Dell™ PowerVault™ Encryption Key Manager

ユーザーズ・ガイド

Dell™ PowerVault™ Encryption Key Manager

ユーザーズ・ガイド

目次

図

表

まえがき

本書について

本書には、Dell™ Encryption Key Manager のインストールおよび操作に必要な情報および説明が記載されています。以下のものに関する概念および手順が含まれています。

- 暗号化対応の LTO 4 および LTO 5 テープ・ドライブ
- 暗号鍵
- デジタル証明書

本書の対象読者

本書は、重要データのセキュリティーおよびバックアップを行うストレージおよびセキュリティーの管理者や、稼働環境における Encryption Key Manager サーバーのセットアップおよびメンテナンスを支援する人を対象としています。読者は、ストレージ・デバイスおよびネットワークについて実践的な知識を有していることが前提です。

本書で使用される規則と用語

本書では、以下のような書体の規則を使用しています。

表 1. 本書で使用される表記法

規則	使用法
太字	**太字**で表示された語や文字は、コマンド名、ファイル名、フラグ名、パス名、選択されたメニュー・オプションなど、示されたとおりにユーザーが入力する必要のあるシステム・エレメントを示します。
モノスペース	モノスペース書体は、例、ユーザーが指定するテキスト、およびシステムが画面に表示する情報を示します。
イタリック	*イタリック* で表示された語や文字は、ユーザーが指定する必要のある可変値を示します。
[項目]	オプション項目を示します。
{項目}	フォーマット記述および構文記述内の、項目を選択する必要があるリストを囲みます。
|	垂直バーは、選択項目のリスト内の項目を区切ります。
<キー>	ユーザーが押すキーを示します。

注意の注記

注意の注記は、プログラム、装置、システム、またはデータに損傷を与える可能性があることを示します。注意の注記には感嘆符シンボルが付いていることがありますが、これは必要条件ではありません。注意の注記の例は、次のとおりです。

重要: 電動ドライバーを使用してこの手順を実行すると、テープを損なう可能性があります。

関連資料

詳細については、以下の資料を参照してください。

- 「*Getting Started with the Dell™ PowerVault™ TL2000 and TL4000 Tape Libraries*」では、インストールに関する情報を記載しています。
- 「*Dell™ PowerVault™ TL2000 Tape Library and TL4000 Tape Library SCSI Reference*」では、サポートされる SCSI コマンドおよび SCSI インターフェースの動作を制御するプロトコルについて説明します。

Linux 情報

Red Hat 情報

以下の URL は、Red Hat Linux® システムに関するものです。

- **http://www.redhat.com**

SuSE 情報

以下の URL は、SuSE Linux システムに関するものです。

- **http://www.suse.com**

Microsoft Windows 情報

以下の URL では、Microsoft® Windows® システムに関する情報にアクセスできます。

- **http://www.microsoft.com**

オンライン・サポート

次の関連資料については、**http://support.dell.com** にアクセスしてください。

「*Dell Encryption Key Manager Quick Start Guide*」では、基本構成のセットアップに関する情報を記載しています。

次の関連資料については、**http://www.dell.com** にアクセスしてください。

「*Library Managed Encryption for Tape*」ホワイト・ペーパーでは LTO テープ暗号化に関する最良事例を紹介しています。

はじめにお読みください

Dell の連絡先

米国の場合、800-WWW-DELL (800-999-3355) に電話してください。

注: アクティブなインターネット接続がない場合、連絡先情報を仕入れ送り状、パッキング・スリップ、請求書、または Dell 製品カタログで見つけることができます。

Dell は、オンラインおよび電話によるサポートおよびサービス・オプションをいくつか提供しています。利用可能かどうかは国および製品によって異なり、一部のサービスはお客様の地域でご利用になれない場合があります。営業、技術サポート、またはカスタマー・サービスについて Dell に問い合わせるには、次のようにします。

1. **http://support.dell.com** にアクセスします。
2. ページ下部にある「**国・地域の選択**」ドロップダウン・メニューでお客様の国または地域を確認します。
3. ページの左側にある「**お問い合わせ**」をクリックします。
4. 必要に応じて適切なサービスまたはサポート・リンクを選択します。
5. お客様のご都合に合った Dell への連絡方法を選択してください。

第 1 章 テープ暗号化の概要

データは、競争の激しいビジネス環境において最も価値の高いリソースの 1 つです。そのデータを保護し、データへのアクセスを制御し、認証性を検証すると同時に、その可用性を維持することは、セキュリティーに対する意識が高い現代社会における優先事項です。データ暗号化は、このようなニーズの多くに対応するツールです。Dell Encryption Key Manager (以下、Encryption Key Manager と表す) は、暗号化のタスクを簡素化します。

LTO 4 および LTO 5 ドライブは、すべての LTO 4 および LTO 5 データ・カートリッジに書き込まれるデータを暗号化することができます。 この新しい機能によって、サーバーで実行される暗号化に伴う処理オーバーヘッドおよび性能低下、または専用装置の経費が生じることなく、保管されたデータに対するさらに強力なセキュリティー手段が提供されます。

テープ・ドライブ暗号化ソリューションは、次の 3 つの主要なエレメントで構成されます。

暗号化対応テープ・ドライブ
ライブラリー・インターフェースを使用して、すべての LTO 4 および LTO 5 テープ・ドライブを使用可能にする必要があります。

テープ・ドライブについて詳しくは、 2-2 ページの『ハードウェアおよびソフトウェアの要件』を参照してください。

暗号鍵の管理
暗号化では、連続的な層にあるいくつかの種類の鍵を使用します。これらの鍵の生成、維持、制御、および伝送は、暗号化テープ・ドライブが取り付けられているオペレーティング環境によって異なります。アプリケーションの中には鍵管理を実行できるものがあります。そのようなアプリケーションが備わっていない環境、またはアプリケーションにとらわれない暗号化が必要な環境向けに、Dell Encryption Key Manager は、必要なすべての鍵管理タスクを実行します。 1-3 ページの『暗号化の管理』で、これらのタスクについて詳しく説明します。

暗号化ポリシー
これは、暗号化を可能にするために使用される方式です。これには、暗号化されるボリュームと、鍵選択のための仕組みを決定する規則が組み込まれています。これらの規則をセットアップする方法と場所は、オペレーティング環境によって異なります。詳しくは、 1-3 ページの『暗号化の管理』を参照してください。

コンポーネント

Encryption Key Manager は、Java 環境の一部であり、Java セキュリティー・コンポーネントをその暗号機能に使用します。(Java セキュリティー・コンポーネントの詳細については、関連資料のセクションを参照してください。) Encryption Key Manager には、その振る舞いを制御するのに使用される主要コンポーネントが 3 つあります。それらは、次のものです。

Java セキュリティー鍵ストア

鍵ストアは、Java Cryptography Extension (JCE) の一部として定義されるもので、Java セキュリティー・コンポーネントの 1 つのエレメントです。つまり、Java Runtime Environment の一部と言えます。鍵ストアは、暗号操作を実行するのに Encryption Key Manager が使用する証明書と鍵 (あるいは証明書および鍵へのポインター) を保持します。必要に合わせて、さまざまな動作特性を提供するいくつかのタイプの Java 鍵ストアがサポートされています。これらの特性については、2-4 ページの『鍵ストアに関する考慮事項』で詳しく説明します。

ご自分の鍵ストア・データを保持することの重要性は、どれほど強調しても強調しすぎることはありません。ご自分の鍵ストアにアクセスできなければ、暗号化されたテープを暗号化解除することはできません。以下のトピックを注意深く読み、ご自分の鍵ストア・データの保護に使用できる方式を理解してください。

構成ファイル

構成ファイルにより、Encryption Key Manager の振る舞いを、組織のニーズに合わせてカスタマイズできます。以下の振る舞いの選択項目について、最初は 2-1 ページの『第 2 章 Encryption Key Manager 環境の計画』、次に 4-1 ページの『第 4 章 Encryption Key Manager の構成』、さらに構成オプションの完全セットについて説明する付録 B という具合に、本書で何度も説明しています。

テープ・ドライブ・テーブル

テープ・ドライブ・テーブルは、Encryption Key Manager がサポートするテープ・デバイスの記録を取るために EKM によって使用されます。テープ・ドライブ・テーブルは、編集不能なバイナリー・ファイルで、その場所は構成ファイルに指定されます。この場所は、必要に合わせて変更できます。

KeyGroups.xml ファイル

このパスワードで保護されたファイルには、すべての暗号キー・グループの名前と各キー・グループに関連付けられた暗号鍵の別名が格納されます。

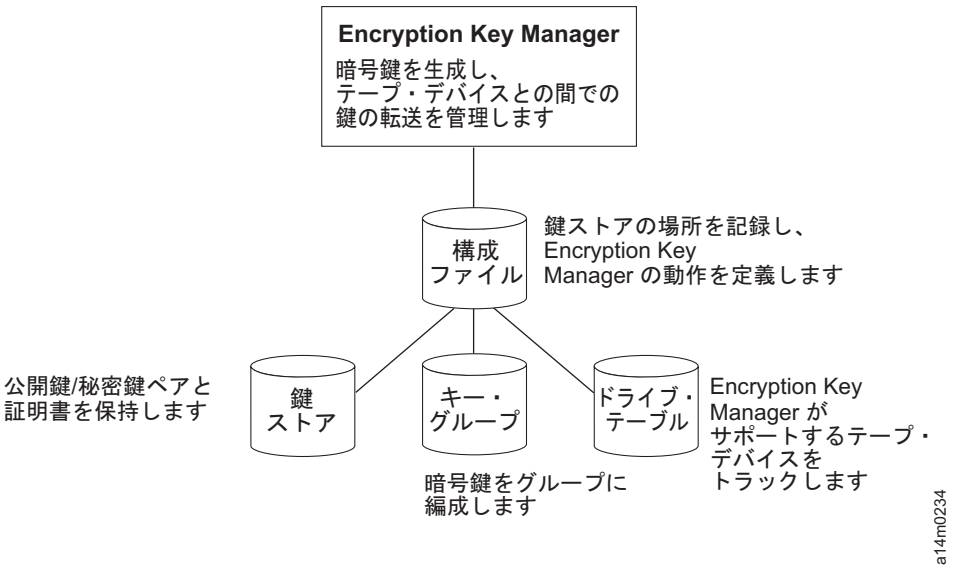

図 1-1. Encryption Key Manager の 4 つの主要コンポーネント

暗号化の管理

Dell Encryption Key Manager は Java™ ソフトウェア・プログラムの 1 つで、テープ・メディア（テープおよびカートリッジ形式）に書き込まれる情報の暗号化、およびテープ・メディアから読み取られる情報の暗号化解除に使用される暗号鍵の生成、保護、保管および保守に際して暗号化対応テープ・ドライブを支援します。Encryption Key Manager は、Linux (SLES および RHEL)および Windows 上で稼働するもので、エンタープライズ内の複数の場所にデプロイされる共用リソースとして、バックグラウンドで実行されるように設計されています。コマンド行インターフェース・クライアントは、堅牢なコマンドのセットを備えており、これらのコマンドにより Encryption Key Manager をユーザーの環境に合わせてカスタマイズしたり、操作をモニターすることができます。また、Dell Encryption Key Manager グラフィカル・ユーザー・インターフェース (GUI) でも多くのカスタマイズおよびモニタリング機能を使用することができます。 Encryption Key Manager は 1 つ以上の鍵ストアを使用して、すべての暗号化タスクに必要な証明書と鍵 (または証明書と鍵へのポインター) を保持します。詳しくは、2-4 ページの『鍵ストアに関する考慮事項』を参照してください。

