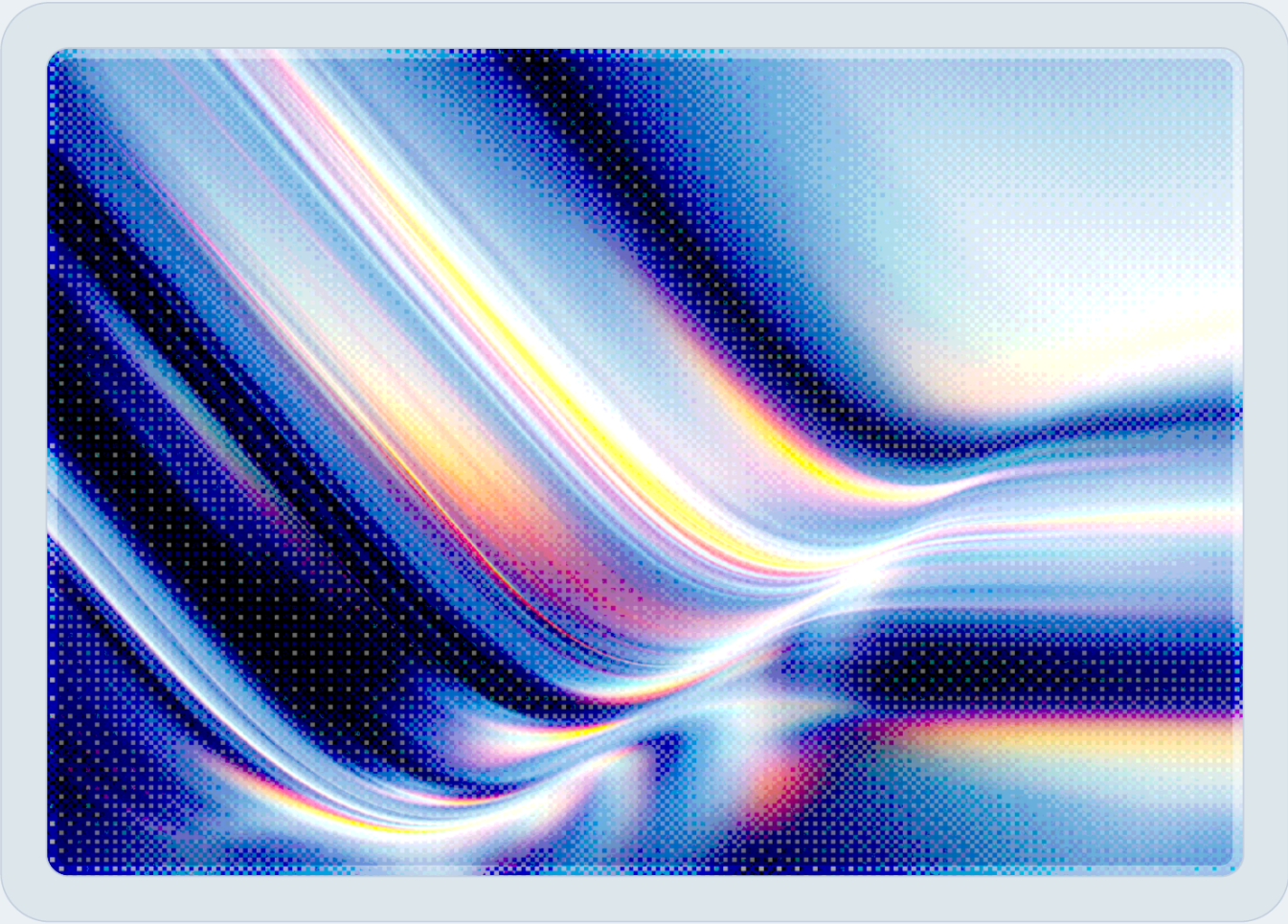
CSAI Foundation | Cloud Security Alliance

GhostApproval: A Trust Boundary Gap in Six AI Coding Agents

How Symlink Confusion Defeats Human-in-the-Loop Review in Amazon Q, Claude Code, Cursor, and Others

