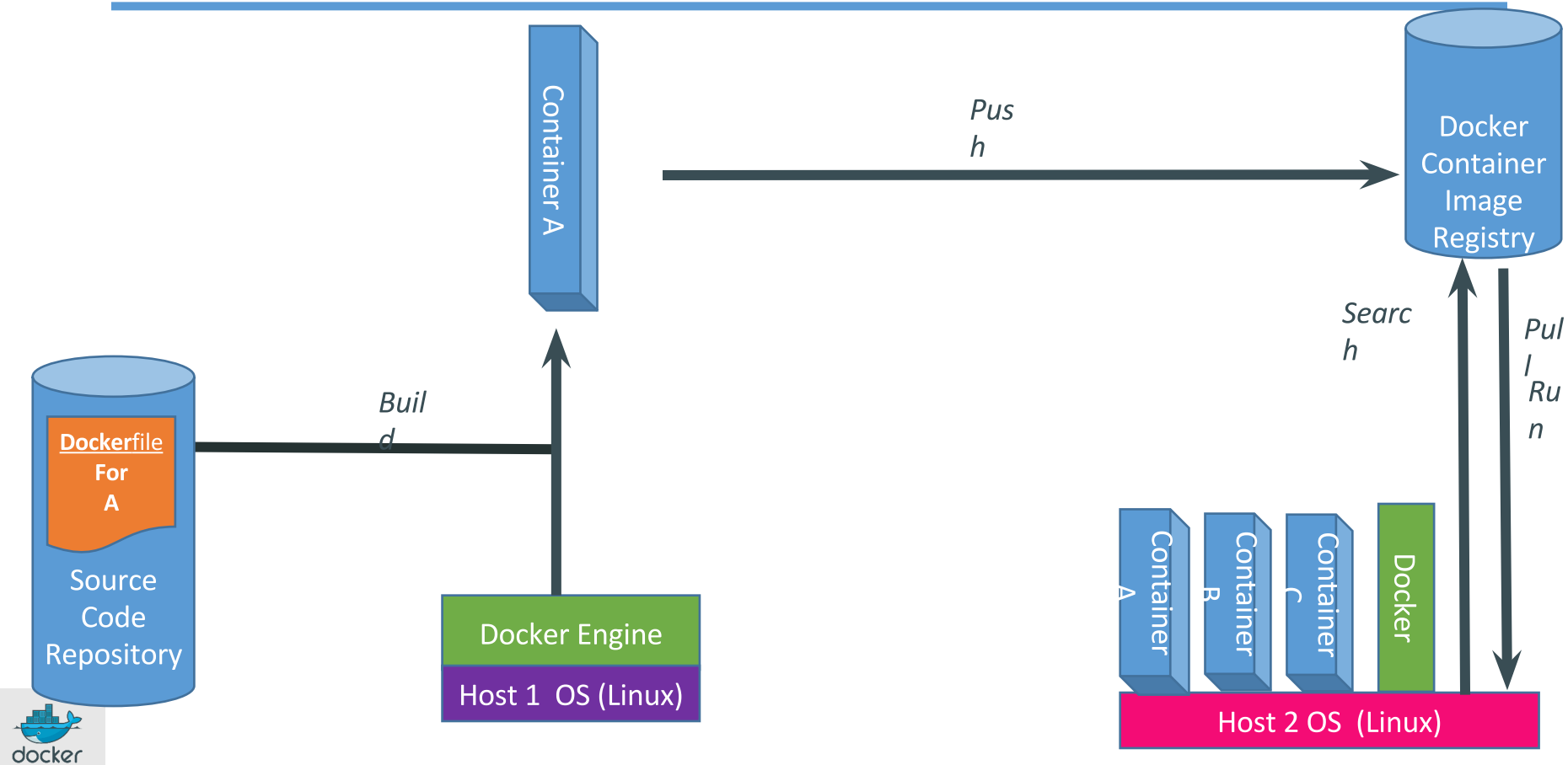
No OS. Can Share bins/libs

Modified App

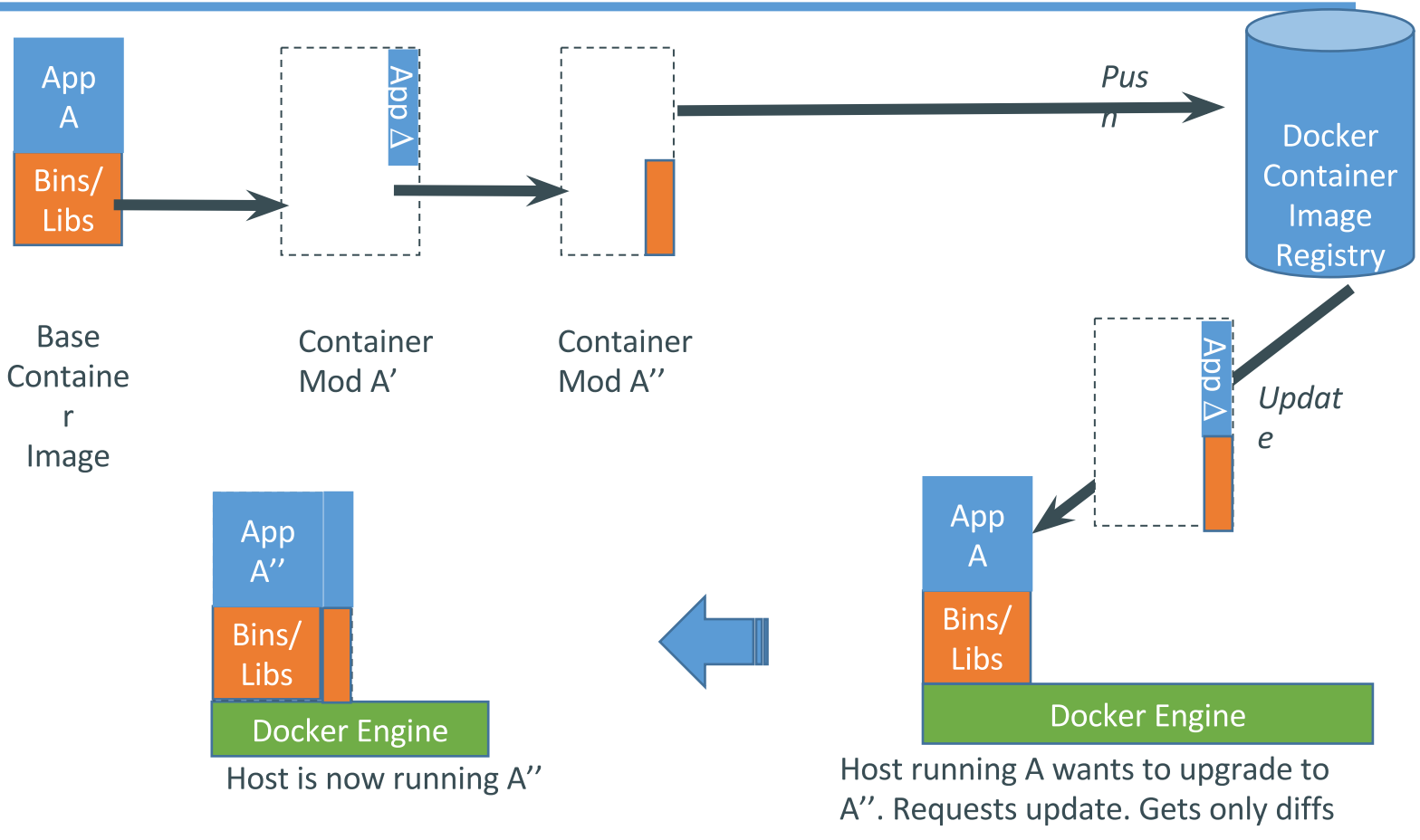
Copy on write capabilities allow us to only save the diffs Between container A and container A'

docker

What are the basics of the Docker system?



