Quantum: Privacy-Preserving Post-Quantum DAG Protocol

Research Design Draft 0.5.4-research and Security Requirements

Phexora AI · phexora.ai

2026-09-05

Abstract

Quantum is a research design for a private note-based DAG protocol intended to combine post-quantum security, privacy and network anonymity by default, at least 1,000 accepted layer-1 transactions per second, independently bounded operation and recovery, and contestable block production with an explicitly reviewed long-run security budget. This manuscript defines the non-negotiable requirements, threat and claim boundaries, candidate architecture, and blocking evidence gates. It does not report a conformant implementation, completed security proof, external audit, testnet, or production-ready protocol. The initial product boundary is deliberately narrow: private post-quantum digital cash and settlement with bounded payment policies, not a general-purpose smart-contract platform.

Keywords: post-quantum cryptography; privacy-preserving ledger; DAG consensus; zero-knowledge proofs; protocol research.

Document status

This document defines the non-negotiable security, privacy, scalability, operability, recovery, producer-contestability, and economic-review requirements for Quantum and a candidate architecture for meeting them. It is a **research design draft**, not a completed protocol specification. There is no conformant implementation, testnet, security proof, external audit, or production network at this revision.

The legacy filename contains "v1" so that existing links remain valid. The normative document version is 0.5.4-research.

The words **MUST**, **MUST NOT**, **REQUIRED**, **SHOULD**, and **MAY** describe research acceptance criteria. They do not imply that an implementation already satisfies them.

Non-negotiable product requirements

Quantum is releasable only if all five properties hold together:

1. **Post-quantum security**: every security-critical primitive and their composition meet an end-to-end post-quantum security target of at least 128 bits against quantum polynomial-time adversaries.
2. **Privacy and anonymity by default**: the ledger has no transparent transaction mode; sender, recipient, amount, and transaction-graph relationships are hidden within a declared threat model, and the network layer does not expose a simple transaction-to-origin mapping.
3. **Scalability**: the complete protocol, including proof generation, verification, state updates, networking, and storage, sustains at least 1,000 accepted layer-1 transactions per second under a published, reproducible workload.
4. **Independent operability and recovery**: executing validation, pruning, restart, bootstrap, and recovery remain inside a frozen, independently obtainable resource profile without reliance on a trusted sole provider.
5. **Contestable production and reviewed long-run security funding**: pooled miners can construct their own block templates, ordering and censorship incentives pass explicit gates, and the exact monetary policy is supported by independently reviewed accounting and economic evidence.

If any requirement cannot be met, the design MUST be changed or the project MUST stop before a production launch. Classical cryptographic fallbacks, optional transparent transactions, and benchmark shortcuts are not permitted.

1. Scope and claim boundary

This document covers:

- a private note-based UTXO transaction model;
- post-quantum authorization and recipient encryption;
- transparent zero-knowledge validity proofs;
- deterministic DAG consensus and state ordering;
- network-layer anonymity requirements;
- supply integrity, rewards, serialization, current-data availability, recovery, producer contestability, and resource limits;
- the evidence required before testnet or production claims.

This document does not claim that combining standardized components automatically produces a secure protocol. Composition, implementation, side channels, consensus integration, and network metadata require separate analysis.

1.1 Initial product boundary and sequencing

Quantum's initial product target is **private post-quantum digital cash and settlement** for people, organisations, and autonomous software agents. The core profile consists of native-QTM transfers, private rewards, wallet recovery, selective user-controlled disclosure, and the smallest deterministic payment-policy surface that passes the same privacy, supply, and performance gates as an ordinary transfer.

The initial profile is not a general-purpose application platform. Arbitrary smart contracts, an EVM/SVM-compatible runtime, general DeFi state, permissionless asset issuance, bridges, and cross-chain message execution are outside the core release scope. They MUST NOT be assumed by the security proof or used to defer a core release requirement. Any later capability requires a separately versioned threat model, validity relation, resource budget, and release decision.

Research MUST proceed through explicit stop/go boundaries:

1. freeze the threat model and common encoding rules, then the task-owned commitment, note, transaction, and proof interfaces before their consumers; run the bounded T006 receiving/transport feasibility screen before T305;
2. implement and benchmark the complete representative private transaction described in Section 9.3;
3. stop or revise the cryptographic profile if that feasibility gate fails;
4. only then integrate wallets, private state, DAG consensus, transport, and full-node operation under the operability, availability, producer, and monetary gates;
5. add bounded payment policies or interoperability adapters only after their additional proof and trust assumptions pass separate gates.

This sequencing does not weaken the five joint production requirements. It prevents node and ecosystem work from concealing a cryptographic design that cannot meet them.

The versioned decentralisation, operability, and security-budget decision at decision revision 0.3.3-research records the boundaries incorporated into this 0.5.4 design, its rejected claims, candidate mechanisms, and human approval points. The decision record has an independent version lifecycle and does not mark a requirement or gate as passed.

1.2 Status vocabulary

Status	Meaning
Requirement	Mandatory property of any release
Selected standard	Exact external standard selected; integration still requires validation
Candidate	Design direction that may change after analysis
Blocking research gate	No implementation or release may rely on this area until its deliverables pass
Verified	Reserved for independently reproducible evidence; no subsystem has this status yet

1.3 Current subsystem status

Subsystem	Current status	Required next evidence
SHAKE256	Selected standard: FIPS 202	Domain-separation vectors and implementation KATs
Spend authorisation	Comparative research gate; FIPS 205 256f incumbent and required 256s comparator	Stateful/stateless comparison, KATs, state-failure analysis, side-channel review, proof-system cost
ML-KEM	Selected standard: FIPS 203, ML-KEM-1024	FIPS KATs and an authenticated composition
Note commitment	Blocking research gate	Exact construction, parameters, reduction, estimator report, review
Zero-knowledge STARK	Blocking research gate	Exact field/transcript/FRI/masking profile and soundness analysis
Representative transaction proof	Blocking research gate	Complete 2-input/2-output prototype and reproducible feasibility report
Early receiving/transport feasibility	Blocking research screen; not run	T006 client scan, offline catch-up and anonymous-transport cost envelope before T305
DAG consensus/state	Blocking research gate	Exact GHOSTDAG profile, deterministic ordering, DAA, state proof
Network anonymity	Blocking research gate	Exact protocol and analysis against the stated observer
Performance	Blocking research gate	End-to-end prototype and reproducible target benchmark
Validator operability	Blocking research gate	Frozen profile and paired G7/G10 evidence
Current data/recovery	Blocking research gate	Role-specific availability, bootstrap, withholding, and recovery evidence
Mining templates/incentives	Blocking research gate	Miner-controlled template path, pooled-mining comparator, and incentive analysis
PoW hardware contestability	Comparative research gate	Candidate measurements before PoW selection
Upgrade governance	Blocking research gate	Exact post-quantum authorisation and activation state machine

Subsystem	Current status	Required next evidence
Economics/genesis	Provisional	Bounded dynamic-fee comparator, DAA-score reward rules, selected monetary-invariant proof, canonical genesis bytes

2. Normative requirements

R1 — Private ledger

- R1.1 Every ordinary transfer MUST be private. There MUST be no transparent amount, transparent recipient, or optional transparent transaction type.
- R1.2 Public ledger data MUST NOT reveal the transferred values, recipient addresses, spend authorization keys, the recipient key targeted by an encrypted output, or a direct input-to-output link.
- R1.3 A valid nullifier MUST prevent a second spend without identifying the note commitment from which it was derived.
- R1.4 The proof system MUST establish authorization, note membership, output correctness, and exact integer balance without revealing private witness data.
- R1.5 User-controlled disclosure capabilities MAY reveal selected wallet or transaction data. The profile MUST distinguish incoming-view, full-wallet, transaction-specific, and auditor-scoped disclosure as defined in Section 5.7.
- R1.6 Disclosure MUST be explicit, least-privilege, and non-global. It MUST NOT create a protocol backdoor, mandatory universal view key, or privacy loss for notes and users outside the granted scope.
- R1.7 Distinct accepted note instances MUST remain independently spendable even when a malicious sender repeats note plaintext, randomness, or a commitment in different transactions. T302 MUST bind nullifiers to a unique authenticated note instance as specified in Section 5.6.1; local duplicate rejection alone is insufficient.

R2 — Network anonymity

- R2.1 Transaction transport MUST be unlinkable to a sender network endpoint within the approved network threat model.
- R2.2 The protocol MUST address timing, volume, route, retry, and peer-selection metadata. Payload encryption alone is insufficient.
- R2.3 Wallet identity keys, note keys, and transaction authorization keys MUST NOT be reused as P2P identities.
- R2.4 Dandelion++ MAY be evaluated as one layer, but MUST NOT by itself be treated as proof against a global observer. Cover traffic, mixing or onion routing, and active-intersection resistance require explicit evaluation.

"Anonymous" in a release claim means that R1 and R2 have passed for the published adversary model. No protocol can hide activity from a compromised endpoint, a voluntarily disclosed view key, or off-chain information supplied by the user; those boundaries MUST be stated with every claim.

R3 — End-to-end post-quantum security

- R3.1 The minimum composed security target is 128 post-quantum bits after multi-user, multi-target, and protocol-lifetime losses.

- R3.2 Authorization, commitments, proofs, key establishment, hashing, authenticated encryption, randomness generation, storage encryption, and upgrade authentication MUST be included in the analysis.
- R3.3 A classical-only handshake or signature MUST NOT be accepted as a fallback.
- R3.4 Every primitive MUST have an exact algorithm identifier, parameter set, canonical encoding, test-vector source, domain-separation rule, required security property, and output length justified by the composed analysis.
- R3.5 A "generic STARK", "lattice-based commitment", or "hybrid" label is not evidence of post-quantum security.
- R3.6 T001 and T204 MUST derive the required wallet master-secret entropy from the approved multi-user, multi-target, and protocol-lifetime bound. A 256-bit BIP-39 mnemonic is the minimum compatibility profile, not automatic evidence that the composed R3 target remains satisfied. If the derived requirement exceeds 256 bits, the wallet MUST add independently generated, domain-separated secret entropy or use a separately reviewed higher-entropy seed format, with exact recovery and composition rules. Lower-entropy legacy mnemonic imports MAY be supported only under an explicit degraded-security profile, MUST present a user-visible warning before use, and MUST NOT inherit the 128-bit post-quantum claim.

