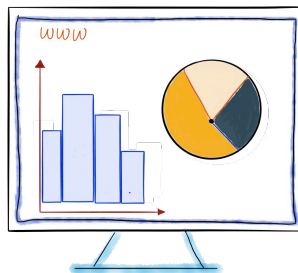


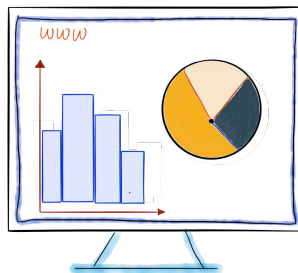
# ESCAPE: a review of idea for Data + Analysis challenge using ATLAS Open Data



Arturo Sánchez Pineda - LAPP

April-May, 2021

# ESCAPE: a review of idea for Data + Analysis challenge using ATLAS Open Data



Arturo Sánchez Pineda - LAPP

Aug-Sep-Oct, 2021

# Overview


**We attempt to describe a series of exercises to perform during DAC21**

<https://docs.google.com/document/d/1ma8dRSc6wrGNKqxn7ZWNjvJ16QNgvQF6mgY7KsCxRck/>

- Data “multiplication” where multiple version of the same data is generated, simulating a data-augmentation process
  - Allowing us to have a sustancial quantity data to be injected to the current analysis examples
- Writing of such “multiplied” data back to the Datalake
  - Defining different RSEs → e.g. LAPP/CNRS and Napoli/INFN
- Examples include the analysis of data stored in the Datalake
  - Writing back the results into the Datalake (small files of ~< 1Mb size each)
  - Analysis can be performed using CLI or the JupyterLab UI
- We are creating instructions for users that can be part of the challenge

**ATLAS** has a rich physics program, from Standard Model searches and measurements, the quest of Dark Matter, and many others. For more visit <https://atlas.cern/discover/physics>

### The Higgs boson

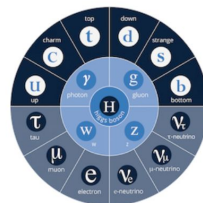


**What is the Higgs boson and why does it matter?**

Physicists describe particle interactions using the mathematics of field theory, in which forces are carried by intermediate particles called bosons. Photons, for example, are bosons carrying the electromagnetic force. In 1964, the only mathematically consistent theory required bosons to be massless. Yet, experiment showed that the carriers of the weak nuclear interaction – the W and Z bosons – had large masses. To solve this problem, three teams of theorists: Robert Brout and François Englert; Peter Higgs; Gerald Guralnik, Carl Hagen, and Tom Kibble independently proposed a solution now referred to as the Brout-Englert-Higgs (BEH) mechanism.

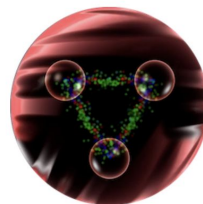
#### What are the basic building blocks of matter?

[The Standard Model](#) describes the elementary subatomic particles of the universe which have been experimentally seen. ATLAS studies these particles and searches for others to determine if the particles we know are indeed elementary or if they are in fact composed of other more fundamental ones.



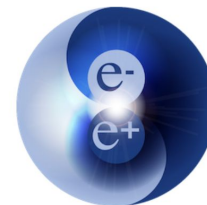
#### What happened to antimatter?

By searching for imbalances in the production of matter and antimatter, we seek to understand why our universe appears to comprise only matter.



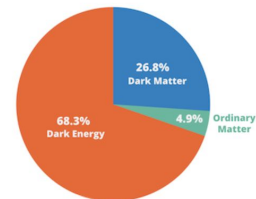
#### What are the forces that govern their interactions?

The Standard Model also describes the fundamental forces of Nature and how they act between fundamental particles. Possible discoveries at the LHC could validate models, such as those incorporating Supersymmetry, where the forces unify at very high energies.



#### What is “dark matter”?

Astronomical measurements support the existence of matter that cannot be directly seen. The hermetic construction of ATLAS, however, makes it possible to search for this “[dark matter](#)”.



# Data

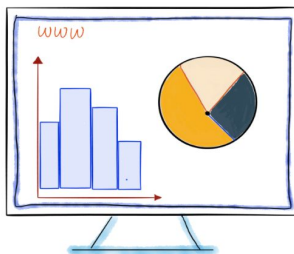
# ATLAS Open Data samples

The ATLAS Collaboration current approach on the release of datasets is intended for Education, Training and Outreach activities around the World. In order to fulfil that objective, the ATLAS Open Data project was created.

**ATLAS Open Data** project aims to provide data and tools to students, as well as teachers and lecturers, to help educate them in physics analysis techniques used in experimental particle physics. Ideally, sharing data collected by the ATLAS experiment aims to generate excitement and enthusiasm for fundamental research, inspiring physicists of the future.

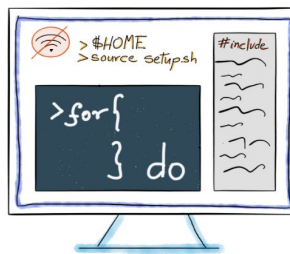
<http://opendata.atlas.cern/about>

Jupyter Notebooks



Let's run some real code and visualisations on your browser

C++/Python frameworks



Perform real HEP analysis as the ATLAS Physicists

Virtual Machine(s)



Slow Internet? run the analyses with minimal installation

# The ATLAS Open Data and analysis examples (i.e. the c++ framework) release in 2020

<https://cds.cern.ch/record/2707171>



ATLAS PUB Note

PUB-OTRC-2020-01

28th April 2020



## Review of the 13 TeV ATLAS Open Data release

The ATLAS Collaboration

The ATLAS Collaboration is releasing a new set of proton-proton collision data to the public for educational purposes. The data has been collected by the ATLAS detector at the Large Hadron Collider at a centre-of-mass energy  $\sqrt{s} = 13$  TeV during the year 2016 and corresponds to an integrated luminosity of  $10 \text{ fb}^{-1}$ . This dataset is accompanied by simulated events describing both several Standard Model processes, as well as hypothetical Beyond Standard Model signal production. Associated computing tools are provided to make the analysis of the dataset easily accessible. This document summarises the properties of the 13 TeV ATLAS Open Data set and the available analysis tools. Several examples intended as a starting point for further analysis work by users are shown. The general aim of the dataset and tools released is to provide user-friendly and straightforward interactive interfaces to replicate the procedures used by high-energy-physics researchers and enable users to experience the analysis of particle-physics data in educational environments.

- This publication contains the analyses that work as validation of the datasets and the software (framework, see next)

ATLAS Note	
Report number	ATL-OREACH-PUB-2020-001
Title	<b>Review of the 13 TeV ATLAS Open Data release</b>
Corporate Author(s)	The ATLAS collaboration
Collaboration	ATLAS Collaboration
Imprint	24 Jan 2020 - 28 p.
Note	All figures including auxiliary figures are available at <a href="https://atlas.web.cern.ch/Atlas/GROUPS/PHYSICS/PUBNOTES/ATL-OREACH-PUB-2020-001">https://atlas.web.cern.ch/Atlas/GROUPS/PHYSICS/PUBNOTES/ATL-OREACH-PUB-2020-001</a>
In:	24th International Conference on Computing in High Energy and Nuclear Physics, Adelaide, Australia, 4 - 8 Nov 2019
Subject category	Particle Physics - Experiment
Accelerator/Facility, Experiment	CERN LHC ; ATLAS
Free keywords	ATLAS ; outreach ; open access ; open data ; education ; open source ; open science ; public data
Abstract	The ATLAS Collaboration is releasing a new set of proton-proton collision data to the public for educational purposes. The data has been collected by the ATLAS detector at the Large Hadron Collider at a centre-of-mass energy $\sqrt{s} = 13$ TeV during the year 2016 and corresponds to an integrated luminosity of $10 \text{ fb}^{-1}$ . This dataset is accompanied by simulated events describing both several Standard Model processes, as well as hypothetical Beyond Standard Model signal production. Associated computing tools are provided to make the analysis of the dataset easily accessible. This document summarises the properties of the 13 TeV ATLAS Open Data set and the available analysis tools. Several examples intended as a starting point for further analysis work by users are shown. The general aim of the dataset and tools released is to provide user-friendly and straightforward interactive interfaces to replicate the procedures used by high-energy-physics researchers and enable users to experience the analysis of particle-physics data in educational environments.

ATL-OREACH-PUB-2020-001  
24 Jan 2020



# ATLAS Open Data datasets in the Datalake

- Samples in the Datalake
  - All the 13 TeV and 8 TeV ATLAS Open Data samples
  - 16 datasets → 940 samples (ROOT files)
  - ~ 320 GB
  - Scope used: **ATLAS\_OD\_EDU** (for **ATLAS** Open **D**ata for **EDU**cation)
  - Source of the datasets:  
<http://opendata.atlas.cern/samples-13tev/> &  
<http://opendata.atlas.cern/samples-8tev/>
  - Another [set of 10 ROOT files](#) to come (dedicated Jet MC samples) → 1 dataset, ~21 GB.



# Software

# ATLAS Open Data → C++ examples framework



SM Higgs boson production in the  $H \rightarrow ZZ$  decay channel in the four-lepton final state

## To run C++ analyses

More computational-complex  
particle physics analysis  
examples using the existing  
publicly available data

More in [Opendata.atlas.cern -  
documentation 13 TeV - physics](https://opendata.atlas.cern/documentation/13%20TeV%20-%20physics)

Also use PROOF, adding a  
parallel component to the  
examples.

### Physics analysis examples

Overview of physics analysis  
examples

Brief introduction to the physics  
of the Higgs boson

SM W-boson production in the  
single-lepton final state

Single-top-quark production in  
the single-lepton final state

Top-quark pair production in the  
single-lepton final state

SM Z-boson production in the  
two-lepton final state

SM Higgs boson production in  
the  $H \rightarrow WW$  decay channel in  
the two-lepton final state

Search for supersymmetric  
particles in the two-lepton final  
state

SM WZ diboson production in  
the three-lepton final state

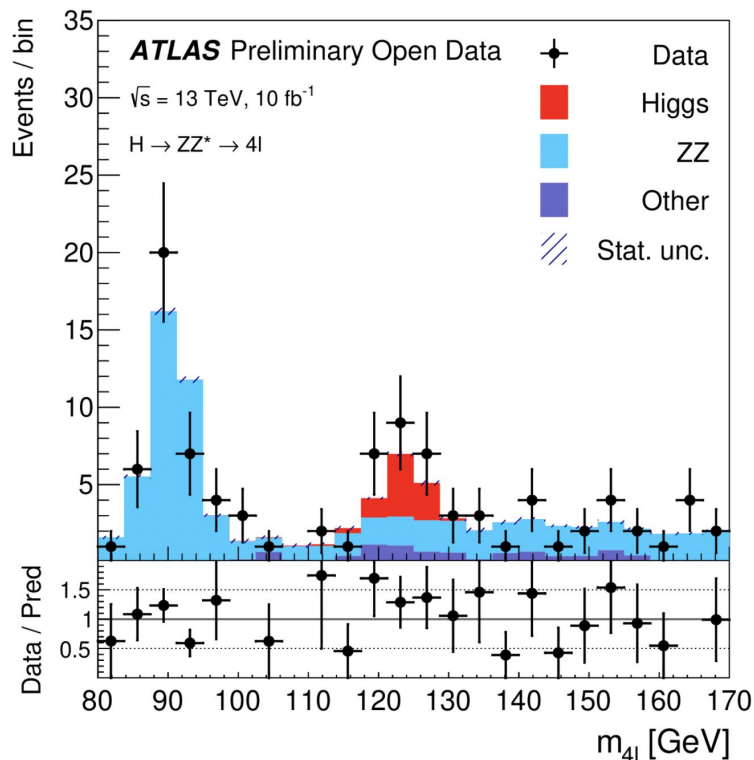
SM ZZ diboson production in the  
four-lepton final state

[SM Higgs boson production in  
the  \$H \rightarrow ZZ\$  decay channel in the  
four-lepton final state](#)

SM Z-boson production in the  
two-tau-lepton final state

Search for BSM  $Z' \rightarrow t\bar{t}$  in the  
single-lepton boosted final state

SM Higgs boson production in  
the  $H \rightarrow \gamma\gamma$  decay channel in the



# Analysis examples software

**The framework** makes use of the C++ language and is interfaced with ROOT, analysis framework by CERN. After cloning/downloading the repository, the only things you need to setup are: you need to have the ROOT framework and a gcc compiler.

