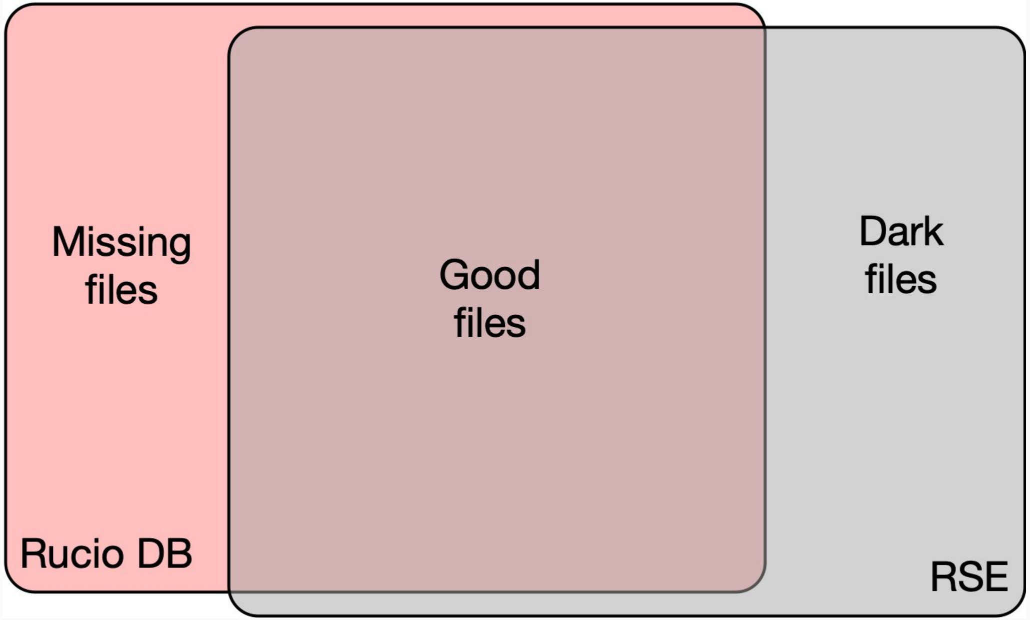
CMS Rucio Consistency

Igor Mandrichenko, Stefan Piperov, Eric Vaandering
Rucio meeting, April 8 2021

Purpose



To make sure Rucio database accurately reflects actual state of RSEs by comparing replica list in the DB and fund in the RSE

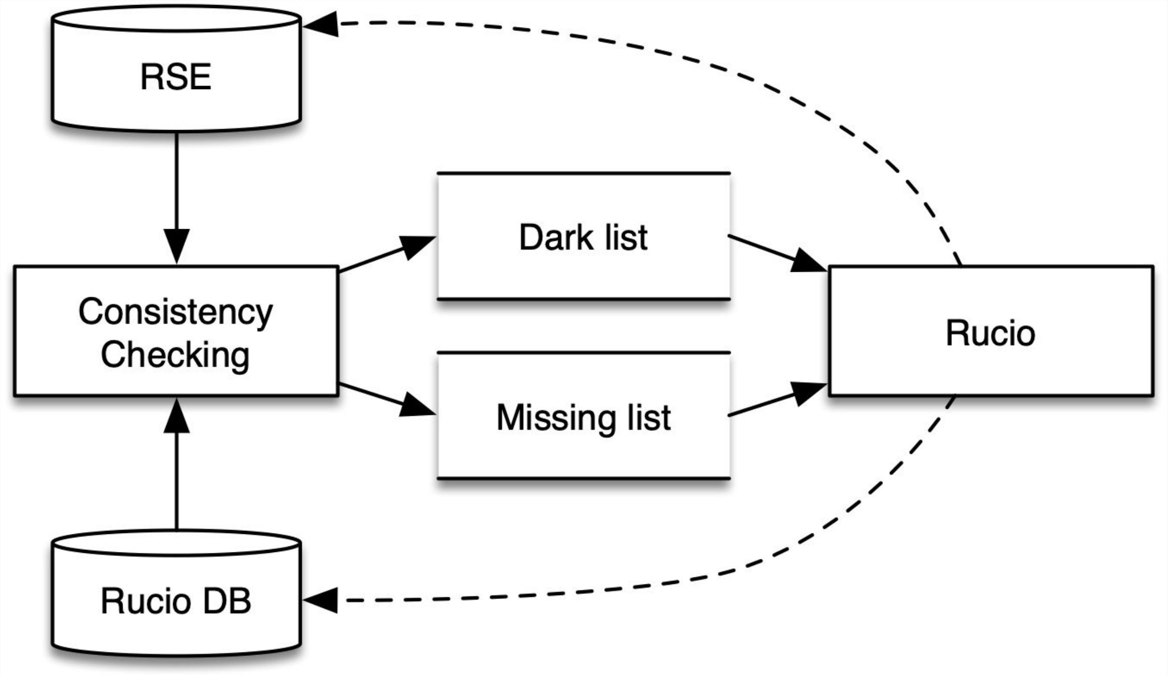
Missing replicas:

