WHITE PAPER

CYTEX™
AI-Powered Resilience Command Center

TeamPCP Backdoors LiteLLM via Trivy CI/CD Compromise

Supply Chain Attack

hello@cytex.io | https://cytex.io

TeamPCP Backdoors LiteLLM

Supply Chain Attack

The threat actor behind recent compromises of Trivy and KICS has struck again, pushing two malicious versions of the popular Python package LiteLLM. The LiteLLM backdoored packages have since been removed from PyPI, but not before they could be deployed across countless environments.

Three-Stage Payload

Stage 1: Credential Harvesting

- Sweeps SSH keys, cloud credentials, Kubernetes secrets, cryptocurrency wallets, and .env files.
- Compresses stolen data into an encrypted archive named tpcp.tar.gz.
- Exfiltrates via HTTPS POST to models.litellm[.]cloud.

Stage 2: Kubernetes Lateral Movement

- Leverages the Kubernetes service account token (if present) to enumerate all nodes in the cluster.
- Deploys a privileged pod to each node.
- The pod chroots into the host file system and installs the persistence dropper on every node.

Stage 3: Persistent Systemd Backdoor

- Installs a systemd service named sysmon[.]service.
- Polls checkmarx[.]zone/raw every 50 minutes for next-stage payloads.
- If the fetched URL contains youtube[.]com, the script aborts—a kill switch pattern consistent with previous TeamPCP incidents.

How was the Code Injected?

Version 1.82.7

- Malicious code embedded in litellm/proxy/proxy_server[.]py.
- Injected during or after the wheel build process.
- Executes at module import time—any process importing the module triggers the payload without user interaction.

Version 1.82.8 – More Dangerous

- Adds a malicious litellm_init[.]pth file at the wheel root.
- .pth files in site-packages are processed automatically at Python interpreter startup.
- The file spawns a child Python process via subprocess.Popen, running the payload in the background.
- The base64-encoded payload decodes to an orchestrator that unpacks the credential harvester and persistence dropper.

The Compromise Chain

TeamPCP appears to have gained initial access through LiteLLM's CI/CD pipeline, which reportedly used Trivy, a tool the same actor compromised earlier. This created a cascading effect:

- Trivy compromised → credentials stolen → access to LiteLLM CI/CD.
- LiteLLM backdoored → credentials harvested from downstream users. → Those credentials become the keys to the next target.

The pattern is clear, CI/CD pipeline compromise leads to package backdoor. Package backdoor leads to credential theft. Credentials lead to the next environment. The supply chain is eating itself.

In a Telegram message, TeamPCP claimed responsibility and signaled escalation:

ʃThese.companies.were.built.to.protect.your.supply.chains.yet.they.can't.even.protect. their.own?.the.state.of.modern.security.research.is.a.joke?.as.a.result.we're.gonna.be. around.for.a.long.time.stealing.terrabytes.of.trade.secrets.with.our.new.partners¡.The. snowball.effect.from.this.will.be.massive?.we.are.already.partnering.with.other.teams. to.perpetuate.the.chaosʃ

TeamPCP compromised five ecosystems: **GitHub Actions, Docker Hub, npm, Open VSX, PyPI.**

The LiteLLM compromise is a part of a sustained, coordinated campaign targeting high-leverage points in the software supply chain, security tools, developer infrastructure, and widely used packages. Each compromised environment yields credentials that unlock the next target.

Mitigations

- Rotate ALL credentials present as environment variables or config files on any system where LiteLLM versions 1.82.7 or 1.82.8 were installed.
- Audit environments for these versions. Revert to a clean version immediately.
- Isolate affected hosts.
- Check Kubernetes clusters for rogue pods.
- Review network logs for egress traffic to models.litellm[.]cloud and checkmarx[.]zone.
- Remove persistence mechanisms (systemd service sysmon[.]service).
- Audit CI/CD pipelines for use of Trivy or KICS during the compromise windows.

Indicators of Compromise (IOCs)

IoC	Type	Status
litellm==1.82.7	PyPI Package	Removed from PyPI
litellm==1.82.8	PyPI Package	Removed from PyPI
8395c3268d5c5dbae1c7c6d4bb3c318c752ba4608cfcd90eb97ffb94a910eac2	SHA-256 (1.82.7 wheel)	Active IoC
d2a0d5f564628773b6af7b9c11f6b86531a875bd2d186d7081ab62748a800ebb	SHA-256 (1.82.8 wheel)	Active IoC
a0d229be8efcb2f9135e2ad55ba275b76ddcfeb55fa4370e0a522a5bdee0120b	SHA-256 (compromised proxy_server.py)	Active IoC
71e35aef03099cd1f2d6446734273025a163597de93912df321ef118bf135238	SHA-256 (litellm_init.pth, 1.82.8 only)	Active IoC
models.litellm.cloud	C2 Domain (exfiltration)	Active
checkmarx.zone	C2 Domain (persistence)	Active
checkmarx.zone/raw	C2 Endpoint (payload delivery)	Active
~/.config/sysmon/sysmon.py	Filesystem (persistence script)	Active IoC
~/.config/systemd/user/sysmon.service	Filesystem (systemd unit)	Active IoC
/tmp/pglog	Filesystem (downloaded binary)	Active IoC
/tmp/.pg_state	Filesystem (state tracking)	Active IoC
node-setup-* pods in kube-system	Kubernetes (attacker pods)	Active IoC
tpcp.tar.gz	Exfiltration archive name	Active IoC
X-Filename: tpcp.tar.gz	HTTP header (exfiltration POST)	Active IoC
litellm_init.pth	Filesystem (.pth payload, 1.82.8 only)	Active IoC

We are stuck in a loop, and TeamPCP has made it clear they intend to keep it spinning.