LHC COMPUTING GRID

LCG - WN_TORQUE - GENERIC CONFIGURATION REFERENCE

Document identifier:	**LCG-GIS-CR-WN_torque**
EDMS id:	**none**
Version:	
Date:	**January 16, 2006**
Section:	**LCG Grid Infrastructure Support**
Document status:	**ACTIVE**
Author(s):	LCG Deployment - GIS team;Retico,Antonio;Vidic,Valentin;
File:	**WN_torque**

Abstract: *Configuration steps done by the YAIM script 'configure_WN_torque'*

CONTENTS

1. INTRODUCTION

This document lists the manual steps for the installation and configuration of a LCG WN_torque Node. Furthermore it provides a specification of the YAIM functions used to configure the node with the script-based configuration.

The configuration has been tested on a standard Scientific Linux 3.0 Installation.

Link to this document:
This document is available on the *Grid Deployment* web site

`http://www.cern.ch/grid-deployment/gis/lcg-GCR/index.html`

2. VARIABLES

In order to set-up a WN_torque node, you need at least the following variables to be correctly configured in the site configuration file (site-info.def):

BDII_HOST : BDII Hostname.

CE_HOST : Computing Element Hostname.

DPM_HOST : Host name of the DPM host, used also as a default DPM for the lcg-stdout-mon .

EDG_WL_SCRATCH : Optional scratch directory for jobs.

EDG_WL_SCRATCH : Optional scratch directory for jobs.

FTS_SERVER_URL : URL of the File Transfer Service server.

GLOBUS_TCP_PORT_RANGE : Port range for Globus IO.

GRIDICE_SERVER_HOST : GridIce server host name (usually run on the MON node).

GSSKLOG : yes or no, indicating whether the site provides an AFS authentication server which maps gsi credentials into Kerberos tokens .

GSSKLOG_SERVER : If GSSKLOG is yes, the name of the AFS authentication server host.

INSTALL_ROOT : Installation root - change if using the re-locatable distribution.

JAVA_LOCATION : Path to Java VM installation. It can be used in order to run a different version of java installed locally.

JOB_MANAGER : The name of the job manager used by the gatekeeper.

MON_HOST : MON Box Hostname.

PX_HOST : PX hostname.

RB_HOST : Resource Broker Hostname.

REG_HOST : RGMA Registry hostname.

SE_LIST : A list of hostnames of the SEs available at your site.

SITE_NAME : Your GIIS.

TORQUE_SERVER : Set this if your torque server is on a different host from the CE. It is ingored for other batch systems.

USERS_CONF : Path to the file containing a list of Linux users (pool accounts) to be created. This file should be created by the Site Administrator, which contains a plain list of the users and IDs. An example of this configuration file is given in /opt/lcg/yaim/examples/users.conf.

VOBOX_HOST : VOBOX hostname.

VOBOX_PORT : The port the VOBOX gsisshd listens on.

VOS : List of supported VOs. For each item listed in the VOS variable you need to create a set of new variables as follows:

VO_<VO-NAME>_SE : Default SE used by the VO. WARNING: VO-NAME must be in capital cases.

VO_<VO-NAME>_SW_DIR : Area on the WN for the installation of the experiment software. If on the WNs a predefined shared area has been mounted where VO managers can pre-install software, then these variable should point to this area. If instead there is not a shared area and each job must install the software, then this variables should contain a dot (.).Anyway the mounting of shared areas, as well as the local installation of VO software is not managed by *yaim* and should be handled locally by Site Administrators. WARNING: VO-NAME must be in capital cases.

3. CONFIGURE LIBRARY PATHS

Author(s): Retico,Antonio
Email : support-lcg-manual-install@cern.ch

This chapter describes the configuration steps done by the *yaim* function *'config_ldconf'*.

In order to allow the middleware libraries to be looked up and dinamically linked, the relevant paths need to be configured.

- If not already there, append the following lines to the file */etc/ld.so.conf*

```
<INSTALL_ROOT>/globus/lib
<INSTALL_ROOT>/edg/lib
<INSTALL_ROOT>/lcg/lib
/usr/local/lib
/usr/kerberos/lib
/usr/X11R6/lib
/usr/lib/qt-3.1/lib
/opt/gcc-3.2.2/lib
```

  where <INSTALL_ROOT> is the installation root of the lcg middleware (*/opt* by default).

- Run the command:

```
> /sbin/ldconfig -v
```

  (this command produces a huge amount of output)

3.1. SPECIFICATION OF FUNCTION: CONFIG_LDCONF

The function *'config_ldconf'* needs the following variables to be set in the configuration file:

INSTALL_ROOT : Installation root - change if using the re-locatable distribution.

The original code of the function can be found in:

```
/opt/lcg/yaim/functions/config_ldconf
```

The code is reproduced also in 21.1..

4. SET-UP EDG CONFIGURATION VARIABLES

Author(s): Retico,Antonio
Email : support-lcg-manual-install@cern.ch

This chapter describes the configuration steps done by the *yaim* function *'config_sysconfig_edg'*.

The EDG configuration file is parsed by EDG daemons to locate the EDG root directory and various other global properties.

Create and edit the file */etc/sysconfig/edg* as follows:

```
EDG_LOCATION=<INSTALL_ROOT>/edg
EDG_LOCATION_VAR=<INSTALL_ROOT>/edg/var
EDG_TMP=/tmp
X509_USER_CERT=/etc/grid-security/hostcert.pem
X509_USER_KEY=/etc/grid-security/hostkey.pem
GRIDMAP=/etc/grid-security/grid-mapfile
GRIDMAPDIR=/etc/grid-security/gridmapdir/
```

where <INSTALL_ROOT> is the installation root of the lcg middleware (*/opt* by default).

NOTE: it might be observed that some of the variables above listed dealing with the GSI (Grid Security Interface) are needed just on service nodes (e.g. CE, RB) and not on others. Nevertheless, for sake of simplicity, *yaim* uses the same definitions on all node types, which has been proven not to hurt.

4.1. SPECIFICATION OF FUNCTION: CONFIG_SYSCONFIG_EDG

The function *'config_sysconfig_edg'* needs the following variables to be set in the configuration file:

INSTALL_ROOT : Installation root - change if using the re-locatable distribution.

The original code of the function can be found in:

```
/opt/lcg/yaim/functions/config_sysconfig_edg
```

The code is reproduced also in 21.2..

5. SET-UP GLOBUS CONFIGURATION VARIABLES

Author(s): Retico,Antonio
Email : support-lcg-manual-install@cern.ch

This chapter describes the configuration steps done by the *yaim* function '*config_sysconfig_globus*'.

Create and edit the file */etc/sysconfig/globus* as follows:

```
GLOBUS_LOCATION=<INSTALL_ROOT>/globus
GLOBUS_CONFIG=/etc/globus.conf
GLOBUS_TCP_PORT_RANGE="20000 25000"
export LANG=C
```

where <INSTALL_ROOT> is the installation root of the lcg middleware (*/opt* by default).

5.1. SPECIFICATION OF FUNCTION: CONFIG_SYSCONFIG_GLOBUS

The function *'config_sysconfig_globus'* needs the following variables to be set in the configuration file:

GLOBUS_TCP_PORT_RANGE : Port range for Globus IO.

INSTALL_ROOT : Installation root - change if using the re-locatable distribution.

The original code of the function can be found in:

```
/opt/lcg/yaim/functions/config_sysconfig_globus
```

The code is reproduced also in 21.3..

6. SET-UP LCG CONFIGURATION VARIABLES

Author(s): Retico,Antonio
Email : support-lcg-manual-install@cern.ch

This chapter describes the configuration steps done by the *yaim* function '*config_sysconfig_lcg*'.

Create and edit the file */etc/sysconfig/lcg* as follows:

```
LCG_LOCATION=<INSTALL_ROOT>/lcg
LCG_LOCATION_VAR=<INSTALL_ROOT>/lcg/var
LCG_TMP=/tmp
```

where <INSTALL_ROOT> is the installation root of the lcg middleware (*/opt* by default).

6.1. SPECIFICATION OF FUNCTION: CONFIG_SYSCONFIG_LCG

The function *'config_sysconfig_lcg'* needs the following variables to be set in the configuration file:

INSTALL_ROOT : Installation root - change if using the re-locatable distribution.

SITE_NAME : Your GIIS.

The original code of the function can be found in:

```
/opt/lcg/yaim/functions/config_sysconfig_lcg
```

The code is reproduced also in 21.4..

7. SET-UP UPDATING OF CRLS

Author(s): Vidic,Valentin
Email : support-lcg-manual-install@cern.ch

This chapter describes the configuration steps done by the *yaim* function '*config_crl*'.

Cron script is installed to fetch new versions of CRLs four times a day. The time when the script is run is randomized in order to distribute the load on CRL servers. If the configuration is run as root, the cron entry is installed in */etc/cron.d/edg-fetch-crl*, otherwise it is installed as a user cron entry.

CRLs are also updated immediately by running the update script (*<INSTALL_ROOT>/edg/etc/cron/edg-fetch-crl-cron*).

Logrotate script is installed as */etc/logrotate.d/edg-fetch-crl* to prevent the logs from growing indefinitely.

7.1. SPECIFICATION OF FUNCTION: CONFIG_CRL

The function *'config_crl'* needs the following variables to be set in the configuration file:

INSTALL_ROOT : Installation root - change if using the re-locatable distribution.

The original code of the function can be found in:

```
/opt/lcg/yaim/functions/config_crl
```

The code is reproduced also in 21.5..

8. SET-UP RFIO

Author(s): Vidic,Valentin
Email : support-lcg-manual-install@cern.ch

This chapter describes the configuration steps done by the *yaim* function '*config_rfio*'.

rfiod is configured on SE_classic nodes by adding the appropriate ports (5001 TCP and UDP) to */etc/services* and restarting the daemon.

For SE_dpm nodes, *rfiod* is configured by *config_DPM_rfio* so no configuration is done here.

All other nodes don't run *rfiod*. However, *rfiod* might still be installed from *CASTOR-client* RPM. If this is the case, we make sure it's stopped and disabled.

8.1. SPECIFICATION OF FUNCTION: CONFIG_RFIO

The function *'config_rfio'* needs the following variables to be set in the configuration file:

INSTALL_ROOT : Installation root - change if using the re-locatable distribution.

