SALT

The State of Agentic AI and API Security

2H 2026 · The Agentic Path Gap



The Agentic Path Gap

Enterprises are deploying AI agents faster than they can see, govern, or understand what they're doing, according to a survey of over 380 security and technology professionals and Salt Labs production telemetry.

Table of Contents:

2 – 3 · Executive Summary: The Agentic Path Gap

4 – 6 · AI Agents Move from Experimentation to Operational Infrastructure

7 – 11 · The Path Visibility and Control Gap

12 – 14 · MCP Expands the Agentic Connectivity Layer

15 – 19 · AI Coding Assistants Shift Security Earlier, but Not Early Enough

20 – 22 · APIs Remain the Foundation of the Agentic Enterprise

23 – 25 · API Security Pressure Persists Despite Maturing Programs

26 – 28 · Salt Labs: What Real-World Attack Telemetry Shows

29 – 31 · Recommendations for CISOs and Application Engineering Leaders

32 · Conclusion: Secure What Agents Do

33 – 34 · Methodology and Research Notes

35 · About Salt Security

Executive Summary

The Agentic Path

Agentic AI has moved from experimentation into the daily operations of the enterprise, and it is moving faster than most organizations can govern. AI agents are no longer models that only produce text. They now act: reaching enterprise data, invoking tools and other agents, calling APIs, and triggering real business processes, often chaining these steps together in sequences no one designed in advance.

That shift has opened a gap between how quickly agents are being deployed and how well organizations can see, govern, and stop them. Across our 2H 2026 survey of over 380 security and technology professionals and Salt Labs production telemetry, a consistent picture emerges: agents are widespread and increasingly autonomous; they are already taking actions their operators did not intend, and most organizations cannot yet trace or contain what their agents do. The findings below quantify that gap, and point to what closing it requires.

Agents are already doing things no one authorized

- **Nearly half (49.8%)** confirmed or suspected an AI agent took an action in the past year the organization did not intend, expect, or authorize.
- **Of those, 53.9%** involved an external API call or transaction, **41.9%** modified or deleted data, and **32.5%** reached data the agent should not have had.

Most organizations can't see what their agents do

- **Only 12.5%** can consistently trace an agent's full path, from instruction through tools and APIs to outcome, across all agents.
- **Just 15.7%** could reconstruct everything an agent did yesterday in minutes; **64.5%** would need hours or days of manual work.
- **Only 17.5%** can automatically stop an agent that starts behaving unexpectedly in production. Most can intervene only manually, or by taking the application down.

Executive Summary

Agents are everywhere, and they act on their own

- **86.4%** report at least one AI agent operating in their organization; **59.5%** run more than 10.
- **76.8%** say their agents invoke other agents or tools that themselves call APIs, chaining actions across systems.

The connective layer is becoming an attack surface

- **32.9%** have found an unauthorized, shadow or misconfigured MCP server in the past year; **54.3%** have found or suspect one.
- Salt Labs sees the same trend in the field: MCP-related infrastructure appeared in **21.6%** of customer environments in August 2026, **up from 16.7%** in January, a roughly **30% relative increase**.
- Agentic-connected API endpoints are already being attacked in **roughly a quarter (26.3%)** of the environments where Salt Labs studied them, an early but directional signal.

AI-generated code is now a live security problem

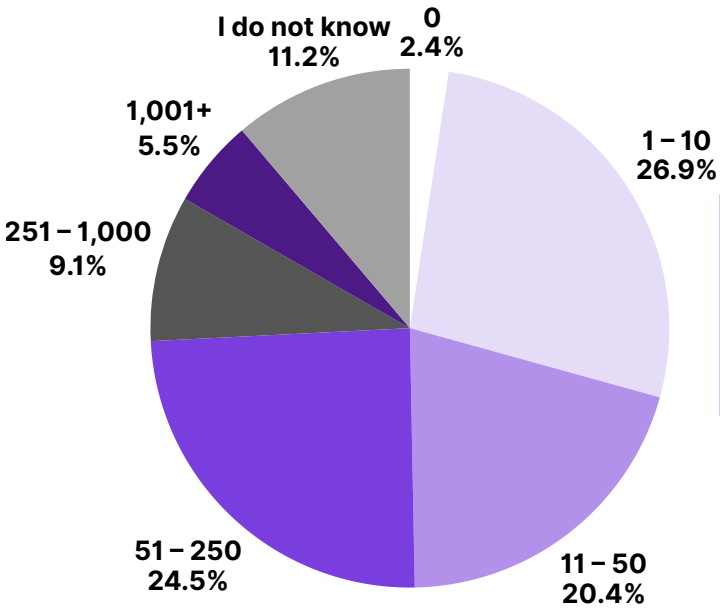
- **83.5%** estimate AI-generated or AI-assisted code accounts for at least some of their application security findings.
- Yet only **22.2%** enforce security and compliance policy across all approved AI coding assistants, and just **6.8%** resolve issues in the assistant at generation time.

The bottom line

Agents are being trusted to act before organizations can see or govern what they do. Closing that gap means knowing which agents exist, what they can reach, and what they are permitted to do, then being able to trace and stop them at machine speed. That is the discipline of agentic security, and it is what lets enterprises give agents autonomy without inheriting unmanaged risk.

AI Agents Move from Experimentation to Operational Infrastructure

The survey indicates that AI agents are no longer isolated proofs of concept. Only **2.4%** report zero agents operating in their organization; **86.4%** report at least one, while **11.2%** do not know. **26.9%** report 1–10 agents, **20.4%** report 11–50, **24.5%** report 51–250, **9.1%** report 251–1,000, and **5.5%** report more than 1,000.



Approximately how many AI agents are currently operating within your organization?

The distribution matters because governance complexity rises with the number of autonomous identities and the systems they can reach. At small scale, security teams can rely on manual reviews and owner knowledge. At dozens or hundreds of agents, controls need to become inventory-driven, identity-aware, and automated.

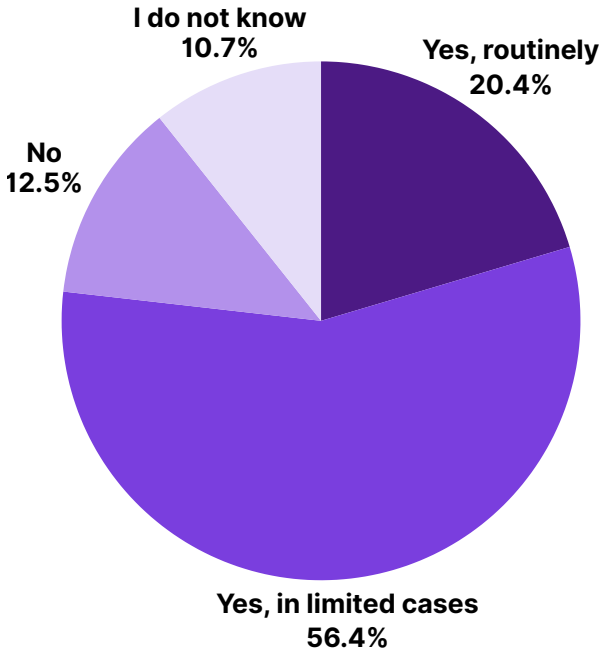
AI Agents Move from Experimentation to Operational Infrastructure

Agents increasingly act through other agents and tools

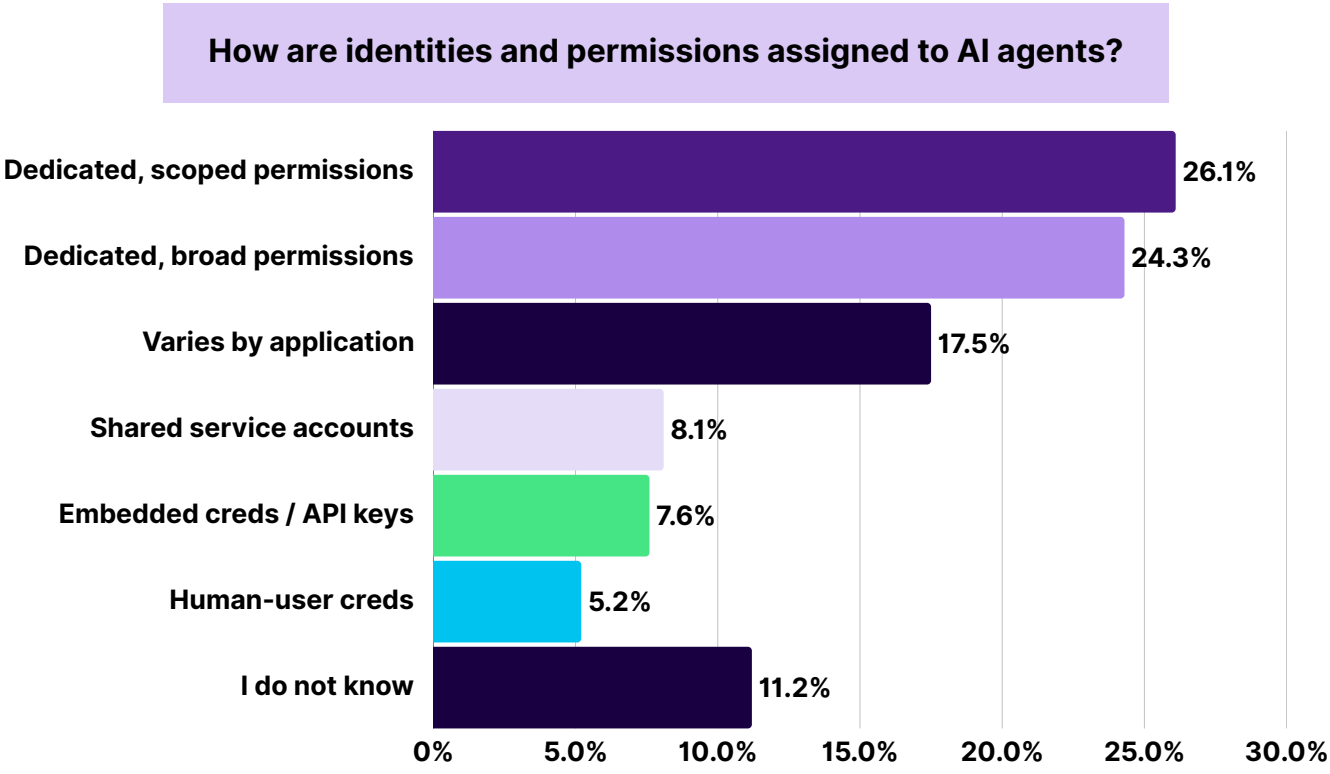
20.4% say their AI agents routinely invoke other agents or tools that themselves call APIs, and **56.4%** say this happens in limited cases. Together, **76.8%** report some degree of chaining. Only **12.5%** say their agents do not behave this way, while **10.7%** do not know.

This is the architectural reason path visibility matters. The original prompt or instruction may be several steps removed from the API request that produces the final business effect. Security controls need to preserve context across those hops rather than evaluating each request as an isolated transaction.

Do your AI agents invoke other agents or tools that themselves call APIs?



AI Agents Move from Experimentation to Operational Infrastructure



Identity is a first-class agentic security control

Only **26.1%** say agents are assigned dedicated identities with scoped permissions. Another **24.3%** use dedicated identities but with broad permissions. **8.1%** use shared service accounts, **5.2%** use human-user credentials, **7.6%** rely on embedded credentials or API keys, **17.5%** say the approach varies by application, and **11.2%** do not know.

