KU LEUVEN

Soundness

Post-Quantum Readiness in Blockchain:

ZK-Based (Post-)Quantum Migration Without Address Changes

Mahdi Sedaghat

TUM Salon | Munich | June 3

Outline:

1. **Motivation:** The quantum threat to blockchains
2. **Background:** Blockchain, quantum attacks, migration strategies
3. **The EdDSA Advantage:** Why EdDSA chains are better positioned
4. **Our Protocol:** PQ-NIZK seed-based migration for EdDSA
5. **Implementation:** Benchmarks and circuit architecture
6. **Conclusion:** Takeaways and future directions

Post-Quantum Readiness in EdDSA Chains

Foteini Baldimtsi[1,2], Kostas Kryptos Chalkias[1], Arnab Roy[1], and Mahdi Sedaghat[3,4]

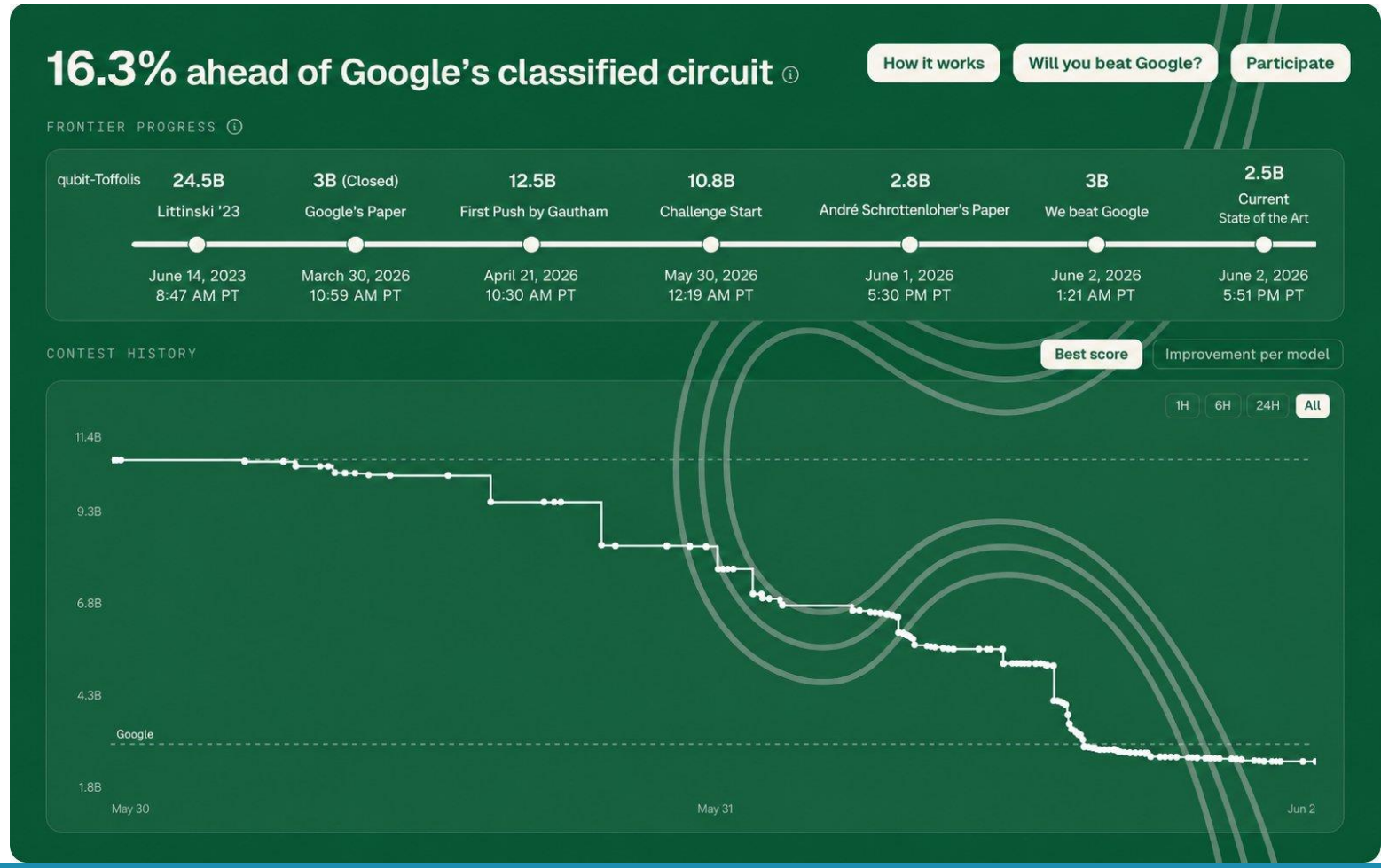
[1] Mysten Labs, {foteini,kostas,arnab}@mystenlabs.com
[2] George Mason University
[3] COSIC, KU Leuven, ssedagha@esat.kuleuven.be
[4] Soundness Labs

