Introduction to SIG Cluster Lifecycle

Alexander Kanevskiy, Di Xu
2019-06-25
Who are we?

Di Xu
Kubernetes Member
Ant Financial, China
@dixudx

Alexander Kanevskiy
Kubernetes Member
Intel, Finland
@kad
Who are SCL?

- 600+ members on mailing list
- 2000+ members in #sig-cluster-lifecycle Slack
- 20+ companies represented during SIG meetings
- 5 continents with contributors
- O(10^3) contributions per cycle
- 15+ SIG sponsored subprojects
“SIG Cluster Lifecycle’s objective is to simplify creation, configuration, upgrade, downgrade, and teardown of Kubernetes clusters and their components.”

-- SIG Cluster Lifecycle Charter
Why?

● To prevent the mistakes of other open source clustering tools, as...
  ○ Kubernetes is the beginning of the story, not the end
  ○ Commoditizing the deployment of the core raises all boats and allows the community to focus on solving end user problems
    ○ “Production Grade” shouldn’t be firewalled
  ○ It should “just work”
  ○ Because cross provider behavior matters (conformance)
● To make the management of (X) clusters across (Y) providers simple, secure, and configurable.
SCL Overview

SCL is one of the biggest Kubernetes SIGs, with 100s of contributors across several companies actively contributing to 17 subprojects and several workgroups.

**k8s cluster provisioners:**
- minikube
- kops
- kubespray
- kind (SIG Testing)
- kubeadm-dind-cluster
- cluster-api-provider-<name>
- ...

**Cluster API**
- cluster-addons
- kubeadm
- etcdadm
Key Subprojects
kubeadm’s task is to set up a **best-practice cluster** for each *minor version*

- The user experience should be *simple*, and the cluster reasonably *secure*

- *kubeadm’s scope is limited*; intended to be a **building block**
  - Only ever deals with the local filesystem and the Kubernetes API
  - Agnostic to *how exactly* the kubelet is run
  - Setting up or favoring a specific CNI network is *out of scope*

- **Composable architecture with everything divided into phases**
  - Allows for **DIY** using other higher order tools as chef/puppet/etc.
kubeadm (GA)

= The official tool to bootstrap a minimum viable, best-practice Kubernetes cluster
kubeadm is built to be part of a higher-level solution
kubeadm Survey

How would you rate the overall kubeadm experience?

Difficult → Easy

Are you running High Availability clusters created by kubeadm?

- yes
- no
- may be
• Kubeadm configuration API v1beta2
• Better certificate management on upgrades
• Entirely new test suite for ensuring stability
• High Availability control plane

• One more thing: Kubeadm new logo!
  • Special thanks to Alex Contini (@alexcontini)
The What and the Why of Cluster API “To make the management of (X) clusters across (Y) providers simple, secure, and configurable.”

- “How do I provision all the other infrastructure I need for a Kubernetes cluster (load balancers, VPC, etc.)?”
- “How do I manage other lifecycle events across that infrastructure (upgrades, deletions, etc.)?”
- “How can I manage any number of clusters in a similar fashion to how I manage deployments in Kubernetes?”
- “How can we control all of this via an API?”
Cluster API

Tools atop of Cluster API

- kops
- kubicorn
- Multiple control plane managers
  - SAP Gardener
  - KaaS layers
With Kubernetes we manage our applications declaratively.

a. Why not for the cluster itself?