The original code of the function can be found in:

```
/opt/lcg/yaim/functions/config_rfio
```

The code is reproduced also in 21.6..

9. SET-UP GLOBUS DAEMONS

Author(s): Vidic,Valentin
Email : support-lcg-manual-install@cern.ch

This chapter describes the configuration steps done by the *yaim* function '*config_globus*'.

The Globus configuration file */etc/globus.conf* is parsed by Globus daemon startup scripts to locate the Globus root directory and other global/daemon specific properties. The contents of the configuration file depend on the type of the node. The following table contains information on daemon to node mapping:

node/daemon	MDS	GridFTP	Gatekeeper
CE	yes	yes	yes
VOBOX	yes	yes	yes
SE_*	yes	yes	no
SE_dpm	yes	no	no
PX	yes	no	no
RB	yes	no	no
LFC	yes	no	no
GridICE	yes	no	no

Note that SE_dpm does not run standard GridFTP server, but a specialized DPM version.

The configuration file is divided into sections:

common Defines Globus installation directory, host certificates, location of gridmap file etc.

mds Defines information providers.

gridftp Defines the location of the GridFTP log file.

gatekeeper Defines jobmanagers and their parameters.

Logrotate scripts *globus-gatekeeper* and *gridftp* are installed in */etc/logrotate.d/*.

Globus initialization script (*<INSTALL_DIR>/globus/sbin/globus-initialization.sh*) is run next.

Finally, the appropriate daemons (*globus-mds*, *globus-gatekeeper*, *globus-gridftp*, *lcg-mon-gridftp*) are started (and configured to start on boot).

9.1. SPECIFICATION OF FUNCTION: CONFIG_GLOBUS

The function *'config_globus'* needs the following variables to be set in the configuration file:

CE_HOST : Computing Element Hostname.

GRIDICE_SERVER_HOST : GridIce server host name (usually run on the MON node).

INSTALL_ROOT : Installation root - change if using the re-locatable distribution.

JOB_MANAGER : The name of the job manager used by the gatekeeper.

PX_HOST : PX hostname.

RB_HOST : Resource Broker Hostname.

SITE_NAME : Your GIIS.

The original code of the function can be found in:

```
/opt/lcg/yaim/functions/config_globus
```

The code is reproduced also in 21.7..

10. SET-UP LCG ENVIRONMENT VARIABLES

Author(s): Retico,Antonio
Email : support-lcg-manual-install@cern.ch

This chapter describes the configuration steps done by the *yaim* function '*config_lcgenv*'.

The LCG middleware needs some environment variables to be set up at boot time. The variable should be available both in 'bash-like' shells and in 'csh-like' shells.

This can be obtained in different ways:

The simplest way, if you have 'root' permissions, is to put a shell script for each of the supported shell 'families' in the directory */etc/profile.d*. The script will be automatically sourced at start up.

If instead you are not a superuser and you are doing the installation in a private directory (e.g. you are installing a Re-locatable Distribution of a Worker Node or a User Interface in the *<INSTALL_ROOT>* directory), you could create the scripts in the directory *<INSTALL_ROOT>/etc/profile.d*, in order to have the variables automatically set up by LCG tools.

The list of the environment variables to be set up follows:

LCG_GFAL_INFOSYS: Hostname of the BDII node.

MYPROXY_SERVER: Hostname of the Proxy server.

PATH: Add to the PATH variable the path */opt/d-cache-client/bin*

LD_LIBRARY_PATH: Add to the LD_LIBRARY_PATH variable the path */opt/d-cache-client/dcap*

SRM_PATH: Installation directory of the srm client. The default value for this variable is */opt/d-cache-client/srm*

VO_<VO-NAME>_SW_DIR: For each virtual organization <VO-NAME> An environment variable VO_<VO-NAME>_SW_DIR is needed. This variable points to the installation directory of the VO software.

VO_<VO-NAME>_DEFAULT_SE: For each virtual organization <VO-NAME> An environment variable VO_<VO-NAME>_DEFAULT_SE is needed. This variable points to the Default Storage Element for that VO.

The examples given hereafter refer to the simple configuration method described above. In the following description we will refer to the two possible locations as to the <LCG_ENV_LOC>.
So, according to the cases above described, either

<LCG_ENV_LOC>=/etc/profile.d

or

<LCG_ENV_LOC>=<INSTALL_ROOT>/etc/profile.d

Examples:

- Example of file *<LCG_ENV_LOC>/lcgenv.sh*:

```
#!/bin/sh
export LCG_GFAL_INFOSYS=lxb1769.cern.ch:2170
export MYPROXY_SERVER=lxb1774.cern.ch
export PATH="${PATH}:/opt/d-cache-client/bin"
export LD_LIBRARY_PATH=${LD_LIBRARY_PATH}:/opt/d-cache-client/dcap
export SRM_PATH=/opt/d-cache-client/srm
export VO_ATLAS_SW_DIR=lxb1780.cern.ch
export VO_ATLAS_DEFAULT_SE=lxb1780.cern.ch
export VO_ALICE_SW_DIR=lxb1780.cern.ch
export VO_ALICE_DEFAULT_SE=lxb1780.cern.ch
export VO_LHCB_SW_DIR=lxb1780.cern.ch
export VO_LHCB_DEFAULT_SE=lxb1780.cern.ch
export VO_CMS_SW_DIR=lxb1780.cern.ch
export VO_CMS_DEFAULT_SE=lxb1780.cern.ch
export VO_DTEAM_SW_DIR=lxb1780.cern.ch
export VO_DTEAM_DEFAULT_SE=lxb1780.cern.ch
```

- Example of file *<LCG_ENV_LOC>/lcgenv.csh*:

```
#!/bin/csh
setenv LCG_GFAL_INFOSYS lxb1769.cern.ch:2170
setenv MYPROXY_SERVER lxb1774.cern.ch
setenv PATH "${PATH}:/opt/d-cache-client/bin"
setenv LD_LIBRARY_PATH ${LD_LIBRARY_PATH}:/opt/d-cache-client/dcap
setenv SRM_PATH /opt/d-cache-client/srm
setenv VO_ATLAS_SW_DIR lxb1780.cern.ch
setenv VO_ATLAS_DEFAULT_SE lxb1780.cern.ch
setenv VO_ALICE_SW_DIR lxb1780.cern.ch
setenv VO_ALICE_DEFAULT_SE lxb1780.cern.ch
setenv VO_LHCB_SW_DIR lxb1780.cern.ch
setenv VO_LHCB_DEFAULT_SE lxb1780.cern.ch
setenv VO_CMS_SW_DIR lxb1780.cern.ch
setenv VO_CMS_DEFAULT_SE lxb1780.cern.ch
setenv VO_DTEAM_SW_DIR lxb1780.cern.ch
setenv VO_DTEAM_DEFAULT_SE lxb1780.cern.ch
```

WARNING: The two scripts must be executable by all users.

```
> chmod a+x ${LCG_ENV_LOC}/lcgenv.csh
> chmod a+x ${LCG_ENV_LOC}/lcgenv.sh
```

10.1. SPECIFICATION OF FUNCTION: CONFIG_LCGENV

The function *'config_lcgenv'* needs the following variables to be set in the configuration file:

BDII_HOST : BDII Hostname.

CE_HOST : Computing Element Hostname.

DPM_HOST : Host name of the DPM host, used also as a default DPM for the lcg-stdout-mon .

EDG_WL_SCRATCH : Set this if you want jobs to use a particular scratch area.

EDG_WL_SCRATCH : Set this if you want jobs to use a particular scratch area.

GLOBUS_TCP_PORT_RANGE : Port range for Globus IO.

GSSKLOG : yes or no, indicating whether the site provides an AFS authentication server which maps gsi credentials into Kerberos tokens .

GSSKLOG_SERVER : If GSSKLOG is yes, the name of the AFS authentication server host.

INSTALL_ROOT : Installation root - change if using the re-locatable distribution.

PX_HOST : PX hostname.

SE_LIST : A list of hostnames of the SEs available at your site.

SITE_NAME : Your GIIS.

VOBOX_HOST : VOBOX hostname.

VOS : List of supported VOs. For each item listed in the VOS variable you need to create a set of new variables as follows:

VO_<VO-NAME>_SE : Default SE used by the VO. WARNING: VO-NAME must be in capital cases.

VO_<VO-NAME>_SW_DIR : Area on the WN for the installation of the experiment software. If on the WNs a predefined shared area has been mounted where VO managers can pre-install software, then these variable should point to this area. If instead there is not a shared area and each job must install the software, then this variables should contain a dot (.).Anyway the mounting of shared areas, as well as the local installation of VO software is not managed by *yaim* and should be handled locally by Site Administrators. WARNING: VO-NAME must be in capital cases.

The function does exit with return code 1 if they are not set.

The original code of the function can be found in:

```
/opt/lcg/yaim/functions/config_lcgenv
```

The code is reproduced also in 21.8..

11. SET-UP REPLICA MANAGER

Author(s): Vidic,Valentin
Email : support-lcg-manual-install@cern.ch

This chapter describes the configuration steps done by the *yaim* function '*config_replica_manager*'.

Variable substitutions are generated in *<INSTALL_ROOT>/edg/etc/edg-replica-manager/edg-replica-manager.conf.values_local*:

```
@EDG.LOCATION@|<INSTALL_ROOT>/edg|location of edg middleware
@LOCALDOMAIN@|<domain>|the local domain
@DEFAULT.SE@|<SE_HOST>|the host of the close SE
@DEFAULT.CE@|<CE_HOST>|the host of the close CE
@INFOSERVICE@|MDS|The info provider to use. It can be Stub, MDS or RGMA
@RLS.MODE@|LrcOnly|The mode the RLS should be run in. LrcOnly or WithRli
@STUBFILE@||The properties file for the static file - only needed in Stub mode
@MDS.HOST@|<BDII_HOST>|The host of the MDS info provider
@MDS.PORT@|2170|The port of the MDS info provider
@ROS.FAILURE@|false|Fail if no ROS is available
@CONF.GCC@|_gcc3_2_2|The gcc suffix as used on the build box (empty for 2.95, _gcc3_2_2 for 3.2.)
@IGNORE.PREFIX@|true|Whether the RM will ignore the lfn and guid prefix.
@GRIDFTP.DCAU@|false|Does GridFTP use Data Channel Authentication (DCAU)
@GRIDFTP.STREAMS.SMALL@|1|The default number of stream to use for a small file
@GRIDFTP.STREAMS.BIG@|3|The default number of stream to use for a big file
@GRIDFTP.FILESIZE.THRESHOLD@|100|The Threshold (in MB) above which a file to transfer is considered "big"
```

The value of <domain> is determined by running *hostname -d*. Using these substitutions and templates in *<INSTALL_ROOT>/edg/etc/edg-replica-manager/*, Replica Manager is configured by generating files in *<EDG_LOCATION>/var/etc/edg-replica-manager*:

```
<INSTALL_ROOT>/edg/sbin/edg-replica-manager-configure <INSTALL_ROOT>/edg/etc/edg-replica-manager/edg-replica-manage
```

11.1. SPECIFICATION OF FUNCTION: CONFIG_REPLICA_MANAGER

The function *'config_replica_manager'* needs the following variables to be set in the configuration file:

BDII_HOST : BDII Hostname.

