AuthZ in the gLite WMS

Marco Cecchi (INFN)

EMI security workshop – May, 24/25th, 2010 - CERN

Overall view

The gLite WMS has a fairly complex architecture made of intertwined relationships between internal and external services.

By the way, the basic authorization steps can be summarized very shortly:

1) check if the user

is able to issue

a given request

on the WMS

2) make sure that the user

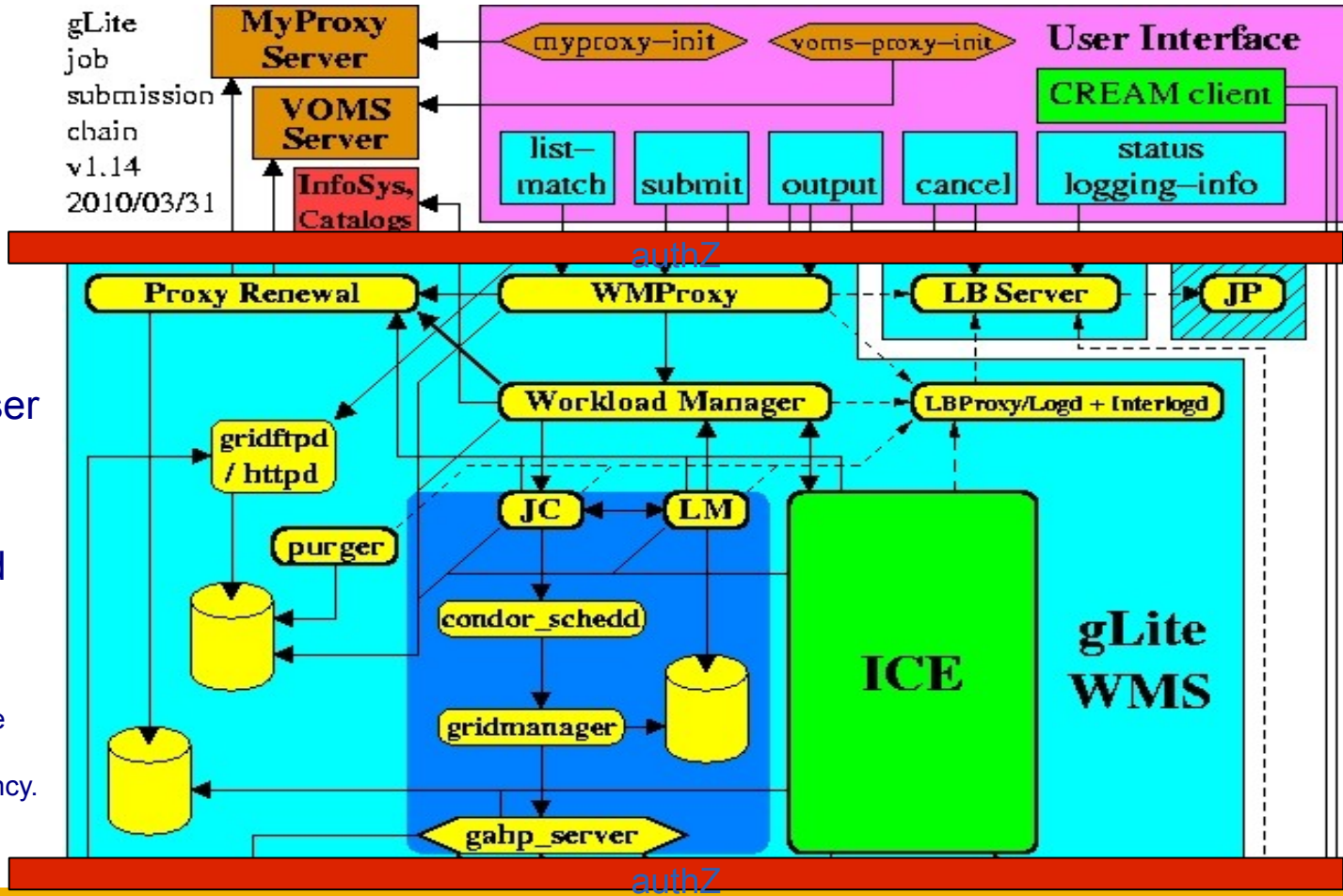
will be able to get to a

given resource, selected

for him by the WMS.

Resubmission is costly, we must be

more than sure about the consistency.



