



LogCabin

A split-view resistant ledger as a
t-log witness

github.com/project-oak/logcabin

Ernesto Ocampo
Google

transparency.dev
Sep 2026

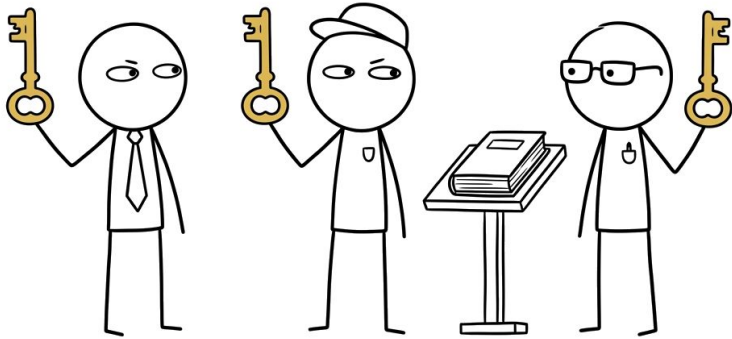
LogCabin

An append-only ledger.

LogCabin

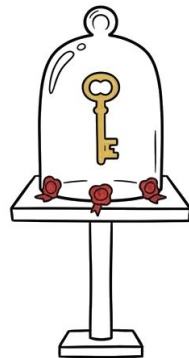
An append-only ledger where split views are impossible

TODAY



Split views **don't** happen:
key-holders would need to collude.

