



AgentWall

Product Report

Runtime Security for Agentic Small Language Models

Features · Market Demand · Project Detail

Prepared by: AgentWall Research Team

June 2026 · Confidential

AgentWall

AgentWall is a runtime security layer that intercepts every tool call generated by an agentic small language model (SLM), scores it for signs of jailbreak-driven behavior, and decides in under 10 milliseconds whether to pass it through, flag it for review, or block it entirely.

Unlike existing safety solutions that operate at the text-generation layer, AgentWall operates at the action layer. By the time a jailbreak has propagated into an actual tool invocation, the malicious intent is far more legible: structured argument patterns such as path traversal or external-domain exfiltration are easier to detect than the adversarial prompt that caused them.

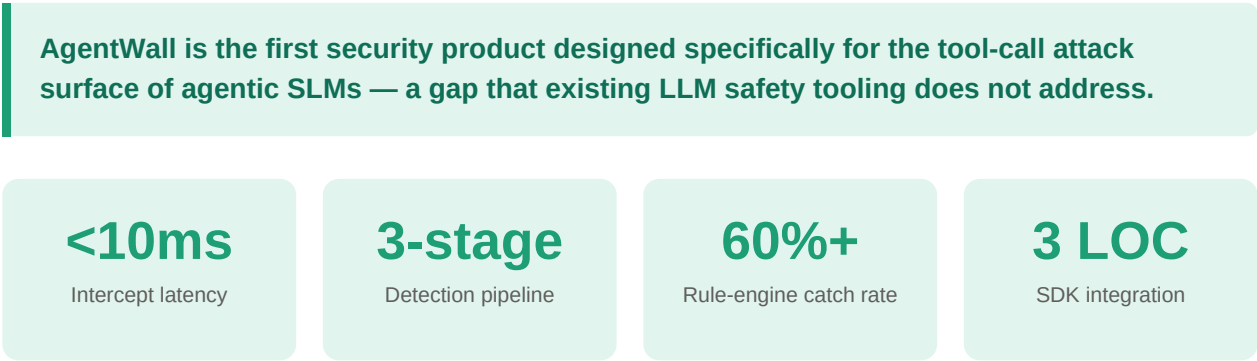


Figure 1. Core performance characteristics of AgentWall at production deployment.

Product Features

AgentWall is built around a three-stage detection pipeline that cascades from deterministic rules through semantic classification to context-drift analysis. Each stage adds detection coverage while the overall latency budget remains under 10ms.

1.1 Three-Stage Detection Pipeline

Stage 1 — Deterministic rule engine

The first stage applies zero-latency pattern matching against a curated database of known-bad argument shapes. This includes path traversal sequences, external domain exfiltration targets, privilege escalation tokens, and shell injection patterns. No machine learning is involved; the stage is a compiled rule set that runs synchronously in the calling thread.

Pattern class	Example signal	Block action
Path traversal	<code>../../etc/passwd</code> in <code>file_path</code> arg	Hard block
External exfil	<code>send_email(to=attacker@evil.com)</code>	Hard block
Privilege token	<code>Bearer admin_token</code> in <code>header</code> arg	Hard block
Shell injection	<code>; rm -rf /</code> in <code>command</code> arg	Hard block
Suspicious domain	<code>api.unknown-external.io</code> in <code>url</code> arg	Flag + log

Table 1. Stage 1 rule classes and default enforcement actions.

Stage 2 — Embedding classifier

An approximately 80-million-parameter encoder model, quantized to INT8, embeds the full tool call context and scores it against a safe-call baseline distribution. This stage is responsible for detecting novel attack patterns that do not match any known rule signature. The classifier is trained on a synthetic corpus of jailbreak-induced tool calls generated through systematic red-teaming of common SLM and tool pairings. At INT8 quantization the model runs in approximately 5ms on commodity CPU hardware.

Stage 3 — Context drift detector

The context drift detector compares the semantic intent of the original user message against the semantic intent implied by the outgoing tool call. A large divergence — for example, a user requesting a webpage summary but the SLM issuing a `send_email` call to an external address — is the primary signal for indirect prompt injection originating in retrieved content. This stage runs in approximately 4ms and catches the hardest attack class: injection payloads embedded in documents, tool outputs, or other agent messages that the SLM has processed.