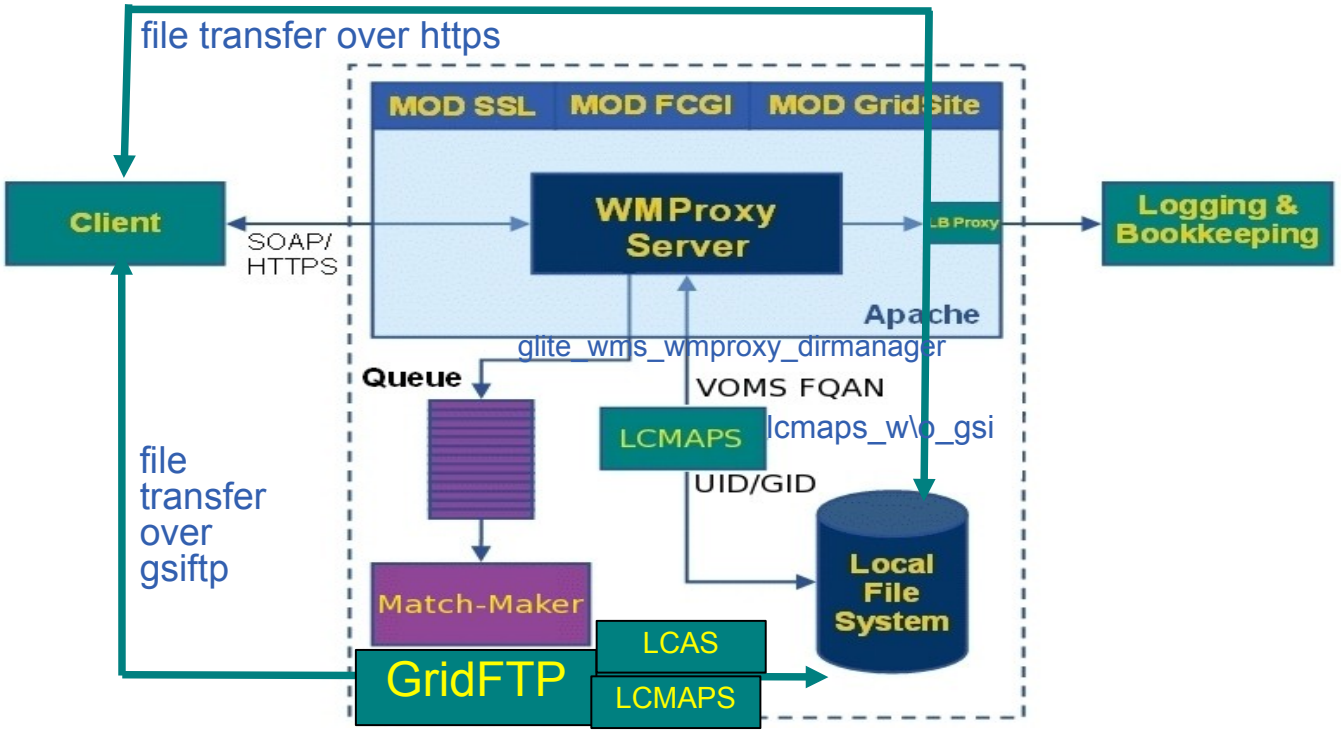
WMS interface

- The WMS interface (“wmproxy”) is a WS-I, SOAP web service running in an Apache container extended with Fast CGI.
- The Fast CGI module provides Common Gateway Interface (CGI) functionality in dynamic multiprocessing