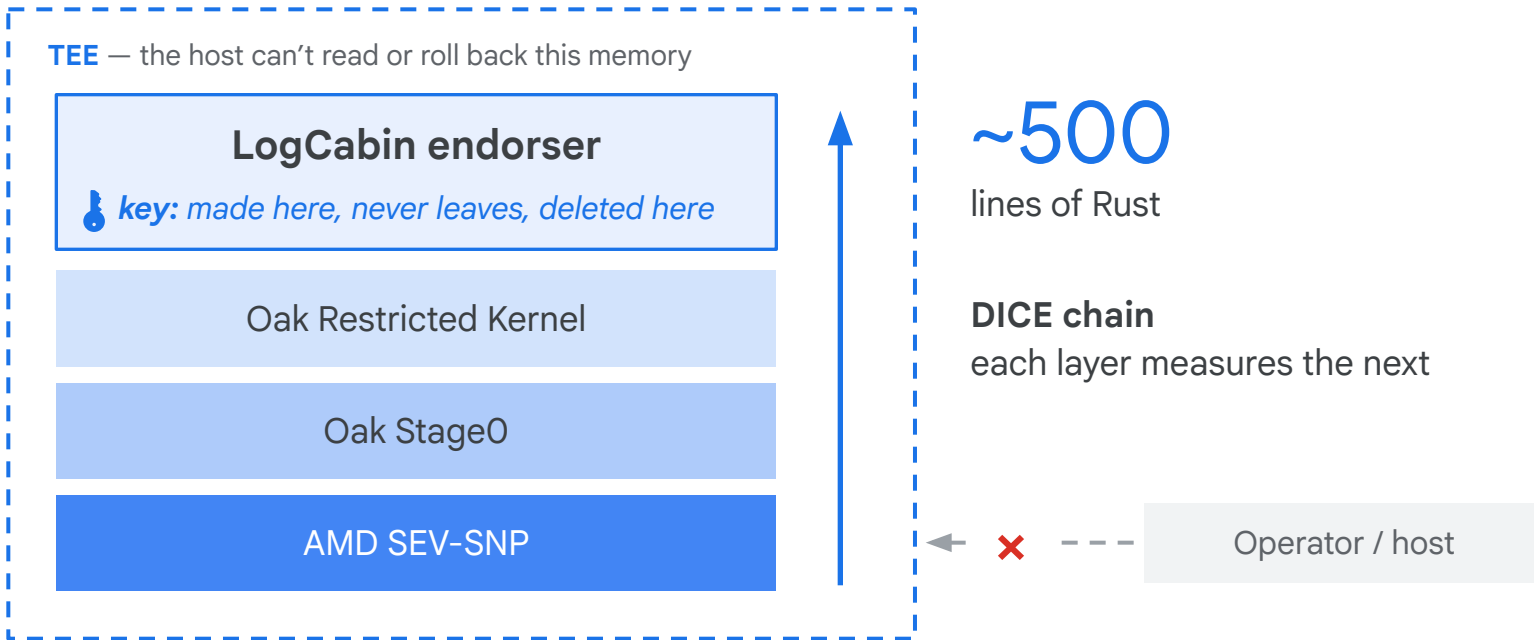
LOGCABIN



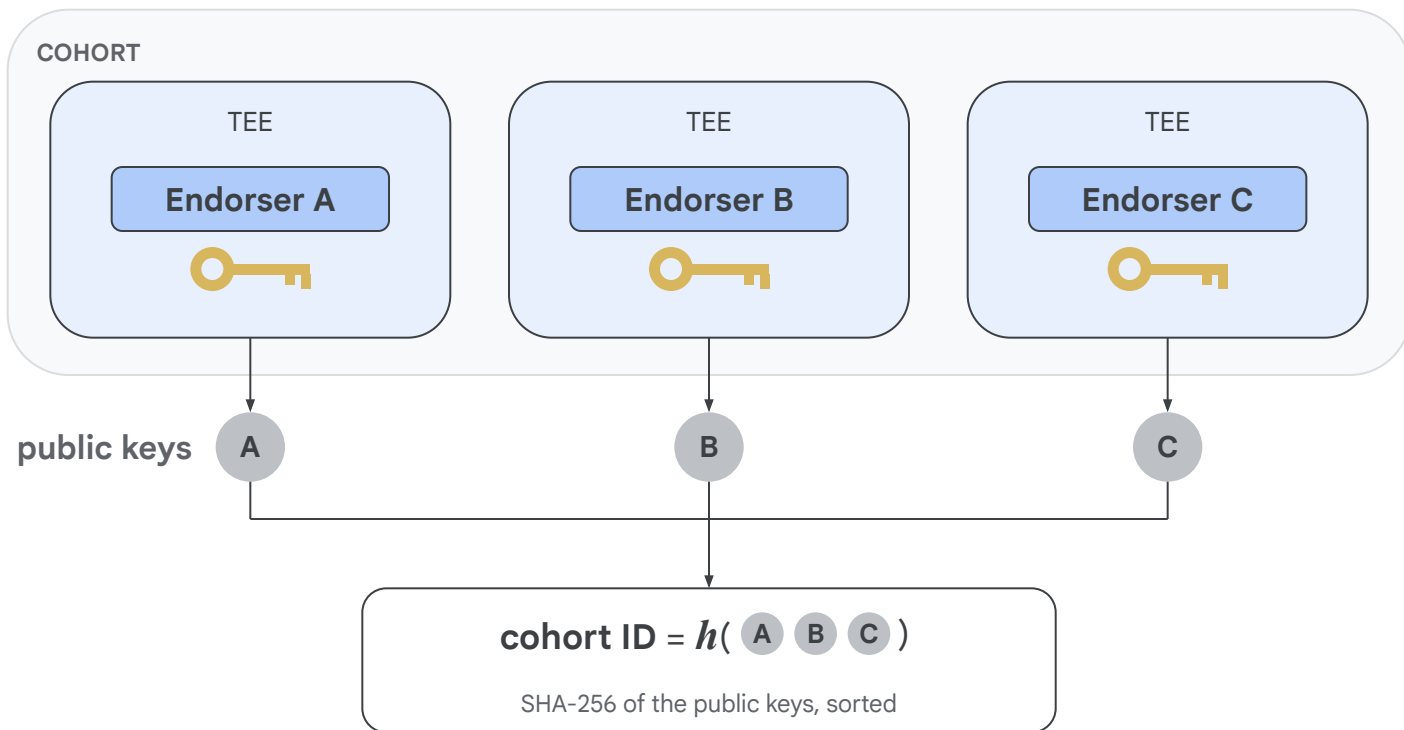
Don't look
at me.

Split views **don't** happen:
nobody holds the keys.

Nobody holds the keys

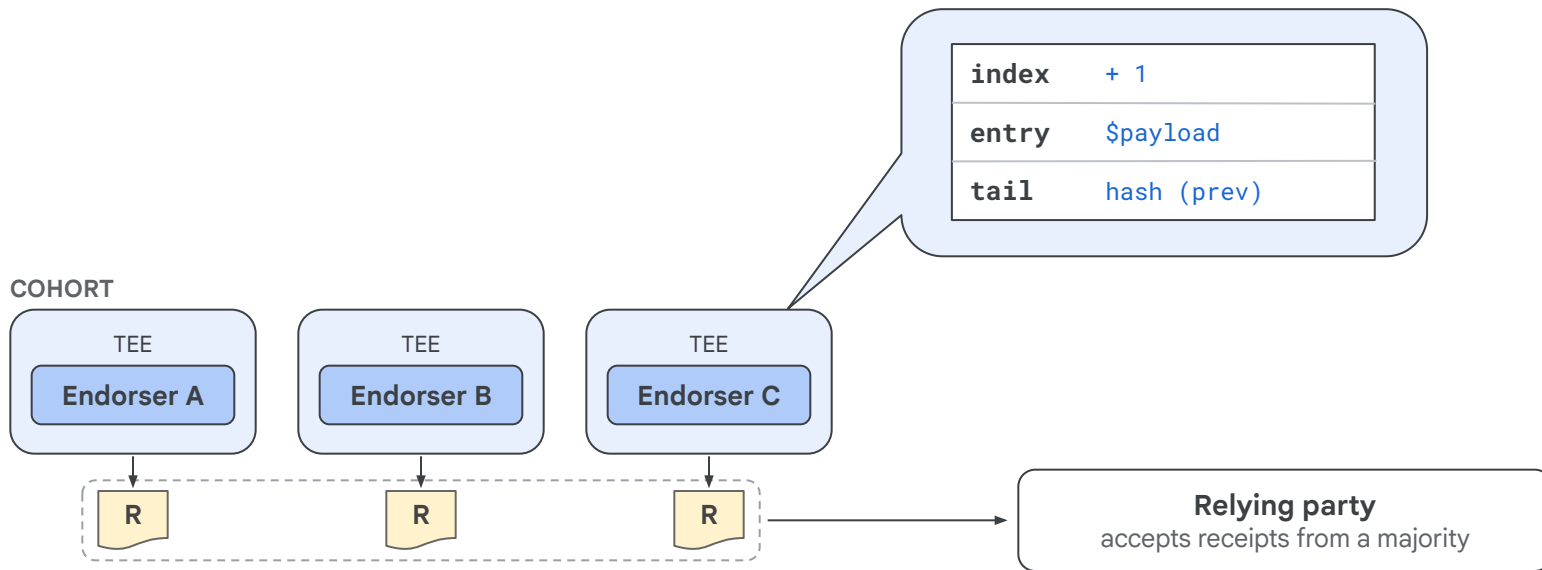


A cohort of endorsers, each in a TEE



Private key never leaves the TEE
The public key is the endorser's identity.

Endorser state



R = an endorser's signature over its current state

Challenge the claims

“Nobody — not service administrators, infrastructure operators, or external attackers — can produce two different views of a ledger that are both accepted as valid by relying parties.”

```
impl Falsifier for YourAttempt {  
  fn attempt(self, endorsers: Vec<Endorser<Uninitialized>>) -> (View, View) {  
    unimplemented!("build two conflicting-but-verifiable views")  
  }  
}
```

<https://github.com/project-oak/logcabin/tree/main/challenge>

Allowed: direct interaction with the endorser state machine.

