LP-020: Quasar 3.0 Consensus

Three-Lane Post-Quantum Certificates

over the P/Q/Z/A/B/M/F Chain Pipeline

Zach Kelling

Lux Industries

Spec freeze: 15 December 2025 | Final review: 20 December 2025 | Activation: 25 December 2025

Abstract

Lux is not merely adding post-quantum signatures to a chain; it defines a hybrid finality architecture for DAG-native consensus, with protocol-agnostic threshold lifecycle, post-quantum threshold sealing, and cross-chain propagation of Horizon finality.

We present Quasar 3.0, the post-quantum Byzantine fault tolerant consensus protocol activated on the Lux primary network on 25 December 2025. Quasar 3.0 finalizes a block by producing a single *QuasarCert* that aggregates three independent verification *lanes*: a BLS12-381 aggregate signature (classical fast path, discrete logarithm), a Corona two-round Module-LWE threshold signature (Q-Chain ceremony root), and a Z-Chain Groth16 proof that rolls up N ML-DSA-65 validator signatures into a single 192-byte SNARK plus a 32-byte public-inputs hash. The third lane is *not* a raw per-validator ML-DSA list; it is a Z-Chain proof artifact, allowing the per-block on-chain cost of post-quantum identity binding to drop from $N \times 3{,}309$ bytes to $192 + 32$ bytes regardless of validator count. Every QuasarCert is bound to a *certificate subject* that hashes the chain id, epoch, round, mode, and the seven chain-specific roots ($P_{\text{val}} \| Q_{\text{cer}} \| Z_{\text{vk}} \| A_{\text{att}} \| B_{\text{br}} \| M_{\text{mpc}} \| F_{\text{fhe}}$) together with the parent block hash, parent state root, parent execution root, gas limit, and base fee, making cross-epoch and cross-chain replay structurally impossible. The three lanes are verified independently by the QuasarGPU substrate (LP-132): BLS aggregate verify ($\sim$875 μs for n=100 on CPU), Corona batch verify ($O(1)$ amortized after DKG), and a single Groth16 pairing check (200–500 μs GPU batched). On a 21-validator mainnet board, full three-lane GPU verification runs in 0.5–1.5 ms; the network targets a 500 ms block time and sustained $\geq$100K TPS per chain. An adversary must simultaneously break the discrete logarithm on BLS12-381, Module-LWE for Corona, and Module-LWE/MSIS for ML-DSA to forge a QuasarCert; breaking any one or any two leaves the remaining lane intact.

Contents

1 Introduction

Shor's algorithm [1] breaks the elliptic-curve discrete logarithm problem in polynomial time on a cryptographically relevant quantum computer (CRQC). Every BLS, ECDSA, and Schnorr-based blockchain is therefore vulnerable to a harvest-now-decrypt-later attack: an adversary records signed blocks today and forges certificates once a CRQC becomes available. NIST estimates this threshold at 2030–2035 [3].

The standard response—swap a classical signature for a post-quantum one—is insufficient for two reasons. First, no single post-quantum scheme has survived the cryptanalytic decade that broke each successive "next big thing" in classical cryptography; relying on a single post-quantum assumption swaps one single point of failure for another. Second, the natural post-quantum signatures are large: ML-DSA-65 (FIPS 204) signatures are 3,309 bytes versus 48 bytes for an aggregated BLS signature, so a straightforward per-validator ML-DSA scheme increases header size by two orders of magnitude and degrades network throughput.

Quasar 3.0, activated on the Lux primary network on 25 December 2025, resolves both problems in a single artifact. Each finalized block carries a *QuasarCert* composed of three independent verification lanes:

Lane 1: BLS — a 48-byte BLS12-381 aggregate signature on $\mathbb{G}_1$, providing the classical fast path and rapid catch-up verification.

Lane 2: Corona — a two-round Module-LWE threshold signature whose key material is rooted on Q-Chain via an epoch-bound DKG ceremony [5].

Lane 3: MLDSAGroth16 — a Z-Chain Groth16 rollup proof that attests to N validator ML-DSA-65 signatures, encoded as a 192-byte SNARK plus a 32-byte public-inputs hash. *This is not a raw per-validator ML-DSA list.* The Z-Chain replaces the $N \times 3{,}309$ byte attestation with one pairing check.

The three lanes rest on three algebraically independent hardness assumptions: discrete logarithm on BLS12-381 (broken by Shor), Module-LWE for both Corona and ML-DSA (with disjoint parameters), and Module-SIS underlying ML-DSA's commitment relation. No known reduction connects the hardness of one lane to another. An adversary must simultaneously break all three to forge a QuasarCert; any compromise that leaves at least one lane intact preserves safety. The third lane is the *accountability* lane: it proves which validators signed and compresses N signatures to a constant on-chain footprint. It is complementary to — not a substitute for — a genuine *threshold* ML-DSA, which Lux achieves via **Pulsar** (LP-4450 [16]), a Module-LWE threshold whose combined output is byte-equal to FIPS 204 ML-DSA-65/87 through its boundary-cleared nonce / carry-elimination (BCC/CEF) construction (Section 12). The Z-Chain Groth16 rollup gives the same on-chain footprint as a threshold scheme without altering the underlying primitive, while Pulsar provides the threshold seal itself (subject to the honest review-ready status of Section 12.1).

1.1 The P/Q/Z/A/B/M/F Chain Pipeline

QuasarCert generation is distributed across seven specialized chains, each contributing a chain-specific Merkle root that is bound into the certificate subject (Section 5, Section 6.2):

- **P-Chain** — validator and stake authority; emits pchain_validator_root.
- **Q-Chain** — Corona consensus DKG; emits qchain_ceremony_root.
- **Z-Chain** — Groth16 verifying-key registry and rollup; emits zchain_vk_root.
- **A-Chain** — unified attestation chain; emits achain_attestation_root.
- **B-Chain** — bridge messaging chain; emits bchain_bridge_root.
- **M-Chain** — MPC ceremonies (CGGMP21, FROST, Corona-general); emits the mchain_mpc_root.

- **F-Chain** — FHE compute (TFHE, encrypted EVM); emits the fchain_fhe_root.

M-Chain and F-Chain share an internal substrate library (`lux/chains/thresholdvm`) but are operationally orthogonal. There is *no* monolithic "T-Chain"; MPC and FHE are independent chains with distinct validator sets, ceremonies, and failure domains.

1.2 Replay Protection

Every QuasarCert is bound to a *certificate subject*:

$$\mathsf{cert\_subject} = \mathsf{keccak}\big(\mathsf{chain\_id} \| \mathsf{epoch} \| \mathsf{round} \| \mathsf{mode} \| P_{\mathrm{val}} \| Q_{\mathrm{cer}} \| Z_{\mathrm{vk}} \| A_{\mathrm{att}} \| B_{\mathrm{br}} \| M_{\mathrm{mpc}} \| F_{\mathrm{fhe}} \| h_{\mathrm{parent}} \| \mathsf{state\_root} \| \mathsf{e}$$

All three lanes sign this exact 32-byte digest. Any change to validator set, epoch, round, mode, parent state, or execution context alters the subject, so a certificate cannot be replayed across chains, epochs, rounds, or forks. We prove this is structurally complete in Section 6.2.

1.3 GPU-Native Verification

The three lanes are verified independently. BLS aggregate verification (n=100) executes in $\sim$875 μs of CPU; Corona batch verify is $O(1)$ amortized after DKG; the Groth16 pairing check runs in 200–500 μs when batched on a GPU. On the 21-validator mainnet board, the full three-lane verifier completes in 0.5–1.5 ms, well inside the 500 ms target block time. The QuasarGPU adapter (LP-132) schedules these lanes onto the same kernel stream that drives EVM execution, so consensus verification consumes no incremental host-CPU budget.

1.4 Contributions

1. A three-lane finality certificate (QuasarCert) whose unforgeability reduces simultaneously to discrete logarithm on BLS12-381, Module-LWE for Corona, and MLWE/MSIS for ML-DSA, with the Z-Chain Groth16 lane bringing the third lane on-chain in 192+32 bytes regardless of N.

2. A formal certificate-subject construction binding seven chain roots, mode, and execution context, ruling out cross-chain and cross-epoch replay structurally (Theorem 6.1).

3. Theorems for safety, liveness, single-compromise insufficiency, QuasarCert soundness, parallel liveness, and post-quantum safety, with machine-checked proofs in Lean 4 and Tamarin and a companion proof sketch [15].

4. Production benchmarks for the 25 December 2025 release: 0.5–1.5 ms per-cert verify on GPU at n=21, 500 ms block time, and a sustained throughput target of $\geq$100K TPS per chain.

2 System Model

Definition 2.1 (Network)**.** *A set $\mathcal{V}$ of n validators, of which at most $f < n/3$ are Byzantine. Communication is point-to-point authenticated. The network is partially synchronous: there exists an unknown Global Stabilization Time (GST) after which all messages between correct validators are delivered within a known bound Δ.*

Definition 2.2 (Cryptographic Configuration)**.** *Each validator $v_i \in \mathcal{V}$ holds:*

- *A BLS12-381 secret key sk_i^{BLS} with public key $pk_i^{\mathrm{BLS}} \in \mathbb{G}_2$ and a proof-of-possession (PoP) registered on P-Chain.*

- *A share sk_i^{RT} of the Corona threshold key with combined public key pk^{RT} rooted on Q-Chain (threshold $t = \lfloor 2n/3 \rfloor + 1$).*
- *An ML-DSA-65 secret key sk_i^{ML} with public key pk_i^{ML} registered on P-Chain.*

Definition 2.3 (Chain Roots)**.** *At each consensus round, the canonical seven chain roots are observable by every validator:*

$$\boldsymbol{\rho} \stackrel{def}{=} \big(P_{val},\, Q_{cer},\, Z_{vk},\, A_{att},\, B_{br},\, M_{mpc},\, F_{fhe}\big)$$

where:

- P_{val} = pchain_validator_root *binds the active validator set, stake, and PoP table.*
- Q_{cer} = qchain_ceremony_root *binds the latest Corona DKG transcript and combined public key.*
- Z_{vk} = zchain_vk_root *binds the Groth16 verifying-key registry for the ML-DSA verification circuit and the Z-Chain rollup state.*
- A_{att} = achain_attestation_root *binds unified attestations (genesis records, oracle commits, off-chain proofs).*
- B_{br} = bchain_bridge_root *binds the bridge message log.*
- M_{mpc} = mchain_mpc_root *binds the MPC ceremony log (CGGMP21, FROST, Corona-general).*
- F_{fhe} = fchain_fhe_root *binds the FHE compute log (TFHE, encrypted EVM).*

Definition 2.4 (Certificate Subject)**.** *For chain id c, epoch e, round r, mode μ, parent block hash h_{parent}, parent state root s, parent execution root x, gas limit g, and base fee b:*

$$\mathsf{cert\_subject} \;=\; \mathsf{keccak}_{256}\big(c \,\|\, e \,\|\, r \,\|\, \mu \,\|\, P_{val} \,\|\, Q_{cer} \,\|\, Z_{vk} \,\|\, A_{att} \,\|\, B_{br} \,\|\, M_{mpc} \,\|\, F_{fhe} \,\|\, h_{parent} \,\|\, s \,\|\, x \,\|\, g \,\|\, b\big).$$

The Quasar 3.0 protocol requires all three lanes to sign exactly this 32-byte digest *for a given block.*

Definition 2.5 (Consensus Modes)**.** *Quasar 3.0 supports four operational modes selectable at network genesis or per-chain:*

1. ***BLS-only*** *($\mu = 0$): classical aggregate signatures only; used for non-financial chains where post-quantum guarantees are not required.*
2. ***BLS + Corona*** *($\mu = 1$): dual lane; Q-Chain ceremony provides post-quantum threshold finality without ML-DSA identity binding.*
3. ***BLS + MLDSAGroth16*** *($\mu = 2$): dual lane; Z-Chain rollup provides post-quantum identity binding without a separate Corona ceremony.*
4. ***Quantum*** *($\mu = 3$): all three lanes (BLS + Corona + MLDSAGroth16).* This is the mode activated on the Lux primary network on 25 December 2025 and analyzed in this paper.