> replicas which are expected to be found in the RSE, but are not
>
> Need to be re-replicated

Dark replicas:

> Replicas which are not supposed to be in the RSE
>
> Occupy space
>
> Need to be deleted

Rucio Consistency



Consistency Checking produce dark and missing lists and feed them back to Rucio

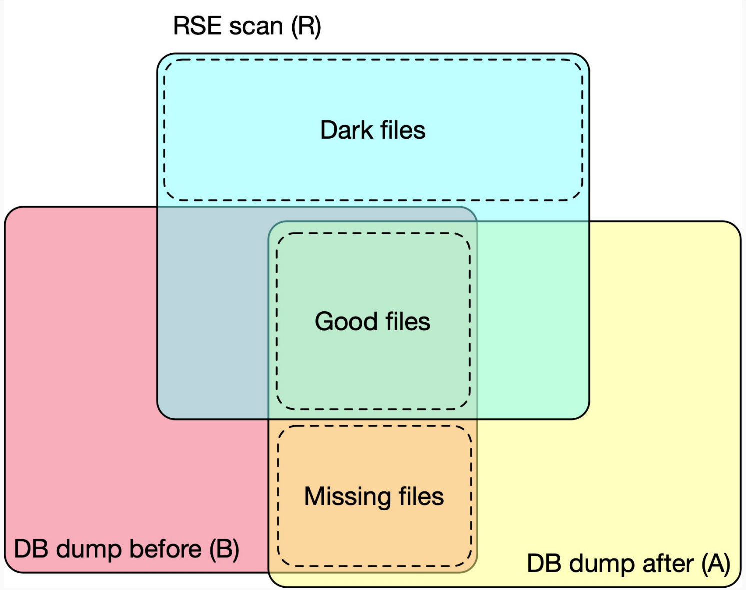
Mostly done by the Auditor daemon

Difficulties

Neither site dump nor the database dump can represent a consistent snapshot

- Take minutes to hours to produce
- Done at different times
- Both database and site state constantly change

3-way comparison

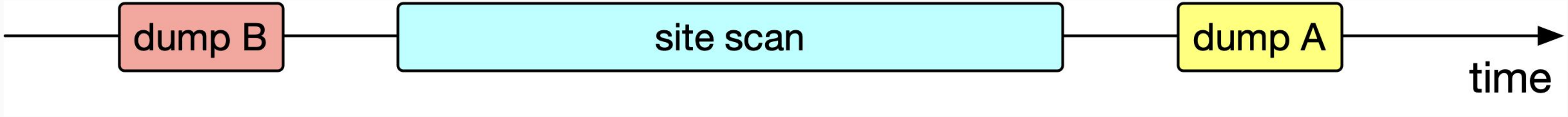


1. List files in DB -> set “B” (before)
2. Scan RSE -> set “R”
3. List files in DB again -> set “A” (after)

Dark = R - (A+B)

