Outerlimit

Securing Agentic AI

A Reference Architecture for Enterprises

Contents

Securing Agentic AI

A Reference Architecture for Enterprises

Purpose of this document

As enterprises move into the evaluation phase of agentic AI security, the consistent feedback we hear from technology and security leaders is that most AI security vendors are indistinguishable.

The confusion is structural, not coincidental. AI security is not a single product category. Instead, it is a stack of distinct control layers, each operating at a different point in the agent lifecycle. Most vendors operate primarily within one or two layers but position across the entire stack, whilst others fail to sit clearly in any single layer. The results: the narratives converge and enterprise buyers cannot tell the products apart.

This document maps the layers, explains what each one controls, and shows where the critical control layer sits for different AI deployment patterns. Outerlimit's layer coverage is described in the final section.

A note on the term "critical control layer"

Throughout this document, the critical control layer is referenced; in each AI deployment, failure of the control directly causes the impact. Other layers add defense in depth, but the critical control layer is the one that must hold. Which layer is critical depends on what the AI is doing. For example, a chat-only LLM assistant has a different critical control layer than an agent in production with database write access.

Outerlimit

The Agentic AI security stack

Six control layers, each operating at a distinct point in the agent lifecycle.

#	Layer	Summary
1	**Model & Supply Chain Integrity**	This layer focuses on securing AI systems before they reach production. It aims to address risks introduced during the training, fine-tuning and distribution of models – including data poisoning, tampered weight and compromised third party components.
2	**Prompt / Inference Layer (LLM Gateway)**	This layer governs what goes into and comes out of an AI model at runtime. It focuses on detecting and blocking prompt injection attacks, jailbreaks and attempts to extract sensitive information through carefully crafted inputs. Controls at this layer often sit between the user and the model – often implemented as an LLM Gateway or inline proxy.
3	**Tool Execution Layer**	As AI systems move beyond question-and-answer interactions into autonomous agents capable of taking real-world actions – this layer becomes ever more critical. It focuses on controlling what tools an agent can invoke, whilst enforcing least-privilege access. The blast radius of a compromised agent is significantly larger than a compromised chatbot, making runtime tool controls essential as agents are given increased autonomy. This layer is the most forward-looking and arguably the most important emerging category.
4	**AI Security Posture Management (AI-SPM)**	This is a fully established market category with lots of vendors claiming capabilities in this layer. It aims to provide companies with the framework and tools necessary to maintain continuous visibility, control and compliance across all AI assets.
5	**Shadow AI Discovery & Governance**	This layer aims to detect unsanctioned AI usage within organizations. Solutions in this space combine CASB capabilities, network traffic analysis and browser-layer monitoring to discover what AI tools are being used, by whom, and what data is flowing into them.
6	**Adversarial Testing & AI Red-Teaming**	This layer provides proactive, structured testing of AI systems aiming to uncover vulnerabilities before attackers do. Often embedded into the AI development lifecycle rather than treated as a one-off assessment.

Layer-by-layer reference

For each layer: what it controls, what it does not control and the deployment patterns where the layer is relevant.

Layer 1 **Model & Supply Chain Integrity**	**WHAT IT DOES** Attempts to secure the model itself before it reaches production. Verifies the integrity of training data, model weights and third-party components. It detects tampering, poisoning and vulnerabilities introduced during the build and fine-tuning process. **WHAT IT DOES NOT DO** It does not observe runtime agent behavior or misuse of agent tools. **VENDOR ARCHETYPES** Model scanners, AI bill-of materials platforms, and ML-pipeline integrity tooling. **CRITICALITY** Critical for organizations training, fine-tuning, or hosting their own models, and for regulated industries that must demonstrate model provenance.
Layer 2 **Prompt / Inference Layer (LLM Gateway)**	**WHAT IT DOES** Captures prompts and responses at the LLM boundary. Detects prompt injection, jailbreak techniques, and sensitive content in inputs and outputs. Redacts PII, secrets, and confidential data before they reach the model or the user. Filters harmful content categories. **WHAT IT DOES NOT DO** Does not control what an agent does once it has decided to act. Does not see tool calls, file operations, or external API requests. Detection is probabilistic and fails open on novel techniques. **VENDOR ARCHETYPES** LLM firewalls/gateways, prompt-layer detection products, and AI-aware DLP for inputs and outputs. **CRITICALITY** Critical for AI deployments where the system primarily produces text rather than taking actions e.g. AI without tool access, or AI whose tools only retrieve content rather than execute actions.
Layer 3 **Tool-Execution Layer**	**WHAT IT DOES** Captures every tool call at the point of execution including caller identity, tool, parameters, destination, response and decision. Evaluates each call against authorization policy before execution and either allows or denies it. Maintains a tamper-evident audit trail of every action and every decision across the estate. Enforces scope boundaries that agents cannot exceed.

