LifeKeeper® for Linux v5.1

MySQL Recovery Kit Administration Guide

October 2010

SteelEye and LifeKeeper are registered trademarks.

It is the policy of SIOS Technology Corp. (previously known as SteelEye Technology, Inc.) to improve products as new technology, components, software, and firmware become available. SIOS Technology Corp., therefore, reserves the right to change specifications without prior notice.

To maintain the quality of our publications, we need your comments on the accuracy, clarity, organization, and value of this book.

Address correspondence to:
ip@us.sios.com

Table of Contents

MySQL Recovery Kit Administration Guide

Introduction

The LifeKeeper® for Linux MySQL Recovery Kit provides an easy way to add LifeKeeper fault-resilient protection for MySQL resources and databases. This enables a failure on the primary database server to be recovered on a designated backup server without significant lost time or human intervention.

Document Contents

This guide explains the following topics:

- **LifeKeeper Documentation.** A list of all the LifeKeeper for Linux documentation and where the information is available.
- **Requirements.** Before you can install and set up the recovery software, your server must meet certain hardware and software requirements. You should refer to the *LifeKeeper for Linux Planning and Installation Guide* for specific instructions on how to install or remove the LifeKeeper MySQL Recovery Kit.
- **Configuring Your Recovery Kit.** To ensure that your LifeKeeper configuration provides the protection and flexibility you require, you need to be aware of the configuration rules. To appropriately plan your configuration, you must understand your network configuration, interface selection, user system setup, hierarchy options and the MySQL configuration tasks. In addition to planning your configuration, this section also includes configuration examples and the specific tasks required to configure your Recovery Kit.
- **Troubleshooting.** This section provides a list of informational and error messages along with recommended solutions.

LifeKeeper Documentation

The following LifeKeeper product documentation is available from SIOS Technology Corp.:

- *LifeKeeper for Linux Release Notes*
- *LifeKeeper for Linux Online Product Manual* (available from the Help menu within the LifeKeeper GUI)
- *LifeKeeper for Linux Planning and Installation Guide*

This documentation, along with documentation associated with optional LifeKeeper Recovery Kits, is available on the SIOS Technology Corp. website at:

http://us.sios.com/support

The following is a list of reference documents associated with the MySQL application and the MySQL Recovery Kit:

- *MySQL Reference Manual* available at: http://www.mysql.com/documentation/index.html
- *MySQL*, Paul DuDois, New Riders Publishing, 2000
- *My SQL & mSQL*, Randy Jay Yarger, George Reese, and Tim King, O'Reilly & Associates, Inc. 1999
- *SQL in a Nutshell*, Kevin Kline with Daniel Kline, Ph.D., O'Reilly & Associates, Inc. 2000

Requirements

Before attempting to install or remove the MySQL Recovery Kit, you must understand the hardware and software requirements for the package and the installation and removal procedures.

Kit Hardware and Software Requirements

Before installing and configuring the LifeKeeper MySQL Recovery Kit, be sure that your configuration meets the following requirements:

- **Servers.** The Recovery Kit requires two or more LifeKeeper supported computers configured in accordance with the requirements described in *LifeKeeper Online Product Manual Online* and the *LifeKeeper Release Notes*, which are shipped with the product media.
- **LifeKeeper software.** You must install the same version of LifeKeeper software and any patches on each server. Please refer to the *LifeKeeper Release Notes* and *LifeKeeper Online Product Manual* for specific LifeKeeper requirements.
- **LifeKeeper IP Recovery Kit.** This kit is required if remote clients will be accessing the MySQL database. You must have the same version of this Recovery Kit on *each* server.
- **IP network interface.** Each server requires at least one Ethernet TCP/IP-supported network interface. In order for IP switchover to work properly, user systems connected to the local network should conform to standard TCP/IP specifications.

  Note: Even though each server requires only a single network interface, you should use multiple interfaces for a number of reasons: heterogeneous media requirements, throughput requirements, elimination of single points of failure, network segmentation, and so forth.
- **TCP/IP software.** Each server also requires the TCP/IP software.
- **MySQL software.** Each server must have the MySQL software installed and configured prior to configuring LifeKeeper and the LifeKeeper MySQL Recovery Kit. The same version should be installed on each server. Consult the *LifeKeeper Release Notes* or your sales representative for the latest release compatibility and ordering information.

You should refer to the *LifeKeeper for Linux Planning and Installation Guide* for specific instructions on how to install or remove the LifeKeeper MySQL Recovery Kit.

Configuring MySQL with LifeKeeper

This section contains definitions and examples of typical LifeKeeper MySQL configurations and information you should consider before you start to configure MySQL.

Please refer to *LifeKeeper Online Product Manual* for instructions on configuring your LifeKeeper Core resource hierarchies.

Specific Configuration Considerations for MySQL

Below are some specific considerations you need to think about concerning your LifeKeeper MySQL environment.

To operate MySQL database services on the primary and backup servers, file systems and disk partitions must be accessible from each server. Before you can begin configuring the MySQL Recovery Kit, be sure you have completed the following preliminary steps and have tested/run the databases on each server. In the instructions below, the user "mysql" refers to the operating system user that will start the MySQL server.

1. Install the MySQL server and client components on all servers. Be sure that all of the servers are running the same version of the MySQL client and server components. The MySQL executables can be located on a local or shared drive.
2. If **mysqld** is running on any of the servers on the socket and/or port where you wish to run the LifeKeeper protected MySQL database server, stop each MySQL server using the **mysqladmin** command.
3. Move the contents of the MySQL data directory to a shared location. By default, the MySQL data directory is installed on a local drive. This location depends on the distribution mechanism. The binary RPM installs the data directory at */var/lib/mysql*. (Be sure that only the contents are moved and the directory remains intact. This allows the MySQL database server to write logs in this directory, if necessary. Make sure that the "mysql" user described in step 4 has permissions to write the logs to this location.)
4. If the installation process did not create the Linux user "mysql", create this user. For security reasons, the MySQL server should not be run as "root." (Refer to the MySQL Administration Guide for a full discussion of the security issues.) Make sure that "mysql" is the only user with read/write permissions in the database directories. The "mysql" user and group should be created on all servers. The user ID and group ID must be the same on all servers.
5. **IMPORTANT:** A server started by */etc/rc.d/init.d/mysql* cannot be under LifeKeeper protection. In addition, the server can not use the same port number or socket as a server under LifeKeeper protection.
6. It is recommended that the socket be written to the data directory on the shared disk. If the socket will be written to a local disk, make sure the path exists on all LifeKeeper servers where your hierarchy will exist. Make sure that the user "mysql" has permissions to write the socket to this location.

