

OUR DOCKER APP GOT HACKED...

NOW WHAT?

Joel Lathrop

JOEL@DIDACTIC-SECURITY.COM

BSidesRDU | October 2018



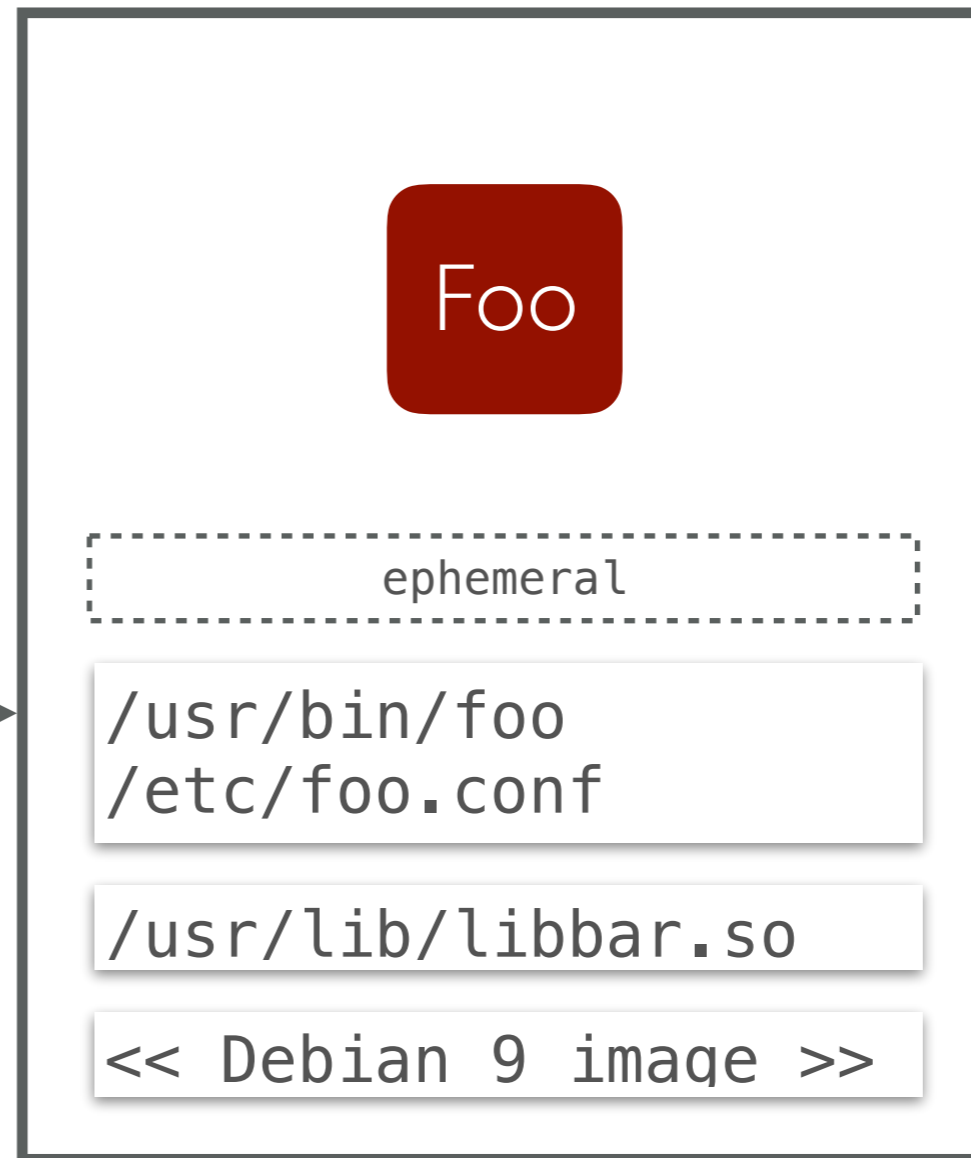
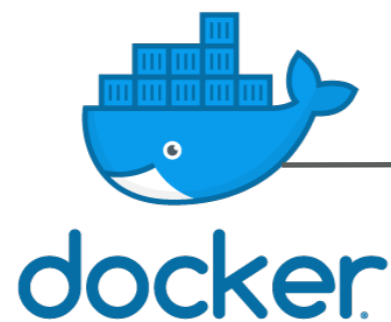
DIDACTIC
SECURITY

GOALS

Know how to forensically capture a Docker container ...

