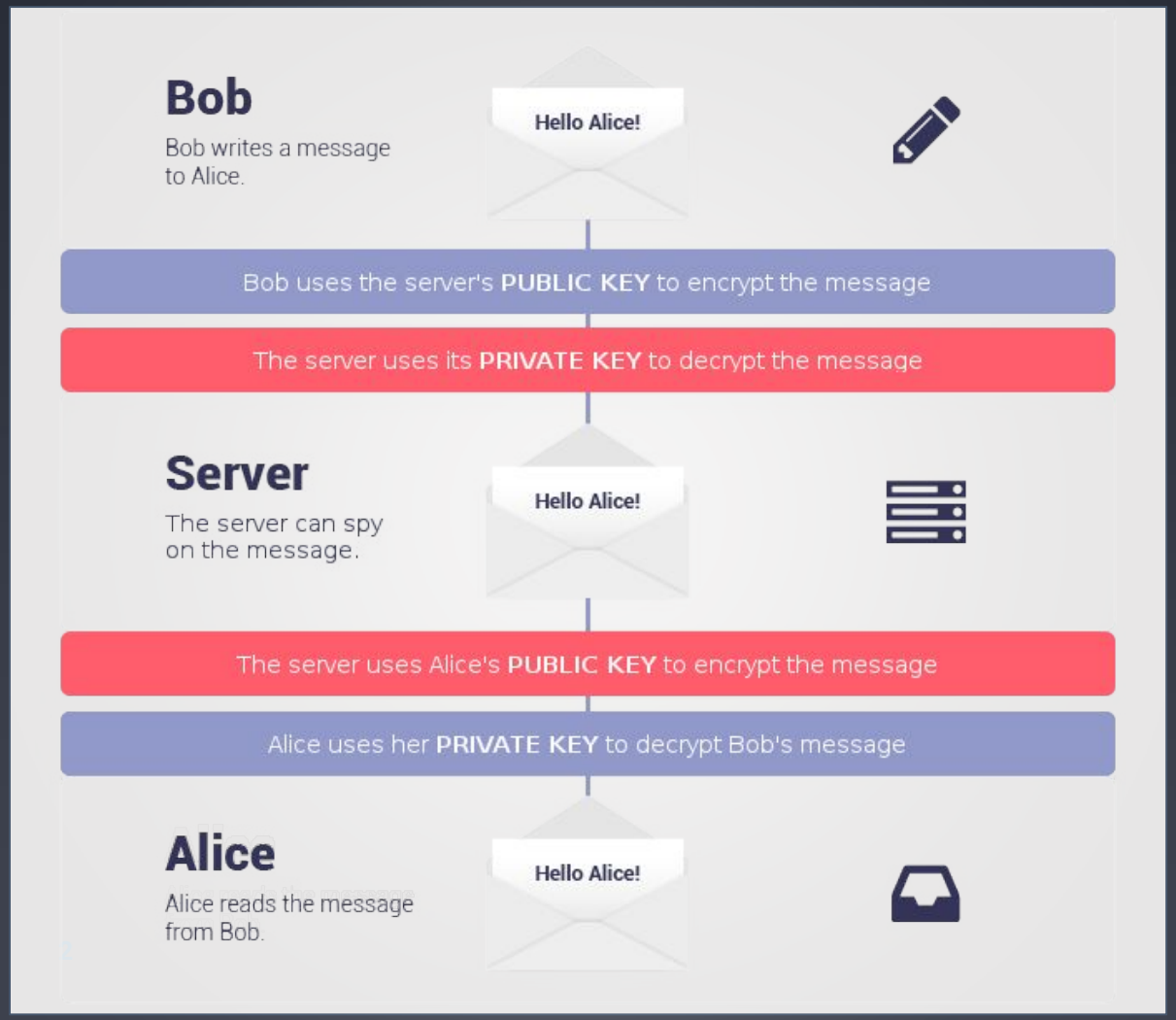
