



LHC COMPUTING GRID

LCG - PX - GENERIC CONFIGURATION REFERENCE

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

Abstract: Configuration steps done by the YAIM script 'configure_PX'



CONTENTS

1. INTRODUCTION	4
2. VARIABLES	5
3. CONFIGURE LIBRARY PATHS	8
3.1. SPECIFICATION OF FUNCTION: CONFIG_LDCONF	8
4. SET-UP EDG CONFIGURATION VARIABLES	9
4.1. SPECIFICATION OF FUNCTION: CONFIG_SYSCONFIG_EDG	9
5. SET-UP GLOBUS CONFIGURATION VARIABLES	10
5.1. SPECIFICATION OF FUNCTION: CONFIG_SYSCONFIG_GLOBUS	10
6. SET-UP LCG CONFIGURATION VARIABLES	11
6.1. SPECIFICATION OF FUNCTION: CONFIG_SYSCONFIG_LCG	11
7. SET-UP UPDATING OF CRLS	12
7.1. SPECIFICATION OF FUNCTION: CONFIG_CRL	12
8. SET-UP RFIO	13
8.1. SPECIFICATION OF FUNCTION: CONFIG_RFIO.....	13
9. SET-UP HOST CERTIFICATES	14
9.1. SPECIFICATION OF FUNCTION: CONFIG_HOST_CERTS	14
10. CREATE EDG USERS	16
10.1. SPECIFICATION OF FUNCTION: CONFIG_EDGUSERS	16
11. SET-UP JAVA LOCATION	17
11.1. SPECIFICATION OF FUNCTION: CONFIG_JAVA.....	17
12. SET-UP R-GMA CLIENT	18
12.1. SPECIFICATION OF FUNCTION: CONFIG_RGMA_CLIENT	18
13. SET-UP GENERIC INFORMATION PROVIDER	19
13.1. SPECIFICATION OF FUNCTION: CONFIG_GIP.....	26
14. SET-UP GLOBUS DAEMONS	29
14.1. SPECIFICATION OF FUNCTION: CONFIG_GLOBUS	29
15. SET-UP MYPROXY	31
15.1. SPECIFICATION OF FUNCTION: CONFIG_PROXY_SERVER	31



16. SOURCE CODE	32
16.1. CONFIG_LDCONF.....	32
16.2. CONFIG_SYSCONFIG_EDG	32
16.3. CONFIG_SYSCONFIG_GLOBUS.....	33
16.4. CONFIG_SYSCONFIG_LCG.....	33
16.5. CONFIG_CRL.....	34
16.6. CONFIG_RFIO	35
16.7. CONFIG_HOST_CERTS	35
16.8. CONFIG_EDGUSERS	36
16.9. CONFIG_JAVA	36
16.10. CONFIG_RGMA_CLIENT	38
16.11. CONFIG_GIP	39
16.12. CONFIG_GLOBUS.....	56
16.13. CONFIG_PROXY_SERVER	59



1. INTRODUCTION

This document lists the manual steps for the installation and configuration of a LCG PX 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 PX 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_BATCH_SYS : Implementation of site batch system. Available values are “torque”, “lsf”, “pbs”, “condor” etc.

CE_CPU_MODEL : Model of the CPU used by the WN (WN specification). This parameter is a string whose domain is not defined yet in the GLUE Schema. The value used for Pentium III is "PIII".

CE_CPU_SPEED : Clock frequency in Mhz (WN specification).

CE_CPU_VENDOR : Vendor of the CPU. used by the WN (WN specification). This parameter is a string whose domain is not defined yet in the GLUE Schema. The value used for Intel is “intel”.

CE_HOST : Computing Element Hostname.

CE_INBOUNDIP : TRUE if inbound connectivity is enabled at your site, FALSE otherwise (WN specification).

CE_MINPHYSMEM : RAM size in kblocks (WN specification).

CE_MINVIRTMEM : Virtual Memory size in kblocks (WN specification).

CE_OS : Operating System name (WN specification).

CE_OS_RELEASE : Operating System release (WN specification).

CE_OUTBOUNDIP : TRUE if outbound connectivity is enabled at your site, FALSE otherwise (WN specification).

CE_RUNTIMEENV : List of software tags supported by the site. The list can include VO-specific software tags. In order to assure backward compatibility it should include the entry 'LCG-2', the current middleware version and the list of previous middleware tags.

CE_SF00 : Performance index of your fabric in SpecFloat 2000 (WN specification). For some examples of Spec values see <http://www.specbench.org/osg/cpu2000/results/cint2000.html>.

CE_SI00 : Performance index of your fabric in SpecInt 2000 (WN specification). For some examples of Spec values see <http://www.specbench.org/osg/cpu2000/results/cint2000.html>.

CE_SMPSIZE : Number of cpus in an SMP box (WN specification).

CLASSIC_HOST : The name of your SE_classic host.

CLASSIC_STORAGE_DIR : The root storage directory on CLASSIC_HOST.

DCACHE_ADMIN : Host name of the server node which manages the pool of nodes.



DPMDATA : Directory where the data is stored (absolute path, e.g./storage).

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

GLOBUS_TCP_PORT_RANGE : Port range for Globus IO.

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

GRID_TRUSTED_BROKERS : List of the DNs of the Resource Brokers host certificates which are trusted by the Proxy node (ex: /O=Grid/O=CERN/OU=cern.ch/CN=host/testbed013.cern.ch).

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.

LFC_CENTRAL : A list of VOs for which the LFC should be configured as a central catalogue.

LFC_HOST : Set this if you are building an LFC_HOST, not if you're just using clients.

LFC_LOCAL : Normally the LFC will support all VOs in the VOS variable. If you want to limit this list, add the ones you need to LFC_LOCAL. For each item listed in the VOS variable you need to create a set of new variables as follows:

VO_<VO-NAME>_QUEUES : The queues that the VO can use on the CE.

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

VO_<VO-NAME>_STORAGE_DIR : Mount point on the Storage Element for 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.

MON_HOST : MON Box Hostname.

PX_HOST : PX hostname.

QUEUES : The name of the queues for the CE. These are by default set as the VO names.

RB_HOST : Resource Broker Hostname.

REG_HOST : RGMA Registry hostname.



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

SITE_EMAIL : The e-mail address as published by the information system.

SITE_LAT : Site latitude.

SITE_LOC : "City, Country".

SITE_LONG : Site longitude.

SITE_NAME : Your GIIS.

SITE_SUPPORT_SITE : Support entry point ; Unique Id for the site in the GOC DB and information system.

SITE_TIER : Site tier.

SITE_WEB : Site site.

TORQUE_SERVER : Set this if your torque server is on a different host from the CE. It is ignored 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.

VO_SW_DIR : Directory for installation of experiment software.



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 dynamically 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 16.1..



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 16.2..



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 16.3..



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 16.4..



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 16.5..



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 16.6..



9. SET-UP HOST CERTIFICATES

Author(s): Retico, Antonio

Email : support-lcg-manual-install@cern.ch

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

The PX node requires the host certificate/key files to be put in place before you start the installation.

Contact your national Certification Authority (CA) to understand how to obtain a host certificate if you do not have one already.

Instruction to obtain a CA list can be found in

<http://markusw.home.cern.ch/markusw/lcg2CAlist.html>

From the CA list so obtained you should choose a CA close to you.

Once you have obtained a valid certificate, i.e. a file

hostcert.pem

containing the machine public key and a file

hostkey.pem

containing the machine private key, make sure to place the two files into the directory

/etc/grid-security

with the following permissions

```
> chmod 400 /etc/grid-security/hostkey.pem
```

```
> chmod 644 /etc/grid-security/hostcert.pem
```

It is IMPORTANT that permissions be set as shown, as otherwise certification errors will occur.

If the certificates don't exist, the function exits with an error message and the calling process is interrupted.

9.1. SPECIFICATION OF FUNCTION: CONFIG_HOST_CERTS

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



The original code of the function can be found in:

`/opt/lcg/yaim/functions/config_host_certs`

The code is reproduced also in 16.7..



10. CREATE EDG USERS

Author(s): Retico, Antonio

Email : support-lcg-manual-install@cern.ch

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

Many of the services running on LCG service nodes are owned by the user *edguser*. The user *edguser* belongs to the group *edguser* and it has got a home directory in */home*.

The user *edginfo* is required on all the nodes publishing information on the Information System. The user belongs to the group *edginfo* and it has got a home directory in */home*.

No special requirements exists for the ID of the above mentioned users and groups.

The function creates both *edguser* and *edginfo* groups and users.

- group *edguser*: the group is created with group ID 995.
- user *edguser*: the user is created with group ID 995 and its home is */home/edguser*.
- group *edginfo*: the group is created with group ID 999.
- user *edginfo*: the user is created with group ID 999 and its home is */home/edguser*.

10.1. SPECIFICATION OF FUNCTION: CONFIG_EDGUSERS

The function '*config_edgusers*' 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_edgusers`

The code is reproduced also in 16.8..



11. 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`.

11.1. 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 16.9..



