PR #26819 完整报告

sgl-project/sglang

Add the KV-canary perturb modes and PD-disaggregation e2e tests

合并时间：2026-05-31 09:59

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/26819

执行摘要

- 一句话：新增 KV-canary 扰动模式与 PD 拆分端到端测试
- 推荐动作：本 PR 值得精读，因为其扰动设计覆盖了 KV 缓存的三种典型损坏场景（活跃使用、缓存孤立、刚写入），展示了一种利用 stream ordering 保证时序的实现方法；同时 slot_picker 中排除 out_cache_loc 避免与首次写入竞争的考虑值得借鉴。

功能与动机

PR body 指出需要增加 canary 自测扰动模式以及预填充 - 解码拆分的端到端测试，以验证 KV-canary 对真实 KV 篡改的检测能力。扰动模式模拟不同生命周期中的 KV 数据损坏（req_to_token 映射错误、正在使用的 slot、缓存的未使用 slot、以及刚写入的 slot），帮助提前发现缓存一致性问题。

实现拆解

1. 新增 slot_picker.py 基础工具：提供 collect_active_slots 函数从 ForwardBatch 收集当前活跃请求的 req_to_token 槽位，并自动排除 out_cache_loc 以避免写竞争；提供 pick_out_cache_loc_slot 用于 post_forward 扰动选取刚写入的槽位。
2. 新增四种扰动执行模块：req_to_token.py 随机选取一个活跃条目并篡改其 req_to_token 值；real_kv_used.py 针对正在使用的物理槽位进行首字节翻转；real_kv_unused_cache.py 通过遍历 radix 缓存找到孤立槽位进行篡改，仅在 sweep 阶段生效；real_kv_post_forward.py 在 TAIL 内核之后选取 out_cache_loc 中的槽位进行翻转，利用 CUDA stream ordering 保证发生在 canary 哈希写入之后。
3. 扩展 PerturbManager（manager.py）：新增 req_to_token_pool 依赖；实现 perturb() 方法依次调用 req_to_token、real_kv_used、real_kv_unused_cache 扰动；新增 perturb_post_forward() 方法仅在 forward 后触发 real_kv_post_forward 扰动。
4. 新增 PD 拆分端到端测试基础设施：pd_fixture.py 创建 CanaryPDFixture，在单进程中同时启动 prefill 和 decode 两个引擎，使用多组 canary buffer；对应的测试文件（如 test_self_e2e_pd_perturb.py）在 prefill 侧启用扰动，验证 decode 侧检测到 real KV hash 违规而 prefill 侧保持干净。对每种扰动分别有 MHA 和 SWA 的测试参数化。

关键文件：

- python/sglang/srt/kv_canary/perturb/manager.py（模块 扰动管理器；类别 source；类型 core-logic；符号 perturb_post_forward, perturb_req_to_token,

perturb_real_kv_used, perturb_real_kv_unused_cache）：核心调度入口，集成所有扰动模式的调用逻辑。

- python/sglang/srt/kv_canary/perturb/slot_picker.py（模块 槽位选择器；类别 source；类型 core-logic；符号 ReqToTokenEntry, collect_active_slots, pick_out_cache_loc_slot）：基础工具模块，提供活跃槽位收集和排除逻辑，被多种扰动使用。
- python/sglang/srt/kv_canary/perturb/real_kv_unused_cache.py（模块 real KV 扰动；类别 source；类型 core-logic；符号 run, _pick_sweep_slot_for_group, _translate_full_slots_to_swa_slots）：实现 unused_cache 扰动，通过遍历 radix 缓存找到孤立槽位并篡改。
- python/sglang/srt/kv_canary/perturb/real_kv_post_forward.py（模块 real KV 扰动；类别 source；类型 core-logic；符号 run）：实现 post_forward 扰动，在 TAIL 内核之后篡改刚写入的 slot。
- python/sglang/test/kv_canary/pd_fixture.py（模块 测试夹具；类别 test；类型 test-coverage；符号 CanaryPDFixture, setUpClass, send_parallel_short_requests, _captured_log_text）：提供 PD 拆分端到端测试的 fixture，同时启动 prefill 和 decode 引擎。
- test/registered/kv_canary/test_self_e2e_pd_perturb.py（模块 端到端测试；类别 test；类型 test-coverage；符号 _PDPerturbBase, setUpClass, test_p_side_perturb_surfaces_real_kv_hash_violation_on_decode_side, TestPDPerturbMhaFull）：PD 拆分扰动端到端测试，验证 prefill 侧扰动后 decode 侧可检测到违规。

关键符号：PerturbManager.perturb, PerturbManager.perturb_post_forward, PerturbManager.perturb_req_to_token, PerturbManager.perturb_real_kv_used, PerturbManager.perturb_real_kv_unused_cache, PerturbManager.perturb_real_kv_post_forward, slot_picker.collect_active_slots, slot_picker.pick_out_cache_loc_slot, real_kv_unused_cache.run, real_kv_unused_cache._pick_sweep_slot_for_group, real_kv_used.run, real_kv_post_forward.run, req_to_token.run

