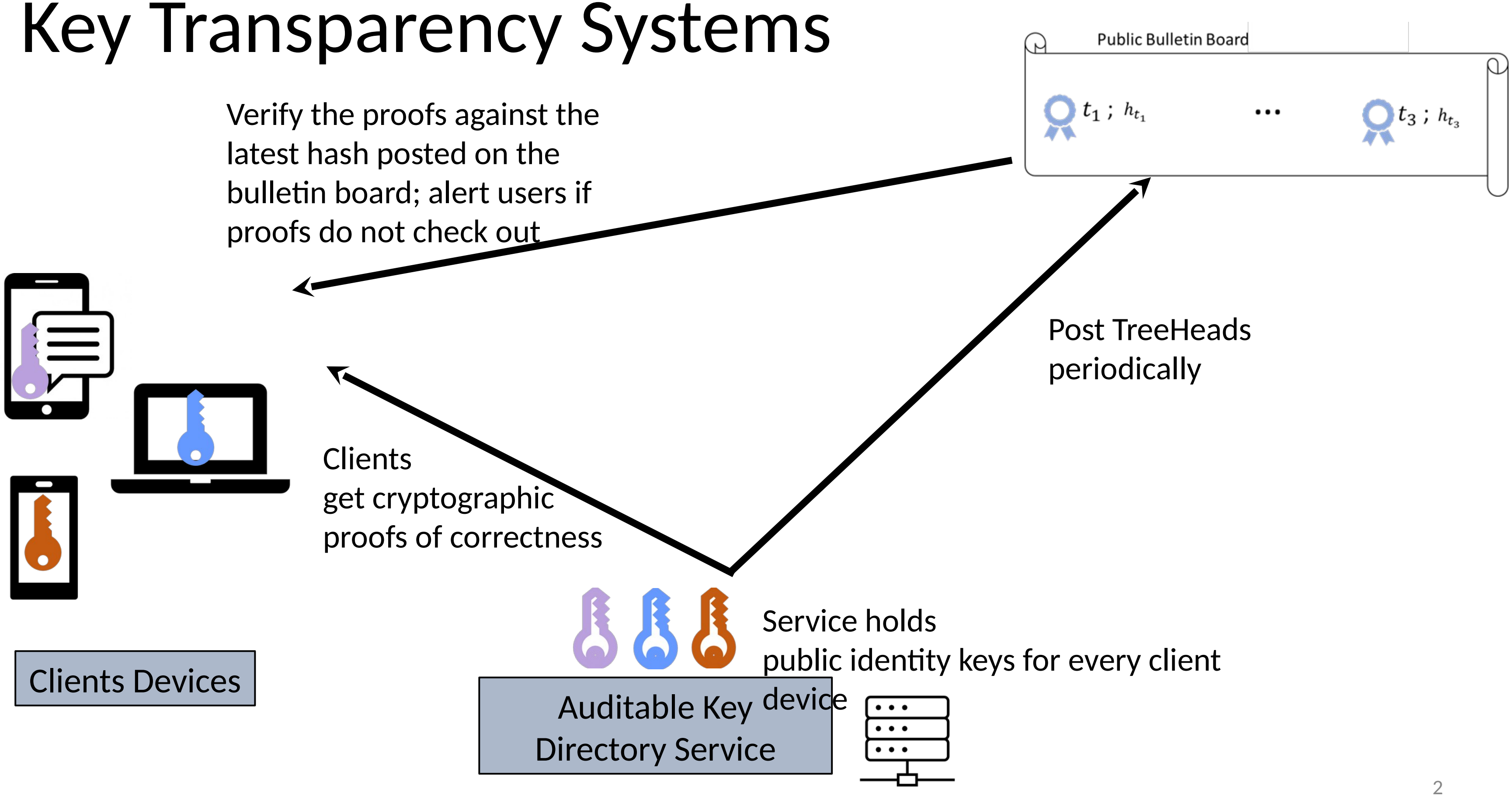
Understanding Security and Privacy Properties of Key Transparency

Esha Ghosh
Microsoft

Kevin Lewi
Meta

Key Transparency Systems



Verify the proofs against the latest hash posted on the bulletin board; alert users if proofs do not check out

