Containers, Containers, Containers

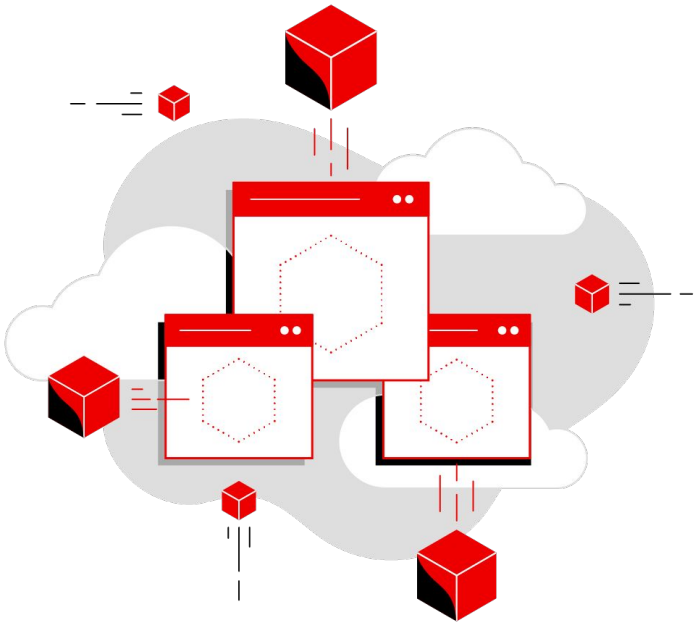
Not Steve Ballmer

Florian Moss
Solution Architect

Red Hat

I want to f**** apologize for this but you have to endure this

...it will get easier at the end!



Red Hat

A brief history of containers

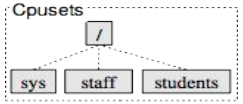
2000:
JAILS ADDED TO FREEBSD

2001:
LINUX -VSERVER PROJECT

2003:
SELINUX ADDED TO LINUX MAINLINE

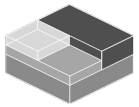
2005:
FULL RELEASE OF SOLARIS ZONES

2006:
PROCESS CONFINEMENT

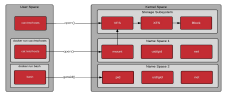


2007:
GPC RENAMED CGROUPS

2008:
LINUX CONTAINER PROJECT (LXC)



2008:
KERNEL & USER NAMESPACES



2013:
DOTCLOUD PYCON LIGHTNING TALK

2013:
DOTCLOUD BECOMES DOCKER

2013:
RED HAT ENTERPRISE LINUX

2014:
GOOGLE KUBERNETES

2015:
RED HAT CONTAINER PLATFORM

2015:
STANDARDS VIA OCI AND CNCF

2000
2005
2010
2015

Red Hat

WHAT ARE CONTAINERS?

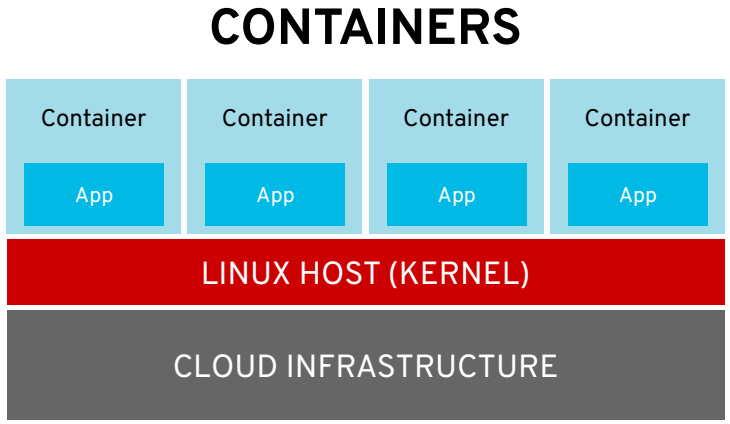
The Internet is wrong

Important corrections

- Containers do not run ON docker.
- Containers are processes - they run on the Linux kernel. Containers are Linux.
- The docker daemon is one of the many user space tools/libraries that talks to the kernel to set up containers

Way of

- Packaging app with all its dependencies
- Securing Isolation of Applications
- Eliminating need for VM Hypervisor
- Having apps portability ⇒ Build once, run everywhere



- Integrated in Linux OS - #theosmatters
- Fully Open Source

Red Hat

WHAT ARE CONTAINERS?

