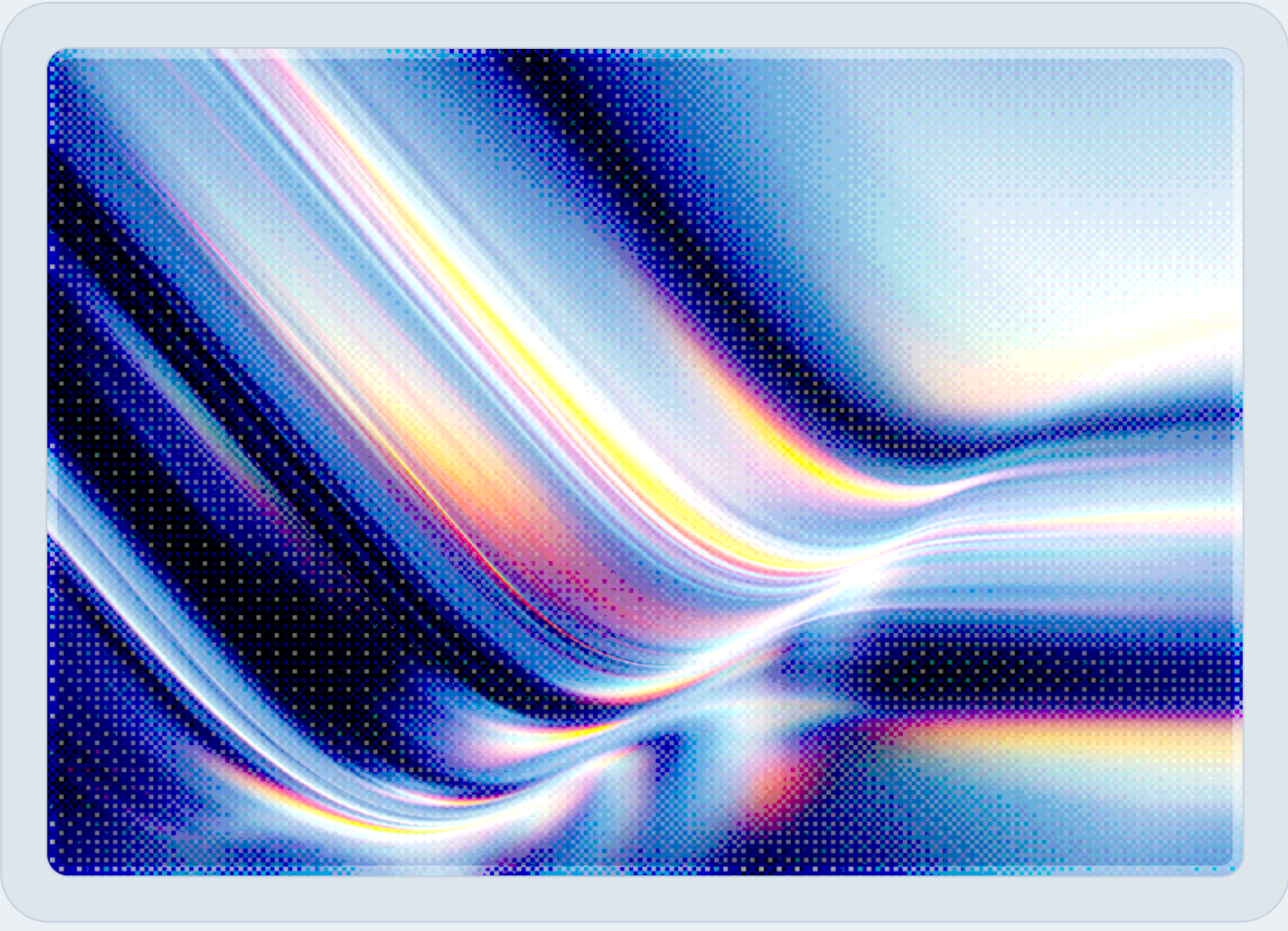
CSAI Foundation | Cloud Security Alliance

JetBrains Marketplace: AI Plugin Campaign Steals LLM API Keys

Fifteen Trojanized AI Coding Assistants Exfiltrate Developer Credentials to Attacker Infrastructure

