← Back to Insights

STRAVORIS

Your Coding Agent Is Sending Your Code Somewhere

Executive Summary

Agentic coding tools — Claude Code, GitHub Copilot, Cursor, OpenCode, and their peers — have become fixtures in professional software development. Over 82% of developers now use AI coding assistants[4], and these tools are generating real chunks of production logic. But a fundamental question remains unanswered by most engineering organizations: **where is the code actually going?**

The answer, for the vast majority of tools, is to cloud-hosted inference endpoints operated by third parties. When a developer pastes a proprietary algorithm into a Claude Code prompt and asks it to debug, that algorithm is transmitted to Anthropic's servers. GitHub Copilot sends code snippets to GitHub/Microsoft's AI servers[11]. Even tools marketed as "open source" or "privacy-first" often have telemetry enabled by default and route inference through cloud endpoints[6]. The distinction between "open-source tool" and "private tool" is not what most developers assume.

This creates a live intellectual property exposure problem for enterprises with regulated or sensitive codebases. The scale of the gap between tool adoption and organizational governance is stark: 45% of developers admit to using unsanctioned code assistants at work[4], yet only 15% of organizations have updated their acceptable use policies to address AI tools[5]. Shadow AI incidents now account for 20% of all data breaches, carrying a cost premium of $4.63 million versus $3.96 million for standard breaches[5].

The tool landscape offers three distinct architectural tiers — Fully Local (zero egress), Self-Hosted Cloud (private infrastructure), and Managed Cloud (vendor endpoints with contractual protections) — but most organizations have not mapped their codebase sensitivity to these tiers. This research brief provides that mapping, analyzes the security postures of the major agentic coding tools, and offers a decision framework for engineering leaders who need to ship a policy before the next compliance audit asks for one.

Evidence Base & Methodology

Search Approach

This research was conducted on 28 March 2026 using web searches across seven research angles: recent developments, industry data, counterarguments, case studies, technical analysis, vendor landscape, and historical context. Three seed URLs from the originating idea file were fetched directly. Additional targeted fetches were performed on the most data-rich pages identified through search results.

Sources Consulted

16 primary sources were used, spanning vendor documentation (Anthropic, GitHub, Microsoft), independent security research (MintMCP, Bright Security, DryRun Security), industry surveys (Stack Overflow, Gartner, IBM, ISACA, Microsoft), developer community analysis (Hacker News, Reddit discussions), and technical guides (Graphite, Knostic, Checkmarx). Evidence spans from mid-2025 through March 2026.

Notable Gaps

DataCamp's comparison of OpenCode vs. Claude Code (a key seed source) returned a 403 access error and could not be fetched directly. Claims attributed to that comparison are drawn from the idea file and corroborated against other sources. Anthropic's detailed data retention timelines for Claude Code specifically (as distinct from the broader Claude platform) are not publicly documented. Cursor's enterprise data handling terms beyond "Privacy Mode" are sparse in public documentation.

The Data Flow Problem: Where Your Code Actually Goes

The Default State Is Cloud Transmission

Every mainstream agentic coding tool, in its default configuration, sends code context to cloud-hosted AI inference endpoints. This is not a bug — it is the fundamental architecture. Large language models require substantial compute resources that exceed what developer laptops can provide, and commercial tools route to their provider's infrastructure accordingly.

What varies is the degree of transparency and control:

- **GitHub Copilot** sends code snippets temporarily to GitHub's AI servers for generating suggestions[11]. On Business and Enterprise tiers, prompts and suggestions are not retained, and code is explicitly excluded from model training[3].
- **Claude Code** processes code and conversations through Anthropic's AI models, which involves sending data to Anthropic's servers[8]. Consumer-tier users (Free, Pro, Max) have their data used for training as of September 2025. Only Commercial/Enterprise tiers (Claude for Work, API, Bedrock) prohibit training by default[8].
- **Cursor** offers a "Privacy Mode" with zero-retention handling where code is not stored by model providers or used for training[7]. However, its MCP integration creates separate data-access paths that require separate monitoring.
- **OpenCode** markets a privacy-first architecture and claims it does not store code or context[6]. However, it still routes inference through cloud endpoints by default (Anthropic, OpenAI, or other providers). Its "Air-gapped Mode" with Ollama provides genuine local execution, but this is an opt-in configuration, not the default.

The "Open Source" Misconception

A recurring pattern in community discussions: developers equate "open source" with "private by default." The OpenCode controversy in early 2026 crystallized this. Despite being marketed as an open-source local coding environment, OpenCode routes inference traffic through cloud endpoints in its standard configuration. Community members discovered telemetry enabled by default, which raised questions about what data was being collected beyond the inference calls themselves[6].