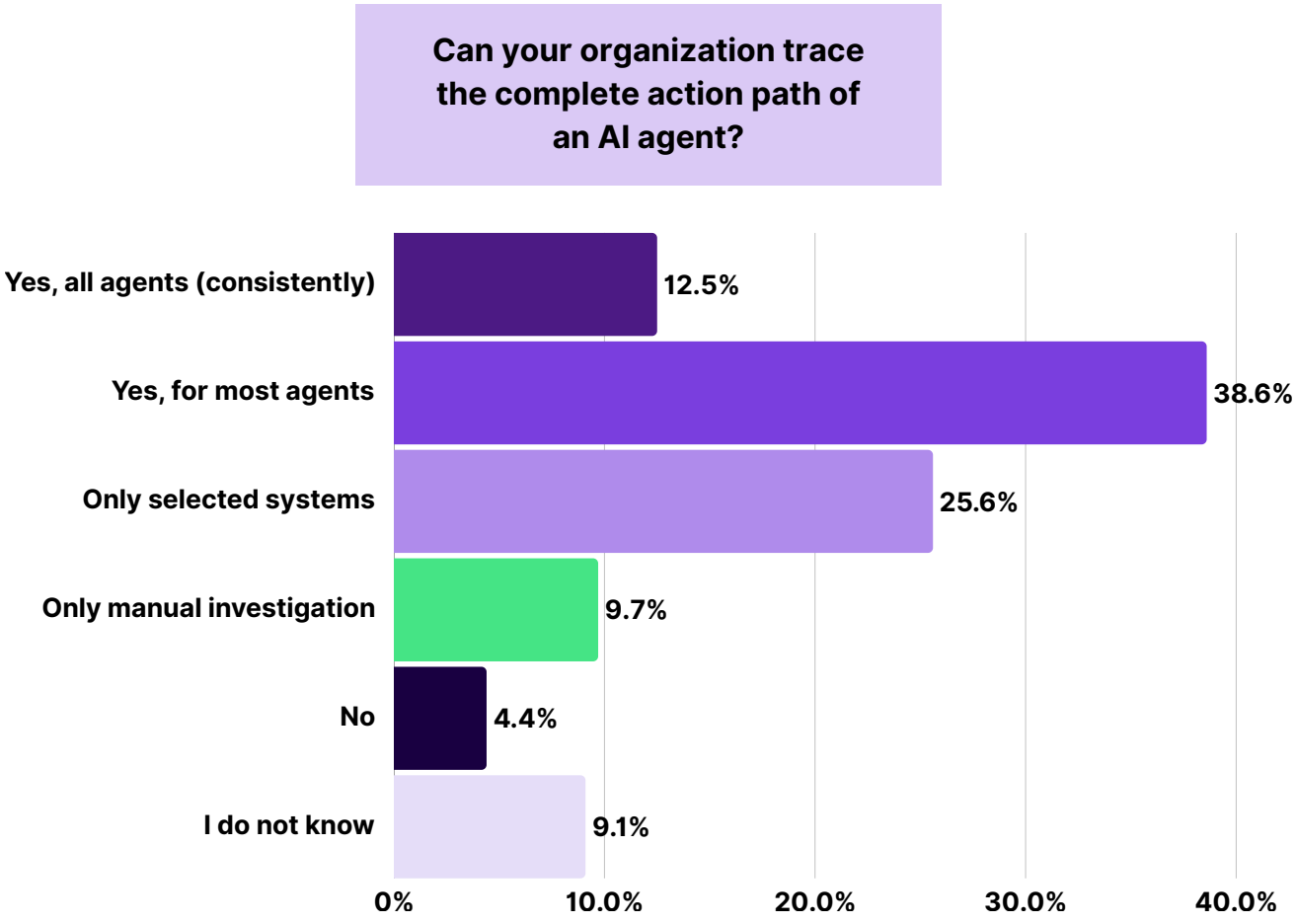
Dedicated identity is necessary but not sufficient. A uniquely identifiable agent with excessive privilege can still produce high-impact outcomes. The stronger pattern is a dedicated non-human identity, narrowly scoped permissions, short-lived or revocable credentials where feasible, an accountable owner, and policy checks tied to the sensitivity of the requested action.

The Path Visibility and Control Gap

Agentic security becomes operational when organizations can answer a simple forensic question: what did this agent do? The survey suggests that many teams can answer part of that question, but fewer can answer it completely and quickly.

Complete traceability is not yet the norm

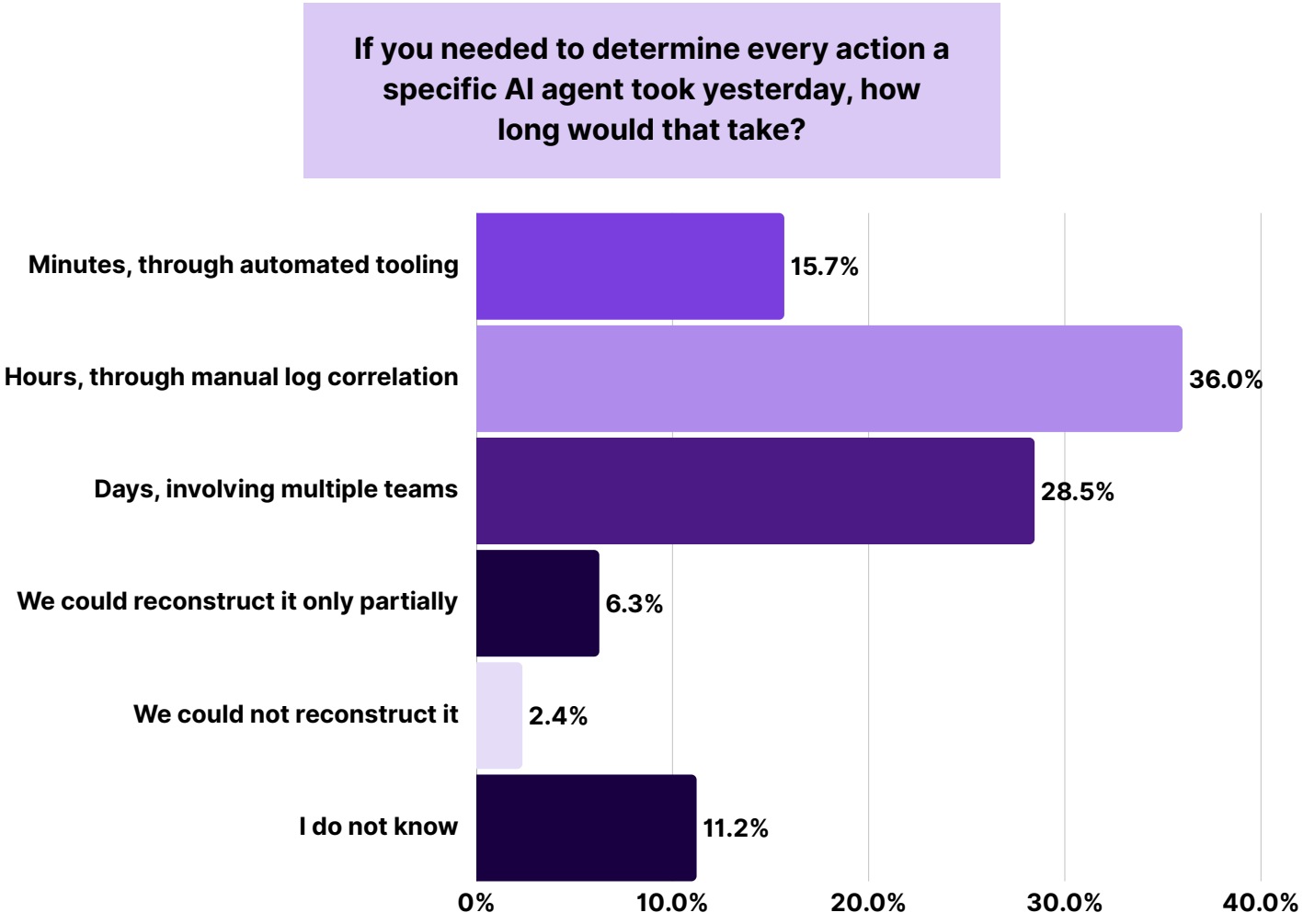
12.5% can consistently trace the complete path an agent takes across all agents, from the initial instruction or user intent through agent decisions, MCP servers and tools, API calls and the resulting outcome. **38.6%** can do so for most agents. **25.6%** have this visibility only for selected agents or systems, **9.7%** can reconstruct it only during manual investigations, **4.4%** cannot, and **9.1%** do not know.



The Path Visibility and Control Gap

When the question is framed around speed, the gap becomes clearer. Only **15.7%** could determine every action a specific AI agent took yesterday in minutes using automated tooling. **36%** would need hours of manual log correlation, and **28.5%** would need days and multiple teams. A further **6.3%** could reconstruct the sequence only partially, **2.4%** could not reconstruct it, and **11.2%** do not know.

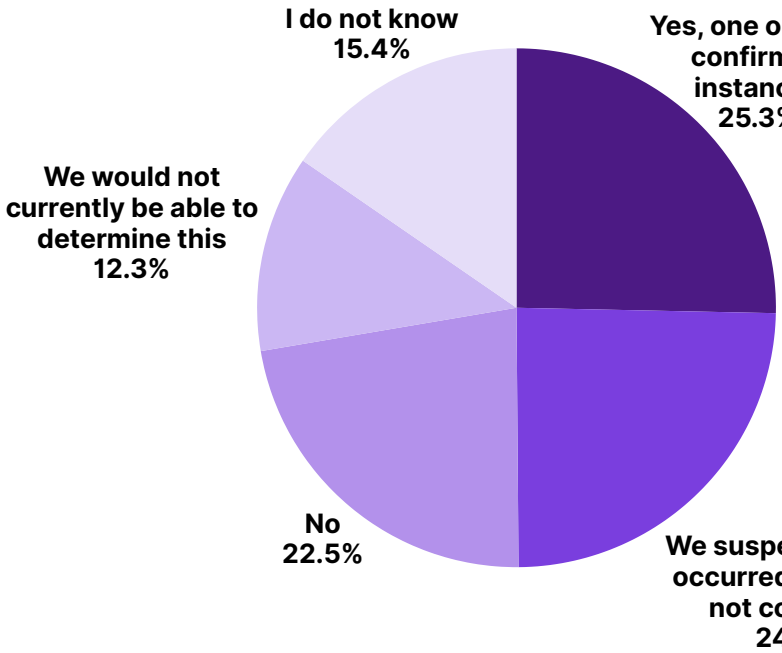
For incident response, hours can be a long time when the actor is software. An agent can repeat an action, invoke multiple services, change configurations, or transact without waiting for a human work cycle. Organizations therefore need telemetry that links agent identity and initiating intent to downstream tool and API activity in near real time.



The Path Visibility and Control Gap

Accountability is stronger than observability, but still incomplete
20.9% can identify the human or team accountable for actions taken by all agents, and **51.2%** can do so for most agents. **14.1%** have ownership mapped only for selected agents or systems, **3.9%** cannot identify an accountable party, and **9.9%** do not know. This is encouraging relative to the visibility data, but accountability without traceability can still leave responders unable to establish exactly what happened.