2026-07-15

AI-assisted Rapid Research



CSAI

CSA cloud security alliance®

Key Takeaways

- Wiz Research disclosed GhostApproval on July 8, 2026, a systemic vulnerability pattern affecting six leading AI coding assistants — Amazon Q Developer, Anthropic Claude Code, Augment, Cursor, Google Antigravity, and Windsurf — that lets a malicious repository redirect an approved file edit to a sensitive location outside the project workspace [1][2].
- The flaw combines a decades-old Unix weakness, symlink following (CWE-61), with a modern interface failure, misrepresentation of the action a user is asked to approve (CWE-451): the agent's confirmation dialog shows the name of a symlinked file rather than the real system path it resolves to [1].
- In at least two of the affected tools, Claude Code and Augment, the agent's own internal reasoning correctly identified the symlink's dangerous target, yet the prompt shown to the human reviewer displayed only the benign-looking filename, meaning the human-in-the-loop safeguard approved an action neither the human nor, in effect, the interface layer accurately represented [1][3].
- Amazon (CVE-2026-12958), Cursor (CVE-2026-50549), and Google treated the report as a vulnerability and shipped fixes between May and June 2026; Augment and Windsurf acknowledged the report but had not shipped fixes as of the July 8 disclosure; Anthropic disputed the classification, calling the scenario "outside our threat model" while still adding a symlink warning to Claude Code in early February 2026 [1][2][4].
- The disclosure underscores that "human-in-the-loop" review is only as trustworthy as the information the interface presents to the human, and that classic filesystem security principles — resolving paths before acting on them — remain directly applicable to agentic AI tools built on conventional operating systems [3].

Background

AI coding assistants embedded in editors and command-line tools have converged on a common safety pattern over the past two years: the agent proposes a file edit, a command, or a configuration change, and a human developer reviews and approves it before it takes effect. This human-in-the-loop model is the primary control that vendors point to when asked how their tools prevent an autonomous agent from taking unwanted or destructive actions on a developer's machine. In practice, vendors commonly point to human review as the reason broader sandboxing or privilege restriction is unnecessary, a pattern reflected in security documentation for several of the tools discussed in this note: if a human always signs off on file writes, the reasoning goes, the blast radius of an over-eager or manipulated agent is bounded by what that human is willing to approve.

That reasoning depends on an assumption that is easy to overlook: the approval dialog must accurately describe the action being authorized. Wiz Research's GhostApproval disclosure, published July 8, 2026, demonstrates that this assumption failed across a majority of the market's leading AI coding assistants [1][2]. Researcher Maor Dokhanian and the Wiz team found that a malicious public repository — the kind a developer might clone to evaluate a library, review a pull request, or follow a tutorial — can contain a symbolic link disguised as an ordinary configuration file. When the agent is instructed, often through nothing more suspicious-looking than a README asking it to "set up the workspace," to edit that file, several tools follow the symlink to its actual target and write there instead, while still displaying the harmless-looking symlink name in the confirmation prompt the developer sees [1][3].

The Register's coverage of the disclosure frames the underlying cause bluntly: this is a Unix-era security problem that autonomous coding agents have reintroduced by failing to apply a lesson the broader software industry learned decades ago, that a path on disk cannot be trusted until it has been resolved to its true target [3]. What makes GhostApproval notable is not the novelty of the symlink attack itself, which long predates AI tooling, but the fact that six independently built products, developed by vendors including Amazon, Anthropic, and Google, converged on the same gap in their approval workflows, and that in at least two cases, Claude Code and Augment, the vendor's internal agent reasoning demonstrably recognized the danger before the confirmation prompt was rendered [1].

Security Analysis

The Attack Mechanism

The GhostApproval attack chain begins with an attacker publishing or contributing to a repository that a target developer is likely to open in an AI-assisted editor or command-line agent. Inside that repository, the attacker plants a symbolic link with an innocuous name, such as `project_settings.json`, that actually points to a sensitive file elsewhere on the developer's filesystem, for example `~/.ssh/authorized_keys` or a shell startup file like `~/.zshrc` [1]. The repository also contains instructions, often embedded in a README or a code comment, directing the agent to perform routine-sounding setup work, such as configuring the project's settings file. When the developer asks their AI coding assistant to follow those instructions, or the agent proactively suggests doing so, the agent resolves the write target through the filesystem rather than validating that the target stays within the project directory, follows the symlink, and writes attacker-controlled content, such as an SSH public key, to the real target outside the workspace [1][3]. If that target is `~/.ssh/authorized_keys`, the result is that the attacker gains remote access to the developer's machine using their own credentials, entirely bypassing any workspace boundary the tool was designed to enforce.

