

# PR #55451 完整报告

vllm-project/vllm

[Bugfix] Unreadable `prompt\_embeds` payload should be a 400, not a 500

合并时间: 2026-09-16 13:53

原文链接: <http://prhub.com.cn/vllm-project/vllm/pull/55451>

## 执行摘要

- 一句话: `prompt_embeds` 不可解析载荷从 500 改判 400, 错误理由截断到 200 字符
- 推荐动作: 值得精读, 尤其是要把「调用方输入错误 → HTTP 状态码」这条链路做对的工程师。关注三处设计决策: 一是 `except Exception` 取代显式异常元组 (作者本人主动标注为相对已批准版本的语义放宽, 并用 OOM 前置 `re-raise` 控制半径), 这是本次最值得记住的取舍; 二是 `_truncated_reason()` 用「截断 + 省略计数」把反射放大压到常量级且不损失调试信息; 三是测试通过 `patch torch.load` 注入已知长度理由, 把断言从「`torch` 恰好给出长消息」解耦出来——这套写法可以直接搬到其它依赖第三方错误文本的校验代码上。阅读时可顺手确认你所在部署的 5xx 告警基线是否需要随 400 迁移而调整, 并注意 `image_embeds` / `audio_embeds` 路径仍是 500 的不一致。

## 功能与动机

PR body 明确指出这是「客户端输入错误却被报成服务端故障」的漏洞: `prompt_embeds` 是调用方自己写的 base64 blob, `safe_load_prompt_embeds` 对其它所有写错方式都已经回答 400 (rank 错、`hidden_size` 错、非浮点 dtype、反序列化出来不是 tensor、body 不是合法 base64), 唯独「`torch.load` 完全读不了」这个洞返回 500。作者用表格逐一列举了原因: 空载荷 `EOFError`、单字节 `IndexError`、随机字节 `UnpicklingError`、ascii 文本 `KeyError`、截断 zip / 无 tensor 的 zip / legacy pickle 都是 `RuntimeError`, 没有一个是 `ValueError`, 所以 `create_error_response` 落到 `InternalServerError` 分支; 而 `render_prompt` 与路由之间没有 try, 调用方只是把 body 写错就拿到 500。作者还说明这是对 OpenAI 入口调用方字段做 fuzzing、把所有异常过一遍 `create_error_response` 后筛出来的 (同一轮 12 万个多模态 content part 没有 500, 故本 PR 只收敛这一处)。

## 实现拆解

变更入口: `vllm/renderers/embed_utils.py::safe_load_prompt_embeds` 是 `prompt_embeds` 的唯一解析入口 (`render_prompt` 直接调用、中间无 try), 它抛什么异常就决定了 HTTP 状态码, 因此改动全部内聚在该函数与新增的私有 helper 上。

1. 收紧解析边界: `pybase64.b64decode(embed, validate=True)` 从 `torch.load` 的实参位置上提为独立的 payload 局部变量, 并加注释说明「不是合法 base64 时仍抛 `binascii.Error`, 它是 `ValueError` 子类、本来就映射 400」——把已经正确的分支挡在新加的 try 之外, 避免被新捕获逻辑改语义。

