PQSigAbstract: A Modular Post-Quantum Signature Verification Architecture for Ethereum

Ankita Virani
University of Colorado Boulder
1 May 2026

June 2, 2026

Contents

Abstract

Ethereum's transaction validity model is currently anchored in ECDSA over secp256k1, whose security assumptions are known to weaken in the presence of large-scale quantum adversaries, particularly under Shor's algorithm [1]. Recent progress in quantum hardware further strengthens the plausibility of this threat model [2]. Although NIST-standardised post-quantum (PQ) signature schemes such as ML-DSA, SLH-DSA, and FALCON provide resistance against quantum attacks [4, 5, 7, 8], integrating these schemes into Ethereum raises challenges that extend beyond straightforward cryptographic replacement.

A closer examination of Ethereum's validation pipeline reveals three structural tensions. First, bounded execution assumptions are strained, as worst-case verification cost G^{PQ} may exceed practical mempool admission limits $G_{\max}$. Second, verification cost is no longer strictly deterministic, since execution depends on input structure, leading to discrepancies between off-chain simulation and on-chain behaviour. Third, adversarial inputs can be crafted to trigger worst-case execution paths, enabling denial-of-service conditions without requiring valid signatures.

These observations suggest that directly embedding PQ verification within existing validation constraints is not viable under a fixed gas bound. To address this, we introduce PQSigAbstract, a modular verification architecture that separates validation into a stateless pre-validation phase V_1 and a deferred cryptographic verification phase V_2. The two phases are linked through a binding commitment $C = H(\sigma \| m \| pk \| \text{nonce})$, ensuring that inputs validated in V_1 cannot be substituted during V_2.

The proposed architecture defines a typed Verification Module (VM) interface with explicit gas estimation, alongside a versioned Scheme Registry that incorporates a quarantine mechanism for safer deployment. To mitigate verification overhead, we further introduce a probabilistic aggregation approach achieving $O(\sqrt{n})$ amortized cost for schemes that do not support native aggregation. In addition, a commitment-based migration mechanism for EIP-7702 [10] is outlined to reduce exposure during account transitions.

Under this design, EU-CMA security is preserved because the full verification procedure remains unchanged in the second phase, while the pre-validation stage introduces no independent acceptance condition. Analytical gas models are derived for ML-DSA-44, FALCON-512, and SLH-DSA-128f, and empirical evaluation indicates that amortized costs remain within approximately 72×–190× of ECDSA. While higher than classical verification, these costs remain practical for high-value accounts and are compatible with ERC-4337 [9] and RIP-7560 [11].

1 Introduction

1.1 Motivation and Quantum Timeline

Ethereum currently relies on ECDSA over the Koblitz curve secp256k1 for transaction authentication. The security of this construction is based on the hardness of the elliptic curve discrete logarithm problem (ECDLP), which requires approximately $\Theta(2^{128})$ classical operations at a 256-bit security level. Shor's quantum algorithm reduces this complexity to $O((\log p)^2 \log\log p \cdot \log\log\log p)$ gate operations [1], effectively eliminating this security margin in the presence of sufficiently capable quantum hardware.

Recent hardware developments indicate that this risk is no longer purely hypothetical. IBM's Condor processor (2023) demonstrated 1,121 physical qubits, and progress towards fault-tolerant logical qubits continues on a credible trajectory [2]. While practical attacks on secp256k1 are still estimated to require 2,330–3,000 logical qubits [2], adversaries can already adopt a *harvest now, decrypt later* strategy by collecting signed transactions today and exploiting them once quantum capabilities mature. This creates a temporal asymmetry, since data can be collected immediately while the capability to exploit it emerges later: the cost of data collection is immediate, while the ability to break it is deferred but increasingly plausible.

NIST's post-quantum standardisation effort, finalised in 2024, produced FIPS 204 (ML-DSA), FIPS 205 (SLH-DSA), and FIPS 206 (ML-KEM) [3]. These schemes are widely regarded as production-ready quantum-resistant primitives. However, their integration into Ethereum is not a straightforward substitution problem, Recent ecosystem discussions have similarly recognised that migration toward quantum-resistant Ethereum accounts will likely require staged deployment strategies rather than immediate protocol-wide replacement [20]. NTT-based modular arithmetic relies on efficient modular reduction techniques, commonly implemented using Montgomery multiplication [13].

1.2 The Systems-Level Incompatibility

Existing work on post-quantum blockchain integration has primarily focused on cryptographic substitution, signature design, or specialised signing models [16, 17]. More recent Ethereum-native approaches have explored account abstraction as a deployment mechanism for custom verification logic [9].However, these approaches leave a gap at the system level, where protocol constraints interact directly with verification cost.

At a closer inspection, this incompatibility emerges from cryptographic inefficiency alone, but from the interaction between PQ scheme characteristics and Ethereum's validation model. This interaction introduces several implicit constraints on signature verification:

1. **Bounded Execution.** Verification cost must satisfy $G(u) \leq G_{\max}$ for all admissible inputs u, preventing unbounded computation at the mempool layer. ECDSA verification is realised via the `ecrecover` precompile at a fixed cost of 3,000 gas. In contrast, ML-DSA-44 verification requires approximately 1.84M gas in pure Solidity over 600× the ECDSA precompile cost and approaching the gas limit of an entire block.

2. **Gas Determinism.** Simulated execution costs must closely match on-chain execution costs. EVM JIT compilation (used by evmone's LLVM backend and by Geth's EVM) differs from interpreter-mode gas consumption by 8–15% for compute-intensive inner loops. For ECDSA (a precompile), this divergence is zero. For NTT-based polynomial arithmetic in Dilithium, which executes 1,024 butterfly operations in a tight inner loop, the JIT-versus-interpreter divergence is significant enough to affect bundler inclusion decisions.

3. **Adversarial Input Resistance.** Verification cost should remain independent of attacker-controlled input structure. For hash-based schemes such as SLH-DSA, the Merkle tree traversal path is determined by the signature itself. An adversary cannot forge a *valid* signature with a malicious path structure that would break EU-CMA security but can submit structurally well-formed yet invalid inputs whose verification consumes maximum computation before rejection.

1.3 Contributions

This work makes several contributions, summarised below.

1. **Formal Incompatibility Proof** (Section 3). It is shown that full PQ verification cannot be safely embedded in V_{mempool} under any fixed bound $G_{\max}$, establishing a structural impossibility result independent of specific scheme parameters.

2. **Verification Module Abstraction** (Section 4). A formally typed interface $\mathcal{VM} = (\mathcal{V}, \mathcal{P}, \mathcal{G}, \mathcal{S}, \mathcal{I})$ binds a PQ scheme to a provably tight gas estimate, with security invariants enforced at registration.

3. **Two-Phase Validation Protocol** (Section 6). Validation is decomposed into a gas-bounded pre-validation stage V_1 and a deferred full verification stage V_2, linked by a cryptographic commitment C. Soundness and completeness are established, and EU-CMA security is preserved under the two-phase construction, since full verification remains unchanged in the second stage.

4. **Lazy Probabilistic Aggregation** (Section 7). A batch verification protocol for non-aggregatable PQ schemes achieves $O(\sqrt{n})$ amortised on-chain cost, with formal detection probability and economic security analysis.

5. **Gas Cost Models and Benchmarks** (Section 8). Closed-form gas models are derived for ML-DSA-44, FALCON-512, and SLH-DSA-128f, validated against empirical measurements on a mainnet fork.

6. **Migration Protocol for EIP-7702** (Section 10). A commitment-protected migration path defends against quantum-accelerated front-running during the ECDSA-to-PQ transition.

2 Background

2.1 Ethereum Transaction Validity

For a transaction **tx**, let $h = H_{\text{keccak}}(\mathbf{tx})$ denote its RLP-encoded hash, with signature tuple (r, s, v). Ethereum protocol validity requires:

$$\text{sender}(\mathbf{tx}) = \texttt{ecrecover}(h, r, s, v) \neq \perp. \tag{1}$$

Consensus clients perform this verification prior to EVM execution, meaning that signature validity is enforced at the protocol level rather than within contract logic. As a result, this mechanism is not subject to gas accounting and cannot be extended without introducing a new transaction type or modifying consensus rules.

2.2 Account Abstraction Paradigms

ERC-4337 [9] defines *UserOperations* (UOs) submitted to an off-chain alternative mempool and bundled into standard transactions by permissionless *bundlers*. Execution is coordinated through the `EntryPoint` contract, which invokes `validateUserOp` on the sender account. This design allows each account to define its own signature verification logic, effectively decoupling authentication from the base protocol.

EIP-7702 [10] enables an externally owned account (EOA) to temporarily delegate execution to a contract through a signed authorisation tuple, allowing EOAs to behave similarly to smart accounts without permanent migration. The initial delegation requires an ECDSA signature, creating a brief exposure window that quantum-capable adversaries may exploit. This introduces a transitional security risk, particularly in scenarios where accounts migrate from ECDSA-based authentication to post-quantum schemes.

RIP-7560 [11] introduces protocol-level support for account abstraction via a new transaction type (`AA_TX_TYPE = 4`), shifting validation into the consensus layer and enabling more efficient integration of PQ verification.

