CI pipeline triggering analysis execution on Analysis Facility

Matteo Bartolini[1], Mattia Lizzo[1], Lorenzo Viliani[1], Tommaso Tedeschi[2]
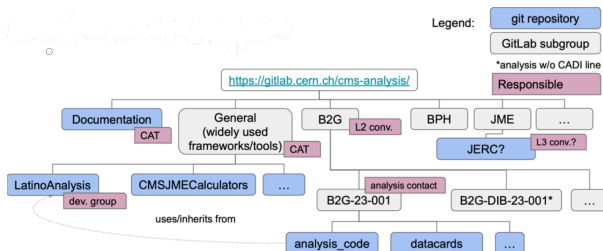
[1]Università e INFN, Firenze
[2]Università e INFN, Perugia

Spoke2 annual meeting

18-20 December 2023

Continuous integration with CMS dataset

- CMS has established a group for the management of analysis tools (CAT)
- CAT provides an area on gitlab.cern.ch for analysis code with instructions to disseminate the use of gitlab CI:
    - Issue: difficult, because authentication is often needed in order to access the dataset$\rightarrow$ CERN gitlab CI runners typically don't have it
    - Solution: Use IAM token to authenticate on AF and voms proxy to access the datasetes

How do we do it?

These are the steps:

- Create a standalone docker image of the needed software stack containing all the libraries needed to run it

  In our case we eliminated the dependency on the CMS software stack

- have a separate reopository with the analysis specific software

- The image is used by the analysis code running on the CI runners everytime a new commit is made.

- The CI runners offload the condor jobs to workers running on the Analysis Facility

- The workers on the Analysis Facility will also run the docker image of the framework and perform all the operations

CI for software stack container

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

```
.gitlab-ci.yml    432 B                                    Edit ▾   Lock  Replace  Delete

1   stages:
2   - build
3
4   include:
5     - project: 'ci-tools/container-image-ci-templates'
6       file:
7         - 'kaniko-image.gitlab-ci.yml'
8
9   variables:
10    CONTEXT_DIR: ""
11    DOCKER_FILE_NAME: "Dockerfile"
12    GIT_SUBMODULE_STRATEGY: recursive
13    PUSH_IMAGE: "true"
14    ACCELERATED_IMAGE: "false"
15    BUILD_ARGS: ""
16    SCAN_IMAGE: "false"
17    REGISTRY_IMAGE_PATH: ${CI_REGISTRY_IMAGE}
18
19  add_to_image:
20    extends: .build_kaniko
21    stage: build
22    tags:
23      - cvmfs
```

CI for analysis specfic code

- The docker image is loaded every time a commit is made in the analysis code

The Analysis Facility token

- Access to the AF resources is done via token (using IAM instance)

```
1   IAM_TOKEN_ENDPOINT=https://cms-auth.web.cern.ch/token
2
3   #IAM_USER=dciangot
4
5   result=$(curl -s -L \
6     -d client_id=${IAM_CLIENT_ID} \
7     -d client_secret=${IAM_CLIENT_SECRET} \
8     -d grant_type=client_credentials \
9     -d username=${IAM_CLIENT_ID} \
10    -d password=${IAM_CLIENT_SECRET} \
11    -d scope="openid profile offline_access wlcg" \
12    ${IAM_TOKEN_ENDPOINT})
13
14  if [[ $? != 0 ]]; then
15    echo "Error!"
16    echo $result
17    exit 1
18  fi
19
20
21  access_token=$(echo $result | jq -r .access_token)
22  refresh_token=$(echo $result | jq -r .refresh_token)
23
24  echo $access_token > my_access_token
25
26
27  export _condor_SCHEDD_NAME=131.154.96.124.myip.cloud.infn.it
28  export _condor_SCHEDD_HOST=131.154.96.124.myip.cloud.infn.it
29  export _condor_COLLECTOR_HOST=131.154.96.124.myip.cloud.infn.it:30618
30  export _condor_SCITOKENS_FILE=$(pwd)/my_access_token
31  export _condor_AUTH_SSL_CLIENT_CAFILE=/ca.crt
32  export _condor_SEC_DEFAULT_AUTHENTICATION_METHODS=SCITOKENS
33  export _condor_TOOL_DEBUG=D_FULLDEBUG,D_SECURITY
```

Final result

Conclusions

- We have been setting up a CI pipeline running a full 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
- It would be nice to understand if we can test this using the cineca infrastructure
- The authentication methods used so far will change in the future $\rightarrow$ using solutions for credential protections