INTELLIGENCE BRIEFING
Security Command Center
TLP:CLEAR
2026-06-17 07:15 UTC

Coordinated JetBrains Marketplace Plugin Campaign Steals AI API Keys from ~70,000 Developer Installs

THREAT CAMPAIGN | **HIGH** | CVSS 7.5

SCC Item ID	SCC-CAM-2026-0494
Type	Threat Campaign
Severity	HIGH
CVSS Base Score	7.5
Affected Products	JetBrains IntelliJ-based IDEs (all versions supporting Marketplace plugins); JetBrains Marketplace; AI provider credentials targeted: OpenAI, DeepSeek, SiliconFlow
Published	2026-06-16T17:54:50
Discovery Source	Rss

Executive Summary

A coordinated supply chain campaign planted at least 15 malicious plugins in the JetBrains Marketplace over eight months, accumulating nearly 70,000 installs before discovery by Aikido Security. The plugins silently harvested AI provider API keys for OpenAI, DeepSeek, and SiliconFlow and transmitted them in plaintext to attacker-controlled servers. Organizations with developers using JetBrains IntelliJ-based IDEs face immediate risk of AI service account takeover, unauthorized API cost charges, and potential exposure of proprietary code or data submitted through compromised AI integrations.

Technical Analysis

Fifteen or more malicious plugins were published across seven vendor accounts on the JetBrains Marketplace, collectively reaching approximately 70,000 installs. Each plugin targeted credentials stored in IDE configuration for OpenAI, DeepSeek, and SiliconFlow, exfiltrating them via plaintext HTTP POST to hardcoded attacker-controlled endpoints (CWE-319: Cleartext Transmission of Sensitive Information). The plugins exploited the trusted Marketplace distribution channel to bypass developer skepticism (CWE-494: Download of Code Without Integrity Check; CWE-522: Insufficiently Protected Credentials). MITRE ATT&CK techniques include T1195.001 (Supply Chain Compromise: Compromise Software Dependencies and Development Tools), T1608.001 (Stage Capabilities: Upload Malware), T1555 (Credentials from Password Stores), T1552.001 (Unsecured Credentials: Credentials in Files), T1041 (Exfiltration Over C2 Channel), and T1071.001 (Application Layer Protocol: Web Protocols). All JetBrains IntelliJ-based IDE versions supporting Marketplace

plugins are affected. No CVE has been assigned. At the time of the Aikido Security disclosure, at least one plugin remained available for download and JetBrains had not issued a public statement. Subsequent actions by JetBrains and plugin removal status should be confirmed in current operations. Discovery credit: Aikido Security.

Action Checklist

1. Step 1: Containment. Immediately audit all JetBrains IDE plugin installations across developer workstations and CI/CD environments. Cross-reference installed plugins against the 15 malicious plugins identified in the Aikido Security report (https://www.aikido.dev/blog/multiple-jetbrains-ide-plugins-caught-stealing-ai-keys). Isolate any workstation with a confirmed malicious plugin from network access (or move to an isolated VLAN for forensic analysis) until the plugin is removed and keys are rotated. Block outbound HTTP traffic to attacker-controlled domains and IPs identified in the Aikido report at the perimeter firewall.
2. Step 2: Detection. Prioritize outbound HTTP (port 80, not HTTPS 443) connections from JetBrains IDE processes, as the campaign explicitly used plaintext transmission. Query endpoint logs, DNS/proxy logs, and network firewall logs for these connections to hardcoded C2 endpoints listed in Aikido's IOC disclosures. Review IDE plugin directories (%APPDATA%\JetBrains\, ~/.config/JetBrains/, ~/Library/Application Support/JetBrains/) for plugins not installed through official organizational policy. Correlate with AU-2 (Event Logging) and AU-6 (Audit Record Review) processes. Check AI provider API key usage dashboards for anomalous consumption patterns or unrecognized API calls (per NIST AU-6).
3. Step 3: Eradication. Remove all identified malicious plugins from every affected IDE installation. Rotate all AI provider API keys (OpenAI, DeepSeek, SiliconFlow) immediately for any developer who had a suspicious plugin installed, regardless of whether exfiltration is confirmed. Revoke and reissue keys via each provider's API management console. Enforce an allowlist of approved plugins through organizational IDE configuration management. Map this control to CIS 2.3 (Address Unauthorized Software) for removal of non-approved plugins and CIS 4.6 (Securely Manage Enterprise Assets and Software) for inventory and governance of approved plugins. Remove all plugins not on the approved list.
4. Step 4: Recovery. Validate that all rotated API keys are active and that old keys are fully revoked in each AI provider's dashboard. Monitor AI provider usage logs for 30 days post-rotation for anomalous requests indicating key reuse. Confirm no malicious plugins remain in developer environments via re-audit. Enable NIST AU-12 (Audit Record Generation) on developer endpoints if not already active. Verify that IDE plugin installation is gated through organizational policy controls before restoring full developer access.
5. Step 5: Post-Incident. Conduct a control gap review against CIS 2.1 (Software Inventory) and CIS 2.3 (Address Unauthorized Software) to assess whether an approved plugin inventory existed and was enforced. Implement a formal third-party plugin vetting process before Marketplace installs are permitted. Establish a recurring review cadence for installed IDE plugins per CIS 7.1 (Vulnerability Management Process). Brief developers on supply chain risks associated with IDE plugin ecosystems. Map findings to NIST AC-6 (Least Privilege), evaluate whether developer workstations have excessive access that amplifies the blast radius of credential theft.

