QANOVRA

YELLOW PAPER v2.4.2

Technical Specification of the Qanovra Layer-1 Protocol

Document control	Canonical value
Status	Technical Specification / Publication Layout Final Candidate
Baseline	Qanovra L1 v3.7.5 - Frozen L1
Date	23 August 2026
Native asset	QNV
Base unit	qnv-atom
Precision	5 decimals; 1 QNV = 100,000 qnv-atom
Maximum supply	10,000,000,000 QNV = 10^15 qnv-atom
Genesis hash	`78cefe355c7fb55c53dd173c9ca9d340306054a7fd5718e70d1d3193be7b93a3`
Chain ID	qnv-78cefe355c7fb55c53dd173c
Candidate SHA-256	Generated after final document freeze; see audit handoff manifest

This paper specifies Qanovra L1 v3.7.5 as the frozen technical baseline and separately records v2.4 publication-policy decisions where the protocol documentation closes economic, governance or disclosure gaps without claiming unproven runtime enforcement.

Abstract

Qanovra is a deterministic Layer-1 ledger whose frozen v3.7.5 baseline binds network identity to genesis, uses a single authoritative account-state ledger, constrains monetary amounts to a cross-client-safe integer domain, separates user accounts from protocol-controlled system accounts, and applies explicit authorization to privileged state transitions. This Yellow Paper describes the protocol as a state-transition system and records the invariants that implementations must preserve.

The document is intentionally narrower than the Qanovra Whitepaper. It does not describe market positioning. It defines identities, encodings, state, transactions, blocks, finalization, validator-set rules, economic accounting, system vaults, staking/slashing policy interfaces, governance controls, replay resistance and release invariants.

1. Normative Language and Scope

The terms MUST, MUST NOT, SHOULD and MAY are normative within this document. The frozen implementation and genesis configuration remain authoritative where this paper summarizes rather than reproduces code.

- Consensus-, genesis-, tokenomics- and node-runtime behavior belongs to the Frozen L1 boundary.
- Wallet, explorer, SDK, operator control plane and recovery orchestration are external protocol consumers unless explicitly referenced as evidence mechanisms.
- An audit PASS is evidence about a named artifact; it is not equivalent to public-mainnet approval.
- Any future change that alters canonical bytes, state transition semantics, genesis identity or consensus acceptance requires explicit protocol versioning and renewed evidence.

2. Symbols and Primitive Domains

Symbol	Meaning
QNV	Human-readable native monetary unit
qnv-atom	Smallest ledger unit
D	Decimal precision = 5
U	Base units per QNV = 100,000
M	Maximum supply = 10,000,000,000 QNV
M_atom	Maximum ledger supply = 1,000,000,000,000,000 qnv-atom
H(x)	SHA-256 digest of byte string x unless a narrower domain specifies otherwise
C(x)	Canonical serialization of x
S	Authoritative AccountState
R(S)	Deterministic state root of S
h	Block height
q(n)	Byzantine quorum threshold for validator-set size n

M_atom = M * U = 10,000,000,000 * 100,000 = 10^15

0 <= amount <= M_atom

The amount domain is deliberately below the cross-client safe-integer canonicalization ceiling 2^53-1. Six decimal places would exceed that contract for the full supply; five do not.

3. Cryptographic and Canonicalization Primitives

Protocol identity depends on deterministic bytes. Qanovra therefore treats canonical serialization as a consensus-adjacent invariant rather than a convenience format.

- SHA-256 is used for protocol hashes and identity binding in the audited baseline.
- Signed application and validator messages use domain-separated signing bytes.
- Ed25519 is used by the verified signing surfaces in the protocol/evidence stack.
- Canonical object keys are ordered consistently across Python, JavaScript and Go using UTF-16 ordering semantics.
- Lone surrogate code points are rejected rather than normalized differently across clients.
- Integer values outside the allowed canonical range are rejected fail-closed.

digest(x) = SHA256(C(x))

The cross-client gate has exercised thousands of canonicalization edge cases, including non-ASCII keys, combining characters, emoji and surrogate rejection. Implementations MUST produce byte-identical canonical output for consensus-relevant objects.

Cryptographic agility and post-quantum scope. The v3.7.5 baseline uses SHA-256 and Ed25519 on the verified surfaces. It MUST NOT be described as post-quantum secure or as already resistant to a cryptographically relevant quantum adversary. Protocol versioning in signing semantics is the migration hook: any future signature/hash-suite transition requires an explicit protocol version, activation rules, new canonical vectors, key-migration/rotation procedures and renewed audit evidence. The v2.4 publication-candidate design note identifies NIST-standardized successor signature families such as ML-DSA and/or SLH-DSA as plausible future targets, together with a staged acceptance window for legacy and successor signatures; this remains a future-version migration objective rather than a property claimed for v3.7.5.

