



Rucio in DOMA

[Martin Barisits](#) (CERN)

Contents



Update on some “hot topics” related to DOMA

- DC24 improvements
- dc_inject tool evolution
- Tokens
- Deletion thoughts

DC24 improvements (1/2)



Summary

- The communities using Rucio reported overall success!
 - Rucio proved to be able to scale and meet the demands of the Data Challenge

Three areas of improvement identified

Sidenote: The DC24 injection method is somewhat artificial and aggravated some of the observed issues

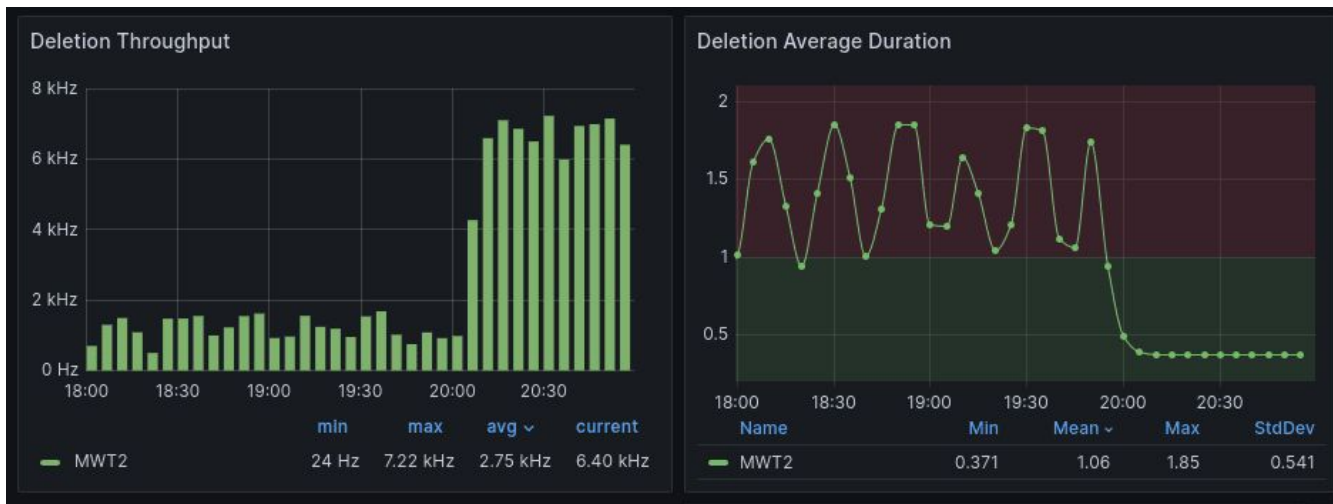
- Skip injecting transfers of expired rules [#6505](#) **MERGED**
- Contention on deleting large rules with ongoing transfers [#6511](#) **ONGOING**
 - Annoyance with no significant effect on overall performance
- Deletion overlap on Slower Sites [#6512](#) **PARTIALLY DONE**
 - Underlying issue is the rate of deletion at the sites themselves
 - Rucio does not handle that well and unnecessarily repeats deletion (making the situation worse)

DC24 improvements (2/2)



Some news on deletion:

- Optimisation work in Rucio reduced the overhead in deletions, over http, significantly
- See [#7386](#) for details



- **dc_inject tool was used in DC24 by several experiments**
 - See Mario's talk in DC24 workshop for [details](#)
 - Executive summary: it's a script, running in a loop, retrieving unique datasets per RSE and submitting rules to inject transfers
- **Works reasonably well, but not trivial to use**
 - It runs as an “external” application to Rucio, but requires an expert operator for input handling
 - Thus, practically impossible to be used without that operator to run e.g. a mini-challenge
 - Has a few shortcomings and needs quality-of-life improvements
- **Idea: Intake this functionality into Rucio**
 - Xuantong Zhang (IHEP, ATLAS) started to work on this in November and made good progress
 - Rucio CLI to create “injection-plans” which can be activated
 - Backend injection daemon which continuously executes the injection plan
 - Work is still ongoing

Tokens (1/4)



