

KU LEUVEN



Proof of Seed

ZK-Based (Post-)Quantum Migration

Without Address Changes

Mahdi Sedaghat

zkSummit · Rome · May 7



Post-Quantum Readiness in EdDSA Chains

Foteini Baldimtsi^{1,2}, Kostas Kryptos Chalkias¹,
Arnab Roy¹, and Mahdi Sedaghat^{3,4}

¹ Mysten Labs, {foteini,kostas,arnab}@mystenlabs.com

² George Mason University

³ COSIC, KU Leuven, ssedagha@esat.kuleuven.be

⁴ Soundness Labs



ZK Episode 399

IS ZK DEAD? OR HAS IT JUST BEGUN?
WITH THE ZK POD CO-HOSTS



ZK Episode 363

BRINGING ZK TO GOOGLE WALLET
WITH ABHI AND MATTEO



ZK Episode 365

ZK IN SUI & ZKAT
WITH KOSTAS KRYPTOS



ZK Episode 383

COUNTDOWN TO Q-DAY WITH PROJECT 11



IS ZK DEAD
WITH

ZK Episode 400

QUANTUM ADVANCES, HYBRID SIGNATURES
& SNARKS TO THE RESCUE
WITH DAN BONEH



TARUN

ANNA

BRINGING
WI



DAN

ANNA

T 11



ABHI

MATTEO

NICO

ANNA

CONOR

ALEX

ANNA

Blockchain: A great bounty for quantum adversary



bc1ql-859v2

Bech32 (P2WPKH)



Bitcoin Address

bc1ql49ydapnjaf15t2cp9zqpjwe6pdgmxxy98859v2

Bitcoin Balance

140574.82562097 • \$14,409,417,261

- 6.9M BTC in dormant wallets can't be moved
- Hard forks require social consensus (years)
- Address changes break immutability guarantees
- Users will lose the migration calls

