



# **LifeKeeper Single Server Protection**

**v8.2.0**

**Technical Documentation**

**October 2013**

This document and the information herein is the property of SIOS Technology Corp. (previously known as SteelEye® Technology, Inc.) and all unauthorized use and reproduction is prohibited. SIOS Technology Corp. makes no warranties with respect to the contents of this document and reserves the right to revise this publication and make changes to the products described herein without prior notification. It is the policy of SIOS Technology Corp. to improve products as new technology, components and software become available. SIOS Technology Corp., therefore, reserves the right to change specifications without prior notice.

LifeKeeper, SteelEye and SteelEye DataKeeper are registered trademarks of SIOS Technology Corp.

Other brand and product names used herein are for identification purposes only and may be trademarks of their respective companies.

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

Address correspondence to:  
[ip@us.sios.com](mailto:ip@us.sios.com)

Copyright © 2013  
By SIOS Technology Corp.  
San Mateo, CA U.S.A.  
All rights reserved

# Table of Contents

---

LifeKeeper Single Server Protection Core .....	1
LifeKeeper Single Server Protection Core ソフトウェア .....	2
File System、Generic Application、および IP の Recovery Kit ソフトウェア .....	2
LifeKeeper GUI ソフトウェア .....	3
VMware HA との連携 .....	3
SteelEye 管理コンソール .....	3
インストールの概要 .....	4
システム要件 .....	4
セットアップの前提条件 .....	5
セットアップの実行 .....	5
ソフトウェアのインストール .....	5
ベースの vCenter と認証情報の設定 .....	6
認証情報の考慮事項 .....	6
SteelEye LifeKeeper Single Server Protection vSphere Client プラグイン .....	6
プラグインの要件 .....	7
vSphere Client プラグインの設定 .....	7
vSphere Client プラグインの登録 .....	7
vSphere Client プラグインの登録解除 .....	7
vSphere Client ユーザーインターフェース .....	8
[SteelEye LifeKeeper Single Server Protection] タブ .....	8
コンテキストメニュー .....	8
Datacenter Level .....	8
ESX or ESXi Level .....	9
Virtual Machine Level .....	9
Manage Plug-Ins .....	9

---

その他の表示 .....	10
認証情報の設定 .....	10
認証情報の追加または変更 .....	10
ストア内の認証情報のリスト表示 .....	11
サーバの認証情報の削除 .....	11
追加情報 .....	11
インストールの検証 .....	11
トラブルシューティング .....	11
vSphere Client プラグインのセキュリティ警告 への対処 .....	11
カスタム証明書の使用 .....	12
証明書の使用方法 .....	12
独自の証明書の使用 .....	13
Application Recovery Kit .....	13
リソース階層 .....	14
リソースタイプ .....	14
リソースの状態 .....	15
階層の関係 .....	15
ステータスの詳細表示 .....	16
リソース階層の情報 .....	17
LifeKeeper Single Server Protection ソフトウェアのインストール .....	19
LifeKeeper Single Server Protection ソフトウェアのインストール .....	20
LifeKeeper Single Server Protection のインストールの検証 .....	22
LifeKeeper Single Server Protection の管理の概要 .....	23
LifeKeeper Single Server Protection Administration Overview .....	23
Starting LifeKeeper Single Server Protection .....	24
Starting LifeKeeper Single Server Protection Processes .....	24
Enabling Automatic LifeKeeper Single Server Protection Restart .....	24
Stopping LifeKeeper Single Server Protection .....	24
Disabling Automatic LifeKeeper Single Server Protection Restart .....	25
Viewing LifeKeeper Single Server Protection Processes .....	25

Viewing LifeKeeper Single Server Protection GUI Server Processes .....	26
Viewing LifeKeeper Single Server Protection Controlling Processes .....	27
VMware HA と LifeKeeper Single Server Protection の連携を有効にする .....	28
ログファイルのサイズを増やす .....	28
VMware HA を有効化した障害検出およびリカバリシナリオ .....	29
VMware HA と通知のみモード .....	29
LifeKeeper Single Server Protection ハートビートと VMware HA .....	30
LifeKeeper Single Server Protection で保護するシステムのメンテナンス .....	31
Creating Resource Hierarchies .....	31
LifeKeeper Single Server Protection Application Resource Hierarchies .....	32
Recovery Kit Options .....	32
Creating a File System Resource Hierarchy .....	32
Creating a Generic Application Resource Hierarchy .....	33
Editing Resource Properties .....	34
Creating a Resource Dependency .....	34
Deleting a Resource Dependency .....	35
Deleting a Hierarchy .....	35
LifeKeeper Single Server Protection のアンインストール .....	35
<b>Chapter 4: ユーザガイド .....</b>	<b>37</b>
LifeKeeper GUI .....	37
LifeKeeper グラフィカルユーザインターフェース .....	37
GUI の概要 - 全般 .....	37
GUI サーバ .....	37
GUI クライアント .....	38
GUI クライアントの終了 .....	38
ステータスの表 .....	38
プロパティパネル .....	39
出力パネル .....	39
メッセージバー .....	39
GUI の終了 .....	40

---

メニュー .....	40
リソースのコンテキストメニュー .....	41
サーバのコンテキストメニュー .....	42
[File] メニュー .....	42
[Edit] メニュー - リソース .....	43
[Edit] メニュー - サーバ .....	43
[View] メニュー .....	44
[View Options] ダイアログ .....	44
[Help] メニュー .....	45
ツールバー .....	45
GUI ツールバー .....	46
リソースのコンテキストツールバー .....	46
サーバのコンテキストツールバー .....	47
LifeKeeper GUI のメッセージ履歴 .....	47
GUI の実行の準備 .....	49
LifeKeeper GUI ソフトウェアパッケージ .....	49
LifeKeeper Single Server Protection GUI の設定 .....	49
GUI 管理用の LifeKeeper Single Server Protection サーバの設定 .....	49
GUI の実行 .....	50
GUI の制限 .....	51
GUI サーバの開始および停止 .....	51
LifeKeeper GUI サーバを開始するには .....	51
トラブルシューティング .....	51
LifeKeeper GUI サーバを停止するには .....	51
LifeKeeper GUI サーバのプロセス .....	52
GUI ユーザの設定 .....	52
Java のセキュリティポリシー .....	53
ポリシーファイルの場所 .....	53
ポリシーファイルの作成と管理 .....	54
ポリシーファイルでの権限の付与 .....	54

---

ポリシーファイルの例 .....	55
Java プラグイン .....	55
Java プラグインのダウンロード .....	56
Java プラグインのトラブルシューティング .....	56
GUI アプレットのブラウザセキュリティパラメータの設定 .....	56
Netscape Navigator と Netscape Communicator .....	57
Firefox .....	57
Internet Explorer .....	57
LifeKeeper Single Server Protection サーバでの GUI の実行 .....	57
LifeKeeper Single Server Protection サーバでの GUI の実行 .....	58
リモートシステムでの GUI の実行 .....	58
リモートシステムでの GUI の設定 .....	58
リモートシステムでの GUI の実行 .....	59
アプレットのトラブルシューティング .....	59
共通の作業 .....	60
サーバからの切断 .....	60
接続サーバの表示 .....	61
サーバのステータスの表示 .....	61
サーバログファイルの表示 .....	62
[Log Viewer] ダイアログ .....	62
サーバプロパティの表示 .....	64
リソースタグおよび ID の表示 .....	64
リソースのステータスの表示 .....	64
サーバリソースのステータステーブル .....	64
リソースのプロパティの表示 .....	65
[Status] ウィンドウの表示オプションの設定 .....	66
Resource Labels .....	67
Resource Tree .....	67
Row Height .....	67
Column Width .....	68

---

メッセージ履歴の表示 .....	68
メッセージ履歴の解釈 .....	68
リソース階層ツリーの展開および折り畳み .....	69
[Resource Properties] ダイアログ .....	69
[General] タブ .....	70
[Relations] タブ .....	72
[Equivalencies] タブ .....	73
[Server Properties] ダイアログ .....	74
[General] タブ .....	74
[Resources] タブ .....	75
オペレータの作業 .....	76
リソースを In Service にする .....	76
リソースを Out of Service にする .....	77
高度な作業 .....	78
LCD .....	78
LifeKeeper 設定 データベース .....	78
LCD のディレクトリ構造 .....	78
/opt/LifeKeeper の LCD のディレクトリ構造 .....	78
LCD 設定データ .....	79
依存関係の情報 .....	79
リソースのステータス情報 .....	79
LCD のリソースタイプ .....	80
リソースのサブディレクトリ .....	80
リソースの動作 .....	81
LifeKeeper Single Server Protection のフラグ .....	81
LCDI のコマンド .....	81
階層の定義 .....	82
LCM .....	82
LifeKeeper Single Server Protection の警報とリカバリ .....	83
警報クラス .....	83

---

警報の処理 .....	84
警報 ディレクトリのレイアウト .....	84
リカバリスクリプト .....	84
アプリケーションインターフェースのレベル .....	84
インターフェースのレベル .....	85
依存関係の定義 .....	85
エラーの検出と処理 .....	85
リカバリ動作 .....	85
共通アプリケーションタイプのインターフェースの問題 .....	86
インターフェースの定義作業 .....	86
スクリプトの種類 .....	87
スクリプトのパラメータ .....	88
restore スクリプト .....	88
restore スクリプトの例 .....	89
remove スクリプト .....	92
remove スクリプトの例 .....	93
remove と restore スクリプトに共通のセクション .....	95
delete スクリプト .....	97
通知スクリプトの例 .....	98
ローカルリカバリスクリプト .....	98
メンテナンス作業 .....	98
リソース階層のメンテナンス .....	98
LifeKeeper Single Server Protection の設定値の変更 .....	98
ファイアウォールを使用した状態での LifeKeeper Single Server Protection の実行 .....	100
LifeKeeper GUI の接続 .....	100
LifeKeeper Single Server Protection の IP アドレスリソース .....	100
ファイアウォールの無効化 .....	100
ファイアウォール経由での LifeKeeper GUI の実行 .....	101
LifeKeeper Single Server Protection のアンインストール .....	102
<b>Chapter 5: FAQ .....</b>	<b>103</b>

---

SMC .....	103
質問 .....	103
回答 .....	103
<b>Chapter 6: トラブルシューティング .....</b>	<b>105</b>
既知の問題と回避策 .....	105
Core .....	105
GUI .....	106
IP .....	106
Oracle .....	107
SAP .....	108
SMC のトラブルシューティング .....	108

LifeKeeper Single Server Protection (SSP) は、単一ノード構成におけるアプリケーション監視を可能にします (つまり、クラスタの要件または制約はありません)。単一ノード環境は、物理的なものでも仮想 (vSphere、KVM) でも構いません。LifeKeeper SSP は、実績がある安定した SteelEye LifeKeeper アーキテクチャ上に構築されます。LifeKeeper SSP は優れたアプリケーション監視機能を提供し、障害が発生したアプリケーションおよびシステムインフラストラクチャ項目 (例: NFS 共有、IP アドレス、ファイルシステム) のリカバリを実行することができます。何らかの理由でアプリケーションをリカバリできない場合、LifeKeeper SSP は、システムのリブートまたは VM とアプリケーション監視を設定された VMware 仮想マシンの VMware HA 再起動によって、ノードの再起動を開始します。

**注記:** LifeKeeper SSP は SteelEye LifeKeeper 技術を使用して構築されているため、ドキュメント全体で LifeKeeper を参照します。また、両製品に共通するトピックについては SteelEye Protection Suite for Linux ドキュメンテーションの情報を参照します。これらの共通のトピックを参照する場合、LifeKeeper SSP には以下の話題は適用されません。

- クラスタリング
- コミュニケーションパス
- 共有ストレージ (要件、構成、...)
- リソース階層の拡張/拡張解除
- ストレージキット (DR、DMMP、HDLM、LVM、MD、PPATH)

**注記:** LifeKeeper SSP にベースとなるストレージキットがない場合、保護されるファイルシステムのマウントに必要なデバイスがシステム起動時にアクティベートされるようにするための手順が必要です (例: ファイルシステムが論理ボリューム上でマウントされる場合、LifeKeeper SSP が起動する前にボリュームがアクティブな状態になっていなければなりません)。

- リソース/マシンのフェイルオーバー (LifeKeeper SSP のデフォルトでは、これによってノードが再起動されます)
- リソースのスイッチオーバー
- 切り替え可能な IP アドレス (LifeKeeper SSP では、保護されるアプリケーションの一部には切り替え可能な IP アドレスが必要ですが、単一ノードしかないため、実際には切り替えは行われません)

LifeKeeper SSP のベースになっている SteelEye LifeKeeper 製品の詳細については、共通するリリース番号の [SteelEye Protection Suite for Linux ドキュメンテーション](#) を参照してください。このドキュメンテーションは、リソース階層の作成、リソースタイプ、状態と関係、グラフィカルユーザインターフェース (GUI)、および共通の作業と高度な作業に関する詳細情報を提供します。

## LifeKeeper Single Server Protection Core

LifeKeeper Single Server Protection Core は、以下の 3 つの主要コンポーネントで構成されています。

- Core ソフトウェア
- File System、Generic Application、および IP の Recovery Kit ソフトウェア
- GUI ソフトウェア

**注記:** LifeKeeper Single Server Protection は SteelEye LifeKeeper テクノロジーを使用して製造されているので、ドキュメンテーション全体を通じて LifeKeeper の説明があります。

## LifeKeeper Single Server Protection Core ソフトウェア

LifeKeeper Single Server Protection Core ソフトウェアは、以下のコンポーネントで構成されます。

- [LifeKeeper 構成データベース \(LCD\)](#) - LifeKeeper Single Server Protection が保護するリソースに関する情報が格納されています。リソースインスタンス、依存関係、イクイバレンス情報、リカバリ方向、LifeKeeper Single Server Protection の操作フラグなどの情報が格納されています。システムの起動後にデータが記憶されているように、データは共有メモリにキャッシュされ、ファイルに保存されます。
- [LCD インターフェース \(LCDI\)](#) - 構成データベース (LCD) にクエリを発行し、LCD 内のデータを取得して変更できます。また、リソースの状態や説明の情報を取得するために、Application Recovery Kit が LCDI を使用することもできます。
- [LifeKeeper 通信マネージャ \(LCM\)](#) - サーバのステータスを確認します。LifeKeeper Single Server Protection によるプロセス間のローカル通信にも使用されます。
- [LifeKeeper Single Server Protection アラームインターフェース](#) - LifeKeeper Single Server Protection アラームインターフェースは、イベントをトリガする基盤を提供します。LifeKeeper Single Server Protection で保護するリソースに障害が発生すると、アプリケーションのデーモンが sendevent プログラムを呼び出します。sendevent プログラムは LCD と通信を行い、リカバリに使用できるスクリプトがあるかどうかを判断します。
- LifeKeeper リカバリアクション制御インターフェース (LRACI) - リソースに実行する適切なリカバリスクリプトを決定し、適切な restore/remove スクリプトを起動します。

## File System、Generic Application、および IP の Recovery Kit ソフトウェア

LifeKeeper Single Server Protection Core は、サーバ上の指定リソースを保護します。リソースを以下に示します。

- **ファイルシステム** - LifeKeeper Single Server Protection ではファイルシステムを定義できます。LifeKeeper Single Server Protection ファイルシステムリソース。 [ファイルシステムの健全性監視](#) がディスクフルと不適切なマウント (またはアンマウント) のファイルシステム条件を検出します。検出した条件に従って、Recovery Kit が警告メッセージのログ記録、ローカルリカバリの試行、またはサーバの再起動を実行します。

File System Recovery Kit に関連するヘルプトピックとして、 [ファイルシステムのリソース階層の作成](#)、 [ファイルシステムの健全性の監視](#) などがあります。

- **Generic Applications** - Generic Application Recovery Kit は、リソースタイプに対して事前定義リカバリキットが指定されていない Generic Application やユーザ定義アプリケーションを保護でき

ます。このキットを使用すると、特定アプリケーションについてカスタマイズした監視スクリプトやリカバリスクリプトを指定できます。

[Generic Application リソース階層の作成](#)を参照してください。

- IP アドレス - IP Recovery Kit には、Life Keeper 環境で、障害が発生した状態から IP アドレスを復旧するメカニズムがあります。LifeKeeper Single Server Protection の保護下にあるアプリケーションは、保護対象の IP アドレスに関連付けられています。

特定の製品、構成、および管理に関する情報については、Recovery Kit に含まれる IP Recovery Kit テクニカルドキュメンテーションを参照してください。

## LifeKeeper GUI ソフトウェア

LifeKeeper GUI は、Java テクノロジーを使用して開発されたクライアント / サーバアプリケーションであり、LifeKeeper Single Server Protection およびその設定データ用のグラフィカルな管理インターフェースです。LifeKeeper GUI クライアントは、スタンドアロンの Java アプリケーション、および Web ブラウザから呼び出される Java アプレットの両方として実装されます。

## VMware HA との連携

「はじめに」セクションで説明したように、LifeKeeper Single Server Protection は、物理環境と仮想環境の両方で使用できるように設計されています。LifeKeeper SSP を VMware VM にインストールした場合、VMware の HA 機能を LifeKeeper SSP と組み合わせ、保護対象リソースやノードの障害を監視し、復旧することができます。これらの機能を有効にする方法については、[VMware HA と LifeKeeper Single Server Protection の関係の有効化](#)を参照してください。さらに、LifeKeeper SSP には、VMware vCenter と連携する管理インターフェースを提供するオプションのコンポーネントが用意されています ([SteelEye 管理コンソール](#)を参照)。

## SteelEye 管理コンソール

*SteelEye 管理コンソール (SMC)* は、VMware HA の構成で LifeKeeper Single Server Protection を実行するときに使用するオプションのコンポーネントです。SMC は、VMware vCenter と連携する管理インターフェースを提供する専用システムです。

VMware vCenter Server、および SteelEye LifeKeeper Single Server Protection と関係して使用するときの SteelEye 管理コンソールの設定、インストール、および動作については、以下のトピックが役立ちます。詳細は以下のカテゴリに分かれています。

[インストールの概要](#)

[システム要件](#)

[セットアップの実行](#)

[vSphere Client プラグイン](#)

[vSphere Client プラグインの設定](#)

[vSphere Client ユーザインターフェース](#)

[認証情報の設定](#)

## インストールの概要

SteelEye 管理コンソール(SMC)のインストールは、いくつかの重要な手順で構成されます。

1. SMC のホストサーバ(仮想または物理)を指定する必要があります。サーバは **SMC を実行するための専用システム**である必要があります。SIOS Technology Corp. は現在、他の目的に使用中のサーバにおける SMC の実行をサポートしていません。小型の仮想マシンで十分であるため、このサーバが非常に強力である必要はありません。サーバの要件の詳細については、[システム要件](#)を参照してください。
2. SMC ソフトウェアは、`setup` スクリプトを CD メディアまたは `.img` ファイルから実行して、インストールする必要があります。このプロセスはシステムに必要な変更を行い、SMC ソフトウェアコンポーネントをインストールして、SMC サービスを開始します。
3. VMware vSphere Client プラグインを vCenter サーバに登録する必要があります。**注記**: SMC は、1 つの vCenter インスタンスのみと関係できます。このため、複数の vCenter インスタンスを展開する場合は、SMC ソフトウェアを個々の vCenter インスタンスにインストールする必要があります。
4. SMC には、vSphere Client が管理する個々の SteelEye LifeKeeper Single Server Protection (または SMC 経由で管理されるサブセット)との通信に必要な認証情報を設定する必要があります。認証情報の特定のセットが、複数の LifeKeeper Single Server Protection システムに有効である場合、これらの認証情報を 1 回で入力することができ、残りの LifeKeeper Single Server Protection ノードを個別に追加する必要があります。詳細については、[認証情報の設定](#)を参照してください。

## システム要件

SteelEye 管理コンソール(SMC)は専用サーバにインストールする必要があります。これは物理サーバでも仮想サーバでもかまいませんが、他の目的に使用することはできません。SIOS Technology Corp. は現在、他の目的に使用中のサーバにおける SMC ソフトウェアの実行をサポートしていません。このサーバは、以下の最小要件を満たす必要があります。

- Red Hat Enterprise Linux/CentOS 6.4 x86\_64 を実行可能な Intel (または AMD) の 64 ビットシステムであること。ベアメタルシステムと仮想マシンのいずれでもかまいません。
- 512 MB 以上の RAM を装備していること。
- ディスク容量が 8 GB 以上であり、`/opt` ファイルシステムに 1 GB 以上が使用可能であること。
- ネットワークアダプタが 1 つ以上あること。
- TCP/IP 経由で、VMware vCenter Server (vSphere Client プラグインを使用する場合)、および SMC が表示 / 管理する SteelEye サーバと直接通信可能であること。SMC のホストサーバを選

択するときには、ネットワークセグメント / ルート / ファイアウォールの考慮事項がある場合もあります。

## セットアップの前提条件

システムを選択した後、SMC をインストールする前に以下の前提条件を設定する必要があります。

- デフォルトのベースソフトウェアパッケージを使用してシステムをインストールする必要があります。特別なパッケージを選択する必要はありません。
- システムのコアオペレーティングシステムの **yum リポジトリを有効**にし、使用可能にする必要があります。これは、ベースシステムのインストールメディアリポジトリ、またはネットワークリポジトリを `/etc/yum.repos.d/` で有効にする必要があるということです。

サポートするオペレーティングシステムがシステム上で動作を開始した後、[セットアップの実行](#)の手順に従って、SMC ソフトウェアコンポーネントをインストールできます。

## セットアップの実行

SteelEye 管理コンソールソフトウェアのコンポーネントは、CD/DVD、または CD イメージを持つ ISO img ファイルからインストールできます。この時点以降、すべてのソフトウェアの手順は root ユーザとして実行する必要があります。

いずれの場合でも、CD または ISO img ファイルをマウントする必要があります。CD は通常の方法でマウントできます。img ファイルは、以下のようなコマンドを使用して、ループバックデバイス経由でマウントできます。

```
mount -o loop <path-to>/smc.img /mnt
```

(/mnt は、イメージのマウントに適する任意の場所にすることが可能)

### ソフトウェアのインストール

イメージをマウントした後、以下のコマンドでインストールを開始できます。

```
cd /mnt; ./setup
```

セットアップツールがインストールプロセスをガイドし、以下の動作を実行してソフトウェアコンポーネントをセットアップします。

- SMC コンポーネントと競合するパッケージがシステム上に存在しないように、システムパッケージのアップグレードまたはアンインストールが実行されます。このプロセスには、事前インストールされた Web サーバ、およびそれに依存するコンポーネントのアンインストールも含まれます。このプロセスには、数分かかることがあります。
- 次に、すべての SMC ソフトウェアコンポーネント用として、セットアップツールにより、SIOS パッケージのインストール / アップグレードが実行されます。これも数分かかることがあり、システムに必要なその他の変更、特にクライアントと SMC との通信を可能にする iptables ファイアウォール設定の変更が含まれます。HTTP トラフィックが SMC サーバのポート 80 および 443 を通過するように、iptables 設定が変更されます。
- 最後にセットアップツールにより、必要な VMware SDK パッケージがインストールされます。インストールするには、VMware SDK のエンドユーザーライセンスに同意する必要があります。SDK のイン

ストールで、ツールのバイナリファイルのインストール先となるファイルパスを入力するように要求されます。これらのファイルについて、デフォルトの場所をそのまま使用することを推奨します。

### ベースの vCenter と認証情報の設定

ソフトウェアコンポーネントのインストール後、セットアップツールは vSphere Client プラグイン、および SteelEye LifeKeeper Single Server Protection ノードの通信に使用するデフォルトの認証情報を設定します。セットアッププロセスを完了するために、以下の動作が実行されます。

- セットアップツールにより、vCenter サーバ名、ユーザ、およびパスワードの入力が要求されます。この情報は、この SMC サーバが提供する vSphere Client プラグインを、指定した vCenter に登録するために使用されます。この手順の後、vCenter サーバはプラグイン用の追加のタブを表示します。最初のインストールの後にはいつでも、プラグインの再登録または登録解除ができます。そのプロセスの詳細については、[vSphere Client プラグインの設定](#) ページを参照してください。
- 最後に、セットアップツールは SteelEye LifeKeeper Single Server Protection ノードとの通信に使用するデフォルトの認証情報の入力を要求します。これらの認証情報は SMC サーバに格納され、特定サーバに固有の認証情報が設定されていない限り、LifeKeeper Single Server Protection サーバとの通信に使用されます。SMC から LifeKeeper Single Server Protection システムをフルに管理できるようにするには、デフォルトの認証情報には LifeKeeper Single Server Protection ノードへの管理者のアクセス権限が必要です (代表的なインストールでは、ユーザがローカルの `/etc/group` ファイルの `lkadmin` グループに属する必要がある)。通常、デフォルトの認証情報は、インストールした LKSSP ノードの `root` ユーザとパスワードです。**注記:** パスワードのストレージは base64 でエンコードされていますが、LifeKeeper の credstore データベースでは暗号化されません。`lkadmin` グループのメンバシップを持つ別の LKSSP システムアカウントを使用することを推奨します。[認証情報の設定](#) ページには、SMC で使用される認証情報を管理する方法の詳細が記載されています。

これで、セットアッププロセスが完了します。ソフトウェアのインストールと設定が正しく実行されたことを検証する方法の詳細については、[インストールの検証](#) ページを参照してください。

### 認証情報の考慮事項

前述のセットアッププロセスの最後の手順では、LifeKeeper Single Server Protection システムにアクセスするためのデフォルトの認証情報を格納しています。この場合のデフォルトの認証情報とは、システムに固有の認証情報が設定されていないときに、LifeKeeper Single Server Protection システムでの認証に使用される認証情報を指します。SMC は常に、デフォルトの認証情報を使用するように戻ります。

このため、SMC の設定を簡略にするために、可能な限りすべての LifeKeeper Single Server Protection システムで同じ認証情報を使用することを推奨します。これは通常、LifeKeeper Single Server Protection システムの `root` ユーザを使用することを意味しますが、`lkadmin` グループに属するユーザである限り、すべてのシステムで共通な任意のユーザを使用できます。

## SteelEye LifeKeeper Single Server Protection vSphere Client プラグイン

SteelEye LifeKeeper Single Server Protection vSphere Client プラグインは VMware vSphere クライアントと連携し、保護対象の仮想マシンのアプリケーション監視ステータスを提供します。プラグインを動作可能にするには、vCenter Server で登録する必要があります。

プラグインは、すべての転送動作にセキュリティで保護された HTTPS 通信を使用します。はじめてプラグインを vSphere クライアントにロードするときに **LK4Linux Valid SMC** の SSL 証明書のセキュリティ警告を受信しますが、これは正常な動作です。SSL 証明書を検査してローカル証明書ストアにインストールした後、セキュリティ警告を安全に無視できます。

## プラグインの要件

- VMware vSphere バージョン 4 またはバージョン 5
- VMware vSphere Client
- クライアントシステムで Javascript とクッキーが有効であること

詳細については、[vSphere Client プラグインの設定](#)、および[vSphere Client プラグインのセキュリティ警告への対処](#)のトピックを参照してください。

## vSphere Client プラグインの設定

インストール後にいつでも、vSphere Client プラグインの再登録、または認証情報の変更ができます。これは、必要に応じて別の vCenter サーバにプラグインを登録する操作も含みます。プラグインを別の vCenter サーバにインストールする場合、まず現在のサーバから登録解除する必要があります。

## vSphere Client プラグインの登録

プラグインの登録は、/opt/LifeKeeper/bin/registerPlugin.pl ツールで行います。このツールは、vCenter サーバ、ユーザ名、およびパスワードの 3 つの引数をとります。vSphere Client プラグインを再登録するには、これらすべてが必須です。このツールの実行例は、以下のようになります (すべてを 1 行に記述)。

```
/opt/LifeKeeper/bin/registerPlugin.pl --server=myvcenter.mydomain.com --username=vcuser  
--password=vcpassword
```

**注記:** shell の解釈を回避するために、shell 文字でもあるパスワード文字はエスケープする必要があります。

## vSphere Client プラグインの登録解除

SMC サーバからインストールしたプラグインは、以下のコマンドを使用してリストできます。

```
/opt/LifeKeeper/bin/registerPlugin.pl --action=list
```

プラグインを削除するには、以下のコマンドを実行できます (すべてを 1 行に記述)。

```
/opt/LifeKeeper/bin/registerPlugin.pl --action=remove --key=com.sios.us.lkssp
```

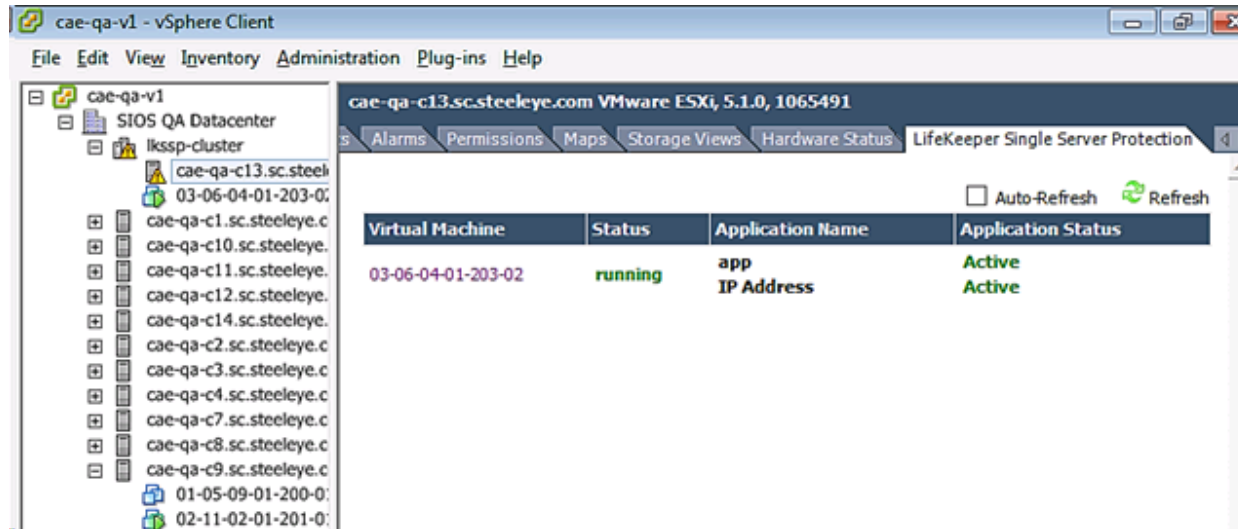
**注記:** プラグインの登録解除に使用するキーは、list 動作で表示されるものと同じにする必要があります。

