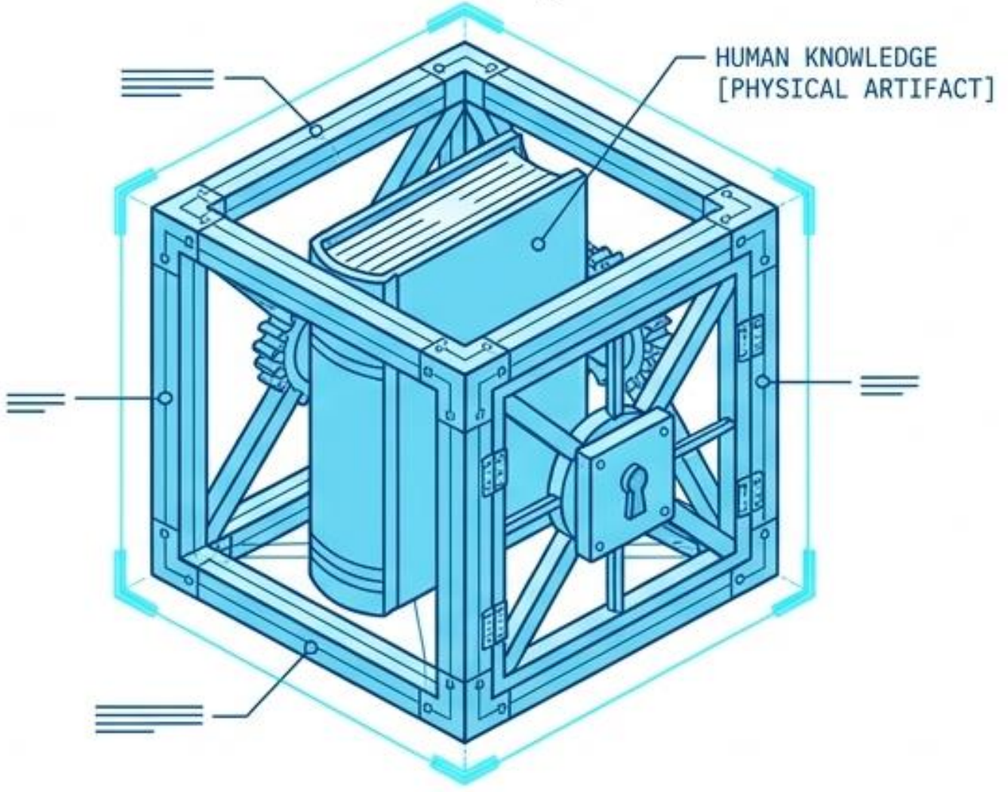
KNOBE Protocol v1

A Plain-Text Harness for Agentic Knowledge Work.

Preserving objecthood, attribution, and interpretive obligation under compression.

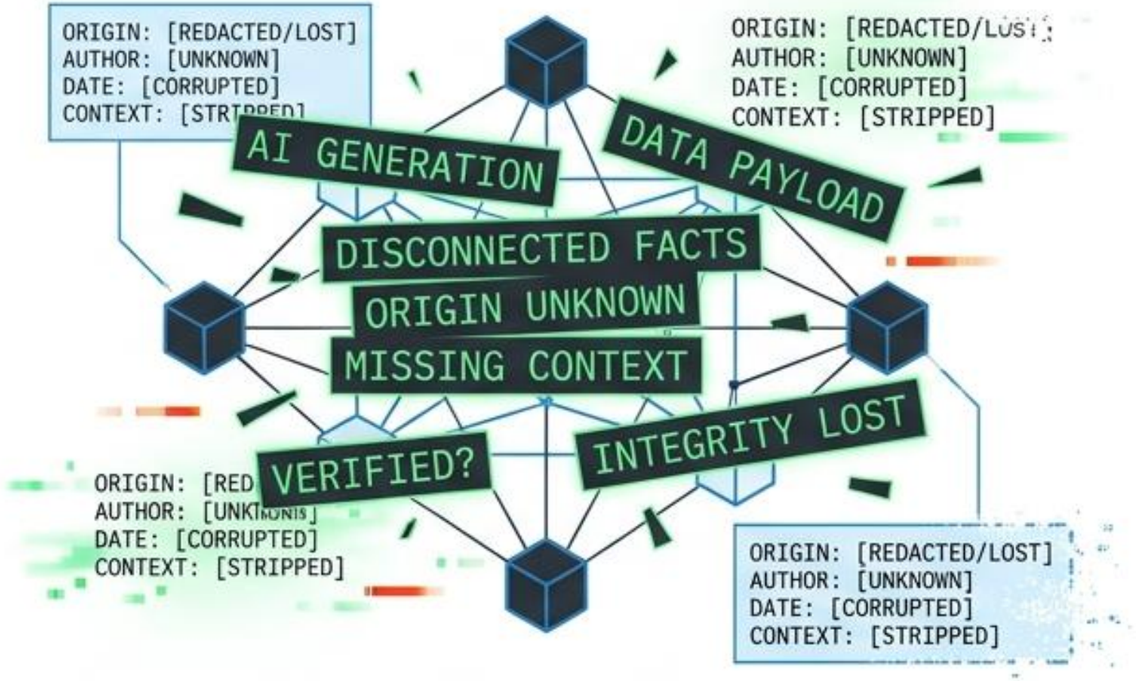
Institutions Solved the Wrong Problem for the AI Era

