TeamPCP: Cascading Supply Chain Attack on AI/ML Tooling

How a Cascading Credential Harvest Backdoored LLM Gateways, Security Scanners, and the npm Ecosystem

Unofficial AI-assisted Research

Cloud Security Alliance AI Safety Initiative

2026-03-30

Key Takeaways

- **TeamPCP** is an operationally sophisticated threat actor—evidenced by multi-stage cascading infrastructure, a novel C2 mechanism, and deliberate pre-positioning beginning months before activation—that executed a cascading software supply chain campaign between March 19–27, 2026, deliberately targeting the AI/ML developer toolchain to harvest API keys for every major commercial LLM provider simultaneously.
- The campaign followed a deliberate kill chain: compromise Trivy (a widely-used security scanner) to steal CI/CD credentials, use those credentials to propagate a self-replicating npm worm, then backdoor both LiteLLM—a unified gateway to 100+ LLM APIs—and Telnyx SDK on PyPI.
- A novel persistence mechanism—a Python `.pth` file installed in site-packages—caused the LiteLLM payload to execute on every Python interpreter invocation, surviving package removal by standard package managers.
- TeamPCP pioneered the operational use of Internet Computer Protocol (ICP) blockchain infrastructure as command-and-control, making traditional domain-based takedown procedures largely ineffective.
- Organizations using any of the affected packages—Trivy, KICS GitHub Action, LiteLLM, or Telnyx SDK—should treat all CI/CD secrets, cloud credentials, and LLM API keys from the affected windows as fully compromised and rotate them immediately.

Background

TeamPCP (tracked also as DeadCatx3, PCPcat, ShellForce, and CanisterWorm) represents a convergence of financially motivated credential theft with what appears to be geopolitical targeting. The group's primary identifier derives from the dropper archive `tpcp.tar.gz`, which was recovered from multiple compromised CI/CD environments and bears a consistent RSA public key signature used for payload encryption across all campaign stages [1][2].

The campaign's infrastructure was established months before the operational phase. Beginning in December 2025, TeamPCP conducted mass exploitation of exposed Docker APIs, Kubernetes clusters, and Redis servers to build a staging environment capable of receiving and processing large-scale credential exfiltration [3]. The multi-month pre-positioning phase suggests a planned, methodical operational tempo rather than opportunistic exploitation. Infrastructure established in December 2025 was not activated until March 2026, a pattern more consistent with targeted campaign planning than reactive threat activity.

What makes this campaign analytically significant is not any single technique in isolation, but the architectural decision to chain multiple compromises sequentially—where each foothold yielded the specific credentials needed to compromise the next target. The selection of targets appears deliberate rather than incidental. Each compromised tool occupies a chokepoint in modern AI and cloud-native development pipelines, and the sequence—as observed in post-incident reconstruction—follows a logical progression to maximize credential harvest yield at each stage. This cascading structure compressed the attack window significantly: the full progression from the initial Trivy compromise to the LiteLLM PyPI backdoor took five days.

Security Analysis

The Cascading Kill Chain

The campaign opened on March 19, 2026, when TeamPCP force-pushed malicious commits to 76 of 77 version tags and 7 setup tags for `aquasec/trivy-action` and `setup-trivy` on GitHub, resulting in CVE-2026-33634 (CVSS 9.4) [1][4]. Trivy is a widely-deployed open-source container and infrastructure vulnerability scanner maintained by Aqua Security. Any CI/CD pipeline referencing these actions by unpinned tag—a practice that remains widespread despite guidance recommending SHA pinning—would execute the malicious code on the next run.

The Trivy payload's primary objective was credential harvesting, not sabotage. It scraped the memory of GitHub Actions runner processes (`Runner.Worker`, `Runner.Listener`) and swept 50+ hardcoded filesystem paths for credentials: AWS configuration files, Docker Hub authentication tokens, PyPI publishing tokens, SSH keys, shell history files, database connection strings, and cryptocurrency wallet files [4][5]. It also issued queries to the AWS Instance Metadata Service (IMDS) to collect cloud provider credentials from any runner executing on EC2 infrastructure. These stolen publishing tokens became the keys to the next phase.

