CloudNativeCon KubeCon

North America 2017

From screen to pods:

Bootstrapping a cloud agnostic system using Kubernetes

Patrick McQuighan
Dec 8, 2017

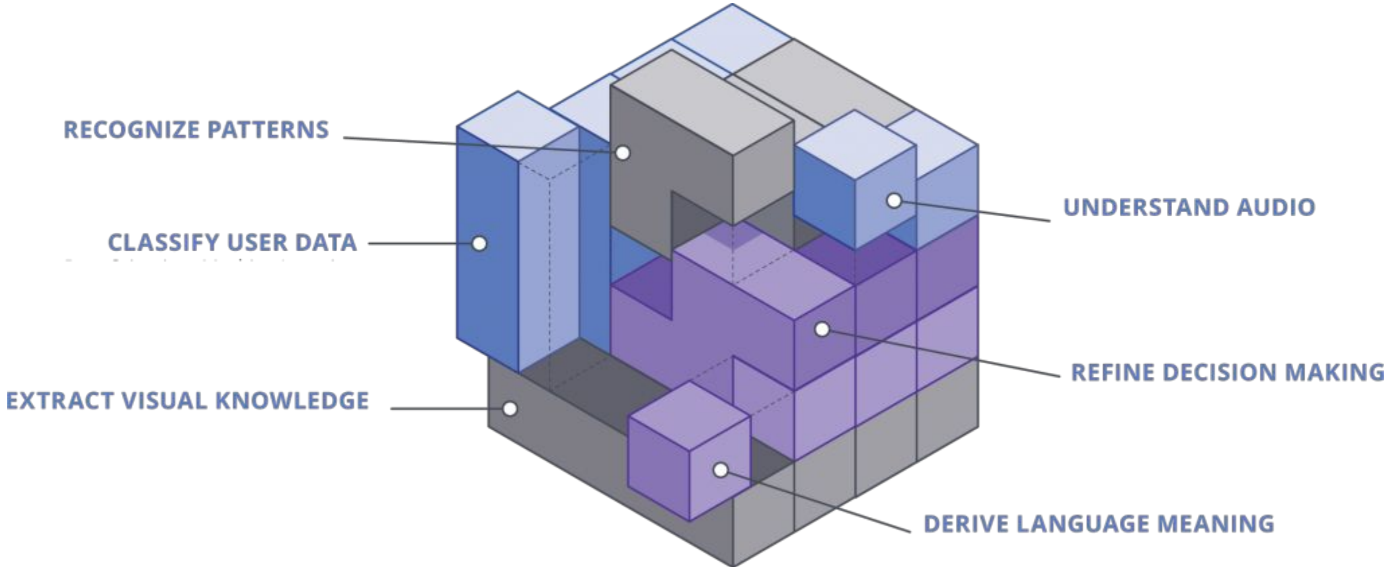
Who am I?

Patrick McQuighan

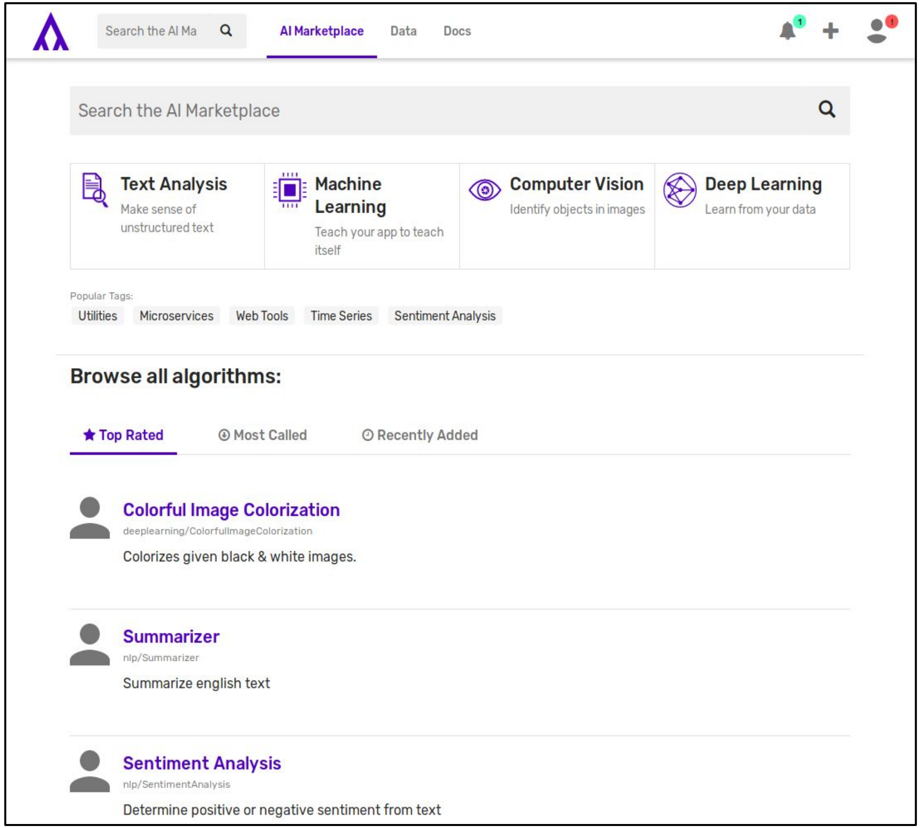
- ➔ Sr Platform Engineer at Algorithmia
- ➔ Mixed hardware - CPU, GPU
- ➔ 10,000s containers daily
- ➔ Migration to Kubernetes late 2016

What do we do?

