CI/CD自动化构建Docker镜像

> 一个通过CI/CD自动构建Docker镜像并用于HPC集群计算的简单示例。

项目源码：https://git.nju.edu.cn/escience/singularity-example

容器化在集群上使用时有很大的优势，主要体现在：

- 容器内模拟的root权限，无需权限即可任意安装自己想要的程序包
- 版本可控，自定义性强
- 可复用性强，一处构建，多处部署

因此，对于一些编译特别复杂但早已有Docker镜像的计算软件，或者自己写的配置环境麻烦的软件，考虑Singularity是个不错的选择。

此外，eScience[illegible]了一

本文将通过一个简单但常见的Conda虚拟环境下Python Numpy库的镜像构建来演示如何使用这一功能。

通过 Dockerfile 构建自己的镜像

通过Dockerfile，可以事先构建一个Docker镜像。

```
FROM continuumio/miniconda3:22.11.1

# 使用南大镜像站conda源
COPY .condarc /root/.condarc

# 创建环境
RUN conda create -n my-env python=3.10 numpy
```

```
# □□□
SHELL ["/bin/bash", "--login", "-c"]
RUN conda init bash
RUN echo "source activate my-env" > ~/.bashrc
ENV PATH /opt/conda/envs/my-env/bin:$PATH
```

`.condarc` `numpy` □□□□□□□□□□□□□□□Python□□□□

□□□□□Do `docker` □□□□□□□

```
docker build -t escience/conda-numpy .
```

> Docker□□□□□Do `docker.nju.edu.cn` □

□□□□□□□□□□□□□□□□□□□□

```
docker run -i escience/conda-numpy python < test.py
```

□□CI/CD

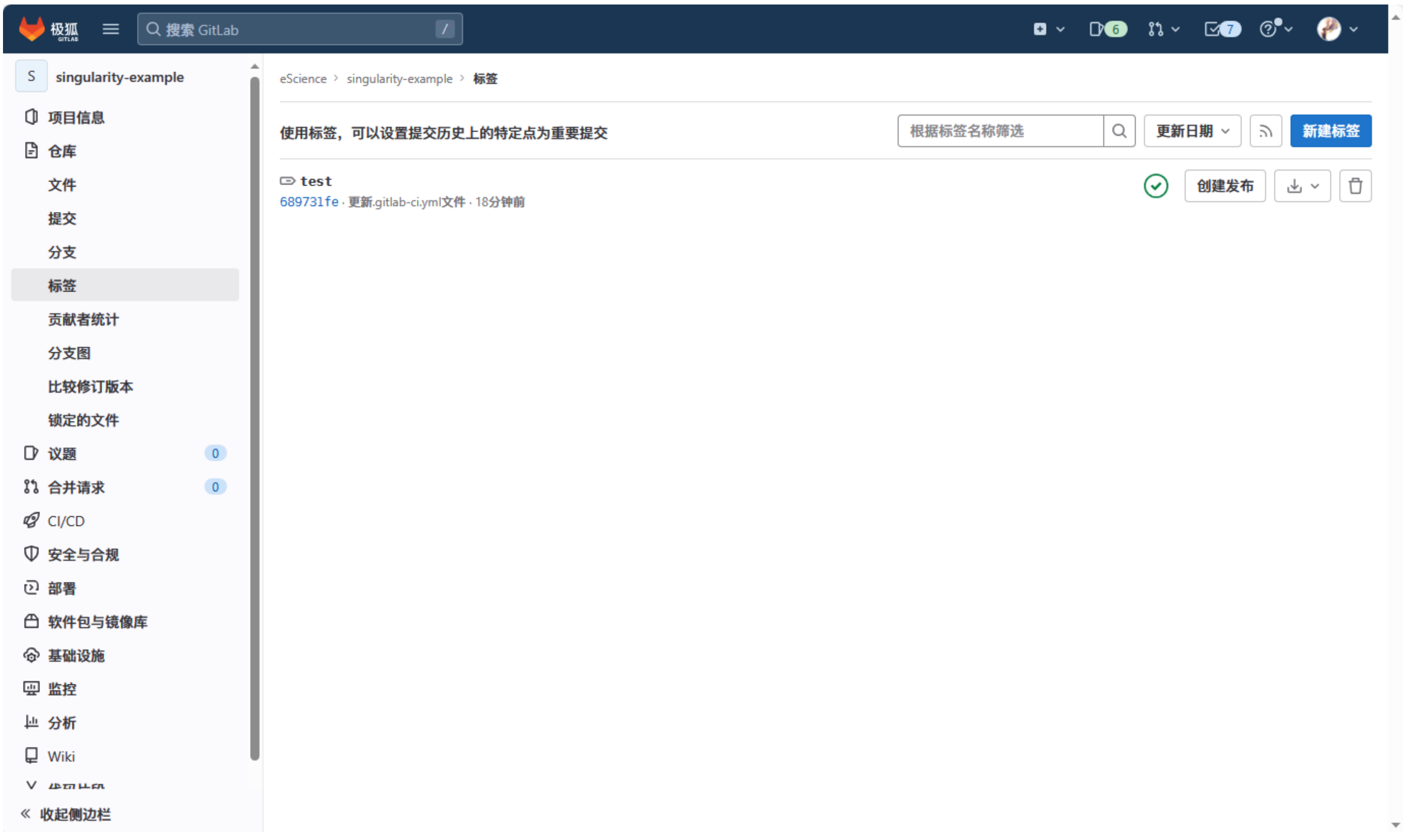
CI/CD `git.nju.edu.cn` `.gitlab-ci.yml` □□□□□CI/CD□□□□□□□□□□□□□□□□□□□□□□□□□□□□□

