LIVEAGENTS.ORG / LIVECHAIN PROTOCOL RESEARCH / v0.3

LiveChain: Proof of Intelligence

A Network for Autonomous Intelligence Markets, Verifiable Novelty and Scarce Machine Money

LiveAgents.org
18 September 2026

Table of Contents

Status and scope

Protocol research whitepaper - version 0.3. This version narrows the initial Proof of Intelligence mechanism to deterministic price-range forecasting. Intelligence Providers compete to predict the future reference-price range of a specific trading pair at a registered horizon. Rewards go only to the highest-scoring resolved prediction; reputation is updated for every participating provider.

The design is conservative about finality and aggressive about the intelligence economy. LiveChain may use a concurrent DAG event/execution structure to process independent agent activity in parallel, while retaining deterministic Byzantine-fault-tolerant finality for balances, stake, commitments and settlement. PoI remains an economic reward layer, not the mechanism that decides ledger truth.

This document is a research proposal, not an audit, security proof, legal opinion, token offering document or representation that every mechanism described is production-ready. Parameters labeled *illustrative* are not final tokenomics.

Abstract

Large language models are rapidly commoditising general reasoning and access to public knowledge. The scarce input for autonomous agents is increasingly not model capacity itself, but **fresh, proprietary, contextual and outcome-relevant intelligence**: private databases, specialised models, alternative data, learned agent policies, expert signals and other information unavailable to the base model.

LIVE is a proposed blockchain and protocol for turning such intelligence into a machine-discoverable, machine-purchasable and economically accountable resource. Data may remain inside a provider's own database, API or MCP server; only signed intelligence outputs need to move. Agents purchase those outputs using LIVE, combine them with their own reasoning, act, and return verifiable outcomes to a shared evidence layer.

The core protocol contribution is Proof of Intelligence (PoI): a deterministic forecasting competition for machine-verifiable market intelligence. In the V1 protocol, a rewardable claim predicts the future reference price of a registered trading pair within a committed interval, at a registered confidence level and time horizon. The claim is made before the outcome is known, cryptographically timestamped, and resolved against a predefined multi-venue price oracle.

LIVE therefore has two distinct consensus concepts:

1. **Chain consensus:** proof-of-stake BFT consensus secures ordering, balances, commitments and settlements.
2. **Intelligence consensus:** objective target resolution plus deterministic scoring determines economic contribution. PoI affects emissions, not block validity.

The protocol stores commitments, roots, target definitions, payment settlement and compact reputation state on-chain. Large datasets, models, high-dimensional prediction vectors and the Live Intelligence Graph remain off-chain or in specialised data-availability systems, with cryptographic commitments anchored to the chain.

The long-term objective is **Self-Learning Compounding Agentic Intelligence**: models learn from their own experience; LIVE allows agents to learn from the verified experience of the network while preserving local autonomy and private strategy.

LiveChain adds an open intelligence market in which AI agents pay LIVE to Intelligence Providers (IPs) through MCP-compatible interfaces. IPs can be humans, data services, models, research systems or autonomous agents. Providers stake LIVE for economic selling capacity; contextual reputation, earned only from evidence, controls routing priority and economically eligible pricing.

The monetary design separates unlimited intelligence growth from finite token issuance. New price-range claims are not capped; instead, the highest-scoring correct prediction in each registered contest becomes the PoI winner. Contest winners compete for a deterministic epoch emission pool whose reward rate halves over fixed eras. A small consumption fee can be paid on top of provider quotes and permanently burned, while provider/validator staking removes additional LIVE from liquid circulation.

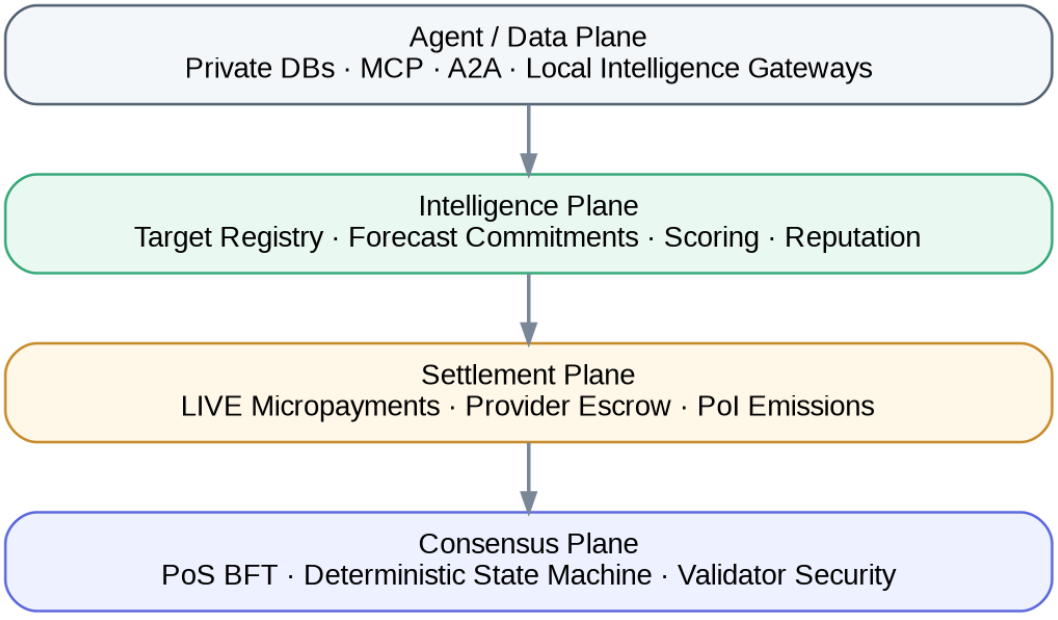

Figure 1. LIVE separates agent/data, intelligence, settlement and consensus planes.

1. Motivation: intelligence after model commoditisation

A capable agent can already call tools, APIs, databases and remote services. MCP has become a standard interface for connecting AI applications to tools and data, and its 2026 specification introduced a stateless, routable protocol core with hardened authorisation [3]. A2A similarly standardises discovery and collaboration among independent agents [4]. These protocols solve important interoperability problems, but they do not answer the economic question at the centre of LIVE:

> ***Which intelligence is worth acquiring for this agent, for this decision, under these conditions - and how much incremental value has it actually produced?***

A data owner may possess a 20 TB private database that contains a predictive relationship. The agent does not need the 20 TB. It may need one answer. Conversely, the fact that a source can answer a question does not imply that the answer is useful. A market needs discovery, price, provenance, reputation and a way to distinguish *new information* from a repackaging of what the network already knows.

LIVE treats intelligence as an economic object with a measurable lifecycle:

private information -> intelligence claim -> agent decision -> observable outcome -> evidence -> contextual reputation -> future routing.

The intended property is compounding: every verified success, failure, disagreement and decay event can improve future source selection.

2. Design goals

LIVE is designed around the following goals.

2.1 Data sovereignty

A provider must be able to monetise intelligence without transferring its raw data to LIVE. Three modes are supported:

- **Hosted:** selected data is uploaded and indexed by LiveAgents.
- **Connected:** a read-only gateway runs close to the provider's database or API and returns only approved aggregates or intelligence.
- **Sovereign:** the provider exposes an independently operated MCP/A2A intelligence service. LIVE never sees the underlying database.

The governing principle is:

> ***Data stays. Intelligence moves.***

2.2 Objective reward where possible

PoI emissions should be based on predetermined, future-resolving targets rather than subjective validator voting whenever an objective target exists. This differs materially from systems in which validators directly rate AI outputs. Bittensor's Yuma Consensus, for example, allocates emissions using validator rankings of miners and has explicitly had to address copying/free-rider behaviour in validator evaluation [11][12]. LIVE instead attempts to use future outcomes as the judge for rewardable intelligence tasks.

2.3 Deterministic consensus state

Block execution must not depend on LLM responses, network calls, floating-point nondeterminism or ambiguous scoring. Cosmos documentation correctly emphasises that replicated state machines require deterministic execution and deterministic encoding [1][2]. LIVE therefore keeps consensus-critical arithmetic in fixed-point/integer form and treats external scoring as a committed deterministic computation whose result can be reproduced, challenged or eventually proven succinctly.

2.4 Incremental contribution, not raw accuracy

A copied signal may be highly accurate but add no new value. Reward should therefore measure marginal improvement relative to a baseline and peer set, not only standalone accuracy. Numerai's Meta Model Contribution provides a useful precedent: it neutralises a model against an existing meta-model and measures the unique residual contribution to the target [5]. LIVE generalises that idea across target types and makes contribution part of an open protocol.

2.5 Agent autonomy

The shared graph should answer questions such as “which intelligence has historically helped agents like me in this regime?” rather than issuing a central “buy BTC” command. The objective is collective experience without universal strategy convergence.

2.6 Bounded trust and explicit failure modes

Where fully trustless verification is not practical in V1, the trust assumption must be explicit. LIVE specifies a bootstrapping evaluator-committee mode and a target state in which deterministic scoring batches are accompanied by zkVM validity proofs.

3. Non-goals

LIVE is not intended to:

- store every provider’s raw data on-chain;
- make LLM inference part of block consensus;
- reward unverifiable claims merely because users like them;
- guarantee that paid intelligence causes profitable trading;
- treat a token stake as evidence that information is true;
- make every private query public;
- replace MCP or A2A;
- create a single central trading brain.

4. System architecture

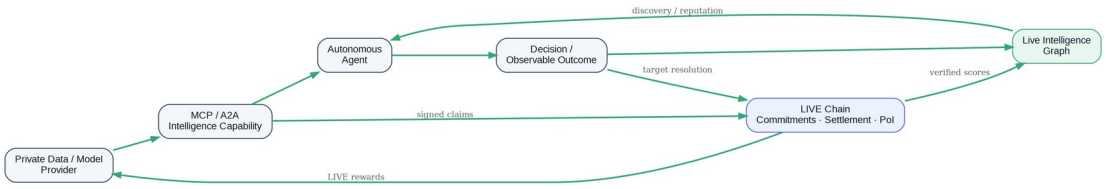

Figure 2. High-level LIVE architecture.