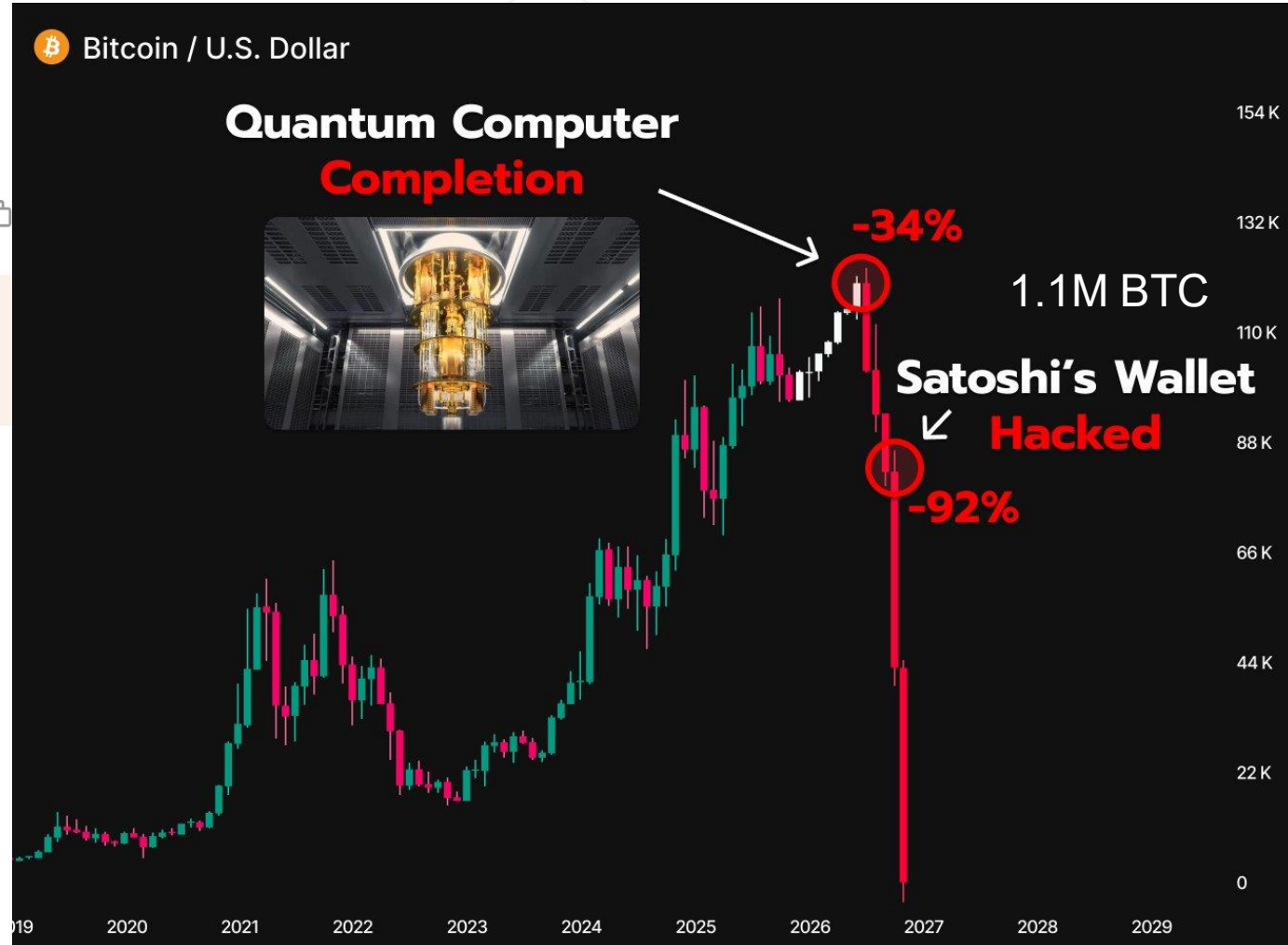
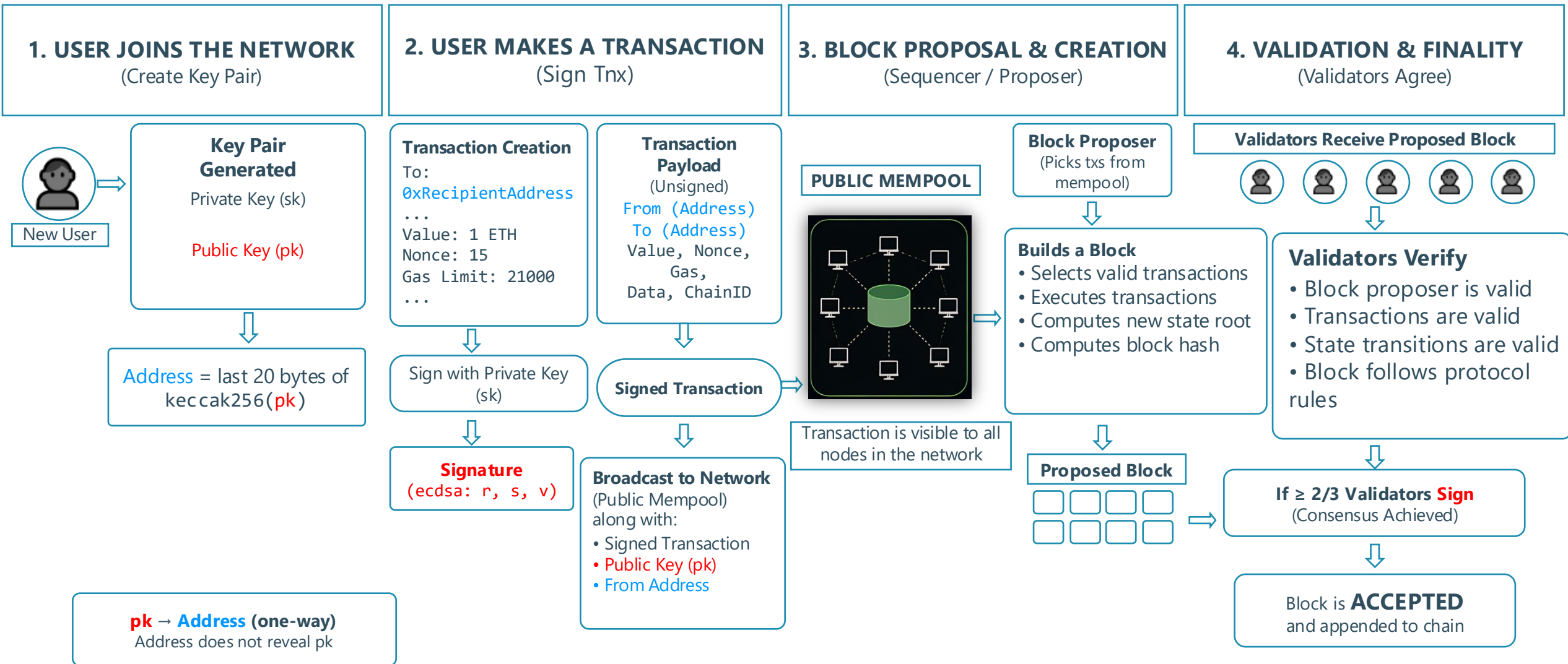


