Un approccio RESTful per la gestione di file richiamati da tape in StoRM

Federica Agostini
INFN-CNAF

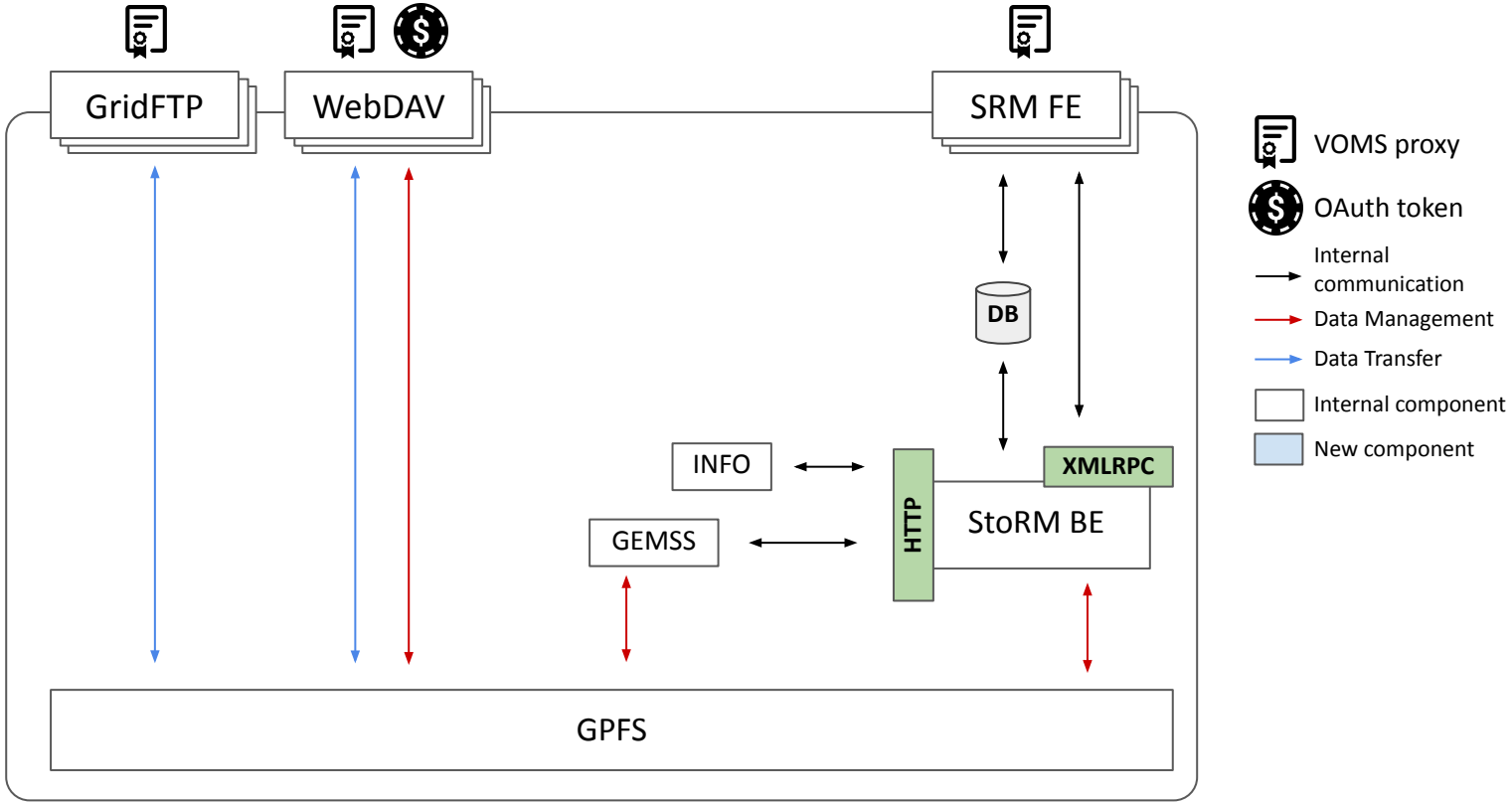
Workshop sul Calcolo nell'INFN - Loano, 22-26 maggio 2023

Outline

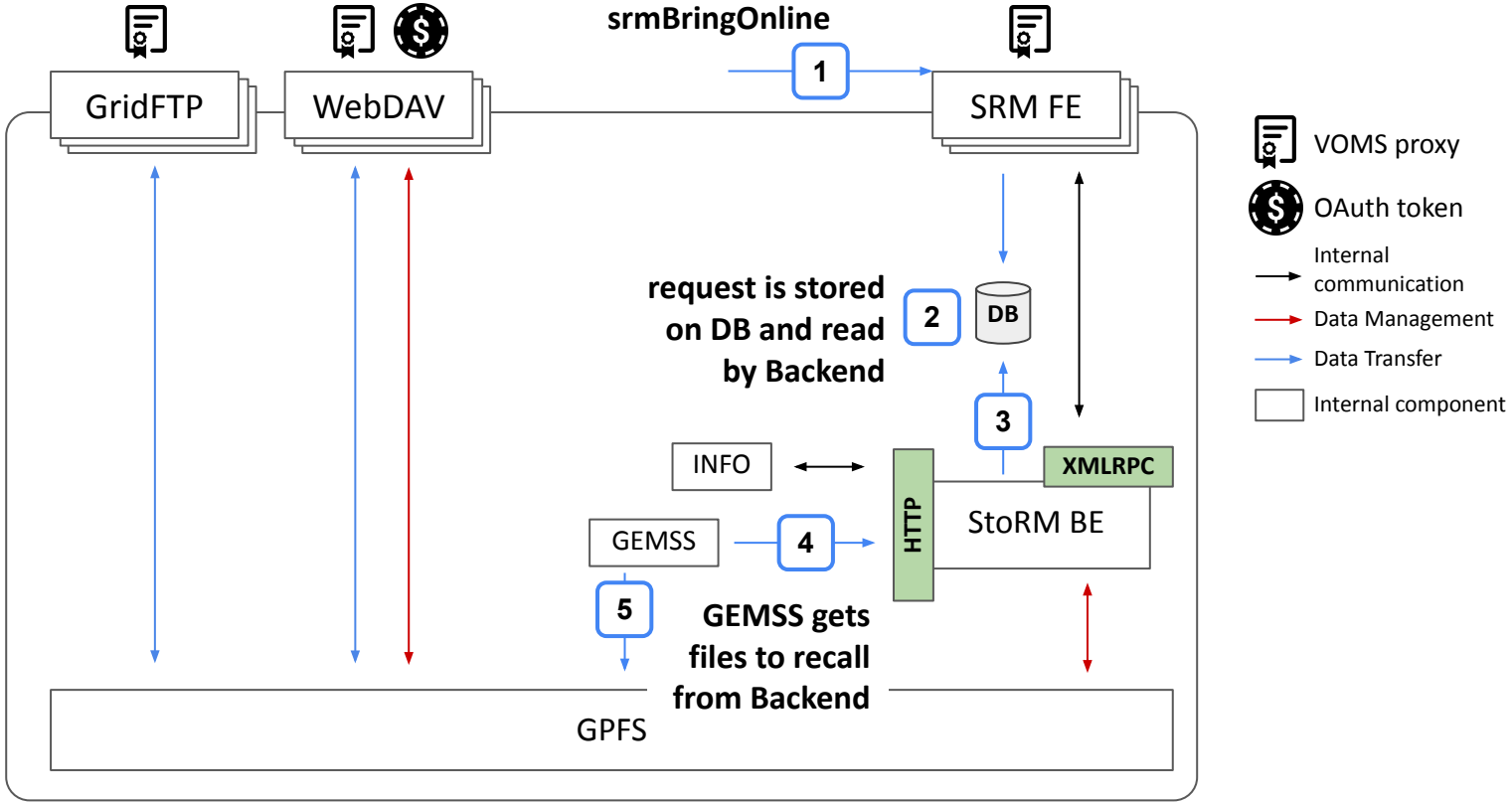
- Current SRM architecture
  - Towards the StoRM Tape REST API deployment scenario
  - The SRM Data Lifecycle
- WLCG Tape REST API: specification & implementation
  - StoRM Tape service
  - NGINX
  - Open Policy Agent
- Testing
- Future looks
  - developments
  - no-SRM scenario

