PR #2545 完整报告

radixark/miles

Reuse one Daytona client per process instead of one per create attempt

合并时间：2026-08-15 05:25

原文链接：http://prhub.com.cn/radixark/miles/pull/2545

执行摘要

- 一句话：Daytona 客户端进程级复用，修复 fd 泄漏致训练停止
- 推荐动作：值得精读。这是一个教科书式的 ' 连接池 / 客户端复用 ' 资源泄漏修复：有完整的现场数据（fd 时间线、SSLEOFError 频率）、清晰的根因链（无 close + 闭包持有 + 重试放大）和有效的验证方法（16 节点复跑、fd 收敛曲线）。值得关注的设计决策包括：用 threading.Lock + 模块级单例做进程级复用并保留延迟导入、测试中用假模块替换 sys.modules 以纯单元方式验证单例语义。对团队而言，可借此建立一条 code review 检查项：凡是 SDK 持有连接池且无 close() 时，禁止在每次调用 / 重试中新建客户端。

功能与动机

PR body 描述了 16 节点 GLM52 tbench2 运行 4.5 小时后训练停止生成的现场：fd 数从 14:01 的 6044 涨到 14:06 的 10309，约 17 个 / 秒单调增长，同时出现 880 次 / 分的 ' SSLError(SSLEOFError...)' 失败，14 分钟没有 Decode batch。作者排除了 Daytona 故障：同一节点新容器对同一端点 5/5 成功、daytona.list() 正常，说明是 API 拒绝该进程而非服务不可用。根因是三点叠加：SDK 无 close()/exit、返回的闭包跨 episode 持有客户端引用、_start_sandbox 每次 create 尝试（含重试）都新建客户端，形成 ' 创建失败 → 退避重试 → 每次重试遗留连接池 → socket 累积 → 边缘拒绝更多 → 更多重试 ' 的正反馈。

实现拆解

1. 变更入口：examples/experimental/openenv/tb2_sandbox_daytona.py 的 make_daytona()。原实现每次调用都执行 from daytona import Daytona, DaytonaConfig 并返回新客户端，而该函数被 _start_sandbox 在每个创建尝试（含退避重试）中调用。
2. 新增模块级单例状态：在模块顶层增加 _client_lock = threading.Lock() 与 _client = None，make_daytona() 改为在锁内检查 _client is None，首次构建后缓存并返回同一实例。锁保证多线程并发首次构建安全，覆盖了 128 并发创建沙箱的场景。
3. 保留延迟导入与配置假设：daytona 包仍在首次构建时才导入，避免模块导入期依赖；api_key（来自 resolve_api_key()）与 api_url（DAYTONA_API_URL 或默认端点）进程内恒定，因此共享单实例不会产生配置冲突，这正是连接池的设计用途。
4. 测试配套：examples/experimental/openenv/tests/test_tb2_sandbox_daytona.py 新增 test_make_daytona_reuses_one_client：用 monkeypatch 重置模块级 _client，将 sys.modules['daytona'] 替换为记录构造次数的假模块，连续调用 5 次 make_daytona()，

断言 len(built) == 1 且所有返回值是同一对象；该文件共 9 个测试全部通过。

5. 现场验证：16 节点同 128 并发负载复跑，fd 每 10 分钟采样：317 → 420 → 430 → 443 → 435，随后在 435-469 区间波动，不再单调增长，全程无 SSLEOFError。
6. 配套语义变化：对 OPENENV_DAYTONA_CREATE_MAX_RETRIES 的调优含义改变——修复前提高重试上限会加速泄漏，修复后重试才真正用于扛过配额竞争。

关键文件：

- *examples/experimental/openenv/tb2_sandbox_daytona.py*（模块 沙箱客户端；类别 source；类型 core-logic；符号 make_daytona, _client, _client_lock）：核心修复文件：make_daytona() 从每次调用新建 Daytona 客户端改为进程级单例（threading.Lock + 模块级 _client），消除连接池泄漏，是 fd 无界增长问题的直接解药。
- *examples/experimental/openenv/tests/test_tb2_sandbox_daytona.py*（模块 沙箱测试；类别 test；类型 test-coverage；符号 test_make_daytona_reuses_one_client, _Daytona）：新增 test_make_daytona_reuses_one_client，用假 daytona 模块验证连续 5 次调用只构造 1 个客户端且返回同一实例，防止单例语义回归。

