RUCIO & DMLITE

Mario Lassnig
European Organisation For Nuclear Research (CERN), Geneva, Switzerland
ph-adp-ddm-lab@cern.ch
mario.lassnig@cern.ch

ATLAS Distributed Computing Jamboree, 2012-12-10

Interaction with ATLAS DDM 1/2

- Historically, there are two primary ways to interact with ATLAS DDM
- Via dedicated DDM clients
  - `dq2-ls, dq2-get, dq2-put, …`
- Via Python API

```
from dq2.clientapi import DQ2
ret = DQ2().listFilesInDataset(…)
```

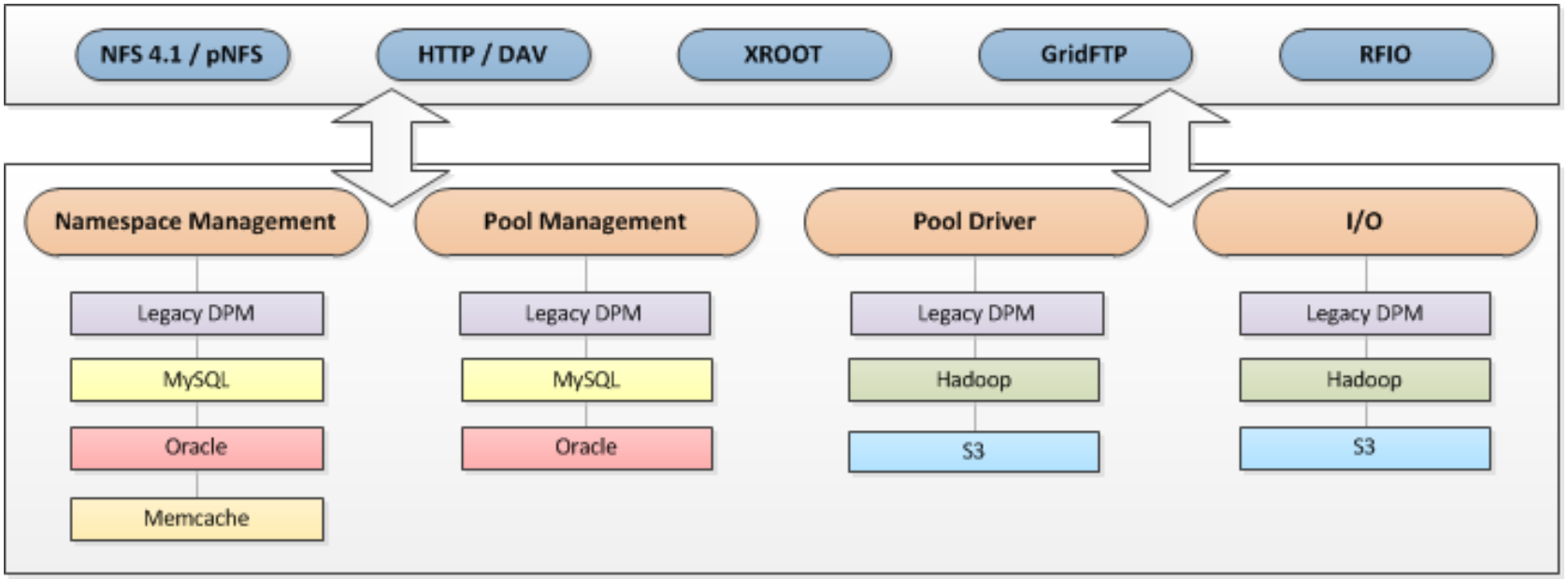
- This is problematic
  - Need to install clients
  - Python version dependency (grid vs local)
  - Native library dependency (LFC)
  - Custom Serialisation/Deserialisation
  - Many different versions deployed
  - *Not portable, not user friendly, and operational burden*

