MONOLYTHIUM

Lightpaper

V5.0 · MAY 2026

Settlement Layer for the Autonomous Economy

NETWORK	Monolythium · LYTH
STEWARDED BY	Mono Labs R&D LLC · Monolythium Foundation
LICENSE	Whitepaper text — CC BY-SA 4.0

TL;DR

Monolythium is a Layer 1 blockchain designed as the **settlement layer for the autonomous economy** — a future in which humans, companies, AI agents, machines, and software services transact directly on open rails.

The chain is **not EVM-compatible at execution**. It is **EVM-connected at the liquidity edge**. And it is designed to compose **underneath** every major agent-payment standard — x402, AP2, ACP, MCP, Visa Intelligent Commerce, Mastercard Agent Pay — as the chain-anchored trust, policy, escrow, identity, and reputation layer those rails leave open.

Six positions define the design:

1. **Post-quantum accounts by default**, with no classical signature acceptance path.
2. **Rust-first smart contracts** compiled to a deterministic RISC-V execution target.
3. **Native modules** for tokens, NFTs, markets, bridges, payments, and agent commerce.
4. **Bifurcated denomination** that separates monetary privacy from commerce so the chain cannot be used as a fungible-anonymous-payment rail.
5. **Cluster marketplace** that turns validator operation into a public, competitive market with distributed validator technology.
6. **No on-chain governance and no perpetual futures**, so the surface that can be captured, gamed, or weaponised is smaller.

The chain is useful even if the agent economy grows slowly. Native tokens, spot markets, post-quantum identity, verifiable bridges, and the bifurcated denomination are valuable on their own. The bet is asymmetric: a strong fit for the autonomous-economy case, a strong general-purpose settlement chain in the meantime.

1. The Thesis

Most production blockchains today are honestly named **trading chains**. They optimize for swap latency, perpetuals throughput, and high-frequency fee surfaces. Trading is fine; trading is not what the next ten years of digital economic activity will primarily look like.

The next ten years are **agentic**. An AI agent that pays for inference, hires a contractor, buys data, settles an invoice, escrows funds against a deliverable, subscribes to compute, or pays another agent for a service is not science fiction — each of those actions is technically possible today. What is missing is open, neutral, enforceable rails on which to settle them.

Today, an agent that wants to spend money does so through an account it shares with its user inside a closed AI platform, through a payment-processor account leased by the user, or through a custodial wallet rented from a single exchange. Each rail is **owned**. The agent is rentable. The identity is rentable. The reputation accrued through years of work is rentable. If the platform changes its terms, the agent's economic life resets.

Monolythium exists to provide a different option — programmable, auditable, permissionless, bridge-connected, post-quantum, and structurally hostile to use cases that combine anonymous payment with anonymous service discovery.

The audience the chain is built for, in plain terms: **anyone who needs a delegated software entity to act with bounded authority across providers and jurisdictions on rails the principal controls.** That includes AI agents — but also fintech automations, organizational treasury policies, vendor pipelines, and routine machine-to-machine commerce. The chain is the same chain it would be if no agent ever showed up. The user is different.

> The moment an AI assistant can say, "Send me 50 USDC to complete this task," the agent-commerce era has started. The chain provides the rails on which that conversation settles.

---

2. The First Wedge

A settlement layer for the autonomous economy is the long horizon. The first commercial wedge is narrower.

The wedge is agent escrow, spending policy, and reputation as the trust layer behind paid APIs and services.

The market is already converging on standardised payment handshakes for agents. x402 (Coinbase) shipped in 2025; AP2 (Google) launched the same year; Stripe and OpenAI released the Agentic Commerce Protocol; AWS Bedrock AgentCore added native stable-coin payment rails; Visa and Mastercard both announced agent-payment programs. These standards solve the payment **handshake** — how an agent discovers a paid end-point, presents payment, and receives a response.

They do not solve four problems that are unavoidable once agents operate against more than trivial workflows:

- **Enforceable principal-level spending policy** — caps, allow-lists, time-of-day rules, expiry, revocation — that the agent itself cannot bypass.
- **Escrow with counter-offer flow and pluggable arbiters** for deliverable-against-payment work.
- **Portable cross-platform reputation tied to the agent identity** that survives a switch of model provider, runtime, or wallet.
- **Chain-anchored consent records** the principal can revoke instantly, leaving in-flight commitments grandfathered.

Monolythium ships those four as native modules. They are the chain's first deliverable to the existing rails — not as competition, but as composition.

3. Composition with Agent-Payment Standards

Monolythium is designed to compose **underneath** the open standards now shipping for AI-agent payments. Each standard solves a different layer of the stack; Monolythium provides the layer they all share.