Image by K Karagiannis

How Blockchain works? A quick reminder



Quantum Vulnerability Map

■ Broken
 ■ Partial
 ■ Safe

Which blockchain primitives break under quantum attack?

Shor's algorithm



Grover's algorithm



Component	Risk	Quantum Attack	Impact on Blockchain
Digital Signatures ECDSA / EdDSA	BROKEN	Shor's algorithm solves ECDLP in polynomial time	Forge any transaction, steal funds, impersonate wallets
Multisignatures BLS (PoS consensus)	BROKEN	Shor's algorithm breaks BLS pairings; no PQ aggregate signatures yet	PoS validator sets compromised; forge attestations, finality attacks
Consensus (PoW) Mining puzzles	PARTIAL	Grover's quadratic speedup on hash inversion	Mining advantage; mitigated by difficulty adjustment
Address Generation Hash(PubKey)	PARTIAL	Safe if PubKey never exposed; vulnerable after first spend	Unspent P2PKH outputs protected; reused addresses at risk
Merkle Trees Hash-based	SAFE	Relies on hash preimage resistance (Grover: manageable)	Transaction integrity preserved with longer hashes
Hash Functions SHA-256, Keccak, Blake2, ...	SAFE	Grover's gives only quadratic speedup (128-bit still hard)	Mining difficulty needs doubling at most; no structural break