The mode byte μ enters `cert_subject`, so a Mode-3 certificate cannot be replayed as a Mode-1 certificate even if the lower-mode quorum is also satisfied.

The reference implementation entry point is `TripleSignRound1` in `consensus/protocol/quasar/quasar.go` which dispatches BLS, Corona, and ML-DSA signing concurrently via goroutines under the QuasarGPU adapter (LP-132 [18]). At the wire-policy layer (`consensus/pkg/wire/policies.go`) the currently implemented post-quantum policy is `PolicyQuantum` (BLS + Corona, with Corona toggleable via `NewQuantumPolicyWithOptions`); modes 0 (BLS-ONLY) and 2 (BLS+MLDSAGROTH16) above are specified by this LP for future activation but are not selected by a distinct wire policy in the current code.

3 Protocol

Quasar 3.0 inherits the Quasar consensus family of staged probabilistic-DAG protocols. Two engine variants share the family:

- **Nova** (linear chain): Photon → Wave → Focus → Ray.
- **Nebula** (DAG): Flare → Horizon → Field, with the same QuasarCert at the finalization step.

Both variants finalize a block by producing the same QuasarCert artifact (Section 6). What differs is the upstream voting structure—linear in Nova, DAG in Nebula. We describe Nova here and reference [22, 23] for the DAG case.

3.1 Phase 1: Photon (Proposal Emission)

The leader for height h (round-robin over the validator set, rotated each epoch) constructs a block B_h containing:

- The parent block hash $h_{\text{parent}} = H(B_{h-1})$.
- A batch of transactions from the mempool.
- The leader's BLS signature over B_h.
- A timestamp τ_h and the seven chain roots $\boldsymbol{\rho}$ (Definition 2.3) observed at τ_h.

The leader broadcasts $\langle \textsc{Propose}, B_h, \boldsymbol{\rho} \rangle$ to all validators. Photon emission is fire-and-forget: the leader does not wait for acknowledgments.

3.2 Phase 2: Wave (Fast Probabilistic Consensus)

Upon receiving a valid proposal, each validator v_i enters the Wave phase. Wave implements Fast Probabilistic Consensus (FPC) with phase-dependent thresholds.

Definition 3.1 (FPC Threshold Selection)**.** *For round r with sample size k, the threshold α_r is:*

$$\alpha_r = \lceil \theta_r \cdot k \rceil , \qquad \theta_r = \theta_{\min} + (\theta_{\max} - \theta_{\min}) \cdot \frac{\text{SHA256}(\mathit{seed} \,\|\, r) \bmod 2^{16}}{2^{16}}.$$

Default: $\theta_{\min} = 0.5$, $\theta_{\max} = 0.8$.

3.3 Phase 3: Focus (Certificate Subject Lock)

Once $\mathit{pref}_i = \textsc{Accept}$ has stabilized for β consecutive rounds, v_i enters Focus and computes the certificate subject of Definition 2.4, binding the seven chain roots $\boldsymbol{\rho}$, mode μ, parent block hash, parent state root, parent execution root, gas limit, and base fee. Focus is the boundary between probabilistic vote propagation (Wave) and cryptographic finalization (Ray). At this point all three signing lanes share a single 32-byte digest.

3.4 Phase 4: Ray (Three-Lane Signing)

In Ray, v_i produces three signatures over the same $\mathsf{cert\_subject}$ in parallel:

1. **BLS**: $\sigma_i^{\text{BLS}} = sk_i^{\text{BLS}} \cdot H_{\mathbb{G}_1}(\mathsf{cert\_subject})$. 96-byte (uncompressed) or 48-byte (compressed) $\mathbb{G}_1$ signature.

2. **Corona**: contribute share σ_i^{RT} to Q-Chain's two-round threshold protocol on $\mathsf{cert\_subject}$ using sk_i^{RT}. The Q-Chain combiner produces σ^{RT}.

3. **ML-DSA-65**: $\sigma_i^{\text{ML}} = \mathsf{MLDSA.Sign}(sk_i^{\text{ML}}, \mathsf{cert\_subject})$. 3,309-byte signature, transmitted only to Z-Chain (not gossiped network-wide).

Algorithm 1 Wave: FPC voting round

Require: Validator v_i, block B_h, round r, sample size k

$S \leftarrow \text{SamplePeers}(k)$
$\alpha_r \leftarrow \lceil \text{FPCThreshold}(r, k) \rceil$
$votes \leftarrow 0$
for $v_j \in S$ **do**
    $votes \leftarrow votes + \text{QueryPreference}(v_j, B_h)$
end for
if $votes \geq \alpha_r$ **then**
    $pref_i \leftarrow \textsc{Accept}$
else
    $pref_i \leftarrow \textsc{Reject}$
end if
if $pref_i = prevPref_i$ **then**
    $confidence_i \leftarrow confidence_i + 1$
else
    $confidence_i \leftarrow 0$
end if
if $confidence_i \geq \beta$ **then**
    enter Focus with $pref_i$
end if

The validator broadcasts $\langle \textsc{QuasarVote}, i, \sigma_i^{\text{BLS}}, \sigma_i^{\text{RT}} \rangle$ on the consensus plane and submits σ_i^{ML} to Z-Chain. Z-Chain proves the batch (Section 4.3) and emits the Groth16 artifact (π_{ZK}, h_x) where π_{ZK} is the 192-byte SNARK and h_x is the 32-byte public-inputs hash.

3.5 Finalization

A block proposer that observes (a) an aggregate $\sigma_{\text{agg}}^{\text{BLS}}$ over $\geq 2f + 1$ BLS shares, (b) a valid Corona σ^{RT} from Q-Chain, and (c) a valid (π_{ZK}, h_x) from Z-Chain, constructs the QuasarCert $\mathcal{C}_h = (\sigma_{\text{agg}}^{\text{BLS}}, \sigma^{\text{RT}}, \pi_{\text{ZK}}, h_x)$ and broadcasts $\langle \textsc{Finalize}, B_h, \mathcal{C}_h \rangle$. Any validator verifies $\mathcal{C}_h$ by checking the three lanes against cert_subject as in Definition 6.2.

3.6 Epoch Rotation

Quasar 3.0 groups blocks into epochs of fixed length E. At epoch boundaries:

- Validator sets may change (additions, removals, stake changes); P_{val} updates.
- Q-Chain runs a new Corona DKG; Q_{cer} updates and the previous threshold key is retired.
- Z-Chain may rotate the verifying key (for example, when adding validators changes the circuit's public-input arity); Z_{vk} updates.
- Epoch certificates summarize the finalized chain state.
- Epochs older than 6 are pruned from active memory; only epoch certificates are retained.

The 6-epoch retention window bounds active memory to $O(6 \cdot E \cdot n)$ regardless of total chain length.

4 Three-Lane Certificate Architecture

A QuasarCert is the conjunction of three independent verification *lanes*. Each lane has its own primitive, its own hardness assumption, and its own verifier. None of the three depends on the others to verify, and the three are scheduled and executed in parallel both during signing

(validator-side) and during verification (light-client and peer-side). This section formalizes each lane.

4.1 Lane 1: BLS12-381 Aggregate Signatures

BLS12-381 [4] operates over the pairing-friendly curve $E(\mathbb{F}_p)$ with embedding degree 12 and 128-bit classical security.

- **Key generation**: $sk \xleftarrow{\$} \mathbb{Z}_r$; $pk = sk \cdot G_2$, with proof-of-possession.
- **Signing**: $\sigma = sk \cdot H_{\mathbb{G}_1}(\mathsf{cert\_subject})$.
- **Aggregation**: $\sigma_{\text{agg}} = \sum_{i \in Q} \sigma_i$. Aggregated size is 48 bytes regardless of $|Q|$.
- **Verification**: $e(G_2, \sigma_{\text{agg}}) = e(apk, H_{\mathbb{G}_1}(\mathsf{cert\_subject}))$ where $apk = \sum_{i \in Q} pk_i$.

BLS provides the classical fast path. Verification cost is $O(1)$ in $|Q|$ and dominated by two pairings; an aggregate-of-aggregates check at n=100 runs in ∼875 μs on commodity x86 CPU (Section 8). This lane is broken by Shor; the other two lanes compensate.

4.2 Lane 2: Corona (Module-LWE Threshold Signatures)

Corona [5] is a lattice-based t-of-n threshold signature scheme over the polynomial ring $R_q = \mathbb{Z}_q[X]/(X^N + 1)$ with parameters $N = 256$, $q = 2^{48} + 2561$ (NTT-friendly), module dimension 8, and threshold $t = \lfloor 2n/3 \rfloor + 1$. It is a two-round, Module-LWE construction (Ringtail-derived parameter set) that builds on the Ringtail line [6].

- **DKG**: Q-Chain runs the Corona DKG once per epoch, producing the combined public key pk^{RT} and Shamir-shared secrets sk_i^{RT}. The DKG transcript hash is $\mathsf{qchain\_ceremony\_root}$.
- **Round 1 (precomputation)**: each party P_i samples a nonce $\mathbf{y}_i \sim D_{\sigma'}^N$ from PRF-correlated randomness, commits $\mathbf{w}_i = \mathbf{A}\mathbf{y}_i$, and broadcasts $\mathbf{w}_i$ with a MAC.
- **Round 2 (signing)**: given message $m = \mathsf{cert\_subject}$, compute challenge $c = H(m \,\|\, \sum_i \mathbf{w}_i)$, response $\mathbf{z}_i = \mathbf{y}_i + c \cdot sk_i^{\text{RT}}$, reject if $\|\mathbf{z}_i\|$ exceeds the bound (rejection sampling).
- **Combination**: the Q-Chain combiner reconstructs $\mathbf{z} = \sum_{i \in T} \lambda_i \mathbf{z}_i$ via Lagrange interpolation over the ring.
- **Verification**: $\|\mathbf{z}\| \le B$ and $H(m \,\|\, \mathbf{A}\mathbf{z} - c\mathbf{t}) = c$. *Batch verify is $O(1)$ amortized after DKG* because the verifier holds a fixed pk^{RT}.

Corona's security reduces to Module-LWE/SIS; the production parameters provide > 128-bit classical and > 130-bit quantum security (Appendix E of [15]).

4.3 Lane 3: MLDSAGroth16 (Z-Chain Rollup of ML-DSA-65)

The third lane is the conceptual heart of Quasar 3.0. The naive design would carry N raw ML-DSA-65 validator signatures in every block header, costing $N \times 3{,}309$ bytes (e.g. ∼69 KB at N=21, ∼330 KB at N=100) and $N \times 254\,\mu$s per verifier. We instead delegate ML-DSA verification to a Z-Chain Groth16 rollup. The on-chain artifact reduces from $N \times 3{,}309$ bytes to a fixed **192-byte SNARK plus 32-byte public-inputs hash**, irrespective of N.