LAYER	STANDARD	MONOLYTHIUM ROLE
HTTP payment handshake	**x402** (Coinbase, x402 Foundation)	Receives x402-initiated stablecoin settlement; provides on-chain receipt, escrow, and reputation against the payee
Agent intent mandate	**AP2** (Google, open)	Anchors the agent's mandate as on-chain consent + spending-policy registration on the agent's sub-account
Checkout flow	**ACP** (Stripe, OpenAI)	Settles ACP checkout outputs; provides escrow with counter-offer flow and dispute resolution for the seller
Tool / connector boundary	**MCP** (Model Context Protocol)	Exposes spending policy, runbooks, escrow, and discovery as MCP tools the assistant can invoke
Card-issuer agent flows	**Visa Intelligent Commerce, Mastercard Agent Pay**	Provides the chain-anchored settlement, identity, and reputation layer behind agent-initiated card payments

The composition is **one-directional**. Monolythium does not require the rails to adopt anything; the rails do their job (initiate payment, transmit intent, complete checkout) and Monolythium handles the parts they leave open (policy enforcement, escrow, dispute, reputation, post-quantum settlement).

Worked example — x402 + Monolythium

1. An assistant requests a paid API. The endpoint returns `HTTP 402` with x402 payment details.
2. The wallet checks the Monolythium spending policy on the agent's sub-account: category allow-listed, price under per-call cap, monthly aggregate under cap.
3. The wallet signs a Monolythium transaction that pays the x402 invoice in stablecoin; `purpose` references the x402 payment ID; the runbook is `pay_vendor`.
4. The protocol verifies the spending-policy constraints **at admission** and refuses if any fail.
5. Settlement happens at anchor finality (three to five seconds). The endpoint serves the response.

6. If the response is disputed, the principal opens an escrow-arbiter dispute referencing the x402 transaction.

The endpoint integrates only with x402. The chain handles everything beyond the handshake. The agent does not implement any of this manually — the wallet, the runbook, and the spending policy carry the discipline.

---

4. What Monolythium Refuses

A chain's identity is shaped at least as much by what it refuses to do as by what it does. Five refusals define Monolythium.

4.1 No on-chain governance

No governance token. No on-chain proposal contract. No protocol-level voting mechanism. No on-chain treasury voting. No signaling extrinsic that pretends to be governance.

A decade of L1 experiments has produced four consistent governance failure modes — capture by large holders, vote-buying markets, low-turnout theatre, and a direct control surface for attackers who obtain a quorum. The chain rejects all four by **not having the mechanism**. Direction is set by accountable maintainers in public; operators choose whether to run new node software; the legitimate path of dissent is forking.

4.2 No perpetual futures or margin

Spot markets, yes. Perpetual futures, on-chain leverage, and margin trading, no — not on mainnet, not as a mintable asset class, not as a planned addition. Once a chain becomes a perpetuals venue, the design pressure on every other surface bends toward the perpetuals product. The chain pays the simplification — smaller audit surface, cleaner identity — to keep the design focused on settlement.

4.3 No EVM execution

Monolythium is **not EVM-compatible at execution**. EVM bytecode does not run on the chain; Solidity is not the default developer model; ERC conventions are not the protocol standards.

The chain is **EVM-connected at the liquidity edge** — value moves in and out through light-client and zero-knowledge bridges, cross-chain swaps, and issuer-supported integrations. Once value arrives, it settles through Mono-native standards on a Rust/RISC-V execution layer.

4.4 No fungibility between public and private money

LYTH has two denominations, **public** and **private**, and they are **not fungible**. The crossing from public to private is one-way; private LYTH cannot ever return to public. Private LYTH supports two operations only: transfer to another private address, and burn. It cannot enter a smart contract, cannot trade on the spot order book, cannot bridge, and cannot pay for any service mediated by the discovery registry. See §9.

4.5 No bundled AI model

The chain does not bundle a specific AI model with the protocol. Agent identity is chain-native; the model behind any agent is the user's choice. Runbooks are typed and signed; they are not "AI prompts." The chain is model-agnostic on principle so that no model vendor's roadmap binds the protocol's neutrality.

5. Six Design Positions

The chain's identity is the product of the six commitments below. Each is structural, not parameterised.

5.1 Post-quantum accounts by default

ML-DSA-65 (NIST FIPS 204, Dilithium Level 3) is the only signature primitive accepted at transaction admission. No classical fallback. No hybrid mode. No ECDSA acceptance path "for compatibility." Every account signs every transaction with a post-quantum primitive from the chain's first block.

Hybrid is rejected for three reasons: a hybrid signature is only as strong as its weakest component, hybrid creates audit ambiguity and downgrade-attack surface, and hybrid postpones the migration twice (once now, once later) instead of doing it once at genesis when there is no installed base of classical-only wallets to support.

The cost of pure post-quantum is signature size — ML-DSA-65 signatures are ~3,309 bytes versus ECDSA's 64 bytes. The cost is real and paid in storage and bandwidth. The alternative is paying the migration cost twice.

