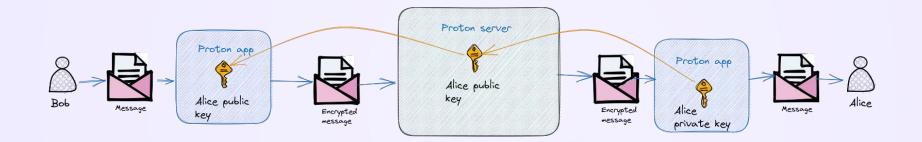
Key Transparency in Proton Mail

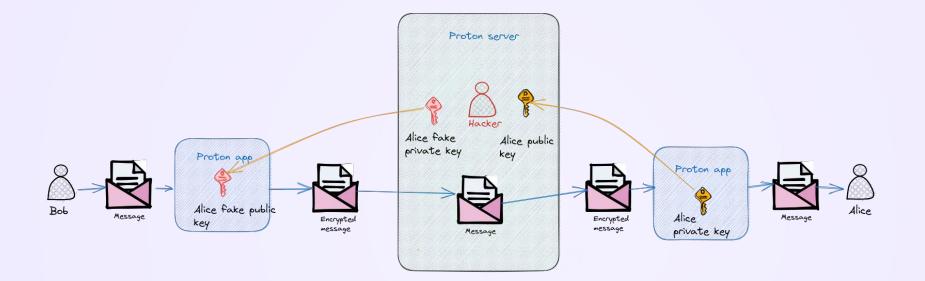
Transparency Summit



Proton's End-to-End Encryption



Problem: Man-in-the-middle attack





Manual verification of keys

Edit email settings

marin.test.2@protonmail.com

Select the email format you want to be used by default when sending an email to this email address.

Email format ()

Automatic

-

Hide advanced PGP settings

To use Address Verification, you must trust one or more available public keys, including the one you want to use for sending. This prevents the encryption keys from being faked. Learn more

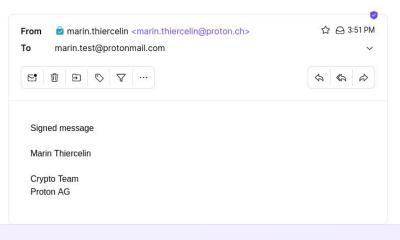
Public keys 🕦

| Fingerprint | Created | Expires | Туре | Status | Actions |
|-------------|--------------|---------|------------|---------|------------|
| dbd12a0bc26 | Dec 17, 2020 | 121 | RSA (2048) | PRIMARY | Download 🔺 |
| | | | | | Trust |
| Cancel | | | | | Save |

| New | message | | — · | ? × |
|------|------------------------------|------|-----|----------|
| Fron | marin.test@protonmail.com ~ | | | |
| То | amarin.test.2@protonmail.com | CC B | scc | ₽ |
| Subj | ect Subject | | | |

Verified message from a sender with a pinned key

X

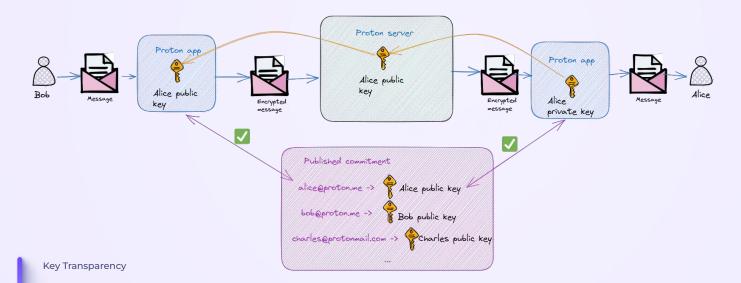




Automatic verification of keys

High-level protocol:

- 1. The server **publishes all keys** for each Proton Mail address
- 2. The user **verifies that their own keys matches** what has been published for their address
- 3. When sending a message, the user **verifies that the recipient public keys matches** what has been published for the recipient address



History of Key Transparency at Proton

- 2018: Started working on KT based on CONIKS
- 2023: Thore Göbel from ETHZ brought in some ideas from SEEMless and Parakeet
- 2023: Launched in beta (opt-in)
- 2024: Whitepaper published

Publishing the list of keys Creating an epoch

High-level protocol:

- 1. The server **publishes all keys** for each Proton Mail address
- 2. The user **verifies that their own keys matches** what has been published for their address
- When sending a message, the user verifies that the recipient public keys matches what has been published for the recipient address