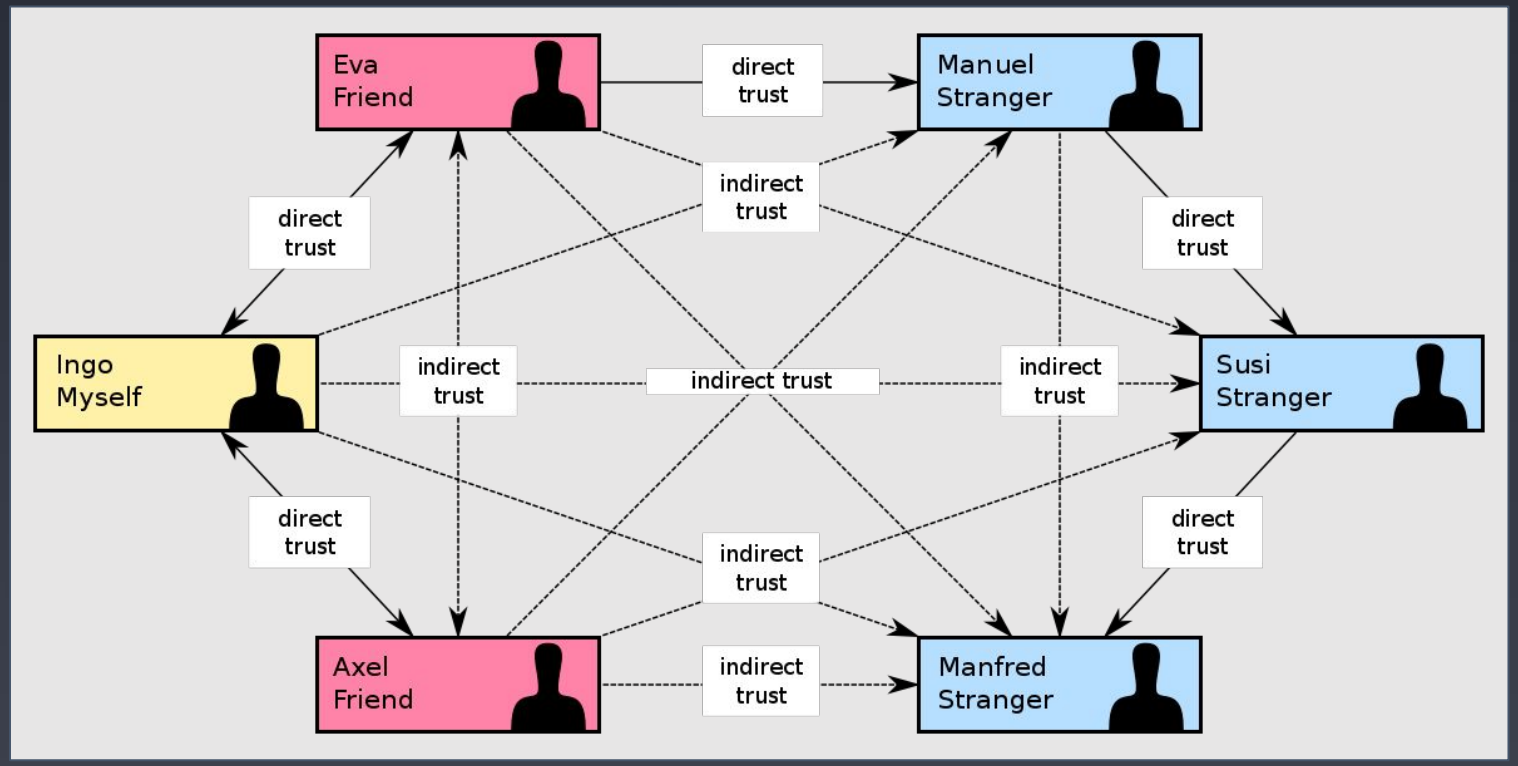
Key Transparency

