Zenon Indigopaper Series

Incentives and Coordination in a Verification-First Network

Series Context

The Zenon Greenpaper defined the architectural foundations of a verification-first network — a system where correctness is proven within declared resource bounds rather than assumed from consensus or availability. Its key insight, that "refusal is correctness-preserving," made it possible for even limited devices to participate in trustless computation.

The Purplepaper extended this foundation into economics: when correctness is bounded and proofs become scarce, verification itself gains market value. Proofs are not just technical artifacts — they become tradable, measurable resources.

This Indigopaper completes the trilogy. It describes how incentives and coordination arise naturally from those foundations, forming a self-reinforcing ecosystem that remains decentralized, efficient, and correctness-preserving.

The trilogy closes a conceptual loop:

Architecture defines possibility → Economics defines behavior → Coordination defines reality.

Abstract

The verification-first model treats correctness as a bounded resource. Each verifier operates within explicit limits $R_V = (S_V, B_V, C_V)$ — representing storage, bandwidth, and computation capacity. When verification cannot complete within these bounds, refusal is the only correct outcome — not a failure.

The Greenpaper formalized these limits; the Purplepaper explored their market effects. This Indigopaper explains how economic coordination and specialization emerge from those same limits.

We introduce an ecosystem of actors — verifiers, relays, sentries, sentinels, supervisors, pillars, and ambient relay nodes (ARNs) — that transform refusals into measurable demand signals. These signals drive caching, replication, and optimization without central authority.

By treating absence as information, the network evolves toward higher availability while preserving local correctness. Correctness remains local, incentives remain emergent, and trust remains unnecessary.

1 Reader Contract

This paper assumes the reader accepts four foundational premises from the Greenpaper:

1. **Bounded Verification** — Each verifier declares limits $R_V = (S_V, B_V, C_V)$ representing storage, bandwidth, and computation resources.
2. **Refusal Is Correctness-Preserving** — If a verifier cannot complete proof evaluation within its limits, the correct action is refusal, not approximation.
3. **Validity Is Source-Agnostic** — The validity of a proof depends only on cryptography, never on who provides it.
4. **Ordering $\neq$ Availability** — Consensus guarantees canonical order, but not the availability or propagation of proofs.

From these premises follow two corollaries:

- **Local Truth Domains:** No participant can assert correctness on behalf of another; each verifier enforces its own bounded truth.
- **Optional Coordination:** Global coordination arises for efficiency, not safety. The network remains correct even in partial isolation.

Refusal is therefore not a failure condition but a measurable, economically meaningful signal. Every refusal both protects correctness and indicates where verification demand exceeds supply. (G2.7, P2.3)

2 The Coordination Problem

2.1 Motivation

In a verification-first architecture, correctness can be proven locally but only within each verifier's declared resource limits. Ordering consensus, such as the Momentum chain, ensures a shared sequence of commitments but cannot guarantee that all participants possess or can verify every proof within that sequence.

The core challenge is therefore:

How can bounded verifiers acquire the proofs they need efficiently, without trusting distributors or weakening refusal semantics?

(G2.3, P7)

2.2 Refusal as a Signal

When a verifier cannot complete verification, it issues a Refusal Witness:

$$w_R = (\text{last\_header}, \text{object\_id}, \text{refusal\_code}, \text{bound\_dimension}, \text{metadata})$$

This witness is not a declaration of invalidity — it's a signed statement that the verifier reached its resource boundary. Other nodes can interpret these witnesses as structured demand indicators:

- DATA_UNAVAILABLE → incentive for relays to cache missing proofs.
- OUT_OF_SCOPE → incentive for archival providers to restore older proofs.
- COST_EXCEEDED → incentive for compression or aggregation specialists.

(P2.3.2)

2.3 From Failure to Feedback

In execution-first blockchains, missing data represents failure. In verification-first systems, it represents opportunity — a visible gradient of unmet verification.

Each refusal contributes to a distributed economic field:

- Clusters of similar refusals reveal scarcity in bandwidth, compute, or storage.
- Relays and supervisors interpret these clusters as signals for caching and bounty placement.
- The network's collective behavior adapts to minimize future refusals.

Thus, refusal becomes a feedback mechanism — transforming local incapacity into global optimization. (G2.7.1, P4–P7)

2.4 Formal Statement

Let each verifier V_i operate under resource bounds $R_i = (S_i, B_i, C_i)$. Let A be the set of proof artifacts required for a given ordered commitment. Define $D_i \subseteq A$ as the subset accessible to V_i.

Verification succeeds if:

$$|A - D_i| = 0 \quad \text{and} \quad \text{Cost}_V(A) \leq R_i$$

Otherwise, V_i emits w_R.

