SIOS Protection Suite for Linux
MySQL Recovery Kit
v9.3.1

管理ガイド

2018年11月

出版物の品質を維持するために、弊社は本書の正確性、明瞭性、構成、および価値に関するお客様のご意見を歓迎いたします。

以下の宛先に電子メールを送信してください。
ip@us.sios.com

目次

Chapter 1: はじめに **1**

MySQL Recovery Kit のドキュメンテーション **1**

SIOS Protection Suite のドキュメンテーション 1

要件 **1**

キットのハードウェア / ソフトウェア要件 1

Chapter 2: 設定上の考慮事項 **3**

MySQL の設定上の考慮事項 **3**

クライアント設定の考慮事項 **5**

設定の要件 **5**

設定例 **6**

アクティブ / スタンバイ設定 **6**

アクティブ / アクティブ設定 **8**

複数データベースサーバ環境に関する考慮事項 **12**

LifeKeeper で mysqld グループを使用する **12**

my.cnf ファイル 12

mysqld_multi コマンド 15

Network Attached Storage の使用 **15**

NAS Recovery Kit の使用 16

エラーメッセージ 16

MySQL 5.0 16

MySQL 5.5 16

解決方法 16

systemd 環境で使用する場合の考慮事項 **19**

Chapter 3: インストール **21**

LifeKeeper での MySQL のインストール / 設定 21

LifeKeeper の設定作業 **21**
MySQL リソース階層の作成 **22**
リソース階層の削除 **25**
リソース階層の拡張 **27**
リソース階層の拡張解除 **31**
Chapter 4: 管理 **33**
リソース階層のテスト 33
GUI による手動スイッチオーバの実行 **33**
GUI による手動スイッチオーバの実行 33
リカバリ動作 33
Chapter 5: トラブルシューティング **34**
共通のエラーメッセージ 34
MySQL 固有のエラーメッセージ 34

Chapter 1: はじめに

MySQL Recovery Kit のドキュメンテーション

SIOS Protection Suite for Linux MySQL Recovery Kit を使用すると、MySQL のリソースとデータベースのための LifeKeeper の障害回復保護機能を簡単に追加できます。これにより、プライマリデータベースサーバで障害が発生しても、わずかな時間で人手を介さずに指定されたバックアップサーバにて復旧することができます。

SIOS Protection Suite のドキュメンテーション

以下の SPS 製品ドキュメンテーションを SIOS Technology Corp から取得可能です。

- SPS for Linux リリースノート
- SPS for Linux テクニカルドキュメンテーション (LifeKeeper GUI の **[Help]** メニューからもアクセス可能)

本ドキュメンテーションおよびオプションのリカバリキットに関連するドキュメンテーションはSIOS テクニカルドキュメンテーションの Web サイトで入手可能です。

要件

キットのハードウェア / ソフトウェア要件

リカバリソフトウェアをインストールおよびセットアップするには、サーバが一定のハードウェア / ソフトウェア要件を満たしている必要があります。LifeKeeper MySQL Recovery Kit をインストールまたは削除する具体的な方法については、SPS for Linux インストールガイドを参照してください。

設定が以下の要件を満たしていることを確認してください。

- **サーバ**。この Recovery Kit には、SPS for Linux テクニカルドキュメンテーションおよび SPS for Linux リリースノートで説明されている要件に従って設定した LifeKeeper 対応の 2 台以上のコンピュータが必要です。
- **LifeKeeper ソフトウェア**。各サーバに同じバージョンの LifeKeeper ソフトウェアとパッチをインストールする必要があります。具体的な LifeKeeper の要件については、SPS for Linux テクニカルドキュメンテーションおよび SPS for Linux リリースノートを参照してください。
- **LifeKeeper IP Recovery Kit**。リモートクライアントが MySQL データベースにアクセスする場合に、このキットが必要になります。同じバージョンの Recovery Kit を各サーバにインストールする必要があります。
- **IP ネットワークインターフェース**。各サーバは、イーサーネット TCP/IP をサポートするネットワークインターフェースを 1 つ以上必要とします。IP スイッチオーバが正しく動作するには、ローカルネットワークに接続されているユーザシステムが標準の TCP/IP 仕様に準拠している必要があります。

注記: 各サーバが必要とするネットワークインターフェースが 1 つだけであっても、heterogeneous な要件、スループット要件、単一障害点の排除、ネットワークのセグメンテーションといった多くの理由のために、複数のインターフェースを使用すべきです。

- **TCP/IP ソフトウェア**。各サーバは TCP/IP ソフトウェアも必要とします。
- **MySQL ソフトウェア**。各サーバは、LifeKeeper と LifeKeeper MySQL Recovery Kit の設定に先だって MySQL ソフトウェアをインストールして設定しておく必要があります。各サーバには同じバージョンをインストールする必要があります。最新のリリースの互換性および購入方法については、SPS for Linux リリースノートを参照するか、営業担当者にお問い合わせください。

Chapter 2: 設定上の考慮事項

本セクションでは、典型的な LifeKeeper MySQL 設定の定義および例と、MySQL の設定を開始する前に検討すべき情報について説明します。

LifeKeeper Core リソース階層の設定方法については、SPS for Linux テクニカルドキュメンテーションのリソース階層 セクションを参照してください。

MySQL の設定上の考慮事項

LifeKeeper MySQL 環境に関して、次のような特有の考慮事項について検討する必要があります。

プライマリサーバとバックアップサーバで MySQL データベースサービスを運用するには、ファイルシステムとディスクパーティションが各サーバからアクセスできなければなりません。MySQL Recovery Kit の設定を開始する前に、必ず次の準備手順を終了し、各サーバ上のデータベースをテストして稼働させる必要があります。以下の手順では、ユーザ「mysql」は MySQL サーバを開始するオペレーティングシステムユーザを表します。

1. すべてのサーバに MySQL サーバとクライアントコンポーネントをインストールします。必ず、すべてのサーバで同じバージョンの MySQL クライアントコンポーネントとサーバコンポーネントが稼働するようにします。MySQL 実行可能ファイルは、ローカルまたは共有ドライブ上に置くことができます。
2. LifeKeeper が保護する MySQL データベースサーバを稼働させたいソケットおよび / またはポート上で mysqld が稼働しているサーバがあれば、`mysqladmin` コマンドを使用して各 MySQL サーバを停止します。
3. MySQL データディレクトリの内容を共有の場所に移動します。デフォルトで、MySQL データディレクトリはローカルドライブにインストールされます。この場所は、ディストリビューションのメカニズムによって異なります。バイナリ RPM は、データディレクトリを */var/lib/mysql* にインストールします(必ず内容のみを移動して、ディレクトリはそのまま残してください。これにより、MySQL データベースサーバが必要に応じてこのディレクトリ内にログを書き込むことができます。ステップ 4 で説明されている「mysql」ユーザが、この場所にログを書き込む権限を持っていることを確認します)。
4. インストール処理で Linux ユーザ「mysql」が作成されなかった場合は、このユーザを作成します。セキュリティ上の理由から、MySQL サーバを「root」として実行しないでください(セキュリティ問題の詳細については、『*MySQL Administration Guide*』を参照してください)。必ず、「mysql」ユーザのみがデータベースディレクトリへの読み取り/書き込み権限を持つようにします。「mysql」ユーザおよびグループはすべてのサーバ上に作成する必要があります。ユーザ ID とグループ ID は、すべてのサーバで同じでなければなりません。
5. **重要:**/etc/rc.d/init.d/mysql によって開始されるサーバは、LifeKeeper の保護下には入れません。さらに、このサーバは LifeKeeper の保護下にあるサーバと同じポート番号またはソケットを使用できません。
6. ソケットを共有ディスク上のデータディレクトリに書き込むことを推奨します。ソケットをローカルディスクに書き込む場合、階層が存在するすべての LifeKeeper サーバ上の同じパスに存在することを確認します。「mysql」ユーザが、この場所にソケットを書き込む権限を持っていることを確認します。

7. 設定に適した mysql デーモンスタートアップコマンドを使用して MySQL サーバを開始します。*my.cnf* ファイルで単一インスタンスを定義している設定の場合は、以下のコマンドを使用してください。

```
<start command> --user=mysql --socket=<socket> --port=<port number>

--datadir=<path to the data directory> --log &
```

<start command> は、mysql バージョン 3.x では `safe_mysqld`、バージョン 4.x では `mysqld_safe` です。

my.cnf ファイルで mysqld グループを使用した設定の場合は、以下のコマンドを使用してください。

```
mysqld_multi start <group number>
```

<group number> は mysqld グループの *my.cnf* ファイルで定義したインスタンス番号を表しています。LifeKeeper で mysqld グループを使用する際の詳細については、LifeKeeper で mysqldGroup を使用するを参照してください。

Systemd を採用しているディストリビューションで MySQL(5.7.6 以降) が Systemd を使用するように設定されている場合、systemctl コマンドを利用する必要があります。詳しくは Systemd 環境で使用する場合の考慮事項を参照してください。

8. 「mysql」という名前で MySQL データベースユーザを作成します。このユーザにパスワードを指定し、「shutdown」権限を付与します。これは、1 台のサーバ上でのみ行います(ユーザの作成と権限付与の詳細については、『*MySQL Administration Guide*』を参照してください)。

9. サンプル設定ファイル *my.cnf* を必要な場所 (*/etc* もしくは */<datadir>*)にコピーします。このファイルには、データベースサーバ用とクライアントプログラム用のオプションが含まれています。

ファイルは、*MySQL* データディレクトリまたは /etc ディレクトリのどちらかに置くことが可能です。*/etc/my.cnf* ファイルにはグローバルオプションが含まれます。マシン上で常に 1 つのデータベースのみが稼働する場合 (つまりアクティブ / スタンバイ設定)、もしくは mysqld グループ機能を使用している場合は */etc* に *my.cnf* ファイルを置きます。(LifeKeeper で mysqld グループを使用するを参照してください。)ファイルが */etc* にある場合、そのファイルを各 LifeKeeper バックアップサーバにコピーする必要があります。データディレクトリ内の *my.cnf* ファイルには、サーバ固有のオプションが含まれている必要があります。複数サーバおよびアクティブ / アクティブ設定の場合、mysqld グループ機能を使用しない限りはこのファイルをリソースインスタンスごとにデータディレクトリに格納する必要があります。(LifeKeeper で mysqld グループを使用するを参照してください。)

