Table of Contents

1. Persistency cluster (lxmrra5001, slc5pf02, slc6pf01)

The Persistency team is maintaining a few development servers that are needed for testing the Persistency Framework software (e.g. a MySQL server and a CORAL server). These services have recently been migrated to a 'persistency' cluster in the Computer Centre.

1.1 Quattor

The following is a list of useful hints for the administration of the 'persistency' cluster through the quattor CDB tools. You may refer to the VOBoxAdminGuide for more details. You may also check the twiki about quattor for IT-ES prepared by several people in IT-ES.

You need ssh access to lxadm (ask lxadm support) and access to the CDB database (ask CDB support). By being logged on lxadm with your account you can, amongst other things:

- use `cdbop`
- use `sms get` and `sms set`
- use `ssh root@<node>`

1.1.1 Using cdbop

On lxadm you may edit profiles using the `cdbop` tool. You need a `.cdbop.conf` file containing the two lines `server=cdbserv` and `use-krb5`. You need a valid AFS token for the user with CDB access privileges (not necessarily the lxadm login).

An introduction to `cdbop` can be found in the VOBoxVOCConf page. Use `get` to download templates to your local filesystem. Use `add` to upload a new template (e.g one that you created by editing a similar one for another cluster or node). Use `update` to upload the changes to an existing template. Do not forget to `commit` your changes after `add` or `update`. You may list all versions of a template using `vls`. You may then retrieve an old version of a template using `vget` to compare it to the current version.

New software packages should be added via `cdbop` on lxadm. You may browse the SLC5 and SLC6 rpms on swrepsrv. On the nodes where the new software is to be installed you should then run `/usr/sbin/ccm-fetch; /usr/sbin/spma_wrapper.sh; /usr/sbin/ncm-ncd --configure -all`. If the installation fails because of conflicts or missing dependencies you may need to sort out some issues manually using `rpm` (see CT700480). The 'stages/prod' version of the configuration and rpms is used for all nodes (see CT661656).

Security updates for the O/S must be regularly picked up by changing the ELFMS_OSDATE variable in the quattor template. See the OsUpdates page for more details. A reasonable choice may be to use the version used on lxplus.

The `/etc/security/limits.conf` settings may be changed through the `"/software/components/interactivelimits/values"` section in the interactivelimits template. Only the values listed in the template are changed (everything else remains untouched and can be manually edited).

Some information from CDB about the persistency cluster may also be found on CDBWeb.

1.2 Monitoring and system administration

A persistency-service egroup has been created and is used as the user contact for the persistency cluster. The archives are available on sharepoint.

A 'Cluster persistency' service has been created in the service database SDB (see SDBUserDoc for more details). Its CDB ID is set to 'persistencycluster' in SDB and the same value is used in the cluster template (`"/system/service/infrastructure" = "persistencycluster"`). The service is of 'infrastructure' type in SDB (otherwise the link between CDB and SDB is not established).

System administration tasks on these nodes are performed by the sysadmin team (see how to prepare machines for administration by the sysadmin team).

The nodes in the cluster are presently in 'standby' default SMS state (this is set in the cluster config template). The 'standby' value ensures that the nodes are monitored by the operators (this is not the case for 'maintenance'), even if they are not yet in production. For more details refer to the CDBMonitoringConfiguration page. All nodes in the cluster currently have 'importance' 10. A value smaller than 50 means that an email is sent during working hours in case of problems (no 24x7 support). See the VOBoxVOCMachineImportance page for more details.

An SMS state different from the default can be temporarely requested using the `sms get` and `sms set` commands on lxadm (see VOBoxVOCConf). Each such request remains enabled (setting the current state to one different from the default) until that request (identifed also by the reason given for the state change) is explicitly cleared. When all state change requests are cleared, the state goes back to the default. All state changes have been cleared and the persistency nodes are all at their default (standby) state.

When the machine is in maintenance, interactive logins are normally not allowed. You may need to login as root and execute `/usr/sbin/spma_wrapper.sh; /usr/sbin/spma_ncm_wrapper.sh; /usr/sbin/sms-set-state` (see VOBoxVOCConf for more details).

