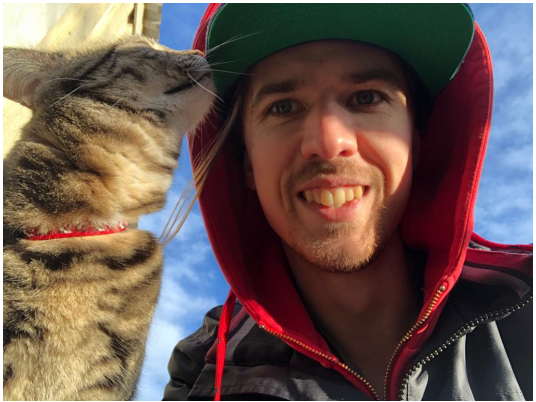
Transparent Logging—An introduction and ongoing work

October 15, 2024

Rasmus Dahlberg

Outline

1. Transparent Logging
   - Why?
   - How?
   - What?
2. “System Transparency Logging”



Let's travel in space and time

- June, 2011
- Netherlands, Beverwijk
- DigiNotar

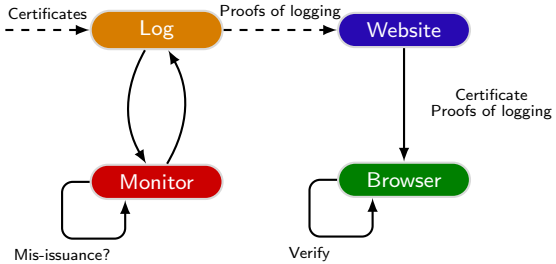
What happened?

- DigiNotar issued web certificates
- Did not live up to expectations
- Then lied about it for weeks

https://www.bbc.com/news/technology-14989334

What to make of this

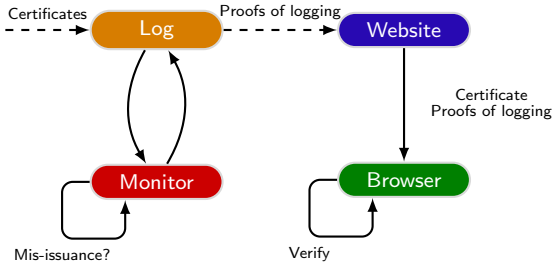
- DigiNotar was neither first nor last[1]
- Detection of certificate mis-issuance?
- Discoverability with transparent logs[2]



[1] https://sslmate.com/certspotter/failures
[2] https://certificate.transparency.dev/

What to make of this

- DigiNotar was neither first nor last[1]
- Detection of certificate mis-issuance?
- Discoverability with transparent logs[2]



Chrome and Safari enforce Certificate Transparency

[1] https://sslmate.com/certspotter/failures
[2] https://certificate.transparency.dev/

Transparency logging is good for more than just certificates

Source code
Binaries
Config files
TPM quotes
Media content
Tax declarations
Documents of ownership
BGP announcements
Tor's consensus
...

Transparency logging is good for more than just certificates

Source code
Binaries
Config files
TPM quotes
Media content
Tax declarations
Documents of ownership
BGP announcements
Tor's consensus
...

The log we are working on is helpful for all these use-cases!

Example use-case#1

Meet Daniel

- The author of `curl`
- Digitally signs new releases
- Long-term signing key-pair

Example use-case#2

Meet the R-B project

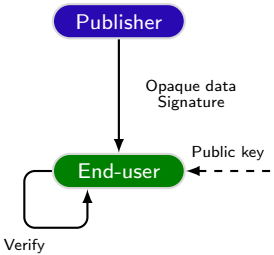
- Same input gives the same output
- Consensus of "valid" checksum?

