Research done by Claude

Building a deterministic policy layer for AI agents

The AI agent guardrails market is consolidating fast — five major acquisitions in 2024-2025 — yet no product has solved the core problem: deterministic, auditable control over what AI agents can do. The market is projected to grow from **$750M to $5.6B by 2030**, with enterprises deploying AI agents at scale but lacking the governance infrastructure to trust them. This creates a once-in-a-decade opportunity to build the "OPA for AI agents" — a policy-as-code enforcement layer that sits beneath the probabilistic LLM world and provides the hard guarantees regulated enterprises need. Here's what the technical architecture should look like, who to sell it to first, and how to price it.

AUI's neuro-symbolic approach reveals the architecture gap

AUI (aui.io) offers the most instructive reference point in this space, though it's solving a different problem. Their product, **Apollo-1**, is a neuro-symbolic foundation model that fuses LLM language capabilities with a symbolic reasoning engine to build task-oriented conversational agents. The architecture separates perception (neural, probabilistic) from reasoning (symbolic, deterministic) — an encoder translates natural language into symbolic state, a stateful reasoning loop computes next actions deterministically, and a decoder generates natural language responses.

What matters for this analysis isn't Apollo-1 itself but its architectural insight: **determinism must be structural, not bolted on**. AUI's symbolic reasoning engine enforces constraints at the model level — "given the same state, the Reasoner always makes the same decision." Every decision is traceable, every rule firing is auditable. This stands in sharp contrast to the guardrails products that attempt to achieve similar outcomes by wrapping probabilistic models with additional probabilistic checks.

AUI has raised **~$60M at a $750M valuation cap**, partnered with Google Cloud, acquired Quack AI (adding dozens of enterprise customers), and targets Fortune 500 companies in

regulated industries. They reached general availability in February 2026 after seven years in stealth. Their customer logos include WalkMe, Yotpo, and Artlist, with unnamed Fortune 500 companies in beta. Their self-reported benchmarks claim **91% on τ-Bench** versus Claude-4's 60%, though these await independent verification.

The key takeaway: AUI builds the agent itself with determinism baked in. The market opportunity this report addresses is different — building the **infrastructure layer** that enforces deterministic policies on *any* AI agent, regardless of how it was built.

Every existing guardrails product has the same fundamental limitation

The current landscape splits into five categories, and each has critical gaps that an infrastructure-native policy enforcement product could exploit.

SDK/Library products (Guardrails AI, NeMo Guardrails) embed directly into application code. Guardrails AI provides composable validators — over 100 community-contributed modules for PII detection, toxicity filtering, schema validation — that wrap LLM calls. NeMo Guardrails goes deeper with Colang, a custom DSL for defining dialogue flows and five types of rails (input, output, dialog, retrieval, execution). Both are open source. The problem: they're **Python-centric, require code changes to every application, and can't enforce policies uniformly across an organization**. Each team must independently adopt and correctly configure the SDK. Research shows NeMo Guardrails can **triple both latency and cost** of standard AI applications due to multiple LLM calls per interaction.

API gateway/proxy products (LiteLLM, Portkey, Kong AI Gateway) intercept LLM traffic at the network layer. They're language-agnostic, require no code changes, and provide centralized enforcement. LiteLLM achieves **8ms P95 latency at 1,000 RPS**. But they only see raw request/response payloads — they have **no visibility into agent state, plans, or multi-step workflow context**. They can't enforce "this agent should never execute a trade above $1M without human approval" because they don't understand what the agent is doing.

Hosted cloud services (AWS Bedrock Guardrails, Azure AI Content Safety) offer managed guardrails via API. Bedrock is the most comprehensive, with a unique **Automated Reasoning** feature that uses formal logic for mathematical verification of factual correctness (up to 99% accuracy). But they create vendor lock-in, can't run on-prem, and offer limited customizability within the provider's taxonomy.

Security-focused products (Lakera → Check Point, CalypsoAI → F5, Protect AI → Palo Alto, Cisco AI Defense) emphasize prompt injection, jailbreak detection, and adversarial threat protection. Cisco AI Defense operates at the network layer, scanning all AI traffic through proprietary security models. CalypsoAI uses adversarial agent swarms to simulate **10,000+ attack patterns monthly**. These products are being absorbed into large security platforms — five acquisitions totaling hundreds of millions in 2024-2025 alone. They're strong on security but **weak on business logic enforcement, tool authorization, and workflow-level policies**.

Evaluation-focused products (Patronus AI, Galileo) specialize in accuracy measurement. Patronus's Lynx model outperforms GPT-4o by **8.3% on medical hallucination detection**. Galileo's Luna SLMs provide real-time quality scoring. But they're primarily about measuring quality, not enforcing hard constraints.

