CMS Rucio Community Report

Rahul Chauhan (CERN) for the CMS Rucio Team

7th Rucio Community Workshop, San Diego Supercomputer Center, US

30 September 2024 to 04 October 2024

CMS and Rucio

Running Rucio for Over 5 Years

- Transition from PhEDEx to Rucio started in Fall 2018.

CMS Data Model (Files, Blocks, Datasets) Aligns with Rucio's Model (Files, Datasets, Containers)

CMS blocks → Rucio datasets	*CMS datasets → Rucio containers*

- Rucio allows each of these to be many-to-many relationships
  - CMS maintains many-to-one relationships
  - Recently started utilising container-container many-to-many relations for partial pileup.

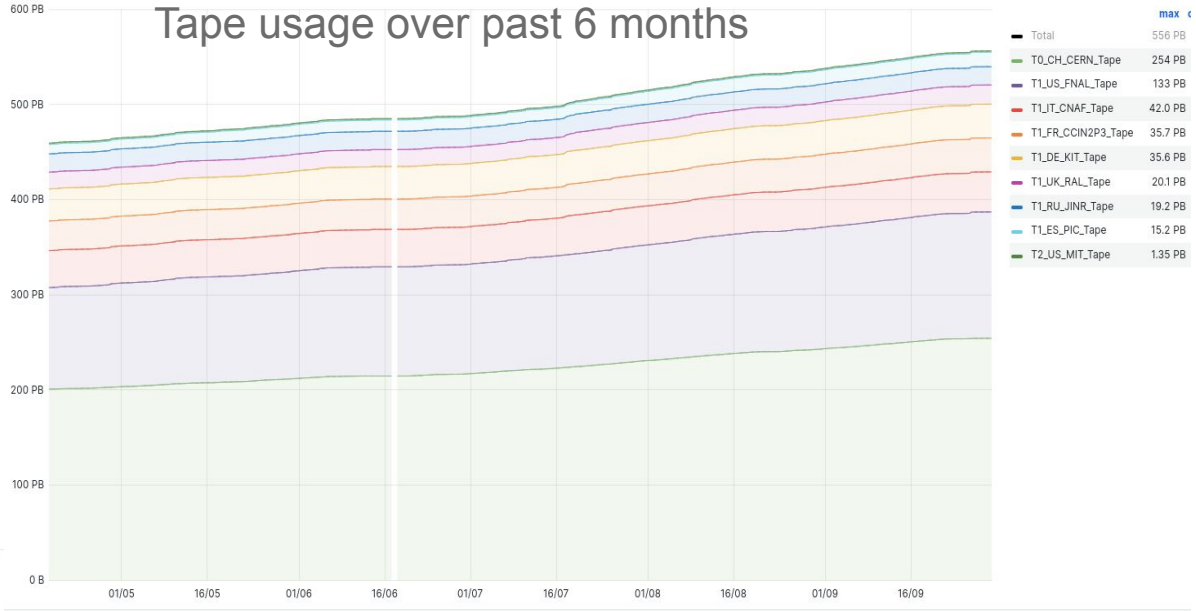
LFN to PFN Mapping

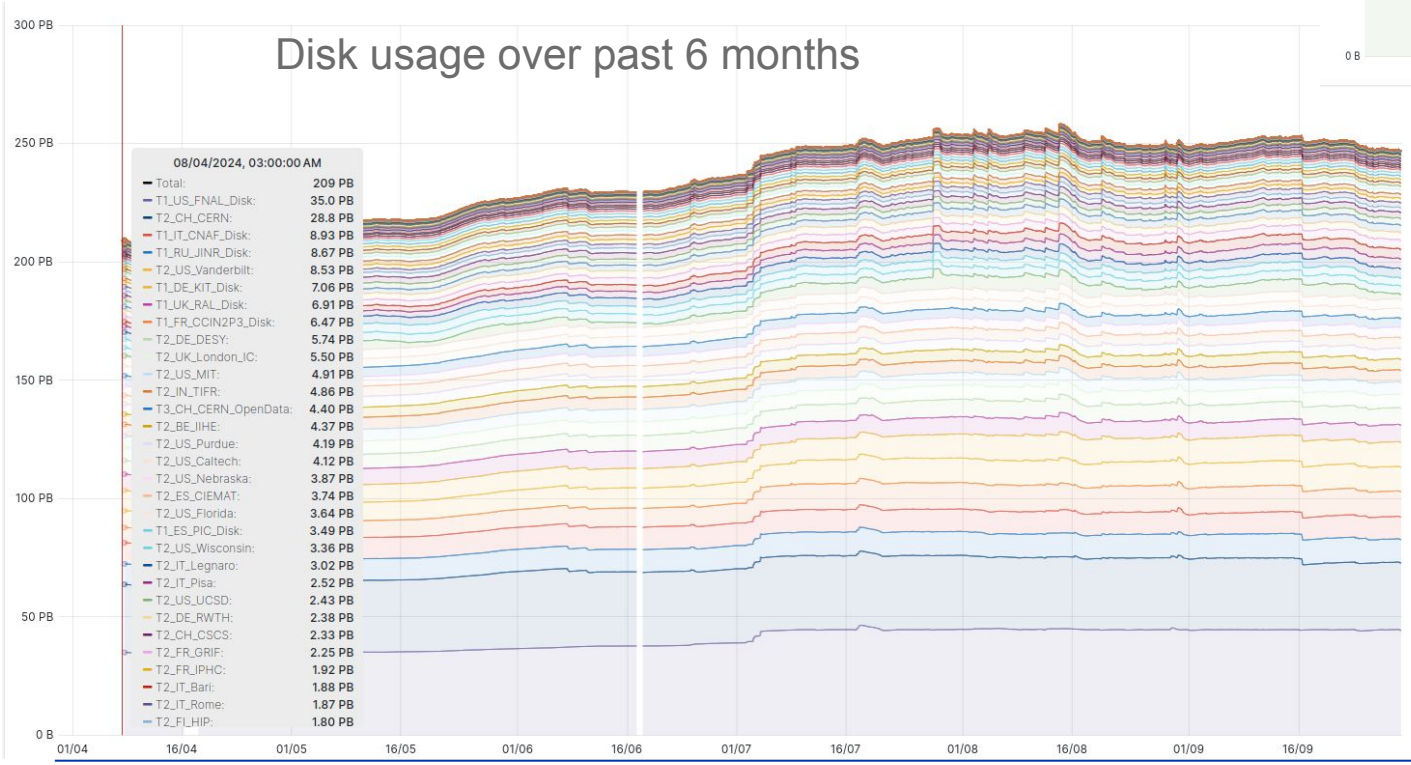
- CMS logical file names (LFNs) are mapped to physical file names (PFNs) using a cascade of regular expressions per site (TFC), via Rucio's pluggable policy packages.
- Now mostly simplified to concatenate a prefix path.

