

So, you want to deploy Key Transparency?

Brendan McMillion
Transparency.dev Summit 2026

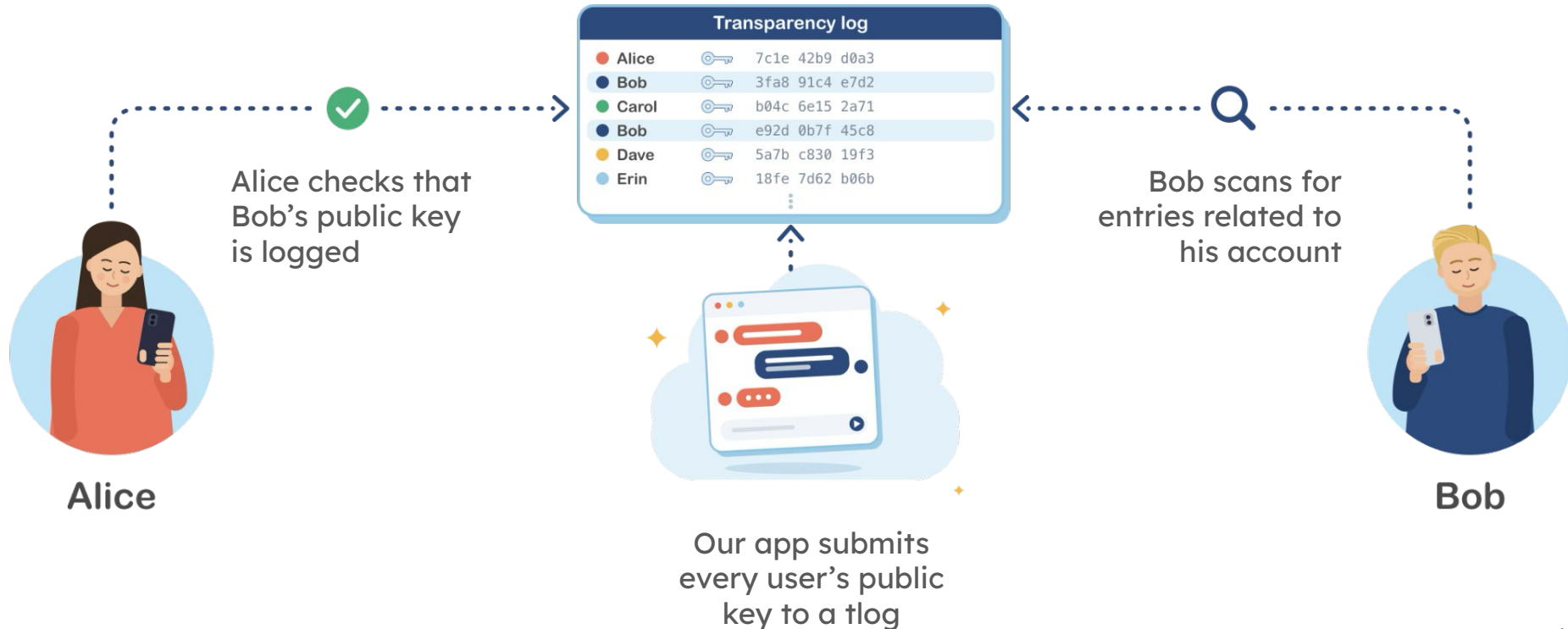
Background



The Problem



Don't tlogs solve this?



Don't tlogs solve this? Not really



What have people been doing instead?

Historically, apps took standard tlogs and built **custom solutions** for:



Fast User Monitoring

Lightweight mechanisms for users to check that their keys are correct



User Privacy

Hiding raw usernames and public keys, in addition to *patterns* of user behavior



Provider Privacy

Size of and activity level of user base as a whole



Trust Model

Is there a trusted third party?
Does there need to be? How are forks prevented?



Pruning & Compliance

Deleting data that's old or from former users without reducing security for current users



Anonymity

Letting users verify proofs without revealing who they're talking to

How to draw an owl

1.



2.



1. Draw some circles

2. Draw the rest of the fucking owl

Which brings us to the IETF...

