

Straiker Raises $64 Million Series A on June 29—Agentic Security Startup Grows Revenue 15× in Under a Year Protecting Fortune 500 AI Agents

A security startup protecting AI agents from AI agents just grew revenue 15× in under a year. The category didn't exist 24 months ago—now Fortune 500 companies treat it as critical infrastructure.

The News: $64 Million for a Problem That Materialized Overnight

Straiker announced a $64 million Series A on June 29, 2026, bringing total funding to $85 million. Marathon Management Partners led the round, joined by Citi Ventures, Illuminate Financial, and Workday Ventures.

The numbers tell a story more compelling than any pitch deck: 15× run-rate revenue growth in less than a year since the platform launched in March 2025. CEO Ankur Shah didn't sugarcoat the trajectory: **"Demand is outpacing anything we forecast."**

That's not startup hyperbole. It's a signal that enterprise security teams are scrambling to solve a problem they couldn't have anticipated two years ago. AI agents—autonomous systems that take actions, not just generate text—are now embedded in procurement workflows, customer service pipelines, and financial operations across the Fortune 500. And nobody built security architectures for software that makes its own decisions.

Straiker's platform does three things: discovers AI agents running across enterprise environments (including shadow deployments teams didn't know existed), conducts adversarial testing before deployment, and provides real-time threat protection once agents go live. The company counts Fortune 500 enterprises and frontier AI labs among its customers, with named clients at launch including People.ai, Coupa Software, and DirecTV.

Why This Matters: The Agent Attack Surface Nobody Planned For

Traditional application security assumes software follows instructions. You secure the inputs, validate the outputs, lock down the APIs, and call it done. AI agents break every one of those assumptions.

An agent doesn't just process a request—it interprets intent, decides on a course of action, potentially invokes other tools or agents, and executes multi-step workflows autonomously. The attack surface isn't a single input field. It's every decision boundary the agent crosses during execution.

Consider what happens when an adversarial prompt convinces your customer service agent to offer unauthorized refunds. Or when a competitor's agent probes your procurement system for pricing thresholds. Or when a compromised agent in your supply chain workflow starts exfiltrating transaction data disguised as routine API calls.

These aren't theoretical risks. They're the reason Straiker grew 15× while most enterprise software companies measure growth in single-digit percentages.

The timing of this funding round isn't coincidental. The White House issued a new AI security order in late June 2026 that requires the NSA to develop classified benchmarking for frontier AI models. The order mandates developers provide the Federal Government with up to 30 days of pre-release access to "covered frontier models."

When the government starts requiring pre-release security audits, the market is sending an unmistakable signal: AI security is no longer optional, and the existing security stack isn't sufficient.

The Winners and Losers

Winners:

- **Purpose-built agentic security vendors** — Straiker, and the competitors that will inevitably emerge, occupy a category with zero incumbent advantage. Palo Alto Networks, CrowdStrike, and other security giants built their architectures for deterministic software. Retrofitting them for probabilistic, autonomous systems is a multi-year engineering challenge.
- **Enterprises deploying agents cautiously** — Companies that invest in pre-deployment adversarial testing will avoid the breach headlines that are coming for organizations treating agents like traditional applications.
- **Frontier AI labs** — Straiker explicitly works with frontier labs to gain early visibility into emerging attack techniques. Those partnerships create a moat: when your security vendor sees tomorrow's exploits today, your agents get patched before the attack is weaponized.

Losers:

- **Traditional SAST/DAST vendors** — Static and dynamic application security testing tools examine code paths and input validation. Agent vulnerabilities emerge from reasoning patterns, tool orchestration sequences, and emergent behaviors that don't exist in the codebase. The tooling gap is architectural, not incremental.
- **Enterprises treating agent security as a fine-tuning problem** — The instinct to "just add safety training" misunderstands the threat model. Adversarial prompts don't care about your RLHF alignment. They exploit the gap between what the agent was

trained to refuse and what it can be manipulated into doing.