Encryption Key Manager ホスト・サーバーの重要な構成情報: データ損失のリスクを最小限にとどめるには、Dell Encryption Key Manager プログラムをホスティングするマシンが、ECC メモリーを使用することを推奨します。Encryption Key Manager は、暗号鍵の生成を要求する機能、およびその鍵を LTO 4 および LTO 5 テープ・ドライブに引き渡す機能を実行します。鍵の構成要素は、Encryption Key Manager による処理時中は、ラップされた形 (暗号化された形式) でシステム・メモリーに常駐します。鍵の構成要素は、カートリッジに書き込まれるデータがリカバリー (暗号化解除) できるように、エラーなしで適切なテープ・ドライブに転送される必要があります。システム・メモリー内のビット・エラーが発生した結果、何らかの理由で鍵の構成要素が破損しており、かつ、その鍵の構成要素をカートリッジへのデータ書き込みに使用する場合、そのカートリッジに書き込まれるデータはリカバリーすること (つまり、後日暗号化解除すること) ができません。このようなデータ・エラーの発生を確実に防ぐために配置されている安全機能があります。ただし、Encryption Key Manager をホスティングするマシンでエラー訂正コード (ECC) メモリーが使用されない場合は、システム・メモリー内にある間に鍵の構成要素が破損し、この破損によりデータ損失が発生する可能性が残されます。この状況が発生する可能性は少ないですが、重要なアプリケーション (Encryption Key Manager など) をホスティングするマシンでは、ECC メモリーを使用することを常に推奨します。

Encryption Key Manager は、それ自身とテープ・ライブラリーとの間の TCP/IP 通信パスを介して送信される鍵生成または鍵取得の要求を待つバックグラウンド・プロセスとして機能します。テープ・ドライブは、暗号化されたデータを書き込むときに、最初に Encryption Key Manager の暗号鍵を要求します。Encryption Key Manager は、要求を受信すると、次のタスクを実行します。

Encryption Key Manager は、鍵ストアから既存の AES 鍵を取り出し、テープ・ドライブへ安全に転送するためラップします。テープ・ドライブで、鍵は着信時にアンラップされ、テープに書き込まれるデータの暗号化に使用されます。

暗号化されたテープが LTO 4 または LTO 5 ドライブによって読み取られると、Encryption Key Manager は、テープ上の鍵 ID の情報に基づいて鍵ストアから必要な鍵を取り出し、安全に転送されるようにラップしてテープ・ドライブに送信します。

暗号化管理の方法として、2 つの方法から選択できます。これらの方法は、暗号化ポリシー・エンジンが常駐する場所、暗号化ソリューションのために鍵管理が実行される場所、およびドライブへの Encryption Key Manager の接続方法が異なります。ご使用の稼働環境により、最適な方法が決定されます。鍵管理および暗号化ポリシー・エンジンが次の2 つの環境層のいずれか 1 つに置かれることがあります。

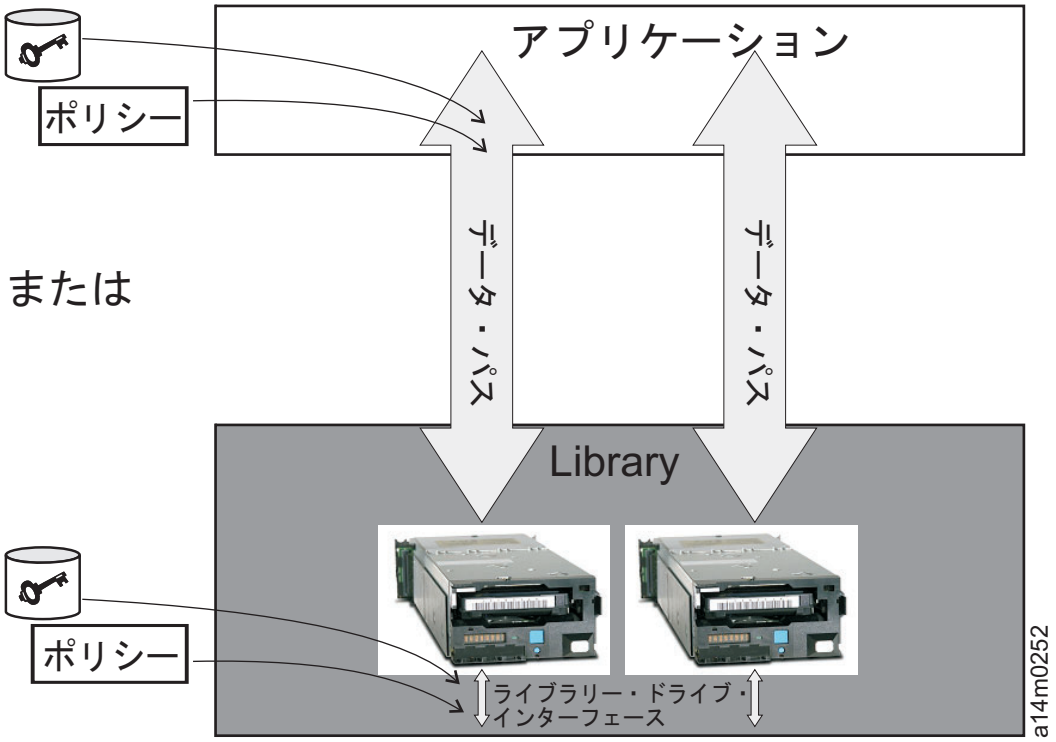

図 1-2. 暗号化ポリシー・エンジンおよび鍵管理の考えられる 2 つの場所

アプリケーション層

Key Manager から独立したアプリケーション・プログラムが、テープ・ストレージのデータ転送を開始します。サポートされるアプリケーションについては、『アプリケーション管理のテープの暗号化』を参照してください。

ライブラリー層

テープ・ストレージ (Dell PowerVault TL2000/TL4000 および ML6000 ファミリーなど) 用のエンクロージャー。最新のテープ・ライブラリーには、収容されている各テープ・ドライブへの内部インターフェースが組み込まれています。

アプリケーション管理のテープの暗号化

この方法は、オペレーティング環境で既に暗号化ポリシーと鍵を生成および管理を実行できるアプリケーションが実行されている場合に最適です。暗号化が使用される場合を指定するポリシーは、アプリケーション・インターフェースを介して定義されます。ポリシーおよび鍵は、アプリケーション層と暗号化テープ・ドライブとの間のデータ・パスを移動します。暗号化は、アプリケーションと暗号化対応テープ・ドライブとの間の相互作用の結果であり、システムおよびライブラリー層に対する変更は必要ありません。アプリケーションが暗号鍵を管理するため、アプリケーションの方式を使用して書き込まれ、暗号化されたボリュームを読み取るには、それらを作成したアプリケーションと同じアプリケーションによって、そのアプリケーション管理暗号化方式を使用するしかありません。

Encryption Key Manager は、アプリケーション管理テープ暗号化では必要とされる、つまり使用されることはありません。

暗号化を管理するために使用できるアプリケーションの最小バージョンは、次のとおりです。

- CommVault Galaxy 7.0 SP1
- Symantec Backup Exec 12

アプリケーション管理テープ暗号化は、以下の LTO 4 および LTO 5 テープ・ドライブでサポートされます。

- Dell™ PowerVault™ TL2000 テープ・ライブラリー
- Dell™ PowerVault™ TL4000 テープ・ライブラリー
- Dell™ PowerVault™ ML6000 テープ・ライブラリー

暗号化ポリシーおよび鍵の管理方法については、ご使用のテープのバックアップ用ソフトウェア・アプリケーションの資料を参照してください。

ライブラリー管理テープの暗号化

この方式は、 Dell™ PowerVault™ TL2000 テープ・ライブラリー、Dell™ PowerVault™ TL4000 テープ・ライブラリー、または Dell™ PowerVault™ ML6000 テープ・ライブラリー 内の LTO 4 および LTO 5 テープ・ドライブに使用します。鍵の生成および管理は、ライブラリー接続のホストで稼働する Java アプリケーションである Encryption Key Manager が行います。ポリシーの制御および鍵は、ライブラリーからドライブのインターフェースを介して渡されるため、暗号化は、アプリケーションには透過的となります。

暗号鍵について

暗号鍵は、特にデータのスクランブルおよびスクランブル解除を行うために生成されるランダム・ビット・ストリングです。暗号鍵は、それぞれの鍵が確実に固有のもので、かつ予測不能であるように設計されたアルゴリズムを使用して作成されます。このように作成された鍵ストリングが長いほど、暗号化コードを解読することが難しくなります。IBM 方式と T10 方式の両方の暗号化では、データの暗号化に 256 ビットの AES アルゴリズムが使用されます。256 ビットの AES は、米国政府によって現在認められ、かつ推奨されている暗号化標準です。この標準では、3 つの異なる鍵長さが許されます。256 ビット鍵は、AES で許される最長のものです。

Encryption Key Manager により、対称アルゴリズムと非対称アルゴリズムという 2 つのタイプの暗号化アルゴリズムが使用されます。対称、つまり秘密鍵暗号化では、暗号化と暗号化解除の両方に単一の鍵が使用されます。対称鍵暗号化は、一般的に大量のデータを効率的に暗号化するために使用されます。256 ビット AES 鍵は対称鍵です。非対称暗号化、つまり公開/秘密暗号化では、1 組の鍵が使用されます。一方の鍵で暗号化されたデータを暗号化解除できるのは、公開/秘密鍵ペアの他方の鍵を使用する場合に限られます。非対称鍵ペアが生成されると、公開鍵は暗号化に使用され、秘密鍵は暗号化解除に使用されます。

Encryption Key Manager では、対称鍵と非対称鍵の両方が使用されます。ユーザーまたはホスト・データの高速暗号化では対称暗号化が使用され、対称鍵の保護には非対称暗号化が使用されます (こちらは、必然的に速度が遅くなります)。

暗号鍵は、Keytool などのユーティリティーによって、Encryption Key Manager に対して生成されます。AES 鍵を生成する責任と、それらの鍵をテープ・ドライブに転送する方法は、暗号化管理の方法によって異なります。 ただし、Encryption Key Manager による暗号鍵の使用法と、他のアプリケーションによる使用法の違いを理解することは有用です。

Dell Encryption Key Manager による暗号鍵の処理

ライブラリー管理テープ暗号化では、暗号化されていないデータが LTO 4 または LTO 5 テープ・ドライブに送信され、Encryption Key Manager で使用可能な鍵ストアからの事前生成対称データ鍵 (DK) を使用して暗号文に変換されてから、テープに書き込まれます。Encryption Key Manager は、事前生成された DK をラウンドロビン方式で選択します。 事前生成されている DK では数が不足する場合、DK は複数のテープ・カートリッジで再利用されます。 DK は、Encryption Key Manager によって暗号化、つまりラップされた形で LTO 4 または LTO 5 テープ・ドライブに送信されます。LTO 4 および LTO 5 テープ・ドライブは、この DK をアンラップして暗号化または暗号化解除のために使用します。ただし、LTO 4 または LTO 5 テープ・カートリッジ上ではどこにも、ラップされた鍵は保管されません。暗号化ボリュームが書き込まれたら、DK は、別名または鍵ラベルに基づいてアクセス可能であり、ボリュームが読み取られるように Encryption Key Manager で使用可能でなければなりません。 1-8 ページの図 1-3 にこのプロセスを示します。