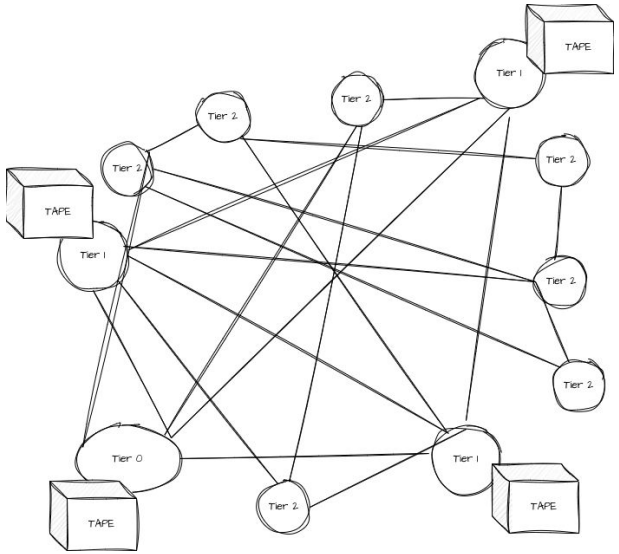
Rucio Infrastructure

CMS Storage Infrastructure

Storage Type	Usage	Tier 0	Tier 1	Tier 2	Tier 3
Disk	200+ PB	1	7	49	31
Tape	550+ PB	1	7	N/A	N/A

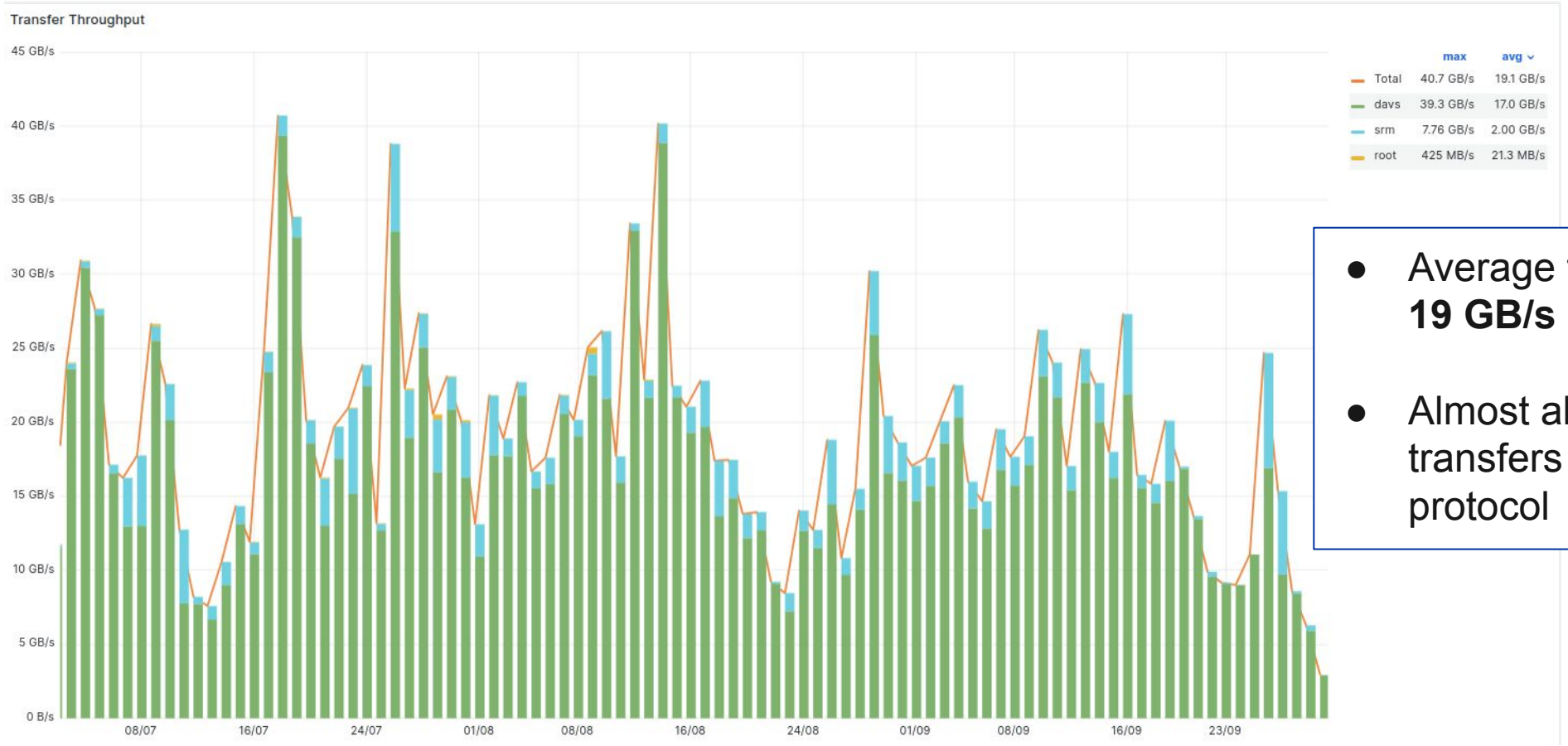






Full mesh topology for data transfers

Data Transfers



- Average throughput of about **19 GB/s**
- Almost all DISK and TAPE transfers migrated to webdav protocol

