ultraviolet

THREAT ADVISORY

TeamPCP Trivy and LiteLLM Supply Chain Attacks

Services Performed By:
UltraViolet Cyber TIDE Team
tide@uvcyber.com

Published Date:
March 31, 2026
TLP:GREEN

Executive Snapshot

The recent TeamPCP campaign highlights how software supply chain attacks are increasingly aimed at the trusted systems organizations depend on to build, scan, and publish software. Rather than relying on a traditional intrusion path alone, the activity associated with TeamPCP appears to have leveraged compromised CI/CD and release mechanisms tied to the Trivy ecosystem, with downstream impact extending into packages such as LiteLLM. This makes the campaign especially important for Security Leadership because it demonstrates how attackers can abuse legitimate development and distribution channels to deliver malicious code, harvest credentials, and potentially pivot from build environments into broader enterprise infrastructure.

- Pin CI/CD dependencies and GitHub Actions to immutable commit hashes or verified internal mirrors, enforce artifact signing and provenance validation, and reduce dependence on mutable tags or live package retrieval during builds. This makes it harder for a threat actor like TeamPCP to weaponize trusted release paths and silently distribute malicious code through established workflows.

- Harden build environments and non-human identities by minimizing credential exposure, restricting token scope, segmenting CI infrastructure from production systems, and rapidly rotating secrets that may have been exposed. Campaigns like this are dangerous because the initial package or pipeline compromise can quickly become a credential theft and lateral movement event.

- Expand monitoring and incident response coverage to include the software delivery pipeline, with detections for suspicious outbound traffic, unexpected package behavior, abnormal startup execution, unauthorized service creation, and unusual Kubernetes or cloud activity originating from developer or build systems. The faster an organization can detect malicious behavior inside its software factory, the better its chances of preventing a supply chain compromise from becoming a wider breach.

TIDE Team Analysis

The TeamPCP campaign underscores how software supply chain attacks have evolved from isolated package tampering into broad compromises of the trust mechanisms that modern development depends on. Rather than targeting a single downstream application, the attackers appear to have moved through CI/CD workflows, release infrastructure, package publishing paths, and container distribution channels in a way that maximized downstream exposure. The central lesson is not simply that one tool or package was compromised, but that adversaries increasingly understand how to weaponize trusted automation at scale. For executive leadership, this incident serves as a reminder that development pipelines, security tooling, and release systems now sit squarely inside the enterprise attack surface and must be protected accordingly.

The Trivy portion of the incident is especially instructive because it demonstrates how attackers can exploit confidence in established tooling rather than relying on traditional malware delivery. By compromising release-related trust paths, the threat actor was able to influence artifacts and automation that many organizations consume as part of normal development and security operations. This model is far more dangerous than a one-off malicious package because it allows the attacker to inherit the legitimacy of the software delivery process itself. In practical terms, that means organizations can be exposed even when developers and engineers follow workflows that appear routine and approved.

The downstream LiteLLM compromise shows how a breach in one part of the software factory can quickly cascade into adjacent ecosystems. What began as a CI/CD and release trust issue appears to have propagated into Python package distribution, creating a path for malicious code execution in environments that installed affected versions. This illustrates the multiplier effect of software supply chain attacks: a compromise in a widely used upstream dependency or security tool can create exposure for countless downstream consumers without requiring the attacker to target each victim individually. Organizations should view this as a structural risk in modern development rather than an anomaly tied to one vendor or project.

The technical behavior reported in connection with the campaign reinforces that supply chain compromise often becomes a credential theft and infrastructure access problem almost immediately. The activity associated with the malicious packages reportedly focused on collecting cloud credentials, SSH material, environment secrets, Kubernetes data, and other sensitive configuration artifacts that are commonly present in development and build environments. That means the real impact may extend well beyond a compromised package installation and into lateral movement, persistence, or follow-on access to production systems. Security teams should therefore approach incidents like this with the assumption that code execution in a build or developer context can rapidly become an identity and cloud compromise event.

A major lesson from this incident is that trusted security tools are now prime targets for abuse because they often operate with elevated access and broad visibility across repositories, pipelines, and infrastructure. Many organizations implicitly trust scanners, CI helpers, deployment actions, and package feeds because they are part of the defensive or operational baseline. That trust can become a weakness when those tools are allowed to update dynamically, run with excessive permissions, or retrieve mutable content during builds. The lesson for security leadership is that defensive tooling must be held to the same, or higher, integrity standards as production software because compromise at that layer can silently affect large portions of the environment.

