Data Injection: MAGIC

Author: Agustin Bruzzese
Date: 26 Oct 2020

Third exercise for the ESCAPE-DIOS project: Data Injection

Aims and Objective

On the one hand, the main objective of this activity is focused on demonstrating the stability of the current RUCIO configuration, injecting data from the different experiments, and accessing the data in the lake with an application/workflow that does use the data and produce a real output.
At the same time, we would like to disable the greedy option for RSEs as we have used in the previous exercise.

In order to meet the objectives described, the following sections are developed below with the intention of convincingly addressing the incorporation of data from the MAGIC experiments.

Commands used

Set up a rucio-client

Below is our Rucio client configuration to inject data to datalake using python bindings.

Code :

```
import sys,os,os.path,io,json,math,re,time,uuid,random,pytz
import numpy as np
from datetime import datetime
# Set Rucio virtual environment configuration

os.environ['RUCIO_HOME']=os.path.expanduser('~/Rucio-v2/rucio')

from rucio.client.client import *
from rucio.client.rseclient import *
from rucio.rse import rsemanager as rsemgr
from rucio.client.client import Client
from rucio.client.didclient import DIDClient
from rucio.client.replicaclient import ReplicaClient
from rucio.client.downloadclient import DownloadClient
import rucio.rse.rsemanager as rsemgr
from rucio.client.ruleclient import RuleClient
from rucio.client.uploadclient import UploadClient
from rucio.common.exception import (AccountNotFound, Duplicate, RucioException, DuplicateRule,
InvalidObject, DataIdentifierAlreadyExists, FileAlreadyExists, RucioException,
                                    AccessDenied, InsufficientAccountLimit, RuleNotFound, AccessDenied,
```

```
InvalidRSEExpression,
                                InvalidReplicationRule, RucioException, DataIdentifierNotFound,
InsufficientTargetRSEs,
                                ReplicationRuleCreationTemporaryFailed, InvalidRuleWeight,
StagingAreaRuleRequiresLifetime)
from rucio.common.utils import adler32, detect_client_location, execute, generate_uuid, md5, send_trace,
GLOBALLY_SUPPORTED_CHECKSUMS

sys.path.append("/usr/lib64/python3.6/site-packages/")
import gfal2
from gfal2 import Gfal2Context, GError

gfal = Gfal2Context()

# Rucio settings

RSE_origin = 'PIC-INJECT'
RSE_destiny = 'PIC-DCACHE'
RSE_destiny_2 = 'INFN-NA-DPM'
RSE_QOS = 'QOS=FAST'

RSEs = {'RSE_destiny':RSE_destiny, 'RSE_QOS':RSE_QOS, 'RSE_destiny_2':RSE_destiny_2}

account = 'bruzzese'
auth_type = 'x509_proxy'
Default_Scope = 'MAGIC_PIC_BRUZZESE'
Client = Client(account=account)
uploadClient = UploadClient()
rulesClient = RuleClient()
downloadClient = DownloadClient()

print(json.dumps(Client.whoami(), indent=2))
```

Output :

```
{
  "status": "ACTIVE",
  "account": "bruzzese",
  "account_type": "SERVICE",
  "created_at": "2020-02-17T14:23:59",
  "suspended_at": null,
  "updated_at": "2020-02-17T14:23:59",
  "deleted_at": null,
  "email": "bruzzese@pic.es"
}
```

Creation of MOCK data

Code :

```
def generate_random_file(size, copies = 1, filename=None):
        """
        generate big binary file with the specified size in bytes
```

```
        :param filename: the filename
        :param size: the size in bytes
        :param copies: number of output files to generate

        """
        n_files = []
        n_files = np.array(n_files, dtype = np.float32)
        for i in range(copies):

        if filename == None :
        date = str(datetime.today().strftime('%Y%m%d'))
        run = str(random.randint(10000000,99999999))
        file = date + '_M1_' + run + '.005_D_1ES1959_650-W0.40_000.root'

        if os.path.exists(file) :
        print ("File %s already exist" %file)

        else:
        print ("File %s not exist" %file)
        try :
                newfile = open(file, "wb")
                newfile.seek(size)
                newfile.write(b"\0")
                newfile.close ()
                os.stat(file).st_size
                print('random file with size %f generated ok'%size)
                n_files = np.append(n_files, file)
        except :
                print('could not be generate file %s'%file)

        return(n_files)
list_files = generate_random_file(size=random.randint(10,99), copies=1)

print(list_files)
```

