zoom Security

Key Transparency at Keybase and Zoom

March 29, 2023

Antonio Marcedone
antonio.marcedone@zoom.us

Presenter

Antonio Marcedone

Cryptography Engineering Manager
Zoom Video Communications

Agenda

- **Keybase**
  - User identity
  - The Keybase Merkle Tree
- Zoom
  - User Identity
  - Identity Provider Attestations
  - The Zoom Transparency Tree
- Compare and contrast

Note: This presentation often simplifies and omits important details. Please check Zoom's Cryptography Whitepaper and Keybase's documentation / client source code for more details.
The statements and information provided are intended for **informational purposes only** and should not be relied upon in making a purchasing decision and may not be incorporated into any contract.

Keybase

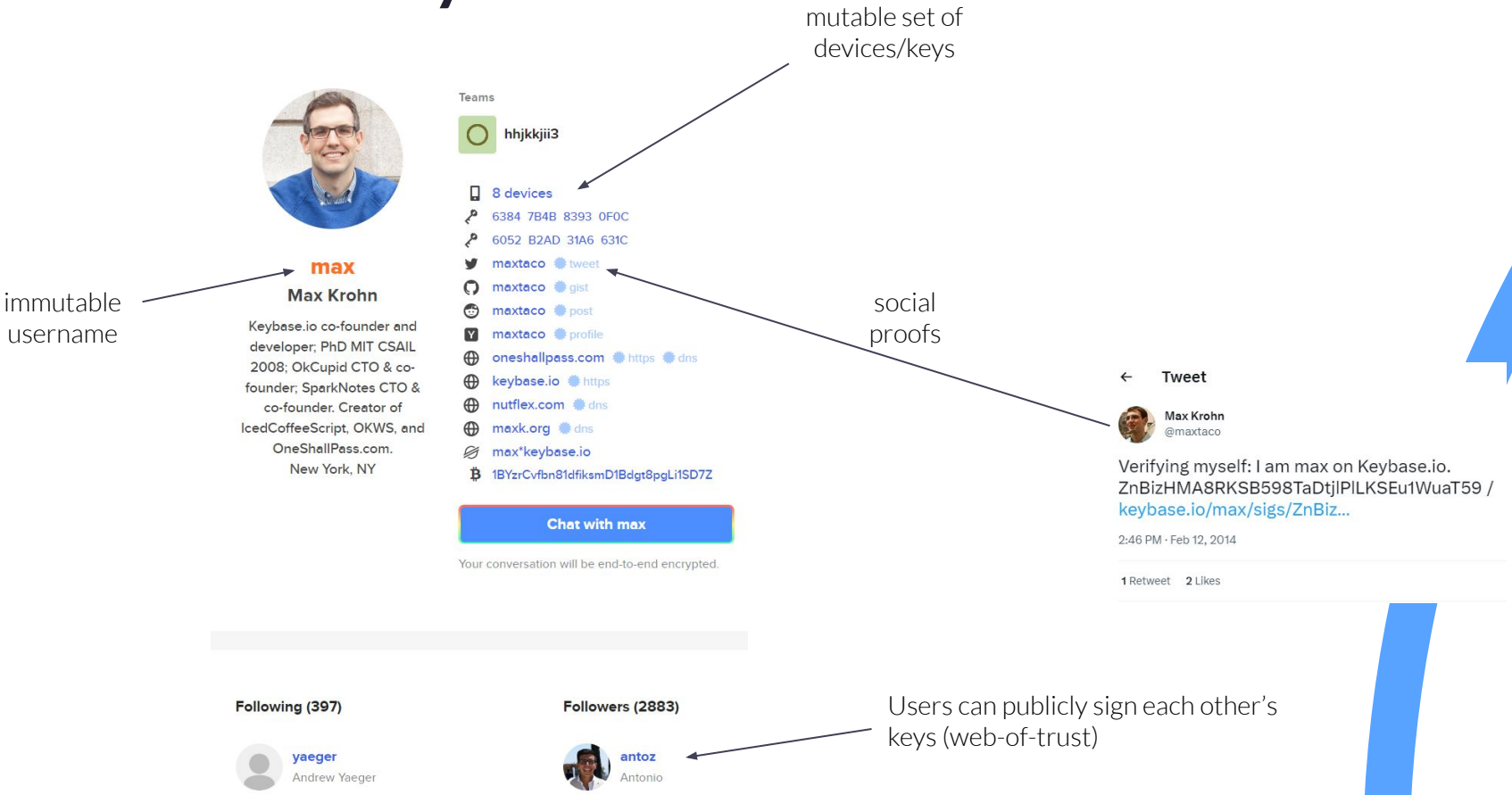
- Keybase maps social media identities to cryptographic keys in a publicly auditable manner.
  - Supports Github, Twitter, Reddit, Facebook, …
  - Users can reason in terms of devices, not keys

- Leverages the key directory to enable several secure applications:
  - Chat
  - Cloud storage/File sharing
  - Git repos
  - Cryptocurrency wallets
  - Teams
  - …

- Only (to my knowledge) Key Transparency directory deployed in production (since 2014!)

