Post-Quantum Readiness in EdDSA Chains

Financial Cryptography and Data Security (FC 2026)

Foteini Baldimtsi, Kostas Kryptos Chalkias,
Arnab Roy, Mahdi Sedaghat

St. Kitts, March 4

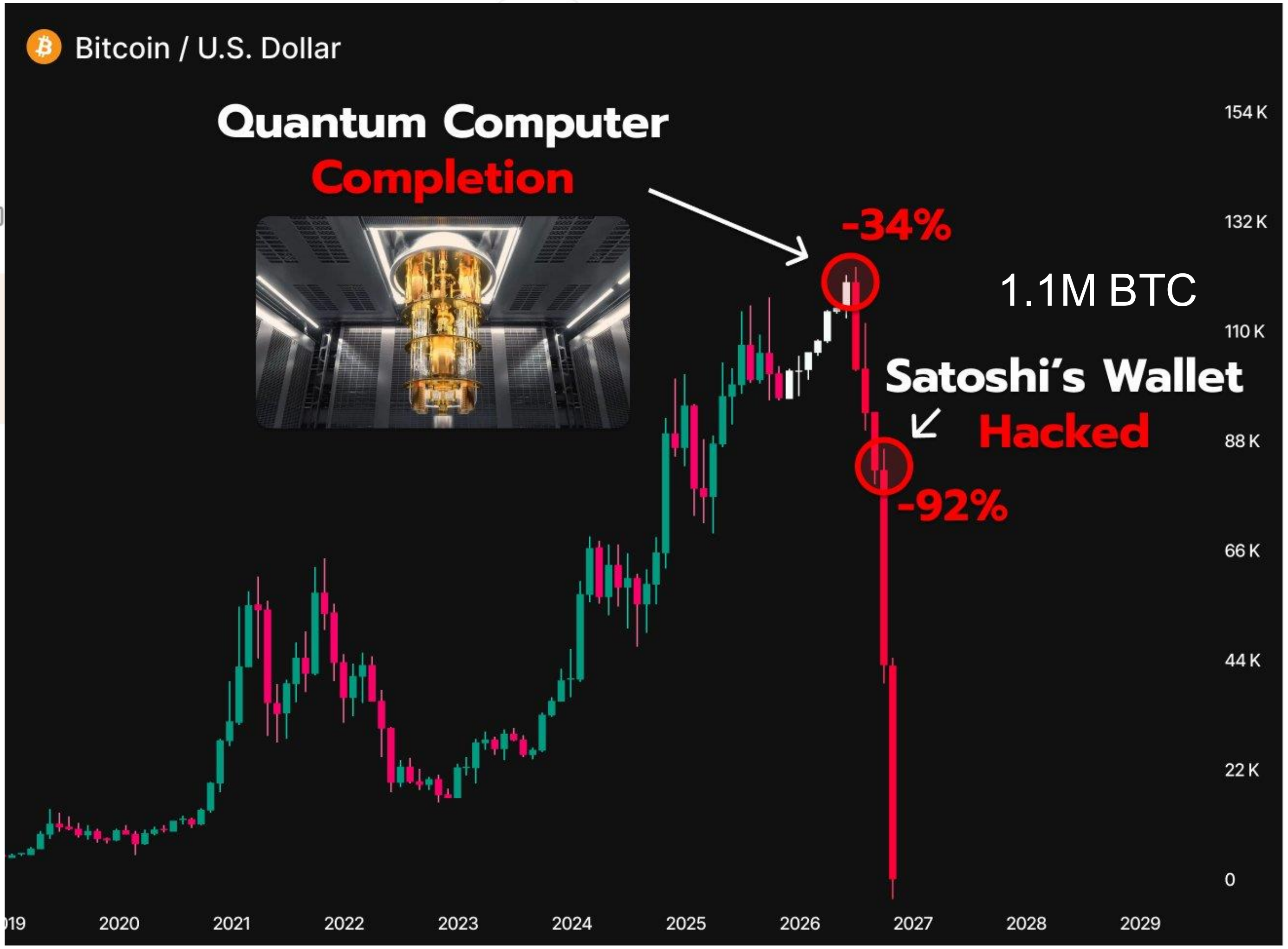
Blockchain: A great bounty for quantum adversary

bc1ql-859v2

Bech32 (P2WPKH)