CE_HOST : Computing Element Hostname.

INSTALL_ROOT : Installation root - change if using the re-locatable distribution.

SE_LIST : A list of hostnames of the SEs available at your site.

The original code of the function can be found in:

```
/opt/lcg/yaim/functions/config_replica_manager
```

The code is also reproduced in 21.9..

12. CREATE POOL ACCOUNTS

Author(s): Vidic,Valentin
Email : support-lcg-manual-install@cern.ch

This chapter describes the configuration steps done by the *yaim* function *'config_users'*.

config_users creates pool accounts for grid users defined in *users.conf*. Each line in this file describes one user:

```
UID:LOGIN:GID:GROUP:VO:SGM_FLAG:
```

First, the format of the *users.conf* file is checked (VO and SGM fields were added recently).

Groups are then created for the supported VOs (listed in <*VOS*> variable) using *groupadd*.

For each of the lines in *users.conf*, a user account is created (with *useradd*) if that user's VO is supported.

Finally, grid users are denied access to *cron* and *at* by adding their usernames to */etc/at.deny* and */etc/cron.deny*.

12.1. SPECIFICATION OF FUNCTION: CONFIG_USERS

The function *'config_users'* needs the following variables to be set in the configuration file:

INSTALL_ROOT : Installation root - change if using the re-locatable distribution.

USERS_CONF : Path to the file containing a list of Linux users (pool accounts) to be created. This file should be created by the Site Administrator, which contains a plain list of the users and IDs. An example of this configuration file is given in /opt/lcg/yaim/examples/users.conf.

VOS : List of supported VOs.

The original code of the function can be found in:

```
/opt/lcg/yaim/functions/config_users
```

The code is reproduced also in 21.10..

13. CREATE EXPERIMENT SOFTWARE DIRECTORIES

Author(s): Vidic,Valentin
Email : support-lcg-manual-install@cern.ch

This chapter describes the configuration steps done by the *yaim* function '*config_sw_dir*'.

For each of the supported VOs, experiment software directory is created as specified by <VO_<vo>_SW_DIR> variable.

Ownership of the created directory is changed to VO's SGM user and group.

It is assumed that <VO_<vo>_SW_DIR> is NFS mounted so that running this function on a WN will create software directories for the whole site.

13.1. SPECIFICATION OF FUNCTION: CONFIG_SW_DIR

The function *'config_sw_dir'* needs the following variables to be set in the configuration file:

VOS : List of supported VOs. For each item listed in the VOS variable you need to create a set of new variables as follows:

VO_<VO-NAME>_SW_DIR : Area on the WN for the installation of the experiment software. If on the WNs a predefined shared area has been mounted where VO managers can pre-install software, then these variable should point to this area. If instead there is not a shared area and each job must install the software, then this variables should contain a dot (.).Anyway the mounting of shared areas, as well as the local installation of VO software is not managed by *yaim* and should be handled locally by Site Administrators. WARNING: VO-NAME must be in capital cases.

The original code of the function can be found in:

```
/opt/lcg/yaim/functions/config_sw_dir
```

The code is also reproduced in 21.11..

14. SET-UP JAVA LOCATION

Author(s): Vidic,Valentin
Email : support-lcg-manual-install@cern.ch

This chapter describes the configuration steps done by the *yaim* function '*config_java*'.

Since Java is not included in the LCG distribution, Java location needs to be configured with yaim.

If <JAVA_LOCATION> is not defined in *site-info.def*, it is determined from installed Java RPMs (if available).

In relocatable distribution, JAVA_HOME environment variable is defined in *<INSTALL_ROOT>/etc/profile.d/grid_en* and *<INSTALL_ROOT>/etc/profile.d/grid_env.csh*.

Otherwise, JAVA_HOME is defined in */etc/java/java.conf* and */etc/java.conf* and Java binaries added to PATH in *<INSTALL_ROOT>/edg/etc/profile.d/j2.sh* and *<INSTALL_ROOT>/edg/etc/profile.d/j2.csh*.

14.1. SPECIFICATION OF FUNCTION: CONFIG_JAVA

The function *'config_java'* needs the following variables to be set in the configuration file:

INSTALL_ROOT : Installation root - change if using the re-locatable distribution.

JAVA_LOCATION : Path to Java VM installation. It can be used in order to run a different version of java installed locally.

The original code of the function can be found in:

```
/opt/lcg/yaim/functions/config_java
```

The code is reproduced also in 21.12..

15. SET-UP R-GMA CLIENT

Author(s): Vidic,Valentin
Email : support-lcg-manual-install@cern.ch

This chapter describes the configuration steps done by the *yaim* function '*config_rgma_client*'.

R-GMA client configuration is generated in *<INSTALL_ROOT>/glite/etc/rgma/rgma.conf* by running:

```
<INSTALL_ROOT>/glite/share/rgma/scripts/rgma-setup.py --secure=no --server=<MON_HOST> --registry=<REG_HOST> --schem
```

<INSTALL_ROOT>/edg/etc/profile.d/edg-rgma-env.sh and *<INSTALL_ROOT>/edg/etc/profile.d/edg-rgma-env.csh* with the following functionality:

- RGME_HOME is set to *<INSTALL_ROOT>/glite*
- APEL_HOME is set to *<INSTALL_ROOT>/glite*
- *<INSTALL_ROOT>/glite/lib/python* is added to PYTHONPATH
- *<INSTALL_ROOT>/glite/lib* is added to LD_LIBRARY_PATH.

These files are sourced into the users environment from *<INSTALL_ROOT>/etc/profile.d/z_edg_profile.sh* and *<INSTALL_ROOT>/etc/profile.d/z_edg_profile.csh*.

15.1. SPECIFICATION OF FUNCTION: CONFIG_RGMA_CLIENT

The function *'config_rgma_client'* needs the following variables to be set in the configuration file:

INSTALL_ROOT : Installation root - change if using the re-locatable distribution.

MON_HOST : MON Box Hostname.

REG_HOST : RGMA Registry hostname.

The original code of the function can be found in:

```
/opt/lcg/yaim/functions/config_rgma_client
```

The code is also reproduced in 21.13..

16. SET-UP WORKLOAD MANAGER ENVIRONMENT

Author(s): Vidic,Valentin
Email : support-lcg-manual-install@cern.ch

This chapter describes the configuration steps done by the *yaim* function '*config_workload_manager_env*'.

<INSTALL_ROOT>/edg/etc/profile.d/edg-wl.csh and *<INSTALL_ROOT>/edg/etc/profile.d/edg-wl.sh* are copied to *<INSTALL_ROOT>/edg/var/etc/profile.d/*.

They are sourced into the user environment upon login (*/etc/profile.d/z_edg-profile.(c)sh*).

16.1. SPECIFICATION OF FUNCTION: CONFIG_WORKLOAD_MANAGER_ENV

The function *'config_workload_manager_env'* needs the following variables to be set in the configuration file:

INSTALL_ROOT : Installation root - change if using the re-locatable distribution.

The original code of the function can be found in:

```
/opt/lcg/yaim/functions/config_workload_manager_env
```

The code is also reproduced in 21.14..

17. SET-UP FILE TRANSFER SERVICE CLIENT

Author(s): Vidic,Valentin
Email : support-lcg-manual-install@cern.ch

This chapter describes the configuration steps done by the *yaim* function '*config_fts_client*'.

If <FTS_SERVER_URL> is set, gLite FTS client is configured by creating <*INSTALL_ROOT*>*/glite/etc/services.xml*

```
<services>

  <service name="EGEEfts">
    <parameters>
      <endpoint><FTS_SERVER_URL>/services/FileTransfer</endpoint>
      <type>org.glite.FileTransfer</type>
      <version>3.0.0</version>
    </parameters>
  </service>

  <service name="EGEEchannel">
    <parameters>
      <endpoint><FTS_SERVER_URL>/services/ChannelManagement</endpoint>
      <type>org.glite.ChannelManagement</type>
      <version>3.0.0</version>
    </parameters>
  </service>

</services>
```

17.1. SPECIFICATION OF FUNCTION: CONFIG_FTS_CLIENT

The function *'config_fts_client'* needs the following variables to be set in the configuration file:

FTS_SERVER_URL : URL of the File Transfer Service server.

INSTALL_ROOT : Installation root - change if using the re-locatable distribution.

The original code of the function can be found in:

```
/opt/lcg/yaim/functions/config_fts_client
```

The code is also reproduced in 21.15..

18. SET-UP GLITE ENVIRONMENT

Author(s): Vidic,Valentin
Email : support-lcg-manual-install@cern.ch

This chapter describes the configuration steps done by the *yaim* function '*config_glite_env*'.

