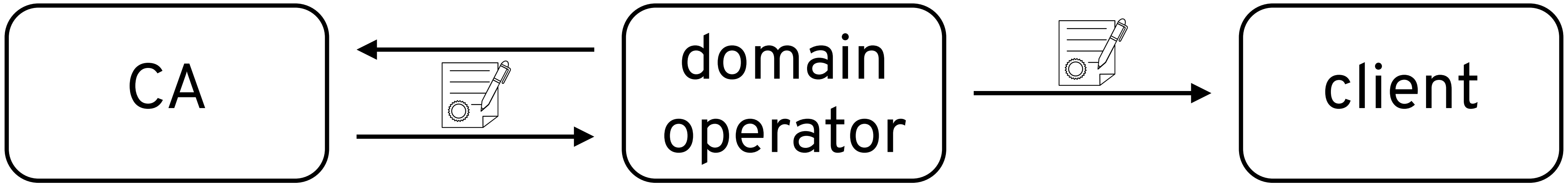
SCT AUDITING IN CERTIFICATE TRANSPARENCY

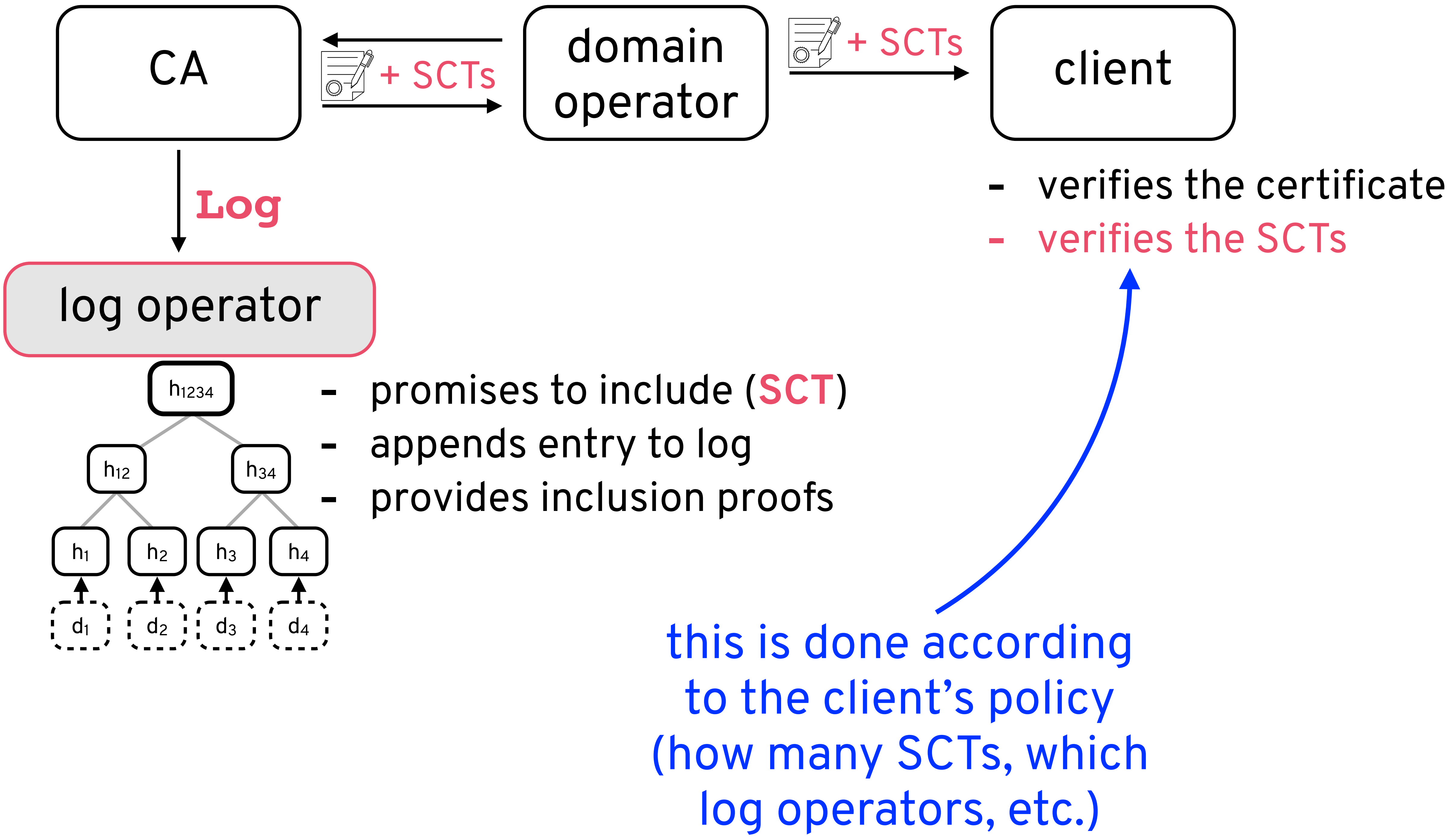
SARAH MEIKLEJOHN, JOE DEBLASIO, DEVON O’BRIEN, CHRIS THOMPSON, KEVIN YEO, AND EMILY STARK (GOOGLE)

