contact@namiq.io · namiq.io

NamiQ

SECURITY ADVISORY

BerriAI LiteLLM exploited

CVE-2026-40217 · on CISA's exploited list since 18 November 2026

Analysis Report

If you run LiteLLM, ask whoever operates it to act before 2 December 2026. First find out whether it is deployed, who holds a key for it and whether it is reachable from outside; if you do not run …

PREPARED BY	NamiQ	DATE	18 November 2026
FACTS AS OF	18 November 2026	VERSION	1.0
SEVERITY	**High: CVSS 3.1 8.8**	STATUS	On CISA's exploited list; due 2 December 2026

NamiQ © 2026 NamiQ. All rights reserved. This document is the property of NamiQ. It may be shared unaltered, with attribution.
Companion article: namiq.io/insights/berriai-litellm-cve-2026-40217

The bottom line

> *If you run LiteLLM, ask whoever operates it to act before 2 December 2026. First find out whether it is deployed, who holds a key for it and whether it is reachable from outside; if you do not run it, stop reading.*

At a glance

Fact	Detail
Product	BerriAI LiteLLM (CVE-2026-40217)
Severity	High: 8.8 (CVSS 3.1)
CISA	On its exploited list since 18 November 2026. Deadline for US federal agencies: 2 December 2026.
Chance of exploitation (EPSS)	0.17, the estimated chance in the next 30 days, as at 18 November 2026
Ransomware use	Not recorded by CISA. That is not the same as none.
Fixed in	No source we read names a fixed version
Who looks first	Anyone running the LiteLLM proxy, especially from the Docker image, with API keys held by staff, contractors or other systems. Firms that only call hosted AI services and do not run the proxy are not affected.

Does this apply to you?

Answer in order. Stop at the first no.

Question	If no	If yes
Do you or your IT provider run a LiteLLM proxy?	Nothing to do for this CVE.	Check the version or image (next question).
Is it the Docker image, or a build from 8 April 2026 or earlier?	Ask the vendor whether your build is affected; the record does not say.	Treat it as affected and go on.
Can anyone you do not fully trust hold a key or login for the proxy?	Risk is lower, but still plan to patch.	Remove or rotate those keys now.
Can the proxy be reached from the internet or a shared network?	Exposure is limited to people inside.	Restrict access to known addresses first.

What it is

A person who is already logged in to the LiteLLM proxy can send Python code to a guardrail test endpoint. The record says a trick that rewrites the code's bytecode lets that code run on the server.

Why it matters

The researchers' advisory says the code runs as the user the LiteLLM process runs as, which is root in the default Docker image. Our reading is that such a proxy usually holds the keys to every AI provider it fronts, so one stolen low-level key could become full control of the host and those keys.

What the vendor's advisory says

We read two pages on github.com and www.x41-dsec.de. In short:

- The X41 D-Sec advisory rates this High, with a score of 8.7 under CVSS 4.0.
- It says the endpoint filters submitted code with regular expressions, and that this filter can be bypassed.
- It confirms the flaw on the main-latest Docker image and lists confirmed patched versions as TBD.
- It says the attack needs an authenticated HTTP API request, and that the CVE and vendor reference were still TBD.
- A GitHub entry for this CVE lists CWE-913 and a high privilege requirement; the advisory says low privilege.

Things people will get wrong

- "It needs a login, so it is not urgent." The advisory says a low-privilege authenticated user is enough. Shared, leaked or over-issued keys count as logins.
- "Updating to the latest image fixes it." The advisory confirms the flaw on main-latest and lists patched versions as TBD. No source names a fixed version.
- "The filter on the endpoint stops it." The advisory says the regex filter is exactly what the bypass defeats.

How we would triage it

This is our judgement.

The commands only read: they print a version or list what is installed. They come from general knowledge of this kind of product, not from the sources, so check them against the vendor's page before you run them. Words in capitals are for you to fill in.

Step 1: Find where LiteLLM runs and which version

- Ask your IT provider to list every LiteLLM proxy, including test machines, and record the image digest or package version.
- Compare the digest with the one the record lists; the record gives no version numbers.

```
docker ps --format '{{.Names}} {{.Image}}'
docker images --digests | grep -i litellm
pip show litellm
```

Step 2: Reduce who can reach it

- List every key and login for the proxy and revoke any that are unused or shared.
- Close it to the internet if it does not need to be open. No source names a workaround, so this is our reading.