The pattern is clear: **no product provides deterministic, auditable policy enforcement across the full agent lifecycle — input, tool use, data access, multi-step workflow, and output — in a way that's language-agnostic, centrally managed, and architecturally sound.** The closest analogy is what OPA did for cloud infrastructure authorization, but for AI agents.

The optimal architecture is a layered hybrid with deterministic core

Based on analysis of every major product's architecture and their respective tradeoffs, the recommended technical design combines four layers, each serving a distinct enforcement function.

Layer 1: Deterministic policy engine (the core)

The foundation should be an embeddable, deterministic policy engine inspired by OPA's architecture. OPA compiles Rego policies to WebAssembly for **sub-millisecond evaluation**, keeps policies and data in memory, and supports deployment as an embedded library, sidecar, or REST API. For AI agent governance, the policy engine should evaluate:

- **Tool authorization** — which tools each agent identity can invoke, with parameter-level constraints (e.g., "trading agent can call `execute_trade` only for amounts ≤ $100K and only equities, not derivatives")
- **Data access control** — what data each agent can read/write, enforced at the query level with PII masking rules
- **Workflow constraints** — multi-step budget limits, required approval gates, time-based restrictions (no production deployments outside business hours)

- **Rate and scope limiting** — maximum actions per time window, maximum cost per session, blast radius constraints

The policy language should be declarative and version-controlled — think YAML for simple rules with a Rego-like expression language for complex logic. Policies must be testable in CI/CD pipelines before deployment.

Layer 2: API gateway/proxy (centralized interception)

A lightweight reverse proxy intercepts all LLM API traffic organization-wide. This provides **language-agnostic, zero-code-change enforcement** for baseline checks: PII detection/redaction via regex patterns, word filters, basic content safety, cost tracking, and comprehensive audit logging. Implementation should target **sub-50ms added latency** following Lakera's benchmark. The proxy handles what doesn't require agent context.

Layer 3: Agent SDK (context enrichment)

A thin SDK (available in Python, TypeScript, Go, and Java) that agents integrate to provide **rich context** to the policy engine: agent identity, current plan state, action history, session metadata. This enables the policy engine to make decisions the proxy can't — like "this agent has already spent $50K of its $75K session budget" or "this is the third retry of a failed action, escalate to human." The SDK should be optional — the proxy provides basic enforcement without it — but unlocks the full policy vocabulary.

Layer 4: ML safety layer (nuanced detection)

For checks that deterministic rules can't handle — prompt injection, toxicity, hallucination, contextual grounding — integrate ML-based evaluators as a second pass. The key design principle: **deterministic checks run first (fast, cheap, auditable); ML checks run only when deterministic checks pass** but the content requires semantic evaluation. This avoids the 3x latency penalty that NeMo Guardrails imposes by running ML on everything. Amazon Bedrock's Automated Reasoning (formal verification) is worth integrating for factual correctness in high-stakes domains.

Control plane: SaaS management layer (the commercial product)

The monetizable surface is the **centralized management plane**: a web UI for policy authoring and testing, compliance dashboards mapping policies to frameworks (OWASP LLM Top 10, NIST AI RMF, EU AI Act, SOX, HIPAA), real-time monitoring of policy decisions across all agents, and audit trail exports for regulators. This is where the open-core model generates enterprise revenue.

The graduated response model is critical — enforcement shouldn't be binary allow/deny. Support **allow, warn, redact, require approval, and block** as response types. A trading agent might be allowed to place small orders autonomously, warned on medium orders, required to get approval for large orders, and blocked from unauthorized asset classes.

Finance is the best first market, DevOps is the fastest path to revenue

Four sectors dominate AI agent adoption, each with distinct characteristics that affect the go-to-market calculus.

Financial services is the highest-value target. Banking AI spending hit **$31.3B in 2024** and is projected to reach **$97B by 2027**. The AI agents market in financial services alone will reach **$4.49B by 2030**. JPMorgan is rolling AI assistants to **140,000+ employees** targeting $1.5B in value. Wells Fargo's AI assistant has handled **200M+ fully autonomous customer interactions**. The regulatory pressure is unmatched: OCC model risk management requirements (SR 11-7), SEC/CFTC scrutiny, FINRA surveillance obligations, BSA/AML compliance, fair lending laws, and the emerging **"Know Your Agent" framework** all create urgent, budget-backed demand for governance tooling. Banks routinely spend on GRC software and are accustomed to paying premium prices for compliance tools. The buying committee includes the Chief Risk Officer, Chief Compliance Officer, CISO, and Head of Model Risk Management. Sales cycles run **6-18 months** but compress under regulatory deadlines.