Keybase User Identity

https://keybase.io/max/



Keybase Sigchains

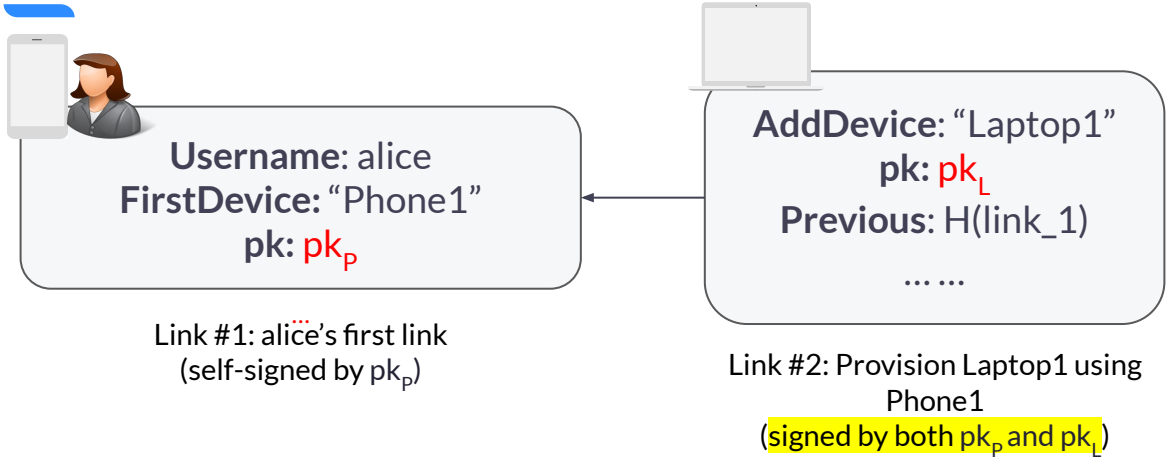
https://keybase.io/max/sigchain

Username: alice
FirstDevice: “Phone1”
pk: pk_P

Link #1: alice’s first link
(self-signed by pk_P)

Keybase Sigchains

https://keybase.io/max/sigchain

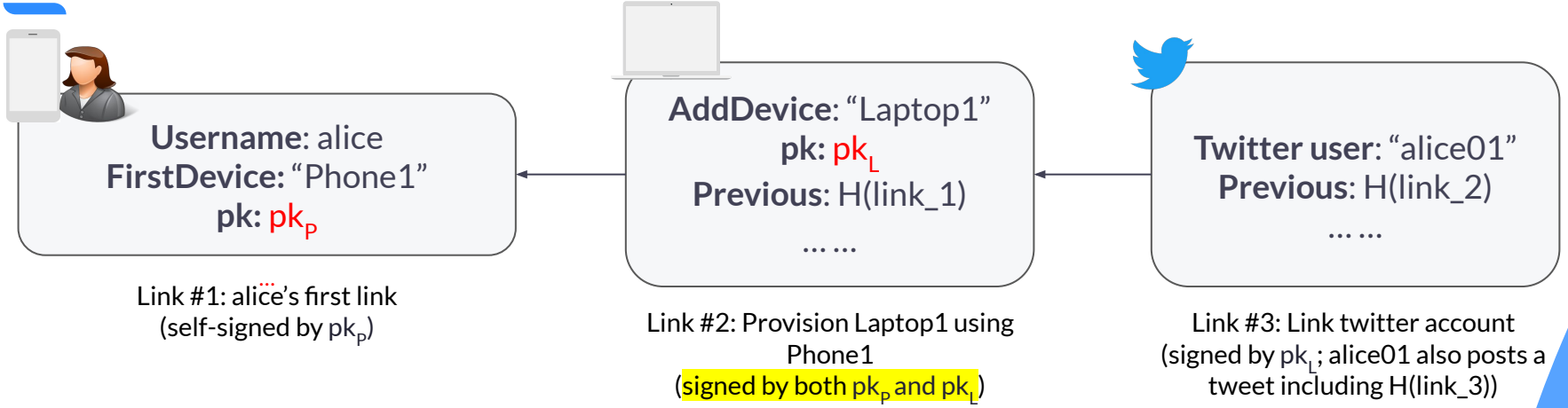


Link #1: alice's first link
(self-signed by pk_P)

Link #2: Provision Laptop1 using Phone1
(signed by both pk_P and pk_L)

Keybase Sigchains

https://keybase.io/max/sigchain



Link #1: alice's first link (self-signed by pk_P)

Link #2: Provision Laptop1 using Phone1 (signed by both pk_P and pk_L)

Link #3: Link twitter account (signed by pk_L; alice01 also posts a tweet including H(link_3))

Keybase Sigchains

https://keybase.io/max/sigchain

Username: alice
FirstDevice: "Phone1"
pk: pk_P

Link #1: alice's first link
(self-signed by pk_P)

AddDevice: "Laptop1"
pk: pk_L
Previous: H(link_1)
… …

Link #2: Provision Laptop1 using Phone1
(signed by both pk_P and pk_L)

Twitter user: "alice01"
Previous: H(link_2)
… …

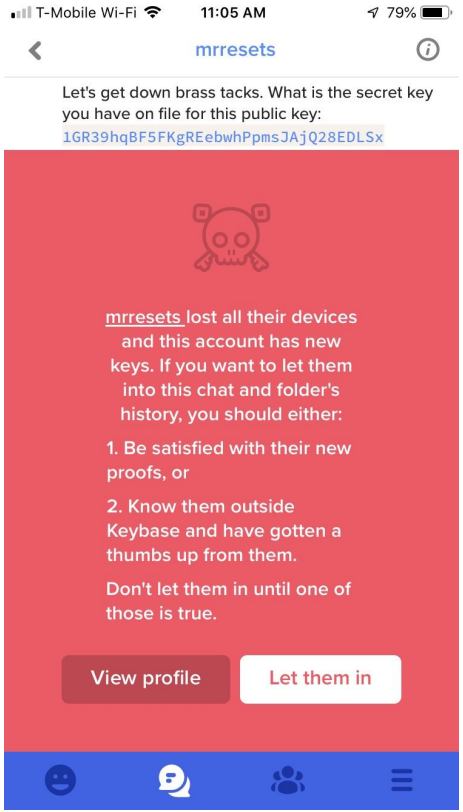
Link #3: Link twitter account
(signed by pk_L; alice01 also posts a tweet including H(link_3))

RevokeDevice: "Laptop1"
pk: pk_L
Previous: H(link_3)
… …

Link #4: Revoke Laptop1
(signed by pk_P)

