A CLOUD GURU

Lesson Guide - Running Your First Podman Container

Every journey has to start somewhere. In this lesson, we're going to jump right into the deep end of the pool and take a look at how to launch and interact with our first container! When you are finished with this lesson you should be able to start, interact with, and stop a container.

Resources

Building, Running, and Managing Linux Containers on Red Hat Enterprise Linux 8 - Red Hat

Getting Started with Podman - podman.io

Instructions

Let's run our first Podman container!

The best way to get experience with containers is to get hands-on and do the things! We're going to jump right into the deep end of the pool and take a look at how to launch and interact with our first container. **Let's go!**

Commands Covered

- `podman ps`: list podman containers
- `podman images`: displays local container images
- `podman search`: search registrie(s) for images
- `podman run`: runs a command in a new container from a given image
- `podman exec`: executes a command in a running container
- `podman stop`: stops a running container
- `podman rm`: removes a container
- `podman rmi`: removes a container image from local storage

Run Your First Podman Container!

Before we launch our first Podman container, let's check and see if there are any containers running. We can use the `-a` or `--all` option to display both running and non-running containers.

As `cloud_user`:

```
podman ps -a
```

As `root`:

```
sudo podman ps -a
```

There should be no containers running.

Let's check our container images:

```
podman images
```

As you can see, we don't have any images yet.

We'd like to run a `httpd-24` container. Let's search for one:

```
podman search httpd-24 | more
```

Let's use the `docker.io/centos/httpd-24-centos8` image.

We're going to run our container in detached mode, with a `tty` to run commands:

```
podman run -dt docker.io/centos/httpd-24-centos8
```

Note that we didn't have to pull the container image to run it. When we ran the container, `podman` pulled the container image for us.

Let's check our containers now:

```
podman ps -a
```

We can see our running container. Congratulations!

Let's check our container images:

```
podman images
```

We see that we now have our `httpd-24-centos:latest` image now.

It's great that we have this running container, but how do we interact with it? Let's try a little experiment.

First, let's see what the operating system is on our server:

```
cat /etc/redhat-release
```

We can see it's RHEL 8.

Now, let's open a `bash` shell in our container:

```
podman exec -it <container_ID> /bin/bash
```

This will give us an interactive session with a `tty`, running a `bash` shell.

Let's see what version of the operating system our container identifies as:

```
cat /etc/redhat-release
```

The container identifies itself as CentOS 8.

If we try accessing the Apache web server running on port `8080` using `curl`:

```
curl http://localhost:8080
```

We see that we get the CentOS Apache server test page, albeit in text.

Let's exit our container shell:

```
exit
```

Back in our server, if we try accessing the Apache web server running on port `8080` in our container using `curl`:

```
curl http://localhost:8080
```

We see that we can't access the Apache web server running in the container. *This is because we haven't published those ports to the host.* We're going to get into that in an upcoming lesson.

Let's check our containers again:

```
podman ps -a
```

We can see our running `httpd-24-centos8:latest` container.

Let's stop our container:

```
podman stop <container_ID>
```

Let's check our containers again:

```
podman ps -a
```

We can see our container is stopped now. We're done with our container.

Let's remove it:

```
podman rm <container_ID>
```

Let's check our containers again:

```
podman ps -a
```

We can see our container has been removed.

Let's check our container images again:

```
podman images
```

Let's also remove our `httpd-24-centos8:latest` image:

```
podman rmi httpd-24-centos8:latest
```

Let's check our container images again:

```
podman images
```

We're all cleaned up.

Summary

Great going, Cloud Guru! We just ran our first Podman container, interacted with it, and cleaned up after ourselves. Just the basics.

Notes

Recording - Environment used: Cloud Playground - Small 2 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!