Benefits

DEVELOPERS

- CLOUD-NATIVE APPS
- SIMPLIFY PACKAGING
- SIMPLIFY TESTING
- FREEDOM

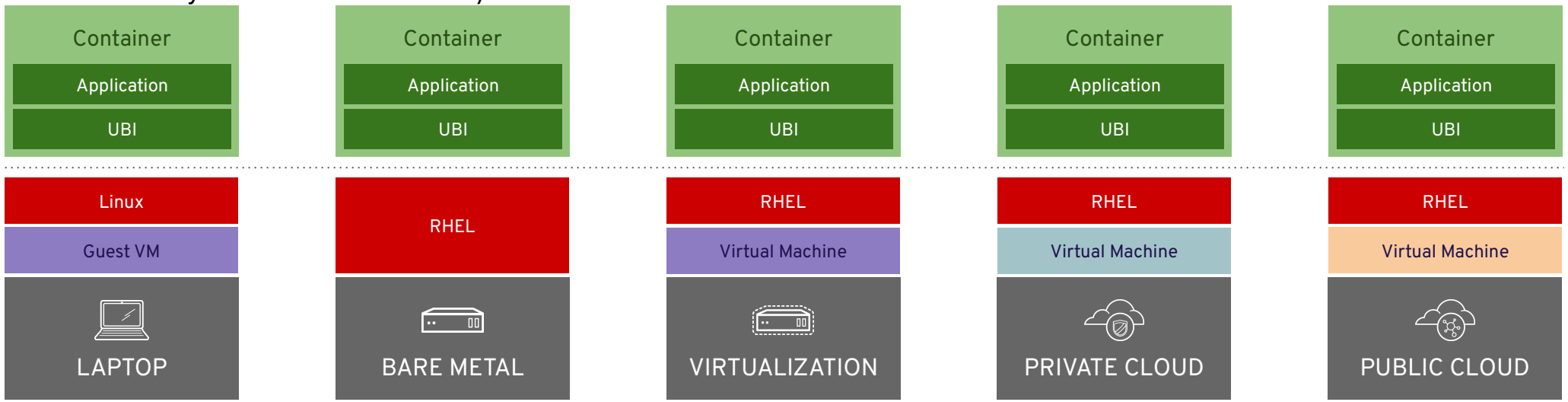
IT OPERATIONS

- CONSISTENT APP DEPLOYS
- AUTOMATED APP DEPLOYS & LC
- IMPROVED APP PERFORMANCE
- MULTI-CLOUD CONSISTENCY

BUSINESS LEADERS

- ENABLE DEVOPS CULTURE
- ENABLE HYBRID CLOUD
- REDUCE VM LICENSING COSTS
- ACCELERATE APP-DEV CYCLES

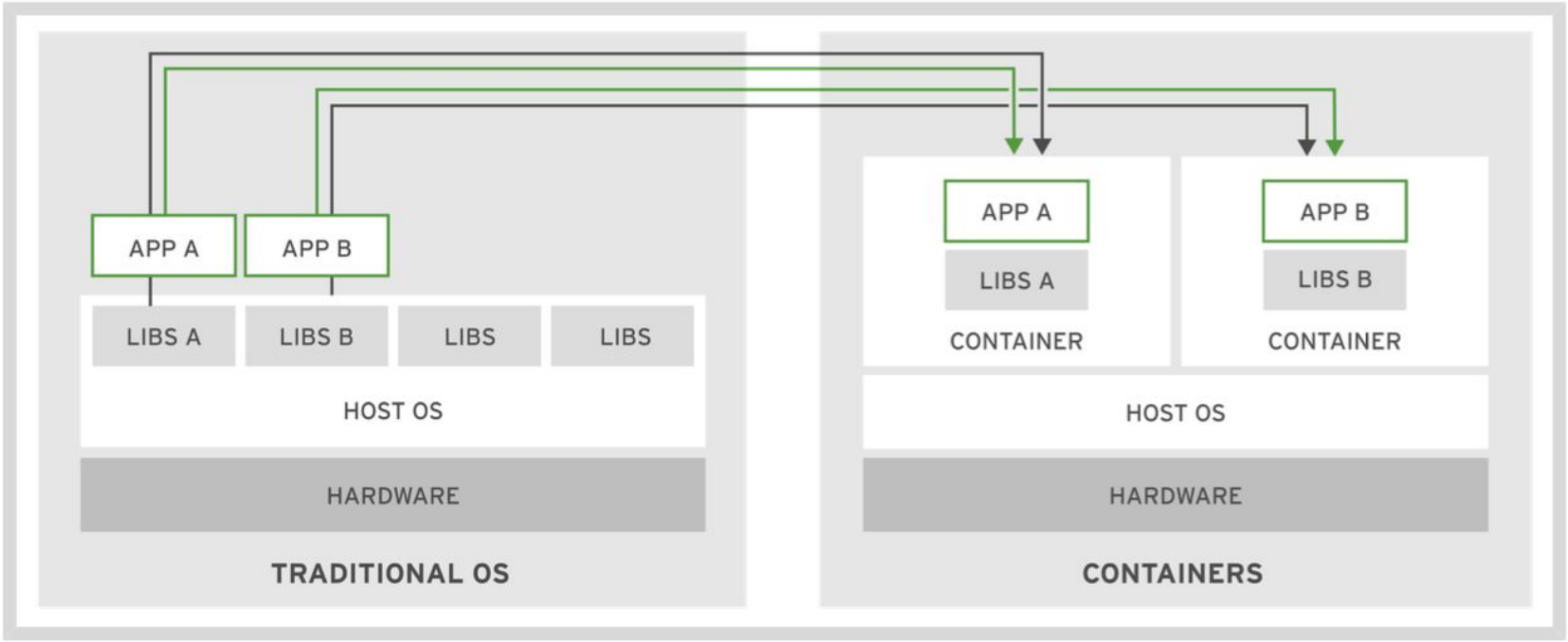
- Apps isolation vs Resources isolation (containers vs vms)
- Portability across all infras - CI / CD



UBI: Red Hat Universal Base Image - Standardize your deps OS

Red Hat

Virtual Machine vs. Containers



Red Hat

Image

....an immutable set of file layers and meta-data
....think about a blueprint/set of instructions
....this is what we “distribute”

```
FROM registry.access.redhat.com/ubi8/ubi:8.1

RUN yum --disableplugin=subscription-manager -y module enable php:7.3 \
  && yum --disableplugin=subscription-manager -y install httpd php \
  && yum --disableplugin=subscription-manager clean all

ADD index.php /var/www/html

RUN sed -i 's/Listen 80/Listen 8080/' /etc/httpd/conf/httpd.conf \
  && sed -i 's/listen.acl_users = apache,nginx/listen.acl_users =/' /etc/php-fpm.d/www.conf \
  && mkdir /run/php-fpm \
  && chgrp -R 0 /var/log/httpd /var/run/httpd /run/php-fpm \
  && chmod -R g=u /var/log/httpd /var/run/httpd /run/php-fpm

EXPOSE 8080
USER 1001
CMD php-fpm & httpd -D FOREGROUND
```

How do I make an image? Buildah.

buildah

BUILDAH

- Now buildah.io
- Builds OCI compliant images
- No daemon - no “docker socket”
- Does not require a running container
- Can use the host’s user’s secrets.
- Single layer, from scratch images are made easy and it ensures limited manifest.
- If needed you can still maintain Dockerfile based workflow

SECURITY FEATURES

Build securely
No daemon
Shrink the attack surface
Fine-grained control of the layers
Run builds isolated
Better secret management

