ChatGPT

Executive Summary

Praxis is a **deterministic execution layer for AI agents**. It sits between an AI agent and its tools or APIs, intercepting every action the agent proposes and checking it against formal policies. If the action violates a business or regulatory rule, Praxis blocks it (with an explanation); otherwise it lets it proceed. This gives **100% guarantee** that AI-driven tasks obey compliance and safety requirements. In effect, Praxis is an “AI firewall” or “control plane” that makes autonomous agents safe for enterprise use.

- **Why now:** AI assistants and agents are moving from question-answering to *taking actions* (e.g. transferring funds, deploying code, accessing patient data). Enterprises hesitate to trust them because **errors are unacceptable** [1] [2]. Regulations like the EU AI Act demand runtime oversight [3]. Praxis fills this gap by enforcing policies *at runtime*, unlocking high-value use cases.
- **Business impact:** By preventing mishaps and compliance breaches, Praxis enables new efficiencies (automating workflows) while avoiding cost overruns from errors. For example, a bank’s AI loan-approval agent can instantly check “>10k requires manager OK” and stop forbidden loans, avoiding fraud losses [4] [1]. An ambulance dispatcher bot cannot reveal PHI in violation of HIPAA [2]. Praxis builds trust in AI by providing provable guarantees.

Introduction (In Plain Terms)

AI agents (think: chatbots that can call APIs, or autonomous scripts guided by language models) are becoming powerful business tools. However, they can **hallucinate or misinterpret commands**, leading to unsafe actions. For instance, an AI “copilot” might authorize a large payment or alter medical records incorrectly. **Praxis is a software layer that sits in front of these AI agents and checks every action**. It contains a **rule engine** (built from company policies, regulations, etc.) and only allows the agent’s action if it passes all rules. In simple terms, “AI says what to do, Praxis says *whether* it can do it”. This separation ensures the agent is *deterministic* about policy compliance, even if the model itself is probabilistic.

Praxis also provides audit logs and explainability: every decision (allow or block) is recorded with the rule that applied. This meets emerging regulatory requirements (e.g. AI Act’s auditability provisions [3]) and helps security teams verify agent behavior. Unlike ad-hoc scripts or LLM prompts, Praxis enforces policies outside the agent’s code [1], making it independent of any particular AI model.

Problem Statement and Real-World Impact

AI agents introduce new **risk vectors** that traditional systems did not face. Key issues include:

- **Hallucinations & Incorrect Outputs:** Models often “make stuff up”. In an agent, a hallucinated answer can become a harmful action. (E.g. a chatbot might attempt a refund it shouldn’t). Even small error rates are unacceptable in finance or healthcare.

- **Undocumented Behavior:** Without a guardrail, the agent might bypass business logic. Enterprises won't deploy an AI "if 'usually works' is our best assurance" [1] [5]. High-stakes use cases *demand* absolute guarantees, not best-effort.
- **Regulatory Non-compliance:** Laws assume human oversight. For example, HIPAA is *the single biggest hurdle* for AI in healthcare [2]; a breach can mean massive fines. Similarly, financial regulators require strict audit trails. Agents need provable compliance.
- **Lack of Runtime Controls:** Existing safeguards (like system prompts or post-hoc checks) are too weak [1]. LLMs cannot be trusted to self-enforce policies (they can be prompt-injected or ignore instructions).

Real-world impact: Banks face fraud and money-laundering risks if AI mishandles transactions. Hospitals risk patient harm and legal penalties if AI assistants violate privacy or medical guidelines. DevOps teams risk major outages if an AI-driven deployment tool misconfigures infrastructure. For example, after the Colonial Pipeline incident, security teams would not allow an "autonomous fix agent" without strict checks. Praxis prevents such costly mistakes by **blocking any action that breaks a rule in real time**.

