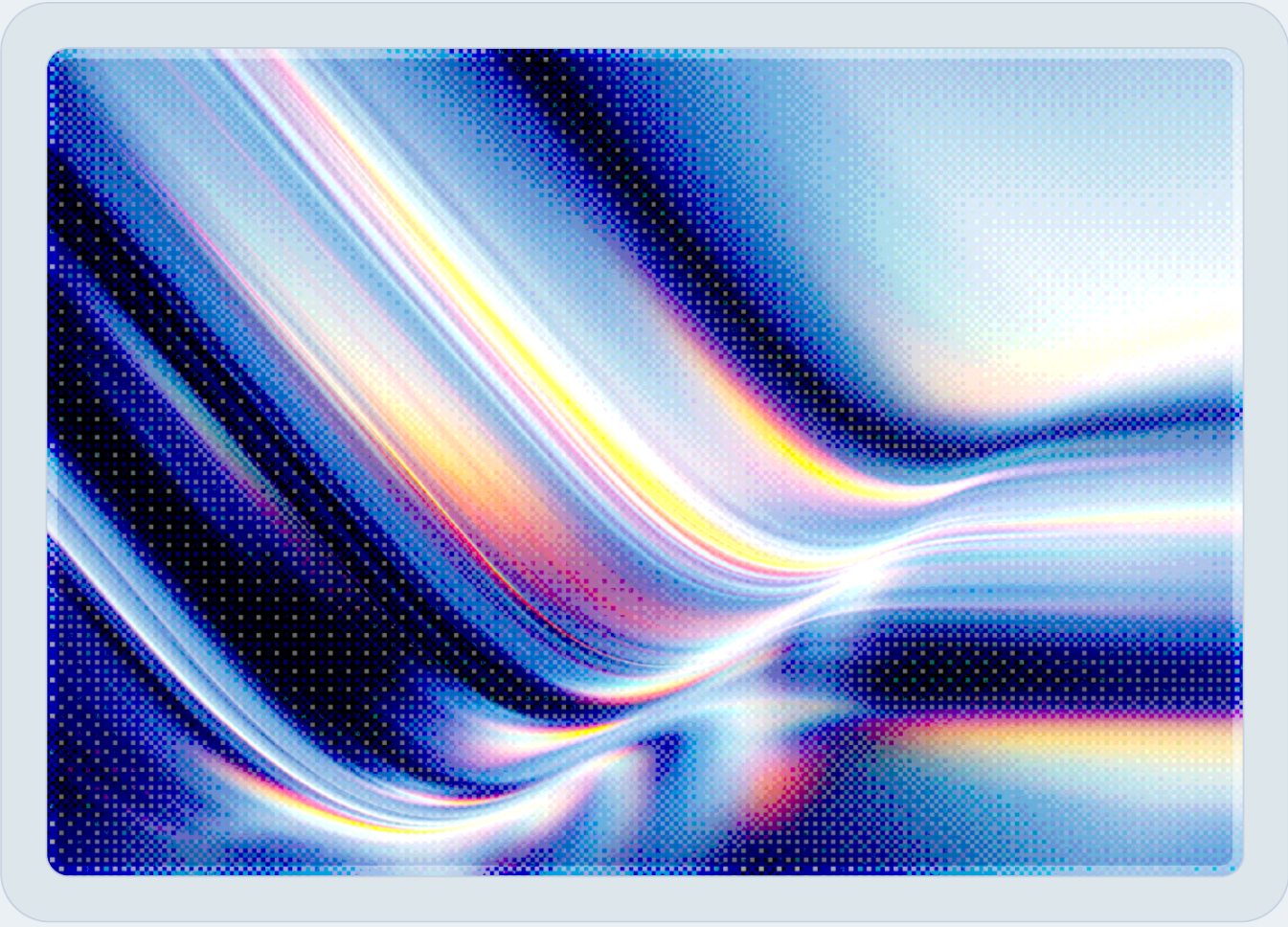
CSAI Foundation | Cloud Security Alliance