Key Transparency ^{had} ~~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

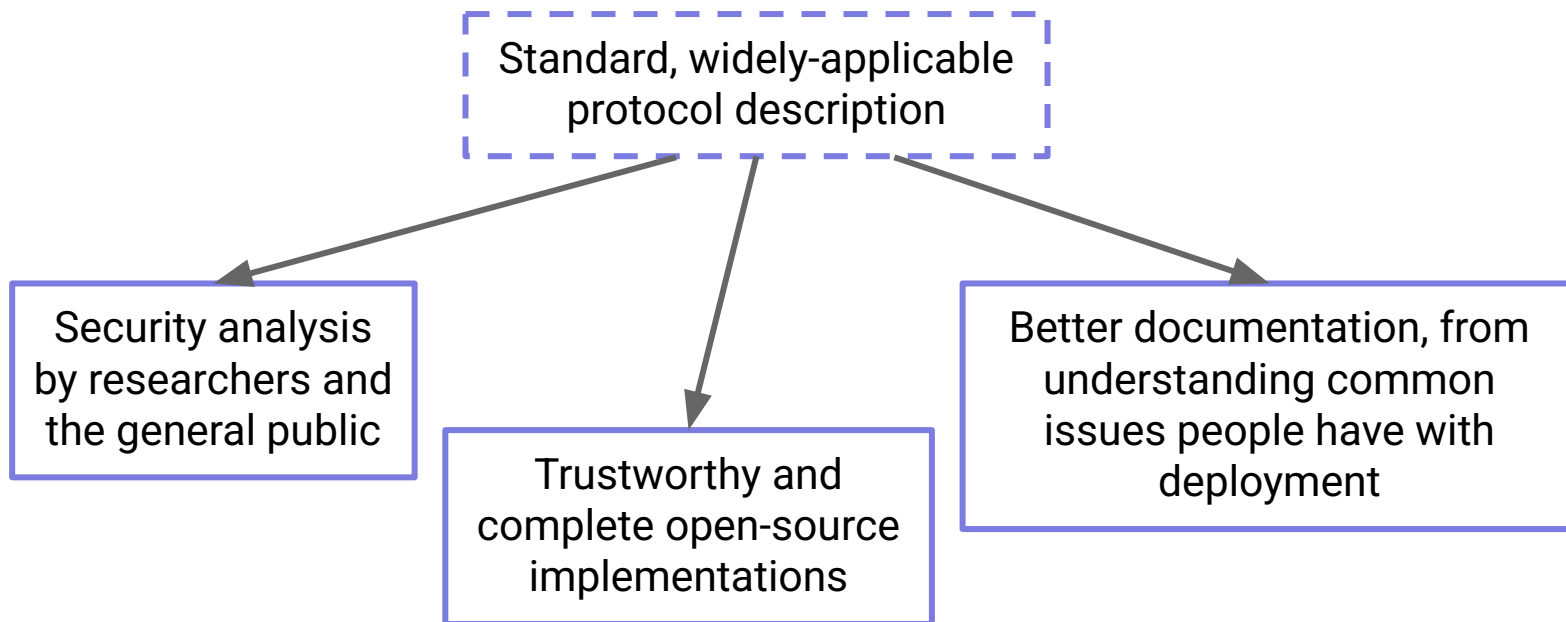
Aside: Trusted Third Parties

- When standardization started, there was **ONE** (recently published!) paper that built KT without any trusted third party
- Requiring a third party is often the *primary* blocker to KT deployment
- But the protocol in that paper didn't address many, many other important deployment problems (user privacy, pruning, anonymity, ...)

Merkle²: A Low-Latency Transparency Log System

Yuncong Hu, Kian Hooshmand*, Harika Kalidhindi*, Seung Jin Yang*, Raluca Ada Popa
University of California, Berkeley
{yuncong_hu, kianhooshmand, jrharika, seungjin, raluca.popa}@berkeley.edu

