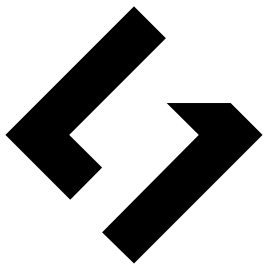

GenesisL1

A Sovereign Layer-1 for Verifiable Scientific Infrastructure

One public trust environment for data, models, computation, and rights.

Technical Whitepaper · Version 1.0

July 2026

Mikhail Fedorov, M.D.

Medical doctor · Bioinformatics and data science

Founder, GenesisL1

github.com/GenesisL1

Research and development contributors: Decentralized Science Labs LLC and Bioinformatics LLC.

Abstract

The scientific infrastructure of tomorrow is a shared, verifiable environment in which data, models, computations, results, and rights form one continuous workflow—and people and software independently verify its history. **GenesisL1** is an operational foundation for that future: a live, public, permissionless Layer-1 built as a universal scientific trust, verification, and collaboration layer. Researchers, laboratories, public institutions, software services, and autonomous systems—including LLM-based agents—can coordinate through the same signed scientific state, independent of any single platform owner, database operator, or model host.

GenesisL1 has already placed the core primitives of a verifiable scientific workflow on one live, permissionless ledger. **MOLNFT** represents molecular structures as on-chain scientific objects, with more than 733,919 tokenized records across first-generation collections and roughly 229,000 PDB parent records represented in second-generation chain state. Complete local reconstruction has been demonstrated for selected structures; checksum-complete corpus validation remains pending. **GL1F** (GenesisL1 Forest) commits fixed-point gradient-boosted models as Model NFTs and executes deterministic EVM inference over publicly inspectable serializations. A compact quantized neural-policy contract extends the demonstration to neural inference inside the EVM. **CIPNFT** adds encrypted capsules, provenance, recipient-bound key envelopes, and an evidence-rich bilateral exchange flow for scientific intellectual property. Together these components make scientific data, model identity, deterministic execution, encrypted disclosure, and governance mutually composable within one decentralized environment.

This integrated trust environment is the principal systems contribution of GenesisL1. As deployed in July 2026, the network combines fully on-chain molecular structures, native fixed-point Model NFTs, deterministic EVM model execution, neural inference, encrypted scientific IP, and on-chain public-goods governance within one live scientific stack. For a researcher, the ledger becomes an executable provenance graph. For an institution, a published model and its inputs can be re-executed and independently examined; an institution operating its own node and indexer retains a locally controlled replica of public scientific state and an institution-controlled query path. For an AI agent, scientific objects and models become discoverable, callable, and economically composable. For builders, the same open substrate can support new domains without asking permission from a platform owner. This integration creates systems-level novelty: a live scientific state graph whose deployment scale and operational composition produce capabilities no isolated component supplies.

GenesisL1 has produced blocks since November 2021. Because it is a sovereign L1, its canonical state, execution, resource policy, and protocol evolution are governed within the network rather than inherited from a host chain. Canonical operation requires no upstream blockspace, foreign gas asset, sequencer, bridge, proprietary model API, or single database operator. The native L1 coin meters consensus transactions, scientific state, and computation; bonds proof-of-stake security; weights governance; and funds public infrastructure through the community pool. The 21,000,000-coin genesis mint was distributed outward through a public one-to-one transition, with no token sale and no reserved or preferential founder or investor allocation. Ledger consensus makes provenance and execution independently inspectable; scientific validity remains grounded in domain evidence and replication. With its core scientific primitives already live, GenesisL1 is a foundation for workflows spanning data, models, computation, rights, and governance. Protein-stability prediction is the next cross-protocol workflow planned for independent evaluation; GLAST-RIGID-1 is the longer-horizon structural-alignment research direction and is not yet validated. GenesisL1 is built as a collaboration and verification layer for humans and autonomous software—and as a backbone for the scientific infrastructure of tomorrow.

Contents

1 Introduction

1.1 The missing public trust layer for science

Modern science is overwhelmingly an exercise in producing, sharing, and recomputing upon data: protein structures, genomic sequences, clinical-trial outcomes, instrument telemetry, simulations, spectroscopic measurements, and the trained machine-learning models that summarize them. Yet the substrate on which this knowledge is mediated has not kept pace with its volume or its epistemic requirements. A widely cited survey published in *Nature* found that more than 70% of researchers had failed to reproduce another scientist's experiment, and over half had failed to reproduce their own [2]. The pillars of public scientific data—the RCSB Protein Data Bank [3], AlphaFold predictions [4, 5], UniProt, ENA, and NCBI—are remarkable feats of curation and remain canonical scientific sources. They are institutionally administered distribution systems: continuity, version history, access policy, and identity ultimately depend on their operators, while executable provenance typically resides in separate metadata, code repositories, and compute environments. What science lacks is a permissionless trust layer in which artifacts, execution, rights, and governance share one ordered public history.

GenesisL1 supplies that layer by applying the strongest property of public blockchains—a globally replicated, Byzantine-fault-tolerant state machine—to the scientific record itself [7, 8, 9]. Where most chains direct consensus toward financial coordination, GenesisL1 makes verifiable knowledge the primary state; monetary mechanisms secure, meter, and govern that state.

1.2 Vision: the chain as a scientific environment

> **Thesis**
>
> GenesisL1 is sovereign scientific trust infrastructure: a public ledger that records, executes, and coordinates scientific objects—molecules, models, datasets, encrypted IP, derivations, and proofs. Its state is shared, signed scientific memory; its smart contracts are reproducible computational apparatus. The same open primitives support every domain that requires verifiable data, deterministic models, durable digital assets, or autonomous software commitments.

Three commitments follow from this reframing.

1. ▶ **Data and protocols are first-class.** GenesisL1 is a sovereign Cosmos/EVM scientific ledger whose protocol and application layers make structured scientific data, addressing, and deterministic computation native public capabilities.
2. ▶ **Governance begins at the network boundary.** Validator candidacy, delegation, and proposal participation are open under protocol rules; the active set is stake-ranked and capped, and its distribution is measured publicly and can evolve through open delegation and participation (section 11.6). Domain-specific scientific DAOs can coordinate on top.

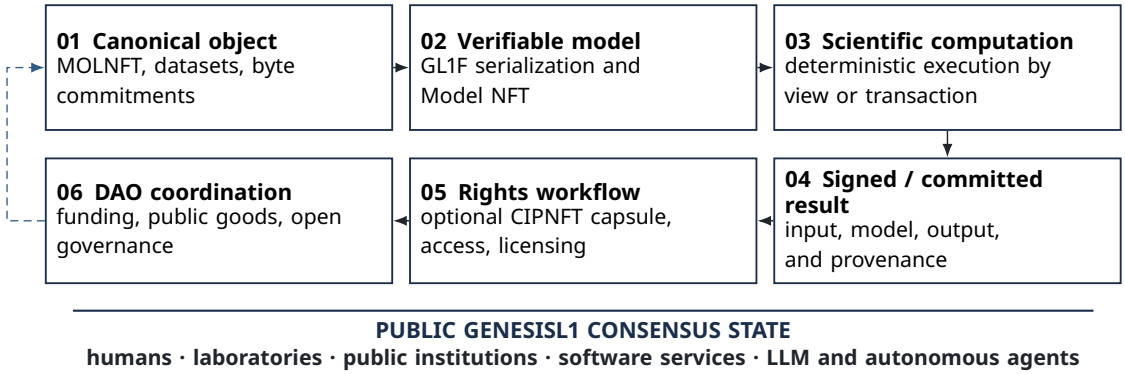

Figure 1. Reference architecture for an end-to-end scientific workflow on one permissionless state layer. Scientific objects and model definitions become inputs to deterministic execution; results, rights, and governance actions remain composable in the same signed environment. Each arrow denotes an explicit identity or state dependency, not scientific validation. The dashed return path denotes iterative discovery and reuse.

3. ▶ **The native asset makes permissionless scientific state sustainable.** L1 meters scarce state and execution, is bonded for consensus, weights governance, and denominates the community pool. It unifies resource admission, security, protocol change, and public-goods funding within the scientific ledger.

> **Scientific trust boundary**
>
> Subject to the specified ledger state, software version, archive availability, and consensus assumptions, GenesisL1 establishes public ordering, byte commitments, account authorization, deterministic contract execution, and provenance. These are the trust guarantees on which scientific workflows compose. Data quality, preprocessing choices, scientific or clinical validity, lawful rights, confidentiality after disclosure, and off-chain availability remain governed by measurement, peer review, law, security practice, and independent replication [1].

1.3 Document scope and contributions

This whitepaper consolidates and supersedes earlier component specifications [39, 40, 41, 42]. Its contributions are:

- a single-document articulation of the *web3-laboratory* thesis of GenesisL1 and the data-first design principles that follow from it (section 3);
- a technical account of the layered protocol architecture—MOLNFT (section 4), GL1F (section 5), on-chain neural inference (section 7), and CIPNFT (section 8)—as concrete instantiations of the thesis;
- a specification of the governance model under which scientific DAOs federate around the chain (section 9); and

- a factual account of L1 coin as gas, stake, governance weight, and public-goods funding medium, with live parameters and concentration metrics stated at a reference height (sections 10 and 11).

Systems-level novelty. GenesisL1's distinctive contribution is the live integration, deployment scale, and cross-component behavior of scientific corpora, deterministic model runtimes, encrypted capsules, and public-goods governance. It unifies established consensus, EVM, cryptographic, and machine-learning foundations into a scientific network whose canonical objects, execution, resource economics, and governance interoperate as one trust layer.

Table 1. Why GenesisL1 is different. The comparison concerns design emphasis, not an assertion that other public ledgers cannot host scientific applications.

Dimension	Common general-purpose emphasis	GenesisL1 scientific implementation
Primary state	Assets, balances, and application state	Canonical scientific objects and commitments
Models	Off-chain services or application-specific model calls	Versioned fixed-point Model NFTs and deterministic inference
Computation	General smart-contract execution and settlement	Scientific workflows over identified data, models, and methods
NFTs	Generic identity, access, or ownership representations	Molecular objects, model identities, datasets, and encrypted capsules
Governance	Protocol and financial coordination	Protocol governance, scientific DAOs, and public goods
Provenance	Transaction and ownership history	Executable lineage linking input, model, computation, result, and optional rights state

1.4 Claim status at a glance

GenesisL1 combines a deployed network with an active research program. The table provides a compact evidence map across operational infrastructure, demonstrated prototypes, work in progress, and planned extensions. Every "live" row can be reproduced from the endpoints and repositories in appendix E.

1.5 From deployed substrate to the first complete scientific vertical

GenesisL1 already provides an integrated scientific substrate: canonical molecular objects, deterministic model runtimes, encrypted IP capsules, search and alignment tooling, and open network coordination. The next unifying demonstration is a peer-reviewed workflow in which a MOLNFT structure is consumed by a scientifically validated GL1F model and independently reproduced from committed inputs.

The scientific-vertical concept is essential: these components are valuable not merely as separately deployed protocols, but because their identities and outputs compose into one inspectable workflow. The sequence in fig. 1 is the reference vertical: MOLNFT object → GL1F model → scientific computation → signed result → optional rights state → DAO and public-goods coordination.

Table 2. Evidence state of the principal achievements and next deployments. Labels distinguish live on-chain infrastructure, bounded proofs of concept, active development, and planned work.

Claim	Status	Where
Chain live and producing since Nov 2021 (Cosmos ID `genesis_29-2`; EVM ID 29)	Live	appendix E
MOLNFT: molecular structures tokenized on-chain (>733k), incl. fully on-chain macromolecules	Live	section 4
GL1F: deterministic gradient-boosted model inference in the EVM, via Model NFTs	Live	section 5
GLAST-RIGID-1: proposed deterministic, GL1F-guided pairwise structural alignment	Research design; implementation and independent validation pending	section 6
Neural policy network executed inside the EVM (4,672-action space)	PoC	section 7
Generalization of on-chain neural inference to production scientific ML	Candidate direction	section 7.5
CIPNFT encrypted capsules, recipient-bound envelopes, controlled disclosure, and staged transfer	Live core; v1 identified-counterparty exchange	section 8
Tokenomics: 21M genesis mint and public 1:1 transition; no sale or reserved founder/investor allocation; governed inflation; non-binding 100M soft-cap target	Live / on-chain record	sections 11 and 11.4
Protein-stability ($\Delta\Delta G$) model consuming MOLNFT inputs	First independent scientific workflow; in progress	sections 5.9 and 14
Canonical genomic format (GL1ENCv2) and reference genome on-chain	Spec / in progress	section 14
Rewarded IPFS sidecar for large-data availability	Planned	section 14
Comprehensive independent security audit of the node integration and canonical core contract suites	Not completed as of July 2026	section 13.7

The near-term evidence milestone is deliberately focused: one scientifically validated, parity-tested result that consumes a canonical on-chain scientific object and publishes an independently reproducible model output. Protein stability is designated as the first workflow to undergo independent scientific validation. Completion would join the deployed data and model components into the first domain-validated vertical and establish a reusable template for additional disciplines. The longer-horizon flagship application, GLAST-RIGID-1 (section 6), extends that template from scalar prediction to deterministic, model-guided scientific search while keeping geometry canonical.

Table 3. Present scientific utility and the next evidence milestone for each deployed component.

Component	Usable now	Next validation milestone
MOLNFT / deployed GLAST sequence search	Reconstruct complete molecular records from chain state; query and perform local sequence alignment over the on-chain corpus.	Publish checksum-complete corpus validation and a state-capacity and replication-cost report.
GL1F	Publish, discover, invoke, and independently re-execute fixed-point gradient-boosted models committed as Model NFTs.	Deploy and independently evaluate a scientific MOLNFT→GL1F workflow with local/on-chain parity.
Neural EVM PoC	Re-run a nontrivial quantized policy network at a pinned contract state.	Publish immutable scientific deployments and domain-specific validation; chess supplies feasibility evidence.
CIPNFT	Anchor ciphertext, provenance, integrity commitments, and recipient-bound disclosure; conduct evidence-rich v1 exchange among attributable counterparties.	Complete the hardened successor, migration path, and independent smart-contract and cryptographic audit.
Network / institutional operation	Operate a full or archive node and local indexer; retain a locally controlled replica of public scientific state; serve internal retrieval and execution; seek validator entry, delegate, vote, and inspect public resource allocation.	Publish institutional operating profiles, broaden independent node ownership, diversify voting power, and eliminate the dormant blocklist surface.

2 Architectural foundations

2.1 Dual-environment design: Cosmos SDK + EVM

GenesisL1 is implemented as a Cosmos-SDK application chain [10] executing CometBFT—the Byzantine-fault-tolerant consensus engine that is the successor to Tendermint Core [9, 11]. Atop the SDK, an integrated Ethereum Virtual Machine (EVM) environment supports common Solidity tooling and interfaces, subject to chain-specific implementation differences, including the ERC-20, ERC-721, and ERC-1155 families [12]. This dual-environment architecture is useful to the data-first design for three reasons.

1. Cosmos-SDK modules (`bank`, `staking`, `distribution`, `slashing`, `gov`, `mint`, community pool, `ibc`) provide widely used open-source building blocks for monetary, governance, and inter-chain operations. Their upstream maturity does not constitute an audit of the GenesisL1 integration.
2. The EVM environment inherits a large, mature body of open-source smart-contract patterns, security audits, tooling, and developer mindshare, lowering the cost of building scientific dApps relative to bespoke virtual machines.
3. *Deterministic, replicated execution* is a precondition for using on-chain computation as a scientific instrument: every conforming full node executing a committed transition under the same historical binary and state must obtain the same result.

Table 4. Core dependencies of GenesisL1 `v1.6.2` (tag commit `ee86be8`). Fork commits, not marketing-level version ranges, are the reproducibility anchors.

Component	Version	What it brings
Cosmos SDK	declared `v0.53.0`; fork `8d0a31e`	Module and store framework on a Crypto.org fork lineage
CometBFT	`v0.38` line; fork `178ea85`	BFT consensus engine and patches
IAVL (state)	`v1.2.6`	Authenticated state tree
go-ethereum EVM	declared `v1.15.5`; fork `a4fbafc`	Crypto.org fork based on `v1.10.20` with selected backports
ibc-go	`v10.1.1`	IBC protocol stack; ICA host disabled at the reference state
Toolchain	Go `1.23`, gRPC `v1.75`	Modern build, indexing, and JSON-RPC surface

2.2 The v1.6.2 protocol stack and coordinated upgrades

Abstract compatibility claims are only as strong as the running software behind them. As of the `v1.6.2` network upgrade, GenesisL1 integrates current Cosmos/CometBFT interfaces with Crypto.org/Cronos-derived forks (table 4). Exact commits matter: upstream version labels alone must not be read as equivalence to unmodified upstream releases. The upgrade rebuilt consensus, state, and execution dependencies together.

