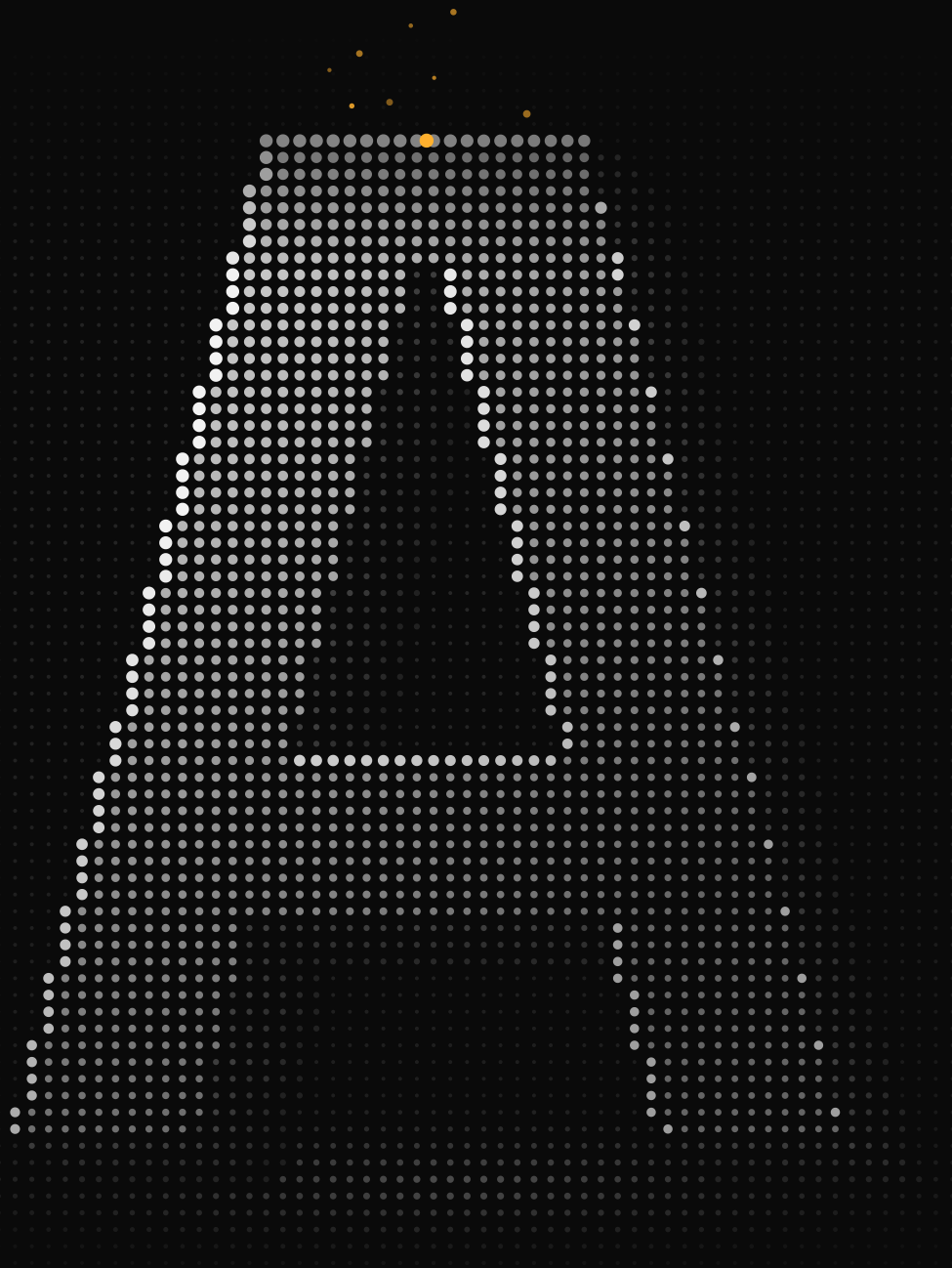


Whitepaper

AIOS: A Sovereign Chain for Verified AI Inference

Technical whitepaper: Proof of Inference, the Neural VM, and the Model and Data Internets

Contents

Figures

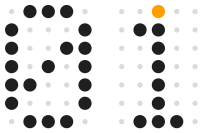
This technical whitepaper specifies the AIOS design together with its assumptions and limits; nothing described here is deployed at the time of writing, numbers marked `illustrative` are calibrated near launch against measured costs and security-budget modeling, and numbers marked `locked` are canon.

Weights are programs. Inference is consensus.

Front matter

Abstract

AIOS is a sovereign Layer 1, built on Cosmos SDK and CometBFT, in which inference is a first-class blockchain operation. Existing decentralized inference networks reward liveness and participation; they do not establish that an output came from the model the caller addressed. AIOS makes that property the object of consensus. Under Proof of Inference, a committee of Miners seated by an unbiasable, work-weighted VRF recomputes each job on the AIOS Module, a pinned integer-deterministic runtime, commits to its output hash before any reveal, and settles on the hash held by more than two thirds of committee work weight. Matching Miners are paid; a job without a supermajority is re-routed and, on exhaustion, refunded in full from caller-owned escrow. Miners post no stake: the mining layer's sole at-risk value is a one-time registration fee that is burned. Blocks are ordered by a work-gated BFT producer set whose small producer-bond, slashable only for equivocation, is the chain's one security stake. The same committee extends the proof to registered corpora and adapters, and small ternary models execute in chain state. Guarantees are stated per regime: small integer models reach byte-identical agreement, larger models carry statistical assurance, TEE execution is attested and never verified, and private memory is self-verifiable. "Verified" means provenance of execution, never that an output is true. The mechanism is unproven at scale, cross-hardware determinism of the AIOS Module is a pre-mainnet gate, and the security of stakeless mining rests on conditions whose parameters remain open. $AIOS has a fixed supply of 300,000,000 `locked`.



Chapter 1

Introduction

This chapter states the problem, the construction, the scope of the central claim, and the contributions of the paper.

1.1 The verification gap

Existing decentralized inference networks coordinate large operator populations and reward liveness and participation: evidence that a node is online and holds the hardware it claims. A node can pass every such check and still return a wrong output, a degraded output, or another model's output. The result is trusted, not proven.

We state the problem precisely. Given a result produced by mutually distrusting operators, establish, without trusting any one of them, that it is the output of specific registered weights on a specific input under a specific decoding rule. We call this property *provenance of execution*. It says nothing about whether the output is true, useful, or safe, and, to our knowledge, no deployed system secures real economic value on consensus over it at scale. Three obstacles stand in the way: floating-point inference is not reproducible bit for bit across heterogeneous hardware [1]; a verifier paid to agree can copy instead of recomputing [2], and identities are cheap [3]; and frontier-scale models cannot run inside a replicated state machine at reasonable cost.

1.2 The construction in brief

AIOS is a Layer 1 built for machine intelligence, in which inference is a first-class blockchain operation. Its mechanism, Proof of Inference, combines deterministic integer execution on a pinned runtime (the AIOS Module), independent recomputation by a committee of Miners seated by an unbiasable, work-weighted VRF, commit-then-reveal of output hashes, and settlement on the hash held by more than two thirds of committee work weight. The name describes the construction, not a category. Around it the chain fuses four pillars: small integer models in chain state, executed by the Neural VM [4]; inference sold as blockspace; the Model Internet, a registry of models and adapters; and the Data Internet, a registry of corpora whose retrieval the committee recomputes. Weights are programs, and inference is consensus.

1.3 Scope of the central claim

On AIOS, *verified* means one thing: a result cleared the Mode A committee consensus path, which establishes provenance of execution. It never means that an output is true, safe, unbiased, or free of prompt injection. Large models carry statistical assurance and are never described as certain; results from a trusted execution environment (TEE) are *attested*; recall over private memory is *self-verifiable*; and verified data means provenance and integrity, never truth. Chapter 9 formalizes these distinctions as assurance classes enforced in schema.

1.4 Contributions

This paper makes the following contributions:

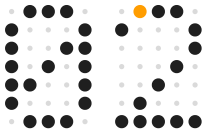
1. **Proof of Inference**: consensus on output hashes of deterministic integer inference recomputed by a work-weighted VRF committee, with reward-on-match settlement and no mining stake, and a conditional security analysis (Chapters 5 and 6).
2. **A determinism contract**: the AIOS Module, a pinned integer runtime in consensus spanning inference, embedding, retrieval, and adapter kernels, gated by differential validation (Chapter 4).
3. **Stakeless mining anchored by a burn**, its securing conditions, and a bonded fallback (Chapter 7).
4. **Work-gated ordering** over an unmodified CometBFT engine (Chapter 8).
5. **Provenance beyond the model**: verifiable retrieval, Verified Memory, and the Adapter Registry (Chapters 10 and 11).
6. **Four assurance classes**, `verified`, `sampling`, `attested`, and `self-verifiable`, carried in every record (Chapter 9).
7. **A conserved dual-home supply** of 300,000,000 $AIOS (Chapters 13 and 14).
8. **An explicit account of limitations** (Chapter 16).

1.5 Related approaches

Liveness-rewarding networks supply operator populations and peer-to-peer patterns this design borrows, but verify no outputs. Designs that pair deterministic integer inference with optimistic single-replica recomputation and restaked, slashable collateral under a single-GPU-architecture policy validate recomputation and correspond to our bonded fallback (Section 7.7); AIOS differs in committee consensus, stakeless reward-on-match, Native Models in chain state, and a hardware-agnostic runtime. Zero-knowledge proofs of inference lack verified block-time proving at 7B parameters and above, TEE attestation proves confidentiality rather than correctness, and judge-model scoring is rejected (Section 5.7). Onchain model registries exist; the Model Internet's distinguishing property is its binding to verification.

1.6 Organization

Chapters 2 to 4 set out the model and the runtime, 5 to 9 the mechanisms, 10 to 12 the registries and the record, 13 to 15 economics and governance, and 16 the limitations.



Chapter 2

System and Threat Model

This chapter fixes the participants, the assumptions, the adversary, and the security goals against which the rest of the paper argues.

2.1 Participants

- ***Callers*** (people, applications, or agents) submit jobs and escrow $AIOS for them.
- ***Miners*** register once by burning a fee, then perform jobs (as the *performer*), recompute them (as the *committee*), and pin, serve, and prove corpora; a TEE-capable subset may serve Mode B.
- ***Producers*** order blocks, admitted by work credit and a standing producer-bond.
- ***Asset owners*** register models, adapters, and corpora, set prices, post a creator-bond, and earn a creator share or royalty.
- ***The foundation multisig*** proposes upgrades and holds halt-only controls; its signers are operator-controlled at launch, a disclosed centralization.
- ***External trust domains*** appear only where named: Ethereum for the Phase-1 token, an audited bridge provider, and silicon vendors' attestation services for Mode B.

2.2 Assumptions

CometBFT is safe under asynchrony and live under partial synchrony [5, 6]; deadlines, dispute windows, and epochs are measured in block heights and inherit that liveness. Miners communicate over libp2p with GossipSub dissemination, Kademlia discovery [12], and relayed NAT traversal, a network with no consensus or verification authority. We assume a collision-resistant hash function H for content hashes, commitments, and Merkle trees [10]; unforgeable secp256k1 signatures; VRFs with uniqueness and pseudorandomness [8]; and an unbiasable beacon, either drand-style threshold randomness [9] or commit-reveal combined with a VRF. One engineering assumption is not yet validated.

> ***Assumption D (Determinism).*** For every admissible hardware platform, the pinned AIOS Module maps identical inputs to identical output bytes.

Every objective penalty stays disabled on any path where Assumption D is unvalidated (Section 4.5). Only Mode B assumes that no adversary can forge quotes under M of the N vendors in a quorum, and only the bridge assumes the honesty of its provider's validator set

or light clients.

2.3 Adversary

The adversary is economically rational where attacks are priced and Byzantine where safety is stated. It may (1) rent transient compute to accumulate or exercise work weight; (2) create many Miner identities, each at the cost of a burn; (3) collude out-of-band, including a performer that leaks its claim before the commit window; (4) try to grind committee and proposer randomness; (5) verify lazily by copying; (6) manufacture wash calls to farm credit, fees, or emission; (7) gain physical access to TEE hardware, which let TEE.Fail (October 2025) forge valid quotes for under 1,000 US dollars [11]; (8) register mislabeled models, weights cheap to register but expensive to run, or state-bloating blobs; (9) accept jobs and withhold results; and (10) equivocate or censor as a Producer, or accrue credit patiently across epochs. It cannot break the hash, forge signatures, or predict the beacon, and it holds less than one third of producer voting power, the ordering assumption, stated rather than derived.

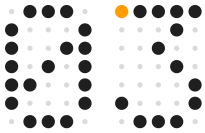
2.4 Security goals and non-goals

Table 1 lists the goals, the mechanism behind each, and the condition under which it holds.

Table 1. Security goals.

Goal	Statement	Mechanism	Condition
1. Output safety	An incorrect output is finalized only if adversarial work weight exceeds two thirds of the committee	Work-weighted supermajority over committed hashes	Assumption D; unbiasable beacon
2. Liveness	Every job settles or is refunded within bounded rounds	Deadlines, re-routing, bounded re-route count	Ledger liveness
3. Caller funds	A caller never pays more than its escrow and is fully refunded on no consensus	Caller-owned, non-custodial, worst-case escrow	By construction
4. Honest principal	Honest divergence or downtime never costs a Miner its registration	Determinism-gated penalties	By construction
5. Ledger safety	No two conflicting blocks are finalized	CometBFT over the work-gated producer set	Byzantine power below one third
6. Conservation	Ethereum circulating plus native circulating plus cumulative burns equals 300,000,000 at every block	Sole native mint authority, lock-and-mint bridge, monitor	Soundness of the audited bridge verifier
7. Labeling	Every result carries its assurance class	Disjoint label spaces in schema	By construction

Non-goals: truth or quality of outputs, relevance of retrieval, confidentiality of Mode A inputs from the committee, prompt-injection resistance, and, at launch, label accuracy (Chapter 16).



Chapter 3

Architecture

This chapter describes the pillars, the two layers, the chain modules, the off-chain stack, and the application surfaces.

3.1 Four pillars and two layers

The pillars are **(A) chain-state-as-model**, small integer models (BitNet-class ternary in the reference design [4]) held in chain state and executed in consensus, toy-scale by physics; **(B) blockspace-as-inference**, jobs recomputed by a committee and committed into blocks, bought like gas; **(C) the Model Internet**, giving every model and adapter an address, owner, price, content hash, and verification mode; and **(D) the Data Internet**, where corpora register as owned, priced assets whose retrieval the committee recomputes. The contribution is their fusion into one record, the Inference Receipt (Section 12.2).

Ordering and mining are separate layers. Ordering runs CometBFT over a work-gated producer set and rests on an honest majority of work-weighted producer power, anchored by an equivocation-slashable bond. Mining is stakeless and rests on the weaker assumption of an honest majority of committed work per committee, anchored by a burned registration fee. The ledger's safety and liveness never depend on the inference layer, but a captured committee, lazy verifiers, or a biased beacon can misdirect value while the ledger stays consistent; Chapters 5 to 7 bound those failures.

3.2 Chain modules

The chain's logic lives in native Cosmos SDK modules, each a separate audit surface (Table 2). The chain uses Bech32 `aios1...` addresses and secp256k1 keys and has no EVM module: all onchain logic is native modules, never Solidity.

Table 2. Native modules.

Module	Responsibility	Status
`x/inference`	Job lifecycle, including the grounded pipeline (embed, retrieve, generate, settle); per-job settlement	Core
`x/neuralvm`	In-consensus execution of Native Models and Native Corpora; honeypot ground truth	Core
`x/registry`	The Model Internet (models, adapters); Miner registration, the burn, the Miner set	Burn and adapters review-gated
`x/data`	The Data Internet: corpora, chunk and embedding commitments, availability bond, retrieval and royalty hooks; Verified Memory by a visibility flag	Review-gated
`x/producer`	Work-gated producer set, producer-bond, equivocation slashing	Review-gated consensus surface
`x/attest`	Mode B attestation verification and quorum settlement	Review-gated before implementation

The AIOS Module is not a chain module but a pinned Rust crate used, in one version, by `x/neuralvm` and the off-chain Miner client. No module custodies, repurposes, or cross-authorizes another's bond or the burn.