The coordination goal is to minimize the global refusal probability:

$$\min \sum_i \mathbb{P}(w_R^i)$$

while preserving local correctness and zero-trust assumptions. (G2.5–G2.7)

2.5 Preview of Resolution

The rest of this paper shows how refusals propagate through a network of specialized actors:

1. Verifiers emit refusals.
2. Relays interpret them as demand.
3. Sentries bundle and optimize proofs.
4. Sentinels monitor for misbehavior.
5. Supervisors aggregate refusals and route incentives.
6. Ambient Relay Nodes (ARNs) move data opportunistically.
7. Pillars maintain global order, unaffected by local failures.

The result is asymptotic availability: verification becomes faster and more local over time, without ever abandoning bounded correctness. (P6, P9)

3 Actor Summary

Overview

Bounded verification divides the network into specialized roles. Each actor type focuses on a stage of the verification supply chain, motivated by incentives that emerge from scarcity and refusal. None are trusted; all outputs remain verifiable.

Role	Function	Incentive	Outputs	Trust Model
Verifier [V]	Performs bounded verification	Minimize cost per verified truth	Verified / Refused results; refusal witnesses	Self-verifying proofs
Relay [R]	Caches and delivers proofs near demand	Profit from refusal density	Proof bundles and descriptors	Clients re-verify
Sentry [S]	Packages and optimizes proof bundles	Fees, responsiveness	Bundle catalogs and optimized proofs	Source-agnostic proofs
Sentinel [N]	Detects misbehavior and publishes evidence	Bounties and reputation	Verifiable watch reports	Evidence must verify
Pillar [P]	Maintains ordering consensus	Block rewards	Ordered commitments and finality proofs	Protocol consensus
Supervisor [U]	Aggregates demand and routes incentives	Coordination fees	Hotset and bounty announcements	Verifiable reputation ledger
ARN [A]	Opportunistic edge caching	Micro-rewards or altruism	Cached proof forwarding	Local verification only

These roles exist only because verification is bounded. If every node could verify all proofs directly, no coordination or market would be necessary. The bounded model instead forces specialization and economic feedback, transforming scarcity into efficiency.

3.1 End Verifiers

Verifiers are the atomic units of truth. Each declares its resource limits $R_V = (S_V, B_V, C_V)$ and must refuse when those limits are exceeded. Their objective is to minimize cost per verified claim while preserving correctness.

Verifiers produce two message types:

- **Verification Results:** definite TRUE or FALSE outcomes.
- **Refusal Witnesses:** structured denials with refusal codes.

Because refusal is safe and verifiable, verifiers cannot be coerced into guessing. Their reliability becomes measurable by their verified-to-refused ratio — a direct indicator of efficiency. (G2.5, P2.3)

3.2 Relays

Relays interpret refusals as economic signals. A concentration of DATA_UNAVAILABLE refusals in a region implies missing proofs. Relays profit by caching and serving those proofs close to where demand originates.

Formally, let ρ represent local refusal density. Relay revenue potential P_R increases with $\partial\rho/\partial t$ — the rate of new refusals relative to their resolution. Efficient relays reduce future refusals and receive micropayments or reputation credits in proportion to their effectiveness.

Relays transport verifiable data only; they never assert validity. Clients always re-verify received proofs. (P6.3)

3.3 Sentries

Sentries are persistent nodes that optimize and advertise proof bundles. Their advantage lies in latency and bandwidth optimization rather than authority. Typical functions include:

- Aggregating small proofs into larger bundles.
- Maintaining catalogs of available bundles and pricing.
- Applying compression to reduce transmission cost.

All bundles remain verifiable; dishonest sentries gain nothing. Revenue depends on successful bundle deliveries confirmed by verifier receipts. (P6.3)

3.4 Sentinels

Sentinels function as the network's immune system. They detect and publish evidence of strategic withholding, duplicate packaging, or false availability claims.

A valid sentinel report must contain:

1. The challenged claim or bundle descriptor.
2. Proof or evidence of absence.
3. Cryptographically signed discrepancies.

False or unverifiable reports damage the sentinel's reputation automatically. (G2.3.5)

3.5 Pillars

Pillars maintain the ordering plane, finalizing commitments without depending on verification success below them. They earn rewards for valid inclusion and are penalized for misordering or equivocation. Separation of ordering and verification ensures temporary unavailability cannot corrupt consensus finality. (G2.4)

3.6 Supervisors

Supervisors are optional overlay participants that aggregate refusals into hotsets and announce bounties to incentivize proof replication. A minimal supervisor ledger includes:

- Hotset identifiers and metadata.
- Associated bounty pools.
- Signed fulfillment receipts.
- Optional participant reputation scores.