Wiz identified two distinct weaknesses compounding to produce this outcome. The first is CWE-61, symlink following: the agents did not canonically resolve file paths before performing write operations, so a symlink inside the workspace could silently redirect a write to anywhere the operating system user had permission to write, regardless of whether that location fell inside or outside the project sandbox [1]. The second, and the more consequential from a human-oversight perspective, is CWE-451, UI misrepresentation of critical information: even in cases where the agent's own path-resolution logic correctly identified that a write target lay outside the workspace, the confirmation dialog presented to the developer displayed only the symlink's local name, not the resolved destination [1]. Dokhanian summarized the resulting gap directly: "The user approves what they believe is a harmless local edit; the agent writes to a sensitive file outside of the project workspace" [1].

Why Human-in-the-Loop Did Not Catch It

The severity of GhostApproval comes less from the symlink trick itself, a well-understood filesystem hazard, than from what it reveals about the reliability of the approval step vendors rely on as their primary safety control. Wiz's research found that in both Claude Code and Augment, the agent's internal reasoning trace explicitly identified the dangerous resolved target before the confirmation prompt was shown, yet the prompt itself displayed only the benign filename, giving the developer no way to know what they were actually authorizing [1]. Dokhanian's broader point applies across the affected tools:

"Human-in-the-loop isn't always the safety net it appears to be. When the confirmation prompt hides critical information, developers can't make informed decisions" [1]. A review gate that shows accurate information about a low-risk action but inaccurate information about a high-risk one is arguably worse than no review gate at all, because it creates a false sense of assurance that a decision has been deliberately and correctly authorized when it has not.

Wiz also documented a second, compounding failure mode in Windsurf and Amazon Q: both tools wrote the file to disk before displaying the confirmation dialog, making the "approval" a notification of an already-completed action rather than a gate that prevents it [1]. In that design, even a developer who noticed something was wrong and declined the prompt would already have had the write executed; the confirmation step functioned, at best, as an opportunity to attempt an undo rather than as a genuine authorization gate. This distinction matters for how organizations evaluate agentic tools going forward: a control that executes an action and then asks for retroactive confirmation does not provide the same guarantee as a control that blocks the action pending approval, even though both may present visually identical dialogs to the end user.

Divergent Vendor Responses

Vendor responses to Wiz's disclosure split along a fault line that is itself informative about how the industry currently reasons about agent trust boundaries. Amazon assigned CVE-2026-12958 (rated High severity) to the flaw in Amazon Q Developer and shipped a fix in Language Server version 1.69.0 [1][4]. Cursor assigned CVE-2026-50549 (rated Critical) and resolved the issue in version 3.0 [1][2]. Google treated its finding in Antigravity as Critical and deployed a fix on May 22, 2026, ahead of the public disclosure, though as of Wiz's publication a CVE number remained pending [1]. Augment and Windsurf both acknowledged receiving the report but had not shipped fixes as of the July 8 disclosure date; Augment characterized the underlying trust relationship as intrinsic to how coding agents function, stating that "a coding agent needs to be able to edit and run code to be useful; when it does that, it operates under your credentials" [1].

Anthropic's response diverges from the rest of the field, and represents the only case among the six vendors where the underlying trust-boundary question was explicitly disputed rather than treated as a defect. Rather than treating the symlink-confusion behavior in Claude Code as a vulnerability warranting a CVE, Anthropic disputed the classification, arguing that a developer who chooses to trust a directory and approves an edit within it owns the consequences of that decision, and that the scenario therefore falls "outside our threat model" [1][2]. Anthropic had, notably, already shipped a symlink warning in Claude Code in early February 2026, predating the formal disclosure timeline, which suggests the

company was aware of symlink-related risk in principle even while declining to treat this specific attack chain as a defect requiring a CVE [2]. The table below summarizes the disclosure and remediation status for each affected tool as reported by Wiz.