Dell Encryption Key Manager により、LTO 暗号化用の対称鍵をキー・グループに編成することができます。 これを使用すると、暗号化するデータのタイプ、鍵へのアクセス権を持つユーザーに従って、あるいはその他の重要な特性によって鍵をグループ化できます。詳しくは、 3-17 ページの『キー・グループの作成および管理』を参照してください。

他のアプリケーションによる暗号鍵の処理

アプリケーション管理テープ暗号化では、暗号化されていないデータが LTO 4 および LTO 5 テープ・ドライブに送信され、アプリケーションによって提供される対称 DK を使用して暗号文に変換されてから、テープに書き込まれます。DK は、テープ・カートリッジ上のどこにも保管されません。暗号化ボリュームが書き込まれたら、DK は、ボリュームが読み取られるように、アプリケーションが使用できる場所、例えばサーバー・データベースに入っている必要があります。

LTO 4 および LTO 5 テープ・ドライブは、Yosemite (Dell PowerVault TL2000 および TL4000 テープ・ライブラリー用)、CommVault、および Symantec Backup Exec などのアプリケーションを使用して、アプリケーション管理暗号化を行うことができます。

あるいは、T10 コマンド・セットを使用するアプリケーションが LTO 4 および LTO 5 テープ・ドライブを使用して暗号化を実行することができます。T10 コマンド・セットは、アプリケーションによって提供される対称 256 ビット AES 鍵を使用します。T10 は、各テープ・カートリッジについて複数の、固有の DK を使用でき、暗号化データおよび平文データを同じテープ・カートリッジに書き込むこともできます。アプリケーションは、テープ・カートリッジを暗号化する際に、アプリケーションによって決定された方法で DK を選択または生成し、それをテープ・ドライブに送信します。鍵は、非対称公開鍵ではラップ**されず**、テープ・カートリッ

ジ上には保管**されません**。暗号化データがテープに書き込まれたら、DK は、データが読み取られるように、アプリケーションが使用できる場所に入っている必要があります。

図 1-3 に、アプリケーション管理暗号化およびライブラリー管理暗号化によるテープ暗号化のプロセスを示します。

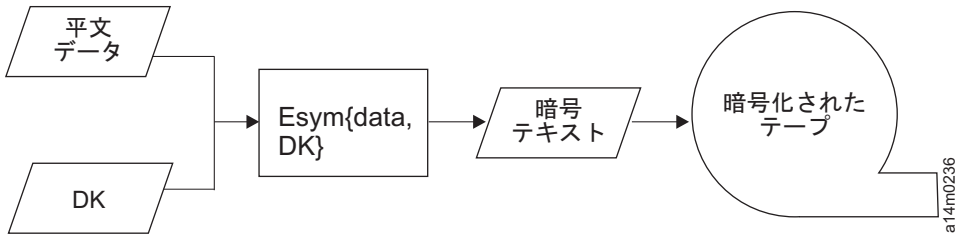

図 1-3. 対称暗号鍵を使用した暗号化： LTO 4 および LTO 5 テープ・ドライブ上でのライブラリー管理およびアプリケーション管理暗号化

要約

各ボリュームで使用できる暗号鍵の数は、暗号化の管理に使用されるテープ・ドライブ、暗号化標準、および方式によって異なります。LTO 4 および LTO 5 の透過暗号化 (すなわち、Encryption Key Manager でライブラリー管理暗号化を使用) の場合、DK が固有であるかどうかは、十分な数の事前生成鍵が Encryption Key Manager で使用可能かどうかによって異なります。

表 1-1. 暗号鍵の要約

暗号化管理方式	**使用される鍵**	
	IBM 暗号化	**T10 暗号化**
ライブラリー管理暗号化	1 つの DK/カートリッジ	該当せず
アプリケーション管理暗号化	複数の DK/カートリッジ	複数の DK/カートリッジ
DK = 対称 AES 256 ビット DK		

第 2 章 Encryption Key Manager 環境の計画

このセクションは、ニーズにあった最適の Encryption Key Manager 構成を判別できるようにする情報の提供を目的としています。暗号化方法をセットアップする方法を計画する際には、さまざまな要因を考慮する必要があります。

暗号化セットアップ作業一覧

テープ・ドライブの暗号化機能を使用する前に、特定のソフトウェアおよびハードウェアの要件を満している必要があります。以下のチェックリストは、これらの要件を満たす際役立つように考えられています。

Encryption Key Manager のセットアップ作業

テープを暗号化するには、まずEncryption Key Manager の構成および実行を行い、それが暗号化テープ・ドライブと通信できるようにしておく必要があります。Encryption Key Manager は、テープ・ドライブの取り付け中に実行している必要はありませんが、暗号化を行うには実行している必要があります。

- Encryption Key Manager サーバーとして使用するシステム・プラットフォームを決定します。
- 必要な場合は、サーバー・オペレーティング・システムをアップグレードします。(2-2 ページの『ハードウェアおよびソフトウェアの要件』を参照。)
- Java 無制限ポリシー・ファイルをインストールします。(2-2 ページの『ハードウェアおよびソフトウェアの要件』を参照。)
- Encryption Key Manager JAR をアップグレードします。(3-1 ページの『最新のバージョンの Key Manager ISO イメージのダウンロード』を参照。)
- 鍵、証明書、およびキー・グループを作成します。

  3-6 ページの『GUI を使用した構成ファイル、鍵ストア、および証明書の作成』

  3-17 ページの『キー・グループの作成および管理』
- 以下のステップは、追加の構成オプションを利用しない場合で、 3-6 ページの『GUI を使用した構成ファイル、鍵ストア、および証明書の作成』 の手順に従う場合は、必要ありません。
  - 必要に応じて、鍵および証明書をインポートします。(3-15 ページの『Keytool -importseckey を使用したデータ鍵のインポート』を参照。)
  - 構成プロパティー・ファイルを定義します。(4-1 ページの『第 4 章 Encryption Key Manager の構成』を参照。)
  - テープ・ドライブを Encryption Key Manager に定義するか、**drive.acceptUnknownDrives** 構成プロパティー値を on に設定します。(5-9 ページの『adddrive』を参照して明示的にドライブを定義するか、 4-1 ページの『テープ・ドライブ・テーブルの自動更新』 を参照してください。)
  - Encryption Key Manager サーバーを始動します。(5-1 ページの『Key Manager サーバーの始動、リフレッシュ、および停止』を参照。)

– コマンド行インターフェース・クライアントを始動します (5-6 ページの『コマンド行インターフェース・クライアント』を参照。)

ライブラリー管理テープ暗号化の計画

暗号化を実行するには、以下のものが必要です。

- 暗号化対応の LTO 4 および LTO 5 テープ・ドライブ
- 鍵ストア
- Dell Encryption Key Manager

ライブラリー管理テープの暗号化作業

1. LTO 4 および LTO 5 テープ・ドライブを取り付けてケーブルを接続します。
   - ライブラリー・ファームウェアを更新します (TL2000、TL4000、ML6000 は必要に応じて)。**http://support.dell.com** にアクセスします。
     – Dell™ PowerVault™ TL2000 テープ・ライブラリー でのファームウェアの必要最小バージョンは、5.xx です。
     – Dell™ PowerVault™ TL4000 テープ・ライブラリー でのファームウェアの必要最小バージョンは、5.xx です。
     – Dell™ PowerVault™ ML6000 テープ・ライブラリー ファミリーでのファームウェアの必要最小バージョンは、415G.xxx です。
   - 必要な場合は、テープ・ドライブ・ファームウェアを更新します。ファームウェアの必要最小バージョンは 77B5 です。
2. LTO 4 および LTO 5 テープ・ドライブおよびテープ・ライブラリーをライブラリー管理テープ暗号化用に有効にします (詳細については、Dell テープ・ライブラリー情報を参照)。
   - Encryption Key Manager サーバー IP アドレスを追加します。
3. ライブラリーの診断機能を使用して、Encryption Key Manager パスおよび暗号化構成を検査します (詳細については、Dell テープ・ライブラリー情報を参照してください)。

ハードウェアおよびソフトウェアの要件

注: 次の各プラットフォームに対する IBM バージョン の Java ランタイム環境 (JRE) のみが、Encryption Key Manager をサポートします。

Linux ソリューション・コンポーネント

オペレーティング・システム

- RHEL 4
- RHEL 5
- SLES 9
- SLES 10
- SLES 11

Encryption Key Manager (Linux で実行された場合)

表 2-1. Linux の場合の最小ソフトウェア要件

プラットフォーム	IBM Software Developer Kit	入手可能な URL
64 ビット AMD/Opteron/EM64T	Java 6.0 SR5	**http://support.dell.com**
32 ビット Intel® 互換		

テープ・ライブラリー

Dell PowerVault TL2000 テープ・ライブラリー、TL4000 テープ・ライブラリー、および ML6000 テープ・ライブラリーについて、ファームウェア・レベルが入手可能な最新のものであることを確認してください。ファームウェア更新については、**http://support.dell.com** にアクセスしてください。

テープ・ドライブ

LTO 4 および LTO 5 テープ・ドライブについて、ファームウェア・レベルが入手可能な最新のものであることを確認してください。ファームウェア更新については、**http://support.dell.com** にアクセスしてください。

Windows ソリューション・コンポーネント

オペレーティング・システム

Windows Server 2003、2008、および 2008 R2

Dell Encryption Key Manager

Encryption Key Manager の必要最小バージョンは、ビルド日付が 20070914 以降の 2.1 と、次の IBM ランタイム環境のいずれかです。

表 2-2. Windows の場合の最小ソフトウェア要件

オペレーティング・システム	IBM Runtime Environment
Windows 2003	• AMD64/EM64T アーキテクチャー上の IBM®64 ビット Runtime Environment for Windows、Java 2 Technology Edition、バージョン 5.0 SR5 • IBM 32 ビット Runtime Environment for Windows、Java 2 Technology Edition、バージョン 5.0 SR5
Windows 2008 および 2008 R2	AMD64/EM64T アーキテクチャー上の IBM 64 ビット Runtime Environment for Windows、Java 2 Technology Edition、バージョン 6.0 SR5

テープ・ライブラリー

Dell™ PowerVault™ TL2000 テープ・ライブラリー、Dell™ PowerVault™ TL4000 テープ・ライブラリー、および Dell™ PowerVault™ ML6000 テープ・ライブラリー について、ファームウェア・レベルが入手可能な最新のものであることを確認してく

ださい。ファームウェア更新については、**http://support.dell.com** にアクセスしてください。

テープ・ドライブ

LTO 4 および LTO 5 テープ・ドライブについて、ファームウェア・レベルが入手可能な最新のものであることを確認してください。ファームウェア更新については、**http://support.dell.com** にアクセスしてください。

鍵ストアに関する考慮事項

ご自分の鍵ストア・データを保持することの重要性は、どれほど強調しても強調しすぎることはありません。ご自分の鍵ストアにアクセスできなければ、暗号化されたテープを暗号化解除することはできません。以下のトピックを注意深く読み、ご自分の鍵ストア・データの保護に使用できる方式を理解してください。

