

AHEAD

The Seam: Securing AI Coding Assistants

A control architecture for mature, regulated enterprises already running Claude Code, OpenAI Codex, GitHub Copilot, and Cursor against production systems.

AHEAD Specialist Solutions Engineering

The question the market is asking vs. the question that matters

Almost all of the AI security writing produced in 2026 is aimed at one question: is the code an AI wrote safe to ship? It is a fair question. We think it is the wrong one on which to build a control program.

The evidence on code quality is not reassuring, and it deserves a paragraph before we set it aside. The foundational study of GitHub Copilot found roughly 40 percent of 1,689 generated programs contained a vulnerability.⁵ Four years and many model generations later, Veracode measured the security pass rate for AI-generated code holding flat at 56 percent across more than a hundred models.⁶

Mature engineering organizations already know how to metabolize that class of risk. It is a code quality problem, and code quality problems route into static analysis, peer review, and release gates that have existed for two decades. What those controls need is retuning for volume, not reinvention.

The harder problem lies in what a coding agent actually is. Treating it as a code generator understates it badly. It is a privileged actor operating inside your engineering estate, with credentials, network access, and the authority to execute. It reads material it did not author and cannot fully vet, acts on that material at machine speed, and in a single session touches more of the estate than most of the engineers supervising it.

Securing the artifact is an application security problem, and one most organizations have already solved in substance. But securing the actor is a different matter: an identity, authorization, and detection problem that in most places we have looked has not been framed yet, let alone funded.

BY THE NUMBERS

Nine in ten organizations use AI to assist with coding. **86%** now ship AI-written production code, rising to **91%** among enterprises.

Anthropic, The 2026 State of AI Agents Report.

Unsupervised iteration makes code worse, not better: a **37.6%** increase in critical vulnerabilities after only five rounds of LLM refinement.

Shukla et al., arXiv:2506.11022. Preprint.

Adaptive attacks exceed **85%** success against current defenses, while most catalogued defenses achieve **under 50%** mitigation.

Review of 78 studies, arXiv:2601.17548. Preprint.

An agent that writes perfectly secure code can still be the reason you lose your source tree. What needs defending is everything that agent can reach.