The confusion is understandable. "Open source" describes the license on the *client code* — it says nothing about where the *inference* happens. A developer running OpenCode with the default Anthropic or OpenAI provider configuration is sending their code to the same cloud endpoints as a Claude Code or Copilot user. The difference is that OpenCode *also* supports fully local inference via Ollama — but the developer must explicitly configure it.

Exfiltration Vectors Beyond Inference

Cloud inference is the most obvious data flow, but not the only one. Several additional vectors have been documented:

- **IDE prompt injection:** Attackers hide invisible white-text instructions in READMEs or comments that direct AI assistants to exfiltrate secrets[4].
- **Malicious extensions:** Microsoft documented malicious AI assistant extensions that harvested full URLs and AI chat content, affecting approximately 900,000 installs across more than 20,000 enterprise tenants[4].
- **CamoLeak (CVSS 9.6):** A demonstrated attack enabling silent exfiltration of private repository code through invisible prompt injection in GitHub Copilot[7].
- **Environment variable leakage:** Claude Code's CVE-2026-21852 (CVSS 5.3) demonstrated API key exfiltration through environment variables[7].

The Tool Landscape: Security Postures Compared

Documented Vulnerabilities

Each major tool has accumulated a vulnerability record through 2025–2026. The following table summarizes the most significant documented CVEs.

Tool	CVE	CVSS	Description
GitHub Copilot	CamoLeak	9.6	Silent exfiltration of private repo code via invisible prompt injection
Claude Code	CVE-2025-59536	8.7	Remote code execution via malicious project configuration files
Cursor	CVE-2025-54135	8.6	Untrusted MCP remote code execution
GitHub Copilot	CVE-2025-62449	6.8	Path traversal enabling unauthorized file access
Claude Code	CVE-2026-21852	5.3	API key exfiltration through environment variables
Cursor	CVE-2025-59944	Critical	Case-sensitivity bypass enabling persistent RCE via .CURSOR/mcp.json
Claude Code	CVE-2026-25725	—	Sandbox bypass enabling unauthorized file system access
GitHub Copilot	CVE-2025-62453	5.0	Improper AI output validation

No tool is immune. The presence of high-severity vulnerabilities across all three major tools demonstrates that security posture is not a differentiator between specific vendors — it is a category-level risk that requires organizational controls regardless of which tool is selected.

Data Handling and Privacy Policies

The following table compares the privacy postures of the major agentic coding tools across key enterprise dimensions.

Dimension	GitHub Copilot (Enterprise)	Claude Code (Commercial)	Cursor (Privacy Mode)	OpenCode (Air-gapped)
Code sent to cloud	Yes (Microsoft/GitHub servers)	Yes (Anthropic servers)	Yes (model provider servers)	No (Ollama local inference)
Training on code	No (Business/Enterprise tiers)	No (Commercial terms only)	No (Privacy Mode)	No (local models)
Data retention	Prompts: not retained; engagement: 2 years	Zero-retention available (enterprise API)	Zero-retention (Privacy Mode)	None (fully local)
IP indemnification	Yes (Enterprise tier)	Not publicly documented	Not publicly documented	N/A (open source)
SSO/SAML	Yes (native)	API-level implementation	Limited	N/A
DPA available	Yes (Microsoft DPA)	Yes (auto-incorporated in Commercial Terms)	Not publicly documented	N/A
Audit logging	Organization-level controls	API usage logging	Limited	Local logs only
Enterprise pricing	$39/user/month	Usage-based API pricing	$40/user/month (Business)	Free (infra costs only)

The Consumer-Enterprise Privacy Gap

A critical distinction that many developers miss: the privacy guarantees of a tool depend on the *tier*, not the *tool*. Anthropic's September 2025 terms update made this explicit — Claude trains on all data from Free, Pro, and Max plans, including when those accounts use Claude Code[8]. Only Commercial Terms (Claude for Work, API, Bedrock) prohibit training. Similarly, GitHub Copilot's free tier may use interactions for model improvement, while Business and Enterprise tiers explicitly exclude customer code from training[3].

This means a developer using Claude Code on a personal Pro subscription to debug their employer's code has just sent that code to Anthropic under terms that permit training. The tool is the same. The data handling is not.