4. Network Identity and Genesis Binding

The Qanovra network identity is derived from the canonical genesis bytes. A node MUST NOT accept a configured Chain ID that is inconsistent with the genesis-derived identity.

Item	v3.7.5 value
Genesis hash	`78cefe355c7fb55c53dd173c9ca9d340306054a7fd5718e70d1d3193be7b93a3`
Chain ID	qnv-78cefe355c7fb55c53dd173c
Derivation	"qnv-" + first 24 hexadecimal characters of genesis hash
Genesis changed from v3.7.3/v3.7.4	No

chain_id = 'qnv-' || hex(SHA256(genesis_bytes))[0:24]

Genesis initialization is deterministic and idempotent for the same genesis hash. Outside laboratory profiles, synthetic genesis authorities are forbidden and a real genesis file is required.

5. Account Model and Namespaces

Qanovra uses one authoritative AccountState. Parallel balance ledgers are prohibited. Tokenomics is a policy layer over AccountState, not a second ledger.

Namespace	Purpose
40-hex user account identifier	Ordinary key-derived user account
qnv:system:vault:*	Protocol-controlled allocation vaults
qnv:system:*	Reserved protocol namespace, including fee/staking system accounts

The reserved qnv:system: prefix cannot be produced by the ordinary user-account derivation format. This creates structural separation between user-controlled and protocol-controlled balances.

vault_account_id(name, authority) = 'qnv:system:vault:' || SHA256(domain || name || authority)[0:40]

A direct ordinary transaction attempting to spend from a protocol vault MUST fail because no ordinary user key derives that system account.

Ordinary user-account derivation boundary. The current evidence set establishes a structurally separate ordinary user-account format and the reserved qnv:system: namespace, but does not provide enough canonical L1 evidence here to publish the exact public-key-to-account derivation function without guessing. Publication of that formula remains a re-audit requirement. Wallet-level address objects MUST NOT be silently substituted for the L1 account identifier: the exact L1 input encoding, hash/truncation rule and test vectors must be taken from the frozen L1 evidence set. User-account migration design note: because ordinary user accounts are key-derived, any future cryptographic-suite migration SHOULD provide an authenticated rebinding or migration transaction that binds the legacy account, the successor public key/account representation and replay-protected authorization by the legacy key; until such a future protocol version is standardized, holder migration would otherwise require explicit balance transfer.

6. State Model

Let S be the complete authoritative ledger state. At minimum, state contains account balances and nonces plus protocol metadata required to enforce supply and system invariants. The persisted chain tip and state root MUST reconcile on startup.

S' = Apply(S, B)

R_expected = B.header.state_root

Accept(B) only if R(S') = R_expected

Finalization is atomic across chain and state persistence. A failed state-root comparison or already-finalized height MUST leave both state and chain unchanged.

- Account balances MUST remain non-negative.
- Supply accounting MUST be checked at commit boundaries.
- State and chain persistence share an atomic transaction boundary in the remediated architecture.
- On restart, tip.state_root MUST equal the current authoritative state root.

7. Supply Invariant

Qanovra's monetary invariant is enforced against AccountState, not against a presentation-layer tokenomics object.

sum(account_balances) + burned_supply = expected_supply

expected_supply = M_atom

Transfers conserve supply. Slashing that is defined as burn increases burned_supply by the exact debited amount. Reward accrual MUST NOT create unbacked units; distributions must be funded from an authorized source or otherwise preserve the invariant.

Supply fuzzing in the v3.7.x remediation line repeatedly preserved the full 10 billion QNV equivalent across randomized state transitions.

8. Transaction Model

A transaction is a signed state-transition request bound to protocol identity. The exact field schema remains implementation-authoritative, but the audited behavior establishes the following acceptance dimensions:

Dimension	Required property
Sender	Must correspond to the signing key/account derivation
Network	Bound to Chain ID / genesis identity
Protocol	Signing bytes include protocol version
Nonce	Must satisfy account replay/ordering rules
Amount	Integer qnv-atom, bounded by protocol amount limits
Fee	Validated and accounted in state
Expiry	Transactions may carry an expiry-height bound
Signature	Must verify over canonical, domain-bound signing bytes

tx_id = H(C(transaction_envelope))

Verify(pk_sender, signing_bytes(tx), signature) = true

A transaction signed for another network identity MUST NOT validate on Qanovra. Protocol upgrades must account for in-flight transactions because protocol_version is part of transaction signing bytes.