On March 20, TeamPCP deployed CanisterWorm, a self-propagating npm worm that automated the token-to-compromise cycle [6]. Given a single stolen npm publishing token, CanisterWorm enumerated every package within that token's publishing scope, incremented version numbers, and inserted malicious code into new releases—completing this cycle in under 60 seconds per token. More than 50 npm packages were affected before automated registry monitoring flagged the activity. The following day, Aqua Security's own GitHub organization was defaced: 44 repositories were altered using credentials stolen from the `aqua-bot` service account, which had possessed elevated repository access [2][7].

On March 23, the stolen CI/CD credentials from Trivy victims gave TeamPCP write access to the Checkmarx KICS GitHub Action and the Checkmarx VS Code extensions on Open VSX Registry (approximately 36,000 combined downloads at the time of compromise) [8]. KICS (Keeping Infrastructure as Code Secure) is used by organizations to scan infrastructure-as-code templates for misconfigurations. Compromising it ensured that developers who explicitly ran security scanning workflows would execute the backdoor.

The most consequential stage followed on March 24, when LiteLLM versions 1.82.7 and 1.82.8 were published to PyPI. LiteLLM is a widely adopted open-source library that provides a unified API gateway to more than 100 LLM providers, including OpenAI, Anthropic, Cohere, Google, and others [9][10]. Any organization with LiteLLM in its Python environment—whether for production AI workloads, internal tooling, or developer experimentation—had configured API keys for all connected LLM providers accessible in memory and on disk. The payload exfiltrated all of them. The malicious versions were removed from PyPI approximately two to three hours after disclosure. LiteLLM's high aggregate download volume—approximately 95–97 million monthly downloads—reflects its broad adoption, but actual installations of the specific malicious versions (1.82.7 and 1.82.8) during that window represent a subset. Organizations should check their installed version explicitly rather than assuming exposure based on LiteLLM's general popularity [9].

On March 27, Telnyx SDK versions 4.87.1 and 4.87.2 were similarly backdoored on PyPI, with a payload variant that incorporated WAV-file steganography to deliver a Kubernetes wiper component [11]. A Kubernetes DaemonSet component named `host-provisioner-iran` was included in the Telnyx payload and was deployed against a subset of victims [11]. The naming convention suggests geopolitical targeting criteria may have been embedded in the payload logic, though whether affected victims were in fact Iranian-affiliated infrastructure has not been independently confirmed. This warrants monitoring but should be interpreted cautiously pending further attribution.

Novel Persistence and C2 Infrastructure

TeamPCP demonstrated deliberate attention to persistence and operational security. Rather than relying on the malicious package itself remaining installed, the LiteLLM payload wrote a `.pth` file—`litellm_init.pth`—into Python's site-packages directory [9]. Python processes `.pth` files at interpreter startup, unconditionally importing any module referenced within them. This means the payload would execute on every Python interpreter invocation in the environment, even after the malicious LiteLLM package had been removed by `pip uninstall`. Defenders who simply uninstalled the affected package and did not audit site-packages would remain persistently compromised.

For command-and-control, TeamPCP employed Internet Computer Protocol (ICP) blockchain canisters—decentralized, immutable smart contracts hosted on the ICP network [2][5]. This is the first documented operational use of ICP infrastructure for C2. Unlike traditional domain-based or IP-based C2, blockchain-hosted endpoints are resistant to conventional seizure, sinkholing, or registry-based takedown procedures. Traditional law enforcement tools are largely ineffective against this infrastructure; defenders must instead rely on network-level egress filtering targeting ICP replica endpoints and engage ICP governance mechanisms where possible. The beacon interval was configured at approximately 50 minutes [2][5], minimizing the frequency of outbound connections to avoid triggering volume-based anomaly detection. Complementary relay infrastructure used Cloudflare Tunnel endpoints—also resistant to simple IP blocking—alongside more traditional staging servers.

Exfiltrated data was encrypted with a hybrid AES-256-CBC and RSA-4096 scheme before transmission, with a 5-minute execution delay at payload initialization intended to evade sandbox analysis systems that timeout after 2–3 minutes [3]. The HTTP exfiltration channel used the identifying header `X-Filename: tpcp.tar.gz`, which provides a reliable network detection signature for retrospective traffic analysis.