With the Cluster API, we can declaratively define the desired cluster state.

a. Operator implementations reconcile the state
b. Use Spec & Status like the rest of k8s
c. Common management solutions for e.g. upgrades, autoscaling and repair
d. Allows for “GitOps” workflows

```yaml
apiVersion: cluster.k8s.io/v1alpha1
kind: MachineDeployment
metadata:
  name: my-nodes
spec:
  replicas: 3
  selector:
    matchLabels:
      foo: bar
  template:
    metadata:
      labels:
        foo: bar
    spec:
      providerConfig:
        value:
          apiVersion: "baremetalconfig/v1alpha1"
          kind: "BareMetalProviderConfig"
          zone: "us-central1-f"
          machineType: "n1-standard-1"
          image: "ubuntu-1604-lts"
          versions:
            kubelet: 1.14.2
            containerRuntime:
              name: containerd
              version: 1.2.0
```
WG Component Standard

● Problem 1: The core Kubernetes components are not consistent in
  ○ how they are configured
  ○ how they should be set up
  ○ what HTTP(S) endpoints they register
  ○ how they do (delegated) auth

● Problem 2: It’s pretty hard to write a k8s-like component with declarative config

● Solution: Factor common component-related code into a `k8s.io/component-base` toolkit repository. Make it easier to write a non-core component that follows the k8s style
ComponentConfig

- **Maintainability:**
  When $component’s flag set grows over 50+ flags, configuring it becomes painful

- **Upgradability:**
  On upgrades, $component still works using versioned config vs. flags

- **Programmability:**
  Configuration expressed as JSON/YAML objects allows for consistent manipulation

- **Possibility:**
  Many types of config simply can’t be expressed as simple key-value

- **Declarative:**
  OpenAPI information can easily be exposed / used for doc generation

- See Lucas’ talk on this here: Configuring Your Kubernetes Cluster on the Next Level
```yaml
apiVersion: kubecontrollermanager.config.k8s.io/v1
kind: KubeControllerManagerConfiguration
controllers:
  csrSigning:
    clusterSigningCertFile: /some/path
namespace:
  concurrentNamespaceSyncs: 5
nodeLifecycle:
  enableTaintManager: true
```

```bash
$ kube-controller-manager --config config.yaml
```
Getting Involved!
Getting Involved!

SIG Cluster Lifecycle

- 100s of contributors across several companies
- We’re working on growing the contributor/reviewers pool
- We have many EMEA contributors

SIG Cluster Lifecycle and China

- SCL China bi-weekly meetings
- Chinese Friendly Time
- Collecting appropriate time slots
- Planned to Start from July
How can you Contribute

- **SIG Cluster Lifecycle New Contributor Onboarding**
- Look for “good first issue”, “help wanted” and “sig/cluster-lifecycle” labeled issues in our repositories (in k/k or in various project repository)
- Attend our Zoom meetings / be around on Slack
- We have “Office Hours” for our projects: weekly for kubeadm and Cluster API, bi-weekly for kops and kubespray
- Full list of SIG meetings and links to minutes and recordings can be found on [SIG page](#)
- Contributing to [SIG Cluster Lifecycle documentation](#)
The SCL Roadmap

We need your help!

There is still a lot of work to do in order to get the full puzzle in place!

Cluster API

- kubeadm
- etcdadm

Component Config

- cluster-addons
- k8s cluster Provisioners

GA
Beta
Alpha
Pre-Alpna
Other Logistics

● Follow the SIG Cluster Lifecycle YouTube playlist

● Check out the meeting notes for our weekly office hours meetings

● Join #sig-cluster-lifecycle, #kubeadm, #cluster-api, #kops-dev, #kops-users, #kubespray, #minikube, …channels

● Check out the kubeadm setup guide, reference doc and design doc

● Read how you can get involved, and watch the new contributor onboarding session!
Other SCL Talks

- Kubespray Deep Dive
  - Tuesday, June 25 • 16:45 - 17:20
  - [https://sched.co/Nrr5](https://sched.co/Nrr5)

- Minikube: Bringing Kubernetes to the Next Billion Users
  - Tuesday, June 25 • 16:45 - 17:20
  - [https://sched.co/Nrmy](https://sched.co/Nrmy)

- Check also SCL sessions from KubeCon Europe’19
Thank you!

Q/A
KubeCon
CloudNativeCon
OPEN SOURCE SUMMIT
China 2019