GUMS vs. LCMAPS

Oscar Koeroo

www.eu-egee.org

index

- General Approach
- Approach to Authentication
- Implementation
- Plug-in Architecture
- Policy Handling
- Mapping Service
- Proposals for future improvements

- Bottomline…

General Approach

GUMS

- Separation of the decision point from the client.
- GUMS is a centralized Policy Decision Point that can serve several clients.
- On the client side, it has the PRIMA libraries. Plugins for GT2, GT4 and gLExec exist.

LCMAPS

- All-in-one solution
- LCMAPS is both the decision point and client library
- The policies are stored in a file system accessible by the LCMAPS framework.

Approach to Authentication

GUMS

- The client verifies the credentials and extracts the DN and the primary FQAN
- The DN+FQAN are sent to GUMS over server-side-only HTTPS using the SAML protocol
- GUMS returns a SAML response + XACML obligations (non standard), like UserID=XYZ

- The current implementation of PRIMA does not validate the validity of FQANs.
  - The GUMS database pre-stores the DN to FQANs group affiliations thus a user can't go out of this boundary

LCMAPS

- **Verifies the user credentials, both the DN and FQANs.**
- **The LCMAPS framework holds all credentials**
  - either from the input (GK, glexec, gridFTP)
  - Or by using one of the acquisition plugins that scavenge credentials
- **The LCMAPS framework extracts and verifies the VOMS information on extraction from the ACs**
  - The LCMAPS framework doesn't do any checks itself, for this you'll have the lcmaps_verify_proxy plugin to do the job
- **When LCMAPS is executed the successful mapping is performed on the current process itself**
- **Credentials passed to LCMAPS can be accumulated and verified; a mapping flows out of that when sufficient credentials are present and verified**
- **The mapping granularity is in control of the sysadmin.**
  - No need to sync with a VOMS server
  - All authZ and mapping can be done according to a local configuration
  - No need to construct a relational database (reconstruction of the VOMS DB on each site) of all the users of all VOs that wish to have a potential mapping on a site

Implementation

GUMS

- GUMS is a web service. Java code inside Tomcat.
- Typically, GUMS needs:
  - Tomcat instance
  - MySQL
- PRIMA available in both C (for GT2 and gLExec) and Java (GT4) implementations.
  - No requirements for PRIMA (just the code).

LCMAPS

- Everything is C based.
- Policy store today:
  - filesystem
- Cross-node mapping consistency can be implemented via NFS lock mechanism
- Traceability via JobRepository DB (an optional plug-in)

Plug-in architecture

GUMS

- **Plug-in based with Java-based plug-ins.**
- **Possibility to add new Classes to add functionality**
- **All configuration held in a single, XML file. The plug-ins configured here too, as attributes of plug-in tags.**

LCMAPS

- **Plug-in based.**
  - Plug-ins are shared libraries.
  - One global text file to list the shared libraries to include.
- **Each plugin is initialized from the lcmaps.db config file.**
  - If needed (like the database password for the Job Repository) plugins could need their own config files.

Policy handling

GUMS

- **By class interfaces: Five main types of class interfaces:**
  - storage
    - database (JDBC) - most used, support both static and dynamic mappings
  - User groups
    - manual - forced one to one mapping
    - VOMS group accounts- load every 6 hours all DN/FQANs from a VOMS and maps them all into one UID
    - VOMS pool account - load every 6 hours all DN/FQANs from a VOMS and maps them all into pool accounts (all different)
    - LDAP
  - host to group mapping
    - given a host expression (like "fcdf*.fnal.gov") list of groups to map to.
  - group to account mapping
    - given a group, one or more account mappers that will return the local account to map to (the first one to return a hit).
  - user group
    - Used by group to account mapping to verify user group/VO membership given user DN+FQAN.
  - account mapper
    - Used by group to account mapping to return account name given user DN.

LCMAPS

- **The policy handling in LCMAPS is based around the plug-ins that it will need to execute**
  - Which means the plug-ins can control the course for the mapping

- **Quite simple state machine:**

<policy name>:

plugin1 (execute this plugin) -> plugin2 (if plugin1 is successfull) | plugin3 (if plugin1 failed)

plugin2 (execute plugin2) -> plugin3 (execute plugin3 when plugin2 is successfull)

