DYTALLIX

Quantum-Secure, Open-Source, Future-Ready

Foundational Whitepaper

January 2026

Abstract

Dytallix is a Layer 1 blockchain engineered for a world where cryptographically relevant quantum computers (CRQCs) exist.

It is post-quantum cryptography (PQC) native: consensus votes, transaction authorization, and network handshakes rely on NIST-standardized PQC primitives rather than classical elliptic-curve cryptography.

This foundational document explains the project's purpose, principles, and design constraints; summarizes the protocol architecture at a systems level; and integrates the core technical and economic mechanics required to operate a PQC-first chain.

Table of Contents

1. Introduction
2. Foundational Principles
3. Design Goals and Non-Goals
4. Competitive Landscape
5. System Overview
6. Cryptographic Baseline
7. Account and Address Model
8. Networking Layer
9. Consensus: Nakamoto-Flow
10. Execution Layer: Dytallix Virtual Machine (DVM)
11. Token Model and Incentive Design
12. Governance and Parameter Control
13. Operational Posture
14. Specification Completion Checklist
15. Business Case and Use Cases
16. Regulatory Positioning and Compliance Strategy
17. Roadmap
18. Products and Services
19. Funding Strategy
20. Risk and Security Posture
21. Call for Contributors
22. Legal Disclaimer
23. Glossary
24. References

1. Introduction

The near-term problem is not speculative: adversaries can capture encrypted traffic today and decrypt it later ("harvest now, decrypt later"). Any data whose confidentiality must survive beyond the "Y2Q" horizon is already at risk if transmitted or stored under classical public-key cryptography.

In distributed systems, the threat is existential: if an attacker can derive private keys from public keys at scale, they can forge transactions, hijack validator identities, and rewrite economic history.
Dytallix is built to remain secure when classical assumptions fail.

2. Foundational Principles

This section sets the non-negotiable principles that constrain every technical and economic decision.

They define what Dytallix optimizes for and what it will not trade away.

2.1 PHILOSOPHICAL IMPERATIVE

Satoshi invented Bitcoin as an exit from fiat currency debasement and centralized control: a system ruled by mathematics rather than decree. That promise remains incomplete as financial rails and digital identity are increasingly mediated by centralized checkpoints.

Quantum computing introduces a deeper threat: if the foundational mathematics fail, political debates about regulation and access become irrelevant.

Dytallix exists to answer that threat.

We assert that quantum-secure, permissionless transactions are a requirement for preserving human agency in a post-quantum era.

If the cryptography breaks, sovereignty and the ability to transact independently vanish.

2.2 ZERO-LEGACY MANDATE

Backward compatibility is not neutral. It preserves assumptions that were made under weaker cryptographic threat models.

Dytallix adopts a zero-legacy mandate: the protocol rejects classical ECDSA-based accounts or “hybrid” transitions as valid.

This avoids building permanent dependencies on compromised primitives.

2.3 OPEN-SOURCE IMPERATIVE

A PQC chain cannot rely on secrecy or trust in a centralized maintainer. PQC primitives are complex and sensitive to implementation errors; the only credible security posture is public auditability.

Dytallix is open source because:

- **Auditability**: security claims must be verifiable by independent reviewers.
- **Forkability**: any party can reproduce and evolve the system without permission.
- **Many-eyes property**: global inspection is structurally superior to black-box trust.

3. Design Goals and Non-Goals

This section defines the target outcomes and the explicit exclusions that keep scope disciplined. The goal is to prevent hidden assumptions from turning into protocol debt.

3.1 GOALS

- PQC-native security across consensus, networking, and transactions.
- Practical throughput despite kilobyte-class signatures.
- Economic spam resistance tuned for PQC bandwidth burdens.
- Crypto-agility: controlled upgrades when new attacks or standards emerge.
- Operational simplicity for validators and developers.

3.2 NON-GOALS

- Perfect privacy by default at all layers (this is a roadmap domain; the baseline focuses on quantum-secure authenticity and transport confidentiality).
- Compatibility with legacy elliptic-curve wallets or signing systems.

4. Competitive Landscape

The blockchain ecosystem is currently bifurcated into two categories:

- **Legacy Incumbents** that are fundamentally vulnerable to quantum attack.
  Hybrid Challengers attempting to patch post-quantum security onto older architectures.
- Dytallix represents a third category: **Native PQC**. By rejecting legacy compatibility, Dytallix eliminates the "cryptographic debt" that constrains competitors.

4.1 COMPARATIVE ANALYSIS MATRIX

The following table contrasts Dytallix against leading legacy chains and emerging PQC-hybrid competitors.

Feature	Dytallix	Ethereum (ETH)	QANplatform (QANX)	Quranium (QUR)	Algorand (ALGO)
Primary Architecture	Native PQC (Zero Legacy)	Legacy ECC (Optimistic Retrofit)	Hybrid (EVM Compatible)	Hybrid (PoW/DAG)	Legacy + State Proofs
L1 Consensus Security	ML-DSA (Dilithium)	ECDSA (Vulnerable)	Proof-of-Randomness (Lattice)	SLH-DSA / Hybrid	Pure PoS (Falcon State Proofs)
VM Environment	WASM (Rust/C++)	EVM (Solidity)	Hyperpolyglot (Multi-lang)	EVM Compatible	AVM (TEAL)
Legacy Compatibility	Rejected (Airlock Only)	Native	Native (EVM)	Native (EVM)	Native
Identity Standard	PQC-Native (D-Addr)	Classical (20-byte)	Hybrid	Hybrid	Classical (Ed25519)
HNDL Defense	Native (ML-KEM Transport)	None (TLS dependent)	Lattice-Link	Unknown	State Proofs Only
Status	Devnet	Mainnet	Testnet	Mainnet	Mainnet

4.2 CATEGORY 1: LEGACY INCUMBENTS (ETHEREUM, SOLANA, BITCOIN)

The current market leaders rely on elliptic curve cryptography (ECC) or RSA, specifically ECDSA (secp256k1) and Ed25519.

- **Status:** Dominant market share and liquidity.
- **Strengths:** Massive developer ecosystems, tooling maturity, and network effects.
- **Weaknesses: Existential Vulnerability.** Their base-layer signature schemes are broken by Shor's algorithm.
- **The "Retrofit" Trap:** These chains plan to address Y2Q via "soft forks" or Layer 2 abstraction. This creates a **"Hybrid Vulnerability"**: as long as the L1 accepts legacy keys to support old accounts, the chain remains vulnerable to a quantum attacker who can derive private keys from public historical data. Migrating these chains involves "wrapping" trillions of dollars of assets, a process fraught with operational risk (loss of keys, bridge hacks).

4.3 CATEGORY 2: HYBRID & EVM-COMPATIBLE CHALLENGERS (QANPLATFORM, QURANIUM)

A new wave of L1s has emerged specifically to address the quantum threat. However, most prioritize **EVM Compatibility** to lower the barrier to entry for developers.

- **Status:** Emerging / Testnet phase.
- **Strengths:** Easier onboarding for Solidity developers; support for existing tooling (Metamask).
- **Weaknesses: Legacy Debt.** By maintaining EVM compatibility, these chains often inherit the EVM's address limitations (20-byte addresses are too small for hash-based PQC commitments) and sequential execution bottlenecks. Furthermore, "Hybrid" states—where PQC and ECC accounts coexist—violate the principle of **Cryptographic Uniformity**, leaving the network as strong as its weakest link.