3.3 The off-chain stack

Miner software follows the Sidecar Mesh: a Go control plane handles chain, network, and Producer duties (job intake, commit-reveal, settlement glue, and, in full-node mode, the Producer role) over go-libp2p, and a pure-Rust sidecar links the pinned Module with no cgo, keeping determinism-critical kernels out of the Go runtime. Their loopback-only gRPC or Unix-socket contract is consensus-relevant: its encoding, sampler configuration, and output-hash contract join the Module content hash, so a skew is a divergence, and it is review-gated before scaffolding. A read-side indexer and asynchronous delivery are deferred to Phase 2. Heavy compute is off-chain and verified or, in Mode B, attested; the chain holds consensus, accounting, registries, escrow, Native Models and Corpora, tallies, and ZK dispute verification.

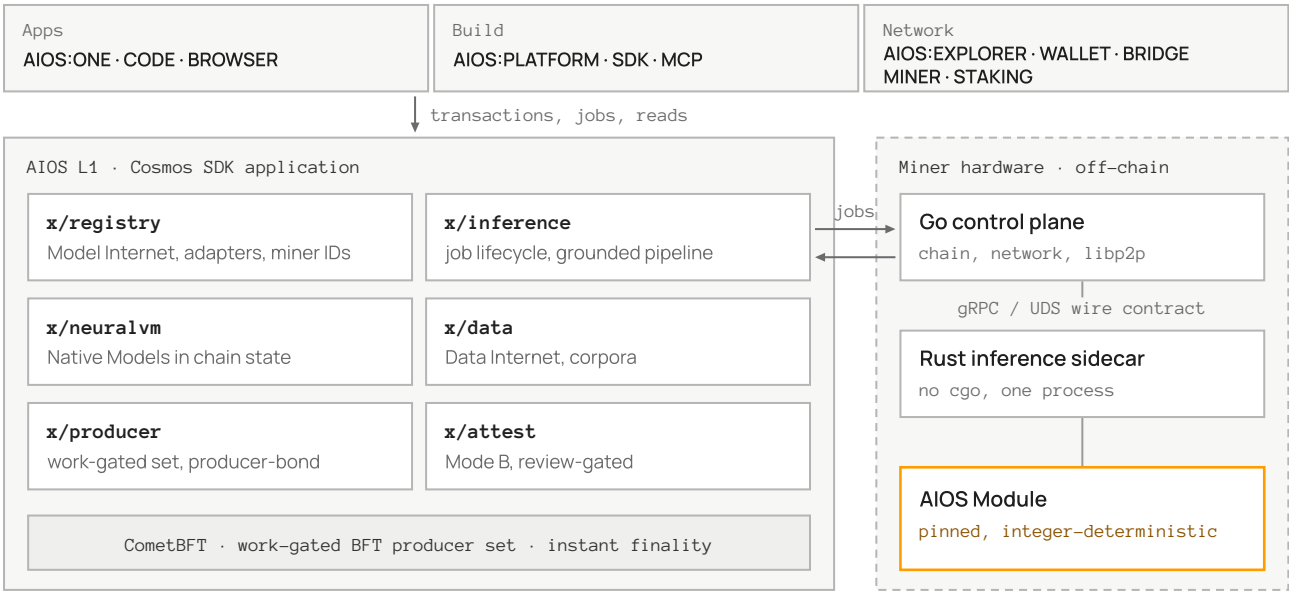


Figure 1. System architecture. The Cosmos SDK application holds six native modules above CometBFT. Heavy inference runs off-chain on Miner hardware through the Sidecar Mesh, whose pinned AIOS Module makes results byte-comparable across machines.

Figure 1 shows the stack, from applications through the six native modules to the Sidecar Mesh. The AIOS Module appears twice, in the chain and on every Miner, in one pinned version, which is what lets the two agree.

3.4 Application surfaces

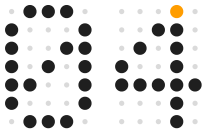
The pillars are packaged as 11 products in three families (Table 3). Brain (verifiable memory) and Agent (actions) are toggles within the apps, not products, and long-tail applications are left to third parties.

Table 3. The product roster.

Family	Product	Role
Apps	AIOS:ONE	Assistant with Brain and Agent toggles and user-defined session scopes (projects)
Apps	AIOS:CODE	Open coding agent, CLI-first, over SSH; provisional steps reconcile when proof lands, so verification never blocks
Apps	AIOS:BROWSER	A full agentic browser from day one, not an extension; a Phase-2 flagship
Build	AIOS:PLATFORM	Console and REST API; wallet sign-in, non-custodial prepaid balance, scoped, capped, revocable keys; creator surface for corpora and adapters
Build	AIOS:SDK	SDK and integration documentation
Build	AIOS:MCP	Agent doorway returning a verification record with every call
Network	AIOS:EXPLORER	Jobs, proofs, Miner standing, models, corpora, the provenance graph

Family	Product	Role
Network	AIOS:BRIDGE	Single-asset bridge (Chapter 14)
Network	AIOS:WALLET	Non-custodial wallet; hold $AIOS, run a Miner
Network	AIOS:MINER	Unified node software (Section 7.6)
Network	AIOS:STAKING	Non-security lock-to-earn pool (Section 13.6)

Every surface renders one record schema (Section 12.2). AIOS:PLATFORM keys are a bounded-spend credential, never custody. Verified inference does not protect agents or browsers from prompt injection, a separate unsolved problem, and no surface claims otherwise.



Chapter 4

Determinism and the AIOS Module

This chapter explains why byte-identical execution is the precondition of the design, specifies the runtime that provides it, and states its validation gate.

4.1 Determinism as the precondition

Definition 1 (Byte-determinism). A runtime R is byte-deterministic over a set of platforms $\mathcal{P}$ if, for every admissible input x and all $p, q \in \mathcal{P}$, the outputs $R_p(x)$ and $R_q(x)$ are identical as byte strings.

Without byte-determinism, honest Miners produce different hashes, no supermajority forms, and the system is no more secure than a single executor: determinism is the mechanism's precondition.

4.2 Why floating point cannot supply it

Floating-point addition is not associative [1], so a reduction's value depends on its order, which varies with the kernel, the tensor-core topology, the batch, and the library version. Batch-invariant kernels restore run-to-run determinism on fixed hardware but not across hardware or versions [13, 14], and the design's own measurement found a 0.0% bitwise match between FP16 outputs of one model on A100 and H100 GPUs. Off-the-shelf FP16 weights therefore cannot be recomputed to agreement across heterogeneous hardware and are not exactly verifiable. Integer arithmetic removes the dependence on order.

Proposition 1 (Order-independence of integer accumulation). Let a reduction sum integers in an accumulator of fixed width w, with either wrapping (modulo 2^w) semantics or a guarantee that no partial sum overflows. Then the result is independent of the order of summation.

Argument. Addition over $\mathbb{Z}$ and over $\mathbb{Z}/2^w\mathbb{Z}$ is associative and commutative, so every order yields the same element. Saturating arithmetic is not associative, which is why accumulator width and overflow policy are pinned per kernel as consensus state; reduction order is pinned too, so byte identity does not rest on this argument alone. □

4.3 The AIOS Module

The AIOS Module is the pinned, integer-deterministic runtime that every Miner and the Neural VM execute.

- **Arithmetic.** Pure-integer fixed point end to end, including matrix multiplication, softmax, and layer normalization, in the integer-only inference lineage [15, 16, 17, 18].
- **Kernels.** Batch-invariant, with pinned reduction order and accumulator widths.
- **Packaging.** One Rust crate pinned by content hash and changed only through the upgrade path of Section 15.1. The same version runs in `x/neuralvm` and in the Miner client; skew between them is itself a divergence, so the version is part of consensus.
- **Boundary.** A deterministic FFI contract: pure integers in, a byte-identical hash out, with no floating-point operation, uninitialized memory, or platform-dependent intrinsic crossing it and a fixed operation order on both sides. One escape silently breaks byte identity, the design's most catastrophic correctness failure, so every change is review-gated.

We write μ for the Module content hash, covering the crate, the FFI contract, the sidecar wire contract (Section 3.3), and every pinned kernel; Miners with different μ are not expected to agree. The lineage is peer-reviewed (IJCAI 2020 [15], ICCV 2023 [18]). A single-developer sovereign testnet has reported integer-only recomputation, but its performance numbers failed independent verification and are treated as reported, not confirmed.

4.4 Decoding and kernel scope

Temperature and top-p sampling are stochastic, so decoding on the Module is greedy, or uses a sampler configuration and seed committed in the job and applied identically by every member; anyone holding the weights, input, and decoding specification can reproduce the hash. Greedy and fixed-seed decoding give less varied, often lower-quality output, so verifiability constrains the generative product, a cost carried openly. Behind the same boundary the Module also pins the integer embedding, exact-retrieval (exhaustive inner-product kNN, or BM25 scored in fixed point), and hash tie-breaking kernels for retrieval (Chapter 11), and the integer low-rank composition for adapters (Section 10.3). Each joins μ and faces the same gate.

4.5 The pre-mainnet determinism gate

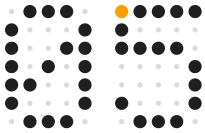
Integer determinism is established on CPUs; byte identity across heterogeneous GPUs, with differing libraries, reduction orders, and tensor cores, must still be demonstrated on the real kernels. The validation harness is differential, comparing a CPU-reference order with every GPU backend, with accumulator width, overflow policy, and reduction order pinned per kernel. A no-float CI check is necessary but not sufficient, because a reordered integer reduction or a width mismatch passes it and still breaks identity. The harness covers the inference, embedding, retrieval, and adapter kernels.

No objective penalty turns on, on any path, until the harness passes, or until the Module pins a single CPU-reference order that every Miner follows. Until then every path is dispute-then-penalty, and an honest Miner on divergent hardware goes unpaid for the job but is never penalized. This is the most important pre-mainnet gate. The first milestone applies

the same discipline to a standalone prototype (Section 15.4) under four gates: G1, a no-float CI gate; G2, a cross-host byte-identical log; G3, a negative test in which a perturbed instance is caught, not silently resolved; and G4, an integer-end-to-end audit. Its graduation artifact is a cross-machine byte-identical recomputation log, not a screenshot. Per-SKU bit-exact emulation of floating-point reductions [19] is tracked as a possible future dispute mechanism.

4.6 Quantization honesty

An integer model on the Module does not reproduce its FP16 source. AIOS proves provenance and correct execution of the registered quantized weights, never equivalence to a floating-point original: the caller receives a verified quantized model, not a verified copy of a named frontier model.



Chapter 5

Proof of Inference

This chapter defines the objects of Proof of Inference and specifies selection, commitment, tally, settlement, re-routing, the verification regimes, and disputes.

5.1 Objects

Definition 2 (Job). A job is a tuple $j = (m, x, \sigma, \pi_{\max}, \nu)$, optionally extended with a corpus c and an adapter a, where m is a registered model address, x the input, σ the decoding specification, $\pi_{\max}$ the maximum price, and ν the verification mode. At submission the caller escrows a protocol-fixed worst-case amount E_j.

Definition 3 (Work credit). The work credit $w_i \geq 0$ of Miner i is a non-transferable score accrued from verified honest work and decayed on repeated mismatch or withholding. It gates eligibility and weights selection, never multiplies a payout, and receives nothing from a registration burn.

Definition 4 (Committee). For job j, the committee C_j is a set of eligible Miners seated by a VRF keyed on an unbiasable beacon, weighted by work credit. The performer u_j is chosen separately, and $u_j \notin C_j$. For a set S of Miners, $W(S) = \sum_{i \in S} w_i$.

Definition 5 (Commitment and reveal). Each member $i \in C_j$ computes an output hash h_i and posts a commitment com_i before any reveal is visible; reveals open only after all commitments are in. The commitment must be hiding and bound to member and job (for instance, a salted hash over h_i, the job identifier, and the member's identity), since a bare hash of h_i could be replayed by a member who has not computed.

Definition 6 (Canonical output). With $W_h = \sum_{i \in C_j : h_i = h} w_i$ the work weight revealing hash h, the canonical output of j is the hash $h^\star$ satisfying

$$W_{h^\star} > \tfrac{2}{3} W(C_j).$$

If no hash satisfies this, the round has no canonical output.

5.2 Committee selection

Committees are seated from a drand-style threshold beacon or commit-reveal with a VRF, since proposer-supplied entropy is grindable. Eligibility requires accumulated verified work, and seating probability rises with work credit (for analysis, proportionally), so influence tracks demonstrated honest compute, never capital; the burn confers no selection weight, vote weight, or producer eligibility. The performer never sits on its own committee and is paid if and only if its claim equals $h^\star$. For high-value jobs, a per-entity influence cap bounds any entity's weight in a committee across all its identities, and a minimum-honest-compute floor must be met before seating; both bound concentrated influence without eliminating it. Anonymized committee assignment is a candidate defense under review.

5.3 Commit-then-reveal

Proposition 2 (No copying of reveals). If commitments are hiding and bound to member and job, each member's revealed hash is fixed before any other reveal is visible, and so is independent of the other reveals.

Argument. Binding fixes h_i at commitment, hiding ensures that no earlier commitment conveys another member's hash, and binding to identity prevents re-posting another member's commitment. A member that has not computed can commit only to a guess. □

The proposition does not stop a member from copying the performer's claim if the performer leaks it before the commit window (Section 6.3).

5.4 Tally and settlement

Proposition 3 (Uniqueness and threshold safety). (a) At most one hash can be canonical in a round. (b) Under Assumption D, if the adversary controls a fraction $\alpha \leq 2/3$ of the committee's work weight, no incorrect hash can be canonical. © Under Assumption D, if $\alpha < 1/3$ and every honest member reveals in time, the correct hash is canonical.

Argument. (a) Two canonical hashes would need disjoint member sets with combined weight above $\frac{4}{3}W(C_j)$. (b) Every honest member reveals the correct hash, so an incorrect one carries at most $\alpha W(C_j) \leq \frac{2}{3}W(C_j)$. © Honest weight $(1-\alpha)W(C_j)$ exceeds $\frac{2}{3}W(C_j)$ and sits on the correct hash. □

An adversary holding between one third and two thirds cannot finalize a wrong output but can block one, a liveness fault answered by re-routing. Honest members whose hardware breaks Assumption D reduce honest weight exactly as withholding does, so determinism failures surface as re-routes, not wrong outputs.

The performer and each member on $h^\star$ receive a per-seat share of the Miner fee plus a per-job bootstrap emission (Section 13.5); more credit means more seats and tally weight, never a larger per-seat reward. Members off $h^\star$ are unpaid and bear their compute; there is no token slashing on the mining layer, and proven fraud, including a failed honeypot, triggers

ejection (Section 7.3). Settlement is atomic per Miner (each on-match Miner paid exactly once, others nothing, no reverting the rest, state committed before transfers, ejection recorded separately), and the producer set commits the result and tally into a block.

5.5 Re-routing and refund

With no canonical hash, the job is re-routed to a fresh or larger committee while escrow is held, up to a maximum count $R_{\max}$ [open]; on exhaustion the caller is refunded in full with a no-consensus result, and nobody forfeits anything.

> ***Proposition 4 (Escrow termination).*** Every job terminates, settled or refunded, within $R_{\max} + 1$ rounds, and the caller never pays more than E_j.

