Why is KubeVirt possible

P E T R  K O T A S
Who is here
Few word about me

Senior developer at Red Hat.
Cloud enthusiast.
Working on KubeVirt – virtualization on Kubernetes
I love talking!
TL;DR

✓ **Custom Resource Definitions**
  They add new Kind to Kubernetes API. Automatically add CRUD for the Kind.

✓ **Custom Controllers**
  Enables to control custom Kind and react to its state.

✓ **Aggregated API**
  Enables sub-resources for custom Kind on top of standard CRUD.

✓ **How they work together**
  They are small nits, that are ought to be learned, because they can bite you back.
THE GOAL
WRITING K8S CONTROLLER

TALK THIS MORNING
Writing kube controllers for everyone, Maciej Szulik

Explained best practices for writing the k8s controllers.

YOU WILL USE IT NOW
I focus on how thing work together in Kubernetes and on how we used it in KubeVirt. With combined knowledge from both talks you are good to go to write your own Kubernetes extension.
What is KubeVirt

KubeVirt technology addresses the needs of development teams that have adopted or want to adopt Kubernetes but possess existing Virtual Machine-based workloads that cannot be easily containerized. More specifically, the technology provides a unified development platform where developers can build, modify, and deploy applications residing in both Application Containers as well as Virtual Machines in a common, shared environment.

Benefits are broad and significant. Teams with a reliance on existing virtual machine-based workloads are empowered to rapidly containerize applications. With virtualized workloads placed directly in development workflows, teams can decompose them over time while still leveraging remaining virtualized components as is comfortably desired.
DO YOU EXPECT ME TO READ THAT?
WHAT IS KUBEVIRT?

Powerful virtualization solution built on top of Kubernetes
KUBEVIRT

VIRTUAL MACHINE MANAGEMENT ON KUBERNETES
Enables the full virtual machines to run as workload managed by Kubernetes through API.

WHAT CAN I DO WITH IT

BRING THE LEGACY
The legacy code is not going anywhere near soon. KubeVirt enables the legacy code to run on modern cloud infrastructure.

MIX AND MATCH
When parts of existing workload are already virtualized, they can be easily brought to the Kubernetes. KubeVirt handles the integration with existing k8s infrastructure.

MIX OS
Sometimes it is necessary to run workload in different operating system. KubeVirt enables to run any OS as k8s workload.
COMPONENTS OF VIRTUALIZATION

The high level components of the virtualization management platform

**STORE**
Place where you can store VMs state and their metadata

**API**
Communication layer enabling you to manage the VMs

**SCHEDULER**
Overseer who knows your cluster and can figure out where to put the VMs

**HANDLER**
The one who can actually control the VMs – start, stop, restart, pause
HOW WOULD YOU BUILD IT?
This is how we built it
On Kubernetes
Of course
KUBERNETES

- kubectl
- kubelet
- API server
- scheduler
- k8s-controller
- etcd
- some-app
ETCD
Provides store for objects. It has drawbacks, but it can be worked with.

KUBE-API
Already accessible API which you can extend.

KUBE-SCHEDULER
Scheduler you can use if you follow its rules.

EXTENSIONS
Which you can use to build handlers to handle VMs.

BENEFITS FROM BUILDING ON KUBERNETES

K8s already provides solid foundation, which does not have to be handled.
Let’s build KubeVirt from scratch
VM IN A CONTAINER

Let’s put a VM into the container as first step and than into the pod.

$ qemu fedora.qcow \
    -boot c -net nic \
    -net user -m 196 -localtime

**SINGLE PROCESS**

QEMU launches the VM as a process, therefore it can run in a container without much hassle.

**COMMAND ARGUMENTS**

QEMU command line is cumbersome. You do not want to build argument builder.

**POD NETWORK**

VM can make use of pod network to reach out to the world. You can use pod network to reach in the VM.

**KVM vs EMULATION**

Also when you want to support both emulation and kvm, you need to build command line.
VM IN A POD

Let’s put a VM into the container as first step and then into the pod.

- **SINGLE PROCESS**
  Single process in a container.

- **COMMAND ARGUMENTS**
  Libvirt handles the complexity for you.

- **POD NETWORK**
  VM can make use of pod network to reach out to the world. You can use pod network to reach in the VM.

- **KVM vs EMULATION**
  Set the flag for libvirt.
VM IN A POD

Let’s put a VM into the container as first step and then into the pod.

**VIRT-LAUNCHER**

- entry point for controlling the VM and collecting metrics
- lost the benefit of single process in a container
VM IN A POD

Let's put a VM into the container as first step and then into the pod.

DEFINING POD SPECIFICATION

• Definition of VM complicates, I need to describe pod with every possible detail
• I also need to provide detailed XML specification for the VM
KUBEVIRT CORE
NOW
WE HAVE A VM.
WHAT’S NEXT?
Teach k8s to know about VM
CUSTOM RESOURCE DEFINITION

Kubernetes extension point allowing you to define a new Kind.