注記 :サーバ固有のオプションを含む *my.cnf* ファイルは /etc ディレクトリと */<datadir>* ディレクトリの両方に存在してはいけません。もし、サーバ固有のオプションを含む *my.cnf* ファイルが /etc ディレクトリに存在し、同時に、保護された *my.cnf* ファイルが */<datadir>* にインストールされている場合、コンフリクトの原因となる場合があります。詳細は MySQL のドキュメントのグローバルオプションとサーバ固有のオプションの設定を参照してください。

以下のエントリを追加または編集します。

a. ファイルの「client」セクションに、接続に使用するユーザとパスワードを指定します。

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

.
.

b. ファイルの「mysqld」セクションには、接続に使用するソケットとポートを、mysqld プロセスに対する pid ファイルの場所とともに指定します。ユーザ変数では、mysqld プロセスを開始するオペレーティングシステムユーザを指定します。

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

注記:このファイルが正しく保護され、ユーザ「mysql」によって所有されていることを確認します。

注記: MySQL 階層を作成した後、my.cnf ファイル内の情報を変更する必要がある場合は、変更を加える前に階層を out-of-service (つまり OSU 状態)にして mysql サーバインスタンスを停止します。

注記:上記の *my.cnf* 設定の例は、単一のデータベースインスタンス *mysqld* について説明しています。mysqld グループを使用した設定例については LifeKeeper で mysqld グループを使用するを参照してください。

注記:include文はサポートしていません。単一のmy.cnfに全ての設定を記述してください。

クライアント設定の考慮事項

MySQL データベースクライアントを設定する上で、次のような考慮事項があります。

- クライアントがリモートホストから接続する場合、クライアントが接続に使用する LifeKeeper 保護下の IP アドレスを 1 つ作成します。
- クライアントは、LifeKeeper が保護する IP アドレス経由でデータベースサーバに接続するよう設定する必要があります。
- クライアントがドメイン名を使用して接続する場合は、各クライアントの hosts ファイル内に保護する IP アドレスのエントリを作成するか、DNS 内に名前を設定します。クラスタ内のすべてのクライアントおよびすべての LifeKeeper サーバから ping を実行して、保護する IP アドレスをテストします。
- 各ユーザは、それぞれのマシンのホームディレクトリに *my.cnf* ファイルを持つことができますが、L-ifeKeeper は */etc* ディレクトリまたはデータディレクトリにある *my.cnf* ファイルのみを使用します。*my.cnf* ファイルにはクライアント接続情報(つまり、ポート、ソケット ID、ユーザ、パスワード)が格納されています。

設定の要件

それぞれの例では、1 つまたは 2 つのデータベースインスタンス (databaseA、databaseB) が使用されています。データベースタグ名は、これらのデータベースインスタンスを LifeKeeper に記述するための任意の名前です。分かりやすい名前を指定することをお勧めします。mysqld グループを使用した設定の LifeKeeper が提示するデフォルトのタグ名は mysql または mysql<group number> です (LifeKeeper で mysqld グループを使用するを参照)。設定例を理解するには、次のような設定要件を留意してください。

- **LifeKeeper 階層。** LifeKeeper の管理を実行するときに、主要な階層は、管理するサーバ上に作成されている階層を参照します。設定の表については、最初の管理画面で入力される情報は Server 1 の観点からのものです。2 番目の画面が表示されると、2 台目のサーバを管理するときに作成される階層を参照します。設定例では、2 台目のサーバは Server 2 です。
- **1 台のサーバによってロックされる共有ディスク。** LifeKeeper を使用するとき、LifeKeeper の保護下にある共有ストレージリソースは、1 台のサーバが使用するためにリザーブしています。これは SCSI リザベーションを使用して行われます。共有デバイスがディスクアレイの場合、LUN 全体がリザーブされます。共有デバイスがディスクの場合は、ディスク全体がリザーブされます。こうすることで、クラスタ内の他のサーバが不用意にデータを破壊してしまうことを避けられます。サーバに障害が発生すると、優先順位が最も高いバックアップサーバがそれまでのリザベーションを解除して新たにリザベーションを行い、他のサーバをすべてロックします。
- **共有ディスク上のデータディレクトリ。** LifeKeeper MySQL Recovery Kit が正しく機能するためには、データベースインスタンスのデータディレクトリ (datadir) が常に共有ディスク上にある必要があります。データディレクトリは、ファイルシステム上になければなりません。このファイルシステムは、プライマリサーバおよびバックアップサーバの両方からマウントできなければなりません。データディレクトリ (datadir) はレプリケーションされたディスクもしくは Network Attached Storage (NAS) 上に存在することが可能です。

設定例

本セクションの例は、MySQL データベースインスタンスの設定方法を示しています。各図は、設定の種類と MySQL パラメータの関係を示しています。また、各設定は、本書で説明している設定の規則と要件に従って、MySQL 設定と LifeKeeper ソフトウェアとの互換性を確保しています。

本セクションでは、設定の要件を説明し、次にそれぞれの設定例を示します。

- アクティブ / スタンバイ
- アクティブ / アクティブ

本セクションに挙げたものは可能な設定の例にすぎませんが、これらの設定を理解し、設定規則に従うことにより、使用するコンピューティング環境で実現可能なソリューションを明確にし設定する際に役立ちます。

設定の要件

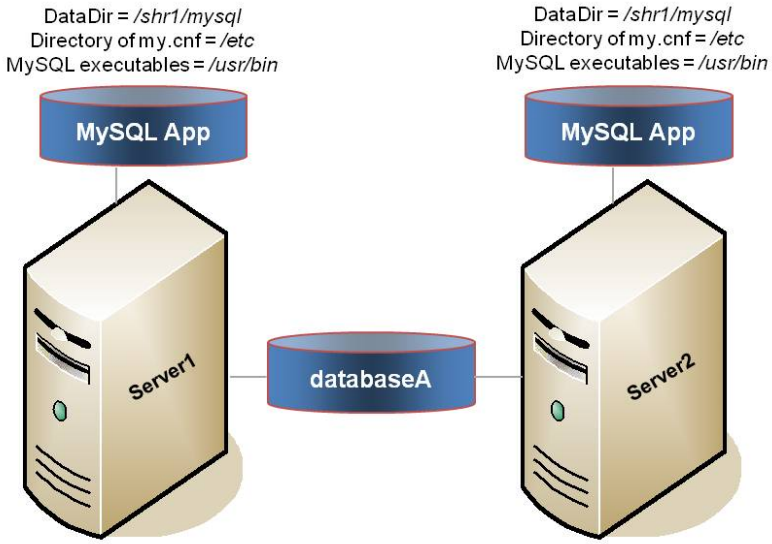
例 1 - アクティブ / スタンバイ設定

例 2 - アクティブ / アクティブ設定

アクティブ / スタンバイ設定

本セクションでは、アクティブ / スタンバイ設定の例を示します。この設定では、Server 1 がデータベースに排他的にアクセスしているため、Server 1 がアクティブであると見なされます。Server 2 は別の処理を実行します。Server 1 に障害が発生すると、Server 2 がデータベースにアクセス可能になり、LifeKeeper はデータベース操作を再確立します。

図 1.アクティブ / スタンバイ設定、例 1



設 定 に関 する注 記 :

- どちらのサーバも共 有 ディスク上 の MySQL データディレクトリ(データベース databaseA が格 納 されている)を使 用 します。
- MySQL データディレクトリへのパスはどちらのサーバでも同じです。
- *my.cnf* 設 定 ファイルはローカルディスクの */etc* にあります。
- MySQL 実 行 ファイルは各 サーバのローカルドライブの */usr/bin* にあります。
- Server 2 は、Server 1 がアクティブである間 は、共 有 ディスク上 のファイルとディレクトリにアクセスできません。

Server 1 でのリソース階 層 の作 成

サーバ:	Server1
my.cnf ファイルがあるディレクトリ:	*/etc*
MySQL 実 行 可 能 ファイルがあるディレクトリ:	*/usr/bin*
データベースタグ:	mysql-on-server1

リソース階 層 の Server 2 への拡 張

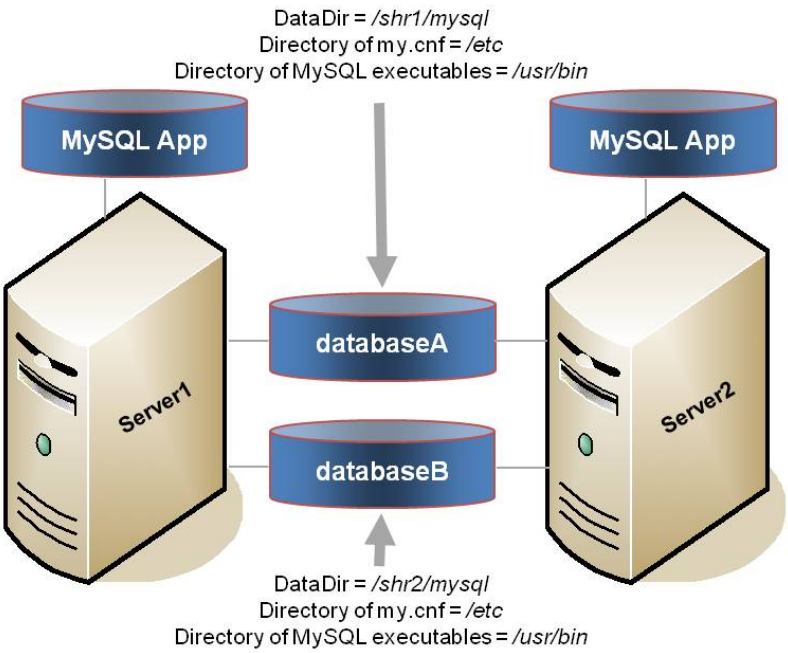
テンプレートサーバ:	Server1
拡 張 するタグ:	mysql-on-server1

ターゲット サーバ:	Server2
ターゲットの優 先 順 位 :	10
my.cnf ファイルがあるディレクトリ:	*/etc*
MySQL 実 行 可 能 ファイルがあるディレクトリ:	*/usr/bin*
データベースタグ:	mysql-on-server2

