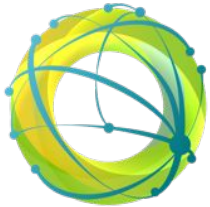
FTS & XRootD Workshop

The Cosener's House, Abingdon, UK, 9-13th September 2024



FTS

File Transfer Service

FTS in the Token World

Mihai Patrascoiu
on behalf of the FTS team

CERN

FTS Token Plan

(presented @ DOMA-BDT, 21st June 2023)

- Standard OAuth2 token support
- Token submission: 1 token to identify with FTS

  \+ 2 tokens per transfer (source and destination)
- DC24 is considered perfect time to test standard OAuth2 flow
- Client-facing developments will be released first
  - Other systems can follow-up on this early
- Tape plan decoupled from TPC → details to follow after DC'24

FTS & Token Commitments

FTS commitments for the DataChallenge ‘24

- Full SciTags Support (Packet Marking)
- Functional OAuth2 Tokens Support (WLCG)
- Monitoring info for SciTags & Token adoption
- Token deployment at CERN for LHC experiments

Beyond

- Token support in official release
- Improve system based on DC’24 feedback
- Discussion on tokens + tape

FTS Token Timeline

June 2023

FTS Token plan presented

"Pre-DC'24 Workshop" Development

- Token development
- Deployment on FTS Pilot/ATLAS/CMS/LHCb
- "Alpha" version of token support

Pre-DC'24

- Refined DC'24 token development

DC'24

Post-DC'24 Era

- Improved congested SQL queries
- Token tests (Aug 2024)

Remaining 2024

- Just-in-time refresh?
- Tape considerations?

The Way of the Token Transfer

FTS Token Outline

- Submission of transfer **using tokens**
- Transfer goes to new `TOKEN_PREP` state
  - **FTS fetches refresh token**
- Transfers goes to `SUBMITTED` state
  - **FTS refreshes access token (as needed)**
- Transfer Agent process starts **(with tokens)**
- Transfer data is reported to Monit system

