Real World Kubernetes Deployments

failure domains, upgrades, high-availability

@coreoslinux
@brandonphilips
Stickers Upfront
Decorate your laptop, dog, kid, phone.

Follow Along Instructions

http://bit.ly/1XeUbMW
Brandon Philips
CTO, CoreOS
github.com/philips
Build, Store and Distribute your Containers

quay.io
MISSION

Secure the Internet
STRATEGY

Separate Apps from OS
STRATEGY

Make Servers Consistent
STRATEGY

Tolerate Machine Failures
STRATEGY

Make Servers Easy to Upgrade
Simplify Application Upgrades
Join us as we dance madly on the lip of the volcano.
Application Packaging
Abstract away app from the OS
Traditional Distro

- kernel
- systemd
- sshd
- docker
- rkt

- Python
- Java
- nginx
- MySQL
- OpenSSL

- App1
- App2
- App3
CoreOS

Your Containers

kernel
systemd
sshd
docker
rkt

Python
App1

Java
App2

nginx
App3

Required software only
2 Linux at Scale
Patches to the OS and kernel are hard

APPLICATION
- Dependency breakage
- Uptime risk

SECURITY
- Retest after updates
- No automation
CoreOS

Your Containers

kernel
systemd
sshd
docker
rkt

Python

Java

nginx

App1

App2

App3

Required software only
Auto-updating browsers fixed security
We got HTML5 at the same time
Operations Paradise

+ Easy scale out

rerotate Painless app upgrades

✔ Tolerant of machine failure
App Req/sec: 6,000
App Healthy: True
App Req/sec: 6,000
App Healthy: True
App Req/sec: 7,000
App Healthy: True
App Req/sec: 8,000
App Healthy: True
App Req/sec: 7,000
App Healthy: True
App Req/sec: 6,000
App Healthy: True
App Req/sec: 8,000
App Healthy: True
App Req/sec: 8,000
App Healthy: True
App Req/sec: 8,000
App Healthy: True
3

- Application packaging
- Linux at scale
- Clustering
Application packaging

Linux at scale

Clustering
Follow Along Instructions

https://github.com/philips/repositories

2016-OSCON-containers-at-scale
CoreOS+Kubernetes
vagrant, aws, bare metal, etc

coreos.com/kubernetes/docs/latest/
kubernetes

architecture in practice
works on 1 node too
kube-aws

Initial Cluster Setup
controller

scheduler, etcd & API

worker kubelet

worker kubelet
Demo

Boot up a Cluster
Demo
Run an App
Demo

Understand the Network
Domains

Let's Talk About Failure
Failure domains are regions or components of the infrastructure which contain a potential for failure.
These regions can be physical or logical boundaries, and each has its own risks and challenges to architect for.
Failure Feud

- Machine Failure
  - Network/Disks/RAM/Processor/Power Supply
- Rack Failure
  - Network/Power
- Data Center Failure
  - Network/Power/Fire/Semi-trucks
- Internet Failure
  - Network/Political/Natural
Failure Analysis

Kid Celebrating
Kid Hitting His Eye Failure Analysis

- Failure is caused by human error
- Celebration continues; eye unnecessary
- Kid has two eyes can continue seeing
- Brain elects new eye automatically
Primary Datastore

etcd operations
/etc distributed hence, the name...
a clustered key-value store

GET and SET operations
a building block for higher order systems

primitives for building reliable distributed systems
Demo

play.etcd.io
[PUT] success! "foo" : "bar" (at 2016-03-16 05:38:17 PST)
etcd1
(Hash: 2219536697)

etcd5
(Hash: 2219536697)

etcd2
(Hash: 221)

etcd4
(Hash: 2219536697)

etcd3
(Hash: 2219536697)

ID: b414e782d52bc45a
Endpoint: 10.128.0.2:1278
State: Leader
Number of Keys: 5
Hash: 2219536697

Log

[PUT] success! "foo": "bar" (at 2016-03-16 05:38:17 PST)
Log

[PUT] success! "foo" : "bar" (at 2015-03-16 05:38:17 PST)
Log

[PUT] success! "foo" : "bar" (at 2016-03-16 05:38:17 PST)
etcd1

Kill  Restart

ID: unknown
Endpoint: 10.128.0.2:1278
State: unreachable
Number of Keys: 0
Hash: 0

[PUT] error grpc: timed out trying to connect (key: "foo")
[PUT] error grpc: timed out trying to connect (key: "foo")
Failure Analysis

etcd
kube-aws

high availability in cloud
etcd protects against

- Machine Failure
  - Replication, automatic leader election

- Flakey Disk Failure
  - CRC checksums on WAL files

- Network Failure
  - Timeouts and linearized state machine
etcd does not protect against

- Denial of Service
  - Future work on proxies

- Lying etcd Peers
  - We do a ton of functional testing a hedge

- Buggy or Broken Clients
  - Client deleting all keys requires restore from backup
Demo

etcd restore backup
Indexes
Kubernetes Control

API Service, Scheduler, Controller Manager
Failure Analysis

Kubernetes
Demo

etcd down for API server
Demo

etcd restore for API server
scheduler
& API
Demo
node partition from API
Demo

node scaling up
scheduler & API

worker kubelet
worker kubelet
worker kubelet
Demo

node scheduled outage API
Demo

node unplanned outage
Demo
	node downgrade/upgrade outage
Future Work

Upstream Kubernetes and Elsewhere
Upstream

- rktnetes
- Auth/OIDC
- Node self-signed TLS
Scaling

- 15x scheduler performance
- 30k pods on 1k nodes
- SIG-scale
Automatic Node Drain

Locksmith Design Doc
Performance etcd3 /ZooKeeper snapshot disabled
Performance etcd3 /ZooKeeper snapshot disabled
Memory

512MB data - 2M 256B keys
Sounds good, but...

Is anyone successful with all this in prod?
Publically traded options exchange
Containers on CoreOS are powering ISE's high-throughput, low-latency financial exchange.

- Running in production
- Bare metal & AWS
- Billions of transactions a day
- 150 million req/sec
MISSION

Secure the Internet
STRATEGY

Separate Apps from OS
STRATEGY

Make Servers Consistent
STRATEGY

Tolerate Machine Failures
STRATEGY

Make Servers Easy to Upgrade
STRATEGY

Simplify Application Upgrades
Thank you!

We’re hiring in all departments!  Email: careers@coreos.com  Positions: coreos.com/ careers