9. Mempool and Admission

Admission is not finalization. A transaction MAY enter a bounded mempool only after basic structural, signature, replay, expiry and policy checks. Final validity is re-evaluated against the state used for block execution.

- Duplicate or stale transaction frames are rejected.
- Resource limits are bounded to prevent unbounded peer or request growth.
- Upgrade activation requires revalidation of pending transactions when signing semantics change.

10. Block and State-Transition Semantics

A block commits an ordered set of transactions and a resulting state root. The proposer constructs a candidate against the current state; validators independently validate the proposal before voting.

S_0 := current authoritative state

S_i := ApplyTx(S_{i-1}, tx_i), i = 1..k

B.header.state_root := R(S_k)

A valid finalization path performs preview/execution checks before the atomic commit. A second block at an already-finalized height MUST be rejected without mutating state.

11. Consensus Quorum

The verified quorum property is Byzantine-majority style and validator-count based in the audited baseline.

$q(n) = \text{floor}(2n/3) + 1$

$q(n) > 2n/3$

$2*q(n) - n > n/3$

The property has been gate-tested across a broad range of validator-set sizes. The validator-set machinery also enforces a minimum of four validators, giving the smallest configured set the ability to tolerate one Byzantine participant under the quorum model.

Important scope note: earlier audit evidence established that the consensus quorum counts validators rather than stake weight. Staking economics and consensus voting power must therefore not be conflated unless a future protocol version explicitly changes that rule.

Sybil-resistance boundary. Because q(n) counts admitted validators rather than stake weight, validator admission is a consensus-security boundary. The 100,000 QNV minimum self-stake is an economic alignment requirement; it is not the mechanism that gives a validator additional consensus weight and it is not, by itself, sufficient Sybil resistance. Qanovra v3.7.5 therefore does not claim permissionless validator admission. P0-06 must remain fail-closed and organizational/administrative independence must be evidenced before an operator enters the active validator set.

12. Validator Set and Reconfiguration

Validator-set transitions are authenticated state changes. A transition is bound to both the old and new set and requires joint verification. The transition machinery constrains churn and enforces the minimum validator count.

- Validator identifiers MUST be unique.
- Transition evidence MUST bind the exact old validator set.
- Both sides of a transition are verified rather than trusting planning output alone.
- Runtime consumers of the validator mapping MUST observe reconfiguration consistently; the remediated implementation updates the shared mapping in place.

13. Proposal, Vote and Finalization Safety

A proposal is eligible only if its proposer belongs to the active validator set. Votes that cannot be members of a valid certificate must not be allowed to corrupt finalization. Finalization requires a consistent certificate satisfying q(n).

Finalizable(B) := ValidProposal(B) AND |ValidVotes(B)| >= q(n)

The frozen baseline preserves safety-focused controls and atomic finalization. v2.4 makes no stronger liveness claim than canonical v3.7.5 implementation evidence supports. The paper does not infer a leader schedule or view-change protocol merely from quorum safety. If the canonical implementation cannot make progress under a failed or unavailable proposer in a given state, safety may remain intact while liveness is lost. The exact scheduling/timeout/view-change mechanism, if any, must be named from the frozen implementation in the re-audit evidence package.

Finality/reorg clarification. Multiple competing candidates or orphaned proposals may exist before finalization. Once a height has been finalized, a second block at that height is rejected without state mutation; explorer reorg/orphan awareness therefore refers to pre-finalization/canonical-selection observations, not a claim that finalized heights routinely reorganize.

14. Fees and Protocol Accounting

Fees are state-accounted amounts denominated in qnv-atom. Fee handling MUST preserve supply and MUST NOT use an unauthenticated or arbitrary destination. Protocol fee pools live in the reserved system namespace.

debit(sender) = transfer_amount + fee

credit(recipient) = transfer_amount

credit(protocol_fee_destination) = fee # unless a defined burn rule applies

All amount-bearing signed tokenomics messages are recursively bounded before canonical serialization. Public tokenomics methods independently apply amount bounds, preventing oversized values from escaping as generic canonicalization errors.

For v2.4 launch economics, protocol fee revenue assigned to network security is a funded reward source and is consumed before any authorized subsidy draw from the Community & network security allocation. This does not create QNV and does not relax the supply invariant. The exact fee-assessment function, minimum fee and any operator commission bounds are not recovered from the canonical L1 evidence currently attached to this document; they remain CANONICAL_EXTRACTION_REQUIRED and are not invented here.