**注記:** セキュリティ警告の詳細については、[vSphere Client プラグインのセキュリティ警告への対処](#)を参照してください。



## ESX or ESXi Level

特定ホストで動作中の仮想マシンのステータスを表示します。



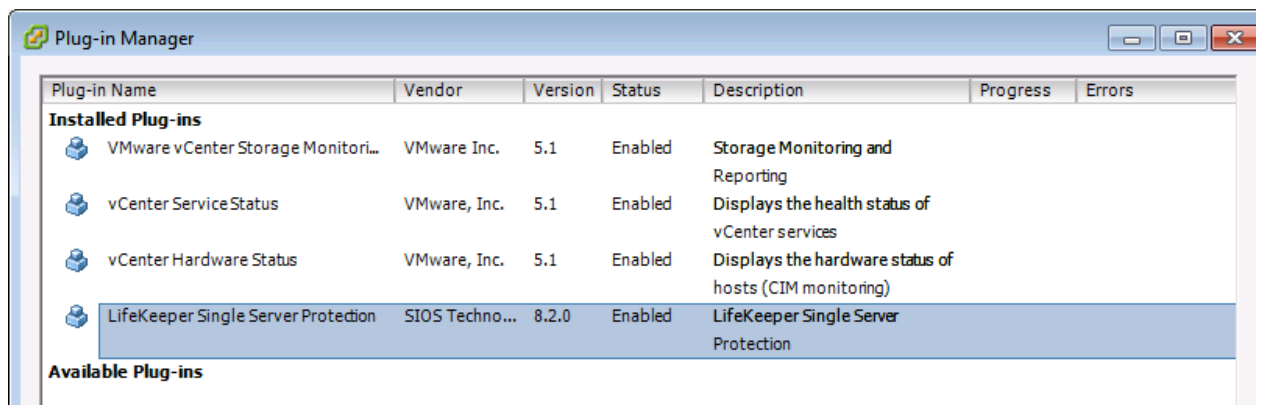
## Virtual Machine Level

以下の特定ノードについて、情報を表示します。仮想マシン名、ホスト名、仮想マシンのステータス、VMware HA のステータス、保護対象のアプリケーション。また、LifeKeeper Single Server Protection のログを表示できる **[View Log]** リンクもあります。



## Manage Plug-Ins

登録済みの LifeKeeper Single Server Protection vCenter プラグインのステータスを表示します。



## その他の表示

LifeKeeper Single Server Protection プラグインは、**[Datacenter]**、**Virtual Application**、および **[Resource Pool]** のレベルでも表示 できます。

## 認証情報の設定

SMC と LifeKeeper Single Server Protection ソフトウェアは、他のシステム (vCenter Server、または SteelEye LifeKeeper Single Server Protection) との通信に使用する認証情報を **認証情報ストア** 経由で管理します。このストアは、例えばプラグインの登録時に使用されます ([vSphere Client プラグインの設定](#)を参照)。このストアは、必要に応じて `/opt/LifeKeeper/bin/credstore` コマンドで管理 できます。このコマンドを使用すると、サーバアクセスに必要な認証情報をサーバごとに設定、変更、削 除することができます。

## 認証情報の追加または変更

認証情報の追加と変更は同じ方法で実行できます。代表的な例として、サーバ `lkssp-server.mydomain.com` の認証情報を追加または変更する場合は次のようになります。

```
/opt/LifeKeeper/bin/credstore -k lkssp-server.mydomain.com myuser
```

この例では、`lkssp-server.mydomain.com` へのアクセスに使用するユーザ名として `myuser` を指定して います。パスワードを入力 / 確認するプロンプト (`passwd` など) が表示されます。

**注記:** LifeKeeper Single Server Protection サーバの認証情報を格納するために SMC で使用する キー名は、LifeKeeper Single Server Protection サーバのホスト名と完全に一致する必要があります (そ の vSphere Client プラグインの **[Hostname:]** フィールドに表示されるものと一致)。ホスト名が FQDN の 場合、認証キーは FQDN である必要があります。ホスト名が短縮名の場合、キーも短縮名にする必 要があります。

[セットアップの実行](#)時に以下の [Credential Considerations] (認証情報に関する考慮事項) が推奨さ れた場合、特定のサーバキーが存在しないときには、対応するユーザ名とパスワードを持つ **デフォルト** キーが認証に使用されます。デフォルトキーを追加、変更するには以下のコマンドを実行してください。

```
/opt/LifeKeeper/bin/credstore -k default myuser
```

## ストア内の認証情報のリスト表示

現在格納されている認証情報をリスト表示するには、以下のコマンドを実行します。

```
/opt/LifeKeeper/bin/credstore -l
```

これにより、認証情報ストア内に格納されているキーが表示されます。この場合の「キー」は、認証情報を使用する対象のサーバを示しています(認証情報自体は秘密情報のため、このコマンドが表示するのは、実際の認証情報の内容ではなくキーのみです)。

## サーバの認証情報の削除

特定のサーバに対する認証情報を削除するには、以下のコマンドを実行します。

```
/opt/LifeKeeper/bin/credstore -d -k lkssp-server.mydomain.com
```

この例では、サーバlkssp-server.mydomain.comの認証情報ストアがストアから削除されます。

## 追加情報

credstore ユーティリティの詳細については、以下のコマンドを実行してください。

```
/opt/LifeKeeper/bin/credstore --man
```

コマンドのマニュアルページがすべて表示されます。

## インストールの検証

SteelEye 管理コンソールのインストールを検証するには、Web ブラウザを使用し、<https://<smcserver>/> に接続します。インストールが正しく実行された場合、SMC サービスが利用可能であることを示すページが表示されます。**注記:** SMC は自己署名の SSL 証明書を使用するので、セキュリティ警告を受信するのは正常な動作です。この警告は安全に無視できます。

ブラウザにページの表示エラーが表示されるか、新規にインストールした SMC サーバへの接続に失敗した場合は、エラーが発生せずにすべてのインストール手順が完了したこと、および SMC サーバがネットワークにアクセス可能であることを確認してください。

## トラブルシューティング

トラブルシューティングについては、[SMC のトラブルシューティング](#)セクションを参照してください。また、セキュリティ警告の詳細については、[vSphere Client プラグインのセキュリティ警告への対処](#)トピックを参照してください。

## vSphere Client プラグインのセキュリティ警告への対処

LifeKeeper Single Server Protection vSphere Client プラグインは、自己署名証明書を使用して SSL 通信を有効にします。プラグインのコンテンツを表示するときに、セキュリティ警告を受信するのは正常

## カスタム証明書の使用

な動作です。セキュリティ警告を低減するには、vSphere Client システムの証明書ストアに「LK4Linux Valid SMC」証明書をインストールする必要があります。さらに、お使いのシステムの「Trusted Root Certification Authorities」証明書ストアに「SIOS Technology, Corp.」認証局 (CA) の証明書をインストールできます。

「LK4Linux Valid SMC」証明書をインストールするには、

1. セキュリティ警告が表示されたら、**[View Certificate]** を選択します。
2. **[Install Certificate]** ボタンをクリックします。
3. ウィザードの手順に従って、証明書をインストールします。

「Trusted Root Certification Authorities」証明書ストアに「SIOS Technology, Corp.」認証局 (CA) の証明書をインストールするには、

1. セキュリティ警告が表示されたら、**[View Certificate]** を選択します。
2. **[Certification Path]** タブをクリックします。
3. **SIOS Technology, Corp.** の証明書を選択します。
4. CA の証明書を表示するには、**[View Certificate]** をクリックします。
5. **[Install Certificate]** ボタンをクリックします。
6. **[Next]** をクリックします。
7. **[Certificate Store Wizard]** ペインの **[Place all certificates in the following store]** ラジオボタンを選択します。
8. **[Browse]** ボタンが有効になります。そのボタンをクリックします。
9. **[Select Certificate Store]** リストから、**[Trusted Root Certification Authorities]** を選択します。
10. **[OK]** をクリックします。
11. **[Next]** をクリックして、ウィザードを完了します。

## カスタム証明書の使用

LifeKeeper Single Server Protection では、異なるシステムとの通信に SSL/TLS が使用されます。デフォルトでは、ノード間で一定の身元確認が可能なデフォルト証明書が SPS と共にインストールされます。このドキュメントでは、デフォルト証明書を組織独自の認証局 (CA) が作成した証明書に置き換える方法を説明します。

### 証明書の使用方法

SteelEye 管理コンソール (SMC) と LifeKeeper Single Server Protection サーバとの通信では、転送するデータを保護するために SSL/TLS が使用されます。双方のシステムは自身を特定する証明書を提示し、証明書を提示されたシステムは、CA 証明書を使用して提示された証明書を SSL 接続経由で確認します。

以下の4種類の証明書が使用されます。

- /opt/LifeKeeper/etc/certs/LK4LinuxValidNode.pem (LifeKeeper Single Server Protection server certificate)
- /opt/LifeKeeper/etc/certs/LK4LinuxValidSMC.pem (SMC server certificate)
- /opt/LifeKeeper/etc/certs/LK4LinuxClient.pem (LifeKeeper Single Server Protection client certificate, installed on all servers)
- /opt/LifeKeeper/etc/certs/LKCA.pem (certificate authority, installed on all servers)

最初の3つの証明書がサーバが実行する検証に合格するためには、4番目の証明書による署名が必要です。証明書の共通名は検証されません。証明書はCAによって署名されるのみということに注意してください。

## 独自の証明書の使用

運用環境によっては、デフォルト証明書を組織内部のCAが作成した証明書に置き換える必要がある場合があります。そのような場合は、上記の4種類の証明書を、同じ証明書ファイル名を持つ新しい証明書に置き換えます。これらの証明書はPEM形式で

す。LK4LinuxValidNode.pem、LK4LinuxValidSMC.pem、およびLK4LinuxValidClient.pemはそれぞれ、キーと証明書の両方を含んでいます。LK4LinuxValidNode.pem および LK4LinuxValidSMC.pem の証明書は、サーバタイプの証明書です。LK4LinuxValidClient.pemは、クライアントタイプの証明書です。

デフォルトの証明書を置換した場合、変更を反映するにはLifeKeeper Single Server Protection および SMC を再起動する必要があります。証明書の設定を間違えると、steeleye-lighttpd デーモンが起動に失敗し、LifeKeeper Single Server Protection のログファイルにエラーが記録されます。問題が発生した場合、このログファイルを参照すると実行すべき完全なコマンドを見ることができます。

## Application Recovery Kit

Application Recovery Kit (ARK) には、LifeKeeper Single Server Protection が特定のアプリケーションを管理および制御するために必要なツールとユーティリティが含まれています。特定のアプリケーション用のARKをインストールすると、LifeKeeper Single Server Protection はそのアプリケーションの状態を監視し、アプリケーションに障害が発生したときに自動的に復旧できます。LifeKeeper Single Server Protection Recovery Kit は non-intrusive であり、LifeKeeper Single Server Protection がアプリケーションを保護するために、アプリケーション内部の変更を要求することは一切ありません。

以下のドキュメンテーションはLifeKeeperのドキュメンテーションであることに注意してください。このため、これらのドキュメントを読むときに注意が必要な点があります。

- LifeKeeper Single Server Protection はシングルノード製品であるため、コミュニケーションパス、クラスタ、バックアップノード、バックアップノードへの階層の拡張、バックアップノードへの階層のフェイルオーバーなどの概念はLifeKeeper Single Server Protection の環境には適用されません。LifeKeeper Single Server Protection の設定および管理の目的では、これらの用語や操作の説明を無視する必要があります。
- LifeKeeper Single Server Protection は、共有ストレージの保護やデータレプリケーションの保護を行わないため、アプリケーションを使用可能な任意のストレージに配置して、LifeKeeper Single Server Protection で保護することができます。これにより、インストールと設定が大幅に簡

## リソース階層

略化されます。これらのガイドを読むときには、アプリケーションデータを共有ディスクに移動する説明は無視できます。

- VMware ベースの製品では、CPU、メモリ、およびディスク容量の要件が仮想マシンに適用される点を除いて、その他のハードウェア要件の説明は該当しません。
- IP アドレスは保護可能 (一部の構成では IP アドレスを保護する必要がある) ですが、「切り替え可能な IP」の説明は無視する必要があります。システムが1つのみの構成では、保護対象の IP は切り替えできませんが、保護対象の IP が定義されているシングルノード上では保護されます。
- GUI を使用して SAP 階層を作成することはできません。対応策については、[トラブルシューティング](#)セクションの「SAP の既知の問題」を参照してください。
- ルートファイルシステム上に Oracle 階層を作成することはできません。対応策については、[トラブルシューティング](#)セクションの「Oracle の既知の問題」を参照してください。

各 Recovery Kit のテクニカルドキュメンテーションには、ソフトウェアパッケージに固有の設定および管理情報が記載されています。Recovery Kit ドキュメンテーションへのリンクについては、弊社 Web サイトの「テクニカルドキュメンテーション」セクションを参照してください。

## リソース階層

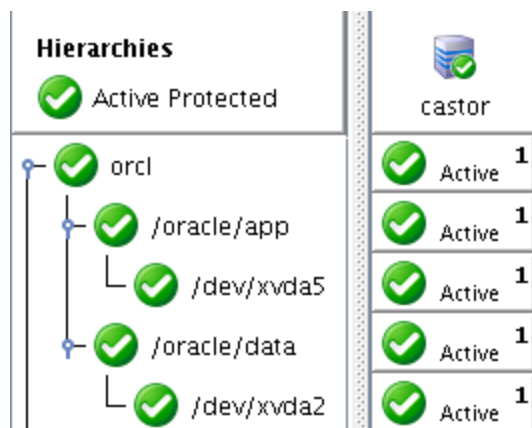
LifeKeeper GUI を使用して、サーバ上にリソース階層を作成できます。リソース階層の作成後、LifeKeeper Single Server Protection が階層内のリソースの停止、開始、監視、およびリカバリを管理します。以下の関連トピックで、階層の指定作業の基本情報を説明しています。

## リソースタイプ

リソースとは保護対象のソフトウェアエンティティであり、アプリケーション、データベース、ファイルシステムなどが該当します。リソースは、タイプ別に分類されます。LifeKeeper Single Server Protection Core は、ファイルシステム、IP アドレス、および Generic Application のリソースタイプを提供します。各種の Recovery Kit が、その他のデータベース、アプリケーション、およびシステムインフラストラクチャのリソースタイプを提供します。

例えば、保護する Oracle データベースのリソース階層には、以下のタイプのリソースが含まれます。

- **database/oracle:** 保護する Oracle データベースインスタンス。
- **gen/filesys:** Linux ファイルシステムリソース。
- **device:** ディスクパーティション、または仮想デバイス (例: `sdcl`)。



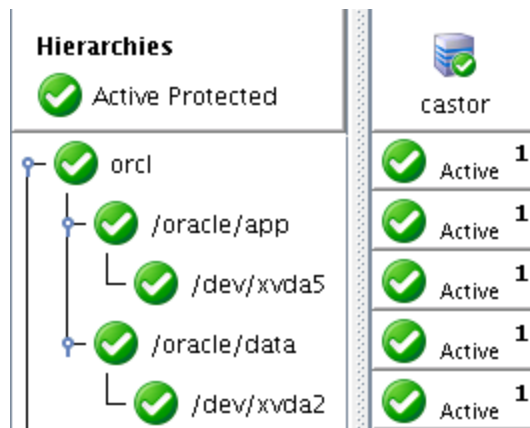
## リソースの状態

状態	意味
In-Service、保護 (ISP)	リソースが動作可能です。LifeKeeper Single Server Protection アプリケーションのリカバリが正常に動作中です。
In-Service、未保護 (ISU)	リソースは動作可能ですが、 <b>警告状態</b> にある可能性があります。
Out-of-Service、障害 (OSF)	リソースが、障害により out-of-service になっています。リカバリは完了していないか、失敗しました。このリソースについて、LifeKeeper Single Server Protection の警告機能は動作不能です。
Out-of-Service、障害なし (OSU)	リソースが out-of-service です。
不正状態 (ILLSTATE)	この状態は、リソースインスタンスについて状態が設定されていない場合に表示されます。通常の場合では、この不正状態が長く続くことはありません。ある状態から別の状態への移行が予測されます。

## 階層の関係

LifeKeeper Single Server Protection を使用すると、リソースインスタンス間の関係を作成できます。最も重要な関係は依存関係です。リソースインスタンスと依存関係を組み合わせたものをリソース階層と呼びます。

## ステータスの詳細表示



(上の画像では、Oracle データベース *orcl* は、2つのファイルシステム */oracle/app* と */oracle/data* に依存する)。

あるリソースが正しく機能するために別のリソースに依存する場合、依存関係が必要です。依存関係は、リソースを *in service* および *out of service* にする順序を定義します。通常、LifeKeeper Single Server Protection の Recovery Kit は、アプリケーションを保護するときに正しい依存関係を作成します。ただし、カスタムの依存関係の作成が必要な場合があります(この一般的な例は、Generic Application Recovery Kit を使用してカスタムアプリケーションを保護する場合)。

## ステータスの詳細表示

このトピックでは、`lcdstatus` コマンドの出力例を使用してステータスの詳細表示で提供される情報のカテゴリについて説明します。この情報を表示する方法については、LCD (1M) のマニュアルページを参照してください。コマンドラインに、`man lcdstatus` または `man LCD` を入力できます。LifeKeeper の GUI で使用できるステータス情報については、[サーバーのステータスの表示](#) または [リソースのステータスの表示](#) を参照してください。

ステータスの詳細表示の例:

### [リソース階層の情報](#)

Resource hierarchies for machine "castor":

ROOT of RESOURCE HIERARCHY

```
orcl: id=orcl app=database type=oracle state=ISP
      initialize=(AUTORES_ISP) automatic restore to IN-SERVICE by
```

LifeKeeper

```
      info=/oracle/app/oracle/product/11.2.0/dbhome_1
```

```
      reason=restore action has succeeded
```

```
      depends on resources: /oracle/data,/oracle/app
```

```
/oracle/data: id=/oracle/data app=gen type=filesystem state=ISP
```

```
      initialize=(AUTORES_ISP) automatic restore to IN-SERVICE by
```

LifeKeeper

```
      info=ext3rw0
```

```
      reason=restore action has succeeded
```

```
      depends on resources: /dev/xvda2
```

```

    these resources are dependent: orcl
/dev/xvda2: id=/dev/xvda2 app=scsi type=DEVNAME state=ISP
    initialize=(AUTORES_ISP) automatic restore to IN-SERVICE by
LifeKeeper
    reason=restore action has succeeded
    these resources are dependent: /oracle/data
/oracle/app: id=/oracle/app app=gen type=filesys state=ISP
    initialize=(AUTORES_ISP) automatic restore to IN-SERVICE by
LifeKeeper
    info=ext3rw0
    reason=restore action has succeeded
    depends on resources: /dev/xvda5
    these resources are dependent: orcl
/dev/xvda5: id=/dev/xvda5 app=scsi type=DEVNAME state=ISP
    initialize=(AUTORES_ISP) automatic restore to IN-SERVICE by
LifeKeeper
    reason=restore action has succeeded
    these resources are dependent: /oracle/app

```

The following LifeKeeper machines are known:

```
machine=castor state=ALIVE
```

## リソース階層の情報

LifeKeeper Single Server Protection は、リソースのステータスを root リソースから表示します。表示には、リソースのすべての依存関係についての情報が含まれます。

各リソース記述の第 1 行には、リソースタグとその後に続くコロン (:) が表示されます (例: */oracle/data*.)。以下の項目は、階層内のリソースを表します。

- **id** - LifeKeeper Single Server Protection が使用する一意のリソース識別子。
- **app** - アプリケーションのタイプを示します。例えば、サンプルリソースはデータベースアプリケーションです。
- **type** - リソースのクラスタイプを示します。例えば、サンプルリソースは *oracle* タイプのアプリケーションです。
- **state** - リソースの現在の状態。
  - ISP - in-service であり、保護されています。
  - ISU - in-service であり、保護されていません。
  - OSF - out-of-service であり、障害が発生しています。
  - OSU - out-of-service であり、障害はありません。
- **initialize** - リソースを初期化する方法を指定します。
- **info** - Recovery Kit が内部で使用する、リソースに固有の情報があります。
- **reason** - 存在する場合、リソースが現在の状態にある原因を示します。例えば、あるアプリケー

## リソース階層の情報

ションがOSUの状態になった原因は、そのアプリケーションがout of serviceになったからです。

- **depends on resources** - 存在する場合、このリソースが依存するリソースのタグ名がリストされます。
- **these resources are dependent** - 存在する場合、このリソースが直接依存するすべての親リソースのタグ名が示されます。

## LifeKeeper Single Server Protection ソフトウェアのインストール

LifeKeeper Single Server Protection 構成内の各サーバに LifeKeeper Single Server Protection ソフトウェアをインストールしてください。各 LifeKeeper Single Server Protection サーバには、オプションのリカバリキットパッケージを含む、設定要件をサポートするために必要なパッケージがインストールされている必要があります。



**重要:** LifeKeeper Single Server Protection をインストールする前に、[Linux の依存関係トピック](#)を参照してください。

LifeKeeper Single Server Protection Core パッケージおよび他のオプションのリカバリキットは、LifeKeeper Single Server Protection インストールイメージファイル (*lkssp.img*) を使用して、コマンドラインでインストールします。このイメージファイルは、LifeKeeper Single Server Protection をシステムにインストールするときに必要なユーザ対話型のシステムセットアップ作業を実行するよう設計されたインストールスクリプト一式を提供します。インストールイメージファイルは、実行中の Linux ディストリビューションを特定し、一連の質問へのユーザの回答に基づいて、LifeKeeper Single Server Protection を正常にインストールするために必要なさまざまなパッケージをインストールします。ライセンスがインストールされた後にサーバの Host ID と Entitlement ID を取得して表示するユーティリティを提供するライセンシングパッケージもインストールされます。Entitlement ID は LifeKeeper Single Server Protection を実行するための有効なライセンスの取得に使用され、ソフトウェアに付属しています。

**注記:** これらのインストール手順は、読者がサーバにインストールされた Linux オペレーティングシステムに精通していることを前提としています。



**重要:**

- LifeKeeper Single Server Protection は共有ストレージサポートまたは I/O フェンシングを提供しません。各サーバはアプリケーションデータにローカルディスクストレージを使用する必要があります。
- すべての LifeKeeper Single Server Protection パッケージは、`/opt/LifeKeeper` ディレクトリにインストールされます。
- LifeKeeper の既存バージョンを再インストールする場合、最初に、古い LifeKeeper パッケージを削除する必要があります。標準の LifeKeeper のインストールには、既存のリソース階層の再定義が必要になります。現在のリソース階層定義を保持するには、を参照してください。
- LifeKeeper Single Server Protection のインストール中に、LifeKeeper Distribution Enabling Package を参照するエラーメッセージが表示された場合、LifeKeeper Single Server Protection インストールイメージファイル上の **setup** スクリプトを実行または再実行する必要があります。

## LifeKeeper Single Server Protection ソフトウェアのインストール

LifeKeeper Single Server Protection は、使用している Linux ディストリビューションに関わらず、コマンドラインでインストールされます。

1. 次のコマンドを使用して、lkssp.img ファイルをマウントしてください。

```
mount PATH/IMAGE_NAME MOUNT_POINT -t iso9660 -o loop
```

ここで、PATH はイメージへのパスです  
IMAGE\_NAME はイメージの名前です  
MOUNT\_POINT はマウント位置へのパスです

2. lkssp.img がマウントされたディレクトリに移動して、次のコマンドを入力してください。

```
./setup
```

3. インストール手順の間に何が行われるかを説明するテキストが表示されます。ここで行われる一連の質問に対して、**Yes** の場合は「y」、**No** の場合は「n」と答えます。質問の種類と順序は、お使いの Linux ディストリビューションによって異なります。

各質問をよく読んで、適切に回答してください。LifeKeeper Single Server Protection インストールを正常に行うために必要なすべての手順を最後まで行うには、各質問に **Yes** と答えることを推奨します。

4. 次に、LifeKeeper Single Server Protection Core パッケージがインストールされます。
5. ここで setup スクリプトが、ライセンスユーティリティのインストールを実行します。詳細については、[ライセンスの取得とインストール](#)を参照してください。

6. setup スクリプトが提示するすべての質問に回答した後、インストールが成功したことが通知され、インストール可能なすべての LifeKeeper Single Server Protection Recovery Kit の一覧が表示されます。

**注記:** setup スクリプトの実行に関する追跡情報が、`/var/log/LK_install.log` に保存されます。

7. インストールするキットを反転選択し、「スペース」キーを押してください。インストール予定のキットの横に「i」のマークが付きます。**Enter** を押してください。

**注記:** 後でキットを追加するには、`-k` を付けて setup スクリプトを実行します。

```
./setup -k
```

LifeKeeper Single Server Protection では、サーバごとに別々のライセンスが必要です。ライセンスは、ランタイムライセンスです。つまり、LifeKeeper Single Server Protection のインストールはライセンスなしでも可能ですが、正常に製品を起動して実行するためには、事前にライセンスをインストールする必要があります。

インストールスクリプトによってインストールされるライセンスユーティリティパッケージは、LifeKeeper Single Server Protection ソフトウェアの初期インストール時にサーバの使用可能なすべての Host ID を取得して表示します。ライセンスがインストールされると、このユーティリティは Entitlement ID (使用可能な場合) または Host ID (使用できない場合) を返します。

**注記:** Host ID が表示される場合は常に NIC の MAC アドレスに基づいています。

SIOS Technology Corp. ライセンス管理ポータルから取得した LifeKeeper Single Server Protection ライセンスには Entitlement ID が含まれ、クラスタ内の特定のノードにロックされることはありません。LifeKeeper Single Server Protection ソフトウェアと一緒に提供された Entitlement ID (認証コード) は、LifeKeeper Single Server Protection ソフトウェアを実行するために必要なパーマネントライセンスを取得するために使用されます。このプロセスを以下の図に示します。



**注記:** ソフトウェアパッケージごとに、サーバごとのライセンスが必要になります。

LifeKeeper Single Server Protection クラスタ内の各サーバについてライセンスを取得してインストールするには、次の手順を行います。

1. **LifeKeeper Entitlement ID (認証コード) があることを確認してください。** ライセンスの取得に必要な Entitlement ID を含むソフトウェアをメールで受け取っているはずですが。
2. **SIOS Technology Corp. ライセンス管理ポータルでライセンスを取得してください。**
  - a. インターネットアクセスが可能なシステムを使用して、[SIOS Technology Corp. ライセンス管理ポータル](#)にログインしてください。
  - b. **[Manage Entitlements]** を選択してください。

**注記:** パスワードを変更する場合は、画面の右上隅にある **[Profile]** ボタンを使用してください。

- c. **[Entitlement ID]** を探して、行項目の左にあるボックスをオンにすることで、その Entitlement ID に関連付けられた各 **[Activation ID]** を選択してください。
  - d. **[Activate]** タブを選択してください。
  - e. 必要なフィールドを定義して、**[Next]** を選択してください。
  - f. **[Add New Host]** をクリックして、新しいホストを作成してください。
  - g. [Node Locked Host] リストから **[Any]** を選択して、**[Okay]** をクリックしてください。
  - h. **[Host ID]** の左にあるボックスをオンにして、**[Generate]** を選択してください。**[Fulfillment ID]** が **[License Summary]** 画面に表示されます。
  - i. **[Fulfillment ID]** の左にあるボックスをオンにして、**[Email License]** タブを選択してください。
  - j. ライセンスの送信先となる有効なメールアドレスを入力して、**[Send]** を選択してください。
  - k. **[Complete]** を選択してください。
  - l. メールを取得してください。
  - m. ファイルを適切なシステムにコピーしてください。
3. ライセンスをインストールしてください。各システムで、ライセンスファイルを `/var/LifeKeeper/license` にコピーするか、または各システムで、`/opt/LifeKeeper/bin/lkkeyins` を実行してファイルに対するファイル名 (フルパスを含む) を指定してください。

## LifeKeeper Single Server Protection のインストールの検証

LifeKeeper Single Server Protection パッケージが正常にインストールされたことを確認するには、コマンドラインで次のように入力してください。

```
rpm -V <package name>
```

**注記:** パッケージが正しくインストールされている場合、このコマンドは何も出力しません。

コマンドラインから照会を実行するには、次のように入力してください。

```
rpm -qi <package name>
```

**注記:** このコマンドの予想される出力は、パッケージ情報です。

## LifeKeeper Single Server Protection の管理の概要

LifeKeeper Single Server Protection は操作時に管理を必要としません。LifeKeeper Single Server Protection は、保護されたリソースを監視し、障害が発生した場合に指定されたリカバリアクションを実行するように、自動的に機能します。以下のケースでは LifeKeeper Single Server Protection GUI を使用します。

- **リソースおよび階層の定義**。LifeKeeper Single Server Protection は次のインターフェースオプションを提供します。
  - LifeKeeper Single Server Protection GUI。
  - LifeKeeper Single Server Protection コマンドラインインターフェース。
- **リソース監視**。LifeKeeper Single Server Protection GUI は、リソースステータス情報および LifeKeeper Single Server Protection ログへのアクセスを提供します。
- **手動での処理**。メンテナンスやその他の管理アクションのために、サーバまたは特定のリソースを停止することが必要になる場合があります。LifeKeeper Single Server Protection GUI には、特定のリソースを稼働させたり停止させたりすることができるメニュー機能が用意されています。アプリケーションが LifeKeeper Single Server Protection の保護下に置かれると、これらの LifeKeeper Single Server Protection のインターフェースを介してのみアプリケーションを起動および停止させることができます。LifeKeeper Single Server Protection の起動および停止は、コマンドラインを介してのみ行われます。

リソース階層の作成など、管理、設定、およびメンテナンス操作を実行する詳細な手順については、SPS for Linux ドキュメンテーションの[管理作業](#)、[GUI の作業](#)、および[メンテナンス作業](#)を参照してください。

## LifeKeeper Single Server Protection Administration Overview

LifeKeeper Single Server Protection does not require administration during operation. LifeKeeper Single Server Protection works automatically to monitor protected resources and to perform the specified recovery actions if a fault should occur. You use the LifeKeeper Single Server Protection GUI in these cases:

- **Resource and hierarchy definition**. LifeKeeper Single Server Protection provides these interface options:
  - LifeKeeper Single Server Protection GUI.
  - LifeKeeper Single Server Protection command line interface.
- **Resource monitoring**. The LifeKeeper Single Server Protection GUI provides access to resource status information and to the LifeKeeper Single Server Protection logs.

- **Manual intervention.** You may need to stop servers or specific resources for maintenance or other administrative actions. The LifeKeeper Single Server Protection GUI provides menu functions that allow you to bring specific resources in and out of service. Once applications have been placed under LifeKeeper Single Server Protection, they should be started and stopped only through these LifeKeeper Single Server Protection interfaces. Starting and stopping LifeKeeper Single Server Protection is done through the command line only.

See [GUI Tasks](#) and [Maintenance Tasks](#) for detailed instructions on performing LifeKeeper Single Server Protection administration, configuration and maintenance operations.

## Starting LifeKeeper Single Server Protection

All LifeKeeper Single Server Protection software is installed in the directory `/opt/LifeKeeper`.

When you have completed all of the [verification tasks](#), you are ready to start LifeKeeper Single Server Protection on both servers. This section provides information for starting the LifeKeeper Single Server Protection server daemon processes. The LifeKeeper Single Server Protection GUI application is launched using a separate command and is described in [Configuring the LifeKeeper Single Server Protection GUI](#). LifeKeeper Single Server Protection provides a [command line interface](#) that starts and stops the LifeKeeper Single Server Protection daemon processes. These daemon processes must be running before you start the LifeKeeper Single Server Protection GUI.

### Starting LifeKeeper Single Server Protection Processes

If LifeKeeper Single Server Protection is not currently running on your system, type the following command as the user `root` on all servers:

```
/etc/init.d/lifekeeper start
```

Following the delay of a few seconds, an informational message is displayed.