Primitive: ML-DSA-65 (FIPS 204).

- Public key 1,952 B; secret key 4,032 B; signature 3,309 B.
- NIST Level 3, ≥128-bit quantum security.
- Verification: 254 μs CPU (cold), 3 μs with cached challenge (hot path).

Z-Chain rollup statement. The MLDSAGroth16 lane is a non-interactive zero-knowledge argument for the relation

$$\mathcal{R}_{\text{ZK}} = \left\{ (x; w) \;:\; x = (\mathsf{cert\_subject}, h_x) \wedge w = \left(\{pk_i^{\text{ML}}\}_{i=1}^N, \{\sigma_i^{\text{ML}}\}_{i=1}^N\right) \wedge \Phi(x, w) \right\}$$

where $\Phi(x, w)$ holds iff:

1. $h_x = H_{\text{Poseidon}}\big(\{pk_i^{\text{ML}}\}_{i=1}^{N} \,\|\, \mathsf{cert\_subject}\big)$.
2. $\sum_{i=1}^{N} [\![\mathsf{MLDSA.Verify}(pk_i^{\text{ML}}, \mathsf{cert\_subject}, \sigma_i^{\text{ML}}) = 1]\!] \;\geq\; t = \lfloor 2N/3 \rfloor + 1$.
3. Each pk_i^{ML} appears in P_{val} (the validator-set Merkle root) at the binding epoch.

The Groth16 prover, executed by Z-Chain validators, produces $\pi_{\text{ZK}} = (A, B, C) \in \mathbb{G}_1 \times \mathbb{G}_2 \times \mathbb{G}_1$ totalling 192 bytes (compressed), and emits the public-inputs hash h_x (32 bytes).

Why this is not a raw per-validator ML-DSA list. The rollup is a *single* succinct argument. The verifier never sees individual ML-DSA signatures; it sees only (π_{ZK}, h_x). This is critical for two reasons. First, on-chain footprint: the per-block post-quantum identity-binding cost is $192 + 32$ bytes regardless of validator count. Second, verification cost: Groth16.Verify is three pairings, executed in 200–500 μs when batched on a GPU (Section 8.1).

This lane is accountability, not a substitute for threshold ML-DSA. This lane proves a quorum of *individual* ML-DSA-65 signatures and compresses them to a constant on-chain footprint; it is the per-validator accountability mechanism, not Lux's threshold scheme. A genuine threshold ML-DSA — byte-equal to FIPS 204 — *is* provided separately by **Pulsar** (LP-4450 [16]): ML-DSA's Fiat–Shamir-with-Aborts requires a rejection-and-hint sequence over the *combined* response that no party can evaluate locally, and Pulsar resolves this by moving the rejection decision offline (boundary-cleared nonces) and recovering the hint from public data (FindHint over $\mathbf{w}' = \mathbf{A}\mathbf{z} - c\mathbf{t}_1 2^d$), never reconstructing $c\mathbf{s}_2$ or $c\mathbf{t}_0$ (Section 12.1). The two lanes are orthogonal: the Z-Chain Groth16 rollup gives the on-chain footprint and per-validator attribution *without altering the underlying primitive* (each validator runs FIPS-204 ML-DSA-65 exactly as standardized), while Pulsar gives the threshold seal under one group key.

Trusted setup. Z-Chain's Groth16 verifying key is the output of a multi-party trusted setup ceremony (powers-of-tau plus circuit-specific phase 2, with ≥ 1024 contributors). The verifying-key hash is rooted on Z-Chain as zchain_vk_root and bound into the certificate subject. A post-setup-compromise attacker can forge π_{ZK} but still cannot produce a Corona signature or break BLS, so single-compromise insufficiency (Theorem 7.4) holds.

5 Chain Separation: P / Q / Z / A / B / M / F

The Lux primary network distributes consensus, threshold cryptography, attestation, bridge, MPC, and FHE responsibilities across seven specialized chains. Each chain owns exactly one class of operation. This separation eliminates the operational and security coupling that any monolithic "T-Chain" design would impose: *consensus signing (Q-Chain) shares no state and no key material with general-purpose MPC (M-Chain) or FHE (F-Chain).*

5.1 Why Seven Chains, Not One

Q-Chain. The consensus-finalization threshold protocol must complete in ∼55 ms wall-clock per round. Pinning Q-Chain to Corona and nothing else (a) keeps the consensus latency budget independent of the custody and FHE subsystems, (b) limits the formal-analysis surface to one interactive subprotocol, and (c) prevents a stalled bridge or FHE ceremony from blocking finality.

Z-Chain. The Groth16 rollup substrate has different performance characteristics than threshold signing (prover wall-clock 200 ms–1 s on CPU; 5–15 ms on GPU). Pushing this work to a dedicated chain decouples block time from prover time and lets Z-Chain validators be specialized prover-class machines.

Chain	Role	Detail
P-Chain	Validator/stake authority	Holds the active validator set, stake table, BLS PoPs, and the ML-DSA / Corona public-key registry. Emits $P_{\text{val}} = \mathsf{pchain\_validator\_root}$.
Q-Chain	Corona consensus DKG	Runs the Corona two-round threshold protocol (Module-LWE, t-of-n) once per epoch and per block. Emits $Q_{\text{cer}} = \mathsf{qchain\_ceremony\_root}$. Does *not* run CGGMP21, FROST, or general MPC.
Z-Chain	Groth16 rollup + verifying-key registry	Each validator submits one ML-DSA-65 signature on $\mathsf{cert\_subject}$. Z-Chain runs the Groth16 prover and emits (π_{ZK}, h_x) with π_{ZK} at 192 B and h_x at 32 B. Also serves as a general-purpose ZK rollup substrate [20]. Emits $Z_{\text{vk}} = \mathsf{zchain\_vk\_root}$.
A-Chain	Unified attestation chain	Anchors genesis records, oracle commits, off-chain proofs, and other attestations [21]. Emits $A_{\text{att}} = \mathsf{achain\_attestation\_root}$.
B-Chain	Bridge messaging	Source of truth for cross-chain messages and lock/mint records. Emits $B_{\text{br}} = \mathsf{bchain\_bridge\_root}$.
M-Chain	MPC ceremonies	CGGMP21 (threshold ECDSA), FROST (threshold Schnorr), and general Corona (separate key material from Q-Chain consensus) for bridge custody and external-chain signing. Emits $M_{\text{mpc}} = \mathsf{mchain\_mpc\_root}$.
F-Chain	FHE compute	TFHE / encrypted EVM for confidential smart contracts. Emits $F_{\text{fhe}} = \mathsf{fchain\_fhe\_root}$.

Table 1: Per-chain responsibilities in Quasar 3.0. M-Chain and F-Chain are operationally orthogonal; they share only the `lux/chains/thresholdvm` substrate library, which provides the common share-management primitives. There is no monolithic "T-Chain".

M-Chain vs F-Chain (the former T-Chain split). A previous design considered a unified "T-Chain" running CGGMP21, FROST, Corona-general, and TFHE in one logical chain. Operational experience showed that MPC and FHE have qualitatively different liveness and resource profiles—MPC ceremonies are short, bursty, and network-bound; FHE compute is sustained, CPU-bound, and high-memory. We therefore split them: M-Chain for ceremonies, F-Chain for compute. They share the `thresholdvm` library substrate (share format, serialization, gossip), but each runs an independent consensus instance, validator set, and failure domain.

X-Chain (verification only). X-Chain remains the UTXO chain. It verifies signatures from multiple cryptographic families via the Fx credential plugin architecture (secp256k1, Ed25519, Corona, ML-DSA), but never initiates threshold signing. X-Chain is not part of the QuasarCert subject binding because it does not contribute a per-block root; UTXO state is recorded on chain via standard transactions.

5.2 Per-Block Pipeline

Lanes 1–3 of the QuasarCert run in parallel for each block B:

1. **BLS aggregation**: each validator signs cert_subject with BLS12-381. The proposer aggregates into a single 48-byte $\sigma_{\text{agg}}^{\text{BLS}}$. Wall-clock: $\sim$3 ms (GPU) or $\sim$100 ms (CPU) at n=21.
2. **Q-Chain (Corona)**: validators run Corona Round 1 (commitments) and Round 2 (responses) on cert_subject. The Q-Chain combiner produces σ^{RT}. Wall-clock: $\sim$55 ms (48 ms network + 7 ms combination).
3. **Z-Chain (Groth16)**: each validator signs cert_subject with ML-DSA-65 (254 μs sign; 254 μs verify cold; 3 μs verify cached) and submits the signature to Z-Chain. Z-Chain proves the batch via Groth16 ($\approx 2^{22.5}$ R1CS constraints per validator, App. B of [15]); prover wall-clock 200 ms (CPU) or 5–15 ms (GPU). Output is (π_{ZK}, h_x).

The block is finalized when all three components are available:

$$\mathsf{QuasarCert}(B) = \big(\sigma_{\text{agg}}^{\text{BLS}},\ \sigma^{\text{RT}},\ \pi_{\text{ZK}},\ h_x\big).$$

6 QuasarCert: Formal Definition and Replay Protection

6.1 QuasarCert

Definition 6.1 (QuasarCert)**.** *A QuasarCert for block B with respect to validator set $\mathcal{V} = \{v_1, \ldots, v_n\}$, certificate subject $m =$ cert_subject (Definition 2.4), and verifying-key root Z_{vk} is a 4-tuple*

$$\mathcal{C} \ = \ \big(\sigma_{\text{agg}}^{\text{BLS}},\ \sigma^{\text{RT}},\ \pi_{\text{ZK}},\ h_x\big)$$

where:

1. $\sigma_{\text{agg}}^{\text{BLS}} \in \mathbb{G}_1$ *is an aggregate BLS12-381 signature on m from $\geq t = \lfloor 2n/3 \rfloor + 1$ validators in P_{val}. Size: 48 bytes (compressed $\mathbb{G}_1$).*
2. σ^{RT} *is a Corona threshold signature on m under the Q-Chain combined public key bound to Q_{cer}. Size: $\sim$56 KB.*
3. $\pi_{\text{ZK}} \in \{0,1\}^{192 \cdot 8}$ *is a Groth16 proof [10] for the relation $\mathcal{R}_{\text{ZK}}$ of Section 4.3, attesting that $\geq t$ ML-DSA-65 signatures on m from validators in P_{val} exist. Size: 192 bytes ($A \in \mathbb{G}_1$, $B \in \mathbb{G}_2$, $C \in \mathbb{G}_1$, all compressed).*
4. $h_x \in \{0,1\}^{256}$ *is the public-inputs hash: $h_x = H_{Poseidon}\big(\{pk_i^{\text{ML}}\}_{i=1}^{N} \,\|\, m\big)$. Size: 32 bytes.*

Definition 6.2 (QuasarCert Verification)**.** $\mathsf{VerifyQuasarCert}(B, \mathcal{V}, \mathcal{C})$ *recomputes $m =$ cert_subject from B and the seven chain roots $\boldsymbol{\rho}$, then returns 1 iff all of:*

$$e\big(G_2,\ \sigma_{\text{agg}}^{\text{BLS}}\big) \ = \ e\big(apk_{\text{BLS}},\ H_{\mathbb{G}_1}(m)\big) \tag{1}$$

$$\mathsf{RT.Verify}\big(pk^{\text{RT}},\ m,\ \sigma^{\text{RT}}\big) = 1 \tag{2}$$

$$\mathsf{Groth16.Verify}\big(vk_Z,\ h_x,\ \pi_{\text{ZK}}\big) = 1 \tag{3}$$

$$h_x \ = \ H_{Poseidon}\big(\{pk_i^{\text{ML}}\}_{i=1}^{N} \,\|\, m\big). \tag{4}$$

vk_Z is recovered from Z_{vk} via Merkle path. The set $\{pk_i^{\text{ML}}\}$ is recovered from P_{val}.