**KIND**

Introduces new object to the Kubernetes, which is than recognized by API and can be stored in etcd.

**API**

Kube-api server provides API endpoints for the newly created Kind. It also creates CRUD endpoints.

**VALIDATION/DEFAULTING**

OpenAPI 3.0 validation or admission webhooks you can provide custom checking and defaulting.

**FINALIZERS**

Neat feature that blocks deletion until external cleanup took place.
KUBE-API ENDPOINTS

Automatically created CRUD endpoints for newly created Kind

GET /apis/virtualmachines.kubevirt.io/v1alpha2/namespaces/*/virtualmachines/*
POST /apis/virtualmachines.kubevirt.io/v1alpha2/namespaces/*/virtualmachines/*
DELETE /apis/virtualmachines.kubevirt.io/v1alpha2/namespaces/*/virtualmachines/*
PATCH /apis/virtualmachines.kubevirt.io/v1alpha2/namespaces/*/virtualmachines/*
CUSTOM RESOURCE DEFINITION

Kubernetes extension point allowing you to define a new Kind.

DEFINING VM SPECIFICATION

• I can define my VM in a language of the VM
• The rest should happen automatically

https://kubernetes.io/docs/tasks/access-kubernetes-api/custom-resources/custom-resource-definitions/
CUSTOM RESOURCE DEFINITION

Kubernetes extension point allowing you to define a new Kind.

LAYERING CUSTOM RESOURCES

https://kubernetes.io/docs/tasks/access-kubernetes-api/custom-resources/custom-resource-definitions/
CUSTOM RESOURCE DEFINITION

Kubernetes extension point allowing you to define a new Kind.

**VirtualMachineInstance**

- Lowest level unit
- Similar to pod
- Single mortal VirtualMachine

[https://kubernetes.io/docs/tasks/access-kubernetes-api/custom-resources/custom-resource-definitions/](https://kubernetes.io/docs/tasks/access-kubernetes-api/custom-resources/custom-resource-definitions/)
CUSTOM RESOURCE DEFINITION

Kubernetes extension point allowing you to define a new Kind.

**VirtualMachine**

- Persistent VM object
- Represents single stateful immortal VirtualMachine

https://kubernetes.io/docs/tasks/access-kubernetes-api/custom-resources/custom-resource-definitions/
CUSTOM RESOURCE DEFINITION

Kubernetes extension point allowing you to define a new Kind.

**VirtualMachineInstanceReplicaSet**

- Behaves closely to Kubernetes ReplicaSet
- Allows for automatic scaling of identical VMIs

https://kubernetes.io/docs/tasks/access-kubernetes-api/custom-resources/custom-resource-definitions/
K8s UNDERSTAND VM. WHAT CAN GO WRONG?
Teach k8s how to validate VM
CUSTOM RESOURCE DEFINITION

Kubernetes extension point allowing you to define a new Kind.

VALIDATION

- OpenAPI 3.0 validation present in kube-api
- In KubeVirt the OpenAPI is generated automatically from the code, thus it is in sync

```
"v1.VirtualMachineInstancePreset": {
  "properties": {
    "apiVersion": {
      "description": "APIVersion defines the versioned schema of this representation of an object. Servers should convert recognized schemas to the latest internal value, and may reject unrecognized values. More info: https://git.k8s.io/community/contributors/devel/api-conventions.md#resources",
      "type": "string"
    },
    "kind": {
      "description": "Kind is a
      https://git.k8s.io/community/contributors/devel/api-conventions.md#resources",
      "type": "string"
    }
  }
}
```
CUSTOM RESOURCE DEFINITION

Kubernetes extension point allowing you to define a new Kind.

VALIDATING/MUTATING WEBHOOKS

- When you need to perform more complex checking
- When you need to perform defaulting of values

https://kubernetes.io/docs/reference/access-authn-authz/extensible-admission-controllers/
KUBEVIRT CORE

- **kubectl**
- **API server**
- **etcd**

**NODE**
- **virt-launcher**
- **libvirtd**
- **VM**
KUBEVIRT CORE

kubectl

API server

validation
defaulting

etcd

NODE

virt-launcher
libvirtd
VM
WHAT ABOUT CUSTOM ACTIONS ON A VM?
Teach k8s how to handle custom sub-resources
AGGREGATED API

Handle custom sub-resources for you

WHAT IT IS

• Standalone extension
• Handle custom actions e.g.:
  • virtualmachine/restart
  • virtualmachine/stats
  • virtualmachine/exec

https://kubernetes.io/docs/concepts/extend-kubernetes/api-extension/apiserver-aggregation/
AGGREGATED API

Handle custom sub-resources for you

WHY?

• Kubernetes is declarative which is not always the best solution
• Enables you to do some action

https://kubernetes.io/docs/concepts/extend-kubernetes/api-extension/apiserver-aggregation/
Thanks to aggregated api.

