

PR #55163 完整报告

vllm-project/vllm

[CI/Build] Upload CPU nightly image to Docker Hub

合并时间: 2026-09-09 17:13

原文链接: <http://prhub.com.cn/vllm-project/vllm/pull/55163>

执行摘要

- 一句话: 为 CPU 镜像添加 nightly 发布流程, 自动推送多架构镜像到 Docker Hub。
- 推荐动作: 这是一个常规的 CI/infra 变更, 实现了有价值的功能扩展 (CPU nightly 镜像)。虽然不涉及核心计算逻辑, 但其配置正确性对发布流程至关重要。建议关注 `buildkite/release-pipeline.yaml` 中条件逻辑的完整性和 `push-nightly-builds-cpu.sh` 脚本的健壮性。对于负责 CI/CD 或需要 CPU nightly 镜像的团队成员, 此 PR 值得精读以理解其发布流程设计。

功能与动机

为了给 CPU 镜像提供与 CUDA 镜像同等的 nightly 发布能力, 确保 CPU 用户能够及时获取最新的 nightly 构建版本。PR body 中明确指出目标是“Add nightly Buildkite step to upload and published CPU nightly images”。此前, CPU 镜像仅在发布流程中手动构建和推送, 缺少自动化的 nightly 更新机制。

实现拆解

1. 调整 CPU 构建流水线配置: 修改 `.buildkite/release-pipeline.yaml` 中的 CPU 镜像构建步骤。为 `block-cpu-release-image-build` 和 `block-arm64-cpu-release-image-build` 添加了 `if: build.env("NIGHTLY") != "1"` 条件, 使得在 nightly 模式下这些阻塞步骤被跳过, 构建自动运行。同时, 修改了构建命令中的镜像标签, 从基于版本号 `$(buildkite-agent meta-data get release-version)` 改为基于提交 SHA `$BUILDKITE_COMMIT-<arch>`, 以支持 nightly 构建的唯一标识。构建步骤被设置为 `allow_dependency_failure: true`, 以增强流水线的容错性。
2. 新增 nightly 发布脚本: 创建 `.buildkite/scripts/push-nightly-builds-cpu.sh`。该脚本负责从 ECR 拉取由上一步构建的 `x86_64` 和 `arm64` 镜像, 将它们重新标记为 `vllm/vllm-openai-cpu:nightly-<arch>` 并推送到 Docker Hub。最后, 它使用 `docker manifest create` 创建并推送两个多架构清单: `vllm/vllm-openai-cpu:nightly` 和 `vllm/vllm-openai-cpu:nightly-<commit>`, 为用户提供统一入口和可追溯的版本。
3. 集成 nightly 发布步骤: 在 `release-pipeline.yaml` 的“Publish release images”组中新增一个名为“Publish nightly CPU multi-arch image to DockerHub”的步骤。该步骤依赖于两个 CPU 构建步骤, 并在 `NIGHTLY` 环境变量为 1 时运行。它调用新脚本进行发布, 并复用 `cleanup-nightly-builds.sh` 来清理旧镜像 (指定了镜像仓库为 `vllm/vllm-openai-cpu`)。

4. 同步更新发布脚本：修改 `.buildkite/scripts/publish-release-images.sh`，将 CPU 镜像的源标签从版本号格式 `v${RELEASE_VERSION}` 调整为提交 SHA 格式 `${COMMIT}-<arch>`，以与流水线中构建步骤产生的新标签保持一致，确保发布流程和 nightly 流程使用相同的镜像源。

关键文件：

- `.buildkite/release-pipeline.yaml`（模块 CI 流水线；类别 `config`；类型 `configuration`）：核心流水线配置，定义了 CPU 镜像的构建条件、依赖和新的 nightly 发布步骤。
- `.buildkite/scripts/push-nightly-builds-cpu.sh`（模块 CI 脚本；类别 `other`；类型 `core-logic`）：新增的 nightly 镜像发布脚本，实现了从 ECR 拉取、重标记和推送多架构 CPU 镜像到 Docker Hub 的完整逻辑。
- `.buildkite/scripts/publish-release-images.sh`（模块 CI 脚本；类别 `other`；类型 `configuration`）：更新发布脚本以匹配新的 CPU 镜像标签约定（从版本号改为提交 SHA），确保发布流程兼容。

