



Migrating HP Serviceguard for Linux MySQL and PostgreSQL Toolkits to Red Hat Cluster Suite for Red Hat Enterprise Linux 5

September 2009

Executive Summary.....	3
Introduction	3
Audience.....	5
The SGLX MySQL Toolkit.....	6
Gathering Serviceguard Parameters.....	7
Gathering Serviceguard Cluster parameters	7
Gathering Serviceguard MySQL Toolkit Parameters.....	8
Saving Serviceguard MySQL Toolkit Parameters.....	12
The SGLX PostgreSQL Toolkit	12
Gathering Serviceguard Parameters.....	15
Gathering Serviceguard Cluster parameters	15
Gathering Serviceguard PostgreSQL Toolkit Parameters	16
Saving Serviceguard PostgreSQL Toolkit Parameters	19
Migrating Common SGLX Package Parameters to RAs	21
Volume groups	21
SCSI-3 PR fencing	21
Other fencing methods	22
File systems	22
IPAddresses	24
MySQL Resource Agent Migration Procedure.....	26
MySQL Toolkit Migration Planning.....	26
Migration Phase	30
Setting up the Cluster, Quorum Rule, Fencing, Failover Domain, and Resources.....	30
Setting up the RHCS MySQL Service	30
Starting the RHCS MySQL Service.....	33
PostgreSQL Resource Agent Migration Procedure.....	35
PostgreSQL Toolkit Migration Planning	35
Migration phase.....	39
Setting up the Cluster, Quorum Rule, Fencing, Failover Domain, and Resources.....	39
Setting up the RHCS PostgreSQL Service.....	40
Editing the new generated configuration file.....	42
Starting the RHCS PostgreSQL Service	42
Terms.....	45

For More Information	47
Appendix	48
Serviceguard Package Configuration worksheet with MySQL and PostgreSQL Toolkit parameters.....	48

Executive Summary

This white paper is one of a series that describes a procedure to migrate an HP Serviceguard for Linux (SGLX) cluster to a Red Hat Cluster Suite (RHCS) cluster. The white paper "Migrating an HP Serviceguard for Linux Cluster to Red Hat Cluster Suite for Red Hat Enterprise Linux 5", available at <http://docs.hp.com->High Availability -> Serviceguard for Linux>, describes the basics of transitioning an SGLX cluster and a user written toolkit. This whitepaper covers transitioning from an SGLX MySQL Toolkit or an SGLX PostgreSQL Toolkit to an RHCS MySQL resource or an RHCS PostgreSQL 8 resource respectively.

It is assumed that there is an existing SGLX cluster, running at least one instance of a MySQL package or a PostgreSQL Package. The conversion can be done in 3 steps: gathering the information for the conversion process (for both the cluster and the toolkit), completing the basic cluster migration based on the earlier white paper, and finally, completing the MySQL or the PostgreSQL Toolkit migration. A step-by-step process describes how to use the configuration information from a MySQL or a PostgreSQL Toolkit package, on an existing SGLX cluster, to quickly create an RHCS service (a collection of RHCS resources) with similar functionality.

Introduction

The white paper "Migrating an HP Serviceguard for Linux Cluster to Red Hat Cluster Suite for Red Hat Enterprise Linux 5", henceforth referred to as the Cluster Migration white paper, describes the process for taking an existing Serviceguard for Linux (SGLX) cluster and, using information extracted from that cluster configuration, creating an RHCS cluster with characteristics that are familiar to an HP SGLX user. That white paper also describes the process to take information related to a user created Serviceguard package and create a similar RHCS service.

A toolkit is really a package template for a specific application. This white paper leverages the process described in the Cluster Migration white paper to extract information used by an SGLX MySQL Toolkit or an SGLX PostgreSQL Toolkit. It uses that information to create a series of resources, including a MySQL resource or a PostgreSQL resource, to create the RHCS MySQL or PostgreSQL service.

This process is described for both the Legacy and Modular type of packages created using an HP SGLX MySQL or PostgreSQL Toolkit. The specifics related to each toolkit are in different sections for ease of reading.

The Serviceguard Package Configuration worksheet, first introduced in the Cluster Migration white paper, is used in this process. A blank copy of the worksheet, with added sections for MySQL and PostgreSQL Toolkit parameters, is included in the Appendix section of this white paper.

The SGLX MySQL Toolkit has the following features that are not part of the RHCS MySQL RA:

- In the SGLX MySQL toolkit, the `MAINTENANCE_FLAG` parameter is used to bring the SGLX MySQL Toolkit into the maintenance mode. Setting this parameter to "yes", enables the maintenance feature in the toolkit. The MySQL Toolkit will look for a file `mysql.debug` in the package directory. If the file exists and if the maintenance feature is enabled, then monitoring is paused. The MySQL server instance may be brought down for maintenance and package will not be failed over to the adoptive node.

There is no similar feature in the RHCS MySQL RA.

- In the SGLX MySQL Toolkit, the `MONITOR_INTERVAL` parameter is used to define the time interval, in seconds, between the successive monitoring attempts. The monitoring script will wait for `MONITOR_INTERVAL` seconds, between checks to make sure that the MySQL server is running. The default value of this parameter is 5 seconds.

In the RHCS MySQL RA, the environment variable `OCF_CHECK_LEVEL` is used to define the monitoring time interval. For the different `OCF_CHECK_LEVEL` values of 0 or 10, the corresponding monitoring interval is 1 minute or 5 minutes.

- In the SGLX MySQL Toolkit, while monitoring the MySQL server, if the PID file is not found, the script recovers the PID value from the previous monitoring run, if it exists, and stores it in a temporary PID file, followed by a package stop and failover. The temporary PID file, `mysqltmp.pid`, is created in the toolkit directory in case of Modular type of packages, and in the package directory in case of Legacy type of packages. When the script stops the package (the MySQL server), if the PID file does not exist, the stop routine tries to find the PID of the MySQL server process from the temporary PID file.

In the RHCS MySQL RA, there is no similar feature that recovers the PID from the previous monitoring run.

There are a number of other differences between the SGLX MySQL Toolkit and the RHCS MySQL RA. These differences are relatively minor and are described in "[The SGLX MySQL Toolkit](#)" section.

The Serviceguard PostgreSQL Toolkit has the following features that are not part of the RHCS PostgreSQL RA, henceforth referred to as the RHCS PostgreSQL RA (PostgreSQL8 refers to the current release of PostgreSQL).

- In the SGLX PostgreSQL toolkit, the `MAINTENANCE_FLAG` parameter is used to bring the SGLX PostgreSQL Toolkit into the maintenance mode. Setting this parameter to "yes", enables the maintenance feature in the toolkit. The PostgreSQL Toolkit will look for a file `pgsql.debug` in the package directory. If the file exists and if the maintenance feature is enabled, then monitoring is paused. The PostgreSQL database server instance may be brought down for maintenance and the package will not be failed over to the adoptive node.

There is no similar feature in the RHCS PostgreSQL RA.

- In the SGLX PostgreSQL Toolkit, the `MONITOR_INTERVAL` parameter is used to define the time interval, in seconds, between the successive monitoring attempts. The monitoring script will wait for `MONITOR_INTERVAL` seconds, between checks to make sure that the PostgreSQL server is running. The default value of this parameter is 5 seconds.

In the RHCS PostgreSQL RA, the environment variable `OCF_CHECK_LEVEL` is used to define the monitoring time interval. For the different `OCF_CHECK_LEVEL` values of 0 or 10, the corresponding monitoring interval is 1 minute or 5 minutes.

- In the SGLX PostgreSQL Toolkit, while starting the PostgreSQL server, the toolkit script waits for `STARTUP_RETRIES` seconds for the PID file to be created, after starting the PostgreSQL server. Though `STARTUP_RETRIES` is configurable by editing the `hapgsq1.sh` script, it is set to 10 (seconds) by default.

In the RHCS PostgreSQL RA, the PID file is created by the startup script before the PostgreSQL server start. However, if the PID file is not updated with the PostgreSQL PID, the monitoring of the PostgreSQL server fails immediately.

The RHCS PostgreSQL RA has the following feature that does not exist in the SGLX PostgreSQL Toolkit.

- In the SGLX PostgreSQL Toolkit, the toolkit does not wait for the completion of a normal shutdown of the PostgreSQL database server before initiating a forceful halt of the application. While stopping the PostgreSQL server, the script kills the PostgreSQL server process by sending the `SIGQUIT` signal to the process. If the kill fails, the script sends the `SIGKILL` signal to the process.

The RHCS PostgreSQL RA, while stopping the PostgreSQL server, kills the process by sending `SIGTERM` to the process. It then waits for `shutdown_wait` seconds for the process to die. If the process still continues to run, the RA logs an error and exits (and there is no failover). Though the default value of `shutdown_wait` is 0 seconds, it is suggested to set it to more than 0 seconds. In our RHCS PostgreSQL resource setup, the value of `shutdown_wait` is 5 seconds.

There are a number of other differences between the SGLX PostgreSQL Toolkit and the RHCS PostgreSQL RA. These differences are relatively minor and are described in "[The SGLX PostgreSQL Toolkit](#)" section.

Red Hat Cluster Suite delivered with Red Hat Enterprise Linux Advanced Platform 5.3 is used as the basis for this white paper. The information in this white paper is not expected to change significantly for later versions of Red Hat 5.

Audience

This document is targeted for users of HP SGLX MySQL and HP SGLX PostgreSQL Toolkits on RHEL5 who wish to migrate to Red Hat Cluster Suite on RHEL5.

It is assumed that the reader has an understanding of HP SGLX, HP SGLX MySQL Toolkit or HP SGLX PostgreSQL Toolkit, and RHCS. It is assumed that the reader has read "Migrating an HP Serviceguard for Linux Cluster to Red Hat Cluster Suite for Red Hat Enterprise Linux 5". Details in that white paper and the Red Hat documentation are not repeated here. For more information on each solution, see <http://www.hp.com/go/sglx>, <http://docs.hp.com> ->'High Availability' and <http://www.redhat.com/docs/manuals/csgfs>.

The SGLX MySQL Toolkit

The Serviceguard for Linux MySQL Toolkit A.04.01 is a separate set of shell scripts that allows users to configure a Serviceguard package to monitor a MySQL database server. The Toolkit consists of an 'rpm' file which, after installation, and configuration, enables an SGLX package to provide high availability for a MySQL database server. It works as a subsystem under the SGLX core software to provide functions for starting, stopping, and monitoring the MySQL server in a cluster environment. The sample SGLX MySQL Toolkit scripts that are included in this white paper are from the MySQL Toolkit version A.04.01.

The "Introduction" section in this document has the description of a few key differences between the SGLX MySQL toolkit and the implementation of the RHCS MySQL RA. The following features are different between the SGLX MySQL Toolkit and the RHCS MySQL RA. These differences are expected to have little or no impact in most environments.

- In the SGLX MySQL Toolkit, the toolkit script after starting the MySQL server waits for `STARTUP_RETRIES` seconds for the PID file to be created. The `STARTUP_RETRIES` variable is set by editing the `hamysql.sh` script. It is set to 10 by default.

In the RHCS MySQL RA, the script waits for `MYSQL_timeout` seconds for the PID file to be created, after starting the MySQL server. The `MYSQL_timeout` variable is set by editing the `mysql.sh` script. It is set to 30 by default.

- In the SGLX MySQL Toolkit, `PID-FILE` parameter holds the PID file location of the MySQL Server process. The user can customize the PID file location.

In the RHCS MySQL RA, the PID file location cannot be specified by the user, but the RA generates it. The PID file is created under the `/var/run/cluster/<MySQL RA scriptname>` directory. The filename is generated with a combination of the MySQL RA script name and the resource instance name. In our RHCS MySQL resource setup, the PID file was created at `/var/run/cluster/mysql/mysql:MySQLdb.pid`

- In the SGLX MySQL Toolkit, the `CONFIGURATION_FILE_PATH` and the `DATA_DIRECTORY` parameters are used to specify the MySQL server configuration file (in the configuration directory/filename format) and the MySQL server data directory respectively. The user can specify either of them. If the `CONFIGURATION_FILE_PATH` is not specified, the configuration file present in the `DATA_DIRECTORY` is used with the assumption that the name of the configuration file is `my.cnf`. If both the values are specified then `CONFIGURATION_FILE_PATH` value is used and `DATA_DIRECTORY` value is ignored.

In the RHCS MySQL RA, only the configuration file path is defined by the user.