- ▶ from a live system
- ▶ from a cold disk image
 - ▶ ... that used the `overlay2` storage backend
 - ▶ ... that used the `devicemapper` storage backend

ANATOMY



LIVE CAPTURE

Container metadata

```
$ docker inspect foo
```

Container filesystem snapshot

```
$ docker commit foo image-of-foo
```

```
$ docker save image-of-foo | gzip > image-of-foo.tar.gz
```

Log of all terminal I/O

```
$ docker logs foo
```

Container process memory

```
$ docker top foo
```

UID	PID	PPID	C	STIME	TTY	TIME	CMD
root	4802	4784	0	17:53	pts/0	00:00:00	/usr/bin/my-app

```
$ gcore 4802
```

COLD CAPTURE

Where are the goodies at?

`/var/lib/docker`

IMAGE LISTING

image/<backend>/repositories.json

```
{
  "Repositories": {
    "alpine": {
      "alpine:latest": "sha256:196d12...",
      "alpine@sha256:621c2f...": "sha256:196d12..."
    },
    "testing": {
      "testing:latest": "sha256:2ba43e..."
    }
  }
}
```

IMAGE METADATA

`image/<backend>/imagedb/content/sha256/<ID>`

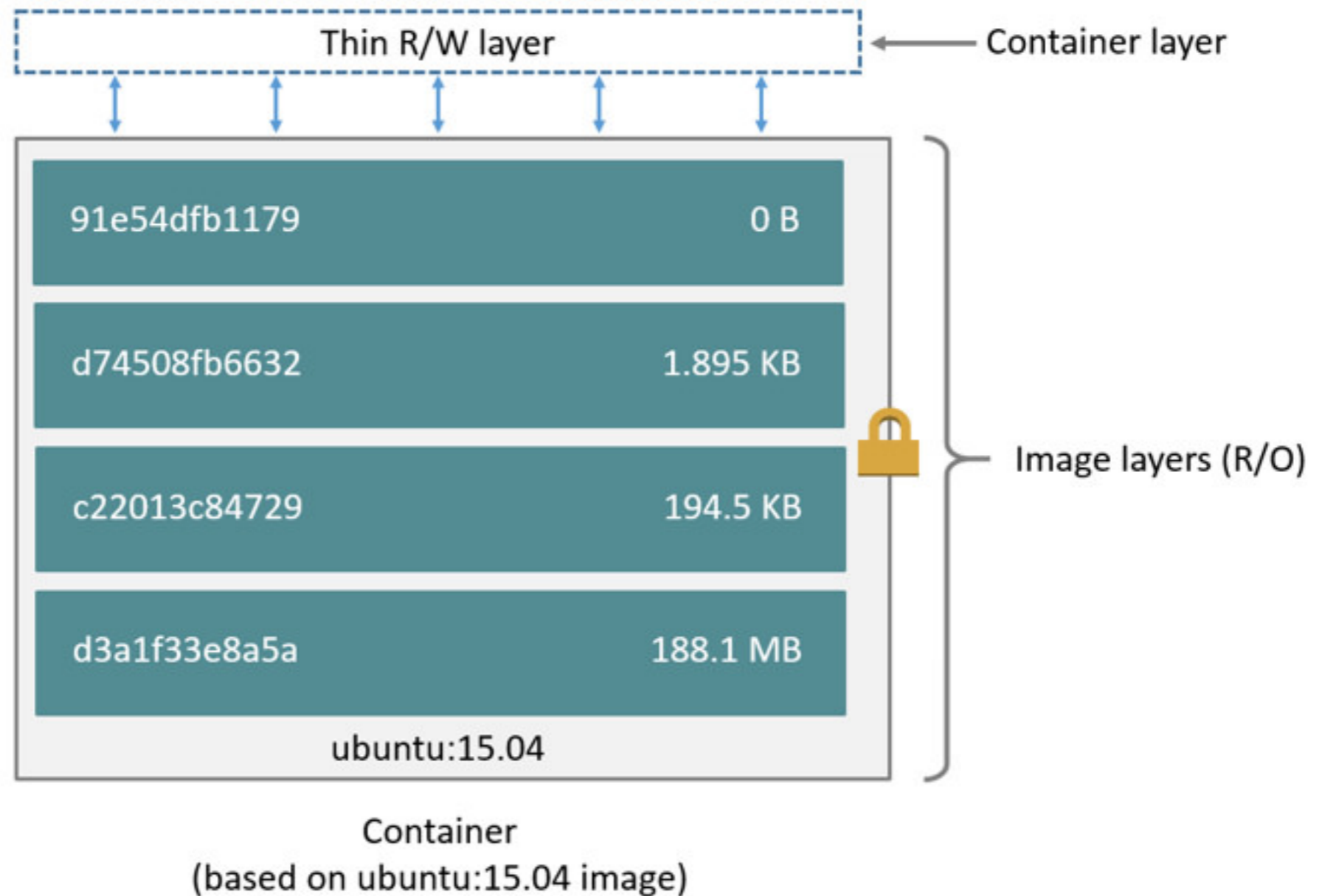
- ▶ A JSON file containing...
 - ▶ Image name
 - ▶ Container entrypoint script
 - ▶ Build history
 - ▶ Creation timestamp
 - ▶ ... and more