4.4 CATEGORY 3: PQC-ADJACENT INNOVATION (ALGORAND)

Algorand has pioneered the use of **Falcon** (a lattice-based signature) for "State Proofs," allowing the chain to attest to its state in a quantum-secure manner for bridging.

- **Status:** Live Mainnet.
- **Strengths:** High throughput, instant finality, academic rigor.
- **Weaknesses: Partial Implementation.** While State Proofs are quantum-secure, the core user accounts and transaction authorizations still typically rely on Ed25519 (ECC). This protects the *history* of the chain but not the *user funds* or *active consensus votes* from a direct Shor's algorithm attack on individual accounts.

4.5 THE DYTALLIX ADVANTAGE: ARCHITECTURAL PURITY

Dytallix is unique in its refusal to compromise security for convenience.

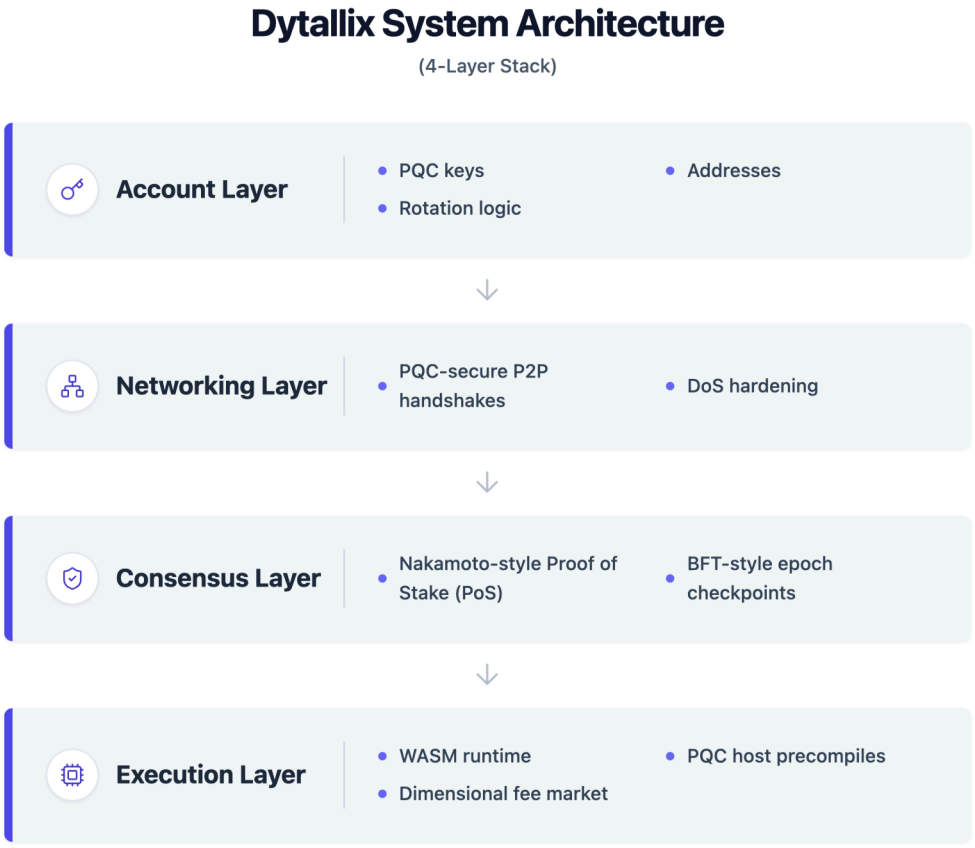
- **Zero-Legacy Identity:** Unlike EVM hybrids, Dytallix addresses (D-Addr) are designed natively for PQC key commitments (Bech32m-BLAKE3). This prevents the "address exhaustion" attacks that threaten retrofitted chains.
- **WASM Performance:** Lattice cryptography requires heavy matrix operations. The EVM is ill-suited for this math. Dytallix's WASM runtime with SIMD (AVX-512) precompiles allows for native-speed verification of ML-DSA signatures, solving the "throughput vs. security" dilemma that plagues hybrid chains.

- **The Airlock Strategy:** Rather than bridging contaminated assets, Dytallix forces a "clean break" via the QuantumVault. This ensures that every asset on Dytallix is **"Born Secure,"** protecting the economic integrity of the system even if there is turbulence during Y2Q.

5. System Overview

Dytallix is composed of four interacting layers:

1. Account layer: PQC keys, addresses, and rotation logic.
2. Networking layer: PQC-secure P2P handshakes and DoS hardening.
3. Consensus layer: Nakamoto-style PoS with BFT-style epoch checkpoints.
4. Execution layer: WASM runtime with PQC host precompiles and a dimensional fee market.



6. Cryptographic Baseline

Dytallix standardizes NIST PQC algorithms as defaults:

- **ML-DSA-65 (FIPS 204 / Dilithium3-derived)** for account authorization and validator voting. Deterministic signatures are used to prevent nonce-reuse failures.
- **ML-KEM-768 (FIPS 203 / Kyber768-derived)** for node handshakes and privacy envelopes.
- **BLAKE3** for state trie construction and content addressing (performance on parallel hardware).
- **SHAKE256 (FIPS 202)** within **ML-DSA/ML-KEM** routines and for the final block hash (strict NIST compliance).
- **SLH-DSA-SHAKE-192s (FIPS 205 / SPHINCS+-derived)** as an optional cold-storage signature system for long-horizon security diversity.

6.1 CRYPTO-AGILITY

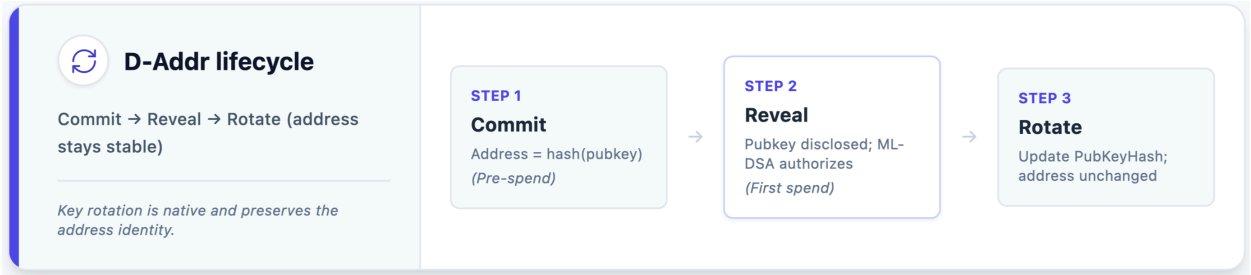
PQC is not treated as “final.”

Dytallix maintains an algorithm lifecycle model (e.g., Experimental → Active → Deprecated → Revoked) and defines governance-controlled transitions, including emergency deprecation under demonstrated breaks.

7. Account and Address Model

This section describes how identities and authorization are represented on-chain, including how Dytallix minimizes public-key exposure and enables protocol-level key rotation without breaking address continuity.

