Google

Transparent Keyservers: Trust at Scale

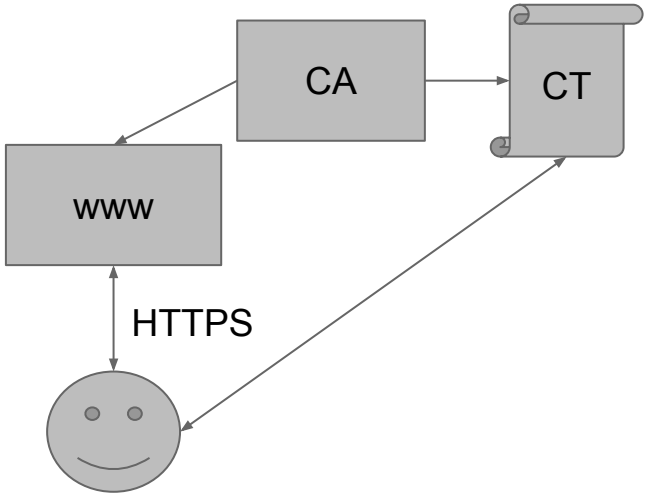
Gary Belvin

@gdbelvin

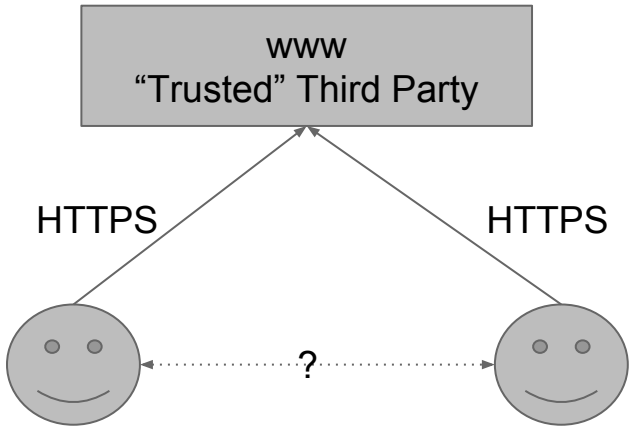
Preliminary work based on CONIKS and Certificate Transparency

