Lux MPTC Security Analysis

Zach Kelling

Lux Industries

2026-03-03

Abstract

We present Lux, a hybrid post-quantum consensus architecture for DAG-native chains. Lux separates fast operational finality from durable post-quantum finality. Validators emit Photons over a Nebula DAG substrate; classical BLS aggregates form Beams for low-latency confirmation, while ML-DSA attestations and Pulsar lattice-threshold Pulses provide post-quantum finality over Nebula roots. Prism binds all certificate lanes to the same transcript, and Quasar reaches Horizon finality when the bound lanes verify.

This document. The security-analysis summary for the NIST MPTC submission. The full proofs live in the canonical proof bucket (`proofs/pulsar/`, `proofs/lss/`, `proofs/quasar/`, `proofs/definitions/`); this document summarizes the theorems and points at the bucket files. Lean / TLA+ / Tamarin mechanizations are also cited.

1 Pulsar SUF-CMA

Theorem 1 (Pulsar SUF-CMA under MLWE/MSIS)**.** *Let $\mathcal{A}$ be a probabilistic polynomial-time adversary against Pulsar with at most q_S signing queries and at most $t-1$ corrupted parties. Then*

$$\mathrm{Adv}^{\mathsf{SUF\text{-}CMA}}_{\mathsf{Pulsar}}(\mathcal{A}) \;\le\; q_S \cdot \mathrm{Adv}^{\mathsf{MLWE}}_{q,\sigma_E}(\mathcal{B}_1) + \mathrm{Adv}^{\mathsf{MSIS}}_{q,[\mathbf{A}|-c\tilde{\mathbf{b}}],2B}(\mathcal{B}_2) + \mathrm{negl}(\lambda).$$

Full proof: `proofs/pulsar/unforgeability.tex`; concrete parameters in `quasar-cert-soundness.tex` App. E give classical $\geq 2^{142}$ and quantum $\geq 2^{130}$ via BDGL sieving + Grover. Lean: `Crypto.Pulsar.corona_pq` [4].

2 Reshare preserves master secret

Theorem 2 (Reshare preserves master secret)**.** *The new shares $\{\mathsf{Share}'_j = G(\beta_j)\}$ are valid $(t_{\mathrm{new}}, |V_{\mathrm{new}}|)$-Shamir shares of the same master secret $\mathbf{s}$, where $G(X) = \sum_{i\in Q} \lambda_i^Q \cdot g_i(X)$. Persistent group public key $(\mathbf{A}, \tilde{\mathbf{b}})$ unchanged.*

Full proof: `proofs/pulsar/reshare-preservation.tex`. Lean: `Crypto.Threshold.Reshare.reshare_sam` `reshare_preserves_public_key`.

3 Activation circuit-breaker safety

Theorem 3 (Activation cert is necessary and sufficient for safe transition)**.** *A successful activation cert verify under unchanged GroupKey implies (i) the new shares recombine to the same master secret, (ii) the new committee can produce verifying signatures, (iii) the KeyEraID is unchanged. Conversely, failure of (i), (ii), or (iii) implies activation cert verify fails with overwhelming probability.*

Full proof: `proofs/pulsar/activation-safety.tex`; companion-cited as Theorem 14 (`thm:activation-so` of [2].

Lemma 4 (Failure-mode containment)**.** *If activation cert verify fails, the chain remains at the prior state. The candidate state is discarded; the LSS RollbackManager retains the prior snapshot. This is a structural property of the chain's state-machine acceptance rule, not a cryptographic assumption.*

Reference: `proofs/pulsar/activation-safety.tex` Lemma `lem:activation-fail-safe`.

4 Hash-suite separation

Theorem 5 (Cross-suite signatures do not collide)**.** *A Pulsar signature under one hash suite (Pulsar-SHA3 or Pulsar-BLAKE3) does not verify under the other, with overwhelming probability over the verifier's hash function.*

Full proof: `proofs/pulsar/hash-suite-separation.tex`. The HashSuiteID is bound into every transcript via TupleHash personalization; cross-suite forgery would require a collision on the personalized TupleHash.

5 LSS lifecycle invariants

Lemma 6 (Generation monotonicity)**.** *g increases monotonically by exactly* 1 *per* DynamicReshare *or* Refresh*. On* Rollback*, $g \mapsto g_{\text{current}} + 1$ with $\mathsf{rb} = g_{\text{current}}$.*

Lemma 7 (KeyEraID stability)**.** *η is unchanged by* Refresh*,* Reshare*, or* Rollback*. Increments only on* Reanchor*.*

Lemma 8 (GroupKey byte-identity)**.** *GroupKey pointer is byte-identical across every* Refresh *and* Reshare *within an era.*

Lemma 9 (Rollback lineage reconstructibility)**.** *The forward lineage of the chain is reconstructible from the snapshot history.*

Full proofs: `proofs/lss/lifecycle-safety.tex`.

6 Rollback safety

Theorem 10 (Rollback never crosses an era boundary)**.** $\mathsf{Rollback}(g_{\text{target}})$ *succeeds iff g_{target} is in the current era's snapshot history. Rollback to a snapshot with $\eta_{\text{snapshot}} \neq \eta_{\text{current}}$ is refused.*

Theorem 11 (Rollback preserves liveness under failed activation)**.** *If state_{g+1}'s activation cert fails verification and the chain calls $\mathsf{Rollback}(g)$, the chain continues to produce signatures under state_g's shares.*

Full proofs: `proofs/lss/rollback-safety.tex`.