CERTIFICATE ISSUANCE



- verifies the certificate

CERTIFICATE TRANSPARENCY



CA

+ SCTs

domain operator

+ SCTs

client

Log

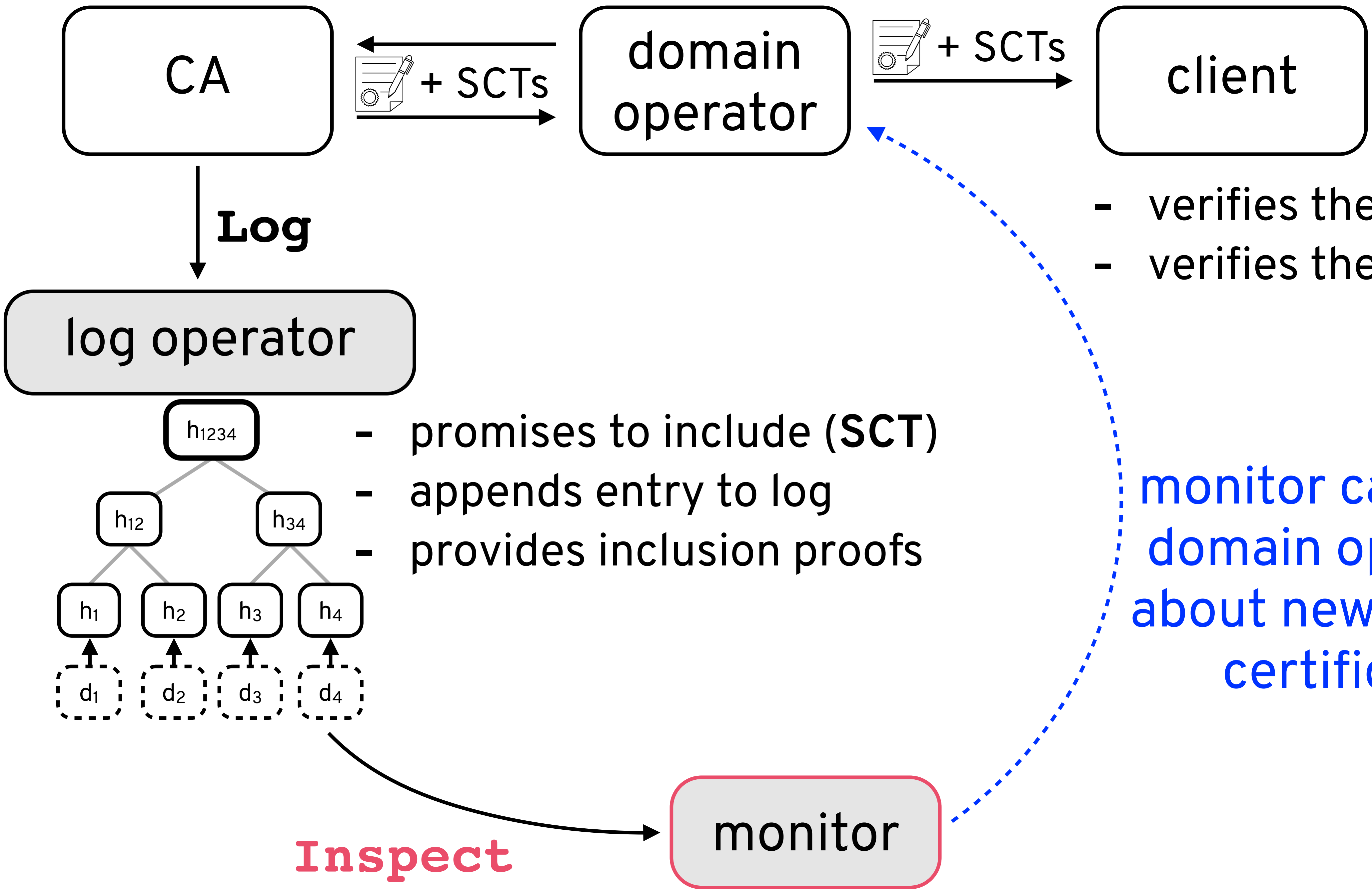
log operator

- promises to include (**SCT**)
- appends entry to log
- provides inclusion proofs

- verifies the certificate
- verifies the SCTs

this is done according to the client's policy (how many SCTs, which log operators, etc.)

