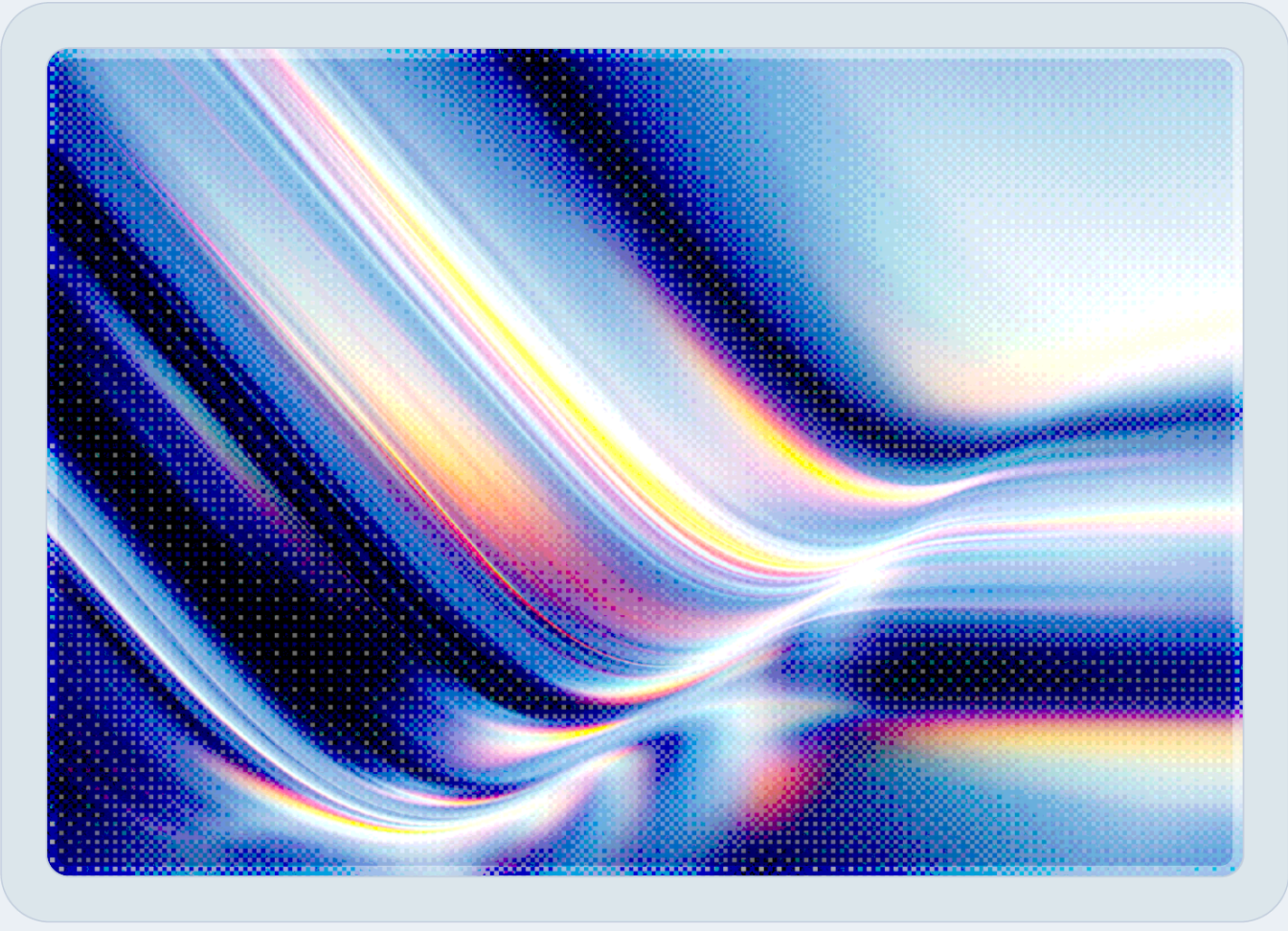
CSAI Foundation | Cloud Security Alliance

TeamPCP's LiteLLM Backdoor: New Data Shows 2,100+ Orgs Exposed

CloudSEK's Reconstructed Dataset Quantifies the March 2026 Trivy-to-LiteLLM Supply Chain Attack

