

Evidence Grounded AI Control Plane for Enterprise Storage

Sneha Gullapalli

Principal Software Engineer | Distributed Storage AI and Observability

Abstract

Enterprise storage teams are beginning to use AI agents to interpret telemetry, investigate incidents, and recommend operational changes. Directly connecting a language model to administrative APIs, however, creates an unsafe coupling between probabilistic reasoning and authoritative infrastructure control. This paper proposes an evidence-grounded AI control plane for distributed storage operations. The architecture separates an AI reasoning plane from a deterministic authority and execution plane. Agents collect metrics, logs, traces, topology, and change history; compare competing hypotheses; and produce a structured action contract. An independent policy gateway verifies identity, scope, preconditions, reversibility, approval, and expected outcome before an allowlisted executor can act. Every operation is idempotent, auditable, and followed by authoritative outcome verification. The framework supports a staged path from read-only diagnosis to bounded remediation while addressing prompt injection, excessive agency, stale telemetry, and cascading actions. It is a reference architecture and does not claim a production deployment or measured operational improvement.

Keywords enterprise AI, agentic operations, object storage, observability, policy enforcement, autonomous remediation

1 Enterprise AI Needs an Authority Boundary

AI is useful in storage operations because the evidence is distributed. A single customer timeout may require correlating gateway latency, metadata calls, device queues, replica health, network loss, repair activity, and a recent configuration change. A model can search and summarize this evidence faster than an engineer working across separate tools. The same flexibility becomes dangerous when model output is accepted as an administrative command.

Language-model reasoning is probabilistic, while storage changes have durable effects. Removing a node, changing replication, throttling repair, or moving traffic can reduce availability or delay recovery. OWASP describes excessive agency as a risk created by excessive functionality, permissions, or autonomy in tool-using AI systems [1]. The correct architectural response is to prevent the model from becoming the authorization system. The model proposes a bounded operation; deterministic software decides whether the operation is allowed.

2 Control Plane Architecture

Enterprise AI storage operations with an independent authority boundary

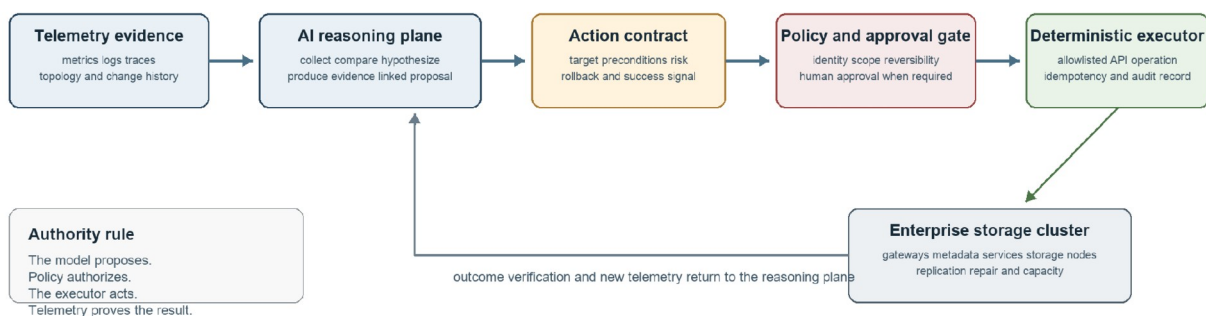


Figure 1 Evidence grounded AI reasoning separated from storage authority and execution

Figure 1 defines two trust domains. The reasoning plane can retrieve approved evidence and create hypotheses, but it has no direct write credential for the storage cluster. The authority plane accepts only a typed action contract. It rechecks the requesting identity, target, permissions, change window, cluster state, risk class, and required approval. The executor exposes narrow operations such as restart one service instance, adjust a bounded throttle, or move traffic from one verified-unhealthy endpoint. It does not expose a general shell or unrestricted database client.

3 Evidence Before Recommendation

An operational recommendation should carry its evidence, not only a narrative. Each evidence record identifies its source, exact query, observation window, resource identity, and collection time. OpenTelemetry supports correlation among logs, traces, and metrics through execution and resource context [2]. For storage systems, the evidence model should also include topology generation, software version, operation type, object-size cohort, replica or fragment state, capacity pressure, and background work.

The reasoning plane first defines the affected cohort and forms at least two plausible hypotheses. It then asks what each hypothesis predicts. If repair I/O is blamed for tail latency, slow requests should overlap repair activity on affected nodes, while comparable nodes without repair should remain healthier. A verifier agent searches for counterevidence and labels missing context. Stale or sampled telemetry lowers confidence; it must never be silently treated as current ground truth.

4 The Action Contract

The action contract converts an AI recommendation into a machine-checkable request. Free-form text can accompany it for human review, but the executor reads only validated fields.

Contract field	Purpose	Example
Target	Limits blast radius	cluster A node 7 gateway service
Preconditions	Prevents action on stale assumptions	node unhealthy and two peers ready
Risk and reversibility	Selects approval policy	bounded reversible restart
Idempotency key	Makes retries safe	incident plus target plus action version
Success signal	Defines authoritative verification	health check passes and error rate recovers
Rollback	Specifies recovery before execution	restore prior routing membership

Table 1 Minimum fields for a storage operation action contract

5 Policy and Human Oversight

Risk policy should classify actions by impact and reversibility. Read-only evidence collection can run unattended within data-access policy. Recommendations can be generated automatically. A reversible single-target change may require one operator approval, while topology reduction, permission changes, destructive storage operations, or actions spanning clusters require stronger separation of duties. NIST’s AI RMF calls for defined human-AI roles, documented oversight, evaluation under deployment-like conditions, and production monitoring [3].

