CSAI

cloud security alliance®

CSAI Foundation

Cloud Security Alliance AI Safety Initiative

TeamPCP: Cascading Supply Chain Assault via Developer Security Tooling

How Trivy, KICS, and LiteLLM Became Vectors in a Five-Day CI/CD Campaign

Unofficial AI-assisted Research

2026-04-05

Key Takeaways

- **TeamPCP**, a financially motivated threat group active since mid-2025, executed a coordinated five-day supply chain campaign in March 2026 that sequentially compromised Aqua Security's Trivy, Checkmarx's KICS, and BerriAI's LiteLLM — three widely deployed developer security tools.
- The attack chain operated as a cascade: credentials stolen from one victim's CI/CD pipeline became the access vector for the next target, demonstrating that developer security tooling occupies a position of extraordinary trust within modern software pipelines.
- The European Commission's use of the compromised Trivy release led to the exfiltration of AWS management credentials, a subsequent data breach affecting 42 internal clients and at least 29 additional EU entities, and the theft of approximately 340 GB of uncompressed data.
- **CVE-2026-33634** (CVSS4B 9.4, Critical) has been assigned to the Trivy ecosystem compromise and was added to CISA's Known Exploited Vulnerabilities Catalog on March 26, 2026 [1].
- Organizations running any version of Trivy between v0.69.4 and v0.69.6, `trivy-action` tags v0.0.1–v0.34.2, `setup-trivy` tags prior to v0.2.6, KICS prior to v2.3.29, or LiteLLM 1.82.7–1.82.8 should treat their CI/CD environments as fully compromised and rotate all secrets immediately.

Background

The Group: TeamPCP

TeamPCP is a financially motivated cybercriminal threat group operating under multiple aliases — PCPcat, ShellForce, DeadCatx3, and Persy_PCP — with Telegram handles @Persy_PCP and @teampcp. The group's earliest documented Telegram activity dates to July 30, 2025. It was first formally characterized by security researcher "Beelzebub" in December 2025 under the designation "Operation PCPcat" [2].

Through late 2025, TeamPCP operated primarily as a cloud-native ransomware and cryptomining actor. Their December 2025 "React2Shell" campaign exploited CVE-2025-55182 to target exposed Docker APIs, Kubernetes clusters, Ray dashboards, and Redis servers, deploying a worm-driven botnet that peaked around December 25 before going quiet [2]. Arctic Wolf Labs assessed the group may operate in part as an Initial Access Broker (IAB), selling harvested credentials to other threat actors including the CipherForce and Vect ransomware affiliates [3].

By mid-March 2026, TeamPCP had pivoted from opportunistic cloud exploitation to precision supply chain compromise. The shift to supply chain targeting required significantly greater preparation — it demanded an understanding of CI/CD trust relationships and the construction of a multi-step access chain — and offered substantially broader victim reach per initial compromise, targeting tools that developers implicitly trust to run with elevated permissions inside their pipelines.

The Attack Architecture: Cascading Credential Theft

What distinguished the March 2026 campaign was not any single compromise but its architecture. Each targeted tool served as both a victim and a vector. Credentials exfiltrated from a Trivy pipeline provided the authentication material needed to compromise KICS; credentials harvested from KICS' downstream victims enabled the LiteLLM PyPI injection. The result was a self-reinforcing cascade in which the population of affected organizations expanded with each stage.

The payload design was consistent across all three targets. A collection stage scraped `Runner.Worker` process memory via `/proc/<pid>/mem`, bypassing GitHub's secret-masking mechanism, and swept fifty or more filesystem paths for SSH keys, cloud provider tokens (AWS, GCP, Azure), Kubernetes configuration files, Docker credentials, environment files, cryptocurrency wallets, and LLM API keys from OpenAI and Anthropic [4]. A second stage encrypted the harvested material using AES-256-CBC with RSA-4096 hybrid encryption into an archive named `tpcp.tar.gz`. Exfiltration proceeded to typosquatted command-and-control domains (e.g., `scan.aquasecurtiy[.]org`, note the deliberate misspelling of "aquasecurity"); if the primary channel failed, the payload was uploaded as a GitHub Release Asset to a repository named `docs-tpcp` created in the victim's own GitHub organization [4]. A secondary C2 tier using Internet Computer Protocol (ICP) canisters provided decentralized, censorship-resistant fallback infrastructure.

