



**HP Serviceguard for Linux から
LifeKeeper for Linux への移行ガイド**

2011年4月1日
第1版

All Right Reserved, Copyright© 2011 SIOS Technology, Inc.

目次

第一章 はじめに

- 1.1 本書の目的
- 1.2 本書の構成と利用方法

第二章 Serviceguard for Linux と LifeKeeper for Linux の相違点

- 2.1 用語の定義
- 2.2 基本機能の比較
- 2.3 システム構成
- 2.4 運用・操作性

第三章 移行手順

- 3.1 対象システム要件
- 3.2 移行時のポイント
- 3.3 構築ステップ

第四章 関連サポート、サービス

- 4.1 LifeKeeper 年間サポート
- 4.2 LifeKeeper プレミアムサポート
- 4.3 LifeKeeper プロフェッショナルサービス

第五章 適用ソリューション

- 5.1 金融業窓口業務

参考情報

第一章 はじめに

1.1 本書の目的

本書は、HP Serviceguard for Linux で構築されたクラスタシステムを LifeKeeper for Linux によるクラスタシステムに移行する場合に留意すべき点、有効な情報を整理したものです。また、本書ではクラスタシステム構築に必要な基本的な技術については触れていませんので、必要に応じて他の関連技術文書を参照願います。

実際に移行作業を行う場合は、HP Serviceguard for Linux 及び LifeKeeper for Linux 双方についての詳細な技術的知識が必要となりますので、双方の製品マニュアル等を必要に応じて参照のうえ作業願います。

1.2 本書の構成と利用方法

本書は以下の章立てで構成されていますので、移行作業フェーズに応じて必要な章を参照願います。

第一章 はじめに

第二章 Serviceguard for Linux と LifeKeeper for Linux の相違点

第三章 移行手順

第四章 関連サポート、サービス

第五章 適用ソリューション

第二章では、Serviceguard for Linux で構築されたクラスタシステムから LifeKeeper for Linux によるクラスタシステムに移行・再構築する上で知っておかなければならない仕様の相違点について説明しています。

第三章では、実際にシステム移行を行う上で事前に留意すべき点、移行する際の手順について説明しています。

第四章では、移行時に有効な LifeKeeper for Linux のサポート、サービスについて説明しています。

第五章では、実際に Serviceguard for Linux から LifeKeeper for Linux に移行したシステム事例について、システム概要、移行のポイント、留意した点、等について説明しています。

参考情報では、本書を活用してシステム構築を行う場合に参考となる関連情報の参照先を示しています。

第二章 Serviceguard for Linux と LifeKeeper for Linux の相違点

2.1 用語の定義

Serviceguard で定義されている用語で LifeKeeper に該当する用語を以下にリストアップします(表 2-1)。留意すべき相違点や詳細の解説は後述します。

*NA:No Available

用語	Serviceguard for Linux	LifeKeeper for Linux
クラスター	クラスター	クラスター
クラスターメンバー	ノード	ノード
サービスの保護を提供するフレームワーク	ToolKit	Application Recovery Kit
複数ノードでの同時アプリケーション実行	マルチノードパッケージ または システムマルチノードパッケージ	NA
保護対象サービスの定義体	リソース	リソース
保護サービスの連携機能	パッケージ	リソース 階層構造
サーバー間での保護サービスの連携機能	NA	共有イクイバレンシ (2.2 項参照)
ハートビート機構	Ethernet, Infiniband	TCP メディア全て、シリアル通信

表 2-1 Serviceguard と LifeKeeper の用語比較

以下は代表的なコマンドの比較です(表 2-2)。各コマンドの目的と留意事項は後述します。

コマンド	Serviceguard for Linux	LifeKeeper for Linux
クラスター起動	cmruncl	lkstart
クラスター停止	cmhaltcl	lkstop
保護するサービスの起動	cmrunpkg	perform_action -t ResourceTag -a restore
保護するサービスの停止	cmhaltpkg	perform_action -t ResourceTag -a remove
クラスター情報の表示	cmviewcl	lcdstatus (-e)
クラスター及びシステムの全構成情報の採取	NA	lksupport
GUI Server の実行	# /etc/init.d/hpsmhd start	lkGUIserver start
GUI 管理ツールの実行	ブラウザによるアクセス	lkGUIapp またはブラウザによるアクセス
アプリケーションの HA 化	cmmakepkg cmcheckconf cmapplyconf	NA (リカバリーキットにより自動化)

表 2-2 Serviceguard と LifeKeeper のコマンド比較

2.2 基本機能の比較

LifeKeeper の機能面での特徴

LifeKeeper の機能面での大きな特徴は、GUI への依存度が高い点にあります。前述のコマンド一覧で紹介されている lkGUIapp により、Java ベースの GUI 管理ツールが実行されます。LifeKeeper における冗長化対象のアプリケーション環境の定義は、原則として全てこの GUI 管理ツールを通して行われています。LifeKeeper の GUI 管理ツールにおける代表的なオペレーションとコマンドの役割は 2.4 運用・操作性で後述します。

機能面での Serviceguard との差異には、同時アプリケーション実行機能が挙げられます。Serviceguard はマルチノードパッケージの利用により同時に複数ノードでのアプリケーション実行が可能ですが、LifeKeeper には該当する機能は存在しません。マルチノードでのアプリケーション実行を要件として検討している場合は、その目的を再検討してください。たとえばレスポンスタイムの減少が目的であれば、サーバーのスケールアップなどによるパフォーマンスの向上を検討してください。