The approval must bind to the exact action contract, not to a conversational summary. If the target, generation, preconditions, or command changes, approval expires. Kubernetes provides a useful analogy: authorization decides whether a request may proceed, and admission control can validate object-specific policy after authorization [4]. An enterprise AI control plane should apply equivalent checks even when the underlying target is not Kubernetes.

6 Safe Execution and Recovery

The executor performs one allowlisted operation and writes a durable execution record before and after the external call. A timeout creates an unknown outcome. The executor must inspect authoritative state before retrying; otherwise, a successful but unacknowledged operation can be duplicated. Idempotency keys, generation checks, and bounded retries make recovery predictable.

Outcome verification is separate from command completion. An API returning success does not prove that latency recovered, replication stabilized, or traffic moved safely. The contract’s success signal is measured from the cluster and compared with a baseline. If the signal fails, the workflow can roll back when safe or escalate with the completed effects preserved. The AI agent may explain the result, but it cannot rewrite the audit record or redefine success after execution.

7 Security Threats

Telemetry, tickets, runbooks, and object metadata are untrusted model inputs. A log line can contain injected instructions, and retrieved documents can be poisoned. These inputs are data, never authority. Tool selection is constrained by policy; credentials are issued to the executor for one operation and one target; secrets are not placed in model context. OWASP notes that prompt injection and compromised peer agents can trigger harmful actions when systems hold excessive permissions [1]. Least privilege and an independent action gateway limit the resulting impact.

Cross-tenant isolation is equally important. Retrieval must enforce the user's identity before evidence enters the model context. A more privileged agent cannot act as a deputy to retrieve data for a less privileged requester. Every evidence access, proposal, approval, execution, and verification event receives a traceable identity and correlation identifier.

8 Staged Adoption and Evaluation

Adoption should advance through measured autonomy levels: offline incident replay, read-only production diagnosis, operator-reviewed recommendations, approved reversible actions, and finally narrow automatic remediation for repeatedly validated cases. Progress depends on evidence, not a calendar. Required measures include root-cause localization accuracy, unsupported-claim rate, action-policy rejection rate, false-remediation rate, recovery time, rollback success, and operator acceptance.

Evaluation must include stale telemetry, missing traces, conflicting evidence, prompt injection in logs, concurrent topology changes, executor timeouts, and failed rollback. A system that performs well only when every source is complete has not demonstrated production readiness. NIST recommends that AI behavior and assurance criteria be evaluated under conditions similar to deployment and monitored in production [3].

A practical release gate should compare the AI workflow with the organization's current incident process on the same replay set. Reviewers should measure whether the system identifies the affected resource, cites evidence that actually supports its conclusion, abstains when observations are insufficient, and selects an action within policy. Evaluation records should preserve prompts, retrieved evidence, model and policy versions, proposed contracts, reviewer decisions, execution results, and verification signals. This makes regressions reproducible and separates improvements in diagnosis from changes in tooling or data quality.

9 Conclusion

Enterprise AI can reduce the work required to understand complex storage incidents, but operational authority must remain deterministic. Evidence-linked hypotheses, typed action contracts, independent policy enforcement, scoped credentials, human approval, idempotent execution, and authoritative outcome verification create a practical boundary between reasoning and control. This architecture supports gradual automation without asking a probabilistic model to serve as the storage system's administrator.

Declaration of AI Assisted Writing

OpenAI Codex [7] was used during preparation of this manuscript to assist with development of the initial draft, organization of the technical discussion, creation of the original diagram, language editing, reference discovery, and document formatting. The author supplied the subject matter, professional context, technical direction, and review criteria. The author subsequently reviewed and revised the manuscript, verified the cited sources and technical statements, and accepts full responsibility for the accuracy, originality, interpretation, and final content of the work.

References

- [1] OWASP Foundation, "LLM06:2025 Excessive Agency," OWASP GenAI Security Project, 2025. [Online]. Available: <https://genai.owasp.org/llmrisk/llm062025-excessive-agency/>.
- [2] OpenTelemetry Authors, "OpenTelemetry Logging and Log Correlation," OpenTelemetry Specification. [Online]. Available: <https://opentelemetry.io/docs/specs/otel/logs/>.
- [3] National Institute of Standards and Technology, "Artificial Intelligence Risk Management Framework: Generative Artificial Intelligence Profile," NIST AI 600-1, July 2024, doi: 10.6028/NIST.AI.600-1.
- [4] Kubernetes Authors, "Authorization," Kubernetes Documentation. [Online]. Available: <https://kubernetes.io/docs/reference/access-authn-authz/authorization/>.
- [5] OWASP Foundation, "Agentic AI Threats and Mitigations," OWASP GenAI Security Project. [Online]. Available: <https://genai.owasp.org/resource/agentic-ai-threats-and-mitigations/>.
- [6] M. Stanley and R. Sheh, "Concept Note: AI RMF Profile on Trustworthy AI in Critical Infrastructure," NIST, Apr. 2026. [Online]. Available: <https://www.nist.gov/programs-projects/concept-note-ai-rmf-profile-trustworthy-ai-critical-infrastructure>.
- [7] OpenAI, "Codex," OpenAI Developers. [Online]. Available: <https://developers.openai.com/learn/codex>.