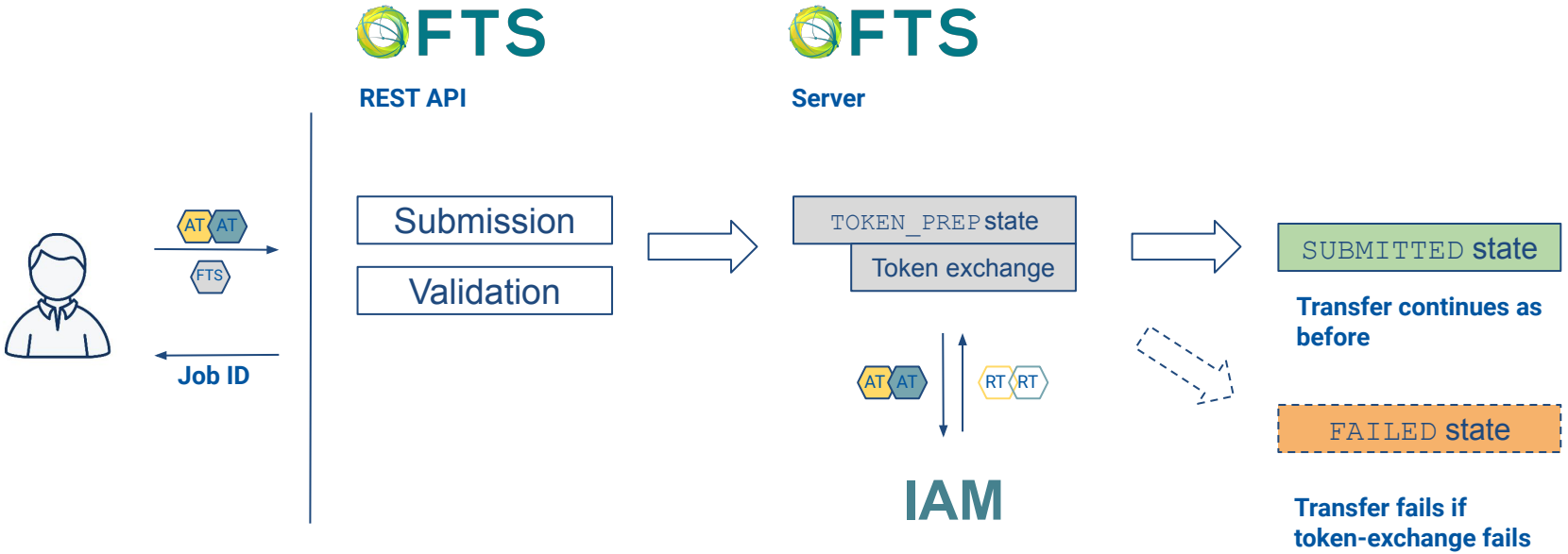
1. Client submission

A **token for each transfer** must be provided!

```
{
  "files": {
    "sources": [URL1, URL2, ...],
    "destinations": [URL3, URL4, ...],
    "checksum": <xsum>,
    "filesize": <size>,
    "metadata": <metadata>
  },
  "params": { ... }
}
```

```
{
  "files": {
    "sources": [URL1, URL2, ...],
    "destinations": [URL3, URL4, ...],
    "source_tokens": [AT1, AT2, ...],
    "destination_tokens": [AT3, AT4, ...],
    "checksum": <xsum>,
    "filesize": <size>,
    "metadata": <metadata>
  },
  "params": { ... }
}
```

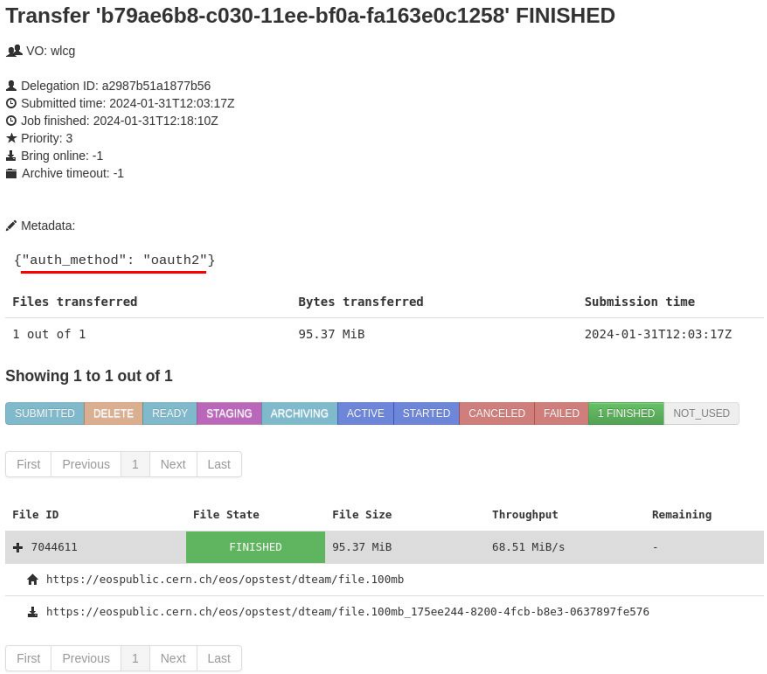
2. The TOKEN_PREP state



2. The TOKEN_PREP state

- Transfers without a refresh token go into **`TOKEN_PREP`** state
- Moves to **`SUBMITTED`** after token-exchange

3. Transfer is scheduled



Transfer 'b79ae6b8-c030-11ee-bf0a-fa163e0c1258' FINISHED

VO: wlcg

Delegation ID: a2987b51a1877b56
Submitted time: 2024-01-31T12:03:17Z
Job finished: 2024-01-31T12:18:10Z
Priority: 3
Bring online: -1
Archive timeout: -1

Metadata:

```
{"auth_method": "oauth2"}
```

Files transferred	Bytes transferred	Submission time
1 out of 1	95.37 MiB	2024-01-31T12:03:17Z