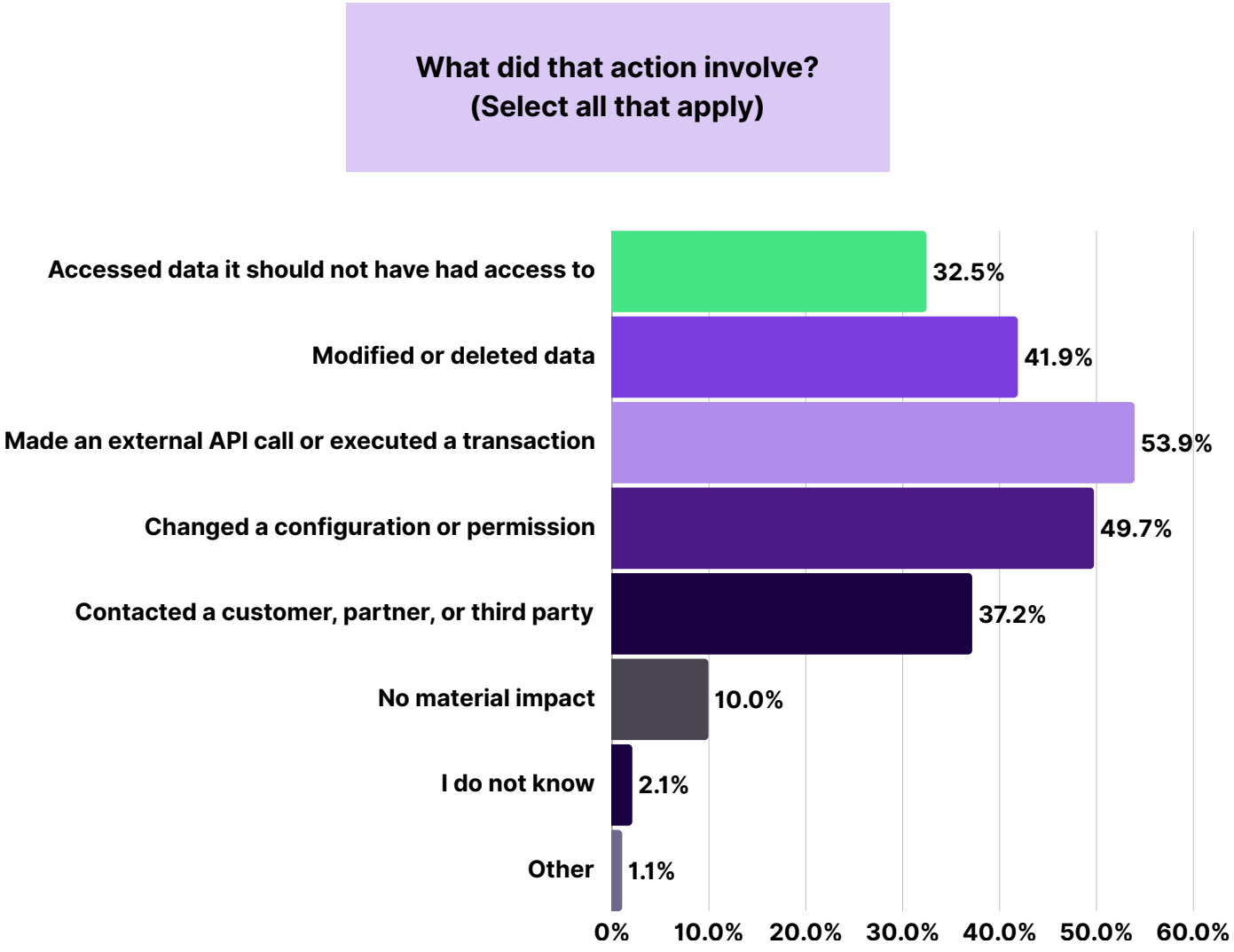
Unintended action is already a reported operational event
25.3% report one or more confirmed instances in which an AI agent took an action the organization did not intend, expect, or explicitly authorize during the past 12 months. Another **24.5%** suspect that this occurred but could not confirm it. Only **22.5%** say it did not occur; **12.3%** say they would not currently be able to determine whether it occurred, and **15.4%** do not know.



In the past 12 months, has an AI agent taken an action that your organization did not intend, expect, or explicitly authorize?

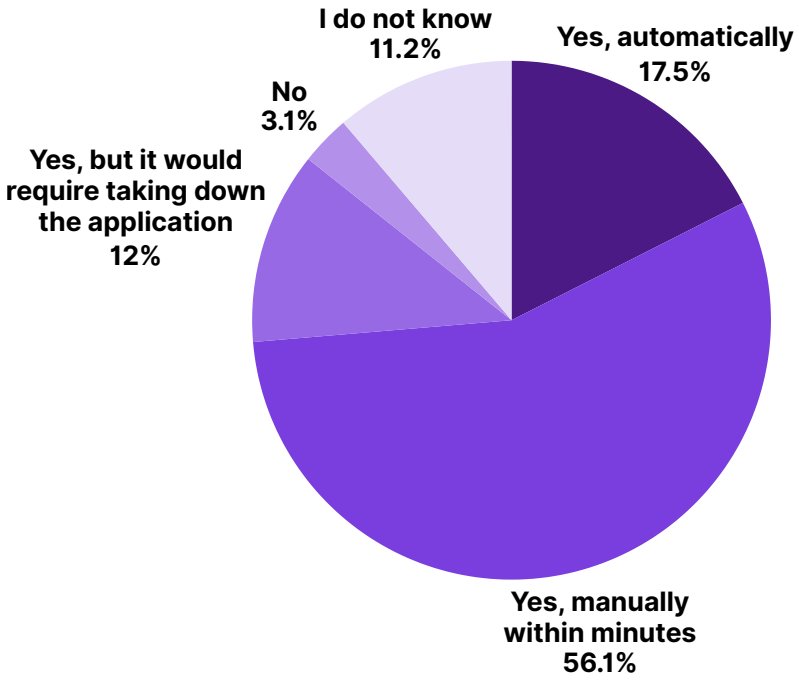
The Path Visibility and Control Gap

The follow-up responses show why this matters. Among the respondents with a confirmed or suspected incident, **53.9%** report an external API call or transaction, **49.7%** a configuration or permission change, **41.9%** modified or deleted data, **37.2%** contact with a customer, partner or third party, and **32.5%** access to data the agent should not have had. Only **10%** say there was no material impact. Because respondents could select multiple outcomes, these percentages are not mutually exclusive.



The Path Visibility and Control Gap

Containment remains human-speed for most organizations
17.5% can automatically stop an agent that begins behaving unexpectedly in production. **56.1%** can stop it manually within minutes. **12%** can stop it only by taking down the application, **3.1%** cannot stop it, and **11.2%** do not know. The result is a meaningful control asymmetry: **85.6%** have some intervention path, but only a small minority can automate that response.



If an AI agent began behaving unexpectedly in production right now, could you stop it mid-action?

For high-impact actions, the more resilient architecture is preventive and layered: least privilege at the agent identity, explicit policy at the tool and API boundary, step-up human approval for sensitive operations, transaction limits where appropriate, auditable action records, and rapid credential or session revocation. A kill switch is useful, but it should not be the only control between anomalous intent and an irreversible action.

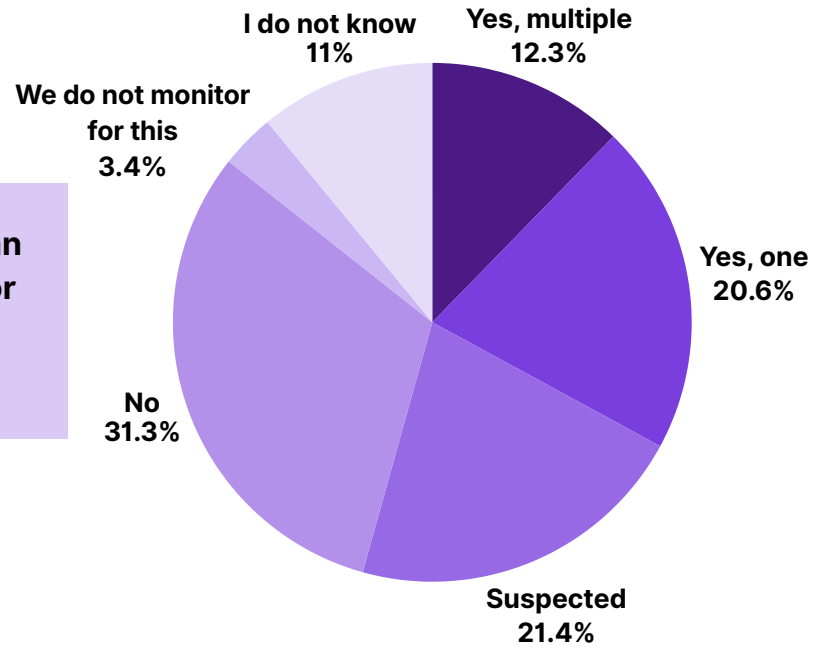
MCP Expands the Agentic Connectivity Layer

MCP gives AI applications a standardized way to discover and use external capabilities. MCP servers can expose tools, resources and prompts, and those tools may in turn reach enterprise APIs, databases, or services. This makes MCP important from a security perspective not because the protocol is inherently unsafe, but because it can concentrate powerful capabilities behind an interface an agent can invoke.

Shadow and misconfigured MCP infrastructure is already being found

12.3% of respondents discovered multiple unauthorized, shadow, forgotten, or misconfigured MCP servers during the past 12 months, while **20.6%** discovered one. Another **21.4%** suspected such a server. Combined, **32.9%** have confirmed at least one and **54.3%** have confirmed or suspected one. **31.3%** report no discovery, **3.4%** do not monitor for this condition, and **11%** do not know.

Has your organization discovered an unauthorized, shadow, forgotten, or misconfigured MCP server in the past 12 months?



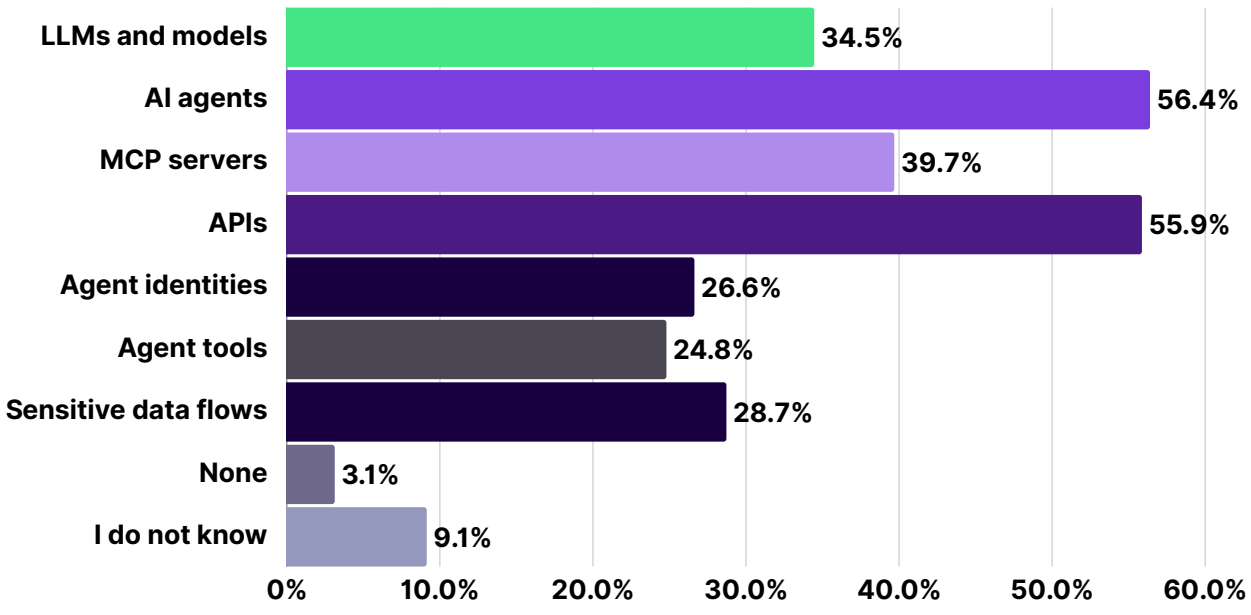
MCP Expands the Agentic Connectivity Layer

Exposure alone should not be treated as proof of vulnerability. A deliberately remote MCP server with strong authentication, authorization, and constrained tools can be appropriate. The risk comes from unknown ownership, excessive tool capability, weak credentials, public reachability where it is not intended, or an inability to understand which downstream APIs and data the server can access.

Dedicated controls are uneven across the agentic stack

Respondents report the strongest dedicated control coverage for AI agents (**56.4%**) and APIs (**55.9%**). Coverage falls to **39.7%** for MCP servers, **34.5%** for LLMs and models, **28.7%** for sensitive data flows, **26.6%** for agent identities and **24.8%** for agent tools. This distribution is notable because identity and tool permissions are precisely where an agent gains the ability to act.