IR / Forensic Enrichment

Triage Priority	IMMEDIATE

Escalation Criteria	Escalate to CISO and legal counsel immediately if any AI provider confirms unauthorized API consumption charges exceeding organizational thresholds, if compromised API keys were scoped to organizational accounts holding sensitive training data or proprietary prompts, or if CI/CD pipeline credentials (not just developer workstation keys) are confirmed exposed, as this expands blast radius to production pipeline integrity and may trigger contractual or regulatory breach notification obligations.
Recovery Notes	After all malicious plugins are removed and AI provider API keys are rotated and confirmed revoked at the provider level, monitor OpenAI, DeepSeek, and SiliconFlow usage dashboards daily for a minimum of 30 days for any API calls using the old key identifiers, which would indicate the attacker cached keys before rotation. Re-audit JetBrains plugin directories across all developer workstations and CI/CD build agents at 7-day and 30-day marks using the approved-plugin manifest to detect re-infection or policy drift. Do not restore unrestricted JetBrains Marketplace access until a formal plugin vetting and allowlist enforcement process is operational, as the campaign's 8-month undetected duration indicates the existing Marketplace review controls are insufficient.
Forensic Artifacts	JetBrains IDE plugin JAR files located at %APPDATA%\JetBrains\\plugins\ (Windows), ~/.config/JetBrains//plugins/ (Linux), or ~/Library/Application Support/JetBrains//plugins/ (macOS) — the malicious JARs contain hardcoded C2 URLs and API key harvesting logic that confirm which credentials were targeted and where they were sent. | JetBrains IDE log files (idea.log, idea.log.1) at %APPDATA%\JetBrains\\log\ — these record plugin load events with timestamps, establishing when the malicious plugin was first activated on each developer workstation and bounding the credential exposure window. | Network proxy and DNS logs filtered to outbound port 80 HTTP connections from IDE process names (idea.exe, pycharm64.exe, goland.exe, etc.) to attacker C2 domains — plaintext HTTP was used for exfiltration, so full request bodies containing the harvested API key strings may be recoverable from proxy inspection logs if SSL inspection was not required. | AI provider API key usage logs from the OpenAI platform.openai.com/usage, DeepSeek platform.deepseek.com, and SiliconFlow developer console dashboards — anomalous consumption spikes, calls from unexpected geographic IPs, or API calls to endpoints inconsistent with the developer's known use cases corroborate active key abuse during the campaign window. | Memory dump of the live JetBrains IDE process (acquired via WinPmem or LiME before host isolation) — in-memory strings analysis may recover plaintext API key values that were loaded by the plugin at runtime, confirming exactly which keys were accessible to the malicious plugin at the time of capture.