CERTIFICATE TRANSPARENCY

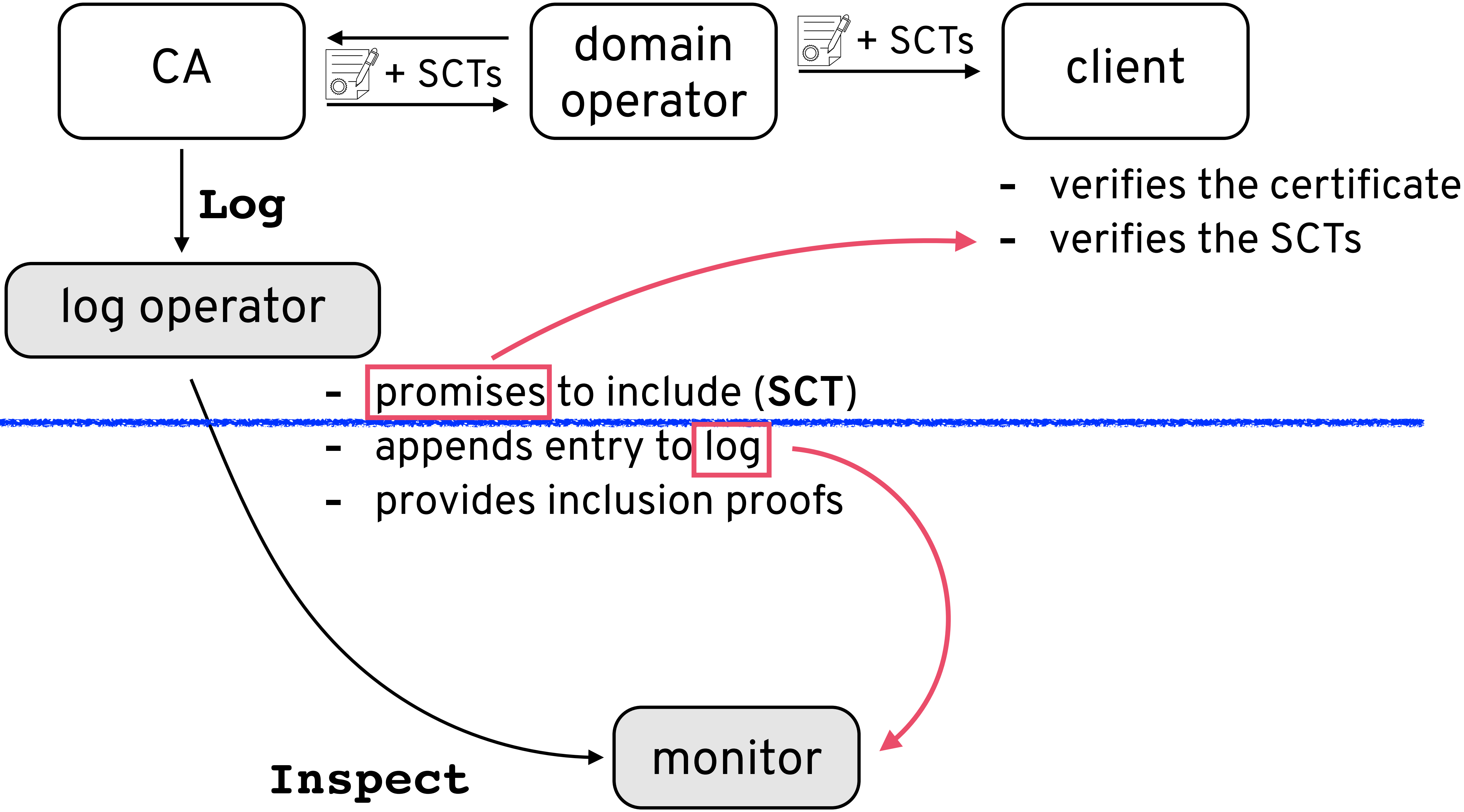


- verifies the certificate
- verifies the SCTs

- promises to include (**SCT**)
- appends entry to log
- provides inclusion proofs

