

sgl-project / sglang Public[Code](#) [Issues 602](#) [Pull requests 2.1k](#) [Discussions](#) [Actions](#) [Security a](#)

fix(security): replace unsafe pickle.loads with SafeUnpickler for CVE-2026-3989 #20904

MergedFridge003 merged 11 commits into `sgl-project:main` from`zwang86:fix/cve-2026-3059-3060-39...` 2 weeks ago[Conversation 23](#)[Commits 11](#)[Checks 118](#)[Files changed 3](#)

zwang86 commented 3 weeks ago • edited ▾

Contributor

Motivation

Three critical CVEs (CVSS 9.8) affect SGLang due to unsafe `pickle.loads()` / `recv_pyobj()` deserialization over unauthenticated ZMQ sockets, enabling Remote Code Execution (RCE):

CVE	Component
CVE-2026-3059	Multimodal generation ZMQ broker
CVE-2026-3060	Encoder Parallel Disaggregation
CVE-2026-3989	<code>replay_request_dump.py</code> script

References: [CERT/CC VU#665416](#), [ShadowMQ \(Oligo Security\)](#), [#5569](#)

This PR uses the existing `SafeUnpickler` (allowlist/denylist approach) as a drop-in replacement for `pickle.loads()` on all CVE-affected paths, blocking known RCE gadget chains (`os.system`, `subprocess.Popen`, `eval`, `exec`, etc.).

Modifications

- `python/sglang/srt/utils/common.py`: Extend `SafeUnpickler.ALLOWED_MODULE_PREFIXES` with `sglang.srt.disaggregation.` and `sglang.multimodal_gen.` prefixes; add `safe_pickle_loads()` and `safe_pickle_load()` helper functions.

- `python/sglang/multimodal_gen/runtime/scheduler_client.py` ([CVE-2026-3059](#)): Replace 4x `unsafe pickle.loads / recv_pyobj` with `safe_pickle_loads` :
- `python/sglang/multimodal_gen/runtime/managers/scheduler.py` ([CVE-2026-3059](#)): Replace 1x `unsafe pickle.loads` with `safe_pickle_loads` :
- `python/sglang/srt/disaggregation/encode_receiver.py` ([CVE-2026-3060](#)): Replace 2x `unsafe pickle.loads` with `safe_pickle_loads` :
- `scripts/playground/replay_request_dump.py` ([CVE-2026-3989](#)): Replace 1x `unsafe pickle.load` with `safe_pickle_load` .
- `docs/developer_guide/contribution_guide.md` : Add security note in code style guidance warning against unsafe pickle deserialization.

Note: `pickle.dumps()` / `send_pyobj()` calls are not changed — serialization is safe; only deserialization is the attack vector.

Accuracy Tests

N/A — This change only affects the deserialization path (restricting which classes can be loaded). No model output or computation logic is modified.

Benchmarking and Profiling

N/A — `SafeUnpickler` adds a negligible `find_class` check per deserialized object, which is not on the hot path.

Checklist

- Format your code according to the [Format code with pre-commit](#).
- Add unit tests according to the [Run and add unit tests](#).
- Update documentation according to [Write documentations](#).
- Provide accuracy and speed benchmark results according to [Test the accuracy](#) and [Benchmark the speed](#).
- Follow the SGLang code style [guidance](#).



[fix\(security\): replace unsafe pickle.loads with SafeUnpickler for CVE...](#)

✖ [2face7e](#)

...



zwang86 requested review from **ByronHsu, ShangmingCai, hnyls2002, mickqian, ping1jing2** and **yhyang201** as [code owners](#) 3 weeks ago

gemi-code-assist bot commented [3 weeks ago](#)

Contributor

 **Warning**

You have reached your daily quota limit. Please wait up to 24 hours and I will start processing your requests again!