The current version of the framework was compiled using gcc v6.20 and ROOT v6.10.04.

Currently, the framework can access the samples in two ways:

- reading them online directly (by default, they are stored in a GitHub repository);
- reading them from a local storage (the samples need to be downloaded locally).

The framework consists of two main parts:

- the analysis part, located within the "Analysis" directory: it performs the particular object selection and stores the output histograms;
- the plotting part, located within the "Plotting" directory: it makes the final Data / Prediction plots.

## Software's description

- Build a framework based on ROOT to analyse high energy physics datasets
- Use C++ programming language
  - One of the main programming languages for high energy physics
  - This is one important piece still missing in the set of ATLAS Open Data public analysis codes
  - Improve the speed and the ability of running in multiple CPU cores at once.
- The framework contains
  - All the needed pieces to run, edit and create physics analyses
  - Seven cut-and-count physics analyses
  - Documentation to guide the user on how to include a new analysis using the same datasets.

# Analysis examples software

## *Ongoing* OSSR Integration

<https://escape2020.pages.in2p3.fr/virtual-environment/home/tsp-higgs/atlas-analyses/> (**Warning!** under construction)

## The code

@Zenodo ESCAPE community

DOI [10.5281/zenodo.5501338](https://doi.org/10.5281/zenodo.5501338)



ATLAS Open Data GitHub

February 10, 2020

Software Open Access

### ATLAS Open Data 13 TeV analysis C++ framework

 Serkin, Leonid;  Sanchez Pineda, Arturo

A repository with 12 high energy physics analysis examples using the ATLAS Open Data 13 TeV dataset released in 2020. It is written in C++ and some bash scripts.

\* Documentation of the code: <http://opendata.atlas.cern/release/2020/documentation/frameworks/cpp.html>

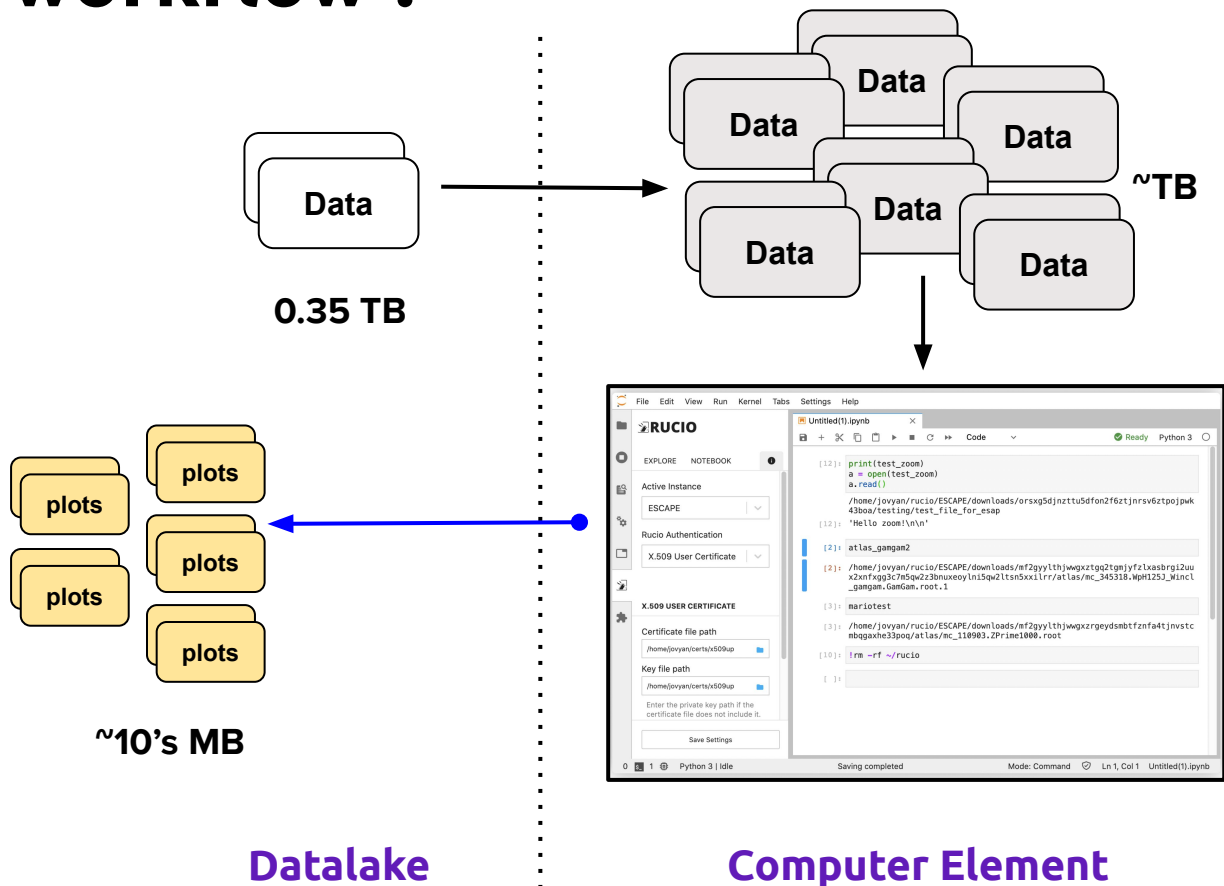
\* Documentation of the analysis: <http://opendata.atlas.cern/release/2020/documentation/physics/intro.html>

more in <http://opendata.atlas.cern> and <https://atlas.cern/updates/press-statement/13-tev-open-data>

# How looks our workflow ?

We will run the analysis examples over the “multiplied” data

- This will help to simulate longer analysis that can last up to several hours
- The original analysis examples usually can take from a few minutes to up to 2-3 hours
- We will also run multiple analysis Jupyter notebooks



# Data Multiplication

# Artificial multiplication of the data

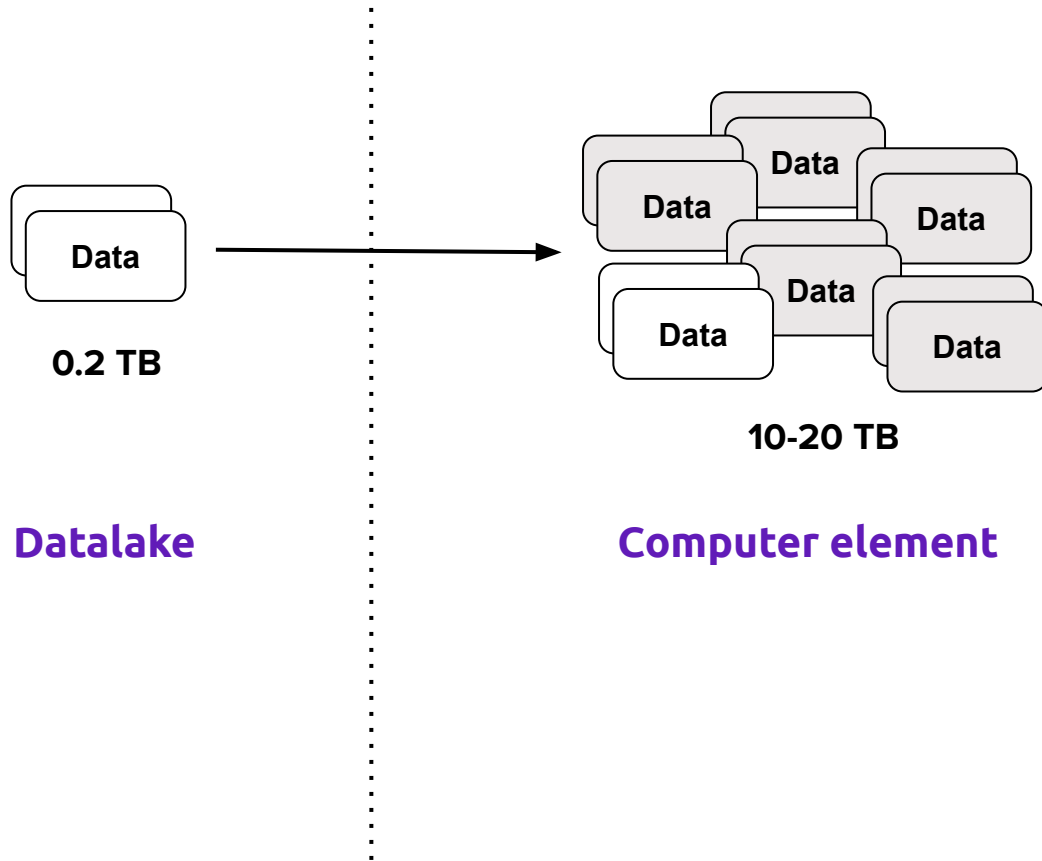
ROOT files can be added when they share an internal structure (i.e. same trees inside). Called “hadd”

We can profit from that property to artificially multiply the datasets.

This process allows augmenting the data to any arbitrary value.

We can use that augmented data to run the analysis examples

- Of course, the results are meaningless.



# Artificial multiplication of the data

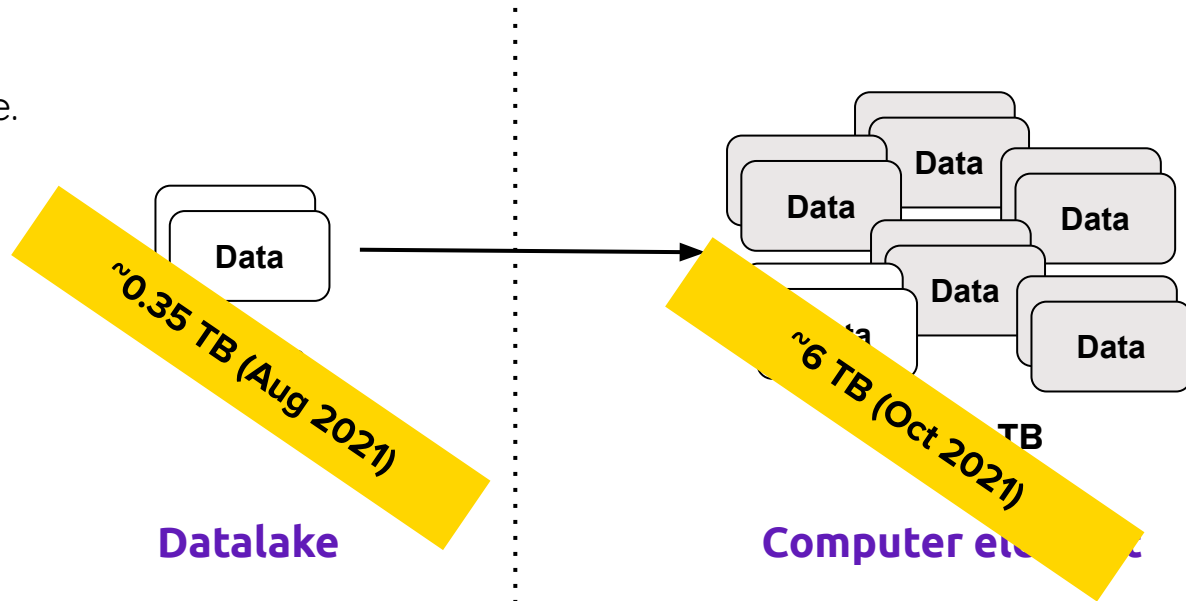
ROOT files can be added when they share an internal structure (i.e. same trees inside). Called “hadd”

We can profit from that property to artificially multiply the datasets.

This process allows augmenting the data to any arbitrary value.

We can use that augmented data to run the analysis examples

- Of course, the results are meaningless.