Showing 1 to 1 out of 1

SUBMITTED | DELETE | READY | STAGING | ARCHIVING | ACTIVE | STARTED | CANCELED | FAILED | 1 FINISHED | NOT_USED

First | Previous | 1 | Next | Last

File ID	File State	File Size	Throughput	Remaining
7044611	FINISHED	95.37 MiB	68.51 MiB/s	-

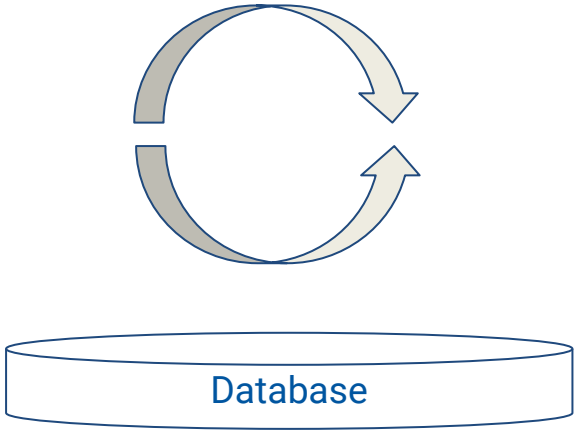
https://eospublic.cern.ch/eos/opstest/dteam/file.100mb

https://eospublic.cern.ch/eos/opstest/dteam/file.100mb_175ee244-8200-4fcb-b8e3-0637897fe576

First | Previous | 1 | Next | Last

- The transfer will use provided access tokens
- Transfer report is sent to Monitoring system