> *Argument.* Each round ends at a deadline in block height, so rounds are finite while the ledger is live, and there are at most $R_{\max} + 1$ of them. E_j covers the maximum committee size and maximum re-route escalation; re-routes draw on it, never on new caller funds, and the remainder returns at settlement. □

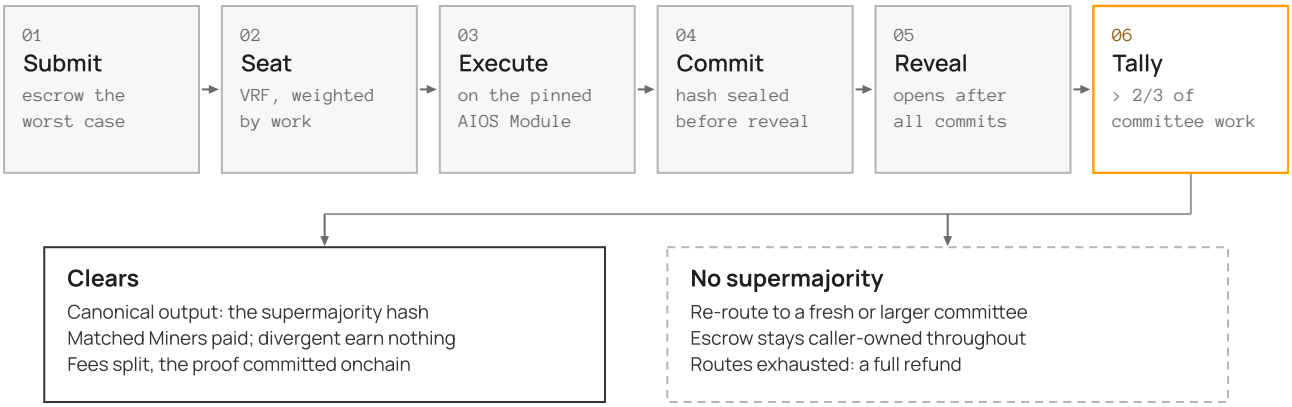


Figure 2. The Proof of Inference job lifecycle. Escrow is caller-owned throughout. A job either clears on a work-weighted supermajority, or it is re-routed, and a job that exhausts its routes is refunded in full.

Figure 2 traces a job from escrow to its two terminal states. There is no third state: nothing waits in escrow forever, and no failed tally charges the caller.

5.6 Verification regimes and disputes

AIOS runs two regimes with different guarantees, and every surface states which applies (Table 4).

Table 4. Verification regimes.

Regime	Used for	Guarantee	Consequence of a wrong result
Full recomputation	Native Models, small integer models	Byte-identical agreement	Objective no-pay; ejection only on proven fraud or a failed honeypot
Selective recomputation	Larger models that run the Module	Statistical, set by the sampling rate	Dispute; never an objective forfeit
Seed-synchronized audit (DiFR-style)	Models that cannot run the Module	Statistical, AUC-class detection	Dispute; never an objective forfeit

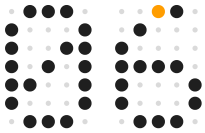
In selective recomputation the performer commits to intermediate states under a Merkle root [10] and the committee recomputes VRF-chosen openings; if a fraction ρ of states is corrupted and s uniform openings are checked, corruption escapes with probability at most $(1-\rho)^s$. Models that cannot run the Module, such as FP16 models whose cross-GPU byte identity is unsolved, are audited under a shared committed seed against a trusted reference run, and a significant divergence opens a dispute; this path is never described as verified by recomputation, and its detection target and reference funding are open. Large models are the actual product, so the product path is statistical, and says so.

Guardrail (determinism-gated penalty). Objective no-pay with ejection for proven fraud on a full-recomputation mismatch is enabled only on validated byte-deterministic paths; until the gate of Section 4.5 passes, GPU paths use dispute-then-penalty, and ejection never follows a divergence the Miner did not cause.

A challenger triggers a dispute; escrow is held and the result marked provisional. It escalates through an optimistic challenge with a loser-pays dispute bond, within a hard-capped window open, to a zero-knowledge proof of inconsistency as the last resort. With no verified block-time proving at 7B parameters and above, an unprovable dispute falls back to expanded recomputation by a larger committee, and where neither is feasible, to no penalty and a refund, so some large-model fraud is unpunishable beyond forgone reward. AIOS:EXPLORER shows each dispute, outcome, accused performer, and bond, but hides the challenger at launch pending a retaliation-risk review.

5.7 Rejected and deferred mechanisms

- **Proof of Quality (rejected).** Judge-model scoring falls to the Model Downgrade Attack: an adversarial 8B model is indistinguishable from an honest 70B model under a judge while human-evaluated quality drops by roughly 40%, verified 3-0. A registered judge model may be a product callers invoke, never a consensus rule.
- **zkML aggregation (deferred).** There is no verified evidence of block-time proving at 7B parameters or more; it is revisited as proving costs fall.
- **TEE attestation as a consensus root (rejected).** Vendor trust cannot root a sovereign chain, and published performance claims for it did not survive verification; TEE is admitted as Mode B (Chapter 9).



Chapter 6

Security Analysis of Proof of Inference

This chapter analyses the principal attacks on the mining layer and what bounds each. Committee size, sampling rates, caps, and floors are open, so the analyses give the shape of the argument, not a security level for a deployed system.

6.1 Committee capture

Let an adversary control a share p of eligible work weight. Assuming equal weights and a large eligible population, each of k seats is adversarial independently with probability p, and the adversarial seat count X is binomial. The adversary captures the committee, finalizing an output of its choice, when $X > 2k/3$:

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

It blocks the committee, forcing a re-route but not a wrong output, when $X \geq k/3$.

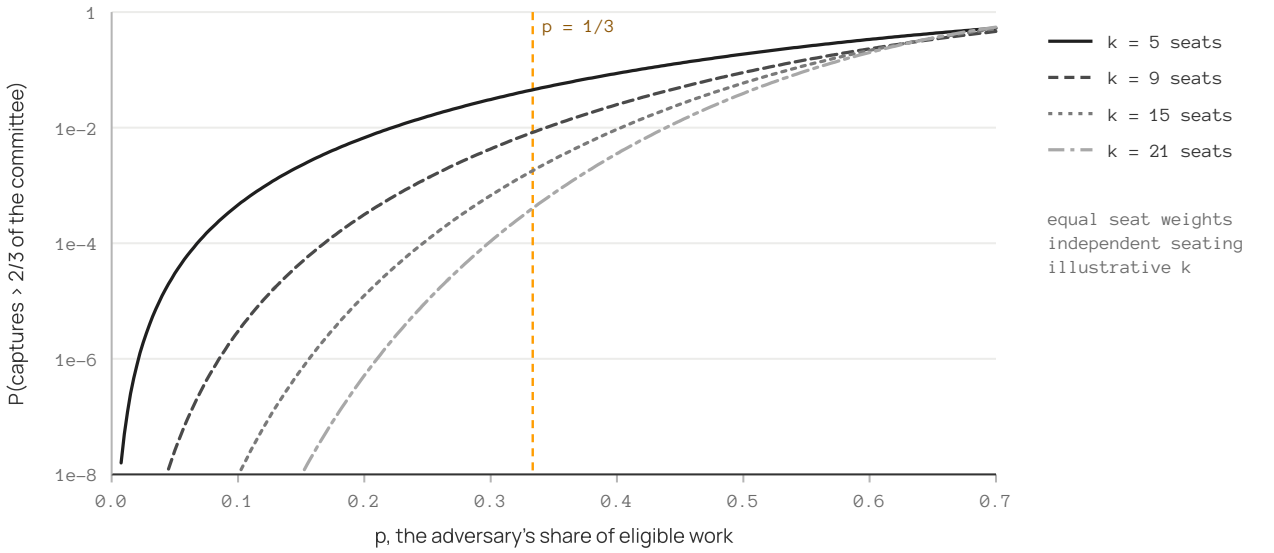


Figure 3. Probability that an adversary holding a share p of eligible work captures a greater-than-two-thirds committee supermajority, under a binomial model with equal seat weights and independent seating. Committee size is an open parameter; the sizes shown are illustrative. Capture falls steeply with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

Figure 3 plots P_{cap} for illustrative sizes $k \in \{5, 9, 15, 21\}$. Size helps greatly while p is well below two thirds: at $p = 0.2$, capture falls from about 6.7×10^{-3} at $k = 5$ to about 5×10^{-7} at $k = 21$. It barely helps near the threshold: at $p = 0.5$ the range is only about 0.19 to 0.04, and at $p = 0.6$ capture stays near 0.2 or above. Security therefore depends on keeping any adversary's effective share well below two thirds, which is the job of the per-entity cap and the honest-compute floor, while the value cap bounds what capture can take: over n jobs worth at most V_{cap}, expected capture loss is at most $n \cdot P_{\text{cap}} \cdot V_{\text{cap}}$. Blocking is far cheaper: above one half at $p = 1/3$ for every size shown.

The model ignores unequal weights (hence per-entity caps over all of an entity's identities) and correlated seats, and p is not fixed: renting compute and accruing credit raise it at a lower, more elastic cost than stake, most easily at genesis, when honest compute is small (the bootstrap-compute cliff).

6.2 Lazy verification and honeypots

Reward-on-match sharpens the verifier's dilemma [2]: the cheapest route to the reward is to agree with the likely majority. Commit-then-reveal stops copying of other members (Proposition 2), and honeypots police the rest. A honeypot is a VRF-selected job whose performer claim is known to be wrong; a member that agrees with it fails. Oracle-free ground truth exists only where the chain computes the answer itself: Native Models in the Neural VM, or corpora with a known top-k. Honeypots must be indistinguishable from real jobs and isolated from determinism divergence; their design is under review. If a fraction h of jobs are honeypots and a lazy member copies on m jobs, it survives with probability

$$S(m) = (1 - h)^m,$$

and the expected number of jobs before detection is $1/h$.

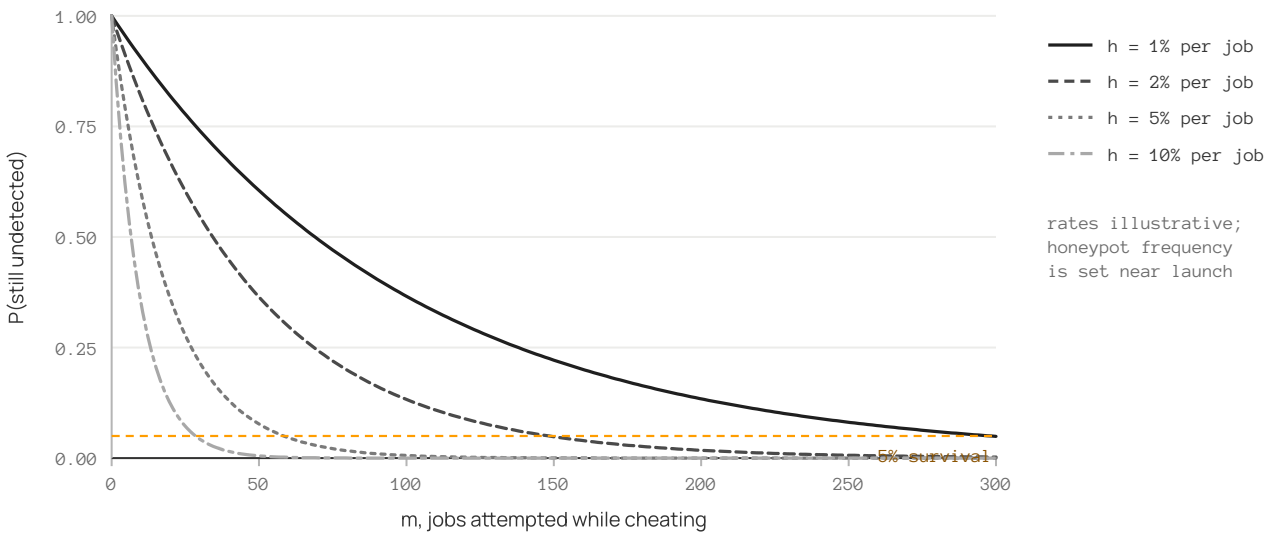


Figure 4. Probability $(1-h)^m$ that a persistent cheater or lazy verifier survives m jobs undetected, for a per-job audit or honeypot rate h. The rates are illustrative; the honeypot frequency is calibrated near launch.

Figure 4 plots $S(m)$ at illustrative rates $h \in \{1\%, 2\%, 5\%, 10\%\}$. At $h = 5\%$ a persistent lazy verifier survives 50 jobs with probability about 0.08; at $h = 1\%$ it survives 100 jobs with probability about 0.37. Persistent laziness is caught quickly, occasional laziness is not, hence the raised rate and the severe consequence: immediate ejection, forfeiture of accrued-but-unpaid reward, and loss of the identity, which only a new burn replaces. Honeypot ground truth and rewards are funded from a named protocol source (the insurance and honeypot share of the protocol slice, or an insurance pool), never from peers or the committee. On the large-model path the defense is commit-before-reveal, sampling, disputes, and the burn.

6.3 Collusion and grinding

A performer can leak its claim to members before the commit window, letting them commit without computing; commit-then-reveal does not stop this. The residual is bounded by the per-entity caps, the honest-compute floor, the value caps, and, on the in-VM path, honeypots: reduced, not closed, with anonymized assignment as the candidate defense. The committee and proposer beacons use the unbiasable construction of Section 5.2; committee-selection grinding is an open security-review item for Phase 2, and validating the proposer beacon is an ordering-layer mainnet blocker (Section 8.3).

6.4 Sybil entry and wash calls

Each identity costs a burn of at least F, so n identities cost at least nF. The burn prices fleet entry but not per-committee participation: one identity sits on many committees for one burn, and a rented-compute capture needs no extra identities, so committee Sybil resistance rests on work-gated eligibility and the caps of Section 6.1. Wash calls take two

forms: a creator manufacturing calls to harvest fees, and a Miner manufacturing jobs to farm work credit, and with it seating and vote weight. Non-transferable credit is necessary but not sufficient; the load-bearing requirement is cost anchoring, meaning credit must cost more in gas, fees, and amortized burn than the advantage it buys. Because credit also gates the producer set, farming threatens ledger safety and is a mainnet blocker. The benchmarks that must show these attacks net-negative (wash inference against emission, targeted eviction against freed seats, register-farm-evict churn) are unrun and must use the lower post-bond cost basis.

6.5 Deterrence: the burn floor and value caps

The floor F is a security floor, not a spam deterrent: it must exceed the dishonest payday of the largest job a Miner can be seated on before detection, or jobs above that level are value-capped below the floor's equivalent. The sketch below shows why F, the cap, and the honeypot rate are calibrated jointly.

> ***Proposition 5 (A sufficient deterrence condition).*** Suppose each dishonest job is detected independently with probability h, a detected Miner loses a replacement cost of at least F plus its accrued-but-unpaid reward, and each undetected dishonest job gains at most $g(V_{\text{cap}})$. Then persistent cheating has negative expected value whenever

$$F > g(V_{\text{cap}})\,\frac{1-h}{h}.$$

> *Argument.* Undetected dishonest jobs before the first detection are geometric with mean $(1-h)/h$, so expected gain before detection is at most $g(V_{\text{cap}})(1-h)/h$, while detection costs at least F. □

This is a sketch, not a calibration. It covers cheating that honeypots or disputes can detect, not a captured committee, which is its own detector and is bounded by Section 6.1 and the value cap. It also shows why burn-only mining is weaker than a per-job bond against one high-value fraud: the burn is paid once, while gain scales with job value. Jobs above the cap need a per-job bond or the bonded tier (Section 7.7).

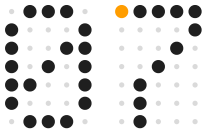
6.6 Summary

Table 5. Threats to the mining layer.