Supervisors route incentives but never assert correctness. Their outputs are ignorable without affecting safety. (P6.3, P8.2)

3.7 Incentive Interactions

Each actor's output drives another's incentive:

- Verifiers → emit refusals.
- Relays → convert refusals into caching opportunities.
- Sentries → optimize proof distribution.
- Sentinels → ensure honesty and accountability.
- Supervisors → align demand and supply through bounties.
- Pillars → anchor order without trust.
- ARNs → extend reach opportunistically.

This loop sustains itself without central coordination: every refusal is both a signal of cost and an opportunity for reward. (P7.2, P8.1)

3.8 Stability and Competition

Competition among actors promotes stability because success depends on verifiable outcomes.

- Invalid relays are discovered immediately.
- Inefficient sentries lose clients.
- Mispriced supervisors lose participants.

Efficiency thus aligns with network health. Equilibrium emerges when refusals persist only at the network's capacity frontier — continuously identifying the next point of optimization. (G2.10, P5–P6)

3.9 Quantitative Outlook

This version defines incentives qualitatively; quantitative modeling is deferred. Open research areas include:

- **Bounty Pricing Function** — relation between refusal density ρ and reward magnitude.
- **Convergence Rate** — rate at which $\rho(t)$ decreases with incentive intensity $I(t)$.
- **Elasticity Constant** k — from the dynamic model $\frac{d\rho}{dt} = -kI(t)$.

These variables will be empirically characterized in subsequent work. (See §8.6)

4 The Verification Supply Chain

The verification supply chain describes how bounded verifiers, economic actors, and overlay systems interact to transform refusals into restored availability. It is not a rigid protocol but an emergent process governed by measurable demand gradients. (G2.10, P6–P7)

4.1 Overview

Verification within a bounded environment follows a recurring cycle:

1. **Attempt:** A verifier tests a claim within its limits.
2. **Refusal:** If limits are exceeded, it emits a Refusal Witness w_R.
3. **Propagation:** Witnesses spread through gossip or relay channels.
4. **Interpretation:** Relays, sentries, and supervisors interpret refusals as demand.
5. **Fulfillment:** Proof bundles are retrieved, packaged, or relocated closer to demand.

6. **Resolution:** The verifier retries and completes verification locally.
7. **Stabilization:** Reduced refusals indicate restored balance.

In this model, absence becomes a coordination signal. No actor promises data existence; all act on verifiable gaps in coverage.

4.2 Refusal Witnesses as Economic Beacons

Each refusal witness contains structured metadata suitable for aggregation:

```
message RefusalWitness {
  bytes last_header_hash = 1;
  bytes object_id = 2;
  uint32 refusal_code = 3;      // DATA_UNAVAILABLE, OUT_OF_SCOPE, COST_EXCEEDED
  uint32 bound_dimension = 4;   // storage, bandwidth, compute
  string metadata = 5;
  bytes signature = 6;
}
```

Analytical nodes or supervisors can aggregate witnesses to locate bottlenecks:

- Repeated DATA_UNAVAILABLE entries → missing data propagation.
- Frequent COST_EXCEEDED codes → proofs too large or inefficient.

The resulting metrics form a live economic heat map of verification stress. (G2.7.2, P2.3.2)

4.3 Relay and Sentry Fulfillment Cycle

Once demand is visible, relays and sentries act to resolve it:

1. **Relay Discovery:** Relays collect refusal data.
2. **Proof Retrieval:** They fetch missing artifacts from archives or peers.
3. **Bundle Formation:** Sentries compress and assemble proof bundles.
4. **Announcement:** Sentries broadcast a BundleDescriptor advertising contents and cost.
5. **Delivery:** Verifiers redeem descriptors, download bundles, and verify locally.
6. **Receipt:** Successful verifications produce signed Fulfillment Receipts.

```
message BundleDescriptor {
  bytes root_hash = 1;
```

```
  repeated bytes object_ids = 2;
  uint64 size_bytes = 3;
  uint64 approx_verify_cost = 4;
  bytes provider_signature = 5;
}
```

Fulfillment receipts confirm successful service and enable micropayments or reputation updates. (P6.3)

4.4 Supervisor Aggregation and Hotsets

Supervisors aggregate refusal witnesses into hotsets — clusters of high-demand proofs requiring replication.

```
message HotsetAnnouncement {
  uint64 epoch = 1;
  repeated BundleDescriptor hot_items = 2;
  repeated Bounty bounties = 3;
  bytes supervisor_signature = 4;
}
```

Hotsets focus economic activity where the marginal reduction in refusal probability is highest. This market-driven allocation ensures that resources flow toward areas of greatest verification scarcity. (P6.3, P8.2)

4.5 Sentinel Oversight

Sentinels monitor for two main adversarial behaviors:

- **Withholding:** claiming to provide proofs but failing to deliver.
- **Duplication Fraud:** repackaging existing proofs without new value.