12. 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> --scheme=<SCHEME>
```

`<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`.

12.1. 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 16.10..



13. SET-UP GENERIC INFORMATION PROVIDER

Author(s): Vidic, Valentin

Email : support-lcg-manual-install@cern.ch

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

Generic Information Provider (GIP) is configured through `<INSTALL_ROOT>/lcg/var/gip/lcg-info-generic.conf`. The start of this file is common for all types of nodes:

```
ldif_file=<INSTALL_ROOT>/lcg/var/gip/lcg-info-static.ldif
generic_script=<INSTALL_ROOT>/lcg/libexec/lcg-info-generic
wrapper_script=<INSTALL_ROOT>/lcg/libexec/lcg-info-wrapper
temp_path=<INSTALL_ROOT>/lcg/var/gip/tmp
template=<INSTALL_ROOT>/lcg/etc/GlueSite.template
template=<INSTALL_ROOT>/lcg/etc/GlueCE.template
template=<INSTALL_ROOT>/lcg/etc/GlueCESEBind.template
template=<INSTALL_ROOT>/lcg/etc/GlueSE.template
template=<INSTALL_ROOT>/lcg/etc/GlueService.template

# Common for all
GlueInformationServiceURL: ldap://<hostname>:2135/mds-vo-name=local,o=grid
```

`<hostname>` is determined by running `hostname -f`.

For CE the following is added:

```
dn: GlueSiteUniqueID=<SITE_NAME>,mds-vo-name=local,o=grid
GlueSiteName: <SITE_NAME>
GlueSiteDescription: LCG Site
GlueSiteUserSupportContact: mailto: <SITE_EMAIL>
GlueSiteSysAdminContact: mailto: <SITE_EMAIL>
GlueSiteSecurityContact: mailto: <SITE_EMAIL>
GlueSiteLocation: <SITE_LOC>
GlueSiteLatitude: <SITE_LAT>
GlueSiteLongitude: <SITE_LONG>
GlueSiteWeb: <SITE_WEB>
GlueSiteOtherInfo: <SITE_TIER>
GlueSiteOtherInfo: <SITE_SUPPORT_SITE>
GlueForeignKey: GlueSiteUniqueID=<SITE_NAME>
GlueForeignKey: GlueClusterUniqueID=<CE_HOST>
GlueForeignKey: GlueSEUniqueID=<SE_HOST>

dynamic_script=<INSTALL_ROOT>/lcg/libexec/lcg-info-dynamic-ce
dynamic_script=<INSTALL_ROOT>/lcg/libexec/lcg-info-dynamic-software <INSTALL_ROOT>/lcg/var/gip/lcg-info-generic.conf

# CE Information Provider
GlueCEHostingCluster: <CE_HOST>
GlueCEInfoGatekeeperPort: 2119
GlueCEInfoHostName: <CE_HOST>
GlueCEInfoLRMSType: <CE_BATCH_SYS>
GlueCEInfoLRMSVersion: not defined
GlueCEInfoTotalCPUs: 0
```



```
GlueCEPolicyMaxCPUTime: 0
GlueCEPolicyMaxRunningJobs: 0
GlueCEPolicyMaxTotalJobs: 0
GlueCEPolicyMaxWallClockTime: 0
GlueCEPolicyPriority: 1
GlueCEStateEstimatedResponseTime: 0
GlueCEStateFreeCPUs: 0
GlueCEStateRunningJobs: 0
GlueCEStateStatus: Production
GlueCEStateTotalJobs: 0
GlueCEStateWaitingJobs: 0
GlueCEStateWorstResponseTime: 0
GlueHostApplicationSoftwareRunTimeEnvironment: <ce_runtimeenv>
GlueHostArchitectureSMPSize: <CE_SMP_SIZE>
GlueHostBenchmarkSF00: <CE_SF00>
GlueHostBenchmarkSI00: <CE_SI00>
GlueHostMainMemoryRAMSize: <CE_MINPHYSMEM>
GlueHostMainMemoryVirtualSize: <CE_MINVIRTMEM>
GlueHostNetworkAdapterInboundIP: <CE_INBOUNDIP>
GlueHostNetworkAdapterOutboundIP: <CE_OUTBOUNDIP>
GlueHostOperatingSystemName: <CE_OS>
GlueHostOperatingSystemRelease: <CE_OS_RELEASE>
GlueHostOperatingSystemVersion: 3
GlueHostProcessorClockSpeed: <CE_CPU_SPEED>
GlueHostProcessorModel: <CE_CPU_MODEL>
GlueHostProcessorVendor: <CE_CPU_VENDOR>
GlueSubClusterPhysicalCPUs: 0
GlueSubClusterLogicalCPUs: 0
GlueSubClusterTmpDir: /tmp
GlueSubClusterWNTmpDir: /tmp
GlueCEInfoJobManager: <JOB_MANAGER>
GlueCEStateFreeJobSlots: 0
GlueCEPolicyAssignedJobSlots: 0
GlueCESEBindMountInfo: none
GlueCESEBindWeight: 0

dn: GlueClusterUniqueID=<CE_HOST>, mds-vo-name=local,o=grid
GlueClusterName: <CE_HOST>
GlueForeignKey: GlueSiteUniqueID=<SITE_NAME>
GlueClusterService: <CE_HOST>:2119/jobmanager-<JOB_MANAGER>-<queue>
GlueForeignKey: GlueCEUniqueID=<CE_HOST>:2119/jobmanager-<JOB_MANAGER>-<queue>

dn: GlueSubClusterUniqueID=<CE_HOST>, GlueClusterUniqueID=<CE_HOST>, mds-vo-name=local,o=grid
GlueChunkKey: GlueClusterUniqueID=<CE_HOST>
GlueSubClusterName: <CE_HOST>

dn: GlueCEUniqueID=<CE_HOST>:2119/jobmanager-<JOB_MANAGER>-<queue>, mds-vo-name=local,o=grid
GlueCEName: <queue>
GlueForeignKey: GlueClusterUniqueID=<CE_HOST>
GlueCEInfoContactString: <CE_HOST>:2119/jobmanager-<JOB_MANAGER>-<queue>
GlueCEAccessControlBaseRule: VO:<vo>

dn: GlueVOViewLocalID=<vo>,GlueCEUniqueID=<CE_HOST>:2119/jobmanager-<JOB_MANAGER>-<queue>, mds-vo-name=local,o=grid
GlueCEAccessControlBaseRule: VO:<vo>
```



```
GlueCEInfoDefaultSE: <VO_<vo>_DEFAULT_SE>
GlueCEInfoApplicationDir: <VO_<vo>_SW_DIR>
GlueCEInfoDataDir: <VO_<vo>_STORAGE_DIR>
GlueChunkKey: GlueCEUniqueID=<CE_HOST>:2119/jobmanager-<JOB_MANAGER>-<queue>
```

```
dn: GlueCESEBindGroupCEUniqueID=<CE_HOST>:2119/jobmanager-<JOB_MANAGER>-<queue>, mds-vo-name=local,o=grid
GlueCESEBindGroupSEUniqueID: <se_list>
```

```
dn: GlueCESEBindSEUniqueID=<se>, GlueCESEBindGroupCEUniqueID=<CE_HOST>:2119/jobmanager-<JOB_MANAGER>-<queue>, mds-vo-name=local,o=grid
GlueCESEBindCEAccesspoint: <accesspoint>
GlueCESEBindCEUniqueID: <CE_HOST>:2119/jobmanager-<JOB_MANAGER>-<queue>
```

where *<accesspoint>* is:

- *<DPMDATA>* for DPM SE
- */storage* for dCache
- *<CLASSIC_STORAGE_DIR>* for SE classic.

Some lines can be generated multiple times for different *<vo>*s, *<queue>*s, *<se>*s etc.

For each of the supported VOs, a directory is created in *<INSTALL_ROOT>/edg/var/info/<vo>*. These are used by SGMs to publish information on experiment software installed on the cluster.

For the nodes running GridICE server (usually SE) the following is added:

```
dn: GlueServiceUniqueID=<GRIDICE_SERVER_HOST>:2136,Mds-vo-name=local,o=grid
GlueServiceName: <SITE_NAME>-gridice
GlueServiceType: gridice
GlueServiceVersion: 1.1.0
GlueServiceEndpoint: ldap://<GRIDICE_SERVER_HOST>:2136/mds-vo-name=local,o=grid
GlueServiceStatus: OK
GlueServiceStatusInfo: No Problems
GlueServiceStartTime: 2002-10-09T19:00:00Z
GlueServiceOwner: LCG
GlueForeignKey: GlueSiteUniqueID=<SITE_NAME>
GlueServiceAccessControlRule:<vo>
```

For PX nodes the following is added:

```
dn: GlueServiceUniqueID=<PX_HOST>:7512,Mds-vo-name=local,o=grid
GlueServiceName: <SITE_NAME>-myproxy
GlueServiceType: myproxy
GlueServiceVersion: 1.1.0
GlueServiceEndpoint: <PX_HOST>:7512
GlueServiceStatus: OK
GlueServiceStatusInfo: No Problems
GlueServiceStartTime: 2002-10-09T19:00:00Z
GlueServiceOwner: LCG
GlueForeignKey: GlueSiteUniqueID=<SITE_NAME>
GlueServiceAccessControlRule: <grid_trusted_broker>
```

For nodes running RB the following is added:



```
dn: GlueServiceUniqueID=<RB_HOST>:7772,Mds-vo-name=local,o=grid
GlueServiceName: <SITE_NAME>-rb
GlueServiceType: ResourceBroker
GlueServiceVersion: 1.2.0
GlueServiceEndpoint: <RB_HOST>:7772
GlueServiceStatus: OK
GlueServiceStatusInfo: No Problems
GlueServiceStartTime: 2002-10-09T19:00:00Z
GlueServiceOwner: LCG
GlueForeignKey: GlueSiteUniqueID=<SITE_NAME>
GlueServiceAccessControlRule: <vo>
```

```
dn: GlueServiceDataKey=HeldJobs,GlueServiceUniqueID=gram://<RB_HOST>:7772,Mds-vo-name=local,o=grid
GlueServiceDataKey: HeldJobs
GlueServiceDataValue: 0
GlueChunkKey: GlueServiceUniqueID=gram://<RB_HOST>:7772
```