Current SRM architecture

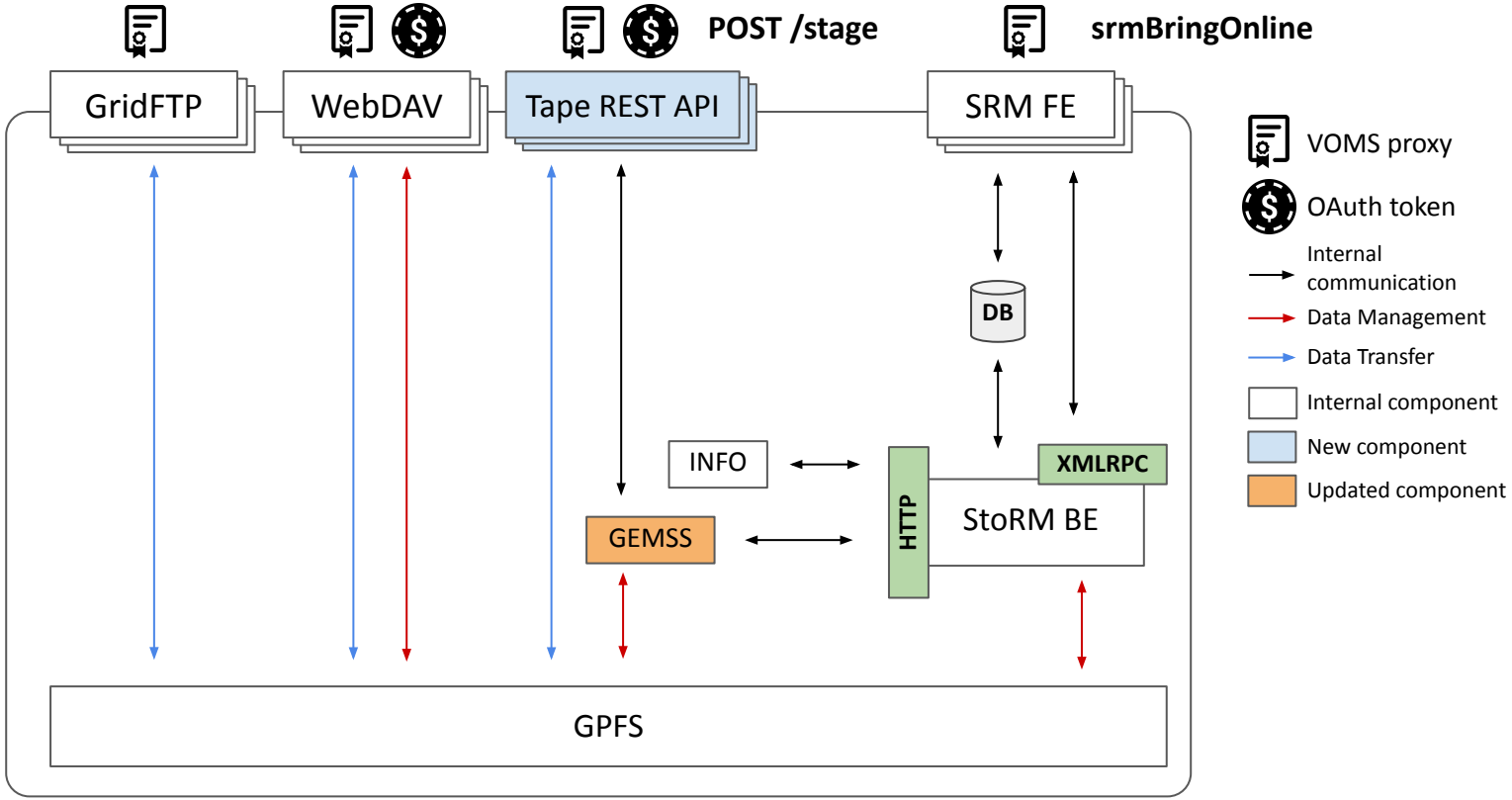
StoRM deployment: current SRM architecture



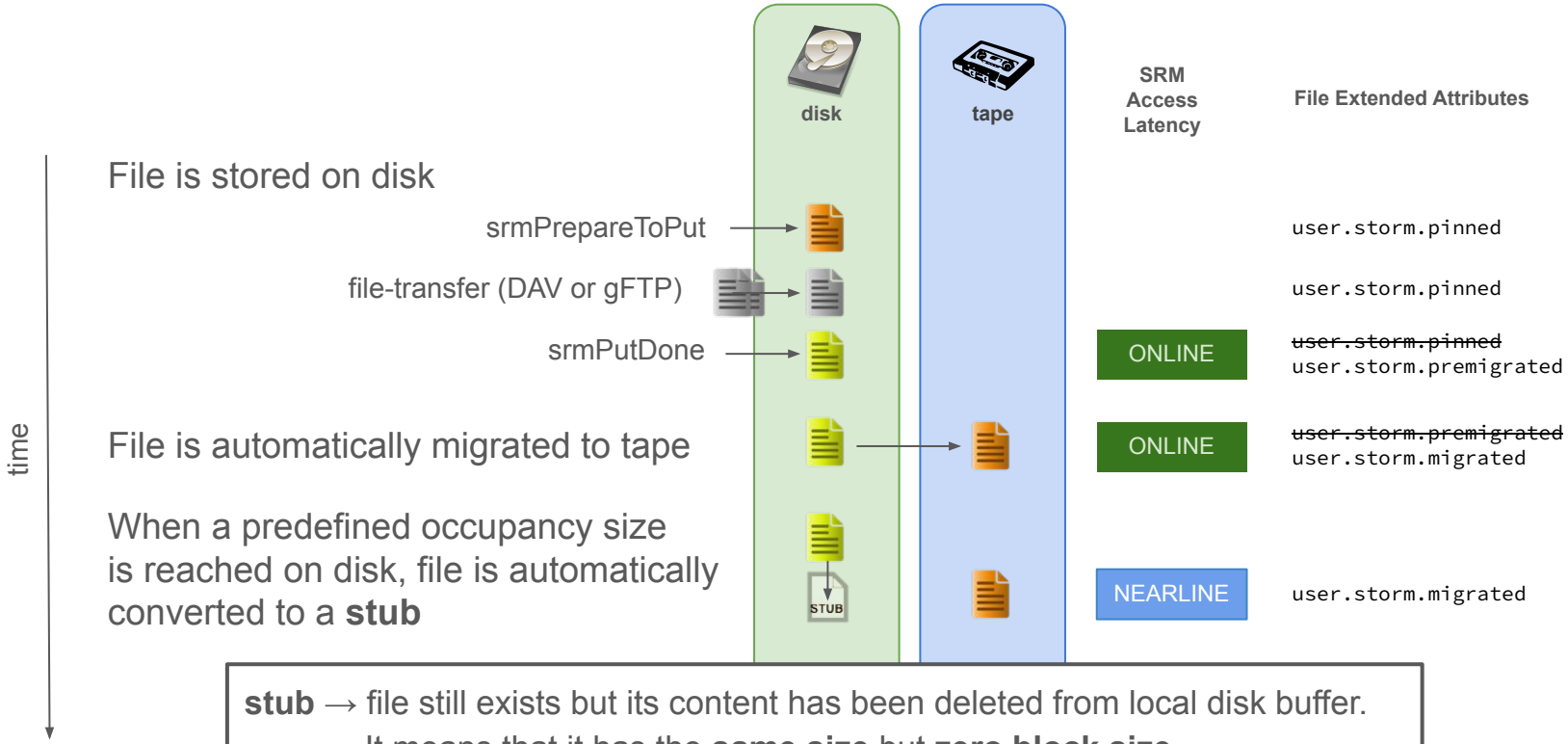
StoRM deployment: current SRM architecture