- In the SGLX MySQL Toolkit, the `RETRY_TIMES` parameter is used to specify the number of times the MySQL server status is verified in case of monitoring failure. While monitoring the MySQL server, if the server is down, the toolkit scripts will retry checking the MySQL server status for `RETRY_TIMES` before returning an error. The delay between the successive retries is 5 seconds, though it can be customized by editing the `hamysql.mon` script. The recommended and default value of the parameter `RETRY_TIMES` is 0.

In the RHCS MySQL RA, if the MySQL server is down, the RA does not recheck the server status and returns an error immediately.

- In the SGLX MySQL Toolkit, the `TIME_OUT` parameter is used to define the time period in seconds that this toolkit waits for a completion of a normal shutdown of the MySQL server before initiating a forceful halt of the MySQL server. The default value of this parameter is 30 seconds but users can specify a different value. The toolkit script sends a `SIGTERM` signal to the MySQL server process. It then waits for `TIME_OUT` seconds. If the MySQL server process continues to exist, the toolkit script sends `SIGKILL` to the process.

In the RHCS MySQL RA, the `shutdown_wait` parameter is used to specify the shutdown wait seconds. While stopping the MySQL resource, the script will wait for `shutdown_wait` seconds for the MySQL server process to die after sending the `SIGTERM` signal to the process. The default value of this parameter is 0 seconds. However, if the process continues to exist after `shutdown_wait` seconds, the RA logs error and exits (and there is no failover).

- The SGLX MySQL Toolkit gives a warning if SELinux security policies are enforced in the system. The documentation provides steps for configuring MySQL in a SELinux environment.

There is no similar feature in the RHCS MySQL RA.

- The `--bind-address` option of the MySQL server binds it to the specific IP address. The SGLX MySQL Toolkit invokes the MySQL binary without the `--bind-address` option. The MySQL server reads the IP address to bind to from the MySQL server configuration file. The user edits the MySQL server process configuration file to include the IP address information.

In the RHCS MySQL RA, the `listen_address` parameter is used to specify the IP address to bind the MySQL server to. This parameter can be left empty. If left empty, the RA finds out the IP addresses available in the MySQL RHCS service and uses the first found IP address. The RA invokes the MySQL server with the `--bind-address` option and passes the IP address as a startup parameter to the MySQL server.

Before migrating to RHCS, it is recommended to backup the most recent copy of the application data. This is to protect against data loss in the event of an operator error during the migration. Use any RHEL5 supported backup software to backup the data.

Gathering Serviceguard Parameters

The Serviceguard cluster parameters need to be saved. They are translated to the equivalent RHCS parameters so that users can have similar functionality. The most important step in the migration is saving the Serviceguard parameters in the Serviceguard Package Configuration worksheet. Part of this process must be completed while the SGLX cluster is still available. The cluster does not need to be up during this part of the process but certain cluster commands do need to be able to run.

Gathering Serviceguard Cluster parameters

The first step is to gather the cluster information, including the names of the packages that are configured in the cluster. Run the following command on any one of the Serviceguard cluster nodes, to collect this information:

```
#cmviewcl -v > cluster.info
```

Following is an example output of this command, where a MySQL Toolkit package is configured. The package "mysqlmod" is a MySQL Toolkit package:

<i>CLUSTER</i>	<i>STATUS</i>				
mysqlcl	up				
<i>NODE</i>	<i>STATUS</i>	<i>STATE</i>			
drama	up	running			
<i>Quorum_Server_Status:</i>					
<i>NAME</i>	<i>STATUS</i>	<i>STATE</i>			
linuxlp64.cup.hp.com	up	running			
<i>Network_Parameters:</i>					
<i>INTERFACE</i>	<i>STATUS</i>	<i>NAME</i>			
PRIMARY	up	eth2			
<i>NODE</i>	<i>STATUS</i>	<i>STATE</i>			
poetry	up	running			
<i>Quorum_Server_Status:</i>					
<i>NAME</i>	<i>STATUS</i>	<i>STATE</i>			
linuxlp64.cup.hp.com	up	running			
<i>Network_Parameters:</i>					
<i>INTERFACE</i>	<i>STATUS</i>	<i>NAME</i>			
PRIMARY	up	eth2			
<i>PACKAGE</i>	<i>STATUS</i>	<i>STATE</i>	<i>AUTO_RUN</i>	<i>NODE</i>	
mysqlmod	up	running	enabled	poetry	
<i>Policy_Parameters:</i>					
<i>POLICY_NAME</i>	<i>CONFIGURED_VALUE</i>				
Failover	configured_node				
Failback	manual				
<i>Script_Parameters:</i>					
<i>ITEM</i>	<i>STATUS</i>	<i>MAX_RESTARTS</i>	<i>RESTARTS</i>	<i>NAME</i>	
Service	up	0	0	mysql_service	
<i>Node_Switching_Parameters:</i>					
<i>NODE_TYPE</i>	<i>STATUS</i>	<i>SWITCHING</i>	<i>NAME</i>		
Primary	up	enabled	drama		
Alternate	up	enabled	poetry (current)		

Gathering Serviceguard MySQL Toolkit Parameters

The next step is gathering the SGLX MySQL Toolkit parameters.

The toolkit configuration information can be obtained using any one of the following methods:

- ✓ Toolkit configuration file (hamysql.conf), residing in the toolkit configuration directory or the package directory
- ✓ Package configuration file (Only for Modular packages)
- ✓ Executing the cmgetpkgenv <pkgname> command (Only for Modular packages)

Toolkit Configuration file (Modular and Legacy packages)

File location in the SGLX MySQL Toolkit Legacy Packages:

In the case of the MySQL Toolkit Legacy packages, the toolkit-specific configuration information can be obtained from the file `hamysql.conf` residing in the package directory. The file, `hamysql.conf`, is in the same location as that of the Toolkit Interface Script (`toolkit.sh`). In the package control script, the user specifies this location as a value for the `SERVICE_CMD` attribute and also in the `customer_defined_run_cmds` and `customer_defined_halt_cmds` subroutines.

Following is an example of the sections in the package control script where the user defines the package directory. In case of Legacy packages, in this example, the `hamysql.conf` is in `/usr/local/cmcluster/mysqltoolkitleg` directory.

```
SERVICE_NAME[0]="mysqld"
SERVICE_CMD[0]="/usr/local/cmcluster/mysqltoolkitleg/toolkit.sh monitor"
SERVICE_RESTART[0]=""

# Setting the log file
log_file=${SG_SCRIPT_LOG_FILE:-$0.log}

# START OF CUSTOMER DEFINED FUNCTIONS

# This function is a place holder for customer define functions.
# You should define all actions you want to happen here, before the service is
# started. You can create as many functions as you need.
#
function customer_defined_run_cmds
{
# ADD customer defined run commands.
: # do nothing instruction, because a function must contain some command.
/usr/local/cmcluster/mysqltoolkitleg/toolkit.sh start
test_return 51
}

# This function is a place holder for customer define functions.
# You should define all actions you want to happen here, after the service is
# halted.
#
function customer_defined_halt_cmds
{
# ADD customer defined halt commands.
: # do nothing instruction, because a function must contain some command.
/usr/local/cmcluster/mysqltoolkitleg/toolkit.sh stop
test_return 52
}
# END OF CUSTOMER DEFINED FUNCTIONS
```

File location in the SGLX MySQL Toolkit Modular Packages:

In case of the SGLX MySQL Toolkit Modular packages, the toolkit configuration file (`hamysql.conf`) is generated in the toolkit configuration directory (`TKIT_DIR`) defined in the package configuration file. The toolkit configuration file has all the required MySQL Toolkit parameters.

Following is an example of how the toolkit configuration directory is defined in the package configuration file in the SGLX MySQL Toolkit package configuration file.

```

# *****LINUX MYSQL TOOLKIT PARAMETERS*****
#
# Define Toolkit configuration directory
#
# This directory holds the toolkit configuration file. This parameter directs
# cmapplyconf to generate the hamysql.conf file under this directory. To put
# toolkit into maintenance mode, create pgsq.debug file under this directory.
# This is synonymous with the package directory acronym.
#
# Legal values for TKIT_DIR: <Any String>
TKIT_DIR                /usr/local/cmcluster/mysqltoolkitmod/

```

The hamysql.conf file contains the same MySQL Toolkit parameters for both the Legacy and Modular type of packages. Following is an example of the toolkit configuration file (hamysql.conf):

```

CONFIGURATION_FILE_PATH=/home/mysql/mysqlhome/my.cnf
DATA_DIRECTORY=/home/mysql/mysqlhome
PID_FILE="/usr/local/cmcluster/pkg/mysqlleg/mysqld.pid"
MAINTENANCE_FLAG=yes
RETRY_TIMES=0
MONITOR_INTERVAL=5
TIME_OUT=30

```

Package configuration file (Only for Modular packages)

In case of the SGLX MySQL Toolkit Modular packages, the package configuration file also has the MySQL Toolkit parameters. To view the current package configuration file, run the following command:

```
# cmgetconf -p <package_name> > pkg.info
```

For this example:

```
# cmgetconf -p mysqlmod > mysqlmod.info
```

Following is an example of the SGLX MySQL Toolkit Modular package configuration file. Only the sections that have MySQL Toolkit parameters are included here for ease of reading. The other sections in the MySQL Toolkit Modular package configuration file that have the common SGLX parameters are not included in this example. The common SGLX parameters are described in the Cluster Migration white paper.

```

TKIT_DIR                /usr/local/cmcluster/mysqltoolkitmod/
CONFIGURATION_FILE_PATH /home/mysql/mysqlhome/my.cnf
DATA_DIRECTORY          /home/mysql/mysqlhome
PID_FILE                /usr/local/cmcluster/pkg/mysqlmod/mysqld.pid
MAINTENANCE_FLAG       yes
RETRY_TIMES             0
MONITOR_INTERVAL       5
TIME_OUT                30

```

Executing the cmgetpkgenv <pkgname> command (Only for Modular packages)

In the SGLX MySQL Toolkit Modular package, all the parameters are stored in the Cluster Database (CDB). Run the following command to view this information:

```
# cmgetpkgenv <pkg_name> > pkg.info
```

For this example:

```
# cmgetpkgenv mysqlmod > mysqlmod.info
```

Following is an example output with a MySQL Toolkit package configured. The package "mysqlmod" is a MySQL Toolkit package. Note that the parameter names may have prefixes in this case.

```
SG_PACKAGE_NAME="mysqlmod"
SG_PACKAGE_DESCRIPTION="Serviceguard Package"
SG_MODULE_NAME[0]="sg/basic"
SG_MODULE_VERSION[0]="1"
SG_MODULE_NAME[1]="sg/failover"
SG_MODULE_VERSION[1]="1"
SG_MODULE_NAME[2]="sg/priority"
SG_MODULE_VERSION[2]="1"
SG_MODULE_NAME[3]="sg/filesystem"
SG_MODULE_VERSION[3]="1"
SG_MODULE_NAME[4]="sg/monitor_subnet"
SG_MODULE_VERSION[4]="1"
SG_MODULE_NAME[5]="sg/package_ip"
SG_MODULE_VERSION[5]="1"
SG_MODULE_NAME[6]="sg/service"
SG_MODULE_VERSION[6]="1"
SG_MODULE_NAME[7]="sg/volume_group"
SG_MODULE_VERSION[7]="1"
SG_MODULE_NAME[8]="tkit/mysql/mysql"
SG_MODULE_VERSION[8]="1"
SG_PACKAGE_TYPE="failover"
SG_NODE_NAME[0]="*"
SG_AUTO_RUN="yes"
SG_NODE_FAIL_FAST_ENABLED="no"
SG_RUN_SCRIPT_TIMEOUT="no_timeout"
SG_HALT_SCRIPT_TIMEOUT="no_timeout"
SG_SUCESSOR_HALT_TIMEOUT="no_timeout"
SG_SCRIPT_LOG_FILE="/usr/local/cmcluster/run/log/mysqlmod.log"
SG_OPERATION_SEQUENCE[0]="$SGCONF/scripts/sg/volume_group.sh"
SG_OPERATION_SEQUENCE[1]="$SGCONF/scripts/sg/filesystem.sh"
SG_OPERATION_SEQUENCE[2]="$SGCONF/scripts/sg/package_ip.sh"
SG_OPERATION_SEQUENCE[3]="$SGCONF/scripts/tkit/mysql/tkit_module.sh"
SG_OPERATION_SEQUENCE[4]="$SGCONF/scripts/sg/service.sh"
SG_LOG_LEVEL="5"
SG_FAILOVER_POLICY="configured_node"
SG_FAILBACK_POLICY="manual"
SG_PRIORITY="no_priority"
SG_CONCURRENT_FSCK_OPERATIONS="1"
SG_CONCURRENT_MOUNT_AND_UMOUNT_OPERATIONS="1"
SG_FS_MOUNT_RETRY_COUNT="0"
SG_FS_UMOUNT_RETRY_COUNT="1"
SG_FS_NAME[0]="/dev/vgsq1/lvsq1"
SG_FS_DIRECTORY[0]="/home/mysql"
SG_FS_TYPE[0]="ext3"
SG_FS_MOUNT_OPT[0]="-o rw"
SG_FS_UMOUNT_OPT[0]=""
SG_FS_FSCK_OPT[0]=""
SG_IP_SUBNET[0]="15.154.62.0"
SG_IP_ADDRESS[0]="15.154.63.181"
SG_SERVICE_NAME[0]="mysql_service"
SG_SERVICE_CMD[0]="$SGCONF/scripts/tkit/mysql/tkit_module.sh mysql_monitor"
SG_SERVICE_RESTART[0]="none"
SG_SERVICE_FAIL_FAST_ENABLED[0]="no"
SG_SERVICE_HALT_TIMEOUT[0]="300"
```

```

SG_VGCHANGE_CMD="vgchange -a y"
SG_VG[0]="vgsql"
MYSQL_TKIT_DIR="/usr/local/cmcluster/mysqltoolkitmod/"
MYSQL_CONFIGURATION_FILE_PATH="/home/mysql/mysqlhome/my.cnf"
MYSQL_DATA_DIRECTORY="/home/mysql/mysqlhome"
MYSQL_PID_FILE="/usr/local/cmcluster/pkg/mysqlmod/mysqld.pid"
MYSQL_MAINTENANCE_FLAG="yes"
MYSQL_RETRY_TIMES="0"
MYSQL_MONITOR_INTERVAL="5"
MYSQL_TIME_OUT="30"