7. Start the MySQL server using the mysql daemon startup command:

   ```
   <start command> --user=mysql --socket=<socket> --port<port number>
   --datadir=<path to the data directory> --log &
   ```

   The start command for mysql versions 3.x is **safe_mysqld**, and the command for version 4.x is **mysqld_safe**.

8. Create a MySQL database user named "mysql". Give this user a password and grant the user "shutdown" permissions. This only has to be done on one server. (Refer to the MySQL Administration Guide for details on creating users and granting permissions).

9. Copy the sample *my.cnf* configuration file to the desired location (*/etc* or /<*datadir*>). This file contains options for the database server and for client programs.

   The file can be located in either the MySQL data directory or the */etc* directory. The */etc/my.cnf* file contains global options. Place the *my.cnf* file in */etc* if only one database will run on the machine at any given time (i.e. an Active/Standby configuration). If the file is located in */etc*, you must copy it to each LifeKeeper backup server. The *my.cnf* file in the data directory should contain server-specific options. For multiple servers and Active/Active configurations, this file must be stored in the data directory for each resource instance.

   Note: The my.cnf file should not exist in both the /etc and /<datadir> locations if both copies will contain server specific options. If a my.cnf file containing server specific options is located in /etc along with a protected my.cnf file installed in the /<datadir> potential conflicts may result. Refer to the MySQL documentation on configuring global settings and server specific options.

   Add or edit the following entries:

   a. In the "client" section of the file, specify the user and the password that should be used for connections.

      ```
      [client]
      user          =clientuser
      password      =password
      .
      .
      .
      ```

   b. In the "mysqld" section of the file, specify the socket and port that should be used for connections, as well as the pid-file location for the mysqld process. The user variable should specify the operating system user that will start the mysqld process.

      ```
      [mysqld]
      socket        =/home1/test/mysql/mysql.sock
      port          =3307
      pid-file      =/home1/test/mysql/mysqld.pid
      user          =osuser
      .
      .
      .
      ```

   Note: Make sure this file is properly protected and owned by the user "mysql."

   Note: Once the MySQL hierarchy is created, if you need to change any of the information in the *my.cnf* file, you must stop the mysql server instance by taking the hierarchy out-of-service (i.e. the OSU state) before making changes.

Considerations for a Multiple Database Server Environment

Following are some configuration considerations if you have multiple MySQL database servers and databases:

- If running active/active or multiple servers, do not mount a shared file system as */var/lib/mysql*. This causes unexpected shutdown of MySQL servers by the mysql startup command (**safe_mysqld** or **mysqld_safe**).
- The *my.cnf* file **must** be stored in the data directory for each of the active/active or multiple servers.
- Additional port numbers for MySQL must be specified in the */etc/services file*.
- Each MySQL database server must be configured to run on a different port and access a different socket file. These configuration options are specified in the *my.cnf* file in the data directory.
- Each server must be configured to access data from a different shared location (i.e. each server must use a different data directory).

Client Configuration Considerations

Following are some configuration considerations for MySQL database clients:

- If clients will connect from remote hosts, create an IP address under LifeKeeper to be used for client connections.
- Clients must be configured to connect to the database server through a LifeKeeper-protected IP address.
- If the clients will connect through a domain name instead, create an entry in each client's *hosts* file for the protected IP address, or configure the name in DNS. Test the protected IP address by pinging it from all clients and all LifeKeeper servers in the cluster.
- Although each user can have a *.my.cnf* file in the home directory of their machine, LifeKeeper only uses the *my.cnf* file located in the */etc* directory or the data directory. The *my.cnf* file stores the client connection information (i.e. the port, socket identification, user, and password).

Configuration Examples

The examples in this section show how MySQL database instances can be configured. Each diagram shows the relationship between the type of configuration and the MySQL parameters. Each configuration also adheres to the configuration rules and requirements described in this document that ensure compatibility between the MySQL configuration and the LifeKeeper software.

This section describes the configuration requirements and then provides these configuration examples:

- Active/Standby
- Active/Active

The examples in this section are only a sample of the configurations you can establish, but understanding these configurations and adhering to the configuration rules will help you define and set up workable solutions for your computing environment.

Configuration Requirements

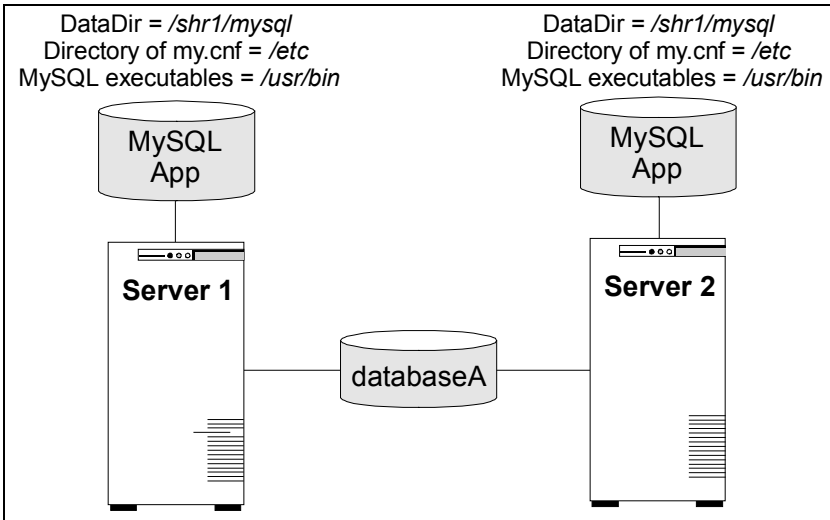
Each of the examples involves one or two databases: **databaseA** and **databaseB**. The Database Tag names are arbitrary names that describe these databases to LifeKeeper. The word on and the system identifier that follows provide clarification, but are not required. The default tag name suggested by LifeKeeper is mysql. To understand the configuration examples, keep these configuration requirements in mind:

- **LifeKeeper hierarchy**. When performing LifeKeeper administration, the primary hierarchy refers to the hierarchy being built on the server you are administering. For the configuration diagrams, the information entered in the first administration screen is from the perspective of Server 1. When a second screen is shown, it refers to the hierarchy being built while administering the second server. In the configuration examples, the second server is Server 2.
- **Shared disk locked by one server**. When you use LifeKeeper, one server reserves shared storage resources that are under LifeKeeper protection for use. This is done using SCSI reservations. If the shared device is a disk array, an entire LUN is reserved; if a shared device is a disk, then the entire disk is reserved. This prevents inadvertent corruption of the data by other servers in the cluster. When a server fails, the highest priority backup server breaks the old reservation and establishes its own reservation, locking out all other servers.
- **Database on shared disk**. In order for the LifeKeeper MySQL Recovery Kit to function properly, the database must always be on a shared disk. The database must be on a file system. The file system must be mountable from both the primary and backup servers.