The Shadow AI Crisis: Developers Operating Without Policy

The Scale of Unsanctioned Use

The gap between AI tool adoption and organizational governance is one of the most consequential findings in this research. The data from multiple independent surveys converges on a consistent picture:

- **45%** of developers admit to using unsanctioned code assistants at work (Stack Overflow 2025 Developer Survey)[4]
- **52%** of developers do not use IT-approved tools[4]
- **69%** of organizations suspect or have evidence that employees use prohibited public GenAI tools (Gartner 2025 cybersecurity survey)[5]
- **71%** of UK employees admitted to using unapproved AI tools at work, with 51% doing so at least weekly (Microsoft research)[5]
- **98%** of organizations report unsanctioned AI use[5]

Meanwhile, organizational governance lags far behind:

- Only **15%** of organizations have updated their acceptable use policies to address AI tools (ISACA)[5]
- Only **37%** of organizations have AI governance policies[5]
- Shadow AI grows **120% year-over-year**[3]

The Financial Impact

Shadow AI is not an abstract governance concern. IBM's 2025 Cost of Data Breach Report found that shadow AI incidents now account for **20% of all breaches** and carry a cost premium: **$4.63 million versus $3.96 million** for standard breaches[5]. Gartner projects that 1 in 4 compliance audits in 2026 will include specific inquiries into AI governance[5].

The risk compounds in coding-specific scenarios. Research cited by MintMCP found that secret leakage rates run **40% higher** in repositories using Copilot — 6.4% versus a 4.6% baseline[7]. Separately, studies show up to 40% of AI-generated code suggestions may introduce potential security vulnerabilities[11]. When developers use these tools without organizational oversight, both the data exposure risk and the code quality risk operate unchecked.

Why Developers Use Unapproved Tools

The Anthropic-OpenCode controversy in January 2026 illustrates the dynamic. When Anthropic blocked Claude Code subscriptions from being used through third-party tools, users who had been paying $100–$200/month for Claude Max lost the ability to use that subscription with OpenCode. Rails creator DHH

called it "very hostile to users," Hacker News erupted, and OpenCode gained 18,000 GitHub stars in two weeks[6]. Developers will find and adopt the tool that works for them, regardless of whether their organization has approved it.

This is the core dynamic that organizational policy must account for: restricting tools without providing approved alternatives does not reduce usage — it pushes usage underground where it becomes invisible to security teams.

The Three-Tier Framework: Matching Sensitivity to Architecture

Tier Definitions

The agentic coding tool landscape maps to three distinct architectural tiers, each with different data flow characteristics, cost profiles, and capability trade-offs.

Tier	Architecture	Data Egress	Examples	Trade-offs
Tier 1: Fully Local	Local model, local inference, zero cloud dependency	None	Ollama + Continue.dev, OpenCode Air-gapped, Tabby, FauxPilot	Lower model capability; requires local GPU; no vendor support
Tier 2: Self-Hosted Cloud	Model runs on organization's own cloud infrastructure	Within org perimeter only	AWS Bedrock (private endpoint), Azure OpenAI (private), self-hosted vLLM	Higher infra cost; operational burden; model updates lag
Tier 3: Managed Cloud	Vendor-hosted inference with contractual protections	To vendor (covered by DPA/commercial terms)	GitHub Copilot Enterprise, Claude Code (Commercial), Cursor Business	Best model quality; vendor lock-in; data leaves perimeter

The Capability Gap Is Closing

A common objection to Tier 1 (Fully Local) is that local models cannot match cloud-hosted frontier models. This was true in 2024 but the gap is narrowing. Current benchmarks show DeepSeek Coder V2 and Codestral achieving completion accuracy within 5–10% of Copilot on standard benchmarks[9]. DeepSeek Coder specifically achieves 94% accuracy on completion tasks versus Copilot's 89%[9]. For code completion and simple refactoring tasks, local models are now viable. For complex agentic workflows — multi-file reasoning, architectural decisions, long-context debugging — frontier cloud models retain a significant advantage.

Decision Matrix: Codebase Sensitivity to Tool Tier

The following matrix maps codebase sensitivity categories to the minimum acceptable tool tier.

Codebase Category	Examples	Minimum Tier	Rationale
Regulated / classified data	HIPAA-covered code, ITAR/EAR controlled, financial trading algorithms	Tier 1 only	Regulatory frameworks prohibit data egress; no DPA is sufficient
Trade secrets / core IP	Proprietary algorithms, pre-patent code, competitive differentiators	Tier 1 or Tier 2	IP exposure risk too high for third-party infrastructure; self-hosted acceptable if perimeter is controlled
Internal tooling / non-regulated	Internal dashboards, build scripts, DevOps automation	Tier 2 or Tier 3	Low IP value; DPA-covered managed cloud acceptable
Open-source / pre-IP-protected	Open-source contributions, published libraries, public documentation	Any tier	Code is already public or intended to be; no data sensitivity concern

Implementation Considerations

Two practical realities complicate this framework:

First, developers context-switch constantly. A developer working on a regulated codebase in the morning and an internal tool in the afternoon would need to switch between Tier 1 and Tier 3 tools — or maintain parallel tool configurations. This friction pushes developers toward a single tool for everything, which usually means the most capable (and least private) option.