A common API for algorithms, functions, and ML models that run as scalable microservices.



What do we do?



Search the AI Marketplace

Text Analysis Make sense of unstructured text

Machine Learning Teach your app to teach itself

Computer Vision Identify objects in images

Deep Learning Learn from your data

Popular Tags:
Utilities · Microservices · Web Tools · Time Series · Sentiment Analysis

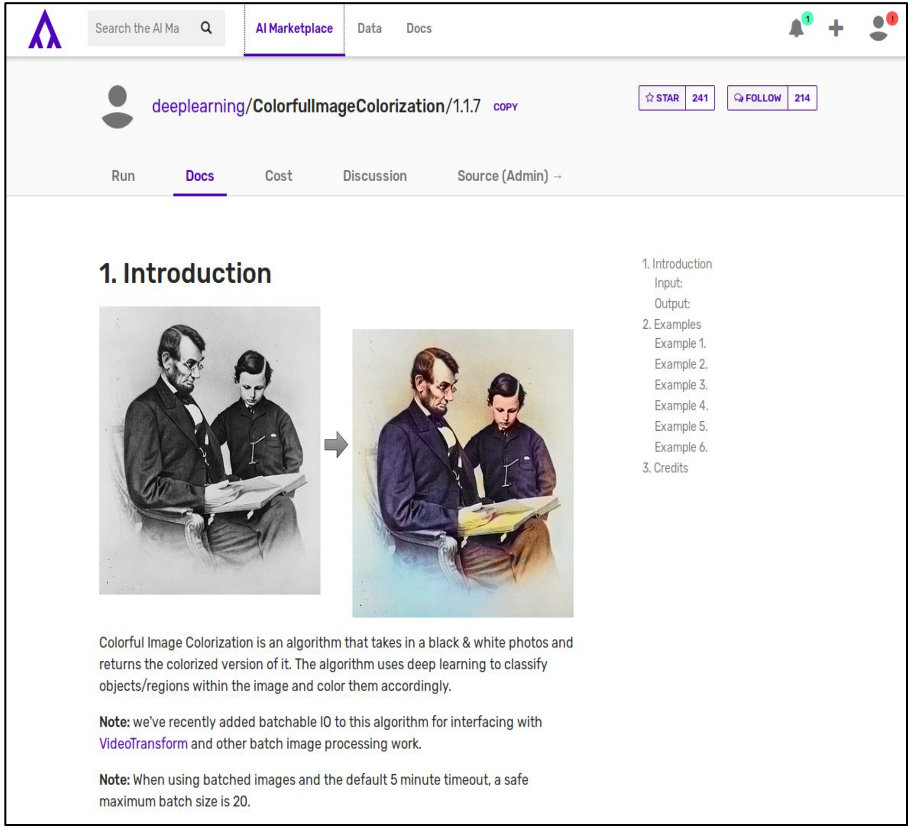
Browse all algorithms:

★ Top Rated · Most Called · Recently Added

Colorful Image Colorization
deeplearning/ColorfullmageColorization
Colorizes given black & white images.

Summarizer
nlp/Summarizer
Summarize english text

Sentiment Analysis
nlp/SentimentAnalysis
Determine positive or negative sentiment from text



