PR #1913 完整报告

radixark/miles

E2B sandbox backend (E2B Cloud / self-hosted AgentENV) + dedicated AgentENV recipe

合并时间：2026-08-06 04:43

原文链接：http://prhub.com.cn/radixark/miles/pull/1913

执行摘要

- 一句话：新增 E2B/AgentENV 沙箱后端，重构沙箱编排共享层
- 推荐动作：值得精读。重点看三个设计：一是 template_alias 的 recipe-digest 别名机制，把“何时重新构建镜像”从人工约定变成确定性哈希；二是 openenv_sandbox_common.create_once 对 asyncio.to_thread 不可取消性的处理，用守护线程 reaper 回收孤儿沙箱，避免资源泄漏与计费；三是启动器层“只转发密钥文件路径、不转发密钥值”的安全契约，对 Ray runtime_env 明文日志问题给出低成本解法。若要扩展新的沙箱 provider，本 PR 的注册表与共享层是很好的模板。

功能与动机

PR body 指出 Daytona 路径存在两堵实测到的扩容墙：org 快照配额与 org 内存配额（约 256 并发）。AgentENV 是 Kimi K3 团队自托管的 Firecracker 微 VM 平台，原生 API 就是 E2B API，快照支撑的沙箱启动可达亚秒级，且模板数量不受配额限制（只受快照存储容量限制）。因此需要新增一个 E2B 后端，让训练可以在 E2B Cloud（零基础设施默认）或自托管 AgentENV 上以 per-episode 沙箱形式运行，同时保持 recipe、agent 循环和评分契约不变。

实现拆解

实现分四步：

1. 物化 E2B 模板与沙箱生命周期：新增 examples/experimental/openenv/tb2_sandbox_e2b.py，把 provider 无关的 tb2_sandbox_recipe 变成 E2B 可执行形态。核心是 template_alias()：基于 base 镜像、server_layer_commands() 全部构建命令和 task_build_resources() 计算 SHA-256 digest，生成 tb2-<task>-<digest[:10]> 确定性别名，recipe 编辑或 task.toml 资源配置变更会自动重新构建模板；ensure_task_template() 用 per-alias 进程内锁去重并发构建，并把 Template.build 放到带超时的线程池里，避免一个任务多个 episode 同时触发多分钟级构建；create_task_sandbox() 从模板 warm-start，后台启动 env server，健康检查后启动 TTL keepalive 线程（dead-man's switch，进程死亡后 provider 自动回收孤儿沙箱）。
2. E2B agent 函数与共享编排层：新增 openenv_e2b_agent_function.py（复用 openenv_agent_function._multi_turn 循环，只提供 run_episode 和 run 入口）和 openenv_sandbox_common.py。共享层把 Daytona leg 原有的 cancel-safe 创建、进程级并发信号量、抖动指数退避、episode 生命周期上下文管理器提炼为 provider 无关骨架：create_once() 处理 asyncio.to_thread 不可取消导致的孤儿泄漏，取消时用守护线程

reaper 等待创建完成后关闭沙箱；lazy_semaphore() 延迟到首次 episode 时绑定事件循环；start_task_sandbox() 只在创建尝试期间持有信号量，退避期间释放以保持流水线满载。

3. 启动器参数化与显式后端解析：修改 openenv_launch_common.py，用 resolve_sandbox_backend() 取代"设置 tasks_dir 就隐式选择 Daytona"的隐式行为，要求 tasks_dir 与后端名成对出现；新增 _sandbox_key_supply() 和 _preflight_sdk()，把 Daytona 的密钥文件契约（只转发路径、不转发值，避免 ray runtime_env 明文泄漏）泛化为 E2B_API_KEY/_FILE，并在启动时预检 e2b SDK 与 tbench2_env 版本。agent_args() 改为从 sandbox_common.AGENT_FUNCTIONS 注册表按后端名解析 agent function 路径。
4. Daytona leg 重构与配套测试 / 文档：openenv_daytona_agent_function.py 删除约 84 行重复编排代码，改为调用共享层；tb2_sandbox_recipe.py 上移 sandbox_labels、resolve_api_key、start_keepalive 等通用函数；tb2_sandbox_daytona.py 相应复用。新增三个测试文件覆盖模板别名确定性 / 内容跟踪 / 资源跟踪、密钥供给、keepalive 生命周期、episode 分发、节流重试 / 放弃 / 取消回收；新增 examples/experimental/agentenv/README.md 自托管配方，更新 openenv/README.md 与 docs/user-guide/environments.md、openenv.md。