Detection depends on cross-verifiable audit trails. A valid sentinel report includes the offending bundle descriptor, signed observations, and cryptographic evidence of inconsistency. Accurate reports earn bounties; false reports self-penalize through wasted resources or lost reputation. (G2.3.5, P8.2)

4.6 Demand–Supply Feedback Dynamics

The global refusal rate $R(t)$ decreases as incentives $I(t)$ motivate proof replication:

$$\frac{dR(t)}{dt} = -k \cdot I(t)$$

where k is a network elasticity constant derived from empirical data. This feedback loop yields self-correcting availability: as demand spikes, incentives rise; as scarcity resolves, incentives naturally taper. (P7.2)

4.7 Temporal Specialization

Over time, specialization emerges:

- Short-term relays → recent, high-traffic proofs.
- Long-term archives → historical or out-of-scope proofs.
- Compression specialists → cost-intensive proofs.
- Reputation brokers → verify relay efficiency via receipts.

This specialization increases systemic throughput and reduces redundant work. Availability scales economically, not bureaucratically.

4.8 Emergent Properties

1. **Decentralized Availability:** No scheduler; actors follow public signals.
2. **Safety by Construction:** All data remains verifiable; no correctness delegation.
3. **Economic Efficiency:** Refusal gradients direct effort automatically.
4. **Continuous Adaptation:** Resources migrate dynamically toward demand.
5. **Bounded Correctness:** No verifier exceeds its declared limits.

These properties realize the Greenpaper's goal: a network where correctness is guaranteed and availability is emergent through incentive alignment. (G2.10, G2.12)

5 Ambient Relay Nodes (ARNs)

Ambient Relay Nodes (ARNs) extend the verification supply chain to the network edge. They are lightweight, proximity-based participants that opportunistically forward proofs using idle resources. ARNs increase proof availability without introducing trust or altering verification semantics — functioning like a mesh layer for verifiable data. (P11)

5.1 Definition

An Ambient Relay Node A_j satisfies:

1. Maintains bounded cache for recently requested proofs.
2. Participates in gossip or lightweight discovery channels.
3. Forwards bundle descriptors or proofs opportunistically.
4. Never asserts correctness; recipients must always verify.

Forwarding is permitted only when:

$$\text{Cost}_A(\text{forward}) < R_j - \text{Reserve}_j$$

ensuring that forwarding never interferes with local verification duties. Each ARN thus contributes availability while staying within safe resource limits.

5.2 Functional Role

ARNs behave like ad hoc relays in a content delivery mesh:

- Proximity forwarding: advertise cached bundles on local subnets.
- Opportunistic fetch: retrieve missing proofs when idle bandwidth allows.
- Ephemeral caching: discard data automatically as limits are reached.

They replicate popular proofs organically — not by coordination, but by response to observable demand.

5.3 Example Operation Flow

1. A verifier refuses due to DATA_UNAVAILABLE.
2. The refusal propagates locally; nearby ARNs learn which proof is missing.
3. An ARN with spare capacity fetches and caches the proof bundle.
4. When the verifier retries, it discovers the cached bundle through local gossip.
5. The verifier verifies the proof and emits a signed fulfillment receipt.

This mirrors the behavior of peer discovery systems (like AirDrop or mesh routing) but without identity, location, or trust semantics — only the propagation of verifiable data.

5.4 Incentive Design

ARNs can be altruistic, but small rewards encourage sustained participation:

- Micro-rewards: earned per verified delivery receipt.
- Reputation accrual: maintained in optional supervisor ledgers.
- Bounty participation: fractional claims for contributing to hotset replication.

Each ARN's contributions are proven by cryptographic receipts linking bundle hash, sender, and receiver. Smart contracts or supervisor logic can automate these payments. (P6.3, P8.2)

5.5 Security and Safety Properties

1. Source-agnostic validity: all proofs remain verifiable regardless of origin.
2. Bounded exposure: ARNs forward only within declared limits; spam is self-limiting.
3. Replay resistance: bundle hashes prevent tampering.
4. Anonymity preservation: no identity exchange required.
5. No correctness delegation: verification occurs only at the destination.

Even if ARNs act maliciously, the worst consequence is wasted bandwidth — never corrupted state. (G2.3)

5.6 Economic Dynamics

Let N_A be the number of active ARNs in a region and R the local refusal rate. Empirically, availability improves exponentially with ARN density:

$$R \propto e^{-\alpha N_A}$$

for some network constant $\alpha > 0$.

Implications:

- Latency reduction: proofs travel shorter distances.
- Localized markets: hotspots form transient micro-economies.
- Adaptive density: popular bundles replicate automatically.
- Self-limiting overhead: long-tail data remains unaffected.

As ARN participation grows, refusal rates fall sharply, then plateau — indicating diminishing but continuous returns to decentralization.