2.3 NIST PQC Signature Schemes

Table 1 lists the key operational parameters of the three NIST-standardised signature schemes relevant to Ethereum. ML-DSA and FALCON are based on lattice problems, specifically MLWE and NTRU, which are currently considered resistant to both classical and quantum attacks. In contrast, SLH-DSA is constructed entirely from hash-based primitives, relying only on the security of its underlying hash functions. This distinction leads to a trade-off: while hash-based schemes offer conservative security assumptions, they incur significantly larger signature sizes. [6–8].

Table 1: NIST PQC Signature Scheme Operational Parameters (128-bit Security Level)

Scheme	PK (B)	Sig (B)	Basis	Hard Problem
ML-DSA-44	1,312	2,420	Lattice	MLWE / SIS
FALCON-512	897	≈666	Lattice	NTRU
SLH-DSA-128f	32	16,976	Hash	OWF
secp256k1	33	64	ECC	ECDLP

3 Problem Formalisation

3.1 Execution Layer Model

Ethereum's transaction lifecycle can be viewed as an ordered sequence of validation contexts:

$$\mathcal{L} = \langle V_{\text{mempool}},\ V_{\text{execution}},\ V_{\text{post}} \rangle, \tag{2}$$

where V_{mempool} performs pre-inclusion checks under a fixed resource bound, $V_{\text{execution}}$ performs on-chain EVM execution, and V_{post} encompasses deferred verification mechanisms such as fraud proofs and ZK proofs.

Definition 1 (Bounded Mempool Context)**.** *A validation function $V : \mathcal{U} \to \{0, 1\}$ is* mempool-safe *if*

$$\sup_{u \in \mathcal{U}} G_V(u) \leq G_{\max}, \tag{3}$$

where $G_V(u)$ is the gas consumed by V on input u and $G_{\max}$ is the protocol- defined admission bound. Furthermore, V must be input-deterministic, meaning that for all u, the cost $G_V(u)$ can be determined without executing V itself.

3.2 Gas Cost Decomposition for PQ Verification

The total gas cost of PQ signature verification decomposes as:

$$G^{PQ}(u) = G_{\text{poly}}(u) + G_{\text{hash}}(u) + G_{\text{mem}}(u) + G_{\text{calldata}}(u), \tag{4}$$

where each term corresponds to a distinct computational or data-related cost component.

Polynomial arithmetic cost. For lattice-based schemes executing the Number Theoretic Transform (NTT) over $R_q = \mathbb{Z}_q[X]/(X^n + 1)$:

$$G_{\text{poly}}(u) = \alpha \cdot k\ell \cdot n \log_2 n, \tag{5}$$

where k, ℓ are the module matrix dimensions, n is the ring degree, and α is the per-butterfly gas constant empirically estimated at 24 gas per textttmulmod+`addmod` operation pair.

Hash evaluation cost. For hash-tree-based schemes:

$$G_{\text{hash}}(u) = \beta \cdot d \cdot h_{\text{tree}}, \tag{6}$$

where d is the number of tree layers, h_{tree} is the tree height, and $\beta = 72$ gas per SHA-256 precompile call on a 32-byte input.

Memory and calldata costs. $G_{\text{mem}}(u)$ captures memory expansion and SSTORE/SLOAD costs for parameter tables. $G_{\text{calldata}}(u)$ captures the calldata cost associated with transmitting the signature and public key at 16 gas per non-zero byte (EIP-2028).

In contrast, ECDSA verification via the `ecrecover` precompile has a *protocol-fixed* cost:

$$G^{\text{ECDSA}} = G_{\text{precompile}} = 3{,}000 \text{ gas (constant)}. \tag{7}$$

This is a gas abstraction baked into the consensus rules, decoupling implementation cost from protocol cost. At present, no comparable abstraction exists for PQ schemes.

3.3 Structural Incompatibility Theorem

Theorem 1 (PQ Mempool Incompatibility)**.** *Let Σ^{PQ} be any NIST Level 1 post-quantum signature scheme whose verification algorithm $\mathcal{V}^{PQ}$ executes polynomial arithmetic of degree $\Omega(\log n)$ in the ring dimension n. Then for any fixed bound $G_{\max} < \infty$:*

$$\sup_{u \in \mathcal{U}} G_{\mathcal{V}}^{PQ}(u) > G_{\max}. \tag{8}$$

That is, $\mathcal{V}^{PQ}$ is not mempool-safe.

Proof. Step 1: Adversarial input construction. Consider an adversary who submits a structurally valid but semantically invalid signature $\sigma^* \in \mathcal{SIG}$ of maximum byte length $|\sigma^*| = |\sigma|_{\max}$. The EVM must execute $\mathcal{V}^{PQ}$ to completion before determining invalidity, since the rejection condition only becomes apparent after full evaluation. since the rejection condition is embedded in the NTT-domain product comparison and no efficient short-circuit path exists in the general case.

Step 2: Cost lower bound. For ML-DSA-44, the NTT matrix-vector product $\mathbf{Az} \in R_q^{k \times 1}$ requires exactly $k\ell$ polynomial multiplications over R_q with $(k, \ell) = (4, 4)$, each comprising one forward NTT, one pointwise multiplication, and one inverse NTT. This leads to the following lower bound on EVM execution cost:

$$G_{\text{poly}} \geq k\ell \cdot 2n \cdot \log_2(2n) \cdot \alpha_{\text{mul}}. \tag{9}$$

For $k = \ell = 4$, $n = 256$, $\alpha_{\text{mul}} = 8$ gas: $G_{\text{poly}} \geq 16 \times 512 \times 9 \times 8 = 589{,}824$ gas for NTT alone. Adding calldata costs ($2{,}420 \times 16 = 38{,}720$ gas) and auxiliary hash operations already places the total within the range of typical ERC-4337 bundler gas caps. Critically, $G_{\max}$ is a fixed constant while G_{poly} scales with scheme parameters: at NIST Level 3 ($k = \ell = 6$) and Level 5

$(k = \ell = 8)$, the gap widens without bound. Therefore, no fixed $G_{\max}$ can bound G^{PQ} across all security levels, which establishes the claimed incompatibility. □

Remark 1. *Theorem 1 highlights a structural limitation rather than a simple gas-pricing issue. Raising $G_{\max}$ to accommodate current ML-DSA-44 costs recreates the violation at the next security level. The correct resolution is to restructure* where *full verification occurs, not to adjust $G_{\max}$.*

3.4 Gas Non-Determinism

A second issue arises from divergence between execution environments. ERC-4337 bundlers simulate `validateUserOp` off-chain before including UOs in a bundle.

Definition 2 (Gas Non-Determinism)**.** *A verification function V exhibits* gas non-determinism *if two execution contexts E_1 (simulation) and E_2 (on-chain) satisfy:*

$$\exists\, u \in \mathcal{U} : \quad G_V^{E_1}(u) \neq G_V^{E_2}(u). \tag{10}$$

For NTT-intensive verification, this occurs due to EVM JIT warm-up: the first invocation of a tight loop incurs JIT compilation overhead, while subsequent invocations use compiled code. The divergence is bounded empirically at 8–15% for 1,024-iteration NTT loops. This creates a practical risk in which a bundler may accept a UO based on simulated gas $\hat{G}(u) \leq G_{\text{limit}}$, while on-chain execution exhausts gas at $G(u) > G_{\text{limit}}$, causing a failed transaction that still consumes the full gas limit and damages bundler economics.

3.5 Limitations of Existing Verification Models

Ethereum's current transaction validation model assumes that signature verification is computationally lightweight, deterministic, and economically bounded.

This assumption holds for secp256k1-based ECDSA verification through the native precompile, where verification cost remains effectively constant across inputs.

However, post-quantum signature schemes violate these assumptions.

Lattice-based schemes such as ML-DSA require polynomial arithmetic over structured rings using Number Theoretic Transform (NTT), while stateless hash-based schemes such as SLH-DSA require repeated authentication path verification over Merkle structures [7].

This creates three structural limitations in existing verification pipelines:

1. **Fixed-cost assumptions fail.** Existing verification models assume near-constant verification cost, while PQ verification exhibits input-dependent execution complexity.

2. **Mempool simulation becomes unsafe.** ERC-4337 bundlers must simulate transactions before inclusion, but worst-case PQ inputs may produce gas divergence between simulation and execution [9].

3. **Single-phase verification introduces denial-of-service risk.** If full PQ verification is required before transaction admission, adversaries can amplify validator workload at negligible signing cost.

These limitations motivate the need for a verification architecture that decouples transaction admission from full cryptographic verification.