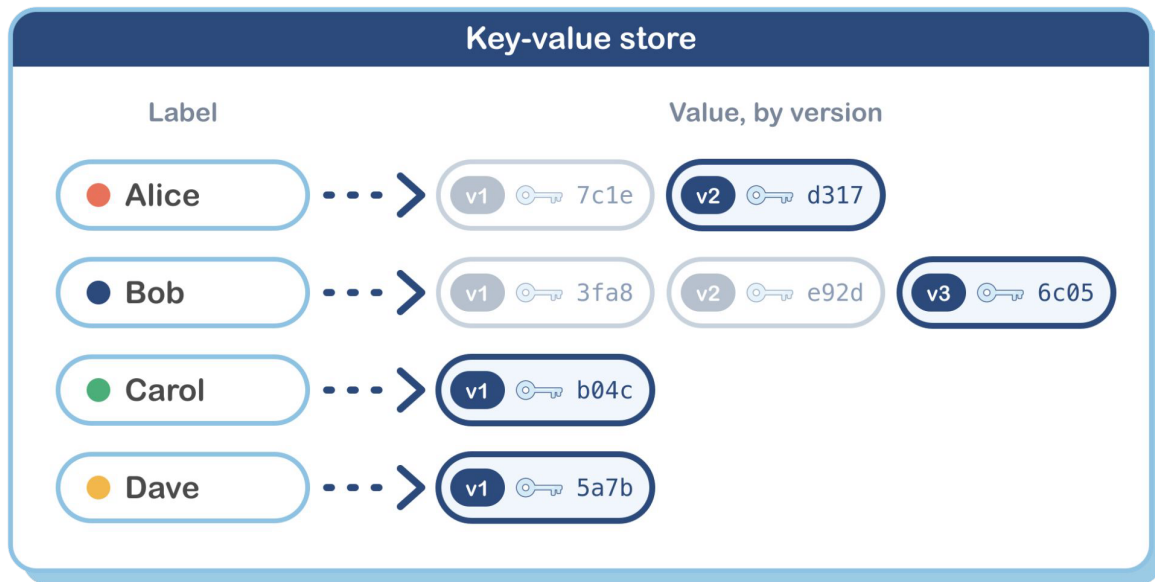
Goals of Standardization



How do I use the IETF KT protocol?

High-Level Abstraction

To apps, a KT server looks exactly like a *versioned* key-value store:



High-Level Operations

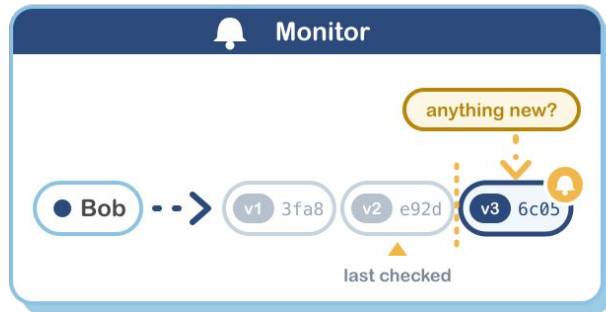
Search returns either the greatest version or a specific version of a label.



Update creates the next version(s) of the label.