See the LCD(1M) man page by entering `man LCD` at the command line for details on the `/etc/init.d/lifekeeper start` command.

### Enabling Automatic LifeKeeper Single Server Protection Restart

While the above command will start LifeKeeper Single Server Protection, it will need to be performed each time the system is re-booted. If you would like LifeKeeper Single Server Protection to start automatically when server boots up, type the following command:

```
chkconfig lifekeeper on
```

See the `chkconfig` man page for further information.

## Stopping LifeKeeper Single Server Protection

If you need to stop LifeKeeper Single Server Protection, type the following command as `root` to stop it:

```
/etc/init.d/lifekeeper stop
```

This command halts all LifeKeeper Single Server Protection daemon processes on the server being administered if they are currently running. Messages similar to the following are displayed:

```
# /etc/init.d/lifekeeper stop

STOPPING LIFEKEEPER AT: Thu Nov 10 16:56:22 EST 2011
Skipping remove of OSU resource oracle.
Skipping remove of OSU resource /oracle/data.
/opt/LifeKeeper/bin/perform_action -G -t /lv_jailbird1 -a remove -- -m
Thu Nov 10 16:56:28 EST 2011 remove: BEGIN remove of file system /lv_jailbird1
LifeKeeper: unmounting file system /lv_jailbird1
          umount /lv_jailbird1
LifeKeeper: file system /lv_jailbird1 successfully unmounted
Thu Nov 10 16:56:29 EST 2011 remove: END remove of file system /lv_jailbird1
(err=0)
/opt/LifeKeeper/bin/perform_action -G -t /dev/mapper/vg_jailbird-lv_jailbird1 -a
remove -- -m
LIFEKEEPER NOW STOPPED AT: Thu Nov 10 16:56:38 EST 2011
```

See `lkstop` under the `LCD(1M)` man page for additional details.

### Disabling Automatic LifeKeeper Single Server Protection Restart

If you do not want LifeKeeper Single Server Protection to automatically restart when the system is restarted, type the following command:

```
chkconfig lifekeeper off
```

See the `chkconfig` man page for further information.

## Viewing LifeKeeper Single Server Protection Processes

To see a list of all LifeKeeper Single Server Protection core daemon processes currently running, type the following command:

```
ps -ef | grep LifeKeeper | grep -w bin | grep -v lklogmsg
```

An example of the output is provided below:

```
root 11663 11662 0 14:03 pts/0 00:00:00 /bin/bash /etc/redhat-lsb/lsh_start_daemon
/opt/LifeKeeper/sbin/runsvdir -P /opt/LifeKeeper/etc/service log: runit just
starte-
d.....-
.....

root 11666 11663 0 14:03 pts/0 00:00:00 /bin/bash -c ulimit -S -c 0 >/dev/null 2> &1 ;
/opt/LifeKeeper/sbin/runsvdir -P /opt/LifeKeeper/etc/service log: runit just
starte-
d.....-
.....

root 11880 11873 0 14:03 ? 00:00:00 /opt/LifeKeeper/bin/lk_logmgr -
/opt/LifeKeeper/out -d/etc/default/LifeKeeper
```

```
root 12240 11877 0 14:04 ? 00:00:00 /opt/LifeKeeper/bin/lcm
root 12247 11879 0 14:04 ? 00:00:00 /opt/LifeKeeper/bin/ttymonlcm
root 12250 11876 0 14:04 ? 00:00:00 /opt/LifeKeeper/bin/lcd
root 12307 11874 0 14:04 ? 00:00:00 /opt/LifeKeeper/bin/lkcheck
root 12311 11875 0 14:04 ? 00:00:00 /opt/LifeKeeper/bin/lkscsid
root 12325 11871 0 14:04 ? 00:00:00 /opt/LifeKeeper/bin/lkvmhad
root 12335 12330 0 14:04 ? 00:00:00 /opt/LifeKeeper/bin/perl
/opt/LifeKeeper/htdocs/cgi-bin/DoRequest.fcgi
```

The run state of LifeKeeper Single Server Protection can be determined via the following command:

```
/opt/LifeKeeper/bin/lktest
```

If LifeKeeper Single Server Protection is running it will output something similar to the following:

```
F S UID PID PPID C CLS PRI NI SZ STIME TIME CMD
4 S root 12240 11877 0 TS 39 -20 6209 14:04 00:00:00 lcm
4 S root 12247 11879 0 TS 39 -20 30643 14:04 00:00:00 ttymonlcm
4 S root 12250 11876 0 TS 29 -10 9575 14:04 00:00:00 lcd
```

If LifeKeeper Single Server Protection is not running, then nothing is output and the command exists with a 1.

**Note:** There are additional LifeKeeper Single Server Protection processes running that start, stop, and monitor the LifeKeeper Single Server Protection core daemon processed along with those required for the Graphical User Interface (GUI). See [Viewing LifeKeeper Single Server Protection Controlling Processes](#) and [Viewing LifeKeeper GUI Server Processes](#) for a list of the processes. Additionally, most LifeKeeper Single Server Protection processes have a child lklogmsg to capture and log any unexpected output.

## Viewing LifeKeeper Single Server Protection GUI Server Processes

To verify that the LifeKeeper Single Server Protection GUI Server is running, type the following command:

```
ps -ef | grep runGuiSer
```

You should see output similar to the following:

```
root 2805 1 0 08:24 ? 00:00:00 sh /opt/LifeKeeper/bin/runGuiServer
```

To see a list of the other GUI Server daemon processes currently running, type the following command:

```
ps -efw | grep S_LK
```

You should see output similar to the following:

```
root 819 764 0 Oct16 ? 00:00:00 java -Xint -Xss3M -DS_LK=true -
Djava.rmi.server.hostname=wake -Dcom.steeleye.LifeKeeper.rmiPort=82 -
Dcom.steeleye.LifeKeeper.LKROOT=/opt/LifeKeeper -DGUI_RMI_REGISTRY=internal
-DGUI_WEB_PORT=81 com.steeleye.LifeKeeper.beans.S_LK
```

To verify that the LifeKeeper Single Server Protection GUI Server Administration Web Server is running type the following command:

```
ps -ef|grep steeleye-light | egrep -v "lklogmsg|runsv"
```

You should see output similar to the following:

```
root 12330 11872 0 14:04 ? 00:00:00 /opt/LifeKeeper/sbin/steeleye-
lighttpd -D -f/opt/LifeKeeper/etc/lighttpd/lighttpd.conf
```

## Viewing LifeKeeper Single Server Protection Controlling Processes

To verify that the LifeKeeper Single Server Protection controlling processes are running, type the following command:

```
ps -ef | grep runsv
```

You should see output similar to the following:

```
root 11663 11662 0 14:03 pts/0 00:00:00 /bin/bash /etc/redhat-lsb/lst_start_daemon
/opt/LifeKeeper/sbin/runsvdir -P /opt/LifeKeeper/etc/service log: runit just
starte-
d.....-
.....

root 11666 11663 0 14:03 pts/0 00:00:00 /bin/bash -c ulimit -S -c 0 >/dev/null 2>&1 ;
/opt/LifeKeeper/sbin/runsvdir -P /opt/LifeKeeper/etc/service log: runit just
starte-
d.....-
.....

root 11667 11666 0 14:03 pts/0 00:00:00 /opt/LifeKeeper/sbin/runsvdir -P
/opt/LifeKeeper/etc/service log: runit just
starte-
d.....-
.....

root 11871 11667 0 14:03 ? 00:00:00 /opt/LifeKeeper/sbin/runsv lkvmhad
root 11872 11667 0 14:03 ? 00:00:00 /opt/LifeKeeper/sbin/runsv steeleye-lighttpd
root 11873 11667 0 14:03 ? 00:00:00 /opt/LifeKeeper/sbin/runsv lk_logmgr
root 11874 11667 0 14:03 ? 00:00:00 /opt/LifeKeeper/sbin/runsv lkcheck
root 11875 11667 0 14:03 ? 00:00:00 /opt/LifeKeeper/sbin/runsv lkcsid
```

```
root 11876 11667 0 14:03 ? 00:00:00 /opt/LifeKeeper/sbin/runsv lcd
root 11877 11667 0 14:03 ? 00:00:00 /opt/LifeKeeper/sbin/runsv lcm
root 11878 11667 0 14:03 ? 00:00:00 /opt/LifeKeeper/sbin/runsv lkguiserver
root 11879 11667 0 14:03 ? 00:00:00 /opt/LifeKeeper/sbin/runsv ttymonlcm
```

These processes start, stop, and monitor LifeKeeper Single Server Protection core daemon processes and must be running to start LifeKeeper Single Server Protection. These processes are configured by default to start when the system boots and this behavior should not be altered.

## VMware HA と LifeKeeper Single Server Protection の連携を有効にする

デフォルトでは、VMware VM 上にインストールした場合、LifeKeeper Single Server Protection と VMware HA の連携は無効になっています。連携を有効にするには、以下の手順が必要です。

1. LifeKeeper Single Server Protection VM に VMware Tools をインストールしてください。
2. `/etc/default/LifeKeeper` を編集して、VMware HA 連携を調整する `HA_DISABLE` の値を 1 から 0 に変更します。
3. LifeKeeper Single Server Protection を再起動します。LifeKeeper Single Server Protection が実行中の場合、`/etc/default/LifeKeeper` の上記の変更内容が検出されるように、停止してから再起動する必要があります。
4. [SteelEye 管理コンソール](#) をインストールします (オプション)。

## ログファイルのサイズを増やす

LifeKeeper Single Server Protection の起動時に、ログファイルに必要な最大容量が割り当てられません。このサイズ (LifeKeeper Single Server Protection ログファイルに割り当てられる容量) は、`/etc/default/LifeKeeper` のパラメータで調整できます。

```
LOGFILE=log:2048
LOGFILE=TTYLCM:256
LOGFILE=LCM:1024
LOGFILE=LCD:512
LOGFILE=remote_execute:512
LOGFILE=SNMP:512
LOGFILE=NOTIFY:512
```

ログファイルのサイズは、512 バイトブロックの数を表す数値の引数によって指定されます。

**注意:** ログファイルのサイズは 2 GB 未満にする必要があります。

## VMware HA を有効化した障害検出およびリカバリシナリオ

アプリケーション内の問題を検出して通知する機能は、最適な総合的耐障害性ソリューションを構築する上で非常に重要です。すべての個々のアプリケーションは、障害発生メカニズムと形式によって異なるため、一般的なメカニズムを示すことはできません。ただし、一般的に、多くのアプリケーションの設定は、LifeKeeper Single Server Protection に用意されている Core システムのエラー検出機能を利用することができます。このトピックでは、LifeKeeper Single Server Protection Core の機能について説明します。

アプリケーションに障害が発生したときに LifeKeeper Single Server Protection が障害を検出しリカバリを実行する仕組みを説明したリカバリシナリオを以下に示します。

1. LifeKeeper Single Server Protection は最初に、アプリケーションを再起動することでリカバリを試みます。
2. リカバリが成功した場合、アプリケーションは正常動作を継続します。
3. リカバリに失敗した場合、以下の処理が実行されます。
  - a. LifeKeeper Single Server Protection が HA を有効 (/etc/default/LifeKeeper で HA\_DISABLE=0) にした VMware ゲスト OS にインストールされている場合にリカバリに失敗すると、LifeKeeper Single Server Protection がアプリケーション監視インターフェースに送信するハートビートを抑制することで VMware HA がトリガされます。VMware HA はサーバを再起動することで応答します。
  - b. LifeKeeper Single Server Protection が VMware ゲスト OS にインストールされていないか、HA を無効 (/etc/default/LifeKeeper で HA\_DISABLE=1) にした VMware ゲスト OS にインストールされている場合にリカバリに失敗すると、システムが強制的に再起動されます。

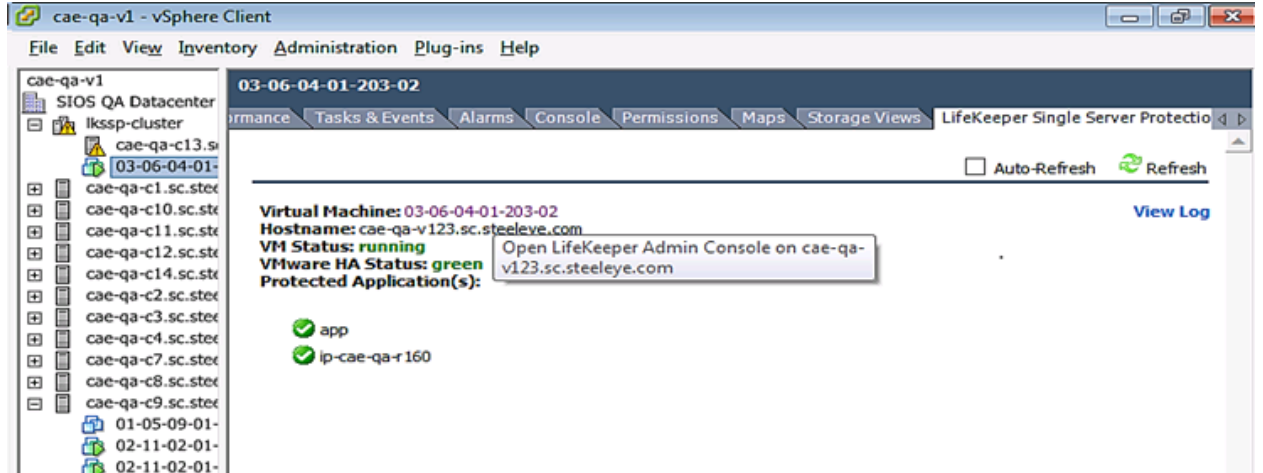
必要に応じて、LifeKeeper Single Server Protection を **通知のみモード** にすることができます。このモードでは、システム再起動の自動トリガは無効になります (以下の VMware HA と通知のみモードセクションを参照)。**通知のみモード**では、ユーザがシステムにログインし、障害の原因になった問題を修正する必要があります。

### VMware HA と通知のみモード

1. VMware ゲスト OS で HA が有効にされ、[LifeKeeper SSP vCenter プラグイン](#)がインストールされた **通知のみモード**では、障害が検出された場合、LifeKeeper Single Server Protection はアプリケーションの再起動を行いません。その代わりに、リソースは **[Failed]** とマークされます。[vCenter プラグイン](#)ダッシュボードのステータス表示画面には障害が表示されます (**[Application Status] : [Failed]**)。

Application Name	Application Status
Login failed: Couldn't resolve host name (6)	N/A
<b>Apache</b>	<b>Failed</b>
Login failed: Couldn't connect to server (7)	N/A
Login failed: Couldn't connect to server (7)	N/A
N/A	N/A
Login failed: Couldn't resolve host name (6)	N/A

2. サーバにログインして、障害の原因になった問題を修正します。
3. CLI を使用するか、vSphere Client ユーザーインターフェース内の保護対象仮想マシンをクリックして、LifeKeeper 管理コンソールを開いてください。



4. アプリケーションを In Service に戻します。
5. vSphere Client ユーザーインターフェース内のダッシュボード表示に移動します。
6. [Refresh] をクリックします。[Application Status] は [Active] に戻ります。

Application Name	Application Status
Login failed: Couldn't resolve host name (6)	N/A
<b>Apache</b>	<b>Active</b>
Login failed: Couldn't connect to server (7)	N/A
Login failed: Couldn't connect to server (7)	N/A
N/A	N/A
Login failed: Couldn't resolve host name (6)	N/A

## LifeKeeper Single Server Protection ハートビートとVMware HA

LifeKeeper Single Server Protection ハートビートは、保護対象のアプリケーションが正常であることを示すために、(VMware ゲスト OS で動作し、HA が有効である場合、10 秒ごとに) VMware HA に送信される信号です。アプリケーションで障害が発生すると、LifeKeeper Single Server Protection は最初にアプリケーションを復旧しようとします。復旧に失敗すると、LifeKeeper Single Server Protection はハートビートを抑制し、VMware HA に VM の再起動を指示します。

## LifeKeeper Single Server Protection で保護するシステムのメンテナンス

LifeKeeper Single Server Protection で保護されているサーバでシステムまたはアプリケーションのメンテナンスを実行するときには、LifeKeeper Single Server Protection による監視を停止するか、保護対象のリソースをメンテナンスモードにしてください。これにより、アプリケーションのリカバリ、および VMware HA の障害イベントのトリガが無効になるので、LifeKeeper Single Server Protection がシステムやアプリケーションのメンテナンス作業に干渉しなくなります。

LifeKeeper Single Server Protection を停止して再起動するには、以下の操作を実行してください。

1. **LifeKeeper Single Server Protection を停止します。** `/etc/init.d/lifekeeper stop-daemons` コマンドを使用して、LifeKeeper Single Server Protection を停止してください。リソースは継続して動作しますが、LifeKeeper Single Server Protection からは監視されなくなります。障害は手動で処理する必要があります。
2. **メンテナンスを実行します。** 必要なメンテナンスを実行します。
3. **SteelEye LifeKeeper Single Server Protection を起動します。** `/etc/init.d/lifekeeper start` コマンドを使用して、LifeKeeper Single Server Protection を開始してください。リソースが保護されている状態になります。

**別の方法** - リソースをメンテナンス(「通知のみ」とも呼ばれる)モードにします。以下の操作を実行してください。

1. **リソースをメンテナンスモードにします。** `/opt/LifeKeeper/bin/lkpolicy -s NotificationOnly --On` コマンドを使用してください。リソースは復旧されなくなり、VMware HA の障害イベントがトリガされなくなります。
2. **メンテナンスを実行します。** 必要なメンテナンスを実行します。
3. **メンテナンスモードをオフにします。** `/opt/LifeKeeper/bin/lkpolicy -s NotificationOnly --Off` コマンドを使用してください。リソースが保護されている状態になります。

## Creating Resource Hierarchies

1. There are four ways to begin creating a resource hierarchy.
  - Right-click on a server icon to bring up the [server context menu](#), then click on **Create Resource Hierarchy**.
  - On the [global toolbar](#), click on the **Create Resource Hierarchy** button.
  - On the [server context toolbar](#), if displayed, click on the **Create Resource Hierarchy** button.
  - On the **Edit** menu, select **Server**, then click on **Create Resource Hierarchy**.
2. A dialog entitled **Create Resource Wizard** will appear with a list of all recognized recovery kits installed within the cluster. Select the Recovery Kit that builds resource hierarchies to

protect your application and click **Next**.

3. Select the **Switchback Type** and click **Next**.
4. Select the **Server** and click **Next**. **Note:** If you began from the server context menu, the server will be determined automatically from the server icon that you clicked on, and this step will be skipped.
5. Continue through the succeeding dialogs, entering whatever data is needed for the type of resource hierarchy that you are creating.

## LifeKeeper Single Server Protection Application Resource Hierarchies

If you install LifeKeeper Single Server Protection without any recovery kits, the Select Recovery Kit list includes options for File System, Generic Application, and IP by default. The Generic Application option may be used for applications that have no associated recovery kits.

See the following topics describing these available options:

- [Creating a File System Resource Hierarchy](#)
- [Creating a Generic Application Resource Hierarchy](#)

The IP Recovery Kit is documented in the IP Recovery Kit Administration Guide.

## Recovery Kit Options

Each optional recovery kit that you install adds entries to the Select Recovery Kit list; for example, you may see Oracle, Apache, and NFS Recovery Kits. Refer to the Administration Guide that accompanies each recovery kit for directions on creating the required resource hierarchies.

## Creating a File System Resource Hierarchy

Use this option to protect a file system only.

1. There are four ways to begin creating a file system resource hierarchy.
  - Right-click on a server icon to bring up the [server context menu](#), then click on **Create Resource Hierarchy**.
  - On the [server context toolbar](#), if displayed, click on the **Create Resource Hierarchy** button.
2. A dialog entitled *Create Resource Wizard* will appear with a **Recovery Kit** list. Select *File System Resource* and click **Next**.
3. Select the **Switchback Type** and click **Next**. *Note: This setting is not relevant for LifeKeeper Single Server Protection.*
4. The *Create gen/filesys Resource* dialog will now appear. Select the **Mount Point** for the file system resource hierarchy and click **Next**.

**Note:** In order for a mount point to appear in the choice list, the mount point must be currently mounted.

5. LifeKeeper Single Server Protection creates a default **Root Tag** for the file system resource hierarchy. (This is the label used for this resource in the status display). You can select this root tag or create your own, then click **Next**.
6. Click **Create Instance**. A window will display a message indicating the status of the instance creation.
7. Click **Next**. A window will display a message that the file system hierarchy has been created successfully.
8. At this point, click **Cancel** to return to the GUI.

## Creating a Generic Application Resource Hierarchy

The generic application recovery kit is used to protect a custom application that has no specific associated recovery kit. Sample scripts are provided in `/opt/LifeKeeper/lkadm/subsys/gen/app/templates`. Copy these samples to another directory before customizing and testing them.

**Note:** For applications that depend on other resources such as a file system or IP address, create each of these resources separately and use **Create Dependency** to create the appropriate dependencies.

1. Begin creating a generic application resource hierarchy.
  - Right-click on a server icon to bring up the [server context menu](#), then click on **Create Resource Hierarchy**
  - On the [server context toolbar](#), if displayed, click on the **Create Resource Hierarchy** button
2. A dialog entitled **Create Resource Wizard** will appear with a **Recovery Kit** list. Select **Generic Application** and click **Next**.
3. Select the **Switchback Type** and click **Next**. **Note:** This setting is not relevant for LifeKeeper Single Server Protection.
4. On the next dialog, enter the path to the **Restore Script** for the application and click **Next**. This is the command that starts the application. A template restore script, `restore.template`, is provided in the templates directory. The restore script must not impact applications that are already started.
5. Enter the path to the **Remove Script** for the application and click **Next**. This is the command that stops the application. A template remove script, `remove.template`, is provided in the templates directory.
6. Enter the path to the **QuickCheck Script** for the application and click **Next**. This is the command that monitors the application. If the script detects a problem, it should exit with a non-zero value in order to trigger a recovery or restart (see [Fault Detection and Recovery Scenario](#)).

7. Enter the path to the **Local Recovery Script** for the application and click **Next**. This is the command that attempts to restore a failed application. A template recover script, `recover.template`, is provided in the `templates` directory.
8. Enter any **Application Information** and click **Next**. This is optional information about the application that may be used by the `restore`, `remove`, `quickCheck`, and `recover` scripts.
9. Select either **Yes** or **No** for **Bring Resource In Service**, and click **Next**. Selecting **No** will cause the resource state to be set to **OSU** following the create; selecting **Yes** will cause the previously provided restore script to be executed. For applications depending upon other resources such as a file system or IP address, select **No** if you have not already created the appropriate dependent resources.
10. Enter the **Resource Tag**, which is a unique name for the resource instance. (This is the label you will see for this resource in the status display.)
11. Click **Create Instance** to start the creation process. A window will display a message indicating the status of the instance creation.
12. Click **Next**. A window will display a message that the hierarchy has been created successfully.
13. At this point, click **Cancel** to return to the GUI.

## Editing Resource Properties

1. To edit the properties of a resource, bring up the **Resource Properties** dialog just as you would for [viewing resource properties](#).
2. If you are logged into that server with the appropriate permissions, the following items will be editable.
  - Resource Configuration (only for resources with specialized configuration settings)
  - [Resource Priorities](#)
3. Once you have made changes, the **Apply** button will be enabled. Clicking this button will apply your changes without closing the window.
4. When you are finished, click **OK** to save any changes and close the window, or **Cancel** to close the window without applying changes.

## Creating a Resource Dependency

While most Recovery Kits create their dependencies during the original resource hierarchy creation task, under certain circumstances, you may want to create new or additional resource dependencies or delete existing ones. An example might be that you wish to change an existing IP dependency to another IP address. Instead of deleting the entire resource hierarchy and creating a new one, you can delete the existing IP dependency and create a new dependency with a different IP address.

1. Right-click on the icon for the parent resource to which you want to add a child dependency. When the [resource context menu](#) appears, click **Create Dependency**

2. Select a **Child Resource Tag** from the drop-down box of existing, valid resources on the server. The dialog will display all the resources available on the server with the following exceptions:

- The parent resource, its ancestors, and its children.
- Any resource that is not in service, if the parent resource is in service.

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

3. The dialog will then confirm that you have selected the appropriate parent and child resource tags for your dependency creation. Click **Create Dependency** to create the dependency on the server.
4. If the [output panel](#) is enabled, the dialog closes, and the results of the commands to create the dependency are shown in the output panel. If not, the dialog remains up to show these results, and you click **Done** to finish when all results have been displayed.

## Deleting a Resource Dependency

1. Right-click on the icon for the parent resource from which you want to delete a child dependency. When the [resource context menu](#) appears, click **Delete Dependency**.
2. Select the **Child Resource Tag** from the drop down box. This should be the tag name of the child in the dependency that you want to delete. Click **Next** to proceed to the next dialog box.
3. The dialog then confirms that you have selected the appropriate parent and child resource tags for your dependency deletion. Click **Delete Dependency** to delete the dependency on the server.
4. If the [output panel](#) is enabled, the dialog closes, and the results of the commands to delete the dependency are shown in the output panel. If not, the dialog remains up to show these results, and you click **Done** to finish when all results have been displayed.

## Deleting a Hierarchy

1. Right-click on the icon for a resource in the hierarchy that you want to delete. When the [resource context menu](#) appears, click **Delete Resource Hierarchy**.
2. The dialog will display a message verifying the hierarchy you have specified for deletion. Click **Delete** to perform the action.
3. If the [output panel](#) is enabled, the dialog closes, and the results of the commands to delete the hierarchy are shown in the output panel. If not, the dialog remains up to show these results, and you click **Done** to finish when all results have been displayed.

## LifeKeeper Single Server Protection のアンインストール

LifeKeeper Single Server Protection 製品をアンインストールするには、付属の rmlk ユーティリティを実行してください。

```
/opt/LifeKeeper/bin/rmlk
```

## LifeKeeper Single Server Protection のアンインストール

これにより、すべての SIOS 製品の rpm がアンインストールされ、システムから `/opt/LifeKeeper` ディレクトリが削除されます。**慎重に使用してください。**

## Chapter 4: ユーザガイド

LifeKeeper Single Server Protection ユーザガイドは、検索可能な総合リソースで、LifeKeeper の GUI で実行できる多くの作業の詳細情報があります。

### LifeKeeper GUI

#### LifeKeeper グラフィカルユーザインターフェース

GUI のコンポーネントは、LifeKeeper Single Server Protection Core のインストールの一部として、すでにインストールされています。

LifeKeeper の GUI は、Java テクノロジーを使用して、LifeKeeper Single Server Protection およびその設定データ用にグラフィカルユーザインターフェースを提供します。LifeKeeper の GUI はクライアント / サーバアプリケーションなので、ユーザはクライアントシステムでグラフィカルユーザインターフェースを実行して、LifeKeeper Single Server Protection が動作中のサーバシステムの監視や管理を行います。クライアントとサーバのコンポーネントは、同一システム上にある場合も、ない場合もあります。

#### GUI の概要 - 全般

必要なメンバシップをユーザが持っている限り、GUI を使用することで、任意のマシンからサーバとリソースの管理、操作、または監視ができます(詳細については、[GUI のユーザの設定](#)を参照)。GUI のサーバとクライアントのコンポーネントについて説明します。

**注記:** ファイアウォールの制約とパフォーマンスの問題があるため、GUI クライアントを WAN または VPN 経由で実行しないことを強くお勧めします。GUI クライアントをリモートで実行する場合は、VNC セッションまたは Windows のリモートデスクトップセッションを使用して、GUI クライアントが動作可能で管理対象のサーバに最も近いローカルシステムにログインすることをお勧めします。

#### GUI サーバ

デフォルトでは、システムの起動時に、各 LifeKeeper Single Server Protection サーバ上にある GUI サーバは初期化されません。GUI サーバは、ハイパーテキスト転送プロトコル (HTTP) とリモートメソッド呼び出し (RMI) を使用して GUI クライアントと通信します。LifeKeeper Single Server Protection の開始 / 停止とは別に GUI サーバの開始 / 停止を実行する場合は、[GUI サーバの開始 / 停止](#)を参照してください。

## GUI クライアント

GUI クライアントは、任意の LifeKeeper Single Server Protection サーバ上の [アプリケーション](#) として、または Java が有効な任意のシステム上の [Web クライアント](#) として実行できます。

クライアントには、以下のコンポーネントがあります。

- 左上の [ステータスの表](#) には、接続しているサーバとそのリソースの上位のステータスが表示されます。
- 右上の [プロパティパネル](#) には、ステータスの表で直前に選択したオブジェクトの詳細情報が表示されます。
- 下部の [出力パネル](#) には、コマンドの出力が表示されます。
- ウィンドウの最下部にある [メッセージバー](#) には、処理のステータスメッセージが表示されます。
- コンテキストツールバー (プロパティパネル内) と [グローバルツールバー](#) を使用すると、頻繁に使用する作業に即座にアクセスできます。
- コンテキストメニュー (ポップアップ) と [グローバルメニュー](#) から、すべての作業にアクセスできます。

## GUI クライアントの終了

[\[File\] メニュー](#) から [\[Exit\]](#) を選択すると、すべてのサーバから切断され、クライアントが終了します。

## ステータスの表

ステータスの表には、接続先サーバとリソースのステータスがグラフィカルに表示されます。以下の項目が表示されます。

- 最上位の行にサーバの状態。
- 左端の列に、各リソースのグローバル状態と親 / 子の関係。
- 残りのセルに、各サーバの各リソースの状態。

サーバとリソースの状態は、グラフィックス、テキスト、および色を使用して表示されます。サーバのテーブルの空白セルは、特定のリソースがそのサーバで定義されていないことを示します。

ステータスの表でサーバまたはリソースを選択した場合、その項目の詳細な状態の情報とコンテキスト依存ツールバーが [プロパティパネル](#) に表示されます (プロパティパネルを有効にするには、[\[View\] メニュー](#) の [\[Properties Panel\]](#) をオンにする)。また、任意の項目のセルを右クリックして [\[Properties\]](#) を選択すると、該当する [サーバのコンテキストメニュー](#) または [リソースのコンテキストメニュー](#) をポップアップ表示できます。

ステータスの表は2つのセクションに分かれています。左右のセクションの境界を移動して、それらのセクションの相対サイズを変更できます。また、ステータスの表を折り畳んで、階層ツリーの上位項目のみを表示できます。ツリーの [リソース項目の折り畳み / 展開](#) を実行すると、表内にリストされる階層に対しても折り畳み / 展開が適用されます。

## プロパティパネル

プロパティパネルには、ステータスの表から選択されたサーバまたはリソースのプロパティが表示されます。プロパティパネルは、[Server Properties] ダイアログまたは [Resource Properties] ダイアログと同じ機能を持ち、さらに一般的に使用するコマンドに即座にアクセスできるコンテキスト依存ツールバーがあります。このパネルの上部には、サーバを選択した場合は **server\_name**、リソースを選択した場合は **server\_name: resource\_name** がキャプションとして表示されます。

プロパティパネルに表示されるコンテキスト依存ツールバーは、[サーバのコンテキストツールバー](#)と[リソースのコンテキストツールバー](#)です。サーバまたはリソースのツールバーもカスタマイズできます。ツールバーのカスタマイズの詳細については、該当するApplication Recovery Kit のドキュメンテーションを参照してください。