R4 — Authorization and supply integrity

- R4.1 Every input value used by the balance equation MUST be bound to the opening of an existing note commitment.
- R4.2 Every output value used by the balance equation MUST be bound to the opening of the corresponding new note commitment.
- R4.3 Balance MUST be proved as bounded integer arithmetic. Equality in a finite field alone is forbidden.
- R4.4 Only a consensus-authorized reward transaction MAY create value.
- R4.5 Cumulative gross issuance MUST never exceed 21,000,000 QTM, represented in a fixed base unit and checked without floating-point arithmetic.
- R4.6 Fees MUST be accounted for as value transferred from accepted ordinary transactions, not as newly issued value. Only the subsidy portion of an authorized reward transition may increase cumulative issuance.
- R4.7 Reward outputs MUST equal explicitly accounted claimed fees plus claimed subsidy. Claimed fees MUST NOT exceed the matured balance of the canonical miner-fee pool; claimed subsidy MUST NOT exceed the consensus-authorized subsidy. Miner-eligible fees that are neither paid nor explicitly burned by the selected policy MUST remain a consensus liability in the fee pool rather than disappearing.
- R4.8 Under the current lifetime gross-issuance cap, burned existing value MUST NOT restore issuance headroom. Any outstanding-supply cap, burn-and- reissuance rule, tail emission, or replacement cap semantic requires a separately versioned product decision and public monetary statement.
- R4.9 The fee-pool balance MUST count toward outstanding supply but MUST NOT count as gross issuance. Its maturity buckets, claims, burns, rounding, remainders, and reorganisation reversal MUST be deterministic and atomic so no fee-pool amount can be lost, recreated, paid twice, or burned implicitly.
- R4.10 Every monetary quantity and intermediate MUST be a non-negative integer. The selected profile MUST fix serialized field widths, accumulator widths, operand maxima, and checked multiplication/addition bounds before T504. An overflow, underflow, wrap, saturation, negative value, zero divisor, or value outside those bounds MUST invalidate the transition.

R5 — Deterministic consensus safety

- R5.1 All honest nodes receiving the same valid DAG and state MUST derive the same ordering, accepted transaction set, difficulty, rewards, and state root.
- R5.2 Consensus calculations MUST use deterministic integer arithmetic. Local wall-clock time and floating-point arithmetic MUST NOT determine a consensus result.
- R5.3 Header-declared difficulty is advisory data only. Validation MUST recompute the expected target from the agreed DAG history and reject any mismatch.
- R5.4 Concurrent nullifier conflicts MUST resolve through one canonical ordering and atomic state transition.
- R5.5 The mining and reward profile MUST permit a miner-controlled full node or Job Declarator to construct and commit to its own candidate transaction set while a separate pool or share aggregator accounts for work.
- R5.6 No release claim may assume that public fees make transaction selection mechanical or that privacy eliminates extractable ordering value. The final profile MUST analyse fee, conflict, timing, censorship, duplicate-inclusion, and reorganisation incentives under its exact public fields and DAG rules.
- R5.7 The PoW algorithm MUST NOT be selected or described as ASIC-resistant before the hardware-contestability gate compares commodity, specialised, supply-chain, cloud, botnet, verification, energy, and block-rate costs.

R6 — Scalability

- R6.1 The release target is at least 1,000 accepted layer-1 transactions per second, sustained for at least 24 hours on a geographically distributed testbed.
- R6.2 The benchmark MUST include proof verification, signature checks, consensus ordering, nullifier/state updates, propagation, and rejected conflicts. Prevalidated or empty transactions do not count.
- R6.3 Reference hardware, topology, software revision, workload, proof sizes, bandwidth, storage growth, latency percentiles, and failure rate MUST be published.
- R6.4 Every full validator MUST either process the complete accepted state transition stream at the target rate or rely on a separately proven consensus-secure partitioning design. Parallel block production alone does not reduce validator work.
- R6.5 Pruning and snapshots MAY reduce operational storage, but archival requirements and trustless recovery MUST remain explicit.

R7 — Transparent setup and upgrade safety

- R7.1 Transaction validity MUST NOT depend on a secret trusted-setup artifact.
- R7.2 Proof aggregation or recursion is allowed only if the outer proof preserves the post-quantum and no-secret-setup requirements.
- R7.3 Protocol upgrades MUST be versioned, domain-separated, replay-safe, and authenticated by the post-quantum governance and activation mechanism owned by T510 and defined before launch.
- R7.4 T306 MUST decide only whether the selected proof system provides a usable transparent accumulation or recursion capability inside the R3 and R9 budgets. It MUST NOT freeze the exact Quantum consensus/state relation before T405, T503, and T504 define bounded-policy semantics, canonical ordering, and monetary state.
- R7.5 If T306 selects that capability, T505 MUST define the exact Quantum relation binding canonical consensus ordering, complete state transitions, rewards, fees, burns, issuance, protocol version, and checkpoint identity.
- R7.6 Proof soundness MUST NOT be presented as evidence of current-data availability, state availability, wallet-witness availability, canonical history selection, censorship resistance, or archive recovery.

- R7.7 A concentrated prover or aggregator MUST be modelled as able to delay, withhold, censor, or selectively prove consensus-valid histories. The final profile MUST state a tested degraded or alternative path.
- R7.8 T510 does not prove governance decentralisation or prevent capture by a formally authorised quorum. Its threat model MUST analyse principal concentration, bribery, collusion, strategic abstention, veto power, client-distribution control, and the boundary between deterministic protocol activation and voluntary social adoption.
- R7.9 T510 MUST publish a consensus-parameter registry classifying historical invariants, normal-upgrade-only protected parameters, and parameters that an emergency transition may change. No normal or emergency upgrade may rewrite cumulative issuance, erase or reclassify an accrued fee-pool liability, or reinterpret an accepted fee. An emergency transition MUST NOT change the 21,000,000 QTM cap semantic, issuance schedule, fee allocation, or value-conservation equations. A prospective monetary-policy change requires a new T506 campaign, the R12.5 approvals, a normal versioned activation path, and new dependent evidence; it cannot be disguised as incident response.
- R7.10 Upgrade activation MUST preserve the spent/unspent identity of every existing note. Changing the active transaction or proof version MUST NOT create a fresh nullifier for an already spent note. T204/T302/T510 MUST specify historical-note spending, wallet recovery, and any explicit migration before activation; an unsupported historical authorisation profile MUST NOT silently strand funds or be replaced by a weaker signature.

R8 — Verifiability

- R8.1 Consensus-critical behavior MUST have canonical byte encodings and cross-implementation vectors.
- R8.2 Security arguments MUST state assumptions, reductions, concrete parameters, and composition losses.
- R8.3 Test results MUST distinguish analytical proof, formal verification, implementation tests, statistical diagnostics, benchmarks, and external review. One category MUST NOT be presented as another.
- R8.4 Production claims require at least two independent implementations, public interoperability vectors, and independent cryptographic and consensus review.
- R8.5 Every capability claim MUST identify its versioned profile and evidence gate. A base-layer proof MUST NOT be presented as evidence for an external bridge, general-purpose application runtime, or regulatory compliance.

R9 — Independent operability

- R9.1 Every named release candidate and benchmark campaign MUST freeze one executing-validator profile before results are interpreted. It MUST state CPU, memory, sustained disk I/O and endurance, operational storage, network, power measurement, accelerator policy, bootstrap limits, retail-availability method, and cost method.
- R9.2 The profile is immutable for that release candidate and campaign. Any change creates a new profile and requires new independently reviewed evidence; it MUST NOT reclassify the prior campaign.
- R9.3 Profile ceilings MUST be justified with dated evidence for hardware and connectivity independently obtainable in multiple regions. This document does not fabricate numeric ceilings before T005.
- R9.4 Executing-validator, succinct-verifier, producer, prover, state-provider, archive-provider, and pool/share-aggregator resources MUST be reported separately. Required work MUST NOT be hidden outside the role being claimed.
- R9.5 G7 and G10 MUST pass as a pair. Reaching R6.1 by exceeding the frozen profile is an operability failure and therefore a release failure.

R10 — Current data availability and recovery

- R10.1 An executing validator MUST obtain and validate the complete current block body, proof representation, and data required by the selected state- transition profile before accepting a block. Sampling alone is not executing validation.
- R10.2 Consensus commitments MUST bind the exact current transaction, proof, state-transition, ordering, and availability representations used by the selected profile.
- R10.3 Pruning and authenticated snapshots MAY bound operational storage, but validator bootstrap, state recovery, wallet recovery, and reconstruction of protocol-required data MUST NOT require a trusted sole archive, snapshot signer, state provider, foundation, or company.
- R10.4 The selected recovery design MUST specify encoding, commitments, reconstruction thresholds, repair, retention and incentive assumptions, selective withholding, and eclipse behavior. Storage proofs or erasure coding labels alone are not evidence.
- R10.5 A public protocol cannot prevent third parties from retaining complete public history. Archive distribution MUST NOT be described as a privacy guarantee. First-seen, IP, peer-route, RPC, and operator metadata remain inside the R2 network-privacy threat model.

R11 — Producer contestability and transaction selection

- R11.1 The profile MUST distinguish the mining device, miner-controlled Template Provider/Job Declarator, block producer, pool/share aggregator, and payout mechanism.
- R11.2 A conforming miner MUST be able to construct its own candidate template, bind pooled work to it, and publish a valid found block without delegating transaction selection to the pool.
- R11.3 A non-custodial pooled-mining design MUST be implemented as a comparator. Its latency, variance, bandwidth, payout, reward-privacy, and revenue overhead MUST be measured against pool-selected and custom-job profiles.
- R11.4 The final ordering profile MUST pass adversarial analysis for fees, fee density, nullifier conflicts, transaction/proof size, anchors and expiry, arrival time, transaction-identifier grinding, duplicates, self-fees, censorship, pool formation, and reorganisations.
- R11.5 Quantum MUST NOT claim to be MEV-free, censorship-proof, pool- decentralised, or ASIC-resistant while the corresponding G12 evidence is absent.
- R11.6 Before T507–T509 results are interpreted, T005 MUST freeze the named metrics, adversarial scenarios, pass/fail thresholds, and STOP conditions for custom-template latency and revenue penalty, direct-publication success, inclusion delay under defined censorship, specialised-hardware efficiency, and manufacturer, fabrication, supplier, cloud, and pool concentration.

R12 — Monetary policy and security budget

- R12.1 The current monetary requirement is a 21,000,000 QTM lifetime gross- issuance cap. The exact finite schedule remains provisional; no perpetual tail subsidy or burn-reissuance headroom is selected at this revision.
- R12.2 Before T504 implementation, T506 MUST compare the current lifetime-cap and long-run fee-funding path against explicitly worded alternatives, including tail emission and an outstanding-supply cap with burn/reissuance.
- R12.3 Every candidate MUST use exact integer accounting and publish its cap semantic, issuance schedule, fee/burn rule, supply trajectory, miner-revenue assumptions, entry/exit behavior, concentration model, and sensitivity to price, fees, hardware, energy, and rental markets.
- R12.4 A positive token-denominated subsidy does not establish adequate attack cost; a finite cap does not establish an adequate future fee market. G13 requires independent monetary-economics evidence and a signed product decision under stated assumptions.