Problem summary

1. Which signatures were produced by what private keys?
2. Consensus of checksums that should be considered valid?

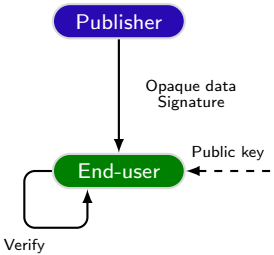
Our starting point

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



Our starting point

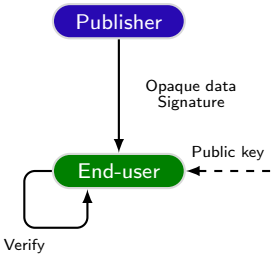
- Data publisher
- End-user
- Assumptions
  - Public key can be located
  - Signed data can be located
  - End-user can install extra tooling



The attacker can compromise the data publisher

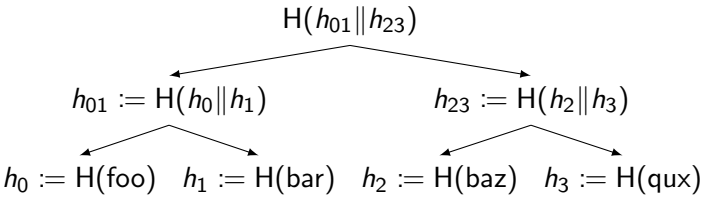
Our starting point

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



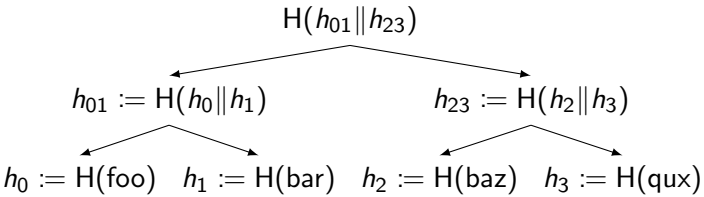
The attacker can compromise the data publisher
The goal is to detect unwanted key-usage

An intuition of transparency log properties



- Tree head
- Consistency proof
- Inclusion proof

An intuition of transparency log properties



- Tree head
- Consistency proof
- Inclusion proof

The attacker can control the log

Preparing a logging request

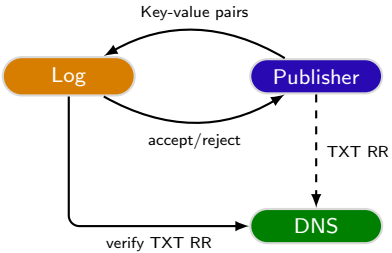
- Select a shard hint and checksum
- Sign using your private key

```
/*
 * 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

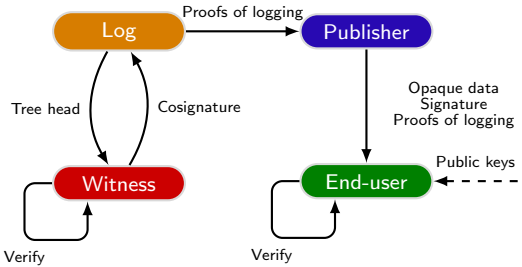
Key-value pairs:

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



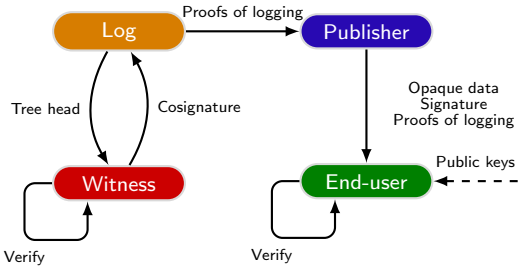
Distributing proofs of public logging

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



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

Example use-case#1

Remember Daniel?

- Sign a checksum of each `curl` release
- Start logging every signed checksum
- Monitor the log for your own leaves

Example use-case#2

Remember the R-B project?

- Sign the expected checksum of each build
- A valid checksum is a logged checksum
- Rebuilders validate logged checksums

Summary and feature overview

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

Current status

- Version v0 README and documentation[3]
- A public instance of the log is up and running
- At least one party is witnessing the log
- Come say hello and contribute if you want!
  - `irc/oftc #siglog`
  - Matrix bridge[4]
  - Open meetings every Tuesday, 1300

[1] https://github.com/system-transparency/stfe/

[2] https://app.element.io/#/room/#siglog:matrix.org