機能	Serviceguard for Linux	LifeKeeper for Linux
最大ノード数	16 ノード (Fibrechannel 共有ストレージの場合。SAS/iSCSI 共有ストレージの場合は 2 ノード)	32 ノード (データミラー型の場合は 8 ノード)
複数ノードでの同時アプリケーション実行	マルチノードパッケージまたはシステムマルチノードパッケージにより可能	不可
クラスターの設定および管理ソフトウェア	Serviceguard Manager またはコマンドラインインターフェース	Java ベース GUI 管理ツールまたはコマンドラインインターフェース
アプリケーション保護情報の管理	保護対象アプリケーションごとにパッケージ構成ファイル、パッケージ制御スクリプトを作成して管理する	LCD(LifeKeeper 構成情報データベース)を各ノードに配置し管理する
クラスターとアプリケーション保護の設定方法	以下コマンドでパッケージ構成ファイル、パッケージ制御スクリプトを生成して定義 cmmakepkg テンプレート作成 cmcheckconf チェック cmapplyconf コンパイル	GUI ウィザードを使用し、Application RecoveryKit の機能を使用して設定
データミラー型クラスターのサポート	不可	可 (DataReplication ARK 必要)
スプリットブレインリカバリー	ロック LUN または Quorum Server	SCSI-3 PR または Quorum Server (Version 7.3 より)

表 2-3 Serviceguard と LifeKeeper の機能比較

リソースとリソース階層構造

LifeKeeper における保護サービスは 2.1 用語の定義 表 2-1 内での記述のとおり、リソースとして定義されます。リソース階層構造とは、一つ以上のリソースと依存関係により構成され、Serviceguard におけるパッケージとほぼ同様の機能を提供します。連携して稼働させるリソースを階層構造グループにまとめることで、あたかも単一のリソースであるかのように動作を制御します。リソース階層構造は、クラスター内の全ノードの LCD (LifeKeeper Configuration Database、2.2 基本機能 表 2-3を参照)に格納され、LCDに同じく格納される共有イクイバレンシにより、ノード間での連携が取られる仕組みになっています。

注. 共有イクイバレンシとは

共有イクイバレンシは LifeKeeper 独自のコンセプトです。上述のとおり、各ノードの LCD に格納されます。個々のリソース階層構造に対応し、ノード間にまたがったリソース階層構造の同等性を保証します。共有イクイバレンシの役割は、同時期にアクティブになるリソース階層構造を 1 ノード上に制限し、各ノードに分散したリソース階層構造の優先順位を定義し、障害時に優先的に切り替えるサーバーを決定します。

アプリケーションリカバリーキットと LifeKeeper コアの役割

LifeKeeper リソースはアプリケーションリカバリーキットにより構成されます。アプリケーションリカバリーキットは、Serviceguard における ToolKit とほぼ同等の役割をサポートします。LifeKeeper のアプリケーションリカバリーキットは、リソースの種類ごとに用意され、作成した対象リソースの制御を行います。

LifeKeeper における仮想 IP 機能や、データ領域の制御など、サービス冗長化を実現するための基本機能を提供するアプリケーションリカバリーキットをコアリカバリーキットと言います。

コアリカバリーキットおよびそれにより提供されるリソースは以下の通りです。

リカバリーキット	提供リソース
IP RecoveryKit	仮想 IP アドレス
Filesystem Recoverykit	ファイルシステムのマウント切り替え (Disk リソースを含む)
Raw I/O Recoverykit	RawDevice の切り替え
Generic Application RecoveryKit	任意のアプリケーションのリソース化

表 2-4 リカバリーキットと提供リソース

特定のアプリケーションの冗長化や、特定のストレージ環境のサポートを実現するアプリケーションリカバリーキットをオプションリカバリーキットと呼びます。

オプションリカバリーキットの対応アプリケーションは巻末に紹介するサイオステクノロジーおよび SIOS Corp.の web サイトを参照して下さい。

アプリケーションリカバリーキットは以下の 3 つのコンポーネントで構成されています。

- (ア) Action コンポーネント
- (イ) Administration コンポーネント
- (ウ) GUI メニューコンポーネント

Action コンポーネントは定義されたリソースを制御するスクリプトです。このスクリプトの特徴的な点は、実際のアプリケーション環境に合わせたカスタマイズが不要な点です。**Serviceguard** の **TookKit** は、対象アプリケーションの制御スクリプトテンプレートとして提供されますので、パッケージの構築にあたって、スクリプトの内容をアプリケーションの要件に従い、修正する必要があります。**LifeKeeper** の場合は、**GUI**メニューが提供する設計ウィザードに従って、アプリケーション情報を登録すれば自動的にリソースが生成・登録され、**Action** コンポーネントスクリプトの制御対象になります。リソースの生成・登録は **Administration** コンポーネントにより実行されます

LifeKeeper コアは、主にサーバー障害検知、ノード間通信、クラスター構成情報の制御、イベント通知機能の制御など、クラスターシステムにおけるノードの中核機能をサポートします。前述の **LCD** は **LifeKeeper** コアの構成要素の一つです。その他のコアの構成要素としては、通信をサポートする **LifeKeeper Communication Manager(LCM)**、さらにイベント通知や、リカバリー動作の制御を行う各種インターフェースが用意されています。**LifeKeeper** コアは障害イベントが発生すれば、リカバリー方法を検索し、判断し、実行をアプリケーションリカバリーキットに要求し、結果を受け取って構成情報のステータスを更新します。

Notes ! **LifeKeeper** のコアの構造を含むアーキテクチャーの詳細は、**LifeKeeper** テクニカルトレーニングで詳細の解説をしています。**LifeKeeper** テクニカルトレーニングプログラムは第四章 関連サポート、サービスの章をご参照ください。

イベント通知機能

Serviceguard は、主要なイベントを **SNMP** で管理する事が可能ですが、**LifeKeeper** も同様に、障害発生時のリカバリーなど重要なイベントが発生した場合、**SNMP** またはメールで通知する機能を用意しています。

2.3 システム構成

Serviceguard と **LifeKeeper** システム構成上の相違点は、**LifeKeeper** はより柔軟で、複数のレベルの信頼性に対応した構成例が選択可能な点にあります。代表的なシステム構成は、以下の構成例が想定されます。