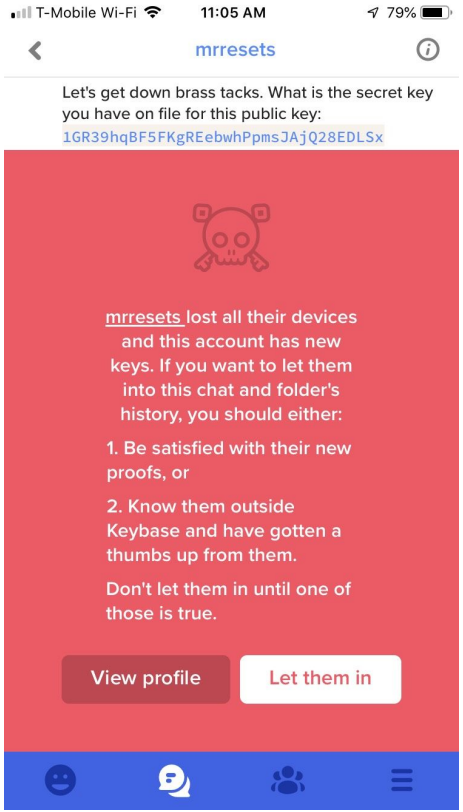
Account resets and Lockdown mode

- Users can "**reset**" their account to keep their username if they have lost all of their devices.



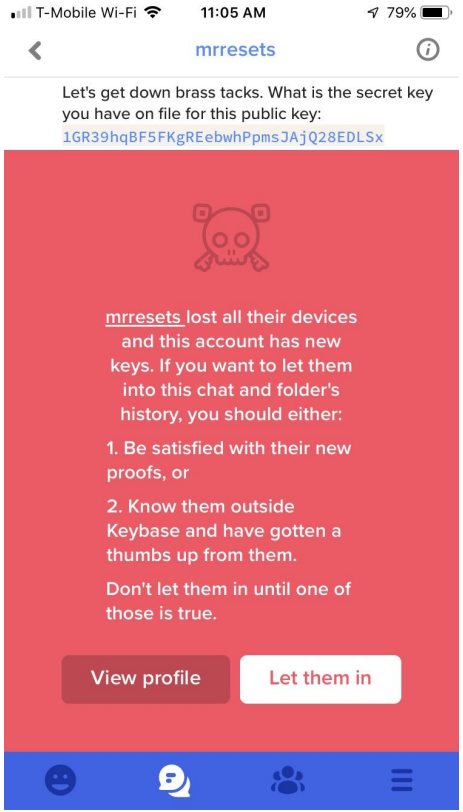
Account resets and Lockdown mode

- Users can "**reset**" their account to keep their username if they have lost all of their devices.
- Their conversation partners need to give **explicit consent** before old data (files, chats) is encrypted for the user's new keys.



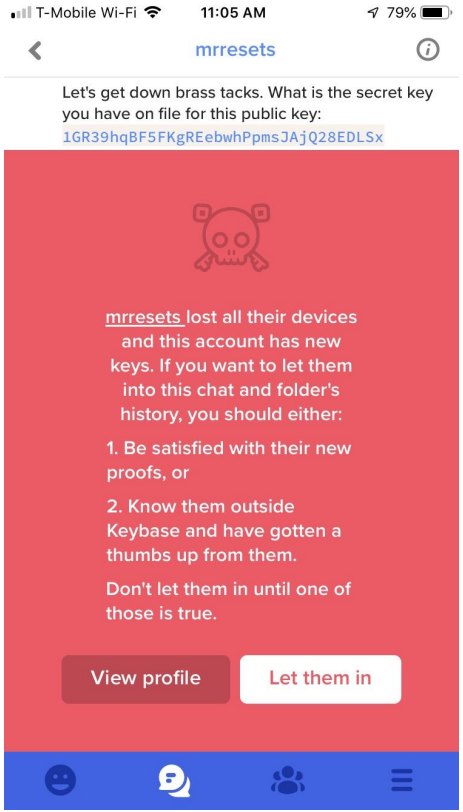
Account resets and Lockdown mode

- Users can "**reset**" their account to keep their username if they have lost all of their devices.
- Their conversation partners need to give **explicit consent** before old data (files, chats) is encrypted for the user's new keys.
- Users can opt into "**Lockdown Mode**". In this mode, the server disallows all account changes made from the website (including resets!).



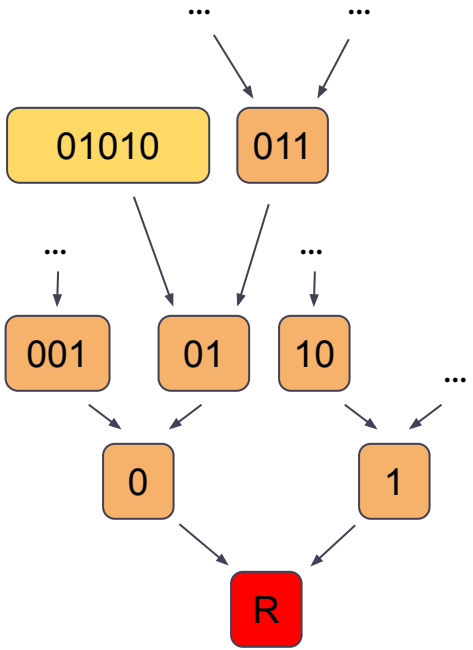
Account resets and Lockdown mode

- Users can "**reset**" their account to keep their username if they have lost all of their devices.
- Their conversation partners need to give **explicit consent** before old data (files, chats) is encrypted for the user's new keys.
- Users can opt into "**Lockdown Mode**". In this mode, the server disallows all account changes made from the website (including resets!).
- A user could also write a **Do-Not-Reset link** to their chain, for maximum security (not implemented yet).