▶ **Selected current EVM features.** Source inspection confirms support for `PUSH0` [16], transient storage (`TSTORE`/`TLOAD`) [17], and `MCOPY` [18]. This paper does not claim Ethereum-equivalent blob data availability: type-3 propagation, KZG handling, and sidecar behavior require a separate end-to-end conformance test. These primitives matter to the scientific protocol stack directly—transient storage, for example, provides cheap intra-transaction scratch space for the tree-traversal accumulators of GL1F inference (section 5) and the parent–child reconstruction of MOLNFT v2 (section 4.4).

▶ **Set-code transactions.** GenesisL1 includes EIP-7702 authorization handling [20]: an externally owned account can delegate execution behavior within the specified transaction model. This is a building block for batched calls, sponsored-gas flows, and programmable wallets, not a complete account-abstraction system. For scientific dApps this lowers a real barrier—a laboratory or DAO can, for instance, sponsor the inference or minting fees of its collaborators, so that using the chain does not require every contributor to first acquire `L1`.

▶ **Recipient-key registry.** The `e2ee` module registers X25519/age recipient keys for accounts and supports client-side encryption workflows. Encryption and decryption occur off-chain; the module is not confidential execution, does not by itself provide forward secrecy, and does not remove endpoint or key-compromise risk. It complements, but is cryptographically separate from, CIPNFT (section 8).

▶ **Upgrade discipline as a security property.** Reaching `v1.6.2` was not a single patch but two coordinated, governance-ratified network upgrades—first to `v1.1.1`, then a single atomic jump that telescoped the `v1.3` → `v1.4` → `v1.5` → `v1.6` release lines

into `v1.6.2`. Each migration reached consensus activation without an observed persistent chain split. Approval scheduled the upgrade; validators still had to obtain and run the intended binary. Reproducible builds, supply-chain review, and staged testing therefore remain security requirements (section 13).

2.3 Consensus, finality, and adversary model

CometBFT's PBFT-derived consensus (the Tendermint protocol) provides immediate finality after a block receives commits from more than two-thirds of bonded voting power [11]. Under the standard model, safety requires less than one-third Byzantine voting power, while progress requires more than two-thirds participation under partial synchrony. Not every practical attack produces complete slashable evidence: key compromise, bribery, correlated operations, and governance capture can reduce effective security. Practical consequences for GenesisL1:

- **Fast, sub-minute finality** for scientific publications on-chain—there is no probabilistic "six-confirmations" wait before a tokenized dataset is considered durable.
- **Observable economic exposure:** bonded `L1` and its concentration are security indicators, not strict fiat attack-cost floors. More than one-third can obstruct progress; control of at least two-thirds can finalize state.
- **Slashing of validator and delegated stake** applies to specified faults. Under the current staking implementation, slashed tokens are burned; the community pool is funded separately.

2.4 State, storage, and content addressing

On-chain state on GenesisL1 stores compact, hashable, cryptographically addressable artifacts: balances, contract code, ERC-721 metadata, encrypted ciphertexts (section 8), gradient-boosted decision-tree coefficients (section 5), and proofs. GenesisL1 supports two complementary strategies for the bulk bytes of science:

Content-addressed reference.
Multi-megabyte artifacts (e.g. coordinate files) are addressed by IPFS content identifiers (CIDs) referenced from on-chain NFTs [22, 23]. A CID is not simply a file hash. CIDv1 is a self-describing typed content address containing a version, a multicodec identifying how the addressed block is interpreted, and a multihash identifying the hash function and digest; its text form additionally uses multibase. For a multiblock file, the digest normally commits the root of a directed acyclic graph rather than the raw file bytes alone. Reproducing a CID therefore requires the import profile—including chunking, DAG layout, codec, CID version, hash algorithm, and any canonical serialization—to be fixed. Given identical bytes and profile, independent importers can reproduce the same CID; a content or profile change can produce a different CID. The commitment establishes identity and integrity, not persistence: availability still requires at least one party to retain and serve the blocks.

Fully on-chain storage.
Where consensus-replicated access is warranted, GenesisL1 stores the bytes *themselves* in contract state or contract code—the "code-as-data" pattern used pervasively by MOLNFT v2 (section 4.4), GL1F model chunks (section 5.5), and CIPNFT

ciphertext (section 8). Here the asset *is* the data. Availability still inherits node liveness, pruning, archive retention, and state-sync assumptions.

2.5 Scientific-scale storage: placement, replication, and operator economics

GenesisL1 treats scalability as a state-placement discipline, not as a claim that storage is free. Every conforming full validator replicates the canonical current state; validators do not choose arbitrary subsets of that state. Selection occurs before admission: the protocol and its users decide which objects warrant universal consensus replication, which large payloads should be represented by content-addressed commitments, and which indexes can remain local and rebuildable.

Table 5. Storage hierarchy for scientific workloads. Replication cost is matched to the guarantee the object needs.

Storage class	Guarantee	Recommended use
Fully on-chain	Current-state bytes are replicated by every full validator and directly readable by contracts.	Canonical scientific objects, compact models, rights state, and bounded artifacts whose composability warrants universal replication.
Content-addressed	A CID, hash or Merkle root, schema, size, and provenance are committed on-chain; independent providers retain the bulk bytes.	Large datasets, imaging, sequencing runs, simulations, and training corpora.
Local indexes	Derived state is rebuilt from the canonical corpus or commitments and is not part of consensus.	Search indexes, caches, feature stores, and alignment indexes.

Logical scientific payload and physical node storage are different quantities. Let P_{full} denote the application payload admitted to fully replicated live state, α the database and representation amplification, S_{protocol} non-application state, H_{history} retained history, and B_{backup} snapshots and backups. A first-order operator model is

$$S_{\text{validator}} \approx \alpha P_{\text{full}} + S_{\text{protocol}} + H_{\text{history}} + B_{\text{backup}}. \tag{1}$$

Pruning changes H_{history} but does not remove the obligation to retain current live state. Rebuildable GLAST indexes are measured separately. A node that voluntarily serves the planned storage sidecar incurs an additional provider-specific bulk-data term; validator operation and sidecar provision are not assumed to be the same role.

The reported approximately 50 GB MOLNFT v2 scientific payload matters because it establishes a material, deployed logical-data baseline. It must not be read as a 50 GB validator-disk measurement: contract, EVM, Cosmos, key–value database, history, index, snapshot, and backup overheads can make physical footprint larger, and the reference figure still requires a height-pinned extraction manifest and checksum-complete independent measurement. A publication-grade capacity report should disclose those quantities together with initial-sync and state-sync time, pruning and archive profiles, read amplification, backup duration, hardware, and annual growth.

The scaling rule is therefore selective *placement*, followed by complete replication of whatever enters consensus. Compact, high-value canonical artifacts belong directly

on-chain. Large corpora can retain GenesisL1 identity, provenance, versioning, and incentive settlement through compact commitments while the bytes are replicated by a separately specified provider network. The rewarded IPFS sidecar exists to make that availability durable and economic: users would request a replication factor and duration, independent providers would retain and serve the committed blocks, and verifiable service would earn `L1` funded by user payments and governance-authorized public-goods support. The sidecar is in development; its proof, repair, Sybil-resistance, and reward mechanisms require specification, testing, governance, and audit before deployment.

2.6 Interoperability

Inter-Blockchain Communication (IBC) [24] is integrated through ibc-go `v10`; actual connectivity depends on open clients, connections, channels, and relayers at the time of use. Interchain Accounts host functionality is disabled at the reference state, so cross-chain invocation of GenesisL1 contracts is a future capability requiring governance activation, integration testing, and explicit security review.

3 The chain as a decentralized laboratory

3.1 Beyond the monetary chain

The dominant frame for public blockchains is monetary: a ledger of balances, payments, and DeFi positions, occasionally annotated with non-financial state. GenesisL1 inverts the priority. The chain's purpose is to be a shared, signed, executable scientific record; monetary primitives are an essential but subordinate component, present because they are required to price scarce computation and to align security incentives.

> **Scientific success metrics**
>
> GenesisL1 measures scientific success through independently checkable external evidence, not TVL or token price, which do not measure scientific validity. Relevant indicators are independently reproduced workflows; reproduction and validation by third-party laboratories; independently operated validators, archive nodes, and institutional nodes or indexers; models and applications deployed by unaffiliated teams; peer-reviewed publications; and citations in the scientific literature. Object counts measure engineering scale, not scientific success by themselves.

3.2 Why a sovereign L1 and a native asset?

For many scientific tasks, a blockchain is unnecessary. If the requirement is only to publish a static artifact, assign a persistent citation, sign a hash, or preserve a reproducible environment, a DOI repository such as Zenodo, a signed content-addressed object, and a versioned container may be the simpler and better tool. GenesisL1 is not proposed as a replacement for those systems; it can commit to and compose with them.

The dedicated-ledger case begins where the problem becomes multi-party and stateful. A dataset, model, result, access right, payment, and governance decision can refer

to one another inside the same consensus-ordered state machine. A contract can consume the exact registered model and input, write a result that points to both, transfer an associated right, and expose the entire dependency graph to the next application without trusting a repository administrator, container registry, model host, or bilateral database integration. Signed content addressing proves that bytes match a hash; it does not by itself supply a common execution environment, atomic cross-application state transitions, open resource allocation, or a governance process for changing the shared substrate.

Table 6. Decision boundary between a conventional reproducibility stack and a permissionless scientific ledger.

Requirement	DOI / signed CID / container	GenesisL1 shared state
Exact artifact identity	Strong: citation, hash, signature, packaged environment	Strong: byte commitment or replicated bytes, signer, ordering, and state history
Re-executable method	Strong when images, inputs, and runtimes remain available	Deterministic for committed on-chain code and state; off-chain preprocessing still requires separate preservation
Multi-application composition	Requires APIs, registry operators, and bilateral integration	Contracts share canonical objects and may update related state atomically
Public resource allocation	External institutional process	Gas, staking, governance, and community-pool decisions share one public ledger
Primary failure mode	Repository, key-registry, image, runtime, and institutional dependencies	Consensus concentration, replicated-state cost, software defects, governance capture, and token-economic burden

> **Decision rule**
>
> Use a repository, signed CID, or container when the problem is publication and preservation. Use GenesisL1 when independent parties or software agents need the same publicly ordered scientific state, deterministic execution over that state, composable rights or settlement, and governance without a single institutional operator.

Where that decision rule is met, GenesisL1 operates as infrastructure in its own right. Canonical scientific state, execution, consensus security, gas, and protocol governance are native to the same ledger. Operation of the on-chain stack requires no host-chain blockspace, rollup sequencer, bridge, foreign gas asset, proprietary model API, or single database operator. Validators finalize GenesisL1 state directly, while the native L1 asset bonds consensus and meters the transactions that change that state. External repositories and content-addressed networks can connect at the edges for source citation and bulk availability; GenesisL1 remains the canonical coordination layer for commitments, identity, provenance, execution, rights, and state transitions.

Sovereignty matters because scientific infrastructure has priorities that need not match those of a general-purpose host. GenesisL1 can organize protocol evolution around long-lived structured data, deterministic model execution, archive access, sponsored scientific transactions, and machine-readable interfaces. Its scientific state cannot be reprioritized or discontinued by an upstream ledger operator. The network itself publicly governs resource policy, upgrade priorities, and public-goods allocation.

Given a need for shared ordered state, the native stack supplies the independent history, execution, resource discipline, institution-operated access, and public governance needed to sustain it.

Table 7. The native sovereignty stack available when scientific collaboration requires shared ordered state.

Infrastructure function	Native mechanism	Scientific consequence
Independent history	GenesisL1 validator set and CometBFT finality	Canonical state is secured and ordered without renting another ledger
Institution-operated access	Full or archive node plus local indexer	A locally retained public-state replica and institution-controlled query path
Unified execution	EVM contracts over shared canonical objects	Data, models, results, rights, and settlement can update one another atomically
Bounded resources	L1 gas for transactions, computation, and persistent state	State growth is priced; compact models and efficient encodings internalize less network cost
Adaptable substrate	On-chain governance plus public, validator-adopted upgrades	Exposed parameters and deeper protocol behavior can evolve around scientific workloads
Scientific public goods	Native community pool	Archives, audits, datasets, tooling, replication, and access programs can be funded transparently

At this permissionless coordination layer, L1 is the operating resource that makes the system open while preserving scarcity and accountability:

1. ▶ **Consensus security.** Validators and delegators bond L1 to order and finalize the state containing scientific objects, models, results, and governance actions. Protocol rewards compensate participation and defined misbehavior is subject to slashing.
2. ▶ **Scarce-state and computation discipline.** Contract deployment, persistent writes, and transaction-executed computation consume gas in L1. Because every persistent byte is replicated across the network, larger artifacts internalize more of their admission cost, discouraging spam and favoring compact encodings and high-value canonical state. Read-only `eth_call` and local inference remain available without a consensus transaction.
3. ▶ **Open model and application competition.** Builders publish models into the same registry, execute them through the same deterministic runtime, and may

denominate deployment, access, inference, licensing, or settlement in L1. Users and software agents can compare provenance, published validation, size, execution cost, access terms, and outputs on a neutral surface. Scientific quality remains determined by validation; the ledger makes the competition open and auditable.

4. ▶ **Governance and public goods.** Bonded L1 weights governance, while the community pool can support validators, archives, audits, reference datasets, scientific tooling, access subsidies, and independent replication. Scientific use, network security, and maintenance of the commons therefore operate through one inspectable resource system.

The same sovereignty makes GenesisL1 adaptable to the applications it serves. Corpus minting stresses persistent state; GL1F stresses code storage and deterministic reads; sponsored transactions shift fee burden; and archive nodes carry long-horizon replication costs. Through `x/gov`, the community can change parameters exposed by the fee-market, mint, staking, governance, and related modules. Changes to EVM semantics, gas accounting, block limits, block timing, or storage behavior proceed through public, versioned software upgrades adopted by validators. Protocol change is therefore visible, collectively authorized, and able to follow measured scientific workloads.

An institution can operate a full or archive node and indexer beside its scientific systems. Routine chain queries, caching, derived indexes, and application access can remain within its operational perimeter, while submitted commitments and computations join the global ledger. The resulting benefit is operational control of a local public-state replica and its query path; regulated plaintext, private keys, and controlled preprocessing environments remain inside the institution's compliant systems.

> **The native-stack argument**
>
> A hosted scientific application controls its contracts; GenesisL1 additionally places the ledger beneath them—scientific state, deterministic execution, consensus security, scarce-resource pricing, public-goods funding, and protocol evolution—within one permissionless network and one public governance domain.

3.3 Properties of the ledger

The chain-as-ledger frame imposes specific technical properties.

▶ **Reproducibility by construction.** A committed contract transition at height h can be replayed when the historical state, node binary, chain configuration, and input transaction are preserved. The ledger makes those execution inputs more inspectable; it does not eliminate dependencies in off-chain preprocessing, training, feature extraction, or experimental design [30, 1].

▶ **Provenance as a primitive.** Every state transition is signed by the originator's key and ordered by validator signatures. "Who deposited this molecule, when, and from what address" is not metadata—it is the transaction itself.

▶ **Open institutional replication and validator candidacy.** Any laboratory, university, agency, public archive, or independent operator may run a full node and may

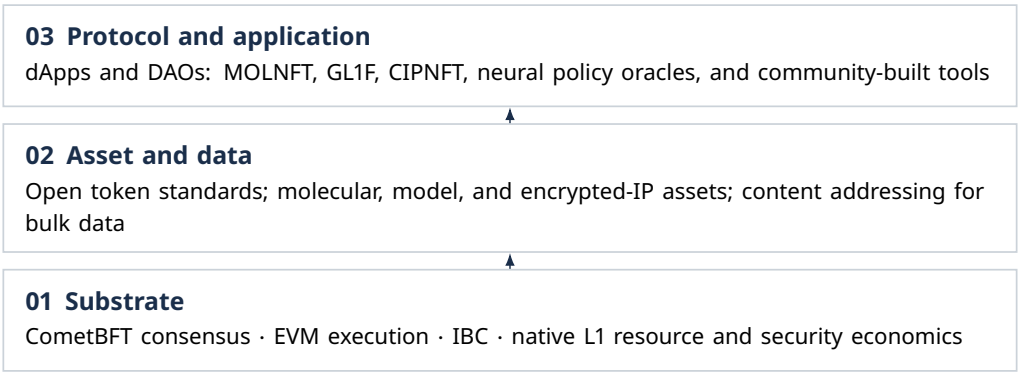

Figure 2. The three conceptual layers of GenesisL1. Each layer depends only on those beneath it; the native L1 coin economics live in the substrate and propagate resource pricing and security upward.

seek delegated stake to enter the active validator set. A locally operated node and indexer provide an independently retained replica of public chain state, an institution-controlled query path, and resilience against public RPC or application-host outages. Consensus participation remains stake-ranked and capped; current distribution is reported in section 11.6.

