LP-305: Quantum Secure Assets

A Post-Quantum Methodology for Protecting Digital Assets on the Lux Network

Zach Kelling

Lux Industries

Original: August 2022 | Revised: March 2026

Abstract

We present the cryptographic architecture underlying Lux Network's post-quantum asset protection layer. The design combines lattice-based threshold cryptography (TFHE over RLWE), post-quantum signature schemes (ML-DSA, Pulsar), fully homomorphic encryption for privacy-preserving computation, and multimodal biometric authentication—providing comprehensive security against adversaries with access to fault-tolerant quantum computers.

This paper consolidates and updates the 2022 "Quantum Secure Assets" internal design document. Where the original described aspirational architecture, this revision maps each component to its production implementation, cites the formal verification where available, and identifies the remaining open problems.

Contents

1 Introduction

1.1 Background

Shor's algorithm [1] renders RSA, ECDSA, and all discrete-logarithm-based signature schemes insecure on a sufficiently powerful quantum computer. NIST's Post-Quantum Cryptography standardization process [2] selected ML-KEM, ML-DSA, and SLH-DSA (FIPS 205, formerly SPHINCS+) as the primary replacements. Lux Network adopted these standards at the consensus layer and extended them with ring-signature privacy (Pulsar) and fully homomorphic encryption (TFHE) for computation over ciphertexts.

1.2 Scope

This paper covers five layers of Lux's post-quantum stack:

1. **Consensus signatures**: BLS-381 + ML-DSA-65 + Pulsar (Quasar protocol, LP-020).
2. **Threshold key management**: TFHE key generation via Shamir LSSS over a 2048-bit safe prime, with Feldman VSS commitments.
3. **Fully homomorphic encryption**: TFHE boolean-gate circuits for encrypted CRDT merge, encrypted search, and private smart contracts.
4. **Non-custodial wallets**: 2-of-3 MPC (CGGMP21 for secp256k1, FROST for Ed25519) with HSM co-signing.
5. **Regulatory disclosure**: viewing keys, threshold disclosure, and ZK attestation anchoring.

1.3 Evolution from the 2022 Design

The 2022 internal document proposed Lamport one-time signatures for vault protection and an OTP-based information-theoretic signing layer. Both were superseded:

- Lamport signatures → ML-DSA-65 (FIPS 204): multi-use, smaller signatures (3.3 KB vs 16 KB per Lamport instance), NIST-standardized.
- Network-layer OTP signing → dropped. The throughput cost of per-transaction OTP generation was incompatible with sub-second finality. ML-DSA provides computational (not information-theoretic) PQ security, which is sufficient given current lattice-hardness estimates [3].

2 Lattice-Based Threshold Cryptography

2.1 Mathematical Foundation

Security rests on the hardness of lattice problems. We define the foundational problems before specializing to RLWE.

Definition 2.1 (Lattice)**.** *A lattice L is the set of all integral combinations of n linearly independent vectors $\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_n \in \mathbb{R}^n$:*

$$L = \left\{ \sum_{i=1}^{n} x_i \mathbf{b}_i : x_i \in \mathbb{Z} \right\}$$

The matrix $B = [\mathbf{b}_1 | \cdots | \mathbf{b}_n]$ is a basis of L.

Definition 2.2 (Shortest Vector Problem (SVP))**.** *Given a lattice L and a norm $\|\cdot\|$, find a non-zero vector $\mathbf{v} \in L$ that minimizes the norm:*

$$\mathbf{v} \in L \setminus \{\mathbf{0}\} \quad \textit{such that} \quad \|\mathbf{v}\| = \lambda_1(L) \triangleq \min_{\mathbf{0} \neq \mathbf{u} \in L} \|\mathbf{u}\|$$

The γ-approximate SVP (γ-SVP) relaxes this to finding $\mathbf{v}$ with $\|\mathbf{v}\| \leq \gamma \cdot \lambda_1(L)$.