Prioritized Use Cases

We focus on high-value, high-risk domains where *errors cost millions*:

- **FinTech (Banking, Payments, Trading):**
  AI can automate tasks like transaction approval, loan processing, and account management. But these are heavily regulated. Praxis enforces rules like "flag transactions over \$X", "require KYC for withdrawals" or "cease-trading if market volatility". *Example:* An AI agent tries to wire \$50k from a customer's account. Praxis checks policy (e.g. "max \$10k without manager sign-off"). It sees a violation and immediately blocks the transfer, preventing loss or compliance breach [4] [1]. This allows banks to use AI for customer service and fraud detection with confidence.
- **DevOps / IT Automation:**
  AI agents can manage infrastructure (deploying code, scaling servers, patching). However, mistakes here cause outages or breaches. Praxis ensures all agent actions adhere to policies like "no production deploy without passing tests" or "security group rules cannot be changed except by Ops". *Example:* An AI CI/CD assistant attempts to deploy an update to production. Praxis intercepts, verifies that required code-signatures and vulnerability scans have passed (per policy), then allows or blocks accordingly. It logs the decision for auditors [6] [7]. This avoids incidents and maintains compliance (e.g. SOC2, PCI DSS).
- **Healthcare:**
  Medical AI agents might schedule appointments, triage patients, or even assist in diagnoses. Privacy and patient safety are paramount. Praxis enforces HIPAA and clinical guidelines. *Example:* A hospital's AI voice assistant is asked to access patient data. Praxis checks "has user consent and BAA agreements been obtained?" or "Is the data request limited to necessary fields (minimum necessary rule)?" [2] [6]. If not, it denies access. Similarly, an AI triage bot suggesting treatments will be constrained by protocols (e.g. "no opioid prescriptions without doctor review"). By integrating HIPAA compliance (massive fines for violation [2]) and FDA guidelines into its rulebase, Praxis makes agentic AI viable in health systems.

These use cases illustrate how **Praxis unlocks productivity** (automating workflows) **while eliminating the risk** of rule-breaking errors. By positioning at the action level, Praxis transforms AI agents from "unsafe assistants" into "trusted digital employees."

Competitive Landscape

Direct Competitors (AI Agent Governance Solutions):
- **AUI (Apollo-1):** AUI's Apollo-1 is a proprietary *neuro-symbolic AI model* trained to follow constraints [8]. It integrates rules inside the model's reasoning. This offers transparency ("white-box reasoning" [9]) and policy compliance, but requires using AUI's platform and model. Praxis differs: it is *model-agnostic and platform-agnostic*. We sit externally to any agent, so teams can keep their own LLMs (OpenAI, Azure, etc.) and simply attach our layer.
- **AWS Bedrock AgentCore (Policy Engine):** Amazon's AgentCore provides a gateway and policy engine for agents [10] [11]. It allows natural-language or Cedar-written policies, with deterministic enforcement *outside* the agent code [11]. While very similar in concept, AgentCore is AWS-specific. Praxis is cloud-neutral (can run on-prem or on any cloud) and not tied to Bedrock. We also aim for an open ecosystem (supporting LangChain, ChatGPT plugins, etc.).
- **Lasso Security:** Lasso's AI Security Platform emphasizes *runtime enforcement and auditing* [1] [3]. Like us, they stress "policies must run outside the model, at the gateway" and provide audit trails for frameworks like the EU AI Act [7] [3]. The difference is that Lasso is a commercial security product, while Praxis will be offered as a developer-facing platform with easy authoring and integration. We emphasize machine-verifiable rules and integration with developer workflows (e.g. versioned policy code).

Indirect Competitors / Partial Solutions:
- **NVIDIA NeMo Guardrails:** An open-source toolkit for LLM safety (content filtering, conversational flows) [12] [13]. Guardrails focuses on preventing toxic outputs or guiding dialogs, not on preventing specific actions (e.g. it can stop an answer, but not a transaction). It lacks a general-purpose policy engine.
- **Microsoft TypeChat:** A library that constrains LLM outputs to a JSON schema using TypeScript types [14]. It increases reliability by design, but only ensures the *format* of responses, not their *legality*. TypeChat says "types constrain domain and model uncertainty" [14], but it does nothing to enforce domain rules or side-effect controls.
- **LLM Validators:** Some teams use LLMs to check other LLM outputs (e.g. OpenAI's evals). These are unreliable and reactive. Studies show LLM "judges" often misclassify (only ~35% accuracy vs humans) [1]. They also only flag after the fact. Praxis, by contrast, rejects actions *before execution*.
- **Manual / Consulting Solutions:** Many organizations encode policies manually (scripts, IAM rules) or hire consultants to audit AI usage. This is expensive, error-prone, and not scalable. Praxis will automate policy enforcement in code, providing a consistent, update-once solution, rather than ad-hoc fixes.