4 Verification Module Abstraction

4.1 Formal Definition

Definition 3 (Verification Module)**.** *A* Verification Module $\mathcal{VM}_\Sigma$ *associated with a signature scheme* Σ *is defined as a tuple* $(\mathcal{V}, \mathcal{P}, \mathcal{G}, \mathcal{S}, \mathcal{I})$ *where:*

- $\mathcal{V} : \{0,1\}^* \times \{0,1\}^{256} \times \{0,1\}^* \to \{0,1\}$ *is a deterministic* `view`*-function predicate which maps* $(pk,\ H_{msg},\ \sigma)$ *to a Boolean output in* $\{0,1\}$.
- $\mathcal{P} = (p_1, \ldots, p_k)$ *is a finite, immutable parameter vector; each* p_i *is a fixed-length byte array registered at deployment time.*
- $\mathcal{G} : \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ *is a gas estimator mapping* $(|pk|, |\sigma|) \mapsto \hat{G}$*, such that* $\hat{G} \geq G_{\mathcal{V}}(pk, \cdot, \sigma)$ *holds for all valid inputs.* $\mathcal{G}$ *and must be computable in* $O(1)$ *time without executing* $\mathcal{V}$.
- $\mathcal{S} \in \{1,2,3,4,5\}$ *is the NIST security level.*
- $\mathcal{I} \in \{0,1\}$ *is the algebraic aggregation flag:* $\mathcal{I} = 1$ *iff* Σ *supports bilinear-pairing-based aggregation.*

4.2 Required Invariants

The following invariants define the conditions under which a VM can be safely registered in the Scheme Registry (Section 5).

Definition 4 (Gas Soundness)**.** $\mathcal{VM}_\Sigma$ *is* gas-sound *if*

$$\forall\, pk, \sigma : \quad G_{\mathcal{V}}(pk, \cdot, \sigma) \leq \mathcal{G}(|pk|, |\sigma|). \tag{11}$$

Definition 5 (Gas Tightness)**.** $\mathcal{VM}_\Sigma$ *is* gas-tight *with constant* $\kappa \geq 1$ *if*

$$\sup_{pk,\sigma} \frac{\mathcal{G}(|pk|, |\sigma|)}{G_{\mathcal{V}}(pk, \cdot, \sigma)} \leq \kappa. \tag{12}$$

We require $\kappa \leq 1.05$ *(5% overestimate tolerance).*

Gas tightness limits overestimation, ensuring that bundlers do not reserve excessive gas for verification, which would otherwise reduce effective block utilisation. For ML-DSA-44 an overestimate of $\kappa \approx 1.02$ is achieved via the closed-form cost model in Section 8.

Definition 6 (Scheme Isolation)**.** $\mathcal{VM}_\Sigma$ *satisfies* scheme isolation *if* $\mathcal{V}$ *makes no external calls (*`CALL`*,* `DELEGATECALL`*,* `STATICCALL`*) to addresses outside the set of approved precompiles* $\mathcal{P}_{pre} = \{0x01, \ldots, 0x08\}$.

Scheme isolation under standard assumptions that a malicious VM cannot invoke attacker-controlled contracts during the verification of unrelated user operations.

4.3 Solidity Interface

Listing 1 presents the proposed `IVerificationModule` interface. The `estimateGas` function is a pure, $O(1)$ computation that bundlers call before simulating a UO, enabling bundlers to reject invalid or infeasible operations early without performing full cryptographic verification.

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.24;

interface IVerificationModule {
    /// Unique scheme id:
    ///   keccak256(abi.encode(name, version, paramHash))
    function schemeId()
        external view returns (bytes32);

    /// NIST security level (1-5)
    function securityLevel()
        external view returns (uint8);

    /// True iff algebraic aggregation is supported
    function supportsAggregation()
        external view returns (bool);

    /// Gas upper bound -- MUST NOT revert, O(1) compute.
    /// Must satisfy gas-soundness (Def. 2) and
    /// gas-tightness (Def. 3) with kappa <= 1.05.
    function estimateGas(uint16 pkLen, uint16 sigLen)
        external pure returns (uint256);