LAYER	PRIMITIVE	USE
User signatures	**ML-DSA-65** (FIPS 204)	Every user-signed transaction
Emergency backup signatures	**SLH-DSA** (FIPS 205, hash-based)	Pre-registered backup; activated under emergency rotation
Key encapsulation	**ML-KEM-768** (FIPS 203)	Peer-to-peer Noise handshakes, RPC TLS, stealth-address derivation, mempool encapsulation
Aggregate signatures	**BLS12-381**	Per-cluster threshold aggregation, VRF, distributed key generation
Threshold key encapsulation	**Ferveo over BLS12-381**	Encrypted-mempool body decryption at anchor inclusion
Zero-knowledge verification	**SP1 zkVM + Groth16-BN254**	zkML attestations, high-value off-chain computation
Hash	**BLAKE3**	State-tree leaves, Merkle commitments, address derivation

5.2 Two-tier finality

The chain delivers two finality signals tuned for different use cases:

- **Anchor-level finality** (three to five seconds, BLS12-381 aggregate). Per-cluster threshold signatures compress 700 individual operator signatures per round into a single verifier check. Used for everyday transfers, application interactions, and mempool admission.
- **Quantum-attested finality** (every ~100 anchors, ML-DSA-65 checkpoint). Operators each sign the canonical state root with their post-quantum keys. Bridges, exchange listings, and high-value cross-chain attestations bind here. A quantum attacker who can forge BLS aggregates cannot forge ML-DSA-65 — the fork dies at the next checkpoint.

Checkpoint cadence is milestone-overridable, so high-value bridge integrations can drop the interval (for example to ~24 seconds) for tighter cross-chain UX without a hard fork.

5.3 Rust on RISC-V

The execution layer has three tiers:

1. **Native protocol modules** for hot and security-critical primitives — token balances, NFTs, multi-assets, spot markets, delegation, bridge proof verification, agent-commerce registries, spending policy, name registry, privacy cordons.
2. **Rust/RISC-V contracts** for application-specific programmable logic.
3. **zkVM-proven computation** for bridge proofs, cross-chain swaps, zkML, and high-cost off-chain verification.

Why this combination:

- **Rust** provides strong compiler feedback, ergonomic testing and fuzzing, and better alignment with AI-assisted development than Solidity. The Rust ecosystem has fifteen years of training data; Solidity does not.
- **RISC-V** is open, simple, portable, deterministic, and aligned with the zero-knowledge proving ecosystem. The chain rides that convergence rather than fighting it.
- **Native modules** mean common asset and market logic is audited once at the protocol level. Applications compose against audited primitives rather than reimplementing them.
- **Cleaner audit surface** because the heaviest financial logic lives in the well-audited core, not in thousands of independent contracts.

5.4 Native modules and MRC standards

Instead of asking every application to rebuild core financial primitives as arbitrary smart contracts, the chain provides audited native modules and Mono-native standards.

ETHEREUM CONVENTION	MONOLYTHIUM TARGET
ERC-20	**MRC-20** fungible tokens
ERC-721	**MRC-721** non-fungible tokens
ERC-1155	**MRC-1155** multi-assets
ERC-4626	**MRC-4626** vaults
ERC-1271 / ERC-4337 patterns	Native smart-account and policy-account standards
Solidity marketplace contracts	Native marketplace modules and Rust/RISC-V contracts
Contract-based AMMs and order books	Native spot-market modules with contract hooks

Wallets and explorers display MRC assets as **first-class assets**, not as arbitrary contract calls. When a user receives an MRC-721, the wallet knows it is an NFT with a typed schema, not an opaque series of bytes that might or might not implement an interface correctly.

5.5 Cluster marketplace

Monolythium does not have validators in the traditional single-operator sense. It has **clusters**: 7-of-10 operator-threshold groups that produce one logical validator vertex per cluster, per consensus round. At target scale, **100 clusters × 10 operators = 1,000 operator positions** (700 active + 300 standby).

Cluster membership is a **public market**. Operators publish their node attestation (hardware class, network metadata, geographic claim, uptime, service-tier capacity), and clusters compose themselves from operators offering complementary skills — a bare-metal operator in one region for throughput, a cloud operator in a second for diversity, a home operator in a third for jurisdiction, a GPU-equipped operator for the prover service tier, an archive-capable operator for RPC.

Stakers see all of this through wallet and explorer surfaces and delegate accordingly. The “best validator” is not picked by social reputation; it is composed in public by people reading the same data the chain reads.

The cluster model gives the chain three structural advantages over single-operator designs:

- **Operator-level fault tolerance** — a single operator’s outage does not stop the cluster (7-of-10 threshold, three operator failures tolerated).

- **Operator diversity** — operators within a cluster can be in different regions, on different network providers, on different hardware classes, and under different jurisdictions.
- **Operator markets** — operator capacity is a public market; reputation accrues to the operator identity over time and is not owned by any single venue.

A hard per-operator multi-cluster cap prevents a successful operator from reproducing single-operator centralisation across the cluster set.

5.6 Bifurcated denomination

LYTH has two denominations and they are not fungible. The crossing from public to private is one-way at the protocol layer. Privacy machinery (stealth addresses, confidential transactions) lives **inside the private denomination only**. See §9 for the full position.

---

6. The Eight Agent-Commerce Primitives

Monolythium ships eight primitives that together enable the autonomous-economy use case. The primitives are minimal, composable, and consensus-critical only where consensus protection is necessary; gameable primitives are deliberately placed at the application or indexer layer.