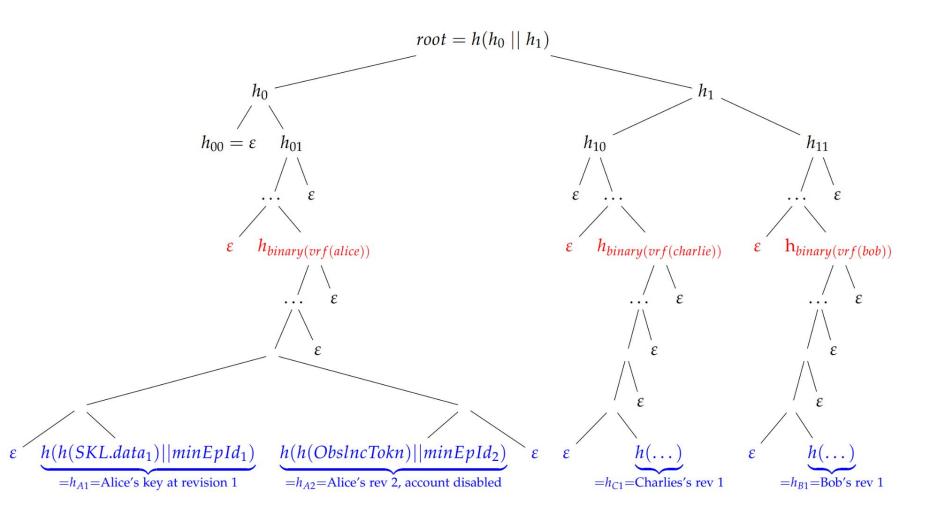
Publishing all keys

Every 4 hours:

- Build a Merkle tree of the keys of every email address at every revision
- Compute tree root hash
- Compute chain hash (hash of root hash and previous chain hash)
- Request certificate for {chainhash[0:32]}.{chainhash[32:64]}.

{issuanceTime}.{epochid}.1.keytransparency.ch

• Certificate gets published to Certificate Transparency logs



Auditing process

Auditors have to check:

- There is only one root hash per epoch ID in CT logs
- The epochs form a consistent chain
- Subtrees are append-only (except for stale entries)

Verifying the user own keys The self audit

High-level protocol:

- 1. The server **publishes all keys** for each Proton Mail address
- 2. The user **verifies that their own keys matches** what has been published for their address
- 3. When sending a message, the user verifies that the recipient public keys matches what has been published for the recipient address

The self audit procedure

At regular intervals:

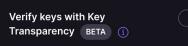
- 1. Fetch the latest epoch & verify the certificate
- 2. Check that the public key fingerprints in the latest revision match the private keys
- 3. Check that previous modifications were included in the epoch
- 4. Check that additional included revisions are signed

Signed Key List

```
Data: [
    {
        Primary: true,
        Flags: 0,
        SHA256Fingerprints: ["0123ABCD...", "4567EFAB..."]
    },
    {
        Primary: false,
        Flags: 1,
        SHA256Fingerprints: ["8901CDEF...", "2345ABCD..."]
    },
    ...
],
Signature: Sign(PrimaryKey, JSON.stringify(Data))
```

Self audit user interface

| ① Upgrade | (marin.test marin.test@willis.proton.black | м |
|-----------|---|---|
| | Go to settings | - |
| | Beta Access Off | |
| <u>š</u> | Theme Proton | 3 |
| | Keyboard shortcuts On | |
| | Mailbox layout Row | |
| | Mailbox density Comfortable | |
| | Composer size Normal | |
| | Default email application | |
| | Key verification | |
| | Open checklist Get free storage | |
| | Clear browser data | |
| | | |



Key verification

×

Last verified: 1 minute ago

Proton Mail periodically verifies that your and your contacts' keys are consistent with the Key Transparency state. Learn more

 \checkmark

The following addresses were successfully audited with Key Transparency:

marin.test@willis.proton.black



Proton

Self audit user interface

| ① Upgrade | marin.test.wror marin.test.wrong.flag@willis.prot | |
|-----------|--|-----|
| | Go to settings | > |
| | Beta Access Off | l↑ |
| | Theme Proton | × |
| | | 23 |
| | Keyboard shortcuts On | 23 |
| | Mailbox layout Row | 23 |
| | Mailbox density Comfortable | 2.5 |
| | Composer size Normal | 23 |
| | Default email application | 23 |
| | Key verification | 23 |
| | Open checklist Get free storage | |

Key verification

X

Last verified: less than a minute ago

Proton Mail periodically verifies that your and your contacts' keys are consistent with the Key Transparency state. Learn more

Proton Mail failed to verify that the keys of this address are consistent with Key Transparency.

