Zero-Knowledge Proof of Seed as a Graceful Emergency Fallback

Stefano Gogioso Hashberg
Fabrizio Genovese 20squares
Nicolò Chiappori Hashberg
Antonio Sanso Ethereum Foundation

ABSTRACT. We propose a practical, deployable mechanism to defend Ethereum from unexpected quantum attacks, leveraging account abstraction and zero-knowledge proofs. Our design does not require any user action prior to a quantum emergency, nor any change to the execution layer, and is expected to protect the majority of wallets and applications.

Draft 04 Feb 2026

1. Introduction

Ethereum's long-term security depends on the hardness of elliptic-curve cryptography. A credible quantum breakthrough would render existing accounts vulnerable, exposing all assets secured by elliptic curve cryptography (ECC). Current mitigation plans focus on replacing the signature scheme at the protocol level, e.g. through new transaction types, precompiles, or consensus-layer upgrades. Such changes will require a hard fork and years of coordination, all the while users' funds remain dependent on ECC. In the event of a quantum emergency, protocol-level post-quantum EIPs juts couldn't deploy fast enough to save the ecosystem.

We propose a different approach, built on existing capabilities of the Ethereum stack. While protocol-level proposals aim to replace ECC at the consensus layer — by upgrading key material — we instead leverage Account Abstraction [1], [2] to establish a user-space migration path:

- Users prove account ownership using quantum-resistant zero-knowledge proofs, implementable today in both hardware and software wallets.
- Proof verification occurs entirely on-chain at Layer 1, with no hard forks, no new transaction types, and no trusted intermediaries.
- No modification is required to existing or future dapps, thanks to account delegation mechanics already available in Ethereum today.

By operating at the account layer, our approach preserves full protocol compatibility, while providing an immediately deployable migration mechanism. Ethereum can achieve practical protection before the protocol itself completes a quantum transition, something no other proposal currently enables (cf. Section 2). This realizes the graceful emergency fallback envisioned in Vitalik Buterin's 2024 proposal [3], but makes it operational today.

The contribution of our proposal is therefore threefold.

1. It provides the first deployable quantum mitigation framework for Ethereum.
2. It serves as a test-bed for post-quantum signature rollouts and recovery workflows.

3. It offers a model for future EIP design, by demonstrating that cryptographic transitions can be achieved top-down — through account abstraction and zero-knowledge verification — without disrupting the protocol itself.

In short, this work transforms account abstraction from a convenience layer into Ethereum's first defence line against the looming quantum threat, bridging today's infrastructure with the cryptographic standards of the post-quantum era.

2. Related Work

Quantum resilience for Ethereum has been discussed extensively at both the protocol and research levels. Existing work can be grouped into three main tracks: (1) protocol-level proposals that modify consensus or transaction formats, (2) cryptographic primitives that extend the EVM to support post-quantum or zero-knowledge verification, and (3) experimental and analytical research targeting direct Layer 1 adoption of post-quantum signatures.

The first group concerns proposals at the protocol and EIP level. The direction for this group is set by Vitalik Buterin's 2024 proposal [3], which outlines an emergency mechanism to recover user funds in the event of a sudden quantum break. The proposal for secondary signature algorithms [4] introduces algorithmic agility for Ethereum transactions — by defining a registry, new transaction container, and decoding precompile — with the intent of allowing Ethereum to accept multiple non-ECDSA schemes (e.g. Dilithium or Falcon) after a protocol upgrade. Complementary drafts propose native verification precompiles for Falcon and ML-DSA (Dilithium) [5], [6], adding post-quantum signature validation as built-in EVM operations.

The second group focuses on cryptographic building blocks at Layer 1. The polynomial number-theoretic transform (NTT) precompile [7], [8] proposes efficient acceleration for both lattice-based signatures and STARK verifiers, addressing some of the computational bottlenecks of post-quantum verification. In parallel, the proposal for a unified on-chain proof verification interface [9] aims to define a standardised, consistent ABI for proofs, which could be used by both post-quantum and zero-knowledge systems within contracts.

The third group of proposals explores research and prototype implementations, mainly targeting Ethereum's direct acceptance of post-quantum signatures. These include analytical series such as the analysis of post-quantum transaction signatures [10], detailing the integration of Falcon and Dilithium at Layer 1, as well as empirical cost studies benchmarking their gas usage and on-chain footprint [11]. Related initiatives examine post-quantum hardening of Ethereum's networking and consensus stack [12] and maintain architectural task lists for full post-quantum migration [13].