#	PRIMITIVE	LAYER	PURPOSE
1	`attestation`	Native module	Foundational signed-hash + typed schema registry
2	`consent`	Native module	Principal-signed consent records with revocation
3	`issuer-registry`	Native module	Permissionless registry of credential issuers
4	`discovery`	Native module	Permissionless service-listing registry
5	`reputation`	Indexer view	Multi-dimensional rating aggregation over chain-emitted events
6	`availability`	Native module	Generic availability state composed by every domain registry
7	`escrow + arbiter-registry`	Native modules	Configurable escrow with counter-offer flow + pluggable arbiters
8	`spending-policy`	Native module (consensus-critical)	Programmable per-account spending caps enforced at admission

Only `spending-policy` consumes the consensus path. The protocol must enforce budget caps because the agent itself cannot be trusted to enforce its own caps — the agent is precisely what the policy is constraining. Reputation lives at the indexer layer because the gaming surface (Sybil rating, wash trades, coordinated reputation farming) does not belong in consensus-critical code.

Two primitives are worth calling out:

Escrow uses an `Open → Negotiating(round_n) → Accepted → InProgress → Submitted → Released | Disputed` state machine. Counter-offers can modify any term (timeline, scope, price, payment schedule, penalty clauses); each counter-offer is signed by both parties before becoming binding; either party can walk away during negotiation with no penalty. Arbiter mode is set per escrow at creation — single arbiter for low-value, quorum-of-N for mid-value, human-credentialed for high-value.

Spending policy is enforced **at admission, not after the fact**. A transaction signed by an agent sub-account that would violate any active policy is rejected before it enters the mempool. The agent cannot opt out of its own policy; the policy is the protocol.

Runbooks

Agents transact through **typed, versioned, signed templates** rather than free-text RPC. A runbook is a JSON-shaped operation definition with a signed parameter schema and a typed result. Initial runbooks include `pay_vendor`, `open_escrow`, `release_escrow`, `refund_payment`, `book_service`, `place_trade`, `set_spending_policy`, `revoke_agent_permission`, `verify_receipt`, and `rate_vendor`.

Runbooks are the bridge between natural language and safe settlement. The user speaks naturally; the agent maps intent into a constrained workflow; the wallet shows the exact action before approval. The protocol enforces the runbook's typed parameters; an agent cannot smuggle a hand-crafted RPC call past a wallet expecting a runbook.

---

7. Identity: Addresses, Mnemonics, and Names

Bech32m-only display

User-facing addresses are displayed in **bech32m** format only. Hex `0x...` display is not the canonical surface. The `mono` human-readable prefix and bech32m checksum make Monolythium addresses visually unmistakable from any other chain's `0x...` format, providing format-as-safety against cross-chain address confusion at exchange listings, bridge UIs, and partner integrations.

Per-type prefixes signal the address's role at a glance:

PREFIX	ADDRESS CLASS
`mono1...`	Standard user account
`monos1...`	Stake / delegation account
`monoc1...`	Contract account
`monok1...`	Cluster account
`monom1...`	Native module account
`monox1...`	Bridge account

PQM-1 mnemonic

Wallets support two recovery formats. The default is an encrypted keystore (Argon2id over the password, XChaCha20-Poly1305 over the seed). For users who want phrase-based backup, the chain defines **PQM-1**: a 24-word BIP-39 phrase whose bytes are interpreted under a post-quantum scheme (algorithm tag + version + 240-bit entropy, expanded via SHAKE256, used as the ML-DSA-65 keygen seed). A typical BIP-39 → secp256k1 EVM mnemonic is **not compatible** — wallets warn the user explicitly before either direction is attempted.

LYTH name registry

A hierarchical, on-chain name registry maps human-readable names to addresses. Lowercase ASCII letters, digits, and hyphens; no mixed case, no Unicode confusables. Names live under the `.mono` namespace and resolve to five structural categories: a bare `<name>.mono` is a human/personal account, an agent registers beneath its human principal as `<name>.agent.<human>.mono`, and clusters, contracts, and protocol entities take `<name>.cluster.mono`, `<name>.contract.mono`, and `<name>.system.mono` (the system category is foundation-only). U-curve pricing dis-

courages both squatting (short names cost more) and clutter (long names pay a small administrative fee). Transfer is propose-accept with a 24-hour acceptance window. Reserved prefixes prevent collisions with bech32m HRPs and `0x` strings.

A person is addressed as `alex-rivera.mono`, their agent as `support-bot.agent.alex-rivera.mono`, and a cluster as `northstar.cluster.mono`. The bech32m address remains the canonical underlying identifier; the name is a human-readable alias.

---

8. Bifurcated Denomination — Privacy Without Contamination

Privacy on a public ledger has, over the last several years, been on a steady regulatory collision course. The two historical trajectories are visible:

- A chain with **fungible** shielded and transparent denominations gets delisted by major exchanges because shielded-to-transparent flow contaminates the entire public denomination. Fungibility breaks the audit story for everyone.
- A chain with a **single private-by-default** denomination gets delisted because there is no public denomination to clear. The audit story is "we cannot tell you anything."