▶ **Institutional data boundary.** Official integrations place public or anonymized scientific artifacts and lawful cryptographic commitments on-chain while regulated plaintext, keys, preprocessing environments, and access records remain inside compliant institutional systems. Running a local node keeps routine chain queries and derived indexes inside the institution's operational perimeter. Because the ledger itself is public and permissionless, this is an application and governance architecture; each party remains responsible for the data it handles under HIPAA, GDPR, and other applicable regimes. The detailed immutable-data threat model appears in section 13.

3.4 Three layers of the ledger

We organize GenesisL1's stack into three conceptual layers, each building on the one beneath it (fig. 2).

1. **Substrate layer.** Consensus, EVM execution, IBC, native L1 coin economics—the chain itself.
2. **Asset and data layer.** Open token standards, NFT contracts for molecules, models, and encrypted IP, and the content-addressing of bulk scientific data.
3. **Protocol and application layer.** dApps and DAOs—MOLNFT, GL1F, CIPNFT, and the long tail of community-built tools.

4 Protocol stack I — MOLNFT: molecular data on-chain

4.1 Motivation

The Protein Data Bank is the canonical international source of experimentally determined biomolecular structures [3]. AlphaFold and ESMFold extend that public structural universe at extraordinary scale [4, 5, 6]. MOLNFT [42] preserves those canonical sources and citations while adding a distinct infrastructure property: selected structures become exact, timestamped, consensus-replicated objects that institutions can retrieve locally and contracts can compose with directly.

4.2 MOLNFT v1: tokenized molecular databases

MOLNFT v1 is a family of ERC-721 contracts on GenesisL1 that mint one non-fungible token per structural entry, with on-chain metadata pointing to an IPFS CID of the underlying coordinate file (PDB, mmCIF, or model PDB):

- **MOLNFT PDB** mirrors the RCSB PDB corpus as ERC-721 tokens; the genesis batch covered the snapshot of experimentally determined structures at minting time.
- **MOLNFT AF** mirrors the AlphaFold Swiss-Prot prediction set, each token pointing to the relevant model coordinate file.
- **MOLNFT HLA** mirrors a corpus of ESMFold-predicted HLA-related structures.

The aggregate footprint exceeds 733,919 tokenized molecular structures, forming a material-scale scientific NFT corpus whose records and commitments are publicly inspectable. Each token discloses, in its on-chain metadata or IPFS payload, the canonical citation to the originating source, thereby *preserving—not displacing*—the academic provenance.

4.3 Infrastructure properties of molecular tokenization

MOLNFT is not a property claim over a structure; no token can "own" a public-domain scientific fact. It provides four distinct infrastructure-level properties.

1. **Cryptographic mirror.** The on-chain commitment binds the exact byte content of the structure at the moment of minting; any future divergence is detectable.
2. **Integrity-verifiable content reference.** A scientist can resolve the recorded CID and verify any retrieved bytes against it. Availability still depends on at least one retained IPFS copy.
3. **Composability.** Any contract on GenesisL1 may reference a MOLNFT token ID as an input: an on-chain bioinformatic pipeline can declare "I operated over PDB entry X at NFT id n" as a *verifiable* assertion.
4. **Discoverability and indexing.** A unified schema across PDB, AlphaFold, and HLA tokens allows cross-database queries that would otherwise span institutional silos.

4.4 MOLNFT v2: complete structures, fully on-chain

MOLNFT v2 writes complete macromolecular structures directly into GenesisL1 state. Canonical public repositories remain the scientific source; MOLNFT adds a byte-complete on-chain representation that contracts, models, and pipelines consume as native shared state.

> **The asset is the data**
>
> Each structure is stored as `base64(gzip(BinaryCIF))`. A GenesisL1 node reconstructs *byte-identical* BinaryCIF from contract state, placing retrieval under transparent ledger-state, node-liveness, and archive-retention assumptions rather than an external object URL.

Scale. The MOLNFT v2 deployment writes roughly 229,000 PDB structures and ~1,000,000 sequences—about 50 GB of reported logical scientific payload—into GenesisL1's replicated state, a material smart-contract scientific-data deployment (canonical contract `0xd58B01f6C18086e5202cdC5D7Ad3E41790360102`).

Parent–child architecture. Because blockchain storage has practical per-transaction and per-object size limits, MOLNFT v2 uses a hierarchical parent–child NFT structure:

- a **parent NFT** represents a full entry (a primary PDB record);
- **child NFTs** store data fragments (chunks of the compressed BinaryCIF payload);
- a `getCombinedData` view reconstructs the entire molecule from its child tokens, transparently to the caller.

Metadata (title, authors, resolution) and binary payloads (coordinates, sequences) are all stored immutably in chain state; retrieval never depends on off-chain URLs or IPFS gateways.

GLAST: web3 bioinformatics. Hosting large datasets on-chain is only half the challenge; search and analysis are equally crucial. **GLAST**—the GenesisL1 Local Alignment Search Tool—provides local sequence alignment akin to BLAST over on-chain data. GLAST indexes on-chain metadata with Whoosh and performs local alignment with Parasail, exposing REST endpoints for text queries (titles, sources, authors) and sequence similarity across millions of on-chain entries. This hybrid model—on-chain storage (MOLNFT) with local indexing (Whoosh) and alignment (Parasail)—delivers rapid queries while the underlying corpus remains decentralized and independently verifiable.

GLAST can run beside an institutional node and indexer. A laboratory or public institution can reconstruct, index, search, and align the replicated corpus through infrastructure it operates, keeping routine query traffic, caching, and derived indexes within its own operational perimeter. The benefit is an independently controlled and verifiable retrieval path; performance remains deployment-specific.

Local-node reconstruction. A July 2026 single-host engineering run reconstructed eight selected MOLNFT v2 structures directly from contract state through a co-located node, confirming the functional local-retrieval path. Methods, recorded timings, limitations, and replication requirements are reported in appendix C; the observation is not used here as comparative performance evidence.

4.5 From molecular objects to working scientific infrastructure

Two tests distinguish scientific infrastructure from a token wrapper. *First,* the underlying object must be real and independently recoverable: complete MOLNFT v2 structures are reconstructable from contract state today, with a checksum-complete corpus release as the next validation step. *Second,* the object must have a genuine consumer: GLAST supplies practical search and alignment over on-chain records, while GL1F supplies a live deterministic model runtime. Together MOLNFT, GLAST, and GL1F already form usable scientific infrastructure. The next integration milestone joins a canonical structure to an independently validated, parity-tested protein-stability result (section 5.9). The chess deployment establishes EVM neural-inference feasibility; scientific validation remains model- and domain-specific.

MOLNFT therefore moves beyond a pointer wrapper through consensus-replicated scientific bytes, working reconstruction, and live analytical consumers. The cross-protocol scientific vertical will turn those capabilities into a reusable end-to-end validation pattern.

5 Protocol stack II — GL1F: on-chain machine learning

5.1 The case for structured-data ML on-chain

Public discourse around AI fixates on large language models, whose parameter counts (10^{10}–10^{12}) and stochastic decoding make them poor candidates for byte-identical on-chain inference. By contrast, gradient-boosted decision trees (GBDT)—XGBoost [25], LightGBM, CatBoost—are the workhorse of structured-data prediction across biomedicine, finance, genomics, and engineering [26]. Their inference is fundamentally a sequence of integer and float comparisons and additions over a forest of trees: small in state, deterministic across implementations when carefully specified, and amenable to verifiable execution. GL1F (GenesisL1 Forest) [39] turns this observation into a full protocol: a browser-native model studio and marketplace that compiles the workflow *train* → *tokenize* → *infer* into a fully verifiable chain of custody.

5.2 System overview and model lifecycle

GL1F consists of three layers: a browser application (dataset ingestion, training in a Web Worker, visualization, packaging, and deployment transactions); a suite of smart contracts (`ModelStore`, `ModelRegistry`, `ForestRuntime`, `ModelNFT`, `ModelMarketplace`); and the GenesisL1 chain itself (EVM, EVM chain ID 29, `0x1d`). The lifecycle of a model (fig. 3) is:

1. **Train** a fixed-depth GBDT on a dataset entirely in the browser—no Python environments, GPU clusters, or vendor cloud accounts required.

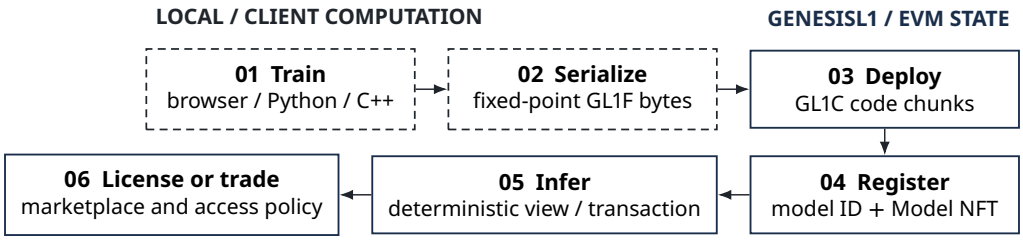

Figure 3. GL1F model lifecycle. Dashed outlines identify local or client-side work; solid outlines identify model state and operations committed to GenesisL1. Training is off-chain, while the serialization, registered identity, and deterministic inference path are publicly inspectable.

2. **Serialize** the model, quantizing tree parameters into fixed-point integers, into the compact GL1F binary format.
3. **Store bytes** by splitting the serialization into chunks; each chunk is deployed as a code-as-data contract.
4. **Publish** a pointer-table contract referencing the chunks; the registry mints an ERC-721 Model NFT with metadata and binds it to a content-addressed model ID.
5. **Infer** via read-only view calls or paid transactions, depending on the model's pricing mode.
6. **Trade** the NFT: ownership governs fee-free inference and access-key control.

5.3 The three-trainer architecture

To span a range of scales, GL1F ships three trainers that target a common model schema:

- a **browser trainer** (JavaScript/WebAssembly) for zero-setup training of small-to-medium tabular datasets;
- a **Python trainer** for standard laboratory pipelines and headless training of larger corpora; and
- a **C++ trainer** for maximum throughput on large-scale training (millions of rows, high tree counts) with strict reproducibility guarantees.

Cross-trainer byte parity is a release-specific conformance property, not a consequence of the shared schema. It requires canonical input parsing and ordering, matched builds and parameters, identical tie-breaking and arithmetic, and published fixtures. The present repositories include parity fixtures; every release should record the exact trainer commits and rerun them before claiming bit-for-bit equality.

5.4 Fixed-depth trees and the GL1F binary format

Gradient boosting constructs an additive model

$$F_M(\boldsymbol{x}) = F_0(\boldsymbol{x}) + \sum_{m=1}^{M} \nu\, f_m(\boldsymbol{x}), \tag{2}$$

where each weak learner f_m is a fixed-depth decision tree, ν is the learning rate, and each new tree is fit to the negative gradient of a loss function [26]. GL1F represents every tree as a *complete* binary tree of fixed depth d: an internal node stores a feature index f and a threshold τ; a leaf stores a value. Fixed depth is a deliberate design choice that makes on-chain decoding cheap: the number of internal nodes is $2^d - 1$ and the number of leaves is 2^d, so array indexing is pure arithmetic (no pointers), and byte offsets become compile-time-like expressions amenable to efficient `EXTCODECOPY`-based reads. "Missing" splits are encoded as forced nodes (τ = `INT32_MAX`), keeping inference constant-time per tree.

The serialization comes in two versions. **GL1F v1** encodes scalar-output models (regression, binary classification) with a 24-byte header followed by M trees. **GL1F v2** encodes vector-output models (multiclass, multilabel) with an explicit class count and per-class base logits, storing trees class-major (all trees for class 0, then class 1, and so on). For depth d, the per-tree byte cost is

$$\text{bytes/tree} = \underbrace{(2^d - 1) \cdot 8}_{\text{internal: u16 feat + i32 thr + u16 pad}} + \underbrace{2^d \cdot 4}_{\text{leaves: i32 value}}. \tag{3}$$

> **Compact models, explicit resource costs**
>
> The formula makes model size predictable before deployment. A scalar model with the suite's default 250 trees at depth 4 occupies 46,024 core bytes; 250 trees at depth 8 occupy 766,024 core bytes, before NFT metadata and pointer-table overhead. At height 13,315,110, the canonical registry contained 12 active Model NFTs: six core serializations were below 1,000,000 bytes, the mean was 2,598,777 bytes, and the maximum was 9,523,096 bytes. The authoring suite applies a 15,000,000-byte client guardrail and reduces tree count or depth when an estimate exceeds it. This is a creation-suite limit rather than a consensus maximum. The native gas market makes the publisher internalize more cost as model state grows, creating a natural on-chain path for compact structured-data models and an incentive for efficient representations.

Why fixed-point. The EVM has no floating-point arithmetic. GL1F therefore quantizes all values into `int32` fixed-point: an input feature x_i is packed as $\lfloor x_i \cdot \texttt{scaleQ} \rceil$, thresholds and leaf values are stored in the same scale, and the runtime accumulates into an `int256` to avoid overflow across many trees. The studio selects the quantization scale to maximize precision without overflow,

$$\texttt{scaleQ} = \min\left(10^6,\ \left\lfloor \frac{2{,}147{,}480{,}000}{\max_i |x_i|} \right\rfloor,\ \left\lfloor \frac{2{,}147{,}480{,}000}{\max |y|} \right\rfloor\right) \tag{4}$$

(omitting the y bound for classification), keeping quantized magnitudes comfortably below the signed-32-bit ceiling $2^{31} - 1 = 2{,}147{,}483{,}647$.

5.5 Code-as-data storage: GL1C chunks

GL1F stores model bytes on-chain by deploying contracts whose *runtime bytecode* begins with the four-byte `GL1C` magic and continues with the data payload—a variant of the well-known store-as-code (SSTORE2) pattern [21]. EIP-170 caps runtime code at 24,576 bytes [13], so each chunk carries at most 24,572 data bytes (reserving four for the `GL1C` magic, `0x474c3143`). To address models larger than one chunk, GL1F stores a *pointer table*—itself a `GL1C` contract whose payload is a sequence of 32-byte words each holding a chunk address—and the runtime reads the table and then the model bytes via `EXTCODECOPY`, avoiding the far costlier `SLOAD`. Deploying an N-chunk model costs $N + 2$ transactions: N chunk writes, one pointer-table write, and one `registerModel` call that mints the NFT and records runtime parameters.

```
uint32 public constant MAGIC = 0x474c3143; // ”GL1C” (GenesisL1 Chunk)
event ChunkWritten(address indexed pointer, uint256 size);

function write(bytes calldata data) external returns (address pointer) {
   require(data.length <= 24_572, ”CHUNK_TOO_LARGE”); // 24576 - 4 magic bytes
   // ... init-code writes (MAGIC || data) as runtime bytecode via CREATE ...
   emit ChunkWritten(pointer, data.length);
}
```

Listing 1. `ModelStore.write` deploys a minimal pointer contract whose runtime code is exactly `GL1C || DATA`, respecting the EIP-170 limit (abridged).

5.6 Deterministic on-chain inference

Inference for a scalar model traverses each tree from root to leaf using comparisons $x_f > \tau$, then adds the leaf value to a wide accumulator:

```
acc = baseQ
for t in 0 .. nTrees-1:
   idx = 0
   for lvl in 0 .. depth-1:
      (f, thrQ) = node(t, idx)
      idx = (idx*2 + 2) if featuresQ[f] > thrQ else (idx*2 + 1)
   acc += leaf(t, idx - (2^depth - 1))
return acc
```

Listing 2. Reference pseudocode for scalar (v1) inference. Because EVM execution is deterministic, every node reaches an identical result to the last bit.

For a scalar M-tree model of depth d, inference costs $O(M \cdot d)$ comparisons plus $O(M \cdot d)$ byte reads. The scientific payoff is direct: a peer reviewer evaluating a paper that cites a GL1F Model NFT can independently run the model on *any* input vector, on any chain node, without trusting the authors' inference server. A research decision-support model can publish a fixed serialization and let reviewers test whether recorded outputs match that version. Clinical use would additionally require external validation, regulatory analysis, monitored deployment, and human oversight; determinism alone is not clinical validity.

5.7 Fee-safe view inference under a spoofable RPC

A subtle but important attack surface arises from the RPC layer. The `eth_call` method accepts an optional `from` field [15], so an off-chain caller can *simulate* a read "as if" from an arbitrary address. Consequently, a paid-required model must not gate free view inference on `msg.sender` alone. GL1F resolves this with three pricing modes—(0) free, (1) tips, (2) paid-required—and two authorization mechanisms for mode 2:

1. a **paid transaction path** (`predictTx`) that enforces `msg.value` ≥ fee (only the current NFT owner may bypass it); and
2. an **EIP-712 signature-gated view path** (`predictOwnerView`, `predictAccessView`) that proves authorization—owner or a subscribed access key—without relying on the spoofable call sender [14].