/etc/profile.d/gliteenv.sh and */etc/profile.d/gliteenv.csh* are created. These scripts set enviroment variables needed to run gLite programs, for example */etc/profile.d/gliteenv.sh*:

```
if test "x${LCG_ENV_SET+x}" = x; then

    GLITE_LOCATION=${GLITE_LOCATION:-/opt/glite}
    GLITE_LOCATION_VAR=${GLITE_LOCATION_VAR:-$GLITE_LOCATION/var}
    GLITE_LOCATION_LOG=${GLITE_LOCATION_LOG:-$GLITE_LOCATION/log}
    GLITE_LOCATION_TMP=${GLITE_LOCATION_TMP:-$GLITE_LOCATION/tmp}

    if [ -z "$PATH" ]; then
        PATH="${GLITE_LOCATION}/bin:${GLITE_LOCATION}/externals/bin"
    else
        PATH="${PATH}:${GLITE_LOCATION}/bin:${GLITE_LOCATION}/externals/bin"
    fi

    if [ -z "$LD_LIBRARY_PATH" ]; then
        LD_LIBRARY_PATH="${GLITE_LOCATION}/lib:${GLITE_LOCATION}/externals/lib"
    else
        LD_LIBRARY_PATH="${LD_LIBRARY_PATH}:${GLITE_LOCATION}/lib:${GLITE_LOCATION}/externals/lib"
    fi

    if [ -z "$PERLLIB" ]; then
        PERLLIB="${GLITE_LOCATION}/lib/perl5"
    else
        PERLLIB="${PERLLIB}:${GLITE_LOCATION}/lib/perl5"
    fi

    if [ -z "$MANPATH" ]; then
        MANPATH="${GLITE_LOCATION}/share/man"
    else
        MANPATH="${MANPATH}:${GLITE_LOCATION}/share/man"
    fi

    export GLITE_LOCATION GLITE_LOCATION_VAR GLITE_LOCATION_LOG GLITE_LOCATION_TMP PATH LD_LIBRARY_PATH PERLLIB MAN

fi
```

/etc/profile.d/gliteenv.csh has the same functionality but for CSH compatible shells.

18.1. SPECIFICATION OF FUNCTION: CONFIG_GLITE_ENV

The function *'config_glite_env'* needs the following variables to be set in the configuration file:

INSTALL_ROOT : Installation root - change if using the re-locatable distribution.

The original code of the function can be found in:

```
/opt/lcg/yaim/functions/config_glite_env
```

The code is also reproduced in 21.16..

19. SET-UP GSI ENABLED SSH

Author(s): Vidic,Valentin
Email : support-lcg-manual-install@cern.ch

This chapter describes the configuration steps done by the *yaim* function '*config_gsissh*'.

First, the generic configuration script (*<INSTALL_ROOT>/globus/setup/gsi_openssh_setup/setup-openssh*) is run (if it exists). It will generate host keys and a default configuration.

At the moment, further configuration is only done for *VOBOX* nodes.

Configuration files for SSH daemon and client are modified to use GSI authentication. All other authentication types and root login are disabled for the daemon.

The startup script is modified to make the *gsisshd* listen on a nondefault port (<VOBOX_PORT>). It is than installed as */etc/init.d/gsisshd*. The *gsisshd* daemon is restarted and configured to start on boot.

19.1. SPECIFICATION OF FUNCTION: CONFIG_GSISSH

The function *'config_gsissh'* needs the following variables to be set in the configuration file:

INSTALL_ROOT : Installation root - change if using the re-locatable distribution.

VOBOX_PORT : The port the VOBOX gsisshd listens on.

The original code of the function can be found in:

```
/opt/lcg/yaim/functions/config_gsissh
```

The code is also reproduced in 21.17..

20. SET-UP TORQUE NODE

Author(s): Vidic,Valentin
Email : support-lcg-manual-install@cern.ch

This chapter describes the configuration steps done by the *yaim* function '*config_torque_client*'.

/var/spool/pbs/server_name is initialized with the hostname where the Torque server is running (usually CE, otherwise set in <TORQUE_SERVER>). This file is used by Torque utils for contacting the server.

Torque mom ports are defined by adding the following lines to */etc/services*:

```
pbs_mom    15002/tcp
pbs_resmon 15003/tcp
pbs_resmon 15003/udp
```

SSH client is configured (via */etc/ssh/ssh_config*) to use Hostbased authentication:

```
Host *
   Protocol 2,1
   RhostsAuthentication yes
   RhostsRSAAuthentication yes
   RSAAuthentication yes
   PasswordAuthentication yes
   EnableSSHKeysign yes
   HostbasedAuthentication yes
```

<INSTALL_ROOT>/edg/etc/edg-pbs-knownhosts.conf is created with the following contents:

```
NODES = <CE_HOST> <SE_HOST>
PBSBIN = /usr/bin
KEYTYPES = rsa1,rsa,dsa
KNOWNHOSTS = /etc/ssh/ssh_known_hosts
```

edg-pbs-knownhosts is run once to initialize */etc/ssh/ssh_known_hosts* and configured to run from cron four times a day. *edg-pbs-knownhosts* uses *pbsnodes* and this config file to acquire the list of nodes. For every node not already present in */etc/ssh/ssh_known_hosts*, host keys are discovered using *ssh-keyscan* and appended to */etc/ssh/ssh_known_hosts*.

Torque mom is configured by creating */var/spool/pbs/mom_priv/config*:

```
$clienthost <TORQUE_SERVER>
$clienthost localhost
$restricted <TORQUE_SERVER>
$logevent 255
$ideal_load 1.6
$max_load 2.1
```

Torque mom is restarted with the new configuration and configured to start on boot.

As Torque mom logs (*/var/spool/pbs/mom_logs*) can get quite big, a cron job is installed to compress them once a day.

20.1. SPECIFICATION OF FUNCTION: CONFIG_TORQUE_CLIENT

The function *'config_torque_client'* needs the following variables to be set in the configuration file:

CE_HOST : Computing Element Hostname.

SE_LIST : A list of hostnames of the SEs available at your site.

TORQUE_SERVER : Set this if your torque server is on a different host from the CE. It is ingored for other batch systems.

The original code of the function can be found in:

```
/opt/lcg/yaim/functions/config_torque_client
```

The code is also reproduced in 21.18..

21. SOURCE CODE

21.1. CONFIG_LDCONF

```
config_ldconf () {

    INSTALL_ROOT=${INSTALL_ROOT:-/opt}

cp -p /etc/ld.so.conf /etc/ld.so.conf.orig

    LIBDIRS="${INSTALL_ROOT}/globus/lib \
     ${INSTALL_ROOT}/edg/lib \
             ${INSTALL_ROOT}/edg/externals/lib/ \
     /usr/local/lib \
             ${INSTALL_ROOT}/lcg/lib \
             /usr/kerberos/lib \
             /usr/X11R6/lib \
             /usr/lib/qt-3.1/lib \
             ${INSTALL_ROOT}/gcc-3.2.2/lib \
             ${INSTALL_ROOT}/glite/lib \
             ${INSTALL_ROOT}/glite/externals/lib"


    if [ -f /etc/ld.so.conf.add ]; then
rm -f /etc/ld.so.conf.add
    fi

    for libdir in ${LIBDIRS}; do
if  ( ! grep -q $libdir /etc/ld.so.conf && [ -d $libdir ] ); then
    echo $libdir >> /etc/ld.so.conf.add
fi
    done

    if [ -f /etc/ld.so.conf.add ]; then
sort -u /etc/ld.so.conf.add >> /etc/ld.so.conf
rm -f  /etc/ld.so.conf.add
    fi

    /sbin/ldconfig

    return 0
}
```

21.2. CONFIG_SYSCONFIG_EDG

```
config_sysconfig_edg(){

INSTALL_ROOT=${INSTALL_ROOT:-/opt}

cat <<EOF > /etc/sysconfig/edg
EDG_LOCATION=$INSTALL_ROOT/edg
```

```
EDG_LOCATION_VAR=$INSTALL_ROOT/edg/var
EDG_TMP=/tmp
X509_USER_CERT=/etc/grid-security/hostcert.pem
X509_USER_KEY=/etc/grid-security/hostkey.pem
GRIDMAP=/etc/grid-security/grid-mapfile
GRIDMAPDIR=/etc/grid-security/gridmapdir/
EDG_WL_BKSERVERD_ADDOPTS=--rgmaexport
EDG_WL_RGMA_FILE=/var/edgwl/logging/status.log
EOF

return 0
}
```

21.3. CONFIG_SYSCONFIG_GLOBUS

```
config_sysconfig_globus() {

INSTALL_ROOT=${INSTALL_ROOT:-/opt}

# If GLOBUS_TCP_PORT_RANGE is unset, give it a good default
# Leave it alone if it is set but empty
GLOBUS_TCP_PORT_RANGE=${GLOBUS_TCP_PORT_RANGE-"20000 25000"}

cat <<EOF > /etc/sysconfig/globus
GLOBUS_LOCATION=$INSTALL_ROOT/globus
GLOBUS_CONFIG=/etc/globus.conf
export LANG=C
EOF

# Set GLOBUS_TCP_PORT_RANGE, but not for nodes which are only WNs
if [ "$GLOBUS_TCP_PORT_RANGE" ] && ( ! echo $NODE_TYPE_LIST | egrep -q '^ *WN_?[[:alpha:]]* *$' ); then
    echo "GLOBUS_TCP_PORT_RANGE=\"$GLOBUS_TCP_PORT_RANGE\"" >> /etc/sysconfig/globus
fi

(
    # HACK to avoid complaints from services that do not need it,
    # but get started via a login shell before the file is created...

    f=$INSTALL_ROOT/globus/libexec/globus-script-initializer
    echo '' > $f
    chmod 755 $f
)

return 0
}
```

21.4. CONFIG_SYSCONFIG_LCG

```
config_sysconfig_lcg(){

INSTALL_ROOT=${INSTALL_ROOT:-/opt}
```

```
cat <<EOF > /etc/sysconfig/lcg
LCG_LOCATION=$INSTALL_ROOT/lcg
LCG_LOCATION_VAR=$INSTALL_ROOT/lcg/var
LCG_TMP=/tmp
export SITE_NAME=$SITE_NAME
EOF

return 0
}
```