プロパティパネル下部にあるボタンは、以下の機能を持ちます。

- **[Apply]** ボタンは、パネルの編集可能なプロパティに対する変更内容を適用します。このボタンが有効になるのは、編集可能なプロパティを変更した場合のみです。
- **[Reset]** ボタンは、サーバにすべてのプロパティの現在の値を照会し、これまで変更した内容を消去します。このボタンは常に有効です。
- **[Help]** ボタンは、プロパティパネルのコンテキスト依存ヘルプを表示します。このボタンは常に有効です。

このパネルの有効と無効を切り替えるには、[\[View\] メニュー](#)の **[Properties Panel]** チェックボックスを使用してください。

## 出力パネル

出力パネルは、LifeKeeper GUI クライアントが送出したコマンドの出力を収集します。コマンドの実行開始時に、タイムスタンプ付きのラベルが出力パネルに追加され、そのラベルの下に、そのコマンドの出力がすべて追加されます。複数のコマンドを同時に実行する場合（通常は異なるサーバ上）、各コマンドの出力が対応するセクションに送られ、各コマンドの結果が見やすくなります。

出力パネルのサイズを増減するには、パネル上部にある境界を上下にスライドしてください。このパネルを開閉するには、[\[View\] メニュー](#)の **[Output Panel]** チェックボックスを使用してください。出力パネルを閉じているときには、各コマンドを開始するダイアログが表示されたままになり、このダイアログを閉じるまで出力がこのダイアログに表示されます。そして、このダイアログを閉じた後はコマンドの出力を確認できなくなります。出力パネルを再び開いた後は、LifeKeeper の GUI はデフォルトの動作に戻ります。

## メッセージバー

メッセージバーは、[Status] ウィンドウの下に表示されます。メッセージが1行のテキストで表示されます。「Connecting to Server X」や「Failure to connect to Server X」などのメッセージが表示されます。

メッセージバーを隠すには、[\[View\] メニュー](#)の **[Message Bar]** チェックボックスをオフにします。

メッセージバーを表示するには、[View] メニューの **[Message Bar]** チェックボックスをオンにします。

## GUI の終了

メッセージバーに表示されたメッセージの履歴を表示する方法については、[メッセージ履歴の表示](#)を参照してください。

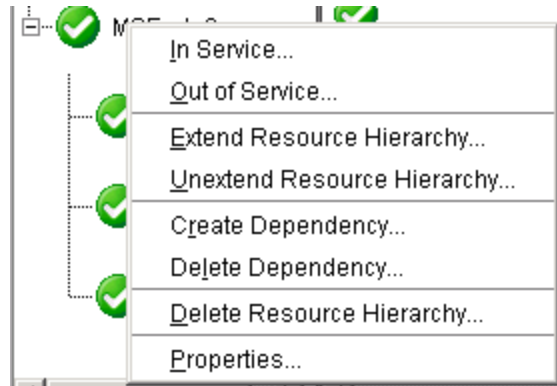
## GUI の終了

サーバから切断し、GUI ウィンドウを閉じるには、[\[File\] メニュー](#)の **[Exit]** を選択してください。

## メニュー

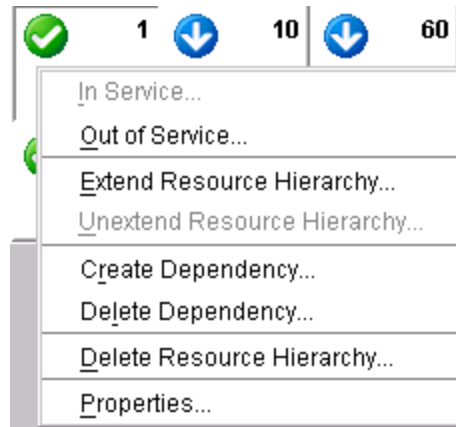
このセクションで説明する GUI メニューから、管理機能を使用できます。

## リソースのコンテキストメニュー



リソースのコンテキストメニューは、[ステータスの表](#) 内にあるグローバルリソース (上図)、またはサーバ固有のリソースインスタンス (下図) を右クリックしたときに表示されます。デフォルトのリソースコンテキストメニューについて、ここで説明しますが、このメニューは特定のリソースタイプについてカスタマイズされていることがあります。この場合、メニューは該当するリソースキットのドキュメンテーションで説明されています。

選択したリソースについて、動作が呼び出されます。特定のサーバのリソースインスタンスを選択して、アクションを実行します。また、グローバルリソースを選択する場合は、アクションを実行するサーバを選択します。



[In Service](#) - リソース階層を in service にします。

[Out of Service](#) - リソース階層を out of service にします。

[Create Dependency](#) - 2つのリソース間に親 / 子の関係を作成します。

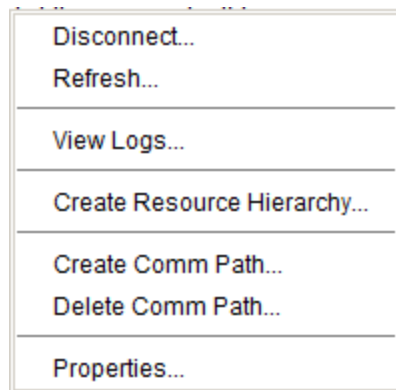
[Delete Dependency](#) - 2つのリソース間にある親 / 子の関係を削除します。

[Delete Resource Hierarchy](#) - リソース階層を削除します。

[Properties](#) - [\[Resource Properties\] ダイアログ](#)を表示します。

## サーバのコンテキストメニュー

サーバのコンテキストメニューは、[ステータスの表](#)内にあるサーバアイコンを右クリックしたときに表示されます。このメニューは [Edit] メニューの [Server] サブメニューと同じですが、動作は常に、最初に選択したサーバ上で呼び出される点が異なります。



[Disconnect](#) - サーバから切断します。

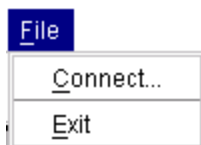
[Refresh](#) - GUI を最新情報に更新します。

[View Logs](#) - 接続しているサーバについて、LifeKeeper Single Server Protection のログメッセージを表示します。

[Create Resource Hierarchy](#) - リソース階層を作成します。

[Properties](#) - [\[Server Properties\] ダイアログ](#)を表示します。

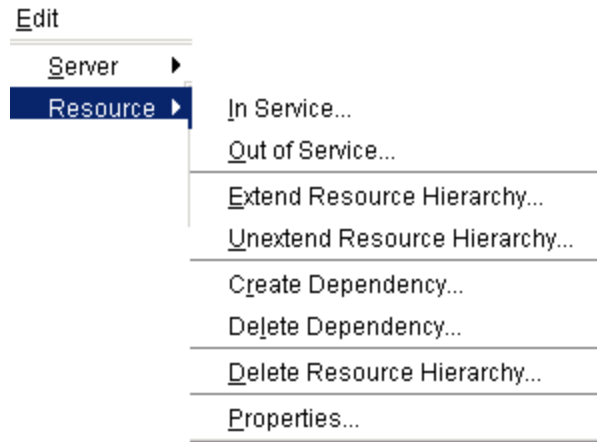
## [File] メニュー



**Connect** - LifeKeeper Single Server Protection サーバに接続します。LifeKeeper Single Server Protection サーバに接続するには、ログイン認証が必要です。

**Exit** - サーバから切断し、GUI ウィンドウを閉じます。

## [Edit] メニュー - リソース



[In Service](#) - リソース階層を in service にします。

[Out of Service](#) - リソース階層を out of service にします。

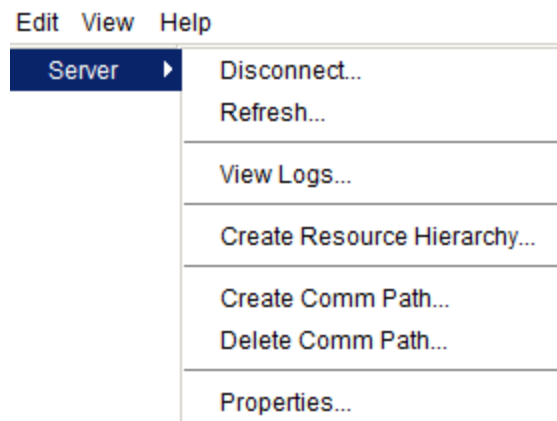
[Create Dependency](#) - 2つのリソース間に親 / 子の関係を作成します。

[Delete Dependency](#) - 2つのリソース間にある親 / 子の関係を削除します。

[Delete Resource Hierarchy](#) - リソース階層を削除します。

[Properties](#) - [\[Resource Properties\] ダイアログ](#)を表示します。

## [Edit] メニュー - サーバ



[Disconnect](#) - サーバから切断します。

[Refresh](#) - GUI を最新情報に更新します。

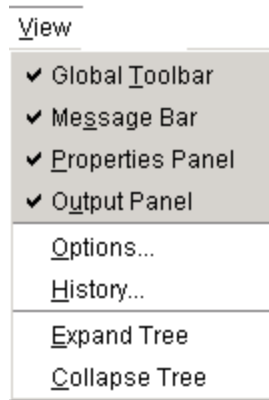
## [View] メニュー

[View Logs](#) - 接続しているサーバについて、LifeKeeper Single Server Protection のログメッセージを表示します。

[Create Resource Hierarchy](#) - リソース階層を作成します。

[Properties](#) - [\[Server Properties\] ダイアログ](#)を表示します。

## [View] メニュー



[Global Toolbar](#) - チェックボックスがオンの場合、このコンポーネントを表示します。

[Message Bar](#) - チェックボックスがオンの場合、このコンポーネントを表示します。

[Properties Panel](#) - チェックボックスがオンの場合、このコンポーネントを表示します。

[Output Panel](#) - チェックボックスがオンの場合、このコンポーネントを表示します。

[Options](#) - GUI の表示プロパティを編集します。

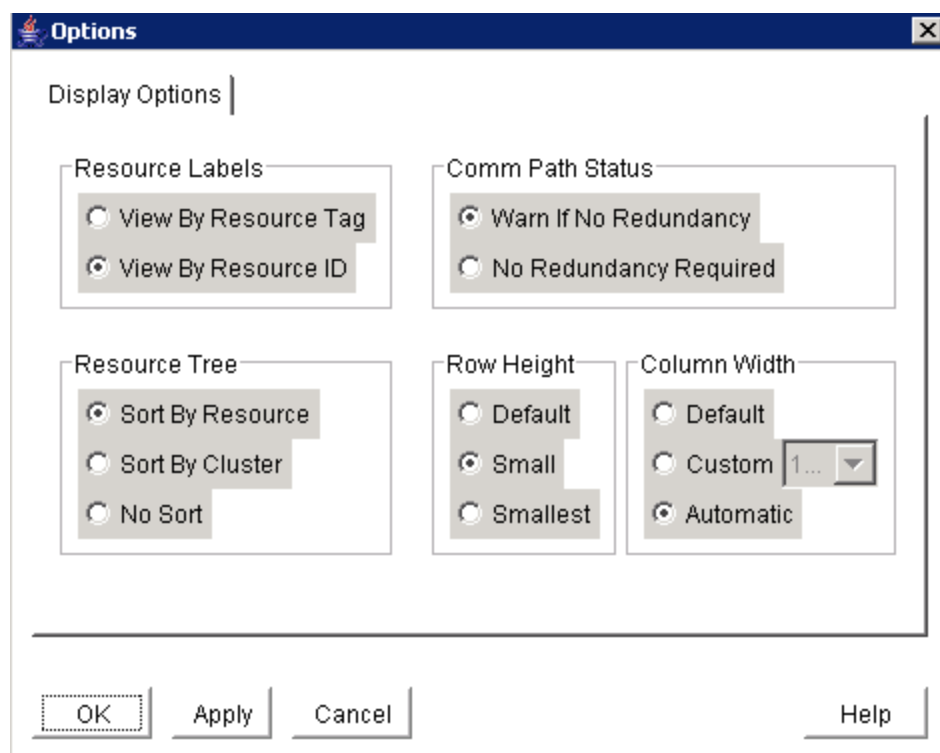
[History](#) - メッセージバーに表示された最新メッセージを、LifeKeeper の GUI の [\[Message History\] ダイアログ](#) ボックスに表示します (最大 1000 行)。

[Expand Tree](#) - リソース階層ツリー全体を展開します。

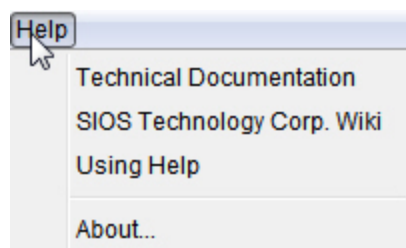
[Collapse Tree](#) - リソース階層ツリー全体を折り畳みます。

## [View Options] ダイアログ

**[View Options]** ダイアログは、**[View]** メニューから表示できます。**[Display Options]** タブで、GUI のさまざまな表示形式を指定できます。



## [Help] メニュー



**Technical Documentation** - すべてのドキュメンテーションへのリンクを持つテクニカルドキュメンテーションの開始ページを表示します。

**Using Help** - テクニカルドキュメンテーションの概要を表示します。

**About...** - LifeKeeper GUI のバージョン情報を表示します。

## ツールバー

このセクションで説明する GUI ツールバーから、管理機能を使用できます。

## GUI ツールバー

このツールバーは、[プロパティパネル](#)に表示されるデフォルトの[サーバ](#)と[リソース](#)のコンテキストツールバーを組み合わせたものですが、このツールバーから動作を実行するときには、サーバとリソースを選択する必要があります。








	<a href="#">Connect</a> - LifeKeeper Single Server Protection サーバに接続します。
	<a href="#">Disconnect</a> - LifeKeeper Single Server Protection サーバを切断します。
	Refresh - GUI を最新情報に更新します。
	<a href="#">View Logs</a> - 接続しているサーバについて、LifeKeeper Single Server Protection のログメッセージを表示します。
	<a href="#">Create Resource Hierarchy</a> - リソース階層を作成します。
	<a href="#">Delete Resource Hierarchy</a> - LifeKeeper Single Server Protection サーバからリソース階層を削除します。
	<a href="#">In Service</a> - リソース階層を in service にします。
	<a href="#">Out of Service</a> - リソース階層を out of service にします。
	<a href="#">Create Dependency</a> - 2つのリソース間に親 / 子の関係を作成します。
	<a href="#">Delete Dependency</a> - 2つのリソース間にある親 / 子の関係を削除します。

## リソースのコンテキストツールバー

[ステータスの表](#)からサーバ固有のリソースインスタンスを選択すると、[プロパティパネル](#)にリソースのコンテキストツールバーが表示されます。

選択したサーバとリソースについて、動作が呼び出されます。灰色表示のリソースについて、動作を選択することはできません。



	<a href="#">In Service</a> - リソース階層を in service にします。
	<a href="#">Out of Service</a> - リソース階層を out of service にします。
	<a href="#">Add Dependency</a> - 2つのリソース間に親 / 子の関係を作成します。
	<a href="#">Remove Dependency</a> - 2つのリソース間にある親 / 子の関係を削除します。
	<a href="#">Delete Resource Hierarchy</a> - リソース階層をすべてのサーバから削除します。

## サーバのコンテキスト ツールバー

[ステータスの表](#) からサーバを選択すると、[プロパティパネル](#)にサーバのコンテキストツールバーが表示されます。選択したサーバについて、動作が呼び出されます。



	<a href="#">Disconnect</a> - LifeKeeper Single Server Protection サーバから切断します。
	Refresh - GUI を最新情報に更新します。
	<a href="#">View Logs</a> - 接続しているサーバについて、LifeKeeper Single Server Protection のログメッセージを表示します。
	<a href="#">Create Resource Hierarchy</a> - リソース階層を作成します。
	<a href="#">Delete Resource Hierarchy</a> - LifeKeeper Single Server Protection サーバからリソース階層を削除します。

## LifeKeeper GUI のメッセージ履歴

メッセージバーに表示されたメッセージの履歴が、**[LifeKeeper GUI Messages History]** ダイアログに表

示されます。履歴リストには、最大 1000 行を表示できます。最大行数を超えた場合、新しいメッセージにより最も古いメッセージが「押し出され」ます。

これらのメッセージは、クライアントとサーバとの間の動作のみを表し、時系列で表示されます。最新のメッセージがリストの上部に表示されます。

<-- は、メッセージがサーバから受信したことを示し、通常は以下の形式をとります。

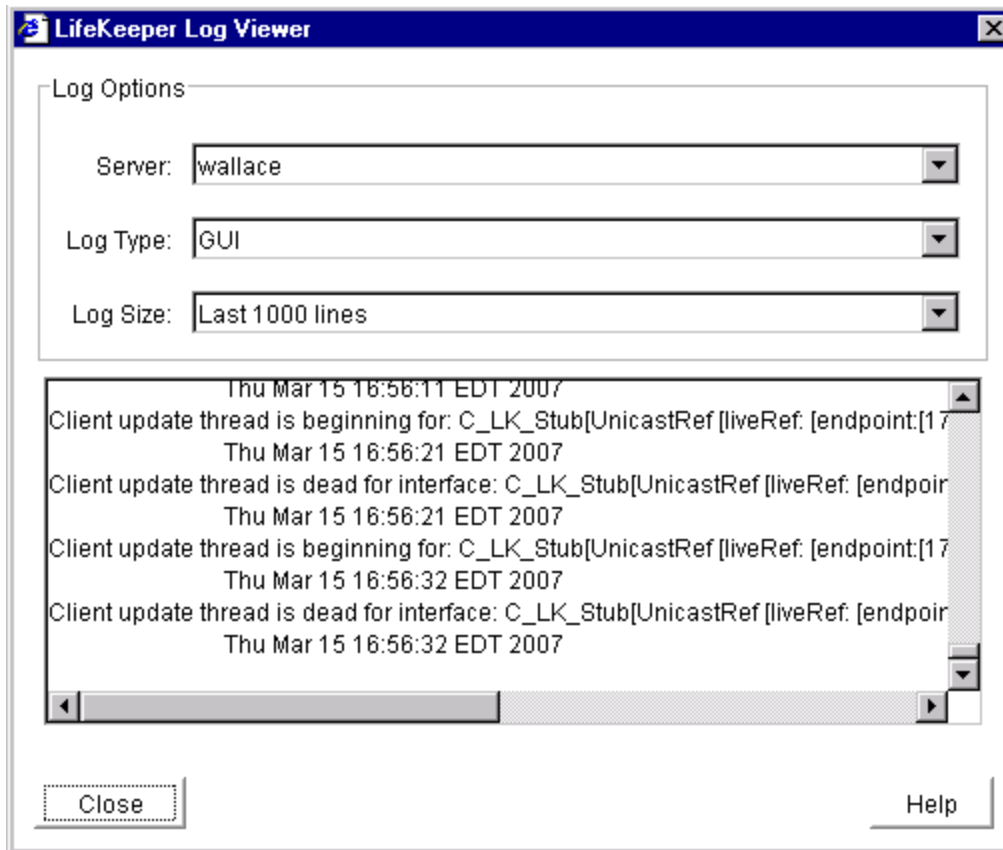
```
<--"server name": "action"  
<--"server name": "app res": "action"  
<--"server name": "res instance": "action"
```

--> はメッセージがクライアントから送信されたことを示し、通常は以下の形式をとります。

```
-->"server name": "action"  
-->"server name": "app res": "action"  
-->"server name": "res instance": "action"
```

[Clear] ボタンをクリックすると、履歴が消去されますが、ダイアログは閉じません。

[OK] ボタンをクリックすると、履歴を消去せずにダイアログが閉じます。



## GUI の実行の準備

このセクションのトピックでは、LifeKeeper のグラフィカルユーザインターフェースの準備について説明します。

## LifeKeeper GUI ソフトウェアパッケージ

LifeKeeper GUI は、LifeKeeper Single Server Protection Core パッケージクラスターにバンドルされている **steel-eye-1kGUI** ソフトウェアパッケージに含まれています。**steel-eye-1kGUI** パッケージは、以下の動作を実行します。

- LifeKeeper GUI アプリケーションクライアントをインストールする。
- LifeKeeper GUI サーバをインストールする。
- LifeKeeper Single Server Protection 管理 Web サーバをインストールする。**注記**: LifeKeeper Single Server Protection 管理 Web サーバは、パブリック Web サーバとは異なるポート 81 を使用するように設定されます。
- ディレクトリ `/opt/LifeKeeper/htdocs/` に Java ポリシーファイルをインストールする。このファイルには、LifeKeeper GUI アプリケーションクライアントの実行に必要な最小限の権限があります。LifeKeeper GUI アプリケーションクライアントは、この場所にある `java.policy` ファイルを使用して、アクセスを制御します。
- GUI 管理用に LifeKeeper Single Server Protection を準備する。

続行する前に、LifeKeeper Single Server Protection サーバに LifeKeeper GUI パッケージがインストール済みであることを確認する必要があります。コマンド `rpm -qi steel-eye-1kGUI` を入力して、このパッケージがインストール済みかどうかを確認できます。GUI パッケージがインストール済みである場合、出力にパッケージ名 `steel-eye-1kGUI` が表示されます。

## LifeKeeper Single Server Protection GUI の設定

### GUI 管理用の LifeKeeper Single Server Protection サーバの設定

各 LifeKeeper Single Server Protection サーバについて、以下の手順を実行してください。各手順には、詳細手順の参照先またはリンクがあります。

1. 各サーバに、Java 実行時環境 (JRE) または Java ソフトウェア開発キット (JDK) をインストールする必要があります。必要な Java のバージョンおよび必要なダウンロードにアクセスするための URL については、LifeKeeper Single Server Protection リリースノートを参照してください。**注記**: JRE は、LifeKeeper Single Server Protection の IMG ファイルから、設定スクリプトを実行し、JRE のインストールのみを選択することでインストールできます (詳細については、LifeKeeper Single Server Protection ISO イメージの使用を参照)。
2. 各サーバで、LifeKeeper GUI サーバを開始してください ([GUI サーバの開始/停止](#) を参照)。**注記**: GUI サーバが後続の初期インストールを開始した後、LifeKeeper Single Server Protection の開始または停止を行うと、GUI サーバを含む LifeKeeper Single Server Protection のすべてのデーモンプロセスの開始または停止が実行されます。

3. root 以外のユーザに GUI の使用を許可するように計画している場合は、[GUI ユーザの設定](#)が必要です。

## GUI の実行

LifeKeeper Single Server Protection サーバ上で LifeKeeper の GUI を実行できます。

GUI の設定と実行を行う方法については、[LifeKeeper Single Server Protection サーバでの GUI の実行](#)を参照してください。

## GUI の設定

項目	説明
GUI のクライアントとサーバの通信	LifeKeeper GUI のクライアントとサーバは、通信に Java のリモートメソッド呼び出し (RMI) を使用します。RMI が正しく動作するためには、クライアントとサーバは解決可能なホスト名または IP アドレスを使用する必要があります。DNS が実装されていない場合 (または、他の名前解決メカニズムを使用して名前が解決できない場合) は、クライアントとサーバのそれぞれについて、 <code>/etc/hosts</code> ファイルを編集し、他のすべての LifeKeeper Single Server Protection サーバの名前とアドレスを含めてください。
GUI サーバの Java プラットフォーム	LifeKeeper GUI サーバには、Java 実行時環境 (JRE) - Java 仮想マシン、Java プラットフォームのコアクラス、およびサポートするファイル-をインストールする必要があります。JRE 5.0 for Linux は、LifeKeeper Single Server Protection IMG ファイル上にあります。または、直接 <a href="http://www.oracle.com/technetwork/java/archive-139210.html">http://www.oracle.com/technetwork/java/archive-139210.html</a> からダウンロードできます。 注記: デフォルトでは、LifeKeeper GUI サーバは、JRE が各サーバのディレクトリ <code>/usr/java/j2re1.5.0_07</code> にインストールされていると予測します。JRE が見つからない場合、GUI サーバはディレクトリ <code>/usr/java/j2sdk1.5.0_07</code> から Java ソフトウェア開発キット (JDK) を探します。JRE または JDK を別のディレクトリの場所を使用する場合は、LifeKeeper Single Server Protection のデフォルトファイル <code>/etc/default/LifeKeeper</code> の PATH を編集し、Java インタープリタ <code>java.exe</code> を含むディレクトリを含めてください。このファイルの編集時に LifeKeeper Single Server Protection が実行中である場合は、変更内容を認識させるために LifeKeeper GUI サーバを停止し、再起動する必要があります。再起動しない場合、LifeKeeper GUI は Java コマンドを見つけることができません。
Java リモートオブジェクトレジストリのサーバポート	LifeKeeper GUI サーバは、各 LifeKeeper Single Server Protection サーバ上の Java リモートオブジェクトレジストリ用にポート 82 を使用します。これにより、サーバは、典型的なファイアウォールの後にあるクライアントからの RMI 呼び出しをサポートできます。
LifeKeeper Single Server Protection のグラフィカルユーザインターフェース	LifeKeeper GUI サーバには、クライアントのブラウザの通信用に管理 Web サーバが必要です。現在、LifeKeeper GUI サーバは、管理 Web サーバとして <code>lighttpd</code> Web サーバのプライベートコピーを使用しています。この Web サーバは、 <code>steeleye-lighttpd</code> パッケージによりインストールと設定が実行され、他の Web サーバとの競合を避けるためにポート 81 を使用します。

## GUI の制限

項目	説明
GUI の相互運用性の制限	LifeKeeper Single Server Protection for Linux クライアントは、Linux サーバ上の LifeKeeper Single Server Protection の管理にのみ使用できます。LifeKeeper for Linux の GUI と LifeKeeper for Windows は、同時には使用できません。

## GUI サーバの開始および停止

### LifeKeeper GUI サーバを開始するには

LifeKeeper GUI サーバが動作していない場合は、*root* として以下のコマンドを入力してください。

```
/opt/LifeKeeper/bin/lkGUIserver start
```

このコマンドは、管理しているサーバで LifeKeeper GUI サーバのデーモンプロセスが現在動作していない場合、それらのデーモンプロセスをすべて開始します。以下のようなメッセージが表示されます。

```
# Installing GUI Log
# LK GUI Server Startup at:
# Mon May 8 14:14:46 EDT 2006
# LifeKeeper GUI Server Startup completed at:
# Mon May 8 14:14:46 EDT 2006
```

LifeKeeper GUI サーバが開始した後、LifeKeeper Single Server Protection を開始すると、必ず LifeKeeper GUI サーバのプロセスが自動的に開始します。

### トラブルシューティング

LifeKeeper GUI は、各サーバのポート 81 を管理 Web サーバ用に、ポート 82 を Java リモートオブジェクトレジストリに使用します。他のアプリケーションがそれらのポートを使用している場合、LifeKeeper GUI は正しく機能しません。これらの値は、LifeKeeper Single Server Protection のデフォルトファイル */etc/default/LifeKeeper* の以下のエントリを編集することにより変更できます。

```
GUI_WEB_PORT=81 GUI_RMI_PORT=82
```

**注記:** これらのポートの値は、起動時に GUI サーバで初期化されます。ポートの値を変更した場合、GUI サーバを停止し、再起動する必要があります。

### LifeKeeper GUI サーバを停止するには

LifeKeeper GUI サーバが動作している場合は、*root* として以下のコマンドを入力してください。

```
/opt/LifeKeeper/bin/lkGUIserver stop
```

このコマンドは、管理しているサーバで LifeKeeper GUI サーバのデーモンプロセスが現在動作している場合、それらのデーモンプロセスをすべて停止します。以下のメッセージが表示されます。

## LifeKeeper GUI サーバのプロセス

```
# LifeKeeper GUI Server Shutdown at:
# Fri May 19 15:37:27 EDT 2006
# LifeKeeper GUI Server Shutdown Completed at:
# Fri May 19 15:37:28 EDT 2006
```

## LifeKeeper GUI サーバのプロセス

LifeKeeper GUI サーバが動作していることを確認するには、以下のコマンドを入力してください。

```
ps -ef | grep runGuiSer
```

以下のような出力が表示されます。

```
root    2805    1 0 08:24 ? 00:00:00 sh/opt/LifeKeeper/bin/runGuiSer
```

現在動作している他の GUI サーバのデーモンプロセスのリストを表示するには、以下のコマンドを入力してください。

```
ps -ef | grep S_LK
```

以下のような出力が表示されます。

```
root    769    764 0 Oct16 ?
00:00:00/usr/jre1.2.2/bin/i386/green_threads/rmiregistry -J-DS_
LK=true 82

root    819    764 0 Oct16 ?
00:00:00/usr/jre1.2.2/bin/i386/green_threads/java -DS_LK=true -
0ss3m -ss3m-Dcom.steeleye.Li
```

## GUI ユーザの設定

GUI ユーザには 3 つのクラスがあり、それぞれ権限が異なります。

1. **Administrator (管理者)** の権限を持つユーザは、GUI から可能な動作のすべてを実行できます。
2. 1 台のサーバ上で **Operator (オペレータ)** の権限を持つユーザは、LifeKeeper Single Server Protection の設定やステータスの情報を表示でき、そのサーバ上のリソースを in service にしたり、out of service にしたりすることができます。
3. 1 台のサーバ上で **Guest (ゲスト)** の権限を持つユーザは、そのサーバの LifeKeeper Single Server Protection の設定やステータスの情報を表示できます。

GUI サーバは、*root* として起動する必要があります。GUI パッケージのインストール時に、*root* のログインとパスワードのエントリが、**Administrator** の権限付きで GUI パスワードファイルに自動設定されるので、*root* は、そのサーバから GUI アプリケーションまたは Web クライアント経由で LifeKeeper Single Server Protection のすべての作業を実行できます。*root* 以外のユーザに LifeKeeper GUI クライアントの使用を許可するように計画している場合は、LifeKeeper GUI のユーザを設定する必要があります。

ユーザ管理は、以下に示すように、コマンドラインインターフェースから *lkpasswd* を使用して実行します。特記ない限り、すべてのコマンドで、ユーザのパスワードを 2 回入力する必要があります。変更内容は、ユーザの次のログイン時または GUI サーバの再起動時 (いずれかの早い時点) で有効になり

ます。各ユーザは、1台のサーバにつき権限を1つ持ちます。サーバで新しい権限が指定された場合、以前の権限エントリは削除されます。

- ユーザに LifeKeeper GUI の **Administrator** 権限を付与するには、以下のコマンドを入力してください。

```
/opt/LifeKeeper/bin/lkpasswd -administrator <user>
```

- ユーザに LifeKeeper GUI の **Operator** 権限を付与するには、以下のコマンドを入力してください。

```
/opt/LifeKeeper/bin/lkpasswd -operator <user>
```

- ユーザに LifeKeeper GUI の **Guest** 権限を付与するには、以下のコマンドを入力してください。

```
/opt/LifeKeeper/bin/lkpasswd -guest <user>
```

- ユーザの権限レベルを変更せずに、既存のユーザのパスワードを変更するには、以下のコマンドを入力してください。

```
/opt/LifeKeeper/bin/lkpasswd <user>
```

- 既存のユーザに LifeKeeper GUI の使用を禁止するには、以下のコマンドを入力してください。

```
/opt/LifeKeeper/bin/lkpasswd -delete <user>
```

