Red Hat Summit

Building Kubernetes Operators with Ansible

Chris Short, Principal Technical Marketing Manager
Michael Hrivnak, Principal Software Engineer
Melvin Hillsman, Senior Site Reliability Engineer
Matt Dorn, Principal Software Engineer
Tim Appnel, Senior Product Manager/Evangelist

Red Hat

Module 5

Ansible K8s Modules

Interacting with Kubernetes

Kubernetes Object Definitions

```
apiVersion: v1
kind: Pod
metadata:
  name: example-app
  labels:
    app: example-app
spec:
  containers:
  - name: example
    image: companyname/example:v1.2.0
    ports:
    - containerPort: 8000
```

```
apiVersion: v1
kind: Service
metadata:
  name: example-service
spec:
  selector:
    app: example-app
  ports:
  - protocol: TCP
    port: 80
    targetPort: 8000
```

Ansible k8s Module

YAML to describe the desired state of the world

KUBERNETES/KUBECTL

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: foo
  namespace: default
data:
  color: red
```

ANSIBLE

```
- name: create foo configmap
  k8s:
    definition:
      apiVersion: v1
      kind: ConfigMap
      metadata:
        name: foo
        namespace: default
      data:
        color: “{{ color }}”
```

Templating Kubernetes resource definitions with Ansible

```
---
- name: create foo configmap
  k8s:
    definition: "{{ lookup('template', '/foo.yml') | from_yaml }}"
```

Red Hat

Ansible Roles

Roles are a package of closely related Ansible content that can be shared more easily than plays alone:

Improves readability & maintainability of complex plays

Eases sharing, reuse and standardization of automation processes

Enables Ansible content to exist independently of playbooks, projects -- even organizations

Provides functional conveniences such as file path resolution and default values

```
memcached/
├── defaults
│   └── main.yml
├── files
├── handlers
│   └── main.yml
├── meta
│   └── main.yml
├── tasks
│   └── main.yml
├── templates
├── tests
└── vars
    └── main.yml
```

Why build Operators with Ansible?

EXISTING SKILLS & ECOSYSTEM

Same tried & trusted Ansible tooling

Utilize existing skills

Supports cloud-native & traditional IT automation with one simple language

Leverages vibrant existing ecosystem

LOWER BARRIER OF ENTRY

No programming required

Faster iterations and easier maintenance

Declarative state definitions like K8s

Templating of resources

Abstraction layer & helpers that reduces necessary K8s API experience

Exercise

Next Up:

Module 6

Operators with Ansible