EUROPEAN MIDDLEWARE INITIATIVE

LCAS

Document version:	**1.0.0**
EMI Component Version:	**1.3.13**
Date:	**April 28, 2011**

This work is co-funded by the EC EMI project under the FP7 Collaborative Projects Grant Agreement Nr. INFSO-RI-261611.

1 Introduction

The LCAS framework is designed to take an authorization decision based on various credentials as input, e.g. a certificate and/or VOMS[1] credentials, and give a yes/no decision. LCAS is a framework that can load and run one or more authorization plugins. The framework will load and run plugins to perform the authorization decision. The LCAS framework exposes various APIs to push credentials into the framework and return the yes/no decision. The *lcas.db* configuration file configures the LCAS plugins and configures the order in which the plugins are launch. Some practical examples are shown below.

LCAS is used by *gLExec*, the *lcas-lcmaps-gt4-interface* to interface with a Globus GT4 and GT5 Gatekeeper, GridFTP daemon and GSI-OpenSSHd, in StoRM and various other places.

2 History

The Gridification subtask of WP4 of the European Datagrid project[2] interfaces the local fabric to other middleware components by a number of services, among which the Local Centre Authorization Service (LCAS) handles authorization requests to the local computing fabric and the Local Credential Mapping Service (LCMAPS) provides all local credentials needed for jobs allowed into the fabric. This document describes LCAS, which is the first component released by the Gridification subtask.

3 The frameworks inner working and plugins

When an application intializes LCAS the plugins will be loaded based on the *lcas.db* configuration file. The application can use one of the APIs to provide credentials as input. The loaded plugins will be executed in the sequence of their listing in the *lcas.db* file.

During a plugin's execution it has access to the credential data in the LCAS core memory. Based upon this information the plugins can use this to formulate a yes or no decision to authorize a person. All configured plugins must return with a statement to authorize the credentials access to the service.

4 Installation

The easiest way of installing LCAS is to use the *EPEL*, *Debian*, *Etics* or Nikhef software repositories[3]. The installation experience is similar when manually build from a tarbal with the command: `./configure && make && sudo make install`.

[1]Virtual Organization Management System by INFN CNAF
[2]European Datagrid project: http://www.eu-datagrid.org
[3]Nikhef software repository: http://software.nikhef.nl/

When you've downloaded the software from our Subversion repository the `./bootstrap` command is mandatory.

The software will be installed in the following system default locations. The locations can be altered at ./configure time or the system might have di erent default location specified then Debian and/or Fedora systems:

/usr/lib{64}/ The core library files will be located here. This includes the front-end API implementing libraries.

/usr/lib{64}/modules/ This is the default path for all the LCAS plugins. The same directory is used to hold the LCAS plugins.

/etc/lcas/ The configuration file *lcas.db* will be placed in this directory by default.

/usr/share/doc/lcas-${version} The documentation and example files will be located here.

4.1 Dependencies

The dependencies of LCAS are:

Globus 2.x through Globus 5.x. Older version of Globus are known to work.

OpenSSL 0.9.7 through 1.0

4.2 Custom configure options

The LCMAPS *configure* script is trying a few methods on finding the depending libraries on the system, including pkg-config. To be able to work with non-system distribution provided installation or personal compiles explicitly we provide several *configure* options to work with:

–enable-headers This switch will only build and install or distribute the header files to LCAS. The header files will contain the required type definitions for both the front-end/application interface as also the plugin interface.

–enable-gsi-mode This option is *on* by default and will build LCAS against the GSI interface libraries and OpenSSL. Switch this option to *no* to create the *lcas-without-gsi* flavor of LCAS. NOTE: This flavor is *NOT* supported.

–with-globus-prefix=PFX Allows you to select an alternative location to find the Globus headers and libraries needed to build LCMAPS.

–with-globus-libdir=DIR Allows you to select an alternative directory for the Globus libraries. The default behaviour is to use the $PFX setting of the previous option and add /lib, e.g. $PFX/lib or $PFX/lib64

–with-globus-thr-flavor=FLAVOR Sets the threaded flavor of Globus. This is not needed in Globus 5 (and up) installation.

–with-globus-nothr-flavor=FLAVOR Sets the non-threaded flavor of Globus. This is not needed in Globus 5 (and up) installation.

5 Common used paths

Here is a small list of commonly used files and paths used in the context of LCAS and its plugins. It's important to know that these paths are not created by LCAS.

/etc/grid-security/grid-mapfile DN-based or VOMS FQAN-based auhtorization and mapping file used by various plugins. It maps DNs and/or FQANs to user accounts.

/etc/grid-security/vomapfile Meant to be exclusively used as VOMS FQAN-based auhtorization and mapping file used by VOMS-specific various plugins. It maps FQANs to user accounts.

/etc/grid-security/groupmapfile VOMS FQAN-based auhtorization and mapping file used by various VOMS plugins. It maps (secondary) Unix groups.

/etc/grid-security/vomsdir/ The VOMS directory will hold the VOMS .lsc files and/or PEM files to authenticate the VOMS Attributes Certificates. Subdirectories are named by the VO name and scope the .lsc and PEM files in their authentication to one particular VO.

/etc/grid-security/certificates/ The Certificate Authority (CA) directory filled with the supported CAs. It also contains the Certificate Revocation List (CRL) files and (Subject DN) signing namespace files of the CAs.