deeplearning/ColorfullmageColorization/1.1.7

1. Introduction
Input:
Output:
2. Examples
Example 1.
Example 2.
Example 3.
Example 4.
Example 5.
Example 6.
3. Credits

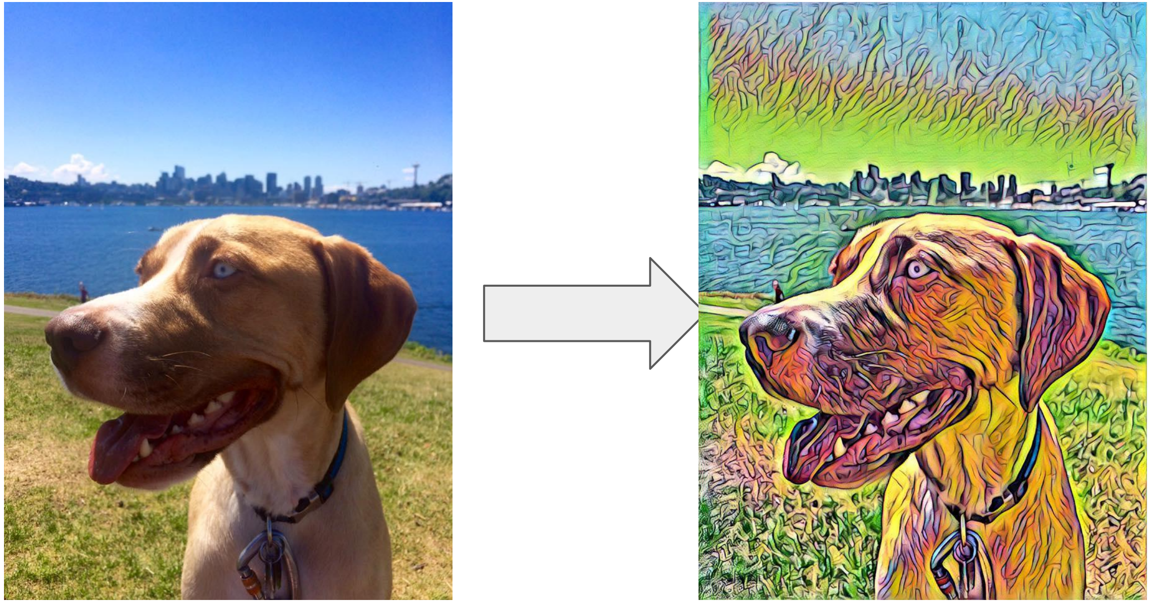
1. Introduction

Colorful Image Colorization is an algorithm that takes in a black & white photos and returns the colorized version of it. The algorithm uses deep learning to classify objects/regions within the image and color them accordingly.

Note: we've recently added batchable IO to this algorithm for interfacing with VideoTransform and other batch image processing work.

Note: When using batched images and the default 5 minute timeout, a safe maximum batch size is 20.

What do we do?

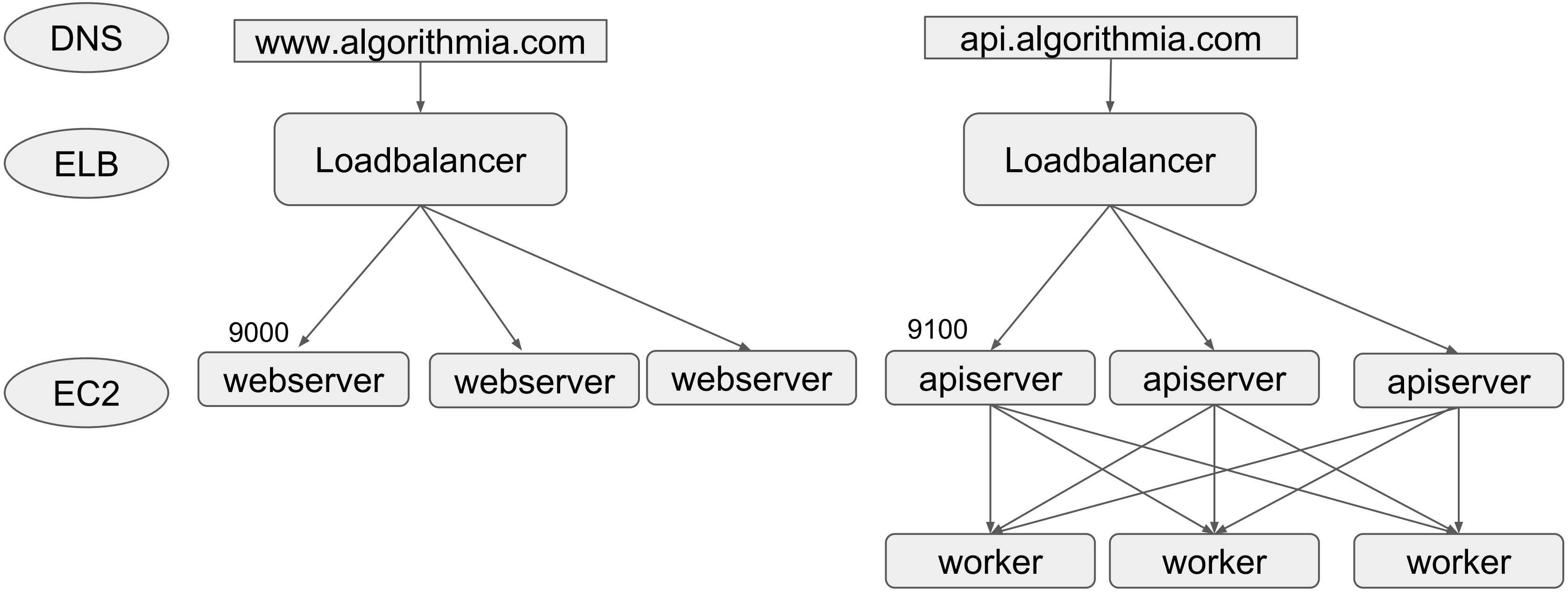


```
$ algo run deeplearning/DeepFilter/0.6.0 -d '{
 "images":["data://pmcq/pictures/loowit-normal.jpg"],
 "savePaths": ["data://pmcq/pictures/loowit-filtered.jpg"],
 "filterName": "sunday"
}'
```

Outline

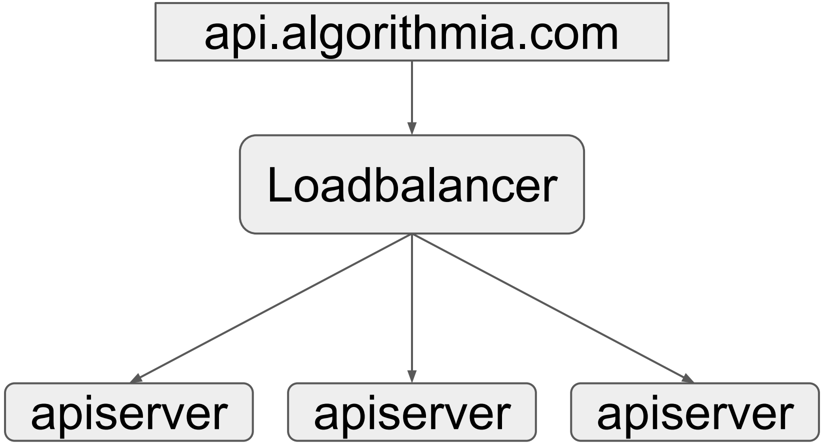
1. **Traditional web application architecture-2015**
2. Why docker? - Aug 2016
3. Why kubernetes? - Jan 2017
4. Migration problems
5. Generalizing for multiple clouds
6. What next?