Problem

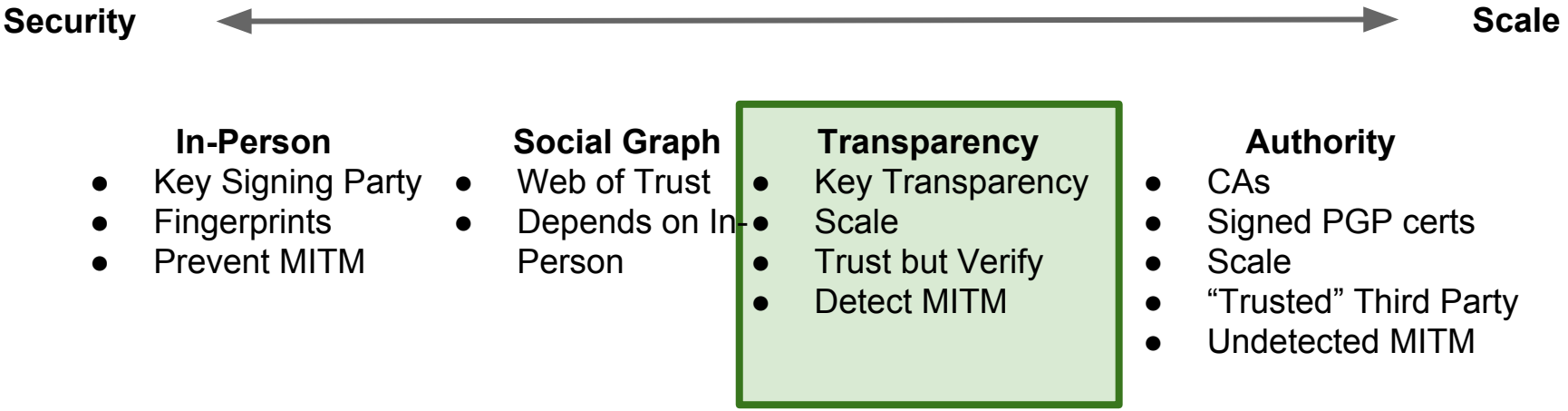
Websites have secure identities



Users don’t

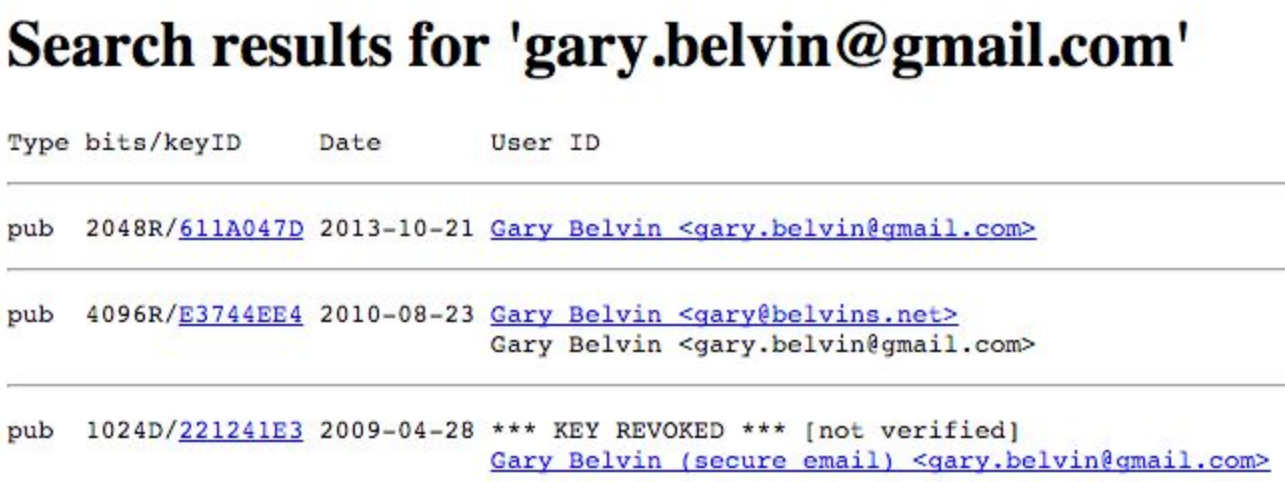


Spectrum of Identity Management



Key Lookup Today

which one should I use?

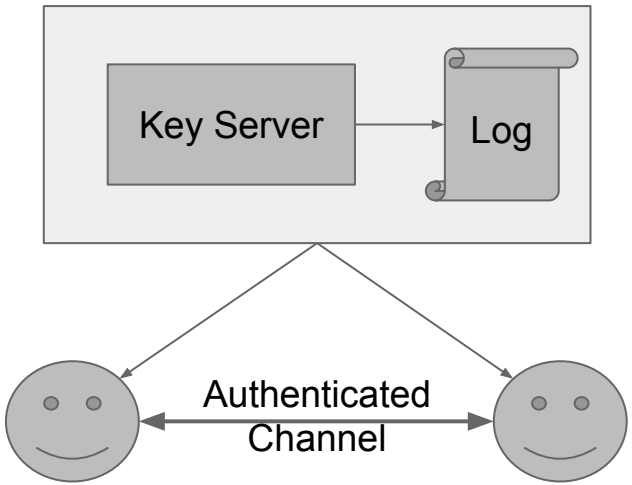


Search results for 'gary.belvin@gmail.com'

Type	bits/keyID	Date	User ID
pub	2048R/611A047D	2013-10-21	Gary Belvin <gary.belvin@gmail.com>
pub	4096R/E3744EE4	2010-08-23	Gary Belvin <gary@belvins.net> Gary Belvin <gary.belvin@gmail.com>
pub	1024D/221241E3	2009-04-28	*** KEY REVOKED *** [not verified] Gary Belvin (secure email) <gary.belvin@gmail.com>

A Solution

Key Server with Transparency for Users



Design Goals:

Cryptographic identities for all the things

Auditability

Privacy

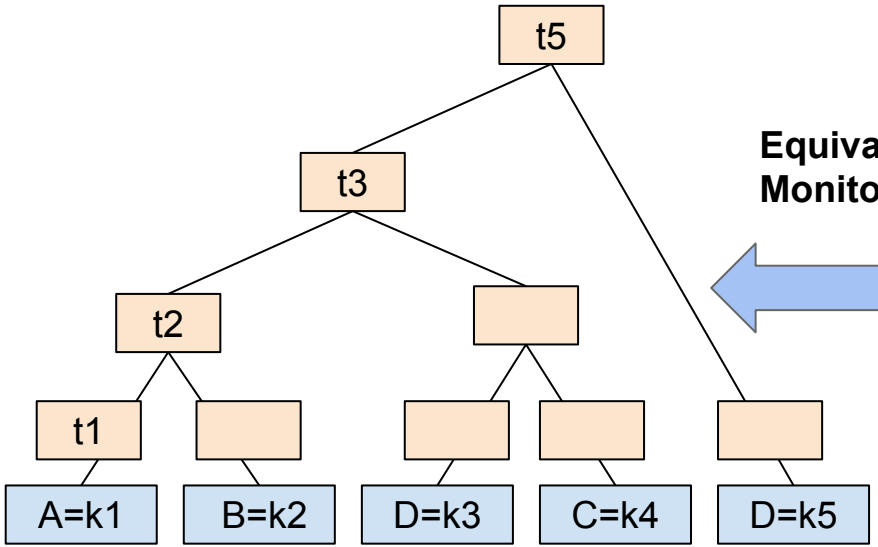
Consistent results

Where we’re going

- Secure identities for websites, and now users
- Cryptographic Logs
- Log of Changes
- Authoritative Entries
- Log of Identity History
- Privacy
- Federation