Interaction with ATLAS DDM 2/2

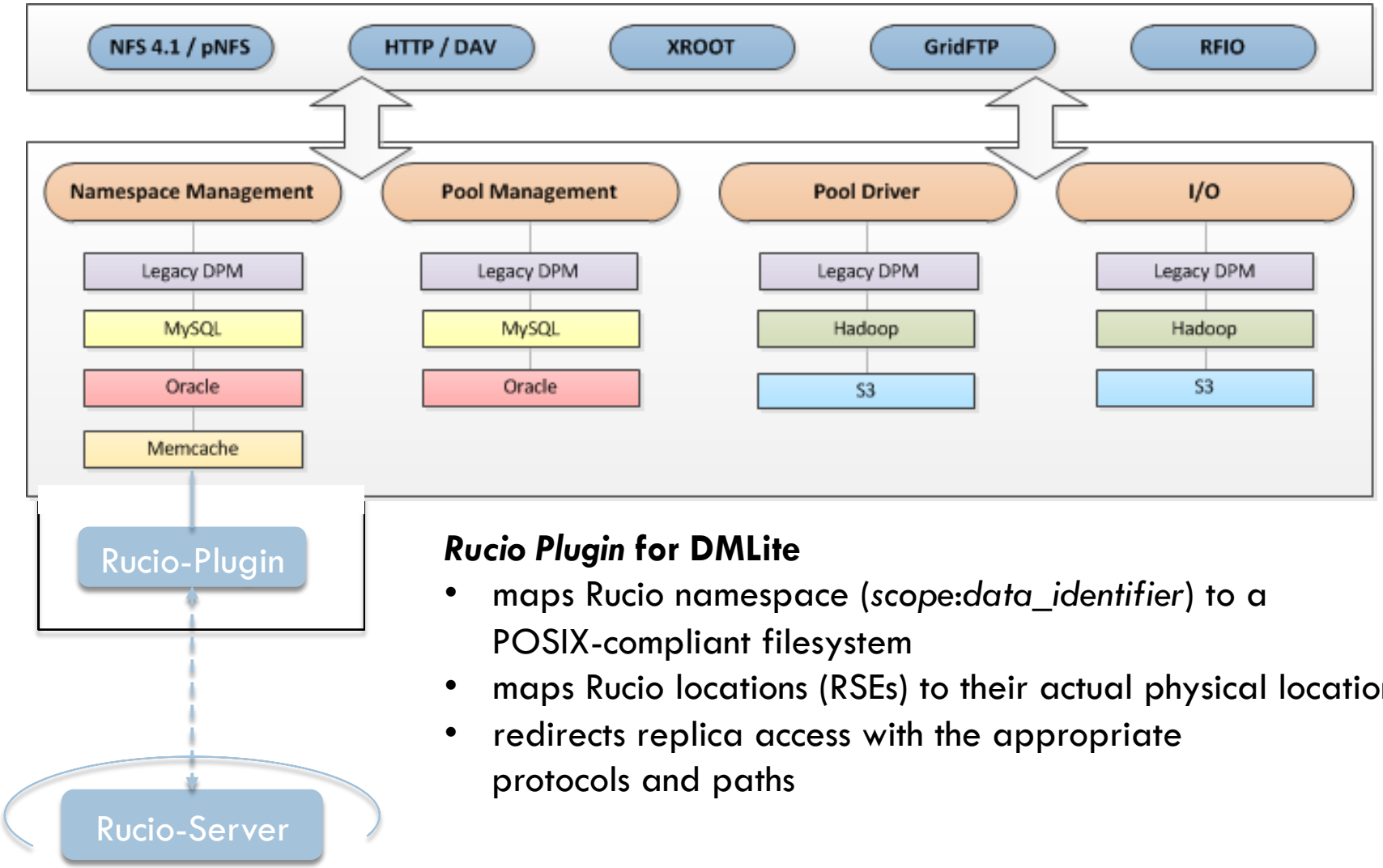
- Wouldn't it be better, if…
  - there are no clients needed to install?
  - you can access the DDM namespace and data directly?
  - from every operating system?
  - get the files you want directly without special software?

- With DMLite, we get this additional functionality
  - Software framework developed by CERN IT-GT
  - Within the DPM project
  - Distributed as a plugin-based library
  - Exposes namespaces, pools, and IO interfaces via several protocols

DMLite


NFS 4.1 / pNFS
HTTP / DAV
XROOT
GridFTP
RFIO
Namespace Management
Pool Management
Pool Driver
I/O
Legacy DPM
MySQL
Oracle
Memcache
Legacy DPM
MySQL
Oracle
Legacy DPM
Hadoop
S3
Legacy DPM
Hadoop
S3