Both trajectories converge on the same outcome. Monolythium's bifurcation is a third option that has not been deployed at scale.

How bifurcation works

The protocol enforces non-fungibility at the consensus layer. Every account holds two balances — public and private — represented as separate state entries with separate spendable conditions. Every transaction operates on exactly one denomination. A native **caller-origin cordon** in the execution layer prevents a public contract or module path from receiving private-denominated value, and prevents private-denominated state from being used in any public contract context. The cordon is enforced by the host, not by user code, so no contract can opt out.

A user can move LYTH from public to private at any time. The protocol records the movement as a **crossing**. The reverse direction does not exist. There is no module, no instruction, no foundational mechanism that allows private LYTH to become public LYTH again.

What this gives a regulator or an exchange

The **public denomination** is fully auditable. Public activity is transparent and traceable by the analytics tooling that already exists for other transparent chains. A KYC-supervised exchange can custody public LYTH with the same audit posture as any other transparent asset.

The **private denomination** is opaque by design. The protocol does not pretend otherwise. An exchange may refuse private deposits, require a proof-of-crossing, or accept them at a higher KYC tier. The chain does not constrain the choice; the chain provides the structural separation that makes the choice coherent.

Side effect — structurally hostile to illicit commerce

The pattern that history calls “the darknet marketplace” requires the simultaneous availability of an anonymous-payment rail and an anonymous-service-discovery surface. Monolythium’s bifurcation makes those two requirements impossible to satisfy on the same chain. The private denomination has no service-discovery surface; the public denomination has full discovery but is fully transparent. A user can have anonymous money or service discovery, never both at once.

> Monolythium separates monetary privacy from commerce. The private denomination supports peer-to-peer transfers only — it cannot pay for services, interact with smart contracts, or trade. The public denomination supports full commerce and is fully transparent and analyzable.

9. Consensus — Starfish-C

The consensus engine is **Starfish-C**, a leaderless DAG-BFT protocol with:

- **Three-second deterministic finality** under partial synchrony (three to five seconds typical, ~eight seconds before view-change).
- **Deterministic linearization** of the DAG — two honest clusters starting from the same DAG state derive byte-identical block sequences.
- **Bounded reorg** capped by protocol parameter; an adversary cannot force a deeper reorg without controlling more than f Byzantine clusters.
- **Threshold-VRF leader selection** with set-independent Lagrange interpolation, so an adversary cannot influence the seed by selectively withholding signatures.
- **100% slash + permanent operator exile** on equivocation. The slash is exemplary, not merely punitive; a protocol that destroys the equivocating operator's stake and bars them forever does not need to tolerate equivocation.

The user-facing finality unit is the **anchor**. Multiple internal layers exist — vertex (cluster's signed round payload), wave (round of vertex production), anchor (deterministic linearization point), and block (preserved only for EVM-style RPC compatibility such as `eth_blockNumber`).

The encrypted-mempool admission rule is binding from genesis. Every transaction enters the mempool encrypted under a per-epoch Ferveo threshold-DKE public key; the body becomes plaintext only at anchor inclusion. No single operator can decrypt; no minority can; only a 7-of-10 cluster collusion could compromise mempool privacy, and even then only within the seconds-long lifecycle between admission and inclusion.

---

10. Tokenomics

10.1 Supply

- **Initial supply.** 100,000,000 LYTH at genesis.
- **Annual inflation cap.** 8% — a protocol-layer hard cap. Lifting it requires a coordinated hard fork.
- **Treasury funding.** Pre-funded from the genesis reserve; no inflation tap. Treasury draws on the reserve and on buyback-burn yield over time.

The supply structure is constitutional. The inflation cap, the treasury's funding mechanism, and the basic distribution rules are not changeable through any in-protocol process.

10.2 Allocation

#	CATEGORY	LYTH	SHARE
1	Community obligations (pre-allocated genesis balances)	32,781,503.90	32.78%
2	Foundation treasury reserve	15,000,000	15.00%
3	Ecosystem & grants	15,000,000	15.00%
4	Core contributors (vested)	13,000,000	13.00%
5	Public sale & community access programs	12,000,000	12.00%
6	Operator incentives & community programs	7,218,496.10	7.22%
7	Liquidity provision & integration support	5,000,000	5.00%
	Total	**100,000,000.00**	**100.00%**

Category 1 is a frozen ledger of 732 historical community-cohort entries pre-allocated as on-chain genesis balances. No claim flow is required; each entry resolves directly to a balance on the holder's chain address.

Category 4 vests on a 12-month cliff with 48-month linear vest.

10.3 Utility — LYTH at every layer

LYTH accrues utility across the entire chain surface, not only at the gas layer:

- **Consensus.** Each active operator position requires a 5,000 LYTH self-bond; standbys bond at the same level. Equivocation slashing burns the entire bond.