Definition 2.3 (Closest Vector Problem (CVP))**.** *Given a lattice L, a norm $\|\cdot\|$, and a target $\mathbf{t} \in \mathbb{R}^n$, find a lattice vector closest to $\mathbf{t}$:*

$$\mathbf{v} \in L \quad \textit{such that} \quad \|\mathbf{v} - \mathbf{t}\| = \min_{\mathbf{u} \in L} \|\mathbf{u} - \mathbf{t}\|$$

Both SVP and CVP are NP-hard under randomized reductions [9]. No polynomial-time quantum algorithm is known for either problem, even approximately with sub-exponential approximation factors.

Definition 2.4 (Learning With Errors (LWE) [5])**.** *For dimension n, modulus q, and error distribution χ over $\mathbb{Z}_q$, the LWE problem is: given a secret $\mathbf{s} \in \mathbb{Z}_q^n$ and samples $(\mathbf{a}_i, b_i = \langle \mathbf{a}_i, \mathbf{s} \rangle + e_i \bmod q)$ where $\mathbf{a}_i \xleftarrow{\$} \mathbb{Z}_q^n$ and $e_i \leftarrow \chi$, recover $\mathbf{s}$.*

Equivalently, given matrix $A \in \mathbb{Z}_q^{m \times n}$ and vector $\mathbf{b} \in \mathbb{Z}_q^m$:

$$\textit{Find } \mathbf{s} \in \mathbb{Z}_q^n \textit{ such that } \|\mathbf{b} - A\mathbf{s}\| < \epsilon$$

where ϵ bounds the error norm. Regev [5] showed that solving LWE is as hard as solving worst-case lattice problems ($\widetilde{O}(n/\alpha)$-approximate SVP) with quantum reductions.

2.1.1 Ring LWE

The Ring LWE (RLWE) variant provides efficiency by working over polynomial rings. Let $R_q = \mathbb{Z}_q[X]/(X^n + 1)$ be a cyclotomic polynomial ring. The decisional RLWE problem asks to distinguish $(a_i, a_i \cdot s + e_i)$ from uniform, where $s \in R_q$ is secret and e_i are drawn from a discrete Gaussian χ_σ.

Definition 2.5 (RLWE Hardness Assumption)**.** *For security parameter λ, ring dimension n, modulus q, and error distribution χ_σ, no probabilistic polynomial-time (or bounded quantum polynomial-time) algorithm can distinguish RLWE samples from uniform with advantage better than* $\mathrm{negl}(\lambda)$.

RLWE reduces to ideal lattice problems in R_q, inheriting the worst-case hardness of SVP over ideal lattices [**?**].

2.2 Implementation: `luxfi/fhe`

Lux's TFHE library (`github.com/luxfi/fhe`) implements:

- **Parameter sets**: PN10QP27 (N=1024, Q=134215681, ∼128-bit classical), PN11QP54 (N=2048, higher precision), and OpenFHE-compatible STD128Q (post-quantum 128-bit).
- **Boolean gates**: AND, OR, XOR, NOT, NAND, NOR, XNOR, MUX, MAJORITY—each with programmable bootstrapping.
- **Integer arithmetic**: FheUint8/16/32/64 with Add, Sub, Mul (schoolbook), comparison chains (Lt, Gt, Eq).
- **Threshold key management**: Shamir secret sharing over the 2048-bit RFC 3526 Group 14 safe prime, with Feldman VSS commitments in the order-q subgroup ($g = 2^2 \bmod p$).
- **Proactive reshare**: additive sub-sharing via Lagrange coefficients—no party materializes the full secret during reshare.

2.3 Threshold FHE Protocol

Key generation follows a t-of-n Shamir scheme. Each party i receives a share $s_i = f(i)$ where f is a random degree-$(t-1)$ polynomial with $f(0) = sk$. Reconstruction uses Lagrange interpolation:

$$sk = \sum_{i \in S} s_i \cdot \lambda_i(0), \qquad \lambda_i(x) = \prod_{j \in S \setminus \{i\}} \frac{x - j}{i - j}$$

Reshare without secret materialization: each old party i picks a random polynomial g_i of degree $(t'-1)$ with constant term $\lambda_i \cdot s_i$. New share $s'_j = \sum_i g_i(j)$. Since $\sum_i g_i(0) = \sum_i \lambda_i \cdot s_i = sk$, the new shares reconstruct the same secret. No single party ever holds sk in cleartext.