15. QNV Monetary Units

Parameter	Normative value
Ticker	QNV
Base unit	qnv-atom
Decimals	5
Base units per QNV	100,000
Max QNV	10,000,000,000
Max qnv-atom	1,000,000,000,000,000

QNV(x_atoms) = x_atoms / 100000

atoms(x_QNV) = x_QNV * 100000

Consensus and ledger calculations SHOULD use integer qnv-atom values. Human-facing decimal formatting is a presentation concern and MUST NOT introduce floating-point ambiguity into signed or state-rooted values.

16. Genesis Allocation and Vault Policy

Genesis allocates the complete supply into policy-controlled destinations. The exact genesis file is authoritative. The public allocation summary currently records:

Allocation	Share	QNV
Community & network security	51.75%	5,175,000,000
Ambassador program	0.25%	25,000,000
Treasury & ecosystem	15.00%	1,500,000,000
Founder	15.00%	1,500,000,000
Core team	8.00%	800,000,000
Remaining canonical allocations	10.00%	1,000,000,000
Total	100%	10,000,000,000

The 10% remainder is intentionally not renamed here without the canonical genesis allocation schedule. A Yellow Paper must not manufacture an allocation label.

Allocation release policy introduced by v2.3: the Founder allocation of 15% is available from Genesis by design and has no protocol-enforced vesting/lockup claim. This is explicitly disclosed as liquidity/concentration risk. Individual Core Team grants from the 8% allocation are subject to a 48-month vesting schedule from grant date, with a 12-month cliff and monthly linear vesting over the remaining 36 months; ungranted balances remain in the designated allocation vault. The Core Team schedule is a launch-policy requirement and MUST NOT be represented as enforced until canonical vault/configuration evidence and tests demonstrate the release path.

17. Treasury Authorization

Treasury control is a protocol policy over AccountState. The legacy zero-threshold treasury runtime was removed. The active treasury policy requires exactly four approvals from seven configured keys and applies a timelock.

TreasuryThreshold = 4

TreasuryKeyCount = 7

Annual treasury caps are tracked as state rather than trusted from caller-provided values. Proposed spends reserve annual capacity so multiple pending proposals cannot collectively bypass the cap.

The exact timelock duration and annual-cap value are security-relevant canonical parameters. They were not recovered from the evidence available to this document and therefore remain CANONICAL_EXTRACTION_REQUIRED. Before final publication/re-audit, both values must be cited from the exact genesis/policy artifact and added to the evidence-to-claim matrix; no numeric value is guessed in v2.4.

18. Staking Model

The economic policy associated with Qanovra specifies a minimum delegation of 100 QNV, a minimum validator self-stake of 100,000 QNV, and a target staking ratio of 55-70%. v2.4 defines the target gross reward corridor as a funded, non-inflationary launch policy rather than an unconditional yield promise.

These policy targets do not imply that consensus voting is stake-weighted. The frozen consensus rule is described in Section 11. Stake is economically relevant through escrow, rewards and slashing policy.

staked_total = staking_escrow_balance

The exact unbonding duration is intentionally not guessed in this document; the canonical protocol configuration remains authoritative for that parameter.

Rewards MUST be paid only from already-funded AccountState balances. Protocol fee revenue assigned to network security is used first; an explicitly authorized draw from the Community & network security allocation may cover a funded shortfall. If funded balances are insufficient, the effective reward rate decreases. No reward path may mint QNV beyond the fixed 10,000,000,000 maximum supply.

Let s be the staking-participation ratio used by the launch reward policy. The v2.4 target gross rate is:

r_target(s) = 0.10, s <= 0.55

r_target(s) = 0.10 - (0.04/0.15)*(s-0.55), 0.55 < s < 0.70

r_target(s) = 0.06, s >= 0.70

Thus r_target(0.625) = 0.08. The effective paid rate is bounded by funded reward capacity and may be lower. If the 5,175,000,000 QNV Community & network security allocation were hypothetically used only for the policy-defined subsidy corridor, the indicative reserve life would be approximately 9.41 years at 55% staking, approximately 10.35 years at 62.5% staking, and approximately 12.32 years at 70% staking. These values follow the declared r_target(s) function rather than a flat-8% assumption. They are not reserve-life guarantees because the allocation has broader purposes and fee revenue can reduce reserve draw.

The exact unbonding duration remains CANONICAL_EXTRACTION_REQUIRED and must be cited from the canonical launch configuration before final publication. Gross APR is before any operator commission; validator commission is an operator-disclosed commercial parameter and is not consensus voting weight.

