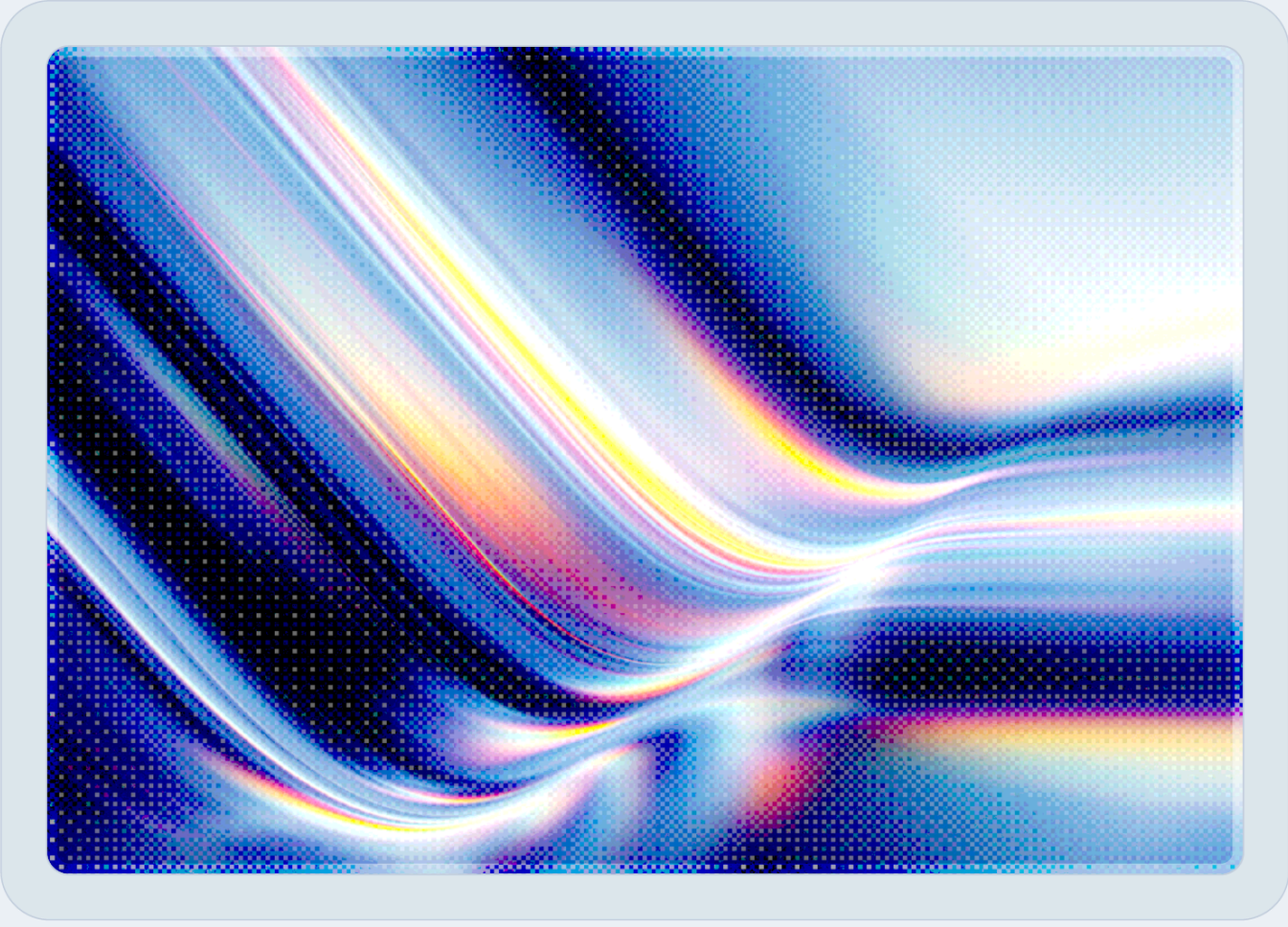
CSAI Foundation | Cloud Security Alliance

From Inference Vendor to IDE Backdoor: The Fireworks–Cursor Chain

How a Subprocessor Key and Unredacted Agent Traces Led to a Trojanized Claude Code Extension

2026-11-16

AI-assisted Rapid Research



CSAI

CSA cloud security alliance®

© 2026 Cloud Security Alliance. Some rights reserved.

