instaclustr

Cassandra and Kubernetes

Adam Zegelin, SVP of Engineering @ Instaclustr

July 2018

/usr/bin/whoami

- Adam Zegelin

  - SVP of Engineering/Co-founder @ Instaclustr

- We provide managed Cassandra, Spark and Kafka in the cloud (AWS, GCP, Azure & Softlayer).

- We provide support (24x7x365) and services for customers in private data centers.

- Manage and support 2k+ nodes.

Agenda

- Containers and Kubernetes

- Kubernetes and state
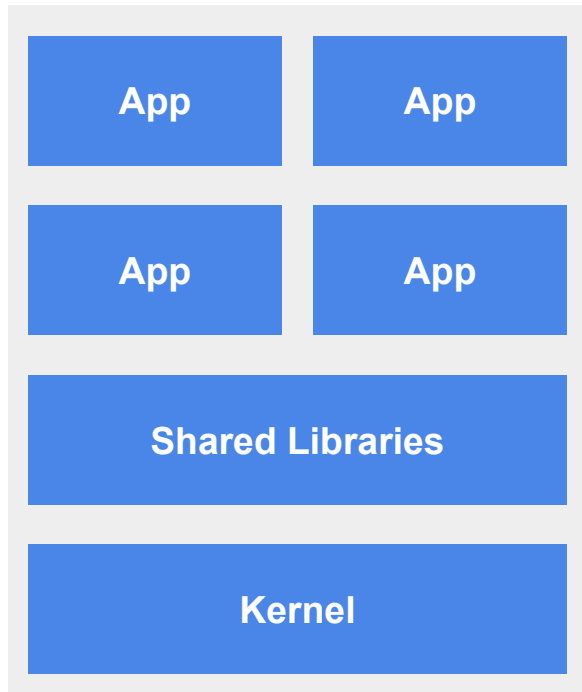
- Running Cassandra on Kubernetes
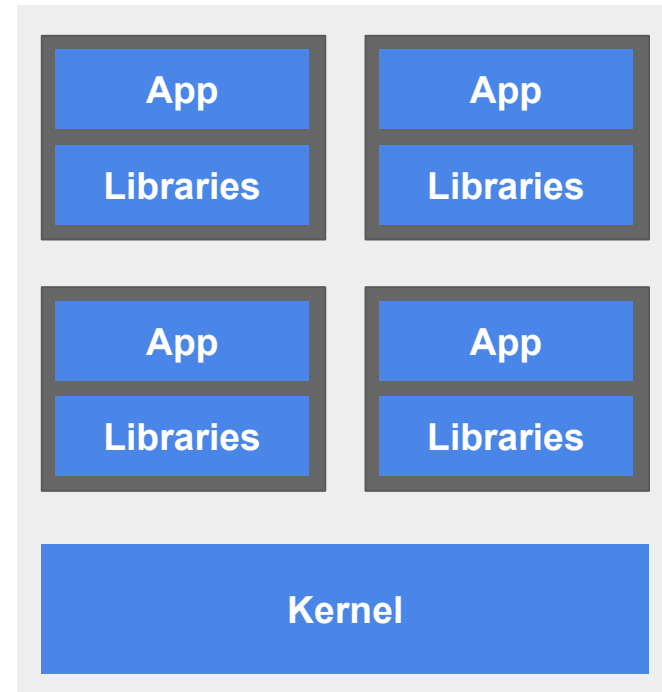
Containers - For managers

Essentially a way to bundle all the dependencies of a given process and keep it isolated…

Containers - For managers

instaclustr

Old: Applications on host

App	App
App	App

Shared Libraries

Kernel

New: Apps in Containers

App	App
Libraries	Libraries
App	App
Libraries	Libraries

Kernel

Containers - For managers

What does this actually get you

- Separation of concerns.
  - Developers can build an application/service and deliver it as a container that has defined interfaces.
  - Operators don't (generally) care what's inside the container.
- Reproducible artefacts that are the same across all environments.
  - That image you built on your laptop can be validated, tested and put into production with no changes.
- "Lightweight VMs"
- Simplifies package management
- A building block for a microservices architecture

Containers - For engineers

A container is made up of a few things:

- Process and resource isolation. Shares the host kernel but can't "see" other processes, can have CPU, IO and RAM limits, etc.

- Equivalent of a chroot environment. Bring your own userland. Need specific/unique libraries, services, binaries, or distro for your app? Done.

- Some sort of image. Contains files, binaries, libraries, for everything that will be run in the isolated environment. Typically an archive, but can also be a disk image (separate filesystem)

Containers

**Containers
(Linux)**

Cgroups
Namespaces
chroot env
AUFS
etc...