Not allowed: reading private keys from memory, rolling back endorser state, etc



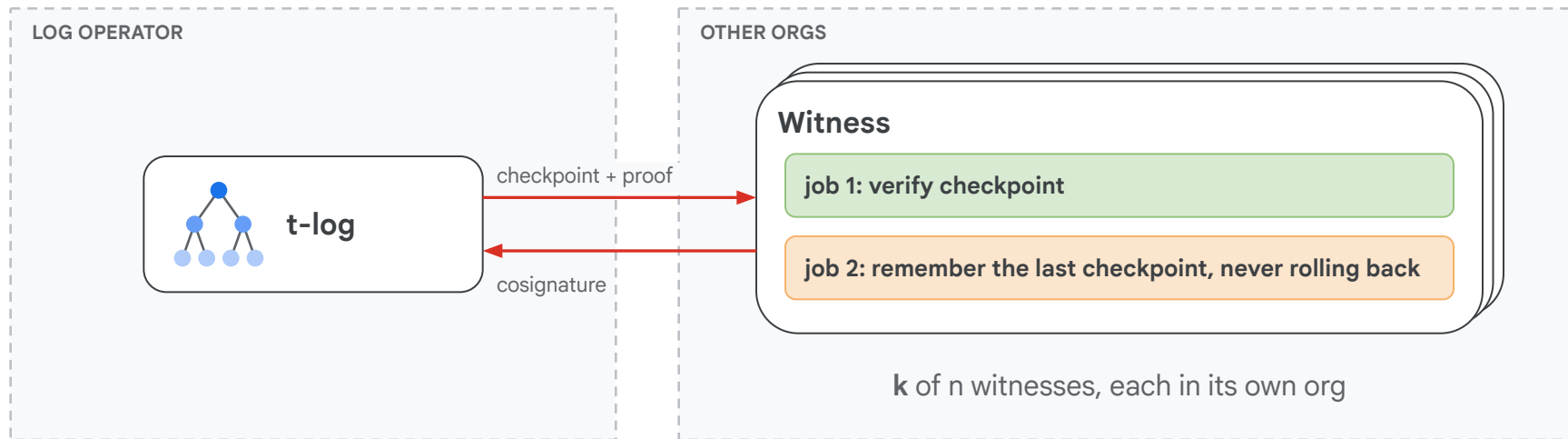
LogCabin is an append-only ledger.

No forks.

BOTTOM LINE

Out of scope: TEE physical attacks.

A C2SP witness has two jobs



RELYING PARTY CHECKS

k cosignatures

from witnesses it trusts

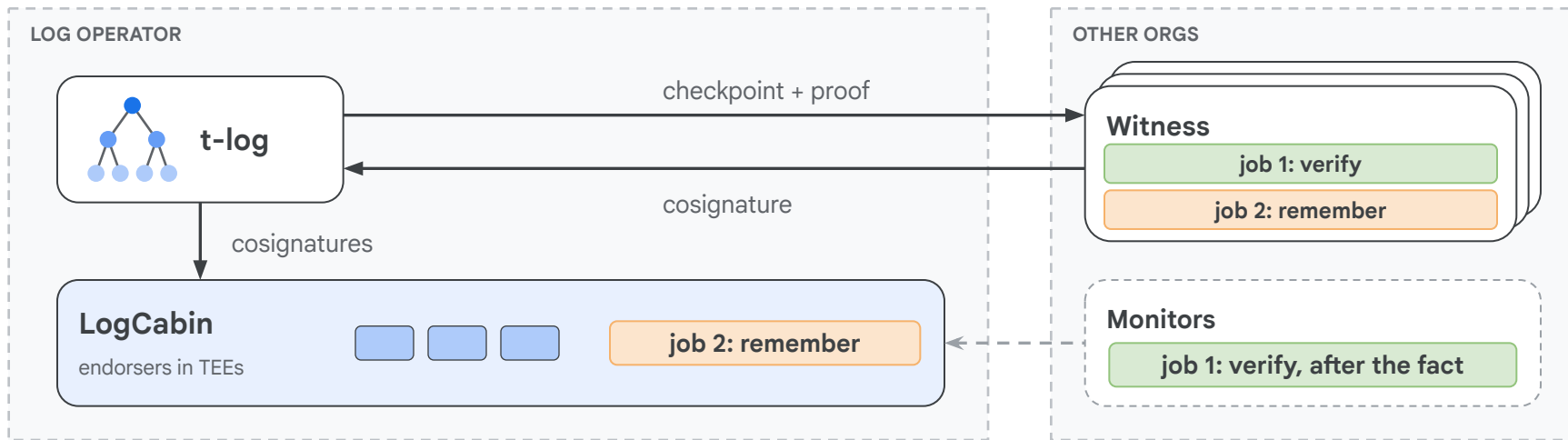
A SPLIT VIEW IS

Possible and may not be detected

if t-log and k witnesses collude

LogCabin does Job 2: it can't roll back, and it can't fork.

a Power t-log



RELYING PARTY CHECKS

K cosignatures + Q receipts + handovers

That's a lot. Ok for servers, not for phones.

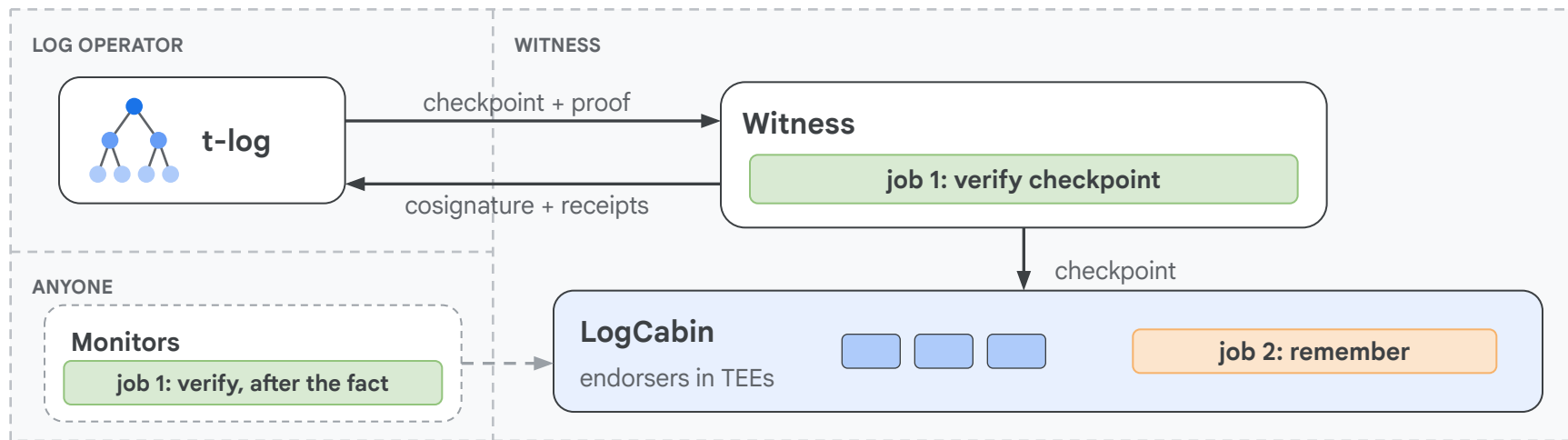
A SPLIT VIEW IS

Detected

a fork stays in the ledger forever; monitors find it

The log **can't roll back** its own history, even if its operator wants to.

b Power witness



RELYING PARTY CHECKS

K cosignatures + Q receipts + handovers

That's a lot. Ok for servers, not for phones.