CI/CD `gcr.nju.edu.cn` □□□□□□□□□□□□□□□

```
# □□□
stages:
  - build
  - test
# □□□□□
build:
  stage: build
  image:
    name: gcr.nju.edu.cn/kaniko-project/executor:debug # □□gcr□□□□
    entrypoint: [""]
```

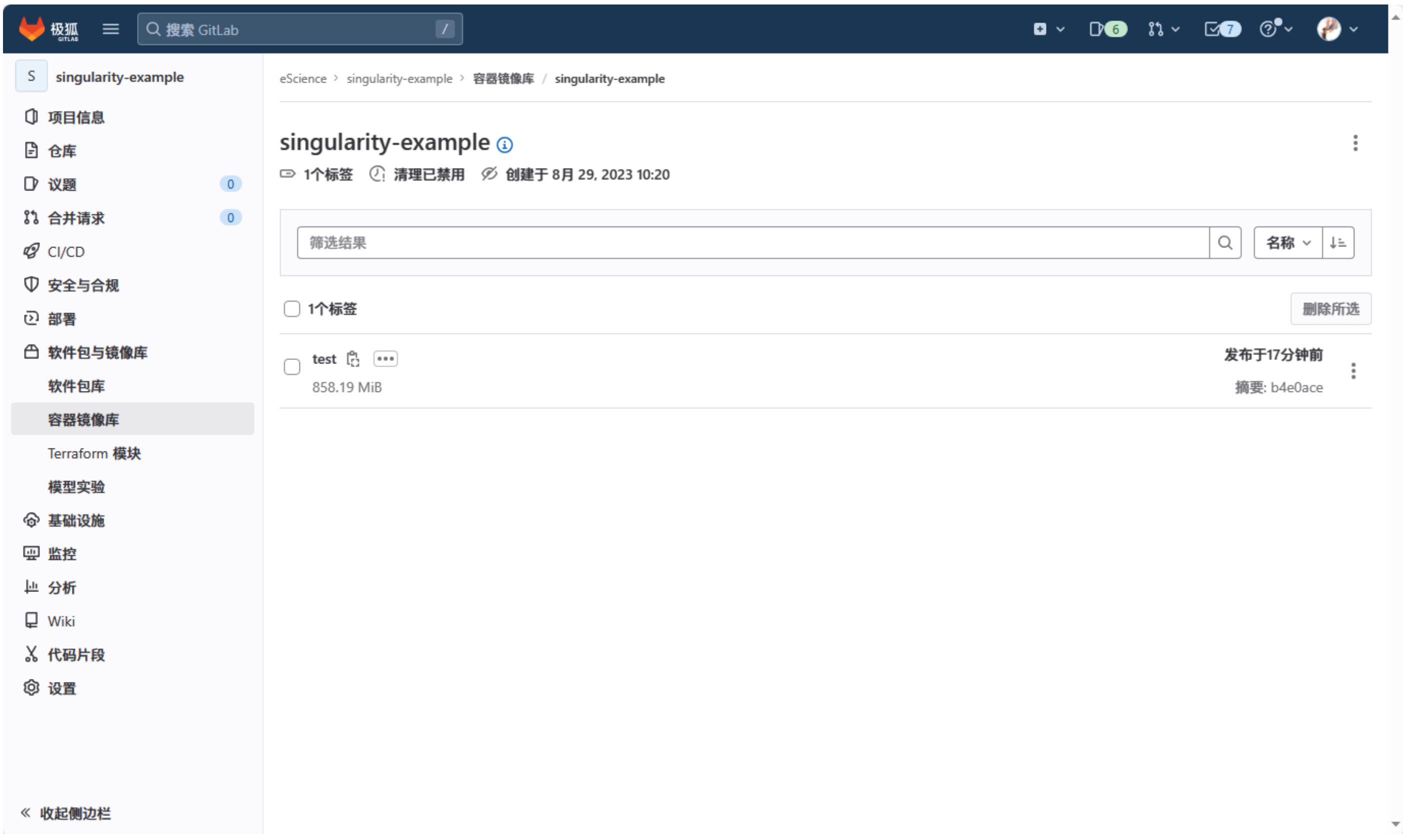
```
  script:
    - /kaniko/executor
      --context "${CI_PROJECT_DIR}"
      --dockerfile "${CI_PROJECT_DIR}/Dockerfile"
      --destination "${CI_REGISTRY_IMAGE}:${CI_COMMIT_TAG}"
  rules: # [illegible]
    - if: $CI_COMMIT_TAG
# [illegible]
test:
  stage: test
  image:
    name: ${CI_REGISTRY_IMAGE}:${CI_COMMIT_TAG}
  script:
    - python "${CI_PROJECT_DIR}/test.py"
    rules:
    - if: $CI_COMMIT_TAG
```