```

Saving Serviceguard MySQL Toolkit Parameters

The Serviceguard parameters, such as Volume Groups, File Systems or IP addresses are common to most toolkits. These parameters are saved in the Serviceguard Package Configuration worksheet. A blank copy of the worksheet, with added MySQL Toolkit parameters, is included in the Appendix section of this white paper.

The unique SGLX MySQL Toolkit parameters and other configuration information need to be saved. The SGLX MySQL Toolkit parameters, collected using one of the methods mentioned in the earlier section, are stored in the following MySQL Toolkit section of the Serviceguard Package Configuration worksheet.

Serviceguard Package Configuration parameters	Red Hat service and resource configuration parameters
CONFIGURATION_FILE_PATH ____/home/mysql/mysqlhome/my.cnf____ OR DATA_DIRECTORY ____/home/mysql/mysqlhome____ TIME_OUT ____30____	Config File ____/home/mysql/mysqlhome/my.cnf____ OR Config File <SGLX DATADIRECTORY>/my.cnf ³ Listen Address ____ 15.154.63.181 ____ mysqld Options _____ Shutdown Wait (seconds) ____30____

Note:

1. The Listen address field of the RHCS MySQL resource can be left empty. If left empty, the RA finds the IP addresses available in the service and uses the first IP address that is found.
2. If DATA_DIRECTORY is specified in the SGLX configuration file as /home/abc/def, then the Config File parameter in the RHCS MySQL RA should be set to /home/abc/def/my.cnf

The SGLX PostgreSQL Toolkit

The Serviceguard for Linux PostgreSQL Toolkit A.04.01 is a separate set of shell scripts that allows users to configure a Serviceguard package to monitor a PostgreSQL Database application. The Toolkit consists of

an 'rpm' file which, after installation and configuration, enables an SGLX package to provide high availability for one or more PostgreSQL Database servers. It works as a subsystem under the SGLX core software to provide functions for configuring, starting, shutting down, and monitoring the PostgreSQL server application in a cluster environment. The sample SGLX PostgreSQL Toolkit scripts that are included in this white paper are from the PostgreSQL Toolkit version A.04.01.

The "Introduction" section in this document has the description of a few key differences between the SGLX PostgreSQL toolkit and the implementation of the RHCS PostgreSQL RA. The following features are different between the SGLX PostgreSQL Toolkit and the RHCS PostgreSQL RA. These differences are expected to have little or no impact in most environments.

- In the SGLX PostgreSQL Toolkit, the `DATA_DIRECTORY` parameter is used to specify the PostgreSQL data directory location.

In the RHCS PostgreSQL RA, the value of PostgreSQL server data directory is not defined by the user. The data directory value is read from the PostgreSQL configuration file. However, if the user does not want to set this in the PostgreSQL configuration file, then this value can be specified in the "Postmaster options" parameter as "-D DATA_DIRECTORY".

- In the SGLX PostgreSQL Toolkit, the `LOG_FILE` parameter is used to specify the PostgreSQL log file location.

In the RHCS PostgreSQL RA, this parameter is not defined by the user. The log file location is read from the PostgreSQL configuration file. If the user wants to change the default log file location defined in the PostgreSQL configuration file, the user must edit the configuration file.

- In the SGLX PostgreSQL Toolkit, the `SERVER_PATH` parameter is used to specify the PostgreSQL server binary location.

In the RHCS PostgreSQL RA, the user cannot set this as a resource parameter. However, the user can edit the PostgreSQL RA script and change the `PSQL_POSTMASTER` variable and set it to an alternate path.

- In the SGLX PostgreSQL Toolkit, the `PID_FILE` parameter holds the process ID file path of the PostgreSQL Server instance. The user can customize the PID file location and it is passed to the PostgreSQL server as a startup parameter.

In the RHCS PostgreSQL RA, the PID file location cannot be specified by the user, but the RA generates the PID file and adds it to the PostgreSQL server configuration file (which is read by the PostgreSQL server during startup). If required, the user can customize the PID file location by editing the PostgreSQL configuration file. If there is more than one PostgreSQL server instance on the same node and the PID file location is edited by the user, then it is the user's responsibility to ensure unique PID file locations. It is also the user's responsibility to set the correct permission and ownership to the PID file(s) and the parent directory.

In the RHCS PostgreSQL RA, the PID file is created under the `/var/run/cluster/<postgres-8-RA-scriptname>` directory. The filename is generated with a combination of the PostgreSQL RA script name and the resource instance name. In our RHCS PostgreSQL resource setup, the PID file was created at `/var/run/cluster/postgres-8/postgres-8:Postgresdb.pid`

- In the SGLX PostgreSQL Toolkit, the configuration file specified by the user is used and a new configuration file is not generated.

The RHCS PostgreSQL RA generates a new configuration file based on the user specified configuration file. The new configuration file is generated at `/etc/cluster/<PostgreSQL RA scriptname>/<PostgreSQL RA scriptname:PostgreSQL resource instancename>/postgresql.conf`. In our RHCS PostgreSQL resource setup, the configuration file is generated at `/etc/cluster/postgres-8/postgres-8:Postgresdb/postgresql.conf`

A new configuration file is generated from the user-specified configuration file (read from the user in the `OCF_RESKEY_config_file` parameter). The following information is included in the new configuration file.

- Comment section that includes instructions on how to edit the new configuration file
- The new external PID file location
- The IP addresses associated with the service
- The listen addresses and external PID file location, if any were present in the original configuration file, are commented out
- The checksum information of the new configuration file (automatically generated and used by the RA to determine if there has been a change to the configuration file)

The new configuration file has to be edited and copied to the same location on all the nodes of the cluster. The default log file location, generated PID file location, and generated listen_address IPs can be changed by editing the newly generated configuration file.

During each PostgreSQL resource start, the generated configuration file is checked to find out whether it has been modified by the user, after it was generated. If it is not modified, a new configuration file is once again generated. If it is modified, the new configuration file is not generated again. The resource is started using the new configuration file. This new configuration file generation is done until the configuration file generated in the previous run is altered.

- In the SGLX PostgreSQL Toolkit, the `RETRY_TIMES` parameter is used to specify the number of times the PostgreSQL server status is verified in case of a monitoring failure. While monitoring the PostgreSQL server, if the PostgreSQL server is down, the toolkit scripts will retry checking the PostgreSQL server status for `RETRY_TIMES` before returning an error. The delay between the successive retries is 5 seconds, though it can be customized by editing the `hapgsq1.mon` script. The recommended and default value of the parameter `RETRY_TIMES` is 0.

In the RHCS PostgreSQL RA, if the PostgreSQL server is down the RA does not recheck the server status and returns an error immediately so that failover can occur.

- The SGLX PostgreSQL Toolkit uses the `pg_ctl` utility to start, stop and restart the PostgreSQL server. The RHCS PostgreSQL RA does not use the `pg_ctl` utility and instead calls the postmaster server directly.

Gathering Serviceguard Parameters

The Serviceguard cluster parameters need to be saved. They are translated to the equivalent RHCS parameters so that users can have similar functionality. The most important step in the migration is saving the Serviceguard parameters in the Serviceguard Package Configuration worksheet. Part of this process must be completed while the SGLX cluster is still available. The cluster does not need to be up during this part of the process but certain cluster commands do need to be able to run.

Gathering Serviceguard Cluster parameters

The first step is to gather cluster information, including the names of the packages that are configured in the cluster. Run the following command on any one of the Serviceguard cluster nodes to collect this information:

```
#cmviewcl -v > cluster.info
```

Following is an example output of this command, where a PostgreSQL Toolkit package is configured. The package "pgsqlmod" is a PostgreSQL Toolkit package:

```
CLUSTER          STATUS
pgsqlcl         up

NODE            STATUS          STATE
drama           up              running

  Quorum_Server_Status:
  NAME          STATUS          STATE
  linuxlp64.cup.hp.com  up              running

  Network_Parameters:
  INTERFACE     STATUS          NAME
  PRIMARY       up              eth2

NODE            STATUS          STATE
poetry         up              running

  Quorum_Server_Status:
  NAME          STATUS          STATE
  linuxlp64.cup.hp.com  up              running

  Network_Parameters:
  INTERFACE     STATUS          NAME
  PRIMARY       up              eth2

PACKAGE         STATUS          STATE          AUTO_RUN      NODE
pgsqlmod        up              running        enabled        poetry

  Policy_Parameters:
  POLICY_NAME    CONFIGURED_VALUE
  Failover       configured_node
  Failback       manual

  Script_Parameters:
  ITEM          STATUS          MAX_RESTARTS  RESTARTS      NAME
  Service       up              0              0              pgsql_service

  Node_Switching_Parameters:
  NODE_TYPE     STATUS          SWITCHING      NAME
```

<i>Primary</i>	<i>up</i>	<i>disabled</i>	<i>drama</i>
<i>Alternate</i>	<i>up</i>	<i>enabled</i>	<i>poetry (current)</i>

Gathering Serviceguard PostgreSQL Toolkit Parameters

The next step is gathering the SGLX PostgreSQL Toolkit parameters.

The toolkit configuration information can be obtained using any one of the following methods:

- ✓ Toolkit configuration file (`hapgsq1.conf`), residing in the toolkit configuration directory or the package directory
- ✓ Package configuration file (Only for Modular packages)
- ✓ Executing the `cmgetpkgenv <pkgname>` command (Only for Modular packages)

Toolkit Configuration file (Modular and Legacy packages)

File location in the SGLX PostgreSQL Toolkit Legacy Packages:

In the case of the SGLX PostgreSQL Toolkit Legacy packages, the toolkit-specific configuration information can be obtained from the file `hapgsq1.conf` residing in the package directory. The file, `hapgsq1.conf`, is in the same location as that of the toolkit Interface Script (`toolkit.sh`). In the package control script, the user specifies this location as a value for the `SERVICE_CMD` attribute and also in the `customer_defined_run_cmds` and `customer_defined_halt_cmds` subroutines.