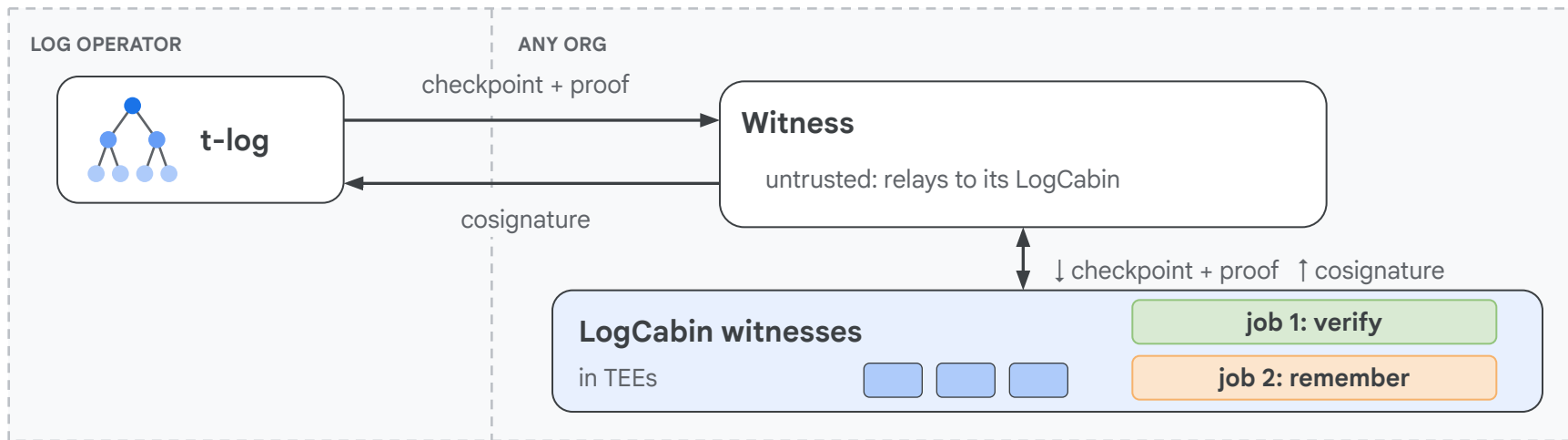
A SPLIT VIEW IS

Detected

it can't roll back or erase a cosignature; monitors find it

Any witness can run it and **upgrade it guarantees**.

c Power² witness



RELYING PARTY CHECKS

Cosignatures / receipts + handovers

Still a lot of data

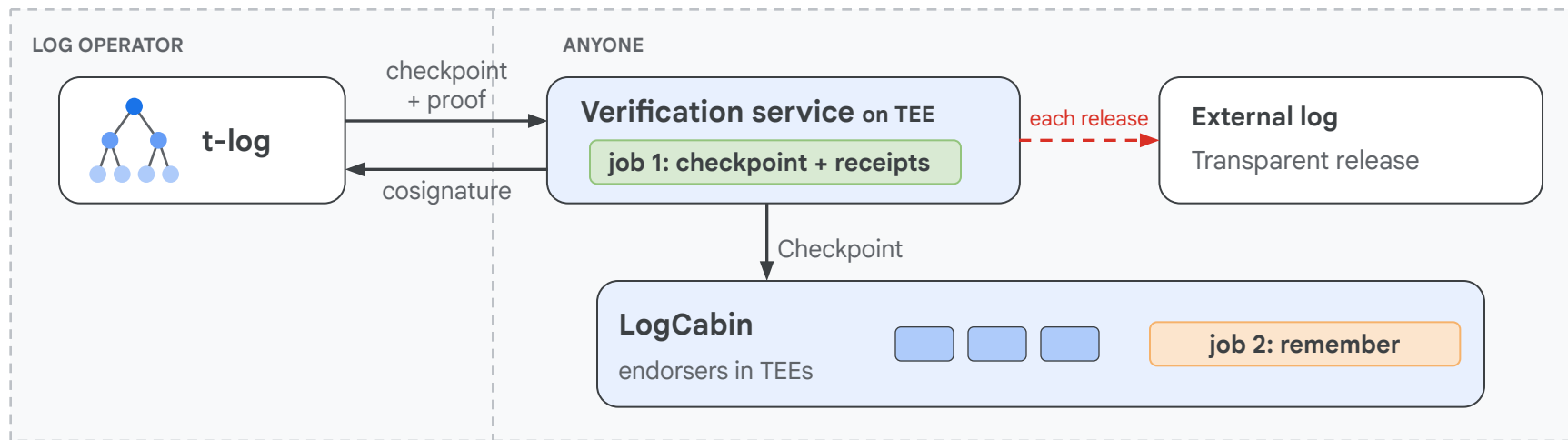
A SPLIT VIEW IS

Prevented

LogCabin won't sign a checkpoint that doesn't extend the last one

No forks. Must handle rotating endorser keys.

d Verification service



RELYING PARTY CHECKS

Verifier's stamp + attestation

Small data. Ok for phones.

A SPLIT VIEW IS

Prevented

unless the TEE breaks, or a verifier release is bad (and public)

Verification service is **transparently released**.

Who has to collude for a split view?