    /// Core verification predicate.
    /// MUST be a view function (no state writes).
    /// MUST NOT call external contracts outside precompiles.
    function verify(
        bytes calldata publicKey,
        bytes32       msgHash,
        bytes calldata signature
    ) external view returns (bool);
}
```

Listing 1: Verification Module interface (Solidity 0.8.x).

5 Scheme Registry

5.1 Registry State Machine

The Scheme Registry maintains a finite-state machine for each registered scheme:

$$\mathcal{R} : \text{SchemeId} \to \big(\text{addr}_{VM},\ \text{State},\ t_{\text{activate}},\ t_{\text{deprecate}}\big), \tag{13}$$

Each entry maps a scheme identifier to its corresponding verification module and lifecycle metadata. with states {Proposed, Quarantine, Active, Deprecated} and transitions:

$$\textsc{Proposed} \xrightarrow{\text{governance vote}} \textsc{Quarantine}, \tag{14}$$

$$\textsc{Quarantine} \xrightarrow{t \geq t_{\text{activate}}} \textsc{Active}, \tag{15}$$

$$\textsc{Active} \xrightarrow{\text{governance vote}} \textsc{Deprecated}. \tag{16}$$

A quarantine period $\Delta_q \geq 30$ days is enforced on-chain. During quarantine, accounts may bind to the scheme but compliant bundlers reject UOs that reference it. This provides an ecosystem-wide observation window in which potential vulnerabilities or malicious behaviours can be identified before the scheme becomes active.

5.2 Parameter Hash Binding

Different implementations of "ML-DSA-44" may vary in their NTT basis, reduction strategy, or byte encoding while both being specification-conformant, and may therefore produce signatures that are not mutually compatible. To eliminate this ambiguity, the VM bytecode hash is incorporated into the scheme identifier:

$$\text{schemeId} = H\big(\text{schemeName} \| \text{version} \| H(\text{VM.bytecode})\big). \tag{17}$$

As a result, even a minor VM modification (e.g., a bug fix) produces a distinct `schemeId`, requiring explicit re-registration and re-binding by affected accounts, and This prevents silent parameter drift, where changes in implementation could otherwise alter verification semantics without being explicitly acknowledged.

5.3 Account-Local Binding

Scheme binding is maintained within the account's own storage rather than in the registry:

$$\text{binding}(\text{addr}) = \big(\text{schemeId},\ pk,\ \text{schemeId}_{\text{fallback}}\big). \tag{18}$$

The optional fallback enables graceful degradation by allowing accounts to revert to a secondary scheme if the primary scheme becomes unavailable. Critically, a registry compromise cannot retroactively alter existing bindings: the registry governs scheme availability, but does not control account-level authentication state.

6 Two-Phase Validation Protocol

6.1 Protocol Specification

The two-phase validation protocol separates UserOperation validation into two distinct phases:

Phase 1 (V_1): Stateless Pre-Validation. Executed during mempool admission simulation, V_1 performs only gas-bounded structural checks that do not require full cryptographic evaluation:

$$V_1(u) = \text{CheckFormat}(u) \wedge \text{CheckGasLimit}(u) \wedge \text{CheckCommitment}(u) \wedge \text{CheckNonce}(u). \tag{19}$$

CheckFormat verifies that the signature and public key byte lengths match the declared scheme's expected sizes.
CheckGasLimit verifies that the UO's `verificationGasLimit` satisfies:

$$\texttt{verificationGasLimit} \geq \mathcal{G}(|pk|,\, |\sigma_{\text{inner}}|)\,. \tag{20}$$

This relies on the VM's $O(1)$ gas estimator and avoids executing the full verification routine.
CheckCommitment verifies the binding commitment:

$$C = H(pk \| m \| \sigma \| \text{nonce} \| \text{addr}). \tag{21}$$

The commitment C must match a value stored in the UO header, binding all components atomically so that any modification to one element invalidates the commitment. This prevents an attacker from submitting a valid commitment during V_1 and later substituting a different signature in V_2.
CheckNonce performs the standard ERC-4337 nonce validity check.

Phase 2 (V_2): Full Cryptographic Verification. Executed on-chain within the gas budget reserved during Phase 1:

$$V_2(u) = \mathcal{V}^{PQ}(pk,\ H(m),\ \sigma). \tag{22}$$

Phase 2 may access the account's associated storage, subject to ERC-4337 v0.7 constraints storage rules, and is bounded by `verificationGasLimit`.

Composition. A transaction is valid iff $V(u) = 1$ where $V(u) = V_1(u) \wedge V_2(u)$. If $V_1(u) = 1$ and $V_2(u) = 0$, the transaction is included, but any resulting state changes are reverted and penalty gas is charged to the account's deposit in the EntryPoint the standard ERC-4337 behaviour for failed validation.

6.2 Gas Bound Proof

Lemma 2 (V_1 Mempool Safety)**.** *The function V_1, as defined in (19), is mempool-safe under any $G_{\max} \geq G_{V_1}^{\max}$, where*

$$G_{V_1}^{\max} = G_{SLOAD} + G_{keccak} \cdot \left\lceil \frac{|pk| + |\sigma|}{136} \right\rceil + G_{compare} + G_{arithmetic}. \tag{23}$$

Proof. Each component of V_1 incurs either constant or linearly bounded cost in the input size. CheckFormat costs $O(1)$; CheckGasLimit calls $\mathcal{G}(|pk|, |\sigma|)$, which is $O(1)$ by requirement; CheckCommitment computes one keccak256 hash over an input whose size is bounded by $|pk| + |\sigma|$; CheckNonce reads one storage slot at $G_{\text{SLOAD}} = 2{,}100$ gas (cold) or 100 gas (warm). The total is at most $G_{V_1}^{\max}$, which is finite and significantly smaller than $G_{\max}$ for all parameter choices under current NIST standards. □

6.3 Security Theorem

Theorem 3 (EU-CMA Preservation Under Two-Phase Decomposition)**.** *Let Σ^{PQ} be a signature scheme that satisfies EU-CMA security. Then the two-phase validation protocol $V = V_1 \wedge V_2$ preserves EU-CMA security: no probabilistic polynomial-time (PPT) adversary $\mathcal{A}$ can produce a valid UserOperation forgery accepted by V without breaking the EU-CMA security of Σ^{PQ}.*

Proof. Suppose, for contradiction, that an adversary $\mathcal{A}$ produces a forgery $u^* = (pk^*, m^*, \sigma^*, C^*)$ such that $V(u^*) = 1$, where (m^*, σ^*) was not produced by the honest signing oracle.

Since $V(u^*) = 1$, it follows that both $V_1(u^*) = 1$ and $V_2(u^*) = 1$.

In particular,

$$\mathcal{V}^{PQ}(pk^*, H(m^*), \sigma^*) = 1.$$

We construct a reduction $\mathcal{B}$ against the EU-CMA game for Σ^{PQ}: (1) $\mathcal{B}$ receives challenge public key pk^*; (2) it deploys a PQ smart account bound to pk^*; (3) it simulates V_1 deterministically, since V_1 requires no signing oracle access (it checks only structure, commitment, and gas); (4) it answers $\mathcal{A}$'s signing oracle queries by forwarding to the EU-CMA challenger; (5) upon $\mathcal{A}$'s forgery output, $\mathcal{B}$ submits (pk^*, m^*, σ^*) as its EU-CMA forgery.

Since V_1 accepts no signatures independently (acceptance requires $V_2 = 1$), $\mathcal{A}$'s advantage equals $\mathcal{B}$'s advantage against EU-CMA, contradicting the EU-CMA security of Σ^{PQ}. □

Corollary 4 (Commitment Binding)**.** *The inclusion of* nonce *in commitment C (Equation 21) prevents signature replay attacks across different nonce values: a signature valid for nonce n cannot be replayed at nonce $n' \neq n$ without producing an invalid commitment.*

7 Lazy Probabilistic Aggregation

7.1 Motivation

ERC-4337's `IAggregator` interface assumes algebraic aggregation (BLS-style), where n signatures collapse to a constant-size aggregate verifiable in $O(1)$ pairings. ML-DSA and SLH-DSA do not natively admit algebraic aggregation: their security constructions bind each signature to its specific signer-message pair in a non-linear fashion. The *lazy aggregation* protocol addresses this gap for schemes with $\mathcal{I} = 0$.

7.2 Protocol

Let $\mathcal{B} = \{(pk_i, m_i, \sigma_i)\}_{i=1}^{n}$ be a bundle of n PQ-signed UserOperations.

Step 1 (Commitment). The bundler first computes the commitment:

$$C = H\big(\text{abi.encode}\big(\{(pk_i, m_i, \sigma_i)\}_{i=1}^{n}\big)\big) \tag{24}$$

and posts C on-chain in the bundle header.

Step 2 (Random Sampling). A random seed is derived post-commitment:

$$r = H(C \| \texttt{PREVRANDAO} \| B), \tag{25}$$

where B is the block number. `PREVRANDAO` (the post-Merge randomness beacon) provides stronger unpredictability than `BLOCKHASH`, since the proposer cannot fully bias it without incurring consensus penalties.

Step 3 (Selective Verification). From r, a subset $S \subset [n]$ of size $s = \lceil \sqrt{n} \rceil$ is derived deterministically for on-chain verification. Signatures not in S are provisionally accepted under the commitment C subject to dispute window Δ_d.

Step 4 (Dispute Resolution). Any network participant may challenge an unverified signature (pk_j, m_j, σ_j) for $j \notin S$ during Δ_d by submitting it for full on-chain verification. A failed verification triggers a bundler penalty equal to the dispute bond.

7.3 Formal Security Analysis

Theorem 5 (Lazy Aggregation Soundness)**.** *Let $\mathcal{B}$ contain m invalid signatures among a total of n signatures. The probability that all m invalid signatures avoid selection is:*

$$\epsilon_{lazy} = \frac{\binom{n-m}{s}}{\binom{n}{s}} = \frac{\prod_{i=0}^{s-1}(n-m-i)}{\prod_{i=0}^{s-1}(n-i)}. \tag{26}$$

In the special case where $m = 1$ (a single invalid signature):

$$\epsilon_{lazy} = \frac{n-s}{n} = 1 - \frac{1}{\sqrt{n}}. \tag{27}$$

Over k independent bundle rounds:

$$\epsilon_{lazy}^{k} = \left(1 - \frac{1}{\sqrt{n}}\right)^{k} \xrightarrow{k\to\infty} 0. \tag{28}$$

Proof. The m invalid signatures are located at fixed positions in $\mathcal{B}$. The sampling set S is chosen uniformly from $\binom{n}{s}$ possible subsets (determined by r, which is unpredictable at submission time due to the post-commitment derivation from `PREVRANDAO`). The number of subsets avoiding all m invalid positions is $\binom{n-m}{s}$. Equation (26) follows directly.Equation 27) specialises to $m = 1$. Equation (28) follows from the independence of r across blocks. □

Remark 2 (Economic Security)**.** *The protocol achieves economic security provided that the bundler's dispute bond B_d satisfies:*

$$B_d \geq \frac{\text{Profit}(\textit{forged UO})}{1 - \epsilon_{lazy}^{k}}. \tag{29}$$

For $k = 10$, $n = 16$: detection probability ≈ 0.944, requiring $B_d \geq 18\times$ the per-UO forgery profit.