19. Slashing Semantics

Slashing distinguishes routine availability degradation from malicious or consensus-threatening behavior. For correlation tiers, c = involved_active_validators / active_validator_count; stake weight is not the unit. The slash principal base under the v2.4 publication-candidate policy is the offending validator's total bonded stake economically attributable to that validator at slash evaluation time, including validator self-stake and all active delegated stake assigned to that validator. The effective principal slash is effective_slash = max(base_slash, correlated_slash(c)); percentages are not additive or multiplicative. A temporary jail lasts at least one complete 10,000-block observation window. This duration is normative in blocks; a real-time equivalent is intentionally not published here until canonical block-cadence extraction is complete. Unjail requires expiry of the window, a signed operator request, a non-tombstoned validator key and successful current eligibility/health checks. A tombstoned key can never be unjailed.

Condition	Policy consequence
Normal/proportionate downtime	0% slash; reduced/lost rewards
<95% expected signatures over rolling 10,000 blocks	0.10% slash + temporary jail
Repeat downtime	0.25% slash + temporary jail
Chronic downtime	up to 0.50% slash + temporary jail
Double-sign / equivocation	5% base slash + permanent tombstone
Byzantine / consensus-threatening behavior	minimum 5% base slash
Correlated <1% of active validator count	5%
Correlated 1% to <5% of active validator count	10%
Correlated 5% to <10% of active validator count	20%
Correlated 10% to <20% of active validator count	50%
Correlated 20% to <1/3 of active validator count	75%
Systemic >=1/3 of active validator count	100%

Slash principal is defined as burn in the v2.4 launch policy and MUST be reflected in burned_supply so total-supply accounting remains exact. No alternative non-burn slash destination is claimed. Because delegated stake is part of the slash-principal base, delegators bear proportional validator-misconduct risk. These completed policy semantics require implementation/conformance evidence before network activation.

20. Privileged Application Authorization

State-mutating privileged modules are subject to explicit authorization semantics. Authorization is verified before replay consumption and before mutation. Signed messages use domain-separated canonical payloads.

- Invalid signatures MUST NOT consume a valid nonce.
- Replay windows are subject-scoped and bounded.
- Amount-bearing privileged messages MUST pass amount-bound validation before serialization.
- Machine-enforced verification semantics guard authorized modules against bypassing required validation paths.

21. Node Request Authentication

The remediated node request-authentication frame binds method, path, peer identity, nonce, issue time and request-body hash. Requests outside the freshness window or exact replays are rejected.

MAC_input = method || '\n' || path || '\n' || peer || '\n' || nonce || '\n' || issued_at || '\n' || SHA256(body)

The audited remediation used a +/-30 second freshness window for the authenticated node HTTP path. Transport-level validator communication additionally uses certificate-bound/mTLS-oriented controls in the validator transport layer.

22. Replay Protection

Replay resistance exists at multiple layers: account nonces for transactions, authenticated request nonces/frames for node APIs, and subject-scoped replay windows for privileged signed operations.

Accept(message, subject, nonce) only if Verify(message) succeeds AND nonce is fresh for subject

The ordering is normative: verify first, consume replay state second. This prevents an attacker from invalidating a future legitimate operation by submitting garbage with its nonce.

23. Protocol Upgrades

Protocol version is included in transaction signing semantics. Therefore an activation event can change which pending signatures are valid. Upgrade logic MUST explicitly define activation height, accepted version window and mempool revalidation behavior.

The frozen v3.7.5 baseline is not silently mutable. Any upgrade that changes consensus-relevant semantics creates a new versioned protocol state and requires renewed canonical vectors, genesis/network-identity analysis where applicable, tests and audit evidence.

24. Deterministic Cross-Client Contract

Qanovra maintains canonicalization implementations across Python, JavaScript and Go. For every consensus-relevant object x:

C_python(x) = C_javascript(x) = C_go(x)

A mismatch is a release failure. The contract includes key ordering, integer bounds, Unicode handling and rejection behavior, not merely successful serialization of common ASCII objects.

25. Persistence and Crash Consistency

The chain and account state must behave as one logical ledger. Finalization uses a shared persistence boundary so that a crash cannot commit account changes without the corresponding block or vice versa.

- Preview state transition.
- Compare computed state root with proposal commitment.
- Begin atomic persistence transaction.
- Apply block state changes without independent commit.
- Append chain record without independent commit.
- Re-check root/supply invariants.
- Commit once.
- On startup, reconcile tip and state root.