Active/Standby Configuration

This section provides an example of an active/standby configuration. In this configuration, Server 1 is considered active because it has exclusive access to the database. Server 2 does other processing. If Server 1 fails, Server 2 gains access to the database, and LifeKeeper reestablishes the database operations.

Figure 1. Active/Standby Configuration, Example 1



Configuration Notes:

- Both servers use the MySQL data directory (which includes the database (databaseA)) on a shared disk.
- The path to the MySQL data directory is the same on both servers.
- The *my.cnf* configuration file is located on a local disk in */etc*.
- The MySQL executables are located on a local drive on each server in */usr/bin*.
- Server 2 cannot access files and directories on the shared disk while Server 1 is active.

Creating a resource hierarchy on Server 1:

Server:	Server1
Directory of my.cnf File Location:	/etc
Directory of MySQL Executables Location:	/usr/bin
Database Tag	mysql-on-server1

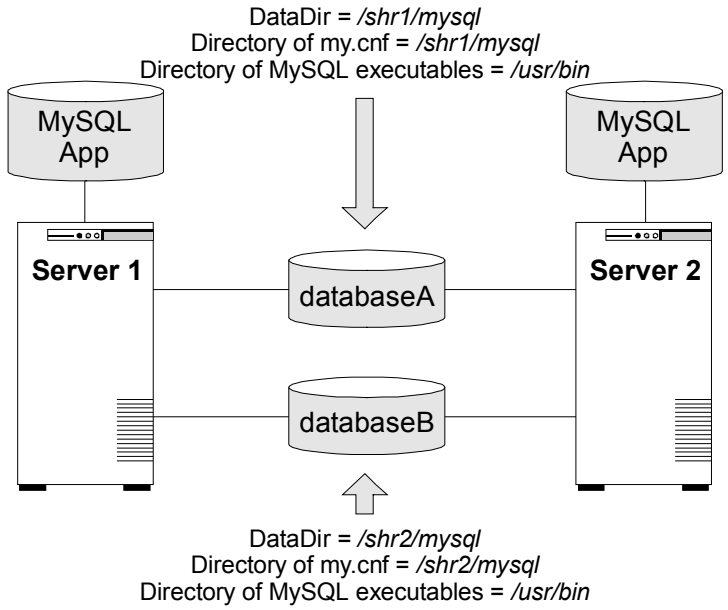
Extending a resource hierarchy to Server 2:

Template Server:	Server1
Tag to Extend	mysql-on-server1
Target Server	Server2
Target Priority:	10
Directory of my.cnf File Location:	/etc
Directory of MySQL Executables Location:	/usr/bin
Database Tag	mysql-on-server2

Active/Active Configurations

An active/active configuration consists of two or more servers actively running a different database instance, with each serving as a backup for each other. The databases **must** be on different shared physical disks.

Figure 2. Active/Active Configuration, Example 1



Configuration Notes:

- Each server uses a different MySQL data directory (which includes the databases (database A and database B) on different shared disks.
- The path to the MySQL data directory is different for each instance defined on the server.
- The *my.cnf* configuration file for each of the databases is located on the shared drive in the data directory for the database. The port and socket definition must be different in each *my.cnf*.
- The MySQL executables are located on a local drive on each server in */usr/bin*.
- Initially, Server 1 runs databaseA and Server 2 runs databaseB. In a switchover situation, one server can run both databases.

Creating the first resource hierarchy on Server 1:

Server:	Server1
Directory of my.cnf File Location:	/shr1/mysql
Directory of my MySQL Executables Location:	/usr/bin
Database Tag:	mysql-shared.example.instance1

Extending the first resource hierarchy to Server 2:

Template Server:	Server1
Tag to Extend:	mysql-shared.example.instance1
Target Server:	Server2
Target Priority:	10
Directory of my.cnf File Location:	/shr1/mysql
Directory of my MySQL Executables Location:	/usr/bin
Database Tag:	mysql-shared.example.instance1

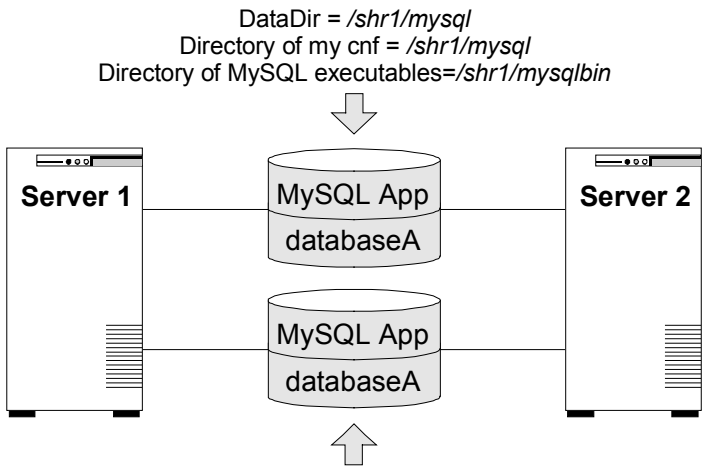
Creating the second resource hierarchy on Server 2:

Server:	Server2
Directory of my.cnf File Location:	/shr2/mysql
Directory of MySQL Executables Location:	/usr/bin
Database Tag:	mysql-shared.example.instance2

Extending the second resource hierarchy to Server 1:

Template Server:	Server2
Tag to Extend:	mysql-shared.example.instance2
Target Server:	Server1
Target Priority:	10
Directory of my.cnf File Location:	/shr2/mysql
Directory of MySQL Executables Location:	/usr/bin
Database Tag:	mysql-shared.example.instance2

Figure 3. Active/Active Configuration, Example 2



Configuration Notes:

- Each server uses a different MySQL data directory (which includes the databases (database A and database B) on different shared disks.
- The path to the MySQL data directory is different for each instance defined on the server.
- The *my.cnf* configuration file for each of the databases is located on the shared drive in the data directory for the database. The port and socket definition must be different in each *my.cnf*.
- There is a copy of the MySQL executables on each of the shared disks that contains the data directories.
- Initially, Server 1 runs databaseA and Server 2 runs databaseB. In a switchover situation, one server can run both databases.

