Hanzo AI Chain (AIVM)

Useful-Work Mining, the Native AI Coin, and Post-Quantum Omnichain Settlement on Lux

Zach Kelling

Hanzo AI

Working draft: 18 August 2026

Abstract

We present Hanzo Proof of AI (PoAI), the useful-work consensus of the Lux A-Chain and the economic substrate of Hanzo Network. A real AI job is admitted before mining and binds its consumer, model, input, deterministic execution policy, proof policy, reward, and payout destination. A CPU or GPU provider executes that job and commits its integer output, graph transcript, and metering. A-Chain PoAI consensus validates that the bound useful work was actually performed, consumes one global nullifier, and finalizes one mining receipt. Z-Chain batches finalized receipt transitions through P3Q, a Plonky3-derived, pairing-free STARK/FRI settlement rollup. Lux Quasar orders and finalizes the A-Chain and Z-Chain states under the activated post-quantum validator policy. Neither Z-Chain nor another destination may validate raw work or create independent AI supply.

The public execution policy is a versioned deterministic kernel profile rather than a hardware mandate. It fixes operator semantics, tensor layout, accumulator and overflow rules, reduction, fixed-point scaling, rounding, and canonical serialization. Conforming CPU and GPU implementations emit the same canonical transcript binding the job, model, input, graph, output, metering, provider, and destination. Cross-backend parity is a release conformance test for that profile, not another proof system or consensus lane.

The native coin **AI** has a hard maximum nominal supply of two trillion: one trillion allocated permanently to the Hanzo DAO and one trillion reserved for proof-earned miner and validator rewards under a Bitcoin-shaped geometric halving schedule. Existing AI clouds can keep their normal workload revenue and earn additional AI by attaching proof evidence to work they already perform; decentralized operators use the same interface for open jobs. Exact-integer commitments permit ordinary CPUs to validate challenged matrix products more cheaply than GPU execution, while complete protocol security additionally requires graph binding, challenge coverage, transcript availability, verifiable metering, and finality. TEE evidence is optional for confidential workloads, and succinct P3Q settlement authority remains activation- and audit-gated.

Contents

1 Introduction

AI compute is valuable but opaque. A customer pays a provider to run a specific model on a specific input and ordinarily receives only an output and invoice. The provider cannot turn the same execution into portable public evidence, and a third party cannot independently distinguish the claimed workload from a copied answer or fabricated meter.

Synthetic "AI-like" proof of work does not solve this problem. Replacing hash grinding with a chain-derived matrix race demonstrates accelerator expenditure, but the product is not necessarily requested or consumed by anyone. Requiring every full node to replay that matrix work on a GPU also moves consensus toward a specialized hardware requirement; delegating replay to signed GPU attestors introduces a new trust authority.

Hanzo Proof of AI takes the opposite approach. The work is useful first: a real customer or governed network service admits a job and fixes its consumer, deliverable, model, input commitment, deterministic execution policy, proof policy, reward, and payout destination before a miner returns a result. The provider then commits evidence produced alongside that same execution. A-Chain PoAI consensus determines that the bound work was actually done, consumes a global nullifier, and finalizes one mining receipt.

1.1 The cloud-provider proposition

An existing cloud does not need to abandon its scheduler or customer contract. It can attach the PoAI adapter to inference, embedding, evaluation, or supported training work it already runs. The cloud retains normal service revenue and may earn declining AI mining rewards for making that execution independently verifiable. A decentralized provider or idle workstation accepts open jobs through the same protocol.

This is a "compute once, earn twice" market, not free issuance. Proof capture must be materially cheaper than the original workload; the job must be admitted independently of the miner; metering must be verifiable; and the same work authorization must never be rewarded twice.

1.2 CPU-verifiable work

The canonical public path uses deterministic integer execution so honest CPU and GPU backends commit the same bytes. Small jobs may be replayed exactly. For a large committed matrix product, an ordinary CPU can verify a post-commitment Freivalds opening in quadratic work rather than repeating the cubic product. Merkle commitments bind that opening into the model graph.

This arithmetic asymmetry is the core, not the complete protocol. Whole-job accountability additionally requires graph wiring, unpredictable challenges, coverage, transcript data availability, proof finality, and verifiable metering. TEE evidence may be added for confidential jobs. An activated P3Q proof may eventually replace optimistic challenges, but generic proof-system code is not production PoAI authority.

1.3 PoAI consensus and P3Q settlement on the Lux PQ base layer

The three layers answer different questions:

- **PoAI consensus on A-Chain** decides whether useful work was actually performed, whether its evidence is final, and whether it may authorize an AI reward.
- **P3Q settlement on Z-Chain** batches finalized A-Chain receipt transitions into a Plonky3-derived, pairing-free STARK/FRI proof. It compresses and settles A-Chain decisions; it cannot admit raw work or authorize mining.

- **Lux Quasar** orders and finalizes the A-Chain and Z-Chain transitions and supplies the activated post-quantum validator context for exporting their settlement root.

There is one mining authority. "Mine into another chain" means the A-Chain job selects that chain as the payout destination. The destination verifies the Z-settled, PQ-authenticated A-Chain receipt and consumes it exactly once; it cannot accept raw work or run an independent emission schedule.

1.4 The AI economy

AI is the native coin of Hanzo Network. Its maximum nominal supply is two trillion: one trillion permanently allocated to the Hanzo DAO and one trillion reserved for proof-earned miner and validator rewards. The mining pool follows a Bitcoin-shaped geometric schedule: 500 billion in the first era, 250 billion in the second, 125 billion in the third, and so on. Fee burns may make net supply deflationary as issuance declines.

1.5 Contributions and outline

This paper specifies:

1. the A-Chain PoAI useful-work state machine, Z-Chain P3Q settlement, and their relation to Lux post-quantum finality (§2);
2. the two-trillion AI hard cap, fixed DAO allocation, one-trillion proof-earned pool, halving schedule, fees, staking, and burns (§3);
3. the HMM model registry and agent identity surfaces (§4, §6);
4. proof-bearing inference, training, evaluation, and capacity markets, including the existing-cloud adapter (§5);
5. receipt-only omnichain settlement rooted in A-Chain, Z-Chain, and Lux (§7); and
6. confidentiality and security boundaries, with explicit launch gates rather than assumed deployment (§8, §9, §10).