In the beginning…

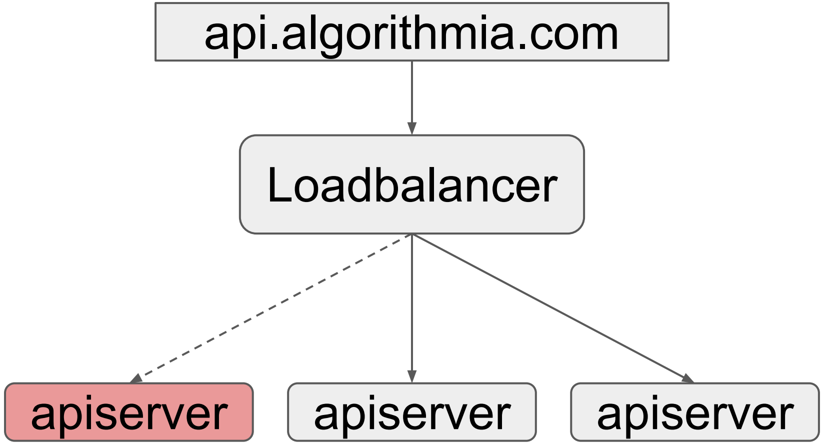


Public Beta - 2015

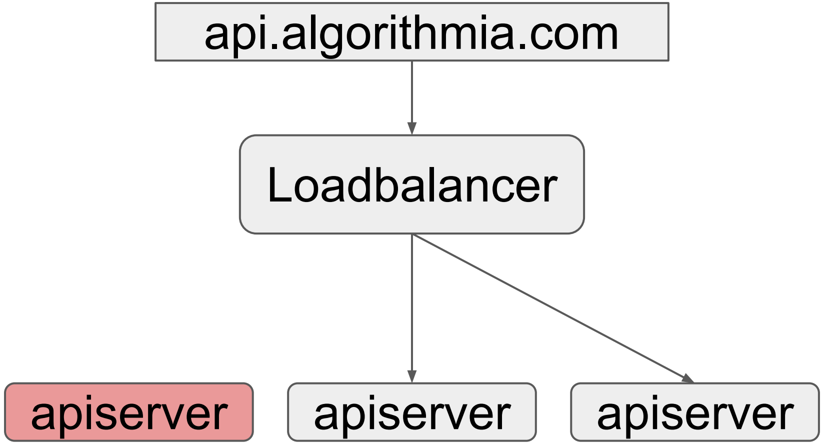
In the beginning…



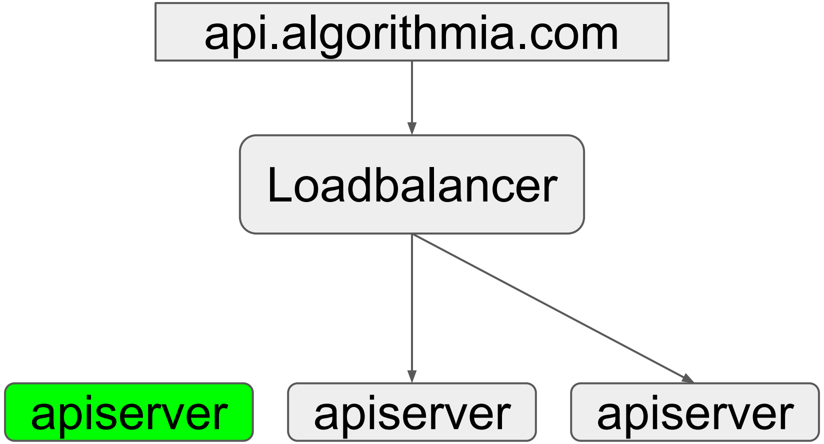
In the beginning…



In the beginning...



In the beginning...


api.algorithmia.com

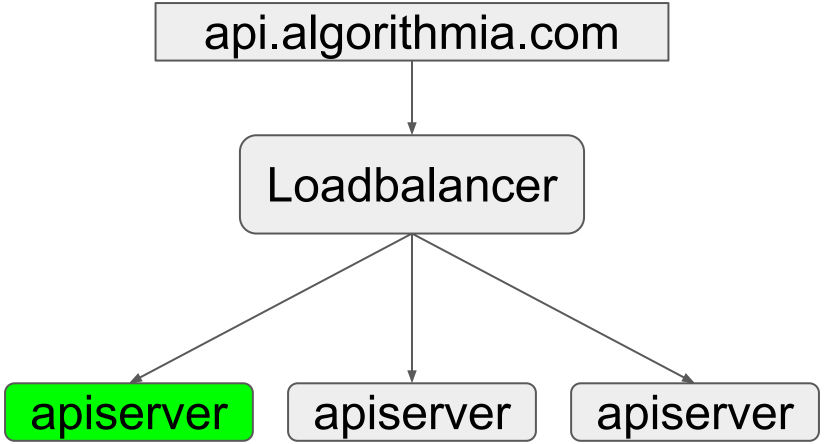
Loadbalancer

apiserver

apiserver

apiserver

In the beginning...



api.algorithmia.com

Loadbalancer

apiserver

apiserver

apiserver

Limitations

- Can only run 1 copy of service per machine
- Hard to add services
  - Configure hosts
  - Create deployment scripts
  - How is it monitored?
  - What happens if process crashes?
  - How is it exposed to the world?

