THINGS I WISH I KNEW ABOUT CONTAINERS SOONER

Alex Juarez
THANK YOU FOR BEING HERE.
WHO IS THIS FOR?

The container converts
The container curious
The container curmudgeons
WHAT DO I HOPE YOU GET FROM THIS
WHAT DO I HOPE YOU GET FROM THIS

- Insight - A new way to think about something
- Perspective - See where someone else might be
- Knowledge - Learn something new
HOW DID WE GET TO THIS TOPIC
WHERE ARE WE GOING

- What is a Container
- Linux Technologies
- Working with Containers
WHAT ARE CONTAINERS?
WHAT ARE CONTAINERS?

Containers are groups of processes running on a Linux system that are isolated from each other.

Podman-in-Action
THEY ARE LIKE VIDEO GAME CARTRIDGES

- Lightweight
- Self-Contained
- Portable
OKAY, BUT WHY?
OKAY, BUT WHY?

Containers as isolated processes, leads to better resource usage and a higher density for a single host.
CONTAINER TERMINOLOGY

Container Image - Base static image file.
example: RHEL Universal Base Image (UBI)

Container Engine - Software (Podman, Docker) for running containers on a single machine.

Container Orchestrators - Software for running containers across multiple machines (Kubernetes, Swarm)
LINUX TECHNOLOGIES

- Namespaces
- CGroups
- SECCOMP
- SELinux
Namespaces are a feature of the Linux kernel that partitions kernel resources such that one set of processes sees one set of resources while another set of processes sees a different set of resources.
Control groups, usually referred to as cgroups, are a Linux kernel feature which allow processes to be organized into hierarchical groups whose usage of various types of resources can then be limited and monitored.
Secure Computing Mode (seccomp) is a kernel feature that allows you to filter system calls to the kernel from a container.
SELINUX

https://stopdisablingselinux.com/
WORKING WITH CONTAINERS

- Container Engines and Runtimes
- Container Storage
- Container Networking
CONTAINER ENGINE

Docker and Podman are popular container engines

- Interface with end-users
- Interface with image registries
- Interface with container runtimes
CONTAINER RUNTIME

Examples of container runtimes are runc and crun

- Manage the container life-cycle.
- Setup the cgroups and namespaces.
- Manage storage and network setup.
- Run and manage the container
CONTAINER VOLUMES

Persistent data and Ephemeral processes
PERSISTENT STORAGE

- Bind Mounts
- Volumes
BIND MOUNTS

Bind mounts allow one part of the filesystem to be mounted in another place in the file system.

```
# mount --bind /var/log/httpd /home/admin/logs
```
BIND MOUNTS IN CONTAINERS

Bind mounts in containers are a way to provide persistent storage by decoupling filesystem storage from the container.

- Provide files for a web server
- Share additional/updated config files
- Test code changes
EXAMPLE BIND MOUNT

```
podman run -d --rm --name ghost-app1
   --network ghost-network --
ip=10.89.0.10 -v
   /var/srv/containers/ghost-content:/var/lib/ghost/content:Z ghost
```
CONTAINER VOLUMES

- Can be re-used amongst containers
- Stored as a file in container storage.
CONTAINER COMMUNICATION
CONTAINER COMMUNICATION

There are a couple of ways communication can happen
CONTAINER COMMUNICATION

There are a couple of ways communication can happen

- App/Host to Container
CONTAINER COMMUNICATION

There are a couple of ways communication can happen

- App/Host to Container
- Container to Container
The easiest way to talk to a container is through any of the exposed ports.

$ podman image inspect docker.io/library/httpd:latest
"Config": {
  "ExposedPorts": {
    "80/tcp": {}
  }
},
Containers can communicate with each other if they are on the same network.

```
$ podman network ls
NETWORK ID    NAME        DRIVER
2a3689189ffd  podman      bridge
$ podman inspect 2a3689189ffd
"Networks": {
    "ghost-network": {
        "EndpointID": "",
        "Gateway": "10.89.0.1",
        "IPAddress": "10.89.0.2",
        "IPPrefixLen": 24,
        "IPv6Gateway": "",
        "GlobalIPv6Address": "",
        "GlobalIPv6PrefixLen": 0,
        "MacAddress": "4e:34:cc:bd:92:b5",
        "NetworkID": "ghost-network"
    }
}
```
CONTAINER ORCHESTARITON
CONTAINER ORCHESTARITON
BUT NOT REALLY...
#!/bin/bash

podman run -d --rm --name ghost-app1 --network ghost-network
podman run -d --rm --name ghost-app2 --network ghost-network
podman run -d --rm -v /root/custom-nginx.conf:/etc/nginx/nginx
IN REVIEW
IN REVIEW

THE THINGS I WISH I KNEW SOONER

1. Containers aren't scary, just a game cartridge
2. CGroups, Namespaces, etc. It's JUST Linux
3. Storage and Networking were really key to learn.
QUESTIONS?
THANK YOU!