2 Architecture

2.1 AI Chain is the Lux A-Chain

Hanzo AI Chain is the public identity of the Lux A-Chain when it operates the open AI protocol. There is not a Hanzo proof chain beside A-Chain. A single canonical state machine owns job admission, proof policy, challenge state, work nullifiers, reward authorization, and receipt roots.

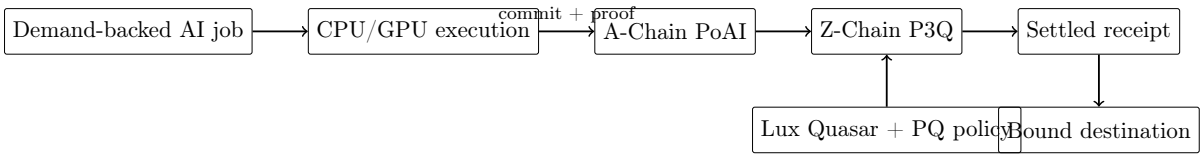

Figure 1: PoAI validates useful work on A-Chain; P3Q settles it on Z-Chain; Lux supplies PQ-rooted finality; destinations consume receipts rather than raw work.

2.2 PoAI useful-work state machine

Each job progresses through

$$\begin{aligned}\textsc{Requested} \to \textsc{Committed} \to \textsc{Verifying/Challengeable} \\ \to \textsc{Finalized} \mid \textsc{Rejected} \mid \textsc{Expired}.\end{aligned} \tag{1}$$

A-Chain fixes the execution and proof policy before commitment. It derives any random challenge afterward, resolves the policy to final or rejected, checks metering and reward allowance, consumes the global nullifier, and creates the destination-bound receipt atomically. An optimistic result cannot become issuance-final while its challenge window remains open.

2.3 Z-Chain settlement and Lux base-layer finality

Z-Chain is the P3Q proof-compression and settlement rollup for finalized A-Chain receipt transitions. Its intended proof system is a Plonky3-derived, pairing-free STARK/FRI stack. Z-Chain does not admit jobs, determine usefulness, own the mining nullifier, or authorize subsidy; its proof must bind exactly to the finalized A-to-Z receipt transition.

Lux Quasar orders and finalizes A-Chain and Z-Chain blocks. Exported roots must be authenticated under the validator and post-quantum policy that is actually activated. The security statement therefore names its assumptions: validator set, threshold, key lifecycle, downgrade prevention, and relay verification. Selecting ML-DSA in a registry does not by itself produce a threshold finality certificate.

Pulsar and Corona explore threshold lattice signatures for consensus and custody. P3Q is the intended Z-Chain settlement substrate, but the exact PoAI AIR, recursion policy, state-transition binding, and verifier remain subject to protocol, audit, integration, and activation gates. This paper does not treat a repository or precompile address as deployment evidence.

2.4 Deterministic execution substrate

The public proof-bearing path admits a bounded deterministic graph whose operations define:

- integer weights and activations, normally `int8`;
- wider accumulators with explicit overflow or saturation behavior;
- canonical tensor layout, serialization, reduction, and rounding;
- deterministic routing, normalization, sampling, and tie-breaking;
- content-addressed model, input, output, runtime, and graph roots.

Execution may occur on a CPU, GPU, NPU, or other backend that produces the same committed bytes. Accelerators improve throughput; they are not a consensus requirement for every validator.

2.5 Execution profile and canonical transcript

Determinism is fixed by a versioned *kernel profile*, not by naming a particular device. The profile commits the admitted operators, tensor layout, integer widths, accumulator bounds, overflow behavior, reduction order, fixed-point scales, rounding, routing tie-breaks, and byte encoding. A cloud provider may run optimized CPU, GPU, or accelerator kernels, but those kernels are PoAI-compatible only when they implement the same profile exactly.

Every execution emits one *canonical transcript*. Its header binds the A-Chain job and policy epoch, kernel-profile version, model, runtime, input, declared graph, output, metering, provider,

destination, and nonce. Its ordered body binds every typed operation to its input and output commitments. The root of this content-addressed serialization is the object challenged by A-Chain and, after PoAI finality, settled by P3Q on Z-Chain. The kernels define the function; the transcript identifies and commits the particular paid invocation.

CPU/GPU parity is a conformance gate for a profile release, not a mining primitive. Golden vectors must cover kernel outputs, operation leaves, transcript roots, challenge derivation, and openings. A backend that diverges is not admitted; consensus does not introduce floating-point tolerance to accommodate it.

2.6 Evidence backends

The job selects one activated evidence profile:

Profile	Role
Exact replay	Canonical CPU or verifier-set recomputation for small jobs.
Optimistic integer	Transcript commitment, data availability, watcher re-execution, and objective challenge openings.
Succinct P3Q	Plonky3-derived Z-Chain settlement only after the exact PoAI AIR, A-to-Z binding, recursion policy, and verifier are audited and activated.
Confidential compute	Optional vendor attestation for private prompts, weights, or outputs, combined with explicit TCB and revocation policy.
Subjective quorum	Attributable judgment that may earn demand fees but cannot alone authorize mining subsidy.

Table 1: Evidence profiles are selected by job policy; none creates another mining authority outside A-Chain.

2.7 Canonical protocol objects

The minimum consensus objects are `MiningJob`, `ExecutionCommitment`, `Challenge`, `Proof Finality`, `WorkNullifier`, and `MiningReceipt`. Public APIs are submission and query surfaces for these objects. An in-memory task server, GPU scheduler, or destination smart contract is not the canonical mining state machine.

2.8 Implementation status

The exact-integer arithmetic core, Go/Rust vectors, Solidity witness fixtures, and A-Chain receipt components exist as implementation evidence. The production system still requires a complete Zen model transcript, data-availability path, finalized-not-pending proof gate, verifiable metering, token-policy upgrade, PQ root export, Z-Chain settlement, and exactly-once destination consumption. Mining remains disabled until the launch gates in §10 pass.

3 The AI Native Coin

3.1 Token vs. coin

LUX is the native coin of the Lux primary network and pays for base-layer security and gas. AI is the native coin of Hanzo Network and rewards useful AI work finalized through A-Chain PoAI consensus. They are independent assets; neither is a wrapper of the other. Lux supplies the post-quantum root of finality, while AI supplies the demand, staking, and reward economy for the open compute network.

