fs-verity

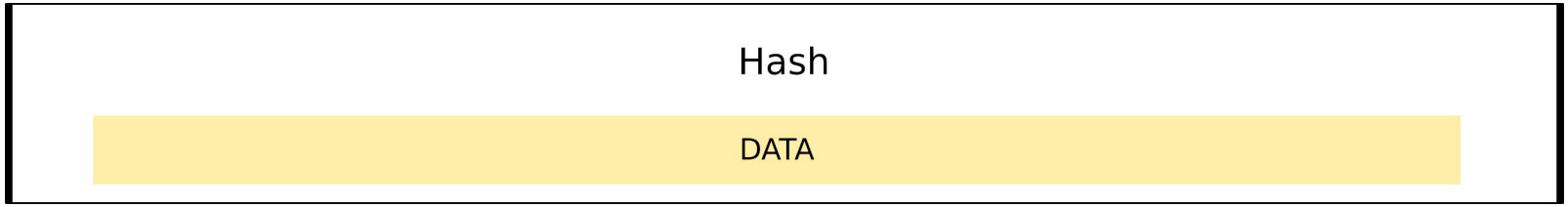
Efficiently Measuring File Contents

Mike Halcrow and Eric Biggers / August 27

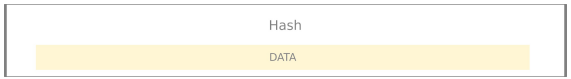
Agenda

- Taking measurements
- dm-verity
- Integrity and Authenticity in the File System
- fs-verity
- fs-verity use cases, e.g. Integrity Measurement Architecture (IMA)