One of the clearest preventative measures is to reduce dependence on mutable trust anchors in CI/CD. Organizations should pin GitHub Actions and similar dependencies to immutable commit hashes, enforce deterministic builds wherever possible, and route package retrieval through controlled internal repositories rather than allowing unrestricted build-time pulls from public sources. Artifact signing, provenance verification, and strong separation of release credentials from day-to-day maintainer workflows should become standard practice. These controls do not eliminate supply chain risk, but they make it substantially harder for attackers to convert a credential compromise into a broadly distributed malicious release.

Organizations should also strengthen runtime and detection controls around their software factory environments. Build systems, developer workstations, and containerized pipelines should be monitored for unusual outbound connections, unexpected system service creation, suspicious Python startup behavior, abnormal Kubernetes pod deployment activity, and access to secrets stores outside of normal workflow patterns. Secrets management should be tightened so CI jobs and developer environments expose only the minimum credentials necessary for the task at hand, and all high-value credentials accessible from affected systems should be assumed compromised during response. Limiting blast radius is critical because the primary value of these attacks often lies in what the attacker can steal or pivot to after initial execution.

The broader lesson from the Trivy and LiteLLM compromises is that software supply chain defense must be treated as an enterprise resilience issue rather than a narrow developer hygiene topic. Security, engineering, platform, and incident response teams need shared ownership of build integrity, non-human identities, release trust, and dependency governance. When organizations assume that upstream tools, actions, and packages are inherently trustworthy, they create the conditions for a single compromise to ripple through development and production at scale. The path forward is to build software delivery pipelines that are verifiable, constrained, and continuously monitored so that compromise of one trusted component does not become a compromise of the enterprise.

Why It Matters

CI/CD and software supply chain attacks matter because they allow adversaries to compromise the systems that organizations use to build, sign, test, and deliver trusted software, which creates a multiplier effect far beyond a single host or application. Recent Google Cloud threat reporting found that third-party and software supply chain compromises continue to appear in real intrusions, including cases where compromised packages were used to harvest credentials and establish persistence in CI/CD environments. That is the strategic danger of campaigns like TeamPCP: once attackers gain access to the software factory, they can inherit trust, reach downstream customers, and move from development tooling into cloud infrastructure with much less friction than in a conventional intrusion.

The operational impact is large because CI/CD systems are often deeply connected to source repositories, container registries, cloud roles, secrets stores, and production deployment paths. Google Cloud described a 2025 incident in which a compromised NPM package led to theft of a developer GitHub token and then abuse of GitHub-to-cloud OIDC trust, resulting in full cloud compromise within 72 hours. That kind of progression shows why supply chain attacks are so disruptive: they do not just poison code, they can collapse identity boundaries between development and production and give attackers a fast path to privileged access, data exposure, and destructive activity.

This matters for security leadership because the problem is no longer limited to package hygiene or developer best practices; it is now an enterprise resilience issue tied to identity, release integrity, and cloud containment. Public guidance increasingly emphasizes stronger controls around CI/CD configuration changes, MFA for infrastructure and repository access, image signing, provenance generation, admission controls, and monitoring for bulk secret access or unauthorized pipeline modification. Organizations that fail to harden these trust paths risk allowing a single upstream compromise to propagate across internal development, customer-facing software, and production cloud environments at the same time.

How to Respond

- Strictly adhere to cybersecurity fundamentals and ensure all personnel undergo annual phishing and social engineering training. Speak with your UltraViolet Cyber TAM Representative to schedule a live phishing engagement.
- Validate and inventory the use of Trivy and LiteLLM in your environment.
- Perform annual tech refresh reviews to gain a holistic understanding of your infrastructure. Speak with your UltraViolet Cyber TAM Representative to schedule a RedTeam or PurpleTeam engagement to gain insight into the vulnerabilities in your environment.

What UltraViolet Cyber is Doing

- Proactively enabling custom detections based on the collected artifacts, tactics, techniques, and procedures identified in this activity.
- Performing hypothesis driven threat hunts based on threat actor behavior and artifacts. UVCyber customers will be informed of the results through secure channels.
- Parsing available victim dump data for any social, financial, business, or technical relations to UVCyber Clients and partner organizations.
- Aggregating threat intelligence from myriad sources and applying the most up-to-date knowledge to proactive threat hunting and response.

About UltraViolet Cyber

UltraViolet Cyber is a leading tech-enabled managed security services provider, delivering unparalleled cybersecurity expertise that fills technology and talent gaps across Global 2000 and Federal Government customers. Founded and operated by security practitioners from the national intelligence community, UltraViolet Cyber connects offensive security, application security, detection and response, and security engineering to deliver a differentiated approach to cybersecurity operations. Transforming customers' security programs, UltraViolet Cyber's flagship security-as-a-service solution, UV Lens, removes complex operational silos, replacing them with integrated security capabilities. UltraViolet is headquartered in McLean, Virginia with technology centers across the world.

443.351.7630 / info@uvcyber.com | UltraViolet Cyber | @uv_cyber

---