The Rucio Data Management system

BNL seminar

Cédric Serfon (BNL)

RUCIO
SCIENTIFIC DATA MANAGEMENT

BROOKHAVEN
NATIONAL LABORATORY

Content

- Introduction
- The basic Rucio concepts
- The Rucio tools
- Experience from ATLAS migration
- Future
- Conclusion

Introduction

What is Rucio ?

Don Quixote de la Mancha

Rucio

- Rucio is an advanced Distributed Data Management system (DDM) originally developed by ATLAS
- It replaced the previous system ATLAS DDM system called DQ2 (Don Quixote 2)
- This is an open source project that is used or is being evaluated by many other collaborations

A bit of history (1)

- ATLAS previous DDM system called DQ2 has been developed starting 2006
- DQ2 was used successfully during LHC Run1, but :
  - During that time scalability issues were identified
  - DQ2 used an external replica catalog (LFC), and it leads sometimes to inconsistencies
  - Difficult to add new features and new technologies
  - Limited functionalities to manage data placement policies
- For all these reasons, it was decided at the end of Run1 to create a new project from scratch. The new project would benefit of all the experience gained over the previous year with DQ2
- The system should be able to scale for LHC Run3 and Run4 (with data volumes >1 EB and billions of files

A bit of history (2)

- 2011-2012 :
  - First discussions on a successor to DQ2
  - Technical proposal
- 2012-2014 :
  - Development, testing, commissioning
  - Migration from DQ2 to Rucio (in parallel from migration prodsys → prodsys2, FTS2 → FTS3)
- 2015 :
  - Optimization for run2
- 2016-now :
  - New features (zip, Lifetime Model, rebalancing, temporary unavailable replicas, etc.)
  - Annual Rucio community workshop since 2018
  - Coding camps since 2018
  - More collaborations adopt Rucio

A growing community

Rucio main functionalities

- Provides many features (can be enabled selectively)
  - File and dataset catalog (logical definition and replicas)
  - Transfers between sites and staging capabilities
  - Web Interface and Command Line Interface to discover/download/upload/transfer data
  - Extensive monitoring
  - Powerful policy engines (rules and subscriptions)
  - Bad file identification and recovery
  - Dataset popularity based replication
  - …

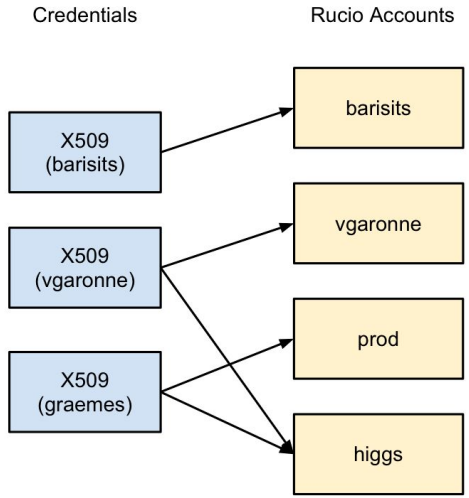
More advanced features

- Rucio can be integrated with Workload and Workflow Management System
  - Already supporting PanDA (ATLAS WFMS)
  - Integration with Dirac ongoing

The main Rucio concepts

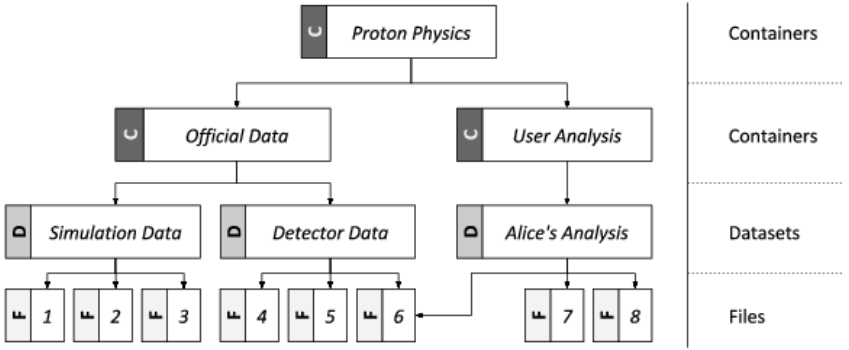
Rucio Account

- To use Rucio you need to be registered in the system via an account
- An account can represent:
  - single users (e.g. serfon)
  - groups (higgs)
  - activities (production_manager).
- Connection to a Rucio account possible via:
  - x509 certificate/proxy
  - Kerberos
  - Username/Password
- N to M mapping :
  - One credential can be used to map to different accounts.
  - Different credentials can map to one account
- Quota and permissions are tunable and applied per account.