1.a) Access to the WMS specific operations

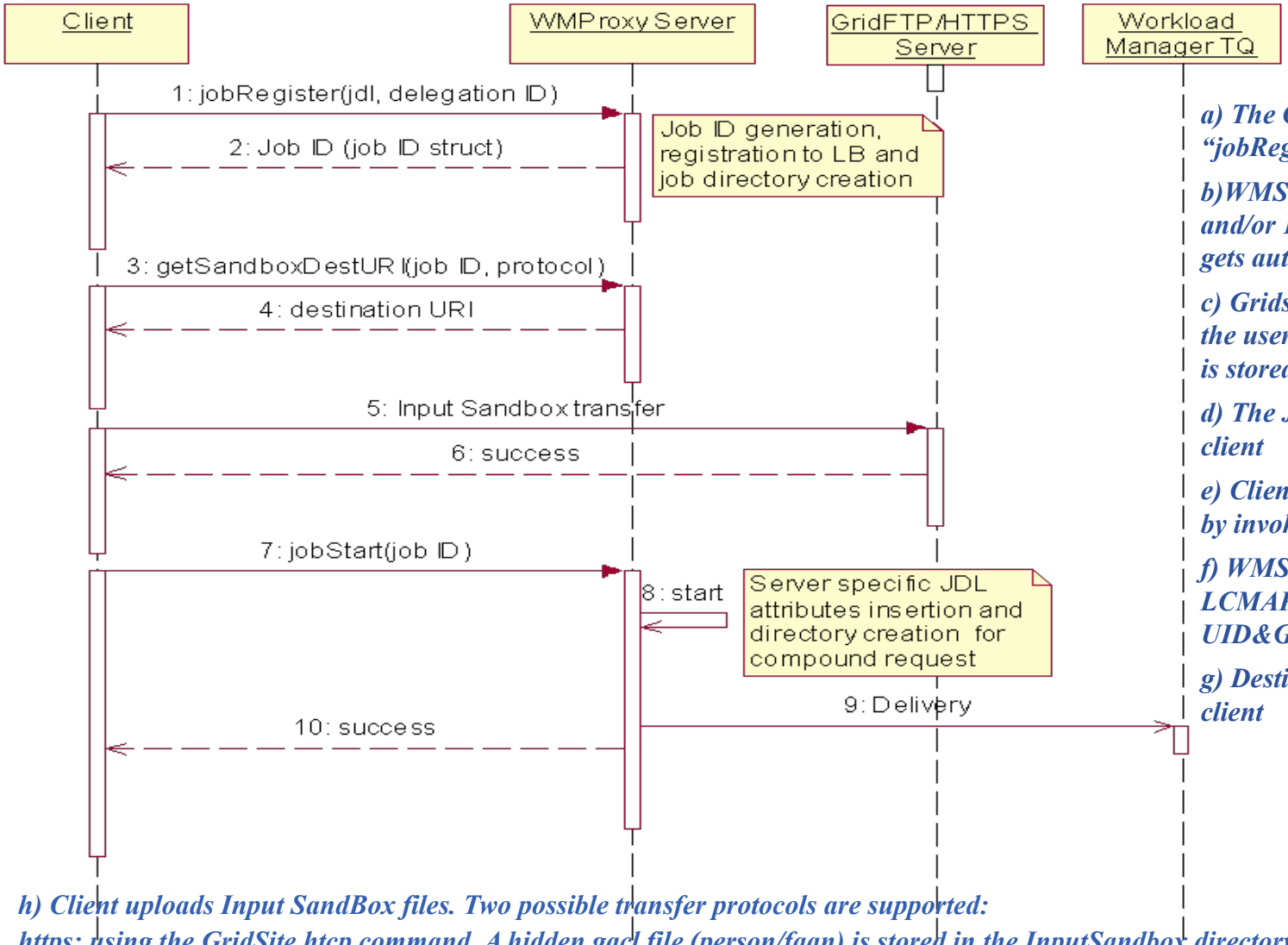
For follow-up requests, a check is performed whether the proxy of the new request matches the proxy with which the job was submitted. The criterium is that the DNs of the two proxies match (modulo the additional CN=proxy, CN=<number>).

1.b) Access to the filesystem for creating/ {up,down}loading the job sandbox



- AuthN by means of X.509 proxy certificates, handled by the Apache HTTP server through of the SSL and Grid Site plugins (mod_ssl, mod_gridsite).
- AuthZ by means of gridsite. When the GACL file is evaluated, GridSite uses all the credentials it has assembled. These are GRST_CRED_AURI_*
environment variables, including DNs and FQANs
- Delegation is done with WS gSOAP based Gridsite delegation.
- This information is then passed on to WMProxy through such environment.

Jobsubmit: a closer look



a) The Client invokes a "jobRegister" method to the WMS

b)WMS makes sure that the DN and/or FQAN of the presented proxy gets authorized by the gridsite GACL

c) Gridsite delegation takes place; the user's delegated proxy (reusable) is stored in the proxy cache

d) The Job ID is returned to the client

e) Client initiates file transfer phase by invoking getSandboxDestURI()

f) WMS creates job directory: calls LCMAPS with user's FQAN to get UID&GID.

g) Destination URI is returned to the client

h) Client uploads Input SandBox files. Two possible transfer protocols are supported:

https: using the GridSite htcp command. A hidden gacl file (person/fqan) is stored in the InputSandbox directory, which is checked by all htcpS.

gridFTP: Calls LCAS/LCMAPS to obtain the UID/GID and puts the files into the job directory (uploading upon submission or OSB transfer)

AuthZ at MM time

- The following expression is evaluated at matchmaking time in order to check whether the owner of a job has access rights to a given queue.