Differentiation:
- **Model-Agnostic & Extensible:** Works with any LLM or agent framework (LangChain, Azure OpenAI, custom bots, etc.) via API/plugin integrations.
- **Deterministic Guarantees:** Intercepts every action at runtime ("every agent action is intercepted outside the agent's code" [11]) to give provable compliance.
- **Formal Rule Engine:** Policies are compiled to machine-verifiable logic (Cedar, Rego, Datalog, or custom). This is more rigorous than heuristic filters.
- **Full Audit & Explainability:** Each decision logs the rule used and context [3] [15], aiding compliance

audits (AI Act, HIPAA, etc.).
- **Developer-Friendly:** Natural-language policy authoring, Git-integrated rules, and support for existing workflows. (Praxis can even ingest policy docs via NLP to help authors.)

In summary, Praxis occupies the niche of "runtime execution governance" for AI agents. No other solution offers the same combination of *application-agnostic enforcement, formal policy proof, and ease of integration*.

Value Proposition & Business Model

Praxis delivers value by **eliminating critical risk** and **enabling new automation**:

- **Risk Mitigation:** By blocking any disallowed action, Praxis prevents data breaches, regulatory fines, fraud losses, and operational failures. This can save tens of millions (e.g. one $1B fraud case was reduced by ML at the US Treasury [4]). We're essentially insurance — customers pay to avoid catastrophic mistakes.
- **Compliance-Ready:** We meet regulatory needs out of the box. E.g., HIPAA fines can be \$1M per violation [2] . Praxis's audit logs and strict access control provide proof of compliance. This unlocks enterprise deals (financial institutions, hospitals) that demand security reviews.
- **Accelerating Innovation:** Developers can deploy AI agents confidently. Instead of budgeting months of manual review, they attach Praxis and know policies are enforced. This speed-to-market is a strong selling point.

Business Model:
- **SaaS Subscription:** We plan a tiered pricing per agent instance or per number of actions per month.
- **Usage-Based:** For high-volume customers, a usage fee (per action evaluated) aligns cost with value.
- **Professional Services:** Onboarding large enterprises (policy translation, custom integrations) as a one-time fee.
- **Partnerships/Channel:** We may partner with cloud providers (AWS, Azure) to offer Praxis in their marketplaces, sharing revenue.

Revenue Potential: The AI governance market is growing fast. One analysis forecasts ~\$418M market size in 2026 (36% CAGR) [16] . Our **TAM** (regulated enterprises using AI) could be on the order of billions (tens of thousands of firms globally). For beachhead, we target industries above (banks, insurers, healthcare, large tech). Even capturing ~100 enterprise customers in 3 years at \$50k+/year each yields multi-million ARR.

Technical Architecture

Praxis consists of four core components, as illustrated below:

```
flowchart LR
  subgraph "AI Agent Platforms"
    A[LangChain / Custom Agent] -->|API call| Gateway
    B[Chatbot / Plugin] -->|API call| Gateway
    C[CI/CD Bot / DevOps Script] -->|API call| Gateway
  end

  subgraph "Praxis Service"
    Gateway[Sidecar / API Gateway] --> Engine
    Engine[Policy Engine / Rule Database] --> External
  end

  subgraph "Enterprise Backend"
    External[(APIs, Databases, Services)]
  end

  Gateway -->|Log decision| Audit[(Audit Logs)]
  Audit --> Compliance[Security/Audit Team]
```

Figure: Platform Integration — Praxis (red) sits between any AI agent and the enterprise backend. The Gateway intercepts actions, the Policy Engine verifies them, and all decisions are logged for compliance.

