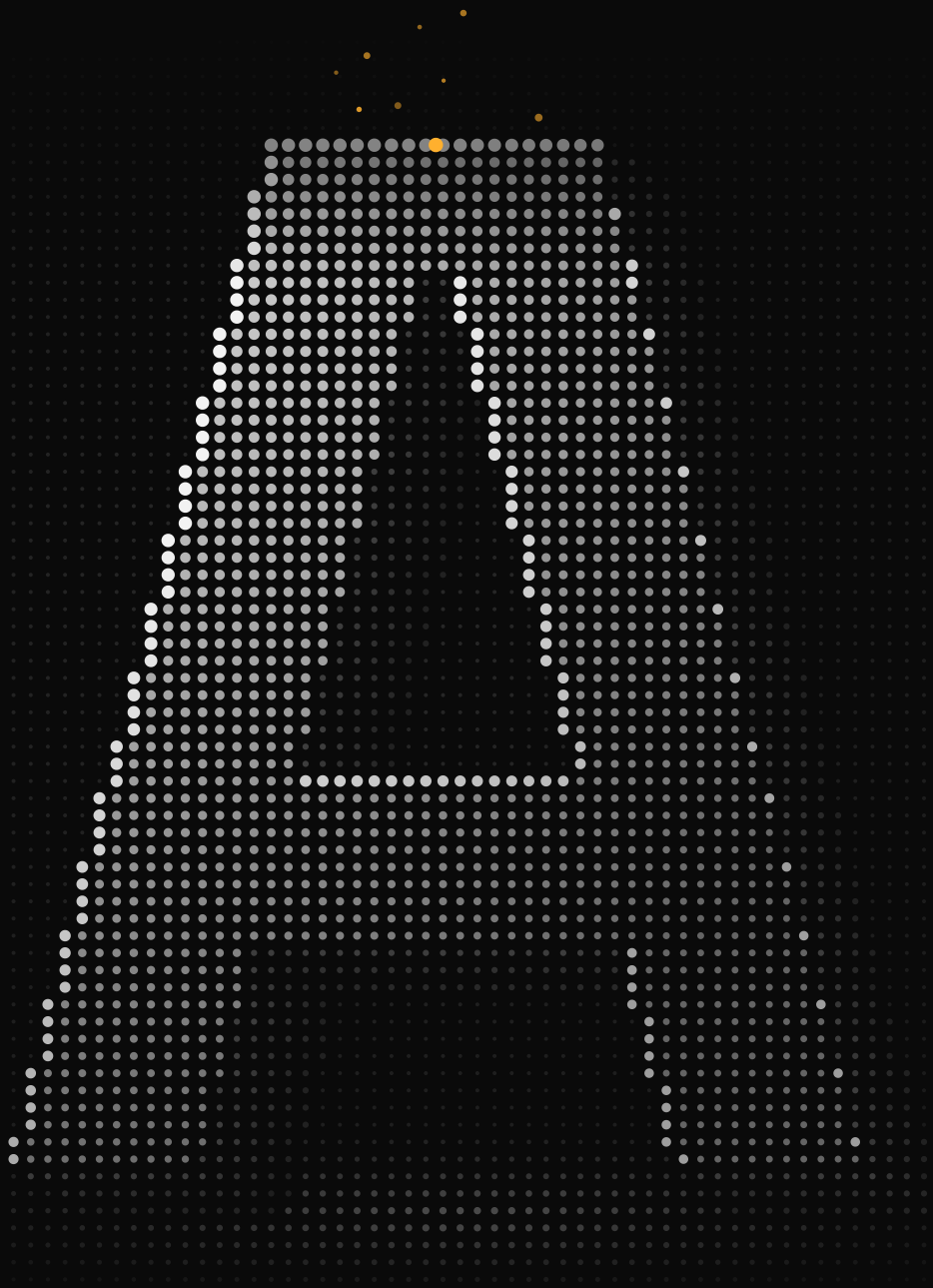


Litepaper

AIOS: Proof of Inference on a Machine-Intelligence Layer 1

Technical Litepaper

Contents

	Abstract	4
1	**Introduction** The problem · Approach · Contributions · Roadmap	5
2	**System and Threat Model** Architecture and participants · Trust assumptions · Adversary · Security goals	7
3	**Proof of Inference** Job lifecycle · The canonical output · Selection, weight, and capture · Lazy verifiers, commit-then-reveal, and honeypots · Rejected and deferred mechanisms	9
4	**Determinism and Verification Regimes** The AIOS Module · Two regimes, three paths · Determinism-gated penalties	13
5	**Verification Modes and Assurance Classes** Mode A and Mode B · Assurance classes	15
6	**Mining and the Registration Burn** Reward on match · The registration burn · Security conditions · Consequences	17
7	**Block Ordering**	20
8	**The Model Internet, Native Models, and Adapters** The Model Internet · Native Models and the Neural VM · The Adapter Registry	21
9	**The Data Internet and Verified Memory** Verifiable retrieval · The proof · Verified Memory · Limits	23
10	**The $AIOS Token** Supply and conservation · Distribution · Fees, royalties, and value · Transfer tax and staking · Governance · Parameter status	26
11	**Products and Roadmap** Products · Phases	30
12	**Risks and Limitations**	32
13	**Conclusion**	34
	References	35

Figures

1	System architecture	7
2	The Proof of Inference job lifecycle	9
3	Committee capture probability by committee size	11
4	The four assurance classes	16
5	The registration-burn baseline B(e)	18
6	Weight storage by numeric precision	22
7	Verifiable retrieval	24
8	The dual-home supply	26
9	Genesis distribution of the 300,000,000 supply	27
10	Proof-first sequencing	30

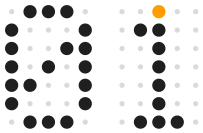
This litepaper is a public technical brief of the AIOS design: nothing described here is deployed, numbers marked `illustrative` are calibrated near launch, and numbers marked `locked` are canon.

Weights are programs. Inference is consensus.

Front matter

Abstract

A caller who pays for a model call receives bytes and an unverifiable claim about which model produced them. AIOS is a sovereign Layer 1 on Cosmos SDK and CometBFT in which inference is a first-class blockchain operation. Under Proof of Inference, a committee of Miners, seated by an unbiasable random beacon and weighted by verified work rather than capital, independently recomputes each job on a pinned integer-only runtime, commits output hashes before revealing them, and accepts an output only when more than two thirds of committee work weight agrees. Matching Miners are paid, mismatching Miners are not, and a job without a supermajority is re-routed and finally refunded in full. The same round proves retrieved context and composed adapters. The guarantee is provenance of execution, never truth of output. It is exact only for small integer models on the validated runtime and statistical for larger ones. The mining layer holds no stake: its sole at-risk value is a burned registration fee, a weaker anchor than staking that is bounded by value caps, honeypots, and per-entity caps. The native token, $AIOS, has a fixed supply of 300,000,000.



Chapter 1

Introduction

This section states the problem, the construction AIOS proposes, and the contributions of this paper.

1.1 The problem

A model call returns an output, a price, and a provider's statement about which model ran; none of these is evidence. Weights can be swapped for a cheaper model, context can be altered before generation, and an output can be cached or fabricated. Existing decentralized inference networks largely reward liveness and participation, establishing that a node is online and holds the hardware it claims. Liveness is not correctness: a present node can still return a wrong or substituted output.

AIOS asks whether a distributed network can establish that a result came from specified registered weights, executed as specified, without trusting any single party. Committee recomputation answers it when every honest participant computes the same bytes. The obstacle is arithmetic: floating-point computation under IEEE 754 [1] is not associative, and reduction order varies across kernels, batch shapes, and accelerators. A check recorded in the AIOS canon on 2026-07-05 found 0.0% bitwise agreement between A100 and H100 GPUs on the same model; batch invariance is not hardware invariance.

1.2 Approach

AIOS removes floating point from the verified path. Every Miner runs the AIOS Module, a pinned integer-only runtime in the lineage of integer-only transformer inference [2, 3, 4, 5], on which the same input is to yield the same output hash on any hardware, an engineering target that must be validated before mainnet. A committee recomputes each job, and agreement weighted by verified work settles it. This is Proof of Inference: correctness becomes a constraint the protocol enforces rather than a claim a provider makes. Weights are programs; inference is consensus.

The chain fuses four pillars:

1. **Chain-state-as-model.** Small integer-quantized Native Models live in chain state and execute in consensus in the Neural VM.
2. **Blockspace-as-inference.** Jobs are verified by committee consensus and committed by a work-gated BFT producer set; callers buy verified inference like gas.

