Composefs

An opportunistically sharing verified image filesystem

Alexander Larsson – Red Hat

a usecase

Ostree verified root filesystem

Ostree repo

Image:

```
/app
/app.conf
/subdir/data.bin
```

Repo:

Ostree repo

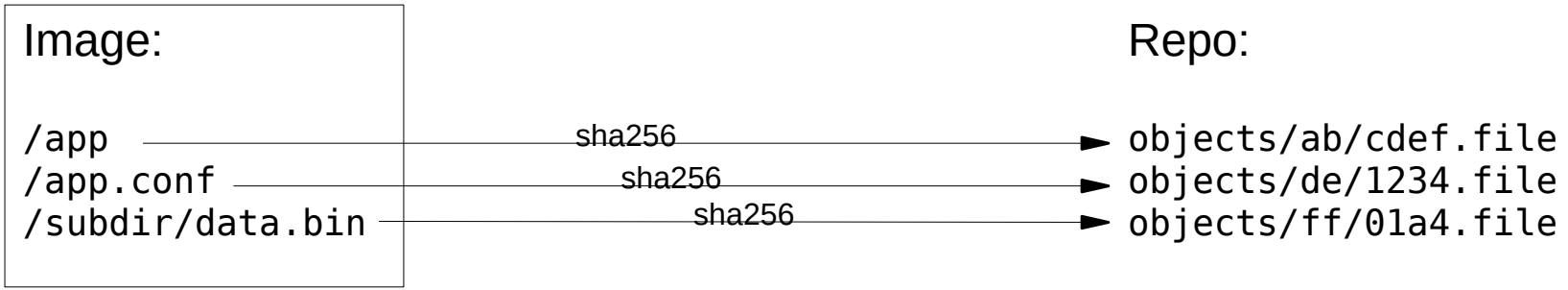


Image:
/app
/app.conf
/subdir/data.bin
sha256
sha256
sha256
Repo:
objects/ab/cdef.file
objects/de/1234.file
objects/ff/01a4.file

Ostree repo

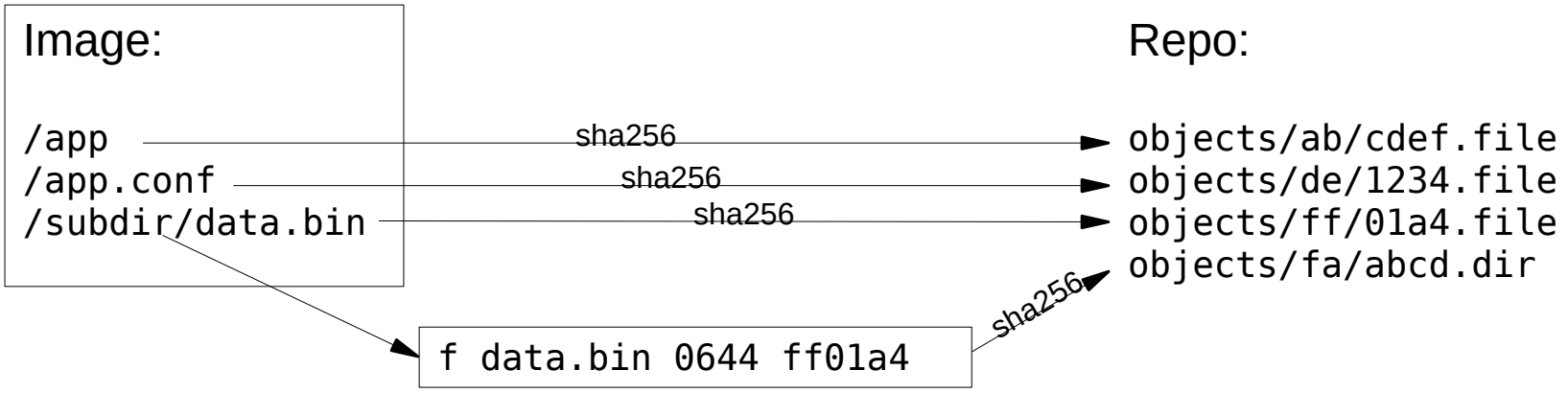


Image:
/app
/app.conf
/subdir/data.bin
sha256
sha256
sha256
Repo:
objects/ab/cdef.file
objects/de/1234.file
objects/ff/01a4.file
objects/fa/abcd.dir
f data.bin 0644 ff01a4
sha256