5.7 Integration with Core Actors

ARNs interact naturally with existing roles:

- Relays seed ARNs with high-demand bundles.
- Sentries broadcast bundle descriptors to nearby ARNs.
- Supervisors include ARN participation in bounty routing.

Unlike institutional relays, ARNs lack identity, persistence, or contractual obligations. They embody pure decentralization — any device can contribute availability at any time without altering correctness semantics.

5.8 Long-Term Implications

As ARNs proliferate, the network evolves into a self-repairing verification mesh. Every refusal becomes a local opportunity for resolution. Refusal rates never reach zero — bounded verification forbids that — but their probability decreases continuously with participation.

ARNs thus manifest the network's economic gravity: demand attracts supply until local equilibrium forms. Availability emerges naturally, without central control. (G2.12)

6 Light Client Transaction Lifecycle and Verification Consensus

A verification-first network must support light clients — devices with limited resources — without requiring them to trust intermediaries. This section describes how transactions move from creation to local verification, showing how bounded verification and incentive-driven coordination maintain both correctness and usability. (G1.4, P11)

6.1 Roles and Boundaries

Light clients operate under tight resource limits for storage, bandwidth, and computation. They author, broadcast, and verify their own transactions locally. Relays, sentries, and ARNs assist in data movement; pillars handle ordering; sentinels detect misbehavior; supervisors manage incentives. No intermediary is trusted — every proof remains independently verifiable.

6.2 Transaction Creation

A light client constructs a transaction as follows:

1. Validates inputs against its latest verified state frontier.
2. Signs the transaction cryptographically, binding it to its account chain.
3. Optionally includes local proofs supporting the claim.

Because correctness and ordering are separate, a client can create valid transactions offline. When connectivity resumes, it can broadcast them without risk of inconsistency.

6.3 Dissemination

Upon reconnection, the client transmits its transaction through one or more untrusted paths:

- Direct peer-to-peer relay.
- Sentry or relay endpoints.
- Opportunistic forwarding by ARNs.
- Delayed broadcast after rejoining the network.

Even if some paths fail, correctness is unaffected. Unavailability delays inclusion but never corrupts verification outcomes. (P11.1–P11.2)

6.4 Ordering and Finality

Pillars collect transactions and commit them into the canonical order. From the client's perspective, correctness requires verifying two facts:

1. The transaction hash appears in a finalized commitment.
2. The commitment itself satisfies consensus rules and quorum proofs.

Clients do not need to observe validator operations — only to verify inclusion proofs within bounded limits. (G2.4)

6.5 Inclusion Verification (Online)

When online, a light client monitors new headers referencing its transactions:

1. Detect inclusion of its transaction in a header.
2. Request the inclusion proof (e.g., Merkle branch).
3. Verify header signatures and consensus lineage.
4. Validate inclusion consistency against its last verified state.
5. Accept if verified; otherwise, issue a Refusal Witness.

Refusal here indicates temporary proof unavailability, not an invalid state. (P2.5)

6.6 Inclusion Verification (Offline)

If offline during finalization:

1. Retrieve the latest headers upon reconnection.
2. Download proof bundles bridging the state gap.
3. Verify headers incrementally within bounds.
4. Confirm inclusion once the anchor header is validated.

Offline operation increases cost but not risk. If verification exceeds local bounds, the client issues a refusal, triggering relays and ARNs to respond to the revealed demand. (P11.1)

6.7 Post-Inclusion State Verification

After confirming inclusion, the client validates the transaction's effects:

- Account and contract state transitions.
- Token or zApp-level balances.
- Associated cryptographic proofs (Merkle or ZK).

Outcomes are:

1. **Verified:** Proofs obtained and validated.
2. **Refused (DATA_UNAVAILABLE):** Proofs exist but are unreachable.
3. **Refused (COST_EXCEEDED / OUT_OF_SCOPE):** Proofs exceed capacity or retention.

Refusal remains a complete, correct outcome until resources or data availability change. (G2.5–G2.7)

6.8 Network Response to Refusal

When a client issues a refusal:

- Relays replicate missing proofs.
- Sentries aggregate and repackage them.
- Supervisors advertise bounties for fulfillment.
- ARNs distribute proofs opportunistically.
- Sentinels monitor for withholding or fraud.

This completes the demand–supply loop: refusal → signal → response → verification → new equilibrium. (P6–P8)

6.9 Layers of Consensus

Consensus in a verification-first system unfolds across three layers:

1. **Ordering Consensus:** Pillars finalize sequence.
2. **Verification Consensus:** Clients verify proofs locally.
3. **Economic Consensus:** Incentives align supply and demand for proofs.

Each layer reinforces the others, but none depend on trust. Consensus becomes a layered assurance model — technical, logical, and economic. (G2.4, P4.2)

6.10 Safety Under Adversarial Conditions

Even if:

- Pillars are honest but data is withheld,
- Relays provide inconsistent bundles, or
- Clients are offline for extended periods,

the system remains correct. The only possible outcome is refusal, which preserves safety and triggers corrective economic feedback.