- このコマンドでは、パスワードを入力する必要はありません。

## Java のセキュリティポリシー

LifeKeeper の GUI は、ポリシーベースのアクセス制御を使用します。GUI クライアントのロード時に、現在有効なセキュリティポリシーに基づいて権限が GUI クライアントに割り当てられます。ポリシーはさまざまな署名者 / 場所からのコードに提供される権限を指定し、外部から設定可能なポリシーファイルから初期化されます。

デフォルトでは、システム全体のポリシーファイルとオプションのユーザポリシーファイルが1つずつあります。システム全体でコードに権限を付与するシステムポリシーファイルが先にロードされ、次にユーザポリシーファイルが追加されます。LifeKeeper GUI がアプリケーションとして起動される場合は、これらのポリシーファイルに加えて、LifeKeeper GUI のポリシーファイルもロードされることがあります。

### ポリシーファイルの場所

デフォルトでは、システムポリシーファイルは以下の場所にあります。

```
<JAVA.HOME>/lib/security/java.policy (Linux)
```

```
<JAVA.HOME>\lib\security\java.policy (Windows)
```

**注記:** JAVA.HOME は、システムのプロパティ「JAVA.HOME」の値を指し、JRE または JDK がインストールされたディレクトリの場所を指定します。

ユーザポリシーファイルは「.」の文字で始まり、デフォルトでは以下の場所にあります。

```
<USER.HOME>\.java.policy
```

## ポリシーファイルの作成と管理

**注記:** USER.HOME は、システムのプロパティ「user.home」の値を指し、ユーザのホームディレクトリを指定します。例えば、Windows NT ワークステーション上にあるユーザ Paul のホームディレクトリは、「paul.000」です。

Windows システムの場合、user.home のプロパティ値のデフォルト値は以下のとおりです。

C:\WINNT\Profiles\Windows NT システム)

C:\WINDOWS\Profiles\Windows 95/98 システム)

C:\WINDOWS (シングル ユーザ **Windows 95/98** システム)

デフォルトでは、LifeKeeper GUI のポリシーファイルは以下の場所にあります。

/opt/LifeKeeper/htdocs/java.policy (**Linux**)

## ポリシーファイルの作成と管理

デフォルトでは、LifeKeeper GUI がアプリケーションとして起動される場合に LifeKeeper GUI のポリシーファイルが使用されます。LifeKeeper GUI をアプレットとして実行する場合、ホームディレクトリにユーザポリシーファイルを作成する必要があります (存在しない場合)。ユーザポリシーファイルは、LifeKeeper GUI を実行するために必要な最低限の権限を指定する必要があります。このトピックの「ポリシーファイルの例」セクションで後述します。

ポリシーファイルの作成と管理は、単純なテキストエディタ、または Java 実行時環境 (JRE) や Java 開発キット (JDK) に含まれるグラフィカルな **Policy Tool** ユーティリティから行うことができます。Policy Tool を使用すると、入力が簡略化され、ポリシーファイルに必要な構文の知識が不要になります。Policy Tool の使用方法の詳細については、<http://docs.oracle.com/javase/1.3/docs/tooldocs/tools.html>にある Policy Tool のドキュメンテーションを参照してください。

LifeKeeper GUI を実行するために必要な最低限の権限を持つ**ユーザポリシーファイルを作成する最も簡単な方法**は、`/opt/LifeKeeper/htdocs/java.policy`にある LifeKeeper GUI のポリシーファイルをホームディレクトリにコピーし、ファイル名を `.java.policy` に変更することです (ファイル名の前にあるドットは必須)。Windows システムでは、ファイル `http://<server name>:81/java.policy` (<server name> は LifeKeeper Single Server Protection サーバのホスト名) を開いてホームディレクトリに `.java.policy` の名前を付けて保存することで、LifeKeeper GUI のポリシーファイルをコピーできます。ユーザポリシーファイルの正しい場所を特定する必要がある場合は、Java のコントロールパネルを使用して Java コンソールを有効にし、LifeKeeper GUI をアプレットとして起動します。ユーザポリシーファイルのホームディレクトリのパスが、Java コンソールに表示されます。

## ポリシーファイルでの権限の付与

権限は、システムリソースへのアクセスを表します。アプレットにリソースへのアクセスを許可するには、対応する権限を、アクセスを試行するコードに明示的に付与する必要があります。権限は通常、名前を持ち (「ターゲット名」として参照される)、場合によっては、1つ以上の動作を含むカンマ区切りリストを持ちます。例えば、以下のコードは、`/tmp` ディレクトリのファイル `abc` に対する読み取りアクセスを表す `FilePermission` オブジェクトを作成します。

```
perm = new java.io.FilePermission("/tmp/abc", "read");
```

この例では、ターゲット名は「`/tmp/abc`」、動作文字列は「`read`」です。

ポリシーファイルは、指定したコードソースからのコードに許可する権限を指定します。この例で、`/home/sysadmin` ディレクトリのコードにファイル `/tmp/abc` への読み取りアクセスを付与するポリシーファイルのエントリは以下のとおりです。

```
grant codeBase "file:/home/sysadmin/" {
    permission java.io.FilePermission "/tmp/abc", "read"; };
```

## ポリシーファイルの例

このポリシーファイルの例には、LifeKeeper GUI の実行に必要な最小限の権限があります。このポリシーファイルは、LifeKeeper GUI パッケージにより `/opt/LifeKeeper/htdoc/java.policy` にインストールされます。

```
/*
 * LifeKeeper GUI に必要な権限。コードベースで
 * これを制限することもできます。ただし、そのようにする場合は、リカバリキットが
 * 任意のコードベース付きの任意の jar コンポーネントを持つことができることに
 * 注意してください。したがって、これらも含めるには
 * grant 文を変更する必要があります。
 */
grant {

/*
 * LifeKeeper クラスタ内のすべてのマシンに対して
 * これを行うことができなければなりません。それに合わせてネットワーク仕様を
 * 制限することもできます。
 */
permission java.net.SocketPermission "*" , "accept,connect,resolve";
/*
 * リモートプロパティファイルおよび
 * jar ファイルを取得するには、URLClassLoaders を使用してください。
 */
permission java.lang.RuntimePermission "createClassLoader";
/*
 * GUI をアプリケーションとして実行する場合のみ以下が必要です
 * (デフォルトの RMI セキュリティマネージャは、
 * ブラウザがアプレット用にインストールするセキュリティマネージャよりも
 * 制限あり)。
 */
permission java.util.PropertyPermission "*" ,"read";
permission java.awt.AWTPermission "*" ;
permission java.io.FilePermission "<<ALL FILES>>" ,"read,execute";

};
```

## Java プラグイン

Netscape 6/7、Mozilla 1.x、Firefox 1.x、Internet Explorer 6/7 のいずれを使用する場合でも、ブラウザが初めて LifeKeeper GUI のロードを試行するときには、Java プラグインソフトウェアを自動ダウンロードするか、Java プラグインソフトウェアのダウンロードとインストールを行う Web ページを表示します。その後

は、Java プラグインソフトウェアのテクノロジーをサポートする Web ページに遭遇するたびに、ブラウザは Java プラグインソフトウェアを自動的に起動します。

## Java プラグインのダウンロード

Java プラグインソフトウェアは、Solaris、Linux、および Windows の Java 実行時環境 (JRE) の一部として含まれています。JRE のダウンロードには、お使いのネットワークとシステム設定のサイズにより、合計で 3 ~ 10 分かかります。ダウンロードの Web ページには、JRE と Java プラグインソフトウェアについての詳細なドキュメンテーションとインストール手順があります。

**注記 1:** プラグインのインストール後、およびプラグインのプロパティを変更するたびに、ブラウザを閉じて再起動する必要があります。

**注記 2:** LifeKeeper Single Server Protection は、Java プラグインのバージョン 1.3.x 以降のみをサポートしています。

## Java プラグインのトラブルシューティング

Netscape 6/7、Mozilla 1.x、または Firefox 1.x を使用している場合、Netscape、Mozilla、または Firefox のプラグインのディレクトリに、`$JAVAHOME` ディレクトリの `libjavaplugin_oji.so` ファイルのパスへのシンボリックリンクの作成が必要になることがあります。

例 (Firefox 1.5 と jre 1.5):

```
cd /usr/lib/mozilla/plugins
ln -s/usr/java/jre1.5.0_07/plugin/i386/ns7/libjavaplugin_oji.so
```

Netscape 4 で、Java プラグインを含む Java 実行時環境をインストールしたにもかかわらずブラウザが Java プラグインを検出しない場合は、`NPX_PLUGIN_PATH` 環境変数を Java プラグインの場所 (`javaplugin.so` ファイルがある場所) に設定してください。つまり、`NPX_PLUGIN_PATH=$JAVAHOME/jre/plugin/i386/ns4` をエクスポートしてください (`$JAVAHOME` は Java 実行時環境をインストールした最上位のディレクトリ)。

Java プラグインは、Java 2 SDK、標準エディション v1.3 のセキュリティモデルをサポートしています。アプレットはすべて、標準アプレットセキュリティマネージャの下で実行されます。詳細については、[Java のセキュリティの FAQ](#)、または GUI アプレットを使用するための[ブラウザのセキュリティパラメータの設定](#)を参照してください。

一部のプラットフォーム/ブラウザの組み合わせでは、Java プラグインソフトウェアが、LifeKeeper の GUI にある Java コンポーネント (スクロールバー、ツールバー、メニューなど) の表示と動作に影響します。これらの多くの状況では、回避策として、ウィンドウのサイズを変更したり、最小化 / 最小化解除 (強制再表示) したりすると問題が解決します。

## GUI アプレットのブラウザセキュリティパラメータの設定

警告: セキュリティを低い値に設定した状態での他のサイトの閲覧には注意してください。

## Netscape Navigator と Netscape Communicator

1. **[Edit]** メニューの **[Preferences]** を選択します。
2. **[Preferences]** ダイアログボックスの **[Advanced Category]** をダブルクリックします。
3. **[Enable Java]** と **[Enable Java Script]** のオプションを選択します。
4. **[OK]** をクリックします。

## Firefox

1. **[Edit]** メニューの **[Preferences]** を選択します。
2. **[Preferences]** ダイアログボックスの **[Content]** を選択します。
3. **[Enable Java]** と **[Enable Java Script]** のオプションを選択します。
4. **[Close]** をクリックします。

## Internet Explorer

セキュリティが最高の状態で Internet Explorer を使用するには、以下の手順で LifeKeeper Single Server Protection サーバを信頼済みサイトのゾーンに追加してください。

1. **[Tools]** メニューの **[Internet Options]** をクリックします。
2. **[Security]** タブをクリックします。
3. **[Trusted Sites]** ゾーンを選択し、**[Custom Level]** をクリックします。
4. **[Reset custom settings]** の **[Medium/Low]** を選択し、**[Reset]** をクリックします。
5. **[Sites]** をクリックします。
6. 接続する LifeKeeper Single Server Protection サーバのサーバ名とポート番号を入力します (例: http://server1:81)。

以下の手順で行う別の方法 (セキュリティが低くなる可能性がある) もあります。

1. **[Tools]** メニューの **[Internet Options]** をクリックします。
2. **[Internet]** または **[Local Intranet]** を選択します。
3. **[Security Level]** バーを **[Medium]** ([Internet] を選択した場合)、または **[Medium-low]** ([Local Intranet] を選択した場合) に調整します。これらは、各ゾーンのデフォルト設定です。
4. **[OK]** をクリックします。

## LifeKeeper Single Server Protection サーバでの GUI の実行

LifeKeeper GUI を実行する最も簡単な方法は、LifeKeeper Single Server Protection サーバでアプリ

ケーションとして実行することです。これにより、事実上、GUI クライアントとサーバを同一システム上で実行できます。

## LifeKeeper Single Server Protection サーバでの GUI の実行

1. GUI 管理用の LifeKeeper Single Server Protection サーバを設定した後、*root* として以下のコマンドを入力することにより、サーバ上で GUI をアプリケーションとして実行できます。

```
/opt/LifeKeeper/bin/lkGUIapp
```

1. *lkGUIapp* スクリプトが適切な環境変数を設定して、アプリケーションを開始します。アプリケーションのロード時に、アプリケーション ID のダイアログが表示されます。
2. アプリケーションのロード後、LifeKeeper の GUI が表示され、接続ダイアログが自動的に表示されます。接続先のサーバ名、およびログインとパスワードを入力してください。
3. 接続が確立した後、GUI のウィンドウに、接続しているサーバにより保護されているリソースとステータスがグラフィックで表示されます。GUI のメニューとツールバーのボタンから、管理機能を使用できます。

## リモートシステムでの GUI の実行

LifeKeeper GUI を Java アプレットとして実行することにより、LifeKeeper Single Server Protection サーバ外の Linux、Unix、または Windows のシステムから LifeKeeper Single Server Protection の管理ができます。リモートでの LifeKeeper GUI の設定と実行について説明します。

## リモートシステムでの GUI の設定

リモートの Unix または Windows のシステムで LifeKeeper GUI を実行するには、使用するブラウザが JDK 1.4 アプレットをフルにサポートする必要があります。LifeKeeper GUI でサポートされているプラットフォームおよびブラウザについては、LifeKeeper Single Server Protection リリースノートを参照してください。

1. LifeKeeper GUI をアプレットとして実行する場合、ホームディレクトリに Java ユーザポリシーファイルを作成する必要があります (存在しない場合)。ユーザポリシーファイルには、LifeKeeper GUI の実行に必要な最小限の権限を指定する必要があります。
  - LifeKeeper GUI を実行するために必要な最小限の権限を持つ**ユーザポリシーファイルを作成する最も簡単な方法**は、`/opt/LifeKeeper/htdocs/java.policy`にある LifeKeeper GUI のポリシーファイルをホームディレクトリにコピーし、ファイル名を **.java.policy** に変更します (ファイル名の前にあるドットは必須)。Windows システムでは、ファイル `http://<server name>:81/java.policy` (<server name> は LifeKeeper Single Server Protection サーバのホスト名) を開いてホームディレクトリに `java.policy` の名前を付けて保存することで、LifeKeeper GUI のポリシーファイルをコピーできます。ユーザポリシーファイルの正しい場所を特定する必要がある場合は、Java のコントロールパネルを使用して Java コンソールを有効にし、LifeKeeper GUI をアプレットとして起動してください。ユーザポリシーファイルのホームディレクトリのパスが、Java コンソールに表示されます。
  - ユーザポリシーファイルがすでにある場合は、LifeKeeper Single Server Protection

サーバの `/SLKROOT/htdocs/java.policy` に指定されている必須エントリを、単純なテキストエディタを使用して既存のファイルに追加できます。詳細については、[Java のセキュリティポリシー](#) トピックを参照してください。

2. ブラウザのセキュリティパラメータを低に設定する必要があります。この設定では通常、Java と Java アプレットが有効になります。さまざまなブラウザとバージョンが存在するので、ブラウザのセキュリティパラメータの設定手順は[GUI アプレットを使用するためのブラウザのセキュリティパラメータの設定](#) トピックで説明しています。**注記:** セキュリティを低く設定した状態で外部サイトを閲覧するときには、注意が必要です。
3. GUI を初めて実行するときに Netscape または Internet Explorer を使用し、かつ必要な Java プラグインがシステムにない場合、プラグインをダウンロードする Web サイトが自動的に表示されません。必要な Java プラグインのバージョンとダウンロードにアクセスするための URL については、LifeKeeper Single Server Protection リリースノートを参照してください。

## リモートシステムでの GUI の実行

上記の作業を完了すると、リモートシステムで LifeKeeper GUI を Java アプレットとして実行できます。

1. LifeKeeper GUI の Web ページの URL `http://<server name>:81` (<server name> は LifeKeeper Single Server Protection サーバの名前) を開いてください。この Web ページには、LifeKeeper Single Server Protection のスプラッシュ画面とアプレットがあります。Web ページが開くと、以下の動作が実行されます。
  - アプレットがロードされる。
  - Java 仮想マシンが開始される。
  - 一部のサーバファイルがダウンロードされる。
  - アプレットが初期化される。

ネットワークとシステムの設定によっては、これらの動作に最大 20 秒かかることがあります。通常、アプレットのロード時と初期化時に、ブラウザには最小のステータスがいくつか表示されます。

すべてのものが正しくロードされた場合、アプレット領域に **[Start]** ボタンが表示されます。スプラッシュ画面に **[Start]** ボタンが表示されない場合、またはアプレットのロードと初期化が失敗した疑いがある場合は、「アプレットのトラブルシューティング」またはネットワーク関連のトラブルシューティングを参照してください。

2. 要求されたら、**[Start]** をクリックしてください。LifeKeeper の GUI が表示され、[Server Connect] ダイアログが自動的に表示されます。サーバが開始され、接続が確立した後、GUI のウィンドウに、接続しているサーバにより保護されているリソースとステータスがグラフィックで表示されます。GUI のメニューとツールバーのボタンから、LifeKeeper Single Server Protection の管理機能を使用できます。

**注記:** 一部のブラウザでは、アプレットで作成されたウィンドウとダイアログに「**Warning: Applet Window**」が表示されます。これは正常なので、無視してください。

## アプレットのトラブルシューティング

アプレットのロードと初期化に失敗した疑いがある場合は、以下の操作を試してください。

## 共通の作業

1. アプレットが失敗したことを確認してください。通常、アプレットの状態を示すブラウザウィンドウ内に、メッセージが出力されます。NetscapeとInternet Explorerでは、テキストのステータスに加えて、アプレットの代わりにアイコンが表示されることがあります。アイコンをクリックすると、失敗の内容が表示される場合があります。
2. Java プラグインをインストールしていることを確認してください。問題がJava プラグインに関連する場合は、[Java プラグイン](#)のトピックを参照してください。
3. ブラウザの設定要件、特にセキュリティ設定を満たしていることを確認してください。詳細については、[GUI アプレットを使用するためのブラウザのセキュリティパラメータの設定](#)を参照してください。設定について明らかな間違いが見つからない場合は、次の手順に進んでください。
4. Java コンソールを開いてください。
  - Firefox、Netscape、および旧バージョンのInternet Explorerの場合は、マシンのコントロールパネルから **Java プラグイン** アプレットを実行し、コンソールを表示するオプションを選択してから、ブラウザを再起動してください。
  - 最新バージョンのInternet Explorerの場合は、**[Tools] > [Sun Java Console]** を選択してください。[Sun Java Console]のメニュー項目が表示されない場合は、**[Tools] > [Manage Add-Ons]** を選択し、コンソールを有効にしてください。その後、コンソールを表示するには、ブラウザの再起動が必要になることがあります。
  - Mozillaの場合は、**[Tools] > [Web Development] > [Sun Java Console]** を選択してください。
5. URL `http://<server name>:81` を再度開いて、GUI アプレットを開始してください。Java プラグインのコンソールパネルを変更した場合は、ブラウザを再起動してください。
6. コンソールに表示されたメッセージを確認してください。メッセージは、問題の解決に役立ちます。問題がネットワークに関係している場合は、ネットワーク関連のトラブルシューティングを参照してください。

## 共通の作業

このセクションでは、すべてのユーザが実行できる基本的なタスクについて説明します。

## サーバからの切断

この作業は、LifeKeeper Single Server Protection サーバから GUI クライアントを切断します。

1. 開始するには、以下の3つの方法があります。
  - [グローバルツールバー](#)の **[Disconnect]** ボタンをクリックする。
  - [\[Edit\] メニュー](#)で **[Server]** を選択し、次に **[Disconnect]** をクリックする。
  - [サーバコンテキストツールバー](#)が表示される場合は、その **[Disconnect]** ボタンをクリックする。
2. 切断するサーバの名前を選択します。

3. [OK] をクリックします。サーバのリストが [Confirmation] ダイアログに表示されます。

4. [Confirmation] ダイアログの [OK] をクリックして、サーバからの切断を確定します。

サーバから切断すると、そのサーバが GUI のステータス表示から消去されます。

## 接続サーバの表示

サーバの状態は、下図に示すように、表内のサーバのグラフィック表示で表されます。サーバアイコンが視覚的に示すサーバの状態の詳細については、[サーバのステータスの表示](#)を参照してください。




## サーバのステータスの表示

サーバの状態は、下図に示すように、表内のサーバのグラフィック表示で表されます。



サーバの状態	状態のシンボル	意味
ALIVE		クライアントはサーバに有効な接続を行うことができます。 このサーバから ALIVE のリモートサーバへのコミュニケーションパスが ALIVE です。 DEAD とマークされたコミュニケーションパス、および DEAD のサーバをターゲットとするコミュニケーションパスは無視されます。これは、DEAD のサーバには DEAD のグラフィックで表されるからです。
ALIVE		クライアントはサーバに有効な接続を行うことができます。 このサーバから指定リモートサーバへの 1 つ以上のコミュニケーションパスが DEAD です。 このサーバから指定リモートサーバへの間には、冗長コミュニケーションパスが存在しません。
DEAD		DEAD として報告されました。

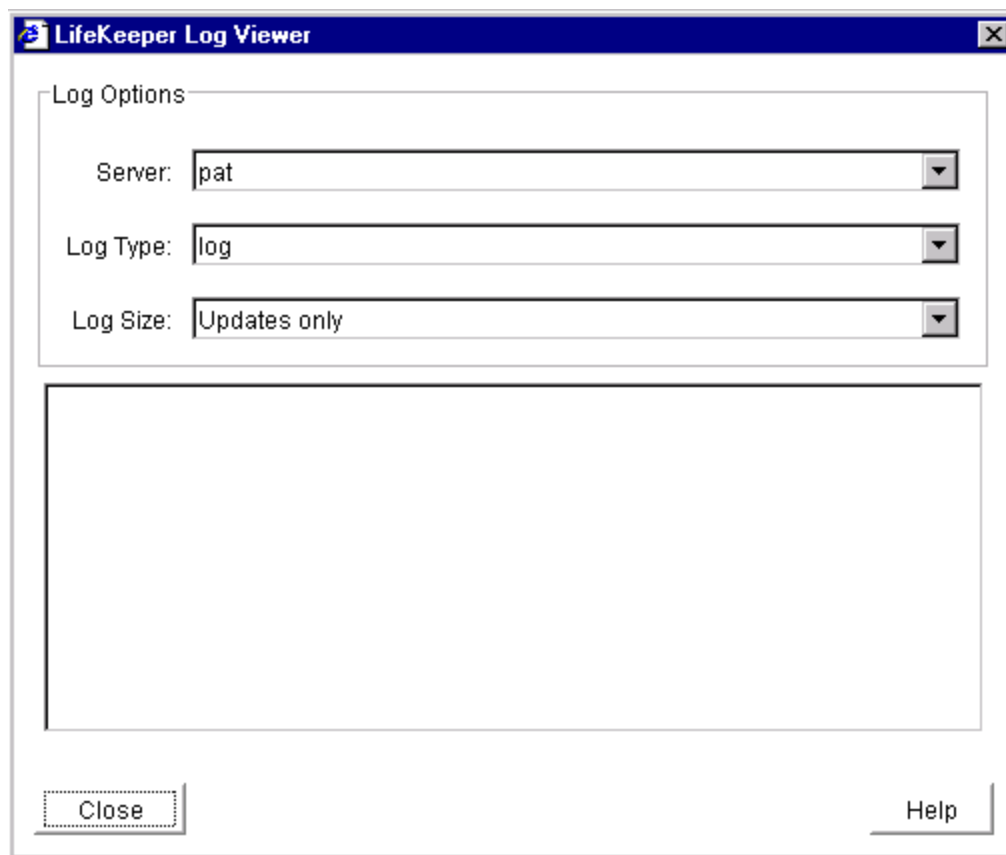
UNKNOWN		ネットワーク接続が失われました。前回の LifeKeeper Single Server Protection の状態は ALIVE です。
---------	---	--

## サーバログファイルの表示

1. 開始するには、以下の4つの方法があります。
  - サーバのアイコンを右クリックして[サーバのコンテキストメニュー](#)を表示し、次に **[View Log]** をクリックして [LifeKeeper Single Server Protection Log Viewer] ダイアログを表示する。
  - [グローバルツールバー](#)の **[View Log]** ボタンをクリックし、[LifeKeeper Single Server Protection Log Viewer](#) ダイアログの [Server] リストから、表示するサーバを選択する。
  - [サーバのコンテキストツールバー](#)が表示される場合は、その **[View Log]** ボタンをクリックする。
  - [\[Edit\] メニュー](#)の **[Server]** をポイントし、**[View Log]** をクリックする。次に、**[LifeKeeper Single Server Protection Log Viewer]** ダイアログの **[Server]** リストから、表示するサーバを選択する。
2. [グローバルツールバー](#)または **[Edit] メニュー**から操作を開始して、別のサーバのログを表示する場合は、[LifeKeeper Single Server ProtectionLog Viewer] ダイアログの **[Server]** リストから、そのサーバを選択します。サーバの[コンテキストメニュー](#)または[サーバのコンテキストツールバー](#)から **[View Log]** を選択した場合は、この機能は使用できません。
3. 確認が完了したら、**[OK]** をクリックして **[Log Viewer]** ダイアログを閉じてください。

## [Log Viewer] ダイアログ

プロパティパネルが有効で、かつサーバのプロパティが表示される場合、**[Log Viewer] ダイアログ**には、[サーバコンテキストメニュー](#)、[グローバルツールバー](#)、または[サーバコンテキストツールバー](#)からアクセスできます。このダイアログには、LifeKeeper Single Server Protection が管理しているログファイルの一部が表示されます。ツールバーまたは **[View]** メニューを使用してダイアログを開いた場合、[Server] リストから選択したサーバのログファイルを参照できます。



**Server** - クラスタに接続したサーバの一覧がドロップダウンリストに表示されます。参照するログファイルが保存されているサーバを選択してください。サーバコンテキストメニューを使用してダイアログを開いた場合、この一覧は表示されません。

**Log Type** - 選択したサーバに保存されているログファイルの一覧がドロップダウンリストに表示されません。参照するログファイルを指定するオプションを選択してください。

- ログ
- TTYLCM
- LCM
- LCD
- remote\_execute
- GUI
- SNMP
- NOTIFY

**Log Size** - 4つのオプションがドロップダウンリストに表示されます。表示するログファイルの行数を指定するオプションを選択してください。

## サーバプロパティの表示

- Updates Only (更新のみ)
- Last 100 lines (最新の 100 行)
- Last 500 lines (最新の 500 行)
- Last 1000 lines (最新の 1000 行)

## サーバプロパティの表示

1. 開始するには、以下の2つの方法があります。
  - プロパティを表示するサーバのアイコンを右クリックします。[サーバのコンテキストメニュー](#)が表示されたら、**[Properties]** をクリックします。サーバのプロパティは、サーバをクリックすると[プロパティパネル](#) (有効になっている場合) にも表示されます。
  - [\[Edit\] メニュー](#)の **[Server]** をポイントし、**[Properties]** をクリックします。ダイアログが表示されたら、表示するサーバを [Server] リストから選択します。
2. 別のサーバのプロパティを表示する場合は、サーバを **[Server]** リストから選択してください。
3. 確認が完了したら、**[OK]** をクリックしてウィンドウを閉じてください。

## リソースタグおよび ID の表示

リソースのタグとIDを即座に表示するには、[Status] ウィンドウのリソースアイコンにカーソルを合わせて、マウスの左ボタンを1回押します(シングルクリック)。優先順位が最も低いサーバのリソースのタグとIDがメッセージバーに表示されます。特定サーバ上にあるリソースのタグとIDを表示する場合は、表内のリソースインスタンスセルを左クリックしてください。

メッセージバーに表示されるメッセージは、以下ようになります。

```
Resource Tag = ipdnet0-153.98.87.73, Resource ID = IP-153.98.87.73
```

特定の状況では、GUIがリソースIDを特定できないことがあります。この場合は、リソースタグのみがメッセージバーに表示されます。






## リソースのステータスの表示

リソースのステータス(状態)は2つの形式で表示されます。**階層リソースのステータス**(左側のペイン)にはリソースの依存関係がツリー形式で表示され、**サーバリソースのステータステーブル**には、各サーバの個々のリソースのステータスが表形式で表示されます。

## サーバリソースのステータステーブル

下図に、リソースステータスを持つサーバを示します。

 castor	 galapagos	 jailbird
 Active 1		
	 Active 1	
	 Active 1	
		 Active 1
		 Active 1
		 Warning 1
		 Active 1

サーバリソースの状態	状態のシンボル	意味
アクティブ		リソースは動作可能であり、保護されています。(ISP)
警告		リソースは動作可能ですが、警告があります。(ISU)
スタンバイ		リソースは out of service です。(OSU)
障害		このサーバ上のリソースに問題が検出されました。例えば、リソースを in-service にする試行が失敗しました。(OSF)
不明		リソースが初期化されていないか (ILLSTATE)、このサーバで LifeKeeper Single Server Protection が動作中ではありません。
	空のパネル	サーバのリソースが定義されていません。

## リソースのプロパティの表示

リソースのプロパティを表示するには、2通りの方法があります。

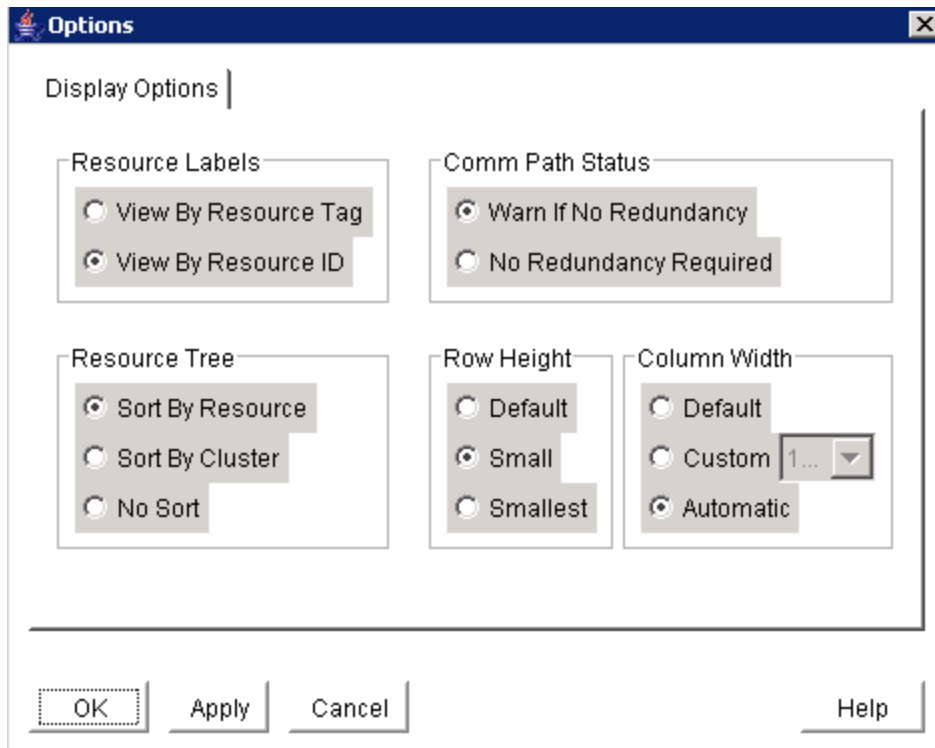
## [Status] ウィンドウの表示オプションの設定

- [プロパティパネル](#)を有効にし、プロパティを表示するリソースをクリックする。
- プロパティを表示するリソースのアイコンを右クリックする。[リソースのコンテキストメニュー](#)が表示されたら、**[Properties]**をクリックする。リソースのプロパティは、[プロパティパネル](#) (有効になっている場合)にも表示されます。

