FTS support for tapes and tokens

4th EOS workshop

Edward Karavakis and Carles Garcia Cabot
on behalf of the FTS team

FTS

- Distributes majority of LHC data across WLCG infrastructure
- 7 WLCG and 13 non-WLCG instances
- ~25 Virtual Organisations
  - ATLAS, CMS, LHCb, AMS, NA62, Compass, ILC, Magic, Belle II , Mice, Xenon, Snoplus, Gridpp, Dune, LZ, Solidexperiment.org, SKA, Ligo, Icecube, Elixir, NP02, CAST, ESCAPE, Eiscat.se, Virgo
- Integrated with experiment frameworks: Rucio, PhEDEx, DIRAC
- Transferred in 2019 more than 800 million files and 0.95 Exabyte of data

FTS Core Features

Simplicity

- Easy user interaction for submitting transfers. Copy one file from one place to another
- WebFTS portal for end-users, Real Time monitoring and Web Admin

Reliability & Integrity

- Checksums and retries are provided per transfer

Flexibility & Scalability

- Multiprotocol support (HTTP, gsiftp, xrootd, SRM, S3,..)
- Different clients to access the service (REST APIs, python bindings)
- Transfers from/to different storages
- Support for bringonline
- FTS can be run "zero config"

Intelligence

- Parallel transfers scheduling and optimisation to get the most from network without burning the storages
- Priorities/Activities support for transfers classification

Integration for tapes

CERN Tape Archive

- CTA is the new tape based solution at CERN that is built on top of EOS exposing an XRootD interface and supporting TPC
  See M.Davis's talk here
  - EOS+CTA integration is production ready – interface to FTS doesn't change for the experiments, everything handled transparently by FTS
    - Implemented staging via XRootD
    - Disk copy eviction on transfer completion, to better handle the reduced buffer size
    - Staging+Multihop supported (to handle data export to T1s)
  - Stress-tested during the ATLAS Data Carousel exercise

    See X.Zhao's talk here

Monitoring the migration to tape

CERN Tape Archive

- When transferring to a tape-backed system, the tape migration is not taken into account by FTS
  - Transfer is successfully completed at the storage system disk buffer level only
  - Clients have to explicitly check on the destination storage if the file has been correctly migrated to tape
- New feature is being implemented to report a transfer as completed only when the file has been migrated to tape successfully
  - Included a new “ARCHIVING” state in the state machine
  - Clients need to enable this feature when submitting the transfer
  - Implemented for both XRootD and SRM (for any other than CTA)
- Buffer-aware scheduling in the future?

Tokens

- Authentication: WLCG is moving from X.509 certificates to HTTP tokens
- FTS 3.10 supports OpenID Connect
- Pre-release server and client RPM packages available
  - http://fts-repo.web.cern.ch/fts-repo/xdc/el7/x86_64/
  - Get fts-rest-cli-3.10.0 to use the client
  - Example next

Token example

- Get an access token and submit a job
  - https://github.com/indigo-dc/oidc-agent

```
$ export tok=`oidc-token wlcg`
```