3. **The Model Internet.** Every model and every adapter, a LoRA-class fine-tune [6], has an onchain address, owner, price, and verification mode bound to Proof of Inference.
4. **The Data Internet.** Corpora register as owned, priced assets, and the retrieval step of retrieval-augmented generation [7] is recomputed by the same committee, so a call's context is proven with its model.

No existing decentralized inference network fuses all four; the differentiation is the fusion and the mechanism. We call AIOS the machine-intelligence Layer 1 and, in its secondary register, the cognitive substrate.

1.3 Contributions

This paper makes the following contributions:

1. **Proof of Inference**: work-weighted committee recomputation on a pinned integer runtime, settled by a commit-then-reveal supermajority, with reward on match and no mining stake (Section 3).
2. **A separation of exact and statistical guarantees**, with penalties gated on validated determinism and assurance classes that confine "verified" to committee consensus (Sections 4 and 5).
3. **A burn-only anchor for stakeless mining**, its security conditions, and an analysis of committee capture and audit survival (Sections 3 and 6).
4. **One proof over model, adapter, retrieved data, and output** (Sections 8 and 9).
5. **A fixed-supply token** dual-homed across Ethereum and the AIOS L1 under a per-block conservation invariant (Section 10).

1.4 Roadmap

Sections 2 to 7 cover the threat model, mechanism, determinism, assurance, mining, and ordering; Sections 8 to 12 cover the model and data layers, the token, products and phases, and risks.

Chapter 2

System and Threat Model

This section fixes the architecture, participants, trust assumptions, adversary, and security goals against which the rest of the paper is argued.

2.1 Architecture and participants

AIOS is a sovereign Cosmos SDK chain running the CometBFT engine [8] unchanged, with its logic in native modules: `x/inference` (jobs and settlement), `x/neuralvm` (Native Models and honeypot ground truth), `x/registry` (models, adapters, miner registration), `x/data` (corpora), `x/producer` (producers), and `x/attest` (TEE attestation, review-gated). Heavy compute runs on Miner hardware in the Sidecar Mesh: a Go control plane for networking, committee duty, and block production, and a pure-Rust sidecar linking the pinned AIOS Module with no cgo, over a consensus-relevant, review-gated localhost gRPC or UDS contract. Figure 1 shows the boundary: the chain orders, verifies, prices, and settles, and heavy compute never runs in consensus.

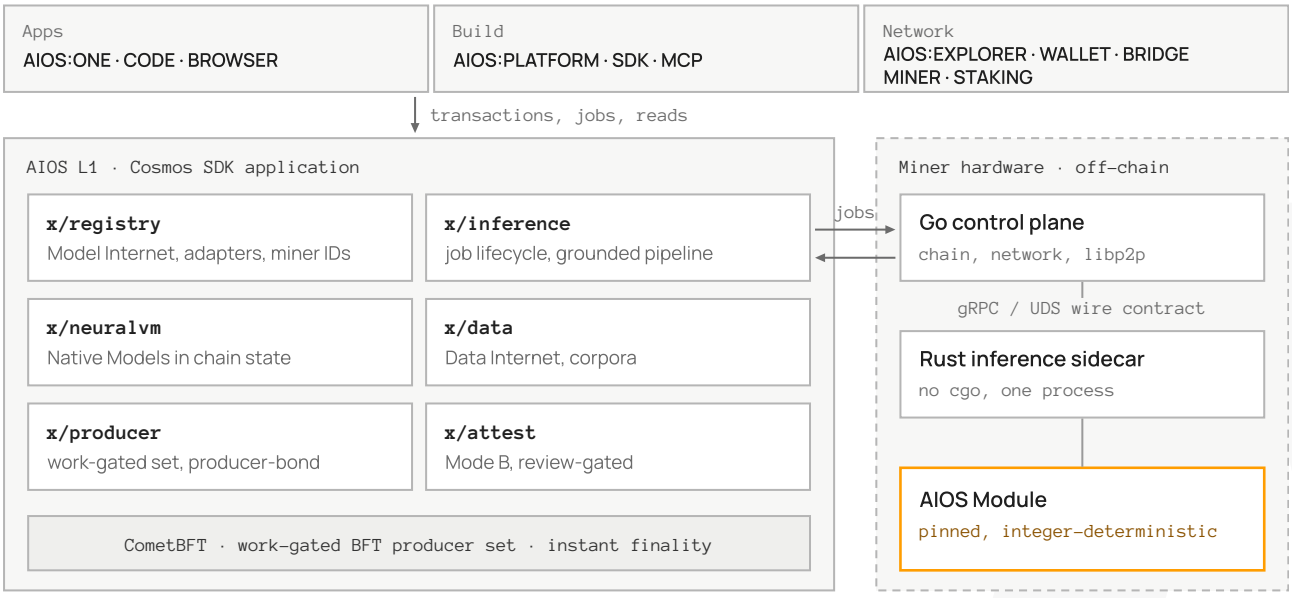


Figure 1. System architecture. The Cosmos SDK application holds six native modules above CometBFT. Heavy inference runs off-chain on Miner hardware through the Sidecar Mesh, whose pinned AIOS Module makes results byte-comparable across machines.

Callers submit jobs and escrow payment. Miners bring their own hardware, perform jobs, and sit on committees, never for their own jobs. Producers order blocks and post a producer-bond. Owners of models, adapters, and corpora set prices, earn fees, and post a creator-bond. A foundation multisig proposes upgrades and holds halt-only backstops; an external bridge provider has its own trust domain.

2.2 Trust assumptions

We assume collision-resistant hashing, unforgeable signatures, and an unbiasable beacon built from a verifiable random function [9] or threshold randomness [10]. Block ordering assumes Byzantine producers hold under one third of voting power. Proof of Inference assumes an honest majority of committed work per committee and, on the exact path, a byte-deterministic AIOS Module, an engineering target; penalties that depend on it stay disabled until it is validated (Section 4.3). No single Miner, performer, or owner is trusted.

2.3 Adversary

The adversary may rent compute transiently; register many identities [11], each costing one registration burn; collude out of band, including a performer with its committee; grind the beacon from a proposer seat; copy hashes as a lazy verifier; farm work credit with wash calls; withhold results after accepting a seat; and gain physical access to TEE hardware, which TEE.Fail [12] showed is enough to forge attestation quotes with equipment costing under 1,000 US dollars. Out of scope by design: the truth or bias of outputs and corpora, the relevance of retrieved context, and prompt injection (Section 5.2).

2.4 Security goals

- **Safety.** A wrong output hash becomes canonical only if the adversary holds more than two thirds of a committee's work weight.
- **Liveness.** A job clears, or is re-routed to a fresh or larger committee up to a bounded count and then refunded in full; disputes run in a hard-capped window.
- **Bounded exposure.** Caller funds are never at risk beyond a caller-owned, non-custodial escrow, whose unused remainder is refunded.
- **Supply integrity.** Supply never exceeds 300,000,000 and is conserved at every block.
- **No principal loss for honest divergence.** Nondeterministic divergence or honest downtime costs a Miner only that job's reward.

Chapter 3

Proof of Inference

This section defines the job lifecycle, the canonical-output rule, committee selection, and the defenses against capture and lazy verification.

3.1 Job lifecycle

A caller submits a model address, input, sampler configuration and seed, maximum price, and verification mode, and escrows a protocol-fixed worst case (maximum committee size and re-route escalation). The beacon seats a performer and a committee. Members check fetched weights against the registry content hash, closing weight substitution; the committee recomputes in full or samples openings (Section 4); every member commits a hash before any reveal; the tally decides. Figure 2 shows the only two exits: settlement to matching Miners, or a full refund. Deadlines turn non-response into re-route or refund plus a liveness penalty, so escrow cannot strand.

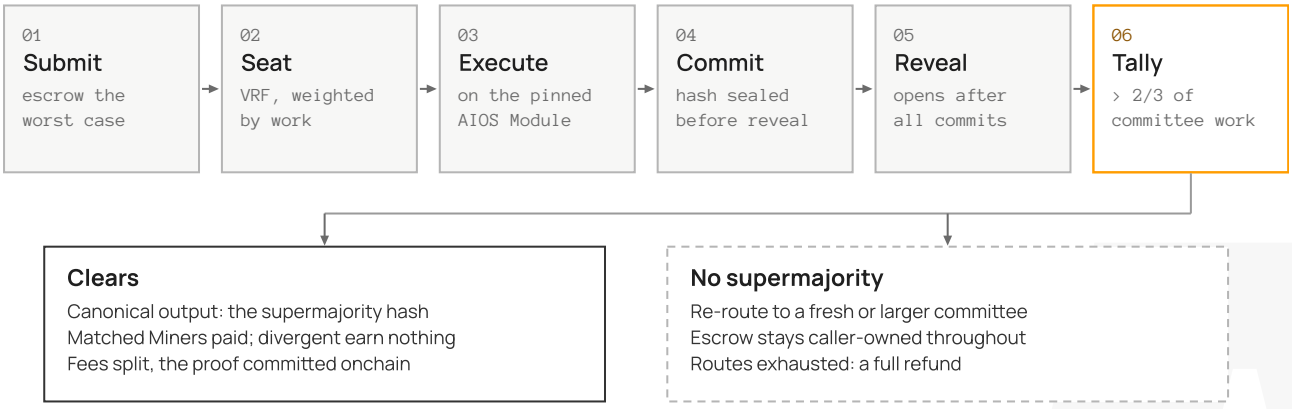


Figure 2. The Proof of Inference job lifecycle. Escrow is caller-owned throughout. A job either clears on a work-weighted supermajority, or it is re-routed, and a job that exhausts its routes is refunded in full.

