
KubeCon
CloudNativeCon
North America 2019

Deep Dive: Cluster API

Who are we?

Vince Prignano
Senior Software Engineer @VMware
@vincepri

Ashish Amarnath
Senior Software Engineer @Salesforce
@ashish-amarnath

Agenda

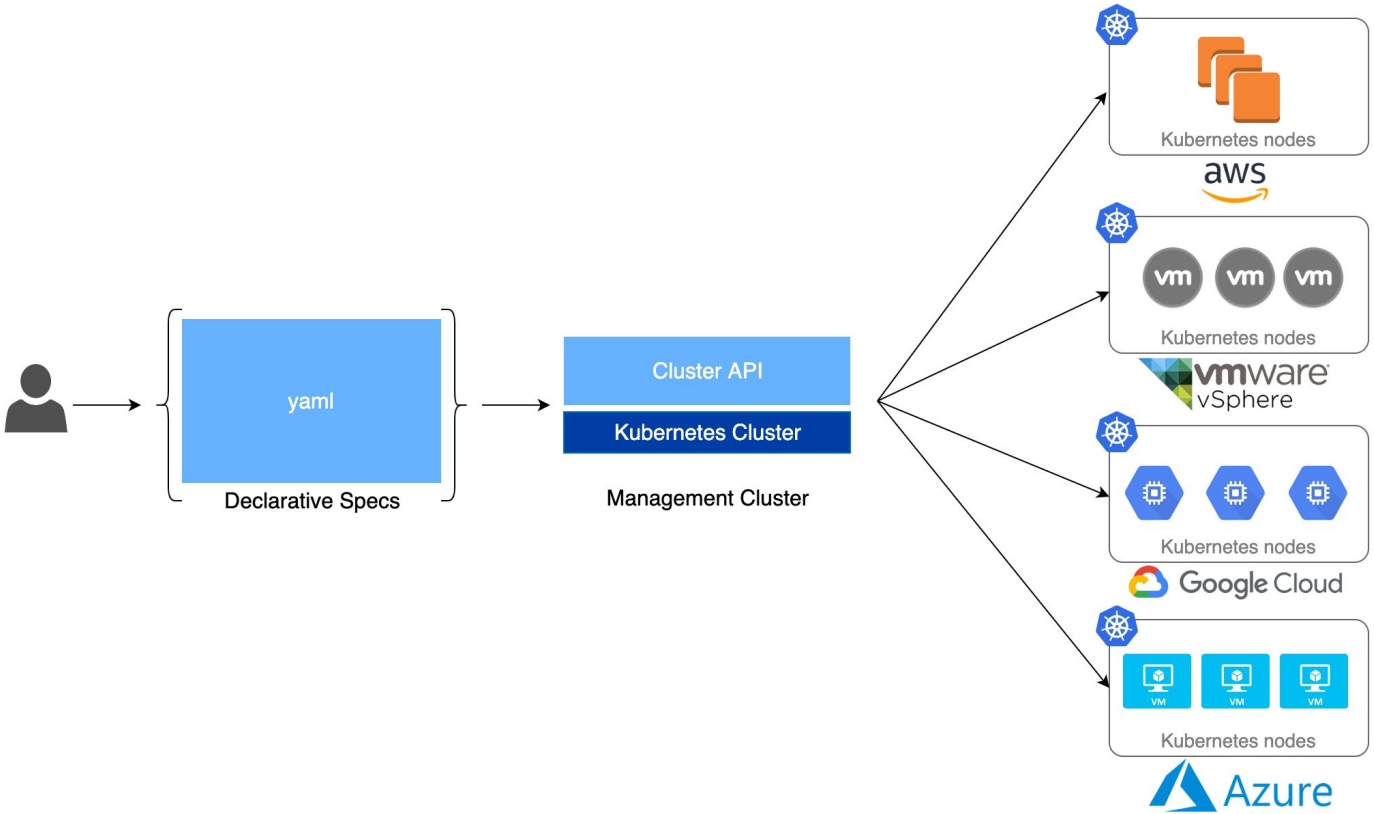
- Project overview
- Walkthrough of v1alpha2 types, architecture
- Demo
- What’s next and planned features
- How to get involved
- Q&A