CMS Workload Management System And Rucio

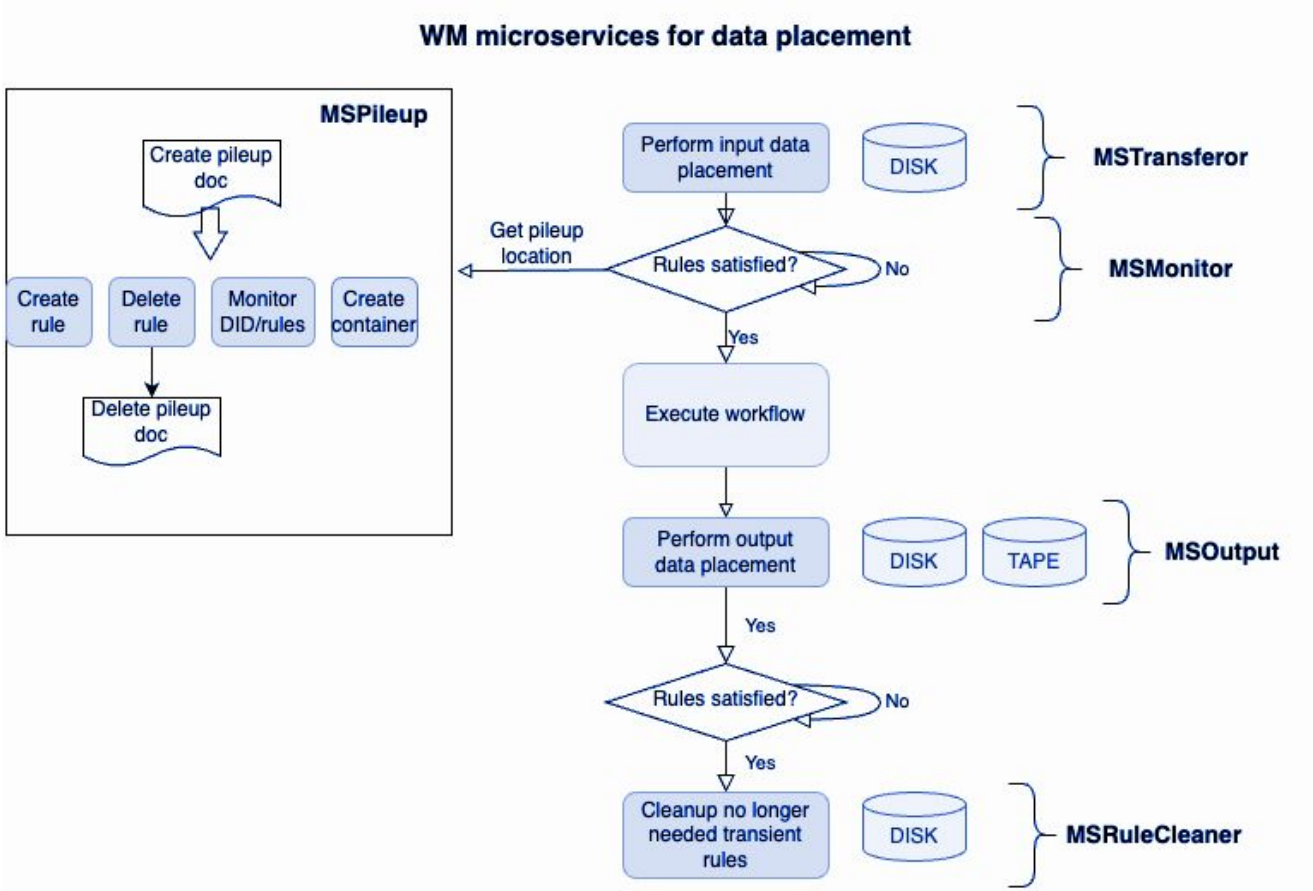
Data placement for WM

- Input data
- Pileup data
- Output data

Different level of locking:

- Transient at workflow execution location (input)
- Transient at origin storage (output)
- Grouping/expiring final @ Disk (output)
- Grouping final @ Tape

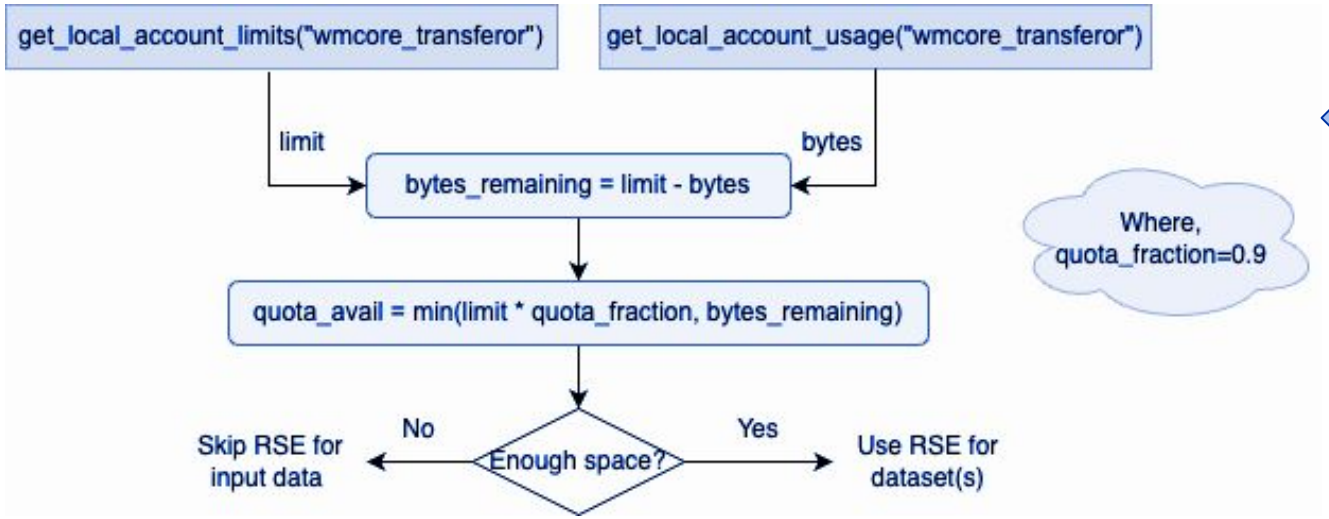
Only keep data locked on Disk if needed by WM!