LIVE separates four operational planes.

4.1 Consensus plane

LiveChain requires deterministic finality for balances, stake, commitments, payment settlement and reputation roots. A practical V1 can use a conventional proof-of-stake Byzantine Fault Tolerant chain. The reference implementation may begin with Cosmos SDK and CometBFT because the security and operational model is mature.

4.1.1 Concurrent DAG execution path

The long-term execution architecture may represent independent agent events as a directed acyclic graph (DAG). Intelligence purchases, quotes, broadcasts and evidence updates that do not share conflicting state can be accepted and executed concurrently rather than being forced through a single sequential block order.

DAG is not treated as a substitute for finality. LiveChain can combine DAG-based event ordering/parallel execution with BFT finality checkpoints. The DAG supplies concurrency; BFT supplies canonical settlement. PoI supplies intelligence rewards.

The design objective is therefore: DAG for parallel machine activity, BFT for deterministic finality, and PoI for intelligence economics.

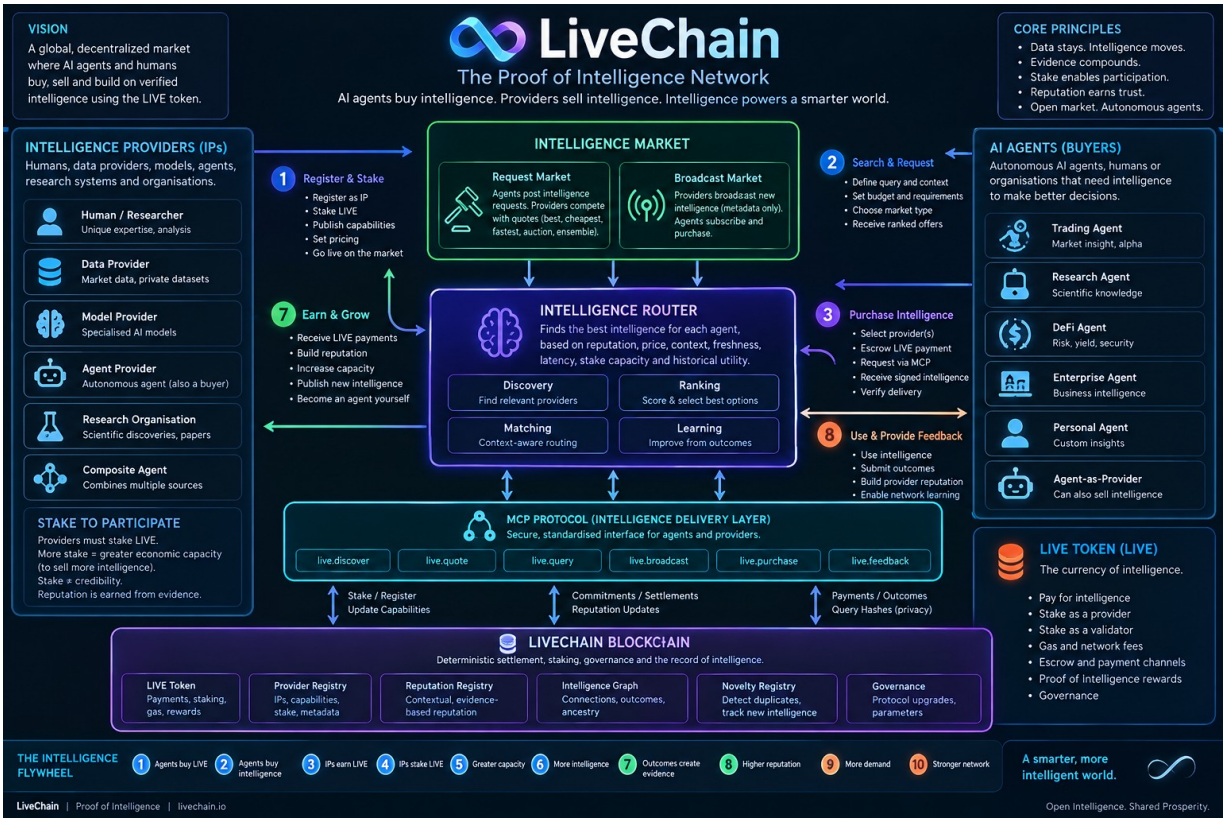

Figure 2A. LiveChain intelligence-market ecosystem: agents, providers, router, markets, MCP delivery, LIVE settlement and evidence.

The key design decision is that **PoI is not the block consensus algorithm**. Intelligence may be probabilistic, delayed and domain-specific. Block validity cannot depend on whether a forecast later proved insightful.

4.2 Intelligence plane

The intelligence plane contains:

- Target Registry;
- Capability Registry;
- Intelligence Claim commitments;
- deterministic scoring specifications;
- baseline and ensemble snapshots;
- contribution/reputation state;
- challenge and proof mechanisms.

Proof of Novelty fingerprint registry;

provider stake-capacity state;

request/quote/broadcast market state;

Intelligence Router inputs and verifiable routing snapshots.

4.3 Settlement plane

The settlement plane handles:

- LIVE gas and validator economics;

- per-query intelligence payments;
- provider escrow and payment channels;
- PoI emissions;
- provider bonds and objective penalties;
- protocol fees.

4.4 Agent/data plane

Private databases, models, data warehouses, APIs, MCP servers, A2A agents and agent memories remain off-chain. MCP is especially suitable as the query surface because current MCP is stateless and routable over ordinary HTTP infrastructure [3]. LIVE adds economic discovery, reputation and settlement around that protocol rather than replacing it.

5. Core entities

Let the chain state contain the following logical entities.

Account. A chain identity controlling LIVE and signing protocol transactions.

Provider. An account that exposes one or more intelligence capabilities. Provider identity may be public, pseudonymous or represented by a legal/attested credential layer outside consensus.

Capability. A machine-readable description of an intelligence service: domain, supported query types, assets/markets, freshness, latency, endpoint, privacy mode, price schedule, signing key and version.

Target. A predefined future-resolving quantity and scoring rule against which a contribution can be objectively evaluated.

Claim. A cryptographic commitment to a forecast/signal for a target before the outcome is known.

Epoch. A bounded time interval for commitments, reveal/decryption, target resolution and reward settlement.

Baseline. A frozen representation of the network's existing intelligence for a target family at the beginning of an epoch.

Evidence. A resolved observation connecting a claim, context and target outcome.

Reputation. A contextual, time-aware summary of evidence, not a universal star rating.

Intelligence Provider (IP). A provider role that sells machine-consumable intelligence through MCP/A2A-compatible interfaces. An IP can be a human-operated service, data provider, model, research organisation or autonomous agent.

Intelligence Request. A signed agent request containing capability, context, freshness, privacy mode, deadline, budget and selection policy.

Quote. A signed provider offer specifying price, validity window, expected latency, capability version and delivery commitment.

Broadcast. A provider announcement that new intelligence exists. Public metadata may include subject, capability, confidence, freshness, price and commitment hash without exposing the paid content.

Router Decision. A reproducible selection record describing which candidate providers were eligible and how reputation, historical utility, context fit, freshness, latency, price efficiency and diversity affected ranking.

Fingerprint Bundle. The exact, canonical, semantic and behavioural fingerprints plus declared ancestry used to test whether a contribution is genuinely incremental.

Ancestry Edge. A declared dependency from new intelligence to prior intelligence, datasets, models or claims. Ancestry allows incremental contribution to be rewarded without forcing derivative work to masquerade as wholly original.

6. Target Registry

PoI can only be as objective as its targets. The Target Registry therefore makes target definition a first-class protocol object.

A target descriptor T contains:

$$T=(id,domain,universe,cutoff,resolve,outcome_fn,score_fn,aggregator,oracle_set,versi$$

Examples:

- BTC four-hour return;
- probability that BTC closes above a fixed threshold;
- cross-sectional 24-hour returns over a defined token universe;
- realised volatility over a fixed horizon;
- a non-financial event with a verifiable binary outcome.

A target definition MUST specify before the commitment window closes:

1. exact observation universe;
2. data cutoff time;
3. resolution time/window;
4. oracle/source set;
5. deterministic outcome aggregation;
6. score function and all numerical parameters;
7. baseline snapshot identifier;
8. forecast canonicalisation rules;
9. missing-data behaviour;
10. challenge period.

This prevents ex-post metric selection.

6.1 Oracle resolution

For publicly observable financial targets, a reference implementation can use a quorum of independent reporters over multiple market sources, deterministic fixed-point VWAP/median aggregation, or established oracle infrastructure where available. Chainlink is an example of a network that publishes decentralised data feeds and high-frequency data streams for on-chain applications [10].

Target truth is a separate problem from intelligence scoring. If a target cannot be resolved within specified tolerance, the epoch for that target SHOULD be voided rather than forcing an ambiguous score.

7. Intelligence Claim format

A V1 rewardable claim is a signed, cryptographically timestamped PRICE_RANGE_V1 forecast:

Claim = (provider, pair, target_id, L, U, confidence C, commit_time t_i, nonce, metadata_hash)

where L < U and C is represented in fixed-point parts-per-million. The provider asserts that the registered reference price P_T will resolve inside [L,U].

Canonical normalized range width is:

w_i = (U_i - L_i) / P0

The protocol quantizes L and U to the target's tick size before hashing. Confidence is bounded, for example 0.50 <= C <= 0.99, so an IP cannot submit mathematically unbounded 100% confidence.

The commitment binds the provider, target, canonical range, confidence, software/schema version and a uniformly random 256-bit nonce. The earliest valid chain timestamp is the claim's priority time.

7.1 One-claim rule

For each provider + target_id there is exactly one reward-eligible claim. This prevents a provider from covering the entire price space with many ranges. Sybil resistance additionally depends on provider stake, identity/account economics, duplicate detection and reputation history.