3.2 The canonical output

Definition 1 (Committee and work weight). For a job J, a committee is a set C of Miners seated by the beacon. Each member $i \in C$ has a work weight $w_i > 0$ derived from its verified work history and commits a hash h_i of its output. Let $W = \sum_{i \in C} w_i$.

Definition 2 (Canonical output). A hash $h^\star$ is canonical for J if and only if

$$\sum_{i \in C\,:\,h_i = h^\star} w_i \;>\; \tfrac{2}{3}\,W.$$

If no hash qualifies, the job has no supermajority and is re-routed. The performer is paid only if its claim equals the canonical hash.

Proposition 1 (Uniqueness). At most one hash is canonical for a given committee.

Argument. Each member commits one hash, so two distinct hashes have disjoint supporting sets; if both exceeded $\frac{2}{3}W$, their total would exceed W.

Proposition 2 (Committee safety and liveness). Suppose honest members are byte-deterministic, so all commit the same correct hash, and the adversary controls weight aW. A wrong hash can be canonical only if $a > \frac{2}{3}$; the correct hash is canonical if $a < \frac{1}{3}$; in between, the adversary can at most prevent a supermajority.

Argument. A wrong hash gathers no honest weight, so it needs adversarial weight above $\frac{2}{3}W$, and honest weight $(1-a)W$ exceeds $\frac{2}{3}W$ exactly when $a < \frac{1}{3}$. The result is conditional on determinism: if honest outputs differ, honest weight splits and the bound fails, which is why penalties are gated on validated determinism (Section 4.3).

A middle-band coalition can force re-route and refund, a liveness cost, but not a wrong output, and blocking is not free: off-supermajority members are unpaid, and repeated mismatch leads to ejection.

3.3 Selection, weight, and capture

Committee seats and vote weight come from verified work, never capital: eligibility is gated by work history and beacon selection is weighted by work credit, which cannot be sold, lent, or delegated and never multiplies a payout. A proposer able to steer seating could pack a committee, so the beacon must be unbiasable [9, 10]; grinding resistance is an open benchmark.

Proposition 3 (Capture probability under equal weights). Let an adversary hold a fraction p of eligible work, let members carry equal weight, and let a committee of size k be drawn from a large population, so the adversary's seat count is approximately binomial. The probability that it holds a supermajority is

$$P_{\text{cap}}(k,p) \;=\; \sum_{j=\lfloor 2k/3 \rfloor + 1}^{k} \binom{k}{j}\, p^{j}(1-p)^{k-j}.$$

Argument. A supermajority needs at least $\lfloor 2k/3 \rfloor + 1$ equal seats, and the sum is the upper binomial tail. This is an analysis under stated simplifications: committee size is open, real weights are unequal, and per-entity caps bound any one entity's committee share.

Figure 3 plots this tail for $k \in \{5, 9, 15, 21\}$: at $p = 0.2$ it falls from about 6.7×10^{-3} at $k = 5$ to about 5.1×10^{-7} at $k = 21$ illustrative. The chart hides repetition: over n independent jobs, rented compute captures at least one committee with probability $1 - (1 - P_{\text{cap}})^n$, so committee size alone cannot secure high-value jobs. The value caps, burn floor, and per-entity caps of Section 6 do that work.

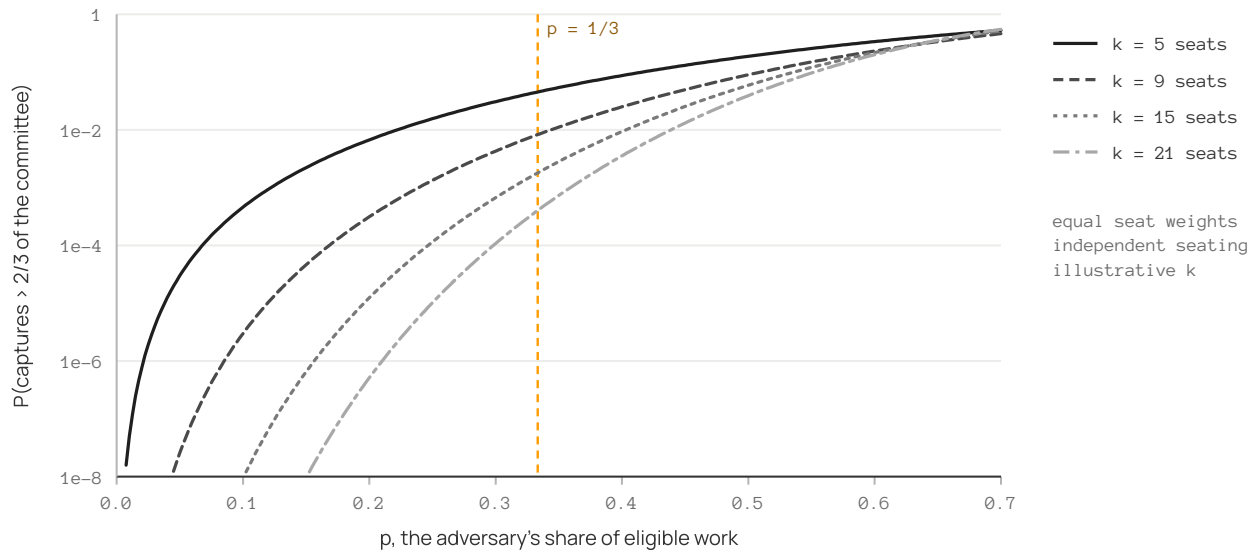


Figure 3. Probability that an adversary holding a share p of eligible work captures a greater-than-two-thirds committee supermajority, under a binomial model with equal seat weights and independent seating. Committee size is an open parameter; the sizes shown are illustrative. Capture falls steeply with k while p stays below one third.

3.4 Lazy verifiers, commit-then-reveal, and honeypots

A member who copies a claimed hash earns the reward at no cost; if enough do, the guarantee collapses, and reward on match sharpens this verifier's dilemma [13]. Commit-then-reveal stops members from copying each other's reveals. Mandatory honeypots at raised frequency carry a known-wrong performer claim whose ground truth the chain computes in the Neural VM; a member that agrees fails, is ejected immediately, and forfeits accrued-but-unpaid reward. Honeypots are funded from the protocol fee slice or the insurance pool, never from peers.

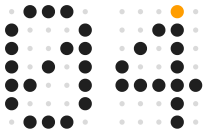
Proposition 4 (Audit survival). If each job a persistent cheater or lazy verifier takes is independently a honeypot or audit with probability h, and audits are indistinguishable from ordinary jobs, the probability of surviving m jobs undetected is $(1 - h)^m$, and jobs to first detection are geometric with mean $1/h$.

Argument. Independence gives the product and the geometric law. Indistinguishability is an assumption the honeypot design must earn; it is an open review item.

At $h = 0.05$, survival after 50 jobs is about 0.077; at $h = 0.01$ it is about 0.61 illustrative, so the rate is calibrated jointly with the burn floor and value cap. Oracle-free ground truth exists only for models the chain computes itself, so honeypots police the in-VM path directly. On the large-model path, commit-then-reveal cannot stop copying a performer's claim shared out of band; that risk is reduced, not closed, and bounded by per-entity caps, a minimum-honest-compute floor on high-value jobs, and value caps.

3.5 Rejected and deferred mechanisms

Proof of Quality, judge-model scoring, is rejected: under a Model Downgrade Attack an 8B adversarial model scored indistinguishably from an honest 70B model while true quality fell by about 40%. Zero-knowledge proof aggregation at consensus is deferred for lack of verified block-time proving evidence at 7B parameters and above; a zero-knowledge proof of inconsistency remains a last-resort dispute tool.