3.2 Hard-cap allocation

Parameter	Value
Maximum nominal supply	2,000,000,000,000 AI
Permanent Hanzo DAO allocation	1,000,000,000,000 AI
PoAI miner and validator pool	1,000,000,000,000 AI
Destination-chain inflation	0 AI
Decimals	18

Table 2: AI has one network-wide supply. Destination representations conserve against canonical settlement and never add a local allocation.

The DAO allocation is fixed at genesis and cannot be reclassified into mining emission. It remains under the Hanzo DAO's constitutional treasury controls for research, public infrastructure, ecosystem development, liquidity, and long-run network stewardship. The proof-earned pool has no discretionary mint path: each unit must fit the active era allowance and a finalized A-Chain mining receipt.

3.3 Bitcoin-shaped proof-earned emission

Let $M = 10^{12}$ AI be the proof-earned pool, h_0 the activation height, and H the governance-fixed halving interval in finalized A-Chain blocks (nominally four years). For era $k = \lfloor (h - h_0)/H \rfloor$ and fractional progress $f \in [0, 1)$, cumulative mining allowance is

$$A(h) = M(1 - 2^{-k}) + fM2^{-(k+1)}. \tag{2}$$

The first era may emit at most 500 billion AI, the second 250 billion, the third 125 billion, and so on. Since $\sum_{k \geq 0} M2^{-(k+1)} = M$, miner and validator rewards converge to exactly one trillion AI and total nominal issuance cannot exceed two trillion AI.

The canonical settlement state tracks `mintedMining` and clamps every receipt to $A(h) -$ `mintedMining`. A destination never evaluates the schedule; it consumes the amount already authorized by A-Chain. The precise within-era split between miners and validators, and the work-normalization function across job classes, must be fixed and tested before activation.

3.4 What earns AI

AI is earned for proof-bearing useful work, not for owning hardware or self-reporting time. A subsidy-eligible receipt must bind:

- a pre-admitted requester or governed network-service job;
- a model, input commitment, deliverable, and consumer;
- a deterministic execution and proof policy;
- a committed output, transcript, and verifiable metering root;
- a post-commitment challenge or activated succinct proof;
- a unique A-Chain work nullifier and payout destination.

A raw GPU-hour, miner-created circular job, copied output, TEE quote, customer signature, or destination-local proof is insufficient. This separation is what lets the protocol reward actual work without turning fake invoices into a mint.

3.5 Compute once, earn twice

PoAI is an incremental revenue rail for existing AI clouds. A provider may keep the ordinary customer fee for an inference or training workload and also earn a declining AI reward by capturing proof evidence during that same execution. A decentralized provider uses the identical interface to accept open jobs when capacity would otherwise be idle.

The economic claim depends on verification asymmetry. For a supported matrix operation the provider performs cubic work, while an ordinary CPU verifies a post-commitment Freivalds opening in quadratic work plus commitment overhead. Small jobs may be replayed exactly. Large jobs require graph coverage and data availability under the optimistic policy, or an activated succinct proof. Proof capture must remain materially cheaper than rerunning the workload.

3.6 Fees, burns, and net deflation

Demand fees and mining subsidy are separate. The requester pays for useful service; the declining mining allowance bootstraps verifiable supply. A governed fraction β of settled protocol fees is burned:

$$\Delta S = \text{PoAI issuance} - \beta\,\text{settled fees}. \tag{3}$$

As geometric issuance tends toward zero, persistent fee burn can make net supply deflationary. Burns do not replenish either the DAO allocation or the mining pool.

3.7 Staking and slashing

Validators stake under the active Lux/A-Chain policy to order and finalize PoAI state. Compute providers bond against objective fraud, transcript withholding, metering fraud, replay, and policy violations. TEE collateral and measurement rules apply only when a confidential job selects that evidence profile. Stake requirements and reward weights are launch parameters, not immutable marketing figures, and require measured economics and governance activation.

4 Hanzo Models (HMMs)

4.1 What Lives On Chain

A Hanzo Model is a content-addressed, versioned bundle of weights, metadata, and license. The chain does *not* store weight blobs; those live in content-addressed storage (Hanzo object storage, `hanzo/storage`, or any compatible CAS) and are referenced by their root hash. The chain stores a small fixed-size record per model version that commits to everything a caller, an operator, or an auditor needs to verify a job:

4.2 Versioning and Provenance

Each new version of a model is a new record with a fresh `model_root` and a `training_history` edge pointing to the parent. This produces a directed acyclic graph rooted at the genesis of each model family. Anyone can walk the graph to see the full provenance: the original pretraining run, every fine-tune, every distillation, and the dataset commitment used at each step.

4.3 License Enforcement

The `license_root` commits to a license document with three machine-readable fields: *usage* (commercial / research / open), *redistribution* (allowed / restricted / prohibited), and *derivative* (allowed / restricted / prohibited). Compute operators are required to read and respect the

Field	Definition
`model_root`	Merkle root over the weight tensors at version V. Content-addressed; pinning the root pins the weights bit-exactly.
`benchmark_root`	Merkle root over a canonical evaluation harness output (e.g. MMLU, HELM, internal evals). Submitted by the model owner; challengeable on chain.
`license_root`	Hash of the license document (commercial, research, or open under explicit terms), with a machine-readable summary.
`inference_price`	A price function $p(\cdot)$ in AI per output token (or per call, per byte for non-text models). May be a constant, a piecewise function, or a reference to an on-chain pricing oracle.
`training_history`	Linked list of parent model_roots, each edge annotated with the training dataset commitment, the trainer identity, and the block height at which training completed.
`owner`	Address authorized to publish new versions and to update license / price; may be a multisig or a DAO.

Table 3: HMM registry record. Six fields per model version. The chain holds the commitments; weights and bench artifacts live in CAS.

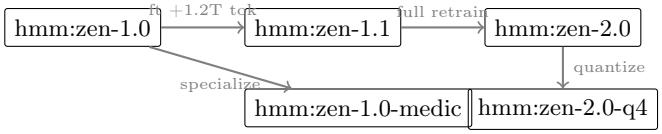

Figure 2: Provenance DAG. Each edge is a chain transaction recording the parent model_root, the training dataset commitment, and the trainer identity.

license at job time; the `INFER_REQUEST` precompile (§2) refuses to schedule a job if the caller has not posted a matching license attestation.