JCEKS 鍵ストア

EKM は、JCEKS 鍵ストア・タイプをサポートします。

JCEKS (Unix System Services ファイル・ベース) は、EKM が実行するすべてのプラットフォーム上でサポートされるファイル・ベースの鍵ストアです。したがって、バックアップおよびリカバリーのためにこの鍵ストアの内容をコピーしたり、フェイルオーバーに備えて 2 つの EKM インスタンスを同期させておくことは、比較的容易です。JCEKS では、セキュリティーのために鍵ストアの内容をパスワード・ベースで保護することができ、比較的高いパフォーマンスが実現されます。FTP などのファイル・コピー方式が使用されることがあります。

暗号鍵と LTO 4 および LTO 5 テープ・ドライブ

Dell Encryption Key Manager およびそのサポートされているテープ・ドライブは、データを暗号化するのに対称 256 ビット の AES 鍵を使用します。このトピックでは、これらの鍵および証明書に関して理解しておく必要があることを説明します。

LTO テープ・カートリッジ用の LTO 4 または LTO 5 テープ・ドライブ上で暗号化タスクを実行するときに、Encryption Key Manager は、256 ビット AES 対称データ鍵のみを使用します。

LTO 4 または LTO 5 が鍵を要求した場合、Encryption Key Manager は、テープ・ドライブに指定された別名を使用します。テープ・ドライブの別名が指定されていない場合、symmetricKeySet 構成プロパティーで指定されたキー・グループ、鍵別名リスト、または鍵別名の範囲から別名が使用されます。テープ・ドライブの特定の別名が欠落している場合、別名は、鍵の使用量が均等になるように、ラウンドロビン方式で他のエンティティーから選択されます。

選択された別名は、鍵ストアにプリロードされている対称データ鍵 (DK) と関連付けられます。Encryption Key Manager は、テープ・ドライブが暗号化解除できる別の鍵を使用してこの DK をラップし、LTO 4 または LTO 5 テープ・ドライブに

送信してデータを暗号化します。DK は、TCP/IP を介して平文で送信されません。選択された別名は、また、データ鍵 ID (DKi) と呼ばれるエンティティーに変換され、暗号化されたデータと一緒にテープに書き込まれます。この方法で、Encryption Key Manager は DKi を使用して、LTO 4 または LTO 5 テープが読み取られるときにデータを暗号化解除するのに必要な、正しい DK を識別できます。

テープ・ドライブに別名を指定する方法は、5-9 ページの『CLI コマンド』 の **adddrive** および **moddrive** のトピックで説明します。3-12 ページの『LTO 4 および LTO 5 上での暗号化のための鍵と別名の生成』を参照してください。ここには、鍵のインポート、鍵のエクスポート、および symmetricKeySet 構成プロパティーでのデフォルト別名の指定に関する情報が記載されています。3-17 ページの『キー・グループの作成および管理』では、キー・グループを定義してそれを鍵ストアから別名で取り込む方法について説明します。

図 2-1 に、暗号化書き込み操作に向けて鍵がどのように処理されるかを示します。

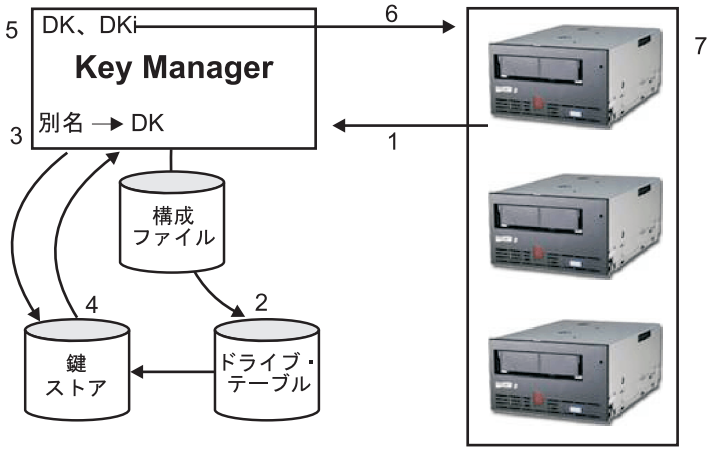

図 2-1. 暗号化書き込み操作に向けた、LTO 4 または LTO 5 テープ・ドライブ要求

1. テープ・ドライブが、鍵にテープを暗号化するよう要求します
2. Encryption Key Manager が、ドライブ・テーブル内のテープ・デバイスを検査します
3. 要求で別名が指定されておらず、かつ別名がドライブ・テーブルに指定されていない場合、Encryption Key Manager は、別名を keyAliasList の別名のセットまたはキー・グループから選択します
4. Encryption Key Manager は、鍵ストアから対応する DK を取り出します
5. Encryption Key Manager は、その別名を DKi に変換し、ドライブが暗号化解除できる鍵を使用して DK をラップします
6. Encryption Key Manager は、その DK および DKi をテープ・ドライブに送信します
7. テープ・ドライブがその DK をアンラップし、暗号化データと DKi をテープに書き込みます。

図 2-2 に、暗号化読み取り操作に向けて鍵がどのように処理されるかを示します。

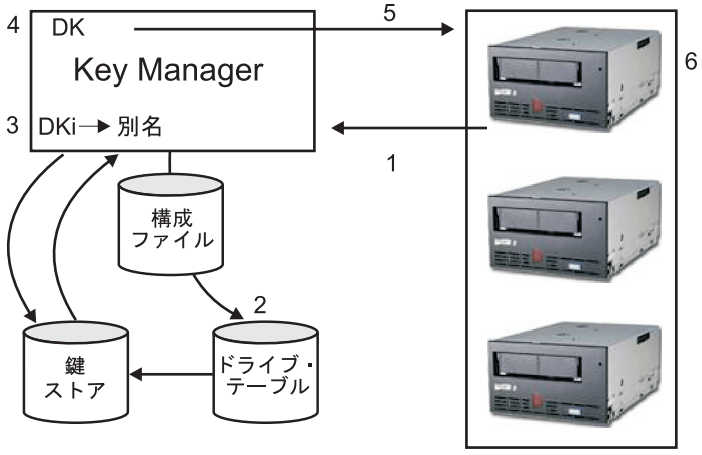

図 2-2. 暗号化読み取り操作に向けた、LTO 4 または LTO 5 テープ・ドライブ要求

1. テープ・ドライブは読み取り要求を受信し、DKi を Encryption Key Manager に送信します
2. Encryption Key Manager が、ドライブ・テーブル内のテープ・デバイスを検査します
3. Encryption Key Manager は、DKi を別名に変換し、対応する DK を鍵ストアから取り出します
4. Encryption Key Manager は、ドライブが暗号化解除できる鍵を使用して DK をラップします
5. Encryption Key Manager は、ラップされたその DK をテープ・ドライブに送信します
6. テープ・ドライブが、DK をアンラップし、それを使用してデータを暗号化解除します

鍵ストア・データのバックアップ

注: 鍵ストア内の鍵が持っている本質的な重要性のため、非暗号化デバイスで鍵ストア・データのバックアップをとり、必要に応じてこのデータをリカバリーし、そのテープ・ドライブまたはライブラリーに関連した証明書を使用して暗号化されたテープを読み取れるようにすることが重要です。鍵ストアが適切にバックアップされないと、暗号化されたデータへのすべてのアクセスを失って取り返しのつかないことになります。

この鍵ストア情報をバックアップする方法は、各種あります。各鍵ストア・タイプは、それぞれ固有の特性を持っています。で詳しく説明していますが、以下の一般ガイドラインはすべてに当てはまります。

- 鍵ストアにロードされた証明書のすべてのコピーを保持します (通常、PKCS12 フォーマット・ファイル)。

- システム・バックアップ機能 (RACF など) を使用して、鍵ストア情報のバックアップ・コピーを作成します (リカバリーのためにこのコピーを暗号化解除することはできないため、暗号化テープ・ドライブを使用してこのコピーを暗号化しないように注意してください)。
- 1 次および 2 次 Encryption Key Manager と鍵ストア・コピーを維持します (フェイルオーバーの予備用だけでなく、バックアップ用)。冗長性を高めるため、1 次および 2 次の両方に対して鍵ストアをバックアップしてください。
- JCEKS 鍵ストアの場合は、鍵ストア・ファイルを単純にコピーし、平文の (暗号化されていない) コピーをボールトのような安全な場所に保管します (リカバリーのためにこのコピーを暗号化解除することはできないため、暗号化テープ・ドライブを使用してこのコピーを暗号化しないように注意してください)。

最低でも、鍵ストア・データを変更したときには必ず鍵ストア・データをバックアップしてください。Encryption Key Manager は鍵ストア・データを変更しません。鍵ストアに対する変更は、適用された変更のみであるため、鍵ストアを変更したら、できる限り早く鍵ストアをコピーしてください。

GUI を使用したファイルのバックアップ

1. GUI がまだ開始されていない場合は、次のように GUI を開きます。

   Windows 上の場合

   c:\ekm\gui にナビゲートして、**LaunchEKMGui.bat** をクリックします。

   Linux プラットフォーム上の場合

   /var/ekm/gui にナビゲートして、`. ./LaunchEKMGui.sh`と入力します。

2. Encryption Key Manager GUI 左側のナビゲーターの「**Backup Critical Files (重要なファイルのバックアップ)**」を選択します。
3. 表示されるダイアログでバックアップ・データのパスを入力します (2-8 ページの図 2-3)。

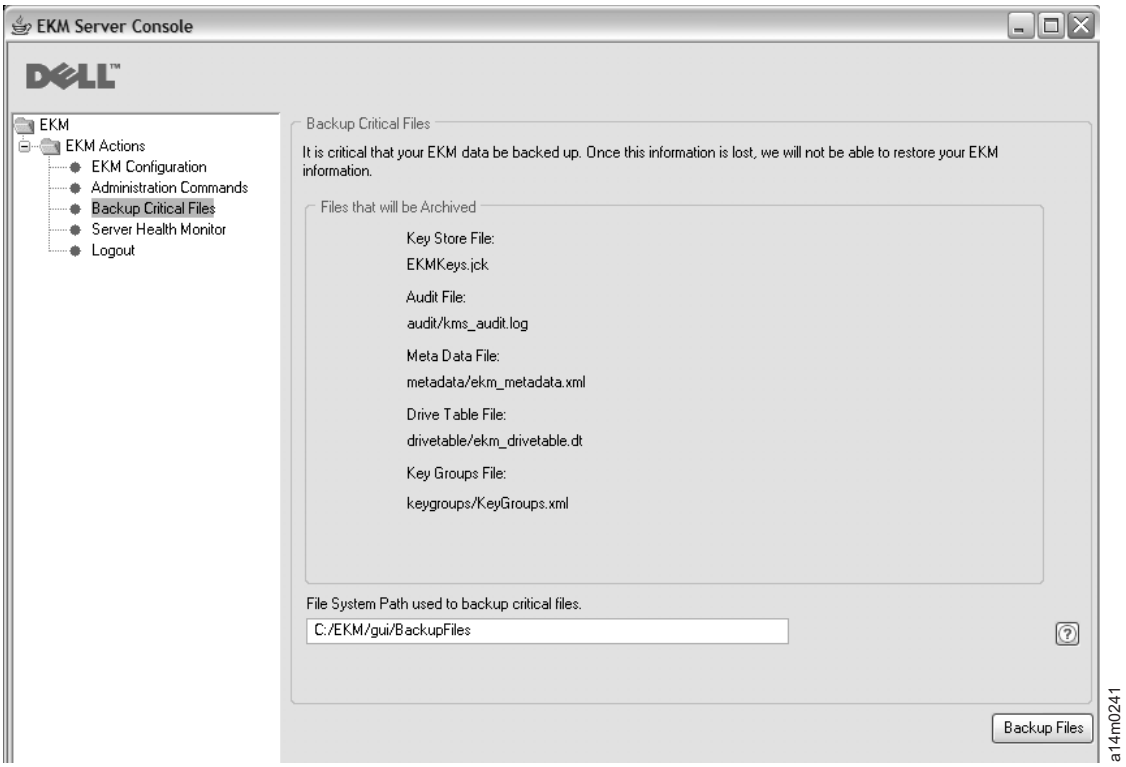

図 2-3. 「重要なファイルのバックアップ (Backup Critical Files)」ウィンドウ

4. 「**Backup Files (ファイルのバックアップ)**」をクリックします。
5. 情報メッセージにより、結果が表示されます。

冗長性を確保するための複数の Key Manager

Encryption Key Manager は、冗長性、ひいては高い可用性を可能にするためにテープ・ドライブおよびライブラリーで機能するように設計されているため、複数の Key Manager が同じテープ・ドライブおよびライブラリーにサービスすることができます。また、これらの Key Manager は、テープ・ドライブおよびライブラリーと同じシステム上に存在する必要はありません。Key Manager の最大数はライブラリーまたはプロキシーによって異なります。 必要なことは、それらの EKM が、TCP/IP 接続を介してテープ・ドライブで使用できることだけです。