- R12.5 Any change to cap semantics or the public 21-million statement MUST be approved on the exact version by a named Product Owner, consensus reviewer, monetary-economics reviewer, and Legal Counsel. Engineering review alone is insufficient.
- R12.6 Before candidate results are interpreted, T506 MUST pre-register the scenario sets, metrics, pass/fail thresholds, and STOP conditions used to compare monetary policies. The registration MUST cover fee and miner-revenue scenarios, price and energy shocks, miner entry/exit and hashrate response, security-expenditure or attack-cost proxies, and concentration. Thresholds selected after viewing candidate results are invalid evidence.
- R12.7 The lifetime-cap comparison MUST include a bounded dynamic-fee candidate that derives a common required fee rate for each fee epoch from canonical DAG state finalised before that epoch. It MUST NOT depend on a node's mempool, local arrival order, wall clock, current candidate block, fiat price, energy price, or another external oracle.
- R12.8 The candidate MUST separate a resource rate from a security rate. The resource rate prices canonical transaction weight and network utilisation; the security rate responds to a pre-registered token-denominated sustainable- miner-budget objective based only on newly miner-eligible fees and authorised subsidy, not burned fees, other allocations, or withdrawals from a prior fee- pool balance. Transaction weight, fee-rate scale, integer rounding, minimum and maximum rates, update lag, window length, and per-window change bounds MUST be exact consensus parameters before the candidate is tested.
- R12.9 Every transaction evaluated under the candidate MUST commit to its fee epoch and pay at least the deterministic required fee for its canonical weight. The profile MUST define an exact expiry and epoch-admission or grace rule so a rate change cannot reinterpret a signed transaction. An optional priority amount or fee class is permitted only if its public encoding, bounds, selection effect, and privacy impact are frozen by T506 and subsequently pass T508.
- R12.10 A low or zero accepted transaction weight MUST follow one explicit bounded controller transition; it MUST NOT cause division by zero, an unbounded fee, additional issuance, restoration of issuance headroom, or an implicit monetary-policy change. An unmet revenue objective is an observable underfunded result, not evidence that miners are adequately paid.
- R12.11 T506 MUST compare direct payout, partial or full delayed pooling, partial payout plus burn, and separate resource-fee and security-fee treatment, then freeze one exact allocation before T504. Every canonically accepted fee MUST enter exactly one canonical acceptance interval on the selected history. A reorganisation MUST atomically reverse the old assignment before the transaction may be assigned exactly once on the new selected history. T508 MUST adversarially test the frozen rule for omission, self-fee, duplicate-inclusion, fee-sniping, controller manipulation, censorship, reorganisation, and pool-formation incentives; it MUST NOT revise that rule in place.
- R12.12 The dynamic-fee candidate MAY reduce miner-revenue variance or adapt user prices to measured use. It MUST NOT be described as guaranteeing a real-world security budget: no transaction demand produces no fee revenue, and a token-denominated target does not establish fiat-denominated attack cost. G13 MUST fail if the candidate misses a pre-registered security or user-cost threshold in any applicable scenario.
- R12.13 Every canonically accepted total fee MUST be decomposed both by charge class—resource, security, and optional priority—and by destination—new miner- eligible fee-pool value, explicit burn on acceptance, and any exact non-miner output. The two sums MUST be equal. Only new miner-eligible fees may enter the fee-derived part of sustainable miner budget; only matured claimable miner- fee-pool value may enter the fee-derived part of payout capacity.

3. Threat model

The final security profile MUST define exact advantage games and corruption thresholds. The minimum research model includes:

- a quantum polynomial-time cryptographic adversary with adaptive chosen-message and chosen-ciphertext capabilities where applicable;
- malicious transaction creators, provers, miners, Template Providers, Job Declarators, pools/share aggregators, peers, state/witness providers, snapshot providers, and archive/recovery providers;
- adaptive network delay, eclipse and Sybil attempts, packet observation, transaction injection, and intersection analysis;
- a global passive network observer for the anonymity target, plus explicitly enumerated active attacks;
- long-term ledger retention and later cryptanalytic improvement;
- recipient-key inference from encrypted-note payloads, including multi-user, chosen-key, chosen-ciphertext, and cross-output correlation attacks;
- crashes, reordering, duplicate delivery, and recovery from snapshots;
- current-data withholding, selective archive withholding, corrupted recovery shares, eclipse during bootstrap, and prover or state-provider outage;
- fee, conflicting-nullifier, timing, duplicate-inclusion, censorship, pool-formation, and reorganisation incentives under hidden transaction semantics;
- hardware, manufacturing, energy, capital, pool, prover, state, archive, and network concentration; and
- side-channel attackers against wallet and validator implementations.

The consensus profile MUST separately state the assumed adversarial work fraction, network-delay model, liveness conditions, and finality rule. These cannot be inferred from the GHOSTDAG name.

Out of scope for a cryptographic anonymity guarantee are compromised endpoints, malicious operating systems, coerced key disclosure, voluntarily published view keys, and identifying off-chain behavior. Implementations still SHOULD minimize the damage of these events. Retained IP, peer-route, first-seen, RPC, and operator-log metadata is a separate compulsion surface even when all ledger payloads remain cryptographically private.

4. Cryptographic profile

4.1 Hash and domain separation

The candidate hash primitive is SHAKE256 from NIST FIPS 202.

For research vectors, define:

```
QH(tag, message, output_length) =
    SHAKE256(
        ASCII("QTM-RD-0.2") ||
        LE16(length(tag)) || ASCII(tag) ||
        LE64(length(message)) || message ||
        LE32(output_length),
        output_length
    )
```

QTM-RD-0.2 is an independently versioned research-vector domain; it does not track the manuscript version. Changing it invalidates the affected vectors and requires a versioned domain-registry decision. It is not the final consensus protocol version.

Tags MUST be non-empty printable ASCII, unique by purpose, and registered in the final protocol profile. Output length is in bytes. Decoders MUST reject unknown protocol versions instead of guessing a legacy rule.

Research vectors:

Vector	Tag	Message hex	Output length
A	test	empty	32
B	txid	00010203	32
C	empty-leaf	empty	32

SHAKE256 output hex, listed separately to preserve the complete values in print:

```
A: c03ab74639696f42275d889eb3ba7753a4effc561f813c67fd61d06c735f7f78
B: b22782c3a5291412ca5bd8cf85edb47aca9d50d4bf845df9e76364bb23dbc13b
C: 432aa478b2724c19a5ea7b5c17f0c983001de15b3edbb347dfcd13e7fa2875a3
```

These vectors validate this wrapper only; they are not a security proof or a substitute for FIPS 202 KATs. Their 32-byte output is a wrapper test parameter, not an approved consensus digest length. Every collision-dependent use MUST derive its output length from the R3 quantum, multi-target, and protocol-lifetime analysis before that use is frozen, including the time/memory models in generic quantum collision research.

4.2 Transaction authorization

The current stateless incumbent is SLH-DSA-SHAKE-256f, with SLH-DSA-SHAKE-256s as a required standardised comparator, from NIST FIPS 205. The old name SPHINCS+ describes the design lineage; protocol identifiers and public claims MUST use the standardized name SLH-DSA.

This parameter profile is an incumbent under test, not a completed protocol decision. TzEL already demonstrates the high-level alternative of WOTS-like one-time authorisation under an XMSS-style tree inside a private-payment STARK. T202 and T305 MUST therefore compare the incumbent against an independently specified TzEL-shaped baseline and, if its controlled state and key-generation requirements fit the wallet model, one exact stateful profile from NIST SP 800-208. No TzEL code may be reused without compatible permission and documented legal review.

Both FIPS 205 profiles are category 5; their signature encodings contain 49,856 and 29,792 bytes respectively. These are witness-size inputs, not measurements of proving time or public proof size. T202/T305 MUST compare complete signing, in-proof verification, memory, and client latency for both profiles. A failure of 256f alone MUST NOT be reported as failure of standardised stateless authorisation. The limited-use profiles in NIST SP 800-230 initial public draft MAY form a separately labelled exploratory arm. As checked on 2026-09-05, that draft is not final and requires at most 2^24 signatures per key across all devices, retries, and restored backups. Applicability and enforcement of that lifetime limit require review before measurement or profile selection.

The signed message MUST be a dedicated transaction authorization digest that binds at least:

- protocol version and chain identifier;
- anchor and state context;
- all nullifiers and output commitments in canonical order;
- encrypted-note digests;
- public fee, explicit fee context, expiry, and transaction flags;
- authorisation profile; and
- the proof policy that permits the selected individual/aggregate representations, as defined in Section 6.1.

The digest MUST follow Section 6.1's acyclic construction. A concrete proof hash, aggregate identifier, or block position produced after signing is bound by the proof envelope and block commitments, never required as an input to the signature whose verification that proof contains.

The final profile MUST pin the FIPS 205 interface, context string, randomness mode, key encoding, signature encoding, KATs, and failure behavior. A profile-specific authorization public key and signature MAY remain private witness data only if the zero-knowledge circuit verifies

them and the note commitment binds the authorized key. Operational signing state remains wallet-private and is covered by state-management evidence rather than assumed to be validated by the proof. The cost and soundness of in-proof verification are a blocking gate.

Every stateful comparison profile MUST additionally pin key-index allocation, crash consistency, backup/restore rollback behavior, concurrency, exhaustion, recovery, and state desynchronisation. The stateless profile may be retained only if it meets every frozen system requirement and its pre-registered security, interoperability, or state-management benefit materially justifies its measured overhead. A stateful profile may be selected only through a versioned specification change and independent review; it is not a hidden fallback.

FIPS 205 notes that applications needing message-bound signatures must account for the collision cost of H_msg or apply an appropriate reviewed transformation. The transaction-authorization analysis MUST decide whether that property is required, include quantum and multi-target losses, and either show that the composed R3 target remains satisfied or select a reviewed mitigation.

4.3 Recipient key encapsulation

The selected KEM candidate is ML-KEM-1024 from NIST FIPS 203, including applicable NIST errata and KATs.

ML-KEM is not an authenticated transport protocol. The final note-encryption and P2P profiles MUST specify KEM composition, transcript binding, key derivation, authenticated encryption, nonce rules, replay protection, identity separation, and failure behavior. An invented "Noise" pattern name MUST NOT be used unless a real, reviewed Noise extension defines the same messages and security properties. The published Noise Protocol Framework is Diffie–Hellman based and does not itself define an ML-KEM handshake.

FIPS 203 does not by itself establish receiver anonymity or recipient-key privacy. The note-encryption profile MUST define and meet an explicit game in which a public encrypted output does not reveal which eligible recipient key was used, including under multi-user, chosen-key, chosen-ciphertext, and cross-output correlation attacks. Any scanning tag or trial-decryption shortcut is part of that disclosure analysis. Link authentication for P2P channels and receiver anonymity for note delivery are separate properties; the final profile MUST cite and instantiate a reviewed definition such as the anonymous-KEM property or a stronger application-appropriate game.

4.4 Commitment construction — blocking gate

No commitment scheme is selected at this revision. The previous square-matrix formula and four-limb encoding were removed because they did not establish a binding, hiding, carry-safe, homomorphic commitment.