With the notable exception of [10], existing proposal don't consider account abstraction (AA) as a key migration mechanism. Current Ethereum quantum-safety designs operate mostly at the protocol layer, and thus require a hard fork to deploy. For example, EIP-7932 (Secondary Signature Algorithms) [4] introduces new transaction structures and modifies consensus verification, but it cannot be implemented at the contract

level. Similarly, EIP-8051/8052 (ML-DSA and Falcon precompiles) [5], [6] provide L1 verification capabilities but still rely on legacy externally owned accounts (EOAs) or protocol-level signature validation. The EIP-7885 (NTT precompile) [7], [8] and ERC-8039 (Unified Proof Verification Interface) [9] proposals supply important performance and standardization primitives, yet neither specifies an account abstraction pattern or deployable user-level migration strategy.

3. Technical Proposal

In Ethereum, externally owned accounts (EOAs) are hard-coded to authenticate through signatures based on elliptic curve cryptography (ECC). Account Abstraction (AA) — defined by ERC-4337 [1] and related proposals — moves authentication logic into contract space instead, allowing any smart contract to act as a fully functional account. This introduces a programmable authentication substrate to Ethereum, which we can leverage to build a quantum-safe migration path without requiring an early hard fork.

In our design, users migrate from traditional EOAs to quantum-safe ERC-4337 smart accounts. Instead of relying on ECC-based signatures, the smart accounts prove ownership by demonstrating knowledge of the seed phrase from which the EOA address was derived, e.g. via BIP-32 hierarchical deterministic derivation [14], [15], [16]. The proof takes the form of a STARK [17], a succinct, transparent and quantum-resistant flavour of zero-knowledge proofs. STARK proofs are generated off-chain and verified by on-chain logic. This derivation approach is inspired by [3] and closely related to a 2023 proposal by [18].

The original EOA is not modified, but rather serves as the source of verifiable ownership which seeds the authority for the quantum-safe smart account. Using mechanics such as ERC-20/ERC-721 approvals [19], [20], permit-based authorisations [21], [22], or ERC-4337 delegated operations [1], users can delegate the quantum-safe account to perform token transfers and dapp operations on behalf of the legacy EOA, establishing continuity of control. Once EIP-7702 [2] is approved, EOAs can run STARK-based authorisation logic on their own, as a mechanism alternative to ECC-based signatures.

The key insight of our proposal is that ERC-4337 smart accounts make it possible for Ethereum to handle a quantum emergency entirely at the consensus level, with no immediate changes needed to the core protocol. We now describe some of the specifics.

Quantum-safe accounts can be deployed by sponsoring parties on behalf of any EOAs which have not yet migrated, because authentication is tied to proof of knowledge of the original seed phrase. This allows a considerable portion of EOAs to be protected in an emergency, not just the ones with the foresight or technical know-how to perform an early migration.[1][2]

[1]It is impossible to conclusively determine the approximate number of EOAs whose addresses arise by hierarchical deterministic derivation, but estimates based on popular wallets — both hardware and software — place a realistic lower bound estimate around the majority line.

A gas vault, coupled with ERC-4337 gas sponsorship mechanics, can be used to reduce the financial burden caused by the additional complexity of on-chain STARK verification. A fraction of the vault can be dedicated to sponsoring early user adoption, with the majority unlocked by custodians in case of an emergency.

Broad integration of the proposal into wallet hardware and software will be necessary for seamless UX. At the enclave/internals level, this consists primarily of the generation of STARK proofs from seed phrase material. At the interface level, this includes deployment and operation of the quantum-safe accounts, one-click delegation for tokens and dapps, and access to gas sponsorship.[3]

The handling of an emergency at consensus level requires validators to stop accepting any transactions from EOAs, other than a standardised set of delegations to the associate quantum-safe accounts, including the transfer of Ether.[4] The deployment and operation of quantum safe accounts proceeds normally, without protocol changes.

From a technical perspective, this requires the introduction of a kill-switch into validator software, as specified by our proposal. The economics of staking make it in the best interest of validators, as a collective, to protect the chain from malicious transactions in the event of a quantum emergency. While the decision to operate the kill-switch is ultimately left into the hands of validators, in accordance with their mandate as ultimate custodians of chain integrity, on-chain oracles integrated at the software level will likely be used to coordinate the response in an emergency.

4. Zero-Knowledge Proof of Seed

The first step in our migration pipeline is the implementation of signature lifting [3], [18], the transformation of a quantum-weak authentication scheme — such as ECDSA, EdDSA, or Schnorr signatures — to a quantum-resistant one via zero-knowledge proof of the pre-image for some cryptographic hashing step — or some other transformation believed to be practically impossible to invert by quantum computers — which was performed as part of the key derivation algorithm.