- **Sidecar / Gateway:** A lightweight proxy or SDK integration that all agent calls pass through. This adds **context** (user ID, conversation state, metadata) and then forwards the action request to the Policy Engine. This module ensures "every agent action is intercepted at the boundary, outside the agent's code" [11]. It can be deployed as a microservice or library (e.g. wrap webhooks, ChatGPT plugin calls, LangChain agent executors, etc.).
- **Policy Engine:** The heart of Praxis. It stores formal rules (written in a policy language like Cedar/Rego/SQL) and evaluates actions against them. The engine supports logical reasoning and threshold checks (e.g. time-based rules, counts). We use an **authoring pipeline** so that business analysts can write policies in plain English or a DSL, which gets compiled into these formal rules.

```
flowchart TB
  Policies[Policy Docs/Regulations] -->|NLP parse| DraftRules
  DraftRules --> HumanReview
  HumanReview --> Compiler
  Compiler --> RuleDB[(Rule Store)]
  RuleDB --> PolicyEngine
  subgraph "Authoring Tools"
    HumanReview --> AuthoringUI
    AuthoringUI --> HumanReview
  end
```

Figure: Rule Authoring Pipeline — Natural-language policies are parsed (possibly with AI assistance), reviewed by experts, and compiled into machine-verifiable rules stored in a Rule DB. These rules are then deployed to the Policy Engine.

- **Audit and Logging:** Every action evaluation (allow/block) is logged with timestamp, user, action, and which rule triggered. This satisfies audit requirements (e.g. EU AI Act, NIST AI RMF) [3]. Logs integrate with SIEM or CloudWatch so compliance teams can trace any incident.

- **AI Components (Supporting):** AI is used *outside* the enforcement loop:

- **Policy Authoring:** We use LLMs to help convert policy text to structured rules (auto-suggest, summarization of regs, etc.).
- **Policy Review:** AI agents may generate candidate rules for human review.
- **Explainability:** When an action is blocked, Praxis can invoke an LLM to generate a human-friendly explanation (e.g. “Your request was denied by rule ‘max_transfer=10k’”).
- **Monitoring & Adaptation:** Over time, machine learning can flag frequently-triggered rules or drift, but enforcement remains purely logical.

Runtime Verification Loop

```
sequenceDiagram
  participant Agent as AI Agent
  participant Gateway as Praxis Sidecar
  participant Engine as Policy Engine
  participant Target as Enterprise Service/API
  participant Audit as Audit Log

  Agent->>Gateway: Action request (with context)
  Gateway->>Engine: Check action against policies
  alt Allowed
    Engine-->>Gateway: "ALLOW"
    Gateway->>Target: Execute action
    Target-->>Gateway: Response
    Gateway-->>Agent: Action result
    Gateway->>Audit: Log(allowed, ruleIDs, context)
  else Blocked
    Engine-->>Gateway: "BLOCK: [RuleID]"
    Gateway-->>Agent: Error (policy violation explanation)
    Gateway->>Audit: Log(blocked, ruleID, context)
  end
```

Figure: Runtime Verification Loop — The AI agent proposes an action. Praxis Gateway sends it to the Policy Engine, which returns ALLOW or BLOCK. Allowed actions proceed to the target system; blocked actions return a safe error. All outcomes are logged.

Key Design Points

- **Always Outside the Model:** As Lasso emphasizes, “policies execute in a control layer (gateway/proxy) outside the model” [1] . Praxis never relies on the agent’s internal state.
- **High Performance:** The Policy Engine must be fast (<10ms decision) and horizontally scalable. We’ll use in-memory rule caches and a compiled policy format.
- **Security by Design:** Praxis runs with strict authentication. It never stores user data beyond decisions. It can run in a VPC with TLS, and we plan to get certifications (ISO27001, HIPAA BAA).

- **Integration Points:** Plugins/SDKs for LangChain (via custom Chains), Azure AI (via Middleware), AWS (through Lambda for Bedrock Agents), ChatGPT (via plugin or API). We will provide sample integrations for common agent frameworks.
- **Data Handling & Compliance:** Only non-sensitive metadata flows through (identifiers, lengths, timing). No permanent storage of user input unless needed for debugging (with consent). Praxis supports GDPR/HIPAA by design: e.g. auto-redacting PII from logs, supporting data purging.

Implementation Plan & Requirements

Tech Stack

- **Core Engine:** Likely built in a fast language (Go/Rust/Python with JIT). We may leverage existing engines (Open Policy Agent/Rego, AWS Cedar) for rule evaluation.
- **Infrastructure:** Containerized microservices on Kubernetes for scalability. Use AWS/GCP/Azure for cloud deployments, with on-prem option for high-security clients.
- **Datastore:** A rules database (SQL or NoSQL) for storing versions. Also use a time-series DB or ELK for logs.
- **Interface:** REST/gRPC API for Gateway, Web UI (React) for rule authoring, SDKs in major languages.
- **AI Tools:** GPT-4 (or open LLM) for NLP rule parsing, and for generating explanations. Possibly LangChain for orchestration.