AI agents are now improving quantum circuits

The circuit that breaks ECC just got cheaper, by 2 students and AI agents

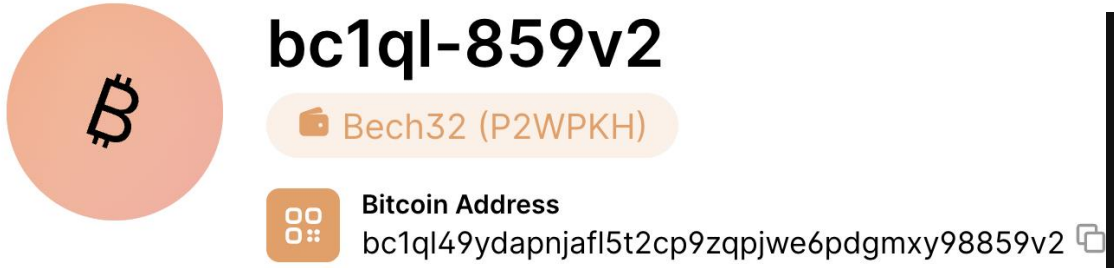
What happened?

- Agents optimized the elliptic-curve circuit Shor needs to break ECDSA.
- No quantum experts; just open benchmarks and agent swarms.
- Google already moved its quantum-readiness date to 2029.