```
dn: GlueServiceDataKey=IdleJobs,GlueServiceUniqueID=gram://<RB_HOST>:7772,Mds-vo-name=local,o=grid
GlueServiceDataKey: IdleJobs
GlueServiceDataValue: 0
GlueChunkKey: GlueServiceUniqueID=gram://<RB_HOST>:7772
```

```
dn: GlueServiceDataKey=JobController,GlueServiceUniqueID=gram://<RB_HOST>:7772,Mds-vo-name=local,o=grid
GlueServiceDataKey: JobController
GlueServiceDataValue: 0
GlueChunkKey: GlueServiceUniqueID=gram://<RB_HOST>:7772
```

```
dn: GlueServiceDataKey=Jobs,GlueServiceUniqueID=gram://<RB_HOST>:7772,Mds-vo-name=local,o=grid
GlueServiceDataKey: Jobs
GlueServiceDataValue: 0
GlueChunkKey: GlueServiceUniqueID=gram://<RB_HOST>:7772
```

```
dn: GlueServiceDataKey=LogMonitor,GlueServiceUniqueID=gram://<RB_HOST>:7772,Mds-vo-name=local,o=grid
GlueServiceDataKey: LogMonitor
GlueServiceDataValue: 0
GlueChunkKey: GlueServiceUniqueID=gram://<RB_HOST>:7772
```

```
dn: GlueServiceDataKey=RunningJobs,GlueServiceUniqueID=gram://<RB_HOST>:7772,Mds-vo-name=local,o=grid
GlueServiceDataKey: RunningJobs
GlueServiceDataValue: 14
GlueChunkKey: GlueServiceUniqueID=gram://<RB_HOST>:7772
```

```
dn: GlueServiceDataKey=WorkloadManager,GlueServiceUniqueID=gram://<RB_HOST>:7772,Mds-vo-name=local,o=grid
GlueServiceDataKey: WorkloadManager
GlueServiceDataValue: 0
GlueChunkKey: GlueServiceUniqueID=gram://<RB_HOST>:7772
```

For central LFC the following is added:

```
dn: GlueServiceUniqueID=http://<LFC_HOST>:8085/,mds-vo-name=local,o=grid
GlueServiceName: <SITE_NAME>-lfc-dli
GlueServiceType: data-location-interface
GlueServiceVersion: 1.0.0
GlueServiceEndpoint: http://<LFC_HOST>:8085/
GlueServiceURI: http://<LFC_HOST>:8085/
```



```
GlueServiceAccessPointURL: http://<LFC_HOST>:8085/
GlueServiceStatus: running
GlueForeignKey: GlueSiteUniqueID=<SITE_NAME>
GlueServiceOwner: <vo>
GlueServiceAccessControlRule: <vo>

dn: GlueServiceUniqueID=<LFC_HOST>,mds-vo-name=local,o=grid
GlueServiceName: <SITE_NAME>-lfc
GlueServiceType: lcg-file-catalog
GlueServiceVersion: 1.0.0
GlueServiceEndpoint: <LFC_HOST>
GlueServiceURI: <LFC_HOST>
GlueServiceAccessPointURL: <LFC_HOST>
GlueServiceStatus: running
GlueForeignKey: GlueSiteUniqueID=<SITE_NAME>
GlueServiceOwner: <vo>
GlueServiceAccessControlRule: <vo>
```

For local LFC the following is added:

```
dn: GlueServiceUniqueID=<LFC_HOST>,mds-vo-name=local,o=grid
GlueServiceName: <SITE_NAME>-lfc
GlueServiceType: lcg-local-file-catalog
GlueServiceVersion: 1.0.0
GlueServiceEndpoint: <LFC_HOST>
GlueServiceURI: <LFC_HOST>
GlueServiceAccessPointURL: <LFC_HOST>
GlueServiceStatus: running
GlueForeignKey: GlueSiteUniqueID=<SITE_NAME>
GlueServiceOwner: <vo>
GlueServiceAccessControlRule: <vo>
```

For dcache and dpm nodes the following is added:

```
dn: GlueServiceUniqueID=httpg://<SE_HOST>:8443/srm/managerv1,Mds-Vo-name=local,o=grid
GlueServiceAccessPointURL: httpg://<SE_HOST>:8443/srm/managerv1
GlueServiceEndpoint: httpg://<SE_HOST>:8443/srm/managerv1
GlueServiceType: srm_v1
GlueServiceURI: httpg://<SE_HOST>:8443/srm/managerv1
GlueServicePrimaryOwnerName: LCG
GlueServicePrimaryOwnerContact: mailto:<SITE_EMAIL>
GlueForeignKey: GlueSiteUniqueID=<SITE_NAME>
GlueServiceVersion: 1.0.0
GlueServiceAccessControlRule: <vo>
GlueServiceInformationServiceURL: MDS2GRIS:ldap://<BDII_HOST>:2170/mds-voname=local,mds-vo-name=<SITE_NAME>,mds-vo-
GlueServiceStatus: running
```

For all types of SE the following is added:

```
dynamic_script=<INSTALL_ROOT>/lcg/libexec/lcg-info-dynamic-se

GlueSEType: <se_type>
GlueSEPort: 2811
GlueSESizeTotal: 0
GlueSESizeFree: 0
```



```
GlueSEArchitecture: <se_type>
GlueSAPType: permanent
GlueSAPolicyFileLifeTime: permanent
GlueSAPolicyMaxFileSize: 10000
GlueSAPolicyMinFileSize: 1
GlueSAPolicyMaxData: 100
GlueSAPolicyMaxNumFiles: 10
GlueSAPolicyMaxPinDuration: 10
GlueSAPolicyQuota: 0
GlueSAStateAvailableSpace: 1
GlueSAStateUsedSpace: 1
```

```
dn: GlueSEUniqueID=<SE_HOST>,mds-vo-name=local,o=grid
GlueSEName: <SITE_NAME>:<se_type>
GlueForeignKey: GlueSiteUniqueID=<SITE_NAME>
```

```
dn: GlueSEAccessProtocolLocalID=gsiftp, GlueSEUniqueID=<SE_HOST>,Mds-Vo-name=local,o=grid
GlueSEAccessProtocolType: gsiftp
GlueSEAccessProtocolPort: 2811
GlueSEAccessProtocolVersion: 1.0.0
GlueSEAccessProtocolSupportedSecurity: GSI
GlueChunkKey: GlueSEUniqueID=<SE_HOST>
```

```
dn: GlueSEAccessProtocolLocalID=rftio, GlueSEUniqueID=<SE_HOST>,Mds-Vo-name=local,o=grid
GlueSEAccessProtocolType: rftio
GlueSEAccessProtocolPort: 5001
GlueSEAccessProtocolVersion: 1.0.0
GlueSEAccessProtocolSupportedSecurity: RFIO
GlueChunkKey: GlueSEUniqueID=<SE_HOST>
```

where *<se_type>* is *srm_v1* for DPM and *dCache* and *disk* otherwise.

For *SE_dpm* the following is added:

```
dn: GlueSALocalID=<vo>,GlueSEUniqueID=<SE_HOST>,Mds-Vo-name=local,o=grid
GlueSARoot: <vo>:/dpm/<domain>/home/<vo>
GlueSAPath: <vo>:/dpm/<domain>/home/<vo>
GlueSAAccessControlBaseRule: <vo>
GlueChunkKey: GlueSEUniqueID=<SE_HOST>
```

For *SE_dcache* the following is added:

```
dn: GlueSALocalID=<vo>,GlueSEUniqueID=<SE_HOST>,Mds-Vo-name=local,o=grid
GlueSARoot: <vo>:/pnfs/<domain>/home/<vo>
GlueSAPath: <vo>:/pnfs/<domain>/home/<vo>
GlueSAAccessControlBaseRule: <vo>
GlueChunkKey: GlueSEUniqueID=<SE_HOST>
```

For other types of SE the following is used:

```
dn: GlueSALocalID=<vo>,GlueSEUniqueID=<SE_HOST>,Mds-Vo-name=local,o=grid
GlueSARoot: <vo>:<vo>
GlueSAPath: <VO_<vo>_STORAGE_DIR>
GlueSAAccessControlBaseRule: <vo>
GlueChunkKey: GlueSEUniqueID=<SE_HOST>
```




For VOBOX the following is added:

```
dn: GlueServiceUniqueID=gsissh://<VOBOX_HOST>:<VOBOX_PORT>,Mds-vo-name=local,o=grid
GlueServiceAccessPointURL: gsissh://<VOBOX_HOST>:<VOBOX_PORT>
GlueServiceName: <SITE_NAME>-vobox
GlueServiceType: VOBOX
GlueServiceEndpoint: gsissh://<VOBOX_HOST>:<VOBOX_PORT>
GlueServicePrimaryOwnerName: LCG
GlueServicePrimaryOwnerContact: <SITE_EMAIL>
GlueForeignKey: GlueSiteUniqueID=<SITE_NAME>
GlueServiceVersion: 1.0.0
GlueServiceInformationServiceURL: ldap://<VOBOX_HOST>:2135/mds-vo-name=local,o=grid
GlueServiceStatus: running
GlueServiceAccessControlRule: <vo>
```

Configuration script is run:

```
<INSTALL_ROOT>/lcg/sbin/lcg-info-generic-config <INSTALL_ROOT>/lcg/var/gip/lcg-info-generic.conf
```

Configuration script generates a ldif file (*<INSTALL_ROOT>/lcg/var/gip/lcg-info-static.ldif*) by merging templates from *<INSTALL_ROOT>/lcg/etc/* and data from *<INSTALL_ROOT>/lcg/var/gip/lcg-info-generic.conf*. Wrapper script is also created in *<INSTALL_ROOT>/lcg/libexec/lcg-info-wrapper*.

<INSTALL_ROOT>/globus/libexec/edg.info is created:

```
#!/bin/bash
#
# info-globus-ldif.sh
#
#Configures information providers for MDS
#
cat << EOF

dn: Mds-Vo-name=local,o=grid
objectclass: GlobusTop
objectclass: GlobusActiveObject
objectclass: GlobusActiveSearch
type: exec
path: <INSTALL_ROOT>/lcg/libexec
base: lcg-info-wrapper
args:
cachetime: 60
timelimit: 20
sizelimit: 250
```

EOF

<INSTALL_ROOT>/globus/libexec/edg.info is created:

```
#!/bin/bash

cat <<EOF
<INSTALL_ROOT>/globus/etc/openldap/schema/core.schema
<INSTALL_ROOT>/glue/schema/ldap/Glue-CORE.schema
<INSTALL_ROOT>/glue/schema/ldap/Glue-CE.schema
```



```
<INSTALL_ROOT>/glue/schema/ldap/Glue-CESEBind.schema  
<INSTALL_ROOT>/glue/schema/ldap/Glue-SE.schema  
EOF
```

These two scripts are used to generate *slapd* configuration for Globus MDS.

`<INSTALL_ROOT>/lcg/libexec/lcg-info-dynamic-ce` is generated to call the information provider appropriate for the LRMS. For Torque the file has these contents:

```
#!/bin/sh  
<INSTALL_ROOT>/lcg/libexec/lcg-info-dynamic-pbs <INSTALL_ROOT>/lcg/var/gip/lcg-info-generic.conf <TORQUE_SERVER>
```

R-GMA GIN periodically queries MDS and inserts the data into R-GMA. GIN is configured on all nodes except UI and WN by copying host certificate to `<INSTALL_ROOT>/glite/var/rgma/certs` and updating the configuration file appropriately (`<INSTALL_ROOT>/glite/etc/rgma/ClientAuthentication.props`). Finally, GIN configuration script (`<INSTALL_ROOT>/glite/bin/rgma-gin-config`) is run to configure the mapping between Glue schema in MDS and Glue tables in R-GMA. *rgma-gin* service is restarted and configured to start on boot.

13.1. SPECIFICATION OF FUNCTION: CONFIG_GIP

The function `'config_gip'` needs the following variables to be set in the configuration file:

BDII_HOST : BDII Hostname.

CE_BATCH_SYS : Implementation of site batch system. Available values are “torque”, “lsf”, “pbs”, “condor” etc.

CE_CPU_MODEL : Model of the CPU used by the WN (WN specification). This parameter is a string whose domain is not defined yet in the GLUE Schema. The value used for Pentium III is “PIII”.

CE_CPU_SPEED : Clock frequency in Mhz (WN specification).

CE_CPU_VENDOR : Vendor of the CPU. used by the WN (WN specification). This parameter is a string whose domain is not defined yet in the GLUE Schema. The value used for Intel is “intel”.

CE_HOST : Computing Element Hostname.

CE_INBOUNDIP : TRUE if inbound connectivity is enabled at your site, FALSE otherwise (WN specification).

CE_MINPHYSMEM : RAM size in kblocks (WN specification).

CE_MINVIRTMEM : Virtual Memory size in kblocks (WN specification).

CE_OS : Operating System name (WN specification).

CE_OS_RELEASE : Operating System release (WN specification).

CE_OUTBOUNDIP : TRUE if outbound connectivity is enabled at your site, FALSE otherwise (WN specification).



CE_RUNTIMEENV : List of software tags supported by the site. The list can include VO-specific software tags. In order to assure backward compatibility it should include the entry 'LCG-2', the current middleware version and the list of previous middleware tags.

CE_SF00 : Performance index of your fabric in SpecFloat 2000 (WN specification). For some examples of Spec values see <http://www.specbench.org/osg/cpu2000/results/cint2000.html>.

CE_SI00 : Performance index of your fabric in SpecInt 2000 (WN specification). For some examples of Spec values see <http://www.specbench.org/osg/cpu2000/results/cint2000.html>.

CE_SMPSIZE : Number of cpus in an SMP box (WN specification).

CLASSIC_HOST : The name of your SE_classic host.

CLASSIC_STORAGE_DIR : The root storage directory on CLASSIC_HOST.

DCACHE_ADMIN : Host name of the server node which manages the pool of nodes.

DPMDATA : Directory where the data is stored (absolute path, e.g./storage).

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

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

GRID_TRUSTED_BROKERS : List of the DNs of the Resource Brokers host certificates which are trusted by the Proxy node (ex: /O=Grid/O=CERN/OU=cern.ch/CN=host/testbed013.cern.ch).

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

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

LFC_CENTRAL : A list of VOs for which the LFC should be configured as a central catalogue.

LFC_HOST : Set this if you are building an LFC_HOST, not if you're just using clients.

LFC_LOCAL : Normally the LFC will support all VOs in the VOS variable. If you want to limit this list, add the ones you need to LFC_LOCAL. For each item listed in the VOS variable you need to create a set of new variables as follows:

VO_<VO-NAME>_QUEUES : The queues that the VO can use on the CE.

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

VO_<VO-NAME>_STORAGE_DIR : Mount point on the Storage Element for 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.

PX_HOST : PX hostname.

QUEUES : The name of the queues for the CE. These are by default set as the VO names.

RB_HOST : Resource Broker Hostname.

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

SITE_EMAIL : The e-mail address as published by the information system.

SITE_LAT : Site latitude.

SITE_LOC : "City, Country".

SITE_LONG : Site longitude.

SITE_NAME : Your GIIS.

SITE_SUPPORT_SITE : Support entry point ; Unique Id for the site in the GOC DB and information system.

SITE_TIER : Site tier.

SITE_WEB : Site site.

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

VOBOX_HOST : VOBOX hostname.

VOBOX_PORT : The port the VOBOX gsisshd listens on.

VOS : List of supported VOs.

VO_SW_DIR : Directory for installation of experiment software.

The original code of the function can be found in:

`/opt/lcg/yaim/functions/config_gip`

The code is also reproduced in 16.11..



14. 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).

14.1. 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 16.12..



15. SET-UP MYPROXY

Author(s): Vidic, Valentin

Email : support-lcg-manual-install@cern.ch

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

<*INSTALL_ROOT*>/edg/etc/edg-myproxy.conf is created with a list of trusted RBs (allowed to renew user certificates) defined in variable <*GRID_TRUSTED_BROKERS*>.

myproxy service is restarted and configured to start on boot.

myproxy server uses the configuration in */etc/myproxy-server.config*. This configuration file is generated from <*INSTALL_ROOT*>/edg/etc/edg-myproxy.conf and *signing_policy* files in */etc/grid-security/certificates* by *myproxy* startup script. *signing_policy* files define users that are allowed to store credentials in *myproxy*.

15.1. SPECIFICATION OF FUNCTION: CONFIG_PROXY_SERVER

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

GRID_TRUSTED_BROKERS : List of the DNs of the Resource Brokers host certificates which are trusted by the Proxy node (ex: /O=Grid/O=CERN/OU=cern.ch/CN=host/testbed013.cern.ch).

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_proxy_server`

The code is also reproduced in 16.13..



16. SOURCE CODE

16.1. 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  
}
```

16.2. 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
}
```

16.3. 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
}
```

16.4. 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
}
```

16.5. 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/log/edg-fetch-crl-cron.log 2>&1
```

```
    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 >> /var/log/edg-fetch-crl-cron.log 2>&1
```

```
    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  
}
```

16.6. 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 rfiiod restart  
  
elif ( echo "${NODE_TYPE_LIST}" | grep -q SE_dpms ); then  
  
    return 0  
  
elif ( rpm -qa | grep -q CASTOR-client ); then  
  
    /sbin/service rfiiod stop  
    /sbin/chkconfig --level 2345 rfiiod off  
  
fi  
  
return 0  
}
```

16.7. CONFIG_HOST_CERTS

```
config_host_certs(){  
  
if [ -f /etc/grid-security/hostkey.pem ]; then  
    chmod 400 /etc/grid-security/hostkey.pem  
elif [ -f /etc/grid-security/hostcert.pem ]; then  
    chmod 644 /etc/grid-security/hostcert.pem  
else
```



```
    echo "Please copy the hostkey.pem and hostcert.pem to /etc/grid-security"
    return 1
fi
return 0
}
```

16.8. CONFIG_EDGUSERS

```
config_edgusers(){
INSTALL_ROOT=${INSTALL_ROOT:-/opt}

check_users_conf_format

if ( ! id edguser > /dev/null 2>&1 ); then
    useradd -r -c "EDG User" edguser
    mkdir -p /home/edguser
    chown edguser:edguser /home/edguser
fi

if ( ! id edginfo > /dev/null 2>&1 ); then
    useradd -r -c "EDG Info user" edginfo
    mkdir -p /home/edginfo
    chown edginfo:edginfo /home/edginfo
fi

if ( ! id rgma > /dev/null 2>&1 ); then
    useradd -r -c "RGMA user" -m -d ${INSTALL_ROOT}/glite/etc/rgma rgma
fi

# Make sure edguser is a member of each group

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
# On some nodes the users are not created, so the group will not exist
# Isn't there a better way to check for group existence??
if ( grep "^${groupname}:" /etc/group > /dev/null ); then
    gpasswd -a edguser $groupname > /dev/null
fi
fi
done

return 0
}
```

16.9. 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
```

```
}
```

16.10. 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
}
```

16.11. CONFIG_GIP

