Privacy

Thierry Sans

Privacy Goals

Why privacy matters

Enabling corporate secrecy

Enabling business by preventing exposure of proprietary financial data

Ensuring personal safety

Publicly visible holdings and transactions make users easy targets for scams, phishing, and theft

Ensuring personal freedom

By preventing surveillance and tracking by corporations and governments

Ensuring privacy in financial transactions

Sender Privacy - The identity of the transaction initiator cannot be determined
Observers cannot reliably identify which user authorized or signed the transaction

Receiver Privacy - The recipient of a transaction cannot be identified
Observers cannot link funds to the beneficiary's real identity

Amount Confidentiality - The value transferred is hidden
Observers cannot see how much currency or tokens were sent

Unlinkability - Multiple transactions cannot be linked to the same user
Even if a user participates multiple times, observers cannot tie activities together

Forward Secrecy - Compromise of a user's long-term private key does not reveal past transactions. Previously received or sent funds remain private even if keys are exposed later

The two-side of the coin

The same Privacy-Enhancing Technology (PET) can also enable

- Money laundering
- Sanctions evasion
- Terrorist financing
- Ransomware payouts
- Tax evasion
- Fraud concealment

➡ Technology is neutral, **but use is not**

The Transparency Problem in Blockchain

Ethereum - Full Traceability

The is no privacy on *Ethereum* since everyone can see

- Addresses and their balances
- Contract state
- Call data (i.e transactions)
- Events (a.k.a logs)

Bitcoin - Pseudonymity, Not Privacy

There is partial and fragile privacy on *Bitcoin*

- Address reuse is allowing straightforward traceability
- Graph and heuristics can be used to link inputs and outputs

➡ Chain analysis companies reconstruct identities

Know Your Customer (KYC)

In many jurisdictions, some services are required to collect customer's identification (a.k.a KYC)

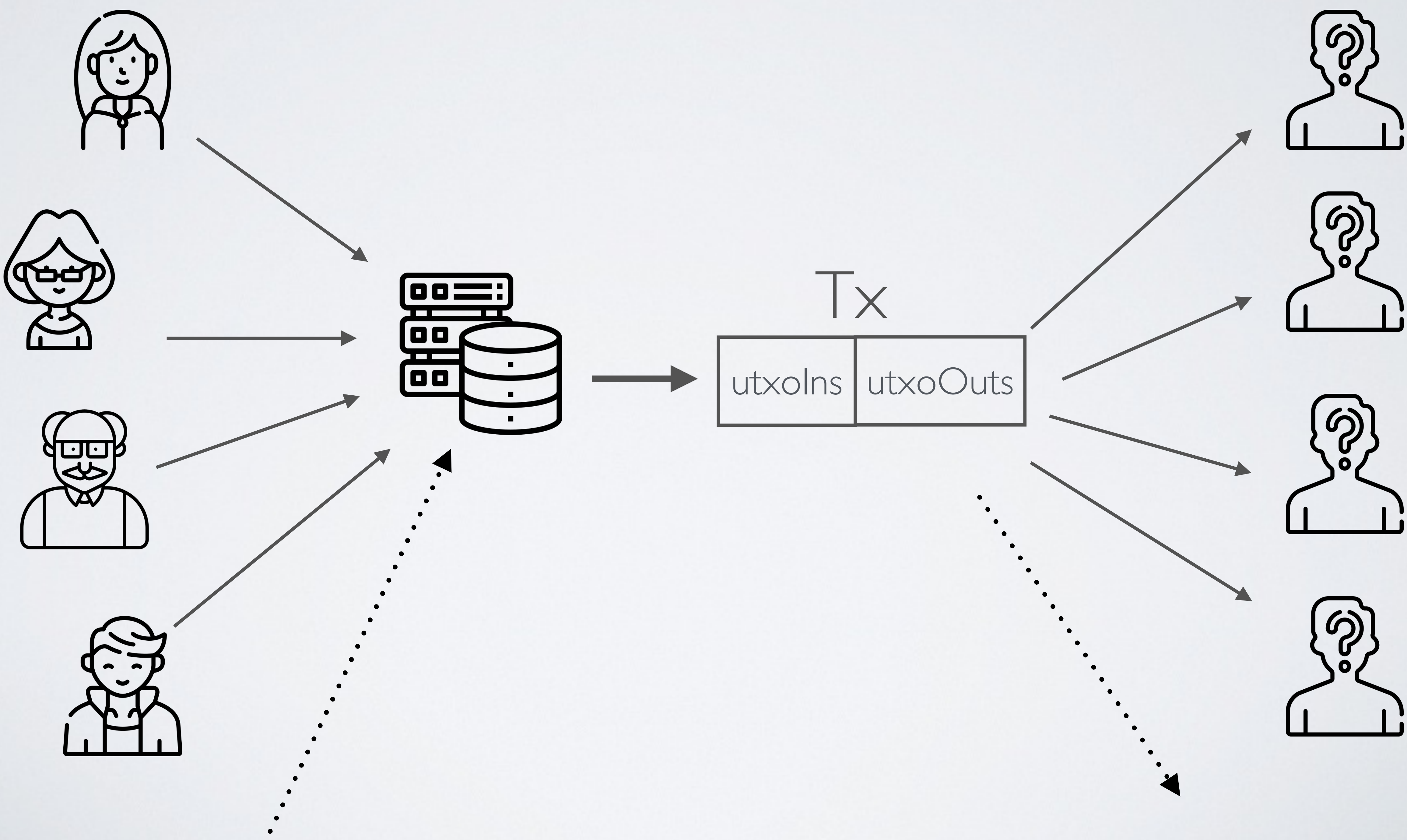
- Centralized Exchanges
- On-Ramp and Off-Ramp Services
  (to convert cryptocurrencies to fiat money and vice versa)

