LiteLLM Python Package Compromised in Supply Chain Attack



LiteLLM version 1.107.0, distributed via PyPI wheels and Docker images, was reported compromised and delivered a credential stealer and a backdoor. PyPI has quarantined the package.

- The compromised release is LiteLLM 1.107.0.
- It was distributed through PyPI wheels and Docker images.
- The payload included a credential stealer and a backdoor.
- SafeDep, Socket, StepSecurity and Upwind reported the incident.
- The package has been quarantined on PyPI.

What was reported

Security researchers at SafeDep, Socket, StepSecurity and Upwind report that the widely used LiteLLM Python package was compromised in a software supply chain attack. The affected release is version 1.107.0, which was distributed both as PyPI wheels and as Docker images. According to the reports, it delivered a credential stealer together with a backdoor.

The package has since been quarantined on the Python Package Index (PyPI). The source excerpt does not provide further technical detail, such as how the compromise occurred, the full scope of exposure or the data targeted.

Why it matters

A credential stealer and a backdoor in a dependency can give an attacker access to secrets held in the environments where the package runs, and a persistent foothold. Because the release reached users through both package and container channels, organisations should check both their application dependencies and their built images.

Recommended approach

Treat any environment that installed or ran version 1.107.0 as potentially exposed until it is verified otherwise. Prioritise confirming where the version is present, then contain and rotate credentials as a precaution. Follow the vendor and researcher advisories for indicators and fixed versions, as the available reporting is limited.

What to do now

1. Search code repositories, lockfiles, CI pipelines and container registries for LiteLLM 1.107.0.
2. Isolate any host or container that installed or ran that version and investigate it for compromise.
3. Rotate API keys, tokens and other credentials accessible from affected environments.
4. Pin dependency versions and use an internal package mirror with review before new releases are adopted.
5. Monitor the researchers' advisories for indicators of compromise and fixed versions, and record the incident under your third-party software risk process.