Which parts of your organization's agentic technology stack are currently covered by dedicated security controls? (Select all that apply)



MCP Expands the Agentic Connectivity Layer

Salt Labs sees MCP prevalence increasing in customer environments

As of August 2026, Salt detected MCP-related API infrastructure in **21.6%** of customer environments, up from **16.7%** at the start of the year, a roughly **30%** relative increase in detected prevalence. Across the same environments, Salt identified thousands of MCP servers and AI agents and tens of thousands of agentic-connected APIs, indicating that agentic infrastructure is now a routine part of the production estate.

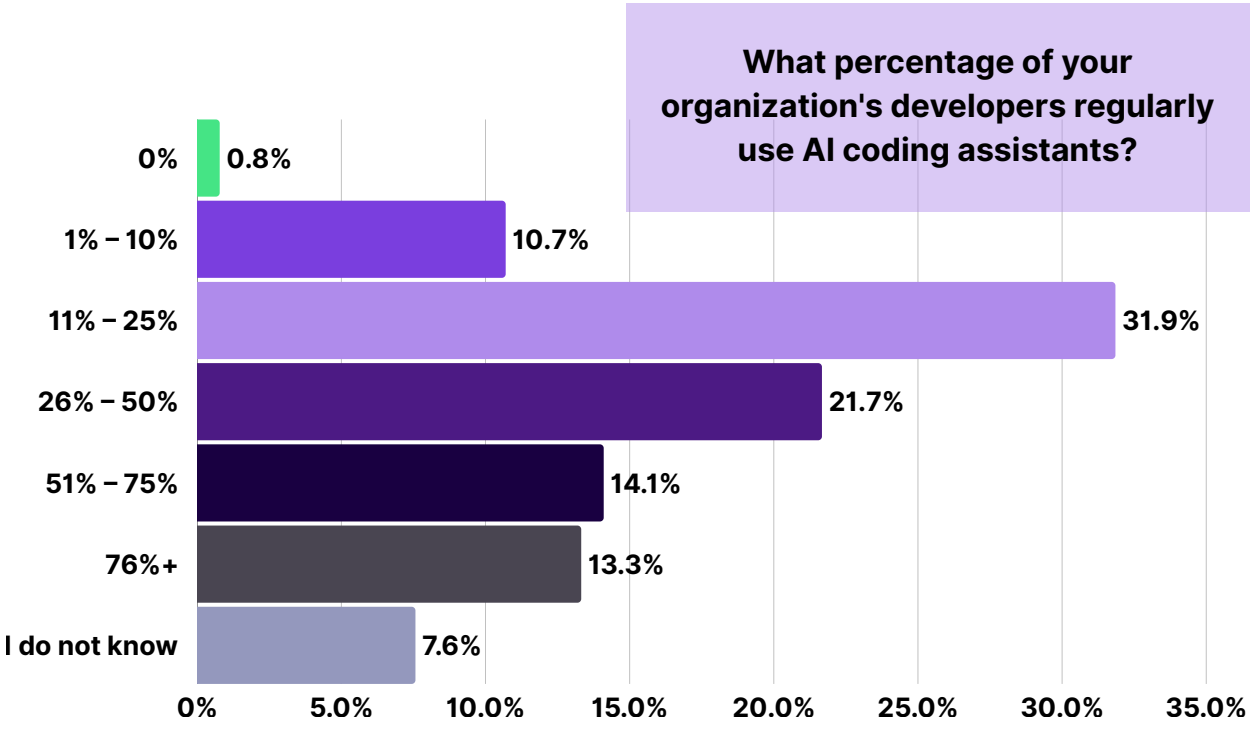
Because the customer population expanded between measurements, this is best described as a change in detected prevalence rather than growth within a fixed cohort. It nevertheless provides a useful empirical signal that MCP and agentic infrastructure are becoming more visible in production environments.

Early attack correlation: directional, but important

Salt Labs experimentally correlated violation records against the inventory of agentic-connected APIs. In **26.3%** of the customer environments with agentic-connected APIs, attacker activity was observed on those paths. Because this is an early, experimental analysis across a limited set of environments, the result should be treated as directional rather than as a market-wide prevalence estimate. It establishes that endpoints connected to agentic infrastructure are already intersecting with real attacker activity, not that this is a hypothetical future risk. As coverage expands, this metric is worth tracking over time.

AI Coding Assistants Shift Security Earlier, but Not Early Enough

AI coding assistants have become a normal part of software engineering workflows. **91.6%** of respondents estimate that at least some developers regularly use them, excluding respondents who selected zero or did not know. **81%** estimate that more than **10%** of developers use them, **49.1%** estimate more than **25%**, and **27.4%** say more than half of developers do.

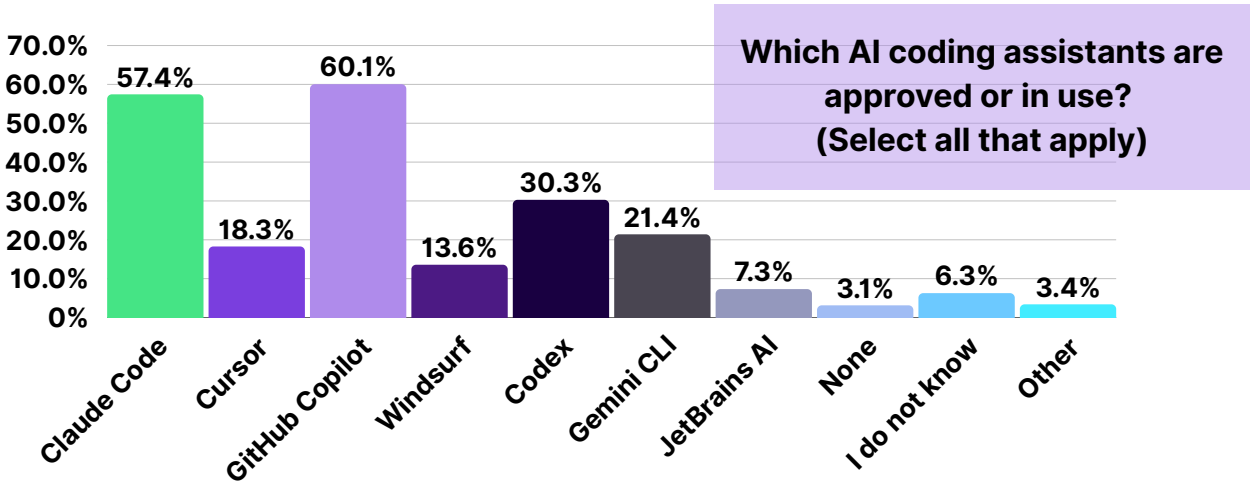


The gap between functional code and secure code

The deeper risk is the gap between code that works and code that is secure. According to Veracode's 2026 GenAI Code Security Report, AI models produce syntactically valid code **99%** of the time, yet introduce a known security vulnerability in roughly **44%** of code-generation tasks when given no security-specific guidance, a security pass rate that has barely moved in two years (Deng, Fan & Meng, 2026). Code that compiles and looks clean invites less scrutiny, which is precisely how vulnerabilities reach production unnoticed.

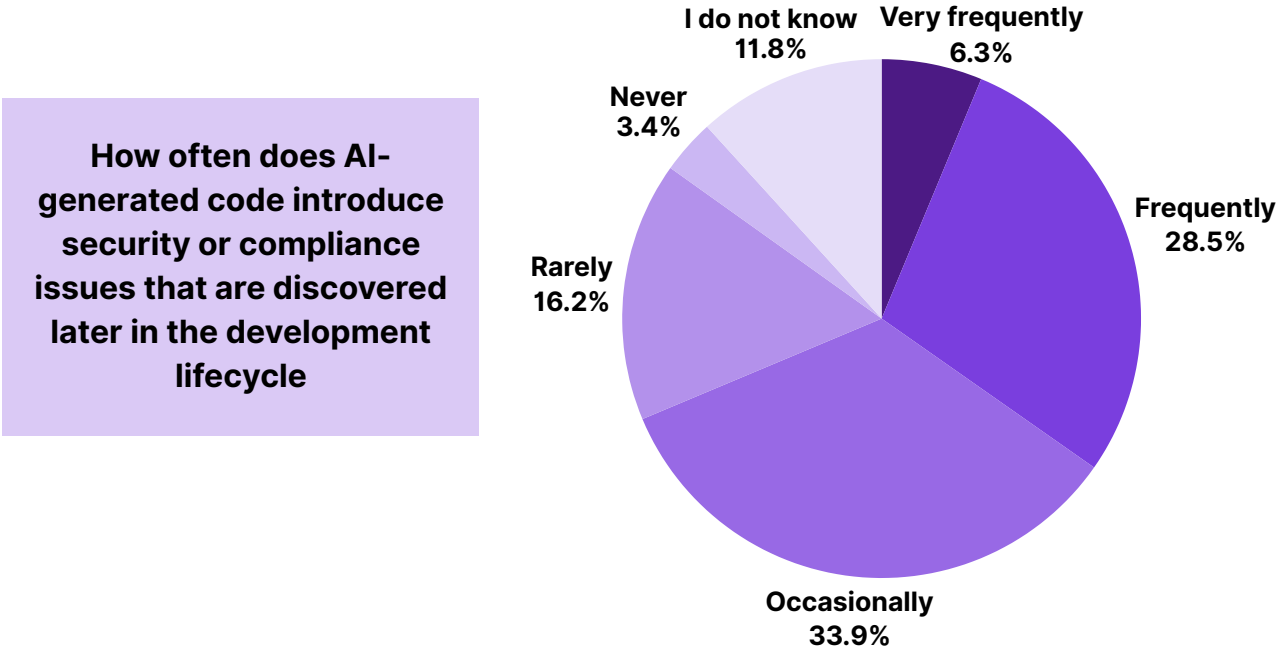
AI Coding Assistants Shift Security Earlier, but Not Early Enough

The tool landscape is fragmented. GitHub Copilot is approved or in use at **60.1%** of organizations, Claude Code at **57.4%**, Codex at **30.3%**, Gemini CLI at **21.4%**, Cursor at **18.3%**, Windsurf at **13.6%**, and JetBrains AI at **7.3%**. Because respondents could select multiple tools, security teams need policy consistency across a heterogeneous assistant ecosystem rather than controls tied to a single interface.



Security findings are showing up after generation

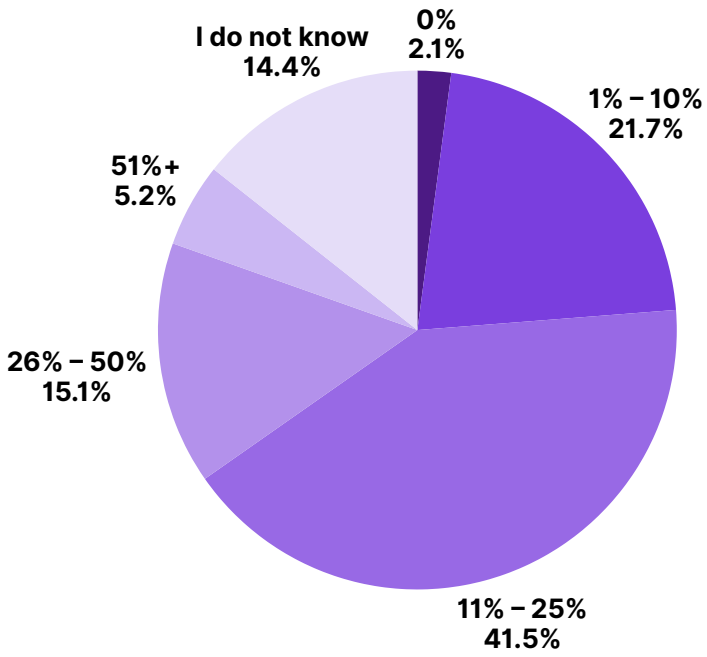
6.3% say AI-generated code very frequently introduces security or compliance issues that are discovered later in the development lifecycle, **28.5%** say frequently, and **33.9%** say occasionally. In total, **68.7%** report this happens at least occasionally. **16.2%** say rarely, **3.4%** never, and **11.8%** do not know.