Access keys are ordinary Ethereum keypairs whose addresses are registered on-chain with a block-height expiry; model owners publish paid access plans, and buyers extend a key's expiry by paying on-chain. Typed-data signatures use EIP-712 domain separation (chain ID + verifying contract), a short deadline to bound the replay window, and a hash of the packed features (rather than the raw bytes) to keep the signature payload small—so a signature cannot be replayed on another chain or another runtime deployment.

> **Public-weight limit.** GL1F model bytes and runtime code are readable from public chain state. Signature gating prevents `eth_call` sender spoofing against the canonical interface; it cannot stop a third party from copying the weights and running or redeploying the model. Paid modes can meter canonical on-chain execution, settlement, subscriptions, or licensed service paths. They do not create cryptographic exclusivity over public weights.

5.8 Discovery, licensing, and interpretability

GL1F maintains a fully on-chain, gas-frugal *title-word inverted index*: the registry maps Keccak256(word) hashes to token IDs and answers AND-queries by intersecting membership sets, enabling discovery with no off-chain indexing infrastructure. Every deployment must explicitly accept the registry's active Terms-of-Service version and license identifier (default: Creative Commons Attribution-ShareAlike 4.0), and the mint transaction records the accepted IDs on-chain. Before minting, the studio offers a preview gate that decodes the model locally and reports two complementary interpretability signals: *split-usage counts* (how often each feature is used in learned, non-forced nodes) and *permutation importance* on a sampled budget (how much the test metric degrades when a feature is permuted). Training itself is deterministic given (dataset, columns, seed, hyperparameters), using seeded `xorshift32` pseudorandomness, histogram or quantile split search, $\lceil \sqrt{n_{\text{feat}}} \rceil$ feature subsampling per split, plateau and piecewise learning-rate schedules, early stopping with optional train+val refit, and an optional in-browser heuristic hyperparameter search.

5.9 Scientific utility now and the domain-complete capstone

GL1F already lets a research group train locally, publish a canonical fixed-point serialization, mint its model identity, and expose deterministic inference that another party can re-run from public state. That is a usable model-publication and execution layer for structured data. Scientific validity remains model-specific: dataset design, feature extraction, experimental measurement, external validation, and clinical interpretation are not supplied by the runtime.

The first domain-complete GenesisL1 example is being built as a protein-stability ($\Delta\Delta G$) predictor that consumes a MOLNFT structure and produces a prediction whose lineage binds the exact input, feature schema, model bytes, weights, quantization, and output. It is not claimed as deployed or scientifically validated in this edition. Completion requires homology-aware splits, accepted external baselines, uncertainty reporting, checksum-verified MOLNFT inputs, local/on-chain parity tests, immutable model identity, and independent reproduction.

That capstone matters because it converts adjacent live components into a scientific dependency graph: structures point to models that consumed them, models point to their training and validation evidence, and predictions point to both. The neural-policy example in section 7 establishes that nontrivial deterministic EVM inference is feasible; it does not replace the biological validation or cross-protocol integration required here.

6 Long-term flagship research direction — GLAST structural alignment

The deployed **GLAST sequence-search service** described in section 4.4 is an operator-local Whoosh metadata index plus Parasail sequence alignment over reconstructed MOLNFT records. The research program described here is distinct: **GLAST-RIGID-1** is the proposed first conformance profile for deterministic, model-guided pairwise three-dimensional protein-structure alignment. It is positioned as a long-term flagship scientific application because it would join canonical molecular objects, learned search decisions, exact geometry, versioned execution, and a structured result in one independently inspectable dependency graph. It is not deployed or scientifically validated in this edition.

> **Learned search; canonical geometry**
>
> GLAST-Forest uses pinned GL1F models to decide *where a geometric algorithm should search*; it does not decide *what geometry is true*. Learned models may rank anchors, transformation hypotheses, correspondence edges, and refinement paths. Canonical rules must still construct correspondences, fit structures, resolve ties, and compute the reported score. Learned prioritization can allocate a search budget without becoming the scientific authority.

Proposed processing profile. The bounded first profile begins with canonical mmCIF/PDB parsing, an explicitly versioned structural representation, and quantized coordinates. It constructs local frames and rotation-invariant descriptors, proposes compatible anchor pairs, and derives candidate rigid transformations. One or more GLAST-Forest models then rank anchors and hypotheses. The selected candidates pass to sparse correspondence construction, rigid fitting, refinement, and a canonical length-normalized score derived from established structural-scoring principles [27]. The target sequence is

parse → canonicalize → local frames → descriptors → anchors → GL1F ranking → transformation voting
→ sparse correspondence → rigid fit → canonical score and result.

Determinism must be specified rather than assumed. Parser behavior, alternate locations, missing atoms, chain ordering, coordinate quantization, feature schemas, integer overflow, tolerance rules, correspondence ordering, tie-breaking, and result serialization all require conformance vectors. A floating-point SVD or hardware-dependent library does not become bit-exact merely because its search controller is deterministic; the conformance profile must either specify a fixed-point fit or bound and test an equivalently deterministic method.

GL1F as the search controller. GLAST-Forest models are trained off-chain and exported into GL1F's canonical fixed-point serialization. Separate forests may rank residue anchors, candidate transformations, or refinement actions, each under a versioned feature schema. Given identical feature bytes, model bytes, quantization rules, and runtime version, the bounded ranking decisions are intended to agree locally and in the EVM. The full combinatorial alignment need not execute inside consensus: a high-throughput engine may run beside a node, while its exact inputs, controller identity, parameters, selected path, and output are committed and independently replayed. This preserves practical computation without substituting a private model API for the controller.

A canonical result manifest should bind the MOLNFT input commitments; parser, representation, algorithm, and feature versions; GL1F model ID and byte hash; search-budget profile; correspondence; rigid transform; score; result bytes; and optional trace. A laboratory can then reconstruct the structures through its own node, reassemble the controller, repeat the alignment, and compare the serialized output. Matching output demonstrates computational repeatability, not that the controller is biologically superior.

Evidence required before performance claims. GLAST-RIGID-1 and GLAST-Forest remain in development. A publication-grade release requires public code, frozen models, leakage-resistant training and held-out sets, exact dataset partitions, hardware and compiler disclosure, conformance vectors, ablations separating learned prioritization from canonical geometry, and controlled comparisons with established structural-search and alignment systems such as Foldseek and GTalign [28, 29]. Alignment quality, score, RMSD, aligned length, coverage, sensitivity, runtime, memory, candidate budget, local/EVM parity, and independent reproduction should be reported together.

Until then, this whitepaper makes no claim of speed, accuracy, production readiness, or scientific superiority.

> **Evidence progression**
>
> Protein-stability workflow → independent reproduction → GLAST-RIGID-1 conformance release → verifiable scientific-search infrastructure → broader computational biology.

7 Protocol stack III — neural inference inside the EVM

7.1 An existence proof against the conventional wisdom

Running neural networks inside an EVM smart contract has long been treated as impractical: dense arithmetic is costly, floating point is absent, memory is constrained, and determinism is mandatory. GenesisL1 provides an existence proof that these barriers are not fundamental [41]. The demonstration is a complete on-chain inference system for chess based on a compact *policy network* deployed as Solidity code and executed by the EVM, producing deterministic move logits and top-K selections under a legal-move mask. The point is not that a client can play chess; it is an engineering proof of execution. The deployed proof-of-concept contract is `0x37D6518FbABd982e1908251A9cb4aD97b48BB989`; its weights remain owner-mutable, so every reproducibility claim must pin a block height and complete weight-state hash until a one-shot or permanently frozen deployment exists.

> **The engineering fact**
>
> A nontrivial learned function—neural-policy inference—can be executed *deterministically inside the EVM*, with outputs that are directly usable by other smart contracts and independently verifiable by any observer at a fixed contract state. Inference is reduced to integer/fixed-point primitives, the action space is made finite and maskable, and selection is implemented with bounded loops.

7.2 State, action, and mask algebra

Each chess position is encoded as a tuple $\boldsymbol{s} = (\texttt{bb}[0..11], \texttt{stm}, \texttt{castling}, \texttt{epFile})$, where $\texttt{bb}[i]$ is a 64-bit bitboard for the i-th piece plane (six piece types per color, twelve total), $\texttt{stm} \in \{0, 1\}$ is the side to move, $\texttt{castling} \in \{0, \dots, 15\}$ is a four-bit castling mask, and $\texttt{epFile} \in \{-1, 0, \dots, 7\}$ is the en-passant file. Squares use fixed indexing $\texttt{sqIdx}(f, r) = 8(r-1) + f$ with $\texttt{a1} = 0$ and $\texttt{h8} = 63$; the encoding is compact and supports fast bitwise manipulation.

A central engineering move transforms chess—a variable-branching game—into a fixed-index classification problem. The action set $\mathcal{A}$ has size $4{,}672 = 64 \times 73$, with

$$\begin{aligned} &\texttt{actionIdx} = 73 \cdot \texttt{fromSqIdx} + \texttt{planeIdx}, \\ &\texttt{fromSqIdx} \in \{0, \dots, 63\}, \qquad \texttt{planeIdx} \in \{0, \dots, 72\}. \end{aligned} \tag{5}$$

where the 73 planes encode 56 sliding moves (8 directions × 7 distances), 8 knight moves, and 9 underpromotions (3 promoted pieces × 3 directions). Legality is represented as a bitmask $\boldsymbol{m} \in \{0,1\}^{4672}$ packed into *584 bytes*; legality of action i is tested by $m_i = (\texttt{maskBytes}[i/8] \gg (i \bmod 8)) \,\&\, 1$, using only indexing, shifts, and bitwise AND. The mask is the bridge between exact chess rules and approximate neural priors: the contract never implements full chess legality, the model never receives legality as an input feature, and correctness is enforced by mask algebra at selection time.

7.3 Quantized inference and on-chain selection

Practical on-chain inference requires integer or fixed-point computation. A quantized affine layer is

$$y_j = \left\lfloor \frac{1}{S}\Big(\sum_i w_{ji}\, x_i + b_j\Big) \right\rfloor, \tag{6}$$

with integer weights w_{ji} and biases b_j, an explicit scaling constant S, and wide (32- or 64-bit) accumulators to prevent overflow; nonlinearities are inexpensive piecewise-linear functions. The contract exposes three deterministic primitives—`weightsReady()`, `inferBest(state, legalMask)`, and `inferTopK(state, legalMask, K)`—acting as an on-chain *policy oracle*. At a specified block and weight hash, outputs are a pure function of the supplied state and mask and may be re-run by any party. A full sort over 4672 elements is expensive; instead a bounded streaming top-K selection runs in $O(|\mathcal{A}| \cdot K)$ for small K (e.g. $K \le 32$), skipping illegal actions via the mask.

7.4 Reported policy accuracy and execution feasibility

The network is trained on approximately 2×10^7 positions drawn from games rated 2200+ Elo, with a cross-entropy objective over the full 4672-dimensional action space and *without* providing the legal mask as an input. The component report states approximately **25% validation top-10 accuracy** for placing the observed expert move among the top ten logits [41]. That metric suggests learned regularities correlated with expert play; it does not by itself demonstrate legal-move understanding, playing strength, or independent validation. Legality is enforced externally by the supplied mask. The practicality of the bounded inference and selection loops is chain-dependent: GenesisL1's EVM execution context is configured to be *compute-friendly*, enabling substantially larger view-call compute budgets than typical mainnet settings, in which an equivalently sized on-chain neural evaluation plus top-K selection would be prohibitively expensive or fail to fit within practical call constraints. This distinction is itself part of the contribution: it identifies a class of EVM environments in which on-chain neural inference transitions from "theoretically definable" to "operationally feasible."

7.5 Beyond chess: a template for verifiable scientific ML

The chess network is a deliberately hard *feasibility demonstration*—it demands combinatorial structure, legality constraints, and tactical sharpness—and it establishes one engineering fact: a nontrivial learned function can execute deterministically inside the EVM. It does *not*, by itself, establish that any particular scientific model has been built, benchmarked, or deployed on-chain; those remain to be done per application. What the demonstration does is identify a *class* of scientific tasks whose structure is compatible with the same techniques: tasks that reduce to compact classification

or regression functions amenable to aggressive quantization—sequence-motif classification and variant-effect scoring in genomics; molecular-property prediction from descriptors in chemistry; calibration functions and anomaly detectors in physics; and event filtering or prioritization in astronomy. For each, an on-chain model *could* provide a versioned scoring function whose contract, block, and weight hash make changes detectable—a globally consistent computation with transparent scoring rules, verifiable eligibility decisions in decentralized data markets, and auditability of pipelines where a model output triggers a protocol event. These are candidate applications enabled by the demonstrated primitive, not completed results. Even "small chat-like" language models fit the pattern when scope is constrained: a small on-chain intent classifier could act as a *deterministic query router*, `prompt` $\mapsto$ `(datasetId, tableId, columnId, queryTemplateId)`, mapping natural-language queries to canonical structured decisions over datasets committed by hash. When inference becomes a public, deterministic primitive, machine learning becomes a component of *protocol design* rather than an external service.

8 Protocol stack IV — CIPNFT: confidential IP on-chain

8.1 Tokenizing intellectual property, not just public data

Not all science is public. Pre-publication manuscripts, proprietary assay results, in-progress drug-discovery records, unfiled provisional patents, and confidential trade-secret know-how require the *opposite* of MOLNFT's full transparency: a way to commit to their content publicly while disclosing plaintext only to selected recipients. CIPNFT—Cryptographic Information Protocol NFT, also rendered Cipher NFT [40]—addresses part of this problem. It stores ciphertext fully on-chain and maintains an updatable *key envelope* that can be re-wrapped to a recipient. A current recipient can decrypt if the envelope and local keys are valid; previous recipients may retain the DEK or plaintext, so ownership changes do not revoke prior knowledge. In this sense CIPNFT records and transfers a *cryptographic capability*—access to decrypt a specific encrypted knowledge capsule—rather than attempting to tokenize information itself (canonical contract `0x05e4F1929947bE93a853F2de155fc1d6137f8446`).

> **The core idea**
>
> Store only ciphertext and integrity commitments on-chain, while delivering decryption capability through recipient-bound envelopes. Confidentiality before disclosure depends on established cryptography and endpoint security rather than a server policy; confidentiality after a recipient learns the DEK or plaintext cannot be revoked cryptographically.

CIPNFT operates at the contract layer and governs encrypted *assets*; it is complemented by the chain-level recipient-key (`e2ee`) module introduced in section 2.2. In both cases encryption and decryption occur in client software, and compromised clients can expose plaintext before it reaches the ledger.

8.2 The cryptographic capsule

A CIPNFT token is a cryptographic capsule

$$\mathcal{T} = (C,\ E_{\text{owner}},\ W_{\text{view}},\ h), \tag{7}$$

where C is the on-chain ciphertext of plaintext M under a data-encryption key DEK, E_{owner} is an owner-only envelope containing DEK, W_{view} is an optional view-key wrap of DEK, and $h = \text{Keccak256}(\text{DEK})$ is a commitment to the key. The construction layers established primitives:

Payload encryption (AEAD).
A 32-byte random DEK encrypts M under XChaCha20-Poly1305 [31, 32], an authenticated stream cipher with a 24-byte nonce and 16-byte tag, suitable for arbitrarily large payloads. Encryption binds associated data AAD = Encode(`chainId`, `contractAddress`), so

$$C \leftarrow \texttt{nonce} \,\|\, \text{XChaCha20-Poly1305.Enc}_{\text{DEK}}(M;\ \text{AAD}). \tag{8}$$

The AAD domain-separates ciphertext: an adversary cannot copy ciphertext and claim it belongs to a different contract or chain context without detection.

Owner envelope, classic mode (0).
DEK is sealed to the owner's X25519 public key via the libsodium sealed-box construction (Curve25519 ECDH + XSalsa20-Poly1305), $E_{\text{owner}} \leftarrow \text{Seal}_{\text{pk}_{\text{owner}}}(\text{DEK})$. For a 32-byte message the sealed box adds 48 bytes of overhead, giving an **80-byte** envelope stored on-chain.

Owner envelope, post-quantum mode (1).
For long-term confidentiality, an ML-KEM-768 envelope (Module-Lattice Key Encapsulation Mechanism, formerly Kyber, standardized in NIST FIPS 203 [33, 34]) may be selected at mint time. Encapsulation to the owner's 1184-byte public key yields (`ct`, `ss`) with a 32-byte shared secret, and DEK is wrapped under `ss` with the same AEAD and AAD. With a 1088-byte ciphertext, $|E_{\text{owner}}| = 1088 + 24 + (32 + 16) = \mathbf{1160}$ bytes—larger, but conjectured secure against quantum adversaries.

View key (optional).
A human-shareable view key v derives a symmetric key $K_v = H(v)$ (BLAKE2b-256) that wraps DEK into a 72-byte W_{view}; anyone who learns v can decrypt, enabling controlled disclosure to reviewers or due-diligence counterparties without transferring ownership.

Commitment.
Because the chain cannot verify that a provided envelope decrypts to the correct DEK, clients validate a recovered key by checking $\text{Keccak256}(\text{DEK}) = h$ before attempting metadata decryption.