DMLite

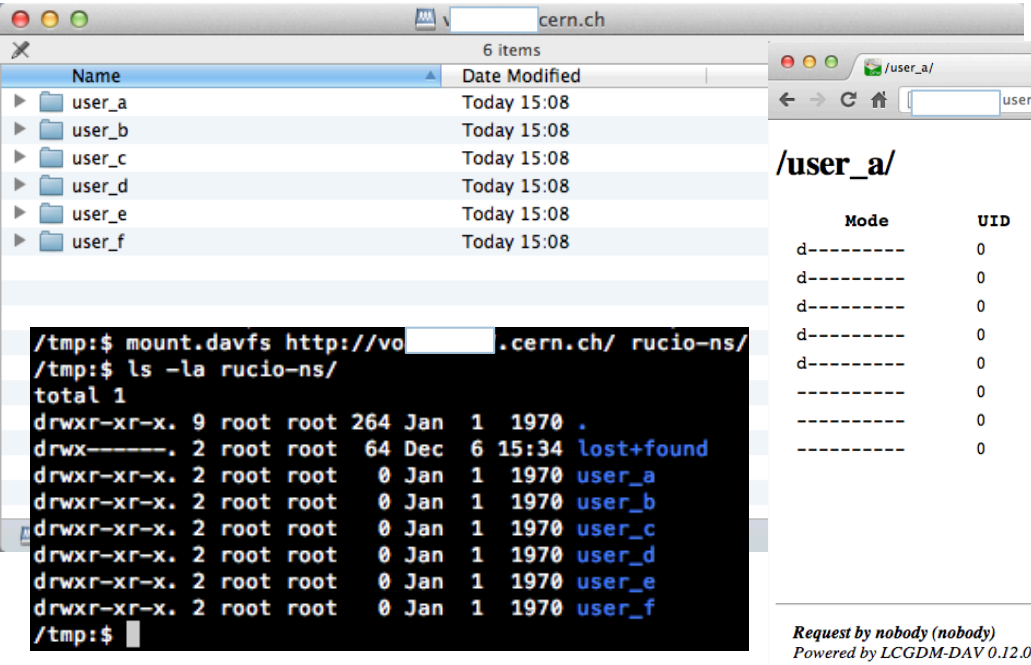


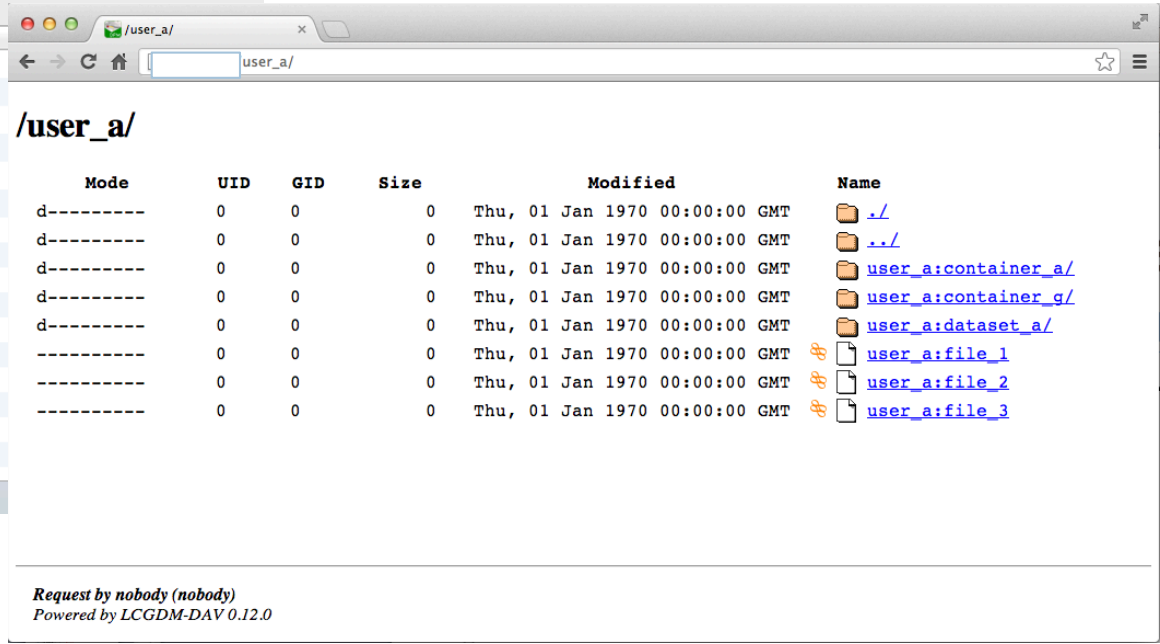
***Rucio Plugin* for DMLite**

- maps Rucio namespace (*scope:data_identifier*) to a POSIX-compliant filesystem
- maps Rucio locations (RSEs) to their actual physical location
- redirects replica access with the appropriate protocols and paths