DevOps/engineering is the fastest adoption market. An extraordinary **90% of engineering teams** now use AI tools. GitHub Copilot has **20M+ users** with **400% YoY growth** and **90% Fortune 100 adoption**. Forty-one percent of all code is now AI-generated. The pain is acute: **29.1% of Copilot-generated Python code** contains potential security weaknesses, **79% of AI coding platforms lack SOC 2 attestation**, and security teams are actively blocking adoption — creating a clear "unlock" value proposition. The sales cycle is **1-6 months**, bottom-up product-led growth works, and developers install tools same-day **81% of the time**. Budgets are smaller per deal but the market is **$7.37B in 2025**, growing to **$30.1B by 2032**.

Healthcare has the highest breach costs but the slowest procurement. AI adoption jumped from **3% to 22%** in two years, and **66% of physicians** used health AI in 2024. HIPAA compliance is stringent — the average healthcare data breach costs **$7.42M** (highest of any sector) — and **73% of healthcare AI implementations reportedly fail HIPAA compliance**. But procurement cycles run **12-24 months**, budgets are constrained by thin margins, and adoption is concentrated in large health systems.

Legal is the smallest, most conservative market with only **21% of law firms** having formally adopted generative AI, though the ethics obligations (attorney-client privilege, duty of competence) create real governance demand among the Am Law 100 firms that are adopting.

Recommended sequencing

Phase 1 (months 0-6): DevOps/engineering. Launch the open-source policy engine targeting AI coding tool governance. Value proposition: "Unlock safe AI coding adoption without 90-day security review bottlenecks." Target VP Engineering and CISO at companies already using Copilot, Cursor, and Claude. PLG motion with a generous free tier. This generates revenue, community adoption, and case studies fast.

Phase 2 (months 3-12): Financial services. Begin enterprise sales targeting Tier 1 banks. Value proposition: "Always-on compliance for AI agents touching regulated workflows — deterministic policy enforcement with complete audit trails for model risk, fraud detection, and customer-facing AI." Target CRO, CCO, and CISO. This becomes the primary revenue engine.

Phase 3 (months 9-18): Healthcare. Leverage banking compliance credibility. Target CMIO and CIO at large health systems with revenue >$1B. Value proposition: "HIPAA-compliant AI agent deployment with automated PHI access controls and audit trails."

Pricing should follow the open-core infrastructure playbook

The most successful developer infrastructure companies follow a consistent pattern: open-source core for adoption, commercial features for revenue, usage-based pricing for alignment with value.

Open-source core (Apache 2.0): The policy engine, SDK, CLI, and basic rule evaluation. This builds developer trust and community — critical for a security/governance tool where customers need to inspect the code. OPA, HashiCorp, and Elastic all proved this model. The core must solve a real problem completely at limited scale.

Free cloud tier: Generous limits for individual developers and startups — enough to evaluate the product thoroughly. Target **10,000 policy evaluations/month** and up to 5 agents. This is the PLG funnel for DevOps teams.

Team tier ($500-2,000/month): Usage-based pricing per policy evaluation or per monitored agent, plus collaboration features (shared policy library, team dashboards, basic audit logs). Model this on **Datadog's per-host pricing** or **LaunchDarkly's per-context pricing**.

Enterprise tier (custom, $50K-500K+/year): SSO/SCIM, compliance reporting mapped to regulatory frameworks (OWASP, NIST, EU AI Act, SOX, HIPAA), data residency options, air-gapped deployment, dedicated support, SLA guarantees, and advanced audit trail features. This is where financial services and healthcare deals land.

Consumption-based flex (for large enterprises): Following HashiCorp's model, allow committed annual spend that burns down across products and features. Banks managing hundreds of AI agents will prefer predictable annual contracts with usage flexibility.

The key metric to track: **free-to-paid conversion rate** (target 3-5% for PLG) and **average contract value** for enterprise deals. Infrastructure products typically see **net revenue retention of 120-140%** as customers expand usage, which makes the initial land critical — even at low ACV.

Conclusion: the market is ready, the architecture is clear

Three converging forces make this the right moment to build deterministic policy enforcement for AI agents. First, **enterprise AI agent adoption has hit the deployment phase** — 79% of organizations have adopted AI agents, but only 14% have implemented at scale, and Gartner warns >40% of agentic AI projects will be canceled by 2027 due to inadequate risk controls. The governance gap is the primary blocker to scaling. Second, **the guardrails market has consolidated around security incumbents** (Cisco, Palo Alto, F5, Check Point), leaving a vacuum for an infrastructure-native, developer-first policy enforcement product — none of the acquirers are building the "OPA for AI agents." Third, **the EU AI Act's high-risk compliance deadlines hit August 2026**, creating a regulatory forcing function that will drive $1B+ in compliance spending.

