HTTP Federation Requirements for the BDII

Dynamic Federations

- Federation: a realtime unified view of the content of multiple file metadata endpoints
  - Databases or Storage Elements
  - Mostly used with Storage Elements
- Each endpoint provides its own metadata
- Each endpoint belongs to a different admin domain
  - The same file is identified by different path prefixes
  - Some information is needed to match them on the fly and present their content as merged
  - So far, we have done that manually
  - Getting this ‘matching’ information from the IS is a tempting idea

This is what we want to see as users

Sites remain independent and participate to a global view

All the metadata interactions are hidden and done on the fly

NO metadata persistency needed here, just efficiency and parallelism

Aggregation

```
/dir1
/dir1/file1
/dir1/file2
/dir1/file3
```

With 2 replicas

Storage/MD endpoint 1

```
.../dir1/file1
.../dir1/file2
```

Storage/MD endpoint 2

```
.../dir1/file2
.../dir1/file3
```

Known endpoints list

- A federator has to know the endpoints it federates
  - The HTTP Dynafeds have a “Known endpoints” list, loaded at startup of each Apache process
  - A few information items are associated to each federated endpoint

- Hence, the main element that is needed is a list of URLs, one per federated site
- **These are the HTTP:// or DAV:// URL prefixes to the federated endpoints namespace.** *What’s that?*

Filenames #1: File prefixes

- Suppose that we have the same file in two places:

https://host1/dpm/cern.ch/home/atlas/file1.root
https://host2/pnfs/desy.de/atlas/file1.root

- The federator must be told that the **BLUE** part is what we want to see, the **RED** part is just site-specific mechanics to be hidden
- We call the **RED** part a “file prefix”
- When we know the file prefix for all the federated endpoints, then we can present their content to the users in a seamless way. The federator will do the translations on the fly

Filenames #2: SRM aliasing

- Historical reasons filled the LFC index databases with fully formed SRM SURLs (there are some differences among experiments, still…)
- This induced various data management problems
- Those same issues affect federations if configured to talk to these DBs

- If a federator is configured to talk to one or more external LFC-like DBs, those DBs will likely refer to files using variously pre-cooked SRM SURLs. Here are some examples. These are not meant to be exhaustive nor exact!

srm://host1:8446/srm/managerv2?SFN=/dpm/domain.org/home/atlas/file1.root

srm://host1:8446/pnfs/domain.org/atlas/file1.root

All together: Name translations

- A Dynamic Federations federator host is able to do on the fly all the name translations, and present a coherent content
  - When presenting to the client (job or user) it will remove the site prefixes on the fly
  - It will silently add them back on the fly when
    - contacting sites to ask “do you have this file?”
    - composing a redirection to a site
  - It just needs the right information about the possible file prefixes for a host, and the trick is done. Here’a an example summary.

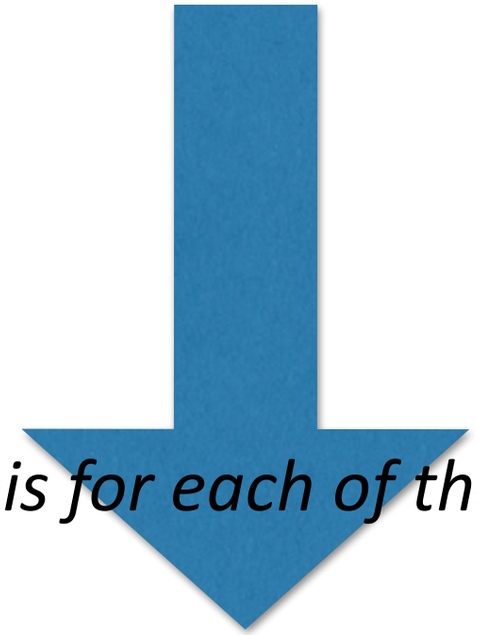
Main_access_pfx-> **http://host1/dpm/domain.org/home/**
SRMalias1-> **srm://host1:8446/srm/managerv2?SFN=/dpm/domain.org/home/**
SRMalias2-> **srm://host1:8446/pnfs/domain.org/**

Where do the prefixes come from

- Until now we have only setup manual, smallish federations
- We have taken those prefixes by looking at the LFC content site by site
  - Not so painful, yet questionable if one has many sites or cannot access the DB directly to make queries
  - On the other hand, errors are impossible
- Getting them from the information system would be a very interesting scenario
  - How to minimize errors?

What's needed

Get the list of sites supporting HTTP[s] and WebDAV[s]



Get this for each of them

Name-> **mysite1**
Main_access_pfx-> **http://host1/dpm/domain.org/home/**
SRMalias1-> **srm://host1:8446/srm/managerv2?SFN=/dpm/domain.org/home/**
SRMalias2-> **srm://host1:8446/pnfs/domain.org/**
SRMalias…