1.2.1 Lemon

The persistency cluster nodes are monitored by Lemon. These are the direct links to Lemon monitoring for the nodes in cluster persistency:

- lxmrra5001 (SLC5 client node)
- slc5pf02, alias coralmysql (MySQL server) and coralsrv01/coralprx01 (CORAL server)
- slc6pf01 (SLC6 client node)

These are the direct links to the nodes that used to belong to the cluster and have since been removed:

- lxbrg2601 (replaced by slc5pf02)

1.3 Node reinstallation

To reinstall a node you should open an ITCM ticket with the sysadmin team via the ITCM web interface.

Before asking for a node to be reinstalled, you should make sure that:

- the filesystem partitions for the specific node(s) have been defined
- SinDes, in particular `passwd.header` and `group.header`, has been prepared for the cluster (see SindesNewCluster - ask sindes support for help via SNOW if needed)

You can also (at your own risk) reinstall the nodes yourself, instead of asking the sysadmin team. See the ELFMs InstallationService documentation (to run `PrepareInstall`, you must have asked CDB support to grant your `lxadm` account access as `sindes@sindes-server`):

- ssh (as yourself) on `lxadm`
- execute `/usr/bin/PrepareInstall` (you may use the `--rootpw` option in this step to set the installation root password for your first login; then the root password will be reset to that managed by quattor via sindes)
- reboot the node

1.3.1 Virtual machines

If your node is a virtual machine (e.g. slc6pf01, slc5pf02), you can trigger the installation process by stopping and restarting the VM from vmm.cern.ch (login and go to 'Manage My Virtual Machine").

- The vmm interface also allows you to access the console (the lxadm script `connect2console` does not work for VMs). Note however that this only works with IE8 (but not on XP) and IE9, so you may need to connect to the terminal server cerntsnew.cern.ch. If you get an ""Virtual Machine Manager lost the connection to the virtual machine because another connection was established to this machine." error while trying to connect to the console, note that this may be due to the use of different accounts in quattor and as vm owner (see MS support): in my case (avalassi on quattor) this disappeared when I changed the VM owner in vmm from valassi to avalassi.
- Note that you can also request new virtual machines from the vmm interface (but those obtained this way will not be quattorized - you must do a hardware procurement request via SNOW if you want a quattor managed machine).
- For virtual machines, it is a good idea to ask for the network interface to be changed (from 'Emulated' to 'Synthetic') after the first successful installation. You will lose the ability to boot over PXE (and hence reinstall...), but you will gain in network performance. This is a privileged operation and you must ask for a superuser to do it for you.

To manage virtual machines, you can also use the cern-cvi-console tool on Linux.

1.3.2 Upgrading from SLC5 to SLC6

To upgrade from SLC5 to SLC6, please have a look at the Slc6WithQuattor page. It may also be interesting to look at the old documentation about upgrading from SLC3 to SLC4.

A big difference on SLC6 is in the authentication mechanism, which uses nslcd instead of ldap. You will need to use different quattor templates accordingly. See the Slc6WithQuattor page and the ELFmsZuulSLC6 page referenced therein. To debug any problems you should have, compare the output from `ncm-query --comp authconfig` on an SLC5 and an SLC6 node.

1.3.3 Node reinstallation history

The persistency cluster consists of three nodes:

- lxmrra5001, delivered in July 2010 (see CT657551), was never reinstalled
- slc6pf01, delivered in February 2012 (see RQF0063527), was last reinstalled with SLC6 in February 2012 (INC103011)
- slc5pf02, delivered in October 2012 (see RQF0150590), was last reinstalled with SLC5 in October 2012 (RQF0150590)

Other nodes used to belong to the cluster and have been retired or replaced:

- lxbrg2601, delivered in July 2010 (see CT661656), was last reinstalled in August 2011 (the memory module had to be changed, fixing slow cached reads reported by hdparm, see ITCM:431190)

1.4 Grid certificates

We may need to generate Grid host certificates for these nodes. In a message in August 2010 to VOBox administrators, Gavin suggested that we should request access for SINDES upload permission from sindes support (but we should not request permission unless we're sure we need it as the permissions need to be processed manually). The CERN CA ACLs are already set to allow access for VOCs. The VOC admin guide has been updated with the instructions.