Component sizes.

Remark 6.1. *In epoch mode (one Corona signature per epoch of k blocks, with per-block BLS and Z-Chain only), the amortized header overhead is $56\,KB/k + 48\,B + 224\,B$ per block. With $k = 1000$ this is $\sim$328 bytes/block.*

Cryptographic assumptions. The QuasarCert relies on:

1. co-CDH on BLS12-381 [4]: given $(g, g^a, h) \in \mathbb{G}_2^2 \times \mathbb{G}_1$, computing h^a is hard.
2. Module-LWE [11]: distinguishing $(\mathbf{A}, \mathbf{As} + \mathbf{e})$ from $(\mathbf{A}, \mathbf{u})$ is hard (Corona basis).

Component	Hardness Assumption	Size
$\sigma_{\text{agg}}^{\text{BLS}}$ (BLS aggregate)	co-CDH on BLS12-381	48 B
σ^{RT} (Corona)	Module-LWE / Module-SIS	∼56 KB
π_{ZK} (Groth16 of ML-DSA)	KEA/AGM + MLWE/MSIS	192 B
h_x (public-inputs hash)	Hash collision resistance	32 B
Total		∼56.27 KB

Table 2: QuasarCert component sizes. The $192 + 32 = 224$ bytes for the post-quantum identity-binding lane is independent of validator count N. At N=100 validators a per-validator-ML-DSA scheme would carry 330,900 bytes for the same lane.

3. MLWE / MSIS [2]: ML-DSA-65 (FIPS 204) is EUF-CMA secure under Module-LWE and Module-SIS at NIST Level 3.
4. Knowledge of Exponent + Algebraic Group Model [10]: Groth16 is knowledge-sound.

6.2 Replay Protection via Subject Binding

Theorem 6.1 (Cross-Chain / Cross-Epoch Replay Impossibility)**.** *Let $\mathcal{C} = (\sigma_{\text{agg}}^{\text{BLS}}, \sigma^{\text{RT}}, \pi_{\text{ZK}}, h_x)$ be a QuasarCert that was produced for block B on chain c at epoch e, round r, mode μ, under chain roots $\boldsymbol{\rho}$ with parent context (h_{parent}, s, x, g, b). For any block B' on chain $c' \neq c$, or epoch $e' \neq e$, or round $r' \neq r$, or mode $\mu' \neq \mu$, or any tuple $(\boldsymbol{\rho}', h'_{parent}, s', x', g', b') \neq (\boldsymbol{\rho}, h_{parent}, s, x, g, b)$, $\mathsf{VerifyQuasarCert}(B', \mathcal{V}', \mathcal{C})$ returns 0 except with probability $\mathsf{negl}(\lambda)$.*

Proof. The verifier of Definition 6.2 recomputes $m = \mathsf{cert\_subject}$ from the inputs of B' before checking each lane. Suppose B' differs from B in any of $c, e, r, \mu, \boldsymbol{\rho}, h_{\text{parent}}, s, x, g, b$. Then the input to keccak_{256} in Definition 2.4 differs by at least one byte. By collision resistance of keccak_{256}, the new m' equals the original m with probability $\leq 2^{-256}$.

If $m' \neq m$:

- Equation (1) fails because BLS verification is message-binding under co-CDH (EUF-CMA).
- Equation (2) fails because Corona is EUF-CMA under MLWE.
- Equation (4) fails because h_x depends on m via Poseidon hashing, and Poseidon is collision-resistant.
- Equation (3) fails because π_{ZK} was proved against the original h_x, not the new one; an adversary adapting π_{ZK} to the new h_x contradicts Groth16 knowledge-soundness in the AGM.

Each lane independently rejects, so verification returns 0.

The forgery probability is bounded by the union of $\mathsf{negl}(\lambda)$ from co-CDH, MLWE, Poseidon collision resistance, and Groth16 knowledge-soundness: $\Pr[\text{replay accepted}] \leq \mathsf{negl}(\lambda)$. □

Corollary 6.2 (Replay across Modes)**.** *A QuasarCert produced in mode $\mu = 3$ (Quantum) cannot be presented as a valid mode-$\mu' \neq 3$ certificate, because μ enters the subject and the lower-mode verifiers would still recompute the original mode-3 subject.*

Remark 6.2. *This binding is* structural*: it does not rely on any nonce registry, sequence number, or external replay cache. The certificate itself encodes its full context, so the verifier can decide replay locally and statelessly.*

7 Security Analysis

We state seven theorems. Full proofs are machine-checked in Lean 4 (`Quasar.lean`, `BFT.lean`, `BLS.lean`, `Corona.lean`, `MLDSA.lean`, `Groth16.lean`) and protocol-level network attacks are

verified in Tamarin (`QuasarConsensus.spthy`). The companion proof sketch [15] contains full circuit analysis (App. B), the adaptive corruption argument (App. C), trusted-setup analysis (App. D), and concrete-quantum security tightness (App. E).

7.1 Safety

Theorem 7.1 (Safety)**.** *If fewer than $f+1$ validators are Byzantine ($f < n/3$), then no two conflicting blocks B_h and B'_h at the same height h can both obtain valid QuasarCerts.*

Proof sketch. A QuasarCert requires an aggregate BLS signature from a quorum Q_h^{BLS} with $|Q_h^{\text{BLS}}| \geq 2f+1$. Suppose both B_h and B'_h have valid certificates. Then

$$|Q_h^{\text{BLS}} \cap Q_{h'}^{\text{BLS}}| \;\geq\; 2(2f+1) - n \;=\; f+2 \;\geq\; 1$$

where $n \leq 3f+1$. The intersection contains at least $f+1$ validators, of which at least one is correct. A correct validator never signs two conflicting subjects at the same height (the certificate subject binds the parent block hash and the height). Contradiction.

The same intersection argument applies independently to the Corona quorum (threshold $t = 2f+1$ contributors required by $\mathsf{qchain\_ceremony\_root}$) and to the Groth16 lane (the rollup proves $\geq t$ valid ML-DSA-65 signatures from validators in P_{val}). Each lane independently prevents conflicting certificates. □

7.2 Liveness

Theorem 7.2 (Liveness)**.** *After GST, if the leader for height h is correct, then a valid QuasarCert for some block at height h is produced within bounded time $3\Delta + T_{\text{sign}}$ where T_{sign} is the maximum signing latency across the three lanes (BLS sign, Corona two rounds, ML-DSA sign + Z-Chain prover).*

Proof sketch. After GST, a correct leader's proposal reaches all correct validators within Δ. Wave converges in finite rounds (since $\theta_{\max} < 1$ and β is finite). At least $2f+1$ correct validators lock their preference and enter Ray. Each correct validator produces a BLS signature (non-interactive), an ML-DSA-65 signature (non-interactive), and contributes to Corona Round 1 and Round 2 (interactive, 2Δ). The Z-Chain prover is non-blocking on network-delivery and bounded by T_{ZK} ($\leq$15 ms GPU batched). Total: proposal delivery (Δ) + Wave + Corona rounds (2Δ) + Z-Chain proof (T_{ZK}) + finalization message (Δ). The dominant term is $3\Delta + T_{\text{sign}}$. □

7.3 QuasarCert Unforgeability

Theorem 7.3 (QuasarCert Unforgeability)**.** *Forging a QuasarCert requires simultaneously breaking: (i) co-CDH on BLS12-381, (ii) Module-LWE for Corona, (iii) MLWE/MSIS for ML-DSA,* and *(iv) Groth16 knowledge-soundness in the AGM. The probability of forgery is bounded by:*

$$\Pr[\text{Forge}_{\text{Quantum}}] \;\leq\; \mathsf{Adv}^{\textit{co-CDH}} \cdot \mathsf{Adv}^{\textit{MLWE-RT}} \cdot \left(\mathsf{Adv}^{\textit{MLWE/MSIS-ML}} + \mathsf{Adv}^{\textit{KEA}}\right).$$

Proof sketch. A valid QuasarCert verifies on all three lanes (Definition 6.2). An adversary without $f+1$ corrupted validators must forge at least one signature in each lane. The three lanes rest on algebraically independent primitives—BLS12-381 over an elliptic curve, Corona over $R_q = \mathbb{Z}_q[X]/(X^N+1)$ with $q \approx 2^{48}$, ML-DSA over R_q with $q = 8380417$ and disjoint module dimensions, and Groth16 over the same pairing as BLS but distinct relation. By a standard hybrid argument the adversary's joint forgery advantage is bounded by the product of the per-lane advantages. □

7.4 Single-Compromise Insufficiency

Theorem 7.4 (Single-Compromise Insufficiency)**.** *An adversary that breaks exactly one of the three cryptographic lanes cannot produce a valid conflicting QuasarCert, provided $f < n/3$ honest validators remain.*

Proof sketch. Case analysis on which lane is compromised.

Case 1: BLS broken. The adversary forges the aggregate $\sigma_{\text{agg}}^{\text{BLS}}$. It still needs (a) a valid Corona σ^{RT} from $\geq t$ Q-Chain shares and (b) a valid Groth16 π_{ZK} proving $\geq t$ ML-DSA signatures. The adversary controls at most f shares and f ML-DSA keys; since $f < t$, neither component can be assembled.

Case 2: Corona broken. Symmetric. The adversary still needs $\geq t$ valid BLS signatures and a valid Groth16 proof.

Case 3: ML-DSA / Groth16 broken. Symmetric. The adversary still needs $\geq t$ valid BLS signatures and a valid Corona signature.

In each case the two intact lanes enforce the quorum-intersection argument of Theorem 7.1. □

7.5 QuasarCert Soundness

Theorem 7.5 (QuasarCert Soundness)**.** *Let $\mathcal{C} = (\sigma_{\text{agg}}^{\text{BLS}}, \sigma^{\text{RT}}, \pi_{\text{ZK}}, h_x)$ be a QuasarCert that verifies for block B under validator set $\mathcal{V}$ with $|\mathcal{V}| = n$ and threshold $t = \lfloor 2n/3 \rfloor + 1$. Under co-CDH, Module-LWE, MLWE/MSIS, and KEA/AGM, at least t validators in $\mathcal{V}$ signed* $\mathsf{cert\_subject}(B)$*. Equivalently, for any $B' \neq B$ not agreed upon by an honest quorum, no QuasarCert for B' verifies except with* $\mathsf{negl}(\lambda)$ *probability.*

Proof. We show forging any single component requires breaking its assumption. Since all three must verify (Definition 6.2), forging the full certificate requires breaking all three.

Lane 1: $\sigma_{\text{agg}}^{\text{BLS}}$. By Boneh–Lynn–Shacham [4], BLS signatures are EUF-CMA secure under co-CDH in the random oracle model. PoP prevents the rogue-key attack. For the aggregate to verify with the expected quorum key, the contributing set S has $|S| \geq t$.

Lane 2: σ^{RT}. By the EUF-CMA theorem for Corona [5], an adversary controlling $f < t$ parties cannot produce a valid threshold signature on a message not signed by an honest quorum. Advantage: $\mathsf{Adv}_{\text{RT}}^{\text{EUF-CMA}} \leq Q_H \cdot \mathsf{Adv}^{\text{MLWE}} + Q_S/2^{128}$.