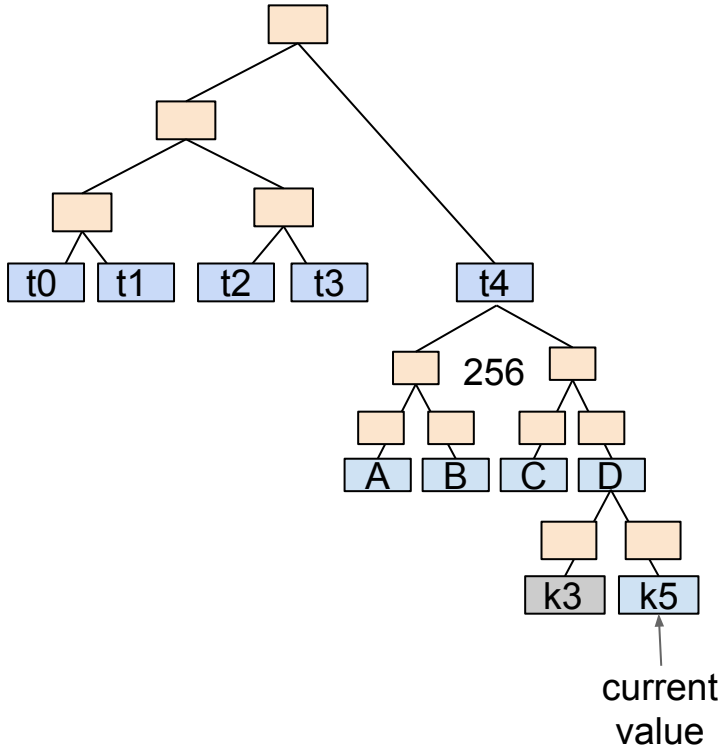
Let's Build This

Log of Changes



Equivalence Monitor

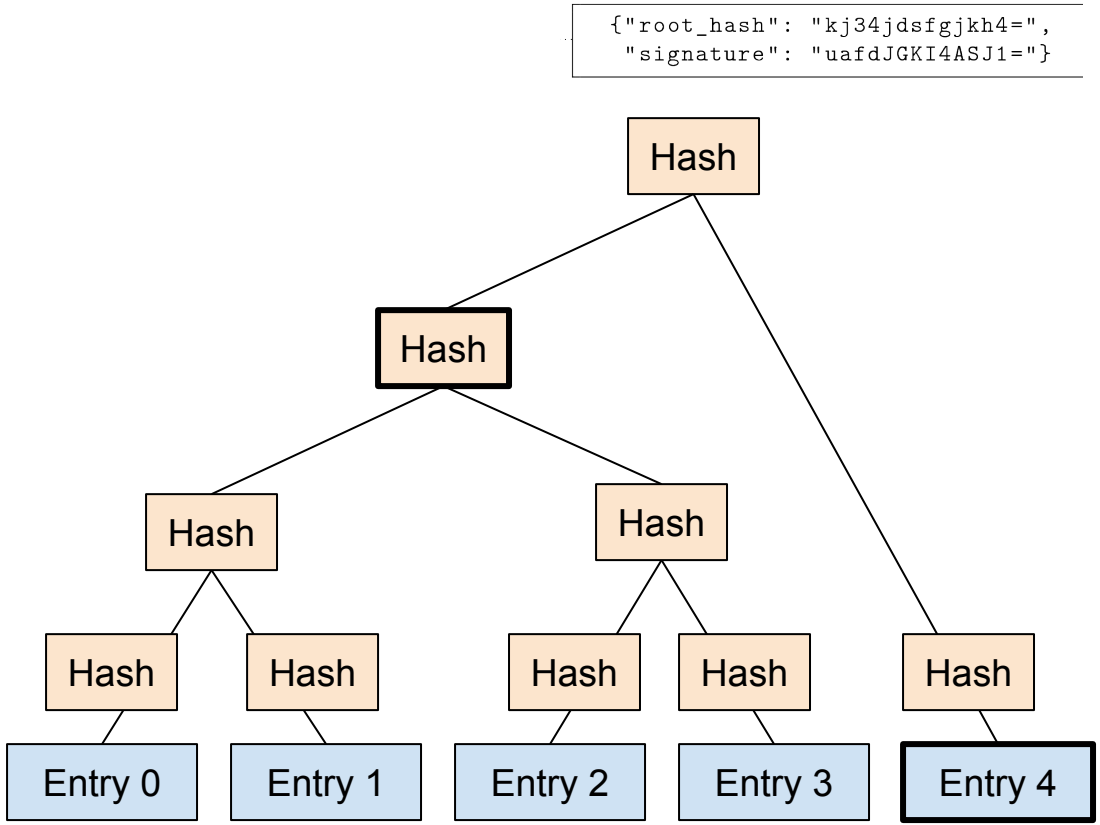
Log of Changing Identities with History



Verifiable Log

Efficiently Verify & Detect

- Split-view attacks
- Entry is included in the log
- Append-only
- Enumerate all entries



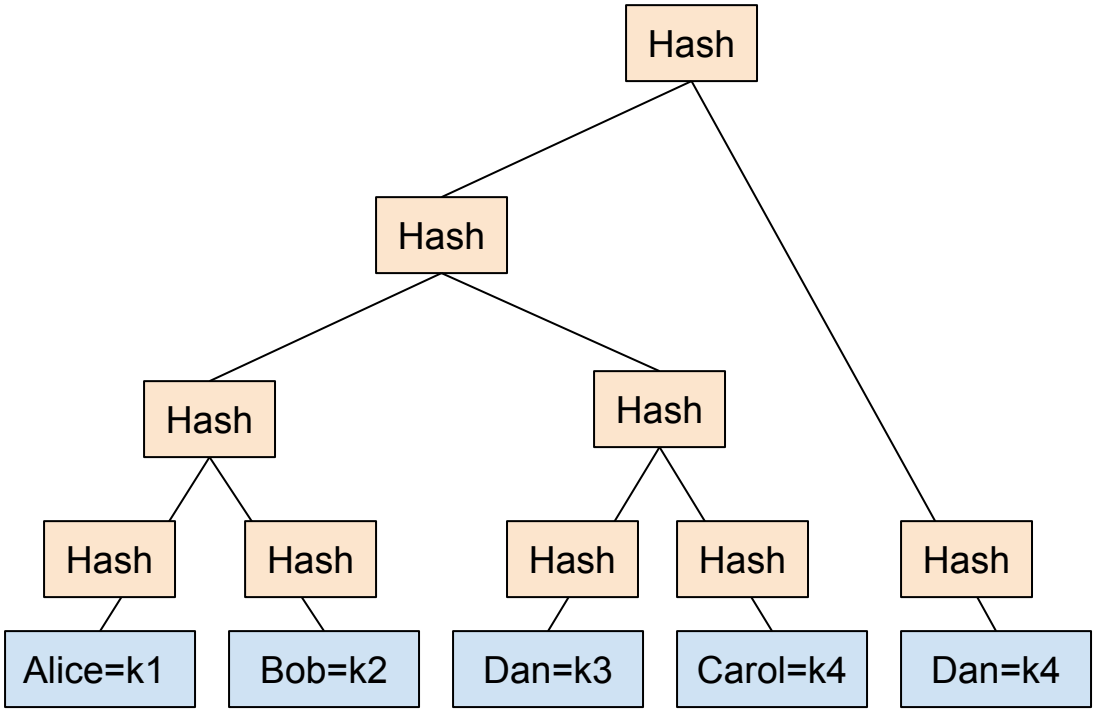
Verifiable Log of Changes

Efficiently Verify & Detect

- Entry is included in the log
- Split-view attacks
- Append-only
- Enumerate all entries