Overview

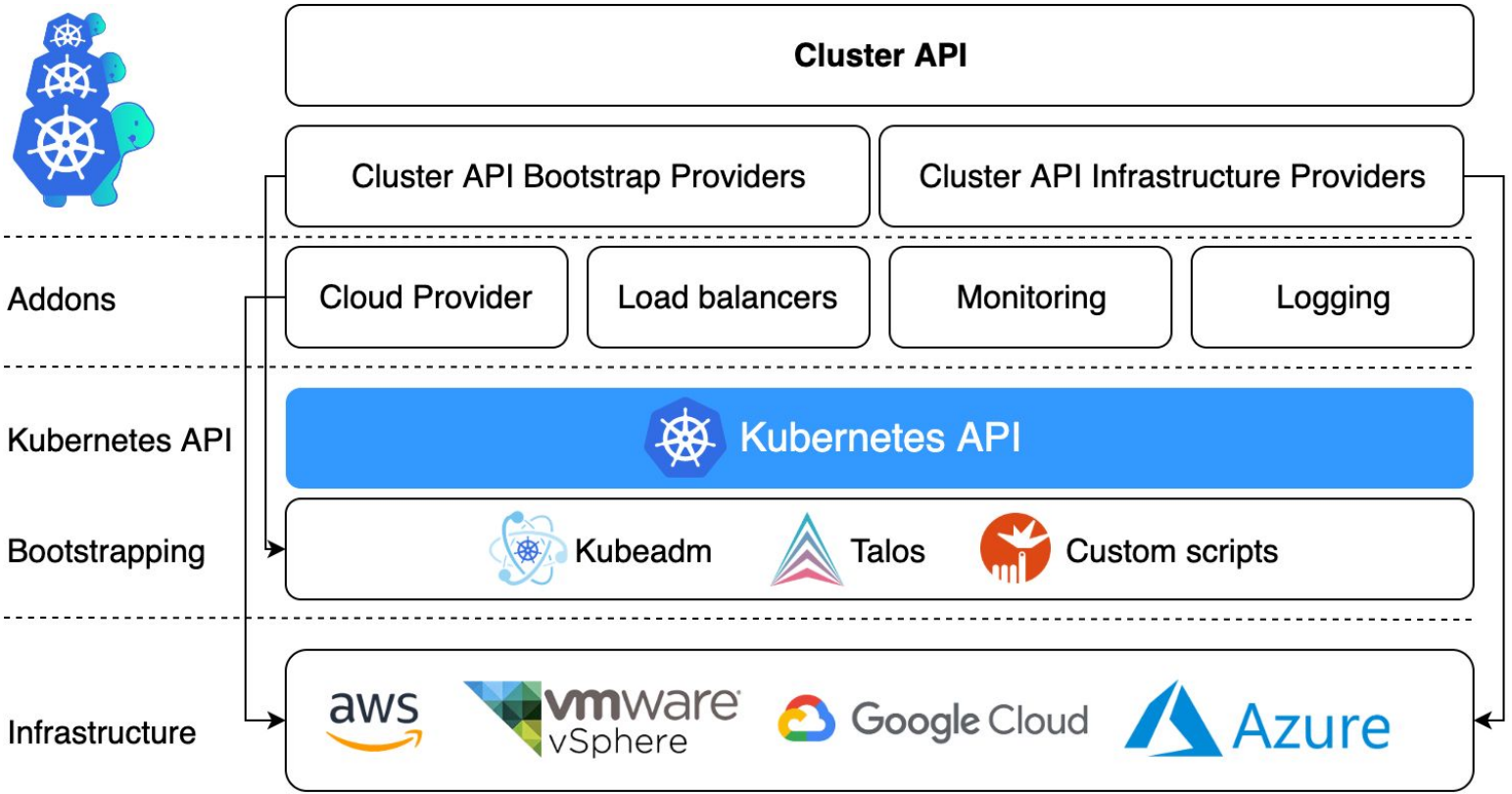
Why Cluster API?

- Cluster lifecycle management is difficult
- Ecosystem is fragmented
- Difficult to build higher order functionality
  - Managed control plane
  - Automation: autoscaling, repair, upgrades
  - Consistent cross-cloud cluster management

What is Cluster API?



Where does Cluster API fit?



Where are we now?

- SIG Cluster-Lifecycle sub-project
- 200+ Contributors
- Alpha stage
  - *v1alpha1*: build an initial implementation
  - *v1alpha2*: solid foundations, infrastructure agnostic bootstrapping, improved extensibility
  - *v1alpha3*: new higher level primitives, QoL improvements, and more! (details on this later)

Where do we want to be?