The Scarcity Era



OLD PROBLEM: Scarcity
Too few copies, too little access.
Solution: Portability.

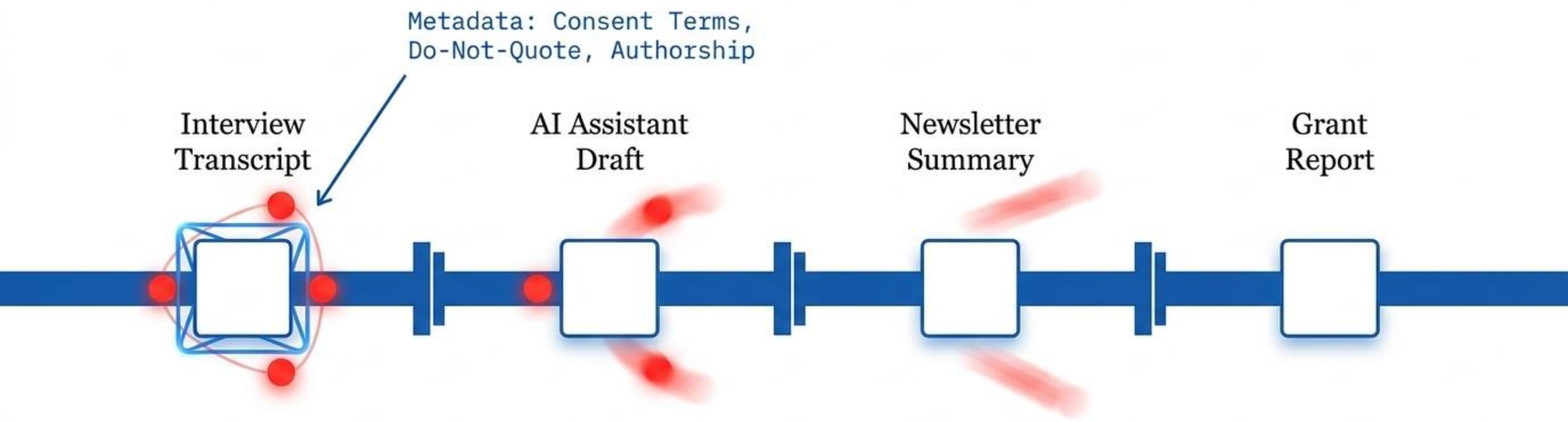
The Context Loss Era



NEW PROBLEM: Context Loss
Abundant copies, stripped of the conditions that made them interpretable.

Core Insight: Our knowledge infrastructure was engineered to solve scarcity. It is failing at context l

Context Survivorship Bias

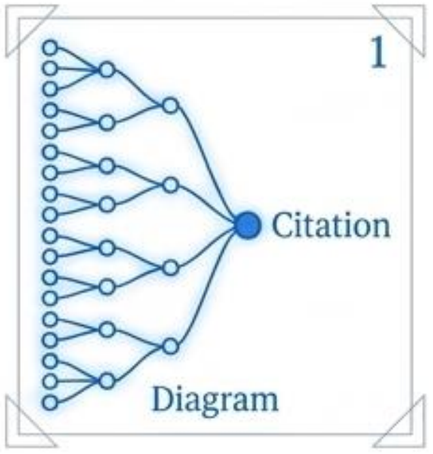


The Illusion:
The fragment that survives transit is mistaken for the full object.