Data placement for the WM (Cont.)

Data managed through different Rucio accounts, with different quotas and rule construction.

- Input data is directed to specific RSEs (respecting usage vs quota)
- Input pileup data is also directed to specific RSEs.
- DISK output data placement is delegated to Rucio through a generic expression.
  - `(tier=2|tier=1)&cms_type=real&rse_type=DISK`
- TAPE output data placement respects Tape readiness and a dynamic weight expression.
  - `rse_type=TAPE&wmcore_output_tape=True\cms_type=test`



Input data logic

Desired RSEs are pre-selected by CompOps.

Define Stored Workflow for Rules / Replicas

Use case:

- Remove rule monitoring responsibilities from the client for multiple rules
- Make certain stored rule creation formats that can be referenced by name

For e.g. define the below logic for a container:

- Lock 2 replicas - 1 on source site the other somewhere else.
  - On container close: Create TAPE copy
  - Delete the DISK locks when TAPE rule is OK and set lifetime has passed, whichever is later.

- An extension to subscriptions perhaps?
  - Perform on client request and not on DID
  - Our metadata does not live in rucio: hard to create filtering rules; suggestions?

CRAB: CMS Remote Analysis Builder

User-Oriented Tool for Grid Computing

Designed for Physicists to leverage Grid resources for tasks like data analysis, simulations, or any compute-heavy operations.

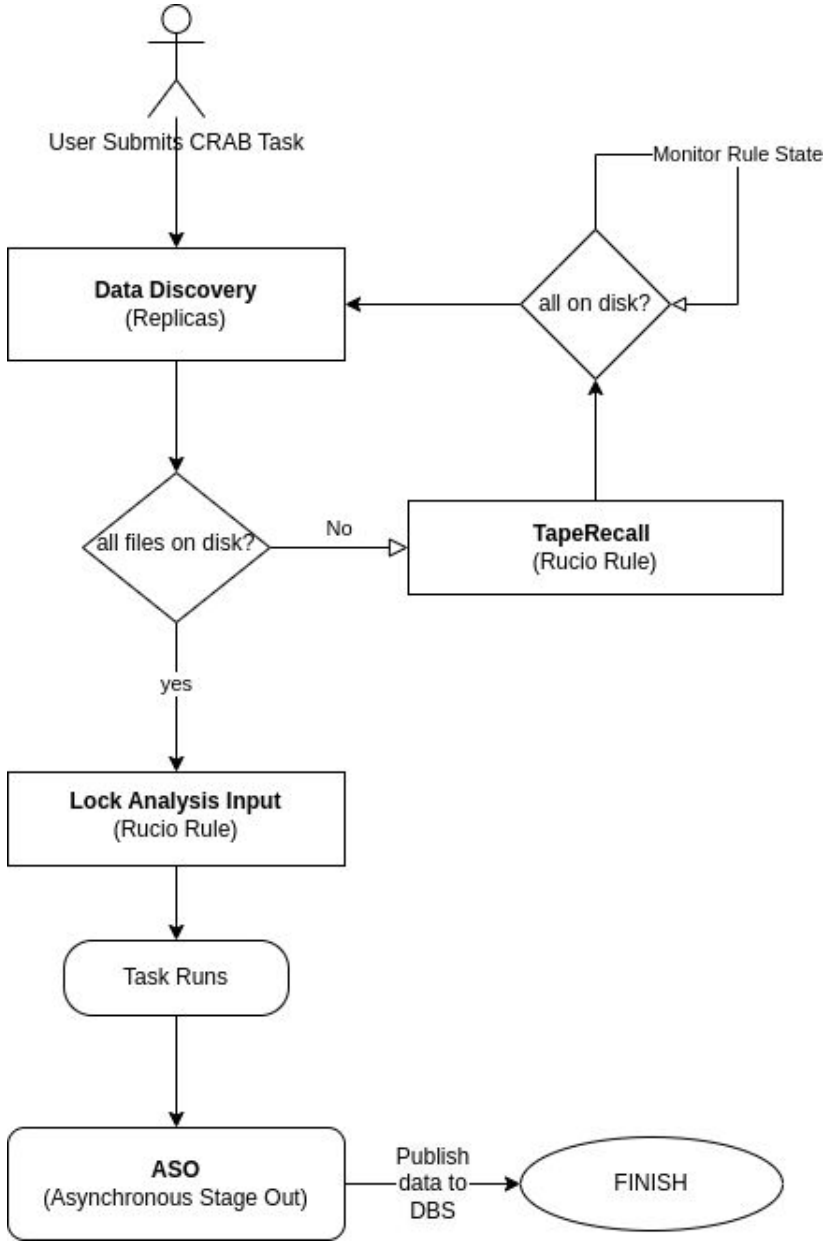
- **Integrated with CMSSW**: A lightweight client embedded in the CMSSW environment.
- **Supports User Code**: Accepts precompiled plugins (.so files) or arbitrary executables.
- **Centralized Service**: Manages communication and status tracking through a central database.