アクティブ / アクティブ設 定

アクティブ / アクティブ設 定 は、異 なるデータベースインスタンスを実 行し、互 いにバックアップの役 割を果 たす 2 台 以 上 のサーバで構 成 されます。各 データベースインスタンスは、別 々 の共 有 物 理 ディスク上 に置 かれていなければなりません。複 数 の MySQL データーベースインスタンス (同じバージョンもしくは異 なるバージョン) をサポートする LifeKeeper の設 定 のために、SIOS では mysqld グループ機 能 をサポートするバージョンの MySQL を使 用 することを推 奨 しています。これらの設 定 では *my.cnf* 設 定 ファイルは */etc* に存 在 します。mysqld グループ機 能 をサポートしないバージョンの MySQL では、*my.cnf* 設 定 ファイルは各 データベースインスタンスの共 有 ファイルシステムの MySQL データディレクトリに存 在 している必 要 があります(例 : 下 記 図 2 では */shr1/mysql* および */shr2/mysql*)。

図 2.アクティブ / アクティブ設 定 、例 2



設 定 に関 する注 記 :

- 各 サーバは異 なる共 有 ディスク上 にある異 なる MySQL データディレクトリ(データベースインスタンス database A と database B を含 む)を使 用 します。

- MySQL データディレクトリへのパスは、サーバ上 に定 義 されているインスタンスごとに異 なります。
- *my.cnf* 設 定 ファイルは */etc* に配 置 され、各 データベースインスタンスの mysqld グループセクションを含 んでいます。各 セクションはデータベースインスタンスに対 する一 意 の MySQL データディレクトリ、ポート、ソケットを定 義 します。 *my.cnf* 設 定 ファイルはクラスタの全 ノードで常 に同 一 の内 容 であることに留 意 してください。mysqld グループをサポートしないバージョンの MySQL が起 動 しているシステムでは、各 データベースインスタンスに対 する *my.cnf* 設 定 ファイルは共 有 ドライブ上 のデータディレクトリに配 置 されています。各 設 定 ファイルはデータベースインスタンスに対 する一 意 の MySQL データディレクトリ、ポート、ソケット定 義 を定 義 します。
- MySQL 実 行 ファイルは各 サーバのローカルドライブの */usr/bin* にあります。
- 最 初 は、Server 1 で databaseA が、Server 2 で databaseB が稼 動 します。スイッチオーバが発 生 した場 合 は、1 台 のサーバで両 方 のデータベースが稼 動 できます。

Server 1 での 1 つ目 のリソース階 層 の作 成

サーバ:	Server1
my.cnf ファイルがあるディレクトリ:	*/etc*
MySQL 実 行 可 能 ファイルがあるディレクトリ:	*/usr/bin*
データベースタグ:	mysql-shared.example.instance1

1 つ目 のリソース階 層 の Server 2 への拡 張

テンプレートサーバ:	Server1
拡 張 するタグ:	mysql-shared.example.instance1
ターゲットサーバ:	Server2
ターゲットの優 先 順 位 :	10
my.cnf ファイルがあるディレクトリ:	*/etc*
MySQL 実 行 可 能 ファイルがあるディレクトリ:	*/usr/bin*
データベースタグ:	mysql-shared.example.instance1

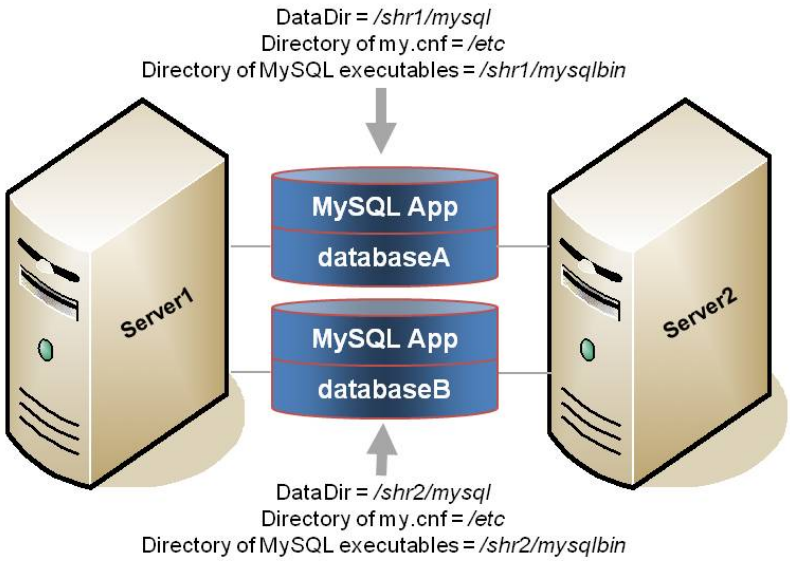
Server 2 での 2 つ目 のリソース階 層 の作 成 :

サーバ:	Server2
my.cnf ファイルがあるディレクトリ:	*/etc*
MySQL 実 行 可 能 ファイルがあるディレクトリ:	*/usr/bin*
データベースタグ:	mysql-shared.example.instance2

2 つ目 のリソース階 層 の Server 1 への拡 張 :

テンプレートサーバ:	Server2
拡 張 するタグ:	mysql-shared.example.instance2
ターゲットサーバ:	Server1
ターゲットの優 先 順 位 :	10
my.cnf ファイルがあるディレクトリ:	*/etc*
MySQL 実 行 可 能 ファイルがあるディレクトリ:	*/usr/bin*
データベースタグ:	mysql-shared.example.instance2

図 3.アクティブ / アクティブ設 定 、例 2



設 定 に関 する注 意 :

- 各 サーバは異 なる共 有 ディスク上 にある異 なる MySQL データディレクトリ(データベースインスタンス database A とdatabase B を含 む)を使 用 します。
- MySQL データディレクトリへのパスは、サーバ上 に定 義 されているデータベースインスタンスごとに異 なりま す。
- *my.cnf* 設 定 ファイルは*/etc* に配 置 され、各 データベースインスタンスの mysqld グループセクションを含 んでいます。各 セクションはデータベースインスタンスに対 する一 意 の MySQL データディレクトリ、ポート、ソケットを定 義 します。 my.cnf 設 定 ファイルはクラスタの全 ノードで常 に同 一 の内 容 であることを留 意 してください。mysqld グループをサポートしないバージョンの MySQL が起 動 しているシステムでは、各 データベースインスタンスに対 する my.cnf 設 定 ファイルがデータベースのデータディレクトリの共 有 ドライブ上 に配 置 されています。各 設 定 ファイルはデータベースインスタンスに対 する一 意 の MySQL データディレクトリ、ポート、ソケットを定 義 します。

- データディレクトリを含 む共 有 ディスクのそれぞれに MySQL 実 行 可 能 ファイルのコピーがあります。
- 最 初 は、Server 1 で databaseA が、Server 2 で databaseB が稼 動します。スイッチオーバが発 生した場 合 は、1 台 のサーバで両 方 のデータベースが稼 動します。

Server 1 での 1 つ目 のリソース階 層 の作 成

サーバ:	Server1
my.cnf ファイルがあるディレクトリ:	*/etc*
MySQL 実 行 可 能 ファイルがあるディレクトリ:	*/shr1/mysqlbin*
データベースタグ:	mysql-shared.example.instance1

1 つ目 のリソース階 層 の Server 2 への拡 張 :

テンプレートサーバ:	Server1
拡 張 するタグ:	mysql-shared.example.instance1
ターゲットサーバ:	Server2
ターゲットの優 先 順 位 :	10
my.cnf ファイルがあるディレクトリ:	*/etc*
MySQL 実 行 可 能 ファイルがあるディレクトリ:	*/shr1/mysqlbin*
データベースタグ:	mysql-shared.example.instance1

Server 2 での 2 つ目 のリソース階 層 の作 成 :

サーバ:	Server2
my.cnf ファイルがあるディレクトリ:	*/etc*
MySQL 実 行 可 能 ファイルがあるディレクトリ:	*/shr2/mysqlbin*
データベースタグ:	mysql-shared.example.instance2

2 つ目 のリソース階 層 の Server 1 への拡 張

テンプレートサーバ:	Server2
拡 張 するタグ:	mysql-shared.example.instance2
ターゲットサーバ:	Server1
ターゲットの優 先 順 位 :	10
my.cnf ファイルがあるディレクトリ:	*/etc*
MySQL 実 行 可 能 ファイルがあるディレクトリ:	*/shr2/mysqlbin*
データベースタグ:	mysql-shared.example.instance2

複数データベースサーバ環境に関する考慮事項

複数のMySQL データベースサーバおよびデータベースがある場合、次のような設定上の考慮事項があります。

- アクティブ / アクティブまたは複数の(同一もしくは異なるバージョンの) MySQL インスタンスを稼動させる場合、可能であれば mysqld グループの機能を使用することを考慮してください。SIOS は、複数の MySQL データベースサーバ構成に対して mysqld グループ (mysqld_multi) を使用することを推奨します。
- アクティブ / アクティブまたは複数の MySQL インスタンスを稼働させる場合、共有ファイルシステムを */var/lib/mysql* としてマウントしないでください。mysql 起動コマンド (`safe_mysqld or mysqld_safe`)により、MySQL サーバが予期せずシャットダウンされるためです。
- mysqld グループの機能を使用しない場合、*my.cnf* ファイルは、各アクティブ/アクティブもしくは複数サーバのデータディレクトリに保存する必要があります。mysqld グループを使用する構成では、*my.cnf* ファイルはデータディレクトリではなく */etc* に保存されます。LifeKeeper および mysqld グループ機能に関する情報は LifeKeeper で mysqld グループを使用するを参照してください。
- MySQL 用の追加のポート番号は、*/etc/services* ファイルに指定する必要があります。
- 各 MySQL データベースサーバは、個別のポート上で稼働し、個別のソケットファイルにアクセスするように設定する必要があります。これらの設定オプションは、データディレクトリにある *my.cnf* ファイル内に指定されます。
- 各サーバは、個別の共有の位置からデータにアクセスするように設定する必要があります(つまり、各サーバは個別のデータディレクトリを使用する必要があります)。

LifeKeeper で mysqld グループを使用する

MySQL Application Recovery Kit は、mysqld_multi を介して管理される mysqld グループ機能を使用した *my.cnf* ファイルをサポートします。この MySQL の機能を使用すると、単一の my.cnf ファイル (通常 */etc* に保存される) を介して、容易に複数の MySQL インスタンスを設定できるようになります。本キットは mysqld グループのフォーマットを利用している my.cnf ファイルを検出し、管理者に対して保護対象とする mysqld グループの番号を選択することを要求します。管理者が選択できる選択リストは、my.cnf ファイルで定義されているグループ番号から、すでに本キットにより保護されているグループ番号を差し引いたものから決定します。