LiteLLM RCE Chain: AI Gateway Under Active Exploitation

CVE-2026-42271 and CVE-2026-48710 Combined for Unauthenticated Remote Code Execution

2026-06-09

AI-assisted Rapid Research



CSAI | CSA cloud security alliance®

Key Takeaways

Two vulnerabilities, individually moderate to high in severity, are being chained in the wild to achieve unauthenticated remote code execution against LiteLLM deployments — a widely deployed AI proxy that holds the API credentials for every large language model an organization uses. CISA added the primary vulnerability to its Known Exploited Vulnerabilities catalog on June 8, 2026 [1]. Organizations running LiteLLM versions prior to 1.83.7, or Starlette versions prior to 1.0.1 (0.46.1 and earlier), should treat this as an active incident and patch immediately.

- CVE-2026-42271 (CVSS 8.7) allows any authenticated LiteLLM user to inject arbitrary commands via Model Context Protocol test endpoints that spawn unvalidated subprocesses [2].
- CVE-2026-48710 "BadHost" is a critical-severity authentication bypass in Starlette, the underlying ASGI framework, which allows an unauthenticated attacker to poison the HTTP Host header and circumvent credential checks entirely [5].
- Chained together, the two CVEs yield an unauthenticated remote code execution path that Horizon3.ai assessed at the maximum CVSS severity of 10.0 [2].
- LiteLLM has been the subject of three critical security incidents in 2026 — a supply chain compromise in March, a CISA KEV-listed SQL injection in May, and this CISA KEV-listed RCE chain in June — establishing a pattern of recurring critical exposure in a single high-value component [8][10].
- The BadHost vulnerability affects Starlette broadly, placing vLLM, FastAPI, the Python MCP SDK, and any Python ASGI application using path-based authentication middleware at risk [5] [7].

Background

LiteLLM is an open-source proxy server created by BerriAI that translates API calls into a unified OpenAI-compatible format across more than one hundred large language model providers, including OpenAI, Anthropic, Google Gemini, Azure OpenAI, and AWS Bedrock [10]. Organizations adopt it as a centralized AI gateway to manage routing, rate limiting, spend tracking, and credential storage across multiple model providers. A single LiteLLM deployment therefore holds API keys for every model

provider an organization has integrated, making it a high-value credential store with elevated appeal to attackers. The package is reportedly downloaded approximately 3.4 million times per day from PyPI [10], reflecting its deep integration into enterprise AI stacks.

LiteLLM's architectural position — sitting between application code and every upstream LLM provider — means a compromise does not merely expose one service. An attacker who controls the proxy can intercept all inbound prompts and outbound completions, harvest stored provider credentials, modify routing rules to redirect traffic, and pivot laterally into the Kubernetes clusters, cloud platforms, or internal services that the proxy process can reach. This attack surface profile has attracted sustained adversarial interest throughout 2026.

The RCE chain disclosed on June 9, 2026 is the third major security incident affecting LiteLLM this year. In March 2026, the package was targeted in a sophisticated supply chain compromise when threat actors deployed malicious versions to PyPI containing a payload designed to harvest credentials and facilitate Kubernetes lateral movement [10][11]. The following month, Bishop Fox and Sysdig independently disclosed CVE-2026-42208, a pre-authentication SQL injection in LiteLLM's proxy API key verification path that attackers began exploiting within 36 hours of public disclosure [18][9]. CISA added CVE-2026-42208 to the KEV catalog on May 8, 2026, requiring Federal Civilian Executive Branch agencies to remediate by May 11, 2026 [17]. CSA's AI Safety Initiative published a dedicated research note covering that vulnerability [16]. The current advisory addresses the next wave: two independently disclosed vulnerabilities, CVE-2026-42271 and CVE-2026-48710, which Horizon3.ai researchers have chained to produce unauthenticated remote code execution against unpatched deployments [2].