The malware was self-attributed in its string table: "TeamPCP Cloud stealer."

Security Analysis

Stage One: Trivy (March 19, 2026)

Trivy is Aqua Security's open-source container and filesystem vulnerability scanner, a widely deployed security scanning tool in CI/CD pipelines. By design, Trivy runs with elevated permissions and access to build secrets — a characteristic that made it a high-value target for credential theft at scale.

The initial breach of Aqua Security infrastructure occurred in late February 2026. A `pull_request_target` workflow misconfiguration — commonly called a "PwnRequest" flaw — provided initial access. Critically, when Aqua Security discovered the breach and attempted to rotate credentials for the `aqua-bot` service account, the rotation window was not completed cleanly. TeamPCP intercepted the newly rotated personal access token during that window, maintaining persistent access under the appearance of legitimate service account activity [4][5]. The attacker then made commits using spoofed contributor identities modeled after known Aqua Security contributors to blend with the project's commit history.

On March 19 at 17:43 UTC, TeamPCP force-pushed version tags to point to malicious commits rather than modifying source code directly. This technique is particularly difficult to detect: diff-based code review tools and dependency pinning practices both fail to catch it, because the tag itself appears unchanged while the commit it references has been silently replaced. Seventy-six of the seventy-seven `trivy-action` tags and all seven `setup-trivy` tags were hijacked [5][16]. Malicious Docker images carrying the payload (v0.69.5 and v0.69.6) were published March 22–23, extending the exposure window by approximately ten hours.

The Trivy binary v0.69.4 was exposed for approximately three hours (18:22–21:42 UTC); `trivy-action` tags were compromised for approximately twelve hours. Safe versions are v0.69.2 and v0.69.3 for the binary; v0.35.0 or later for the action; v0.2.6 or later for setup-trivy [5].

Stage Two: KICS (March 23, 2026)

KICS — "Keeping Infrastructure as Code Secure" — is Checkmarx's open-source scanner for Terraform, CloudFormation, Kubernetes manifests, and other infrastructure-as-code configurations. Its position in CI/CD pipelines mirrors Trivy's: it runs early in the pipeline with access to credentials needed for cloud environment validation.

TeamPCP leveraged GitHub personal access tokens stolen from Trivy campaign victims to authenticate as the `cx-plugins-releases` service account (GitHub user ID 225848595). Between 12:58 and 16:50 UTC on March 23, all 35 KICS version tags were force-pushed to malicious commits — the same technique used against Trivy, applied with speed sufficient to remain active for nearly four hours before a user filed a GitHub issue triggering repository takedown [6][18]. Version 2.3.28 was additionally poisoned. The KICS payload added a Kubernetes-specific persistence mechanism: the malware attempted to deploy a highly privileged DaemonSet to any Kubernetes cluster accessible via harvested kubeconfig files.

Concurrently, a worm variant designated CanisterWorm was observed using the ICP canister C2 infrastructure to propagate across Kubernetes environments, masquerading as `systemd` services and the PostgreSQL utility `pgmon`, and executing recursive file deletion in affected clusters [4].

Stage Three: LiteLLM (March 24, 2026)

LiteLLM, developed by BerriAI, is an open-source Python library and proxy server that provides a unified interface for over one hundred LLM provider APIs, including OpenAI, Anthropic, Cohere, and others. The library reports approximately 95 million monthly PyPI downloads [21], reflecting its broad adoption as AI infrastructure in production environments.

TeamPCP obtained LiteLLM's PyPI publishing credentials from CI/CD pipelines that had been running the compromised Trivy GitHub Action — specifically, pipelines in which Trivy's credential harvest had captured the secrets needed to publish to PyPI. At 10:39 UTC on March 24, versions 1.82.7 and 1.82.8 were published to PyPI. PyPI quarantined both packages approximately forty minutes later [7].