Ostree repo

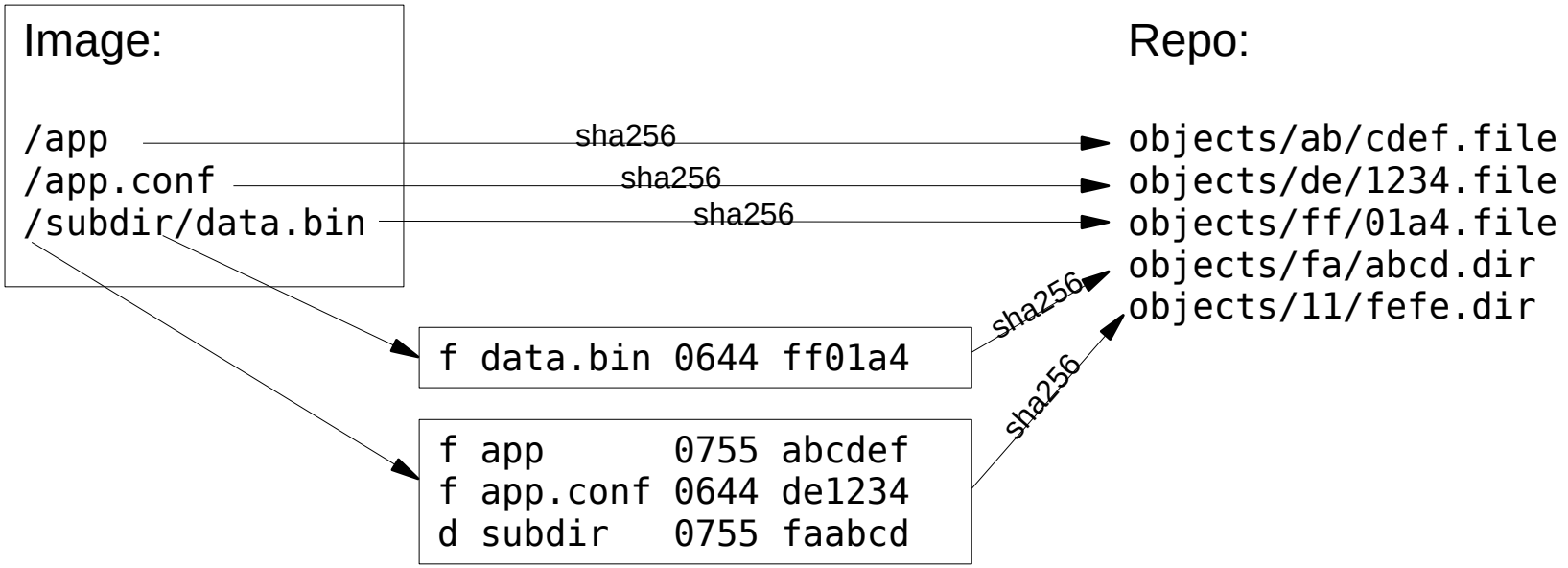


Image:
/app
/app.conf
/subdir/data.bin
sha256
sha256
sha256
Repo:
objects/ab/cdef.file
objects/de/1234.file
objects/ff/01a4.file
objects/fa/abcd.dir
objects/11/fefe.dir
f data.bin 0644 ff01a4
sha256
f app 0755 abcdef
f app.conf 0644 de1234
d subdir 0755 faabcd
sha256