7.1 DYTALLIX ADDRESS (D-ADDR)



A Dytallix address is a 32-byte hash of a PQC public key encoded for user safety:

- D-Addr = Bech32m(BLAKE3(ML-DSA-65_PubKey)) with HRP dyt

Pre-spend safety: until an address spends, its public key remains hidden behind a 256-bit hash.

Post-spend security: once a transaction reveals the public key, security relies on ML-DSA-65.

7.2 NATIVE KEY ROTATION

Accounts support key rotation without changing the public address.

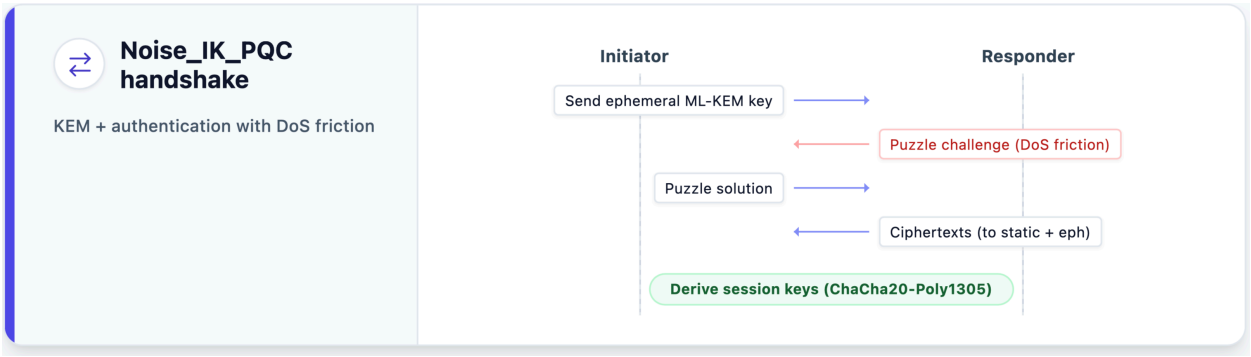
- The state maps D-Addr → Current_PubKey_Hash.
- Users can update the underlying PQC key (including migrating to a different PQC scheme) while preserving the address.

8. Networking Layer

This section summarizes the network security model: quantum-secure transport, handshake structure, and defensive measures that prevent PQC verification costs from becoming a DoS lever.

8.1 PQC-NOISE HANDSHAKE (NOISE_IK_PQC)

Dytallix implements a Noise-derived handshake where classical Diffie–Hellman is replaced by KEM + authentication.



High-level flow:

- Initiator generates ephemeral ML-KEM keypair and sends the ephemeral public key.
- Responder encapsulates shared secrets against the initiator's static identity key and ephemeral key and returns ciphertexts.

- Both sides derive a session key and use **ChaCha20-Poly1305** for symmetric encryption.

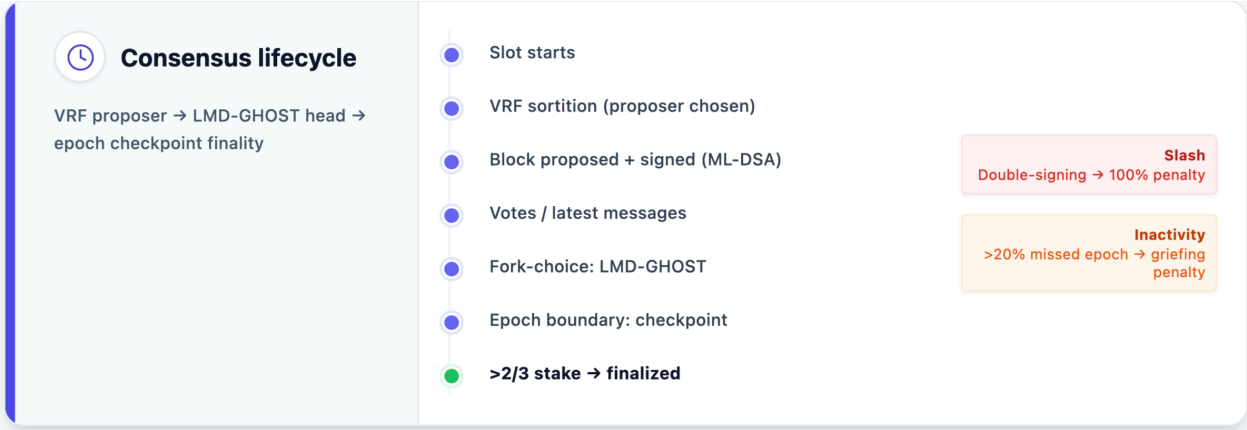
8.2 DOS MIGRATION

PQC verification is heavier than ECC; Dytallix hardens the handshake path:

- Pre-handshake puzzle: a small memory-hard client puzzle (e.g., Equihash-lite) prior to ML-KEM decapsulation.
- Peer scoring: peers that deliver invalid PQC artifacts are penalized and disconnected.

9. Consensus: Nakamoto-Flow

Dytallix uses a Nakamoto-style proof-of-stake chain rule augmented with BFT epoch checkpoints.



9.1 TIME STRUCTURE

- **Slot**: 2 seconds.
- **Epoch**: 100 slots (~3.33 minutes).

Validator sets and weights update at epoch boundaries.

9.2 LEADER ELECTION VIA VRF

Leader selection is private until the slot begins.

A validator computes:

$$y = VRF_s k(\eta_t || slot_t)$$

and is eligible if y meets a stake-weighted threshold.

The block header includes a proof that allows public verification without revealing the secret key.

9.3 BLOCK VALIDATION FLOW

1. Verify the VRF proof and proposer ML-DSA signature on the header.
2. Verify transactions using **AVX2-optimized batch verification**.
3. Apply fork choice (LMD-GHOST) using the latest stake-weighted votes.

9.4 FINALITY GADGET: EPOCH CHECKPOINTS

At epoch end, a BFT committee signs a **Checkpoint Block**

- **Justification threshold**: $> \frac{2}{3}$ of total active stake.
- **Aggregation model:** because ML-DSA does not support constant-size aggregation, Dytallix uses a **Bitfield-Compressed Merkle Proof**: a signer bitfield plus a single aggregate root.
- **Finality**: once justified and finalized, deep reorgs are not possible without slashable equivocation.

9.5 SLASHING

- **Equivocation (double-signing):** 100% stake slash.
- **Liveness faults:** validators missing more than 20% of an epoch incur a griefing penalty (0.1% slash).

10. Execution Layer: Dytallix Virtual Machine (DVM)

This section explains the execution environment and the resource-accounting model that makes PQC-heavy verification economically sustainable. It covers the WASM runtime, native PQC primitives, and fee mechanics.