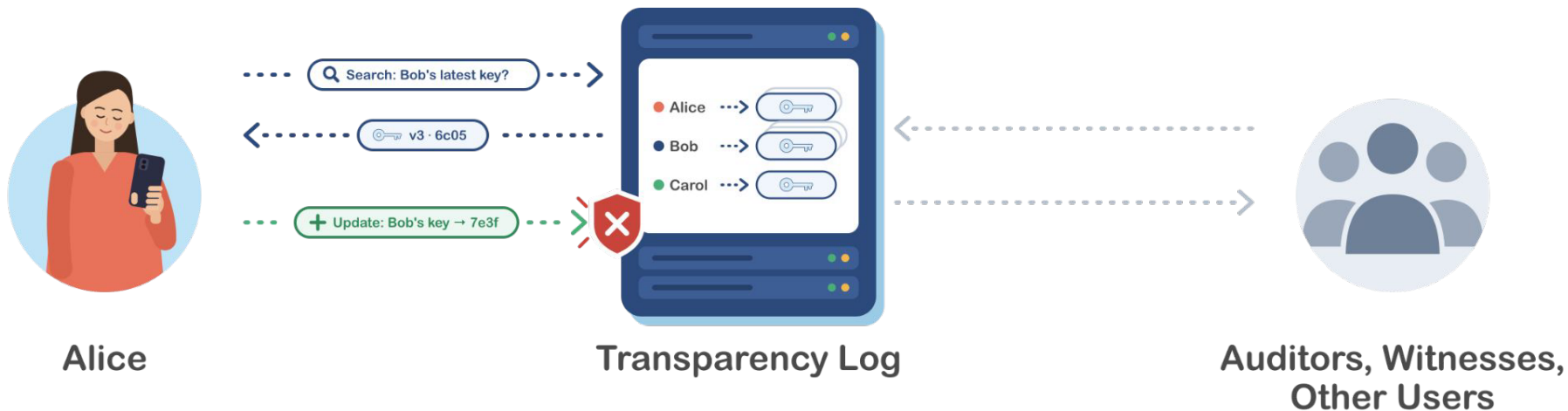


Monitor efficiently informs users when new versions of a label exist



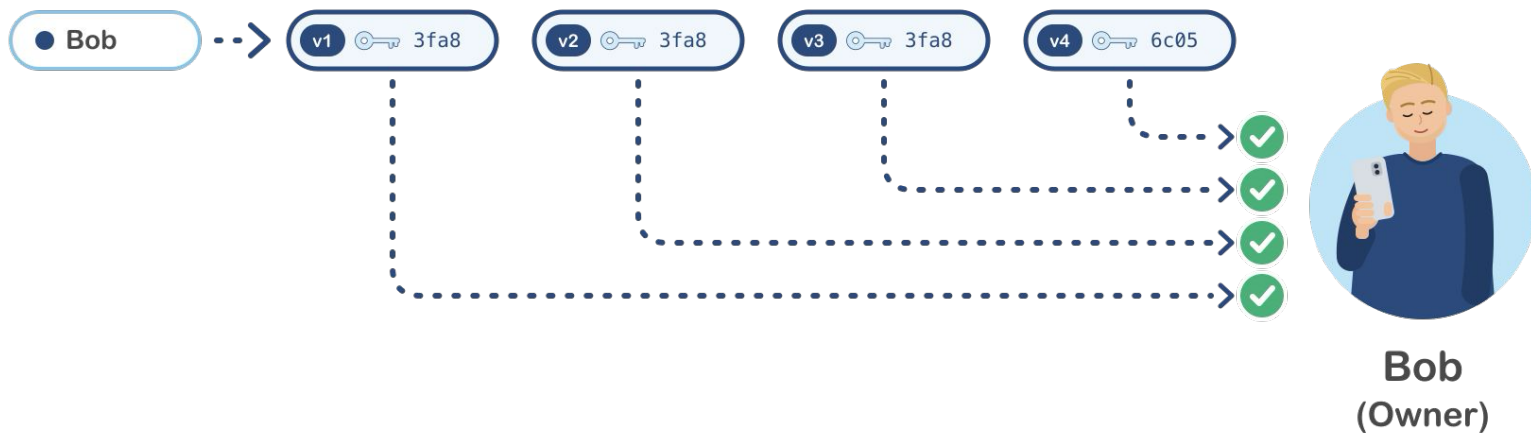
Communication Model

- Users only need direct communication with the Transparency Log.
- Access control is enforced at the “transport” layer