Following is an example of the sections in the package control script where the user defines the package directory. In the case of Legacy packages, in this example, the `hapgsq1.conf` is in the `/usr/local/cmcluster/postgresq1toolkitleg` directory.

```
SERVICE_NAME[0]="postgressvc"
SERVICE_CMD[0]="/usr/local/cmcluster/postgresq1toolkitleg/toolkit.sh monitor"
SERVICE_RESTART[0]=""

# Setting the log file
log_file=${SG_SCRIPT_LOG_FILE:-$0.log}

# START OF CUSTOMER DEFINED FUNCTIONS

# This function is a place holder for customer define functions.
# You should define all actions you want to happen here, before the service is
# started. You can create as many functions as you need.
#
function customer_defined_run_cmds
{
# ADD customer defined run commands.
: # do nothing instruction, because a function must contain some command.
/usr/local/cmcluster/postgresq1toolkitleg/toolkit.sh start
test_return 51
}

# This function is a place holder for customer define functions.
# You should define all actions you want to happen here, after the service is
# halted.
#
function customer_defined_halt_cmds
{
# ADD customer defined halt commands.
```

```

: # do nothing instruction, because a function must contain some command.
/usr/local/cmcluster/postgresqltoolkitleg/toolkit.sh stop
test_return 52
}

# END OF CUSTOMER DEFINED FUNCTIONS

```

File location in the SGLX PostgreSQL Toolkit Modular Packages:

In case of the SGLX PostgreSQL Toolkit Modular packages, the toolkit configuration file (`hapgsq1.conf`) is generated in the toolkit configuration directory (`TKIT_DIR`) defined in the package configuration file. The toolkit configuration file has all the required PostgreSQL Toolkit parameters.

Following is an example of how the toolkit configuration directory is defined in the package configuration file in the SGLX PostgreSQL Toolkit package configuration file:

```

# *****LINUX POSTGRESQL TOOLKIT PARAMETERS*****
#
# Define Toolkit configuration directory
#
# This directory holds the toolkit configuration file. This parameter directs
# cmapplyconf to generate the hapgsq1.conf file under this directory. To put
# toolkit into maintenance mode, create pgsq1.debug file under this directory.
# This is synonymous with the package directory acronym.
#
# Legal values for TKIT_DIR: <Any String>

TKIT_DIR                /usr/local/cmcluster/postgresqltoolkitmod

```

The `hapgsq1.conf` file contains the same PostgreSQL Toolkit parameters for both the Legacy and Modular type of packages. Following is an example of the toolkit configuration file (`hapgsq1.conf`):

```

DATA_DIRECTORY=/home/postgres/postgreshome
LOG_FILE=/home/postgres/postgreshome/pgsq1.log
PID_FILE=$DATA_DIRECTORY/postmaster.pid
SERVER_PATH=/usr/bin/postmaster
MAINTENANCE_FLAG=yes
RETRY_TIMES=2
MONITOR_INTERVAL=5

```

Package configuration file (Only for Modular packages)

In case of the SGLX PostgreSQL Toolkit Modular packages, the package configuration file also has the PostgreSQL Toolkit parameters. To view the current package configuration file, run the following command:

```
# cmgetconf -p <package_name> > pkg.info
```

For this example:

```
# cmgetconf -p pgsq1mod > pgsq1mod.info
```

Following is an example of the SGLX PostgreSQL Toolkit Modular package configuration file. Only the sections that have PostgreSQL Toolkit parameters are included here for ease of reading. The other sections in the PostgreSQL Toolkit Modular package configuration file that have the common SGLX parameters are

not included in this example. The common SGLX parameters are described in the Cluster Migration white paper.

TKIT_DIR	/usr/local/cmcluster/postgresqltoolkitmod
DATA_DIRECTORY	/home/postgres/postgreshome
LOG_FILE	/home/postgres/postgreshome/pgsql.log
PID_FILE	\${DATA_DIRECTORY}/postmaster.pid
SERVER_PATH	/usr/bin/postmaster
MAINTENANCE_FLAG	yes
MONITOR_INTERVAL	5
RETRY_TIMES	2

Executing the `cmgetpkgenv <pkgname> command (Only for Modular packages)`

In case of the SGLX PostgreSQL Toolkit Modular package, all the parameters are stored in the Cluster Database (CDB). Run the following command to view this information:

```
# cmgetpkgenv <pkg_name> > pkg.info
```

For this example:

```
# cmgetpkgenv pgsqmod > pgsqmod.info
```

Following is an example output with a PostgreSQL Toolkit package configured. The package "pgsqmod" is a PostgreSQL Toolkit package. Note that the parameter names may have prefixes in this case.

```
SG_PACKAGE_NAME="pgsqmod"
SG_PACKAGE_DESCRIPTION="Serviceguard Package"
SG_MODULE_NAME[0]="sg/basic"
SG_MODULE_VERSION[0]="1"
SG_MODULE_NAME[1]="sg/failover"
SG_MODULE_VERSION[1]="1"
SG_MODULE_NAME[2]="sg/priority"
SG_MODULE_VERSION[2]="1"
SG_MODULE_NAME[3]="sg/filesystem"
SG_MODULE_VERSION[3]="1"
SG_MODULE_NAME[4]="sg/package_ip"
SG_MODULE_VERSION[4]="1"
SG_MODULE_NAME[5]="sg/service"
SG_MODULE_VERSION[5]="1"
SG_MODULE_NAME[6]="sg/volume_group"
SG_MODULE_VERSION[6]="1"
SG_MODULE_NAME[7]="tkit/pgsql/pgsql"
SG_MODULE_VERSION[7]="1"
SG_PACKAGE_TYPE="failover"
SG_NODE_NAME[0]="*"
SG_AUTO_RUN="yes"
SG_NODE_FAIL_FAST_ENABLED="no"
SG_RUN_SCRIPT_TIMEOUT="no_timeout"
SG_HALT_SCRIPT_TIMEOUT="no_timeout"
SG_SUCESSOR_HALT_TIMEOUT="no_timeout"
SG_SCRIPT_LOG_FILE="/usr/local/cmcluster/run/log/pgsqmod.log"
SG_OPERATION_SEQUENCE[0]="$SGCONF/scripts/sg/volume_group.sh"
SG_OPERATION_SEQUENCE[1]="$SGCONF/scripts/sg/filesystem.sh"
SG_OPERATION_SEQUENCE[2]="$SGCONF/scripts/sg/package_ip.sh"
SG_OPERATION_SEQUENCE[3]="$SGCONF/scripts/tkit/pgsql/tkit_module.sh"
SG_OPERATION_SEQUENCE[4]="$SGCONF/scripts/sg/service.sh"
SG_LOG_LEVEL="5"
SG_FAILOVER_POLICY="configured_node"
```

```

SG_FAILBACK_POLICY="manual"
SG_PRIORITY="no_priority"
SG_CONCURRENT_FSCK_OPERATIONS="1"
SG_CONCURRENT_MOUNT_AND_UMOUNT_OPERATIONS="1"
SG_FS_MOUNT_RETRY_COUNT="0"
SG_FS_UMOUNT_RETRY_COUNT="1"
SG_FS_NAME[0]="/dev/vgsql/lvsq1"
SG_FS_DIRECTORY[0]="/home/postgres"
SG_FS_TYPE[0]="ext3"
SG_FS_MOUNT_OPT[0]="-o rw"
SG_FS_UMOUNT_OPT[0]=""
SG_FS_FSCK_OPT[0]=""
SG_IP_SUBNET[0]="15.154.62.0"
SG_IP_ADDRESS[0]="15.154.63.181"
SG_SERVICE_NAME[0]="pgsql_service"
SG_SERVICE_CMD[0]="$SGCONF/scripts/tkit/pgsql/tkit_module.sh pgsql_monitor"
SG_SERVICE_RESTART[0]="none"
SG_SERVICE_FAIL_FAST_ENABLED[0]="no"
SG_SERVICE_HALT_TIMEOUT[0]="300"
SG_VGCHANGE_CMD="vgchange -a y"
SG_VG[0]="vgsql"
PGSQL_TKIT_DIR="/usr/local/cmcluster/postgresqltoolkitmod"
PGSQL_DATA_DIRECTORY="/home/postgres/postgreshome"
PGSQL_LOG_FILE="/home/postgres/postgreshome/pgsql.log"
PGSQL_PID_FILE="${DATA_DIRECTORY}/postmaster.pid"
PGSQL_SERVER_PATH="/usr/bin/postmaster"
PGSQL_MAINTENANCE_FLAG="yes"
PGSQL_MONITOR_INTERVAL="5"
PGSQL_RETRY_TIMES="2"

```

Saving Serviceguard PostgreSQL Toolkit Parameters

The Serviceguard parameters, such as Volume Groups, File Systems or IPAddresses are common to most toolkits. These parameters are saved in the Serviceguard Package Configuration worksheet. A blank copy of the worksheet, with a separate table for PostgreSQL Toolkit parameters, is included in the Appendix section of this white paper.

The unique SGLX PostgreSQL Toolkit parameters and other configuration information need to be saved. The SGLX PostgreSQL Toolkit parameters collated from one of the methods mentioned in the earlier section is stored in the following PostgreSQL Toolkit section of the Serviceguard Package Configuration worksheet.

Serviceguard Package Configuration parameters	Red Hat service and resource configuration parameters
DATA_DIRECTORY ____/home/postgres/postgreshome____	Config File ____ /home/postgres/postgreshome/postgresql.conf Postmaster Options ____ -D /home/postgres/postgreshome ____ Postmaster User ____ postgres ____ Shutdown Wait (seconds) ____ 5 ____

LOG_FILE _____	Edit the postgresQL configuration file
PID_FILE ____ "\${DATA_DIRECTORY}/postmaster.pid" _	Edit the postgresQL configuration file
SERVER_PATH ____/usr/bin/postmaster____	Edit the PostgreSQL RA script

Notes:

1. The log file location and name are defined in the PostgreSQL configuration file. Edit the PostgreSQL generated configuration file to change the default log file location. It is the user's responsibility to set the correct permission and ownership for the log file and the parent directory.
2. The PID file location is generated by the PostgreSQL RA. Edit the generated configuration file to change the PID file location. If there is more than one PostgreSQL server instance on the same node and the PID file location has been edited, then it is the user's responsibility to ensure a unique PID file location for each PostgreSQL instance. It is also the user's responsibility to set the correct permission and ownership for the PID file and the parent directory. **It is recommended NOT to change the generated PID file location.**
3. The PostgreSQL server path is defined in the PostgreSQL RA, `/usr/share/cluster/postgres-8.sh`, as follows: `declare PSQL_POSTMASTER="/usr/bin/postmaster"`. Edit the variable only if required to change the default PostgreSQL server location.

Migrating Common SGLX Package Parameters to RAs

There are a number of functions that are common to most Serviceguard for Linux packages. These include the Volume groups, File systems, and IPAddresses. In an RHCS cluster, these are managed by a number of resource agents. This entire section is common to both MySQL and PostgreSQL setups.

Volume groups

Volume groups are treated one of two ways. If SCSI-3 PR fencing is used, the volume groups are converted to CLVM type volumes. The process for this is described in the Cluster Migration white paper and is repeated here. Once converted, these volume groups are managed by CLVMD and are activated at system start. Therefore, no volume group is controlled by a resource agent. For other fencing methods traditional volume groups are used, as they are in the MySQL Toolkit and PostgreSQL Toolkit, and therefore an LVM RA is used.

SCSI-3 PR fencing

To convert the volume groups to CLVM, edit the `/etc/lvm/lvm.conf` file to set the parameter `locking_type` to 3. To convert the non-clustered Serviceguard volume group `vgsql` to a clustered volume group, run the command `vgchange -c y vgsql`.

Following is an example of the commands execution.

1. Open the file `/etc/lvm/lvm.conf` in the vi editor. Edit the `locking_type` and set it to 3. Following is a section of the `lvm.conf` file, where `locking_type` is set.

```
# Type of locking to use. Defaults to local file-based locking
(1) .
# Turn locking off by setting to 0 (dangerous: risks metadata
corruption
# if LVM2 commands get run concurrently).
# Type 2 uses the external shared library locking_library.
# Type 3 uses built-in clustered locking.
locking_type = 3
```

2. Run the command `vgchange -c y vgsql` in the same node where the `lvm.conf` is edited. Following is an example output of this command:

```
# vgchange -c y vgsql
Volume group "vgsql" successfully changed
```

3. Execute `vgdisplay vgsql` to verify whether the conversion has happened properly. Following is an example output of this command.

```
# vgdisplay vgsql
--- Volume group ---
VG Name                vgsql
System ID
Format                 lvm2
Metadata Areas        1
```

```

Metadata Sequence No  544
VG Access              read/write
VG Status              resizable
Clustered              yes
Shared                 no
MAX LV                 0
Cur LV                1
Open LV                0
Max PV                 0
Cur PV                1
Act PV                 1
VG Size                10.00 GB
PE Size                4.00 MB
Total PE               2559
Alloc PE / Size        2509 / 9.80 GB
Free PE / Size         50 / 200.00 MB
VG UUID                bJlnP8-00ce-HH9p-JR4b-N3Ca-Np44-6jYekU

```

Other fencing methods

The `vg` parameter in the package configuration file specifies the LVM volume group used by the package. Multiple vgs can be defined for a package.

