A KUBERNETES-NATIVE INFRASTRUCTURE

DEEP DIVE

9-May-2019
MEET THE SPEAKERS

Will McGrath

- Product Marketing Manager for Storage / HCI
- New Hampshire
- Joined Red Hat in May 2016
- 30+ years of industry experience

LinkedIn: https://www.linkedin.com/in/wmcgrath/
Mail: wmcgrath@redhat.com

Steve Gordon

- Manager, Product Management, Cloud Platforms BU
- Toronto, Canada
- Joined Red Hat in August 2010, various roles:
  - Technical Writer for Virtualization
  - Product Manager for OpenStack

Mail: sgordon@redhat.com / @xsgordon
AGENDA

1. Customer Goals
2. Introducing Kubernetes-native infrastructure
3. Architectural Deep Dive
4. Call out next steps action items
CUSTOMER GOALS

- Enjoy simplicity and agility of public cloud in an on-prem environment
- Create a consistent experience across public and on-prem
- Plan for growth in container adoption while still running VMs
REALIZING THE VISION WITH KUBERNETES POWERED HYBRID CLOUD
KUBERNETES POWERED
OPEN HYBRID CLOUD

Gives developers the freedom to innovate faster across on-premise and public clouds
RED HAT MAKES OPEN HYBRID CLOUD REAL

Containers | VMs | Serverless Functions

Red Hat OpenShift 4 + Red Hat OpenShift Container Storage + Red Hat Enterprise Linux CoreOS

BARE METAL | VIRTUAL | PRIVATE CLOUD | PUBLIC CLOUDS
CONCEPT DEMO
SEEN DURING PAUL CORMIER KEYNOTE

- Get the infrastructure for your on-prem app dev up and running quickly
- Making OpenShift on bare metal servers behave like cloud
- Run virtualized workloads side-by-side with newer container workloads
- Manage entire infrastructure from Kubernetes

#redhat #rhsummit
INITIAL TARGET USE CASES

Quick way to get started with full container infrastructure

Developer Cloud (Web & mobile app dev)

Latency sensitive applications

Sample Target Apps

Containers

- MongoDB
- Kafka
- Couchbase

VMs

- Windows
- Red Hat Enterprise Linux
- Microsoft SQL Server
LET’S DIG INTO THE TECHNOLOGY

- Kubernetes Operators Framework
- Rook
- KubeVirt
- Metal³
- Kubernetes
- Ceph
ARCHITECTURAL COMPONENTS DEEP DIVE
KUBERNETES-NATIVE INFRASTRUCTURE

Containers | VMs | Serverless Functions

Red Hat OpenShift 4 + Red Hat OpenShift Container Storage + Red Hat Enterprise Linux CoreOS

BARE METAL | VIRTUAL | PRIVATE CLOUD | PUBLIC CLOUDS
Trusted enterprise Kubernetes
● Trusted Host, Content, Platform
● Full Stack Automated Install
● Over the Air Updates & Day 2 Mgt

A cloud-like experience, everywhere
● Hybrid, Multi-Cluster Management
● Operator Framework
● Operator Hub & Certified ISVs

Empowering developers to innovate
● OpenShift Service Mesh (Istio)
● OpenShift Serverless (Knative)
● CodeReady Workspaces (Che)

Developer Preview: try.openshift.com
FULL STACK AUTOMATED INSTALL

OPENSHIFT 3

OPENSHIFT PLATFORM

OPERATING SYSTEM

INFRASTRUCTURE

OPENSHIFT 4

OPENSHIFT PLATFORM

OPERATING SYSTEM

RED HAT ENTERPRISE LINUX CoreOS

INFRASTRUCTURE
LIFE-CYCLE MANAGEMENT
OVER-THE-AIR UPDATES

- OpenShift retrieves list of available updates
- Admin selects the target version
- OpenShift is updated over the air
- Auto-update support
Operators codify operational knowledge and workflows to automate lifecycle management of containerized applications with Kubernetes.
KEY TERMS

CRD  Custom Resource Definition; Schema Extension to Kubernetes API
# KEY TERMS

<table>
<thead>
<tr>
<th>CRD</th>
<th>Custom Resource Definition; Schema Extension to Kubernetes API</th>
</tr>
</thead>
<tbody>
<tr>
<td>CR</td>
<td>Custom Resource; One record/instance/object, conforming to a CRD</td>
</tr>
</tbody>
</table>
KEY TERMS

CRD  Custom Resource Definition; Schema Extension to Kubernetes API
CR   Custom Resource; One record/instance/object, conforming to a CRD
OPERATOR  Daemon that watches for changes to resources; built from framework using the operator-sdk tools
KEY TERMS

CRD  Custom Resource Definition; Schema Extension to Kubernetes API
CR   Custom Resource; One record-instance/object, conforming to a CRD
OPERATOR Daemon that watches for changes to resources; built from framework using the operator-sdk tools
CONTROLLER Object inside the operator responsible for reconciling changes to the resource to make reality match the request
KEY TERMS

CRD: Custom Resource Definition; Schema Extension to Kubernetes API
CR: Custom Resource; One record/instance/object, conforming to a CRD
OPERATOR: Daemon that watches for changes to resources; built from framework using the operator-sdk tools
CONTROLLER: Object inside the operator responsible for reconciling changes to the resource to make reality match the request
ACTUATOR: Driver interface within some controllers used to implement custom logic while sharing common code
THE PARTS OF A RESOURCE DEFINITION