- CRAB converts requests into Grid jobs, executes them with retries, retrieves output files to a user-specified location, and logs the metadata in the Dataset Bookkeeping System (DBS) for further processing.

```
crab submit -c crabConfig_Data_analysis.py
```

CRAB and Rucio

- Locate dataset storage to determine optimal job execution sites.
- Process also datasets described **only as** Rucio Containers, with no DBS information
- **Input Data Management**: **Lock** input data on disk until job completion.
- **TapeRecall:** Utilize Rucio to recall data from tape to disk as needed
- **Rucio Client Integration in CRAB Client**: Provide users with dataset and file location details, data validity, and status (including possible corruption).



ASO (Asynchronous StageOut)

Move user job output to their local user space by leveraging the FTS client

Rucio-Managed CRAB ASO

Output Transfer

- Moves user job output from the temporary RSE (at the execution site) to the user's preferred RSE.

User Scope & Quota Management

- Provides the user with a Rucio container within their own scope, with the transfer charged to their storage quota.

Quota per scope?

- Our policy already restricts user scopes to directories

Work in Progress

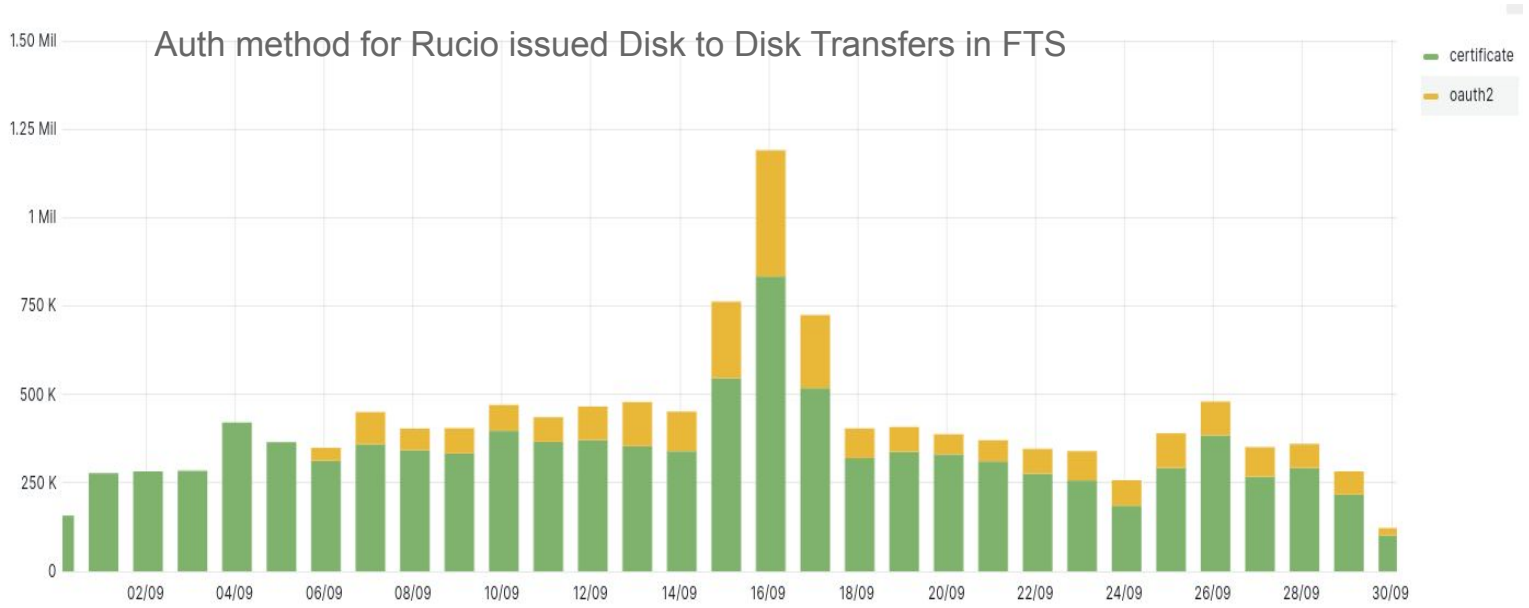
- Permissions need to be finalized.
- Tested only by developers, no user adoption yet