Taking Measurements

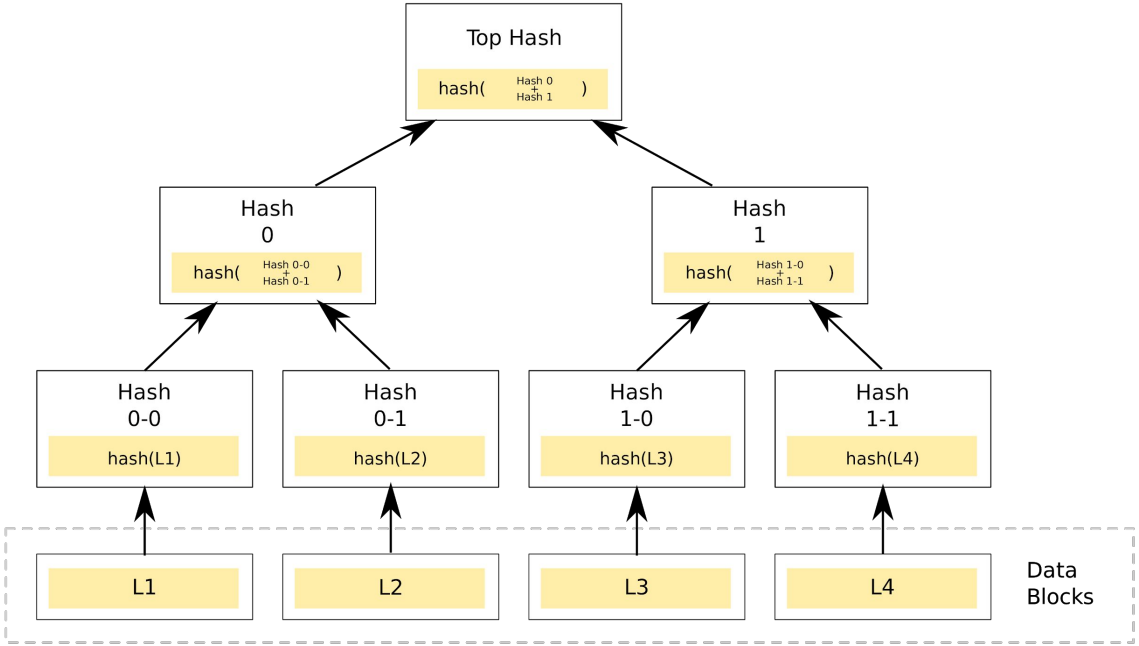


Taking Measurements

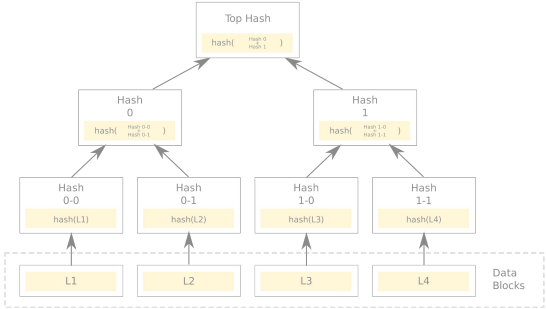


- **Entire object** measured and validated prior to further action.
- Large objects incur **significant latency** on initial access.
- Trade-off: No revalidation on paging back in.
    - Malicious data source (file server or **disk/controller firmware**).
        - Firmware attacks: EquationDrug, GrayFish.

Taking Measurements



Taking Measurements



- Authenticated dictionary structures enable **partial measurements**

  while ensuring **comprehensive validation**.

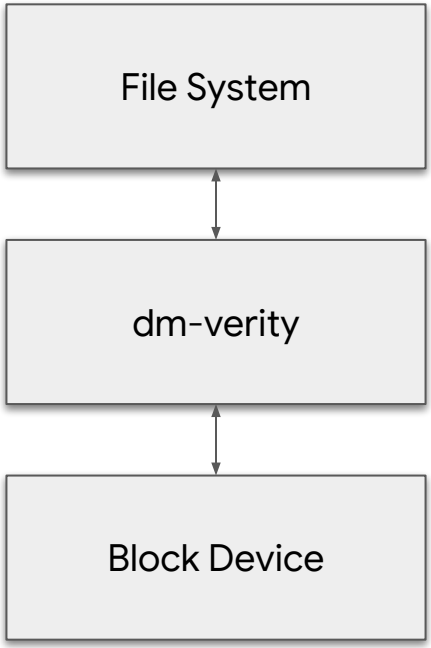
- **Log(Object Size)** latency to start reading on first access.

- Trade-off: I/O Errors possible while processing is in-flight.

dm-verity


Your device software can't be checked for corruption. Please lock the bootloader.
Visit this link on another device:
g.co/ABH

dm-verity



dm-verity

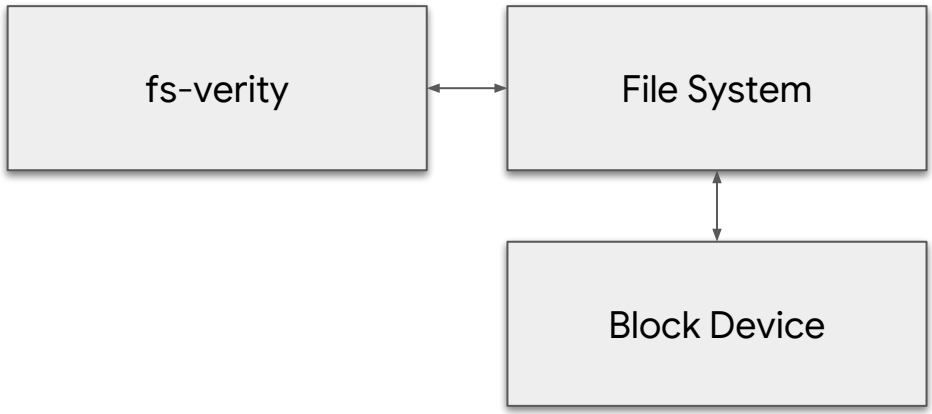
- Full Disk: Protects **all file system file content** and **metadata**.
- Incremental updates require **regenerating the entire auth tree**.
  - Logistics would require **packaging together system image updates**.
  - **Intractable complexity** when dealing with the Android partner ecosystem.