2. 按行为分类 `torch.load` 的失败：在 `check_sparse_tensor_invariants_threadsafe()` 上下文内包 `try/except`。第一个分支 `except (torch.OutOfMemoryError, MemoryError):` `raise` 保证服务端 OOM 仍是 500 (`torch.OutOfMemoryError` 是 `RuntimeError` 子类，不先拦会被下面的宽泛捕获吃掉)；第二个分支 `except Exception as exc` 转为 `VLLMValidationError(parameter="prompt_embeds")`。注释列出 `torch` 的 6 类异常，解释「按行为而非按类型捕获」同时获得对 `torch` 版本变化的鲁棒性。
3. 让错误信息有界：新增 `_MAX_EMBED_ERROR_REASON_CHARS: Final = 200` 与 `_truncated_reason(exc)`，超限时返回前 200 字符加 ... (N more characters truncated)，未超限原样透传。动机是 `torch` 的理由由调用方字节构造 (unpickler 会引用调用方选择的名称)，不设上限等于原样回显。
4. 测试配套 (新增)： `tests/renderers/test_prompt_embeds_payload_validation.py` (+163) 定义 `UNPARSABLE_PAYLOADS` 覆盖 7 类载荷； `test_unreadable_prompt_embeds_payload_is_a_client_error` 断言 `parameter == "prompt_embeds"` 且经真实 `create_error_response` 得到 400，钉的是 HTTP 码而非异常类； `test_the_error_body_stays_bounded` 在 100 KB padding 下断言消息 < 500 字符；两条截断规则测试用 `patch("vllm.renderers.embed_utils.torch.load", side_effect=RuntimeError(reason))` 注入已知长度理由； `test_a_well_formed_payload_still_loads` 与 `test_a_payload_that_is_not_base64_is_still_a_client_error` 作为正向对照。
5. 既有测试适配： `tests/renderers/test_sparse_tensor_validation.py` (4 处断言) 与 `tests/renderers/test_sparse_tensor_concurrent_race.py` (5 处断言) 把 `prompt-embeds` 载荷的期望从 (`RuntimeError`, `ValueError`) 改为 `VLLMValidationError`；同一文件里 `ImageEmbeddingMediaIO` / `AudioEmbeddingMediaIO` 的断言刻意不动，因为 `MediaIO` 路径仍抛 `torch` 的 `RuntimeError`，本 PR 不覆盖。
6. 构建与流程配套： `pre-commit` 的 typos (`UNPARSEABLE` → `UNPARSABLE`) 与 `check-forbidden-imports` (源码与测试各一处 `import pickle`) 迫使移除 `pickle` 依赖，间接把异常匹配从显式元组改成 `except Exception`；CI 侧因分支落后 `main` 两次被拒、并出现 3 个 job 同一秒 `exit status 94` 的基础设施抖动，靠多次 `merge main` 与 `/ci retry` 收敛。

关键文件：

- `vllm/renderers/embed_utils.py` (模块 嵌入解析；类别 `source`；类型 `core-logic`；符号 `_MAX_EMBED_ERROR_REASON_CHARS`, `_truncated_reason`, `safe_load_prompt_embeds`)：唯一源码改动：把 `torch.load` 的不可读载荷按行为捕获并转成 `VLLMValidationError`，新增 `_MAX_EMBED_ERROR_REASON_CHARS` 与 `_truncated_reason()`，把 `base64` 解码移出 `try`，并先 `re-raise` OOM 类异常，直接决定客户端拿到 400 还是 500。
- `tests/renderers/test_prompt_embeds_payload_validation.py` (模块 嵌入校验测试；类别 `test`；类型 `test-coverage`；符号 `model_config`, `_encode`, `_zip_without_a_tensor`, `_saved_tensor`)：新增 163 行测试是本次「回归已被证明」的核心证据：7 类不可读载荷全部断言 `parameter == "prompt_embeds"` 且经真实 `create_error_response` 得到 400，另有响应体长度上限与两条确定性截断规则测试，以及两个正向对照。

- tests/renderers/test\_sparse\_tensor\_validation.py (模块 稀疏校验测试; 类别 test; 类型 test-coverage; 符号 test\_malicious\_sparse\_tensor\_rejected, test\_extremely\_large\_indices\_rejected, test\_negative\_indices\_rejected, test\_attack\_scenario\_completions\_api) : 4 处 prompt-embeds 断言从 (RuntimeError, ValueError) 改为 VLLMValidationError, 记录并固定了恶意稀疏张量拒绝路径的状态码由 500 变 400; 同文件 MediaIO 断言保持不变, 明确划出本次未覆盖的范围。
- tests/renderers/test\_sparse\_tensor\_concurrent\_race.py (模块 并发竞态测试; 类别 test; 类型 test-coverage; 符号 test\_malicious\_sparse\_rejected\_by\_prompt\_loader, test\_concurrent\_loads\_all\_reject\_invalid, test\_flag\_restored\_after\_exception, test\_flag\_restored\_after\_concurrent\_exceptions) : 5 处断言与 contextlib.suppress 的异常元组补上 VLLMValidationError, 保证并发竞态、全局 flag 恢复与跨 loader 锁共享等场景在状态码语义变化后仍然有效。