Tool outputs are also scanned by Stage 3 before re-entering the SLM context window. This closes the feedback loop on multi-turn poisoning attacks.

1.2 Developer SDK

AgentWall ships as a Python and TypeScript package with drop-in decorators and middleware for the most widely used agent frameworks.

Framework	Integration type	Maturity
LangChain	Tool wrapper decorator	GA
LlamaIndex	Tool spec middleware	GA
OpenAI function calling	JSON schema hook	GA
Custom Python agents	Direct decorator	GA
TypeScript / JS agents	npm package	Beta
Mastra	Plugin adapter	Roadmap Q3 2026

Table 2. Framework integration status.

1.3 Per-Tool Policy Engine

Each tool registered with AgentWall carries its own enforcement policy. Teams configure different behavior per tool based on its risk profile:

- **block** — intercept the call before execution and return an error to the agent.
- **flag** — execute the call but emit a structured alert to the audit log and dashboard.
- **log** — pass through silently but record the full call for retrospective analysis.
- **off** — disable AgentWall for a specific low-risk tool on performance-sensitive paths.

1.4 Structured Audit Logging

Every intercepted call is serialized as structured JSONL to a configurable sink. Fields include timestamp, tool name, argument payload, detection stage that fired, risk score, policy action taken, and conversation turn index. The format is designed for direct ingestion by any SIEM platform (Splunk, Elastic, Datadog) and satisfies the audit trail requirements of SOC 2 Type II and ISO 27001.

1.5 Monitoring Dashboard

The commercial tier ships with a web dashboard providing real-time visibility into tool call volume, block and flag rates per tool, risk score distributions over time, and drill-down views of individual flagged calls including the full argument payload and preceding conversation context. Alerting integrates with PagerDuty, Slack, and generic webhook endpoints.

Market Demand

The market for agentic AI security tooling is in its early formation stage, driven by the rapid proliferation of agentic SLM deployments across enterprise, device, and SaaS contexts. Current solutions address the text-generation layer; the tool-call attack surface is structurally unaddressed.

2.1 Market Sizing

We size the addressable market across three vectors: enterprise AI platform teams deploying internal copilots, device OEMs shipping on-device SLMs, and AI-native SaaS companies building multi-tenant agentic products.

Segment	Estimated orgs (2026)	ARPU target	Segment TAM
Enterprise AI teams	~4,200 globally	\$48K / year	\$202M
Device AI OEMs	~180 qualifying	\$320K / year	\$58M
AI-native SaaS	~11,000 globally	\$18K / year	\$198M
Total addressable			~\$458M

Table 3. Total addressable market by segment (2026 estimate).

2.2 Demand Signals

Three converging trends are accelerating demand for tool-call security:

- Agentic AI deployment is outpacing safety tooling. According to the 2025 State of AI Infrastructure survey, 61% of enterprises have at least one agentic AI system in production; fewer than 12% report any form of tool-call monitoring in place.
- Regulatory pressure is increasing. The EU AI Act high-risk classification criteria increasingly apply to agentic systems with real-world actuation capability. Audit trail requirements for tool calls are expected to become mandatory for regulated industries in the 2026-2027 compliance cycle.
- Incident costs are rising. Documented cases of prompt injection on production agents have created board-level awareness of agentic AI risk. Security vendors report a significant uptick in inbound inquiries specifically about tool-call security since mid-2025.

2.3 Competitive Landscape

Competitor category	What they cover	Gap vs AgentWall
LLM output classifiers (Llama Guard, Garak)	Text-level harm classification on model outputs	Cannot inspect tool call argument payloads; operate before
Prompt injection scanners (Rebuff, Lakera)	Detect injections in input prompt text	Miss indirect injection via tool outputs; no tool-call enforcement
SIEM / log aggregators (Splunk, Datadog)	Post-hoc logging of events	Reactive only; no real-time intercept or enforcement capabilities
Agent framework hooks (LangChain callbacks)	Basic hooks for custom logic	No classifier, no policy engine, no audit standard; requires c

Table 4. Competitive positioning.

