Deterministic Governance Inference & Validation Architecture for Continuous Compliance Intelligence

AIRS Platform — Technical White Paper

Version 1.0 | February 2026

Purvansh Bhatt

Security Engineering & AI Governance

AI Incident Readiness & Security (AIRS)

github.com/purvanshbhatt/AIRS

Abstract

Enterprise adoption of AI-assisted security tooling has exposed a fundamental tension: large language models excel at synthesizing human-readable narratives, but they cannot serve as the system of record for governance, risk, and compliance (GRC) decisions. This paper presents the architecture behind the AIRS platform's dual-engine design — an AI Narrative Engine for consultant-grade text and a Deterministic Governance Engine for auditable, reproducible compliance inference. We formalize the Governance Health Index (GHI), a weighted composite metric, and describe the Internal Governance Validation Framework (IGVF) that provides continuous regression assurance over governance logic without LLM dependency.

1. The "Black Box" Problem

Modern AI security platforms increasingly rely on LLMs to generate risk assessments, maturity scores, and compliance recommendations. While the natural-language output appears authoritative, this approach introduces three structural failures that disqualify it from enterprise audit standards:

Non-determinism.
Given identical inputs, an LLM may produce different severity classifications, score justifications, or framework mappings across successive invocations. Auditors require that the same organizational profile, finding set, and technology stack produce the exact same governance score every time. Stochastic outputs cannot satisfy SOC 2 Type II evidence requirements or ISO 27001 Annex A control traceability.

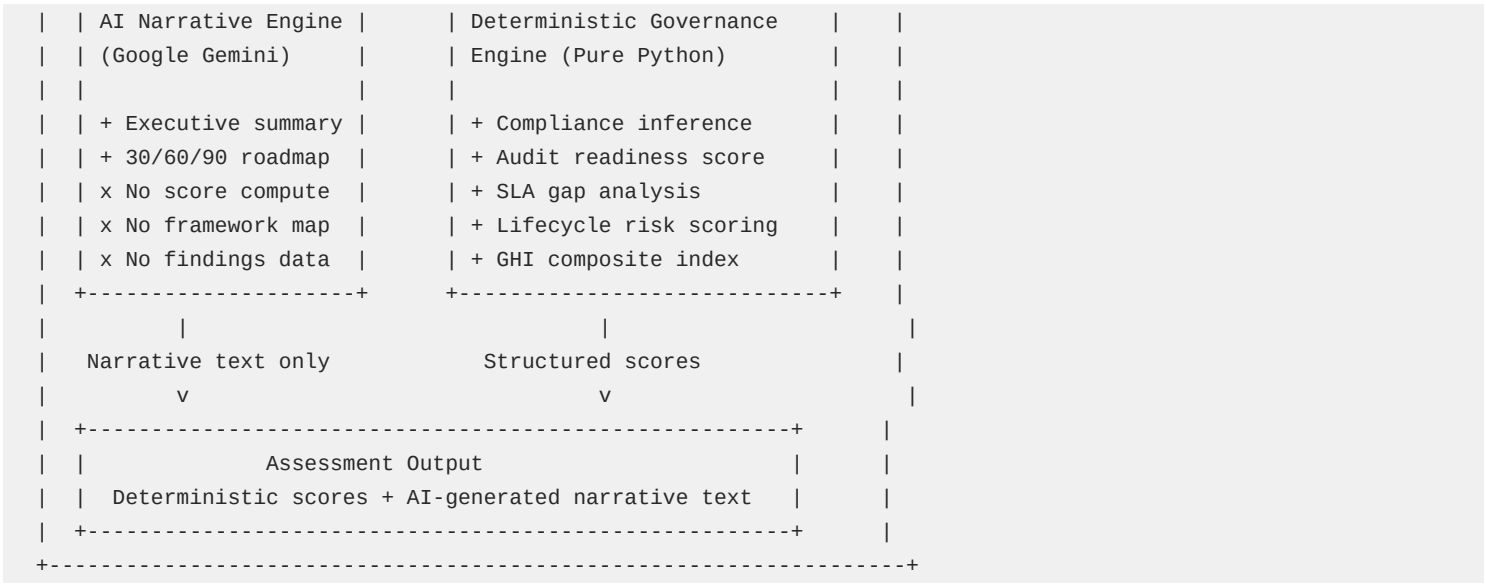
Opacity of inference.
When an LLM determines that an organization is "HIPAA-applicable," the reasoning chain is embedded in transformer attention weights — not in an auditable rule set. An external assessor cannot inspect, version-control, or unit-test the decision boundary. This makes the compliance determination itself an unverifiable artifact.