2026-06-19

AI-assisted Rapid Research



CSAI

CSA cloud security alliance®

Key Takeaways

- Fifteen malicious plugins published across seven JetBrains Marketplace publisher accounts accumulated approximately 70,000 combined installations between October 2025 and June 2026, operating for eight months before discovery [1].
- Every plugin posed as a legitimate AI coding assistant built atop popular large language models, but silently exfiltrated any AI provider API key entered by the developer the moment the developer clicked "Apply" in plugin settings [2].
- The attack exploits a fundamental architectural gap: JetBrains plugins run fully unsandboxed with the same operating system privileges as the IDE itself, with no fine-grained permission model to restrict outbound network connections or credential access [3].
- JetBrains removed all 15 plugins and terminated the seven associated publisher accounts on June 16, 2026, and activated a remote kill-switch to disable surviving installations upon IDE restart; however, any API key entered into these plugins while they were active should be considered compromised and must be rotated immediately [1].
- This campaign is part of a documented pattern of threat actors targeting developer tooling ecosystems as a high-value supply chain attack surface, following similar incidents targeting PyPI, npm, and VS Code extension marketplaces [4][5][6].

Background

The JetBrains family of integrated development environments — IntelliJ IDEA, PyCharm, WebStorm, GoLand, and related products — serves tens of millions of software developers worldwide according to JetBrains' reported figures. These environments depend on a plugin ecosystem distributed through JetBrains Marketplace, which hosts tens of thousands of third-party extensions [3]. Plugins can add language support, integrate external services, and, increasingly, embed AI-powered coding assistants that communicate directly with LLM providers such as OpenAI, DeepSeek, and Anthropic. Developers adopting these AI tools commonly paste provider API keys into plugin configuration dialogs so the extension can make API calls on their behalf.

This configuration pattern creates a concentrated credential surface attractive to threat actors, because AI provider API keys are billable credentials: unauthorized use translates directly to financial loss, and harvested keys can be resold or consumed for compute-intensive tasks. The same keys may be bound to organizational accounts with access to proprietary codebases, making their theft a potential precursor to intellectual property exfiltration or further lateral movement into cloud environments.

Beginning in late October 2025, security researchers at Aikido Security identified a coordinated campaign in which threat actors published 15 JetBrains Marketplace plugins under seven separate vendor accounts [2]. The plugins were all styled as AI-powered developer utilities — covering code chat, unit test generation, Git commit message drafting, code review, and static bug finding — and every one of them functioned as advertised. The functional authenticity appears deliberate, as a plugin that immediately breaks attracts user complaints and rapid takedown; one that works correctly while quietly harvesting credentials can persist undetected.

The campaign reached developers using JetBrains IDEs across all major platforms and accumulated roughly 70,000 installations before Aikido Security researcher Ilyas Makari disclosed the findings to JetBrains on June 16, 2026 [2][1]. The two most-downloaded plugins — DeepSeek AI Assist (27,727 downloads) and CodeGPT AI Assistant (25,571 downloads) — accounted for the majority of the exposed install base [2][7].

Security Analysis

Attack Architecture

The malicious code was embedded in each plugin's settings persistence handler. When a developer opened the plugin configuration panel, entered an API key for OpenAI, DeepSeek, or SiliconFlow, and clicked "Apply," the plugin's `save()` method executed two actions in sequence: it stored the key in the IDE's local settings store as expected, and it simultaneously dispatched the key to an attacker-controlled server [2][3]. This dual-action design ensured that the plugin appeared to work correctly while the exfiltration occurred invisibly in the background.

The exfiltration logic included targeted validation: the code specifically identified keys beginning with the prefix `sk-` and exactly 51 characters in length — a pattern matching the standard format for OpenAI API keys — before forwarding them [2]. The connection was made via unencrypted HTTP POST to a hardcoded command-and-control endpoint at `39.107.60[.]51/api/software/key`, authenticating with a static token embedded in the plugin JAR file (`F48D2AA7CF341F782C1D`) [2]

[1]. To prevent the IDE's security infrastructure from flagging the unencrypted connection, the plugins implemented a custom `X509TrustManager` that disabled TLS certificate validation, effectively suppressing any runtime warnings about the plaintext transmission [1].

The choice to use plain HTTP rather than HTTPS is analytically significant. It suggests either operational speed over security on the attacker's part, or a deliberate decision to avoid the certificate infrastructure that would be required to establish a plausible encrypted endpoint. Either way, the absence of TLS is a detectable signal — one that JetBrains has indicated it will incorporate into future automated screening [1].

Monetization Model

Aikido Security's analysis identified a two-tier monetization structure in the attacker's backend infrastructure [2]. The server at the command-and-control IP not only received harvested keys from free plugin users, but also appeared capable of distributing API keys to a second cohort — users who had paid the plugin operators for access. The operational implication is that threat actors appear to have been running a credential laundering service — the server architecture suggests stolen keys from free-tier users could have been resold as "access" to paying customers, who would have had no knowledge that their provided credentials were stolen property. This business model creates a financial incentive to maximize the install base of free-tier plugins, explaining the volume of publishing activity across seven separate accounts over eight months.