26. Security Invariants

Invariant	Required condition
Network identity	Chain ID derived from canonical genesis hash
Canonicalization	Byte-identical across qualified clients
Supply	balances + burned = expected supply
State atomicity	invalid/failing finalization leaves chain and state unchanged
Height uniqueness	already-finalized height cannot be finalized again
User/system separation	ordinary key derivation cannot produce qnv:system:*
Treasury	exact 4-of-7 policy
Replay	invalid signature cannot consume legitimate nonce
Amount safety	signed monetary values bounded before canonicalization
Validator quorum	q(n) > 2n/3
Validator floor	n >= 4 in validator-set transitions

27. Threat Model

The protocol threat model includes adversarial clients, malicious or faulty validators, replay attempts, malformed canonical objects, oversized monetary values, crashes during persistence, stale configuration, compromised operational credentials, long-range/history manipulation attempts, eclipse isolation, network partitions and targeted denial-of-service. It does not assume that software alone can guarantee physical custody, legal independence of operators or perfect network availability.

Validator-set transitions bind the exact previous and new set and require authenticated transition evidence; this provides a verifiable chain of set changes from an accepted trust anchor and is relevant to long-range/history defense. It is not represented as a complete weak-subjectivity solution without corresponding sync/checkpoint evidence. Eclipse and partition resilience depend on real peer/operator diversity and network deployment; DoS resistance depends on bounded admission/resource controls and external infrastructure qualification. Safety claims remain bounded by the implemented consensus model, and liveness can fail under sufficiently adverse network/proposer conditions without implying a safety break.

28. Verification and Audit Evidence

The v3.7.5 technical handoff records 34/34 release checks passing, 284 Python tests with zero failures, cross-client genesis/canonicalization evidence, and supply-fuzz evidence preserving the full ledger supply. The candidate manifest contains 501 entries with no unmanifested files in the reviewed handoff.

These measurements support the frozen baseline but do not replace formal independent attestation. The release flags for broader deployment remain conservative until external gates are completed.

29. External Gates Outside the Protocol Proof

- Formal independent attestation of the v3.7.5 L1 candidate.
- Real 4-of-7 key ceremony using qualified non-exporting HSM/KMS/Vault infrastructure.
- Independent validator-operator onboarding.
- Genuinely geographic disaster-recovery rehearsal.
- Independent security assessment/red-team qualification where required.
- Legal and regulatory approval for the intended launch context.

A software test cannot truthfully mark these real-world events complete.

30. Conformance Requirements

A conforming Qanovra v3.7.5 implementation MUST reproduce the canonical network identity and MUST preserve every invariant in Section 26. It MUST fail closed on unsupported canonical values, invalid signatures, mismatched genesis identity, state-root mismatch, unauthorized privileged mutations and supply violations.

A client MAY implement additional read-only functionality, indexing or observability, provided it does not alter consensus-relevant bytes or acceptance semantics.

31. Known Specification Boundaries

Known specification boundaries are explicit rather than guessed. The exact unbonding duration, treasury timelock duration, annual treasury cap, fee-assessment/minimum-fee parameters, block/transaction/network resource limits, exact ordinary user-account derivation formula, and canonical liveness scheduling/view-change details remain CANONICAL_EXTRACTION_REQUIRED unless supplied in the evidence set. The descriptive label of the remaining 10% genesis allocation remains governed by the canonical genesis allocation schedule. v2.4 adds publication-policy decisions for reward funding, Core Team vesting, slash-principal basis and user-account migration design boundaries, but does not claim runtime enforcement until evidence demonstrates it.

A Yellow Paper should be more precise than a Whitepaper, not more imaginative: unresolved canonical values are publication tasks, not invitations to invent them.

32. Frozen Baseline Summary

Property	v3.7.5 baseline
L1	Qanovra v3.7.5
Genesis	`78cefe355c7fb55c53dd173c9ca9d340306054a7fd5718e70d1d3193be7b93a3`
Chain ID	qnv-78cefe355c7fb55c53dd173c
Unit	QNV / qnv-atom
Decimals	5
Max supply	10 billion QNV / 10^15 qnv-atom
Ledger	Single authoritative AccountState
Treasury	Exact 4-of-7
Consensus quorum	floor(2n/3)+1, validator-count based
Validator minimum	4
Cross-client contract	Python / JavaScript / Go canonical parity
Release state	Frozen technical baseline; external launch gates remain separate

33. Conclusion

Qanovra v3.7.5 is specified as a deterministic, genesis-bound state machine with a single monetary ledger, explicit system-account boundaries, cross-client canonicalization, Byzantine-majority quorum rules, atomic finalization and policy-controlled economic operations. Its strongest design principle is that critical claims are expressed as invariants that can be tested against artifacts rather than as informal assurances.