Second, the "commercial tier" requirement is easily violated. A developer using Claude Code with a personal Pro subscription is on consumer terms where training is permitted. The same developer using Claude Code through their company's API account is on commercial terms where training is prohibited[8]. The tool is identical. The terms are not. Organizations must enforce which *accounts* are used, not just which *tools*.

Key Assumptions & Uncertainties

What the Evidence Does Not Resolve

- **Actual incident rates from coding tool data exposure.** While IBM reports shadow AI accounts for 20% of breaches, no public data isolates breaches specifically caused by AI coding tools sending proprietary code to vendor endpoints. The risk is well-documented in theory; confirmed incidents attributable to this specific vector are not publicly reported.
- **Anthropic's detailed data retention for Claude Code.** Anthropic offers zero-retention agreements for enterprise API customers, but the exact data lifecycle for Claude Code sessions — what is logged, for how long, and where — is not detailed in public documentation[7].
- **Cursor's enterprise data handling beyond Privacy Mode.** Cursor's Privacy Mode claims zero-retention handling, but the specifics of its MCP integration data paths — which create separate data-access routes — are not fully documented[7].
- **The real-world capability gap for agentic tasks on local models.** Benchmark numbers for code completion (DeepSeek at 94% vs. Copilot at 89%) are promising, but agentic workflows — multi-step reasoning, tool use, file navigation — are not well-benchmarked for local models. The gap for these tasks is likely larger than completion benchmarks suggest.

Where Expert Opinion Diverges

- **Whether DPAs and commercial terms provide sufficient protection.** Some security professionals argue that a DPA from Anthropic or Microsoft, combined with no-training guarantees, makes Tier 3 acceptable for most enterprise code. Others argue that any data egress to a third party creates unacceptable risk for sensitive IP, regardless of contractual protections — contracts are legal remedies, not technical controls.
- **Whether local models will close the gap for agentic tasks.** The trajectory of open-source model performance is strong, but whether local models will match frontier capabilities for complex agentic coding within 12–18 months is genuinely uncertain.

Assumptions in This Analysis

- Vendor claims about data handling (no training, zero retention) are taken at face value. Independent audits of these claims are sparse.
- The shadow AI statistics from Stack Overflow, Gartner, and Microsoft surveys may overcount or undercount depending on self-reporting bias. The directional conclusion — that unsanctioned use is widespread — is robust across all sources.
- CVE data reflects publicly disclosed vulnerabilities. Additional vulnerabilities may exist in responsible disclosure or may not have been discovered.

Strategic Implications & Actionable Insights

1. **Ship a policy before the audit asks for one.** Gartner projects 1 in 4 compliance audits in 2026 will include AI governance inquiries[5]. The absence of a written policy for AI coding tool usage is now an audit finding, not a backlog item. A one-page policy that maps codebase categories to approved tool tiers is a minimum viable response.

2. **Enforce account tiers, not just tool choice.** The privacy guarantees of Claude Code, Copilot, and Cursor are tier-dependent. A developer using Claude Code on a personal Pro account exposes code to training; the same tool on a commercial API account does not[8]. Organizations must provide and enforce commercial-tier accounts, not just approve tool names.

3. **Provide approved alternatives or accept shadow AI.** With 45–52% of developers using unsanctioned tools[4] and shadow AI growing 120% YoY[3], prohibition without provision is not a viable strategy. Organizations must provide an approved tool at each tier their developers need, or accept that developers will find their own.

4. **Treat "open source" as a licensing descriptor, not a privacy guarantee.** OpenCode, the most popular open-source agentic coding tool, routes to cloud endpoints by default[6]. "Open source" means the client code is inspectable — it says nothing about where inference happens. Evaluate data flow architecture, not license type.

5. **Invest in Tier 1 capability for your most sensitive code.** Local models have closed the gap to within 5–10% of cloud models for code completion[9]. For regulated or trade-secret code, the minor capability trade-off is far less costly than the IP exposure risk. Continue.dev + Ollama or Tabby provides a zero-egress development environment that is production-viable today.

6. **Monitor for exfiltration vectors beyond inference.** Prompt injection attacks, malicious extensions (900,000 installs across 20,000+ enterprise tenants[4]), and environment variable leakage represent attack surfaces that exist regardless of which tier or tool is selected. Content exclusion rules, extension allowlists, and secret scanning are complementary controls.