Per-Action IR Details

Step 1: Containment — Immediately audit all JetBrains IDE plugin installations across developer workstations and CI/CD environments. Cross-reference installed plugins against the 15 malicious plugins identified in the Aikido Security report (https://www.aikido.dev/blog/multiple-jetbrains-ide-plugins-caught-stealing-ai-keys). Disable network access for any workstation where a suspicious plugin is confirmed. Block outbound HTTP traffic to attacker-controlled domains and IPs identified in the Aikido report at the perimeter firewall.

NIST Phase: Containment

Reference: NIST 800-61r3 §3.3 — Containment Strategy: isolate affected assets and block attacker communication channels before eradication to prevent ongoing exfiltration of AI API keys.

Controls: NIST AC-4 (Information Flow Enforcement), CIS 4.4 (Implement and Manage a Firewall on Servers), CIS 4.5 (Implement and Manage a Firewall on End-User Devices)

Compensating: On Windows developer workstations without EDR, run: `Get-ChildItem 'C:\Users\*\AppData\Roaming\JetBrains\' -Recurse -Filter 'plugins' | Get-ChildItem` to enumerate installed plugin directories, then diff against your approved plugin list. Use Windows Firewall (`netsh advfirewall firewall add rule`) or

iptables (Linux/macOS) to block outbound port 80/443 to attacker C2 IPs from the IDE process. On CI/CD servers, apply host-based firewall rules immediately and revoke outbound internet access at the security group or network ACL level.

Evidence: BEFORE isolating any workstation, capture: (1) full memory dump using WinPmem (Windows) or LiME (Linux) to preserve in-memory API key strings and plugin runtime state; (2) active network connections via `Get-NetTCPConnection` (Windows) or `ss -tunap` (Linux/macOS) filtered to idea.exe/pycharm/goland PIDs, capturing any live plaintext HTTP sessions to C2 endpoints; (3) snapshot of the JetBrains plugin directory contents with timestamps (`%APPDATA%\JetBrains\\plugins\` or `~/.config/JetBrains//plugins/`) before any plugin removal; (4) running process list with full command-line arguments to confirm which IDE process loaded the malicious plugin JAR.

Step 2: Detection — Query endpoint logs and DNS/proxy logs for outbound plaintext HTTP connections (port 80) originating from JetBrains IDE processes (idea.exe, idea, pycharm, goland, etc.) to hardcoded C2 endpoints listed in Aikido's IOC disclosures. Review IDE plugin directories (%APPDATA%\JetBrains\, ~/.config/JetBrains/, ~/Library/Application Support/JetBrains/) for plugins not installed through official organizational policy. Correlate with AU-2 (Event Logging) and AU-6 (Audit Record Review) processes. Check AI provider API key usage dashboards for anomalous consumption patterns or unrecognized API calls (per NIST AU-6).

NIST Phase: Detection Analysis

Reference: NIST 800-61r3 §3.2 — Detection and Analysis: correlate network telemetry, endpoint artifacts, and AI provider usage anomalies to determine scope of API key exfiltration across the developer fleet.

Controls: NIST AU-2 (Event Logging), NIST AU-6 (Audit Record Review, Analysis, And Reporting), NIST AU-3 (Content Of Audit Records), CIS 8.2 (Collect Audit Logs)

Compensating: Without a SIEM, use Zeek or Wireshark packet captures on the network egress point filtered to `port 80 and http.request` to identify plaintext POST/GET requests from IDE processes carrying API key strings. On Windows endpoints, deploy Sysmon with EventID 3 (Network Connection) configured to log outbound connections from idea.exe, pycharm64.exe, and goland.exe. Run: `Get-WinEvent -LogName 'Microsoft-Windows-Sysmon/Operational' | Where-Object { $_.Id -eq 3 -and $_.Message -match 'idea|pycharm|goland' }`. On Linux/macOS, use `auditd` or `osquery` (`SELECT * FROM process_open_sockets WHERE pid IN (SELECT pid FROM processes WHERE name LIKE '%idea%' OR name LIKE '%pycharm%')`) to enumerate live socket connections from IDE processes.