https://www.ecdsa.fail/

Blockchain: A great bounty for quantum adversary



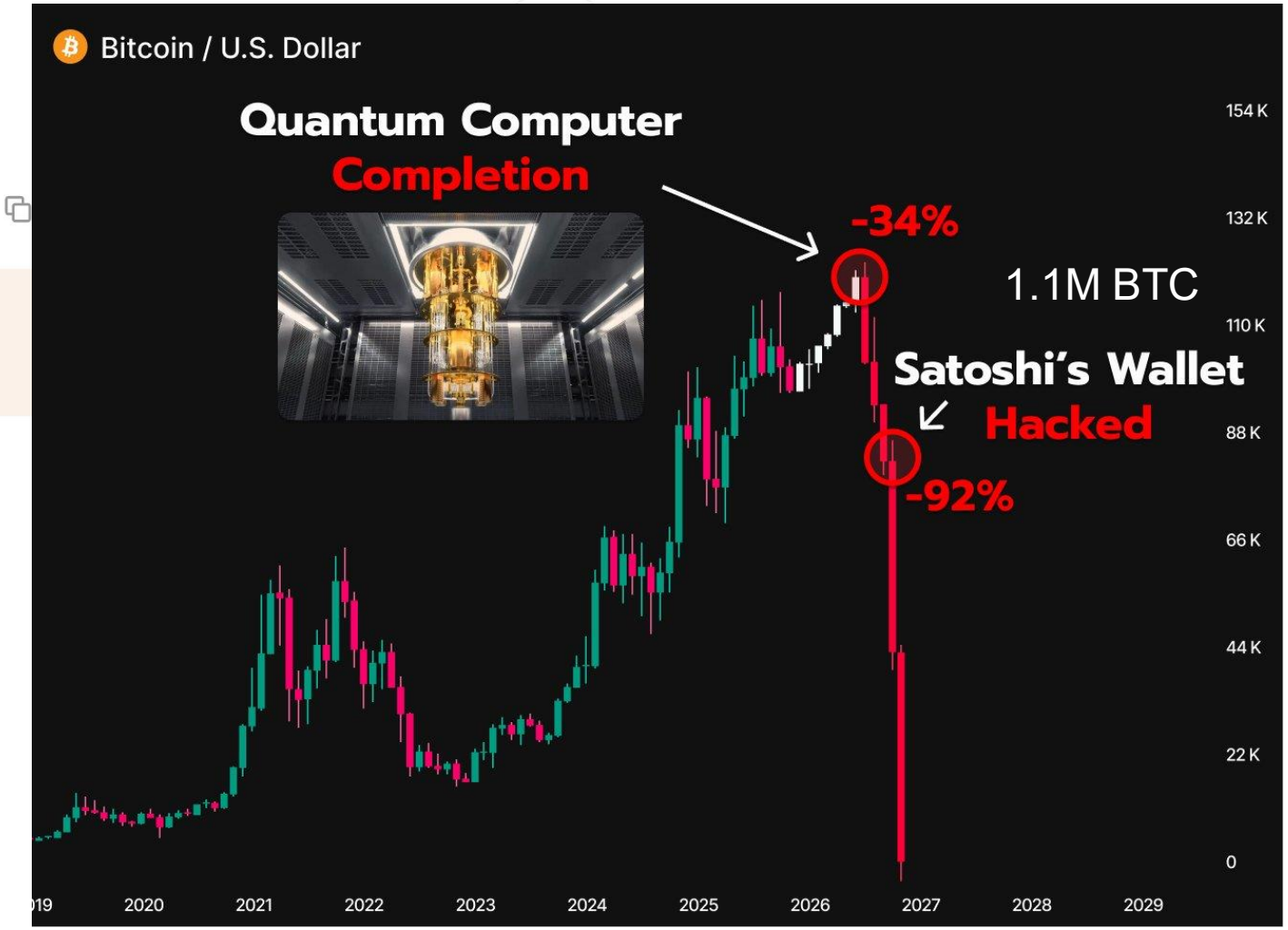


Image by K Karagiannis

- 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

KU LEUVEN

How Blockchain works? A quick reminder

1. USER JOINS THE NETWORK
(Create Key Pair)

New User

Key Pair Generated

Private Key (sk)

Public Key (pk)

Address = last 20 bytes of keccak256(pk)

pk → Address (one-way)
Address does not reveal pk

2. USER MAKES A TRANSACTION
(Sign Tnx)

Transaction Creation

```
To:
0xRecipientAddress
...
Value: 1 ETH
Nonce: 15
Gas Limit: 21000
...
```

Sign with Private Key (sk)

Signature
(ecdsa: r, s, v)

Transaction Payload
(Unsigned)

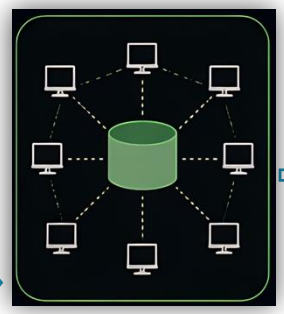
```
From (Address)
To (Address)
Value, Nonce,
Gas,
Data, ChainID
```

Signed Transaction

Broadcast to Network
(Public Mempool) along with:
- Signed Transaction
- Public Key (pk)
- From Address

3. BLOCK PROPOSAL & CREATION
(Sequencer / Proposer)

PUBLIC MEMPOOL



Transaction is visible to all nodes in the network

Block Proposer
(Picks txs from mempool)