The reference contract enforces $|M| \le 64$ KiB, $|C| \le |M| + 40$, $|E_{\text{owner}}| \in \{80, 1160\}$, and $|W_{\text{view}}| \in \{0, 72\}$; a short plaintext title (bounded to 80 bytes and 5 words) and the Terms-of-Service version accepted at mint are stored alongside.

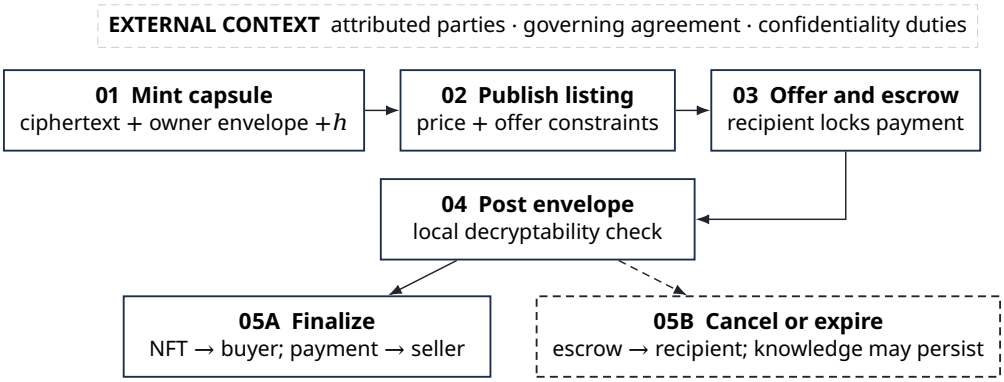

Figure 4. CIPNFT v1 transfer for attributable counterparties. The ledger records the offer, escrow, recipient-envelope posting, finalization or refund, and resulting token state. The dashed branch identifies the cancellation or expiry path addressed by the hardened successor.

8.3 CIPNFT v1: recipient-bound transfer among attributable counterparties

Standard ERC-721 permits arbitrary transfers via `transferFrom` and approvals. CIPNFT intentionally disables these: transferring ownership without updating E_{owner} would strand the ciphertext under the wrong envelope, breaking the intended recipient-envelope relationship. Transfer semantics are therefore redefined through an escrowed offer flow that updates the envelope within the same logical deal. The recipient validates decryptability locally before finalization, while the ledger records each attributed address, envelope, escrow action, and resulting token state.

> **Operating context of CIPNFT v1**
>
> CIPNFT v1 is intended principally for attributable, documented exchanges among laboratories, universities, biotechnology companies, inventors, funders, licensees, and other counterparties that ordinarily establish identity, authority, confidentiality duties, governing law, and commercial terms outside the chain. The protocol remains permissionless at the address layer. GenesisL1 adds a publicly verifiable chronology to the institutional and legal controls that govern the relationship.

This differs materially from an anonymous one-shot marketplace. When wallet ownership is reliably attributed, any observer can verify which address opened and funded an offer, which address posted the recipient envelope, whether the offer was finalized or refunded, and the resulting token and escrow state. That record can support contractual enforcement, compliance review, dispute resolution, and reputational accountability. Plaintext receipt and interpretation remain off-chain facts addressed by the surrounding agreement and evidence.

The canonical Alice-and-Bob flow: Bob calls `createOffer(tokenId, expiry)` and escrows the exact asking price; Alice decrypts her owner envelope to recover DEK, rewraps it to Bob's registered public key, and posts `deliverOffer(tokenId,buyer,newOwnerEncDEK)`—*no payment is released*; Bob decrypts the delivered envelope, checks Keccak256(DEK) $= h$, and confirms that C decrypts to M; only then does Bob call

`finalizeOffer(tokenId)`, at which point the contract updates the on-chain owner envelope, transfers the NFT, and releases the escrow to Alice.

After Alice posts the recipient envelope, the deployed v1 contract still permits Bob to cancel or reach an expiry-refund path after locally recovering the DEK or plaintext. In an attributable bilateral relationship, the ledger's tamper-evident chronology can materially support contractual, compliance, dispute, and reputational enforcement. The capsule, integrity commitment, timestamped provenance, and recipient-bound disclosure remain independently usable when no sale occurs.

The hardened successor will remove or explicitly bound this refund asymmetry while preserving invalid-envelope protection, publish a migration path, and complete independent smart-contract and cryptographic audit. Audit scope, report, remediation status, and deployment identifiers are release gates for broader trust-minimized use. Anti-spam rules—a minimum price, bounded duration, and exact-price escrow—already improve operation and abuse resistance.

8.4 Post-quantum resistance and ledger survivability

Large-scale quantum computers threaten classical public-key cryptography: Shor's algorithm breaks elliptic-curve Diffie–Hellman in polynomial time, so any confidentiality resting on X25519 may be compromised retroactively by a "store-now-decrypt-later" adversary. Symmetric cryptography is more resilient—Grover's algorithm yields only a quadratic speedup, so 256-bit symmetric keys remain conservative. In CIPNFT the payload uses 256-bit AEAD (robust), so the *envelope* is the dominant long-term confidentiality risk; the experimental ML-KEM-768 mode addresses it with a standards-based KEM-DEM construction that has not received an independent implementation audit. GenesisL1 account signatures and wallet custody remain secp256k1-based and are not post-quantum. A hybrid envelope storing both classical and post-quantum wraps is a natural future direction for high-value, long-lived IP.

Because a capsule is portable as bytes, CIPNFT can outlive any particular chain. The tuple $\mathcal{T} = (C, E_{\text{owner}}, W_{\text{view}}, h, \texttt{encMode}, \texttt{title}, \texttt{tosVersion})$ is entirely on-chain data and does not inherently depend on GenesisL1's consensus rules; the confidentiality of M depends on the secrecy of DEK and the envelope, not on any chain's consensus algorithm. Two migration patterns follow: *re-mint after decryption* (clean domain separation under a new AAD, requiring plaintext access), and *capsule preservation with a continuity certificate* (a wallet signature σ over a message linking old and new token IDs plus Keccak256$(C \| h)$, forming a portable audit trail verifiable against the owner's on-chain address history).

8.5 Scientific use cases and assurance model

CIPNFT supports pre-publication manuscripts and figures with timestamped, encrypted commits useful for priority disputes; drug-discovery datasets shared under selective view-key disclosure; patent-pending specifications anchored as an existence proof without revealing claim language; and confidential clinical-trial protocols for DAO-coordinated multi-site studies. On GenesisL1 it also serves as a reusable confidentiality primitive for domain-specific assets: in a peptide-design studio, candidate peptides and their design rationales can be tokenized as encrypted

molecular NFTs (CIPMolNFTs) for selective disclosure to collaborators and escrowed transfer to commercialization partners.

The encrypted-capsule core and staged settlement path are separate. The core supplies ciphertext integrity, timestamped provenance, recipient-bound disclosure, and a portable evidentiary record with or without a sale. The v1 exchange is designed for attributable counterparties operating under explicit external terms. Legal IP rights, post-disclosure conduct, remedies, the refund path, historical persistence, and administrative parameters are specified in section 13.6. Broader trust-minimized exchange is gated on the hardened, migrated, and independently audited implementation.

A ledger timestamp and ciphertext commitment may contribute evidence that specified bytes existed under an address at a given ledger time. They do not by themselves prove authorship, inventorship, legal priority, ownership, validity, confidentiality compliance, or transfer of intellectual-property rights.

9 Scientific DAOs and community governance

9.1 Why DAOs are native to science

The norms of open science—peer review, the commons, public critique—map naturally onto on-chain DAO mechanics: weighted voting, transparent treasuries, public proposals, and on-chain execution where supported [35]. A scientific DAO is, in effect, a public, auditable instantiation of the social structures already implicit in a research collaboration: a steering committee (a multisig), a working budget (a treasury contract), a publication record (NFT mints), and a mechanism for ratifying decisions.

9.2 Federation around GenesisL1

GenesisL1 provides four composable governance primitives rather than prescribing a single DAO model.

- **Chain-level governance**—the `x/gov` module, in which all bonded `L1` holders may submit and vote on network-wide proposals (parameter changes, community-pool spending, software upgrades).
- **Contract-level DAOs**—domain-specific DAOs instantiated as Solidity contracts (Governor patterns, multisigs, quadratic-voting modules), each free to choose its voting token, membership criteria, and execution scope.
- **Federated assets**—a contract-level DAO can own MOLNFT collections, GL1F Model NFTs, CIPNFT IP, treasuries in `L1`, and even validator-operator NFTs.
- **Domain-specific instruments**—a contract-level DAO may create governance, membership, access, or contribution records under separately assessed technical and legal terms. Such instruments are application-layer choices, not properties or endorsements of the base network.

9.3 Example archetypes

▶ **The bioinformatics DAO.** A consortium of academic and independent bioinformaticians stewards a curated MOLNFT subset, maintains the canonical mirror, funds new mints from the community pool, and arbitrates schema upgrades.

▶ **The ML-research DAO.** GL1F Model NFTs targeting specific biomedical endpoints (protein thermostability, solubility, ADMET screening) are jointly trained and curated by a researcher DAO; inference fees flow into a contract that automatically rewards contributors.

▶ **The publication DAO.** A peer-review group anchors manuscript hashes on-chain at submission, decrypts review-only CIPNFT copies for participating reviewers, and produces a public, tamper-evident record of decisions.

▶ **The art and natural-history DAO.** GenesisL1's mandate extends beyond biomedicine: digitized natural-history collections, museum metadata, and scientific illustration may be tokenized under the same standards, with their own DAOs.

10 The L1 coin: resource, security, and public-goods coordination

10.1 Operational roles

L1 is the native infrastructure asset of GenesisL1. Its base-protocol functions are factual, observable, and inseparable from operation of the sovereign ledger:

- **Gas.** Consensus transactions and state writes are metered in L1; read-only RPC calls may be offered without an on-chain payment by the node operator.
- **Stake.** Validators and delegators bond L1 to obtain voting power and rewards while accepting protocol-defined slashing and unbonding risk.
- **Governance weight.** Eligible bonded voting power participates in proposals and parameter changes under the deployed `x/gov` rules.
- **Public-goods denomination.** The community pool and its on-chain disbursements are denominated in L1.

These functions do not make L1 a share in a company, a claim on a balance sheet, a dividend, or a contractual redemption right. The protocol provides no fiat, commodity, or stable-asset backing. Legal characterization depends on jurisdiction, distribution, communications, use, and other facts; this paper makes no legal classification and offers no investment recommendation.

10.2 Resource pricing, model competition, and settlement

Every consensus transaction requires L1 for gas, staking requires bonded L1, and application contracts may denominate service charges in L1. MOLNFT publication, GL1F deployment and transaction-executed inference, CIPNFT state changes, governance actions, and other scientific writes therefore use the native asset at the point where they consume consensus or persistent state. Read-only `eth_call`, local model execution, and local indexing can be served by an institution or node operator without creating a consensus transaction.

For GL1F, this resource discipline is concrete. Model bytes are deployed in code chunks, so larger forests require more state, more write transactions, and more gas.

Publishers can choose free, tip-supported, or paid canonical access modes, while every model remains addressable through a common registry and deterministic runtime. Compactness, reproducibility, published performance, interpretability, access terms, and execution cost are visible together. The gas market creates an admission price for scarce shared state and a transparent competitive surface; scientific evidence determines model quality.

This produces a measurable protocol-economic loop: scientific state writes and paid calls use L1; bonded L1 secures the resulting state; and governance directs the community pool toward validators, archives, audits, datasets, models, tooling, and replication. Human users and software agents invoke and settle against the same contracts. The native asset also preserves ledger independence: security, gas, resource policy, and public-goods allocation do not rely on a foreign issuer or host-chain monetary system.

The loop is evaluated through observable utilization—gas consumed, fee-paying addresses, contract calls, state added, bonded stake, community-pool activity, independently deployed models, and scientific application use. These measures describe protocol use and security rather than investment performance. Applications remain free to provide subsidized access, use read-only calls, or build additional settlement logic.

Ongoing issuance funds recurring consensus security and public infrastructure while fee revenue remains limited, and its dilution is measurable. Governance should evaluate continuation against validator security, fee revenue, state growth, archive capacity, independent use, and delivered public goods, reducing issuance when those outcomes cease to justify it.

10.3 Security exposure

Bonded L1 makes validator voting power and slashable exposure visible, but its market value is not a strict lower bound on attack cost. Validator-key compromise, bribery, correlated infrastructure, borrowed exposure, beneficial-owner concentration, and governance capture may all reduce effective security. The relevant controls are the one-third and two-thirds voting-power thresholds, together with validator diversity, key management, monitoring, evidence handling, state archival, and social recovery procedures. Current stake and concentration measurements are reported in section 11.6.

10.4 Community-pool funding

The community pool provides a protocol-native public-goods budget. Once a passed spend reaches a recipient address, custody and execution rest with that recipient. Proposals should therefore define milestones, recipient disclosures, reporting, audit rights, and follow-up review.

11 Tokenomics

11.1 Genesis supply and distribution: 21,000,000 L1, publicly claimed

GenesisL1 launched with an initial supply of 21,000,000 L1, an intentional homage to the upper-bound supply of Bitcoin [7]. This initial figure is a transparent *starting point*, not a permanent cap: L1 is an inflationary asset whose configurable monetary parameters are presently changed through on-chain governance (section 11.2). The full 21,000,000 was minted at genesis to a single bootstrap account and distributed to the public *after* chain launch, by on-chain transaction, through a voluntary one-to-one transition (section 11.4). Every figure in this section is reproducible from the network's own state and public repositories; verification endpoints are collected in appendix E. The distinguishing properties of the distribution, all independently verifiable on-chain, are:

- **Public claim, not a genesis airdrop of balances.** The canonical genesis file holds the full supply in one bootstrap account (`genesis1tf30dj…`); balances were then distributed by transaction as holders of the Ethereum origin token claimed native L1 one-to-one. That bootstrap account, which held 100% of supply at genesis, holds sub-1 L1 dust today on both chain segments—it distributed everything out and retained no allocation.
- **No backing by any party.** L1 is not collateralized by fiat, commodities, stablecoin reserves, any company's revenues, or any future promise. Its functional value is the demand for gas, for staking, and for participation in dApp economies—not a redemption right.
- **No token sale, no reserved or preferential founder or investor allocation.** There was no pre-sale, public sale, or private sale of the native coin, and no allocation to founders, venture investors, or insiders on terms unavailable to any participant. The origin token was an unpriced gift (section 11.4); all supply left unclaimed was socialized to the governance-controlled community pool.
- **Governance-parameterized issuance.** Under the current protocol, post-genesis issuance follows `x/mint` parameters that can be changed through governance. The proposal and transaction records are public (section 11.4, appendix E).

11.2 Inflation and the community pool

After genesis, new L1 enters circulation only via protocol inflation, parameterized by on-chain governance. At each block, L1 is minted by the `x/mint` module into the fee-collection and distribution flow. The distribution module applies a configurable community tax to distributable rewards and routes the remainder to validators and delegators, subject to validator commission. The inflation rate is governance-tunable within a target staking-ratio band: if the bonded fraction is below target, inflation rises to incentivize staking and consensus security; if above target, inflation falls. The target band, floor and ceiling rates, community-pool fraction, and goal-seek velocity are parameters that governance can update. A software upgrade could also change module behavior, so the governing binary and upgrade process are part of the monetary-policy trust model.

This tunability is exercised in practice, not merely in principle, and the on-chain record documents it precisely. The *community tax*—the fraction of distributable rewards routed to the community pool—was set to 2 % at the launch of chain segment `genesis_29-1` and was subsequently raised by governance to 20 %, a deliberately high allocation (most Cosmos chains use 2 %) reflecting an aggressive commitment to funding public goods. The inflation ceiling itself has been repeatedly adjusted by vote: an early proposal raised `InflationMax` to 42 % during the network's bootstrapping phase, and a later proposal ("Reduce Max Inflation to 10%") *lowered* the ceiling to 10 %. As of the reference state, inflation bounds are 5 %–10 % with observed inflation at approximately 10 %. The community has furthermore declared its own long-run monetary policy in a sequence of on-chain proposals, *rejecting* an "unlimited supply" declaration and a 1,000,000,000 soft cap, and *passing* a **100,000,000 (100M) soft-cap declaration**: a non-binding governance policy target to hold supply below 100,000,000 `L1` by later reducing issuance. It is not a protocol hard cap and requires future governance action; unchanged positive inflation would eventually exceed it. All figures here are reproducible from the endpoints in appendix E.

11.3 The community pool as a protocol-native public-goods budget

The community pool is a protocol-native public-goods budget, funded through the community tax on distributable rewards, direct contributions, and governance-directed transfers. Slashed stake is burned under the current implementation; it does not accrue to this pool. The pool is disbursed through governance spend actions, which are public, on-chain, and subject to a voting period during which any staker can vote, veto, or abstain. Acceptable spending targets have historically included, and the community may continue to fund: core protocol development, audits, and security bounties; grants to scientific DAOs; indexers, RPC nodes, IPFS pinning, and other public infrastructure that subsidizes free access; public datasets, MOLNFT extensions, and reference Model NFTs; and education, documentation, conferences, and hackathons.