Very ugly at the moment:

```
AuthorizationCheck = (
   member(other.CertificateSubject, GlueCEAccessControlBaseRule) ||
   member(strcat("VO:",other.VirtualOrganisation), GlueCEAccessControlBaseRule) ||
   FQANmember(strcat("VOMS:",other.VOMS_FQAN), GlueCEAccessControlBaseRule)
) && !FQANmember(strcat("DENY:",other.VOMS_FQAN), GlueCEAccessControlBaseRule);
```

- this expression is appended to the job requirements by the WMS, so that the MM will also filter out unauthorized resources

Some final thoughts

- Delegation and proxy renewal are costly operations
  - while it looks like that for the moment we can't get away with the former, the idea of having a 'DN-centric' proxy renewal (now it is job-centric) is worth being pursued.

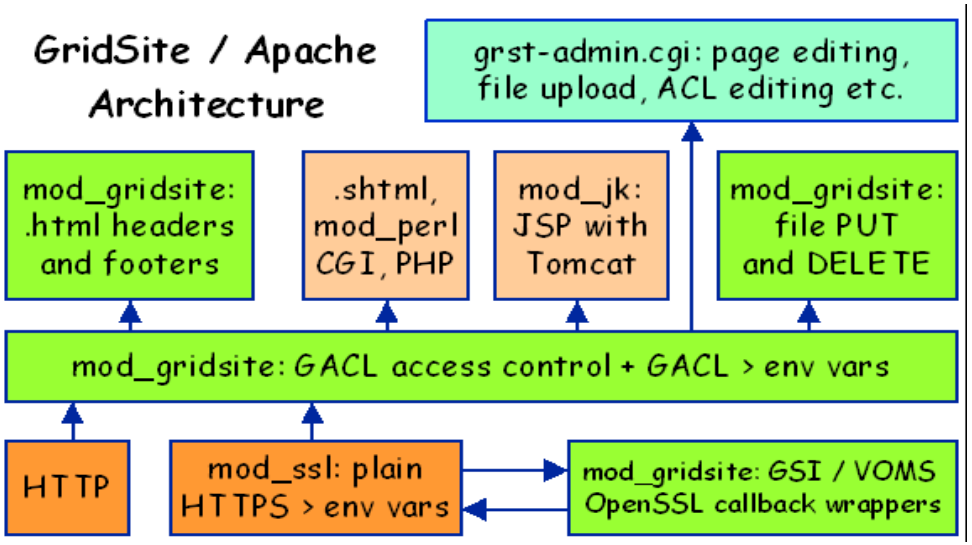
- AuthZ at MM time must be done soon
  - how WLCG experiments will react in practice? / is there an agreed roadmap?

- The current production BDII has more than 8000 entries
  - Will we be able to perform 'bulk' authorization requests?
  - At present, matchmaking against the whole BDII takes ~1 second and basically stresses only CPU, given that information collected from the Information Providers (including authZ information) is cached.
  - Also, average network utilization in production nodes is roughly:
    - In ~300kb/sec / Out ~1Mb/sec
      - Transfer of huge XMLs should not significantly impact on these numbers.

Gridsite

- provides a module extending the Apache webserver for use within Grid frameworks by Adding support for Grid security credentials such as Grid Security Infrastructure (GSI) and VOMS, and **file transfer** over HTTPS.
It provides a library for handling Grid Access Control Lists (GACL).

mod_gridsite is a pluggable module for the Apache web server which provides access control and page formatting for GridSite HTTP(S) Fileservers, Websites and Web Services hosts. **mod_gridsite also intercepts some processing in the standard mod_ssl module to support GSI Proxies and VOMS attribute certificates**. GT2 proxies are supported by the current production version, GT4 (rfc proxy-style) will be also supported starting from WMS 3.3.
The verification of these credentials is handled by functions in mod_gridsite without the need to patch or rebuild mod_ssl.



GACL

Authorization can be either Fully Qualified Attribute Name (FQAN) or Distinguished Name (DN)

Based. GACL allows policies to be written in terms of common Grid credentials:

•X.509 identities

```
<person> <dn>/O=Grid/CN=Name</dn> </person>
```

•VOMS attribute certif cates

```
<voms> <fqan>/vo.dom.ain/group</fqan> </voms>
```

• lists of X.509 identities

```
<dn-list> <url>https://www.vo.dom.ain/dn-lists/group</url> </dn-list>
```

•any user

```
<any-user/>
```

Four permissions are supported: `write`; `read`; `list`; `exec`.

For example:

```
<gacl>
<entry>
        <any-user/ ><allow><exec/ ><list/ ></ allow><deny><write/ ></ deny></ entry>
<entry>
        <person> <dn>/ C=IT/ O=abcd/ OU=Personal Certif cate/ L=XYZ/ CN=John Doe</ dn>
        </ person>
        <allow><write/ ></ allow> </ entry>
</ gacl>
```