关键符号: safe\_load\_prompt\_embeds, \_truncated\_reason, create\_error\_response, test\_unreadable\_prompt\_embeds\_payload\_is\_a\_client\_error, test\_the\_error\_body\_stays\_bounded, test\_a\_long\_reason\_is\_truncated\_and\_says\_how\_much\_was\_left\_out, test\_a\_short\_reason\_is\_passed\_through\_whole, test\_a\_well\_formed\_payload\_still\_loads, test\_a\_payload\_that\_is\_not\_base64\_is\_still\_a\_client\_error

## 关键源码片段

### vllm/renderers/embed\_utils.py

唯一源码改动: 把 torch.load 的不可读载荷按行为捕获并转成 VLLMValidationError, 新增 \_MAX\_EMBED\_ERROR\_REASON\_CHARS 与 \_truncated\_reason(), 把 base64 解码移出 try, 并先 re-raise OOM 类异常, 直接决定客户端拿到 400 还是 500。

```
# vllm/renderers/embed_utils.py
```

```
# torch 的理由由调用方发送的字节构成, 所以这里给它设上限, 而不是整段回显。
_MAX_EMBED_ERROR_REASON_CHARS: Final = 200
```

```
def _truncated_reason(exc: Exception) -> str:
```

```
    """取 exc 的消息并加上限, 被截断时说明省略了多少字符。
```

```
    没有计数时, 截断后的理由读起来和完整理由一样, 调试大载荷的调用方
    就无法知道 torch 其实说得更多。
```

```
    """
```

```
    reason: Final = str(exc).strip() or type(exc).__name__
```

```
    omitted: Final = len(reason) - _MAX_EMBED_ERROR_REASON_CHARS
```

```
    if omitted <= 0:
```

```
        return reason # 没超限就原样透传, 不附加任何标注
```

```
    return (
```

```
        f"{reason[:_MAX_EMBED_ERROR_REASON_CHARS]}"
```

```
        f"... ({omitted} more characters truncated)"
```

```
)
```

```

def safe_load_prompt_embeds(
    model_config: "ModelConfig",
    embed: bytes,
) -> torch.Tensor:
    if not model_config.enable_prompt_embeds:
        raise VLLMValidationError(
            "You must set `--enable-prompt-embeds` to input `prompt_embeds`.",
            parameter="prompt_embeds",
        )

    # base64 解码放在 try 之外：非法 base64 抛 binascii.Error，它是
    # ValueError 子类，entrypoints 的兜底分支本来就映射成 400，语义不变。
    payload: Final = pybase64.b64decode(embed, validate=True)

    with check_sparse_tensor_invariants_threadsafe():
        try:
            tensor = torch.load(
                BytesIO(payload),
                weights_only=True,
                map_location=torch.device("cpu"),
            )
        except (torch.OutOfMemoryError, MemoryError):
            # 内存耗尽是服务端状况、不是调用方的错，必须先于下面的宽泛
            # 捕获重新抛出（前者是 RuntimeError 子类，不先拦就会被吃掉）。
            raise
        except Exception as exc:
            # torch 对「这不是一个 tensor 文件」没有统一的异常类型：
            # 空载荷 EOFError、单字节 IndexError、随机字节 UnpicklingError、
            # ascii 文本 KeyError、截断 zip / 无 tensor 的 zip RuntimeError。
            # 它们都不是 ValueError，所以以前全部落到 500。按行为而非按
            # 类型捕获，也能扛住 torch 更换具体异常类型。
            #
            # torch 的理由有价值（非法稀疏张量会点出越界索引），但它由
            # 调用方的字节构造，因此截断后再回显。
            raise VLLMValidationError(
                "`prompt_embeds` could not be deserialized as a torch tensor: "
                f"{_truncated_reason(exc)}",
                parameter="prompt_embeds",
            ) from exc
        tensor = safe_to_dense(tensor, parameter="prompt_embeds")

    # 之后的 rank / hidden_size / dtype 校验保持不变，仍各自抛
    # VLLMValidationError，因此全部统一为 400。