关键文件：

- examples/experimental/openenv/tb2_sandbox_e2b.py（模块 沙箱物化；类别 source；类型 core-logic；符号 template_alias, task_build_resources, _connection_opts, _build_lock）：E2B 后端的核心物化模块：模板别名、构建锁、TTL keepalive、沙箱创建与 URL 解析，是本 PR 的主体新增代码。
- examples/experimental/openenv/openenv_sandbox_common.py（模块 共享编排；类别 source；类型 core-logic；符号 resolve_backend, lazy_semaphore, get, create_once）：provider 无关的共享编排层：取消安全创建、懒信号量、节流退避、episode 上下文管理器，Daytona 与 E2B 两条腿共同依赖，是本 PR 的架构核心。
- examples/experimental/openenv/openenv_e2b_agent_function.py（模块 Agent 函数；类别 source；类型 core-logic；符号 _extra_throttle_patterns, _is_throttle_error, _start_sandbox, _start_task_sandbox）：E2B 的 agent 函数入口，把共享编排层与 tbench2 agent 循环接起来，是后端对 rollout 的适配面。
- examples/experimental/openenv/openenv_launch_common.py（模块 启动参数；类别 source；类型 configuration；符号 resolve_sandbox_backend, agent_args, _sandbox_key_supply, _preflight_sdk）：启动器参数化改造：显式后端解析、密钥文件契约泛化、SDK 预检，直接影响所有 openenv 启动器。
- examples/experimental/openenv/openenv_daytona_agent_function.py（模块 Agent 函数；类别 source；类型 refactor；符号 _get_create_sem, _start_task_sandbox, _episode_env）：Daytona leg 重构为复用共享编排层，删除约 84 行重复代码，是共享层落地的验证样例。
- examples/experimental/openenv/tb2_sandbox_recipe.py（模块 沙箱配方；类别 source；类型 refactor；符号 sandbox_labels, resolve_api_key, start_keepalive, add_task_selection_args）：把 sandbox_labels、resolve_api_key、start_keepalive 等通用能力上移到 provider 无关层，供 Daytona 与 E2B 共用。

- examples/experimental/openenv/tests/test_tb2_sandbox_e2b.py（模块 沙箱测试；类别 test；类型 test-coverage；符号 _patch_recipe, test_template_alias_is_deterministic_and_sanitized, test_template_alias_tracks_recipe_content, test_template_alias_tracks_build_resources）：E2B 模板别名、构建锁、密钥供给与 keepalive 生命周期的离线测试，是保证 digest 机制正确性的关键配套。
- examples/experimental/openenv/tests/test_openenv_e2b_agent_function.py（模块 Agent 测试；类别 test；类型 test-coverage；符号 run_async, test_e2b_leg_dispatch, fake_episode_env, test_e2b_leg_eval_error_yields_no_verdict）：E2B episode 分发、节流重试 / 放弃 / 取消回收的离线测试，验证共享编排层行为。

关键符号：template_alias, task_build_resources, ensure_task_template, create_task_sandbox, base_url, _start_keepalive, resolve_backend, lazy_semaphore, create_once, start_task_sandbox, episode_env, resolve_sandbox_backend, _sandbox_key_supply, _preflight_sdk, run_episode, run