StoRM deployment: the Tape REST API scenario

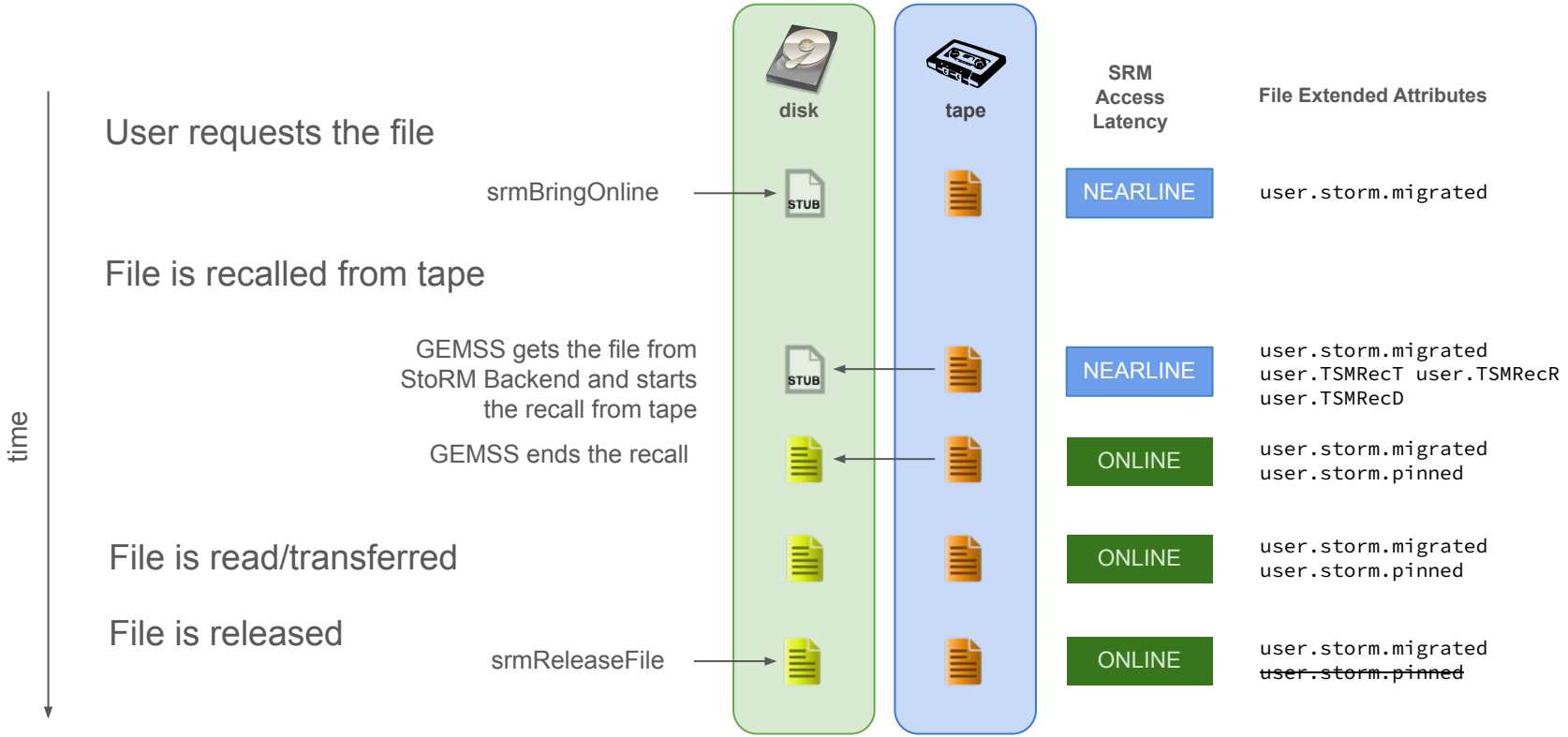


The lifecycle of a migration request with SRM



stub → file still exists but its content has been deleted from local disk buffer.
It means that it has the **same size** but **zero block size**.

The lifecycle of a recall request with SRM



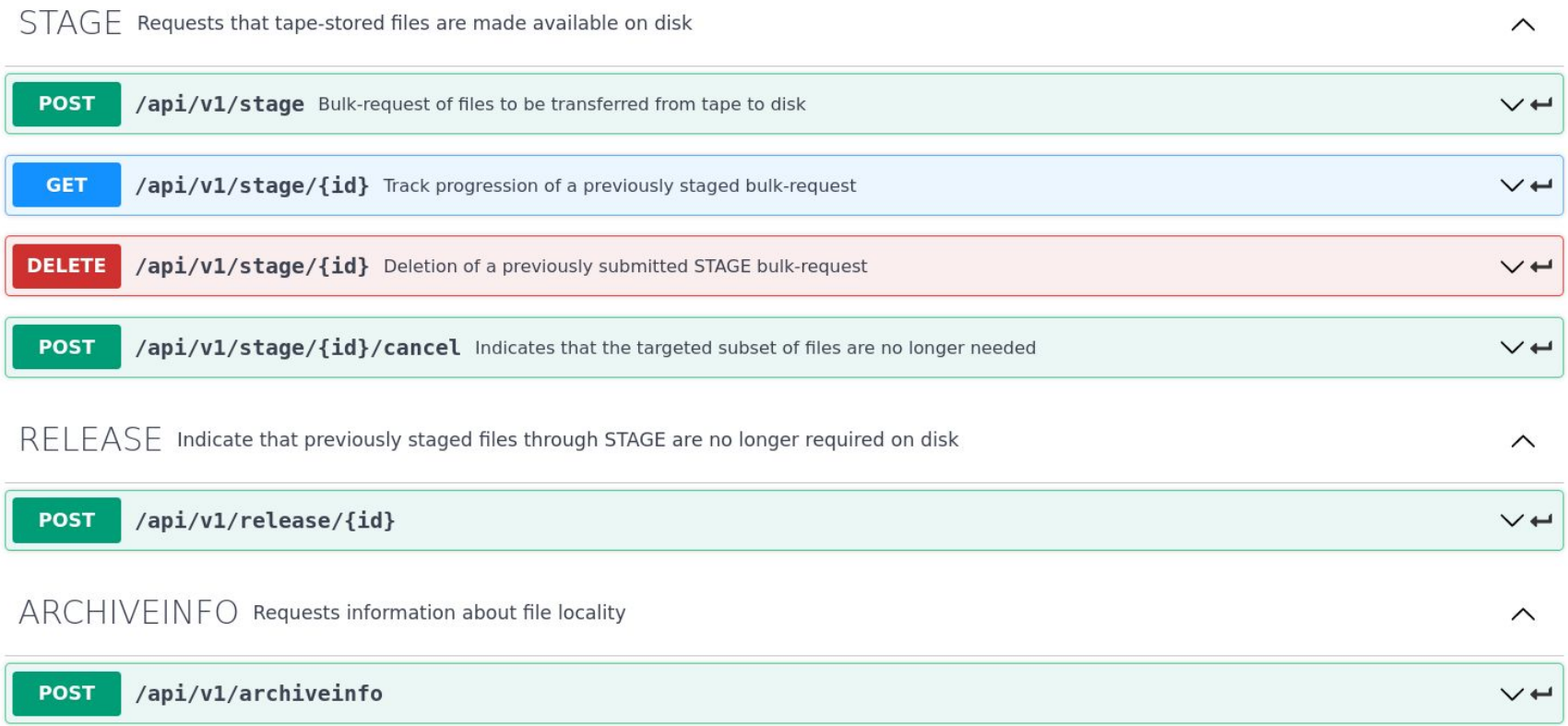
WLCG Tape REST API: specification & StoRM implementation

WLCG Tape REST API project

- New specification defined within the WLCG Tape REST API working group
- A common HTTP interface which allows clients to recall files stored on tape
  - all tape file transfer and management operations can be done with simple HTTP
  - much simpler than the equivalent SRM BringOnLine
- Storage-agnostic API: same protocol used by all storage systems
- The API will be accessed via authentication mechanisms like X.509/VOMS or WLCG JSON Web Tokens (JWT)
- Collaboration between different actors:
  - WLCG storage providers → StoRM, dCache, EOS+CTA
  - main clients → FTS

WLCG Tape REST API specification

Generated with Swagger, source here



StoRM Tape REST API

- The StoRM Tape REST API service (storm-tape) is written in C++, based on the Crow framework and uses SOCI library as abstraction layer over the SQLite database engine
- The service checks file locality directly from the underlying storage system (GPFS). Information on the files are handled using extended attributes:
  - `user.storm.migrated`
  - `user.TSMReCT`
- It provides an additional endpoint for GEMSS to replicate the current interaction with StoRM
  - `GET https://<storm-tape-host>/recalltable/cardinality/tasks/readyTakeOver`
  - `PUT https://<storm-tape-host>/recalltable/tasks`
- Deployed as a standalone component (not within StoRM WebDAV)
- Packaged as a Docker image or as RPM
- AuthN/Z is handled by external services (see later)

Ready as a preview!

