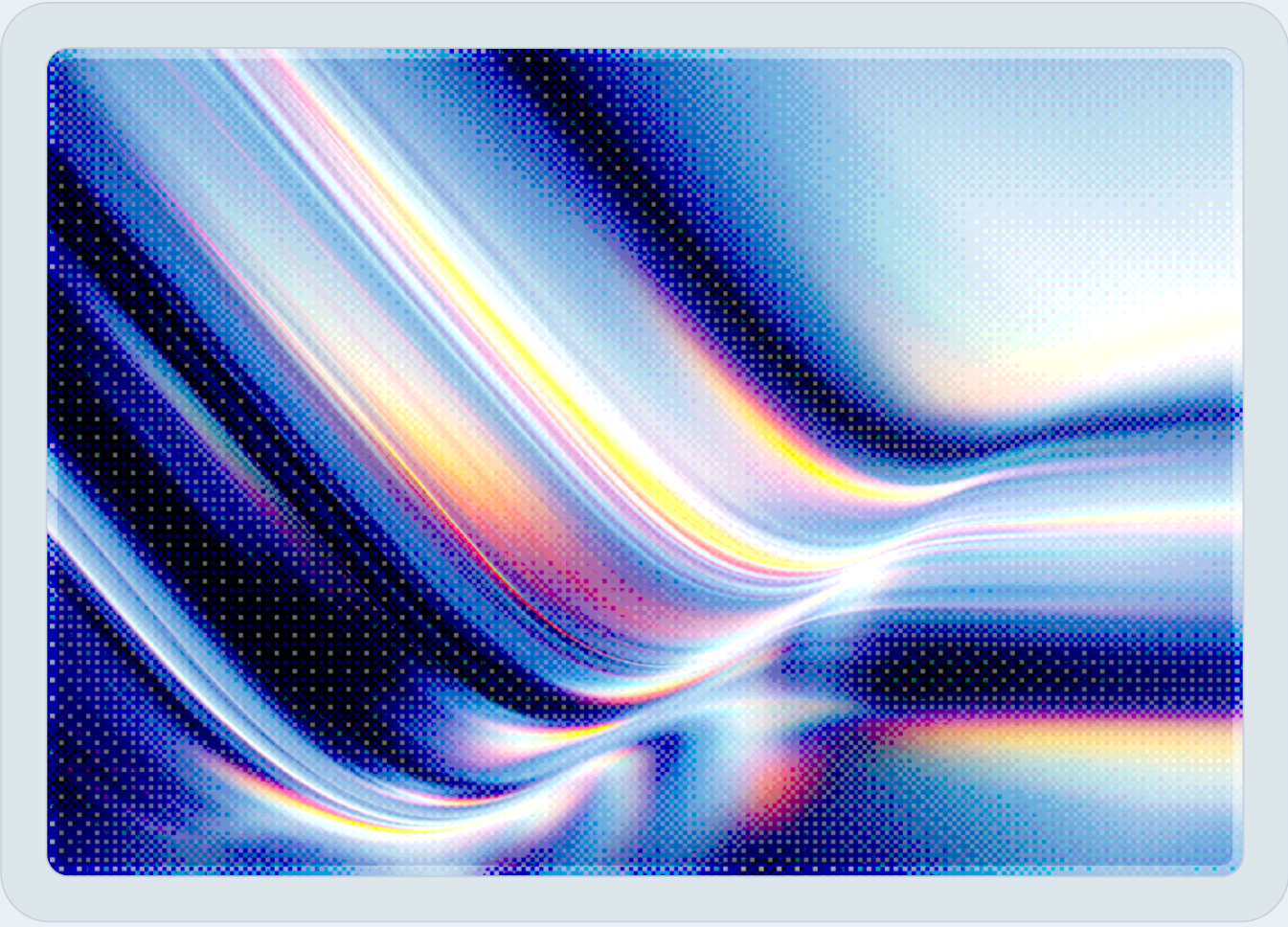
CSAI Foundation | Cloud Security Alliance

