Building OCI Images Without Privilege

tycho@tycho.ws, tycander@cisco.com
github.com/tych0


OPEN
CONTAINER
INITIATIVE

CISCO™

Requirements.

Open Container Initiative

Image Format Specification

This specification defines an OCI Image, consisting of a manifest, an image index (optional), a set of filesystem layers, and a configuration.

The goal of this specification is to enable the creation of interoperable tools for building, transporting, and preparing a container image to run.

Table of Contents

- Introduction

$ not #

```
yum -y install ...
./configure ...
sed -i ...
```

What exists?

shiftfs

- In kernel solution to uid-map files based on namespace map
- Author uses it for building container images
- Other interesting applications
- https://lwn.net/ml/linux-fsdevel/1529098514.4048.41.camel@HansenPartnership.com/

“rootless” containers

- umoci has rootless support without user namespaces
- Buildah has (recent) support for user namespaces
- https://github.com/genuinetools/img supports exactly the Docker “API” unprivileged

stacker

How do I use it?

```
first:
    from:
        type: docker
        url: docker://centos:latest
    import:
        - config.json
        - install.sh
    run: |
        mkdir -p /etc/myapp
        cp /stacker/config.json /etc/myapp/
        /stacker/install.sh
```

How do I use it?

```
first:
    from:
        type: docker
        url: docker://centos:latest
    import:
        - config.json
        - install.sh
    run: |
        mkdir -p /etc/myapp
        cp /stacker/config.json /etc/myapp/
        /stacker/install.sh
```

How do I use it?

```
first:
    from:
        type: docker # or tar, oci, etc.
        url: docker://centos:latest
    import:
        - config.json
        - install.sh
    run: |
        mkdir -p /etc/myapp
        cp /stacker/config.json /etc/myapp/
        /stacker/install.sh
```

How do I use it?

```
first:
    from:
        type: docker
        url: docker://centos:latest
    import:
        - config.json
        - install.sh
    run: |
        mkdir -p /etc/myapp
        cp /stacker/config.json /etc/myapp/
        /stacker/install.sh
```

How do I use it?

```
first:
    from:
        type: docker
        url: docker://centos:latest
    import:
        - config.json
        - install.sh
    run: |
        mkdir -p /etc/myapp
        cp /stacker/config.json /etc/myapp/
        /stacker/install.sh
```

How does it work?

- liblxc
- go-lxc
- umoci
- skopeo
  - No API :(
- btrfs
  - Multiple images built from the same source are only extracted once

What does the run environment look like?

- User namespaces
- Host network namespace
- Bind mounted /etc/resolv.conf
- /proc/sys and /proc/sysrq-trigger readonly (proc:mixed in LXC)
- Reasonable devices in /dev (lxc.autodev = 1)
- Bind mounted /sys from host
- /stacker directory mounted r/o for import:s
- Reasonable default $PATH
- Mostly looks like a reasonable system, yum, apt, etc. work fine

odds & ends

--shell-fail

```
$ stacker inspect --oci-dir oci
a
        layer 0: sha256:256b176b... (75 MB)
        layer 1: sha256:276a625d... (156 kB)
Annotations:
  ws.tycho.stacker.stacker_yaml: ...
Image config:
{
  "created": "2018-08-06T16:33:04.379695767-06:00",
  "os": "linux",
  "config": {
    "Env": [
      "PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin"
    ], ...
}
```

updating.

A strategy for container updating

```
A:
  from:
    type: docker
    url: docker://centos:latest
  run:
    - yum install openssl
    - yum install python3
    - git clone https://example.com/A
    - ./A/install
```

```
B:
  from:
    type: docker
    url: docker://centos:latest
  run:
    - yum install openssl
    - yum install python3
    - git clone https://example.com/B
    - ./B/install
```

A strategy for container updating

```
python3:
  from:
    type: docker
    url: docker://centos:latest
  run:
    - yum install python3
ssl:
 from:
    type: docker
    url: docker://centos:latest
  run:
    - yum install openssl
```

```
A:
  from:
    type: docker
    url: docker://centos:latest
  apply:
    - docker://ssl:latest
    - docker://python3:latest
  run:
    - git clone https://example.com/A
    - ./A/install
```

A strategy for container updating

```
python3:
  from:
    type: docker
    url: docker://centos:latest
  run:
    - yum install python3
```

```
8ab6c5e1cb34a35a35... -> python:latest
e05fab2a890d758805... -> centos:latest
39ad9e63562e5d7087...
```

A strategy for container updating

```
python3:
  from:
    type: docker
    url: docker://centos:latest
  run:
    - yum install openssl
```

```
64fabd853e4de75a7e... -> ssl:latest
e05fab2a890d758805... -> centos:latest
39ad9e63562e5d7087...
```

End result

```
e05fab2a890d758805... -> centos:latest
39ad9e63562e5d7087...
```

End result

```
64fabd853e4de75a7e... -> ssl:latest, included verbatim
e05fab2a890d758805... -> centos:latest
39ad9e63562e5d7087...
```

End result

```
8ab6c5e1cb34a35a35... -> python:latest, included verbatim
64fabd853e4de75a7e... -> ssl:latest, included verbatim
e05fab2a890d758805... -> centos:latest
39ad9e63562e5d7087...
```

End result

```
c34553482dda4a28dd... -> diff from app install
8ab6c5e1cb34a35a35... -> python:latest, included verbatim
64fabd853e4de75a7e... -> ssl:latest, included verbatim
e05fab2a890d758805... -> centos:latest
39ad9e63562e5d7087...
```

End result

```
c34553482dda4a28dd...
8ab6c5e1cb34a35a35...
64fabd853e4de75a7e...
e05fab2a890d758805...
39ad9e63562e5d7087...
```

```
A:
  from:
    type: docker
    url: docker://centos:latest
  apply:
    - docker://ssl:latest
    - docker://python3:latest
  run:
    - git clone https://example.com/A
    - ./A/install
```

For posterity

- https://github.com/anuvu/stacker (https://github.com/anuvu/stacker/blob/master/doc/tutorial.md)
- https://github.com/opencontainers/image-spec
- https://github.com/lxc/lxc

Thanks!

We are hiring! Linux, containers, secure boot, etc.

tycho@tycho.ws, tycander@cisco.com

http://github.com/tych0