Chapter 4

Determinism and Verification Regimes

This section explains why determinism is the hard boundary of the design, what the AIOS Module pins, and how guarantees differ by model size.

4.1 The AIOS Module

The AIOS Module is one pinned Rust crate behind a deterministic FFI boundary. Matrix multiplication, softmax, and layer normalization run in pure-integer fixed-point arithmetic with batch-invariant kernels and a fixed reduction order; no floating-point operation, uninitialized memory, or platform intrinsic crosses the boundary. The same version runs in the Neural VM and every Miner client, so the version is consensus state; the Module also pins integer embedding, exact-retrieval, and adapter kernels. Decoding is greedy, or the sampler configuration and seed are committed in the job, at an openly accepted cost in output variety. Integer-only inference has peer-reviewed support [2, 3, 4, 5]; cross-GPU byte identity of AIOS's kernels does not yet.

4.2 Two regimes, three paths

Exact agreement is affordable only when recomputing the whole job is cheap. Table 1 separates the three paths.

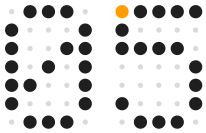
Table 1. Verification paths and their guarantees.

Path	Used for	Guarantee	Consequence of a wrong result
Full recomputation on the AIOS Module	Native and small models	Byte-identical certainty	Objective no-pay; ejection only on proven fraud or a failed honeypot
Sampled recomputation	Larger models on the Module	Statistical, set by the sampling rate	Loser-pays dispute before any penalty
Seed-synchronized auditing (DiFR-style)	Models that cannot run the Module	Statistical divergence from a trusted reference, never exact hash	Dispute-then-penalty only

Under sampled recomputation the performer commits intermediate states under a Merkle root [14] and the committee recomputes beacon-chosen openings, so cheating confined to unsampled regions escapes with bounded but nonzero probability. Larger models are the actual inference product, so the product path runs on statistical assurance, stated wherever it is described. Every regime establishes only that the registered integer-quantized weights executed correctly; quantization changes behavior, so a caller gets a verified quantized model, not a verified copy of a named frontier model.

4.3 Determinism-gated penalties

Objective consequences beyond no-pay apply only on paths with validated byte determinism. The gate is a differential harness comparing a CPU reference order with each GPU backend, pinning accumulator width, overflow policy, and reduction order per kernel as consensus state. Until it passes on the real kernel set (inference, embedding, retrieval, and adapter kernels), every path resolves by dispute-then-penalty, and an honest Miner on divergent hardware is not paid for that job: a determinism bug costs forgone reward, never principal. Per-SKU bit-exact emulation of accelerator arithmetic [15] is a watch item as a future dispute mechanism.



Chapter 5

Verification Modes and Assurance Classes

This section defines the two verification modes, the four assurance classes a caller sees, and the claims AIOS does not make.

5.1 Mode A and Mode B

Mode A, committee consensus, is public, vendor-independent, the flagship, and the only path ever called verified. **Mode B** runs a job in a trusted execution environment and proves confidential, attested execution: registered code ran untampered in a genuine enclave on inputs the operator never saw. It does not prove the arithmetic; its trust root is the chip vendors plus a value cap, not AIOS. The modes are mutually exclusive per job, because re-computation needs plaintext: public verified inference or private attested inference, never both.

Mode B settles on an M-of-N quorum of attestations from several independent vendors, checked by `x/attest`. A single-vendor root is prohibited, value caps bind at submit, faults resolve by dispute-then-penalty, and Mode B never secures consensus. Its sub-modes are `tee-fastpath` (a deterministic model back-checked by sampled recomputation), `tee-private` (confidential and hard value-capped; the floating-point case awaits a comparison rule), and `tee-attested` (deferred). Mode B is review-gated and sequenced after Mode A. TEE hardware is a centralized premium minority, and Mode B is never called decentralized or verified.

5.2 Assurance classes

> ***Definition 3 (Verified).*** An inference is *verified* if and only if it cleared Mode-A committee consensus under Definition 2. Verified means provenance of execution: the registered weights, and where bound the registered adapter and corpus, produced this output under the specified program. It never means the output is true, correct about the world, safe, or unbiased.

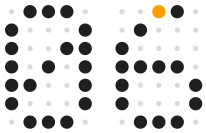
Every result carries one of four assurance classes, stamped into its verification record; weaker assurance prices lower. Figure 4 orders them with their trust roots; a caller should read the class before the output.

Assurance	Class and path	What it proves	Trust root
	verified Mode A · committee consensus	provenance of execution, recomputed independently	AIOS's own recomputation; public, trustless
	sampling Sampled recomputation	statistical assurance at the audit sampling rate	the sampling rate and loser-pays disputes
	attested Mode B · TEE	confidential attested execution, not correctness	chip vendors, M-of-N quorum, a value cap
	self-verifiable Private memory	the memory version and that a recall occurred	the owner's keys; the owner can recompute

Figure 4. The four assurance classes. Only the Mode-A committee-consensus path is labelled verified; each other class states a narrower property and rests on a different trust root.

- `verified` : Mode-A committee consensus, rooted in independent recomputation.
- `sampling` : statistical assurance from sampled recomputation or seed-synchronized auditing, never described as certain.
- `attested` : Mode-B execution, rooted in hardware vendors plus the value cap.
- `self-verifiable` : the private-memory path (Section 9.4). The network guarantees the memory-version commitment and that a recall occurred; the key-holding owner can recompute it; the network does not attest correctness, and no third party can verify it.

The schema partitions these labels, so a floating-point or confidential path can never carry a Mode-A class, and a session's assurance is the floor of its jobs, never an average. Image, video, and audio generation carry provenance and billing only. Verified inference does not solve prompt injection or agent and browser security, which remain separate and unsolved. A registered model is not an endorsed one.



Chapter 6

Mining and the Registration Burn

This section describes how Miners earn, the one-time burn that replaces stake on the mining layer, and the conditions under which that anchor is securable.

6.1 Reward on match

Inference Mining is performing and recomputing verified inference for reward, with no stake, no per-job bond, and no token slashing. A matching Miner receives, in one settlement, a fee share (Section 10.3) and a per-job bootstrap emission from the fixed mining pool of 100,000,000 $AIOS `locked`, decaying as fees ramp; a mismatching Miner earns nothing and loses its compute. Settlement is per job and solo, with no pools and no delegation, which removes a Sybil amplifier without making the network collusion-resistant. Miners fund their own commodity GPU or CPU hardware, with TEE hardware optional; AIOS runs no paid GPU cloud. A zero-install WebGPU browser client, beside desktop and CLI clients, lowers the install barrier but never the determinism bar. Miners also serve corpora (Section 9), and liveness is proven apart from correctness, by resource challenges with strike penalties.

6.2 The registration burn

At mainnet, entry to mining costs a one-time registration burn, 10,000 $AIOS at genesis `locked`, paid in native $AIOS on the L1 and destroyed on payment: never held, returned, or repurposable, and never a bond. It buys only a persistent miner ID with beacon eligibility, conferring zero work credit, committee weight, vote weight, or producer eligibility. Miners fund it by bridging the Phase-1 ERC-20, so every Miner needs a wallet.

> ***Definition 4 (Registration cost).*** Let $e \in [0,1]$ be the fraction of the 100,000,000-token mining pool emitted, R_0 the genesis cost, F a floor, Cap a ceiling, $\gamma > 0$ a curvature, and M a demand multiplier bounded by $M_{\min}$ and $M_{\max}$. Then

$$B(e) = F + (R_0 - F)\,(1-e)^{\gamma}, \qquad \mathrm{cost} = \mathrm{clamp}\big(B(e)\cdot M,\ F,\ \mathrm{Cap}\big),$$

and each controller epoch the multiplier tracks the registration rate r against a target $r^{\star}$:

$$M \leftarrow \mathrm{clamp}\big(M \cdot r/r^{\star},\ M_{\min},\ M_{\max}\big).$$

The baseline runs from $B(0) = R_0$ to $B(1) = F$, monotone in e when $R_0 \geq F$. Only R_0 and the shape are operator-set; F, γ, $r^\star$, and Cap are open. Because e tracks realized, fee-conditioned emission, including the producers' share, cost glides with network maturity rather than calendar or node count, and M prices out bursts like a difficulty adjustment; the computation is pinned fixed-point consensus state. Figure 5 shows that a larger γ cheapens entry sooner, a smaller one keeps it dear longer, and all curves end at the floor.