Exposing the namespace

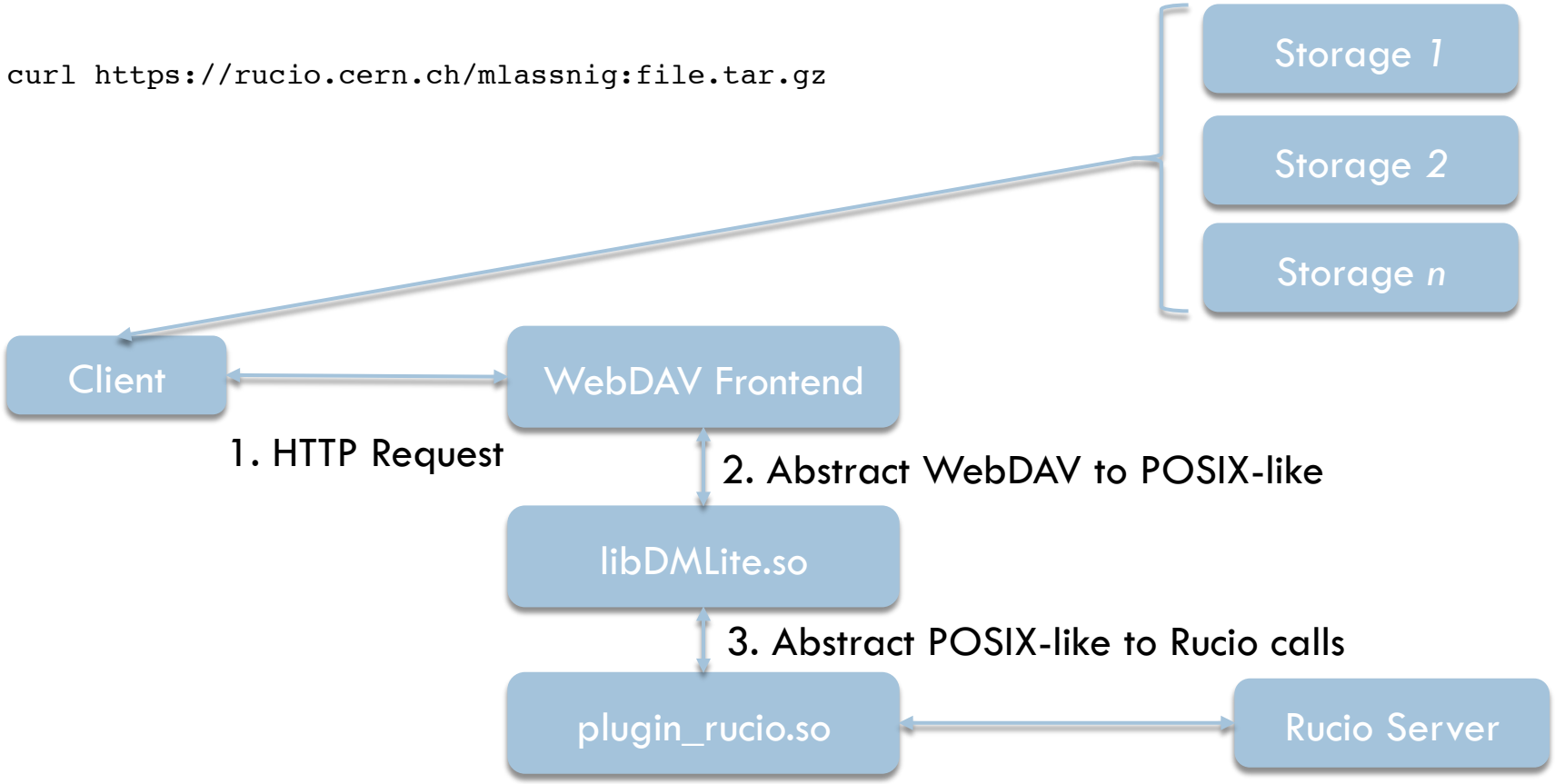
- Primarily, through a WebDAV frontend
  - Lightweight Apache plugin, using dmlite library and plugins
- Mount the Rucio namespace like a regular filesystem
  - Finder, Nautilus, dav2fs
- Or just a webbrowser or any HTTP client
  - curl, neon, aria2, …
  - Retrieve files automatically via HTTP redirection