LiteLLM CVE-2026-42208: Pre-Auth SQL Injection in AI Proxy

Exploitation Within 36 Hours Exposes AI Provider Credential Stores

2026-04-30

AI-assisted Rapid Research



CSAI | cloud security alliance®

Key Takeaways

- CVE-2026-42208 is a critical (CVSS 9.3) pre-authentication SQL injection vulnerability in LiteLLM — a widely deployed open-source AI proxy that routes API traffic to over 100 LLM providers — affecting versions 1.81.16 through 1.83.6, with a patch available in version 1.83.7-stable [1].
- Unauthenticated attackers can inject arbitrary SQL via a crafted `Authorization: Bearer` header on any LLM API route, reaching the proxy's PostgreSQL backend without presenting any credentials, because LiteLLM concatenates the raw token value directly into SQL rather than using parameterized queries [2].
- A threat actor began active exploitation within 36 hours of the advisory being indexed in the public GitHub Advisory Database, using 17 distinct UNION-based payloads to enumerate the database schema and immediately targeting the three tables that hold the highest-value assets: `LiteLLM_VerificationToken`, `litellm_credentials`, and `litellm_config` [3].
- For organizations that store organization-level or account-root provider credentials in LiteLLM — rather than project-scoped application keys — successful exploitation exposes the entire credential store, which typically contains upstream LLM provider API keys for OpenAI, Anthropic, and AWS Bedrock simultaneously, in addition to virtual and master proxy keys. In such deployments, a single compromised instance is operationally comparable to a full cloud account compromise [2].
- Organizations running internet-facing LiteLLM instances should update to version 1.83.7 immediately and rotate all provider credentials stored in the affected deployment, regardless of whether exploitation evidence is present in logs [1][3].

Background

LiteLLM is an open-source Python SDK and proxy server developed by BerriAI that provides a unified interface to more than 100 large language model providers, including OpenAI, Anthropic, Google Gemini, AWS Bedrock, Azure OpenAI, Cohere, and VertexAI, among others [4]. By exposing a single OpenAI-compatible API endpoint that handles routing, load balancing, cost tracking, and authentication

internally, LiteLLM has become a widely deployed component in enterprise AI infrastructure — the component through which all LLM traffic flows for organizations running LiteLLM that use multiple providers or need governance controls over AI spending and access. The project has accumulated over 45,000 GitHub stars and approximately 97 million monthly downloads on PyPI [4]. As a central aggregation point for provider credentials, LiteLLM concentrates an unusually high density of sensitive secrets in a single deployment: a typical configuration stores API keys for several LLM providers alongside virtual keys, master keys, and PostgreSQL connection strings in a single database.

These two incidents in early 2026 suggest that LiteLLM's architectural role as a credential aggregation point has drawn adversarial attention. In late March 2026, the threat actor cluster designated TeamPCP compromised LiteLLM's PyPI packages through a cascading supply chain attack that originated with a poisoned CI/CD security scanner, harvesting LLM provider credentials from any organization that installed the affected packages [5]. CVE-2026-42208, disclosed in April 2026, represents a structurally different attack path — a vulnerability in LiteLLM's own authentication code that requires no supply chain compromise and can be exploited directly over the network. The two incidents together underscore that LiteLLM's architectural role as a universal AI credential broker has made it a high-priority target whose security posture warrants corresponding scrutiny.

The vulnerability was addressed in LiteLLM 1.83.7-stable, released April 19, 2026, with the corresponding security advisory published to the project's GitHub repository on April 20 and indexed in the global GitHub Advisory Database on April 24 [1]. Within 36 hours of that indexing, Sysdig Threat Research documented the first confirmed exploitation attempts in the wild, with active payloads targeting the specific database tables that store the most valuable credentials [3].

Security Analysis

The Vulnerability

CVE-2026-42208 is a SQL injection defect in the code path LiteLLM executes when it validates an incoming API key against its PostgreSQL backend. When a request arrives at a LiteLLM proxy route — such as `POST /chat/completions` — the proxy checks the `Authorization: Bearer` header value against the `LiteLLM_VerificationToken` table to determine whether the caller is authorized. The vulnerable code constructs the database query by concatenating the raw Bearer token string directly into the SQL statement rather than using parameterized queries or prepared statements. Because this check is performed before any authentication is established, the injection point is reachable by any network-accessible client without prior authentication [2].

An attacker can trigger the injection by submitting a token value that contains a single quote or similar SQL metacharacter to break out of the intended string context. A subsequent UNION-based payload can then append attacker-controlled SQL that reads from arbitrary tables in the same database. The attack requires no credentials and no prior knowledge of the target organization's LiteLLM configuration. Any endpoint that invokes the key-verification path — which includes every standard LLM API route — serves as a valid injection entry point [2]. The vulnerability was assigned a CVSS base score of 9.3, reflecting the combination of network reachability, absence of authentication requirements, and high impact on confidentiality from the credential data exposed [1].

Timeline and Exploitation

The patch was released on April 19, 2026, and the advisory was published publicly on April 20 [1]. Four days later, on April 24, the advisory was indexed in the GitHub Advisory Database, which feeds automated vulnerability scanning and monitoring tools used by security researchers and threat actors alike. The window between public database indexing and confirmed first exploitation was approximately 36 hours and 7 minutes: Sysdig Threat Research documented the initial attack beginning at 04:24 UTC on April 26, 2026 [3].