You may download, store, display, view, print, redistribute, and link to this document in its original, unmodified form, provided that attribution to the Cloud Security Alliance is maintained and all trademark and copyright notices remain intact.

This document may not be modified or altered. You may quote portions of the document as permitted by the Fair Use provisions of the United States Copyright Act, provided that attribution is given to the Cloud Security Alliance.

This document may be shared on professional and social media platforms in its original form with attribution.

This document was generated with AI assistance and has not undergone official CSA review and approval processes.

Key Takeaways

On November 13, 2026, Cursor published the root cause of the October 29 incident in which its extension marketplace served trojanized builds of Anthropic's Claude Code extension [1]. The chain did not begin in Cursor's build pipeline or at Anthropic. It began at Fireworks AI, Cursor's inference and training provider. An attacker obtained an access key that Cursor had issued to Fireworks, and Cursor states that Fireworks exposed it. The attacker used the key to copy a read-only storage bucket of coding-agent traces that Cursor shared with Fireworks [1]. Some of those traces still contained users' credentials. Those credentials led into SaaS tools Cursor uses, and Cursor says a credential found there was used to place the malicious extension builds in the cache that serves Cursor's marketplace [1][5].

Four points matter most for security leaders. First, agent session traces can function as an unmanaged credential store. Coding agents record what they read and run, and that record often captures secrets, so any party holding unredacted traces may hold those secrets as well. Second, the compromise moved across four trust boundaries: an inference subprocessor, a vendor's shared dataset, the vendor's SaaS estate and finally a software distribution channel that reaches developer endpoints. Third, the malicious build was a modified downstream copy of a trusted publisher's package. Anthropic's own releases were not altered, and the Visual Studio Marketplace, Open VSX and npm packages were not affected [1][8]. Fourth, organizations whose developers installed Anthropic.claude-code 2.1.312 or 2.1.313 from Cursor's marketplace during the October 29 window should treat those hosts as potentially backdoored. Cursor warns that "uninstalling the extension alone may not be enough" [1][2].

Background

Cursor is an AI-native code editor built on the VS Code codebase. It runs its own extension marketplace, which serves extensions sourced from Open VSX through a caching layer [1][8]. Like many AI application vendors, Cursor sends inference and some training workloads to specialized providers. Fireworks AI is one of them: a platform for serving and fine-tuning open-weight models. To support training, Cursor shared selected data with Fireworks through a read-only storage bucket. That data included coding-agent traces from sessions where users had not enabled Privacy Mode, mostly from 2025. Cursor states that those sessions were permitted for model training [1][5].

Fireworks disclosed two security incidents in October 2026. Its first bulletin reports that on October 6 it identified "unauthorized use of internal Fireworks credentials." The intruder reached environment variables, including API keys and tokens stored as secrets, in evaluation jobs belonging to a limited number of customer accounts [3]. A November 9 update widened the scope. A third-party cloud provider hosting Fireworks research and development machines had a security incident on October 23, and on October 24 the intruder used that access to reach Fireworks development machines. The update also reports impact to account and billing data and, for a limited set of accounts, to secrets, datasets and incomplete model weights [4]. Fireworks attributes the activity to a financially motivated group aimed at selling tokens from unauthorized access to large language models to illicit routing gateways, a practice known as LLMJacking [4].