 **github-actions** bot added the **diffusion** label [3 weeks ago](#)

 **zwang86** mentioned this pull request [3 weeks ago](#)

[Bug] [SECURITY] Critical security incidents of SGLang #5569

 Closed

 5 tasks

ZailiWang commented [3 weeks ago](#)

Contributor

Hi, thanks for fixing this which keeps SGL safe and trusted.

A small suggestion is that: shall we add a note in [the contribution guide doc](#) that `pickle` should be avoided in the project as it is not secure, as mentioned in [the official python document](#).

 1

 [docs: add pickle security warning to contribution guide](#) ...

 [7fe5f0a](#)

 **github-actions** bot added the **documentation** label [3 weeks ago](#)

zwang86 commented [3 weeks ago](#)

Contributor

Author

@ZailiWang Thank you, that's a good suggestion. I added a note in contribution guide warning against unsafe pickle deserialization and instruct to use `safe_pickle` instead.

 1

 **ShangmingCai** reviewed [3 weeks ago](#)

View reviewed changes



ShangmingCai left a comment

Collaborator

Thx for the PR. Security is important for open-source projects. Please use `pre-commit run --all-files` to fix lint.

To avoid performance degradation, could you provide some numbers? It will help us to know how much this negligible impact would be.

[fix: lint fixes for CVE patch \(isort ordering, remove unused pickle i...](#)✖ [1f9b17a](#)

...

zwang86 commented [2 weeks ago](#)

Contributor

Author

Hi @ShangmingCai , thank you for the feedback, the lint issue is fixed.

For performance, I ran a simple benchmark with `pickle.loads()` vs `safe_pickle_loads()` (100K iterations each):

Payload	Size	<code>pickle.loads</code>	<code>safe_pickle_loads</code>	Overhead
small dict ({"method": "ping"})	31 B	189 ns	512 ns	+323 ns (+170%)
medium dict (batch params + 128 tokens)	393 B	1,746 ns	2,139 ns	+393 ns (+22%)
large list (1000 ints)	2,760 B	11,122 ns	11,580 ns	+458 ns (+4.1%)
nested structure (50 requests)	1,987 B	9,578 ns	10,076 ns	+497 ns (+5.2%)

- The absolute overhead is a fixed ~300-500 ns per deserialization call, regardless of payload size.
- For realistic payloads (medium/large), the relative overhead is only 4-22%.
- The small dict shows +170% relative overhead because the base cost is already tiny (189 ns).
- Compared to ZMQ network round-trip (~ms) and GPU inference (~ms to seconds), this overhead is completely negligible.
- These deserialization paths are not on the inference hot path (model forward); they handle control-plane messages between scheduler/broker components.

  **zwang86** requested a review from **ShangmingCai** [2 weeks ago](#)

 **ShangmingCai** reviewed [2 weeks ago](#)

[View reviewed changes](#)

scripts/playground/replay_request_dump.py

Outdated

 Show resolved

 **ShangmingCai** approved these changes [2 weeks ago](#)

[View reviewed changes](#)



ShangmingCai left a comment

Collaborator

LGTM. Although I think the inference service is typically deployed in a data center with a private/guarded network (the security issues are not as severe as depicted in the CVE), we can merge this PR once the nit has been addressed. I agree that nano latency is negligible. If future performance regression has been reported, we can discuss together to find a better solution. Thank you so much for the fix.

 1



[fix: use context manager for file handle in replay_request_dump.py](#)

 [c79e8c4](#)

ShangmingCai commented [2 weeks ago](#)