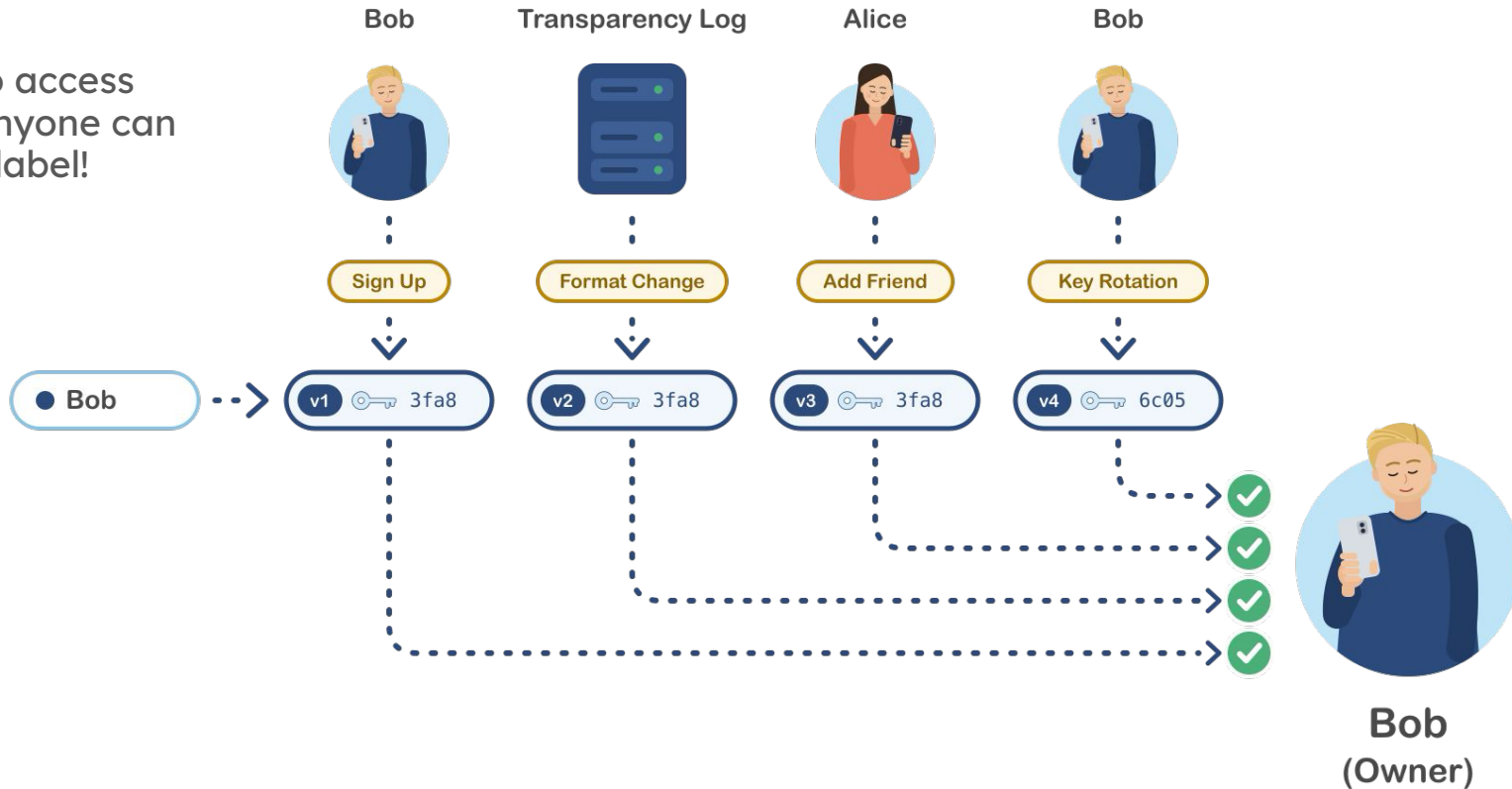
Ownership

A label's **owners** are the users who decide whether the label's value is acceptable or not.



Ownership Allows Collaboration

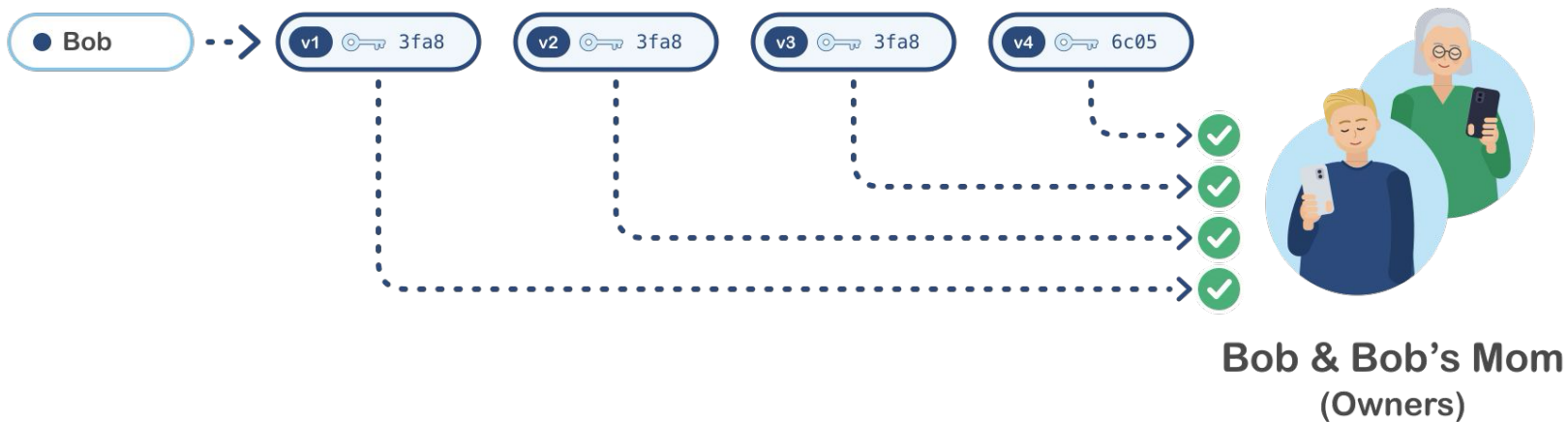
Subject to access control, anyone can modify a label!