Output :

```
File 20201026_M1_36973941.005_D_1ES1959_650-W0.40_000.root not exist
random file with size 58.000000 generated ok
['20201026_M1_36973941.005_D_1ES1959_650-W0.40_000.root']
```

Used Rucio commands for data injection, replication, and delete rules

Code :

```
result_dict = dict()
for n in range(0, len(list_files)) :

        name_file = list_files[n]
        print(name_file)

        """
        generate a dictionary with the information for the upload
        :param path: the filename
        :param rse: the destination RSE name
        :param did_scope: The scope of the DID.
        :param lifetime: Seconds of DID lifetime
        """

        file = {'path': "./"+name_file, 'rse': RSE_destiny, 'did_scope': Default_Scope,
'lifetime':5}

        result_dict[name_file] = {}
        result_dict[name_file]['Scope'] = Default_Scope
        result_dict[name_file]['Replicated'] = {'Local' : {'registered':
datetime.utcnow().replace(tzinfo=pytz.utc).strftime('%Y-%m-%dT%H:%M:%SZ'),'checksum':getchec
ksum(name_file), 'path':os.path.join(os.getcwd(),name_file)}}

        # Perform upload
        client_upload = uploadClient.upload([file])

        # Create a dataset
        dataset_name = look_for_run(name_file)

        createDataset(new_dataset = dataset_name, new_scope = Default_Scope)

        registerIntoGroup(n_file = name_file, new_dataset = str(dataset_name))

        for rse in RSEs :
        # Contruct a dictionary with the destiny RSE
        if name_file in result_dict :
                temp_dict = dict()
                temp_dict[name_file] = {}
                temp_dict[name_file]['Replicated'] = {RSEs[rse] : {'registered':
datetime.utcnow().replace(tzinfo=pytz.utc).strftime('%Y-%m-%dT%H:%M:%SZ')}}


result_dict[name_file]['Replicated'].update(temp_dict[name_file]['Replicated'])

        rule = addReplicaRule(destRSE = RSEs[rse], group = dataset_name, new_scope =
Default_Scope)

        # update a rule so it 'll be automatically be deleted once it has been successfully
```

```
replicated
        update = Client.update_replication_rule(rule_id=rule, options={'lifetime': 60,
'child_rule_id':rule, 'purge_replicas':True})


        if json_check() == True :
        result_dict.update(stateCheck())

# Save the uploads, replication and time into a json file
json_write(result_dict)
```

