CI pipeline triggering analysis execution on Analysis Facility

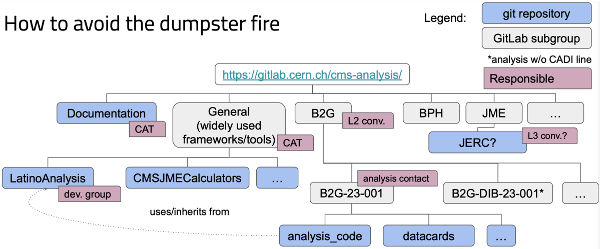
Matteo Bartolini[1], Mattia Lizzo[1], Lorenzo Viliani[1], Tommaso Tedeschi[2]

[1]Università e INFN, Firenze
[2]Università e INFN, Perugia

Bi-weekly WP2 meeting

5 December 2023

Continuous integration with CMS dataset



- The CAT should allow analyzers to setup CI pipelines running on CMS dataset for code checking purposes (e.g. check effect of a commit on cut-flow yields...)
- `https://indico.cern.ch/event/1180058/contributions/5569735/attachments/2718208/4722157/CATcmsweekSept2023-2.pdf`
- Issue: difficult, because authentication is often needed in order to access the dataset→ CERN gitlab CI runners typically don't have it
- Solution: leverage EOS tokens, service in place to provide them, can be activated with a couple of lines in .gitlab-ci.yml

The mkShapeRDF case

- mkShapesRDF, port to RDF of the “latinos” framework, to steer template based analyses with configuration files
- The framework is used, in this case, by the code WpWmJJpolarizations to perform the analysis
- This analysis needs to:
  - access the dataset stored somewhere on EOS
  - submit jobs to condor
  - store the output somewhere
  - Use the output to make plots or run fits
- The condor jobs will run on the Analysis Facility:
  - `https://infn-cms-analysisfacility.readthedocs.io/en/latest/introduction/`
  - EOS and AFS are not mounted → need xrdfs to access datasets stored on eos

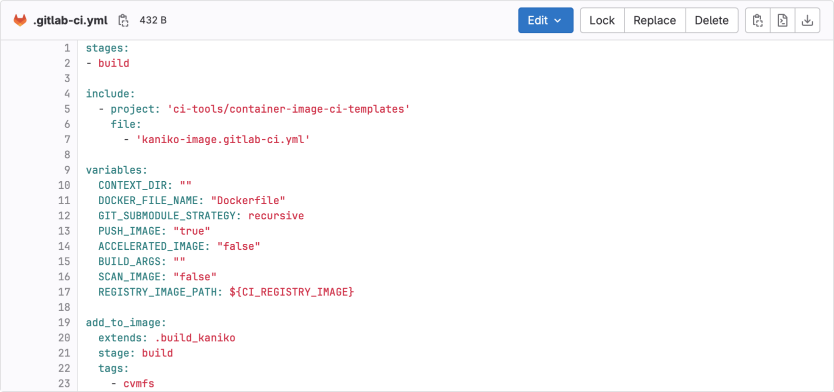
How do we do it?

These are the steps:

- Create a standalone docker image of mkShapesRDF containing all the libraries needed to run it
- The image is used by the analysis code running on the CI runners everytime a new commit is made.
- The CI runners will submit the condor jobs to workers running on the AF
- The workers on the AF will also run the docker image of the framework and perform all the operations

The CI tool

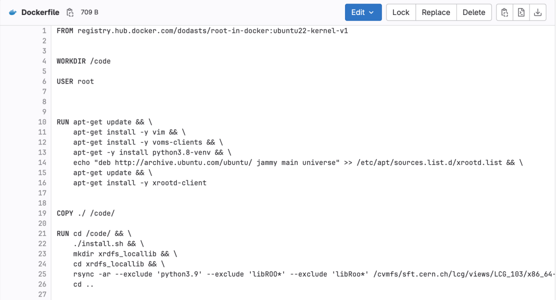
- To build a docker image of your framework the CI tool is needed
- This project has the objective to supply an easy to use gitlab CI template to build images



```yaml
stages:
- build

include:
  - project: 'ci-tools/container-image-ci-templates'
    file:
      - 'kaniko-image.gitlab-ci.yml'

variables:
  CONTEXT_DIR: ""
  DOCKER_FILE_NAME: "Dockerfile"
  GIT_SUBMODULE_STRATEGY: recursive
  PUSH_IMAGE: "true"
  ACCELERATED_IMAGE: "false"
  BUILD_ARGS: ""
  SCAN_IMAGE: "false"
  REGISTRY_IMAGE_PATH: ${CI_REGISTRY_IMAGE}

add_to_image:
  extends: .build_kaniko
  stage: build
  tags:
    - cvmfs
```