Motivations

Public marketplace is not always a good fit for customers

- Data privacy requirements
- Bandwidth costs
- Latency requirements
- Internal or private package dependencies

Barriers for Enterprise Deployments

1. Delivering applications & updates
2. Operating System - Ubuntu, CentOS
3. 3rd party software
4. Cost for proof-of-concept
5. Many moving pieces - host config, VPC, firewalls, …
6. Cloud provider - AWS, Azure, Openstack, GCP

Outline

1. Traditional web application architecture
2. **Why docker?**
3. Why kubernetes?
4. Migration problems
5. Generalizing for multiple clouds
6. What next?

Barriers for Enterprise Deployments

➡ 1. Delivering applications & updates

➡ 2. Operating System - Ubuntu, CentOS

3. 3rd party software
4. Cost for proof-of-concept
5. Many moving pieces - host config, VPC, firewalls, …
6. Cloud provider - AWS, Azure, Openstack, GCP

Benefits of Containerizing

- No longer tied to host’s libraries or operating system*
- Distribution and upgrade of application images
- Unified way to view logs and kill applications
- Multiple copies on the same machine

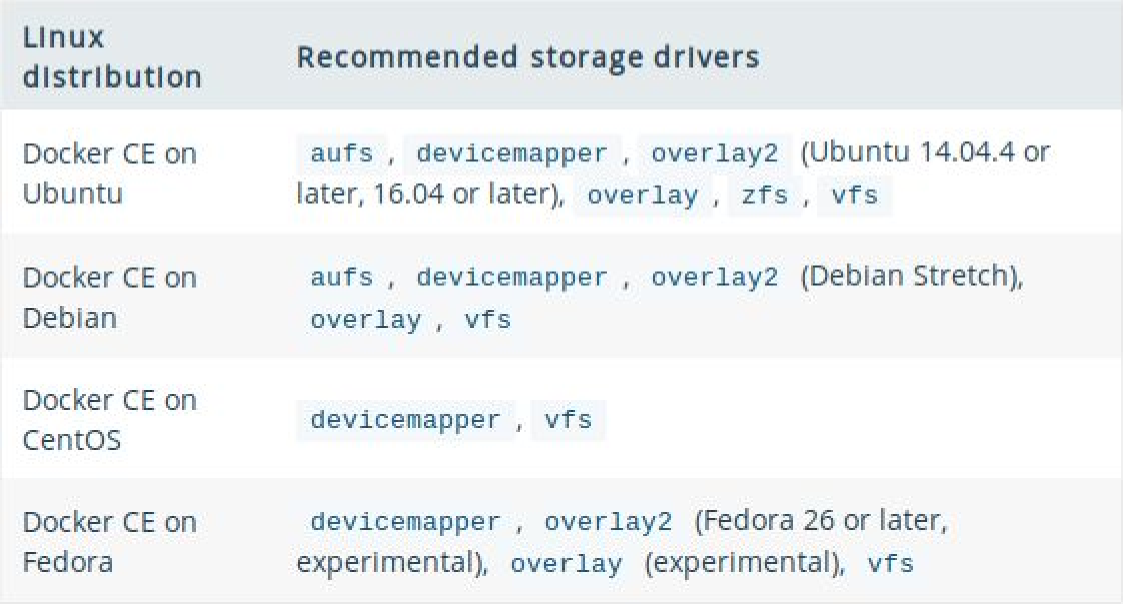
Our state circa August 2016

Caveats

- Host OS differences with permissions such as SELinux or default iptables
- GPUs and other devices and drivers
- Docker group ID and /var/run/docker.sock
  - moby/moby#21184

Issues

Choice of storage driver



Linux distribution	Recommended storage drivers
Docker CE on Ubuntu	`aufs` , `devicemapper` , `overlay2` (Ubuntu 14.04.4 or later, 16.04 or later), `overlay` , `zfs` , `vfs`
Docker CE on Debian	`aufs` , `devicemapper` , `overlay2` (Debian Stretch), `overlay` , `vfs`
Docker CE on CentOS	`devicemapper` , `vfs`
Docker CE on Fedora	`devicemapper` , `overlay2` (Fedora 26 or later, experimental), `overlay` (experimental), `vfs`

Issues

Choice of storage driver

- AUFS requires linux-extras package
- We had process auto-updating kernel versions but not extras package
- Reboots resulted in docker daemon crash-loop

Issues

EBS volume performance is terrible until warmed-up

- Process can take several hours
- Write-heavy workloads (such as spawning many containers) would lock up the volume
- Need empty EBS volume - can't pre-fetch images
- moby/moby#26452
- openshift/origin#7243

Outline

1. Traditional web application architecture
2. Why docker?
3. **Why kubernetes?**
4. Migration problems
5. Generalizing for multiple clouds
6. What next?