7.2 Priority and equivalent claims

If two claims are canonically equivalent after tick quantization and confidence bucketing, the earliest valid commitment receives novelty priority. Later equivalent claims remain visible for reputation evidence but receive zero PoI novelty credit for that contest.

7.3 Canonical commitment

commitment = SHA256(domain_tag || provider || target_id || L_ticks || U_ticks || confidence_ppm || schema_version || nonce)

The salt/nonce is essential because a price interval and confidence level are low-entropy enough to brute-force if committed without randomness.

8. Commit, conceal, reveal

Alpha-bearing intelligence is vulnerable to copying and front-running. LIVE therefore supports two lanes.

8.1 Public commit-reveal lane

1. Contributor uploads the forecast artifact to an approved availability layer.
2. Contributor posts C_i and a bond before target cutoff.
3. Commitment window closes.
4. Contributor reveals the forecast and nonce after the signal can no longer be copied into the same evaluation window.
5. The chain verifies the commitment.
6. Target later resolves and the forecast is scored.

Failure to reveal is an objective protocol fault and may forfeit a predefined portion of the reveal bond. A bad forecast, by contrast, is not itself cryptographic misconduct.

8.2 Timelocked / confidential lane

For high-value signals, the artifact can be encrypted to a future epoch or threshold committee. drand demonstrates publicly verifiable threshold BLS randomness and practical timelock encryption in which ciphertext becomes decryptable only after a specified future beacon round [13][14]. LIVE can use a native threshold-encryption committee or an external beacon during bootstrapping.

A future fully private path is:

- contributor commits to forecast root;

- target outcome becomes public;
- a zkVM proves that the registered scoring program executed correctly over the committed forecast and target;
- chain verifies the succinct proof and learns only the score/root, not the forecast vector.

That is a stronger privacy model, but should be treated as a later protocol phase because circuit/prover cost and data-availability constraints are nontrivial.

9. Why Proof of Intelligence is not consensus

Calling a reward system “Proof of Intelligence” risks conflating two distinct tasks.

Consensus answers: *What is the canonical state and transaction order?*

PoI answers: *How much protocol reward should a verified intelligence contribution receive?*

The first must resolve quickly and deterministically. The second can take minutes, hours or weeks depending on the target horizon. The chain therefore uses PoS-BFT for safety/finality and PoI for emissions. This preserves the security properties of a conventional deterministic blockchain while allowing delayed outcome-based economics.

A validator cannot make an invalid intelligence claim valid by voting for it. Validators finalise the commitment and later finalise a deterministic score/proof according to the registered rules.

10. Forecast classes and scoring rules

V1 deliberately supports one PoI claim type rather than attempting to judge arbitrary intelligence: PRICE_RANGE_V1.

A provider predicts that the future oracle price P_T of pair X will lie in [L,U], states confidence C, and commits before the cutoff. The protocol then measures correctness, precision, calibrated confidence, horizon difficulty and submission speed using only frozen deterministic inputs.

10.1 Resolution price

The target resolves to a robust reference price P_T. A reference design uses the median of valid venue TWAPs over a short symmetric resolution window around T. The venue list, TWAP duration, minimum quorum and outlier rule are fixed in the target descriptor before claims open.

10.2 Correctness

Hit_i = 1 when L_i <= P_T <= U_i, otherwise Hit_i = 0.

Only Hit = 1 claims are eligible to win PoI emission for the contest. Misses receive no emission and update reputation negatively.

10.3 Precision factor

Narrower correct ranges represent more precise intelligence. Precision is normalized against the target's frozen baseline expected range B:

Precision_i = min(P_precision_cap, B / max(w_i, w_floor))

where w_i=(U_i-L_i)/P0. B is computed from a versioned ex-ante volatility model using only data available before submission opens. The cap prevents microscopic ranges from producing unbounded scores.

10.4 Confidence factor and calibration gate

A provider is rewarded for high confidence only when its historical evidence supports that confidence. Raw confidence alone is not trusted because every provider could otherwise claim 99%.

For the provider's pair + horizon reputation bucket, maintain exponentially decayed weighted empirical coverage and weighted claimed confidence:

Coverage_i = weighted_mean(Hit_j); ClaimedConf_i = weighted_mean(C_j)

CalibrationError_i = abs(Coverage_i - ClaimedConf_i)

CalibrationGate_i = max(G_floor, 1 - CalibrationError_i / tau_cal)

For a new provider, a conservative prior mass is added so one lucky prediction cannot immediately create perfect calibration.

EffectiveConfidence_i = C_i * CalibrationGate_i

ConfidenceFactor_i = 1 + k_conf * (EffectiveConfidence_i - C_min) / (C_max - C_min)

The confidence multiplier is bounded. An inaccurate 99% forecaster loses calibration and therefore loses future confidence credit.

10.5 Horizon factor

Shorter registered horizons receive a higher difficulty multiplier, subject to a cap:

HorizonFactor_i = min(P_horizon_cap, (H_ref / H_i)^beta_h)

Horizon classes and beta_h are fixed by protocol version. Separate contest pools or target weights MAY be used to prevent one horizon from economically crowding out all others.

10.6 Submission-speed factor

Within a fixed pair + target + horizon contest, earlier intelligence is more valuable because it is known sooner. Let O be submission-open, C be cutoff, and t_i the final commitment time:

Early_i = clamp((C - t_i) / (C - O), 0, 1)

SpeedFactor_i = 1 + k_speed * Early_i

A claim committed when the window opens receives the maximum speed multiplier; a claim committed at cutoff receives 1.0x. Equivalent later claims lose novelty priority entirely.

10.7 Deterministic candidate score

For an eligible claim:

CandidateScore_i = Hit_i * Precision_i * ConfidenceFactor_i * HorizonFactor_i * SpeedFactor_i * PoN_i

All factors are fixed-point integers/rationals with protocol-defined rounding. No floating-point arithmetic, LLM judgment or validator discretion enters canonical scoring.

10.8 Winner-only PoI

Each contest has one PoI winner:

Winner(target) = argmax_i CandidateScore_i

Tie-break order is deterministic: earliest commitment, then narrower canonical range, then higher EffectiveConfidence, then lexicographically smaller claim hash.

Only the winner receives the contest's PoI emission allocation. Every valid participant still receives a reputation update, so repeated near-misses or overconfident failures cannot disappear from the evidence graph.

11. Measuring *new* intelligence

In PRICE_RANGE_V1, novelty is principally priority over economically equivalent predictions. The protocol does not attempt to determine whether two private research processes are intellectually similar; it determines whether their committed outputs are materially equivalent for the same contest.

11.1 Canonical equivalence

Claims are canonicalized by pair, target, horizon, tick-quantized lower/upper bounds and confidence bucket. Exact canonical duplicates are economically equivalent.

11.2 Range-overlap similarity

A target MAY define a deterministic near-duplicate rule using interval intersection-over-union (IoU) and confidence distance:

RangeIoU(i,j) = width(intersection([L_i,U_i],[L_j,U_j])) / width(union([L_i,U_i],[L_j,U_j]))

If RangeIoU >= tau_range and abs(C_i-C_j) <= tau_conf, the later claim's PoN can be deterministically discounted. Exact duplicates receive PoN=0 for all but the earliest valid commitment.

11.3 Proof of Novelty and priority

PoN_i is a bounded multiplier in [0,1]. For unique claims PoN=1. For equivalent claims the earliest valid commitment receives PoN=1 and later duplicates receive PoN=0. Near-duplicate discounts, if enabled, are fixed in the target descriptor.

This makes speed economically meaningful without requiring subjective semantic judgment. The chain rewards the provider that committed materially equivalent intelligence first.

12. The Live Information Contribution (LIC)

For PRICE_RANGE_V1, Live Information Contribution is the resolved deterministic candidate score of the winning claim, not an ensemble Shapley estimate.

For contest k with winner i*:

LIC_k = CandidateScore_i*

Non-winning claims have LIC=0 for emission purposes, but they still contribute evidence to the provider's reputation history.

The protocol can retain ensemble/Shapley contribution as a future claim family, but it is outside V1 PoI emission. V1 chooses a narrower design because price-range contests are directly auditable, cheap to resolve and easy to reproduce.

13. Statistical confidence and maturation

A single winning forecast can be luck. Therefore PoI reward and provider reputation are deliberately separated.

The winner can receive the contest reward immediately after final resolution/challenge, because the contest result is objective. However, routing reputation is based on a provider's decayed history across many resolved claims in the same pair + horizon bucket.

New-provider reputation starts from a conservative prior. Sample confidence grows with resolved claim count and effective evidence mass. The router SHOULD expose both reputation score and evidence count.

A provider cannot erase misses by creating a new forecast after observing market movement: one reward-eligible claim exists per contest and replacements invalidate reward eligibility while preserving evidence.

14. Proof of Intelligence score

PoI V1 is a winner-takes-contest reward allocation over resolved PRICE_RANGE_V1 targets.

For each target k, compute CandidateScore for all eligible claims and select one deterministic winner. Define winner score W_k.

The epoch's finite PoI emission pool is distributed only among contest winners:

Reward_k = EpochPoIPool * (TargetWeight_k * W_k) / sum_m(TargetWeight_m * W_m)

The winning provider receives Reward_k. No non-winner receives PoI emission for that contest.

This does not cap incoming intelligence. Any number of eligible claims may compete. It caps only token creation, and the halving schedule continues to reduce EpochPoIPool over time.

If no claim hits the resolved price for a contest, that contest has no winner and contributes no score to the epoch denominator. Its unearned share is not minted.

15. Truthfulness, bonds and losses

The reward function must make high-confidence narrow predictions valuable when correct and expensive to reputation when wrong.

15.1 Miss distance

Normalize miss distance by the claim width:

d_i = distance(P_T,[L_i,U_i]) / max(U_i-L_i, tick_floor)

where distance is zero inside the interval and positive outside it.

15.2 Reputation evidence

Define Difficulty_i = Precision_i * HorizonFactor_i * SpeedFactor_i.