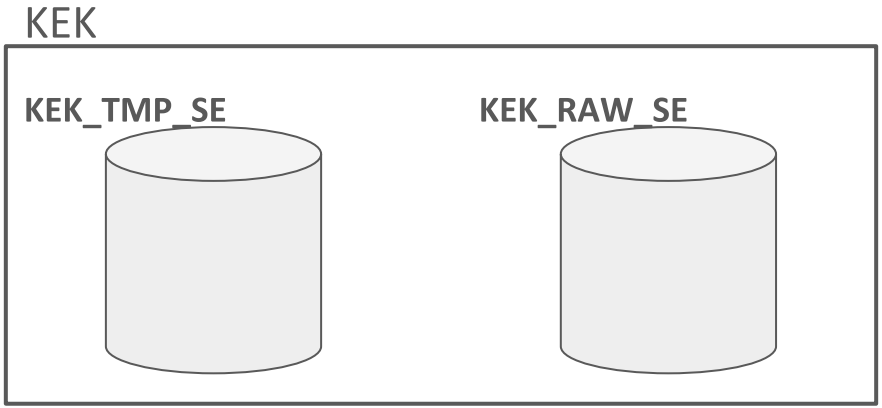
Data IDentifier

- To address data, Rucio uses Data Identifier (DID)
- A DID is a string composed of a scope and a name
  - Name and scope are divided by a colon
  - The name is only unique within its scope
  - E.g.: `user.serfon:mytest.root`
- Three **logical** units can be addressed by a DID
  - Files
  - Datasets : Collection of files
  - Containers : Collection of datasets and/or containers
- The DID is globally unique (over all three units)



Rucio Storage Elements

- To maintain storage, Rucio defines Rucio Storage Elements (RSE).
- A RSE is an abstraction layer for a storage end-point.
- RSEs are described by various attributes.
  - site=KEK
  - type=TMP_SE
  - country=jp
- RSEs attributes can be used to build RSE expressions : e.g.
  - site=KEK&type=RAW
  - (tier=0&tape=TAPE)|(tier=1&freespace>100)



Replication rules

- Rucio uses replication rules to specify how a DID will be distributed over RSEs using RSE expressions :
  - E.g., there need to be 1 replica of dataset x on RSEs with tier=1&type=DATADISK
- Rucio evaluates the rules and ensures :
  - Only the minimum number of replicas will be created
  - The number of transfers is minimized
- When a DID is changed (e.g. new files added/removed), Rucio will automatically ensure that the rule is still fulfilled (e.g. by transferring the new files/removing)
- Rules protects DID from being deleted from a location or a set of location
- Rules can have lifetime and will be removed automatically when the lifetime expires.

Subscriptions

- Subscriptions are replication policies based on DIDs metadata.
  - Matched against any DID that will be produced in the future.
  - The subscription will generate the replication rules according to the policy described in the subscription
  - They are evaluated as soon as the dataset appear
- Example :
  - Make 2 replicas of datasets with `scope=data15_13TeV` and `datatype=RAW` on `tier=1&type=DATADISK`

The Rucio tools

Basic functionalities

- Rucio provides catalogs to keep track of all the DIDs and their replicas
  - These catalogs can be accessed via different tools (REST API, python APIs and CLI, WebUI)
  - Rucio also stores a set of built-in metadata (e.g. GUID, metadata, #events, etc.). Generic metadata was added recently but is not used at large scale
- Rucio provides various daemons to evaluate the rules and subscriptions and to transfer, stage (when files are on TAPE) and delete data
- For the deletion there are 2 modes :
  - “greedy” mode (i.e. the data will disappear from disk as soon as the rule are gone)
  - Standard mode : Rucio decides when the data will be cleaned based on a water mark and on a LRU algorithm
  - Remember : Only file replicas w/o rules can be deleted

More functionalities

- In addition to these functionalities, Rucio has advanced ones :
  - Popularity based replication engine : All the accesses to DIDs are recorded by Rucio, and this can be used to make extra replication rules based in the popularity
  - Consistency checks and recovery : Tools to discover lost or corrupted replicas and to automatically fix them
  - Automatic rebalancing of data : If a site gets full, Rucio will automatically rebalance some data to sites with more space
  - New : Possibility to follow a dataset : To inform users or application in case a change is done to a DID (e.g. lost files, dataset is going to expires, etc.)
  - And many more (archive support, multihop)
- We regularly add new features, e.g. coming soon :
  - Multi-VO support
  - Support of other authentication services