共有ストレージによるHAクラスター構成

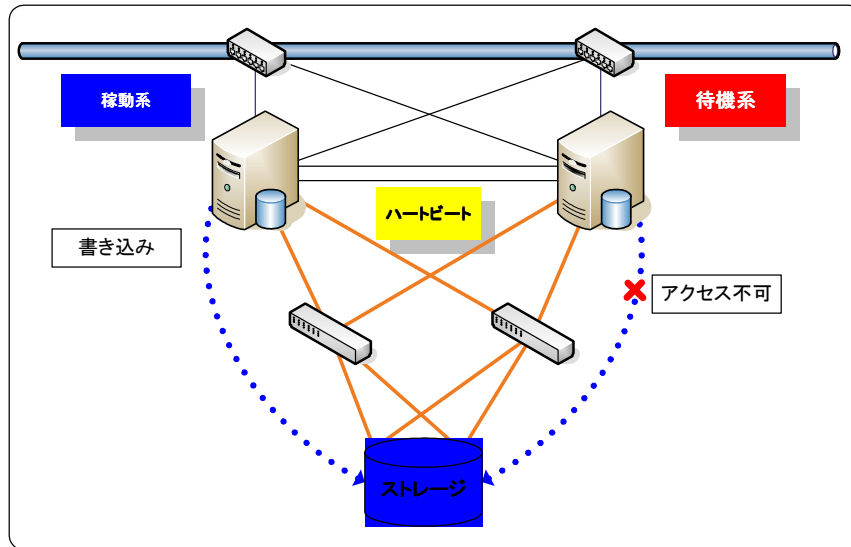


図 2-1 共有ストレージによるHAクラスター構成

共有ストレージを使用した構成は HA クラスターの代表的な構成例であり、Serviceguard、LifeKeeper ともにサポートしています。LifeKeeper で共有ストレージを使用した構成を実現するためには、サポート可能なストレージおよびホストアダプタを選定する必要があります。また選択したストレージ種類および構成によっては追加のストレージ用 ARK が必要になります。サポートするノード数は最大 32 ノードです。

サポートストレージ環境および追加 ARK については、後述するページで紹介するサイオステクノロジーまたは SIOS Corp の web サイトをご参照下さい。

共有ストレージ環境は、ストレージシステムの各部品単位での冗長性の担保が可能であり、システム全体での信頼性を向上させることが可能です。ただし、IT 基盤の総保有コストが拡大する傾向があります。

以下の構成は LifeKeeper でのみサポートされるシステム構成です。一般的な共有ストレージ環境と比較して、比較的システム投資を安価に抑えることが可能であり、ユーザーは要求される IT 基盤の信頼度に応じた適切な投資額を選択することができます。

NAS による LifeKeeper クラスター構成

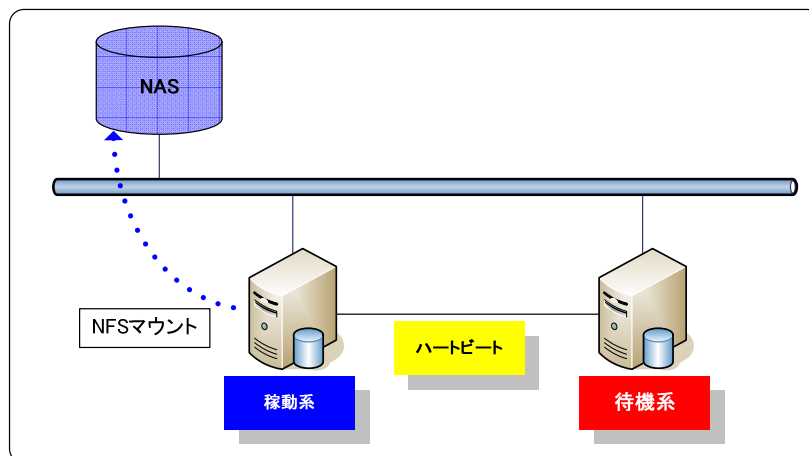


図 2-2 NAS による LifeKeeper クラスタ構成

LifeKeeper は NAS ストレージを使用した HA クラスタ構成をサポートします。NAS の場合は、機種は問いません。この構成には NAS ARK が必要です。サポートする最大ノード数は 32 ノードです。

データミラー型 LifeKeeper クラスタ

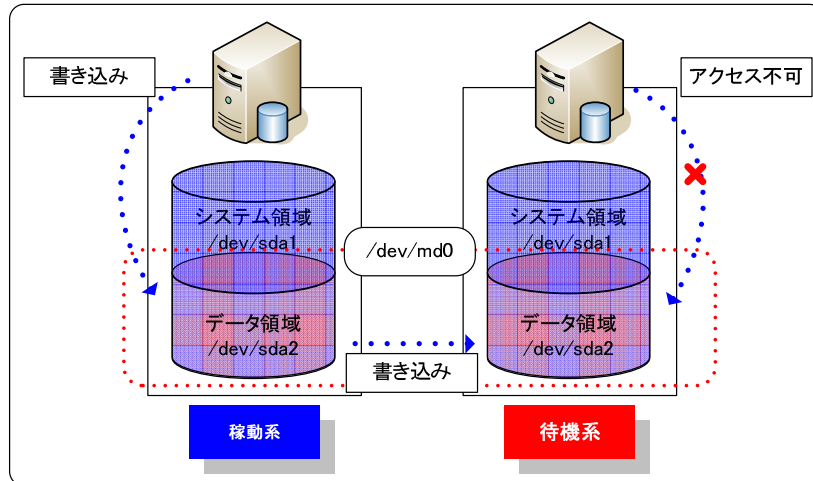


図 2-3 データミラー型 LifeKeeper クラスタ

LifeKeeper はデータミラー型 HA クラスタ構成をサポートします。データミラー型構成の実現には、各ノードに対して、DataReplication ARK の追加導入が必要になります。データミラー型 HA クラスタはサーバーノードのローカルストレージをネットワーク越しにミラーリングすることでデータの共有を実現するため、共有ストレージ環境への追加投資が必要なく、最も安価な構成での導入が可能です。データミラー型構成の場合、最大ノード数は 8 ノードに限定されます。

