FROM scratch

A Recipe for Trustworthy Container Images

znerol

Cosin 2023

Sysadmin Dilemma

Never touch a running system

- Security (Patch now!)
- Availability (Patch never!)

Sysadmin Dilemma

Trustworthy Supply Chain

Linux Distributions: Crowdsourced Maintenance of Software Packages

Sysadmin Dilemma

Trustworthy Supply Chain

Linux Distributions:
Trustworthiness through
Technology and Procedures

Sysadmin Dilemma

Trustworthy Supply Chain

Linux Distributions:

- Source Code Management
- Reproducible Builds
- Cryptography (signed artifacts)
- Web of Trust
- Project Governance
- ...

Sysadmin Dilemma

Trustworthy Supply Chain

Linux distros help with the sysadmin dilemma:

Running systems with a reasonable level of security and availability is much less effort for admins relying on a distro.

OCI Containers

Yolo Supply Chain

```
docker pull node
docker run -it \
  --publish 8080:8080 \
  node npx --yes -- \
  http-server --port 8080
```

OCI Containers

Yolo Supply Chain

`docker pull:`

No cryptographic signature checked, neither on the manifest nor on artifacts.

OCI Containers

Yolo Supply Chain

Workflows based on docker pull and docker run sacrifice thrustworthyness of the distro supply chain.

OCI Containers

Reasonable Supply Chain

Goals in this Session:

Reproducible base image with tools available in Debian.

