ClawSecure

The AI Agent Threat Report

The Responsibility Gap

Who is accountable for securing AI agents

Sourced from The AI Agent Threat Report, two volumes of original red-team research by ClawSecure.

Authored by

J.D. Salbego

Founder and CEO, ClawSecure

with the ClawSecure RedTeam

Date

September 2026

Contact

Research inquiries: research@clawsecure.ai

Press and media: media@clawsecure.ai

All other inquiries: contact@clawsecure.ai

The AI labs made prompt injection YOUR problem. There are 123 million of you already building on AI.

We reported prompt injection through OpenAI's API via their official security intake, which operates under OpenAI's own scope policy. Their blunt answer: these guardrails are **the developer's responsibility**.

What OpenAI's official security intake told us, verbatim

> "We are interested in issues like these, but not those that rely entirely on API access, as implementing the appropriate guardrails in that context is the developer's responsibility.
> If you are able to demonstrate data exfiltration through the ChatGPT UI, please feel free to create a new submission along with a video PoC which proves your claim."

They are not fixing it. They are handing it to you.

Google's security team told us the same thing, in different words

Google reviewed the same class of finding and confirmed the behavior reproduces. They were explicit that they consider indirect prompt injection leading to data exfiltration **"a valid and serious security risk."** Then they described where their protection actually sits.

> "Google's primary defense against this specific exfiltration primitive is enforced at the rendering layer. Products like the Gemini web app utilize output sanitization that actively strips or neutralizes unauthorized markdown images and links before they are ever rendered to the user."

> "Because your payload was tested against a raw endpoint and did not bypass these downstream client-side sanitizers, the GET request could not fire, and the exfiltration impact **was not realized in our flagship applications**."

And then, in the same message, the instruction to everyone building on the API:

> "We advise all integrators building agents on our models to treat tool-result channels as untrusted and to implement strict output sanitization (stripping or neutralizing unauthorized markdown image tags) before rendering model outputs."

Read those together and the line is drawn explicitly. The defense is in the web app. The web app is the flagship application. The raw API endpoint sits outside it, and the developers building agents there are told to implement the sanitization themselves.

That same API is the product they market hardest. The public pages sell these models to developers building production agents, positioned as the newest and most capable available, with nothing anywhere indicating that the protection described above stops at the web app.

None of this is stated publicly, and we went looking. We captured and SHA-256 hashed four of the provider's own public pages covering these models and their safety posture. Across all four, the words "injection" and "indirect" appear **zero times**.

What those pages do contain is thinner than the missing term. They carry a safety heading and the assurance that the products are built with safety in mind. Follow that through to the safety material itself and the substance is a single sentence about continuously improving safeguard coverage, plus a scope limited to chemical, biological, radiological, nuclear and cyber-offense misuse. There is no account of what protects an ordinary user of these models, no mention of the class we reported, and nothing to suggest the defense they described to us privately does not extend past their own web application.

The scoping is real, it is written down, and it exists only inside a private disclosure thread.

Two labs. Two different answers. The same result. OpenAI's official security intake told us that securing this over the API is the developer's responsibility. Google told us that securing the API surface is the integrator's responsibility. Different words, different tone, one outcome: **the party that builds on the API is the party left to defend it.**

Neither of them says so publicly. Both statements exist only because we filed a coordinated disclosure and then pressed, repeatedly and over weeks, to get a direct answer. A developer who never files one has no way to learn where the protection stops.

The developer is now 123 million people already using AI

"The developer" used to mean a professional engineer at a company with a security team. It does not mean that anymore. These are people already building and running on AI today, and they inherit the security obligation the moment they build on the API.

Who is already building on AI, and who now inherits the responsibility	How many
Small development teams building with AI	**40M+**
No-code and vibe-coding AI builders	**70M+**
Small and medium businesses using AI	**13M+**
Total	**123M+**

Almost none of them have a security team. Most have never heard of prompt injection. All of them are now responsible for stopping it.

The handoff does not stop at individuals

The people told to build their own AI guardrails are also the platforms that tens of millions of people build on top of.

- **The no-code and AI build platforms** that tens of millions of people build on are themselves API consumers. They inherit the same responsibility, and pass the exposure to every user downstream.
- **Enterprise API consumers** are in the same position. Anything wired to the API layer inherits the guardrail obligation, no matter how large the company is.
- **Desktop agent tooling** that runs against the API rather than a consumer UI sits on the same side of the line.

This is not a niche developer problem. It is the default configuration of the modern AI stack.

(We did not test any third-party build platform. They are referenced as API consumers, not as tested targets.)

This is not theoretical. We measured what happens next.

- **14 models across 5 labs**, one identical hardened attack, **zero clean defenders**
- The best-defended model on the market still obeyed hidden attacker commands **about one in four times**
- **3 top production MCP platforms cracked end to end**, with the gap in the protocol itself, not in any one company's code
- The most aligned model on the market **read a live AWS credential off a disk and shipped it to an attacker on its own**, then reported that everything was fine

The vulnerability is real, it is measured, and responsibility for stopping it has been handed to people who cannot carry it.

Someone has to hold this line

The fix cannot be the AI's own judgment, because the best-aligned model on the market shipped a real credential to an attacker while believing it was doing honest work. **And it cannot be 123 million people each building their own defenses.**

Security has to sit one layer below the model, inspecting the content it reads and every tool call it makes, no matter what the model believes it is doing. **That is the gap ClawSecure closes:** the only end-to-end AI agent security platform built for exactly the people who just inherited this problem, non-technical users, prosumers, small dev teams, and SMBs with zero security staff.

The full two-volume AI Agent Threat Report, the reproductions, and the raw evidence are available on request.

Market sources: SlashData/JetBrains 2026; SBA 2024; company reports 2026. Segment figures are multi-source and hard-sourced. Partial cross-platform overlap exists, which is why the filtered, conservative framing is used.