4. Behind-the-scenes housekeeping



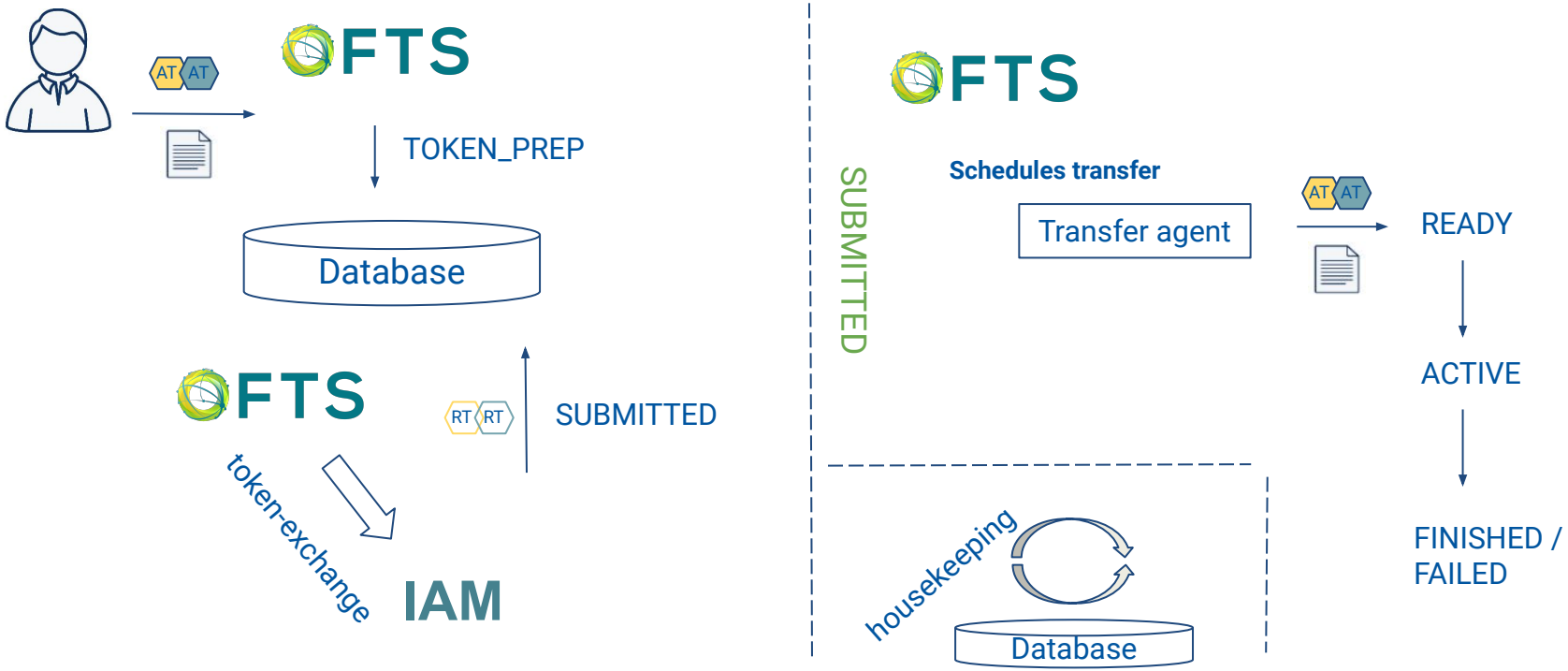
Token Refresher daemon

Scan the DB regularly and refresh tokens close to expiry

Token Housekeeper daemon

Scan the DB regularly and remove unused token entries

FTS Token Lifecycle management (simplified)



Just-in-time refresh?

Replace the TokenRefresher daemon with just-in-time refresh

- Just-in-time: refresh in the transfer agent, right before using the token
- Would ensure FTS refreshes **only the tokens it needs**
- Last piece missing to call token development “complete”

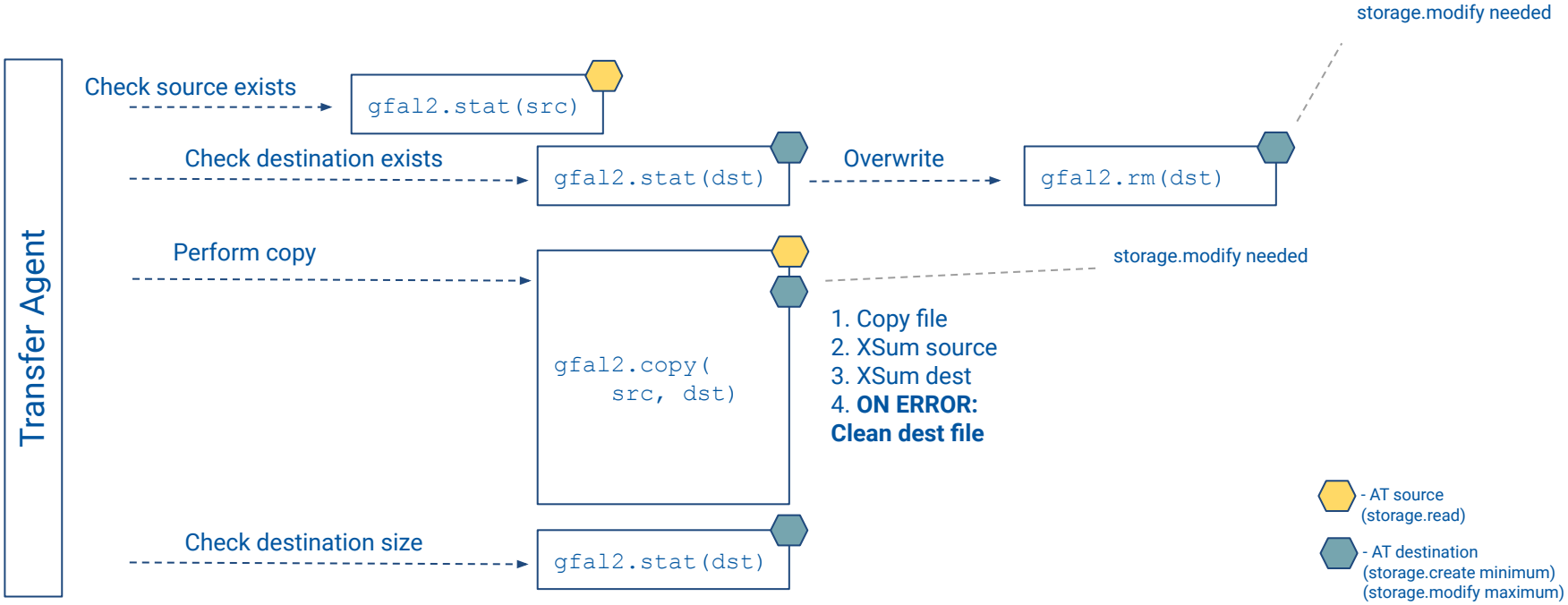
Current reliance on Gfal2 prevents this (too many operations are hidden under `gfal2.copy(..)`)