2.4 運用・操作性

クラスタ環境設定の手順

Serviceguard でクラスタ環境を構築する場合は、保護するアプリケーションごとにパッケージ構成ファイルとパッケージ制御スクリプトを作成する必要があります。

これらのファイルは通常はスクラッチから作成するより、テンプレートを生成してテンプレートから作成するのが一般的です。テンプレート作成用のコマンドは `cmmakepkg` です。また事前にカスタマイズ済のテンプレートも Toolkit により提供されます。

パッケージ構成ファイルのテンプレート `pkg.conf` を生成する方法

```
# cmmakepkg -p /usr/local/cmcluster/conf/pkg01/pkg.conf
```

パッケージ制御スクリプトのテンプレート `pkg.cntl` を生成する方法

```
# cmmakepkg -s /usr/local/cmcluster/conf/pkg01/pkg.cntl
```

生成されたテンプレートを実際の保護アプリケーションの環境にあわせて修正し、各クラスターノードにコピーします。各クラスターノード上で構成ファイルと制御スクリプトのチェックとコンパイルを行います。実行するコマンドは以下の通りです。

パッケージ構成ファイルのチェック

```
# cmcheckconf -P /usr/local/cmcluster/conf/pkg01/pkg.conf
```

パッケージ構成ファイルのコンパイル

```
# cmapplyconf -P /usr/local/cmcluster/conf/pkg01/pkg.conf
```

LifeKeeper の場合、パッケージに該当するリソース階層は、インストール時に各ノードに生成される LCD に格納されるため、個別に定義体を用意する必要はありません。LCD にリソースを登録する作業は、下記の GUI ツールによってウィザード形式で実行することが出来ます。

強力な GUI 管理ツール

LifeKeeper における冗長化設定と運用管理は、原則として全て GUI 管理ツールが使用されます。保護対象のサービスがオプションリカバリーキットで対応しているアプリケーションで構成されている場合、リソースを制御するスクリプトの開発は不要ですので、設計から運用にいたるまでの全ステップを GUI 上の設定のみで終了させることが可能です。

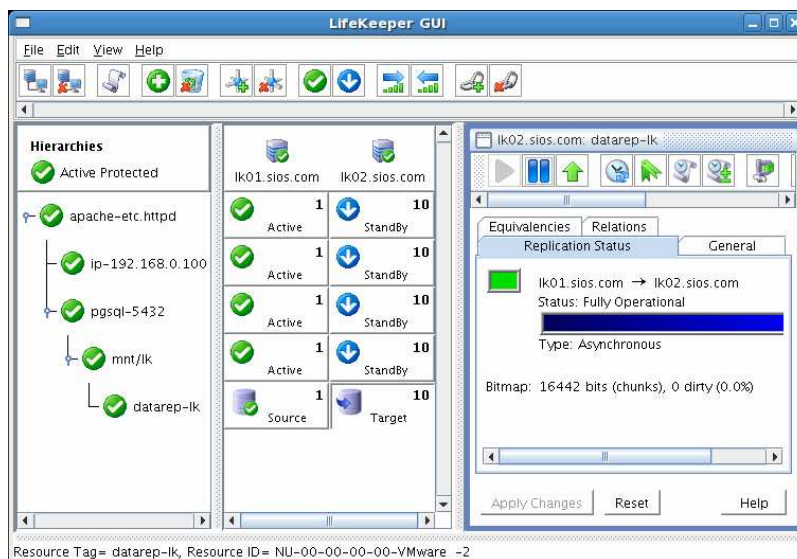


図 2-4 LifeKeeper 操作画面

LifeKeeper GUI 管理ツールの役割と構造

LifeKeeper GUI 管理ツールはネットワーク経由でアクセスするサーバークライアント型アプリケーションです。LifeKeeper の冗長化構成の設計は GUI 管理ツールでの実行を前提としており、GUI の全操作をコマンド操作で代替することは出来ません。LifeKeeper の設計構築作業の遂行にあたっては、GUI 環境の構成は必須です。運用フェーズにおいても、クラスター環境のモニターとオペレーションは、GUI 管理ツールを中心に実行するのが一般的です。

GUI 管理ツールは Java アプリケーションであり、実行には Java Runtime Environment が必要です。Java Runtime Environment は LifeKeeper インストールサポートにより提供されます。

GUI 管理ツールのクライアントインターフェースは X Window アプリケーションとして構成されているため、LifeKeeper ノード上で管理ツールのクライアントインターフェースを実行する場合は、X Window 環境が必要です。管理ツールのクライアントインターフェースは web ブラウザから JavaApplet として実行することもできます。

LifeKeeper におけるコマンドラインインターフェースの役割

LifeKeeper は処理の大半を GUI 経由で実行しますが、コマンドによる操作が必要な場合も存在します。コマンド操作が必要な第一のポイントとしては、LifeKeeper 初期導入後のクラスターシステムの起動が挙げられます。

クラスター起動は以下のコマンドで行います。（\$ LKROOT は LifeKeeper インストールディレクトリです。デフォルトは/opt/LifeKeeper です。

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

lkstart 初回実行後は、LifeKeeper 関連サービスはシステム起動時に自動実行されるように構成されるため、通常は初回一回のみの起動で差し支えありません。メンテナンスなどの理由で明示的に停止する場合は、以下のコマンドを実行します。

```
# $LKROOT/bin/lkstop
```

LifeKeeper GUI 環境の実行も同様にコマンドで行います。

```
# $LKROOT/bin/lkGUIserver start LifeKeeper GUI サーバー起動
```

```
# $LKROOT/bin/lkGUIapp LifeKeeper GUI 管理ツール起動
```