Missing = (A*B) - R

Inconsistencies of both kinds guaranteed to be discovered *eventually*



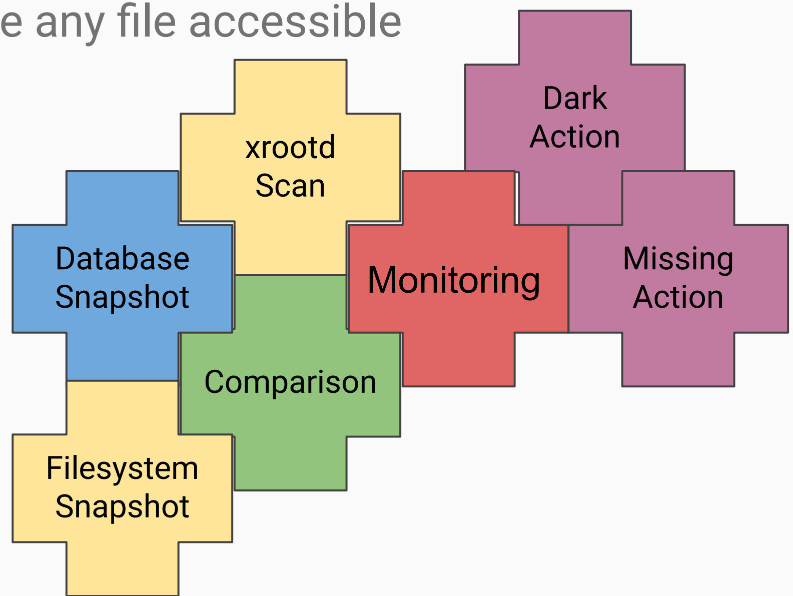
CMS motivation to change things

CMS has traditionally done this checking in a very different way from ATLAS

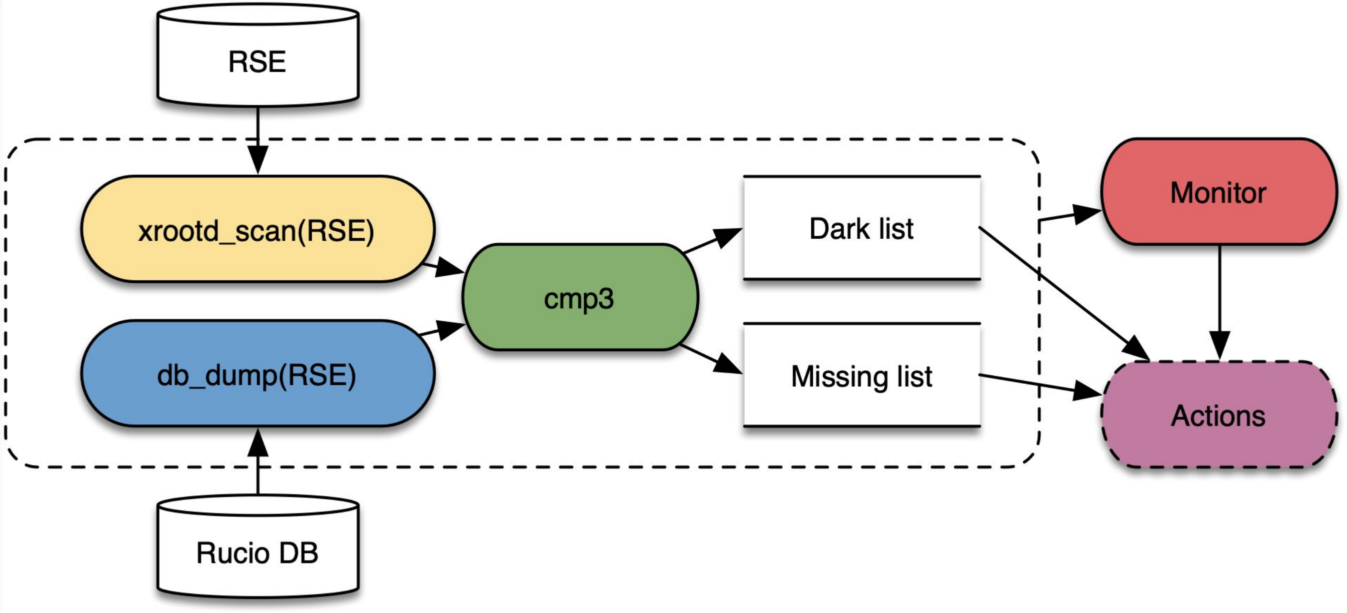
Rather than rely on sites to supply snapshots of storage over time (tried), CMS centrally uses xrootd to scan sites

> Built on our heavy use of xrootd (AAA) to make any file accessible anywhere

Instead of using the Auditor as is, integrated into Rucio, we've broken it down into pieces which can be treated as a "toolbox" outside of Rucio to check consistency (and extended for new/other uses)