Bitcoin Address
bc1ql49ydapnjafl5t2cp9zqpjwe6pdgmxy98859v2

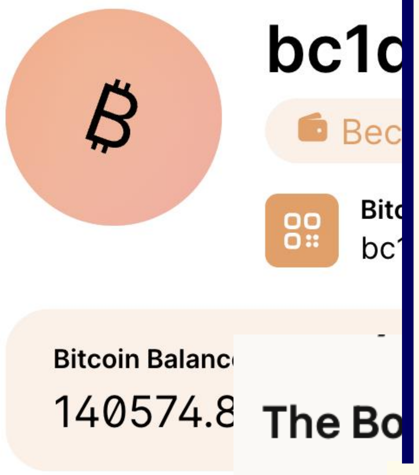
Bitcoin Balance
140574.82562097 • $14,409,417,261

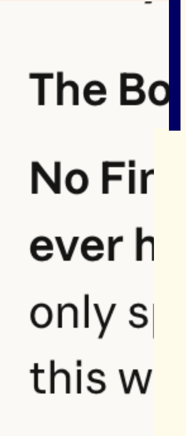


- Many sleeping accounts at risk
- Users won't notice migration calls
- Touching every wallet: **insane cost + risk**

KU LEUVEN

Blockchain: A ... t ... f ...





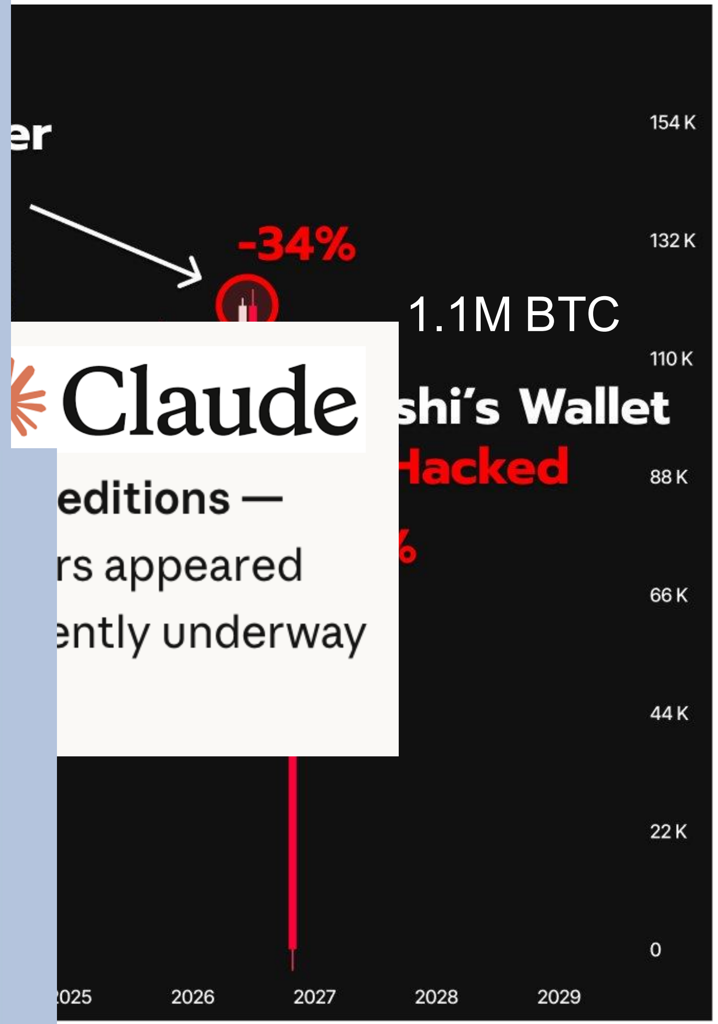
Program

Hallucinated Financial Cryptography and Data Security 2027

31st International Conference
24–28 February 2027
Hilton Barbados Resort
Bridgetown, Barbados

Please note, this program is provisional and subject to change.