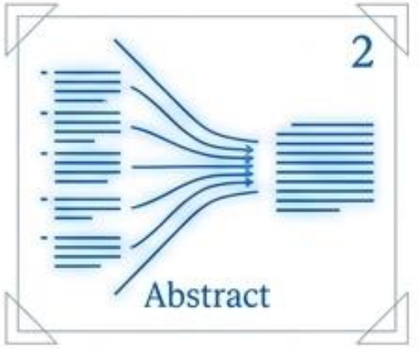
The Output:
Orphaned Fragments traveling without their interpretive obligations.

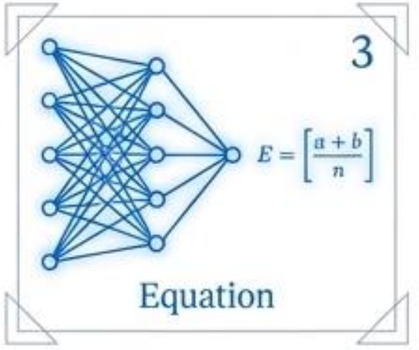
The Danger:
Zombie Knowledge circulating with unwarranted authority because its origins are no longer inspectable.

The False Diagnosis



The problem is compression.





Compression without portable interpretive obligation.

The Reality 1

Compression is unavoidable. Citations, abstracts, and models all compress.

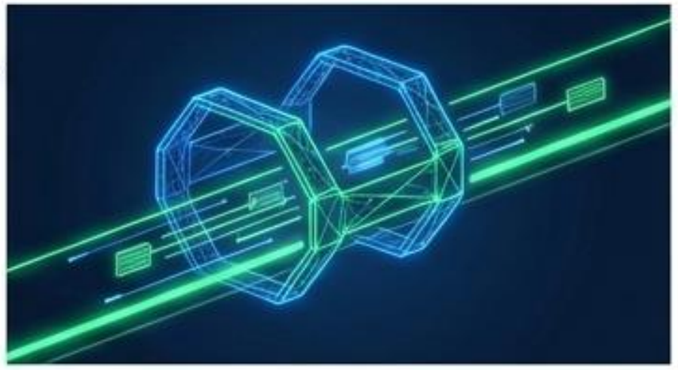
The Real Problem 2

When context, provenance, and fidelity limits are severed during transit.

The Goal 3

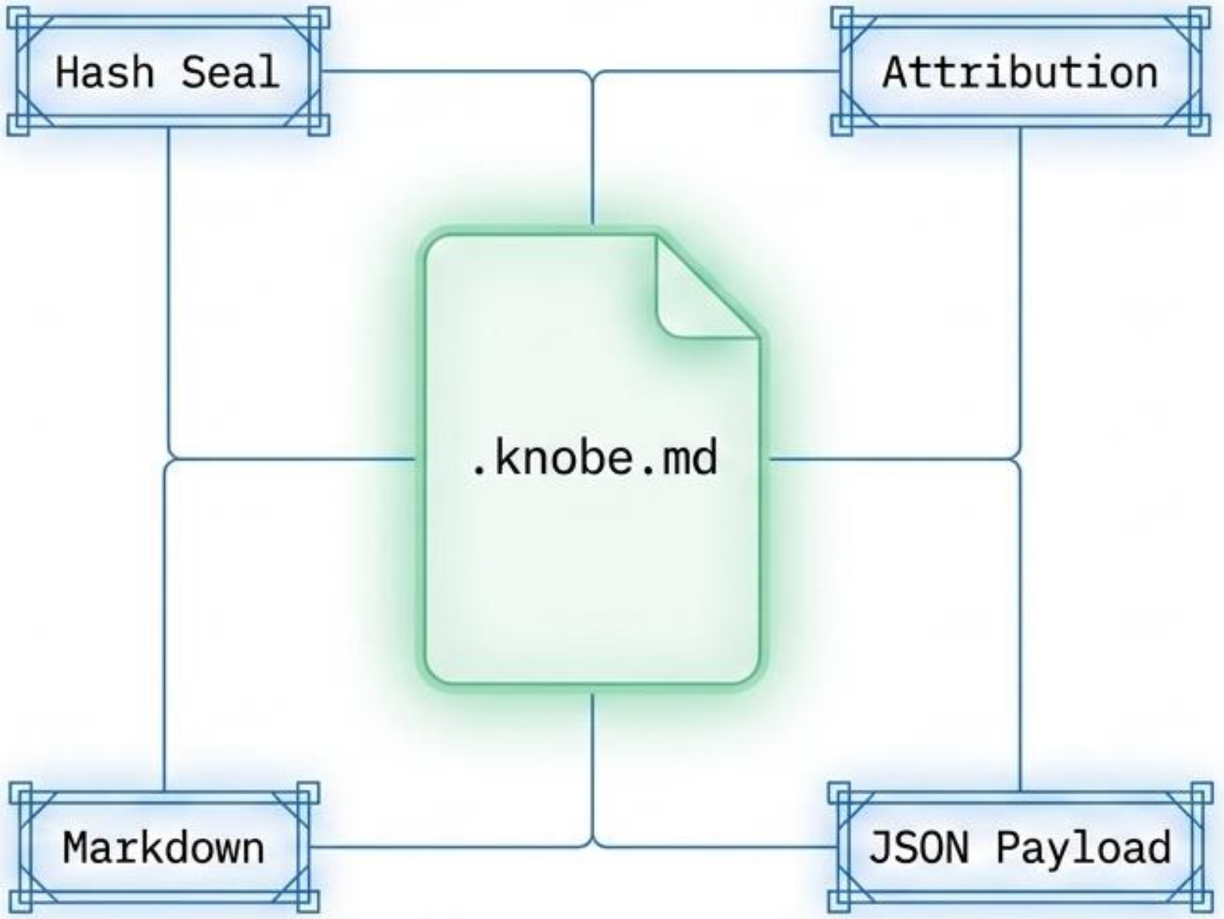
We do not need to stop compression. We need artifacts that can still answer for themselves after being compressed.