Post TreeHeads periodically

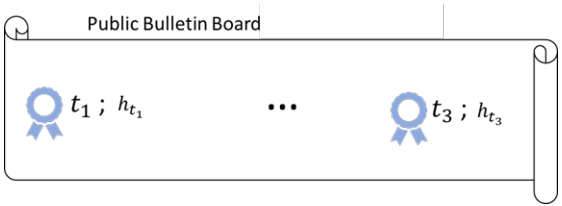
Clients get cryptographic proofs of correctness

Clients Devices

Service holds public identity keys for every client device

Auditable Key Directory Service

Periodic publish



Username	Public Key
Alice	
Bob	
…	…

t_1

Username	Public Key
Alice	
Bob	
…	…

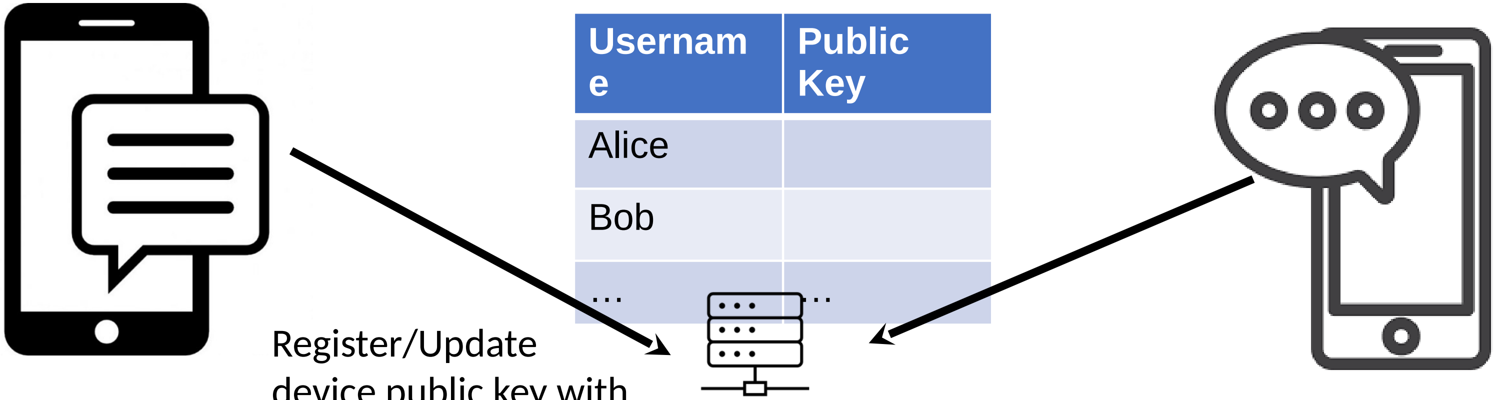
t_2

Username	Public Key
Alice	
Bob	
…	…

t_3

Service Provider (SP)