2026-08-12

AI-assisted Rapid Research



CSAI | CSA cloud security alliance®

Key Takeaways

- A dataset that CloudSEK reconstructed from roughly 434,000 files captured by the attackers maps potential exposure from the March 2026 Trivy-to-LiteLLM supply chain campaign to more than 2,100 organizations, a scale that was not quantified when the incident was first disclosed five months ago [1][3].
- The campaign began on March 19, 2026, when the threat cluster tracked as TeamPCP exploited an incomplete credential rotation at Aqua Security to force-push malicious commits into 76 of 77 `trivy-action` version tags and all seven `setup-trivy` tags, then used the stolen CI/CD credentials to cascade into Checkmarx KICS on March 23 and LiteLLM on March 24 [2][4][5].
- This note assesses the LiteLLM step as the campaign's most consequential: two poisoned PyPI releases (1.82.7 and 1.82.8) used a Python `.pth` auto-execution file to harvest cloud credentials, SSH keys, Kubernetes secrets, database passwords, and AI provider API keys from any organization that installed the compromised gateway, which routes traffic for more than 100 large language model providers [2][6].
- CISA added the campaign's vulnerability, CVE-2026-33634, to its Known Exploited Vulnerabilities catalog on March 26, 2026, and the FBI's July 2 FLASH advisory warned that credentials stolen during the initial compromise window are likely to be weaponized by affiliated actors long after the fact, independent of any new CloudSEK findings [7][8].
- CloudSEK's dataset lists high-confidence exposure matches for major organizations including NVIDIA, Cisco, Siemens, Deloitte, and Volkswagen, but CloudSEK is explicit that a match indicates potential exposure rather than confirmed compromise, and that affected organizations must independently verify whether the poisoned package was actually downloaded, cached, or executed in their environment [1][3].

Background

Trivy is an open-source vulnerability scanner maintained by Aqua Security that is widely embedded in continuous integration and continuous delivery (CI/CD) pipelines to scan container images, filesystems, and infrastructure-as-code for known vulnerabilities before deployment. Because it runs inside the pipeline itself, a Trivy scan step typically executes with access to the same secrets, tokens, and network

reach as the rest of the build job — a property that made it an attractive entry point rather than an end target for the threat cluster now tracked as TeamPCP, and by Google's Threat Intelligence Group as UNC6780 [4][5]. TeamPCP had reportedly gained an initial foothold in Aqua's publishing infrastructure in February 2026; when the resulting credential rotation was incomplete, the group retained enough access to generate new, valid publishing tokens ahead of the rotation and used them on March 19 to overwrite 76 of the 77 version tags in the `aquasecurity/trivy-action` GitHub Action, all seven tags in `aquasecurity/setup-trivy`, and to publish a trojanized Trivy binary as version 0.69.4 [2][4]. Because GitHub Actions are commonly referenced by mutable version tags rather than pinned commit hashes, any pipeline that invoked `trivy-action` by tag during the compromise window silently pulled the malicious code on its next run.

The campaign then moved downstream in a deliberate sequence rather than a single blast. On March 23, using credentials harvested from compromised Trivy runners, TeamPCP compromised the GitHub Actions workflow for Checkmarx KICS — a static-analysis tool with a similar elevated-pipeline-privilege profile — and published trojanized OpenVSX extensions, including a poisoned `cx-dev-assist` 1.7.0; Checkmarx removed the malicious code within roughly three hours of discovery [2][5]. The following day, March 24, the campaign reached LiteLLM, an open-source Python proxy and software development kit that gives organizations a single API interface for over 100 large language model providers, including OpenAI, Anthropic, Google Gemini, and AWS Bedrock. LiteLLM's own publishing token had been exposed through a poisoned Trivy binary running inside its CI pipeline, and attackers used it to push two malicious releases, 1.82.7 and 1.82.8, directly to PyPI [1][6]. LiteLLM's maintainers detected the intrusion and suspended new releases within hours, but by then the packages had already been downloaded by an unknown number of the tool's substantial user base. The campaign concluded on March 27 with a compromise of the Telnyx Python SDK, which introduced a novel evasion technique — a steganographic payload embedded in audio files — that had not appeared in the earlier stages [2][4].

The reason the LiteLLM step drew the most sustained attention is architectural rather than incidental. Organizations deploy LiteLLM as a centralized "AI gateway" that sits between internal applications and upstream model providers, handling routing, rate limiting, and logging. That role means a single LiteLLM instance frequently holds authentication material for an organization's entire AI provider ecosystem in one place — master keys, per-provider API keys, database connection strings, and routing secrets — making it a high-value target whose compromise has a disproportionately large blast radius relative to its footprint in a typical technology stack. This concentration risk is the same structural issue CSA's own research on LiteLLM's later-disclosed vulnerability chains has independently flagged, and it is what elevated the March supply chain compromise from a routine credential-theft incident to a campaign with cross-industry reach.