21.5. CONFIG_CRL

```
config_crl(){

INSTALL_ROOT=${INSTALL_ROOT:-/opt}

let minute="$RANDOM%60"

let h1="$RANDOM%24"
let h2="($h1+6)%24"
let h3="($h1+12)%24"
let h4="($h1+18)%24"

if !( echo "${NODE_TYPE_LIST}" | grep TAR > /dev/null ); then

    if [ ! -f /etc/cron.d/edg-fetch-crl ]; then
echo "Now updating the CRLs - this may take a few minutes..."
$INSTALL_ROOT/edg/etc/cron/edg-fetch-crl-cron >> /var/log/edg-fetch-crl-cron.log 2>&1
    fi

cron_job edg-fetch-crl root "$minute $h1,$h2,$h3,$h4 * * * $INSTALL_ROOT/edg/etc/cron/edg-fetch-crl-cron >> /var/lc

    cat <<EOF > /etc/logrotate.d/edg-fetch
/var/log/edg-fetch-crl-cron.log  {
    compress
    monthly
    rotate 12
    missingok
    ifempty
    create
}
EOF

else

    cron_job edg-fetch-crl `whoami` "$minute $h1,$h2,$h3,$h4 * * * $INSTALL_ROOT/edg/etc/cron/edg-fetch-crl-cron >>
    if [ ! -d $INSTALL_ROOT/edg/var/log ]; then
mkdir -p $INSTALL_ROOT/edg/var/log
    fi
    echo "Now updating the CRLs - this may take a few minutes..."
    $INSTALL_ROOT/edg/etc/cron/edg-fetch-crl-cron >> $INSTALL_ROOT/edg/var/log/edg-fetch-crl-cron.log 2>&1
```

```
fi

return 0
}
```

21.6. CONFIG_RFIO

```
config_rfio() {

INSTALL_ROOT=${INSTALL_ROOT:-/opt}

# This function turns rfio on where necessary and
# just as important, turns it off where it isn't necessary

if ( echo "${NODE_TYPE_LIST}" | grep -q SE_classic ); then

    if [ "x`grep rfio /etc/services | grep tcp`" = "x"  ]; then
echo "rfio     5001/tcp" >> /etc/services
    fi

    if [ "x`grep rfio /etc/services | grep udp`" = "x"  ]; then
echo "rfio     5001/udp" >> /etc/services
    fi

    /sbin/service rfiod restart

elif ( echo "${NODE_TYPE_LIST}" | grep -q SE_dpm ); then

    return 0

elif (  rpm -qa | grep -q CASTOR-client ); then

    /sbin/service rfiod stop
    /sbin/chkconfig --level 2345 rfiod off

fi

return 0

}
```

21.7. CONFIG_GLOBUS

```
config_globus(){
# $Id: config_globus,v 1.34 2006/01/06 13:45:51 maart Exp $


requires CE_HOST PX_HOST RB_HOST SITE_NAME

GLOBUS_MDS=no
```

```
GLOBUS_GRIDFTP=no
GLOBUS_GATEKEEPER=no

if ( echo "${NODE_TYPE_LIST}" | grep '\<'CE > /dev/null ); then
    GLOBUS_MDS=yes
    GLOBUS_GRIDFTP=yes
    GLOBUS_GATEKEEPER=yes
fi
if ( echo "${NODE_TYPE_LIST}" | grep VOBOX > /dev/null ); then
    GLOBUS_MDS=yes
    if ! ( echo "${NODE_TYPE_LIST}" | grep '\<'RB > /dev/null ); then
GLOBUS_GRIDFTP=yes
    fi
fi
if ( echo "${NODE_TYPE_LIST}" | grep '\<'SE > /dev/null ); then
    GLOBUS_MDS=yes
    GLOBUS_GRIDFTP=yes
fi
# DPM has its own ftp server
if ( echo "${NODE_TYPE_LIST}" | grep SE_dpm > /dev/null ); then
    GLOBUS_GRIDFTP=no
fi
if ( echo "${NODE_TYPE_LIST}" | grep '\<'PX > /dev/null ); then
    GLOBUS_MDS=yes
fi
if ( echo "${NODE_TYPE_LIST}" | grep '\<'RB > /dev/null ); then
    GLOBUS_MDS=yes
fi
if ( echo "${NODE_TYPE_LIST}" | grep '\<'LFC > /dev/null ); then
    GLOBUS_MDS=yes
fi
if ( echo "${NODE_TYPE_LIST}" | grep SE_dpm > /dev/null ); then
    X509_DPM1="x509_user_cert=/home/edginfo/.globus/usercert.pem"
    X509_DPM2="x509_user_key=/home/edginfo/.globus/userkey.pem"
else
    X509_DPM1=""
    X509_DPM2=""
fi
if [ "$GRIDICE_SERVER_HOST" = "`hostname -f`" ]; then
    GLOBUS_MDS=yes
fi

INSTALL_ROOT=${INSTALL_ROOT:-/opt}

cat <<EOF > /etc/globus.conf
########################################################################
#
# Globus configuraton.
#
########################################################################
[common]
GLOBUS_LOCATION=${INSTALL_ROOT}/globus
globus_flavor_name=gcc32dbg
x509_user_cert=/etc/grid-security/hostcert.pem
```

```
x509_user_key=/etc/grid-security/hostkey.pem
gridmap=/etc/grid-security/grid-mapfile
gridmapdir=/etc/grid-security/gridmapdir/

EOF

if [ "$GLOBUS_MDS" = "yes" ]; then
cat <<EOF >> /etc/globus.conf

[mds]
globus_flavor_name=gcc32dbgpthr
user=edginfo
$X509_DPM1
$X509_DPM2

[mds/gris/provider/edg]

EOF

cat <<EOF >> /etc/globus.conf
[mds/gris/registration/site]
regname=$SITE_NAME
reghn=$CE_HOST

EOF
else
echo "[mds]"  >> /etc/globus.conf

fi

if [ "$GLOBUS_GRIDFTP" = "yes" ]; then

    cat <<EOF >> /etc/globus.conf
[gridftp]
log=/var/log/globus-gridftp.log
EOF

    cat <<EOF > /etc/logrotate.d/gridftp
/var/log/globus-gridftp.log /var/log/gridftp-lcas_lcmaps.log {
missingok
daily
compress
rotate 31
create 0644 root root
sharedscripts
}
EOF

else
    echo "[gridftp]"  >> /etc/globus.conf
fi


if [ "$GLOBUS_GATEKEEPER" = "yes" ]; then
```

```
if [ "x`grep globus-gatekeeper /etc/services`" = "x" ]; then
    echo "globus-gatekeeper   2119/tcp" >> /etc/services
fi

cat <<EOF > /etc/logrotate.d/globus-gatekeeper
/var/log/globus-gatekeeper.log {
nocompress
copy
rotate 1
prerotate
killall -s USR1 -e /opt/edg/sbin/edg-gatekeeper
endscript
postrotate
find /var/log/globus-gatekeeper.log.20???????????.*[0-9] -mtime +7 -exec gzip {} \;
endscript
}
EOF

cat <<EOF >> /etc/globus.conf
[gatekeeper]

default_jobmanager=fork
job_manager_path=\$GLOBUS_LOCATION/libexec
globus_gatekeeper=${INSTALL_ROOT}/edg/sbin/edg-gatekeeper
extra_options=\"-lcas_db_file lcas.db -lcas_etc_dir ${INSTALL_ROOT}/edg/etc/lcas/ -lcasmod_dir \
${INSTALL_ROOT}/edg/lib/lcas/ -lcmaps_db_file lcmaps.db -lcmaps_etc_dir ${INSTALL_ROOT}/edg/etc/lcmaps -lcmapsmod_d
logfile=/var/log/globus-gatekeeper.log
jobmanagers="fork ${JOB_MANAGER}"

[gatekeeper/fork]
type=fork
job_manager=globus-job-manager

[gatekeeper/${JOB_MANAGER}]
type=${JOB_MANAGER}

EOF
else
cat <<EOF >> /etc/globus.conf
[gatekeeper]
default_jobmanager=fork
job_manager_path=${GLOBUS_LOCATION}/libexec

jobmanagers="fork "

[gatekeeper/fork]
type=fork
job_manager=globus-job-manager
EOF
fi

$INSTALL_ROOT/globus/sbin/globus-initialization.sh 2>> $YAIM_LOG
```

```
if [ "$GLOBUS_MDS" = "yes" ]; then
    /sbin/chkconfig globus-mds on
    /sbin/service globus-mds stop
    /sbin/service globus-mds start
fi
if [ "$GLOBUS_GATEKEEPER" = "yes" ]; then
    /sbin/chkconfig globus-gatekeeper on
    /sbin/service globus-gatekeeper stop
    /sbin/service globus-gatekeeper start
fi
if [ "$GLOBUS_GRIDFTP" = "yes" ]; then
    /sbin/chkconfig globus-gridftp on
    /sbin/service globus-gridftp stop
    /sbin/service globus-gridftp start
    /sbin/chkconfig lcg-mon-gridftp on
    /etc/rc.d/init.d/lcg-mon-gridftp restart
fi

return 0
}
```