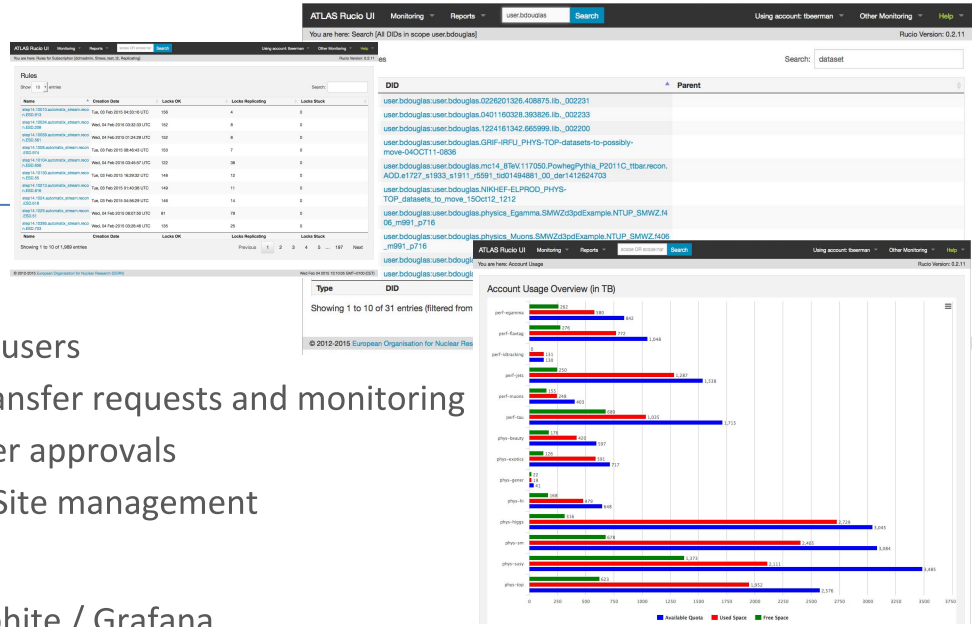
Quotas and permissions

- Quota per RSE and since the last release global quotas can be set for each user e.g. :
  - User jdoe has 10 TB quota on site A
  - User phys-higgs has 300 TB on tier=1&type=DATADISK
- Permission/policies highly tunable based on accounts, scopes, RSEs. Generic permission/policies files that can be overwritten to fit different collaborations needs, e.g. :
  - Account jdoe can only write in user.jdoe scope
  - All replication rules on type=SCRATCHDISK will expire after 2 weeks
  - Account with a country admin role us can delete the rules from any users on BNL-OSG2_LOCALGROUPDISK

Interfaces

- REST API (you can implement a client in whatever language you prefer), e.g. :
  - GET /accounts/ will get all the account in Rucio
  - Full documentation here : https://rucio.readthedocs.io/en/latest/rest.html
- Python API and CLI available, documentation here :
  - API : https://rucio.readthedocs.io/en/latest/api.html
  - CLI : https://rucio.readthedocs.io/en/latest/man/rucio.html
- Side note : Rucio (not only the API and CLI) is python 2.7 and 3.6 compatible. The 2.7 compatibility might be dropped in the future

Monitoring & analytics



- RucioUI
  - Provides several views for different types of users
  - Normal users: Data discovery and details, transfer requests and monitoring
  - Site admins: Quota management and transfer approvals
  - Central administration: Account / Identity / Site management
- Monitoring
  - Internal system health monitoring with Graphite / Grafana
  - Transfer / Deletion / … monitoring built on HDFS, ElasticSearch, and Spark
  - Messaging with STOMP
- Analytics and accounting
  - e.g., Show which the data is used, where and how space is used, …
  - Data reports for long-term views
  - Built on Hadoop and Spark