Cursor was not the only downstream victim. Uber disclosed a breach through an unnamed service provider, which TechCrunch reported to be Fireworks [5]. Notion has disclosed that some workspace data was affected by the Fireworks incident. It states that Fireworks had provided a contractual commitment to zero data retention for Notion's use of its platform, yet data was retained from the Fireworks endpoints Notion used [13]. The Fireworks incident also sits inside a wider wave. Wiz counted at least nine AI developer and hosting platforms that disclosed incidents in October 2026, from Fireworks on October 6 to Northflank on October 30. It reported that attackers were "explicitly hunting stored environment variables, with a focus on those granting AI access" [7]. In two enterprise cases Wiz investigated, the first unauthorized model call came within 30 minutes of the reported credential access at the vendor [7]. CSA has described this pattern as a campaign against inference providers that runs through the intermediary platforms holding their keys [10].

Security Analysis

Reconstructing the Chain

Cursor's account, read together with the Fireworks bulletins and Cursor's October 29 security notice, supports a four-stage reconstruction. Public sources leave some links open, and those gaps are noted in the table and the discussion that follows. Cursor has not said which of the two Fireworks incidents exposed its key, or when the key was taken. Fireworks' own bulletins do not mention Cursor or any partner-issued key [3][4]. Cursor has also not said how long the attacker held the bucket data before the marketplace was tampered with. Defenders should therefore treat the timeline between initial exposure and October 29 as unknown, not short.

Stage	Trust boundary crossed	What the attacker obtained	Source status
1. Inference subprocessor	Fireworks environment	An access key Cursor had issued to Fireworks	Reported by Cursor [1]; Fireworks does not name the key [3] [4]
2. Shared training dataset	Cursor's read-only bucket	Coding-agent traces (mostly 2025, non-Privacy Mode), some with unredacted credentials	Reported by Cursor [1]
3. Vendor SaaS estate	Cursor's SaaS tools and an internal system	Archived non-production repositories, support emails, usage metrics, agent conversation summaries, credentials users sent to support	Reported by Cursor [1][5][6]
4. Distribution channel	Cursor's marketplace cache	Ability to serve tampered Anthropic.claude-code 2.1.312 and 2.1.313 builds	Reported by Cursor [1][2]; the exact credential and system used are not disclosed

The scale figures come from Cursor, and the two percentages use Cursor's own, differing denominators. Coding-agent traces were exposed for fewer than 0.05% of users who were not in Privacy Mode, and API credentials were exposed for less than 0.01% of customers [1][6]. Cursor estimates that at most 5,000 users requested the affected cached versions during the window and may have received the malicious build [1]. Cursor reports no evidence of impact on customer code repositories, its production environment, official application releases, other marketplace extensions, or users with Privacy Mode enabled [1][5]. The last attacker activity Cursor observed was on November 1, and it was blocked [1].

Agent Traces as an Unmanaged Credential Store

The control at the center of this incident is not the extension marketplace. It is the handling of agent session data before it is shared. A coding-agent trace records the prompts a developer typed, the files the agent read, the commands it ran and the output it saw. In ordinary work that can include secrets pasted into a prompt, values read from `.env` files, tokens printed by a CLI and connection strings in

configuration. When such traces are kept for model training and shared with a subprocessor, the secrets go with them. Cursor reports that some of the shared traces contained unredacted credentials, which indicates that secret removal was not consistently applied to this dataset before it was shared [1]. Its remediation list now includes added secret scanning across its code repositories and expanded scanning of other internal data sources [1].

For enterprise customers, exposure depended in part on how Privacy Mode was configured. Privacy Mode opts users out of training use [5]. Cursor's current documentation states that it is enabled by default for team members and that administrators can enforce it organization-wide so members cannot disable it [14][15]. CSA could not confirm whether the same defaults and enforcement controls applied throughout 2025, the period most of the exposed traces come from. In organizations that had not enforced Privacy Mode centrally, exposure depended on individual developer settings that a security team may not have tracked. The lesson reaches beyond Cursor. Any AI tool that keeps agent sessions for product improvement or training is holding a secondary copy of the secrets those sessions touched. Organizations should ask each AI tool vendor whether training opt-out can be enforced at the organization level, confirm that such enforcement is turned on, and ask whether secrets are removed before retention or sharing.