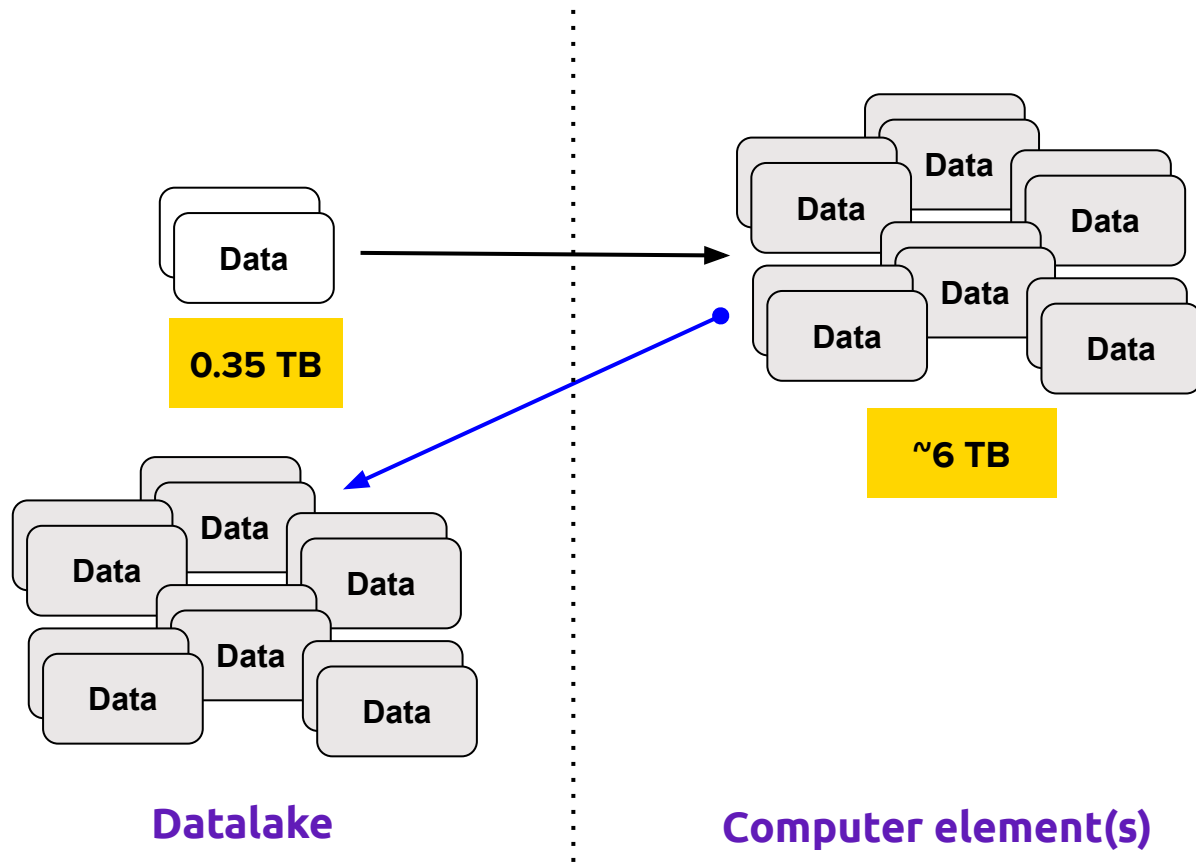
# Artificial multiplication of the data

The “augmented” data

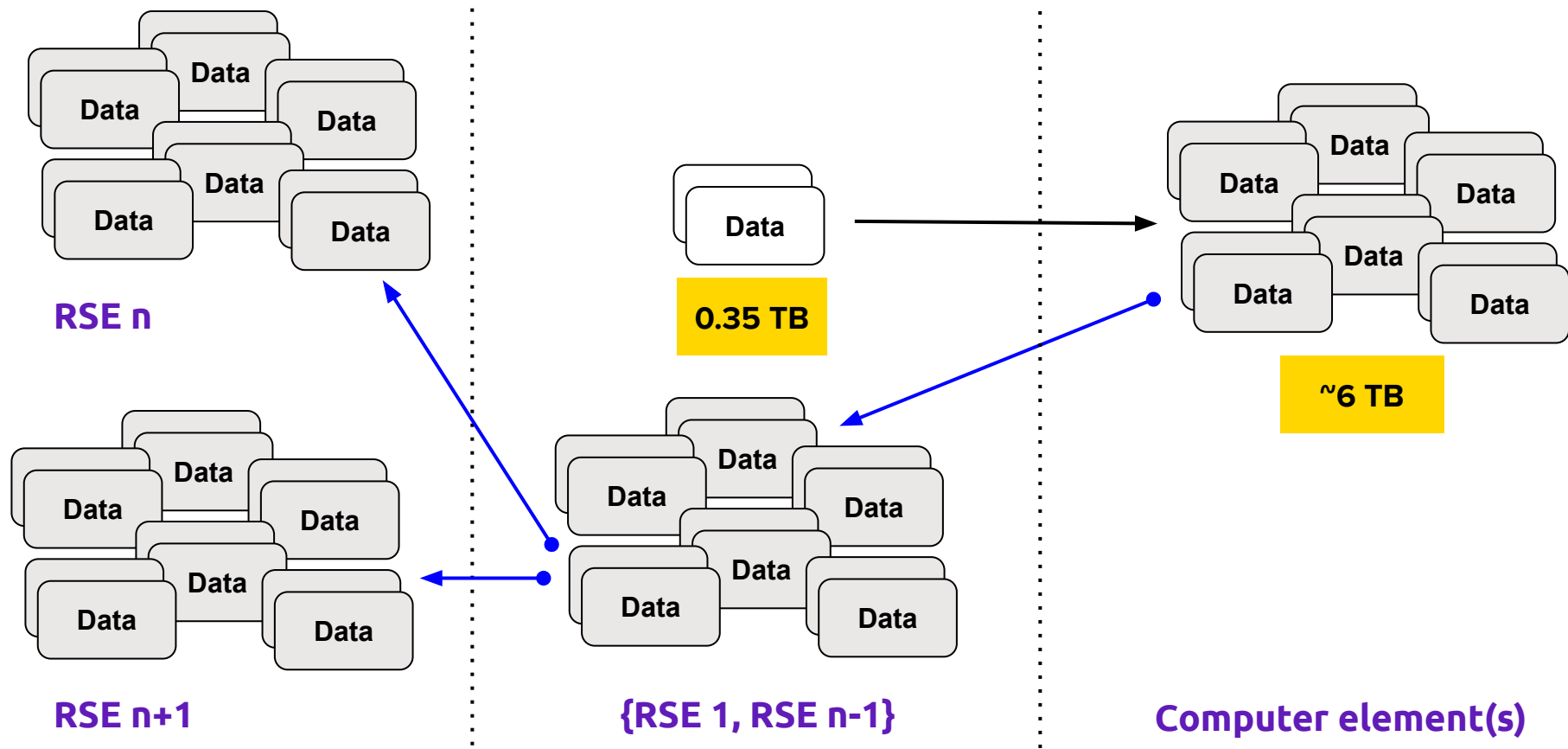
- More and/or larger files
- The process to add multiples files also use computing

After (or during the process), the code automatically can write such new data back to the Datalake

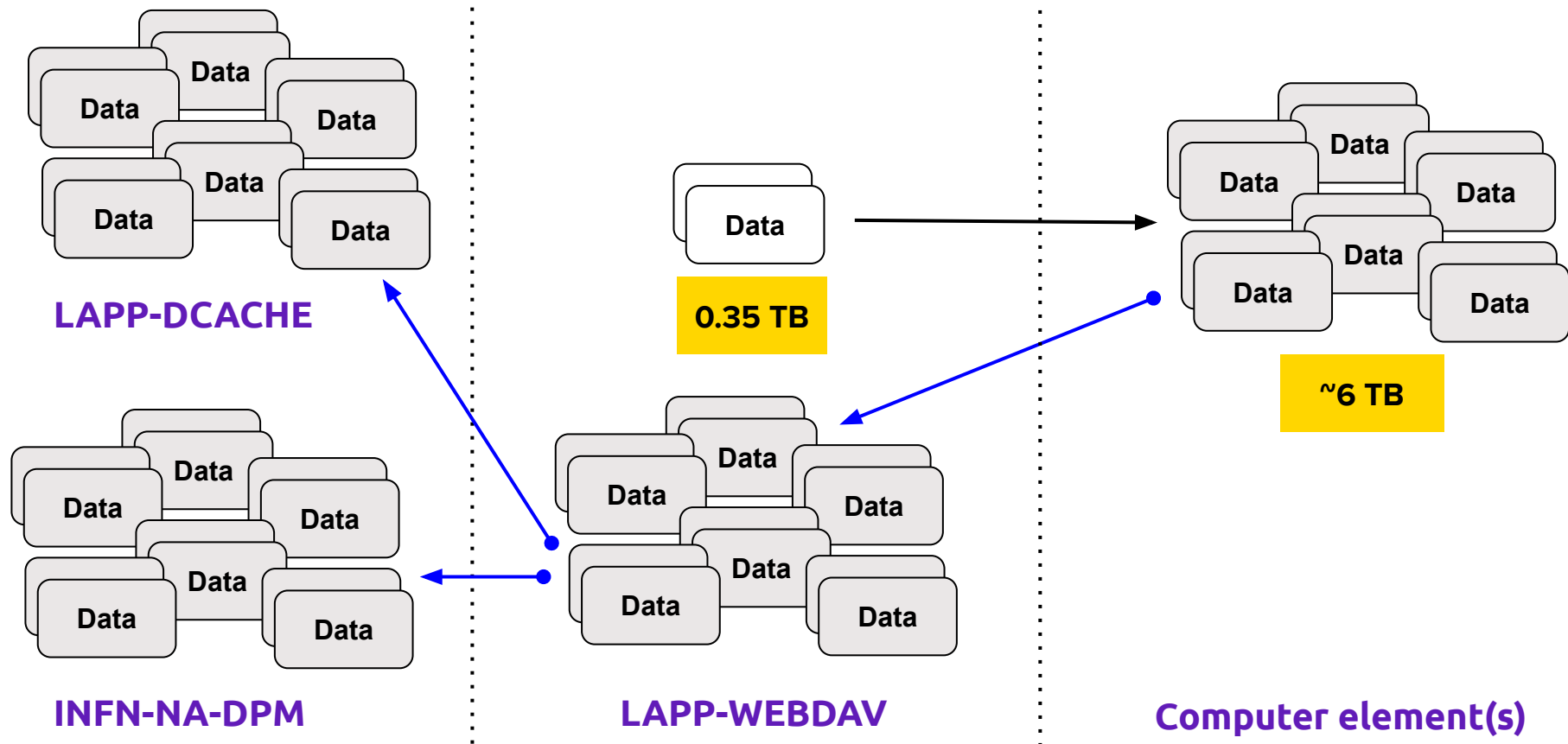
- We can also replicate in multiples RSEs as part of the challenge's tests
- Use a proper lifetime so to clean after the tests (to be tested)



# Artificial multiplication of the data



# Artificial multiplication of the data



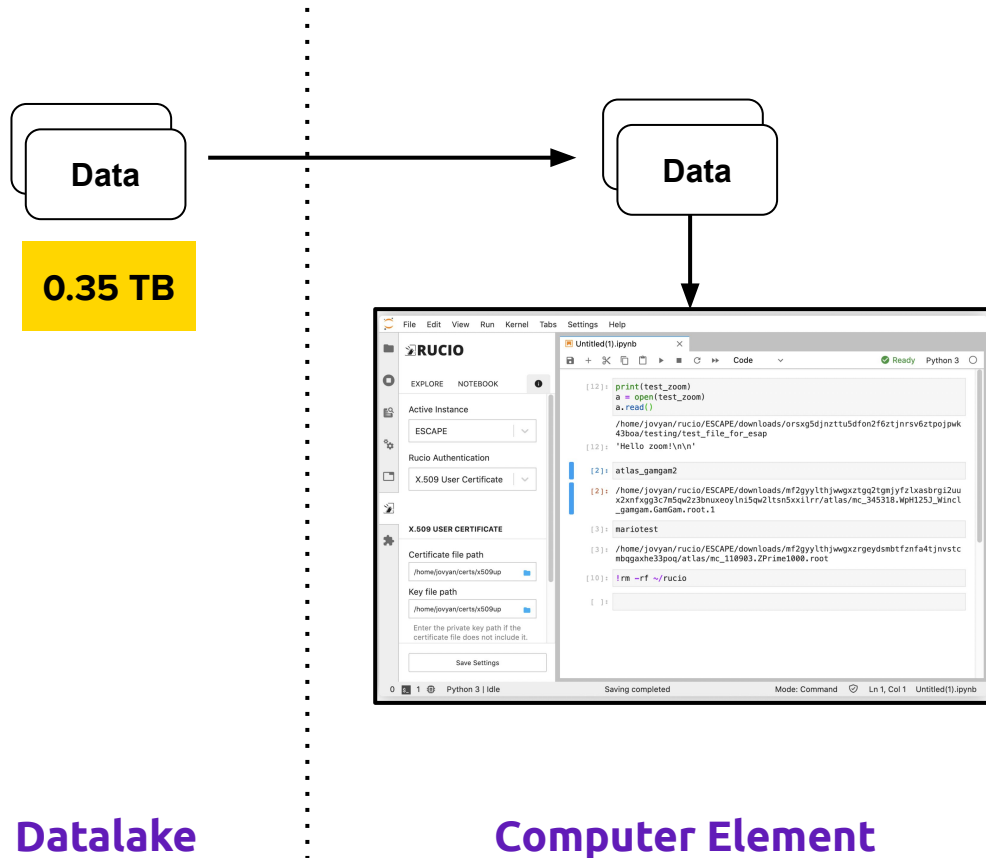
# Data Analysis

# Analysis examples

We will use the current ATLAS Open Data analysis examples to retrieve and use datasets from the Datalake