10.1 WASM RUNTIME

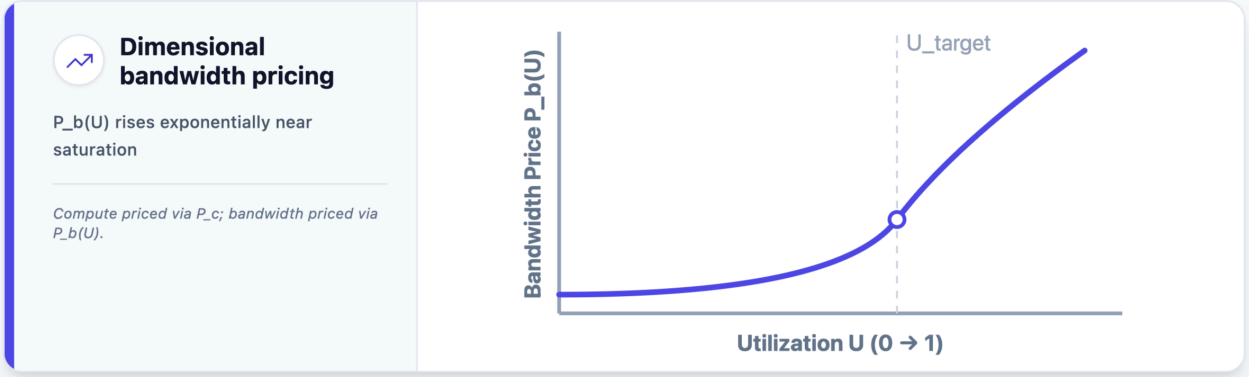
DVM is a high-performance **WASM** execution environment.

- Sandboxed memory model reduces common VM exploitation classes.
- Parallel execution is supported for disjoint-state transactions.

10.2 DIMENSIONAL FEE MARKET (PQC-AWARE)

A transaction fee calculates compute and bandwidth prices separately.

- **Total fee:** $F_{tx} = C_{gas} \cdot P_c + B_{gas} \cdot P_b(U)$
- **C_gas:** compute units (WASM instruction count).
- **B_gas:** bandwidth units (bytes of signature + calldata).



Bandwidth price scales exponentially by block utilization:

$$P_b(U) = P_{base} \cdot e^{\alpha (U - U_{target})}$$

This defends against PQC signature spam and state bloat by making sustained high-bandwidth load economically prohibitive.

10.3 PQC PRECOMPILES

To avoid contract bloat, DVM exposes native host functions:

- **0x10:** verify_mldsa65(pubkey, msg, sig) -> bool
- **0x11:** verify_slhdsa_shake(pubkey, msg, sig) -> bool
- **0x12:** kem_decapsulate(privkey, ciphertext) -> shared_secret

10.4 STATE COMMITMENT

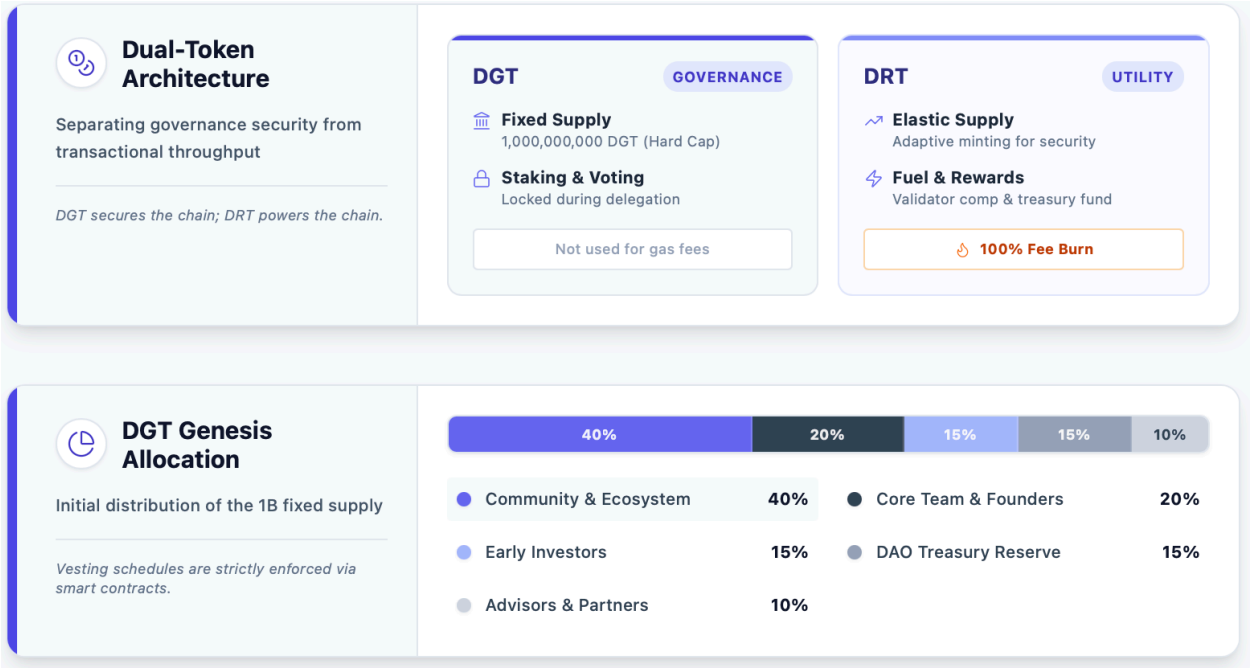
Global state is committed via a **Sparse Merkle Tree** optimized for 32-byte keys. Large PQC keys are hashed down to standard 256-bit addresses prior to insertion.

11. Token Model and Incentive Design

This section defines the core economic functions governing supply, fees, emissions, and reward distribution.

11.1 DUAL-TOKEN ARCHITECTURE

Dytallix separates governance security from transactional throughput via two tokens.



11.2 FEE FUNCTIONS (BURN DYNAMICS)

Let a transaction consume compute and bandwidth:

- $C_{gas}(\text{tx})$: compute units
- $B_{gas}(\text{tx})$ bandwidth units (bytes)

Let prices be:

- P_c : compute price
- $P_b(U)$: bandwidth price as a function of utilization U

Transaction fee (in DRT):

$$F_{tx} = C_{gas}(\text{tx}) \cdot P_c + B_{gas}(\text{tx}) \cdot P_b(U)$$

Burn per block:

$$B_{block} = \sum_{\text{tx} \in \text{block}} F_{tx}$$

Burn per time window (epoch/day):

$$B_{window} = \sum_{b \in \text{window}} B_{block}(b)$$

11.3 ADAPTIVE EMISSION CONTROLLER (PID)

DRT issuance is governed by a PID-controlled emission controller that reacts to utilization.

Let U_{target} be the target utilization and $U(t)$ the observed utilization:

We define the utilization error as:

$$e(t) = U_{target} - U(t)$$

Interpretation:

- If $U(t) > U_{target}$ then $e(t) < 0$ so the controller tends to reduce emissions.
- If $U(t) < U_{target}$ then $e(t) > 0$, so the controller tends to increase emissions.

