PR #51796 完整报告

vllm-project/vllm

[Bugfix] Reject NUL byte in structured_outputs.regex

合并时间：2026-08-14 08:08

原文链接：http://prhub.com.cn/vllm-project/vllm/pull/51796

执行摘要

- 一句话：拒绝 regex 中的 NUL 字节并返回 HTTP 400
- 推荐动作：这是一个小而清晰的修复，适合快速浏览。值得关注的是其设计模式：在请求校验阶段提前拒绝、并在后端入口做纵深防御，这种双层校验思路可复用于其他边界输入校验场景。不需要精读。

功能与动机

PR body 明确指出：'A NUL byte is never meaningful in a regex pattern and is not handled by the regex-to-grammar conversion. Currently a structured_outputs.regex containing a NUL is passed through to the backend instead of being rejected.' 目的是避免无效输入在后端选择后才被发现，导致 auto 模式静默 fallback 或 xgrammar 原生转换异常。

实现拆解

变更分为三层：

1. 请求级校验前置拒绝：在 vllm/sampling_params.py 的 _validate_structured_outputs 中，于所有后端选择之前新增检查 if self.structured_outputs.regex and "\x00" in self.structured_outputs.regex，命中即抛 VLLMValidationError，从而在默认 auto 模式下也返回 HTTP 400，而非静默回退其他后端。
2. xgrammar 后端纵深防御：在 vllm/v1/structured_output/backend_xgrammar.py 的 validate_xgrammar_grammar 中、调用 xgr.Grammar.from_regex 之前新增相同检查，抛出 ValueError。这样即使请求绕过请求级校验（如直接使用后端 API），NUL 也不会进入 xgrammar 原生代码。
3. 测试覆盖：在 tests/v1/structured_output/test_validation.py 新增参数化测试 test_regex_with_nul_byte_rejected，覆盖单独 NUL、NUL 后跟其他控制字符、嵌入 NUL 三种情况，断言请求校验抛 VLLMValidationError、xgrammar 守卫抛 ValueError。

无配置、schema 或部署配套改动。

关键文件：

- vllm/sampling_params.py（模块 参数校验；类别 source；类型 core-logic；符号 _validate_structured_outputs）：核心修复：在请求校验阶段提前拒绝包含 NUL 字节的 regex，确保所有后端模式下均返回干净的 HTTP 400，避免默认 auto 模式静默回退。

- vllm/v1/structured_output/backend_xgrammar.py（模块 结构化输出；类别 source；类型 core-logic；符号 validate_xgrammar_grammar）：通过在后端验证入口添加 NUL 检查，提供纵深防御，防止 NUL 到达 xgrammar 原生代码。
- tests/v1/structured_output/test_validation.py（模块 测试覆盖；类别 test；类型 test-coverage；符号 test_regex_with_nul_byte_rejected）：为新行为添加测试覆盖，验证请求校验与 xgrammar 守卫双重拒绝。

关键符号：_validate_structured_outputs, validate_xgrammar_grammar, test_regex_with_nul_byte_rejected

关键源码片段

vllm/sampling_params.py

核心修复：在请求校验阶段提前拒绝包含 NUL 字节的 regex，确保所有后端模式下均返回干净的 HTTP 400，避免默认 auto 模式静默回退。

```
# vllm/sampling_params.py —— 请求级校验入口
# 已有空字符串语法校验，避免引擎端崩溃
if (
    isinstance(self.structured_outputs.grammar, str)
    and self.structured_outputs.grammar.strip() == ""
):
    raise VLLMValidationError(
        "structured_outputs.grammar cannot be an empty string"
    )

# 新增：NUL 字节在正则中无意义，且 regex-to-grammar 转换不会处理它。
# 在后端选择前拒绝，保证默认 auto 模式也返回干净的 HTTP 400，
# 而不是静默回退到其他后端。
if self.structured_outputs.regex and "\x00" in self.structured_outputs.regex:
    raise VLLMValidationError(
        "structured_outputs.regex must not contain a NUL character ('\\x00')"
    )

# 之后才按 backend 配置进入各后端校验
from vllm.v1.structured_output.backend_xgrammar import validate_xgrammar_grammar
if backend.startswith("xgrammar"):
    validate_xgrammar_grammar(self)
# …… 其他后端分支省略 ……
```