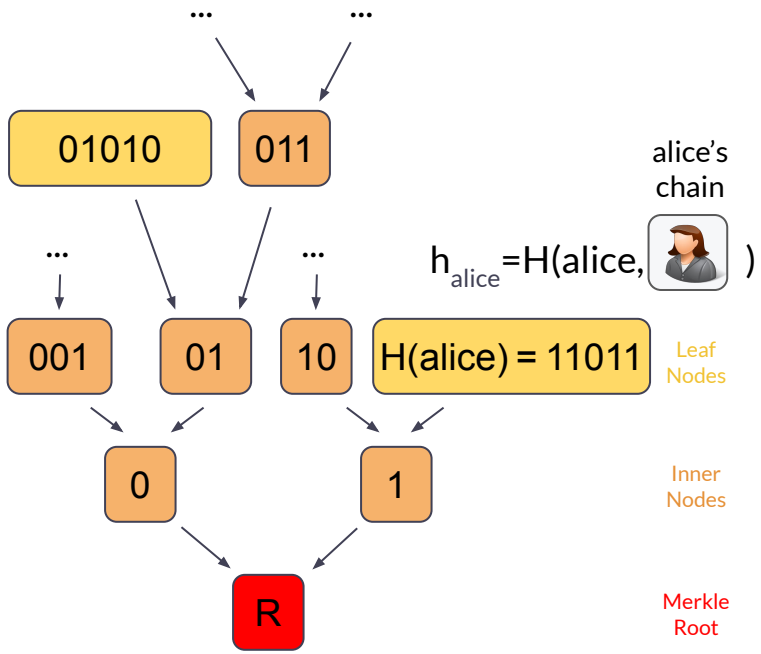
Keybase's KT Merkle Tree (simpl.)

- (Compressed) Prefix Tree, each node has an associated label and value



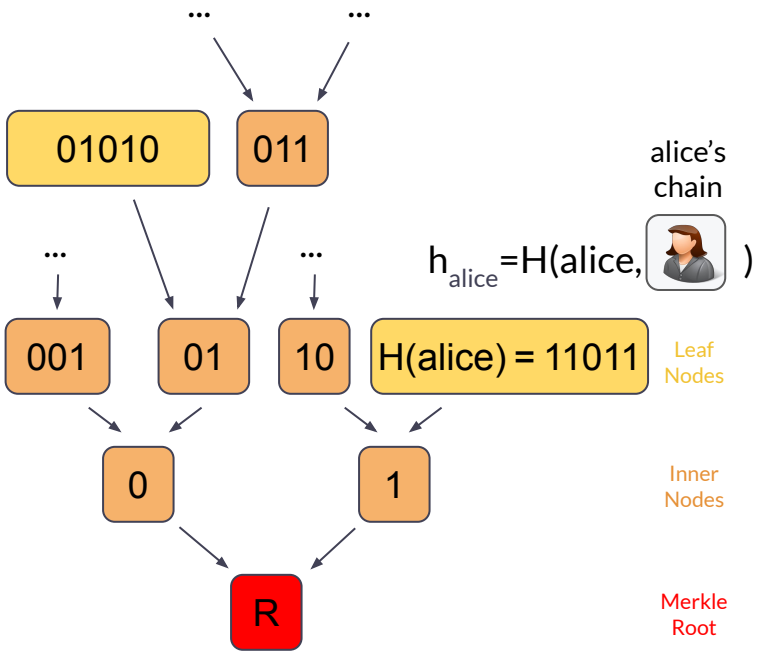
Keybase's KT Merkle Tree (simpl.)

- (Compressed) Prefix Tree, each node has an associated label and value
- Each user corresponds to a leaf: the label is the hash of the username, the value is the hash of the sigchain.



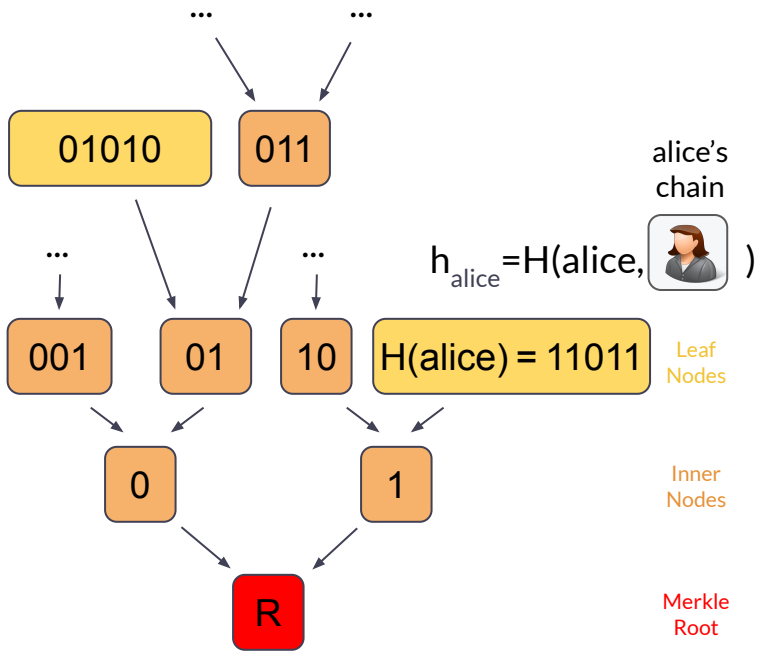
Keybase's KT Merkle Tree (simpl.)

- (Compressed) Prefix Tree, each node has an associated label and value
- Each user corresponds to a leaf: the label is the hash of the username, the value is the hash of the sigchain.
- Requires fast updates!