关键源码片段

python/sglang/srt/kv_canary/perturb/manager.py

核心调度入口，集成所有扰动模式的调用逻辑。

```
# manager.py — PerturbManager 核心调度逻辑
class PerturbManager:
    def __init__(self, *, config, req_to_token_pool, buffer_groups, outer_step_counter_getter,
    swa_window_size=0, sweep_interval=0):
        # 保存依赖：配置、req_to_token 池、buffer 组、步数获取器等
        self._config = config
        self._req_to_token_pool = req_to_token_pool
        self._buffer_groups = buffer_groups
        self._outer_step_counter_getter = outer_step_counter_getter
        self._swa_window_size = swa_window_size
```

```
        self._sweep_interval = sweep_interval
        self._radix_cache = None
        self._warmup_gate = WarmupGate(config=config, outer_step_counter_getter=outer_step_
        counter_getter)

    def attach_radix_cache(self, radix_cache):
        self._radix_cache = radix_cache

    def perturb(self, *, maybe_inaccurate_forward_batch):
        """在每次 forward 前调用，依次应用 req_to_token、real_kv_used、real_kv_unused_cache
        扰动"""
        self.perturb_req_to_token(maybe_inaccurate_forward_batch)
        self.perturb_real_kv_used(maybe_inaccurate_forward_batch)
        self.perturb_real_kv_unused_cache(maybe_inaccurate_forward_batch)

    def perturb_post_forward(self, *, maybe_inaccurate_forward_batch):
        """在 forward 完成后调用，仅触发 real_kv_post_forward 扰动（在 TAIL 内核之后）"""
        self.perturb_real_kv_post_forward(maybe_inaccurate_forward_batch)

    # 各扰动的具体调度方法，分别委托给对应的 run 函数
    def perturb_req_to_token(self, maybe_inaccurate_forward_batch):
        req_to_token.run(maybe_inaccurate_forward_batch=maybe_inaccurate_forward_batch,
        config=self._config, req_to_token_pool=self._req_to_token_pool, warmup_gate=self._
        warmup_gate)

    def perturb_real_kv_used(self, maybe_inaccurate_forward_batch):
        real_kv_used.run(maybe_inaccurate_forward_batch=maybe_inaccurate_forward_batch,
        config=self._config, req_to_token_pool=self._req_to_token_pool, buffer_groups=self._
        buffer_groups, swa_window_size=self._swa_window_size, warmup_gate=self._warmup_
        gate)

    def perturb_real_kv_unused_cache(self, maybe_inaccurate_forward_batch):
        real_kv_unused_cache.run(maybe_inaccurate_forward_batch=maybe_inaccurate_forward_
        batch, config=self._config, buffer_groups=self._buffer_groups, radix_cache=self._radix_
        cache, swa_window_size=self._swa_window_size, sweep_interval=self._sweep_interval,
        outer_step_counter=self._outer_step_counter_getter(), warmup_gate=self._warmup_gate)

    def perturb_real_kv_post_forward(self, maybe_inaccurate_forward_batch):
        real_kv_post_forward.run(maybe_inaccurate_forward_batch=maybe_inaccurate_forward_
        batch, config=self._config, buffer_groups=self._buffer_groups, warmup_gate=self._
        warmup_gate)
```

评论区精华

PR 无 review 讨论。唯一的评论来自 gemini-code-assist[bot] 的配额提示，无关。

- 暂无高价值评论线程

风险与影响

- 风险：所有扰动默认通过概率环境变量（默认 0）禁用，正常用户无影响。若启用，篡改真实 KV 数据可能触发 canary 检测，但若检测逻辑有缺陷（如哈希计算错误、指针映射失效）可能导致漏报或误报。real_kv_post_forward 依赖 stream ordering，若 GPU 调度异常可能导致竞争。PD 拆分测试涉及多进程，存在进程间同步和超时风险。
- 影响：对最终用户透明，无行为变更。对系统增加了内建自测能力，有助于在生产环境或 CI 中提前暴露 KV 缓存一致性问题。对团队增加了测试覆盖，提高对 KV-canary 检测机制的信心。
- 风险标记：默认禁用 ，核心路径变更 ，多进程测试稳定性

关联脉络

- PR #26820 Add a sliding-window-attention divergence reporter for the KV-canary: 同为 KV-canary 增强，新增 SWA divergence reporter；本 PR 的 unused_cache 扰动和 PD 测试也涉及 SWA 场景，两者互补。
- PR #26821 Add periodic KV-canary stats logging and kernel-run-counter health check: 同为 KV-canary 增强，添加统计日志和健康检查；本 PR 增加扰动检测能力，配合 stats 可更全面观察。
- PR #26779 [core] Compute dimensions/return_pooled_hidden_states in ForwardBatch.init_new: 改动了 ForwardBatch 初始化，本 PR 的 slot_picker 依赖 ForwardBatch 字段，可能受其影响。