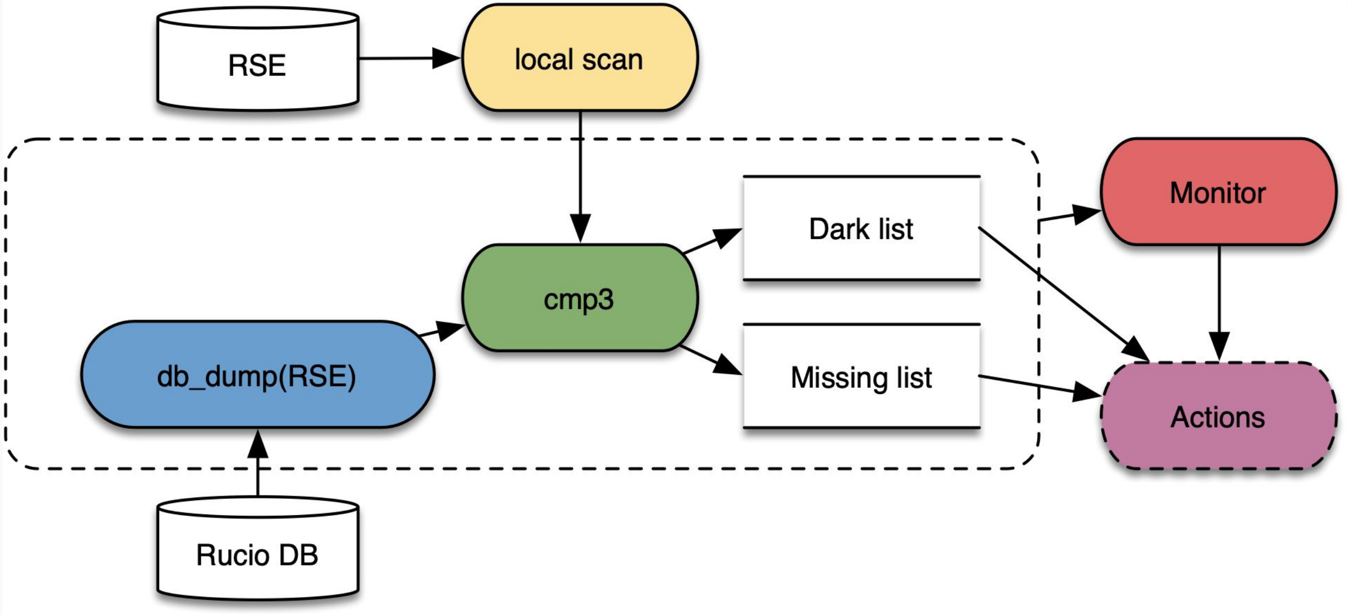


CMS Consistency Architecture



“Standard” xrootd-scannable RSE

CMS Consistency Architecture



Non-scannable RSE

Files are big !

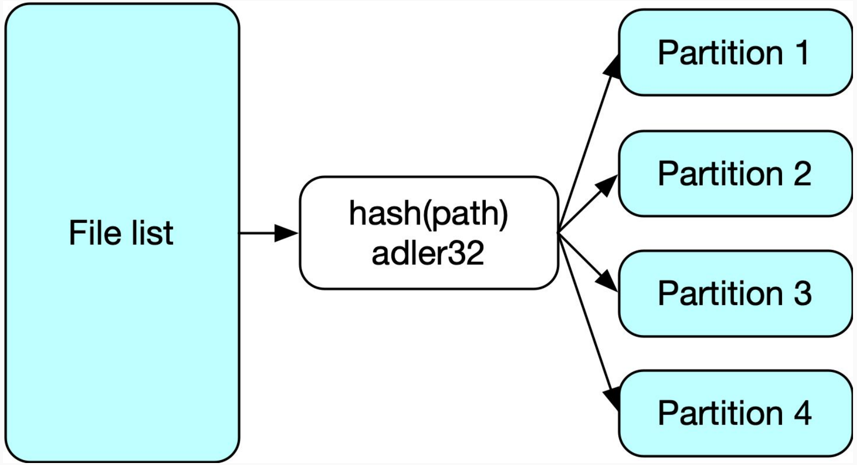
File lists (B, R, A) can get large, up to ~10GB

- 100M files * 100 bytes/file = 10GB

Straightforward approach:

- Sort each list, then compare line-by-line
- Sorting takes a lot of time*memory
    - Either try to sort in memory - faster but need up to 10GB of memory
    - Or sort using disk as the buffer - slow