2. MySQL server (slc5pf02) and MySQL server on demand (dbod-coolmyod)

A MySQL server has been deployed on slc5pf02 (previously lxbrg2601), one of the nodes of the persistency cluster, and is currently used for the nightly tests. Another server has been deployed in December 2013 on the IT-DB database-on-demand service (dbod-coolmyod, see RQF0286991). The following steps were taken to install and configure the two MySQL servers from scratch.

2.1 Add the mysql software via quattor (slc5pf02 only)

Install the mysql software by adding the following lines to the relevant quattor template mysqlserver.tpl that is included only in profile_slc5pf02.tpl:

```
# Software configuration: mysql server 5.0.95
"/software/packages" = pkg_add("mysql");
"/software/packages" = pkg_add("mysql-server");
"/software/packages" = pkg_add("perl-DBD-MySQL");
```

The lines above also create a `mysql` local user. If you also want to enable interactive logins for that user, add also the following lines:

```
# Interactive login of local users (SLC5)
"/software/components/useraccess/users/mysql/acls" = list("system-auth");
```

2.2 Configure the server and start it for the first time (slc5pf02 only)

Move the (empty) mysql data directory to /data and create a symlink:

```
mkdir /data/
mv /var/lib/mysql /data/
ln -sf /data/mysql /var/lib/
```

Change the selinux security context for /data to default_t. The previous context was undefined (file_t) and caused mysqld to fail.

```
ls -d --lcontext /data
chcon -t default_t /data
```

The data directory in /etc.my.conf points to /var/lib/mysql. You need to change it to /data/mysql even if the symlink is in place.

```
mv /etc/my.cnf /etc/my.cnf.original
cat /etc/my.cnf.original | sed 's|/var/lib|/data|' > /etc/my.cnf
```

You can now start the MySQL database for the first time. This will create all relevant system databases.

```
/sbin/service mysqld start
```

You can then secure the installation.

```
/usr/bin/mysql_secure_installation
```

I chose the following options:

- I defined a password for the mysql root user
- I removed anonymous users
- I disallowed remote root login
- I removed the test database
- I reloaded the privileges table

2.3 Open the mysql port in iptables via quattor (slc5pf02 only)

The mysql port is closed by default in the firewall and must be opened. See for instance

- http://www.cyberciti.biz/tips/how-do-i-enable-remote-access-to-mysql-database-server.html
- http://www.cyberciti.biz/tips/linux-iptables-18-allow-mysql-server-incoming-request.html

The Linux firewall may be configured using `iptables`. This can be controlled via quattor using the ncm-iptables component. As an example, see the arc iptables configuration.

To list all iptables chains:

```
/sbin/iptables --list
```

Open the mysql ports for all IPs inside the CERN network. Add the following lines to iptables.tpl:

```
# Enable mysql from CERN LAN (see also )
include components/iptables/rules_lan_mysql;
```

2.4 Configure the server runlevels via quattor (slc5pf02 only)

Custom services installed on the cluster can be configured to be started/stopped at different Linux runlevels using chkconfig. This can be controlled via quattor using the ncm-chkconfig component. As an example, see the afs_client chkconfig configuration.

By default the server is off for all runlevels including 345. To check the current runlevels:

```
/sbin/chkconfig  --list mysqld
```

The server was initially configured to be on for runlevels 345 (start/stop automatically on startup/shutdown), by adding the following lines to mysqlserver.tpl:

```
# Service configuration (runlevels)
"/software/components/chkconfig/service/mysqld/on"        = "345";
"/software/components/chkconfig/service/mysqld/add"       = true;
"/software/components/chkconfig/service/mysqld/startstop" = true;
```

Configure the server to be off all the time, including for runlevels 345, by replacing the first line by another line declaring the server to be off.

```
# Service configuration (runlevels)
"/software/components/chkconfig/service/mysqld/off"       = "0123456";
#"/software/components/chkconfig/service/mysqld/on"        = "345";
"/software/components/chkconfig/service/mysqld/add"       = true;
"/software/components/chkconfig/service/mysqld/startstop" = true;
```

The MySQL server on slcpf02 is currently switched off.