4.4 Price Function

`inference_price` is a first-class on-chain object. The simplest form is a constant `uint256` in AI per output token. More expressive forms are supported:

- *Piecewise tiered*: a price ladder that lowers per-token cost above batch thresholds.
- *Time-of-day*: a function over block timestamp, used to clear demand at peak.
- *Oracle-backed*: a reference to an on-chain price oracle whose own update mechanism is visible on chain.

The model owner sets the function and may update it; updates are rate-limited (one update per epoch by default) to prevent surprise price moves between job submission and execution.

4.5 Benchmark Commitments and Challenges

A model owner publishes a `benchmark_root` when registering a version. Any address may post a *challenge*: a deposit-bonded claim that the published benchmark output does not match the actual output of the published `model_root` on the canonical eval harness. Challenges are resolved under an activated A-Chain proof policy: deterministic replay or an optimistic integer transcript for public jobs, with optional TEE evidence when the evaluation is confidential. If the challenge succeeds, the challenger collects the model owner's benchmark bond; if it fails, the challenger forfeits theirs. This matches the dispute pattern used elsewhere in Lux for attestation challenges (LP-134 §6).

4.6 Candidate genesis model registry

The proposed genesis state may seed the following Hanzo model families after their exact roots, licenses, execution policies, and proof profiles are frozen:

- `hmm:zen-text` (text-to-text foundation model);
- `hmm:zen-vision` (multimodal vision-language);
- `hmm:zen-audio` (speech and audio);
- `hmm:zen-code` (code-specialized);
- `hmm:zen-embed` (embedding model).

Registry activation is separate from ordinary gateway availability. A model is PoAI-mineable only after its deterministic execution and proof policy pass the deployment gates in §10.

5 Useful-Compute Markets

Hanzo Network exposes inference, training, evaluation, and capacity markets through one A-Chain job format. A market determines who requests the work and who pays for it. PoAI consensus determines whether the bound execution was actually completed and whether a mining reward may be finalized. These are separate decisions.

5.1 Canonical mining job

Before execution, A-Chain commits a job containing

$$J = H(\text{job id}, \text{requester}, \text{consumer}, \text{model root}, \text{input root}, \\ \text{execution policy}, \text{proof policy}, \text{reward budget}, \text{expiry}, \text{destination}, \text{recipient}). \tag{4}$$

The job must be admitted before a miner commits its result. This prevents the miner from choosing an arbitrary completed computation and declaring it useful after the fact. "Useful" means useful under an explicit demand or governed network-service policy; it does not claim universal truth or social value.

5.2 Execution and commitment

A CPU or GPU provider runs the specified integer program and commits its output, typed graph transcript, metering, provider identity, and runtime identifier. The canonical path defines tensor layout, serialization, accumulator width, overflow, rounding, saturation, routing, and

tie-breaking. An honest CPU and GPU must therefore produce identical committed integer values.

The commitment precedes any random challenge. A job-specific nonce, policy epoch, and execution commitment form the global work nullifier, so the same work authorization cannot be replayed into a second chain or reward.

5.3 CPU validation

PoAI does not require every validator to repeat a GPU-scale workload. The active proof policy chooses one of four paths:

- **Exact replay**: appropriate for small jobs; an ordinary CPU executes the canonical integer program.
- **Optimistic integer proof**: the provider publishes a Merkle-rooted transcript; watchers re-execute and may challenge any invalid operation. For an opened matrix product $C = AB$, a CPU checks $A(Br) = Cr$ over $\mathbb{F}_{2^{61}-1}$ after r is derived from a post-commitment beacon. The check is quadratic rather than cubic.
- **Succinct proof**: an audit- and activation-gated P3Q circuit may replace the challenge game. A generic STARK/FRI library is not, by itself, a production PoAI verifier.
- **Confidential execution**: an approved TEE profile may add vendor provenance for private prompts, weights, or outputs. It is an optional trust policy and not the definition of PoAI.

Freivalds supplies local matrix-product soundness, not automatic whole-model soundness. The complete policy must also enforce graph wiring, challenge coverage, transcript availability, metering integrity, and a finality delay that prevents minting while a challenge remains open.

5.4 Inference market

A requester posts a model, input commitment, output contract, latency bound, and maximum demand fee. Providers bid or accept the open price. On successful PoAI finality, the consumer receives the output, the provider receives its demand fee, and the receipt may receive a mining reward under the current era budget. The mining reward is not a substitute for the customer's payment; it rewards making the execution independently accountable.

5.5 Training and evaluation markets

A training job additionally commits the parent model, dataset root, optimizer state, loss, deterministic randomness, checkpoint cadence, and target metric. Only training graphs expressible in the activated deterministic operation set are proof-bearing. Floating-point distributed training is not silently treated as consensus deterministic.

An evaluation job commits the model, dataset or sealed test root, metric, and aggregation policy. A subjective judge quorum may earn a demand fee for a signed assessment, but a quorum signature alone cannot authorize mining subsidy.

5.6 Capacity market

The capacity market routes open jobs to advertised CPU/GPU inventory. Hardware class, memory, region, price, and availability affect matching, but raw rented seconds do not mint AI. A provider earns mining rewards only for an admitted job whose proof and metering reach PoAI finality.

5.7 Existing cloud-provider adapter

The same job format is intended for hyperscalers and independent operators. A cloud adapter captures the model/input commitment and execution policy when the customer submits an ordinary workload, records the integer transcript alongside normal execution, and sends the proof commitment to A-Chain. The provider keeps its existing invoice revenue and may receive AI when PoAI consensus finalizes the work. Customer opt-in, confidentiality policy, and export controls remain part of job admission.

This gives existing clouds a direct reason to participate: new revenue with no need to abandon their scheduler, a portable cryptographic service receipt for customers, and access to decentralized overflow capacity. The network gains real workload volume instead of synthetic benchmark puzzles.

5.8 Anti-farming rules

The reward policy must reject circular or unverifiable demand. At minimum:

- related requester/provider control is declared and may be capped or excluded;
- demand fees and network-service budgets are bounded and non-recyclable under the active policy;
- metering is committed and objectively challengeable;
- output copying without the bound transcript fails the challenge;
- every finalized receipt consumes one global A-Chain nullifier;
- aggregate payouts never exceed the current halving-era allowance.