Ostree repo

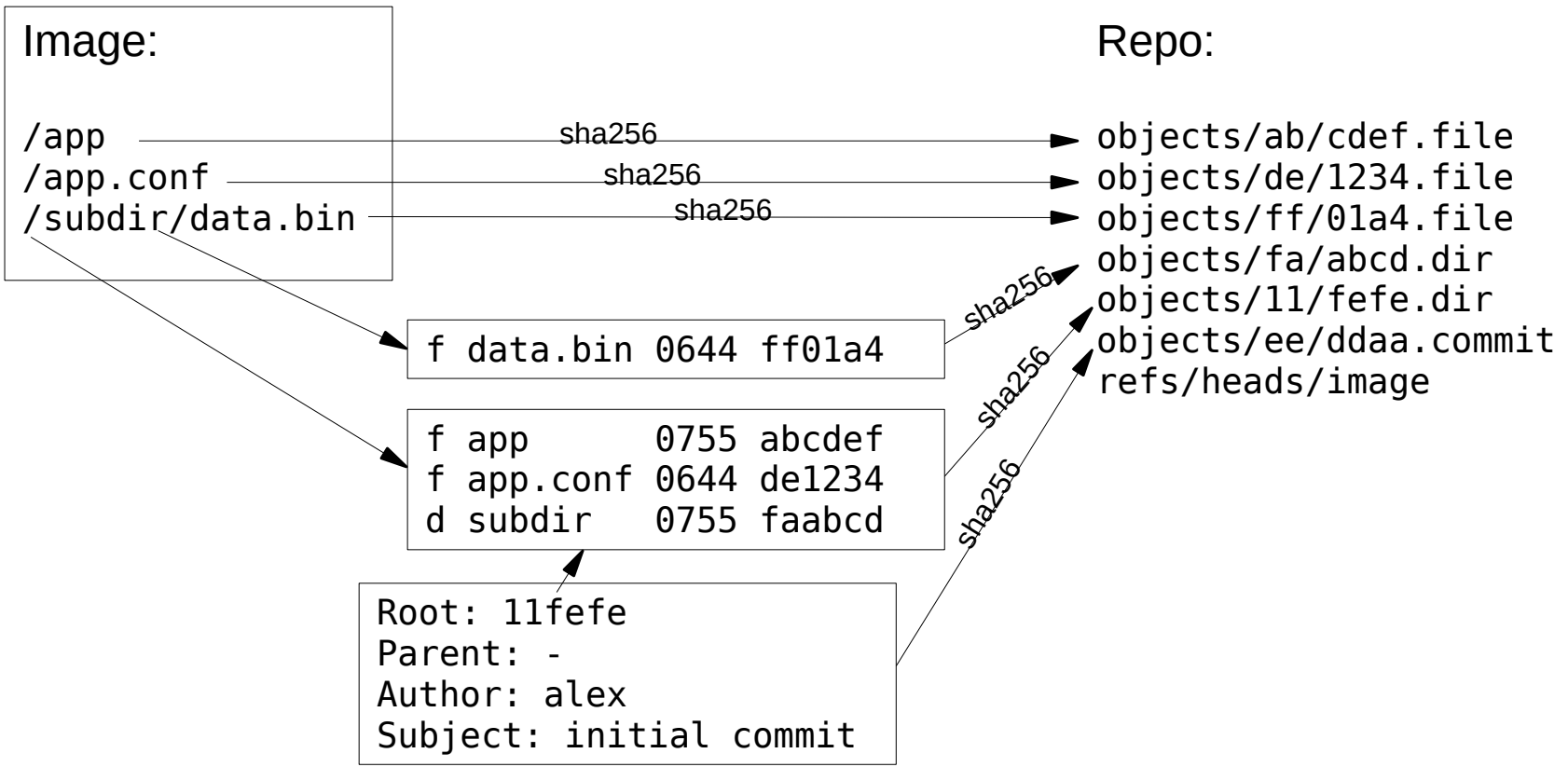


Image:
/app
/app.conf
/subdir/data.bin
sha256
sha256
sha256
Repo:
objects/ab/cdef.file
objects/de/1234.file
objects/ff/01a4.file
objects/fa/abcd.dir
objects/11/fefe.dir
objects/ee/ddaa.commit
refs/heads/image
f data.bin 0644 ff01a4
f app 0755 abcdef
f app.conf 0644 de1234
d subdir 0755 faabcd
Root: 11fefe
Parent: -
Author: alex
Subject: initial commit
sha256
sha256
sha256