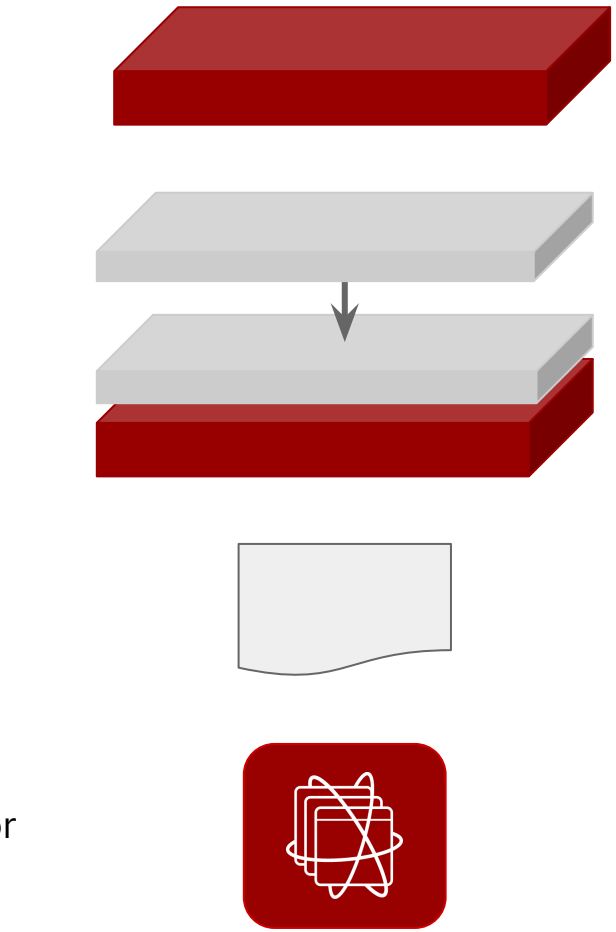
buildah

Start from an existing image or from scratch

Generate new layers and/or run commands on existing layers

Commit storage and generate the image manifest

Deliver image to a local store or remote OCI / docker registry



hands-on to see it in action

https://lab.redhat.com

Red Hat

Container

....an instantiation of an image that is mutable
....if the image is the blueprint, the container is the house
....this is what we are “running”

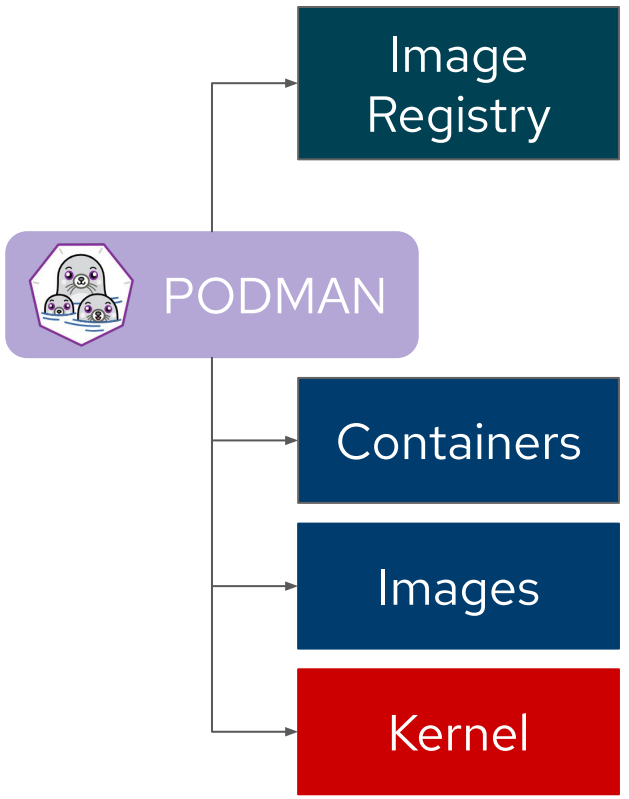
```
[root@41a215b684a8 ~]# podman ps -a
CONTAINER ID  IMAGE                           COMMAND               CREATED         STATUS                     PORTS                   NAMES
90fc2b5be35b  localhost/rhel8-httpd:latest  /usr/sbin/httpd -...  17 seconds ago  Exited (0) 7 seconds ago  0.0.0.0:8081->80/tcp  inspiring_euclid
[root@41a215b684a8 ~]#
```

How do I interact with/run my image? Podman.

PODMAN

The new container CLI

- A daemon-less CLI/API for running, managing, and debugging OCI containers and pods
- Fast and lightweight
- Leverages runC
- Provides a “docker-like” syntax for working with containers
- Standard CNI networking
- Provides systemd integration and advanced namespace isolation
- Shares state with CRI-O and with Buildah!
- Remote management Varlink / Restfull API