LifeKeeper GUI 管理ツール起動後は、基本のオペレーションは GUI 操作により実行します。しかし、LifeKeeper GUI 管理ツールはネットワーク障害が発生している場合などには、信頼性のある情報のモニターは困難になります。障害発生時には、各種コマンドを併用し、クラスター管理とモニターを行う必要があります。問題発生時に使用される代表的なコマンドとしては、LifeKeeper 構成の現在ステータス情報の取得を実行するコマンドとして `lcdstatus`、ステータス情報の他に LifeKeeper 及びシステムの構成情報とログをまとめて採取して `tar` ファイルを作成し、リモートでの問題解析をサポートするコマンドとして、`lksupoort` が用意されています。

第三章 移行手順

3.1 対象システム要件

LifeKeeper のハードウェア、ソフトウェア要件を記載します。

LifeKeeper for Linux で対応可能な主な HP 社製のサーバーおよびストレージは以下のようなものがあります。サーバーに関しては IA サーバー 32bit システムおよび 64bit システム (x86,AMD64,EM64T)に対応しています。ストレージに関してはリリースノートのサポートマトリクスに記載がありますので適時確認願います。

ハードウェア要件

サーバー: HP ProLiant Server ML/DL/SL/BL

ストレージ: HP StorageWorks MSA2300 シリーズ
 HP StorageWorks P2000 G3 FC
 HP StorageWorks P4300 G2 BaseModel
 HP StorageWorks EVA4400/6400/8400
 HP StorageWorks XP20000/XP24000/P9000

ソフトウェア要件

オペレーティングシステム: Red Hat Enterprise Linux v4 , v5 , v6
 SUSE Linux Enterprise Server v10 , v11

ミドルウェア: Apache , Postfix , Oracle , DB2 , Infomix , MySQL ,
 PostgreSQL/PostgresPlus , NFS Server , Samba , WebSphere MQ , JP1/AJS3
 Manager , JP1/AJS3 Agent

3.2 移行時のポイント

保護対象のサービスとして LifeKeeper の Recovery Kit オプションがあるものはスクリプト開発が不要*

Serviceguard for Linux では Toolkit と呼ばれるテンプレートスクリプトをベースに監視対象のアプリケーションの挙動を記述しパッケージ化しベリファイやコンパイルを経て、対象のクラスタノードにコピーし設定します。一方 LifeKeeper は Oracle や Apache といった主要なサービスに対して既に定義済みのスクリプト集である Recovery Kit が用意されています。ユーザーが個別に定義する必要はなく、GUI をウィザード形式で設定する事でそのサービスのクラスタ設定が対象の全ノードに対して完了できます。

*Recovery Kit オプションがないサービス(アプリケーション)に対しても Serviceguard と同様にコードを記載することにより、監視対象とすることができます。

スプリットブレイン対策は SCSI Reservation による排他制御

Serviceguard では Splitbrain 対策 (タイブレーカ) としてロック方式とクォーラムサーバ方式があります。ロック方式は共有ストレージ上に専用の領域を設け、クォーラムサーバ方式はクラスタノード以外のノードを調停役として設置します。LifeKeeper はどちらの方式も意識して設定する事はありません*。つまり、専用領域も追加ノードも不要です。代わりにスプリットブレイン対策は共有ディスクの排他制御の仕組みを利用して行います。

<参考>

[Linux][Windows]共有ストレージを使用しています。ハートビートが全て切断された場合、どのような挙動を示しますか？

<http://sios-steeleye.sios.com/modules/smartfaq/faq.php?faqid=15>

*LifeKeeper for Linux v7.2 (米国リリースバージョン 日本では v7.3 で対応)からはオプションで Quorum モデルのノードを立てる事が可能です。主に災害対策での信頼性の高い遠隔地フェイルオーバーを実現します。

LifeKeeper では LVM の定義は必要な場合だけ実施

Serviceguard の場合、監視対象のサービス毎に LVM のディスクグループを構成する必要があります。この LVM の仕組みは同時アクセスを防ぐ排他制御 (ホスタグ機能) にも利用されています。

LifeKeeper では LVM による論理ボリュームは必須ではありません。共有ストレージ上に LVM を構成した場合は、LVM Recovery Kit オプションが必要となります。

LifeKeeper と Serviceguard のクラスタサービス自身の起動・停止に関連する違い

- LifeKeeper では保護下のサービスを停止せずにクラスタサービス (LifeKeeper そのもの) を停止できるため、オンラインでのメンテナンスをする際に便利です。
Serviceguard ではサービス (パッケージ) を切り替えてからメンテナンスを行う必要があります。
- Serviceguard ではクラスタサービスの起動 (cmruncl) を行うと必ず優先度の高いノードでサービスが開始されます。LifeKeeper の場合、設定された優先度に関わらず「前回起動していたノード」でサービスが起動されます。

3.3 構築ステップ

次のステップで Serviceguard から LifeKeeper の移行を行います。

1. 計画と準備
2. Serviceguard の停止
3. LifeKeeper のインストールと設定
4. テスト
5. Serviceguard のアンインストール

1. 計画と準備

既存の Serviceguard コンフィグレーションの確認を行います。主に、下記の観点で調査を行い Serviceguard で冗長化を行うために施した OS 上の設定、ミドルウェアの設定を記録し、整理します。

- ハートビート構成
- 保護対象(監視対象のネットワーク)IP アドレス
- 保護対象(障害時に切替え、回復させるための)サービス
- 保護対象データ

(ア) HA 方式の決定

- クラスタシステムのハートビート構成

LifeKeeper ではハートビートとして TCP/TTY が利用できます。シリアルポート (TTY) によるパスは 1 つ、Ethernet(TCP) によるパスは最大 99 個まで構成可能です。TCP のみもしくは TTY、TCP の組み合わせでハートビートの経路を最低 2 つ用意可能か確認してください。必要に応じて NIC の増設などを行ってください。

(構成例)