Evidence: This step is read-only analysis and does not alter live state; no volatile pre-capture is required before querying logs. However, if live IDE processes are still running on suspected hosts, capture `Get-NetTCPConnection` output and memory before any subsequent containment actions. Key artifacts to review: proxy/DNS logs for queries resolving to C2 domains from Aikido's IOC list; JetBrains IDE log files at `%APPDATA%\JetBrains\\log\idea.log` (Windows) or `~/.config/JetBrains//log/idea.log` (Linux) for plugin load events and network errors; plugin JAR files in the plugin directory for static analysis to identify hardcoded C2 URLs and key-harvesting code; OpenAI, DeepSeek, and SiliconFlow API key usage dashboards for calls originating from unexpected IPs or at unusual hours.

Step 3: Eradication — Remove all identified malicious plugins from every affected IDE installation. Rotate all AI provider API keys (OpenAI, DeepSeek, SiliconFlow) immediately for any developer who had a suspicious plugin installed, regardless of whether exfiltration is confirmed. Revoke and reissue keys via each provider's API management console. Enforce an allowlist of approved plugins via organizational IDE configuration management (CIS 2.3: Address Unauthorized Software; CIS 4.6: Securely Manage Enterprise Assets and Software). Remove all plugins not on the approved list.

NIST Phase: Eradication

Reference: NIST 800-61r3 §3.4 — Eradication: remove malicious plugin artifacts from all affected IDE installations and revoke compromised AI provider credentials to terminate attacker access to OpenAI, DeepSeek, and SiliconFlow accounts.

Controls: NIST AC-2 (Account Management), NIST AC-12 (Session Termination), CIS 2.3 (Address Unauthorized Software), CIS 4.6 (Securely Manage Enterprise Assets and Software), CIS 5.1 (Establish and Maintain an Inventory of

Accounts)

Compensating: Without enterprise software management tooling, script plugin removal via PowerShell: `Remove-Item -Recurse -Force "$env:APPDATA\JetBrains\\plugins\"` across all developer machines using PSRemoting or a simple bash equivalent for Linux/macOS. For API key rotation, invoke each provider's REST API or use their CLI: OpenAI keys are revoked at platform.openai.com/api-keys; DeepSeek at platform.deepseek.com; SiliconFlow at their developer console. Maintain a plaintext approved-plugin registry in version control and enforce it via a pre-commit hook or onboarding script that diffs installed plugins against the approved list.

Evidence: BEFORE removing plugins or rotating credentials: (1) preserve a copy of each malicious plugin JAR file to `/plugins/` for static reverse engineering — these JARs contain the hardcoded C2 URLs and key-harvesting logic that confirm exfiltration scope; (2) export IDE log files (`idea.log`, `idea.log.1`) from all affected hosts, as they record plugin load events with timestamps that establish when the malicious plugin was first activated; (3) capture a full directory listing with file hashes (`Get-FileHash`) of the plugin directory before deletion to preserve chain of custody; (4) screenshot or export AI provider API key usage logs from OpenAI/DeepSeek/SiliconFlow dashboards showing anomalous call volumes prior to rotation, as these may constitute evidence of unauthorized use.

Step 4: Recovery — Validate that all rotated API keys are active and that old keys are fully revoked in each AI provider's dashboard. Monitor AI provider usage logs for 30 days post-rotation for anomalous requests indicating key reuse. Confirm no malicious plugins remain in developer environments via re-audit. Enable NIST AU-12 (Audit Record Generation) on developer endpoints if not already active. Verify that IDE plugin installation is gated through organizational policy controls before restoring full developer access.

NIST Phase: Recovery

Reference: NIST 800-61r3 §3.5 — Recovery: restore developer environments to verified clean state, confirm old API keys are fully invalidated at the provider level, and establish monitoring to detect any residual attacker access via stolen-but-not-yet-rotated keys.