一般的には、mysqld グループ機能を使用して複数の MySQL インスタンスを設定および制御するほうが容易になります。SIOS は、アクティブ / アクティブもしくは複数インスタンスの設定時には、このアプローチを使用することを推奨します。

my.cnf ファイル

mysqld グループ機能を使用する場合、以下が必須となります。

a. 単一の my.cnf ファイルが、データベースインスタンスの mysqld グループを定義するために使用される必要があります。

b. my.cnf ファイルは共有ストレージ上に配置しないでください。

c. 各クラスタノードに my.cnf ファイルの完全なコピーが必要です (*/etc/my.cnf* が望ましい)。

d. my.cnf ファイルに対して行ったすべての変更は、LifeKeper のすべてのクラスタノードに反映される必要があります。

Recovery Kit は、my.cnf ファイルが mysqld グループを使用していることを検知すると mysqld_multi コマンドを使用します。これに基づいて、LifeKeeper の制御下に配置する前に mysqld_multi を使用して MySQL インスタンスをテストすることができます。

以下は、mysqld_multi によって制御されている 2 つのデータベースインスタンスについて mysqld グループを使用した比較的複雑な my.cnf ファイルの解説です。`mysqld_multi` コマンド (および MySQL LifeKeeper Recovery Kit) は、管理者にさまざまな設定を行うためのオプションを提供しています。下記の例、`[mysqld1]` では、さまざまな MySQL コマンドの大部分についてデフォルトのロケーションを使用する比較的単純な MySQL インスタンスを定義しています。その次の例、`[mysqld55]` では、より多くの変更を行っています。コメントは各セクションにおいて LifKeeper がどのように MySQL と相互作用しているかについて説明しているので参考にしてください。

```
# The following client section defines which username/password combination
will be used for
# LifeKeeper  connections. The username/password combination needs to be
defined in each MySQL
# Database instance that will be d escribed in this my.cnf file.
[client]
user  =  steeleye
password =  password

# This next section describes the default version of mysqld and mysqldadmin
that mysqld_multi
# will use when processing mysqld_multi commands. The username/password combo
defines the
# MySQL account that mysqld_multi will  use when working with the database
instances. This
# username and password combo needs to be d efined in  each MySQL Database
instance that will be
# controlled by mysqld_multi. See how to set up the  multi_admin account in the
MySQL Reference
# Manual, by issuing "mysqld_multi --example".
[mysqld_multi]
mysqld  = /usr/bin/mysqld_safe
mysqladmin = /usr/bin/mysqladmin
user  = multi_admin
password  = password

# The next section defines the first of two MySQL Database instances in this
my.cnf file. Note
# that each section starts  with a [mysqldNN] where NN is the mysqld group
number (or instance).
# Each group name must have a number. There are a  number of directives that
the LifeKeeper MySQL
# Recovery Kit will be looking for in these sections.
[mysqld1]
```

```
datadir  = /s11/mysql-data5077  # Defines where the data files for the
instance will live. For
   #  LifeKeeper, this directory must be on LifeKeeper protected
   #  (shared or replicated) storage.
mysqld  = /usr/bin/mysqld_safe  # Defines specifically which mysqld command
will be used for
   #  starting the instance. This one is using the
   #  default  mysqld_safe that came with the distribution.
socket=/s11/mysql-data5077/moe.socket # Defines the location of the socket for
this instance.
   #  If the socket is not on LifeKeeper protected storage, it
   #  needs to be defined in exactly the same place on each
   #  node in the cluster and be owned by the "user" defined
   #  below.
port  =  3307  # Each instance needs its own, unique TCP/IP port.
pid-file = /var/run/mysqld/mysqld.pid # The pid-file can be on LifeKeeper
protected or
   #  non-LifeKeeper protected storage.
log-error= /var/log/mysqld.log  # Location of the MySQL error log for this
instance. Can be
   #  on LifeKeeper protected or non-LifeKeeper protected
   #  storage.
user  = mysql  # The Linux user name that will run the MySQL processes.

# The next section defines the more complicated of the two MySQL instances.
Instance "55" is not
# using the default MySQL that  came with the Linux distribution as it is
using the 5.5.12 version
# of MySQL that was installed from source. The binaries for this  version were
installed onto shared
# storage, and the binary directory is LifeKeeper protected.
[mysqld55]
datadir = /s11/mysql-data5512  # Same as above; this instance uses a different
data
   #  d irectory, and  this directory is on LifeKeeper
   #  protected storage.
mysqld  =/s11/mysql5512/bin/mysqld_safe # For this instance, a different
version of mysqld_safe
   #  is used; the  one that is included with 5.5.12.
socket=/s11/mysql-misc5512/larry.socket # This instance has the socket on
LifeKeeper protected
   #  storage, but not  in the default location (datadir).
port  = 3308  # This instance has a unique TCP/IP port as well.
pid-file = /var/run/mysqld/mysqld55.pid # This instance's pid-file is not on
LifeKeeper protected
   #  storage.
log-error = /var/log/mysqld55.log  # This instance's log-error (error log) is
not on
   #  LifeKeeper protected  storage.
log-bin  = /s11/mysql-log5512/larry  # The log-bin directive specifies where
the binary
```

```
   #  transaction logs are  located for this instance.
   #  These logs must be on LifeKeeper  protected storage
   #  (the recovery kit will enforce this). By default,
   #  these logs are in the datadir.
user  = mysql  # The Linux user name that will run the MySQL processes.
```

マルチインスタンス用 [mysqld<N>] の設定とシングルインスタンス用 [mysqld] の設定を併記する場合は、シングルインスタンス用の設定を最後に記載してください。
例:

```
[mysqld1]
(mysqld1の設定)

 [mysqld2]
(mysqld2の設定)

 [mysqld55]
(mysqld55の設定)

 [mysqld]
(シングルインスタンス用mysqldの設定)
```

mysqld_multi コマンド

例として mysql コマンドを実行します。

```
#  mysqld_multi start 1
```

my.cnf で[mysqld1]として定義した mysqld グループ 1 インスタンスを、依存するすべての LifeKeeper 保護リソースが LifeKeeper のいずれかのノードで In Service であると仮定します。

以下のコマンドを実行します。

```
# mysqld_multi report 1
```

このインスタンスの状態がレポートされます (例 : running (起動) または not running (停止))。このインスタンスが起動すると、LifeKeeper でのリソース作成が容易になります。

mysqld_multi 形式の my.cnf ファイルの設定方法に関する情報については以下のコマンドを実行してください。

```
# mysqld_multi --example
```

Network Attached Storage の使用

クラスタストレージとして NFS ファイルサーバ (Network Attached Storage) を使用するために LifeKeeper を設定するにあたっていくつかの特別な考慮事項があります。

NAS Recovery Kit の使 用

オプションの Network Attached Storage (NAS) Recovery Kit は、LifeKeeper for Linux で NFS サーバを共 有 スト レージアレイとして使 用 する場 合 に必 要 になります。各 クラスタノードに NAS Recovery Kit (およびライセンス) を インストールしてください。詳 細については、NAS Recovery Kit のド キュメンテーションを参 照してください。

エラーメッセージ

MySQL と Network Attached Storage (NAS) を使 用 する際に、システムクラッシュにより My SQL インスタンスが フェイルオーバ後 に再 起 動 できないといった事 象 に遭 遇 する可 能 性 があります。MySQL エラーログで、エラーの 原 因 が分 かります。

MySQL 5.0

```
110523 22:10:58  mysqld started
InnoDB: Unable to lock ./ibdata1, error: 11
InnoDB: Check that you do not already have another mysqld process
InnoDB: using the same InnoDB data or log files.
110523 22:10:58  InnoDB: Retrying to lock the first data file
InnoDB: Unable to lock ./ibdata1, error: 11
InnoDB: Check that you do not already have another mysqld process
InnoDB: using the same InnoDB data or log files.
```

MySQL 5.5

```
110524 10:52:20 InnoDB: The InnoDB memory heap is disabled
110524 10:52:20 InnoDB: Mutexes and rw_locks use GCC atomic builtins
110524 10:52:20 InnoDB: Compressed tables use zlib 1.2.3
110524 10:52:20 InnoDB: Initializing buffer pool, size = 128.0M
110524 10:52:20 InnoDB: Completed initialization of buffer pool
InnoDB: Unable to lock ./ibdata1, error: 11
InnoDB: Check that you do not already have another mysqld process
InnoDB: using the same InnoDB data or log files.
110524 10:52:20  InnoDB: Retrying to lock the first data file
InnoDB: Unable to lock ./ibdata1, error: 11
InnoDB: Check that you do not already have another mysqld process
InnoDB: using the same InnoDB data or log files.
```

これは LifeKeeper によって制 御 されている NFS ファイルシステム上 で、MySQL `mysqld` プロセスが *ibdata1* ファ イルを NFS lock に設 定 しているということを示 しています。システムクラッシュによりロックがクリアされなかったの で、LifeKeeper は MySQL インスタンスを In Service の状 態 に戻 すことができません。MySQL は何 か別 のプロセ スが *ibdata1* ファイルを使 用 しているとみなします。

解 決 方 法

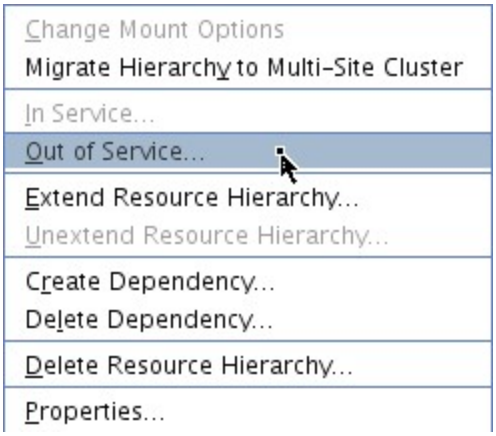
この事 象 を解 決 するためには、ファイルシステムリソースを作 成 する前 に「nolock」とよばれる NFS のオプションで *ibdata1* を格 納 する予 定 の NFS ファイルシステムをマウントしてください。デフォルトでは、NFS はファイルをロックさ