The lifecycle of a recall request with HTTP

time

User requests the file

`POST /stage`
(returns a {request-id})

File is recalled from tape

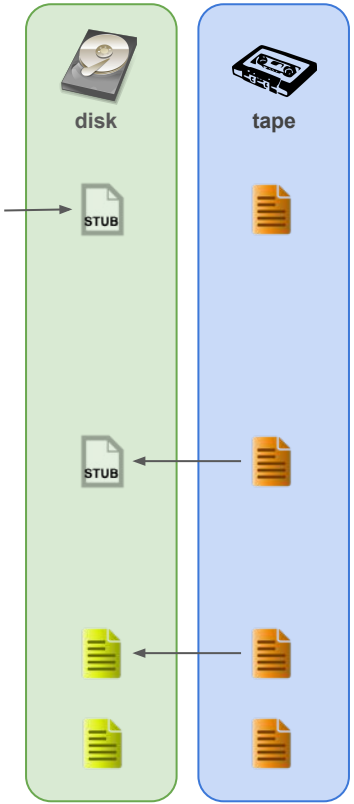
GEMSS gets the list of the requested files from storm-tape with a `PUT /recalltable`

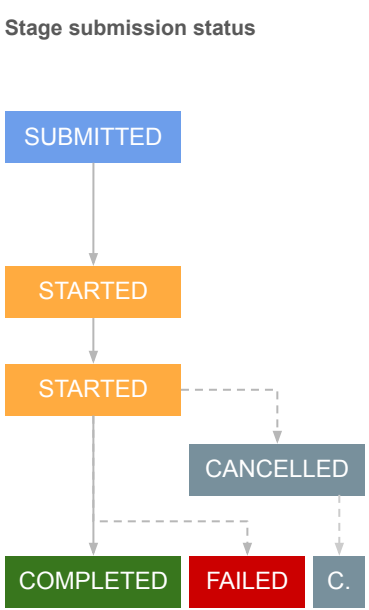
GEMSS starts the recall

(optional) User cancels the stage request
`POST /stage/{request-id}/cancel`

GEMSS ends the recall

File is read/transferred



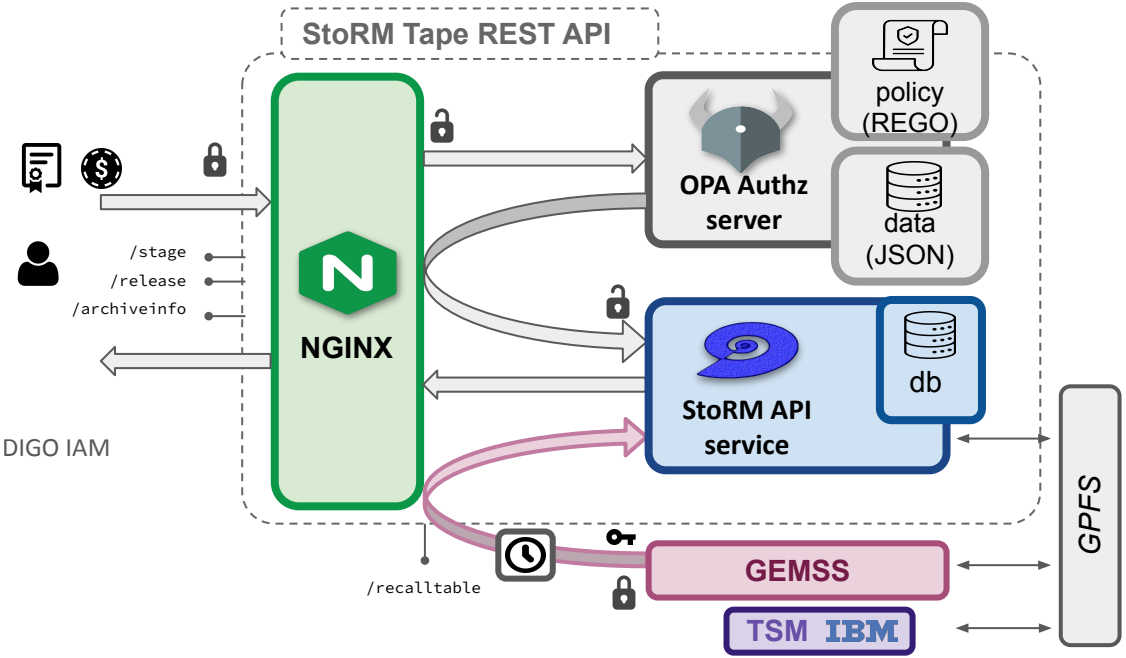


File Properties (bs = block size)	File Extended Attributes
blocks * bs < size	`user.storm.migrated`
blocks * bs < size	`user.storm.migrated` `user.TSMRecT`
blocks * bs < size	`user.storm.migrated` `user.TSMRecT`
blocks * bs < size	`user.storm.migrated` `user.TSMRecT`
blocks * bs ≥ size	`user.storm.migrated`

StoRM Tape REST API: deployment

The StoRM Tape REST API relies on external components for authN/Z

- NGINX → authentication
- OPA → authorization



JWT token issued by INDIGO IAM
VOMS proxy
Basic authentication
HTTPs communication
HTTP communication

From CHEP 2023 poster session

NGINX role in the StoRM Tape deployment

- NGINX is an open-source HTTP server and reverse proxy known for
  - high performance
  - high stability
  - rich feature set
  - simple configuration
  - low resource consumption
- The service has been chosen as part of this deployment for
  - TLS termination
  - Authentication with JWT
  - Authentication with VOMS/X509

An **auth_engine.js** module (written at CNAF) is used to

- check the presence of a JWT in the HTTP Header and, in case, validate it
- check the presence of X.509/VOMS variables (`voms_fqans, ssl_client_s_dn`)
- pass the above data to OPA and handle its response

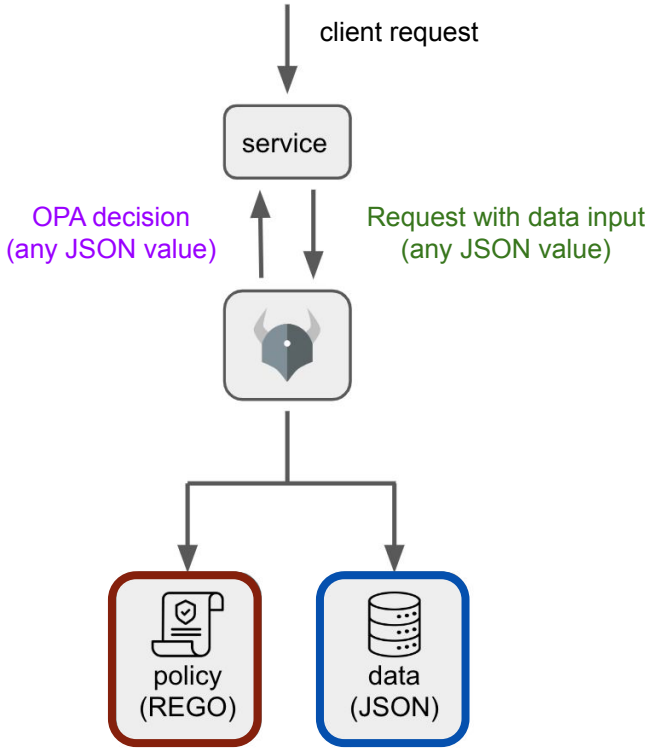
NGINX+VOMS role in the StoRM Tape deployment