## [Status] ウィンドウの表示オプションの設定

[Options] ダイアログは、**[View]**メニューから表示できます。[Options] ダイアログで、LifeKeeper Single Server Protection のさまざまな表示形式を指定できます。これらの設定、およびチェックボックスのメニュー項目の全ての設定とさまざまなウィンドウサイズは、クライアントマシンのホームフォルダにあるファイル *.lkGUIpreferences* で複数のセッションにわたって保存されます。このファイルは、Web クライアントとアプリケーションクライアントの両方が使用します。各クライアントマシンの優先設定は、他のマシンの優先設定から独立しています。2台のマシンで優先設定を同期する場合は、優先ファイルを恒常的に共有するか、一時的にマシン間でコピーを移動します。

1. [\[View\]メニュー](#)の**[Options]**をクリックしてください。[View Options] ダイアログが表示されます。



2. [Status] ウィンドウでのリソースの表示位置を変更するには、**[Display Options]** タブをクリックし、変更するオプショングループを選択してください。以下に示すオプショングループの詳細説明を参照してください。
3. **[OK]** をクリックして設定を保存し、[Status] ウィンドウに戻ってください。

## Resource Labels

このオプショングループを使用すると、リソース階層ツリー内のリソースを、タグ名別とID別のいずれかで表示するかを指定できます。

**注記:** リソース階層ツリーに表示されるリソースタグ/IDは、優先順位が最も低い番号を持つサーバに属します。特定サーバ上にあるリソースのタグ/IDを表示する場合は、表内のリソースインスタンスセルを左クリックしてください。メッセージバーにそのタグ/IDが表示されます。

**By tag name:**



**By ID:**



## Resource Tree

このオプショングループを使用すると、リソース階層ツリー内に表示するリソースのソート順序を指定できます。

- **Sort By Resource** - リソースラベルのみを基準にしてリソースをソートします。
- **No Sort** - ソートが無効にします。リソースは、GUIが検出した順序で表示されます。

リソース階層ツリーの上位リソースは、ツリー内のリソースを左クリックして新しい位置に「ドラッグ」することで手動ソートできます。順序は、移動したリソース、およびツリー内の移動先の位置によって異なります。

**注記:** 「0」と「9」のキーは、リソース階層ツリーの展開/折り畳みを即座に実行するホットキー/アクセラレータキーとして指定されています。マウスも、ツリー全体の展開や折り畳みに使用できます。リソース階層ツリーのタイトル領域をクリックし、ダブルクリックするとツリーが展開します。クリックするとツリーが折り畳まれます。

## Row Height

このオプショングループを使用すると、表内の行の高さを指定できます。選択肢は **[Default]**、**[Small]**、および **[Smallest]** です。

**注記:**「+」と「-」のキーは、リソース階層ツリー内と表内にあるリソースのサイズを即座に変更するホットキー/アクセラレータキーとして指定されています。

## Column Width

このオプショングループを使用すると、表内のサーバとリソースの列幅を指定できます。選択肢は以下のとおりです。

- **Default:** 標準の幅。
- **Custom:** ドロップダウンリストから幅 (ピクセル単位) を選択できます。
- **Automatic:** 使用可能な領域全体に収まるように、すべての列のサイズを自動変更します。

**注記:**「7」と「8」のキーは、リソース階層の表内にあるリソース列のサイズを即座に変更するホットキー/アクセラレータキーとして指定されています。

## メッセージ履歴の表示

1. [\[View\] メニュー](#)の **[History]** をクリックしてください。[LifeKeeper GUI の \[Message History\]](#) ダイアログが表示されます。
2. 履歴のメッセージをすべて消去する場合は、**[Clear]** をクリックしてください。
3. ダイアログを閉じるには、**[OK]** をクリックしてください。

**[Message History]** ダイアログには、メッセージバーからの最新のメッセージが表示されます。履歴リストには、最大 1000 行を表示できます。最大行数を超えた場合、新しいメッセージにより最も古いメッセージが「押し出され」ます。

これらのメッセージは、クライアントとサーバとの間の動作のみを表し、時系列で表示されます。最新のメッセージがリストの上部に表示されます。

## メッセージ履歴の解釈

<- は、メッセージがサーバから受信したことを示し、通常は以下の形式をとります。

```
<-"server name": "action"  
<-"server name": "app res": "action"  
<-"server name": "res instance": "action"
```




-> はメッセージがクライアントから送信されたことを示し、通常は以下の形式をとります。

```
->"server name": "action"  
->"server name": "app res": "action"  
->"server name": "res instance": "action"
```



**[Clear]** ボタンをクリックすると、履歴が消去されますが、ダイアログは閉じません。

**[OK]** ボタンをクリックすると、履歴を消去せずにダイアログが閉じます。

## リソース階層ツリーの展開および折り畳み

 <p>The screenshot shows a tree structure with three nodes, each with a green checkmark icon. The top node is 'file_system_2' with an expanded icon (a square with a right-pointing arrow) to its left. Below it is 'device-nfs18047' with an expanded icon. The bottom node is 'nfs-/opt/qe_auto/NFS/export1' with a collapsed icon (a square with a down-pointing arrow) to its left.</p>	<p>このツリーのセグメントでは、リソース <code>file_system_2</code> が展開されており、リソース <code>nfs-/opt/qe_auto/NFS/export1</code> が折り畳まれています。</p> <p> は、ツリーを展開すると、リソースアイコンの左側に表示されます。</p> <p> は、折りたたまれている場合に表示されます。</p>
--	--

リソース階層ツリーを展開するには、



-  をクリックするか、
-  の右側にあるリソースアイコンをダブルクリックする。

リソース階層ツリーをすべて展開するには、

- **[View] メニューの [Expand Tree]** をクリックするか、
- **[Status] ウィンドウの左ペインにある列ヘッダの [Resource Hierarchy Tree]** ボタンをダブルクリックしてください。

**注記:** リソース階層ツリーに表示されるリソースタグ/ID は、優先順位が最も低い番号を持つサーバに属します。特定サーバ上にあるリソースのタグ/ID を表示する場合は、表内のリソースインスタンスセルを左クリックしてください。メッセージバーにそのタグ/ID が表示されます。

リソース階層ツリーを折り畳むには、

-  をクリックするか、
-  の右側にあるリソースアイコンをダブルクリックする。

リソース階層ツリーをすべて折り畳むには、

- **[View] メニューの [Collapse Tree]** をクリックするか、
- **[Status] ウィンドウの左ペインにある列ヘッダの [Resource Hierarchy Tree]** ボタンをダブルクリックしてください。

**注記:** 「9」と「0」のキーは、すべてのリソース階層ツリーに対して即座に展開 / 折り畳みを実行するホットキー / アクセラレータキーとして指定されています。

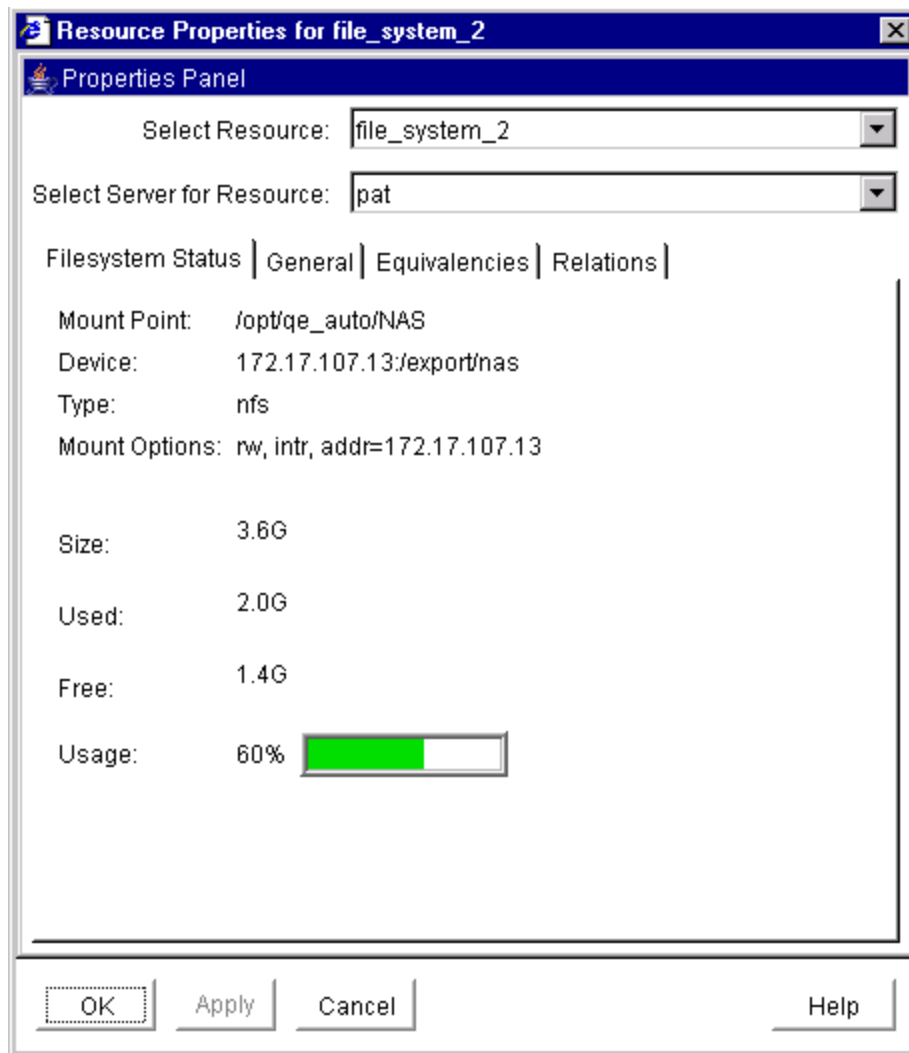
## [Resource Properties] ダイアログ

[Resource Properties] ダイアログは、[\[Edit\] メニュー](#)や[リソースコンテキストメニュー](#)から使用できます。このダイアログには、サーバ上にある特定のリソースのプロパティが表示されます。[Edit] メニューからアクセスした場合は、リソースとサーバを選択できます。リソースコンテキストメニューからアクセスした場合は、

## [General] タブ

サーバを選択できます。

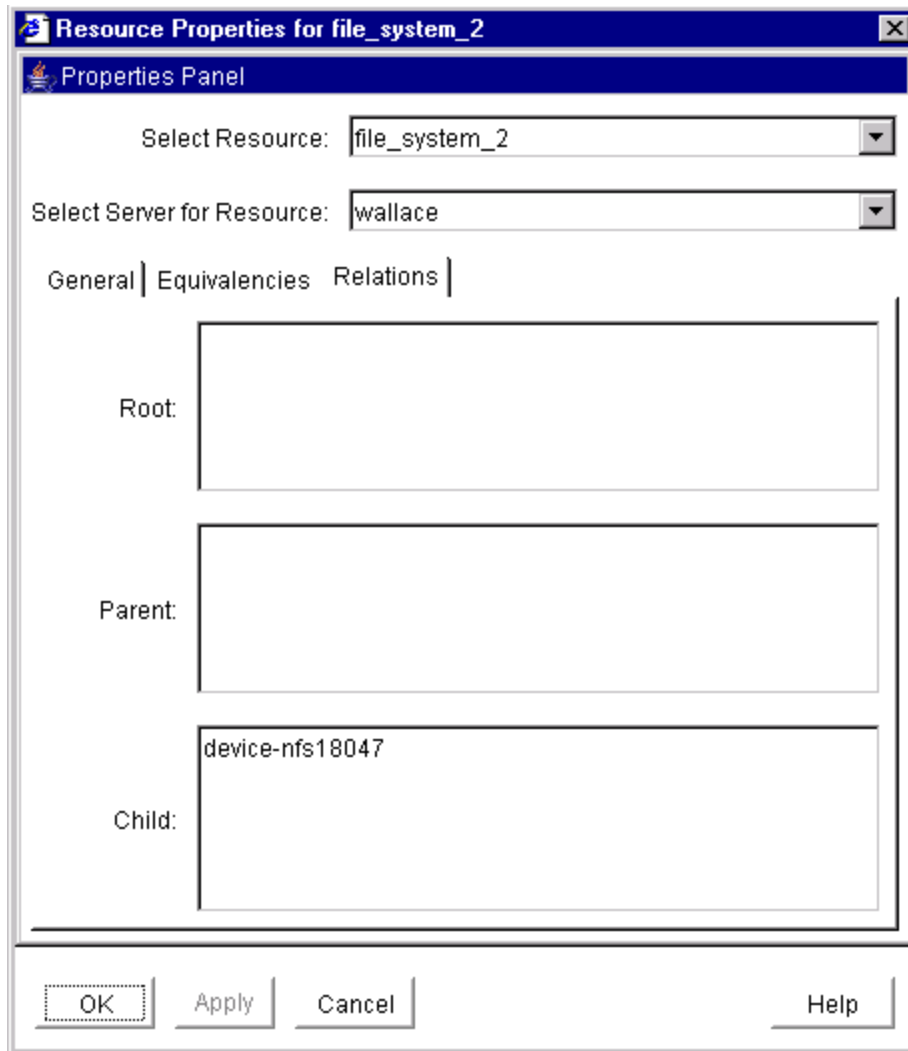
## [General] タブ



- **Tag** - リソースインスタンスの名前。システムに対して一意で、管理者にリソースを示します。
- **ID** - リソースインスタンスに関連する文字列であり、リソースタイプのすべてのインスタンス間で一意です。関連するアプリケーションソフトウェアに対して、リソースインスタンスの内部特性のいくつかを示します。
- **State** - リソースインスタンスの現在の状態。
  - **Active** - ローカルで in service であり、保護されています。
  - **Warning** - ローカルで in service ですが、ローカルリカバリは試行されません。

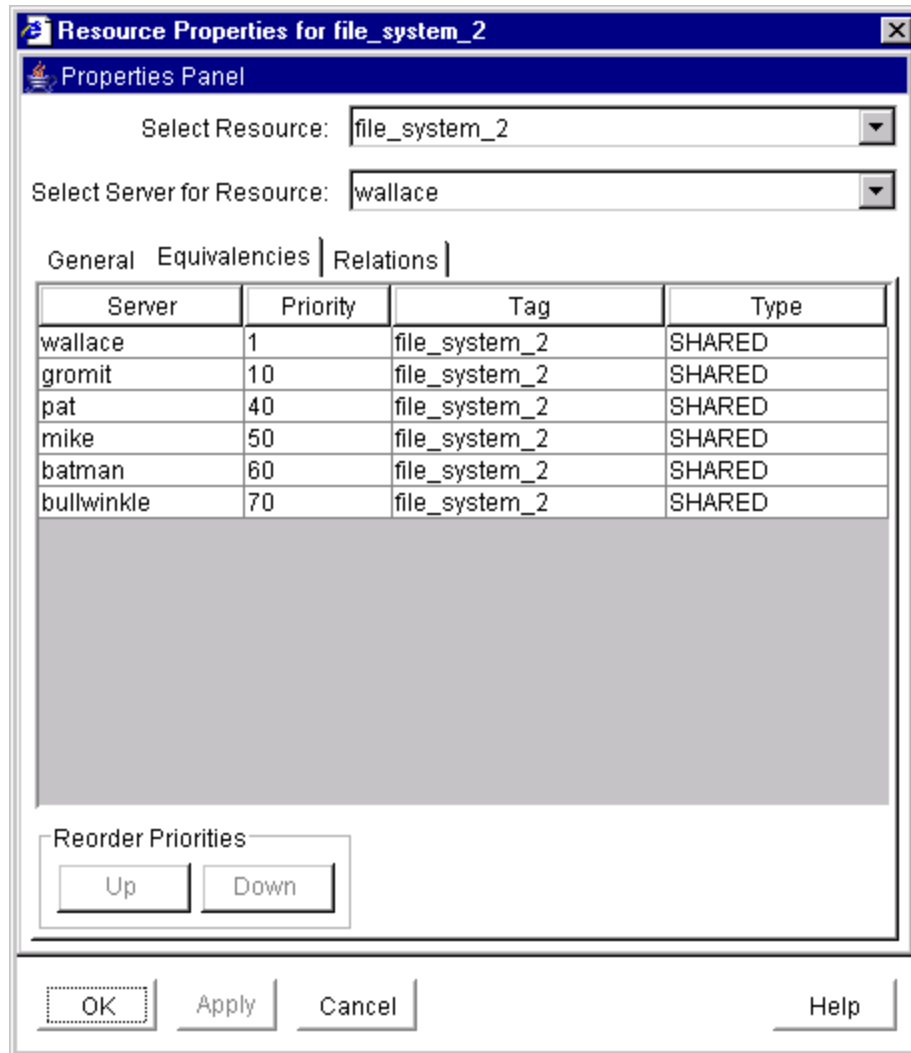
- **Failed** - out of service、障害あり。
- **Standby** - out of service、障害なし。
- **ILLSTATE** - LifeKeeper Single Server Protection の起動シーケンスの一部として実行されるリソース初期化プロセスにより、リソースの状態が適切に初期化されていません。この状態のリソースは、LifeKeeper Single Server Protection で保護されていません。
- **UNKNOWN** - リソースの状態を特定できませんでした。GUI サーバが使用できない可能性があります。
- **Reason** - 存在する場合、リソースが現在の状態にある原因 (つまり、最後の状態変化の原因) を示します。例えば、galahad 上にあるアプリケーションの状態が OSU である原因は、tristan 上にある共有プライマリリソース *ordbf saa-on-tristan* の状態が ISP か ISU であることです。共有リソースは、グループ内の 1 つのシステムでのみ同時にアクティブにできます。
- **初期化**。起動時のリソースの初期化動作を決定する設定であり、`AUTORES_ISP`、`INIT_ISP`、`INIT_OSU` などがあります。

## [Relations] タブ



- **Parent** - このリソースに直接依存するリソースのタグ名を示します。
- **Child** - このリソースが依存するすべてのリソースのタグ名を示します。
- **Root** - このリソース階層で、親を持たないリソースのタグ名。

## [Equivalencies] タブ



- **Server** - リソースが定義済みの同等性を持つサーバ名。
- **Tag** - 同等のサーバ上にあるこのリソースのタグ名。
- **Type** - 同等性のタイプ (SHARED、COMMON、COMPOSITE)。
- **Reorder Priorities** (管理者権限を持つユーザは編集可能) - [Up] または [Down] ボタンを選択すると、現在ハイライト表示されている行が1つ上または下の行と入れ替わります。

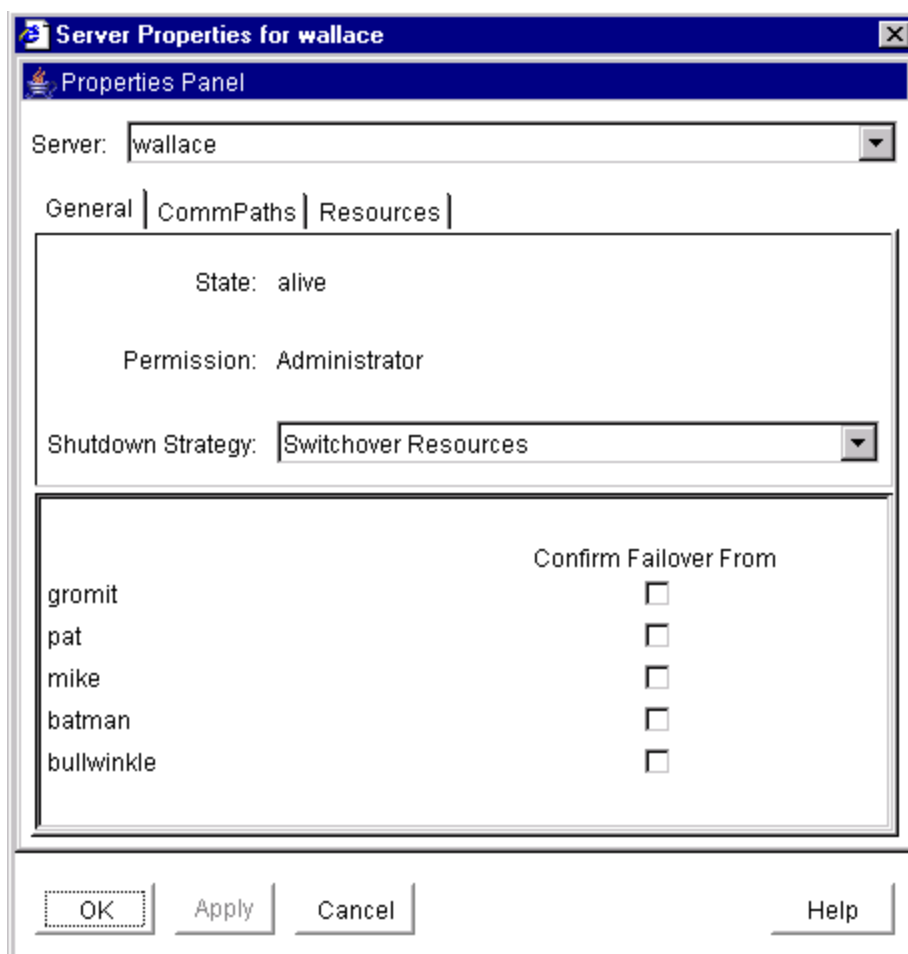
[OK] ボタンをクリックすると、変更内容が適用されてウィンドウが閉じます。[Apply] ボタンをクリックすると、変更内容が適用されます。[Cancel] ボタンをクリックすると、最後に [Apply] をクリックして以降の変更内容を保存せずに、ウィンドウが閉じます。

## [Server Properties] ダイアログ

[Server Properties] ダイアログは、サーバのコンテキストメニューや[Edit]メニューから使用できます。このダイアログには、特定のサーバのプロパティが表示されます。サーバのプロパティは、[プロパティパネル](#)(有効になっている場合)にも表示されます。

このダイアログの3つのタブについて説明します。[OK] ボタンをクリックすると、変更内容が適用されてウィンドウが閉じます。[Apply] ボタンをクリックすると、変更内容が適用されます。[Cancel] ボタンをクリックすると、最後に[Apply] をクリックして以降の変更内容を保存せずに、ウィンドウが閉じます。

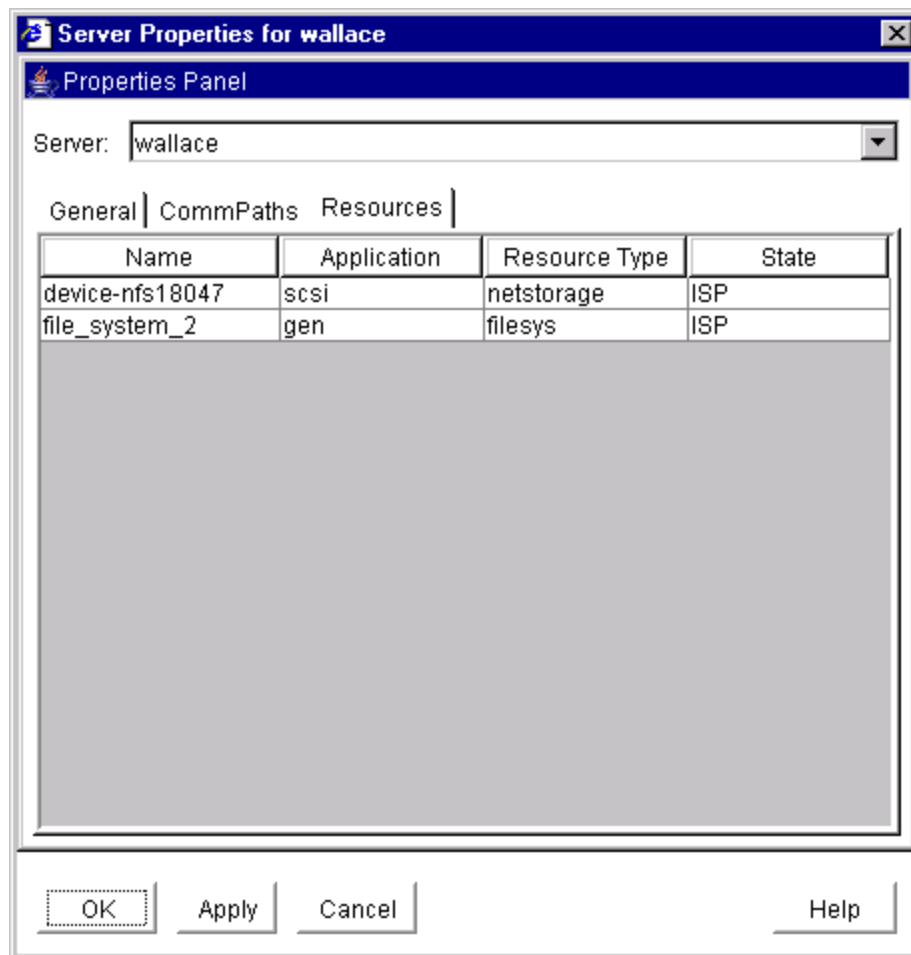
### [General] タブ



- **Name** - 選択したサーバの名前。
- **State** - サーバの現在の状態。サーバの状態は以下の値をとります。

- *ALIVE* - サーバが使用可能。
  - *DEAD* - サーバが使用不可。
  - *UNKNOWN* - リソースの状態を特定できませんでした。GUI サーバが使用できない可能性があります。
- **Permission** - そのサーバに現在ログインしているユーザの権限レベル。権限は以下の値をとります。
    - *Administrator* - ユーザは LifeKeeper Single Server Protection のすべての作業を実行できます。
    - *Operator* - ユーザは、LifeKeeper Single Server Protection のリソースとサーバのステータスを監視でき、リソースを in service や out of service にできます。
    - *Guest* - ユーザは LifeKeeper Single Server Protection のリソースとサーバのステータスを監視できます。

## [Resources] タブ



- **Name** - 選択したサーバ上にあるリソースインスタンスのタグ名。
- **Application** - リソースタイプのアプリケーション名 (gen、scsi など)。
- **Resource Type** - サービスを提供するリソースタイプ、ハードウェアのクラス、ソフトウェアのクラス、またはシステムのエンティティのクラス (app、fileysys、nfs、device、disk など)。
- **State** - リソースインスタンスの現在の状態。
  - *ISP* — ローカルで In-service であり、保護されています。
  - *ISU* - ローカルで In-service ですが、ローカルリカバリは試行されません。
  - *OSF* - Out-of-service、障害。
  - *OSU* - Out-of-service、障害なし。
  - *LLSTATE* - LifeKeeper Single Server Protection の起動シーケンスの一部として実行されるリソース初期化プロセスにより、リソースの状態が適切に初期化されていません。この状態のリソースは、LifeKeeper Single Server Protection で保護されていません。
  - *UNKNOWN* - リソースの状態を特定できませんでした。GUI サーバが使用できない可能性があります。

## オペレータの作業

以下のトピックは、オペレータの権限を必要とする高度な作業です。

- [リソースを In Service にする](#)
- [リソースを Out of Service にする](#)

## リソースを In Service にする

1. 開始するには、次の5つの可能な方法があります。
  - In Service にするリソース / サーバの組み合わせのアイコンを右クリックします。[リソースのコンテキストメニュー](#)が表示されたら、**[In Service]** をクリックします。
  - In Service にするグローバルリソースのアイコンを右クリックします。[リソースのコンテキストメニュー](#)が表示されたら、**[In Service]** をクリックします。ダイアログが表示されたら、in service にするリソースが存在するサーバを **[Server]** リストから選択し、**[Next]** をクリックします。
  - [グローバルツールバー](#)の **[In Service]** ボタンをクリックします。ダイアログが表示されたら、In Service にするリソースが存在するサーバを **[Server]** リストから選択し、**[Next]** をクリックします。次のダイアログで、in service にするリソースを1つ以上 **[Resource(s)]** リストから選択し、**[Next]** をもう一度クリックします。
  - [リソースのコンテキストツールバー](#)が表示される場合は、その **[In Service]** ボタンをクリックします。
  - [\[Edit\] メニュー](#)の **[Resource]** をポイントし、**[In Service]** をクリックします。ダイアログが表

示されたら、**In Service** にするリソースがあるサーバを **[Server]** リストから選択し、**[Next]** をクリックします。次のダイアログで、in service にするリソースを 1 つ以上 **[Resource(s)]** リストから選択し、**[Next]** をもう一度クリックします。

2. 選択したサーバとリソースを in service にすることを示すダイアログボックスが表示されます。親リソースを in service にせずに依存する子リソースを in service にしようとする場合、このダイアログには警告も表示されます。**[In Service]** をクリックして、依存する子リソースと共にリソースを in service にしてください。
3. **出力パネル**が有効の場合は、ダイアログが閉じ、リソースを in service にするコマンドの結果が**出力パネル**に表示されます。有効でない場合は、ダイアログが表示されたままこれらの結果が表示されます。すべての結果が表示されたら、**[Done]** をクリックして終了します。in service になった追加の依存 (子) リソースが、ダイアログまたは**出力パネル**に表示されます。
4. リソースを in service にする際に発生したエラーは、LifeKeeper Single Server Protection ログおよびリソースを in service にするサーバの GUI ログに記録されます。

## リソースを Out of Service にする

1. 開始するには、次の4つの可能な方法があります。
  - Out of Service にするグローバルリソース、またはリソース / サーバの組み合わせのアイコンを右クリックします。**リソースのコンテキストメニュー**が表示されたら、**[Out of Service]** をクリックします。
  - **グローバルツールバー**の **[Out of Service]** ボタンをクリックします。**[Out of Service]** ダイアログが表示されたら、Out of Service にするリソースを 1 つ以上 **[Resource(s)]** リストから選択し、**[Next]** をクリックします。
  - **リソースのコンテキストツールバー**が表示される場合は、**[Out of Service]** ボタンをクリックします。
  - **[Edit] メニュー**の **[Resource]** をポイントし、**[Out of Service]** をクリックします。**[Out of Service]** ダイアログが表示されたら、out of service にするリソースを 1 つ以上 **[Resource(s)]** リストから選択し、**[Next]** をクリックします。
2. 選択したリソースが Out of Service になることを示す **[Out of Service]** ダイアログボックスが表示されます。親リソースを Out of Service にせずに依存する子リソースを Out of Service にしようとする場合、このダイアログには警告も表示されます。**[Out of Service]** をクリックして、次のダイアログボックスに進みます。
3. **出力パネル**が有効の場合は、ダイアログが閉じ、リソースを Out of Service にするコマンドの結果が**出力パネル**に表示されます。出力パネルが無効の場合は、これらの結果を表示するダイアログが表示されたままになり、結果がすべて表示されたら **[Done]** をクリックします。
4. リソースを out of service にする際に発生したエラーは、LifeKeeper Single Server Protection ログおよびリソースを out of service にするサーバの GUI ログに記録されます。

## 高度な作業

### LCD

#### LifeKeeper 設定 データベース

LifeKeeper 設定 データベース (LCD) は、LifeKeeper Single Server Protection が既知のすべてのリソースタイプについて、オブジェクト指向のリソース階層情報を管理し、リカバリ方向の情報を保存します。データは共有メモリにキャッシュされ、ファイルに保存されるので、システムの再起動後もデータが保持されます。LCD には、リカバリが必要なリソースインスタンスについての状態の情報、および特定の詳細情報もあります。

