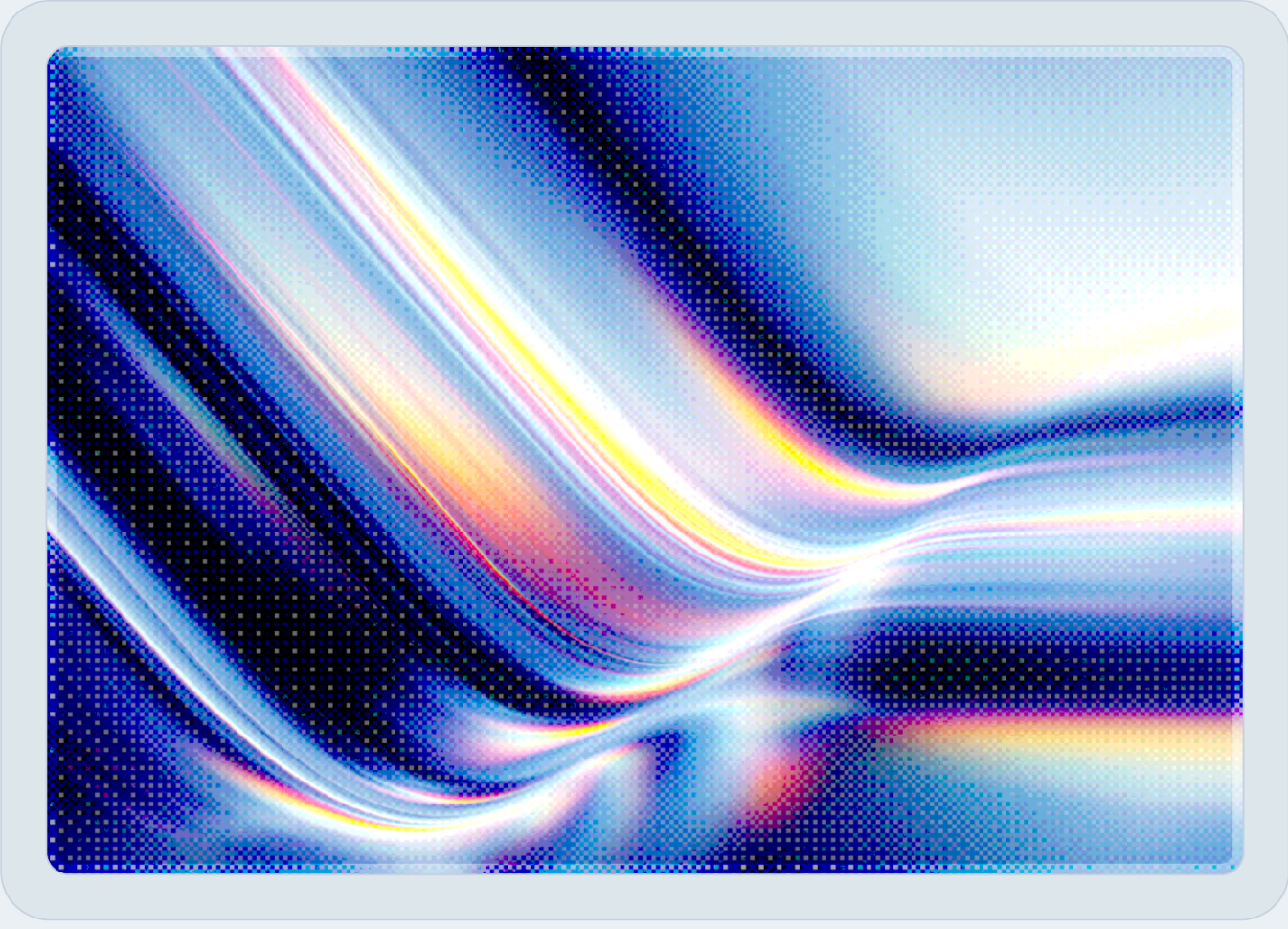
CSAI Foundation | Cloud Security Alliance

LiteLLM AI Gateway: Critical Vulnerability Chain Exposes API Keys

A Cascading Series of Supply Chain, Injection, and RCE Vulnerabilities Targets Centralized AI Credential Stores