Ostree repo

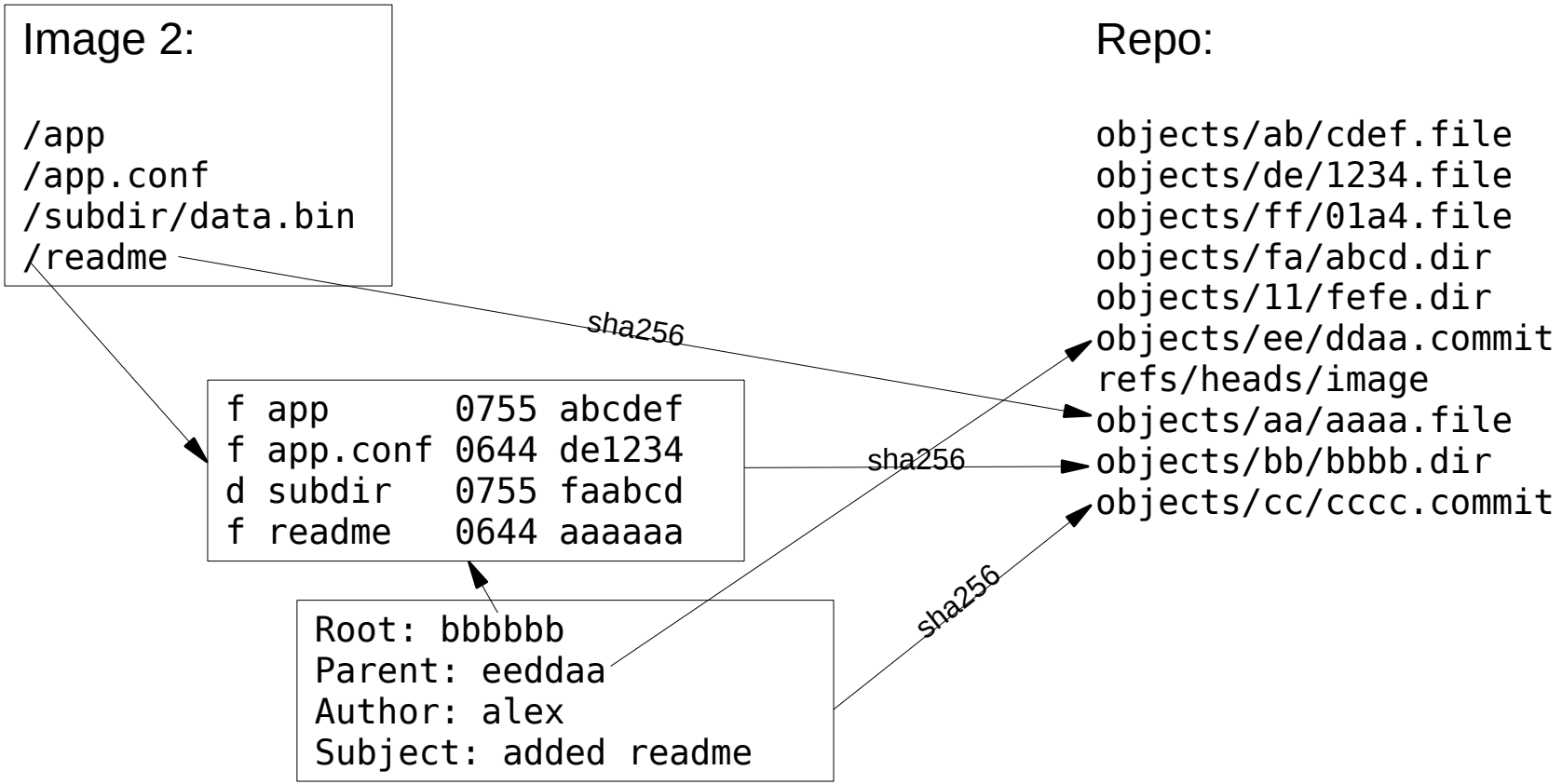


Image 2:
/app
/app.conf
/subdir/data.bin
/readme
sha256
f app 0755 abcdef
f app.conf 0644 de1234
d subdir 0755 faabcd
f readme 0644 aaaaaa
sha256
Root: bbbbbb
Parent: eeddaa
Author: alex
Subject: added readme
sha256
Repo:
objects/ab/cdef.file
objects/de/1234.file
objects/ff/01a4.file
objects/fa/abcd.dir
objects/11/fefe.dir
objects/ee/ddaa.commit
refs/heads/image
objects/aa/aaaa.file
objects/bb/bbbb.dir
objects/cc/cccc.commit

Using ostree repo

- Download new files to /ostree/repo
- Create deploy directory structure:
  - /ostree/$commit/root
- Files are hardlinks to repo objects
- Bind mount root from deploy directory
  - Done by initrd
  - $commit specified on kernel commandline

Ostree advantages

- Efficient downloads of updates
- Efficient storage of multiple versions
- Can store unlimited nr of versions
- Atomic updates and rollbacks
- Self-verifiable format

Runtime verification

- Verification happens at download and deploy
- What about runtime verification?
- Options
  - Dm-verity
    - Block level only
  - Fs-verity
    - Only applies to individual file, not structure