Remark 3 (Dispute Latency Limitation)**.** *The dispute window Δ_d introduces a finality delay for unverified signatures. An adversary who predicts which signatures will not be disputed (e.g., due to low economic incentive to challenge low-value UOs) can exploit this. In practice, Δ_d and B_d must be calibrated jointly to ensure economic viability of disputes across all UO value ranges.*

8 Gas Cost Models

8.1 ML-DSA-44 Closed-Form Model

ML-DSA-44 verification is parameterised by $(k, \ell, n, q) = (4, 4, 256, 8{,}380{,}417)$. Its cost components are:

1. **Matrix expansion.** SHAKE-128 expansion of the public matrix $\mathbf{A}$ from seed ρ: $G_\rho \approx$ 29,440 gas.

2. **NTT-based modular arithmetic relies on efficient modular reduction techniques, commonly implemented using Montgomery multiplication [13]** $k\ell = 16$ polynomial multiplications, each comprising a forward NTT (512 butterfly operations), a pointwise multiplication (256 operations), and an inverse NTT (512 butterflies): $G_{\text{NTT}} = 16 \times (512 + 256 + 512) \times 24 = 491{,}520$ gas.

3. **Hint processing.** Verification of hint vector and reconstruction of $\mathbf{w}_1'$: $\approx$ 12,000 gas.

4. **Challenge hash.** SHAKE-256 computation of μ: $\approx$ 8,000 gas.

5. **Calldata.** σ = 2,420 B, pk = 1,312 B, total $3{,}732 \times 16 = 59{,}712$ gas.

Summing these components yields the following analytic estimate:

$$\mathcal{G}_{\text{ML-DSA-44}} \approx 29{,}440 + 491{,}520 + 12{,}000 + 8{,}000 + 59{,}712 \approx 600{,}672 \text{ gas (compute only).} \quad (30)$$

Empirical measurements indicate a total cost of approximately 1,840,000 gas with an overhead factor $\kappa \approx 3.06$ over the analytic estimate. This overhead is dominated by EVM interpreter dispatch costs per opcode, which are not captured by arithmetic-only cost models and become dominant in high-iteration loops.

8.2 FALCON-512 Model

FALCON-512 verification consists of four primary components: (1) constant-time Huffman-style decompression of the compressed signature representation ($\approx$ 45,000 gas), (2) NTT over the degree-512 NTRU ring with q = 12,289 ($\approx$ 420,000 gas), (3) norm-bound check ($\approx$ 8,000 gas), and (4) calldata: $(666 + 897) \times 16 = 25{,}008$ gas. This yields an analytic estimate of approximately 498,008 gas, while empirical measurements indicate a total of roughly 990,000 gas.

8.3 SLH-DSA-128f Model

SLH-DSA-128f verification involves two main components: (1) FORS tree hashing: 14 chains of length 4, yielding $14 \times 4 = 56$ SHA-256 calls per tree, 14 trees, $14 \times 56 = 784$ calls; (2) hypertree layers: $\approx$ 1,139 SHA-256 calls. Using the SHA-256 precompile, assumed at 72 gas per call:

$$\mathcal{G}_{\text{SLH-DSA-128f}} \approx (784 + 1{,}139) \times 72 + 17{,}008 \times 16 = 138{,}456 + 272{,}128 \approx 410{,}584 \text{ gas (analytic).} \quad (31)$$

Empirical total: $\approx$ 621,000 gas. Here, calldata dominates total cost; as a result, economic viability depends heavily on calldata pricing, particularly improvements introduced by blob-based mechanisms such as EIP-4844.

8.4 Comparative Benchmark

9 Hybrid Classical-PQ Transition Model

9.1 Hybrid Signature Construction

To support incremental migration from classical to post-quantum cryptography, we define a hybrid signature $\sigma_{\text{hyb}} = (\sigma_{\text{ECDSA}}, \sigma_{\text{PQ}})$, together with a binding commitment:

$$C_{\text{bind}} = H(pk_{\text{ECDSA}} \| pk_{\text{PQ}} \| m \| \sigma_{\text{ECDSA}} \| \sigma_{\text{PQ}}) \quad (32)$$

Table 2: Empirical Gas Costs (evaluated on a mainnet fork using evmone 0.12, Solidity 0.8.24, and optimised compilation)

Scheme	Calldata	Compute	Total	vs. ECDSA
secp256k1 (precompile)	1,024	3,000	~6,000	1×
ML-DSA-44	59,712	1,780,000	~1,840,000	307×
FALCON-512	25,008	965,000	~990,000	165×
SLH-DSA-128f	272,128	349,000	~621,000	104×
ML-DSA-44†	59,712	180,000	~240,000	40×
FALCON-512†	25,008	85,000	~110,000	18×
SLH-DSA-128f†	272,128	20,000	~292,000	49×

† Projected with client-native NTT/hash precompile support.

which under standard assumptions that both signature components are jointly bound to the same message and public keys. This prevents cross-signature substitution, where an attacker attempts to mix components from different signatures.

9.2 Verification Policies

We consider three verification policies, corresponding to different stages of the transition.

AND-Mode (Strict).

$$V_{\text{hyb}}^{\wedge} = V_{\text{ECDSA}} \wedge V_{\text{PQ}}.$$

The corresponding security bound satisfies:

$$\text{Adv}^{\text{hyb}} \leq \min(\text{Adv}^{\text{ECDSA}}, \text{Adv}^{\text{PQ}}).$$

Intuitively, a successful forgery requires breaking both schemes, so the overall security is dominated by the stronger component.

OR-Mode (Transitional).

$$V_{\text{hyb}}^{\vee} = V_{\text{ECDSA}} \vee V_{\text{PQ}}.$$

In this case:

$$\text{Adv}^{\text{hyb}} \leq \text{Adv}^{\text{ECDSA}} + \text{Adv}^{\text{PQ}}.$$

This policy is suitable for backward compatibility, but it introduces a downgrade risk: if an adversary can break ECDSA, then the PQ component is effectively bypassed.

Staged-Mode.

$$V_{\text{staged}}(t) = \begin{cases} V_{\text{ECDSA}} & t < t_1, \\ V_{\text{ECDSA}} \wedge V_{\text{PQ}} & t_1 \leq t < t_2, \\ V_{\text{PQ}} & t \geq t_2. \end{cases} \tag{33}$$

Here, the system transitions gradually from classical to post-quantum verification. The transition points t_1 and t_2 are governance parameters defined on-chain via the Scheme Registry.

9.3 Forward Security Limitation

ECDSA signatures are publicly recorded on-chain and remain permanently accessible. If large-scale quantum capabilities become available, an adversary could retrospectively recover private keys from previously observed signatures, compromising all past ECDSA-based authentication.

As a consequence, AND-mode primarily delays exposure rather than eliminating it, while OR-mode inherits the weakest component and becomes insecure once ECDSA is broken. Only the final stage of the staged policy, where verification relies solely on PQ signatures, provides long-term quantum resistance.

Hybrid constructions should therefore be viewed as transitional mechanisms that facilitate migration, rather than as permanent security solutions.

10 Migration Protocol for EIP-7702

10.1 First-Use Vulnerability

EIP-7702 requires an ECDSA-signed delegation transaction as the initial step in smart account migration. A quantum-capable adversary observing this transaction in the public mempool could: (1) recover the ECDSA private key from the delegation signature via Shor's algorithm; (2) construct a competing delegation to an attacker-controlled account; and (3) front-run the user's delegation transaction.

10.2 Commitment-Protected Migration Protocol

Phase A (Commitment, pre-migration). The user first commits to their future PQ public key on-chain before initiating the delegation:

$$C_{\mathrm{PQ}} = H(pk_{\mathrm{PQ}} \| \mathrm{addr}_{\mathrm{EOA}} \| \mathrm{salt}). \tag{34}$$

This transaction writes only a hash; pk_{PQ} is not exposed.

Phase B (Delegation, migration). During delegation, the target account's `initialize` function verifies:

$$H(pk_{\mathrm{PQ}} \| \mathrm{addr}_{\mathrm{EOA}} \| \mathrm{salt}) = C_{\mathrm{PQ}}. \tag{35}$$

An attacker who attempts to front-run using a different target account cannot satisfy this check without knowledge of pk_{PQ}.

Proposition 6 (Migration Protocol Security)**.** *The commitment-protected migration protocol is secure against quantum front-running, assuming the preimage resistance of H. A quantum adversary that recovers the ECDSA private key from the Phase B transaction cannot derive*

pk_{PQ} from C_{PQ} without inverting H.Grover's algorithm provides only a quadratic speedup, reducing preimage security from 128 bits to 64 bits which remains computationally infeasible under current and near-term projected quantum capabilities.

11 Comparative Evaluation

11.1 Multi-Paradigm Compatibility

Table 3: Architecture Compatibility Across Account Abstraction Paradigms

Feature	**4337 v0.6**	**4337 v0.7**	**RIP-7560**	**EIP-7702**	**EIP-3074**
VM Interface	✓	✓	✓	✓	✗
V_1/V_2 Split	Partial	✓	✓	N/A	✗
Lazy Aggregation	✓	✓	✓	✗	✗
Scheme Registry	✓	✓	✓	✓	✗
Migration Protocol	N/A	N/A	N/A	✓	✗

EIP-3074 is structurally incompatible with the VM interface, since its `AUTH` opcode hardcodes ECDSA recovery at the EVM level. Supporting PQ schemes in EIP-3074-style patterns would require a new opcode (`AUTH2`) that accepts a `schemeId` parameter and delegates verification to the Scheme Registry.

11.2 Comparison with Prior Work

While the compatibility and coverage of features of these solutions are summarised in the tables in the section, such an overview does not explain *why* these previous solutions are structurally inadequate for a post-quantum execution model. This is done systematically in the subsection below by going through each class of previous or contemporary solutions and pinpointing the exact reason that class fails to meet the criteria in Section 3.

11.2.1 ECDSA Precompile Verification

Ethereum's native transaction authentication mechanism invokes `ecrecover` at a protocol-fixed cost of 3,000 gas (Equation 7). This cost is not computed at runtime it is a consensus constant encoded in the EVM specification. The abstraction is possible because ECDSA verification over secp256k1 executes in $O(1)$ group operations, and the `ecrecover` precompile is implemented in native client code outside the EVM interpreter loop. This gives a perfect match for off-chain simulation in the bundler with what happens on chain, as verification costs for ECDSA are gas-deterministic, mempool-safe, and simulation-faithful.

There are at least three independent reasons why a naive post-quantum drop-in substitution into such a model fails, which are detailed in Section 3. First, PQ verification algorithms execute $\Theta(k\ell n \log n)$ arithmetic operations (Equation 5), which is asymptotically unbounded relative to

any fixed gas constant. Theorem 1 shows that no single $G_{\max}$ can bound verification cost across all NIST security levels. Second, NTT-intensive inner loops exhibit JIT warm-up divergence of 8–15% between simulation and on-chain environments (Section 3, Gas Non-Determinism), meaning that precompile-style cost abstraction cannot be replicated at the EVM layer without a new native precompile or opcode. Third, hash-based schemes such as SLH-DSA expose adversarially controllable tree traversal paths (the Merkle path is determined by the signature, which is submitted by the user), creating worst-case execution paths that do not arise for algebraic schemes. These three failure modes are independent: resolving any one of them does not resolve the others [16, 17].