Security Analysis

The technical mechanism behind the LiteLLM compromise combined two distinct payload paths. Malicious code was inserted directly into `proxy_server.py`, LiteLLM's core routing module, while version 1.82.8 additionally shipped a file named `litellm_init.pth`, which exploits a long-standing Python packaging feature: files with a `.pth` extension placed in a package's site-packages directory are automatically executed by the interpreter at startup, before any application code runs [2]. This gave the malware execution on essentially every process that imported the poisoned package, independent of whether the application ever actually invoked LiteLLM's proxy functionality. Once running, the payload performed systematic reconnaissance and exfiltration: it collected environment variables containing provider keys such as `OPENAI_API_KEY` and `ANTHROPIC_API_KEY`, searched for AWS and GCP credentials, extracted Kubernetes service account tokens from the standard `/run/secrets/kubernetes.io/serviceaccount` mount path, harvested connection strings for MySQL, PostgreSQL, MongoDB, Redis, and HashiCorp Vault, scanned for SSH keys and Slack or Discord webhook URLs, and probed for cryptocurrency wallet data associated with the Solana blockchain [2]. Stolen data was encrypted and exfiltrated to a typosquatted command-and-control domain, `models.litellm[.]cloud`, with a secondary exfiltration path through a GitHub repository the attackers controlled [2]. Trivy's own compromise used a parallel typosquat, `scan.aquasecurtiy[.]org`, and the Checkmarx stage used `checkmarx[.]zone` [2].

Two additional payload behaviors distinguish this campaign from a conventional credential-stealing worm. First, researchers identified a geofenced destructive component: if the malware detected that a compromised system's time zone or primary language indicated Iran, it would attempt to wipe the host Kubernetes cluster and its nodes rather than simply exfiltrate data, while systems elsewhere in the world instead received a self-replicating payload researchers named CanisterWorm, which propagated laterally across the npm ecosystem [2][17]. Second, TeamPCP's broader infrastructure reportedly used Internet Computer Protocol blockchain canisters for parts of its command-and-control layer, a technique that resists conventional domain-based takedown because canister addresses are not subject to registrar or DNS-level seizure in the way typosquatted domains are [9][15]. Downstream, credentials harvested during the March compromise window were later linked to a breach of Cisco's development environment in which more than 300 internal repositories, including source code for Cisco AI products, were exfiltrated [9][13]. The same threat cluster, TeamPCP/UNC6780, was separately responsible for an 18-minute trojanization of the Nx Console VS Code extension in May that resulted in the theft of roughly 3,800 internal GitHub repositories, though that intrusion's immediate access vector was a distinct compromise of TanStack npm packages beginning May 11 rather than a confirmed direct reuse of the March credential set [9][14].

The scale disclosed in August 2026 is new information layered onto an already-documented campaign rather than a newly discovered breach. CloudSEK obtained and reconstructed a dataset built from approximately 434,000 files the attackers had captured during the compromise window, and used it to map potential exposure to more than 2,100 organizations by The Hacker News's count and more than 2,500 by CloudSEK's own published figure [1][3]. The dataset includes high-confidence exposure matches — meaning data associated with the organization appeared in the captured files, not confirmation that the organization's environment was compromised or that stolen data was misused — for NVIDIA, AWS, Samsung, Salesforce, Cisco, ServiceNow, Siemens, S&P Global, Deloitte, Vodafone, BT, Zscaler, and X Corp, among others [1][3]. A smaller number of confirmed, rather than high-confidence-matched, impacts have been separately reported: Checkmarx confirmed unauthorized access to its own GitHub repositories stemming from the campaign, the AI-driven recruiting platform Mercor confirmed unauthorized activity traceable to the malicious LiteLLM versions, and CERT-EU reported that the European Commission had approximately 91.7 gigabytes of compressed data exfiltrated from an AWS account as a direct result of the compromise [1]. Table 1 summarizes the campaign's sequence.