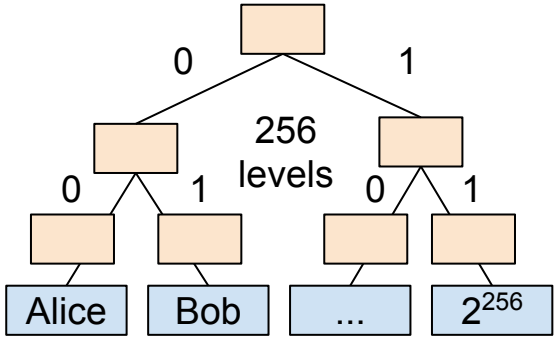
Build a Key Value Map

- Enumerate all changes
- In chronological order



Sparse Merkle Tree

- Key value mapping
- Entry is included in the tree
- Single value for each user at H(user_id)
- Proof of absence
- Tree size of 2^{256}



Prefix Tree Optimization

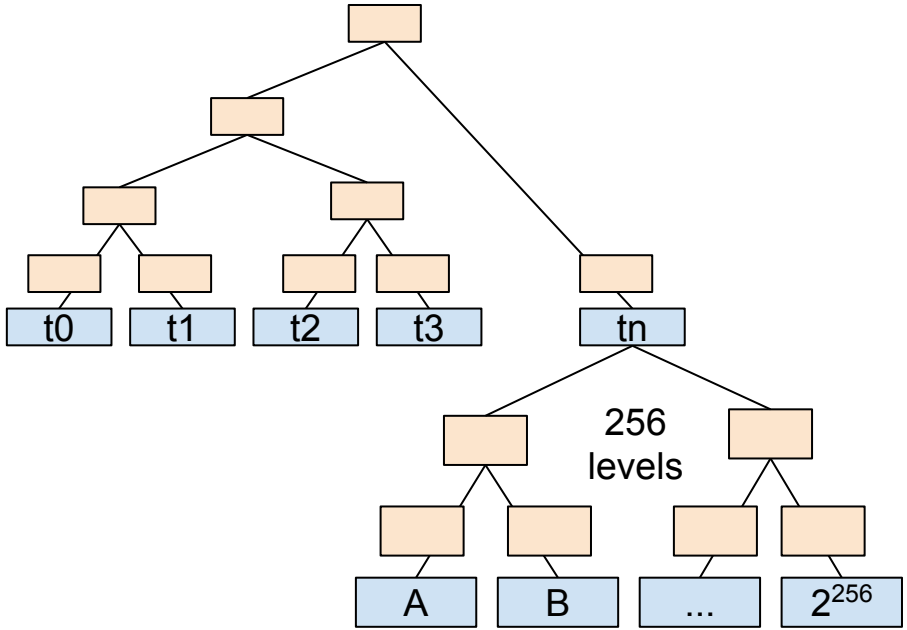
- Compress empty paths
- Longest path is O(log_2(N))

Verifiable Log of Sparse Merkle Tree Heads

Key, value mapping that can be **updated**

Monitor

- Verifies mapping between log of changes and sparse merkle tree heads.



Sparse Merkle Tree of Values over Time

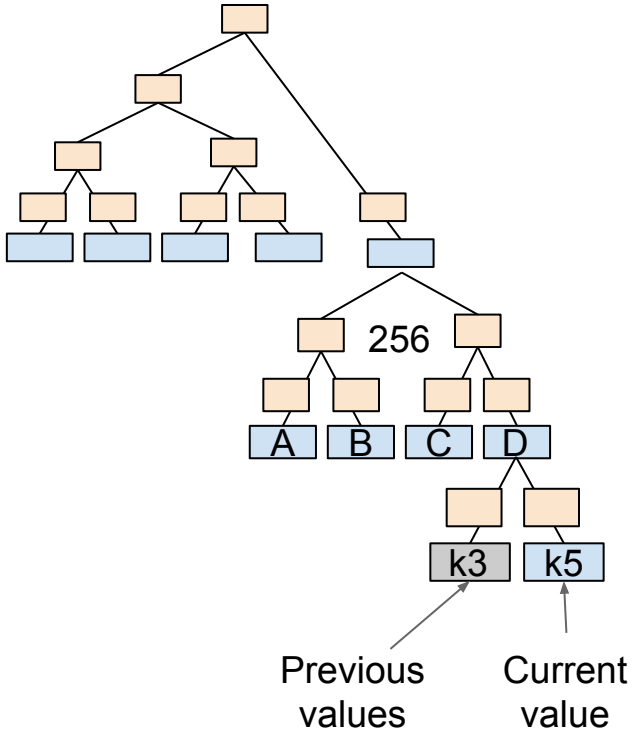
Key, value mapping that can be **updated**