Creating the first resource hierarchy on Server 1:

Server:	Server1
Directory of my.cnf File Location:	/shr1/mysql
Directory of MySQL Executables Location:	/shr1/mysqlbin
Database Tag:	mysql-shared.example.instance1

Extending the first resource hierarchy to Server 2:

Template Server:	Server1
Tag to Extend:	mysql-shared.example.instance1
Target Server:	Server2
Target Priority:	10
Directory of my.cnf File Location:	/shr1/mysql
Directory of MySQL Executables Location:	/shr1/mysqlbin
Database Tag:	mysql-shared.example.instance1

Creating the second resource hierarchy on Server 2:

Server:	Server2
Directory of my.cnf File Location:	/shr2/mysql
Directory of MySQL Executables Location:	/shr2/mysqlbin
Database Tag:	mysql-shared.example.instance2

Extending the second resource hierarchy to Server 1:

Template Server:	Server2
Tag to Extend:	mysql-shared.example.instance2
Target Server:	Server1
Target Priority:	10
Directory of my.cnf File Location:	/shr2/mysql
Directory of MySQL Executables Location:	/shr2/mysqlbin
Database Tag:	mysql-shared.example.instance2

LifeKeeper Configuration Tasks

You can perform the following configuration tasks from the LifeKeeper GUI. The following four tasks are described in this guide, as they are unique to a MySQL resource instance, and different for each Recovery Kit.

- **Create a Resource Hierarchy.** Creates an application resource hierarchy in your LifeKeeper cluster.
- **Delete a Resource Hierarchy.** Deletes a resource hierarchy from all servers in your LifeKeeper cluster.
- **Extend a Resource Hierarchy.** Extends a resource hierarchy from the primary server to a backup server.
- **Unextend a Resource Hierarchy.** Unextends (removes) a resource hierarchy from a single server in the LifeKeeper cluster.

The following tasks are described in the GUI Administration section within the *LifeKeeper Online Product Manual*, because they are common tasks with steps that are identical across all Recovery Kits.

- **Create a Resource Dependency.** Creates a parent/child dependency between an existing resource hierarchy and another resource instance and propagates the dependency changes to all applicable servers in the cluster.
- **Delete a Resource Dependency.** Deletes a resource dependency and propagates the dependency changes to all applicable servers in the cluster.
- **In Service.** Brings a resource hierarchy into service on a specific server.
- **Out of Service.** Takes a resource hierarchy out of service on a specific server.
- **View/Edit Properties.** View or edit the properties of a resource hierarchy on a specific server.

Note: Throughout the rest of this section, we explain how to configure your Recovery Kit by selecting certain tasks from the **Edit** menu of the LifeKeeper GUI. You can also select each configuration task from the toolbar. You can also right-click a global resource in the Resource Hierarchy Tree (left-hand pane) of the status display window to display the same drop-down menu choices as the **Edit** menu.

You can also right-click a resource instance in the Resource Hierarchy Table (right-hand pane) of the status display window to perform all the configuration tasks, except *Creating a Resource Hierarchy*, depending on the state of the server and the particular resource.

Creating a MySQL Resource Hierarchy

IMPORTANT:

In a shared environment where the MySQL data directory files are on a shared disk, you must make sure that the shared file system is mounted and that the *fstab* file on the primary/template server has been updated with the shared device(s) information. If the file system resource is created first, the shared file system **MUST** be mounted on the same mount point on each server. It is also important to remember that a working communication path (i.e. heartbeat) is required before you can create your resource.

To create a resource instance from the primary server, you should complete the following steps:

1. From the LifeKeeper GUI menu, select **Edit**, then **Server**. From the drop-down menu, select **Create Resource Hierarchy**.

   If you wish to change a selection you have already entered or encounter an error message during any step in the creation of your MySQL resource hierarchy, you will generally be able to back up and change your selection or make corrections (assuming the **Back** button is enabled).

   Important: The MySQL database server daemon (**mysqld**) must be running when you create the resource.

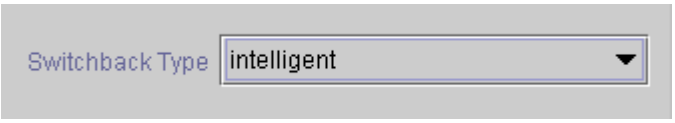
   A dialog box will appear with a drop-down menu listing all recognized Recovery Kits installed within the cluster. Select **MySQL Database** from the drop-down menu.

   Click **Next**.

   If you click the **Cancel** button at any time during the sequence of creating your hierarchy, LifeKeeper will cancel the entire creation process.

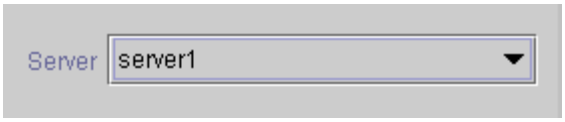
2. Select the **Switchback Type.** This dictates how the MySQL instance will be switched back to this server when it comes back into service after a failover to the backup server. You can choose either *intelligent* or *automatic*. Intelligent switchback requires administrative intervention to switch the instance back to the primary/original server. Automatic switchback means the switchback will occur as soon as the primary server comes back on line and reestablishes LifeKeeper communication paths.

   The switchback type can be changed later, if desired, from the General tab of the Resource Properties dialog box.

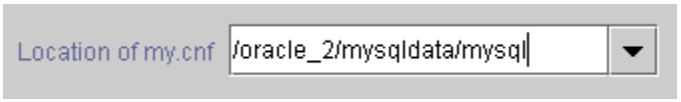
   Click **Next**.

3. Select the **Server** where you want to place the MySQL database (typically this is referred to as the primary or template server). All the servers in your cluster are included in the drop-down menu.



Click **Next** to proceed to the next dialog box.

4. Select or enter the **Location of my.cnf**. This is the full path name (excluding the file name) where the MySQL configuration file (*my.cnf*) is located.



Click **Next** to proceed to the next dialog box.