Remember to use `/sbin/chkconfig --del mysqld` before relaunching the quattor configuration if you modify these runlevels.

2.5 Configure the server to use the mysql ANSI mode (slc5pf02 and dbod-coolmyod)

COOL tests fail (since many years) if the ANSI mode is not used. You must modify the server configuration to use ANSI mode.

On slc5pf02, modify /etc/my.cnf then restart the database:

```
cat /etc/my.cnf | sed 's|user=mysql|user=mysql\nport=3306\nsql-mode=ansi\n|' > /etc/my.cnf.new
\mv /etc/my.cnf.new /etc/my.cnf
cat /etc/my.cnf | sed 's|user=mysql|user=mysql\n#default-character-set=utf8|' > /etc/my.cnf.new
\mv /etc/my.cnf.new /etc/my.cnf
cat /etc/my.cnf | sed 's|user=mysql|user=mysql\n#character-set-server=utf8|' > /etc/my.cnf.new
\mv /etc/my.cnf.new /etc/my.cnf
 /sbin/service mysqld restart
```

On dbod-coolmyod, use the https://cern.ch/DBOnDemand web interface to download file my.cnf and add the following line at the bottom, then upload the modified my.cnf file. Then shutdown and start up again the database using the Web interface. Do not try to modify also default-character-set and character-set-server (this was attempted but the database would not start up again!).

```
sql-mode = ansi
```

ANSI mode is enabled in your database if the following query returns the following output (as discussed in the MySQL manual).

```
mysql> SELECT @@global.sql_mode;
+-------------------------------------------------------------+
| @@global.sql_mode                                           |
+-------------------------------------------------------------+
| REAL_AS_FLOAT,PIPES_AS_CONCAT,ANSI_QUOTES,IGNORE_SPACE,ANSI |
+-------------------------------------------------------------+
```

2.6 Create users and databases (slc5pf02 and dbod-coolmyod)

On slc5pf02, connect to the database as root

```
/usr/bin/mysql -pxxx
```

On dbod-coolmyod, connect to the database as admin:

```
/usr/bin/mysql -uadmin -hdbod-coolmyod -P5500 -pxxx
```

Create all databases

```
create database LCG_COOL;
create database LCG_COOL_NIGHT;
create database LCG_CORAL_NIGHT;
create database LCG_POOL_NIGHT;
create database AVALASSI;
create database AALVAREZ;
```

Create all users

```
GRANT ALL ON LCG_COOL_NIGHT.* TO 'LCG_COOL_NIGHT'@'%' identified by 'xxx';
GRANT ALL ON LCG_COOL.* TO 'LCG_COOL'@'%' identified by 'xxx';
GRANT ALL ON LCG_CORAL_NIGHT.* TO 'LCG_CORAL_NIGHT'@'%' identified by 'xxx';
GRANT ALL ON LCG_POOL_NIGHT.* TO 'LCG_POOL_NIGHT'@'%' identified by 'xxx';
GRANT ALL ON AVALASSI.* TO 'AVALASSI'@'%' identified by 'xxx';
GRANT ALL ON AALVAREZ.* TO 'AALVAREZ'@'%' identified by 'xxx';
```

Extra grants

```
GRANT SELECT ON LCG_COOL_NIGHT.* TO 'AVALASSI'@'%';
GRANT SELECT ON LCG_COOL.* TO 'LCG_COOL_NIGHT'@'%';
```

Flush privileges

```
flush privileges;
```

2.7 Configure DNS aliases and XML files for nightly tests

The XML files that are used for the nightly tests are those installed on AFS (for Linux, and copied to local private directories for Windows and Mac):

```
ls /afs/cern.ch/sw/lcg/app/pool/db/authentication.xml
ls /afs/cern.ch/sw/lcg/app/pool/db/dblookup.xml
```

These XML files were initially configured to execute MySQL tests (`mysql://coralmysql.cern.ch/...`) against the server referenced by the `coralmysql` network alias. This alias currently points to slc5pf02, as can be checked in the network database. As of December 2013, MySQL tests will be performed against `mysql://dbod-coolmyod.cern.ch:5500/...` instead. **The MySQL server on slcpf02 is currently switched off**.