Create a separate LVM resource for every LVM volume group entry in the package configuration. The example of the LVM RA setup is shown in the XML in the [“Setting up the RHCS MySQL service”](#) section.

File systems

At start time, a Serviceguard package activates one or more logical volumes (associated with the file systems) and then mounts the file systems. At halt time, the package un-mounts the file systems. In the case of Legacy packages, the package control script contains the necessary file system commands to complete the mount, un-mount, and fsck operations.

In Serviceguard, the `fs_name` parameter, in conjunction with `fs_directory`, `fs_type`, `fs_mount_opt`, `fs_umount_opt`, and `fs_fsck_opt` parameters, specifies a file system that is to be mounted by the package. The `fs_name` (parameter `LV` in Legacy packages) specifies the block device file for a logical volume and the parameter `fs_directory` (parameter `FS` in legacy packages) is the mount point of the file system specified by `fs_name`.

In case of Modular packages, the file system information is specified as follows in the package configuration file:

<code>fs_name</code>	<code>/dev/vgsql/lvsq1</code>
<code>fs_directory</code>	<code>/home/mysql</code>
<code>fs_type</code>	<code>ext3</code>
<code>fs_mount_opt</code>	<code>"-o rw"</code>
<code>fs_umount_opt</code>	<code>" "</code>
<code>fs_fsck_opt</code>	<code>" "</code>

In case of Legacy packages, the file system information is specified as follows in the package control script:

```
LV[0]=/dev/vgsql/lvsq1; FS[0]=/home/mysql; FS_TYPE[0]="ext3";
FS_MOUNT_OPT[0]="-o rw"
FS_UMOUNT_OPT[0]=""; FS_FSCK_OPT[0]=""
```

In RHCS, file systems are treated in one of the following two ways:

- If SCSI-3 PR fencing is not used, the underlying volume groups are LVM resources. The LVM resources configured in the “Volume group” migration step use lvm tags and thereby ensure VG exclusive activation. Create a separate file system resource for every file system entry in the package configuration.
- If SCSI-3 PR fencing is used, the underlying volume groups are CLVM type volumes. Create a separate file system resource for every file system entry in the package configuration. This approach is simple, but it does not ensure VG exclusive activation. If the user wants the cluster to ensure the protection to data integrity provided by VG exclusive, the file system has to be converted to a cluster file system (GFS) and then create a separate GFS file system resource for every file system entry in the package configuration.

In RHCS, the SGLX parameters translate to the `mountpoint`, `device`, `fstype`, and `options` parameter of the file system resource agent as shown in the following table:

SGLX Modular	SGLX Legacy	Description	RHCS
<code>fs_name</code>	<code>LV[x]</code>	Logical volume	<code>device</code>
<code>fs_directory</code>	<code>FS[x]</code>	Mount point directory	<code>mountpoint</code>
<code>fs_type</code>	<code>FS_TYPE[x]</code>	File system type (e.g. ext3)	<code>fstype</code>
<code>fs_mount_opt</code>	<code>FS_MOUNT_OPT[x]</code>	Option used for mounting	<code>options</code>
<code>fs_umount_opt</code>	<code>FS_UMOUNT_OPT[x]</code>	Options used for unmounting	N/A

Following are a few recommendations to keep in mind when migrating Serviceguard packages to RHCS:

- For every `fs_name` entry in the Serviceguard package, a corresponding RHCS file system resource needs to be created. Ensure that resources are defined in the `/etc/cluster/cluster.conf` file in the order in which the `fs_name` parameters are defined in the Serviceguard package configuration file.
- Set the `force_fsck` option for the Red Hat file system resource so that a file system check is done on the file system before mounting it.
- Use `force_umount` option in conjunction with `self_fence` so that if the umount fails the node will be reset.

The example of the file system RA setup is shown in the XML in the [“Setting up the RHCS MySQL service”](#) section.

IPAddresses

At service start time, a Serviceguard package can have one or more IP addresses associated with it. When a package is started, the relocatable IP addresses are added to the specified IP subnet. When the package is stopped, the relocatable IP addresses are deleted from the specified subnet. In the case of Legacy packages, the package control script contains the necessary Serviceguard commands to add the IP addresses to the specified subnet. In the case of Modular packages, the Serviceguard IP module has the necessary Serviceguard commands to add the IP addresses to the specified subnet.

In case of Legacy packages, the `IP` and `SUBNET` parameters specify the IP and Subnet address pairs used by the package. Following is an example of the IP address information in the Legacy package control script.

```
IP[0]="15.154.63.181"  
SUBNET[0]="15.154.62.0"
```

In the case of Modular packages, the `ip_subnet` and `ip_address` parameters specify the IP and subnet address pairs used by the package. Following is an example of the IP address information in the modular package configuration file.

```
ip_subnet      15.154.62.0  
ip_address     15.154.63.181
```

Create a separate IP resource for every `ip_address` entry in the SGLX package configuration.

In RHCS, only the IP address can be specified. The subnet monitoring is not available on RHCS. The `ip_address` or `IP[x]` parameter of SGLX translates to the `IP Address` parameter of the IP Resource Agent (RA) in RHCS.

Following are a few recommendations to keep in mind while migrating Serviceguard packages to RHCS:

- An RHCS IP resource needs to be created for every `ip_address` (or `IP[x]`) entry in the Serviceguard package. Ensure that resources are defined in the `/etc/cluster/cluster.conf` file in the order in which the IPAddresses are defined in the Serviceguard package configuration file.
- Set the `monitor_link` option for the RHCS IP resource so that the `status` function will fail if the link on the NIC bounded to the IP address is not present. Setting the `monitor_link` option will trigger the resource manager to relocate the service to another node in the cluster when the NIC, to which the IP is tied, fails.
- Any `monitored_subnets` from an SGLX package that are not associated with an `ip_address` will not be monitored by the RHCS cluster. Only the subnets of the monitored IP addresses are monitored on the RHCS cluster.

The example of the IP address RA setup is shown in the XML in the "[Setting up the RHCS MySQL service](#)" section.

Note:

Red Hat Cluster *Suite* supports, both IPv4 and IPv6 addresses for cluster service usage. For cluster communication (heartbeat), only IPv4 addresses are supported.

MySQL Resource Agent Migration Procedure

This section covers the details that are unique to MySQL Toolkit in the migration process. A step by step procedure is described for a MySQL package. Any differences related to Legacy and Modular packages are described as well.

MySQL Toolkit Migration Planning

In this section, the required RHCS resources are determined. The Serviceguard Package Configuration worksheet is filled with values from the Serviceguard cluster configuration files.

Following are the package configuration parameters of the MySQL Toolkit Modular package:

<code>package_name</code>	<code>mysqlmod</code>
<code>package_description</code>	<code>"Serviceguard Package"</code>
<code>module_name</code>	<code>sg/basic</code>
<code>module_version</code>	<code>1</code>
<code>module_name</code>	<code>sg/failover</code>
<code>module_version</code>	<code>1</code>
<code>module_name</code>	<code>sg/priority</code>
<code>module_version</code>	<code>1</code>
<code>module_name</code>	<code>sg/filesystem</code>
<code>module_version</code>	<code>1</code>
<code>module_name</code>	<code>sg/package_ip</code>
<code>module_version</code>	<code>1</code>
<code>module_name</code>	<code>sg/service</code>
<code>module_version</code>	<code>1</code>
<code>module_name</code>	<code>sg/volume_group</code>
<code>module_version</code>	<code>1</code>
<code>module_name</code>	<code>tkit/mysql/mysql</code>
<code>module_version</code>	<code>1</code>
<code>package_type</code>	<code>failover</code>
<code>node_name</code>	<code>*</code>
<code>auto_run</code>	<code>yes</code>
<code>node_fail_fast_enabled</code>	<code>no</code>
<code>run_script_timeout</code>	<code>no_timeout</code>
<code>halt_script_timeout</code>	<code>no_timeout</code>
<code>successor_halt_timeout</code>	<code>no_timeout</code>
<code>script_log_file</code>	<code>\$\$SRUN/log/\$\$SG_PACKAGE.log</code>
<code>operation_sequence</code>	<code>\$\$SGCONF/scripts/sg/volume_group.sh</code>
<code>operation_sequence</code>	<code>\$\$SGCONF/scripts/sg/filesystem.sh</code>
<code>operation_sequence</code>	<code>\$\$SGCONF/scripts/sg/package_ip.sh</code>
<code>operation_sequence</code>	<code>\$\$SGCONF/scripts/tkit/mysql/tkit_module.sh</code>
<code>operation_sequence</code>	<code>\$\$SGCONF/scripts/sg/service.sh</code>
<code>log_level</code>	<code>5</code>
<code>failover_policy</code>	<code>configured_node</code>
<code>failback_policy</code>	<code>manual</code>
<code>priority</code>	<code>no_priority</code>
<code>concurrent_fsck_operations</code>	<code>1</code>
<code>concurrent_mount_and_umount_operations</code>	<code>1</code>
<code>fs_mount_retry_count</code>	<code>0</code>
<code>fs_umount_retry_count</code>	<code>1</code>
<code>fs_name</code>	<code>/dev/vgsql/lvsq1</code>
<code>fs_directory</code>	<code>/home/mysql</code>
<code>fs_type</code>	<code>ext3</code>
<code>fs_mount_opt</code>	<code>"-o rw"</code>
<code>#fs_umount_opt</code>	

```

#fs_fsck_opt
ip_subnet                15.154.62.0
#ip_subnet_node
ip_address               15.154.63.181
service_name             mysql_service
service_cmd              $SGCONF/scripts/tkit/mysql/tkit_module.sh mysql_monitor"
service_restart         none
service_fail_fast_enabled no
service_halt_timeout    300
vgchange_cmd            "vgchange -a y"
vg                       vgsq1
TKIT_DIR                 /usr/local/cmcluster/mysqltoolkitmod/
CONFIGURATION_FILE_PATH /home/mysql/mysqlhome/my.cnf
DATA_DIRECTORY          /home/mysql/mysqlhome
PID_FILE                /usr/local/cmcluster/pkg/mysqlmod/mysqld.pid
MAINTENANCE_FLAG        yes
RETRY_TIMES             0
MONITOR_INTERVAL        5
TIME_OUT                30

```

Following are the package configuration parameters of the MySQL Toolkit Legacy package:

```

PACKAGE_NAME             mysqlleg
PACKAGE_TYPE             FAILOVER
NODE_NAME                *
AUTO_RUN                 YES
NODE_FAIL_FAST_ENABLED  NO
RUN_SCRIPT               /usr/local/cmcluster/pkg/mysqlleg/mysqlleg.sh
HALT_SCRIPT              /usr/local/cmcluster/pkg/mysqlleg/mysqlleg.sh
RUN_SCRIPT_TIMEOUT      NO_TIMEOUT
HALT_SCRIPT_TIMEOUT     NO_TIMEOUT
SUCCESSOR_HALT_TIMEOUT  NO_TIMEOUT
SCRIPT_LOG_FILE         /usr/local/cmcluster/run/log/mysqlleg.log
FAILOVER_POLICY         CONFIGURED_NODE
FAILBACK_POLICY         MANUAL
PRIORITY                 NO_PRIORITY
SERVICE_NAME            mysqlsvc
SERVICE_FAIL_FAST_ENABLED no
SERVICE_HALT_TIMEOUT   300

```

Following are the package control script parameters of the MySQL Toolkit Legacy package:

```

VG[0]="vgsq1"
LV[0]="/dev/vgsq1/lvsq1"; FS[0]="/home/mysql"; FS_TYPE[0]="ext3";
FS_MOUNT_OPT[0]="-o rw"
FS_UMOUNT_OPT[0]="" ; FS_FSCK_OPT[0]=""
FS_UMOUNT_COUNT=1
FS_MOUNT_RETRY_COUNT=0
CONCURRENT_FSCK_OPERATIONS=1
CONCURRENT_MOUNT_AND_UMOUNT_OPERATIONS=1
IP[0]="15.154.63.181"
SUBNET[0]="15.154.62.0"
SERVICE_NAME[0]="mysqlsvc"
SERVICE_CMD[0]="/usr/local/cmcluster/mysqltoolkitleg/toolkit.sh monitor"
SERVICE_RESTART[0]=""

# Setting the log file
log_file=${SG_SCRIPT_LOG_FILE:-$0.log}

# START OF CUSTOMER DEFINED FUNCTIONS

```

```

# This function is a place holder for customer define functions.
# You should define all actions you want to happen here, before the service is
# started. You can create as many functions as you need.
#
function customer_defined_run_cmds
{
# ADD customer defined run commands.
: # do nothing instruction, because a function must contain some command.
/usr/local/cmcluster/mysqltoolkitleg/toolkit.sh start

    test_return 51
}

# This function is a place holder for customer define functions.
# You should define all actions you want to happen here, after the service is
# halted.
#
function customer_defined_halt_cmds
{
# ADD customer defined halt commands.
: # do nothing instruction, because a function must contain some command.
/usr/local/cmcluster/mysqltoolkitleg/toolkit.sh stop

    test_return 52
}

# END OF CUSTOMER DEFINED FUNCTIONS

```

The following table illustrates how this information is used to populate the Serviceguard Package Configuration worksheet.