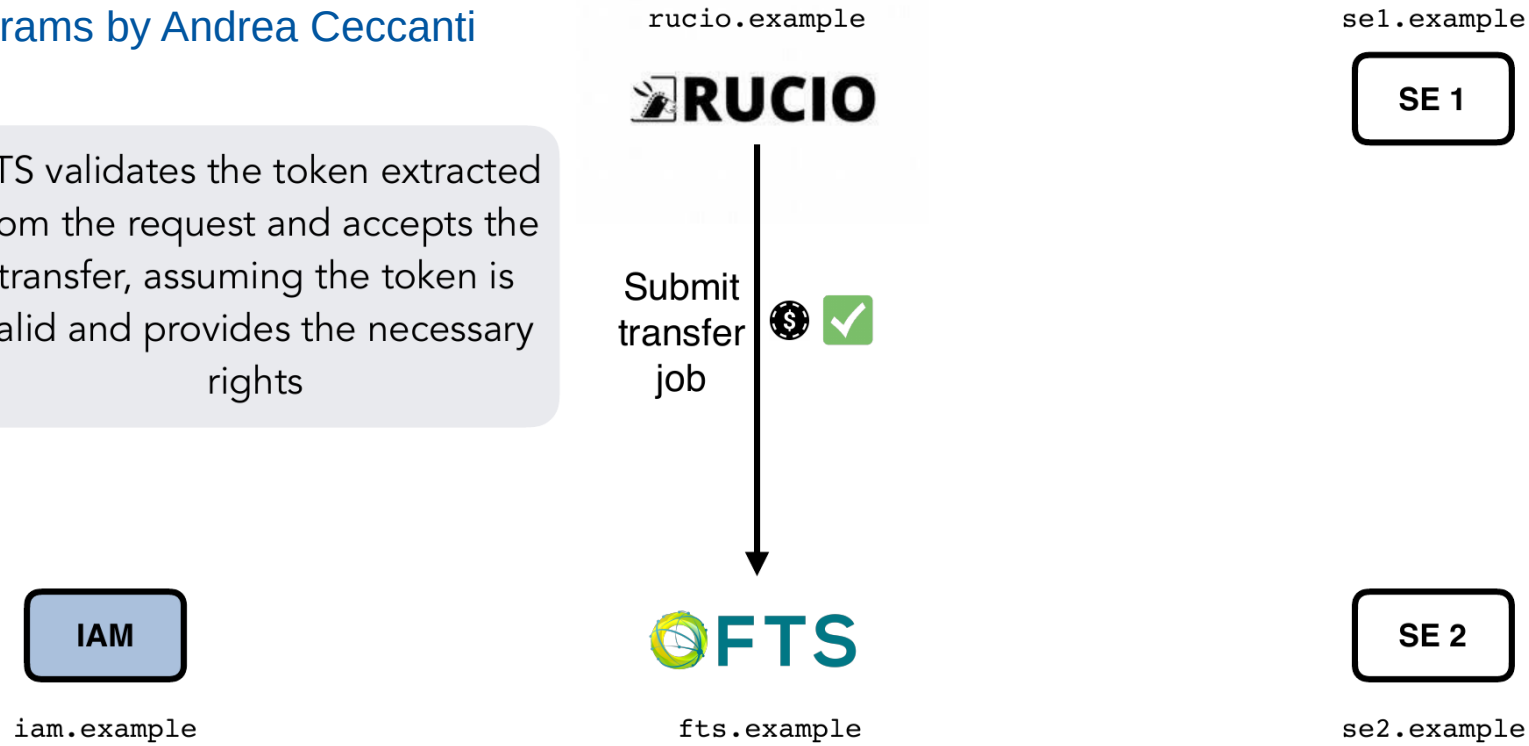
```
$ fts-rest-transfer-submit \
-s https://fts3-xdc.cern.ch:8446 \
--access-token $tok \
https://prometheus.desy.de/Users/carles/source \
https://prometheus.desy.de/Users/carles/destination

Job successfully submitted.
Job id: 2fb22734-360f-11ea-9524-fa163e362acc
```

Token workflow

Diagrams by Andrea Ceccanti

FTS validates the token extracted from the request and accepts the transfer, assuming the token is valid and provides the necessary rights



Token workflow

rucio.example

RUCIO

se1.example

SE 1

FTS then **exchanges the obtained token** with a couple of tokens, an access token and refresh token, that will be used to manage the transfer

IAM

iam.example

FTS

fts.example

SE 2

se2.example

Token workflow

FTS requests the following scopes:
storage.read:/
storage.create:/
offline_access

rucio.example

RUCIO

se1.example

SE 1

```
POST /token HTTP/2
Host: iam.example
Authorization: Basic u89…
Accept: */*
Content-Length: ...
Content-Type: application/x-www-form-urlencoded

grant_type=urn:ietf:params:oauth:grant-type:token-exchange
&subject_token=eyJra…HvBfTpM
&audience=se1.example%20se2.example
&scope=storage.read%3A%2F%20storage.create%3A%2F%20offline_access
```

Token exchange request

IAM

iam.example

FTS

fts.example

SE 2

se2.example

Token workflow

rucio.example

RUCIO

se1.example

SE 1

```
{
  "access_token": “e7nd…HvBfTpM”,
  “refresh_token": “9njuk…”,
  "token_type": "Bearer",
  "expires_in": 3599,
  "scope": “storage.read:/ storage.create:/ offline_access’
}
```

Token exchange response

IAM

iam.example

FTS

fts.example

SE 2

se2.example

Token workflow

FTS then submits the third-party transfer against SE 2, including the token in the request

rucio.example

RUCIO

se1.example

SE 1

```
COPY /example/file HTTP/2
Host: se2.example
Source: https://se1.example/example/file
Authorization: Bearer e7nd…
TransferHeaderAuthorization: Bearer e7nd…
```