11.4 The transition and the verified distribution record

The precursor to the native coin was a standard ERC-20 token (name `GENESIS`, symbol `L1`) deployed on Ethereum mainnet—an unmodified OpenZeppelin ERC-20 with a fixed constructor mint of 21,000,000 units and no owner, no post-construction mint function, no pause, and no upgrade mechanism. The contract's own source states its nature immutably on-chain: an "experimental utility token with no default value of any kind," allocated "in the form of a gift without intrinsic value" to broad communities of blockchain *users* and *builders*, whose sole function was to enable "a transition to a new Layer-1 blockchain." Holders converted to native `L1` at a fixed one-to-one ratio by sending the ERC-20 to the origin contract's own address, where—because the contract exposes no function to move its own balance—the tokens are permanently and irrecoverably stranded; the origin balance at that address remains a public record of the amount migrated. The transition was opt-in, symmetric (no bonus or insider rate), publicly announced, and open for approximately six months. Per-address transition records are published in the project repositories (appendix E).

The material distribution record after genesis is publicly queryable. At the reference state it contains **112 recorded proposals: 95 passed and 17 rejected**. Rejected

and contested votes demonstrate that proposals are not universally rubber-stamped; they do not, by themselves, prove independence of beneficial owners or absence of coordinated control. Representative passed proposals include:

- **#2** — Allocation of 2,500,000 L1 to *genesis.gift*, an egalitarian airdrop to Cosmos ATOM delegators (snapshot published in the `genesis.gift` repository).
- **#42** / **#47** — Gift of 1,050,000 L1 (5% of initial supply) to the Evmos community, explicitly acknowledging GenesisL1 as a source-code fork of Evmos v0.3 (an earlier such proposal, #32, was *rejected* before a revised version passed—governance at work).
- **#3, #5, #6, #48** — OSMO/L1 Osmosis liquidity provisioning and rewards.
- **#85** — The 100,000,000 soft-cap tokenomics declaration (with #81 "unlimited" and #84 "1 BLN" both *rejected*), later implemented by **#127**, "Reduce Max Inflation to 10%."
- **#89** — Recycling of 600,000 L1 to mint the Protein Data Bank on-chain as the Atom-NFT collection—a direct instantiation of the web3-laboratory thesis.
- **#79** — 250,000 L1 to a contributor for an open-source on-chain pairwise-sequence-alignment dApp; **#75** establishing a community tradition of L1-for-IPFS-pinning spends.
- **#121–#125** — The v1.6.2 protocol upgrade and delivery bounties.
- Numerous 50,000-L1 validator-onboarding gifts, many *rejected*, funding decentralization of the validator set.

The bootstrap account that held 100% of supply at genesis holds sub-1 L1 dust today, confirming that it distributed the supply outward and retained no allocation. This is directly and currently verifiable (appendix E).

11.5 Distribution overview

Table 8. GenesisL1 monetary parameters (verified on-chain).

Parameter	Specification
Genesis supply (minted)	21,000,000 L1 (to a single bootstrap account)
Supply model	Inflationary and governance-parameterized; non-binding 100,000,000 policy target
Genesis distribution	Public 1:1 transition by transaction; unclaimed supply socialized to community pool
Token sale / reserved founder or investor allocation	None; full genesis mint temporarily held by disclosed bootstrap distributor
External backing / collateral	None
Issuance authority	On-chain governance (`x/gov`); no external mint
Inflation at reference state	10 %; configured range 5 %–10 %
Community tax	20 % on `genesis_29-2`
Community-pool disbursement	Exclusively via passed `CommunityPoolSpendProposal`
Slashing destination	Burned under the current staking implementation
Fee market	L1 gas; parameters governance-tunable

Total supply is inflationary and reproducible from the live endpoints (appendix E). Supply has grown from the 21,000,000 genesis figure via governance-parameterized inflation: to ≈28,784,317 L1 by mid-2023 (height 6,751,399), and to 46,475,691.0104 L1 at reference height 13,313,640. The community pool held 5,112,381.2157 L1 (approximately 11.0 % of supply), and bonded stake was 23,982,866.5695 L1 (approximately 51.6 %). The genesis bootstrap account—which held 100% of supply at launch—held approximately 0.73 L1 dust. All values are height-stamped rather than fixed.

11.6 Decentralization snapshot

Decentralization is a measurable condition, not a cover-page adjective. At height 13,313,640, the active bonded set contained 20 validator monikers and the protocol parameter allowed up to 50. Address-level data do not prove that every operator has an independent beneficial owner, cloud provider, or key-management stack. Concentration is therefore reported directly (table 9).

Table 9. Consensus and governance snapshot at height 13,313,640 (2026-07-19 13:07:53 UTC).

Measure	Observed value
Active bonded validators	20 (protocol maximum: 50)
Bonded voting power	23,982,866.5695 L1; 51.6 % of total supply
Largest validator	13.09 % of bonded voting power
Top 3 / top 5 validators	35.62 % / 51.07 % cumulative voting power
Threshold concentration	3 validators exceed one-third; 8 exceed two-thirds if coordinated
Governance record	112 proposals: 95 passed, 17 rejected (15.18 % rejected)

The top-three share makes delegation diversity a current liveness and governance priority. The figures are compatible with open validator candidacy but not with a claim of maximal decentralization. Future releases should add beneficial- owner attestations, infrastructure and geographic diversity, governance turnout, and named multisig/upgrade-key disclosures.

11.7 Why this design

The design trades a fixed hard cap for configurable security rewards and public-goods funding. Its strengths are an inspectable genesis custody path, absence of a sale or reserved founder/investor allocation, explicit module parameters, and a public proposal record. Its risks are equally concrete: inflation and policy uncertainty, stake concentration, governance capture, recipient custody after community-pool transfers, and dependence on validators adopting approved software. The relevant standard is not whether the design is rhetorically "fair," but whether current state and every change remain independently auditable.

12 Governance, on-chain

12.1 The proposal lifecycle

GenesisL1 inherits the Cosmos-SDK `x/gov` module. *Deposit period*: any address able to provide the required deposit may submit; the proposal becomes active for voting once the threshold is reached, with deposit disposition governed by the deployed module rules. *Voting period*: eligible bonded voting power (validators and their delegators) may cast Yes, No, NoWithVeto, or Abstain, with voting power proportional to bonded L1. *Tally*: a proposal passes when quorum is met, Yes exceeds the threshold of non-abstaining votes, and NoWithVeto stays below its veto threshold. *Execution*: passed proposals trigger their defined on-chain effects at the end of the voting period. A software-upgrade proposal schedules an upgrade height, but validators must still obtain and run the intended binary; a community-pool transfer cannot guarantee a recipient's subsequent off-chain or cross-chain execution.

12.2 Categories of proposals

- **Parameter-change proposals** adjust any governance-controlled module parameter (inflation floor and ceiling, target staking ratio, block-reward distribution, unbonding time, slashing severity, gas constants).
- **Software-upgrade proposals** ratify a new protocol binary at a target height.
- **Community-pool-spend proposals** transfer `L1` from the pool to a named recipient for stated purposes.
- **Text proposals** provide non-binding social signaling (ratification of standards, statements of intent).

12.3 Validator obligations and delegator powers

Validators perform the work of consensus and may charge commission on staking rewards. Delegators can vote independently of their validator and can redelegate subject to protocol restrictions, slashing exposure, and redelegation limits. These powers reduce, but do not eliminate, risks of validator or beneficial-owner concentration.

13 Security considerations

13.1 Consensus-level attacks

Consensus security depends on voting-power distribution, validator and delegator incentives, key security, network connectivity, evidence handling, and correct software. Relevant threats include more-than-one-third liveness attacks, two-thirds safety failures, validator-key compromise, long-range or stale-state attacks against weak clients, correlated cloud/provider failure, state-sync checkpoint manipulation, denial of service, and malicious or compromised upgrade binaries. Bonded value is one indicator; it does not subsume these operational risks. Current concentration is reported in section 11.6.

13.2 Inherited administrative and censorship surface

The v1.6.2 execution module retains a Cronos-derived administrator parameter and encrypted blocklist mechanism capable, if activated, of filtering transaction signers, destinations, and EIP-7702 authorization addresses. At reference height 13,313,640, both the `CronosAdmin` parameter and stored blocklist were empty. `MsgStoreBlockList` accepts submissions only from the configured administrator; with that parameter empty, no ordinary account can activate filtering. The governance module can, however, authorize a parameter update that designates an administrator, after which that address can submit an encrypted list applied during transaction and proposal validation.

The surface is therefore inactive, not removed. Until the code path is deleted and covered by regression tests, GenesisL1 should be described as providing permissionless transaction submission and open validator candidacy with no active account-level filter at the stated reference height, rather than as unconditionally censorship-resistant.

13.3 Smart-contract-level risks

Contracts on GenesisL1 inherit the standard EVM threat surface: re-entrancy, integer over/underflow (mitigated in Solidity ≥ 0.8), oracle manipulation, access-control errors, and upgrade-pattern mistakes [38]. Official protocol contracts (MOLNFT, GL1F core, CIPNFT) are open-source but are not represented here as having completed comprehensive independent audits or formal bug-bounty programs. GL1F's EIP-712 path addresses sender spoofing at the canonical contract interface, while public weights remain copyable. CIPNFT's encrypted-capsule and provenance core is operational independently of settlement. Its v1 transfer lets a recipient check an envelope but retains a disclosed refund asymmetry; the intended context is attributable-counterparty exchange supported by external legal and institutional controls (section 8). The chain provides no implicit guarantee about official or third-party contracts; users must evaluate exact addresses, bytecode, administrators, and audit status.

13.4 Cryptographic agility

CIPNFT includes an experimental, standards-based ML-KEM-768 envelope mode. The deployed contract supports two enumerated modes; adding algorithms requires a new version or deployment, so broad algorithm agility should not be assumed. The implementation has not received an independent cryptographic audit, and chain accounts remain secp256k1-based. Any migration should preserve ciphertext commitments, domain separation, owner authorization, and decryptability tests.

13.5 Data risks

GenesisL1's official tooling is designed so that the chain anchors only anonymized public-science artifacts and cryptographic commitments; every official dApp and its documentation prohibits the submission of PHI or GDPR-defined personal data, and the recommended architecture for sensitive data is off-chain storage in a compliant environment with only a non-identifying commitment where lawful and necessary. CIPNFT ciphertext should not be treated as a compliance safe harbor: encrypted personal data may remain regulated data. This is an operational control, not a protocol guarantee: because the chain is permissionless, it cannot technically prevent an arbitrary user from writing non-compliant or unlawful data (including personal data, or its ciphertext) in a raw transaction. This creates a real and unavoidable risk class—*immutable bad data*: anything committed to a public ledger is permanent, cannot be redacted from history, and may include data that is later deemed unlawful in some jurisdiction. Mitigations are architectural and social (official front-ends that neither solicit nor write regulated data, an `eraseTokenData` path that scrubs current storage slots though not historical chain data, and community governance over official tooling), and parties handling regulated data remain responsible for their own compliance. Users must further note that CIPNFT ciphertext is durable by design, and cryptographic confidentiality after disclosure cannot be revoked.

13.6 Protocol-specific threat models

The consensus- and contract-level risks above are shared with any EVM chain. Because GenesisL1's protocols carry scientific data, models, and encrypted IP, each also has a domain-specific threat surface. The tables below summarize the principal risks and the mitigations in place or planned; they are a working threat model, not an audit, and are maintained alongside the contracts.

Table 10. MOLNFT threat model (molecular data on-chain).

Risk	Mitigation / status
Corrupted or wrong source data at mint	Canonical citation and content hash recorded at mint; any divergence from the cited source is detectable, though correctness of the *original* deposit is out of scope.
Wrong PDB→token mapping	Deterministic identifier scheme; mapping is publicly auditable against the source repository.
Decompression bombs (gzip/BCIF)	On-chain chunk bounds do not bound the decompressed output; every consumer must enforce decompressed-size, recursion, memory, and CPU limits before parsing untrusted payloads.
State bloat from large corpora	Chunked parent–child storage; growth is metered by gas and governed by community funding of mints.
Contract-immutability bugs	Open-source contracts under iterative review; no upgrade key that could silently rewrite stored data.
Metadata drift	On-chain metadata is fixed at mint; off-chain indexes (GLAST) are advisory and rebuildable from chain state.

Table 11. GL1F threat model (on-chain gradient-boosted inference).

Risk	Mitigation / status
Malicious or low-quality models	Models are content-addressed and non-authoritative; discovery surfaces interpretability signals, and the marketplace exposes provenance rather than endorsing correctness.
Overfitted or misrepresented benchmark claims	Determinism lets any party re-run a model on held-out inputs; model-card and benchmark NFTs are a planned, on-chain, checkable record.
Paid-inference bypass via `eth_call` spoofing	EIP-712 signature-gated view paths bind authorization to a signature, not the spoofable call sender (section 5).
Public-weight copying	Inherent: model bytes are readable from state. Paid paths meter the canonical service or settlement, not exclusive access to weights or local inference.
Feature-scaling / quantization mismatch	Fixed-point scale recorded with the model; reproducibility checklist requires byte-parity tests between local and on-chain inference.
Model poisoning (training-time)	Out of protocol scope; mitigated socially by provenance and by consortia curating their own training data.

Table 12. CIPNFT threat model (confidential IP on-chain).

Risk	Mitigation / status
Client-side / browser compromise during encryption	Encryption is local; a compromised client is outside protocol control, mitigated by open-source front-ends and reproducible builds (recommended).
Key loss	Ownership and decryption capability are key-bound; loss is unrecoverable by design (no custodian), a deliberate trade for trustlessness — users must manage key backup.
Invalid recipient envelope in a transfer	Staged envelope posting lets the recipient validate decryptability locally before funds release. This protects the recipient but does not establish atomic fair exchange.
Decrypt-then-refund / seller loss	Residual v1 risk: the ledger records escrow, envelope posting, cancellation, and refund, but a recipient may retain the DEK or plaintext. Reliable address attribution makes the chronology useful for contractual and reputational enforcement; it cannot prove decryption or prevent retained knowledge. Use should remain within identified, contractually bound relationships until the hardened and independently audited release.
Counterparty attribution and legal enforceability	Identity is not protocol-enforced. Address-to-entity attribution, authority, governing law, IP-transfer terms, confidentiality obligations, and remedies must be established externally. The ledger supplements those controls with tamper-evident evidence; it does not replace them.
Prior-owner retention	Unavoidable after disclosure: a former recipient may retain the DEK or plaintext; ownership transfer is not cryptographic revocation.
Ciphertext permanence	Ciphertext is durable and cannot be erased from history; `eraseTokenData` clears current slots only.
Post-disclosure copying	Explicitly out of cryptographic scope—CIPNFT is not DRM; confidentiality after disclosure is a legal/governance matter.
ML-KEM implementation risk	Experimental standards-based mode pending independent review; account signatures are not post-quantum, and new envelope algorithms require a new version or deployment.
Administrative parameters	Terms, fees, and offer controls remain administrator-managed in the deployed contract; exact privileges and migration authority must be disclosed.
Phishing / front-end supply-chain attack	Mitigated by verifiable front-end builds and user verification of contract addresses; a general web3 risk, not specific to CIPNFT.

Table 13. Governance and economic threat model.

Risk	Mitigation / status
Low governance turnout	Quorum requirements; delegators vote independently of validators; the historical record shows contested, non-rubber-stamped voting (section 11.4).
Validator cartel / whale capture	Voting power is stake-weighted and transparent; delegators can redelegate without penalty; validator-decentralization grants are an ongoing community-pool priority.
Community-pool abuse	Every disbursement is a public, on-chain proposal subject to veto; the substantial historical rejection rate is direct evidence of a genuinely contested pool.
Vote-buying / bribe markets	A recognized open problem for all on-chain governance; mitigated by transparency and social accountability, not eliminated.
Economic attack on consensus	Voting-power thresholds and bonded exposure are observable, but bribery, key compromise, beneficial-owner concentration, and correlated infrastructure can reduce effective attack cost.
Dormant blocklist authority	The inherited execution module contains an encrypted blocklist mechanism. Administrator and list are empty at the reference state, but governance can activate the surface; permanent removal is a roadmap candidate.

13.7 Audit and assurance status

As of July 2026, no comprehensive independent security audit has been completed and published for the GenesisL1 node integration or the canonical MOLNFT, GL1F, and CIPNFT contract suites. Open-source code and internal development review are useful evidence, but not audit substitutes. Material limits include copyable GL1F weights, mutable neural-policy weights, and the residual refund asymmetry in CIPNFT v1.