C2SP WITNESSES

today

Log operator

+

Witnesses

LOGCABIN ONLY

options c, d

Log operator

+

TEE vendor

MIX

k of n, at least one attested

Log operator

+

Witnesses

+

TEE vendor

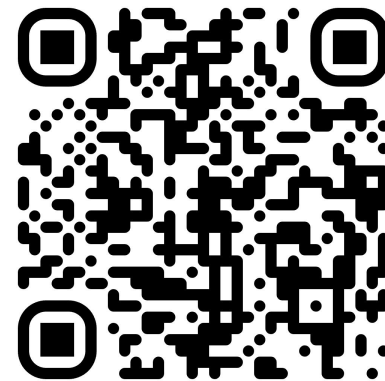
- + An orthogonal failure domain
- + A richer ecosystem
- + More players must collude
- + A lower bar to be trusted

A new kind of witness

Anyone can run a service that everyone can trust.

BOTTOM LINE 2

- + An orthogonal failure domain
- + A richer ecosystem
- + More players must collude
- + A lower bar to be trusted



github.com/project-oak/logcabin

A new kind of witness

Anyone can run a service that everyone can trust.

Thank you

: -)

Project status

TEE endorser

Mostly done · open source

~500 lines of Rust, runs on Oak Restricted Kernel

Coordinator

In progress

Untrusted, but it keeps the instance alive

C2SP witness bridge

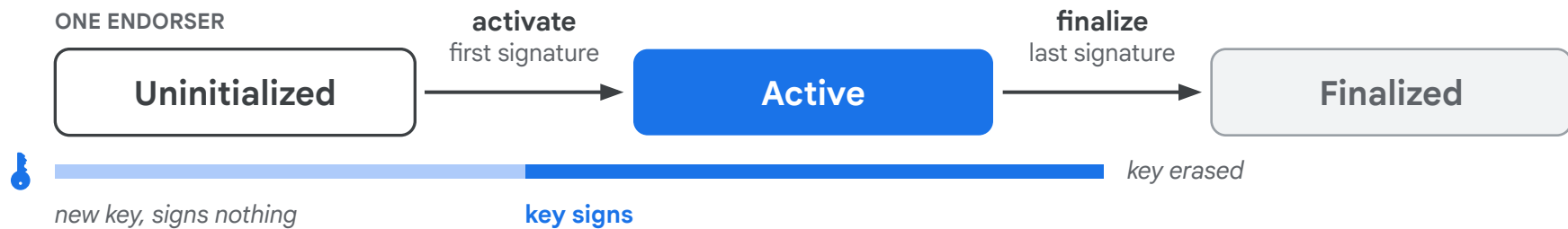
Design

Options from this talk. Feedback welcome.

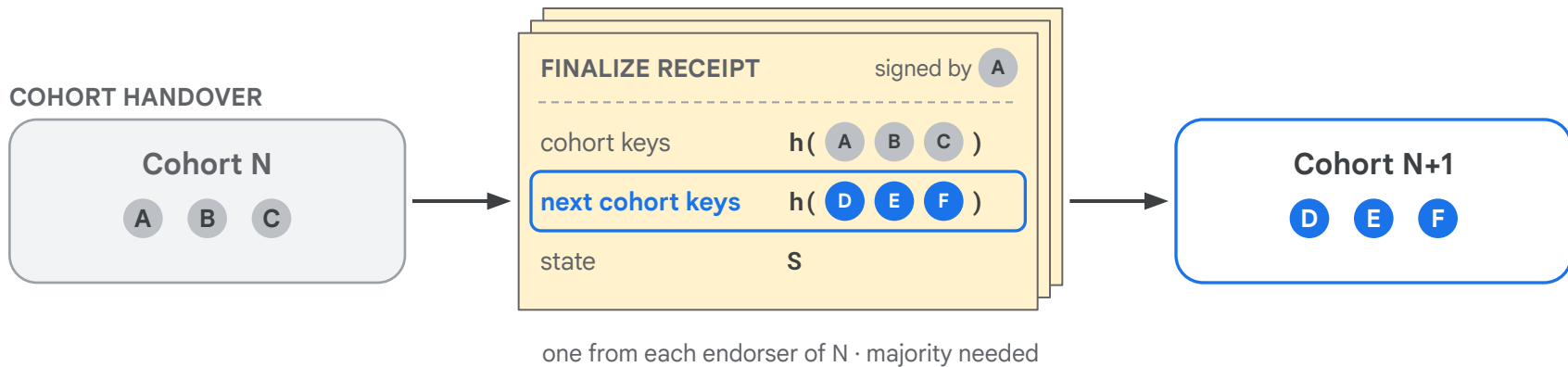
Endorsers guarantee safety. Coordinators look after liveness.