Signature lifting is a broad technique, likely applicable to many practical key generation pipelines, but in this proposal we focus specifically on BIP-32 hierarchical deterministic wallets [14], arguably the most common key derivation technique in use today.

The BIP-32 key derivation spec presents multiple opportunities for signature lifting:

[2]While we will endeavour to support as many EOAs as possible, there is an ultimate limitation to our approach. Unless the private key was derived by means of some quantum-resistant one-way function [18], there is no purely cryptographic way to definitively determine the ownership of an EOA once advanced quantum capabilities have been demonstrated. In the absence of pre-emptive migration steps, such situations will likely require off-chain resolution.

[3]Note that the quantum-safe account address can be deterministically derived from the corresponding EOA address, making it possible to safely associate the two well in advance of an emergency.

[4]Unlike delegations, the transfer of Ether to a quantum-safe account will be subject to authorisation by the quantum-safe account itself. This is to prevent accidental or malicious transfers for EOAs whose private key was not hierarchically deterministically derived, which would result in permanent locking of the Ether into the quantum-safe account.

- Every private child key is derived from a parent private key via an application of HMAC-SHA512. If the parent public key has not been exposed, knowledge of the parent private key can be used as a witness for signature lifting of non-master keypairs.
- The private master key itself is derived from a master seed via an application of HMAC-SHA512. The master seed is not exposed — unless compromised, see below — and knowledge of its value can be used as a witness for signature lifting of the master keypair.

Lifting opportunities for BIP-32 end at the master seed: in case of compromise, there might not be a further step available for lifting. If the master seed has been derived from a BIP-39 mnemonic, however, signature lifting becomes possible even in case of master seed compromise. Indeed, the BIP-39 spec [15] prescribes 2048 applications of HMAC-SHA512 to derive the master seed from the UTF-8 NFKD bytes of a mnemonic sentence: as wallets typically store the master seed itself, rather than any of the intermediate hashes, the preimage to the last (or last few) HMAC-SHA512 applications can be used as witness for post-quantum lifting of the master seed itself.

Having laid out the plan, the challenge shifts to implementation. We have two desiderata:

- The zero-knowledge proof generation process must be suitable for execution on a variety of platforms and architectures, since it must be possible to integrate it with almost all wallets in current usage.
- The zero-knowledge proof verification process must suitable for execution on-chain, as part of the Account Abstraction framework, and in particular both proofs and verifier must be succinct.

Here, we encounter a problem: zero-knowledge technology has progressed enormously over the last few years, but a fundamental compromise remains to be made between succinctness at the verifier side and workload at the prover side. The vast majority of recent efforts have been focused in the direction of making proofs lightweight and succinct, but often at the expense of heavier prover workloads.

STARK verifiers are already used in production for on-chain verification, and recent EIPs [7], [8] are poised to make the process more efficient by introducing specialised precompiles. This makes STARKs very attractive for the verification side of proof of seed. Unfortunately, the STARK proof generation process is extremely computational intensive: while they might be streamlined enough to be workable on portable devices, there is no realistic prospect of compatibility with secure elements, the higher end of the wallet security ladder.

Secure elements are widely used as hardware wallets, performing critical computations in a tamper-resistant environment. However, these devices often impose a number of stringent limitations on computational resources, the terminal blocker for STARK prover implementation being available RAM in the order of a few dozen KBs.

While memory constraints rule out the vast majority of modern general-purpose, public-verifier ZK protocols, an important exception is represented by MPC-in-the-

Head (MPCitH) protocols [23] such as ZKBoo [24], [25], where succinctness on the verifier side is traded for a significantly lighter workload on the prover side. Subsequent MPCitH developments [26], [27], [28] improve signature size, verification time and prover performance, but they do so at the expense of additional complexity and higher memory footprint, making them less attractive than the original formulation in the context of secure elements.

With a number of considerations, detailed in a separate technical draft [29], the ZKBoo protocol can be tweaked to generate proofs in a fully streamed fashion, with memory requirements independent of the trace length of the prover, the length of the individual proof responses, or the security parameter of the proof. This brings memory requirements for proof generation down to within a small factor — slightly above 3x — of the minimal memory requirements for the computation being proven, well within the capabilities of secure elements such as those used by Ledger wallets.