- Analysis can be notebooks or analysis frameworks
- They can take from a few minutes (e.g. 5-30 min)
- To several (e.g. 4) hours

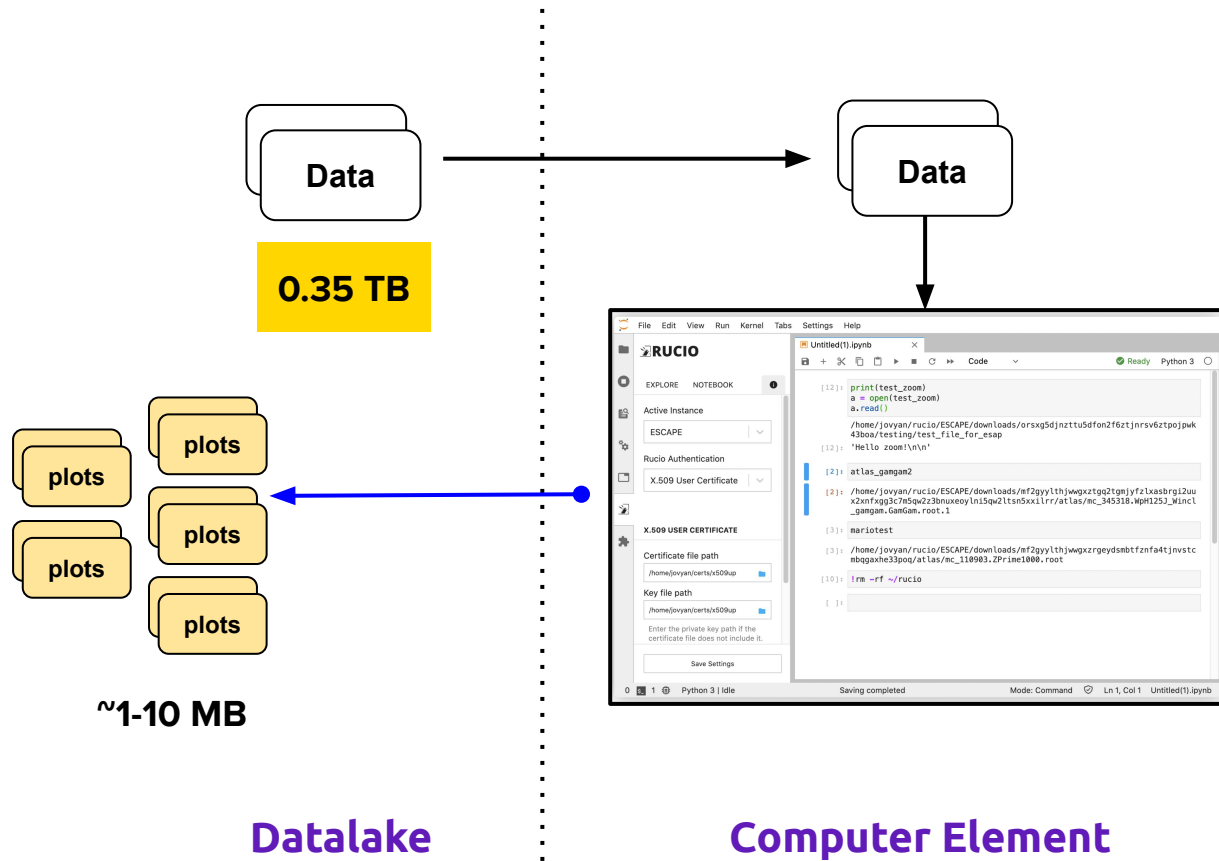
Also, write back the outputs to the Datalake



# Analysis examples

The outputs of the analysis can be upload to the Datalake

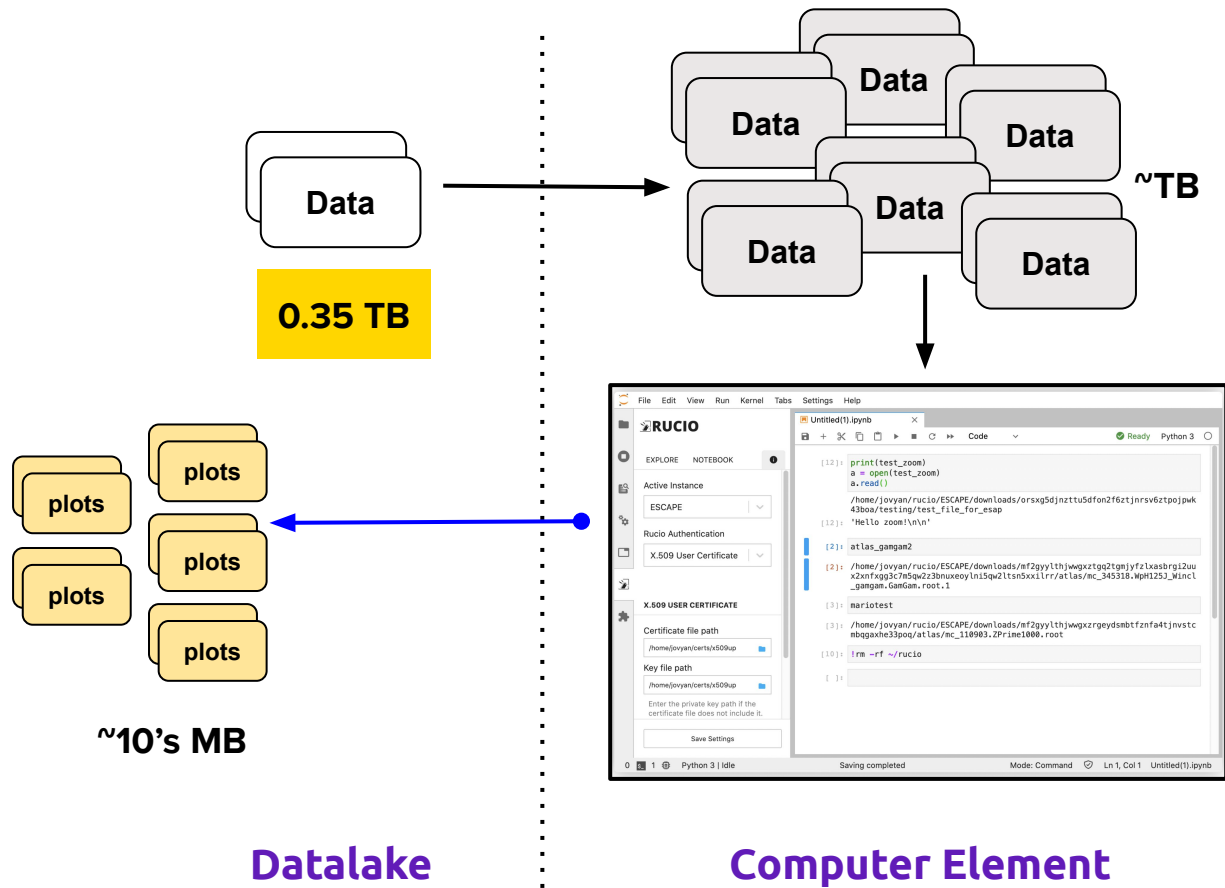
- The outputs are small; they are plots that can also be store in ROOT files
- No intention to upload single PNG files to The Datalake



# Analysis examples

We can also run the analysis examples over the “multiplied” data

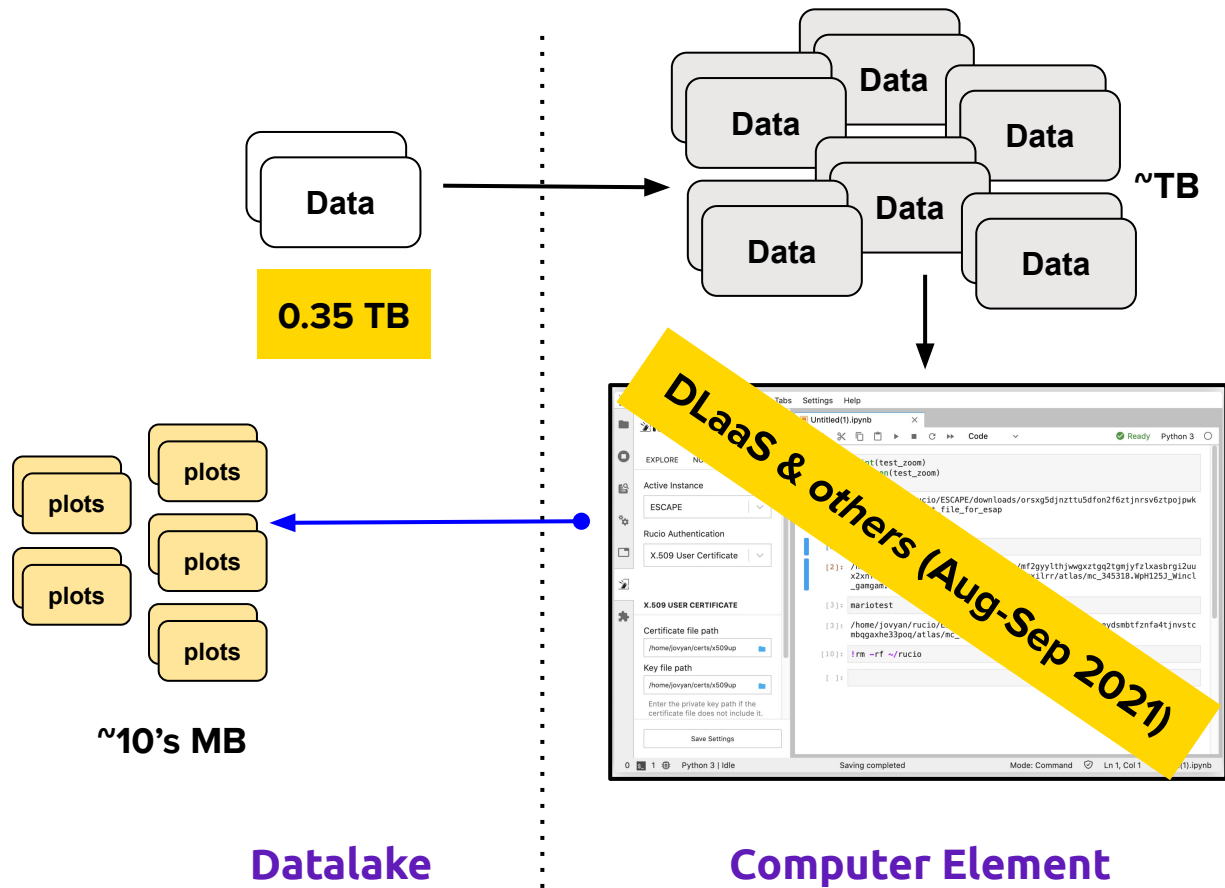
- This can help to simulate longer analysis that can last several hours (e.g. ~8-12 hours)
- In case this kind of “stress” is useful in this challenge