6.11 Summary

Light clients demonstrate the verification-first principle in full:

- Correctness is never delegated.
- Availability is emergent and incentive-driven.
- Refusal is a legitimate, correctness-preserving state.

Online operation improves responsiveness; offline operation preserves safety. Through continuous feedback, the network converges toward both economic efficiency and epistemic soundness.

7 Narwhal and Tusk Coordination Overlays

Narwhal and Tusk are dissemination and ordering protocols originally designed to decouple data propagation from consensus. Within a verification-first network, they can operate as optional overlays — improving throughput and data visibility without altering the semantics of refusal or bounded verification. (G2.4)

7.1 Rationale for Integration

In a verification-first design, the ordering plane (Momentum consensus) and verification plane (local proof validation) are separate. Narwhal enhances data dissemination by providing verifiable propagation evidence, while Tusk sequences already-disseminated batches into ordered commitments. Together, they improve performance while preserving correctness guarantees.

- Narwhal: provides attestations of data availability.

- Tusk: orders only those batches with proven dissemination.

These overlays strengthen liveness and efficiency without affecting correctness.

7.2 Narwhal as a Data Plane Overlay

Narwhal constructs a Directed Acyclic Graph (DAG) of certified data batches. Each vertex represents a batch of proofs or transactions, signed by multiple peers.

When adapted to proof dissemination:

1. Relays and sentries broadcast proof batches.
2. Peers issue attestations confirming receipt and integrity.
3. The resulting DAG encodes verifiable dissemination coverage.

An artifact's empirical availability can be estimated by:

$$A(a) = \frac{|W(a)|}{|P|}$$

where $|W(a)|$ is the number of attestations and $|P|$ the total peers in that round. This enables the network to measure data availability cryptographically rather than assume it. (P7.1)

7.3 Tusk as an Ordering Plane Overlay

Tusk sequences references to Narwhal batches, finalizing only those with sufficient attestations. In this adaptation:

- The ordering plane commits to references, not to raw proofs.
- Batches lacking adequate attestations are skipped safely.
- Finalization applies only to well-propagated data.

This structure ensures that ordering work is economically aligned with data accessibility — avoiding waste on unavailable proofs. (P7)

7.4 Coordination Between Layers

Layer	Primary Role	Inputs	Outputs	Depend
Narwhal	Measure dissemination	Proof bundles + attestations	DAG of availability evidence	Independ
Tusk	Establish canonical order	DAG vertices	Ordered commitments	Independ
Verification Plane	Local proof checking	Ordered references	TRUE / FALSE / REFUSED	Independ

Each layer reinforces the others but none rely on cross-layer correctness. Verification remains local and safe even if overlays fail.

7.5 Integration with Refusal Semantics

Narwhal and Tusk reduce avoidable refusals (due to transient unavailability) but do not alter refusal logic itself. Verifiers still refuse in the following cases:

- DATA_UNAVAILABLE — proof data missing within bounds.
- OUT_OF_SCOPE — data expired or archived.
- COST_EXCEEDED — proof exceeds declared limits.

Overlays improve the first category by making availability measurable, but refusal remains the safety boundary. (G2.7.2)

7.6 Incentive Alignment

Participation in these overlays can be incentivized through:

- Micro-rewards for valid attestations.
- Reduced rework and faster finality for ordering participants.
- Supervisor bounties targeting under-disseminated batches.

This creates measurable reputation metrics — availability proven by signatures, not assumptions. (P8.2)

7.7 Implementation Notes

Narwhal and Tusk are optional. Subnetworks may adopt them independently; light clients can ignore them entirely. They operate at the transport and ordering levels, leaving verification semantics untouched. Minimalist deployments can omit them with no loss of correctness.

7.8 Conceptual Summary

Narwhal and Tusk act as coordination amplifiers. They make dissemination verifiable rather than assumed. They measure, not replace, availability. Verification correctness remains entirely local; refusal remains valid even when overlays are absent or incomplete. (P10, P11)

8 Supervisor Overlay and Economic Coordination

Supervisors form an optional coordination layer above the verification and dissemination processes. They aggregate refusal data, quantify scarcity, and route incentives—turning information about what's missing into targeted economic signals. This overlay creates a structured feedback loop without introducing authority or trust. (G2.10, P6.3–P8.2)

8.1 Motivation

Refusal witnesses reveal where verification demand exceeds supply. Individually, these signals are local; aggregated, they map the network's economic pressure. Supervisors collect these refusals, identify hotspots of unmet demand, and announce corresponding bounties. Their role is to convert raw scarcity data into structured coordination, enabling efficient response by relays and sentries.