Targeting the AI Developer Toolchain

The selection of LiteLLM as a primary target reflects its function as a credential aggregation point: rather than compromising individual LLM provider integrations one at a time, a single LiteLLM backdoor yields API keys for every provider an organization has connected. For enterprises running AI workloads in production, this means potential unauthorized access to OpenAI, Anthropic, and other provider accounts simultaneously—with associated cost implications from unauthorized usage, data exposure risk from API key theft enabling query replay, and supply chain trust erosion for downstream AI services [10].

The GitHub Actions compromise of both Trivy and KICS is similarly strategic. These are security tooling products—scanners whose explicit function is to detect vulnerabilities and misconfigurations. Organizations that treated the presence of these scanners in their pipelines as a security control were, for the duration of the campaign window, running a mechanism for credential theft in their CI/CD pipelines instead. The SANS Institute framed this dynamic in its post-campaign analysis as "when the security scanner became the weapon" [12].

A representative associated with TeamPCP claimed publicly that Anthropic's Claude was used to generate malware components and automate intrusion stages [3]. This claim has not been independently verified and warrants cautious interpretation; such assertions are sometimes made for reputational effect. However, it is consistent with the observed campaign sophistication and rapid operational tempo, and underscores the emerging risk of AI-assisted offensive tooling development.

Recommendations

Immediate Actions

Organizations should audit their exposure to the campaign before taking any other action. The following packages and versions were confirmed malicious and should be treated as indicators of active compromise, not merely vulnerable software:

- **Trivy Docker images**: `aquasec/trivy:0.69.4`, `0.69.5`, `0.69.6`
- **LiteLLM PyPI**: versions `1.82.7` and `1.82.8`
- **Telnyx SDK PyPI**: versions `4.87.1` and `4.87.2`
- **GitHub Actions**: `aquasec/trivy-action` or `aquasecurity/setup-trivy` pinned to any tag between March 19–22, 2026

Any environment that ran pipelines or installed packages from the above during the affected windows should be treated as fully compromised. All CI/CD secrets, cloud provider credentials, LLM API keys, Docker Hub tokens, PyPI tokens, and SSH keys accessible from those environments must be rotated immediately and without exception [1][4][8]. If LiteLLM was installed in any Python environment, administrators should inspect site-packages directories on all affected systems for the presence of `litellm_init.pth` and remove it—package uninstallation alone is insufficient [9].

Network telemetry from the affected windows should be reviewed for outbound connections to the known C2 domains and IPs documented in vendor advisories, and for HTTP traffic bearing the `X-Filename: tpcp.tar.gz` header [5].

The detection indicators from this campaign are consolidated in the table below as an operational reference for SOC and threat hunting teams:

Indicator Type	Value	Significance
HTTP header	`X-Filename: tpcp.tar.gz`	Payload exfiltration channel; check retrospective network logs
File artifact	`litellm_init.pth` in site-packages	Definitive persistence indicator; present even after package removal
Malicious package	LiteLLM 1.82.7, 1.82.8 (PyPI)	Backdoored LLM gateway; rotate all connected API keys
Malicious package	Telnyx SDK 4.87.1, 4.87.2 (PyPI)	Includes Kubernetes wiper component
Malicious Docker image	aquasec/trivy:0.69.4–0.69.6	Credential harvesting payload
GitHub Actions	aquasec/trivy-action or setup-trivy (tags, March 19–22)	Any pipeline run during this window should be treated as compromised
C2 infrastructure	ICP blockchain canisters; Cloudflare Tunnel endpoints	Not blockable via standard domain or IP blocking

Short-Term Mitigations

The Trivy campaign exploited the widespread practice of referencing GitHub Actions by mutable tag rather than by commit SHA. Tags are mutable references that repository owners—or attackers who have compromised repository access—can reassign to point to different commits. Pinning GitHub Actions to a specific commit SHA (e.g., `aquasec/trivy-action@abc1234`) eliminates this attack surface, because commit SHAs are immutable in git's content-addressing model [4][12]. Organizations should audit all GitHub Actions workflows and enforce SHA pinning as a policy, using tooling such as Dependabot or StepSecurity Harden-Runner to automate compliance.

PyPI package integrity verification should be implemented where possible. PyPI's implementation of PEP 740 Trusted Publishers and the emerging adoption of sigstore-based provenance attestations allow consumers to verify that a package release was produced by the expected build pipeline, providing a cryptographic link between the published artifact and the repository that produced it [13]. LiteLLM has since published provenance attestations; organizations should configure tooling to verify them.