Submit
third-party transfer

IAM

iam.example

FTS

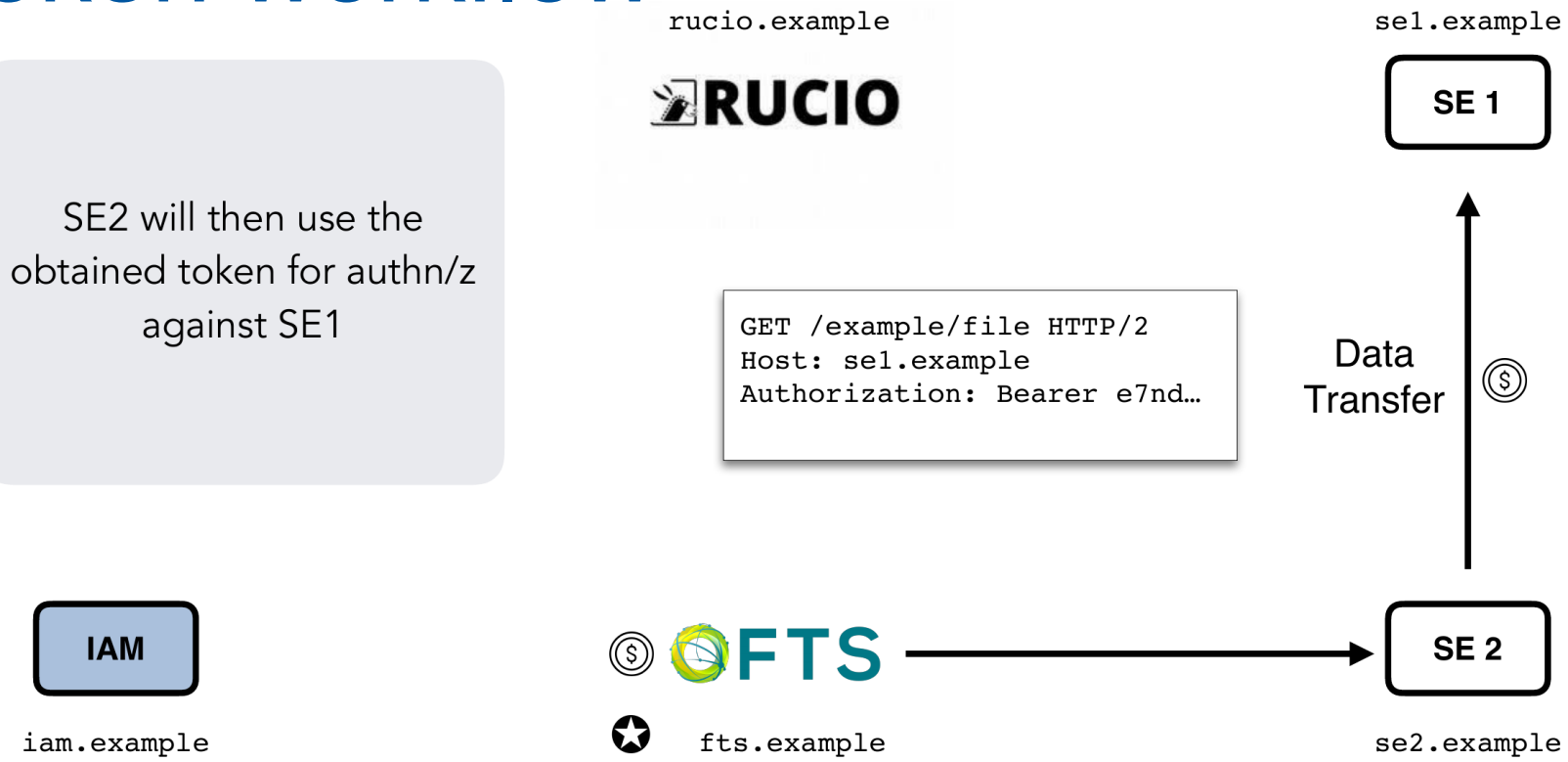
fts.example

SE 2

se2.example

Token workflow

SE2 will then use the obtained token for authn/z against SE1



Conclusion

- FTS continues to evolve with the infrastructure as WLCG's principal data movement service
- Full support planned for token auth
- Integration with CERN's new tape archival system CTA