Threat	Principal controls	Status
Capture with rented compute	Committee size, work weighting, per-entity cap, honest-compute floor, value caps, unbiasable VRF	Bounded, not eliminated
Lazy verification (in-VM)	Commit-then-reveal, honeypots	Bounded by the honeypot rate
Lazy verification via performer collusion	Caps, floor, value caps; anonymized assignment (candidate)	Reduced, not closed

Threat	Principal controls	Status
Beacon grinding	Unbiasable beacon	Validation is a mainnet blocker
Sybil fleet entry	Registration burn	Priced
Wash calls, credit farming	Cost-anchored credit, caps, floor	Open mainnet blocker
Weight substitution	Members check weights against the registry hash	Closed by construction
Liveness griefing	Deadlines, re-routing, weight decay, ejection for sustained griefing	Bounded
Adversarial weights (gas, state bloat)	Registration costs, review	Open review item

In Table 5 only weight substitution is closed outright; each open entry is a mainnet blocker or review item, not an accepted risk.



Chapter 7

Mining and the Registration Burn

This chapter specifies how a Miner enters, what it risks, how it is paid and penalized, and how the Miner set is maintained.

7.1 Stakeless mining

Mining on AIOS is the performance and recomputation of verified inference. Miners lock no standing stake, post no per-job bond, and face no token slashing; the per-job work-bond and its module, `x/workbond`, were removed on 2026-06-21. Something must still be at risk: outputs are not self-proving, and free entry with rentable compute makes capture cheap at electricity cost. The security review found zero-cost mining not securable and burn-only mining securable with conditions (Section 7.3).

> ***Definition 7 (Registration burn).*** A registration burn is a one-time payment in native tokens, made through `x/registry` and destroyed on payment. It mints a persistent, enumerable Miner identity bound to the registrant's wallet and makes that wallet VRF-eligible. It confers zero work credit, selection weight, vote weight, and producer eligibility, and, being destroyed, cannot be refunded, redistributed, captured, or repurposed as any bond.

A burn is lost, in the only sense destroyed value can be, when its identity is ejected and re-entry needs a fresh burn. AIOS has one mainnet Miner set, with no subnets. Funding runs from the Phase-1 ERC-20 through the lock-and-mint bridge to a native burn; the ERC-20 never burns, and the path adds no mint or custody authority. Every Miner needs a wallet. Phase-0 registration is free and allowlisted, with no points program.

7.2 The self-regulating burn schedule

Let $e \in [0, 1]$ be the fraction of the 100,000,000-token mining-incentive pool already emitted, read from onchain release accounting. The maturity baseline is

$$B(e) = F + (R_0 - F)\,(1 - e)^{\gamma},$$

with genesis value R_0, floor F, and curvature $\gamma > 0$, and one registration costs

$$c = \min\big(\max(B(e) \cdot M,\ F),\ \mathrm{Cap}\big),$$

where Cap is a ceiling and the demand multiplier M is updated each controller epoch from the observed registration rate r against a target $r^\star$:

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

The genesis value is 10,000 $AIOS locked. Only it and the schedule's shape are operator-set; the rest are open, calibrated with the wash-call cost anchor and per-entity caps. The all-in cost includes the 1% transfer tax on acquisition, unless bought through the exempt pair or router.

Since $B(0) = R_0$, $B(1) = F$, and B is non-increasing when $R_0 > F$, registration is dearest early and glides to the floor. Pool release is fee-conditioned and shared with the ordering sink, so e tracks realized depletion, not calendar time or node count. The multiplier prices out bursts like a difficulty adjustment. The computation is consensus state, using pinned fixed-point arithmetic, a deterministic controller epoch and window, and a rate signal that registration-rate griefing cannot manipulate.

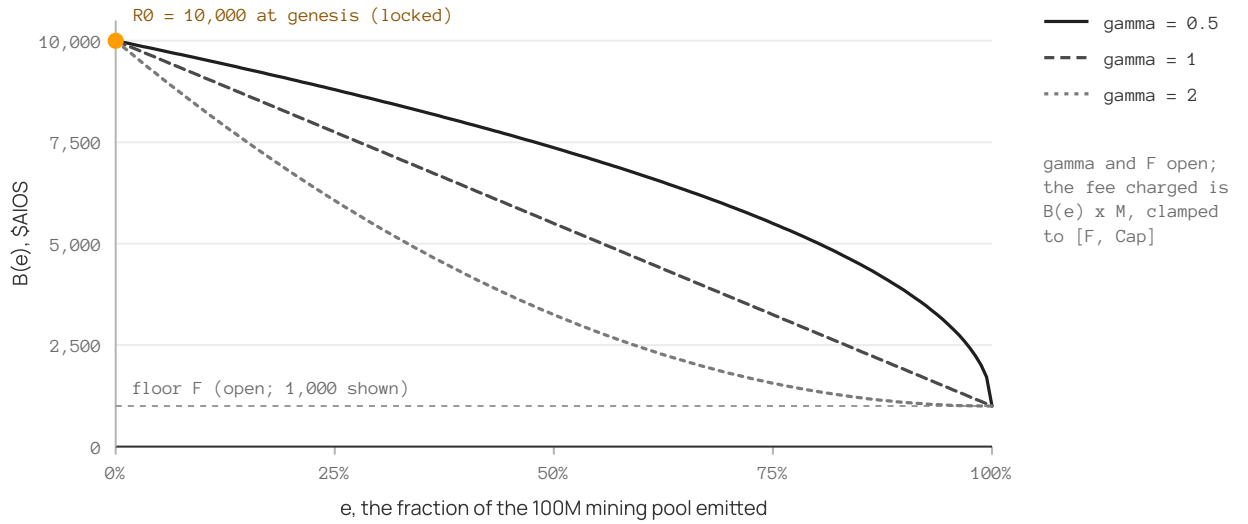


Figure 5. The registration-burn baseline $B(e) = F + (R0 - F)(1 - e)^{\gamma}$, where e is the fraction of the mining pool already emitted. R0 = 10,000 $AIOS is locked; F and γ are open, and the floor is drawn at an illustrative 1,000.

Figure 5 shows $B(e)$ for $\gamma \in \{0.5, 1, 2\}$ with an illustrative F. Larger γ front-loads the decline, smaller γ holds the price near R_0 longer, and in every case F bounds the cost from below: F, not R_0, is the security parameter.

7.3 Conditions on the burn-only anchor

The security review's nine conditions are load-bearing. (1) Ejection for proven fraud or a failed honeypot is immediate and objective only on validated byte-deterministic full-recomputation paths, dispute-then-penalty elsewhere, and never follows divergence or honest downtime. (2) The floor is a security floor (Section 6.5). (3) Value caps, enforced at submis-

sion relative to the floor and expected jobs to detection, are load-bearing; larger jobs need a per-job bond or the bonded tier, and removing the cap is a Hard Stop. (4) Honeypot frequency is raised and funded from a named protocol or insurance source, never peers. (5) Per-entity caps and the honest-compute floor span all of an entity's identities. (6) The registration allowlist lasts longer, tied to a higher honest-compute floor. (7) The cost-anchor and eviction benchmarks are re-run at the lower post-bond cost basis before any allowlist decay, a mainnet blocker. (8) Lost value is destroyed, never paid to the committee, which would reward manufactured mismatches (a Hard Stop). (9) A determinism bug costs honest Miners forgone reward, never principal.

7.4 Consequences and eviction

Table 6 lists every consequence a Miner can face; none is a token slash.

Table 6. Consequence tiers.

Tier	Trigger	Consequence	Resolution
No-pay	Off the canonical hash	Forgone reward, spent compute	Objective
Ejection	Proven fraud; failed honeypot	Identity ejected, accrued-but-unpaid reward forfeited, fresh burn to re-enter	Objective on validated byte-deterministic paths, else dispute-then-penalty
Dispute	Failed opening; statistical divergence	Penalty only after resolution	Challenge, ZK proof, expanded recomputation, or refund
Mode B fault	Failed, missing, stale attestation	Re-route with liveness penalty, or dispute	Never objective; revoked firmware ejectable via the subjective tier
Decay	Repeated mismatch; accept-then-withhold	Lower selection weight; ejection only for sustained griefing	Reputational

Honest downtime, indistinguishable from malice, never ejects for cause; the boundary is open. A guardian can halt but never originate an ejection, and an executed wrongful ejection is reimbursed from the insurance pool only if a guardian-authorized review confirms it, else it is final and uncompensated.

The active set is soft-capped open. Registering into a full set deregisters the Miner with the lowest sustained verified work after an immunity period open, computed deterministically from the decayed verified-work score alone, with an unbiasable tie-break, no discretion, and never on divergence no-pays or downtime false positives. The evicted Miner loses its sunk burn without restitution, a deliberate exception defensible because the burn is an entry cost, never a deposit or guaranteed yield. The cap and immunity period change only by producer adoption (Section 15.1).

7.5 Per-job, solo mining and work credit

Every settlement is scoped to one job, and income is the sum of matched jobs: no pooled reward, mining pool, or delegation. Solo-only removes a Sybil amplifier but does not make the network collusion-resistant; off-protocol collusion is bounded by committee size, the

VRF, the burn floor, value and per-entity caps, and the honest-compute floor, not eliminated. Omitting pools assumes steady demand, since AIOS pays every matching member rather than one winner per rare block; under sparse demand, off-protocol pooling may re-emerge past the per-entity cap, an open item.

Work credit is an eligibility gate and a selection weight, and nothing else: it never multiplies a payout, cannot be sold, lent, or delegated, and binds to the Miner that earned it. It also feeds the producer set, under the firewall of Section 8.4.

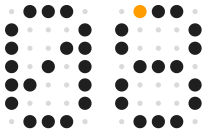
7.6 Hardware, liveness, and AIOS:MINER

Miners bring their own hardware; AIOS funds none of it. Commodity GPUs or CPUs suffice for Mode A. TEE-capable hardware (H100, H200, and B200-class GPUs on confidential-VM hosts) is optional, serves only Mode B, and is never required or on the liveness-critical path. A liveness-and-resource proof, separate from correctness, shows a Miner is online and holds its claimed hardware through a periodic Merkle commitment over a resource probe, a deadline-bound challenge, and strike penalties; it carries no authority over results.

AIOS:MINER is the unified node software: a zero-install browser client over WebGPU, the mass-onboarding path from Phase 1, which lowers the installation barrier but never the determinism bar; a desktop application; and a command-line client whose full-node mode can also run a Producer. It carries the data-serving role (Section 11.4). TEE-capable Miners pay the same burn and gain no extra weight, and AIOS:MINER is not ASIC-mineable.

7.7 The bonded fallback

If burn-only mining proves insecure at mainnet grade, the reserved fallback reintroduces a per-Miner bond on the inference layer, slashed for proven fraud, beside the verified-inference reward. It concerns mining only, adds no delegated token stake, needs no redesign elsewhere, and remains a novel, buildable system.



Chapter 8

Block Ordering: The Work-Gated Producer Set

This chapter specifies how Producers are admitted, what they stake, what safety ordering provides, and what keeps the producer set federated.

8.1 Admission

The CometBFT engine runs unmodified: proposer rounds, precommits, and deterministic instant finality [6]. Only admission changes. `x/producer` replaces the stock delegated-stake module, and a node joins only with work credit above an eligibility floor and a standing producer-bond, plus a top-N ordering rank and the epoch gate; neither credit nor bond suffices alone. The set is bounded and epoched, with membership and weight fixed per epoch, and each height's proposer is chosen by the unbiasable beacon so no Producer can grind for heights. At genesis the set is seated from a publicly disclosed allowlist that decays toward open admission as a measured floor of open, non-allowlisted, honest, bonded Producers is reached, not on a calendar. The allowlist is a third distinct authority, with no cross-authorization with either bridge authority (Section 14.2).

> ***Definition 8 (Ordering weight).*** The voting power of Producer p in an epoch is $v_p = f(w_p, b_p)$, a function of its work credit w_p and producer-bond b_p, fixed at the epoch boundary and never a function of delegated stake. The form of f is open.

8.2 The producer-bond

The producer-bond is standing, slashable capital, and therefore economically a stake: stakelessness holds for the mining layer only, and the claim of zero stake anywhere is false. It is slashable only on equivocation (two conflicting signatures at one height and round), the one self-proving, objectively attributable ordering fault; invalid-block, liveness, and censorship faults are answered by weight decay and non-re-seating unless they become reducible to CometBFT-native evidence. The bond earns nothing and is an accountability floor, not a capital moat, sized not to bar entry or recreate an honest-majority-of-stake model. Its magnitude is open. Slashed value goes to burn or a non-committee insurance pool, never to other Producers, which would reward manufactured equivocation claims. The unbonding delay satisfies

$$\delta_{\text{unbond}} > \delta_{\text{evidence}},$$

where δ_{evidence} is the window for detecting and adjudicating equivocation, so no double-signer exits before its slash resolves.

8.3 Safety and rental resistance

Proposition 6 (Ordering safety). Let $V = \sum_p v_p$ be an epoch's total voting power and V_{byz} the power of Byzantine Producers. If $V_{\text{byz}} < V/3$, no two conflicting blocks are finalized at one height, and blocks continue to finalize once the network is synchronous.

Argument. This is the standard BFT bound [5, 6, 7] over weighted power. Finalization needs precommits from more than $2V/3$; two such quorums intersect in more than $V/3$, which includes an honest Producer, and no honest Producer precommits two conflicting blocks at one height. Liveness follows from the engine's rounds under partial synchrony. □

The bound is on power, not headcount, and a minimum set size N_{min} open limits power concentration. Because weights are fixed at the epoch boundary from previously accrued credit, no attacker can rent compute mid-epoch to swing that epoch: the precise claim is spot-rental resistance within an epoch, and patient multi-epoch rental and self-farming are not yet resisted. Three benchmarks must close before the set opens past its federated allowlist: (1) cost anchoring, so farming the credit that seats a Producer costs more than the weight it buys; (2) the multi-epoch rental bound, how many epochs of rented accrual buy a power supermajority and at what cost; and (3) beacon-grinding resistance of the proposer beacon. None has been run; until all three close the set stays federated and allowlisted, a condition rather than a calendar.

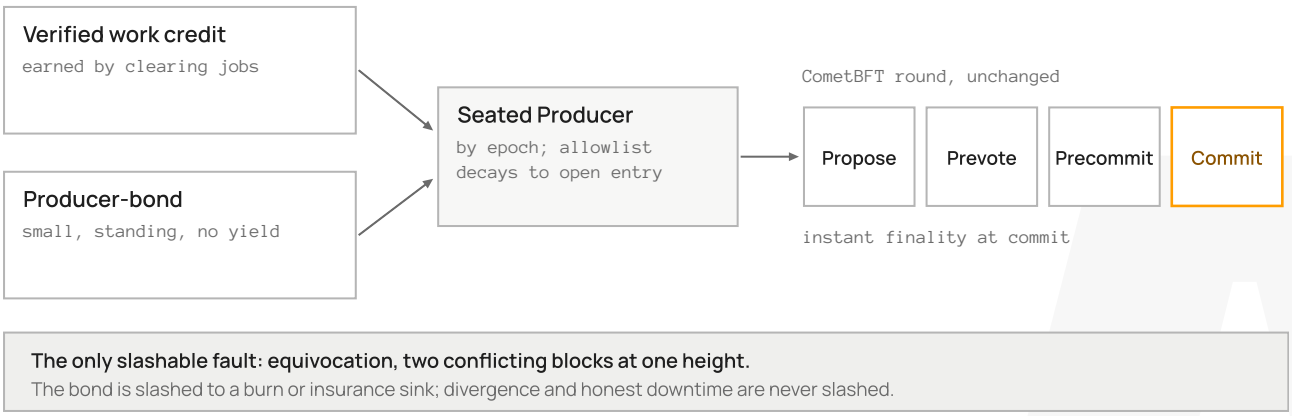


Figure 6. Block ordering. A Producer is seated by verified work credit together with a small standing producer-bond. Ordering runs unchanged CometBFT rounds, and equivocation is the only slashable fault.

Figure 6 summarizes the layer: credit and a bond seat a Producer, CometBFT finalizes blocks, and only equivocation reaches the bond.

