Post-Quantum Security for Digital Securities Infrastructure

Zach Kelling

Lux Industries

2024

Abstract

Digital securities have multi-decade lifetimes: a 30-year bond issued today must remain unforgeable in 2056. Current blockchain infrastructure relies on ECDSA (secp256k1) and Ed25519, both broken in polynomial time by Shor's algorithm on a cryptographically relevant quantum computer (CRQC). We present the post-quantum security architecture of the Lux network, which deploys three NIST-standardized post-quantum algorithms as native EVM precompiles: ML-KEM (FIPS 203) for key encapsulation, ML-DSA (FIPS 204) for digital signatures, and SLH-DSA (FIPS 205) for stateless hash-based signatures. The precompiles are activated at genesis block 0, making post-quantum cryptography available from chain launch rather than requiring a hard fork. We describe the hybrid classical/post-quantum migration path: during the transition period, transactions carry both an ECDSA signature and an ML-DSA signature; the chain accepts the transaction if either signature verifies, enabling gradual migration without breaking existing tooling. We prove that the hybrid scheme achieves "best of both worlds" security: it is unforgeable as long as *either* the classical or the post-quantum assumption holds. We give gas cost benchmarks for all three algorithms at NIST security levels 2, 3, and 5, and explain why securities infrastructure specifically—with its long asset lifetimes and regulatory record-keeping requirements—must adopt post-quantum cryptography before the quantum threat materializes.

Contents

1 Introduction

The quantum threat to elliptic curve cryptography is not a question of whether but when. Shor's algorithm [1] solves the elliptic curve discrete logarithm problem in polynomial time on a quantum computer. Once a CRQC exists, every ECDSA private key is recoverable from its public key, and every historical signature becomes forgeable.

For cryptocurrencies with short-lived state (e.g., daily trading balances), the quantum threat can be mitigated by migration at the time CRQCs become imminent. For digital securities, this strategy fails for three reasons:

1. **Long asset lifetimes.** Bonds, real estate tokens, and structured products have maturities of 10–30 years. A security issued in 2026 with a 30-year maturity must remain unforgeable until 2056.
2. **Harvest-now, decrypt-later.** An adversary can record signed transactions today and forge them once a CRQC is available. For securities, this means retroactive forgery of ownership transfers.
3. **Regulatory record-keeping.** SEC Rule 17a-4 requires broker-dealers to retain records for 6 years; FINRA Rule 4511 requires 6 years. If signatures become forgeable during the retention period, the audit trail is compromised.

The Lux network addresses this by deploying post-quantum cryptography as native EVM precompiles at genesis. This is not a future roadmap item; it is a day-one capability.

2 Post-Quantum Algorithms

2.1 ML-KEM (FIPS 203): Key Encapsulation

Definition 2.1 (Module-Lattice-Based Key Encapsulation)**.** *ML-KEM provides IND-CCA2 secure key encapsulation based on the Module Learning With Errors (MLWE) problem. It replaces ECDH for key agreement.*

- ***KeyGen*** $\rightarrow (pk, sk)$*: generates a lattice-based key pair.*
- ***Encapsulate***$(pk) \rightarrow (ct, ss)$*: produces a ciphertext* ct *and shared secret* ss*.*
- ***Decapsulate***$(sk, ct) \rightarrow ss$*: recovers the shared secret.*

ML-KEM is used on the Lux network for:

- Encrypted peer-to-peer communication between validators.
- Key agreement for threshold FHE key distribution.
- Secure channel establishment for MPC signing sessions.

Level	k	PK (B)	SK (B)	CT (B)	SS (B)
ML-KEM-512 (Level 1)	2	800	1,632	768	32
ML-KEM-768 (Level 3)	3	1,184	2,400	1,088	32
ML-KEM-1024 (Level 5)	4	1,568	3,168	1,568	32

Table 1: ML-KEM parameter sizes by security level.

2.2 ML-DSA (FIPS 204): Digital Signatures

Definition 2.2 (Module-Lattice-Based Digital Signature)**.** *ML-DSA provides EUF-CMA secure digital signatures based on the MLWE and Module-SIS problems. It replaces ECDSA and Ed25519 for transaction signing.*

- ***KeyGen*** $\rightarrow (pk, sk)$*: generates a lattice-based signing key pair.*
- ***Sign***$(sk, m) \rightarrow \sigma$*: produces a signature on message* m*.*

- ***Verify***$(pk, m, \sigma) \to \{0, 1\}$*: verifies the signature.*

ML-DSA is the primary post-quantum signature scheme for on-chain operations: transaction signing, smart contract authorization, and validator attestations.

Level	PK (B)	SK (B)	Sig (B)	Security
ML-DSA-44 (Level 2)	1,312	2,560	2,420	128-bit
ML-DSA-65 (Level 3)	1,952	4,032	3,309	192-bit
ML-DSA-87 (Level 5)	2,592	4,896	4,627	256-bit

Table 2: ML-DSA parameter sizes by security level.