- **Network.** Every transaction pays gas in LYTH, including transactions moving stablecoins, MRC assets, or NFTs. A portion of every transaction's gas is burned.
- **Service tiers.** RPC calls, archival reads, GPU prover requests, and oracle feed consumption are denominated in LYTH and paid directly to the serving operator. The on-chain prover market is a native LYTH-denominated venue.
- **Bridges.** Each bridge route charges a fee in LYTH on top of the asset-denominated value; bridge insurance and reserve programs pay in LYTH.
- **Registries.** Discovery listings, issuer registrations, arbiter registrations, and name-registry entries all pay LYTH bonds or fees.
- **Markets.** Native order-book maker/taker fees pay in LYTH or in the asset traded (with a LYTH-denominated discount path).
- **Agent commerce.** Spending-policy registration, escrow opening, dispute escalation, and arbiter compensation all settle in LYTH.

Agents paying USDC for an API call still pay LYTH gas. Bridges moving Ethereum-resident value still pay LYTH route fees. Clusters serving traffic still earn LYTH. The token captures the chain's economic activity at every layer above the payment-asset itself.

10.4 Liquid bonding and distributed delegation

Monolythium's staking model is **liquid bonding**: stakers retain custody of their LYTH at all times. There is no staking contract to send funds to. There is **zero unbonding period** for delegators. A delegator can exit a delegation instantly. Slashing applies to operator self-bonds, not to delegators.

The core anti-capture mechanism is the **per-wallet delegation cap**: any single wallet can delegate at most X% of its total LYTH holdings to any single cluster. The cap binds on **capital**, not on wallet count, so an adversary splitting capital across many wallets gains no advantage.

NETWORK CONDITION	PER-CLUSTER CAP	MINIMUM DIVERSIFICATION
Early operating set	50%	2 clusters
Growing operating set	25%	4 clusters
Mature operating set	15%	7 clusters
Steady-state operating set	10%	10 clusters

Tightening is triggered by sustained network conditions (number of independent clusters, geographic distribution, operator entropy), not by calendar dates and not by any in-protocol vote. Modifying the schedule requires a coordinated hard fork.

The cap composes with a **quadratic reward curve** at the reward layer: doubling a cluster's stake does not double its reward share. Decentralization is the highest-yield strategy at every margin, not just at the boundary.

10.5 The four-button autovote

Delegation is enforced at the protocol layer; day-to-day delegation happens in the wallet. The reference wallet ships an **Intelligent Autovote** surface with four named modes:

- **Max Yield** — allocate to highest-APR clusters consistent with the per-cluster cap.
- **Max Diversity** — spread across as many independent clusters as the current cap allows.
- **Max Decentralization** — actively route stake away from clusters with high correlated-preference or geographic concentration scores.
- **Custom** — manual per-cluster allocation, with the cap enforced at submission time.

11. Bridges and the Liquidity Edge

Monolythium needs liquidity but does not need EVM execution to get it. The liquidity strategy has three layers:

1. **Zero-knowledge or light-client bridges** for major external assets where proof-bound verification is feasible.
2. **Cross-chain swaps** where proof-bound settlement is better than a full bridge route.
3. **Issuer-supported native assets** where the network earns enough adoption to justify direct issuer integrations.

Wrapped assets are labeled honestly. A bridged stablecoin is not the same as a native issuer-minted stablecoin. Wallets and explorers show the route, trust model, cooldown, proof status, drain caps, circuit-breaker state, and risk metadata.

> The goal is not to hide bridge risk; the goal is to make bridge risk legible.

Bridge cooldowns are **route-specific safety parameters**, not a single global constant. Long human-dispute windows make sense for weaker, trusted bridges; proof-verified routes can run on much tighter cooldowns. Every bridge route exposes its drain caps, circuit-breaker policy, verification model, and reserve information.

---

12. Hardware Sovereignty — Monarch OS

Production operators run on **Monarch OS** — an immutable substrate built on a minimal Linux base, hardened for institutional security and designed for permissionless accessibility. A home operator with a recent gaming PC runs on the same substrate a co-located bare-metal operator runs.

- **Immutable signed image.** No package manager. No SSH. No interactive shell. The system is purpose-built for operator workloads.
- **Verified rooted filesystem.** Every block of the filesystem is verified against a signed Merkle root at read time.
- **TPM 2.0 measured boot.** Boot-sequence hash measurements are extended into TPM PCRs. The system can prove what it booted via TPM PCR quotes.
- **TPM-sealed operator key shares.** A cluster's BLS share is sealed against the local TPM. Opening the chassis or modifying firmware breaks the seal and forces re-onboarding.
- **No userspace foothold for kernel exploits.** A modern class of kernel local-privilege-escalation bugs requires a local non-privileged userspace foothold; Monarch OS structurally denies that foothold.
- **Aggressively trimmed kernel.** The userspace cryptographic kernel API is disabled at kernel-config level — all crypto runs in-process via Rust libraries, not via kernel crypto sockets. The entire class of kernel-CVE attacks against userspace crypto APIs is closed off going forward.