21.8. CONFIG_LCGENV

```
config_lcgenv() {

INSTALL_ROOT=${INSTALL_ROOT:-/opt}

if !( echo "${NODE_TYPE_LIST}" | grep TAR > /dev/null ); then
LCG_ENV_LOC=/etc/profile.d
else
LCG_ENV_LOC=${INSTALL_ROOT}/etc/env.d
fi

requires BDII_HOST SITE_NAME CE_HOST

if ( ! echo "${NODE_TYPE_LIST}" | grep -q UI ); then
    requires VOS VO__SW_DIR SE_LIST
fi

default_se="${SE_LIST%% *}"

if [ "$default_se" ]; then
    for VO in `echo $VOS | tr '[:lower:]' '[:upper:]'`; do
if [ "x`eval echo '$'VO_${VO}_DEFAULT_SE`" = "x" ]; then
    eval VO_${VO}_DEFAULT_SE=$default_se
fi
    done
fi

########## sh ##########
cat << EOF > ${LCG_ENV_LOC}/lcgenv.sh
#!/bin/sh
```

```
if test "x\${LCG_ENV_SET+x}" = x; then
export LCG_GFAL_INFOSYS=$BDII_HOST:2170
EOF

if [ "$PX_HOST" ]; then
echo "export MYPROXY_SERVER=$PX_HOST" >> ${LCG_ENV_LOC}/lcgenv.sh
fi

if ( echo "${NODE_TYPE_LIST}" | egrep -q 'WN|VOBOX' ); then
    if [ "$SITE_NAME" ]; then
echo "export SITE_NAME=$SITE_NAME" >> ${LCG_ENV_LOC}/lcgenv.sh
    fi

    if [ "$CE_HOST" ]; then
echo "export SITE_GIIS_URL=$CE_HOST" >> ${LCG_ENV_LOC}/lcgenv.sh
    fi
fi

if [ -d ${INSTALL_ROOT}/d-cache/srm/bin ]; then
    echo export PATH="\${PATH}:${INSTALL_ROOT}/d-cache/srm/bin:${INSTALL_ROOT}/d-cache/dcap/bin" >> ${LCG_ENV_LOC}/
fi

if [ -d ${INSTALL_ROOT}/d-cache/dcap/lib ]; then
    echo export LD_LIBRARY_PATH=\${LD_LIBRARY_PATH}:${INSTALL_ROOT}/d-cache/dcap/lib >> ${LCG_ENV_LOC}/lcgenv.sh
fi

if [ -d ${INSTALL_ROOT}/d-cache/srm ]; then
    echo export SRM_PATH=${INSTALL_ROOT}/d-cache/srm >> ${LCG_ENV_LOC}/lcgenv.sh
fi

if [ "$EDG_WL_SCRATCH" ]; then
    echo "export EDG_WL_SCRATCH=$EDG_WL_SCRATCH"  >> ${LCG_ENV_LOC}/lcgenv.sh
fi

for VO in `echo $VOS | tr '[:lower:]' '[:upper:]'`; do

    default_se=`eval echo '$'VO_${VO}_DEFAULT_SE`
    if [ "$default_se" ]; then
echo "export VO_${VO}_DEFAULT_SE=$default_se" >> ${LCG_ENV_LOC}/lcgenv.sh
    fi

    if ( ! echo "${NODE_TYPE_LIST}" | grep -q UI ); then
sw_dir=`eval echo '$'VO_${VO}_SW_DIR`
if [ "$sw_dir" ]; then
    echo "export VO_${VO}_SW_DIR=$sw_dir" >> ${LCG_ENV_LOC}/lcgenv.sh
fi
    fi

done

if [ "$VOBOX_HOST" ]; then
   requires GSSKLOG
   if  [ "${GSSKLOG}x" == "yesx" ]; then
      requires GSSKLOG_SERVER
```

```
      echo "export GSSKLOG_SERVER=$GSSKLOG_SERVER" >> ${LCG_ENV_LOC}/lcgenv.sh
   fi
fi

if [ "${DPM_HOST}" ]; then
    echo "export DPNS_HOST=${DPM_HOST}" >> ${LCG_ENV_LOC}/lcgenv.sh
    echo "export DPM_HOST=${DPM_HOST}" >> ${LCG_ENV_LOC}/lcgenv.sh
fi

if [ "$GLOBUS_TCP_PORT_RANGE" ]; then
    echo "export MYPROXY_TCP_PORT_RANGE=\"${GLOBUS_TCP_PORT_RANGE/ /,}\"" >> ${LCG_ENV_LOC}/lcgenv.sh
fi

if ( echo $NODE_TYPE_LIST | egrep -q UI ); then
    cat << EOF >> ${LCG_ENV_LOC}/lcgenv.sh
if [ "x\$X509_USER_PROXY" = "x" ]; then
    export X509_USER_PROXY=/tmp/x509up_u\$(id -u)
fi
EOF
fi

echo fi >> ${LCG_ENV_LOC}/lcgenv.sh
########## sh ##########


########## csh ##########
cat << EOF > ${LCG_ENV_LOC}/lcgenv.csh
#!/bin/csh
if ( ! \$?LCG_ENV_SET ) then
setenv LCG_GFAL_INFOSYS $BDII_HOST:2170
EOF

if [ "$PX_HOST" ]; then
echo "setenv MYPROXY_SERVER $PX_HOST" >>  ${LCG_ENV_LOC}/lcgenv.csh
fi

if ( echo "${NODE_TYPE_LIST}" | egrep -q 'WN|VOBOX' ); then
    if [ "$SITE_NAME" ]; then
echo "setenv SITE_NAME $SITE_NAME" >>  ${LCG_ENV_LOC}/lcgenv.csh
    fi

    if [ "$CE_HOST" ]; then
echo "setenv SITE_GIIS_URL $CE_HOST" >>  ${LCG_ENV_LOC}/lcgenv.csh
    fi
fi

if [ -d ${INSTALL_ROOT}/d-cache/srm/bin ]; then
    echo setenv PATH "\${PATH}:${INSTALL_ROOT}/d-cache/srm/bin:${INSTALL_ROOT}/d-cache/dcap/bin" >> ${LCG_ENV_LOC}/
fi

if [ -d ${INSTALL_ROOT}/d-cache/dcap/lib ]; then
    echo setenv LD_LIBRARY_PATH \${LD_LIBRARY_PATH}:${INSTALL_ROOT}/d-cache/dcap/lib >> ${LCG_ENV_LOC}/lcgenv.csh
fi
```

```
if [ -d ${INSTALL_ROOT}/d-cache/srm ]; then
    echo setenv SRM_PATH ${INSTALL_ROOT}/d-cache/srm >> ${LCG_ENV_LOC}/lcgenv.csh
fi

if [ "$EDG_WL_SCRATCH" ]; then
    echo "setenv EDG_WL_SCRATCH $EDG_WL_SCRATCH"  >> ${LCG_ENV_LOC}/lcgenv.csh
fi

for VO in `echo $VOS | tr '[:lower:]' '[:upper:]'`; do

    default_se=`eval echo '$'VO_${VO}_DEFAULT_SE`
    if [ "$default_se" ]; then
echo "setenv VO_${VO}_DEFAULT_SE $default_se" >> ${LCG_ENV_LOC}/lcgenv.csh
    fi

    if ( ! echo "${NODE_TYPE_LIST}" | grep -q UI ); then
sw_dir=`eval echo '$'VO_${VO}_SW_DIR`
if [ "$sw_dir" ]; then
    echo "setenv VO_${VO}_SW_DIR $sw_dir" >> ${LCG_ENV_LOC}/lcgenv.csh
fi
    fi

done

if [ "$VOBOX_HOST" ]; then
   requires GSSKLOG
   if [ "${GSSKLOG}x" == "yesx" ]; then
   requires GSSKLOG_SERVER
      echo "setenv GSSKLOG_SERVER $GSSKLOG_SERVER" >> ${LCG_ENV_LOC}/lcgenv.csh
   fi
fi


if [ "${DPM_HOST}" ]; then
    echo "setenv DPNS_HOST ${DPM_HOST}" >> ${LCG_ENV_LOC}/lcgenv.csh
    echo "setenv DPM_HOST ${DPM_HOST}" >> ${LCG_ENV_LOC}/lcgenv.csh
fi

if [ "$GLOBUS_TCP_PORT_RANGE" ]; then
    echo "setenv MYPROXY_TCP_PORT_RANGE \"${GLOBUS_TCP_PORT_RANGE/ /,}\"" >> ${LCG_ENV_LOC}/lcgenv.csh
fi

if ( echo $NODE_TYPE_LIST | egrep -q UI ); then
    cat << EOF >> ${LCG_ENV_LOC}/lcgenv.csh
if ( ! \$?X509_USER_PROXY ) then
   setenv X509_USER_PROXY /tmp/x509up_u\`id -u\`
endif
EOF
fi

echo endif >> ${LCG_ENV_LOC}/lcgenv.csh
########## csh ##########

chmod a+xr ${LCG_ENV_LOC}/lcgenv.csh
```

```
chmod a+xr ${LCG_ENV_LOC}/lcgenv.sh

return 0
}
```

21.9. CONFIG_REPLICA_MANAGER

```
config_replica_manager(){

# SE_HOST and CE_HOST are not strictly required
requires BDII_HOST

se_host="${SE_LIST%% *}"

INSTALL_ROOT=${INSTALL_ROOT:-/opt}

if [ -f ${INSTALL_ROOT}/edg/etc/edg-replica-manager/edg-replica-manager.conf.values_local ]; then
        mv -f ${INSTALL_ROOT}/edg/etc/edg-replica-manager/edg-replica-manager.conf.values_local /tmp/edg-replica-ma
fi

cat <<EOF > ${INSTALL_ROOT}/edg/etc/edg-replica-manager/edg-replica-manager.conf.values_local
@EDG.LOCATION@|$INSTALL_ROOT/edg|location of edg middleware
@LOCALDOMAIN@|`hostname -d`|the local domain
@DEFAULT.SE@|$se_host|the host of the close SE
@DEFAULT.CE@|$CE_HOST|the host of the close CE
@INFOSERVICE@|MDS|The info provider to use. It can be Stub, MDS or RGMA
@RLS.MODE@|LrcOnly|The mode the RLS should be run in. LrcOnly or WithRli
@STUBFILE@||The properties file for the static file - only needed in Stub mode
@MDS.HOST@|$BDII_HOST|The host of the MDS info provider
@MDS.PORT@|2170|The port of the MDS info provider
@ROS.FAILURE@|false|Fail if no ROS is available
@CONF.GCC@|_gcc3_2_2|The gcc suffix as used on the build box (empty for 2.95, _gcc3_2_2 for 3.2.)
@IGNORE.PREFIX@|true|Whether the RM will ignore the lfn and guid prefix.
@GRIDFTP.DCAU@|false|Does GridFTP use Data Channel Authentication (DCAU)
@GRIDFTP.STREAMS.SMALL@|1|The default number of stream to use for a small file
@GRIDFTP.STREAMS.BIG@|3|The default number of stream to use for a big file
@GRIDFTP.FILESIZE.THRESHOLD@|100|The Threshold (in MB) above which a file to transfer is considered "big"
EOF

oldEDG_LOCATION=$EDG_LOCATION
oldEDG_LOCATION_VAR=$EDG_LOCATION_VAR
export EDG_LOCATION=${INSTALL_ROOT}/edg
export EDG_LOCATION_VAR=${INSTALL_ROOT}/edg/var

${INSTALL_ROOT}/edg/sbin/edg-replica-manager-configure \
${INSTALL_ROOT}/edg/etc/edg-replica-manager/edg-replica-manager.conf.values_local >> $YAIM_LOG

export EDG_LOCATION=$oldEDG_LOCATION
export EDG_LOCATION_VAR=$oldEDG_LOCATION_VAR

return 0
}
```