The exploitation pattern indicated prior familiarity with LiteLLM's database schema. Rather than a generic SQL error-probing sweep, the threat actor deployed 17 distinct UNION-based payloads directed specifically at the three database tables that hold the most sensitive material: the `LiteLLM_VerificationToken` table, the `litellm_credentials.credential_values` column, and the `litellm_config.config_value` column [3]. This immediate and precise schema targeting — without an observable reconnaissance phase to enumerate table names — indicates that the actor either had prior detailed familiarity with LiteLLM's database layout or conducted schema reconnaissance from a separate, unlogged IP address before the observed attack sequence began. Sysdig identified two IP addresses associated with the attack: 65.111.27.132 for the initial probe phase and 65.111.25.67 for subsequent payload refinement, both resolving to AS200373, operated by 3xK Tech GmbH in Germany [3]. The actor has not been attributed to a named threat group as of this writing.

Impact Scope

The practical consequences of successful exploitation extend well beyond the immediate LiteLLM deployment. The `litellm_credentials` table characteristically stores upstream LLM provider credentials — including OpenAI organization API keys with associated spend caps, Anthropic console API keys with workspace-level administrative rights, and AWS Bedrock IAM credentials. The

`litellm_config` table contains runtime configuration values including database connection strings and environment variable references. The `LiteLLM_VerificationToken` table holds the virtual and master keys used to authenticate downstream callers to the proxy itself [2]. An attacker who successfully reads these three tables therefore obtains not only the keys needed to impersonate any user of the proxy, but also the credentials needed to make direct, unremediated calls to every LLM provider the organization uses — bypassing any spend limits, content filters, or audit logging that the proxy layer was providing. For organizations whose LiteLLM deployment holds organization-level or account-root provider credentials, the aggregate blast radius is comparable to a full cloud account compromise.

A temporary workaround was available prior to patching: setting `disable_error_logs: true` under `general_settings` in the LiteLLM configuration file disables the error-handling code path through which the vulnerable SQL query is reached [1]. This workaround reduces exploitation risk but is not a remediation and does not address the underlying injection defect. The only complete remediation is upgrading to version 1.83.7 or later.

Recommendations

Immediate Actions

Organizations running LiteLLM should upgrade to version 1.83.7-stable or later without delay. The patch replaces string concatenation in the token-verification query with parameterized statements, eliminating the injection vector entirely [1]. Any LiteLLM instance that was reachable from the internet during the period between initial release of the affected versions (1.81.16) and the application of the patch should be treated as potentially compromised regardless of whether logs show suspicious activity, because exploitation that successfully reads data from the database may not generate anomalous log entries in the application layer. Every virtual API key, master key, and upstream provider credential stored in the affected deployment — including OpenAI, Anthropic, Cohere, AWS Bedrock, and any other configured provider API keys — should be rotated as a precautionary measure.

Log review should focus on inbound requests to LLM API routes that contain SQL metacharacters in the `Authorization` header value, particularly single quotes, UNION keywords, and comments (`--`, `/**/`). Network logs should be examined for outbound connections to unfamiliar IP addresses around the time of known exploitation windows, particularly April 26, 2026. Organizations should also inspect PostgreSQL query logs, if enabled, for evidence of UNION SELECT operations against the credential and configuration tables.

Short-Term Mitigations

Organizations that cannot patch immediately should apply the `disable_error_logs: true` configuration workaround and place the LiteLLM administrative interface behind a network access control that restricts direct internet reachability. LiteLLM deployments exposed to the public internet with no network-layer access restrictions represent the highest-risk configuration; moving the proxy behind an API gateway or VPN endpoint substantially reduces attack surface while patching is arranged.

Credential stores within LiteLLM should be audited and scoped to enforce least privilege. Provider API keys used by LiteLLM should be created as project- or application-scoped keys with the minimum required permissions and spend limits, not as organization-level or account-root credentials, wherever the upstream provider API supports that scoping. AWS Bedrock integrations should use IAM roles with resource-based policies scoped to specific model ARNs rather than broadly permissive IAM user credentials. Key rotation intervals for all LLM provider credentials should be reviewed and shortened in light of this incident.

Strategic Considerations

CVE-2026-42208 illustrates a systemic risk that the expanding adoption of AI proxy infrastructure has created: the architectural concentration of credentials for an organization's entire AI provider portfolio in a single software component transforms any vulnerability in that component into a potentially organization-wide credential exposure. This risk was previously manifested through the TeamPCP supply chain attack on LiteLLM's PyPI packages in March 2026 [5], and it now manifests through a direct code vulnerability in the same software. The common thread is not a specific attack technique but the architectural reality that LiteLLM and its equivalents — whether open-source gateways like LiteLLM or commercial offerings — serve as a single point of trust for credentials that individually have very high value.