3 Fully Homomorphic Encryption

3.1 Homomorphic Property

The defining property of FHE is that it supports both addition and multiplication on ciphertexts. Formally, an FHE scheme $(\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Eval})$ satisfies [6]:

Theorem 3.1 (Homomorphic Correctness)**.** *For any circuit C of depth d, keys $(pk, sk) \leftarrow \mathsf{KeyGen}(1^\lambda)$, and plaintexts x, y:*

$$\mathsf{Dec}(sk, \mathsf{Eval}(pk, +, \mathsf{Enc}(pk, x), \mathsf{Enc}(pk, y))) = x + y \tag{1}$$

$$\mathsf{Dec}(sk, \mathsf{Eval}(pk, \times, \mathsf{Enc}(pk, x), \mathsf{Enc}(pk, y))) = x \cdot y \tag{2}$$

provided the accumulated noise does not exceed the modulus bound.

The noise growth per operation is the fundamental obstacle. For LWE-based schemes [12], multiplication increases noise quadratically. Gentry's bootstrapping technique [6] resets noise by homomorphically evaluating the decryption circuit, enabling unbounded computation depth at the cost of one bootstrapping operation per gate.

TFHE [14] achieves gate-by-gate bootstrapping: each boolean gate evaluation simultaneously computes the gate and refreshes the ciphertext noise in a single operation, avoiding noise accumulation entirely.

3.2 TFHE Gate-by-Gate Bootstrapping

Each boolean gate $G \in \{\text{AND}, \text{OR}, \text{XOR}, \ldots\}$ is evaluated via a single bootstrapping operation that simultaneously computes the gate and refreshes the noise:

$$\text{Bootstrap}(c, G) = \text{BlindRotate}(c) \cdot G_{\text{LUT}}$$

where G_{LUT} encodes the gate's truth table as a test polynomial in R_q.

3.3 Encrypted CRDT Merge

CRDT merge under FHE enables conflict-free replication without decryption. The LWW-Register merge circuit compares encrypted timestamps MSB-to-LSB:

$bGtA \leftarrow \bot$; $eqSoFar \leftarrow \top$
for $i = \text{MSB}$ **downto** 0 **do**
  $bitGt \leftarrow \text{ANDNY}(a_i, b_i)$
  $bitEq \leftarrow \text{XNOR}(a_i, b_i)$
  $bGtA \leftarrow \text{OR}(bGtA, \text{AND}(eqSoFar, bitGt))$
  $eqSoFar \leftarrow \text{AND}(eqSoFar, bitEq)$
end for

8: **for** each bit i **do**
9: $\quad result_i \leftarrow \mathrm{MUX}(bGtA, b_i, a_i)$
10: **end for**

Gate count: $O(\text{bitsTS})$ comparisons + $O(\text{bitsVal} + \text{bitsTS})$ MUX selections. At PN10QP27 with 8-bit timestamps and 8-bit values: ∼40 bootstraps ≈ 4 seconds.

3.4 Performance Budget

Operation	Gates	Time (PN10QP27)	Ciphertext
LWW Merge (8+8 bit)	40	4 s	136 KB
GCounter Max (32 bit/node)	128	13 s	544 KB
OR-Set Merge	0 (tag union)	<1 ms	tags only

4 Non-Custodial Wallet Architecture

Every user receives a 2-of-3 MPC wallet on the Lux EVM:

1. **User device** — key share derived from WebAuthn PRF.
2. **ATS co-signer** — auto-signs after compliance checks.
3. **Backup (cold)** — Shamir recovery via `luxfi/age` (X25519 + ML-KEM-768 X-Wing hybrid).

Protocols: CGGMP21 [4] for ECDSA (secp256k1), FROST for EdDSA (Ed25519). HSM co-signing for settlement intents (AWS KMS, GCP, Zymbit).