2026-06-16

AI-assisted Rapid Research



CSAI

CSA cloud security alliance®

Key Takeaways

- LiteLLM, an open-source AI gateway proxy with 95 million monthly PyPI downloads [1], has been struck by a cascading series of critical vulnerabilities in 2026 that collectively expose the enterprise API keys it is designed to manage.
- A supply chain compromise in March 2026 (CVE-2026-33634, CVSS 9.4) resulted in malicious LiteLLM packages being published to PyPI, with payloads designed to harvest AI provider credentials and spread laterally into Kubernetes infrastructure [2].
- A pre-authentication SQL injection vulnerability disclosed in April 2026 (CVE-2026-42208, CVSS 9.3) allows any unauthenticated attacker who can reach a LiteLLM proxy port to extract all stored API keys and provider credentials from its PostgreSQL backend [3].
- A command injection flaw (CVE-2026-42271) in LiteLLM's MCP server test endpoints, when chained with a Starlette host header bypass (CVE-2026-48710), achieves unauthenticated remote code execution; Horizon3.ai assessed the chained exploit as a CVSS 10.0 critical-severity scenario [4]. CISA added CVE-2026-42271 to its Known Exploited Vulnerabilities catalog on June 8, 2026, with a federal remediation deadline of June 22, 2026 [5][17].
- Organizations that ran any affected LiteLLM version from an internet-accessible host should treat all API keys, cloud credentials, and CI/CD tokens as potentially compromised and rotate them as a precautionary measure; absence of observed indicators does not rule out credential extraction, as harvesting operations leave minimal forensic trace. Upgrading to v1.83.10-stable is the required remediation.

Background

LiteLLM is an open-source Python library and proxy server that provides a unified, OpenAI-compatible HTTP interface across more than one hundred large language model providers, including OpenAI, Anthropic, Google Gemini, Amazon Bedrock, Azure OpenAI, Mistral, and Hugging Face [1]. The project, maintained by BerriAI, has accumulated more than 50,000 GitHub stars and records approximately 95 million monthly downloads on PyPI, reflecting widespread adoption across the AI developer ecosystem [1][2].

For enterprise security purposes, the highest-risk deployment pattern is proxy mode: organizations run LiteLLM as a centralized gateway service, route all AI traffic through it, and store every upstream provider credential in its PostgreSQL-backed secret store. This architecture provides real operational benefits—centralized rate limiting, per-team virtual keys, budget enforcement, and unified audit logging—but it also creates a single high-value credential repository. A successful attack on a centralized LiteLLM proxy does not yield one provider's API key; it yields every provider's API key simultaneously, along with cloud credentials and any secrets stored in its configuration.

The series of vulnerabilities disclosed and exploited between March and June 2026 has demonstrated precisely this risk. Attackers targeting LiteLLM are not interested in any single AI provider account. They are targeting the aggregated credential value that the gateway's design inherently concentrates.

Security Analysis

The Supply Chain Compromise: CVE-2026-33634

The first major incident of 2026 was a supply chain compromise attributed to the threat actor group TeamPCP. The group had been conducting a coordinated campaign against the developer toolchain ecosystem beginning in mid-March: on March 19, they compromised the Trivy open-source vulnerability scanner; on March 21, they hijacked the Checkmarx KICS GitHub Action [6][18]. LiteLLM's CI/CD pipeline depended on Trivy for security scanning, and when the compromised Trivy tooling executed within that pipeline, it exfiltrated the project's PyPI publishing tokens to the attacker's infrastructure [7].

Armed with LiteLLM's PyPI publishing credentials, TeamPCP published two malicious versions—v1.82.7 and v1.82.8—on March 24, 2026 [2]. PyPI quarantined the packages within roughly three hours [6], but during that window the packages accumulated tens of thousands of downloads [2]. CVE-2026-33634 received a CVSS 4.0 score of 9.4 [2].

The malicious payload was a multi-stage attack. The first stage harvested credentials from the host environment: AI provider API keys, cloud platform credentials from AWS, GCP, and Azure metadata services, SSH private keys, and CI/CD pipeline tokens. The second stage attempted lateral movement into Kubernetes clusters by enumerating the service account tokens accessible to the compromised pod. The third stage installed a persistent systemd backdoor that polled for additional payloads over an encrypted channel, ensuring attacker persistence even if the malicious package was later removed [7][8]. All harvested credentials were encrypted before exfiltration.