# Analysis examples

We can also run the analysis examples over the “multiplied” data

- This can help to simulate longer analysis that can last several hours (e.g. ~8-12 hours)
- In case this kind of “stress” is useful in this challenge

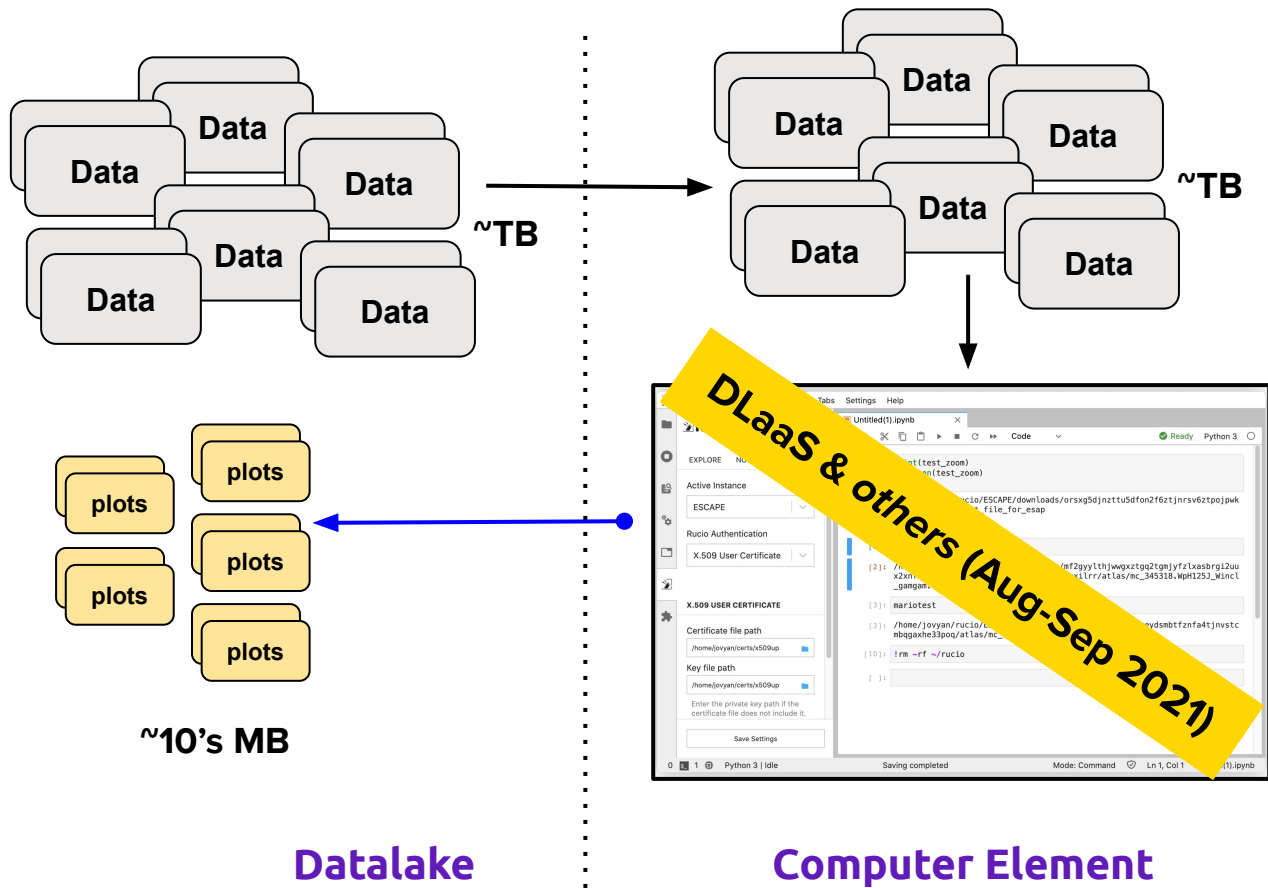




# Analysis examples

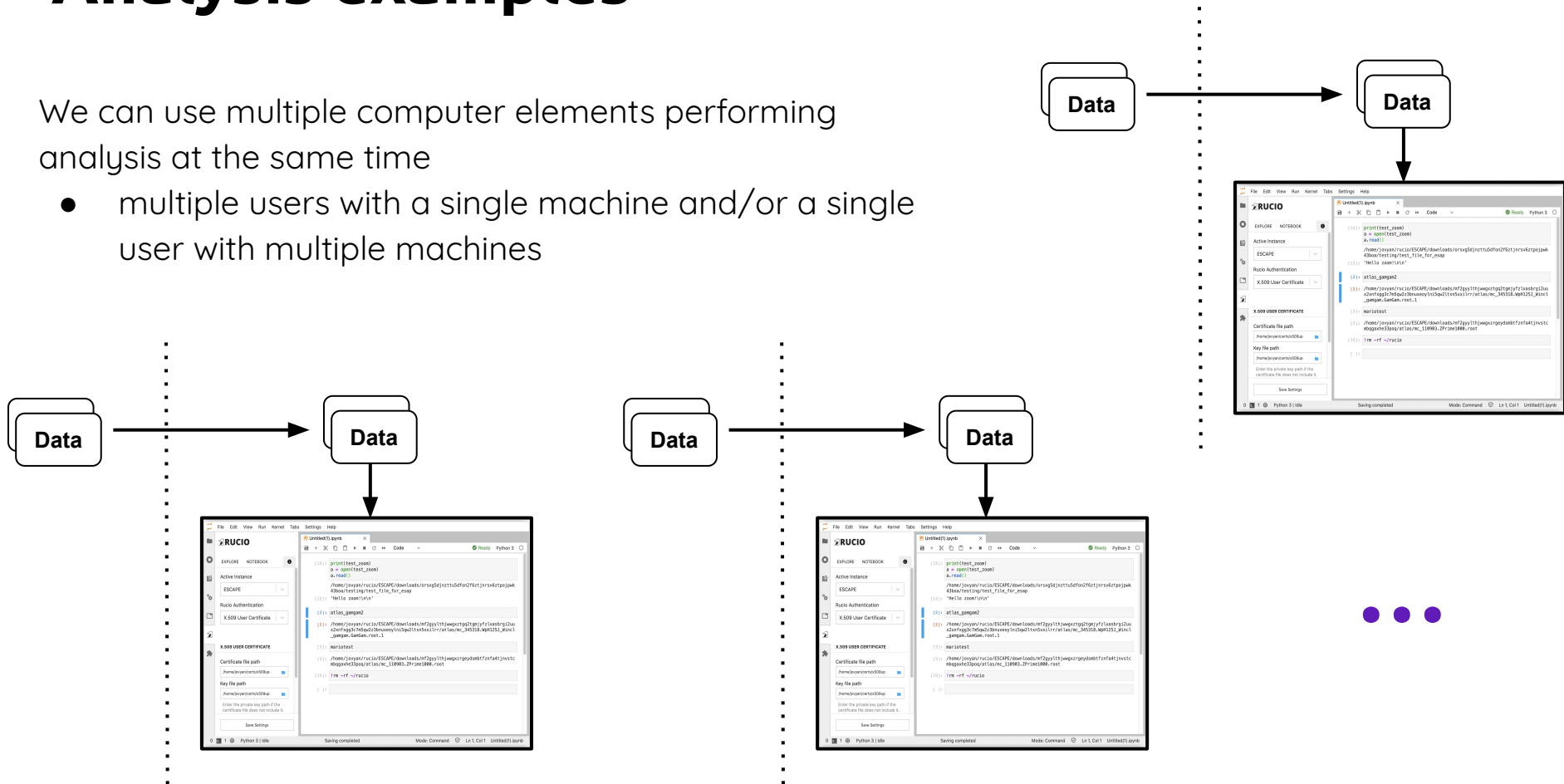
We can also run the analysis examples over the “multiplied” data

- This can help to simulate longer analysis that can last several hours (e.g. ~8-12 hours)
- In case this kind of “stress” is useful in this challenge



We can use multiple computer elements performing analysis at the same time

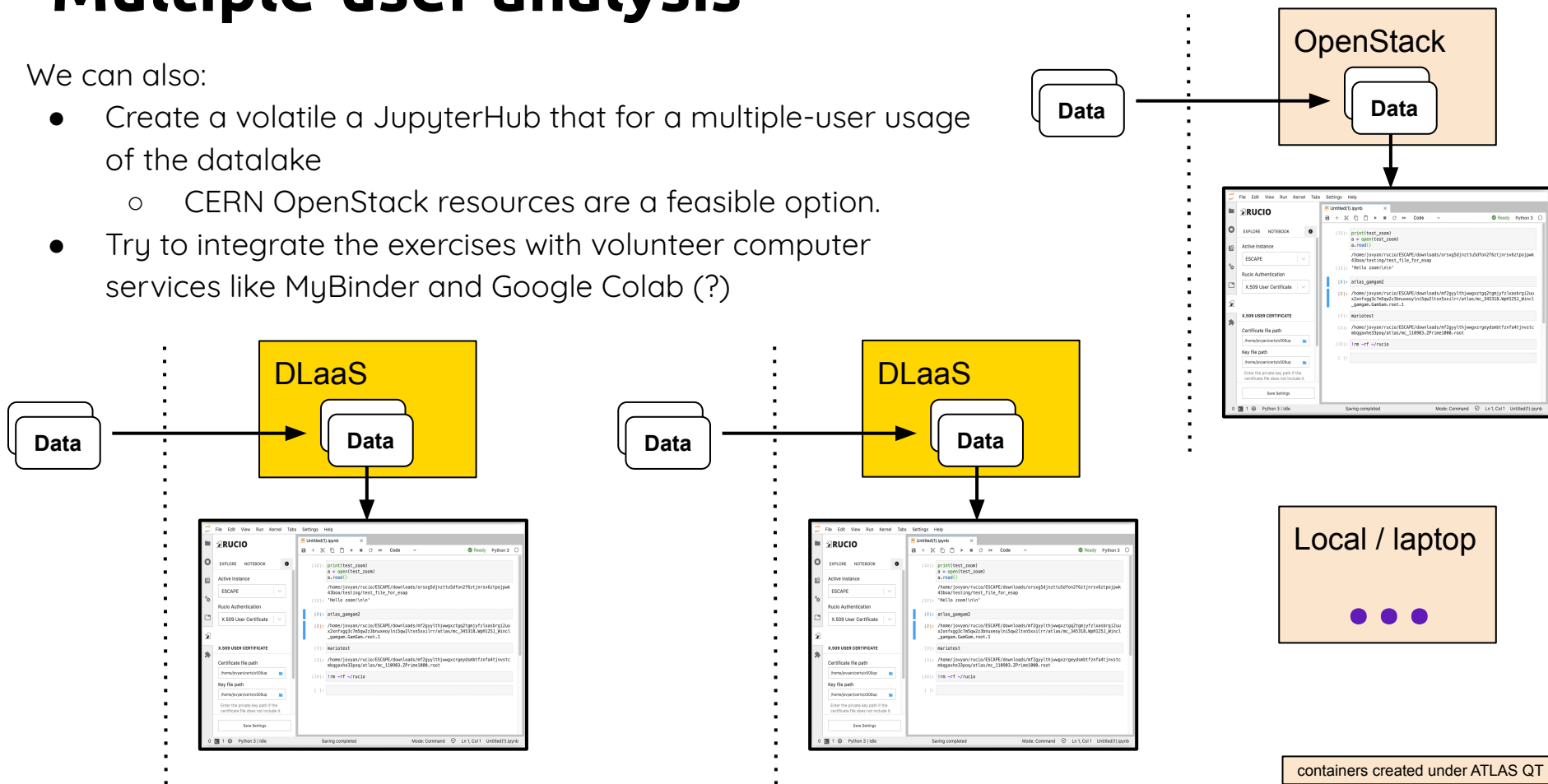
- multiple users with a single machine and/or a single user with multiple machines



# Multiple-user analysis

We can also:

- Create a volatile a JupyterHub that for a multiple-user usage of the datalake
  - CERN OpenStack resources are a feasible option.
- Try to integrate the exercises with volunteer computer services like MyBinder and Google Colab (?)