github.com/project-oak/logcabin

Endorser state machine



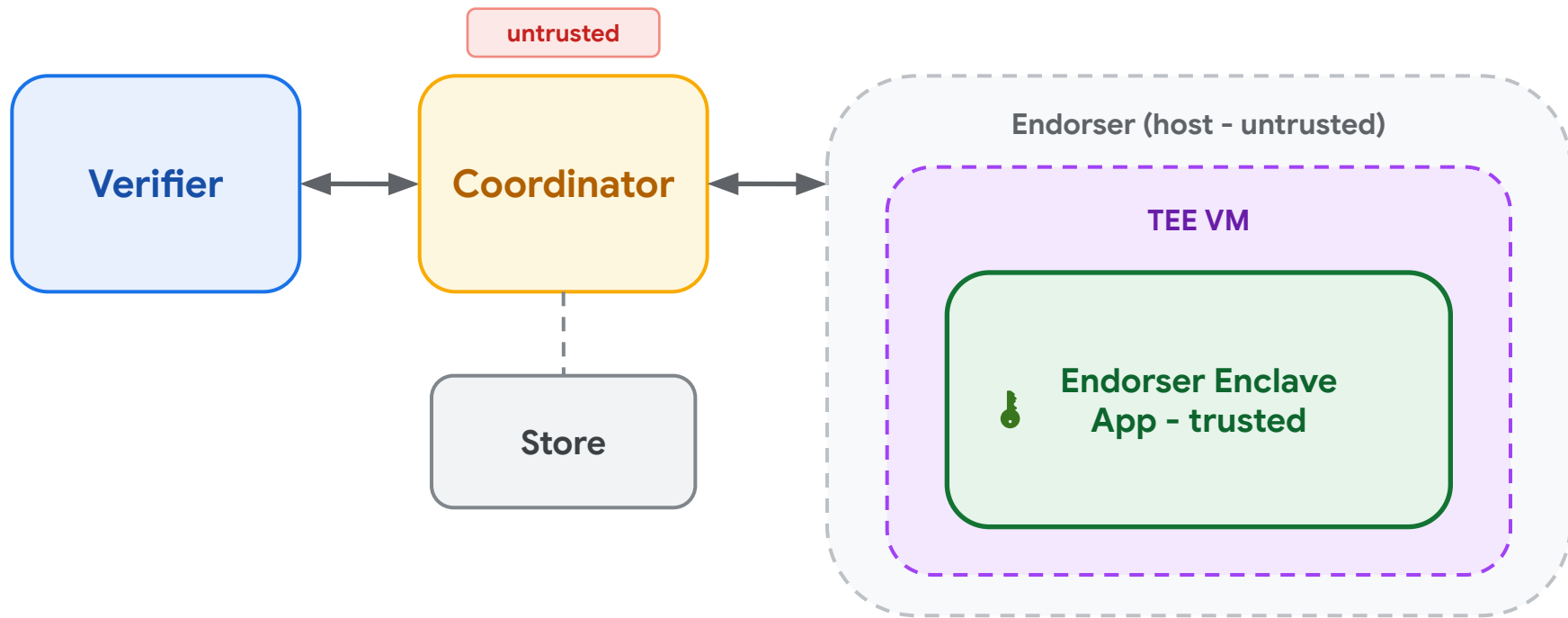
Cohort handover



A finalizing endorser **erases its key** → a cohort can hand over only once → **no fork**.

LogCabin Architecture

Untrusted orchestration layer with hardware-isolated TEE endorser enclaves



LogCabin vs Nimble

Same core idea as Nimble. Main changes:

1 Two-phase handover (not three)

Simpler endorsers **~500 lines of Rust**. No “appends” parameter on activation - endorsers only check majority finalized on **same** state. Shorter handovers pauses. Linearization moves to the untrusted coordinator.

2 Unconditional appends

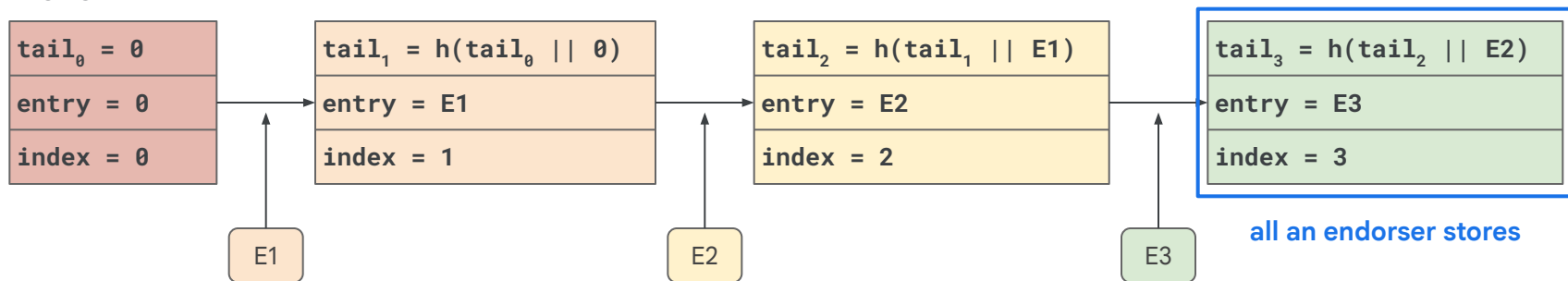
Clients can submit append requests without knowing the next index. Saves round trips and avoids race conditions when clients are distributed.

3 Far less data per handover

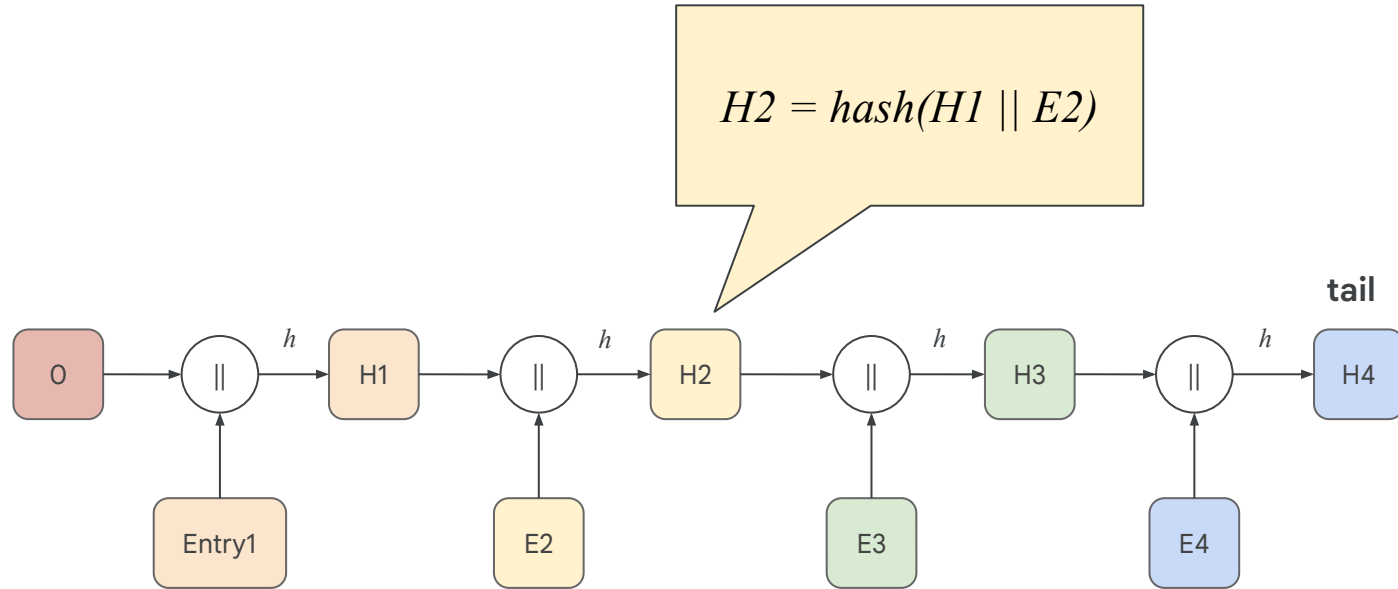
Activation does not need old endorser’s ledgers nor Nimble’s “Appends” parameter.
 $O(\text{new} \times \text{old} \times \text{ledgers}) \rightarrow O(\text{new} \times (\text{old} + \text{ledgers}))$

