TECHNICAL WHITEPAPER

Enterprise AI Governance Without Compromise

The case for zero-knowledge BYOC architecture as the only governance model that doesn't trade data sovereignty for visibility.

DAX AI, Inc. June 2025 v1.0

The Problem

Every major SaaS AI governance platform today shares a common architecture: your prompts flow through their servers, your completions are processed on their infrastructure, and your employees' AI interactions are stored in their databases. In exchange, you receive dashboards, policy enforcement, and audit trails.

This is the governance paradox: the tools built to protect your data require you to surrender it.

> **The SaaS Governance Paradox:** You cannot govern AI data sovereignty by sending your data to a third-party governance vendor. The solution to a data risk cannot be another data risk.

Consider what governance actually requires: tracking which models were called, by whom, with what policies applied, and what the outcomes were. None of that requires seeing prompt content. Token counts, cost metrics, latency, and status codes are sufficient for comprehensive governance — and they carry zero confidential information.

The regulatory pressure

The EU AI Act (enforcement begins February 2026 for high-risk systems), NIST AI RMF, and ISO/IEC 42001 all impose requirements on organizations deploying AI in production. Regulated industries face additional obligations: HIPAA for patient data, GDPR for EU personal data, and sector-specific guidance from regulators still writing the rules.

These requirements create a dilemma: enterprises need governance tooling, but the tooling itself must not create new compliance exposure. A HIPAA-covered entity cannot route clinical prompts through a SaaS vendor's servers, even with a BAA.

Zero-Knowledge Architecture

DAX Enterprise is built on a single architectural principle: **the governance layer should never see the governed data**.

This is achievable because inference governance can be decomposed into two planes with different data requirements:

Control plane (governance metadata only)

The control plane manages policies, identities, approvals, audit logs, and budgets. It receives usage events containing: model ID, user ID, token counts, cost estimate, latency, HTTP status code, and timestamp. It never receives prompt content, completion text, or any user-generated content.

Data plane (inference only)

The data plane is an OpenAI-compatible inference proxy that runs in the customer's GCP project exclusively. It validates access tokens, enforces rate limits, applies guardrail policies, and proxies requests to the configured provider.

> **JWT offline validation:** The gateway validates tokens using a public key fetched once at startup. The control plane is never contacted during inference — no per-request callback, no session lookup, no hot-path dependency.

The zero-knowledge guarantee

The guarantee is architectural, not contractual. The control plane has no endpoint that accepts prompt content. The gateway has no code path that transmits prompts to the control plane. The usage event structure contains only numeric and identifier fields. This is not a privacy policy — it is a code constraint.

BYOC Design

Bring-Your-Own-Cloud means both planes run in the customer's GCP project. DAX operates no infrastructure on the customer's behalf and has no access to their data, network, or credentials.

COMPONENT	LOCATION	DAX ACCESS
Control plane API	Customer Cloud Run	None
Portal (Next.js)	Customer Cloud Run	None
Inference gateway	Customer Cloud Run / GKE	None
PostgreSQL	Customer Cloud SQL	None
Redis	Customer Memorystore	None
Provider API keys	Customer Secret Manager	None
Prompt / completion data	Never stored	N/A

Workload Identity for Vertex AI

When routing inference to Google Vertex AI, the gateway uses Workload Identity — the GCP service account bound to Cloud Run is granted `roles/aiplatform.user` directly. No API key is needed and there is no secret rotation burden for Vertex models.

ADG Framework Implementation

The EC-Council ADG framework defines 12 Minimum Controls for enterprise AI governance. DAX Enterprise implements all 12 as native platform features.