SGLX package configuration parameters

RHCS service and resource configuration parameters

<p>Package Name _____mysqlmod_____</p> <p>Node_name _____drama_____</p> <p>Node_name _____poetry_____</p> <p>Failover_policy _____configured_node_____</p> <p>Failback_policy _____MANUAL_____</p> <p>Auto_run _____YES_____</p> <p>Node_fast_fail _____NO_____</p> <p>Package dependency</p> <p>dependency_name _____NOT_USED_____</p> <p>dependency_condition _____NOT_USED_____</p> <p>dependency_location _____NOT_USED_____</p>	<p>Service configuration</p> <p>Service name _____MySQL_svc_____</p> <p>Failover Domain _____MySQLdom_____</p> <p>Service Resources List</p> <p> _____MySQLlv_____</p> <p> _____MySQLfs_____</p> <p> _____MySQLdb_____</p> <p>Recovery policy _____Relocate_____</p> <p>(Restart, relocate, disable)</p> <p>Auto start service _____YES_____</p> <p>Run exclusive _____NO_____</p> <p>Hard recovery _____NO_____</p> <p>Depend _____NOT_USED_____</p> <p>Depend mode _____NOT_USED_____</p> <p>Managed Resources/Failover Domain configuration</p> <p>Domain name _____MySQLdom_____</p>
---	--

<p>Logical Volumes and File Systems</p> <p>fs_name ___/dev/vgsql/lvsq1___ fs_directory ___/home/mysql_____ fs_mount_opt ___-o rw_____ fs_umount_opt ___" "_____ fs_fsck_opt ___" "_____ fs_type ___ext3_____</p> <p>Network information</p> <p>IP ___15.154.63.181___ Subnet ___15.154.62.0_____ Monitored_Subnet ___NOT USED_____ Monitored_subnet_access ___NOT USED_</p> <p>LVM volume groups</p> <p>VG ___vgsql_____</p> <p>Service configuration</p> <p>service_name ___mysql_service___ service_cmd ___"\$SGCONF/scripts/tkit/mysql/tkit_m odule.sh mysql_monitor"___ service_restart ___none_____ service_fail_fast_enabled ___no___ service_halt_timeout ___300___</p> <p>Non-Serviceguard environment variable NOT USED</p> <p>External script</p> <p>external_script NOT USED</p> <p>priority ___no_priority___</p> <p>Priority _____ #Weight Name _____</p>	<p>Member node ___drama_____</p> <p>Member node ___poetry_____</p> <p>Restrict failover ___Yes_____</p> <p>Prioritized list ___drama, poetry_____</p> <p>Resources/File system</p> <p>Name ___MySQLfs_____</p> <p>File system type ___ext3_____</p> <p>Mount point ___/home/mysql____</p> <p>Device ___/dev/vgsql/lvsq1___</p> <p>Options ___rw_____</p> <p>File System ID _____</p> <p>Force unmount ___YES_____</p> <p>Reboot if unmount fails ___NO_____</p> <p>Check file system before mounting ___YES_____</p> <p>Resources/IP address</p> <p>IP address ___15.154.63.181_____</p> <p>Monitor Link ___YES_____</p> <p>Resource/LVM</p> <p>Name ___MySQLlv_____</p> <p>Volume group name ___vgsql_____</p> <p>Logical volume name ___lvsq1_____</p> <p>Script Resource NOT USED</p>
---	--

<pre>#Weight Value _____ #STORAGE_GROUP _____ #USER_NAME _____ #USER_HOST _____ #USER_ROLE _____ MySQL Toolkit variables CONFIGURATION_FILE_PATH _____/home/mysql/mysqlhome/my.cnf____ DATA_DIRECTORY _____/home/mysql/mysqlhome____ TIME_OUT _____30_____</pre>	<pre>Resource MySQL Client Resource Config File _____/home/mysql/mysqlhome/my.cnf____ Edit the my.cnf file and set the datadir Listen Address _____ 15.154.63.181____ Shutdown Wait (seconds) ____30_____</pre>
---	--

Migration Phase

During this phase, the RHCS cluster is configured and the RHCS MySQL resource is setup using the Package configuration worksheet. The actual migration is done in the following steps:

1. Setting up the cluster, quorum rule, fencing, failover domain, and resources
2. Setting up the RHCS MySQL service
3. Starting the RHCS MySQL service

Setting up the Cluster, Quorum Rule, Fencing, Failover Domain, and Resources

Set up the RHCS cluster. Set up the quorum rule and fencing. See the “Migrating an HP Serviceguard for Linux Cluster to Red Hat Cluster Suite for Red Hat Enterprise Linux 5” for information on setting up the cluster, quorum rule, and fencing.

The failover domains required for the migration of the Serviceguard package `mysqlmod` are setup. For information on the list of domains and the properties, see the Serviceguard Package Configuration worksheet prepared during the planning phase.

From the package configuration worksheet, `MySQLdom` is identified as the failover domain for the package `mysqlmod` and is of type “restricted, ordered”. The domain members are `drama` and `poetry`. Use the RHCS configuration utility to create the `MySQLdom` domain.

Setup the `LVM`, `Filesystem`, `IP` and `MySQL` resources as required for the migration of the Serviceguard package `mysqlmod`. For information on the resource configuration details, see the Serviceguard Package Configuration worksheet prepared during the planning phase.

Setting up the RHCS MySQL Service

The next step in the migration is setting up the RHCS MySQL cluster service, `MySQLsvc`. See the Serviceguard Package Configuration worksheet prepared during the planning phase for information on the service, its failover domain, and child resources.

After the configuration of the failover domains, the resources, and the service for MySQL, verify that the following required XML definitions are present in the `/etc/cluster/cluster.conf` file.

Following is an example of the `/etc/cluster/cluster.conf` with MySQL service setup. SCSI-3 PR fencing with automatic reset is also set up.

```
<?xml version="1.0"?>
<cluster alias="clust" config_version="76" name="clust">
  <fence_daemon clean_start="0" post_fail_delay="0" post_join_delay="3"/>
  <clusternodes>
    <clusternode name="poetry.ind.hp.com" nodeid="1" votes="1">
      <fence>
        <method name="1">
          <device name="SCSI_fence"
node="poetry.ind.hp.com"/>
        </method>
      </fence/>
    </clusternode>
    <clusternode name="drama.ind.hp.com" nodeid="2" votes="1">
      <fence>
        <method name="1">
          <device name="SCSI_fence"
node="drama.ind.hp.com"/>
        </method>
      </fence/>
    </clusternode>
  </clusternodes>
  <cman expected_votes="1" two_node="1"/>
  <fencedevices>
    <fencedevice agent="fence_scsi" name="SCSI_fence"/>
  </fencedevices>
  <rm>
    <failoverdomains>
      <failoverdomain name="MySQLdom" nofailback="0"
ordered="1" restricted="1">
        <failoverdomainnode name="drama.ind.hp.com"
priority="1"/>
        <failoverdomainnode name="poetry.ind.hp.com"
priority="1"/>
      </failoverdomain>
      <failoverdomain name="dramadom" nofailback="0"
ordered="0" restricted="1">
        <failoverdomainnode name="drama.ind.hp.com"
priority="1"/>
      </failoverdomain>
      <failoverdomain name="poetrydom" nofailback="0"
ordered="0" restricted="1">
        <failoverdomainnode name="poetry.ind.hp.com"
priority="1"/>
      </failoverdomain>
    </failoverdomains>
    <service autostart="1" domain="MySQLdom" exclusive="0"
name="MySQL_svc" recovery="relocate">
      <fs device="/dev/vgsql/lvsq1" force_fsck="1"
force_unmount="1" fsid="53777" fstype="ext3" mountpoint="/home/mysql"
name="MySQLfs" options="rw" self_fence="1"/>
      <ip address="15.154.63.181" monitor_link="1"/>
      <mysql config_file="/home/mysql/mysqlhome/my.cnf"
listen_address="15.154.63.181" name="MySQLdb" shutdown_wait="30"/>
    </service>
    <service autostart="0" domain="dramadom" exclusive="0"
```

```

max_restarts="0" name="watchdog_drama" recovery="restart"
restart_expire_time="0">
    <script ref="watchdog_script"/>
    </service>
    <service autostart="0" domain="poetrydom" exclusive="0"
max_restarts="0" name="watchdog_poetry" recovery="restart"
restart_expire_time="0">
    <script ref="watchdog_script"/>
    </service>
</rm>
<quorumd interval="2" label="qdisk_mysql" min_score="1" tko="5"
votes="1">
    <heuristic interval="2" program="/home/mysql/checkpvtlink.sh
eth1" score="1"/>
    </quorumd>
    <totem consensus="4800" join="60" token="20000"
token_retransmits_before_loss_const="20"/>
</cluster>

```

Note that the lvm line is not used in the above example as the volumes have been converted to CLVM. Additional failover domains are required if SCSI-3 PR fencing with automatic reset is setup.

Following is another example of the RHCS `/etc/cluster/cluster.conf` with MySQL service setup. In this example, iLO fencing is used.

```

<?xml version="1.0"?>
<cluster alias="clust" config_version="76" name="clust">
    <fence_daemon clean_start="0" post_fail_delay="0" post_join_delay="3"/>
    <clusternodes>
        <clusternode name="drama.ind.hp.com" nodeid="1" votes="1">
            <fence>
                <method name="1">
                    <device name="dramaiLO"/>
                </method>
            </fence>
        </clusternode>
        <clusternode name="poetry.ind.hp.com" nodeid="2" votes="1">
            <fence>
                <method name="1">
                    <device name="poetryiLO"/>
                </method>
            </fence>
        </clusternode>
    </clusternodes>
    <cman expected_votes="1" two_node="1"/>
    <fencedevices>
        <fencedevice agent="fence_ilo" hostname="drama.ind.hp.com"
login="root" name="drama-ilo" passwd="sgttest123"/>
        <fencedevice agent="fence_ilo" hostname="poetry.ind.hp.com"
login="root" name="poetry-ilo" passwd="sgttest123"/>
    </fencedevices>
    <rm>
        <failoverdomains>
            <failoverdomain name="MySQLdom" nofailback="0"
ordered="1" restricted="1">
                <failoverdomainnode name="drama.ind.hp.com"
priority="1"/>
                <failoverdomainnode name="poetry.ind.hp.com"
priority="1"/>
            </failoverdomain>
        </failoverdomains>
    </rm>

```

```

        </failoverdomain>
    </failoverdomains>
<service autostart="1" domain="MySQLdom" exclusive="0" name="MySQL_svc"
recovery="relocate">
    <lvm lv_name="lvsql" name="MySQLlv" vg_name="vgsql"/>
    <fs device="/dev/vgsql/lvsql" force_fsck="1"
force_unmount="1" fsid="53777" fstype="ext3" mountpoint="/home/mysql"
name="MySQLfs" options="rw" self_fence="1"/>
    <ip address="15.154.63.181" monitor_link="1"/>
    <mysql config_file="/home/mysql/mysqlhome/my.cnf"
listen_address="15.154.63.181" name="MySQLdb" shutdown_wait="30"/>
    </service>
    <quorumd interval="2" label="qdisk_mysql"
min_score="1" tko="5" votes="1">
    <heuristic interval="2" program="/home/mysql/checkpvtlink.sh
eth1" score="1"/>
    </quorumd>
    <totem consensus="4800" join="60" token="20000"
token_retransmits_before_loss_const="20"/>
</cluster>

```

Note that the file system is not converted to a cluster filesystem (GFS) in this example. If the file system is converted to GFS, then the `fs` line should be replaced with `clusterfs`.

Starting the RHCS MySQL Service

Enable the startup of services `cman`, `clvmd`, `qdiskd`, `scsi_reserve`, and `rgmanager` at system boot time, by executing the `chkconfig` command for each.

