Kubernetes Operators
made with Ansible

Michael Hrivnak
Principal Software Engineer
@michael_hrivnak
1. What is Kubernetes
apiVersion: v1
class: Pod
metadata:
  name: example-app
  labels:
    app: example-app
spec:
  containers:
  - name: example
    image: companyname/example:v1.2.0
  ports:
  - containerPort: 8000

apiVersion: v1
class: Service
metadata:
  name: example-service
spec:
  selector:
    app: example-app
  ports:
  - protocol: TCP
    port: 80
    targetPort: 8000
2. Ansible k8s module
K8s YAML

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

Ansible Task

```yaml
---
- name: create foo configmap
  k8s:
    definition:
      apiVersion: v1
      kind: ConfigMap
      metadata:
        name: foo
        namespace: default
      data:
        color: "{{ color }}"
```
- name: create foo configmap
  k8s:
    definition: "{{ lookup('template', 'foo.yaml.j2') }}"
Ansible Role

- Packages related Ansible code for **re-use**
- Create a Role that deploys and manages your application
- Ansible Galaxy: central location to share Roles with the world

```
memcached/
│   └── defaults
│     └── main.yml
│   └── files
│   └── handlers
│     └── main.yml
│   └── meta
│     └── main.yml
│   └── README.md
│   └── tasks
│     └── main.yml
│   └── templates
│   └── tests
│     └── inventory
│     └── test.yml
│   └── vars
│     └── main.yml
```
Why use Ansible with Kubernetes

- Similar patterns of writing declarative/idempotent YAML
- Many ops teams are already familiar with Ansible
- Easy to learn, uses Jinja templating
- Capable of full day-2 management
3. Operators
What is an Operator?

- Kubernetes Controller
- Deploys and manages an application
- Human operational knowledge in code
  - Deploy
  - Upgrade
  - Scale
  - Backup
  - Repair
  - ...

Extending the Kubernetes API

- You can define Custom Resources
- Choose what fields a user can “specify”

```yaml
apiVersion: cache.example.com/v1alpha1
display: Memcached
metadata:
  name: example-memcached
spec:
  size: 3
```
Operator Pattern

- Custom Resource
  - MyApplication
  - State

- k8s API

- Controller
  - Watch Events
  - Reconciliation

- My Application
Ansible k8s modules will be used to create resources in kubernetes.
Ansible Operator

Executes Ansible code using Ansible-Runner

- Operator SDK
- Watch File
- Playbook or Role
- ansible-runner

Custom Resource

MyApplication State

k8s API

My Application

proxy

Ansible Operator will manage the status of the resource
watches.yaml

- Maps a Group Version Kind (GVK) to a role or playbook.

```yaml
# watches.yaml
---
- version: v1alpha1
  group: cache.example.com
  kind: Memcached
  playbook: /path/to/playbook
```
● Helps you create an operator
● Write using Go, Ansible, or Helm
● https://github.com/operator-framework/operator-sdk/
Phase I
Basic Install
Automated application provisioning and configuration management

Phase II
Seamless Upgrades
Patch and minor version upgrades supported

Phase III
Full Lifecycle
App lifecycle, storage lifecycle (backup, failure recovery)

Phase IV
Deep Insights
Metrics, alerts, log processing and workload analysis

Phase V
Auto Pilot
Horizontal/vertical scaling, auto config tuning, abnormal detection, scheduling tuning

HELM
ANSIBLE
GO
Spec To Parameters

**Custom Resource**

```yaml
apiVersion: <Group-Version>
kind: <kind>
metadata:
  name: <name>
spec:
  <key>: <value>
  ...
status:
  <key>: <value>
  ...
```

**Ansible Operator**

Spec values will be translated to Ansible extra vars.

Status will be a generic status defined by the operator. This will use ansible runner output to generate meaningful output for the user.
Anatomy of Operator Image

From a base Ansible Operator image:

- Add **watches.yaml**, which is a mapping of Group-Version-Kinds to a playbook or role.
- Add one or more **Ansible roles**.
Ansible Developer Experience

$ operator-sdk new memcached-operator
   --api-version=cache.example.com/v1alpha1 --kind=Memcached
   --type=ansible

Creates:

- Ansible Role
- Mapping File (watches.yaml)
- Custom Resource Definition
- Deploy manifest for the new Operator
Hybrid Use Case

- The Ansible Operator is a first class citizen of operator-sdk.
- You can extend/change your operator with golang code to make a hybrid.
- Allows you to change, compose, or reuse the Ansible Operator.

![Diagram showing relationship between Your Operator and Operator SDK with Golang code and Ansible Role]
4. Advanced Patterns
Finalizers

- A way to run code before an object gets deleted.

```
# watches.yaml
---
- version: v1alpha1
  group: cache.example.com
  kind: Memcached
  role: /opt/ansible/roles/memcached
  finalizer:
    name: finalizer.memcached.cache.example.com
    role: /opt/ansible/roles/memfin
```

This role will run while an object is being deleted.
Upgrade

- when: version < 1.2
  block:
    - name: "run upgrade tool"
      shell: runupgrade.sh --version 1.2
    - name: "do more upgrade stuff"
      shell: ...

Can be an expression using any available variables.
Backup / Restore

Create a CRD and Controller just for Backup

```yaml
# watches.yaml
---
- version: v1beta2
  group: etcd.database.coreos.com
  kind: EtcdCluster
  playbook: /opt/ansible/playbook.yaml

- version: v1beta2
  group: etcd.database.coreos.com
  kind: EtcdBackup
  playbook: /opt/ansible/backup_playbook.yaml

- version: v1beta2
  group: etcd.database.coreos.com
  kind: EtcdRestore
  reconcilePeriod: 10h
  playbook: /opt/ansible/restore_playbook.yaml
```

Defines how and when a backup should be created.

Defines workflow logic for backup.
Building Operators with Ansible

**CHALLENGE**
Operators are usually written in Go by software developers that are highly familiar with Kubernetes.

**SOLUTION**
Ansible® is a first class citizen in the Operator SDK. Ansible-based Operators provide a lower barrier to entry, faster iterations, and the power of Ansible and its ecosystem. Put more simply, an Operator is designed to watch and respond to the resources in your cluster to enable your application to run as desired. After the Operator SDK is invoked, it's Ansible code as opposed to a common approach of handling these events with Go code.

Learn the basics of Kubernetes
Key components, architecture, and how to get started
We are hiring!
Questions

Michael Hrivnak
Principal Software Engineer
@michael_hrivnak