Integration Points

- **LangChain:** Custom LLMChain that routes actions through Praxis API.
- **Azure OpenAI:** A filter layer in the logic app or function calling the Azure OpenAI API.
- **AWS:** Can be deployed as a Lambda authorizer for API Gateway, or integrated with Bedrock AgentCore as a custom “Policy Engine”.
- **ChatGPT Plugins:** A Praxis-hosted plugin could mediate actions from ChatGPT Enterprise.

Performance & Security

- **Latency:** Most checks are simple conditionals, so expect <50ms overhead. For complex rules, use caching or async validation if needed.
- **Scalability:** Stateless Gateway + scalable Engine. Use k8s auto-scaling and load balancers.
- **Security:** Gateway authenticates agents (API keys, JWTs). Policy Engine enforces RBAC. Audits by TLS and log integrity (immutable logs).
- **Resilience:** Graceful degradation: if Praxis fails, it should fail-closed (block everything) or route to safe fallback.

Data & Compliance

- **Data Minimization:** Praxis does not need entire message content. We can send only action parameters and metadata.
- **Encryption:** All traffic encrypted; logs encrypted at rest.
- **Retention:** Configurable logs retention policy (30/90 days) with option to offload to customer’s syslog or SIEM.
- **Third-Party Audits:** Plan SOC2 Type II and HIPAA compliance certification.

Testing & Verification

- **Unit Tests:** Every policy and rule will have automated tests (e.g. simulate edge-case actions).
- **Fuzz/Adversarial Testing:** Use generative tests to find violations. For example, give malformed or malicious actions to ensure blocking.
- **Performance Testing:** Simulate thousands of concurrent checks to ensure scale.
- **Security Testing:** Penetration tests on the gateway.
- **Pilot Beta:** Deploy with one customer for real-world feedback before full launch.

MVP Roadmap (6–12 months)

1. **Core POC (0–3m):** Sidecar + simple rule engine, 5 basic policies, CLI. Demo on a sample agent (e.g. Slack bot).
2. **Beta Release (3–6m):** Full policy CRUD API, web UI for rule authoring, integration with one major cloud platform (e.g. AWS). Customer onboarding materials.
3. **Enterprise Alpha (6–9m):** Harden security, support multiple agents (LangChain, ChatGPT), add logging/integration with SIEM. Begin design-partner pilots.
4. **General Availability (9–12m):** Expand multi-cloud support, fine-tune UI, launch sales/marketing, apply for certifications.

Risks & Mitigations:
- *Policy Gaps:* Teams may write incomplete rules. Mitigate with default-deny stance and “policy health” dashboards.
- *Agent Resistance:* Agents might “work around” rules (e.g. iterative prompts). Mitigate by improving gateway parsing and catching indirect calls.
- *Latency Sensitivity:* High-frequency agents might push limits. Mitigate with rule caching and split-checks.
- *Market Adoption:* If users prefer built-in solutions (e.g. AWS), we position as multi-cloud, easily integrated alternative.

Go-To-Market Strategy

- **Beachhead Customers:** Target large, regulated enterprises: top banks, insurance companies, hospital networks, and major tech firms with heavy DevOps. These customers have the highest need for safe AI and budgets to pay for it.
- **Sales Motion:** Direct enterprise sales and strategic partnerships. We’ll leverage existing relationships with our VC and co-founders’ networks in FinTech/healthcare. Pilot projects (PoCs) with design partners will build case studies.
- **Pricing Model:** SaaS subscription. Tiers by number of agents and rule count. Enterprise plans (including on-prem option) with volume discounts. Possibly an “AgentCore+” addon plan on cloud marketplaces.
- **Partnerships:**
- *Cloud Platforms:* AWS, Azure, Google (so that if a customer uses Bedrock or Azure AI, they can plug in Praxis).
- *Tooling Providers:* LangChain, OpenAI (to integrate into their frameworks).