- ngx_http_voms_module is a module for NGINX which
  - enables client-side authentication based on X.509 proxy certificates
  - developed at INFN-CNAF
- it defines a set of embedded variables whose values are extracted from the Attribute Certificate
  - *e.g.* the **`voms_fqans`**

```
subject   : /DC=org/DC=terena/DC
issuer    : /DC=org/DC=terena/
identity  : /DC=org/DC=terena/DC
type      : RFC3820 compliant
strength  : 2048
path      : /tmp/x509up_u1000
timeleft  : 00:59:35
key usage : Digital Signature, K
=== VO wlcg extension informatic
VO        : wlcg
subject   : /DC=org/DC=terena/DC
issuer    : /DC=org/DC=terena/
attribute : /wlcg
attribute : /wlcg/mc
attribute : /wlcg/pilots
attribute : /wlcg/xfers
timeleft  : 11:59:53
uri       : wlcg-voms.cloud.cr
```

OPA role in the StoRM Tape deployment

- Open Policy Agent (OPA) is an open-source authorization engine that
  - unifies policy enforcement across the stack
  - is based on an high-level declarative language
  - allows the definition of policies as code
- Deployed and tested at INFN-CNAF for **authorization** with X509/VOMS or JWT
- It seems flexible enough to replace other authorization engines
  - *e.g.* Argus



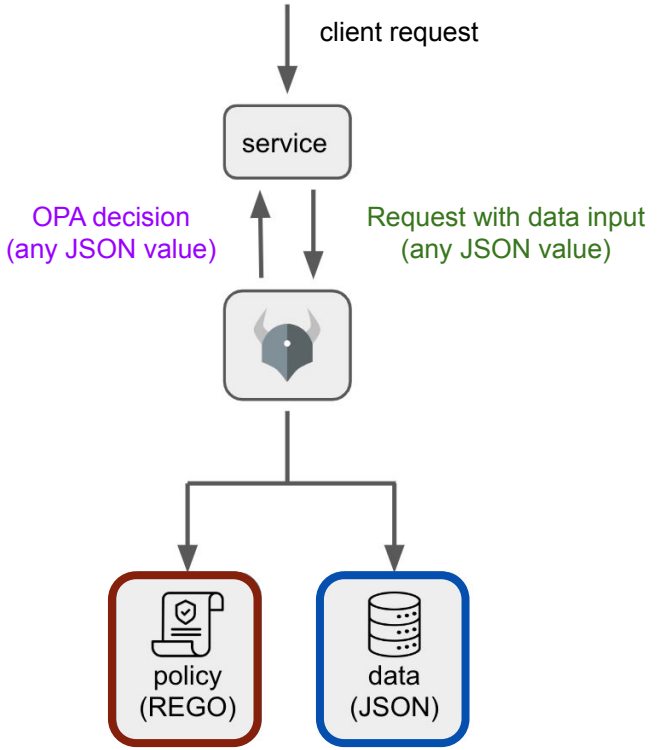
OPA role in the StoRM Tape deployment: example

```
{
  "method": "GET",
  "path": "/api/v1/stage/9a8e34bd-73fe-4b43-9139-1c5f6711577c",
  "client_s_dn": "CN=test0,O=IGI,C=IT"
}
```

```
{
  "allowed_dn": [
  "CN=John Doe jhondoe@infn.it,O=Istituto Nazionale di Fisica
Nucleare,C=IT,DC=tcs,DC=terena,DC=org",
  "CN=test0,O=IGI,C=IT"
  ],
  …
}
```

```
# GET /api/v1/stage/<id>
allow if {
    input.method == "GET"
    glob.match("/api/v1/stage/*", ["/"], input.path)

    any([read_scopes_allowed, voms_fqans_allowed, certificate_dn_allowed])
}
```

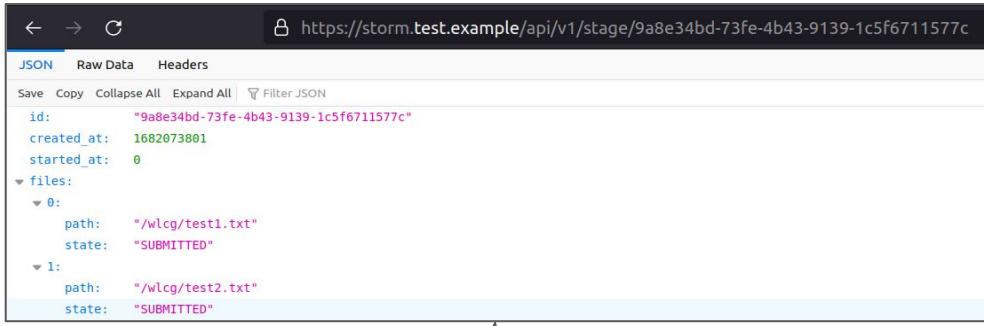
has allowed WLCG scopes? OR has allowed FQANs? OR has allowed DN?



OPA role in the StoRM Tape deployment: example

```
{
  "method": "GET",
  "path": "/api/v1/stage/9a8e34bd-73fe-4b43-9139-1c5f6711577c",
  "client_s_dn": "CN=test0,O=IGI,C=IT"
}
```

```
{
  "allowed_dn": [
  "CN=John Doe jhondoe@infn.it,O=Istituto Nazionale di Fisica
Nucleare,C=IT,DC=tcs,DC=terena,DC=org",
  "CN=test0,O=IGI,C=IT"
  ],
  …
}
```



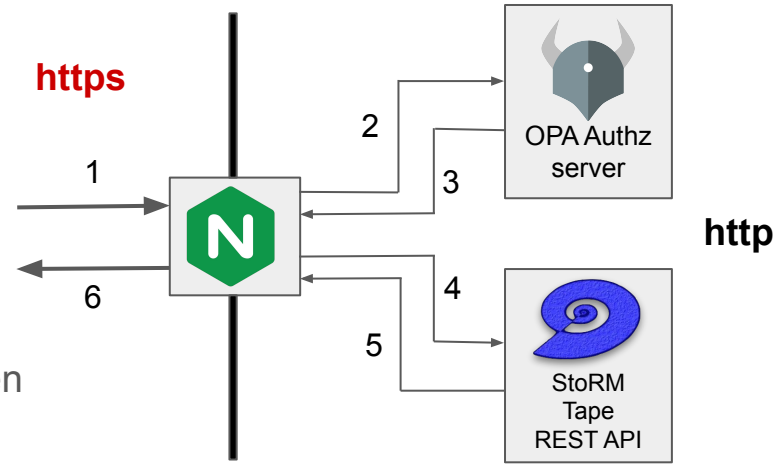

```
# GET /api/v1/stage/<id>
allow if {
    input.method == "GET"
    glob.match("/api/v1/stage/*", ["/"], input.path)

    any([read_scopes_allowed, voms_fqans_allowed, certificate_dn_allowed])
}
```

```
{
  “allow”: “true”
}
```

has allowed WLCG scopes? ✗ OR has allowed FQANs? ✗ OR has allowed DN? ✓

NGINX + OPA AuthN/Z

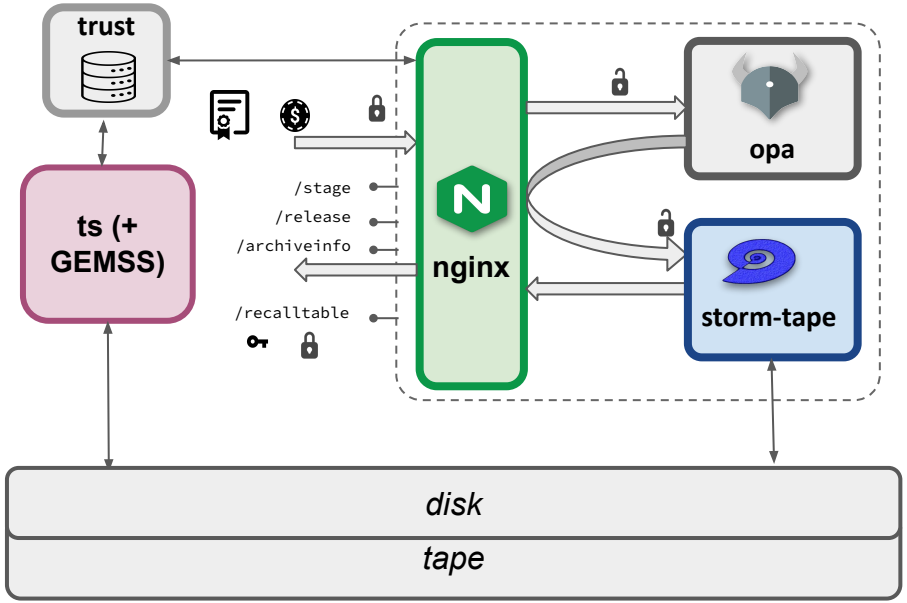
1. The client submits an API request, which is VOMS/TLS terminated by NGINX
2. NGINX sends the request to the OPA engine
3. OPA makes the authZ decision using its policies and data and sends it back to NGINX
    - in case of negative authZ, it returns 403 Forbidden
4. In case of successful authZ, the request is forwarded to the StoRM Tape REST API service
5. (and 6.) The response from the service service is relayed to the client via NGINX