Three properties that no previous developer tool combined

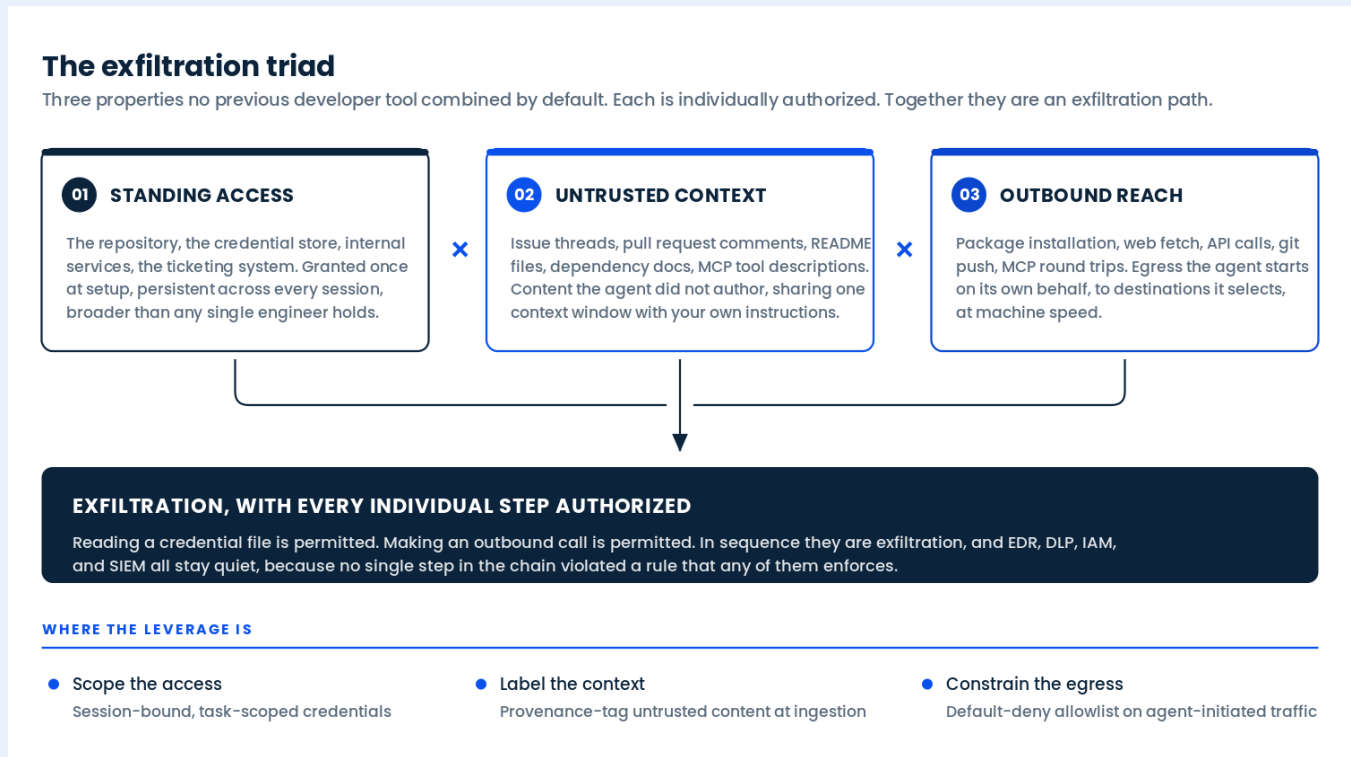


Figure 1. A sector-agnostic evolution of Simon Willison's widely cited “lethal trifecta” framing, adapted to the specific privileges a coding agent holds inside an engineering estate.

Every step is authorized. The sequence is the attack.

Three properties converge in a coding agent that had never previously converged in one process. Each is individually legitimate, and each was approved by someone with the standing to approve it. Together they form a path that no control in a conventional security stack is positioned to interrupt.

Reading a credential file is permitted. So is making an outbound network call, and so is committing to a branch. In sequence, they are exfiltration, and so endpoint detection, data loss prevention, identity governance, and the SIEM all stay quiet because no single step in the chain violated a rule that any of them enforces.

This is not speculative. Academic measurement of indirect prompt injection against tool-using agents found a ReAct-prompted GPT-4 agent complied with injected instructions 24% of the time, roughly doubling when the injection was reinforced.⁸

The most important structural fact is that the agent cannot reliably distinguish your instructions from content it read. Both occupy the same context window. This is a property of how these systems work rather than a defect awaiting a patch, which is why the durable defenses tend to be architectural ones.

WHERE THE LEVERAGE IS

Break any one and the chain fails:

- 1) Scope the access to session-bound, task-scoped credentials.
- 2) Label untrusted content with its provenance at the point of ingestion.
- 3) Constrain egress with a default-deny allowlist on agent-initiated traffic.

All three are achievable with technology that exists today.