The required interface is:

```
Setup(security_profile) -> public_parameters
Commit(public_parameters, note_plaintext, randomness) -> commitment
Open(public_parameters, commitment, note_plaintext, randomness) -> boolean
```

The committed note plaintext MUST bind:

- immutable note-creation profile and asset identifier;
- 64-bit base-unit value;
- spend-authorization key digest;
- nullifier-key digest;
- recipient/diversifier data required by the final address scheme;
- creation-domain data needed to prevent cross-chain or cross-version reuse.

The selected construction MUST provide correctness, quantum computational hiding and binding, canonical encodings, domain separation, unbiased randomness sampling, parameter-generation rules, and concrete security at the composed target. If a Module-LWE/Module-SIS

construction such as the BDLOP line of work is selected, the actual construction and parameter analysis—not the family name—must be reviewed. Byte reduction modulo a sampler range is forbidden unless rejection sampling removes bias.

Homomorphism is not required: exact conservation is proved inside the transaction validity relation. If homomorphism is later added, carry semantics and all algebraic assumptions require a separate proof.

4.5 Zero-knowledge proof system — blocking gate

The candidate family is a transparent hash-based STARK, informed by the original STARK work and DEEP-FRI. The final profile MUST pin:

- base and extension fields and their encodings;
- AIR constraints and maximum degrees;
- trace padding and public-input encoding;
- Fiat–Shamir transcript order and every domain tag;
- FRI variant, blowup, query count, grinding, and batching;
- zero-knowledge masking/randomization and leakage analysis;
- rejection sampling and challenge bias rules;
- concrete classical and quantum soundness after composition;
- proof size, verifier memory, and denial-of-service limits.

A 64-bit base field alone cannot deliver a 128-bit soundness claim. A generic STARK is not automatically zero knowledge. Empirical failure-free testing cannot prove a soundness probability. The final analysis MUST reach at least the R3 composed target; the former 2^-100 target is insufficient.

An aggregated block format MUST choose exactly one consensus representation:

1. individual transaction proofs; or
2. an aggregate proof plus authenticated references to transactions whose individual proofs are omitted.

Keeping all individual proofs and adding an aggregate does not reduce network or storage load.

Wallet and block proving are separate roles. A wallet MAY produce an individual transaction proof for admission and propagation. If an aggregate mode is selected, the block producer or dedicated aggregator MUST recursively or otherwise soundly combine admitted proofs, and the consensus block MUST carry the compact aggregate plus authenticated transaction references while omitting the individual proofs it replaces. The aggregate relation MUST bind the exact ordered transaction set, public inputs, pre-state, and post-state; aggregation MUST NOT turn proof generation into a trusted service.

T304/T305 may first measure this relation against an explicitly labelled test state context. T503–T505 supply the final canonical DAG and monetary context before integration; an earlier surrogate-state proof is not evidence of canonical consensus-state validity.

Transaction aggregation and accumulated historical validity are different profiles. T306 first evaluates the proof system's generic accumulation or recursion capability inside the R3 and R9 budgets; it does not freeze Quantum's consensus relation. After T405, T503, and T504 define bounded-policy semantics, canonical ordering, and monetary state, T505 owns the exact consensus-bound relation and bootstrap/state-validity profile. Neither task may count a compact proof as current block data, current state, a wallet witness, or recovery data, and a succinct verifier MUST NOT be reported as an executing validator.

4.6 Entropy, mnemonics, and key derivation

Randomness MUST come from an operating-system CSPRNG and MUST fail closed if entropy acquisition fails. Test seeds MUST never be accepted by production builds.

If mnemonic import/export is supported, it MUST implement BIP-39 exactly:

- mnemonic and passphrase are UTF-8 NFKD normalized;
- salt is UTF-8 NFKD “mnemonic” || passphrase;
- PBKDF2-HMAC-SHA512 uses 2,048 iterations;
- output is a 64-byte seed.

BIP-39 derives only the wallet master seed. Quantum child keys MUST then use a separately reviewed, domain-separated post-quantum derivation profile. Wallet storage encryption MUST have its own memory-hard password-KDF profile and MUST not describe the BIP-39 PBKDF as storage protection.

A conforming wallet MUST support a 24-word BIP-39 mnemonic generated from 256 bits of uniform entropy as the minimum compatibility profile. T001 and T204 MUST derive the full wallet master-secret entropy required after the approved multi-user, multi-target, and lifetime losses. The 24-word mnemonic alone MUST NOT be presented as evidence that the composed R3 target is met. If the bound requires more than 256 source-entropy bits, the wallet MUST combine the mnemonic with one or more independently generated, domain-separated secret components, or use a separately reviewed higher-entropy seed format. The profile MUST specify generation, backup, recovery, combination, and failure behaviour and MUST NOT count derived output length as additional entropy.

A lower-entropy mnemonic MAY be imported for legacy recovery only under a separately identified degraded-security profile. The wallet MUST show the downgrade before funds or keys are used and MUST NOT label that wallet as meeting the 128-bit post-quantum target. PBKDF2 output length and a 512-bit derived seed do not increase the entropy of the imported mnemonic.

5. Private note and transaction model

5.1 Note model

A note is private witness data:

```
Note {
    note_creation_profile
    asset_id
    value_u64
    spend_authorization_key_digest
    nullifier_key_digest
    recipient_data
    commitment_randomness
}
```

Public state contains a note commitment, not the plaintext. A recipient learns the note through an authenticated encrypted payload. Its associated data MUST bind the commitment, output index, chain and transaction context using the pre-encryption context in Section 6.1. The final transaction identifier then binds the resulting ciphertext; the ciphertext MUST NOT depend on that final identifier. The committed note-creation profile fixes historical interpretation and remains distinct from the active spending-transaction version.

5.2 Public transaction fields

The public transaction contains only:

- protocol version and chain identifier;
- one approved anchor root and its consensus context;
- a bounded vector of nullifiers;
- a bounded vector of new note commitments;
- corresponding encrypted-note payloads or authenticated payload digests;
- public fee in base units and an explicitly tagged fee context, including the fee epoch and weight-profile identifier if the dynamic candidate is selected;
- expiry/finality context and transaction flags;

- one active authorisation profile identifier;
- the authorised proof policy; and
- a separate proof envelope containing the actual representation and proof bytes or authenticated aggregate reference.

Every vector length and byte string MUST have a versioned maximum before parser implementation. Parsers MUST reject overlong, truncated, duplicate, non-canonical, unknown-version, unknown-authorisation-profile, and trailing-byte encodings before expensive cryptographic work.

5.3 Private witness

For every input, the witness contains the full note plaintext, commitment randomness, membership path, nullifier secret, authorization public key, and profile-specific authorization witness. Stateful signing state remains in the wallet and is not part of the transaction proof witness. For every output, the witness contains the full new note plaintext, commitment randomness, and encryption witness required to bind the public encrypted payload.

Input commitments and their openings are mandatory witness relations. They MUST NOT be omitted merely because only the anchor root is public.

5.4 Transaction validity relation

A verifier accepts only if the proof establishes all of the following:

1. **Canonical context**: all public and private encodings use the selected version, chain, asset, and domain tags.
2. **Input opening**: for each input i, Commit(pp, input_note[i], input_rho[i]) equals the commitment leaf authenticated by its membership path.
3. **Membership**: every input leaf is a member of the public anchor root under the exact tree hash and depth.
4. **Nullifier correctness**: each public nullifier is derived from the opened note, its bound nullifier key, authenticated leaf position, chain and immutable note-creation profile under Section 5.6.1.
5. **Authorization**: the opened note binds the authorization key, the declared active transaction profile verifies its complete witness over the exact transaction authorization digest. Every input uses the same authorised signature profile, with historical-note compatibility explicitly resolved by T204/T510 before a profile change can activate.
6. **Output opening**: for each output j, Commit(pp, output_note[j], output_rho[j]) equals the corresponding public output commitment.
7. **Encryption binding**: every public encrypted note or digest authenticates the same output plaintext and output position.
8. **Range**: every value and fee is a canonical unsigned 64-bit base-unit integer and obeys per-note and asset supply bounds.
9. **Integer conservation**: sum(input_values) = sum(output_values) + fee as an integer, except in the separately typed reward transaction.
10. **No local duplicates**: input nullifiers and output commitments are unique within the transaction.
11. **Public consistency**: counts, indices, flags, expiry, fee context, authorisation profile, and proof policy agree across the staged digests, encrypted payloads, and proof. The envelope representation is permitted by that policy and binds the same transaction effects.

The verifier then performs state checks outside the proof: the anchor is currently admissible, each nullifier is globally unused, the transaction is not expired, resource limits hold, and the proof version is active.

5.5 Carry-safe balance constraints

Field equality is insufficient because field arithmetic can wrap. Each u64 value MUST be decomposed into four 16-bit limbs, with a range constraint for every limb. The circuit MUST:

1. add all input values into an accumulator wide enough for the maximum input count, constraining every base-2^16 carry;
2. add all output values and the fee into an equally wide accumulator, constraining every carry;
3. constrain every accumulator limb and final carry equal;
4. reject overflow beyond the protocol-wide maximum sum.

The maximum vector length determines the accumulator width. This relation is over integers represented in the proof field; no unchecked field reduction is allowed. The 16-bit limb representation is an arithmetic encoding, not a claim that a commitment is homomorphic across carries.

5.6 Commitment tree

The candidate note tree depth is 64, subject to benchmark and state-layout review. MERKLE_HASH_BYTES is deliberately unresolved: T001, T201, and T302 MUST derive it from the concrete quantum collision, second-preimage, multi-target, and protocol-lifetime analysis. Empty nodes are then defined recursively:

```
empty[0] = QH("empty-leaf", empty, MERKLE_HASH_BYTES)
empty[d + 1] = QH(
    "merkle-node",
    empty[d] || empty[d],
    MERKLE_HASH_BYTES
)
```

Leaves and internal nodes MUST use different tags. Insertion order MUST come from the canonical DAG state order. An anchor-acceptance window MUST be derived from measured proof-generation latency, propagation, finality, and reorg risk; a fixed "last 100 headers" rule is not acceptable without that derivation.

5.6.1 Note-instance identity and upgrade continuity

The candidate instance identifier consists of chain identity, the committed note-creation profile, canonical leaf position, and note commitment. T302 MUST instantiate and review the exact nullifier function over that instance and the bound nullifier secret. This is an interface requirement, not a new hash construction or a security proof. The position used by the function MUST be the same constrained position verified by the membership path.

Within one canonical history, distinct accepted leaf instances MUST have distinct nullifiers except with the negligible probability justified by T001, even for sender-chosen duplicate notes or randomness. The same instance MUST have one nullifier across admissible anchors and active transaction versions. This addresses the note-spendability problem discussed in the Zcash protocol, Section 8.4 and the position binding in the TzEL whitepaper; their constructions and security claims are not adopted automatically.