Ownership is Not Exclusive

Strictly speaking, ownership is just a protocol mechanism for seeing all versions of a label.

Any number of users can own a label.



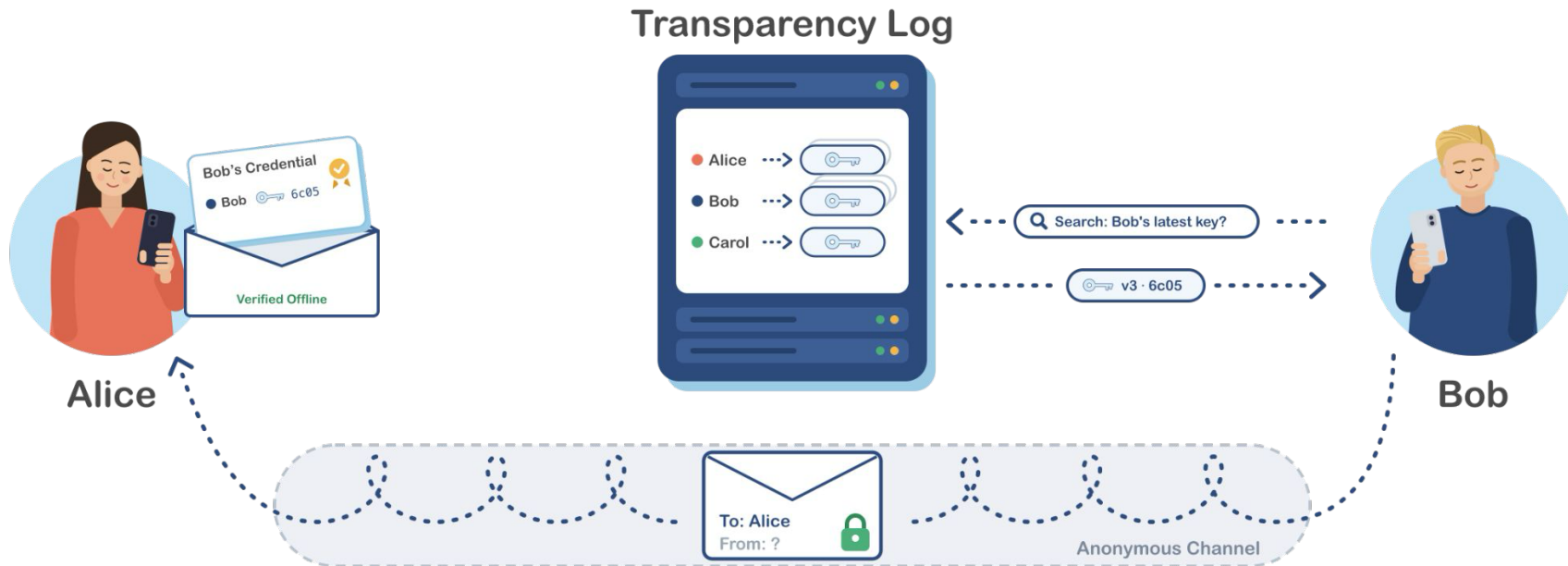
⇒ Platform for E2EE Apps

Application	Label	Value	Label Owner(s)
User Profiles	Long-Term User ID	v3  Public Key, Profile, Contact Info, ...	User
User Discovery	Short-Term User ID	v1  Long-Term User ID	User
MLS Groups	Group ID	v5  MLS GroupInfo	Current Group Members
Security-Critical Configuration	“global config”	v2  Global configuration	Service Provider, Third-Party Auditors

* Several labels can be modified atomically, which allows consistent relationships between labels!