TCP で2系統のハートビート

192.168.0.0/24 のネットワークと 172.16.0.0/16 のネットワーク

TCP と TTY で1つずつで2系統のハートビート

192.168.0.0/24 のネットワークと/dev/ttyS0

- 保護対象サービスネットワーク、IP アドレス

Serviceguard では「再配置可能 IP アドレス、パッケージ IP アドレス、移動 IP アドレス」と呼ばれるものです。

LifeKeeper では「仮想 IP アドレス」と呼び、クラスタシステムが外部に待ち受ける IP アドレスを Linux の IP エイリアス機能で付与します。このアドレスを障害発生時にノード間で切替えます。

MAC アドレスの引継ぎは行いませんが、切替え時には ARP キャッシュの書き換え要求を行います。

- チーミングされたネットワークインタフェース (bond0 など) に対応
- VLAN インタフェースに対応

LifeKeeper では仮想 IP アドレスの監視に Broadcast または Unicast の ping を発行しています。そのため、ping に応答可能な他ノード(ネットワーク機器やクラスタノード以外のサーバー)が同一セグメント上に配置されている必要があります。

□ 保護対象サービス

サービスの保護、つまりフェイルオーバーの単位は Serviceguard では「パッケージ」として扱われています。LifeKeeper ではこの「パッケージ」にあたるものを「リソース階層構造」と呼び、各サービスの単位を「リソース」と呼びます。「リソース階層構造」は一連のリソース(サービスリソース、仮想 IP アドレス、共有ディスクなど)が紐づいたツリー形で構成されます。LifeKeeper のオプションである Recovery Kit はこの一連のリソースツリーを保護対象のサービスに紐づけ自動的に作ります。それぞれリソースが適切な依存関係(共有イクイバレンシ)を持って構成されるため、ユーザーがサービス毎にフェイルオーバー設計する必要はありません。これによりまず、LifeKeeper の Recovery Kit オプションを使って構成可能な範囲を確認し、Recovery Kit にないアプリケーションの保護が必要であれば Generic ARK*の構成を検討します。

*Application Recovery Kit にないサービスを保護対象とする場合は、別途スクリプトを作成し Generic ARK という LifeKeeper の機能を使って、リソースとして登録します。その際、他のリソースとの階層はユーザー側で定義する必要があります。

□ 保護データおよび格納領域

保護対象のサービスが利用するデータは必ず共有ストレージ上、もしくは後述するミラーボリューム上に配置する必要があります。例えば共有すべきデータには以下のようなものがあります。

- DBMS のデータベース(表領域/テーブルスペース、トランザクションログなど)
- メールのキューやスプール
- Web のコンテンツ
- ファイルサーバのデータ領域

次に、保護対象のサービスが利用する共有データの配置を検討します。LifeKeeper では一般的な外部ストレージ上の領域をクラスタノード間で共有する方法に加え、外部ストレージを用いないシェアードキャッシングの構成も可能です。この構成は SteelEye Data Replication(以下 SDR、図 3-1)を用いクラスタノードそれぞれに接続されるローカルのディスクをネットワーク越しにミラーし、そのミラーボリュームを共有します。

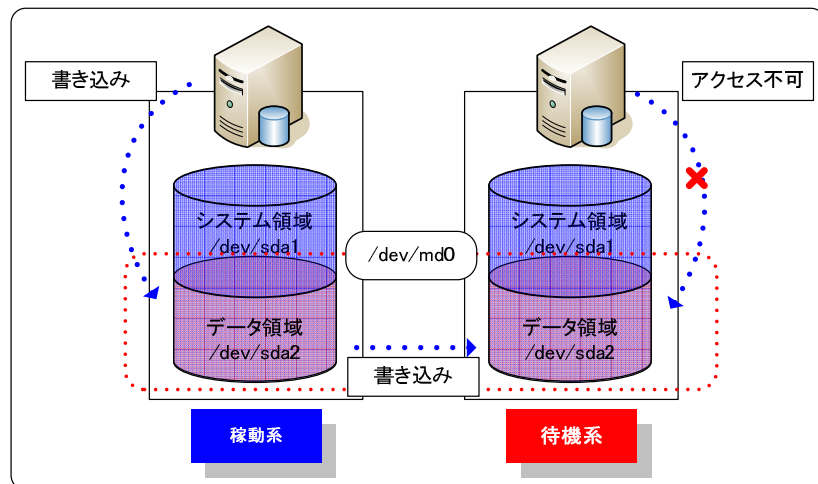


図 3-1 共有ディスクを使用しない構成

共有ストレージの構成では DAS、FC-SAN、IP-SAN (iSCSI)、NAS (図 3-2) を利用できます。その場合、認定されているストレージについては、LifeKeeper のリリースノートでご確認ください。後者の SDR 構成はストレージには依存していません。

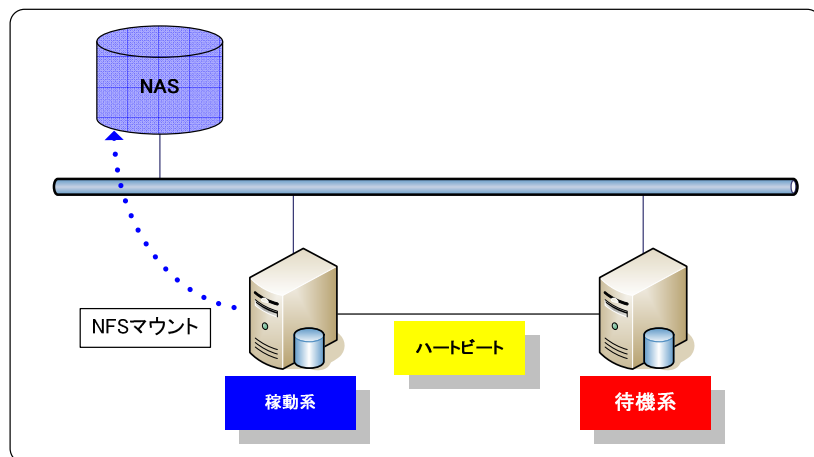


図 3-2 NAS による共有ストレージ構成

(イ) LifeKeeper の必要条件確認

- ディストリビューションの確認
サポートされているディストリビューションやカーネルバージョンを確認してください。
- クラスタシステムの名前解決
LifeKeeper ではクラスタシステムがお互いに名前を解決できる必要があります。ローカルの `hosts` ファイルや DNS、その他ネームサービスにて解決できるように設定されているかを確認してください。また、`localhost` も 127.0.0.1 に解決される必要があります。

