TReqS : Tape Request Scheduler

Bernard Chambon, CC-IN2P3

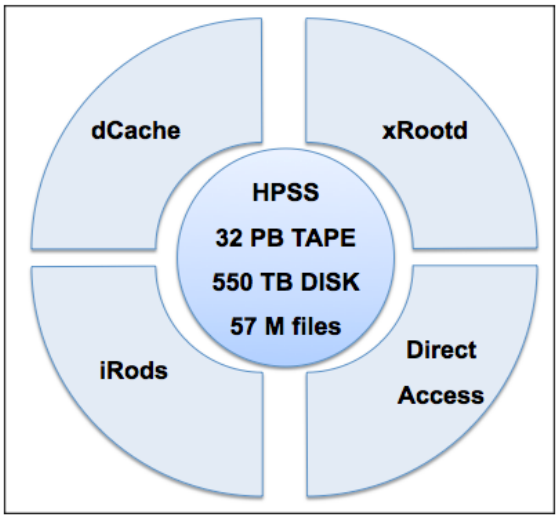
HEPIX meeting, Zeuthen
18 - 22 April 2016

Outline

- Mass storage at CC-IN2P3
- TReqS motivation
- TReqS-2 overview
  - Features
  - Architecture
  - Development process
- TReqS-2 status

Mass storage at CC-IN2P3

- Schema



All access via RFIO using CLI tools (rfdir, rfcp)
(from CERN RFIO built using HPSS API, maintained at CC-IN2P3)

- Numbers
  HPSS average [1] access per day :
  2000 tape mounts : 50 K files accessed (16 K for reading)
  16 K for reading : 10 K staged by TReqS-1 [2] (HPSS staging = tape → disk)

1. Peek access up to 10000 tape mounts. +8 PB tape storage planned for 2016
2. TReqS-1 : Old version of TReqS currently on production

TReqS motivation

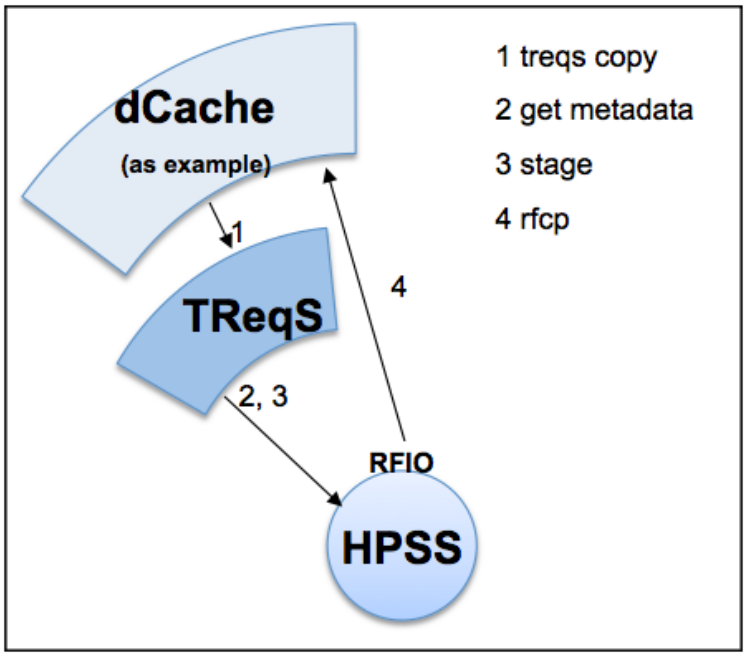
- Motivation for TReqS
  Regulation
    No limitation in amount of requests sent to HPSS
    No limitation in amount of tape drives used for reading (and HPSS needs drives for writing)
    HPSS requests are processed in FIFO mode
  Optimization
    Possibly several mounts of same tape for several requests [2]
    Possibly several forward | rewind of ribbon to read files
- Motivation for TReqS-2
  A TReqS-1 running for 6 years, but ...
  Old architecture, not fully reliable, without any (possible) evolution
  $\Rightarrow$ new architecture and implementation started at fall 2015