vllm/v1/structured_output/backend_xgrammar.py

通过在后端验证入口添加 NUL 检查，提供纵深防御，防止 NUL 到达 xgrammar 原生代码。

```
# vllm/v1/structured_output/backend_xgrammar.py —— xgrammar 后端校验入口
def validate_xgrammar_grammar(sampling_params: SamplingParams) -> None:
    # 校验请求是否支持结构化输出，不支持时抛出 ValueError
    if sampling_params.structured_outputs is None:
        return
```

```
    so_params = sampling_params.structured_outputs

    if so_params.regex:
        # 新增：NUL 字节无法被 xgrammar 原生 regex 转换器处理，
        # 必须在进入 from_regex 前拒绝（原 try/except 不覆盖此情况）
        if "\x00" in so_params.regex:
            raise ValueError(
                "structured_outputs.regex must not contain a NUL character ('\\x00')"
            )
        try:
            compile_regex_with_timeout(
                xgr.Grammar.from_regex,
                so_params.regex,
            )
        except Exception as err:
            raise ValueError(
                f"Failed to transform regex into a grammar: {err}"
            ) from err
    # …… choice / json 分支省略 ……
```

tests/v1/structured_output/test_validation.py

为新行为添加测试覆盖，验证请求校验与 xgrammar 守卫双重拒绝。

```
# tests/v1/structured_output/test_validation.py —— 新增 NUL 字节拒绝测试
@pytest.mark.parametrize(
    "regex",
    [
        "\x00", # 单个 NUL
        "\x00\x01\x02\x1f", # NUL 后跟其他控制字符
        "[0-9]\x00", # 嵌入式 NUL
    ],
)
def test_regex_with_nul_byte_rejected(regex):
    # NUL 字节在结构化输出正则中无意义，且 xgrammar 原生转换器不处理它。
    # 它必须在所有后端模式下于请求校验阶段被拒绝（返回干净的 400），
    # 而不是到达原生代码，或在默认 auto 模式下静默回退到其他后端。
    params = SamplingParams(
        structured_outputs=StructuredOutputsParams(regex=regex)
    )

    # 后端选择前即被拒绝，因此 auto 模式也会返回 400
    with pytest.raises(VLLMValidationError, match="NUL"):
        params._validate_structured_outputs(
            _StubModelConfig(is_diffusion=False),
            StructuredOutputsConfig(),
            tokenizer=object(),
        )

    # xgrammar 后端也直接拒绝（纵深防御），避免 NUL 进入原生 from_regex
```

```
from vllm.v1.structured_output.backend_xgrammar import validate_xgrammar_grammar

with pytest.raises(ValueError, match="NUL"):
    validate_xgrammar_grammar(params)
```

评论区精华

本 PR 无实质性的 review 评论或讨论线程。claude[bot] 自动审查说明来自 fork 的 PR 不能自动审查；维护者 vadiklyutiy 直接批准。没有发现设计争议或未解决疑虑。

- 暂无高价值评论线程

风险与影响

- 风险：
  1. 该检查假定 structured_outputs.regex 是字符串；若其他调用方传入非字符串序列类型，in 运算符可能抛出 TypeError 而非预期的 VLLMValidationError。不过 HTTP 解析路径中 StructuredOutputsParams 通常会强制字符串，风险较低。
  2. backend_xgrammar.py 中的 NUL 检查位于原有 try/except 之外，错误消息不带 Failed to transform regex into a grammar: 前缀，错误格式略有变化，但不影响正常拒绝流程。
  3. 仅拒绝含 NUL 的正则，合法正则行为不受影响；额外开销为对正则的一次线性扫描，性能影响可忽略。 - 影响：用户：此前携带 NUL 正则的请求可能收到 500 错误或静默回退结果，现在会得到清晰可操作的 HTTP 400 错误。系统：验证层增加一个低成本字符串检查，不影响运行路径。团队：没有任何维护负担，后续可在同一模块继续补充类似输入校验。
- 风险标记：正则类型假定为字符串，默认 auto 模式行为变化

关联脉络

- PR #50595 [Bugfix][Structured Output] Mask request stop tokens in xgrammar until grammar terminates: 同一结构化输出验证功能线的先前修复，聚焦 xgrammar 相关输入边界处理，与本 PR 的请求校验改进方向一致。