Controls: NIST AU-12 (Audit Record Generation), NIST AU-6 (Audit Record Review, Analysis, And Reporting), CIS 7.1 (Establish and Maintain a Vulnerability Management Process), CIS 2.1 (Establish and Maintain a Software Inventory)

Compensating: Without enterprise monitoring, configure a daily cron job or scheduled task to diff installed plugin directories against the approved-plugin manifest and alert via email or Slack webhook on any new additions. For AI provider key monitoring, use each provider's usage API endpoint to pull daily token consumption metrics and compare against a 30-day baseline — a simple Python script using the OpenAI `/v1/usage` endpoint or DeepSeek's equivalent can alert on consumption spikes exceeding 2x historical average. Re-audit plugin directories by re-running the enumeration script from Step 1 on all developer workstations before access is restored.

Evidence: This step primarily validates clean state and does not alter live volatile data. However, before restoring full developer network access: confirm that Sysmon EventID 3 logging for IDE processes is active and shipping to a central log store so any re-infection attempt is captured; retain the AI provider usage dashboard exports from the anomalous pre-rotation period for 90 days as they may be needed for cost recovery disputes or regulatory reporting; preserve the re-audit plugin inventory scan results as a clean-state baseline for future drift detection comparisons.

Step 5: Post-Incident — Conduct a control gap review against CIS 2.1 (Software Inventory) and CIS 2.3 (Address Unauthorized Software) to assess whether an approved plugin inventory existed and was enforced. Implement a formal third-party plugin vetting process before Marketplace installs are permitted. Establish a recurring review cadence for installed IDE plugins per CIS 7.1 (Vulnerability Management Process). Brief developers on supply chain risks associated with IDE plugin ecosystems. Map findings to NIST AC-6 (Least Privilege) — evaluate whether developer workstations have excessive access that amplifies the blast radius of credential theft.

NIST Phase: Post Incident

Reference: NIST 800-61r3 §4 — Post-Incident Activity: document lessons learned specific to IDE supply chain risk, update the software allowlist policy to include plugin vetting controls, and brief the developer population on credential hygiene for AI provider API keys.

Controls: NIST AC-6 (Least Privilege), NIST AU-6 (Audit Record Review, Analysis, And Reporting), CIS 2.1 (Establish and Maintain a Software Inventory), CIS 2.2 (Ensure Authorized Software is Currently Supported), CIS 2.3 (Address Unauthorized Software), CIS 7.1 (Establish and Maintain a Vulnerability Management Process), CIS 7.2 (Establish and Maintain a Remediation Process)

Compensating: For plugin vetting without a dedicated AppSec team, establish a lightweight review checklist: (1) confirm plugin publisher identity matches JetBrains Marketplace verified vendor badge; (2) check GitHub repository for the plugin and review recent commits for network-calling code; (3) run the plugin JAR through a free static analysis tool such as semgrep with a custom rule targeting `java.net.HttpURLConnection` or `okhttp3` calls to external hosts; (4) require a minimum install count and published review period before organizational approval. Store the approved plugin list in a shared Git repository as the single source of truth. For developer awareness, conduct a 30-minute tabletop exercise using this specific campaign as the scenario.

Evidence: No volatile evidence capture is required at the post-incident phase. Preserve the following for the lessons-learned record and potential regulatory reporting: (1) the full list of affected developer accounts and their associated AI provider keys with exfiltration timestamps; (2) the Aikido Security report IOC list cross-referenced against your environment's confirmed hits; (3) timeline reconstruction showing when each malicious plugin was first installed versus when it was detected, to quantify the dwell time; (4) AI provider API cost reports covering the campaign window (approximately 8 months prior to discovery) to document financial impact from unauthorized key use.

Detection Guidance

Primary detection vector: outbound plaintext HTTP (port 80) from JetBrains IDE processes to hardcoded external IPs or domains. Query proxy/firewall logs for HTTP (not HTTPS) connections initiated by processes matching JetBrains IDE executables (idea, pycharm, webstorm, goland, clion, rider, datagrip, etc.). Flag any HTTP connections to non-JetBrains infrastructure, excluding whitelisted internal JetBrains update and license servers. Cross-reference destination IPs and domains against the IOC list published by Aikido Security in their original disclosure. Behavioral indicator: API key files or IDE configuration files (stored under user profile JetBrains directories) accessed by plugin classloader processes shortly before or during an outbound HTTP connection. On AI provider platforms, review API key usage dashboards for spikes in token consumption, calls from unrecognized IP ranges, or requests at unusual hours inconsistent with developer working patterns. NIST AU-6 (Audit Record Review) and AU-13 (Monitoring for Information Disclosure) apply directly. D3FEND countermeasures: D3-SFA (System File Analysis) to monitor IDE config and credential file access; D3-LAM (Local Account Monitoring) to detect anomalous process behavior on developer workstations; D3-UAP (User Account Permissions) to restrict IDE plugin write access.