CONTROL	REQUIREMENT	DAX IMPLEMENTATION
MC-1	AI System Inventory	Model catalog with lifecycle status, provider, type, and governance metadata
MC-2	Risk Classification	risk_tier + eu_ai_act_category on every model
MC-3	Data Governance	BYOC: inference data never leaves customer cloud by architectural constraint
MC-4	Human Oversight	Approval workflow + autonomy_tier enforcement for autonomous models
MC-5	Incident Response	Incident log with open→investigating→resolved lifecycle + webhooks
MC-6	Bias & Fairness	bias_assessment_url field + compliance dashboard coverage check
MC-7	Explainability	explainability_doc_url field + EU AI Act compliance check
MC-8	Red-Teaming	Guardrail engine: PII, injection, secrets, custom patterns
MC-9	Vendor Risk	Zero vendor data access; shared responsibility model documented in-product
MC-10	Monitoring	Usage analytics across org/model/user/team/project + gateway heartbeat
MC-11	Shared Responsibility	Built-in printable shared responsibility matrix page
MC-12	Continuous Improvement	Prompt template registry with draft→review→approved lifecycle

The EU AI Act imposes tiered obligations based on risk level, with enforcement for high-risk systems beginning February 2026.

Risk classification (Articles 6, 51)

DAX implements four-tier risk classification on every model: `high`, `medium`, `low`, and `minimal`. Each model also carries an `eu_ai_act_category` field. The compliance dashboard tracks classification coverage.

Human oversight (Article 14)

The autonomy tier field classifies each model as Assistive (HITL), Conditional (HOTL), or Autonomous (HOOTL). Autonomous models require an assigned guardrail policy before activation. The approval workflow satisfies Article 14's oversight requirements.

Transparency (Article 13)

The `explainability_doc_url` field links each model to its decision-scope documentation. The compliance dashboard measures coverage and reports a pass/fail check.

Incident reporting (Article 73)

The incident log provides the workflow required for serious incident reporting. Guardrail blocks above a configured severity threshold automatically create incidents, flowing through open→investigating→resolved with timestamped resolution notes.

NIST AI RMF Alignment

The NIST AI RMF organizes AI governance into four functions: Govern, Map, Measure, and Manage. DAX implements controls across all four.

GOVERN

log provides the accountability evidence trail.

MAP

The model catalog with risk tier, autonomy tier, EU AI Act category, and use-case context captured via justification and intended_use on access requests satisfies MAP requirements.

MEASURE

Usage events with org/model/user/team/project dimensions give the measurement data MEASURE requires. The incident log with severity classification provides incident metrics.

MANAGE

The guardrail engine implements risk treatment controls. Budget enforcement with hard blocks prevents runaway spend. The incident lifecycle implements response and recovery processes.

Shared Responsibility Model

AREA	DAX	CUSTOMER
Governance engine code	Provides + maintains	Deploys + operates
Inference gateway	Provides + maintains	Deploys in own GCP
Prompt / completion data	Never receives	Solely responsible
Provider API keys	Secret Manager integration	Creates + manages secrets
Model risk classification	Provides fields + dashboard	Classifies each model
Guardrail policy content	Provides engine + templates	Configures policies

SSO / identity	Provides SAML framework	Manages IdP + enforces MFA
GCP IAM + VPC	No access	Configures + manages
Formal certification	Readiness tooling	Engages auditors

The critical distinction is in the prompt/completion row: DAX has no access because there is nothing to access. The gateway never sends this data to the control plane. This is not a contractual promise — it is an architectural constraint enforced in code.

Getting Started

Option 1: Self-hosted (Community, free)

```
$ git clone https://github.com/dax-ai/dax-enterprise
$ cd dax-enterprise
$ docker compose up --build
# Portal → http://localhost:3001
# API docs → http://localhost:8090/docs
```

Option 2: Live pilot (hosted by DAX)

The full platform is running at app.enterprise.dax.studio. Sign up with your work email — no credit card, no sales call.

Option 3: Deploy to your GCP project

```
$ cd data-plane/infra/gcp
$ terraform init
$ terraform apply -var="project_id=YOUR_GCP_PROJECT"
```

Design Partner Program: The first 5 enterprise customers receive Pro tier for 6 months in exchange for direct product feedback. Contact hello@dax.studio.