`test` [illegible]



[illegible]“CI/CD”-“[illegible]”[illegible]

□“□□□□□□□□”□“□□□□□”



> □□□□□□ `docker push` `reg.nju.edu.cn` □□□□□□Singularity□Dockerfile□□□□□□□□□□□□□□□□□□□□□□□□□Docker□□□

□□□□□□

`hpc.nju.edu.cn` □□□[□□□□□□□□□□□□□□□□□□□□□]

□□□□□□□□□Singularity□□□

```
singularity build conda-numpy.sif docker://reg.nju.edu.cn/escience/singularity-example:test
```

```
INFO:    Starting build...
Getting image source signatures
Copying blob 3389e9eb8624 done
Copying blob 3f4ca61aafcd done
Copying blob baee49be4542 done
Copying blob 7b4354700ca4 done
Copying blob 69a5d9e1ecd6 done
Copying blob 8a7a15cee421 done
Copying blob 5f4e24b7e321 done
Copying config 51871566f8 done
Writing manifest to image destination
Storing signatures
2023/08/29 21:46:55  info unpack layer: sha256:3f4ca61aafcd4fc07267a105067db35c0f0ac630e1970f3cd0c7bf552780e985
2023/08/29 21:46:57  info unpack layer: sha256:69a5d9e1ecd6566da53d0978004bdf37dddfaba1d8a6117966f397b41cbbc529
2023/08/29 21:46:59  info unpack layer: sha256:7b4354700ca480732ead22a553cc45916dc5466709ca64d964c4647b5b9343e9
2023/08/29 21:47:02  info unpack layer: sha256:baee49be454261f20f9770566da694b7e7845cf7d279cc2421c6b3eed68c012c
2023/08/29 21:47:02  info unpack layer: sha256:8a7a15cee4219b244df17b401664119dfd5b7e52a5659107f1ee3ec210722373
2023/08/29 21:47:20  info unpack layer: sha256:3389e9eb8624f21330c272ba23defc59ba31d46273e38cde475e7256edea80cb
2023/08/29 21:47:20  info unpack layer: sha256:5f4e24b7e32113995e0417eb5c4de5cbef9be70a4a3841a19cdb2fa1f3f12a34
INFO:    Creating SIF file...
INFO:    Build complete: conda-numpy.sif
```

虽然镜像较大，但是可以看见速度仍然很快，这正是开头全程校园内网带来的好处。

`numpy` `job.lsf` 我们对应地使用作业脚本即可。创建一个

```
#BSUB -q 6140ib
#BSUB -n 1

module load singularity/latest

SINGULARITY="singularity run --env MKL_NUM_THREADS=$LSB_DJOB_NUMPROC conda-numpy.sif"
${SINGULARITY} python test.py
```

然后执行

```
bsub < job.lsf
```

`bjobs` `bpeek` 来查看状态，可以看到，任务成功提交并正确执行了。

总结

通过CI/CD自动构建Docker镜像并用于HPC集群计算的简单示例，我们尝试将eScience的下列服务进行了结合：

- `docker.nju.edu.cn` `gcr.nju.edu.cn` `mirror.nju.edu.cn`
- CI/CD `git.nju.edu.cn` `reg.nju.edu.cn`

- hpc.nju.edu.cn

Revision #6
Created 29 August 2023 22:40:04 by LadderOperator
Updated 1 March 2024 16:35:06 by LadderOperator