Credentials

- Credentials are Search operations that can be verified against local state
- Allows **anonymous** applications, since users don't have to directly query the Transparency Log about their contacts

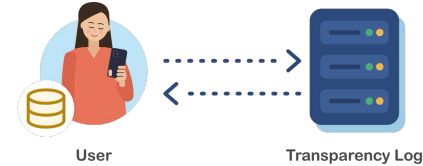


**What do I need to know about
operating a KT log?**

Deployment Modes

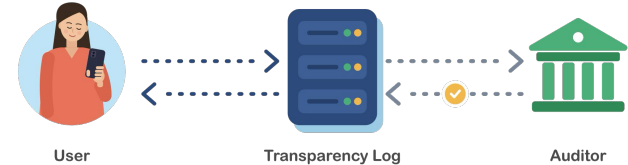
Contact Monitoring

Clients retain a small amount of state to ensure correct operation. Requires some way to detect forks: anonymous channel, p2p communication, witness.



Third-Party Auditing

A trusted third party auditor crawls the log in the background and signs the Transparency Log's head if it is behaving honestly.



Third-Party Management

The Transparency Log is operated by the trusted third party and the Service Operator sits in front to enforce access control and prevent forks.



User Privacy

Service Providers enforce per-label access control, with these guarantees:

- Users that have never been allowed to lookup a label have no information on whether it exists or not
- Users that were *previously* allowed to lookup a label have no information on whether new versions of it exist or not

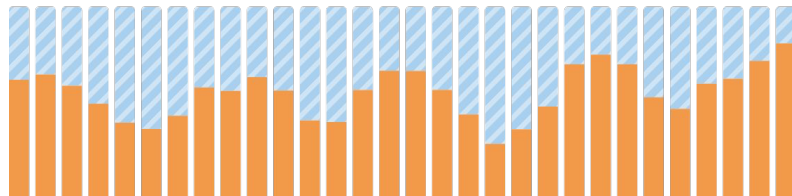
Provider Privacy

Two primary metrics:

- Total number of users
- Avg activity level of users

Service Providers pick an upper bound that they can reveal and insert “padding” entries over real user activity to match.

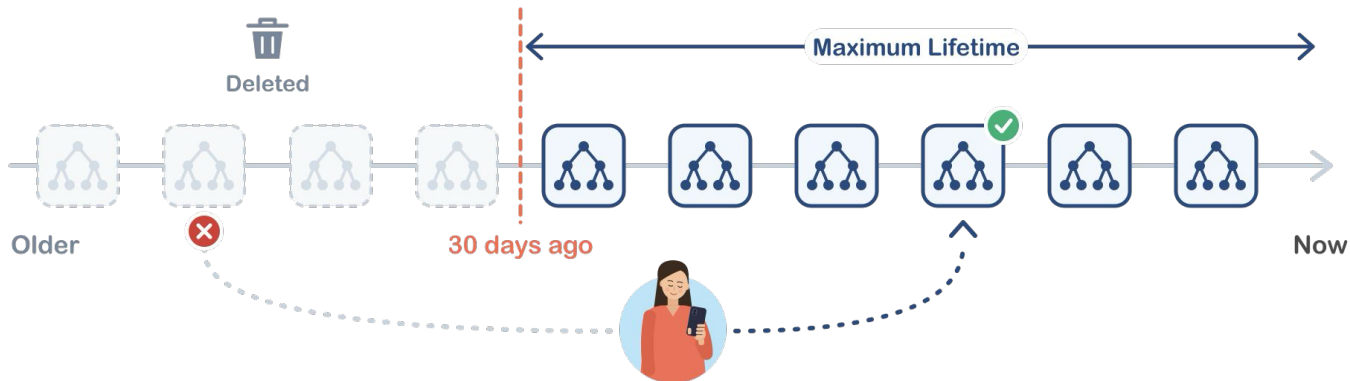
To outsiders, padding activity is indistinguishable from real user activity.



Pruning

Maximum Lifetime: (optional configuration) The oldest version of the Transparency Log that a client is allowed to request

- KT servers can freely delete anything older than the Max Lifetime
- *However*, users that are offline longer than the Max Lifetime have to reset their local state, which might weaken security



Metrics (YMMV)

Proof Sizes:

- Greatest-Version Search: 13 - 22 kB
- Standard Credential: 2 - 8 kB
- Owner Monitoring: 5 - 10 kB per day

Write Throughput: >4,000 modifications per second possible (DynamoDB)

Recap



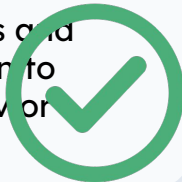
Fast User Monitoring

Lightweight mechanisms for users to check that their keys are correct



User Privacy

Hiding raw usernames and public keys, in addition to *patterns* of user behavior



Provider Privacy

Size of and activity level of user base as a whole



Trust Model

Is there a trusted third party?
Does there need to be? How are forks prevented?