Following is an example of the command execution:

```

$> chkconfig cman on
$> chkconfig clvmd on
$> chkconfig qdiskd on
$> chkconfig scsi_reserve on
$> chkconfig rgmanager on

```

If the cluster is setup using SCSI-3 PR fencing with automatic reset, disable the startup of service `watchdogd` at system boot time, by executing the `chkconfig` command.

Following is an example of the command execution:

```

# chkconfig watchdogd off

```

Start the following cluster services:

```

$> service cman start
$> service clvmd start
$> service qdiskd start
$> service scsi_reserve start
$> service rgmanager start

```

Enable the watchdog controller cluster services, `watchdog_drama` on `node1` and `watchdog_poetry` on `node2`.

```

On node1 execute the command as follows:
$> clusvcadm -e watchdog_drama

```

On node2 execute the command as follows:
\$> clusvcdm -e watchdog_poetry

Enable the MySQL service, MySQL_svc, on node1 (or on node2).

On node1 (or node2) execute the following command:
clusvcdm -e MySQL_svc

Using the command *clustat* verify that the services, MySQL_svc and watchdog_drama are up on node1 and service watchdog_poetry is up on node2.

Following is an example output of the *clustat* command:

```
Cluster Status for clust @ Thu Jul 30 10:20:32 2009
Member Status: Quorate

Member Name                               ID      Status
-----
poetry.ind.hp.com                          1      Online, rgmanager
drama.ind.hp.com                           2      Online, Local, rgmanager

Service Name                               Owner (Last)      State
-----
service:MySQL_svc                         drama.ind.hp.com  started
service:watchdog_drama                    drama.ind.hp.com  started
service:watchdog_poetry                   poetry.ind.hp.com  started
```

PostgreSQL Resource Agent Migration Procedure

This section covers the details that are unique to PostgreSQL in the migration process. A step by step procedure is described for a PostgreSQL package. Any differences related to Legacy and Modular packages are described.

PostgreSQL Toolkit Migration Planning

In this section, the required RHCS resources are determined. The Serviceguard Package Configuration worksheet is filled with values from the Serviceguard cluster configuration files.

Following are the package configuration parameters of the PostgreSQL Toolkit Modular package:

<i>package_name</i>	<i>pgsqlmod</i>
<i>package_description</i>	<i>"Serviceguard Package"</i>
<i>module_name</i>	<i>sg/basic</i>
<i>module_version</i>	<i>1</i>
<i>module_name</i>	<i>sg/failover</i>
<i>module_version</i>	<i>1</i>
<i>module_name</i>	<i>sg/priority</i>
<i>module_version</i>	<i>1</i>
<i>module_name</i>	<i>sg/filesystem</i>
<i>module_version</i>	<i>1</i>
<i>module_name</i>	<i>sg/package_ip</i>
<i>module_version</i>	<i>1</i>
<i>module_name</i>	<i>sg/service</i>
<i>module_version</i>	<i>1</i>
<i>module_name</i>	<i>sg/volume_group</i>
<i>module_version</i>	<i>1</i>
<i>module_name</i>	<i>tkit/pgsql/pgsql</i>
<i>module_version</i>	<i>1</i>
<i>package_type</i>	<i>failover</i>
<i>node_name</i>	<i>*</i>
<i>auto_run</i>	<i>yes</i>
<i>node_fail_fast_enabled</i>	<i>no</i>
<i>halt_script_timeout</i>	<i>no_timeout</i>
<i>successor_halt_timeout</i>	<i>no_timeout</i>
<i>script_log_file</i>	<i>\$\$SRUN/log/\$\$SG_PACKAGE.log</i>
<i>operation_sequence</i>	<i>\$\$SGCONF/scripts/sg/volume_group.sh</i>
<i>operation_sequence</i>	<i>\$\$SGCONF/scripts/sg/filesystem.sh</i>
<i>operation_sequence</i>	<i>\$\$SGCONF/scripts/sg/package_ip.sh</i>
<i>operation_sequence</i>	<i>\$\$SGCONF/scripts/tkit/pgsql/tkit_module.sh</i>
<i>operation_sequence</i>	<i>\$\$SGCONF/scripts/sg/service.sh</i>
<i>log_level</i>	<i>5</i>
<i>failover_policy</i>	<i>configured_node</i>
<i>failback_policy</i>	<i>manual</i>
<i>priority</i>	<i>no_priority</i>
<i>concurrent_fsck_operations</i>	<i>1</i>
<i>concurrent_mount_and_umount_operations</i>	<i>1</i>
<i>fs_mount_retry_count</i>	<i>0</i>
<i>fs_umount_retry_count</i>	<i>1</i>
<i>fs_name</i>	<i>/dev/vgsql/lvsq1</i>
<i>fs_directory</i>	<i>/home/postgres</i>
<i>fs_type</i>	<i>ext3</i>
<i>fs_mount_opt</i>	<i>"-o rw"</i>
<i>#fs_umount_opt</i>	
<i>#fs_fsck_opt</i>	

```

ip_subnet                15.154.62.0
#ip_subnet_node
ip_address               15.154.63.181
service_name            pgsql_service
service_cmd              "$SGCONF/scripts/tkit/pgsql/tkit_module.sh pgsql_monitor"
service_restart         none
service_fail_fast_enabled no
service_halt_timeout    300
vgchange_cmd            "vgchange -a y"
vg                      vgsql
TKIT_DIR                /usr/local/cmcluster/postgresqltoolkitmod
DATA_DIRECTORY          /home/postgres/postgreshome
LOG_FILE                /home/postgres/postgreshome/pgsql.log
PID_FILE                ${DATA_DIRECTORY}/postmaster.pid
SERVER_PATH             /usr/bin/postmaster
MAINTENANCE_FLAG        yes
MONITOR_INTERVAL        5
RETRY_TIMES             2

```

Following are the package configuration parameters of the PostgreSQL Toolkit Legacy package:

```

PACKAGE_NAME            postgreleg
PACKAGE_TYPE            FAILOVER
NODE_NAME               *
AUTO_RUN                YES
NODE_FAIL_FAST_ENABLED NO
RUN_SCRIPT              /usr/local/cmcluster/pkg/postgresleg/postgresleg.sh
HALT_SCRIPT             /usr/local/cmcluster/pkg/postgresleg/postgresleg.sh
RUN_SCRIPT_TIMEOUT     NO_TIMEOUT
HALT_SCRIPT_TIMEOUT    NO_TIMEOUT
SUCCESSOR_HALT_TIMEOUT NO_TIMEOUT
SCRIPT_LOG_FILE        /usr/local/cmcluster/run/log/pgsqlmod.log
FAILOVER_POLICY        CONFIGURED_NODE
FAILBACK_POLICY        MANUAL
PRIORITY               NO_PRIORITY

SERVICE_NAME          postgresvc
SERVICE_FAIL_FAST_ENABLED no
SERVICE_HALT_TIMEOUT 300

```

Following are the package control script parameters of the PostgreSQL Toolkit Legacy package:

```

VGCHANGE="vgchange -a y"
VG[0]="vgsql"

LV[0]="/dev/vgsql/lvsq1"; FS[0]="/home/postgres"; FS_TYPE[0]="ext3";
FS_MOUNT_OPT[0]="-o rw"
FS_UMOUNT_OPT[0]=""; FS_FSCK_OPT[0]=" "
FS_UMOUNT_COUNT=1
FS_MOUNT_RETRY_COUNT=0
CONCURRENT_FSCK_OPERATIONS=1

IP[0]="15.154.63.181"
SUBNET[0]="15.154.62.0"

SERVICE_NAME[0]="postgresvc"
SERVICE_CMD[0]="/usr/local/cmcluster/postgresqltoolkitleg/toolkit.sh monitor"
SERVICE_RESTART[0]=" "

```

```

# Setting the log file
log_file=${SG_SCRIPT_LOG_FILE:-$0.log}

# START OF CUSTOMER DEFINED FUNCTIONS

# This function is a place holder for customer define functions.
# You should define all actions you want to happen here, before the service is
# started. You can create as many functions as you need.
#
function customer_defined_run_cmds
{
# ADD customer defined run commands.
: # do nothing instruction, because a function must contain some command.
/usr/local/cmcluster/postgresqltoolkitleg/toolkit.sh start
test_return 51
}

# This function is a place holder for customer define functions.
# You should define all actions you want to happen here, after the service is
# halted.
#
function customer_defined_halt_cmds
{
# ADD customer defined halt commands.
: # do nothing instruction, because a function must contain some command.
/usr/local/cmcluster/postgresqltoolkitleg/toolkit.sh stop
test_return 52
}

# END OF CUSTOMER DEFINED FUNCTIONS

```

The table below illustrates how this information is used to populate the Serviceguard Package Configuration worksheet.

SGLX package configuration parameters

RHCS service and resource configuration parameters

<p>Package Name _____ pgsqmod_____</p> <p>Node_name _____ drama_____</p> <p>Node_name _____ poetry_____</p> <p>Failover_policy _____ configured_node_____</p> <p>Failback_policy _____ MANUAL_____</p> <p>Auto_run _____ YES_____</p> <p>Node_fast_fail _____ NO_____</p> <p>Package dependency</p> <p>dependency_name _____ NOT_USED_____</p> <p>dependency_condition _____ NOT_USED_____</p> <p>dependency_location _____ NOT_USED_____</p>	<p>Service configuration</p> <p>Service name _____ Postgressvc_____</p> <p>Failover Domain _____ Postgresdom_____</p> <p>Service Resources List</p> <p>_____ Postgreslvm_____</p> <p>_____ Postgresfs_____</p> <p>_____ Postgresdb_____</p> <p>Recovery policy (Restart, relocate, disable)</p> <p>_____ Relocate_____</p> <p>Auto start service _____ YES_____</p> <p>Run exclusive _____ NO_____</p> <p>Hard recovery _____ NO_____</p> <p>Depend _____ NOT_USED_____</p> <p>Depend mode _____ NOT_USED_____</p>
--	---

<p>Logical Volumes and File Systems</p> <p>fs_name ___/dev/vgsql/lvsq1___ fs_directory ___/home/postgres_____ fs_mount_opt ___ -o rw_____ fs_umount_opt ___ ""_____ fs_fsck_opt ___ ""_____ fs_type ___ext3_____</p> <p>Network information</p> <p>IP ___15.154.63.181___ Subnet ___ 15.154.62.0_____ Monitored_Subnet ___NOT USED_____ Monitored_subnet_access ___NOT USED___</p> <p>LVM volume groups</p> <p>VG ___vgsql_____</p> <p>Service configuration</p> <p>service_name ___pgsql_service___ service_cmd ___"\$SGCONF/scripts/tkit/postgresql/tkit _module.sh pgsql_monitor"___ service_restart ___none___ service_fail_fast_enabled ___no___ service_halt_timeout ___300___</p> <p>Non-Serviceguard environment variable</p> <p>NOT USED</p> <p>External script</p> <p>external_script ___NOT USED___ priority ___no_priority___</p> <p>Priority _____ #Weight Name _____ #Weight Value _____</p>	<p>Managed Resources/Failover Domain configuration</p> <p>Domain name ___Postgresdom_____ Member node ___drama_____ Member node ___poetry_____ Restrict failover ___Yes_____ Prioritized list ___drama, poetry_____</p> <p>Resources/File system</p> <p>Name ___Postgresfs_____ File system type ___ext3_____ Mount point ___/home/postgres_____ Device ___/dev/vgsql/lvsq1___ Options ___rw_____ File System ID _____ Force unmount ___YES_____ Reboot if unmount fails ___NO_____ Check file system before mounting ___YES_____</p> <p>Resources/IP address</p> <p>IP address ___15.154.63.181_____ Monitor Link ___YES_____</p> <p>Resource/LVM</p> <p>Name ___Postgreslvm_____ Volume group name ___vgsql_____ Logical volume name ___lvsq1_____</p> <p>Script Resource</p> <p>NOT USED</p>
--	--

Setup the LVM, Filesystem, IP, and PostgreSQL resources as required for the migration of the Serviceguard package pgsq1mod. For information on the resource configuration details, see the Package Configuration worksheet prepared during the planning phase.

Setting up the RHCS PostgreSQL Service

The next step in the migration is setting up the RHCS PostgreSQL cluster service, Postgressvc. See the Package Configuration worksheet prepared during the planning phase for information on the service, its failover domain, and child resources.

After the configuration of the failover domains, the resources, and the service for PostgreSQL, verify that the following required XML definitions are present in the `/etc/cluster/cluster.conf` file.