これにより、1 つの Key Manager が使用不能になった場合に、フェイルオーバーのほか、鍵ストアに関する重要な情報の組み込みバックアップを備えた、互いのミラー・イメージである 2 つの Encryption Key Manager を持つことができます。デバイス (またはプロキシー) を構成するときに、2 つの Key Manager を指すことができます。どのような理由であっても、1 つの Key Manager が使用不能になると、ご使用のデバイス (またはライブラリー) は、単純に代替の Key Manager を使用します。

2 つの Encryption Key Manager を同期させておくこともできます。重要データの本来のバックアップと、テープ操作での障害を防止するためのフェイルオーバー機能の両方のために、必要な場合にこの重要な機能を利用することは重大な意味を持ちます。 4-2 ページの『2 つの Key Manager サーバー間でのデータの同期化』を参照してください。

注: 同期化には、鍵ストアは含まれません。これらを手動でコピーする必要があります。

Encryption Key Manager サーバー構成

Encryption Key Manager は、単一のサーバー上にも、あるいは複数のサーバー上にもインストールできます。次の例は、Key Manager が 1 つの構成と 2 つの構成を示していますが、ライブラリーはそれよりも多く構成できます。

シングル・サーバー構成

図 2-4 に示されているシングル・サーバー構成は、最も単純な Encryption Key Manager 構成です。ただし、冗長性がないため、お勧めしません。この構成では、すべてのテープ・ドライブが、バックアップなしで、単一の Key Manager サーバーに依存します。サーバーが故障した場合、鍵ストア、構成ファイル、KeyGroups.xml ファイル、およびドライブ・テーブルは使用不能になり、暗号化されたテープは読み取れなくなります。シングル・サーバーの構成では、サーバーのコピーが失われた場合に EKM 機能を代替サーバーで再作成できるように、Encryption Key Manager とは別個に、鍵ストア、構成ファイル、KeyGroups.xml ファイル、およびドライブ・テーブルのバックアップ・コピーが安全な場所に確実に保持されるようにする必要があります。

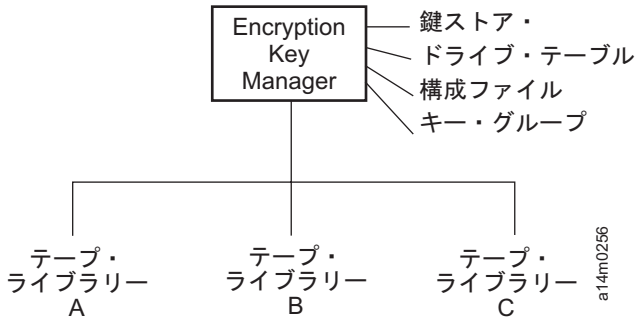

図 2-4. シングル・サーバー構成

Two-Server Configurations (2 サーバー構成)

2 サーバー構成をお勧めします。どのような理由であっても 1 次 Key Manager がアクセス不能になった場合、この Encryption Key Manager 構成は自動的に 2 次 Key Manager にフェイルオーバーします。

注: 同じセットのテープ・ドライブからの要求を処理するのに、異なる Encryption Key Manager サーバーが使用される場合、関連した鍵ストアの情報は、同じでなければなりません。これは、連絡を取る Key Manager に関係なく、テープ・ドライブからの要求をサポートするのに必要な情報を使用できるようにするために必要です。

Identical configurations (同一の構成): 2-10 ページの図 2-5 に示されているような、2 つの Encryption Key Manager サーバーが同じ構成を持つ環境では、1 次 Key Manager が故障した場合、処理は、自動的に 2 次 Key Manager にフェイルオーバーします。そのような構成では、2 つの Key Manager サーバーを同期する必要がありま

す。1 つの Key Manager サーバーの構成ファイルおよびドライブ・テーブルに対する更新は、**sync** コマンドを使用して、もう一方の Key Manager サーバー上で自動的に複写できますが、1 つの鍵ストアに対する更新は、使用する鍵ストアに固有の方式を使用して、もう一方にコピーする必要があります。鍵ストアおよびキー・グループ XML ファイルは、手動でコピーする必要があります。 詳しくは、 4-2 ページの『2 つの Key Manager サーバー間でのデータの同期化』を参照してください。

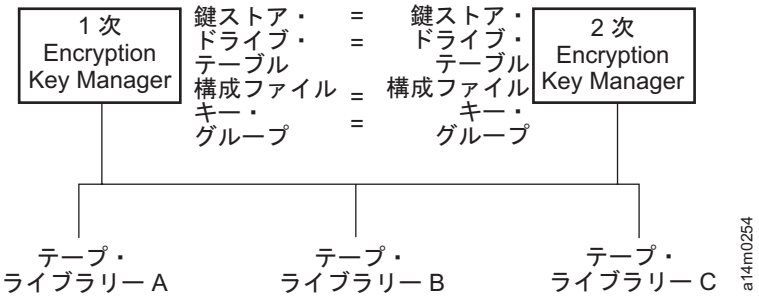

図 2-5. 共用構成を持つ 2 つのサーバー

異なる構成: 2 つの Encryption Key Manager サーバーが、1 つの共通鍵ストアとドライブ・テーブルを共有する一方で、2 つの異なる構成ファイルと 2 つの異なるキー・グループのセットをこれらの XML ファイルに定義することができます。この場合に唯一必要となる条件は、共通テープ・ドライブにサービスするのに使用される鍵がそれぞれのサーバーに対して同じでなければならいということです。 これにより、各 Key Manager サーバーは、固有のプロパティーのセットを持つことができます。このタイプの構成では、図 2-6 に示されているとおり、Key Manager サーバー間でドライブ・テーブルのみを同期する必要があります。(詳しくは、 4-2 ページの『2 つの Key Manager サーバー間でのデータの同期化』を参照してください。)構成ファイルの上書き防止のために必ず、`sync.type = drivetab` を指定してください (config または all は指定しないでください)。

注: サーバー間で構成を部分的に共用する手段はありません。

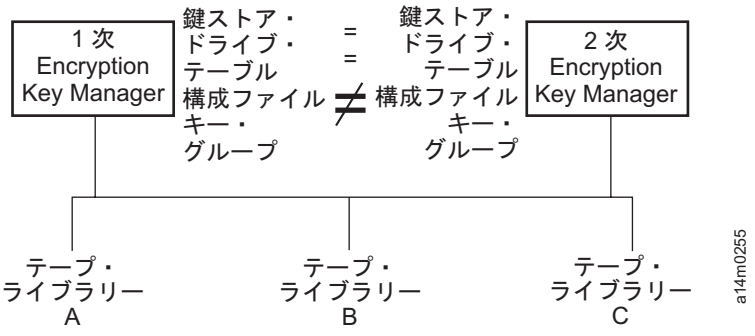

図 2-6. 同じデバイスにアクセスする、異なる構成を持つ 2 つのサーバー

災害時回復サイトについての考慮事項

災害時回復 (DR) サイトの利用を計画している場合、Encryption Key Manager は、そのサイトで暗号化されたテープの読み取りおよび書き込みが可能になるように多くのオプションを提供します。以下のものです。

- DR サイトで複製 Encryption Key Manager を作成する。

  ご使用のローカル Encryption Key Manager と同じ情報 (構成ファイル、テープ・ドライブ・テーブル、キー・グループ XML ファイル、および鍵ストア) を使用して、DR サイトで複製 Encryption Key Manager をセットアップします。そうすると、この Key Manager は、既存の実動 Key Manager の 1 つに取って代わり、暗号化されたテープの読み書きを引き継ぐことができます。
- 必要に応じて回復できるように 3 つの Encryption Key Manager データ・ファイルのバックアップ・コピーを作成する。

  Encryption Key Manager が必要とする 4 つのデータ・エレメント (構成ファイル、テープ・ドライブ・テーブル、キー・グループ XML ファイル、および鍵ストア) の現行のコピーを作成すると、DR サイトで複製として機能するように、いつでも Key Manager を始動することができます (機能する Key Manager なしではこのデータを暗号化解除できないため、Encryption Key Manager を使用してこれらのファイルのコピーを暗号化しないでください)。ご使用の DR サイトが、1 次サイトからのさまざまなテープ・ドライブを使用している場合は、構成ファイルとテープ・ドライブ・テーブルには、その DR サイトに関する正確な情報が含まれていなければなりません。

暗号化されたテープをオフサイトで共用するための考慮事項

注: ビジネス・パートナーから受け取った証明書に最終的に署名した認証局 (CA) にその証明書を戻すという信頼の連鎖を検査することによって証明書の妥当性を検証することが重要です。CA を信頼する場合は、その証明書を信頼できます。あるいは、証明書が転送中に安全に保護されていた場合、その証明書の妥当性は検証できます。これらの方法のいずれかによって証明書の妥当性を検証できないと、「中間者」攻撃が可能になります。

LTO 4 および LTO 5 テープの共用

LTO 4 または LTO 5 テープ上の暗号化データを共用するには、相手方の組織がテープを読み取れるように、そのテープ上のデータを暗号化するのに使用された対称鍵のコピーを相手方の組織で使用できるようにする必要があります。対称鍵を共用するために、相手方の組織はその公開鍵をお客様と共用する必要があります。この公開鍵は、keytool を使用して対称鍵が Encryption Key Manager 鍵ストアからエクスポートされる際に、その対称鍵をラップするのに使用されます (3-15 ページの『Keytool -exportseckey を使用したデータ鍵のエクスポート』を参照)。相手方の組織が対称鍵を自らの Encryption Key Manager 鍵ストアにインポートすると、対称鍵は、対応する秘密鍵を使用してアンラップされます (3-15 ページの『Keytool -importseckey を使用したデータ鍵のインポート』を参照)。これにより、秘密鍵の保有者のみが対称鍵をアンラップできるため、対称鍵を安全に転送することができます。相手方の組織は、Encryption Key Manager 鍵ストア内のデータを暗号化するのに使用された対称鍵を使って、テープ上のデータを読み取ることができます。