Reorganisation rollback MUST remove orphaned insertions and dependent spends atomically before replay assigns new positions. A wallet MUST identify notes by authenticated instance, not commitment alone, and refresh witnesses after reorgs. An orphaned position or proof cannot be transplanted into the new history. An upgrade cannot reinterpret old creation profiles or reset the spent-nullifier state. Any migration MUST consume the old instance exactly once, preserve value, and specify historical authorisation, recovery, expiry, and offline-wallet handling before activation. T302/T403/T510 require vectors for duplicate commitments across transactions, changed positions, version changes, migration replay, and unspent historical-note recovery.

5.7 Selective disclosure capabilities

Selective disclosure is a wallet capability, not a consensus bypass. The initial profile MUST specify non-overlapping key or capability types for:

- **incoming view**: discover and decrypt received notes without spend authority;
- **full-wallet view**: reconstruct the explicitly authorised incoming and outgoing wallet history without spend authority;
- **transaction disclosure**: reveal one transaction's selected plaintext and cryptographic binding without exposing unrelated wallet history; and
- **auditor-scoped disclosure**: derive a capability restricted by a canonical scope such as account branch, counterparty set, transaction class, or time interval.

Each exported capability MUST state exactly what it reveals, what it cannot prove, its scope encoding, and its forward/backward visibility. Wallets MUST warn that data already disclosed cannot be cryptographically retracted. Revocation MAY stop future access only where the underlying construction supports it; it MUST NOT be described as erasing copied history. No consensus participant, operator, foundation, or regulator receives a universal recovery or viewing key by design.

Disclosure receipts or proofs MAY support an external audit workflow, but they do not themselves establish legal compliance, identity, source of funds, or an absence of undisclosed transactions.

5.8 Bounded payment policies and interoperability boundary

The candidate application surface is a versioned, finite set of private payment policies rather than arbitrary bytecode. Candidate policies include absolute/relative timelocks, hashlocks, threshold or multisignature authorisation, escrow, recurring-payment authorisations, atomic-swap conditions, and conditional release. Every enabled policy MUST have:

- one canonical encoding and explicit state-transition semantics;
- bounded execution, witness, proof, verification, and storage costs;
- a validity relation that preserves note privacy and exact supply;
- domain-separated authorisation and replay protection;
- positive, negative, timeout, reorg, and recovery vectors; and
- independent review under the same release gates as ordinary transfers.

Unknown policy identifiers and unsupported versions MUST fail closed. There is no fallback interpreter and no implicit general-purpose virtual machine. Private asset issuance is deferred until an asset-specific conservation, authorisation, disclosure, and lifecycle profile passes a separate release decision; asset_id in the note schema reserves domain separation but does not claim that such issuance exists.

Cross-chain atomic swaps or adapters MAY be researched after the core transaction feasibility gate. External consensus, custody, multisignature/MPC, oracle, light-client, and bridge assumptions remain outside Quantum's base security claim. An adapter MUST NOT be described as post-quantum secure, trustless, private, or production-ready merely because the Quantum side meets its own gates.

6. Serialization and identifiers

Consensus serialization MUST be specified field by field before implementation:

- fixed little-endian integers of explicit width;
- length prefixes of explicit width followed by bounded data;
- no floating-point values;
- no implicit maps, platform-dependent structs, Unicode identifiers, or alternate encodings;
- shortest/canonical form only;

- dedicated tags for transaction ID, authorization digest, note commitment, nullifier, Merkle leaf/node, block ID, PoW, KDF, and address checksum.

The candidate human-readable address is:

```
"qtm_" || lower_base32(version || network || payload || checksum)
```

lower_base32 uses the RFC 4648 alphabet abcdefghijklmnopqrstuvwxyz234567, lower case only, without padding. The checksum is the first eight bytes of QH("address-checksum", version || network || payload, 32). The final profile MUST first fix the payload and maximum length; until then, addresses are research-only and MUST NOT be used to receive value.

6.1 Acyclic transaction construction and proof carriage

T002 owns common encodings and domain rules. T401 owns the note/encryption interface, T402 owns the canonical transaction codec and digest schedule, and T303/T304 own proof encodings. Each must freeze its exact bytes before its consumer is implemented. The required data dependency is:

```
semantic fields -> encryption context -> encrypted outputs
semantic fields + encrypted-output digests -> effects ID (= txid)
effects ID -> dedicated authorisation digest -> signatures
effects + signatures + private witness -> individual proof
ordered effects + admitted proofs + state context -> aggregate envelope
```

The semantic fields contain version, chain, anchor/context, ordered nullifiers and commitments, counts, flags, fee/context, expiry, authorisation profile, and proof policy. They contain no ciphertext, signature, proof bytes, concrete aggregate reference, or block position. T402 MUST specify the exact domain-separated encryption-context digest over those fields. T401 uses that context, output index and commitment as associated data; T402 then computes the effects ID over the semantic fields and resulting ciphertext digests. The authorisation digest is a dedicated domain-separated commitment to that effects ID. No digest may depend on a value computed downstream of it.

The proof envelope is excluded from the effects ID and authorisation digest. It MUST authenticate every referenced effect, its canonical order and the applicable state context. The block commits to both effects and proof carriage. Replacing an individual proof with an aggregate permitted by the signed policy MUST NOT change the txid, fee, outputs, or require resigning. Changing an effect or proof policy invalidates authorisation. Swapping, omitting, or reordering effects under an envelope invalidates that envelope. The separation of effects from authorising data in Zcash ZIP 244 is a structural reference; its hash parameters and classical cryptography are not the Quantum profile.

If fees depend on resource weight, that weight MUST be computable before signing from canonical effects and a frozen resource profile. Exact byte lengths or conservative, enforced profile bounds may be used; the fee MUST NOT depend on the eventual variable proof encoding or aggregate membership. The profile must still account for actual proof verification and carriage costs in T305/T506. An oversized proof or envelope fails admission, rather than retrospectively changing the signed fee. Static and dynamic fee contexts must have distinct explicit encodings; neither is a fallback for the other.

T402/T304 MUST supply vectors covering every stage, changed ciphertexts and fees, proof re-randomisation, aggregate replacement, and forbidden self-reference. This schedule freezes dependencies, not unresolved digest lengths, cryptographic constructions, or consensus parameters.

7. DAG consensus and state

7.1 Consensus profile — blocking gate

Quantum evaluates proof-of-work GHOSTDAG, whose research basis includes Sompolinsky, Wyborski and Zohar. The paper name is not a complete executable specification. Before implementation, a versioned consensus profile MUST pin:

- parent selection and well-formedness;
- anticone parameter and blue/red classification;
- selected-parent chain and merge-set ordering;
- accumulated work/blue-work calculation;
- finality and pruning rules;
- timestamp validity;
- block weight and data-availability limits;
- network-delay and adversarial-work assumptions.

"Highest blue score wins" and sorting blocks by score are not sufficient definitions.

7.2 Header and proof of work

The final header layout MUST have one canonical byte encoding and one block-hash function. At minimum it binds version, network/chain ID, parent set, transaction root, pre-state root, post-state root, timestamp, nonce, claimed target, consensus score/work commitments, and extension commitment.

Block validation MUST:

1. derive the canonical past from the DAG profile;
2. recompute the expected target using deterministic integer arithmetic;
3. require the header target to equal that expected target;
4. hash the canonical header under the PoW domain;
5. require the hash integer to be at most the expected target;
6. validate timestamp bounds from consensus history, using local time only for a non-consensus future-admission check.

An unverified miner-supplied target would make mining trivial and is forbidden.

7.3 Canonical state transition

T503 MUST define one execution context for each state commitment: its block or checkpoint identity, selected-parent history, ordered merge set, exactly which bodies are evaluated, and whether the committing block's own body is included or deferred. A root computed in one context MUST NOT be checked against a header from another context. Transaction-body commitments and accepted-state commitments have distinct meanings.

The selected-parent chain and ordered merge set MUST produce one deterministic transaction sequence in that context. Starting from its canonical pre-state, validators classify candidates and apply accepted transactions atomically:

1. validate public format and cheap resource limits before proof work;
2. reject malformed or cryptographically invalid bodies under the block- validity rule frozen by T503;
3. classify repeated txids, used nullifiers, expired transactions and inadmissible anchors under the ordered-acceptance rule;
4. skip contextually rejected transactions with no output, nullifier, fee, pool, or issuance effect; an honest parallel conflict alone does not invalidate an otherwise valid block;
5. append accepted outputs at canonical positions and update nullifier, commitment and exact T504 monetary state together;
6. commit the accepted transaction order and rejection classifications; and
7. require the computed root to equal the commitment for this same context.

For concurrent spends, the first transaction in canonical order may succeed; later uses of the same nullifier MUST fail. The design requires a proof that all honest nodes derive the same result across reorgs, pruning, and recovery.

7.3.1 Worked merge/reorg example for the T503 reference model

This symbolic example specifies expected conflict and accounting behaviour; it is not a selected GHOSTDAG profile or a cryptographic test vector. A deferred-acceptance candidate is used: sibling headers commit their shared past, and a later merge block commits the evaluation of the sibling bodies. T503 must either instantiate this candidate or supply an equally explicit context mapping before consensus implementation.

Let parent P have state S0 with unspent instance N worth 10 units. Sibling B carries X, spending N into a 9-unit output with fee 1. Sibling C carries Y, spending N into an 8-unit output with fee 2. Both bodies are individually well-formed and both refer to an admissible P anchor. The selected candidate orders B before C when merge block M evaluates them.

Context/event	Accepted effects	Note supply change	Fee-pool change	Issuance change
B and C headers: their shared past	Neither own body evaluated yet	0	0	0
M evaluates B then C	X accepted; Y rejected for used nullifier	-1	+1	0
Revert M's evaluated interval	Undo X, its output, nullifier and fee	+1	-1	0
Replay a selected history ordering C before B	Y accepted; X rejected	-2	+2	0

For this example alone, all accepted fees enter an unmatured miner pool; there is no burn, other allocation, subsidy, or reward claim. Thus outstanding supply is unchanged in each complete transition and the rejected spend pays no fee. B/C's body commitments remain intact; M's contextual state root is never substituted for either sibling's past-state root. M's own body is deferred in this example. A replay yielding the same selected order must reproduce the same symbolic state; a different selected order may choose the other spend after rollback, never retain both.

Signed table entries are explanatory changes, not encoded monetary values; all resulting balances and consensus arithmetic remain non-negative under R4.

T503/T504 must extend this model with the selected reward-creation context, blue/red eligibility, maturity, claims, controller epochs, burns, malformed bodies, stale anchors and descendant spends. T503 uses an injected accounting interface; T504 supplies and validates the selected monetary rules. No reward is inferred from a rejected transaction. Published byte/root vectors require the actual frozen profiles and independently checked implementations.

7.4 Difficulty adjustment

The previous wall-clock/float pseudocode is withdrawn. The final DAA MUST be specified in exact integer equations over canonical DAG history, with clamping, overflow behavior, timestamp manipulation analysis, test vectors, and cross-implementation tests. A node's current time MUST NOT alter the expected target for an already received DAG.