- Make managing the lifecycle of Kubernetes clusters boring
- Batteries included
  - High level tooling for the 80% use case
  - Ship with Kubeadm
  - Support target environments from development to cloud to bare metal
- Support advanced use cases: code, plugins, documentation
  - Security requirements
  - Different topologies
  - Complex scenarios

Today: v1alpha2

Cluster


```yaml
apiVersion: cluster.x-k8s.io/v1alpha2
kind: Cluster
metadata:
  name: capi-quickstart
spec:
  clusterNetwork:
    pods:
      cidrBlocks: ["192.168.0.0/16"]
  infrastructureRef:
    apiVersion: infrastructure.cluster.x-k8s.io/v1alpha2
    kind: AWSCluster
    name: capi-quickstart
```

InfraCluster


```
apiVersion: infrastructure.cluster.x-k8s.io/v1alpha2
kind: AWSCluster
metadata:
  name: capi-quickstart
spec:
  region: us-east-1
  sshKeyName: default
```

Machine

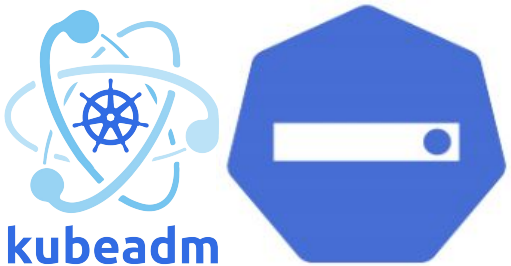

```yaml
apiVersion: cluster.x-k8s.io/v1alpha2
kind: Machine
metadata:
  name: capi-quickstart-controlplane-0
  labels:
    cluster.x-k8s.io/control-plane: "true"
    cluster.x-k8s.io/cluster-name: "capi-quickstart"
spec:
  version: v1.15.3
  bootstrap:
    configRef:
      apiVersion: bootstrap.cluster.x-k8s.io/v1alpha2
      kind: KubeadmConfig
      name: capi-quickstart-controlplane-0
  infrastructureRef:
    apiVersion: infrastructure.cluster.x-k8s.io/v1alpha2
    kind: AWSMachine
    name: capi-quickstart-controlplane-0
```

InfraMachine


```yaml
apiVersion: infrastructure.cluster.x-k8s.io/v1alpha2
kind: AWSMachine
metadata:
  name: capi-quickstart-controlplane-0
spec:
  instanceType: t3.large
  iamInstanceProfile: "control-plane.cluster-api-provider-aws"
  sshKeyName: default
```

BootstrapConfig


```
apiVersion: bootstrap.cluster.x-k8s.io/v1alpha2
kind: KubeadmConfig
metadata:
  name: capi-quickstart-controlplane-0
spec:
  initConfiguration:
    nodeRegistration:
      name: '{{ ds.meta_data.hostname }}'
      kubeletExtraArgs:
        cloud-provider: aws
  clusterConfiguration:
    apiServer:
      extraArgs:
        cloud-provider: aws
    controllerManager:
      extraArgs:
        cloud-provider: aws
```