Main objective : Improve staging

2. Although HPSS is configured to keep tape in drive up to 20 mn

TReqS-2 overview

- Positioning (example through dCache)



- Main guidelines
  Scalable and reliable server : tape usage increasing (e.g. +20% (+8PB) planned for 2016)
  Configurable and portable software (nothing specific to CC-IN2P3)
  Open source
  Client / server model

TReqS-2 Features

- Regulation (to provide control upstream to HPSS)
  - Limit number of allocated drives per drive-model (e.g. 25 drives for T10K-D tape model).
  - Restrict access to granted users only
  - Fair share of drives among users (planned)
  - High priority request (planned)
- Optimization (to decrease number of tape movements)
  - Aggregate requests over time, per tape, to reduce number of tape mounts
  - Possibly add files to be staged while a tape is currently being read
  - Sort files to be staged in most efficient way, currently FPOT [1]
- From client point of view
  - Provide copy mode (= staging + transfer command (e.g. rfcp))
  - Provide staging only mode, possibly in bulk mode
  - API and CLI to submit, query, cancel requests and also for administrative tasks
  - Monitoring (web pages)

1. FPOT: logical File Position On Tape

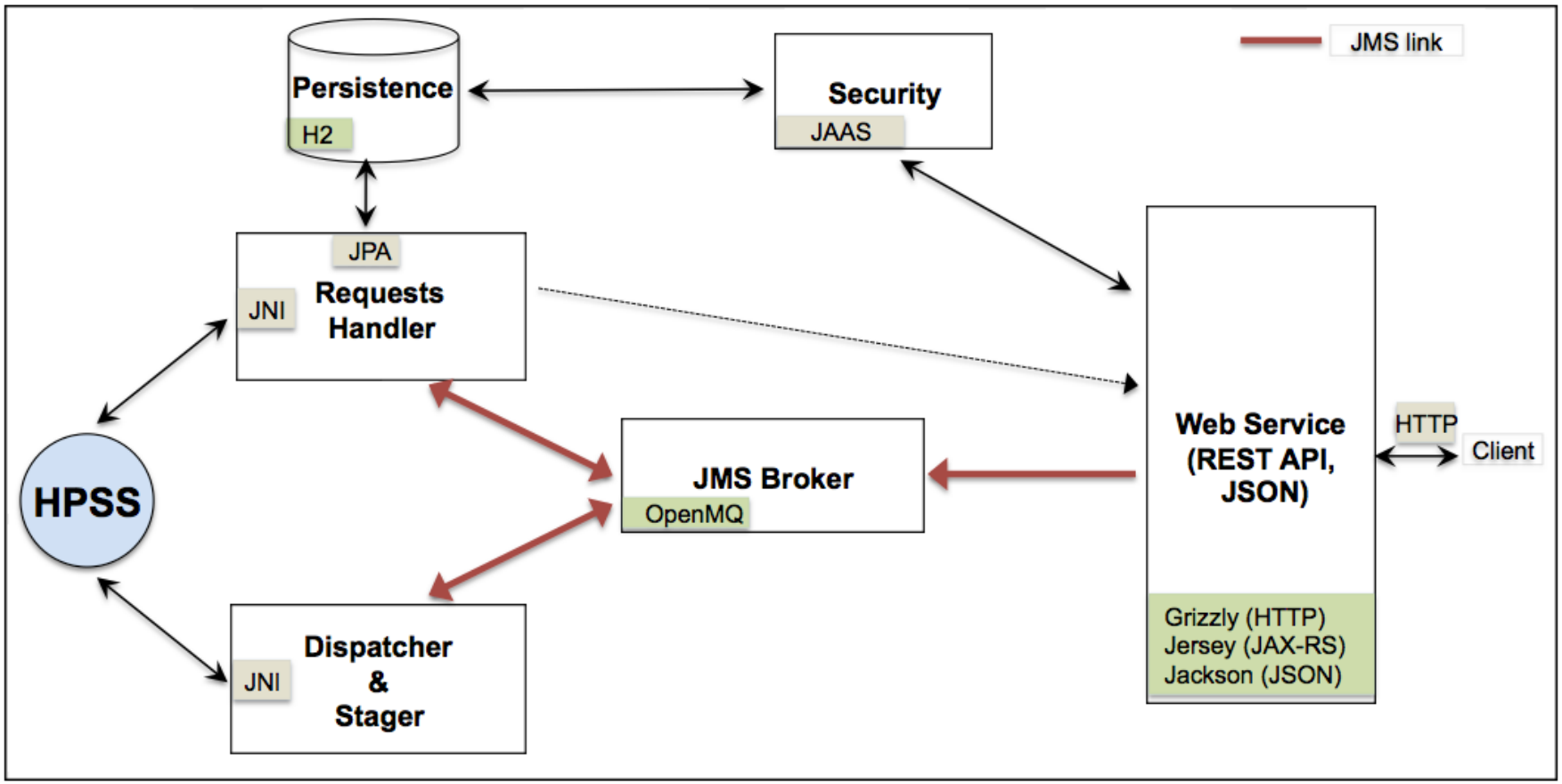
TReqS-2 architecture 1/2

- Server
  Written in java, using the following components

  REST API (http, json) : Lightweight client (CLI or WUI), easy to develop
  JMS [1] : Components with well delimited scope, less shared data structures
  H2 DB as persistence : Fast, embedded (or server), 100% java, freely available
  JAAS [2] : For site customization
  HPSS API via JNI [3]
  Mustache+DataTables for integrated monitoring
- Client
  Written in python
  Using REST API for interaction with server
  Using transfert agent rfcp [4] to copy file from HPSS mover to local area

Main guideline : Standard and Modularity

1. JMS : Java Messaging Service
2. JAAS : Java Authentication & Authorization Service
3. JNI : Java Native Interface, since HPSS provide only a C API
4. rfcp : file transfer command, used at CC-IN2P3, based on CERN RFIO, built locally with HPSS libs


JMS link
Persistence
H2
Security
JAAS
JPA
JNI
Requests Handler
HPSS
JMS Broker
OpenMQ
Web Service (REST API, JSON)
HTTP
Client
Grizzly (HTTP)
Jersey (JAX-RS)
Jackson (JSON)
JNI
Dispatcher & Stager