Six control planes, and the seam where four of them live

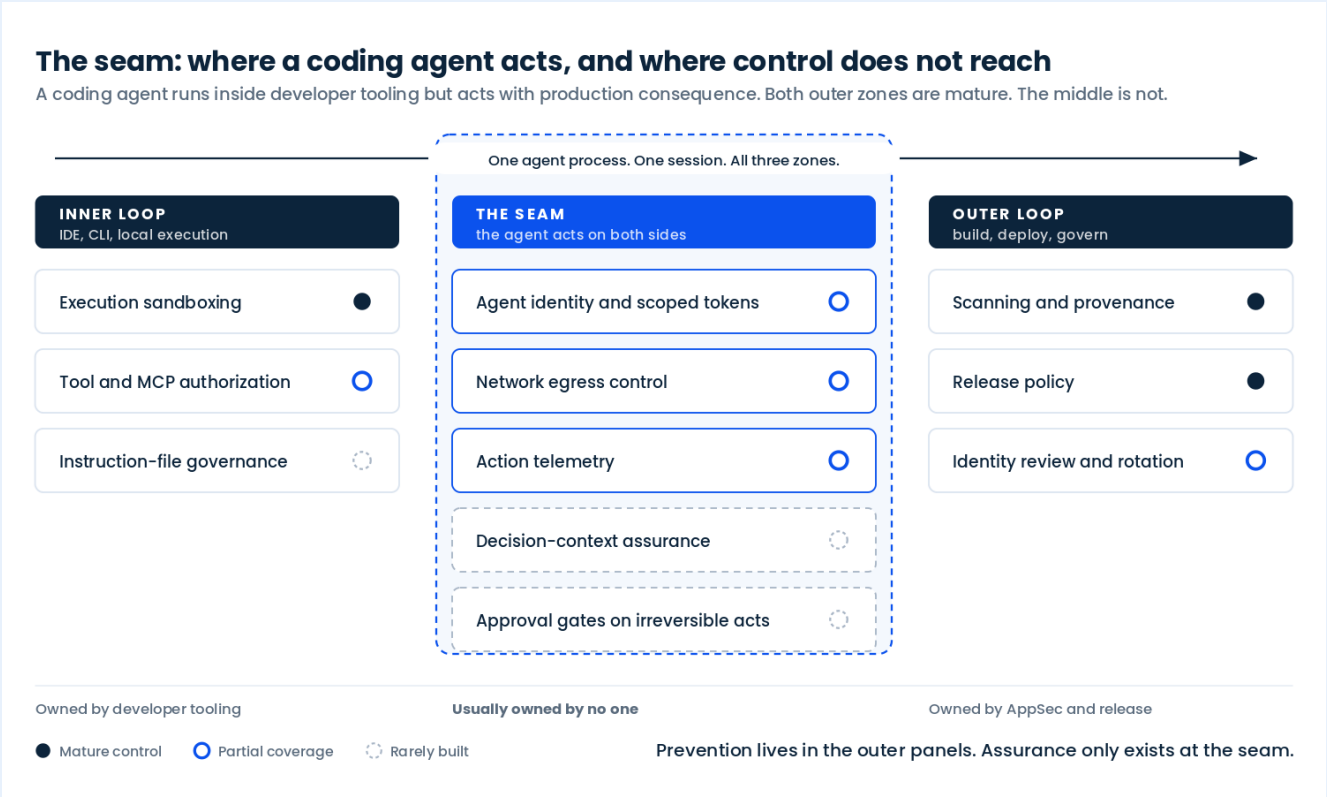
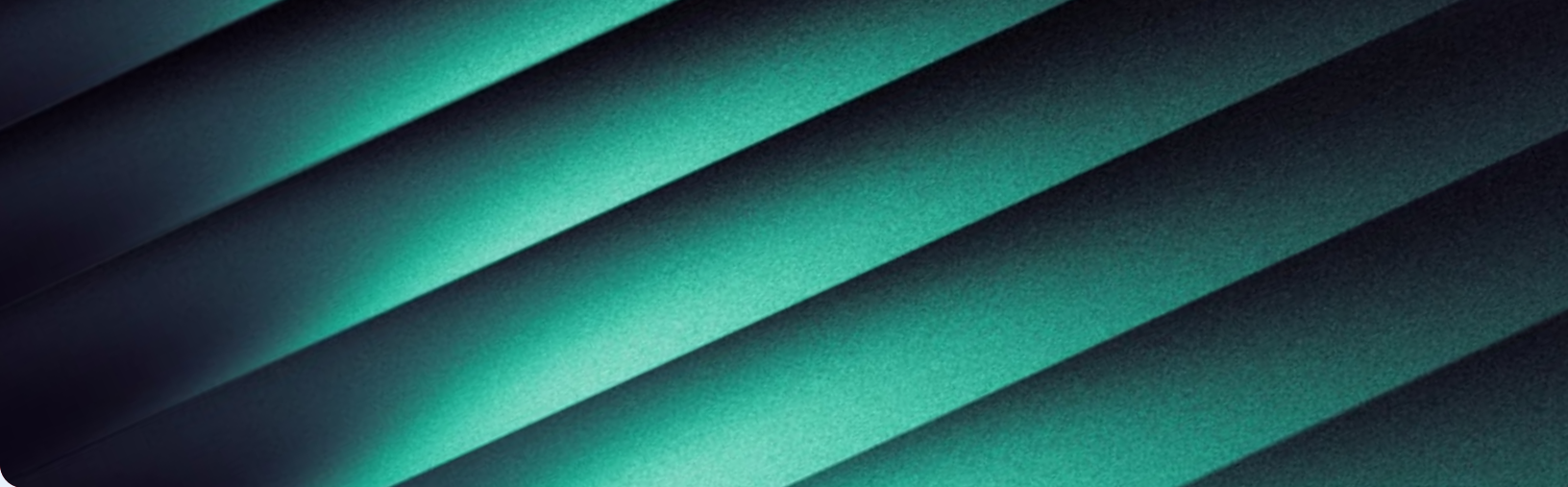


Figure 2. Control-plane model informed by CISA and Five Eyes agentic AI guidance,²¹ NIST SP 800-218A,²² and the OWASP Top 10 for Agentic Applications.²⁵ Status markers reflect typical maturity observed across AHEAD enterprise engagements, not a vendor capability assessment.