SECURITY FEATURES

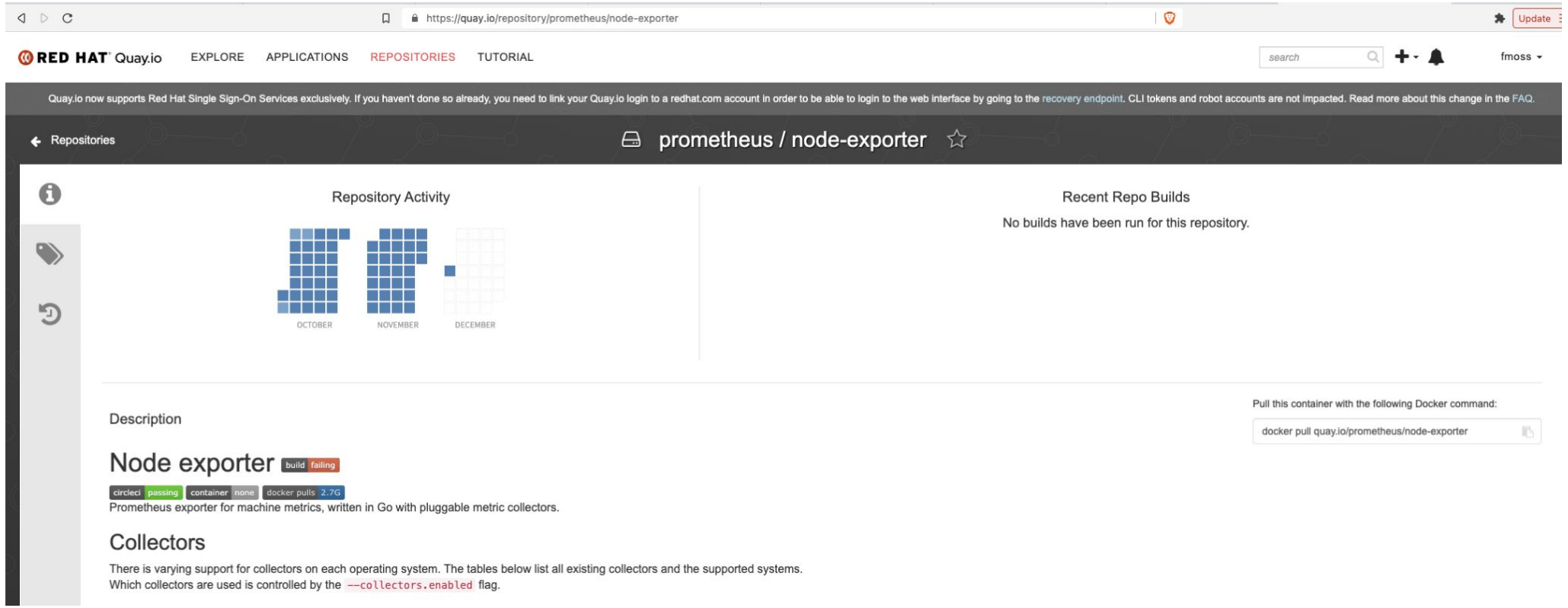
Run and develop securely
No daemon
Run without root
Isolate with user namespaces
Audit who runs what

hands-on to see it in action

https://lab.redhat.com

Image Registry

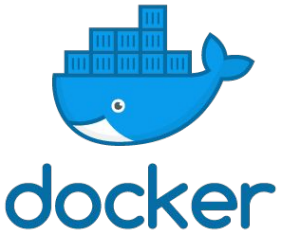
....don't call it container registry, this makes no sense
....stores our images so others can use them



Red Hat

Container Runtime

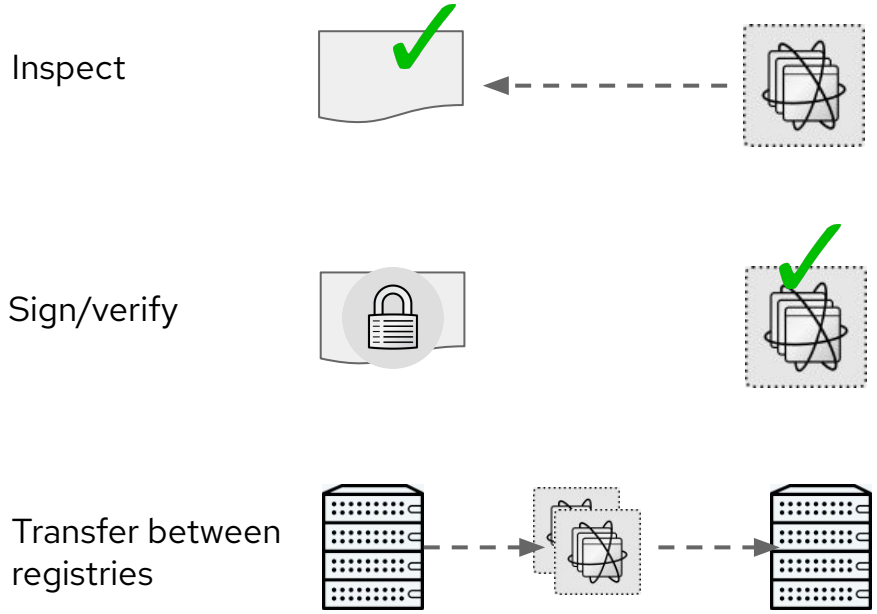
....the part you don't care about
....handles the way your container executes on a machine
....there are different implementations, but you don't care