The Trivy-to-LiteLLM attack path illustrates a compounding risk visible across multiple recent AI toolchain incidents: security tooling that runs with elevated CI/CD permissions is itself a high-value target, because its compromise grants implicit trust from downstream projects that depend on it.

Pre-Authentication SQL Injection: CVE-2026-42208

Disclosed in April 2026, CVE-2026-42208 is a critical pre-authentication SQL injection vulnerability affecting LiteLLM versions 1.81.16 through 1.83.6 [3]. The flaw resides in the authentication middleware that processes the `Authorization: Bearer` header. LiteLLM passes the bearer token value directly into a SQL query without parameterization to look up the corresponding virtual key record. Because this lookup occurs before the authentication decision is made—before the system determines whether a request is authorized—any HTTP client that can reach the proxy port can inject arbitrary SQL statements into the PostgreSQL backend without presenting any credentials [3].

Three tables contain the highest-value data exposed by this vulnerability. The `LiteLLM_VerificationToken` table stores all virtual API keys including the master key. The `litellm_credentials` table holds stored provider credentials for upstream AI services. The `litellm_config` table contains proxy environment variables, which frequently include additional secrets and configuration values [3]. An attacker who can execute arbitrary `SELECT` statements against these tables effectively retrieves the entire credential inventory of the gateway.

Sysdig Threat Research documented the first confirmed in-the-wild exploitation attempts within 36 hours of CVE-2026-42208 being indexed in the GitHub Advisory Database [16]. The 36-hour exploitation window suggests that threat actors were actively monitoring the advisory database for new LiteLLM vulnerabilities, consistent with awareness of the credential value concentrated in gateway deployments [16]. BerriAI released a patch in v1.83.7-stable and recommends that any organization that ran an affected version with the proxy reachable from an untrusted network review their PostgreSQL query logs for evidence of exploitation [9].

Command Injection Chained to Unauthenticated RCE: CVE-2026-42271 and CVE-2026-48710

The most recently exploited vulnerability in this series, CVE-2026-42271, is a command injection flaw introduced alongside LiteLLM's MCP (Model Context Protocol) server integration [4]. Two endpoints—`POST /mcp-rest/test/connection` and `POST /mcp-rest/test/tools/list`—were designed to let users preview an MCP server configuration before saving it. Both endpoints accepted a complete server configuration object in the request body, including the `command`, `args`, and `env`

fields used by the stdio transport. LiteLLM passed this input directly to a subprocess invocation on the proxy host with no validation or sandboxing, granting the subprocess the full privileges of the LiteLLM process [4].

Standing alone, CVE-2026-42271 carries a CVSS score of 8.7 and requires authentication. Horizon3.ai researchers identified that chaining it with CVE-2026-48710—a host header validation bypass in the Starlette web framework underlying LiteLLM—allows an attacker to sidestep the authentication check entirely [4]. The combined exploit chain achieves unauthenticated remote code execution against any internet-exposed LiteLLM proxy; Horizon3.ai assessed the chained scenario as CVSS 10.0 critical-severity, reflecting that authentication is fully bypassed [4].

The practical consequence of successful exploitation mirrors the SQL injection scenario but with far greater attacker control. An adversary achieving RCE on the proxy host can read all secrets from the process environment, query the PostgreSQL database directly, install persistent backdoors, and pivot to connected AI infrastructure. CISA added CVE-2026-42271 to its Known Exploited Vulnerabilities catalog on June 8, 2026, and directed Federal Civilian Executive Branch agencies to remediate by June 22, 2026 under Binding Operational Directive 22-01 [5][17].

The Systemic Pattern

Viewed together, these vulnerabilities reveal a systemic pattern rather than isolated bugs. The supply chain compromise demonstrated that AI gateway projects are high-priority targets for credential harvesting at scale. The SQL injection and RCE chain confirmed that attackers are actively targeting LiteLLM's credential store through any available exploit path. An earlier vulnerability, CVE-2025-0330, had already shown that LiteLLM's error-handling code could leak third-party API keys (specifically Langfuse keys) due to insufficient secret isolation in team-settings parsing [10]. The repository now carries eight official GitHub Security Advisories covering three critical and five high-severity findings across 2024, 2025, and 2026 [10].