The technical insight that differentiates this opportunity: existing guardrails products are fundamentally reactive — they wrap probabilistic models with additional probabilistic checks, adding latency and cost while providing no hard guarantees. A deterministic policy engine that treats AI agents as first-class identities with roles, permissions, and auditable decision trails fills a structural gap in the stack. The architecture is proven (OPA handles billions of policy decisions daily across cloud infrastructure), the market is large ($5.6B in AI governance by 2030), and the buyer — the Chief Risk Officer, CISO, and VP Engineering — has budget allocated and urgency mounting. The first product that delivers this cleanly will define the category.

What Praxis is essentially trying to build

Think of it like a bouncer at a club. The AI agent is the person trying to get in. The bouncer (Praxis) checks if they're allowed in, whether they're following the rules, and keeps a record of everyone who came and went. The AI agent never talks directly to your bank system, your hospital database, or your code deployment pipeline. It always has to go through Praxis first, which checks "is this action allowed?" and either lets it through or blocks it.

That's it. That's the core idea. An enforcement layer that sits between AI agents and the real world.

Is this a good idea and does the market want it?

Yes, genuinely. Here's the proof:

Gartner projects the AI governance platforms market will reach $1B+ driven directly by global AI regulations coming into effect. The EU AI Act compliance deadlines hit August 2026 — that's a hard regulatory forcing function that will make enterprises spend on this category whether they want to or not.

79% of organizations have adopted AI agents but only 14% have implemented them at scale — and the number one reason they're not scaling is they don't trust them enough. Praxis is literally the unlock for that.

And the competitive space is wide open. The big security companies like Cisco, Palo Alto, and F5 have been buying up guardrails startups, but none of them are building what Praxis is building. They're all focused on filtering what AI says, not controlling what AI does.

How does it actually differ from what you have in your doc?

Your doc is directionally correct but there are two important nuances.

First, **AUI is not really a competitor**. They build the AI agent itself with determinism baked into the model. Praxis sits on top of any agent, regardless of how it was built. Different layer entirely.

Second, your doc describes Praxis purely as an enterprise gateway, but the smarter path to building this is **open-source core first, enterprise sales second**. The companies that won in

infrastructure — HashiCorp, Elastic, Datadog — all did this. You give the engine away free, build a developer community, and then sell the dashboard, compliance reports, and enterprise features to the big companies. This matters because enterprises don't trust governance tools they can't read the code of.

How should you actually build this as a product?

Don't build a chatbot. Don't fine-tune your own model. Don't build a separate interface where people pick from different AI models. That's a completely different product.

What you're building is **infrastructure**, like a database or a security firewall. Here's the simplest way to think about the three layers you need:

The first piece is the **policy engine**. This is the brain. It holds all the rules — "never approve a transaction over $50K without human review", "never let the agent access patient records outside business hours", "block any code deployment to production on a Friday." These rules are written in a simple declarative format, stored and versioned like code. This is the hardest and most important part to build and it's what no one else has done properly for AI agents.

The second piece is the **API gateway**. This is the physical enforcement point. Every time an AI agent tries to do something — call a tool, hit an API, query a database — the request passes through your gateway first. The gateway calls the policy engine, gets a decision, and either forwards the request or blocks it. This is relatively straightforward to build, similar to how LiteLLM or Kong work but with your policy engine behind it.

The third piece is the **dashboard and audit trail**. This is what you sell to enterprises. Compliance teams need to see every decision ever made, why it was made, what rule fired, and who reviewed it. This is the management plane — the UI, the reports, the alerts. This is what turns the infrastructure into a product that a Chief Risk Officer will pay $200K/year for.

Where should you start and who should you sell to first?

Start with **DevOps and engineering teams**. Here's why: 90% of engineering teams now use AI tools, and GitHub Copilot alone has 20M+ users with 90% Fortune 100 adoption. But 79% of AI coding platforms lack SOC 2 attestation, so security teams are actively blocking engineers from using these tools. Praxis is the thing that unblocks them. The sales cycle is fast, engineers are comfortable trying new dev tools, and you can get paying customers within months.

Then move to **banking and finance**. AI agents are going to test the limits of bank compliance in ways the industry is actively worried about. Banks have massive budgets for compliance tooling, they're deploying AI agents fast, and regulators are watching. This is where your $100K+ contracts come from.

Healthcare comes later — the need is real but procurement takes 12-24 months and it'll slow you down early.

In one sentence: what is Praxis?

Praxis is the policy enforcement infrastructure that lets enterprises deploy AI agents safely by controlling exactly what those agents are allowed to do — not by filtering their words, but by governing their actions.

That's your pitch. That's what you build. That's what you sell.