monitor can notify domain operators about newly issued certificates

PROMISE VS. REALITY



CA

+ SCTs

domain operator

+ SCTs

client

- verifies the certificate
- verifies the SCTs

Log

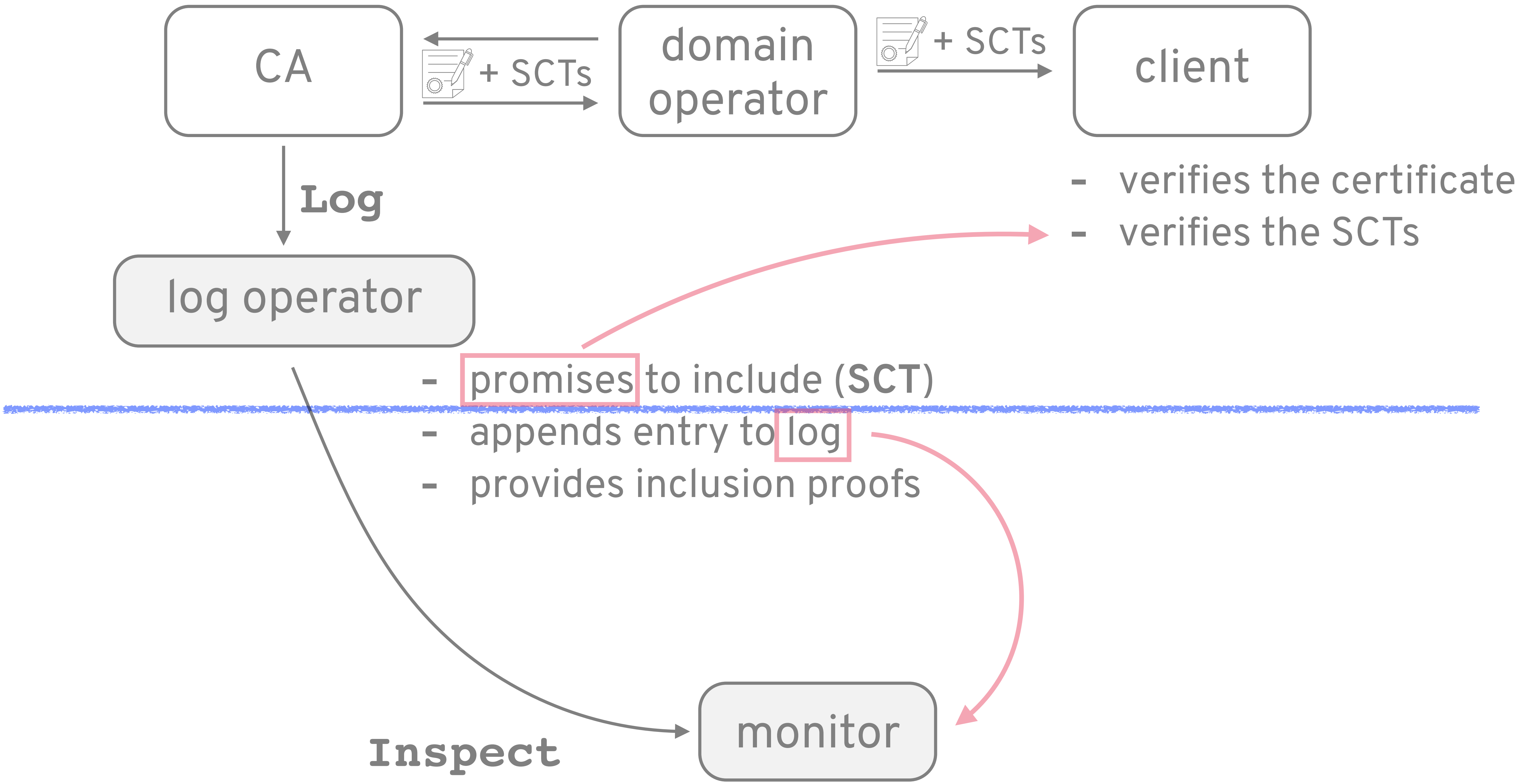
log operator

- promises to include (**SCT**)
- appends entry to log
- provides inclusion proofs