Configuration is not enforcement

Most security programs already own the outer panels. Prevention in the IDE is a solved procurement. Governance at release is a mature function with named owners, audit evidence, and a seat in change management.

The middle is different. Of the six planes, four are enforced only at the seam where a coding agent acts: agent identity and scoped tokens, network egress control, human approval gates, and observability, which resolves in practice into two distinct records, action telemetry and decision-context assurance. The remaining two are configured in the inner-loop tool. Both are only ever exercised at the moment the agent acts.

Least privilege is necessary and insufficient

Least privilege bounds what an agent can reach. It says nothing about what the agent should be allowed to do unsupervised, in what order, or who signs off. A permission model has no vocabulary for any of that. Behavioral policy is where those questions have to live. Very few programs have extended recertification to cover sequences of authorized actions, which is where the agent risk actually sits.

The approval-by-name problem

When an agent asks permission to run a named tool and the user approves, the default failure mode in shipping clients has been to bind that approval to the tool's name rather than its content. The definition can change afterward while the approval persists. For a tool or MCP definition, bind approval to a content hash of the schema. For a shell command or a deployment gate, bind it to the fully resolved command and argument values. Otherwise the gate is decorative.

Sandboxing is real, and it is not a boundary

A 2026 evaluation of container escape found frontier models achieving a 0.40 success rate on the hardest then-tractable tier. A revision in August 2026 reported a newer model saturating the benchmark outright.¹⁰ A February 2026 Claude Code advisory described untrusted sandboxed code creating a configuration file that injected hooks executing with host privileges on restart.¹¹ The sandbox was real and competently built. What it was not was independent of input the agent itself controlled.