```

### tests/renderers/test\_prompt\_embeds\_payload\_validation.py

新增 163 行测试是本次「回归已被证明」的核心证据：7 类不可读载荷全部断言 `parameter == "prompt_embeds"` 且经真实 `create_error_response` 得到 400，另有响应体长度上限与两条确定性截断规则测试，以及两个正向对照。

```
# tests/renderers/test_prompt_embeds_payload_validation.py
```

```
# 七类「torch.load 完全读不了」的载荷；断言只要求 VLLMValidationError,  
# 因此对 torch 具体抛哪种异常不敏感。
```

```
UNPARSABLE_PAYLOADS = [  
    pytest.param(b"", id="empty"), # EOFError  
    pytest.param(b"\x80", id="single-byte"), # IndexError  
    pytest.param(b"\x00\x01\x02\x03not-a-tensor", id="random-bytes"),  
    pytest.param(b"hello world this is not a torch file", id="ascii-text"),  
    pytest.param(b"PK\x03\x04" + b"\x00" * 20, id="truncated-zip"),  
    pytest.param(_zip_without_a_tensor(), id="zip-without-a-tensor"),  
    # pickle.dumps(42): 合法 pickle 流但不是 torch checkpoint。  
    # 直接写成字面量，避免本文件 import pickle（被 pre-commit 禁止）。  
    pytest.param(b"\x80\x04K*.", id="legacy-pickle-of-an-int"),  
]
```

```
@pytest.mark.parametrize("payload", UNPARSABLE_PAYLOADS)  
def test_unreadable_prompt_embeds_payload_is_a_client_error(model_config, payload):  
    with pytest.raises(VLLMValidationError) as excinfo:  
        safe_load_prompt_embeds(model_config, _encode(payload))  
  
    assert excinfo.value.parameter == "prompt_embeds"  
    # 断言经过真实的 create_error_response，钉住的是 entrypoints 实际  
    # 产出的 HTTP 状态码，而不只是异常类型。  
    assert int(create_error_response(excinfo.value).error.code) == 400
```

```
@pytest.mark.parametrize("payload", UNPARSABLE_PAYLOADS)  
def test_the_error_body_stays_bounded(model_config, payload):  
    # torch 的理由由调用方字节构造，追加 100 KB padding 后整体必须仍然小。  
    padding = b"A" * 100_000  
    with pytest.raises(VLLMValidationError) as excinfo:  
        safe_load_prompt_embeds(model_config, _encode(payload + padding))  
  
    assert len(str(excinfo.value)) < 500
```

```
def test_a_long_reason_is_truncated_and_says_how_much_was_left_out(model_config):  
    """截断规则用已知长度的理由钉住，而不是依赖某个真实载荷。
```

```
    哪些 torch 消息会超过上限是 torch 自己的事、且随版本变化，所以这里  
    注入理由，让断言针对本代码的截断规则，而非已安装 torch 的错误文本。  
    """
```

```
    reason = "E" * (_MAX_EMBED_ERROR_REASON_CHARS + 37)
```

```
    with (  
        patch(  
            "vllm.renderers.embed_utils.torch.load", side_effect=RuntimeError(reason)
```

```

    ),
    pytest.raises(VLLMValidationError) as excinfo,
):
    safe_load_prompt_embeds(model_config, _encode(b"ignored"))

    message = str(excinfo.value)
    assert "E" * _MAX_EMBED_ERROR_REASON_CHARS in message
    assert "E" * (_MAX_EMBED_ERROR_REASON_CHARS + 1) not in message
    assert "... (37 more characters truncated)" in message