Changes and Updates



Integration

- http://docs.docker.io/en/latest/api/docker_remote_api/
  - Using hostname:port pair.
    - 127.0.0.1:4243
  - Using unix socket. (default since Remote API v1.3)
    - unix:///var/run/docker.sock

Ecosystem Support

- Operating systems
  - Virtually any distribution with a 2.6.32+ kernel
  - Red Hat/Docker collaboration to make work across RHEL 6.4+, Fedora, and other members of the family (2.6.32 +)
  - CoreOS–Small core OS purpose built with Docker
- OpenStack
  - Docker integration into NOVA (& compatibility with Glance, Horizon, etc.) accepted for Havana release
- Private PaaS
  - OpenShift
  - Solum (Rackspace, OpenStack)
- Public PaaS
  - Deis, Voxoz, Cocaine (Yandex), Baidu PaaS
- Public IaaS
  - Native support in Rackspace, Digital Ocean,+++
  - AMI (or equivalent) available for AWS & other
- DevOps Tools
  - Integrations with Chef, Puppet, Jenkins, Travis, Salt, Ansible +++
- Orchestration tools
  - Mesos, Heat, ++
  - Shipyard & others purpose built for Docker

Want to learn more?

- www.docker.io
- Github: dotcloud/docker
- IRC: freenode/#docker
- Google groups: groups.google.com/forum/#!forum/docker-user
- Twitter: follow @docker

Hands-on

Setup

- Vagrant
  - git clone github.com/dotcloud/docker
  - vagrant up
- boot2docker
  - curl https://raw.github.com/steeve/boot2docker/master/boot2docker > boot2docker
  - curl -o docker http://get.docker.io/builds/Darwin/x86_64/docker-latest
  - export DOCKER_HOST=tcp://
  - boot2docker init
  - boot2docker up

Docker command line

- http://docs.docker.io/en/latest/commandline/
  - Containers
    - run, start, stop, attach, kill, rm, diff, logs, ...
  - Images
    - commit, pull, push, rmi, search, ...
  - Misc
    - version, events, ...

Famous run command

Usage: docker run [OPTIONS] IMAGE[:TAG] [COMMAND] [ARG...]

-a=map[]: Attach to stdin, stdout or stderr.

-c=0: CPU shares (relative weight)

-cidfile="": Write the container ID to the file

-d=false: Detached mode: Run container in the background, print new container id

-e=[]: Set environment variables

-h="": Container host name

-i=false: Keep stdin open even if not attached

-privileged=false: Give extended privileges to this container

-m=0: Memory limit (in bytes)

-n=true: Enable networking for this container

-p=[]: Map a network port to the container

-t=false: Allocate a pseudo-tty

-u="": Username or UID

-dns=[]: Set custom dns servers for the container

-v=[]: Create a bind mount with: [host-dir]:[container-dir]:[rw|ro]. If "host-dir" is missing, then docker creates a new volume.

-volumes-from="": Mount all volumes from the given container.

-entrypoint="": Overwrite the default entrypoint set by the image.

-w="": Working directory inside the container

-lxc-conf=[]: Add custom lxc options -lxc-conf="lxc.cgroup.cpuset.cpus = 0,1"

Container sharing

- docker export 1aba8bfac7d5 > imgexport.tgz
- cat imgexport.tgz | sudo docker import - myimg
- sudo docker run myimg uptime

Dockerfile

Dockerfile contents:

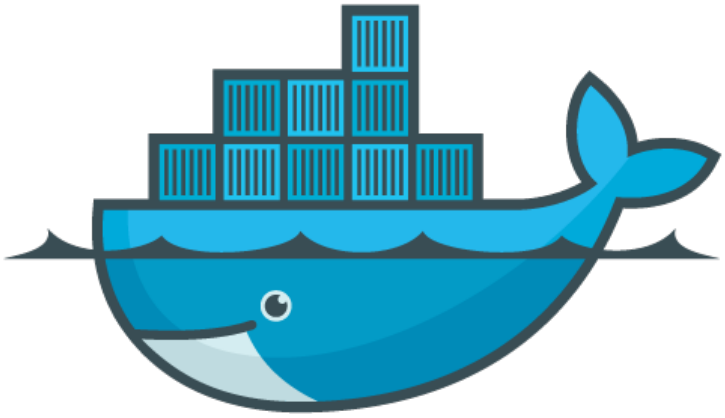
```
# Build the image of ubuntu 12.04 LTS
from ubuntu:precise

# Install node.js
run wget -O - http://nodejs.org/dist/v0.10.19/node-v0.10.19-linux-x64.tar.gz | tar -C
/usr/local/ --strip-components=1 -zxv

# Expose port 80 to the host machine
expose 80
```

Build it:

```
docker build -t our-app .
```



docker

www.docker.io