/etc/grid-security/gridmapdir/ The directory where all the user account mappings are held made by LCMAPS plugins that support the mapping of pool accounts. It will make a hardlink between a file entry representing the (Unix) account and the (encoded representation of the) certificate subject DN. The link is persistent and should be seen as a database that exposes the link between the Grid identity and the Unix identity.

/etc/grid-security/groupmapdir/ A similarly purposed directory, explicitly used by a VOMS-aware plugin that maps FQANs to a pool of (Unix) groups.

6 Control through environment variables

The library can be steered using environment variables used in the running process. It's important to understand that plugins can use an additional set of environment variables for plugins specific purposes. Feature overloading the following set of environment variables is not adviced.

GATEKEEPER_JM_ID Extra Gatekeeper log message to be able to more easily track a Job Manager ID.

GLOBUSID See $GATEKEEPER_JM_ID.

JOB_REPOSITORY_ID See $GATEKEEPER_JM_ID, but explicitly for the purpose of the LCMAPS Job Repository plugin.

LCAS_DB_FILE Override the build-in default filename for the *lcas.db* configuration file with the value of this environment variable.

LCAS_DEBUG_LEVEL Tune the logging output verbosity. For near silence, set the value to '1'. For very verbose output, set the value to '5' (which is the maximum).

LCAS_DIR The base directory of the $LCAS_DB_FILE parameter. This variable is concatinated with the $LCAS_DB_FILE

LCAS_ETC_DIR See $LCAS_DIR

LCAS_LOG_FILE Overrides the build-in default file path to log the output to. When set, the logging will not go to Syslog.

LCAS_LOG_STRING Prepend all log output messages with value of this environment variable

7 Configuration

The default configuration file for LCAS is *lcas.db*. This file defines the plug-in configurations and the order in which the plugins are to be executed. As it is impossible to provide a default configuration that a) makes sense and b) doesn't require at least one plug-in that is not strictly required in every case. A few commonly used configration will be explained. The system administrator should pick and choose which parts are needed. The selected plug-ins are provided in separate packages.

For more information, see lcas.db(5) and the documentation for each individual plugin, or go to `http://www.nikhef.nl/pub/projects/grid/gridwiki/index.php/Site_Access_Control`

7.1 GridFTPd example

In this example the LCAS Userban plugin is called to see if the subject is on the blacklist. If not, then continue to the authorization list based on VOMS credentials.

```
# LCAS database/plugin list
#
# Format of each line:
# pluginname="<name/path of plugin>", pluginargs="<arguments>"
```

```
#
pluginname=/opt/glite/lib64/modules/lcas_userban.mod,pluginargs=ban_users.db
pluginname=/opt/glite/lib64/modules/lcas_voms.mod,pluginargs="-vomsdir /etc/grid-security/vomsdir/
```

7.2 gLExec on CREAM CE example

Here is an example seen on a CREAM CE used by the gLExec component. It runs the VOMS credentials based authorization plugin first, and on success it will authorize the intended first executable based on the configured whitelist. The final plugin is intended to scope the usage of gLExeco only to these configured scripts and/or binaries.

```
pluginname=/opt/glite/lib64/modules/lcas_voms.mod,pluginargs="-vomsdir /etc/grid-security/vomsdir/
pluginname=/opt/glite/lib64/modules/lcas_check_executable.mod,pluginargs="-exec /bin/pwd:/bin/echo
```

8 Debugging LCAS

LCAS will typically be logging in the same location as the service or tool that is initiating the LCAS library and its plugins. The plugin log records are embedded in the LCAS main framework logrecords.

To start debugging a problem in LCAS, we recommend to look at the log file and increase the logging verbosity ultimately to level ”5” by using the *LCAS_LOG_LEVEL*. You can now see the plugins loading succesfully:

```
Initialization LCAS version 1.3.13
lcas.mod-lcas_init(): Reading LCAS database /etc/lcas/lcas-glexec.db
[...]
lcas.mod-lcas_db_fill_entry(): creating new list entry
lcas.mod-lcas_init(): initializing plugin lcas_userban.mod (db entry 0)
lcas.mod-PluginInit(): found "plugin_initialize()"
lcas.mod-PluginInit(): found "plugin_confirm_authorization()"
lcas.mod-get_procsymbol(): dlsym error: /usr/lib64/modules/lcas_userban.mod: undefined symbol: plu
lcas.mod-PluginInit(): creating first pluginlist entry
[...]
```

The start of an authorization request initiated using the LCAS PEM API:

```
LCAS authorization request
lcas.mod-lcas_run_va(): got input for LCAS_ARG_PEM execution
lcas.mod-lcas_run_va(): Extracting X509 Chain from PEM string
Got individual certificate with subject: /O=dutchgrid/O=users/O=nikhef/CN=Oscar Koeroo/CN=proxy
lcas.mod-lcas_run_va(): Parsing of PEM string succeeded
lcas.mod-lcas_run_va(): user is /O=dutchgrid/O=users/O=nikhef/CN=Oscar Koeroo
[...]
```

What follows is a series of sequences initiated by the plugins loaded by LCAS. More details can be found in the documentation for these plugins. When LCAS is finished it will state that it has terminated in the log:

```
Termination LCAS
```