```

## 评论区精华

1. 截断信息要告诉读者省了多少：DarkLight1337 直接给出修改指令——「Let's add an indication of how many characters were truncated if the error length exceeds the max value」。作者在 7bdc391f3 抽出 `_truncated_reason()` 落地，并强调「the count only shows up when it is telling the reader something」，同时补两条单测；coderabbitai 确认该线程已解决 (🔗 Review thread resolved)。
  2. 测试确定性的交锋：coderabbitai 指出「appending padding does not ensure that the reason contains the appended bytes」，只用长度断言的测试可能在一段自然就很短的理由上通过；作者接受并以 `patch torch.load` 注入已知长度理由重写，同时保留真实载荷上的端到端 < 500 字符断言。
  3. 捕获范围放宽需要维护者复签：作者在 pre-commit 修复后主动声明「This is a semantic widening from the version you approved, so please re-check that you are happy with it; the narrow tuple plus an allowlist entry is the alternative if you would rather」——即显式元组 + pickle 白名单 vs `except Exception + OOM` 前置 re-raise 的取舍，DarkLight1337 随后以 APPROVED 回复「Thanks」确认接受。
  4. 稀疏张量路径状态码变更被单独验证：Manny7717 在 base e473e903 / head 1b5abf8d 上本地复现，「14/14 FAIL on base / all pass at head」，并确认「The reason string is bounded to 200 chars — good call, since torch's unpickler error text quotes caller-chosen names (reflection bound)」，以及「confirmed subclass of `RuntimeError` → genuinely server-side condition stays a 500」。
- 截断后的错误理由要说明省略了多少字符 (design): 建议被采纳并落地为独立 helper，规则集中在一处，由两条单测固定。
  - 截断测试的确定性：注入理由而非依赖 padding (testing): 测试从「torch 恰好给出长消息」解耦，断言对象变为本代码的截断规则。
  - 捕获范围由显式异常元组放宽为 `except Exception`，需要维护者复签 (correctness): DarkLight1337 随后 APPROVED (回复「Thanks」) 接受放宽；风险由 OOM 前置 re-raise、捕获点限于单次 `torch.load` 以及行为式捕获对 torch 版本变化的鲁棒性共同缓解。
  - 恶意稀疏张量拒绝路径的状态码由 500 变 400 (correctness): 评审接受，PR body 第 1 条显式标注该副作用；MediaIO 路径的不一致被明确排除在本次范围外。
  - CI: 分支落后 main 与 exit status 94 的基础设施抖动 (other): 判定为基础设施抖动，作者发起 `/ci retry` 重试 3 个失败 job。

## 风险与影响

- 风险:

1. 捕获范围从显式异常元组放宽到 `except Exception (vllm/renderers/embed_utils.py)`  
: 任何在这一次 `torch.load` 内发生的非 OOM 异常都会被改判为 400, 包括 `torch` 自身 bug、反序列化期的内部错误等真正的服务端故障, 这类情况的 5xx 可观测性会被削弱、告警阈值可能失准。缓解点是捕获点被严格限制在单次 `torch.load` 调用内, 且 `torch.OutOfMemoryError / MemoryError` 已前置 `re-raise`, `KeyboardInterrupt / SystemExit` 因继承 `BaseException` 不受影响。
2. HTTP 语义变更 (500 → 400): 不仅 `prompt_embeds` 的不可读载荷, 恶意稀疏张量的拒绝路径也由 500 变 400 (PR body 第 1 条自陈)。依赖 5xx 触发重试的客户端将不再重试 (符合预期), 但监控看板会看到 5xx 下降、4xx 上升的跳变, 需要同步到告警基线与客户端错误处理文档。
3. 同构路径行为不一致: `ImageEmbeddingMediaIO / AudioEmbeddingMediaIO` 对同类不可读载荷仍抛 `torch` 的 `RuntimeError` → 500, 本 PR 明确划出范围外 (`tests/renderers/test_sparse_tensor_validation.py` 里 `MediaIO` 断言保持 (`RuntimeError, ValueError`))。同一个调用方在 `prompt_embeds` 与 `image_embeds` 上会得到不同的状态码, 未来大概率被再次报告。
4. 错误体回显调用方字节: 响应体包含 `torch` 的理由文本, 其中可能引用调用方提供的名字, 现已截断到 200 字符 + 计数后缀, 整体长度由 `test_the_error_body_stays_bound ed (< 500 字符)` 兜底, 反射放大风险被压到常量级; 但对把错误信息直接落日志或渲染的部署方, 仍是调用方可控字符串进入下游链路。
5. 测试对 `torch` 行为的隐式依赖: `UNPARSABLE_PAYLOADS` 的 7 类载荷依赖 `torch` 当前的解析路径, 断言只要求 `VLLMValidationError` 所以较稳; 若未来某个载荷变成「能读」, 该用例会失败, 属可接受的回归信号而非脆弱点。
6. CI 层面: Buildkite #89242 出现 3 个 job 同一秒 `exit status 94` (非 `pytest` 退出码) 的疑似基础设施抖动, 其中 `entrypoints-integration-mu` 当时未重试, 合并时 `buildkite/ci/pr` 一度为红; 承载本 PR 测试的 `computer-cpu-reasoning-plus-renderers job` 通过。

- 影响:

- 调用方 (用户): 通过 OpenAI 兼容入口传入 `prompt_embeds` 的客户端, 在写错载荷时会拿到 400 和可读的 `torch` 原因 (截断后), 不必再把它当成服务端故障去重试或报障; 同时 4xx 与 5xx 的区分让客户端错误处理逻辑第一次是准确的。
- 系统: `vllm/renderers/embed_utils.py::safe_load_prompt_embeds` 是 `prompt embeds` 的唯一解析入口, 因此所有 `embedding` 输入路径的错误分类口径由此统一; `entrypoints` 的 `create_error_response` 映射规则本身未改动, 改动被限制在「抛什么异常」这一层。生效前提是服务端启用了 `--enable-prompt-embeds` (默认关闭), 影响面可控。
- 可观测性: `prompt_embeds` 与稀疏张量拒绝相关的 5xx 计数会下降、对应 4xx 计数上升, 监控与 SLO 基线需要一次性对账。

- 团队与工程实践：沉淀了两个可复用范式——「按行为而非按类型捕获第三方库异常」与「把调用方字节构造的错误文本截断并标注省略量」；同时把「fuzzing 入口调用方字段 → 用 `create_error_response` 归类 → 只修有 500 的那一处」这套方法带进仓库，后续多模态部分（`_parse_chat_message_content_mm_part`）已被同法扫过并被判定无 500。
- 风险标记：捕获范围放宽为 `except Exception`, HTTP 500→400 语义变更，`image/audio` 嵌入路径仍为 500（行为不一致），错误体回显调用方字节（已截断至 200 字符），CI 基础设施抖动（`exit status 94`）

## 关联脉络

- PR #56505 [Pooling] Cap max-length padding for chunked embeddings: 同属 OpenAI 兼容入口对嵌入类调用方输入做边界收敛与语义修正：那个 PR 把 chunked 嵌入的 padding 上限改为 `min(max_model_len, max_embed_len)`，本 PR 则把 `prompt_embeds` 的解析失败分类与错误体长度收敛，二者都改的是「调用方写错时服务端如何回应」这一层。
- PR #54098 [Bugfix] Format kernel-import errors eagerly so `warning_once` does not retain them: 同为异常对象处理不当引发的隐患：那个 PR 处理 `warning_once` 缓存异常对象导致的内存泄漏，本 PR 处理异常类型与 HTTP 状态的错配并要求错误文本有界，都是「异常 → 对外表现」链路上的卫生修复。