Security Analysis

CVE-2026-42271: Command Injection via MCP Test Endpoints

CVE-2026-42271 was introduced alongside LiteLLM's Model Context Protocol integration, which provides preview endpoints allowing users to test MCP server connectivity directly from the proxy. Two endpoints — `POST /mcp-rest/test/connection` and `POST /mcp-rest/test/tools/list` — accept a full MCP server configuration object in the request body, including the `command`, `args`, and `env` fields used by the stdio transport [2][13]. When an MCP server is configured to use stdio, LiteLLM spawns the specified command as a subprocess on the proxy host. The test endpoints pass the caller-supplied configuration directly to this subprocess launch without sanitizing or restricting the command field.

The vulnerability is compounded by missing role-based access controls on these endpoints. Although the affected versions required a valid proxy API key to reach the endpoints in isolation, no authorization check distinguished between administrative users and low-privilege internal users with basic key access. Any holder of a valid proxy API key could therefore execute arbitrary commands with the operating-system privileges of the LiteLLM proxy process [2][13]. In containerized or Kubernetes deployments, this typically means access to the pod's service account token, mounted secrets, and the container's file system — including credentials written to environment variables or configuration files.

The vulnerability affects LiteLLM versions 1.74.2 through 1.83.6 inclusive [2]. BerriAI addressed it in version 1.83.7 by removing the unvalidated command execution path from the test endpoints and adding authorization checks.

CVE-2026-48710 (BadHost): Authentication Bypass in Starlette

CVE-2026-48710, dubbed "BadHost," is a host header validation flaw in Starlette, the lightweight ASGI framework that underlies FastAPI and, transitively, LiteLLM [5][6]. Starlette builds the `request.url` object — including `request.url.path` — from the raw HTTP `Host` header it receives. Authentication middleware commonly uses `request.url.path` to determine whether a request targets a protected endpoint. An attacker can exploit the trust relationship between these two components by injecting a `/`, `?`, or `#` character into the `Host` header value. When Starlette parses the resulting malformed URL, `request.url.path` resolves to `/` (or another unprivileged path) while the router still dispatches to the actual requested endpoint based on the true request target [5][7].

The practical consequence is that authentication middleware examining `request.url.path` concludes the request targets an unprotected public route and does not enforce credential checks, while the request handler executes the actual privileged or sensitive endpoint the attacker intended to reach. The vulnerability is architectural rather than application-specific: because it resides in how Starlette constructs URLs from raw headers, it affects every application built on the framework that uses path-based auth middleware, regardless of application-layer logic [5]. Security firm X41 D-Sec, in an OSTIF-sponsored audit, publicly disclosed the technical findings [6]. Starlette has more than 400,000 dependent projects on GitHub, according to reported counts [7], reflecting the breadth of the affected ecosystem. Beyond LiteLLM, confirmed affected stacks include vLLM, FastAPI applications using path-based auth, and the Python MCP SDK [7].

Starlette 1.0.1 addresses the flaw by validating the `Host` header before using it to construct the request URL [5]. Applications that have not updated remain at risk even if they are not running LiteLLM.

The Attack Chain: Unauthenticated RCE at CVSS 10.0

Horizon3.ai researchers demonstrated that CVE-2026-48710 and CVE-2026-42271 compose into a single, credential-free exploit path [2]. In the chained attack, an adversary sends a `POST` request targeting `/mcp-rest/test/connection` or `/mcp-rest/test/tools/list` with a Host header crafted to contain a trailing `?` or `#`. The BadHost flaw causes Starlette to report `request.url.path` as `/` to the authentication middleware, which concludes the request is accessing a public endpoint and passes it through without checking the `Authorization` header. The router processes the true target path normally, dispatching the request to the MCP test handler. That handler receives the attacker-controlled request body — containing an arbitrary `command` field — and spawns it as a subprocess, yielding code execution on the proxy host [2][3].

Horizon3.ai assessed the resulting attack chain at the maximum CVSS severity (10.0) [2]. Successful exploitation enables an attacker to run arbitrary operating system commands as the LiteLLM proxy process; read all API keys and model provider credentials stored in the proxy's database and environment; intercept or modify all LLM traffic passing through the gateway; and pivot laterally into cloud platforms, Kubernetes clusters, or other internal systems that the proxy process can reach [3][4]. CISA added CVE-2026-42271 to the Known Exploited Vulnerabilities catalog on June 8, 2026, indicating that exploitation against real-world deployments has been confirmed [1][14][15].

