Taking "Build Once, Run Anywhere" to the Edge

Ygal Blum

Agenda

- Introduction
- What does “Build Once, Run Anywhere” mean?
- What do you need to “run a container”?
- podman kube play
- Podman is daemonless
- Systemd and Podman
- How do I bring it to the Edge?
    - OSBuild steps
- Demo

Introduction

- Ygal Blum
- Principal Software Engineer @RedHat
- Ecosystem Engineering
- Container on Wheels Team

What does “Build Once, Run Anywhere” mean?

- Originally coined by Sun Microsystems
- Ability to write Java code once and run it anywhere
- Expanded by the use of Containers Images
- Package the entire application and all of its dependencies

What do you need to “run a container”?

- Two base elements:
  - Container Image
  - Running Instructions
- The instructions format may vary:
  - Command line arguments
  - Docker-Compose file
  - Kubernetes YAML

podman kube play

- Using “podman kube play” users can reuse K8S YAML file
- “Build Once, Run Anywhere” can be employed for the running instructions

Podman is daemonless

- But, Podman is daemonless
- What will monitor the container?

Systemd and Podman

- Systemd already monitors processes
- If we ran Podman as a damon, we will need Systemd to monitor it
- Let’s have Systemd monitor our containers
- Tools like “podman generate systemd” and soon “Quadlet”facilitate the creation of systemd unit files

How do I bring it to the Edge?

- OSBuild is a tool for composing O/S images
- OSBuild allows embedding container images into the O/S image
- OSBuild allows embedding files such as the K8S YAML and Systemd unit into the O/S image
- OSBuild allows enabling of services in the image
- We can compose an image for an edge device with everything we need already embedded

Red Hat

OSBuild Steps

Embedding the container image

```
- type: org.osbuild.skopeo
   inputs:
     images:
       type: org.osbuild.containers
       origin: org.osbuild.source
       mpp-resolve-images:
         images:
           - source: registry.gitlab.com/centos/automotive/sample-images/demo/auto-apps
             tag: latest
           - source: registry.gitlab.com/centos/automotive/sample-images/demo/vsomeip
             tag: v0.1
   options:
     destination:
       type: containers-storage
       storage-path: /usr/share/containers/storage
```

OSBuild Steps

Embed the Unit and K8S YAML files

```yaml
- type: org.osbuild.copy
   inputs:
     ocp-vsomeip:
       type: org.osbuild.files
       origin: org.osbuild.source
       mpp-embed:
         id: vsomeip.yml
         path: ../files/ocp/vsomeip.yml
     unit-vsomeip:
       type: org.osbuild.files
       origin: org.osbuild.source
       mpp-embed:
         id: vsomeip.service
         path: ../files/ocp/vsomeip.service
   options:
     paths:
     - from:
         mpp-format-string: input://ocp-vsomeip/{embedded['vsomeip.yml']}
       to: tree:///demo/ocp/vsomeip.yml
     - from:
         mpp-format-string: input://unit-vsomeip/{embedded['vsomeip.service']}
       to: tree:///usr/lib/systemd/system/vsomeip.service
```

OSBuild Steps

Enable the Service

```
- type: org.osbuild.systemd
   options:
     enabled_services:
     - radio.service
```

Demo

Questions?

Ygal Blum

ygal@redhat.com

@YgalBlum

ygalblum

linkedin.com/company/red-hat

youtube.com/user/RedHatVideos

facebook.com/redhatinc

twitter.com/RedHat