**Zones
(Solaris)**

First class
concept

**Jails
(BSD)**

First class
concept

VMs

First class
concept

Containers

This mix of components is not a bug, but a feature!

- VMs, Jails, Zones, etc. do everything for you, with minimal choice.

- Containers, everything is optional or pluggable

  - Want to allow two container to share the same network namespace? Sure go for it!

  - Don't want AUFS? Fine, use BTRFS.

  - Want a good filesystem? Mount a host directory into the container (yay XFS)

- Docker, rkt, containerd, Kubernetes, etc. all try to give you sane defaults so that containers work (somewhat) like VM/Jail/Zones.

Containers

Awesome, so a container is an isolated process that gets its own userspace,
which has the side effect of making operations easier!

Kubernetes - For managers

A service that runs your containers for you, across lots of servers and tries to be smart about it.

Kubernetes - For managers

Officially: Kubernetes is an open-source platform designed to automate deploying, scaling, and operating application containers.

Kubernetes - For managers

It won the war:

AWS ECS, Mesosphere, Docker Swarm

All support Kubernetes as a first class citizen

Kubernetes - For managers

And it's taking over the world:

AWS EKS

Google Cloud Kubernetes Engine

Azure Kubernetes Service

Red Hat OpenShift

Pivotal Kontainer Service

CoreOS

Mesosphere

Docker Swarm

Kubernetes - For engineers

Kubernetes is made up of a few things:

- A database that manages state.

- Services that manage your system and move it from its current state to its intended state

- Tools, APIs, methods and formats for telling Kubernetes what state you want it to be in.

Kubernetes

What do you get with Kubernetes? A lot!

- Managing dependent/related containers

- Managing storage

- Distributing secrets

- Managing application health

- Replication

- Scaling

- Load balancing

- Updates

- RBAC!

- more

Fundamentals of Kubernetes

Before we get any deeper, an introduction to some Kubernetes specific terms

- K8s (industry approved abbreviation for Kubernetes)
- Pod - Represents a running process on your cluster.
- Controller - A control loop that resolves intended state to actual, the fundamental automation process in Kubernetes. E.g.
  - ReplicaSet - A object that defines how pods should be replicated. i.e. N pods for a ReplicaSet
  - Deployment controller - declarative updates for Pods and ReplicaSets.

Fundamentals of Kubernetes

Controllers are the primary method of mutating infrastructure in Kubernetes. All controllers use the following basic control loop:

- Observe - Gather the current state of the system
- Analyze - Determine the differences between the current state and intended state
- Act - Implement a single action to drive current state closer to intended state.

Kubernetes

instaclustr

But also it sucks… at dealing with state

Kubernetes - Baby steps

Kubernetes has evolved on managing state as it has matured:

- PetSets in Kubernetes 1.3

- StatefulSets in Kubernetes 1.5 (beta)

- StatefulSets in Kubernetes 1.9 (GA)

Kubernetes - StatefulSets

- StatefulSet maintains a sticky identity for each of their Pods.
- StatefulSets are managed by a controller like any other Kubernetes component.
- You use StatefulSets when you need any of the following:
    - Stable, unique network identifiers.
    - Stable, persistent storage.
    - Ordered, graceful deployment and scaling.
    - Ordered, graceful deletion and termination.
    - Ordered, automated rolling updates.

Kubernetes

So we now have the building blocks for managing state in Kubernetes

Kubernetes

instaclustr

Let's take a step back

Putting it all together

- Containers - Build, run and deploy things easier

- Kubernetes - Run, manage, operate things easier

- Kinda hard to run stateful things, but the fundamentals are there.

So... what about Cassandra?

- As Kubernetes becomes a defacto orchestration API, people will (and do) want to run Cassandra on Kubernetes

- It's easy to get started, harder to run.

- For Instaclustr, Kubernetes does a lot of what we had to do in the past

  - It abstracts the environment we run in

  - Let's us focus on doing cool Cassandra things

  - Less focus on doing boring cloud things

Introducing Cassandra-operator

- Let's build something that runs and **operates** Cassandra in Kubernetes
- Cassandra-as-a-Service on top of Kubernetes
- Instaclustr in a box

Introducing Cassandra-operator

instaclustr

- Let's build something that runs and **operates** Cassandra in Kubernetes
- Cassandra-as-a-Service on top of Kubernetes
- Instaclustr in a box
  - **Open Source!**

Awesome!...what does it get me?

instaclustr

- Operations "free" Cassandra
- Consistent, reproducible environments
- Best practices are built-in
- Let's your team focus on what matters