Federal Information Processing Standard (連邦情報処理標準) 140-2 に関する考慮事項

Federal Information Processing Standard (連邦情報処理標準) 140-2 は、連邦政府がすべての暗号提供者が FIPS 140 認定であることを求めているため、重要なものとなりました。この標準は、成長著しいプライベート・セクター・コミュニティーでも採用されました。政府標準に準拠したサード・パーティーによる暗号機能の認定は、このセキュリティーを重視する世界において価値が増大してきたように思われます。

Encryption Key Manager 自体は暗号機能を提供しないため、FIPS 140-2 認証を必要とせず、取得することもできません。しかし、Encryption Key Manager は、IBM Java Cryptographic Extension コンポーネント内の IBM JVM の暗号機能を利用して、FIPS 140-2 レベル 1 認定を受けている IBMJCEFIPS 暗号提供者の選択および使用を許可します。構成プロパティー・ファイルで **fips** 構成パラメーターを**オン**に設定することによって、Encryption Key Manager は、すべての暗号機能に対して IBMJCEFIPS 提供者を使用できるようになります。

特定のハードウェアおよびソフトウェア暗号製品が FIPS 140-2 認定を受けているかどうかについては、それぞれの提供者からの資料を参照してください。

第 3 章 Encryption Key Manager および鍵ストアのインストール

Encryption Key Manager は、IBM Java Virtual Machine インストール・システムと一緒に出荷されます。EKM には、IBM Software Developer Kit for Linux および IBM Runtime Environment for Windows が必要です (2-2 ページの『ハードウェアおよびソフトウェアの要件』を参照)。 ご使用のオペレーティング・システムに該当する手順に従ってください。

- 3-2 ページの『Linux 上での Encryption Key Manager のインストール』
- 3-3 ページの『Windows 上での Encryption Key Manager のインストール』

Encryption Key Manager のバージョンが最新であるかどうか不明である場合は、『最新のバージョンの Key Manager ISO イメージのダウンロード』を参照して新しいバージョンが入手可能であるかどうかを判別してください。Java インストール・システムに含まれていない最新バージョンの Encryption Key Manager を入手することをお勧めします。 詳細については、**http://support.dell.com** を参照してください。

Encryption Key Manager ホスト・サーバーの重要な構成情報: データ損失のリスクを最小限にとどめるには、Dell Encryption Key Manager プログラムをホスティングするマシンが、ECC メモリーを使用することを推奨します。Encryption Key Manager は、暗号鍵の生成を要求する機能、およびその鍵を LTO 4 および LTO 5 テープ・ドライブに引き渡す機能を実行します。鍵の構成要素は、Encryption Key Manager による処理時中は、ラップされた形 (暗号化された形式) でシステム・メモリーに常駐します。鍵の構成要素は、カートリッジに書き込まれるデータがリカバリー (暗号化解除) できるように、エラーなしで適切なテープ・ドライブに転送される必要があります。システム・メモリー内のビット・エラーが発生した結果、何らかの理由で鍵の構成要素が破損しており、かつ、その鍵の構成要素をカートリッジへのデータ書き込みに使用する場合、そのカートリッジに書き込まれるデータはリカバリーすること (つまり、後日暗号化解除すること) ができません。このようなデータ・エラーの発生を確実に防ぐために配置されている安全機能があります。ただし、Encryption Key Manager をホスティングするマシンでエラー訂正コード (ECC) メモリーが使用されない場合は、システム・メモリー内にある間に鍵の構成要素が破損し、この破損によりデータ損失が発生する可能性が残されます。この状況が発生する可能性は少ないですが、重要なアプリケーション (Encryption Key Manager など) をホスティングするマシンでは、ECC メモリーを使用することを常に推奨します。

最新のバージョンの Key Manager ISO イメージのダウンロード

最新バージョンの Dell ISO イメージをダウンロードするには、http://support.dell.com にアクセスしてください。

Linux 上での Encryption Key Manager のインストール

Linux 上でのEncryption Key Manager の CD からのインストール

1. Dell Encryption Key Manager CD を挿入し、CD のルート・ディレクトリーから `Install_Linux` と入力します。

   インストールにより、CD からすべてのコンテンツ (文書、GUI ファイル、および構成プロパティー・ファイル) をご使用のオペレーティング・システムの適切なハード・ディスクにコピーします。インストール中に、システムにより、正式な IBM Java ランタイム環境がないか確認されます。見つからない場合は、自動的にインストールされます。

   インストールが完了すると、グラフィカル・ユーザー・インターフェース (GUI) が起動します。

Linux 上ので Software Developer Kit の手動によるインストール

CD からインストールしない場合は、次のステップを実行します。

1. **http://support.dell.com** から、ご使用のオペレーティング・システムに基づいて、適切な Runtime Environment for Java をダウンロードします。
   - Java 6 SR 5 (32 ビット) 以降
   - Java 6 SR 5 (64 ビット) 以降
2. Java linux rpm ファイルを作業ディレクトリーに置きます。

   ```
   mordor:~ #/tape/Encryption/java/1.6.0# pwd
   /tape/Encryption/java/1.6.0
   mordor:~ #/tape/Encryption/java/1.6.0# ls
   ibm-java-i386-jre-6.0-5.0.i386.rpm
   ```

3. rpm パッケージをインストールします。

   ```
   mordor:~ #rpm -ivh -nodeps ibm-java-i386-jre-6.0-5.0.i386.rpm
   ```

   これによって、ファイルが、**/opt/ibm/java-i386-60/** dir に置かれます。

   ```
   mordor:~ #/opt/ibm/java-i386-60/jre # ls
   .systemPrefs  bin  javaws  lib
   ```

4. ファイル **/etc/profile.local** を、JAVA_HOME、CLASSPATH、およびユーザーがインストールした Java の bin dir で編集 (または必要に応じて作成) します。以下の 3 つの行を追加します。

   ```
   JAVA_HOME=/opt/ibm/java-i386-60/jre
   CLASSPATH=/opt/ibm/java-i386-60/jre/lib
   PATH=$JAVA_HOME:opt/ibm/java-i386-60/jre/bin/:$PATH
   ```

5. ログアウトしてから、ホストにもう一度ログインして **/etc/profile.local** 項目を有効にするか、あるいは、エクスポート・コマンド行コマンドを実行します。

   ```
   mordor:~ # export JAVA_HOME=/opt/ibm/java-i386-60/jre
   mordor:~ # export CLASSPATH=/opt/ibm/java-i386-60/jre/lib
   mordor:~ # export PATH=/opt/ibm/java-i386-60/jre/bin/:$PATH
   ```

6. もう一度ログイン後、**java -version** コマンドを発行します。次のような結果が示されるはずです。

   ```
   mordor:~ # java -version
   java version "1.6.0"
   Java(TM) SE Runtime Environment (build pmz60sr5-20090529(SR5))
   ```

```
IBM J9 VM (build 2.4, J2RE 1.6.0 IBM J9 2.4 Linux x86-32 jvmxi3260-20090519_35743 (JIT enabled)
...
mordor:~ # which java
/opt/ibm/java-i386-60/jre/bin/java
```

Windows 上での Encryption Key Manager のインストール

1. Dell Encryption Key Manager CD を挿入します。

   インストールにより、CD からすべてのコンテンツ (文書、GUI ファイル、および構成プロパティー・ファイル) をご使用のオペレーティング・システムの適切なハード・ディスクにコピーします。インストール中に、システムにより、正式な IBM Java ランタイム環境がないか確認されます。見つからない場合は、自動的にインストールされます。

   インストールが完了すると、グラフィカル・ユーザー・インターフェース (GUI) が起動します。

2. InstallShield Wizard が開いたら、「**Next (次へ)**」をクリックします。
3. 「License Agreement (ご使用条件)」を読んで、「**Yes (はい)**」をクリックします。
4. 「Choose Destination Location (宛先ロケーションの選択)」ウィンドウ (3-4 ページの図 3-1) が開いたら、フォルダーを選択して、それをメモします。Encryption Key Manager を起動するのに、この Java パスが必要になります。

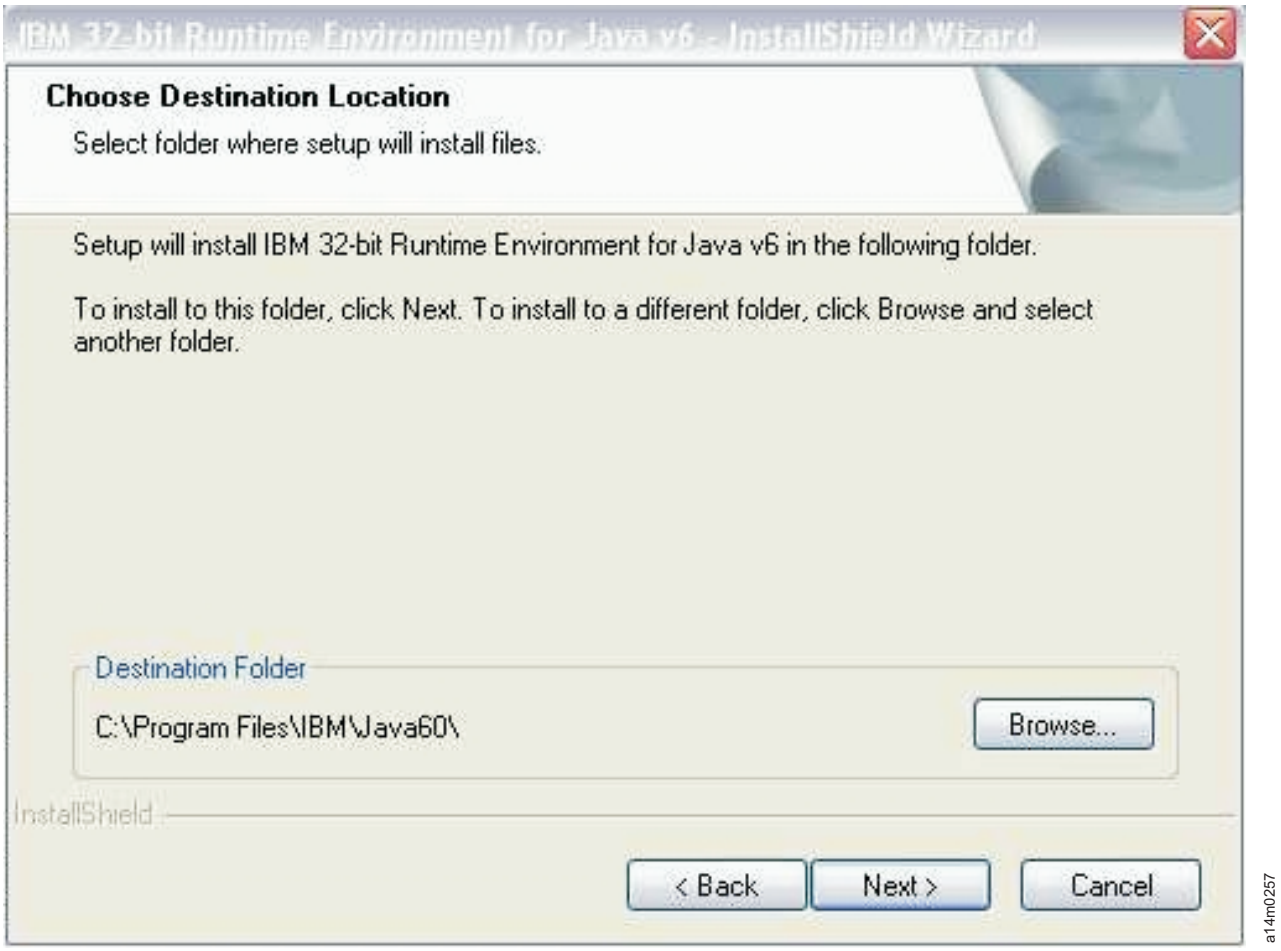

図 3-1. 「Choose Destination Location (宛先ロケーションの選択)」ウィンドウ

「**Next (次へ)**」をクリックします。

5. ウィンドウが開き、この Java Runtime Environment をデフォルトのシステム JVM とするかどうかを尋ねてきます (図 3-2)。

図 3-2. このバージョンの JVM をデフォルトに設定する

「**No**」をクリックします。

6. 「Start Copying Files (ファイルのコピー開始)」ウィンドウが開きます (3-5 ページの図 3-3)。必ず、ターゲット・ディレクトリーをメモしてください。