Output :


```
{
        "20201026_M1_36973941.005_D_1ES1959_650-W0.40_000.root": {
        "Scope": "MAGIC_PIC_BRUZZESE",
        "Replicated": {
        "Local": {
                "registered": "2020-10-26T11:26:21Z",
                "checksum": "22031453e4c3a1a0d47b0b97d83d8984",
                "path":
"/nfs/pic.es/user/b/bruzzese/Rucio-v2/20201026_M1_36973941.005_D_1ES1959_650-W0.40_000.root"
        },
        "PIC-DCACHE": {
                "registered": "2020-10-26T11:26:24Z",
        },
        "QOS=FAST": {
                "registered": "2020-10-26T11:26:25Z"
        },
        "INFN-NA-DPM": {
                "registered": "2020-10-26T11:26:25Z"
        }
        }
        },
        "20201026_M1_38290737.005_D_1ES1959_650-W0.40_000.root": {
        "Scope": "MAGIC_PIC_BRUZZESE",
        "Replicated": {
        "Local": {
                "registered": "2020-10-26T11:23:54Z",
                "checksum": "0b867e53c1d233ce9fe49d54549a2323",
                "path":
"/nfs/pic.es/user/b/bruzzese/Rucio-v2/20201026_M1_38290737.005_D_1ES1959_650-W0.40_000.root"
        },
        "PIC-DCACHE": {
                "registered": "2020-10-26T11:23:57Z",
        },
        "QOS=FAST": {
                "registered": "2020-10-26T11:23:57Z"
        },
        "INFN-NA-DPM": {
                "registered": "2020-10-26T11:23:58Z"
        }
        }
        },
        "20201026_M1_95000364.005_D_1ES1959_650-W0.40_000.root": {
```

```
        "Scope": "MAGIC_PIC_BRUZZESE",
        "Replicated": {
        "Local": {
                "registered": "2020-10-26T11:23:14Z",
                "checksum": "0b91f1d54f932dc6382dc69f197900cf",
                "path":
"/nfs/pic.es/user/b/bruzzese/Rucio-v2/20201026_M1_95000364.005_D_1ES1959_650-W0.40_000.root"
        },
        "PIC-DCACHE": {
                "registered": "2020-10-26T11:23:17Z",
        },
        "QOS=FAST": {
                "registered": "2020-10-26T11:23:18Z"
        },
        "INFN-NA-DPM": {
                "registered": "2020-10-26T11:23:18Z"
        }
        }
        },
        "20201026_M1_76842933.005_D_1ES1959_650-W0.40_000.root": {
        "Scope": "MAGIC_PIC_BRUZZESE",
        "Replicated": {
        "Local": {
                "registered": "2020-10-26T11:20:32Z",
                "checksum": "53553242d57214aaa5726a09b05fe7bc",
                "path":
"/nfs/pic.es/user/b/bruzzese/Rucio-v2/20201026_M1_76842933.005_D_1ES1959_650-W0.40_000.root"
        },
        "PIC-DCACHE": {
                "registered": "2020-10-26T11:20:34Z",
        },
        "QOS=FAST": {
                "registered": "2020-10-26T11:20:35Z"
        },
        "INFN-NA-DPM": {
                "registered": "2020-10-26T11:20:35Z"
        }
        }
        },
        "20201026_M1_95088901.005_D_1ES1959_650-W0.40_000.root": {
        "Scope": "MAGIC_PIC_BRUZZESE",
        "Replicated": {
        "Local": {
                "registered": "2020-10-26T11:10:37Z",
                "checksum": "e6b62b76fb2eb2a0e0adde0c067da680"
        },
        "PIC-DCACHE": {
                "registered": "2020-10-26T11:10:40Z",
        },
        "QOS=FAST": {
                "registered": "2020-10-26T11:10:40Z"
        },
        "INFN-NA-DPM": {
                "registered": "2020-10-26T11:10:41Z"
        }
        }
        },
```

Used Rucio commands for accessing data, perform checksum and save result in a json file

Code :

```
# Download a file fomr a datalake, and perform a checksum.
# Then save it in a json file
for n in range(0, len(list_files)) :

        name_file = list_files[n]
        download =
downloadClient.download_dids(items=[{'did':'{}:{}'.format(Default_Scope,name_file)}], num_threads=2,
trace_custom_fields={}, traces_copy_out=None)

        result_dict = stateCheck()
        if json_check() == True :
        if name_file in result_dict.keys() :
        temp_dict = dict()
        temp_dict[name_file] = {}
        temp_dict[name_file]['Replicated'] = {download[0]['sources'][0]['rse'] : {'downloaded':
datetime.utcnow().replace(tzinfo=pytz.utc).strftime('%Y-%m-%dT%H:%M:%SZ'),'checksum':gfal.checksum('file
:///'+download[0]['dest_file_paths'][0],'md5')}}

result_dict[name_file]['Replicated'][download[0]['sources'][0]['rse']].update(temp_dict[name_file]['Repl
icated'][download[0]['sources'][0]['rse']])
        json_write(result_dict)
        os.remove(download[0]['dest_file_paths'][0])
        os.remove(result_dict[name_file]['Replicated']['Local']['path'])
```

