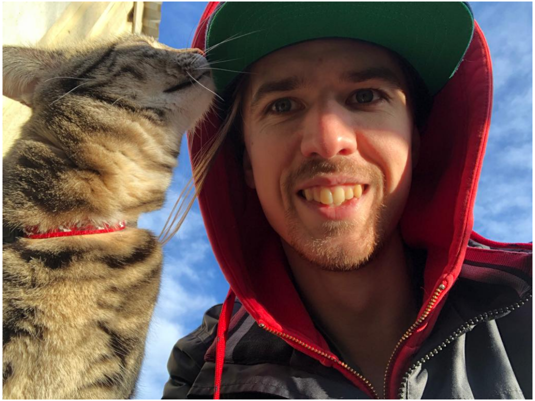
An Introduction to System Transparency Logging

May 7, 2021

Rasmus Dahlberg

Outline

1. Explore the problem area
2. A bird's view of the design
3. Revisit the problem area
4. How to get involved



Meet Daniel, the author of `curl`

- Digital signing using gpg
- Long-term RSA public key

Meet the R-B project

- Same input gives the same output
- Consensus of “valid” checksum?

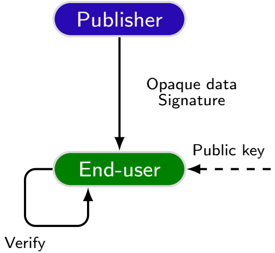
https://creativecommons.org/licenses/by-sa/4.0/

Problem summary

1. What signatures were produced by a given private key?
2. Consensus of checksums that should be considered valid?

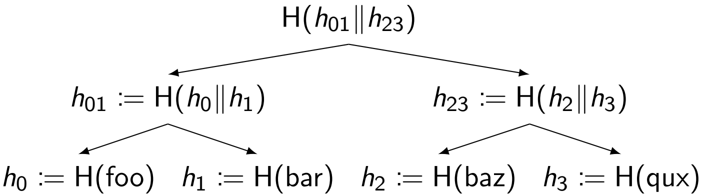
Our starting point

- Data publisher
- End-user
- Assumptions
    - Public key can be located
    - Signed data can be located
    - End-user can install extra tooling



The attacker can compromise the data publisher
The goal is to detect unwanted key-usage

A quick step back—Transparency log crash course



$H(h_{01} \| h_{23})$

$h_{01} := H(h_0 \| h_1)$ $h_{23} := H(h_2 \| h_3)$

$h_0 := H(\text{foo})$ $h_1 := H(\text{bar})$ $h_2 := H(\text{baz})$ $h_3 := H(\text{qux})$

- Tree head
- Consistency proof
- Inclusion proof

The attacker can control the log

Preparing a logging request

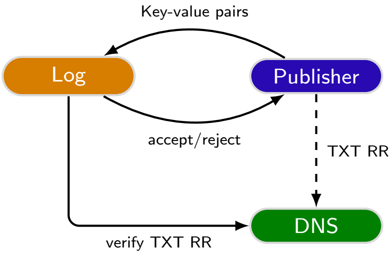
- Select a shard hint and checksum
- Sign using your private key

```
/*
 * The logged Merkle tree leaf data
 */
struct tree_leaf {
    u64 shard_hint;
    u8 checksum[32];
    u8 signature[64];
    u8 key_hash[32];
}
```

Submitting a logging request

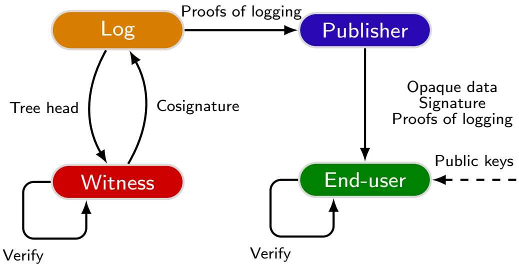
Key-value pairs:

- Shard hint
- Checksum
- Signature
- Public key
- Domain hint



Distributing proofs of public logging

- End-user will not talk to the log
- Proofs of logging
  - Inclusion proof
  - Tree head
- Witness cosigning



The attacker can control a threshold of witnesses

Summary and additional details

- Signed checksums
- Sharding
- Preserved data flows
- Anti-spam
- Global consistency
- Few simple parsers
- No cryptographic agility

Remember Daniel?

- (Cross-)sign with Ed25519
- Backwards compatible verification?
  1. Verify RSA gpg signature
  2. Verify the rest with tlog tooling
- Monitor the log for your own leaves

https://creativecommons.org/licenses/by-sa/4.0/

Remember the R-B project?

- Sign and log the expected checksums
- Valid checksum is a logged checksum
- Rebuilders monitor the log

Reproducible Builds

https://creativecommons.org/licenses/by-sa/4.0/

Get involved

- Feedback on our v0 design[1]and API[2]?
- Is this a service that you would use? Why (not)?
- Want to run an experimental log or witness?
- Implementation and tooling is still early-days
- Reach out via slack[3], GitHub, or DM

[1] https://github.com/system-transparency/stfe/blob/design/doc/design.md
[2] https://github.com/system-transparency/stfe/blob/design/doc/api.md
[3] https://communityinviter.com/apps/system-transparency/join