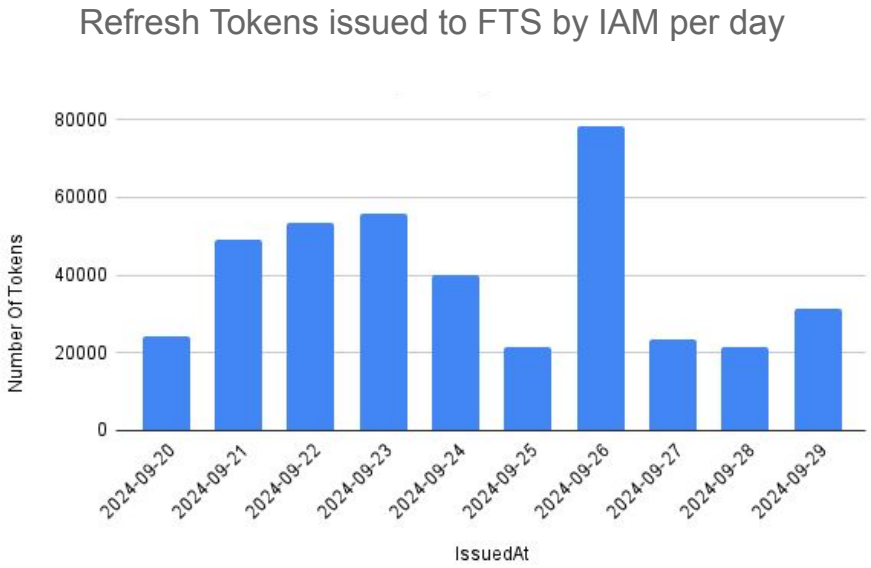
AI Generated: Powered by DALL·E 3

Tokens

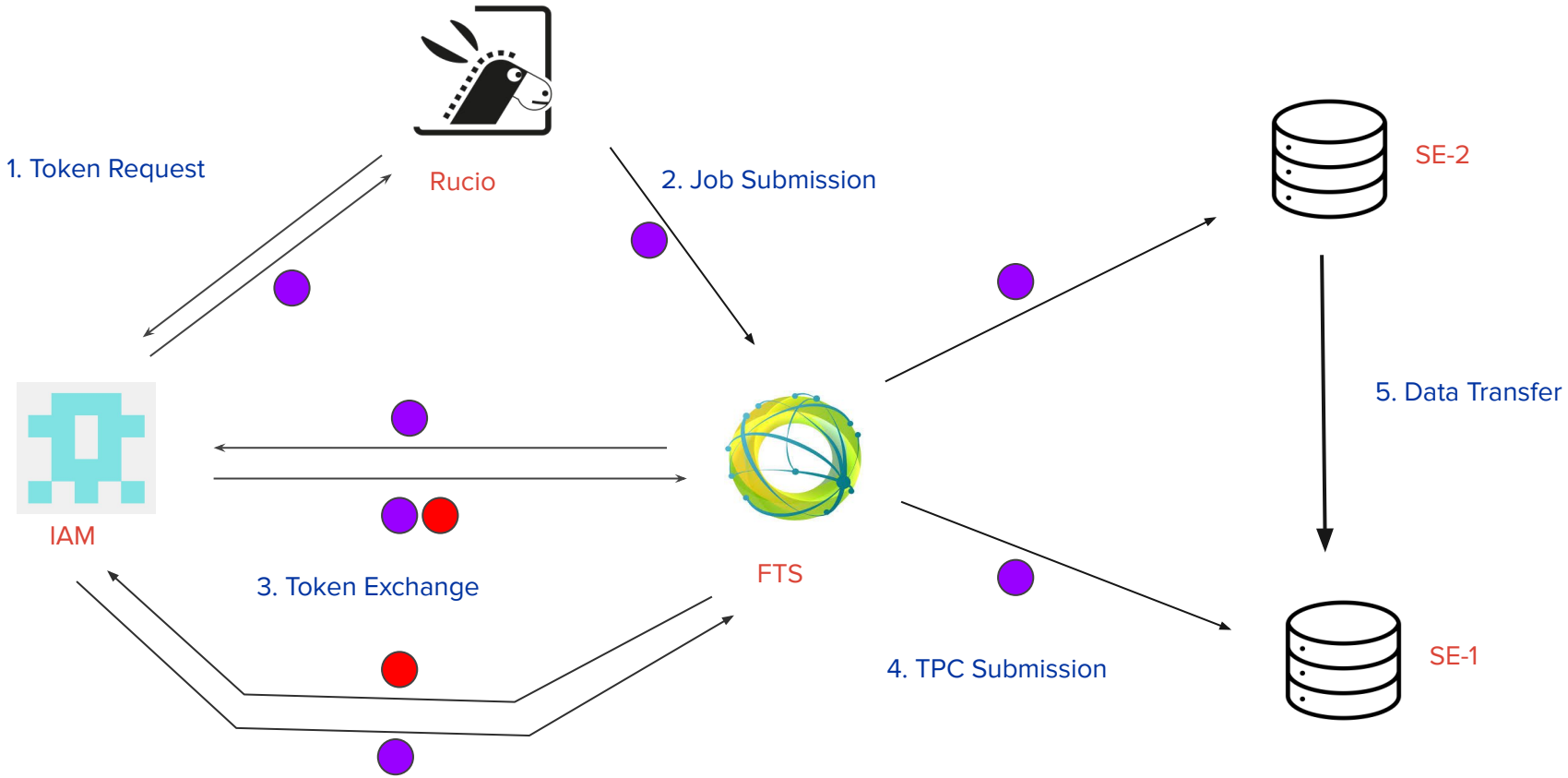
- Running token in production for about 25 days
  - Patched CMS Rucio for setting desired scope
  - About 30% transfers use Oauth2 auth method

- Token are scoped to datasets:
  - CMS datasets can be directly mapped to a directory
- On average 300K Refresh Tokens stored in IAM Database for FTS at any given time





Rucio - FTS - SE : Token Exchange flow



Access Token
Short Lived
Duration of a single transfer
~ 6 hrs

Refresh Token
Long Lived
Accounting for delayed submission; retries etc.
~ 7 days