5.9 Spot and reserved service

Providers may quote spot or reserved prices. Spot jobs trade cancellation risk for lower price; reserved jobs escrow a forward capacity commitment. Both use the same A-Chain job, proof, nullifier, receipt, and settlement path. Pricing is a market concern; proof validity is not.

6 Agent Registry

6.1 Agents as First-Class Citizens

A Hanzo Agent (`~/work/hanzo/agents`) is an autonomous program that calls models, holds state, and takes actions on behalf of a principal. Without an on-chain registry, agents are indistinguishable from any other API caller and cannot accumulate durable identity, capability, or reputation. The AI Chain promotes agents to first-class citizens with three primitives: *persistent identity*, *capability declaration*, and *on-chain reputation*.

6.2 Persistent Identity

An agent is created on chain with a record:

The agent's identity is its `agent_id`. It does not require an externally owned account in the EVM sense; it is referenced as a chain object addressable from any contract through the `AGENT_AUTH` precompile (§2).

Field	Definition
`agent_id`	Hash of (creator address, nonce). Stable for the agent's lifetime.
`controller`	Address authorized to update the agent record (or rotate keys, retire the agent).
`principal`	Address on whose behalf the agent acts. May equal controller.
`model_set`	Set of allowed HMM `model_root`s the agent may invoke.
`capability_root`	Merkle root of the agent's capability declaration (see below).
`spend_cap`	Per-epoch ceiling on AI the agent may spend on compute.
`reputation`	A monotonically updated score in $[0, 1]$ (§6.4).

Table 4: Agent registry record.

6.3 Capability Declaration

A capability declaration is a typed list of actions the agent is authorized to take. Declarations are content-addressed (committed to via `capability_root`) and machine-readable. Examples:

- `infer:hmm:zen-text@*`: invoke any version of zen-text;
- `spend:ai:max=10`: may spend up to 10 AI per epoch;
- `cross:f-chain:fhe-infer:hmm:medic-7b`: may issue confidential inference calls to F-Chain on the named model;
- `settle:c-chain:erc20:USDC@<addr>`: may transfer USDC on the EVM C-Chain.

The declaration is the contract between the agent's principal and the chain; precompiles enforce it at every action. An agent attempting to exceed its declaration is rejected at the precompile boundary, before any external side effects.

6.4 Reputation

Reputation is a function of the agent's compute and settlement history:

$$\text{rep}(a) = \sigma(w_1 \cdot \text{success\_rate}(a) + w_2 \cdot \log(1 + \text{volume}(a)) + w_3 \cdot \text{age}(a) - w_4 \cdot \text{disputes}(a)) \tag{5}$$

with the logistic σ mapping to $[0, 1]$. Weights w_i are governance parameters. `success_rate` is the fraction of jobs that produced finalized A-Chain receipts under the selected proof policy; `volume` is total AI spent; `age` is epochs since registration; `disputes` is the count of dispute resolutions against the agent.

Reputation is read by markets and by other agents: an inference operator may price differently for high- vs. low-reputation callers; an agent network may decline to delegate work to a low-reputation peer. The chain publishes reputation; consumers act on it.

6.5 Cross-Chain Authorization

Agents transact across the Lux multi-chain layout (§7) through capability-bound precompiles. To call the EVM C-Chain on the agent's behalf, a contract invokes `AGENT_AUTH` with (agent_id,

target_chain, action). The precompile verifies that the agent's capability declaration permits the action, decrements `spend_cap`, and emits a B-Chain bridge message. The cross-chain side then sees a single canonical authorization rooted at the agent record on AI Chain.

6.6 Privacy of Agent Memory

The agent registry does not record agent memory or working state on chain; only commitments. Agent memory lives in the agent's private storage (capsule, in Hanzo Cloud parlance), encrypted under keys custodied by M-Chain MPC if the agent so configures. The chain knows which agent acted; it does not know what the agent thought.

6.7 Worked Example: A Research Agent

A user creates a research agent with the following capability declaration:

```
infer:hmm:zen-text@*
infer:hmm:zen-vision@*
spend:ai:max=50
settle:c-chain:none
cross:f-chain:none
```

The agent is registered with `spend_cap` 50 AI/epoch and `model_set` = {zen-text, zen-vision}. Over the next epoch the agent runs 1,200 inference calls on zen-text and 240 calls on zen-vision, spending 38.7 AI. Each call produces a PoAI-finalized receipt referenced by the agent's record. After 30 successful epochs and zero disputes, the agent's reputation has saturated near 1.0 and downstream markets begin offering it preferred pricing on reserved capacity.

7 Post-Quantum Omnichain Settlement

Hanzo's omnichain claim is deliberately asymmetric. Proof-of-AI consensus exists only on Lux A-Chain. Any supported chain may receive a payout, but no destination may accept raw work, run a local mining verifier, maintain a competing work-spent set, or instantiate another AI emission schedule. Z-Chain is the P3Q settlement rollup for finalized A-Chain transitions, not another PoAI authority.

7.1 A-Chain work truth, Z-Chain settlement, and Lux finality

The settlement path is:

1. A-Chain admits a useful job and fixes the execution policy, proof policy, reward, destination chain, and destination recipient.
2. A provider commits its output, transcript, and metering after executing the job on CPU or GPU.
3. PoAI consensus resolves the configured proof policy, validates that the bound work was actually performed, and consumes the global nullifier.
4. The final mining receipt is committed under the A-Chain receipt root.
5. Z-Chain batches the finalized receipt transition and verifies its activated Plonky3-derived P3Q STARK/FRI settlement proof. It does not reconsider usefulness or accept an unrelated raw-work claim.
6. Lux Quasar finalizes the corresponding A-Chain and Z-Chain blocks; the activated post-quantum validator policy authenticates their roots, heights, and validator context.

7. Warp/Teleport, a native atomic boundary, or an authenticated relay carries the Z-settled receipt and inclusion proof to the named destination.

8. The destination verifies finality, Z settlement, inclusion, chain binding, recipient, amount, and local non-consumption, then releases the representation once.

Thus "mine into chain D" means that the A-Chain job names D before work is committed. It never means that D can decide whether the work occurred.