CONTAINER METADATA

`containers/<ID>/config.v2.json`

- ▶ Name
- ▶ Image ID
- ▶ Driver
- ▶ State.StartedAt & State.FinishedAt
- ▶ Path & Args
- ▶ Config.Env
- ▶ MountPoints
- ▶ LogPath
- ▶ For JSON logger will be `<ID>-json.log` in same directory.
- ▶ ... and more

WHAT ABOUT DISK CONTENT?



Source: <https://docs.docker.com/storage/storagedriver>

EXAMPLE CONTAINER

Dockerfile

```
FROM alpine

ADD lemons pears /fruit/
ADD entrypoint.sh /

ENTRYPOINT ["/bin/sh", "/entrypoint.sh"]
```

entrypoint.sh

```
echo 'Empires are the best!' > /fruit/apples
echo 'Makes a good sugary drink.' > /fruit/lemons
echo 'This looks interesting...' > /fruit/durian

sync # force modifications to disk

rm /fruit/durian # can't stand the smell
rm /fruit/pears # over-ripen as soon as you turn your back
```

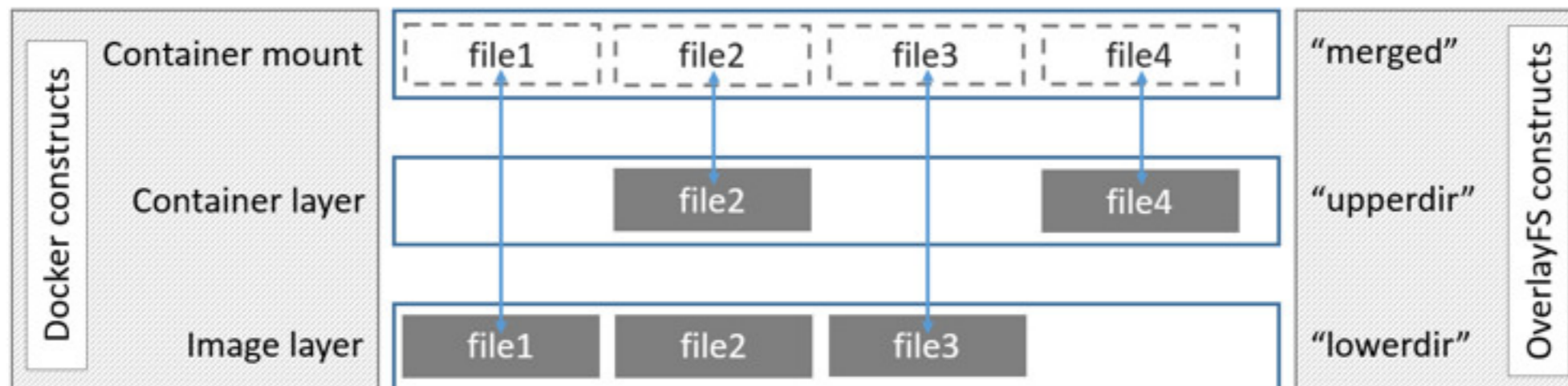
ANTICIPATED LAYERS

apples lemons

lemons pears

< alpine linux >

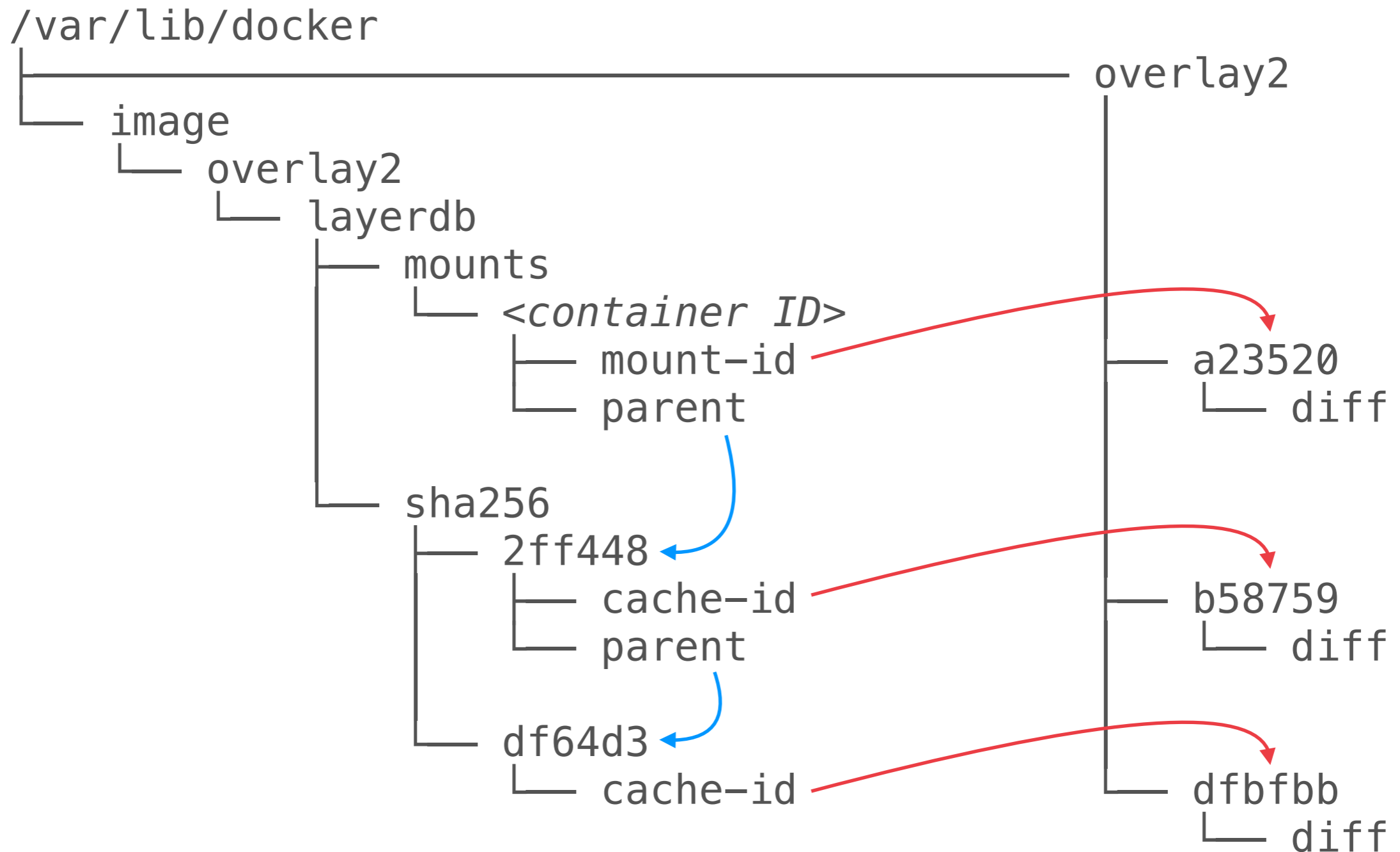
STORAGE BACKEND: OVERLAY2



Source: <https://docs.docker.com/storage/storagedriver/overlayfs-driver>

- ▶ Linux kernel module: OverlayFS
- ▶ Stacked directory tree differences
- ▶ Uses host filesystem
- ▶ Docker's preferred storage backend