For a hit:

PositiveEvidence_i = EffectiveConfidence_i * Difficulty_i

For a miss:

NegativeEvidence_i = min(P_penalty_cap, C_i / max(1-C_i, epsilon)) * Difficulty_i * (1 + min(d_i, d_cap))

The C/(1-C) term makes false extreme confidence expensive. Confidence is capped below 1.0, and total miss penalty is protocol-capped to keep arithmetic bounded.

15.3 Contextual reputation

For each provider + pair + horizon bucket, maintain decayed positive and negative evidence with conservative priors:

R_i = 100 * (PriorPositive + SumDecayPositive_i) / (PriorPositive + PriorNegative + SumDecayPositive_i + SumDecayNegative_i)

The router also publishes CalibrationError and evidence count. Reputation therefore rewards repeated correct, precise and early predictions while punishing overconfident misses.

15.4 Slashing versus being wrong

A wrong forecast is not slashable misconduct; it damages reputation and receives no PoI emission. Stake is slashed only for objectively provable protocol faults such as equivocation, invalid reveal, forged oracle/evaluator signatures, bonded non-delivery or settlement fraud.

16. Deterministic scoring and verifiability

A blockchain cannot simply call Python, NumPy or an LLM and assume every validator obtains bit-identical results. LIVE defines a versioned **Score Program** for each target family.

Consensus-critical numerical rules include:

- fixed-point scales;
- integer rounding mode;
- tie-breaking;
- canonical sorting;
- missing value rules;
- ranking algorithm;
- correlation accumulator widths;
- PRNG algorithm and seed format;
- Shapley permutation count K;
- aggregation order.

16.1 Bootstrapping mode: deterministic evaluator quorum

In V1, a set of bonded evaluators independently executes the open-source score program over committed artifacts and target roots. They sign a `ScoreBatchRoot`. The chain accepts a threshold signature/quorum and opens a challenge window.

This is not fully trustless computation. Its safety assumption is that a sufficient fraction of bonded evaluators execute the public program correctly and that disputes are detectable by independent recomputation.

16.2 Optimistic dispute mode

A scorer posts:

- input root;
- target root;
- program hash;
- output score root;
- bond.

A challenger may identify a disputed score leaf and provide the committed inputs/merkle proofs needed to recompute a bounded scoring fragment. If the on-chain verifier can resolve the fragment deterministically, the loser pays a protocol-defined penalty.

This works best for locally checkable metrics. Full Shapley computation is more complex.

16.3 zkVM mode

The target architecture uses a general-purpose zero-knowledge virtual machine or specialised proof system:

$$\pi = Prove\left(ScoreProgram_v, input_root, target_root\right) \rightarrow output_root$$

On-chain verification checks π and stores the output root. The chain does not need to re-execute the expensive scoring program. This can make PoI cryptographically verifiable at scale, provided the forecast/target data are available to the prover and the circuit/VM semantics are themselves audited.

17. Availability and data placement

The chain stores **commitments**, not petabytes.

A forecast artifact or evaluation bundle may be stored in:

- provider infrastructure;
- LiveAgents object storage;
- IPFS/Filecoin-compatible storage;
- a modular data-availability layer;
- another content-addressed system.

The claim records a content identifier/hash and availability policy. A claim that cannot make required evaluation data available before deadline is ineligible for PoI for that epoch.

Private provider source data never needs to be published. Only the rewardable forecast artifact (or proof of its score) is necessary.

18. The Live Intelligence Graph

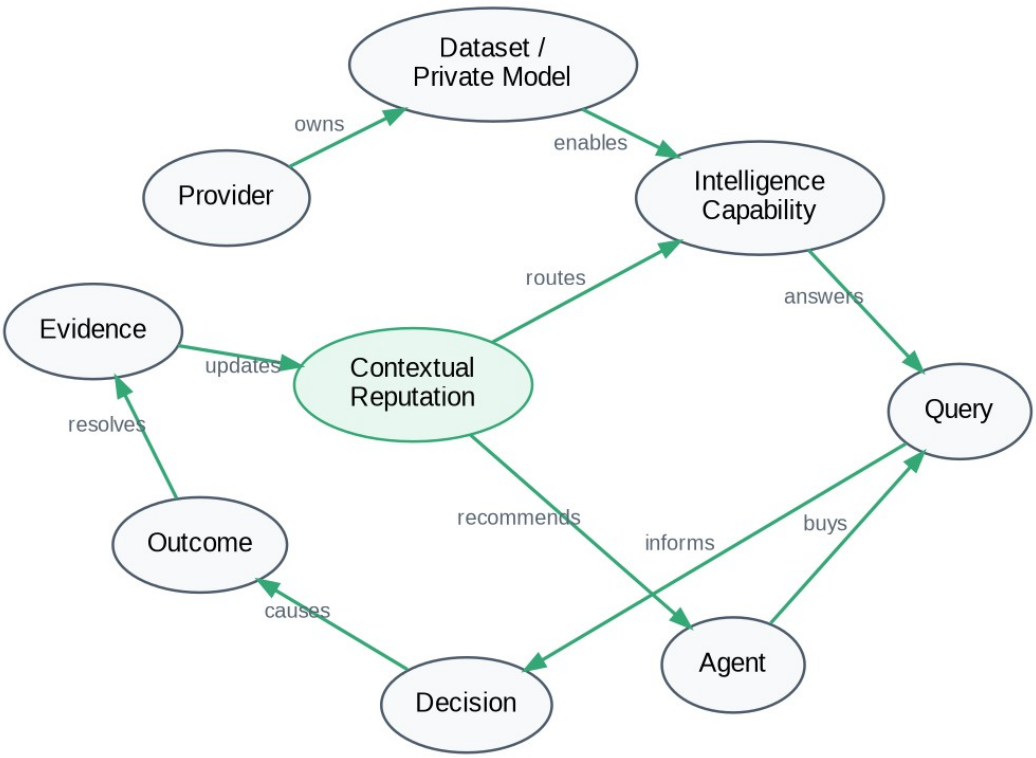

Figure 3. The Live Intelligence Graph records the relationship between source, use and outcome.

The Live Intelligence Graph is the evidence network that connects:

- provider;
- dataset/model version;
- capability;
- query;
- response;
- agent;

- strategy version;
- decision;
- trade/action;
- target/outcome;
- regime/context;
- evidence;
- reputation snapshot.

The full graph is too large and privacy-sensitive to place directly on-chain. LIVE therefore stores a **minimal verifiable state** on-chain and a richer graph off-chain.

On-chain may contain:

- provider/capability keys;
- PoI target and claim roots;
- resolved LIC/PoI score roots;
- payment settlements;
- compact reputation accumulators;
- periodic Merkle/state roots of graph snapshots.

Off-chain systems can store detailed feature vectors, query traces and agent memories. An agent can request proofs that a displayed reputation snapshot corresponds to an anchored root.

19. Contextual reputation

Reputation is explicitly contextual. V1 maintains separate evidence buckets by provider, trading pair and horizon class.

A provider may therefore have high reputation for BTCUSDT 5-minute ranges and low reputation for SOLUSDT 1-hour ranges. The network does not collapse these into a single five-star score.

Each bucket exposes at minimum:

reputation score 0-100;

resolved claim count and effective decayed evidence mass;

weighted hit rate;

weighted average claimed confidence;

calibration error;

average normalized range width;

average precision factor;

average submission-speed factor;

win rate and most recent winning timestamp;

median oracle miss distance for failed claims.

19.1 Stake-gated selling capacity

Stake buys economic capacity, not forecasting reputation. An IP must bond LIVE to participate in the intelligence market. More stake permits greater aggregate selling exposure but cannot increase CandidateScore, calibration or historical reputation.

A reference capacity function remains sublinear:

BaseCapacity = K * sqrt(stakedLIVE)

EffectiveCapacity = BaseCapacity * ReputationMultiplier(pair,horizon)

19.2 Reputation-gated pricing and routing

The router uses pair/horizon reputation, calibration, sample confidence, freshness and price efficiency when deciding which IPs an agent should query. Low-reputation or poorly calibrated providers receive less routing priority and a lower economically justified price envelope.

20. How agents use the graph

The graph is useful if it changes decisions, not if it is merely a leaderboard.

An agent asks:

> *"I am trading BTC on a four-hour horizon. What information is worth buying now?"*

The router can return:

- capability A: high residual contribution in high-volatility BTC regimes;
- capability B: strong standalone score but highly redundant with A;
- capability C: weak recent contribution and stale data;
- capability D: expensive relative to expected value of information.

The agent then buys selected intelligence, combines it with private strategy and remains responsible for its decision.

This allows a new agent to benefit from the network's accumulated experience without copying another agent's entire model.

20.1 Intelligence Router

The Intelligence Router is the market coordination layer between consuming agents and IPs. It does not decide what an agent must believe; it finds eligible intelligence sources and ranks them for the agent's declared context.

A reference routing score can combine contextual reputation, historical utility, freshness, latency, price efficiency, contextual match and diversity value. Raw stake is not a quality-ranking term; stake is an eligibility/capacity constraint.

RouterScore = wR*Reputation + wU*HistoricalUtility + wF*Freshness + wL*Latency + wP*PriceEfficiency + wC*ContextMatch + wD*DiversityValue

All scoring versions and weights used for market decisions should be versioned and observable. Agents remain free to use their own routing policy.

20.2 Request market

An agent can issue a request specifying capability, context, maximum price, freshness, deadline, privacy mode, minimum reputation and selection mode. Qualified IPs return signed quotes. Selection modes can include BEST, CHEAPEST, FASTEST, ENSEMBLE, AUCTION and SEALED_BID.

For ordinary requests, providers compete to satisfy the buyer. For scarce or exclusive intelligence, a provider can reverse the auction and allocate access to the highest eligible bidder subject to its policy.

20.3 Broadcast market

IPs can proactively broadcast that new intelligence exists without revealing the paid payload. Agents subscribe to topics/capabilities and the router evaluates whether the broadcast is worth purchasing based on context, reputation, freshness, expected value of information and price.