Lane 3: (π_{ZK}, h_x). Groth16 is knowledge-sound under KEA in the AGM [10]: a valid proof admits an extractor that recovers a witness, here the $\geq t$ ML-DSA-65 signatures. Each extracted signature is valid under pk_i^{ML}, which by FIPS 204 EUF-CMA implies validator v_i produced it. The Poseidon binding $h_x = H(\{pk_i^{\text{ML}}\} \,\|\, m)$ ensures the extracted signatures are over the recomputed certificate subject m.

Composition. By a union bound on independent reductions:

$$\mathsf{Adv}^{\mathsf{forge}} \leq \mathsf{Adv}^{\text{co-CDH}} + \mathsf{Adv}^{\text{MLWE-RT}} + \mathsf{Adv}^{\text{MLWE/MSIS-ML}} + \mathsf{Adv}^{\text{KEA}} \;=\; \mathsf{negl}(\lambda).$$

□

7.6 Parallel Liveness

Theorem 7.6 (Parallel Liveness)**.** *If $f < n/3$ and the network is synchronous after GST with delay $\leq \Delta$, then for every valid block B proposed by an honest leader, a valid QuasarCert is produced within $\max(\Delta_{\text{BLS}}, \Delta_{\text{RT}}, \Delta_{\text{ZK}}) + O(\Delta)$. An adversary must stall all three lanes simultaneously to prevent finality.*

Proof. Each lane completes independently:

BLS path. Each honest validator signs cert_subject on receipt of B. The proposer collects $\geq t$ signatures within 2Δ. Aggregation is non-interactive ($\sim$3 ms GPU). Total: $2\Delta + O(1)$.

Corona path. Round 1 commitments arrive within Δ; Round 2 responses arrive within 2Δ once $\geq t$ valid Round 1 messages are collected; Q-Chain combination in 7 ms. Total: $2\Delta + O(1)$.

Z-Chain path. Each honest validator signs cert_subject with ML-DSA-65 (254 μs) and submits to Z-Chain. Z-Chain collects honest signatures within $\Delta + O(1)$ and runs the Groth16 prover ($\approx 2^{22.5}$ R1CS constraints; App. B of [15]); 200 ms CPU or 5–15 ms GPU. For missing signatures from Byzantine validators Z-Chain waits up to T_{ZK} then proves the relation for the validators that did sign. Total: $\Delta + T_{\mathrm{ZK}} + O(1)$.

Why all three must be stalled. Stalling any single lane requires preventing $> n - t \geq n/3$ validators from contributing. With $f < n/3$ no bounded adversary can stall even one lane, much less all three. □

7.7 Post-Quantum Safety

Theorem 7.7 (Post-Quantum Safety)**.** *If a quantum adversary breaks BLS12-381 co-CDH (Shor on the pairing group), QuasarCert verification restricted to the Corona and MLDSAGroth16 lanes still guarantees soundness under Module-LWE and MLWE/MSIS. The Groth16 lane has a pre-configured fallback to direct verification of $\geq t$ raw ML-DSA-65 signatures.*

Proof. **Corona under quantum attack.** Best known quantum attack on Module-LWE is lattice sieving with Grover speedup, achieving $2^{0.265\beta}$ time for BKZ blocksize β. With Corona parameters (N=256, $q \approx 2^{48}$, module dimension 8, LWE dimension 2048), the cost is $\geq 2^{130}$ operations (App. E of [15]). $\mathcal{A}_Q$ cannot forge σ^{RT}.

ML-DSA under quantum attack. ML-DSA-65 is parameterized for NIST Level 3 ($\geq$128-bit quantum security) [2]. Best quantum attack (Core-SVP with quantum sieving) requires $\geq 2^{128}$ operations.

Groth16 under quantum attack. A quantum adversary breaks KEA on the pairing group (Shor extracts discrete logs, enabling proof forgery). In this case Quasar 3.0 falls back to the VerifyPQ predicate:

$$\mathsf{VerifyPQ}(B, \mathcal{V}, \mathcal{C}) = \mathsf{RT.Verify}(\cdot) \ \wedge \ \mathsf{MLDSA.VerifyQuorum}(\cdot)$$

where MLDSA.VerifyQuorum checks $\geq t$ raw ML-DSA-65 signatures (the proposer-side "signal flare" broadcasting validators' raw signatures). This costs $t \cdot 254\,\mu$s extra CPU per block and is acceptable in the post-CRQC fallback regime.

Explicit reduction. Suppose $\mathcal{A}_Q$ forges under VerifyPQ. Construct $\mathcal{R}$ from MLWE: embed the challenge as either the Corona public key or as a target ML-DSA-65 public key, simulate the protocol, and extract a short vector solving the SIS instance dual to the LWE challenge. By a standard hybrid argument:

$$\mathsf{Adv}^{\mathsf{VerifyPQ}}_{\mathcal{A}_Q} \ \leq \ 2 \cdot \max\big(\mathsf{Adv}^{\mathrm{MLWE\text{-}RT}}, \mathsf{Adv}^{\mathrm{MLWE/MSIS\text{-}ML}}\big) \ = \ \mathsf{negl}(\lambda).$$

□

8 Performance (Quasar 3.0, December 2025)

All measurements correspond to the 25 December 2025 production release on commodity hardware. Single-validator-process numbers were taken on a standard mainnet board (32-core x86 CPU, NVIDIA RTX 6000 Ada-class GPU, 128 GB RAM); GPU lane numbers correspond to the QuasarGPU adapter (LP-132 [18]). Each benchmark ran for $\geq 10^4$ iterations.

Lane	Operation	CPU latency	GPU latency	Notes
BLS	Aggregate verify, n=21	∼260 μs	<50 μs	2 pairings, batched
BLS	Aggregate verify, n=100	∼875 μs	<120 μs	2 pairings, batched
ML-DSA	Verify per validator, cold	254 μs	50 μs	FIPS 204
ML-DSA	Verify per validator, cached	3 μs	1 μs	challenge cache
Groth16	Verify (3-pairing)	1.2 ms	200–500 μs	batched on GPU
Groth16	Prove (N=21 ML-DSA)	200 ms	5–15 ms	$\approx 2^{22.5}$ R1CS
Corona	Round 1 share	1.8 ms	—	precomputable
Corona	Round 2 sign+combine	9 ms	—	per round
Corona	Batch verify	$O(1)$ amortized	—	cached after DKG

Table 3: Per-lane verification primitives. The Groth16 verify step is the dominant cost on the GPU verification path; Corona batch verify amortizes to $O(1)$ once the DKG is loaded.

8.1 Per-Lane Verification Costs

8.2 Full QuasarCert Verification

Validators (n)	Threshold (t)	CPU verify	GPU verify	Cert size
4	3	1.5 ms	0.5 ms	56.3 KB
21	14	2.4 ms	0.5–1.5 ms	56.3 KB
50	34	3.0 ms	1.0–1.5 ms	56.3 KB
100	67	3.5 ms	1.5 ms	56.3 KB

Table 4: End-to-end QuasarCert verification at the 25 December 2025 release. Latency is the parallel max of BLS aggregate, Corona batch verify, and Groth16 verify on the QuasarGPU substrate. Certificate size is constant in n because the third lane is a Z-Chain rollup (192 B SNARK + 32 B public-inputs hash) rather than per-validator ML-DSA signatures.

8.3 Same-message BLS aggregate batching (cevm v0.47.1)

Quasar consensus rounds repeatedly verify n signatures over the same 32-byte cert subject (Section 6.2). cevm v0.47.1 (commit `6bf0e232`) added a 2-way set-associative 4096-slot pubkey-decompression cache (FNV1a-64 over compressed point bytes plus a 48-byte memcmp guard) on top of the same-message batched verifier introduced in v0.45. At n=1024 this lifts the speedup from 9.24× to **16.51**× versus the flat host-blst baseline (≥ 10× target exceeded by 65%). Pairing math remains the canonical `luxcpp/crypto/bls` c-abi body; the batching plus pubkey-cache amortization is what produces the published number. Source: `LP-137-BENCHMARKS.md`, Phase-3 roll-up.

8.4 Stage 5b BLS pairing structural ceiling on Apple M1 Max

A pure-Metal single-pairing measurement (luxcpp/crypto commit `9ff799fd`, 2026-04-27) lands at ∼475 ms on M1 Max versus ∼510 μs for host blst, a 930× slowdown. Cause: ∼280 dispatches per pairing at ∼10 μs each (init / add+line / dbl+line / sqr_ret / fold_line / finalize for Miller; conj / inv / cyclo_sqr / mul / frob for final exponentiation). The serial Fp12 chain mismatches the SIMD GPU shape; per-dispatch GPU compute is ∼770 μs for a single-thread Fp12 op. The architecture is proven byte-equal blst across 2 746 vectors; the SoTA single-pairing path is Linux+CUDA, batched-N parallel kernel saturation (1 warp per pairing on M1 Max ≥ 32-way), or Karabina compressed cyclo squarings. Production binaries from cevm v0.46.0 onward link zero blst symbols (CI-asserted) and route through the canonical c-abi body.

8.5 Block Time and Throughput

The mainnet target for the 25 December 2025 release is:

- **Block time**: 500 ms.
- **Throughput**: ≥100K TPS sustained per chain.
- **Per-cert verify on the 21-validator board**: 0.5–1.5 ms.
- **Catch-up verify (skip-list mode)**: ≤2 ms per epoch boundary thanks to the constant-size third lane.

The cert-verification latency is dominated by the parallel max of three lanes; with QuasarGPU streaming, none of the three exceeds 1.5 ms at n=21. Block time is therefore signing-bound (Corona two-round network latency, ∼55 ms) rather than verification-bound.

8.6 Primitive Costs

Lane	Operation	Latency	Sig Size	PK Size
BLS12-381	Key generation	87 µs	—	96 B
BLS12-381	Sign	371 µs	96 B	—
BLS12-381	Aggregate (100 sigs)	4.8 µs	48 B	—
ML-DSA-65	Key generation	142 µs	—	1,952 B
ML-DSA-65	Sign	254 µs	3,309 B	—
ML-DSA-65	Verify (cold)	254 µs	—	—
ML-DSA-65	Verify (cached)	3 µs	—	—
Corona	DKG (5-of-7)	12.4 ms	—	∼56 KB
Corona	Round 1 (precomp)	1.8 ms	—	—
Corona	Round 2 (sign)	2.1 ms	∼56 KB	—
Corona	Verify	0.9 ms	—	—

Table 5: Per-primitive costs on the production mainnet board.

8.7 ZAP Wire Protocol

Consensus messages travel on ZAP (Zero-Allocation Protocol), a binary wire format that eliminates serialization overhead on the consensus hot path [19].

Configuration	Throughput	Per-Message
Single connection	114K TPS	8.7 µs
20 parallel connections	376K TPS	2.7 µs
50 conn + batch 1000	20.26M TPS	49 ns

Table 6: ZAP transport throughput. Batch mode approaches memory bandwidth.

8.8 End-to-End Comparison

Network	Consensus	Wire TPS	Protocol
Lux (Quasar 3.0)	Photon/Wave/Focus/Ray + QuasarGPU	**≥100K (sustained chain)**	ZAP
Avalanche	Snowball	246K (consensus messages)	gRPC / Protobuf
Solana	Tower BFT	65K (network)	Custom binary

Table 7: Consensus-layer wire and chain throughput.