12:00–13:30	*Lunch* Location: Ocean Grill
13:30–14:45	**Session 2: Post-Quantum Signatures** *Mosaic: Hash-Based Aggregate Signatures for Post-Quantum Blockchains*. Peter Schwabe, Zhenfei Zhang, Pratyay Mukherjee *Benchmarking Lattice Signature Verification on Resource-Constrained Validators*. Vadim Lyubashevsky, Thomas Prest, Gregor Seiler *Migration Without Disruption: A Framework for Post-Quantum Key Rotation in Live Networks*. Nadia Heninger, Joseph Bonneau, Cas Cremers *On the Cost of Forgetting: Revocation Challenges in Post-Quantum Certificate Infrastructures*. Tibor Jager, Eike Kiltz, Russ Housley



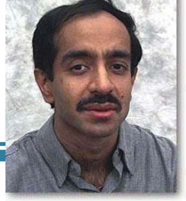
- Many sleeping
- Users won't no
- Touching every

KU LEUVEN

Background

Blockchain fundamentals and quantum attack landscape

Quantum Vulnerability Map

Shor's algorithm  Grover's algorithm 

Which blockchain primitives break under quantum attack?

Broken · Partial · Safe

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

Key insight: The cryptographic backbone (signatures + key derivation) is the primary quantum target, hashing remains resilient

KU LEUVEN

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

KU LEUVEN

The EdDSA Advantage

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

EdDSA PQ-better than ECDSA?

✗ 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

✓ PQ Sign with seed

SHA-512(seed)

Deterministic expansion

sk

Derived from hash output

pk = H2S(SHA-512(seed)[:32]) · G

EC scalar multiplication

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 can serve as a **ZK witness**.

PQ-NIZK Relation:

```
Rel = {(pk, msg, hx) | ∃ seed s.t. pk = H2S(SHA512(seed)[:32])·G
                    ∧ hx = Hash(msg, seed) }
```

Requirements: The proving system must be:

1. PQ secure
2. Enables Client-Side proving
3. It is memory efficient

KU LEUVEN

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

Dual-Mode Signature (DMS) Security

Mode 1: Classical

- Sign1(sk1, msg) = standard EdDSA
- EUF-CMA secure (classical)
- Backward compatible

+

Mode 2: Post-Quantum

Sign2(sk2, msg') = PQ-NIZK proof

EUF-CMA-2 secure (post-quantum)

Secure even if Mode 1 is broken!

Security Properties (Game-Based Proof)

- **DMS Security:** Both modes independently unforgeable
- **Backward Compatible:** Mode 1 identical to legacy EdDSA
- **Shared VK:** Single key (pk) for both modes, no address change