Inspect

monitor

PROMISE VS. REALITY

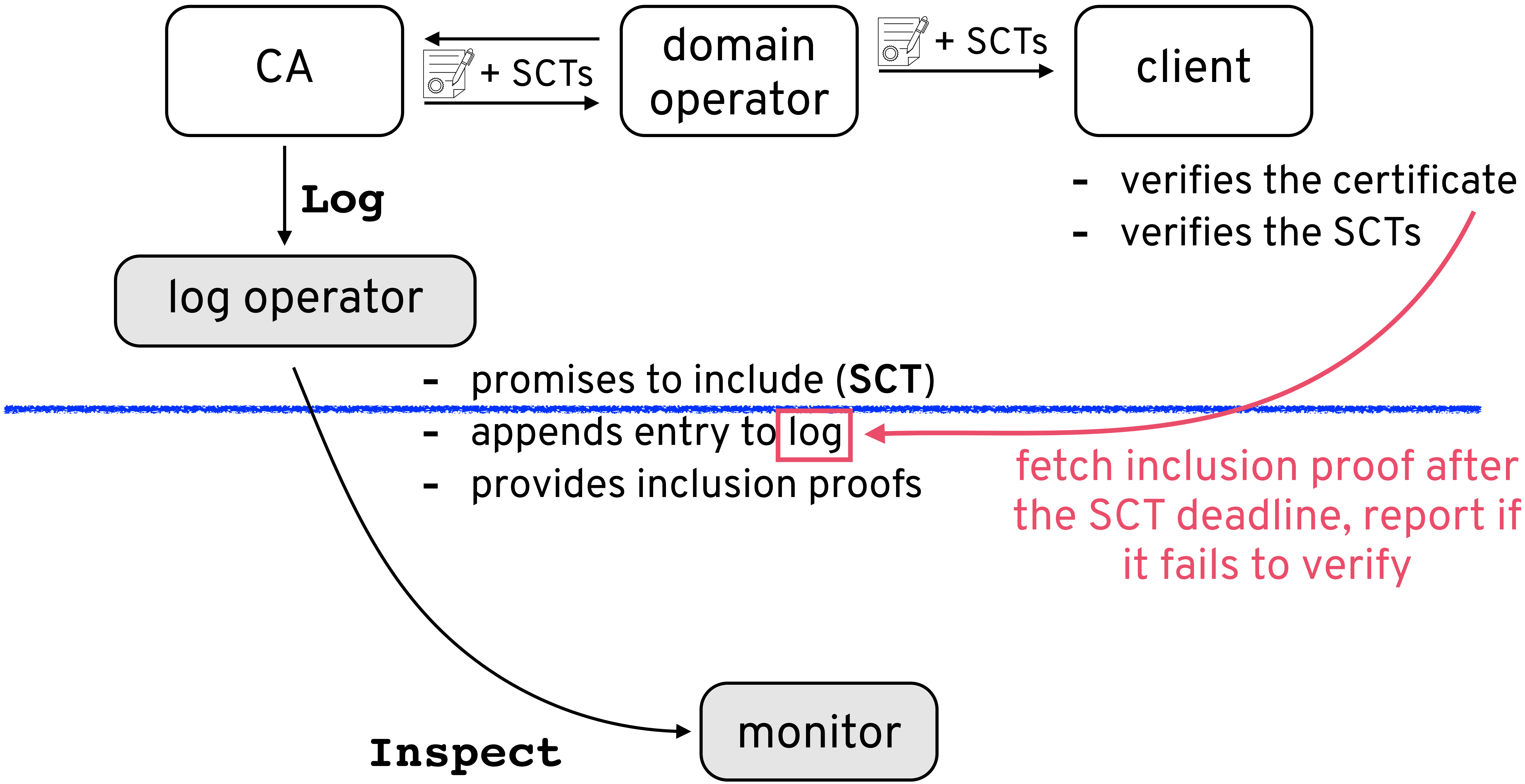


- verifies the certificate
- verifies the SCTs

- promises to include (**SCT**)
- appends entry to log
- provides inclusion proofs

HOW DO WE CHECK THAT CERTIFICATES SEEN BY CLIENTS ARE REALLY IN THE LOG?