- **Security teams without AI expertise** — The skills required to assess agent vulnerabilities look nothing like traditional penetration testing. Red-teaming an agent means understanding prompt injection vectors, tool-use exploitation, and multi-turn manipulation strategies. Most security teams don't have that bench strength yet.

Technical Depth: How Straiker's Architecture Works

Straiker's platform operates on a three-phase model that deserves examination because it represents what I believe will become the standard architecture for agentic security.

Phase 1: Discovery

Most enterprises don't know how many AI agents they're running. Shadow IT was a problem when employees installed unauthorized SaaS tools. Shadow AI is worse—engineers are deploying agents in side projects, testing frameworks, and automation scripts without security review.

Straiker's discovery layer scans enterprise environments to identify both sanctioned and unsanctioned agent deployments. This includes third-party agents embedded in vendor software, internally developed agents running in production, and experimental agents living in development environments that have production data access.

The discovery problem is harder than it sounds. Agents don't announce themselves. They look like API calls, webhook consumers, or background processes. Identifying them requires behavioral analysis: what's making autonomous decisions versus executing pre-defined logic?

Phase 2: Pre-Deployment Adversarial Testing

Here's where Straiker's technical differentiation becomes clear. The company claims to use "the industry's most comprehensive agentic exploit dataset" to power its testing engine.

Building that dataset required partnerships with frontier AI labs—the same labs developing the capabilities that create new attack vectors. When a new model architecture introduces

novel reasoning patterns, Straiker gets early access to understand how those patterns can be manipulated.

Pre-deployment testing isn't penetration testing. You're not looking for SQL injection or buffer overflows. You're probing for:

- **Prompt injection vulnerabilities** — Can external input override the agent's instructions?
- **Tool-use exploitation** — Can the agent be manipulated into misusing the tools it has access to?
- **Multi-turn manipulation** — Can a sequence of seemingly benign interactions lead to unauthorized actions?
- **Cross-agent attacks** — Can one agent compromise another agent it communicates with?
- **Reward hacking** — Can the agent be tricked into pursuing proxy objectives that diverge from intended behavior?

The testing engine generates adversarial scenarios calibrated to each agent's specific capabilities, tool access, and decision boundaries.

Phase 3: Runtime Threat Protection

Testing finds vulnerabilities. Runtime protection catches exploits in production.

Straiker's runtime layer monitors agent behavior in real-time, comparing actions against established baselines and policy constraints. When an agent starts behaving anomalously—executing tool calls it doesn't normally make, accessing data outside its typical scope, or following reasoning patterns that diverge from its training—the runtime engine intervenes.

The elegant architectural choice here is the **shared intelligence layer** connecting testing and runtime. Production threats detected by runtime protection get fed back into the adversarial testing dataset. Vulnerabilities discovered in testing get converted into runtime detection rules.

This creates a flywheel: more customers running in production means more threat data,

which means better testing, which means more effective runtime protection, which attracts more customers. In platform dynamics, this is the holy grail—a defensible competitive advantage that strengthens with scale.

What the Architecture Reveals About the Threat Model

Straiker's three-phase approach implicitly acknowledges a truth that many organizations haven't internalized: **agent security is not a deployment-time problem. It's a continuous lifecycle problem.**

You can't test an agent once and declare it secure. The threat landscape evolves as new attack techniques emerge. The agent's behavior may drift as it interacts with new data. The tools and systems the agent accesses may change in ways that create new vulnerabilities.

The feedback loop between pre-deployment testing and runtime protection is architectural recognition that agentic security must be dynamic. Static security assessments don't work when the thing you're securing is designed to adapt.

The Contrarian Take: What Most Coverage Gets Wrong

Overhyped: The "AI vs. AI" Narrative

The hook I led with—"protecting AI agents from AI agents"—is accurate but incomplete. The more significant threat isn't malicious agents attacking your agents. It's humans using AI-augmented techniques to exploit your agents at scale.