```
config_gip () {
    INSTALL_ROOT=${INSTALL_ROOT:-/opt}

    requires CE_HOST RB_HOST PX_HOST
```



```
#check_users_conf_format

#set some vars for storage elements
if ( echo "${NODE_TYPE_LIST}" | grep '\<SE' > /dev/null ); then
    requires VOS SITE_EMAIL SITE_NAME BDII_HOST VOS SITE_NAME
    if ( echo "${NODE_TYPE_LIST}" | grep SE_dpm > /dev/null ); then
requires DPM_HOST
se_host=$DPM_HOST
se_type="srm_v1"
control_protocol=srm_v1
control_endpoint=httpg://${se_host}
    elif ( echo "${NODE_TYPE_LIST}" | grep SE_dcache > /dev/null ); then
requires DCACHE_ADMIN
se_host=$DCACHE_ADMIN
se_type="srm_v1"
control_protocol=srm_v1
control_endpoint=httpg://${se_host}
    else
requires CLASSIC_STORAGE_DIR CLASSIC_HOST VO__STORAGE_DIR
se_host=$CLASSIC_HOST
se_type="disk"
control_protocol=classic
control_endpoint=classic
    fi
fi

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

    # GlueSite

    requires SITE_EMAIL SITE_NAME SITE_LOC SITE_LAT SITE_LONG SITE_WEB \
SITE_TIER SITE_SUPPORT_SITE SE_LIST

    outfile=$INSTALL_ROOT/lcg/var/gip/lcg-info-static-site.conf

    # set default SEs if they're currently undefined
    default_se='set x $SE_LIST; echo "$2"'
    if [ "$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

    cat << EOF > $outfile
dn: GlueSiteUniqueID=$SITE_NAME
GlueSiteUniqueID: $SITE_NAME
GlueSiteName: $SITE_NAME
GlueSiteDescription: LCG Site
GlueSiteUserSupportContact: mailto: $SITE_EMAIL
GlueSiteSysAdminContact: mailto: $SITE_EMAIL
GlueSiteSecurityContact: mailto: $SITE_EMAIL
```




```
GlueSiteLocation: $$SITE_LOC
GlueSiteLatitude: $$SITE_LAT
GlueSiteLongitude: $$SITE_LONG
GlueSiteWeb: $$SITE_WEB
GlueSiteSponsor: none
GlueSiteOtherInfo: $$SITE_TIER
GlueSiteOtherInfo: $$SITE_SUPPORT_SITE
GlueForeignKey: GlueSiteUniqueID=${SITE_NAME}
EOF

$INSTALL_ROOT/lcg/sbin/lcg-info-static-create -c $outfile -t \
$INSTALL_ROOT/lcg/etc/GlueSite.template > \
$INSTALL_ROOT/lcg/var/gip/ldif/static-file-Site.ldif

# GlueCluster

requires JOB_MANAGER CE_BATCH_SYS VOS QUEUES CE_BATCH_SYS CE_CPU_MODEL \
CE_CPU_VENDOR CE_CPU_SPEED CE_OS CE_OS_RELEASE CE_MINPHYSMEM \
CE_MINVIRTMEM CE_SMP_SIZE CE_SI00 CE_SF00 CE_OUTBOUNDIP CE_INBOUNDIP \
CE_RUNTIMEENV

outfile=$INSTALL_ROOT/lcg/var/gip/lcg-info-static-cluster.conf

for VO in $VOS; do
    dir=${INSTALL_ROOT}/edg/var/info/$VO
    mkdir -p $dir
f=$dir/$VO.list
[ -f $f ] || touch $f
    # work out the sgm user for this VO
    sgmuser=`users_getsgmuser $VO`
sgmgroup=`id -g $sgmuser`
chown -R ${sgmuser}:${sgmgroup} $dir
chmod -R go-w $dir
done

cat <<EOF > $outfile

dn: GlueClusterUniqueID=${CE_HOST}
GlueClusterName: ${CE_HOST}
GlueForeignKey: GlueSiteUniqueID=${SITE_NAME}
GlueInformationServiceURL: ldap://`hostname -f`:2135/mds-vo-name=local,o=grid
EOF

for QUEUE in $QUEUES; do
    echo "GlueClusterService: ${CE_HOST}:2119/jobmanager-$JOB_MANAGER-$QUEUE" >> $outfile
done

for QUEUE in $QUEUES; do
    echo "GlueForeignKey:" \
"GlueCEUniqueID=${CE_HOST}:2119/jobmanager-$JOB_MANAGER-$QUEUE" >> $outfile
done

cat << EOF >> $outfile
```



```
dn: GlueSubClusterUniqueID=${CE_HOST}, GlueClusterUniqueID=${CE_HOST}
GlueChunkKey: GlueClusterUniqueID=${CE_HOST}
GlueHostArchitectureSMPSize: $CE_SMP_SIZE
GlueHostBenchmarkSF00: $CE_SF00
GlueHostBenchmarkSI00: $CE_SI00
GlueHostMainMemoryRAMSize: $CE_MINPHYSMEM
GlueHostMainMemoryVirtualSize: $CE_MINVIRTMEM
GlueHostNetworkAdapterInboundIP: $CE_INBOUNDIP
GlueHostNetworkAdapterOutboundIP: $CE_OUTBOUNDIP
GlueHostOperatingSystemName: $CE_OS
GlueHostOperatingSystemRelease: $CE_OS_RELEASE
GlueHostOperatingSystemVersion: 3
GlueHostProcessorClockSpeed: $CE_CPU_SPEED
GlueHostProcessorModel: $CE_CPU_MODEL
GlueHostProcessorVendor: $CE_CPU_VENDOR
GlueSubClusterName: ${CE_HOST}
GlueSubClusterPhysicalCPUs: 0
GlueSubClusterLogicalCPUs: 0
GlueSubClusterTmpDir: /tmp
GlueSubClusterWNTmpDir: /tmp
GlueInformationServiceURL: ldap://`hostname -f`:2135/mds-vo-name=local,o=grid
EOF
```

```
for x in $CE_RUNTIMEENV; do
    echo "GlueHostApplicationSoftwareRunTimeEnvironment: $x" >> $outfile
done
```

```
$INSTALL_ROOT/lcg/sbin/lcg-info-static-create -c $outfile -t \
$INSTALL_ROOT/lcg/etc/GlueCluster.template > \
$INSTALL_ROOT/lcg/var/gip/ldif/static-file-Cluster.ldif
```

```
# GlueCE
```

```
outfile=$INSTALL_ROOT/lcg/var/gip/lcg-info-static-ce.conf
```

```
cat /dev/null > $outfile
```

```
for QUEUE in $QUEUES; do
    cat <<EOF >> $outfile
```

```
dn: GlueCEUniqueID=${CE_HOST}:2119/jobmanager-${JOB_MANAGER}-${QUEUE}
GlueCEHostingCluster: ${CE_HOST}
GlueCEName: $QUEUE
GlueCEInfoGatekeeperPort: 2119
GlueCEInfoHostName: ${CE_HOST}
GlueCEInfoLRMSType: $CE_BATCH_SYS
GlueCEInfoLRMSVersion: not defined
GlueCEInfoTotalCPUs: 0
GlueCEInfoJobManager: ${JOB_MANAGER}
GlueCEInfoContactString: ${CE_HOST}:2119/jobmanager-${JOB_MANAGER}-${QUEUE}
GlueCEInfoApplicationDir: ${VO_SW_DIR}
GlueCEInfoDataDir: ${CE_DATADIR:-unset}
GlueCEInfoDefaultSE: $default_se
```



```
GlueCEStateEstimatedResponseTime: 0
GlueCEStateFreeCPUs: 0
GlueCEStateRunningJobs: 0
GlueCEStateStatus: Production
GlueCEStateTotalJobs: 0
GlueCEStateWaitingJobs: 0
GlueCEStateWorstResponseTime: 0
GlueCEStateFreeJobSlots: 0
GlueCEPolicyMaxCPUTime: 0
GlueCEPolicyMaxRunningJobs: 0
GlueCEPolicyMaxTotalJobs: 0
GlueCEPolicyMaxWallClockTime: 0
GlueCEPolicyPriority: 1
GlueCEPolicyAssignedJobSlots: 0
GlueForeignKey: GlueClusterUniqueID=${CE_HOST}
GlueInformationServiceURL: ldap://`hostname -f`:2135/mds-vo-name=local,o=grid
EOF

    for VO in `echo $VOS | tr '[:lower:]' '[:upper:]'`; do
        for VO_QUEUE in `eval echo '$VO_${VO}_QUEUES`; do
            if [ "${QUEUE}" = "${VO_QUEUE}" ]; then
                echo "GlueCEAccessControlBaseRule:" \
"VO:`echo $VO | tr '[:upper:]' '[:lower:]'`" >> $outfile
            fi
        done
    done

for VO in `echo $VOS | tr '[:lower:]' '[:upper:]'`; do
    for VO_QUEUE in `eval echo '$VO_${VO}_QUEUES`; do
        if [ "${QUEUE}" = "${VO_QUEUE}" ]; then
            cat << EOF >> $outfile

dn: GlueVOViewLocalID=`echo $VO | tr '[:upper:]' '[:lower:]'`,\
GlueCEUniqueID=${CE_HOST}:2119/jobmanager-${JOB_MANAGER}-${QUEUE}
GlueCEAccessControlBaseRule: VO:`echo $VO | tr '[:upper:]' '[:lower:]'`
GlueCEStateRunningJobs: 0
GlueCEStateWaitingJobs: 0
GlueCEStateTotalJobs: 0
GlueCEStateFreeJobSlots: 0
GlueCEStateEstimatedResponseTime: 0
GlueCEStateWorstResponseTime: 0
GlueCEInfoDefaultSE: `eval echo '$VO_${VO}_DEFAULT_SE`
GlueCEInfoApplicationDir: `eval echo '$VO_${VO}_SW_DIR`
GlueCEInfoDataDir: ${CE_DATADIR:-unset}
GlueChunkKey: GlueCEUniqueID=${CE_HOST}:2119/jobmanager-${JOB_MANAGER}-${QUEUE}
EOF

                fi
            done
        done

        $INSTALL_ROOT/lcg/sbin/lcg-info-static-create -c $outfile -t \
$INSTALL_ROOT/lcg/etc/GlueCE.template > \
$INSTALL_ROOT/lcg/var/gip/ldif/static-file-CE.ldif
```