Fourth-Party Exposure Without a Contract

Many, likely most, Cursor enterprise customers had no direct relationship with Fireworks. In third-party risk terms, Fireworks was a fourth party, and the cloud provider whose incident gave access to Fireworks development machines was effectively a fifth [4][9]. In CSA's experience, many vendor assessments ask whether a supplier encrypts data and holds certifications, but do not ask which inference and training subprocessors receive customer-derived data, under which data-processing path, and with what credentials. The Notion disclosure suggests that a zero-data-retention commitment may not, in practice, prevent customer data from persisting in training or analytics datasets. Notion reports that data was retained despite Fireworks' contractual commitment, and whether the cause lay in how the commitment was scoped or how it was implemented has not been publicly settled [9][13].

The Cursor case adds a second issue. The key Cursor issued to its subprocessor was a credential with standing access to a shared bucket, and once that key leaked from the subprocessor, it worked for whoever held it. Cursor's remediation includes expanding network access restrictions across its service providers, where possible, so that compromised credentials cannot be used outside their intended location [1]. This suggests network restrictions on service-provider credentials were absent or incomplete for this key at the time.

Downstream Distribution of a Trusted Publisher's Package

The final stage turned a data breach into endpoint compromise. Cursor's notice describes the tampered builds as legitimate extension bundles with a JavaScript loader that ran when the extension loaded [2][8]. The loader collected host and user details, including hostname, username, Git email, public IP and the names of environment variables, and sent them to an HTTP beacon endpoint on port 8099 at 134[.]209[.]82[.]225. It then appended an attacker SSH key labelled `svc-telemetry` to `authorized_keys`, tried to enable the local SSH service and tried to open a reverse SSH tunnel on port 22 to the same address. For persistence it used a LaunchAgent named `com.apple.sysmond` on macOS and a scheduled task named `SysMon` with a Startup-folder fallback on Windows, retrying the tunnel about every 60 seconds [2].

Cursor states that the SSH enablement and tunnel step usually failed, citing errors in the malicious code and the need for administrator rights. It also states that the extension did add the attacker's SSH key, install persistence and send host details to the attacker [2]. BleepingComputer reports that researchers found coding errors that probably stopped the backdoor from working on many systems where SSH was not already enabled [2][6]. Failed tunnel setup therefore does not mean a host is clean. The loader wrote artifacts outside the extension directory, and environment-variable names give an attacker a map of which secrets exist on a developer workstation [2][8].

Cursor also states that exposure depends on where and when the extension was downloaded, not on the version number alone, because the same version on Open VSX was clean [1][8]. This is the structural point for IDE forks. A marketplace that mirrors or caches another registry's packages becomes its own distribution authority. It serves the original publisher's name and reputation, but its integrity depends on the mirror operator's credentials and controls. Unless the client checks the publisher's own signature or a hash from the origin registry, users cannot tell a faithful mirror from a tampered one.

What This Chain Suggests About AI Vendor Risk

Taken together, the stages point to a risk pattern that CSA expects to recur. AI application vendors combine three assets that are each attractive on their own: credentials to inference providers, large volumes of developer interaction data, and distribution channels that run code on customer endpoints. A compromise of one asset can lead to the others when they are linked by shared credentials or by data that contains credentials. The October wave of attacks on AI developer platforms shows that attackers are actively hunting the first asset [7][10]. The Cursor chain shows that at least one attacker was able to move from the first asset into the second and third when the path was open, and CSA expects others to

attempt the same. Defenders should not assume that the attacker in this case set out to reach developer endpoints from the start, because the public record does not settle intent. What it does show is that the path existed, and that it ran through controls that most customers could not see.

Recommendations

Immediate Actions

Organizations should first determine whether any developer installed or updated Anthropic.claude-code to 2.1.312 or 2.1.313 from Cursor's marketplace between approximately 2:27 a.m. and 1:35 p.m. PT on October 29, 2026 [2]. Version numbers alone are not conclusive, so extension-install telemetry and the package hashes in Cursor's notice should be used to identify candidate hosts. The response on any candidate host then proceeds through detection, containment and credential rotation. The table below summarizes those steps, drawn from Cursor's guidance [1][2] and, for Fireworks customers, from Fireworks' bulletins [3][4].