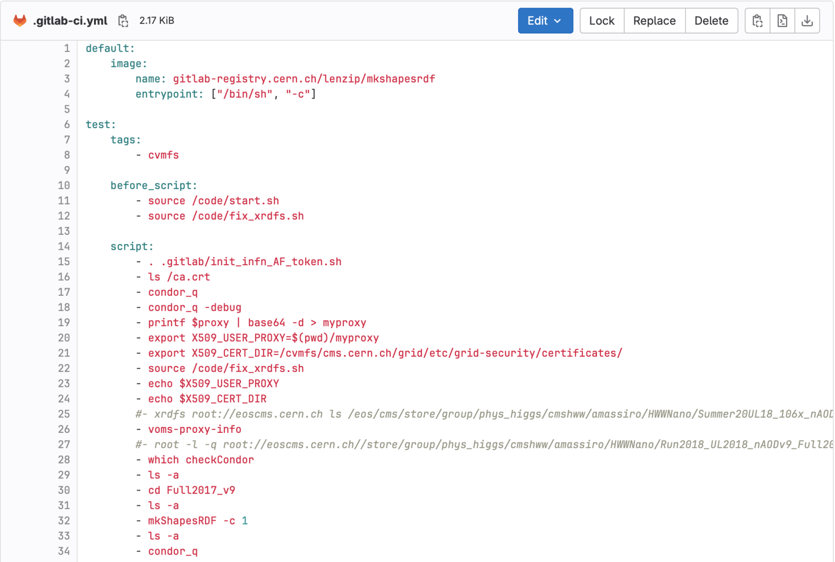
The dockerfile in mkShapesRDF

- The dockerfile used by the CI tool should contain all the commands to build a standalone image of your framework
- In this case our image is built on top of the image of ubuntu 20
- If the building is successfull, the image will be created and pushed to gitlab-registry.cern.ch/lenzip/mkshapesrdf in this case



```
FROM registry.hub.docker.com/dodasts/root-in-docker:ubuntu22-kernel-v1


WORKDIR /code

USER root



RUN apt-get update && \
    apt-get install -y vim && \
    apt-get install -y voms-clients && \
    apt-get -y install python3.8-venv && \
    echo "deb http://archive.ubuntu.com/ubuntu/ jammy main universe" >> /etc/apt/sources.list.d/xrootd.list && \
    apt-get update && \
    apt-get install -y xrootd-client


COPY ./ /code/

RUN cd /code/ && \
    ./install.sh && \
    mkdir xrdfs_locallib && \
    cd xrdfs_locallib && \
    rsync -ar --exclude 'python3.9' --exclude 'libROO*' --exclude 'libRoo*' /cvmfs/sft.cern.ch/lcg/views/LCG_103/x86_64-
    cd ..

```

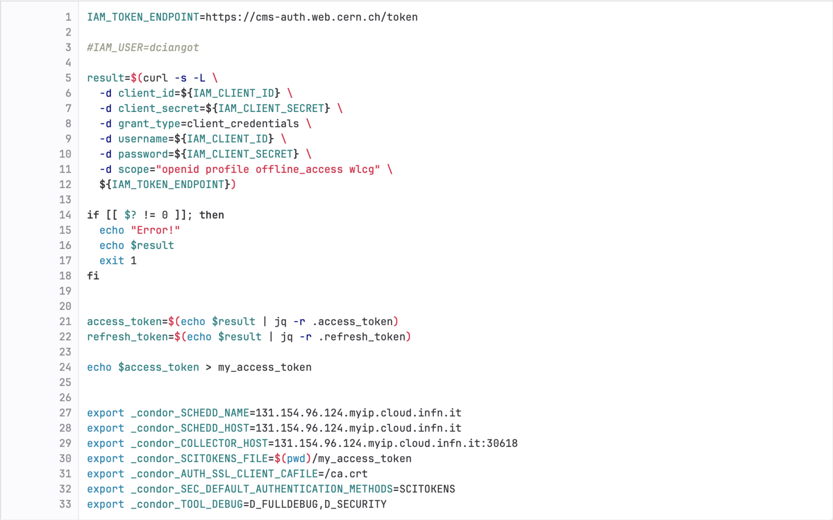
The analysis code:WpWmJJpolarizations



```
default:
    image:
        name: gitlab-registry.cern.ch/lenzip/mkshapesrdf
        entrypoint: ["/bin/sh", "-c"]

test:
    tags:
        - cvmfs

    before_script:
        - source /code/start.sh
        - source /code/fix_xrdfs.sh

    script:
        - . .gitlab/init_infn_AF_token.sh
        - ls /ca.crt
        - condor_q
        - condor_q -debug
        - printf $proxy | base64 -d > myproxy
        - export X509_USER_PROXY=$(pwd)/myproxy
        - export X509_CERT_DIR=/cvmfs/cms.cern.ch/grid/etc/grid-security/certificates/
        - source /code/fix_xrdfs.sh
        - echo $X509_USER_PROXY
        - echo $X509_CERT_DIR
        #- xrdfs root://eoscms.cern.ch ls /eos/cms/store/group/phys_higgs/cmshww/amassiro/HWWNano/Summer20UL18_106x_nAOD
        - voms-proxy-info
        #- root -l -q root://eoscms.cern.ch//store/group/phys_higgs/cmshww/amassiro/HWWNano/Run2018_UL2018_nAODv9_Full20
        - which checkCondor
        - ls -a
        - cd Full2017_v9
        - ls -a
        - mkShapesRDF -c 1
        - ls -a
        - condor_q
```