Migration experience from ATLAS

Migration experience from ATLAS

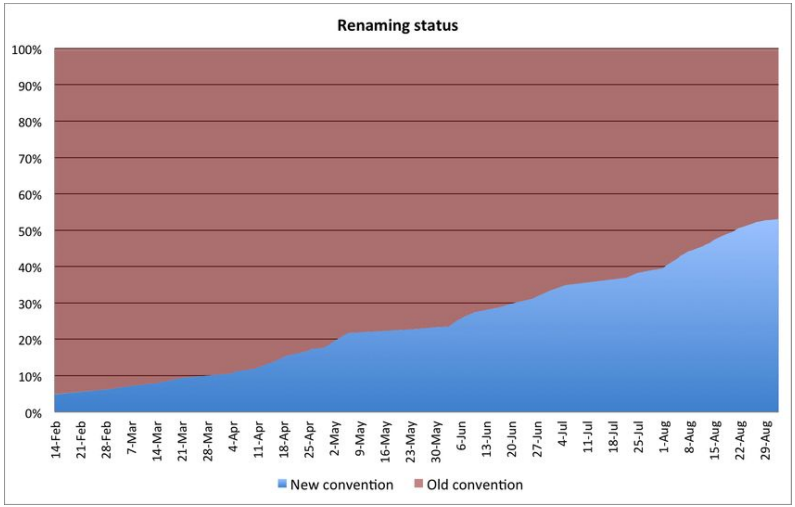
- The migration from the old DDM system to Rucio was a complicated process that took roughly 18 months.
- Many constraints :
  - Cannot have an extended downtime (not more than 1 day)
  - Need to help the users and applications to do the transition in a transparent way
  - Must be fully in operation more than 6 months before the start of LHC Run2
  - Some services which Rucio is interacting with changed at the same time (e.g. FTS)
- To address these constraints, a 3 steps scenario was designed and is detailed in the next slides

Migration step 1

- Rucio can use a “deterministic” location for the files (i.e. the location is determined exactly by the use of a function f(scope, name)), whereas in DQ2, it was not the case, e.g. :
  - DQ2 convention : data16_13TeV/AOD/f731_m1659/data16_13TeV.00306448.physics_Main.merge.AOD.f731_m1659/data16_13TeV.00306448.physics_Main.merge.AOD.f731_m1659._lb0346._0005.1
  - Rucio convention : data16_13TeV/18/9d/data16_13TeV.00306448.physics_Main.merge.AOD.f731_m1659._lb0346._0005.1
- The migration to Rucio needed to change the physical location of all the files on DISK to follow this new convention
  - The location needs also to be updated in the catalog used by DQ2 so that all the applications can still access the files at their new location

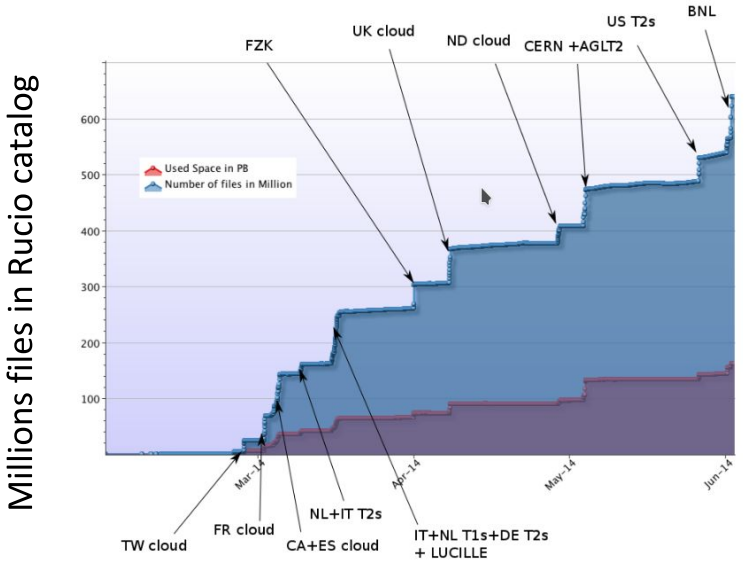
Migration step 1

- ~300M files over O(100) sites to “move”
- Not a physical move, but metadata operation
- To perform this, a dedicated tool was developed to perform the action remotely w/o action from the sites :
  - It move the file on the SE (not physical move, but metadata operation (WebDAV MOVE))
  - Change at the same time the location of the file in LFC
- The whole operation took about one year and was done in a background mode w/o disruption



Migration step 2

- LFC to Rucio file catalog migration
- Rucio become the replica catalog :
  - Change applied to the DQ2 client to be able to handle this new file replica catalog
  - Migration done step by step (around 10 sites at a time) using PL/SQL procedure over 4 months
  - For each of these steps we had to switch the production/analysis on the sites to migrate for ~1 day (i.e. we lose ~10% of the capacity).
  - The migration itself just takes a couple of hours



Migration step 2

- At a given time, a site uses only one replica catalog (either Rucio or LFC)
  - Main reason of this approach is that it's difficult to keep the synchronization between 2 catalogs is difficult
  - Only DQ2 knows which catalog is used for the site, the users and applications don't need to care
- Need DQ2 client changes :
  - For write and delete operations : Need to select the proper catalog
  - For read operations (e.g. list all the replicas of a files) : Need to aggregate the content of the 2 catalogs
- Rollback still possible but more complicated than step 1