The payload methodology for LiteLLM introduced a meaningful escalation. Version 1.82.7 embedded a base64-encoded stager directly within `litellm/proxy/proxy_server.py`, which executed on `import litellm.proxy`. Version 1.82.8 added a `litellm_init.pth` file to the Python `site-packages/` directory — a technique that fires the payload on every Python interpreter startup, regardless of whether LiteLLM is imported at all [7]. The C2 domain `models.litellm[.]cloud` typosquatted LiteLLM's legitimate domain. The payload specifically targeted LLM provider API keys — a logical extension of the group's interest in high-value credentials that can be monetized directly or sold to other actors.

The last clean release before the compromise is version 1.82.6.

The European Commission Breach

The European Commission's cyber operations team was running Trivy as part of its cloud CI/CD pipeline. On March 19, during the active compromise window, the Commission's pipeline executed the malicious Trivy release and exfiltrated an AWS API key carrying management rights over multiple Commission AWS accounts. TeamPCP used that key to access the Commission's AWS environment, created additional access keys on existing AWS users to maintain persistence, and conducted reconnaissance across the environment [8].

CERT-EU was notified on March 25. The Commission's security operations center had detected suspicious activity the previous day, March 24, and revoked the compromised credentials [8]. On March 27, the Commission publicly disclosed the incident. The stolen dataset — approximately 91.7 GB compressed, estimated at 340 GB uncompressed — included personal data (names, email addresses, usernames), 51,992 files of outbound email communications, and website databases affecting 42 internal Commission clients and at least 29 additional EU entities using the europa.eu web hosting service [8][9].

On March 28, the data extortion group ShinyHunters published the stolen dataset on their dark web leak site, apparently having independently obtained or purchased data that TeamPCP had already exfiltrated. TeamPCP publicly denied collaboration with ShinyHunters, suggesting the secondary leak was an opportunistic action by a separate actor operating on stolen material [10]. The Commission began notifying affected clients on March 31.

Scope and Scale

Across the five-day campaign, TeamPCP reportedly exfiltrated approximately 500,000 credentials and more than 300 GB of data from the broader victim population [4]. Using harvested npm publish tokens gathered from CI/CD pipelines, the group infected at least 45 additional npm packages within a sixty-second window using pre- and post-install script injection [20]. The Telnyx Python SDK versions 4.87.1 and 4.87.2 were also compromised on March 27 using a distinct technique: WAV steganography concealed encrypted second-stage payloads inside audio files included with the SDK distribution [20]. At least 16 victim organizations have been publicly named across threat intelligence disclosures [19].

Recommendations

Immediate Actions

Organizations that ran any of the affected tools within their CI/CD pipelines during the exposure windows described above should treat the environment as compromised. The following actions are not precautionary — they are incident response steps.

Rotate all secrets accessible to the affected pipeline, without exception. This includes cloud provider access keys and service account credentials (AWS, GCP, Azure), GitHub personal access tokens and service account PATs, container registry credentials, PyPI and npm publish tokens, SSH keys, Kubernetes service account tokens, and all LLM API keys (OpenAI, Anthropic, and any other provider). The payload was specifically designed to sweep for exactly these credential types, and selective rotation is insufficient.

Upgrade to safe versions immediately: Trivy v0.69.2 or v0.69.3 (binary), trivy-action v0.35.0 or later, setup-trivy v0.2.6 or later, KICS v2.3.29 or later, and LiteLLM v1.82.9 or later (the first clean release after v1.82.6) [5][6][7]. Pin all GitHub Actions to full commit SHAs rather than mutable tags — this is the structural defense against tag-hijacking techniques and should be implemented regardless of whether compromise is confirmed.

Review CI/CD logs for the March 19–27 window for any of the indicator domains or file patterns associated with this campaign. Key indicators include outbound connections to `scan.aquasecurtiy[.]org`, `checkmarx[.]zone`, `models.litellm[.]cloud`, and `tdtqy-oyaaa-aaaae-af2dq-cai.raw.icp0[.]io`; the presence of files named `kamikaze.sh`, `kube.py`, `prop.py`, `tpcp.tar.gz`, or `litellm_init.pth`; and the appearance of repositories named `docs-tpcp` or `tpcp-docs-*` within any GitHub organization with pipeline access [4].