File Edit View Run Kernel Tabs Settings Help

# RUCIO

EXPLORE NOTEBOOK ⓘ

Active Instance

ESCAPE ▾

Rucio Authentication

X.509 User Certificate ▾

## X.509 USER CERTIFICATE

Certificate file path

/opt/rucio/etc/client.crt

Key file path

/opt/rucio/etc/client.key

Enter the private key path if the certificate file does not include it. Passphrase-protected certificate is not supported.

Account

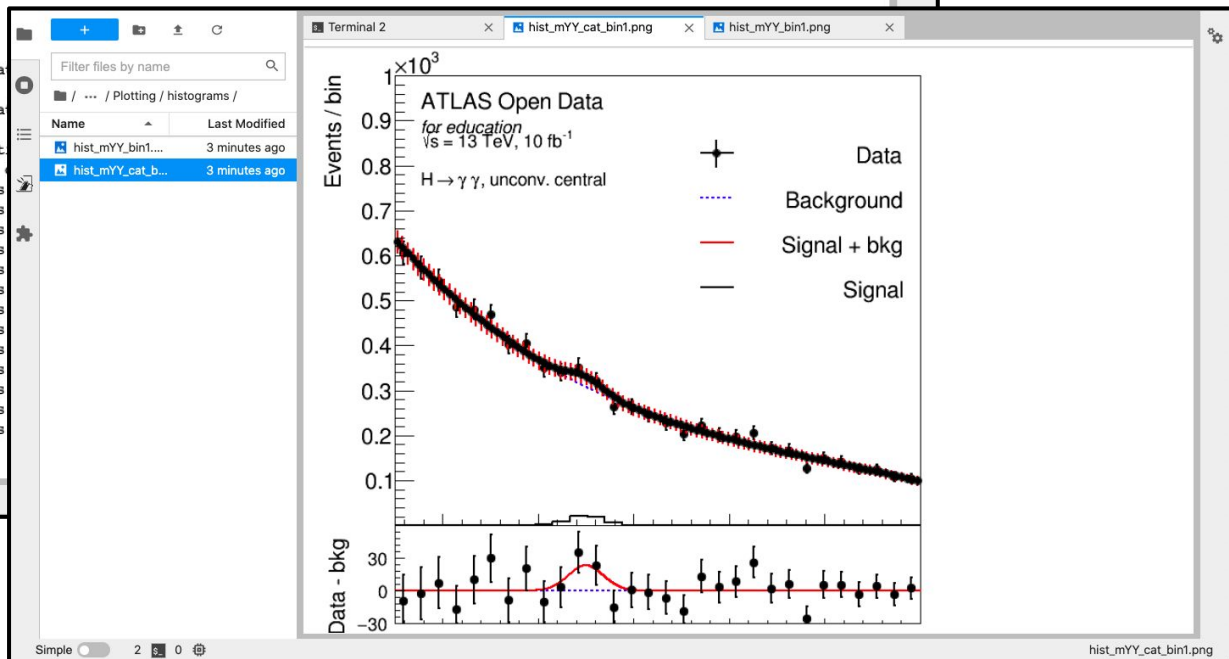
arturos

Show Advanced Settings

Simple 2 0

Terminal 2

```
HyyAnalysis.h Output_HyyAnalysis      main_HyyAnalysis_web.C run_web.sh
sh-4.2$ vim main_HyyAnalysis.C
sh-4.2$ ./run.sh
Which option should I run?
Options are:
0 = run all data and MC one after another
1 = run data only (can be run in parallel)
2 = run MC samples only (can be run in parallel)
0
Option is 0
Should I use PROOF? (will make things faster)
Options are:
0 = NO
1 = YES
0
PROOF option is 0
starting ROOT
Info in <TUnixSystem::ACLi>: crea
yAnalysis_C.so
Info in <TUnixSystem::ACLi>: crea
ysis_C.so
Starting analysis with process opt
Analysed a total of: 50000 events
Analysed a total of: 100000 events
Analysed a total of: 150000 events
Analysed a total of: 200000 events
Analysed a total of: 250000 events
Analysed a total of: 300000 events
Analysed a total of: 350000 events
Analysed a total of: 400000 events
Analysed a total of: 450000 events
Analysed a total of: 500000 events
Analysed a total of: 550000 events
Analysed a total of: 600000 events
Analysed a total of: 650000 events
Analysed a total of: 700000 events
```

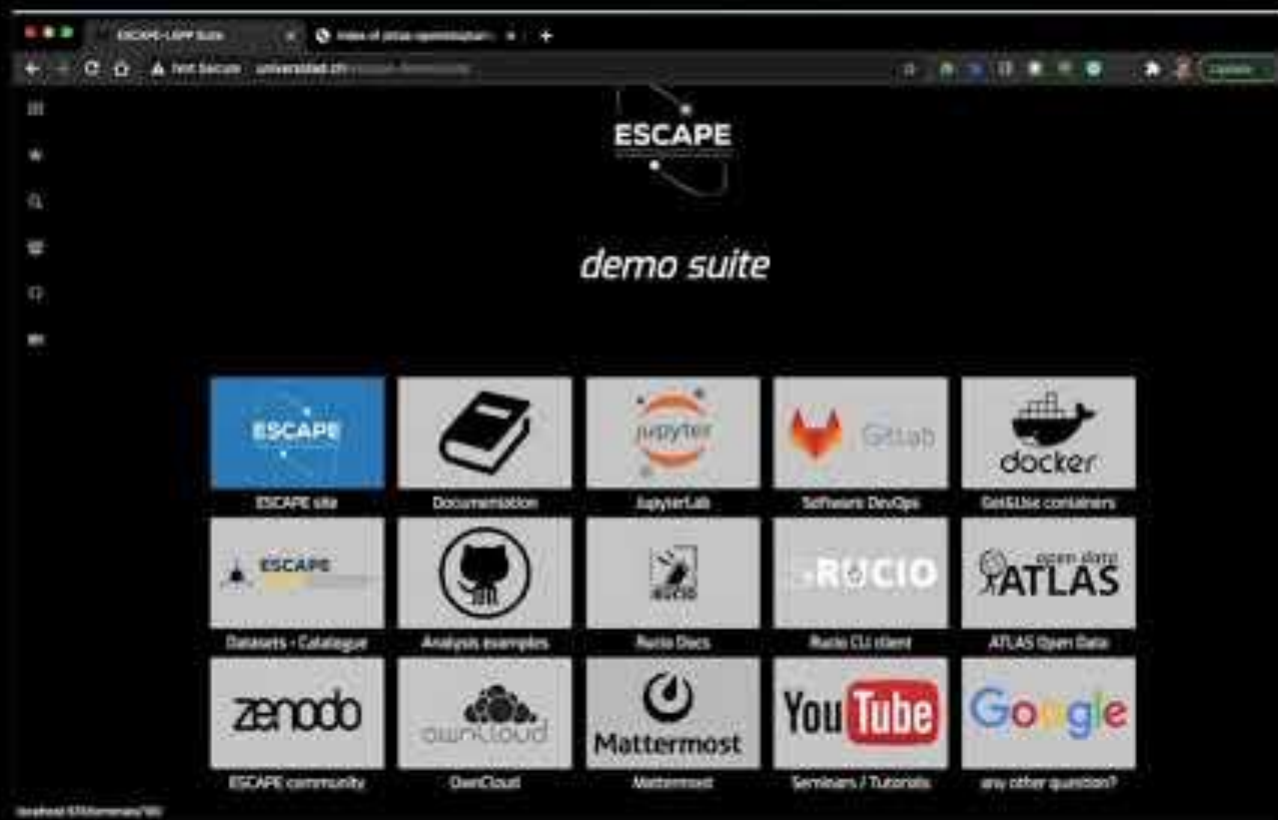




# Backup

# The RUCIO CLI client

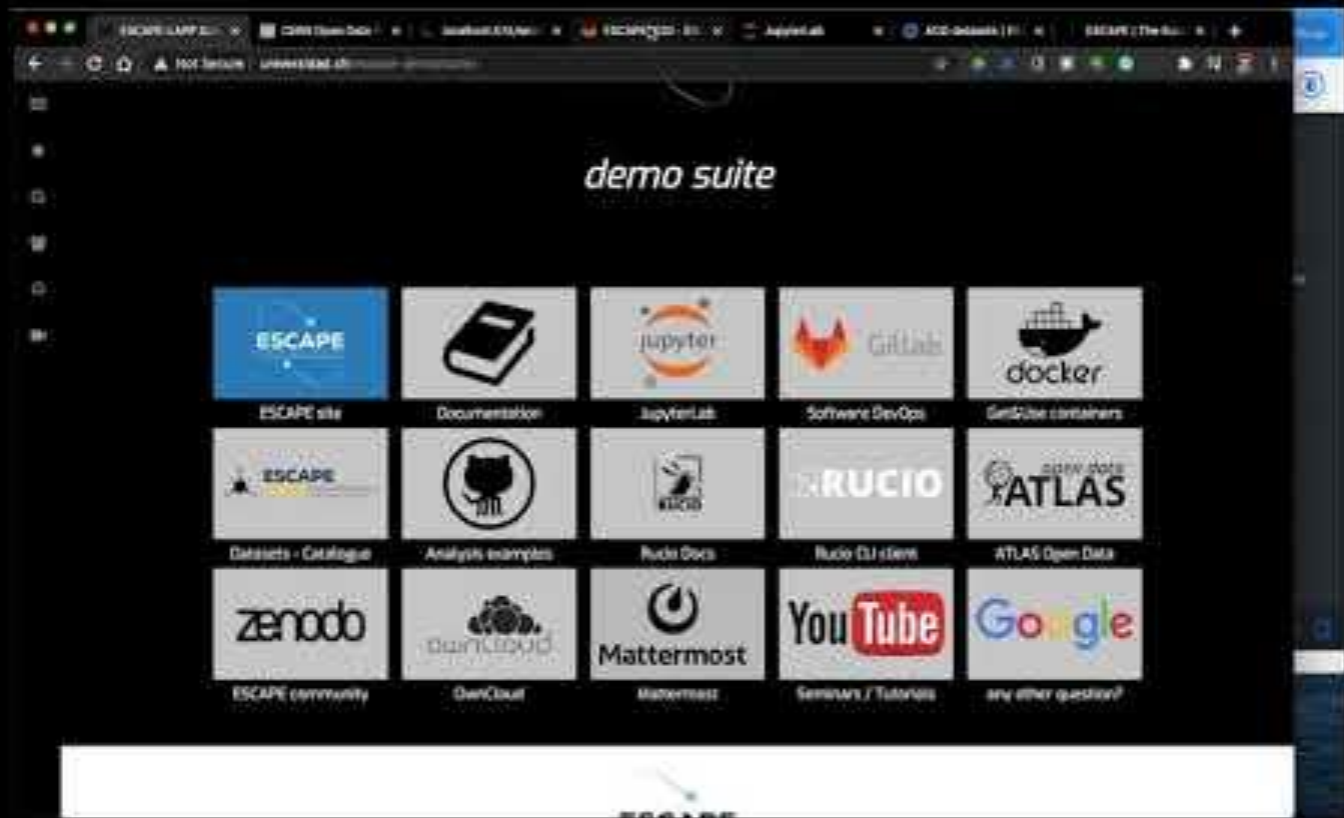
(a 90 sec video, mainly  
for new users)