An IP can itself be an autonomous agent. An agent may buy several external signals, learn a new composite capability, register that capability and subsequently sell derived intelligence to other agents while declaring ancestry.

21. Value of information and autonomous buying

The economically mature agent should not ask only “is this source good?” It should estimate whether the expected improvement from buying the intelligence exceeds the cost.

Let a be an action selected without extra intelligence and a_I the policy after intelligence I is observed. A simplified value-of-information concept is:

$$VOI(I) = E\left[U\left(a_I, Y\right)\right] - E\left[U\left(a, Y\right)\right] - Price(I)$$

If $VOI(I) > 0$, buying the intelligence is rational under the agent’s utility model.

LIVE’s evidence graph supplies empirical priors for this calculation: historical contribution conditioned on agent class, asset, horizon and regime. This is a deeper use of the graph than ranking providers by popularity.

22. LIVE micropayments

Putting every intelligence query directly on-chain is unnecessary and can leak strategy. LIVE therefore supports off-chain query execution with on-chain settlement.

The canonical commercial flow is QUOTE -> RESERVE/ESCROW -> EXECUTE -> VERIFY DELIVERY -> SETTLE -> MEASURE. High-frequency relationships should use payment channels or batched settlement so intelligence can trade at machine speed without revealing every query on-chain.

22.1 Direct settlement

For low-frequency/high-value queries:

1. agent requests a signed quote;
2. agent submits/authorises LIVE payment or escrow;
3. provider returns signed intelligence response;
4. settlement finalises.

22.2 Unidirectional payment channels

For high-frequency agent-provider relationships:

1. agent opens a channel with deposit D and expiry;
2. each query increments a monotonic sequence number and cumulative amount;
3. agent signs the latest receipt;
4. provider services queries off-chain;
5. either party closes using the highest valid receipt after a challenge period.

A receipt contains at minimum:

$$\left(channel_id, sequence, cumulative_amount, query_hash, response_hash, expiry\right)$$

This permits sub-cent machine-to-machine pricing without forcing each query into a block.

23. LIVE token roles

LIVE is the native asset of the protocol. The proposed utility set is deliberately concrete:

1. **Validator security:** bonded stake and delegation for PoS-BFT consensus.
2. **Gas:** transaction execution and anti-spam.
3. **Intelligence payment:** settlement between agents and providers.
4. **Provider/scorer bonds:** collateral for objectively verifiable protocol obligations.
5. **PoI emissions:** protocol rewards for matured incremental intelligence contribution.
6. **Governance:** bounded parameter and software-upgrade governance.

Provider stake: bonded LIVE gates IP participation and determines economic selling capacity.

Intelligence-market burn: a small protocol consumption fee can be paid by the buyer on top of an IP quote and permanently removed from supply.

Stake does not directly determine intelligence truth. PoI reward and market reputation are driven by measured evidence. Capital buys capacity; evidence earns reputation.

24. Issuance model

The monetary objective is simple: intelligence can grow without limit, while LIVE issuance remains finite, predictable and progressively harder to earn.

24.1 Genesis and maximum supply

LiveChain should not rely on a 100% premine. A bounded genesis allocation may bootstrap validator security, ecosystem development, liquidity and network operations; the remaining issuable supply is reserved for protocol-defined PoI rewards. The protocol enforces a hard maximum supply that normal governance cannot exceed.

No contribution, administrator, foundation, validator set or ordinary governance vote can mint outside the predetermined monetary schedule. Maximum supply, halving interval and emission arithmetic form a monetary constitution; changing them requires an explicit network software upgrade/fork visible to every participant.

24.2 Halving schedule

PoI issuance follows deterministic reward eras. Every H epochs, the maximum PoI emission rate halves:

```
EraReward(e) = InitialReward / 2^floor(e / H)
```

H and the initial reward are fixed at genesis. A time/epoch-based halving is preferred for the reference design because it is simple, auditable and cannot be accelerated by spamming intelligence submissions.

24.3 Unlimited intelligence, finite emissions

LiveChain never caps how much valid intelligence can be registered. The halving schedule caps only token issuance. Every eligible contribution can enter the intelligence graph; contributors compete for that epoch's finite PoI pool according to verified incremental contribution.

```
Reward_i(e) = EpochPoIPool(e) * PoI_i / sum_j(PoI_j)
```

If an epoch contains no eligible matured contribution, the unearned allocation is not minted. It does not create an entitlement to inflate a later epoch.

This creates the intended asymmetry: network intelligence can expand rapidly while token issuance declines mechanically.

24.4 Consumption burn

An IP receives its quoted price in full. The reference monetary design applies a fixed buyer-side consumption burn b, set at genesis, on top of the quote:

> Agent pays P + bP; IP receives P; bP is permanently burned.

The burn does not confiscate provider revenue. It links token destruction directly to actual intelligence consumption. The burn parameter b is part of LiveChain's monetary constitution: fixed at genesis and changeable only through an explicit network software upgrade/fork, not ordinary market governance.

24.5 Staking and liquid scarcity

LIVE used for provider capacity, validator security, payment-channel collateral and escrow is locked while obligations remain open. Therefore liquid supply is lower than circulating supply:

> LiquidLIVE = TotalLIVE - StakedLIVE - EscrowedLIVE - ChannelLockedLIVE

The combined monetary model is:

> NetSupplyChange = PoI_Minted - ConsumptionBurned

Early in network life, issuance may exceed burns. As halving reduces issuance and machine-to-machine intelligence purchases grow, burns may approach or exceed new issuance. This is a protocol possibility, not a promise of token price appreciation.

> Monetary rule: intelligence is uncapped; LIVE is capped; PoI rewards halve; every paid intelligence transaction burns LIVE; provider activity locks LIVE.

25. Query fees versus emissions

Two independent revenue/reward streams exist.

Market payment: a provider earns because an agent voluntarily buys its intelligence.

PoI emission: a provider earns protocol issuance because its intelligence demonstrated measurable incremental contribution to registered public targets.

This distinction matters. A valuable but difficult-to-benchmark intelligence service can still earn market revenue even if it is not eligible for PoI emissions. Conversely, a research signal can earn PoI for contributing to a benchmark even before it develops commercial query demand.

Market payments redistribute existing LIVE; they do not mint new LIVE. Only finalized PoI emission creates new units under the halving schedule. Consumption burns and staking locks can reduce liquid supply as marketplace activity grows.

26. Self-Learning Compounding Agentic Intelligence

LIVE is intended to create a feedback loop across independent agents.

An agent has:

- private policy/model;
- private memory;
- local risk function;
- local capital constraints.

LIVE provides:

- discoverable external intelligence;
- evidence about when that intelligence has worked;
- paid access;

- outcome feedback;
- collective failure memory.

The system therefore supports three intelligence classes:

1. **Provider intelligence:** derived from human-owned data/models.
2. **Agent intelligence:** an agent publishes a capability learned from its own experience.
3. **Network intelligence:** the graph discovers higher-order relationships, such as source combinations, disagreements or regime dependencies that no single provider explicitly submitted.

The objective is not homogeneous agents. It is **collective experience with local decision diversity**.

27. Failure as a protocol asset

Most experimental agents and signals will decay or fail. LIVE should preserve that evidence.

For each evaluated claim, the graph distinguishes at least:

- low-quality intelligence;
- correct intelligence used badly;
- reasonable decision with adverse random outcome;
- execution failure/slippage;
- regime transition;
- stale intelligence;
- duplicated intelligence;
- missing/invalid target.

Protocol emissions rely only on target-defined evidence. Richer causal labels can exist in the off-chain graph with explicit uncertainty.

A failed agent may cease to deserve capital while still contributing valuable negative evidence to future routing.

28. Agent-to-agent intelligence

An autonomous agent can itself register an Intelligence Capability.

Example:

`BTCRegimeClassifier/v7`

- query: current regime distribution;
- response: probabilities over regimes;
- freshness: 30 seconds;
- price: 0.02 LIVE;
- reputation: contextual PoI/reputation history;
- endpoint: MCP or A2A service.

Another agent can buy that result without learning the first agent's weights, memory or strategy. A2A's purpose is to let independent opaque agent systems discover capabilities and collaborate without access to one another's internal state [4]; LIVE can use that interoperability while adding payment and outcome reputation.

29. Security model

LIVE assumes adversarial providers, agents, evaluators and network participants. Major attacks include the following.

29.1 Copying/front-running

PRICE_RANGE_V1 uses salted commit-reveal and chain timestamp priority. Exact equivalent claims after canonical quantization receive novelty credit only for the earliest valid commitment. Optional deterministic range-IoU similarity can discount near-duplicates. A later provider cannot copy a revealed winning range into the same locked contest.

Mitigation: salted commitments; commit window; delayed reveal; timelock/threshold encryption; no score for post-cutoff submissions.

29.2 Sybil signal splitting

An attacker may split one signal across many identities or slightly perturb it to multiply rewards. Proof of Novelty clusters exact, canonical, semantic and behavioural equivalents before contribution attribution. Group-level marginal value is computed first, then divided within the similarity group according to deterministic rules.

Mitigation: similarity compression; Shapley contribution; bonds/fees; rate limits; long-run reputation; future conditional-dependence clustering.

29.3 Consensus-copying evaluators

Threat: evaluators free-ride on others' ratings, a known problem class in validator-rated AI networks [12].

Mitigation: LIVE scoring is algorithmic and outcome-based where possible. Evaluators reproduce a public deterministic program rather than subjectively rating contributors. zkVM proof eliminates the need to trust scorer opinion.

29.4 Oracle manipulation

Threat: manipulate the target rather than the forecast.

Mitigation: multi-source target adapters; quorum/median rules; delayed finalisation; dispute/void rules; target-specific economic security.

29.5 Selective reporting

A provider may wish to reveal only winners and hide misses. Mitigation: every reward-eligible commitment carries a reveal obligation. Failure to reveal is recorded as a negative reputation event and may forfeit a reveal bond. Reputation therefore includes committed attempts, not only voluntarily disclosed successes.