FINDING CONTAINER STORAGE



CONTAINER RW LAYER

```
$ mkdir /tmp/view

$ mount -t overlay overlay
      -o ro,lowerdir="./overlay2/a23520/diff:
                          ./overlay2/b58759/diff:
                          ./overlay2/dfbfbb/diff"
      /tmp/view

$ ls -l /tmp/view/fruit
total 20
-rw-r--r-- 1 root root 22 Oct 12 13:45 apples
-rw-r----- 1 root root 27 Oct 12 13:45 lemons
```

TOP IMAGE LAYER

```
$ mkdir /tmp/view  
  
$ mount -t overlay overlay  
      -o ro,lowerdir="/overlay2/b58759/diff:  
                  /overlay2/dfbfbb/diff"  
      /tmp/view  
  
$ ls -l /tmp/view/fruit  
total 20  
-rw-r----- 1 root root  17 Oct 11 20:54 lemons  
-rw-r----- 1 root root  33 Oct 11 20:54 pears
```

RAW OVERLAYFS

```
$ ls -l overlay2/a23520/diff/fruit/  
total 20  
-rw-r--r-- 1 root root 22 Oct 12 13:45 apples  
-rw-r----- 1 root root 27 Oct 12 13:45 lemons  
c----- 1 root root 0, 0 Oct 12 13:45 pears
```

Represents a deleted file

WHAT ABOUT THE DURIAN?

```
$ ifind -f ext4 -n /var/lib/docker/overlay2/a23520/diff/fruit
    /dev/sda1
259987

$ fls /dev/sda1 259987
r/r 259988:    apples
r/r 259989:    lemons
r/l * 259990: durian
c/c 259974:    pears
```

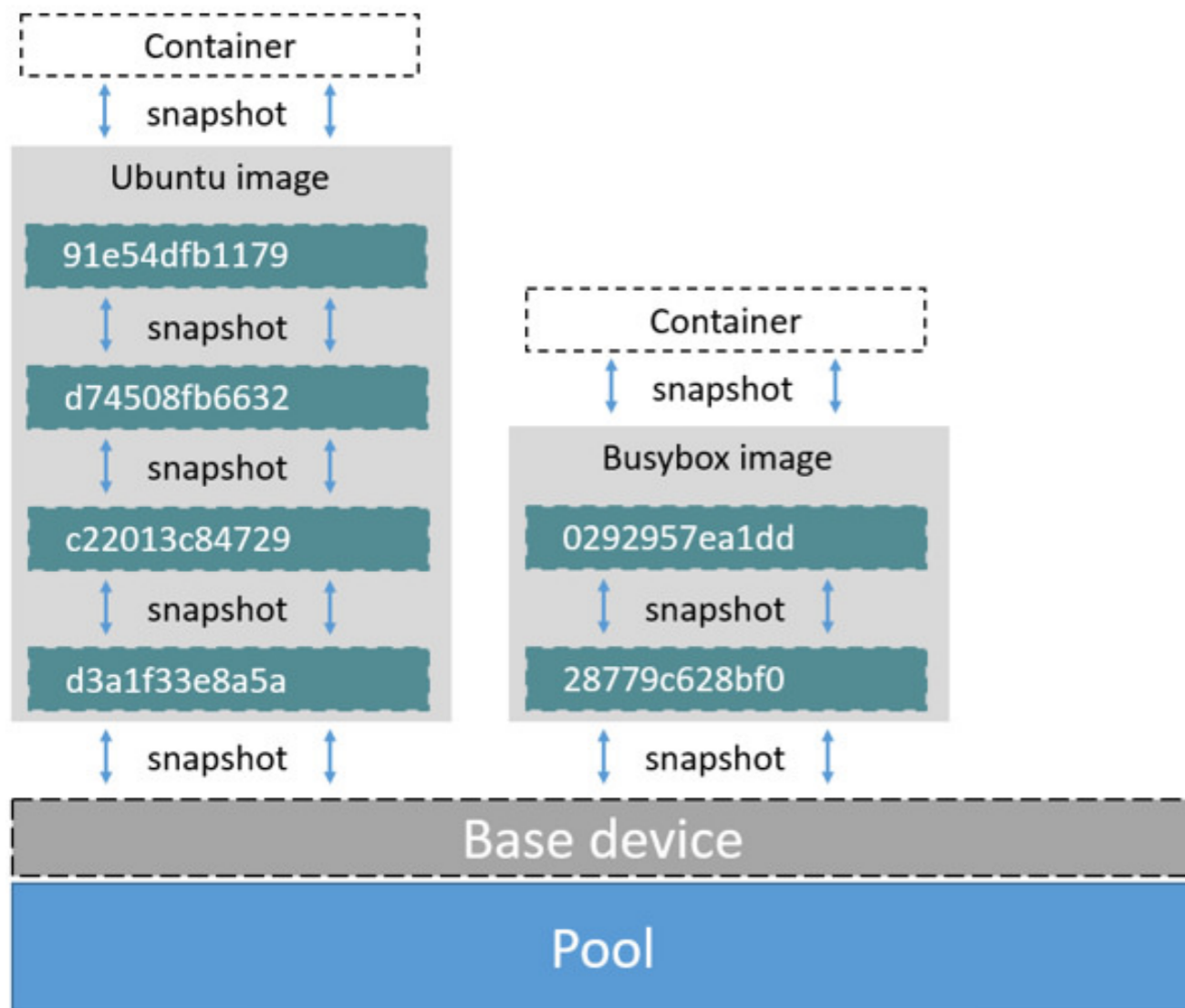
EASY BUTTON!