You cannot secure an agent on top of a pipeline you cannot attest

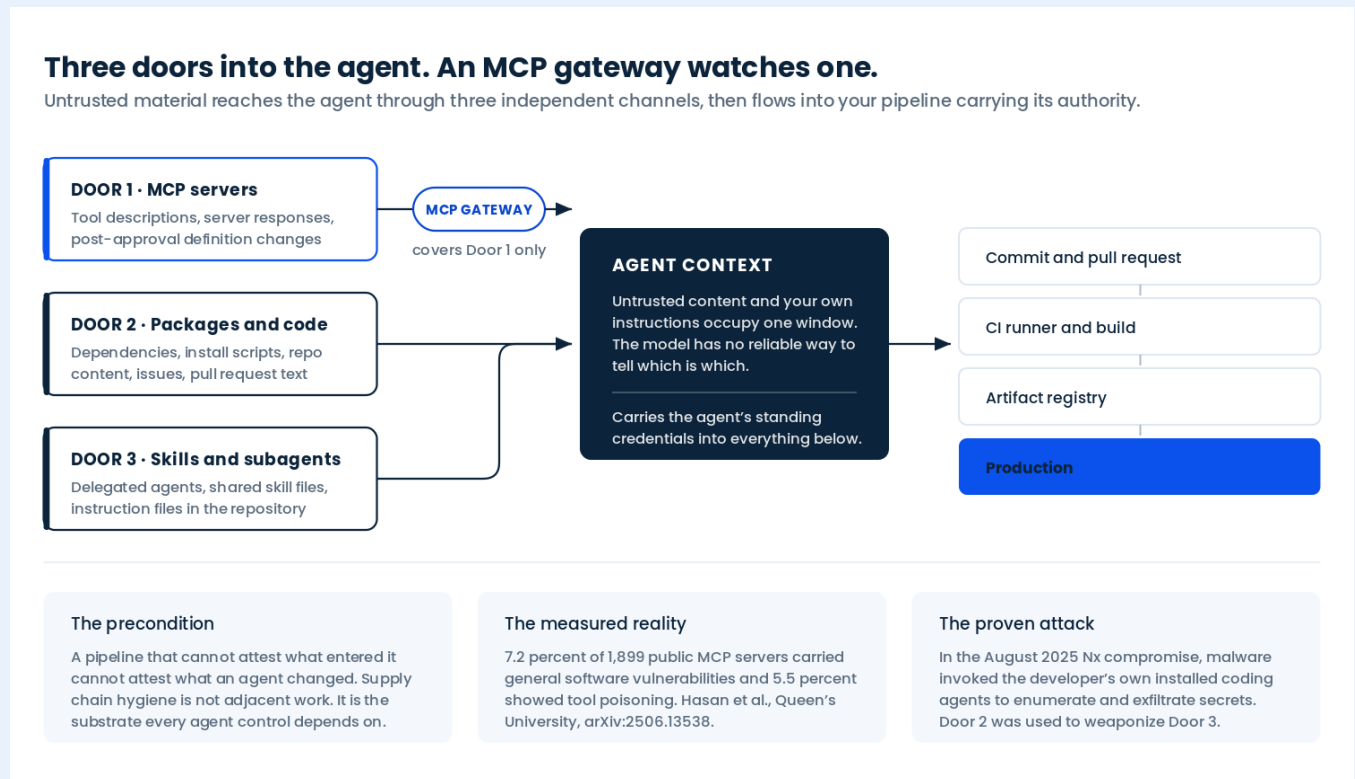


Figure 3. An MCP gateway is a legitimate and useful control positioned on one of three ingress paths. Programs that treat protocol-layer governance as agent security are covering roughly a third of the surface.

The second door has already been used at scale

In August 2025, attackers compromised a widely used npm package and shipped a post-install script that did something novel. Rather than performing reconnaissance itself, it invoked the developer's own installed AI coding agents with their permission-bypass flags and instructed them to enumerate and dump the filesystem. Researchers documented it as the first case of attackers weaponizing installed coding agents as part of a supply chain attack.²

Those bypass flags exist for an ordinary reason. Approval prompts are friction, and an engineer under deadline reaches for whatever setting stops the tool asking. Every control in this document narrows what an agent may do without a human in the loop, and every one of them is only as durable as a developer's incentive to leave it enabled.

A program that does not measure bypass-flag usage and unsanctioned tool adoption is inventorying the agents it configured rather than the agents in use, which is the gap the first stage of the roadmap addresses.

Two ingress paths ride inside these doors and get missed routinely. The response side of a web fetch or search call is untrusted inbound content, not merely outbound reach, and every major coding agent ships that capability. So is content a human pastes into the session from an email, a ticket, or a chat thread, which arrives through the trusted user channel while carrying text the attacker wrote.

7.2% of 1,899 public MCP servers carried software vulnerabilities and **5.5%** showed tool poisoning.

Hasan et al., Queen's University, arXiv:2506.13538.

Hallucinated package recommendations at **5.2%** for commercial models and **21.7%** for open source, across 576,000 samples.

Spracklen et al., USENIX Security 2025.

Between **a third and a half** of systems compromised in the Nx attack had an AI command-line client installed.

GitGuardian and Wiz Research counts differ.

WHAT THIS MEANS FOR SEQUENCING

LSA build provenance, signed artifacts and in-toto attestations, dependency pinning, and hardened CI runners are not prerequisites you can defer until after the agent program. They are the mechanism by which any agent control becomes auditable. Without them, an organization can assert that an agent behaved correctly but has no way to demonstrate it.

Clean code is what makes your other controls real

There is a discipline argument that gets dismissed as engineering hygiene. The loose version of it is wrong.

Clean architecture does not change how an agent behaves. A module boundary will not stop one from reading a credential file. Its value is visibility: a bad action becomes something the humans and gates downstream can actually catch.

A codebase with explicit interfaces, small functions, and meaningful tests gives a reviewer a baseline against which a suspicious diff stands out. Without those properties there is no baseline, and review is the control you are actually relying on.

This is why review burden rather than generation speed is the binding constraint most organizations report on AI-assisted delivery, and why security degrades across unsupervised refactoring iterations. The agent is rewriting code whose intent was never expressed structurally, so nothing flags the drift.⁷

The practical form this takes is unglamorous. Architectural decision records that capture why a design holds, pre-commit and CI gates that fail closed rather than warn, and a standing rule that agents open pull requests instead of writing to the main branch.

None of this constrains the agent itself. It limits how long a bad change survives unnoticed, which is the only sense in which any of it functions as a control.

Real incidents and what they’ve taught us

INCIDENT	DATE	WHAT IT DEMONSTRATES
Nx npm compromise	Aug 2025	Malware invoked installed coding agents with permission-bypass flags to harvest credentials. The agent was the attack tool, not the target. ²
GitHub MCP toxic agent flow	May 2025	An injection planted in a public issue caused an agent to pull private repository content and publish it. Architectural, so not patchable server-side. ¹⁴
EchoLeak, CVE-2025-32711	Jun 2025	A single crafted email caused Microsoft 365 Copilot to exfiltrate internal data with no user interaction. ¹⁵
Claude Code CVE-2026-25725	Feb 2026	Sandboxed code wrote a configuration file injecting hooks that executed with host privileges on restart. Fixed in 2.1.2. ¹¹
Agent-caused production data loss	Apr 2026	An over-scoped API token, found by an agent in an unrelated file, was used to delete a production volume and its co-located backups. No attacker involved. Data restored within roughly an hour. ¹⁶
Shadow AI 8-K filing	May 2026	A US bank holding company filed an SEC Item 1.05 material incident report for unauthorized employee AI use exposing customer non-public information. ¹⁷

Three of these incidents either lack a described attacker entirely or do not require one. Blast radius is a real problem independent of adversary activity. Most AI security briefings don’t open with this side of the risk conversation.