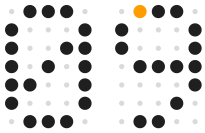
8.4 The work-credit firewall, the bond inventory, and rewards

Work credit gates committee seating and producer admission but confers no authority across them: credit alone does not seat a Producer, and producing grants no committee authority beyond earned credit. The protocol holds two bonds and one burn, none substitutable for another (Table 7); `x/producer` cannot touch the burn, `x/registry` cannot touch the producer-bond, and the burn is never ordering weight, because it is destroyed.

Table 7. The bond inventory: two bonds and the registration burn.

Value	Layer	Held?	Lost on	Destination
Producer-bond	Ordering	Standing	Equivocation only	Burn or non-committee insurance pool
Creator-bond	Registration of models, corpora, adapters	Held, with unbonding delay	Unavailability; misrepresentation once an arbiter exists	Burn or non-committee insurance pool
Registration burn	Mining	Destroyed on payment	Ejection or eviction (identity lost)	Already destroyed

Producers earn a work-gated block reward of gas, a producer fee share from the protocol slice, and a bounded bootstrap emission from the mining-incentive pool until fees carry ordering (Section 13.5). It is earned by producing blocks, never a return on the bond, and never expressed as a yield or APY.



Chapter 9

Verification Modes and Assurance Classes

This chapter defines the public verification mode, the private attested mode, the attestation module, and the assurance classes that label every result.

9.1 Why two modes

> ***Proposition 7 (Confidentiality excludes recomputation).*** No job can both keep its input confidential from the committee and be verified by committee recomputation.

> *Argument.* Recomputation requires each member to evaluate the model on the exact input, and a member without the plaintext cannot compute the hash it must compare. A job is therefore public and verified (Mode A) or private and attested (Mode B), and privacy costs the correctness backstop. □

A model registers an allowed-mode set and the caller picks a mode per job by privacy, latency, and price; no caller can retroactively privatize a public-only model or demand recomputation certainty on a confidential job. The mode fixes the job's assurance class, stamped into its record and price.

9.2 Mode A and Mode B

Mode A is the default, the flagship, and the only path whose results are called verified: the mechanism of Chapters 5 and 6, rooted in AIOS's own recomputation and independent of any vendor. It proves provenance of execution to a byte-identical hash or, in the sampling class, to a stated statistical standard; it does not prove confidentiality, since the committee sees the plaintext, and never proves an output true.

Mode B became a first-class second mode on 2026-06-19. It is subordinate, never secures consensus, and is never called verified. A Mode B job runs in a TEE-capable GPU enclave on a confidential-compute host, which loads the registered weights, checks their hash inside the enclave, runs the model on inputs the operator never sees, and signs an attestation binding the enclave measurement, weights hash, input hash, output hash, a fresh per-job nonce, and the chain height; `x/attest` verifies it instead of recomputing. Mode B proves confidential, attested execution, not correct arithmetic, and a compromised enclave's valid attestation is indistinguishable from an honest one's. Its trust root is the silicon vendors,

their attestation services, and a value cap, explicitly weaker than recomputation. It serves two needs recomputation cannot (confidentiality for healthcare, finance, legal, and proprietary-model operators, and private execution of real FP16 models), so it is additive, not a downgrade.

9.3 Sub-modes

Table 8. Mode B sub-modes, in decreasing defensibility.

Sub-mode	What runs	Independent check	Registry label	Standing
`tee-fastpath`	A deterministic model on the pinned integer Module, for latency and optional privacy	Yes: a VRF-selected recomputation back-check on a sampled fraction; recomputation stays the root	`tee-attested-backchecked`	First-class; the defensible near-term path
`tee-private`	A confidential job, possibly FP16, settled by a multi-vendor quorum	No; correctness rests on the quorum	`tee-confidential-attested`	First-class for deterministic models; FP16 only after an enclave-disagreement comparison rule is specified; hard value-capped
`tee-attested`	Attestation only; no recomputation or confidentiality requirement	No	`tee-only`	Deferred for silent-forgery risk; later only hard value-capped

Table 8 lists the sub-modes. FP16 needs a comparison rule because two honest FP16 enclaves may not agree byte for byte. Attestation honeypots police the pipeline but cannot catch a correctly attested yet wrong FP16 output, and a quorum of broken enclaves agreeing on a forgery is indistinguishable from an honest one.

9.4 The `x/attest` module, quorums, and caps

`x/attest` is a consensus and value-release surface, review-gated before any implementation and re-audited when scaffolded. For each quote it (1) walks the certificate chain to a governance-pinned vendor root in chain state; (2) checks the enclave measurement against a multi-vendor allowlist and, by golden-value comparison, the registered reference; (3) rejects revoked measurements and stale firmware; (4) checks that weights hash, input hash, nonce, and height match the issued job, closing swaps and replay, with AIOS issuing the nonce because enclaves are assumed stateless; and (5) above the single-enclave cap, requires M valid attestations across N independent vendors. Vendor revocation and reference lists are mirrored into chain state by governance, off the liveness-critical path: on vendor or mirror failure the chain degrades to Mode A or re-routes and refunds, and never halts. No validator, VRF, or signing key transits an enclave for custody, and stored reports are scrubbed of key material.

A single-vendor TEE root is a Hard Stop. TEE.Fail forged quotes for under 1,000 US dollars with physical access [11]; the quorum raises the bar to breaking several vendors at once, bounding the risk without removing it. The value cap on the no-backstop paths, `tee-private` over FP16 and `tee-attested`, is enforced at submission and is their single load-bearing control: removing it is a Hard Stop, and jobs above it need a recomputation-rooted mode. Attestation faults resolve by dispute-then-penalty or re-route, never objective forfeit; running revoked firmware is ejectable through the subjective tier. TEE-capable hardware is a centralized premium minority, and Mode B is never called decentralized.

9.5 Assurance classes

Every result carries one of four public assurance classes, ordered by the strength and independence of the evidence behind them:

$$\text{verified} \succ \text{sampling} \succ \text{attested} \succ \text{self-verifiable}.$$

Table 9. Assurance classes.

Class	Establishes	Trust root	Registry sub-labels
`verified`	Provenance of execution by byte-identical committee agreement	AIOS committee recomputation	`byte-identical-certain`
`sampling`	Provenance of execution to a stated detection probability	AIOS sampled recomputation or statistical audit	`statistical-re-exec`
`attested`	Confidential execution of registered code in a genuine enclave; not correctness	Silicon vendors, attestation services, value cap	`tee-attested-backchecked`, `tee-confidential-attested`, `tee-only`
`self-verifiable`	That a recall ran over a committed memory version, recomputable by its owner; the network does not attest correctness	The owner, optionally a TEE	Private memory and private adapters

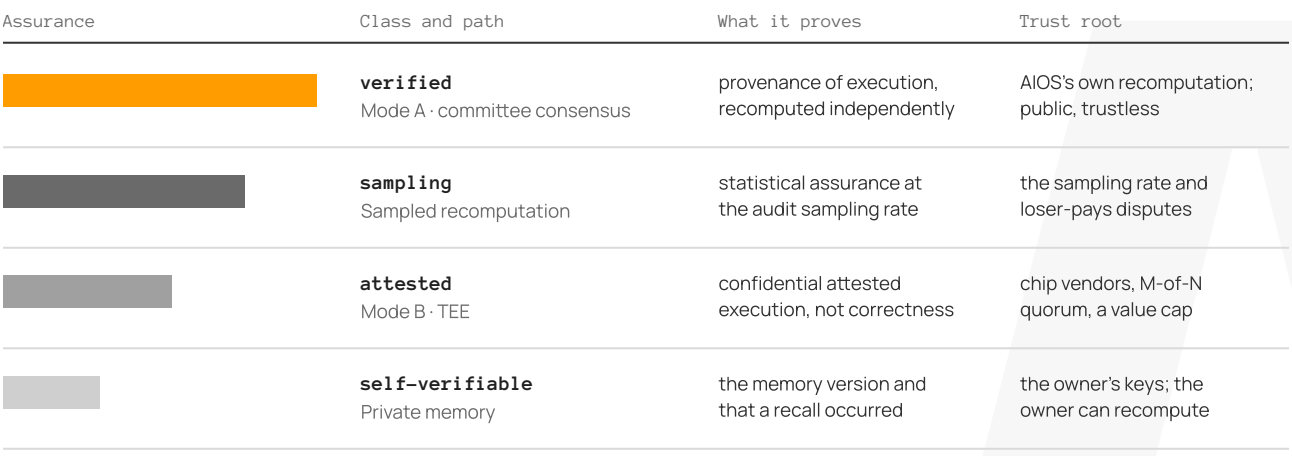
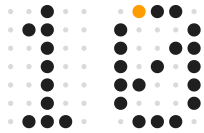


Figure 7. The four assurance classes. Only the Mode-A committee-consensus path is labelled verified; each other class states a narrower property and rests on a different trust root.

Table 9 and Figure 7 set the classes side by side. The order ranks how independent of any single party the evidence is, not output quality, and only the first class is ever called verified.

Three invariants hold by construction. Verified is reserved for the Mode A committee path, so TEE results are attested and private-memory results self-verifiable, never third-party verifiable. Mode B labels are a label space disjoint from Mode A's, enforced in schema, so no floating-point or confidential path can emit `byte-identical-certain` or `statistical-re-exec`, and every Mode B record carries a mandatory label stating that it is not re-execution-verified and rests on a hardware-vendor trust assumption. And the class is shown before submission and priced, weaker classes pricing lower. Section 12.3 covers sessions that mix classes.



Chapter 10

Native Models, the Model Internet, and the Adapter Registry

This chapter specifies models held in chain state, the registry that records every model and adapter, and the adapter layer that makes fine-tunes owned, verifiable assets.

10.1 The Neural VM and Native Models

The Neural VM, `x/neuralvm`, executes integer-quantized models as first-class chain objects. A Native Model is state: its quantized weights live in chain state, and a call advances it. Weights are programs. The ceiling is physical: weights occupy about $Nb/8$ bytes for N parameters at b bits each, so at 4 bits a 7B model needs about 3.5 GB and a 70B model about 35 GB, far beyond a sane state budget, and in-consensus execution costs gas in proportion to the arithmetic. Frontier-scale models cannot run in consensus, now or later; natively trained ternary models at about 1.58 bits per weight [4] are the frugal end of that ceiling, not an escape from it.

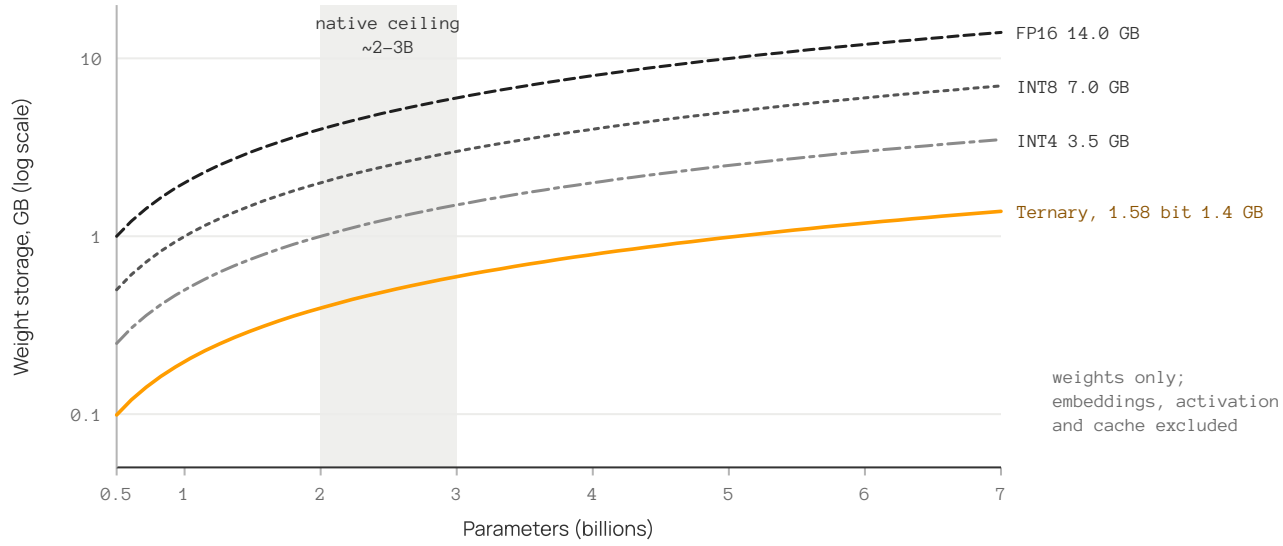


Figure 8. Weight storage by numeric precision. Ternary weights at 1.58 bits make a 2-3B model a sub-gigabyte object, which is where the native ceiling sits; the binding limits are chain-state size and per-block compute, not model quality.

Figure 8 makes this concrete: a 7B model takes about 14 GB at FP16, 7 GB at INT8, 3.5 GB at INT4, and 1.4 GB ternary, and a 2-3B ternary model roughly 0.4-0.6 GB. The credible ceiling for a Native Model is a genuinely useful natively trained ternary model of about 2-3B parameters: BitNet b1.58 2B4T is benchmark-competitive at its size [20], and openly released 3B ternary checkpoints fit in under 1 GB. The binding constraints are state size and per-block compute, not model quality, and native ternary models of 7B and above are unshipped. The realistic onchain class is small and specialized: classifiers, routers, small reasoning and embedding models, and judge models callers may invoke (never a consensus judge, Section 5.7). The Phase-0 reference keeps its in-block tier at the small end and runs its 2B-class tier off-chain (Section 15.4).

The Neural VM supplies oracle-free honeypot ground truth and runs deterministic retrieval over Native Corpora (Chapter 11). Its threats are determinism escapes (Section 4.3), gas exhaustion by weights cheap to register and expensive to run, and state bloat from large blobs; a determinism escape is a serious incident but costs honest Miners reward, not principal. Larger models run off-chain under selective recomputation or statistical audit, and only their class-stamped results enter blocks; the two paths are never flattened into one claim that a language model runs onchain, verified.

10.2 The Model Internet and the creator-bond

The Model Internet, in `x/registry`, gives every model a permanent onchain record (Table 10).

Table 10. The model record.

Field	Content
Address	Onchain identity in the Bech32 `aios1...` namespace
Weights content hash	Hash of the quantized weights, with a pointer to content-addressed storage or a Miner content network
Price	Per-call price in $AIOS, set by the creator, priced by assurance class
Verification modes	Allowed-mode set. Mode A: `native-in-VM`, `full-re-execution`, `selective-re-execution`. Mode B: `tee-fastpath`, `tee-private`, `tee-attested` (deferred)
Assurance class	Mandatory, machine-readable, shown before submission
Owner and creator-bond	Who controls and earns from the model, and the bond behind it
Standing	Bond size, slash history, served volume

Members check fetched weights against the registered hash before recomputing, which closes weight substitution. Weight availability, creator-pinned and protocol-replicated, is a registration precondition; weights not verifiable by the deadline refund the caller and penalize the creator-bond, a fault distinct from any Miner's. The cold-load cost of large weights is not yet priced and who serves them at scale is open. Creators may deregister or re-price, in-flight jobs keeping their terms; calls from other chains are future work.

Registering a model, corpus, or adapter posts a creator-bond from one class: a floor plus a component scaling with stated price and served volume, both open. Proof of Inference proves the registered weights ran correctly, not that they match their label; registered is not endorsed. The bond is designed to be forfeitable for misrepresentation through an optimistic challenge with a challenger reward and loser-pays for frivolous claims, a subjective tier. At launch, however, AIOS ships no fraud arbiter: the misrepresentation slash is deferred, the bond is only a registration cost and spam deterrent, and the arbitration authority is open. The unbonding delay still exceeds the challenge window, and forfeits go to burn or a non-committee insurance pool.