SKOPEO

Skopeo was originally created for remote inspection of image metadata. It has grown into a comprehensive tool and library to inspect, sign, and transfer images

- Inspect image manifests
- Sign and verify image manifests
- Pull / push images
- Currently the only tool capable of copying images between registries
- Disconnected environments
- Same code base as the /containers/image library which is used by buildah, podman, & CRI-O



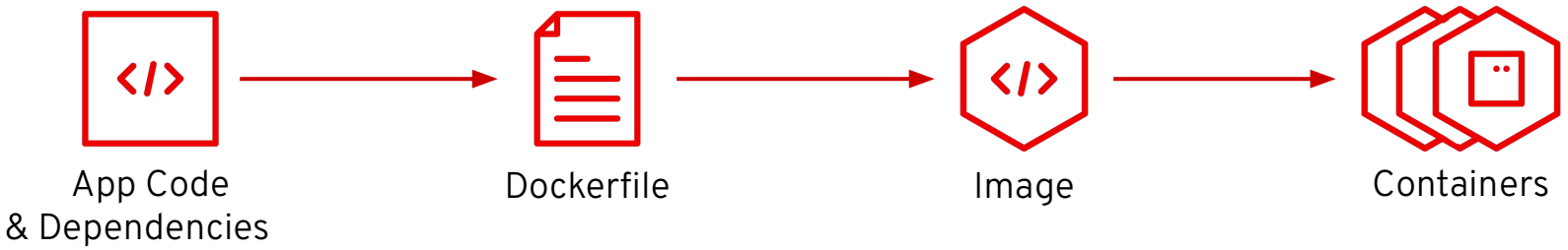
SECURITY FEATURES

Share securely
No daemon
Inspect remote images
No pulling potentially malicious images
Non-root copy. Bridge between registries.