3. CORAL server (slc5pf02)

Two CORAL server instances (a production and a development version) have been deployed on slc5pf02 (previously lxbrg2601), one of the nodes of the persistency cluster, and are currently used for the nightly tests. The following steps were taken to install and configure these servers from scratch.

3.1 Add a coralsrv user via quattor

Create a custom local user `coralsrv` to run the CORAL servers (just like the `mysql` user runs the MySQL server). This can be controlled via quattor using the ncm-accounts component. As an example, see the gridpx cluster configuration.

To enable `ncm-accounts`, create the custom `coralsrv` user and enable interactive logins for this user, add the following lines to useraccess.tpl:

```
# Local users and groups
include components/accounts/config;
"/system/accounts/local" = push('coralsrv');
"/software/components/accounts/users/coralsrv/comment" = 'coralServer user';
"/software/components/accounts/users/coralsrv/createHome" = true;
"/software/components/accounts/users/coralsrv/groups" = list('coralsrv');
"/software/components/accounts/users/coralsrv/homeDir" = '/home/coralsrv';
"/software/components/accounts/users/coralsrv/shell" = "/bin/bash";
"/software/components/accounts/users/coralsrv/uid" = 201;
"/software/components/accounts/groups/coralsrv/comment" = 'coralServer group';
"/software/components/accounts/groups/coralsrv/gid" = 201;
# Interactive login of local users (SLC5)
"/software/components/useraccess/users/coralsrv/acls" = list("system-auth");
```

3.2 Grant write privileges to CORAL developers on /home/coralsrv

As `root`, grant write privileges to Andrea and Raffaello on /home/coralsrv so that they can install and build the software with their own (AFS) accounts. Grant the same privileges also to the coralsrv user so that it is not locked out in its own account!

```
setfacl -b -R -m u:coralsrv:rwx -m u:avalassi:rwx -m u:aalvarez:rwx /home/coralsrv/
setfacl -R -dm u:coralsrv:rwx -dm u:avalassi:rwx -dm u:aalvarez:rwx /home/coralsrv/
mkdir /home/coralsrv/CORAL
```

3.3 Install the CORAL software

As `coralsrv` or using your own AFS account, check out the CORAL software.

```
mkdir /home/coralsrv/CORAL/CORAL_2_3-patches
cd /home/coralsrv/CORAL/CORAL_2_3-patches
svn co svn+ssh://svn.cern.ch/reps/lcgcoral/coral/tags/CORAL_2_3-patches src
mkdir /home/coralsrv/CORAL/CORAL-preview
cd /home/coralsrv/CORAL/CORAL-preview
svn co svn+ssh://svn.cern.ch/reps/lcgcoral/coral/tags/CORAL-preview src
```

Build the setup and cleanup scripts:

```
setenv CMTCONFIG x86_64-slc5-gcc46-opt
cd /home/coralsrv/CORAL/CORAL_2_3-patches/src/config/cmt
source CMT_env.csh
```

```
cmt config
cd /home/coralsrv/CORAL/CORAL-preview/src/config/cmt
source CMT_env.csh
cmt config
```

Create symbolic links to `23x` and `24x` for the two installations:

```
cd /home/coralsrv/CORAL
\rm -f 23x 24x
ln -sf CORAL_2_3-patches 23x
ln -sf CORAL-preview 24x
```

Install the `/etc/init.d` scripts for `coralserver23` and `coralserverdev24` using symbolic links:

```
\rm -f /etc/init.d/coralserver23
\rm -f /etc/init.d/coralserver24
ln -sf /home/coralsrv/CORAL/23x/src/CORAL_SERVER/CoralServer/scripts/coralserver /etc/init.d/co
ln -sf /home/coralsrv/CORAL/24x/src/CORAL_SERVER/CoralServer/scripts/coralserver /etc/init.d/co
```