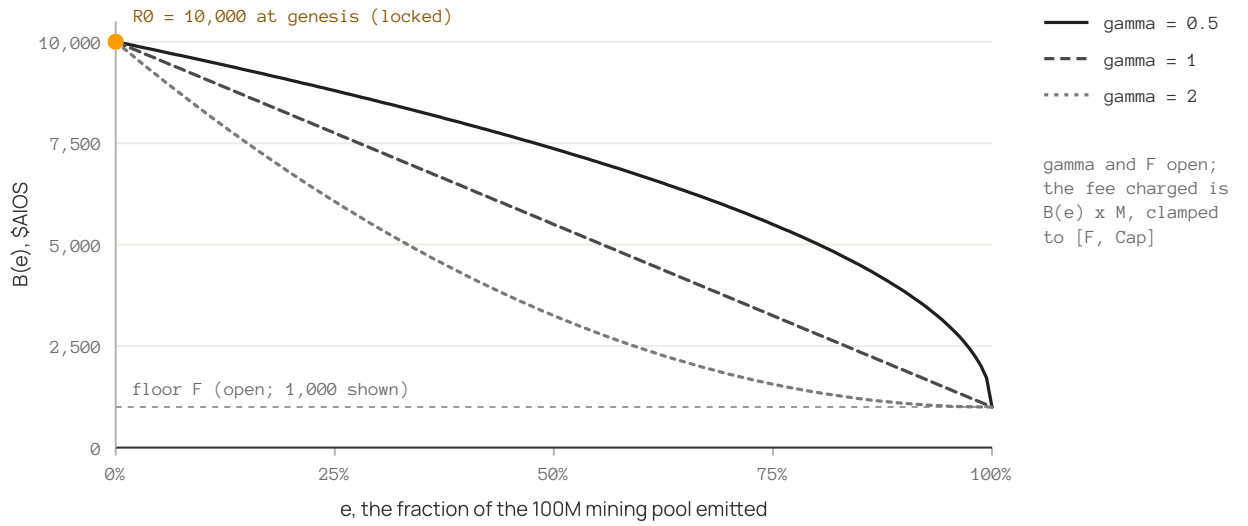


Figure 5. The registration-burn baseline $B(e) = F + (R0 - F)(1 - e)^{\gamma}$, where e is the fraction of the mining pool already emitted. R0 = 10,000 $AIOS is locked; F and γ are open, and the floor is drawn at an illustrative 1,000.

The active set is soft-capped: registering into a full set deregisters the Miner with the lowest sustained verified work after an immunity period, chosen deterministically without discretion, and it loses its sunk burn. This is competitive eviction, not a slash, and a deliberate exception to the rule that honest behavior never loses value.

6.3 Security conditions

The burn prices Sybil-fleet entry, since N identities cost N burns, but not per-committee participation: one identity sits on many committees for one burn, and a per-committee supermajority can be rented at electricity cost, repeatedly. Pure zero-cost mining is not securable; the security review found burn-only mining securable with conditions, all load-bearing:

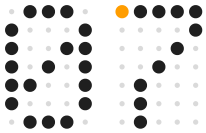
1. **Burn floor as a security floor.** F must exceed the dishonest payday of the largest job a Miner can be seated on before detection, or such jobs are value-capped below the F-equivalent.
2. **Value caps at submit**, bounded relative to F and expected jobs to detection (Proposition 4); a job above the cap needs a per-job bond or a bonded-mining tier.
3. **Raised honeypot frequency**, funded from the protocol fee slice or insurance pool.

4. **Per-entity influence caps and a minimum-honest-compute floor** across every Miner ID one entity controls, now the main per-committee-capture control.
5. **A longer decaying genesis allowlist** for high-value seating, tied to a higher honest-compute floor.
6. **Recalibration**: cost-anchor and eviction benchmarks re-run at the lower per-job cost basis before any allowlist decay, since earlier calibrations assumed a per-job bond; a mainnet blocker.

Forfeited value is destroyed, never paid to peers, which would reward manufactured disagreement. If burn-only mining proves insufficient, the fallback is a bonded-mining tier requiring per-job bonds on high-value jobs, buildable without redesigning the base layer.

6.4 Consequences

No consequence is a token slash. Mismatch on the exact path is objective no-pay. Proven fraud or a failed honeypot brings immediate ejection and forfeiture of accrued-but-unpaid reward, and re-entry needs a fresh burn. Ejection is objective only on a validated byte-deterministic full-recomputation path; elsewhere it follows dispute-then-penalty, escalating through a loser-pays challenge, a zero-knowledge proof of inconsistency where feasible, and expanded full recomputation. Sustained liveness griefing decays selection weight and escalates to ejection. Determinism divergence and honest downtime never eject and never forfeit the burn.



Chapter 7

Block Ordering

This section describes how blocks are ordered without proof of stake, and states plainly where a stake remains.

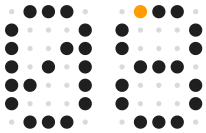
AIOS runs the CometBFT engine verbatim, with proposer rounds, precommits, and instant finality [8, 16]; only admission changes. A node joins the epoched, bounded producer set by accruing work credit above a floor and posting a standing producer-bond, and neither alone suffices. Ordering weight is fixed per epoch, and each height's proposer is chosen by the beacon. With v_j the ordering weight of producer j, $\mathcal{P}$ the producer set, and $\mathcal{B} \subseteq \mathcal{P}$ the Byzantine producers, safety holds while

$$\sum_{j \in \mathcal{B}} v_j \;<\; \tfrac{1}{3} \sum_{j \in \mathcal{P}} v_j,$$

the $f < n/3$ bound over weighted power rather than headcount; a minimum set size is open.

The producer-bond is slashed only on equivocation, the one self-proving ordering fault; its unbonding delay exceeds the evidence window, and it earns nothing. Producers receive a work-gated block reward of gas, a fee share, and bootstrap emission from the same mining pool, never a yield, and committee credit alone never seats a producer.

The bond is standing, slashable capital, economically a small stake: "stakeless" holds for the mining layer only, and "zero stake anywhere" is false. The set starts from a disclosed, decaying genesis allowlist and, like the mining allowlist, stays federated until three mainnet-blocker benchmarks close: cost anchoring of work credit against wash calls and self-farming, a bound on patient multi-epoch rental, and beacon-grinding resistance. Ordering weight resists spot rental within an epoch only, and since work credit admits producers, wash-call farming threatens ledger safety.



Chapter 8

The Model Internet, Native Models, and Adapters

This section describes the model registry, the models that live in chain state, and adapters as owned specializations of a shared base.

8.1 The Model Internet

The Model Internet is the registry and billing layer in `x/registry`. Table 2 lists its record for each model.

Table 2. The Model Internet record.

Field	Content
Address	Onchain identity (`aios1…`)
Weights content hash and pointer	Checked by committees against fetched weights
Price	Per call, set by the owner, by assurance class
Verification mode	An allowed set; the caller picks per job
Assurance class	Machine-readable, caller-visible, mandatory
Owner and creator-bond	Who controls and earns, and the bond behind it

Proof of Inference proves that registered weights ran correctly, not that they are the model the label names. The creator-bond puts a vouching party behind the label, but no fraud arbiter ships at launch: the bond is a registration cost and spam deterrent, misrepresentation slashing is deferred until an arbitration authority exists, and the label gap is only partly closed. Prices are set in $AIOS with no oracle, so owners re-price as the token moves.

8.2 Native Models and the Neural VM

Native Models are integer-quantized models in chain state, executed in consensus by the Neural VM; the onchain reference is a ternary BitNet-class model [17]. The binding constraints are state size and per-block compute: even at 4 bits a 7B model is about 3.5 GB and a 70B model about 35 GB. Figure 6 shows why ternary weights matter: at 1.58 bits per

weight a 2-3B model fits under a gigabyte, and the credible Native Model ceiling is a genuinely useful natively trained ternary LLM of about 2-3B parameters, a class shown benchmark-competitive at its size [18].