□ GUI 環境について

LifeKeeper はクラスタシステムを構築する際に GUI 上で行います。そのため X Window System やデスクトップ環境がインストールされている必要があります。また、GUI 環境が使用できない場合などは、リモートから Web ブラウザ経由で GUI を起動し、ブラウザからクラスタシステムを構築・運用等することも可能です。ただし、この場合 GUI を操作するリモート端末からもクラスタシステムの名前解決が可能であり、さらに以下の通信経路を確保しておく必要があります。

- WebUI(mhttpd)=81 番
- GUI(RMI 通信に使用される)=82 番
- 1024 番以降のポート

2. Serviceguard の停止

Serviceguard を停止します。また、OS 起動時に自動的に Serviceguard が自動起動しないように設定を行います。

アプリケーションパッケージの停止

```
# cmhaltpkg <パッケージ名>
```

Serviceguard の停止

```
# cmhaltcl
```

Serviceguard が自動起動しないように、自動起動を定義しているファイルを編集します。

```
# vi /usr/local/cmcluster/conf/cmcluster.rc
```

「AUTOSTART_CMCLD=1」を「AUTOSTART_CMCLD=0」に変更

Serviceguard クラスタを構成していた際に実施していた、OS 上の設定を必要に応じて元にもどしてください。(例:リモートアクセス制御や、パケットフィルタ等)

3. LifeKeeper のインストールと設定

(ア) LifeKeeper のインストール

LifeKeeper のインストールにつきましては LifeKeeper のスタートアップガイドをご参照ください。

LifeKeeper for Linux スタートアップガイド

<http://sios-steeleye.sios.com/modules/mydownloads/viewcat.php?cid=1>

(イ) 設定

「LifeKeeper for Linux スタートアップガイド」では コミュニケーションパス(ハートビート)と仮想 IP アドレスの設定までを解説しています。Web や Database といったサービスを LifeKeeper で保護するためにはそれぞれのサービス用に提供されている Recovery Kit オプションのマニュアルをご参照ください。

LifeKeeper Recovery Kit に関連するドキュメントは、次のサイトからオンラインで入手できます。

http://www.steeleye.com/Linux_95.htm

4. テスト
クラスタ設定の確認や、フェイルオーバーテストを行ってください。
5. Serviceguard のアンインストール
Serviceguard アンインストール手順については、「HP Serviceguard for Linux リリースノート」の「Serviceguard for Linux のアンインストール」をご参照ください。

第四章 関連サポート、サービス

以下にサイオステクノロジーが提供可能な LifeKeeper 関連サポートおよびサービスをご案内致します。

本ページに掲載しているサポートおよびサービスの内容は、予告なく変更・改訂する場合がありますので、予めご了承下さい。

4.1 LifeKeeper 年間サポート

LifeKeeper 環境の技術的な問題へのテクニカルサポートをメールで提供するサービスです。

年単位でご契約頂き、以下の時間にメールでのお問い合わせが可能です。

サポート受付時間

平日月曜日-金曜(年末年始、祝祭日を除く)

時間 9:00-17:30

問い合わせ回数 無制限

主な内容

- LifeKeeper 導入作業やメンテナンス時に発生した問題についてのお問い合わせ。
- LifeKeeper 製品の仕様等に関する各種問合せ。
- LifeKeeper エラーやログの内容に関する問合せ。
- LifeKeeper ライセンスの取得、導入に関する問合せ。

4.2 LifeKeeper プレミアムサポート

LifeKeeper クラスターを導入したシステムにおいて、シビリティレベル1に相当する問題が発生した場合に電話でのサポートを提供するサービスです。このサービスにご契約いただくには、LifeKeeper 年間サポートにもご契約いただいている必要があります。

シビリティレベル 1(緊急)

シビリティ1の問題は、お客様の業務システムのダウンなどにより、機能していない、またサービスが提供できていないといった状況が適合します。

サポート受付時間

日曜-土曜 365 日対応

0:00-24:00

問い合わせ回数 無制限(シビリティ1のみ)

4.3 LifeKeeper プロフェッショナルサービス

サイオステクノロジープロフェッショナルサービスは、LifeKeeper 導入予定または検討中のお客様に、課題・要件定義から運用に至るまでの幅広いフェーズで、経験豊富なエキスパートが技術支援を提供します。Serviceguard によるクラスター環境からのマイグレーションやリプレースをご検討されている場合は、専用のツールを用い、最適なシステム設計をご提案致します。

システム可用性向上サービス

- ・ ワークショップ、インタビューを通し、お客様のシステム環境の課題やリスクをアセスメントいたします。
 - ・ システムの課題に対応しキャパシティプランニング、パフォーマンスプランニングを行い、最適なシステム環境の提案をいたします。
- #### システム冗長化設計支援サービス
- ・ システム環境の課題に基づき、具体的なシステム冗長構成の基本設計および詳細設計の策定支援を行います。

Generic ARK スクリプト開発サービス

- ・ LifeKeeper 標準 ARK 非対応のアプリケーションを HA 化するためのスクリプト開発を行います。

LifeKeeper 導入サービス

- ・ 事前のヒアリングを行いクラスターシステムの設計支援を実施し、LifeKeeper の導入と導入後の動作テストをサイオスのエンジニアが実施いたします。構築情報とテスト結果は文書化し納品いたします。