- Apologies for bringing up tokens, again... 😊
- Rucio is one of the focal points in the token ecosystem
- Overall difficult and multifaceted challenge for us
 - Need to give the communities/experiments the choice to utilise tokens to their preference, risk, service level, and security profile
 - Lifetimes
 - Scopes
 - Token profile (Essentially only WLCG for now)
 - Token issuers (IAM, CiLogon, EGI Checkin, self-mint?, ...)
 - At the same time need to produce secure, efficient, concise, and **maintainable** functionality
- Other communities are looking at WLCG to follow our path
 - Timelines are however different and more demanding (E.g. DUNE)

Tokens (2/4)



- Perfect world model
 - File specific tokens for EVERY Rucio community, somewhat replicating what pre_signed URLs and ALICE do
 - Most secure, and we would only have to maintain one model
 - Some communities require that level of security/isolation (Life sciences, ...)
 - **Unrealistic**, and for ATLAS/CMS a less specific model is acceptable and provides multiple benefits
- Envisioned model for ATLAS & CMS, with some differences between the two
 - Motivated by three factors
 - Security implications (Fallout when a token leaks)
 - Pressure on token-issuer (Request rate of Access Tokens)
 - Token-issuer stability (Fallout when the token-issuer is down)
 - RSE-wide/Dataset specific storage.read and storage.create tokens with long(er) lifetimes for TPC
 - File/Dataset specific storage.modify tokens with long(er) lifetimes for TPC
 - Might not even be needed, see Mihai's DOMA BDT talk later
 - → O(1-10M) token requests per day

Tokens (3/4)



- **Fallout scenarios**
 - **Token leaks**
 - RSE-wide/Dataset specific storage.read → Attacker can read our data → **Acceptable**
 - RSE-wide/Dataset specific storage.write → Attacker can fill up storage → **Acceptable**
 - File/Dataset specific storage.modify → Attacker can delete the dataset/file, but nothing else → **Acceptable**
 - **Token-issuer is down**
 - Transfers queued in FTS will continue normally, if long enough lifetime was given
 - New transfers can be submitted to FTS, but only without the semi-required storage.modify token
 - Transfers which require overwrite/cleanup will fail → **Acceptable**
 - Jobs can continue by being served cached (long lived) tokens for reading/writing
 - Central deletion stops
 - Storages will eventually run full → **Fallout over time to be investigated**
 - Could be mitigated by longer lived RSE-wide tokens for central deletion (Probably ok, since that token is just used between Rucio & storage, no other service is involved)
 - Investigate: Binding central deletion tokens to service certificates ([RFC 8705](#), [RFC 9449](#))

Tokens (4/4)



- **Alternative model: self-minting of tokens**
 - Mitigates the token-issuer stability and pressure concern
 - Introduces other concerns: Complexity (For Rucio and the entire ecosystem), security policy (revocation, etc.), and introduces yet another model to maintain

Deletion thoughts



- Current model
 - Rucio central deletion workflow deletes files one after the other
 - Agent selects one storage and deletes ~1k files (one after the other)
 - Thus deletion of one file is dominated by several factors: Connection to storage + Authentication overhead + Actual deletion
 - For one storage, the maximum of parallel deletion threads per storage is capped to **5** (adaptable per storage)
 - At request of sites, to protect storage from overload
- Is this model sufficient for HL-LHC? Rucio can scale this by increasing the number of parallel deletion threads (5 → 25/50/100?) but will sites accept that?
- Do we need to look into different solutions (with storage) → http pipelining/multiplexing, bulk submissions, etc.
- DOMA should track deletion throughput as a (minor?) target next to network throughput

Additional information



- Website  <http://rucio.cern.ch>
- Documentation  <https://rucio.cern.ch/documentation>
- Repository  <https://github.com/rucio/>
- Containers  <https://hub.docker.com/r/rucio/>
- Online support  http://rucio.cern.ch/doc../join_rucio_mattermost/
- Journal article  <https://doi.org/10.1007/s41781-019-0026-3>
- Mastodon  <https://fosstodon.org/@rucio>