8.9 Multi-Host Distributed Measurements (2026-06-03)

The numbers in Sections 8.1–4 characterize the in-process verification primitives on a single mainnet board. Production network rounds further pay the *cross-host* cost of: (a) each validator's local share computation, (b) the partial-signature transport leg to the coordinator, and (c) the constant-time cert composition once the t-th partial arrives. The wall-clock finality observed by a client is the parallel max of the configured legs plus propagation and compose.

To bound this we ran a distributed bench on a 3-host cluster:

- **ra** – Apple M1 Max, 10-core, macOS 26.5 (darwin-arm64).
- **spark** – NVIDIA arm64 server (Blackwell), Ubuntu 24.
- **evo** – AMD Ryzen AI MAX+ 395 (x86_64), WSL2 Linux 6.6.

Each host ran one `thresholdd` process exposing `bls.{keygen,sign}`, `pulsar.{keygen,sign}`, and `corona.{keygen,sign}` over JSON-RPC. A coordinator on `ra` drove 5 rounds per (scale, profile) cell using HTTP/JSON-RPC over an SSH tunnel (a conservative proxy for the QUIC transport that production deploys; SSH/TCP adds 10–30 ms per round vs. QUIC 0-RTT). Per-host sign cost was measured directly; the coordinator-side leg arrival time is max(per-host sign ms)+ measured one-way prop ms.

Scale	Profile	Pulsar leg	Corona leg	Mag. leg	Finality p_{50}
10v	Aurora	189	1 204	—	1 204
10v	Polaris	197	1 258	137	1 258
10v	Hybrid-1s	203	1 135	141	1 135
100v	Aurora	374	2 682	—	2 682
100v	Polaris	366	2 658	132	2 658
100v	Hybrid-1s	298	2 593	134	2 593
500v	Aurora	427	2 542	—	2 542
500v	Polaris	489	2 692	175	2 692
500v	Hybrid-1s	408	2 655	146	2 655

Table 8: End-to-end leg arrival (p_{50}, milliseconds, including network propagation) on the 3-host distributed bench (CPU baseline, GPU acceleration for Corona pending). Hybrid-1s adds a BLS fast-path that announces preliminary finality at 159–187 ms across all scales; the Polaris cert tier still arrives behind the dominant Corona leg. "Validators" is the modeled t/n ratio ($\approx$67%) at n=20 ceiling per process (Caveat 1).

Three structural findings follow.

Finding 1: BLS dominates fast-path. At every scale the BLS leg in Hybrid-1s is finished in 160–190 ms while the Polaris cert tier (Pulsar+Corona+Magnetar) takes 1.1–2.7 s. This validates the LP-073 cert-tier degradation design: a hybrid chain can publish a useful BLS-finality block at t_{BLS} and let the PQ tier land asynchronously without blocking the client-visible block clock.

Finding 2: Corona leg is the bottleneck. Corona sign latency scales near-linearly in t in our measured range: 651 ms at n=10, 1 204 ms at n=15, 1 721 ms at n=20 on the M1 Max (Section 8.10). This is CPU-baseline; GPU offload (Blackwell on spark, Vulkan on evo) is expected to bring this to sub-second based on luxfi/threshold's published GPU lane work-in-flight, but our cluster did not yet saturate the GPU during this run ($\leq$ 16% utilization).

Finding 3: t/n-ratio scaling holds. Holding the threshold ratio at $\approx$67%, Pulsar sign scales sub-linearly in n: 48 ms at n=10 rises only to 144 ms at n=20 on the M1 Max. The cert-compose cost is bounded by the constant-size signature plus a fixed-size hash; the dominant order-of-magnitude jump from 10v to 100v finality is the Corona leg, not Pulsar.

8.9.1 Caveats

1. **Cap on benchmark n.** The `thresholdd` keygen ceremony at n=100 exceeds 1 GB of RAM per process and takes 30+ seconds; we cap n_{bench}=20 and preserve the t/n ratio. The Phase-5 sweep (Section 8.10) confirms that sign cost (the per-round bottleneck) scales near-linearly in t in the measured range, so the projection from n=20 to n=100 on the sign leg is sound. Keygen is a one-time per-epoch cost, not a per-round cost, so the cap does not affect the block-time numbers.
2. **Magnetar via published number, not per-round sign.** The JSON-RPC bench driver does not expose `magnetar.sign` as a measurable in this run; the Magnetar column uses the per-host SLH-DSA-128f published sign cost (2–5 ms across the three hosts) added to the measured prop. This is consistent with luxfi/magnetar's published numbers.
3. **HTTP/JSON-RPC, not QUIC.** The harness uses HTTP over SSH tunnels in place of QUIC. SSH/TCP adds 10–30 ms per round relative to QUIC 0-RTT. Production deployment of QUIC will lower the finality numbers in Table 8 by approximately one network RTT.
4. **Corona GPU parity pending.** A sister agent (the *Corona/Pulsar parity* effort) is in flight to land GPU-accelerated Corona sign on the Blackwell lane. Once that work completes and the bench re-runs, the Corona leg should drop by 5−20×, moving 100v Aurora finality to ∼500−700 ms and the Hybrid-1s preliminary finality to BLS-dominated 160−190 ms. The CPU numbers here are the worst case.

8.10 Phase-5 Tuning Sweep

We swept the t/n ratio at fixed ≈67% on each host to characterize per-primitive scaling.

		Pulsar sign p_{50} (ms)			**Corona sign p_{50} (ms)**			**BLS sign p_{50} (ms)**		
n	t	ra	spark	evo	ra	spark	evo	ra	spark	evo
10	7	48	58	19	652	762	877	9.3	22.4	12.3
15	11	124	108	77	1 205	1 470	1 675	13.3	35.1	18.2
20	14	144	133	155	1 722	2 200	2 491	16.9	44.5	23.0

Table 9: Per-host primitive sign cost vs. n at threshold ratio 0.67. BLS scales sub-linearly (amortized Lagrange interpolation). Pulsar scales near-linearly in n. Corona scales linearly in n with slope 150−250 ms per added participant on the CPU baseline.

Note on the requested QUIC/DHT/gossip sweep. The distributed-bench harness runs over HTTP/JSON-RPC on the existing SSH tunnel, not the luxfi/consensus QUIC transport, so it cannot directly vary `MaxStreamsBidi`, `InitialStreamReceiveWindow`, the DHT k-bucket, or the gossip fanout. The above sweep instead varies t, n – the correct axis for the crypto-primitive cost, which sits underneath any transport tuning. A separate QUIC-transport bench in luxfi/consensus would be required to sweep the four transport params; this report does not include it.

8.11 Memory Bounds

Epoch-based pruning with a 6-epoch retention window bounds consensus state to ∼5.9 MB after 10^6 finality events. The bound holds because each epoch stores at most E block certificates and n validator states, and $6 \cdot E \cdot (|\mathcal{C}| + n \cdot |state|)$ is constant for fixed parameters. The per-block QuasarCert footprint of 56.3 KB is dominated by the Corona signature; in epoch mode (one Corona per k blocks), the per-block amortized overhead drops to 328 bytes at $k = 1000$.

9 Implementation

Quasar 3.0 ships in the `github.com/luxfi/consensus` package on 25 December 2025. Implementation highlights:

- **Language**: Go 1.24. Single binary, explicit error handling, no runtime panics on the consensus path.
- **Wire protocol**: ZAP (`github.com/luxfi/zap`), a zero-allocation binary protocol with optional PQ-TLS 1.3 (X25519MLKEM768) transport encryption [19].
- **Storage**: SQLite WAL mode via ZapDB for consensus state. Epoch-based pruning removes state older than 6 epochs.
- **Cryptography**: `luxfi/crypto/bls` for BLS12-381, `luxfi/crypto/mldsa` for ML-DSA-44/65/87, `luxfi/corona` for Module-LWE threshold signing, `luxfi/zk` for the Z-Chain Groth16 prover/verifier.
- **QuasarGPU adapter**: LP-132 [18] provides a GPU-native verification substrate that schedules BLS, Corona, and Groth16 verifies onto the same kernel stream as EVM execution. Verification consumes no incremental host-CPU budget on the hot path.
- **Three-lane signing**: `TripleSignRound1` (`consensus/protocol/quasar/quasar.go`) dispatches BLS, Corona, and ML-DSA signing in separate goroutines. σ_i^{ML} is submitted to Z-Chain rather than gossipped, so the mainnet message size remains $\mathcal{O}(96 + 56\text{KB})$ rather than $\mathcal{O}(N \cdot 3{,}309)$.
- **Fail-closed Corona**: if Q-Chain's threshold round fails (insufficient shares, rejection sampling failure, network partition during DKG), the protocol is specified to fall back to Mode 2 (BLS + MLDSAGroth16). The current wire layer (`PolicyQuantum`) requires Corona by default (`requireRT = true`); the Mode 2 fallback is governed by this flag and an out-of-band operator decision rather than an automatic in-protocol degradation. It does not degrade to BLS-only without explicit operator action.
- **Z-Chain prover degradation**: if the Z-Chain prover misses T_{ZK}, the proposer broadcasts the raw $\geq t$ ML-DSA-65 signatures (the post-quantum "signal flare" fallback referenced in Theorem 7.7) instead of the Groth16 proof.
- **FPC integration**: the Wave FPC threshold selector (`protocol/wave/fpc/fpc.go`) uses SHA-256-based PRF for deterministic phase-dependent threshold selection.
- **DAG finalization**: pre-allocated ring buffers for vote counting, frontier-set reuse for DAG traversal, zero GC pressure during consensus rounds. Finalization check 13.75 ns at depth 10, 113 ns at depth 100 (sub-linear via frontier caching).

9.1 Module Structure

9.2 Configuration

Quasar 3.0 auto-configures consensus parameters from validator count: `GetConfig(nodeCount)` computes k, α, and β to satisfy safety and liveness for the given n. Three presets: `Config.SingleValidator()`, `Config.Testnet()`, and `Config.Mainnet()`. The 25 December 2025 mainnet activates with `Config.Mainnet()` at $n = 21$ and Mode $\mu = 3$ (Quantum, all three lanes).

10 Formal Verification

The safety, liveness, replay-impossibility, and unforgeability theorems are machine-checked in Lean 4. The protocol's resistance to network-level attacks (replay, reflection, MITM) is verified in Tamarin.

The Lean development imports Module-LWE and ML-DSA assumptions as axioms; QuasarCert soundness is then derived constructively. The companion proof sketch [15] (Appendices A–E)

Package	Function
`protocol/photon/`	Proposal emission
`protocol/wave/`	FPC voting, confidence tracking
`protocol/wave/fpc/`	PRF-based threshold selector
`protocol/focus/`	Certificate-subject derivation
`protocol/ray/`	Three-lane signing dispatch
`protocol/quasar/`	QuasarCert construction & verification
`protocol/nova/`	Linear-chain consensus mode
`protocol/nebula/`	DAG consensus mode
`protocol/prism/`	DAG geometry and cutting
`protocol/flare/`	Nebula proposer (DAG variant of Photon)
`protocol/horizon/`	Nebula DAG voting
`protocol/field/`	Nebula finalization
`core/dag/`	Horizon, flare, frontier algorithms
`engine/chain/`	Linear-chain engine
`engine/dag/`	DAG engine
`engine/pq/`	Post-Quasar certificate engine
`types/`	Bag, config, consensus types

Table 10: Package structure of the Quasar 3.0 implementation.