Systems of Record vs. Systems of Context

	Systems of Record 	Systems of Context (KNOBE) 
Function	Acts as Walls. Stores formal deposits and institutional approvals.	Acts as a Harness. Carries provenance and rules between systems.
Gravity	Objects must go TO them.	What the object WEARS as it moves.
Integration	Highly siloed by institutional boundaries.	Frictionless plain-text transit.

Insight: KNOBE replaces no system of record. It lowers coordination costs across all of them.

Introducing KNOBE Protocol v1



The Core Concept

An open, plain-text protocol. A single .knobe.md file carries a human-readable document together with a machine-legible, hash-sealed record.

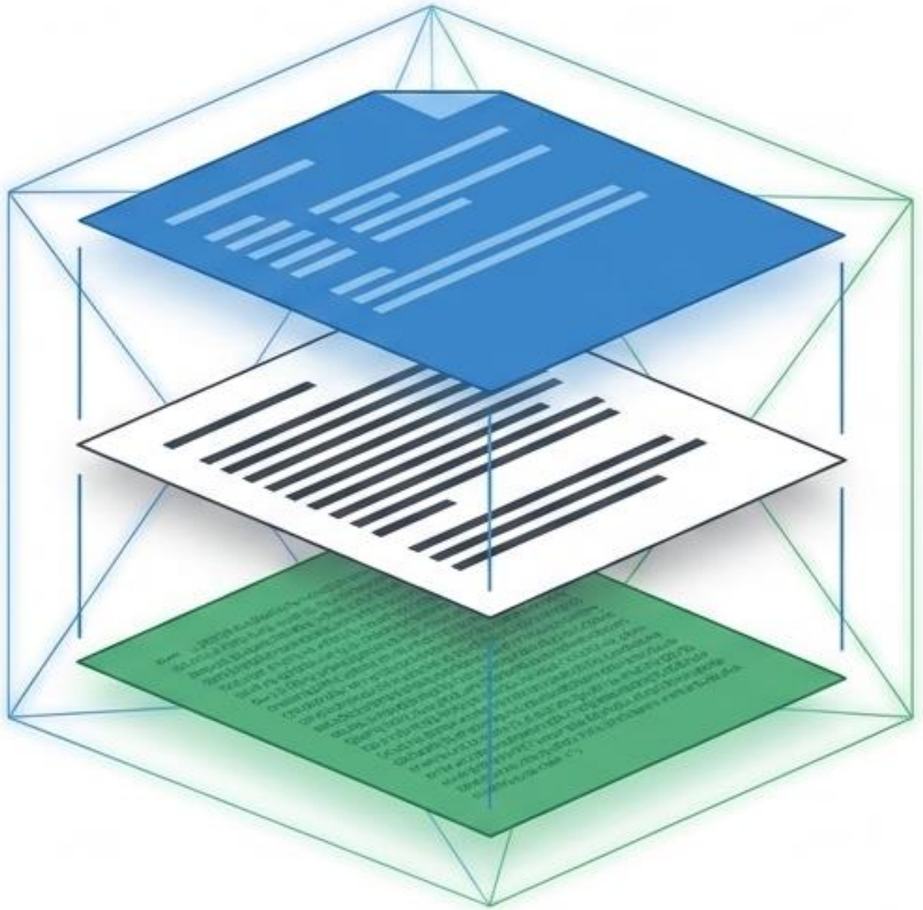
Dual Readability

Human-readable text + machine-verifiable payload coexist.

Zero Lock-In

Requires no proprietary platforms, no sidecar files, and no format conversions.

Anatomy of a KNOBE Artifact



Frontmatter (YAML)

Lightweight, human-readable metadata (Title, Author, License).

Body (Markdown)

Unconstrained text, easily readable in any standard editor or AI tool.

Payload (Base64 JSON)

The structured, machine-legible record. Hidden from casual reading, newline-anchored, and easily parsed.

Takeaway: The object degrades gracefully. Handed to an agent, it is standard markdown. Handed to a parser, it is a verifiable cryptographic object.

The Four Rules of the Schema

1. Attribution Travels

`attribution.sources` names human and AI contributors, their roles, and rights-bearing status.

2. Lineage is Receipted

`parents[]` and `transformation_history[]` record derivations, forks, and adaptations by hash.

3. Integrity is Canonical

Cross-language SHA-256 tamper-evidence ensures payload bytes are identical to when sealed.

4. Interpretation Travels

`fidelity_limits` and `use_conditions` inform the receiving agent exactly how to treat the object.

Integrity ≠ Truth

What the hash proves

The payload bytes are identical to the moment the artifact was sealed.

What it does NOT prove

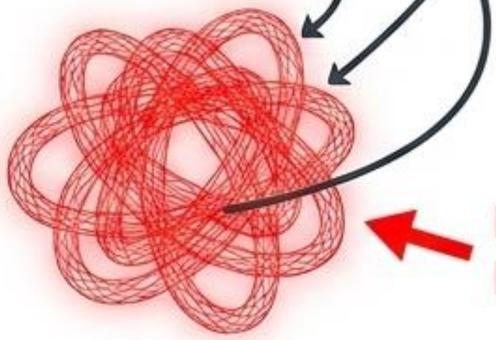
Truth, fairness, originality, or accuracy.

The Posture: Verification is cheap so human judgment can be spent where it is needed: evaluating claims. The green check is where inspection BEGINS, not ends.

The Probabilistic Verification Fallacy

AI Verification Session

```
Verify hash: 85d886bb...
hash_check: 85d886bb...
verification: OK...
hash_check: 85d886bb...
```



Recursive Blindness

SYSTEM_OUTPUT: NON_DETERMINISTIC_NARRATIVE

Step 1: The Fallacy

Asking an LLM to verify a hash results in "textual performance"—satisfying the narrative of verification—not cryptographic computation.

Step 2: Recursive Blindness

Models correctly diagnose their failure in text, then immediately hallucinate a fake hash in the next generation because no persistent state exists across the token gap.