Barriers for Enterprise Deployments

1. ~~Delivering applications & updates~~
2. ~~Operating System - Ubuntu, CentOS~~
3. 3rd party software
4. Cost for proof-of-concept
5. Many moving pieces - host config, VPC, firewalls, …
6. Cloud provider - AWS, Azure, Openstack, GCP

Barriers for Enterprise Deployments

1. ~~Delivering applications & updates~~
2. ~~Operating System - Ubuntu, CentOS~~
3. ➡ 3rd party software
4. ➡ Cost for proof-of-concept
5. Many moving pieces - host config, VPC, firewalls, …
6. Cloud provider - AWS, Azure, Openstack, GCP

Filling in the Gaps

- **Services** - loadbalancing, service discovery
- **Deployments/Replica sets** - N copies running, rolling updates
- **CNI plugin** - routing between containers, network policies
- **Labels/annotations** - added resiliency
- **Daemonsets** - monitoring tools or log aggregation
- **ConfigMaps/Secrets** - distributing application configuration
- **Jobs** - cluster tasks and initialization

What don’t we use Kubernetes for?

- Precise scheduling of algorithm containers
  - **Optimizations** - image fetched, related data files cached, machine utilization, pending requests, …
  - **Security** - containers are per-user for data security, proxied requests
  - **Performance** - <10ms to start container (if needed)
  - Possible to change, but haven’t seen a need

What don’t we use Kubernetes for?

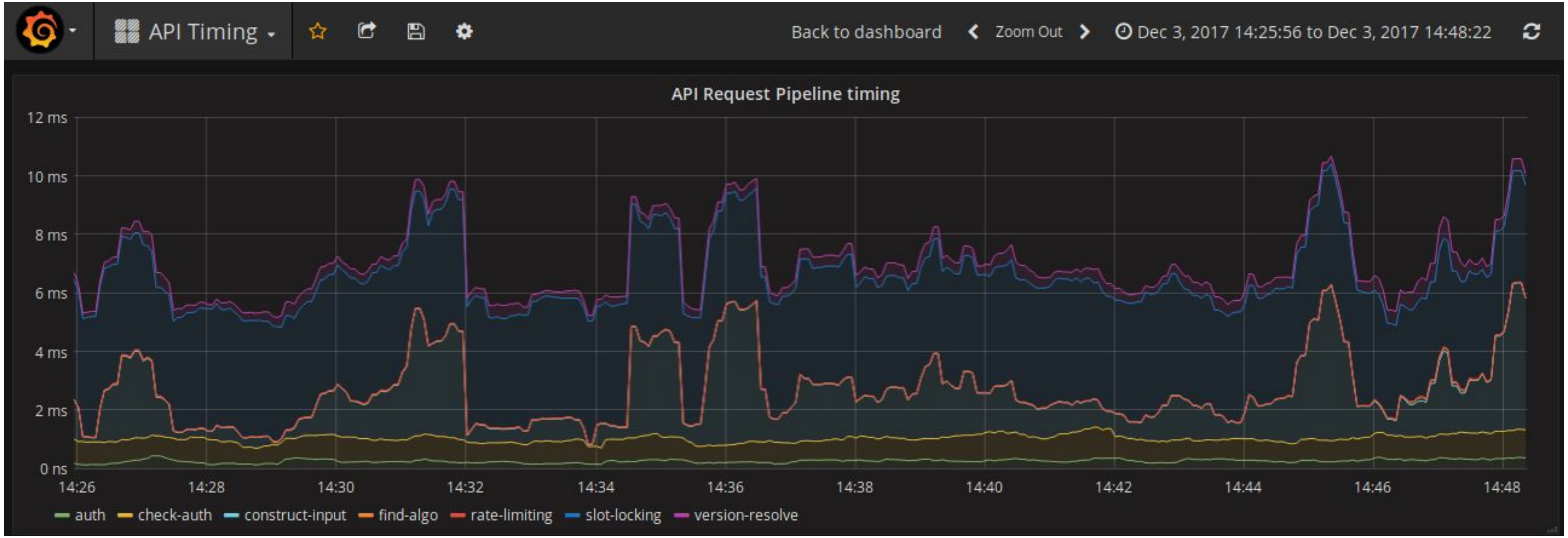
- GPU management
    - Deep monitoring we track for sharing purposes
- Ingress controllers
    - Lots of customization for ourselves and customers
    - Trusted CAs, numbers of certificates, X.509 validation
    - Likely to change

End Results

- Migration completed January 2017
- Minimum number of servers dropped from 12 -> 6
- Down to a single loadbalancer
- Replaced 3rd party search & deployment
- Easily added services to the stack

Prometheus & Grafana

Easily added in about 1 day thanks to Kubernetes!



Outline

1. Traditional web application architecture
2. Why docker?
3. Why kubernetes?
4. **Migration problems**
5. Generalizing for multiple clouds
6. What next?

Issues - Project Management

Projects are moving *fast*

- Not always compatible with each other
  kubernetes/kubernetes#40182
- Lots of versions in development and use at the same time

What do you target if migration is a multi-month project?