Response step	Action
Detect	Search `authorized_keys` (and `administrators_authorized_keys` on Windows) for the `svc-telemetry` key, and search endpoint and network telemetry for the published indicators, including connections to 134[.]209[.]82[.]225 on ports 22 and 8099.
Contain and clean	Isolate any host showing payload execution, then remove the key, the LaunchAgent, scheduled task, cron or Startup entries, and any tunnel processes, or rebuild the host.
Rotate host-reachable credentials	From a trusted system, rotate every credential and session that was reachable from an affected host, prioritizing secrets whose environment-variable names may have been collected.
Rotate trace-exposed credentials	Rotate any credentials named in a Cursor notification. Separately, rotate secrets that developers may have exposed in Cursor agent sessions without Privacy Mode, even where no notification was received, if the organization cannot rule out such exposure.

Response step	Action
Fireworks customers	Rotate third-party credentials at the issuing provider, because deleting a stored secret does not invalidate it, and review accounts for unrecognized keys, evaluators and datasets [3][4].

The broader rotation of trace-exposed secrets can be expensive, so organizations should sequence it by impact, starting with long-lived cloud and AI provider keys before moving to shorter-lived or narrowly scoped credentials.

Short-Term Mitigations

Over the next quarter, security teams should list every AI coding tool in use and ask each vendor four questions. Are agent session traces retained? Are they shared with inference or training subprocessors? Can training opt-out or zero retention be enforced at the organization level? Are secrets removed from traces before retention or sharing? Where organization-level enforcement exists, it should be turned on and checked. Developer guidance should also discourage pasting live credentials into agent sessions. Short-lived, scoped tokens limit the damage when traces are exposed, and existing secret scanners can be pointed at agent configuration and session artifacts as well as repositories.

For IDE extensions, organizations should control the source as well as the package name. Where possible, pin approved extensions to a specific registry and version, allow-list by publisher and hash, and route installs through an internal mirror whose integrity the organization checks. Endpoint detection rules should cover new `authorized_keys` entries, SSH services enabled on developer laptops, reverse SSH tunnels, and new LaunchAgents or scheduled tasks created by editor processes. These behaviors are uncommon in most developer workflows and appeared in this incident [2], although rules should be tuned to account for legitimate remote-development setups.

Strategic Considerations

Third-party risk programs for AI vendors should include subprocessors as a standing part of the review. That means asking for the list of inference and training providers, which data each receives, and how credentials issued to them are scoped, bound to network locations and rotated. Contracts should require prompt notice when a subprocessor reports a security incident. They should also make clear whether zero-retention commitments cover training and analytics paths as well as live inference, and they should give customers verification or audit rights so that a retention commitment can be checked rather than

assumed. Whether Cursor's key was exposed in the incident Fireworks identified on October 6 (first bulletin October 9) or in the October 23–24 intrusion, Cursor customers had no public notice of the link before the October 29 compromise, and the full timeline cannot be explained from public sources [1][3][4]. That gap shows why customers need to know when their vendor's suppliers are breached.

IDE fork vendors and other marketplace operators should treat mirrored packages as code they distribute and secure the path accordingly. That includes separating cache write credentials from general SaaS access, checking publisher signatures or origin hashes when packages are served, and monitoring for content that differs from the upstream registry. Publishers of widely used extensions, in turn, may need to publish verifiable signatures that downstream marketplaces and clients can check on their own.