- *Security/Consulting:* MSSPs and consultancies (Accenture, Deloitte AI practice) that recommend AI governance solutions.

- **TAM/SAM/SOM:**

- *TAM:* All enterprises using autonomous AI. Assume ~50,000 large orgs worldwide. At \$10k–\$50k/yr each, TAM is \$0.5–2.5B/year.
- *SAM:* Regulated sectors (Finance/Healthcare/Gov) ~10,000 orgs.
- *SOM:* Our initial 5-year target ~100–200 customers. At \$100k average ACV, that's \$10–20M ARR. These estimates will be refined bottom-up (e.g. count banks with >\$1B AUM, number of hospitals, etc.).

Marketing & Messaging: Emphasize that *"Fortune 1000 firms are already enforcing AI policy (88% have security oversight boards)* [17] *"*. Use whitepapers and compliance webinars to educate. Show ROI by contrasting potential loss vs cost of Praxis.

Suggested Slide Deck Figure Placements

1. **Title Slide:** [No image]
2. **Problem Slide:** Use *Fig 1 (Enforcement Architecture)* to visually summarize the "control plane" concept.
3. **Use Cases Slide:** [No image, or icon set] (brief bullets about FinTech/DevOps/Healthcare).
4. **Solution Slide:** *Fig 2 (Rule Authoring Pipeline)* illustrating policy creation and enforcement flow.
5. **Architecture Slide:** *Fig 3 (Runtime Verification Loop)* showing step-by-step how an action is checked.
6. **Integration Slide:** *Fig 4 (Platform Integration)* showing how Praxis connects to various agent platforms.
7. **Competition Slide:** [No image, list of competitors with quick notes].
8. **Business Model Slide:** [No image, list (e.g. subscription, usage-based)].
9. **Roadmap/Next Steps Slide:** [No image].

Conclusion & Next Steps

Praxis is the **execution guardrail** that AI agents need. By enforcing compliance and safety rules *before execution*, we transform risky agents into reliable tools. Given the rapid push into AI-driven automation, this capability is in high demand.

Next steps for Praxis include: - **Refining the MVP** with one or two design partners (e.g. a bank and a hospital) to validate real-world policies. - **Performance and Security Audits** to ensure enterprise readiness. - **Fundraising** or partnerships to accelerate development and go-to-market.

Sources: Official product docs and blogs for AI agent governance (AUI Apollo-1 [8], AWS AgentCore [10] [11], Lasso [1] [3], NVIDIA NeMo Guardrails [12] [13], TypeChat FAQ [14]), industry whitepapers and conference presentations [6] [17], and regulatory/security resources (FDA/HIPAA guidelines [2], Treasury AI strategy [4], etc.) were used to compile this report. These ensure the analysis is grounded in primary sources and current best practices.

---

[1] [3] [7] AI Policy Enforcement: Protect Data, Models & Systems
https://www.lasso.security/blog/ai-policy-enforcement

[2] 7 Best HIPAA Compliant AI Tools and Agents for Healthcare (2026)
https://aisera.com/blog/hipaa-compliance-ai-tools/

[4] home.treasury.gov
https://home.treasury.gov/system/files/136/Artificial-Intelligence-in-Financial-Services.pdf

[5] Apollo-1: The Foundation Model for Controllable Agents - AUI
https://www.aui.io/resources/beyond-generative-ai/

[6] [17] How AI Agents Can Help Enforce DevOps Compliance | RSAC Conference
https://www.rsaconference.com/library/blog/how-ai-agents-can-help-enforce-devops-compliance

[8] [9] Technology - AUI
https://www.aui.io/technology/

[10] [11] [15] Amazon Bedrock AgentCore Policy: Control Agent-to-Tool Interactions - Amazon Bedrock AgentCore
https://docs.aws.amazon.com/bedrock-agentcore/latest/devguide/policy.html

[12] [13] Guardrail Concepts — NVIDIA NeMo Microservices Documentation
https://docs.nvidia.com/nemo/microservices/latest/about/core-concepts/guardrails.html

[14] Frequently Asked Questions (FAQ) - TypeChat
https://microsoft.github.io/TypeChat/docs/faq/

[16] AI Governance Market Size & Share | Industry Report, 2033
https://www.grandviewresearch.com/industry-analysis/ai-governance-market-report