Testing

Deployment tests with Docker

Deployment tests based on docker-compose collects all the necessary services




```
SERVICE         STATUS              PORTS
storm-tape      running
nginx           running             0.0.0.0:443->443/tcp,
opa             running
trust           exited (0)
ts              running
```

Shared volumes between **ts** and **storm-tape** allows to simulate the interaction with the filesystem

Deployment tests with Robot Framework

The **ts** service contains the source code for testing.

Deployment tests using a data-driven Robot Framework based on python have been performed.

Different kind of functionalities has been tested:

- authN/Z
  → **opa**, **token authz** and **voms-authz**
- compliance with the specification
  → **stage**, **archiveinfo** and **release**
- integration with GEMSS
  → **gemss**

Total Statistics	Total	Pass	Fail	Skip	Elapsed	Pass / Fail / Skip
All Tests	74	67	7	0	00:01:27	

Statistics by Tag	Total	Pass	Fail	Skip	Elapsed	Pass / Fail / Skip
archiveinfo	16	16	0	0	00:00:12	
gemss	8	8	0	0	00:00:06	
opa	33	33	0	0	00:00:58	
release	12	10	2	0	00:00:13	
stage	43	38	5	0	00:01:01	
token-authz	50	43	7	0	00:00:48	
voms-authz	17	17	0	0	00:00:37	

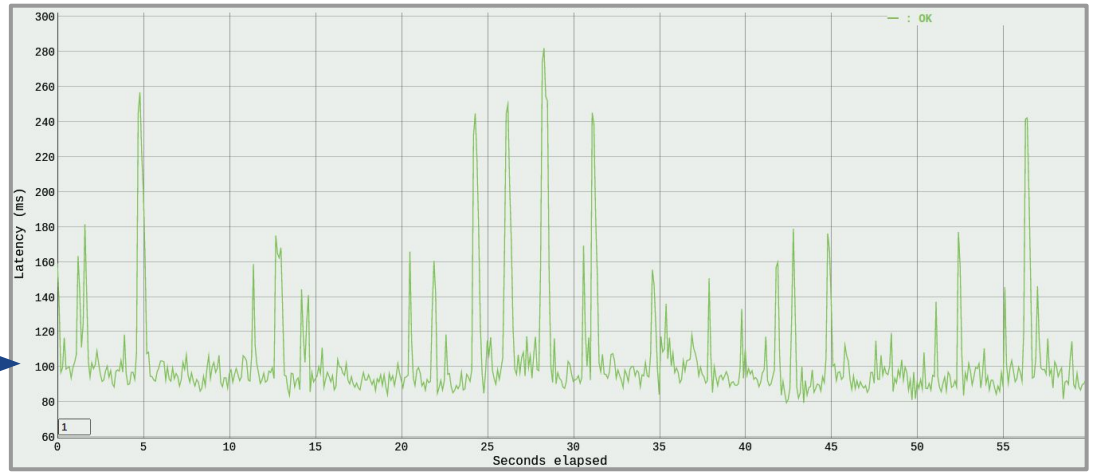
Statistics by Suite	Total	Pass	Fail	Skip	Elapsed	Pass / Fail / Skip
Test	74	67	7	0	00:01:28	
Test.Archiveinfo	10	10	0	0	00:00:06	
Test.Authorization	33	33	0	0	00:00:58	
Test.Gemss	5	5	0	0	00:00:03	
Test.General	1	1	0	0	00:00:02	
Test.Release	7	5	2	0	00:00:05	
Test.Stage	18	13	5	0	00:00:14	

Fixes on failure functionalities are undergoing!

Load tests with Vegeta

Vegeta is a versatile HTTP load testing tool built out of a need to drill HTTP services with a constant request rate

```
shtimmerman@shtimmerman:~$ echo "POST https://storage-tape-rest.cr.cnaf.infn.it/api/v1/stage" | vegeta attack -duration=40s -rate=20 -body stage.json -insecure -header "Authorization: Bearer $AT" | vegeta report
Requests      [total, rate, throughput]         800, 20.03, 0.00
Duration      [total, attack, wait]             40s, 39.95s, 50.592ms
Latencies     [min, mean, 50, 90, 95, 99, max]  43.07ms, 59.358ms, 53.637ms, 69.843ms, 106.512ms, 141.565ms, 155.793ms
Bytes In      [total, mean]                     0, 0.00
Bytes Out     [total, mean]                     0, 0.00
Success       [ratio]                           0.00%
Status Codes  [code:count]                      0:800
Error Set:
Post "https://storage-tape-rest.cr.cnaf.infn.it/api/v1/stage": EOF
```



Average latency is **100 ms**.

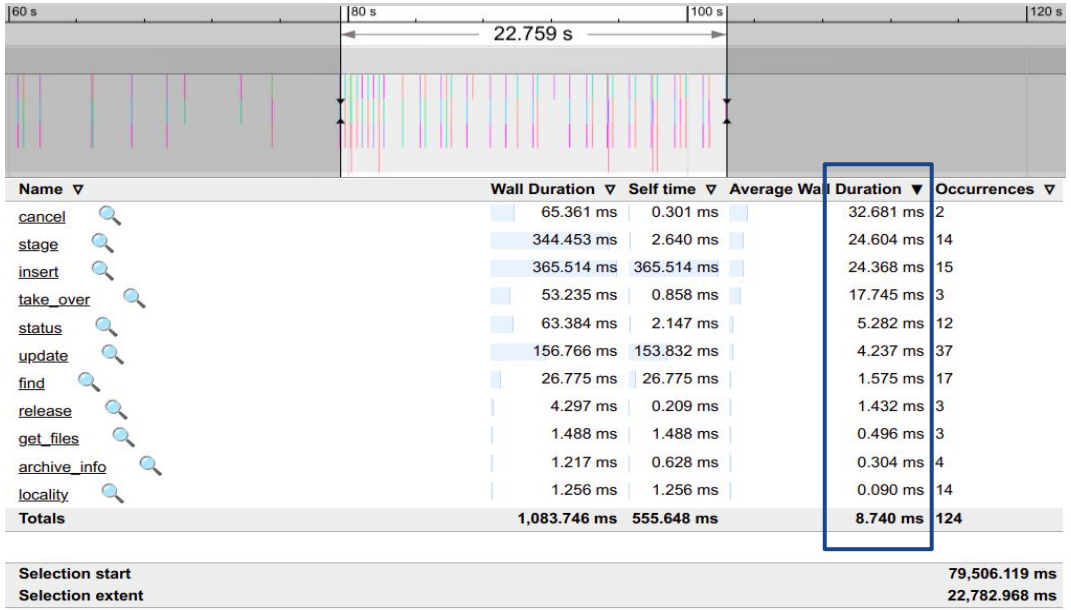
It includes the full interaction with NGNX, OPA and Tape service

Basic profiling

The functions of the StoRM Tape REST API source code have been instrumented to contain tracing informations.