() marin.test.wrong.flag@willis.proton.black



Verifying the public keys of other users

High-level protocol:

- 1. The server **publishes all keys** for each Proton Mail address
- 2. The user **verifies that their own keys matches** what has been published for their address
- When sending a message, the user verifies that the recipient public keys matches what has been published for the recipient address

Proton Key Transparency

Verifying public keys from the server

In the composer

When Alice wants to write to bob@proton.me:

- 1. The client asks for the public keys for bob@proton.me
- 2. Fetch the latest epoch & verify the certificate
- 3. Check that the epoch includes the public keys

| New messa | ge | _ 2 × |
|-----------|---|-----------|
| From | marin.test@willis.proton.black ${\scriptstyle\checkmark}$ | |
| То | i marin.test.kt@willis.proton.black | CC BCC +≙ |
| Subject | Subject | |

| New messa | ge | | | 2 | × |
|-----------|---|----|-----|----------------|---|
| From | marin.test@willis.proton.black ~ | | | | |
| То | ① marin.test.wrong.flag <marin.test.wrong.fl th="" ×<=""><th>cc</th><th>BCC</th><th>Q⁴</th><th></th></marin.test.wrong.fl> | cc | BCC | Q ⁴ | |
| Subject | Subject | | | | |

Special case: the recipient keys changed recently

Asynchronous verification

If Bob's current keys are not in KT yet:

- 1. Alice's client accepts the keys without verifying the proof
- 2. Alice's client stores the keys in local storage
- 3. At the next periodic audit, Alice's client verifies that the new keys are included in the next epoch

When the asynchronous verification fails

The server might have lied

| Go to settings | > |
|---------------------------------|---|
| Beta Access Off | L↑ |
| Theme Proton | × |
| Keyboard shortcuts | 23 |
| | 23 |
| Mailbox density Comfortable | 23 |
| Composer size Normal | 23 |
| Default email application | 123 |
| Key verification | 23 |
| Open checklist Get free storage | |
| | Go to settings Beta Access Off Theme Proton Keyboard shortcuts On Mailbox layout Row Mailbox density Comfortable Composer size Normal Default email application • |

Key verification

×

Last verified: 11 minutes ago

Proton Mail periodically verifies that your and your contacts' keys are consistent with the Key Transparency state. Learn more

Proton Mail detected that the keys used in the past for this address may not be authentic.

() key.change.ignored@fake.address

To verify the security of your end-to-end encryption with this contact, ask them for the fingerprint of the first key that is displayed under Settings > Encryption & Keys for this address in their account, and check that it matches the following: e9b17ed69c279d8fba6ef36adea31ab4b6616e80



Proton

Other Edge Cases

Key Transparency



Deleted & Transferred Accounts

Account ownership might have changes

| Andy Yen | ☆ ◀ Aug 13 |
|-------------|------------|
| From 🚵 | ☆ ▲ Aug 13 |
| To Andy Yen | × × |
| | \$ \$ \$ |

Sampling of Other Issues

Random Sentry errors we've gotten

| s ☆ sentry@protontech.ch | WEB-MAIL-3TQ9 - [KeyTransparency] Error: Error signing message: Could not find valid self-signature in key 70a65c |
|--------------------------|---|
| s ☆ sentry@protontech.ch | WEB-MAIL-3TP6 - [KeyTransparency] SyntaxError: Invalid character: '\0' |
| s ☆ sentry@protontech.ch | WEB-MAIL-3TNT - [KeyTransparency] TypeError: Cannot create property '10' on number '9' |
| s ☆ sentry@protontech.ch | WEB-MAIL-3TM6 - [KeyTransparency] No keys detected |
| s ☆ sentry@protontech.ch | WEB-MAIL-3TJJ - [KeyTransparency] TypeError: Right side of assignment cannot be destructured |
| s 🏠 sentry@protontech.ch | WEB-MAIL-3TJA - [KeyTransparency] Error: Epoch certificate alternative name does not match |
| s ☆ sentry@protontech.ch | WEB-MAIL-3TH6 - [KeyTransparency] StatusCodeError: Not Found |
| s ☆ sentry@protontech.ch | WEB-MAIL-3TGM - [KeyTransparency] Error: Minified Redux error #3; visit https://redux.js.org/Errors?code=3 for th |
| s 🏠 sentry@protontech.ch | WEB-MAIL-3TFE - [KeyTransparency] SyntaxError: Unexpected token '<', " " is not valid JSON</td |

More details

https://proton.me/files/proton_keytransparency_whitepaper.pdf





Thanks!

Key Transparency



Any questions?

Key Transparency





The signed key list: a text representation of keys

Instead of publishing a list of keys for each address, the server publishes a text representation called the Signed Key List.

It is a JSON string with information about the keys of the address, signed by the user client with PGP.