7.2 Receipt object

A mining receipt binds at least

$$\begin{aligned} R = (&\text{job id}, \text{work nullifier}, \text{miner}, \text{model}, \text{input}, \text{output}, \\ &\text{execution policy}, \text{proof policy}, \text{proof finality}, \text{measured work}, \\ &\text{authorized reward}, \text{destination chain}, \text{destination recipient}, \text{A-Chain height}). \end{aligned} \tag{6}$$

The destination acts only on a completed, nonzero, finalized receipt under an authenticated monotonically advancing Z-settlement root that binds the A-Chain receipt. A raw miner signature, Freivalds opening, TEE quote, or output quorum is not an omnichain authorization.

7.3 Z-Chain P3Q settlement rollup

P3Q is the intended Plonky3-derived, pairing-free STARK/FRI substrate for compressing finalized A-Chain receipt transitions on Z-Chain. The statement proved must include the prior A-Chain receipt root, finalized transition batch, global-nullifier effects, reward authorizations, and resulting root. Before the exact PoAI AIR, recursive batching policy, A-to-Z binding, verifier, and node integration are audited and activated, Z-Chain may record only an independently authenticated final A-Chain root and must not claim succinct proof authority.

7.4 Supply conservation

There is one canonical AI ledger and one proof-earned allowance. A destination either releases escrowed canonical AI or mints a representation backed by the canonical settlement rail. It does not reproduce the two-trillion cap, DAO allocation, halving calculation, or global nullifier. The sum of canonical AI and unbacked representations must never exceed the canonical supply.

7.5 C-Chain and external EVMs

An EVM receipt adapter verifies the Z-settled, PQ-authenticated A-Chain receipt, Merkle inclusion, destination binding, and exactly-once consumption. Application contracts may use the receipt to pay a provider, release a model, advance a workflow, or record provenance. They cannot reinterpret the evidence or award a different mining amount.

7.6 F-Chain confidentiality

F-Chain may supply encrypted execution or confidentiality services to an A-Chain job. Its ciphertext commitments and output root are bound into the same PoAI receipt. F-Chain does not become a mining authority, and a confidential execution profile still requires an explicit proof or attestation policy.

7.7 M-Chain threshold custody

M-Chain may custody model keys or authorize confidential-weight release under a threshold policy. That authorization is an input to the A-Chain job, not proof that the job ran. Pulsar or Corona may authenticate consensus and custody flows after their selected protocol and deployment pass review; choosing ML-DSA alone does not create threshold security.

7.8 B-Chain and Warp/Teleport transport

B-Chain, Warp/Teleport, and native atomic interfaces are transport and settlement mechanisms. Their security obligation is to carry the exact Quasar-finalized A/Z settlement root and receipt proof without changing the destination or amount. The same receipt may be relayed by anyone but consumed only once at its bound destination.

7.9 Topology summary

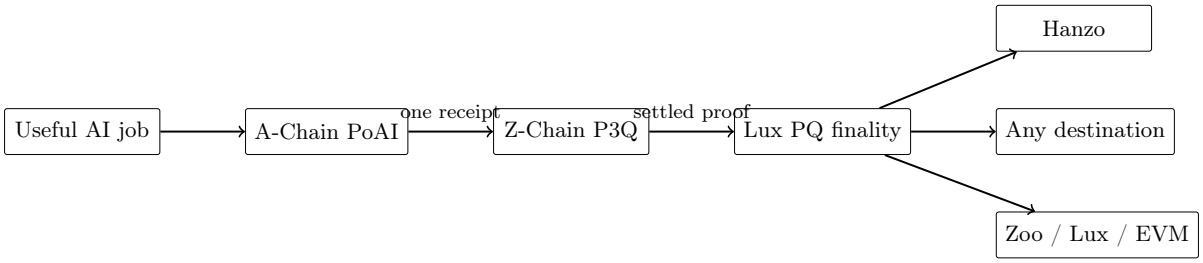

Figure 3: One A-Chain PoAI decision is proved and settled through P3Q on Z-Chain, then becomes a PQ-authenticated omnichain receipt. Destinations consume it; they do not validate mining work.

8 Privacy

8.1 Threat Model

The AI Chain assumes three independent adversaries:

- A *network-level* adversary that observes traffic between callers, operators, and the chain.
- A *provider-level* adversary that operates GPUs and runs inference on caller data.
- A *chain-level* adversary that observes everything written on chain.

Plaintext on chain is public to all three. The privacy primitives below move sensitive material outside the chain entirely or hide it under encryption that no level of the adversary can decrypt without keys.

8.2 Public Inference

The default mode is public inference. The caller sends plaintext input; the operator returns plaintext output; the receipt commits to both. This mode is appropriate for non-sensitive workloads (general chat, public data analysis, code completion on open-source code). The chain records input_root and output_root commitments — Merkle roots over the plaintext — but the underlying bytes live in CAS and are not stored on chain.

8.3 Confidential Inference (F-Chain Delegation)

For sensitive workloads — medical, financial, defense, and any case where the input itself is the sensitive asset — the AI Chain delegates to F-Chain (§7) under an activated confidentiality profile. That profile may use an approved homomorphic-encryption scheme such as CKKS, confidential hardware, threshold decryption, or a governed composition. Its exact parameters, supported operations, leakage model, and trust assumptions must be fixed before the job is admitted. A TEE is optional rather than implied by homomorphic execution, and no profile may claim that an operator cannot see plaintext unless the activated construction actually provides that property.

The F-Chain inference path inherits the LP-067 confidential ERC-20 pattern in spirit: the on-chain object commits to a value and an authority but does not reveal the value. Here the object is a model query and an output, not a token amount. When a TEE profile is selected, its quote binds the input ciphertext root, model commitment, output ciphertext root, job, runtime measurement, and freshness to the declared hardware identity. A non-TEE profile must supply the proof and key-management bindings specified by its own policy.

8.4 Threshold Custody for Model Weights

Model owners may custody decryption keys for their weights under M-Chain threshold MPC (§7). This addresses two scenarios:

- *Selective release*: the model's weights are encrypted at rest; an operator that wants to load the model must obtain a threshold-signed decryption shard. Custody quorums can be drawn to satisfy contractual or jurisdictional requirements.
- *Revocable inference*: a model that has been served for a time can be retired by retiring the custody key — operators cannot reload the weights once the custody quorum withholds the shard. Useful for time-bounded research access or for deprecating compromised checkpoints.