LiteLLM's trajectory through 2026 — supply chain compromise, pre-authentication SQL injection, and now unauthenticated RCE — reflects a broader dynamic in AI infrastructure security. Three distinct critical incidents against a single component in one year suggest that AI middleware aggregating credentials and traffic attracts elevated adversarial interest — an expected consequence of the compounding impact a successful compromise enables.

Recommendations

Immediate Actions

Organizations should treat this vulnerability chain as requiring emergency response if they are running LiteLLM versions 1.74.2 through 1.83.6, or if their dependency tree includes Starlette versions prior to 1.0.1 (0.46.1 and earlier). Both upgrades are independent and should be applied without waiting for the other:

- Upgrade LiteLLM to version 1.83.7 or later, which removes the unvalidated command execution path and adds authorization controls to the MCP test endpoints [2].
- Upgrade Starlette to version 1.0.1 or later, which validates the `Host` header before using it in URL construction [5].
- If immediate patching is not operationally feasible, block `POST /mcp-rest/test/connection` and `POST /mcp-rest/test/tools/list` at the reverse proxy or API gateway as an interim measure [2].
- Rotate all API keys and model provider credentials stored in or accessible to the LiteLLM proxy. If exploitation has occurred against a vulnerable deployment, treat all stored secrets as compromised.
- Review process execution logs and network egress from LiteLLM host environments for anomalous subprocess activity or unexpected outbound connections that may indicate prior exploitation.

Organizations that have not yet applied the Starlette 1.0.1 update should treat CVE-2026-48710 as an independent remediation item, given its broad impact across the ASGI ecosystem regardless of LiteLLM deployment status.

Short-Term Mitigations

Beyond emergency patching, organizations should harden the network and access posture of LiteLLM deployments to reduce exposure in the event of future vulnerabilities. LiteLLM should never be directly internet-accessible; it should be placed behind an authenticated reverse proxy or API gateway that enforces network-level access controls before requests reach the application [4]. Ingress rules should restrict access to trusted internal CIDR ranges, and `Host` header normalization should be performed at the load balancer layer to prevent raw malformed headers from reaching the ASGI application. Where LiteLLM runs in Kubernetes, pod security policies should restrict the subprocess execution capabilities available to the container, and the service account token should be scoped to the minimum necessary permissions.

Organizations should also review the logs from their reverse proxy and LiteLLM instances for request patterns consistent with BadHost exploitation: requests containing `Host` header values with embedded `/`, `?`, or `#` characters, particularly when targeting the MCP test endpoints [4]. A community-developed detection scanner is publicly available to assess whether exposed Starlette deployments are vulnerable to CVE-2026-48710 [5]. Asset discovery tools such as runZero have published guidance for locating LiteLLM instances within internal networks [4].

Strategic Considerations

The 2026 pattern of LiteLLM vulnerabilities — recurring critical-severity flaws in a component that aggregates AI credentials and traffic — surfaces a structural risk that organizations should address architecturally rather than through repeated reactive patching. AI gateways occupy a uniquely privileged position in the AI stack: they hold the credentials for all upstream model providers, see every prompt and completion, and typically run with broad network access to internal services. This privileged position makes them high-value targets, and it means that any vulnerability in a gateway has compounding impact across the entire AI environment. Organizations should apply the same rigor to AI middleware selection and lifecycle management that they apply to identity providers and secrets management systems.

Concretely, this means ensuring that LiteLLM and equivalent AI gateways are included in software composition analysis pipelines and receive vulnerability feed monitoring on the same cadence as other critical infrastructure. Model provider credentials should be stored in a dedicated secrets manager with rotation policies rather than in the gateway's local database, minimizing the value of a credential-harvesting attack. Security teams should also evaluate whether their gateway deployment surface is strictly necessary — the extent to which the MCP test endpoints, administrative APIs, and other development-oriented features are exposed in production environments should be audited and pruned.