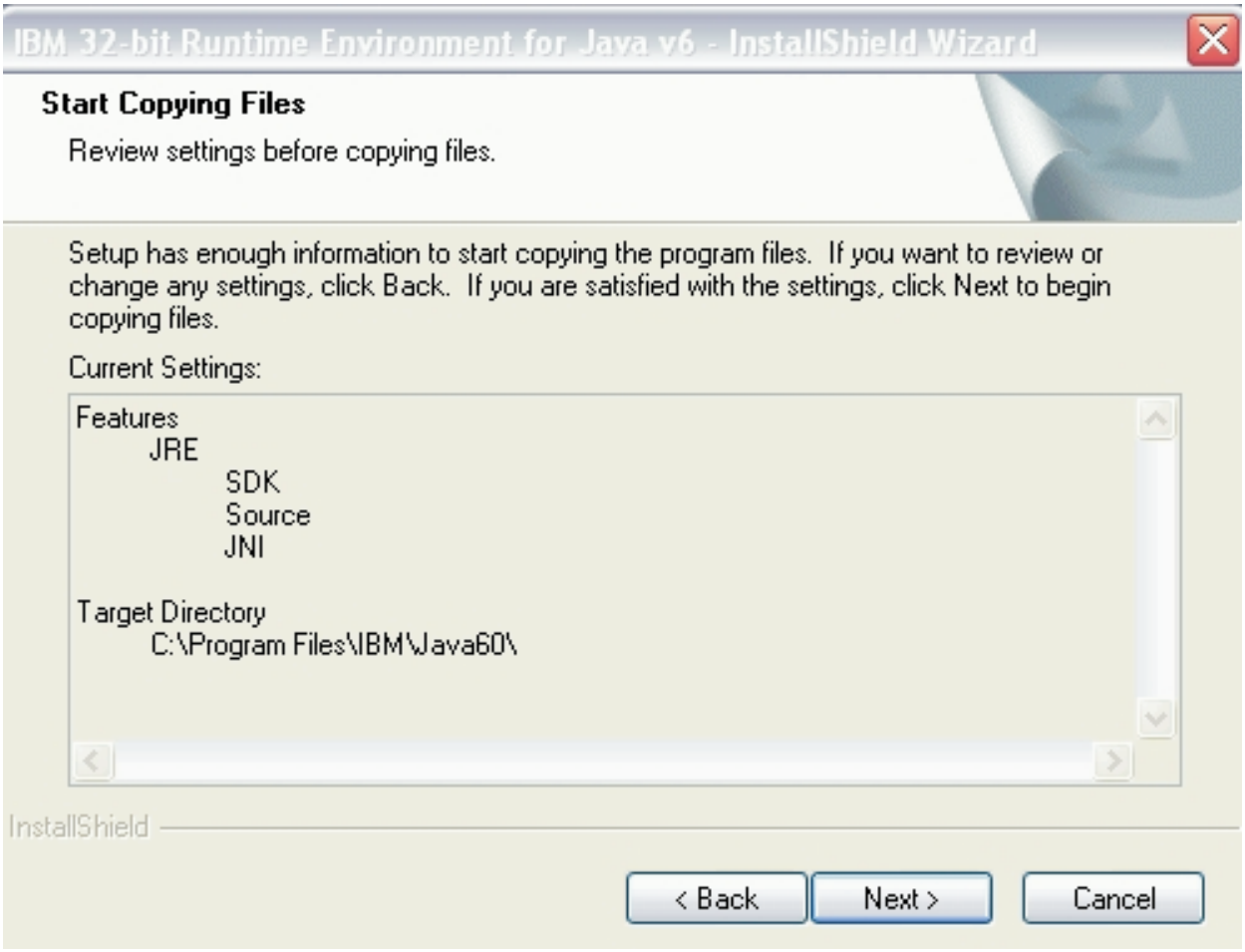

図 3-3. 「Start Copying Files (ファイルのコピー開始)」ウィンドウ

「**Next (次へ)**」をクリックします。

7. 状況ウィンドウに、インストールの進捗状況が示されます。
8. 「Browser Registration (ブラウザーの登録)」ウィンドウが開きます。Encryption Key Manager で使用するブラウザーを選択します。「**Next (次へ)**」をクリックします。
9. InstallShield Wizard が開いたら、「**Finish (終了)**」をクリックします。

   インストール後、コマンド・プロンプトを開いて、次のように、インストールされた Javaバージョンを照会できます。

```
C:¥WinEKM>C:¥"Program Files"¥IBM¥Java60¥jre¥bin¥java -version
java version "1.6.0"
Java(TM) SE Runtime Environment (build pwi3260sr5-20090529_04(SR5))
IBM J9 VM (build 2.4, J2RE 1.6.0 IBM J9 2.4 Windows Server 2003 x86-32 j9vmwi3223-20090
519_35743 (JIT enabled, AOT enabled)
...
```

10. PATH 変数を次のように更新します (Encryption Key Manager 2.1 の場合は必須ですが、ビルド日付が 05032007 以前の場合はオプションです)。

    コマンド・ウィンドウから java SDK を呼び出す場合に、コマンドの絶対パスを入力するのではなく、ディレクトリーから Java JRE 実行可能ファイル (java.exe) を実行できるようにしたい場合は、PATH 変数を設定することになり

ます。PATH 変数を設定しない場合は、実行可能ファイルを実行するたびに、そのファイルへの次のような絶対パスを指定しなければなりません。

```
C:>¥Program Files¥IBM¥Java60¥jre¥bin¥java ...
```

PATH を永続的に設定するには (Encryption Key Manager 2.1 の場合は必須)、java bin ディレクトリーの絶対パスを PATH 変数に追加します。一般に、この絶対パスは次のようなものです。

```
C:¥Program Files¥IBM¥Java60¥jre¥bin
```

Microsoft Windows 2003、2008、および2008 R2 で PATH を永続的に設定するには次のようにします。

注: PATH 変数は、コマンド行から設定しても機能しません。

a. 「スタート」メニューから「**設定**」を選択し、「**コントロール パネル**」を選択します。
b. 「**システム**」をダブルクリックします。
c. 「**詳細設定**」タブをクリックします。
d. 「**環境変数**」をクリックします。
e. 「システム環境変数」のリストをスクロールして PATH 変数を見つけ、「**編集**」をクリックします。
f. PATH 変数の先頭に IBM JVM パスを追加します。

   デフォルトのインストール・ディレクトリーは、C:¥PROGRA~1¥IBM¥Java60¥jre¥bin です。

   重要: 追加するパスの終わりにセミコロンを挿入し、パス・リスト内の他のディレクトリーと区別するようにしてください。
g. 「**OK**」をクリックします。

GUI を使用した構成ファイル、鍵ストア、および証明書の作成

Encryption Key Manager を起動する前に、新しい鍵ストアと自己署名証明書を少なくとも 1 つずつ作成する必要があります。 Dell Encryption Key Manager サーバー・グラフィカル・ユーザー・インターフェース (GUI) を使用して、Encryption Key Manager 構成プロパティー・ファイル、鍵ストア、証明書、および鍵を作成することができます。このプロセスの結果として、簡単な CLI 構成プロパティー・ファイルも作成されます。

1. GUI がまだ開始されていない場合は、次のように GUI を開きます。

   Windows 上の場合
   c:\ekm\gui にナビゲートして、**LaunchEKMGui.bat** をクリックします。

   Linux プラットフォーム上の場合
   /var/ekm/gui にナビゲートして、`. ./LaunchEKMGui.sh` と入力します。
2. GUI 左側のナビゲーターの「**EKM Configuration (EKM 構成)**」を選択します。

3. 「EKM Server Configuration (EKM サーバー構成)」ページ (図 3-4) で、アスタリスク * で示された必須フィールドすべてにデータを入力します。その他のフィールドは、必要に応じて入力してください。 説明を表示する場合は、データ・フィールド右側の疑問符 (?) をクリックします。 「**Next (次へ)**」をクリックします。

注: 鍵ストア・パスワードを設定したら、セキュリティーが破られない限り、そのパスワードを**変更しないでください**。パスワードは機密漏れを排除するために暗号化されます。鍵ストア・パスワードを変更すると、**keytool** コマンドを使用してその鍵ストア内のすべてのパスワードを個別に変更する必要があります。 3-14 ページの『鍵ストア・パスワードの変更』を参照してください。

