k8gb meets Cluster API

Jirka Kremser
This slide deck: bit.ly/k8gb-capi
3D Printing
jkremser & life, slide #2
road trippin’
2 kids (boys, 8 and 10)

drone “pilot” - [youtube link]
Motivation

“High availability of services as code”

- no vendor lock-in
- no external health checks
- no webui clicking / cloud provider cli tweaks
Part 1 - k8gb

- dns based global load balancer
- relies on k8s readiness probes
- decentralized - no SPoF
- CRD or annotation based
- ships internally own coredns and external dns components

![k8gb-gif](/k8gb-io/k8gb-manim/blob/master/example/k8gb.gif)
Topography

Clusters share common configuration and same workload applications. K8gb is not opinionated on how the config is delivered to the cluster.
Clusters share common configuration and same workload applications. K8gb is not opinionated on how the config is delivered to the cluster.
GitOps

How to address configuration drift?
What if we want also k8s clusters as a code?

- flux/argocd
- something that can create and manage k8s clusters
Part 2 - Cluster API

```bash
clusterctl generate cluster my-cluster
   --kubernetes-version v1.24.11
   --control-plane-machine-count=1
   --worker-machine-count=3 | kubectl apply -f -
```

```
$ example-capi-clusters master ?16
> clusterctl describe cluster -n org-giantswarm gcapeverde

<table>
<thead>
<tr>
<th>NAME</th>
<th>READY</th>
<th>SEVERITY</th>
<th>REASON</th>
<th>SINCE</th>
<th>MESSAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cluster/gcapeverde</td>
<td>True</td>
<td></td>
<td></td>
<td>3d19h</td>
<td></td>
</tr>
<tr>
<td>ClusterInfrastructure - VSphereCluster/gcapeverde</td>
<td>True</td>
<td></td>
<td></td>
<td>3d19h</td>
<td></td>
</tr>
<tr>
<td>ControlPlane - KubeadmControlPlane/gcapeverde</td>
<td>True</td>
<td></td>
<td></td>
<td>3d19h</td>
<td></td>
</tr>
<tr>
<td>__Machine/gcapeverde-4pl9j</td>
<td>True</td>
<td></td>
<td></td>
<td>3d19h</td>
<td></td>
</tr>
<tr>
<td>Workers</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MachineDeployment/gcapeverde-worker</td>
<td>True</td>
<td></td>
<td></td>
<td>3d19h</td>
<td></td>
</tr>
<tr>
<td>3 Machines...</td>
<td>True</td>
<td></td>
<td></td>
<td>3d19h</td>
<td></td>
</tr>
</tbody>
</table>
```

See gcapeverde
Management cluster

Workload cluster (my-cluster1)

A --- B
A is a management cluster for cluster B
A is a management cluster for cluster B.
workload cluster (my-cluster1)
workload cluster (my-cluster2)
workload cluster (my-cluster3)
workload cluster (my-cluster4)
management cluster
kind@localhost
AWS
vSphere
Azure
clusterctl move ..
Components

- Core
- Infrastructure
- Bootstrap
- Control Plane
Components

- Core
- Infrastructure
- Bootstrap
- Control Plane

<table>
<thead>
<tr>
<th>controllers</th>
<th>CRDs</th>
</tr>
</thead>
<tbody>
<tr>
<td>capi-controller-manager</td>
<td>Cluster</td>
</tr>
<tr>
<td></td>
<td>MachineDeployment</td>
</tr>
<tr>
<td></td>
<td>MachineHealthCheck</td>
</tr>
<tr>
<td></td>
<td>MachinePool</td>
</tr>
<tr>
<td></td>
<td>Machine</td>
</tr>
<tr>
<td></td>
<td>MachineSet</td>
</tr>
<tr>
<td></td>
<td>ClusterClass</td>
</tr>
</tbody>
</table>
## Components

- **Core**
- **Infrastructure**
- **Bootstrap**
- **Control Plane**

VMs, Networking, security groups (aws)