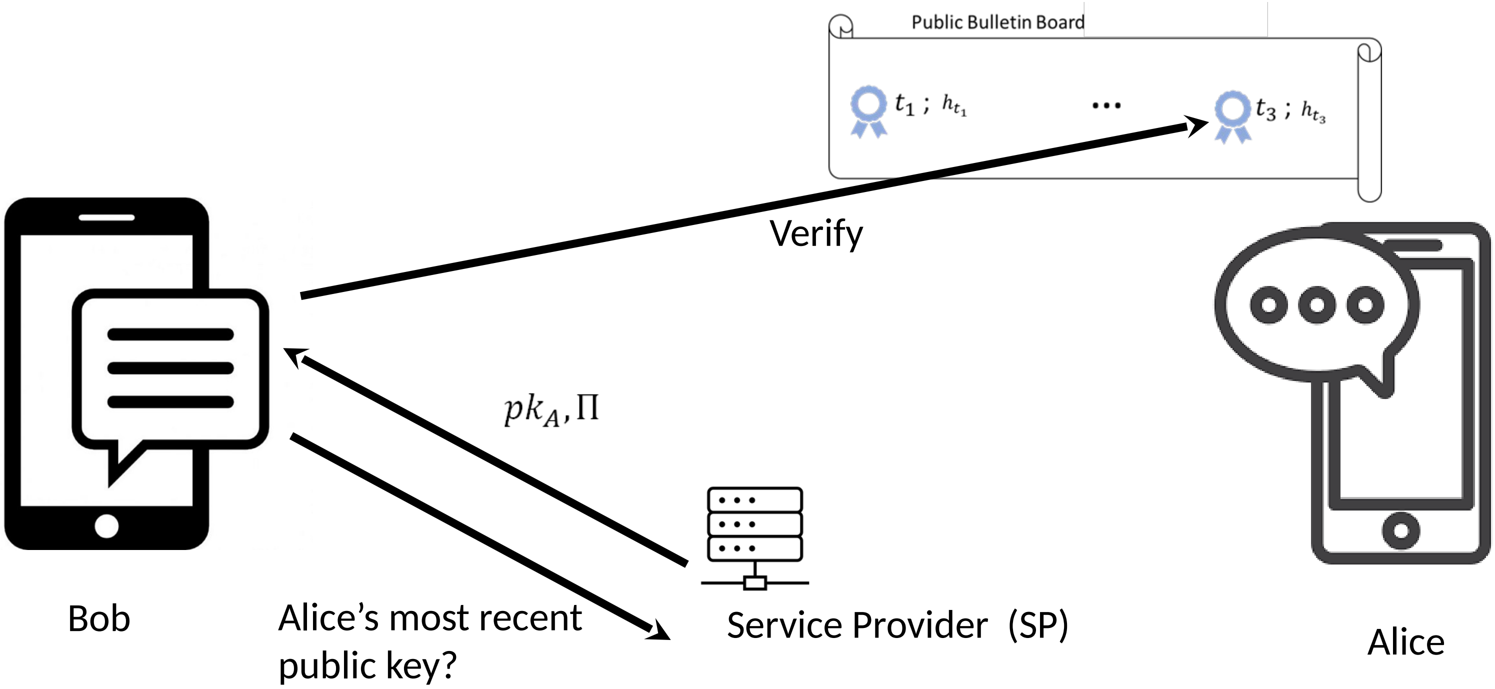
Registration and Update



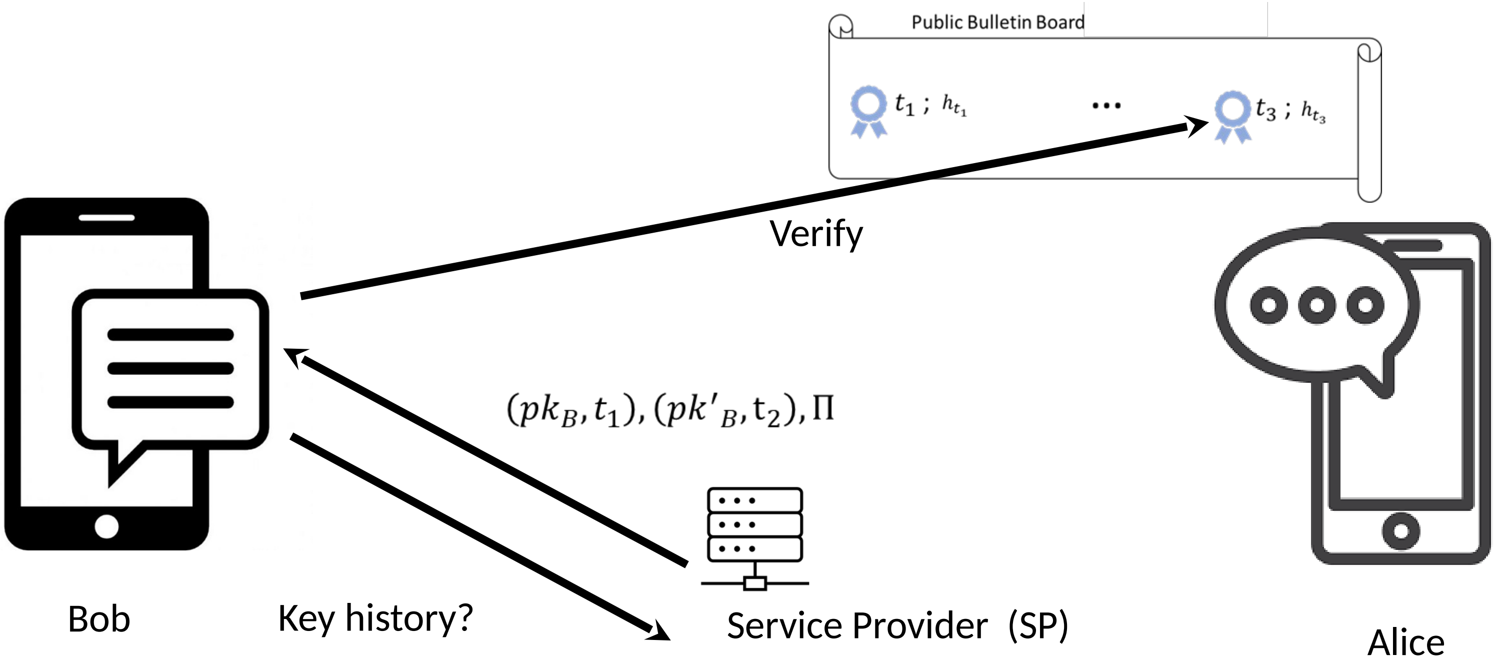
Username	Public Key
Alice	
Bob	
…	…

Register/Update device public key with username.

Lookup contact’s key



Monitor own key



Public Bulletin Board

t_1 ; h_{t_1} … t_3 ; h_{t_3}

Verify

$(pk_B, t_1), (pk'_B, \mathrm{t}_2), \Pi$

Bob

Key history?

Service Provider (SP)

Alice

Goals of this talk

- Decide on the important security and privacy properties of a KT system
    - Correctness
    - Consistency
    - Privacy

Assumption: Dissemination of TreeHeads

- Bulletin Board: Disseminates TreeHeads through a third-party Bulletin Board

- Gossip Channel: Dissemination of TreeHeads happen through gossip among the participants (comparing TreeHeads directly with each other)

Consistency Properties

- Bob’s latest key: 🔑
- When Alice queries for Bob’s latest key, she sees a fake key: 🔑
- When and Who will detect this consistency?

Correctness

- If the Service Provider is honest and user Alice is honest, another user Bob will always receive the correct key for Alice

Consistency Properties (cont.)

- Only the owner of the key (Bob) will be able to authoritatively decide that 🔑 was a fake key.

- So, at least 2 checks needs to happen:
  - Alice needs to ensure that the key they are seeing is committed by the server in the latest tree head
  - Bob needs to see this key distributed on their behalf while auditing for their own key

Strong Consistency

- Bob detects the inconsistency

the first time they come online

since the distribution of the fake

key and audits their key history

Weak Consistency

- Bob cannot detect the inconsistency the first time they come online since the distribution of the fake key and audits their key history
- Additional checks need to be performed (by Alice or other parties) for this to be detected