关键符号：未识别

关键源码片段

`.buildkite/scripts/push-nightly-builds-cpu.sh`

新增的 nightly 镜像发布脚本，实现了从 ECR 拉取、重标记和推送多架构 CPU 镜像到 Docker Hub 的完整逻辑。

```
#!/bin/bash
# SPDX-License-Identifier: Apache-2.0
# SPDX-FileCopyrightText: Copyright contributors to the vLLM project
#
# Push CPU nightly images from ECR to Docker Hub as
# vllm/vllm-openai-cpu:nightly and vllm/vllm-openai-cpu:nightly-<commit>,
# combining the x86_64 and aarch64 architectures into a multi-arch manifest.
# Run when NIGHTLY=1 after the CPU release images have been pushed to ECR.
#
# Local testing (no push to Docker Hub):
# BUILDKITE_COMMIT=<commit-with-cpu-images-in-ecr> DRY_RUN=1 bash .buildkite/scripts/
push-nightly-builds-cpu.sh
# Requires: AWS CLI configured (for ECR public login), Docker. For full run: Docker Hub login.

set -ex

# Use BUILDKITE_COMMIT from env (required; set to a commit that has CPU images in ECR for
local test)
BUILDKITE_COMMIT="${BUILDKITE_COMMIT:?Set BUILDKITE_COMMIT to the commit SHA
that has the CPU images in ECR (e.g. from a previous release pipeline run)}"
DRY_RUN="${DRY_RUN:-0}"

TAG_NAME="nightly"
TAG_NAME_COMMIT="nightly-${BUILDKITE_COMMIT}"
```

```

# arch-dependent source images in ECR (pushed by the CPU release image build steps)
X86_ORIG_TAG="public.ecr.aws/q9t5s3a7/vllm-cpu-release-repo:${BUILDKITE_COMMIT}-x86_64"
ARM64_ORIG_TAG="public.ecr.aws/q9t5s3a7/vllm-arm64-cpu-release-repo:${BUILDKITE_COMMIT}-arm64"

echo "Pushing CPU release images from ECR to Docker Hub as $TAG_NAME and $TAG_NAME_COMMIT"
echo " x86_64: $X86_ORIG_TAG"
echo " arm64: $ARM64_ORIG_TAG"
[[ "$DRY_RUN" == "1" ]] && echo "[DRY_RUN] Skipping push to Docker Hub"

# Login to ECR and pull the arch-dependent images
aws ecr-public get-login-password --region us-east-1 | docker login --username AWS --password-stdin public.ecr.aws/q9t5s3a7
docker pull "$X86_ORIG_TAG"
docker pull "$ARM64_ORIG_TAG"

# Tag arch-dependent images for Docker Hub
docker tag "$X86_ORIG_TAG" vllm/vllm-openai-cpu:"$TAG_NAME"-x86_64
docker tag "$ARM64_ORIG_TAG" vllm/vllm-openai-cpu:"$TAG_NAME"-arm64

if [[ "$DRY_RUN" == "1" ]]; then
    echo "[DRY_RUN] Would push vllm/vllm-openai-cpu:$TAG_NAME-x86_64 and vllm/vllm-openai-cpu:$TAG_NAME-arm64"
    echo "[DRY_RUN] Would create and push manifests vllm/vllm-openai-cpu:$TAG_NAME and vllm/vllm-openai-cpu:$TAG_NAME_COMMIT"
    echo "[DRY_RUN] Local tags created. Exiting without push."
    exit 0
fi

# Push arch-dependent images to Docker Hub
docker push vllm/vllm-openai-cpu:"$TAG_NAME"-x86_64
docker push vllm/vllm-openai-cpu:"$TAG_NAME"-arm64

# Create and push the arch-independent manifests
docker manifest create vllm/vllm-openai-cpu:"$TAG_NAME" vllm/vllm-openai-cpu:"$TAG_NAME"-x86_64 vllm/vllm-openai-cpu:"$TAG_NAME"-arm64 --amend
docker manifest create vllm/vllm-openai-cpu:"$TAG_NAME_COMMIT" vllm/vllm-openai-cpu:"$TAG_NAME"-x86_64 vllm/vllm-openai-cpu:"$TAG_NAME"-arm64 --amend
docker manifest push vllm/vllm-openai-cpu:"$TAG_NAME"
docker manifest push vllm/vllm-openai-cpu:"$TAG_NAME_COMMIT"

echo "Pushed vllm/vllm-openai-cpu:$TAG_NAME and vllm/vllm-openai-cpu:$TAG_NAME_COMMIT"

```

评论区精华

Review 讨论主要集中在命名一致性上。评审者 [bigPYJ1151](#) 指出新脚本中关于 arm64 镜像的命名 ([aarch64](#)) 应与常规发布版本保持一致, 使用 [arm64](#)。作者 [wincent8](#) 接受了此建议, 并在后续提交中进行了修改。此讨论确保了 nightly 镜像与正式发布镜像在架构标识上的一致性, 避免了用户混淆。讨论结论明确, 已解决。

- CPU 镜像架构命名一致性 (style): 作者 [wincent8](#) 同意并修改了相关命名, 确保 nightly 镜像与正式发布镜像使用相同的架构标识符。

风险与影响

- 风险:

1. 流水线行为变更: CPU 构建步骤从依赖 input-release-version 改为在 nightly 模式下自动运行, 并依赖于 BLOCK_* 步骤的条件跳过。如果条件逻辑或 allow_dependency_failure 设置不当, 可能导致 nightly 构建被意外阻塞或失败。
2. 镜像标签约定变更: CPU 构建产物的标签从 v\${RELEASE_VERSION} 变为 \${COMMIT}-<arch>。虽然此 PR 同步更新了 publish-release-images.sh, 但若仓库中有其他脚本或外部流程硬编码了旧标签格式, 可能导致发布失败。
3. 新脚本引入: push-nightly-builds-cpu.sh 是新增的发布脚本, 其正确性依赖于 ECR 中镜像的命名约定。脚本中的 DRY_RUN 模式提供了本地测试能力, 降低了风险。但脚本首次运行时, 依赖于先前的构建步骤成功推送了正确命名的镜像到 ECR。

- 影响:

1. 用户影响: CPU 镜像用户现在可以自动获得每日更新的 vllm/vllm-openai-cpu:nightly 镜像, 获取最新功能和修复, 与 CUDA 镜像体验对齐。这提升了 CPU 平台的易用性和功能时效性。
2. 系统影响: CI/CD 流水线增加了新的 nightly 发布路径, 完善了 vLLM 的镜像发布矩阵。流水线配置变得更灵活, 能够根据 NIGHTLY 标志切换自动构建和手动发布模式。
3. 团队影响: 为 Intel 等 CPU 后端贡献者提供了标准化的 nightly 发布通道, 降低了维护和分发成本。同时, 引入的命名规范和流程模式 (如使用提交 SHA 作为标签) 可能为其他平台镜像的 nightly 发布提供参考。 - 风险标记: CI 流水线行为变更, 镜像标签约定变更

关联脉络

- 暂无明显关联 PR