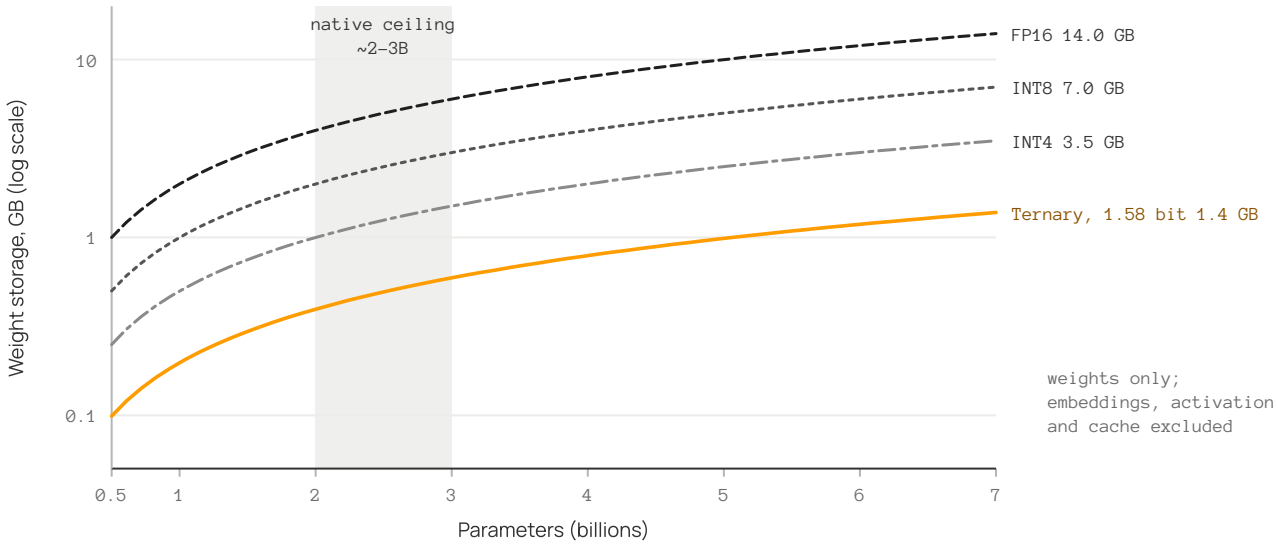


Figure 6. Weight storage by numeric precision. Ternary weights at 1.58 bits make a 2-3B model a sub-gigabyte object, which is where the native ceiling sits; the binding limits are chain-state size and per-block compute, not model quality.

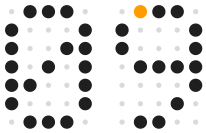
Frontier LLMs cannot run in consensus, ever: their floating-point arithmetic does not reproduce across hardware, and their size exceeds any sane state budget. Larger models run off-chain on Miner hardware, exactly verifiable only on the pinned AIOS Module and otherwise statistically, never by exact hash.

8.3 The Adapter Registry

An adapter is a LoRA-class fine-tune [6], a low-rank delta on a base model; a rank-8 adapter on a 2-3B base is about 2-11 MB. The Adapter Registry, a locked draft of 2026-08-23, registers adapters in `x/registry` with an address, owner, price, delta content hash, a `base_model_hash` binding (an unregistered base is rejected), and the same creator-bond class as models and corpora; small ranks may live in chain state.

A job may bind a base, an adapter, and a query; the committee recomputes the composition and commits a hash over the output and every bound commitment, so a different adapter falls off the supermajority. The verified class requires an integer or quantized adapter, pinned in the Module and served un-merged to stay swappable; floating-point adapters are attested or statistical only. A private adapter is the weights analogue of Verified Memory, self-verifiable by its owner; published, it earns an adapter-royalty on every call.

An adapter on a 2-3B base is still a 2-3B model; LoRA adds no scale or knowledge. The determinism gain needs a pinned integer adapter kernel, validated before mainnet. Stacking, routing, and merging are labeled products, not guarantees: a merge's determinism is enforceable, its quality is not.



Chapter 9

The Data Internet and Verified Memory

This section extends Proof of Inference to the retrieval step of grounded generation and describes the private form of the same machinery.

9.1 Verifiable retrieval

Most model calls are retrieval-augmented: the model is grounded on context from a corpus [7]. A corpus registers through `x/data` with an address and owner, a Merkle commitment [14] over its chunks and integer embeddings, a retrieval pointer to content-addressed storage or pinning Miners, a pinned retrieval specification (embedding model, k, index type, metric, tie-break rule), a price, a verification mode, and a creator-bond of the same single bond class. Small Native Corpora may live in chain state.

A grounded job binds a model, a corpus, and a query. Each seated Miner embeds the tokenized query with integer embeddings, runs exact k-nearest-neighbor search by integer inner product, or BM25 [19], over the committed index, takes the top chunks with ties broken by content hash, and generates on that context. It commits a hash over the output and the retrieval manifest, the ordered chunk hashes used as context, and Definition 2 applies unchanged. Figure 7 shows retrieval inside the same commit-reveal round: different chunks fall off the supermajority exactly as different text does.

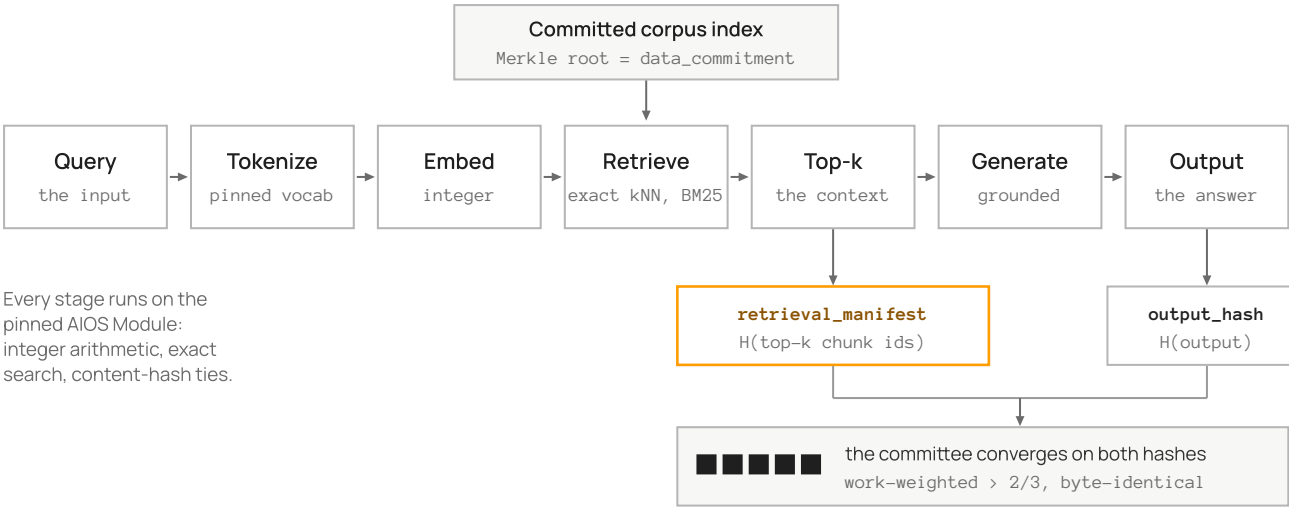


Figure 7. Verifiable retrieval. Every stage of the grounded pipeline runs on the pinned AIOS Module, so the committee can converge on the retrieval-manifest hash as well as the output hash: the context is proven, not only the model.

Approximate indexes such as HNSW and IVF are excluded, since their results depend on build order, threads, and hardware. Floating-point or approximate retrieval is verified by sampled recomputation, never as exact hash; a private corpus may retrieve in a TEE, attested and never called verified. A corpus the committee cannot fetch and check by the deadline triggers a refund and a creator-bond penalty.

9.2 The proof

A cleared job produces the proof, a verification record committed onchain; Table 3 lists its fields.

Table 3. Fields of the proof record.

Field	Binds
`model_commitment`	The registered weights content hash
`adapter_commitment`	The adapter hash and its `base_model_hash`, if composed
`data_commitment`	The corpus Merkle root, for grounded jobs
`retrieval_manifest`	The ordered top-k chunk hashes used as context
`input_hash`, `output_hash`	The input and the canonical output
Committee, tally, class, height	Who recomputed, the supermajority, the assurance class, and the block

Together these records form a provenance graph tracing any output to its base, adapter, corpus, and exact chunks; provable usage makes it the basis for royalties.

9.3 Verified Memory

Verified Memory is `x/data` with a visibility flag, not a new module: a corpus is public (listed, priced, committee-verified, royalty-earning) or private (owner-scoped, encrypted, unlisted). Long-term memory is chunked, embedded, and indexed as a personal vector store, append-only and Merkle-committed so its state at any height is provable, encrypted under the user's keys, and portable, exportable, and revocable; the network holds commitments and encrypted pointers, never plaintext. User-defined session scopes are both retrieval and permission boundaries, with scoped, capped, revocable access.

Private memory is self-verifiable: the key-holding owner can recompute a recall, the strongest proof for private data, and a TEE can optionally attest it. It is never committee-verified, since that would expose it; shared memory is committee-verified and earns a data-royalty. Storage carries a small pinning cost AIOS never subsidizes, and recalling one's own memory costs the inference fee only. Verified Memory powers the Brain of AIOS:ONE, the AIOS:RECALL workflow, and verifiable agent continuity.