For CIPNFT specifically, independent smart-contract and cryptographic audit is a release requirement for the hardened transfer implementation, not a status achieved by v1. Audit scope, report, remediation decisions, final deployment addresses, and migration procedure must be published before that implementation is represented as broadly production-ready. The same evidence standard applies across the core: commission independent review, publish findings and remediation commits, and establish a formal bug-bounty scope (section 14).

13.8 Legal and regulatory risk

GenesisL1 touches several areas of active and jurisdiction-dependent regulation, and this section flags—rather than resolves—that exposure; nothing here is legal advice or a legal conclusion. *Asset characterization:* the protocol assigns L1 gas, staking, and governance functions and supplies no dividend or contractual redemption right (section 10); whether a given token is a security, commodity, or other regulated instrument depends on the jurisdiction and the specific facts, and classification may differ across regimes. *Data protection:* as a permissionless chain, GenesisL1 cannot guarantee that no user ever writes regulated personal data, so data- protection compliance rests on the off-chain-plus-commitment architecture and on the parties handling such data (section 13). *Intellectual property:* tokenization of public scientific data preserves upstream

citations and asserts no ownership over public-domain facts, while CIPNFT concerns confidential IP whose lawful handling remains the user's responsibility. *Cross-border and sanctions:* a permissionless network is globally accessible, and participants remain responsible for compliance with the laws applicable to them. The project's posture is to engage qualified counsel where appropriate and to avoid categorical legal claims in public materials; readers should conduct their own regulatory diligence.

14 Roadmap

The live `v1.6.2` stack (section 2.2) opens the next phase: convert deployed scale into institutional-grade assurance, complete the first domain-validated scientific vertical, enable independent replication, and expand the scientific surface. Engineering assurance and scientific development proceed in parallel under public governance and published evidence.

> **Primary evidence milestone**
>
> An independently reproduced MOLNFT-to-GL1F result will connect GenesisL1's deployed data, model, execution, and provenance layers in one citable scientific workflow.

14.1 Phase I: institutional-grade assurance and capacity

Phase I equips the operating foundation for higher-value scientific and institutional use.

▶ **CIPNFT hardening and migration.** Replace the v1 bilateral transfer flow with a formally specified state machine that removes or tightly bounds the post-envelope decrypt-and-refund asymmetry while preserving protection against invalid envelopes. The work should include an explicit threat model, adversarial tests, envelope-validation rules, timeout and recovery behavior, upgrade or migration procedures, and an independent smart-contract and cryptographic audit. Audit scope, findings, remediation commits, deployed bytecode hashes, canonical addresses, and migration instructions should be public. Until then, v1 is appropriately described as an evidence-rich exchange mechanism for attributable, contractually bound counterparties pending a hardened settlement design.

▶ **Core assurance.** Canonical MOLNFT, GL1F, and related cryptographic components should receive independent review proportionate to their control surfaces. Neural-model deployments intended as permanent scientific references should use immutable or one-shot weight initialization rather than continuing owner mutability. The inherited blocklist capability should be removed at the protocol level, rather than merely remaining empty and dormant.

▶ **Reproducibility and systems evidence.** Node, contract, and front-end builds should be reproducible from pinned source, compiler, dependency, and configuration manifests. A publication-grade retrieval release should include input and output checksums, byte-identity tests, raw timing records, cold- and warm-cache runs, local-file and repository baselines, multiple independently operated hosts, and session-level uncertainty. A separate storage-sustainability report should quantify application payload,

EVM and Cosmos state overhead, database amplification, snapshot and state-sync costs, archive-node requirements, replication burden, pruning assumptions, and plausible growth scenarios. The resulting capacity model will establish the operational and economic range of sustainable replicated state.

14.2 Phase II: the first domain-complete scientific vertical

The primary scientific evidence milestone is an independently reproduced MOLNFT-to-GL1F protein-stability workflow, beginning with a $\Delta\Delta G$ predictor. A complete vertical should bind an exact on-chain molecular structure to a versioned feature pipeline, immutable model identity, deterministic GL1F execution, and a citable result. Any off-chain preprocessing must be specified, versioned, hashed, and independently executable rather than hidden behind the on-chain prediction call.

Evaluation should use leakage-resistant, homology-aware splits; established external baselines; prospective or genuinely held-out validation where feasible; uncertainty and calibration analysis; documented quantization error; exact dataset, structure, feature, code, and model hashes; and local/on-chain parity tests. A third party should reconstruct the MOLNFT input, reproduce preprocessing, invoke the same GL1F model, and recover the published output without privileged infrastructure. Model cards, benchmark manifests, parity vectors, and result commitments should be archived as first-class objects. Completion will transform the existing runtime and data layers into a fully demonstrated scientific instrument and establish a reusable validation template for subsequent domains.

After that evidence gate, GLAST-RIGID-1 is the flagship long-term application path: publish the conformance profile and reference engine, freeze the first scientifically meaningful GLAST-Forest controller, benchmark it against accepted baselines, and obtain independent reproduction before treating it as scientific-search infrastructure (section 6).

14.3 Phase III: independent deployment and replication

Success should then be measured outside the originating contributors: independently operated validators with disclosed voting-power distribution; archival mirrors; third-party nodes and indexers; models or applications deployed by unaffiliated builders; laboratory or institutional reproduction and external scientific validation of the Phase II result; external datasets connected through published interfaces; peer-reviewed publications and subsequent citations; and documented use in research or public-interest workflows. Replication grants and public test suites can lower entry costs, while validator- and archive-node programs can broaden operational responsibility. The relevant signals are inspectable external artifacts, signed replication reports, distinct deployment addresses, public methods, and citations that can be checked without relying on GenesisL1's authors.

14.4 Phase IV: expansion from the common substrate

Research and development can progress in parallel under accurate status labels. As assurance and the first scientific vertical mature, the same primitives can support a broader scientific surface.

▶ **Genomic infrastructure: a canonical sequence format and the reference genome on-chain.** A major initiative places the human genome reference on-chain not as a symbolic gesture but as infrastructure, and it rests on a concrete foundation already in development: **GL1-Seq**, a canonical encoding and tokenization layer for genetic information. At its core is **GL1ENCv2**, a deterministic sequence codec that converts DNA, RNA, protein, and symbolic sequences into a canonical byte representation suitable for on-chain hashing and storage. The codec selects the most compact eligible encoding automatically: a 2-bit packing for pure nucleotide genomes (`A/C/G/T`), a 4-bit IUPAC-ambiguity bitmask supporting fast ambiguity-aware matching and gaps, a 6-bit alphabet for protein and symbolic strings, and an ASCII fallback. Crucially, the format is specified for *strict on-chain/off-chain parity*—a Solidity implementation must reproduce the Python encoder byte-for-byte, with canonical-validity rules (fixed padding, stable encoding identifiers, versioned wire format) that forbid alternate encodings of the same logical sequence. This is what makes a sequence commitment verifiable rather than merely stored: the same bytes hash identically on-chain and off. On top of the codec, **SEQNFT** tokenizes sequences as first-class on-chain assets and a companion alignment specification (**GL1ALIGN**) defines on-chain sequence comparison, extending to genomics the same reproducibility guarantees MOLNFT provides for structures. Two-bit packing reduces nucleotide payloads but does not, by itself, make whole-genome replication economically or operationally justified. The implementation should first benchmark canonical compression, commitments, deltas, state overhead, sync cost, and archive requirements using only public reference sequences. The version-aware design—references, differences, updates, and provenance as first-class operations, conceptually a "Git for biological data"—is what lets a living reference be maintained rather than frozen. Together with MOLNFT's structural corpus (section 4) and GLAST's search layer, GL1-Seq gives bioinformatics, molecular NFTs, open datasets, and biological AI a citable, comparable, machine-usable public anchor for humanity's reference genome. The specifications and reference implementation are developed openly (appendix E). This work is at the specification-and-implementation stage; writing the full reference genome on-chain is the milestone it builds toward, not a completed fact.

▶ **Rewarded storage for big scientific data: a rewarded IPFS sidecar.** On-chain state and IPFS pointers together handle compact artifacts and content commitments well, but frontier science routinely produces datasets—whole-genome sequence sets, sequencing runs, imaging archives, simulation trajectories, cohort data, large training corpora—whose sheer size makes naive on-chain storage impossible and best-effort IPFS pinning unreliable. To close this gap, a rewarded *IPFS sidecar* is planned: a decentralized hosting layer in which independent operators are paid, in `L1`, to durably store and serve large datasets that are committed to and addressed from the chain. The chain holds a compact cryptographic commitment—a content-addressed Blob ID, Merkle root, size, and metadata, exactly as GL1-Seq's encoding already specifies for IPFS payload bytes—while the bulk bytes live in the sidecar, replicated across independent operators. This separates two concerns cleanly: the ledger guarantees *integrity*

and provenance under its hash and consensus assumptions, while the rewarded sidecar would target *availability at scale*. Operators could earn ongoing L1 rewards for evidence of availability—funded, as proposal #75's IPFS-pinning tradition already anticipates, from the community pool or from per-dataset fees—so that persistence is incentivized rather than charitable, mirroring at data scale the security logic of the ledger itself. The IPFS sidecar is the natural counterpart to GL1-Seq and MOLNFT: the codecs and NFT layers define *what* a scientific object is and *how* it is committed, while the sidecar provides the incentivized substrate to hold the underlying big data—MOLNFT's structural corpus, GL1-Seq's genomic data, `datasetnft.org`'s datasets—at the volumes real science demands. This is a design direction under active development, not a deployed service: proof-of-storage or retrievability, challenge sampling, repair, Sybil resistance, collusion, geographic diversity, reward leakage, and data-removal obligations remain unresolved design questions. The on-chain commitment format it depends on exists today, while the operator network, reward parameters, and rollout remain subject to design, testing, governance, and audit.

▶ **Agentic science.** Two developments open GL1F to autonomous workflows. An *Agent Hub* API lets external AI models, programs, and systems—local or remote—access GL1F Forest directly, turning the platform into open scientific infrastructure callable by agents and tools. Building on it, *GL1F Autopilot* lets any LLM, through Agent Hub, create, coordinate, and manage model-publication workflows: AI that helps build AI while ledger-recorded steps remain inspectable. Off-chain prompts, training, tools, and preprocessing are not made reproducible merely by recording the resulting transactions.

▶ **DeSci institutions.** A proposed DAO engine would let scientific communities choose transparent funding, contributor-management, access, and incentive mechanisms under separately assessed legal and security constraints, extending the governance and federation model of section 9. Optional financial applications remain outside the scientific core. In parallel, `datasetnft.org` is being rebuilt as a permissionless dataset service intended to connect dataset commitments, client-side training, and model deployment while keeping each off-chain dependency explicit.

Together these directions extend one architecture: datasets, models, computation, encrypted IP, funding, and governance in a single inspectable environment. Governance determines implementation; published evidence makes every new capability independently legible.

15 Conclusion

GenesisL1 is a live sovereign scientific ledger. At the July 2026 reference state, more than 733,919 tokenized molecular records, roughly 229,000 PDB parent records represented in second-generation chain state, deterministic GL1F Model NFTs, a bounded neural-inference proof of concept, encrypted CIPNFT capsules, public governance, and public-goods funding coexist in one consensus history. Complete local reconstruction has been demonstrated for selected MOLNFT v2 structures; checksum-complete corpus validation remains a pending evidence milestone.

The near-term scientific milestone is the first independently reproduced and domain-evaluated MOLNFT-to-GL1F protein-stability workflow. Capacity measurement, independent audits, CIPNFT hardening, immutable neural deployments, and broader independent node operation remain requirements rather than completed assurances. GLAST-RIGID-1 is the long-term flagship research application and remains subject to specification, implementation, benchmarking, and independent evaluation.

GenesisL1 is not merely a blockchain for publishing scientific artifacts. It is a public scientific execution environment in which canonical data, deterministic models, reproducible computation paths, encrypted scientific capsules and rights-related state, and open governance share one consensus history. That history can establish ordering, byte identity, authorization, and deterministic execution; scientific validity still comes from domain evidence, peer review, and independent replication.

A Glossary

BFT — Byzantine Fault Tolerant
A consensus protocol that remains safe against adversarial behavior by up to a bounded fraction of participants.

BinaryCIF (BCIF)
A compact binary encoding of macromolecular structural data; MOLNFT v2 stores it as `base64(gzip(BinaryCIF))` fully on-chain.

CID — Content Identifier
A self-describing content address combining a CID version, content multicodec, and multihash; a CIDv1 string representation additionally uses multibase. A CID identifies an encoded block or DAG root and does not by itself guarantee canonicalization, availability, or scientific validity.

DAO — Decentralized Autonomous Organization
A community whose coordination rules are encoded in on-chain smart contracts.

DEK — Data Encryption Key
The symmetric key with which a plaintext payload is encrypted, itself wrapped under one or more recipient public keys.

DeSci — Decentralized Science
The use of public-blockchain primitives for funding, curating, and coordinating scientific work.

EVM — Ethereum Virtual Machine
The deterministic execution environment defined by the Ethereum protocol and inherited by EVM-compatible chains.

GBDT — Gradient-Boosted Decision Trees
A widely used ensemble machine-learning method for structured-data tasks.

GLAST — GenesisL1 Local Alignment Search Tool
The deployed operator-local metadata and sequence-search service over reconstructed MOLNFT data (Whoosh indexing + Parasail alignment).

GLAST structural program
The in-development research direction for deterministic, model-guided three-dimensional alignment of scientific structures; it is distinct from the deployed sequence-search service.

GLAST-Forest
Versioned GL1F models proposed to prioritize candidate search and refinement while canonical geometry defines correspondences and the reported alignment score.

GLAST-RIGID-1
The proposed first conformance profile for deterministic pairwise rigid protein-structure alignment; it is not a deployed or validated capability in this edition.

GL1C — GenesisL1 Chunk
The runtime-code magic (`0x474c3143`) marking a GenesisL1 code-as-data chunk contract used by GL1F.

GL1F — GenesisL1 Forest
The on-chain GBDT model studio, marketplace, and deterministic inference runtime.

IBC — Inter-Blockchain Communication
A protocol for authenticated message-passing between sovereign chains in the Cosmos ecosystem.

L1 coin
The native GenesisL1 asset used for gas, proof-of-stake bonding, governance voting power, and the community pool. Application-layer uses are optional.

ML-KEM
Module-Lattice Key Encapsulation Mechanism; the NIST-standardized post-quantum key-encapsulation primitive (FIPS 203), used in CIPNFT's optional quantum-resistant envelope.

Model NFT
An ERC-721 token whose metadata commits to a GL1F model's weights, making the model an ownable, transferable, and on-chain-invocable asset.

Slashing
Protocol-defined reduction of validator and delegated stake for specified faults; under the current GenesisL1 staking implementation, slashed tokens are burned.

B Reference contract surfaces (informative)

The following are indicative protocol-level interfaces, in abbreviated Solidity-style notation. They are documentation, not normative source code. Canonical deployed addresses on EVM chain ID 29 at the reference state include: MOLNFT v2 `0xd58B01f6C18086e5202cdC5D7Ad3E41790360102`; GL1F Store `0x9CdbC23392648Bd27B4A5eD09c0fEa9452454B54`, Registry `0x33c9844F77a07e36B98f0FFf8201B8A8b02c2a69`, Model NFT `0x44Dc1c54B8D579B42d78cC21cf8260DC0A3279fA`, Runtime `0xD2fD0cf461a6cb56Fc08d9aEc120833D8E79044E`, and Market `0xA9bfa0a719b7F73cE85CA1E7f23af626D383fB46`; neural policy proof of concept `0x37D6518FbABd982e1908251A9cb4aD97b48BB989`; and CIPNFT `0x05e4F1929947bE93a853F2de155fc1d6137f8446`. Address publication is not an audit endorsement.

```
// MOLNFT v2 (fully on-chain; parent-child reconstruction)
function getCombinedData(uint256 tokenId) external view returns (bytes memory); // base64(
    gzip(BCIF))
function structureName(uint256 tokenId) external view returns (string memory);

// GL1F Model NFT + runtime (ERC-721 + deterministic inference)
function predictView(bytes32 modelId, bytes calldata packedFeaturesQ) external view returns
    (int256);
function predictOwnerView(bytes32 modelId, bytes calldata packedFeaturesQ, uint256 deadline,
     bytes calldata sig)
   external view returns (int256); // EIP-712 gated; fee-safe under eth_call spoofing
```

```
function predictTx(bytes32 modelId, bytes calldata packedFeaturesQ) external payable returns
     (int256);
function inferenceFee(bytes32 modelId) external view returns (uint256);
function searchTitleWords(bytes32[] calldata words, uint256 cursor, uint256 limit)
   external view returns (uint256[] memory tokenIds, uint256 nextCursor);

// Neural policy proof of concept (weights owner-mutable in this deployment)
function weightsReady() external view returns (bool);
function inferBest(bytes calldata state, bytes calldata legalMask) external view returns (
    uint16 idx, int256 logit);
function inferTopK(bytes calldata state, bytes calldata legalMask, uint256 K)
   external view returns (uint16[] memory idx, int256[] memory logits);

// CIPNFT v1 (encrypted capsule; attributable bilateral transfer, not atomic fair exchange)
function mintEncrypted(bytes calldata ciphertext, bytes calldata ownerEnvelope, uint8
    encMode /*0=X25519,1=ML-KEM*/)
   external payable returns (uint256 tokenId);
function createOffer(uint256 tokenId, uint256 expiry) external payable; // buyer escrows
    exact price
function deliverOffer(uint256 tokenId, address buyer, bytes calldata newOwnerEncDEK)
    external; // seller re-wraps
function finalizeOffer(uint256 tokenId) external; // buyer finalizes after verification
```

C MOLNFT single-host engineering observation (informative)

The MOLNFT v2 retrieval experiment (section 4.4) is implemented by `bench/molnft_rpc_statbench.py` in the `GenesisL1/molnft_v2` repository. It measures *fetch-by-token-ID*—assuming a pre-existing local PDB-ID-to-token-ID index—as `getCombinedData(token)` → base64-decode → gunzip → BCIF, and compares wall-clock latency with an HTTP request to the RCSB BinaryCIF endpoint.