At the end of the service run, a JSON file `results.json` is stored locally.

chrome://tracing



Name	Wall Duration	Self time	Average Wall Duration	Occurrences
cancel	65.361 ms	0.301 ms	32.681 ms	2
stage	344.453 ms	2.640 ms	24.604 ms	14
insert	365.514 ms	365.514 ms	24.368 ms	15
take_over	53.235 ms	0.858 ms	17.745 ms	3
status	63.384 ms	2.147 ms	5.282 ms	12
update	156.766 ms	153.832 ms	4.237 ms	37
find	26.775 ms	26.775 ms	1.575 ms	17
release	4.297 ms	0.209 ms	1.432 ms	3
get_files	1.488 ms	1.488 ms	0.496 ms	3
archive_info	1.217 ms	0.628 ms	0.304 ms	4
locality	1.256 ms	1.256 ms	0.090 ms	14
Totals	**1,083.746 ms**	**555.648 ms**	**8.740 ms**	**124**

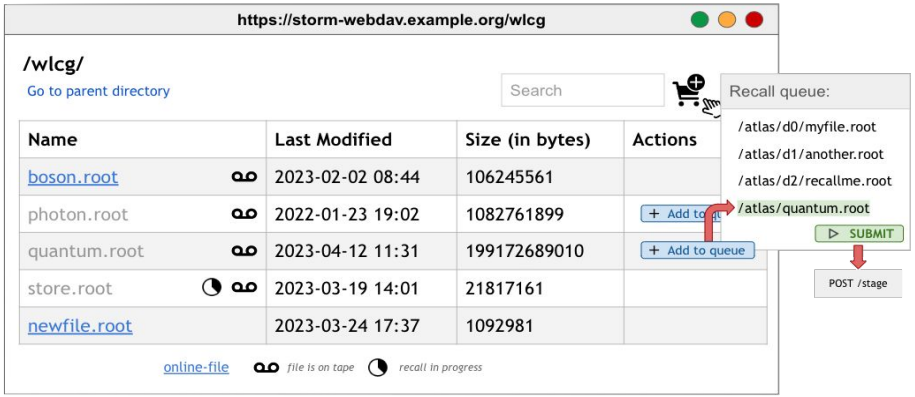
Selection start	**79,506.119 ms**
Selection extent	**22,782.968 ms**

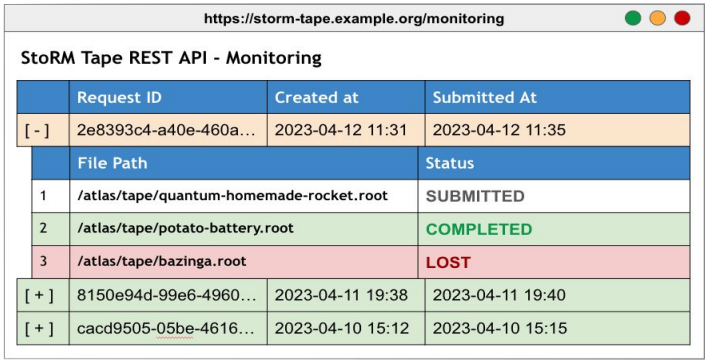
Future look

Future developments

StoRM WebDAV integration

StoRM WebDAV allows users to navigate storage areas with a browser and some privileged user could also be able to trigger a bulk stage request through the folder view





Monitoring

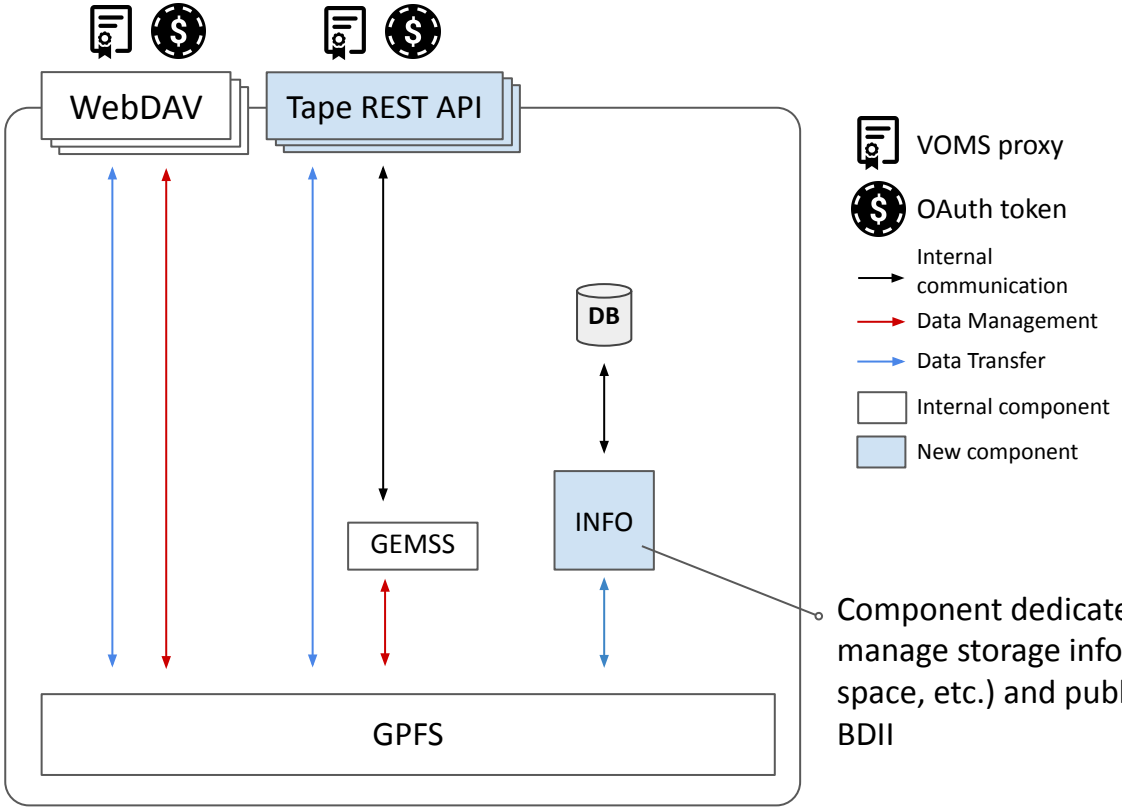
A monitoring dashboard will be available for some privileged users in order to monitor the status of the recall

Privileged users = users identified by a JWT group membership or a DN (for instance)

Future architecture: no-SRM deployment

All Globus **GridFTP** will be turned off soon.

The introduction of the **StoRM Tape REST API** will allow tape enabled no-SRM deployments → no need for StoRM Frontend and also a very big part of current StoRM Backend



Conclusions

Conclusions

- The WLCG Tape REST API working group has defined a common HTTP interface which allows clients to manage access to files stored on tape and observe the progress of file transfer on disk
- The INFN-CNAF has developed its own implementation of the specification: the **StoRM Tape REST API** service
- Initially, this service has to coexist with the current deployment, based on the StoRM storage provider
- The StoRM Tape REST API relies on NGINX/OPA external services to handle client authentication/authorization, based on VOMS proxies and JWTs
    - A proper configuration of NGINX and OPA has been setup and agreed within the CNAF Storage group
- Deployment tests on the entire setup have been carried out in a containerized environment. It appears to fulfill our requisites

Useful references

- WLCG Tape REST API specification
- StoRM documentation
- StoRM Tape REST API source code
- Crow
- SQLite documentation
- A RESTful approach to tape management in StoRM, CEPH 2023
- NGINX documentation
- ngx_http_voms_module source code
- Open Policy Agent documentation
- GEMSS source code
- StoRM Tape testsuite source code
- Robot Framework documentation
- Vegeta

Brain-StoRM-ing session: questions?

Bkp

The SRM Data lifecycle - Recognize a stub

GEMSS recognizes a stub by running a **ls -ls** and then comparing the number of blocks (1st column) with the file size (6th column)

```
$ [root@storm-test ~]# ls -ls /storage/gemss_test1/tape/testfile
0 -rw-r--r-- 1 storm 5059 5242880 14 gen  2021 /storage/gemss_test1/tape/testfile
```

blocks=0
size=5242880

is blocks * BLOCK_SIZE less than size ? If yes it is a stub.