Output :

```
{
        "20201026_M1_36973941.005_D_1ES1959_650-W0.40_000.root": {
        "Scope": "MAGIC_PIC_BRUZZESE",
        "Replicated": {
        "Local": {
                "registered": "2020-10-26T11:26:21Z",
                "checksum": "22031453e4c3a1a0d47b0b97d83d8984",
                "path":
"/nfs/pic.es/user/b/bruzzese/Rucio-v2/20201026_M1_36973941.005_D_1ES1959_650-W0.40_000.root"
        },
        "PIC-DCACHE": {
                "registered": "2020-10-26T11:26:24Z",
                "downloaded": "2020-10-26T11:26:26Z",
                "checksum": "22031453e4c3a1a0d47b0b97d83d8984"
        },
        "QOS=FAST": {
                "registered": "2020-10-26T11:26:25Z"
        },
        "INFN-NA-DPM": {
                "registered": "2020-10-26T11:26:25Z"
        }
        }
```

```
        },
        "20201026_M1_38290737.005_D_1ES1959_650-W0.40_000.root": {
        "Scope": "MAGIC_PIC_BRUZZESE",
        "Replicated": {
        "Local": {
                "registered": "2020-10-26T11:23:54Z",
                "checksum": "0b867e53c1d233ce9fe49d54549a2323",
                "path":
"/nfs/pic.es/user/b/bruzzese/Rucio-v2/20201026_M1_38290737.005_D_1ES1959_650-W0.40_000.root"
        },
        "PIC-DCACHE": {
                "registered": "2020-10-26T11:23:57Z",
                "downloaded": "2020-10-26T11:23:59Z",
                "checksum": "0b867e53c1d233ce9fe49d54549a2323"
        },
        "QOS=FAST": {
                "registered": "2020-10-26T11:23:57Z"
        },
        "INFN-NA-DPM": {
                "registered": "2020-10-26T11:23:58Z"
        }
        }
        },
        "20201026_M1_95000364.005_D_1ES1959_650-W0.40_000.root": {
        "Scope": "MAGIC_PIC_BRUZZESE",
        "Replicated": {
        "Local": {
                "registered": "2020-10-26T11:23:14Z",
                "checksum": "0b91f1d54f932dc6382dc69f197900cf",
                "path":
"/nfs/pic.es/user/b/bruzzese/Rucio-v2/20201026_M1_95000364.005_D_1ES1959_650-W0.40_000.root"
        },
        "PIC-DCACHE": {
                "registered": "2020-10-26T11:23:17Z",
                "downloaded": "2020-10-26T11:23:19Z",
                "checksum": "0b91f1d54f932dc6382dc69f197900cf"
        },
        "QOS=FAST": {
                "registered": "2020-10-26T11:23:18Z"
        },
        "INFN-NA-DPM": {
                "registered": "2020-10-26T11:23:18Z"
        }
        }
        },
        "20201026_M1_76842933.005_D_1ES1959_650-W0.40_000.root": {
        "Scope": "MAGIC_PIC_BRUZZESE",
        "Replicated": {
        "Local": {
                "registered": "2020-10-26T11:20:32Z",
                "checksum": "53553242d57214aaa5726a09b05fe7bc",
                "path":
"/nfs/pic.es/user/b/bruzzese/Rucio-v2/20201026_M1_76842933.005_D_1ES1959_650-W0.40_000.root"
        },
        "PIC-DCACHE": {
                "registered": "2020-10-26T11:20:34Z",
                "downloaded": "2020-10-26T11:20:36Z",
                "checksum": "53553242d57214aaa5726a09b05fe7bc"
        },
```

```
        "QOS=FAST": {
                "registered": "2020-10-26T11:20:35Z"
        },
        "INFN-NA-DPM": {
                "registered": "2020-10-26T11:20:35Z"
        }
        }
        },
        "20201026_M1_95088901.005_D_1ES1959_650-W0.40_000.root": {
        "Scope": "MAGIC_PIC_BRUZZESE",
        "Replicated": {
        "Local": {
                "registered": "2020-10-26T11:10:37Z",
                "checksum": "e6b62b76fb2eb2a0e0adde0c067da680"
        },
        "PIC-DCACHE": {
                "registered": "2020-10-26T11:10:40Z",
                "downloaded": "2020-10-26T11:12:38Z",
                "checksum": "e6b62b76fb2eb2a0e0adde0c067da680"
        },
        "QOS=FAST": {
                "registered": "2020-10-26T11:10:40Z"
        },
        "INFN-NA-DPM": {
                "registered": "2020-10-26T11:10:41Z"
        }
        }
        },
```