Date (2026)	Target	Mechanism	Outcome
Mar 19	Trivy / `trivy-action`	Stolen publishing tokens used to force-push malicious commits to 76 of 77 version tags	Trojanized binary (v0.69.4) and GitHub Action; CVE-2026-33634 assigned
Mar 23	Checkmarx KICS / OpenVSX	Credentials harvested from compromised Trivy CI runners	Trojanized `cx-dev-assist` 1.7.0 extension; removed within ~3 hours
Mar 24	LiteLLM (PyPI)	LiteLLM's publishing token exposed via poisoned Trivy binary in its own CI pipeline	Malicious releases 1.82.7, 1.82.8; multi-stage credential stealer via `.pth` auto-execution
Mar 27	Telnyx Python SDK	Credentials from prior stages	Steganographic payload embedded in audio files

CISA added CVE-2026-33634 to its Known Exploited Vulnerabilities catalog on March 26, 2026, with a CVSS score of 9.4, reflecting a flaw that grants an attacker access to essentially everything present in a compromised CI/CD environment: tokens, SSH keys, cloud credentials, database passwords, and any

sensitive configuration held in memory [7]. The FBI's July 2 FLASH advisory, FLASH-20260702-01, reiterated that the campaign's downstream risk does not expire with initial remediation; the bureau specifically warned that TeamPCP-affiliated actors are likely to continue weaponizing credentials exfiltrated during the March compromise window for an extended period, a warning the Cisco breach appears to substantiate directly. The Nx Console/GitHub breach reflects the same actor group's continued operational tempo, though it proceeded through a separate access chain — a May 2026 compromise of TanStack npm packages — rather than a confirmed reuse of the March credential set specifically [8][9][13][14].

Recommendations

Immediate Actions

Organizations that ran `trivy-action`, `setup-trivy`, Trivy binaries, Checkmarx KICS GitHub Actions, or LiteLLM versions 1.82.7 or 1.82.8 at any point between March 19 and March 27, 2026, should treat any credentials present in those CI/CD environments during that window as compromised and rotate them, rather than waiting for confirmation of misuse. This includes cloud provider access keys, Kubernetes service account tokens, database connection credentials, AI provider API keys (OpenAI, Anthropic, and others routed through LiteLLM), package-registry publishing tokens, and SSH keys accessible from affected runners. Security teams should also search build logs and network telemetry from that period for outbound connections to the campaign's known typosquatted infrastructure — `models.litellm[.]cloud`, `scan.aquasecurtiy[.]org`, and `checkmarx[.]zone` — and for the presence of a `litellm_init.pth` file in any Python environment where LiteLLM was installed, since its presence is a direct indicator of the malicious 1.82.8 release [2].

Short-Term Mitigations

Because GitHub Actions referenced by mutable tags were the initial entry point for this entire campaign, organizations should pin third-party GitHub Actions, including security-scanning actions, to specific commit hashes rather than version tags, and should apply the same pinning discipline to PyPI, npm, and other package dependencies used inside CI/CD pipelines. Standing up a private package mirror with an approval gate for new versions of high-privilege developer tooling — vulnerability scanners, static analyzers, and AI gateways among them — adds a review step that could plausibly have interrupted this campaign at multiple stages, provided reviewers were equipped to detect anomalous release behavior rather than merely confirming the publisher's identity. Organizations should also review whether their

CI/CD runners need the broad outbound network access and long-lived credential exposure that made the exfiltration paths in this campaign effective, since restricting egress from build environments limits the value of any single compromised runner regardless of which tool was the entry point.

Strategic Considerations

The recurrence of credential-concentration risk across this campaign — first in CI/CD runners generally, then specifically in LiteLLM's role as a single point holding an organization's entire AI provider credential set — argues for architectural change rather than only faster patching. Organizations that operate an AI gateway should move toward scoped, per-application API keys and externalized secrets management (such as HashiCorp Vault or a cloud provider's secrets manager) rather than storing master provider credentials directly within the gateway's configuration or environment, so that a future compromise of the gateway process does not automatically yield the entire organization's AI provider credential set. More broadly, the FBI's warning that stolen credentials from this campaign will likely be weaponized well after the initial compromise means organizations should not treat "we patched in March" as closing the incident; a durable response requires periodic re-verification that credentials present in affected environments during the compromise window have in fact been rotated, not merely that vulnerable software has been updated.