2.3 SLH-DSA (FIPS 205): Hash-Based Signatures

Definition 2.3 (Stateless Hash-Based Digital Signature)**.** *SLH-DSA provides EUF-CMA secure digital signatures based solely on the security of hash functions. Unlike ML-DSA (which relies on lattice assumptions), SLH-DSA's security rests on the one-wayness and collision-resistance of SHA-256 / SHAKE-256. It serves as a conservative fallback: even if lattice assumptions are broken by future advances, SLH-DSA remains secure as long as hash functions are secure.*

SLH-DSA is used for:
- Root of trust signatures (genesis block, validator set changes).
- Long-term archival signatures on regulatory records.
- Fallback signing if ML-DSA is ever weakened.

Variant	PK (B)	SK (B)	Sig (B)	Note
SLH-DSA-128s	32	64	7,856	Small, slow
SLH-DSA-128f	32	64	17,088	Fast, large sig
SLH-DSA-192s	48	96	16,224	Small, slow
SLH-DSA-192f	48	96	35,664	Fast, large sig
SLH-DSA-256s	64	128	29,792	Small, slow
SLH-DSA-256f	64	128	49,856	Fast, large sig

Table 3: SLH-DSA parameter sizes. The "s" variants minimize signature size; the "f" variants minimize signing time.

3 EVM Precompile Integration

All three algorithms are exposed as native EVM precompiles at fixed addresses in the Lux C-Chain and Liquid EVM genesis configuration:

Address	Precompile	Gas (Verify)
`0x0300...0010`	ML-KEM	50,000 (Decapsulate)
`0x0300...0020`	ML-DSA	100,000 (Verify)
`0x0300...0030`	SLH-DSA	150,000 (Verify)
`0x0300...0040`	PQCrypto	10,000 (Parameter query)

Table 4: Post-quantum precompile gas costs.

For comparison, a Solidity implementation of ML-DSA verification consumes >15M gas (exceeding most block gas limits), while the precompile provides the same operation for 100,000 gas—a 150× reduction. The precompile executes compiled Go code with SIMD-optimized NTT (Number Theoretic Transform) operations.

3.1 Input/Output ABI

Each precompile accepts ABI-encoded input:

```
ML-DSA Verify:
  Input:  abi.encode(bytes pk, bytes msg, bytes sig)
  Output: abi.encode(bool valid)

ML-KEM Decapsulate:
  Input:  abi.encode(bytes sk, bytes ct)
  Output: abi.encode(bytes32 sharedSecret)

SLH-DSA Verify:
  Input:  abi.encode(bytes pk, bytes msg, bytes sig)
  Output: abi.encode(bool valid)
```

A Solidity interface contract wraps each precompile for ergonomic use:

```
interface IMLDSA {
    function verify(
        bytes calldata pk,
        bytes calldata msg,
        bytes calldata sig
    ) external view returns (bool);
}
```

4 Hybrid Migration Path

Immediate full migration to post-quantum signatures is impractical: existing wallets, hardware devices, and tooling use ECDSA. The Lux network implements a hybrid signature scheme for the transition period.

Definition 4.1 (Hybrid Signature)**.** *A hybrid signature on message m is a pair $(\sigma_{ECDSA}, \sigma_{ML\text{-}DSA})$ where:*

- $\sigma_{ECDSA} = \mathsf{ECDSA.Sign}(sk_{ECDSA}, m)$
- $\sigma_{ML\text{-}DSA} = \mathsf{ML\text{-}DSA.Sign}(sk_{ML\text{-}DSA}, m)$

The chain accepts the transaction if either *signature verifies.*

Theorem 4.1 (Hybrid Security: Best of Both Worlds)**.** *The hybrid signature scheme is EUF-CMA secure as long as either ECDSA (under the ECDLP assumption) or ML-DSA (under the MLWE/MSIS assumptions) is EUF-CMA secure.*

Proof. Suppose an adversary $\mathcal{A}$ forges a hybrid signature $(\sigma^*_{\text{ECDSA}}, \sigma^*_{\text{ML-DSA}})$ on a new message m^*. For the forgery to be accepted, at least one of σ^*_{ECDSA} or $\sigma^*_{\text{ML-DSA}}$ must verify. If σ^*_{ECDSA} verifies, then $\mathcal{A}$ has forged an ECDSA signature, contradicting the ECDLP assumption. If $\sigma^*_{\text{ML-DSA}}$ verifies, then $\mathcal{A}$ has forged an ML-DSA signature, contradicting the MLWE/MSIS assumptions. Therefore, the forgery probability is at most $\epsilon_{\text{ECDSA}} + \epsilon_{\text{ML-DSA}}$, which is negligible as long as either ϵ_{ECDSA} or $\epsilon_{\text{ML-DSA}}$ is negligible. □

4.1 Migration Phases

1. **Phase 0 (current).** Both classical and PQ precompiles are available. Transactions may use either. Most users continue with ECDSA.
2. **Phase 1 (recommended).** Wallets begin generating hybrid key pairs (one ECDSA + one ML-DSA) and attaching hybrid signatures to transactions. Either signature is accepted.
3. **Phase 2 (transition).** The chain begins requiring ML-DSA signatures on high-value operations (e.g., token issuance, validator registration). ECDSA remains valid for standard transfers.
4. **Phase 3 (post-quantum).** When CRQCs are imminent or operational, the chain deprecates ECDSA-only transactions. Only ML-DSA or hybrid signatures are accepted.