A NAIVE APPROACH



CA

+ SCTs

domain operator

+ SCTs

client

- verifies the certificate
- verifies the SCTs

Log

log operator

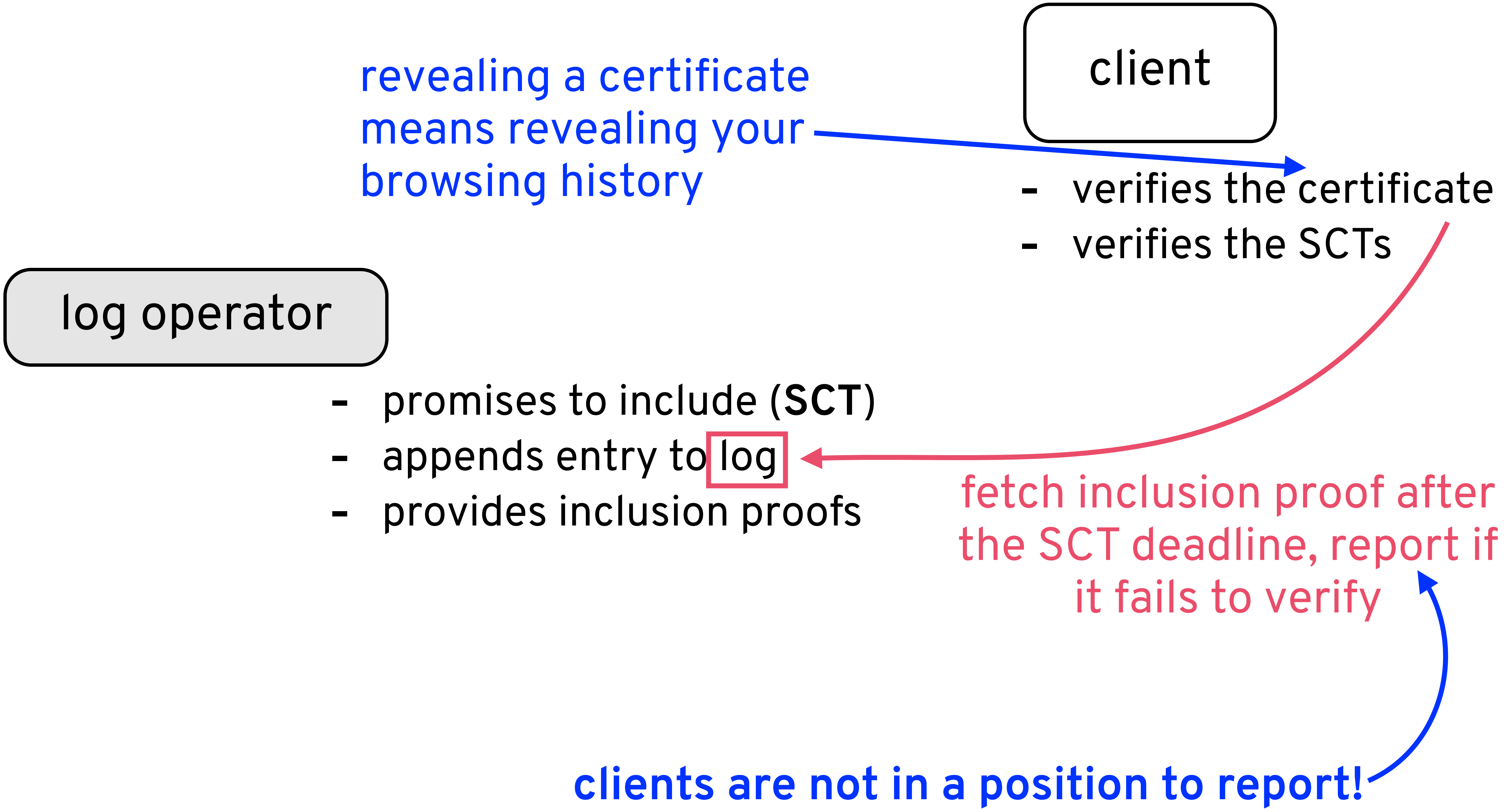
- promises to include (**SCT**)
- appends entry to log
- provides inclusion proofs