せるよう設定しています。「nolock」オプションがリソース作成前に使用されると、LifeKeeper はこのオプションを認識し、ファイルシステムリソースを In Service の状態にするたびにこのオプションを使用します。LifeKeeper は (クラスタノードから) ibdata1 を含んだファイルシステムへのアクセスを制御するので、通常、ロックは重要ではありません。テスト中に使用される NFS マウントオプションは「`rw,sync,tcp,nfsvers=3,nolock`」です。

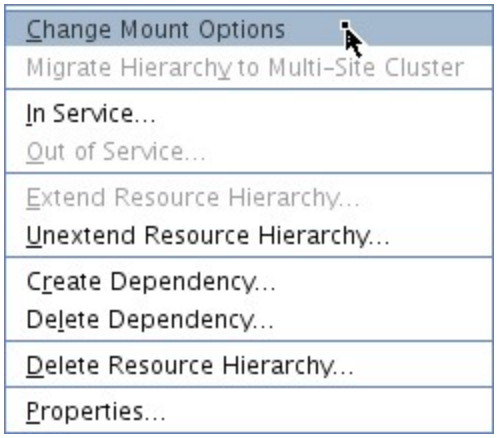
MySQL のバイナリが格納されているファイルシステムといったような MySQL リソース階層によって使用されるその他のファイルシステム上で「nolock」を使用する必要はありません。

NAS ファイルシステムリソースがすでに「nolock」オプション設定なしに作成されている場合は以下の手順に従ってマウントオプションを変更してください。

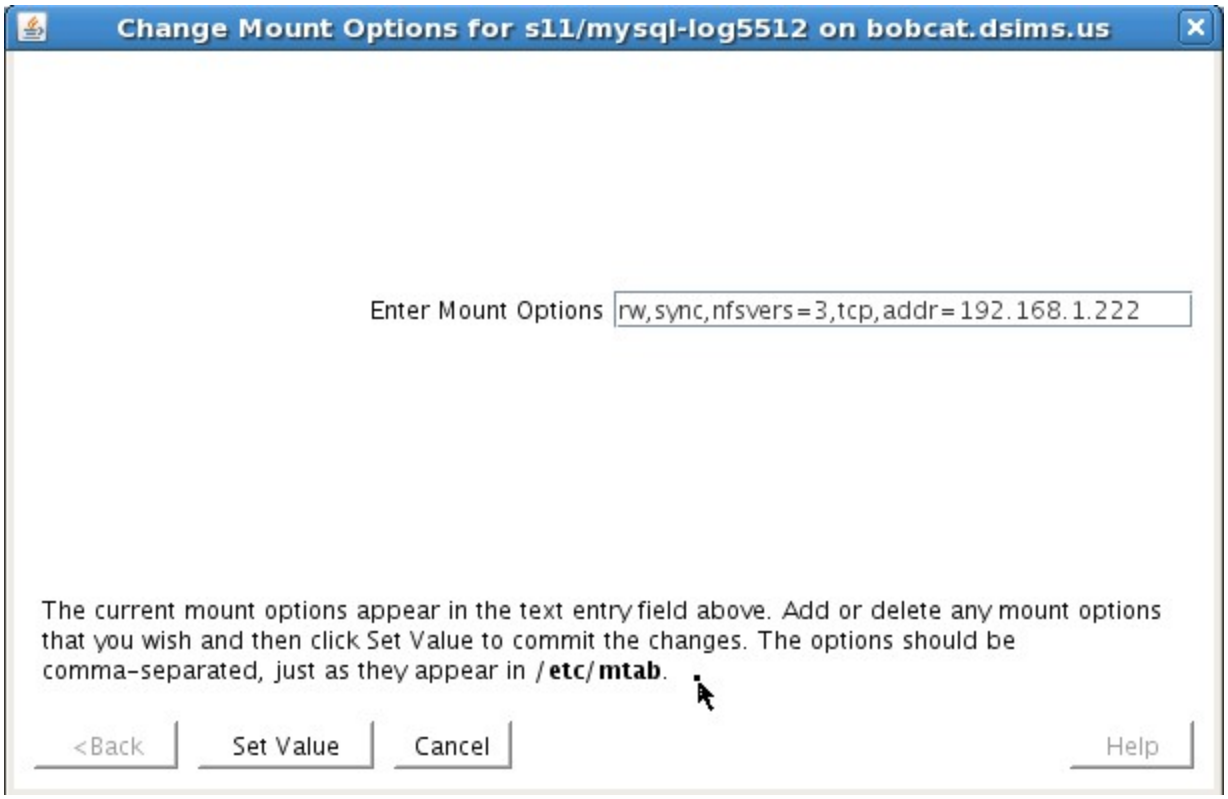
1. LifeKeeper GUI を使用して変更する必要があるファイルシステムリソースに対して Out of Service 操作を行ってください。この操作は LifeKeeper GUI でファイルシステムリソース上にポインタを置き、右クリックしてドロップダウンメニューから **[Out of Service]** を選択することで可能です。この操作は親リソースも同じく Out of Service 状態にします。



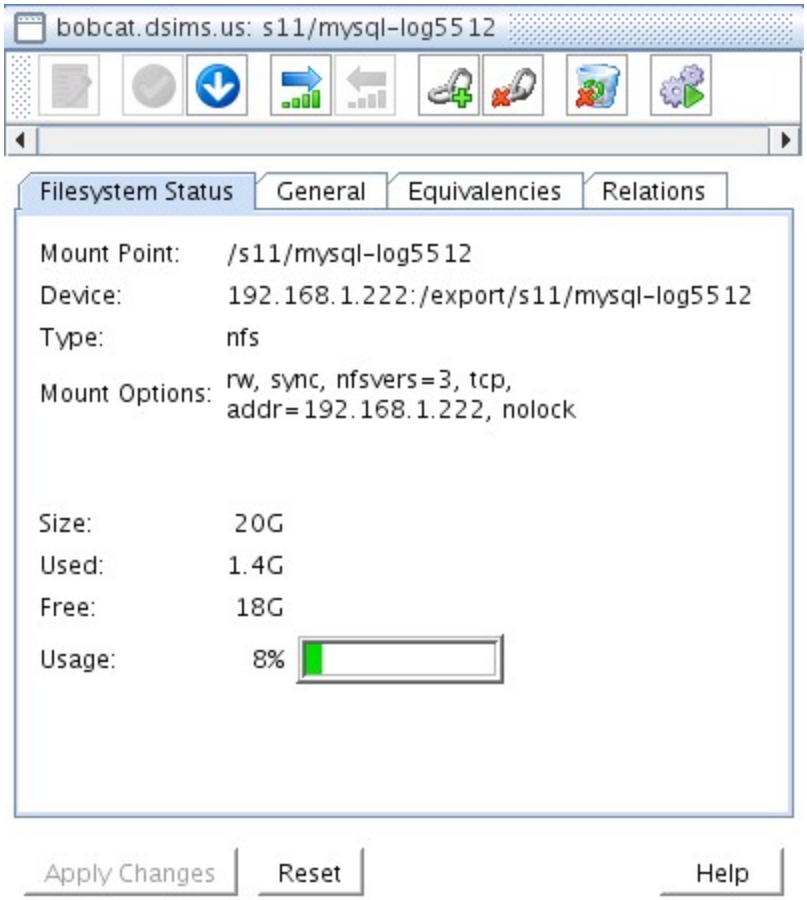
2. **[Out of Service]** 操作を実行し、プロセスを完了させてください。
3. 一旦ファイルシステムリソースが Out of Service 状態になったら、ポインタをリソース上に置き右クリックし、ドロップダウンメニューから **[Change Mount Options]** を選択してください。



4. ポップアップウィンドウで、オプションの行に**「nolock」**を追加し、**[Set Value]** をクリックしてください。クラスタの各ノードで手順 3 および 4 を繰り返す必要があります。



5. マウスを右クリックし、**[In Service]** を選択し、NAS ファイルシステムリソースを In Service の状態に戻してください。
6. ファイルシステムリソースのプロパティパネルが、「nolock」が現在のマウントオプションであることを表示します。



systemd 環境で使用する場合の考慮事項

Systemd を採用しているディストリビューションで MySQL(5.7.6以降) が Systemd を使用するように設定されている場合、従来の mysqld_safe コマンドや mysqld_multi コマンドが使えません。これらのコマンドが使えない場合、LifeKeeper は systemctl コマンドを使って MySQL サービスの起動・停止を行います。従って、systemctl コマンドで起動・停止できる必要があります。

Managing MySQL Server with systemd を参考にして、MySQL の設定を行ってください。特に PID File の設定は、Systemd も LifeKeeper も必要とします。必ず Systemd の MySQL 設定と my.cnf の設定を同一にしてください。

リソース作成時は、以下の設定項目に注意してください。

設定項目	設定値
Location of my.cnf	systemctl コマンドで MySQL サービスを起動する際に使用される my.cnf が格納されているディレクトリの絶対パス
Location of MySQL executables	mysqladmin コマンドがインストールされているディレクトリの絶対パス

注記: "Location of my.cnf" の設定は、LifeKeeper 内部で my.cnf の設定を読み出すために使用されます。ここで設定した値が systemctl コマンドの起動で使われるわけではありません。デフォルトと異なるパスにある my.cnf を使う場合は、Systemd の MySQL 設定を正しく設定し、期待通り MySQL サービスが起動・停止できることを確認してください。
また、include 文はサポートしていません。単一の my.cnf に全ての設定を記述する必要があります。

Chapter 3: インストール

LifeKeeper での MySQL のインストール / 設定

LifeKeeper の設定作業

設定作業は LifeKeeper GUI を使用して実行できます。以下の 4 つの作業は、MySQL リソースインスタンス特有のものであり、Recovery Kit ごとに異なるため、本セクションで説明しています。

- リソース階層の作成。アプリケーションリソース階層を LifeKeeper クラスタに作成します。
- リソース階層の削除。リソース階層を LifeKeeper クラスタ内のすべてのサーバから削除します。
- リソース階層の拡張。リソース階層をプライマリサーバからバックアップサーバへ拡張します。
- リソース階層の拡張解除。リソース階層を LifeKeeper クラスタ内の 1 つのサーバから拡張解除 (削除) します。

以下の作業については、すべての Recovery Kit で同じ手順を使用する共通の作業であるため、SPS for Linux テクニカルドキュメンテーションの管理 セクションを参照してください。