11.2.2 ERC-4337 Account Abstraction

ERC-4337 [9] introduced a programmable validation layer through the `validateUserOp` interface, which allows each smart account to define arbitrary signature verification logic. This provides the necessary execution flexibility for post-quantum verification, but it does not by itself solve the systems-level constraints introduced by computationally expensive signature schemes.

The primary limitation of baseline ERC-4337 is the absence of a mechanism for enforcing mempool-safe gas bounds for high-cost verification routines. Bundlers simulate `validateUserOp` off-chain to estimate verification cost, and include a UserOperation only if the estimated cost remains within `verificationGasLimit`.

For ECDSA-based accounts, this simulation is faithful because ECDSA verification is implemented through the native `ecrecover` precompile, whose gas cost is fixed by protocol design.

For post-quantum smart accounts implementing full verification in Solidity, simulation becomes unreliable due to gas non-determinism, as formalised in Definition 2. Differences between off-chain simulation and on-chain execution environments may cause actual gas consumption to exceed the simulated gas limit, leading to bundle failure while still consuming the bundler's deposit.

Furthermore, ERC-4337 does not define a typed gas estimator interface. As a result, an ML-DSA-44 verifier deployed under vanilla ERC-4337 must assign `verificationGasLimit` by convention, without any formal guarantee that the verification cost is upper-bounded.

The Verification Module abstraction introduced in Section 4 resolves this by requiring each registered verifier to expose an $O(1)$ estimator $\mathcal{G}(|pk|, |\sigma|)$ satisfying Definitions 11 and 12, thereby converting gas estimation from heuristic simulation into a provably bounded interface.

11.2.3 ERC-7579 Modular Smart Accounts

ERC-7579 [12] standardises modular smart accounts by decomposing account logic into interchangeable validator, executor, hook, and fallback modules.

A validator module implementing a post-quantum signature scheme can, in principle, be installed in any ERC-7579-compliant account, and the standard supports multiple simultaneous validators.

However, ERC-7579 does not enforce the gas-safety invariants required for post-quantum verification.

Although the standard specifies how modules are installed and invoked, it does not require validator modules to expose a gas estimator, does not constrain worst-case verification cost, and does not define a separate pre-validation phase distinct from full cryptographic verification.

As a result, a post-quantum validator deployed under ERC-7579 inherits the three incompatibilities established in Theorem 1: unbounded cost, gas non-determinism, and adversarial triggering of worst-case execution paths.

The "Partial" EU-CMA entry in Table 4 reflects an additional limitation.

ERC-7579 does not enforce scheme isolation during validation. A validator module may invoke external contracts during signature verification, introducing reentrancy risk and dependence on external state that lies outside the EU-CMA security model of the underlying post-quantum scheme.

By contrast, the PQSigAbstract Verification Module interface requires the verification function to satisfy , which prohibits external calls outside the approved precompile set $\mathcal{P}_{\text{pre}}$.

This eliminates external-state dependency by construction.

11.2.4 Prior Modular Verification Research

Belles-Munoz et al. [14] studied efficient elliptic curve arithmetic over twisted Edwards curves for zero-knowledge circuits. While their techniques are relevant to ZK-based verification backends, the work is not scheme-agnostic: the circuit constructions are specific to the Edwards25519 group and do not generalise to lattice or hash-based post-quantum primitives. The approach also does not address the gas non-determinism or mempool admission problems identified in Section 3.

Atapoor et al. [15] proposed threshold ECDSA with identity-based access control for decentralised key management. Their construction achieves EU-CMA security and supports multiple signers through a threshold scheme, making it relevant to multi-party Ethereum accounts. However, the underlying primitive remains ECDSA over a classical elliptic curve; the scheme inherits ECDSA's vulnerability to Shor's algorithm [1] and provides no quantum resistance. Furthermore, the scheme is not scheme-agnostic: the threshold protocol is algebraically coupled to the ECDSA signing equation and cannot be directly applied to ML-DSA or SLH-DSA, which do not admit an analogous linear secret-sharing decomposition.

Fernández-Caramés and Fraga-Lamas [17] and Buser et al. [16] provide surveys of PQ signature schemes in blockchain contexts. These works identify the correct direction for long-term migration and document the cryptographic properties of candidate schemes, but they do not address the EVM-layer systems problem. Specifically, they do not formalise the interaction between PQ scheme verification cost and Ethereum's mempool admission constraints, do not propose a VM abstraction or gas estimator interface, and do not consider the EIP-7702 first-use vulnerability or the two-phase validation decomposition required to maintain mempool safety.

Fan [20] discusses a layered migration roadmap for post-quantum Ethereum and explicitly recognises that staged deployment strategies are necessary. This work corroborates the phased

approach adopted in the hybrid transition model (Section 9) and the migration protocol (Section 10). However, Fan's discussion is architectural and policy-oriented; it does not provide formal gas models, a typed VM interface, or the EU-CMA preservation proof established in Theorem 3.

11.2.5 Why PQSigAbstract Succeeds Where Prior Approaches Do Not

What all the failure modes above share is that previous approaches either inherit ECDSA's classical security assumptions or push the verification to an application layer, which does not require the structural invariants necessary to ensure mempool admission safety and gas determinism.

The PQSigAbstract approach addresses the root cause rather than each separate symptom. It rests on four structural properties that jointly ensure none of the incompatibilities of Theorem 1.

First, the V_1/V_2 split (Section 6) resolves the mempool admission problem by ensuring that only a gas-bounded structural pre-check is performed prior to bundle inclusion, deferring full cryptographic verification to on-chain execution where the gas limit is enforced contractually through the EntryPoint. Lemma 2 proves that V_1 is mempool-safe for any parameter choice under current NIST standards.

Second, the VM abstraction introduced in this paper (Section 4) remedies this by requiring every registered verifier to expose a gas-sound, gas-tight estimator $\mathcal{G}(|pk|, |\sigma|)$ satisfying Definitions 11–12, thereby turning bundler-side gas prediction into a formal guarantee rather than a best-effort heuristic.

Third, the commitment binding $C = H(pk\|m\|\sigma\|\text{nonce}\|\text{addr})$ (Equation 21) prevents the signature substitution attack that arises in any two-phase design: an adversary cannot submit a lightweight pre-image in V_1 and substitute a different signature in V_2, because any such substitution invalidates the commitment and causes V_1 to reject the modified UserOperation.

Fourth, EU-CMA security is preserved unconditionally under the two-phase decomposition (Theorem 3), because V_1 imposes no independent acceptance condition on signatures: a UserOperation is accepted only when both V_1 and V_2 return 1, and V_2 performs the identical full cryptographic verification that would be performed in a single-phase design. The two-phase structure therefore adds no attack surface beyond that of the underlying PQ scheme itself.

These four properties can be interpreted as a proof that PQSigAbstract is mempool safe, gas deterministic, commitment secure, and EU-CMA preserving: none of these can be guaranteed for any of the previous attempts reviewed above. able 4 summarises this comparison across the six axes most relevant to practical deployment.

A key limitation shared by all prior approaches is the absence of a gas-bounded estimator $\mathcal{G}$ that enables safe bundler inclusion without requiring full simulation, as well as the absence of analysis addressing the EIP-7702 first-use vulnerability. These gaps are addressed by the VM abstraction (Section 4), the two-phase protocol (Section 6), and the commitment-protected migration mechanism (Section 10) respectively.

Table 4: Comparison with Related Approaches

Approach	Scheme Agnostic	Gas Bounded	**Aggregation**	Multi AA	EU-CMA Secure	**Migration**
Belles-Munoz et al. [14]	✗	✗	✗	✗	✓	✗
Atapoor et al. [15]	✓	✗	✗	✗	✓	✗
ERC-7579 [12]	✓	✗	✗	Partial	Partial	✗
PQSigAbstract (this work)	✓	✓	Lazy	✓	✓	✓

12 Discussion

12.1 Economic Implications

The 100–300× gas overhead of PQ verification represents asignificant economic barrier under the current EVM cost structure. At 20 gwei per gas and ETH priced at approximately $2,000, a single ML-DSA-44 verification costs around $0.07, versus $0.0002 for ECDSA. For institutional and DAO accounts, this overhead is frequently negligible relative to the asset values being protected. For consumer wallets executing frequent low-value transactions, it can become prohibitive.

This naturally implies a tiered adoption model. High-security accounts, including institutional custodians, enterprise treasuries, and DAO governance contracts, can migrate immediately under the current EVM cost structure, since their per-operation economic value far exceeds the marginal verification cost. Broader consumer adoption will likely depend on PQ precompile support, which reduces costs to 18×–49× ECDSA (Table 2), bringing PQ verification within a range comparable to the overhead already accepted by users of Layer-2 rollups relative to Layer-1 execution. This trajectory mirrors the historical pattern of ECC replacing RSA in TLS deployments: early adoption in high-value server infrastructure preceded broad consumer endpoint migration by several years.