What the field has built, and what it proves is possible

Uber's ADR, and the case for open agentic detection

In 2026 Uber open-sourced ADR, an Agentic Detection and Response system, and published the design and production results at MLSys 2026.¹⁸ The core diagnosis is stated plainly in the paper, and it is the sentence to read aloud to any team that believes endpoint detection already covers this: existing tools see file writes but not the agent reasoning, prompts, or causal chains linking intent to execution.

A sensor normalizes local agent logs into a common event schema, with native parsers for Cursor, Cline, and Claude Code. An explorer performs pre-deployment red teaming. A detector runs two-tier online detection, triaging cheaply at volume and escalating only suspicious sessions to expensive context-aware reasoning.

The paper states the sensor and detection framework are released under Apache 2.0. At the time of writing the repository carries the license and a notice that the initial release is still being prepared, so treat the code as announced rather than available and verify before building a plan around it.

For a large enterprise, the practical value of ADR is less the code than the reference architecture and the vocabulary. It establishes that agentic telemetry is a distinct data class, that two-tier triage is the answer to inference cost at enterprise volume, and that the prompt-to-action causal chain is the unit of detection. Those are design decisions any enterprise can adopt regardless of which vendor it ultimately buys from.

Ten months in production across more than **7,200 hosts** and more than **10,000 agent sessions daily**.

Li et al., Uber, arXiv:2605.17380.

Hundreds of credential exposures found across **26 categories**, with prevention running at **97.2% precision**.

Production deployment results.

67% of attacks detected at **zero false positives**, beating three published baselines by two to four times on FI.

ADR-Bench, 302 tasks, 133 MCP servers.

67% detection at zero false positives falls well short of solved. But it still proves the seam is instrumentable, that the telemetry already exists, and that an enterprise can run this at scale without drowning its SOC.

Architectural defenses are outperforming behavioral ones

The research direction that has produced the most durable results treats prompt injection as a control-flow problem rather than a content-filtering problem. Google DeepMind's CaMeL constrains untrusted data so it cannot influence program control flow, and reported solving 77%

of AgentDojo26 benchmark tasks with provable guarantees, against 84% completion for the same agent with no protection at all.¹⁹

That is an explicit, quantified security and utility trade-off, which is exactly the shape of decision a risk committee is equipped to make. Preference-optimization defenses have separately driven attack success below 10% while generalizing to unseen attacks.²⁰ The read for an architect is that defensible controls constrain what an agent can do rather than trying to detect what it was told. Detection is still necessary, but it should not carry the load alone.

WHY THIS MATTERS FOR PROCUREMENT

A vendor that offers detection without constraint is selling you the weaker half of the problem. Ask any prospective supplier which of the six control planes their product enforces rather than observes, and require the answer in writing.

Where the regulators actually are, including where they are absent

The clearest guidance is non-binding, and the binding framework most organizations would reach for has explicitly excluded itself. Some instruments below apply to any large organization. Others are sector-specific, and financial services is simply where the edges have been drawn most sharply so far, which makes it a bellwether for everyone else.

INSTRUMENT	STATUS	WHAT IT MEANS FOR A CODING AGENT PROGRAM
ISO/IEC 42001	2023 Certifiable	The AI management system standard. Sector-neutral, third-party auditable, and increasingly requested in enterprise procurement. The closest thing to a portable control narrative for an agent program. ²⁸
NIST SP 800-218A	Jul 2024 Profile	Generative AI community profile augmenting the Secure Software Development Framework. The most examiner-legible mapping available for agent-assisted development. ²²
Careful Adoption of Agentic AI Services	Apr 2026 Advisory	Joint CISA, NSA, and Five Eyes guidance. Treat agents as untrusted identities with minimum, time-bound permissions. ²¹
NIS2 Directive (EU) 2022/2555	From Oct 2024	Cross-sector risk management and supply chain obligations for essential and important entities well beyond finance, including energy, health, transport, and digital infrastructure. ²⁹
SEC Item 1.05	Effective 2023	Applies to every US public company in any sector. Material incident disclosure within four business days of the materiality determination. ¹⁷