关键源码片段

examples/experimental/openenv/openenv_sandbox_common.py

provider 无关的共享编排层：取消安全创建、懒信号量、节流退避、episode 上下文管理器，Daytona 与 E2B 两条腿共同依赖，是本 PR 的架构核心。

```
# examples/experimental/openenv/openenv_sandbox_common.py（核心片段）
# 后端注册表：launcher 与兄弟工具都从这里解析后端名，
# 保证"agentenv 是 e2b 的别名"这一映射不会在各处漂移。
AGENT_FUNCTIONS = {
    "daytona": "openenv_daytona_agent_function.run",
    "e2b": "openenv_e2b_agent_function.run",
}
_ALIASES = {"agentenv": "e2b"}


def resolve_backend(name: str | None) -> str:
    # 故意不提供默认值：选哪个 provider 决定整个 rollout 消耗谁的配额、
    # 需要哪些凭证，留空就是一种需要回答的问题，而不是靠猜。
    backend = (name or "").strip().lower()
    allowed = ", ".join(sorted([*AGENT_FUNCTIONS, *_ALIASES]))
    if not backend:
        raise ValueError(f"no sandbox backend named; choose one of: {allowed}")
    backend = _ALIASES.get(backend, backend)
    if backend not in AGENT_FUNCTIONS:
        raise ValueError(f"unknown sandbox backend {name!r}; choose one of: {allowed}")
    return backend


async def create_once(start_fn: StartFn, task_id: str, tasks_dir: str, *, logger):
    """单次创建，对取消安全。

    asyncio.to_thread 不可取消：episode 墙钟到点取消协程时，
```

```
    工作线程仍会继续创建。若结果被丢弃就会泄漏沙箱，直到 provider 侧
    TTL 兜底回收。因此把结果记录在线程侧，取消后交给 reaper 线程，
    等创建完成立即关闭孤儿沙箱。
    """
    result: list[tuple[Any, str]] = []
    done = threading.Event()

    def _start() -> tuple[Any, str]:
        try:
            result.append(start_fn(task_id, tasks_dir))
        finally:
            done.set()
        return result[0]

    try:
        return await asyncio.to_thread(_start)
    except asyncio.CancelledError:

        def _reap() -> None:
            done.wait()
            for close_fn, _url in result:
                try:
                    close_fn()
                    logger.info(f"Closed sandbox orphaned by cancelled episode: {task_id}")
                except Exception as e:
                    logger.warning(f"Failed to close orphaned sandbox for {task_id}: {e}")

        threading.Thread(target=_reap, name=f"tb2-sandbox-reap-{task_id}", daemon=True).start()
        raise


async def start_task_sandbox(
    task_id, tasks_dir, *, start_fn, is_throttle, sem,
    max_retries, backoff_base_s, backoff_cap_s, logger, provider,
):
    """进程级节流 + 抖动指数退避。信号量只在创建尝试期间持有，
    退避期间释放，其他 episode 可以继续推进流水线。"""
    attempt = 0
    while True:
        try:
            async with sem:
                return await create_once(start_fn, task_id, tasks_dir, logger=logger)
        except Exception as e:
            if not is_throttle(e) or attempt >= max_retries:
                raise
            attempt += 1
            delay = min(backoff_cap_s, backoff_base_s * (2 ** (attempt - 1))) * (0.5 + random.
            random())
            logger.warning(
```

```
            f"{provider} create throttled for {task_id} (attempt {attempt}/{max_retries}); retrying
            in {delay:.1f}s"
        )
        await asyncio.sleep(delay)
```

examples/experimental/openenv/openenv_e2b_agent_function.py

E2B 的 agent 函数入口，把共享编排层与 tbench2 agent 循环接起来，是后端对 rollout 的适配面。

```
# examples/experimental/openenv/openenv_e2b_agent_function.py（核心片段）
# 节流分类：SDK 把 HTTP 429 类型化为 RateLimitException，
# 文本匹配兜底代理与自托管服务的纯文本限流提示，
# OPENENV_E2B_THROTTLE_PATTERNS 允许按部署扩展可重试的容量错误。
def _is_throttle_error(exc: BaseException) -> bool:
    # 只在该函数内导入：这是 RateLimitException 唯一使用点，
    # 且只在失败路径执行（此时 e2b 已因抛出 exc 而在 sys.modules 中）
    try:
        from e2b.exceptions import RateLimitException
        if isinstance(exc, RateLimitException):
            return True
    except ImportError: # pragma: no cover - 无 SDK 时
        pass
    s = str(exc).lower()
    if "too many requests" in s or "429" in s or "rate limit" in s:
        return True
    return any(p in s for p in _extra_throttle_patterns())


async def _start_task_sandbox(task_id: str) -> tuple[Any, str]:
    """创建承载 env server 的沙箱。

    返回 (close_fn, base_url)；close_fn 负责 kill 沙箱。
    取消安全创建、进程级节流、限流退避等编排都在
    openenv_sandbox_common 中，本 leg 只贡献 start hook 与节流分类。
    """
    return await common.start_task_sandbox(
        task_id,
        os.getenv("OPENENV_TB2_TASKS_DIR", "").strip(),
        start_fn=lambda tid, tdir: _start_sandbox(tid, tdir),
        is_throttle=_is_throttle_error,
        sem=_get_create_sem(),
        max_retries=_CREATE_MAX_RETRIES,
        backoff_base_s=_CREATE_BACKOFF_BASE_S,
        backoff_cap_s=_CREATE_BACKOFF_CAP_S,
        logger=logger,
        provider="E2B",
    )

async def run_episode(policy, model_name, messages, request_kwargs, metadata):
```

```
    """在自己独立的 E2B 沙箱中跑一个 episode，使用调用方自己的 policy。
    与 openenv_agent_function.run_episode 契约一致；episode 结束即 kill 沙箱，
    无额外事后清理。"""
    return await oaf._multi_turn(
        oaf._load_tbench2(),
        policy,
        model_name,
        messages,
        request_kwargs,
        metadata,
        run_body=_sandbox_run_body,
    )
```