Migration Requirements

Requires Update

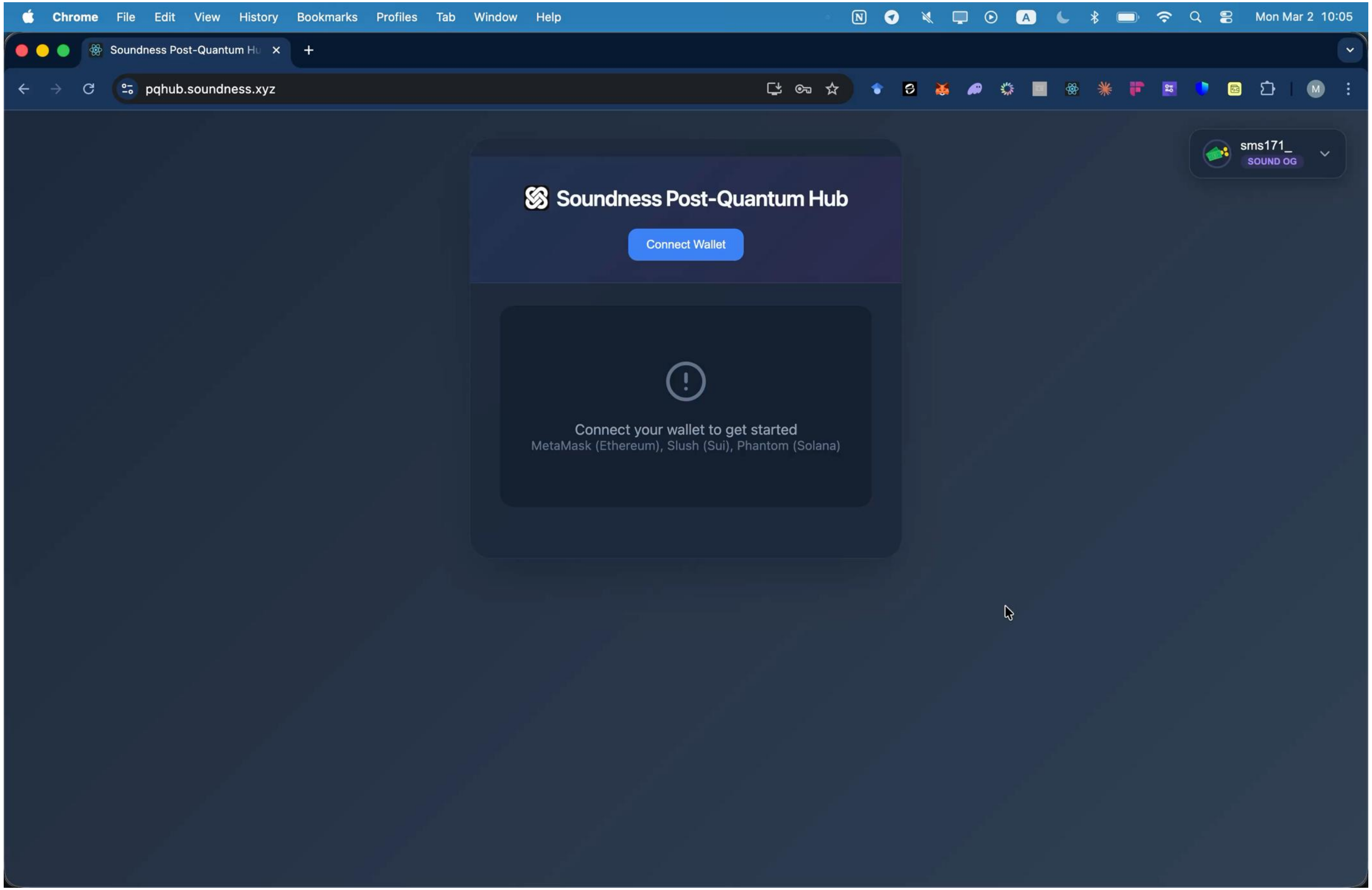
- **Validators/Miners**
  - Accept Mode 2 (PQ-NIZK) signatures
- **Smart Contracts**
  - Add Mode 2 verification logic
- **Wallet Software**

No Change Needed

- Account addresses
- Account structures
- Token/NFT mappings
- On-chain state

Gradual Migration Strategy

Validators accept both Mode 1 & Mode 2 during transition, then enforce Mode 2 only


Chrome File Edit View History Bookmarks Profiles Tab Window Help
Mon Mar 2 10:05
Soundness Post-Quantum Hu
pqhub.soundness.xyz
sms171_
SOUND OG
Soundness Post-Quantum Hub
Connect Wallet
Connect your wallet to get started
MetaMask (Ethereum), Slush (Sui), Phantom (Solana)

KU LEUVEN

Benchmarks

Proof of concept using Ligetron zkVM

Performance Benchmarks: Ed25519 through SLIP10

Emulation (Current)

6.2 sec	2.3 sec	5.4 MB
Proving Time	**Verification Time**	**Proof Size**
331K linear + 4.6M quadratic constraints	MacBook Pro M4, 12 cores, 24 GB	One-time attestation (not per-tx)

↓ 7.75× faster ↓ 7.67× faster ↓ 1.8× smaller

Optimized

0.8 sec	0.3 sec	3 MB
Proving Time	**Verification Time**	**Proof Size**
30K linear + 400K quadratic constraints	MacBook Pro M4, 12 cores, 24 GB	One-time attestation (not per-tx)

onchain attestation?

Ligero → FRI-based (WHIR, STIR)

KU LEUVEN

Conclusion & Future Directions

Key Takeaways

- EdDSA's **deterministic seed-to-key derivation** = structural hook for PQ auth
- **Seamless migration without address changes** for ed-chains such as Sui, Solana, Near
- Formal DMS security model + practical PoC: **6.2s** proving, **2.3s** verification

Future Work

- Reduce proof size via proximity-gap techniques
- Explore GKR-based approaches
- MPC-based Accounts

Broader recommendation: All future sig schemes should define canonical seed-based key derivation

Thank You!

ssedagha@esat.kuleuven.be

mahdi@soundness.xyz