The Ideal PQ Upgrade Path

✗ Naïve Solution

Switch to PQ signature → requires **asset transfers** + **address rotation**



✓ Our Goal

Backward-compatible PQ upgrade → **no address changes needed**

1 Preserve Existing Addresses

No address changes or asset transfers needed

2 Exposed Keys OK

Works even when public key is already revealed onchain

3 Protect Sleeping Accounts

Sleeping/lost accounts secured retroactively

4 Key-Gen Agnostic

Works regardless of key generation method -> self-custody to custody



Lattice-based
Code-based
Multivariate
Isogeny-based

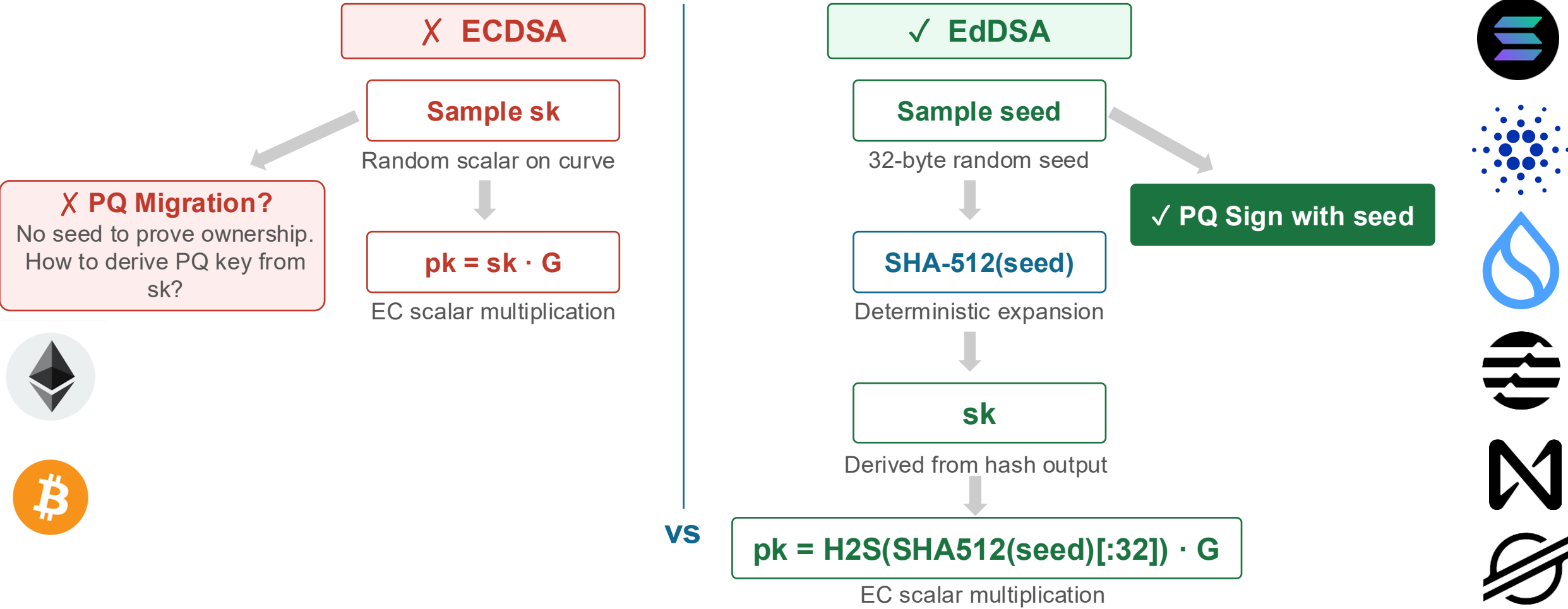


Hash-based

Backward Compatibility in Blockchain

Why EdDSA-based chains are structurally better
positioned for PQ migration

EdDSA PQ-better than ECDSA? Self-custody/Software wallets



Bottomline: EdDSA's seed-based key generation creates a natural PQ upgrade path that ECDSA lacks

Core Idea: Seed as ZK Witness

EdDSA derives sk from seed via SHA-512. The **seed** remains quantum-safe and serves as a **ZK witness**.

PQ-NIZK Relation:

$$\text{Rel} = \{(pk, \text{msg}, hx) \mid \exists \text{seed s.t.} \\ pk = H2S(\text{SHA512}(\text{seed})[:32]) \cdot G \wedge hx = \text{Hash}(\text{msg}, \text{seed})\}$$

PQ Secure

Hash-based soundness survives quantum attacks

Client-Side Proving

No trusted prover -- seed never leaves device

Memory Efficient

Runs on mobile / browser without server

Same seed → same address → no migration needed → backward compatible

One-Time Proof Certification

How It Works?

- Set **msg = pqpk** (e.g., Dilithium or Falcon public key)
- Generate **one-time ZK proof** binding PQ key to legacy account
- After on-chain attestation, use standard PQ signatures forever

Key Advantage

- Large proof size acceptable: **generated only once!**
- If PQ scheme later broken, re-generate proof with new pqpk

What we need: Simulation Extractability

- **Non-malleability:** even after seeing simulated proofs, the adversary cannot forge a fresh π for any new statement.
- **Cheap fix:** bind a **sid** into every Fiat-Shamir hash. No proof-size hit.

