Table of Contents

CREAM Service Reference Card

Daemons running

The following processes should run on a CREAM CE:

- tomcat (e.g. `/usr/lib/jvm/java/bin/java -server -Xms128m -Xmx512m -Dglite.log.path=/var/log/cream -Dcatalina.ext.dirs=/usr/share/tomcat5/shared/lib:/usr/share/tomcat5/commo -Djavax.sql.DataSource.Factory=org.apache.commons.dbcp.BasicDataSource`)
- blahpd (`/usr/bin/blahpd`). This is automatically started by starting tomcat
- resource BDII (`/usr/sbin/slapd -f /etc/bdii/bdii-slapd.conf -h ldap://0.0.0.0:2170 -u ldap`, `/usr/bin/python /usr/sbin/bdii-update -c /etc/bdii/bdii.conf -d`)
- gridftp server (`/usr/sbin/globus-gridftp-server`)
- LB locallogger (`/usr/bin/glite-lb-logd`, `/usr/bin/glite-lb-interlogd`)
- mysqld (`/usr/libexec/mysqld`). Actually mysqld could be deployed and run in another machine different than the CREAM CE)
- new blparser daemons (`/usr/bin/BNotifier`, `/usr/bin/BUpdaterxxx`) if the new BLAH blparser is used. They are automatically started by blahpd, which is started by tomcat
- old blparser daemon (xxx) if the old BLAH blparser is used. xxx runs where the batch system log files are available (which can be the CREAM CE node or a different node)

Init scripts and options (start|stop|restart|...)

- Init script for tomcat: `/etc/init.d/tomcat5 {start|stop|restart|condrestart|try-restart|reload|force-reload|status|ver`
- Init script for resource BDII: `/etc/init.d/bdii {start|stop|restart|condrestart|status}`
- Init script for gridftp server: `/etc/init.d/globus-gridftp {start|stop|restart|status}`
- Init script for LB locallogger: `/opt/glite/etc/init.d/glite-lb-locallogger {start,stop,restart,status}`
- Init script for mysql: `/etc/init.d/mysqld {start|stop|status|condrestart|restart}`
- Init script for old blparser: `/etc/init.d/glite-ce-blparser {start|stop|restart|status}`
- Init script for new blparser: `/etc/init.d/glite-ce-blahparser {start|stop|restart|status}`. Actually the new blparser is automatically started by blahpd, which is started by tomcat

Configuration files location with example or template

- CREAM configuration file (`/etc/glite-ce-cream/cream-config.xml`). This file is created by yaim-cream-ce. A template is installed as `/etc/glite-ce-cream/cream-config.xml.template`. An example of this configuration file is available here
- BLAH configuration file (`/etc/blah.config`). This file is created by yaim-cream-ce. A template is installed as `/etc/blah.config.template`. An example of this configuration file is available here
- BLparser configuration file (`/etc/blparser.conf`) only for the old blparser. This file is created by yaim-cream-ce. A template is installed as `/etc/blparser.conf.template`). An example of this configuration file is available xxx.
- glexec configuration file (`/etc/glexec.conf`). This file is created by yaim-cream-ce. An example of this configuration file is available here
- LCAS configuration file for glexec (`/etc/lcas/lcas-glexec.db`). This file is created by yaim-cream-ce. An example of this configuration file is available here
- LCMAPS configuration file for glexec (`/etc/lcmaps/lcmaps-glexec.db`). This file is created by yaim-cream-ce. An example of this configuration file is available here
- LCAS configuration file for gridftpd (`/etc/lcas/lcas.db`). This file is created by yaim-core (only when Argus is not used). An example of this configuration file is available xxx
- LCMAPS configuration file for gridftpd (`/etc/lcmaps/lcmaps.db`). This file is created by yaim-core (only when Argus is not used). An example of this configuration file is available xxx
- ARGUS configuration file for gridftpd (`xxx`) only when the CREAM CE is configured to use ARGUS. This file is created by yaim-cream-ce. An example of this configuration file is available here

Logfile locations (and management) and other useful audit information

The relevant log files are:

- The tomcat log file (`/usr/share/tomcat5/logs/catalina.out`)
- The trustmanager log file (`/usr/share/tomcat5/logs/trustmanager.log`)
- The CREAM log file (`/var/log/cream/glite-ce-cream.log`). The verbosity of this file can be increased modifying the file `/etc/glite-ce-cream/log4j.properties` replacing:

```
log4j.logger.org.glite=info, fileout
```

$: with:

```
log4j.logger.org.glite=debug, fileout
```

$: You may also change the attributes `log4j.appender.fileout.MaxFileSize` and `log4j.appender.fileout.MaxBackupIndex` to change the maximum file size and the maximum

number of log files to be kept.