Automated prompt injection attacks, adversarial input generation, and AI-assisted social engineering represent the near-term threat landscape. Autonomous agent-to-agent warfare is a real but more distant concern.

Straiker's value proposition works against both threat vectors, but the coverage focusing on "AI attacking AI" misses the immediate danger: humans are already using AI tools to find and exploit agent vulnerabilities faster than defenders can patch them.

Underhyped: The Shadow Agent Problem

Straiker's discovery capability might be its most strategically important feature, yet it gets the least attention in coverage.

Enterprise IT teams spent the last decade fighting shadow IT—unauthorized SaaS applications proliferating across organizations. Shadow AI is the same problem amplified. Engineers can deploy an agent in an afternoon using off-the-shelf frameworks. That agent might have access to customer databases, financial systems, or authentication credentials depending on the environment it runs in.

You cannot secure agents you don't know exist. The discovery phase isn't a nice-to-have preliminary step. It's foundational to the entire security posture.

I expect Straiker's enterprise sales motion increasingly leads with discovery. The "how many agents are running in your environment right now?" question is uncomfortable for security teams to answer—and that discomfort drives urgency.

Misunderstood: The Relationship Between AI Labs and Security Vendors

Straiker's partnerships with frontier AI labs aren't just about early access to new models. They're about understanding attack vectors before they're weaponized in the wild.

When Anthropic or OpenAI discovers that a new architecture has unexpected prompt injection sensitivities, that information has time value. Security vendors who learn about vulnerabilities pre-release can build protections before attackers figure out the same exploits through black-box testing.

This is a fundamental shift from traditional security dynamics. Historically, vulnerability disclosure flowed from independent researchers to vendors under coordinated disclosure frameworks. In agentic security, the AI labs developing the capabilities are also the first to understand their failure modes.

The vendors who build deep relationships with frontier labs will have a structural intelligence advantage. Straiker's explicit positioning around these partnerships signals they understand the strategic importance.

Practical Implications: What You Should Do Now

For CTOs and Security Leaders

1. Audit your agent inventory immediately.

Before evaluating any security solution, you need visibility into what's running. Conduct an internal survey of AI agent deployments across engineering, operations, and business units. Include third-party vendor software that may embed agentic capabilities.

The question isn't whether you have shadow agents. The question is how many.

2. Classify agents by risk tier.

Not all agents require the same security posture. An internal documentation assistant with read-only data access is different from a procurement agent authorized to approve transactions.

Develop a classification framework based on:

- Data access scope (read-only vs. read-write, internal vs. customer data)
- Tool access (what actions can the agent take?)
- Autonomy level (human-in-the-loop vs. fully autonomous)
- External exposure (internal users only vs. customer-facing)

3. Implement pre-deployment adversarial testing as a gate.

No agent should reach production without adversarial evaluation. This doesn't require a vendor—open-source frameworks for prompt injection testing exist—but it does require making testing a mandatory pipeline stage.

If you're deploying agents faster than you can test them, you're accumulating security debt that compounds with every deployment.

4. Establish runtime behavioral baselines.

Even without dedicated agentic security tooling, you can instrument agents to log decision

paths, tool invocations, and data access patterns. Anomaly detection against those baselines catches exploits that evade prompt-level defenses.

For Engineers Building Agents

1. Design with adversarial inputs as the default assumption.

Every piece of external input—user messages, API responses, data from other agents—should be treated as potentially adversarial. This means:

- Separating instruction context from user input context
- Implementing output validation before tool execution
- Using structured output formats that constrain action spaces

2. Limit tool access to minimum required scope.

Agents should follow the principle of least privilege. If a customer service agent doesn't need database write access, don't grant it. If a scheduling agent doesn't need access to financial systems, don't connect them.

The blast radius of a compromised agent is proportional to its capability surface.

3. Implement circuit breakers for autonomous workflows.

Multi-step autonomous workflows should include checkpoints where execution pauses for validation. This might be automated policy checks or human-in-the-loop approvals depending on action sensitivity.