3.4 Update and build (or rebuild) the CORAL software

Update and build/rebuild the CORAL_2_3-patches software:

```
cd /home/coralsrv/CORAL/CORAL_2_3-patches/src
svn update
date > __CORAL_2_3-patches.date__
cd /home/coralsrv/CORAL/CORAL_2_3-patches/src/config/cmt
setenv CMTCONFIG x86_64-slc5-gcc46-opt
source CMT_env.csh
cmt br cmt make all_groups
```

Update and build/rebuild the CORAL-preview software:

```
cd /home/coralsrv/CORAL/CORAL-preview/src
svn update
date > __CORAL-preview.date__
cd /home/coralsrv/CORAL/CORAL-preview/src/config/cmt
setenv CMTCONFIG x86_64-slc5-gcc46-opt
source CMT_env.csh
cmt br cmt make all_groups
```

3.5 Open the coralserver ports in iptables via quattor

Open the coralserver ports (40007 for for `coralserver24`, 40009 for `coralserver23`) for all IPs inside the CERN network. Add the following lines to iptables.tpl:

```
"/software/components/iptables/filter/rules" = push(nlist(
    "command", "-A", "chain", "INPUT", "source", "137.138.0.0/16",
    "protocol", "tcp", "dst_port", "40007", "target", "ACCEPT"));
"/software/components/iptables/filter/rules" = push(nlist(
    "command", "-A", "chain", "INPUT", "source", "128.141.0.0/16",
    "protocol", "tcp", "dst_port", "40007", "target", "ACCEPT"));
"/software/components/iptables/filter/rules" = push(nlist(
    "command", "-A", "chain", "INPUT", "source", "128.142.0.0/16",
    "protocol", "tcp", "dst_port", "40007", "target", "ACCEPT"));
"/software/components/iptables/filter/rules" = push(nlist(
    "command", "-A", "chain", "INPUT", "source", "137.138.0.0/16",
    "protocol", "tcp", "dst_port", "40009", "target", "ACCEPT"));
"/software/components/iptables/filter/rules" = push(nlist(
    "command", "-A", "chain", "INPUT", "source", "128.141.0.0/16",
    "protocol", "tcp", "dst_port", "40009", "target", "ACCEPT"));
```

```
"/software/components/iptables/filter/rules" = push(nlist(
    "command", "-A", "chain", "INPUT", "source", "128.142.0.0/16",
    "protocol", "tcp", "dst_port", "40009", "target", "ACCEPT"));
```

3.6 Configure the server runlevels via quattor

Configure the servers (23x and 24x) to be on for runlevels 345 (start/stop automatically on startup/shutdown). By default the servers do not exist (they will only exist once the software is installed and the quattor configuration is rerun). To check the current runlevels:

```
/sbin/chkconfig  --list coralserver23
/sbin/chkconfig  --list coralserver24
```

Add the following lines to the relevant quattor template coralserver.tpl that is included only in profile_slc5pf02.tpl:

```
# Service configuration (runlevels)
"/software/components/chkconfig/service/coralserver23/on"        = "345";
"/software/components/chkconfig/service/coralserver23/add"       = true;
"/software/components/chkconfig/service/coralserver23/startstop" = true;
"/software/components/chkconfig/service/coralserver24/on"        = "345";
"/software/components/chkconfig/service/coralserver24/add"       = true;
"/software/components/chkconfig/service/coralserver24/startstop" = true;
```

Note that this will install (using symlinks) the `/etc/init.d/coralserver23` and `/etc/init.d/coralserver24` scripts (e.g. in `/etc/rc3.d/S94coralserver24`), so you should not hardcode an expected `/etc/init.d/coralserver23` name into the script itself (see this bug fix). If the services do not seem to start/stop automatically on boot/shutdown, it will be difficult to use boot logs to identify the issue (see this RedHat bug); instead, execute `/etc/rc.d/rc` manually to start/stop the relevant services for the current runlevel(you may check the current runlevel using `who -r` and change it using `telinit` for tests).