Following is an example of the `/etc/cluster/cluster.conf` with PostgreSQL service setup. SCSI-3 PR fencing with automatic reset is also set up.

```
<?xml version="1.0"?>
<cluster alias="clust" config_version="76" name="clust">
  <fence_daemon clean_start="0" post_fail_delay="0" post_join_delay="3"/>
  <clusternodes>
    <clusternode name="drama.ind.hp.com" nodeid="1" votes="1">
      <fence>
        <method name="1">
          <device name="SCSI_fence"
node="poetry.ind.hp.com"/>
        </method>
      </fence/>
    </clusternode>
    <clusternode name="poetry.ind.hp.com" nodeid="2" votes="1">
      <fence>
        <method name="1">
          <device name="SCSI_fence"
node="poetry.ind.hp.com"/>
        </method>
      </fence/>
    </clusternode>
  </clusternodes>
  <cman expected_votes="1" two_node="1"/>
  <fencedevices>
    <fencedevice agent="fence_scsi" name="SCSI_fence"/>
  </fencedevices>
  <rm>
    <failoverdomains>
      <failoverdomain name="Postgresdom" nofailback="0"
ordered="1" restricted="1">
        <failoverdomainnode name="drama.ind.hp.com"
priority="1"/>
        <failoverdomainnode name="poetry.ind.hp.com"
priority="1"/>
      </failoverdomain>
      <failoverdomain name="dramadom" nofailback="0"
ordered="0" restricted="1">
        <failoverdomainnode name="drama.ind.hp.com"
priority="1"/>
      </failoverdomain>
      <failoverdomain name="poetrydom" nofailback="0"
ordered="0" restricted="1">
        <failoverdomainnode name="poetry.ind.hp.com"
```

```

priority="1"/>
        </failoverdomain>
    </failoverdomains>
    <service autostart="1" domain="Postgresdom" exclusive="0"
name="Postgressvc" recovery="relocate">
        <fs device="/dev/vgsql/lvsq1" force_fsck="1"
force_unmount="1" fsid="45727" fstype="ext3" mountpoint="/home/postgres"
name="Postgresfs" options="rw" self_fence="1"/>
        <ip address="15.154.63.181" monitor_link="1"/>
        <postgres-8
config_file="/home/postgres/postgreshome/postgresql.conf" name="Postgresdb"
postmaster_options="-D /home/postgres/postgreshome" postmaster_user="postgres"
shutdown_wait="5"/>
    </service>
    <service autostart="0" domain="dramadom" exclusive="0"
max_restarts="0" name="watchdog_drama" recovery="restart"
restart_expire_time="0">
        <script ref="watchdog_script"/>
    </service>
    <service autostart="0" domain="poetrydom" exclusive="0"
max_restarts="0" name="watchdog_poetry" recovery="restart"
restart_expire_time="0">
        <script ref="watchdog_script"/>
    </service>
</rm>
<quorumd interval="2" label="qdisk_postgresql" min_score="1" tko="5"
votes="1">
        <heuristic interval="2"
program="/home/postgresql/checkpvtlink.sh eth1" score="1"/>
    </quorumd>
    <totem consensus="4800" join="60" token="20000"
token_retransmits_before_loss_const="20"/>
</cluster>

```

Note that the LVM line is not used in the above example as the volumes have been converted to CLVM. Additional failover domains are required if SCSI-3 PR fencing with automatic reset is setup.

Following is another example of the RHCS /etc/cluster/cluster.conf with PostgreSQL service setup. In this example, iLO fencing is used.

```

<?xml version="1.0"?>
<cluster alias="clust" config_version="76" name="clust">
    <fence_daemon clean_start="0" post_fail_delay="0" post_join_delay="3"/>
    <clusternodes>
        <clusternode name="drama.ind.hp.com" nodeid="1" votes="1">
            <fence>
                <method name="1">
                    <device name="dramaiLO"/>
                </method>
            </fence>
        </clusternode>
        <clusternode name="poetry.ind.hp.com" nodeid="2" votes="1">
            <fence>
                <method name="1">
                    <device name="poetryiLO"/>
                </method>
            </fence>
        </clusternode>
    </clusternodes>
    <cman expected_votes="1" two_node="1"/>

```

```

    <fencedevices>
        <fencedevice agent="fence_ilo" hostname="drama.ind.hp.com"
login="root" name="drama-ilo" passwd="sgtest123"/>
        <fencedevice agent="fence_ilo" hostname="poetry.ind.hp.com"
login="root" name="poetry-ilo" passwd="sgtest123"/>
    </fencedevices>
</rm>
    <failoverdomains>
        <failoverdomain name="Postgresdom" nofailback="0"
ordered="1" restricted="1">
            <failoverdomainnode name="drama.ind.hp.com"
priority="1"/>
            <failoverdomainnode name="poetry.ind.hp.com"
priority="1"/>
        </failoverdomain>
    </failoverdomains>

    <service autostart="1" domain="Postgresdom" exclusive="0"
name="Postgressvc" recovery="relocate">
        <lvm lv_name="lvsql" name="Postgreslvm"
vg_name="vgsq1"/>
        <fs device="/dev/vgsq1/lvsql" force_fsck="1"
force_unmount="1" fsid="45727" fstype="ext3" mountpoint="/home/postgres"
name="Postgresfs" options="rw" self_fence="1"/>
        <ip address="15.154.63.181" monitor_link="1"/>
        <postgres-8
config_file="/home/postgres/postgreshome/postgresql.conf" name="Postgresdb"
postmaster_user="postgres" shutdown_wait="5"/>
    </service>

</rm>
    <quorumd interval="2" label="qdisk_postgresql" min_score="1" tko="5"
votes="1">
        <heuristic interval="2"
program="/home/postgresql/checkpvtlink.sh eth1" score="1"/>
    </quorumd>
    <totem consensus="4800" join="60" token="20000"
token_retransmits_before_loss_const="20"/>
</cluster>

```

Note that the file system is not converted to a cluster filesystem (like GFS) in this example. If the file system is converted to GFS, then the fs line should be replaced with clusterfs.

Editing the new generated configuration file

The new configuration file has to be edited and copied to the same location on all the nodes of the cluster. The default log file location, generated PID file location, and generated listen_address IPs can be changed by editing the new generated configuration file.

Starting the RHCS PostgreSQL Service

Enable the startup of services cman, clvmd, qdiskd, scsi_reserve, and rgmanager at system boot time, by executing the chkconfig command.

Following is an example of the command execution:

```
$> chkconfig cman on
$> chkconfig clvmd on
$> chkconfig qdiskd on
$> chkconfig scsi_reserve on
$> chkconfig rgmanager on
```

If the cluster is setup using SCSI-3 PR fencing with automatic reset, disable the startup of service watchdogd at system boot time, by executing the `chkconfig` command.

Following is an example of the command execution:

```
# chkconfig watchdogd off
```

Start the following cluster services:

```
$> service cman start
$> service clvmd start
$> service qdiskd start
$> service scsi_reserve start
$> service rgmanager start
```

Enable the watchdog controller cluster services, `watchdog_drama` on node1 and `watchdog_poetry` on node2.

On node1 execute the command as follows:

```
$> clusvcadm -e watchdog_drama
```

On node2 execute the command as follows:

```
$> clusvcadm -e watchdog_poetry
```

Enable the PostgreSQL service, `Postgressvc`, on node1 (or on node2).

On node1 (or node2) execute the following command:

```
# clusvcadm -e Postgressvc
```

Using the command `clustat` verify that the services, `Postgressvc` and `watchdog_drama` are up on node1 and the service `watchdog_poetry` is up on node2.

Following is an example output of the `clustat` command:

```
Cluster Status for clust @ Thu Jul 30 10:39:20 2009
Member Status: Quorate

Member Name                ID      Status
-----
drama.ind.hp.com           1      Online, rgmanager
poetry.ind.hp.com          2      Online, Local, rgmanager

Service Name                Owner (Last)                State
-----
service:Postgressvc         drama.ind.hp.com            started
```

<i>service:watchdog_drama</i>	<i>drama.ind.hp.com</i>	<i>started</i>
<i>service:watchdog_poetry</i>	<i>poetry.ind.hp.com</i>	<i>started</i>

Terms

CDB

Cluster Database

CLVM

Clustered Logical Volume Manager

CMAN

The Cluster Manager is responsible managing the cluster quorum and keeps track of the cluster membership

GFS

Red Hat Global File System is a cluster file system that allows a cluster of nodes to simultaneously access a block device that is shared among nodes

HA

High Availability

Legacy package

The only package configuration method in HP Serviceguard for Linux A.11.16 and earlier

LVM

Logical volume management facilities on Linux

LVM2

A new user space tool set that provides logical volume management facilities on Linux

Modular package

Single package configuration file, introduced in HP Serviceguard A.11.18. The package configuration information is included in only the package configuration (ASCII) file, whereas in versions prior to A.11.18, package configuration information is in both the package ASCII file and the package control script.

RA

Resource Agent

RHCS

The Red Hat Cluster Suite is an integrated set of software components that can be deployed to suit the needs for performance, high-availability, load balancing, scalability, file sharing and economy.

RHEL5

Red Hat Enterprise Linux 5

SGLX

HP Serviceguard for Linux

XML

The Extensible Markup Language is a general-purpose specification for creating custom markup languages.

For More Information

- HP Serviceguard for Linux product documentation at <http://docs.hp.com>
- Red Hat Cluster Suite for RHEL5, configuration guide at <http://www.redhat.com/docs/manuals/csgfs>
- HP Serviceguard for Linux certification matrix showing servers, storage, and software versions supported: <http://www.hp.com/info/sglx>

Appendix

Serviceguard Package Configuration worksheet with MySQL and PostgreSQL Toolkit parameters

SGLX package configuration

Package Name _____
Node_name _____
Node_name _____
Failover_policy _____
Failback_policy _____
Auto_run _____
Node_fast_fail _____

Package dependency

dependency_name _____
dependency_condition _____
dependency_location _____

Logical Volumes and File Systems

fs_name _____
fs_directory _____
fs_mount_opt _____
fs_umount_opt _____
fs_fsck_opt _____
fs_type _____

Network information

IP _____
Subnet _____
Monitored_Subnet _____
Monitored_subnet_access _____

RHCS service, common resources, and failover domain configuration

Service configuration

Service name _____
Failover Domain _____
Service Resources List _____

Recovery policy _____
(Restart, relocate, disable)

Auto start service _____
Run exclusive _____
Hard recovery _____
Depend _____
Depend mode _____

Managed Resources/Failover Domain configuration

Domain name _____
Member node _____
Member node _____
Restrict failover _____
Prioritized list _____

Resources/File system

Name _____
File system type _____
Mount point _____
Device _____
Options _____
File System ID _____
Force unmount _____
Reboot if unmount fails _____
Check file system before mounting _____

Resources/IP address

IP address _____
Monitor Link _____

LVM volume groups

VG _____

Service configuration

Service_Name _____

Command _____

Restart _____

Fast_fail_enabled _____

Resources/LVM

Name _____

Volume group name _____

Logical volume name _____

Script Resource

Name of script _____

Non-Serviceguard environment variable

External script

External script file _____

Priority _____

#Weight Name _____

#Weight Value _____

#STORAGE_GROUP _____

#USER_NAME _____

#USER_HOST _____

#USER_ROLE _____

MySQL Toolkit Parameters

CONFIGURATION_FILE_PATH _____

DATA_DIRECTORY _____

TIME_OUT _____

PostgreSQL Toolkit Parameters

DATA_DIRECTORY _____

Resources/ MySQL

Config File _____

Edit the my.cnf file and set the datadir

Listen Address _____

Shutdown Wait (seconds) _____

Resources/ postgres-8

Config File _____

Postmaster Options _____

Postmaster User _____

Shutdown Wait (seconds) _____

LOG_FILE	_____	Edit the postgresQL configuration file
PID_FILE	_____	Edit the postgresQL configuration file
SERVER_PATH	_____	Edit the PostgreSQL RA script

© Copyright 2009 Hewlett-Packard Development Company, L.P. The information contained herein is subject to change without notice. The only warranties for HP products and services are set forth in the express warranty statements accompanying such products and services. Nothing herein should be construed as constituting an additional warranty. HP shall not be liable for technical or editorial errors or omissions contained herein.

Linux is a U.S. registered trademark of Linus Torvalds. Microsoft and Windows are U.S. registered trademarks of Microsoft Corporation. UNIX is a registered trademark of The Open Group.

5900-0333, September 2009

