Glite Resource Broker Adapter Readme

Thomas Zangerl

August 7, 2008

Contents

1 VOMS-Proxy Creation

1.1 Frequent errors

1.1.1 "Unknown CA" error

The proxy classes report an "Unknown CA" error (Could not get stream from secure socket). Probably the VomsProxyManager is missing either your root certificate or the root certificate of the server you are communicating with. It is best, if you include all needed certificates in /.globus/certificates/. (e.g. you can copy the /etc/grid-security/certificates directory from an UI machine of the VO you are trying to work with to that location).

If this doesn't suffice, you should try to include a file called cog.properties in the /.globus/ directory. The content of this file could be something like this:

```
Java CoG Kit Configuration File
#Thu Apr 05 15:59:23 CEST 2007
usercert=~/.globus/usercert.pem
userkey=~/.globus/userkey.pem
proxy=/tmp/x509up_u<your user id>
cacert=/etc/grid-security/certificates/
ip=<your ip address>
```

Also check the vomsHostDN preference value for typos/errors.

1.1.2 "Error while setting CRLs"

Try to create a cog.properties file in /.globus and set the cacert property to /.globus/certificates. This will cause the VOMS Proxy classes not to look for CRLs in /etc/grid-security/certificates

1.1.3 "pad block corrupted"

Check whether you have given all necessary information (password, host-dn, location of your user-certificate etc., see section 1.3) as global preferences of the GAT-context.

If you have given all necessary information, check the password you have given, for typos.

1.1.4 Could not create VOMS proxy! failed: null:

See answer in section 1.1.3, maybe you forgot to specify the vomsServerPort preference value.

1.2 Preference keys for VOMS Proxy creation

The **necessary** preference keys are:

vomsHostDN	distinguished name of the VOMS host	/DC=at/DC=uorg /O=org/CN=somesite	compulsory
vomsServerURL	URL of the voms server, without protocol	skurut19.cesnet.cz	compulsory
vomsServerPort	port on which to connect to the voms server	7001	compulsory
VirtualOrganisation	name of the virtual organisation for which the voms proxy is created	voce	compulsory
vomsLifetime	he desired proxy lifetime in seconds	3600	optional

Additionally you need a CertificateSecurityContext which points to your user certificate and your user key. Add that CertificateSecurityContext to the GATContext.

With the compulsory preferences, the proxy is created with a standard lifetime of 12 hours. If you want to have a different lifetime, add the optional vomsLifetime preference key.

Do something like

```
GATContext context = new GATContext();
CertificateSecurityContext secContext =
new CertificateSecurityContext(new URI(your_key), new URI(your_cert), cert_pwd);
Preferences globalPrefs = new Preferences();
globalPrefs.put("vomsServerURL", "voms.cnaf.infn.it");
...
context.addPreferences(globalPrefs);
context.addSecurityContext(secContext);
```

1.3 Minimum configuration to make VOMS-Proxy creation work

- A cog.properties file with lines as in section 1.1.1.
- The following global preferences set in the gat context
  - vomsHostDN
  - vomsServerURL
  - vomsServerPort
  - VirtualOrganisation

1.4 (Not) reusing the VOMS proxy

If multiple job submissions to the same VO happen in a small time interval, it is not necessary to create a new VOMS proxy for every submission. Hence, if the user sets the system property glite.reuseProxy to true, the system will check, whether a valid VOMS-Proxy exists. If such a proxy is found, it is determined, whether the proxy lifetime exceeds the one specified in the vomsLifetime property. If the vomsLifetime property is unspecified, it is checked wether the proxy is still valid for more than 10 minutes. Very frequently, all job submissions of an application will happen within the same VO. In case there is the need to submit jobs to multiple VOs, it won't be possible to to reuse the old proxy due to the VO-specific attribute certificates stored in the proxy file.

2 The gLite-Adaptor

2.1 Adaptor-specific system properties

Indeed, the mechanisms provided by the GAT-API alone did not suffice to provide all the control we found desirable for the adaptor. Hence, a few proprietary properties where introduced. They are useful in controlling adaptor behaviour but are by no means necessary if one just wants to use the adaptor. Nonetheless, they are documented here.

If you want to use them, set them using System.setProperty(); for example write System.setProperty("glite.pollIntervalSecs", "15"). The following properties are supported:

- glite.pollIntervalSecs - how often should the job lookup thread poll the WMS for job status updates and fire MetricEvents with status updates (value in seconds, default 3 seconds)
- glite.deleteJDL - if this is set to true, the JDL file used for job submission will be deleted when the job is done ("true"/"false", default is "false")
- glite.reuseProxy - if this is set to true, don't create a new proxy if the lifetime of the old one is still sufficient

2.2 Supported SoftwareDescription attributes

The minimum supported attributes from the software description seem to be derived from the features that RSL (the globus job submission file format) provides. Hence, they are easy to translate to RSL properties. However, the format used for glite job submission is JDL and attributes like count or hostCount are hard to translate to JDL. Most of the attributes that **are** supported are not even achieved by the JDL format itself, but by adding GLUE requirements. Sadly, the JDL format does not provide much of the functionality covered by RSLs, hence many attributes remain unsupported.

On the other hand, to enable working with the features that the JDL format provides additionally to the RSL format, a new attribute was introduced.

Set glite.retryCount to some String or Integer in order to use the retry count function of glite.

2.3 Setting arbitrary GLUE requirements

If you would like to specify any GLUE-Requirements that are not covered by the standard set of SoftwareResourceDescription or HardwareResourceDescription keys, you may set

glite.other either as Software- or HardwareResourceDescription and add a *full* legal GLUE Requirment as entry, such as for example
`!( other.GlueCEUniqueID == "some_ce_of_your_choice")`.