- The glexec log files. glexec by default logs on syslog but it is also possible to log on file instead. Check the meaning of the variables `GLEXEC_CREAM_LOG_DESTINATION`, `GLEXEC_CREAM_LOG_FILE`, `GLEXEC_CREAM_LCASLCMAPS_LOG` in the yaim reference guide . The verbosity can be changed editing the glexec configuration file `/etc/glexec.conf`.

- The new BLAH blparser log files (`/var/log/cream/glite-ce-bnotifier.log` and `/var/log/cream/glite-ce-bupdater.log`) if the new blparser is used

* The gridftp log files (`globus-gridftp.log` and `gridftp-session.log`)

Open ports

Service	From node	From port	To node	To port	Other info
CREAM Service	{UI, WMS}	*	CREAM-CE	8443	
Gridftp control	{{UI, WMS, WN}	C	CREAM-CE	2811	
Gridftp data	{UI, WMS, WN}	C	CREAM-CE	C	
Notifications by BLparser and JobWrapper	{WN, Blparser host}	*	CREAM-CE	9091	Specified by `LRMS_EVENT_LISTENER_PORT` in CREAM conf file
CREAM job sensor	CEMon host	*	CREAM-CE	9909	Specified by `CREAM_JOB_SENSOR_PORT` in CREAM conf file. CEMON Host is usually the CREAM CE
LB locallogger	WN	C	CREAM-CE	9002	
LB locallogger	CREAM-CE	C	LB server	9001	
mysql	CREAM-CE	*	mysql server	3306	By default the mysql server is the CREAM CE
BDII	Site BDII	*	CREAM-CE	2170	
Old BLparser listening port	CREAM-CE	*	Blparser host	33333	Specified by yaim variable `BLP_PORT`
Old BLparser CREAM listening port	CREAM-CE	*	BLparser host	56565	Specified by yaim variable `CREAM_PORT`

C: Controllable Ephemeral range (e.g. 20000-25000). Note: In practice, although this port-range is locally configurable using the `GLOBUS_TCP_PORT_RANGE` environment variable, the values applying at a remote service cannot be predicted. Consequently reliable connection can only be established if all ports >1023 are left open for outbound connections.

Possible unit test of the service

TBD

Where is service state held (and can it be rebuilt)

CREAM job related information are kept in the CREAM DB and in the filesystem in the directory referred by `CREAM_SANDBOX_DIR` (default `/var/glite/cream_sandbox`) in the CREAM configuration file (`/etc/glite-ce-cream/cream-config.xml`).

Cron jobs

The cron jobs can be found in `/etc/cron.d` and are:

- `bdii-proxy`
- `fetch-crl`
- `lcg-expiregridmapdir`
- `cleanup-grid-accounts`
- `locallogger.cron`
- `glite-apel-lsf-parser` (just for LSF)

Security information

Access control Mechanism description (authentication & authorization)

Authentication

Authentication in CREAM is managed via the trustmanager.

The Trust Manager is the component responsible for carrying out authentication operations. It is an implementation of the J2EE security specifications. Authentication is based on PKI. Each user (and Grid service) wishing to access CREAM is required to present an X.509 format certificate. These certificates are issued by trusted entities, the Certificate Authorities (CA). The role of a CA is to guarantee the identity of a user. This is achieved by issuing an electronic document (the certificate) that contains the information about the user and is digitally signed by the CA with its private key. An authentication manager, such as the Trust Manager, can verify the user identity by decrypting the hash of the certificate with the CA public key. This ensures that the certificate was issued by that specific CA. The Trust Manager can then access the user data contained in the certificate and verify the user identity.