9.4 Limits

Verified data means provenance and integrity, never that the data is true, clean, current, or unbiased. Retrieval integrity is not relevance: consensus proves Miners retrieved the same chunks, not the best ones. Deterministic retrieval is unproven engineering, and until its harness passes the Data Internet serves statistical verification only. Data is the context models execute on; AIOS is not a data-availability or blob-storage layer, and no external oracle or live feed sits on the verified path.

Chapter 10

The $AIOS Token

This section specifies supply and conservation, distribution, fees, the transfer tax, staking, and governance.

10.1

Supply and conservation

The supply is fixed at 300,000,000 $AIOS `locked`, minted once at the Phase-1 ERC-20 deployment on Ethereum mainnet. The ERC-20 has no mint function, is non-upgradeable, and is a permanent home, never deprecated or burned. After L1 genesis the AIOS L1 is the sole mint authority: bridging in locks ERC-20 $AIOS and mints native $AIOS against the finalized lock, and bridging out burns native $AIOS and unlocks the ERC-20. Figure 8 shows the single issuer, an audited verifier on the L1 that mints only against a finalized, unconsumed lock of equal amount.

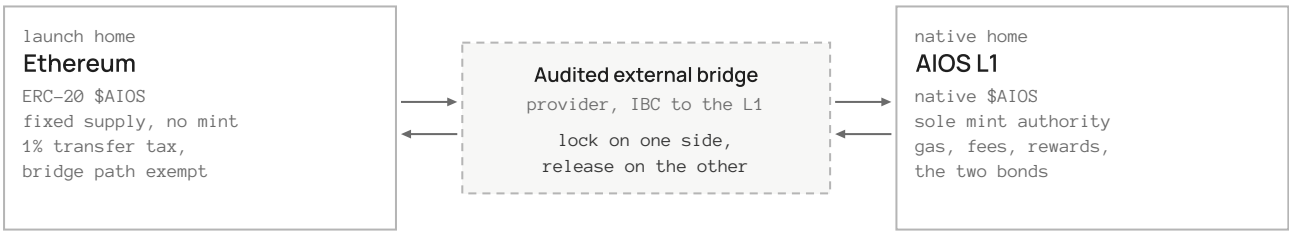


Figure 8. The dual-home supply. The ERC-20 on Ethereum and native $AIOS on the L1 are one asset: an audited external provider locks on one side and releases on the other, and the conservation invariant is checked every block.

Proposition 5 (Conservation). If every native mint is bound to a finalized, unconsumed lock of equal amount, every unlock to a finalized native burn of equal amount, and every other native burn is permanent, then at every block

$$\text{eth\_circulating} + \text{native\_circulating} + \text{cumulative\_burned} = 300{,}000{,}000.$$

> *Argument.* The fixed ERC-20 supply splits into circulating and locked balances. A mint moves a quantity into the lock and adds it to native circulation, a return reverses both, and a fee or registration burn moves native circulation into the burned total; each preserves the sum. Equivalently, the lock equals native circulation plus cumulative burns, and circulating supply is $300{,}000{,}000 - \text{cumulative\_burned}$, a ceiling that only falls.

The invariant is checked every block; an upward divergence halts the bridge and is the primary intrusion signal. AIOS builds no bridge cryptography: an audited external provider handles the cross-VM Ethereum leg, and other Cosmos chains connect by IBC. Outbound mints wait for Ethereum finality, caps gate both legs, and the bridge multisig can halt but never mint. Single-asset scope reduces bridge risk without eliminating it, and a permanent dual home is more standing attack surface than a migrate-once design. The bridge is secured by the provider's trust set; "safe," "trustless," and "secured by AIOS" do not describe it.

10.2 Distribution

Genesis distribution `locked` is 150,000,000 to liquidity (50%), 100,000,000 to the mining pool (33.33%), 30,000,000 to the staking pool (10%), and 20,000,000 to the team (6.67%), vesting over a 12-month cliff and 36 months linear `locked`. Figure 9 shows half the supply seeding liquidity and a third reserved for verified work, with no treasury or ecosystem bucket: the operator self-funds. The mining pool funds bootstrap rewards for Miners and producers from one carve-out, released against verified work on a fee-conditioned decay; the two sinks compete for one finite pool, and neither is staking yield.

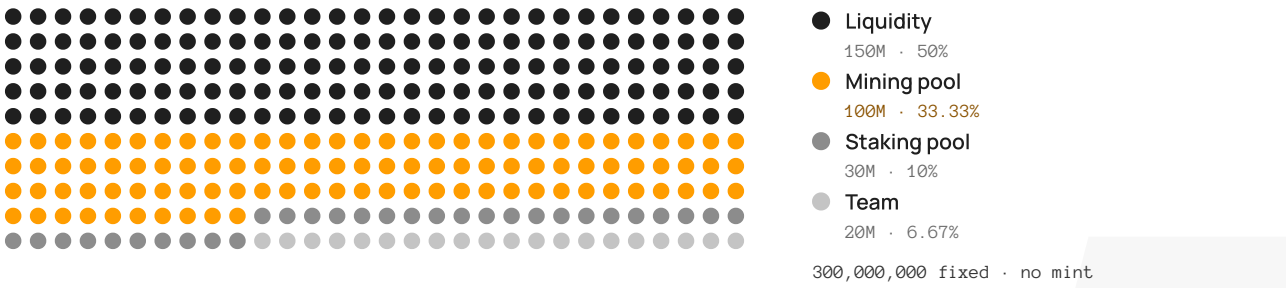


Figure 9. Genesis distribution of the fixed 300,000,000 $AIOS supply, one dot per million. There is no mint function and no treasury allocation.

10.3 Fees, royalties, and value

A call costs the owner's price plus a protocol base fee, never more than the escrowed worst case. The fee splits approximately 75% to matching Miners, 15% to the model creator, and 10% to the protocol `illustrative`; the protocol slice is mostly burned and also funds the producer fee share and the insurance and honeypot pool. Grounded calls add a corpus price and a data-royalty to the corpus owner, and calls composing a published adapter add

an adapter-royalty. Both royalties are additive, funded by the higher price of such calls and never carved from the Miner share, and are fee transfers, never mints. Data-serving rewards are fee-funded, not drawn from the mining pool.

$AIOS has seven roles: gas, inference payment, mining reward, block-producer reward, data-royalty, adapter-royalty, and ownership of the asset, one's models, adapters, corpora, and keys. Value accrues through usage (fees, gas, royalties, and burns); no security lockup provides a demand floor if usage does not arrive.

10.4 Transfer tax and staking

Ethereum ERC-20 transfers carry a flat 1% transfer tax `locked` with an immutable 1% ceiling; the rate can only move down, toward zero. The LP pair, router, bridge lock contract and relayer hops, staking pool, and tax-collector sink are exempt, and tax accrues in $AIOS, swept to a multisig distinct from the deployer and admin key.

AIOS has no proof-of-stake base and no staking for security. AIOS:STAKING is a non-security lock-to-earn pool funded by the 30,000,000 staking bucket through a front-loaded, decaying emission that ends. Its rate floats inversely with the total staked; an early rate of about 10% is illustrative and never guaranteed. Unstaking enters a 7-day cooldown without rewards, an anti-dump choice, not a security requirement. Principal is always eventually withdrawable, never slashed or paused, and staking secures nothing.

10.5 Governance

AIOS has no token voting and no DAO. A foundation multisig publishes a versioned upgrade with a migration height, and producers adopt it by running the new version at that height or fork off; the Module content hash and consensus parameters change only this way. Decentralization means broadening who must adopt, conditional on the Section 7 blockers. At launch the multisig and a halt-only forfeit backstop are operator-controlled, a disclosed centralization, and the upgrade path is review-gated before first use.

10.6 Parameter status

Table 4 separates fixed values from those awaiting measured costs and security-budget calibration.

Table 4. Parameter status.

Parameter	Value	Status
Total supply	300,000,000, no mint function	Locked
Genesis distribution	150M / 100M / 30M / 20M	Locked
Team vesting	12-month cliff, 36-month linear	Locked

Parameter	Value	Status
Ethereum transfer tax	1% flat, immutable ceiling, lower-only	Locked
Registration burn at genesis and its shape	10,000; glide to floor, multiplier, clamp	Locked
Staking cooldown	7 days	Locked
Supermajority threshold	Over 2/3 of committee work weight	Canon target
Fee split; early staking rate	~75 / 15 / 10; ~10%, floating	Illustrative
Burn floor, curvature, target rate, cap	Not set	Open
Committee size, sampling and honeypot rates, value cap, bonds, emission, pool split, re-routes, dispute window, bridge caps	Not set	Open

Chapter 11

Products and Roadmap

This section lists the eleven products that package the four pillars and the evidence gates between phases.