Builds a Block
- Selects valid transactions
- Executes transactions
- Computes new state root
- Computes block hash

Proposed Block

4. VALIDATION & FINALITY
(Validators Agree)

Validators Receive Proposed Block

Validators Verify
- Block proposer is valid
- Transactions are valid
- State transitions are valid
- Block follows protocol rules

If ≥ 2/3 Validators Sign
(Consensus Achieved)

Block is **ACCEPTED**
and appended to chain

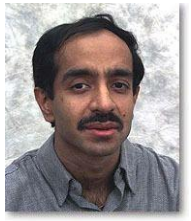
Quantum Vulnerability Map

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**

Preserve Existing Addresses

No address changes or asset transfers needed

Exposed Keys OK

Works even when public key is already revealed onchain

Protect Sleeping Accounts

Sleeping/lost accounts secured retroactively

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

✗ ECDSA

Sample sk

Random scalar on curve

pk = sk · G

EC scalar multiplication

✗ PQ Migration?

No seed to prove ownership. How to derive PQ key from sk?

vs

✓ EdDSA

Sample seed

32-byte random seed

SHA-512(seed)

Deterministic expansion

sk

Derived from hash output

pk = H2S(SHA512(seed)[:32]) · G

EC scalar multiplication

✓ PQ Sign with seed

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:

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

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.

