PR #49514 完整报告

vllm-project/vllm

[ROCm][CI] Use the same-build wheel in Python-only CI

合并时间：2026-08-17 11:52

原文链接：http://prhub.com.cn/vllm-project/vllm/pull/49514

执行摘要

- 一句话：ROCM Python-only CI 改用同构建 wheel 与源码 overlay
- 推荐动作：值得精读，尤其关注 run-amd-test.sh 中「git archive overlay + VLLM_VERSION_OVERRIDE + 同构建 wheel」的组合处理模式，这是 vLLM native/artifact CI 架构的关键一环，对理解后续 ROCm CI 演进很有帮助；同时它示范了「测试必须验证同一次构建产物」的 CI 最佳实践，对任何依赖预编译产物的项目都有参考价值。

功能与动机

PR body 明确要求 "Package the verified checkout into the native artifact workspace and install the wheel produced by that same build"，并 "Preserve the source tree and merge-base identity needed by the Python-only installation test"。此前 ROCm native artifact 工作区只包含 wheel 与测试文件，没有源码树和 .git，python_only_compile.sh 只能从 wheels.vllm.ai 拉取 merge-base 对应的预编译 wheel——这既依赖发布流水线进度（需要 5 分钟 × 5 次轮询等待 metadata 发布），又导致测试验证的并非当前 commit 的构建产物，可能掩盖新回归或产生假失败。

实现拆解

1. 测试脚本 wheel 来源重构（tests/standalone_tests/python_only_compile.sh）：先探测 is_rocm（VLLM_TARGET_DEVICE / ROCM_PATH / /opt/rocm / rocminfo），ROCm 环境下优先读取 VLLM_PRECOMPILED_WHEEL_LOCATION 指定的同构建 wheel（校验文件存在且 basename 匹配 vllm-*.whl 并 realpath），否则从 legacy 镜像的 /opt/vllm-wheels 取唯一 wheel；两者都不存在时才回退到旧逻辑（CI_STANDALONE_MERGE_BASE 或 git merge-base + wheels.vllm.ai metadata.json 轮询）。安装分支新增 VLLM_PRECOMPILED_WHEEL_LOCATION 环境变量路径，走 setup.py develop --no-deps，避免无编译器环境下触发源码编译。
2. native workspace 的源码树 overlay（.buildkite/scripts/hardware_ci/run-amd-test.sh）：prepare_native_workspace 新增 test_commands 参数；当命令包含 python_only_compile.sh 时，要求 BUILDKITE_BUILD_CHECKOUT_PATH 存在且为 git worktree，校验 checkout 的 HEAD 与 artifact 记录的 recorded_commit 一致，然后用 git archive 将 verified commit 打包解压到 workspace（而非直接拷贝 worktree，避免 dirty/untracked 文件污染），并检查 setup.py、pyproject.toml、vllm 目录是否齐全。

3. 版本与 wheel 环境变量注入：由于 overlay 故意不带 .git，setuptools-scm 无法从 git 推导版本，脚本用 importlib.metadata 读取已安装 vllm 的版本写入 VLLM_VERSION_OVERRIDE 并导出，同时把 artifact 中的 wheel 路径写入 VLLM_PRECOMPILED_WHEEL_LOCATION；在 native runtime 分支对 Python-only 作业强制 VLLM_TARGET_DEVICE=rocm，明确该 no-GPU 作业验证的是 ROCm 预编译 /editable 安装路径而非 CPU 平台选择。
4. 作业平台迁移与管道配置：.buildkite/test-amd.yaml 将 Python-only Installation 从 mi250_1（mirror_hardwares: amdgfx90anightly/amdmi250）迁到 mi300_1（amdgfx942nightly/amdmi300）并加 dind: false；.buildkite/test_areas/misc.yaml 的 AMD mirror 同步改为 device: mi300_1 与 dind: false。此前手动触发时出现目录创建权限问题，经 infra 排查后解决，review 阶段又按 tjtanaa 建议重新启用了该作业（不再是 input-gated，但 optional 标志保留）。
5. 测试与配置配套：没有新增单元测试，验证依赖多次 Buildkite 全量 CI 与 AMD CI 手动触发结果；最终由 tjtanaa 批准合入。

关键文件：

- tests/standalone_tests/python_only_compile.sh（模块 安装测试；类别 test；类型 test-coverage）：Python-only 安装测试的核心脚本，重构 wheel 来源选择逻辑，优先使用同构建 wheel，是本次变更的验证目标所在。
- .buildkite/scripts/hardware_ci/run-amd-test.sh（模块 作业脚本；类别 infra；类型 infrastructure）：native workspace 准备逻辑的核心改动：为 Python-only 作业 overlay 完整源码树并注入 wheel/ 版本环境变量，是本次变更的机制中枢。
- .buildkite/test-amd.yaml（模块 流水线配置；类别 config；类型 configuration）：AMD 专属 CI 流水线定义，Python-only Installation 作业从 mi250_1 迁移到 mi300_1 并启用 dind: false。
- .buildkite/test_areas/misc.yaml（模块 测试编排；类别 config；类型 configuration）：通用 CI 测试区域中的 AMD mirror 配置同步调整，跟随作业平台迁移。