Python environments hosting AI workloads or internal tooling should monitor for the creation of unexpected `.pth` files in site-packages directories that reference modules not associated with a known installed package. The specific indicator in this campaign—`litellm_init.pth`—should be treated as a definitive compromise indicator; any `.pth` file whose content references a module not corresponding to a currently installed package warrants investigation. Note that legitimate `.pth` files (e.g., those created by editable installs or namespace package tooling) will also be present and should be baselined before alerting.

Strategic Considerations

The TeamPCP campaign demonstrates that supply chain risk for AI systems extends substantially beyond model weights and training data—the developer toolchain itself is a critical attack surface. Organizations building on AI APIs and LLM-powered services should apply the same supply chain rigor to AI tooling dependencies that they apply to infrastructure dependencies. This includes software composition analysis (SCA) scanning of Python and npm environments with policies that fail builds on newly introduced or unvetted package versions, and isolation of AI workload environments so that LLM API credentials are scoped and do not coexist with CI/CD publishing credentials.

The use of ICP blockchain infrastructure for C2—described as the first documented operational use of this mechanism—may signal an emerging direction in threat actor infrastructure selection: decentralized or censorship-resistant channels that do not yield to conventional domain seizure or sinkholing. Defenders should anticipate that this technique, once demonstrated as viable, may be replicated by other actors. This means defenders should not rely solely on domain or IP blocklisting as a detection strategy and should invest in behavioral network analysis and DNS-over-HTTPS monitoring. The 50-minute beacon interval and 5-minute execution delay reflect awareness of standard detection mechanisms; defenders should extend behavioral analysis windows accordingly.

A threat researcher has reported alleged collaboration between TeamPCP and LAPSUS$, and multiple outlets have reported approximately 300 GB of credential exfiltration and active extortion activity following March 25 [3]. These figures originate from TeamPCP's own Telegram announcements and represent threat-actor self-reporting rather than independently verified measurements. Nonetheless, the scale of the claimed exfiltration and ongoing extortion activity suggests that the full downstream

impact of this campaign will unfold over months. Organizations should operate under the assumption that stolen credentials are being actively used and should monitor for unauthorized API usage across all affected LLM provider accounts.

CSA Resource Alignment

This campaign directly implicates several areas of the CSA AI Safety Initiative's published guidance and frameworks.

The MAESTRO threat modeling framework for agentic AI systems identifies supply chain compromise of AI tooling as a Tier 5 risk (infrastructure and integration layer), noting that agents and AI pipelines that rely on third-party libraries for LLM connectivity inherit the security posture of those dependencies [14]. The TeamPCP campaign is a concrete realization of this threat model: the LiteLLM backdoor would have affected any agentic AI system using LiteLLM as its LLM routing layer, potentially enabling credential theft that allowed adversaries to impersonate the agent or redirect its queries.

The CSA AI Organizational Responsibilities framework emphasizes that organizations deploying AI systems in production bear responsibility for the security of the full software supply chain supporting those systems—not merely the models themselves [15]. The practice of accepting new PyPI or npm package versions without integrity verification or provenance attestation represents a gap in that responsibility framework that TeamPCP exploited directly.

The CSA Cloud Controls Matrix (CCM) addresses software supply chain security under the Supply Chain Management and Transparency (STA) domain. CCM control STA-04 requires organizations to conduct due diligence on third-party software components and establish processes for monitoring for compromise. The TeamPCP campaign illustrates why passive monitoring is insufficient: proactive controls such as SHA pinning, attestation verification, and behavioral monitoring of newly installed packages are recommended complements [16].

CSA's Zero Trust guidance is directly applicable to the credential hygiene failures this campaign exploited. Limiting the scope and blast radius of CI/CD credentials—so that a CI runner's credentials cannot publish to PyPI, access LLM APIs, and modify Docker Hub repositories simultaneously—would have broken the cascading kill chain at multiple stages [17].