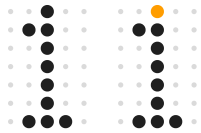
10.3 The Adapter Registry

Low-rank adaptation [21] puts no large model onchain, and the ceiling of Section 10.1 is unchanged. It offers the cheapest granularity of ownable intelligence: a fine-tune is a delta of a few megabytes (a rank-8 adapter on a 2-3B base is about 2-11 MB), so a shared base with swappable adapters makes owned specializations cheap and verifiable. The layer is locked as a draft (2026-08-23) and review-gated before implementation.

Definition 9 (Adapter). An adapter is an asset in `x/registry` with an address, owner, price, verification mode, delta content hash and pointer, and a `base_model_hash` binding to a registered base; an unregistered base is rejected. Small-rank deltas may sit in chain state beside a Native Model, larger ones as a commitment and pointer. An adapter posts the same creator-bond as a model or corpus; there is no new bond class.

For base weights W and an adapter of rank r with integer factors $A \in \mathbb{Z}^{r \times d}$ and $B \in \mathbb{Z}^{d' \times r}$, the composed layer computes $Wx + s \cdot B(Ax)$ with a pinned fixed-point scale s, served unmerged so the delta stays swappable. The committee recomputes the composition and commits $H(\text{output} \,\|\, \text{model commitment} \,\|\, \text{adapter commitment} \,\|\, \dots)$, which subsumes the retrieval manifest when a corpus is bound; a member that composed a different adapter or output is unpaid.

The `verified` class requires an integer or quantized adapter (BitLoRA and LoraQuant class) whose multiplication and composition are pinned in the Module; FP16 adapters register only as attested or statistical, never verified. A personal fine-tune is the weights analog of Verified Memory (Section 11.6): owned, encrypted, portable, self-verifiable by its owner, optionally attested, never committee-verified, and, once published, earning an adapter-royalty (Section 13.4). Composition beyond one adapter, by routing and stacking (mixture-of-LoRA) or merging (TIES [22] or DARE), is a labeled, approximate product whose determinism is enforceable but whose quality is not: registered is not endorsed, and verified is not correct. An adapter on a 2-3B base is still a 2-3B model, adding no scale or knowledge, so frontier-class capability through onchain adapters is not on offer, and adapter determinism depends on an integer kernel validated before mainnet.



Chapter 11

The Data Internet and Verified Memory

This chapter extends Proof of Inference from the model to the context it executes on: registered corpora, verifiable retrieval, and owned memory.

11.1 The gap and the corpus record

Many model calls are grounded on context retrieved from a corpus [23]. The Model Internet proves the model and Proof of Inference the compute, but neither proves which documents were retrieved, from which corpus, and whether faithfully. The Data Internet pulls retrieval inside the committee, reusing the lifecycle, registry pattern, committee, and determinism engine; it was locked on 2026-07-09 and is review-gated before implementation. A corpus, meaning a dataset or knowledge base, registers through `x/data` like a model (Table 11).

Table 11. The corpus record.

Field	Content
Address and owner	An `aios1...` address and its accountable owner
Index commitment	Merkle root over the content-addressed chunks and their integer embeddings
Retrieval pointer	Off-chain storage location; raw bytes never go onchain
Retrieval specification	Embedding-model address, number of chunks retrieved, exact-index type (flat inner-product kNN or BM25), similarity metric, tie-break rule; joins the corpus commitment
Price	Per-call data price in $AIOS, set by the owner, with no oracle
Verification mode	`verified` (exact), `sampling` (statistical), or `attested` (confidential)
Creator-bond	The asset creator-bond class, extended to corpora

11.2 The grounded pipeline

A grounded job binds a model, a corpus, a query, and parameters. Every seated Miner (1) tokenizes the query with a pinned tokenizer; (2) embeds it as an integer vector; (3) retrieves the top k chunks by exact kNN or BM25 over the committed index, breaking ties by chunk

content hash; (4) generates with the registered model on the query and retrieved context; (5) commits and reveals a hash over output and manifest; and (6) settles with the corpus and retrieval commitments beside the model commitment.

> ***Definition 10 (Retrieval manifest).*** The retrieval manifest of a grounded job is the ordered list $(H(c_1), \ldots, H(c_k))$ of the content hashes of the chunks used as context. Each member commits to $H(\text{output} \,\|\, \text{manifest})$.

A Miner that retrieved different chunks or generated a different output is unpaid, and no supermajority means re-route and refund: one committee, one commit-reveal, one settlement. Ungrounded jobs are unchanged.

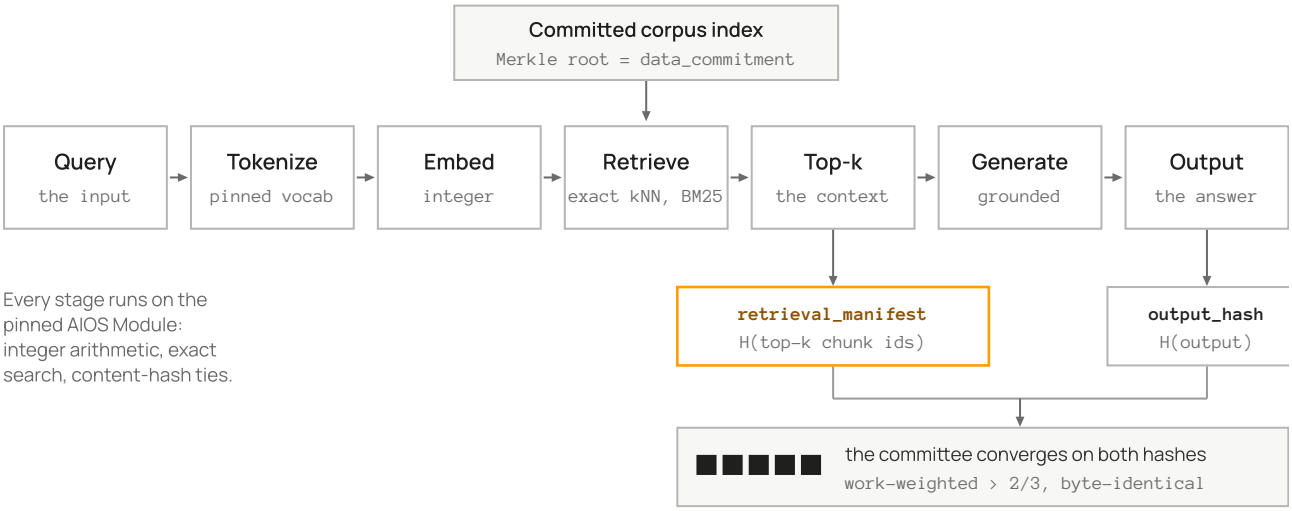


Figure 9. Verifiable retrieval. Every stage of the grounded pipeline runs on the pinned AIOS Module, so the committee can converge on the retrieval-manifest hash as well as the output hash: the context is proven, not only the model.

Figure 9 shows the pipeline. Output and manifest are bound into one committed hash, so no Miner can agree on the answer while disagreeing about its context.

11.3 Deterministic retrieval

Retrieval is committee-checkable only if embedding and retrieval are byte-identical across hardware. The Module pins integer embedding kernels, an exact-retrieval kernel (exhaustive inner-product kNN, or BM25 with fixed-point scoring [24]), and total-order tie-breaking by chunk hash. Approximate nearest-neighbor indexes, including HNSW [25], IVF, and LSH, are banned on the verified path, being nondeterministic across build order, thread count, and hardware. The kernels and retrieval specification join the Module hash, and a deterministic integer embedding reference model must land before any `verified` mode in `x/data`.

Proposition 8 (Determinism of exact retrieval). Suppose the query embedding $q \in \mathbb{Z}^d$ is computed byte-identically, each chunk embedding $x_c \in \mathbb{Z}^d$ is fixed by the index commitment, scores $s(c) = \langle q, x_c \rangle$ are integer under the conditions of Proposition 1, and search is exhaustive. Then ordering by $s(c)$ descending and, on ties, by $H(c)$ ascending is a strict total order (absent hash collisions), and the top-k manifest is identical on every honest Miner.

Argument. Proposition 1 makes scores platform-independent, exhaustive search removes traversal-order dependence, and distinct hashes break every tie, so the first k elements are determined. The argument extends to BM25 scored in pinned fixed point. □

11.4 Regimes, faults, and the data role

On the exact path (`verified`), integer embeddings and an exact index yield a byte-identical manifest and output, with objective no-pay on mismatch. Where float embeddings or approximate retrieval are unavoidable, the manifest is checked by seed-synchronized sampled recomputation at an open rate (`sampling`, never exact-hash). A confidential corpus runs retrieval in a TEE under Mode B (`attested`), value-capped and quorum-gated. The deterministic-retrieval harness is a hard pre-mainnet blocker and the Data layer's highest-priority gap: until it passes, `x/data` offers statistical retrieval verification only.

A corpus index not verifiable by the deadline refunds the caller and penalizes the corpus creator-bond, the unavailability fault generalized to data. Miners gain a data role, pinning, serving, and proving corpora: retrieval recomputation is rewarded on match, while availability, a liveness property, is rewarded for passing random-access challenges over the committed index, keeping liveness separate from correctness. The role rides the existing identity and burn; a lazy or malicious data-miner earns nothing and is ejected on proven fraud or a failed retrieval honeypot. Data-serving rewards come from grounded-call fees, not the mining-incentive pool.

11.5 Data economics and the provenance graph

A grounded call's fee splits among Miners, model creator, corpus owner, and protocol. The owner's data-royalty is additive and never carved from the Miner share (Section 13.4), so whoever curated the knowledge is paid on every grounded call that provably retrieves from it: data attribution and compensation answered by construction. Every output commits to its lineage, so the chain accumulates a provenance graph from any output to its base model, adapter, corpus, and exact chunks; the indexer that exposes it is deferred to Phase 2.

11.6 Verified Memory

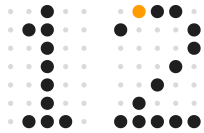
Verified Memory is not a new module but `x/data` with a visibility flag: a corpus is public (listed, priced, verified under Mode A, earning royalty) or private (owner-scoped, encrypted, unlisted), with the same commitments and pipeline.

- **Structure.** Long-term memory (conversations, facts, documents) is chunked, embedded, and indexed as a personal vector store, append-only and Merkle-committed, so its state at any height is provable; only roots of about 32 bytes and encrypted pointers touch chain state.
- **Ownership.** Memory is encrypted under its owner's keys, never held in plaintext by the network, and portable, exportable, and revocable: no provider can read it, lose it, or lock its owner out.
- **Scopes.** User-defined session scopes (projects) are versioned sub-corpora, each a retrieval boundary, so work and personal memory never bleed, and a permission boundary granting agents scoped, capped, revocable access: access, never custody.
- **Privacy.** Committee verification would expose plaintext, so private memory is self-verifiable: each recall commits its memory-version root and manifest, and the owner, holding the keys, can recompute it. A TEE may attest a recall to a third party under Mode B. Private memory is never committee-verified; shared or published memory is verified under Mode A and earns royalty.
- **Economics.** Storage is an off-chain pinning cost AIOS never subsidizes; one's own recall costs the inference fee (model price plus gas), with no self-royalty; publishing earns royalty.

Verified Memory powers the Brain capability of AIOS:ONE, the AIOS:RECALL workflow, and verifiable agent continuity.

11.7 Limits

Verified data means provenance and integrity, never that data is true, clean, current, or unbiased: faithfully retrieved garbage is verifiably garbage. Retrieval integrity is not relevance, since consensus guarantees the same top k chunks, not the best ones. Deterministic retrieval is an unvalidated target of the Module's class, and corpora or embedding models that cannot be made byte-deterministic fall to statistical verification. Data lives in the model and inference layer (Native Corpora in state, large corpora committed onchain and stored offchain); AIOS is not a data-availability, blob-storage, or data-marketplace layer, and no oracle or live feed sits on the verified path. Verified Memory proves what was stored and retrieved, never truth, and is not onchain storage.



Chapter 12

The Job Lifecycle and the Inference Receipt

This chapter traces a job end to end and specifies the record it leaves.

12.1

Lifecycle

A job moves through seven steps. (1) **Submit**: the caller names the model, input, decoding specification, maximum price, mode, and any corpus or adapter, pays gas to the producer set, and escrows the worst case from a published cost curve keyed to model and mode. (2) **Route**: the beacon seats a performer and a committee, which need only active identities, and members verify fetched weights and any corpus index against registered hashes. (3) **Perform**: the performer runs the job on the Module, in `x/neuralvm` for Native Models. (4) **Verify**: members recompute or sample, commit, and reveal. (5) **Settle**: matching Miners, the creator, and any corpus or adapter owner are paid, the protocol slice applies, unused escrow returns, proven fraud ejects with lost value destroyed, and the result and tally enter a block. (6) **Fault**: non-response re-routes or refunds, and accept-then-withhold costs reward and weight, with ejection only for sustained griefing. (7) **Unavailability**: unverifiable weights or corpus index refund the caller and penalize the creator-bond. The caller pays the model price, any corpus and adapter prices, and the realized base fee, never more than the escrow.

12.2

The Inference Receipt

Every job produces an onchain record, the Inference Receipt, whose public name is the proof and whose API name is the verification record.

> ***Definition 11 (Inference Receipt).*** The Inference Receipt of a job contains the job identifier and height; input and output hashes; the `model_commitment` (weights content hash), model address, verification mode, and assurance class; the decoding specification; every member's committed hash, the supermajority work weight, and each seat's match outcome; the cost breakdown; and the caller identity. A grounded job adds the `data_commitment` (corpus index root) and the `retrieval_manifest`, and a job composing an adapter adds the `adapter_commitment` (adapter hash and `base_model_hash` binding).

Definition 12 (Provenance of execution). A receipt establishes provenance of execution when its tally has a canonical hash in the sense of Definition 6 binding the output to the recorded model, adapter, corpus, manifest, input, and decoding commitments; it then records which base weights, fine-tune, corpus and chunks, and input produced which output, under which assurance class.

Raw inputs and outputs never go onchain; the receipt carries authoritative hashes and a content pointer, and fetched bytes are trusted only after re-hashing. Private content never leaves the enclave. Every surface renders the same record: a chat turn in AIOS:ONE, an agent step in AIOS:CODE, a tool response from AIOS:MCP, and a job page at `/job/{job_id}` on AIOS:EXPLORER.

Inference Receipt committed per job

`job_id, height`	the job and its place in the chain
`input_hash`	the input, hashed; content stays off-chain
`model_commitment`	content hash of the registered weights
`adapter_commitment`	adapter hash and its base_model_hash, when composed
`data_commitment`	corpus Merkle index root, when grounded
`retrieval_manifest`	ordered hashes of the top-k chunks, when grounded
`output_hash`	the canonical output
`committee`	seated Miner IDs, commits, reveals, per-seat match
`tally, class`	matched work over committee work; the assurance class

Figure 10. Fields of the Inference Receipt, committed for every job. Content stays off-chain; the hashes are authoritative, and fetched bytes are trusted only after they re-hash to the committed values.

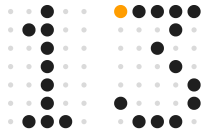
Figure 10 lays out the receipt: data, model, adapter, and output, bound by one tally at one height.

12.3 Sessions and boundaries

A session, such as a conversation or an agent run, has the assurance of its weakest job under the order of Section 9.5, never an average:

$$A(S) = \min_{j \in S} A(j),$$

so a private or attested step cannot be laundered into a verified session. Mode A inputs and outputs are visible to the committee and may appear onchain. The record does not yet price weight serving, and it proves execution, not truth: evidence of which computation ran, never an endorsement of what it said.



Chapter 13

$AIOS Economics

This chapter specifies the token's roles, supply, distribution, fees, rewards, the staking pool, and how value accrues.