7 Quasar Horizon soundness (consensus composition)

Theorem 12 (Horizon-final implies all lanes verify the same transcript)**.** *A Prism-accepted Horizon certificate cannot exist for two distinct* NebulaRoot*s at the same* (epoch, round) *except with negligible probability under (i) BLS12-381 co-CDH, (ii) MLWE/MSIS, and (iii) FIPS 204 ML-DSA EUF-CMA. The lanes are independent; quantum break of one does not break the others.*

Full proof: `proofs/quasar/horizon-soundness.tex`; companion-cited from [2] Theorem 7.5 (classical soundness), Theorem 7.6 (parallel-lane liveness), Theorem 7.7 (PQ safety). Lean: `Consensus/Quasar.lean`: `triple_requires_all`, `quantum_safe_triple`, `independent_pq_from_classical`, `cert_intersection`, `unique_per_round`.

8 L2 co-validation security inheritance

Theorem 13 (L2 inherits from primary)**.** *An L2's safety bound is at least the minimum of the primary's safety bound and the L2's own validator-set safety bound. Forging conflicting L2 finality requires (i) compromising enough of the L2's validators, AND (ii) either compromising the primary's validators or breaking transcript binding.*

Full proof: `proofs/quasar/l1-l2-covalidation.tex`.

9 Warp 2.0 envelope unforgeability

Theorem 14 (Warp 2.0 envelope unforgeability)**.** *A successful* Verify_{V2} *implies any forgery on the same* (sourceChainID, sourceNebulaRoot, sourceKeyEraID, sourceGeneration) *tuple with a different* payload *requires breaking either co-CDH (Beam) or MLWE/MSIS (Pulse).*

Full proof: `proofs/quasar/warp-pq-soundness.tex`.

10 Lane independence under quantum advice

Lemma 15 (Lane independence)**.** *A polynomial-time adversary* $\mathcal{A}$ *with quantum oracle access to BLS12-381's discrete log gains no MLWE/MSIS-relevant computational advantage; the Pulse lane remains unforgeable. Symmetrically, an MLWE-oracle-equipped* $\mathcal{A}$ *enjoys no advantage against BLS.*

Full argument: `proofs/pq-finality-no-bls.tex`; [1].

11 Honest classification of the Groth16 wrapper

Remark 1 (Groth16 wrapper does not contribute PQ security)**.** *The Z-Chain Groth16 rollup proof of ML-DSA cert-set verification is a* classical succinct proof of post-quantum signature verification. *Pairing-based SNARKs over BLS12-381 break under Shor. Horizon's PQ soundness flows from ML-DSA signatures and Pulsar Pulse alone; the Groth16 proof is a compatibility / compression artifact.*

Reference: Remark `rem:groth16-wrapper` of `proofs/quasar/horizon-soundness.tex`; Remark `rem:groth16-not-pq` of `proofs/definitions/finality-definitions.tex`; LP-105 [5] §"Proof-lane classification".

The classification is enforced in code: `warp/pulsar/IsPQRootOfTrust` returns false for Groth16-wrapped lanes (`warp/pulsar/groth16_classification_test.go`).

12 Mechanization summary

Theorem	Mechanization
Pulsar SUF-CMA	Lean: `Crypto.Pulsar.corona_pq_unforgeability`
Reshare preserves master secret	Lean: `Crypto.Threshold.Reshare.reshare_same_secret`
GroupKey byte-identity	Lean: `Crypto.Threshold.Reshare.reshare_preserves_public_key`
Triple consensus composition	Lean: `Consensus/Quasar.lean`
BFT safety / liveness / finality	Lean: `Consensus/{Safety,Liveness,Finality,BFT}.lean`
Quasar protocol model	TLA+: `tla/Quasar.tla` (model-checked)
Quasar protocol handshake	Tamarin: `tamarin/QuasarConsensus.spthy`
FROST threshold (Lens)	Tamarin: `tamarin/FROSTThreshold.spthy`
PQ encryption (Lumen)	Tamarin: `tamarin/pq_encryption.spthy`
Property-based testing	Go: `proofs/property/`

The mechanizations are not exhaustive (e.g., a UC treatment is open work); they cover the most security-critical theorems. The full proof bucket at `github.com/luxfi/proofs/` is the source of truth.

13 Adversary model

The adversary is computationally bounded with possibly quantum-aided lower-bound queries. It adaptively corrupts up to f validators with $f < n/3$. It controls the network (delay, drop, reorder) but not the chain's state-machine acceptance rule. Its quantum and classical oracles are independent: a quantum DL oracle gives no MLWE/MSIS advantage, and vice versa.

14 Side-channel posture

Constant-time review documented separately in the side-channel-considerations document of this submission. Summary: the lattice math (Pulsar) and curve math (Lens) have been reviewed; GPU paths are research, not production; the lattigo deserializer hardening (Section "Fuzz-Discovered Vulnerability" of [3]) is in production.

15 Audit status

The Lux research audit ([2] App. + `papers/AUDIT.pdf`) covers the consensus engine and threshold-cryptography stack. The audit cites the proof bucket; mechanizations and the production-stack tests are the audit-supporting evidence. The audit is internal to Lux; an external third-party audit is in the production-research roadmap.

References

[1] Zach Kelling. Post-quantum finality for lux quasar. Lux Industries; `lux/proofs/pq-finality-no-bls.tex`, 2026.

[2] Zach Kelling. QuasarCert soundness: Canonical formal specification. Lux Industries; `lux/proofs/quasar-cert-soundness.tex`, 2026.

[3] Zach Kelling. Warp 2.0: Pulse-backed cross-chain source finality. Lux Industries, Inc.; `papers/lux-warp-v2/`, 2026.

[4] Lux Industries. Lean4 cryptographic proof tree. https://github.com/luxfi/proofs/tree/main/lean/Crypto, 2026.

[5] Lux Industries. LP-105: The lux finality stack. Lux Improvement Proposal, 2026.