The PID controller computes an unconstrained emission recommendation:

$$E^*(t) = E_{base} + K_p\, e(t) + K_i \sum_{j=t-W}^{t} e(j) + K_d\Big(e(t) - e(t-1)\Big)$$

The recommendation is then bounded to safety limits:

$$E(t+1) = \min\left(E_{max},\ \max\left(E_{min}, E^*(t)\right)\right)$$

Where:

- $E(t)$ is the DRT emitted per block (or per epoch, depending on implementation)
- W is the integral window size
- K_p, K_i, K_d are PID gains
- E_{min} and E_{max} are governance-set safety bounds

11.4 EMISSION DISTRIBUTION FUNCTION

For each emission event $E(t)$ the distribution is:

- Validator rewards: $R_{val}(t) = 0.40 \cdot E(t)$
- Staker rewards: $R_{stake}(t) = 0.30 \cdot E(t)$
- Treasury funding: `$R_{treasury}(t) = 0.30 \cdot E(t)$

11.5 STAKING REWARD ACCOUNTING (REWARD INDEX)

Let:

- S_{total} = total delegated (staked) DGT
- S_i = DGT delegated by delegator i
- R` = global reward index (fixed-point)
- SCALE = fixed scaling constant (e.g., 1e18)

On each distribution event:

$$R \leftarrow R + \frac{R_{stake}(t) \cdot \mathrm{SCALE}}{S_{total}}$$

Delegator accrual since last claim:

$$\Delta_i = \frac{S_i \cdot (R - R_{i,prev})}{\mathrm{SCALE}}$$

11.6 NET SUPPLY FUNCTION (MINT VS BURN)

Let:

- $M_{window} = \sum E(t)$(minted DRT in a window)

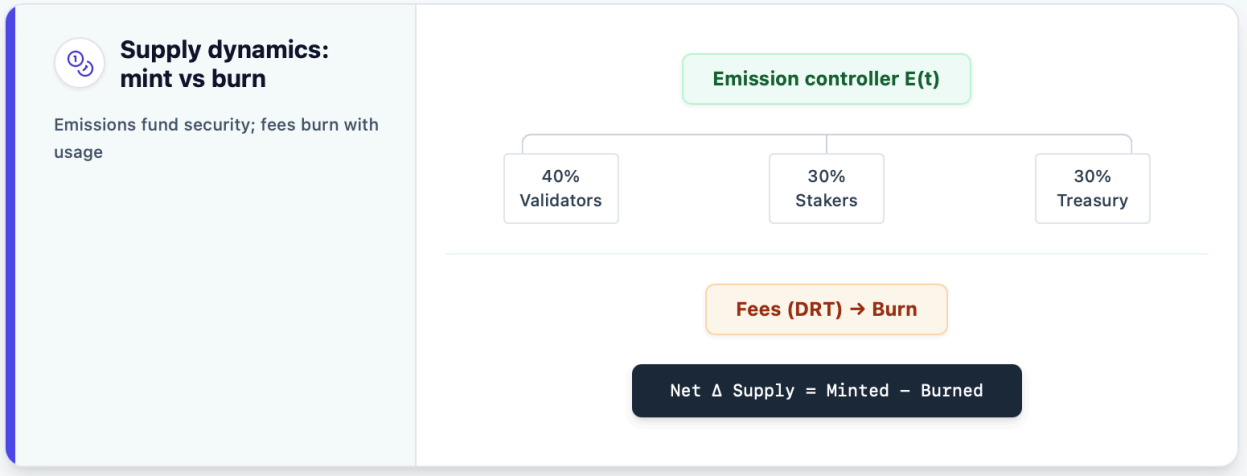
- B_{window} (burned DRT in the same window)

Net supply change:

- $\Delta S_{DRT} = M_{window} - B_{window}$

Circulating supply evolution:

- $S_{DRT}(t+1) = S_{DRT}(t) + \Delta S_{DRT}$



This makes the system's monetary stance explicit: emissions fund security and operations; fee burn applies deterministic deflationary pressure as usage increases.

10.9 TREASURY FLOW (HIGH LEVEL)

Treasury intake per event:

- $T_{in}(t) = R_{treasury}(t)$

Treasury outflows are authorized via typed governance proposals and timelocks (see Section 12).

12. Governance and Parameter Control

This section explains how Dytallix evolves over time while maintaining its security assumptions. Governance is limited to adjusting a defined set of parameters within enforced limits, without bypassing consensus rules or executing arbitrary commands against the base protocol.

12.1 SEPARATION OF GOVERNANCE AND UTILITY

- **DRT** provides no governance power.
- Governance is exercised via **DGT**.

This separation prevents transaction demand from translating into political control.

12.2 GOVERNANCE FLOW (PROPOSAL → VOTE → EXECUTE)

Governance follows an on-chain pipeline:

1. **Typed proposal:** the action type and parameters are declared up front.
2. **Vote:** DGT-weighted voting during a fixed window.
3. **Thresholds:** quorum and approval thresholds must be met.
4. **Timelock:** a mandatory delay before execution.
5. **Execute:** only the governance executor may call privileged update functions.

12.3 GOVERNANCE BOUNDARIES

- **Mutable (examples):** bounded fee-market coefficients, bounded emission-controller parameters, treasury spend authorizations, and cryptographic algorithm lifecycle state (Active/Deprecated/Revoked) via the algorithm registry.
- **Immutable:** core consensus validity rules, signature verification requirements, or any behavior that is not exposed through explicitly defined update functions.

12.4 ENFORCEMENT MECHANISMS

Governance authority is enforced mechanically by contract boundaries:

- The **emission controller** is the only entity authorized to mint DRT.
- The **governance executor** is the only entity authorized to update tunables.
- Proposals are **schema-validated** on-chain; invalid proposals are non-executable even if they receive votes.

12.5 MATHEMATICAL AND ECONOMIC SAFETY CONSTRAINTS

Sensitive parameters are guarded by:

- **Hard bounds** (e.g., $E_{min} \leq E(t) \leq E_{max}$)
- Max step-size limits per governance event
- **Cooldowns** that limit update frequency

These constraints reduce parameter-shock risk and create predictable governance behavior.

12.6 TREASURY CONTROL

Treasury outflows are authorized only via typed proposals and executed through the timelock.

All actions are on-chain, enumerable, and attributable to a specific vote.

13. Operational Posture

- PQC-first defaults: no classical-key transaction path.
- Upgrade discipline: algorithm and parameter upgrades occur through explicit lifecycle controls.
- Audit posture: third-party audits focus on consensus-core and WASM precompiles.

14. Specification Completion Checklist

This section makes the foundational paper operationally complete by specifying concrete formats and pseudocode.

These are normative unless marked as optional.

14.1 FORK-CHOICE PSEUDOCODE (LMD-GHOST)

Inputs:
- Genesis block G
- Block tree with parent links
- LatestVote[v] = (block_hash, weight) for each validator v (only latest message counts)

LMD-GHOST Fork Choice Rule

```
Function WeightSubtree(b)
   Input: A block b
   Output: The cumulative weight of supporting votes for subtree(b)

   return sum_{v where LatestVote[v].block_hash is in subtree(b)} LatestVote[v].weight


Function LMD_GHOST(head)
   Input: The current known head of the chain
   Output: The preferred tip of the chain

   b := head

   while b has children do
      choose child c of b such that WeightSubtree(c) is maximized
      b := c
   end while

   return b
```

Rule:
- Start from the last finalized checkpoint (or genesis if none)
- Compute head = LMD_GHOST(finalized_checkpoint_block)
- Build/extend on head

13.4 COMMITTEE AND ATTESTATION FORMATS

14.2.1 BLOCK VOTE MESSAGE

```
BlockVote {
  uint64   slot;
  bytes32  head_block_hash;      // vote target for fork choice
  bytes32  justified_checkpoint;  // latest justified checkpoint hash
  bytes32  finalized_checkpoint;  // latest finalized checkpoint hash
  uint32   validator_index;
  bytes    mldsa65_signature;     // signature over domain-separated hash of the above
}
```