Vendor / Product	Severity	CVE	Status as of July 8, 2026
Amazon Q Developer	High	CVE-2026-12958	Fixed (Language Server v1.69.0+)
Cursor	Critical	CVE-2026-50549	Fixed (v3.0)
Google Antigravity	Critical	Pending	Fixed (deployed May 22, 2026)
Anthropic Claude Code	Disputed by vendor	None assigned	Symlink warning added; classification disputed
Augment	Critical (per Wiz)	None assigned	Acknowledged; unpatched
Windsurf	Critical (per Wiz)	None assigned	Acknowledged; unpatched

Wiz's disclosure timeline shows reports were submitted to all six vendors between February 10 and March 6, 2026, giving vendors roughly four to five months to respond before the July 8 public disclosure. The staggered pace of remediation, ranging from patches shipped in May to two vendors still unpatched at disclosure, illustrates how unevenly the industry currently treats filesystem trust-boundary defects — a pattern this note's authors did not observe to the same degree in coverage of more conventional vulnerability classes such as injection or authentication bypass, though a rigorous cross-class comparison is beyond this note's scope [1][2].

A Category-Level Problem, Not Six Isolated Bugs

Both Wiz and The Register frame GhostApproval as a category-level design gap rather than a set of unrelated implementation bugs, and the pattern across six independently engineered products supports that framing [1][3]. Each affected tool made the same underlying design choice: trust that the file path a user or an instruction file names is the path that will actually be written to, without validating that assumption against the filesystem's own symlink-resolution behavior. That choice was made independently by engineering teams at Amazon, Anthropic, Cursor, Google, and at least two smaller

vendors, suggesting the gap reflects a genuine blind spot in how the industry has approached agent sandboxing rather than a single team's oversight. As agentic coding tools race to ship increasingly autonomous capabilities, from multi-file refactors to test generation to dependency management, the boundary between what a user has actually authorized and what the underlying operating system will actually execute needs a clearer, more conservative default than "display the path the instruction named and trust it."

Recommendations

Immediate Actions

Organizations using Amazon Q Developer, Cursor, or Google Antigravity should confirm they are running the patched versions identified by each vendor — Language Server 1.69.0 or later for Amazon Q, version 3.0 or later for Cursor, and the post-May 22, 2026 build for Antigravity — since earlier versions remain exposed to the full GhostApproval attack chain [1][4]. Teams using Augment or Windsurf should treat both tools as currently unpatched against this vulnerability class and apply compensating controls, such as restricting these agents from operating in directories containing sensitive credential files, until each vendor ships a fix. Teams using Claude Code should be aware that Anthropic has not classified this behavior as a vulnerability and does not plan a fix beyond the symlink warning already shipped, and should factor that stance into their own risk acceptance decisions rather than assuming vendor remediation is forthcoming [1][2]. Across all six tools, security teams should audit repositories cloned from external or untrusted sources for symlinks pointing outside the project directory before allowing an AI coding agent to operate on them, since a symlink pointing outside the project directory, particularly one embedded without clear explanation in a repository's setup instructions, warrants manual review before an AI coding agent is permitted to act on it.

Short-Term Mitigations

Because GhostApproval reflects a design pattern repeated across the industry rather than a single patchable defect, organizations should not treat a vendor's fix as closing the underlying risk category. Development teams should configure AI coding agents, where the tool supports it, to run with reduced filesystem privileges or inside a container or virtual machine that does not have direct access to the host's SSH keys, shell configuration files, or other credential material, so that even an undetected symlink-confusion write cannot reach a sensitive target. Security teams should extend existing secure code review practices to explicitly include symlink inspection for any repository originating outside the

organization, treating an unexplained symlink the way a reviewer would treat an unexplained pre-commit hook or postinstall script: as something requiring justification before it is trusted. Organizations should also request, from any AI coding assistant vendor under evaluation or already in use, an explicit statement of whether the tool resolves symlinks to their true target before displaying a confirmation prompt, since Wiz's findings show this single design detail was the deciding factor in whether human review functioned as intended.

Strategic Considerations

GhostApproval is a useful case study for security leaders assessing how much weight to place on "human-in-the-loop" as a stated control in vendor security questionnaires and marketing claims. The disclosure shows that a review gate is only as trustworthy as the fidelity of the information presented at that gate, and that a vendor's claim of human oversight should prompt a follow-up question about whether the interface can misrepresent the action being approved, not just whether a human is nominally in the approval path. Anthropic's position, that a developer who trusts a directory and approves an edit within it accepts the resulting risk, is a defensible engineering philosophy in isolation, but it places the full burden of accurate risk assessment on a developer who, per Wiz's own findings, was shown a confirmation dialog that did not disclose the information needed to make that assessment. Organizations building governance frameworks for agentic AI coding tools should treat trust-boundary and confirmation-fidelity failures as a distinct risk category worth evaluating explicitly, alongside more familiar concerns like model output accuracy or data handling, when selecting and configuring these tools for developer use.