Migration step 3

- Migration of all the DQ2 functionalities (i.e. catalog, agents) to Rucio
- This step required :
  - To provide a DQ2 client able to talk to Rucio. Not easy cause Rucio and DQ2 concept were quite different. The client was supported for ~2 years to allow the applications/end-user to switch to Rucio native client.
  - A complete shutdown of all the computing activities for a full day
- The migration was carefully run in dry-run many times on DB snapshots before the D-day
- Once the computing activities are restarted, we fully use Rucio
- Rollback :
  - Possible but difficult as long as one doesn't use specific Rucio feature (e.g. rules with RSE expression)
  - Not possible when start using Rucio specific features

Migration experience from ATLAS

- Migration shouldn’t be underestimated for the collaborations already running in production :
  - Need to provide tools so that the application/end-users’ work is not disrupted (new client, close work with the workload management system expert)
  - Tests, tests, tests
  - Enough time/manpower should be dedicated to it (estimate for ATLAS migration ~0.5 FTE over 18 months)
- Things that I would do differently :
  - On the migration itself nothing
  - On some initial choices how we use Rucio :
    - Better deterministic convention : Create too many subdirectories
    - Better use of the scope : Now we have a few very big scopes and a lot of small ones. Need careful evaluation of the partitioning before

Future

Rucio and Kubernetes

- ATLAS currently runs Rucio on Virtual Machines. Will move in the coming months to containers using Kubernetes
- “Kubernetes (K8s) is an open-source system for automating deployment, scaling, and management of containerized applications.”
- Interesting features :
  - Automated rollouts and rollbacks
  - Self-healing
  - Can provide automatic scale-up/down

Integration with Workflow Management

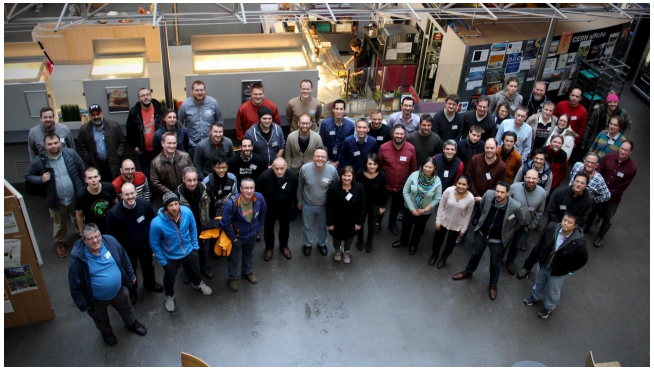
- Rucio is tightly coupled to panda (developed by BNL).
  - We should take this opportunity to provide a full DDM + Workflow Management System (WFMS) to communities that need a full bundle
- Many of the other communities evaluating Rucio use another WFMS called Dirac :
  - “Dirac INTERWARE is a software framework for distributed computing providing a complete solution to one or more user community requiring access to distributed resources.”. Which means it has both WMS and DDM functionalities
  - In the facts the DDM functionalities are less advanced than Rucio, which explains why these communities have a look at Rucio
  - Work ongoing to integrate Rucio and Dirac

Longer term development

- ATLAS plans to use Rucio for its run 3 and 4
- Up-to now, most of the new features requirements came from ATLAS, but one can expect that the number of requirements from other communities will grow in the near future
- From my point of view, the main things to address :
  - Improve the scalability from some components (e.g. rule evaluation) to jumbo datasets (>200k files)
  - Optimization of TAPE recall (in the context of data carrousel)
  - Automatize the user support (e.g. use of expert agents to answer/fix the problem of the end-users)

Rucio workshops and coding camps

- Rucio Community Workshops [2018] [2019]
  - Next at FNAL in March 2020 (more news soon)
- Rucio Coding Camps [2018] [2019]
- Development Meetings [Weekly]



Conclusion

- In a few years, Rucio has turned from the new kid on the block to a widely used DDM tool in the HEP/astro area
- It has a wide community behind it that goes beyond the initial ATLAS/CERN team
- It provides advanced DDM features for all type of experiments (from the small ones with O(1M) files/TB scale to the big ones O(1B) files/EB scale)
- If you want to know more, join the community workshop next year at Fermilab or have a look at the resources on the next slide
- I'm also happy to answer any questions anytime this week

Thanks for your attention

Website	http://rucio.cern.ch
Documentation	https://rucio.readthedocs.io
Repository	https://github.com/rucio/
Images	https://hub.docker.com/r/rucio/
Online support	https://rucio.slack.com/messages/#support/
Developer contact	rucio-dev@cern.ch
Journal article	https://doi.org/10.1007/s41781-019-0026-3
Twitter	https://twitter.com/RucioData