A chain of entries

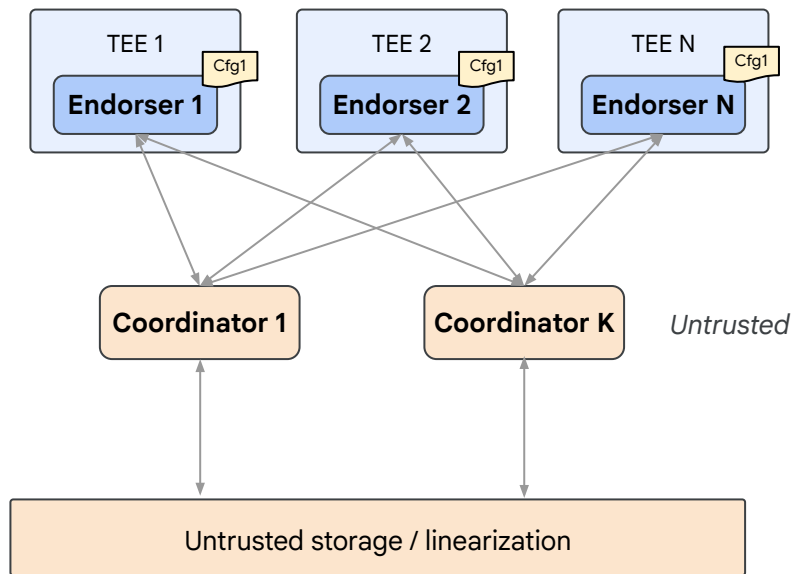
HASH CHAIN



A hash chain

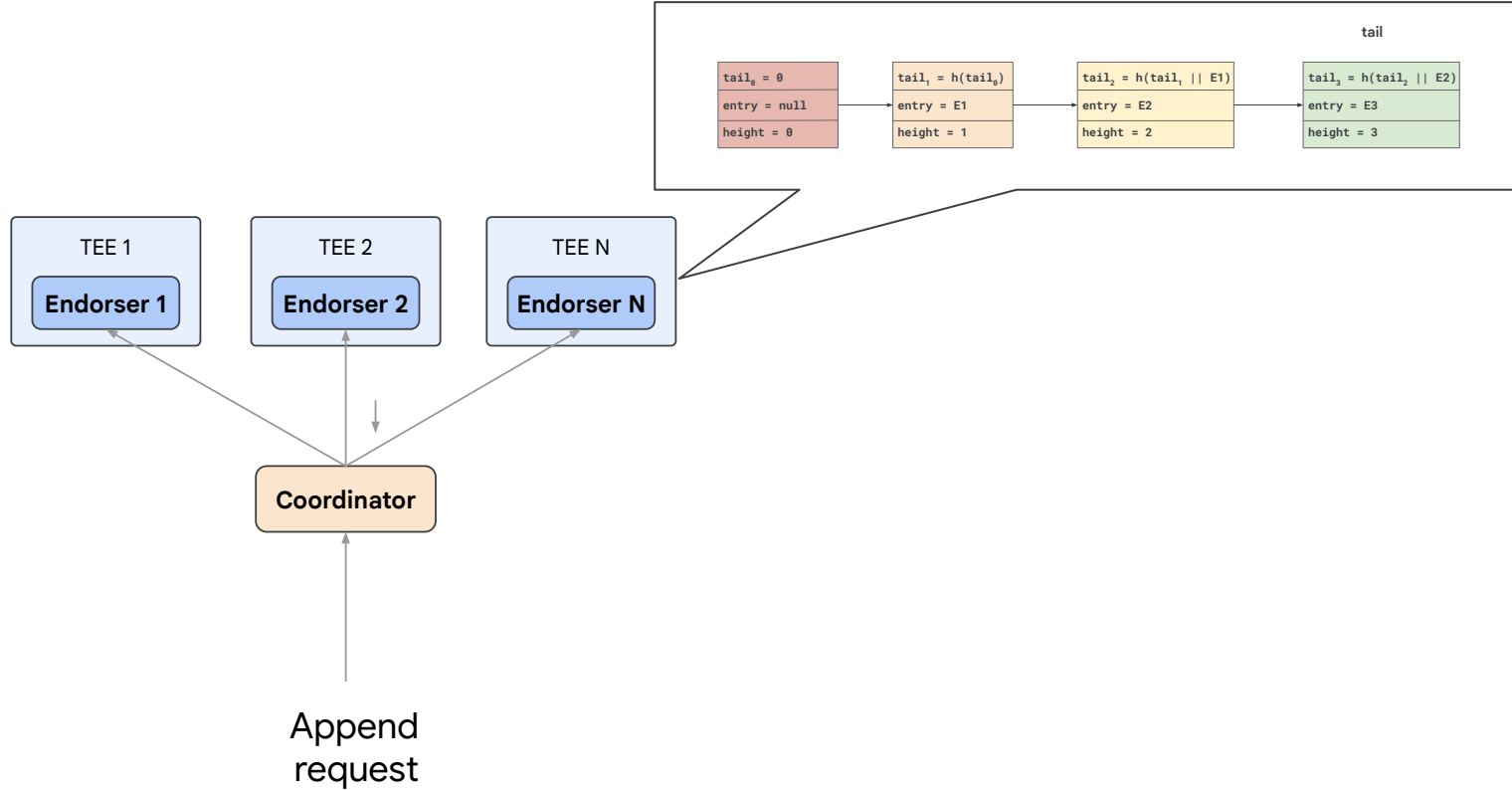


Concepts

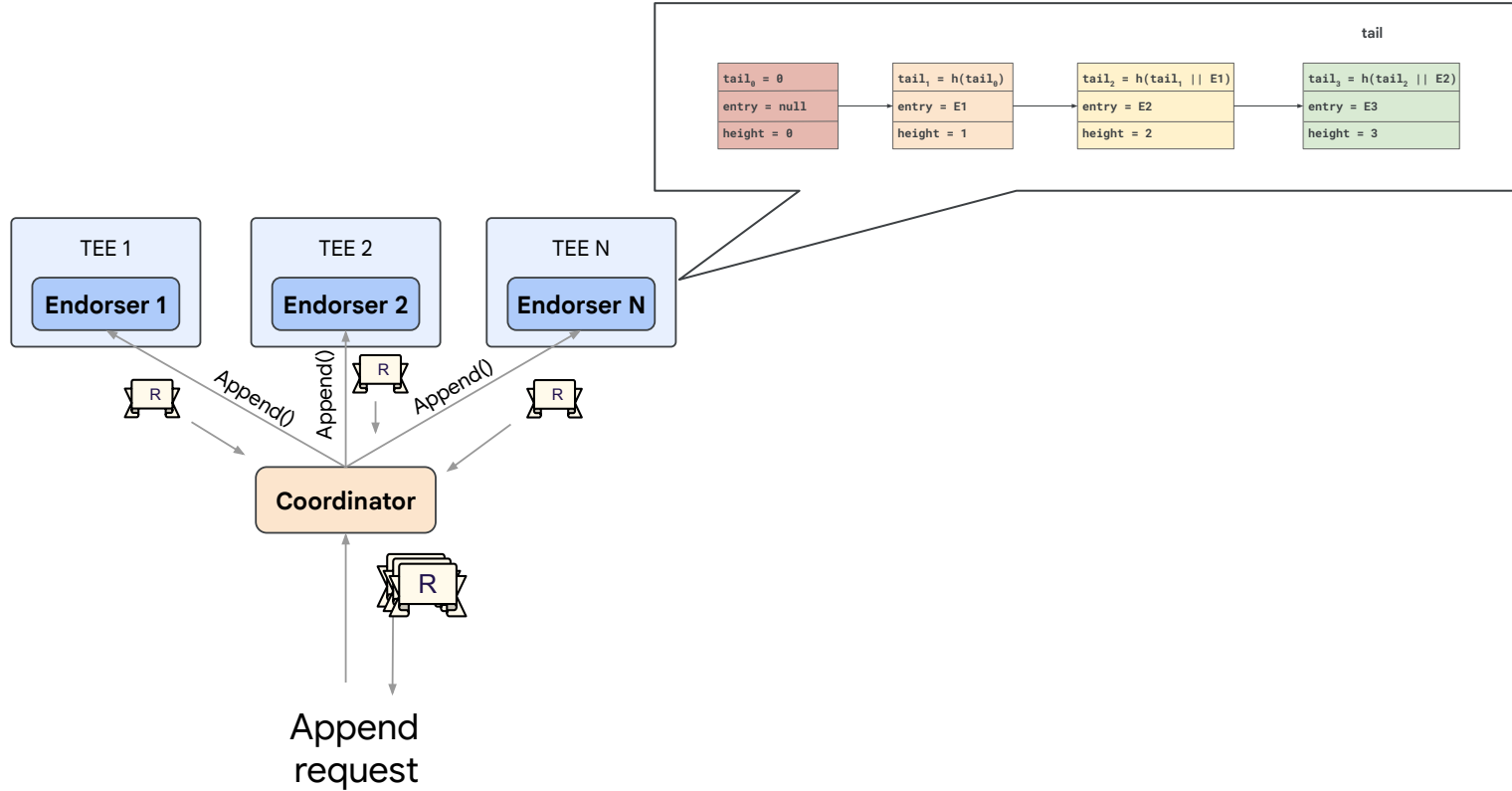


- **Endorser**
 - Signs things
 - Keeps a hash chain
 - Private Key never shared
 - Public key bound to hardware attestation
- **Coordinator**
 - Does a bunch of important admin
- **Cohort**
 - *A set of endorsers working together. All have the same config.*
- **Config**
 - *The set of all endorser public keys in the cohort.*
- **Handover**
 - Handing over of one cohort to the next
- **Instance**
 - A service from uninterrupted hand-overs from one cohort to the next

Processing an append



Processing an append



Cohort Handover



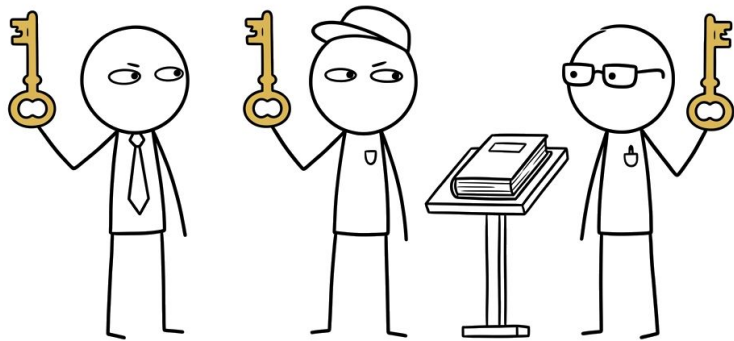
Properties

- Majority quorum based
 - Relying party verifies receipts from individual endorsers
- Split views are impossible as long as:
 - We are running “true” LogCabin
 - If a malicious or buggy endorser runs, it’s auditable forever
 - The LogCabin instance stays alive - i.e. the majority of endorsers stay alive
 - The TEE is honest
 - Collusion between t-log operator and witnesses → collusion with hardware manufacturer
 - Out of scope: physical access attacks
- If an instance is lost, it can’t be recovered

LogCabin

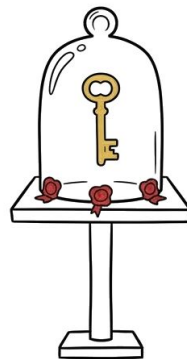
An append-only ledger where split views are impossible

SPLIT TRUST



Split views **don't** happen:
key-holders would need to collude.

LOGCABIN



Don't look
at me.

Split views **don't** happen:
nobody holds the keys.