fetch inclusion proof after the SCT deadline, report if it fails to verify

Inspect

monitor

A NAIVE APPROACH



client

- verifies the certificate
- verifies the SCTs

revealing a certificate means revealing your browsing history

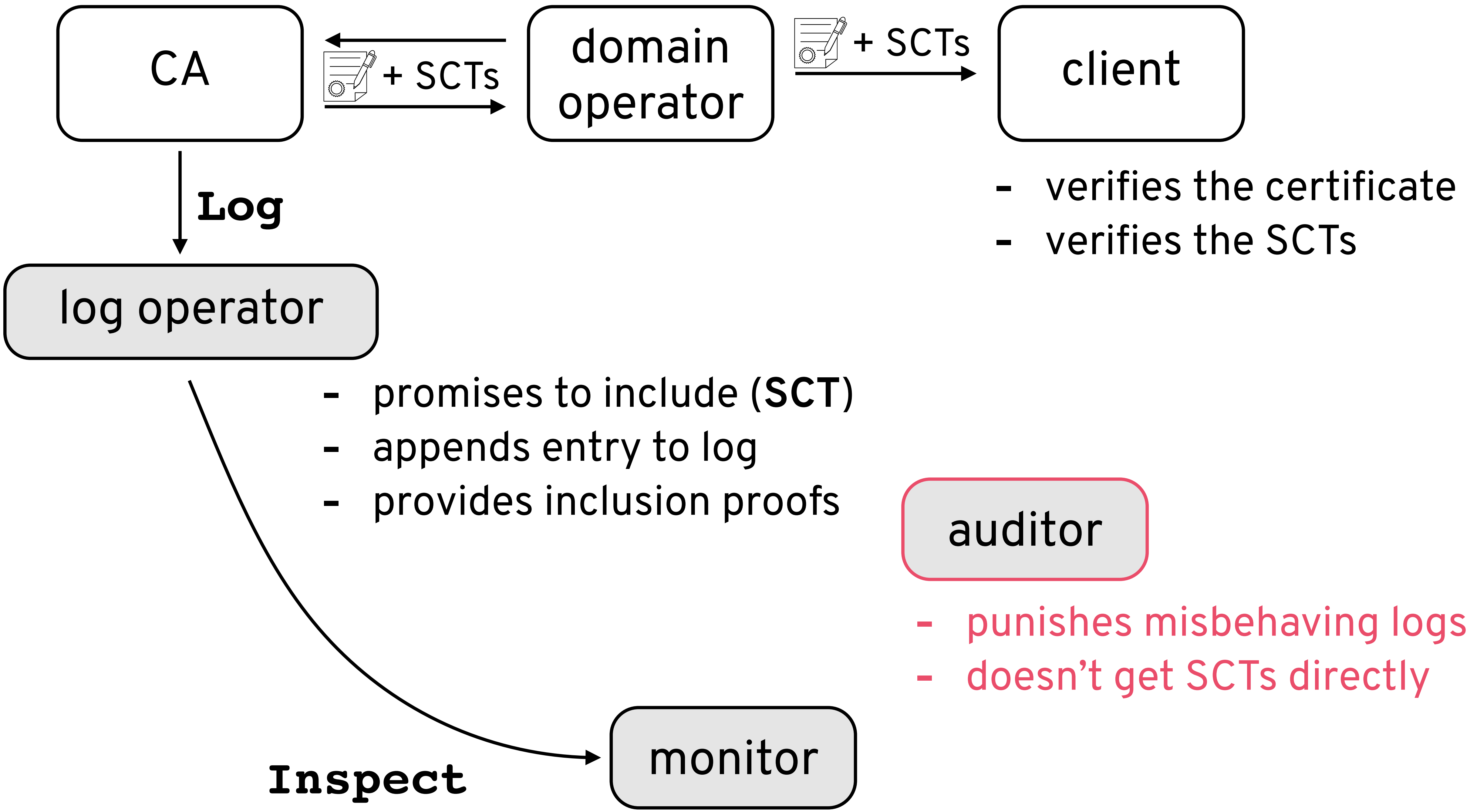
log operator

- promises to include (**SCT**)
- appends entry to log
- provides inclusion proofs

fetch inclusion proof after the SCT deadline, report if it fails to verify

clients are not in a position to report!

AUDITING AND REPORTING



CA

+ SCTs

domain operator

+ SCTs

client

- verifies the certificate
- verifies the SCTs

Log

log operator

- promises to include (**SCT**)
- appends entry to log
- provides inclusion proofs