Pruning & Compliance

Deleting data that's old or from former users without reducing security for current users

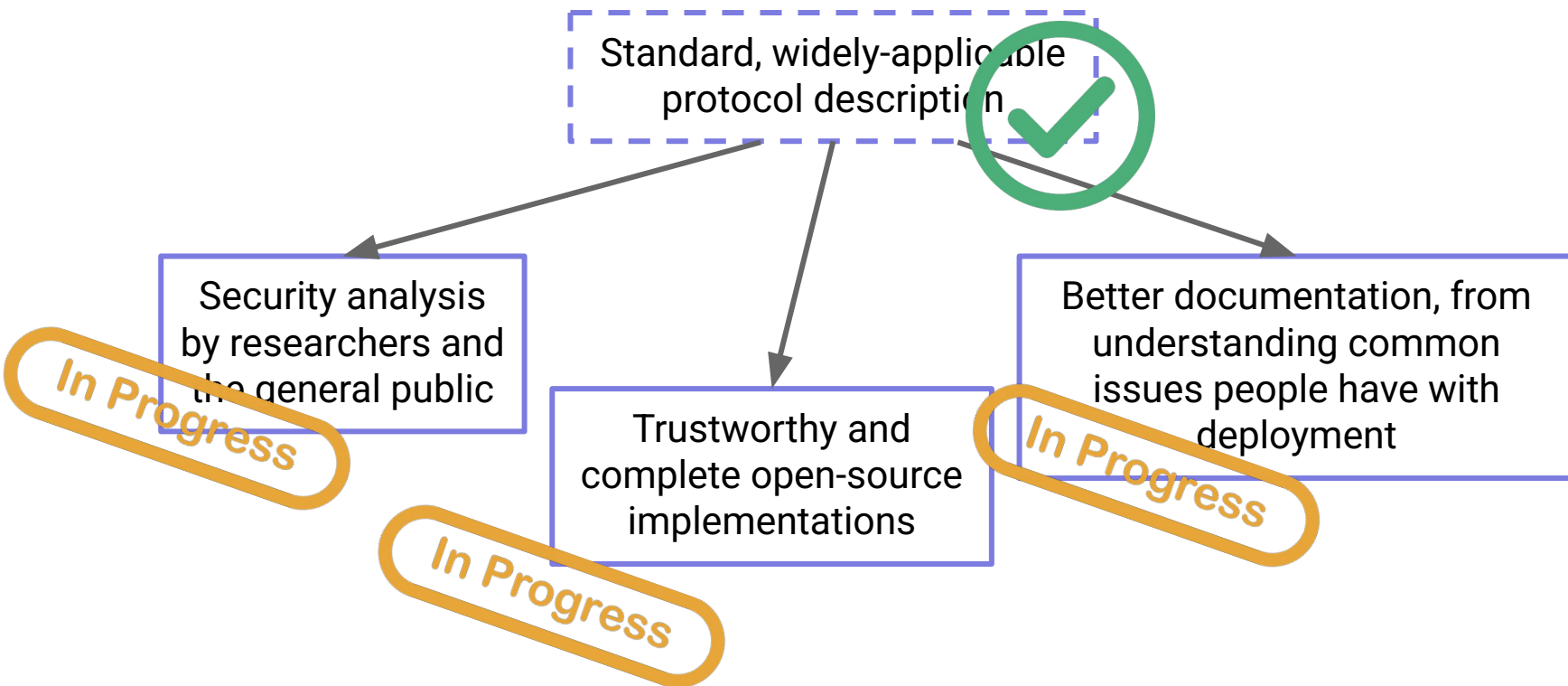


Anonymity

Letting users verify proofs without revealing who they're talking to



Goals of Standardization



That's all, thanks :)

IETF KEYTRANS working group:

[https://datatracker.ietf.org/wg/keytrans/about/
keytrans@ietf.org](https://datatracker.ietf.org/wg/keytrans/about/keytrans@ietf.org)

Interested in working on the spec / implementing it /
deploying it? Reach out!

brendanmcmillion@gmail.com