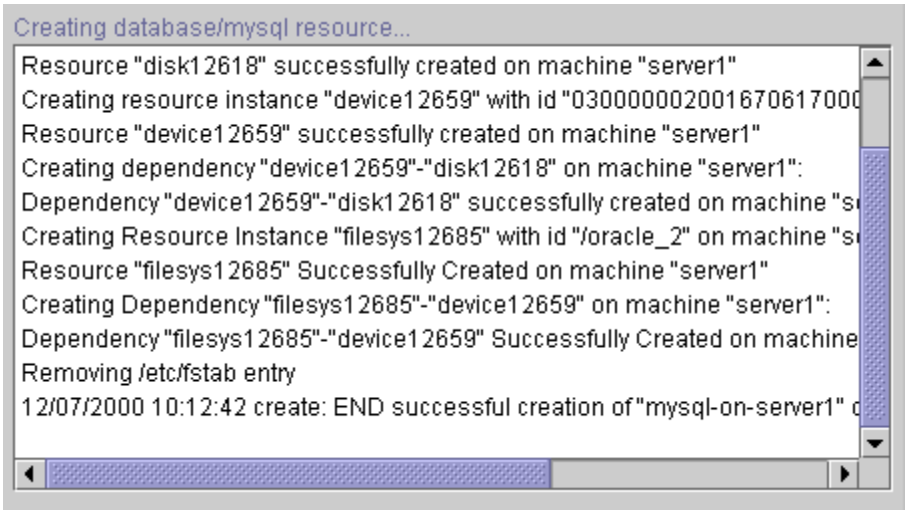
5. Select or enter the **Location of MySQL executables** location**.** This is the full path name of the binaries used to start and monitor the MySQL database server daemon.

Note: At this point, LifeKeeper will validate that you have provided valid data to create your MySQL resource hierarchy. If LifeKeeper detects a problem with either of this validation, an ERROR will appear on the screen. If the directory paths are valid, but there are errors with the MySQL configuration itself, you may pause to correct these errors and continue with the hierarchy creation.

Click **Next** to proceed to the next dialog box.

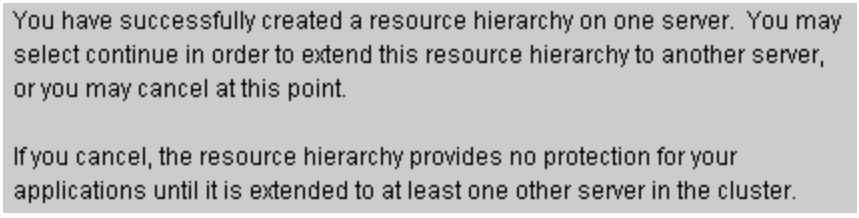
6. Select or enter the **Database Tag.** This is a tag name given to the MySQL hierarchy. You can select the default or enter your own tag name.

When you click **Create**, the Create Resource Wizard will create your MySQL resource.



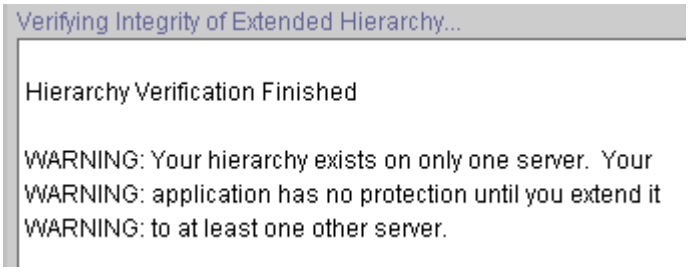
Note: The MySQL resource hierarchy should be created successfully at this point.

7. Another information box will appear explaining that you have successfully created an MySQL resource hierarchy, and you must **Extend** that hierarchy to another server in your cluster in order to place it under LifeKeeper protection.



When you click **Continue**, LifeKeeper will launch the Pre-Extend Wizard that is explained in the next section.

If you click **Cancel** now, a dialog box will appear warning you that you will need to come back and extend your MySQL resource hierarchy to another server at some other time to put it under LifeKeeper protection.



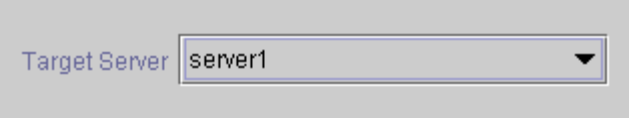
8. Click **Done** to exit.

Deleting a Resource Hierarchy

To delete a resource hierarchy from all the servers in your LifeKeeper environment, complete the following steps:

1. From the LifeKeeper GUI menu, select **Edit**, and then **Resource**. From the drop-down menu, select **Delete Resource Hierarchy**.
2. Select the name of the **Target Server** where you will be deleting your MySQL resource hierarchy.

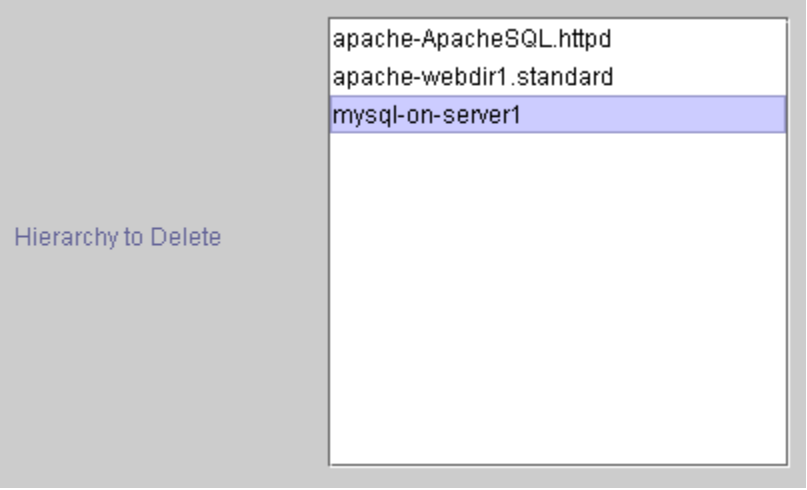
   Note: If you selected the Delete Resource task by right-clicking from the right pane on an individual resource instance, or from the left pane on a global resource where the resource is on only one server this dialog box will not appear.

   Click **Next**.

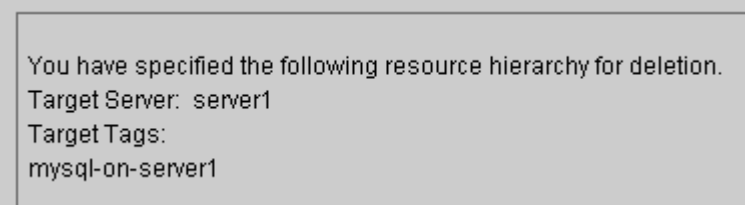
3. Select the **Hierarchy to Delete**. Identify the resource hierarchy you wish to delete, and highlight it.

   Note: If you selected the Delete Resource task by right-clicking from either the left pane on a global resource or the right pane on an individual resource instance, this dialog will not appear.