The AF token

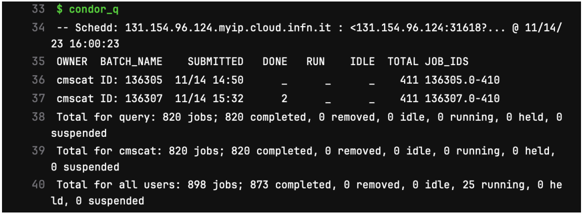
- this script is used to get the token needed to access the condor workers on the AF machine



```
IAM_TOKEN_ENDPOINT=https://cms-auth.web.cern.ch/token

#IAM_USER=dciangot

result=$(curl -s -L \
  -d client_id=${IAM_CLIENT_ID} \
  -d client_secret=${IAM_CLIENT_SECRET} \
  -d grant_type=client_credentials \
  -d username=${IAM_CLIENT_ID} \
  -d password=${IAM_CLIENT_SECRET} \
  -d scope="openid profile offline_access wlcg" \
  ${IAM_TOKEN_ENDPOINT})

if [[ $? != 0 ]]; then
  echo "Error!"
  echo $result
  exit 1
fi


access_token=$(echo $result | jq -r .access_token)
refresh_token=$(echo $result | jq -r .refresh_token)

echo $access_token > my_access_token


export _condor_SCHEDD_NAME=131.154.96.124.myip.cloud.infn.it
export _condor_SCHEDD_HOST=131.154.96.124.myip.cloud.infn.it
export _condor_COLLECTOR_HOST=131.154.96.124.myip.cloud.infn.it:30618
export _condor_SCITOKENS_FILE=$(pwd)/my_access_token
export _condor_AUTH_SSL_CLIENT_CAFILE=/ca.crt
export _condor_SEC_DEFAULT_AUTHENTICATION_METHODS=SCITOKENS
export _condor_TOOL_DEBUG=D_FULLDEBUG,D_SECURITY
```

The proxy

- In order to access the files on eos a token is needed
- The token is personal, has a limited validity, and is generated with the command:
  - voms-proxy-init –rfc –voms cms -valid 192:00
  - base64 -w0 X509USERPROXY
  - go to the gitlab page containing your project, create a variable inside Settings → CI/CD → Variables
  - copy paste the content of x509_user_proxy to that variable
- If everything goes smoothly, you should see the following result and be finally able to submit your jobs to condor via the CI