Authorization for the CREAM service

Authorization in the CREAM CE can be implemented in two different ways (the choice is done at configuration time):

- Authorization with ARGUS
- Authorization with gJAF

Argus is a system meant to render consistent authorization decisions for distributed services (e.g. compute elements, portals). In order to achieve this consistency a number of points must be addressed. First, it must be possible to author and maintain consistent authorization policies. This is handled by the Policy Administration Point (PAP) component in the service. Second, authored policies must be evaluated in a consistent manner, a task performed by the Policy Decision Point (PDP). Finally, the data provided for evaluation against policies must be consistent (in form and definition) and this is done by the Policy Enforcement Point (PEP). Argus is also responsible to manage the Grid user - local user mapping.

gJAF (Grid Java Authorization Framework) provides a way to invoke a chain of policy engines and get a decision result about the authorization of a user. The policy engines are divided in two types, depending on their functionality. They can be plugged into the framework in order to form a chain of policy engines as selected by the administrator in order to let him set up a complete authorization system. A policy engine may be either a PIP or a PDP. PIP collect and verify assertions and capabilities associated with the user, checking her role, group and VO attributes. PDP may use the information retrieved by a PIP to decide whether the user is allowed to perform the requested action, whether further evaluation is needed, or whether the evaluation should be interrupted and the user access denied. In CREAM CE VO based authorization is supported. In this

scenario, implemented via the VOMS PDP, the administrator can specify authorization policies based on the VO the jobs' owners belong to (or on particular VO attributes). When gJAF is used as authorization mechanism, the Grid user - local user mapping is managed via glexec,

For what concerns authorization on job operations, by default each user can manage (e.g. cancel, suspend, etc.) only her own jobs. However, the CREAM administrator can define specific super-users who are empowered to manage also jobs submitted by other users.

Authorization for gridftpd

When the CREAM CE is configured to use Argus, ARGUS manages also the authorization for gridftp server

If instead gJAF is used to manage the authorization, LCAS and LCMAPS are used to implement authorization for gridftp server.

How to block/ban a user

If ARGUS is used as authorization system, ARGUS can be used to ban users.

If instead gJAF is used, add the DN of the user to be banned in `/etc/lcas/ban_users.db`. Please note that the DN must be in quotes in this file.

Security recommendations

- It is recommended to close port 9091 (that is `LRMS_EVENT_LISTENER_PORT` in the CREAM conf file) to all nodes, expect the WNs and the one running the BLparser
- It is recommended to close port 9909 (that is `CREAM_JOB_SENSOR_PORT`) to all nodes except the one running CEMon (which by default is the CREAM_CE node)
- Use pool accounts instead of static accounts where possible
- Configure enough pool accounts so that recycling occur rarely . Releasing accounts should be an exception to the rule and should happen only then, if there's other way round. However, if this happens, it should be taken care of, that really all jobs of that account are finished
- Ensure that each VO software area is only writable by the software manager for that VO (group writable only for the "sgm" accounts group, not the VO)
- Do not mount the VO software areas on the CREAM CE node, but only on the WNs
- If there are more than one computing elements at a site, that access the same worker nodes, it`s recommended to set up individual, unique accounts for all CEs (such as "user001" ... "user100" on CE1 and "user101" ... "user200" on CE2 and so on) or to centrally mount gridmapdir, in order to guarantee, that no user is able to read the data of any other users.If ARGUS is used, gridmapdir must be centrally mounted.

Other security relevant comments

- A user can manage (check status, cancel, etc.) only her jobs
- It is possible to define admins for a CREAM CE. An admin of a CREAM CE can manage all (not only her own) jobs submitted to that CE and it can also disable/re-enable job submissions to that CREAM CE:

- An admin can decide to disable new job submissions (e.g. because that CE will have to be turned off)
  - This is done calling a proper operation (provided by the CREAM CLI as glite-ce-disable-submission)
  - When job submissions have been disabled, the other operations (e.g. job status, etc.) are still allowed
- An admin can then re-enable job submissions
  - This is done calling a proper operation (provided by the CREAM CLI as glite-ce-enable-submission)

- To specify an admin for a CREAM CE, just add her DN on /etc/grid-security/admin-list

-

-- MassimoSgaravatto - 2011-04-07

This topic: CREAM > ServiceReferenceCard
Topic revision: r7 - 2011-05-05 - MassimoSgaravatto