Property	Tool	File
Safety (Thm. 7.1)	Lean 4	`Quasar.lean`
Liveness (Thm. 7.2)	Lean 4	`BFT.lean`
Replay impossibility (Thm. 6.1)	Lean 4	`Subject.lean`
BLS unforgeability	Lean 4	`BLS.lean`
Corona EUF-CMA	Lean 4	`Corona.lean`
ML-DSA-65 EUF-CMA	Lean 4	`MLDSA.lean`
Groth16 knowledge-soundness	Lean 4	`Groth16.lean`
QuasarCert unforgeability (Thm. 7.3)	Lean 4	`Quasar.lean`
QuasarCert soundness (Thm. 7.5)	Lean 4	`QuasarCert.lean`
Protocol authentication	Tamarin	`QuasarConsensus.spthy`
Replay resistance (network)	Tamarin	`QuasarConsensus.spthy`

Table 11: Formal-verification artifacts shipped with the 25 December 2025 release. Lean 4 proofs compile against `mathlib4@v4.10`; Tamarin runs against version 1.8.0.

provides the human-readable argument and the concrete-quantum security parameters used to discharge the axioms numerically.

11 Related Work

Classical BFT. PBFT [7] introduced practical Byzantine fault tolerance with $O(n^2)$ message complexity. HotStuff [8] reduced this to $O(n)$ with a leader-based pipelining approach. Quasar 3.0 inherits HotStuff's linear message complexity but replaces the single-signature finality artifact with a three-lane QuasarCert that binds seven chain roots into the certificate subject.

Probabilistic consensus. The Snow protocol family [9] uses repeated subsampled voting with metastable convergence. Quasar's Wave phase uses FPC, a related mechanism with PRF-based adaptive thresholds that provides deterministic threshold selection per round, eliminating the need for common randomness beacons.

Post-quantum blockchain. Prior work on post-quantum blockchains has focused on replacing individual signature schemes (e.g., XMSS for transaction signing). Quasar 3.0 is the first deployed protocol to bind *three* algebraically independent finality lanes into a single certificate, with the third lane delivered as a Z-Chain Groth16 rollup of FIPS-204 ML-DSA signatures.

Threshold lattice signatures. FIPS 204 specifies ML-DSA for single-signer use only, but a byte-equal *threshold* ML-DSA is nonetheless achievable: **Pulsar** (LP-4450 [16]) is a Module-LWE threshold whose combined output verifies under unmodified FIPS 204 ML-DSA-65/87, via boundary-cleared nonce transcripts and public-data hint recovery (BCC/CEF). Corona [5] is the sibling Module-LWE leg — the Ringtail-derived two-round scheme optimized for permissionless DKG — composed alongside Pulsar for intra-lattice diversity. Section 12 gives the BCC/CEF construction, its honest review-ready status, and the relationship to the Z-Chain accountability rollup.

ZK aggregation of post-quantum signatures. Recent proposals aggregate ML-DSA or ML-DSA signatures via Halo 2 or Plonky2 SNARKs. Quasar 3.0 uses Groth16 because its proof size (192 B) and verification cost (3 pairings) are constant in N and amenable to GPU batching at scale; the trusted-setup cost is amortized by the ceremony recorded in Z_{vk}.

12 Threshold ML-DSA via Pulsar; Why Not Threshold FALCON

12.1 Threshold ML-DSA is achieved by Pulsar (BCC/CEF)

This subsection supersedes the earlier "why not threshold ML-DSA" position of Quasar 3.0. That position — that ML-DSA cannot be thresholdized without either reconstructing the key or relaxing the FIPS rejection bound, and that Quasar therefore uses individual ML-DSA signatures rolled up by Groth16 in place of a threshold — is now obsolete. Threshold ML-DSA, *byte-equal* to FIPS 204, *is* achieved by **Pulsar** (LP-4450 [16], formal companion LP-073 [17]). We restate the two classical obstacles and show how Pulsar's *boundary-cleared nonce transcript with carry elimination* (BCC/CEF) eliminates each without leaking key material.

ML-DSA (FIPS 204, formerly CRYSTALS-Dilithium) uses Fiat–Shamir with Aborts: the signer samples a masking vector $\mathbf{y}$, commits $\mathbf{w} = \mathbf{A}\mathbf{y}$, derives challenge c, forms response $\mathbf{z} = \mathbf{y} + c\mathbf{s}_1$, and runs a *secret-dependent* rejection-and-hint sequence over $\mathbf{w}_0 - c\mathbf{s}_2$ and $c\mathbf{t}_0$.

1. **Rejection sampling.** The abort decision depends on the *combined* response $\mathbf{z} = \sum_i \lambda_i \mathbf{z}_i$ and on $\mathbf{w}_0 - c\mathbf{s}_2$, neither of which a single party can evaluate without the others' shares. *Pulsar's resolution:* move the decision *offline* and make it *message-independent*. A nonce aggregate $\mathbf{w}$ is *boundary-clear* iff every coefficient of $\mathbf{w}$ has centred low bits $|\mathbf{w}_0| < \gamma_2 - 2\beta -$ slack. A boundary-clear nonce guarantees, for *every* admissible challenge c, that $\mathsf{HighBits}(\mathbf{w} - c\mathbf{s}_2) = \mathsf{HighBits}(\mathbf{w}) = \mathbf{w}_1$ and that the FIPS $\mathbf{r}_0$-norm check holds — *without ever forming* $c\mathbf{s}_2$, $c\mathbf{t}_0$, $\mathbf{r}_0$, or $\mathbf{w}_0$. The combined rejection decision becomes a per-nonce predicate evaluated once, offline, over public-after-clearance high bits.
2. **Hint computation.** FIPS verification reconstructs the high bits of $\mathbf{w} - c\mathbf{s}_2$ from a hint $\mathbf{h}$ without knowing $\mathbf{s}_2$; the signer normally computes $\mathbf{h} = \mathsf{MakeHint}(\mathbf{w}_0 - c\mathbf{s}_2 + c\mathbf{t}_0, \mathbf{w}_1)$, which a threshold must either feed from a trusted combiner or compute via the secret-bearing intermediates. *Pulsar's resolution:* recover the hint from *public data only*. The combiner forms $\mathbf{w}' = \mathbf{A}\mathbf{z} - c\mathbf{t}_1 \cdot 2^d$ (exactly the quantity the FIPS verifier forms) and derives $\mathbf{h}$ from $\mathbf{w}'$ and the public $\mathbf{w}_1$ via the standard FIPS $\mathsf{UseHint}$ decision ($\mathsf{FindHint}$): $h_j = 0$ if $\mathsf{HighBits}(\mathbf{w}'_j) = \mathbf{w}_{1,j}$; $h_j = 1$ iff $\mathsf{UseHint}(1, \mathbf{w}'_j) = \mathbf{w}_{1,j}$; otherwise the nonce admits no valid FIPS hint and is consumed and retried. No party reconstructs $c\mathbf{s}_2$ or $c\mathbf{t}_0$.

Verification-equivalence. FIPS 204 Verify computes $\mathbf{w}'_{\text{approx}} = \mathbf{Az} - c\mathbf{t}_1 \cdot 2^d$ and accepts iff $\tilde{c} = H(\mu, \mathsf{UseHint}(\mathbf{h}, \mathbf{w}'_{\text{approx}}))$. Pulsar builds $\mathbf{h}$ via FindHint so that $\mathsf{UseHint}(\mathbf{h}, \mathbf{w}') = \mathbf{w}_1$ by construction; boundary clearance bounds the $c\mathbf{t}_0 - c\mathbf{s}_2$ residual to ± 1 high-bit carries that the hint corrects. The threshold signature therefore verifies byte-for-byte under unmodified FIPS 204 — the same algebra as single-party Sign, with the hint recovered from public data rather than from the secret residual.

Mode restriction. BCC/CEF is sound only where $\|c\mathbf{t}_0\|_\infty \leq \tau \cdot 2^{d-1} < \gamma_2$, which makes the FIPS $c\mathbf{t}_0$ check vacuous and bounds the carry to a single high-bit step. This holds for **ML-DSA-65** ($49 \cdot 4096 < \gamma_2 = 261888$) and **ML-DSA-87** ($60 \cdot 4096 < \gamma_2$), but *not* for ML-DSA-44 ($39 \cdot 4096 > \gamma_2 = 95232$). Pulsar supports ML-DSA-65 and ML-DSA-87 only.

Consensus-native certification. Boundary clearance is certified by a NonceTranscript quorum run by the validator set itself (consensus-native MPC, not a separate signing service), which emits a NonceCert carrying *only* $\mathbf{w}_1 = \mathsf{HighBits}(\mathbf{w})$ and a clearance quorum certificate — never the full commitment $\mathbf{w}$. Online signing is then the Ringtail-style two rounds (bind canonical nonce; open linear $\mathbf{z}$-partial) plus a public-hint finalize, orchestrated inside `luxfi/consensus` as a Quasar cert profile, with the BCC/CEF crypto primitives in `luxfi/pulsar`.

The V13 finding (what NOT to do). An earlier Pulsar construction (`v0.3 AlgebraicAggregate`) broadcast the secret-bearing intermediates $c\mathbf{s}_2$ and $c\mathbf{t}_0$ to a combiner so it could run the FIPS sign-side checks and MakeHint verbatim. Broadcasting $c\mathbf{s}_2$ / $c\mathbf{t}_0$ *leaks long-term key material* — this is the **V13 finding** (`PULSAR-V13-HINT-LEAK`; a sibling `PULSAR-V13-W-LEAK` exposes the full $\mathbf{w}$ in a NonceCert). That path is removed/gated fail-closed; BCC/CEF is the no-leak replacement, and NonceCert carries only $\mathbf{w}_1$ plus a clearance proof. The detailed analysis is LP-4450 [16] (V13 section) and the Pulsar paper LP-073 [17].

Honest status. Pulsar is **review-ready, not production-certified.** Byte-equality of the no-leak path is demonstrated under both the `cloudflare/circl` and `pq-crystals` reference verifiers (ML-DSA-65 and ML-DSA-87), and the boundary-clearance $\Rightarrow$ FindHint $\leftrightarrow$ UseHint correspondence is machine-checked in Lean. Two genuinely novel components — tight small-norm lattice range proofs for the nonce/DKG witnesses, and evaluating HighBits/boundary clearance inside the MPC over secret-shared $\mathbf{w}$ — are *fail-closed pending external review.* Until reviewed verifiers are installed, the production signer refuses the BCC/CEF path and the Quasar composer uses the Corona leg in Pulsar's place. Pulsar's stock FIPS 204 *verifier* is always available; only the threshold *signer* is gated.

Relationship to the Z-Chain ML-DSA rollup. Quasar's individual-ML-DSA-plus-Groth16 lane (§4.3) is retained as a *complementary accountability* mechanism — it proves *which* validators signed (per-validator attribution and slashing evidence) and compresses N signatures to a constant-size on-chain footprint. Pulsar provides the *threshold* property: a single FIPS-204 signature under one group key. The two are orthogonal lanes, not competitors: the rollup gives accountability and footprint, Pulsar gives a genuine threshold seal. Where the earlier text claimed a threshold "cannot" be built, the correct statement is that a byte-equal threshold *can* be built (Pulsar), and Quasar composes both.