The core architectural tension is this: a gateway that aggregates credentials from dozens of AI providers creates a high-value, concentrated target, and that target value scales with each additional provider integrated. Every organization that uses LiteLLM in proxy mode should treat the security posture of that proxy as equivalent to the combined security requirements of every AI provider account it manages.

Recommendations

Immediate Actions

Organizations running LiteLLM should upgrade to v1.83.10-stable without delay and simultaneously update Starlette to version 1.0.1 or later to close the host header bypass component of the RCE chain [4] [9]. This single upgrade addresses CVE-2026-42208, CVE-2026-42271, and CVE-2026-48710. For teams that cannot immediately complete an upgrade cycle, blocking network access to the `/mcp-rest/test/` endpoint path at an API gateway, reverse proxy, or network perimeter firewall provides a partial compensating control for the RCE chain, though it does not address the SQL injection [5].

Organizations that installed LiteLLM versions 1.82.7 or 1.82.8 should treat all credentials accessible to the affected host as potentially compromised and rotate them as a precautionary measure; absence of observed indicators does not rule out credential extraction, as harvesting operations leave minimal forensic trace. This includes AI provider API keys, cloud platform credentials, SSH private keys, and CI/CD tokens. Any Kubernetes service accounts accessible from the affected host should be audited for unauthorized activity [2][7].

Organizations that ran any version from 1.81.16 through 1.83.6 with the proxy accessible from an untrusted network should review PostgreSQL query logs for evidence of SQL injection exploitation, specifically looking for anomalous bearer token values in authentication queries [9].

Short-Term Mitigations

Access control hardening is a critical defensive measure that reduces exposure to the full vulnerability class, independent of any specific patch cycle. LiteLLM's proxy port should never be exposed directly to the public internet; it should be reachable only from authenticated internal clients or through an authenticated API gateway that enforces its own access controls. Network segmentation that isolates the LiteLLM host and its PostgreSQL instance from general corporate infrastructure limits the lateral movement potential of any future compromise [3][4].

Dependency verification should be integrated into all pipelines that install LiteLLM or any of its transitive dependencies. Teams should verify package hashes against known-good values before installation and monitor PyPI for any unexpected new package versions. The supply chain compromise was contained relatively quickly by PyPI's response, but the multi-hour window and resulting download volume demonstrate that passive reliance on registry safety controls is insufficient [6][7].

Secret management practices for AI gateway deployments warrant immediate review. Provider API keys stored in LiteLLM's credential store should be scoped to the minimum permissions required for the use case—for example, read-only inference keys where write access to provider account settings is not needed. Organizations using secrets management platforms such as HashiCorp Vault or AWS Secrets Manager should explore whether provider credentials can be injected at runtime rather than persisted in the gateway database, reducing the value of a database-level credential extraction [3].

Strategic Considerations

The LiteLLM incident series highlights that enterprise AI gateway architecture requires the same defense-in-depth thinking applied to other high-value credential aggregation points such as identity providers and secrets managers. Organizations standardizing on a centralized AI gateway should evaluate whether their chosen solution undergoes third-party security audits, maintains a responsible disclosure program, and publishes timely security advisories. The availability of a comprehensive CVE history for a security-sensitive project is a positive indicator of transparency, not a disqualifying factor—but the tempo of critical findings in 2026 warrants scrutiny of the development and release process.

Dependency minimization and supply chain governance for AI infrastructure components should be treated as first-class security requirements. The Trivy compromise that enabled the LiteLLM supply chain attack succeeded because a security tool ran with privileges that allowed credential exfiltration. Build pipelines that handle secrets should run with least-privilege service accounts and should treat all third-party tools—including security scanners—as potentially untrusted inputs.

From a broader risk perspective, AI gateway components are functioning as a new category of privileged infrastructure. Their compromise yields credentials for AI capabilities that are increasingly integrated with sensitive business workflows. Security teams should include AI gateway deployments explicitly in their threat models and incident response playbooks rather than treating AI infrastructure as a developer-managed concern outside the security perimeter.