Performance Benchmarks: Ed25519 through SLIP10

All benchmarks on M4 MacBook Pro

Baseline
Ligetron zkVM

Prove Time
6.2s

Verify Time
2.3s

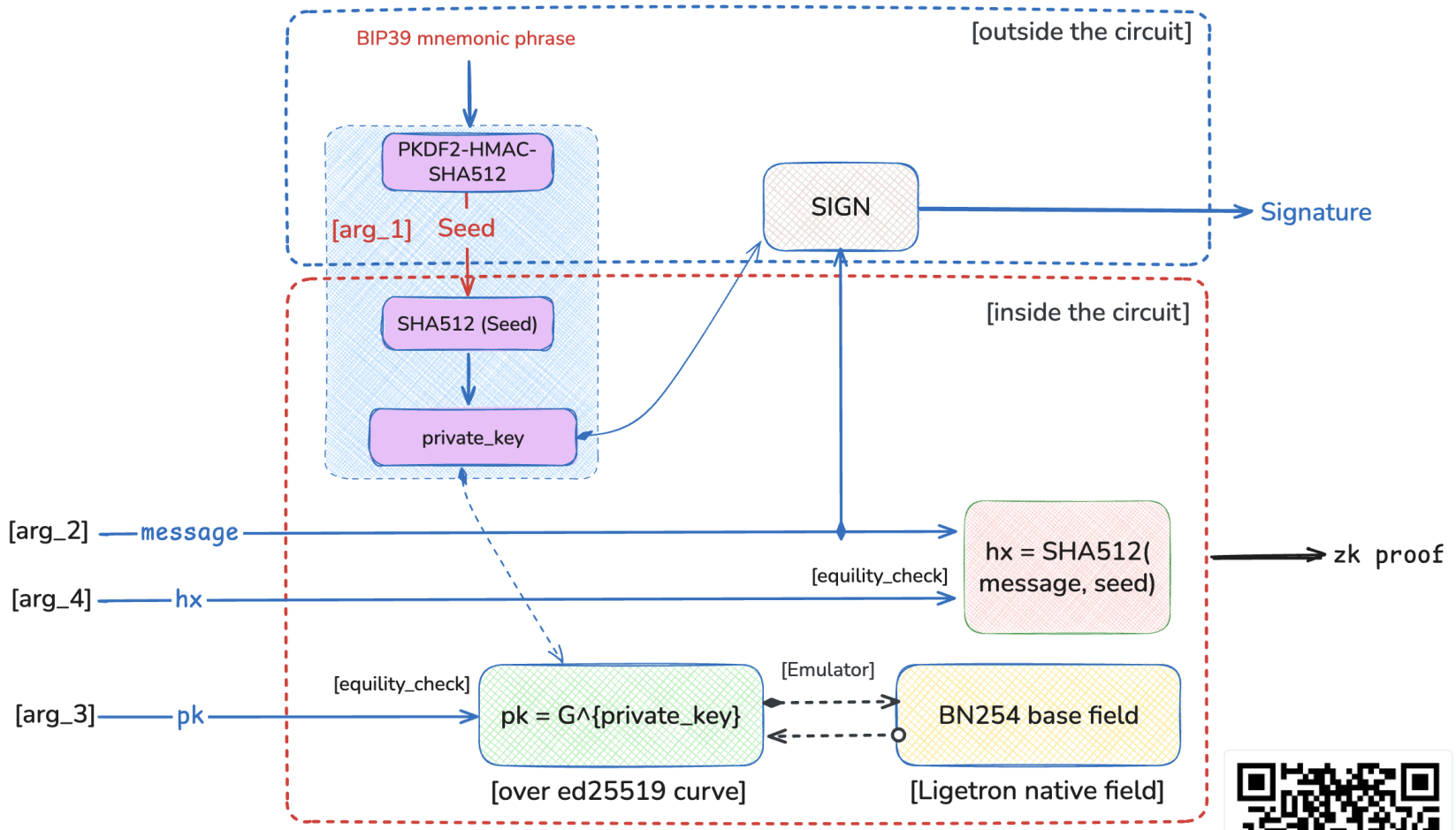
Proof Size
5.4 MB

Optimized STARKs
With WHIR-P3 PCS