Step 3: Apply the vendor's mitigation

- CISA says to apply the vendor's mitigations. Ask the vendor or your provider what they are, since no source names a fixed version.
- Do not rely on the guardrail testing feature being unused; the record gives no way to turn it off.

Step 4: Check for signs of misuse

- Review proxy logs for calls to the guardrail test endpoint from keys that do not normally use it.
- If seen, rotate every provider key stored in the proxy.

To close it, read the version from the product itself, not from an old inventory, and save the output with the date. If you cannot update today, write down who owns it, the new date, and what limits exposure until then. CISA's wording adds that you should stop using the product if no mitigation is available.

Send this to your IT provider

Copy it, change the names and dates, and send it. It is a draft for you to check, not a template tested with any provider.

> *Hello, we need to know whether we run BerriAI LiteLLM, and if so where. It is covered by CVE-2026-40217, which allows code execution through the guardrail test endpoint. Please tell us which hosts run it, the version or image digest, and who holds keys for it. Please also tell us the fixed version, or the vendor's mitigation if none is named. CISA asks for mitigation by 2 December 2026. Please confirm in writing, by the end of this week, the version we run and what has been done.*

Signs it was used

An update closes the hole. It does not tell you whether someone used it first. CISA's required action points to its forensic triage requirements (BOD 26-04). Look for:

- Requests to the guardrail test endpoint, especially from keys that do not normally use it.
- Unexpected child processes or new files created by the LiteLLM process.
- Provider API keys used from unfamiliar addresses after such requests.

The weakness behind it

The record classes this as CWE-420 (Unprotected Alternate Channel).

Likely ATT&CK mapping

MITRE ATT&CK is a public catalogue of the steps attackers take, from getting in to causing damage. No source has mapped this flaw to it yet, so the steps below are our prediction from what the record says the flaw allows. They are not MITRE's, the vendor's or CISA's, and an attacker may do other things.

Steps an attacker would likely take with this flaw

The step, in plain words	Stage of an attack	Why we think so
Exploit Public-Facing Application (T1190): Breaking in through a service that faces the internet, such as a web server, VPN or login page.	Initial Access: getting in	The flaw is reachable over HTTP on a proxy that may face the network, so exploitation would likely start as an attack on a public-facing application.
Command and Scripting Interpreter (T1059): Running commands or scripts on a machine, such as a shell or PowerShell.	Execution: running the attacker's code	The record describes running arbitrary code on the server, which would typically give the attacker a way to run commands as the process user.

How to use this: ask your IT provider which control or log would show each step, for example whether sign-in attempts from the internet are recorded. If nobody can answer, that is the gap to close first.

MITRE ATT&CK® is a registered trademark of The MITRE Corporation. Technique names are MITRE's.

The sources, in their words

CISA's catalogue entry:

> *BerriAI LiteLLM contains an unprotected alternate channel vulnerability that could allow remote attackers to execute arbitrary code via bytecode rewriting at the /guardrails/test_custom_code URI.*

The CVE record, published by Mitre on 10 April 2026:

> *LiteLLM through 2026-04-08 allows remote attackers to execute arbitrary code via bytecode rewriting at the /guardrails/test_custom_code URI.*

Versions of BerriAI LiteLLM the record lists as affected:

- bb0639701796218a3447160e55c0f1097446e4e6085df7dfd39f476d4143743f

CISA's required action:

> *Apply mitigations in accordance with vendor instructions.*

Check your own project

If you build software, you can compare what a project declares with this CVE from a terminal. The tool reads requirements files, package.json, Maven, Gradle, Cargo, CMake and Dockerfile files, runs nothing, and only the CVE id leaves your machine. Naming the CVEs needs no key.

```
lattice-scan project . --cve CVE-2026-40217
```

Each row says in-range, outside-range, undetermined or lead, with the file and line. That is evidence about what a manifest declares, not about what runs, and a person decides. The verdicts, the CSV columns and a pipeline recipe are in the guide.