关键符号：make_daytona, test_make_daytona_reuses_one_client

关键源码片段

examples/experimental/openenv/tb2_sandbox_daytona.py

核心修复文件：make_daytona() 从每次调用新建 Daytona 客户端改为进程级单例（threading.Lock + 模块级 _client），消除连接池泄漏，是 fd 无界增长问题的直接解药。

```
# 模块级单例状态：_client_lock 保护并发下的首次构建
_client_lock = threading.Lock()
_client = None


def make_daytona():
    """返回进程内共享的 Daytona 客户端。

    旧实现每次调用都新建客户端，而 SDK 自身持有连接池且不提供
    close()/__exit__；每次 create 尝试（含重试）都会遗留一个连接池，
    由调用方闭包长期引用，GC 无法回收。API 一旦抖动就形成
    失败 → 重试 → 更多孤儿连接池 → 句柄耗尽 → 更多失败 的正反馈。
    """
    global _client
    with _client_lock:
        if _client is None:
            # 延迟导入让模块导入本身无副作用；api_key/api_url 在进程内恒定，
            # 因此共享单个客户端正是连接池语义的合理用法。
            from daytona import Daytona, DaytonaConfig

            _client = Daytona(
                DaytonaConfig(
                    api_key=resolve_api_key(),
```

```
                api_url=os.getenv("DAYTONA_API_URL", "https://app.daytona.io/api"),
            )
        )
    return _client
```

评论区精华

审核人 nblintao 仅有的 3 条评论均为非阻塞意见，最终给出 LGTM 并批准：

1. docstring 数字细节：nblintao 在 make_daytona() 的 docstring 上提问 'numbers may be too detailed for docstring?'，针对 diff 中 '9144 file descriptors' 等具体观测数据。作者随即将具体数字从 docstring 移除，head 版本只保留机制描述（'Built once and shared: the SDK owns a connection pool and exposes no close()...'），评论状态可视为已解决。
2. 单例实现的 TODO：nblintao 在 _client 赋值处（line 175）留言 'My TODO: https://github.com/radixark/miles/issues/2547'，将关于客户端复用后续改进（可能是连接池清理 / 关闭能力）记录到独立 issue，未阻塞本 PR。
3. 测试的 TODO：nblintao 在 test_make_daytona_reuses_one_client 处留言 'My TODO: https://github.com/radixark/miles/issues/2546'，同样外链 issue 跟踪测试层面的后续工作。

- docstring 中数字细节是否过多 (style): 作者随后精简了 docstring，head 版本只保留机制描述（无具体数字），reviewer 最终 APPROVED/LGTM，视为已解决。
- 客户端复用后续工作 TODO（issue 2547） (other): PR 内未解决，由 reviewer 外链 issue 跟踪，未阻塞合入。
- 测试后续工作 TODO（issue 2546） (other): PR 内未解决，由 reviewer 外链 issue 跟踪，未阻塞合入。

风险与影响

- 风险：
  1. 全局单例的线程安全依赖 SDK：锁只保护首次构建，后续所有线程共享同一客户端，正确性依赖 daytona Python SDK 连接池的线程安全性；本 PR 在 128 并发下复跑验证通过，但 SDK 升级后存在回归风险，建议在 CI 或例行运行中关注。
  2. 单例不可重建：进程内 _client 一旦构建，若 api key 轮换或客户端内部状态损坏（如连接池混入坏连接），无法在不重启 worker 的情况下恢复。对长跑实验，这意味着一旦出现此类问题需要整体重启。
  3. 测试使用假模块：测试通过替换 sys.modules['daytona'] 验证单例语义，未覆盖真实 SDK 的 under-the-hood 行为（如连接池是否真的被复用）；若 SDK 内部存在模块级全局状态，测试可能失真，但目前语义简单，风险可控。
  4. 影响范围有限：变更仅作用于 examples/experimental/openenv 示例代码，不触及核心训练 /rollout 路径，因此对主库稳定性风险低。 - 影响：对用户：运行 openenv / GLM5.2 TB2 等长时间实验的用户不再遭遇 ' 跑几小时后 Daytona 拒绝进程、训练停产 ' 的问题，fd 占用从无界增长变为稳定收敛，实验可连续运行数小时以上。对系统：每进

程仅保留一个 Daytona 连接池，减少了大量重复 TLS 连接与 socket 占用，降低了对云端 API 的拒绝触发概率。对团队：为大规模 rollout 实验提供了更稳的资源底座，同时改变了 OPENENV_DAYTONA_CREATE_MAX_RETRIES 的调优语义——修复前提高重试上限会加速泄漏。影响程度中等偏小：改动仅 2 个文件、影响模块局限于 openenv 实验示例。 - 风险标记：进程级全局单例，依赖 SDK 连接池线程安全，示例实验路径变更

关联脉络

- PR #2280 [example] GLM-5.2 744B-A40B LoRA agentic launcher (TB2 on Daytona): 同属 Daytona/TB2 实验线：2280 新增基于 Daytona 的 GLM5.2 TB2 启动器，本 PR 修复该实验线长跑时的 Daytona 客户端资源泄漏。
- PR #2544 Do not kill the run when one sample's collect_samples loses its connection: 同属长时间 rollout 运行的连接故障韧性主题：2544 处理采集（collect）阶段的断连不拖垮训练，本 PR 处理创建（create）阶段连接池泄漏导致的运行停产。