Major Privacy Approaches

Bitcoin CoinJoin - Concept of **Mixer**

Multiple users combine inputs in one transaction

- Still vulnerable to graph and heuristics approach but it blur things



Ethereum Mixer

The mixer is implemented as a smart contract

➡ Use Zero-Knowledge proofs
with a commitment and nullifier scheme

Example: *Tornado Cash*

✓ This is assignment #3

Taking Zero-Knowledge Proofs further

Zcash
Same idea as a mixer and blend into a utxo-based blockchain

Aztec
Adding smart contract support through the *Noir* programming language

SGX-Based Privacy

Encrypt information (balances, contract state and so on) on the blockchain and use special hardware to decrypt and compute data

➡ Validators use hardware-based security technology (Intel SGX mostly) to isolate sensitive code and data in protected, encrypted memory regions called enclaves

✓ Prevent validators from dumping the memory to look into the data

◉ Require complex key distribution and prone to hardware vulnerabilities

Example:

- *Secret Network*
- *Oasis Network*

Fully Homorphic Encryption (FHE)

Fully Homomorphic Encryption (FHE) cryptographic technique allowing computations to be performed on encrypted data without decrypting it

✓ No need for specialized hardware

◉ Yet very limited set of operations

Example: Fhenix

Monero

1. **Ring Signatures**
    - Hide sender among N decoys
2. **Stealth Addresses**
    - One-time receiver addresses
3. **Confidential Transactions (RingCT)**
    - Using zero-knowledge proof with a commitment scheme (a.k.a proof of commitment)

Legal Issues

- *Monero* and *Zcash* are being delisted from regulated exchanges
- Mixers are illegal in many jurisdictions
- The US DOJ (Department of Justice) has lead many legal actions against entities "supporting" those services
- *Tornado Cash* creators were arrested for facilitating money laundering

Yet, Privacy is a priority for Ethereum

Stealth Addresses (ERC-5564)

Senders can derive a <u>fresh and unlinkable</u> *address* so recipients can receive funds privately

Creation of the Privacy Stewards of Ethereum (PSE) & Ethereum Foundation Privacy Roadmap

Build privacy features <u>with selective disclosure</u> in Ethereum

Zero-Knowledge Proofs

ZK proofs in a nutshell

A Zero-Knowledge Proof lets a **prover** proves to a verifier it <u>knows a secret without revealing it</u>

1. **Proof generation**
   The prover generates a zero-knowledge proof <u>with a secret input</u>

2. **Proof verification**
   The verifier verifies the proof <u>without the secret input</u>

➡ The verifier **does not know the secret** (privacy) but is convinced that the prover knows the secret since it can prove it using a ZK-proof