8.5 Receipt Privacy

Even with FHE-protected inputs and outputs, the existence of a receipt is observable on chain. The chain records (caller, operator, model_root, block, payment). For workloads where even this metadata is sensitive, callers may use a stealth-address pattern: the caller is a one-shot address, the payment is denominated in a confidential AI balance on F-Chain, and the receipt is encrypted to the caller's session key. The on-chain footprint becomes a generic "some address paid some operator for some model."

8.6 Privacy Layering Summary

Mode	Input	Output	Identity
Public	plain	plain	visible
Confidential (F)	FHE	FHE	visible
Custody (M)	plain	plain	visible
Custody + Confidential	FHE	FHE	visible
Confidential + Stealth	FHE	FHE	hidden

Table 5: Privacy modes available on AI Chain. Each row composes independent primitives; the chain validates the receipt regardless of which combination the caller chose.

8.7 What Is Not Hidden

The chain does not hide block height, the validator set, the existence of a receipt, the finality proof, or the operator's stake state. These are network invariants and must be public for the consensus itself to function. Anything tied to the workload (input, output, model weights when custodied, optionally the caller identity) can be hidden behind the primitives above.

9 Security

9.1 Three complementary protocol layers

A-Chain PoAI consensus decides whether an admitted useful-work claim is valid, final, unique, and reward-bearing. Z-Chain P3Q compresses and settles finalized A-Chain receipt transitions without becoming a mining authority. Lux Quasar supplies validator ordering, fork choice, and finality for both states. The activated post-quantum validator authentication policy secures exported A/Z settlement roots. These are complementary layers: PoAI is not an alternate block-leader race, P3Q is not raw-work admission, and Quasar finality alone does not prove that an AI workload ran.

Security claims must be made against the protocol actually activated. ML-DSA is a standardized signature, not evidence of computation and not a threshold protocol by itself. P3Q is the intended Plonky3-derived, pairing-free Z-Chain settlement substrate, but a PoAI AIR, recursive policy, A-to-Z state binding, and verifier are not authoritative until implementation, audit, parameter review, node wiring, and activation are complete. Pulsar and Corona likewise require their selected threshold protocol and networking, not only compatible signature bytes.

9.2 Execution soundness

For a committed integer matrix product $C = AB$ over $\mathbb{F}_p$, $p = 2^{61} - 1$, a post-commitment Freivalds challenge accepts an invalid product with probability at most $1/p$ per independent vector. Honest integer work has zero false rejection when serialization, overflow, reduction, and challenge derivation are identical.

This local bound does not automatically cover a full graph. If s operations are sampled uniformly from a graph of L operations with b invalid nodes, a simple upper bound on acceptance is

$$\Pr[\text{accept invalid trace}] \leq (1 - b/L)^s + s/p^k, \tag{7}$$

where k is the number of independent challenge vectors per opening. Sparse fraud is dominated by the coverage term. A production optimistic policy therefore needs complete transcript availability and independent watchers that may challenge any invalid operation, or a coverage parameter justified against the reward at risk.

9.3 CPU validation claim

For supported operations, CPU validation is practical because an opened matrix product is checked in quadratic work instead of rerunning the provider's cubic product. Merkle inclusion and graph binding ensure the opened matrices are part of the committed job. Small jobs may be replayed exactly. This does not imply that every arbitrary floating-point training system is CPU-verifiable; only activated deterministic operations with bounded payloads fall under the claim.

9.4 Availability and metering

An optimistic proof is ineffective when the provider withholds the transcript. The complete transcript, or erasure-coded data sufficient for every challenge, must remain available through the finality window. Issuance remains reversible or disabled until that window closes.

Correct output does not prove the claimed work quantity. Reward calculation must bind a metering commitment that is derived from the admitted graph and execution policy. Wall-clock time, GPU SKU, token count, and FLOP claims cannot be accepted solely from the provider.

9.5 Usefulness and anti-self-dealing

Cryptography proves execution of the specified job; protocol admission proves that the work had an identified consumer or governed network purpose before the result existed. Neither proves universal truth or social value. A miner-created circular job, recycled demand fee, copied output, or colluding requester must not turn the mining pool into an unbounded faucet. Admission caps, declared related parties, non-recyclable budgets, work-class normalization, and per-era reward limits are consensus-critical economics.

9.6 TEE scope

TEE evidence is optional and explicitly vendor-dependent. A confidential profile must validate the complete certificate chain, TCB collateral, measurement, freshness, revocation, model, input, output, runtime, job, and provider binding. A quote proves provenance under that vendor trust model; it does not prove semantic correctness and does not replace proof-policy finality.

9.7 Replay and omnichain safety

The global work nullifier includes the A-Chain network, policy epoch, job, and execution commitment. A-Chain consumes it atomically with receipt finalization. The receipt binds one destination chain and recipient. Each destination records consumption of that exact receipt, but its local consumed set does not decide whether work was valid.

An accepted destination payout must require a monotonically advancing, PQ-authenticated, Quasar-finalized Z-Chain settlement commitment to the A-Chain receipt root (or an explicitly activated native A/Z direct-finality proof) and Merkle inclusion of the full receipt. Accepting a raw proof, TEE quote, miner signature, or local subsidy calculation would create a second mining authority and violate the supply cap.

9.8 Launch security gates

Mining remains disabled until the production system demonstrates:

- byte-identical CPU/GPU commitments for a real model path;
- objective detection of a fabricated intermediate operation;
- transcript recovery and challenges under node loss;
- no irreversible issuance while proof status is pending;
- verifiable metering and anti-self-dealing admission;
- atomic global-nullifier consumption and reward finalization;
- Z-settled, PQ-authenticated receipt export and exactly-once consumption in at least two destination environments;

- enforcement of the two-trillion total cap and one-trillion mining pool under replay, reorganization, and bridge failure.

10 Deployment and Activation

This document is a working protocol and implementation report. Historical calendar dates and benchmark targets are not activation evidence. PoAI mining, the two-trillion AI monetary policy, and omnichain payout remain fail-closed until governance fixes their parameters and the gates below pass.