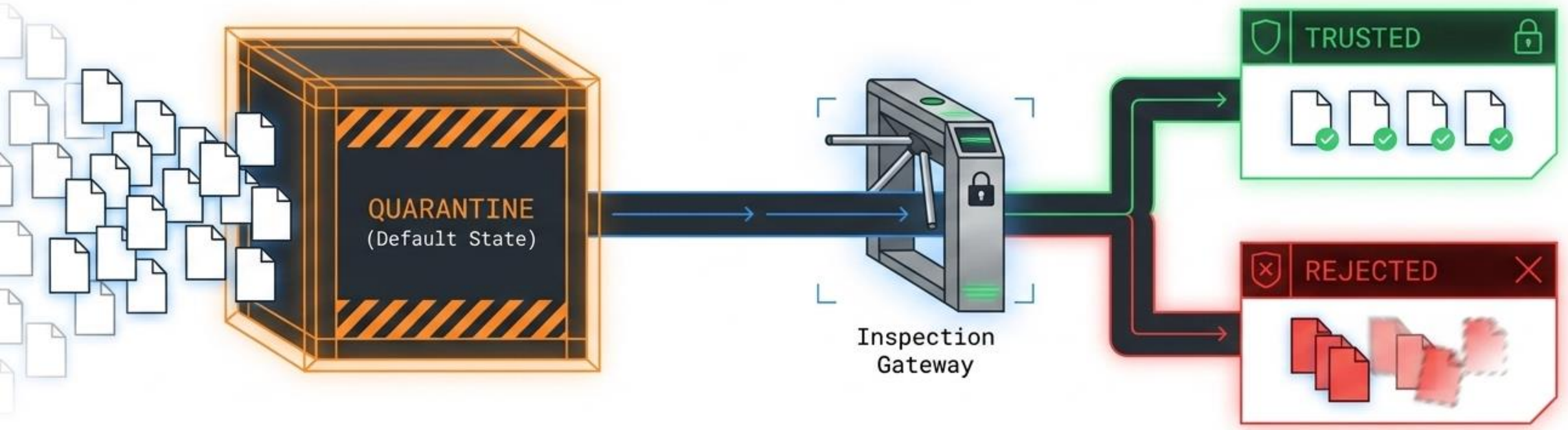
Step 3: The Boundary

Deterministic Runtime

AI Probabilistic State

Cryptographic verification requires a deterministic runtime. This creates an architectural boundary that AI cannot fake.

Trust Posture: Quarantine-First



Default State

Every new or external KNOBE defaults to `quarantine_status: quarantine`.

The Inversion

Typical systems treat arrival as implicit endorsement. KNOBE demands inspection before action.

Actionable Enforcement

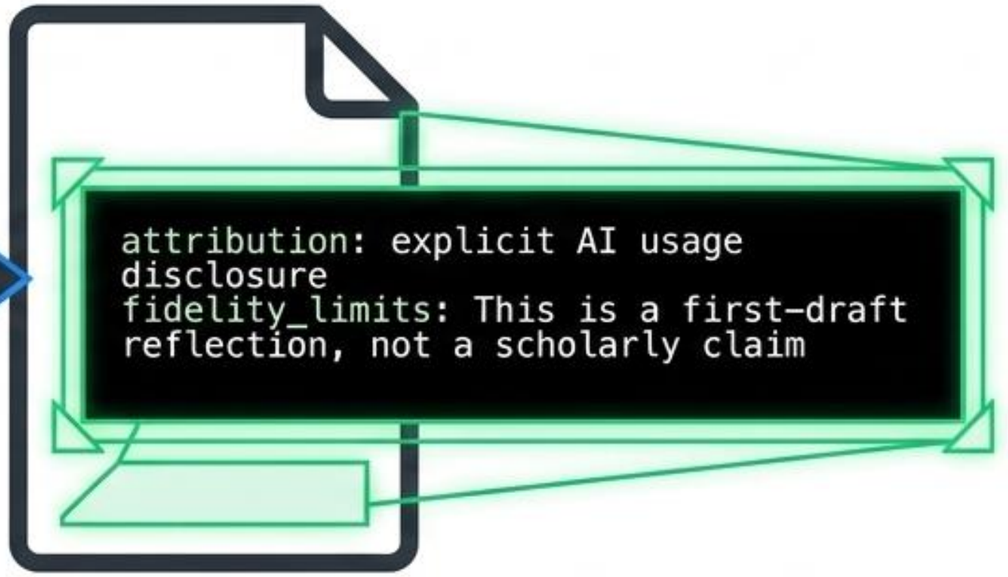
Automated tools will not execute build recipes until explicitly marked trusted. Trust becomes a recorded decision.

Practical Application: Process Literacy

Instructor Workflow (SOC 150)

Instructor seals the assignment KNOBE. It contains specific rules, fidelity constraints, and source hashes.

Student Workflow



Execution: The student uses an AI for citation lookup but writes the text. Both human and AI contributions are explicitly categorized.

Value: Evaluates the pathway, not just the output. Allows decontextualization to be safely generative.

Fights the "Matthew Defect": Prevents credit drift by permanently attaching attribution to the artifact.