Summary Table:

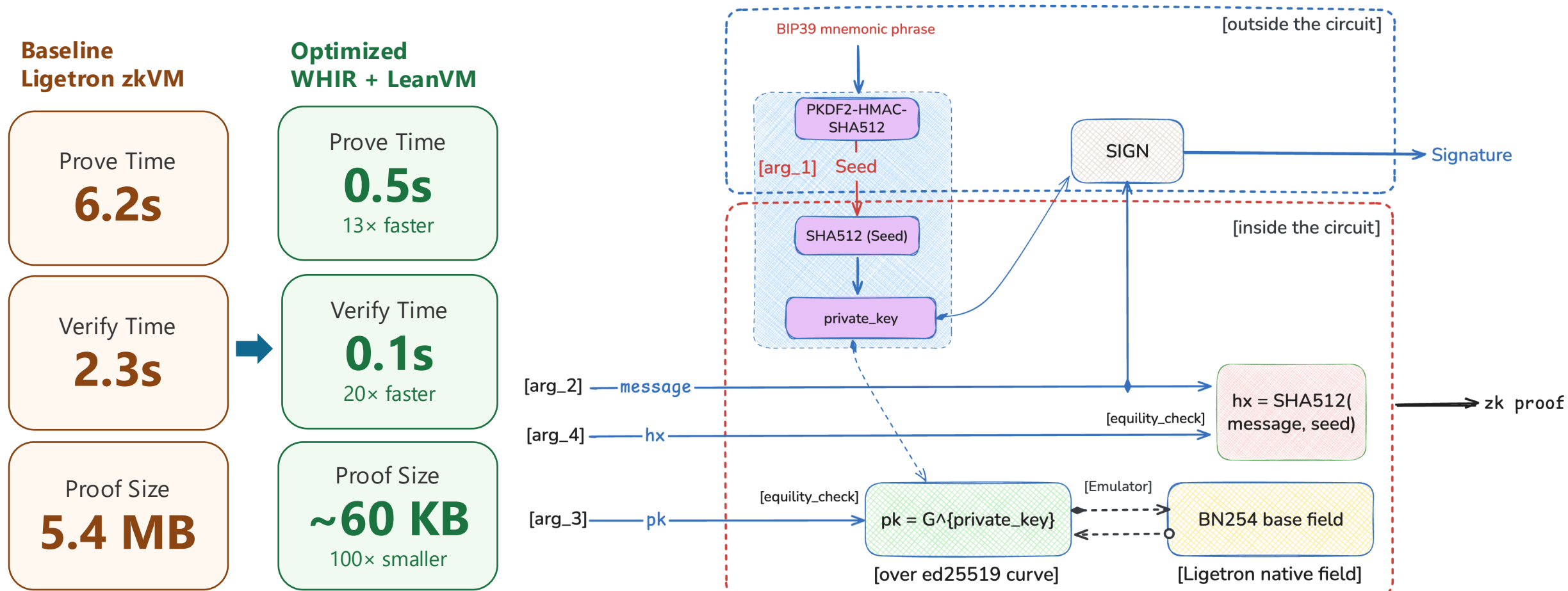
We summarize the 4 strategies (including the naive one) and the properties achieved in the following table

	Plug-and-Play Replacement	Private Keys as Hash Outputs	Move to 2-of-2	Move to 1-of-2
P1 (current security)	NO	YES	YES	YES
P2 (PQ security)	YES (P2a)	YES (P2b)	YES (P2a)	YES (P2b)
P3 (cost now)	NO	YES	NO	YES
P4 (change)	NO	YES	DEPENDS	DEPENDS
NOTES:		PQ signing is not efficient so not a long-term solution. P2a is achieved as long as only hashed addresses are revealed.		Essentially BIP-360 for Bitcoin. P2a is achieved as long as only hashed addresses are revealed.

https://assets.ctfassets.net/sygt3q11s4a9/6EjYavuGdtJDYCqaJrASj9/9f464a8bf26f44bd6c85710fe7e4a29f/Quantum_Computing_and_Blockchain_v10.3_15April2026.pdf