Partitioning



Instead of sorting/comparing entire lists:

- Split B,R,A lists into multiple files
- Use common hash function to send each file path to its own partition 100MB-1GB size
    - part_inx = hash(path) % N
- Compare each partition separately in memory
    - B_0, R_0, A_0 -> D_0, M_0
    - B_1, R_1, A_1 -> D_1, M_1
    - ...
- Only one of the 3 lists needs to be held in memory, 2 others are scanned line by line

Tools: xrootd_scanner.py

- Operates on single RSE
  - Xrootd server host/port
  - List of directories to scan
- Uses “xrdfs ls” spawning shell subprocess
- Spawns multiple subprocesses to scan in parallel (configurable)
- Tries to scan recursively first (xrdfs ls -R) and non-recursively as the fall-back
- Configurable on per/RSE basis with defaults
- Converts physical paths to LFNs (configurable)
- Produces partitioned R list

Tools: db_dump.py

- Reads Rucio database replicas table to find all “active” replicas for the RSE
- Uses SQL Alchemy
- Produces partitioned list of LFNs (B and A)

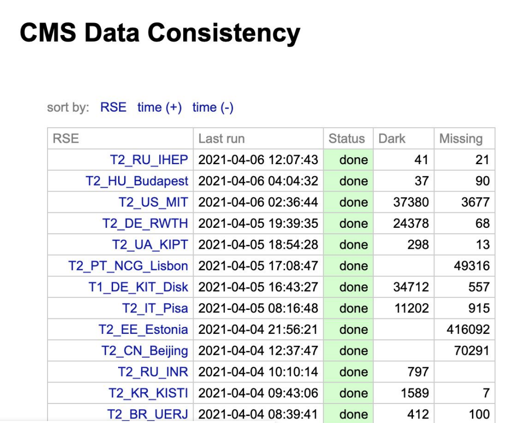
Tools: cmp3.py

- Compares 3 partitioned lists $(B_0, B_1, B_1, \ldots, B_N)$, $(R_0, R_1, R_1, \ldots, R_N)$, $(A_0, A_1, A_1, \ldots, A_N)$
- Produces 2 files
  - Dark list
  - Missing list

Tools: partition.py

- Can be used to partition a single file or re-partition a partitioned list
- Has some line filtering/editing capabilities (regular expressions)
- Can be used to partition site dump produced by “non standard” sites, which can not be scanned by the xrootd scanner (RAL, CTA, etc.)

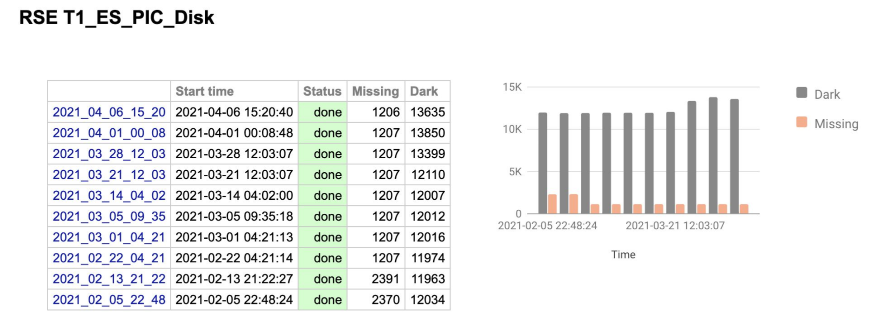
Web GUI Monitor



CMS Data Consistency

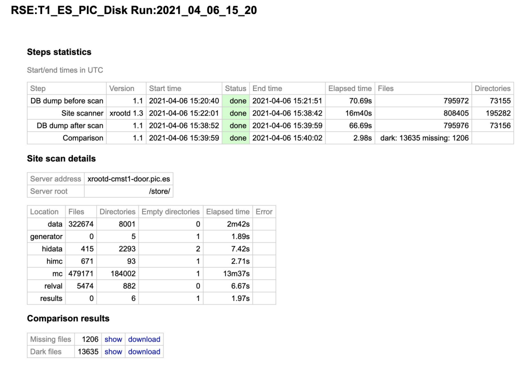
sort by: RSE time (+) time (-)