14.2.2 CHECKPOINT ATTESTATION MESSAGE

```
CheckpointAttestation {
  uint64   epoch;
  bytes32  checkpoint_block_hash;
  bytes32  state_root;
  uint32   validator_index;
  bytes    mldsa65_signature;     // signature over domain-separated hash
}
```

14.2.3 CHECKPOINT BLOCK AGGREGATION (BITFIELD-COMPRESSED MERKLE PROOF)

```
CheckpointAggregate {
  uint64   epoch;
  bytes32  checkpoint_block_hash;
  bytes32  state_root;
  bitset   signer_bitfield;       // length = committee_size
  bytes32  signatures_merkle_root;
  // Off-chain data retrievable via p2p or archival services:
  //   - Merkle proof for any included signature leaf
  //   - Signature leaves (validator_index, signature)
}

SignatureLeaf {
  uint32  validator_index;
  bytes   mldsa65_signature;
}

LeafHash = BLAKE3(validator_index || signature)
Root     = MerkleRoot(LeafHash[] for signers)
```

14.3 VRF CONSTRUCTION REFERENCE AND ASSUMPTIONS

The consensus requires a VRF-like primitive for private sortition with public verifiability.

Baseline construction (deterministic signature derived):

```
VRF_sk(m):
  π = ML-DSA-65.Sign_det(sk, DomainSep(m))
  y = BLAKE3(π)
  return (y, π)

Verify(pk, m, y, π):
  require ML-DSA-65.Verify(pk, DomainSep(m), π) == true
  require BLAKE3(π) == y
  return true
```

Assumptions:

- ML-DSA-65 is EUF-CMA secure under Module-LWE/Module-SIS assumptions.
- Deterministic signing prevents nonce-based key leakage.

- Hashing the proof yields pseudorandom output under a standard random-oracle-style modeling of the hash.

If a future standardized PQC VRF emerges, it can be swapped in under the crypto-agility framework.

14.4 SLASHING PROOF FORMATS

14.4.1 DOUBLE-VOTE (EQUIVOCATION) PROOF

```
DoubleVoteProof {
  BlockVote vote_a;
  BlockVote vote_b;
}

Validity:
  - vote_a.validator_index == vote_b.validator_index
  - vote_a.slot == vote_b.slot
  - vote_a.head_block_hash != vote_b.head_block_hash
  - both signatures verify

Penalty:
  - slash 100% of delegated stake for the validator
```

14.4.2 LIVENESS FAULT EVIDENCE

Liveness penalties are computed from on-chain participation statistics.

```
LivenessEvidence {
  uint64 epoch;
  uint32 validator_index;
  uint32 missed_slots;      // or missed_attestations
  uint32 total_slots;
}

Validity:
  - derived from canonical chain data
  - missed_slots / total_slots > 0.20

Penalty:
  - griefing slash (0.1%)
```

14.5 FEE PARAMETERIZATION AND ON-CHAIN TUNABLES

Dimensional fee market parameters (governance-controlled):

- P_c (compute unit price)
- P_{base} (bandwidth base price)
- α (bandwidth elasticity coefficient)
- U_{target} (target utilization)

Block limits (max bandwidth units / target utilization definition)

Emission controller parameters (governance-controlled):**

- E_{base}, E_{min}, E_{max}
- PID gains: K_p, K_i, K_d
- Integral window: W
- Emission split: 40/30/30 constants (or governance-tunable if explicitly permitted)

Protocol timing parameters (governance-controlled, but high-friction):

- Slot duration (default 2s)
- Epoch length (default 100 slots)

14.6 LIGHT CLIENT PROOF FORMAT (QUANTUM-COMPATIBLE)

Light clients must verify chain finality and selected state elements without executing full blocks.

14.6.1 HEADER SYNC

```
LightHeader {
  bytes32  parent_hash;
  bytes32  block_hash;          // final block hash (SHAKE256-based)
  uint64   slot;
  bytes32  state_root;          // SMT root (BLAKE3-based merkleization)
  bytes32  tx_root;
  bytes    proposer_sig;        // ML-DSA-65
  bytes    vrf_proof;           // per VRF construction
}
```

Light clients accept a header chain only when it is anchored by a finalized checkpoint.

14.6.2 FINALITY PROOF (CHECKPOINT)

```
FinalityProof {
  CheckpointAggregate checkpoint;
  // plus enough signature leaves and merkle branches to prove
  // stake-weighted supermajority participation.
  SignatureLeaf[] leaves;
  bytes32[][]     branches;
}
```

Verification steps:

- Verify each leaf signature.
- Verify each leaf hashes into `signatures_merkle_root`
- Use `signer_bitfield` + validator weights to confirm $> \frac{2}{3}$ stake.
- Accept the checkpoint as finalized.

14.6.3 STATE PROOF (SMT)

```
SMTProof {
  bytes32 key;            // 32-byte key (e.g., hashed D-Addr)
  bytes   value;          // encoded account/contract value
  bytes32[] side_nodes;   // sparse merkle proof nodes
}
```

The light client verifies inclusion against `state_root` using BLAKE3 and the SMT rules.

14.7 BENCHMARK METHODOLOGY AND REPRODUCIBLE CONFIGS

To make performance claims reproducible, benchmarks must specify:

- **Hardware:** CPU model, cores/threads, RAM, storage type.
- **Software:** OS, compiler versions, build flags (e.g., AVX2 enabled), crate versions.
- **Workload:** signature mix (ML-DSA vs SLH-DSA), batch sizes, block sizes, utilization level.
- **Metrics:** verification throughput, end-to-end block validation time, bandwidth per block, p95/p99 latency.

Reproducible benchmark template:

```
bench_name: mldsa65_batch_verify
cpu: <model>
ram_gb: <n>
os: <distro + kernel>
compiler: rustc <ver>
flags: -C target-cpu=native
pqc_core: <git sha/tag>
batch_sizes: [64, 256, 1024]
message_size_bytes: 32
sig_scheme: ML-DSA-65
runs: 30
metrics: [sign_ops_per_s, verify_ops_per_s, p95_verify_ms]
```

15. Business Case and Use Cases

Dytallix is not built for the speculative economy of next quarter.

It is built for the durability economy of the next century: industries where data longevity exceeds the timeline to quantum viability (the Y2Q horizon).

For these domains, remaining on legacy cryptography is not a risk, it is a guarantee of future failure.

15.1 WHY THIS MATTERS ECONOMICALLY

- **Hard deadline:** “harvest now, decrypt later” creates present-day exposure for long-horizon confidentiality.

- **Existential integrity risk:** once classical signatures can be forged, ownership and authorization fail.
- **Institutional migration risk:** retrofitting legacy systems under crisis conditions creates operational loss events.

Dytallix's value proposition is simple: assets and systems issued on a PQC-native ledger are "born secure" and do not require a rushed cryptographic migration later.