WLCG Tape REST API project

The WLCG Tape REST API allows users to

- stage bulk-request of tape-stored files, making them available on disk
- track progress of a previously staged bulk-request
- cancel a previously staged file replicas from disk
- retrieve information about the progress of file staging

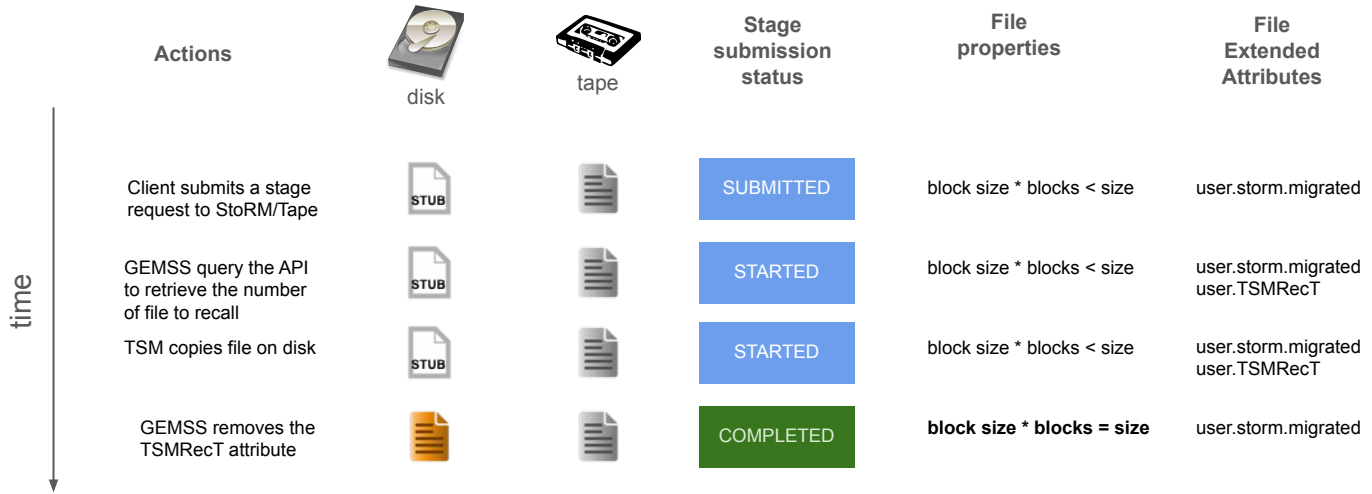
The API will be accessed via authentication mechanisms like X.509/VOMS or WLCG JSON Web Tokens (JWT)

WLCG Tape REST API specification

- STAGE: requests that tape-stored files are made available on disk
  - `POST /api/v1/stage`: bulk-request of files to be transferred from tape to disk
  - `GET /api/v1/stage/<req-id>`: track progression of a previously staged bulk-request
  - `POST /api/v1/stage/<req-id>/cancel`: indicates that the targeted subset of files are no longer needed
  - `DELETE /api/v1/stage/<req-id>`: deletion of a previously submitted STAGE bulk-request
- RELEASE: indicate that previously staged files through STAGE are no longer required on disk
  - `POST /api/v1/release/<req-id>`
- ARCHIVEINFO: requests information about disk latency
  - `POST /api/v1/archiveinfo`

How to recognize a file locality with StoRM/Tape

The file metadata are directly retrieved from the filesystem, using the extended attributes.
For files in recall, they are:



GEMSS role in the StoRM Tape deployment

- GEMSS is the Grid-Enabled Mass Storage System in use at the INFN Tier-1
- It currently retrieves the list of ready-to-recall files from the StoRM Backend
- The StoRM Tape REST API provides an additional endpoint for GEMSS to replicate the current interaction with StoRM
  - `GET https://<storm-tape-host>/recalltable/cardinality/tasks/readyTakeOver`
  - `PUT https://<storm-tape-host>/recalltable/tasks`
- Access to this endpoint is restricted (with NGINX)
  - to a limited list of IP addresses, AND
  - with Basic Authentication (username and password)
- A testbed of GEMSS supporting calls to the StoRM tape REST API has been successfully run

The StoRM Tape REST API is ready as a preview!

NGINX role in the StoRM Tape deployment

- NGINX is an open-source HTTP server and reverse proxy known for
  - high performance
  - high stability
  - rich feature set
  - simple configuration
  - low resource consumption
- The service has been chosen as part of this deployment for
  - VOMS/TLS termination
  - Authentication with JWT

nginx.conf

```
load_module modules/ngx_http_voms_module.so;
load_module modules/ngx_http_js_module.so;
…
server{
   …
   location /api/v1 {
   auth_request /authz;
   proxy_set_header        X-SSL-Client-S-Dn $ssl_client_s_dn;
   proxy_set_header        x-voms_fqans $voms_fqans;
   …
   proxy_pass              http://storm-tape:8080;
 }
 location /authz {
   internal;
   js_var $trusted_issuers
"https://wlcg.cloud.cnaf.infn.it/,https://cms-auth.web.cern.ch/";
   js_content auth_engine.authorize_operation;
 }
 location /_opa {
   internal;
   …
   proxy_pass http://opa:8181/;
 }
}
```

E.g.: POST https://storm-tape.test.example**/api/v1/stage**

NGINX role in the StoRM Tape deployment

An **auth_engine.js** module has been written at CNAF in order to

- check the presence of a JWT in the HTTP Header and, in case, validate it
- check the presence of X.509/VOMS variables (**voms_fqans**, **ssl_client_s_dn**)
- pass the above data with a POST request to OPA and handle its response

auth_engine.js

```
async function authorize_operation(r) {
  …
  r.subrequest("/_opa", opts, function (opa_res) {

    const body = JSON.parse(opa_res.responseText);

    if (!body || !body.allow) {
      r.return(403);
      return;
    }
    r.return(200);
  }
}
```

nginx.conf

```
load_module modules/ngx_http_voms_module.so;
load_module modules/ngx_http_js_module.so;
…
server{
   …
   location /api/v1 {
   auth_request /authz;
   proxy_set_header        X-SSL-Client-S-Dn $ssl_client_s_dn;
   proxy_set_header        x-voms_fqans $voms_fqans;
   …
   proxy_pass              http://storm-tape:8080;
 }
 location /authz {
   internal;
   js_var $trusted_issuers
"https://wlcg.cloud.cnaf.infn.it/,https://cms-auth.web.cern.ch/";
   js_content auth_engine.authorize_operation;
 }
 location /_opa {
   internal;
   …
   proxy_pass http://opa:8181/;
 }
}
```

GEMSS+NGINX role in the StoRM Tape deployment

Access to the `/recalltable` endpoint, that allows backward compatibility for GEMSS, is restricted

- to a limited list of IP addresses, AND
- with Basic Authentication (username and password)
    - an `.htpasswd` file with hashed passwords have to be filled and sourced

nginx.conf

```
server{
   …
 location /recalltable {
   proxy_pass                http://storm-tape:8080;

   satisfy all;
   allow                     172.0.0.0/8;
   allow                     192.168.0.0/16;
   deny all;
   auth_basic                "closed site";
   auth_basic_user_file      /etc/nginx/.htpasswd;
 }
}
```

E.g.: PUT https://storm-tape.test.example**/recalltable/tasks**

Deployment tests with Docker

- Deployment tests based on docker-compose collects all the necessary services
  - ***trust***: used for grid certificates shared with other services
  - ***storm-tape***: Tape REST API service
  - ***ts***: used for running the testsuite. `oidc-agent` and other clients are installed here. It mimics also the GEMSS component
  - ***nginx***: used as reverse proxy for the API running on the *storm.test.example* host
  - ***opa***: adds rules to access the *storm-tape* service based on VOMS attributes (`/wlcg/xfer`) or scopes in the JWT (`storage.stage:/` and `storage.read:/`)
- Shared volumes (disk and tape) between *ts* and *storm-tape* allows to simulate the interaction with the filesystem

```
$ docker-compose ps
NAME                 COMMAND                  SERVICE             STATUS              PORTS
storm-tape           "/scripts/run-servic…"   storm-tape          running
storm-tape-nginx-1   "nginx -g 'daemon of…"   nginx               running             0.0.0.0:443->443/tcp, :::443->443/tcp
storm-tape-opa-1     "/opa run --server /…"   opa                 running
storm-tape-trust-1   "/update-trust-ancho…"   trust               exited (0)
storm-tape-ts-1      "sleep infinity"         ts                  running
```