21.10. CONFIG_USERS

```
config_users(){
#
# Creates the Pool Users.
#
# Takes the users, groups and ids from a configuration file (USERS_CONF).
# File format:
#
# UserId:User:GroupId:Group
#

INSTALL_ROOT=${INSTALL_ROOT:-/opt}

requires USERS_CONF VOS

if [ ! -e $USERS_CONF ]; then
    echo "$USERS_CONF not found."
    return 1
fi

check_users_conf_format

# Add each group required by $VOS
awk -F: '{print $3, $4, $5}' ${USERS_CONF} | sort -u | while read gid groupname virtorg; do
    if ( [ "$virtorg" ] && echo $VOS | grep -w "$virtorg" > /dev/null ); then
groupadd -g $gid $groupname 2> /dev/null
    fi
done

grid_accounts=
newline='
'

# Add all the users for each VO in ${VOS}
for x in `cat $USERS_CONF`; do
    # ensure that this VO is in the $VOS list
    virtorg=`echo $x | cut -d":" -f5`
    if ( [ "$virtorg" ] && echo $VOS | grep -w "$virtorg" > /dev/null ); then
user=`echo $x | cut -d":" -f2`
id=`echo $x | cut -d":" -f1`
group=`echo $x | cut -d":" -f3`
if ( ! id $user > /dev/null 2>&1 ); then
    useradd -c "mapped user for group ID $group" -u $id -g $group $user
fi

# grid users shall not be able to submit at or cron jobs
for deny in /etc/at.deny /etc/cron.deny; do
    tmp=$deny.$$
    touch $deny
    (grep -v "^$user\$" $deny; echo "$user") > $tmp && mv $tmp $deny
done

grid_accounts="$grid_accounts$newline$user"
```

```
    fi
done

(
    cga=$INSTALL_ROOT/lcg/etc/cleanup-grid-accounts.conf
    cga_tmp=$cga.$$

    [ -r $cga ] || exit

    (
sed '/YAIM/,$d' $cga
echo "# next lines added by YAIM on `date`"
echo "ACCOUNTS='$grid_accounts$newline'"
    ) > $cga_tmp

    mv $cga_tmp $cga
)

let minute="$RANDOM%60"
let h="$RANDOM%6"
f=/var/log/cleanup-grid-accounts.log

if ( echo "${NODE_TYPE_LIST}" | grep '\<CE' > /dev/null ); then

    cron_job cleanup-grid-accounts root "$minute $h * * * \
$INSTALL_ROOT/lcg/sbin/cleanup-grid-accounts.sh -v -F >> $f 2>&1"

    cat <<EOF > /etc/logrotate.d/cleanup-grid-accounts
$f {
    compress
    daily
    rotate 30
    missingok
}
EOF

elif ( echo "${NODE_TYPE_LIST}" | grep '\<WN' > /dev/null ); then

    cron_job cleanup-grid-accounts root "$minute $h * * * \
$INSTALL_ROOT/lcg/sbin/cleanup-grid-accounts.sh -v >> $f 2>&1"

    cat <<EOF > /etc/logrotate.d/cleanup-grid-accounts
$f {
    compress
    daily
    rotate 30
    missingok
}
EOF

fi

return 0
}
```

21.11. CONFIG_SW_DIR

```
function config_sw_dir () {

requires VO__SW_DIR

dir=""
for VO in `echo $VOS | tr '[:lower:]' '[:upper:]'`; do
dir=`eval echo '$'VO_${VO}_SW_DIR`
if [ "$dir" = "." ]; then
    return 0
fi
if [ ! -d "$dir" ]; then
    mkdir -p $dir
fi
sgmuser=`users_getsgmuser $VO`
if [ "$sgmuser" ]; then
    sgmgroup=`id -g $sgmuser`
    chown ${sgmuser}:${sgmgroup} $dir
fi
done


return 0
}
```

21.12. CONFIG_JAVA

```
function config_java () {

INSTALL_ROOT=${INSTALL_ROOT:-/opt}

# If JAVA_LOCATION is not set by the admin, take a guess
if [ -z "$JAVA_LOCATION" ]; then
    java=`rpm -qa | grep j2sdk-` || java=`rpm -qa | grep j2re`
    if [ "$java" ]; then
JAVA_LOCATION=`rpm -ql $java | egrep '/bin/java$' | sort | head -1 | sed 's#/bin/java##'`
    fi
fi

if [ ! "$JAVA_LOCATION" -o ! -d "$JAVA_LOCATION" ]; then
   echo "Please check your value for JAVA_LOCATION"
   return 1
fi

if ( echo "${NODE_TYPE_LIST}" | grep TAR > /dev/null ); then

# We're configuring a relocatable distro

    if [ ! -d ${INSTALL_ROOT}/edg/etc/profile.d ]; then
mkdir -p ${INSTALL_ROOT}/edg/etc/profile.d/
    fi
```

```
    cat  > $INSTALL_ROOT/edg/etc/profile.d/j2.sh <<EOF

JAVA_HOME=$JAVA_LOCATION
export JAVA_HOME
EOF

    cat  > $INSTALL_ROOT/edg/etc/profile.d/j2.csh <<EOF

setenv JAVA_HOME $JAVA_LOCATION
EOF

    chmod a+rx $INSTALL_ROOT/edg/etc/profile.d/j2.sh
    chmod a+rx $INSTALL_ROOT/edg/etc/profile.d/j2.csh

    return 0

fi # end of relocatable stuff

# We're root and it's not a relocatable

if [ ! -d /etc/java ]; then
    mkdir /etc/java
fi

echo "export JAVA_HOME=$JAVA_LOCATION" > /etc/java/java.conf
echo "export JAVA_HOME=$JAVA_LOCATION" > /etc/java.conf
chmod +x /etc/java/java.conf

#This hack is here due to SL and the java profile rpms, Laurence Field

if [ ! -d ${INSTALL_ROOT}/edg/etc/profile.d ]; then
    mkdir -p ${INSTALL_ROOT}/edg/etc/profile.d/
fi

cat << EOF > $INSTALL_ROOT/edg/etc/profile.d/j2.sh
if [ -z "\$PATH" ]; then
   export PATH=${JAVA_LOCATION}/bin
else
   export PATH=${JAVA_LOCATION}/bin:\${PATH}
fi
EOF

chmod a+rx $INSTALL_ROOT/edg/etc/profile.d/j2.sh

cat << EOF > $INSTALL_ROOT/edg/etc/profile.d/j2.csh
if ( \$?PATH ) then
    setenv PATH ${JAVA_LOCATION}/bin:\${PATH}
else
    setenv PATH ${JAVA_LOCATION}/bin
endif
EOF

chmod a+rx $INSTALL_ROOT/edg/etc/profile.d/j2.csh
```

```
return 0

}
```

21.13. CONFIG_RGMA_CLIENT

```
config_rgma_client(){

requires MON_HOST REG_HOST

INSTALL_ROOT=${INSTALL_ROOT:-/opt}

# NB java stuff now in config_java, which must be run before

export RGMA_HOME=${INSTALL_ROOT}/glite

# in order to use python from userdeps.tgz we need to source the env
if ( echo "${NODE_TYPE_LIST}" | grep TAR > /dev/null ); then
    . $INSTALL_ROOT/etc/profile.d/grid_env.sh
fi

${RGMA_HOME}/share/rgma/scripts/rgma-setup.py --secure=yes --server=${MON_HOST} --registry=${REG_HOST} --schema=${R

cat << EOF > ${INSTALL_ROOT}/edg/etc/profile.d/edg-rgma-env.sh
export RGMA_HOME=${INSTALL_ROOT}/glite
export APEL_HOME=${INSTALL_ROOT}/glite

echo \$PYTHONPATH | grep -q ${INSTALL_ROOT}/glite/lib/python && isthere=1 || isthere=0
if [ \$isthere = 0 ]; then
    if [ -z \$PYTHONPATH ]; then
        export PYTHONPATH=${INSTALL_ROOT}/glite/lib/python
    else
        export PYTHONPATH=\$PYTHONPATH:${INSTALL_ROOT}/glite/lib/python
    fi
fi

echo \$LD_LIBRARY_PATH | grep -q ${INSTALL_ROOT}/glite/lib && isthere=1 || isthere=0
if [ \$isthere = 0 ]; then
    if [ -z \$LD_LIBRARY_PATH ]; then
        export LD_LIBRARY_PATH=${INSTALL_ROOT}/glite/lib
    else
        export LD_LIBRARY_PATH=\$LD_LIBRARY_PATH:${INSTALL_ROOT}/glite/lib
    fi
fi
EOF

chmod a+rx ${INSTALL_ROOT}/edg/etc/profile.d/edg-rgma-env.sh

cat << EOF > ${INSTALL_ROOT}/edg/etc/profile.d/edg-rgma-env.csh
setenv RGMA_HOME ${INSTALL_ROOT}/glite
setenv APEL_HOME ${INSTALL_ROOT}/glite
```

```
echo \$PYTHONPATH | grep -q ${INSTALL_ROOT}/glite/lib/python && set isthere=1 || set isthere=0
if ( \$isthere == 0 ) then
    if ( -z \$PYTHONPATH ) then
        setenv PYTHONPATH ${INSTALL_ROOT}/glite/lib/python
    else
        setenv PYTHONPATH \$PYTHONPATH\:${INSTALL_ROOT}/glite/lib/python
    endif
endif

echo \$LD_LIBRARY_PATH | grep -q ${INSTALL_ROOT}/glite/lib && set isthere=1 || set isthere=0
if ( \$isthere == 0 ) then
    if ( -z \$LD_LIBRARY_PATH ) then
        setenv LD_LIBRARY_PATH ${INSTALL_ROOT}/glite/lib
    else
        setenv LD_LIBRARY_PATH \$LD_LIBRARY_PATH\:${INSTALL_ROOT}/glite/lib
    endif
endif
EOF

chmod a+rx  ${INSTALL_ROOT}/edg/etc/profile.d/edg-rgma-env.csh

return 0
}
```

21.14. CONFIG_WORKLOAD_MANAGER_ENV

```
config_workload_manager_env(){

INSTALL_ROOT=${INSTALL_ROOT:-/opt}

mkdir -p $INSTALL_ROOT/edg/var/etc/profile.d

cp $INSTALL_ROOT/edg/etc/profile.d/edg-wl.csh $INSTALL_ROOT/edg/etc/profile.d/edg-wl.sh $INSTALL_ROOT/edg/var/etc/p

return 0
}
```