15.2 ILLUSTRATIVE USE CASES

Pharmaceuticals

- **Problem:** drug development cycles span 10–15 years, followed by decades of patent and commercialization value.
- **Failure mode:** proprietary formulas encrypted under RSA/ECC can be captured today and decrypted later.
- **Dytallix solution:** PQC-secure custody and immutable audit trails for integrity, provenance, and priority proofs over long horizons.

Critical Infrastructure: Water, Power, Transport

- **Problem:** firmware updates and command channels rely on digital signatures.
- **Failure mode:** quantum-enabled signature forgeries allow command injection, malicious updates, and safety failures.
- **Dytallix solution:** PQC-native device identity and command authorization. Updates and control messages are authorized under ML-DSA, preserving integrity even under CRQC conditions.

Sovereign Wealth and Pension Funds

- **Problem:** funds operate on 30–50 year horizons; they hold assets whose integrity must outlast the quantum transition.
- **Failure mode:** emergency migrations during a cryptographic crisis cause custody failures and irreversible loss.
- **Dytallix solution:** tokenized assets on Dytallix remain cryptographically valid across decades because the ledger is PQC-native from inception.

Institutional Product Surfaces

Dytallix separates the base protocol from institutional integration surfaces:

- **Base protocol (permissionless):** PQC-native settlement for transactions, execution, and finality proofs.
- **QuantumVault (optional):** custody + issuance controls for high-assurance workflows and Airlock Migration.
- **Assurance modules (optional):** application-layer tooling that supports institutional operations without changing the permissionless consensus boundary (see Section 17).

16. Regulatory Positioning and Compliance Strategy

Navigating global regulation is a strategic pillar. Dytallix is designed so that institutional adoption does not require permissioning the base layer.

Compliance attaches at custody, issuance, and application boundaries, not at the consensus boundary.

16.1 ASSET CLASSIFICATION

Dytallix establishes a clear delineation of utility:

- **Dytallix Governance Token (DGT):** governance/staking-weight token. It grants voting rights and staking capability but carries no claim on revenue, dividends, or ownership of any entity.
- **Dytallix Reward Token (DRT):** a consumable resource (gas) and operational reward. Its elastic supply and fee-burn mechanism support its function as a utility commodity powering network execution and security.

16.2 INSTITUTIONAL COMPLIANCE POSTURE

For enterprise partners, regulatory compliance is binary: pass or fail.

Dytallix's institutional posture is built around:

- **Standards alignment:** protocol defaults use NIST-standardized PQC primitives (FIPS 203/204/205).
- **Auditability:** open-source code, reproducible builds, and third-party audits for consensus-core and PQC precompiles.
- **Compliance at the edge:** regulated workflows can be implemented using permissioned pools/allowlisted contracts, verifiable credentials, and custody controls—without changing base-layer permissionless validation.

16.3 PRACTICAL COMPLIANCE PRIMITIVES

- **Verifiable Credentials (VCs):** vapplication-layer identity claims for regulated participation.
- **Permissioned pools/allowlisted contracts:** contract-enforced compliance constraints for specific markets.
- **Travel-rule compatible metadata:** optional structured fields for regulated transfers.

17. Roadmap

The Dytallix roadmap is a phased execution plan moving from controlled development to a decentralized mainnet.

Each phase has technical milestones, operational KPIs, and ecosystem targets.

PHASE 1: Q1 2026 TO Q3 2026

Focus: core infrastructure, devnet stability, and cryptographic primitive validation. This phase prioritizes correctness over throughput and establishes the Zero Legacy baseline.

Technical Milestones

- ML-DSA-65, ML-KEM-768, BLAKE3 integration.
- PQC-Noise transport validation and DoS hardening.
- Devnet packaging: dockerized nodes, faucet, explorer.

Operational KPIs

- **Validator count:** 10–50 (internal + core partners).
- **Stability:** 99% uptime for 7 consecutive days.

PHASE 2: Q4 2026 TO Q1 2027

Focus: public validator onboarding, tokenomics stress testing, and governance bootstrapping (incentivized testnet).

Technical Milestones

- Dimensional gas activation (C-Gas vs B-Gas) and bandwidth-price tuning.
- QuantumVault MVP and first Airlock Migration issuance flows.
- WASM contract engine hardening; audit of PQC precompiles.

Product Rollouts (Enterprise Layer)

- **Consul (Governance Diplomat):** testing release target Q3 2026.
- **Aegis / Garrison:** continued testnet operation in preparation for mainnet availability.

Ecosystem & Growth

- Incentivized validator program.
- Bug bounty targeting cryptography, consensus, and networking.

Operational KPIs

- Validator count: 500+ distributed nodes.
- Sustained throughput targets under PQC load testing.

PHASE 3: Q2 2027 TO Q4 2027

Focus: Genesis, token generation, and institutional on-ramp.

Technical Milestones

- Genesis distribution procedures and emission controller activation.
- QuantumVault production release and formal operational controls.
- Governance activation and first parameter ratification.

Product Rollouts (Enterprise Layer)

- **Aegis (Active Defense):** mainnet availability at/after genesis.
- **Garrison (Compliance Assurance):** mainnet availability at/after genesis.
- **Horizon (Threat Monitoring):** target rollout Q4 2026.
- **Vector (Predictive Risk):** target rollout Q1 2027.

Ecosystem & Growth

- Exchange listings for DGT/DRT (jurisdiction-dependent).

- First enterprise pilot partners (e.g., pharma/energy/finance).

Operational KPIs

- **Validator count:** 1,000+.
- **Target decentralization:** Nakamoto coefficient > 30.
- **Finality UX target:** <4s soft confirmation under normal network conditions; epoch checkpoint finality as defined in Section 8.

PHASE 4: Q1 2028 TO 2029

Focus: Scalability and L2 ecosystem.

Technical Milestones

- L2 rollups settling to Dytallix (quantum-compatible proof systems).
- Vote and signature compression improvements.
- Archival incentives and long-term state sustainability controls.

Operational KPIs

- **Validator count:** 3,000+ globally.
- Sustained utilization without destabilizing fees.

PHASE 5: 2030 AND BEYOND

Focus: Y2Q readiness and long-horizon hardening.

Technical Milestones

- Algorithm registry review cycles and parameter upgrades as needed.
- Sunset pathways for legacy-facing interfaces where feasible.
- Governance posture hardened for stability and conservative change.

18. Products and Services

Dytallix separates the permissionless base protocol from optional enterprise products. These products may generate attestations or tooling for regulated operators, but they do not alter the protocol's consensus rules or permissionless validation.

18.1 QUANTUMVAULT (CUSTODY + AIRLOCK MIGRATION)

QuantumVault is the institutional custody and control plane for organizations migrating value into a PQC-native environment. It provides the protocol's optional Airlock Migration interface and operational controls required by enterprise custody workflows (auditability, policy enforcement, and controlled issuance/redemption).

18.2 AI-ENHANCED ASSURANCE MODULES

These modules address security, compliance, and governance assurance needs for institutional participants. Where useful, they can publish PQC-signed attestations that are verifiable on-chain.

Aegis (Active Defense)

- **Function**: real-time assessment of transaction and wallet vulnerability.
- **Mechanism:** produces PQC-signed risk attestations (e.g., ML-DSA Level 5 profiles where required) based on wallet age, interaction history, and cryptographic posture.
- **Status:** active on testnet; planned for mainnet.