12.2 Second-Order Effects on Mempool Dynamics

Changes in mempool dynamics go well beyond the impact on individual transaction costs when smart accounts with PQ signatures are introduced.

Bundle composition. A bundler including even one SLH-DSA UserOperation consumes $\approx$ 621,000 gas for verification alone. Bundlers will likely need to adopt per-scheme gas budgeting policies, effectively partitioning their block space allocation across classical and PQ accounts. This is analogous to the block space management already required by the EIP-4844 blob fee market [19], where different data categories compete under separate pricing mechanisms.

Priority fee dynamics. Gas-heavy PQ UserOperations must offer higher priority fees to compete with cheaper ECDSA transactions during periods of block space contention. This creates a fee premium for quantum-secure accounts during the transition period, which may slow consumer adoption if left unaddressed at the protocol level.

Censorship risk. If PQ transactions are consistently outcompeted on priority fees, quantum-resistant accounts may face de facto censorship during high-load periods. Protocol-level fee subsidies or reserved block space for approved PQ schemes, analogous to EIP-4844's separate blob fee market, may be required in practice. The Scheme Registry's governance mechanism (Section 5) provides a natural on-chain coordination point for such policies, since approved schemes are already enumerated with their security levels and cost parameters.

12.3 Real-World Application Domains

Beyond the abstract protocol design, PQSigAbstract addresses concrete deployment scenarios that are already relevant today and will become critical as quantum hardware matures.

Smart wallets and consumer accounts. The majority of Ethereum's user-facing value is controlled by externally owned accounts that rely exclusively on ECDSA. Smart wallet frameworks such as those built on ERC-4337 provide the deployment path for PQ migration without requiring consensus-layer changes [9]. Under PQSigAbstract, a user can deploy a smart account bound to an ML-DSA or FALCON verification module, migrate their assets through the commitment-protected EIP-7702 protocol (Section 10), and thereafter authenticate with a quantum-resistant key without any change to wallet software beyond support for the new signing algorithm.

The VM interface (Section 4) ensures that wallet front-ends can retrieve an accurate gas estimate before submitting a UserOperation, enabling accurate fee quotation without requiring simulation.

Enterprise custody. Institutional custodians holding digital assets on behalf of clients operate under regulatory and fiduciary obligations that require long-term key security guarantees. For assets expected to be held for multi-year periods, the harvest-now-decrypt-later threat (Section 1) implies that the effective security horizon of ECDSA-protected accounts has already begun to shorten. PQSigAbstract's hybrid transition model (Section 9) allows custodians to deploy AND-mode verification, requiring valid signatures under both ECDSA and a PQ scheme, during a transitional period. This provides immediate defence against harvest attacks while preserving backward compatibility with existing settlement infrastructure. The staged-mode policy formalised in Section 9 allows the transition to pure PQ verification to be enforced at a governance-specified block height, enabling auditable, scheduled migration without operational disruption.

Institutional settlement and financial infrastructure. Decentralised finance protocols handling large daily settlement volumes represent some of the highest-value targets on Ethereum. For these protocols, the absolute cost of PQ verification is secondary to the assurance that signed transactions cannot be forged by a quantum adversary. The formal EU-CMA preservation proof (Theorem 3) provides the cryptographic foundation required for security audits of settlement contracts, since it establishes that the two-phase architecture introduces no additional forgery surface beyond the underlying PQ scheme. Lazy probabilistic aggregation (Section 7) further reduces verification overhead for high-throughput settlement scenarios, achieving $O(\sqrt{n})$

amortised on-chain cost per bundle, with economic security calibrated through the bundler dispute bond (Equation 29).

Layer-2 rollups. Rollup sequencers aggregate large volumes of user transactions and post compressed state updates to Ethereum mainnet. Post-quantum security for rollup users requires that both the sequencer's commitment and the individual user's transaction signature are quantum-resistant. PQSigAbstract is compatible with RIP-7560 [11], which introduces native account abstraction at the rollup execution layer, enabling PQ verification modules to be deployed on rollup networks without modifications to the underlying bridging or fraud proof infrastructure. The VM registry can in principle be synchronised across rollup deployments, allowing a scheme approved on Ethereum mainnet to be automatically recognised by compliant rollup implementations, reducing the coordination overhead of cross-chain PQ migration. Cross-chain VM registry synchronisation remains an open research question (see Section 12, Open Problems), but the architecture provides the necessary on-chain coordination primitive.

Decentralised identity systems. Ethereum-based decentralised identity frameworks, including those built on the W3C Decentralised Identifier (DID) specification, use on-chain public keys to authenticate identity documents with lifetimes potentially spanning decades. [18] An identity key protected only by ECDSA may be compromised retrospectively by a future quantum adversary, since historical transactions that expose the public key are permanently recorded on-chain. The commitment-protected migration protocol (Section 10) allows identity systems to migrate their on-chain key commitments to PQ public keys without exposing the new key to quantum front-running during the transition transaction. The scheme registry's quarantine mechanism further ensures that a newly registered PQ scheme has been subject to a minimum observation window before it can be used to authenticate identity assertions, providing a governance backstop against premature deployment of insufficiently audited implementations.

12.4 Commercialisation Pathways

The architecture described in this work is designed as an open protocol layer, but several components lend themselves to commercial deployment or integration.

The Scheme Registry and VM interface are natural standardisation targets, analogous to the role that ERC-4337 [9] and ERC-7579 [12] have played in unifying the smart account ecosystem. A standardised registry, maintained by a neutral governance body or deployed as an immutable singleton contract, would allow wallet developers, auditors, and DeFi protocols to rely on a shared, authoritative list of approved PQ verification modules without duplicating audit costs. This creates economic value through reduced integration overhead and shared security assurance across the ecosystem.

Verification Module implementations themselves represent audit and deployment products. A correctly implemented, gas-tight ML-DSA-44 verification module, certified against the gas soundness and tightness invariants of Section 4, is a deployable smart contract with direct utility for any ERC-4337-compatible wallet. Security firms already provide audited reference implementations of ECDSA and BLS verification contracts; PQ verification modules represent the natural next product category in this space.

The lazy probabilistic aggregation protocol (Section 7) is particularly relevant to bundler infrastructure. Bundler operators who implement aggregation-aware inclusion policies can offer lower effective verification costs to PQ account holders, potentially commanding a premium for this service in an open bundler market. The dispute bond mechanism (Equation 29) introduces a financial instrument that could be structured as an insured service: a bundler posting a sufficiently large dispute bond effectively offers a verification guarantee to its clients, with the bond size calibrated to the economic value of the operations being bundled.

Finally, the EIP-7702 migration protocol (Section 10) is directly productisable as a user-facing migration service. A wallet provider offering a one-click PQ migration flow would handle the commitment pre-registration (Phase A), delegation transaction (Phase B), and key binding atomically on behalf of the user, abstracting the cryptographic complexity of the migration protocol into a consumer product. This is commercially analogous to the key management services already offered by hardware security module (HSM) providers and institutional custody platforms, but extended to support quantum-resistant key material.

12.5 Adoption Pathways and Deployment Trajectory

The PQSigAbstract architecture does not require any modification to Ethereum's consensus rules or core client implementations to become operational. The VM interface, Scheme Registry, and two-phase validation protocol are all deployable as smart contracts on the existing EVM, with full compatibility with ERC-4337 v0.7. This means that deployment can begin immediately, in parallel with longer-horizon efforts to introduce native PQ precompiles or protocol-level account abstraction via RIP-7560 [11].

A realistic adoption trajectory proceeds in three stages. In the near term, reference VM implementations for ML-DSA-44 and FALCON-512 can be deployed and audited on mainnet, with the Scheme Registry initialised in quarantine mode to allow ecosystem observation before any accounts rely on them for authentication. This stage is entirely compatible with the current EVM and requires no coordination beyond the smart contract deployment itself.

In the medium term, as PQ precompiles are proposed through the EIP process (a direction already discussed in the Ethereum research community [20]), the VM interface's pluggable architecture allows existing accounts to transparently migrate to precompile-backed VMs by updating their scheme binding, without changing their public keys or migrating their assets. The 15×–20× gas cost reduction projected in Table 2 would unlock consumer-scale adoption.