/apis/virtualmachines.kubevirt.io/v1alpha2/namespaces/default/virtualmachines/my_vm/restart
AGGREGATED API

Handle custom sub-resources for you

SHOULD I USE IT?

• Usually no

BUT I NEED IT

• https://github.com/kubernetes-incubator/apiserver-builder

https://kubernetes.io/docs/concepts/extend-kubernetes/api-extension/apiserver-aggregation/
VM JUST SITS THERE. WHAT IS MISSING?
Teach k8s how to control VM
VIRT CONTROLLER

Let’s operate with VM and make it start.

WHAT IS CONTROLLER?

• Controller is a state keeper.
• It watches for the desired state of its controlees.
• Once the desired state has been requested, controller puts things in a motion until the desired state is reached.

https://borismattijssen.github.io/articles/kubernetes-informers-controllers-reflectors-stores
WHAT IS CONTROLLER?

• It is actually quite simple concept

https://borismattijssen.github.io/articles/kubernetes-informers_controllers_reflectors_stores
WHAT IS CONTROLLER?

- It is actually quite simple concept
- Controller keep monitoring the etcd for changes in monitored resource
- Once there was a change it starts to apply actions

https://borismattijssen.github.io/articles/kubernetes-informers-controllers-refectors-stores
VIRT CONTROLLER

Let’s operate with VM and make it start.

EXAMPLE

vm-controller keeps an eye for a VM objects and reflects its states
VIRT CONTROLLER

Let’s operate with VM and make it start.

EXAMPLE

vm-controller keeps an eye for a VM objects and reflects its states

https://borismattijssen.github.io/articles/kubernetes-informers-controllers-reflectors-stores
VIRT CONTROLLER

Let’s operate with VM and make it start.

EXAMPLE

vmi-controller keeps an eye for a VMI objects. Once there is a VMI it creates new Pod with the guest VM.

https://borismattijssen.github.io/articles/kubernetes-informers-controllers-reflectors-stores
Let’s operate with VM and make it start.

**EXAMPLE**

scheduler keeps an eye for a POD without assigned node. It will take the pod and decide where it will be placed.

https://borismattijssen.github.io/articles/kubernetes-informers-controllers-reflectors-stores
VIRT CONTROLLER

Let’s operate with VM and make it start.

EXAMPLE

kubelet watches for a pods assigned to his node. Once there is a POD for his node, it creates it.
VIRT CONTROLLER

Let’s operate with VM and make it start.

CONTROLLERS IN KUBEVIRT

https://borismattijssen.github.io/articles/kubernetes-informers-controllers-reflectors-stores
KUBEVIRT CONTROLLERS

WE HAVE ONE FOR EVERY OBJECT
KUBEVIRT CORE

NODE

- virt-launcher
- libvirtd
- VM

API server
- virt-api
- scheduler
- k8s-controller
- kubectl
- virtctl
- virt-controller
- etcd
Unless you have to, use boilerplate for you controller
CONTROLLER BOILERPLATE

WHY?
Controller have simply too much of repeating code.

WHAT?
- https://github.com/kubernetes-sigs/kubebuilder
WHAT IS VIRT-HANDLER?
Well, glad you have asked
VIRT-HANDLER

Daemonset, present on each node. Handles every VM running on that node.

WHAT IT DOES

• Enables us to monitor and control VMs
• Communicate through sockets, since originally there was not other channel to the POD
• Calls RPC to execute commands carried on by virt-launcher
  • restart
  • stop
  • update
  • status

https://kubernetes.io/docs/concepts/workloads/controllers/daemonset/
KUBEVIRT CORE

- kubectl
- virtctl
- API server
- virt-api
- scheduler
- k8s-controller
- virt-controller
- etcd
- kubelet
- virt-launcher
- libvirtd
- VM
- virt-handler
SUMMARY

✓ Custom Resource Definitions
The simplest way of adding new Kind to Kubernetes. Use it.

✓ Custom Controllers
Write your own only when you need fine grained control. Otherwise use kube-builder or operator-sdk.

✓ Aggregated API
Write one only, when you absolutely need sub-resources. Use kube-api-builder as start.

✓ How things work together
In Kubernetes stuff can take time. Please keep that in mind when designing apps.
WAIT!
THERE IS ONE MORE THING
VIRTUAL KUBELET

WHAT?
Kubelet masquerade enabling you to plug almost anything as a k8s node.

EXAMPLES
• Alibaba Cloud ECI Provider
• Azure Container Instances Provider
• Azure Batch GPU Provider
• AWS Fargate Provider

WHERE?
• https://github.com/virtual-kubelet/virtual-kubelet
WANNA EXTEND KUBERNETES?

USE KUBEVIRT AS INSPIRATION

- https://kubevirt.io
- https://github.com/kubevirt/kubevirt
QUESTIONS
Thank you for your attention

Petr Kotas

EMAIL
petr.kotas@gmail.com

WEBSITE
kotas.tech