Source: Lorem ipsum dolor sit amet, consectetur adipiscing elit. Duis non erat sem

Integrity and Authenticity in the File System

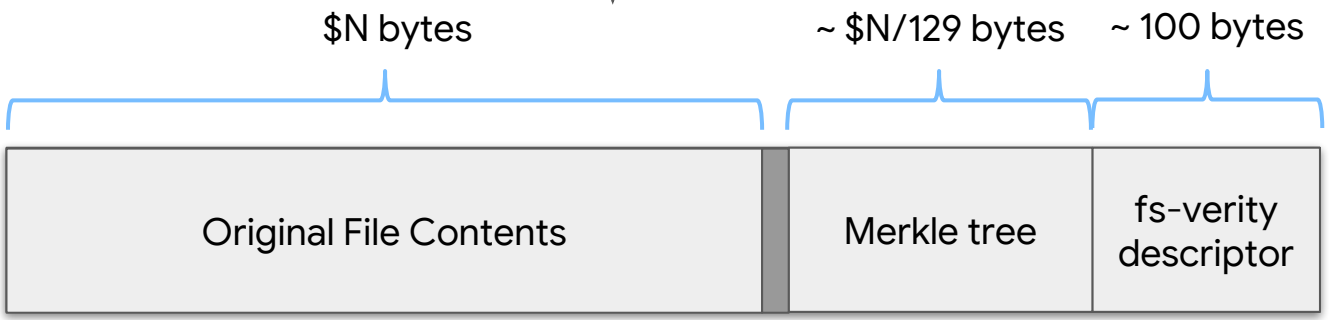
- Partial Disk: Protects **selected file system file content**.
- Facilitates **incremental updates** to **arbitrary subsets** of the file system.
- **Significantly reduces complexity** in deployment.
- Trade-off: File system **metadata unauthenticated**.
    - Opportunity for attacker to creatively undermine the authenticity of the system.

fs-verity

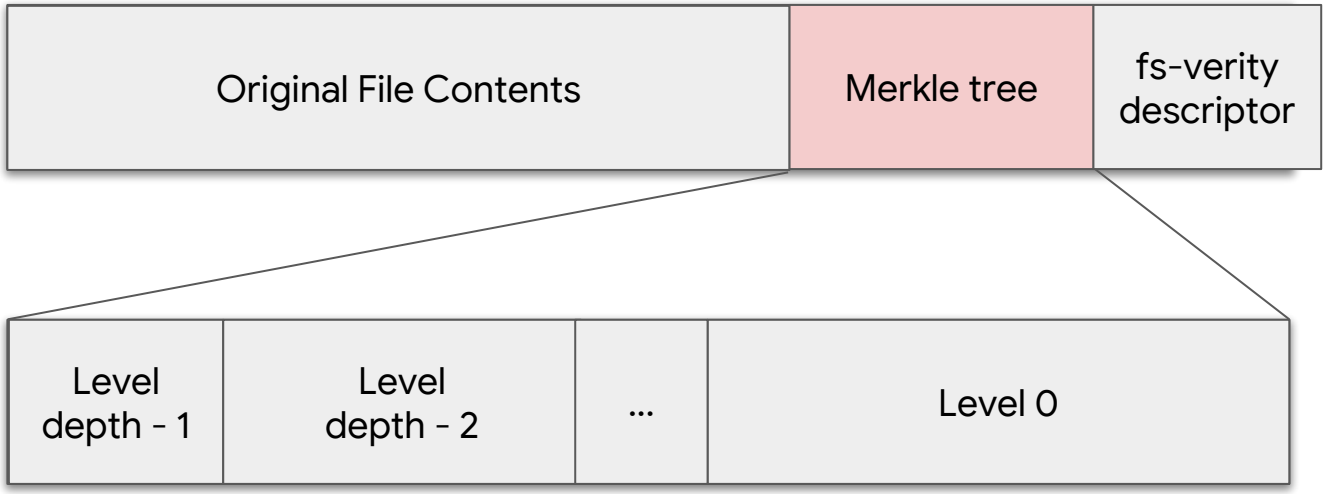


fs-verity: File format

```
$ head -c $N /dev/urandom > file
$ fsverity setup file
```