Why Key Transparency?



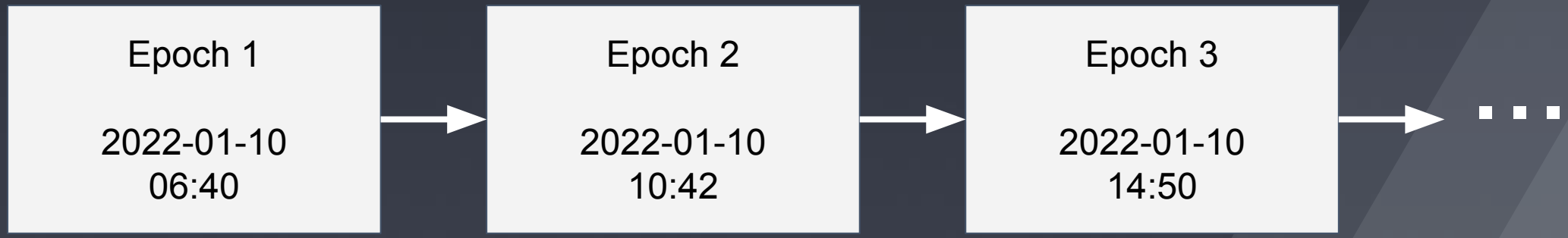
- Server provides a fake version of Alice's key to Bob
- Server can intercept all communication between them transparently
- Same for signatures
- Relevant for WKD where the domain controls mail delivery and key distribution

Why Key Transparency?



What is Key Transparency?

Publishing a **snapshot** of all the addresses and keys (epoch) that is **easily verifiable and identical for every user**



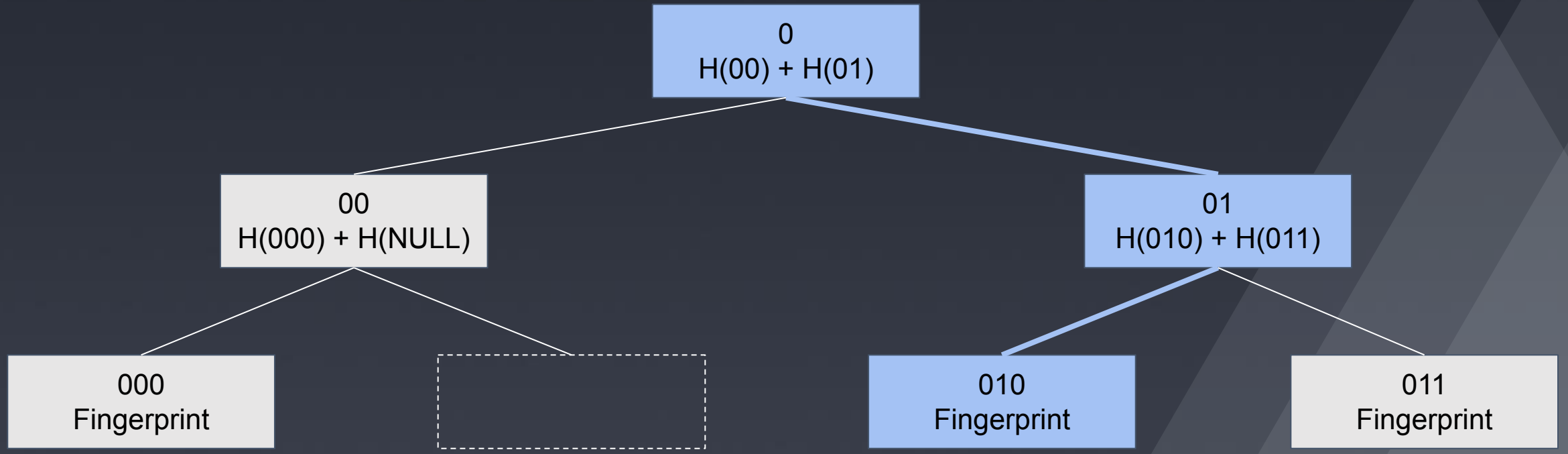
Principle: I can verify my own address and everyone is seeing the same

General principles behind KT

- When requesting keys for an email we return the corresponding epoch hash, the revision number, and a proof
  - If they all verify then the user can be sure that this same data is being provided to all parties

- We publish a commitment file and hash that we can’t change
  - Auditors can verify that our commitment is consistent without knowing the emails, keys, or proofs.

Merkle trees!



The leaf path is determined from the email

```
VRF("test@pm.me") = 010
```

Each address is mapped to a single leaf

Client verification tasks

In order to ensure an address is correct in KT clients must ensure that the fingerprint in KT matches the local key.

If verification fails, we will warn the user and explain that their public keys could not be verified.

Auditor verification tasks

External auditors can verify that:

- The epoch IDs are consecutive and unique
- The chain hashes match the ones received by other auditors (via gossiping)

Questions?