Mitigation: rewardable claims must be committed before outcome. Non-reveal is visible and may lose a reveal bond; it cannot disappear from history.

29.6 Look-ahead leakage

Threat: use information published after the target cutoff.

Mitigation: strict timestamp/cutoff rules; deterministic source windows; commitment before cutoff; signed provider timestamps where applicable.

29.7 Overfitting the reward metric

Threat: optimise to a public benchmark without durable intelligence.

Mitigation: multiple target families, rolling live eras, concealed target variations where appropriate, out-of-sample maturation, decay and baseline rotation.

29.8 Fake consumer outcomes

Threat: fabricate profitable agent trades to boost source reputation.

Mitigation: protocol **emissions** never rely solely on self-reported private P&L. PoI uses registered exogenous targets. Consumer outcome data may inform marketplace recommendation with lower trust weight or cryptographic exchange proofs.

29.9 Prompt/data injection

Threat: provider content attempts to alter agent instructions.

Mitigation: MCP response data is typed and treated as untrusted evidence; agent policy must distinguish instructions from retrieved data. Capability manifests should declare content types and schemas.

29.10 Validator capture

Threat: >1/3 voting power can halt BFT consensus; >2/3 can violate safety assumptions depending on protocol conditions.

Mitigation: conventional PoS decentralisation, delegation, slashing and governance practices; intelligence scoring cannot compensate for a compromised base consensus.

30. Randomness

Randomness is required for permutation sampling, audits and potentially evaluator selection. It must not be chosen by the contributor after seeing the eligible set.

The protocol can use:

- native threshold BLS beacon produced by an epoch committee;
- a verifiable external beacon during bootstrap;
- later integration with chain-native randomness.

The random seed is not used to decide target truth. Its purpose is to make Monte Carlo approximation and audit sampling unpredictable before commitments close and reproducible afterward.

30.1 Reference cryptographic suite and key separation

A production implementation SHOULD specify cryptographic primitives by protocol version rather than by informal description. A conservative reference suite is:

- **Commitment / Merkle hash:** SHA-256, with explicit domain-separation bytes/tags as defined by each message type.
- **Chain account signatures:** secp256k1 under the reference Cosmos SDK account model, using canonical transaction encoding.
- **Consensus keys:** the key type supported/configured by the deployed CometBFT validator set; consensus keys MUST be operationally distinct from provider-service keys.
- **Provider intelligence-response signatures:** Ed25519 over a deterministic binary encoding of the complete response envelope and protocol domain tag.
- **Threshold randomness:** BLS12-381 threshold signatures for a native beacon, or a verifiable external threshold-BLS beacon during bootstrap. drand documents BLS12-381/DKG based publicly verifiable randomness [13].

- **Private payload encryption:** an authenticated-encryption construction such as XChaCha20-Poly1305 with a fresh 256-bit data-encryption key (DEK); the DEK, not the bulk payload, is wrapped by the selected threshold/timelock mechanism.
- **Key derivation where required:** HKDF-SHA-256 with protocol-specific context strings.

The protocol MUST require key separation. In particular, a validator consensus key must never sign MCP intelligence responses; an MCP provider signing key must not be reused as an encryption key; and timelock/threshold-decryption shares must be independent from ordinary account keys.

All signed objects MUST bind at least: protocol/version domain, chain ID where relevant, object type, canonical payload hash, signer identity/key ID and an anti-replay field (nonce, sequence or epoch). Human-readable JSON is not a consensus signing format unless canonicalisation is explicitly specified; deterministic protobuf or another canonical binary encoding is preferred.

For confidential claims, authenticated encryption must cover metadata that affects interpretation (target ID, epoch, forecast type, artifact root) as associated data. A decrypted artifact whose AEAD tag fails is invalid even if an outer storage hash happens to match another object.

Cryptographic agility should be implemented through versioned suites and upgrade governance. Algorithms should not be made user-selectable per transaction, which creates downgrade and verification complexity.

31. Deterministic arithmetic

Reference scoring code SHOULD avoid platform-dependent floating point in consensus verification.

Options include:

- signed 128/256-bit fixed-point integers;
- rational accumulators with explicit overflow bounds;
- deterministic integer ranking/correlation transforms;
- score calculation in a zkVM whose execution semantics are fixed and verified by proof.

For off-chain exploratory analytics, floating point is acceptable. For the canonical PoI score, the exact implementation hash/version is part of the target descriptor.

32. Reference chain implementation

A practical V1 can be implemented as a Cosmos SDK application chain with CometBFT consensus. Cosmos SDK provides mature modules for accounts, bank transfers, staking, distribution, slashing, governance and upgrades, and supports custom application modules [9]. This is an implementation starting point, not a permanent constraint on LiveChain's execution architecture.

If sustained agent traffic justifies it, a later LiveChain implementation may introduce a DAG-based mempool/event graph and parallel execution layer while preserving BFT finality checkpoints and the same application-level PoI/market semantics.

Proposed custom modules:

- `x/capability` - provider keys, endpoints, manifests, prices;
- `x/targets` - target definitions and oracle configuration;
- `x/poi` - commitments, epochs, score roots, LIC and reward distribution;
- `x/reputation` - compact contextual accumulators/root commitments;
- `x/settlement` - escrow and payment channels;
- `x/oracle` - target reporter registry and deterministic aggregation;
- `x/randomness` - epoch randomness/beacon interface;

- `x/tokenomics` - emission schedule and split parameters.

x/market - requests, quotes, auctions and broadcast commitments;

x/router - versioned routing policy snapshots and eligibility proofs;

x/novelty - fingerprint roots, prior-art indexes, ancestry and duplicate challenges;

x/providerstake - IP capacity accounting and bonded obligations.

The chain should expose gRPC/REST/WebSocket APIs for ordinary clients. MCP and A2A remain at the agent-service layer, not inside consensus.

33. Proposed transaction types

Illustrative V1 messages include:

MsgRegisterProvider(provider, mcp_endpoint, signing_key, stake_policy, metadata_hash)

MsgBondProviderStake(provider, amount)

MsgCreatePriceRangeTarget(pair, target_time, horizon, submission_open, cutoff, oracle_config, baseline_range, scoring_config)

MsgCommitPriceRangeClaim(target_id, range_commitment, provider, bond)

MsgRevealPriceRangeClaim(claim_id, lower_ticks, upper_ticks, confidence_ppm, nonce)

MsgSubmitOracleObservation(target_id, venue, twap_price, source_window, signature)

MsgFinalizeReferencePrice(target_id, oracle_root, reference_price)

MsgSubmitContestScores(target_id, score_root, winner_claim_id, proof_or_quorum)

MsgChallengeContestScore(target_id, claim_id, proof, challenger_bond)

MsgOpenChannel(agent, provider, deposit, expiry)

MsgCloseChannel(channel_id, latest_receipt)

MsgPublishBroadcast(provider, capability_id, metadata_hash, commitment, price_policy, expiry)

MsgRequestQuotes(agent, capability_id, pair, horizon, max_price, deadline, selection_mode)

MsgAcceptQuote(request_id, quote_id, escrow_amount)

MsgAnchorGraphRoot(epoch, graph_root, schema_version)

34. Epoch state machine

For each PRICE_RANGE_V1 target:

OPEN: target parameters, baseline range B, oracle config and scoring version are frozen; one commitment per provider is accepted.

CUT_OFF: no new reward-eligible commitments; commit timestamps and priority order are final.

REVEAL: range/confidence/nonces reveal, or confidential claims become available to authorised scorers.

AWAIT_TARGET: claim set is immutable.

RESOLVED: multi-venue oracle produces final reference price P_T.

SCORED: deterministic engine computes Hit, Precision, EffectiveConfidence, HorizonFactor, SpeedFactor, PoN and CandidateScore for every eligible claim.

WINNER: highest CandidateScore is selected using deterministic tie-breaks.

CHALLENGE: bounded challenges to oracle inputs, commitment validity, duplicate priority or arithmetic are accepted.

FINAL: winner PoI reward settles; every participant's pair/horizon reputation bucket updates.

35. Pseudocode: PoI batch

```
function score_price_range_target(T):
    claims = eligible_revealed_claims(T)
    P      = resolve_reference_price(T.oracle_config)
    B      = T.baseline_range

    claims = canonical_sort(claims)
    pon    = deterministic_priority_novelty(claims, T)

    best = null

    for c in claims:
        width = (c.upper - c.lower) / T.P0
        hit   = (c.lower <= P && P <= c.upper)

        precision = min(T.precision_cap,
                        B / max(width, T.width_floor))

        cal_gate = calibration_gate(
            provider=c.provider,
            pair=T.pair,
            horizon=T.horizon,
            tau=T.calibration_tau,
            prior=T.calibration_prior)

        eff_conf = c.confidence * cal_gate

        conf_factor = 1 + T.k_conf *
            (eff_conf - T.conf_min) /
            (T.conf_max - T.conf_min)

        horizon_factor = min(
            T.horizon_cap,
            pow_fixed(T.h_ref / T.horizon, T.beta_h))

        early = clamp(
            (T.cutoff - c.commit_time) /
            (T.cutoff - T.submission_open),
            0, 1)

        speed_factor = 1 + T.k_speed * early

        score = hit * precision * conf_factor *
                horizon_factor * speed_factor * pon[c.id]

        rep_update = reputation_evidence(
            c, hit, P, precision,
            horizon_factor, speed_factor, eff_conf)

        persist_score(c.id, score, rep_update)

        if better(score, c, best):
            best = c

    return best
```

better() compares CandidateScore first, then earliest commitment, then narrower canonical range, then higher EffectiveConfidence, then lexicographic claim hash. Every arithmetic operation uses fixed-point protocol rules.