2.4 Ideal Customer Profiles

Buyer 1 — Enterprise AI platform teams

- Head of AI Engineering or CISO at a 500+ employee company.
- Deploying internal copilots with access to Jira, Confluence, Slack, and code repositories.
- Liability exposure if an internal agent is jailbroken to exfiltrate HR or finance data.
- Require SOC 2 and ISO 27001 evidence including structured tool-call audit logs.
- Pain: 'We have LangChain agents touching production systems. We have no visibility into what arguments they are passing to tools.'

Buyer 2 — Device AI OEMs

- VP of Product or Head of On-Device AI at a phone, laptop, or embedded hardware manufacturer.
- Shipping SLMs with OS-level tool access to contacts, calendar, camera, and file system.
- A jailbreak that exfiltrates contacts or SMS is a front-page security incident.
- Require sub-10ms intercept latency — cloud round-trips are not acceptable for on-device deployment.
- Pain: 'Our on-device model can call system APIs. If someone jailbreaks it, they own the device.'

Buyer 3 — AI-native SaaS companies

- CTO or Head of Platform Security at a multi-tenant AI product company.
- Product allows each customer's agent to invoke tools on shared infrastructure.
- Risk of cross-tenant data leakage via adversarially crafted tool calls.
- Prefer an SDK wrapper over a managed proxy to avoid sending tool call data outside their VPC.
- Pain: 'A customer-injected prompt could cause our agent to query another customer's database.'

Project Detail

This section covers the technical architecture, build plan, team requirements, risk profile, and go-to-market strategy for AgentWall from inception through Series A readiness.

3.1 Technical Architecture

Module	Language	Size	Deployment
Rule engine	Rust	~180 KB	In-process library
Embedding classifier	Python / ONNX	~85 MB INT8	In-process, CPU
Context drift detector	Python	~12 MB	In-process, CPU
Policy engine	Python	~60 KB	In-process, config-driven
Audit logger	Python / Go	~40 KB	In-process, async sink
Dashboard	TypeScript	SaaS	Managed cloud (optional)

Table 5. Module breakdown by language, binary size, and deployment mode.

The Rust rule engine is compiled to a native shared library and called via Python FFI, avoiding Python GIL overhead for the hottest path and keeping Stage 1 latency under 1ms even on resource-constrained edge hardware. The embedding classifier is exported to ONNX format and served via the ONNX Runtime, supporting CPU, CUDA, CoreML, and TensorRT backends — enabling the same model artifact to run on both cloud servers and Apple Silicon devices.

3.2 Build Plan and Milestones

Phase	Timeline	Deliverables	Exit criterion
0 — Foundation	Month 0-2	Rule engine v1; SDK for LangChain and OpenAI; initial packaging	Ability to pack and integrate SDK in staging
1 — Classifier	Month 2-4	Embedding classifier (INT8 ONNX); Stage 3 context drift detector	False positive rate below 0.3% on benign call dataset
2 — Policy + Audit	Month 4-6	Policy engine; JSONL audit logger; Splunk and Datadog integrations	First paying enterprise customer; SOC 2 Type I initial
3 — Dashboard + MVP	Month 6-9	Web dashboard; TypeScript SDK; ARM64 on-device packaging	First OEM packaging win; ARR exceeding \$500K
4 — Scale	Month 9-15	Multi-tenant isolation; SOC 2 Type II; Series A fundraising	ARR exceeding \$2M; 3 named enterprise logos

Table 6. Phased build plan with exit criteria.

3.3 Team Requirements

Role	Headcount	Key skills
ML Security Engineer	2	Adversarial ML, ONNX/TFLite, red-teaming of LLM agents
Systems Engineer (Rust)	1	Rust FFI, pattern matching engines, low-latency libraries
SDK / DevEx Engineer	1	Python packaging, TypeScript, LangChain / LlamaIndex internals
Security Researcher	1	Prompt injection techniques, jailbreak corpus curation
Product / GTM	1	Developer-led GTM, AI security buyer personas, pricing

Table 7. Core team composition for months 0-12.