Operators publish **TPM PCR quotes** every epoch to the on-chain node registry. A change in PCR values that is not preceded by a coordinated upgrade announcement is treated as an anomaly. The chain stores the attestation; explorers display it; cluster members verify against expected values. It becomes cryptographically observable when a corporate hypervisor or unauthorized management layer is introduced underneath a Monarch node.

Network-level and geographic diversity scoring is layered on top — distinct IP per operator, on-chain ASN recording, geographic claims cross-checked against IP geolocation and latency triangulation, hosting class derived from PCR patterns. Diverse clusters earn delegator preference; concentrated clusters lose it.

13. Recovery Posture

The chain's recovery posture is designed for two scenarios that are normally underspecified in L1 protocols.

Emergency-key registry

Users can pre-register a **backup key in a different cryptographic family** — specifically SLH-DSA (FIPS 205, hash-based) — alongside their primary ML-DSA-65 key. The backup key is dormant unless the protocol declares an emergency algorithm rotation. At that point, users with a registered backup sign with the backup key; users without are **frozen** — never drainable by the attacker, recoverable through a runbook-based claim process.

Hash-based signatures rest on different mathematical assumptions than lattice-based signatures. An attack that breaks ML-DSA does not, on current understanding, break SLH-DSA, and vice versa.

Emergency freeze — scope and limits

A multi-signature Foundation key with a declared signer set and ratification window can pause transaction admission during a confirmed cryptographic-primitive break, a confirmed active bridge exploit, or a confirmed adversarial fork. The freeze is **not** for routine upgrades, parameter changes, protocol-direction decisions, asset confiscation, ongoing supervision, or censorship of specific accounts.

It is a circuit breaker — a documented, accountable, time-bounded path through a worst-case event, instead of relying on improvisation under duress. The chain's no-governance design (§4.1) and the freeze mechanism are compatible exactly because the freeze is scoped to events with clear, observable triggers, not to ongoing decisions about how the protocol should evolve.

14. Threat Model in One Page

Monolythium's threat model assumes some operators are Byzantine, bridges are an active attack surface, private keys can be stolen, governance can be socially engineered (which is one reason the chain has none), and cryptographic primitives may eventually fail. The defensive posture is **separation of blast radius** — different surfaces have different protection budgets, and a failure at one surface should not compromise another.

SURFACE	PRIMARY PROTECTION	FAILURE SCOPE
Consensus	Cluster threshold + equivocation slash + DAG-BFT mathematics	Halts safety; protocol rejects equivocating operators and degrades gracefully
User signatures	ML-DSA-65 + emergency-key registry + algorithm rotation	Primitive break triggers rotation; users with backup keys survive; users without are frozen, not drained
Bridges	Light-client / zero-knowledge proof verification + drain caps + circuit breakers + per-route cooldown	Bridge failure bounded to route's drain cap; consensus and accounts elsewhere unaffected
Mempool	Threshold-DKE encryption + lifecycle-bounded confidentiality	Exposure bounded to seconds between admission and inclusion
Application contracts	Audit + native modules + sandbox boundaries	Contract bug damages the contract's users; native modules and consensus layer insulated
Hardware	TPM PCR attestation + immutable substrate + diversity scoring	Compromised operator detectable through PCR drift; concentrated hosting class detectable through diversity scoring
Recovery	Emergency-key registry + frozen-account claim flow	Primitive break or coordinated attack does not allow draining; affected accounts are recoverable

What the chain does not promise: that operators will not be compromised, that bridges will never be exploited, that smart contracts will not have bugs, that cryptography will work forever, or that no user will lose funds to social engineering. Honesty about limits is part of the threat model — a chain that claims invulnerability is a chain whose users are unprepared when invulnerability fails.

MEV

MEV is bounded structurally. The encrypted mempool forecloses front-running based on mempool visibility. Threshold-VRF leader selection forecloses leader-grinding through block-content selection. The native order book settles deterministically against the consensus order rather than against a single sequencer's discretion. The chain does not claim MEV is zero — some backrunning of public events and inter-market arbitrage exists wherever transactions are visible — but the largest extractive categories (front-running, sandwich attacks based on mempool reads) are not available on Monolythium.

15. Market Positioning

Monolythium does not have a perfect mirror competitor. Its differentiated combination is:

- AI-agent settlement as a primary category;
- Rust/RISC-V-native execution from the base layer;
- post-quantum accounts as default, not optional;
- native MRC token, NFT, market, and agent-commerce modules;
- zero-knowledge and light-client bridge liquidity rather than EVM execution compatibility;
- no on-chain governance;
- no mainnet perpetuals;
- structurally non-fungible public/private denomination;
- a public cluster marketplace with distributed validator technology;
- focused use of zero-knowledge proofs at bridges, swaps, zkML, and high-value verification;
- direct composition with the major agent-payment standards as the chain-anchored trust and settlement layer.

For agent commerce, neutrality matters because agents transact across companies, model providers, wallets, jurisdictions, and service providers. A payment rail owned by one AI lab or one platform may work inside that platform, but it is less credible as a universal settlement substrate. A bank or fintech that wants its agent flow to interoperate with the rest of the agent economy needs a substrate that is not owned by a competitor. Monolythium provides that.

