A CLOUD GURU

Lesson Guide - Using Podman to Create a Kubernetes YAML File

Kubernetes uses YAML files to define the resources that make up containers and pods. Podman can be used to capture the configuration of local containers and pods and generate Kubernetes YAML configuration files. In this lesson, we will examine how we can use Podman to create YAML configuration files. Upon completion of this lesson, you will be able to use Podman to create Kubernetes YAML configuration files.

Resources

podman-generate-kube

Instructions

We need a `nginx` pod for Kubernetes!

We need to create a `nginx` pod for our Kubernetes environment. Since we're already familiar with how to stand up a pod using Podman, we're going to use Podman to create both the pod and generate the YAML file.

Can it be that easy?

Commands Covered

- `podman generate kube`: generates Kubernetes pod and service YAML from a Podman container or pod
- `podman pod`: manages pods
- `podman run`: runs a command in a new container from the given image
- `podman ps`: displays information about containers

Generate a Kubernetes YAML File From an Existing Podman Pod

Before we can create a YAML file, we need a pod and some containers. Let's stand up a `nginx` pod!

Create an `nginx` Pod and Containers

Before we start, let's check for pods and containers:

```
podman pod ps
```

```
podman ps -a --pod
```

We'll start by creating our pod. Remember, we want to publish port `80` in the pod to `8080` on the host. We want to name the pod `test-pod`.

To do this, we run:

```
podman pod create --name test-pod -p 8080:80
```

First, let's start the `nginx` container:

```
podman run -d --restart=always --pod=test-pod --name test-nginx nginx
```

Checking for pods and containers:

```
podman pod ps
```

```
podman ps -a --pod
```

You should see your `test-pod` pod and your `test-nginx` and infra containers.

Checking with a `curl` command:

```
curl -s http://localhost:8080
```

We can see that the default `nginx` page is working! We can now log in with our web browser, using port 8080.

Generate a Kubernetes YAML File Using a Pod

We're going to use the `podman generate kube` command to generate our Kubernetes YAML file.

Let's take a look at the command:

```
podman generate kube --help
```

Once again, checking for pods and containers:

```
podman pod ps
```

```
podman ps -a --pod
```

Let's create a YAML file from our `test-pod` pod:

```
podman generate kube test-pod -f test-pod.yml
```

Taking a look at our `test-pod.yml` file:

```
more test-pod.yml
```

We can see quite a lot of information.

If we want to take a look at our containers and pods:

```
grep test- test-pod.yml | grep name | uniq
```

We see our `test-pod` pod, and our `test-nginx` container.

Let's add another container to our pod!

Add a Container to the Pod and Generate Another YAML File

Let's add another container to our `test-pod` pod:

```
podman run -d --restart=always --pod=test-pod --name=test-ubi8 ubi8
```

Once again, checking for pods and containers:

```
podman pod ps
```

```
podman ps -a --pod
```

Let's create a second YAML file from our `test-pod` pod:

```
podman generate kube test-pod -f test-pod-2.yml
```

Taking a look at our new `test-pod-2.yml` file:

```
more test-pod-2.yml
```

Once again, we can see quite a lot of information.

If we want to take a look at our containers and pods now:

```
grep test- test-pod-2.yml | grep name | uniq
```

We see our `test-pod` pod, and our two containers, `test-nginx` and `test-ubi8`.

So, it's totally painless to generate a Kubernetes YAML file using Podman!

Notes

Recording - Environment used: Cloud Playground - Medium 3 unit RHEL 8 Cloud Server

Environment Setup:

Create your Cloud Playground server and log in.

Install the `container-tools` Application Stream:

```
sudo yum -y module install container-tools
```

You're ready to go!

Contents of `test-pod.yml` file:

```
# Generation of Kubernetes YAML is still under development!
#
# Save the output of this file and use kubectl create -f to import
# it into Kubernetes.
#
# Created with podman-2.2.1
apiVersion: v1
kind: Pod
metadata:
  creationTimestamp: "2021-03-23T21:15:58Z"
```

```
    labels:
      app: test-pod
    name: test-pod
  spec:
    containers:
    - command:
      - nginx
      - -g
      - daemon off;
      env:
      - name: PATH
        value: /usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin
      - name: TERM
        value: xterm
      - name: container
        value: podman
      - name: NJS_VERSION
        value: 0.5.2
      - name: PKG_RELEASE
        value: 1~buster
      - name: NGINX_VERSION
        value: 1.19.8
      - name: HOSTNAME
        value: test-pod
      image: docker.io/library/nginx:latest
      name: test-nginx
      ports:
      - containerPort: 80
        hostPort: 8080
        protocol: TCP
      resources: {}
      securityContext:
        allowPrivilegeEscalation: true
        capabilities: {}
        privileged: false
        readOnlyRootFilesystem: false
        seLinuxOptions: {}
      workingDir: /
    restartPolicy: Always
  status: {}
  ---
  metadata:
    creationTimestamp: null
  spec: {}
  status:
    loadBalancer: {}
```

Contents of `test-pod-2.yml` file:

```
# Generation of Kubernetes YAML is still under development!
#
# Save the output of this file and use kubectl create -f to import
# it into Kubernetes.
#
# Created with podman-2.2.1
apiVersion: v1
kind: Pod
metadata:
  creationTimestamp: "2021-03-23T21:16:57Z"
  labels:
    app: test-pod
  name: test-pod
spec:
  containers:
  - command:
    - nginx
    - -g
    - daemon off;
    env:
    - name: PATH
      value: /usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin
    - name: TERM
      value: xterm
    - name: container
      value: podman
    - name: NJS_VERSION
      value: 0.5.2
    - name: PKG_RELEASE
      value: 1~buster
    - name: NGINX_VERSION
      value: 1.19.8
    - name: HOSTNAME
      value: test-pod
    image: docker.io/library/nginx:latest
    name: test-nginx
    ports:
    - containerPort: 80
      hostPort: 8080
      protocol: TCP
    resources: {}
    securityContext:
      allowPrivilegeEscalation: true
      capabilities: {}
      privileged: false
      readOnlyRootFilesystem: false
      seLinuxOptions: {}
    workingDir: /
  - command:
    - /bin/bash
```

```
    env:
    - name: PATH
      value: /usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin
    - name: TERM
      value: xterm
    - name: container
      value: oci
    - name: HOSTNAME
      value: test-pod
    image: registry.access.redhat.com/ubi8:latest
    name: test-ubi8
    resources: {}
    securityContext:
      allowPrivilegeEscalation: true
      capabilities: {}
      privileged: false
      readOnlyRootFilesystem: false
      seLinuxOptions: {}
    workingDir: /
  restartPolicy: Always
status: {}
---
metadata:
  creationTimestamp: null
spec: {}
status:
  loadBalancer: {}
```