The canonical implementation, genesis configuration and signed release evidence remain the final authority for the frozen L1. This Yellow Paper v2.4 is the technical companion to the Qanovra Whitepaper v2.4 and is a publication candidate: high-severity document-policy gaps are closed, while canonical values not actually recovered remain explicitly queued for evidence extraction before final publication.

Appendix A - Core Equations

M_atom = 10^10 * 10^5 = 10^15 qnv-atom

chain_id = 'qnv-' || hex(SHA256(genesis_bytes))[0:24]

q(n) = floor(2n/3) + 1

q(n) > 2n/3

2*q(n) - n > n/3

S' = Apply(S, B)

Accept(B) => R(S') = B.header.state_root

sum(balances) + burned_supply = expected_supply = 10^15

C_py(x) = C_js(x) = C_go(x)

Appendix B - Document Control

Field	Value
Document	Qanovra Yellow Paper
Version	2.4.2 - Publication Layout Final Candidate
Status	Technical Specification / Publication Layout Final Candidate
Baseline date	21 August 2026 substantive baseline; layout revision 23 August 2026
Protocol baseline	Qanovra L1 v3.7.5 Frozen L1
Companion document	Qanovra Whitepaper v2.4.2 Publication Layout Final Candidate
Change rule	Consensus/genesis/tokenomics/runtime changes require versioning and renewed evidence
Supersedes	Yellow Paper v2.4.1 Technical Specification / Publication Layout Revision (delta-audited, 23 August 2026)
Document history	v2.0 initial technical audit candidate; v2.2 readiness refresh; v2.3 external-audit remediation; v2.4 publication-candidate closure of re-audit findings; v2.4.1 typography/pagination revision; v2.4.2 hash-cell typography micro-fix only

Appendix C - Evidence-State Separation (Non-Normative)

This appendix records the implementation/evidence state surrounding the frozen protocol. It is non-normative and does not modify Qanovra L1 v3.7.5 consensus, genesis, state-transition or economic semantics.

Gate / area	Current status	Audit treatment
Protocol baseline	v3.7.5 frozen; genesis and Chain ID unchanged.	Normative baseline remains implementation/genesis authoritative.
Cross-platform reproducibility	Windows/macOS comparison reports matching runtime and vendor trees and zero unexplained differences.	Evidence property; not a consensus rule. Final observer attestation belongs in the audit package.
Real-device qualification	Windows/Android/iOS matrix recorded complete in the current release line.	Wallet/client qualification; outside L1 consensus semantics.
Operational runbooks	Current project line records P1-04 runbooks completed/executed.	Operational evidence; not proof of production infrastructure.
Independent validators	Five onboarding packages ready; 0/5 assigned and 0/5 admitted.	External network property remains OPEN / BLOCKED_REAL_INFRA.
Production HSM / ceremony	Dry-run/technical threshold evidence does not by itself establish production-qualified non-exportable signing.	Require live provider/key evidence and signed ceremony artifacts for closure.

Audit note: Only evidence-backed states are represented as complete. Prepared packages, internal rehearsals and technical prechecks are not treated as substitutes for independent real-world execution.

Appendix D - Audit Reproducibility Requirements

- Verify the v3.7.5 genesis bytes and recompute the published genesis hash and Chain ID.
- Re-run the canonicalization parity vectors across the qualified Python, JavaScript and Go implementations.
- Re-run supply, state-root, persistence and validator-quorum invariants against the frozen artifact set.
- Verify that validator quorum remains validator-count based and that staking economics are not silently interpreted as consensus voting weight.
- Verify exact 4-of-7 treasury policy from canonical genesis/policy artifacts.
- Verify document claims against named evidence artifacts; operational REAL_INFRA gates must not be closed by synthetic data, templates or examples.
- Generate final DOCX/PDF SHA-256 values only after formatting freeze and record them in the audit handoff manifest.
- Re-audit v2.4 reward-funding/APR function and verify that reward payments cannot exceed funded AccountState balances.
- Verify canonical enforcement for Core Team vesting and the completed slashing/jail policy before treating those v2.4 launch-policy statements as active protocol behavior.
- Extract and cite exact unbonding, treasury timelock/annual cap, fee/resource-limit, user-account derivation and liveness mechanism parameters from canonical evidence.
- Do not describe v3.7.5 as post-quantum secure; any future cryptographic migration requires explicit protocol versioning and renewed audit vectors.

