Key Transparency: Problem Statement

Brendan McMillion

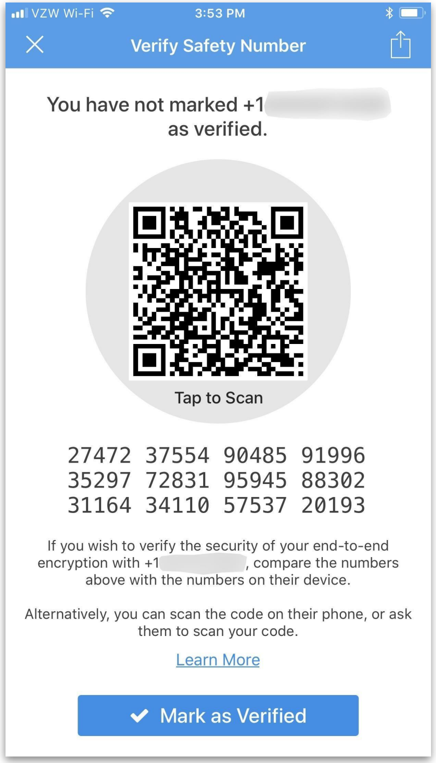
IETF 116 / March 29, 2023

Problem

- E2EE service providers often have difficulty finding secure ways to distribute the long-term identity keys of end-users

- Users can sometimes manually verify the public key of each user they communicate with (but people rarely actually do this)

- Compromised key management can undermine any encryption



VZW Wi-Fi 3:53 PM

Verify Safety Number

You have not marked +1[redacted] as verified.

Tap to Scan

```
27472 37554 90485 91996
35297 72831 95945 88302
31164 34110 57537 20193
```

If you wish to verify the security of your end-to-end encryption with +1[redacted], compare the numbers above with the numbers on their device.

Alternatively, you can scan the code on their phone, or ask them to scan your code.

Learn More

✓ Mark as Verified

Solution: Key Transparency

`From bofreq:`

“Key Transparency (KT) is a safe, publicly-auditable way to distribute cryptographically-sensitive data like public keys.”

`Works like a key-value database with two main, cryptographically-assured properties:`

1. Alice’s key as seen by Alice = Alice’s key as seen by everyone else
2. Alice’s key today = Alice’s key yesterday + Anything new

Current approach:
Users manually verify that a public key belongs to a specific, real life person

→

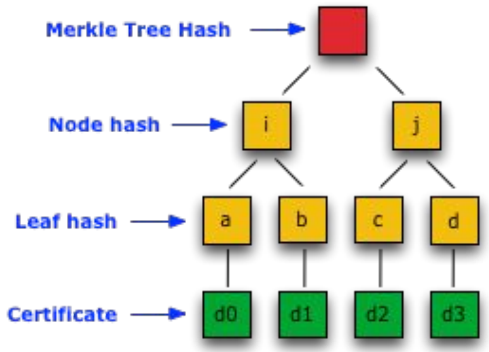
Key Transparency approach:
A user’s device monitors their account for unexpected changes that could be impersonation

Relation to other IETF efforts

Many WGs rely on “transparency logs” in their work:

- SCITT (Supply Chain Integrity)
- TIGRESS (Digital Credentials)
- **TRANS (Certificate Transparency)**

Built as fully public, append-only logs:



Merkle Tree: Each leaf contains the hash of some data. Every other node contains the hash of its children.

KT builds on top of append-only logs to provide:

- Efficient search / users don’t need to download the entire log
- Better privacy properties

Much more appropriate for E2EE!

This all sounds great but why are you telling me?

Key Transparency has relatively little serious adoption – why?

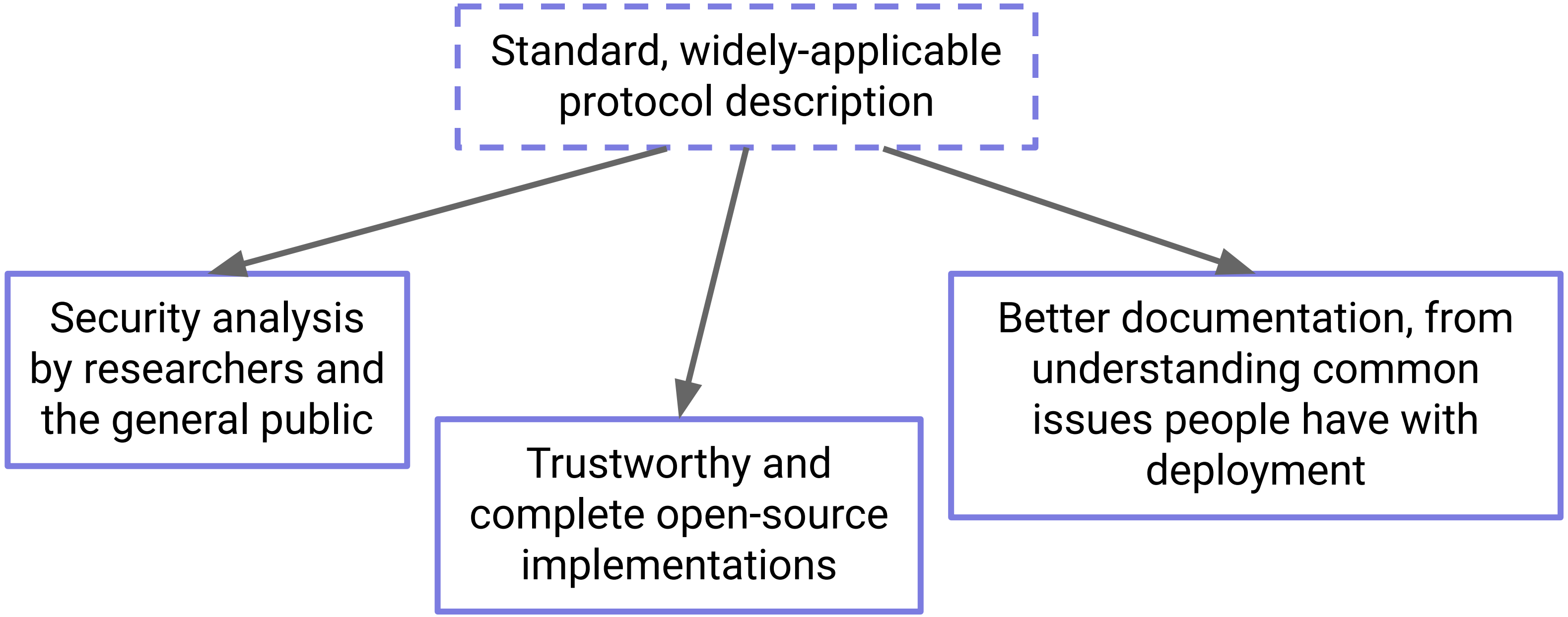
`Deploying KT is incredibly difficult:`

- Very technically complicated
- Large amount of academic literature
- No guidance on what the “right” choices in the design space are
- Few existing implementations, and those that exist often leave important aspects unresolved
- **No trusted, one-size-fits-all protocols or implementations**

Even very dedicated implementers get overwhelmed and give up*

* Or their manager tells them to stop

Ideal End Goal


Standard, widely-applicable protocol description
Security analysis by researchers and the general public
Trustworthy and complete open-source implementations
Better documentation, from understanding common issues people have with deployment

Actually Getting There

YOU ARE HERE

Understand the state of what’s been deployed and what’s possible → Align a community on a set of common, achievable requirements → Write a protocol that achieves those goals

Questions?
Thoughts?