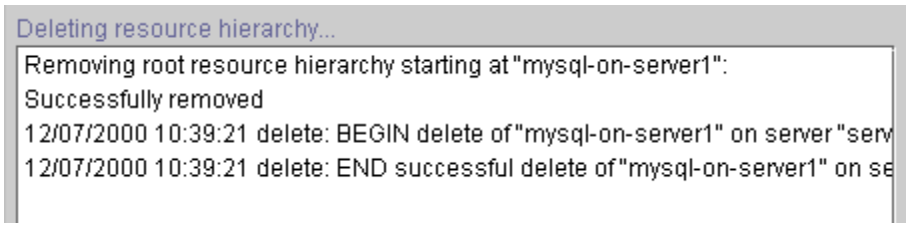
   Click **Next**.

4. An information box appears confirming your selection of the target server and the hierarchy you have selected to delete.

   Click **Delete**.

5. Another information box appears confirming that the MySQL resource was deleted successfully.



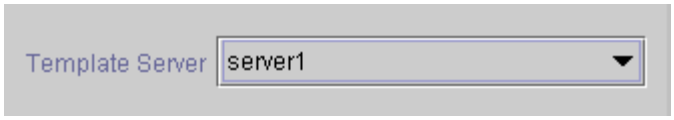
6. Click **Done** to exit.

Extending Your Hierarchy

After you have created a hierarchy, you will want to extend that hierarchy to another server in the cluster. There are three possible scenarios to extend your resource instance from the template server to a target server. The first scenario is when you "Continue" from creating the resource into extending that resource to another server. The second scenario is when you enter the Extend Resource Hierarchy task from the edit menu as shown below. The third scenario is when you right click on an unextended hierarchy in either the left or right hand pane. Each scenario takes you through the same dialog boxes (with a few exceptions, which are clearly detailed below).

1. If you are entering the Extend wizard from the LifeKeeper GUI menu, select **Edit**, then **Resource**. From the drop-down menu, select **Extend Resource Hierarchy**. This will launch the Extend Resource Hierarchy wizard.
2. The first dialog box to appear will ask you select the **Template Server** where your MySQL resource hierarchy is currently in service. It is important to remember that the **Template Server** you select now and the **Tag to Extend** that you select in the next dialog box represent an *in service* resource hierarchy. An error message will appear if you select a resource tag that is not in service on the template server you selected. The drop-down box in this dialog provides the names of all the servers in your cluster.

   Note: If you are entering the Extend Resource Hierarchy task immediately following the creation of a MySQL resource hierarchy, this dialog box will not appear, since the wizard has already identified the template server in the create stage. This is also the case when you right-click either the MySQL resource icon in the left pane or right-click on the MySQL resource box in the right pane the of the GUI window and choose *Extend Resource Hierarchy*.



   It should be noted that if you click **Cancel** at any time during the sequence of extending your hierarchy, LifeKeeper will cancel the extension process to that particular server. However, if you have already extended the resource to another server, that instance will continue to be in effect until you specifically unextend it.

   For example, let us say you have created your resource on Server 1 and extended that resource to Server 2. In the middle of extending the same resource to Server 3, you change your mind and click **Cancel** inside one of the dialog boxes. This will cancel only your action

to extend the resource to Server 3, not the extension you created to Server 2. If you want to remove Server 2 from this hierarchy, you must unextend the resource from Server 2.

Click **Next** to proceed to the next dialog box.

3. Select the **Tag to Extend**. This is the name of the MySQL instance you wish to extend from the template server to the target server. The wizard will list in the drop-down menu all the resources that you have created on the template server, which you selected in the previous dialog box.

   Note: Once again, if you are entering the Extend Resource Hierarchy task immediately following the creation of a MySQL resource hierarchy, this dialog box will not appear, since the wizard has already identified the tag name of your MySQL resource in the create stage. This is also the case when you right-click either the MySQL resource icon in the left hand pane or on the MySQL resource box in the right hand pane of the GUI window and choose *Extend Resource Hierarchy*.

   Click **Next**.

4. Select the **Target Server** where you are extending your MySQL resource hierarchy. The drop-down box provides the names of the servers in your cluster that are not already in the selected hierarchy.

   Click **Next**.

5. Select the **Switchback Type.** This dictates how the MySQL instance will be switched back to this server when it comes back into service after a failover to the backup server. You can choose either *intelligent* or *automatic*. Intelligent switchback requires administrative intervention to switch the instance back to the primary/original server. Automatic switchback means the switchback will occur as soon as the primary server comes back online and reestablishes LifeKeeper communication paths.

   The switchback type can be changed later, if desired, from the General tab of the Resource Properties dialog box.

   Click **Next**.

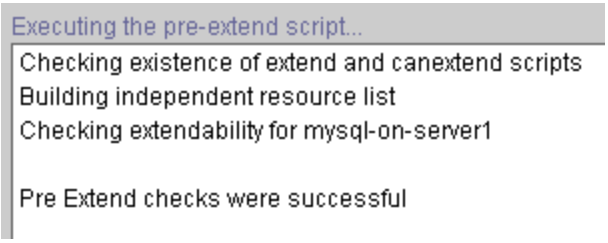
6. Select or enter a **Template Priority**. This is the priority for the Informix hierarchy on the server where it is currently in service. Any unused priority value from 1 to 999 is valid, where a lower number means a higher priority (1=highest). The extend process will reject any priority for this hierarchy that is already in use by another system. The default value is recommended. **Note:** This selection will appear only for the initial extend of the hierarchy.

Click **Next.**

7. Select or enter the **Target Priority**. This is the priority for the new extended MySQL hierarchy relative to equivalent hierarchies on other servers. Any unused priority value from 1 to 999 is valid, indicating a server's priority in the cascading failover sequence for the resource. A lower number means a higher priority (1=highest). Note that LifeKeeper assigns the number "1" to the server on which the hierarchy is created by default. The priorities need not be consecutive, but no two servers can have the same priority for a given resource.

Click **Next**.

8. An information box will appear explaining that LifeKeeper has successfully checked your environment and that all the requirements for extending this MySQL resource have been met. If there were some requirements that had not been met, LifeKeeper would not allow you to select the **Next** button, and the **Back** button would be enabled.

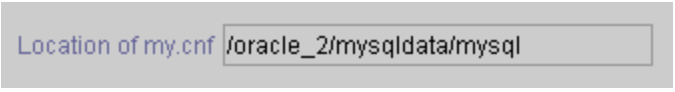


If you click **Back**, you can make changes to your resource extension according to any error messages that may appear in the information box.