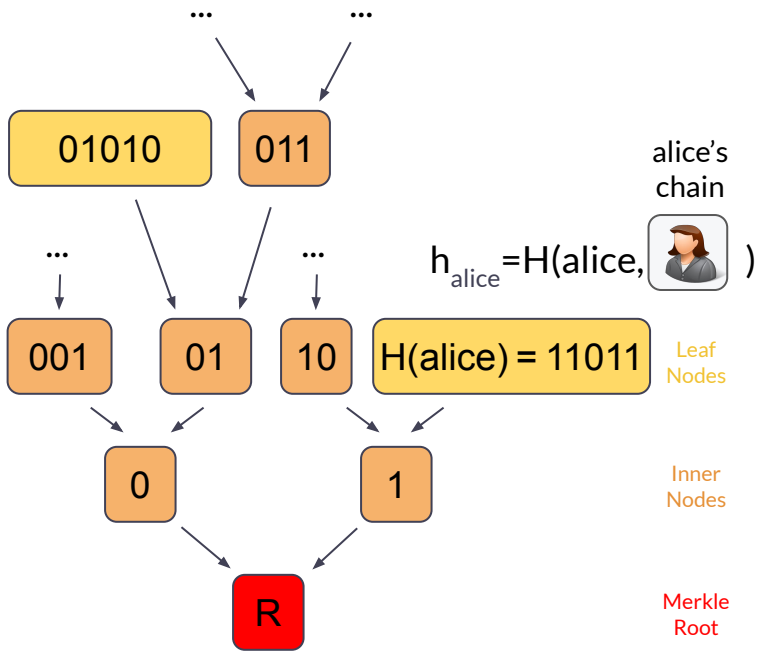
Keybase's KT Merkle Tree (simpl.)

- (Compressed) Prefix Tree, each node has an associated label and value
- Each user corresponds to a leaf: the label is the hash of the username, the value is the hash of the sigchain.
- Requires fast updates!
- Publicly available/auditable: https://keybase.io/_/api/1.0/merkle/root.json?seqno=1 (block.json and path.json)



Keybase's KT Merkle Tree (simpl.)

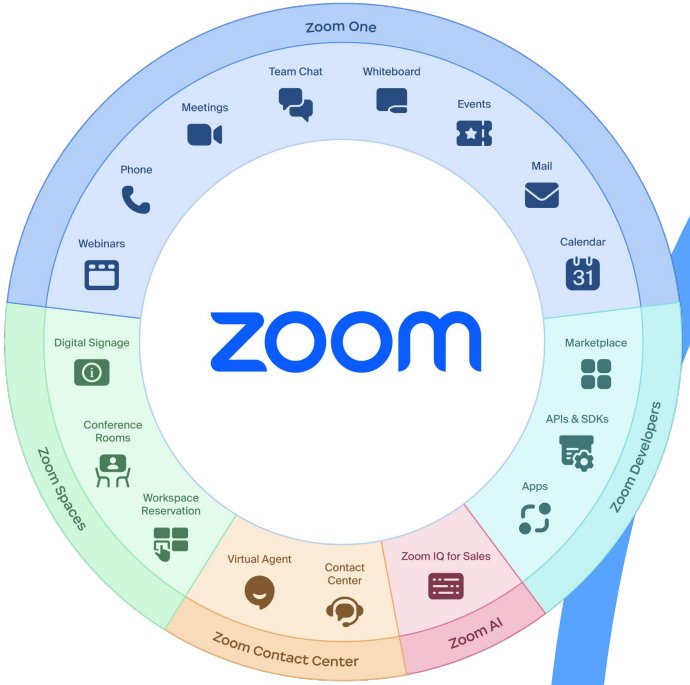
- (Compressed) Prefix Tree, each node has an associated label and value
- Each user corresponds to a leaf: the label is the hash of the username, the value is the hash of the sigchain.
- Requires fast updates!
- Publicly available/auditable: https://keybase.io/_/api/1.0/merkle/root.json?seqno=1 (block.json and path.json)
- Root is periodically posted to the Stellar (previously Bitcoin) blockchain



Agenda

- Keybase
  - User identity
  - The Keybase Merkle Tree
- **Zoom**
  - User Identity
  - Identity Provider Attestations
  - The Zoom Transparency Tree
- Compare and contrast

- Communications platform
- In May 2020, we published a whitepaper describing a plan to bring end to end encryption and a strong multi-device notion of identity to our users.
  - Includes a Key Transparency system!
- Currently offers end-to-end encryption for:
  - Zoom Meetings
  - 1-on-1 Zoom Phone calls (intra-account only)
  - Zoom Mail Service (only for emails sent directly between active Zoom Mail Service users)