```
# GlueCESEBind

outfile=$INSTALL_ROOT/lcg/var/gip/lcg-info-static-cesebind.conf
echo "" > $outfile

for QUEUE in $QUEUES
do
    echo "dn: GlueCESEBindGroupCEUniqueID=${CE_HOST}:2119/jobmanager-$JOB_MANAGER-$QUEUE" \
>> $outfile
    for se in $SE_LIST
    do
        echo "GlueCESEBindGroupSEUniqueID: $se" >> $outfile
    done
done

for se in $SE_LIST; do

case "$se" in
"$DPM_HOST") accesspoint=$DPMDATA;;
"$DCACHE_ADMIN") accesspoint="/pnfs/hostname -d`/data";;
*) accesspoint=$CLASSIC_STORAGE_DIR ;;
esac

    for QUEUE in $QUEUES; do

        cat <<EOF >> $outfile

dn: GlueCESEBindSEUniqueID=$se,\
GlueCESEBindGroupCEUniqueID=${CE_HOST}:2119/jobmanager-$JOB_MANAGER-$QUEUE
GlueCESEBindCEAccesspoint: $accesspoint
GlueCESEBindCEUniqueID: ${CE_HOST}:2119/jobmanager-$JOB_MANAGER-$QUEUE
GlueCESEBindMountInfo: $accesspoint
GlueCESEBindWeight: 0

EOF

        done
    done

    $INSTALL_ROOT/lcg/sbin/lcg-info-static-create -c $outfile -t \
    $INSTALL_ROOT/lcg/etc/GlueCESEBind.template > \
    $INSTALL_ROOT/lcg/var/gip/ldif/static-file-CESEBind.ldif

# Set some vars based on the LRMS

case "$CE_BATCH_SYS" in
condor|CONDOR) plugin="${INSTALL_ROOT}/lcg/libexec/lcg-info-dynamic-condor /opt/condor/bin/ $INSTALL_ROOT/lcg/e
lsf|LSF)      plugin="${INSTALL_ROOT}/lcg/libexec/lcg-info-dynamic-lsf /usr/local/lsf/bin/ $INSTALL_ROOT/lcg/e
pbs|PBS)      plugin="${INSTALL_ROOT}/lcg/libexec/lcg-info-dynamic-pbs /opt/lcg/var/gip/ldif/static-file-CE.l
vo_max_jobs_cmd="";;
*)            plugin="${INSTALL_ROOT}/lcg/libexec/lcg-info-dynamic-pbs /opt/lcg/var/gip/ldif/static-file-CE.l
vo_max_jobs_cmd="$INSTALL_ROOT/lcg/libexec/vomaxjobs-maui";;
```



```
esac

# Configure the dynamic plugin appropriate for the batch sys

cat << EOF > ${INSTALL_ROOT}/lcg/var/gip/plugin/lcg-info-dynamic-ce
#!/bin/sh
$plugin
EOF

chmod +x ${INSTALL_ROOT}/lcg/var/gip/plugin/lcg-info-dynamic-ce

# Configure the ERT plugin

cat << EOF > ${INSTALL_ROOT}/lcg/var/gip/plugin/lcg-info-dynamic-scheduler-wrapper
#!/bin/sh
${INSTALL_ROOT}/lcg/libexec/lcg-info-dynamic-scheduler -c ${INSTALL_ROOT}/lcg/etc/lcg-info-dynamic-scheduler.conf
EOF

chmod +x ${INSTALL_ROOT}/lcg/var/gip/plugin/lcg-info-dynamic-scheduler-wrapper

if ( echo $CE_BATCH_SYS | egrep -qi 'pbs|torque' ); then

cat <<EOF > $INSTALL_ROOT/lcg/etc/lcg-info-dynamic-scheduler.conf
[Main]
static_ldif_file: $INSTALL_ROOT/lcg/var/gip/ldif/static-file-CE.ldif
vomap :
EOF

for vo in $VOS; do
    vo_group=`users_getvogroup $vo`
    if [ $vo_group ]; then
echo "    $vo_group:$vo" >> $INSTALL_ROOT/lcg/etc/lcg-info-dynamic-scheduler.conf
    fi
done

cat <<EOF >> $INSTALL_ROOT/lcg/etc/lcg-info-dynamic-scheduler.conf
module_search_path : ../lrms:../ett
[LRMS]
lrms_backend_cmd: $INSTALL_ROOT/lcg/libexec/lrmsinfo-pbs
[Scheduler]
vo_max_jobs_cmd: $vo_max_jobs_cmd
cycle_time : 0
EOF
    fi

# Configure the provider for installed software

if [ -f $INSTALL_ROOT/lcg/libexec/lcg-info-provider-software ]; then
cat <<EOF > $INSTALL_ROOT/lcg/var/gip/provider/lcg-info-provider-software-wrapper
#!/bin/sh
$INSTALL_ROOT/lcg/libexec/lcg-info-provider-software -p $INSTALL_ROOT/edg/var/info -c $CE_HOST
EOF
chmod +x $INSTALL_ROOT/lcg/var/gip/provider/lcg-info-provider-software-wrapper
fi
```



```
fi #endif for CE_HOST

if [ "$GRIDICE_SERVER_HOST" = "`hostname -f`" ]; then

    requires VOS SITE_NAME SITE_EMAIL

outfile=$INSTALL_ROOT/lcg/var/gip/lcg-info-static-gridice.conf

    cat <<EOF > $outfile

dn: GlueServiceUniqueID=${GRIDICE_SERVER_HOST}:2136
GlueServiceName: ${SITE_NAME}-gridice
GlueServiceType: gridice
GlueServiceVersion: 1.1.0
GlueServiceEndpoint: ldap://${GRIDICE_SERVER_HOST}:2136/mds-vo-name=local,o=grid
GlueServiceURI: unset
GlueServiceAccessPointURL: not_used
GlueServiceStatus: OK
GlueServiceStatusInfo: No Problems
GlueServiceWSDL: unset
GlueServiceSemantics: unset
GlueServiceStartTime: 1970-01-01T00:00:00Z
GlueForeignKey: GlueSiteUniqueID=${SITE_NAME}
EOF

    for VO in $VOS; do
        echo "GlueServiceAccessControlRule: $VO" >> $outfile
    echo "GlueServiceOwner: $VO" >> $outfile
    done

FMON='--fmon=yes'

    $INSTALL_ROOT/lcg/sbin/lcg-info-static-create -c $outfile -t \
$INSTALL_ROOT/lcg/etc/GlueService.template > \
$INSTALL_ROOT/lcg/var/gip/ldif/static-file-GRIDICE.ldif

fi #endif for GRIDICE_SERVER_HOST

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

    requires GRID_TRUSTED_BROKERS SITE_EMAIL SITE_NAME

outfile=$INSTALL_ROOT/lcg/var/gip/lcg-info-static-px.conf

    cat << EOF > $outfile

dn: GlueServiceUniqueID=${PX_HOST}:7512
GlueServiceName: ${SITE_NAME}-myproxy
GlueServiceType: myproxy
GlueServiceVersion: 1.1.0
GlueServiceEndpoint: ${PX_HOST}:7512
GlueServiceURI: unset
```



```
GlueServiceAccessPointURL: myproxy://${PX_HOST}
GlueServiceStatus: OK
GlueServiceStatusInfo: No Problems
GlueServiceWSDL: unset
GlueServiceSemantics: unset
GlueServiceStartTime: 1970-01-01T00:00:00Z
GlueServiceOwner: LCG
GlueForeignKey: GlueSiteUniqueID=${SITE_NAME}
EOF

    split_quoted_variable $GRID_TRUSTED_BROKERS | while read x; do
        echo "GlueServiceAccessControlRule: $x" >> $outfile
    done

    $INSTALL_ROOT/lcg/sbin/lcg-info-static-create -c $outfile -t \
$INSTALL_ROOT/lcg/etc/GlueService.template > \
$INSTALL_ROOT/lcg/var/gip/ldif/static-file-PX.ldif

fi #endif for PX_HOST

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

    requires VOS SITE_EMAIL SITE_NAME

outfile=$INSTALL_ROOT/lcg/var/gip/lcg-info-static-rb.conf

    cat <<EOF > $outfile

dn: GlueServiceUniqueID=${RB_HOST}:7772
GlueServiceName: ${SITE_NAME}-rb
GlueServiceType: ResourceBroker
GlueServiceVersion: 1.2.0
GlueServiceEndpoint: ${RB_HOST}:7772
GlueServiceURI: unset
GlueServiceAccessPointURL: not_used
GlueServiceStatus: OK
GlueServiceStatusInfo: No Problems
GlueServiceWSDL: unset
GlueServiceSemantics: unset
GlueServiceStartTime: 1970-01-01T00:00:00Z
GlueForeignKey: GlueSiteUniqueID=${SITE_NAME}
EOF

    for VO in $VOS; do
        echo "GlueServiceAccessControlRule: $VO" >> $outfile
    echo "GlueServiceOwner: $VO" >> $outfile
    done

    cat <<EOF >> $outfile
dn: GlueServiceDataKey=HeldJobs,GlueServiceUniqueID=gram://${RB_HOST}:7772
GlueServiceDataKey: HeldJobs
GlueServiceDataValue: 0
GlueChunkKey: GlueServiceUniqueID=gram://${RB_HOST}:7772
```