Prove Time
0.5s
13× faster

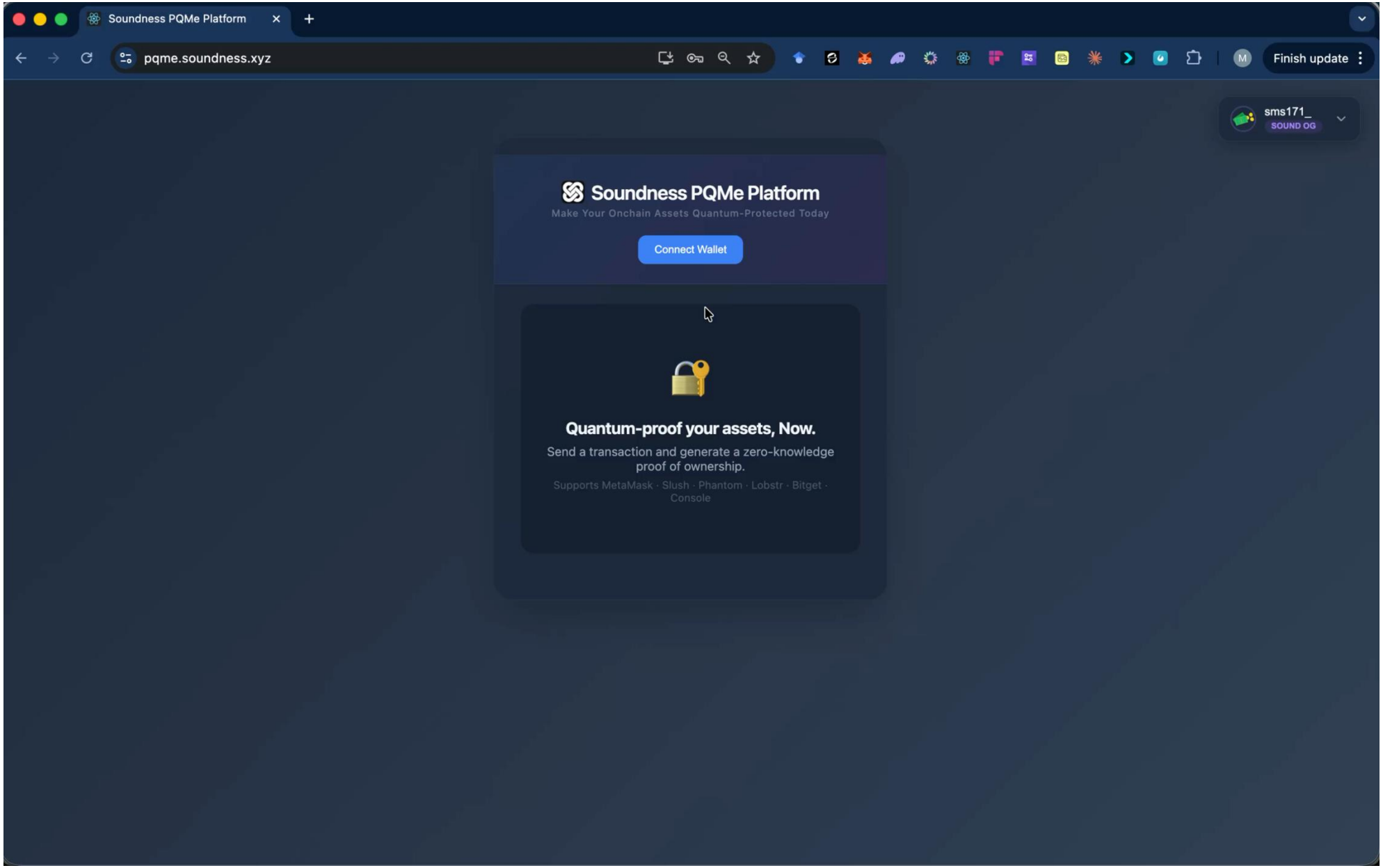
Verify Time
0.01s
200× faster

Proof Size
~90 KB
60× smaller



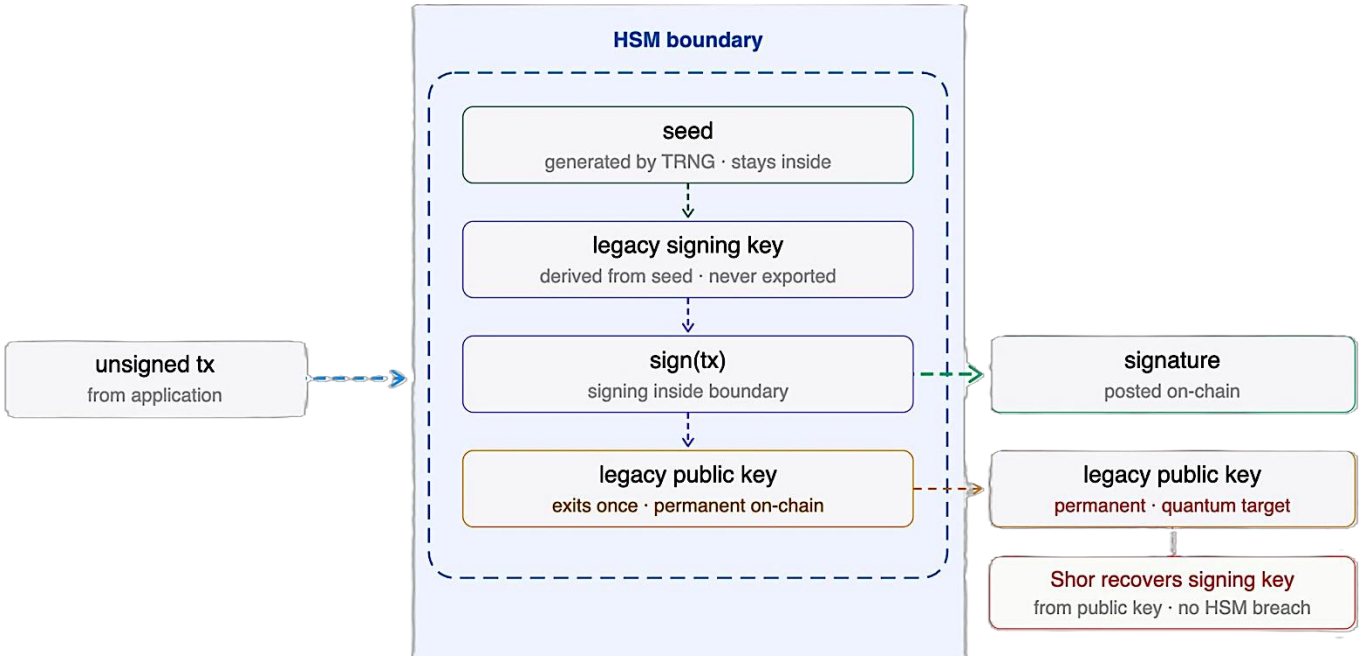
Looking ahead (decomposition trick). Split the circuit into 2 parts