16. Honest Limitations

The chain's direction is a strategic bet. The risks are real and named.

- Some developers will not migrate from Solidity to Rust.
- Liquidity may arrive more slowly without direct EVM compatibility.
- Wallets and explorers require more custom work than they would on an EVM chain.
- Native MRC standards must earn trust before they reach ubiquity comparable to ERC standards.
- Rust/RISC-V contract tooling must be excellent — anything less than excellent loses to a familiar EVM environment.
- Zero-knowledge bridge circuits are complex and require serious audit work.
- The market for agent commerce may take longer to mature than expected.
- Post-quantum signatures are larger than classical signatures, raising storage and bandwidth costs.
- Distributed validator technology has not been deployed at the chain's target scale in a leaderless DAG-BFT configuration; the operational learning is ahead, not behind.
- The bifurcated denomination is structurally hostile to certain user expectations from existing privacy chains; users coming from those chains will find the constraints unusual.
- The composition stance depends on the major agent-payment standards remaining open and composable. A standard that closes its integration surface would constrain the wedge for that particular rail.

These are not footnotes. They are the cost of choosing a distinct category. The counter-risk is also real: becoming another EVM-compatible chain may make adoption easier at first but leave the project buried under stronger incumbents, deeper liquidity, and thousands of similar competitors. The chain accepts the harder path because the easier path leads to a category in which it is already too late to win.

---

17. What Success Looks Like

Monolythium succeeds if it becomes a credible settlement layer for AI agents acting on behalf of human or organizational principals, human-to-agent commerce, agent-to-agent commerce, payments, token and NFT issuance, spot markets, cross-chain swaps, safer bridge liquidity, and long-lived post-quantum digital identities.

Structural success indicators independent of any single market hype cycle:

- the number of independent clusters and the geographic and ASN distribution of their operators;
- the depth of the discovery registry and the number of legitimate providers it hosts;
- the volume of escrowed agent-to-agent and human-to-agent transactions;
- the share of bridge volume moving through proof-bound routes versus trusted-multisig routes;
- the volume of agent-payment-standard composition (x402, AP2, ACP, MCP) settling against Monolythium spending policies and escrows;
- the diversity of MRC asset issuers;
- the survival of agent identities across model providers — the structural portability of reputation.

A network that scores well on these is a network that has delivered what the design promised. A network that scores well on token price alone has not.

18. Closing

Monolythium is a deliberate break from the default Layer-1 playbook.

It does not try to win by becoming a slightly faster EVM chain. It chooses Rust on a deterministic RISC-V target, post-quantum accounts as default, native asset standards, native markets, native agent-commerce primitives, zero-knowledge and light-client bridge liquidity, focused use of zero-knowledge proofs at the highest-value boundaries, a structurally non-fungible privacy denomination, a public cluster marketplace, and a smaller protocol surface.

It composes underneath the major agent-payment standards rather than fighting them, providing the chain-anchored policy, escrow, identity, and reputation layer those rails leave open.

That choice is harder. It means more tooling, more education, and a slower path to existing Solidity liquidity. It also gives the chain a clearer identity and a stronger foundation for the workloads it expects to serve.

The thesis is straightforward: the next major settlement layer will not be the chain that copies Ethereum most closely. It will be the chain that gives new economic actors — autonomous agents, organizations delegating to software, services that want neutral cross-platform rails — safer rails to transact across applications, markets, and jurisdictions.

Read Further

This lightpaper distils the Monolythium Whitepaper v5.0. For implementation detail, full threat model, cryptographic specifications, the consensus mathematics, the cluster ceremony, the recovery runbook, and complete citations, the whitepaper is the canonical reference (~110 pages).

Entities

Monolythium is operated and stewarded by two entities:

- **Mono Labs R&D LLC** — the operating company, based in San Francisco, California. Mono Labs R&D LLC develops the Monolythium protocol, operates the reference client and SDK, and is the seller of record for the LYTH token in jurisdictions where a seller of record is required.
- **Monolythium Foundation** — the protocol steward, based in the Cayman Islands. The Foundation operates the chain's emergency-key registry, the Foundation multisig treasury, the genesis name reserve, the emergency-freeze ratification window, and other constitutional-layer functions that require an entity rather than a company.

The two entities are independent, with separate governance, separate financial accounts, and separate roles in the protocol's operation.

License

The text of this lightpaper is licensed under the **Creative Commons Attribution-ShareAlike 4.0 International License (CC BY-SA 4.0)**, the same license as the whitepaper it summarises.

Attribution string:

> Monolythium Lightpaper, v5.0 (May 2026), Mono Labs R&D LLC, licensed under CC BY-SA 4.0.

The Monolythium protocol source code is licensed separately under the Business Source License 1.1, with a four-year commercial restriction window before automatic conversion to a permissive license. Selected execution crates ship under MIT.

End of Monolythium Lightpaper v5.0.