In the long term, native account abstraction via RIP-7560 [11] would elevate the VM interface from an application-layer convention to a consensus-layer standard, enabling PQ verification at the same protocol depth as ECDSA today. At this stage, the Scheme Registry would likely transition to a protocol governance mechanism, with scheme approval integrated into the EIP standardisation process.

Fan [20] has proposed a complementary layered roadmap for PQ Ethereum migration that is broadly consistent with this trajectory. The contribution of this work relative to that roadmap is to provide the concrete cryptographic and systems-level specification that the VM interface, gas models, and security proofs, required to implement the near-term stage of that roadmap on

production infrastructure.

12.6 Open Problems

1. **Stateless Client Compatibility.** Ethereum's stateless client roadmap (EIP-6800, Verkle trees) requires state witnesses for all accessed storage. Encoding VM parameters in contract bytecode, provable via code witnesses, is a possible alternative, but it restricts dynamic parameter updates and requires re-deployment to change scheme parameters.

2. **Quantum-Safe Address Derivation.** Ethereum addresses are derived from ECDSA public keys. Accounts that have submitted at least one transaction expose their public key on-chain; a sufficiently capable quantum adversary could derive the corresponding private key from any such transaction. Handling the address-space transition for existing accounts is a problem that is distinct from, and arguably more challenging than, PQ signature integration.

3. **Cross-Chain PQ Consistency.** Layer-2 networks inherit execution-layer semantics but differ in mempool policies andgas pricing. Cross-chain VM registry synchronisation and unified gas models across heterogeneous rollup environments remain open research questions.

4. **ZK-PQ Verification.** Generating a ZK proof of a valid ML-DSA signature can reduce on-chain cost to approximately 300,000 gas (PLONK or Groth16 verifier), at the cost of approximately 200–500 ms of proof generation latency client-side [14]. The VM interface's `estimateGas` and `verify` functions provide natural integration points for ZK-backed VM implementations, where the verifier contract checks the ZK proof rather than performing native NTT arithmetic. This direction could reduce PQ verification overhead to within 50× of ECDSA without requiring precompile support, at the cost of a proof generation burden on the signing client.

13 Conclusion

Post-quantum signature adoption in Ethereum is best understood as a verification architecture problem, rather than purely a cryptographic parameter problem. We show that full PQ verification is incompatible with mempool-layer validation under fixed gas bounds, and formally characterise the conditions under which this incompatibility arises. PQSigAbstract, a modular multi-layer architecture, addresses these failure modes without requiring modifications to Ethereum's consensus rules.

The main contributions of this work include the Verification Module abstraction, a two-phase validation protocol with preservation of EU-CMA security, lazy probabilistic aggregation with formally analysed soundness, and a commitment-protected migration protocol. Together, these components provide a deployable and forward-compatible pathway for quantum-resistant Ethereum accounts. The architecture is scheme-agnostic, compatible with ERC-4337 v0.7,

RIP-7560, and EIP-7702, and can benefit from protocol-level precompile support as it becomes available.

The urgency is not purely speculative. Harvest-now-decrypt-later strategies already apply to publicly exposed signatures. The migration window for long-lived accounts is effectively already open, and the verification architecture required for a safe transition should be established before these risks materialise at scale.

A Formal System Model

Definition 7 (System Model)**.** *The PQSigAbstract system can be formalised as a tuple* $\mathcal{S} = (\mathcal{U}, \mathcal{VM}, \mathcal{R}, \mathcal{E}, \Pi)$*, where* $\mathcal{U}$ *is the set of user operations,* $\mathcal{VM}$ *is the set of registered verification modules,* $\mathcal{R}$ *is the scheme registry (Section 5),* $\mathcal{E}$ *is the EVM execution environment (including gas accounting and storage model), and* $\Pi = \{V_1, V_2, C\}$ *is the two-phase validation protocol.*

B Asymptotic Complexity Summary

Table 5: Asymptotic Complexity (ring dimension n, module dimensions $k \times \ell$)

Metric	**ECDSA**	**PQ (ML-DSA)**
Time (native)	$O(1)$	$O(k\ell \cdot n \log n)$
Time (EVM)	$O(1)$	$O(k\ell \cdot n \log n)$
Signature space	$O(1)$	$O(n)$
Public key space	$O(1)$	$O(kn)$
Gas (EVM)	Constant (precompile)	$\Theta(k\ell \cdot n \log n)$
Calldata gas	$O(1)$	$O(n)$

C Adversarial Bound

For any probabilistic polynomial-time (PPT) adversary $\mathcal{A}$ interacting with the system:

$$\mathrm{Adv}_{\mathcal{A}}^{\text{PQSigAbstract}} \leq \mathrm{Adv}_{\Sigma,\mathcal{B}}^{\text{EU-CMA}}, \tag{36}$$

for the efficient reduction $\mathcal{B}$ of Theorem 3. This bound holds regardless of $\mathcal{A}$'s ability to exploit the registry, aggregation protocol, or migration pathway, as each component reduces to the security of the underlying PQ signature scheme.

References

[1] P. W. Shor, "Algorithms for quantum computation: Discrete logarithms and factoring," in *Proc. 35th Annu. Symp. Foundations of Computer Science (FOCS)*, Santa Fe, NM, USA, Nov. 1994, pp. 124–134, doi: 10.1109/SFCS.1994.365700. [Online]. Available: https://ieeexplore.ieee.org/document/365700

[2] M. Webber, V. Elfving, S. Weidt, and W. K. Hensinger, "The impact of hardware specifications on reaching quantum advantage in the fault tolerant regime," *AVS Quantum Sci.*, vol. 4, no. 1, p. 013 801, Mar. 2022, doi: 10.1116/5.0073075.

[3] National Institute of Standards and Technology, *Module-Lattice-Based Key-Encapsulation Mechanism Standard (ML-KEM)*, FIPS Pub. 203, NIST, Gaithersburg, MD, USA, 2024. [Online]. Available: https://doi.org/10.6028/NIST.FIPS.203

[4] National Institute of Standards and Technology, *Module-Lattice-Based Digital Signature Standard (ML-DSA)*, FIPS Pub. 204, NIST, Gaithersburg, MD, USA, 2024. [Online]. Available: https://doi.org/10.6028/NIST.FIPS.204

[5] National Institute of Standards and Technology, *Stateless Hash-Based Digital Signature Standard (SLH-DSA)*, FIPS Pub. 205, NIST, Gaithersburg, MD, USA, 2024. [Online]. Available: https://doi.org/10.6028/NIST.FIPS.205

[6] P.-A. Fouque, J. Hoffstein, P. Kirchner, V. Lyubashevsky, T. Pornin, T. Prest, T. Ricosset, G. Seiler, W. Whyte, and Z. Zhang, "Falcon: Fast-Fourier lattice-based compact signatures over NTRU," NIST Post-Quantum Cryptography Standardization, Version 1.2, 2020. [Online]. Available: https://falcon-sign.info/falcon.pdf

[7] L. Ducas, E. Kiltz, T. Lepoint, V. Lyubashevsky, P. Schwabe, G. Seiler, and D. Stehle, "CRYSTALS-Dilithium: Algorithm specifications and supporting documentation," NIST Post-Quantum Cryptography Standardization, Round 3 Submission, Version 3.1, Feb. 2021. [Online]. Available: https://pq-crystals.org/dilithium/data/dilithium-specification-round3-20210208.pdf

[8] D. J. Bernstein, A. Hülsing, S. Kölbl, R. Niederhagen, J. Rijneveld, and P. Schwabe, "SPHINCS$^+$: Submission to the NIST post-quantum project," NIST Post-Quantum Cryptography Standardization, Version 3.1, 2022. [Online]. Available: https://sphincs.org/data/sphincs+-r3.1-specification.pdf

[9] V. Buterin, Y. Weiss, T. Forbs, A. Maller, S. Shahaf, and D. Holz, "ERC-4337: Account abstraction using alt mempool," Ethereum Improvement Proposals, ERC-4337, Sep. 2021. [Online]. Available: https://eips.ethereum.org/EIPS/eip-4337

[10] V. Buterin, S. Cairus, A. Papini, and M. Garneau, "EIP-7702: Set EOA account code for one transaction," Ethereum Improvement Proposals, EIP-7702, May 2024. [Online]. Available: https://eips.ethereum.org/EIPS/eip-7702

[11] N. Drinovsky and V. Buterin, "RIP-7560: Native account abstraction," Rollup Improvement Proposals, RIP-7560, Oct. 2023. [Online]. Available: https://ethereum-magicians.org/t/rip-7560-native-account-abstraction/16664

[12] D. Finestone, F. Roos, M. Schoedon, and Z. Gluck, "ERC-7579: Minimal modular smart accounts," Ethereum Improvement Proposals, ERC-7579, Nov. 2023. [Online]. Available: https://eips.ethereum.org/EIPS/eip-7579

[13] P. L. Montgomery, "Modular multiplication without trial division," *Math. Comput.*, vol. 44, no. 170, pp. 519–521, Apr. 1985, doi: 10.1090/S0025-5718-1985-0777282-X. [Online]. Available: https://www.ams.org/journals/mcom/1985-44-170/S0025-5718-1985-0777282-X/S0025-5718-1985-0777282-X.pdf

[14] M. Belles-Munoz, A. Bureau, J. Munoz-Tapia, A. Rubio, and J. Baylina, "Twisted Edwards elliptic curves for zero-knowledge circuits," *Mathematics*, vol. 10, no. 23, p. 4483, Nov. 2022, doi: 10.3390/math10234483.

[15] S. Atapoor, K. Baghery, and D. Cozzo, "Efficient threshold ECDSA with identity-based access control for decentralised key management," in *Proc. Int. Conf. Financial Cryptography and Data Security (FC 2023)*, Bol, Croatia, May 2023, pp. 342–360, doi: 10.1007/978-3-031-47754-6_20. [Online]. Available: https://link.springer.com/chapter/10.1007/978-3-031-47754-6_20

[16] M. Buser, R. Dowsley, M. Esgin, C. Gritti, S. Kasra Kermanshahi, V. Kuchta, J. A. Liu, R. Steinfeld, Z. Xu, and D. Liu, "A survey on exotic signatures for post-quantum blockchain: challenges and research directions," *ACM Comput. Surv.*, vol. 55, no. 12, pp. 1–32, Dec. 2022, doi: 10.1145/3572771. [Online]. Available: https://dl.acm.org/doi/10.1145/3572771

[17] T. M. Fernández-Caramés and P. Fraga-Lamas, "Towards post-quantum blockchain: A review on blockchain cryptography resistant to quantum computing attacks," *IEEE Access*, vol. 8, pp. 21 091–21 116, Jan. 2020, doi: 10.1109/ACCESS.2020.2969850. [Online]. Available: https://ieeexplore.ieee.org/document/8967098/

[18] M. Sporny, D. Longley, M. Sabadello, D. Reed, O. Steele, and C. Allen, "Decentralized identifiers (DIDs) v1.0," W3C Recommendation, Jul. 2022. [Online]. Available: https://www.w3.org/TR/did-core/

[19] D. Feist, D. Buterin, and M. Garneau, "EIP-4844: Shard blob transactions," Ethereum Improvement Proposals, EIP-4844, Feb. 2022. [Online]. Available: https://eips.ethereum.org/EIPS/eip-4844

[20] X. Fan, "Smooth migration to post-quantum Ethereum: A layered roadmap," Ethereum Research Forum, Ethereum Foundation, May 2024. [Online]. Available: https://ethresear.ch/t/smooth-migration-to-pq-ethereum/19484