PR #27190 完整报告

sgl-project/sglang

[Fix] Emulate PDEATHSIG on macOS to prevent orphaned worker processes

合并时间：2026-06-11 08:43

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/27190

执行摘要

- 一句话：使用 kqueue 模拟 PDEATHSIG，修复 macOS 孤儿工作进程泄漏
- 推荐动作：值得精读，特别是需要跨平台适配的团队。展示了如何利用 kqueue 模拟 Linux 的 PR_SET_PDEATHSIG，以及事件驱动 vs 轮询的设计权衡。模块封装实践也值得参考。

功能与动机

修复 macOS 上因缺少 PR_SET_PDEATHSIG 导致的孤儿进程问题和资源泄漏。PR body 描述了在 macOS 运行 MLX 后端时，sglang::scheduler 等子进程在父进程退出后成为孤儿，累积 27 GB 内存并导致端口冲突和 OOM。关联 Issue #23627 专门跟踪此非 Linux 路径问题。

实现拆解

1. 新增 kqueue 监视器模块：在 python/sglang/srt/hardware_backend/mlx/parent_watchdog.py 中实现 start_parent_death_watcher()，启动一个 daemon 线程，通过 macOS 原生 select.kqueue 注册 EVFILT_PROC / NOTE_EXIT 事件监视父进程退出。
2. 竞态条件处理：在注册事件前后检测父进程是否已退出，若已退出则立即自杀，避免死等。
3. 平台分支扩展：在 python/sglang/srt/utils/common.py 的 kill_itself_when_parent_died() 中添加 elif sys.platform == "darwin": 分支，懒导入并调用监视器。
4. 其他平台保持警告：非 Linux 非 macOS 平台仍打印警告但无操作。
5. 测试配套：CI 不支持 macOS，但作者手动进行了 A/B 验证，确认修复有效。

关键文件：

- python/sglang/srt/hardware_backend/mlx/parent_watchdog.py（模块 后端模块；类别 source；类型 dependency-wiring；符号 start_parent_death_watcher, _watch_parent）：新增的核心模块，实现 kqueue 事件驱动的父进程死亡监视器，是修复的关键所在。
- python/sglang/srt/utils/common.py（模块 工具层；类别 source；类型 dependency-wiring）：修改 kill_itself_when_parent_died 函数，增加 darwin 分支，调用 MLX 后端监视器，其他平台警告更新。

关键符号：start_parent_death_watcher, _watch_parent, kill_itself_when_parent_died

关键源码片段

python/sglang/srt/hardware_backend/mlx/parent_watchdog.py

新增的核心模块，实现 kqueue 事件驱动的父进程死亡监视器，是修复的关键所在。

```
"""Parent-death watchdog for MLX workers on Apple Silicon.

macOS has no ``PR_SET_PDEATHSIG`` equivalent, so the kernel will not signal a
worker process when its parent dies; the worker would be reparented to PID 1
and leak (holding GPU/host memory and ports). This module emulates PDEATHSIG
with a daemon thread that watches the parent PID via kqueue and SIGKILLs the
current process once it gets orphaned.
"""

import os
import select
import signal
import threading


def start_parent_death_watcher() -> None:
    """SIGKILL this process once its current parent exits (macOS only).

    kqueue with an ``EVFILT_PROC`` / ``NOTE_EXIT`` filter is the native,
    event-driven mechanism on macOS (exposed via ``select.kqueue`` /
    ``select.kevent``), so the watcher thread blocks until the parent actually
    exits instead of waking up to poll.

    ``SIGKILL`` is sent from this watcher thread and is uncatchable /
    unblockable, so it works even when the main thread is stuck inside a
    blocking native call (e.g. an MLX/Metal ``mx.eval`` / ``.tolist()``).
    """
    original_ppid = os.getppid()

    def _watch_parent():
        kq = select.kqueue()
        # 注册一个 kqueue 事件，监视父进程退出 (NOTE_EXIT)
        kev = select.kevent(
            original_ppid,
            filter=select.KQ_FILTER_PROC,
            flags=select.KQ_EV_ADD,
            fflags=select.KQ_NOTE_EXIT,
        )
        try:
            # 注册监视器；若父进程已退出会抛出 ProcessLookupError (ESRCH)
            kq.control([kev], 0, None)
        except (ProcessLookupError, OSError):
            # 父进程在注册之前已退出，立即自杀
            os.kill(os.getpid(), signal.SIGKILL)
            return
        # 防止获取 original_ppid 和注册监视器之间的竞态条件
```

```
        if os.getppid() != original_ppid:
            os.kill(os.getpid(), signal.SIGKILL)
            return
        # 阻塞直到父进程退出（事件到达），然后自杀
        kq.control(None, 1, None)
        os.kill(os.getpid(), signal.SIGKILL)

    watcher = threading.Thread(
        target=_watch_parent,
        name="parent-death-watcher",
        daemon=True,
    )
    watcher.start()
```

评论区精华

- reviewer yeahdongcn 建议使用 kqueue 替代轮询：避免 CPU 空转，改为事件驱动。作者采纳并重写实现。
- reviewer yeahdongcn 建议将 macOS 逻辑移入 MLX 后端模块：使平台调度更清晰。作者采纳，将核心代码封装在 parent_watchdog.py，common.py 只保留导入调用。
  - 使用 kqueue 替代轮询 getppid (design): 作者采纳，提交记录中 'fix: use kqueue' 的 commit 验证了更改。
  - 将 macOS 逻辑移入 MLX 后端模块 (design): 作者采纳，创建 parent_watchdog.py，common.py 仅保留导入调用。

风险与影响

- 风险：风险较低，仅影响 macOS 平台，Linux 行为完全不变。新代码使用 SIGKILL 确保可靠终止，但需确保 select.kqueue 在目标 macOS 版本上可用（macOS 10.6+ 均支持）。未引入同步或性能开销，daemon 线程在正常情况下阻塞在 kqueue.control，几乎无 CPU 占用。缺少自动化测试覆盖，但作者已手动验证。
- 影响：对 macOS 用户：彻底解决孤儿进程导致的端口冲突和内存泄漏，提升开发和测试体验。对 Linux 用户：无影响。四种 worker 入口点（scheduler、detokenizer、DP controller、tokenizer router）均被覆盖，无功能降级。
- 风险标记：平台特定变更，缺少自动化测试

关联脉络

- 暂无明显关联 PR