- リソース依存関係の作成。既存のリソース階層と別のリソースインスタンスとの間に親子の依存関係を作成し、クラスタ内のすべての対象サーバに依存関係の変更を反映します。
- リソース依存関係の削除。リソースの依存関係を削除して、クラスタ内のすべての対象サーバに依存関係の変更を反映します。
- In Service リソース階層を特定のサーバで In Service の状態にします。
- Out of service 。リソース階層を特定のサーバで Out of Service の状態にします。
- 表示 / 編集。特定のサーバでリソース階層のプロパティを表示または編集します。

注記 :このセクションの残りの部分では、LifeKeeper GUI の **[Edit]** メニューから作業を選択することによって、Recovery Kit を設定する方法を説明します。設定作業はツールバーから選択することもできます。状況表示ウィンドウの **[Resource Hierarchy Tree]** (左側のペイン)でグローバルリソースを右クリックし、ドロップダウンメニューに **[Edit]** メニューと同じ選択項目を表示することもできます。

状況表示ウィンドウの **[Resource Hierarchy Table]** (右側のペイン)でリソース・インスタンスを右クリックして、サーバおよび特定リソースの状態に応じて、リソース階層の作成を除くすべての設定作業を実行することもできます。

MySQL リソース階層の作成

重要:

MySQL データディレクトリファイル(datadir)が共有ディスク上にある LifeKeeper クラスタ環境では、共有ファイルシステムが primary/template サーバにマウントされる必要があります。ファイルシステムリソースが最初に作成される場合、共有ファイルシステムは必ず各サーバの同じマウントポイントにマウントします。また、重要事項として、リソースを作成するには稼働しているコミュニケーションパス(すなわちハートビート)が必須であることを念頭においてください。MySQL データディレクトリは、共有ディスク、レプリケーションされたディスク、もしくは Network Attached Storage (NAS) 上に存在することが可能です。

プライマリサーバからリソースインスタンスを作成するには、以下の手順を実行します。

1. LifeKeeper GUI メニューから **[Edit]** を選択し、次に **[Server]** を選択します。ドロップダウンメニューから、**[Create Resource Hierarchy]** を選択します。

   すでに行った選択を変更する場合、または MySQL リソース階層を作成するステップのいずれかでエラーメッセージが表示された場合、通常は 1 つ前に戻って、選択を変更するか訂正することができます。([Back] ボタンが有効な場合)。

   重要: リソースの作成時には、保護対象の MySQL データベースサーバデーモン (mysqld) が稼働していなければなりません。

   ダイアログボックスが表示され、ドロップダウンメニューに、クラスタにインストールされ認識されているすべての Recovery Kit が表示されます。ドロップダウンメニューから **[MySQL Database]** を選択します。

   [Next] をクリックします。

   階層作成の途中で **[Cancel]** ボタンをクリックすると、作成処理全体が取り消されます。

2. **[Switchback Type]** を選択します。この選択によって、バックアップサーバへのフェイルオーバの後、MySQL インスタンスが In Service に戻るときに、このサーバにどのようにスイッチバックされるのかが決まります。***[intelligent]*** または ***[automatic]*** を選択できます。***[intelligent]*** の場合、インスタンスをプライマリ / オリジナルサーバにスイッチバックするときに管理者の介入が必要になります。***[automatic]*** の場合、プライマリサーバがオンラインに戻り、LifeKeeper コミュニケーションパスを再確立した直後に自動的にスイッチバックが行われます。

スイッチバックタイプは、必要な場合 **[Resource Properties]** ダイアログボックスの **[General]** タブで後から変更できます。

[Next] をクリックします。

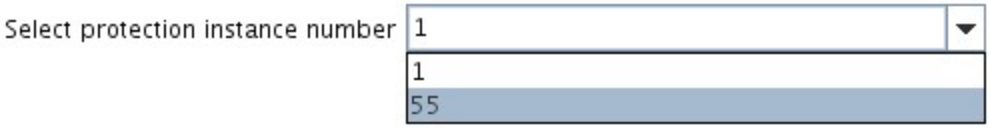
3. **[Server]** で、MySQL データベースインスタンスを配置するサーバ(通常これをプライマリサーバまたはテンプレートサーバと呼ぶ)を選択します。ドロップダウンメニューには、クラスタ内の全サーバが表示されます。

[Next] をクリックして次のダイアログボックスに進んでください。

4. **[Location of my.cnf]** を選択または入力します。これは MySQL 設定ファイル (my.cnf) が存在する場所のフルパス名 (ファイル名は除く) です。

[Next] をクリックして次のダイアログボックスに進んでください。

5. mysqld_multi 形式の my.cnf ファイルを保有している場合は **[Protection Instance Number]** を選択してください。従来の形式の my.cnf ファイルを保有している場合は、この画面は表示されません。



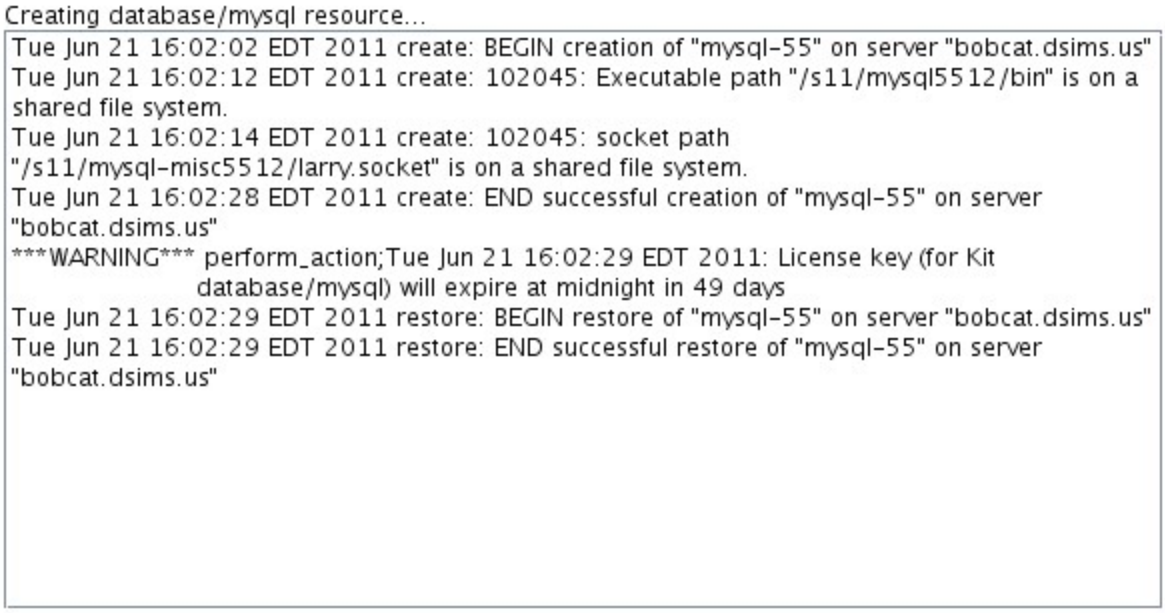
6. **[Location of MySQL executables]** の場所を選択または入力します。これは、MySQL データベースサーバデーモンの開始と監視に使用されるバイナリのフルパス名です。

注記:この時点で、入力したデータがMySQLリソース階層の作成に妥当かどうかが検査されます。この検査で問題が検出された場合、エラーが画面に表示されます。ディレクトリのパスは妥当で、MySQL設定自体にエラーがある場合は、ここでそのエラーを訂正してから階層の作成を続けることができます。

[Next]をクリックして次のダイアログボックスに進んでください。

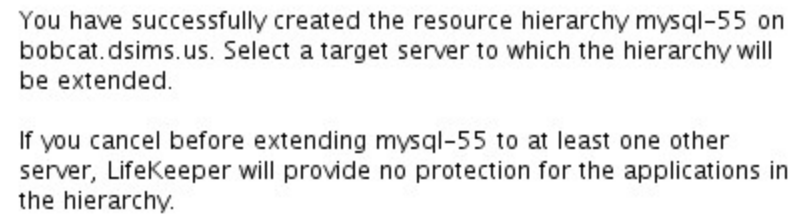
7. **[Database Tag]**を選択または入力します。これはMySQL階層につけるタグ名です。デフォルトを選択するか、独自のタグ名を入力することができます。

[Create]をクリックすると、**[Create Resource Wizard]**によりMySQLリソースが作成されます。



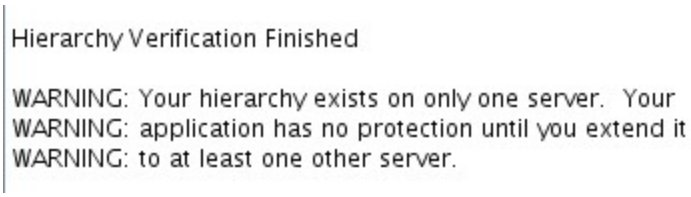
注記:この時点で、MySQLリソース階層が正常に作成されます。

8. 別の情報ボックスが表示され、MySQLリソース階層が正常に作成されたこと、MySQLリソース階層をLifeKeeperの保護下に置くには、その階層をクラスタ内の別のサーバに拡張する必要があることが示されます。



[Continue] をクリックすると、本セクションで後述する **[Pre-Extend Wizard]** が起動されます。

ここで **[Cancel]** をクリックすると、別のダイアログボックスが表示され、いずれかの時点でここに戻り、MySQL リソース階層を別のサーバに拡張して、LifeKeeper の保護下に置く必要があることが警告されます。



9. **[Done]** をクリックして終了します。

リソース階層の削除

LifeKeeper 環境内のすべてのサーバからリソース階層を削除するには、以下の手順を実行します。

1. LifeKeeper GUI メニューから **[Edit]** を選択し、次に **[Resouce]** を選択します。ドロップダウンメニューから、**[Delete Resource Hierarchy]** を選択します。
2. **[Target Server]** で MySQL リソース階層を削除するターゲットサーバの名前を選択します。

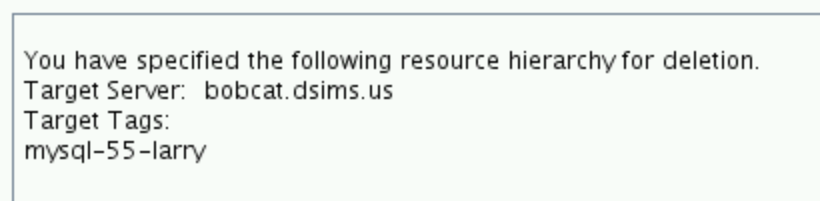
   注記:右側ペインで個々のリソースインスタンスを右クリックするか、リソースが 1 台のサーバ上のみにある場合に左側ペインでグローバルリソースを右クリックして **[Delete Resource]** 作業を選択した場合、このダイアログボックスは表示されません。