Red Hat

Containerizing Applications

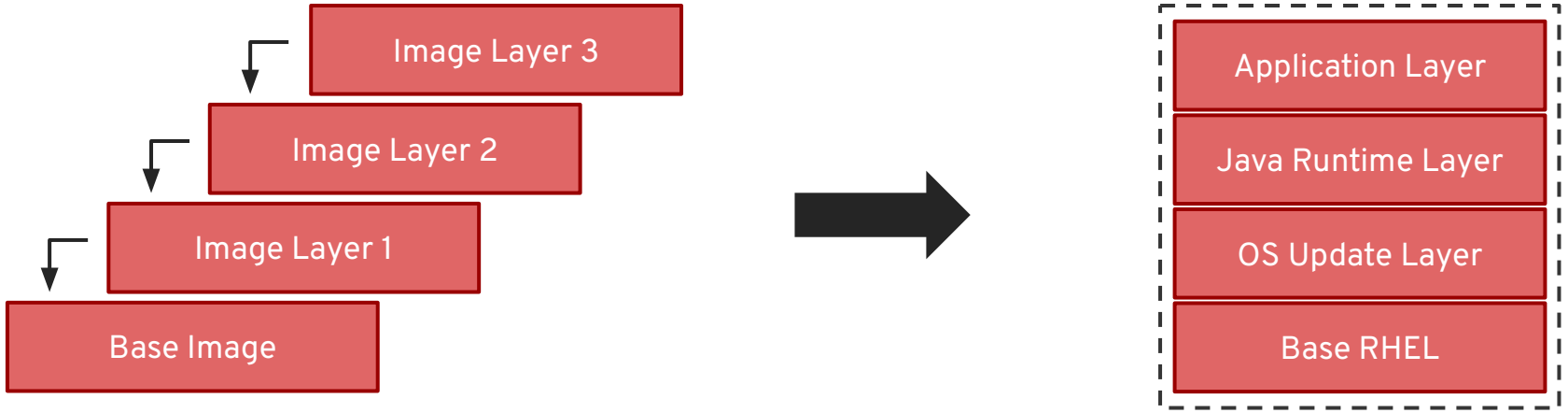
Containerization Process



Red Hat

Containerizing Applications

Anatomy of Container Images



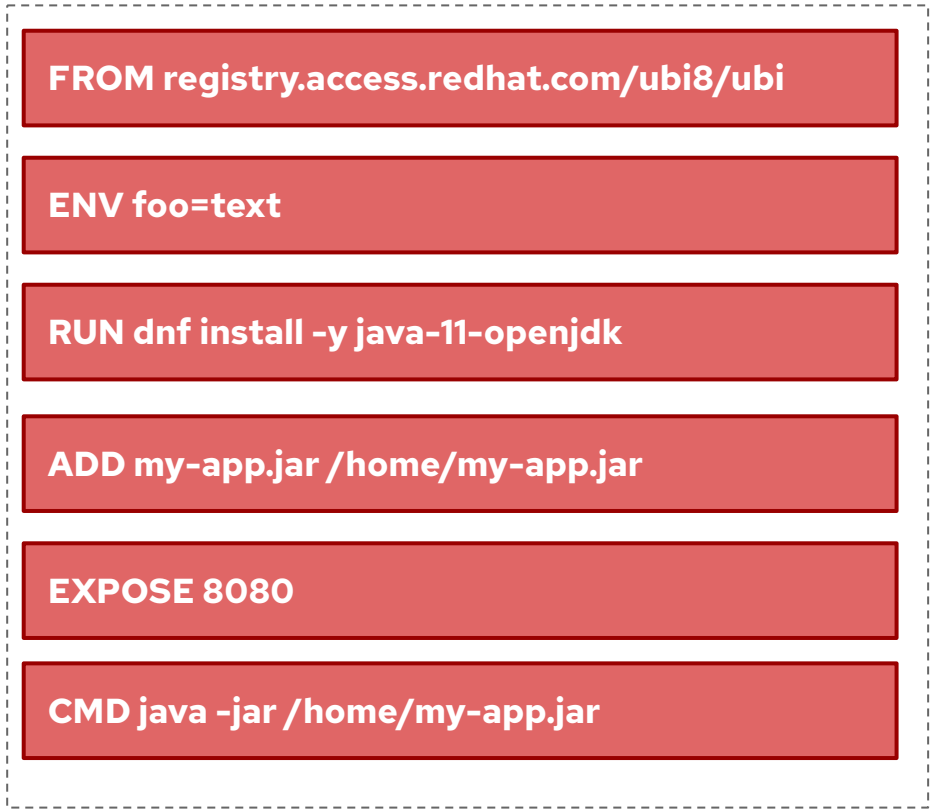
Container Image Layers

Container Image

Red Hat

Containerizing Applications

Anatomy of Dockerfiles



1. Inherit from a base image
2. Parameters as environment variables
3. Install dependencies (tooling from base image)
4. Add your app as a new Layer
5. Expose the port your app will use
6. Run the app

Example for Java app

Red Hat

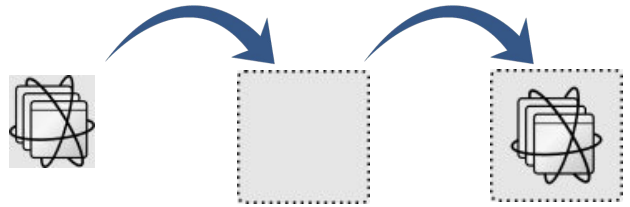
CONTAINERS TOOLS

Providing stability, flexibility and performance with containers and images

- Conform to the OCI image and runtime specifications
- Create, run, and manage, Linux Containers with an enterprise life cycle.
- Daemon-less
- Rootless capable
- OS-native container tooling
- Separation of concerns
- Part of Red Hat Enterprise Linux.
  Available, fully supported at no additional costs
- Check spare slides for labs shortcut