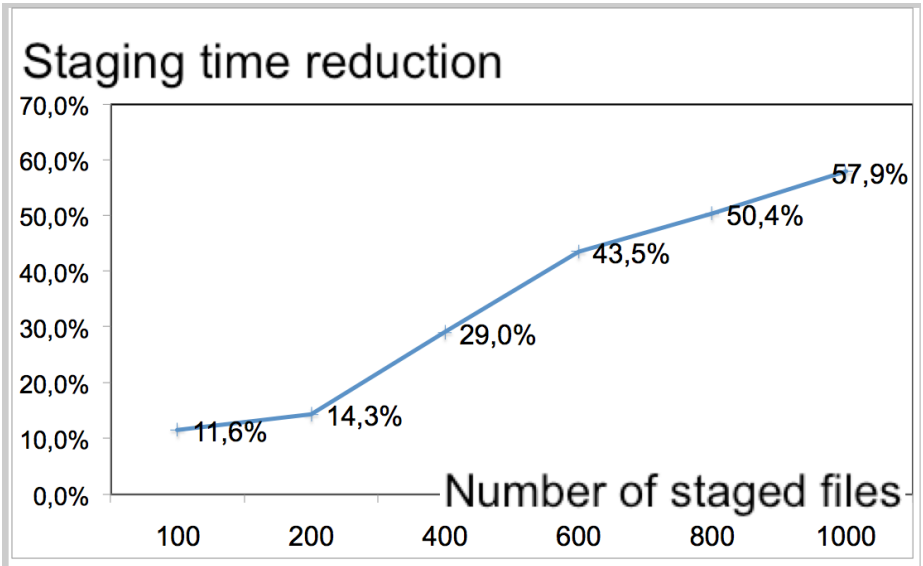
TReqS-2 development process

- Tools
  - Maven for project mgmt
  - Git as code repository (gitlab.in2p3.fr, private access)
  - Jenkins for continuous integration process
  - Sonar for code audit
- Tests methods
  - Unit & integration tests
  - Test set of files to simulate HPSS (querying metadata, staging)
- Metrics
  - Load tests of the server
  - Estimation of the benefit of TReqS (see example in next slide)

Impact of using sorted FPOT (logical File Position On tape)

- Example using a T10K-D filled with 8,089 files of 1GB each

Staging time reduction using **sorted** position vs random position



- Comments
  - o Staging improvement increase with number of staged files
    E.g. staging time reduction of **20%**, if you stage ∼300 files (⇔ 3.7% of total files)
  - o Low improvement if staging small amount of files
    ⇒ triggering staging (=config parameter) must be adapted to production context
  - o From TReqS-1, often seen a few tens of files per queue
    ⇒ for this case, but not only, new TOR [1] feature of HPSS should be investigated

1. TOR : Tape-Ordered Recall, available in HPSS 7.4.3p3 & HPSS 7.5

TReqS-2 status

- Short term schedule
  Alpha version, available since a few weeks, for tests by TReqS admins, with simulated HPSS.
  (∼8,000 lines of code for server)

  Beta version planned for 2016/05
    Open source code, LGPL-V3
    Will be available from gitlab.in2p3.fr, access to granted users (not public)

  Production version planned for 2016/09
- Long term schedule (features not planned for production version of 2016/09)
  1) Fair share between users & priority requests management
  2) TOR investigation
     Requires new release of HPSS (current production instance is HPSS 7.4.2)

Thank you for your attention

On behalf of
Lionel Schwarz (TReqS software developer)
Pierre-Emmanuel Brinette (HPSS & TReqS administrator)

Annexes

- Client commands overview
- Staging demo

Annex : Client commands overview

Table 1 : Client commands

treqs

```
>treqs.py
usage: treqs.py [-h] [-v] [--url SERVICE_URL] commands ...

  commands
    copy              --help
    stage             --help

    show              --help

    cancel            --help

    ping              --help


optional arguments:
  -h, --help          show this help message and exit
  -v, --version       show program's version number and exit
  --url SERVICE_URL   HTTP URL of the remote TReqS service
```

treqsadmin

```
>treqsadmin.py
usage: treqsadmin.py [-h] [-v] [--url SERVICE_URL] commands ...

  commands
    create            --help
    show              --help
    update            --help
    delete            --help

    lock              --help
    unlock            --help

    grant             --help
    deny              --help

    dump              --help


optional arguments:
  -h, --help          show this help message and exit
  -v, --version       show program's version number and exit
  --url SERVICE_URL   HTTP URL of the remote TReqS service
```

Annex : Client commands overview

Table 2 : treqs copy options

```
>treqs.py copy --help
usage: treqs.py copy [-h] -f HPSS -d DEST

Stage a HPSS file then copy it to a local destination. Copy uses the transfer
server set via the HPSS URL or using the one set via TREQS_TRANSFER_SERVER
environment variable or the one set in $HOME/.treqsrc see README fordetails

arguments:
  -h, --help            show this help message and exit
  -f HPSS, --file HPSS  HPSS URL filename only or URL with transfer server
  -d DEST, --dest DEST  Local destination path file or directory
```

Table 3 : treqs stage options

```
>treqs.py stage --help
usage: treqs.py stage [-h] -f HPSS | -l HPSS_LIST

Stage a HPSS file or a list of HPSS files either from a file or from stdin

arguments:
  -h, --help            show this help message and exit
  -f HPSS, --file HPSS  HPSS filename
  -l HPSS_LIST, --list HPSS_LIST
                        Filename containing a list of HPSS files, or '-' to
                        get the list from stdin ctrlD as end of list
```