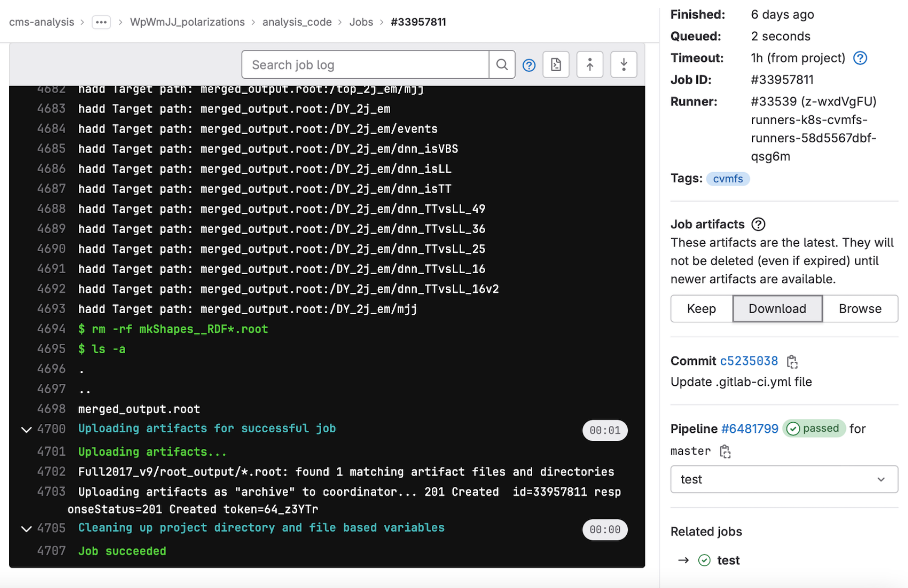


```
$ condor_q
-- Schedd: 131.154.96.124.myip.cloud.infn.it : <131.154.96.124:31618?... @ 11/14/
23 16:00:23
OWNER  BATCH_NAME    SUBMITTED   DONE   RUN    IDLE  TOTAL JOB_IDS
cmscat ID: 136305  11/14 14:50      _      _      _    411 136305.0-410
cmscat ID: 136307  11/14 15:32      2      _      _    411 136307.0-410
Total for query: 820 jobs; 820 completed, 0 removed, 0 idle, 0 running, 0 held, 0
suspended
Total for cmscat: 820 jobs; 820 completed, 0 removed, 0 idle, 0 running, 0 held,
0 suspended
Total for all users: 898 jobs; 873 completed, 0 removed, 0 idle, 25 running, 0 he
ld, 0 suspended
```

Submitting jobs to condor from the CI

- When submitting jobs to Condor from the CI the jdl file should contain a line pointing to the docker image that will be used by the workers:
  - +SingularityImage: /cvmfs/unpacked.cern.ch/gitlab-registry.cern.ch/lenzip/mkshapesrdf:latest/
- Once the jobs have been submitted a scripts is used to check the status of the condor jobs every n seconds to keep the CI busy
- Once all the jobs are done running the script will exit the loop and all data are transferred back to the CI runner, merged together and added to artifacts
- The full pipeline is here: `https://gitlab.cern.ch/cms-analysis/smp/wpwmjj_polarizations/analysis_code/-/jobs/33957811`

Final result



cms-analysis › ••• › WpWmJJ_polarizations › analysis_code › Jobs › **#33957811**

```
hadd Target path: merged_output.root:/top_2j_em/mjj
hadd Target path: merged_output.root:/DY_2j_em
hadd Target path: merged_output.root:/DY_2j_em/events
hadd Target path: merged_output.root:/DY_2j_em/dnn_isVBS
hadd Target path: merged_output.root:/DY_2j_em/dnn_isLL
hadd Target path: merged_output.root:/DY_2j_em/dnn_isTT
hadd Target path: merged_output.root:/DY_2j_em/dnn_TTvsLL_49
hadd Target path: merged_output.root:/DY_2j_em/dnn_TTvsLL_36
hadd Target path: merged_output.root:/DY_2j_em/dnn_TTvsLL_25
hadd Target path: merged_output.root:/DY_2j_em/dnn_TTvsLL_16
hadd Target path: merged_output.root:/DY_2j_em/dnn_TTvsLL_16v2
hadd Target path: merged_output.root:/DY_2j_em/mjj
$ rm -rf mkShapes__RDF*.root
$ ls -a
.
..
merged_output.root
Uploading artifacts for successful job                                00:01
Uploading artifacts...
Full2017_v9/root_output/*.root: found 1 matching artifact files and directories
Uploading artifacts as "archive" to coordinator... 201 Created  id=33957811 resp
onseStatus=201 Created token=64_z3YTr
Cleaning up project directory and file based variables                00:00
Job succeeded
```

Finished: 6 days ago
Queued: 2 seconds
Timeout: 1h (from project)
Job ID: #33957811
Runner: #33539 (z-wxdVgFU) runners-k8s-cvmfs-runners-58d5567dbf-qsg6m
Tags: cvmfs

Job artifacts

These artifacts are the latest. They will not be deleted (even if expired) until newer artifacts are available.

Commit c5235038

Update .gitlab-ci.yml file

Pipeline #6481799 passed for master

Related jobs

→ ✓ **test**

Conclusions

- We have been setting up a CI pipeline running a full latino analysis on an INFN analysis facility
- We overcame the initial struggles with authentication and tokens
- Detailed instructions will be thoroughly documented and made available
- Job submission is entirely based on condor at the moment, but we plan to start experimenting soon the use of dask to improve handling and merging of the full dataset