If you click **Cancel** now, you will need to come back and extend your MySQL resource hierarchy to another server at some other time to put it under LifeKeeper protection.

When you click **Next**, LifeKeeper will launch you into the Extend Resource Hierarchy configuration task.

9. This dialog box is for information purposes only. You cannot change the **Location of my.cnf** that appears in the box. The MySQL instance acquired the location information from its configuration file.



Click **Next**.

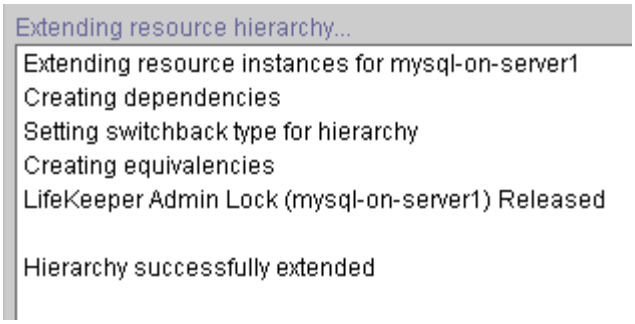
10. Select or enter the **Location of MySQL executables**. This is the full path name of the binaries used to start and monitor the MySQL database server daemon.

Click **Next**.

11. Select or enter the **Database Tag.** This is a tag name given to the MySQL hierarchy. You can select the default or enter your own tag name.

Click **Extend**.

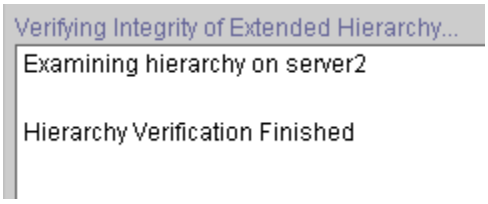
12. An information box will appear verifying that the extension is being performed.



Click **Next Server** if you want to extend the same MySQL resource instance to another server in your cluster. This will repeat the Extend Resource Hierarchy operation.

If you click **Finish**, LifeKeeper will verify that the extension of the MySQL resource was completed successfully.

13. If you clicked **Finish**, the following screen appears.



14. Click **Done** in the last dialog box to exit.

Note: Be sure to test the functionality of the new instance on *both* servers.

Unextending Your Hierarchy

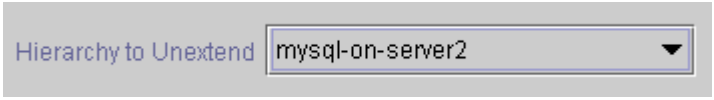
1. From the LifeKeeper GUI menu, select **Edit**, and **Resource**. From the drop-down menu, select **Unextend Resource Hierarchy**.
2. Select the **Target Server** where you want to unextend the MySQL resource. It cannot be the server where the MySQL resource is currently in service.

   Note: If you selected the Unextend task by right-clicking from the right pane on an individual resource instance this dialog box will not appear.

Click **Next**.

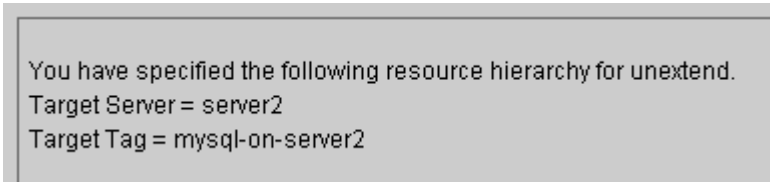
3. Select the MySQL **Hierarchy to Unextend.**

   Note: If you selected the Unextend task by right-clicking from either the left pane on a global resource or the right pane on an individual resource instance, this dialog will not appear.



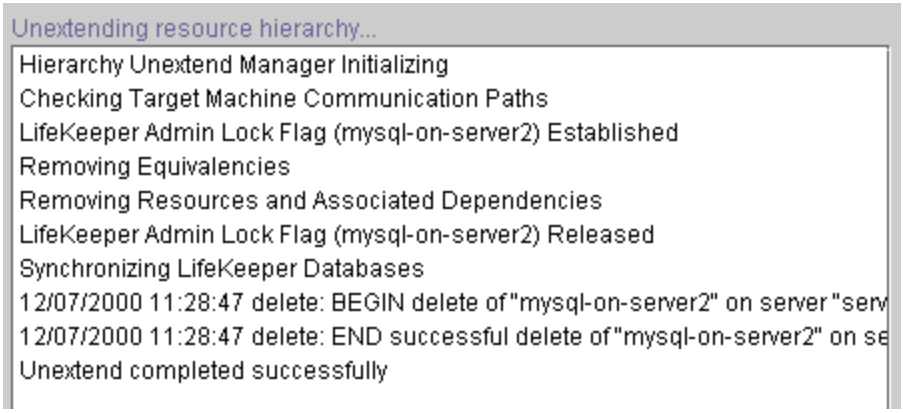
   Click **Next**.

4. An information box appears confirming the target server and the MySQL resource hierarchy you have chosen to unextend.



   Click **Unextend**.

5. Another information box appears confirming that the MySQL resource was unextended successfully.



6. Click **Done** to exit.

Testing Your Resource Hierarchy

You can test your MySQL resource hierarchy by initiating a manual switchover. This will simulate a failover of a resource instance from the primary server to the backup server.

Performing a Manual Switchover from the GUI

You can initiate a manual switchover from the LifeKeeper GUI by selecting **Edit**, **Resource**, and **In Service** from the drop-down menu. For example, an *in service* request executed on a backup server causes the application hierarchy to be placed in-service on the backup server and taken out of service on the primary server. At this point, the original backup server is now the primary server and original primary server has now become the backup server.

If you execute the **Out of Service** request, the application is taken out of service without bringing it in service on the other server.

LifeKeeper does not regulate or control internal operations such as rollbacks and backing-up archives. Tape archiving and restoration are the responsibility of the application administrator.

Recovery Operations

When the primary server fails, the MySQL Recovery Kit software performs the following tasks:

- Mounts the file system on the shared disk on the backup server
- Starts the daemon processes related to MySQL

Troubleshooting

Common Error Messages

The following error messages are common to all LifeKeeper for Linux Recovery Kits.

Note: In the Error Message column, a word in quotations and all capital letters refers to the name of a resource on the server (for example, "SERVER" might actually be a server named "Server1").

All Operations

The error messages that might be displayed during any operation are listed below, along with a suggested explanation for each.