Example policies

GUMS

```
<gums>
          <persistenceFactories>
            <persistenceFactory
              name="mysql"
              className="gov.bnl.gums.hibernate.HibernatePersistenceFactory"
              hibernate.connection.driver_class="com.mysql.jdbc.Driver"
              hibernate.dialect="net.sf.hibernate.dialect.MySQLDialect"
              hibernate.c3p0.min_size="3"
              hibernate.c3p0.max_size="20"
              hibernate.c3p0.timeout="180"
              hibernate.connection.url="jdbc:mysql://localhost:49251/GUMS_1_1"
              hibernate.connection.username="****"
              hibernate.connection.password="****"
              hibernate.connection.autoReconnect="true"/>
            </persistenceFactories>
            <groupMappings>
              <groupMapping name="atlas">
                <userGroup
                  className="gov.bnl.gums.LDAPGroup"
                  server="grid-vo.nikhef.nl"
                  query="ou=lcg1,o=atlas,dc=eu-datagrid,dc=org"
                  persistenceFactory="mysql"
                  name="atlas"/>
                <accountMapping
                  className="gov.bnl.gums.GroupAccountMapper"
                  groupName="usatlas1"/>
              </groupMapping>
              <groupMapping name="vomsAtlas">
                <userGroup
                  className="gov.bnl.gums.VOMSGroup"
                  url="https://lcg-voms.cern.ch:8443/voms/atlas/services/VOMSAdmin"
                  persistenceFactory="mysql"
                  sslCertfile="/etc/grid-security/gumscert.pem"
                  sslKey="/etc/grid-security/gumskey.pem"
                  matchFQAN="ignore"
                  acceptProxyWithoutFQAN="true"
                  voGroup="/atlas"
                  name="vomsatlas"/>
                <accountMapping
                  className="gov.bnl.gums.GroupAccountMapper"
                  groupName="usatlas1"/>
              </groupMapping>
               <groupMapping
                name="cdfPool"
                accountingVo="cdf"
                accountingDesc="CDF">
                <userGroup
                  className="gov.bnl.gums.VOMSGroup"
                  url="https://voms.cnaf.infn.it:8443/voms/cdf/services/VOMSAdmin"
                  persistenceFactory="mysql"
                  name="osgcdf"
                  voGroup="/cdf"
                  sslCertfile="/etc/grid-security/gumscert.pem"
                  sslKey="/etc/grid-security/gumskey.pem"
                  matchFQAN="ignore"
                  acceptProxyWithoutFQAN="true"/>
                <compositeAccountMapping>
                  <accountMapping
                    className="gov.bnl.gums.AccountPoolMapper"
                    persistenceFactory="bnl"
                    name="bnlPool.cdf"/>
                </compositeAccountMapping>
              </groupMapping>
            </groupMappings>
            <hostGroups>
              <hostGroup
                className="gov.bnl.gums.CertificateHostGroup"
                cn="cdfonly*.fnal.gov"
                groups="cdfPool"/>
             <hostGroup
                className="gov.bnl.gums.CertificateHostGroup"
                cn="osg*.fnal.gov"
                groups="cdfPool,vomsAtlas"/>
             <hostGroup
                className="gov.bnl.gums.CertificateHostGroup"
                cn="lcg*.fnal.gov"
                groups="cdfPool,atlas"/>
            </hostGroups>
        </gums>
```

LCMAPS

```
voms:
        vomslocalgroup -> vomspoolgroup
        vomspoolgroup -> vomspoolaccount | vomspoolaccount
        vomspoolaccount -> posix_enf

legacy:
        localaccount -> posix_enf | poolaccount
        poolaccount -> posix_enf
```

Mapping service

GUMS

- Maps DN+primary FQAN into a UID and optionally a GID, too.
- PRIMA passes these values to the calling client.

LCMAPS

- Maps DN+primary FQAN into (UID,GID)
  - All secondary FQANs are mapped to secondary GIDs
- LCMAPS is the calling client itself

Proposals for future improvements

GUMS

- Convert GUMS to use XACML. This way we can relinquish PRIMA and use standard XACML libraries.
- If possible, integrate GUMS functionality into the Globus CAS.

LCMAPS

- Create central site AuthZ and Mapping service
- Split the VOMS Acquisition from the LCMAPS framework (like it was 2 years ago)
- Try to find a more common way to store credentials in the framework
  - treat them as arbitrary sources for mappings.
  - In this way we can support:
    - Globus CAS
    - Shibboleth
    - VOMS
    - other OIDs and any other possible type of credential.

Bottomline…

- **GUMS+PRIMA performs the same task as LCMAPS but has a quite different design**
  - Due to different views and impacts of that design we can’t use GUMS+PRIMA directly (at least not) on the European sites
    - Current GUMS uses mkgridmap-style VO member propagation based on DN string matching only (not signed assertions)
    - Not going to convert LCMAPS to work with GUMS natively within a foreseeable future
    - LCMAPS (and LCAS) will also sport a central AA/mapping service
    - Wire protocol compatibility is more viable route
  - GUMS may alter its design to be more compatible
    - Needs a internal reimplementation on the mapping sequences
  - Plug-ins created by a 3rd-parties (like GPBox and AFS plug-in, and the upcoming Shib plug-in)
    based on the LCMAPS interfaces and will need to be re-implemented to be used in a GUMS environment

?