Collaborator

/tag-and-rerun-ci

 1



 **github-actions**  added the  label [2 weeks ago](#)

ping1jing2 commented 2 weeks ago

Collaborator

/rerun-failed-ci



Fridge003 and others added 5 commits 2 weeks ago

- Merge branch 'main' into fix/cve-2026-3059-3060-3989 ✖ 4ff19b5
- Merge branch 'main' into fix/cve-2026-3059-3060-3989 ✖ af4154e
- Merge branch 'main' into fix/cve-2026-3059-3060-3989 ✖ 675b7a1
- Merge branch 'main' into fix/cve-2026-3059-3060-3989 ✖ 8efb0b9
- Merge branch 'main' into fix/cve-2026-3059-3060-3989 ✖ 211f9ff

kpham-sgl requested changes 2 weeks ago

[View reviewed changes](#)

kpham-sgl left a comment

Collaborator

@zwang86 Unfortunately I think we have to use msgpack or similar unpack method to fully resolve these CVEs. Do you want to work on it together?

```
python/sglang/multimodal_gen/runtime/scheduler_client.py Outdated
27 | 28 |           # 1. Receive a request from an offline client
28 | 29 |           payload = await socket.recv()
29 | -   |           request_batch = pickle.loads(payload)
   | 30 | +   |           request_batch = safe_pickle_loads(payload)
```

kpham-sgl 2 weeks ago

Collaborator

SafeUnpickler is unfortunately still bypassable no matter how tight we control the allow/deny list. It will become a game of cat and mouse :(if we try to expand the allow/deny list.

Here is one example of a bypass from Claude `getattr(__import__("os"), "system")("malicious command")`



zwang86 [2 weeks ago](#)

Contributor Author

hi [@kpham-sgl](#) I'd love to help with that. The reason I use `SafeUnpickler` is to try the minimal effort fix, but you're right that `SafeUnpickler` is bypassable. To use `msgpack`, we may need different approach for each component, some contains fields like `torch.Tensor` that not supported by `msgpack`. I can break it down to multiple PR/stages.



zwang86 [2 weeks ago](#)

Contributor Author

I believe for the other [CVE-2026-3059/3060](#), we can use other unpack method to fully solve the issues. But for [CVE-2026-3989](#), `replay_request_dump.py` is reading existing `.pk1` files, so I feel using `SafeUnpickler` would be the only feasible option unless we change the dump format. It's also a local playground script (not a server component), so the attack vector is very narrow — an attacker would need to place a malicious `.pk1` file on a developer's machine and wait for them to manually run the script. If this makes sense, I can keep the `SafeUnpickler` fix for `replay_request_dump.py` in this PR, revert the others, and work on a `msgpack`-based approach for [CVE-2026-3059/3060](#) in follow-up PRs.



kpham-sgl [2 weeks ago](#)

Collaborator

If this makes sense, I can keep the `SafeUnpickler` fix for `replay_request_dump.py` in this PR, revert the others, and work on a `msgpack`-based approach for <https://github.com/advisories/GHSA-rgq9-fqf5-fv58/3060> in follow-up PRs.

I think this makes sense. Can you also add a warning to user to tell them to load from only trusted files?



kpham-sgl [2 weeks ago](#)

Collaborator

Also can you update the PR title and description to state that we are only resolving [GHSA-hvwj-8w5g-28rg](#)?

kpham-sgl commented [2 weeks ago](#)

Collaborator

[@zwang86](#) Unfortunately I think we have to use `msgpack` or similar unpack method to fully resolve these CVEs. Do you want to work on it together?

Another alternative is to sign with HMAC encryption (similar to [TensorRT LLM](#)). This is probably the cleanest approach

zwang86 commented [2 weeks ago](#)

Contributor

Author

@kpham-sgl yes I actually considered HMAC, it's feasible but we will need to handle key generation/distribution/rotation and update all sender/receiver, so it won't be simpler than use `msgpack`. The advantage is we don't need to rewrite serialization logics. I'd prefer use `msgpack` since it prohibits RCE from the root.