评论区精华

Review 由 Shi-Dong 发起，均为较轻量意见，作者逐条修复：

- import 提升到脚本开头（style，4 处）：Shi-Dong 多次要求把 openenv_sandbox_common 等 import 提升到文件顶部，作者回复“Fixed.”，最终 head 版本已改为顶部导入。
- Daytona 是否不再是默认（question）：Shi-Dong 在 run-openenv-tbench2.py 的 docstring 变更处提问“I guess Daytona is not the default now?”，作者确认“Good catch. Fixed.”，修正了文档表述（tasks_dir 不再隐式选择 Daytona，需显式指定后端）。
- _connection_opts 命名（style）：Shi-Dong 建议去掉下划线前缀，认为它已不再是私有函数，作者回复“Fixed.”。

最终 Shi-Dong 给出 APPROVED 评论“LGTM!”。讨论没有触及正确性或架构层面的分歧。

- import 位置：要求提升到脚本开头 (style): 作者逐一回复“Fixed.”，head 版本已将 import 统一提升到文件顶部。
- Daytona 是否不再是默认后端 (question): 作者确认“Good catch. Fixed.”，修正文档表述：现在必须显式指定 OPENENV_SANDBOX_BACKEND。
- _connection_opts 命名是否去掉下划线 (style): 作者回复“Fixed.”，最终保留为模块内函数被 tb2_sandbox_e2b 内部使用，未再展开讨论。

风险与影响

- 风险：
  1. 行为变更风险：resolve_sandbox_backend() 把“设置 openenv_tb2_tasks_dir 即隐式选择 Daytona”改为必须显式指定 openenv_sandbox_backend，已有启动器若未同步更新会在启动时报错（这是刻意设计的 fail-fast），但需要确认所有旧启动器都已完成迁移。
  2. 模板陈旧风险：template_alias() 的 digest 覆盖 base 镜像、构建命令和构建资源，若未来新增影响产物但未纳入 digest 的输入（如环境变量），会导致静默使用陈旧模板；现有测试已覆盖别名对内容与资源的敏感性，但未覆盖环境变量维度。
  3. 构建超时后的半成品模板：ensure_task_template() 超时后 provider 侧构建可能仍在继续，调用方已释放 alias 锁；同进程后续 episode 可能通过 alias_exists 命中尚未完成的

模板。AgentENV/E2B 的模板构建原子性需要在实际部署中验证。

4. keepalive 误杀：_start_keepalive 连续 3 次 beat 失败即退出，若 provider 短暂抖动（如网关 5xx），沙箱 TTL 到期可能被 provider 杀掉，导致长 episode 中断。
5. SDK 兼容性：所有 e2b import 均懒加载，SDK 版本升级若改变 Template.build、Sandbox.create 或 set_timeout 签名，需要回归测试；PR 已用真实 SDK 验证过 RateLimitException 类型，但测试套件中该断言依赖 importorskip（无 SDK 时跳过）。 - 影响：影响范围集中在 examples/experimental/openenv 与 examples/experimental/agentenv，不触及 miles 核心训练 /rollout 代码。对使用 OpenEnv tbench2 进行 agentic RL 训练的团队：新增零基础设施的 E2B Cloud 默认路径，以及可自托管、无配额墙的 AgentENV 路径，可支撑更高并发（超过 Daytona 约 256 并发限制）；引入的 OPENENV_SANDBOX_BACKEND 使启动器参数更显式，减少隐式默认带来的误配置。对团队内部维护者：共享编排层消除了 Daytona/E2B 两条腿的重复代码，后续新增 provider 只需实现 start hook 与节流分类器。文档同步更新了用户指南，降低上手成本。 - 风险标记：行为变更（后端需显式指定），模板 digest 未覆盖的隐性输入，keepalive 失败阈值误杀长 episode，懒加载 SDK 版本兼容依赖，实验性目录未纳入仓库级 pytest

关联脉络

- PR #2136 openenv/tbench2: drop the daytona bake CLI, which nothing can consume: 同一目录 examples/experimental/openenv，且同样改动 tb2_sandbox_daytona.py；本 PR 为 openenv/tbench2 新增了 E2B 后端，2136 则清理了 Daytona 的 bake CLI，两者构成 openenv tbench2 功能线的持续演进。
- PR #1710 Daytona sandbox leg (PR body 引用): PR body 明确提到本 PR 是 Daytona leg（#1710/#1711/#1675）之后的第三次 per-task sandbox 物化，共享编排层正是从 Daytona leg 的私有实现提炼而来。
- PR #1675 Daytona sandbox leg (PR body 引用): 与 #1710/#1711 一起构成 Daytona 沙箱后端的先例，本 PR 在其基础上引入 agentenv 别名并重构共享层。