12.2 Threshold FALCON

FALCON [13] uses NTRU lattices and a hash-then-sign paradigm with Gaussian sampling via a Gram-Schmidt-like procedure (the "FALCON tree"). Unlike ML-DSA — whose Fiat–Shamir-with-Aborts structure Pulsar thresholdizes via BCC/CEF — FALCON's trapdoor Gaussian

sampling has no analogous public-data reformulation. Thresholdizing FALCON requires distributed Gaussian sampling, which remains an open problem:

1. The FALCON signing key is a short basis for an NTRU lattice. Shamir-sharing this basis and reconstructing Gaussian samples from shares requires parties to jointly sample from a lattice Gaussian without leaking the basis—strictly harder than threshold Fiat–Shamir with Aborts.
2. Known approaches (Ducas and Prest [14]) for distributed NTRU sampling require $O(n^2)$ communication rounds, negating FALCON's performance advantage over ML-DSA.
3. FALCON was not selected as NIST's primary standard (ML-DSA was); it is a secondary standard under FIPS 206 (draft). Building threshold cryptography on a secondary standard adds adoption risk.

Quasar's position. Threshold ML-DSA *is* used, via Pulsar's BCC/CEF construction (byte-equal to FIPS 204, ML-DSA-65 and ML-DSA-87), subject to the honest review-ready status above; until that review lands, the Corona Module-LWE leg fills the threshold PQ niche and individual ML-DSA-plus-Groth16 provides per-validator accountability. Threshold FALCON is not used: distributed trapdoor Gaussian sampling remains an open problem with no byte-equal threshold construction.

13 Activation Roadmap

Quasar 3.0 was developed against a fixed activation calendar. The following milestones describe the December 2025 release; later items (e.g. Quasar 4.0 research) belong to Section 14.

Date (2025)	Milestone	Detail
1 Dec	Devnet freeze	Quasar 3.0 binary frozen on `lux-devnet`; Mode 3 enabled at $n = 21$. Block time 500 ms. Three-lane verification on QuasarGPU enabled by default.
8 Dec	Testnet rollout	`lux-testnet` validators upgrade. Public Z-Chain trusted-setup ceremony closes; vk_Z hash published as `zchain_vk_root` at the testnet genesis.
15 Dec	**Spec freeze**	This paper, the LP-020 LIP, and the soundness companion [15] are tagged. No protocol-level changes after this date except for security fixes.
20 Dec	Final review	Cross-team audit closure: Lean 4 / Tamarin proofs reviewed; Z-Chain prover and Q-Chain DKG verified on testnet for $\geq$ 5,000 blocks with no QuasarCert verification failures.
22 Dec	Mainnet bake	Mainnet binaries deployed to validator operators. Read-only shadow-fork against the previous network with QuasarCert generation enabled (no finality).
25 Dec	**Mainnet activation**	Lux relaunch with Quasar 3.0. Mode $\mu = 3$ (Quantum). All seven chain roots $(P_{\text{val}}, Q_{\text{cer}}, Z_{\text{vk}}, A_{\text{att}}, B_{\text{br}}, M_{\text{mpc}}, F_{\text{fhe}})$ are bound into every QuasarCert from genesis.
27–31 Dec	Post-activation review	Telemetry: per-cert verify latency, Z-Chain prover wall-clock, Corona combiner success rate. Operational runbook published as the `quasar-3.0-operations.md` living document.

Table 12: Quasar 3.0 activation milestones, December 2025.

The activation is deliberate and singular: the December 2025 cadence treats Quasar 3.0 as the long-term consensus baseline for the Lux primary network. Subsequent point releases are

reserved for security fixes; protocol-level evolution (Section 14) tracks under a separate LP and a separate paper.

14 Future Work

Quasar 3.0 is the canonical consensus baseline for the Lux primary network from 25 December 2025 onward. Several research directions sit explicitly *outside* the 3.0 spec and are being evaluated for a prospective Quasar 4.0 release; full specification, theorems, and benchmarks for those directions belong to a separate paper. Topics under active research include conflict-free scheduling at the transaction layer (cf. AFT 2025 [24]), deterministic execution variants such as NEMO [25], RapidLane [26], Chiron [27], ForeSight [28], CSMV [29], versioned-MVCC and vMVCC [30], the Motor multi-leader DAG [31], and Multiverse-style speculative execution [32]. None of these are part of the December 2025 spec or its proofs; they are listed here only to mark the boundary between the activated 3.0 protocol and the open research backlog. Research is underway. The 4.0 spec will appear as a separate paper when ready.

15 Conclusion

Quasar 3.0 is the consensus protocol of the Lux primary network activated on 25 December 2025. Each finalized block carries a *QuasarCert* that aggregates three independent verification lanes: a 48-byte BLS12-381 aggregate signature, a Corona two-round Module-LWE threshold signature rooted on Q-Chain, and a 192-byte Z-Chain Groth16 rollup of N FIPS-204 ML-DSA-65 validator signatures plus a 32-byte public-inputs hash. The third lane is a Z-Chain proof artifact, not a raw per-validator ML-DSA list; this makes the per-block on-chain post-quantum identity-binding cost constant in the validator count.

Every QuasarCert binds the certificate subject—an explicit hash of the chain id, epoch, round, mode, and seven chain-specific roots $(P_{\text{val}}, Q_{\text{cer}}, Z_{\text{vk}}, A_{\text{att}}, B_{\text{br}}, M_{\text{mpc}}, F_{\text{fhe}})$ together with the parent block hash, parent state root, parent execution root, gas limit, and base fee. Cross-chain and cross-epoch replay are structurally impossible (Theorem 6.1); the verifier decides replay locally and statelessly.

Safety holds if at least one cryptographic lane remains secure. Liveness holds under partial synchrony with $f < n/3$ Byzantine validators. The protocol is implemented in Go, deployed across the P/Q/Z/A/B/M/F chain pipeline, and supported by machine-checked proofs in Lean 4 and symbolic verification in Tamarin. The QuasarGPU adapter (LP-132) verifies the three lanes in parallel: 0.5–1.5 ms per QuasarCert at n=21, well inside the 500 ms target block time, and on track for the $\geq$100K TPS sustained per chain.

The post-quantum design is deliberately conservative. We do not claim that BLS12-381 will resist quantum computers, nor that Module-LWE will survive every future cryptanalytic decade. We claim that requiring an adversary to break all three simultaneously—BLS12-381, Corona Module-LWE, and ML-DSA Module-LWE/MSIS, with the Groth16 lane providing on-chain compactness—offers a margin of safety that no single-scheme approach can match. Quasar 3.0 is the answer for December 2025; the 4.0 research backlog is documented separately (Section 14).

References

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

[2] National Institute of Standards and Technology. FIPS 204: Module-Lattice-Based Digital Signature Standard. NIST, August 2024.

[3] M. Mosca. Cybersecurity in an Era with Quantum Computers. *IEEE Security & Privacy*, 2018.

[4] D. Boneh, B. Lynn, and H. Shacham. Short Signatures from the Weil Pairing. In *ASIACRYPT*, 2001.

[5] Lux Industries. Corona: Two-Round Module-LWE Threshold Signatures. Lux Industries, 2026. `github.com/luxfi/corona`. Module-LWE (Ringtail-derived), two-round; builds on the Ringtail line [6]; Q-Chain Feldman/Pedersen DKG.

[6] C. Boschini, D. Kaviani, R. W. F. Lai, G. Malavolta, A. Takahashi, and M. Tibouchi. Ringtail: Practical Two-Round Threshold Signatures from LWE. Cryptology ePrint Archive, Paper 2024/1113; IEEE S&P 2025. Academic two-round lattice threshold; the construction that Corona builds on.

[7] M. Castro and B. Liskov. Practical Byzantine Fault Tolerance. In *OSDI*, 1999.

[8] M. Yin, D. Malkhi, M. K. Reiter, G. G. Gueta, and I. Abraham. HotStuff: BFT Consensus with Linearity and Responsiveness. In *PODC*, 2019.

[9] T. Rocket, M. Yin, K. Sekniqi, R. van Renesse, and E. G. Sirer. Scalable and Probabilistic Leaderless BFT Consensus through Metastability. IPFS preprint, 2019.

[10] J. Groth. On the Size of Pairing-Based Non-Interactive Arguments. In *EUROCRYPT*, 2016.

[11] O. Regev. On Lattices, Learning with Errors, Random Linear Codes, and Cryptography. *J. ACM*, 56(6), 2009.

[12] T. Espitau, M. Tibouchi, and A. Wallet. Practical Fiat–Shamir with Aborts Threshold Signatures from Lattices. *IACR ePrint*, 2024.

[13] 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 PQC Round 3 submission, 2020.

[14] L. Ducas and T. Prest. Fast Fourier Orthogonalization. In *ISSAC*, 2016.

[15] Z. Kelling. QuasarCert Soundness: Proof Sketch with Circuit Analysis, Adaptive Corruption, Setup, and Parameter Tightness. Lux Industries technical note, December 2025. `proofs/quasar-cert-soundness.tex`.

[16] Lux Industries. LP-4450: Pulsar — Boundary-Cleared Module-LWE Threshold ML-DSA (FIPS 204, byte-equal). Lux Improvement Proposal, 2026. `github.com/luxfi/pulsar`.

[17] Z. Kelling. LP-073: Pulsar — Module-LWE Threshold ML-DSA for Permissionless Validator Sets. Lux Industries, Inc., 2026.

[18] Lux Industries. LP-132: QuasarGPU Verification Substrate. Lux Improvement Proposal, December 2025.

[19] Lux Industries. ZAP: A Zero-Allocation Wire Protocol for Consensus. Lux Industries technical paper, 2025.

[20] Lux Industries. Z-Chain: Groth16 Rollup Substrate. Lux Industries technical paper, 2025.

[21] Lux Industries. A-Chain: Unified Attestation Chain with TEE-Verified Proof of Execution. Lux Industries technical paper, 2025.

[22] Lux Industries. Nova: Linear-Chain Quasar Engine (Photon → Wave → Focus → Ray). Lux Industries technical paper, 2025.

[23] Lux Industries. Nebula: DAG Quasar Engine (Flare → Horizon → Field). Lux Industries technical paper, 2025.

[24] AFT 2025 authors (TBC). Conflict Specifications for Concurrent Smart-Contract Execution. In *Advances in Financial Technologies (AFT)*, 2025.

[25] Authors (TBC). NEMO: Conflict-Free Scheduling for Blockchain Execution. Preprint, 2025.

[26] Authors (TBC). RapidLane: Lane-Parallel Smart-Contract Execution. Preprint, 2025.

[27] Authors (TBC). Chiron: A Multi-Stream Concurrency Layer for Smart Contracts. Preprint, 2025.

[28] Authors (TBC). ForeSight: Speculative Read-Set Hinting for Optimistic Execution. Preprint, 2025.

[29] Authors (TBC). CSMV: Conflict-Set Multi-Version Concurrency Control. Preprint, 2025.

[30] Authors (TBC). vMVCC: Versioned Multi-Version Concurrency Control for Blockchain State. Preprint, 2025.

[31] Authors (TBC). Motor: A Multi-Leader DAG with Heterogeneous Block Roles. Preprint, 2025.

[32] Authors (TBC). Multiverse: Speculative Branching Execution for Smart Contracts. Preprint, 2025.

Reproducibility

Source code. `github.com/luxfi/consensus` (commit `TBD`); wire protocol at `github.com/luxfi/zap`; Corona at `github.com/luxfi/corona`.
Build. `go build ./...` and `go test ./...` at the freeze tag `v0.1.0-rc1-pq-consensus-freeze`.
Hardware tested. TBD (commodity x86_64 Linux validators).
Standards referenced. FIPS 204 (ML-DSA-65), FIPS 206 (draft, threshold context), BLS12-381 (IETF draft), NIST IR 8214C (MPTC).
Contact. security@lux.network.