7.5 Rewards, supply, and genesis

The current monetary target is a lifetime gross-issuance cap of 21,000,000 QTM with no premine, ICO allocation, or founder reward. The exact finite emission curve remains provisional. Section R12 requires a separately signed decision before this cap semantic can change.

A reward transaction MUST be a distinct consensus type. Its permitted subsidy MUST be calculated from a canonical DAA score or other explicitly defined DAG measure—not an ambiguous linear block height—and MUST specify eligibility for blue, red, merged, and stale blocks.

For each atomic accounting interval spanning canonical ordinary-transaction acceptance and its reward transition, validators MUST derive exact integers:

```
accepted_total_fees = accepted_resource_fees
                    + accepted_security_fees
                    + accepted_priority_fees
accepted_total_fees = miner_eligible_new_fees
                    + fees_burned_on_acceptance
                    + other_explicit_fee_outputs
available_miner_fee_pool = previous_miner_fee_pool
                         + miner_eligible_new_fees
0 <= matured_claimable_fee_pool <= available_miner_fee_pool
0 <= claimed_fees <= matured_claimable_fee_pool
0 <= claimed_subsidy <= authorized_subsidy
reward_outputs = claimed_fees + claimed_subsidy
foregone_subsidy = authorized_subsidy - claimed_subsidy
0 <= fees_burned_from_pool <= available_miner_fee_pool - claimed_fees
burned_existing_value = fees_burned_on_acceptance
                      + fees_burned_from_pool
next_miner_fee_pool = available_miner_fee_pool
                    - claimed_fees
                    - fees_burned_from_pool
next_cumulative_issuance = cumulative_issuance + claimed_subsidy
next_note_supply = previous_note_supply
                 - accepted_total_fees
                 + reward_outputs
                 + other_explicit_fee_outputs
previous_outstanding_supply = previous_note_supply
                            + previous_miner_fee_pool
next_outstanding_supply = next_note_supply + next_miner_fee_pool
next_outstanding_supply = previous_outstanding_supply
                        + claimed_subsidy
                        - burned_existing_value
```

All terms in these equations MUST be non-negative. The selected policy MUST determine the charge-class and destination split, claimed_fees, claimed_subsidy, other_explicit_fee_outputs, and both burn components exactly; failure to claim miner-eligible value does not authorise a burn. In this candidate, an other explicit allocation MUST settle as a named protocol-authorised non-miner output in the same atomic interval. A T506 profile MUST define its recipient/eligibility and privacy semantics; an unspecified treasury or producer-selected recipient is forbidden. A future delayed non-miner liability would require its own outstanding-supply state and a separately versioned accounting rule. The authorized subsidy MUST be non-negative and no greater than 21,000,000 QTM - cumulative_issuance. The fee portion is transferred value and MUST NOT increment cumulative issuance. A fee awaiting later payout is held in the miner fee_pool: it is existing value and part of outstanding supply, but is not a spendable note and is not gross issuance. The direct-payout case is the special case in which the applicable pool balance returns to zero. burned_existing_value is already-issued value destroyed only by an explicit selected rule. foregone_subsidy was never issued and is not a burn; under the current cap it does not

create an additional future schedule entitlement. Burned existing value does not restore issuance headroom at this revision. Reward outputs and fee claims MUST use the same private note system without exposing recipient data beyond the final consensus disclosure budget. The profile MUST define fee-pool maturity buckets or their equivalent, maximum retention, reorg reversal, reward eligibility, rounding and remainder ownership, and atomic application so that a fee, subsidy, foregone subsidy, or burn cannot be lost or counted twice. Supply accounting MUST prove these equations and the selected monetary-policy invariant under rounding and reorganisation.

T506 must treat cap semantics, the integer issuance schedule, the fee/burn rule, and the economic security model as separate decision axes. Greater use creates more fee transfer; it creates more burn only if the selected rule explicitly destroys part of that existing value. A halving or other schedule step changes the subsidy component, not necessarily total miner revenue, which also includes fees and responds through entry, exit, hashrate, and difficulty.

7.5.1 Bounded dynamic-fee and reward-window candidate

The retained lifetime-cap baseline now includes a dynamic-fee candidate for T506/T508 comparison. This records an interface and failure boundary, not a selected fee schedule or evidence that the revenue objective can be met.

Let e identify a fee/reward epoch derived from non-overlapping ranges of canonical DAA score. The candidate interface is:

```
fee_rate_sum[e] = add_checked(resource_rate[e], security_rate[e])
fee_numerator(tx, e) = mul_checked(
    transaction_weight(tx), fee_rate_sum[e]
)
required_fee(tx, e) = ceil_div_nonnegative(
    fee_numerator(tx, e), fee_rate_scale
)
total_fee(tx, e) = add_checked(required_fee(tx, e), priority_fee(tx, e))
0 <= priority_fee(tx, e) <= maximum_priority_fee[e]

accepted_total_fees[e] = sum(total_fee(tx, e) for each transaction
                             canonically accepted in e)
accepted_total_fees[e] = accepted_resource_fees[e]
                         + accepted_security_fees[e]
                         + accepted_priority_fees[e]
accepted_total_fees[e] = miner_eligible_new_fees[e]
                         + fees_burned_on_acceptance[e]
                         + other_explicit_fee_outputs[e]
sustainable_miner_budget[e] = authorized_subsidy[e]
                              + miner_eligible_new_fees[e]
payout_capacity[e] = authorized_subsidy[e]
                     + matured_claimable_miner_fee_pool[e]
claimed_miner_revenue[e] = reward_outputs[e]
revenue_gap[e] = 0
    if sustainable_miner_budget[e] >= target_miner_budget[e]
    else sub_checked(target_miner_budget[e], sustainable_miner_budget[e])

resource_rate[e + 1] = bounded_resource_controller(
    resource_rate[e], finalized_utilization[e]
)
security_rate[e + 1] = bounded_security_controller(
    security_rate[e], revenue_gap[e], accepted_weight[e]
)
```

For non-negative a and positive b, the consensus operation ceil_div_nonnegative(a, b) MUST compute q = a div b, r = a mod b, and q + 1 iff r != 0, otherwise q. It MUST NOT use an unchecked (a + b - 1) expression. T506 MUST freeze the exact unsigned widths of monetary

fields, transaction weight, rates, fee numerator, per-interval accumulators, target/gap values, and controller state plus every operand maximum. fee_rate_scale MUST be positive. add_checked, mul_checked, summation, and the final division MUST reject any out-of-range input or intermediate rather than wrap or saturate.

miner_eligible_new_fees means newly accepted value scheduled by the frozen rule to become claimable by miners. It excludes every amount already designated for burn or a non-miner output. A later explicit expiry burn is permitted only after the amount had the defined opportunity to mature and be claimed; T506 reports that burn separately from sustainable budget and actual claimed revenue.

transaction_weight MUST be one canonical integer function of serialised bytes, proof-verification work, and state/witness impact. The exact components and coefficients remain a T506 decision followed by T508 validation. Every controller input MUST come from the same finalised canonical state and be delayed far enough that a transaction knows its fee epoch and required rate before authorisation. The transaction authorization digest and validity relation MUST bind the fee epoch, fee, weight/profile identifier, and any permitted priority class. Validators MUST reject an unknown epoch/profile or an underpayment before expensive proof verification.

The selected controller MUST define an exact integer transition for zero accepted weight and every boundary value. Rate clamps limit price movement; they do not create missing revenue. The target is a pre-registered QTM-denominated comparison input, not an oracle claim about energy expenditure or attack cost. The controller uses sustainable miner budget, excluding burns, non-miner outputs, and withdrawals from the prior fee-pool balance, so a producer cannot raise a later fee rate merely by foregoing a valid reward, and neither an old pool balance nor destroyed value can masquerade as new miner revenue. T506 separately reports total accepted fees, each charge and destination component, claimed miner revenue, payout capacity, and fee-pool balance. If either applicable rolling metric remains below its registered threshold, the result is UNDERFUNDED for G13 and MUST NOT authorize extra issuance or a cap bypass.

Fees are accounted for only in their one canonical acceptance interval on the selected history. A reorganisation reverses the former interval assignment, pool change, burn, and output before any exactly-once reassignment. T506 must compare whether each finalised window assigns them directly to the first-including eligible work, shares or pools a fixed portion, burns an explicit fixed portion, or treats resource and security components differently. Any retained miner- eligible portion MUST stay in the canonical fee-pool liability and use deterministic work weights, maturity, maximum retention, rounding, and remainder rules. T506 freezes the exact candidate for one campaign before T504 implements it. T508 then attempts to falsify that frozen implementation against inclusion, free- riding, censorship, self-fee, duplicate, reorganisation, controller, and pool-formation attacks. If T508 rejects it, the dependent T504/T507/T508 evidence is invalid for selection and a new versioned T506 campaign is required; T508 MUST NOT silently mutate the rule or create a task-graph back edge.

There is no canonical genesis block at this revision. Before a network launch, one immutable genesis byte string, timestamp unit, message field, target, and hash algorithm MUST be published with vectors. Seconds and milliseconds MUST not be mixed, and genesis MUST use the same block hashing rules as later blocks. The selected genesis profile MUST also commit to exact values satisfying:

```
initial_miner_fee_pool = 0
initial_cumulative_issuance = genesis_claimed_subsidy
initial_note_supply = genesis_claimed_subsidy
initial_controller_epoch = GENESIS_FEE_EPOCH
resource_rate[GENESIS_FEE_EPOCH] = GENESIS_RESOURCE_RATE
security_rate[GENESIS_FEE_EPOCH] = GENESIS_SECURITY_RATE
```

The epoch and both rates MUST be canonical constants inside the frozen T506 profile and fit its exact integer widths. Under the current no-premine, no-ICO, and no-founder-reward baseline,

genesis_claimed_subsidy MUST be zero; changing that value requires the same separately approved monetary and public- claim replacement as any other premine.

8. Network anonymity and transport

The P2P design has two separate obligations:

1. **link security**: post-quantum authenticated, replay-protected, downgrade-resistant channels; and
2. **origin privacy**: resistance to mapping a transaction to its source.

The link profile may combine ML-KEM-1024, SLH-DSA, a transcript KDF, and a 256-bit authenticated cipher only after a reviewed composition fixes every message and failure path. Authentication MUST not create a stable user identity or link wallet activity.

The origin-privacy profile MUST evaluate stem/fluff diffusion, mix or onion routing, cover traffic, delayed batching, peer rotation, route failures, eclipse resistance, and active tagging. Dandelion++ is relevant research (paper) but promises probabilistic de-correlation under a model, not universal prevention of correlation.

Release evidence MUST include network simulation, adversarial experiments, privacy metrics with confidence intervals, and independent review. Simple byte uniformity or Pearson-correlation tests are diagnostics, not anonymity proofs.