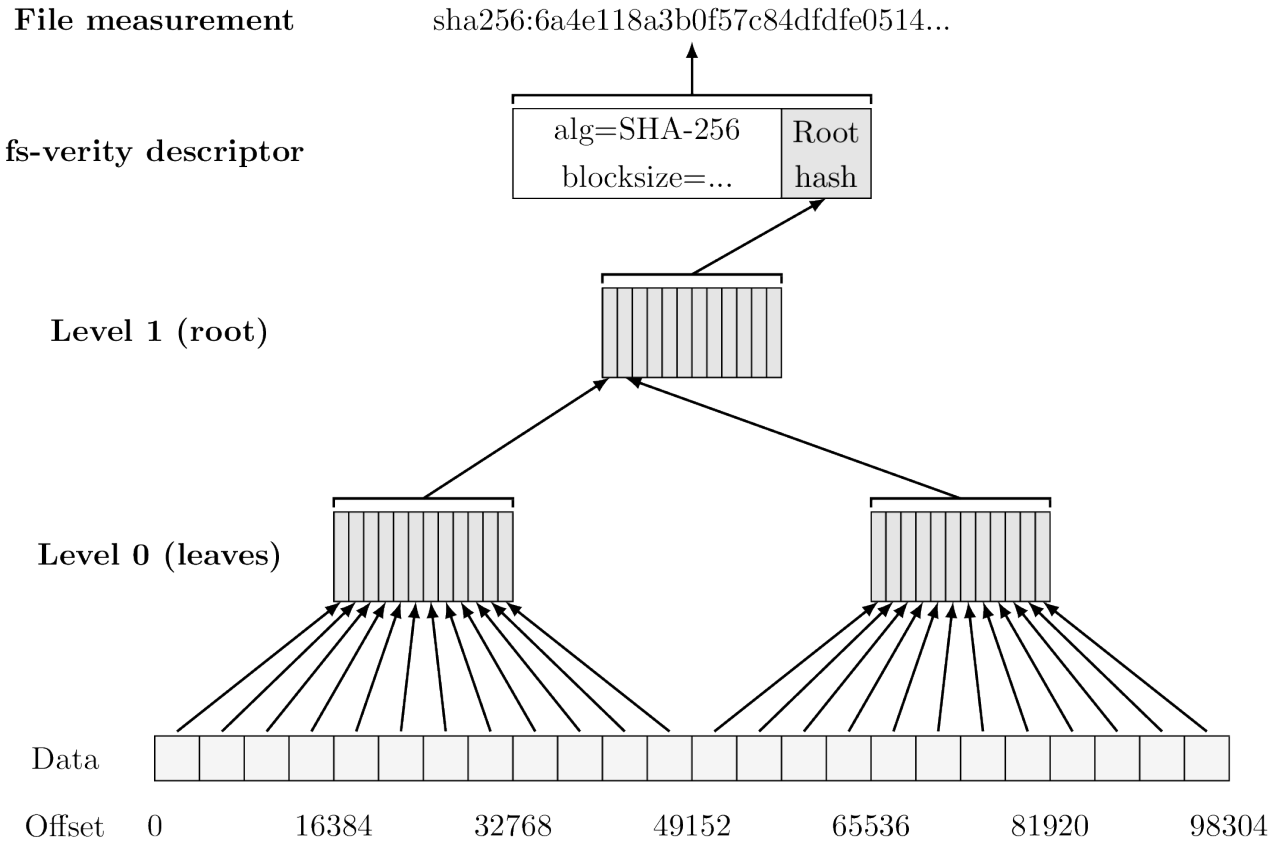
fs-verity: Merkle tree format



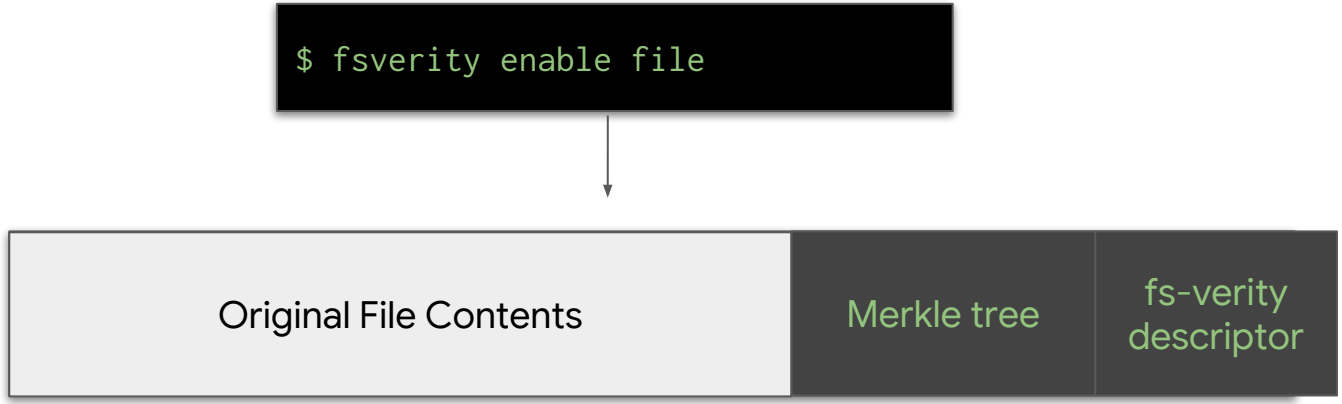
fs-verity: Additional metadata

```
struct fsverity_descriptor {
        ...
        __u8 log_data_blocksize; /* e.g. 12 = 4096-byte blocks */
        ...
        __le16 data_algorithm;   /* e.g. 1=SHA-256, 2=SHA-512 */
        ...
        __le64 orig_file_size;

}; /* followed by variable-length metadata items (extensions) */

/* extension items */
#define FS_VERITY_EXT_ROOT_HASH         1
#define FS_VERITY_EXT_SALT              2
#define FS_VERITY_EXT_PKCS7_SIGNATURE   3
```