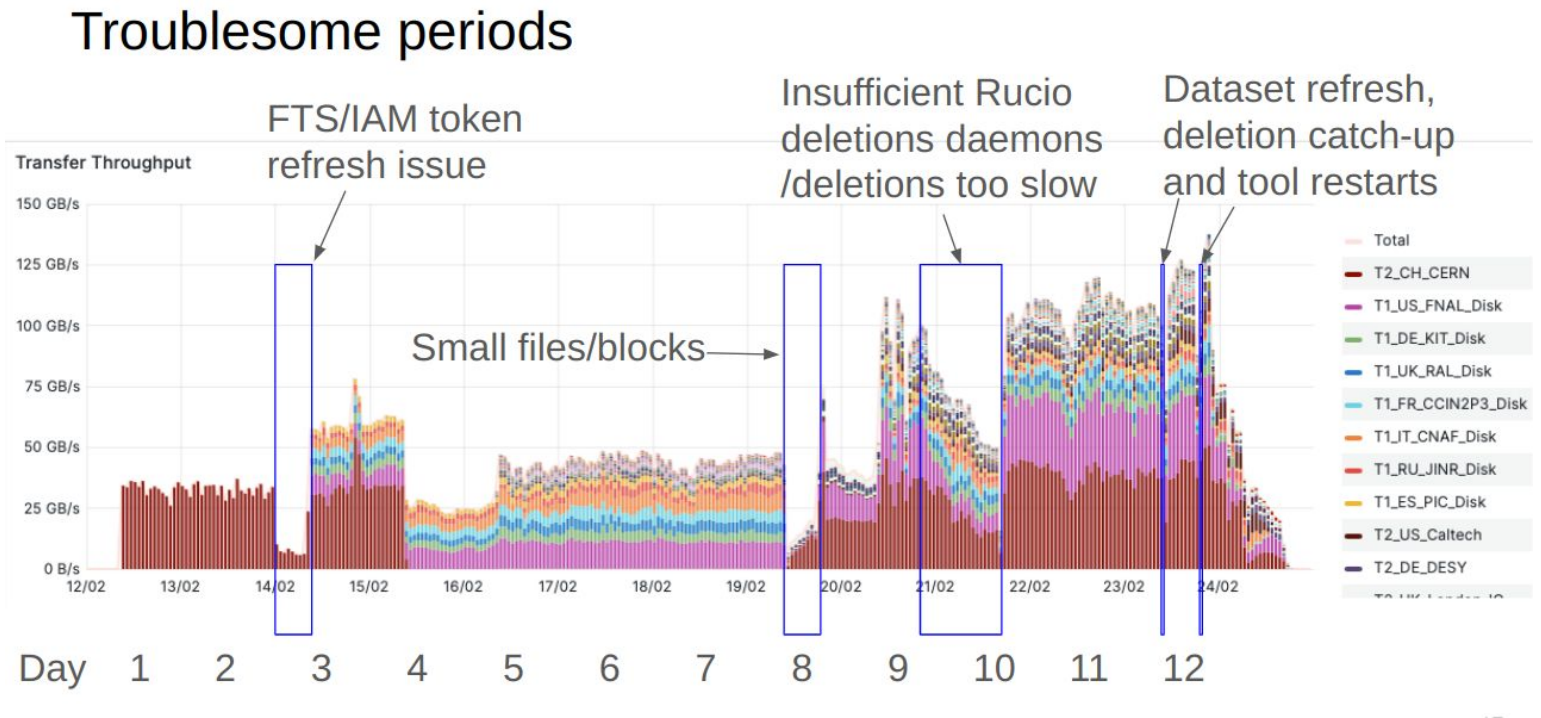
DC-24

DC24 was a great success!

- We achieved our overall target: 1 Tb/s.
- We gained valuable insights, especially on scaling our instances of FTS and Rucio.
- Several sites reached out, noting that the challenge revealed issues they are now actively investigating.

Rucio performed exceptionally well for us.

A big thanks to everyone involved!



Katy Ellis - WLCG/HSF Workshop 2024

Rebrand the DC-Inject Tool

- Was the driver of all transfers during the data challenge.
  - Written by Mario

- Would be great if turned into a daemon
  - Help perform load testing on demand
    - E.g. upcoming Mini Challenges envisioned by Katy

  - Facilitate of various types of load testing:
    - Spike (High Load; Short Duration)
    - Stress (High Load; Long Duration)
    - Soak (Average Load; Very Long Duration)
    - Breakpoint (Increasing load till failure): Fun one!

New Daemons Deployed

- **BB-8 (Rebalancer)**

- **Suspicious Replica Recoverer**

- **Conveyor Throttler**
  - Can manage the queue of transfer requests in Rucio
  - Add a transfer to the queue based on the transfer limits of the current active transfer
  - Good to have for emergencies!!

Daemons - BB8

- **Responsible for rebalancing data between RSEs**
  - Keep the sites' lock space safe
  - Move data out of a specified location
- **How it works?**
  - Single RSE
    - Move rules having generic RSE expression* from specified site to other site fulfilling the expression
  - RSE Expression
    - Global ratio across all the matching RSEs to determine which sites are relatively full or empty
    - Data is rebalanced from RSEs above the ratio to those below the ratio
    - Amount to rebalance is determined dynamically based on the relative fullness of the RSEs
  - Keep number of replicas
  - Move Datasets with only 1 Lock (grant we are freeing space)
  - Configuration and attributes passed through rucio config file and CLI parameters
- **Implementation**
  - Test pod continuously running. No real rebalancing being performed