```
dn: GlueServiceDataKey=IdleJobs,GlueServiceUniqueID=gram://{RB_HOST}:7772
GlueServiceDataKey: IdleJobs
GlueServiceDataValue: 0
GlueChunkKey: GlueServiceUniqueID=gram://{RB_HOST}:7772

dn: GlueServiceDataKey=JobController,GlueServiceUniqueID=gram://{RB_HOST}:7772
GlueServiceDataKey: JobController
GlueServiceDataValue: 0
GlueChunkKey: GlueServiceUniqueID=gram://{RB_HOST}:7772

dn: GlueServiceDataKey=Jobs,GlueServiceUniqueID=gram://{RB_HOST}:7772
GlueServiceDataKey: Jobs
GlueServiceDataValue: 0
GlueChunkKey: GlueServiceUniqueID=gram://{RB_HOST}:7772

dn: GlueServiceDataKey=LogMonitor,GlueServiceUniqueID=gram://{RB_HOST}:7772
GlueServiceDataKey: LogMonitor
GlueServiceDataValue: 0
GlueChunkKey: GlueServiceUniqueID=gram://{RB_HOST}:7772

dn: GlueServiceDataKey=RunningJobs,GlueServiceUniqueID=gram://{RB_HOST}:7772
GlueServiceDataKey: RunningJobs
GlueServiceDataValue: 14
GlueChunkKey: GlueServiceUniqueID=gram://{RB_HOST}:7772

dn: GlueServiceDataKey=WorkloadManager,GlueServiceUniqueID=gram://{RB_HOST}:7772
GlueServiceDataKey: WorkloadManager
GlueServiceDataValue: 0
GlueChunkKey: GlueServiceUniqueID=gram://{RB_HOST}:7772
```

EOF

```
    $INSTALL_ROOT/lcg/sbin/lcg-info-static-create -c $outfile -t \
$INSTALL_ROOT/lcg/etc/GlueService.template > \
$INSTALL_ROOT/lcg/var/gip/ldif/static-file-RB.ldif
```

```
fi #endif for RB_HOST
```

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

```
outfile=$INSTALL_ROOT/lcg/var/gip/lcg-info-static-lfc.conf
cat /dev/null > $outfile
```

```
    requires VOS SITE_EMAIL SITE_NAME BDII_HOST LFC_HOST
```

```
    if [ "$LFC_LOCAL" ]; then
```

```
lfc_local=$LFC_LOCAL
```

```
    else
```

```
# populate lfc_local with the VOS which are not set to be central
```

```
unset lfc_local
```

```
for i in $VOS; do
```

```
    if ( ! echo $LFC_CENTRAL | grep -qw $i ); then
```

```
lfc_local="$lfc_local $i"
```




```
fi
done
fi

if [ "$LFC_CENTRAL" ]; then

cat <<EOF >> $outfile
dn: GlueServiceUniqueID=http://${LFC_HOST}:8085/
GlueServiceName: ${SITE_NAME}-lfc-dli
GlueServiceType: data-location-interface
GlueServiceVersion: 1.0.0
GlueServiceEndpoint: http://${LFC_HOST}:8085/
GlueServiceURI: http://${LFC_HOST}:8085/
GlueServiceAccessPointURL: http://${LFC_HOST}:8085/
GlueServiceStatus: OK
GlueServiceStatusInfo: No Problems
GlueServiceWSDL: unset
GlueServiceSemantics: unset
GlueServiceStartTime: 1970-01-01T00:00:00Z
GlueForeignKey: GlueSiteUniqueID=${SITE_NAME}
EOF

for VO in $LFC_CENTRAL; do
    echo "GlueServiceOwner: $VO" >> $outfile
    echo "GlueServiceAccessControlRule: $VO" >> $outfile
done

echo >> $outfile

cat <<EOF >> $outfile
dn: GlueServiceUniqueID=${LFC_HOST}
GlueServiceName: ${SITE_NAME}-lfc
GlueServiceType: lcg-file-catalog
GlueServiceVersion: 1.0.0
GlueServiceEndpoint: ${LFC_HOST}
GlueServiceURI: ${LFC_HOST}
GlueServiceAccessPointURL: ${LFC_HOST}
GlueServiceStatus: OK
GlueServiceStatusInfo: No Problems
GlueServiceWSDL: unset
GlueServiceSemantics: unset
GlueServiceStartTime: 1970-01-01T00:00:00Z
GlueForeignKey: GlueSiteUniqueID=${SITE_NAME}
EOF

for VO in $LFC_CENTRAL; do
    echo "GlueServiceOwner: $VO" >> $outfile
    echo "GlueServiceAccessControlRule: $VO" >> $outfile
done

    echo >> $outfile

fi
```



```
if [ "$lfc_local" ]; then

    cat <<EOF >> $outfile
dn: GlueServiceUniqueID=http://${LFC_HOST}:8085/,o=local
GlueServiceName: ${SITE_NAME}-lfc-dli
GlueServiceType: local-data-location-interface
GlueServiceVersion: 1.0.0
GlueServiceEndpoint: http://${LFC_HOST}:8085/
GlueServiceURI: http://${LFC_HOST}:8085/
GlueServiceAccessPointURL: http://${LFC_HOST}:8085/
GlueServiceStatus: OK
GlueServiceStatusInfo: No Problems
GlueServiceWSDL: unset
GlueServiceSemantics: unset
GlueServiceStartTime: 1970-01-01T00:00:00Z
GlueForeignKey: GlueSiteUniqueID=${SITE_NAME}
EOF

    for VO in $lfc_local; do
        echo "GlueServiceOwner: $VO" >> $outfile
        echo "GlueServiceAccessControlRule: $VO" >> $outfile
    done

    echo >> $outfile

cat <<EOF >> $outfile
dn: GlueServiceUniqueID=${LFC_HOST},o=local
GlueServiceName: ${SITE_NAME}-lfc
GlueServiceType: lcg-local-file-catalog
GlueServiceVersion: 1.0.0
GlueServiceEndpoint: ${LFC_HOST}
GlueServiceURI: ${LFC_HOST}
GlueServiceAccessPointURL: ${LFC_HOST}
GlueServiceStatus: OK
GlueServiceStatusInfo: No Problems
GlueServiceWSDL: unset
GlueServiceSemantics: unset
GlueServiceStartTime: 1970-01-01T00:00:00Z
GlueForeignKey: GlueSiteUniqueID=${SITE_NAME}
EOF

for VO in $lfc_local; do
    echo "GlueServiceOwner: $VO" >> $outfile
    echo "GlueServiceAccessControlRule: $VO" >> $outfile
done

fi

    $INSTALL_ROOT/lcg/sbin/lcg-info-static-create -c $outfile -t \
$INSTALL_ROOT/lcg/etc/GlueService.template > \
$INSTALL_ROOT/lcg/var/gip/ldif/static-file-LFC.ldif

fi # end of LFC
```



```
if ( echo "${NODE_TYPE_LIST}" | egrep -q 'dcache|dpm_(mysql|oracle)' ); then

    outfile=$INSTALL_ROOT/lcg/var/gip/lcg-info-static-dse.conf

    cat <<EOF > $outfile

dn: GlueServiceUniqueID=httpg://${se_host}:8443/srm/managerv1
GlueServiceName: ${SITE_NAME}-srm
GlueServiceType: srm_v1
GlueServiceVersion: 1.0.0
GlueServiceEndpoint: httpg://${se_host}:8443/srm/managerv1
GlueServiceURI: httpg://${se_host}:8443/srm/managerv1
GlueServiceAccessPointURL: httpg://${se_host}:8443/srm/managerv1
GlueServiceStatus: OK
GlueServiceStatusInfo: No Problems
GlueServiceWSDL: unset
GlueServiceSemantics: unset
GlueServiceStartTime: 1970-01-01T00:00:00Z
GlueServiceOwner: LCG
GlueForeignKey: GlueSiteUniqueID=${SITE_NAME}
EOF

    for VO in $VOS; do
echo "GlueServiceAccessControlRule: $VO" >> $outfile
done

    cat <<EOF >> $outfile
GlueServiceInformationServiceURL: \
MDS2GRIS:ldap://${BDII_HOST}:2170/mds-vo-name=${SITE_NAME},o=grid
GlueServiceStatus: OK
EOF