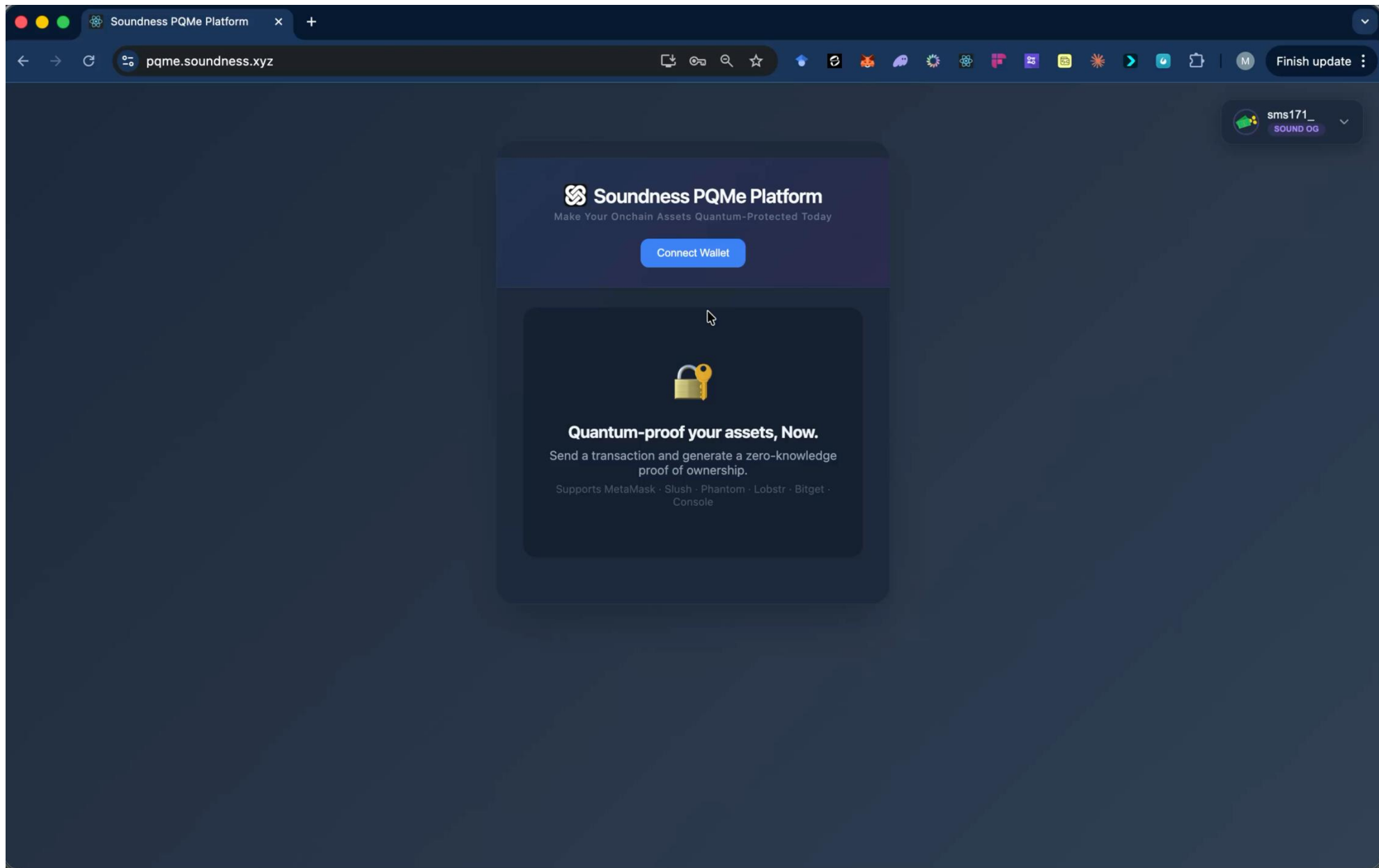
Performance Benchmarks: Ed25519 through SLIP10