CSA Resource Alignment

CSA's most directly relevant prior work is TeamPCP (UNC6780): AI Supply Chain's Most Active Threat Actor, published in May 2026, which tracks this same threat cluster across the Trivy, Checkmarx, LiteLLM, Telnyx, and Bitwarden CLI compromises and documents the downstream Cisco breach that reused credentials from the March campaign, as well as the Nx Console/GitHub breach carried out by the same actor group through a separate May 2026 access chain [9]. CSA also published a contemporaneous note within days of the March 24 LiteLLM compromise, LiteLLM PyPI Backdoor: Credential Theft in AI Toolchains, which documented the initial technical mechanism before the downstream exposure scale was known [16]. This note's contribution is narrower and more current than either: it quantifies, for the first time using CloudSEK's reconstructed dataset, the scale of organizational exposure from the LiteLLM stage specifically, a data point that postdates both of CSA's earlier notes on this campaign.

CSA's npm Supply Chain Under Siege: TeamPCP, Miasma, and npm v12 whitepaper [10] provides the deeper structural analysis of why this campaign succeeded and connects it to the ecosystem-level response, including npm's move to disable install-script execution by default and require explicit

allowlisting for Git-sourced dependencies. Organizations implementing this note's short-term mitigations around dependency pinning and registry mirroring should read that whitepaper's governance analysis alongside CSA's AI Controls Matrix (AICM) v1.1, particularly its Supply Chain Management and Threat and Vulnerability Management domains, for control-level guidance on vetting and monitoring high-privilege developer tooling [12].

CSA's LiteLLM AI Gateway: KEV-Listed Attack Chain Enables Full Takeover research note [11], while centered on a separate set of vulnerabilities disclosed later in 2026, independently reaches the same structural conclusion this note does about LiteLLM's role as a credential-concentration point, and its recommendations on scoped API keys and externalized secrets management directly reinforce the strategic guidance above. Read together, the four artifacts show a consistent pattern across 2026: LiteLLM's architectural position as a unified AI gateway has made it a recurring target through both supply chain compromise and direct vulnerability exploitation, and organizations should evaluate their LiteLLM deployments against both threat categories rather than treating them as unrelated incidents.

References

[1] The Hacker News, "Malicious LiteLLM Releases Tied to Trivy Hack May Have Exposed 2,100+ Organizations," The Hacker News, August 2026.

[2] Kaspersky, "Trojanization of Trivy, Checkmarx, and LiteLLM solutions," Kaspersky Daily, 2026.

[3] CloudSEK, "2,500+ Companies and 434,000 CI/CD Pipelines Exposed in the Largest AI Supply Chain Breach of 2026," CloudSEK Blog, August 2026.

[4] Arctic Wolf, "TeamPCP Supply Chain Attack Campaign Targets Trivy, Checkmarx (KICS), and LiteLLM (Potential Downstream Impact to Additional Projects)," Arctic Wolf Blog, 2026.

[5] The Register, "LiteLLM infected with credential-stealing code via Trivy," The Register, March 24, 2026.

[6] BerriAI/LiteLLM, "Security Update: Suspected Supply Chain Incident," LiteLLM Documentation Blog, March 2026.

[7] Cybersecurity and Infrastructure Security Agency, "CISA Adds One Known Exploited Vulnerability to Catalog," CISA, March 26, 2026.

[8] Federal Bureau of Investigation, "FLASH-20260702-01: TeamPCP Supply Chain Campaign," FBI Internet Crime Complaint Center, July 2, 2026.

[9] Cloud Security Alliance, "TeamPCP (UNC6780): AI Supply Chain's Most Active Threat Actor," CSA AI Safety Initiative, May 2026.

[10] Cloud Security Alliance, "npm Supply Chain Under Siege: TeamPCP, Miasma, and npm v12," CSA AI Safety Initiative, June 2026.

[11] Cloud Security Alliance, "LiteLLM AI Gateway: KEV-Listed Attack Chain Enables Full Takeover," CSA AI Safety Initiative, June 17, 2026.

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

[13] SANS Internet Storm Center, "TeamPCP Supply Chain Campaign: Update 007 – Cisco Source Code Stolen via Trivy-Linked Breach," SANS ISC, April 8, 2026.

[14] Help Net Security, "GitHub, Grafana Labs breaches traced back to TanStack supply chain compromise," Help Net Security, May 21, 2026.

[15] Aikido Security, "CanisterWorm Gets Teeth: TeamPCP's Kubernetes Wiper Targets Iran," Aikido Blog, March 2026.

[16] Cloud Security Alliance, "LiteLLM PyPI Backdoor: Credential Theft in AI Toolchains," CSA AI Safety Initiative, March 27, 2026.

[17] Krebs on Security, "'CanisterWorm' Springs Wiper Attack Targeting Iran," Krebs on Security, March 2026.