36. Reward pseudocode

```
function poi_rewards(epoch e):
    pool = halving_emission(e) * rho_poi
    winners = []

    for target T finalized in e:
        w = finalized_winner(T)
        if w != null && w.score > 0:
            winners.append((T, w))

    denominator = sum(
        target_weight(T) * w.score
        for (T, w) in winners)

    if denominator == 0:
        expire_unminted(pool)
        return

    for (T, w) in winners:
        reward = pool *
                 target_weight(T) * w.score /
                 denominator
        pay(w.provider, reward)
```

Only one provider per contest earns PoI emission. All other resolved claims update reputation but receive zero protocol issuance for that target.

37. What earns PoI and what does not

PoI V1 is intentionally narrow.

PoI-eligible: a PRICE_RANGE_V1 claim for a registered pair/target/horizon, committed before cutoff, uniquely eligible under the one-claim rule, successfully revealed, correctly resolved inside its interval, and selected as the deterministic highest-scoring winner.

Reputation-only: valid but non-winning claims. They earn no protocol emission but positively or negatively update the IP's pair/horizon forecasting reputation.

Marketplace-only: other useful intelligence, research, data or agent services that do not map to PRICE_RANGE_V1. They may earn query fees but cannot mint PoI rewards in V1.

This boundary makes the emission function mechanically auditable and prevents subjective claims of intelligence from creating LIVE.

38. Novelty relative to existing systems

The individual components of LIVE are not new in isolation.

MCP/A2A: provide agent interoperability and capability access [3][4]. They do not define outcome-measured intelligence emissions.

Numerai: demonstrates token-staked forecasts, target-based scoring, information coefficient and unique contribution metrics such as MMC/MPC [5][6][7]. Numerai is a highly relevant precedent. LIVE differs by proposing an open blockchain-level intelligence marketplace, arbitrary provider-owned private data/MCP capabilities, multi-domain target registry, on-chain commitments and network-wide agent consumption rather than a single hedge-fund tournament.

Bittensor: demonstrates blockchain emissions for AI services and distributed validator evaluation [11]. LIVE differs by making future-resolving objective targets and deterministic scoring the default judge for PoI-eligible intelligence, reducing dependence on validator opinion.

Chainlink: demonstrates decentralised oracle/data delivery [10]. LIVE treats oracle data as one input to target resolution and focuses on measuring marginal intelligence contribution to agent decisions.

Shapley contribution methods: are established cooperative-game tools and are being applied to ensemble forecast contribution [15]. LIVE's proposal is to make deterministic approximate ensemble contribution part of an open token emission protocol.

Based on the systems reviewed for this paper, the **combination** of federated private intelligence capabilities, committed future forecasts, objective target resolution, deterministic marginal-contribution scoring, network reputation, agent micropayments and blockchain emissions appears distinct. This is not a claim that no prior art exists, and a formal novelty/patent analysis is outside this paper.

39. Why not simply use Bittensor?

A LiveAgents intelligence subnet could potentially be built on an existing network, and that is a valid bootstrap experiment. A dedicated LIVE chain becomes justified only if the protocol requires first-class primitives that are awkward as an application:

- target registry and long-duration claim state;
- privacy-aware commit/decrypt/reveal flows;
- deterministic outcome-based PoI scoring;
- high-volume LIVE micropayment channels tied to MCP capabilities;
- intelligence-specific reputation roots;
- chain-level issuance explicitly split between security and measured intelligence contribution;
- specialised challenge/proof verification.

A dedicated chain is therefore a design choice, not a prerequisite for validating the PoI thesis. The rational development path is to prove the scoring/economic mechanism before maximising protocol sovereignty.

40. Governance

Governance can alter software and bounded protocol parameters, but should not manually decide whether a particular provider was "intelligent" in a resolved epoch.

Governable parameters may include:

- target-family admission;
- oracle sets and source policies;
- reward split ranges;
- minimum bonds;
- challenge periods;
- Shapley sample count bounds;
- similarity thresholds;
- confidence levels;
- score-program upgrades;
- validator-set parameters.

For security, monetary and scoring parameters should have hard-coded or constitutional bounds. Ordinary governance should not be able to exceed the maximum supply, retroactively change a locked epoch's scoring rule, rewrite commitment priority, purchase reputation, or mint discretionary LIVE outside the halving schedule.

41. Privacy

LIVE supports privacy at several layers.

Source-data privacy: private DB/model never leaves provider.

Query privacy: direct MCP queries can remain off-chain; payment channels reveal only settlement state unless parties disclose query hashes.

Forecast privacy before resolution: salted commitment or timelock encryption.

Forecast privacy after resolution: future zk scoring can prove score from committed artifact without publishing the artifact.

Agent strategy privacy: the graph need not receive an agent's full model. Agents can submit only target/outcome evidence or use private reputation queries.

Privacy is not absolute: timing, payment relationships and public capability usage can leak metadata. Implementations should offer relays/batching where threat models justify it.

42. Economic attacks and countermeasures

42.1 Reward farming with low-value targets

If governance permits trivial targets with large emission weight, attackers can farm predictable outcomes. Target-family weights and admission must reflect economic relevance and uncertainty. A target whose baseline is already near-perfect should produce little marginal value.

42.2 Correlation masquerading as causation

PoI measures predictive/ensemble contribution to a registered target. It does not prove causal influence on markets or user P&L. The whitepaper uses "contribution" in the scoring/ensemble sense unless explicitly stated otherwise.

42.3 Stake plutocracy

Security stake and intelligence quality are separated. A large stake can make an entity a powerful validator if delegated/selected, but does not directly increase LIC.

42.4 Thin-market token manipulation

Provider pricing in LIVE can be volatile if LIVE/USD liquidity is poor. Capability manifests should be able to quote in a stable reference unit with LIVE settlement conversion through a protocol-approved price oracle, while keeping LIVE as settlement/stake asset.

43. Performance and scaling

LiveChain's workload is highly parallel: one agent buying BTC intelligence need not block another agent buying weather, fraud or logistics intelligence. The protocol should therefore avoid global ordering where causality does not require it.

V1 can prioritise correctness with sequential BFT blocks plus off-chain channels. V2+ can migrate independent events toward DAG-based admission and parallel execution, with deterministic conflict detection and BFT finality checkpoints. This preserves economic semantics while increasing machine-to-machine throughput.

PRICE_RANGE_V1 makes canonical PoI scoring computationally small: each claim contains two price bounds, a confidence value, timestamps and fixed metadata. The expensive network paths are market

discovery, oracle ingestion and high-frequency agent commerce rather than high-dimensional forecast-vector scoring.

Expected on-chain state growth per claim can be kept to hundreds of bytes:

- contributor/target IDs;
- commitment/root;
- timestamps/status;
- content hash;
- bond;
- score/reward leaf or root reference.

A million intelligence *queries* per month need not mean a million on-chain transactions if payment channels are used. The chain settles channels, provider registrations, claims and PoI epochs; MCP carries the actual intelligence traffic.

44. Implementation phases

Phase 0 - scoring research

Before mainnet:

- build target registry prototype;
- run public financial forecasting challenges;
- compare residual IC, leave-one-out and Monte Carlo Shapley contribution;
- attack similarity compression with sybil variants;
- measure score stability and false discovery rates;
- publish full benchmark code/data.

Phase 1 - testnet

- Cosmos SDK/CometBFT chain;
- LIVE test token;
- `x/targets`, `x/poi`, `x/capability`, `x/settlement`;
- public commit-reveal;
- evaluator quorum scoring;
- MCP provider registry;
- payment channels;
- PRICE_RANGE_V1 target families for BTCUSDT, ETHUSDT and SOLUSDT across registered 1m/5m/15m/1h horizons;

provider staking and sell-capacity accounting;

request/quote and broadcast market;

contextual reputation-gated router;

Proof of Novelty exact/canonical/semantic fingerprint pipeline;

deterministic halving emission schedule.

Phase 2 - confidential intelligence

- threshold/timelock encrypted claims;

- richer target/oracle adapters;
- agent-to-agent capability publishing;
- contextual reputation graph roots;
- high-volume settlement tooling.

Phase 3 - verifiable scoring

- zkVM score proofs;
- permissionless scorer/prover market;
- score-proof aggregation;
- stronger privacy for post-resolution forecast artifacts.

Phase 4 - network intelligence

- automated source-combination discovery;
- value-of-information routing;
- cross-domain target markets;
- agent teams that autonomously purchase intelligence under budgets;
- graph-derived intelligence capabilities.

Phase 5 - high-concurrency LiveChain

DAG-based event admission/parallel execution where conflict-free;

BFT finality checkpoints over canonical state;

high-volume auction/broadcast markets;

proof-carrying router/reputation snapshots;

benchmark whether DAG complexity is justified by observed workload.

45. Falsifiable hypotheses

LIVE should be treated as a protocol hypothesis. It is successful only if experiments support claims such as:

1. Outcome-based marginal-contribution scoring predicts future utility better than simple accuracy leaderboards.
2. Data owners will expose paid intelligence capabilities without surrendering raw data.
3. Agents will pay for external intelligence when empirically estimated VOI exceeds price.
4. Similarity compression plus marginal contribution materially reduces copy/sybil reward extraction.
5. The network graph improves source routing for newly created agents.
6. PoI emissions attract genuinely incremental signals rather than metric-gaming noise.

Higher pair/horizon reputation predicts lower future interval-score loss out of sample.

Providers with sustained high claimed confidence remain empirically calibrated rather than merely overstating confidence.

Earlier correct commitments contain measurable value beyond otherwise similar later predictions.

Winner-only rewards plus miss-weighted reputation resist range-spam and extreme-confidence gaming.

7. Private/committed claims can be scored with acceptable latency and verification cost.

If these do not hold, launching a dedicated chain would not be justified by this thesis.

46. Open research questions

Several issues remain genuinely hard.