AI Coding Assistants Shift Security Earlier, but Not Early Enough

Respondents also estimate that AI-assisted code is contributing meaningfully to application security findings. **41.5%** estimate that 11–25% of AppSec findings originate from AI-generated or AI-assisted code, **15.1%** estimate 26–50%, and **5.2%** estimate more than half. In total, **83.5%** estimate AI-assisted code accounts for at least some findings, and **61.8%** estimate it accounts for more than 10%. This measure is explicitly an estimate and should not be presented as instrumented attribution.

What percentage of your application security findings do you estimate originate from AI-generated or AI-assisted code?

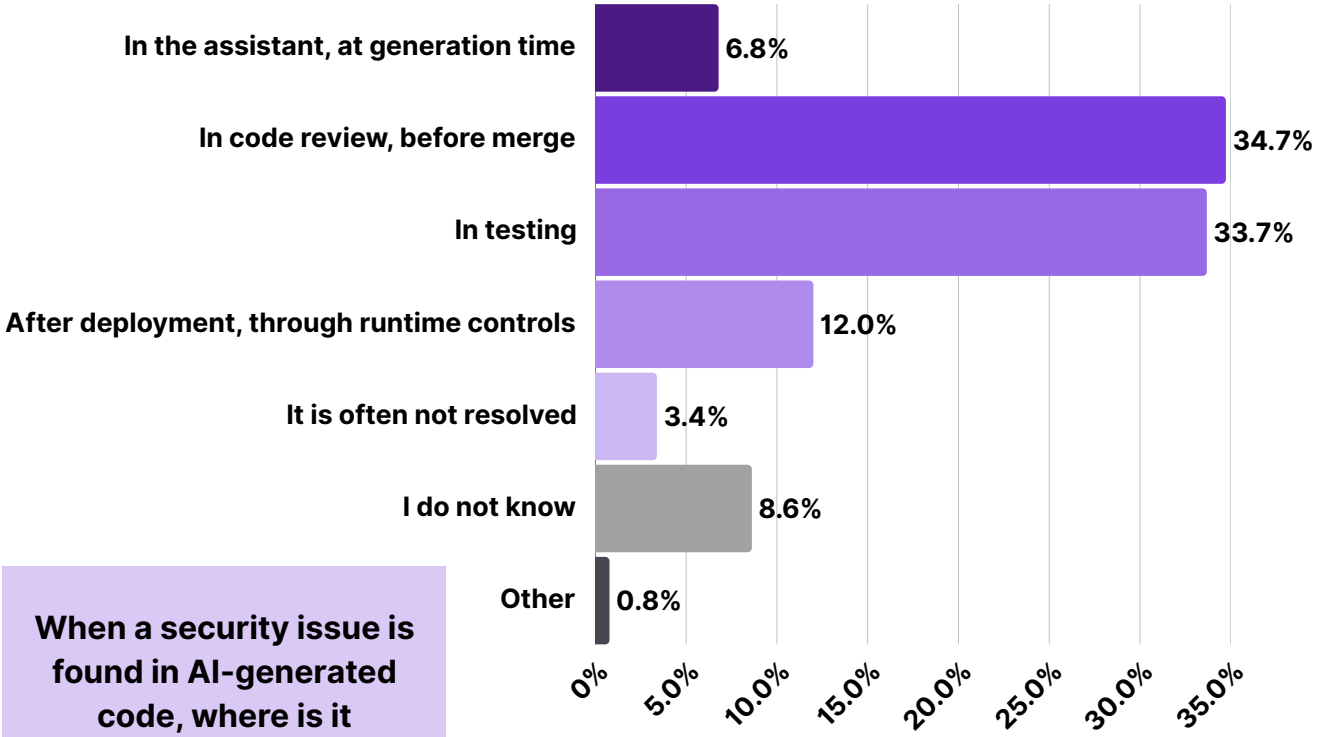


Most remediation still happens after the assistant

Only **6.8%** say a security issue found in AI-generated code is typically resolved in the assistant at generation time. **34.7%** resolve the issue during code review before merge, **33.7%** in testing, and **12%** after deployment through runtime controls. **3.4%** say issues are often not resolved, **8.6%** do not know, and **0.8%** selected another approach.

AI Coding Assistants Shift Security Earlier, but Not Early Enough

These results argue for extending secure-by-design controls into the AI-assisted development workflow. Traditional SAST, DAST, software composition analysis, code review, and runtime protection remain essential, but AI coding assistants add another policy enforcement point before code enters the repository. Guardrails at generation time can reduce avoidable rework while downstream controls provide independent verification.

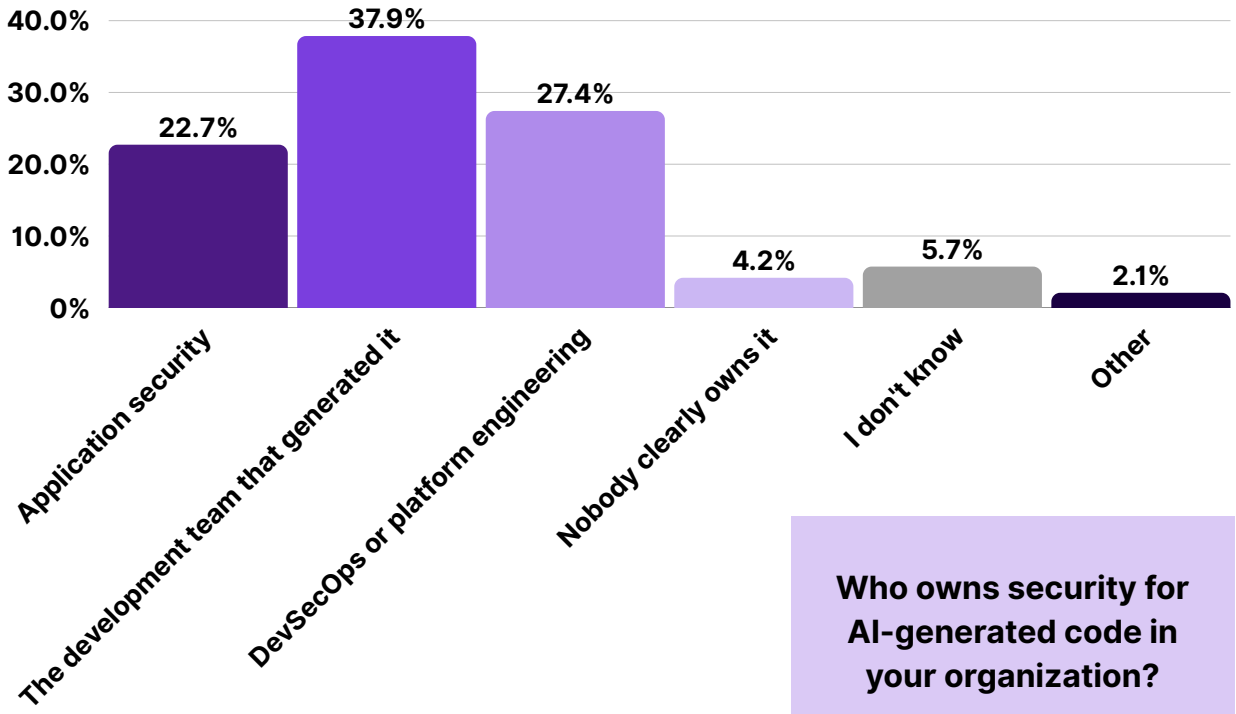


When a security issue is found in AI-generated code, where is it typically resolved?

Ownership is distributed; enforcement is incomplete

Security ownership for AI-generated code most often sits with the development team that generated it (**37.9%**), followed by DevSecOps or platform engineering (**27.4%**) and application security (**22.7%**). **4.2%** say nobody clearly owns it and **5.7%** do not know. This distribution reinforces the need for controls developers can use directly rather than relying exclusively on a centralized security review queue.

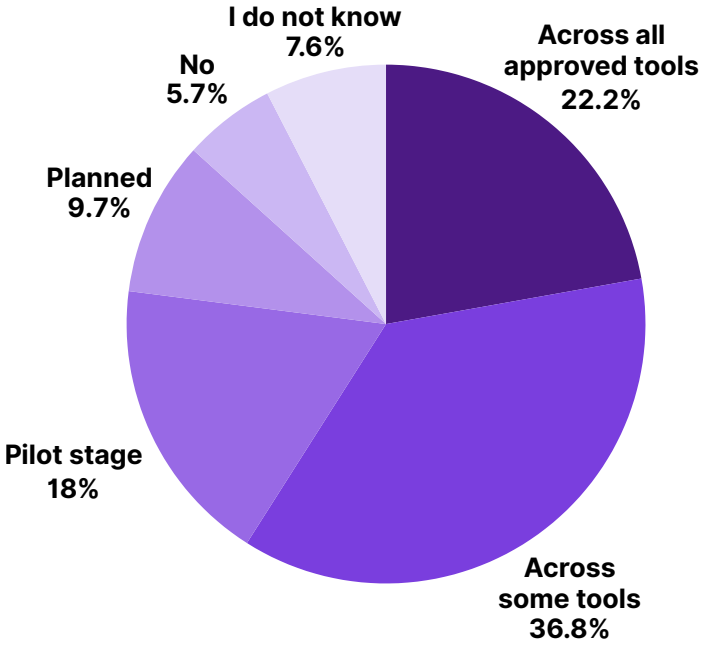
AI Coding Assistants Shift Security Earlier, but Not Early Enough



Who owns security for AI-generated code in your organization?

59% report automatic security and compliance policy enforcement in all or some AI coding assistants, but only **22.2%** have it across all approved tools. **18%** are in pilot, **9.7%** have it planned, **5.7%** have no automated enforcement, and **7.6%** do not know. The policy challenge is increasingly one of consistency: security standards need to follow the developer across assistants, repositories, and deployment paths.

Are your security and compliance policies automatically enforced within developers' AI coding assistants?