Efficiently Verify

- Enumerate all values for a key
- “Has my key ever been compromised?”

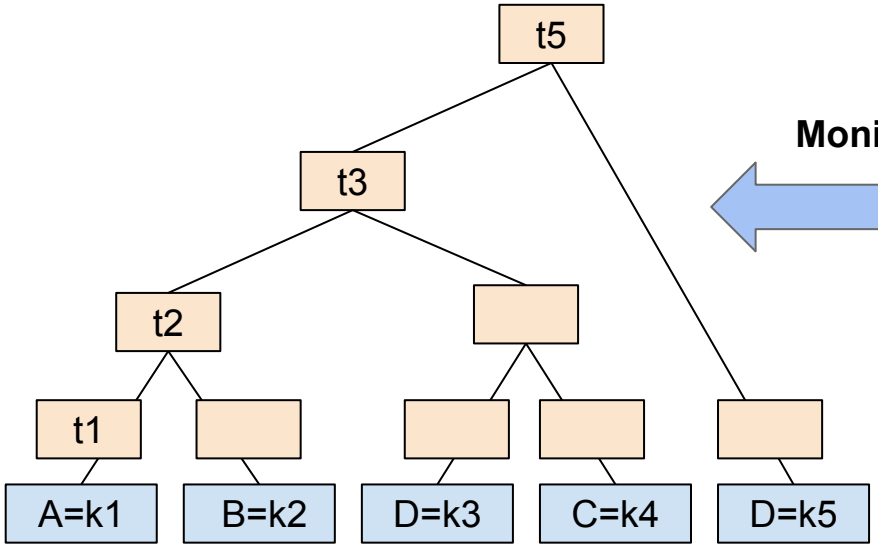
Monitor

- Verifies that all changes for a key are present in the sparse merkle tree entry.



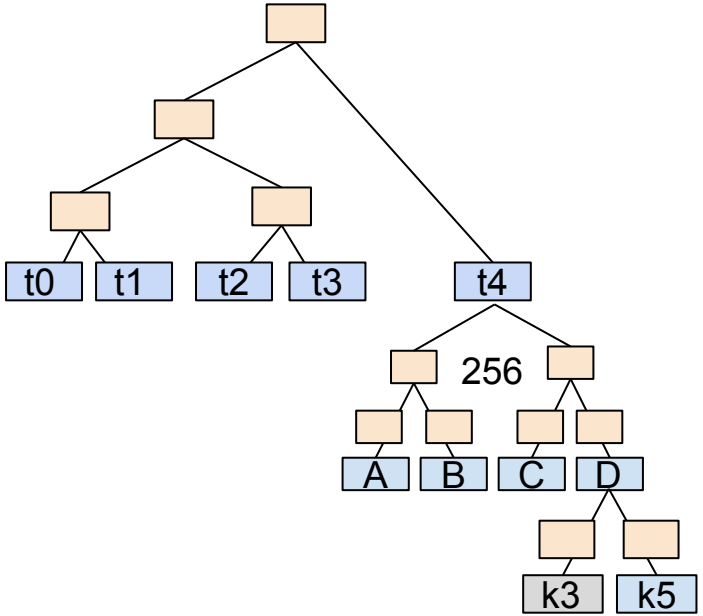
All Together

Log of Changes



Monitor

Log of Changing Identities with History



Spam & Abuse Controls

- Online lookup for location in tree
- H(user_id) would leak user_ids
- H(Deterministic signature)
- Online lookup for key
- Keys can contain PII
- Store commitments to keys

Federation

UserID to provider resolution

- @domain responsible for identity
- Fallback to ordered list of providers

Provider discovery

- List of known providers in app