THE AKRETIC ENGINE

Deterministic governance for sovereign, cloud-native GenAI in highly regulated environments.

Website Whitepaper | Version 1.2 | February 2026

Audience: C-suite executives, CTOs, CISOs, security architects, and regulated AI program owners.

Executive Summary

The Akretic Engine is a sovereign AI runtime and deterministic governance layer for autonomous GenAI workloads.

It assumes Large Language Models (LLMs) are powerful but fundamentally untrusted, enforcing access, tool execution, and audit evidence *outside* the model via memory-safe infrastructure. The platform is hardware-interoperable—deploying natively across heterogeneous Kubernetes environments (AWS EKS, Oracle OKE, Azure AKS) and air-gapped bare-metal servers—while natively complementing high-performance inference stacks like NVIDIA NIM™.

Enterprises deploy Akretic in **Shadow Enforcer Mode** first to measure policy drift without breaking legacy workflows, then transition to **Active Enforcement** when baselines are mathematically proven.

Outcome: Faster GenAI adoption with evidence-grade controls, structurally neutralized prompt injections, and cryptographically verifiable audit trails for incident response and regulatory reviews.

---

1. Why regulated enterprises are stuck

Enterprises are moving from legacy chatbots to agentic systems that query internal data stores, call APIs, and execute code. That jump is where the risk lives. LLMs are probabilistic and easily manipulated via prompt injection, indirect payload poisoning (via RAG), and tool abuse. In regulated environments (Finance, Healthcare, Defense), the failure mode is not a "bad answer"—it is unauthorized biometric access, evidence gaps, and uncontrolled code execution.

Common failure modes we structurally neutralize:

- **Prompt injection** that coerces an agent into using authorized tools in an unauthorized sequence (smash-and-grab via allowlisted tools).
- **RAG poisoning** where a seemingly valid document contains embedded instructions that hijack downstream behavior.

- **SSRF (Server-Side Request Forgery)** and internal network probing when tools are tricked into fetching internal cloud metadata URLs.
- **Audit gaps:** The inability to mathematically prove to an auditor *who* accessed what, *why*, and *what* was returned to the model at the time of decision.

---

2. The Akretic approach: deterministic perimeter, untrusted model

Akretic does not try to make the model trustworthy. We make the environment trustworthy. We treat the LLM and its agents as Non-Human Identities (NHIs) and enforce deterministic controls at the perimeter: identity, authorization, context access, execution, egress, and audit evidence.

The Paradigm Shift in AI Security:

Legacy: Prompt-Level Guardrails	**Akretic: Deterministic Governance Perimeter**
Model is *asked* to behave.	System *enforces* what is allowed.
Controls live in probabilistic system prompts and API wrappers.	Controls live in memory-safe Gate0, RAG DMZ, and Airlock.
Hard to prove to regulators and auditors.	Produces cryptographic evidence artifacts and WORM audit chains.
Breaks silently under adversarial injection.	Constrains blast radius mathematically, even when the model is tricked.

---

3. Core primitives

3.1 Gate0: Deterministic authorization and policy evaluation

Gate0 is the Policy Decision Point (PDP). It evaluates every privileged action (data access, tool calls, egress intents) against a strictly deny-by-default posture. Policies are expressed in a formal language (AWS Cedar) and bound to cryptographic session tokens (PASETO v4 Local).

- **Capability tokens:** Short-lived, scope-limited permissions for a specific action graph, not blanket API keys.
- **Zero-bottleneck evaluation:** Deployed as a compiled Rust binary close to the inference node, policies evaluate in <1.0ms, avoiding network hop bottlenecks.

3.2 RAG DMZ: Context sovereignty for retrieval

The RAG DMZ sits in front of enterprise vector databases. It enforces identity-aware retrieval, provenance tagging, and classification gates so the model never receives context it should not see.

- **Integrated metadata filtering:** Retrieval enforces RBAC/ABAC labels as part of the database query path (e.g., Hardware-Level Vector ACLs), neutralizing indirect prompt injections.
- **Provenance:** Every retrieved chunk is traceable to its source, classification label, and the specific Gate0 retrieval decision.