Annex : Staging demo

Table 4 : Checking server, tape model and user

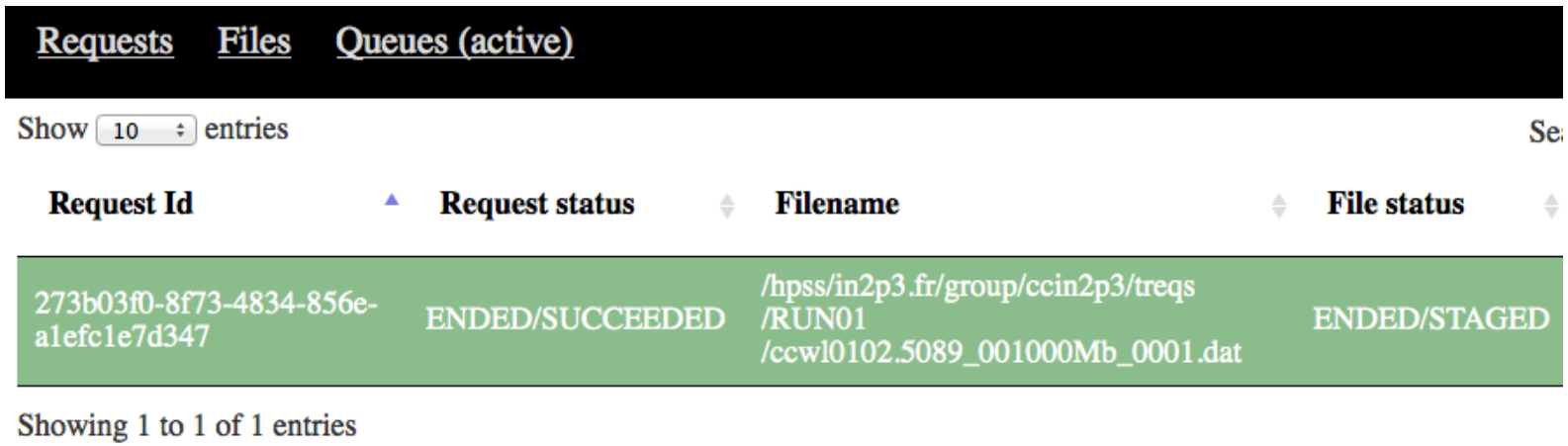
```
>treqs.py ping
status       ok

>treqs.py show tapemodel --model T10K-D
name    status      max_parallel_staging    reading_rate MiB/s
-------------------------------------------------------------
T10K-D  ENABLED     10                      250

>treqs.py show user
username    status
------------------
treqs       ENABLED
```

Table 5 : Running stage request, getting info

```
>treqs.py stage -f /hpss/in2p3.fr/group/ccin2p3/treqs/RUN01/ccwl0102.5089_001000Mb_0001.dat
Request 273b03f0-8f73-4834-856e-a1efc1e7d347 submitted

a few seconds later ...

>treqs.py show request -r 273b03f0-8f73-4834-856e-a1efc1e7d347
request_id                           status    sub_status  submitted_date       username  filename
-------------------------------------------------------------------------------------------------
273b03f0-8f73-4834-856e-a1efc1e7d347 ENDED     SUCCEEDED   2016-04-07T11:59:05Z treqs     /hpss/in2p3.fr/...

>treqs.py show file -f /hpss/in2p3.fr/group/ccin2p3/treqs/RUN01/ccwl0102.5089_001000Mb_0001.dat
filename                                              size B      FPOT tapename  status    sub_status
-------------------------------------------------------------------------------------------------------
hpss/in2p3.fr/.../ccwl0102.5089_001000Mb_0001.dat     1048576000  5221 KT343800  ENDED     STAGED
```

Annex : Monitoring web pages of previous staging

Requests view (partial)



Files view (partial)