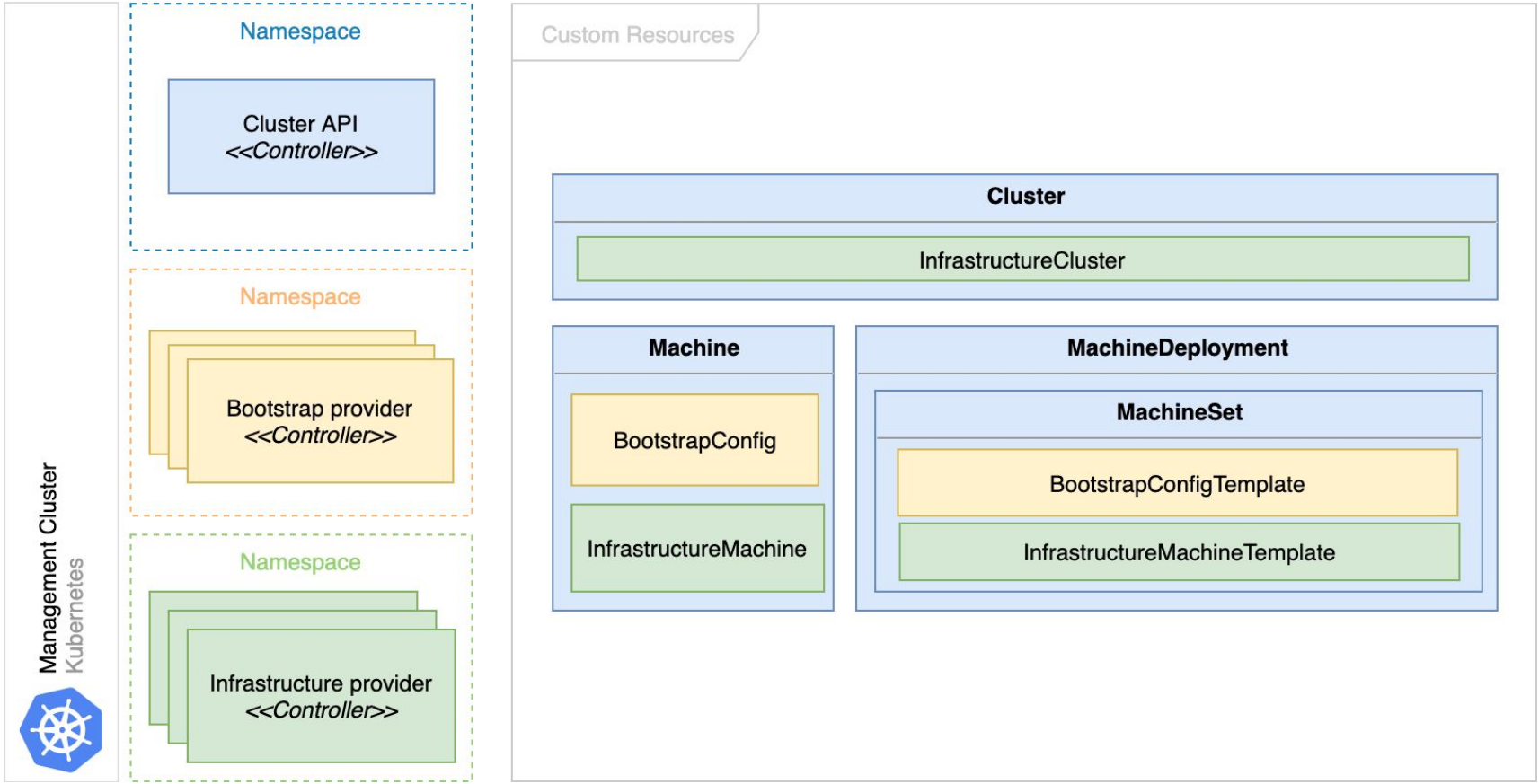
MachineDeployment


```yaml
apiVersion: cluster.x-k8s.io/v1alpha2
kind: MachineDeployment
metadata:
  name: capi-quickstart-worker
  labels:
    cluster.x-k8s.io/cluster-name: capi-quickstart
    nodepool: nodepool-0
spec:
  replicas: 1
  selector:
    matchLabels:
      cluster.x-k8s.io/cluster-name: capi-quickstart
      nodepool: nodepool-0 [ . . . ]
```

MachineDeployment


```
[ . . . ]
metadata:
  labels:
    cluster.x-k8s.io/cluster-name: capi-quickstart
    nodepool: nodepool-0
spec:
  version: v1.15.3
  bootstrap:
    configRef:
      name: capi-quickstart-worker
      apiVersion: bootstrap.cluster.x-k8s.io/v1alpha2
      kind: KubeadmConfigTemplate
  infrastructureRef:
    name: capi-quickstart-worker
    apiVersion: infrastructure.cluster.x-k8s.io/v1alpha2
    kind: AWSMachineTemplate
```

Architecture



Demo

Next: v1alpha3

Planned features

Control plane management

- Based on Kubeadm and Machines.
- Includes upgrading Kubernetes versions.

MachinePool

- Support high level primitives for cloud provider agnostic auto-scaling.

Failure domains

- Spread control plane machines and nodes across availability zones.

Clusterctl v2

- Improve user experience with a new high level tool.

Planned features

Node remediation

- Detect failures and act on them.

Testing framework

- Write end-to-end tests for Cluster API and its providers with a new simplified framework.

Validation webhooks

- Improve user experience, provide error reasons for configuration errors, force immutability.

Load balancer providers

- Provide support for pluggable load balancer implementations, especially useful for on-prem.


KubeCon
CloudNativeCon
North America 2019
Get involved!

How can you help?

📝 *Writing skills?*

- Document our Book: quick start, architecture diagrams, contracts, troubleshooting sections, and so on!

🧬 *Product skills?*

- Gather use cases, compile user pulse surveys, draw roadmaps.
- Work with project's maintainers and the community to shape our product.
- Help with backlog grooming, maintain milestones.

👩‍💻 *Coding skills?*

- Review pull requests, become an approver.
- Search for help wanted, or good first issues across our repositories.

🙌 *Other skills? Have feedback, use cases, demos, or questions?*

- Join weekly community meetings, slack, mailing list.
- Open issues, bring the what and whys.


KubeCon
CloudNativeCon
North America 2019
Q&A