CSA Resource Alignment

The LiteLLM RCE chain aligns with threat scenarios documented in CSA's MAESTRO framework for agentic AI threat modeling. MAESTRO's Layer 5 (Tool and Plugin Management) identifies unauthorized tool execution as a primary attack vector in AI systems that integrate external tools through mechanisms such as MCP. The vulnerable MCP test endpoints in CVE-2026-42271 represent an uncontrolled extension of this execution surface: they expose the stdio subprocess launch mechanism — designed for controlled tool integration — to arbitrary command injection. Layer 7 of MAESTRO covers AI gateway compromise as a threat that grants attackers control over the full AI environment, consistent with the lateral movement and credential harvesting scenarios enabled by this RCE chain.

AICM's supply chain and operational security domains provide relevant controls for this vulnerability class, including guidance on third-party component validation, dependency pinning, and runtime integrity monitoring. Organizations that have implemented AICM's AI operational security controls — including network segmentation for AI infrastructure and secrets management for model provider

credentials — will have meaningfully reduced the blast radius of this attack chain even before patching. The AICM framework applies equally to the LiteLLM application layer and to its transitive dependency on Starlette, reinforcing the importance of tracking the full dependency chain in AI middleware stacks.

Zero Trust principles require that AI credential aggregators receive micro-segmentation, continuous authentication, and least-privilege network controls regardless of their internal network position. LiteLLM's architectural role as a credential aggregator for AI workloads makes it a natural candidate for Zero Trust network controls that verify every request regardless of network origin. Organizations conducting STAR for AI assessments of their AI vendor ecosystem should include AI gateway and middleware components in scope, evaluating vendors against their patch cadence, vulnerability disclosure practices, and security architecture.

References

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

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

[3] The Hacker News. "LiteLLM Flaw CVE-2026-42271 Exploited in the Wild, Chains to Unauthenticated RCE." The Hacker News, June 2026.

[4] runZero. "LiteLLM Proxy Vulnerabilities: How to Find Impacted Assets." runZero Research, June 2026.

[5] BadHost Project. "BadHost — CVE-2026-48710: Starlette Host-Header Auth Bypass." badhost.org, 2026.

[6] OSTIF. "Disclosing the BADHOST Vulnerability in Starlette." OSTIF, 2026.

[7] MLQ.ai. "Critical 'BadHost' Flaw in Starlette Exposes Millions of AI Agent Deployments to Auth Bypass." MLQ.ai, 2026.

[8] The Hacker News. "LiteLLM CVE-2026-42208 SQL Injection Exploited within 36 Hours of Disclosure." The Hacker News, April 2026.

[9] Sysdig. "CVE-2026-42208: Targeted SQL Injection Against LiteLLM's Authentication Path Discovered 36 Hours Following Vulnerability Disclosure." Sysdig, 2026.

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

[11] LiteLLM. "Security Update: Suspected Supply Chain Incident." LiteLLM Blog, March 2026.

[12] Red Hat. "CVE-2026-48710." Red Hat Security, 2026.

[13] SentinelOne. "CVE-2026-42271: LiteLLM RCE Vulnerability." SentinelOne Vulnerability Database, 2026.

[14] CISA. "Known Exploited Vulnerabilities Catalog." CISA, 2026.

[15] CybersecurityNews. "Hackers Exploiting LiteLLM RCE Vulnerability in the Wild to Run Arbitrary Commands." CybersecurityNews, June 2026.

[16] CSA AI Safety Initiative. "LiteLLM CVE-2026-42208: AI Proxy SQL Injection." CSA Labs, May 2026.

[17] Security Affairs. "U.S. CISA Adds a Flaw in BerriAI LiteLLM to Its Known Exploited Vulnerabilities Catalog." Security Affairs, May 2026.

[18] Bishop Fox. "CVE-2026-42208: Pre-Authentication SQL Injection in LiteLLM Proxy." Bishop Fox, April 2026.