This monetization pattern also has secondary security implications for organizations. A stolen API key consumed by a third party generates usage that appears on the legitimate key owner's billing account and in their provider audit logs. Without active monitoring of API usage anomalies, affected developers might not detect the compromise until an unexpected bill arrives or a provider flags the activity.

Marketplace Security Gaps

The JetBrains Marketplace employs a combination of automated upload checks and human review prior to plugin approval [3]. Despite these controls, the campaign persisted undetected for approximately eight months, encompassing multiple new plugin publications across the period [8]. Several structural factors explain this persistence.

First, the plugins were functionally legitimate. Manual review of plugin capabilities would not surface the credential-theft behavior, because the AI coding features worked as described. The malicious code path activated only in response to a specific user action — entering an API key and saving settings — which is not a behavior that static or behavioral pre-publication analysis would necessarily simulate. Second,

JetBrains' plugin architecture does not enforce a sandboxed permission model [3]. Unlike browser extensions, which must declare network access permissions that appear in installation prompts, JetBrains plugins inherit full IDE-level access to the network, the filesystem, and the operating system. There is no manifest of declared capabilities for users to review. Third, the use of multiple publisher accounts allowed the campaign to distribute risk: even if one account had been flagged, the remaining accounts and their associated plugins would continue operating.

Relationship to the AI Developer Tooling Threat Surface

This campaign is not an isolated event. It follows a pattern that CSA has documented across multiple supply chain attack investigations in 2025 and 2026. The LiteLLM PyPI backdoor incident in March 2026 demonstrated that threat actors actively target the toolchains developers use to integrate with LLM APIs [4]. The TeamPCP campaign compromised security tooling used in AI/ML build pipelines, exploiting the trusted position those tools occupy in CI/CD workflows [5]. The Mini Shai-Hulud campaign targeted multiple package ecosystems simultaneously, exploiting developer trust in official registries [6].

What distinguishes the JetBrains campaign is its targeting of the IDE layer — the single development tool that sits at the center of a professional developer's day. IDEs hold or have access to source code, cloud credentials, SSH keys, signing certificates, Git authentication tokens, and now LLM API keys. A plugin operating inside an IDE operates inside one of the most trusted processes on a developer's workstation. This positioning is precisely why plugin marketplaces are attractive to threat actors: a successful implant is invisible, persistent across IDE restarts, and has broad access without requiring any escalation of privilege.

The prevalence of DeepSeek branding across most of the malicious plugins suggests a deliberate attempt to exploit the rapid developer adoption of DeepSeek's models — plugins impersonating a widely-discussed service are presumably installed with less scrutiny, particularly when developers are actively seeking integrations that were recently announced or heavily discussed in technical communities.

Recommendations

Immediate Actions

Organizations and individual developers who installed any of the following plugins should treat their AI provider API keys as compromised and rotate them immediately: DeepSeek Junit Test, DeepSeek Git Commit, DeepSeek FindBugs, DeepSeek AI Chat, DeepSeek Dev AI, DeepSeek AI Coding, AI FindBugs, AI Git Commitor, AI Coder Review, DeepSeek Coder AI, AI Coder Assistant, DeepSeek Code Review, CodeGPT AI Assistant, DeepSeek AI Assist, and Coding Simple Tool [1][2][9]. Key rotation should be performed through the provider's developer console — OpenAI's API key management page, DeepSeek's platform settings, or SiliconFlow's console as appropriate — and any old keys should be permanently revoked, not merely disabled.

Following key rotation, developers should review their AI provider usage logs for the period spanning October 2025 through June 2026. Unusual API call volumes, unexpected geographic origins, or usage during hours when the developer was not working are indicators of active key abuse. Organizations with centrally managed API keys should also review any shared key usage that might have been touched by affected plugin installations.

Network defenders should block outbound connections to `39.107.60[.]51` at the perimeter firewall and endpoint security tooling. While JetBrains has disabled surviving plugin installations via its remote kill-switch, blocking the known C2 IP provides defense-in-depth against any installations that have not yet restarted.

Short-Term Mitigations

Developer organizations should establish governance practices for IDE plugin installation analogous to those maintained for package dependencies. Plugins should be sourced from verified publishers, and any plugin requesting network access or handling credentials deserves additional scrutiny before approval. Where JetBrains IDEs are deployed on managed endpoints, security teams should consider whether a plugin allowlist policy is appropriate, particularly in environments where developers handle sensitive credentials or proprietary source code.