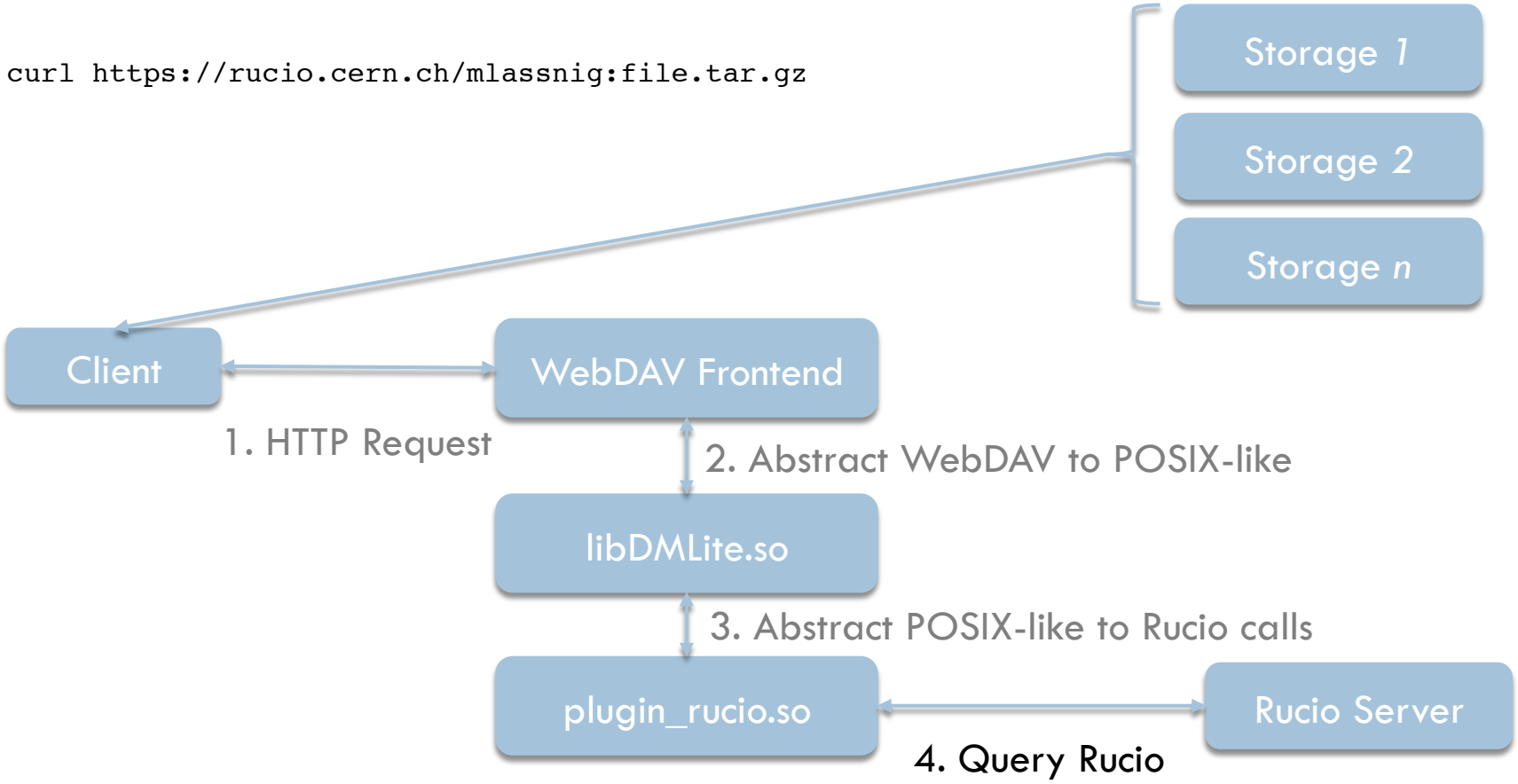
Retrieving a file WebDAV style


```
curl https://rucio.cern.ch/mlassnig:file.tar.gz
```