8.2 Core Responsibilities

A minimal supervisor performs four verifiable functions:

1. **Aggregate Refusal Witnesses** — collect and deduplicate w_R messages.
2. **Identify Hotsets** — detect clusters of proofs with high refusal density.
3. **Publish Hotset Announcements** — broadcast signed summaries with suggested bounties.
4. **Route Fulfillment Receipts** — track successful deliveries and distribute rewards.

Every supervisor output is cryptographically verifiable. No correctness assumptions are required—if a supervisor fails or acts maliciously, verification remains unaffected. (G2.10)

8.3 Hotset Formation

A hotset H represents a region of high verification demand:

$$H = \{(b_i, \rho_i, \tau_i) \mid \rho_i = \text{RefusalDensity}(b_i, t \in \Delta T)\}$$

where b_i is a proof bundle, ρ_i its refusal density, and τ_i a timestamp window.

Supervisors announce hotsets once refusal density passes a threshold and attach proportional bounties:

```
message HotsetAnnouncement {
  uint64 epoch = 1;
  repeated BundleDescriptor hot_items = 2;
  repeated Bounty bounties = 3;
  bytes supervisor_signature = 4;
}
```

Hotsets allow incentive targeting, ensuring replication efforts yield the greatest marginal reduction in refusals. (P6.3, P8.2)

8.4 Reputation Ledger

Supervisors maintain an optional reputation ledger using verifiable fulfillment receipts:

```
message FulfillmentReceipt {
  bytes bundle_root = 1;
  bytes requester_id = 2;
  bytes provider_signature = 3;
  bytes verifier_confirmation = 4;
}
```

Each confirmed receipt raises a provider's score; invalid or unfulfilled claims lower it. Since both sender and receiver sign each receipt, falsification is cryptographically impossible. Reputation improves market efficiency by signaling trustworthy actors without central authority. (P6.3)

8.5 Incentive Routing

Reward flow mirrors the information flow:

1. Verifiers emit refusal witnesses.
2. Supervisors aggregate and publish bounties.
3. Relays and sentries fulfill proof demand.
4. Supervisors verify receipts and release rewards.

Smart contracts or automated channels ensure atomic delivery of both proofs and payments, maintaining economic honesty and auditability. (P6–P8)

8.6 Market Dynamics

Supervisors operate as decentralized marketplaces:

- **Demand Side:** refusals signal scarcity.
- **Supply Side:** relays and sentries compete to fill it.
- **Pricing:** determined dynamically by bounty competition.
- **Equilibrium:** achieved when marginal replication cost equals marginal bounty value.

Multiple supervisors can coexist, competing for efficiency and accuracy in their aggregation and pricing. (P4.1)

8.7 Fault Tolerance and Ignorability

If a supervisor fails, the network continues operating safely:

- Refusal witnesses remain valid without aggregation.
- Verifiers can directly query relays.
- Multiple supervisors can redundantly track the same demand regions.

Malicious supervisors can waste effort or misprice bounties, but cannot compromise correctness. (G2.3.5)

8.8 Integration with Narwhal/Tusk Overlays

Supervisors can utilize Narwhal's DAG to disseminate hotset announcements or Tusk's ordering layer to sequence bounty references. This integration ties economic coordination directly to measurable dissemination metrics, linking data availability to incentive precision. (P7–P8)

8.9 Evolution Toward Specialization

As the network matures, supervisors specialize:

- Latency Optimizers: prioritize rapid proof delivery.
- Archival Brokers: focus on costly historical retrievals.
- Computation Markets: incentivize proof compression or aggregation.

Each specialization increases efficiency without introducing control hierarchies—supervisors remain coordinators, not governors. (P8.2)

8.10 Summary

The supervisor overlay embodies the principle of refusal as coordination:

- Refusals signal unmet verification demand.
- Supervisors convert signals into incentives.
- Relays and sentries act to restore balance.
- Fulfillment receipts verify results.

The outcome is a self-regulating verification economy—trustless, measurable, and adaptive—where incentives continually rebalance verification supply and demand. (G2.12, P8.1–P8.2)

9 Closing Synthesis

The verification-first model replaces global trust with local proof. Its architecture, economics, and coordination mechanisms are not separate layers but expressions of the same design law: bounded verification, with refusal as the safety boundary. The Greenpaper defined the rule; the Purplepaper explored its market dynamics; this Indigopaper shows how those dynamics manifest as a living coordination system. (G2.12, P9)

9.1 The Loop Closed

The trilogy forms a self-consistent logic:

1. **Greenpaper (Architecture):** establishes correctness via bounded verification and refusal.
2. **Purplepaper (Economics):** treats proofs and refusals as economic primitives.
3. **Indigopaper (Coordination):** shows how incentives and specialization emerge from scarcity.