Two types of zero-knowlege proofs

Interactive proofs
A back-and-forth conversation to prove something

Non-interactive proofs
A single message to prove something

- e.g. digital signature, **ZK-snarks**, ZK-starks

Zero-Knowledge Proofs using zk-SNARK

Arithmetic Circuit $\mathbf{C}(w_{priv}, w_{pub})$

Proof Generator $\mathbf{G}$
Proving Key p_k
$pf = G(w_{priv}, w_{pub}, p_k)$

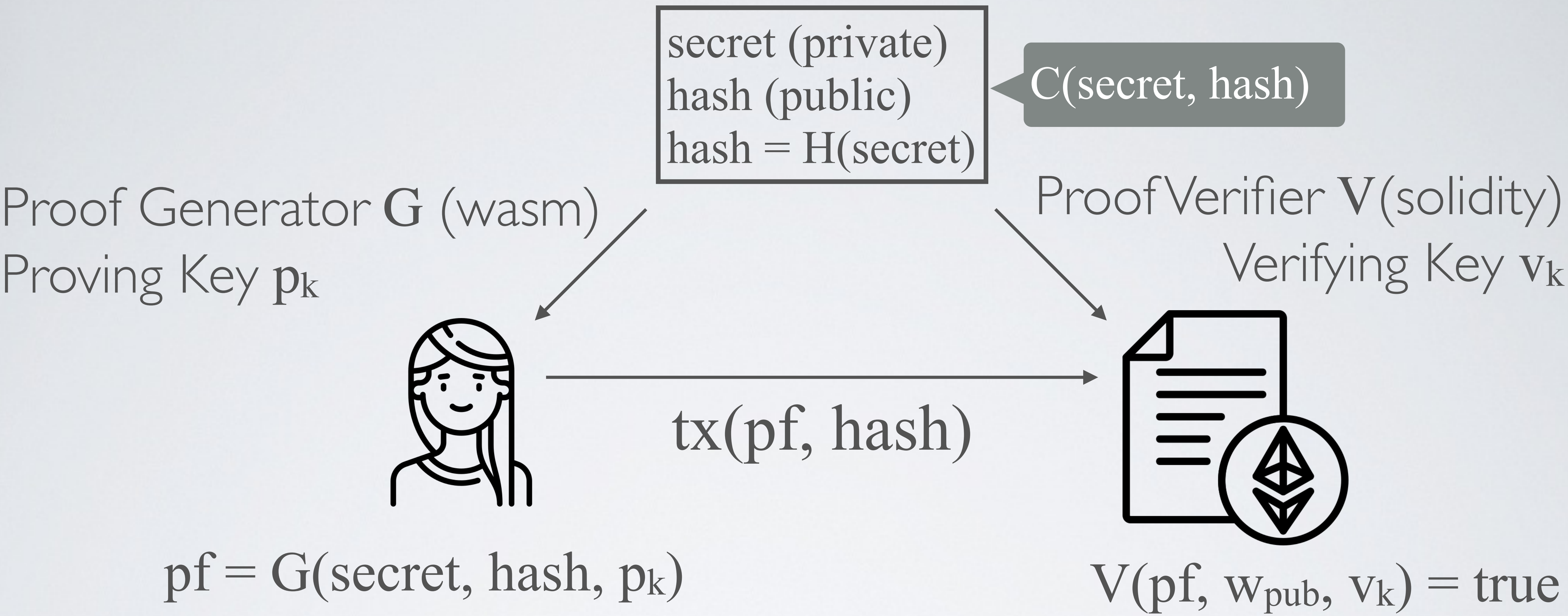
Proof Verifier $\mathbf{V}$
Verifying Key v_k
$V(pf, w_{pub}, v_k) = true$

✓ **Soundness**
can always generate a valid proof pf knowing w_{priv}, w_{pub}

✓ **Completeness**
Cannot generate a valid proof pf knowing w_{pub} only

✓ **Zero-Knowledge**
Verifying pf using w_{pub} does not reveal anything about w_{priv}

Proof of Commitment



The generator code is compiled as wasm module (Web Assembly)

The verifier code is compiled into a solidity smart contract

➡ Alice can prove that she knows the secret input without revealing it

✓ The proof and the hash in the transaction does not reveal anything about the secret