关键符号：prepare_native_workspace, is_native_runtime

关键源码片段

tests/standalone_tests/python_only_compile.sh

Python-only 安装测试的核心脚本，重构 wheel 来源选择逻辑，优先使用同构建 wheel，是本次变更的验证目标所在。

```
# 优先使用同一次构建（same-build）的 ROCm wheel，而不是去 wheels.vllm.ai
# 拉取 merge-base 对应的预编译产物，保证 Python-only CI 验证的就是当前
# commit 构建出来的 wheel。
rocm_wheel=""
is_rocm=0
_vllm_target_lower="$(printf '%s' "${VLLM_TARGET_DEVICE:-}" | tr '[:upper:]' '[:lower:]')"
if [[ "${_vllm_target_lower}" == "rocm" || -n "${ROCM_PATH:-}" || -d /opt/rocm ]] \
        || command -v rocminfo >/dev/null 2>&1; then
    is_rocm=1
```

```
fi
unset -v _vllm_target_lower

if [[ "${is_rocm}" == "1" ]]; then
    # Native CI 通过环境变量显式传入已验证的 wheel 产物路径；
    # legacy ROCm 镜像则把同一个构建的 wheel 放在 /opt/vllm-wheels。
    if [[ -n "${VLLM_PRECOMPILED_WHEEL_LOCATION:-}" ]]; then
        rocm_wheel="${VLLM_PRECOMPILED_WHEEL_LOCATION}"
        # 校验文件存在且命名符合 vllm-*.whl，避免误传其他路径
        if [[ ! -f "${rocm_wheel}" || "$(basename "${rocm_wheel}")" != vllm-*.whl ]]; then
            echo "ERROR: invalid ROCm wheel location: ${rocm_wheel}" >&2
            exit 1
        fi
        rocm_wheel="$(realpath -- "${rocm_wheel}")"
    elif [[ -d /opt/vllm-wheels ]]; then
        shopt -s nullglob
        rocm_wheels=(/opt/vllm-wheels/vllm-*.whl)
        shopt -u nullglob
        if [[ "${#rocm_wheels[@]}" -ne 1 ]]; then
            echo "ERROR: expected exactly one vLLM wheel in /opt/vllm-wheels, found ${#rocm_
            wheels[@]}." >&2
            exit 1
        fi
        rocm_wheel="${rocm_wheels[0]}"
    fi
fi

# 只有拿不到本地 wheel 时才回退到 wheels.vllm.ai 的 merge-base 发布产物
if [[ -n "${rocm_wheel}" ]]; then
    echo "INFO: using same-build ROCm wheel: ${rocm_wheel}"
else
    # 旧逻辑：解析 CI_STANDALONE_MERGE_BASE 或 git merge-base，
    # 再轮询 wheels.vllm.ai 的 metadata.json（5 次 × 5 分钟）
    # ...（此处省略与 base 版本一致的 merge-base 解析与轮询代码）
fi

# ROCm 场景走 setuptools develop + 预编译 wheel；--no-deps 避免在
# 无编译器环境下触发依赖源码编译。
if [[ -n "${rocm_wheel}" ]]; then
    VLLM_PRECOMPILED_WHEEL_LOCATION="${rocm_wheel}" VLLM_USE_PRECOMPILED=1
    python3 setup.py develop --no-deps
elif [[ "${is_rocm}" == "1" ]]; then
    VLLM_PRECOMPILED_WHEEL_COMMIT=$merge_base_commit VLLM_USE_PRECOMPILED=1
    python3 setup.py develop --no-deps
else
    VLLM_PRECOMPILED_WHEEL_COMMIT=$merge_base_commit VLLM_USE_PRECOMPILED=1
    pip3 install -vvv -e .
fi
```