	WHAT IT DOES NOT DO Does not inspect prompt content or detect jailbreak attempts. Does not scan model files or training data. Does not actively probe agents for adversarial weaknesses. **VENDOR ARCHETYPES** Agent authorization platforms, MCP proxies and gateways, tool-call enforcement layers and agent runtime enforcement platforms. There is a limited number of pure-play vendors that operate across this layer. Outerlimit is one example. **CRITICALITY** Critical for any production agentic AI deployment. Agents that take actions (write to systems, call APIs, modify data, execute code) require visibility and deterministic enforcement at the action boundary because the impact is the action, not the prompt that produced it. Observability and enforcement at this layer are typically deployed in sequence: observe first, build the evidence base, then enforce.
Layer 4 **AI Security Posture Management (AI-SPM)**	**WHAT IT DOES** Discovers AI services, models, and agents across cloud and SaaS estates. Maps data access paths and surfaces misconfiguration. Risk-scores assets based on exposure, sensitivity, and privilege. Aims to provide continuous visibility across an organizations AI estate. **WHAT IT DOES NOT DO** Does not enforce policy at runtime. Does not block tool calls or contain in-progress incidents. Posture is a planning surface, not an execution control. **VENDOR ARCHETYPES** AI-SPM platforms and AI-extended CNAPP / cloud security posture vendors. **CRITICALITY** Not applicable under the strict critical control layer definition, though most important for organizations with sprawling cloud and SaaS AI estates that need a single inventory and risk view across multiple AI fabrics.
Layer 5 **Browser, CASB & Shadow-AI Discovery**	**WHAT IT DOES** Identifies unsanctioned AI usage across the organization, including consumer applications, browser extensions and unauthorised API connections and provides visibility into what data is flowing into them. **WHAT IT DOES NOT DO** Does not observe enterprise agent activity inside the application boundary. Does not enforce policy on tool calls made by sanctioned agents. **VENDOR ARCHETYPES** AI-aware CASB and SSPM platforms, enterprise browsers, and SSE-layer DLP for AI traffic.

	CRITICALITY Critical for organizations whose primary AI risk is employees pasting sensitive data into consumer LLMs from a browser.
Layer 6 **Adversarial Testing & AI Red-Teaming**	**WHAT IT DOES** Runs continuous adversarial prompts and attack simulations. Tests agent scope-violation paths and tool-misuse scenarios. Validates whether existing controls block discovered exploits. Maintains a research feed of emerging jailbreak and injection techniques. **WHAT IT DOES NOT DO** Does not provide runtime enforcement or posture management. It is a validation discipline that surfaces gaps for other layers to remediate. **VENDOR ARCHETYPES** Automated AI red-teaming platforms, AI-specific breach-and-attack-simulation tooling, and adversarial-research vendors. **CRITICALITY** Not relevant under the strict critical control layer definition. Adversarial testing is a validation discipline for the rest of the stack. It tells you whether your critical layer controls are holding against an evolving threat landscape. Most important for organizations with mature AI deployments where the cost of an undetected control failure is high.

Why the tool-execution layer becomes increasingly important

In AI deployments without tool access, the prompt layer is the critical control layer. The system can only produce text and as such there is a well-understood failure surface – it might leak personally identifiable information embedded in its training data, expose confidential documents it has been given access to, or be manipulated into bypassing content policies through prompt injection.

Agentic AI changes this picture dramatically. When an AI system is given the ability to take actions in the world via tools to call APIs, execute code, send emails, manage files or interact with external services, the failure surface expands from what the agent says to what the agent does. A misconfigured or compromised agent is no longer just a source of bad output; it becomes an actor capable of exfiltrating data, modifying records, triggering financial transactions or escalating its own privileges within connected systems. Connecting tools changes the structural failure mode. The system no longer just produces text; it produces tool invocations that execute actions. The impact is no longer in the content. The impact is on the action.

Three implications follow:

1. **Layers before the action cannot see the action.** The prompt layer sees what the model receives and produces; the model and posture layers see configuration and deployment state. None see the tool call itself, the destination, the parameters and the cumulative effect of the call sequence. Only the tool-execution layer does.

2. **Many agentic incidents do not originate from malicious prompts at all.** Agents misinterpret legitimate prompts at scale, chain tool calls in unexpected ways or take actions whose individual steps are each authorized, but whose composition is not.