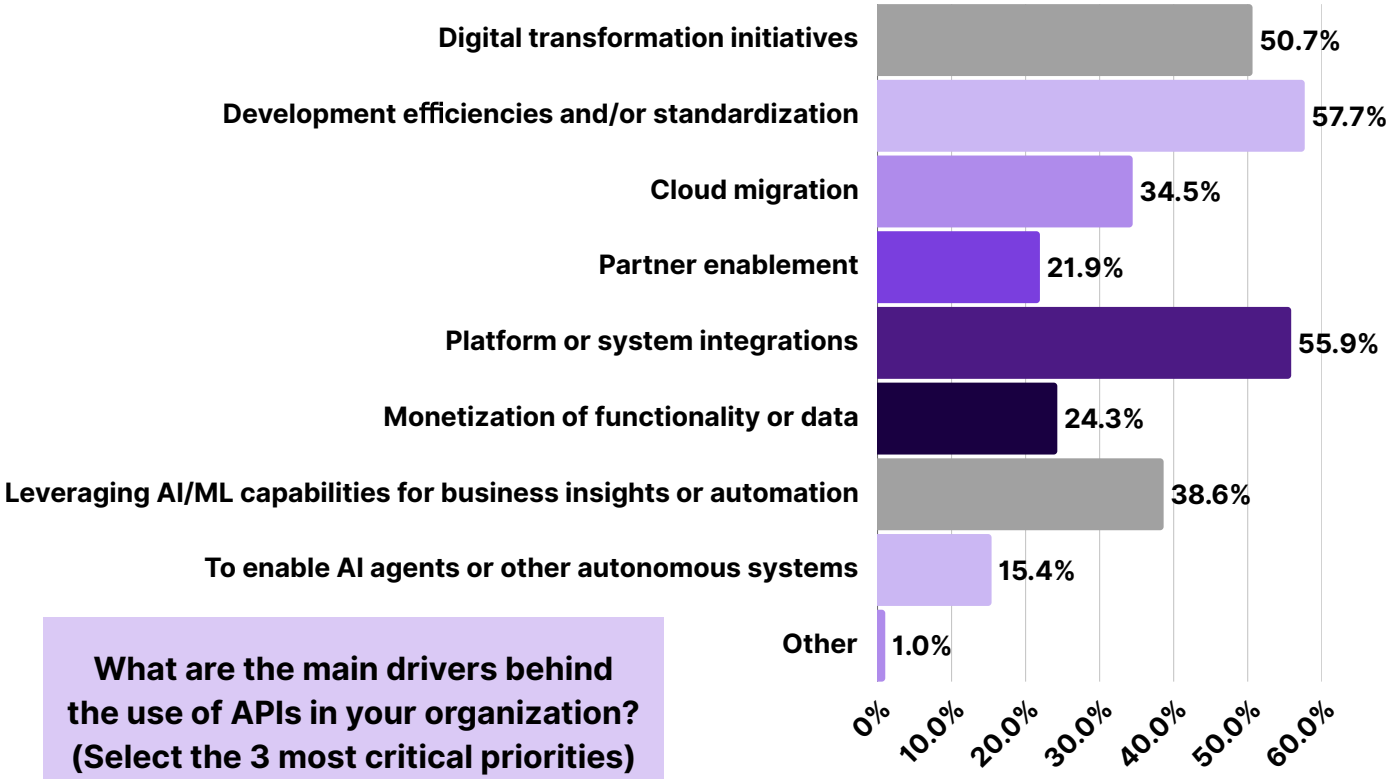


APIs Remain the Foundation of the Agentic Enterprise

Agentic AI is changing what enterprise APIs do, but not why organizations need them. APIs still enable the basic work of connecting systems, standardizing development and delivering digital services. What is changing is the number of non-human actors using those interfaces and the speed at which they can initiate requests and chain operations together. Every meaningful action an agent takes eventually becomes an API call, which is why the API layer matters more in the agentic era, not less.

API adoption is still led by development and integration

Respondents continue to rank development efficiency and standardization as the leading API driver at **57.7%**, followed closely by platform or system integrations at **55.9%** and digital transformation at **50.7%**. AI-related use cases are now meaningful contributors: **38.6%** selected AI/ML capabilities for business insights or automation, and **15.4%** selected enabling AI agents or other autonomous systems among their three most critical API priorities.



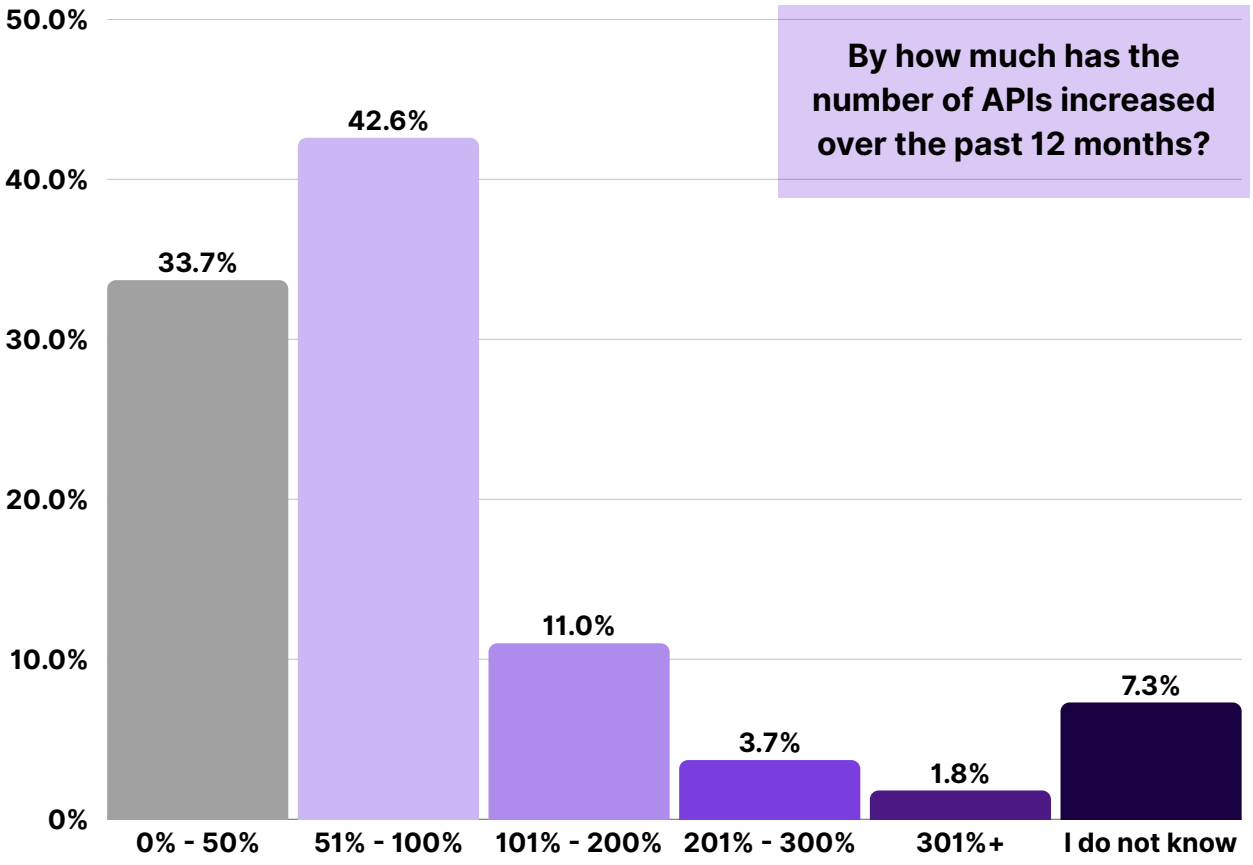
What are the main drivers behind the use of APIs in your organization? (Select the 3 most critical priorities)

APIs Remain the Foundation of the Agentic Enterprise

Compared with our prior report, AI/ML analytics and automation increased from **32%** to **38.6%**, while platform/system integrations increased from **50%** to **55.9%**. Cloud migration declined from **43%** to **34.5%**. These shifts suggest that APIs are increasingly being used to orchestrate applications and automation rather than primarily to support infrastructure migration.

API growth remains structurally high

59.1% of respondents report API growth above 50% in the past 12 months. The largest group, **42.6%**, experienced growth between 51% and 100%; another **11%** report growth of 101% to 200%, **3.7%** report 201% to 300%, and **1.8%** report growth above 300%. **33.7%** report growth of up to 50%, while **7.3%** do not know.



APIs Remain the Foundation of the Agentic Enterprise

For security teams, volume is only part of the issue. API growth multiplies the number of identities, data flows, authorization relationships, and business processes that teams must understand. Agentic systems add another dimension: one user request can produce a sequence of machine-generated calls across multiple services, increasing the importance of maintaining context across the entire execution chain.

Inventory confidence is improving, but shadow exposure has not disappeared

30.8% say they have a fully automated, real-time inventory of every API the organization develops, delivers, or integrates. **53.3%** are somewhat confident and acknowledge that shadow or undocumented APIs likely exist. **10.2%** rely on manual documentation that is often outdated, **4.4%** have no formal mechanism for tracking API sprawl, and **1.3%** do not know.

The key distinction in an agentic architecture is that inventory must extend beyond endpoints. Teams increasingly need to understand the relationships among agents, MCP servers, tools, APIs, data, and identities. A flat API list cannot show whether a low-risk endpoint becomes high impact when an autonomous agent with broad credentials can reach it.

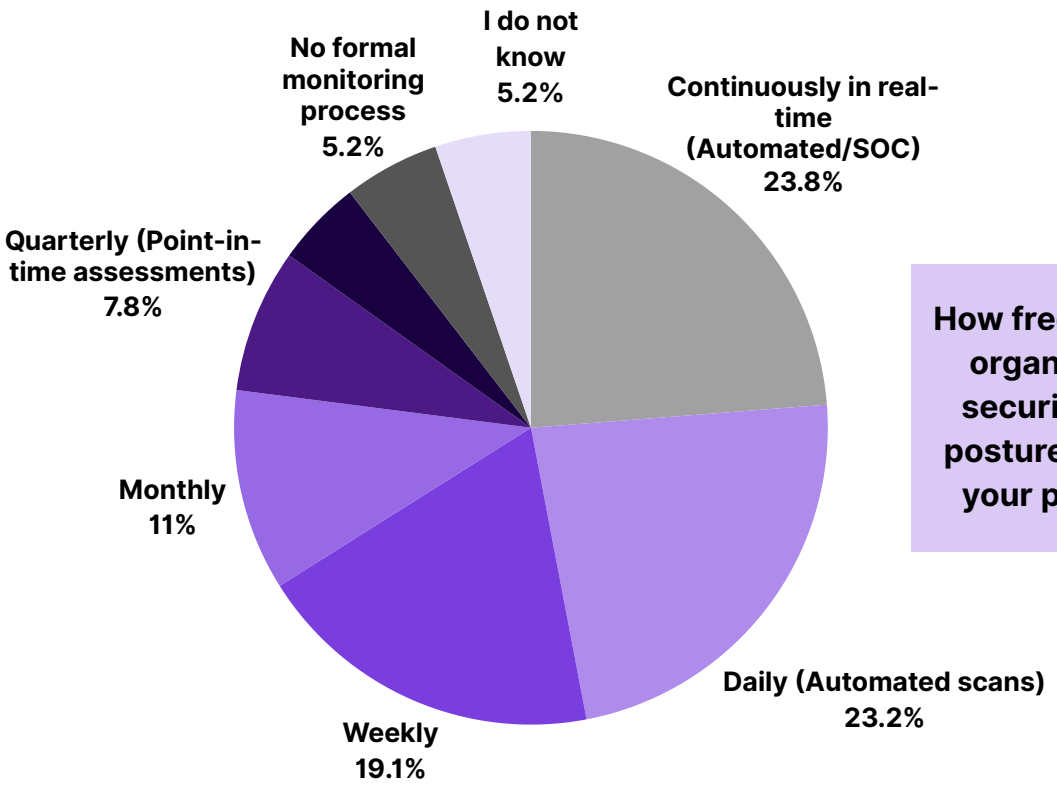
API Security Pressure Persists Despite Maturing Programs

The survey contains a notable tension: organizations report stronger API security programs than in the previous report, but API security continues to create operational friction and production findings remain common.

Program maturity has improved

21.7% now describe their API security strategy as advanced, with dedicated API testing and protection, compared with **8%** in the prior report. **27.9%** describe an intermediate program, **20.6%** basic, **24.3%** planning-stage, and **5.5%** non-existent. The advanced share increased by **13.7** percentage points, while planning-stage programs declined by **8.7** points.

Monitoring remains less mature than strategy labels suggest. **23.8%** perform continuous, real-time monitoring or posture assessment, **23.2%** do so daily, and **19.1%** weekly. **11%** monitor monthly, **7.8%** quarterly, **4.7%** only during annual penetration testing, **5.2%** have no formal monitoring process, and **5.2%** do not know.