The key insight is that Phase 0 is already active: the precompiles exist at genesis. No hard fork is needed to enable post-quantum cryptography. The migration is purely a wallet-side upgrade.

5 Why Securities Must Act Now

5.1 The 30-Year Argument

A corporate bond issued in 2026 with a 30-year maturity expires in 2056. The private key that signed the issuance transaction, the signatures on every ownership transfer, and the key encapsulation protecting confidential terms—all must remain secure until 2056.

Conservative estimates place CRQC availability at 2035–2045. This means a bond issued today with ECDSA signatures will have its ownership history become forgeable during its lifetime.

5.2 Harvest-Now, Decrypt-Later

Nation-state adversaries are known to record encrypted communications and signed transactions for future quantum decryption. For securities:

- Recorded ECDSA signatures can be forged retroactively once the public key is recovered from the signature.
- Recorded ML-KEM ciphertexts (if not PQ-protected) can be decrypted, revealing confidential deal terms.
- Recorded key exchange transcripts (ECDH) can be replayed to recover session keys.

Post-quantum cryptography deployed today protects against harvest-now attacks, even if CRQCs are decades away.

5.3 Regulatory Imperative

- **NIST.** Post-quantum standards finalized August 2024. Federal agencies must begin migration.
- **NSA CNSA 2.0.** National Security Systems must support post-quantum algorithms by 2025 and exclusively use them by 2033.
- **CISA.** Issued guidance for critical infrastructure to inventory cryptographic dependencies and plan migration.
- **SEC.** Rule 17a-4 record-keeping integrity depends on unforgeable signatures. If ECDSA is broken, the records are compromised.

6 Performance

Benchmarks on Apple M4 Max (Go implementation with SIMD NTT):

Algorithm	KeyGen	Sign/Encap	Verify/Decap
ECDSA (secp256k1)	0.02 ms	0.03 ms	0.05 ms
Ed25519	0.01 ms	0.02 ms	0.04 ms
ML-KEM-768	0.04 ms	0.05 ms	0.04 ms
ML-DSA-65	0.10 ms	0.35 ms	0.12 ms
SLH-DSA-128f	0.01 ms	3.50 ms	0.20 ms
Hybrid (ECDSA + ML-DSA-65)	0.12 ms	0.38 ms	0.17 ms

Table 5: Cryptographic operation latencies.

ML-DSA verification is approximately 2.4× slower than ECDSA verification. For a blockchain that processes 4,000 transactions per second, this adds 480 μs of verification overhead per block of 1,000 transactions—negligible compared to consensus latency.

The larger signature sizes (2,420 bytes for ML-DSA-44 vs. 65 bytes for ECDSA) increase transaction data by approximately 2.3 KB. At 4,000 TPS, this is 9.2 MB/s of additional bandwidth—well within the capacity of modern validator networks.

7 Formal Verification

The post-quantum precompiles are formally verified in the `lux/proofs` repository:

- `Crypto.MLDSA`: correctness and EUF-CMA security model.
- `Crypto.MLKEM`: IND-CCA2 security model, decapsulation correctness.
- `Crypto.SLHDSA`: hash-based security reduction, statelessness.
- `Crypto.Hybrid`: best-of-both-worlds composition theorem.
- `Crypto.Pulsar`: lattice-based threshold signatures for post-quantum MPC.

8 Conclusion

Digital securities infrastructure cannot afford to wait for quantum computers to arrive before deploying quantum-resistant cryptography. The 30-year lifetime of financial instruments, combined with the harvest-now-decrypt-later threat model and regulatory record-keeping requirements, demands that post-quantum cryptography be available today. The Lux network deploys ML-KEM, ML-DSA, and SLH-DSA as native EVM precompiles at genesis, provides a hybrid migration path that preserves backward compatibility, and proves that the hybrid scheme achieves best-of-both-worlds security. The performance overhead is negligible for modern validator hardware.

References

[1] P. Shor. *Algorithms for Quantum Computation: Discrete Logarithms and Factoring*. FOCS 1994.

[2] NIST. *FIPS 203: Module-Lattice-Based Key-Encapsulation Mechanism Standard*. August 2024.

[3] NIST. *FIPS 204: Module-Lattice-Based Digital Signature Standard*. August 2024.

[4] NIST. *FIPS 205: Stateless Hash-Based Digital Signature Standard*. August 2024.

[5] NSA. *CNSA 2.0: Commercial National Security Algorithm Suite 2.0*. September 2022.

[6] D. Bernstein, T. Lange. *Post-Quantum Cryptography.* Nature, 549:188–194, 2017.

[7] M. Mosca. *Cybersecurity in an Era with Quantum Computers: Will We Be Ready?* IEEE Security & Privacy, 16(5):38–41, 2018.

[8] O. Regev. *On Lattices, Learning with Errors, Random Linear Codes, and Cryptography.* STOC 2005.

[9] L. Ducas, T. Prest. *Fast Fourier Orthogonalization.* ISSAC 2016.