CSA Resource Alignment

This note builds on CSA's earlier coverage of this incident and of the attack wave behind it. *Trojanized Claude Code Extension on Cursor's Marketplace* [8] documented the October 29 payload, its indicators and endpoint response steps, and noted that no public account then explained how the attacker gained publishing access. The present note answers that question from Cursor's November 13 disclosure. It shifts the focus from endpoint cleanup to the upstream data and credential controls that made the compromise possible. CSA's *AI Toolchain Hijacked: IDE Plugin API Key Theft* [16] examined IDE plugin ecosystems as a channel for AI credential theft, and the Cursor marketplace stage is a further instance of that channel being used against developers.

Your Vendor's Inference Provider: Fourth-Party AI Subprocessor Risk [9] analyzed the Notion side of the Fireworks breach. It argued that inference and training subprocessors belong in third-party risk scope, and that zero-retention commitments may not cover every processing path. The Cursor chain extends that analysis. The fourth-party breach led to data exposure and then to endpoint compromise at customers through the vendor's own distribution channel. *Inference Providers as the Prize: Platform-Chain Campaigns Against AI Serving* [10] described the October pattern of intrusions aimed at AI provider credentials held by intermediary platforms. Fireworks is one of the incidents it examined, and Cursor is now a confirmed downstream path. CSA's *Approved Once, Leaked Forever: Secrets in Claude Code Settings Files* [11] covers a related problem: credentials that persist in AI coding-tool artifacts long after the developer stopped thinking about them. Agent traces held for training are one more artifact of that kind.

At the control level, CSA's AI Controls Matrix (AICM) v1.1 [12] provides the mapping for the recommendations above. Its Supply Chain Management, Transparency and Accountability domain covers subprocessor inventory and incident notification. Its Data Security and Privacy Lifecycle Management domain covers secret removal from training data and retention scope, and its Identity and Access

Management domain covers scoping and network binding of credentials issued to service providers. Organizations assessing AI coding-tool vendors can use these AICM controls to structure the questions in the Short-Term Mitigations section.

References

[1] Cursor. "An update on our recent security incident." Cursor Blog, November 13, 2026.

[2] Cursor. "Security notice — malicious version of Anthropic.claude-code (2.1.312 and 2.1.313) identified in the Cursor VS Code Extension Marketplace." Cursor Blog, October 29, 2026 (updated October 30, 2026).

[3] Fireworks AI. "Fireworks October 2026 security incident." Fireworks AI Security Bulletins, October 9, 2026 (updated October 10, 2026).

[4] Fireworks AI. "Update on October 2026 security incident." Fireworks AI Security Bulletins, November 9, 2026.

[5] Lakshmanan, Ravie. "Cursor Says Stolen Fireworks AI Key Led to Trojanized Claude Code Extension Incident." The Hacker News, November 15, 2026.

[6] Abrams, Lawrence. "Cursor confirms security breach led to malicious Claude Code extension." BleepingComputer, November 14, 2026.

[7] Wiz Research. "\"Black October\" for AI Developer Platforms." Wiz Blog, October 30, 2026 (updated through November 10, 2026).

[8] Cloud Security Alliance. "Trojanized Claude Code Extension on Cursor's Marketplace." CSA Labs, November 3, 2026.

[9] Cloud Security Alliance. "Your Vendor's Inference Provider: Fourth-Party AI Subprocessor Risk." CSA Labs, November 12, 2026.

[10] Cloud Security Alliance. "Inference Providers as the Prize: Platform-Chain Campaigns Against AI Serving." CSA Labs, October 23, 2026.

[11] Cloud Security Alliance. "Approved Once, Leaked Forever: Secrets in Claude Code Settings Files." CSA Labs, November 8, 2026.

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

[13] Notion. "Notice about the Fireworks AI incident that may impact some Notion workspaces." Notion Help Center, November 2026.

© 2026 Cloud Security Alliance. AI-assisted Rapid Research.

[14] Cursor. "Privacy and data." Cursor Docs, accessed November 16, 2026.

[15] Cursor. "Privacy and Data Governance." Cursor Docs, accessed November 16, 2026.

[16] Cloud Security Alliance. "AI Toolchain Hijacked: IDE Plugin API Key Theft." CSA Labs, June 2026.