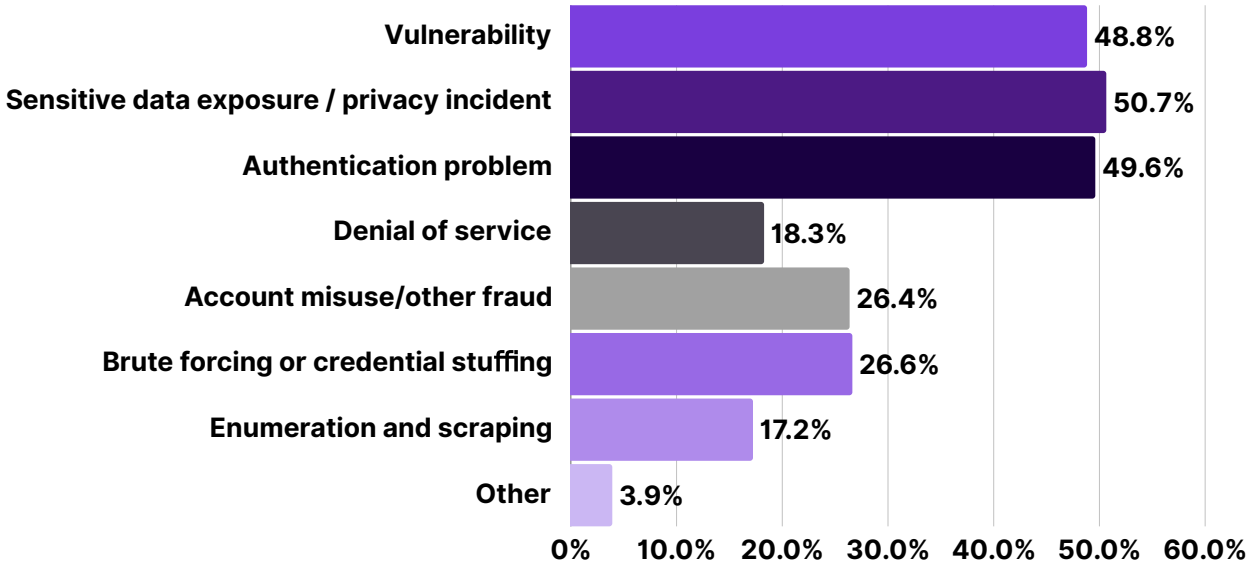
How frequently does your organization perform security monitoring or posture assessments on your production APIs?

API Security Pressure Persists Despite Maturing Programs

Incidents are relatively stable; release friction is increasing
30.8% report an API security incident in the past 12 months, close to the **32%** level in the previous report. Yet **52.5%** have slowed the rollout of a new application because of API security concerns, up from **47%**. This suggests that security is increasingly affecting delivery decisions even as headline incident prevalence remains broadly stable.

Production findings are rising across several categories
Sensitive data exposure or privacy incidents are the most commonly reported production problem at **50.7%**, followed by authentication problems at **49.6%** and vulnerabilities at **48.8%**. Brute force/credential stuffing (**26.6%**) and account misuse or other fraud (**26.4%**) follow, then denial of service at **18.3%** and enumeration/scraping at **17.2%**. Several categories are higher than in the prior report. These are respondent-reported findings rather than normalized incident rates, so the changes should be interpreted as shifts in reported experience, not as proof that the underlying threat frequency rose by the same amount.

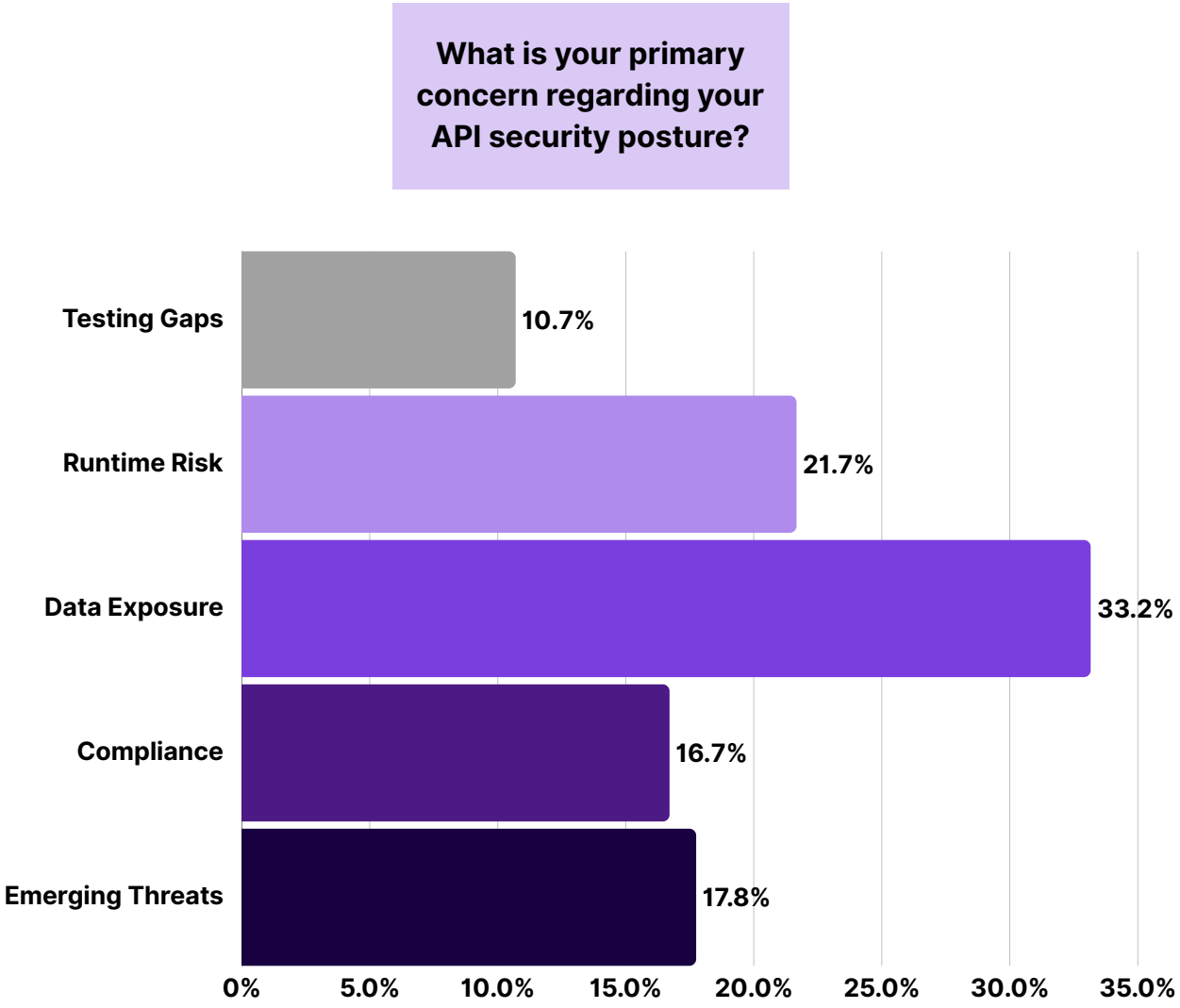
In the past 12 months, what security problems have you found in production APIs? (Select all that apply)



API Security Pressure Persists Despite Maturing Programs

Data exposure remains the leading posture concern

When asked to choose a single primary API security posture concern, **33.2%** selected lack of visibility into PII or sensitive data leakage. Runtime protection against active production attacks followed at **21.7%**, identifying non-human or automated traffic at **17.8%**, compliance at **16.7%**, and insufficient testing during development at **10.7%**. The ranking reflects the centrality of data context: defenders increasingly need to know not only that an API exists, but what sensitive data it exposes and which human or machine identities can reach it.



Salt Labs: What Real-World Attack Telemetry Shows

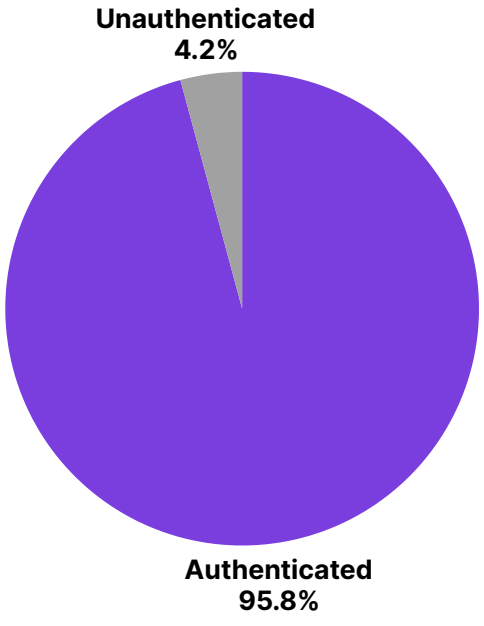
Survey responses capture how organizations perceive risk and maturity. Salt Labs telemetry provides a separate view of what is happening across production API environments. The analysis spans across hundreds of customer environments, **5,319,583** API endpoints, and **104,339** attacker instances over a 12 month period.

Authenticated activity still dominates

95.8% of attacker instances were authenticated, compared with **99%** in the prior period. Unauthenticated instances increased from **1%** to **4.2%**. The shift is consistent with an increase in opportunistic or credential-less scanning, but the telemetry alone cannot establish the cause. More importantly, the continued dominance of authenticated activity reinforces a core API security principle: authentication is only the beginning of trust. Defenders also need object and function-level authorization, behavioral context, and visibility into how valid credentials are being used.

This metric should not be interpreted as meaning authenticated attackers are necessarily AI agents or compromised machine identities. An authenticated attacker instance can reflect many forms of credential, token, or session abuse. Agent identity requires separate telemetry and correlation.

Attack attempts from authenticated vs. unauthenticated attackers



Salt Labs: What Real-World Attack Telemetry Shows

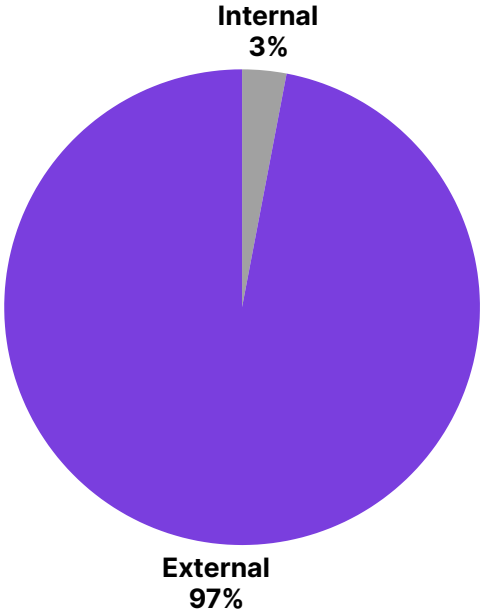
External APIs receive most observed attacker activity

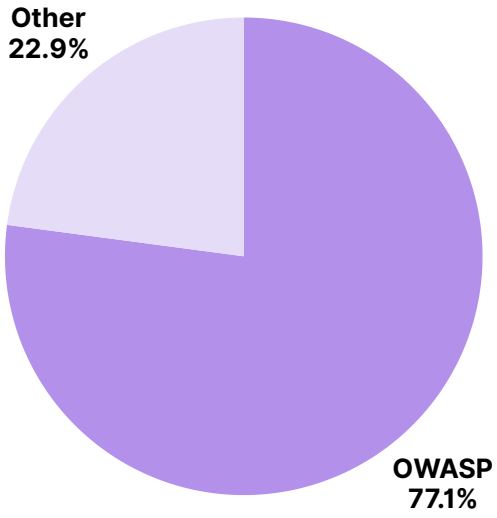
97% of attacker instances targeted external-facing API endpoints, and **3%** targeted internal endpoints, compared with **99%** and **1%** in the prior period. The internal share remains small, but its increase is worth monitoring as more enterprise workflows rely on service-to-service and agent-to-service communication.

Known API weakness patterns still dominate

77.1% of attacker instances mapped to OWASP API Security Top 10 categories, almost unchanged from **78%** in the previous period. For longitudinal consistency, Salt Labs is using category labels aligned to the OWASP API Security Top 10 2019 taxonomy in this dataset. That distinction is important because OWASP changed several category names and numbers in the 2023 edition.