Deactivation of the greed option in the RSEs

```
[bruzzese@rucio01 ~]$ rucio-admin rse set-attribute --key greedyDeletion --value False --rse PIC-INJECT
/home/bruzzese/.local/lib/python2.7/site-packages/requests/__init__.py:91: RequestsDependencyWarning:
urllib3 (1.25.9) or chardet (2.2.1) doesn't match a supported version!
  RequestsDependencyWarning)
Added new RSE attribute for PIC-INJECT: greedyDeletion-False
```

Data injection script output

Below we include the output of the script that can be found at https://github.com/pic-es/rucio-client/blob/pic-dev-test/ESCAPE/ESCAPE-inject-read-delete.ipynb and the executable script https://github.com/pic-es/rucio-client/blob/pic-dev-test/ESCAPE/Rucio-inject-read-delete.py . Please see the next section for more details on performance and data injection over time.

Performance and Failed output messages

In this section, we describe the dataflow for MAGIC mock data that was run in the ESCAPE Data Lake. The data follows a specific workflow that is described below, which meets the aims described in previous sections.

First, a file is generated locally and it is uploaded to the destination RSE PIC-DCACHE. From this upload two replication rules are created organized in datasets that contain the file. The replication rules destination are the RSE of INFN-NA-DPM and PIC QOS. We proceed to update these rules, so that they are automatically deleted once the files have been successfully transferred. In this first part, a json file is generated with detailed information on the name of the uploaded file, its checksum (calculated from the local file), the replicated RSEs, and it's time (output-3).

Then, the file is downloaded locally from the datalake, and then a checksum is carried out to fulfill the point of accessing the data and performing actions on it. Finally, the json file is updated with the checksum value of the downloaded file (output-4).

Finally, the counting plugin was installed to monitor the number of files at the RSE PIC-DCACHE. This way, we can see that the elimination of a rule (Fig. 1) does indeed trigger the deletion of its corresponding file in the RSE, shown by the drop of the green line.

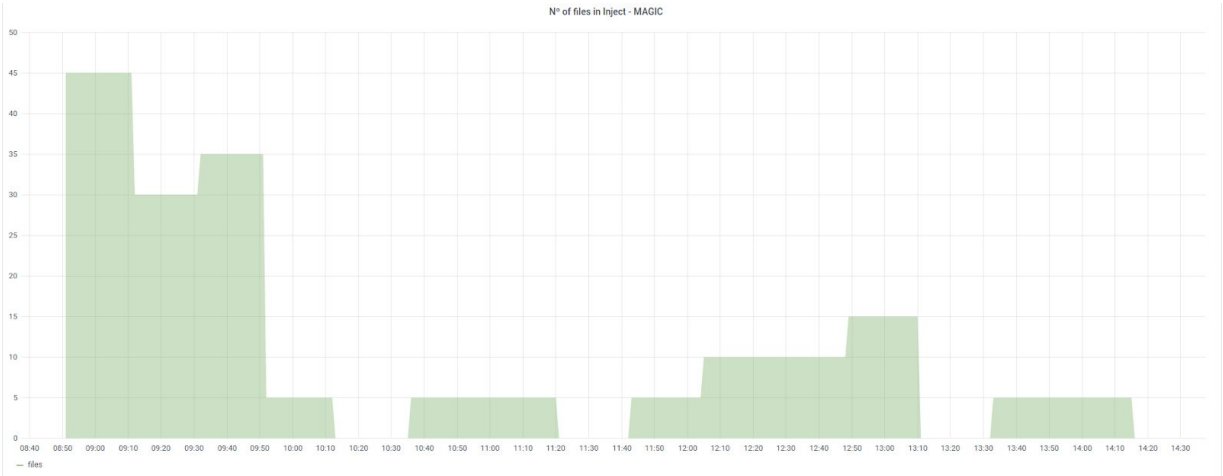

Figure 1. Deletion of files at PIC-DCACHE

To automate these processes, cron is configured to run the workflow every hour. Even so, in the following link, you can see an example of running the script: https://github.com/pic-es/rucio-client/blob/pic-dev-test/ESCAPE/ESCAPE-inject-read-delete.ipynb

Difficulties encountered if any

-

Personal feedback