Running VM Workloads Side by Side with Container Workloads
Sebastian Scheele
CEO & Co-Founder Loodse

Twitter: @sscheele
Virtualization Extension

Kubernetes: Container Platform
Virtualization Extension

Kubernetes: Container Platform

KubeVirt: Virtualization Extension
But... Why KubeVirt?

- Already have on-premise solutions like Openstack, oVirt
- And then there’s the public cloud, AWS, GCP, Azure.
- Why would we do this VM management stuff yet again?
Infrastructure Convergence

Old Way … Multiple Workloads, Multiple Stacks

VM Workload

VM Platform

OS

Metall

Logging, Metrics, Monitoring

Knowledge

Scheduling, Storage, Network

2x

Container Workload

Kubernetes

OS

Metall
Infrastructure Convergence

KubeVirt way… Multiple Workloads, One Stack

- VM Workload
- Container Workload
- Kubernetes
  - Logging, Metrics, Monitoring
  - Knowledge
  - Scheduling, Storage, Network
- OS
- Metall

1x
Workflow Convergence

- Converging VM management into **container management workflows**.
- Same tooling (**kubectl**)
- **Declarative API** for VM management (just like pods, deployments, etc...)

```
# Creating a POD
$ cat <<EOF | kubectl create -f
- apiVersion: v1
  kind: Pod
...

# Creating a Virtual Machine
$ cat <<EOF | kubectl create -f -
apiVersion: kubevirt.io/v1alpha1
kind: VirtualMachineInstance
...

spec:
  domain:
    cpu:
      cores: 2
    devices:
      disk: fedora29
...
```
Simplicity

- Drops into any existing kubernetes cluster.
- No runtime level configuration required
- No per node configuration required.
- Just works.

# As simple as posting one of our release manifests to a kubernetes cluster
$ kubectl create -f
https://github.com/kubevirt/kubevirt/releases/download/${KUBEVIRT_VERSION}/kubevirt-operator.yaml

# Then start posting Virtual Machine manifests to launch VMs.
$ kubectl create -f my-vm.yaml
Demo
Multi-Cluster with KubeVirt

- Logging, Metrics, Monitoring
- Knowledge
- Scheduling, Storage, Network
ClusterAPI

- Declarative, Kubernetes-style APIs to cluster creation, configuration, and management.
- Machine API manages the lifecycle of machines in Kubernetes.
- [https://github.com/kubernetes-sigs/cluster-api](https://github.com/kubernetes-sigs/cluster-api)
Kubernetes Native Integration

- Generic cluster scaling
- Pets vs cattle for nodes
- Implementation of generic auto scaling possible
- Very similar setup for different provider

```
kubectl create machines
```
A "Machine" is the declarative spec for a Node, as represented in Kubernetes core.

After provisioning a new Node matching the Machine spec is registered.
MachineController

Watches for new Machine resources

Provision the machine using provider-specific drivers
  • Machine joins the cluster
  • Kubelet creates the Node resource
How to implement a new provider?

```go
// Provider exposed all required functions to interact with a cloud provider

type Provider interface {

    // Validate validates the given machine's specification.
    // In case of any error a "terminal" error should be set,
    // See v1alpha1.MachineStatus for more info
    Validate(machinespec v1alpha1.MachineSpec) error

    // Get gets a node that is associated with the given machine.
    // Note that this method can return what we call a "terminal" error,
    // which indicates that a manual interaction is required to recover from this state.
    // See v1alpha1.MachineStatus for more info and TerminalError type
    Get(machine *v1alpha1.Machine) (instance.Instance, error)

    GetCloudConfig(spec v1alpha1.MachineSpec) (config string, name string, err error)

    // Create creates a cloud instance according to the given machine
    Create(machine *v1alpha1.Machine, update MachineUpdater, userdata string) (instance.Instance, error)

    Delete(machine *v1alpha1.Machine, update MachineUpdater, instance instance.Instance) error
}
```
How to implement a new provider?

```go
type Provider interface {
    Validate(machinespec v1alpha1.MachineSpec) error
    Get(machine *v1alpha1.Machine) (instance.Instance, error)
    GetCloudConfig(spec v1alpha1.MachineSpec) (config string, name string, err error)
    Create(machine *v1alpha1.Machine, update MachineUpdater, userdata string) (instance.Instance, error)
    Delete(machine *v1alpha1.Machine, update MachineUpdater, instance instance.Instance) error
}
```

github.com/kubermatic/machine-controller/../kubevirt/provider.go
Where to get it and how to use it?

- [github.com/kubevirt/kubevirt](https://github.com/kubevirt/kubevirt)
- [github.com/kubermatic/machine-controller](https://github.com/kubermatic/machine-controller)
Q&A