Short-Term Mitigations

The core structural vulnerability in this campaign was the gap between how these tools are trusted and how the supply chains that produce them are secured. Pipeline jobs that run third-party scanning tools should operate under the principle of least privilege: scanning jobs should not have access to production deployment credentials, and those credentials should never be co-present in the same execution environment unless operationally required. GitHub Actions should use per-job OIDC tokens with tightly scoped IAM policies rather than long-lived secrets injected as environment variables.

Implement cryptographic verification for all third-party tooling consumed in CI/CD pipelines. Sigstore/cosign verification for container images, SLSA provenance attestations for build artifacts, and pip-audit or npm audit checks on package installation are concrete controls that would have materially limited the blast radius of this campaign. SLSA Level 2 and above provenance attestations, in environments where build-system access is not also compromised, create a detectable gap between published provenance and any attacker-substituted commit. Where tooling does not yet support these verification mechanisms, consider substituting alternatives that do, or accepting the residual risk with documented compensating controls.

Audit `pull_request_target` workflow configurations across all GitHub-hosted repositories. This workflow trigger is a well-documented attack surface — it grants write permissions to workflows that can be triggered by external pull requests — and its misconfiguration was the root initial access vector in the Trivy compromise [5]. Workflows that require write access should require manual approval for first-time contributors and should not expose secrets to untrusted code paths.

Strategic Considerations

This campaign exemplifies an increasingly prevalent and sophisticated threat model for developer security tooling — one in which the tools responsible for pipeline security become the attack vector. Supply chain attacks targeting CI/CD infrastructure are not novel: the SolarWinds compromise (2020), the Codecov breach (2021), and the XZ Utils incident (2024) all demonstrated that developer tooling represents a high-value target. What distinguishes the March 2026 campaign is the cascading credential-as-vector design and the speed with which TeamPCP pivoted across three separate targets within five days, transforming each victim organization into an unwitting staging ground for the next compromise.

The cascading structure of this campaign also illustrates the systemic risk embedded in shared CI/CD credential patterns. When a vulnerability scanner, an IaC checker, and an AI proxy all authenticate to the same credential stores — or when a compromise in any one of them yields credentials usable against the others — the aggregate attack surface is the union of all three. Compartmentalizing CI/CD credentials by pipeline function, enforcing separate identity contexts for security scanning vs. deployment operations, and auditing token scopes against a least-privilege baseline are organizational practices that reduce cascade potential regardless of which tool becomes the initial entry point.

CSA Resource Alignment

This incident maps directly to multiple layers of the CSA research corpus and underscores the operational relevance of several current frameworks.

The **Six Pillars of DevSecOps** framework — particularly Pillar 4 (Bridging Compliance and Development) — addresses Software Composition Analysis and Compliance-as-Code practices that, when applied with integrity verification controls, could have flagged the use of tampered tooling in consuming organizations' pipelines. Pillar 6's guidance on continuous monitoring metrics provides the measurement framework needed to detect anomalous behavior from security scanning jobs [12].

The **MAESTRO** framework's application to CI/CD pipeline security (specifically the CSA blog analysis "Applying MAESTRO to Real-World Agentic AI Threat Models from Framework to CI/CD Pipeline") addresses the threat model under which security tooling operates as a privileged agent within automated pipelines. The TeamPCP campaign is a concrete instance of the threat scenarios MAESTRO Layer 3 (Agent Frameworks) and Layer 4 (Execution Environments) describe, in which compromised orchestration components propagate adversarial control across the pipeline [13].

CSA's **Software Supply Chain Security** guidance on SBOM adoption and SLSA framework implementation directly addresses the artifact provenance gap exploited in this campaign. Tag-hijacking attacks succeed because, in the absence of cryptographic integrity verification for tagged releases, consumers have no mechanism to distinguish a legitimate tag from a silently substituted one. SLSA Level 2 provenance attestation specifically addresses this gap by binding a signed build provenance record to each published artifact [14].