Current observation. Eight selected structures span approximately 151 kB to 3.3 MB. Each source–structure pair receives 30 warm interleaved repetitions. Sources are a public balancer, a remote GenesisL1 node, a localhost node, and the RCSB endpoint. The recorded pooled medians were 59.9 ms for localhost GenesisL1, 85.4 ms for RCSB, 543.4 ms for the remote node, and 1.141 s for the balancer. These values describe one co-located engineering run, not a comparative performance claim.

Known limitations. The script does not currently publish hashes proving identity between reconstructed and RCSB objects; the raw timing JSON and a complete environment manifest were not included in the repository snapshot reviewed for this edition; hardware, location, timestamp, cache state, connection reuse, node database configuration, and network conditions are not fully reported; the structures are hand-selected; and repeated timings from the same eight objects are correlated. Consequently, pooled tests that treat 240 timings as independent observations exaggerate precision, and 30 repetitions do not support strong p99 claims.

Required publication-grade release. Archive the exact script commit, raw timing records, source and output checksums, node/version configuration, hardware and operating system, cache protocol, geography, timestamps, and connection reuse. Add local filesystem, locally mirrored PDB, IPFS/object-store, cold-cache, and PDB-ID lookup baselines; repeat across days, hosts, and regions; pre-specify sampling; and estimate uncertainty at the structure or independent session level. The current result should be cited only as a July 2026 single-host engineering observation.

```
pip install web3 numpy scipy
python3 bench/molnft_rpc_statbench.py --iterations 30 \
  --localhost-url http://localhost:8545 --json rpc_stats.json
```

D Independent verification and reproducibility (informative)

> **Verification path**
>
> Establish chain context → verify pinned contracts and state → reconstruct MOLNFT → reassemble GL1F → execute the specified pipeline → compare outputs → publish a signed evidence bundle.

An independent reproduction of a MOLNFT-to-GL1F result should follow the same explicit path:

1. **Establish chain context.** Pin the Cosmos and EVM chain IDs, genesis hash, block height and hash, block time, node release and commit, and a verified header or explicitly disclosed trusted checkpoint. An RPC response alone is not independent verification of the chain.
2. **Verify deployed state.** Record canonical contract addresses, runtime bytecode hashes, relevant storage or event state, and the historical block tag used for every call.
3. **Reconstruct the scientific object.** Reassemble the ordered MOLNFT parent–child payload, decode it under the pinned format profile, and compare its checksum with the published input manifest.
4. **Reconstruct the model.** Resolve the Model NFT and registry entry, reassemble ordered GL1C chunks, decode the GL1F serialization, and verify its model ID and byte hash.
5. **Reproduce preprocessing.** Execute the pinned parser, feature schema, quantization, parameters, and build; archive every intermediate checksum needed to distinguish input drift from runtime drift.
6. **Compare execution.** Run the local reference implementation and the historical-state EVM call, then compare outputs and released parity vectors bit-for-bit. Any permitted tolerance must be specified before evaluation.
7. **Publish evidence and evaluate science separately.** Sign and archive the inputs, software and build identifiers, environment, hashes, intermediate artifacts, calls, and outputs. Then assess scientific validity against held-out evidence, accepted baselines, calibration, uncertainty, and—where relevant—laboratory replication.

> **Verification boundary**
>
> Matching an on-chain result establishes computational repeatability under the pinned trust assumptions. It does not by itself establish biological correctness, clinical utility, authorship, lawful rights, or scientific replicability.

GL1F. Record the dataset bytes or availability location and hash, split specification, preprocessing and feature schema, trainer source/build, seed, hyperparameters, serialization version, `scaleQ`, overflow checks, quantization error, Keccak256(modelBytes), ordered chunk pointers, canonical contract addresses, license/terms, and a released parity suite comparing local and on-chain inference. Dataset hashes prove identity, not availability or scientific adequacy.

Neural policy proof of concept. Pin the contract address, block height, complete weight-state hash, model architecture, parameter count, data source and licence, train/validation/test split by independent game or player, top-k metrics with uncertainty, legal-action rate before masking, quantization delta, gas or call-budget evidence, and execution latency. Freeze weights or treat every result as block-specific.

CIPNFT. Preserve the originating AAD = (`chainId`, `contract`), ciphertext hash, envelope mode, implementation versions, recipient-key fingerprint, terms version, client build, counterparty wallet attribution, and the public offer–escrow–envelope-posting–finalization or refund chronology. Successful decryption does not establish lawful ownership, acceptance, or compliance with off-chain terms. The v1 transfer flow should be used only between attributable counterparties operating under explicit external terms. Before it is represented for broader trust-minimized exchange, deploy the hardened state machine, publish the migration, and complete an independent smart-contract and cryptographic audit.

Node and governance state. Pin the Cosmos chain ID, EVM chain ID, genesis hash, node tag and commit, reproducible build inputs, block height/time, validator snapshot, module parameters, active administrator values, contract bytecode hashes, and raw endpoint responses. A reproducible claim should be re-creatable from an immutable evidence bundle, not from a moving endpoint alone.

E Verification endpoints and repositories

All dynamic network-state values in this edition are frozen to reference height 13,313,640 on Cosmos chain `genesis_29-2`, timestamped 2026-07-19 13:07:53 UTC. The EVM chain ID is 29 (`0x1d`). At that state:

- total supply: 46,475,691.0103954182 L1;
- bonded stake: 23,982,866.5695161663 L1;
- community pool: 5,112,381.2157407622 L1;
- observed inflation: 10 %; configured range: 5 %–10 %; and
- community tax: 20 %.

The 20-validator concentration and governance counts are reported in table 9. Later queries will differ because the chain continues to produce blocks.

Live query surfaces:

- Supply: `https://1317.genesisl1.org/cosmos/bank/v1beta1/supply`
- Community pool and distribution parameters: `https://1317.genesisl1.org/cosmos/distribution/v1beta1/community_pool` and `https://1317.genesisl1.org/cosmos/distribution/v1beta1/params`
- Mint parameters and inflation: `https://1317.genesisl1.org/cosmos/mint/v1beta1/params` and `https://1317.genesisl1.org/cosmos/mint/v1beta1/inflation`
- Staking pool and bonded validators: `https://1317.genesisl1.org/cosmos/staking/v1beta1/pool` and `https://1317.genesisl1.org/cosmos/staking/v1beta1/validators?status=BOND_STATUS_BONDED&pagination.limit=200`
- Governance proposals: `https://1317.genesisl1.org/cosmos/gov/v1/proposals?pagination.limit=500`
- ICA host parameters: `https://1317.genesisl1.org/ibc/apps/interchain_accounts/host/v1/params`
- EVM JSON-RPC: `https://rpc.genesisl1.org`; CometBFT status: `https://26657.genesisl1.org/status`

Load balancers and mirrors can lag or cache responses. A reproducible citation should save the raw JSON, HTTP timestamp, reported height, endpoint, query parameters, and a checksum in an immutable evidence bundle. This edition does not claim that a one-command verifier is published unless an exact repository path and commit are supplied with the release.

Canonical source repositories:

- Node: `https://github.com/GenesisL1/genesis-crypto`, tag `v1.6.2`, commit `ee86be82183d4972c72a1a9acaaf0c9dc5c34787`.
- MOLNFT v2: `https://github.com/GenesisL1/molnft_v2`.
- GL1F Forest: `https://github.com/GenesisL1/Forest`.
- Neural-policy proof of concept: `https://github.com/GenesisL1/GL1chess`.
- CIPNFT: `https://github.com/GenesisL1/cipnft`.
- GL1-Seq specification: `https://github.com/bioinformatics-company/gl1-seq`.

The archived node tag and contract addresses in appendix B support inspection; they do not substitute for reproducible build hashes, bytecode verification, independent audits, or scientific replication.

F Authorship, research contributions, and availability

Author. Mikhail Fedorov, M.D., is a medical doctor whose work spans bioinformatics, data science, and decentralized scientific systems. He holds a Graduate Certificate in Bioinformatics from Harvard University Extension School. He founded GenesisL1 and remains a continuing contributor.

Research and development contributors. Decentralized Science Labs LLC and Bioinformatics LLC are U.S. research-and-development companies that have contributed to the research and software development described in this whitepaper.

Author contributions and affiliations. Mikhail Fedorov synthesized and wrote this whitepaper. Contributions to the underlying research and software are attributed through the cited repositories and commit histories. His founder role and affiliations with the R&D companies named above are stated for transparency.

Data, code, and licensing. Public endpoints, source repositories, deployed contract addresses, and release identifiers are listed in appendices B and E. On-chain records derived from PDB, AlphaFold, UniProt/Swiss-Prot, and ESMFold retain their source attribution and remain subject to applicable source terms and licences. This whitepaper is licensed under CC BY 4.0; software licensing is specified in each repository.

Correspondence. Whitepaper correspondence may be directed to Mikhail Fedorov at mikhail.fedorov@genesisl1.com.

Versioning and technical discussion. Source repositories and issue tracking are available at github.com/GenesisL1. Each whitepaper release identifies its author, publication date, material changes, and supporting code or evidence identifiers.

References

[1] National Academies of Sciences, Engineering, and Medicine. *Reproducibility and Replicability in Science*. National Academies Press, Washington, DC, 2019. https://doi.org/10.17226/25303.

[2] M. Baker. 1,500 scientists lift the lid on reproducibility. *Nature*, 533(7604):452–454, 2016.

[3] H. M. Berman et al. The Protein Data Bank. *Nucleic Acids Research*, 28(1):235–242, 2000.

[4] J. Jumper et al. Highly accurate protein structure prediction with AlphaFold. *Nature*, 596(7873):583–589, 2021.

[5] M. Varadi et al. AlphaFold Protein Structure Database: massively expanding the structural coverage of protein-sequence space. *Nucleic Acids Research*, 50(D1):D439–D444, 2022.

[6] Z. Lin et al. Evolutionary-scale prediction of atomic-level protein structure. *Science*, 379(6637):1123–1130, 2023.

[7] S. Nakamoto. Bitcoin: A peer-to-peer electronic cash system. https://bitcoin.org/bitcoin.pdf, 2008.

[8] V. Buterin. Ethereum: A next-generation smart contract and decentralized application platform. Ethereum White Paper, 2014.

[9] J. Kwon. Tendermint: Consensus without mining. Draft v0.6, 2014.

[10] J. Kwon and E. Buchman. Cosmos: A network of distributed ledgers. Cosmos White Paper, 2016.

[11] E. Buchman. *Tendermint: Byzantine Fault Tolerance in the Age of Blockchains*. PhD thesis, University of Guelph, 2016.

[12] W. Entriken, D. Shirley, J. Evans, and N. Sachs. EIP-721: Non-fungible token standard. Ethereum Improvement Proposals, 2018.

[13] Ethereum Improvement Proposals. EIP-170: Contract code size limit. https://eips.ethereum.org/EIPS/eip-170.

[14] Ethereum Improvement Proposals. EIP-712: Typed structured data hashing and signing. https://eips.ethereum.org/EIPS/eip-712.

[15] Ethereum Improvement Proposals. EIP-1474: Remote procedure call specification (optional `from` in `eth_call`). https://eips.ethereum.org/EIPS/eip-1474.

[16] A. Beregszaszi, H. Wang, and P. Kong. EIP-3855: `PUSH0` instruction. Ethereum Improvement Proposals. https://eips.ethereum.org/EIPS/eip-3855.

[17] A. Akhunov and M. Moreno. EIP-1153: Transient storage opcodes. Ethereum Improvement Proposals. https://eips.ethereum.org/EIPS/eip-1153.

[18] P. Sisco and G. Fu. EIP-5656: `MCOPY` — memory copying instruction. Ethereum Improvement Proposals. https://eips.ethereum.org/EIPS/eip-5656.

[19] V. Buterin, D. Feist, et al. EIP-4844: Shard blob transactions. Ethereum Improvement Proposals. https://eips.ethereum.org/EIPS/eip-4844.

[20] V. Buterin, S. Wilson, et al. EIP-7702: Set EOA account code. Ethereum Improvement Proposals. https://eips.ethereum.org/EIPS/eip-7702.

[21] 0xSequence. SSTORE2: cheaper storage in contract bytecode, read via `EXTCODECOPY`. https://github.com/0xsequence/sstore2.

[22] J. Benet. IPFS — content-addressed, versioned, P2P file system, 2014.

[23] IPFS Standards. Content Identifier (CID) specification, permanent version dated 26 June 2026. https://specs.ipfs.tech/cid/. See also IPFS Documentation, Content Identifiers. https://docs.ipfs.tech/concepts/content-addressing/ (accessed 20 July 2026).

[24] C. Goes. The inter-blockchain communication protocol: an overview, 2020.

[25] T. Chen and C. Guestrin. XGBoost: A scalable tree boosting system. In *Proc. 22nd ACM SIGKDD*, 785–794, 2016.

[26] J. H. Friedman. Greedy function approximation: a gradient boosting machine. *The Annals of Statistics*, 29(5):1189–1232, 2001.

[27] Y. Zhang and J. Skolnick. Scoring function for automated assessment of protein structure template quality. *Proteins: Structure, Function, and Bioinformatics*, 57(4):702–710, 2004. https://doi.org/10.1002/prot.20264.

[28] M. van Kempen et al. Fast and accurate protein structure search with Foldseek. *Nature Biotechnology*, 42:243–246, 2024. https://doi.org/10.1038/s41587-023-01773-0.

[29] M. Margelevicius. GTalign: spatial index-driven protein structure alignment, superposition, and search. *Nature Communications*, 15:7305, 2024. https://doi.org/10.1038/s41467-024-51669-z.

[30] V. Stodden, J. Seiler, and Z. Ma. An empirical analysis of journal policy effectiveness for computational reproducibility. *PNAS*, 115(11):2584–2589, 2018.

[31] D. J. Bernstein. ChaCha, a variant of Salsa20. In *Workshop Record of SASC 2008*, 2008.

[32] Y. Nir and A. Langley. ChaCha20 and Poly1305 for IETF protocols. RFC 8439, IETF, 2018.

[33] National Institute of Standards and Technology. Module-lattice-based key-encapsulation mechanism standard. FIPS 203, 2024.

[34] J. Bos et al. CRYSTALS-Kyber: a CCA-secure module-lattice-based KEM. In *IEEE EuroS&P*, 353–367, 2018.

[35] P. De Filippi and A. Wright. *Blockchain and the Law: The Rule of Code*. Harvard University Press, 2018.

[36] The Maker Team. The Dai stablecoin system. MakerDAO White Paper, 2017.

[37] M. Carlsten, H. Kalodner, S. M. Weinberg, and A. Narayanan. On the instability of Bitcoin without the block reward. In *Proc. ACM CCS*, 154–167, 2016.

[38] N. Atzei, M. Bartoletti, and T. Cimoli. A survey of attacks on Ethereum smart contracts (SoK). In *Principles of Security and Trust (POST 2017)*, 164–186. Springer, 2017.

[39] Decentralized Science Labs. GenesisL1 Forest (GL1F): a scientific explanation and technical documentation for an on-chain EVM gradient-boosted tree studio, 2026. https://gl1f.com/GL1F.pdf.

[40] Decentralized Science Labs. CIPNFT on GenesisL1: cryptographic information protocol NFTs for confidential IP tokenization and exchange, 2026. https://cipnft.com/CIPNFT_on_GenesisL1_Whitepaper.pdf.

[41] Decentralized Science Labs. Neural-policy chess inference inside the EVM: a GenesisL1 case study and a general template for on-chain scientific machine learning, 2025.

[42] Decentralized Science Labs. MOLNFT: molecular NFTs on GenesisL1, 2024–2026. https://molnft.org/; source https://github.com/GenesisL1/molnft_v2.