→ Break-down the large `gfal2.copy(..)` into individual sub-calls [thanks Louis!]

→ Paves the way for FTS to decouple from Gfal2

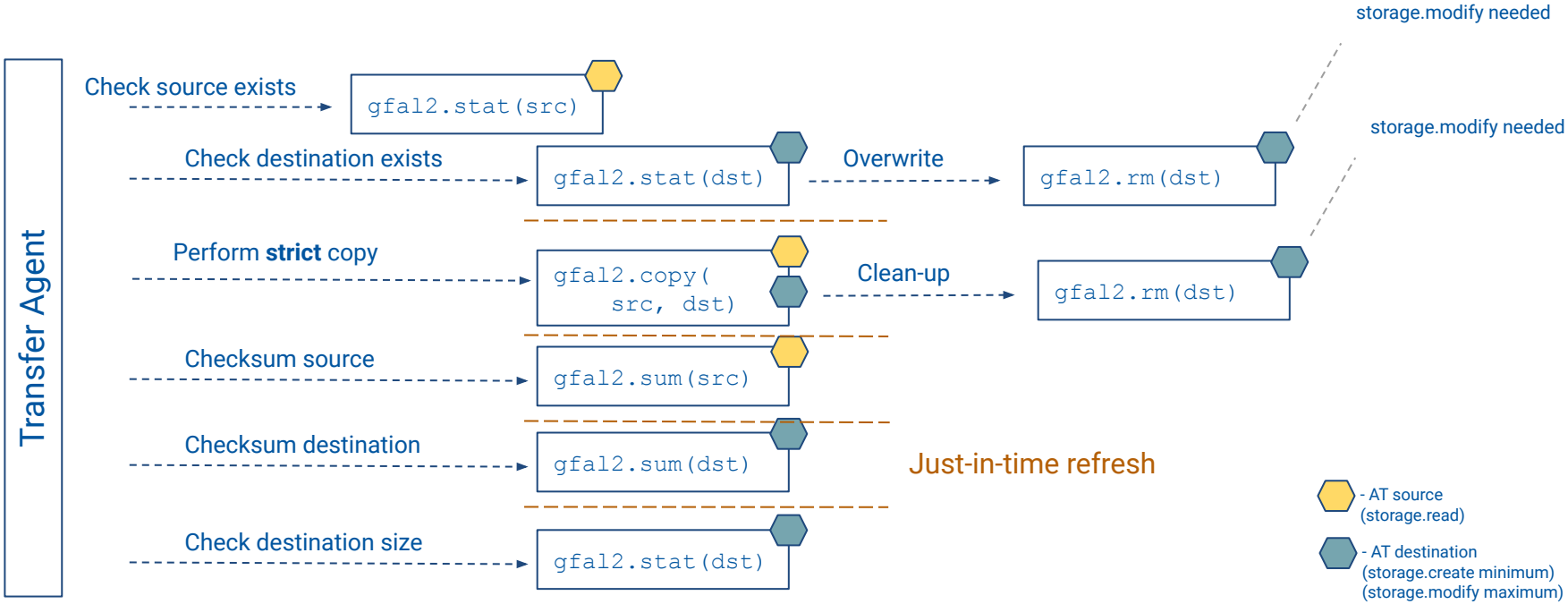
→ Allows FTS transfer agent to refresh token before any SE contact