5 Regulatory Disclosure

Three composable modes, implemented in `Disclosure.sol`:

1. **Viewing keys** (Zcash-style): owner registers a per-viewer wrapped decryption key. Revocable on-chain; actual key revocation requires off-chain rekey.
2. **Threshold disclosure** (M-of-N, regulator is one party): uses the existing `lsss.go` Shamir infrastructure. On-chain share submission with per-party dedup. Combination happens off-chain.
3. **ZK attestation anchor**: accepts opaque proof bytes with a `claimType` tag. On-chain verifier registry (owner-gated) routes to Groth16/PlonK verifier contracts when deployed.

6 Open Problems

1. **Deterministic keygen from seed**: resolved in `luxfi/lattice` v8 (seeded-PRNG `KeyGenerator`). Consensus validators now derive identical FHE keys from a shared seed, enabling threshold reshare without trusted setup.
2. **FHE throughput**: ∼0.24 ops/sec at production parameters (PN10QP27). GPU acceleration via NTT (LP-203) is the primary mitigation path for high-frequency use cases.
3. **Biometric + FHE fusion**: encrypting biometric templates under TFHE so servers can *match* without seeing the template remains an active engineering direction.

7 Conclusion

The design combines lattice-based threshold cryptography, TFHE-backed homomorphic CRDT merge, non-custodial MPC wallets (CGGMP21 + FROST), and three-mode regulatory disclosure. Every component is anchored in Lux production code — `luxfi/fhe`, `luxfi/lattice`, `luxfi/mpc`, `luxfi/age`, `luxfi/standard` — with formal verification artefacts in `luxfi/proofs`.

References

[1] P. W. Shor, "Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer," *SIAM Review*, vol. 41, no. 2, pp. 303–332, 1999.

[2] NIST, "Post-Quantum Cryptography Standardization," https://csrc.nist.gov/projects/post-quantum-cryptography, 2024.

[3] NIST, "Status Report on the Third Round of the NIST Post-Quantum Cryptography Standardization Process," NISTIR 8413, 2022.

[4] R. Canetti, R. Gennaro, S. Goldfeder, N. Makriyannis, and U. Peled, "UC Non-Interactive, Proactive, Threshold ECDSA with Identifiable Aborts," IACR ePrint 2021/060, 2021.

[5] O. Regev, "On lattices, learning with errors, random linear codes, and cryptography," *STOC*, pp. 84–93, 2005.

[6] C. Gentry, "A fully homomorphic encryption scheme," Ph.D. dissertation, Stanford University, 2009.

[7] L. Lamport, "Constructing digital signatures from a one-way function," Technical Report SRI-CSL-98, 1979.

[8] T. Kivinen and M. Kojo, "More Modular Exponential (MODP) Diffie-Hellman groups for Internet Key Exchange (IKE)," RFC 3526, 2003.

[9] M. Ajtai, "Generating hard instances of lattice problems (extended abstract)," *STOC*, pp. 99–108, 1996.

[10] D. Micciancio and O. Regev, "Lattice-based cryptography," in *Post-Quantum Cryptography*, Springer, pp. 147–191, 2009.

[11] V. Lyubashevsky, C. Peikert, and O. Regev, "On ideal lattices and learning with errors over rings," *Journal of the ACM*, vol. 60, no. 6, 2013.

[12] C. Gentry, A. Sahai, and B. Waters, "Homomorphic encryption from learning with errors: Conceptually-simpler, asymptotically-faster, attribute-based," *CRYPTO*, 2013.

[13] D. J. Bernstein, A. Hülsing, S. Kölbl, R. Niederhagen, J. Rijneveld, and P. Schwabe, "The SLH-DSA (FIPS 205, formerly SPHINCS+) signature framework," *ACM CCS*, 2019.

[14] I. Chillotti, N. Gama, M. Georgieva, and M. Izabachène, "Faster Fully Homomorphic Encryption: Bootstrapping in less than 0.1 Seconds," *ASIACRYPT*, 2016.