RSE	Last run	Status	Dark	Missing
T2_RU_IHEP	2021-04-06 12:07:43	done	41	21
T2_HU_Budapest	2021-04-06 04:04:32	done	37	90
T2_US_MIT	2021-04-06 02:36:44	done	37380	3677
T2_DE_RWTH	2021-04-05 19:39:35	done	24378	68
T2_UA_KIPT	2021-04-05 18:54:28	done	298	13
T2_PT_NCG_Lisbon	2021-04-05 17:08:47	done		49316
T1_DE_KIT_Disk	2021-04-05 16:43:27	done	34712	557
T2_IT_Pisa	2021-04-05 08:16:48	done	11202	915
T2_EE_Estonia	2021-04-04 21:56:21	done		416092
T2_CN_Beijing	2021-04-04 12:37:47	done		70291
T2_RU_INR	2021-04-04 10:10:14	done	797	
T2_KR_KISTI	2021-04-04 09:43:06	done	1589	7
T2_BR_UERJ	2021-04-04 08:39:41	done	412	100
	7:24:31	done	1823	113
	0:03:25	started		
	5:48:21	done		1467961



RSE T1_ES_PIC_Disk

	Start time	Status	Missing	Dark
2021_04_06_15_20	2021-04-06 15:20:40	done	1206	13635
2021_04_01_00_08	2021-04-01 00:08:48	done	1207	13850
2021_03_28_12_03	2021-03-28 12:03:07	done	1207	13399
2021_03_21_12_03	2021-03-21 12:03:07	done	1207	12110
2021_03_14_04_02	2021-03-14 04:02:00	done	1207	12007
2021_03_05_09_35	2021-03-05 09:35:18	done	1207	12012
2021_03_01_04_21	2021-03-01 04:21:13	done	1207	12016
2021_02_22_04_21	2021-02-22 04:21:14	done	1207	11974
2021_02_13_21_22	2021-02-13 21:22:27	done	2391	11963
2021_02_05_22_48	2021-02-05 22:48:24	done	2370	12034



RSE:T1_ES_PIC_Disk Run:2021_04_06_15_20

Steps statistics

Start/end times in UTC

Step	Version	Start time	Status	End time	Elapsed time	Files	Directories
DB dump before scan	1.1	2021-04-06 15:20:40	done	2021-04-06 15:21:51	70.69s	795972	73155
Site scanner	xrootd 1.3	2021-04-06 15:22:01	done	2021-04-06 15:38:42	16m40s	808405	195282
DB dump after scan	1.1	2021-04-06 15:38:52	done	2021-04-06 15:39:59	66.69s	795976	73156
Comparison	1.1	2021-04-06 15:39:59	done	2021-04-06 15:40:02	2.98s	dark: 13635 missing: 1206	

Site scan details

Server address	xrootd-cmst1-door.pic.es
Server root	/store/

Location	Files	Directories	Empty directories	Elapsed time	Error
data	322674	8001	0	2m42s	
generator	0	5	1	1.89s	
hidata	415	2293	2	7.42s	
himc	671	93	1	2.71s	
mc	479171	184002	1	13m37s	
relval	5474	882	0	6.67s	
results	0	6	1	1.97s	

Comparison results

Missing files	1206	show	download
Dark files	13635	show	download

https://cmsweb-k8s-prod.cern.ch/rucioconmon/index

Actions

We rely on existing Rucio methods to perform the actions on dark/missing replicas.

- add_quarantined_replicas() to delete dark files using the Dark Reaper;
- declare_bad_file_replicas() to re-transfer the missing replicas;

The plan is to encapsulate the whole functionality of deleting/re-transferring the dark/missing files in a separate daemon, which will be using the output files from the scanner and 3-way comparison for all RSEs.

We want to be safe in the automated deletions, so

  - Perhaps we should check that a file shows up as ‘dark’ in two consecutive scans before deleting.
  - And the file has to be sufficiently old (a month?), to avoid acting on transient files.

Future

- Continue development of the part, which acts on the findings
  - What to do with dark files? Are we brave enough to automatically delete them?
  - Gray files - files which are legitimately in storage already but have not been declared to Rucio yet
  - Non-standard sites (not xrootd-friendly)
  - RAL
  - CTA
  - The set of tools we have seems to work well so far
- Right now this code exists in our own repos but we would like to contribute it to the Rucio project
- We are still testing in the integration deployment. When we gain sufficient confidence that everything works as expected we'll start using it in production.