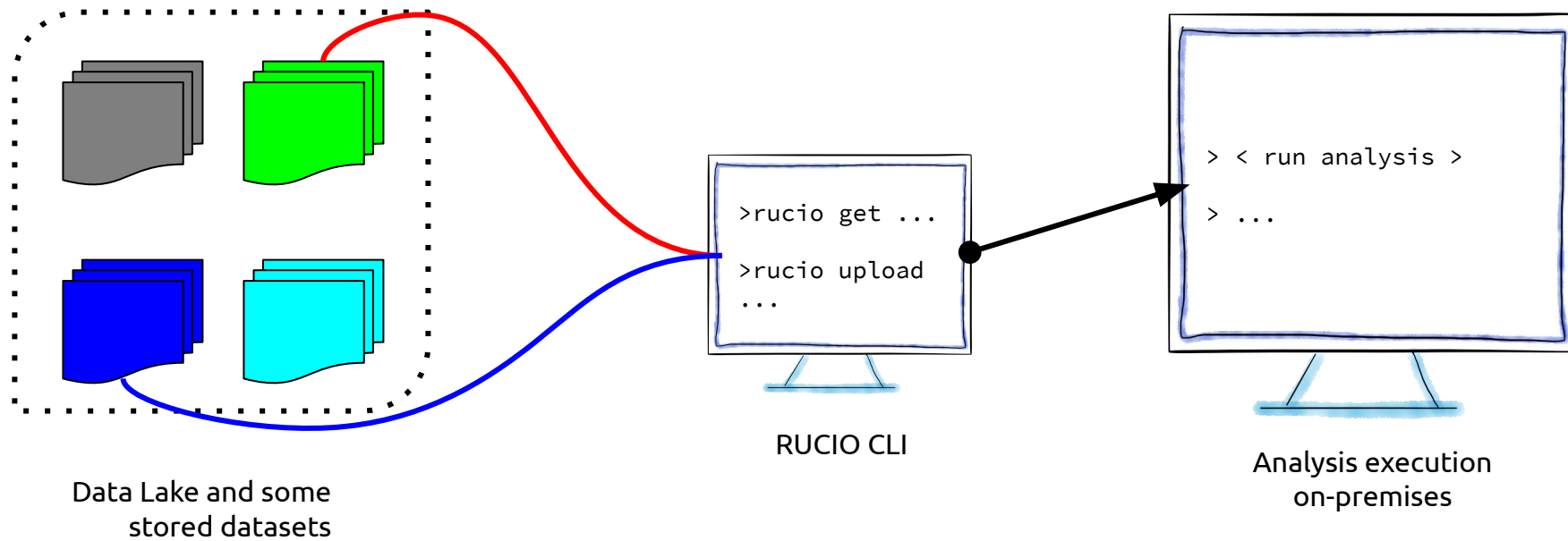
# Ongoing developments with **JupyterLab & RUCIO** **extension**

(a 150 sec video)

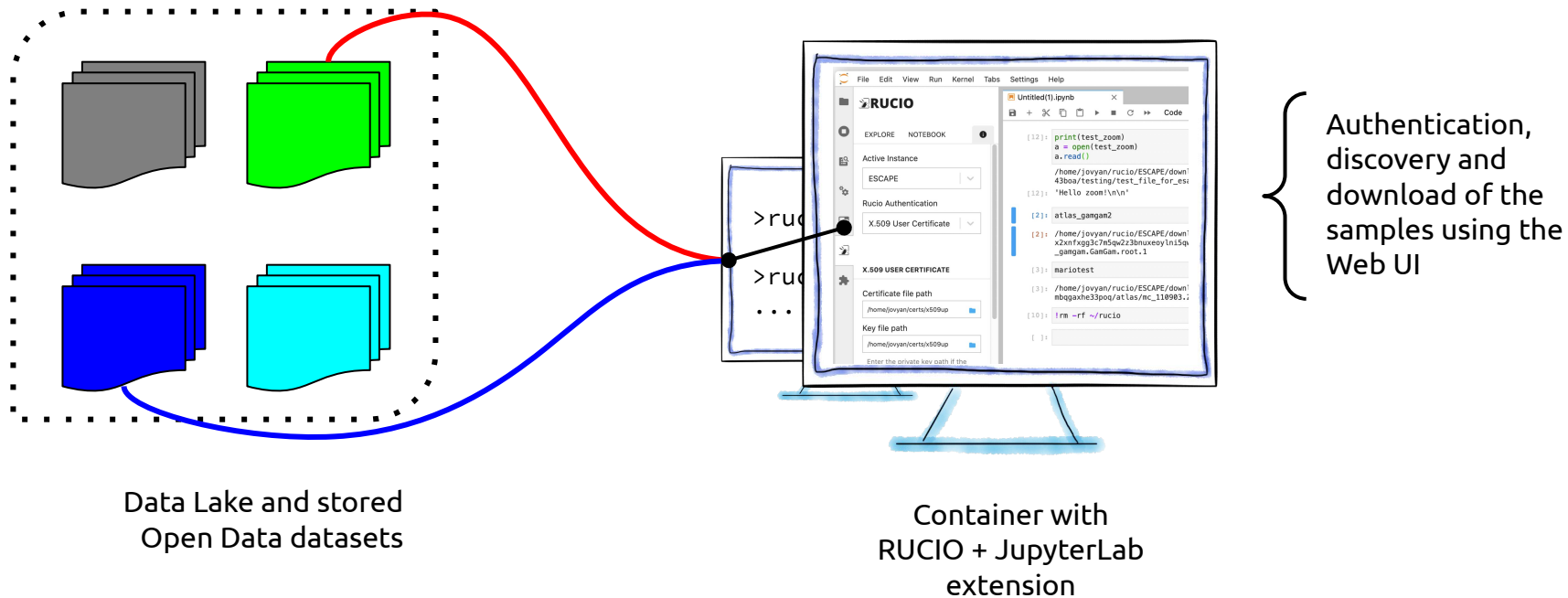
More tools to finish to  
integrate in the container, like  
more kernels, PROOF, CVMFS





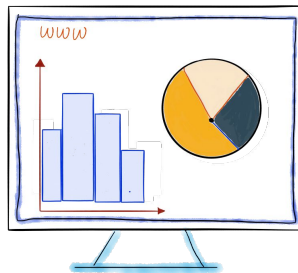


**CLI interaction with samples**



**RUCIO+JupyterLab (container) interaction for users**

# ESCAPE: update on RUCIO + JupyterLab + ATLAS Open Data integration



**Arturo Sánchez Pineda**

7-10<sup>th</sup> May 2021, LAPP

# Overview

## A series of exercises to perform during the ESCAPE DAC21 in November

<https://docs.google.com/document/d/1mHZiVA7S2mgRKyMVUb39fwT9OUUfKZjHAlstWc6hrs4/>

- Data “multiplication” where multiple version of the same data is generated, simulating a data-augmentation process
- Writing of such “multiplied” data back to the Datalake
- Exercises include the analysis of data stored in the Datalake
- Create clear instructions for other users that can be part of the challenge

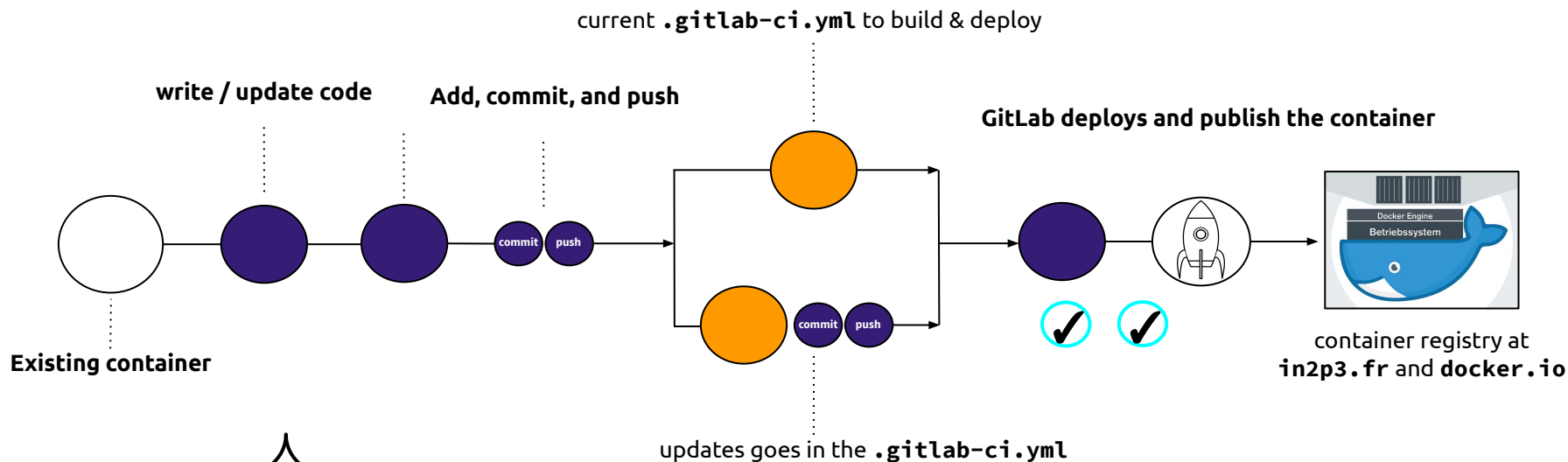
## The Jupyter client and RUCIO + ATLAS Open Data as a demo construction for final users

<https://atlas-lapp-outreach-education.pages.in2p3.fr/escape-demo-suite-site/>

- Activities relative to a integration and consolidation of the **Data Lake usage** via RUCIO CLI client and a friendly UI, JupyterLab + rucio-extension
- Efforts to consolidate those in a single collection of containers
- And how ATLAS Open Data is used as a analysis test pool for such integration for “normal” users
- Explore how this can be used no only for *single-user-laptop* or *single-user-vm* but integrated as a modular way to deliver containers in a multi-user platform like JupyterHub

# Containers review

# A view to the current container



## Container CI / CD

- The series of resources is package in a single container
- The CI setup automatically handles the publication of the container

Frédéric &  
Berkay's job

## Several developments and deployment already in place

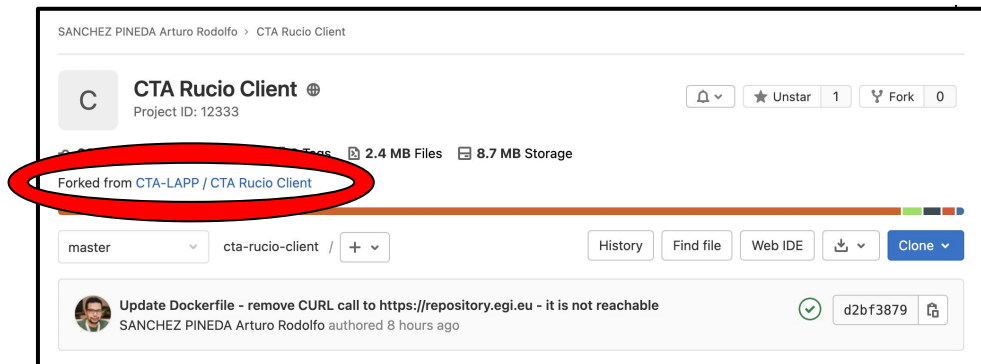
- The compendium of resources includes the current rucio client + JupyterLab + RUCIO extension, proxy & authentication, ...