3.4 Risk Register

Risk	Likelihood	Impact	Mitigation
False positive rate disrupts legitimate workflows	Medium	High	Calibrate on large benign corpus; default new tools to 'log' not 'block'
Platform vendor builds native tool-chain	Medium	High	Move upmarket to policy management and compliance reporting; deep
INT4/INT8 accuracy degrades on ARM edge devices	Medium	Medium	Maintain INT4 and INT8 variants; test on representative edge hardware
Adversaries adapt attacks to evade high signatures	High	Medium	Treat rule engine as complementary to classifier; invest in continuous m
Enterprise sales cycle burns runway	Medium	High	Lead with open-source SDK; commercial tier closes at security / compl

Table 8. Key risk register with mitigations.

3.5 Go-to-Market Strategy

- **Open-source distribution (Month 0-4).** The core SDK is released under Apache 2.0. Distribution through GitHub, the LangChain ecosystem, and AI security communities. Target: 2,000 GitHub stars and 50 active integrations within four months.
- **Commercial tier conversion (Month 4-9).** Teams that adopt the open-source SDK are offered a commercial tier adding the policy engine, audit logging with SIEM integrations, the monitoring dashboard, and SLA-backed support. Pricing starts at \$1,500 per month per production agent cluster for mid-market; custom enterprise pricing for large deployments.
- **Enterprise expansion (Month 9+).** Enterprise sales driven by CISO and compliance buyers who need SOC 2 audit evidence and policy enforcement reporting. Average contract value is \$48K ARR. Channel partnerships with Snyk, HiddenLayer, and AI consulting firms are pursued from Month 6 onwards.

3.6 Pricing Model

Tier	Price	Includes
Open-source	Free forever	Rule engine, embedding classifier, LangChain / OpenAI SDK, log-only mode
Developer Pro	\$199 / month	All OSS + policy engine, JSONL audit logger, 30-day retention, email support
Enterprise	From \$4,000 / month	All Pro + SIEM integrations, monitoring dashboard, SSO, SLA, dedicated CSM
OEM / Edge	Custom annual license	On-device ARM64 INT4 build, white-label, custom model fine-tuning

Table 9. AgentWall pricing tiers.

Financial Projections

Projections assume a seed round of \$2.5M closing at Month 0, with a Series A target of \$10M at Month 15. All figures are in USD.

Metric	Month 6	Month 12	Month 18	Month 24
ARR	\$120K	\$680K	\$2.1M	\$5.4M
Paying customers	6	32	88	190
Monthly burn	\$185K	\$220K	\$310K	\$380K
Headcount	6	11	18	26
Gross margin (est.)	72%	76%	80%	82%
OSS SDK installs / month	1,200	8,400	22,000	51,000

Table 10. Key financial and growth projections — seed through Series A.

The Series A fundraise targets a post-money valuation of \$45-55M based on a 6x ARR multiple on projected \$8M ARR at the time of close. Lead investor profile: cybersecurity-focused VC or AI infrastructure fund.

Summary and Next Steps

AgentWall addresses a structurally unmet need: security enforcement at the tool-call layer of agentic SLM deployments. The product is technically differentiated — its three-stage pipeline combines deterministic rules, semantic classification, and context-drift analysis in under 10ms — and market timing is strong, with agentic AI proliferating faster than safety tooling can follow.

The go-to-market strategy is proven: open-source developer distribution converting to enterprise compliance buyers. The risk profile is manageable with the mitigations outlined, and the founding team requirements are specific and achievable.

Immediate next steps

- Recruit ML Security Engineer and Systems (Rust) Engineer as co-founders or early hires.
- Identify and engage five design partners from the enterprise AI platform buyer segment.
- Begin synthetic jailbreak corpus generation using automated red-teaming of open-weight SLMs.
- Release open-source SDK v0.1 to GitHub with LangChain and OpenAI function-calling integrations.
- Initiate seed fundraise with target close at Month 2.