Romain Gayon automated mounting `overlay2` container storage

<https://github.com/google/docker-explorer>

(Only works for `overlay`, `overlay2`, and `aufs` Docker storage backends.)

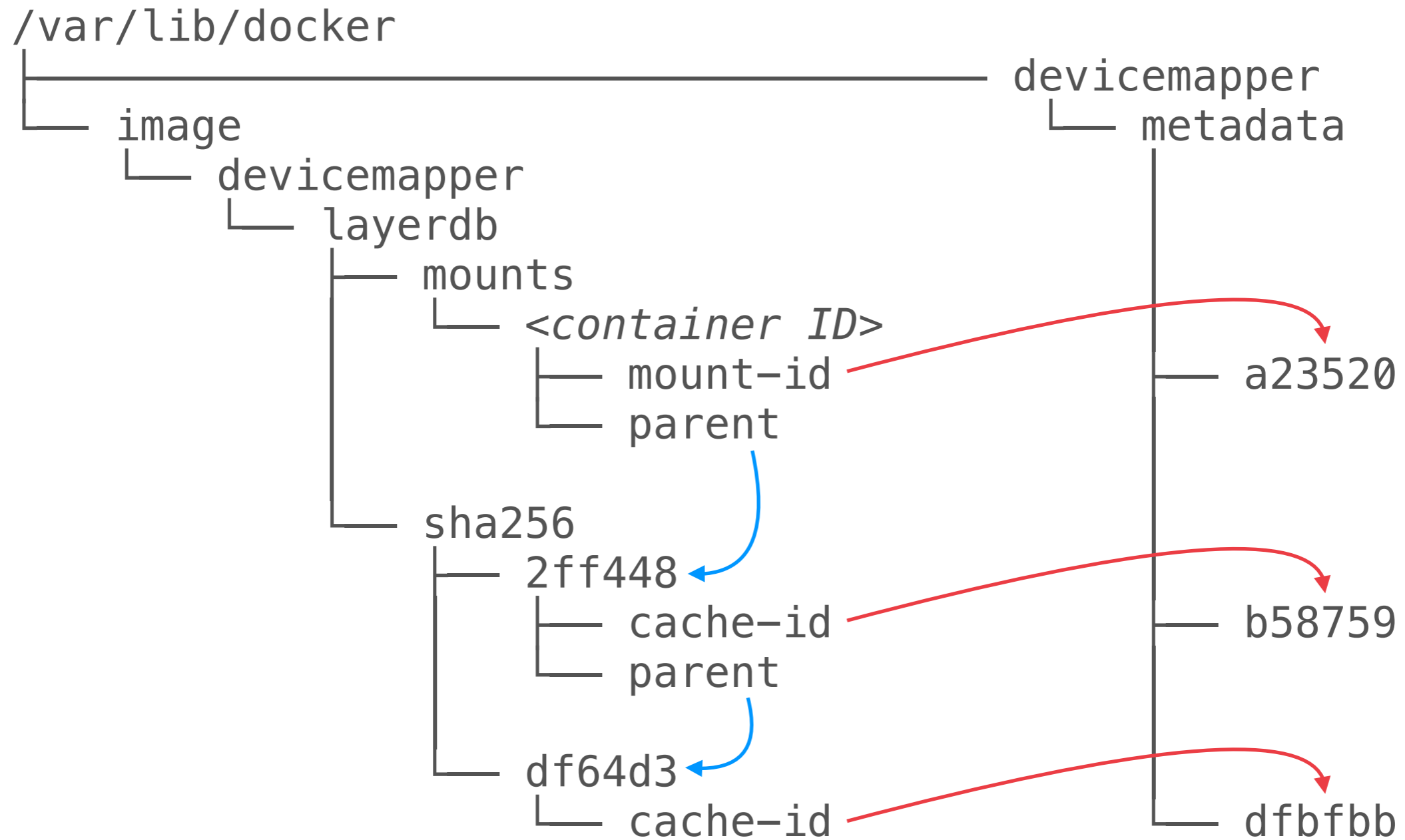
STORAGE BACKEND: DEVICEMAPPER



Source: <https://docs.docker.com/storage/storagedriver/device-mapper-driver>

- ▶ Uses an **LVM storage pool** for production deployments
- ▶ Leverages "thin" device mapper **snapshots**
- ▶ Unlike OverlayFS, the base units are **block devices**, not files
- ▶ You'll likely encounter this if the host is **RedHat** or CentOS

FINDING CONTAINER STORAGE



THIN POOL DEVICES

new device name

```
dmsetup create my-new-device -table  
"0 10240 thin /dev/mapper/pool-device 3"
```

starting sector

size in sectors

pool device

volume ID

ACCESSING A LAYER*

```
$ jq . devicemapper/metadata/a23520
{
  "device_id":      13,
  "size":           10737418240,
  "transaction_id": 16,
  "initialized":    false,
  "deleted":        false
}

$ dmsetup create dk-my-container --table \