Issues - Project Management

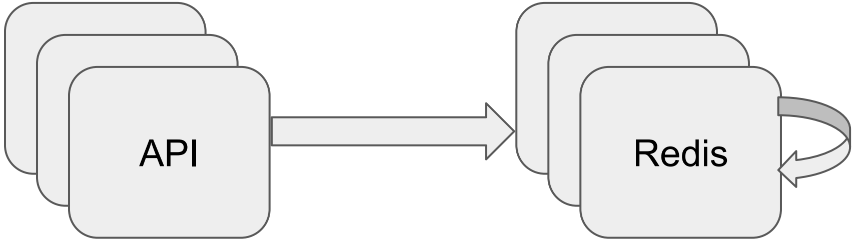
Lots of new concepts - small errors break everything

- Readiness checks
- preStop hooks
- Grace periods
- rollingUpdate strategy
- initContainers
- …

Issues - Technical

IP address routability

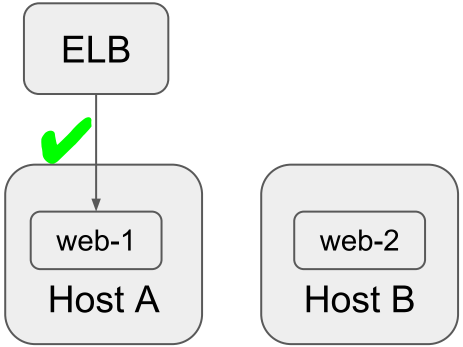
- Container IP addresses only accessible within the kube cluster
- Can make awkward migration if your services have dependencies
- Need to have clear picture of network topology!



Issues - Technical

Loadbalancer & connection pooling

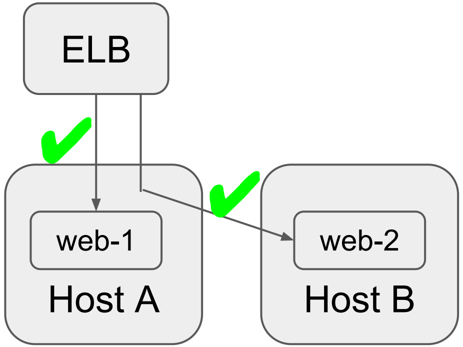
- ELB believes it is talking to machine A on port 9000



Issues - Technical

Loadbalancer & connection pooling

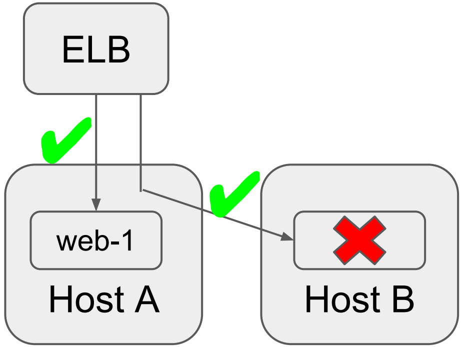
- ELB believes it is talking to machine A on port 9000
- Kube-proxy forwards this traffic to a different host



Issues - Technical

Loadbalancer & connection pooling

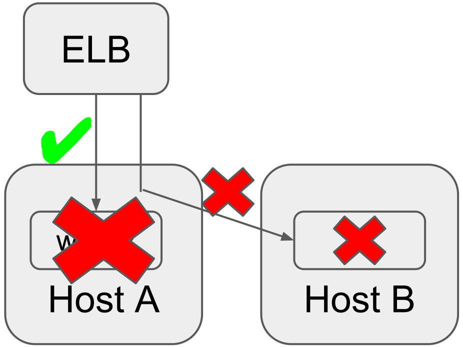
- ELB believes it is talking to machine A on port 9000
- Kube-proxy forwards this traffic to a different host
- Deployment updates a pod



Issues - Technical

Loadbalancer & connection pooling

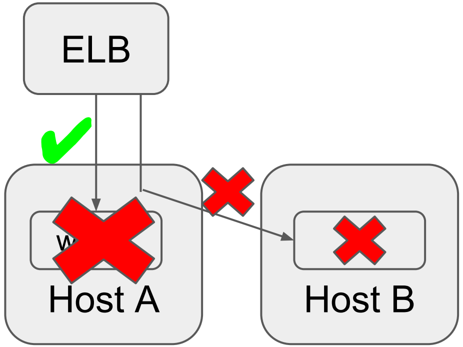
- ELB believes it is talking to machine A on port 9000
- Kube-proxy forwards this traffic to a different host
- Deployment updates a pod
- ELB mistakenly thinks machines are down - outages!



Issues - Technical

Loadbalancer & connection pooling

- ELB believes it is talking to machine A on port 9000
- Kube-proxy forwards this traffic to a different host
- Deployment updates a pod
- ELB mistakenly thinks machines are down - outages!