Error Number	Error Message
000002	Usage error
000010	Error getting resource information
000011	Both Tag and ID name not specified
000019	Resource not found on local server
000022	END failed hierarchy "TAG" in service on server "SERVER"
000026	END failed ACTION for "TAG" on server "SERVER" due to "SIGNAL" signal

Hierarchy Creation

The error messages that might be displayed during a hierarchy creation extension are listed below, along with a suggested explanation for each. In some cases a corrective action is given.

Error Number	Error Message
000012	Switchback type not specified
000013	Usage error
000014	Resource with either matching tag “TAG” or ID exists
000015	ins_create failed on server “SERVER”
000018	Error creating resource “TAG” on server “SERVER”
000021	Removing resource instance "TAG" from server "SERVER" due to an error during creation
000023	Error bringing resource "TAG" in service on server "SERVER"
000024	Failed resource creation of resource “TAG” on server “SERVER”
000027	Removing file system dependency from "PARENT TAG" to "CHILD TAG" on server "SERVER" due to an error during creation
000028	Removing file system hierarchy "FILESYS TAG" created by "PARENT TAG" on server "SERVER" due to an error during creation
000029	Switchback type mismatch between parent "PARENT TAG" and child "CHILD TAG" on server "SERVER" **Action:** Switchback mismatches can lead to unexpected behavior. You can manually alter switchback types for resources using the ins_setas command to eliminate this mismatch.
000030	create: tag name not specified or extend: tag name not specified

Hierarchy Extension

The error messages that might be displayed during a hierarchy extension are listed below, along with a suggested explanation for each.

Error Number	Error Message
000003	Template resource “TAG” on server “SERVER” does not exist
000004	Template resource "TAG" cannot be extended to server "SERVER" because it already exists there
000005	Cannot access canextend script on server “SERVER”
000006	Cannot access extend script “PATH TO EXTEND” on server “SERVER”
000007	Cannot access depstoextend script "PATH TO DEPSTOEXEND" on server "SERVER"
000008	Cannot extend resource "TAG" to server "SERVER"
000009	Either “TEMPLATESYS” or “TEMPLATETAG” argument missing
000014	Resource with either matching tag “TAG” or ID exists
000015	ins_create failed on server “SERVER”
000018	Error creating resource “TAG” on server “SERVER”
000025	END failed resource extension of "TAG" on server "SERVER" due to a "SIGNAL" signal - backing out changes made to server
000030	create: tag name not specified or extend: tag name not specified

Restore

The error messages that might be displayed during a restore operation are listed below, along with a suggested explanation for each.

Error Number	Error Message
000023	Error bringing resource "TAG" in service on server "SERVER"

Resource Monitoring

The error messages that might be displayed during resource monitoring are listed below, along with a suggested explanation for each.

Error Number	Error Message
000001	Calling sendevent for resource "TAG" on server "SERVER"

MySQL Specific Error Messages

The following error messages are specific to the LifeKeeper for Linux MySQL Recovery Kit.

Note: In the Error Message column, a word in quotations and all capital letters refers to the name of a resource on the server (for example, “SERVER” might actually be a server named “Server1”).

All Operations

The error messages that might be displayed during any MySQL operation are listed below, along with a suggested explanation for each. In some cases a corrective action is given.

Error Number	Error Message
102001	Usage: “SCRIPT NAME” sysname dbvarname cnfpath exepath
102002	Usage: “SCRIPT NAME” cnfpath
102003	Usage: “SCRIPT NAME” exepath cnfpath
102004	Unable to obtain a valid value for the “socket” variable in “PATH”/my.cnf **Action:** There must be an entry for the “socket” in the 'mysqld' section of the my.cnf configuration file
102005	Unable to obtain a valid value for the “port” in “PATH”/my.cnf **Action:** There must be an entry for the “port” in the 'mysqld' section of the my.cnf configuration file
102006	Unable to obtain the data directory location **Action:** Please make sure that the database is running using the socket and port specified.

Error Number	Error Message
102007	Must specify the absolute path to the my.cnf configuration file
102008	Must specify the absolute path to the MySQL executables
102009	The file my.cnf does not exist in the path specified
102010	The MySQL executables do not exist in the path specified
102011	LifeKeeper was unable to start the MySQL database server
102012	LifeKeeper successfully started the MySQL database server
102013	LifeKeeper was unable to stop the MySQL database server
102014	LifeKeeper successfully stopped the MySQL database server
102015	The port “PORT NUMBER” is in use on the target server “SERVER”
102016	The MySQL database server is not running on server “SERVER”
102017	Unable to open the configuration file “PATH”/my.cnf
102018	Unable to get the Data Directory information for resource "TAG" on server "SERVER"
102019	Unable to get the configuration file location information for resource "TAG" on server "SERVER"
102020	Unable to get the executable location information for resource "TAG" on server "SERVER"
102021	The argument for the configuration file path is empty
102022	The argument for the executable path is empty
102023	The path “PATH” is not on a shared filesystem
102024	Unable to get the information for resource "TAG" on system “SYSTEM”
102025	The MySQL data directory “DATADIR” is already under LifeKeeper protection
102026	The port variables in the file /etc/my.cnf on “SERVER1” and “SERVER2” do not match
102027	The socket variables in the file /etc/my.cnf on “SERVER1” and “SERVER2” do not match
102028	Unable to obtain a valid value for the “user” variable in “PATH”/my.cnf **Action:** There must be a valid entry for the “user” variable in the 'client' section of the my.cnf configuration file
102029	Unable to obtain a valid value for the “password“ variable in “PATH”/my.cnf **Action:** There must be a valid entry for the “password”

Error Number	Error Message
	variable in the 'client' section of the my.cnf configuration file
102030	The user variables in the file /etc/my.cnf on “SERVER1” and “SERVER2” do not match
102031	The password variables in the file /etc/my.cnf on “SERVER1” and “SERVER2” do not match
102032	Unable to obtain the pid file location **Action:** There must be an entry for the “pid-file” variable in the ‘mysqld’ section of the my.cnf configuration file
102033	Unable to obtain a valid value for the "user" variable in “PATH”/my.cnf **Action:** The OS user must be specified using the "user" variable in the 'mysqld' section of the my.cnf configuration file
102034	WARNING: A my.cnf file exists at %s, which may override the values specified in the file at %s/my.cnf.
102035	The mysql system user "%s" does not exist on target server "%s"
102036	The mysql system user "%s" uids are different on target server "%s" and template server "%s"
102037	The mysql system user "%s" gids are different on target server "%s" and template server "%s"