Anatomy of the transfer agent



Anatomy of the transfer agent

(just-in-time refresh-friendly)



(August 2024)

Token Testing

ATLAS Token Testing

- Long-lived (9 days) per-file (`storage.modify:<full-path>`) tokens
- FTS will not manage the lifecycle for these tokens (lacking `offline_access` scope)
  - FTS won't request a refresh token (token-exchange), nor try to refresh them
- Details in Dimitrios' presentation

- Small amount of development needed to skip token lifecycle management
- No operational concerns or effects on production traffic (according to FTS monitoring)
- Sporadic measurements showed up to 600k access tokens in the FTS database

(→need for constant monitoring of # of tokens in database)

CMS Token Testing

- Pragmatic combination of audiences + scopes per dataset
- FTS will perform full token lifecycle management
- Details in Rahul's presentation

- Slightly more involved in debugging certain transfer failures
  → uncovered certain shortcomings once tokens are considered "no longer used"
  → would only address with the just-in-time refresh
- Large token submission "incident" (28k tokens for exchange vs average of ~20-50 / min)
  → behaved surprisingly well, but only ran at 10Hz (50 threads on FTS side) / ~45m
- Sporadic measurements showed up to 100k tokens in the database

DataChallenge’24 Reflections

Not another

~~DataChallenge'24 Reflections~~

(Experience backed)

Token Reflections

- Tokens are more secure
  - Tokens will leak, no matter what *(FTS blamed in the past for EGI-SVG-2024-02)*
  - Are we ready to mitigate this? (i.e.: how wide should the scope be?)

- Tokens will simplify things
  - ATLAS & CMS take different approaches
  - FTS had to integrate with: INDIGO-IAM, EGI CheckIn, CILogon
  - No guarantee a future TokenProvider will work out of the box *(in fact, not likely)*

- Tokens are an industry standard
  - Perhaps, but is refreshing the right way to go?

    (operational experience shows refresh tokens are difficult to deal with)

➢ Tokens are flexible

- Too much room also for interpretation *(why can't we have a VO field?)*
- Profile definitions can change, but software cannot as easily

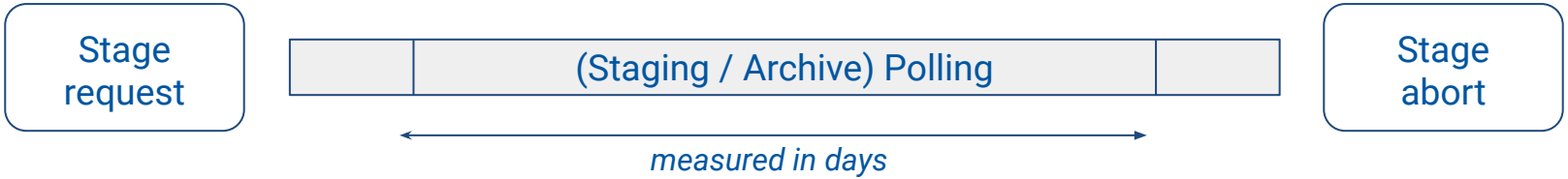
---

- As FTS, not (yet) convinced refreshing was the right approach
- Likely headed to a 2-tier system:
  - Small users → use FTS refresh + token lifecycle management
  - Large users → move with a different approach *(e.g.: Rucio for ATLAS and/or CMS)*

(Avoided any explicit mention of IAM)

Tokens + Tape?

- Tokens + Tape discussion still not started
- Tape interaction presents 3 distinct actions:



- Staging + stage abort will require FTS to manage the token lifecycle
- Is it worth it to refresh for the lifespan of the Polling operation?
  - Could FTS be given by the tape storage a “polling token” *(e.g.: API token)*?
- FTS creates staging batches based on (SE, credID) pairs
  - How will this look like with tokens / scopes involved?

Thank you!

Backup

FTS Token Lifecycle management