All benchmarks on M4 MacBook Pro



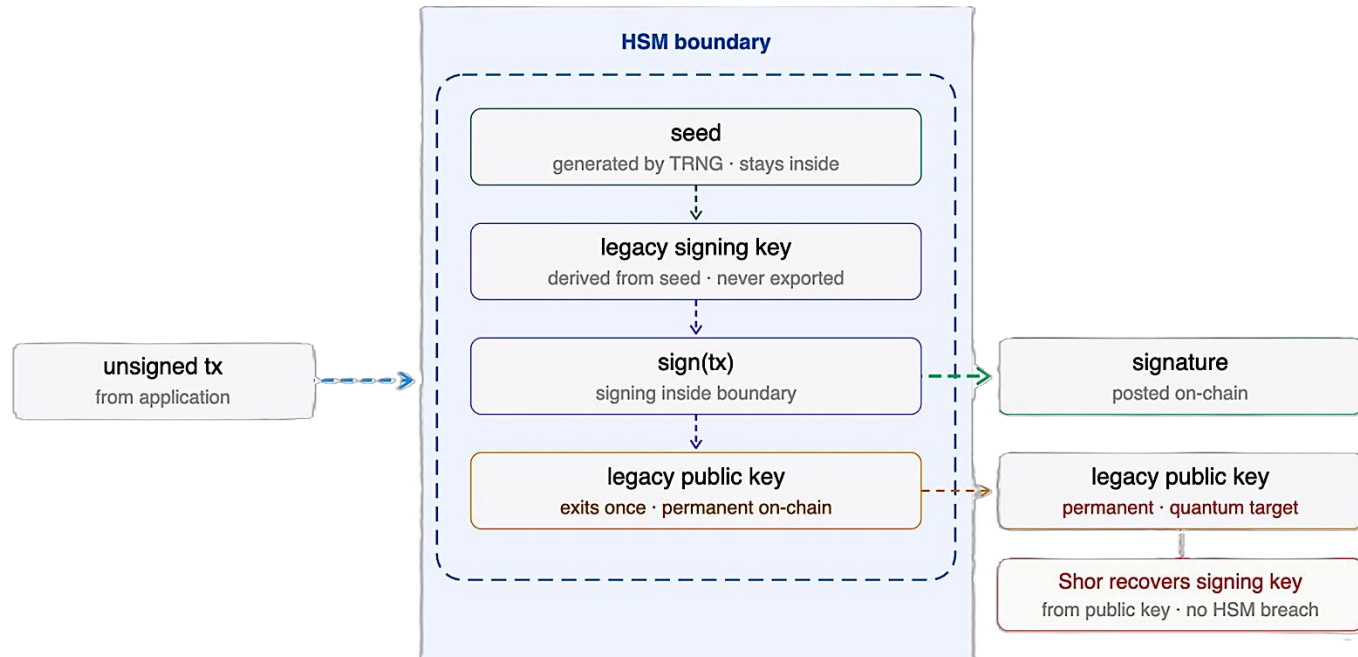
Looking ahead (decomposition trick). Split the circuit into 2 parts

<https://github.com/SoundnessLabs/pqchain>

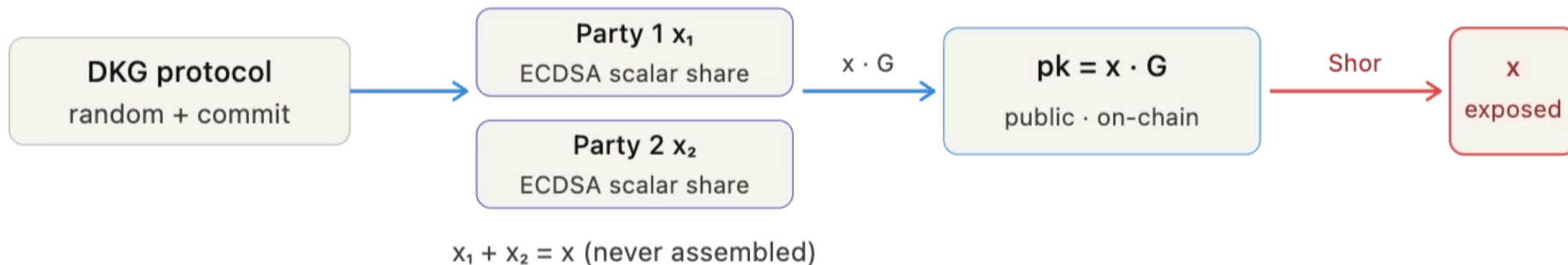


Where this idea doesn't work?