Score contamination risk.
If the same model that generates narrative text also computes numeric scores, there is no architectural guarantee that a prompt injection, model update, or hallucination will not silently alter a maturity rating. The blast radius of a single model failure extends across the entire assessment output.

These are not hypothetical concerns. They represent the gap between "AI-assisted" and "audit-ready" — a gap that the AIRS architecture is specifically designed to close.

2. Architecture: Dual-Engine Separation

AIRS enforces a strict separation of concerns between two independent processing paths. The AI Narrative Engine and the Deterministic Governance Engine operate in isolation with clearly defined boundaries:



2.1 AI Narrative Engine

The Narrative Engine wraps Google Gemini (via the google-genai SDK) and is scoped exclusively to text generation. It receives pre-computed assessment data — scores, findings, maturity levels — as a read-only input payload and produces two outputs: an executive summary paragraph and a 30/60/90-day remediation roadmap narrative. The LLM cannot modify numeric scores, maturity tiers, finding counts, severity classifications, or any structured data. If the LLM fails, the system falls back to deterministic template-based text with zero impact on governance scores.

2.2 Deterministic Governance Engine

The Governance Engine is implemented as a pure Python package (app.services.governance) with no LLM dependency. It contains four sub-engines:

Sub-Engine	Module	Function
Compliance	compliance_engine	Rule-based framework mapping
Audit Calendar	audit_calendar	Scheduling & forecasting
Lifecycle	lifecycle_engine	Version lifecycle intelligence
Tech Stack	tech_stack	Risk classification & detection

Every function in this package is deterministic: same inputs produce the same outputs and the same audit trail. The entire decision surface is unit-testable, version-controlled, and diff-auditable.

3. The Governance Health Index (GHI)

The GHI is a composite governance posture metric that collapses four independent dimensions into a single 0–100 score with a letter grade:

```
GHI = (Audit x 0.4) + (Lifecycle x 0.3) + (SLA x 0.2) + (Compliance x 0.1)
```

3.1 Dimension Definitions

Audit Readiness (weight: 0.4)
Measures the severity burden of open findings. Starting from 100, deductions are applied per finding severity:

```
Audit = max(0, 100 - (Critical x 15) - (High x 8) - (Medium x 3))
```

Low-severity findings carry zero deduction weight. Only open/in_progress findings are evaluated. This dimension receives the highest weight because unresolved critical findings represent the most immediate governance risk.

Lifecycle Risk (weight: 0.3)

```
Lifecycle = max(0, 100 - (EOL x 25) - (Deprecated x 15) - (Outdated x 5))
```

A component is classified as outdated when it is 2+ major versions behind. Lifecycle status is resolved from a static, versioned lifecycle_config.json — no live API calls — ensuring reproducibility in air-gapped environments.

SLA Gap (weight: 0.2)

Condi		
Target meets tier requirement	on_track	100
Gap <= 0.5%	at_risk	60
Gap > 0.5%	unrealistic	20
Not configured	not_configured	0

Compliance (weight: 0.1)
Measures governance profile completeness. If the organization's attributes trigger applicable frameworks (e.g., processes_phi → HIPAA), score = 100. Configured but no frameworks = 50. Unconfigured = 0. This has the lowest weight because it measures awareness, not control implementation depth.