API key management should be treated with the same rigor as other secrets management. AI provider API keys should not be entered directly into plugin configuration dialogs without first verifying that the plugin's network behavior is consistent with its stated purpose. Where possible, organizations should use

short-lived, scoped API keys with spending limits, so that unauthorized use is bounded in impact even if a key is compromised. Monitoring API key usage through provider dashboards or SIEM integrations can provide early warning of unauthorized consumption.

Plugin integrity verification is a gap that JetBrains is working to address, but developers should not wait for platform-level controls. Reviewing the changelog, publisher history, and community feedback for any AI-related plugin before installation is a practical first line of defense. Plugins with very recent publication dates, no meaningful user reviews, and publisher accounts with no other published work are risk indicators worth treating with suspicion.

Strategic Considerations

This incident illustrates that the attack surface for AI credential theft has expanded substantially as developer adoption of LLM APIs has accelerated. Security programs that track secrets management for cloud credentials, database connection strings, and SSH keys should explicitly extend that scope to include AI provider API keys. These keys are billable credentials with direct financial consequences from unauthorized use, and they may provide access to organizational AI deployments with sensitive data exposure implications.

IDE plugin ecosystems across the industry — JetBrains Marketplace, the VS Code Extension Marketplace, and others — are generally understood to operate with less formalized security vetting than major package registries, despite the greater privilege level that IDE extensions typically possess [3]. Industry-wide improvements to plugin security will require cooperation between IDE vendors, security researchers, and the broader developer community. CSA supports efforts to establish baseline security standards for plugin and extension marketplaces, including mandatory permission declarations, code signing with verified publisher identities, and automated behavioral analysis for credential-handling code paths.

CSA Resource Alignment

This incident maps directly to the threat categories analyzed in CSA's MAESTRO framework for agentic AI threat modeling, particularly its treatment of supply chain compromise as a mechanism for introducing malicious code into trusted AI-integrated development environments. The attacker's exploitation of the trusted developer tooling layer is consistent with MAESTRO's analysis of how adversaries target the agentic pipeline at its most trusted points.

The CSA AI Controls Matrix (AICM) addresses third-party integration risk under its AI Supply Chain and Dependency controls, providing organizations with a structured framework for evaluating the security posture of AI-adjacent tooling. The principle that any component handling LLM credentials should be subject to the same vetting as infrastructure handling cloud access keys is embedded in AICM's guidance on AI workload isolation and secrets management.

CSA's prior research on developer supply chain attacks provides directly relevant precedent. The LiteLLM PyPI backdoor investigation documented how AI toolchain components become high-value targets once they sit in the credential flow between developers and LLM providers [4]. The TeamPCP and Shai-Hulud campaign analyses established that organized threat actors are conducting multi-vector, multi-ecosystem supply chain campaigns with AI infrastructure as a primary target [5][6]. The JetBrains campaign should be understood as the IDE-layer expression of the same adversarial logic: compromise the trusted tool, harvest the credential, monetize the access.

CSA's Zero Trust guidance is also applicable here. A zero-trust posture applied to IDE plugin behavior would treat outbound network connections from plugin processes as untrusted by default, requiring explicit policy authorization for each plugin to reach named destinations. Until IDE vendors implement this at the platform level, network-layer controls at the organization perimeter provide a partial mitigation.

References

[1] JetBrains. "JetBrains Marketplace Ecosystem Security Update: Addressing Malicious Third-Party AI Plugins." JetBrains Blog, June 2026.

[2] Makari, Ilyas. "Multiple JetBrains IDE Plugins Caught Stealing AI Keys." Aikido Security, June 2026.

[3] JetBrains. "Understanding Plugin Security." JetBrains Marketplace Documentation, 2026.

[4] Cloud Security Alliance AI Safety Initiative. "LiteLLM PyPI Backdoor: Credential Theft in AI Toolchains." CSA Labs, March 2026.

[5] Cloud Security Alliance AI Safety Initiative. "TeamPCP: Cascading Supply Chain Attack on AI/ML Tooling." CSA Labs, March 2026.

[6] Cloud Security Alliance AI Safety Initiative. "Mini Shai-Hulud: Multi-Ecosystem Developer Supply Chain Attack." CSA Labs, 2026.

[7] Abrams, Lawrence. "Malicious JetBrains Marketplace Plugins Steal AI API Keys from Developers." BleepingComputer, June 2026.

[8] The Hacker News. "Malicious JetBrains Plugins Steal AI API Keys as Chrome Extensions Capture Chatbot Chats." The Hacker News, June 2026.

[9] Infosecurity Magazine. "Fifteen JetBrains Marketplace Plugins Steal API Keys." Infosecurity Magazine, June 2026.