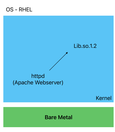
Deploy a Webserver

1. Dedicated Servers
2. Virtualization
3. Cloud
4. Containers (Can be implemented on all 3)

1. Dedicated Server



Cpu, RAM, Storage, nic

PROBLEMS

- RHEL is updated and now the lib is updated.... Lib.so.1.2 to lib.so.1.5
- MySQL ---- lib.so.1.6
- Migration ---- time taking
- Resource Utilisation
- Default Program
- Cannot run 2 different OS on same hardware

2. Virtualisation

Problems

- RHEL updates, and the lib gets updated as well
- Migration
- Hosting two different applications on same OS is not possible
- Default Programs

Cloud

You do not need to maintain any sort of physical hardware

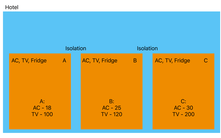
Container

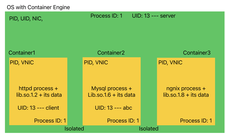
Benefits:

1. Low Hardware Footprint
2. Environment Isolation
3. Quick Deployment
4. Reusability

Container:
A unit of software packaging all the codes and services alongside all dependencies of that particular application

Name Space

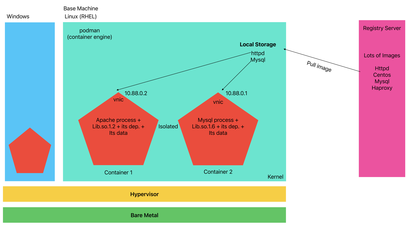




OCI: Open Container Initiative

It is an open governance body that has decided to generalise and make sure that all containers follow a specific format and runtime.

Container Architecture



ISO Image: BaseOS + Many Programs + All Their Data + Their Lib + Their Dependencies + kernel ==== run this image ==== OS

Container Image: BaseOS + One Program + All Its Data + All Its Lib + all its dependencies ==== run this image ==== container

PODMAN

Podman is owned by RedHat

Podman is open source, dameon less, Linux native tool which can be used to manage run, find, build, share etc, application via Containers through the OCI.

Podman is compatible with Kubernetes, preferred for OpenShift.

It is very similar to Docker, you can simply alias it (podman == docker)

Container Engine Alternatives:

Docker
Rocket
Drawbridge

Registry Server

Online repository accessible by anyone
Contains the official as well as user made images

Public:
Registry servers that can be accessed by anyone to pull or push images from or to.

Private:
Registry Servers which are not public, meaning you need to have some kind of authorization to access and to push\pull your images

docker.io/library/nginx:latest

<registry_name>/<name_of_user>/<image_name>:<image_version (tag)>

IMAGES

OS Based Images

We will use these OS Based images to develop out own Daemon based images

Ex. RHEL, Ubuntu, CentOS

Daemon Based Images

They are preconfigured with one program

Ex. Nginx, apache, haproxy

httpd: os base image (centos) + apache program + its lib + its dep. + its data

Change Default Search Registries

vim /etc/containers/registries.conf

RUNNING CONTAINERS

```
podman  run  <image_name:tag>
```

-d: --detach, allows container to run in background
--name: Giving the container a name according to us
-p: To assign Ports Ex. **80:80** [host_port:container_port]
-e: Define Environment Variables

```
podman rm <container_name>
podman stop <container_name>
podman kill <container_name>
```

```
podman  exec is to run commands inside a container
```

-it: Make it Interactive
-t: Provide us a Terminal

EX.

```
podman  exec  -it  <container_name>  <command>
```

If you want to enter the container

```
podman  exec  -it  <container_name>  bash
```

TASK:

Pull an nginx image, and you have to create container with the name "mynginx" and connect it with port 80 of container and port 8080 of host machine, it should run in background and env. "USER=REDHAT"

GITHUB: DTG2468

You have to enter the container and add a line to /htdocs/index.html
The line is "HELLO EVERYONE!"
Verify using Firefox or curl localhost:8080