fs-verity: Computing the file measurement

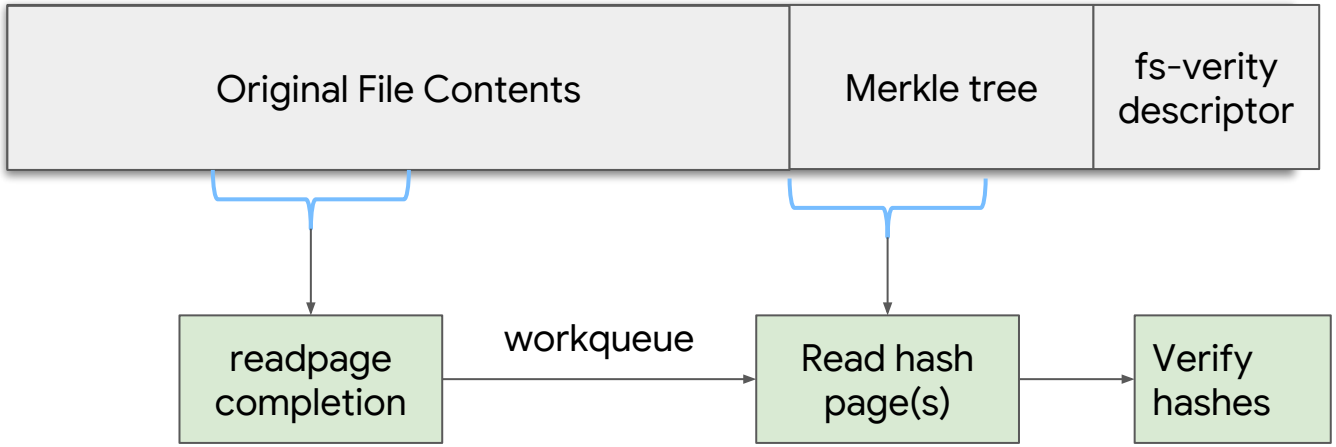


fs-verity: Enabling



- FS_IOC_ENABLE_VERITY
- File becomes read-only!
- Metadata is hidden from userspace

fs-verity: Reading data



- `->readpages()` hook covers both `read()` and `mmap()` accesses
- Hash pages are cached in page cache for efficiency
- Direct I/O is forbidden (falls back to buffered I/O)

fs-verity: File measurements

fs-verity provides file measurements (hashes) in constant time

- … subject to on-access enforcement
  - Applications get EIO at runtime if they try to read corrupted data