3. **The action is the trust boundary.** Reading a sensitive file, writing to a production database, sending an email, executing a shell command, these are the moments where data crosses domains, state changes, and the impact becomes irreversible. The control that fires at that boundary is the one that prevents loss.

The tool-execution layer is the only layer that can enforce policy after the agent has decided to act and before the action takes effect. It is the deterministic backstop that prompt, model, and posture layers cannot provide. For organizations putting agents into production with real tool access, the tool-execution layer becomes critical. As AI Agents become more deeply embedded in enterprise workflows, the controls at this layer determine not just whether an AI system is safe, but whether it is trustworthy enough to be given meaningful autonomy.

How to evaluate the Agentic AI security market

1. **Identify which layers your AI estate requires.** Not every organization needs all six layers. An enterprise whose primary exposure is employees using consumer AI tools in the browser has very different security needs to an organization running production agentic workflows in AWS Bedrock or Microsoft Copilot Studio. Identify which layers represent a genuine risk rather than theoretical exposure.

2. **Prioritize runtime controls over pre-deployment testing alone.** There is a tendency to focus investment on red-teaming and pre-deployment assessments. However, AI systems are dynamic: models are updated, new tools are connected, and agent behaviors evolve in production. Runtime controls across the tool execution layer provides continuous protection which point-in-time testing cannot replicate. It also ensures you are not locked into models or agent platforms.

3. **Distinguish between controls that describe state, and controls that observe and govern action.** Posture and inventory tools tell you what you have and where the risk is; runtime controls observe what happens and operate at the point of action. They are not substitutes. Conflating them, for example, expecting AI-SPM to reliably prevent agents from acting outside of scope, is a common evaluation mistake in this market.

4. **Avoid assuming platform consolidation equals coverage.** Several vendors are marketing "AI Security" as part of a broader platform play. Whilst consolidation has operational advantages, a single platform early in a market rarely provides deep capability across all six layers – particularly the more specialized ones such as tool execution controls. Once you have identified your primary risk vector, start with vendors who lead in the corresponding layer.

5. **Ensure capabilities connect to reduce risk.** Agent inventories, tool discovery, and risk scoring are now claimed by vendors across multiple layers, but the same capability does very different jobs depending on what it feeds. An inventory that ends at a dashboard is a planning surface; an inventory with risk scoring and a remediation path that feeds a policy engine is part of an enforcement loop. When evaluating overlapping capabilities, the question isn't whether the feature is present, it's whether it connects to action that meaningfully reduces risk.

6. **Expect to stack multiple layers.** Single-platform pitches are a yellow flag in an early market. The vendors who offer best of breed capabilities within the layers combined with seamless cross-layer integrations will provide the most customer value.

7. **Prioritize the layers where the impact is irreversible.** For agentic AI in production, that is almost always the tool-execution layer.

About Outerlimit

At its core, Outerlimit operates at the tool-execution layer. We provide the universal security layer for agentic and tool connected AI. A decentralized authorization network that observes every tool call an agent attempts to make and evaluates it against policy-as-code before execution. Compile-time policy adherence using zero-trust principles. Agents never hold credentials, eliminating credential compromise as a failure mode. Deterministic, auditable, framework-agnostic.

Where Outerlimit telemetry overlaps with AI-SPM (agent inventory, tool inventory, risk scoring), discovery is sequenced rather than static: it begins as a read-only mapping of agents, action surfaces, and exposed tools, and deepens as action-surface integrations are deployed to provide complete visibility of all agentic-to-tool execution paths. Where pure-play AI-SPM stops at risk surfacing, Outerlimit's discovery capabilities quantify risk and present a remediation path to enforcement, ensuring agents never act outside of scope.

Outerlimit can pair with prompt-layer and adversarial-testing vendors, though we do not compete in those layers.

Outerlimit is currently working with an exclusive group of design partners ahead of general availability. If you are deploying agents in production and want to evaluate tool-execution-layer enforcement against your specific environment, we would welcome the conversation.

Outerlimit provides a universal security layer for Agentic AI, offering a single control and observability plane across your agentic ecosystem.

As enterprises evolve their adoption of AI and give agents access to tools via APIs, databases and MCP servers, exponential risk is introduced into their environment. Shadow AI and the unpredictability of agents create new attack surfaces and vulnerabilities. To quantify and manage this risk, the ability to discover and control an expanding agentic footprint becomes critical.

Outerlimit closes this visibility and control gap, enabling enterprises to discover and observe their agentic activity, quantify risk, and enforce run-time guardrails at the tool-execution layer.