9. Scalability, operability, and recovery budget

9.1 End-to-end target

The 1,000 transactions-per-second figure is an acceptance target, not a current capability claim. A benchmark passes only when accepted state transitions—not submitted requests—sustain the target while every selected R1–R12 mechanism and required operating role remains enabled and separately measured.

The reference workload MUST include a published mix of input/output counts, note scans, conflicts, reorgs, proof modes, peer delays, and malformed traffic. Results MUST report at least p50/p95/p99 latency and resource use.

9.2 Feasibility budget

At 1 Gbit/s, a link has a theoretical 125 MB/s before protocol overhead. With 20% headroom and four outgoing gossip copies, only about 25 MB/s remains for unique transaction data: roughly 25 KB per transaction at 1,000 tx/s. A 110 KB transaction therefore cannot meet that example topology, even before headers and retransmission.

Likewise, 25 KB × 1,000 tx/s is roughly 788 TB/year of unique transaction-data flow at continuous target utilisation. It is not automatically live-state, on-disk ledger, or required archive growth. Reports MUST distinguish network bytes including gossip copies, unique canonical bytes, current authenticated state, prunable history, and durable recovery data. A bounded operational-node target is possible only with a measured combination of compact representations, pruning/state commitments, and a separate recovery design. These calculations are mandatory design inputs, not optional optimizations.

The final profile MUST publish budgets for:

- average and worst-case wire transaction size;
- proof size and aggregate amortization;
- verifier operations and memory per transaction;
- inbound/outbound propagation amplification;
- nullifier, commitment, block, and archive growth;
- wallet scan bandwidth and time;
- snapshot size, creation time, verification time, and recovery trust;

- executing-validator, succinct-verifier, producer, prover, state-provider, pool/share-aggregator, and archive-provider resources separately;
- bootstrap and restart bytes/time inside the frozen R9 profile; and
- current-data withholding, archive reconstruction, repair, and eclipse behavior.

If an encrypted note payload is represented on layer 1 only by a digest, the payload has not disappeared from the payment system. T305 and every later capacity report MUST separately state consensus-transaction bytes, external encrypted-payload bytes, total bytes generated per payment, provider/retrieval traffic, and availability and retention assumptions. Digest-only carriage MUST NOT reduce the reported client or wire feasibility cost by moving required bytes to an unreported provider.

9.3 Pre-node cryptographic feasibility gate

Before full-node or GHOSTDAG integration begins, two independent implementations MUST execute a representative private transaction with exactly two inputs and two outputs for every retained authorisation arm. Apart from authorisation and its required state, the statement, witness relation, proof profile, hardware, and measurement method MUST remain identical. The measured relation MUST include:

- both input commitment openings and Merkle membership paths;
- nullifier derivation and uniqueness constraints;
- both output openings and encrypted-note binding;
- complete authorisation verification inside the proof for the declared profile, including exact SLH-DSA-SHAKE-256f and the required SLH-DSA-SHAKE-256s comparator;
- 64-bit ranges, carry-safe integer conservation, and the public fee; and
- the selected STARK transcript, zero-knowledge masking, verifier, and, if proposed, aggregation path.

The non-normative T305 research protocol pre-registers the literature boundary, exact experiment, measurement fields, independent-implementation rules, and stop/go evidence for this gate. It may make the experiment more specific but cannot omit or weaken a requirement in this specification. The versioned T305 prior-art and reuse decision records why the experiment is comparative, which public work must be replicated or labelled author-reported, and which code may not be adopted. Named T001 owner and reviewer signatures remain mandatory before implementation.

T004 MUST pin the benchmark method, artifact harness, reference desktop and constrained-client environment, workload, and parallelism controls. T005 MUST then freeze the numerical proof-latency, memory, verifier, proof-size, wire-size, external-payload, total-payment-byte, provider-traffic, and aggregate- amortisation thresholds before results are interpreted. The report MUST publish single-wallet latency separately from aggregate throughput and MUST keep consensus bytes, external payloads, total payment bytes, and provider traffic distinct. Multiplying ideal parallel jobs is not a wallet-latency result.

Each retained stateless arm must independently meet the frozen material-benefit rule against qualifying stateful arms. T305 may retain the 256f incumbent or recommend 256s; selecting 256s or a stateful arm requires a versioned authorisation-profile decision before integration. A failing 256f arm cannot exclude a qualifying 256s arm. If no arm meets the frozen feasibility budget, that is a design failure, not a node optimisation backlog. The signature profile, commitment, validity relation, proof system, or explicit system requirements MUST be revised, or the project MUST stop before consensus integration. No fabricated “hundreds” or “thousands” of proofs per second threshold substitutes for the published hardware and end-to-end budget.

9.3.1 Early receiving and transport feasibility screen

T006 MUST screen receiving and anonymous-transport costs before the full T305 campaign. T004 supplies the method and T005 freezes client/provider budgets, offline durations, workload and

packet-size ranges, delivery thresholds and privacy metrics before screening results are viewed. Actual T401 encrypted outputs are required for scan/decryption measurements; not-yet-built proofs and transport components may be modelled only as explicitly labelled bounds or simulation inputs, never as implemented security or achieved TPS.

The screen MUST compare a full local scan with every proposed private retrieval path, including malicious outputs, missed notes, false positives, offline catch-up, provider work and query leakage. It MUST also evaluate the traffic and latency cost of candidate batching, mixing and cover traffic against the named observer, including idle clients and correlated retries. At 1,000 two-output transactions/s, one 1,568-byte ML-KEM-1024 ciphertext per output alone gives 270,950,400,000 bytes/day before payloads and overhead. This is a lower-bound workload calculation, not an implemented wallet cost.

T006 returns NOT_RULED_OUT or STOP for the frozen envelope. NOT_RULED_OUT permits T305, but does not pass recipient privacy, R2, G6, or a full-system capacity gate. Missing required evidence cannot produce NOT_RULED_OUT. T305 must check that its actual format and sizes remain inside that envelope; an out-of-envelope result requires a new reviewed T006 campaign before integration. T404/T602/T604 still own the complete implementation and final evidence. The later discovery and anonymity paper templates remain templates.

9.4 End-to-end performance gates

Before a public scalability claim:

- the exact benchmark revision and toolchain are pinned;
- the same consensus parser and verifier used by nodes are measured;
- at least two independent implementations interoperate;
- a 24-hour target run and a longer stability run complete without state-root divergence;
- overload behavior remains bounded and malformed inputs are rejected before disproportionate work;
- raw artifacts and reproduction commands are published.

9.5 Operability gate

T004 owns the benchmark method and artifact harness. T005 then freezes the named R9 executing-validator profile and every material threshold before T006, T305, T306, T508, T509, or T602–T605 results are interpreted. This ordering contains no reverse dependency: a failed campaign may inform a later version, but cannot pass by changing the profile under test.

Pre-registration must state how "independently obtainable" is measured across regions and how hardware, connectivity, power, and bootstrap cost are compared. This revision intentionally supplies no unsupported CPU, memory, storage, bandwidth, power, or price ceilings. G7 passes only when G10 passes for the same revision, workload, campaign, and profile.

9.6 Validity, state, and data roles

The final design must publish a role/data matrix. An executing validator must receive and validate complete current data under R10. A succinct verifier may check a compact checkpoint proof but is not an executing validator. A producer, prover, or state/witness provider may require more resources, but those costs and concentration risks remain part of G10–G12 and cannot be omitted from a system-level throughput claim.

Witness-carrying authenticated state, accumulated-validity proofs, erasure- coded recovery, and data-availability sampling are candidate mechanisms. Each must be selected only after its task passes. None substitutes for the others: an authenticated witness is not historical data, a validity proof is not current data, sampling is not full execution, and archive reconstruction is not network-origin privacy.

10. Assurance and release gates

10.1 Required analytical artifacts

1. complete protocol and threat model;
2. commitment construction with concrete post-quantum security analysis;
3. STARK soundness and zero-knowledge analysis, including QROM composition;
4. transaction validity proof covering both input and output openings;
5. GHOSTDAG/state-ordering safety and liveness analysis;
6. DAA and issuance correctness proof;
7. network anonymity analysis;
8. end-to-end multi-target security budget;
9. role-specific operability and current-data/recovery analysis;
10. transaction-selection, censorship, pool, and hardware-contestability analysis; and
11. monetary-security comparison with exact cap, schedule, fee, burn, and miner-revenue assumptions.

10.2 Required implementation evidence

1. pinned toolchains and reproducible builds;
2. official KATs for FIPS 202, FIPS 203, and FIPS 205;
3. canonical cross-language vectors for every Quantum encoding;
4. differential tests between two independent implementations;
5. fuzzing, property tests, malformed-input and resource-exhaustion tests;
6. constant-time and side-channel review for secret-dependent operations;
7. state/reorg/recovery model checking or formal verification where feasible;
8. reproducible performance and anonymity experiments;
9. current-data withholding, bootstrap, snapshot, archive reconstruction, repair, and eclipse experiments; and
10. miner-controlled template, pooled-mining, ordering, censorship, and hardware-comparison artifacts.

10.3 Independent review

Testnet promotion requires named human owners and independent specialist review for cryptography, proof systems, consensus, networking, implementation security, data availability/storage, mining and pool incentives, performance, and monetary economics. A change to token, cap, issuance, burn, or public monetary wording additionally requires a named Product Owner and Legal Counsel. Production requires closure or explicit rejection of every high-severity finding and a public statement of residual risk.

An AI system may assist with implementation or analysis, but MUST NOT approve its own cryptographic design, proof, audit, benchmark, or release.

10.4 Go/no-go matrix

Gate	Pass condition	Current state
G1 Requirements	R1–R12 traceable to tasks and tests	Drafted, not independently reviewed
G2 Commitment	Exact scheme and 128-bit composed PQ analysis	Blocked
G3 Proof	Complete AIR/transcript/ZK/soundness profile	Blocked

Gate	Pass condition	Current state
G3A Transaction feasibility	Complete 2-input/2-output proof meets frozen client, verifier, size, and aggregation budgets	Not run
G3B Early receiving/transport screen	T006 returns NOT_RULED_OUT and actual T305 output remains inside its frozen envelope; no anonymity claim	Not run
G4 Private transaction	Prevention of unauthorised issuance and authorisation relation reviewed	Draft relation only
G5 DAG state	Deterministic consensus and conflict handling proved/tested	Blocked
G6 Network anonymity	Threat model, design, experiments, review pass	Blocked
G7 Scalability	1,000 accepted tx/s end to end with artifacts	Not run
G8 Interoperability	Two independent implementations and vectors	Not started
G9 External audit	Critical/high findings resolved	Not started
G10 Operability	G7 passes inside the frozen executing-validator profile with bootstrap/restart evidence	Not started
G11 Current data/recovery	Current-body validation, snapshot/bootstrap, withholding, and no-trusted-sole-provider recovery pass	Not started
G12 Producer contestability	Pre-registered template, publication, inclusion, hardware, and concentration thresholds pass with the pooled-mining comparator	Not started
G13 Monetary security	Pre-registered monetary scenarios and thresholds, selected policy invariant, accounting proof, stress model, and required reviews pass	Not started

No "specification complete," "quantum-secure," "fully anonymous," or "1,000 TPS achieved" claim is permitted while the corresponding gate is open. G7 does not pass G10, proof soundness does not pass G11, deterministic consensus does not pass G12, and a cap or positive subsidy does not pass G13.