Indicators of Compromise

Type	Value	Context	Confidence
URL	`https://www.aikido.dev/blog/multiple-jetbrains-ide-plugins-caught-stealing-ai-keys`	Aikido Security original disclosure — contains specific plugin names, vendor accounts, and C2 IOCs. Consult directly for current IOC list.	HIGH

Framework Mappings

MITRE-ATTACK

- **T1071.001** — Web Protocols

- **T1041** — Exfiltration Over C2 Channel
- **T1555** — Credentials from Password Stores
- **T1078.004** — Cloud Accounts
- **T1608.001** — Upload Malware
- **T1195.001** — Compromise Software Dependencies and Development Tools
- **T1552.001** — Credentials In Files

NIST-800-53R5

- **CA-7** — Continuous Monitoring
- **SC-7** — Boundary Protection
- **SI-4** — System Monitoring
- **SC-8** — Transmission Confidentiality and Integrity
- **SI-7** — Software, Firmware, and Information Integrity
- **CM-3** — Configuration Change Control
- **IA-5** — Authenticator Management
- **SR-2** — Supply Chain Risk Management Plan

OWASP-TOP10-2021

- **A02:2021** — Cryptographic Failures
- **A08:2021** — Software and Data Integrity Failures
- **A04:2021** — Insecure Design
- **A07:2021** — Identification and Authentication Failures

CIS-V8

- **3.10** — Encrypt Sensitive Data in Transit
- **2.5** — Allowlist Authorized Software
- **2.6** — Allowlist Authorized Libraries
- **5.2** — Use Unique Passwords
- **6.3** — Require MFA for Externally-Exposed Applications
- **15.1** — Establish and Maintain an Inventory of Service Providers

HIPAA-SECURITY

- **164.312(e)(1)** — Transmission Security
- **164.308(a)(5)(ii)(D)** — Password Management
- **164.312(d)** — Person or Entity Authentication

SOC2-TSC

- **CC6.1** — Logical access security software, infrastructure, and architectures
- **CC9.2** — Manages risks associated with vendors and business partners

NIST-CSF-2

- **GV.SC-01** — Cybersecurity supply chain risk management program

ISO-27001-2022

- **A.5.21** — Managing information security in the ICT supply chain

MITRE ATT&CK Mapping

Technique ID	Technique Name	Tactic
T1071.001	Web Protocols	Command-And-Control
T1041	Exfiltration Over C2 Channel	Exfiltration
T1555	Credentials from Password Stores	Credential-Access
T1078.004	Cloud Accounts	Defense-Evasion
T1608.001	Upload Malware	Resource-Development
T1195.001	Compromise Software Dependencies and Development Tools	Initial-Access
T1552.001	Credentials In Files	Credential-Access

Sources

Source	URL	Tier
Security News	https://www.bleepingcomputer.com/news/security/malicious-jetbrains-...	T3
Multiple JetBrains IDE plugins caught stealing AI keys - Aikido Security	https://www.aikido.dev/blog/multiple-jetbrains-ide-plugins-caught-s...	T3
Updates for security issue affecting IntelliJ-based IDEs 2023.1+ and ...	https://blog.jetbrains.com/security/2024/06/updates-for-security-is...	T3
How safe are intellij plugins : r/IntelliJIDEA - Reddit	https://www.reddit.com/r/IntelliJIDEA/comments/1e7f4bv/how_safe_are...	T3
JetBrains AI Enterprise: Securely leverage the power of AI and ...	https://www.jetbrains.com/ide-services/ai-enterprise/	T3

Solutions is not liable for any damages resulting from the use of this information.

Generated 2026-06-17 07:15 UTC by TJS Security Command Center