Anyway either approach is not for [CVE-2026-3989](#) (`replay_request_dump.py`), I feel we can rewrite the serialization with `msgpack` in followup PR, only update `replay_request_dump.py` and docs in this one. Let me know if you think this make sense. Thank you!

kpham-sgl commented [2 weeks ago](#)

Collaborator

@kpham-sgl yes I actually considered HMAC, it's feasible but we will need to handle key generation/distribution/rotation and update all sender/receiver, so it won't be simpler than use `msgpack`

Afaict TensorRT is generating the key **once** during server start, put it in an env var, and distribute that key (the env var) to participating nodes. I think we can do something similar for SGLang. This should also solves the RCE at root (the attackers have to somehow be able to read the HMAC key stored in the env var)

I prefer this approach over `msgpack` because we can later easily fix zmq method like `send_pyobj` and `recv_pyobj` (not directly relevant to the CVEs but still a security risk) like

https://github.com/NVIDIA/TensorRT-LLM/blob/ec909660ff02923a7856e38543d009e5c9021613/tensorrt_llm/executor/ipc.py#L4.

Let's discuss this in slack [@zwang86](#)



kpham-sgl self-assigned this [2 weeks ago](#)



[refactor: revert CVE-2026-3059/3060 changes, keep only CVE-2026-3989 fix](#) [✖ 6955832](#)



zwang86 changed the title [fix\(security\): replace unsafe pickle.loads with SafeUnpickler for CVE-2026-3059/3060/3989](#) [fix\(security\): replace unsafe pickle.loads with SafeUnpickler for CVE-2026-3989](#) [2 weeks ago](#)

kpham-sgl commented [2 weeks ago](#)

Collaborator

@zwang86 sorry whats your slack account? Can you respond to me under this slack thread 😊
<https://sgl-fru7574.slack.com/archives/C08FWFM82T1/p1774501081914049>

zwang86 commented [2 weeks ago](#)

Contributor

Author

Hi **@kpham-sgl**, sorry I think I talked to the wrong person/ account (I sent DM to Khoa Pham with radixark email). Just send you a DM.

✓ **kpham-sgl** approved these changes [2 weeks ago](#)

[View reviewed changes](#)

kpham-sgl left a comment

Collaborator

LGTM! Thanks **@zwang86**

🔗 Merge branch 'main' into [fix/cve-2026-3059-3060-3989](#)

✗ [7af6793](#)

🏷️ **Fridge003** added the **high priority** label [2 weeks ago](#)

🔗 **Fridge003** merged commit **5fc5c18** into [sgl-project:main](#) [2 weeks ago](#)
512 of 566 checks passed

[View details](#)

🔗 **Fridge003** pushed a commit that referenced this pull request [2 weeks ago](#)

fix(security): replace unsafe pickle.loads with SafeUnpickler for CVE... ✗ [9b0c470](#)

...

avioligo commented [2 weeks ago](#)

@ShangmingCai @kpham-sgl @ByronHsu @hnyls2002 @mickqian @yhyang201 @ping1jing2


Please see following NEW security advisories ASAP

1. <https://github.com/sgl-project/sglang/security/advisories/GHSA-29hj-p36c-7jgg>
2. <https://github.com/sgl-project/sglang/security/advisories/GHSA-chqp-x9rv-cwj4>

This is urgent. Thank you.

 **satyamk7054** pushed a commit to satyamk7054/sglang that referenced this pull request 5 days ago

 `fix(security): replace unsafe pickle.loads with SafeUnpickler for CVE...` [2a86faa](#)
...

 **JustinTong0323** pushed a commit to JustinTong0323/sglang that referenced this pull request 18 hours ago

 `fix(security): replace unsafe pickle.loads with SafeUnpickler for CVE...` [3427b24](#)
...

[Sign up for free](#) to join this conversation on **GitHub**. Already have an account? [Sign in to comment](#)

Reviewers

-  **ShangmingCai** ✓
-  **kpham-sgl** ✓
-  **ByronHsu** ●
-  **hnyls2002** ●
-  **mickqian** ●
-  **yhyang201** ●
-  **ping1jing2** ●

Assignees

-  **kpham-sgl**

Labels

- diffusion documentation high priority run-ci

Projects

None yet

Milestone

No milestone

Development

Successfully merging this pull request may close these issues.

None yet

7 participants