- File measurements available in kernel, but also exposed to userspace via FS_IOC_MEASURE_VERITY:

```
$ fsverity measure /bin/ls
sha256:9fef94de94184dc647a6f98f055896e2c13bf90052c73ca6324c0eb2bffc7991 /bin/ls
```

fs-verity: Use cases

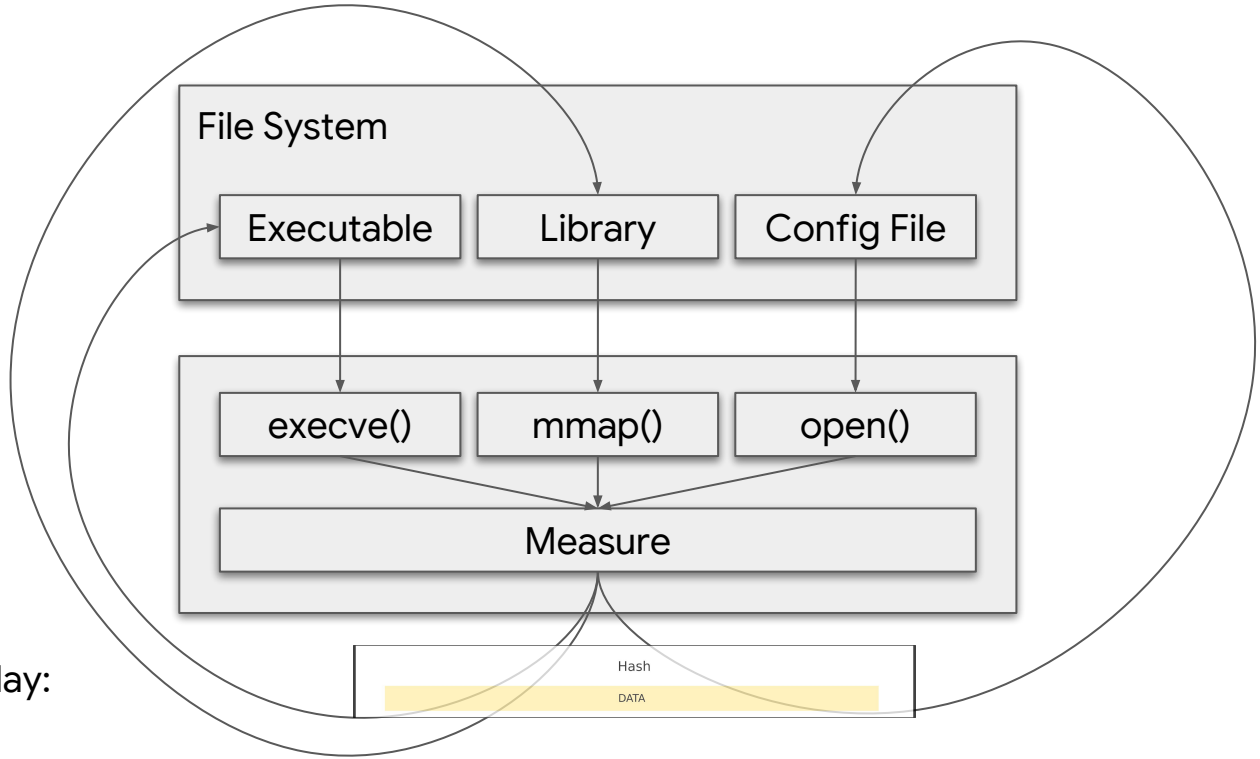
Categories of use cases:

- Integrity-only
  - Detect/prevent accidental corruption only
- Audit
  - Log the file measurement, but no enforcement
- Authenticity ("appraisal")
  - Detect/prevent both accidental and malicious changes

Users will be able to choose how to use fs-verity:

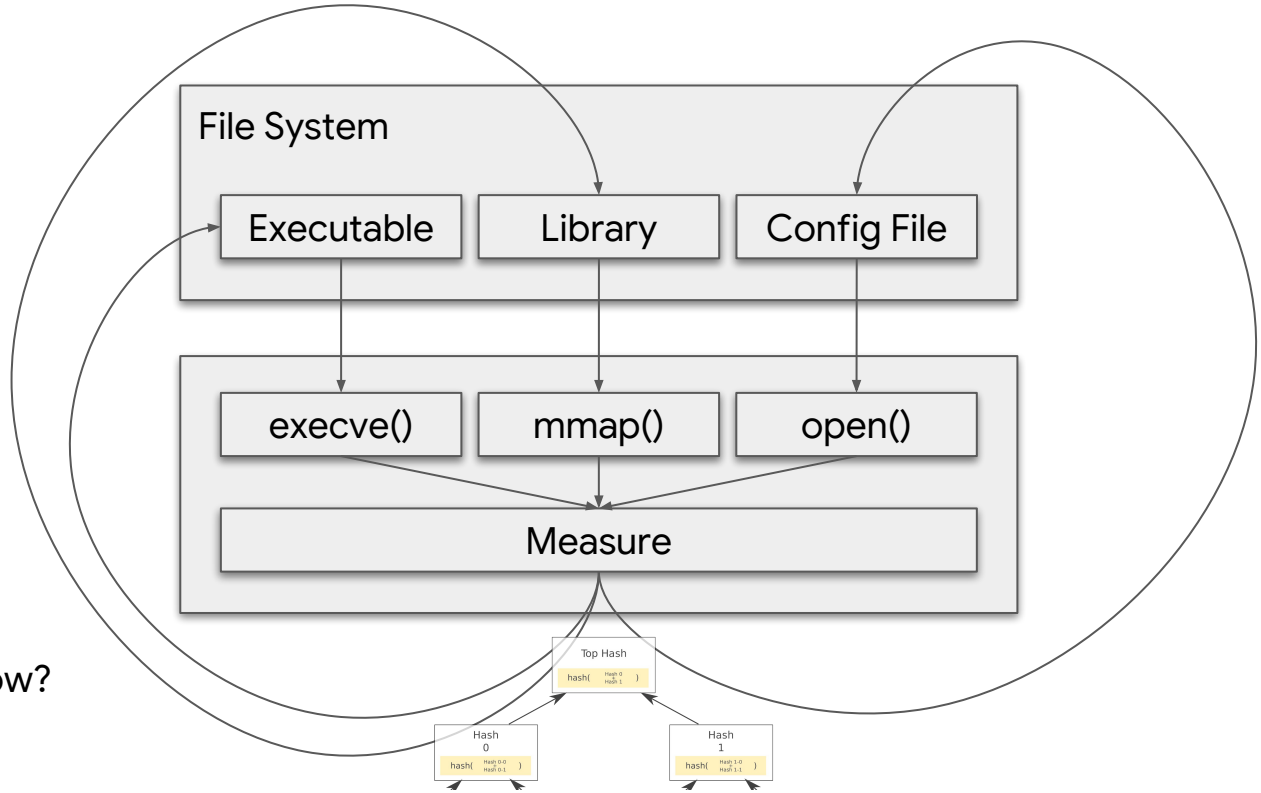
- IMA (Integrity Measurement Architecture) policy
  - Complex, but most feature-rich
  - Planned (not yet in patchset)
- Userspace-only policy, using FS_IOC_MEASURE_VERITY
- Built-in signature verification against fs-verity keyring

Integrity Measurement Architecture (IMA)



Today:

Integrity Measurement Architecture (IMA) with fs-verity



Tomorrow?

fs-verity: Resources

- Linux kernel patchset
  - https://git.kernel.org/pub/scm/linux/kernel/git/ebiggers/linux.git/log/?h=fsverity
- Userspace utility
  - https://git.kernel.org/pub/scm/linux/kernel/git/ebiggers/fsverity-utils.git
- Tests
  - https://git.kernel.org/pub/scm/linux/kernel/git/ebiggers/xfstests-dev.git/log/?h=fsverity

Thank You