    "0 $((10737418240 / 512)) thin /dev/docker/thinpool 13"

$ file -s -L /dev/mapper/dk-my-container
/dev/mapper/dk-base: SGI XFS filesystem data
                    (blkisz 4096, inosz 256, v2 dirs)
```

* You will need to load the LVM volumes first.

MOUNTING THE LAYER

```
$ mkdir /mnt/my-container
```

```
$ mount -o ro,nouuid /dev/mapper/dk-my-container /mnt/my-container
```

```
$ ls -l /mnt/my-container/rootfs/fruit/
```

```
total 8
```

```
-rw-r--r--. 1 root root 22 Oct 13 16:40 apples
```

```
-rw-r-----. 1 root root 27 Oct 13 16:40 lemons
```

FURTHER ANALYSIS

- ▶ Thin pool devices \Rightarrow Virtual disks
 - ▶ Use your regular imaging & analysis tools!
- ▶ Can compare with lower layers
- ▶ `thin_dump` reveals block-level differences

SUMMARY

- ▶ Metadata in JSON files
- ▶ Follow the `layerdb` chain of parent pointers
- ▶ `overlay2`: Layers are directory differences
- ▶ `devicemapper`: Layers are virtual disk snapshots



Questions?

JOEL@DIDACTIC-SECURITY.COM

Bonus!

<https://www.didactic-security.com/resources/docker-forensics-cheatsheet.pdf>