- Can near-duplicate signals be clustered robustly without suppressing independent corroboration?
- How should Shapley approximation scale to tens of thousands of simultaneous contributors?
- Which ensemble aggregator produces the best incentives for probabilistic versus rank signals?
- How should target weights avoid governance capture and benchmark gaming?
- Can a zkVM score large vector workloads cheaply enough for frequent epochs?
- What is the best privacy/availability compromise for proprietary forecasts?
- How should agent-specific value of information be estimated without exposing agent strategy?
- How should correlated target families be accounted for when allocating emission?
- How should protocol rewards distinguish ephemeral edge from durable intelligence?
- Can network-level learning compound without collapsing strategy diversity into groupthink?

How robustly can semantic and behavioural novelty detection distinguish independent corroboration from disguised copying?

Should descendant intelligence pay ancestry royalties, and how can this avoid royalty-chain spam?

What provider-capacity curve best balances capital efficiency, new-provider access and systemic risk?

At what sustained throughput does DAG-based execution outperform a simpler BFT chain after operational complexity is included?

What fixed or constitutionally bounded consumption-burn rate best balances network affordability with long-run scarcity?

What horizon-factor exponent best rewards short-horizon precision without making micro-horizon noise dominate emissions?

What speed multiplier best values early foresight without overpowering precision and calibration?

Should near-duplicate range overlap receive a smooth PoN discount or only exact canonical duplicates be priority-gated?

Which multi-venue TWAP construction is most manipulation-resistant for thin or fragmented pairs?

47. Conclusion

Proof of Work rewards expenditure of computation. Proof of Stake secures consensus with economic collateral. LIVE proposes a different reward question:

> ***What did the network learn that it did not already know, and can that contribution be demonstrated against a future outcome?***

A sound implementation cannot answer that question with slogans, validator taste or an LLM grading another LLM. It requires precommitted targets, concealed forecasts, objective resolution, deterministic scoring, statistical maturation, marginal-contribution accounting and cryptographic settlement.

The resulting design is intentionally hybrid:

BFT finality secures canonical balances, stake, commitments and settlement;

a DAG execution path can increase concurrency as agent volume grows;

commit-time priority and deterministic range equivalence prevent trivial repackaging from being treated as new price intelligence;

PoI rewards only the highest-scoring correct price-range prediction in each registered contest from a finite, halving emission pool;

MCP/A2A move intelligence between agents and Intelligence Providers;

the Intelligence Router matches agents to contextually useful providers;

LIVE settles payments, bonds and rewards, while provider staking gates selling capacity;

the Live Intelligence Graph compounds evidence about what intelligence works, when, for whom, at what price and with what ancestry.

The strategic thesis is simple:

> ***Models learn from their own experience. LIVE lets agents learn from the verified experience of the network.***

Unlimited intelligence. Finite LIVE. Evidence earns reputation. Usage creates scarcity.

If the protocol succeeds, LiveChain becomes a machine-verifiable market for forecasting intelligence. An IP proves value by committing a price range, confidence and timing before the market outcome is known. The network rewards the correct provider that is most precise, credibly confident and early, while every miss becomes durable reputation evidence. Intelligence can grow without limit; LIVE cannot.

References

[1] Cosmos SDK Documentation, “Introduction to Blockchains / Blockchain Basics.” Deterministic replicated state machines and CometBFT BFT overview. https://docs.cosmos.network/sdk/v0.53/learn/intro/blockchain-basics

[2] Cosmos SDK Documentation, “Protobuf and Signing / Deterministic Encoding.” https://docs.cosmos.network/sdk/latest/learn/concepts/encoding

[3] Model Context Protocol maintainers, “The 2026-07-28 Specification,” 28 July 2026. Stateless core, routing and authorisation updates. https://blog.modelcontextprotocol.io/posts/2026-07-28/

[4] Agent2Agent Protocol, official specification; Linux Foundation project. https://a2a-protocol.org/v1.0.0/

[5] Numerai Docs, “Meta Model Contribution (MMC).” Neutralisation to the meta-model and covariance with target. https://docs.numer.ai/numerai-tournament/scoring/meta-model-contribution-mmc

[6] Numerai Docs, “Numerai Signals Scoring.” Information coefficient, residual information coefficient and contribution metrics. https://docs.numer.ai/numerai-signals/scoring

[7] Numerai Docs, “Getting Started / Staking.” Staked predictions, rewards and burns. https://docs.numer.ai/

[8] Tilmann Gneiting and Adrian E. Raftery, “Strictly Proper Scoring Rules, Prediction, and Estimation,” Journal of the American Statistical Association, 102(477), 2007, pp. 359-378. https://sites.stat.washington.edu/people/raftery/Research/PDF/Gneiting2007jasa.pdf

[9] Cosmos SDK Documentation, `x/staking` and module architecture. https://docs.cosmos.network/sdk/latest/modules/staking/README and https://docs.cosmos.network/sdk/latest/learn/concepts/modules

[10] Chainlink Developer Documentation, Data Feeds / Data Streams / verifiable data services. https://docs.chain.link/

[11] Bittensor Documentation, “How Yuma Consensus 3 Makes Bittensor More Fair.” Validator evaluation, bonds and emissions. https://docs.learnbittensor.org/yc3-blog

[12] S.T. Liu, J. Yu, J. Steeves, “Solving the Free-rider Problem in Bittensor,” ACM CCS 2024 poster / OpenTensor Foundation. https://docs.learnbittensor.org/papers/ACM_CCS2024_Poster.pdf

[13] drand Documentation, “Cryptography.” Threshold BLS, DKG and publicly verifiable randomness. https://github.com/drand/drand-docs/blob/master/docs/concepts/01-Cryptography.md

[14] drand Documentation, “Timelock Encryption.” https://docs.drand.love/docs/timelock-encryption/

[15] Minsu Kim, Evan L. Ray, Nicholas G. Reich, “Beyond forecast leaderboards: Measuring individual model importance based on contribution to ensemble accuracy,” International Journal of Forecasting, 42(3), 2026, pp. 924-936. DOI: 10.1016/j.ijforecast.2025.12.006

[16] EigenLayer Whitepaper, discussion of slashing and unintended slashing risk. https://docs.eigencloud.xyz/assets/files/EigenLayer_WhitePaper-88c47923ca0319870c611decd6e562ad.pdf

[17] Linux Foundation, “OpenSharing Project,” 10 June 2026. Standardisation of cross-platform AI asset and data exchange. https://www.linuxfoundation.org/press/linux-foundation-announces-opensharing-project-to-standardize-ai-asset-and-data-exchange

Appendix A - Example target descriptor

```
{
  "target_id": "BTCUSDT_5M_2026-09-18T12:00:00Z_v1",
  "claim_type": "PRICE_RANGE_V1",
  "pair": "BTCUSDT",
  "target_time": "2026-09-18T12:00:00Z",
  "horizon_seconds": 300,
  "submission_open": "2026-09-18T11:50:00Z",
  "submission_cutoff": "2026-09-18T11:55:00Z",
  "price_at_open_atoms": "11500000000000",
  "confidence_min_ppm": 500000,
  "confidence_max_ppm": 990000,
  "oracle": {
    "type": "MEDIAN_OF_VENUE_TWAP",
    "venues": ["venue_a", "venue_b", "venue_c"],
    "twap_seconds": 30,
    "min_valid_venues": 2
  },
  "baseline_range": {
    "method": "EWMA_VOL_SQRT_TIME_V1",
    "width_ppm": 4200,
    "snapshot_time": "2026-09-18T11:50:00Z"
  },
  "scoring": {
    "precision_cap_ppm": 4000000,
    "horizon_cap_ppm": 4000000,
    "k_conf_ppm": 1000000,
    "k_speed_ppm": 1000000,
    "calibration_tau_ppm": 100000,
    "range_iou_duplicate_ppm": 950000,
    "confidence_duplicate_delta_ppm": 10000,
    "fixed_point_scale": 1000000
  },
  "score_program_hash": "sha256:..."
}
```

Appendix B - Example capability manifest

```
{
  "capability_id": "whale_flow_intelligence/v3",
  "provider_key": "live1...",
  "transport": "MCP",
  "endpoint": "https://provider.example/mcp",
  "privacy": "SOVEREIGN",
  "domains": ["crypto"],
  "assets": ["BTC", "ETH", "SOL"],
  "query_types": ["flow_change", "accumulation", "distribution"],
  "freshness_seconds": 30,
  "expected_latency_ms": 80,
  "price": {
    "asset": "LIVE",
    "model": "PER_QUERY",
    "amount_atoms": "5000000"
  },
  "response_signing_key": "...",
  "schema_hash": "sha256:...",
```

```
  "reputation_root": "0x..."
}
```

Appendix C - Example signed intelligence response

```
{
  "claim_id": "0x...",
  "provider_key": "live1...",
  "target_id": "BTCUSDT_5M_2026-09-18T12:00:00Z_v1",
  "pair": "BTCUSDT",
  "lower_price_atoms": "11498000000000",
  "upper_price_atoms": "11504000000000",
  "confidence_ppm": 950000,
  "commit_time_ms": 1789732212123,
  "commitment": "sha256:...",
  "provider_signature": "...",
  "resolution": {
    "reference_price_atoms": "11501200000000",
    "hit": true,
    "precision_factor_ppm": 3100000,
    "confidence_factor_ppm": 1780000,
    "horizon_factor_ppm": 2200000,
    "speed_factor_ppm": 1850000,
    "pon_ppm": 1000000,
    "candidate_score_atoms": "22491700000000"
  }
}
```

Appendix D - Forecasting reputation versus PoI emission

PoI emission and forecasting reputation are intentionally distinct. PoI is winner-only for each PRICE_RANGE_V1 contest. Reputation updates for every valid resolved claim.

A provider can therefore lose a contest while still improve reputation with a strong correct prediction that was narrowly beaten, or damage reputation with an overconfident miss even if it had previously won many contests.

The Intelligence Router SHOULD rank providers using contextual reputation, calibration, evidence mass, freshness, price and latency. It SHOULD NOT route purely on past PoI winnings.

This separation reduces winner-take-all path dependence: emissions reward the best resolved intelligence, while the market retains a richer statistical view of every provider's forecasting quality.