10.1 Phase 0: arithmetic and wire freeze

- Freeze field, serialization, domain tags, accumulator and overflow rules, graph operation set, and challenge derivation.
- Publish byte-identical CPU/GPU vectors for every activated operation.
- Demonstrate a production Zen model emitting the complete integer graph and transcript on its normal execution path.
- Freeze the A-Chain job, commitment, nullifier, receipt, and root formats, plus the A-to-Z settlement statement.

10.2 Phase 1: exact replay and shadow proofs

Small jobs may use canonical exact replay. Larger jobs produce optimistic or succinct shadow evidence without authorizing subsidy. The network measures proof-capture overhead, CPU validation cost, transcript size, data availability, challenge latency, and cross-backend divergence under production load. Shadow evidence has no mint authority. This stage is deployment safety and measurement, not a weaker proof class accepted by consensus.

10.3 Phase 2: optimistic PoAI activation

Optimistic proof authority may activate only after:

- honest CPU/GPU commitments agree byte-for-byte;
- fabricated operations are objectively caught and honest work is never falsely rejected;
- independent watchers can reconstruct and challenge all committed data;
- no receipt authorizes irreversible issuance before challenge finality;
- metering and anti-self-dealing rules withstand adversarial tests;
- one global nullifier and one halving allowance remain correct across reorganization and failure;
- an authenticated A-Chain receipt is checkpointed through the fail-closed Z-Chain settlement path and consumed exactly once by at least two destination environments.

10.4 Phase 3: cloud-provider pilot

At least one existing cloud scheduler and one independent provider integrate the same job adapter. The pilot must show that proof capture adds acceptable cost to ordinary paid inference or training, that CPU validators do not repeat the full GPU workload, that confidential jobs fail closed when TEE collateral is stale, and that customer privacy/authorization is preserved.

10.5 Phase 4: succinct-proof authority

P3Q may replace the optimistic challenge path and activate as the Z-Chain settlement rollup only after the exact PoAI AIR, parameters, recursive batch policy, A-to-Z state-transition binding, verifier, node integration, resource bounds, and audits are complete. P3Q is Plonky3-derived and pairing-free; that design label is not deployment evidence. Shadow parity with the already active policy is required before authority changes. Here "parity" means that the candidate P3Q statement accepts exactly the same finalized A-Chain receipt transitions; it is an activation test, not part of the public mining story.

10.6 Monetary activation

Governance must fix the genesis DAO address and controls, A-Chain activation height h_0, halving interval H, miner/validator split, job-class reward normalization, per-epoch caps, fee-burn fraction, and emergency back-off policy. The canonical ledger must enforce the one-trillion DAO allocation, one-trillion proof-earned pool, and two-trillion hard cap. Every legacy one-billion or per-chain mint contract must be disabled or upgraded first.

10.7 Back-off

If a proof, availability, metering, PQ authentication, bridge, or economic invariant fails, governance disables new job admission and mining issuance while preserving already finalized receipts and canonical balances. A destination adapter may be paused independently, but it may never fall back to raw proof acceptance.

11 Conclusion

Hanzo Proof of AI makes a stronger claim than AI-themed proof of work. The mined object is a real, pre-admitted AI workload whose consumer, model, input, execution policy, proof policy, reward, and payout destination are fixed before the result. A CPU or GPU provider performs that workload and commits its exact integer output, transcript, and metering. A-Chain PoAI consensus validates that the bound useful work was actually done, consumes the global nullifier, and finalizes one reward receipt.

Ordinary CPUs can validate supported work without repeating the full accelerator execution: small jobs use exact replay, while challenged matrix products use a quadratic Freivalds check against a Merkle-bound transcript. This local check is not the whole security story. Graph binding, coverage, data availability, verifiable metering, challenge finality, and anti-self-dealing admission are consensus-critical. Confidential jobs may add TEE evidence; audited P3Q proofs may eventually replace the optimistic path and settle finalized A-Chain receipt transitions through Z-Chain.

The economic proposition is direct. Existing AI clouds keep the revenue from workloads they already serve and may earn additional AI for making those executions independently accountable. Decentralized providers use the same interface for open jobs and idle capacity. The network rewards consumed AI results rather than chain-derived matrix theater.

AI has a hard two-trillion nominal cap: one trillion permanently allocated to the Hanzo DAO and one trillion reserved for PoAI miners and validators. The proof-earned pool emits geometrically—500 billion, 250 billion, 125 billion, and so on—while fee burns can make net supply deflationary as issuance tends to zero.

The omnichain vision has one work authority and one finalized settlement root. PoAI consensus exists only on Lux A-Chain. Z-Chain compresses and settles those finalized receipt transitions through P3Q; it does not originate mining claims. Lux Quasar and the activated post-quantum validator policy finalize and authenticate the A/Z settlement root. Any supported

chain may consume the destination-bound receipt exactly once, but no destination may accept raw work or create a second AI supply. Hanzo is therefore an open useful-compute market rooted in Lux post-quantum finality, not another isolated AI chain.

The remaining milestone is operational, not rhetorical: a production Zen model must emit the full deterministic transcript, CPU/GPU vectors must agree, fraud and withholding must be caught, metering and anti-farming controls must hold, the two-trillion monetary policy must be enforced, and a P3Q-settled, PQ-authenticated A-Chain receipt must settle exactly once in multiple destinations. Mining should remain disabled until that end-to-end test is green.

Cross-references. The arithmetic and settlement protocol is specified in the Lux AIVM paper (`lux/papers/lux-aivm`) and LP-5000/LP-5200/LP-5300/LP-5301. The intended post-quantum proof substrate is `luxfi/p3q`; threshold-signature research lives in `luxfi/pulsar` and `luxfi/corona`. None of those names is a substitute for an audited, activated end-to-end deployment.

References

[1] R. Freivalds. *Probabilistic machines can use less running time.* Information Processing, 1977.

[2] National Institute of Standards and Technology. *FIPS 204: Module-Lattice-Based Digital Signature Standard.* 2024.

[3] I. Komargodski and O. Weinstein. *Proofs of Useful Work from Arbitrary Matrix Multiplication.* arXiv:2504.09971, 2025.

[4] Z. Kelling. *AIVM: Proof of AI—The Settlement Primitive for Verifiable Machine Computation on Lux.* Lux Papers, 2026.