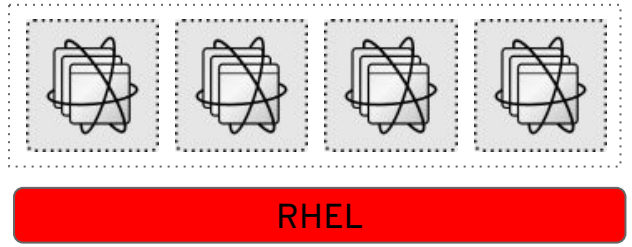
```
#yum module install container-tools
```

buildah



Build OCI/docker Images

podman



run, manage, debug containers

skopeo

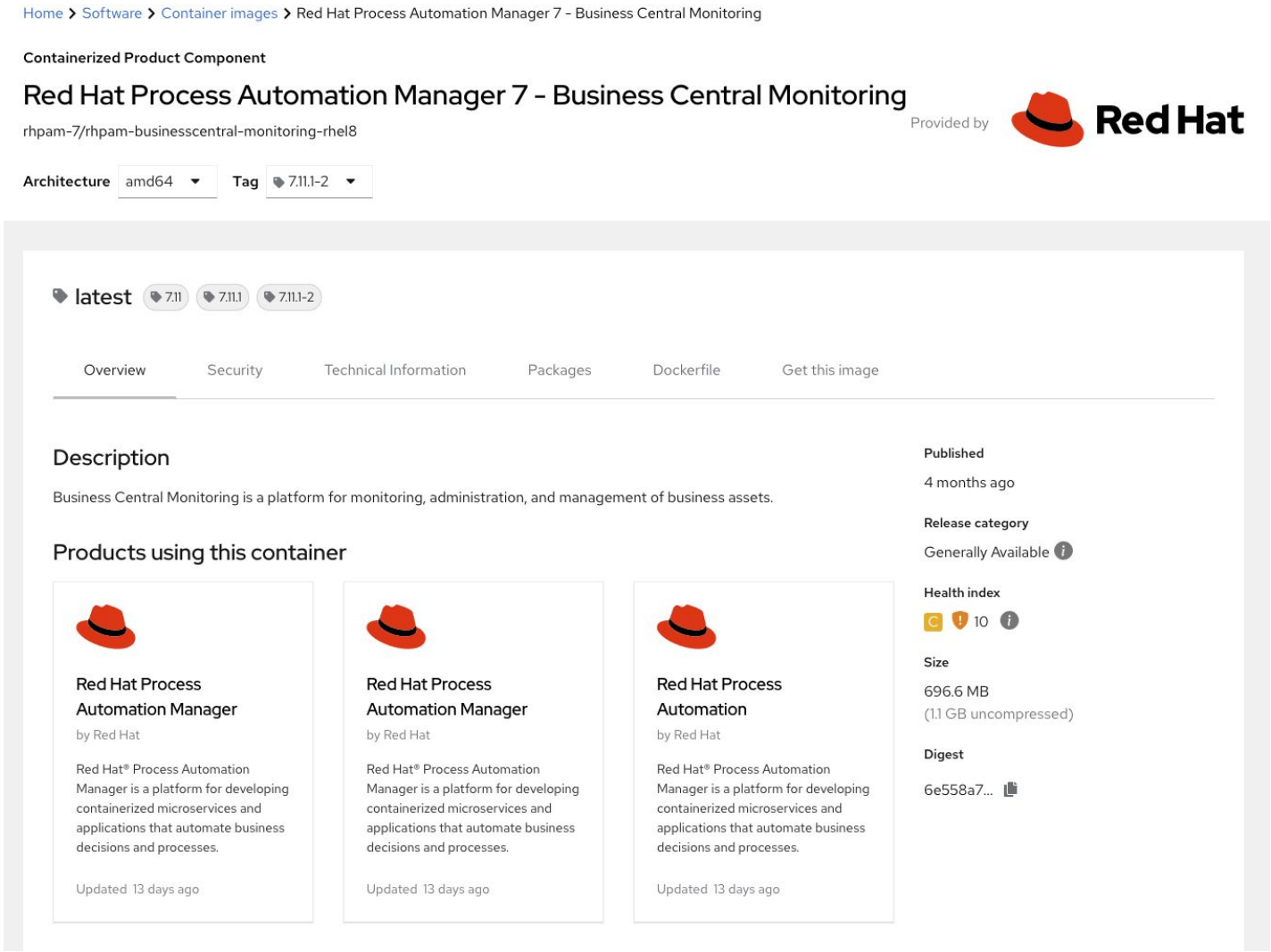
Inspect, copy, & sign Images

OPEN CONTAINER INITIATIVE

Red Hat

Let's Investigate

Example of a containerized application



>Link<

Red Hat