<table>
<thead>
<tr>
<th>controllers</th>
<th>CRDs</th>
</tr>
</thead>
<tbody>
<tr>
<td>cap(a, z, g, v, vcd, d, ..)-controller-manager</td>
<td>FooCluster</td>
</tr>
<tr>
<td></td>
<td>FooMachineTemplate</td>
</tr>
<tr>
<td></td>
<td>FooMachine</td>
</tr>
<tr>
<td></td>
<td>FooClusterIdentity</td>
</tr>
<tr>
<td></td>
<td>FooControlPlane</td>
</tr>
<tr>
<td></td>
<td>FooMachinePool</td>
</tr>
<tr>
<td></td>
<td>...</td>
</tr>
</tbody>
</table>

let Foo ∈ {AWS, Azure, GCP, VSphere,..}
Components

● Core
● Infrastructure
● Bootstrap
● Control Plane

<table>
<thead>
<tr>
<th>controllers</th>
<th>CRDs</th>
</tr>
</thead>
<tbody>
<tr>
<td>capi-kubeadm-bootstrap-controller-manager</td>
<td>KubeadmConfigTemplate</td>
</tr>
<tr>
<td></td>
<td>KubeadmConfigs</td>
</tr>
</tbody>
</table>

Controller converts KubeadmConfig bootstrap object into a cloud-init or ignition script that is going to turn a Machine into a Kubernetes Node using kubeadm (also MicroK8s impl, EKS, Talos)
Components

- **Core**
- **Infrastructure**
- **Bootstrap**
- **Control Plane**

<table>
<thead>
<tr>
<th>controllers</th>
<th>CRDs</th>
</tr>
</thead>
<tbody>
<tr>
<td>capi-kubeadm-control-plane-controller-manager</td>
<td>KubeadmControlPlaneTemplate</td>
</tr>
<tr>
<td></td>
<td>KubeadmControlPlane</td>
</tr>
</tbody>
</table>

etcd, coredns, image repo
KubeadmControlPlane CR has also KubeadmConfig included
(each CP is also a node)
Whole Picture
Whole Picture

Cluster

KubeadmControlPlane

MachineDeployment

KubeadmConfigTemplate

VSphereMachineTemplate

MachineSet

Machine

KubeadmConfig

VSphereMachine

VSphereCluster

VSphereClusterIdentity

N x workers, M x CPs
Whole Picture

- Cluster
  - AWSManagedCluster
  - AWSManagedControlPlane
  - AWSMachineTemplate
  - MachinePool
    - AWSMachinePool
    - EKSConfig
  - .spec.controlPlaneRef
  - .metadata.ownerReferences
  - .spec.infrastructureRef
  - .spec.template.spec.infrastructureRef
  - .spec.template.spec.bootstrap.configRef
  - .spec.template.spec.infrastructureRef
Demo

Setup

- Argo CD
- ClusterAPI
- example app
- k8gb

"..then we merged a pr"
Demo AWS cluster in eu

Kind cluster

git repo

AWS cluster in us

What happened?

1. Merged PR with new cluster
2. k8s cluster was created
3. c{n,s,p}i was installed
4. k8gb was installed and configured
5. example app and nginx was installed

repo: jkremser/fosdemo-clusters
ascii recording
What’s there (by CAPI)

- control plane for other clusters
- scaling (nodes)
- cluster updates (k8s version)
- infrastructure as code
What’s also there (bring your own YAMLs)

- CNI (Container Network Interface) - Cilium
- CSI (Container Storage Interface) - aws-ebs-csi-driver
- CPI (Cloud Provider Interface) - AWS’ cloud-controller-manager
- global load balancing using k8gb
- nginx as ingress controller
- app
Content Delivery

- CAPI + ClusterResourceSet
- `cluster-api-addon-provider-helm` + HelmChartProxy
- Flux + HelmRelease
- secrets? (SOPS, Vault CSI / agents)
- operator in MC doing something to WCs → custom
How to start

- **CAPD**
  a. create local cluster with Kind
  b. `clusterctl init --infrastructure docker`
  c. `clusterctl generate cluster foo | k apply -f -`

- **k8gb** - local playground setup described at k8gb.io
Takeaways

- k8gb and other distributed systems that require similar configuration across multiple clusters fits well with CAPI

- we achieved real high availability as code
Thank You!

Q&A

bit.ly/k8gb-capi

jiri.kremser@gmail.com