    $INSTALL_ROOT/lcg/sbin/lcg-info-static-create -c $outfile -t \
$INSTALL_ROOT/lcg/etc/GlueService.template > \
$INSTALL_ROOT/lcg/var/gip/ldif/static-file-dSE.ldif

fi # end of dcache,dpm

if ( echo "${NODE_TYPE_LIST}" | egrep -q 'SE_dpm_(mysql|oracle)' ); then

    # Install dynamic script pointing to gip plugin
    cat << EOF > ${INSTALL_ROOT}/lcg/var/gip/plugin/lcg-info-dynamic-se
#! /bin/sh
${INSTALL_ROOT}/lcg/libexec/lcg-info-dynamic-dpm ${INSTALL_ROOT}/lcg/var/gip/ldif/static-file-SE.ldif
EOF

    chmod +x ${INSTALL_ROOT}/lcg/var/gip/plugin/lcg-info-dynamic-se

fi # end of dpm

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

outfile=$INSTALL_ROOT/lcg/var/gip/lcg-info-static-se.conf
```



```
# dynamic_script points to the script generated by config_info_dynamic_se<se_type>
#   echo "">> $outfile
#   echo "dynamic_script=${INSTALL_ROOT}/lcg/libexec5A/lcg-info-dynamic-se" >> $outfile
#   echo >> $outfile           # Empty line to separate it form published info

cat <<EOF > $outfile
dn: GlueSEUniqueID=${se_host}
GlueSEName: $SITE_NAME:${se_type}
GlueSEPort: 2811
GlueSESizeTotal: 0
GlueSESizeFree: 0
GlueSEArchitecture: multidisk
GlueInformationServiceURL: ldap://`hostname -f`:2135/mds-vo-name=local,o=grid
GlueForeignKey: GlueSiteUniqueID=${SITE_NAME}

dn: GlueSEAccessProtocolLocalID=gsiftp, GlueSEUniqueID=${se_host}
GlueSEAccessProtocolType: gsiftp
GlueSEAccessProtocolEndpoint: gsiftp://${se_host}
GlueSEAccessProtocolCapability: file transfer
GlueSEAccessProtocolVersion: 1.0.0
GlueSEAccessProtocolPort: 2811
GlueSEAccessProtocolSupportedSecurity: GSI
GlueChunkKey: GlueSEUniqueID=${se_host}

dn: GlueSEAccessProtocolLocalID=rftio, GlueSEUniqueID=${se_host}
GlueSEAccessProtocolType: rftio
GlueSEAccessProtocolEndpoint:
GlueSEAccessProtocolCapability:
GlueSEAccessProtocolVersion: 1.0.0
GlueSEAccessProtocolPort: 5001
GlueSEAccessProtocolSupportedSecurity: RFIO
GlueChunkKey: GlueSEUniqueID=${se_host}

dn: GlueSEControlProtocolLocalID=$control_protocol, GlueSEUniqueID=${se_host}
GlueSEControlProtocolType: $control_protocol
GlueSEControlProtocolEndpoint: $control_endpoint
GlueSEControlProtocolCapability:
GlueSEControlProtocolVersion: 1.0.0
GlueChunkKey: GlueSEUniqueID=${se_host}
EOF

for VO in $VOS; do

    if ( echo "${NODE_TYPE_LIST}" | grep SE_dpm > /dev/null ); then
storage_path="/dpm/`hostname -d`/home/${VO}"
storage_root="${VO}:${storage_path}"
        elif ( echo "${NODE_TYPE_LIST}" | grep SE_dcachecache > /dev/null ); then
storage_path="/pnfs/`hostname -d`/data/${VO}"
storage_root="${VO}:${storage_path}"
        else
storage_path=$( eval echo '$VO`echo ${VO} | tr '[:lower:]' '[:upper:]'`_STORAGE_DIR )
storage_root="${VO}:${storage_path}${CLASSIC_STORAGE_DIR}"
    fi
done
```



```
fi

cat <<EOF >> $outfile

dn: GlueSALocalID=$VO,GlueSEUniqueID=${se_host}
GlueSARoot: $storage_root
GlueSAPath: $storage_path
GlueSAPType: permanent
GlueSAPolicyMaxFileSize: 10000
GlueSAPolicyMinFileSize: 1
GlueSAPolicyMaxData: 100
GlueSAPolicyMaxNumFiles: 10
GlueSAPolicyMaxPinDuration: 10
GlueSAPolicyQuota: 0
GlueSAPolicyFileLifeTime: permanent
GlueSAStateAvailableSpace: 1
GlueSAStateUsedSpace: 1
GlueSAAccessControlBaseRule: $VO
GlueChunkKey: GlueSEUniqueID=${se_host}
EOF

done

    $INSTALL_ROOT/lcg/sbin/lcg-info-static-create -c $outfile -t \
$INSTALL_ROOT/lcg/etc/GlueSE.template > \
$INSTALL_ROOT/lcg/var/gip/ldif/static-file-SE.ldif

fi #endif for SE_HOST

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

outfile=$INSTALL_ROOT/lcg/var/gip/lcg-info-static-vobox.conf

    for x in VOS SITE_EMAIL SITE_NAME VOBOX_PORT; do
        if [ "x`eval echo '$x'" = "x" ]; then
            echo "\$x not set"
            return 1
        fi
    done

    for VO in $VOS; do
        dir=${INSTALL_ROOT}/edg/var/info/$VO
        mkdir -p $dir
        f=$dir/$VO.list
        [ -f $f ] || touch $f
        # work out the sgm user for this VO
        sgmuser=`users_getsgmuser $VO`
        sgmgroup=`id -g $sgmuser`
        chown -R ${sgmuser}:${sgmgroup} $dir
        chmod -R go-w $dir
    done

    cat <<EOF > $outfile
```



```
dn: GlueServiceUniqueID=gsissh://${VOBOX_HOST}:${VOBOX_PORT}
GlueServiceName: ${SITE_NAME}-vobox
GlueServiceType: VOBOX
GlueServiceVersion: 1.0.0
GlueServiceEndpoint: gsissh://${VOBOX_HOST}:${VOBOX_PORT}
GlueServiceURI: unset
GlueServiceAccessPointURL: gsissh://${VOBOX_HOST}:${VOBOX_PORT}
GlueServiceStatus: OK
GlueServiceStatusInfo: No Problems
GlueServiceWSDL: unset
GlueServiceSemantics: unset
GlueServiceStartTime: 1970-01-01T00:00:00Z
GlueServiceOwner: LCG
GlueForeignKey: GlueSiteUniqueID=${SITE_NAME}
EOF

    for VO in $VOS; do
        echo "GlueServiceAccessControlRule: $VO" >> $outfile
    done

    echo >> $outfile

    $INSTALL_ROOT/lcg/sbin/lcg-info-static-create -c $outfile -t \
$INSTALL_ROOT/lcg/etc/GlueService.template > \
$INSTALL_ROOT/lcg/var/gip/ldif/static-file-VOBOX.ldif

fi #endif for VOBOX_HOST

cat << EOT > $INSTALL_ROOT/globus/libexec/edg.info
#!/bin/bash
#
# info-globus-ldif.sh
#
#Configures information providers for MDS
#
cat << EOF

dn: Mds-Vo-name=local,o=grid
objectclass: GlobusTop
objectclass: GlobusActiveObject
objectclass: GlobusActiveSearch
type: exec
path: $INSTALL_ROOT/lcg/libexec/
base: lcg-info-wrapper
args:
cachetime: 60
timelimit: 20
sizelimit: 250

EOF

EOT
```



```
chmod a+x $INSTALL_ROOT/globus/libexec/edg.info

if [ ! -d "$INSTALL_ROOT/lcg/libexec" ]; then
    mkdir -p $INSTALL_ROOT/lcg/libexec
fi

cat << EOF > $INSTALL_ROOT/lcg/libexec/lcg-info-wrapper
#!/bin/sh

export LANG=C
$INSTALL_ROOT/lcg/bin/lcg-info-generic $INSTALL_ROOT/lcg/etc/lcg-info-generic.conf

EOF

chmod a+x $INSTALL_ROOT/lcg/libexec/lcg-info-wrapper

cat << EOT > $INSTALL_ROOT/globus/libexec/edg.schemalist
#!/bin/bash

cat <<EOF
${INSTALL_ROOT}/globus/etc/openldap/schema/core.schema
${INSTALL_ROOT}/glue/schema/ldap/Glue-CORE.schema
${INSTALL_ROOT}/glue/schema/ldap/Glue-CE.schema
${INSTALL_ROOT}/glue/schema/ldap/Glue-CESEBind.schema
${INSTALL_ROOT}/glue/schema/ldap/Glue-SE.schema
EOF

EOT

chmod a+x $INSTALL_ROOT/globus/libexec/edg.schemalist

# Configure gin
if ( ! echo "${NODE_TYPE_LIST}" | egrep -q '^UI$|^WN[A-Za-z_]*$' ); then
    if [ ! -d ${INSTALL_ROOT}/glite/var/rgma/.certs ]; then
        mkdir -p ${INSTALL_ROOT}/glite/var/rgma/.certs
    fi

    cp -pf /etc/grid-security/hostcert.pem /etc/grid-security/hostkey.pem \
    ${INSTALL_ROOT}/glite/var/rgma/.certs
    chown rgma:rgma ${INSTALL_ROOT}/glite/var/rgma/.certs/host*

    (
    egrep -v 'sslCertFile|sslKey' \
    ${INSTALL_ROOT}/glite/etc/rgma/ClientAuthentication.props
    echo "sslCertFile=${INSTALL_ROOT}/glite/var/rgma/.certs/hostcert.pem"
    echo "sslKey=${INSTALL_ROOT}/glite/var/rgma/.certs/hostkey.pem"
    ) > /tmp/props.$$
    mv -f /tmp/props.$$ ${INSTALL_ROOT}/glite/etc/rgma/ClientAuthentication.props

    #Turn on Gin for the GIP and maybe FMON
    export RGMA_HOME=${INSTALL_ROOT}/glite
    ${RGMA_HOME}/bin/rgma-gin-config --gip=yes ${FMON}
    /sbin/chkconfig rgma-gin on
    /etc/rc.d/init.d/rgma-gin restart 2>${YAIM_LOG}

```



```
fi
```

```
return 0  
}
```

16.12. 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
```




```
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
}
```

16.13. CONFIG_PROXY_SERVER

```
config_proxy_server (){
INSTALL_ROOT=${INSTALL_ROOT:-/opt}

requires GRID_TRUSTED_BROKERS

if [ -f ${INSTALL_ROOT}/edg/etc/edg-myproxy.conf ]; then
    rm -f ${INSTALL_ROOT}/edg/etc/edg-myproxy.conf
fi

split_quoted_variable $GRID_TRUSTED_BROKERS | while read x; do
    echo "$x" >> ${INSTALL_ROOT}/edg/etc/edg-myproxy.conf
done
}
```



done

```
/sbin/chkconfig --add myproxy
```

```
/etc/init.d/myproxy stop < /dev/null
```

```
/etc/init.d/myproxy start < /dev/null
```

```
}
```