LifeKeeper 構築支援サービス

- ・ お客様の実施する LifeKeeper の導入設計作業と動作テストにサイオスのエンジニアがオンサイトでの立会い、または電話にて待機いたします。構築作業中問題が発生した場合は、原因調査と問題解決に対応いたします。

LifeKeeper トレーニングサービス

- ・ 以下の二つの定期トレーニングプログラムおよびお客様の運用する LifeKeeper 環境にあわせたカスタマイズトレーニングを提供します。
 - ・ LifeKeeper for Linux テクニカルトレーニング I
LifeKeeper によるクラスター環境構築に必要なステップを学習します。
 - ・ LifeKeeper for Linux テクニカルトレーニング II
LifeKeeper 運用フェーズにおける基本知識とトラブルシューティング、さらにその背景にあるアーキテクチャーを学習します。

障害復旧支援パック

- ・ お客様の環境に応じた LifeKeeper 環境の障害時対策マニュアルを提供いたします。

バックサポート契約

- ・ ご契約のお客様に選任のテクニカルアカウントマネージャー (TAM) をご用意し、担当 TAM が一環してお客様の LifeKeeper/DataKeeper 環境のテクニカルサポートの窓口としてご質問を承ります。
問題解決に必要な場合、担当 TAM はオンサイトでのサポートも実施いたします。

第五章 適用ソリューション

5.1 金融業窓口業務

【対象業務、システム概要】

対象業務: 銀行支店窓口業務

システム概要:

窓口業務処理を行う情報系システム。AP サーバーと対象各種データを格納する DB サーバー(Oracle)からなる三層構造システム。LifeKeeper により DB サーバーを HA 化。

【システム構成概要】

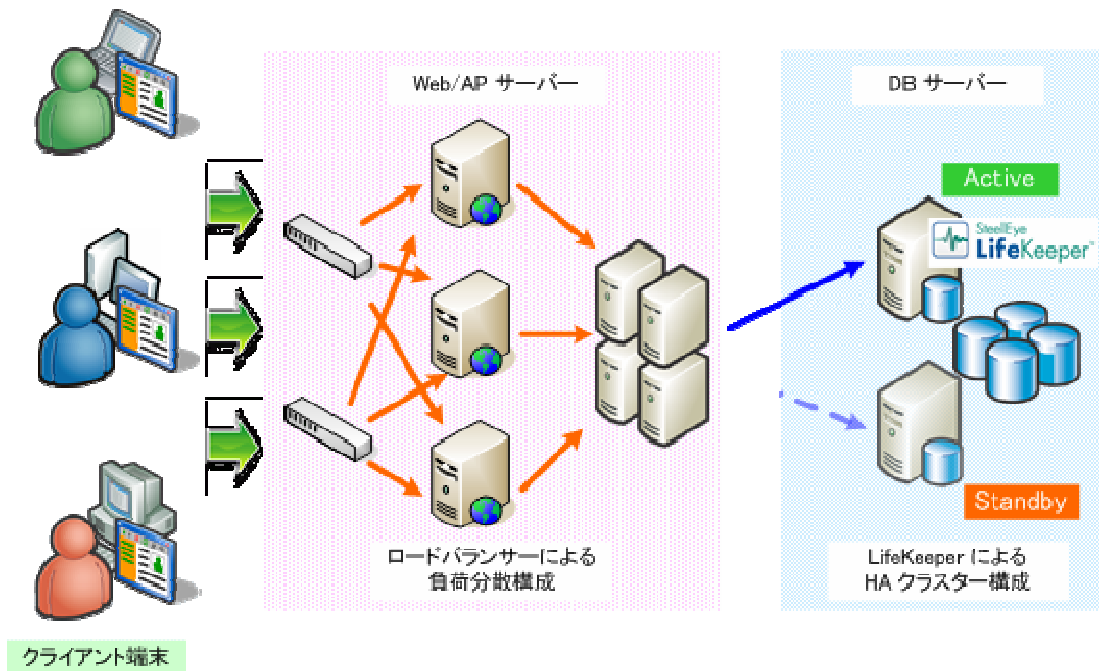


図 5-1 金融業窓口業務システム構成概要

【移行のポイント】

業務特性としてシステム障害時に早期解決が必要なため、Serviceguard では個別にログ及びシステム情報採取ツールを作成し適用。LifeKeeper ではログ、システム情報採取機能(lksupport)が製品で標準提供されており個別に情報採取ツールを開発する必要がなかった。

【移行後のコメント】

- LifeKeeper の場合、監視対象としてアプリケーション、ミドルウェア、OS、ハードウェアといったそれぞれの層について個別に管理できるので運用管理が楽になった。
- システム構築時 LifeKeeper は操作が GUI ベースのため作業が容易であり工数が従来に比べ少なくて済む。

【留意点】

- LifeKeeper では、システム構築時 GUI での操作を推奨しているため、GUI が使用できる環境を準備して構築する必要がある

参考情報

- LifeKeeper / DataKeeper ユーザーサイト

<http://sios-steeleye.sios.com/>

- SIOS Technology Corp. (Document)

http://www.steeleye.com/Linux_95.htm

- 日本 HP 社での LifeKeeper 関連情報

http://h50146.www5.hp.com/products/software/oe/linux/mainstream/support/doc/other/ha_cluster/

【免責事項】

本書の内容は、予告なしに変更されることがあります。また、本書に記載された情報は可能な限り正確な情報を記載するよう務めておりますが、校正上等の誤りにつきましては責任を負いかねますのでご了承ください。本書を利用した結果についてはいかなる保証も行いかねますのでご了承ください。

【商標】

Linux は Linus Torvalds の登録商標です。

HP/UX、HP Serviceguard は、Hewlett-Packard Company の米国及びその他の国における登録商標です。

Java は、Oracle Corporation 及びその子会社、関連会社の米国及びその他の国における登録商標です。

LifeKeeper は、SIOS Technology Corp.の登録商標です。