A critical limitation of ZKBoo — and one of the reasons why it doesn't hold much mindshare in the modern zero-knowledge landscape — is that neither its proofs nor its verifier are succinct, in that both the size of the former and the runtime of the latter are linear in the trace length. This makes ZKBoo proofs unsuitable for direct use in the on-chain verification process, but that was not the reason why this technique was chosen: its job is to allow the practical extraction of a proof of knowledge from a computationally constrained secure environment, wrapped into a zero-knowledge container.

Once the ZKBoo proof has been produced, more powerful machines can be used to perform a second step of recursive verification, resulting in succinct lifting: where the ZKBoo proof might prove the statement "I know the seed from which this public key was derived", the lifted proof would succinctly prove the statement "I known a valid ZKboo proof for the statement that I know the seed from which this public key was derived.". (It goes without saying that additional information should be included in the ZKBoo proof to bind it to a specific authentication instance, such as a specific transaction to be authorised. But these are entirely standard considerations.)

Importantly, the external machine performing the succinct lifting does not need to be trusted, because the ZKBoo proof itself already guarantees zero-knowledge. For the same reason, the technique used for succinct lifting does not itself need to be zero-knowledge. This latter observation opens the door to applications of succinct ZK systems — such as ones for binary fields [30], [31] — which be well-suited to proving statements heavy in bitwise operations — such as those involved in the ZKBoo proof verification — but might not themselves be formulated in a zero-knowledge way.

Bibliography

[1] V. Buterin *et al.*, 'ERC-4337: Account Abstraction Using Alt Mempool'. [Online]. Available: https://eips.ethereum.org/EIPS/eip-4337

[2] V. Buterin, S. Wilson, A. Dietrichs, and lightclient, 'EIP-7702: Set Code for EOAs'. [Online]. Available: https://eips.ethereum.org/EIPS/eip-7702

[3] V. Buterin, 'How to hard fork to save most users' funds in a quantum emergency'. [Online]. Available: https://ethresear.ch/t/how-to-hard-fork-to-save-most-users-funds-in-a-quantum-emergency/18901

[4] J. Kempton, 'EIP-7932: Secondary Signature Algorithms'. [Online]. Available: https://eips.ethereum.org/EIPS/eip-7932

[5] S. Masson, 'EIP-8051: ML-DSA Verification'. [Online]. Available: https://ethereum-magicians.org/t/eip-8051-ml-dsa-verification/25857

[6] R. Dubois, 'EIP-8052: Precompile for Falcon support'. [Online]. Available: https://ethereum-magicians.org/t/eip-8052-precompile-for-falcon-support/25860

[7] R. Dubois, 'EIP-7885: Precompile for NTT operations'. [Online]. Available: https://ethereum-magicians.org/t/eip-7885-precompile-for-ntt-operations/22895

[8] R. Dubois, 'NTT as PostQuantum and Starks settlements helper precompile'. [Online]. Available: https://ethresear.ch/t/ntt-as-postquantum-and-starks-settlements-helper-precompile/21775/1

[9] W. Khemiri, 'ERC-8039: Unified interface for on-chain proof verification'. [Online]. Available: https://ethereum-magicians.org/t/erc-8039-unified-interface-for-on-chain-proof-verification/25612

[10] A. Sanso, 'The road to Post-Quantum Ethereum transaction is paved with Account Abstraction (AA)'. [Online]. Available: https://ethresear.ch/t/the-road-to-post-quantum-ethereum-transaction-is-paved-with-account-abstraction-aa/21783

[11] P. Juaristi, I. Agudo, R. Rios, and L. Ricci, 'Benchmarking Post-quantum Cryptography in Ethereum-Based Blockchains', in *Computer Security. ESORICS 2024 International Workshops*, Springer Nature, 2025, pp. 340–353. [Online]. Available: https://www.nics.uma.es/wp-content/papers/agudo2024cbt.pdf

[12] A. Bienvenido and G. Du, 'Towards a Quantum-Safe P2P for Ethereum'. [Online]. Available: https://pse.dev/blog/towards_a_quantum-safe_p2p_for_ethereum

[13] P. Miller, 'Tasklist for post-quantum ETH'. [Online]. Available: https://ethresear.ch/t/tasklist-for-post-quantum-eth/21296

[14] P. Wuille, 'BIP-32: Hierarchical Deterministic Wallets'. [Online]. Available: https://bips.dev/32/

[15] M. Palatinus, P. Rusnak, A. Voisine, and S. Bowe, 'BIP-39: Mnemonic code for generating deterministic keys'. [Online]. Available: https://bips.dev/39/