HSM Wallets



MPC Wallets



The Most Immediate Solution for all Wallet Types

From Ephemeral keys to Anonymous Credentials

Hidden-PK Wallets: Quantum-Migration for all account type

Borrowing the core idea of Anonymous Credentials

Anonymous Credentials

“Prove you hold a valid credential, without revealing it.”
The credential stays private; only validity is shown.

Hidden-PK Wallets

“Prove you hold a valid signing key, without revealing it.”
The pk stays private; only spend authority is shown.

Hidden-PK relation

$$\text{Rel} = \{ (\text{pkHash}, e) \mid \exists (\text{pk}, r, s): \text{ECDSA.verify}(\text{pk}, (r, s), e) = 1 \\ \wedge H(\text{pk}) = \text{pkHash} \}$$

Same address forever

No key rotation, no per-tx state,
no migration.

No mempool exposure

The proof reveals nothing about
pk or (r, s), a quantum adversary
has nothing to attack during

Works today

ERC-4337 only. Any EVM chain,
any L2; no protocol changes.

Hidden-PK Wallet

hiddenpk.soundness.xyz

prover online Sepolia

Connect MetaMask

Hidden-PK OFF
Direct eth_sendTransaction

RECIPIENT
0x...

AMOUNT (ETH)
0.0

Send

© 2026 Soundness Labs. All rights reserved.

<https://github.com/SoundnessLabs/longfellow-zk-hiddenpk>

One Idea, Two Constructions

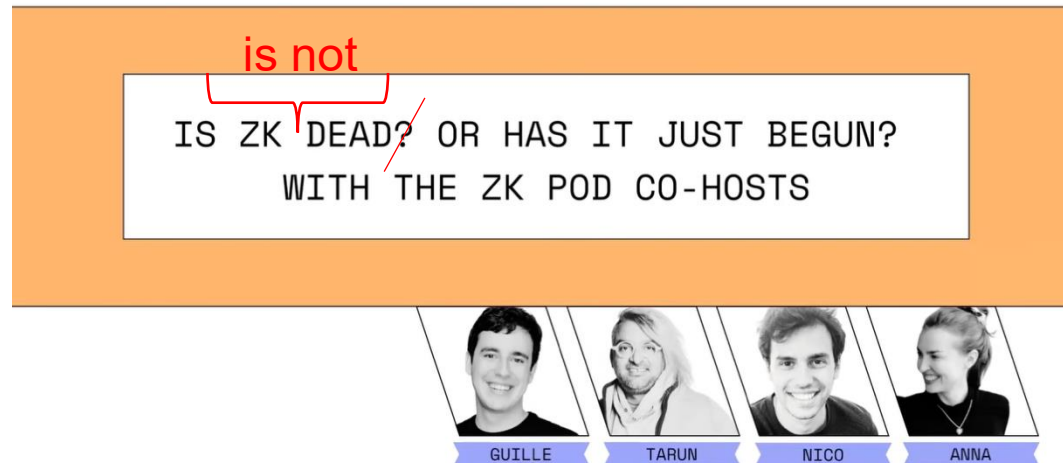
EdDSA chains (Solana, Sui, Near...)

- Seed = PQ-stable witness behind SHA-512.
- One-time PQ-NIZK certifies a fresh pqpk.
- No address change, no hard fork, no migration.

From MPC to HSM:

- Hidden-PK: pk stays inside the proof, address = $H(pk)$.
- Per-tx ZK on Longfellow-ZK, prove ~87 ms on M1.

Episode 399



KU LEUVEN



Thank You!

ssedagha@esat.kuleuven.be

mahdi@soundness.xyz