What is an operator?
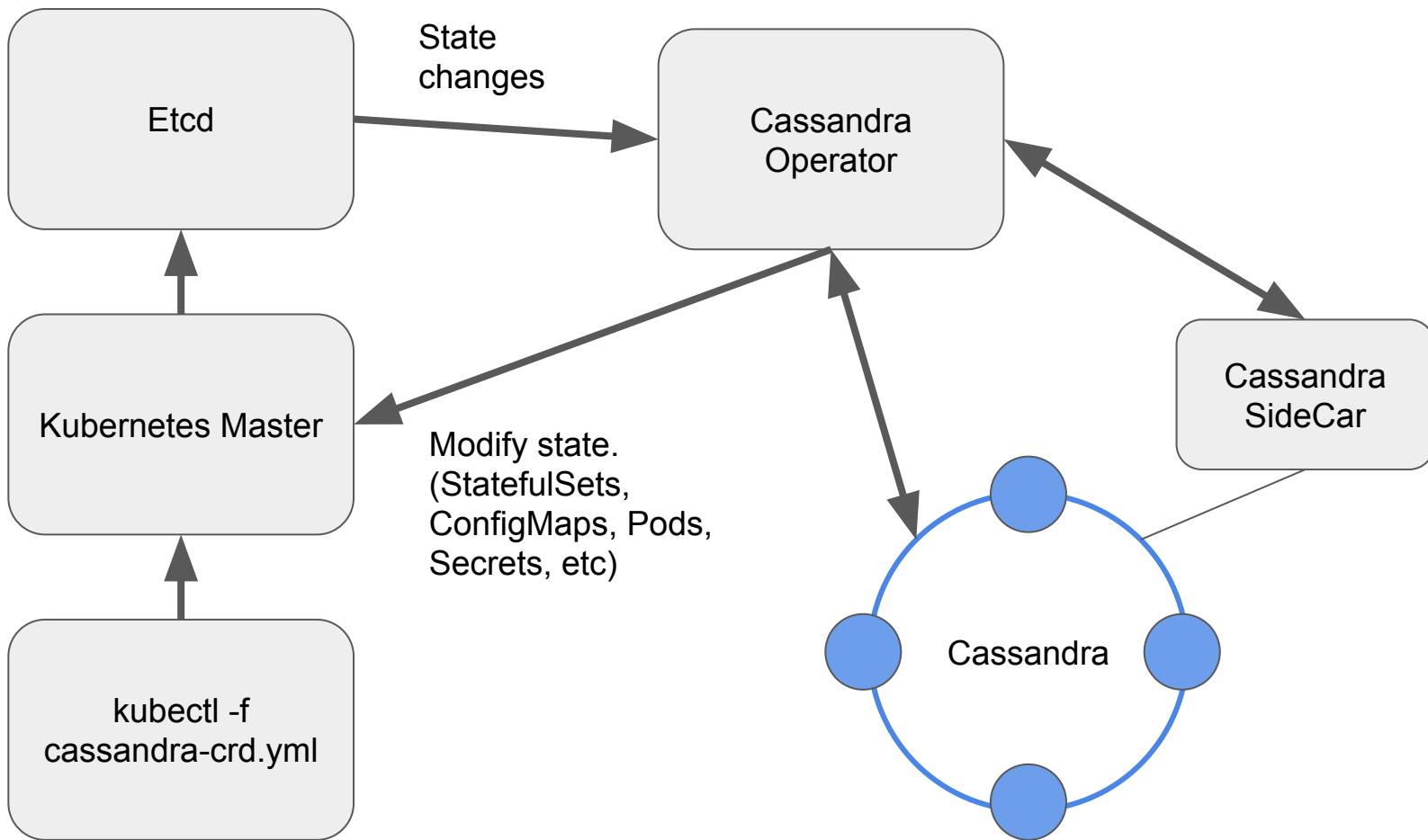
A Kubernetes operator consists of two components:

- A controller
- A Custom Resource Definition

How does it work?

- A custom resource definition (CRD) allows end users to create "Cassandra" objects -- Clusters and Data Centers -- in Kubernetes.
  - Contains configuration options for Cassandra (e.g. node count, JVM tuning options).
- The Cassandra controller listens to state changes on the Cassandra CRDs.
- Modifies StatefulSets and other K8s resources to match the requirements specified in the Cassandra CRD.

How does it work?

◌ instaclustr

Etcd

State changes

Cassandra Operator

Kubernetes Master

Modify state. (StatefulSets, ConfigMaps, Pods, Secrets, etc)

Cassandra SideCar

kubectl -f cassandra-crd.yml

Cassandra

Where to get it

https://github.com/instaclustr/cassandra-operator

- Pull requests accepted
- See the project issue tracker for current and future features