The goal is preventing a single compromised decision from cascading through an entire workflow.

Vendors to Watch

Straiker's funding round signals a category emerging, not a winner declared. Other vendors building in this space:

- **HiddenLayer** — Focused on ML model security, expanding into agentic use cases

- **Protect AI** — ML supply chain security with growing attention to runtime threats
- **Lakera** — Prompt injection defense, building toward broader agent security

The category is early enough that vendor selection should prioritize depth of adversarial testing capabilities and quality of threat intelligence over feature checklists. Ask vendors where their exploit datasets come from and how frequently they update.

Forward Look: Where This Goes in 12 Months

Prediction 1: Agentic security becomes a procurement requirement

Enterprise procurement questionnaires already include SOC 2, penetration testing, and data handling requirements for vendors. Within 12 months, expect questions about AI agent security posture to become standard.

"How do you secure AI agents in your platform?" will join "How do you handle customer data?" as a deal-blocking procurement question.

Prediction 2: The first major agent-related breach makes headlines

Straiker's 15× revenue growth indicates enterprises are buying proactively. But the broader market typically responds to incidents, not threat forecasts.

The first major breach attributed to agent exploitation—a compromised procurement agent authorizing fraudulent transactions, a customer service agent leaking PII at scale, a financial agent manipulated into erroneous trades—will accelerate adoption curves by 3-5× beyond current projections.

That breach is coming. The attack surface is too large and the defenses too immature for it not to happen.

Prediction 3: Traditional security vendors acquire or build

Palo Alto Networks, CrowdStrike, Microsoft, and other major security vendors cannot cede a critical emerging category to startups. Expect acquisition activity or significant internal development announcements within 12 months.

Ankur Shah's background is relevant here—he scaled Palo Alto Networks' Prisma Cloud business as SVP and GM. He knows the playbook for building categories that eventually get absorbed by platform vendors. Straiker's speed suggests they're racing to establish market position before the incumbents mobilize.

Prediction 4: Regulatory frameworks crystallize

The White House AI security order is a leading indicator, not an outlier. EU AI Act implementation continues through 2026, with high-risk AI system requirements that will encompass many agentic deployments.

Security vendors who can demonstrate compliance with emerging frameworks—through auditable testing processes, documented threat detection capabilities, and certified runtime protections—will have significant competitive advantages in regulated industries.

Prediction 5: The talent market inverts

Today, AI security expertise is scarce and expensive. Organizations struggle to staff red teams capable of adversarial agent testing.

Within 12 months, expect the emergence of AI-augmented security tools that enable traditional security professionals to conduct agent assessments. The expertise gap won't close—it will be automated around.

The Broader Pattern

Straiker's funding round represents something larger than one startup's success. It marks the formal emergence of a security category that mirrors the pattern we've seen with every major infrastructure shift.

Cloud computing created cloud security. Mobile computing created mobile security. API-first architecture created API security. Agentic AI is creating agentic security.

The pattern is consistent: new computing paradigms initially inherit security approaches from their predecessors, those approaches prove insufficient for novel threat models, purpose-built solutions emerge, the market consolidates around winners, and eventually

the capability gets absorbed into platform offerings.

Straiker is racing through the early stages of that pattern. The $85 million in total funding buys runway to establish market position before the inevitable platform consolidation.

For organizations deploying AI agents, the strategic calculus is straightforward. You can wait for the breach headlines and regulatory mandates that force action. Or you can recognize that 15× revenue growth in a security category reflects genuine enterprise pain and get ahead of the problem.

The agents are already in your environment. The question is whether you know what they're doing.

The organizations that treat agentic security as infrastructure rather than insurance will be the ones still operating agents when the first major breach reshapes the market.



WRITTEN BY

Artur Markus

Software engineer in Basel building AI infrastructure and agentic systems, with a background in pharmaceutical automation, GxP compliance, and validation. I help teams turn AI from a chatbot into a reliable operational layer.

Book a meeting