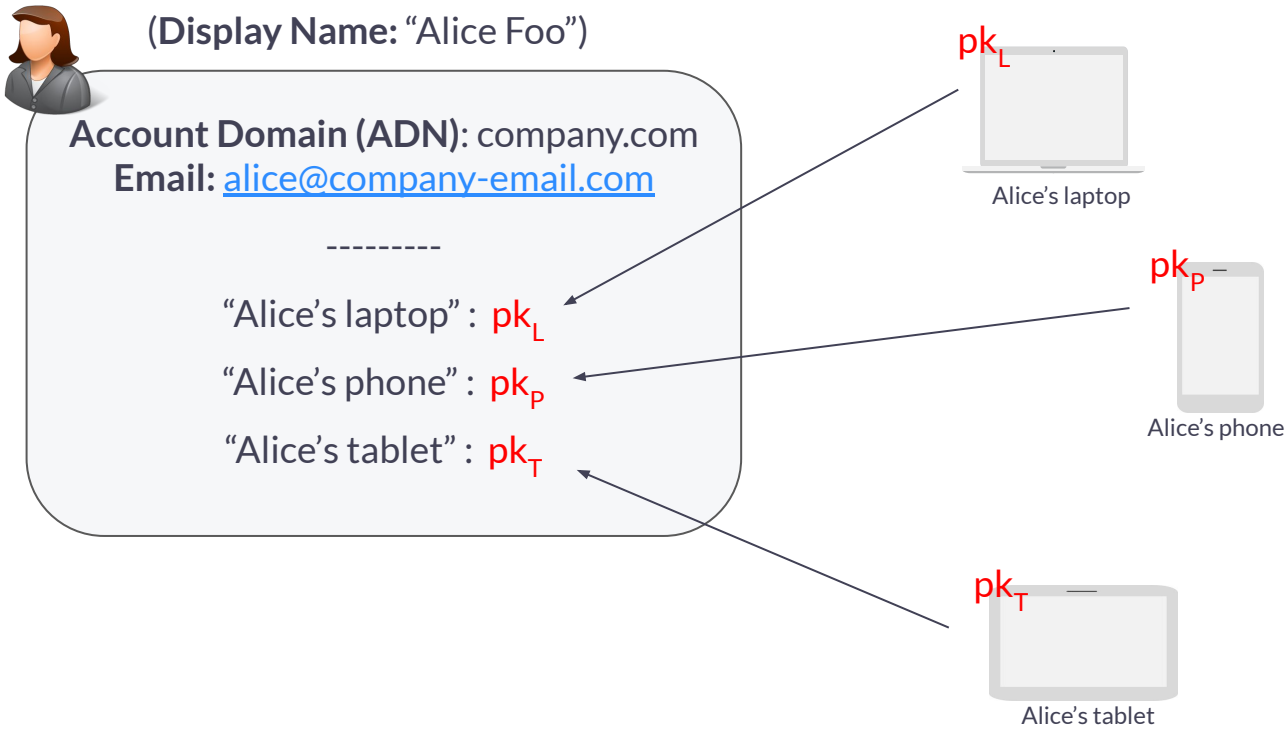


Zoom User Identity

(**Display Name:** “Alice Foo”)

Account Domain (ADN): company.com
Email: alice@company-email.com

Zoom User Identity



Zoom User Identity

(**Display Name:** “Alice Foo”)

Account Domain (ADN): company.com
Email: alice@company-email.com

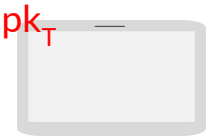
“Alice’s laptop” : pk_L

~~“Alice’s phone” : pk_P~~ **(REVOKED)**

“Alice’s tablet” : pk_T

Alice’s laptop

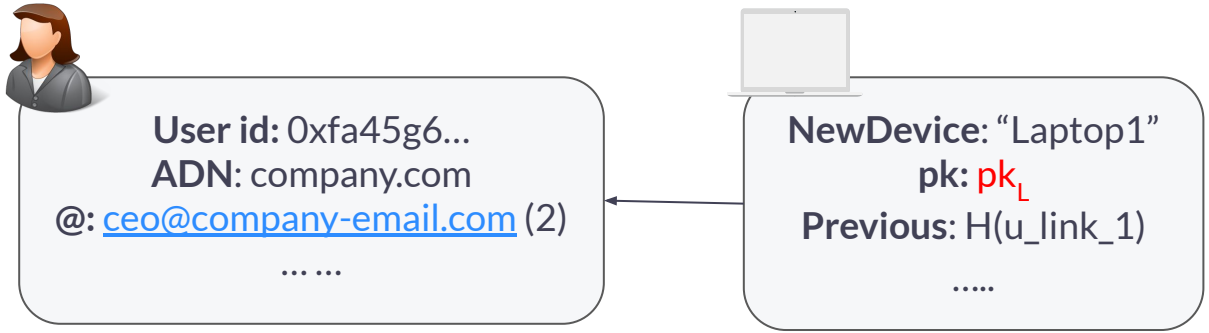
Alice’s phone



Alice’s tablet

Zoom Sigchains (simpl.)

User sigchain:



User Link #1: Alice’s identifiers

User Link #2: Provision laptop (self-signed)

Zoom Sigchains (simpl.)

User sigchain:

User Link #1: Alice’s identifiers

User Link #2: Provision laptop (self-signed)

User Link #3: Provision phone (**self-signed**)

Zoom Sigchains (simpl.)

User sigchain:

User id: 0xfa45g6…
ADN: company.com
@: ceo@company-email.com (2)
… …

User Link #1: Alice's identifiers

NewDevice: "Laptop1"
pk: pk_L
Previous: H(u_link_1)
…..

User Link #2: Provision laptop (self-signed)

NewDevice: "Phone1"
pk: pk_P
Previous: H(u_link_2)
…..

User Link #3: Provision phone (**self-signed**)

Approver: "Laptop1", pk_L
Approved: "Phone1", pk_P
…..

Link #4: Approve Phone1
(signed by pk_L)

Zoom Sigchains (simpl.)

User sigchain:

User id: 0xfa45g6…
ADN: company.com
@: ceo@company-email.com (2)
… …

User Link #1: Alice's identifiers

NewDevice: "Laptop1"
pk: pk_L
Previous: H(u_link_1)
…..

User Link #2: Provision laptop (self-signed)

NewDevice: "Phone1"
pk: pk_P
Previous: H(u_link_2)
…..

User Link #3: Provision phone (**self-signed**)

cross reference

Email sigchain:

@: ceo@company-email.com
User id: 0x001122…

Email Link #1

@: ceo@company-email.com
User id: 0xfa45g6… (1)
Previous: H(link_1)

Email Link #2

IdP attestations (“Okta authentication for E2EE”)

External Identity Providers can attest to their user’s Zoom identities:

company.com

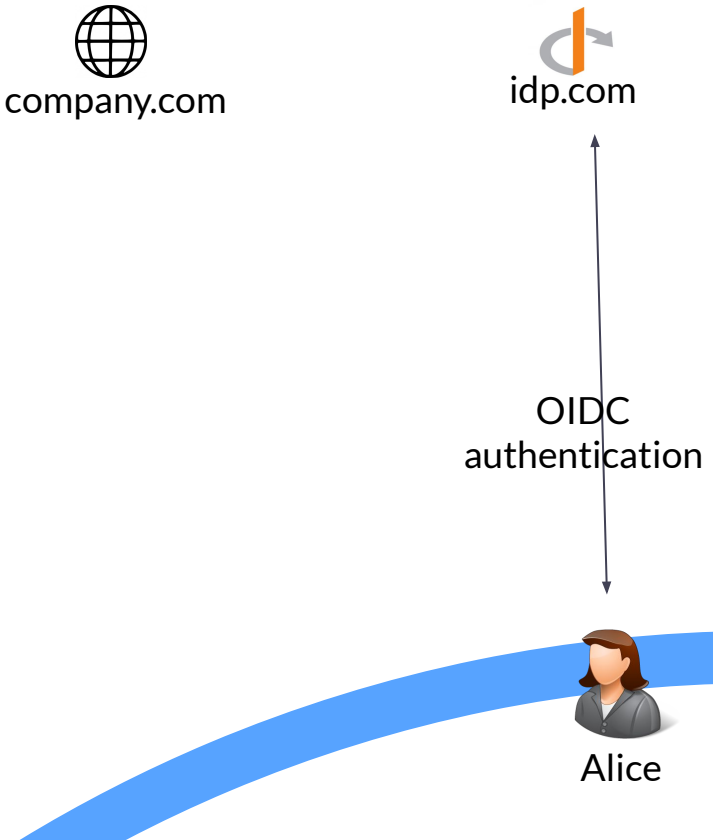
idp.com

Alice

IdP attestations (“Okta authentication for E2EE”)

External Identity Providers can attest to their user’s Zoom identities:

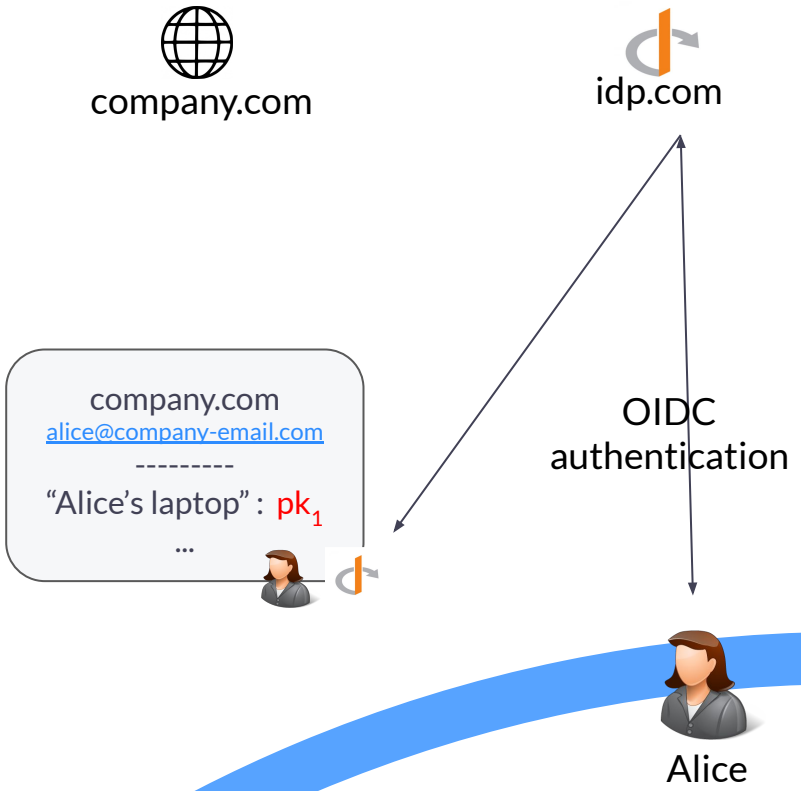
1. Alice authenticates to the IdP (OpenID Connect)



IdP attestations (“Okta authentication for E2EE”)

External Identity Providers can attest to their user’s Zoom identities:

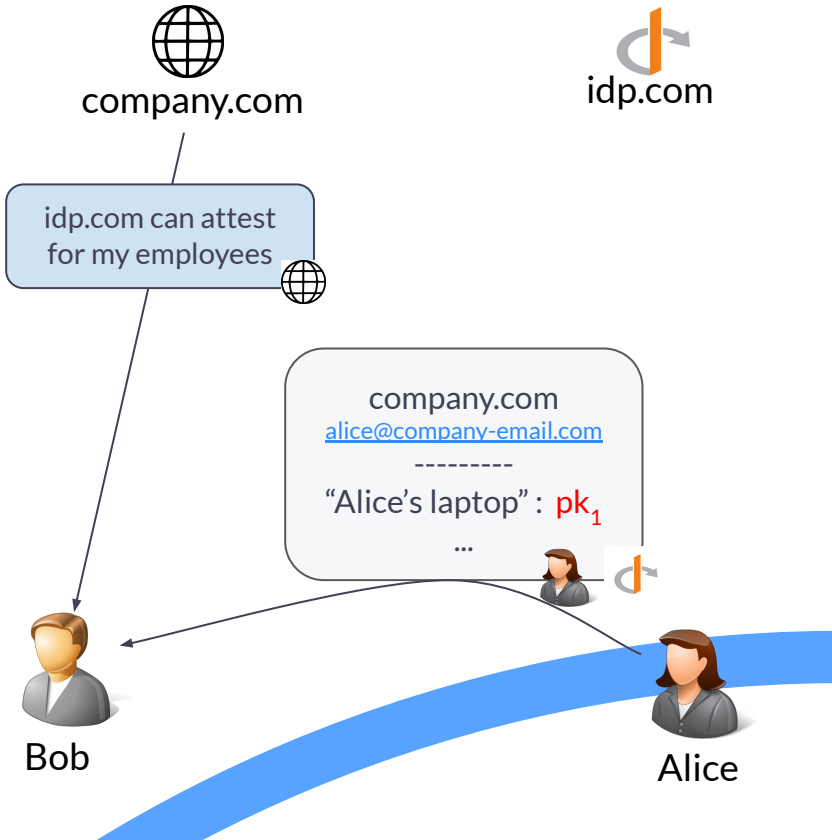
1. Alice authenticates to the IdP (OpenID Connect)
2. Alice requests a modified OIDC Identity token (signed by the IdP) which contains a hash of her Zoom identity



IdP attestations (“Okta authentication for E2EE”)