KU LEUVEN

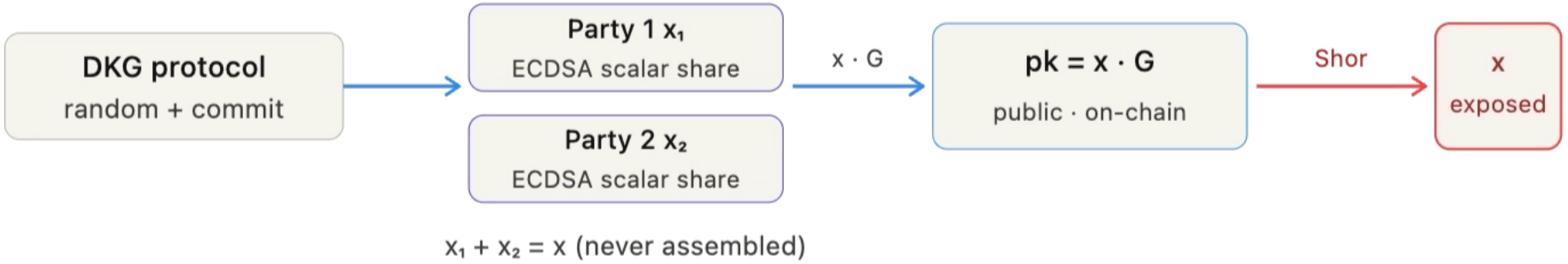

Soundness PQMe Platform
pqme.soundness.xyz
Finish update
sms171_
SOUND OG
Soundness PQMe Platform
Make Your Onchain Assets Quantum-Protected Today
Connect Wallet
Quantum-proof your assets, Now.
Send a transaction and generate a zero-knowledge proof of ownership.
Supports MetaMask · Slush · Phantom · Lobstr · Bitget · Console