OCI Container Image

Anatomy

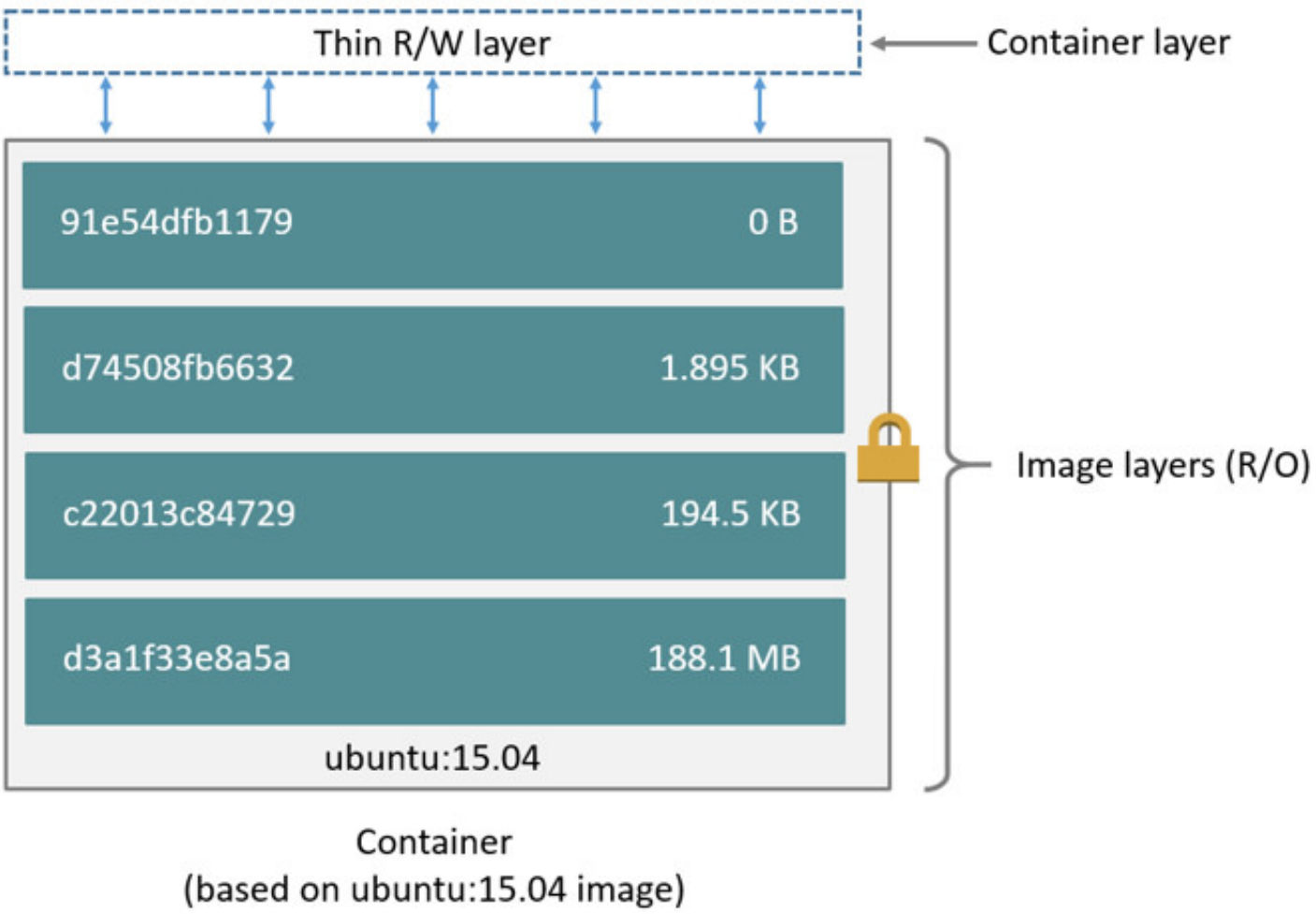

Figure: Image Layers

`https://docs.docker.com/storage/storagedriver/`

OCI Container Image

Specification

`https://github.com/opencontainers/image-spec`

OCI Container Image

Layer

Content Addressable Storage:

```
id = sha256(layer.tar)
```

If there is a way to reproducibly generate a tar file, then there is a way to reproducibly generate an OCI image layer.

OCI Container Image

Containerfile / Dockerfile

```
FROM: ...
```

OCI Container Image

Containerfile / Dockerfile

```
FROM: scratch
```

OCI Container Image

Containerfile / Dockerfile

```
$ truncate --size=1024 \
  empty.tar
$ sha256sum empty.tar
5f70bf18...  empty.tar
```

The `scratch` pseudo image is easily reproducible.

OCI Container Image

Generate rootfs - debootstrap

```
$ sudo debootstrap \
    bookworm dbr/
$ (cd dbr && \
    sudo tar -c . \
  ) > dbr.tar
$ podman import \
    dbr.tar dbr/
$ podman run -it \
    --rm dbr bash
```

OCI Container Image

Generate rootfs - debootstrap issues

- Requires `root`
- Contains generated files.
  E.g., `/etc/machine-id`
- Non reproducible

OCI Container Image

Generate rootfs - mmdebstrap

```
$ mmdebstrap bookworm \
    > mbr.tar
$ podman import \
    mbr.tar mbr/
$ podman run -it \
    --rm mbr bash
```

OCI Container Image

Generate rootfs - mmdebstrap

- Uses `unshare` (no `root` required)
- Removes generated files. E.g., `/etc/machine-id`
- Reproducible if `SOURCE_DATE_EPOCH` is set to some unix timestamp.

OCI Container Image

Generate rootfs - debuerreotype

Official debian images should be reproducible using `debuerreotype`. Regrettably I wasn't able to make that work.

`https://docker.debian.net/`

Running Containers

Fresh Images

Always use **fresh** images and update them regularely.

```
man 1 podman-auto-update
```

Running Containers

Specify a service user

- `-user=www-data` or `-user=33:33`
- Do not use `-user=nobody`

Running Containers

Map user/group namespace

- `--subuidname` or `--uidmap`
- `--subgidname` or `--gidmap`

Running Containers

Hardening

- `-read-only`
- `-cap-drop ALL`
- `-security-opt=no-new-privileges`

Running Containers

Secret management (podman)

```
podman secret create \
  my-secret - < priv.pem
podman run --secret
  my-secret:priv.pem
  httpd
```