13.1

Roles and supply

$AIOS is the native coin, with seven roles: gas (registering models, corpora, adapters, and Miners, submitting jobs, calling models); inference payment; the mining reward; the block-producer reward; the data-royalty; the adapter-royalty; and ownership of the asset, one's registered assets, and one's keys, with no voting rights. The producer-bond and registration burn are at-risk uses, not payments. $AIOS is the unit of verified cognitive work.

The maximum supply is 300,000,000 $AIOS `locked`, minted once, in full, at the Phase-1 deployment of an ERC-20 on Ethereum mainnet that has no mint function, is non-upgradeable, and is not pausable; there is never protocol minting on Ethereum again. After L1 genesis the native chain is the sole mint authority, callable only by the audited bridge verifier against a verified, finalized, unconsumed lock of equal amount (Chapter 14). Emission and staking rewards are releases from pre-allocated buckets and royalties are fee transfers, so no other mint path exists. Burns (the protocol fee burn, registration burns, burned bond forfeitures) retire native supply permanently: circulating supply is 300,000,000 minus cumulative burns and only falls.

13.2

Distribution

Table 12. Genesis distribution `locked`.

Bucket	Share	$AIOS	Lock and release	Purpose
Liquidity	50%	150,000,000	Time-locked LP position	Self-funded liquidity on Ethereum mainnet
Mining-incentive pool	33.33%	100,000,000	Contract-locked, decay-released, verified-work-gated	Bootstrap mining and ordering rewards
Staking pool	10%	30,000,000	Contract-locked, time-bounded decaying emission	AIOS:STAKING rewards; secures nothing
Team	6.67%	20,000,000	12-month cliff, then 36-month linear vesting	Long-vested team allocation

In Table 12 the token amounts are canonical and sum exactly to 300,000,000; 33.33% and 6.67% are rounded labels for exactly 100,000,000 and 20,000,000. There is no presale, no airdrop at launch, and no treasury bucket; the operator self-funds the build from off-protocol funds and Phase-1 tax revenue. Both pools sit at identified, contract-locked, non-trading addresses with no operator-discretionary withdrawal, the mining pool releasing only on its decay schedule against verified work and the staking pool only as staker rewards, and neither can fund operations. Mining-pool releases are security budget; the staking pool buys no security.

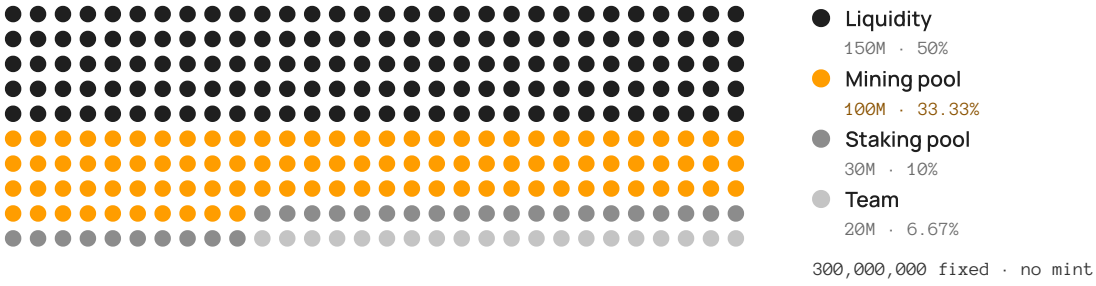


Figure 11. Genesis distribution of the fixed 300,000,000 $AIOS supply, one dot per million. There is no mint function and no treasury allocation.

Figure 11 shows the split: half is liquidity, a third is the bootstrap security budget, and the smallest bucket, the team's, pays nothing for a year.

13.3 The Phase-1 transfer tax

Ethereum ERC-20 transfers carry a flat 1% tax `locked`, also an immutable ceiling: a monotonic setter can only lower it toward zero, and no upgrade path can raise it. The exemption set comprises the liquidity pair, the router, the bridge lock, mint, and burn contracts and any relayer hop, the tax-collector sink, the AIOS:STAKING contract, and the LP-lock contract and deployer during seeding; changes are multisig-gated and emit events, so an inserted exemption is observable. Tax accrues in $AIOS and is swept, never auto-swapped, to a multisig distinct from the deployer and admin key (1-of-1 at launch, a disclosed centralization), keeping a reentrancy and sandwich surface off the transfer path.

13.4 Fees, royalties, and pricing

A call costs the creator's model price plus a protocol base fee (gas for submission, routing, commitment, and settlement), plus the corpus or adapter price where one is used. The fee splits about 75% to Miners on the canonical hash, 15% to the model creator, and 10% to the protocol `illustrative`; what is locked is that the shares sum to the whole fee with no gap, and the figures are re-peggable against measured recomputation cost, since the Miner share must cover honest recomputation or the security budget collapses. The protocol slice funds a burn (its largest part), the producer fee share, and insurance and honeypot funding under an open split: most of it is burned, not all.

Royalties are additive. A grounded call is priced higher because the committee also recomputes embedding and retrieval, and its fee splits among Miners, creator, corpus owner, and protocol under its own sums-to-the-whole invariant, the data-royalty never carved from the Miner share. The adapter-royalty is paid to a published adapter's owner on canonical match, likewise additive. Both are fee transfers, never mints, owners pay themselves nothing, and magnitudes are open. Pricing tracks the assurance class, weaker classes pricing lower, and Mode B fees must recover premium hardware and, for `tee-fastpath`, the back-check.

Version-1 prices are set by creators in $AIOS with no price oracle, keeping a trust dependency and manipulation surface off settlement; if $AIOS appreciates, competition with cheaper unverified APIs forces re-pricing (fewer tokens per job, each worth more), and sticky re-pricing is a real risk. A USD-denominated oracle is deferred until stale prices demonstrably cost volume, and would first need a threat model covering single-source feeds, manipulation-resistant aggregation, fail-closed staleness, settle-time front-running, and caller exposure bounded by escrow.

13.5 Mining and ordering rewards

A Miner earns two streams per matched job, paid per job and per individual: its share of the Miner slice, the long-term engine that scales with usage, and a per-job bootstrap emission from the 100,000,000 $AIOS mining-incentive pool, substituting for thin fees at cold start. The emission decays with fee revenue rather than the calendar, so reward does not collapse while fees are near zero, but the pool is finite and exhausts if fees never ramp. It pays only on canonical match, cannot be farmed by idle identities, and is calibrated jointly with the cost anchor, since wash-inference self-farming is an open blocker. The same pool funds the bootstrap ordering reward through an open split, so the two sinks trade runway. Releases move pre-allocated balance and never increase supply.

13.6 AIOS:STAKING

AIOS:STAKING is a DeFi lock-to-earn pool, not network staking. Staked $AIOS orders no blocks, gates no committee, carries no work credit, confers no producer eligibility, and is never slashed. The producer-bond is the chain's one security stake; AIOS:STAKING is a non-security emission incentive, and staking secures nothing.

A holder stakes, earns, unstakes, waits a 7-day cooldown without rewards, and claims. Principal is always eventually withdrawable: no administrative path can extend, block, or pause a cooldown, and after seven days the claim succeeds. The cooldown is an anti-dump measure, not a security requirement. Rewards come from the fixed 30,000,000 $AIOS pool, pro rata to stake and time, through a standard accumulator with continuous-time accrual (zero elapsed time earns zero), crediting the balance actually received, with cumulative emission never exceeding the pool.

The rate floats. With pool emission rate $\eta(t)$ and total stake $\Sigma(t)$, the annualized rate is approximately

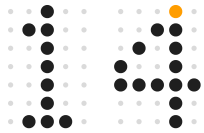
$$a(t) = \frac{\eta(t)\,\tau}{\Sigma(t)},$$

with τ one year, so more stake means a lower rate and liability is capped by the pool.

An early rate of about 10% is illustrative only and never guaranteed `illustrative`. Emission is front-loaded, decaying, and finite, ending when the pool is exhausted, with window, curve, and per-period magnitude open. The pool adds mild sell pressure and a temporary, non-security lockup: a bootstrap tool with an end date, not a value driver. Its contract is review-gated under conditions S1 to S3 (a non-discretionary pool that cannot fund operations, reentrancy-safe accounting that never locks or pauses principal, and framing enforced in copy).

13.7 Value accrual

Value accrues through usage: every verified call and chain operation is paid in $AIOS, most of the protocol slice is burned, and every registration burn retires supply. There is no staking-for-security demand, governance-token demand, or lockup-driven floor; the producer-bond is a small non-yield accountability lock and the burn a deflationary sink. The model is cleaner if inference demand materializes and more exposed to token-first bleed if not; security is the honest work rewards attract, not the value of locked tokens.



Chapter 14

Dual-Home Supply and the Bridge

This chapter specifies how $AIOS lives on two chains without its supply ever exceeding 300,000,000, and what the bridge secures.

14.1

Topology and conservation

$AIOS is permanently dual-homed: a fixed-supply ERC-20 on Ethereum mainnet and the native coin of the AIOS L1. Moving value to the L1 locks Ethereum-side tokens in the provider's vault and mints an equal native amount against the finalized lock; moving it back burns native tokens and unlocks the equal amount. Ethereum never mints or burns, the L1 is the sole mint authority, and the ERC-20 is never deprecated: this is not a two-way burn-and-mint bridge.

Let E_t be Ethereum circulating supply at block t (outside the vault), L_t the vault balance, N_t native circulating supply, β_t cumulative native burns, and $S = 300{,}000{,}000$. The ERC-20 total is fixed, so $E_t + L_t = S$, and the system maintains

$$E_t + N_t + \beta_t = S, \qquad \text{equivalently} \qquad L_t = N_t + \beta_t,$$

written in canon as `eth_circulating + native_circulating + cumulative_burned == 300,000,000`, exact to the smallest unit at every block.

> ***Proposition 9 (Conservation).*** Every admissible transition preserves the invariant.

> *Argument.* Lock-and-mint of a changes (E, L, N) by $(-a, +a, +a)$ and burn-and-unlock by $(+a, -a, -a)$; a native burn of b changes (N, β) by $(-b, +b)$; transfers, releases, and royalties move balances within E or N. Each leaves $E + N + \beta$ fixed, so only a mint without a finalized, unconsumed lock, or an unlock without a native burn, can break it. □

The invariant, with total native mints bound to total finalized locks, is monitored continuously; any upward divergence is the primary intrusion signal and halts the bridge. Circulating supply, $E_t + N_t = S - \beta_t$, is a deflationary ceiling that burns lower without tripping the alert.

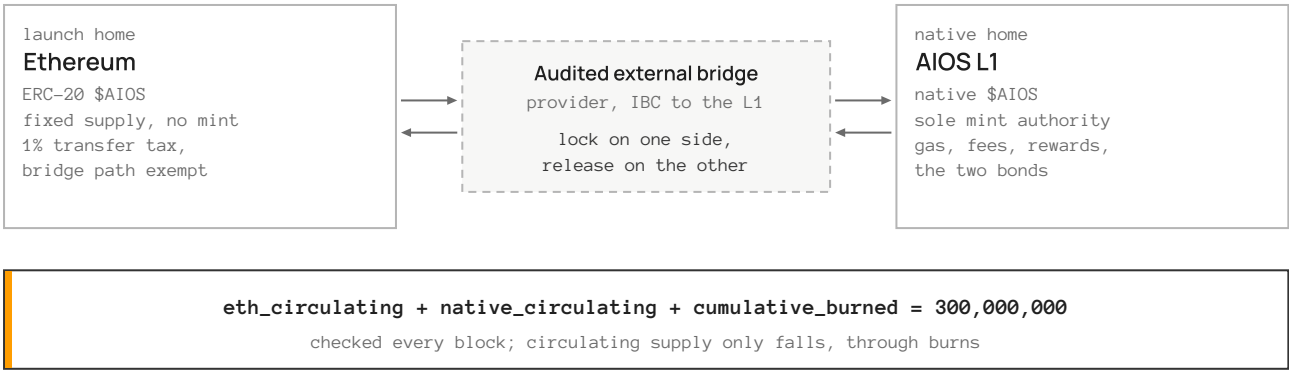


Figure 12. The dual-home supply. The ERC-20 on Ethereum and native $AIOS on the L1 are one asset: an audited external provider locks on one side and releases on the other, and the conservation invariant is checked every block.

Figure 12 shows both homes and both directions of flow: the bridge creates supply on neither side, and the one chain that mints does so only against a lock it can verify.

14.2 Mint authority and bridge conditions

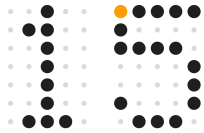
- **Mint authority.** Only the audited bridge verifier can mint, never the deployer, an externally owned account, or a multisig; there is no manual mint path, upgrade key over mint semantics, or single-signer custody, and it mints exactly the verified locked amount.
- **Finality asymmetry.** The outbound mint waits for Ethereum finality, since a reorganization could unwind a lock; the inbound unlock follows a CometBFT-finalized burn, which cannot reorganize.
- **Replay protection.** A consumed-message set, persistent across restarts, blocks attestation re-use.
- **Circuit breakers.** Per-transaction and per-epoch caps `open` gate both legs, are tightest at genesis, loosen only behind a timelock, tighten instantly, and halt the bridge when breached.
- **Halt-only control.** The bridge multisig can halt the bridge or a pending mint or unlock but cannot originate a mint, move locked funds, or choose a recipient; locked backing sits in the provider contract or light client, never an AIOS wallet.
- **Distinct authorities.** The one-time genesis attestation authority (a light-client proof preferred), the design's highest-value review item, is separate from the standing mint authority, neither authorizing the other, and retires after cutover.
- **Address validation.** The mint leg validates the Bech32 `aios1...` recipient before the lock finalizes and the unlock leg the EIP-55 recipient; with no native EVM module there is no second line of defense against a forged cross-VM message, so provider message authentication is a named selection-review item.
- **Tax neutrality.** Bridge legs are tax-exempt and mints are bound to the amount received. A provider assuming received equals sent without exemption is disqualified, as is one with single-signer mint custody, a stale or absent audit, or a mint-behavior upgrade key.

14.3 Genesis cutover and what AIOS does not build

At genesis the protocol buckets (the mining pool, the team allocation, and the staking pool if run natively) lock their Ethereum allocations and are minted one-to-one to native successor addresses, without a claim step. Holders bridge on demand, without deadline. The genesis lock-and-mint is gasless and abort-safe: no native mint finalizes before its lock, so value never exists on both sides, and an aborted cutover leaves every holder whole on Ethereum. AIOS builds no bridge cryptography (no bespoke bridge, light client, or signature scheme), since the bridge is the industry's highest-loss attack surface and not AIOS's edge. The Ethereum-to-native leg is cross-VM (`0x...` to `aios1...`) and cannot use IBC, since an EVM chain is not an IBC counterparty, so an audited external provider, chosen under review, handles it; links to other Cosmos chains use IBC.

14.4 Trust domains

The bridge is secured by its provider's trust set, a validator set or light clients, distinct from AIOS's consensus, and is never described as safe, trustless, or secured by AIOS. Single-asset scope reduces bridge risk without eliminating it, and a permanent dual-home bridge is more standing surface than a migrate-once design. Conversely, the inbound unlock trusts a native burn only as far as the producer set that finalized it, so a captured producer set could unlock tokens against a burn an honest chain would not produce; the bridge is not safer than the chain that finalizes it, and the bond, set bounds, caps, and conservation halt limit this without eliminating it.



Chapter 15

Governance, Parameters, and Roadmap

This chapter specifies how the protocol changes, which numbers are fixed, and the phased path to mainnet.

15.1

No token voting; upgrades by producer adoption

AIOS ships no token-weighted governance and no DAO: no coin-weighted vote on parameters, upgrades, or spending, and no protocol treasury. A foundation multisig publishes a versioned upgrade proposal with a migration height, and Producers adopt it by running the new version there; a Producer that does not is no longer producing on the canonical chain. The multisig proposes and coordinates but cannot force anyone to run code, so an upgrade is real only if the Producers securing the chain adopt it, the rough-consensus-and-running-code model of established public chains, including Ethereum.