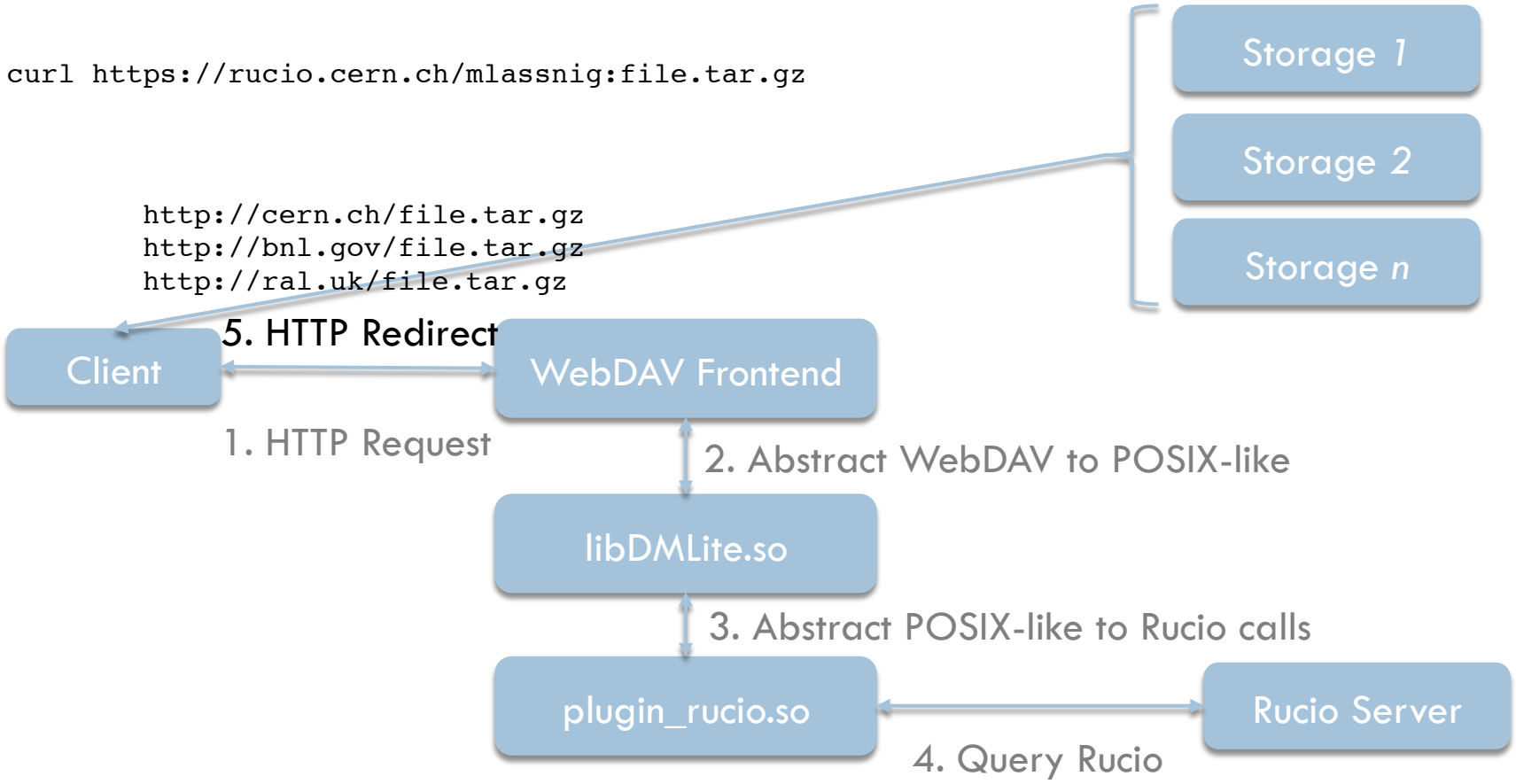
Retrieving a file WebDAV style


```
curl https://rucio.cern.ch/mlassnig:file.tar.gz
```



