PR #51174 完整报告

vllm-project/vllm

[ROCm] Work around DeepEP teardown SIGSEGV in MoE test harness

合并时间：2026-08-06 03:44

原文链接：http://prhub.com.cn/vllm-project/vllm/pull/51174

执行摘要

- 一句话：ROCm 下 DeepEP 测试 teardown SIGSEGV 的 workaround
- 推荐动作：该 PR 改动小而聚焦，但提交信息中的根因分析（ROCr Runtime::Unload() 与 GpuAgent::ReleaseResources() 之间的 use-after-free）值得阅读，可作为理解 HIP teardown 时序的案例。适合关注 ROCm 测试稳定性的工程师精读；对于一般读者，了解其 workaround 思路即可，不建议作为长期方案。

功能与动机

PR body 明确说明：在 ROCm 上 test_deep_ep_moe 每个 rank 以 SIGSEGV 退出，且无 traceback；MoE 计算正确，崩溃是 teardown 阶段的 ROCr use-after-free（存在于 HIP 的 atexit handler，上游修复见 https://github.com/ROCm/rocm-systems/pull/6942）。需要在不改变计算逻辑的前提下，让测试能正确报告通过 / 失败信号，同时不影响非 ROCm 平台。

实现拆解

1. 在测试 worker 中引入平台检测：tests/kernels/moe/parallel_utils.py 新增 from vllm.platforms import current_platform 导入，用于在运行时判断是否为 ROCm 平台。
2. 跟踪 worker 退出码：在 _worker_parallel_launch 中新增局部变量 exit_code = 0；当 worker 抛出异常时，打印异常和 traceback 后置 exit_code = 1，再 raise 以便父进程能捕获异常详情。
3. 在 finally 块中针对 ROCm 强制退出：在 torch.distributed.destroy_process_group() 之后，若 current_platform.is_rocm() 为真，则依次 sys.stdout.flush()、sys.stderr.flush() 后调用 os._exit(exit_code)。这样绕过 HIP atexit 处理器中的 use-after-free，同时保留 flush 后的 traceback 输出和退出码语义。
4. 后续移除标记：代码中留有 TODO (Rohan138): Remove after ROCm 10.0 lands，并注释了上游修复 PR 链接，便于基础镜像更新后回退。
5. 无生产代码改动：变更仅涉及测试工具函数，parallel_launch 的对外接口不变，非 ROCm 平台行为与之前完全一致。

关键文件：

- tests/kernels/moe/parallel_utils.py（模块 测试工具；类别 test；类型 test-coverage；符号 _worker_parallel_launch）：该文件是 DeepEP MoE 测试的并行启动工具，本次变更在 _worker_parallel_launch 的 finally 块中针对 ROCm 平台引入 os._exit 绕过 teardown

阶段的 SIGSEGV，是 PR 唯一改动的文件，集中体现了 workaround 的全部逻辑。

关键符号：_worker_parallel_launch

关键源码片段

tests/kernels/moe/parallel_utils.py

该文件是 DeepEP MoE 测试的并行启动工具，本次变更在 _worker_parallel_launch 的 finally 块中针对 ROCm 平台引入 os._exit 绕过 teardown 阶段的 SIGSEGV，是 PR 唯一改动的文件，集中体现了 workaround 的全部逻辑。

```
def _worker_parallel_launch(
    local_rank: int,
    world_size: int,
    world_local_size: int,
    node_rank: int,
    init_method: str,
    worker: Callable[Concatenate[ProcessGroupInfo, P], None],
    *args: P.args,
    **kwargs: P.kwargs,
) -> None:
    rank = node_rank * world_local_size + local_rank
    torch.accelerator.set_device_index(local_rank)
    device = torch.device("cuda", local_rank)
    torch.distributed.init_process_group(
        backend="nccl",
        init_method=init_method,
        rank=rank,
        world_size=world_size,
    )
    barrier = torch.tensor([rank], device=device)
    torch.distributed.all_reduce(barrier)

    exit_code = 0
    try:
        worker(
            ProcessGroupInfo(
                world_size=world_size,
                world_local_size=world_local_size,
                rank=rank,
                node_rank=node_rank,
                local_rank=local_rank,
                device=device,
            ),
            *args,
            **kwargs,
        )
    except Exception as ex:
        print(ex)
```

```
        traceback.print_exc()
        exit_code = 1 # 标记失败，用于 os._exit 传递退出码
        raise
    finally:
        torch.distributed.destroy_process_group()
        if current_platform.is_rocm():
            # 绕过 ROCm teardown use-after-free（HIP atexit handler 中的 SIGSEGV，
            # 上游修复见 https://github.com/ROCm/rocm-systems/pull/6942）。
            # 先 flush 再 os._exit，因为 os._exit 跳过缓冲刷新，以保留失败 traceback。
            # TODO (Rohan138): Remove after ROCm 10.0 lands
            sys.stdout.flush()
            sys.stderr.flush()
            os._exit(exit_code)
```

评论区精华

该 PR 的 review 评论很少：claude[bot] 自动回复说明 fork PR 不启用自动 review；maintainer AndreasKaratzas 直接批准（LGTM），无实质技术辩论。技术决策已在 PR body 和提交信息中充分阐述，没有遗留的未解决疑虑。

- 暂无高价值评论线程

风险与影响

- 风险：风险集中在测试工具层面，不触及生产代码：
  1. os._exit 跳过清理：会绕过 Python 和第三方库的退出钩子，可能导致某些资源（如 NCCL communicator、文件句柄）未释放，但该行为仅发生在 ROCm 平台且仅用于测试进程，影响有限。
  2. 异常信号传播：raise 与 os._exit 组合可能使父进程无法看到原始 traceback（已通过 flush 缓解），且 exit_code 仅区分 0/1，无法传递具体异常类型，但足以判断测试通过与否。
  3. 平台判断依赖：current_platform.is_rocm() 在非 ROCm 环境为 false，行为不变，但若未来平台判断逻辑变更需同步关注。
  4. 上游依赖：若基础镜像提前包含 ROCm 修复，该 workaround 仍会生效（os._exit 无害），但会造成不必要的 teardown 跳过，需按 TODO 及时移除。 - 影响：影响范围限定在 tests/kernels/moe/test_deepep_moe.py 及相关 MoE 测试的 ROCm 运行环境：此前的 28 个失败用例（每个 rank SIGSEGV）将全部转为通过，显著提升 ROCm CI 的稳定性和可观测性。对生产推理代码、非 ROCm 平台、其他测试无影响。团队收益是减少 CI 噪音，并提供了清晰的 root cause 文档，便于后续移除 workaround。 - 风险标记：测试环境 workaround，os._exit 跳过清理，依赖上游修复

关联脉络

- PR #51173 [ROCm][CI] Keep rocprofiler-sdk out of DeepEP HT MoE test workers: 同为 DeepEP MoE 测试在 ROCm 下的 CI 稳定性修复，与本 PR 修改同一测试文件（tests/kernels/moe/parallel_utils.py 被 #51173 间接关联），二者共同解决 ROCm 环境下

的测试崩溃类问题。