3.2 Grade Mapping

GHI Range	Grade	Interpretation
90-100	A	Exceeds requirements
80-89	B	Strong with minor gaps
60-79	C	Acceptable, improvements needed
40-59	D	Significant gaps, remediation needed
0-39	F	Critical deficiencies

An organization passes IGVF validation only when it has zero critical issues AND a GHI >= 60.

4. Assurance: Internal Governance Validation Framework

Deterministic logic is only trustworthy if it is continuously verified. The IGVF is the platform's internal assurance layer — a staging-only subsystem that prevents governance logic regression.

4.1 Architecture

The IGVF operates through three interfaces:

- Validation Engine (validation_engine.py) — The core computation module. It orchestrates all four dimension engines, computes the GHI, determines pass/fail status, and emits structured JSON log events traceable by organization_id with no PII exposure.
- Internal API Endpoint (/internal/governance/validate) — Returns HTTP 404 (not 403) when ENV != staging, making it invisible in production. Protected by admin token authentication separate from Firebase auth.
- CLI Tool (scripts/validate_governance.py) — Command-line interface with --org, --json, and --brief flags. Returns exit code 1 on any failure, enabling CI/CD pipeline integration.

4.2 CI/CD Integration

The GitHub Actions CI pipeline includes a dedicated governance-validation job that runs after the main test suite:

```
governance-validation:
  needs: backend-tests
  steps:
    - run: pytest tests/test_igvf.py -v    # 79 unit tests
    - run: python scripts/validate_governance.py --brief
```

Any regression in audit scoring, compliance rules, SLA thresholds, lifecycle classification, or GHI aggregation will fail the pipeline before code reaches staging. The 79-test suite covers boundary conditions and API-level behavior.

4.3 Structured Audit Logging

Each dimension computation emits a structured JSON log containing inputs, calculations, and output score — never PII. These logs enable post-hoc reconstruction of any governance score for SOC 2, ISO 27001, and FedRAMP continuous monitoring evidence.

```
{
  "event": "audit_readiness_inputs",
  "organization_id": "org-12345",
  "critical_count": 1,
  "high_count": 2,
  "deductions": {"critical": 15, "high": 16, "medium": 12},
  "score": 57.0
}
```

5. Evidence-Based GRC: From Self-Reporting to Verified Posture

Traditional GRC workflows rely on self-reported questionnaires: an organization claims it encrypts data at rest, claims it patches within SLA, claims it has no end-of-life components. The AIRS architecture moves toward evidence-based governance through three verification tiers:

Tier	Source	Verification	Status
1: Declared	Self-reported attrs	Awareness	Implemented
2: Observed	Scanners, webhooks	Evidence	Partial
3: SIEM	SIEM/SOAR telemetry	Assurance	Roadmap

The GHI is architected to incorporate higher-fidelity evidence as integration depth increases. The compliance dimension weight (currently 0.1) is intentionally low because Tier 1 awareness provides limited assurance. As Tier 2 and Tier 3 data sources are connected, the model can be recalibrated without altering the composite formula structure — only the per-dimension scoring functions evolve.

This progression mirrors FedRAMP's continuous monitoring maturity model: organizations begin with self-attestation, move to automated scanning evidence, and achieve continuous authorization through real-time telemetry. The AIRS platform provides the computational substrate for each stage.

6. Conclusion

The separation of AI narrative generation from deterministic governance computation is not an implementation detail — it is an architectural invariant that determines whether a platform's output can survive an external audit. By formalizing governance posture into the GHI, gating that logic with the IGVF's 79-test regression suite, and designing for progressive evidence integration, the AIRS platform delivers compliance intelligence that is reproducible, auditable, and extensible — properties that no LLM alone can guarantee.

References

- NIST SP 800-53 Rev. 5 — Security and Privacy Controls for Information Systems
- NIST AI RMF 1.0 — Artificial Intelligence Risk Management Framework
- ISO/IEC 27001:2022 — Information Security Management Systems
- FedRAMP Continuous Monitoring Strategy Guide
- SOC 2 Type II — Trust Services Criteria (AICPA)
- PCI DSS v4.0 — Payment Card Industry Data Security Standard