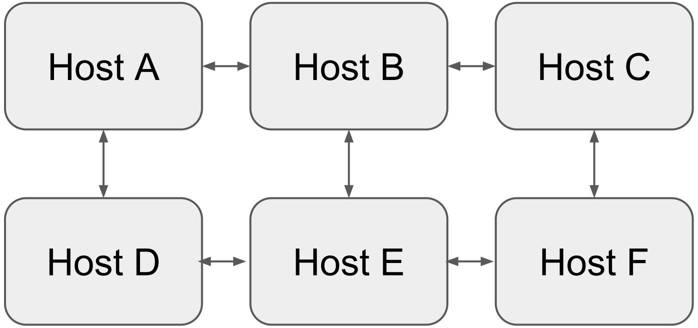
Be cautious with connection pooling:

- Connection: close during shutdown
- TCP backend checks

Issues - Technical

Suspending hosts & network connectivity

- We use weave-net plugin
- We frequently downscale our cluster as needed
- Can result in disconnected nodes if not careful with settings



Host A

Host B

Host C

Host D

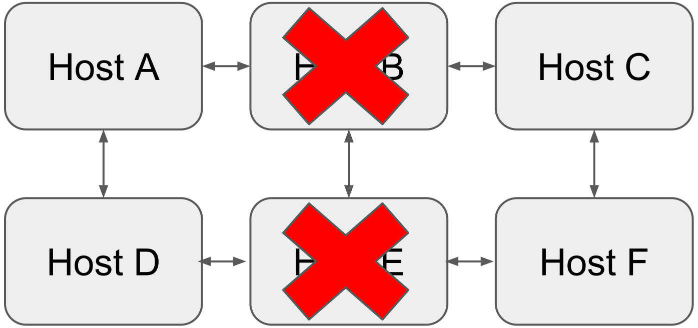
Host E

Host F

Issues - Technical

Suspending hosts & network connectivity

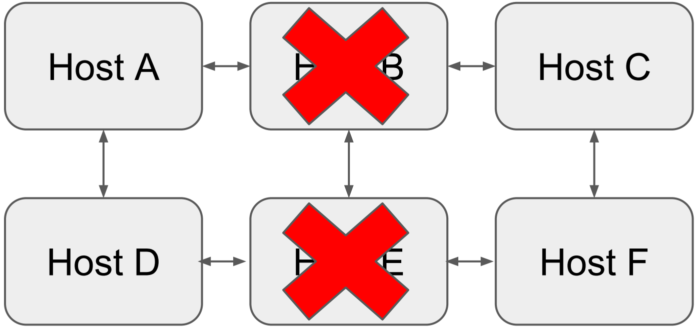
- We use weave-net plugin
- We frequently downscale our cluster as needed
- Can result in disconnected nodes if not careful with settings



Issues - Technical

Suspending hosts & network connectivity

- We use weave-net plugin
- We frequently downscale our cluster as needed
- Can result in disconnected nodes if not careful with settings



- ➔ N+1 connections needed if N nodes go down
- ➔ Monitor connection counts, % failed
- ➔ Consider topology during scaling

Outline

1. Traditional web application architecture
2. Why docker?
3. Why kubernetes?
4. Migration problems
5. **Generalizing for multiple clouds**
6. What next?

Barriers for Enterprise Deployments

1. ~~Delivering applications & updates~~
2. ~~Operating System - Ubuntu, CentOS~~
3. ~~3rd party software~~
4. ~~Cost for proof-of-concept~~

➡ 5. Many moving pieces - host config, VPC, firewalls,

➡ 6. Cloud provider - AWS, Azure, Openstack, GCP

Considerations

- **Vendor lock-in** - avoiding services like DynamoDB, Kinesis, ECS, …
- **Use Kubernetes** - services vs loadbalancers
- **Specific/Generic?** - scripts and host configurations per provider?
- **How to provision?**

Provisioning

Just map the services from AWS to Azure, right?

Differences in Primitives

- **DNS** - external vs internal
- **Network cards** - number and ordering
  - kubernetes/kubeadm#102
- **Devices** - where and how are they mounted
- **API Versions** -
  - Rackspace isn't Openstack
  - Openstack isn't Openstack

Differences in Primitives

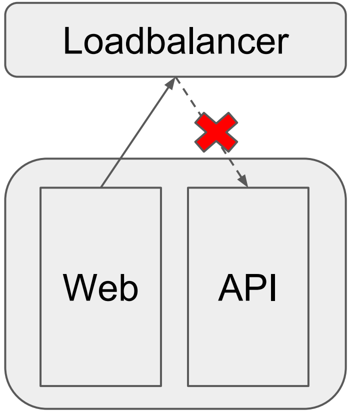
Loadbalancers

- SSL - termination, certificate generation
- Associations
- Routing - running multiple applications per machine

Differences in Primitives

Loadbalancers

- SSL - termination, certificate generation
- Associations
- Routing - running multiple applications per machine



Barriers for Enterprise Deployments

1. ~~Delivering applications & updates~~
2. ~~Operating System - Ubuntu, CentOS~~
3. ~~3rd party software~~
4. ~~Cost for proof-of-concept~~
5. ~~Many moving pieces - host config, VPC, firewalls, …~~
6. ~~Cloud provider - AWS, Azure, Openstack, GCP~~

We can provision a VPC and system in < 1 hour!

Outline

1. Traditional web application architecture
2. Why docker?
3. Why kubernetes?
4. Migration problems
5. Generalizing for multiple clouds
6. **What next?**

The Future

- **Performance improvements**
  - GPU and other device management enhancements
  - Image compression - moby/moby#1266
- **Multi-cloud**
  - Spanning datacenters can increase cost and latency
  - Security concerns with data traveling over the internet
  - Federated “Ubernetes”

Questions?

Contact: patrick@algorithmia.com

LinkedIn: https://www.linkedin.com/in/pxmcq

We're Hiring! https://algorithmia.com/jobs/

Try us out, use **KubeCon2017** for $50 in free credits