Verifiable random function (VRF)

An asymmetric keyed hash

Usual hash functions:

• hash = Hash.hash(email)

Verifiable random functions

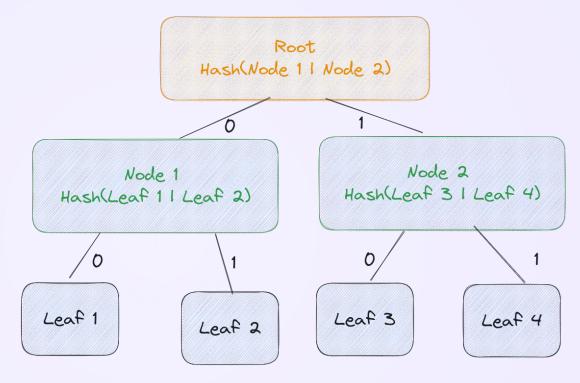
- (hash, proof) = VRF.hash(email, secret_key)
- yes / no = VRF.verify(email, hash, proof, public_key)

Regular hash functions can be computed by anyone, VRFs can only be computed by someone with the secret_key.

Merkle tree

A single hash for a list of values

- Each leaf has an assigned "path" in the binary tree
- The server can efficiently prove that a given value is at the right path

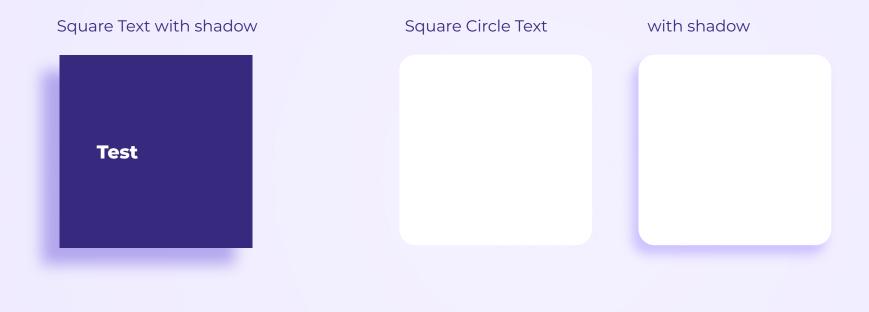


Putting it all together: Creating an Epoch

Periodically, the server releases an Epoch

- 1. Make a snapshot of the address keys DB
- 2. For each address, compute the vrf hash: (hash, proof) = Vrf.hash(address.Email)
- 3. For each address compute the leaf value: leaf = Hash(address.SKL)
- 4. Create a merkle tree with all leaves, where the path of each leaf is derived from the VRF hash.
- 5. Publish the root of the merkle tree

01 Guide Lines Graphics Light Background



02 Guide Lines Graphics Light Background

Image with shadow



No shadow



Top Line Light

Bottom Line Light 06 Guide Lines Graphics Dark Background

 $\overline{\mathbf{O}}$ **ļ**ļļ Ę (\mathbf{I}) Œ Ē Q Ś \bigcirc **{** Ð \$ € £ ₹ ¥ $\mathbf{+}$ \bigcirc $\langle \rangle$ ▶ ₹ \triangle \bigcirc \checkmark $\mathbf{\vee}$ \wedge 0 P ି £ 😣 B Ð Ì ſĴ, $\langle \rangle$ ர R= Ç.) \square (+) \propto° \odot + P ப **□ 1 2 0 2 2** \odot \bigcirc (•)) → ↑↓ $\overline{\odot}$ \bigcirc € ĺn

06 Guide Lines Graphics Light Background

 $\textcircled{O} \rightarrow \textcircled{O} \rightarrow \downarrow \land \land \leftarrow \downarrow \leftarrow \rightarrow \land \textcircled{O} \bigcirc \bigcirc \bigcirc$ ŶţŶ Ę $(\mathbf{1})$ Ē Q 9 **63** Ð \$ E £ ₹ \vee \wedge \langle \rangle \bigcirc ¥ Ð $\mathbf{\bullet}$ Ł \bigcirc $(\mathbf{ })$ \checkmark 0 Q \mathcal{O} ÷ R⁼ \mathcal{D} Ð $\mathbf{\Theta}$ ピ 1 C (+)ഹ്ല Ø Ŵ \mathbf{C} መ \square \odot + \odot \odot \bigcirc în Ð

04 **Guide Lines** Assets



31

9

Proton





Proton Calendar

Proton Drive

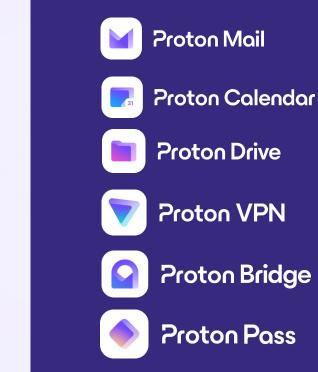


Proton VPN





Proton Proton



31

05 Logo Guide Lines

Small

Proton | Privacy by default

Medium



Large



Proton | Privacy by default

Privacy by default