INSTRUMENT	STATUS	WHAT IT MEANS FOR A CODING AGENT PROGRAM
NYDFS industry letter on frontier AI	May 2026 Guidance	Addressed to CISOs of regulated entities. Recommends “additional testing and validation procedures, including human oversight, for AI-generated code prior to deployment in production environments.” DFS states it imposes no new requirements. ³
SR 26-2 model risk management	Apr 2026 Supervisory	Supersedes SR 11-7. Footnote 3 states generative and agentic AI models “are not within the scope of this guidance,” while directing that the organization’s own risk practices should determine appropriate controls. ⁴

The obligations nobody has asked about yet

Change management

One obligation reaches almost every mature organization and has so far gone largely unexamined. Any company subject to internal control over financial reporting maintains IT general controls over change management for in-scope systems, and those controls rest on an assumption that a named human authored, reviewed, and approved each change.

An agent opening a pull request against an in-scope system does not obviously break that assumption, but it strains it, and the strain stays invisible until an auditor asks who approved a change and the honest answer is a session transcript rather than a person. Settling how agent-authored changes are attributed and evidenced is cheaper before an external auditor raises it than after.

Third-party risk

No mature, published third-party risk framework yet treats model providers, agent vendors, and hosted MCP servers as a distinct diligence category. That is a real gap, and one an organization can close without waiting for anybody.

Three extensions do most of the work:

- Maintain an approved AI vendor list recording delivery method, integration depth, and data classes touched.
- Treat every third-party MCP server as a code dependency, pinned by content hash and reviewed like any other.
- Recognize that a vendor security certification does not guarantee that AI functionality arriving through a routine product update still meets your data handling requirements.

THE QUESTION THAT PRECEDES EVERY CONTROL PLANE

Resolve, per tool and before rollout, whether the model is reached as public consumer SaaS, as a contracted enterprise endpoint with retention and training terms in writing, or as a provider-hosted endpoint inside your own cloud account. The answer determines what network egress control, data residency, and audit evidence actually have to prove, and it determines where a secret lives after an agent has legitimately read it.

Sequencing that survives a steering committee



Figure 4. Horizons are indicative for an organization beginning from partial visibility. The sequence is the load-bearing element, not the day counts.

Three points on execution

More often than underinvestment, the failure mode we see is sequencing. Containment gets bought before inventory is finished, which secures the agents you already knew about and leaves the rest where they were.

Requirements first, then the tool

The vendor market for coding agent runtime security is young, moving quickly, and converging on syscall-level telemetry correlated with prompt and reasoning context. Categories are not yet stable enough to select from. Write the requirements against the six control planes and accept

that some will have no product answer in 2026. Configuration of agent permissions as a first-class, governed object is one of them.

Where a requirement has no product answer, close it explicitly rather than leaving it open. Build the narrow control yourself where the surface is small and stable, compensate with process and a named human gate where it is not, and formally accept and date the residual risk where neither is proportionate. An undocumented gap turns into an audit finding later. Give it a date, an owner, and a compensating control, and it becomes a position you can defend.

Route agent telemetry into the SOC you already have

The instinct to stand up a parallel monitoring stack for AI is expensive and also fragments response. The agentic event stream is a new data class, but it should terminate in the same SIEM, the same on-call rotation, and the same incident process as everything else.

Measure the controls rather than assuming them

Guardrail validation is the friction point most consistently skipped. Controls get deployed without a baseline for what good looks like, which means the first honest assessment of efficacy happens during an incident. Published benchmarks now make this testable in a way it was not eighteen months ago.

Where AHEAD starts with your agentic security program

AHEAD's Security for AI practice solves from architecture rather than from procurement. A typical engagement opens with a discovery of the live agent estate across developer endpoints, CI runners, and MCP servers, produces a requirements set mapped to the six control planes and to the regulatory instruments above, and scores the incumbent stack against it before any net-new spend is considered.

In most enterprise engagements, a material share of the requirements turn out to be closable with tooling the organization already licenses. The work is finding out which ones are not.

To discuss this point of view or to scope an agent estate discovery, contact your AHEAD account team or the Specialist Solutions Engineering practice.