21.15. CONFIG_FTS_CLIENT

```
function config_fts_client () {

INSTALL_ROOT=${INSTALL_ROOT:-/opt}

if [ -z "$FTS_SERVER_URL" ]; then
    return 0
fi

cat > ${INSTALL_ROOT}/glite/etc/services.xml <<EOF
<?xml version="1.0" encoding="UTF-8"?>
```

```
<services>

  <service name="EGEEfts">
    <parameters>
      <endpoint>${FTS_SERVER_URL}/services/FileTransfer</endpoint>
      <type>org.glite.FileTransfer</type>
      <version>3.0.0</version>
    </parameters>
  </service>

  <service name="EGEEchannel">
    <parameters>
      <endpoint>${FTS_SERVER_URL}/services/ChannelManagement</endpoint>
      <type>org.glite.ChannelManagement</type>
      <version>3.0.0</version>
    </parameters>
  </service>

</services>
EOF

return 0

}
```

21.16. CONFIG_GLITE_ENV

```
function config_glite_env () {

INSTALL_ROOT=${INSTALL_ROOT:-/opt}

cat > /etc/profile.d/gliteenv.sh <<'EOF'

if test "x${LCG_ENV_SET+x}" = x; then

    GLITE_LOCATION=${GLITE_LOCATION:-/opt/glite}
    GLITE_LOCATION_VAR=${GLITE_LOCATION_VAR:-$GLITE_LOCATION/var}
    GLITE_LOCATION_LOG=${GLITE_LOCATION_LOG:-$GLITE_LOCATION/log}
    GLITE_LOCATION_TMP=${GLITE_LOCATION_TMP:-$GLITE_LOCATION/tmp}

    if [ -z "$PATH" ]; then
PATH="${GLITE_LOCATION}/bin:${GLITE_LOCATION}/externals/bin"
    else
PATH="${PATH}:${GLITE_LOCATION}/bin:${GLITE_LOCATION}/externals/bin"
    fi

    if [ -z "$LD_LIBRARY_PATH" ]; then
LD_LIBRARY_PATH="${GLITE_LOCATION}/lib:${GLITE_LOCATION}/externals/lib"
    else
LD_LIBRARY_PATH="${LD_LIBRARY_PATH}:${GLITE_LOCATION}/lib:${GLITE_LOCATION}/externals/lib"
    fi
```

```
    if [ -z "$PERLLIB" ]; then
PERLLIB="${GLITE_LOCATION}/lib/perl5"
    else
PERLLIB="${PERLLIB}:${GLITE_LOCATION}/lib/perl5"
    fi

    if [ -z "$MANPATH" ]; then
MANPATH="${GLITE_LOCATION}/share/man"
    else
MANPATH="${MANPATH}:${GLITE_LOCATION}/share/man"
    fi

    export GLITE_LOCATION GLITE_LOCATION_VAR GLITE_LOCATION_LOG GLITE_LOCATION_TMP PATH LD_LIBRARY_PATH PERLLIB MAN

fi

EOF

cat  > /etc/profile.d/gliteenv.csh <<'EOF'

if ( ! $?LCG_ENV_SET ) then

    if ( ! $?GLITE_LOCATION ) then
setenv GLITE_LOCATION "/opt/glite"
    endif

    if ( ! $?GLITE_LOCATION_VAR ) then
setenv GLITE_LOCATION_VAR "${GLITE_LOCATION}/var"
    endif

    if ( ! $?GLITE_LOCATION_LOG ) then
setenv GLITE_LOCATION_LOG "${GLITE_LOCATION}/log"
    endif

    if ( ! $?GLITE_LOCATION_TMP ) then
setenv GLITE_LOCATION_TMP "${GLITE_LOCATION}/tmp"
    endif

    if ( ! $?PATH ) then
setenv PATH "${GLITE_LOCATION}/bin:${GLITE_LOCATION}/externals/bin"
    else
setenv PATH "${PATH}:${GLITE_LOCATION}/bin:${GLITE_LOCATION}/externals/bin"
    endif

    if ( ! $?LD_LIBRARY_PATH ) then
setenv LD_LIBRARY_PATH "${GLITE_LOCATION}/lib:${GLITE_LOCATION}/externals/lib"
    else
setenv LD_LIBRARY_PATH "${LD_LIBRARY_PATH}:${GLITE_LOCATION}/lib:${GLITE_LOCATION}/externals/lib"
    endif

    if ( ! $?PERLLIB ) then
setenv PERLLIB "${GLITE_LOCATION}/lib/perl5"
    else
setenv PERLLIB "${PERLLIB}:${GLITE_LOCATION}/lib/perl5"
```

```
    endif

    if ( ! $?MANPATH ) then
setenv MANPATH "${GLITE_LOCATION}/share/man"
    else
setenv MANPATH "${MANPATH}:${GLITE_LOCATION}/share/man"
    endif

endif

EOF

return 0

}
```

21.17. CONFIG_GSISSH

```
config_gsissh(){

INSTALL_ROOT=${INSTALL_ROOT:-/opt}
export GLOBUS_LOCATION=${INSTALL_ROOT}/globus
export PERLLIB=${PERLLIB}:${INSTALL_ROOT}/gpt/lib/perl


#configuring gsissh server

if [ -f ${INSTALL_ROOT}/globus/setup/gsi_openssh_setup/setup-openssh ]; then
        ${INSTALL_ROOT}/globus/setup/gsi_openssh_setup/setup-openssh
else
        echo "Impossible to configure gsissh: setup-openssh configuration script not found"
        exit 1
fi

if ( echo "${NODE_TYPE_LIST}" | grep VOBOX > /dev/null ); then

requires VOBOX_PORT

cat ${INSTALL_ROOT}/globus/sbin/SXXsshd | sed "s/SSHD_ARGS=\"\"/SSHD_ARGS=\"-p ${VOBOX_PORT}\"/" > ${INSTALL_ROOT}/

if [ $? == 0 ]; then
   mv ${INSTALL_ROOT}/globus/sbin/SXXsshd.TMP ${INSTALL_ROOT}/globus/sbin/SXXsshd
   chmod +x ${INSTALL_ROOT}/globus/sbin/SXXsshd
else
   echo "Unable to modify GSISSH startup script."
   exit 1
fi


if [ "x`grep LCG ${INSTALL_ROOT}/globus/etc/ssh/sshd_config`" = "x"  ]; then
cat <<EOF >>${INSTALL_ROOT}/globus/etc/ssh/sshd_config
####################################
```

```
#ADDs ON from LCG

PermitRootLogin no
RSAAuthentication no
PubkeyAuthentication no
PasswordAuthentication no
ChallengeResponseAuthentication no
####################################

EOF

fi


# install the service under /etc/init.d and configure it


if  [ ! -f /etc/init.d/gsisshd ]; then
ln -s ${INSTALL_ROOT}/globus/sbin/SXXsshd /etc/init.d/gsisshd
fi

/sbin/chkconfig --add gsisshd
/sbin/chkconfig gsisshd on
/sbin/service gsisshd restart

fi


#configure the client

if [ ! -d ${INSTALL_ROOT}/globus/etc/ssh ]; then
    mkdir -p ${INSTALL_ROOT}/globus/etc/ssh
fi


if [ "x`grep LCG ${INSTALL_ROOT}/globus/etc/ssh/ssh_config`" = "x"  ]; then

cat <<EOF >>${INSTALL_ROOT}/globus/etc/ssh/ssh_config
####################################
#ADDs ON from LCG

GssapiAuthentication yes
GssapiKeyExchange yes
GssapiDelegateCredentials yes

####################################
EOF

fi

}
```

21.18. CONFIG_TORQUE_CLIENT

```
config_torque_client(){

TORQUE_SERVER=${TORQUE_SERVER:-${CE_HOST}}

requires CE_HOST TORQUE_SERVER

se_host="${SE_LIST%% *}"

# INSTALL_ROOT is not used, as torque isn't managed by the relocatable dist
# INSTALL_ROOT=${INSTALL_ROOT:-/opt}

echo "$TORQUE_SERVER" > /var/spool/pbs/server_name

if [ "x`grep pbs_mom /etc/services`" = "x"  ]; then
    echo "pbs_mom     15002/tcp" >> /etc/services
fi

if [ "x`grep pbs_resmon /etc/services | grep tcp`" = "x"  ]; then
    echo "pbs_resmon     15003/tcp" >> /etc/services
fi

if [ "x`grep pbs_resmon /etc/services | grep udp`" = "x"  ]; then
    echo "pbs_resmon     15003/udp" >> /etc/services
fi

cat <<EOF > /etc/ssh/ssh_config
Host *
Protocol 2,1
   RhostsAuthentication yes
   RhostsRSAAuthentication yes
   RSAAuthentication yes
   PasswordAuthentication yes
   EnableSSHKeysign yes
   HostbasedAuthentication yes
EOF

cat << EOF >  /opt/edg/etc/edg-pbs-knownhosts.conf
NODES     = $CE_HOST $se_host
PBSBIN    = /usr/bin
KEYTYPES   = rsa1,rsa,dsa
KNOWNHOSTS = /etc/ssh/ssh_known_hosts

EOF

# workaround for duplicate key entries (Savannah bug 5530)
for hostname in $CE_HOST $se_host; do
   if [ -f /etc/ssh/ssh_known_hosts ];then
      grep -v $hostname /etc/ssh/ssh_known_hosts  > /etc/ssh/ssh_known_hosts.tmp
      /usr/bin/ssh-keyscan -t rsa $hostname >> /etc/ssh/ssh_known_hosts.tmp 2>/dev/null

      if [ $? = 0 ]; then
         mv /etc/ssh/ssh_known_hosts.tmp  /etc/ssh/ssh_known_hosts
```

```
      fi
   fi
done

/opt/edg/sbin/edg-pbs-knownhosts

cat << EOF > /var/spool/pbs/mom_priv/config
\$clienthost $TORQUE_SERVER
\$clienthost localhost
\$restricted $TORQUE_SERVER
\$logevent 255
\$ideal_load 1.6
\$max_load 2.1
EOF

/sbin/chkconfig pbs_mom on
/etc/rc.d/init.d/pbs_mom stop
sleep 1
/etc/rc.d/init.d/pbs_mom start

cron_job edg-pbs-knownhosts root "03 1,7,13,19 * * * /opt/edg/sbin/edg-pbs-knownhosts"

cron_job mom_logs root "33 3 * * * find /var/spool/pbs/mom_logs -mtime +7 -exec gzip -9 {} \; 2> /dev/null"

return 0
}
```