[16] M. Palatinus and P. Rusnak, 'BIP-44: Multi-Account Hierarchy for Deterministic Wallets'. [Online]. Available: https://bips.dev/44/

[17] E. Ben-Sasson, I. Bentov, Y. Horesh, and M. Riabzev, 'Scalable, Transparent, and Post-Quantum Secure Computational Integrity', technical report 2018/046, 2018. [Online]. Available: https://eprint.iacr.org/2018/046

[18] O. Sattath and S. Wyborski, 'Protecting Quantum Procrastinators with Signature Lifting: A Case Study in Cryptocurrencies', technical report 2023/362, 2023. [Online]. Available: https://eprint.iacr.org/2023/362

[19] F. Vogelsteller and V. Buterin, 'ERC-20: Token Standard'. [Online]. Available: https://eips.ethereum.org/EIPS/eip-20

[20] W. Entriken, D. Shirley, J. Evans, and N. Sachs, 'ERC-721: Non-Fungible Token Standard'. [Online]. Available: https://eips.ethereum.org/EIPS/eip-721

[21] M. Lundfall, 'ERC-2612: Permit Extension for EIP-20 Signed Approvals'. [Online]. Available: https://eips.ethereum.org/EIPS/eip-2612

[22] S. Fremaux and W. Schwab, 'ERC-4494: Permit for ERC-721 NFTs'. [Online]. Available: https://eips.ethereum.org/EIPS/eip-4494

[23] Y. Ishai, E. Kushilevitz, R. Ostrovsky, and A. Sahai, 'Zero-knowledge from secure multiparty computation', in *Proceedings of the thirty-ninth annual ACM symposium on Theory of computing*, 2007, pp. 21–30. [Online]. Available: https://dl.acm.org/doi/abs/10.1145/1250790.1250794

[24] I. Giacomelli, J. Madsen, and C. Orlandi, 'ZKBoo: Faster Zero-Knowledge for Boolean Circuits', 2016, [Online]. Available: https://eprint.iacr.org/2016/163

[25] M. Chase *et al.*, 'Post-Quantum Zero-Knowledge and Signatures from Symmetric-Key Primitives', 2017, doi: 10.1145/3133956.3133997.

[26] J. Katz, V. Kolesnikov, and X. Wang, 'Improved Non-Interactive Zero Knowledge with Applications to Post-Quantum Signatures'. [Online]. Available: https://eprint.iacr.org/2018/475

[27] C. Baum and A. Nof, 'Concretely-Efficient Zero-Knowledge Arguments for Arithmetic Circuits and Their Application to Lattice-Based Cryptography'. [Online]. Available: https://eprint.iacr.org/2019/532

[28] Y. Gvili, J. Ha, S. Scheffler, M. Varia, Z. Yang, and X. Zhang, 'TurboIKOS: Improved Non-interactive Zero Knowledge and Post-Quantum Signatures'. [Online]. Available: https://eprint.iacr.org/2021/478

[29] S. Gogioso and N. Chiappori, 'Boo in a Box: ZK Proofs for Secure Elements', technical report, 2026. [Online]. Available: https://github.com/ProofofSeed/boo-in-a-box/

[30] B. E. Diamond and J. Posen, 'Succinct Arguments over Towers of Binary Fields'. [Online]. Available: https://eprint.iacr.org/2023/1784

[31] B. E. Diamond and J. Posen, 'Polylogarithmic Proofs for Multilinears over Binary Towers'. [Online]. Available: https://eprint.iacr.org/2024/504

[32] A. Sanso, 'So you wanna Post-Quantum Ethereum transaction signature'. [Online]. Available: https://ethresear.ch/t/so-you-wanna-post-quantum-ethereum-transaction-signature/21291

[33] W. Radomski, A. Cooke, P. Castonguay, J. Therien, E. Binet, and R. Sandford, 'ERC-1155: Multi Token Standard'. [Online]. Available: https://eips.ethereum.org/EIPS/eip-1155

[34] F. Giordano, M. Condon, P. Castonguay, A. Bandeali, J. Izquierdo, and B. Masius, 'ERC-1271: Standard Signature Validation Method for Contracts'. [Online]. Available: https://eips.ethereum.org/EIPS/eip-1271

[35] T. Daubenschütz and Anders, 'ERC-5192: Minimal Soulbound NFTs'. [Online]. Available: https://eips.ethereum.org/EIPS/eip-5192

[36] STMIcroelectronics, 'ST33K: High-speed secure MCU with 32-bit Arm Cortex-M35P CPU'. [Online]. Available: https://www.st.com/en/secure-mcus/st33k1m5c.html