LCD のディレクトリ構造、保存されるデータタイプ、使用できるリソースタイプ、およびアプリケーションスクリプトの使用の詳細については、以下の[関連トピック](#)および[その他のトピック](#)を参照してください。

#### LCD のディレクトリ構造

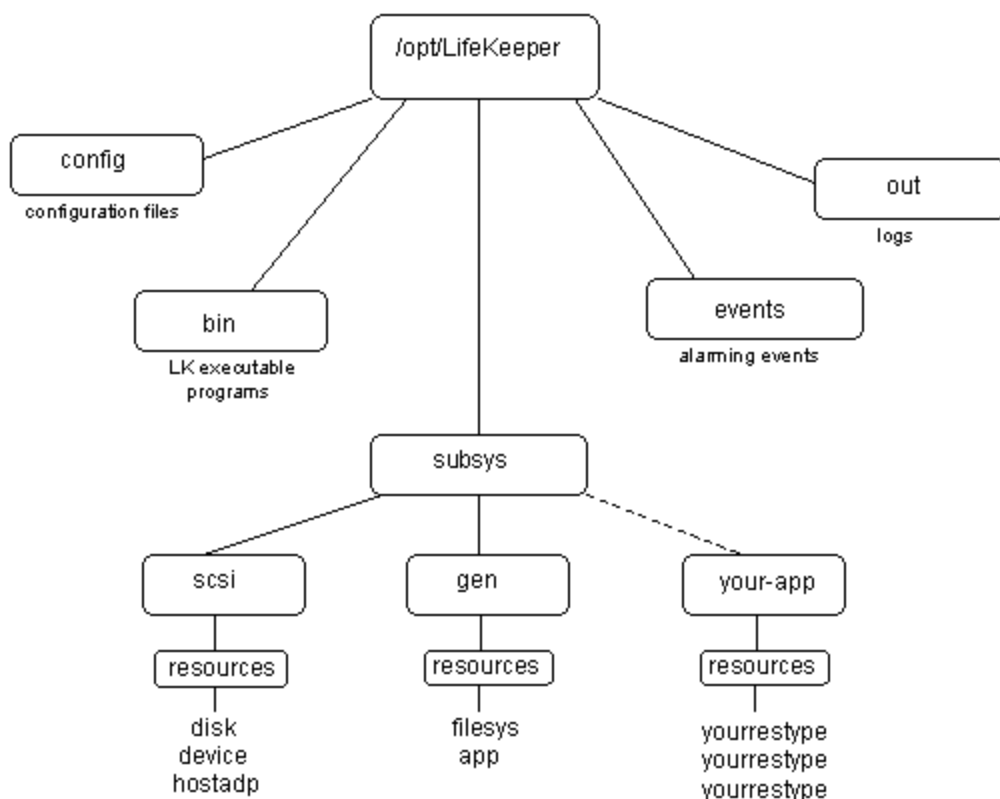
`/opt/LifeKeeper` の下にある主なサブディレクトリを示します。

- **config** - LifeKeeper Single Server Protection の設定ファイル。イクイバレンシ情報を含みます。
- **bin** - LifeKeeper Single Server Protection の実行可能プログラム。is\_recoverable などがあります。詳細については、[障害検出とリカバリのシナリオ](#)を参照してください。
- **subsys** - リソースとタイプ。LifeKeeper Single Server Protection は、リソースとタイプの定義を scsi で、Generic Application のメニュー機能を gen で提供します。アプリケーションのインターフェースを定義する場合は、subsys の下にディレクトリを作成してください。
- **events** - 警報イベント。[LifeKeeper Single Server Protection の警報とリカバリ](#)を参照してください。
- **out** - LifeKeeper Single Server Protection のログ。LifeKeeper Single Server Protection は、各種のエラーとメッセージをこのディレクトリの異なるログに送ります。lk\_log(8) マニュアルページを参照してください。

`/opt/LifeKeeper` 内の LCD ディレクトリの構造については、[/opt/LifeKeeper 内の LCD ディレクトリの構造](#)のトピックを参照してください。

#### `/opt/LifeKeeper` の LCD のディレクトリ構造

以下の図に、`/opt/LifeKeeper` のディレクトリ構造を示します。



## LCD 設定データ

LCD には、以下の関連するデータタイプが保存されます。

- 依存関係の情報
- リソースのステータス情報
- サーバ間の同等性の情報

### 依存関係の情報

定義した各リソースについて、LifeKeeper Single Server Protection は依存関係のリスト、および依存物 (あるリソースに依存するリソース) のリストを保持します。詳細については、LCDI\_relationship (1M) と LCDI\_instances (1M) のマニュアルページを参照してください。

### リソースのステータス情報

LifeKeeper は、各リソースインスタンスのステータス情報をメモリに保持します。LCD が認識するリソースの状態は、ISP、ISU、OSF、OSU、および ILLSTATE です。システムイベントが発生した場合、または管理者が特定の操作を行った場合に、リソースがある状態から別の状態に変化することがあります。

す。リソースの状態が変化した場合、ステータスの変化が、ローカルサーバの LCD、およびそのリソースのダイアログサーバ上にあるデータベースに反映されます。

## LCD のリソースタイプ

LCD は共有メモリ、および `/opt/LifeKeeper` ディレクトリの両方に保持されます。[ディレクトリ構造の図](#)に示すように、`subsys` には、アプリケーションインターフェースの指定に使用できるアプリケーションリソースセットが1つあります。

- `gen` - Generic Application とファイルシステムの情報

このサブディレクトリの詳細については、[リソースのサブディレクトリ](#)を参照してください。

## リソースのサブディレクトリ

`gen` ディレクトリには、リソースのサブディレクトリがあります。このディレクトリの内容は、LifeKeeper Single Server Protection が提供するリソースタイプのリストです。

**gen のリソースタイプ。**これらのリソースタイプは、`/opt/LifeKeeper/subsys/gen/resources` ディレクトリにあります。

- `filesystem` - ファイルシステム
- `app` - 汎用またはユーザ定義のアプリケーションであり、他のリソースに依存することがある

各リソースタイプのディレクトリには、以下のものが1つ以上あります。

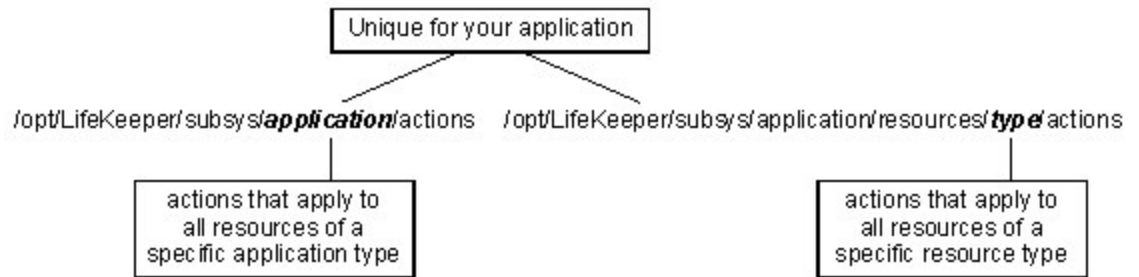
- **インスタンス。**このファイルは、LCD に保存されている、リソースインスタンスに関する恒久的な情報を反映します。このリソースタイプに関連付けられたリソースインスタンスの記述的な情報があります。

**警告:** インスタンスファイル(または LCD ファイル)を直接変更しないでください。リソースインスタンスの作成や操作を行うには、LifeKeeper の GUI の機能、または `ins_create`、`ins_remove`、`ins_gettag`、`ins_setas`、`ins_setinfo`、`ins_setinit`、`ins_setstate`、および `ins_list` の LifeKeeper Single Server Protection の `LCDI_instances` コマンドのみを使用してください。これらのコマンドの詳細については、`LCDI_instances` のマニュアルページを参照してください。

- **recovery。**このオプションのディレクトリには、障害が検出されたリソースのローカルリカバリの試行に使用されるプログラムがあります。recovery ディレクトリには、`sendevent` に渡されるイベントクラスに対応するディレクトリがあります。ディレクトリの名前は、`sendevent` プログラムに渡されるクラスパラメータ (-C) と一致する必要があります。( [LifeKeeper Single Server Protection の警報とリカバリ](#)を参照 )。

各サブディレクトリに、アプリケーションは対応するイベントタイプを処理するリカバリプログラムを入れることができます。これらのプログラムの名前は、`sendevent` の `-E` パラメータで渡される文字列と一致する必要があります。このオプションのディレクトリは、複数のアプリケーションに使用されるように存在することはできません。

- **actions。**このディレクトリには、特定のリソースタイプのリソースインスタンスについてのみ動作するリカバリ実行プログラムのセットがあります。使用するアプリケーションについて、アプリケーション内のすべてのリソースに適用する動作がある場合は、その動作を、**resource type** ディレクトリではなく、アプリケーションディレクトリの **actions** サブディレクトリに入れてください。



リカバリ指示ソフトウェアが、リソースインスタンスの変更や復旧に使用されます。各リソースタイプの **actions** ディレクトリに、**remove** と **restore** の2つの動作が必要です。

## リソースの動作

リソースタイプの **actions** ディレクトリには、特定のアプリケーションの動作を記述するプログラム(多くの場合は shell スクリプト)があります。各リソースタイプについて、**restore** と **remove** の2つの動作が必要です。

**remove** と **restore** のプログラムは、正反対の機能を実行する必要があります。つまり、相互の動作を元に戻す必要があります。これらのスクリプトは、絶対に手動で実行しないでください。これらのスクリプトは、LifeKeeper Single Server Protection のリカバリ動作と制御のインターフェース (LRACI) の **perform\_action** shell プログラムのみが実行する必要があります (LRACI-perform\_action (1M) マニュアルページを参照)。

詳細については、[リカバリスクリプト](#)を参照してください。

## LifeKeeper Single Server Protection のフラグ

LifeKeeper Single Server Protection では、システムに設定されたフラグの一覧が[ステータスの詳細表示](#)の最後付近に表示されます。共通タイプは、プロセスのロックが動作を完了するまで他のプロセスを確実に待機させるために使用する LCD のロックフラグです。LCD のロックの標準フォーマットは以下のとおりです。

```
!action!processID!time!machine:id.
```

一般的な LCD のロックフラグの例を示します。

- **!action!02833!701236710!<servername>:filesys.**ファイルシステム階層を作成すると、このフォーマットでステータス表示にフラグが生成されます。*filesys* の指定は、他のアプリケーションリソース階層では別のリソースタイプである場合も、一般的なアプリケーションやユーザー定義アプリケーションでは *app* である場合もあります。

## LCDI のコマンド

LifeKeeper Single Server Protection には、アプリケーションのリソース階層を定義するためのメカニズムが2つ用意されています。

## 階層の定義

- LifeKeeper GUI
- LifeKeeper 設定データベースのインターフェース (LCDI) コマンド

LCDI は LifeKeeper Single Server Protection が提供するインターフェースコマンドのセットで、使用するアプリケーションのニーズに合わせてカスタムリソース階層構成を作成できます。構成にはコマンドラインインターフェースを使用できます。

コマンドの詳細については、LCDI のマニュアルページを参照してください。ここでは、GUI とコマンドラインの両方を使用して、リソース階層を作成する構成シナリオについて説明します。

## 階層の定義

アプリケーション階層を作成するために必要な作業を示します。

1. **ファイルシステムリソースを作成する。** LifeKeeper の GUI には、ファイルシステムリソースを作成するメニューがあります。[ファイルシステムリソース階層の作成](#)を参照してください。

この定義作業の最後で、LCD では2つのファイルシステムリソースが以下のように定義されます。リソースのステータスを表示するには、`lcdstatus -q`を実行してください。

TAG	ID	STATE
my-fs-1	/mnt/fs1	ISP
my-fs-2	/mnt/fs2	ISP

**注記:** LifeKeeper Single Server Protection で使用されるタグ名には意味はありません。単なるラベルです。表内のタグ名は LifeKeeper Single Server Protection のデフォルト値です。

2. **アプリケーションリソースを定義する。** Generic Application (gen/app) リソース `my-app` を作成してください。[Generic Application リソース階層の作成](#)を参照してください。
3. **依存関係を定義する。** ファイルシステムリソースに依存するアプリケーションを作成するには、以下のように `dep_create` コマンドを使用してリソースの依存関係を作成してください。

```
dep_create -p my-app -c my-fs-1
```

```
dep_create -p my-app -c my-fs-2
```

4. **リソースを In Service にする。** LifeKeeper GUI にアクセスし、アプリケーションリソースを右クリックして **[In-Service]** を選択し、リソースを in service にします。

## LCM

LifeKeeper 通信マネージャ (LCM) は、LifeKeeper Single Server Protection サーバに高信頼性のプロセス間通信を提供します。このプロセスは、冗長コミュニケーションパスを使用できるので、1つのコミュニケーションパスに障害が発生しても、LifeKeeper Single Server Protection やそれが保護するリソースには障害が発生しません。LCM は、RS-232 (TTY) と TCP/IP の接続を含む多様な通信方法をサポートしています。

LCM は以下の機能を提供します。

- **LifeKeeper Single Server Protection のハートビート:** VMware HA と定期的に通信して、システムが動作を継続しているかどうかを判断します。LifeKeeper Single Server Protection が復旧不可能な障害を検出すると、ハートビートを抑制し、VMware HA にサーバを再起動するように通知します。
- **管理サービス:** LifeKeeper Single Server Protection の管理機能は、LCM の機能を使用してリモート管理を実行します。LCM を使用すると、一元管理、構成の検証、および管理アクションの健全性検査を実行できます。
- **設定とステータスの通信:** LifeKeeper 構成データベース (LCD) は、LCM を介して、リソースのステータス、可用性、および構成を追跡します。LCM の機能により、LCD はプライマリとセカンダリのシステム間で整合性のあるリソース情報を保持できます。

LCM が提供する LifeKeeper Single Server Protection のサービスに加えて、shell コマンドセットによりアプリケーションによる信頼性の高いシステム間通信が可能です。これらのコマンドとして、snd\_msg, rcv\_msg, can\_talk などがあります。これらのコマンドの詳細については、LCMI\_mailboxes (1M) のマニュアルページを参照してください。LCM はシステム上でリアルタイムプロセスとして動作し、システムのハートビートが送信されるなどの重要な通信が確実に実行されるようにします。

## LifeKeeper Single Server Protection の警報とリカバリ

LifeKeeper Single Server Protection のエラー検出と通知は、イベント警報メカニズム sendevent をベースにしています。**sendevent** メカニズムの重要な概念は、独立したアプリケーションが重要なコンポーネントについて警報を受信できるように登録できることです。警報を開始する側のコンポーネントと受信する側のアプリケーションのいずれも、他のアプリケーションの存在を知るように変更する必要はありません。アプリケーションに固有のエラーが、**sendevent** 機能経由で LifeKeeper Single Server Protection のリカバリメカニズムをトリガできます。

このセクションでは、警報クラス、警報の処理、および警報ディレクトリのレイアウトを含む警報に関するトピックを説明し、次に警報の概念を示す処理シナリオを示します。

### 警報クラス

`/opt/LifeKeeper/events` ディレクトリには、アラームクラスのセットがリストされます。これらのクラスは、イベントを生成するシステムの特定制サブコンポーネントに対応します (例: `filesystem`)。各警報クラスのサブディレクトリには、可能性のある警報のセットがあります (例: `badmount`, `diskfull`)。shell スクリプトまたはプログラムを適切なディレクトリに入れることで、これらの警報を受信するようにアプリケーションを登録できます。

LifeKeeper Single Server Protection は基本的な警報通知機能を使用しています。この警報機能により、イベントについて登録されたすべてのアプリケーションで、該当する警報の発生時に sendevent により処理プログラムが非同期で実行されます。LifeKeeper Single Server Protection が存在する場合、**sendevent** プロセスははじめに、LifeKeeper Single Server Protection のリソースオブジェクトがクラスとイベントを処理できるかどうかを判断します。LifeKeeper Single Server Protection がクラス / イベントの一致を検出した場合、適切なリカバリシナリオが実行されます。

**sendevent** 警報機能の追加スクリプトを定義することは任意です。LifeKeeper Single Server Protection リソースを定義すると、LifeKeeper Single Server Protection が基本的な警報機能を提供します。その詳細は、この章の処理シナリオで後述します。

**注記:** リソースインスタンスのローカルリカバリは、LifeKeeper Single Server Protection の制御下にあるアプリケーションが、中断されたリソースサービスを、イベントが発生したシステムのエンドユーザーに返そうとする試行です。

## 警報の処理

LifeKeeper Single Server Protection の注意が必要な可能性のあるイベントを検出するアプリケーションまたはプロセスは、**sendevent** プログラムを実行し、各エラークラス、エラー名、および障害のあるインスタンスの引数を渡すことにより、イベントを報告できます。必須の詳細、オプションのパラメータ、および構文については、**sendevent** のマニュアルページを参照してください。

## 警報ディレクトリのレイアウト

`/opt/LifeKeeper/events` ディレクトリには、2種類の内容があります。

- **LifeKeeper Single Server Protection が提供するクラス:** LifeKeeper Single Server Protection は、`events` ディレクトリの下に `lifekeeper` と `filesys` の2つの警報クラスを用意しています。警報イベントの例として、`noaccess` と `diskfull` があります。警報クラスは、**sendevent** コマンドの `-C` オプションで渡される文字列に対応し、警報イベントは `-E` オプションで渡される文字列に対応します。LifeKeeper Single Server Protection の警報クラスは、LifeKeeper Single Server Protection によって、LifeKeeper Single Server Protection のサブシステム内のイベント報告用に内部的に使用されます。
- **アプリケーションに固有のクラス:** 特定のアプリケーションで警報クラスの定義が必要な場合、`events` ディレクトリに他のサブディレクトリが追加されます。アプリケーションは shell スクリプトまたはバイナリプログラムをそのサブディレクトリに入れることで、これらの警報を受信するように登録します。これらのプログラムの名前は、属するアプリケーションパッケージの名前に由来します。

## リカバリスクリプト

LifeKeeper Single Server Protection に対するアプリケーションのインターフェースの定義は、LifeKeeper の GUI、コマンドインターフェース、またはそれらの組み合わせを使用して実行できます ([LCDI のコマンドインターフェース](#) のトピックの例を参照)。いずれの方法を使用する場合でも、少なくともリソースの開始と停止を行うリカバリスクリプトを指定する必要があります。

アプリケーションのインターフェースに必要なスクリプトを定義するための参考情報については、[スクリプトのタイプとスクリプトのパラメータ](#) を参照してください。これらのトピックでは、[remove](#)、[restore](#)、[delete](#)、および [ローカルリカバリ](#) の4種類のスクリプトについて詳しく説明しています。remove と restore のスクリプトの説明には、スクリプトの例があります。

## アプリケーションインターフェースのレベル

LifeKeeper Single Server Protection には、イベント検出とリカバリ制御の基礎、およびアプリケーションの可用性を確保するための環境とツールセットが用意されています。このトピック、および [共通アプリケーションタイプのインターフェースの問題](#) と [インターフェースの定義作業](#) では、以下の内容について説明しています。

- 可用性を確保するために検討が必要なアプリケーションインターフェースのレベル
- 共通アプリケーションタイプのインターフェースを提供するために必要な動作
- アプリケーションとLifeKeeper Single Server Protectionとの間にインターフェースを作成する手順。

## インターフェースのレベル

アプリケーションの高い可用性を確保するための作業は、アプリケーションの適切な選択と設定から始まります。例えば、データベースのトランザクションを実行するアプリケーションについて高い可用性が必要な場合は、ログ記録、アーカイブ作成、ロールフォワード/ロールバックなどの機能、および制御可能な内部リカバリ方式を持つアプリケーションを選択するか、開発する必要があります。

次に、LifeKeeper Single Server Protection がサポートする以下の3レベルのアプリケーションのインターフェースを実装する方法を決定する必要があります。

- 依存関係の定義
- エラーの検出と処理
- リカバリ動作

## 依存関係の定義

可用性を確保するためには、アプリケーションの他のシステムリソースに対する依存関係を特定し、それらのリソースで発生したエラーの検出方法と処理方法を決定する必要があります。LifeKeeper Single Server Protection では、LifeKeeper の GUI またはコマンドラインインターフェースを使用して、依存関係 (階層) を定義できます。

## エラーの検出と処理

アプリケーション内に問題の検出と警告の機能を装備することは、最良の総合的な障害への耐性ソリューションを作成するために重要です。アプリケーションごとに障害のメカニズムとフォーマットが異なるので、汎用メカニズムの1セットで提供することはできません。ただし、一般的に、多くのアプリケーションの設定は、LifeKeeper Single Server Protection が装備しているコアシステムのエラー検出を使用できます。[障害検出とリカバリのシナリオ](#)トピックでは、一般的な3つの障害状況を説明し、LifeKeeper Single Server Protection の Core 機能を示します。

LifeKeeper Single Server Protection は、リカバリ手順をトリガできるエラー、警告、およびイベントを定義できる総合環境も提供します。このインターフェース作成には通常、システムエラーログ (`/var/log/messages`) のパターンマッチ定義、またはアプリケーションに固有のカスタム監視プロセスが必要です。

## リカバリ動作

障害への耐性は、予測されるさまざまな障害条件に対するリカバリ動作の定義によって異なります。LifeKeeper Single Server Protection は、shell スクリプトを使用して、保護するリソースに対するリカバリ動作を指定します。LifeKeeper Single Server Protection のアプリケーションリソースに必要な2つのリカバリスクリプトは [restore](#) と [remove](#) であり、LifeKeeper Single Server Protection がアプリケーションを

in service にするときには restore スクリプト、out-of-service にするときには remove スクリプトが常に使用されます。

[リカバリスクリプト](#) のトピックには、LifeKeeper Single Server Protection が認識する多様な種類のスクリプトのリストがあり、カスタムスクリプトの作成時に役立つ説明と例があります。

## 共通アプリケーションタイプのインターフェースの問題

アプリケーションと LifeKeeper Single Server Protection との間にインターフェースを提供するために実行する作業は、アプリケーションの種類と複雑さによって異なります。以下に例を示します。

**ファイルシステムサービス:** LifeKeeper の GUI では、メニューベースでファイルシステム階層を設定できます。

**DBMS アプリケーション:** LifeKeeper Single Server Protection 製品ファミリには、MySQL、Oracle、DB2、Informix などの RDBMS アプリケーション用のオプションのリカバリキットパッケージがあります。これらのリカバリパッケージは、データベースインスタンスを LifeKeeper Single Server Protection で保護するように設定するためのメニューオプションを LifeKeeper の GUI に表示します。また、これらのパッケージは、関連するデータベースの初期化コマンドを使用するデフォルトの [restore](#) と [remove](#) のスクリプトも提供します。

**単純なアプリケーション:** 1つのファイルシステムやディスクデバイスに依存する設定の場合、LifeKeeper の GUI に階層の作成機能のメニューがあります。それらのアプリケーション自体の [remove](#) と [restore](#) のスクリプトを用意するだけですみます。

**複雑なアプリケーション:** 対応する LifeKeeper Single Server Protection のアプリケーションリカバリキットが存在しない複雑な設定の場合、LifeKeeper 設定データベースインターフェース (LCDI) コマンドを使用できます。例えば、コマンドを使用して、カスタムのリレーショナルデータベースや、複数のファイルシステムやディスクパーティションに依存するアプリケーションのインターフェースを用意できます。また、必須の [restore](#) と [remove](#) のスクリプトも用意する必要があります。詳細については、[LCDI のコマンドインターフェース](#) を参照してください。

## インターフェースの定義作業

アプリケーションと LifeKeeper Single Server Protection との間にインターフェースを作成する作業を開始する前に、適切なハードウェアとソフトウェア (アプリケーションと LifeKeeper Single Server Protection を含む) がインストールされており、正しく動作していることを確認する必要があります。

コンポーネントが正しく設定され、動作していることを確認した後、以下の手順に従って LifeKeeper Single Server Protection とのインターフェースを作成する必要があります。

1. **アプリケーションスクリプト (またはプログラム) を作成する。** LifeKeeper Single Server Protection は、アプリケーションの開始と停止にスクリプト (またはプログラム) のセットを使用します。LifeKeeper Single Server Protection の管理下にあるアプリケーションに通常必要な 2 つのスクリプトは、[restore](#) と [remove](#) です。また、アプリケーションまたはシステムが特定の動作条件にどのように反応するかを指定するその他の動作スクリプトも指定できます。
2. **LifeKeeper Single Server Protection リソース階層を定義する。** システム障害や管理操作のイベントで LifeKeeper Single Server Protection がアプリケーションを制御するためには、リソース階層を定義する必要があります。この定義とは、アプリケーションのリソースインスタンスと関係を作成することです。

LifeKeeper Single Server Protection リソース階層を作成するには、以下の3つの方法があります。

1. **LifeKeeper の GUI を使用する。**
2. **LifeKeeper 設定 データベースインターフェース (LCDI) のコマンドを使用する** (複数のディスクパーティションやファイルシステムに依存するアプリケーション、その他の任意のアプリケーションタイプの場合)。
3. **GUI とコマンド関数の組み合わせを使用する。** [LCDI コマンドインターフェース](#) のトピックには、GUI と LCDI コマンドの両方を使用する作成例があります。
4. **オプションのアプリケーションの障害検出を設定する。** アプリケーション内に問題の検出と警告の機能を装備できることは、最良の総合的な耐障害ソリューションを作成するために重要です。アプリケーションごとに障害のメカニズムとフォーマットが異なるので、汎用メカニズムの1セットがすべてのニーズを満たすことはありません。ただし、LifeKeeper Single Server Protection は、リカバリ手順をトリガできるエラー、警告、およびイベントを定義できる総合環境を提供します。詳細については、[LifeKeeper Single Server Protection の警報とリカバリ](#) のトピックを参照してください。

## スクリプトの種類

LifeKeeper Single Server Protection は、[remove](#)、[restore](#)、および [delete](#) の3種類の主要なリカバリスクリプトを使用します。これらのスクリプトの作成後、適切なディレクトリ `/opt/LifeKeeper/subsys/application/resources/type/actions` に保存すると、障害が発生した場合に LifeKeeper Single Server Protection はそれらのスクリプトを使用します。

LifeKeeper の GUI を使用してアプリケーションリソース階層を定義した場合、LifeKeeper Single Server Protection が適切なディレクトリに自動的にスクリプトを入れます。

各リソースタイプに固有の `restore`、`remove`、および `delete` のスクリプトに加えて、LifeKeeper Single Server Protection は、`restore`、`remove`、または `delete` の前後に実行されるグローバルな動作を指定するリカバリスクリプトとも相互作用します。

- **preremove** - remove スクリプトの実行前に必要な動作。preremove スクリプトは、`/opt/LifeKeeper/subsys/application/actions` ディレクトリに保存してください。
- **postremove** - remove スクリプトの実行後に必要な動作。postremove スクリプトは、`/opt/LifeKeeper/subsys/application/actions` ディレクトリに保存してください。
- **prerestore** - restore スクリプトの実行前に必要な動作。prerestore スクリプトは、`/opt/LifeKeeper/subsys/application/actions` ディレクトリに保存してください。
- **postrestore** - restore スクリプトの実行後に必要な動作。postrestore スクリプトは、`/opt/LifeKeeper/subsys/application/actions` ディレクトリに保存してください。
- **predelete** - delete スクリプトの実行前に必要な動作。predelete スクリプトは、`/opt/LifeKeeper/subsys/application/actions` ディレクトリに保存してください。
- **postdelete** - delete スクリプトの実行後に必要な動作。postdelete スクリプトは、`/opt/LifeKeeper/subsys/application/actions` ディレクトリに保存してください。

これらのスクリプトは、対象とするアプリケーションやリソースのタイプには関係なく実行されることに注意してください。例えば、サービス停止の動作を実行して通信アプリケーションがサービス停止になった後に

実行する postremove スクリプトを指定した場合、このスクリプトは通信の remove スクリプトの実行後にも動作します。

## スクリプトのパラメータ

remove、restore、および delete のスクリプトを作成するときには、スクリプトの実行時に LifeKeeper Single Server Protection が以下のタイプのパラメータを渡すと仮定してください。

- **-t タグ** - 動作を実行する対象のリソースのタグ名。
- **-i id** - 動作を実行する対象のリソースの ID 名。
- **[-R]** - エラーログ記録機能が使用するように渡されるオプションのフラグ。

## restore スクリプト

LifeKeeper Single Server Protection は、特定のリソースタイプについて、そのインスタンスを in service にする必要がある場合は常に、そのリソースタイプの restore スクリプトを実行します。ファイルシステムタイプのリソースの restore スクリプトは、ディレクトリ `/opt/LifeKeeper/subsys/gen/resources/filesys/actions` にあります。

多くの場合、アプリケーションの動作スクリプトには標準コンポーネントがあります。例えば、ファイルシステムの restore スクリプトの冒頭の 3 つのコンポーネントは、remove スクリプトと同じです。スクリプトのこれらの部分をコピーして、アプリケーション用のスクリプトをカスタマイズできます。

restore スクリプトを実行する LRACI プログラムは、正常に完了した場合 (終了コード 0) にリソースインスタンスを ISP 状態にし、プログラムが失敗した場合は OSF 状態にします。また、restore スクリプトは、すでに動作可能なアプリケーションやリソースの「サービス起動」を正しく処理する必要があります。

ファイルシステムの restore スクリプトの例に、機能的な処理のセクションを示します。スクリプトコードで、以下の重要な項目に注意してください。

- 行番号 101 ~ 108 は、ファイルシステムがすでにマウントされているかどうかをチェックします。マウントされている場合、目的の結果が達成されているのでエラーとしては扱われません。この状況は、`lstop -f` の後に LifeKeeper Single Server Protection を再起動した場合に発生することがあります。restore スクリプトの冒頭に類似のチェックを含めると、後のソフトウェアアップグレードが促進されます。
- 行番号 110 は、インスタンス情報を取得します。
- 行番号 112 ~ 126 は、下のデバイス名とファイルシステムの情報を取得します。
- 行番号 128 ~ 141、181 ~ 188 は、ファイルシステムが root であるという不適切な状況进行处理します。優れた shell スクリプトの特徴は、発生する可能性があるエラー状況を予測し、処理できることです。
- 行番号 143 ~ 151 は、必要な場合にマウントポイントを作成します。
- 行番号 153 ~ 158 は、ファイルシステムのマウントを試行し、マウントに成功した場合に終了します。
- 行番号 160 ~ 179 は、ファイルシステムをクリーンアップします。これは通常、システムの障害リカ

バリで実行されます。その理由は、ファイルシステムが正しくアンマウントされなかったからです。

- 行番号 189 ~ 195 は、ファイルシステムのマウントを再度試行し、失敗した場合は他のオプションを試行します。
- 行番号 196 ~ 214 は、マウントポイントの移動を試行します。
- 行番号 215 は、ファイルシステムの最後のマウントを試行します。
- 行番号 149、207、213、223、229 は、失敗により終了します。LifeKeeper Single Server Protection がファイルシステムインスタンスを OSF としてマークし、この階層を in service にするプロセスを中止します。