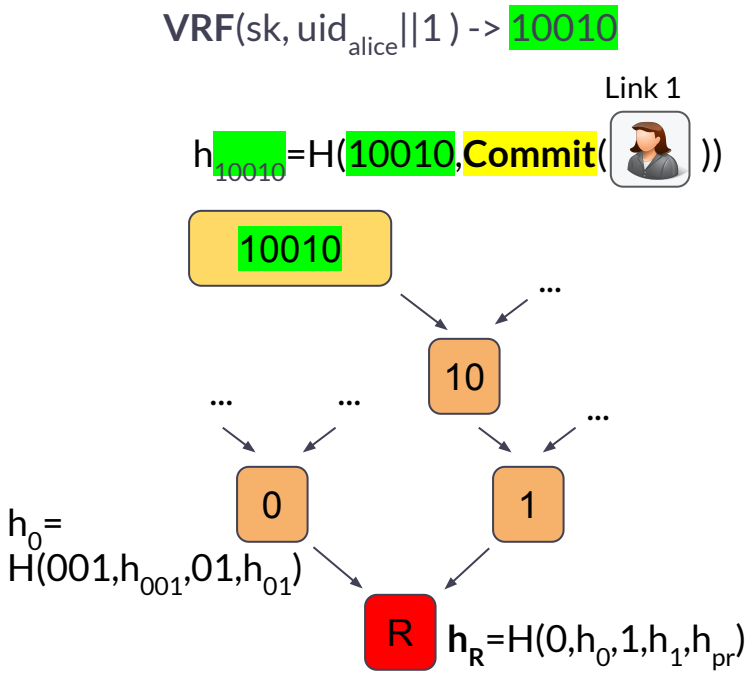
External Identity Providers can attest to their user’s Zoom identities:

1. Alice authenticates to the IdP (OpenID Connect)
2. Alice requests a modified OIDC Identity token (signed by the IdP) which contains a hash of her Zoom identity
3. Participants check that the IdP is authorized by company.com by making a DNS request to the domain
4. Participants check the Identity and token



Zoom Transparency Tree (simpl.)

- Each sigchain link is stored as a separate entry in the set (as in SEEMLess)
  - Leaves are never removed/updated
- Backed by a Rotatable Zero Knowledge Set (Asiacrypt'22):
  - Uses a (Rotatable) Verifiable Random Function (VRF) to compute the position of each link
  - VRF key can be periodically rotated, while still proving that the append only property is respected.
- If Alice's chain consists of 5 links, a proof their chain is in the tree consists of inclusion proofs for the first 5 links, and an exclusion proof for the 6th label



Agenda

- Keybase
  - User identity
  - The Keybase Merkle Tree
- Zoom
  - User Identity
  - Identity Provider Attestations
  - The Zoom Transparency Tree
- **Compare and contrast**

Comparison

- Immutable username

- Mutable identifiers (email, ADN)

Comparison

- Immutable username
- Social proofs

- Mutable identifiers (email, ADN)
- Identity Provider Attestations

Comparison

- Immutable username
- Social proofs
- Adding a device requires signature from existing device

- Mutable identifiers (email, ADN)
- Identity Provider Attestations
- Devices can be “approved” after they have been added

Comparison

- Immutable username
- Social proofs
- Adding a device requires signature from existing device
- User identities are public

zoom

- Mutable identifiers (email, ADN)
- Identity Provider Attestations
- Devices can be “approved” after they have been added
- Identities are private by default

Comparison

- Immutable username
- Social proofs
- Adding a device requires signature from existing device
- User identities are public
- Simple Merkle Tree: leaves can be updated

- Mutable identifiers (email, ADN)
- Identity Provider Attestations
- Devices can be “approved” after they have been added
- Identities are private by default
- Rotatable Zero Knowledge set: Append only, with privacy, VRF key can be rotated.

Comparison

- Immutable username
- Social proofs
- Adding a device requires signature from existing device
- User identities are public
- Simple Merkle Tree: leaves can be updated
- Leaks when each sigchain is updated

zoom

- Mutable identifiers (email, ADN)
- Identity Provider Attestations
- Devices can be “approved” after they have been added
- Identities are private by default
- Rotatable Zero Knowledge set: Append only, with privacy, VRF key can be rotated.
- Limited leakage about sigchain updates

Comparison

- Immutable username
- Social proofs
- Adding a device requires signature from existing device
- User identities are public
- Simple Merkle Tree: leaves can be updated
- Leaks when each sigchain is updated
- Efficiency: single inclusion proof per sigchain

zoom

- Mutable identifiers (email, ADN)
- Identity Provider Attestations
- Devices can be “approved” after they have been added
- Identities are private by default
- Rotatable Zero Knowledge set: Append only, with privacy, VRF key can be rotated.
- Limited leakage about sigchain updates
- Efficiency: one inclusion proof per sigchain link; auditing requires VRF rotation proofs

Thank You

Antonio Marcedone
antonio.marcedone@zoom.us

Connect With Us

@amarcedone

@zoom_us | blog.zoom.us

zoom Security

Bonus Content

Auditing

To audit an update between epoch t and t+1, auditors would check that:

- The root at epoch t+1 includes the correct hash for the one at epoch t
- No label/value pairs are removed from the tree
- If the value corresponding to a user's sigchain is updated, check that the new hash extends the old one
- (For teams, members perform probabilistic audits by asking for the value at random epochs)

zoom

- The root at epoch t+1 includes the correct hash for the one at epoch t
- The append only property is respected
- VRF rotation proof (where applicable)

Verifiable Random Functions

- **Gen**() -> sk,pk
- **VRF**(sk, label) -> out, proof
- **Verify**(pk,label,out,proof) -> 0/1

Security:

- **Uniqueness**
- **Pseudorandomness**
- **Collision Resistance**

Example (simplified): Let (G,g) be a DDH group

- **Gen**() -> sk = x, pk = g^x
- **VRF**(x, label) = $H(label)^x$, DH proof
- **Verify**(): Check proof