Introducing composefs

```
# ls -ld image/**
drwxr-xr-x. 3 root root 4096 Jan 31 09:04 image/
-rwxr-xr-x. 1 root root   28 Jan 31 09:04 image/app
-rw-r--r--. 1 root root   10 Jan 31 09:04 image/app.conf
drwxr-xr-x. 2 root root 4096 Jan 31 09:04 image/subdir
-rw-r--r--. 1 root root    4 Jan 31 09:04 image/subdir/data.bin

# mkcomposefs --digest-store=objects image image.cfs

# ls -l image.cfs objects/*/*
-rw-r--r--. 1 root root 4229 Jan 31 09:05 image.cfs
-rw-r--r--. 1 root root   10 Jan 31 09:05 objects/bb/eb6146ae4c3…
-rw-r--r--. 1 root root   28 Jan 31 09:05 objects/cf/a32da1f6770…
-rw-r--r--. 1 root root    4 Jan 31 09:05 objects/df/2a46acc498d…

# cat objects/cf/a32da1f6770…
#!/usr/bin/bash
echo HELLO!
```

Introducing composefs (cont)

```
# mount -t composefs -o basedir=objects image.cfs mnt

# ls -ld mnt/**
drwxr-xr-x. 3 root root 4096 Jan 31 09:04 mnt/
-rwxr-xr-x. 1 root root   28 Jan 31 09:04 mnt/app
-rw-r--r--. 1 root root   10 Jan 31 09:04 mnt/app.conf
drwxr-xr-x. 2 root root 4096 Jan 31 09:04 mnt/subdir
-rw-r--r--. 1 root root    4 Jan 31 09:04 mnt/subdir/data.bin

# cat mnt/app
#!/usr/bin/bash
echo HELLO!
```