Security teams evaluating AI proxy infrastructure should apply the same threat modeling discipline to these gateways as they would to identity providers or secrets management systems. This includes requiring the proxy to run in an isolated, minimally permissioned environment with its own secret store rather than sharing credentials with other workloads; establishing rotation schedules for all upstream provider credentials that are independent of suspected compromise; and implementing continuous monitoring for anomalous LLM API usage patterns that might indicate stolen credentials being used outside the proxy layer. Given the 36-hour exploitation window documented for this CVE, the practical question for enterprise defenders is not whether widely-deployed critical vulnerabilities attract exploitation, but whether patching and credential rotation can be completed before opportunistic attackers reach their specific deployment.

CSA Resource Alignment

CVE-2026-42208 maps directly to several priority areas within CSA's AI Security Initiative frameworks. The MAESTRO threat modeling framework for agentic AI systems identifies the orchestration layer — the component that routes requests between AI consumers and AI providers — as a critical trust boundary requiring integrity and access controls equivalent to those applied to identity infrastructure [6]. An AI proxy that stores provider credentials in a SQL-injectable database and exposes that database to unauthenticated network callers represents a failure of the fundamental trust controls MAESTRO prescribes for this layer. Organizations using MAESTRO to model their AI architecture should explicitly include the proxy tier as an adversarial target in their threat models and evaluate whether their current proxy deployment would contain or amplify a credential theft event.

The AI Controls Matrix (AICM), which supersedes and extends the Cloud Controls Matrix to address AI-specific risks, provides directly applicable controls in the Identity and Access Management and Secrets Management domains [7]. The AICM includes controls addressing least-privilege credential scoping, secrets rotation policy, and dependency integrity verification, each of which is directly implicated by this vulnerability and the exploitation pattern it attracted. Specifically, the concentration of organization-level provider credentials in a proxy with a history of supply chain attacks and now a direct SQL injection vulnerability makes a strong case for implementing the AICM's recommended credential compartmentalization: each downstream application should consume provider credentials scoped only to its specific use case, rather than relying on a shared credential pool in the proxy tier.

CSA's Zero Trust guidance is applicable to the credential-use phase of this attack class [8]. A Zero Trust posture for LLM API access — in which every call to a provider API is authenticated at the provider level using a credential scoped to the specific calling workload — would significantly limit the value of credentials stolen through proxy-layer vulnerabilities. If each application workload holds only the key it needs for its specific models and budget, the total exposure from any single proxy compromise is bounded to that workload's scope rather than the entire organization's AI stack. The CSA STAR program provides an assessment framework that AI tool vendors, including LiteLLM's commercial support entity, can use to demonstrate security posture to enterprise customers evaluating the risk of deploying third-party AI gateway software.

References

[1] BerriAI / LiteLLM. "Security Advisory: CVE-2026-42208 — Pre-Auth SQL Injection in API Key Verification Path." GitHub Security Advisories, April 2026.

[2] BleepingComputer. "Hackers Are Exploiting a Critical LiteLLM Pre-Auth SQLi Flaw." BleepingComputer, April 2026.

[3] Sysdig Threat Research Team. "CVE-2026-42208: Targeted SQL Injection Against LiteLLM's Authentication Path Discovered 36 Hours Following Vulnerability Disclosure." Sysdig Blog, April 2026.

[4] BerriAI. "LiteLLM: Open-Source LLM Gateway." GitHub, 2026.

[5] Datadog Security Labs. "LiteLLM and Telnyx Compromised on PyPI: Tracing the TeamPCP Campaign." Datadog Security Labs, March 2026.

[6] Cloud Security Alliance. "MAESTRO: Multi-Agent Environment, Security, Threat, and Risk Ontology." CSA AI Safety Working Group, 2025.

[7] Cloud Security Alliance. "AI Controls Matrix (AICM) v1.0." CSA AI Safety Initiative, 2025.

[8] Cloud Security Alliance. "Zero Trust Advancement Center." CSA, 2025.

Further Reading

The following sources provide additional coverage of CVE-2026-42208 and were consulted during research but are not directly cited in the analysis above.

- The Hacker News. "LiteLLM CVE-2026-42208 SQL Injection Exploited Within 36 Hours of Disclosure." April 2026.
- Security Affairs. "CVE-2026-42208: LiteLLM Bug Exploited 36 Hours After Its Disclosure." April 2026.
- Cyber Security News. "Critical LiteLLM SQL Injection Vulnerability Exploited in the Wild." April 2026.