Note also that the name of the symlinked script (e.g. `/etc/rc3.d/S94coralserver24`) depends on the runlevel priority that is specified inside the `chkconfig` section at the top of the script itself (`# chkconfig: -94 06`, in this case). If you want to modify these priorities, do not forget to erase the existing `rcN.d` scripts (e.g. `/sbin/chkconfig --del coralserverdev`) before rerunning the quattor configuration commands.

3.7 Configure interactive limits for the coralsrv user via quattor

There are some (yet unconfirmed) indications that the 24x version of the CORAL server executable needs unlimited virtual memory (see bug #86734). This can be controlled via quattor using the ncm-interactivelimits component.

Configure the `coralsrv` user to have unlimited virtual memory (the server crashed with 2GB virtual memory). Add the following lines to the interactivelimits.tpl template. This will modify the interactive limits in `/etc/seccurity/limits.conf` (

```
"/software/components/interactivelimits/active" = true;
"/software/components/interactivelimits/values" = list (
    list( "*", "soft", "as", "2048000" ), # Soft limit 2GB
    list( "*", "hard", "as", "4096000" ), # Hard limit 4GB
    list( "coralsrv", "soft", "as", "unlimited" ), # Soft limit unlimited
    list( "coralsrv", "hard", "as", "unlimited" ), # Hard limit unlimited
);
```

3.8 Configure DNS aliases and XML files for nightly tests

The XML files that are used for the nightly tests have been configured to execute CORAL server tests (`coral://coralsrv01.cern.ch/...`) against the server referenced by the `coralsrv01` network alias. This alias currently points to slc5pf02, as can be checked in the network database.

These XML files need to be copied locally from AFS. As `coralsrv`, execute:

```
\cp /afs/cern.ch/sw/lcg/app/pool/db/authentication.xml /home/coralsrv
\cp /afs/cern.ch/sw/lcg/app/pool/db/dblookup.xml /home/coralsrv
```

3.9 Configure TNS_ADMIN for nightly tests

The CORAL server scrpts were recently modified to take TNS_ADMIN from local directories, as a workaround for some Kerberos-related issues leading to ORA-12687 errors (bug #103532). These directories must be modified from the original versions on AFS:

```
cd /home/coralsrv/CORAL/24x/src/config/cmt
setenv CMTCONFIG x86_64-slc5-gcc46-opt
source CMT_env.csh
source setup.csh
cd /home/coralsrv/CORAL/24x
\rm -rf admin
\cp -dpr $TNS_ADMIN/ admin
\mv admin/sqlnet.ora admin/sqlnet.ora.OLD
cat admin/sqlnet.ora.OLD | sed 's/SQLNET.KERBEROS5_CONF_MIT/#SQLNET.KERBEROS5_CONF_MIT/' | sed
cd /home/coralsrv/CORAL/23x/src/config/cmt
setenv CMTCONFIG x86_64-slc5-gcc46-opt
source CMT_env.csh
source setup.csh
cd /home/coralsrv/CORAL/23x
\rm -rf admin
\cp -dpr $TNS_ADMIN/ admin
\mv admin/sqlnet.ora admin/sqlnet.ora.OLD
cat admin/sqlnet.ora.OLD | sed 's/SQLNET.KERBEROS5_CONF_MIT/#SQLNET.KERBEROS5_CONF_MIT/' | sed
```

3.10 Notes about CoralServerProxy

The nightly tests are currently not executed against a centrally maintained CoralServerProxy. One of the reasons for this is that currently the CoralServerProxy can only be reset by being shut down and restarted. Such tests are instead performed as part of the release validation process described in PersistencyReleaseProcess.

A `coralprx01` network alias, in any case, has been added to the XML files that are used for the nightly tests, for possible future tests against a CORAL server proxy (`coral://coralprx01.cern.ch/...`). This alias currently also points to slc5pf02, as can be checked in the network database.

-- AndreaValassi - 19-Oct-2012

This topic: Persistency > OLDPersistencyQuattor
Topic revision: r1 - 2014-10-22 - AndreaValassi

Copyright &© 2008-2019 by the contributing authors. All material on this collaboration platform is the property of the contributing authors.
Ideas, requests, problems regarding TWiki? Send feedback