Appendix E - v2.4 Audit & Re-Audit Closure Matrix

This appendix maps the external v2.2 audit findings and the v2.3 re-audit findings to the v2.4 publication-candidate state. CLOSED_DOC means the ambiguity or policy omission is resolved in the documents. IMPLEMENTATION_EVIDENCE_REQUIRED means the policy is now explicit but MUST NOT be represented as enforced until canonical configuration/code evidence and tests confirm it. CANONICAL_EXTRACTION_REQUIRED means the document intentionally refuses to invent a frozen-runtime value that was not recovered from the evidence set.

ID	v2.4 status	Remediation	Required next evidence
PAP-H1	CLOSED_DOC / IMPLEMENTATION_EVIDENCE_REQUIRED	Non-inflationary fee-first reward funding, target function and policy-consistent funding-horizon math added.	Bind launch policy to canonical reward accounts/config and tests.
PAP-H2	CLOSED_DOC / IMPLEMENTATION_EVIDENCE_REQUIRED	Founder no-lock status disclosed; Core Team 48m/12m-cliff policy defined.	Prove Core Team vault/grant enforcement.
PAP-H3	CLOSED_DOC / IMPLEMENTATION_EVIDENCE_REQUIRED	Count-based tiers complete through >=1/3; slash-principal basis, max() rule, burn and jail/unjail semantics defined.	Conformance-test implementation.
PAP-M1	OPEN_CANONICAL_EXTRACTION	No guessed unbonding/timelock/cap values.	Extract exact values and artifact hashes.
PAP-M2	CLOSED_DOC	Controlled admission identified as Sybil boundary for count-based quorum.	Verify P0-06 admission controls.
PAP-M3	PARTIAL / CANONICAL_EXTRACTION	Safety-vs-liveness limitation explicit.	Name exact v3.7.5 scheduling/view-change behavior.
PAP-M4	OPEN_CANONICAL_EXTRACTION	No fabricated TPS/block/network limits.	Add canonical block/tx/resource/network parameters.
PAP-M5	PARTIAL / CANONICAL_EXTRACTION	Fee-to-reward relationship and gross-APR/commission distinction defined.	Extract fee assessment/minimum and any commission bounds.
PAP-M6	OPEN_CANONICAL_EXTRACTION	Wallet address objects explicitly not substituted for L1 account derivation.	Publish exact L1 public-key/account formula and vectors.
PAP-M7	CLOSED_DOC	Long-range/history, eclipse, partition and DoS included with bounded mitigations.	Operational evidence remains separate.
PAP-M8	CLOSED_DOC	Explicit no-PQ-security claim for SHA-256/Ed25519 v3.7.5; migration design note adds plausible successor suites and versioned transition concept.	Align external marketing.

ID	v2.4 status	Remediation	Required next evidence
PAP-M9	CLOSED_DOC	Reorg/orphan terminology restricted to pre-finalization candidates.	Conformance check only.
PAP-M10	PARTIAL	Document lineage stated in companion WP; Heritari repin remains separate.	Detailed v3.6.9->v3.7.5 changelog + Heritari review.
PAP-M11/N1/N3/N4	CLOSED_DOC / HANDOFF	Legal naming/separation in WP; labels, metadata and final-hash process normalized.	Final legal review remains external.
PAP-N2	PARTIAL	10% label unified; no label invented.	Attach canonical genesis-allocation schedule.
REA-M1	CLOSED_DOC	Funding-horizon illustration recalculated from the declared APR function (55%: 9.41y; 62.5%: 10.35y; 70%: 12.32y).	Bind reward-funding policy to canonical reward accounts/config and tests.
REA-M2	CLOSED_DOC / IMPLEMENTATION_EVIDENCE_REQUIRED	Slash-principal basis defined as total bonded stake attributable to the offending validator, including delegated stake.	Conformance-test slashing base and delegator accounting.
REA-M3	CLOSED_DOC / FUTURE_PROTOCOL_REQUIRED	Migration design note adds target PQ successor families, a versioned transition concept, and an authenticated user-account rebinding requirement.	Future protocol-version artifact, vectors and implementation evidence before any suite migration.
REA-N1	CLOSED_DOC	Formation timeline clarified in the companion Whitepaper (initiation vs notarized formation).	Update registry status when available.
REA-N2	CLOSED_ARTIFACT	Final filenames aligned with the manifest; PDF handoff included in the v2.4 package.	None beyond handoff verification.
REA-N3	PARTIAL / CANONICAL_EXTRACTION	Jail duration remains normatively block-based; a real-time equivalent is intentionally not guessed until block-cadence extraction is complete.	Extract canonical block/resource cadence and update descriptive real-time guidance.