3.3 Airlock: Isolated tool execution and egress defense

Airlock executes approved tool intents in ephemeral, tightly constrained environments. Implemented via short-lived Docker containers utilizing explicit Linux kernel capability restrictions (cgroups, --pids-limit 256).

- **No default network:** Containers run with --network none. Any outbound fetch must be routed through the Akretic Orchestrator (Egress Broker), which structurally blocks SSRF attacks against internal Cloud Metadata IPs (169.254.169.254).
- **No shell injection:** Tools are executed via direct argument passing, bypassing vulnerable shell interpreters.

3.4 Iron Ledger & Witness: Tamper-evident audit evidence

Akretic records every policy decision, retrieval, and execution result into an append-only SQLite audit stream.

- **Tamper-evident cryptography:** Events are hash-chained (Merkle-style) and checkpointed with NIST FIPS 204 post-quantum signatures (ML-DSA).
- **Evidence artifacts:** Cryptographically proves who requested what, what policy was evaluated, what was retrieved, and what was returned to the model.

---

4. Deployment and adoption

Akretic is designed to be deployed as a containerized service fleet (via Kubernetes Helm charts or Docker Compose) in customer-controlled infrastructure, including air-gapped SCIF environments.

- **Phase 1: Shadow Enforcer Mode.** Initial pilots run in observation mode. Akretic observes agent behavior and logs what *would* have been blocked without breaking live traffic. The output is a cryptographic **Policy Gap Report** that quantifies policy drift and risky tool sequences for the C-Suite.
- **Phase 2: Active Enforcement.** After baselines are validated, Gate0 policies move to active deny. Policy updates follow a managed GitOps-style approval pipeline to prevent accidental wildcards.

---

5. Cloud-Native & Hardware Interoperable

Akretic is strictly compute-agnostic. We decouple the probabilistic AI inference layer from the deterministic security layer, allowing enterprises to deploy Zero-Trust governance today without waiting on specific hardware allocation timelines.

- **Universal Deployment:** Scales horizontally via standard Kubernetes (Helm) into Oracle Cloud (OKE), AWS (EKS), Azure AKS, or fully disconnected, sovereign bare-metal environments.
- **NVIDIA Ecosystem Synergy:** As an elite infrastructure partner, Akretic natively complements high-performance NVIDIA inference stacks (NIM, TensorRT-LLM). You can pilot on standard cloud instances or legacy Hopper GPUs, and migrate to sovereign Blackwell superpods tomorrow without rewriting a single security policy.
- **Future-Proofed Procurement:** This interoperable strategy reduces integration risk, prevents vendor lock-in, and aligns strictly with agile enterprise procurement patterns. Your compliance architecture will safely outlive your current hardware lifecycle.

---

6. Engineering State & Roadmap

Available in v1.0 (General Availability):

- Gate0 policy evaluation with PASETO v4 session binding and Cedar decisions.
- Airlock transactional execution using ephemeral constrained containers.
- RAG DMZ retrieval enforcement and provenance logging.
- Iron Ledger tamper-evident event chains with ML-DSA post-quantum signatures.
- Operational runbooks for Docker Compose deployment.

Roadmap for Enterprise Hardening:

- Asynchronous execution tier with policy-controlled resource quotas.
- Expanded network egress enforcement patterns (certificate pinning).
- Hardware security offload (e.g., compiling Gate0 directly onto **NVIDIA BlueField DPUs / SmartNICs**) for network-layer enforcement.
- Native Database Translators (e.g., Dynamic Vector ACL injection for **Oracle 23ai AI Vector Search**).

---

7. Compliance and engagement

Akretic produces the exact evidence artifacts required to satisfy common regulatory control families (Zero Trust, HIPAA/GINA, DORA, SEC 13f-2). *This document is not legal advice; customers should validate regulatory mapping with counsel and internal compliance teams.*

Next steps:

1. Request a Design Partner briefing and deployment workshop.
2. Run a Shadow Enforcer pilot to generate your infrastructure Policy Gap Report.
3. Proceed to Active Enforcement to confidently scale your GenAI deployment.