## restore スクリプトの例

この例は、ファイルシステムの restore スクリプトの機能的な処理のセクションです。

```
70 getchildinfo() {
71 OKAPP=$1
72 PTAG=$2
73 ret=1
74 if $LKROOT/bin/dep_list -c $PTAG | sed "s/_/g; s/_/_g" >/tmp/DL$$
75 then
76 for i in `cat /tmp/DL$$`
77 do
78 I=`echo "$i" | sed "s/_/g; s/_/_g"`
79 CHDTAG=`echo "$I" | cut -d_ -f2`
80 if CHDDATA=`$LKROOT/bin/ins_list -t$CHDTAG`
81 then
82 APP=`echo "$CHDDATA" | cut -d_ -f2`
83 if [ "$OKAPP" != "$APP" ]
84 then
85 continue
86 fi
87 ret=0
88 break
89 fi
90 done
```

## restore スクリプトの例

```
91 fi
92 if [ "$ret" != 0 ]
93 then
94 pl "LifeKeeper:*ERROR* getchildinfo found no $OKAPP child for $PTAG"
95 fi
96 return $ret
97 }
98
99 pl "LifeKeeper: mounting file system $FSNAME"
100
101 # Check if the filesystemfile system exists
102 f=`sed -n "\?^.*$FSNAME ?p" /etc/mnttab`
103 if [ "$f" != "" ]
104 then
105 pl "LifeKeeper: file system $FSNAME already mounted"
106 err=0
107 exit 0
108 fi
109
110 if v=`$LKROOT/bin/ins_list -t"$TAG"`
111 then
112 if getchildinfo scsi $TAG
113 then
114 CTAG=$CHDTAG
115 else
116 exit 1
117 fi
118
119 # Get the mount information
120 INFO=`echo "$v" | cut -d_ -f6`
```

```
121 FSTYPE=`echo "$INFO" | cut -d_ -f1`
122 FSPERM=`echo "$INFO" | cut -d_ -f2`
123
124 fp=`echo "$CHDDATA" | cut -d_ -f5`
125 FPName=/dev/dsk/$fp
126 FPRawName=/dev/rdisk/$fp
127
128 # Test for root filesystemfile system
129 test $FSNAME = /
130 if [ "$?"-eq 0 ]
131 then
132 ROOT=1
133 else
134 ROOT=0
135 fi
136
137 # If root filesystemfile system: return code 0 means it is OK and mounted
138 if [ "$ROOT" -eq 1 ]
139 then
140 err=0
141 exit 0
142 else # If other than root filesystemfile system, mount it
143 if [ !-d $FSNAME ]
144 then
145 mkdir $FSNAME
146 if [ $?!= 0 ]
147 then
148 pl "LifeKeeper: can't make file system $FSNAME mount point"
149 exit 1
150 fi
151 fi
```

## remove スクリプト

LifeKeeper Single Server Protection は、特定のリソースタイプについて、そのインスタンスを out of service にする必要がある場合は常に、そのリソースタイプの remove スクリプトを実行します。このセクションでは、ファイルシステムタイプのリソースの remove スクリプトについて説明します。このスクリプトは、LifeKeeper Single Server Protection システムのディレクトリ `/opt/LifeKeeper/subsys/gen/resources/filesys/actions` にあります。

remove スクリプトを実行する LRACI プログラムは、正常に完了した場合 (終了コード 0) にリソースインスタンスを OSU 状態にし、プログラムが失敗した場合は状態を変更しません。

[remove と restore スクリプトに共通のセクション](#) のトピックは、remove と [restore](#) のスクリプトの両方で同一である冒頭の 3 つのセクションについて説明しています。スクリプト開発の開始地点として、これらのセクションをコピーすることが推奨されます。[remove スクリプトの例](#) のトピックには、ファイルシステム remove スクリプトの機能的な処理セクションがあり、**remove** スクリプトの残りの部分を示します。

リカバリスクリプトには、以下の 5 つの基本部分があります。

- **初期化**: 特定の必須の環境変数、特に **PATH** と **LKROOT** を設定するために、LifeKeeper Single Server Protection のすべてのスクリプトは `/etc/default/LifeKeeper` をソースにします。
- **パラメータ解析の呼び出し**: LifeKeeper Single Server Protection は、少なくとも以下に示す 2 つのオプションを使用して、各リカバリスクリプトを呼び出します。
  - `-t instance_tagname`
  - `-i instance_id`

スクリプトは、これらのパラメータの両方または一方を使用して、remove するインスタンスを指定できます。この例では、ファイルシステムの **remove** スクリプトは、`instance_id` (ファイルシステムのマウントポイント) のみを使用します。インスタンスがユーザ作成のリソースタイプである場合は、`instance_id` のフォーマットと意味を指定します。**-R** パラメータはオプションですが、`prfuncs` が変数 **RCVARG** を使用するので、リカバリスクリプトの行番号 37 ~ 39 を常に含める必要があります。

- **エラーのログ記録**: 行番号 56 は、`prfuncs` ファイルの内容をスクリプトに読み込みます。`prfuncs` ファイルには、スクリプト全体で必要な多数のエラーのログ記録ユーティリティがあります。`prfuncs` のマニュアルページに、関数 `log`、`pl`、および `pt` の説明があります。
- **終了処理**: スクリプトの終了時に、それが正常終了かどうかには無関係に、スクリプトがそれ自体をクリーンアップできるようにシグナルトラップを含めるのが、優れた shell プログラミング方法です。**remove** の例では、トラップ機能が一時ファイルを削除し、完了メッセージを LifeKeeper Single Server Protection のログに送信します。
- **機能的な処理**: [remove スクリプトの例](#) のトピックに示す remove スクリプトの部分は、実際にファイルシステムインスタンスをサービス停止にします。スクリプトは、以下の動作を指定します。
  - **ファイルシステムがマウントされているかどうかをチェックする**。スクリプトははじめに、いくつかのチェックを実行して、指定したファイルシステムがマウントされているかどうかを確認します。ファイルシステムがマウントされていない場合、プロセスはアンマウントする必要がなく、目的の結果が達成されているので正常に終了します。

- ファイルシステムを使用しているプロセスを強制終了する。ファイルシステムをアンマウントする前に、スクリプトがファイルシステムを使用しているプロセスを強制終了して、アンマウントできるようにします。
- ファイルシステムをアンマウントする。プロセスがすぐにファイルシステムをアンマウントできない場合、ユーザプロセスが終了するための十分な時間が確保されるように、スクリプトが待機します。このスクリプトは、正常なアンマウントの後でのみ、**err** を **0** に設定します。

## remove スクリプトの例

この例は、ファイルシステムの remove スクリプトの機能的な処理のセクションです。

```

72 if f=`sed -n "\?^.*$FSNAME ?p" /etc/mnttab`
73 then
74 if [ "$f" != "" ]
75 then
76 # Check if the file system is being used by a process;
77 # if so, kill the process
78 pl "LifeKeeper: must kill off any processes accessing file system $FSNAME before it can be
unmounted."
79
80 # A timing window exists here -- additional users could
81 # access the FS after fuser issues the kill. In this
82 # case the umount would fail. So, we try 3 times or
83 # until fuser reports no processes
84 for try in 1 2 3
85 do
86 if [ `fuser -k -c "${FSNAME}" 2>/dev/null | wc -c` -eq 0 ]
87 then
88 # No more processes...(we have to be quick now!)Unmount
89 # the file system. The umount is done immediately -- this
90 # might mess up the log files if an error occurs, but if
91 # we issue the messages first, it just lengthens the timing
92 # window.
93 if umount "${FSNAME}" 2>/tmp/UM$$
94 then

```

remove スクリプトの例

```
95 # umount worked... issue the messages now (better late
96 # than not at all!)
97 pl "LifeKeeper: unmounting file system $FSNAME"
98 pl "\tumont ${FSNAME}"
99 pl "LifeKeeper: file system $FSNAME successfully unmounted"
100 err=0
101 exit 0
102 else
103 pl "LifeKeeper: unmounting file system $FSNAME"
104 pl "\tumont ${FSNAME}"
105 cat /tmp/UM$$ >&2
106 pl "LifeKeeper:*ERROR* file system $FSNAME failed unmount; will try again"
107 fi
108 fi
109 sleep 3 # processes might be slow to die, so wait a bit
110 done
111 else
112 pl "LifeKeeper:File system ${FSNAME} is not mounted."
113 err=0
114 exit 0
115 fi
116 else
117 exit 1
118 fi
119
120 #
121 # If we get here, the fuser above failed to kill all the active processes.
122 # The unmount of the file system will most likely fail, but what can we
123 # do?
124
```

```
125 sleep 5
126
127 pl "LifeKeeper: unmounting file system $FSNAME"
128 pl "\tumont ${FSNAME}"
129 if umount "${FSNAME}"
130 then
131 pl "LifeKeeper: file system $FSNAME successfully unmounted"
132 err=0
133 exit 0
134 else
135 pl "LifeKeeper:*ERROR* file system $FSNAME failed unmount"
136 exit 1
137 fi
```

## remove と restore スクリプトに共通のセクション

```
1 #!/usr/bin/ksh
2 #ident "@(#)remove 1.1.1.2"
3 # Copyright 2000 SIOS Technology Corp., Mountain View, CA
4 #
5 # usage: remove -t tagname -i fsname (full path)
6 #
7
8 DEFAULT_FILE=/etc/default/LifeKeeper
9 if [ -z "$LKROOT" ]
10 then
11 PATH=
12 .$DEFAULT_FILE
13 LKROOT=${LKROOT:=/opt/LifeKeeper}
14 PATH=${PATH:=/opt/LifeKeeper/bin:/usr/bin:/usr/sbin:/bin:/sbin}
15 export LKROOT PATH
16 fi
```

## remove と restore スクリプトに共通のセクション

```
17
18 TAG=
19 FSNAME=
20 RCVARG=
21
22 while [ $# != 0 ]
23 do
24 case "$1" in
25 -t*)
26 if TAG=`$LKROOT/subsys/actions/testflag -t "$1"$2`
27 then
28 shift
29 fi
30 ;;
31 -i*)
32 if FSNAME=`$LKROOT/subsys/actions/testflag -i "$1"$2`
33 then
34 shift
35 fi
36 ;;
37 -R)
38 RCVARG=-R
39 ;;
40 esac
41 shift
42 done
43
44 if [ "$TAG" = "" ]
45 then
46 echo "$0: -t flag not specified"
```

```

47 exit 1
48 fi
49
50 if [ "$FSNAME" = "" ]
51 then
52 echo "$0: -i flag not specified"
53 exit 1
54 fi
55
56 . $LKROOT/subsys/actions/prfuncs
57
58 log "LifeKeeper: REMOVE FILE SYSTEM $FSNAME STARTAT:\n\t`date`"
59
60 err=1
61
62 trapfunc() {
63 rm -f /tmp/??$$
64 log "LifeKeeper: REMOVE FILE SYSTEM $FSNAME END err=$errAT:\n\t`date`"
65 exit $err
66 }
67
68 trap "trapfunc" 0 1 2 3 4 6 7 8 10 12 13 15 19

```

## delete スクリプト

delete スクリプトの作成が必要になるのは、インスタンスを正常に削除するために完了する必要がある [remove](#) スクリプトにとって不適切な作業がいくつかある場合のみです。例えば、**ファイルシステムのリソースタイプ**には、**削除**スクリプトが含まれています。これは、**ファイルシステム**インスタンスの作成プロセスが、両方のシステムの `/etc/fstab` を変更したからです。このインスタンスを削除した場合、**fstab** ファイルを元の状態にリストアする必要があります。

LifeKeeper の GUI の機能を使用してリソースを作成した場合は、LifeKeeper Single Server Protection が自動的に削除機能を正しく管理します。LifeKeeper Single Server Protection がインスタンスを削除するときに同様な種類の動作を元に戻す必要があり、かつその動作を LifeKeeper Single Server Protection で自動実行させる場合は、**delete** スクリプトのメカニズムを使用できます。

ただし、LifeKeeper Single Server Protection の delete プロトコルは複雑なので、**delete** スクリプトを作成するときには注意してください。例として、ファイルシステムの delete スクリプトを使用できます。これは、ディレクトリ `/opt/LifeKeeper/subsys/gen/resource/filesys/actions` にあります。

**注記:** delete スクリプトは、絶対に手動で実行しないでください。ins\_remove プログラムの一部としてのみ実行してください。

## 通知スクリプトの例

このスクリプト例は、ディスクフルの条件が検出され、その状況をシステム管理者に報告する必要がある場合に呼び出されます。この条件を解消するには、管理操作が必要です。このスクリプトは、管理者に電子メールを送信する例です。これらの通知機能を有効にするには、`/opt/LifeKeeper/events/filesys/diskfull/notify` にあるこのスクリプトを編集してください。

## ローカルリカバリスクリプト

LifeKeeper Single Server Protection のサーバ間リカバリ機能よりも優先されるローカルリカバリ動作を指定するスクリプトを作成することもできます。ローカルリカバリスクリプトの例が、ディレクトリ `/opt/LifeKeeper/subsys/gen/resources/filesys/recovery/filesys` にあります。

## メンテナンス作業

このセクションには、LifeKeeper Single Server Protection のメンテナンスに関するトピックがあります。

## リソース階層のメンテナンス

システム上のその他すべての階層を LifeKeeper Single Server Protection で保護した状態で、あるリソース階層のメンテナンスを実行できます。このためには、メンテナンスが必要な階層を out of service にし、メンテナンス作業の完了後にその階層を in service にします。

リソース階層のメンテナンスを実行するには、以下の手順に従ってください。

1. **階層を out of service にします。** LifeKeeper の GUI を使用して、メンテナンスを実行する必要があるリソース階層をすべて out of service にします。詳細については、[リソースを Out of Service にする](#)を参照してください。
2. **メンテナンスを実行します。** リソース階層で必要なメンテナンスを実行します。
3. **階層をリストアします。** LifeKeeper の GUI を使用して、リソース階層を in service にします。詳細については、[リソースを In Service にする](#)を参照してください。

## LifeKeeper Single Server Protection の設定値の変更

LifeKeeper Single Server Protection の設定とセットアップを行った後に、LifeKeeper Single Server Protection の多数の値を変更しなければならない場合があります。変更が必要な値の例として、LifeKeeper Single Server Protection サーバの uname、IP リソースのアドレス、タグ名などがあります。これらの値を変更するには、注意して以下の手順に従ってください。

1. 以下のコマンドを使用して、LifeKeeper Single Server Protection を停止してください。

```
LKROOT/bin/lkstop
```

2. LifeKeeper Single Server Protection サーバの `uname` を変更する場合は、`hostname(1)` コマンドを使用してサーバのホスト名を変更してください。
3. LifeKeeper Single Server Protection の複数の値を変更する場合は、古い値と新しい値を以下のフォーマットで指定する必要があります。

```
old_value1=new_value1
```

```
....
```

```
old_value9=new_value9
```

4. `lk_chg_value` コマンドを実行し、出力を確認して、変更内容により予測しなかった副作用が発生していないことを確認してください。変更する値が複数ある場合は、以下のコマンドを実行してください。

```
$LKROOT/bin/lk_chg_value -Mvf file_name
```

`file_name` は、手順 4 で作成したファイルの名前です。

変更する値が1つのみの場合は、以下のコマンドを実行してください。

```
$LKROOT/bin/lk_chg_value -Mvo old_value -n new_value
```

`-M` オプションは、LifeKeeper Single Server Protection のすべてのファイルに対して変更を行わないことを指定します。

5. クラスタ内のすべてのサーバで、`-M` オプションを指定せずに `lk_chg_value` コマンドを実行して、LifeKeeper Single Server Protection のファイルを変更してください。変更する値が複数ある場合は、以下のコマンドを実行してください。

```
$LKROOT/bin/lk_chg_value -vf file_name
```

`file_name` は、手順 4 で作成したファイルの名前です。

変更する値が1つのみの場合は、以下のコマンドを実行してください。

```
$LKROOT/bin/lk_chg_value -vo old_value -n new_value
```

6. 以下のコマンドを使用して、LifeKeeper Single Server Protection を再起動してください。

```
$LKROOT/bin/lkstart
```

**注記:** GUI を閉じて再起動する必要がある場合があります。

**注記:**

- LifeKeeper Single Server Protection のファイルを変更せずに `lk_chg_value` による変更内容を表示するには、`-M` オプションを使用してください。 `lk_chg_value` が調べるファイルを表示するには、`-v` を使用してください。タグ名を変更しない場合は、`-T` オプションを使用してください。リソース ID を変更しない場合は、`-I` オプションを使用してください。

## ファイアウォールを使用した状態での LifeKeeper Single Server Protection の実行

以下のネットワークアクセス要件を満たす場合、LifeKeeper Single Server Protection for Linux は、同一サーバ上にファイアウォールを設定した状態で実行できます。

**注記:** ファイアウォールを単に無効にする場合は、後述の[ファイアウォールの無効化](#)を参照してください。

### LifeKeeper GUI の接続

LifeKeeper GUI は、デフォルトの初期接続ポートであるポート 81 と 82 を含めて、特定の TCP ポートを多数使用します。また、LifeKeeper GUI は、ポート 1024 以降をオブジェクトの送受信に使用するリモートメソッド呼び出し (RMI) も使用します。これらすべてのポートが、各 LifeKeeper Single Server Protection サーバのファイアウォールで、少なくとも GUI クライアントが動作する外部システムに対して開いている必要があります。

### LifeKeeper Single Server Protection の IP アドレスリソース

IP アドレスに関連付けられたアプリケーションにアクセスする必要があるクライアントシステムから、LifeKeeper Single Server Protection の階層にある IP アドレスリソースにアクセスできるように、ファイアウォールを設定する必要があります。

また、LifeKeeper Single Server Protection は、ブロードキャスト ping のテストを使用して、IP アドレスリソースの健全性を定期的にチェックします。このテストでは、仮想 IP アドレスからブロードキャスト ping パケットを送信し、ローカルサブネット上の他のいずれかのシステムが最初に応答するまで待ちます。このテストが失敗しないようにするには、各 LifeKeeper Single Server Protection サーバ上のファイアウォールが以下のタイプのネットワーク動作を許可するように設定する必要があります。

- 仮想 IP アドレスからのインターネット制御メッセージプロトコル (ICMP) パケットの送信 (アクティブな LifeKeeper Single Server Protection サーバがブロードキャスト ping を送信できる)
- 仮想 IP アドレスからの ICMP パケットの受信 (他の LifeKeeper Single Server Protection サーバがブロードキャスト ping を受信できる)
- 任意のローカルアドレスからの ICMP 応答パケットの送信 (他の LifeKeeper Single Server Protection サーバがブロードキャスト ping に応答できる)
- 仮想 IP アドレスでの ICMP 応答パケットの受信 (アクティブな LifeKeeper Single Server Protection サーバがブロードキャスト ping への応答を受信できる)

### ファイアウォールの無効化

ファイアウォールを無効にする場合は、以下の手順に従ってください。

1. 以下のコマンド (お使いのファイアウォールパッケージによって異なる) を使用して、ファイアウォールを停止してください。

```
/etc/init.d/ipchains stop or
```

```
/etc/init.d/iptables stop
```

IPv6 環境を使用している場合は、かならず `ip6tables` を考慮してください。

```
/etc/init.d/ip6tables stop
```

SuSE Linux Enterprise Server を実行している場合

```
/etc/init.d/SuSEfirewall2_init stop
```

```
/etc/init.d/SuSEfirewall2_setup stop
```

2. パッケージを削除するか (`rpm -e` を使用)、以下のいずれかのコマンド (お使いのファイアウォールパッケージによって異なる) を使用して起動を無効にしてください。

```
/sbin/chkconfig --del ipchains or
```

```
/sbin/chkconfig --del iptables
```

```
/sbin/chkconfig --del ip6tables
```

SuSE Linux Enterprise Server を実行している場合は、`SuSEfirewall2` の設定を管理する必要があります。

## ファイアウォール経由での LifeKeeper GUI の実行

場合によっては、LifeKeeper クラスタが会社のファイアウォール内に配置され、管理者はファイアウォールの外側にあるリモートシステムから LifeKeeper GUI を実行します。

LifeKeeper は、GUI のサーバとクライアントとの通信にリモートメソッド呼び出し (RMI) を使用します。RMI クライアントは、それぞれの方向に通信を確立できる必要があります。RMI クライアントは動的ポートを使用するので、クライアントには推奨ポートを使用できません。

解決法としては、以下のように `ssh` を使用して、ファイアウォールを通過する方法があります。

1. 社内の IT 部門が、ファイアウォール内にアクセスするために十分にセキュリティの高い shell ポートを社内ファイアウォールに開けていることを確認します。多くの場合、IT 部門がアクセスを許可するマシンは、実際にはクラスタ内のマシンではなく、そこからクラスタ内にアクセスできる中間マシンです。このマシンは、Unix または Linux が動作するマシンである必要があります。
2. 中間マシンと LifeKeeper サーバの両方が、`sshd` (Secure Shell デモン) を実行していること、および X11 ポート転送が有効になっていること (これは通常、`etc/ssh/sshd_config` の `'X11Forwarding yes'` 行にある) を確認してください。不明の場合は、IT 部門に依頼してください。
3. X の Unix クライアントから以下のコマンドを使用して、中間マシンにトンネルを作成します。

```
ssh -X -C <intermediate machine>
```

`-C` は「トラフィックの圧縮」を意味し、低速のインターネットリンクから受信する場合に役立つことが多々あります。

4. 中間マシンから以下のコマンドを使用して、LifeKeeper サーバにトンネルを作成します。

```
ssh -X <LifeKeeper server>
```

中間マシンは LifeKeeper サーバとの間にかなり高い帯域幅の接続をもつはずなので、このコマンドには圧縮は不要です。

5. すべての操作が良好に完了した場合、以下のコマンドを実行してください。

```
echo $DISPLAY
```

「localhost:10.0」のような値に設定されます。値が設定されない場合、X11 の転送がいずれかの sshd 設定ファイルで無効になっています。

6. 以下のコマンドを実行して、LifeKeeper サーバから単純な `xterm` をポップアップ表示できることを確認してください。

```
/usr/X11R6/bin/xterm
```

7. `xterm` が表示された場合、以下のコマンドを使用して、LifeKeeper で `lkGUIapp` を実行できます。

```
/opt/LifeKeeper/bin/lkGUIapp
```

8. GUI コンソールが表示されるまで待ってください。Java は多くのグラフィックス動作を使用し、低速リンクで伝播するには時間がかかります (圧縮している場合でも)。しかし、最終的には GUI コンソールが表示されます。

## LifeKeeper Single Server Protection のアンインストール

LifeKeeper Single Server Protection 製品をアンインストールするには、付属の `mlk` ユーティリティを実行してください。

```
/opt/LifeKeeper/bin/mlk
```

これにより、すべての SIOS 製品の rpm がアンインストールされ、システムから `/opt/LifeKeeper` ディレクトリが削除されます。**慎重に使用してください。**

## Chapter 5: FAQ

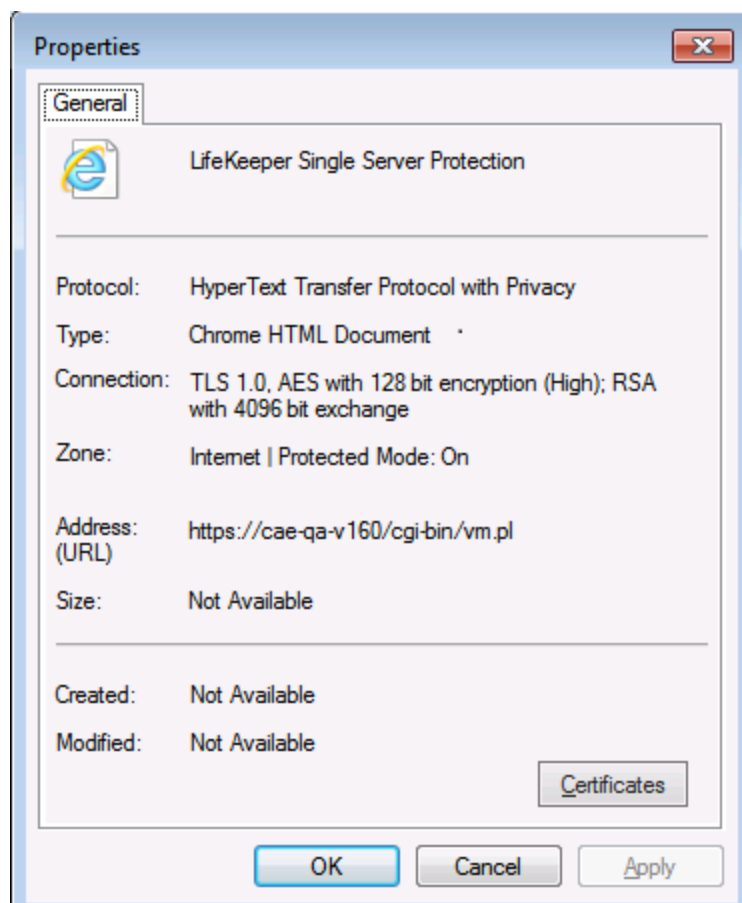
### SMC

#### 質問

(プラグインから) 使用している SMC を調べる方法 ありますか。

#### 回答

右クリックして、プラグイン Web ページの **[Properties]** を表示します。





## Chapter 6: トラブルシューティング

このセクションには、LifeKeeper Single Server Protection の制限または既知の問題と、SMC のトラブルシューティングのヒントが記載されています。

トラブルシューティング情報の詳細については、LifeKeeper テクニカルドキュメンテーションの LifeKeeper テクニカルノートおよびトラブルシューティングの各トピックを参照してください。

### 既知の問題と回避策

下記に、LifeKeeper Single Server Protection で明らかになっている制限または既知の問題を示します。

#### Core

##### バグ 2257

**LifeKeeper Single Server Protection および SteelEye Protection Suite のノードに**

**credstore 経由でアクセスするときに、正しい credstore キーが必要です**

**解決方法:** credstore を使用して、LifeKeeper Single Server Protection または SteelEye Protection Suite のノードの認証情報を保存するときに、credstore の認証情報キーについてホスト名の正しい形式 (`credstore -k <hostname>`) を使用する必要があります。

LifeKeeper Single Server Protection プラグインの場合、LifeKeeper Single Server Protection プラグイン画面の **[Hostname:]** フィールドに表示されるシステムのホスト名を使用して credstore を実行する必要があります。

SteelEye Protection Suite の場合、認証情報の保存に使用するホスト名は、コマンドラインツール (`lkpolicy` など) の `-d` 引数に使用するものと同じである必要があります。例えば、`lkpolicy -d mynode1` を実行する場合、`credstore -k mynode1` を使用して認証情報を保存する必要があります。この場合、認証情報の保存に FQDN を使用することはできません。FQDN を使用する場合は、`lkpolicy -dFQDN` を実行する必要があります。

**対応策:** LifeKeeper Single Server Protection または SteelEye Protection Suite のすべてのノードで機能するデフォルトの認証情報セット (`credstore -k default`) を保存した場合、この問題の影響を受けることはありません。

**バグ 2408****HA ハートビートが不正に有効になります**

2番目のリソースに障害が発生した後、lkvmhad が HA ハートビートを不正に有効にします。

**対応策** :`/etc/default/LifeKeeper` の `LKCHECKINTERVAL` に、VMware HA の [VM Monitoring Failure Interval] (VM の障害監視間隔) よりも大きい値を設定してください。 **注**

**記** :`LKCHECKINTERVAL` のデフォルト値は 120 秒です。また、これは、VMware HA の VM 監視の監視感度「low」のデフォルト値でもあります。

## GUI

**LifeKeeper Single Server Protection GUI の更新の問題**

GUI のリソースツリーが不規則に表示されることがあります (リソースの依存関係が正しく表示されない)。

**対応策** : GUI の更新を実行してください。

## IP

**バグ 2398****bonding NIC に割り当てられているものの、「暫定的な」状態のアドレスでは、IPv6 リソースが ISP としてレポートされます**

LifeKeeper Single Server Protection の IPv6 保護リソースが、'active-backup' (1) 以外のモード、かつ 2.6.21 以前の Linux カーネルの bonding インターフェース上にある場合、SLES システムでは、この IPv6 保護リソースが、「In Service Protected」(ISP: in service の保護) と不正に識別されません。IPv6 の bonding リンクは、解決できないアドレスを持つ「暫定的な」状態のままになります。

**対応策** : bonding インターフェースモードを 'active-backup' (1) に設定します。または、'active-backup' (1) 以外のモードの場合、リンク状態を「tentative (暫定的)」から「valid (有効)」に設定する更新したカーネルで操作します。

## Oracle

### バグ 2387

LifeKeeper Single Server Protection 環境の root ファイルシステムに、Oracle 階層を作成できません。

対応策: 以下の手順に従って、Oracle を新しいファイルシステムにコピーしてください。

Oracle データを格納できる十分に大きい新しいディスク (例: /dev/sdb) を作成してください。(注記: ディスクの大きさを見積もるために、/oracle ディレクトリの大きさを参考にできます。ログを含めるために、50 % 以上増加してください)。

fdisk を使用して、そのディスクに新しいパーティションを作成してください。

```
fdisk /dev/sdb
```

ファイルシステムを作成してください。

```
mkfs -t ext3 /dev/sdb1
```

このファイルシステムをマウントしてください (この例では /mnt/oracle を使用)。

```
mkdir /mnt/oracle
```

```
mount /dev/sdb1 /mnt/oracle
```

Oracle と Listener を停止してください。

Oracle を新しいファイルシステムにコピーしてください。

```
cd /oracle
```

```
cp -a * /mnt/oracle
```

(注記: データ量により、この手順には時間がかかることがあります)

新しいファイルシステムをアンマウントしてください。

```
umount /mnt/oracle
```

新しいファイルシステムを /oracle にマウントしてください。

```
mount /dev/sdb1 /oracle
```

Listener を開始し、次に Oracle を開始してください。

## SAP

### バグ 2388

SAP の場合、GUI を使用して階層を作成することはできません。

**対応策:** コマンドラインオプションを使用して、階層を作成してください。ただし、以下に示すように、コマンドラインの最後に番号 76 を指定してください。

```
$LKROOT/lkadm/subsys/appsuite/sap/bin/create <primary sys> <tag> <SAP SID>  
<SAP Instance> <switchback type> <IP Tag> <Protection Level> <Recovery  
Level> <Additional SAP Dependents> 76
```

コマンドラインの詳細については、「コマンドラインによる SAP の設定」を参照してください。

## SMC のトラブルシューティング

以下に、トラブルシューティングに関するヒントを示します。

**LifeKeeper Single Server Protection が監視するゲストの仮想マシンには、VMware Tools がインストール済みである必要があります。**

LifeKeeper Single Server Protection は、VMware Tools が有効にインストールされていない仮想マシンに接続できません。SteelEye 管理コンソールはこれらのマシンに接続できないので、マシンとそのリソースの完全なステータスを取得できません。vSphere Client 内の LifeKeeper Single Server Protection プラグインは、これらのゲスト (ツールがインストールされていない場合) に関するエラーメッセージを表示します。

**解決方法:** LifeKeeper Single Server Protection が保護するゲストマシンには、必ず VMware Tools をインストールしてください。