Source register

1. Anthropic, The 2026 State of AI Agents Report, 2026.
2. Nx npm supply chain compromise, August 2025. Analysis: GitGuardian; Wiz Research, singularity's Aftermath; StepSecurity; Socket. Nx advisory GHSA-cxm3-wv7p-598c.
3. New York State Department of Financial Services, industry letter, Heightened Cybersecurity Risks Associated with Frontier AI Models, 21 May 2026.
4. Federal Reserve, OCC, and FDIC, SR 26-2: Revised Guidance on Model Risk Management, 17 April 2026, attachment n.3.
5. H. Pearce et al., Asleep at the Keyboard?, IEEE Symposium on Security and Privacy 2022. arXiv:2108.09293.
6. Veracode, 2026 GenAI Code Security Report, 28 July 2026.
7. S. Shukla, H. Joshi, R. Syed, Security Degradation in Iterative AI Code Generation, 2025. arXiv:2506.11022. Preprint.
8. Q. Zhan et al., University of Illinois Urbana-Champaign, InjecAgent, ACL 2024 Findings. arXiv:2403.02691.
9. N. Maloyan, D. Namiot, Prompt Injection Attacks on Agentic Coding Assistants, January 2026. arXiv:2601.17548. Preprint.
10. R. Marchand et al., Quantifying Frontier LLM Capabilities for Container Sandbox Escape, March 2026, revised August 2026. arXiv:2603.02277. Preprint.
11. Anthropic, GitHub Security Advisory GHSA-ff64-7w26-62rf, CVE-2026-25725, 6 February 2026. CVSS v4.0 7.7. Fixed in Claude Code 2.1.2.
12. M. M. Hasan et al., Queen's University, Model Context Protocol (MCP) at First Glance. arXiv:2506.13538.
13. J. Spracklen et al., University of Texas at San Antonio, We Have a Package for You!, USENIX Security 2025. arXiv:2406.10279.
14. M. Milanta, L. Beurer-Kellner, Invariant Labs, GitHub MCP Exploited, 26 May 2025.
15. CVE-2025-32711, Microsoft 365 Copilot. Discovered by Aim Security; remediated server-side June 2025.
16. Incident involving a Cursor agent and an over-scoped Railway API token, April 2026, as reported by The Register. Reported first-hand by the affected founder and corroborated on the record by Railway's chief executive. Not confirmed by Cursor.
17. CB Financial Services, Inc., SEC Form 8-K Item 1.05, filed 11 May 2026. Analysis: Wilson Sonsini, May 2026.
18. C. Li et al., Uber, ADR: An Agentic Detection System for Enterprise Agentic AI Security, MLSys 2026 Industry Track. arXiv:2605.17380.
19. E. Debenedetti et al., Google DeepMind and ETH Zurich, Defeating Prompt Injections by Design (CaMeL), 2025. arXiv:2503.18813.
20. S. Chen et al., UC Berkeley and Meta FAIR, SecAlign, ACM CCS 2025. arXiv:2410.05451.
21. ASD ACSC, CISA, NSA, Canadian Centre for Cyber Security, NCSC-NZ, NCSC-UK, Careful Adoption of Agentic AI Services, 30 April 2026.
22. NIST, SP 800-218A, 26 July 2024.
23. OCC Bulletin 2023-17, Third-Party Relationships, June 2023.
24. Regulation (EU) 2026/1744 (Digital Omnibus on AI), signed 8 July 2026, in force 27 July 2026.
25. OWASP GenAI Security Project, Top 10 for Agentic Applications, December 2025.
26. E. Debenedetti et al., ETH Zurich, AgentDojo, NeurIPS 2024. arXiv:2406.13352.
27. Regulation (EU) 2022/2554 (DORA), applicable from 17 January 2025.
28. ISO/IEC 42001:2023, Information technology. Artificial intelligence. Management system, December 2023.
29. Directive (EU) 2022/2555 (NIS2), transposition deadline 17 October 2024.



Combining cloud-native capabilities in software and data engineering with an unparalleled track record of modernizing infrastructure, we're uniquely positioned to help accelerate the promise of digital transformation.

Visit us at ahead.com.

Global Hub Offices

CHICAGO

444 W. Lake Street
Suite 3000
Chicago, IL 60606
United States

NEW YORK

500 5th Avenue
17th Floor
New York, NY 10010
United States

UNITED KINGDOM

310 Wharfedale Rd
Winnersh, Wokingham
RG41 5TP
United Kingdom

NETHERLANDS

Kerkenbos 10-123
6546 BJ Nijmegen
The Netherlands

INDIA

L-21, One Horizon Center
Golf Course Road, DLF Phase V
Sector 43, Gurugram 122002
India