Correctness → Scarcity → Incentives → Coordination → Availability. Each step feeds back into the next, forming a closed, self-improving system.

9.2 Systemic Properties

Property	Origin	Effect
Bounded Verification	Greenpaper	Ensures safety under finite resources.
Refusal as Correctness	Greenpaper	Prevents unsafe approximation.
Proofs as Economic Objects	Purplepaper	Creates measurable incentives.
Coordination through Absence	Indigopaper	Turns scarcity into organization.
Economic Specialization	Purplepaper / Indigopaper	Improves efficiency without hierarchy.
Optional Overlays	Indigopaper	Enhances liveness without risking safety.

These invariants define the logic of a verification-first network: correctness remains local, incentives global, and trust unnecessary.

9.3 Correctness, Availability, and Coordination

- **Correctness is absolute** — determined only by verifiable proofs within resource limits.
- **Availability is emergent** — it increases with participation and incentive alignment.
- **Coordination is optional but inevitable** — efficiency arises naturally as actors respond to scarcity.

Network health is measurable through refusal density. Low refusal rates indicate balanced resources; high rates identify optimization frontiers. Refusal is both a safety valve and a diagnostic instrument. (P7.2)

9.4 Economic Steady State

At equilibrium:

- Each verifier operates at full capacity without overextension.
- Each relay or sentry earns profit equal to marginal cost.
- Supervisors issue bounties proportional to aggregate scarcity.
- Sentinels maintain accountability through verifiable reporting.
- ARNs diffuse demand across space and time.

No global optimization is required. Equilibrium emerges spontaneously from local interactions and bounded verification. (G2.12)

9.5 Network Evolution

Scalability in a verification-first network arises not from relaxing correctness but from increasing coordination density. As participation grows:

- More refusals reveal new scarcity gradients.
- New relays and sentries exploit those gradients.
- Compression, caching, and incentive algorithms evolve competitively.
- Supervisors continuously tune incentives based on verifiable results.

Every adaptation respects the invariant: correctness can never be borrowed, only verified.

9.6 Future Research Directions

Open research areas extend from this foundation:

1. **Refusal Thermodynamics:** formal models of entropy and energy in verification scarcity.
2. **Verification Market Design:** dynamic pricing for proof delivery and caching.
3. **Bound Optimization:** adaptive algorithms for changing verifier limits $R_V(t)$.
4. **ARN Density Modeling:** empirical mapping between node density and refusal probability.
5. **Supervisor Composability:** interoperability between independent economic overlays.

Each topic deepens understanding of refusal as a universal coordination primitive. (G2.11, P10)

9.7 Concluding Statement

A verification-first network transforms computation into an epistemic economy. Every verifier enforces its own truth boundary. Every refusal preserves that boundary. Every incentive aligns with improving the next verification attempt.

Availability grows not from trust, but from bounded correctness and economic feedback. Verification remains local. Availability remains emergent. Refusal remains safe.

These are not design choices — they are consequences of accepting finite verification as the only reliable foundation for distributed truth.

Appendix A — Reading Map

Indigopaper Section	Greenpaper Reference	Purplepaper Reference
Reader Contract	§2.7 Refusal Semantics	§2.3 Refusal as Third Truth Value
Coordination Problem	§2.7.1 Operational Behavior	§7 Absence as State
Network Actors	§2.3.5 Trust Boundaries	§8 Markets Replace Services
Verification Supply Chain	§2.10 Operational Consequences	§6 Proofs as Economic Objects
Ambient Relay Nodes (ARNs)	§2.3.2 Network Model	§11 Offline Verification
Light Client Lifecycle	§1.4 Architectural Implications	§11 Delay-Tolerant Payments
Narwhal/Tusk Overlays	§2.4 Ordering and Availability	§7 Silence as Information
Supervisor Overlay	§2.10–§2.12 Operational Consequences	§6.3 Proof Relay Network
Closing Synthesis	§2.12 Conclusion	§9 Emergent Order

End of the Zenon Indigopaper

Summary Reflection

The Indigopaper completes the Zenon trilogy — a continuous narrative from bounded verification to emergent coordination. It defines how correctness, economics, and self-organization coexist without hierarchy or assumption of trust. From individual verifiers enforcing their limits to global coordination markets balancing scarcity, every mechanism aligns with the same invariant: truth must be proven, not presumed.

Bounded correctness ensures safety. Economic feedback ensures liveness. Coordination emerges as a byproduct of both.

This architecture is not merely a network design — it is an epistemic structure for computation under constraint. Where traditional systems seek consensus to approximate truth, verification-first systems derive truth from proof, and consensus only from ordering.

By treating absence as a signal and refusal as protection, the network achieves something profound: it transforms limitation into structure, scarcity into incentive, and verification into the foundation of a trustless civilization.

End of Document.