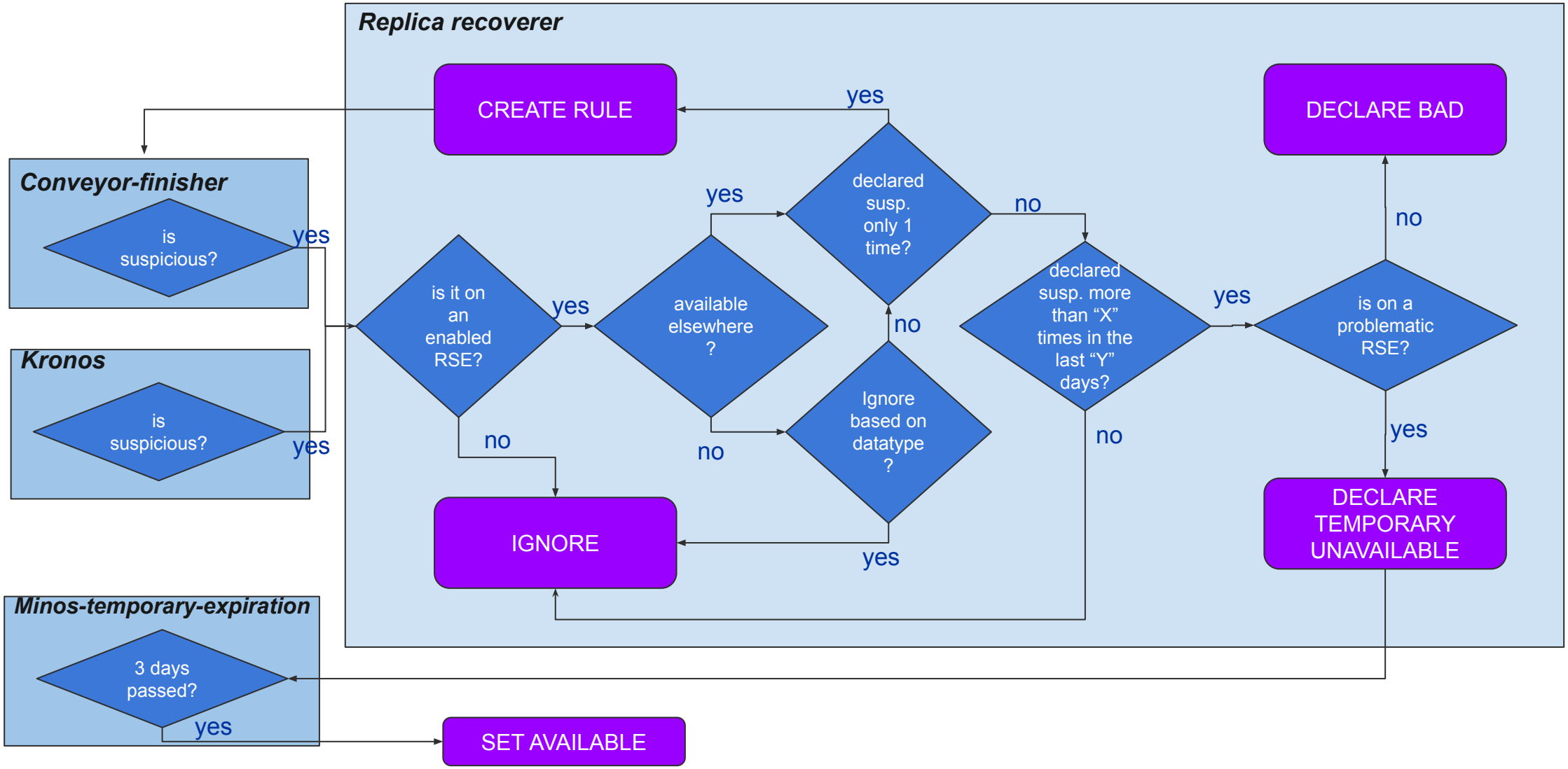
Daemons - BB8

- **Experience**
  - 100TB experiment triggered 1.8PB of transfers
  - Multiple replicas created for the same file
  - Several unnecessary transfer requests

- **Problems**
  - Logic based on move dataset, not in moving rule
    - Rules over containers are moved N times (creating N replica numbers)
    - Imprecise locks count. Some datasets can have 1 and others can have more
    - Real space saved miscalculated. Move containers, calculate for datasets (rule picked because dataset A has 1 lock and dataset B not because it has 2)
  - Tied up to only one site, we lose the generic RSE expression advantages

- **Potential Solutions**
  - Change Dataset approach to rule approach
  - Keep original RSE Expression excluding the original site and sites above the quota

Daemons - Suspicious replica recoverer



Daemons - Suspicious replica recoverer

- **Several complaints from users each month about corrupt replicas**
- **Daemon wasn't ready to be used by CMS out of the box due to following issues:**
  - Experiment specific code
    - RSE expression based on experiment specific RSE attrs
    - Requires non-empty "datatype" field (CMS doesn't use this DID field)
      - Needed to decide which files to ignore
      - **Solution**: Determine datatype of files based on a configurable regex
  - PFN to scope:name translation wasn't compatible with CMS
    - Prevented conveyor-finisher to declare suspicious replicas
    - **Solution**: Plugin based PFN to LFN translation

Daemons - Suspicious replica recoverer

- **All these issues are fixed and we deployed the daemon last week!**
  - Monitoring daemon activity for a single RSE

- **Communities should now be able to adapt the daemon without extra development**
  - Many thanks to Cedric and Christoph for answering our questions and Maggie for helping with development

- **Concerns:**
  - Not possible to ignore files if they have multiple replicas (CMS SAM files)

Todos ..

- Operations dashboards for daemons
- A test client for all major api endpoints
- Explore migration of CMS-loadtests to “rucio-automatix”.
- Configuring patches through flux
- WebUI 2.0 Deployment

FeedBack

CRAB

- **Rucio Python Client: Better Documentation Needed**
  - Easy to use, but lacks documentation
  - Had to refer to REST API source code for input dicts
  - Return values and exceptions learned only through trial and error

- **Happy with server uptime:** “It worked so smoothly that we did not put try/except and the rare time that rucio server is down we fail miserably w/o wait-retry loops”.

- **Slow/stuck rules - we need better information to guide troubleshooting**
  - WebUI error reporting is limited, with missing FTS links for transfers
  - Access to successful FTS and Failed FTS jobs would aid troubleshooting
  - Often forced to browse FTS raw data in OpenSearch

FeedBack

Workload Management

- **WM happily explores the concept of Rucio containers to create custom pileup containers with only a subset of the data.**
  - Creating new DID container plus attaching already existing dataset DID(s)

- **Could profit of extra rule relationship to ease data management in WM**

- **WM could leverage from a more functional Rucio Integration environment, providing:**
  - Protection/isolation for real DID removal (in case it shares the same actual data as Rucio Prod)
  - Friendly process to migrate data between Rucio instances

Thank you

Rucio works extremely well for CMS!

We owe it to the commitment of an outstanding team and are grateful to be part of such a strong and supportive community.

Image Creator in Bing: Powered by DALLE-3
Prompt: CMS similar to its logo as a particle detector and Rucio similar to its logo as a donkey shaking hands with LHC ring in the background or around them, making it look like a friendly collaboration