This is also a determinism requirement: the Module hash changes only by adoption at a height, never by multisig hot swap or token vote, since a change 51% approve and 49% do not run would fork determinism. Consensus-affecting parameters (the Module hash, committee, VRF, and slashing parameters, the Miner soft cap and immunity period) change only by producer adoption, and loosening bridge caps also needs the bridge timelock. The mechanism is a consensus surface, review-gated before first mainnet use, and must abort cleanly below the adoption threshold and never leave two live forks.

15.2

Progressive decentralization and the halt-only backstop

At launch, upgrade authority concentrates in the core multisig, whose signers the operator controls: disclosed centralization and a single point of failure, since a compromised multisig could propose a malicious upgrade before the producer set can refuse it. Co-signers are added over time, and adoption power shifts to Producers as their set opens, on a timeline conditional on the unrun ordering benchmarks. A forfeit-veto backstop can halt a wrongful forfeit or ejection within its window and never originate one; it is operator-controlled at launch, so until co-signers join its holder could halt a forfeit affecting itself or an ally. This is disclosed, and it is not a governance vote.

15.3 Parameters and their status

Table 13 lists the parameters. Locked values are canon; open values are set near launch with a written rationale, by measurement where possible, and never guessed; illustrative values only show shape.

Table 13. Parameters and their status.

Parameter	Value or shape	Status
Total supply	300,000,000; no ERC-20 mint function; L1 sole mint authority	Locked
Genesis distribution	150,000,000 liquidity, 100,000,000 mining pool, 30,000,000 staking pool, 20,000,000 team (50 / 33.33 / 10 / 6.67%)	Locked
Team vesting	12-month cliff, 36-month linear	Locked
Transfer tax	1% flat; 1% immutable ceiling; lower-only ratchet	Locked
Launch sequence	Ethereum mainnet ERC-20, then the AIOS L1	Locked
Conservation invariant	eth_circulating + native_circulating + cumulative_burned = 300,000,000	Locked
Registration burn at genesis	R0 = 10,000	Locked
Burn schedule	B(e) = F + (R0 - F) (1 - e) ^gamma, times M, clamped to [F, Cap]	Shape locked
F, gamma, r*, Cap, multiplier bounds	F is a security floor	Open
Canonical supermajority	Above 2/3 of committee work weight	Canon target; calibration open `illustrative`
Committee size; per-entity cap; honest-compute floor	Bound capture (Section 6.1)	Open
Sampling rates; honeypot rate; audit detection target	Set statistical guarantees	Open
Value cap on PoI-settled jobs	Load-bearing; enforced at submission	Open
Maximum re-route count; worst-case cost curve	Size the escrow	Open
Dispute and challenge windows	Hard-capped	Open
Producer-bond; evidence window; minimum set size; epoch length	Unbonding delay exceeds evidence window	Open
Creator-bond floor and scaling	One class for models, corpora, adapters	Open
Miner soft cap; immunity period	Producer adoption only	Open
Mode B quorum M and N; enclave cap; no-backstop cap; back-check rate	Removing the value cap is a Hard Stop	Open

Parameter	Value or shape	Status
Fee split	About 75 / 15 / 10 (Miners / creator / protocol)	Illustrative
Protocol-slice split; producer fee share; royalty magnitudes	Royalties additive to the Miner slice	Open
Bootstrap emission and decay; mining and ordering split	Fee-conditioned shape locked	Open
Staking rate	About 10% early, floating	Illustrative, never guaranteed
Staking cooldown	7 days	Locked
Staking window, decay, per-period emission	Front-loaded, decaying, finite	Open
Bridge caps (per transaction, per epoch)	Tightest at genesis	Open
Cost-anchor ratio for selection credit	Credit costs more than the advantage it buys	Open

15.4

Roadmap

The sequence is proof-first: the mechanism is demonstrated before any token exists, and each transition is conditioned on evidence, not a date.

Phase 0: proof-testnet. A lean technical proof on the operator's own and allowlisted nodes, with no token, no $AIOS spent, no points program, and no mass onboarding. The first milestone is the standalone reference demo: publish the pinned runtime specification, port an open ternary checkpoint through AIOS's own pure-integer Rust forward pass and freeze it by content hash, build the prototype Module and agreement harness with its negative test, and produce the cross-machine byte-identical log under gates G1 to G4 (Section 4.5). The reference model has two tiers: Tier A, an in-consensus toy from the small end of the openly released ternary model class, the Neural VM's identity piece in chain state; and Tier B, an off-chain showcase of the BitNet b1.58 2B4T class on Miner hardware [20]. Neither uses bitnet.cpp or GGUF runtimes, which accumulate in FP32 with float scales and keep non-linear operations in floating point, so cannot recompute byte-identically; AIOS builds the runtime and one reference model, not a model zoo. The testnet then builds the module spine (`x/registry`, `x/data`, `x/inference`, `x/producer`, `x/neuralvm`), the verification stack with inference and retrieval honeypots, and the Sidecar Mesh, tested against retrieval faking, corpus-unavailability griefing, and Verified Memory leakage. Mode B follows the Mode A spine as a review-gated addition, off the mainnet critical path.

Phase 1: proof-backed token launch. $AIOS launches as a fixed-supply ERC-20 on Ethereum mainnet with the testnet live as evidence: 300,000,000 supply `locked`, no mint function, a 1% transfer tax `locked`, self-funded liquidity, and a minimal site leading with the mechanism. Mass Miner onboarding, including the browser client, starts here, and with no Phase-0 points program there is no conversion or retroactive claim. The token has a working artifact behind it on day one, but proof of mechanism is not proof of demand.

Phase 2: mainnet and bridge. Native L1 genesis, gated on the verification layer surviving adversarial test and on the determinism harness passing for the inference, embedding, retrieval, and adapter kernels. The dual-home bridge opens, the ERC-20 stays live indefinitely, the producer set ships federated until the ordering blockers close, and AIOS:BROWSER, the read-side indexer, and asynchronous delivery arrive.

The gates are not negotiable: security review before any deployment touching the token, consensus, mining penalties, value caps, the verification stack, wallets, keys, or the bridge; user-facing review before any user-facing merge; regression and invariant tests before any release; both determinism harnesses before mainnet; and the cost-anchor and eviction rerun before any allowlist decay.

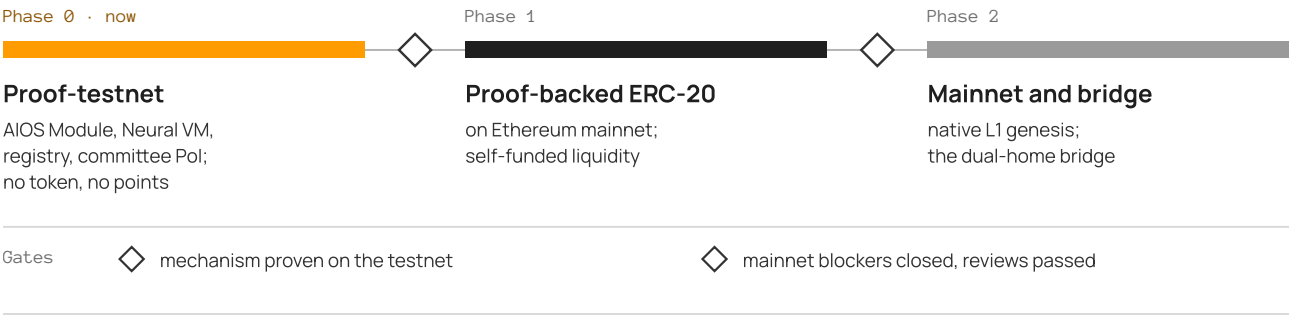
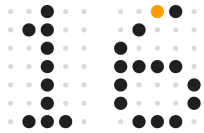


Figure 13. Proof-first sequencing. Each phase opens only when the gate before it closes; the gates are engineering and security benchmarks, not calendar dates.

Figure 13 shows the three phases and the gates between them: the gates, not the dates, are the schedule.



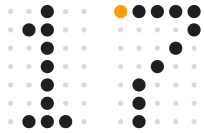
Chapter 16

Limitations and Open Problems

This chapter collects the residual risks in one register; each is stated in full where its mechanism is defined, and none is a qualifier to be minimized.

1. **Proof of Inference is unproven at scale.** No project has shipped verified-inference consensus securing real value at scale. Committee recomputation is the most proven candidate, but its one reported testnet, built by a single developer, published throughput numbers that failed independent verification. Throughput at target model size is an open benchmark, validated before mainnet.
2. **Cross-hardware determinism is the gate** (Section 4.5); until it passes, every path is dispute-then-penalty.
3. **The product path is statistical** (Section 5.6): small corruptions can escape, some large-model fraud is unpunishable, performer-verifier collusion is reduced rather than closed, and deterministic decoding constrains quality.
4. **Frontier models cannot run in consensus, ever** (Sections 4.2 and 10.1). Native Models top out near a useful 2-3B ternary model, with 7B and above unshipped; off-chain models are exactly verifiable only on the pinned Module, which must be built and validated before mainnet; FP16 models verify statistically, never by exact hash; and per-SKU emulation [19] is a watch item. AIOS never presents a frontier chat model as running onchain.
5. **Stakeless mining is weaker than staking** (Chapter 7). The burn prices Sybil-fleet entry, not per-committee participation. Value caps, the floor F above the dishonest payday of the largest reachable job, raised honeypots, and per-entity caps bound the risk without closing it; a decaying allowlist mitigates the bootstrap-compute cliff; the cost-anchor and eviction benchmarks must be re-run at the post-bond cost basis before allowlist decay; and the fallback is bonded mining. Divergence never costs principal, so honest Miners face no mass slashing, and because work credit seats Producers, this governs the ledger itself.
6. **Work-credit farming is ledger-critical** (Section 8.3); its benchmarks are unrun mainnet blockers.
7. **Retention and deterrence are weaker** (Section 7.4): Miners leave when reward falls, and ejection is softer than slashing.
8. **Deterministic retrieval is unproven** (Section 11.4); non-deterministic corpora fall to sampled verification, verified data is never truth, integrity is not relevance, and AIOS is not a data-availability layer.

9. **Mode B moves the trust root to silicon vendors** (Section 9.4): confidentiality without correctness, forgeable quotes under physical access [11], quorum and cap as bounds, no backstop for confidential jobs, and hardware that is a centralized minority, never called decentralized or verified.
10. **Bridge risk is permanent** (Chapter 14): an audited external provider in a separate trust domain, no AIOS-built bridge cryptography, and exposure to producer-set capture.
11. **Launch-time centralization** (Section 15.2): operator-controlled signers make the multi-sig a single point of failure until co-signers join.
12. **Token-first bleed is mitigated, not eliminated.** A testnet is not mainnet, the token trades before native fee revenue, and proof of mechanism is not proof of demand; willingness to pay for verification is the core, unvalidated bet.
13. **Operating runway is tight.** With no treasury, operations run on operator funds and 1% tax revenue, which may shrink or be lowered; native fees and the staking pool fund no operations.
14. **Market limits.** Mode A inputs are visible to the committee (FP16 private inference is deferred; threshold encryption is an open alternative), oracle-free prices depend on repricing, staking adds sell pressure, and the registry space is partly occupied, so the claim is the verification binding, never priority.
15. **Scope.** Verified is provenance of execution, never truth; it does not solve prompt injection; labels are unenforced at launch; adapter composition is approximate. These are the design's tolerances, not disclaimers.



Chapter 17

Conclusion

This paper has specified a Layer 1 in which inference is a first-class operation whose provenance is settled by consensus: a committee seated by work, not capital, recomputes each job on a pinned integer runtime, commits before revealing, and settles on more than two thirds of its work weight. Mining carries no stake, only a burned entry fee, and ordering carries one small equivocation-slashable bond. The same committee proves the corpus and adapter behind an output, and every result states exactly what was proven.

The claims are conditional, and the conditions are named: Assumption D must be validated on real hardware, the stakeless anchor calibrated and benchmarked, and the ordering blockers closed. Nothing is deployed at the time of writing, and Phase 0 exists to prove the mechanism before a token depends on it. On the full-recomputation path the design is deterministic by construction and verified by committee; on every other path it states what it proves and nothing more.

Document history. Version 1.1 (2026-06-21) made the burned registration fee the sole mining-layer anchor. Version 1.2 (2026-07-09) added the Data Internet and Verified Memory. An interim revision (2026-08-23) added the Adapter Registry and the strict conservation invariant. Version 1.3 (2026-09-27) rewrites it as a technical specification with a threat model, propositions, a parameter table, four assurance classes, and the 11-product roster.

Back matter

References

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

[2] L. Luu, J. Teutsch, R. Kulkarni, and P. Saxena. Demystifying incentives in the consensus computer. ACM CCS, 2015.

[3] J. R. Douceur. The Sybil attack. IPTPS, 2002.

[4] S. Ma, H. Wang, et al. The era of 1-bit LLMs: all large language models are in 1.58 bits. arXiv:2402.17764, 2024.

[5] C. Dwork, N. Lynch, and L. Stockmeyer. Consensus in the presence of partial synchrony. *Journal of the ACM* 35(2), 1988.

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

[7] M. Castro and B. Liskov. Practical Byzantine fault tolerance. OSDI, 1999.

[8] S. Micali, M. Rabin, and S. Vadhan. Verifiable random functions. IEEE FOCS, 1999.

[9] C. Cachin, K. Kursawe, and V. Shoup. Random oracles in Constantinople: practical asynchronous Byzantine agreement using cryptography. ACM PODC, 2000.

[10] R. C. Merkle. A digital signature based on a conventional encryption function. Advances in Cryptology (CRYPTO), 1987.

[11] TEE.Fail. Public disclosure of attestation-quote forgery against confidential-computing hardware with physical access, October 2025. Cited as recorded in the AIOS design canon.

[12] P. Maymounkov and D. Mazières. Kademlia: a peer-to-peer information system based on the XOR metric. IPTPS, 2002.

[13] H. He. Defeating nondeterminism in LLM inference. Technical note, 2025.

[14] arXiv:2506.09501, 2025. Numerical nondeterminism in language-model inference across GPU count, GPU type, and batch size. Cited as recorded in the AIOS design canon.

[15] Y. Lin et al. Towards fully 8-bit integer inference for the Transformer model. IJCAI, 2020.

[16] S. Kim, A. Gholami, Z. Yao, M. W. Mahoney, and K. Keutzer. I-BERT: integer-only BERT quantization. ICML, 2021.

[17] B. Jacob et al. Quantization and training of neural networks for efficient integer-arithmetic-only inference. IEEE CVPR, 2018.

[18] Z. Li and Q. Gu. I-ViT: integer-only quantization for efficient vision transformer inference. IEEE/CVF ICCV, 2023.

[19] arXiv:2606.00279, 2026. Bitwise reproducibility of floating-point inference within a GPU generation and its emulation across generations. Cited as recorded in the AIOS design canon.

[20] S. Ma, H. Wang, et al. BitNet b1.58 2B4T technical report, 2025.

[21] E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, and W. Chen. LoRA: low-rank adaptation of large language models. ICLR, 2022.

[22] P. Yadav, D. Tam, L. Choshen, C. Raffel, and M. Bansal. TIES-Merging: resolving interference when merging models. NeurIPS, 2023.

[23] P. Lewis, E. Perez, A. Piktus, et al. Retrieval-augmented generation for knowledge-intensive NLP tasks. NeurIPS, 2020.

[24] S. Robertson and H. Zaragoza. The probabilistic relevance framework: BM25 and beyond. *Foundations and Trends in Information Retrieval* 3(4), 2009.

[25] Y. A. Malkov and D. A. Yashunin. Efficient and robust approximate nearest neighbor search using hierarchical navigable small world graphs. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 42(4), 2020.

AIOS

Correctness is a constraint, not a claim.

AIOS Whitepaper · Version 1.3 · 2026-09-27