The **Agentic AI Red Teaming Guide** explicitly catalogs Supply Chain and Dependency Attacks as a test category within its twelve-category framework. LiteLLM's position as AI infrastructure — a tool that mediates access to LLM providers and handles API keys for multiple AI services — means its compromise sits at the intersection of traditional supply chain risk and emergent AI system risk. Organizations deploying AI proxies in production should include supply chain compromise scenarios in their red team exercises [15].

Finally, the **CCM** control domain DSP-07 (Supply Chain Transparency) and SEF-05 (Incident Management) are directly applicable. Organizations leveraging STAR assessments for cloud service provider evaluations should now include explicit inquiry into the security posture of developer tooling used within vendor CI/CD pipelines — because, as this campaign demonstrates, those tools represent a meaningful component of the vendor's security boundary.

References

[1] CISA. "CISA Adds One Known Exploited Vulnerability to Catalog (CVE-2026-33634)." CISA, March 26, 2026.

[2] Flare.io. "TeamPCP: Cloud-Native Ransomware Group Threat Alert." Flare Threat Intelligence, March 2026.

[3] Arctic Wolf. "TeamPCP Supply Chain Attack Campaign Targets Trivy, Checkmarx KICS, and LiteLLM." Arctic Wolf Labs, March 2026.

[4] Unit 42 / Palo Alto Networks. "Weaponizing the Protectors: TeamPCP's Supply Chain Campaign." Palo Alto Networks, April 2026.

[5] Aqua Security. "GHSA-69fq-xp46-6x23: Trivy GitHub Actions Supply Chain Compromise." GitHub Security Advisory, March 2026.

[6] Checkmarx. "Checkmarx Security Update: KICS GitHub Action Compromise." Checkmarx Blog, March 2026.

[7] Datadog Security Labs. "LiteLLM and Telnyx Compromised on PyPI: TeamPCP Supply Chain Campaign Continues." Datadog, March 2026.

[8] CERT-EU. "European Commission Cloud Breach via Trivy Supply Chain Attack." CERT-EU, March 2026.

[9] Bleeping Computer. "CERT-EU: European Commission Hack Exposes Data of 30+ EU Entities." Bleeping Computer, March 2026.

[10] Dark Reading. "TeamPCP Attacks, Hacker Infighting: ShinyHunters Leaks Data Group Claims It Didn't Share." Dark Reading, March 2026.

[11] Wiz. "Three's a Crowd: TeamPCP Trojanizes LiteLLM in Continuation of Campaign." Wiz Blog, March 2026.

[12] Cloud Security Alliance. "The Six Pillars of DevSecOps: Pillar 4 — Bridging Compliance and Development." CSA, 2024. *(URL inaccessible as of 2026-04-05; search CSA website for current link.)*

[13] Cloud Security Alliance. "Applying MAESTRO to Real-World Agentic AI Threat Models: From Framework to CI/CD Pipeline." CSA Blog, February 2026.

[14] Cloud Security Alliance. "Securing the Software Supply Chain: Transparency in the Age of the Software-Driven Society." CSA, 2025. *(URL inaccessible as of 2026-04-05; search CSA website for current link.)*

[15] Cloud Security Alliance. "Agentic AI Red Teaming Guide." CSA, 2026. *(URL inaccessible as of 2026-04-05; search CSA website for current link.)*

[16] Wiz. "Trivy Compromised by TeamPCP: Supply Chain Attack Details." Wiz Blog, March 2026.

[17] The Hacker News. "Trivy Security Scanner GitHub Actions Breached, 75+ Tags Hijacked." The Hacker News, March 2026.

[18] The Hacker News. "TeamPCP Hacks Checkmarx via Stolen CI Credentials, KICS GitHub Action Poisoned." The Hacker News, March 2026.

[19] Phoenix Security. "TeamPCP's Five-Day Siege: Trivy, Checkmarx, GitHub Actions, npm, and CanisterWorm." Phoenix Security, April 2026.

[20] ramimac.me. "TeamPCP IOC Timeline and Indicators of Compromise." Independent Security Research, April 2026.

[21] PyPI Stats. "LiteLLM Download Statistics." pypistats.org, accessed April 2026.