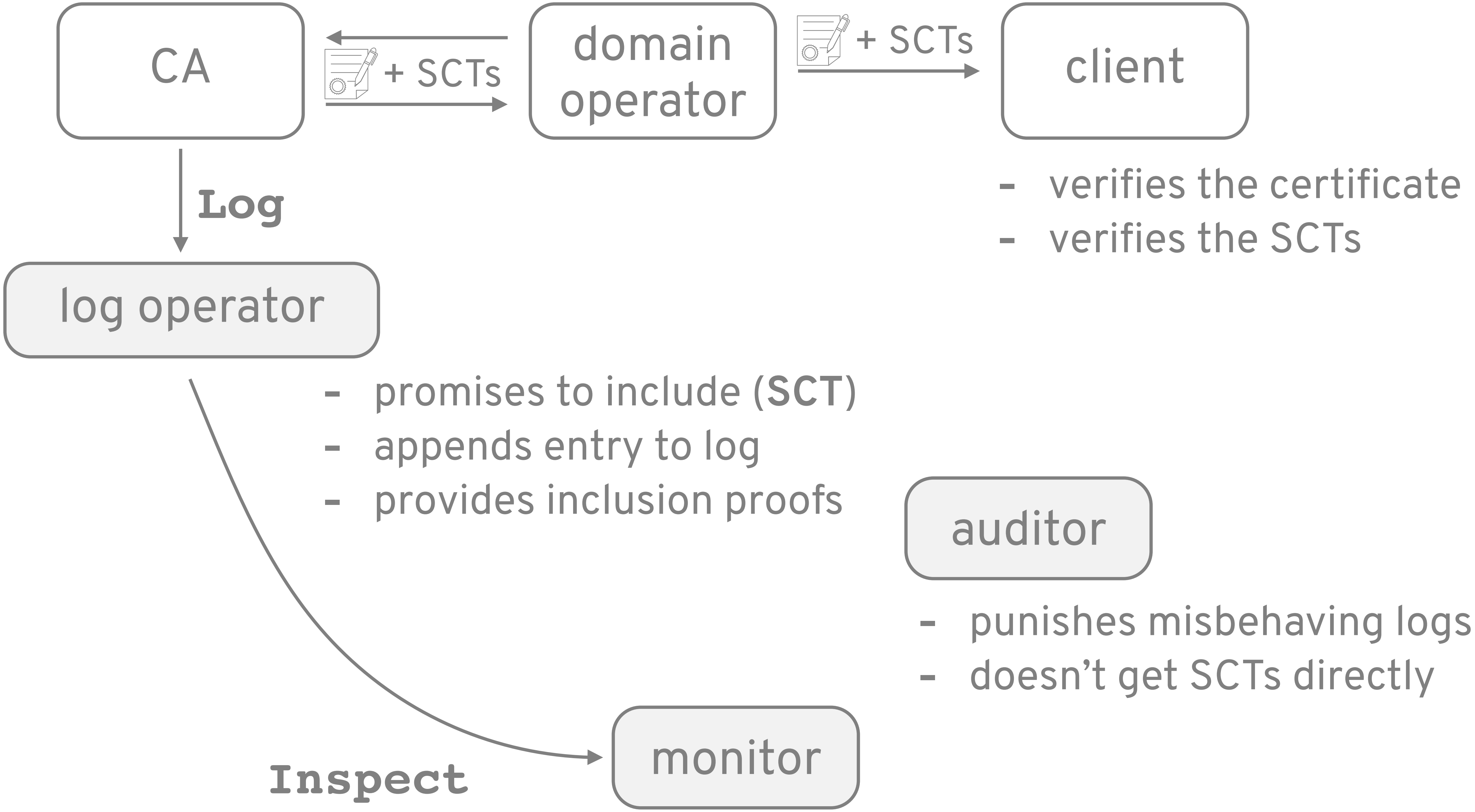
auditor

- punishes misbehaving logs
- doesn’t get SCTs directly

Inspect

monitor

SCT AUDITING



client:

- verifies the certificate
- verifies the SCTs

log operator:

- promises to include (**SCT**)
- appends entry to log
- provides inclusion proofs

auditor:

- punishes misbehaving logs
- doesn't get SCTs directly

HOW DO WE CHECK THAT CERTIFICATES SEEN BY CLIENTS ARE REALLY IN THE LOG, AND INFORM AN AUDITOR IF NOT?

GATHERING REQUIREMENTS

Consider querying and reporting phases

1. **Functionality**: Does it work?

GATHERING REQUIREMENTS

Consider querying and reporting phases

1. **Functionality**: Does it work?
2. **Privacy**: What information do which parties learn?
    1. No privacy
    2. k-anonymity
    3. Provable unlinkability

GATHERING REQUIREMENTS

Consider querying and reporting phases

1. **Functionality**: Does it work?
2. **Privacy**: What information do which parties learn?
3. **Client-side performance**
    1. Bandwidth
    2. Computation
    3. Storage

GATHERING REQUIREMENTS

Consider querying and reporting phases

1. **Functionality**: Does it work?
2. **Privacy**: What information do which parties learn?
3. **Client-side performance**
4. **Issuance latency**: How much longer does it take to issue a cert?

GATHERING REQUIREMENTS

Consider querying and reporting phases

1. **Functionality**: Does it work?
2. **Privacy**: What information do which parties learn?
3. **Client-side performance**
4. **Issuance latency**: How much longer does it take to issue a cert?
5. **Server-side performance**: What would it cost to run this?

GATHERING REQUIREMENTS

Consider querying and reporting phases

1. **Functionality**: Does it work?
2. **Privacy**: What information do which parties learn?
3. **Client-side performance**
4. **Issuance latency**: How much longer does it take to issue a cert?
5. **Server-side performance**: What would it cost to run this?
6. **Threat model**: What trust assumptions are needed?

GATHERING REQUIREMENTS

Consider querying and reporting phases

1. **Functionality**: Does it work?
2. **Privacy**: What information do which parties learn?
3. **Client-side performance**
4. **Issuance latency**: How much longer does it take to issue a cert?
5. **Server-side performance**: What would it cost to run this?
6. **Threat model**: What trust assumptions are needed?
7. **Near-term deployability**: Could this be deployed in 2-3 years?

GATHERING PROPOSALS

Identified proposals that were compatible with Certificate Transparency as it exists today in publications, experimental deployments, and posts on mailing lists

Also considered proposals for related problems of:

- Safe Browsing
- Checking for certificate revocation
- Checking for compromised credentials

OUTCOMES

Existing proposals don’t take into account all the dimensions of this problem:

- Certificates cannot contain sequence number (rapid issuance)
- Clients operate constrained devices
- Privacy degrades if each query achieves only k-anonymity
- Web servers are slow to change
- Clients need to report to an auditor in a private way

https://arxiv.org/pdf/2203.01661.pdf

THANKS!
ANY QUESTIONS?