11.1 Products

The pillars ship as eleven `AIOS:` products in three families. **Apps:** AIOS:ONE (an assistant with Brain and Agent toggles and user-defined session scopes), AIOS:CODE (a CLI-first coding agent that marks steps provisional until the proof lands), and AIOS:BROWSER (a full agentic browser). **Build:** AIOS:PLATFORM (console and REST API: wallet sign-in, non-custodial prepaid balance, scoped revocable keys, corpus publishing), AIOS:SDK, and AIOS:MCP (an agent doorway with a verification record per call). **Network:** AIOS:EXPLORER (jobs, proofs, Miner standing, models, corpora, provenance graph), AIOS:BRIDGE, AIOS:WALLET (non-custodial), AIOS:MINER (node software that mines, can run a Producer, and serves corpora), and AIOS:STAKING. AIOS never custodies user funds or owners' keys, and never pays participants' compute bills.

11.2 Phases

Figure 10 shows three phases gated on evidence rather than dates.

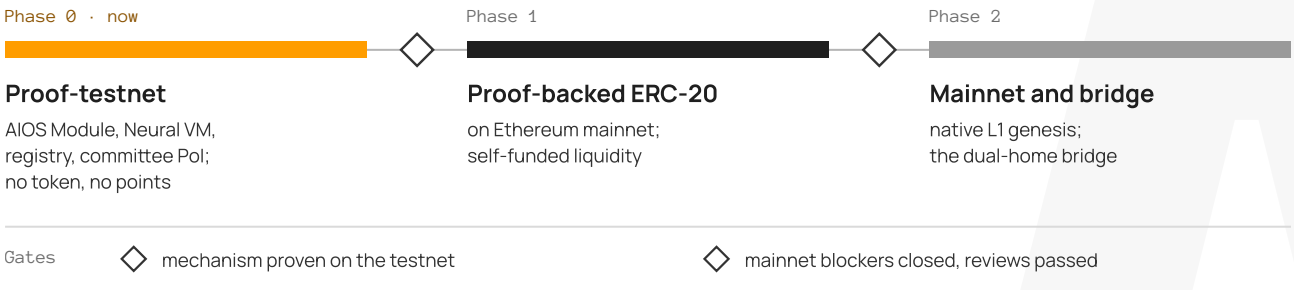


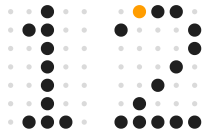
Figure 10. Proof-first sequencing. Each phase opens only when the gate before it closes; the gates are engineering and security benchmarks, not calendar dates.

Phase 0, proof-testnet. The chain comes first, on operator-run and allowlisted nodes, with no token and no points program; registration there is free. The first milestone demonstrates that deterministic integer inference yields byte-identical output hashes across ma-

chines on a pinned AIOS Module prototype, with a negative test that catches a perturbed instance; it graduates on a cross-machine byte-identical recomputation log. The testnet then adds the Neural VM and reference model, the Model and Data Internets, committee Proof of Inference, and Miner software.

Phase 1, proof-backed token. $AIOS launches as an ERC-20 on Ethereum mainnet with the testnet live as evidence: 300,000,000 fixed, no mint function, a 1% transfer tax, a self-funded liquidity pool on a decentralized exchange, and a minimal site leading with the mechanism. Proof of mechanism is not proof of demand.

Phase 2, mainnet and bridge. L1 genesis waits on the determinism harnesses passing on the real kernel set and on adversarial soundness of the verification layer. The dual-home bridge then opens, and the ERC-20 stays live indefinitely.



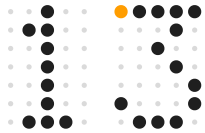
Chapter 12

Risks and Limitations

This section states, openly, the bets the design rests on.

1. **Proof of Inference is unproven at scale.** Committee recomputation is the most-proven candidate, but its closest demonstration is a single-developer testnet whose throughput numbers failed independent verification. Throughput and the model-size ceiling at target scale are open benchmarks, and cross-GPU byte determinism on AIOS's own kernels is the pre-mainnet gate.
2. **Frontier LLMs cannot run in consensus, ever.** Native Models top out near a 2-3B natively trained ternary model, floating-point models verify statistically, and native ternary models at 7B and above are unshipped.
3. **Deterministic retrieval is unproven.** Float embeddings and approximate indexes are nondeterministic by construction; what cannot meet the exact path falls to statistical verification.
4. **Stakeless mining is weaker than staking.** The burn prices entry, not per-committee participation, and is weaker on single high-value jobs than a per-job bond. The bootstrap-compute cliff is acute at genesis, when honest compute is small, rentable, and unpriced by any staked market capitalization. Value caps, the burn floor, honeypots, per-entity caps, and the allowlist bound this without closing it; the fallback is bonded mining.
5. **Mode B shifts the trust root to chip vendors.** Attestation proves confidential execution, not correctness; TEE.Fail forged quotes cheaply with physical access [12]; confidential jobs have no correctness backstop.
6. **Inputs are visible to the committee.** The market for verified inference is callers comfortable with committee-visible inputs.
7. **Some large-model fraud may be unpunishable.** Without a feasible zero-knowledge proof or affordable full recomputation, a dispute ends in no penalty plus refund.
8. **Token-first bleed is mitigated, not eliminated.** The token trades before native fee revenue exists, and willingness to pay a premium for verified inference over a cheaper unverified API is the core commercial bet.

9. **The model-registry space is partly claimed.** Onchain model registries exist; the wedge is the verification binding, not priority.
10. **Bridge risk is permanent and inherits the producer set.** A captured producer set, or while federated a misbehaving one, could finalize a native burn the bridge treats as irreversible.
11. **The operating runway is narrow.** With no treasury, operations depend on operator funds and transfer-tax revenue.



Chapter 13

Conclusion

This section summarizes the construction and its boundary.

AIOS makes inference a consensus object: a committee weighted by verified work recomputes each job on a pinned integer runtime, commits before revealing, and settles on more than two thirds of its work weight, proving retrieved context and composed adapters in the same round. The guarantee is provenance of execution: exact for small integer models, statistical for larger ones, attested for confidential jobs, self-verifiable for private memory. Mining rests on a burned registration fee rather than stake; ordering carries one small stake, the producer-bond. Deterministic by construction; verified by committee.

Back matter

References

[1] IEEE. *IEEE Standard for Floating-Point Arithmetic*. IEEE Std 754-2019.

[2] Lin, Y., et al. Towards Fully 8-bit Integer Inference for the Transformer Model. IJCAI 2020.

[3] Li, Z., and Gu, Q. I-ViT: Integer-only Quantization for Efficient Vision Transformer Inference. ICCV 2023.

[4] Kim, S., et al. I-BERT: Integer-only BERT Quantization. ICML 2021.

[5] Jacob, B., et al. Quantization and Training of Neural Networks for Efficient Integer-Arithmetic-Only Inference. CVPR 2018.

[6] Hu, E. J., et al. LoRA: Low-Rank Adaptation of Large Language Models. ICLR 2022.

[7] Lewis, P., et al. Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks. NeurIPS 2020.

[8] Buchman, E., Kwon, J., and Milosevic, Z. The latest gossip on BFT consensus. arXiv:1807.04938, 2018.

[9] Micali, S., Rabin, M., and Vadhan, S. Verifiable Random Functions. FOCS 1999.

[10] Cachin, C., et al. Random Oracles in Constantinople: Practical Asynchronous Byzantine Agreement Using Cryptography. *Journal of Cryptology* 18(3), 2005.

[11] Douceur, J. R. The Sybil Attack. IPTPS 2002.

[12] TEE.Fail. Public disclosure, October 2025: forged TEE attestation quotes with physical access, as cited in the AIOS canon.

[13] Luu, L., et al. Demystifying Incentives in the Consensus Computer. ACM CCS 2015.

[14] Merkle, R. C. A Digital Signature Based on a Conventional Encryption Function. Advances in Cryptology (Crypto), 1987.

[15] arXiv:2606.00279, 2026. Per-SKU bit-exact emulation of accelerator arithmetic, cited in the AIOS canon as a watch item.

[16] Castro, M., and Liskov, B. Practical Byzantine Fault Tolerance. OSDI 1999.

[17] Ma, S., et al. The Era of 1-bit LLMs: All Large Language Models are in 1.58 Bits. arXiv:2402.17764, 2024.

[18] Ma, S., et al. BitNet b1.58 2B4T Technical Report. arXiv preprint, 2025.

[19] Robertson, S., and Zaragoza, H. The Probabilistic Relevance Framework: BM25 and Beyond. *Foundations and Trends in Information Retrieval* 3(4), 2009.

AIOS

Correctness is a constraint, not a claim.

AIOS Litepaper · Version 1.4 · 2026-09-27