   [Next] をクリックします。
3. **[Hierarchy to Delete]** を選択します。削除するリソース階層を特定して、強調表示にします。

注記:左側ペインのグローバルリソースまたは右側ペインの個々のリソースインスタンスを右クリックして**[Delete Resource]**作業を選択した場合、このダイアログボックスは表示されません。

[Next]をクリックします。

4. 選択したターゲットサーバと、削除の対象として選択した階層を確認する情報ボックスが表示されます。



[Delete]をクリックします。

5. MySQLリソースが正常に削除されたことを確認する別の情報ボックスが表示されます。

6. **[Done]** をクリックして終了します。

リソース階層の拡張

階層の作成後、クラスタ内の別のサーバに拡張する必要があります。3通りのシナリオで、テンプレートサーバからターゲットサーバにリソースインスタンスを拡張できます。最初のシナリオは、リソース作成後、**[Continue]** をクリックして、別のサーバにリソースを拡張する場合です。2番目のシナリオは、以下に示すように **[Edit]** メニューから **[Extend Resource Hierarchy]** を選択する場合です。3番目のシナリオは、左側または右側のペインから拡張されていない階層を右クリックする場合です。どのシナリオでも同じダイアログボックスが表示されます(いくつかの例外については、以下に詳細を明記する)。

1. LifeKeeperGUI メニューから **[Extend]** ウィザードを開始する場合は、**[Edit]** を選択し、次に **[Resource]** を選択してください。ドロップダウンメニューから **[Extend Resource Hierarchy]** を選択してください。これで **[Extend Resource Hierarchy]** ウィザードが起動されます。

2. 最初に表示されるダイアログボックスで、MySQL リソース階層が現在 in service の **[Template Server]** を選択してください。ここで選択する **[Template Server]** と次のダイアログボックスで選択する **[Tag to Extend]** によって、*in service* のリソース階層が表されることを認識しておくことが重要です選択したテンプレートサーバでサービス中でないリソースタグを選択すると、エラーメッセージが表示されます。このダイアログのドロップダウンボックスには、クラスタ内の全サーバの名前が表示されます。

   注記: MySQL リソース階層作成後、直ちに **[Extend Resource Hierarchy]** 作業に入った場合は、作成段階でテンプレートサーバが特定されているため、このダイアログボックスは表示されません。これは、

GUI ウィンドウの左側ペインの MySQL リソースアイコンまたは右側ペインの MySQL リソースボックスを右クリックして、**[Extend Resource Hierarchy]** を選択した場合も同様です。

階層を拡張する手順の間に **[Cancel]** をクリックすると、どの時点であってもそのサーバへの拡張処理がキャンセルされますので注意してください。ただし、すでにリソースを別のサーバに拡張している場合は、明示的に拡張解除するまで、そのインスタンスの拡張は有効です。

たとえば、Server1 にリソースを作成し、それを Server2 に拡張しているとします。同じリソースを Server3 に拡張している最中に、気が変わっていずれかのダイアログボックス内の **[Cancel]** をクリックしたとします。この場合は、Server3 へのリソース拡張だけがキャンセルされ、Server2 への拡張はキャンセルされません。この階層から Server2 を削除したい場合は、Server2 からリソースを拡張解除する必要があります。

[Next] をクリックして次のダイアログボックスに進んでください。

3. **[Tag to Extend]** を選択してください。これは、テンプレートサーバからターゲットサーバに拡張する MySQL インスタンスの名前です。ウィザードのドロップダウンメニューには、前述のダイアログボックスで選択したテンプレートサーバ上に作成されたすべてのリソースが表示されます。

   注記: ここでも、MySQL リソース階層作成後、ただちに [Extend Resource Hierarchy] 作業に入った場合は、作成段階で MySQL リソースのタグ名が特定されているため、このダイアログボックスは表示されません。GUI ウィンドウの左側ペインの MySQL リソースアイコンまたは右側ペインの MySQL リソースボックスを右クリックして、**[Extend Resource Hierarchy]** を選択した場合も同様です。

   [Next] をクリックしてください。

4. MySQL リソース階層を拡張する **[Target Server]** を選択してください。ドロップダウンボックスには、選択された階層に入っていないクラスタ内のサーバ名の一覧が表示されます。

   [Next] をクリックしてください。

5. **[Switchback Type]** を選択してください。この選択によって、バックアップサーバへのフェイルオーバの後、MySQL インスタンスがサービス中に戻るときに、このサーバにどのようにスイッチバックされるのかが決まります。***[intelligent]*** または ***[automatic]*** を選択できます。[intelligent] の場合、インスタンスをプライマリ / オリジナルサーバにスイッチバックするときに管理者の介入が必要になります。[automatic] の場合、プライマリ

サーバがオンラインに戻り、LifeKeeperコミュニケーションパスを再確立した直後に自動的にスイッチバックが行われます。

スイッチバックタイプは、必要な場合 **[Resource Properties]** ダイアログボックスの **[General]** タブで後から変更できます。

[Next] をクリックしてください。

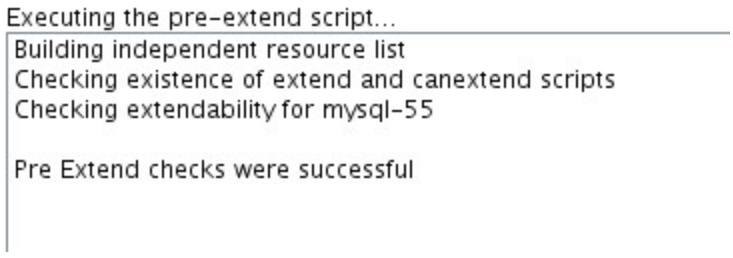
6. **[Template Priority]** を選択または入力してください。これはサーバで現在 in service の MySQL 階層の優先順位です。優先順位は、1 ～ 999 の範囲で未使用の値が有効で、小さい数字ほど優先順位が高くなります(数字 1 が最高の優先順位に相当します)。拡張処理時に、別のシステムですでに使用中の優先順位をこの階層に対して指定することはできません。デフォルト値を推奨します。**注記:** このフィールドは階層を最初に拡張するときだけ表示されます。

   [Next] をクリックします。

7. **[Target Priority]** を選択または入力します。これは、別サーバにある同等の階層に対する新しく拡張する MySQL 階層の優先順位です。1 ～ 999 の範囲で、まだ優先順位として使用されていない値が有効で、リソースのカスケーディングフェイルオーバシーケンスにおけるサーバの優先順位を示します。数値が小さいほど優先順位は高くなります(1 は最高の優先順位を表す)。LifeKeeper のデフォルトでは、階層が作成されたサーバに「1」が割り当てられることに注意してください。優先順位は連続している必要はありませんが、特定のリソースについて 2 つのサーバに同じ優先順位を割り当てることはできません。

   [Next] をクリックしてください。

8. 情報ボックスが表示され、LifeKeeper が環境のチェックを正常に完了し、この MySQL リソースを拡張するためのすべての要件が満たされていることが示されます。満たされていない要件がある場合は、**[Next]** ボタンは選択できなくなり、**[Back]** ボタンが有効になります。



[Back] をクリックした場合、情報ボックスに表示されるエラーメッセージの内容に従って、リソースの拡張を変更できます。

ここで **[Cancel]** をクリックした場合は、別の時点でここに戻り、MySQL リソース階層を他のサーバに拡張して、LifeKeeper の保護下に置く必要があります。

[Next] をクリックすると、**[Extend Resource Hierarchy]** 設定作業に入ります。

9. このダイアログボックスは情報の表示専用です。このボックス内に表示される **[Location of my.cnf]** は変更できません。MySQL インスタンスは設定ファイルから場所情報を取得しています。

[Next] をクリックしてください。

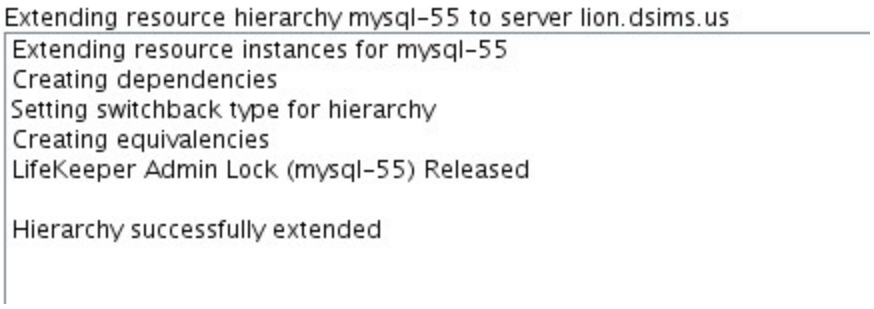
10. **[Location of MySQL executables]** を選択または入力してください。これは、MySQL データベースサーバデーモンの開始と監視に使用されるバイナリのフルパス名です。

[Next] をクリックしてください。

11. **[Database Tag]** を選択または入力してください。これは MySQL 階層につけるタグ名です。デフォルトを選択するか、独自のタグ名を入力することができます。

[Extend] をクリックします。

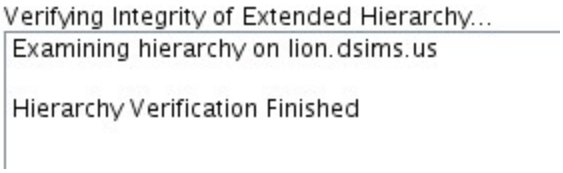
12. 拡張が実行されていることを示す情報ボックスが表示されます。



同じ MySQL リソースインスタンスをクラスタ内の別のサーバに拡張する場合は **[Next Server]** をクリックしてください。その場合は、**リソース階層の拡張**の操作が繰り返されます。

[Finish] をクリックすると、LifeKeeper は MySQL リソースの拡張が正常に完了したことを確認します。

13. **[Finish]** をクリックした場合、次の画面が表示されます。



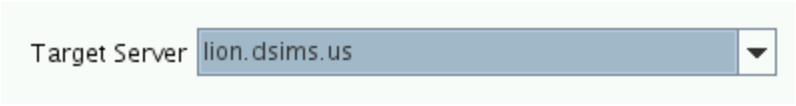
14. 最後のダイアログボックスで **[Done]** をクリックして終了してください。