CSA Resource Alignment

GhostApproval's core failure, an autonomous agent taking a consequential action after a human believed they had reviewed and constrained it, is the central risk that CSA's survey report Autonomous but Not Controlled: AI Agent Incidents Now Common in Enterprises [7] documents at an organizational scale. That research finds persistent gaps in monitoring and lifecycle oversight of AI agents even where organizations believe governance controls are in place, and GhostApproval is a concrete technical instance of exactly that gap: a control (the confirmation dialog) that organizations and developers reasonably assumed was functioning as a safeguard, but which the underlying tool had quietly rendered unreliable through incomplete path resolution. CSA's companion survey, Securing Autonomous AI Agents [8], reaches a related conclusion from an identity and governance angle, finding that enterprise AI agent adoption is outpacing the authentication, traceability, and oversight capabilities needed to govern it; GhostApproval demonstrates that even the oversight capability organizations do have, direct

human approval of file writes, cannot be assumed reliable without verifying what information the approval mechanism actually surfaces. A third CSA survey, Identity and Access Gaps in the Age of Autonomous AI [9], sharpens this point further: it finds organizations relying on inherited permissions and fragmented ownership rather than identity-centric controls to govern what AI agents can reach, which is precisely the mechanism GhostApproval exploits, since the agent writes to the filesystem using the developer's own credentials with no independent identity boundary constraining it, regardless of whether the confirmation dialog itself can be trusted.

The Agentic Trust Framework (ATF), the CSA/CSAI Foundation-stewarded Zero Trust specification for governing autonomous AI agents, is directly applicable to the graduated-autonomy question GhostApproval raises [5]. ATF's four-level maturity model, in which agents earn broader autonomy (from read-only "Intern" through "Principal" acting independently within defined boundaries) rather than receiving it by default, offers organizations a structured way to decide how much unsupervised filesystem access an AI coding agent should hold, rather than relying solely on a per-action confirmation dialog whose reliability, as this disclosure shows, cannot be taken for granted. Applying ATF's behavior and segmentation elements to coding agents specifically would mean bounding which directories and file types an agent can write to outside of an explicitly elevated, audited state, providing a defense-in-depth layer that does not depend on the confirmation prompt alone being trustworthy.

Finally, organizations building a formal risk assessment or vendor evaluation process for AI coding assistants should incorporate the AI Controls Matrix (AICM) v1.1's application and infrastructure security domains, which address the need for AI system components to enforce least-privilege access and validate inputs and resource references before acting on them — a control objective GhostApproval shows was not consistently met even by well-resourced vendors, and one organizations should test for directly rather than assume from a vendor's human-in-the-loop marketing claims [6].

References

[1] Wiz Research. "GhostApproval: A Trust Boundary Gap in AI Coding Assistants." Wiz Blog, July 8, 2026.

[2] The Hacker News. "GhostApproval Symlink Flaws Could Let Malicious Repos Run Code in AI Coding Agents." The Hacker News, July 2026.

[3] The Register. "Bug in Top AI Coding Agents Shows That Unix-Era Security Headaches Never Really Die." The Register, July 8, 2026.

[4] Infosecurity Magazine. "GhostApproval Flaw Hits Six Major AI Coding Assistants." Infosecurity Magazine, July 2026.

[5] CSAI Foundation / Josh Woodruff, MassiveScale.AI. "Agentic Trust Framework (ATF) v0.9.1." CSAI Foundation, April 2026.

[6] Cloud Security Alliance. "AI Controls Matrix (AICM) v1.1." Cloud Security Alliance, 2026.

[7] Cloud Security Alliance. "Autonomous but Not Controlled: AI Agent Incidents Now Common in Enterprises." Cloud Security Alliance, 2026.

[8] Cloud Security Alliance. "Securing Autonomous AI Agents." Cloud Security Alliance, 2026.

[9] Cloud Security Alliance. "Identity and Access Gaps in the Age of Autonomous AI." Cloud Security Alliance, 2026.