Garrison (Compliance Assurance)

- **Function:** smart contract assurance combining automated compliance checks with safety inspection.
- **Mechanism:** maps requirements to code via AST/CFG analysis and applies formal verification workflows where feasible.
- **Status:** active on testnet; planned for mainnet.

Consul (Governance)

- **Function:** governance proposal impact simulation before voting.
- **Mechanism:** agent-based modeling and game-theoretic analysis to flag governance attacks and unintended treasury drains.
- **Status/ETA:** testing; Q3 2026.

Horizon (Threat Monitoring)

- **Function:** unified monitoring for network traffic, liquidity pools, and migration/airlock flows.
- **Mechanism:** anomaly detection using graph/time-series methods and flow analysis for exploit detection.
- **Status/ETA:** in development; Q4 2026.

Vector (Predictive Risk)

- Function: predictive fraud and financial-threat advisor for compliance teams.
- Mechanism: behavioral risk scoring with privacy-preserving intelligence sharing where applicable.
- Status/ETA: in development; Q1 2027.

19. Funding Strategy

Dytallix operates with a disciplined capital strategy that prioritizes sovereign sustainability and product-led revenue over speculative dilution.

19.1 SELF-FUNDING AND FINANCIAL INDEPENDENCE

Dytallix is currently self-funded through founder contributions.

This is intentional:

- **Operational autonomy:** removes short-term capital pressures and allows prioritizing cryptographic correctness and long-horizon security over rapid expansion.

- **Runway durability:** development is funded to reach mainnet maturity without being forced into premature compromises.

19.2 EXTERNAL CAPITAL POLICY

Private/VC funding is considered only when operational scaling, market expansion, or specific business requirements justify outside capital.

Capital is treated as an instrument, not a dependency.

19.3 PRODUCT-LED REVENUE AND REINVESTMENT

- **Base protocol economics**: defined in Section 10 (DRT emissions fund security and operations; fees are burned).
- **Product revenue:** enterprise products (QuantumVault and AI modules; Section 17) are intended to generate non-speculative revenue.

Net proceeds are intended to be reinvested into development, security, audits, and long-term maintenance through treasury processes and governance-controlled budgeting.

20. Risk and Security Posture

- **PQC implementation risk**: mitigated via open-source auditability and standardized primitives.
- **Side-channel risk**: mitigated with constant-time routines and masking in ML-KEM routines.
- **Governance capture risk**: mitigated by fixed-supply governance token and explicit cost-of-corruption framing.
- **Bandwidth exhaustion risk:** mitigated by exponential bandwidth pricing and handshake DoS defenses.

21. Call for Contributors

Dytallix is not a theoretical system. It is running code. The transition to a quantum-secure web requires more that just code and protocol. It requires people: auditors, cryptographers engineers willing to test the limits of this new stack.

We invite the technical community to move from observation to participation.

- **Get the SDK:** Install the @dytallix/sdk via Cargo to start generating ML-DSA signatures and building PQC-native dApps today.
- **Run a Node:** Join the Devnet by pulling the Docker container. Help us stress-test the PQC noise handshake under real-world network partitions.
- **Audit the Math:** Our cryptographic implementations are open-source. We challenge cryptographers to review our adaptation of Dilithium and Kyber for the blockchain context.

Visit www.dytallix.com to access the developer documentation, join the technical Telegram, and begin building the infrastructure that will survive the next era of computing.

22. Legal Disclaimer

This whitepaper is for information purposes only. Dytallix (DGT/DRT) is not an offer of securities or a solicitation for investment. The Dytallix protocol is experimental software utilizing novel cryptographic primitives. Participants acknowledge the risk of code failure, regulatory uncertainty, and total loss of funds. The roadmap represents forward-looking statements that may change based on technical feasibility and community governance.

23. Glossary

This glossary defines terms used in this paper that are either Dytallix-specific or commonly overloaded.

- Airlock Migration: Custodial deposit-and-reissue process for bringing legacy-chain value into Dytallix under a separately disclosed trust model (not a trustless bridge).
-
- **Algorithm Registry**: On-chain registry of cryptographic algorithms and their lifecycle state (Active / Deprecated / Revoked) used for crypto-agility.
- **BLAKE3:** Fast hash used for state hashing, content addressing, and Merkleization where performance is critical.
- **Bitfield-Compressed Merkle Proof:** Checkpoint vote compression method using a signer bitfield plus a Merkle root over signature leaves.
- **Checkpoint:** An epoch boundary block used as an anchor for justification/finalization.
- **C-Gas / B-Gas**: Dimensional fee components: compute units (C-Gas) and bandwidth bytes (B-Gas).
- **Dimensional fee market**: Fee model pricing compute and bandwidth separately:
  $F_{tx} = C_{gas} \cdot P_c + B_{gas} \cdot P_b(U)$
- **D-Addr:** Dytallix address format: Bech32m(BLAKE3(PubKey)), hiding the public key until first spend.
- **DGT / DRT:** Dual-token model: DGT is fixed-supply governance/staking weight; DRT is elastic reward/utility used for fees (burned) and rewards.
- **Justified / Finalized:** Checkpoint states. “Justified” means supermajority support; “Finalized” means the checkpoint is irreversibly anchored absent slashable faults.
- L**MD-GHOST:** Fork-choice rule selecting the subtree with the most stake-weighted latest votes.
- **ML-DSA-65:** NIST PQC signature used for transactions and validator voting.
- **ML-KEM-768**: NIST PQC key establishment (KEM) used for node handshakes and transport security.
- **Noise_IK_PQC**: Noise IK-style handshake adapted by replacing Diffie–Hellman with KEM + authentication.
- **PID emission controller**: Controller that adjusts DRT issuance based on utilization error: proportional + integral + derivative terms with bounds.
- **SHAKE256:** NIST XOF used in PQC primitive internals and for final block hashing where strict compliance is required.
- **SLH-DSA**: NIST hash-based signature (optional) intended for long-horizon / cold-storage use.
- **SMT (Sparse Merkle Tree)**: Commitment structure enabling compact state inclusion proofs against a state root.

- **Utilization (U)**: Block fullness metric driving bandwidth pricing $P_b(U)$ and emission control targets.
- **VRF (Verifiable Random Function):** Sortition primitive producing unpredictable output with a publicly verifiable proof; used for leader election.
- **Zero legacy**: Protocol stance excluding classical ECC/RSA authorization paths from the security boundary.

24. References

- NIST FIPS 204: Module-Lattice-Based Digital Signature Standard (ML-DSA). National Institute of Standards and Technology, 2024.
- NIST FIPS 203: Module-Lattice-Based Key-Encapsulation Mechanism Standard (ML-KEM). National Institute of Standards and Technology, 2024.
- Nakamoto, S.: Bitcoin: A Peer-to-Peer Electronic Cash System. 2008.
- Lyubashevsky, V., et al.: CRYSTALS-Dilithium: Algorithm Specifications and Supporting Documentation. 2020.
- Avanzi, R., et al.: CRYSTALS-Kyber: Algorithm Specifications and Supporting Documentation. 2020.
- Bernstein, D. J., et al.: SPHINCS+: Submission to the NIST Post-Quantum Cryptography Project. 2020.
- Dytallix Team: Additional Documents