.buildkite/scripts/hardware_ci/run-amd-test.sh

native workspace 准备逻辑的核心改动：为 Python-only 作业 overlay 完整源码树并注入 wheel/ 版本环境变量，是本次变更的机制中枢。

```
# prepare_native_workspace 的收尾部分：对 Python-only 安装作业，把
# Buildkite checkout 中已验证 commit 的完整源码树 overlay 到 artifact
# workspace，因为 artifact 只含 wheel + 测试工作区，没有 setup.py 和
# vllm 源码树。
if [[ "${test_commands}" == *python_only_compile.sh* ]]; then
    checkout="${BUILDKITE_BUILD_CHECKOUT_PATH:-}"
    if [[ -z "${checkout}" || ! -d "${checkout}" ]]; then
        echo "Python-only native CI requires BUILDKITE_BUILD_CHECKOUT_PATH" >&2
        return 1
    fi
    # 校验 checkout 的 commit 必须与 artifact 记录的 commit 一致，
    # 防止把其他 commit 的源码混进本次构建产物
    checkout_commit=$(
        git -c "safe.directory=${checkout}" -C "${checkout}" rev-parse HEAD
    ) || return 1
    if [[ "${checkout_commit}" != "${recorded_commit}" ]]; then
        echo "Buildkite checkout ${checkout_commit} does not match ROCm artifact ${recorded_
        commit}" >&2
        return 1
    fi

    # overlay 故意不带 .git，setuptools-scm 无法推导版本号，
    # 所以从已安装的 wheel 里取版本，保证 editable 安装与产物一致
    VLLM_VERSION_OVERRIDE=$(
        python3 -c 'import importlib.metadata as m; print(m.version("vllm"))'
    ) || return 1
    export VLLM_VERSION_OVERRIDE
    VLLM_PRECOMPILED_WHEEL_LOCATION="${wheels[0]}"
    export VLLM_PRECOMPILED_WHEEL_LOCATION

    # 用 git archive 而不是直接拷贝 worktree，
    # 避免 dirty/untracked 文件污染 artifact 对应的 workspace
    git -c "safe.directory=${checkout}" -C "${checkout}" \
        archive --format=tar "${recorded_commit}" \
        | tar --no-same-owner -C "${workspace_dir}" -xf - || return 1
    for required_source in setup.py pyproject.toml vllm; do
        if [[ ! -e "${workspace_dir}/${required_source}" ]]; then
            echo "Full source checkout is missing ${required_source}" >&2
            return 1
        fi
    done
fi
```

评论区精华

- tjtanaa 在合并前提出顺序依赖："I think we should merge this after this PR #49365. I have added review to #49365."，保证两个 native workspace 相关改动按序合入。
- tjtanaa 手动触发 AMD CI 后发现 Python-only 测试组不通过："This test group is not triggered on AMD CI... result not passing"；AndreasKaratzas 定位为 "permission issue regarding creating a dir. Will check with infra"，与基础设施团队确认后修复。
- tjtanaa 在批准时再次指出门控问题："LGTM, but I think that step is still gated behind input block? Should we enable it so that it always run now that we fixed the test?"，AndreasKaratzas 回复 "I re-enabled it"，随后作业恢复常态化触发。
- 合并顺序：先合 #49365 (other): 按协作顺序等待 #49365 先合，本 PR 随后合入。
- AMD CI 上 Python-only 测试组未自动触发且手动运行失败 (other): 与基础设施团队协作解决权限问题后测试通过。
- 作业是否仍被 input block 门控 (other): 作业已重新启用，纳入常规触发（optional 标志保留）。

风险与影响

- 风险：
  1. run-amd-test.sh 的 commit 一致性校验非常严格：若 Buildkite checkout 与 artifact 记录的 commit 不一致，Python-only 作业会直接失败（fail fast）；但由于分支只对包含 python_only_compile.sh 的作业生效，不影响其他 native 作业。
  2. VLLM_VERSION_OVERRIDE 依赖已安装 wheel 的版本元数据（importlib.metadata），若 artifact 中 vllm 未安装或安装损坏会失败；同时 overlay 用 git archive 整体覆盖 workspace，若 artifact 中恰好有同名目录（如 vllm/）会被源码树覆盖。
  3. 作业平台从 mi250_1 迁移到 mi300_1 并启用 dind: false：若 mi300 镜像缺少运行该脚本的条件（如 /opt/vllm-wheels 布局差异），作业可能失败；目前有 optional: true（test-amd.yaml）与 soft_fail: true（misc.yaml mirror）缓冲。
  4. 对 wheels.vllm.ai 外部依赖的解除是正向变化，但保留了完整回退路径，双路径维护略微增加脚本复杂度。 - 影响：影响范围为 AMD/ROCm CI 的 Python-only Installation 测试组及其运行平台：测试保真度提升（验证同构建产物）、稳定性提升（不再受 wheels.vllm.ai 发布时序和 5 分钟轮询影响）、并解除了对 merge-base 旧产物的隐式依赖。对产品代码和最终用户无感知；对 CI 维护团队而言，作业平台迁移（mi250→mi300）与 input gate 重新启用会带来一定的运维关注成本，但整体收益大于成本。 - 风险标记：平台迁移（mi250→mi300），VLLM_VERSION_OVERRIDE 依赖已安装 wheel，git archive overlay 覆盖工作区文件

关联脉络

- PR #49365 （标题不在输入上下文中）：tjtanaa 在评论中明确要求先合并 #49365 再合本 PR，二者同属 ROCm native/artifact workspace CI 演进；该 PR 不在输入的历史列表中，标题无法确认。