Attack attempts against internal and external facing API endpoints





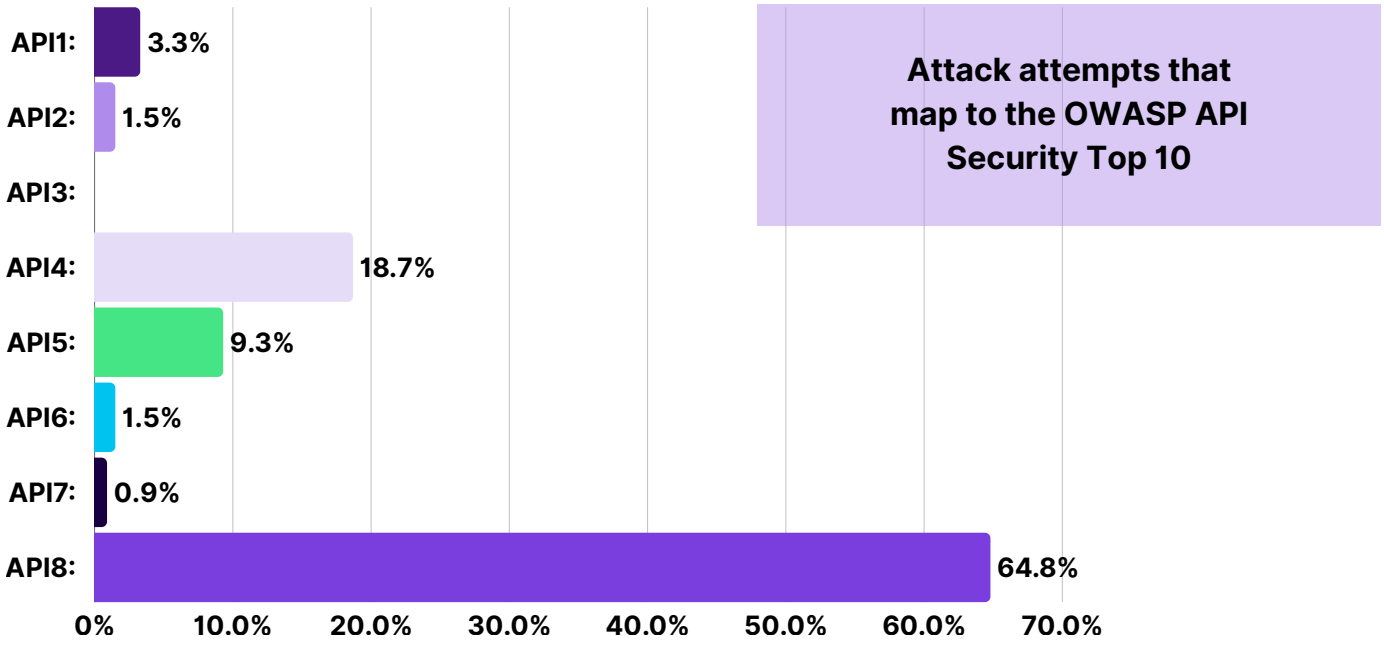
Attack attempts leveraging the OWASP API Security Top 10 vs other attack types

Salt Labs: What Real-World Attack Telemetry Shows

Among OWASP-mapped attacker instances, API8:2019 Injection / Code Injection remained dominant at **64.8%**, essentially unchanged from **65%**. API4:2019 Lack of Resources & Rate Limiting rose from **13%** to **18.7%**, a **5.7** percentage-point increase. API5:2019 Broken Function Level Authorization represented **9.3%**, API1:2019 Broken Object Level Authorization **3.3%**, API2:2019 Broken User Authentication **1.5%**, API6:2019 Mass Assignment **1.5%**, and API7:2019 Security Misconfiguration **0.9%**.

The API inventory remains heavily exposed by design

Across the endpoints analyzed, **72.9%** were external-facing and **27.1%** internal, while **69.4%** were unauthenticated and **30.6%** authenticated. An unauthenticated endpoint is not inherently vulnerable (many public APIs intentionally permit unauthenticated requests), but the volume makes data classification, rate/resource controls, input validation, and business-logic protection especially important. Endpoint exposure should be assessed in the context of the data and functions reachable behind it.



Recommendations for CISOs and Application Engineering Leaders

The report data suggests that the most effective agentic security programs will not be built around a single model control. They connect discovery, governance, runtime detection and response around the agentic path. The priorities below are ordered to follow that path, from knowing what exists to stopping it when it goes wrong.

1. Build a living inventory of the agentic system, not only the API estate.
Continuously map AI agents, MCP servers, tools, APIs, sensitive data flows, and ownership, and record the relationships among them so teams can see which capabilities become reachable when an agent is granted a new tool or credential.

2. Give every production agent a bounded, accountable identity.
Prefer dedicated non-human identities with least-privilege permissions over shared service accounts, human credentials, or long-lived embedded keys. Make ownership explicit and design for rapid revocation.

3. Treat MCP as governed infrastructure.
Inventory local and remote MCP servers, verify ownership, constrain exposed tools, authenticate remote access, apply least privilege downstream, and monitor which APIs and sensitive resources MCP-enabled tools can reach.

4. Trace the complete path from intent to outcome.
Preserve the initiating user or process, agent decision context, MCP/tool invocation, API requests, and resulting state changes in a correlated audit trail. Make reconstruction a minutes-scale operation rather than a multi-team forensic project.

5. Move containment to the action boundary.
Apply policy checks before high-impact tool or API calls execute. Use transaction limits, scoped authorization, approval gates for sensitive actions, and automated blocking or credential revocation where the risk justifies it.

6. Modernize runtime API security for machine-speed behavior.
Monitor continuously where possible, prioritize identity- and behavior-aware detection, protect authorization and business logic, and correlate activity across sessions and endpoints rather than relying only on individual request signatures.

7. Put secure development policy inside AI-assisted coding workflows.
Treat the coding assistant as an additional control point. Enforce organization-specific security and compliance rules at generation time, then retain independent code review, SAST/DAST/SCA, testing and runtime controls as defense in depth.

8. Measure controls against business outcomes.
Track the percentage of agents with scoped identities, path-trace coverage, mean time to reconstruct and contain an agent action, MCP discovery lag, the share of AI-code findings prevented at generation, API rollout delays, and recurring authorization or data-exposure findings.

Framework alignment

Use current frameworks as organizing aids rather than as substitutes for architecture-specific risk analysis. For API programs, OWASP API Security Top 10 2023 is the current edition; where Salt Labs retains the 2019 taxonomy for historical comparison, it is labeled explicitly. For AI governance, the NIST AI Risk Management Framework and its Generative AI Profile provide voluntary, use-case-agnostic guidance. These frameworks help structure governance, but they do not remove the need to model agent-specific identities, tool permissions and paths.

Conclusion: Secure What Agents Do

The defining security change in the agentic era is not simply that AI can generate more content or code. It is that AI systems are increasingly authorized to act. They can invoke tools, call APIs, change permissions, transact, and touch sensitive data across multiple systems in response to a single instruction. That makes the path from intent to action a new unit of security analysis.

The 2H 2026 data shows both progress and exposure. Agent deployment is widespread, API security programs are maturing, and many organizations have begun implementing controls across AI and API infrastructure. At the same time, nearly half of respondents report confirmed or suspected unintended agent actions, only a minority can automatically stop an agent mid-action, and complete traceability remains uneven. Salt Labs is also beginning to see agentic-connected API paths intersect with attacker activity in production telemetry.

For CISOs, the priority is governance that can survive machine speed. For application and platform engineers, it is building systems where identities, permissions, and actions remain observable and constrained as autonomous workflows become more complex. The organizations that can answer “what did this agent do, why was it allowed, and can we stop it?” will be better positioned to scale agentic AI without turning velocity into unmanaged risk.

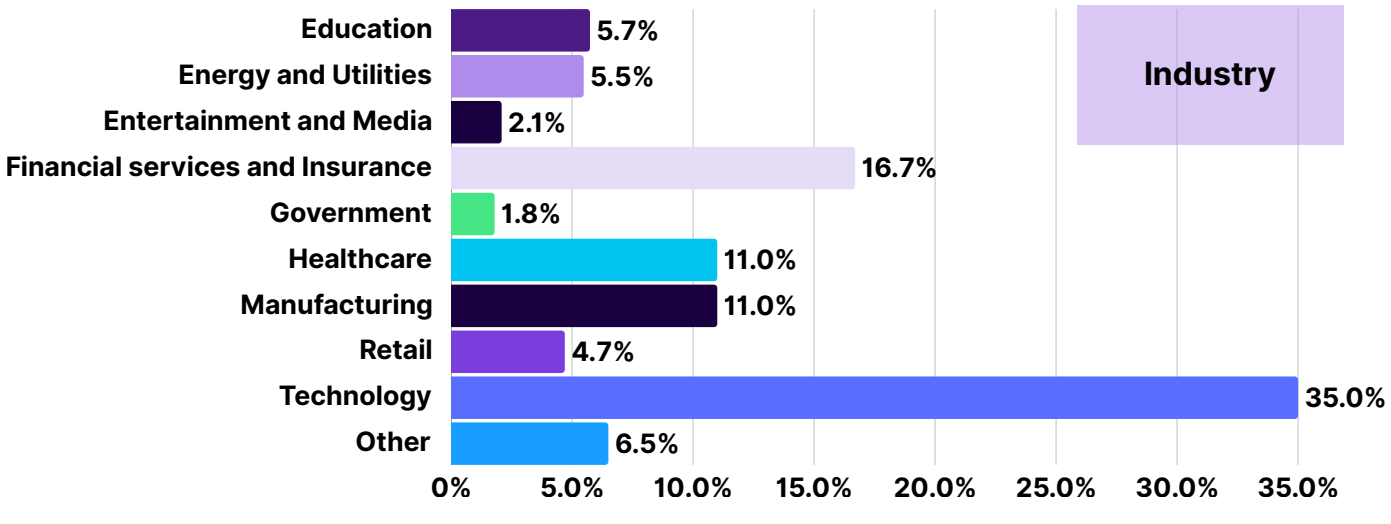
Methodology and Research Notes

Survey

The 2H 2026 Agentic AI and API Security Survey contains over 380 analyzed responses across security, technology, architecture, application security, API platform, DevOps, and adjacent roles. The respondent base spans multiple industries; technology represents 35%, financial services and insurance 16.7%, healthcare 11%, and manufacturing 11%, with additional respondents across education, energy/utilities, retail, government, media, and other sectors. Company sizes range from 1–100 employees through 10,001+ employees.

Percentages are rounded to the nearest tenth. Multi-select questions may total more than 100%. Derived statistics (for example, “more than 10 agents”) are calculated from the underlying response distribution. The question on what an unintended action involved is a follow-up asked only of respondents who reported a confirmed or suspected event.

Longitudinal comparisons use our prior 1H 2026 report where question wording and response categories are sufficiently comparable. The surveys are not a fixed respondent panel, so changes should be interpreted as directional shifts across survey populations, not as within-organization change.



Methodology and Research Notes

Salt Labs telemetry

Salt Labs analyzed more than 100,000 distinct attacker instances across hundreds of customer environments over the September 2025–August 2026 lookback period. Endpoint inventory figures reflect the cumulative API surface as of August 2026. The agentic attack analysis is experimental: Salt Labs correlated violation records with agentic-connected API inventory across a limited set of environments, and the result should be treated as directional and not generalized to the broader market.

Third-party research

Where external research is cited, it is attributed to its source. The AI-generated code security figures in Section 4 are from Veracode's 2026 GenAI Code Security Report and are not Salt data.

Terminology

Agentic path: the full sequence an AI agent follows from an initial instruction or intent, through the model's reasoning, the MCP servers and tools it invokes, and the APIs it calls, to the resulting action or outcome.

About Salt Security

Salt Security is the leader in agentic and API security, protecting the world's most innovative enterprises from AI agent and API attacks. The Salt Security Agentic Security Platform discovers the agents, MCP servers, and APIs operating in an environment, governs the connections and permissions between them, monitors agent behavior at runtime, and eliminates vulnerabilities before they reach production. Salt Security was founded in 2016 and is backed by Sequoia Capital, S Capital, Tenaya Capital, Salesforce Ventures, Advent International, and other leading investors. For more information, visit salt.security.

See Salt for yourself

The Salt Agentic Security Platform gives you complete visibility into your entire agentic landscape in less than 10 minutes. See exactly what is exposed, what is at risk, and where to act first.

REQUEST A DEMO

© 2026 Salt Security. All rights reserved. This document is for informational purposes only. Product capabilities are subject to change.