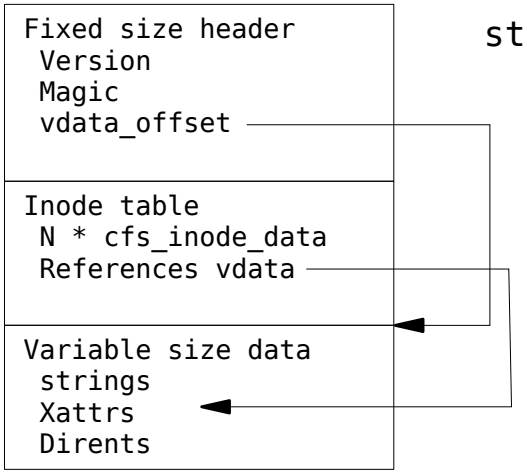
Composefs w/ fs-verity

```
# fsverity enable image.cfs
# fsverity digest --compact image.cfs
52ec887a5221e6d72…

# mount -t composefs -o basedir=objects,digest=52ec887a5221e6d72…
image.cfs mnt
# cat mnt/app
#!/usr/bin/bash
echo HELLO!

# rm objects/cf/a32da1f67701068…
# echo FAKED > objects/cf/a32da1f67701068…
# fsverity enable objects/cf/a32da1f67701068…
# cat mnt/app
WARNING: composefs backing file 'a32da1f67701068…' has the wrong fs-
verity digest
cat: mnt/app: Input/output error
```

Composefs file format




```
struct cfs_inode_data {
        __le32 st_mode;
        __le32 st_nlink;
        __le32 st_uid;
        __le32 st_gid;
        __le32 st_rdev;
        __le64 st_size;
        __le64 st_mtim_sec;
        __le32 st_mtim_nsec;
        __le64 st_ctim_sec;
        __le32 st_ctim_nsec;
        struct cfs_vdata variable_data;
        struct cfs_vdata xattrs;
        struct cfs_vdata digest;
};
```

```
struct cfs_dirent {
        __le32 inode_num;
        __le32 name_offset;
        u8 name_len;
        u8 d_type;
        u16 _padding;
};
```

```
struct cfs_dir_header {
        __le32 n_dirents;
        struct cfs_dirent dirents[];
};
```

```
struct cfs_vdata {
        __le64 off;
        __le32 len;
};
```

Using composefs w/ ostree

- Libcomposefs
  - Allows making descriptor directly
  - Can reuse existing repo
  - 100% reproducible images
- Embedd digest in signed commit
- Recreate image during pull, match digest
- Use composefs mount as rootfs

Other usecase: OCI container

- Use new eStargz tar format
  - 100% compatible with OCI
  - Index allows partial downloads
  - Combine with composefs
- Share files
  - On disk
  - In page cache
- Combines lower layers
  - Less negative lookups

Alternative: Reuse Overlayfs

- Existing features:
  - overlay.redirect
  - overlay.metacopy
- Two lower layers:
  - Object directory
  - Read-only filesystem (erofs, squashfs)
    - containing structure
    - Whiteouts of object directory
    - Files redirect to objects
    - Use dm-verity to protect read-only fs
- Issues:
  - Needs overlay.fs-verity features
  - More complex to use
  - Worse performance

Performance comparison

Composefs:

```
Benchmark 1: ls -lR mnt, warm caches
  Time (mean ± σ):     390.1 ms ±   3.7 ms    [User: 140.9 ms, System: 247.1 ms]
  Range (min … max):   381.5 ms … 393.9 ms    10 runs

Benchmark 2: ls -lR mnt, cold caches
  Time (mean ± σ):     701.0 ms ±  21.9 ms    [User: 153.6 ms, System: 373.3 ms]
  Range (min … max):   662.3 ms … 725.3 ms    10 runs
```

Overlayfs + erofs (with sparse files):

```
Benchmark 1: ls -lR mnt-ovl, warm caches
  Time (mean ± σ):     523.9 ms ±   4.2 ms    [User: 150.4 ms, System: 369.1 ms]
  Range (min … max):   517.4 ms … 530.6 ms    10 runs

Benchmark 1: ls -lR mnt-ovl, cold caches
  Time (mean ± σ):      1.776 s ±  0.037 s    [User: 0.184 s, System: 1.008 s]
  Range (min … max):    1.694 s …  1.815 s    10 runs
```

Questions

https://github.com/containers/composefs