11. Legal and licensing boundary

Repository-authored research text is dedicated under CC0-1.0 as described in the repository LICENSE file. CC0 does not grant patent or trademark rights and does not make a design "patent-free." Third-party standards, papers, names, and implementations retain their own rights and licenses.

This document is technical research, not legal, financial, tax, or investment advice. It is not an offer to sell a token, security, or network service. Privacy features do not remove obligations under applicable law. Obtain specialist legal advice before any launch or token distribution.

12. References

Primary references:

1. NIST, FIPS 202: SHA-3 Standard.
2. NIST, FIPS 203: Module-Lattice-Based Key-Encapsulation Mechanism Standard.
3. NIST, FIPS 205: Stateless Hash-Based Digital Signature Standard.
4. NIST, SP 800-208: Recommendation for Stateful Hash-Based Signature Schemes.
5. NIST, SP 800-230 initial public draft: Additional SLH-DSA Parameter Sets for Limited-Signature Use Cases, 2026; non-final at this revision.
6. NIST, Post-Quantum Cryptography project and standardization status.
7. TzEL contributors, TzEL whitepaper,

   2026.
8. Alupotha, Boyen and McKague, LACT+: Efficient Lattice-Based Aggregatable Confidential Transactions, 2023.
9. Ben-Sasson et al., Scalable, transparent, and post-quantum secure computational integrity.
10. Ben-Sasson et al., DEEP-FRI.
11. Baum et al., More efficient commitments from structured lattice assumptions.
12. Sompolinsky, Wyborski and Zohar, PHANTOM/GHOSTDAG.
13. Fanti et al., Dandelion++.
14. Noise Protocol Framework.
15. Bitcoin BIPs, BIP-39.
16. Hosoyamada and Sasaki, Finding Hash Collisions with Quantum Computers.
17. Béguinet et al., GeT a CAKE: Generic Transformation from KEM to PAKE.
18. Perešíni et al., DAG-Oriented Protocols PHANTOM and GHOSTDAG under Incentive Attack via Transaction Selection Strategy, arXiv preprint, 2021.
19. Mike Zak and Ro Ma, KIP-15: Canonical Transaction Ordering and SelectedParent Accepted Transactions Commitment.
20. Michael Sutton, Maxim Biryukov and Hans Moog, KIP-21: Partitioned Sequencing Commitment with O(activity) Proving, with its reserved proving companion, snapshot 2026-08-09.
21. Stratum V2, Job Declaration Protocol.
22. Dryja, Utreexo: A dynamic hash-based accumulator optimized for the Bitcoin UTXO set.
23. Al-Bassam, Sonnino and Buterin, Fraud and Data Availability Proofs.
24. Carlsten et al., On the Instability of Bitcoin Without the Block Reward, CCS 2016.
25. Zcash, Protocol specification, Section 8.4: Faerie Gold attack and fix, consulted 2026-09-05.
26. Nuttycombe, Hopwood and Grigg, ZIP 244: Transaction Identifier Non-Malleability, consulted 2026-09-05.

Appendix A — Decisions deliberately not frozen

The following are intentionally unresolved because choosing numbers without analysis would create false precision:

- the commitment construction and parameters;
- the proof field extension, AIR, FRI, masking, and aggregation profile;
- transaction and vector byte maxima;
- note-encryption and P2P authenticated-channel composition;
- address payload and derivation hierarchy;
- wallet master-secret entropy and any supplemental secret-composition profile;
- post-quantum upgrade authorisation, activation, recovery, parameter registry, and emergency-changeable allowlist;
- GHOSTDAG anticone/finality parameters and DAA;
- PoW function and measured hardware-contestability thresholds;

- monetary-policy decision, reward curve, dynamic-fee controller parameters and exact integer widths, charge-to-destination allocation, fee/reward-window allocation, fee/burn rule, security-budget model, and genesis constants;
- state representation, witness-carrying-state decision, accumulated-validity decision and fallback;
- miner-template, pooled-mining, ordering and incentive profiles;
- current-data, snapshot, archive encoding, reconstruction, repair and retention profiles;
- anonymity network parameters;
- the R9 executing-validator hardware, connectivity, power, cost, bootstrap, and latency ceilings beyond the 1,000 tx/s acceptance target.

Each item has an owner task in the verification guide. A value becomes normative only with rationale, vectors, tests, and review.

Appendix B — Revision record

0.5.4-research — 2026-09-05

- required authenticated note-instance uniqueness, upgrade-stable nullifiers, and explicit historical-note spending and migration evidence;
- separated pre-encryption context, transaction effects, authorisation and proof carriage into an acyclic schedule with pre-signing fee weight;
- required context-specific DAG roots and explicit block-invalid versus transaction-rejected outcomes; added a symbolic merge/reorg accounting case;
- added FIPS 205 256s as a required comparator, corrected the SP 800-230 reference and bounded any exploratory limited-use arm;
- added the T006/G3B early receiving/transport screen and aligned task-owned interface freezes; and
- retained all research/no-result labels and the current monetary policy.

0.5.3-research — 2026-08-25

- retained the 21,000,000 QTM lifetime gross-issuance cap while adding a bounded dynamic-fee and finalised reward-window design as a required T506/T508 comparator;
- separated resource pricing from a token-denominated miner-revenue controller, required prior-finalised inputs, integer clamps, an explicit zero-use branch, and an observable UNDERFUNDED outcome without extra issuance;
- added fee-pool liability and outstanding-supply accounting for delayed payout and explicit burn;
- separated accepted fee classes from miner-eligible, burn, and non-miner destinations; required exact integer/overflow semantics and canonical genesis controller state;
- prevented emergency upgrades from silently changing protected monetary rules; and
- made T506 freeze the payout/pool/burn rule before implementation while T508 adversarially tests it and triggers a new campaign after rejection.

0.5.2-research — 2026-08-09

- made the wallet master-secret entropy requirement depend on the T001/T204 multi-user, multi-target, and lifetime bound rather than treating a 256-bit BIP-39 mnemonic as automatic evidence for R3;
- bound the exact T505 policy-state relation to T405 in the verification graph;
- added governance-capture and voluntary-social-adoption boundaries to R7.8;
- required digest-only encrypted note payloads and their provider traffic to remain visible in T305 and later wire/capacity reports; and

- linked the independently versioned 0.3.1 decision record explicitly to this design revision without selecting a replacement monetary policy.

0.5.1-research — 2026-08-09

- required 256 bits of uniform mnemonic entropy for wallets claiming the full R3 target and isolated lower-entropy legacy imports behind a visible degraded profile;
- assigned post-quantum upgrade governance and activation to T510;
- limited T306 to generic proof-system accumulation capability and assigned the exact Quantum consensus/state relation to T505 after T503 and T504;
- required pre-registered falsifiable metrics, scenarios, thresholds, and STOP conditions for G12 and G13 without inventing their numerical values;
- replaced cap-only and no-inflation terminology with selected monetary-policy invariants and prevention of unauthorised issuance; and
- documented the independent lifecycle of the QTM-RD-0.2 research-vector domain.

0.5.0-research — 2026-08-09

- added independent operability/recovery and producer-contestability/economic- review properties to the joint release boundary;
- introduced R9–R12 and paired gates G10–G13 without marking any mechanism or result verified;
- distinguished executing validators, succinct verifiers, producers, provers, state/witness providers, archive/recovery providers, and pools/share aggregators;
- kept recursion, witness-carrying state, ordering rules, recovery coding, and pooled-mining mechanisms as candidates behind explicit tasks;
- required miner-controlled templates and separate transaction-selection, censorship, pool-formation, and PoW hardware-contestability evidence;
- separated proof soundness, current-data availability, state access, wallet witnesses, archive recovery, and network-origin privacy;
- corrected the distinction among claimed fees, claimed and foregone subsidy, and burned existing value;
- retained the 21,000,000 QTM lifetime gross-issuance cap pending a separately signed product, monetary-economics, consensus, and legal decision; and
- linked the versioned decentralisation, operability, and security-budget decision and primary-source evidence boundaries.

0.4.0-research — 2026-07-23

- recorded TzEL as the closest public note/nullifier/ML-KEM/hash-authorisation/STARK engineering baseline identified by the scoped review and rejected a generic protocol novelty claim;
- changed FIPS 205 SLH-DSA-SHAKE-256f from an assumed final profile to the stateless incumbent in a comparison governed by a pre-registration requirement;
- required independently specified TzEL-shaped and applicable NIST SP 800-208 stateful comparators under the same complete transaction relation;
- made state-management failure analysis and a material-benefit rule mandatory before retaining the stateless profile;
- linked the versioned T305 prior-art and reuse decision, required named sign-off before implementation, and prohibited unlicensed source-code adoption.

0.3.0-research — 2026-07-21

- narrowed the initial product boundary to private post-quantum cash and settlement rather than a general-purpose smart-contract platform;

- added a mandatory complete 2-input/2-output proving feasibility gate before wallet, node, or DAG integration;
- separated wallet transaction proofs from block aggregation and required an aggregate to replace, not duplicate, individual proofs on the consensus wire;
- defined incoming, full-wallet, transaction-specific, and auditor-scoped disclosure capabilities without a universal viewing backdoor;
- introduced a finite, proof-bounded payment-policy direction and made unknown policy versions fail closed;
- made private assets and cross-chain adapters separate post-core profiles whose external trust assumptions do not inherit Quantum's base claims;
- recorded the product position as a private post-quantum settlement research path for people, organisations, and autonomous software agents.

0.2.0-research — 2026-07-09

- converted the document from a claimed complete formal specification to an evidence-gated research design;
- retained post-quantum security, default anonymity, and 1,000 layer-1 TPS as hard release requirements;
- closed the specification-level inflation gap by binding every input and output value to a commitment opening and requiring carry-safe integer conservation;
- removed the unsupported commitment formula, biased sampler, incorrect homomorphism claim, and incomplete STARK security arithmetic;
- replaced SPHINCS+ protocol naming with FIPS 205 SLH-DSA;
- removed the invented post-quantum Noise pattern and separated link security from origin anonymity;
- made target validation, deterministic DAA, DAG state ordering, rewards, and genesis explicit blocking deliverables;
- added realistic bandwidth/storage feasibility budgets and proof-aggregation semantics;
- corrected BIP-39 derivation and CC0 patent/trademark wording.
- made consensus digest lengths conditional on concrete quantum collision, second-preimage, multi-target, and lifetime analysis;
- added an explicit recipient-key-privacy requirement for encrypted notes;
- defined fee transfer, claimed subsidy, burns, reward maturity, and cumulative issuance as one state-transition invariant.