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

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

Same address forever

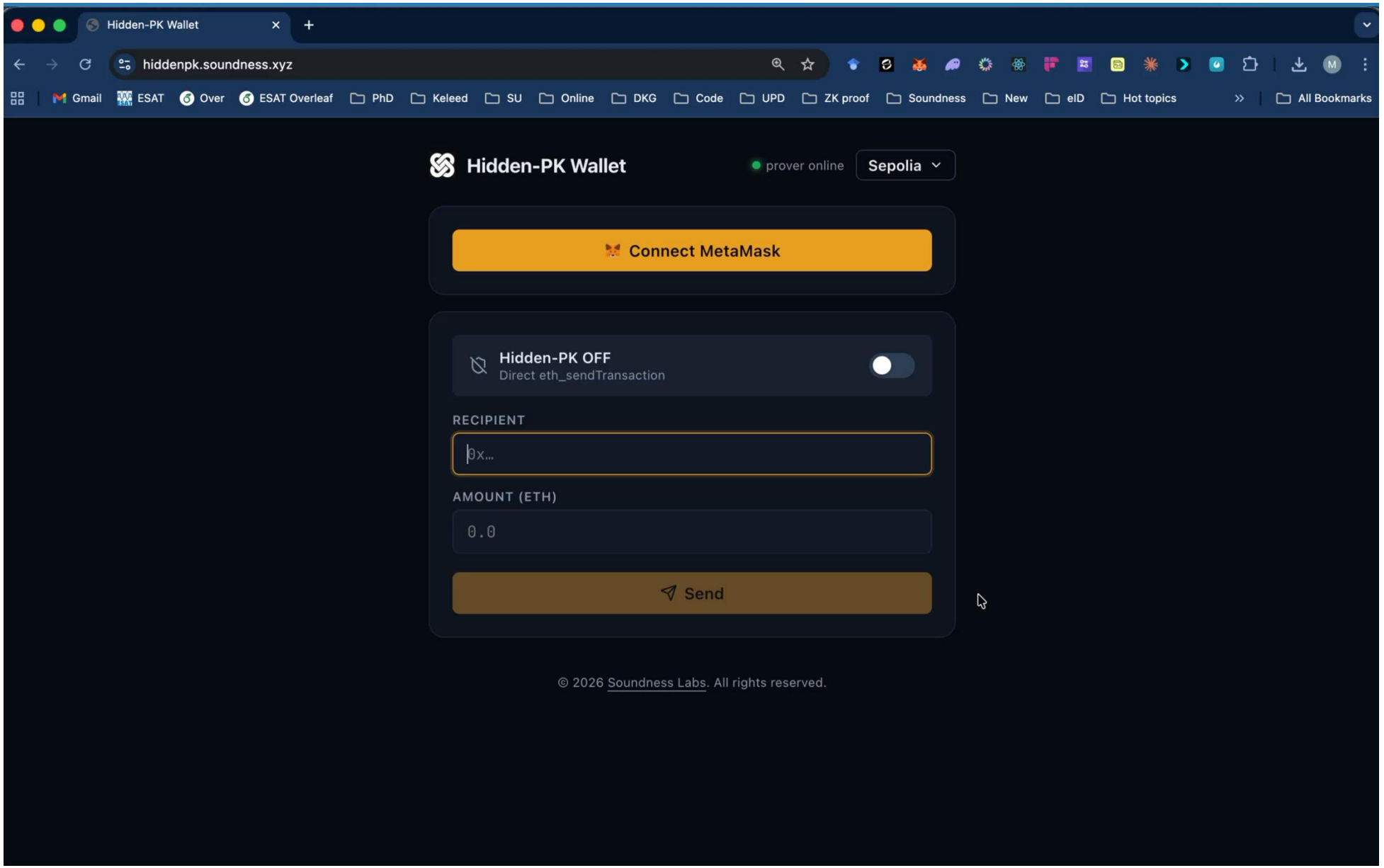
No key rotation, no per-tx state, no migration.

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
Gmail
ESAT
Over
ESAT Overleaf
PhD
Keleed
SU
Online
DKG
Code
UPD
ZK proof
Soundness
New
eID
Hot topics
All Bookmarks
Hidden-PK Wallet
prover online
Sepolia
Connect MetaMask
Hidden-PK OFF
Direct eth_sendTransaction
RECIPIENT
0x...
AMOUNT (ETH)
0.0
Send
© 2026 Soundness Labs. All rights reserved.

KU LEUVEN

Key Takeaways

One idea, two constructions. A PQ-stable witness behind the signature makes EdDSA chains quantum-ready, and the same trick carries from MPC to HSM.

Nothing changes on-chain. No address change, no hard fork, no migration, the public key stays inside the proof and the address is still H(pk).

Practical today. A one-time PQ-NIZK certifies a fresh PQ key; per-tx proofs run in ~87 ms on an M1.

Open problems

- ECDSA chains (Bitcoin, Ethereum) need the same guarantee, the SHA-256 + secp256k1 circuit is heavier.
- Trusted-setup-free proofs and on-chain verifier cost still need hardening for production.

KU LEUVEN

Thank You!

ssedagha@esat.kuleuven.be

mahdi@soundness.xyz