# A view to the current container

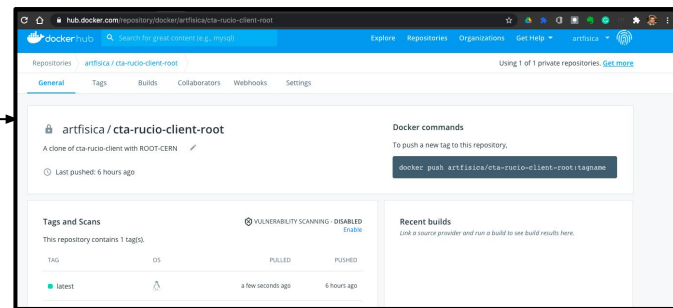
current `.gitlab-ci.yml` to build & deploy

write / update code

Add, commit, and push



GitLab deploys and publish the container



updates goes in the `.gitlab-ci.yml`

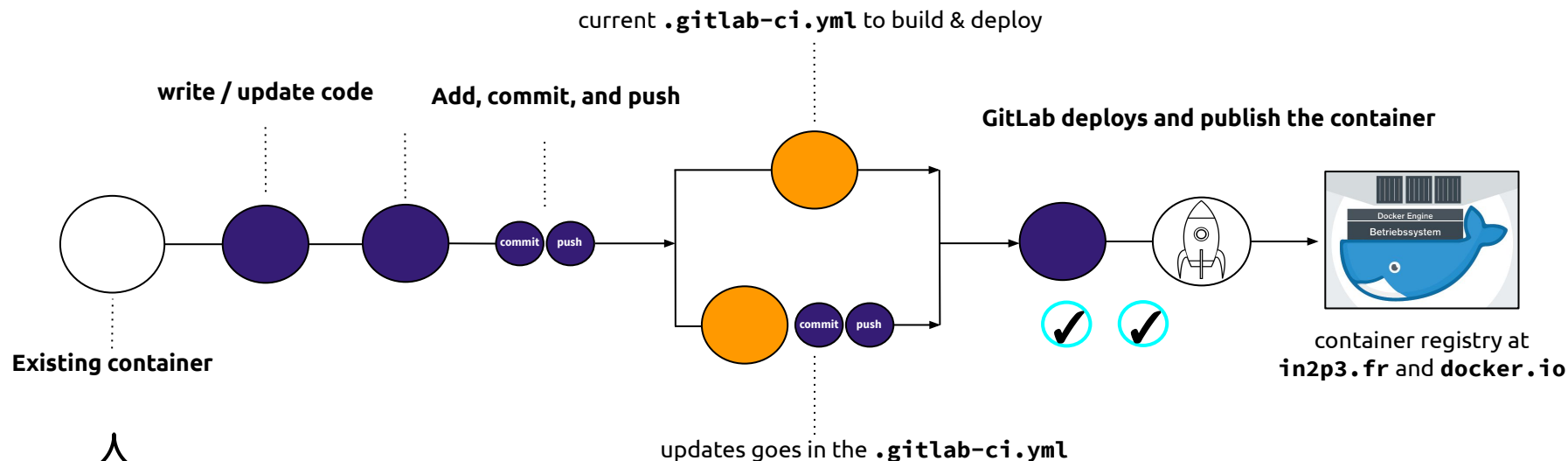
## Container CI / CD

- The series of resources is package in a single container
- The CI setup automatically handles the publication of the container

## Several tools and updates added

- Mainly ROOT + some dependencies and extra tools...
- Jupyter conf file to handle the usage of the rucio extension (Muhammad feedback, see later)
- From JupyterLab-3 the widgets are installed using ipywidgets instead of labextension

# A reminder



## Container CI / CD

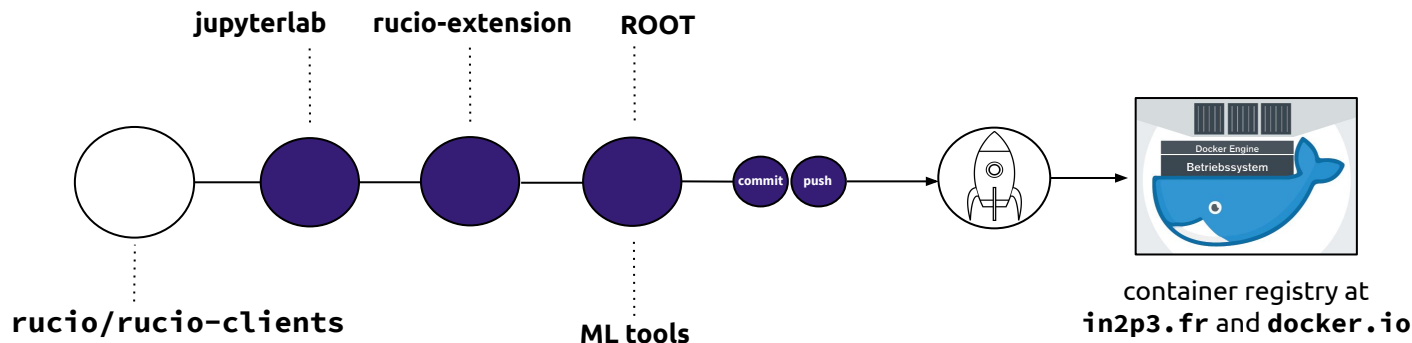
- The series of resources is package in a single container
- The CI setup automatically handles the publication of the container

## Several developments and deployment already in place

- The compendium of resources includes the current rucio client + JupyterLab + RUCIO extension, proxy & authentication, ...



# BASE IMAGE = rucio/rucio-clients

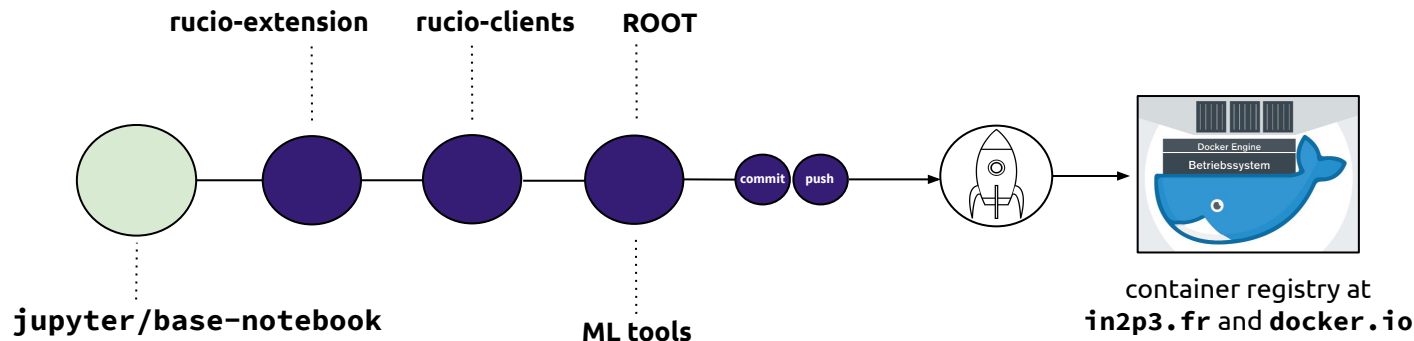


## The containers start with a base image

- The one I have been using from CTA uses the rucio-clients as base image
- From there, I add extra HEP-related tools

- We are exploring different base container and software structure to create those to be used by the users with JupyterLab + rucio + other tools

# BASE IMAGE = jupyter/base-notebook

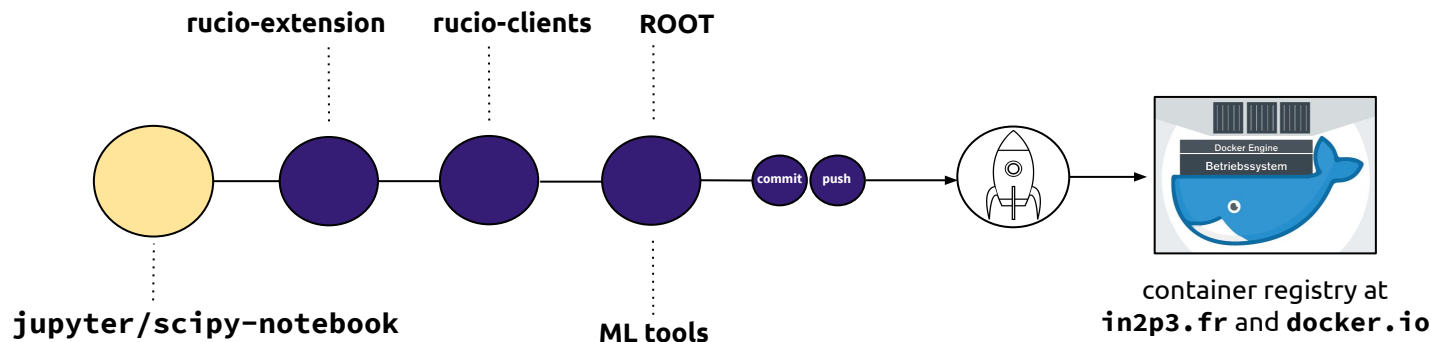


## The containers start with a base image

- In ATLAS Open Data we have been working also in containers that use jupyter as the base
- From there, I add extra HEP-related tools

- This activity is done in synchrony with ongoing developments in the ATLAS Open Data project to deliver containers for training in HEP

# BASE IMAGE = jupyter/scipy-notebook

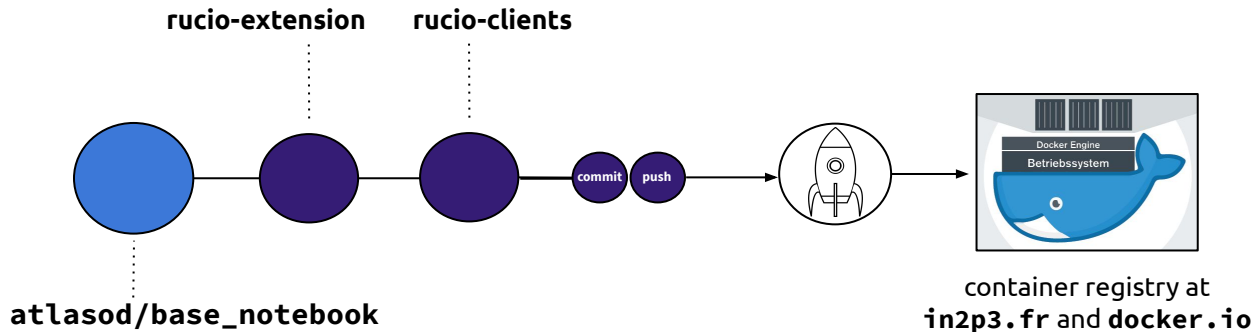


## The containers start with a base image

- Jupyter has a family of “popular” containers that can be used to further decrease customizations from our side, and building times
- The idea is to use the flexibility of using different bases

- Allow us to use a combination of containers using [JupyterHub](#) [Docker Spawner](#) to deploy multiple-user JupyterHub, e.g. in already existing infrastructure

# BASE IMAGE = atlasod/base\_notebook



## The containers start with a base image


- In this approach, we can imaging adding the rucio functionalities to already established “experiment containers”
  - In this examples, ATLAS Open Data has a container that is enhanced with ESCAPE-rucio tools.
- This may reduce significantly the maintenance time and rely on well-supported tools (OS, Jupyter, python, LCG, RUCIO, ROOT) as much as possible. Also, allow for an easier way to keep the container up to date

# A containers collection

Such tests ancontaines are growing in  
<https://gitlab.in2p3.fr/container-collection>


A good place to deploy in a “controlled environment” is the coming ESCAPE school. No access to the Datalake but to the tools

E


**ESCAPE-container-collection**   
Group ID: 9580

**Subgroups and projects** Shared projects Archived projects


---




A

**aod-jupyter-rucio**  Maintainer  
A fork of the CTA container using JupyterLab and rucio{client,extension}

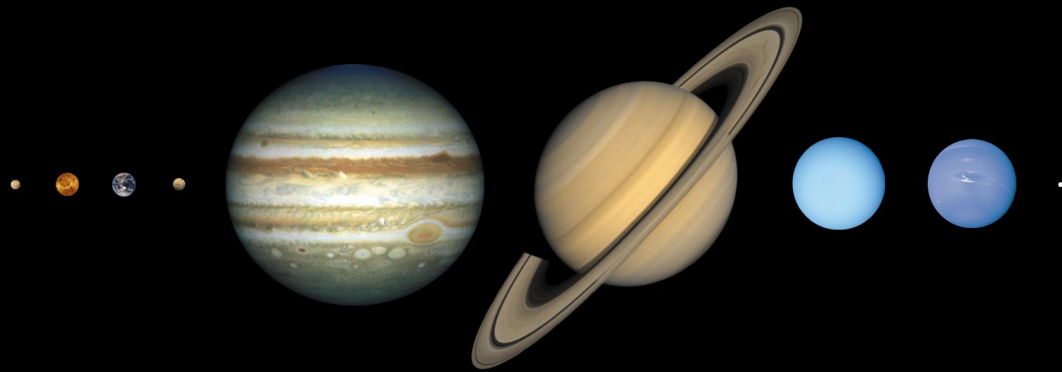
---



C

**CTA Rucio Client**  Maintainer  
A fork of the CTA container using JupyterLab and rucio{client,extension}

# A supported container *system*



A collection of supported containers will allow maximising the reach of the target audiences while keeping a realistic objective in term of human capital for the creation, maintenance and user's support

- **Mercury** → it is the collection's base container. It has a minimal setup, including JupyterLab and rucio.
- **Venus** → a "hotter" version of Mercury, with standard DevOps software tools.
- **Earth** → The most popular container. Including a series of common HEP tools that *most* users have requested.
- **Mars** → A dedicated HEP container, slimming version of the Earth.
- **Jupyter** → the largest container. It has all the tools. For who want to have it "all".
- **Saturn** → Some experiment's custom version. Same with **Uranus**, **Neptune**?