Organizations are encouraged to consult the following CSA resources:

- *MAESTRO: A Threat Modeling Framework for AI Systems* — for supply chain risk analysis in AI pipelines

- *AI Organizational Responsibilities* — for governance of AI software dependencies
- *Cloud Controls Matrix v4* — STA domain controls for supply chain management
- *Zero Trust Advancement Center* guidance — for credential scoping and least-privilege CI/CD design

---

References

[1] Arctic Wolf, "TeamPCP Supply Chain Attack Campaign Targets Trivy, Checkmarx (KICS), and LiteLLM," Arctic Wolf Blog, March 2026. https://arcticwolf.com/resources/blog/teampcp-supply-chain-attack-campaign-targets-trivy-checkmarx-kics-and-litellm-potential-downstream-impact-to-additional-projects/

[2] ReversingLabs, "Inside the TeamPCP Cascading Supply Chain Attack," ReversingLabs Blog, March 2026. https://www.reversinglabs.com/blog/teampcp-supply-chain-attack-spreads

[3] C. Stromblad, "Threat Assessment: TeamPCP — CanisterWorm & Kubernetes Wiper Campaign," cstromblad.com, March 2026. https://cstromblad.com/posts/threat-actor-profile-teampcp/

[4] Palo Alto Networks Unit 42, "When Security Scanners Become the Weapon: Breaking Down the Trivy Supply Chain Attack," Palo Alto Networks Blog, March 2026. https://www.paloaltonetworks.com/blog/cloud-security/trivy-supply-chain-attack/ *(URL returned HTTP 403 at time of publication; article title and claims corroborated by refs [1] and [5].)*

[5] Datadog Security Labs, "LiteLLM and Telnyx Compromised on PyPI: Tracing the TeamPCP Supply Chain Campaign," Datadog Security Labs, March 2026. https://securitylabs.datadoghq.com/articles/litellm-compromised-pypi-teampcp-supply-chain-campaign/

[6] Aikido Security, "TeamPCP Deploys Worm via npm Trivy Compromise," Aikido Security Blog, March 2026. https://www.aikido.dev/blog/teampcp-deploys-worm-npm-trivy-compromise

[7] Sysdig, "TeamPCP Expands: Supply Chain Compromise Spreads from Trivy to Checkmarx GitHub Actions," Sysdig Blog, March 2026. https://www.sysdig.com/blog/teampcp-expands-supply-chain-compromise-spreads-from-trivy-to-checkmarx-github-actions

[8] Mend.io, "CanisterWorm: The Self-Spreading npm Attack That Uses a Decentralized Server to Stay Alive," Mend.io Blog, March 2026. https://www.mend.io/blog/canisterworm-the-self-spreading-npm-attack-that-uses-a-decentralized-server-to-stay-alive/

[9] Trend Micro, "Your AI Gateway Was a Backdoor: Inside the LiteLLM Supply Chain Compromise," Trend Micro Research, March 2026. https://www.trendmicro.com/en_us/research/26/c/inside-litellm-supply-chain-compromise.html

[10] Endor Labs, "TeamPCP Isn't Done: Threat Actor Behind Trivy and KICS Compromises Now Hits LiteLLM," Endor Labs Blog, March 2026. https://www.endorlabs.com/learn/teampcp-isnt-done

[11] Help Net Security, "TeamPCP Strikes Again: Backdoored Telnyx PyPI Package Delivers Malware," Help Net Security, March 27, 2026. https://www.helpnetsecurity.com/2026/03/27/teampcp-telnyx-supply-chain-compromise/

[12] SANS Institute, "When the Security Scanner Became the Weapon: Inside the TeamPCP Supply Chain Campaign," SANS Blog, March 2026. https://www.sans.org/blog/when-security-scanner-became-weapon-inside-teampcp-supply-chain-campaign

[13] Python Packaging Authority, "PEP 740 – Index Support for Digital Attestations," PyPA, 2024. https://peps.python.org/pep-0740/

[14] Cloud Security Alliance, "MAESTRO: A Threat Modeling Framework for AI Systems," CSA AI Safety Initiative, 2025. https://cloudsecurityalliance.org/research/working-groups/artificial-intelligence/

[15] Cloud Security Alliance, "AI Organizational Responsibilities," CSA AI Safety Initiative, 2025. https://cloudsecurityalliance.org/research/working-groups/artificial-intelligence/

[16] Cloud Security Alliance, "Cloud Controls Matrix v4," CSA, 2024. https://cloudsecurityalliance.org/research/cloud-controls-matrix/

[17] Cloud Security Alliance, "Software-Defined Perimeter and Zero Trust," CSA, 2024. https://cloudsecurityalliance.org/research/working-groups/software-defined-perimeter/