7. **Require DPA and no-training guarantees for any Tier 3 deployment.** If managed cloud tools are approved for non-regulated code, require at minimum: a signed DPA, explicit no-training guarantees, defined data retention windows, and SOC 2 Type II or ISO 27001 certification from the vendor[11]. GitHub Copilot Enterprise and Anthropic's Commercial Terms meet these criteria; free tiers and consumer subscriptions do not.

References

1. Anthropic Engineering, "Claude Code: Sandboxing," Anthropic, 2026. https://www.anthropic.com/engineering/claude-code-sandboxing. Accessed 28 March 2026.
2. DryRun Security, "Top AI SAST Tools 2026," DryRun Security Blog, 2026. https://www.dryrun.security/blog/top-ai-sast-tools-2026. Accessed 28 March 2026.
3. Microsoft Community Hub, "Demystifying GitHub Copilot Security Controls: Easing Concerns for Organizational Adoption," Microsoft, 2025. https://techcommunity.microsoft.com/blog/azuredevcommunityblog/demystifying-github-copilot-security-controls-easing-concerns-for-organizational/4468193. Accessed 28 March 2026.
4. IT Pro, "Shadow AI Is Creeping Its Way Into Software Development," IT Pro, 2025–2026. https://www.itpro.com/software/development/shadow-ai-is-creeping-its-way-into-software-development. Accessed 28 March 2026.
5. JumpCloud, "11 Stats About Shadow AI in 2026," JumpCloud Blog, 2026. https://jumpcloud.com/blog/11-stats-about-shadow-ai-in-2026. Accessed 28 March 2026.
6. InfoQ, "OpenCode: An Open-source AI Coding Agent Competing with Claude Code and Copilot," InfoQ News, February 2026. https://www.infoq.com/news/2026/02/opencode-coding-agent/. Accessed 28 March 2026.
7. MintMCP, "Claude Code vs Cursor vs Copilot: 2026 Security Comparison," MintMCP Blog, 2026. https://www.mintmcp.com/blog/claude-code-cursor-vs-copilot. Accessed 28 March 2026.
8. Anthropic Privacy Center, "I Have a Zero Data Retention Agreement with Anthropic. What Products Does It Apply To?" Anthropic, 2026. https://privacy.claude.com/en/articles/8956058. Accessed 28 March 2026.
9. Local AI Master, "Best Local AI for Coding 2026: 10 Models Tested & Ranked," Local AI Master, 2026. https://localaimaster.com/blog/best-local-ai-models-programming. Accessed 28 March 2026.
10. Microsoft Security Blog, "Malicious AI Assistant Extensions Harvest LLM Chat Histories," Microsoft, 5 March 2026. https://www.microsoft.com/en-us/security/blog/2026/03/05/malicious-ai-assistant-extensions-harvest-llm-chat-histories/. Accessed 28 March 2026.
11. Graphite, "Privacy and Security Considerations When Using AI Coding Tools," Graphite Guides, 2026. https://graphite.com/guides/privacy-security-ai-coding-tools. Accessed 28 March 2026.
12. Anthropic Privacy Center, "How Do I View and Sign Your Data Processing Addendum (DPA)?" Anthropic, 2026. https://privacy.claude.com/en/articles/7996862. Accessed 28 March 2026.
13. AMST Legal, "Anthropic's Claude AI Updates — Impact on Privacy & Confidentiality," AMST Legal, 2025. https://amstlegal.com/anthropics-claude-ai-updated-terms-explained/. Accessed 28 March 2026.
14. DEV Community, "Self-Host Your AI Code Assistant With Continue.dev + Ollama," DEV Community, 2026. https://dev.to/signal-weekly/self-host-your-ai-code-assistant-with-continuedev-ollama-vs-code-copilot-without-the-3ofe. Accessed 28 March 2026.
15. Hacker News, "OpenCode — Open Source AI Coding Agent," Y Combinator, 2026. https://news.ycombinator.com/item?id=47460525. Accessed 28 March 2026.
16. GitGuardian Blog, "GitHub Copilot Privacy: Key Risks and Secure Usage Best Practices," GitGuardian, 2025. https://blog.gitguardian.com/github-copilot-security-and-privacy/. Accessed 28 March 2026.

Author: Krishna Gandhi Mohan

Web: stravoris.com

LinkedIn: linkedin.com/in/krishnagmohan

This research brief is part of the AI Tactical Playbook series by Stravoris.

STRAVORIS

INNOVATE. INTEGRATE. ELEVATE.