

PR #39035 完整报告

sgl-project/sglang

[Session] Fix session idle timeout after rejected requests

合并时间: 2026-09-12 07:59

原文链接: <http://prhub.com.cn/sgl-project/sglang/pull/39035>

执行摘要

- 一句话: 修复被拒请求续期导致会话永不超时
- 推荐动作: 建议快速阅读: 3 行修复背后是清晰的时序语义修正 (副作用放到成功路径)。
若团队后续在 session 模块增加新的请求形态, 可借鉴该模式, 并建议补一个针对 abort 路径不续期的单元测试, 防止回归。

功能与动机

PR body 明确指出: "Rejected session requests reset the idle timer, so repeated invalid requests can keep a session alive."。当一个客户端反复发送非法请求 (如 streaming 中的 replace、inflight 期间的追加等) 时, 虽然请求被拒绝, 但进入 `create_req()` 仍会刷新空闲时间, 使空闲超时永不触发, session 及其占用的 prefix cache / KV 资源无法回收。作者在 body 中给出受控时钟验证: 10 秒超时下, 拒绝请求把 last active 从 100 刷到 109, 导致 111 秒时仍存活; 修复后保持 100, 111 秒即过期。

实现拆解

1. 变更入口: python/sglang/srt/session/session_controller.py 中的 `Session.create_req()`, 这是所有会话请求 (普通和 streaming) 的创建入口。
2. 移除无条件刷新: 删除 `assert req.session_params is not None` 之后的那行 `self.last_active_time = time.monotonic()`, 使“进入 create_req”不再等同于“会话活跃”。
3. 在接受分支刷新: 在 `elif self.streaming:` 与普通 (非 stream、非 abort) 分支中各加入一行 `self.last_active_time = time.monotonic()`, 与 `_inflight` 置位、`SessionReqNode` 写入 `req_nodes` 共同构成“接受请求”的原子语义。
4. abort 路径不刷新: `if abort:` 分支只调用 `new_req.set_finish_with_abort()`, 不触碰计时器, 因此被拒绝的请求不会续期。
5. 测试与验证: 本 PR 未新增自动化测试。作者用受控时钟 + 10 秒空闲超时验证 6 种拒绝场景由“不超时”变为“按原时间超时”, 并说明 13 个既有 session token-sharing 与 streaming-session 单测全部通过。

关键文件:

- python/sglang/srt/session/session_controller.py (模块 会话管理; 类别 source; 类型 core-logic; 符号 `Session.create_req`): 唯一变更文件, 也是所有会话请求的创建入口; 修复的核心是 `last_active_time` 刷新时机从入口移到接受分支。

关键符号：Session.create_req

关键源码片段

python/sclang/srt/session/session_controller.py

唯一变更文件，也是所有会话请求的创建入口；修复的核心是 `last_active_time` 刷新时机从入口移到接受分支。

```
# Session.create_req 的核心分支：被拒绝的请求不再刷新 idle 计时器。
# 修复前，函数开头会无条件执行 self.last_active_time = time.monotonic(),
# 导致所有进入本函数的请求（包括 abort）都能让会话“续命”。
# 修复后，刷新动作被移到接受分支，abort 路径完全不触碰计时器。
def create_req(self, req, tokenizer, vocab_size, eos_token_ids=None):
    assert req.session_params is not None
    session_params = req.session_params

    last_req_node = None
    last_req = None
    abort = False
    abort_message = ""

    # 前置校验：streaming 会话只允许简单追加；
    # replace / drop_previous_output / offset 等操作会被标记为 abort。
    # （此处省略非 streaming 分支的 replace / rid 查找逻辑）

    # 构造 new_req (TokenizedGenerateReq 等) 的代码省略，
    # 关键在下面的三分支：
    if abort:
        # 请求被拒绝：不刷新 last_active_time，让会话按原 idle 时间回收
        new_req.set_finish_with_abort(abort_message)
    elif self.streaming:
        # streaming 请求被接受：续期并标记 inflight
        self.last_active_time = time.monotonic()
        # 注意：req_nodes 不在此处更新，由 finish_req() 处理
        self._inflight = True
    else:
        # 普通请求被接受：续期并挂载请求节点
        self.last_active_time = time.monotonic()
        new_req_node = SessionReqNode(new_req, last_req_node)
        self.req_nodes[req.rid] = new_req_node

    return new_req
```

评论区精华

本 PR 没有任何 review 评论。作者在 PR body 中自述验证过程：所有 6 种拒绝场景修复前失败、修复后通过，已接受的请求在两种模式下都通过；同时跑通 13 个既有 session 单测及 Ruff、formatting、isort 等检查。另有 `/tag-and-rerun-ci` 评论用于触发 CI。

- 暂无高价值评论线程

风险与影响

- 风险：

1. 核心路径回归风险：create_req() 是所有会话请求的必经入口，虽然改动小，但后续若在函数中新增提前 return 的校验分支，容易漏刷计时器，导致已接受请求的会话提前过期。
2. 测试缺口：本次没有伴随的单元测试；现有 13 个 session 单测未必覆盖“被拒绝请求不续期”这一场景，后续重构可能无感知地回退该修复。
3. 语义约定：刷新点从入口变为分支后，未来开发者需要牢记“只有接受请求才续期”的约定，否则容易在新增接受路径时忘记刷新。
 - 影响：对用户与系统而言，修复了 session 资源不回收的问题：恶意或异常客户端无法再通过重复发送无效请求无限保活会话，空闲超时配置重新生效，session 占用的 prefix cache 与 KV 资源能被及时回收。对正常请求，行为完全不变。对团队而言，改动极小、易于审查，但为 session 模块建立了更清晰的时序语义，值得在后续演进中保持一致。
 - 风险标记：核心路径变更，缺少测试覆盖

关联脉络

- 暂无明显关联 PR