- Either Alice or Bob will be able to detect it after Bob audits their own key history and the additional checks are preformed by the other parties

Contact-Statefulness

- Each client can either remember the last keys, version numbers and possibly other auxiliary information for their contact's keys or not. We define a client to be contact-stateful if they remember any information about their contacts' keys in their states.

- We define the clients to be contact-stateless if they do not have to remember any information about their contact's keychain.

Owner-Signing

- If a malicious server publishes a tree head at a certain time and compromises a user’s device some time after that, the server still should not be able to have clients who hold that tree head accept any keys that the user’s device did not authorize before being corrupted

- This property is orthogonal to the consistency properties

Privacy

- Content Hiding:
  - Hides public keys and usernames from unauthorized parties
  - Achieved using Verifiable Random Function (VRF) and Commitments

- Metadata Hiding
  - Also, hides information such as when each user first registered in the KT, when and how often their keys change, correlations between multiple updates, etc.

Post Compromise Security

- Healing from a compromise of the server’s VRF secret key.

- Guarantees that privacy is regained for newly added directory entries after the compromise, once the server rotates their VRF key after the compromise.

Deployment Modes

- 3rd Party Management Mode (3PM):
  - TreeHeads are disseminated through a trusted third party (or a quorum) who audits the TreeHeads for consistency before distributing them.
  - In terms of the academic literature on KT, this mode is equivalent to having a trusted auditor who pre-emptively audits the TreeHeads.

- 3rd Party Auditing Mode (3PA):
  - The auditor can audit the TreeHeads with a lag
  - This mode leads to eventual detection rather than preemptive detection of misbehavior