注記: *必ず両方のサーバ*で新しいインスタンスの機能をテストしてください。

リソース階層の拡張解除

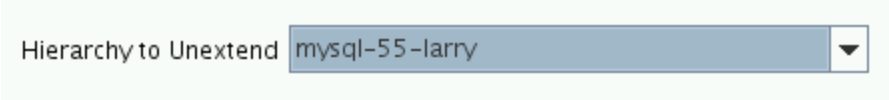
1. LifeKeeper GUI メニューから **[Edit]** を選択し、次に **[Resouce]** を選択します。ドロップダウンメニューから、**[Unextend Resource Hierarchy]** を選択します。
2. **[Target Server]** で、MySQL リソースを拡張解除するターゲットサーバを選択します。MySQL リソースが現在 In Service のサーバは選択できません。

   注記:右側のペインから個々のリソースインスタンスを右クリックして **[Unextend]** 作業を選択した場合、このダイアログボックスは表示されません。

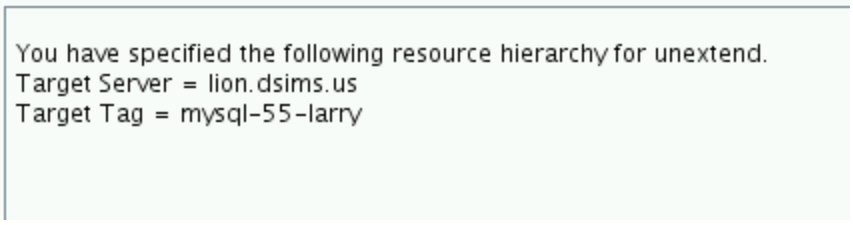


   [Next] をクリックします。
3. **[Hierarchy to Unextend]** で、拡張解除する MySQL 階層を選択してください。

   注記:左側のペインからグローバルリソースを右クリックするか、右側のペインから個々のリソースインスタンスを右クリックして **[Unextend]** 作業を選択した場合、このダイアログボックスは表示されません。

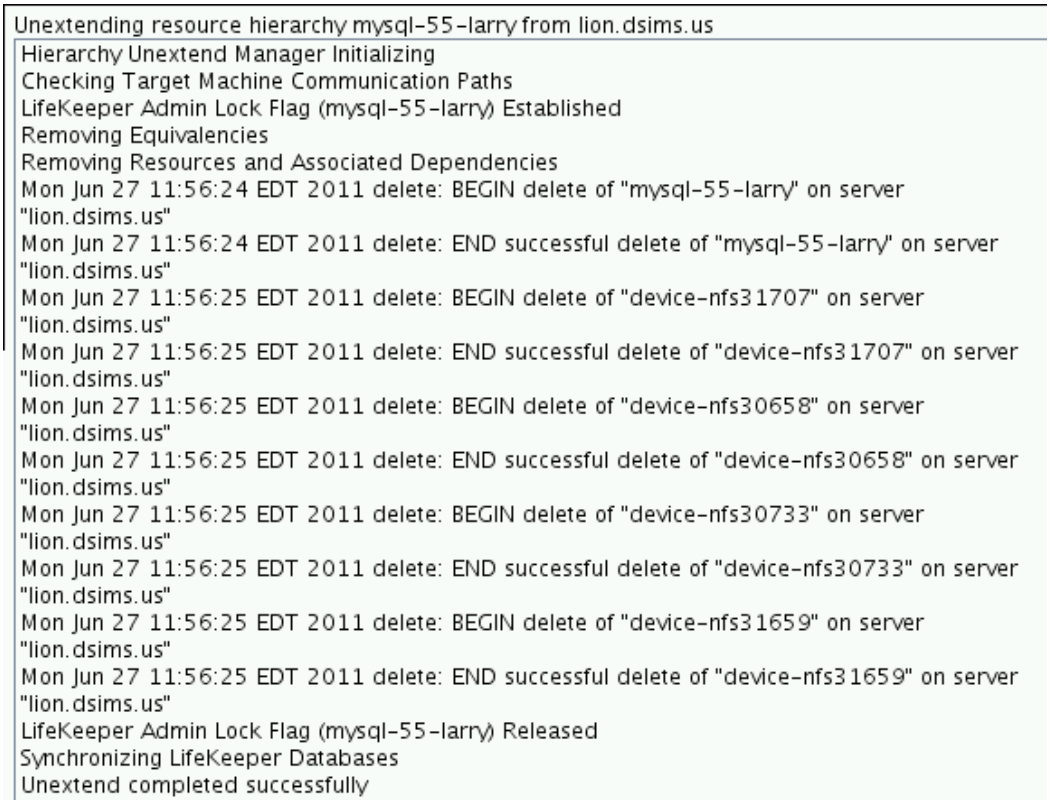


   [Next] をクリックします。
4. 拡張解除の対象として選択したターゲットサーバと MySQL リソース階層を確認する情報ボックスが表示されます。



[Unextend] をクリックします。

5. MySQL リソースが正常に拡張解除されたことを確認する別の情報ボックスが表示されます。



6. **[Done]** をクリックして終了します。

リソース階層のテスト

手動スイッチオーバを開始することによって、MySQL リソース階層をテストできます。これにより、プライマリサーバからバックアップサーバへのリソースインスタンスのフェイルオーバをシミュレートします。

GUI による手動スイッチオーバの実行

GUI による手動スイッチオーバの実行

手動スイッチオーバを開始することによって、MySQL リソース階層をテストできます。これにより、プライマリサーバからバックアップサーバへのリソースインスタンスのフェイルオーバをシミュレートします。

GUI による手動スイッチオーバの実行

LifeKeeper GUI で、ドロップダウンメニューから **[Edit]** > **[Resource]** > **[In Service]** を選択すると、手動スイッチオーバを開始できます。例えば、バックアップサーバで **[In Service]** 要求を実行すると、アプリケーション階層はバックアップサーバで In Service になり、プライマリサーバでは停止されます。この時点で、元のバックアップサーバがプライマリサーバに、元のプライマリサーバがバックアップサーバに変わります。

[Out-of-Service] 要求を実行すると、アプリケーションは他のサーバで In Service にならずに、out of service になります。

LifeKeeper では、ロールバックやバックアップアーカイブのような内部処理の調整や制御は行いません。テープアーカイブやリストアの作業は、アプリケーション管理者の責任です。

リカバリ動作

プライマリサーバに障害が発生すると、MySQL Recovery Kit ソフトウェアは以下の処理を実行します。

- ファイルシステム (共有もしくは複製) をバックアップサーバ上にマウントします。
- MySQL に関連するデーモンプロセスを開始します。

Chapter 5: トラブルシューティング

共通のエラーメッセージ

本セクションでは、SPS MySQL リソース階層の作成時や拡張時、またはリソースの削除時やリストア時に表示される可能性のあるメッセージの一覧を示します。必要に応じて、エラーの原因およびエラー状態を解消するために必要な処置についても説明しています。

他の SPS コンポーネントからメッセージが出されることもあります。そのような場合は、メッセージカタログ (場所は、SIOS テクニカルドキュメンテーション Web サイトの「エラーコードの検索」の下) を参照してください。メッセージカタログには、操作、管理、GUI など、SIOS Protection Suite for Linux の使用中に遭遇する可能性のあるすべてのエラーコードが列挙されています。また、エラーコードの原因に関する追加の説明や、問題解決のために必要な処置についても、必要に応じて記載されています。この完全なリストを検索すると、受信したエラーコードを見つけることができます。また、関連する SPS コンポーネントの個別のメッセージカタログに直接アクセスすることもできます。

MySQL 固有のエラーメッセージ

注記: エラーメッセージ列で、二重引用符で囲まれたすべて大文字の語はサーバ上のリソースの名前を示します (例えば、"SERVER" は実際には「Server1」という名前のサーバである場合があります)。

エラー番号	エラーメッセージ
102001	Usage: “SCRIPT NAME” sysname dbvarname cnfpath exepath instance
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 **処置:** my.cnf 設定ファイルの「mysqld」セクションには、「socket」のエントリがなければなりません。
102005	Unable to obtain a valid value for the “port” in “PATH”/my.cnf **処置:** my.cnf 設定ファイルの「mysqld」セクションには、「port」のエントリがなければなりません。
102006	Unable to obtain the data directory location "PATH" **処置:** データベースが指定したソケットとポートを使用して稼働していることを確認してください。
102007	Must specify the absolute path to the my.cnf configuration file
102008	Must specify the absolute path to the MySQL executables

エラー番号	エラーメッセージ
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” for directive "DIRECTIVE" 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 **処置:** my.cnf 設定ファイルの「client」セクションには、「user」変数の有効なエントリがなければなりません。
102029	Unable to obtain a valid value for the “password“ variable in “PATH”/my.cnf **処置:** my.cnf 設定ファイルの「client」セクションには、「password」変数の有効なエントリがなければなりません。
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 **処置:** my.cnf 設定ファイルの mysqld セクションには、「pid-file」変数のエントリがなければなりません。
102033	Unable to obtain a valid value for the "user" variable in “PATH”/my.cnf **処置:** my.cnf 設定ファイルの「mysqld」セクションには、「user」変数を使用して OS ユーザを指定する必要があります。

エラー番号	エラーメッセージ
102034	警告: A my.cnf file exists at "PATH", which may override the values specified in the file at "PATH"/my.cnf.
102035	The mysql system user "USER" does not exist on target server "SERVER"
102036	The mysql system user "USER" uids are different on target server "SERVER1" and template server "SERVER2"
102037	The mysql system user "USER" gids are different on target server "SERVER1" and template server "SERVER2"
102038	LifeKeeper was unable to stop the MySQL database server using a graceful shutdown.Issuing kill for pid(s):"PROCESS ID LIST".
102039	LifeKeeper will ignore failed connection as possible max connections error, due to existence of process pid "PROCESS ID".
102040	The mysql action for resource tag :TAG" returned:"COMMAND OUTPUT".
102041	LifeKeeper was unable to start the MySQL database server using the defaults-file option.Retrying with individual options.
102042	The LifeKeeper "ACTION" action detected the flag "FLAG", and will exit.
102043	END of "ACTION" action on due to a(n) "SIGNAL" signal.
102044	The file my.cnf does not exist in the stored path "PATH".
102045	"DIRECTIVE" path "PATH" is on a shared filesystem.
102046	Starting mysqld daemon with databases from "PATH".