Check the servers you run

The same id can be checked against a host. Run this on the machine itself, not on the laptop that deploys to it: it reads installed packages (dpkg, rpm or apk), snaps, the hardware (maker, model, firmware), the running container images with their operating system and packages, and the kernel version, compares them on the spot, and only the CVE ids leave the machine; the inventory stays on

it, in ~/.lattice-scan. Naming the CVEs needs no key. Do it on each server after a deployment, and again when a new entry lands.

```
lattice-scan --cve CVE-2026-40217
```

A match is a candidate for a person to confirm, not a finding. A distribution can fix a CVE under an unchanged version number, so a match is a candidate; on Debian-family hosts the report also shows whether the package's own changelog names the CVE. By default it opens the images your containers are running and reads their operating system and package list; a deep scan (--deep, a few minutes) opens every local image and the Python, Node, Java and Ruby packages inside them; for what an image declares, use the project check on its Dockerfile.

Guide: namiq.io/lattice/guides/choose-your-channel. MCP server (io.namiq/lattice in the MCP registry): namiq.io/lattice. Package: pypi.org/project/lattice-scan, or namiq.io/apt.

What we do not know yet

- Which version first fixes this; the advisory lists patched versions as TBD.
- Whether builds other than the Docker image main-latest are affected; the record lists only one image digest.
- Whether the guardrail test endpoint can be turned off or blocked without breaking normal use.
- Whether the record's CWE-420 or the GitHub entry's CWE-913 is the right weakness.
- We have not tested this against any environment. Versions and scores are as the sources state them and as at 18 November 2026; the vendor's advisory wins where it differs.
- The sections headed "The bottom line", "Does this apply to you?", "What it is", "Why it matters", "Exposure by situation", "Things people will get wrong", "Likely ATT&CK mapping", "How we would triage it" and "Send this to your IT provider" are our interpretation of the sources, not a statement by CISA or the vendor.

Appendix 1. Severity decoded

CVE-2026-40217: score 8.8, `CVSS:3.1/AV:N/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H` As the CVE record gives it. The record's own score can differ from the base score on the NVD record.

Metric	Value	What it means here
Attack vector	Network	Reachable over a network, with no need to be on the same segment.
Attack complexity	Low	No special conditions beyond the flaw itself.
Privileges required	Low	An ordinary account is enough.
User interaction	None	No one has to do anything.
Scope	Unchanged	Damage stays inside the vulnerable component.
Confidentiality	High	Total loss of confidentiality.
Integrity	High	Total loss of integrity.
Availability	High	Total loss of availability.

Appendix 2. Timeline

Date	Event
10 April 2026	CVE record published by Mitre.
18 November 2026	CISA added it to its list of exploited vulnerabilities.
2 December 2026	CISA's deadline for US federal agencies.

Appendix 3. Exposure by situation

These priorities are our judgement from the attacker prerequisites in the sources. They are not vendor ratings.

Your situation	Priority	Timeframe
Proxy reachable from the internet, many users or keys, on the Docker image	High	within CISA's window; start now
Proxy internal only, but keys shared with contractors or other systems	Medium	this week
Proxy internal only, one trusted administrator holds the only key	Low	next patch cycle

You do not run LiteLLM	None	none for this CVE

Appendix 4. Definition of done

- Every BerriAI LiteLLM you run is listed, with its version.
- Each is outside the affected list, or the vendor's mitigation is in place.
- The version output or a screenshot is saved with the date.
- Your projects' declared dependencies were checked against this CVE, and every row has a written reason.
- Logs for the weeks before the update were searched, and anything odd was written down.
- Anything not done has an owner and a date.

Sources

- CISA Known Exploited Vulnerabilities catalogue: https://www.cisa.gov/known-exploited-vulnerabilities-catalog
- CVE record for CVE-2026-40217: https://www.cve.org/CVERecord?id=CVE-2026-40217
- FIRST: Exploit Prediction Scoring System: https://www.first.org/epss/
- Advisory or release notes read: github.com/BerriAI/litellm/security/advisories/GHSA-wxxx-g: https://github.com/BerriAI/litellm/security/advisories/GHSA-wxxx-gvqv-xp7p
- Advisory or release notes read: www.x41-dsec.de/lab/advisories/x41-2026-001-litellm/: https://www.x41-dsec.de/lab/advisories/x41-2026-001-litellm/