CSA Resource Alignment

CSA's LLM Threats Taxonomy [11] provides a classification framework directly applicable to the vulnerabilities described in this note. The supply chain compromise aligns with the taxonomy's treatment of training and serving pipeline threats, while the SQL injection and RCE exploits map to inference serving and API endpoint attack categories. Organizations using the taxonomy for threat modeling should consider AI gateway components as in-scope for the serving infrastructure threat surface.

CSA's guidance on Securing LLM Backed Systems: Essential Authorization Practices [12] addresses the authorization design principles most relevant to preventing credential aggregation risks. The document's recommendations on virtual key scoping, per-team access controls, and audit logging are directly applicable to LiteLLM's proxy deployment model and should be reviewed against current gateway configurations.

CSA's Zero Trust for LLM Environments guidance [13] provides architectural principles applicable to hardening AI gateway deployments. The Zero Trust model's requirement that no network location be implicitly trusted aligns with the recommendation that LiteLLM proxy ports be isolated behind authenticated access controls rather than exposed on internal networks that may be reachable from compromised endpoints.

The CSA AI Controls Matrix (AICM) [14], as a superset of the Cloud Controls Matrix, provides the control structure most directly applicable to AI infrastructure governance. Controls in the AI Security domain covering supply chain integrity, secrets management, and API access controls are all engaged by the LiteLLM incident series and should be used to structure an organizational assessment of AI gateway security posture.

Finally, CSA's Agentic AI Red Teaming Guide [15] is particularly relevant for organizations using LiteLLM to power multi-agent or agentic AI workflows, where a compromised gateway could affect not only credential security but also the integrity of agent reasoning and action pipelines. Red teaming exercises for agentic deployments should explicitly target the gateway layer.

References

[1] BerriAI. "LiteLLM GitHub Repository." GitHub, 2026.

[2] Bitsight. "Major Security Event: Supply Chain Compromise in LiteLLM Versions 1.82.7 and 1.82.8." Bitsight Blog, March 2026.

[3] Security Boulevard. "CVE-2026-42208: Pre-Authentication SQL Injection in LiteLLM Exposes API Credentials." Security Boulevard, May 2026.

[4] Horizon3.ai. "CVE-2026-42271: LiteLLM Unauthenticated RCE." Horizon3.ai Attack Research, June 2026.

[5] CISA. "CISA Adds Two Known Exploited Vulnerabilities to Catalog." CISA Alerts, June 8, 2026.

[6] Snyk. "How a Poisoned Security Scanner Became the Key to Backdooring LiteLLM." Snyk Blog, March 2026.

[7] Datadog Security Labs. "LiteLLM and Telnyx compromised on PyPI: Tracing the TeamPCP supply chain campaign." Datadog Security Labs, March 2026.

[8] Trend Micro. "Your AI Gateway Was a Backdoor: Inside the LiteLLM Supply Chain Compromise." Trend Micro Research, March 2026.

[9] BerriAI. "Security Update: CVE-2026-42208 in LiteLLM Proxy." LiteLLM Documentation Blog, April 2026.

[10] Vulert. "CVE-2025-0330: LiteLLM API Key Leakage Vulnerability." Vulert Vulnerability Database, 2025.

[11] Cloud Security Alliance. "Large Language Model (LLM) Threats Taxonomy." Cloud Security Alliance, 2024.

[12] Cloud Security Alliance. "Securing LLM Backed Systems: Essential Authorization Practices." Cloud Security Alliance, 2024.

[13] Cloud Security Alliance. "Using Zero Trust to Secure Enterprise Information in LLM Environments." Cloud Security Alliance, 2026.

[14] Cloud Security Alliance. "AI Controls Matrix." Cloud Security Alliance, 2024.

[15] Cloud Security Alliance. "Agentic AI Red Teaming Guide." Cloud Security Alliance, 2025.

[16] Sysdig. "CVE-2026-42208: Targeted SQL injection against LiteLLM's authentication path discovered 36 hours following vulnerability disclosure." Sysdig Threat Research, May 2026.

[17] Help Net Security. "LiteLLM vulnerability under active attack, CISA warns (CVE-2026-42271)." Help Net Security, June 9, 2026.

[18] ReversingLabs. "Inside the TeamPCP cascading supply chain attack." ReversingLabs Blog, March 2026.