SPEC  Fields the user writes to
STATUS Fields meant to be read-only for the user
LABELS Standard metadata construct for organizing objects; searchable
ANNOTATIONS Metadata for third-party clients to decorate objects; not searchable

https://github.com/kubernetes/community/blob/master/contributors/devel/sig-architecture/api-conventions.md
Largely event driven
Resource spec settings are the source of truth
Controller makes incremental changes to bring reality into alignment with request
Writes data via Kubernetes API
No multi-object transactions; watch for race conditions
Fail fast on errors
Return “done”, “reschedule”, or “error”
BARE-METAL
## KUBERNETES MACHINE API OPERATOR

### USING KUBERNETES TO PROVISION KUBERNETES CLUSTERS

<table>
<thead>
<tr>
<th>NAME</th>
<th>NAME</th>
<th>NAMESPACE</th>
<th>REGION</th>
<th>AVAILABILITY ZONE</th>
</tr>
</thead>
<tbody>
<tr>
<td>robzumski-000-master-0</td>
<td>openshift-cluster-api</td>
<td>us-east-2</td>
<td>us-east-2a</td>
<td></td>
</tr>
<tr>
<td>robzumski-000-master-1</td>
<td>openshift-cluster-api</td>
<td>us-east-2</td>
<td>us-east-2a</td>
<td></td>
</tr>
<tr>
<td>robzumski-000-master-2</td>
<td>openshift-cluster-api</td>
<td>us-east-2</td>
<td>us-east-2a</td>
<td></td>
</tr>
<tr>
<td>robzumski-000-worker-us-east-2a-001</td>
<td>openshift-cluster-api</td>
<td>us-east-2</td>
<td>us-east-2a</td>
<td></td>
</tr>
<tr>
<td>robzumski-000-worker-us-east-2a-002</td>
<td>openshift-cluster-api</td>
<td>us-east-2</td>
<td>us-east-2a</td>
<td></td>
</tr>
</tbody>
</table>

#redhat #rhsummit
Metal³ project:
- Pronounced: Metal Kubed
- Enables bare metal host management for Kubernetes.
METAL³

Metal³ ([http://metal3.io/](http://metal3.io/)) project:

- Pronounced: Metal Kubed
- Enables *bare metal host management for Kubernetes.*

This means Metal³:

- Runs on Kubernetes.
- Is managed through Kubernetes interfaces.
METAL³

- **Actuator** conforms to the cluster-api provider interface that handles **Machine** objects by allocating bare metal hosts from the inventory maintained through the Kubernetes API.

- **Baremetal-operator** using **Ironic** to manage physical server hardware and provision images represented as **BareMetalHost** objects.
IMMUTABLE INFRASTRUCTURE WITH RHEL COREOS

- Minimal Linux distribution
- Optimized for running containers
- Decreased attack surface
- Over-the-air automated updates
- Immutable foundation for OpenShift clusters
- Ignition-based Metal and Cloud host configuration
VIRTUALIZATION INFRASTRUCTURE
Add virtual machines to your OpenShift projects as easily as application containers. Easily leverage existing VM-based services from your new workloads!
Tried and trusted RHEL kernel based virtual machine (KVM) hypervisor.
Resultant virtual machines are able to run side by side directly on the same OpenShift nodes as application containers.
STORAGE INFRASTRUCTURE
OPENSCHIFT CONTAINER STORAGE
Application portability through consistent consumption, management, and operations

ANY CLOUD. ANY APP. NO LOCK IN.
Rook project (https://rook.io/):
- Cloud-native Storage orchestrator
- Automates deployment and life cycle management of storage
  - Bootstrapping
  - Configuration, provisioning, scaling, upgrading, migration, disaster recovery, monitoring, and resource management
ROOK ARCHITECTURE

New Objects:
- Storage
- Clusters
- Storage Pools
- Object Store
- File Store

Objects:
- Deployments
- DaemonSets
- Pods
- Services
- StorageClass / PV / PVC
- ClusterRole
- Namespace
- Config Maps

Rook Operators

Client Pods
(RBD/CephFS Clients)

Management

Attach/Mount

Daemons

Rook Agent (flex)

Ceph CSI Driver

Kubelet

kubectl

Rook

Kubernetes

Operators

API

kubectl

ROOK ARCHITECTURE

#redhat #rhsummit
CEPH ON OPENSIFT WITH ROOK

ROOK pods

Agent
Agent
Operator
Agent
Agent
Agent

Mon  MGR
MDS
Mon  RGW
RGW
Mon

OSD's
OSD's
OSD's
OSD's
OSD's
OSD's

#redhat #rhsummit
RE-CAP
MAKE KUBERNETES ON-PREM INFRASTRUCTURE EASY

Easy to design
Easy to purchase
Easy to install
Easy to manage
WHAT NEXT?

Follow and participate in various upstream communities

Please contact your Red Hat sales team so we can have a focused discussion based on your needs
THANK YOU

linkedin.com/company/Red-Hat
youtube.com/user/RedHatVideos
facebook.com/RedHatInc
twitter.com/RedHat