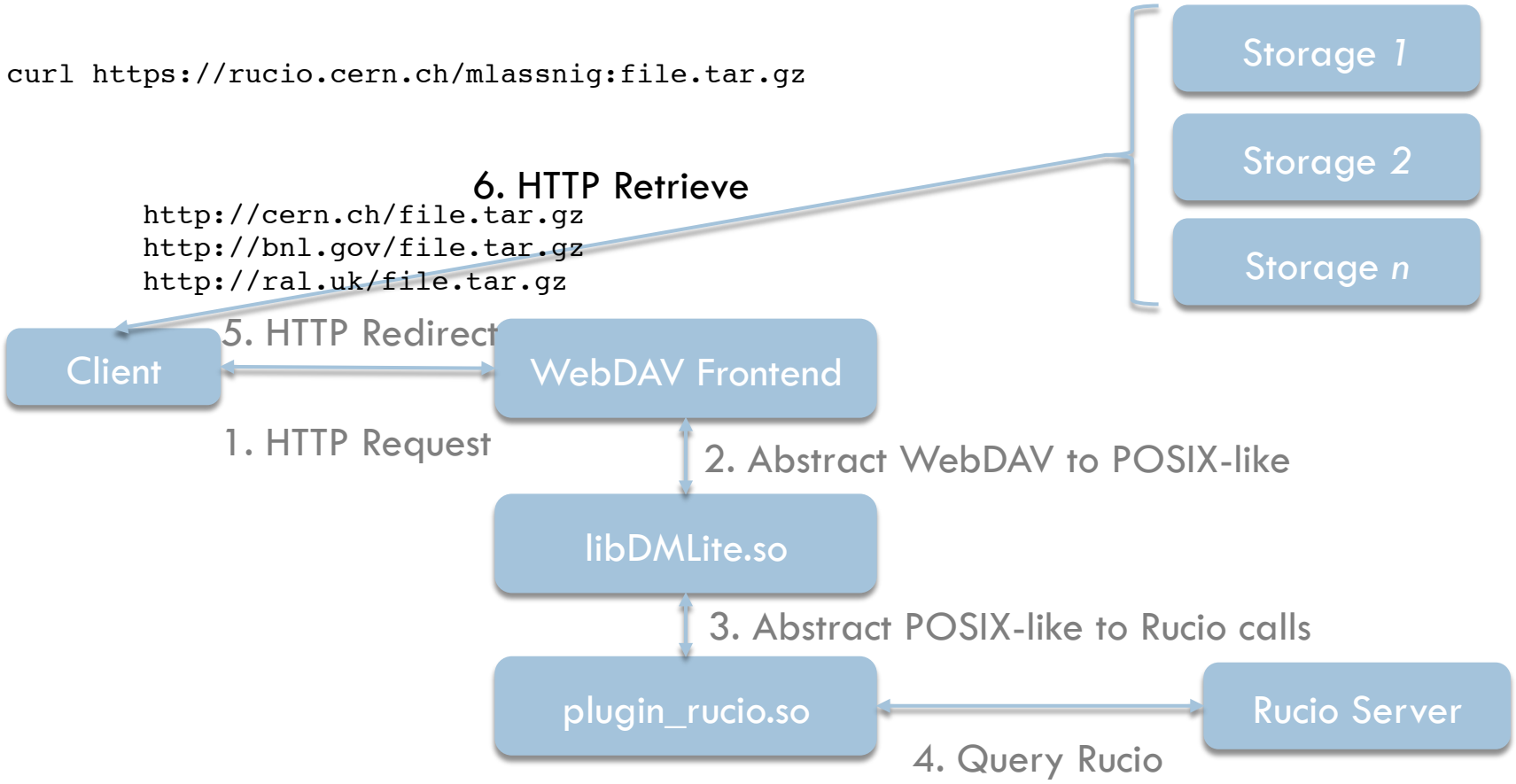

```
mlassnig:file.tar.gz → CERN, BNL, RAL
```

Retrieving a file WebDAV style



mlassnig:file.tar.gz → CERN, BNL, RAL

Retrieving a file WebDAV style



mlassnig:file.tar.gz → CERN, BNL, RAL

Next steps

- First iteration
  - browse the Rucio namespace *(done)*
  - make sure everything scales as needed *(in progress)*
- Second iteration
  - create new data, retrieve data, aggregate data (todo)

- There are three limitations
  - Scoped namespaces are not naturally mappable to a stateless tree
  - Rucio metadata cannot be naturally exposed through POSIX
  - For certain protocols client libraries are still needed

- Some additional features we get for free
  - metalink support (transparent, parallel, segmented transfer)
  - third party transfers
  - ROOT has HTTPS support
  - It's really fast