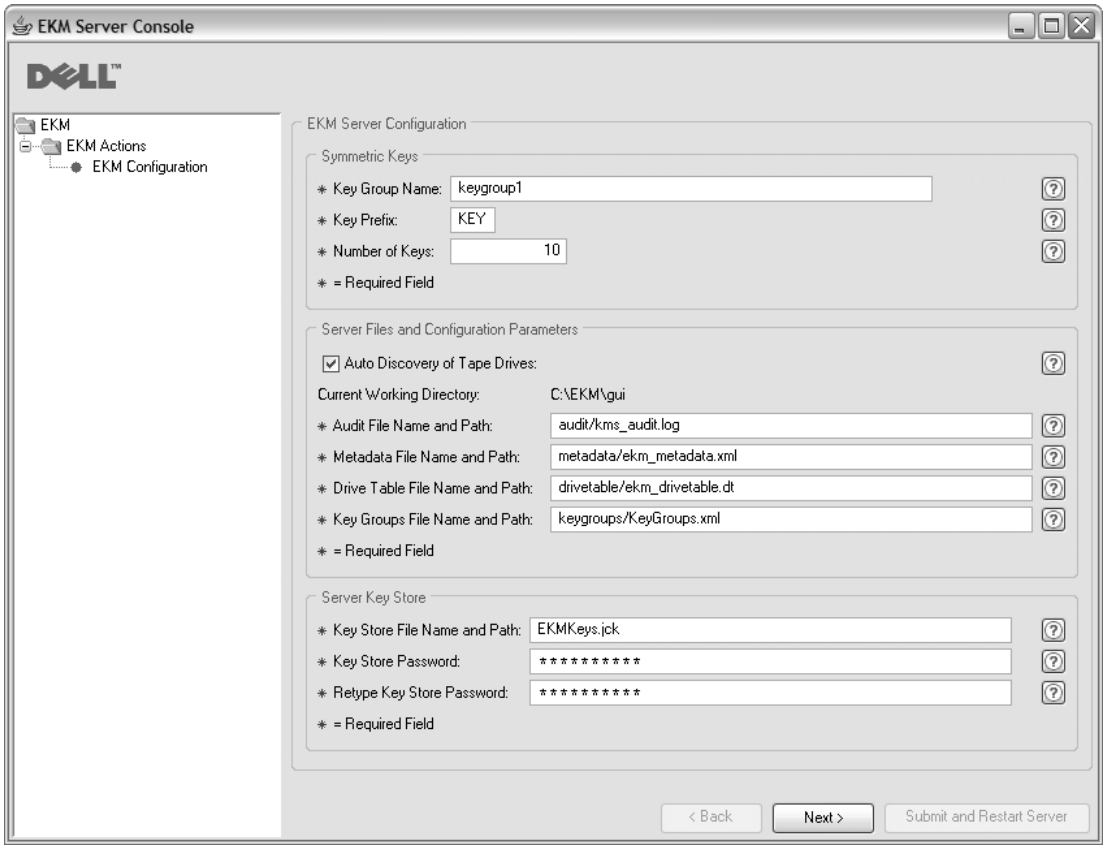

図 3-4. EKM Server Configuration (EKM サーバー構成) ページ

Dell Encryption Key Manager 鍵ストアに対して生成可能な鍵の数には制限がありませんが、鍵を生成するのに必要な時間が要求される鍵の数に応じて増加します。 Encryption Key Manager では、10 個の鍵を生成するのに 15 秒かかり、10000 個の鍵を生成するには 30 分以上を要します。鍵の数は、ホスト・サーバー・リソース (サーバー内のメモリー) によって制限されることに留意してください。Encryption Key Manager アプリケーションは、実行中は鍵ストアのリストをシステム・メモリー内に保持します。これは、ライブラリーが鍵要求をドライブから送信する際に、鍵へのアクセスを迅速にするためです。

注: 鍵生成中に Encryption Key Manager GUI を中断した場合、Encryption Key Manager の再インストールが必要になります。

Encryption Key Manager の鍵生成プロセスが完了する前に停止された場合、鍵ストア・ファイルが破損します。このイベントからリカバリーするには、以下のステップに従ってください。

- Encryption Key Manager の初期インストール中に Encryption Key Manager が中断された場合は、Encryption Key Manager ディレクトリー (例えば、x:\ekm) があるディレクトリーにナビゲートします。そのディレクトリーを削除し、インストールを再開します。
- 新規のキー・グループを追加中に Encryption Key Manager が中断された場合は、Encryption Key Manager サーバーを停止し、最新のバックアップ鍵ストア (このファイルは、x:\ekm\gui\backupfiles フォルダー内にあります) を使用して、鍵ストア・ファイルを復元してください。バックアップ・ファイルには、ファイル名の一部として日時スタンプが含まれていることに注意してください (例えば、2007_11_19_16_38_31_EKMKeys.jck)。この日時スタンプは、ファイルを x:\ekm\gui ディレクトリーにコピーしたら、削除する必要があります。Encryption Key Manager サーバーを再始動して、以前に中断されたキー・グループを追加します。

4. 「EKM Server Certificate Configuration (EKM サーバーの証明書の構成)」ページ (3-9 ページの図 3-5) で、鍵ストア別名および必要な追加データを入力します「**Submit and Restart Server (サブミットしてサーバーを再始動)**」をクリックします。

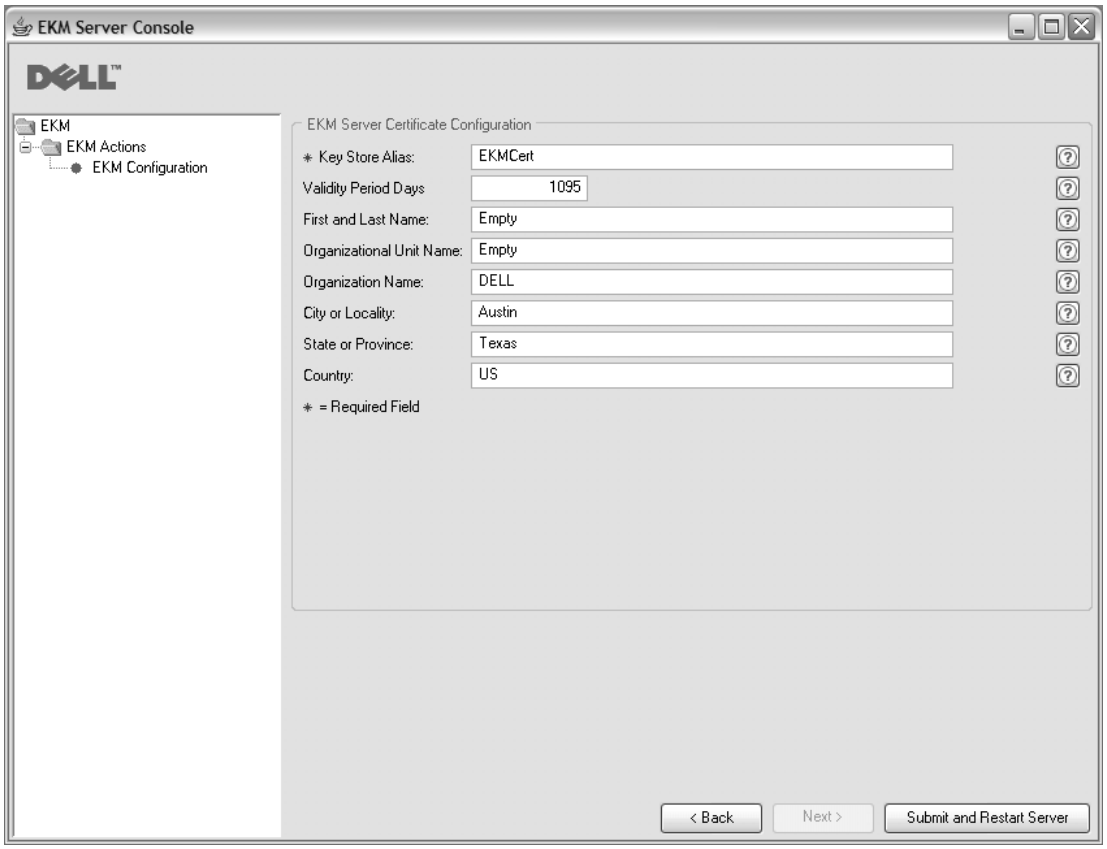

図 3-5. EKM Server Certificate Configuration (EKM サーバーの証明書の構成) ページ

5. 「Backup Critical Files (重要なファイルのバックアップ)」ウィンドウ (3-10 ページの図 3-6) が開き、Encryption Key Manager データ・ファイルをバックアップするよう促します。

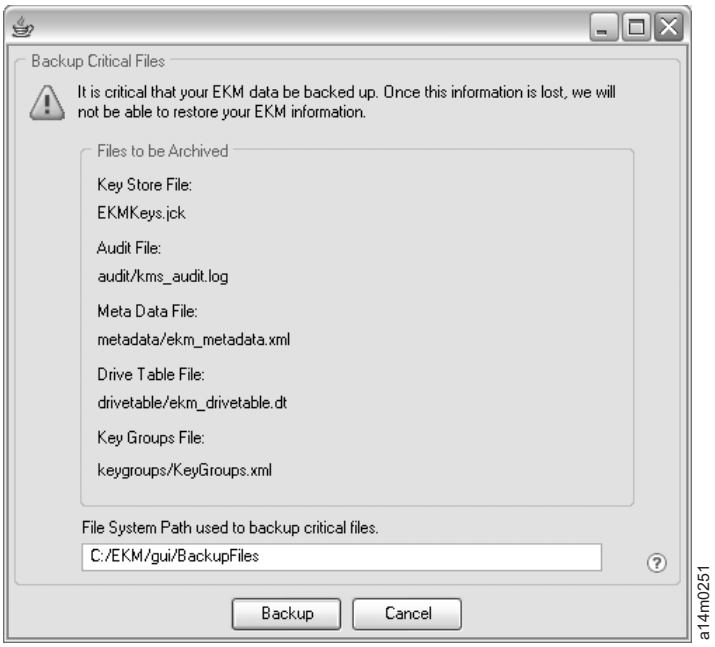

図 3-6. 「重要なファイルのバックアップ (Backup Critical Files)」ウィンドウ

パスを確認して、「**Backup (バックアップ)**」をクリックします。 Dell Encryption Key Manager サーバーがバックグラウンドで起動します。

Encryption Key Manager は、Encryption Key Manager サーバー構成を変更して「**OK**」をクリックするたびに、または「Backup Critical Files (重要なファイルのバックアップ)」ウィンドウで「**Backup (バックアップ)**」をクリックするたびに、バックアップ・ファイルのセットを生成します。「*Files to be Archived (アーカイブするファイル)*」としてリストされているファイルは、c:/ekm/gui/BackupFlies ディレクトリーに保存されます。各ファイル名には、先頭に日付と時刻が付いています。例えば、2007 年の 11 月 26 日の午後 2 時 58 分 46 秒にバックアップされたファイルのセットには、すべて名前の先頭に「2007_11_26_14_58_46_FileName」という日付と時刻のスタンプが付いています。バックアップ・ファイルは上書きされません。

6. GUI ナビゲーターで「**Server Health Monitor (サーバーの正常性モニター)**」を選択し、Encryption Key Manager サーバーが起動していることを確認します。

既存の鍵ストアに鍵を追加する場合は、3-17 ページの『GUI を使用したキー・グループの定義および鍵の作成』を参照してください。

適切なホスト IP アドレスを見つける方法

現在の Encryption Key Manager GUI における制限により、Encryption Key Manager のホスト IP アドレスが「Server Health Monitor (サーバーの正常性モニター)」に表示されない場合があります。

- ホストが IPv6 アドレスで構成されている場合、Encryption Key Manager アプリケーションは IP アドレスを表示できません。
- Encryption Key Manager アプリケーションが Linux システムにインストールされている場合、Encryption Key Manager アプリケーションは実際にアクティブである IP ポートではなく、ローカル・ホスト・アドレスを表示します。

1. ホスト・システムの実際の IP アドレスを取得するには、ネットワーク構成にアクセスし、IP ポート・アドレスを見つけてください。
   - Windows システムでは、コマンド・ウィンドウを開き、`ipconfig` と入力します。
   - Linux の場合は、`isconfig` と入力します。

EKM SSL ポートを識別する方法

1. コマンド行を使用して、Encryption Key Manager サーバーを始動します。
   - Windows では、cd `c:\ekm` にナビゲートし、**startServer.bat** をクリックします。
   - Linux プラットフォームでは、`/var/ekm` にナビゲートし、`startServer.sh` と入力します。
   - 詳しくは、 5-1 ページの『Key Manager サーバーの始動、リフレッシュ、および停止』を参照してください。
2. コマンド行を使用して、CLI クライアントを開始します。
   - Windows では、cd `c:\ekm` にナビゲートし、**startClient.bat** をクリックします。
   - Linux プラットフォームでは、`/var/ekm` にナビゲートし、`startClient.sh` と入力します。
   - 詳しくは、 5-6 ページの『コマンド行インターフェース・クライアント』を参照してください。
3. 以下のコマンドを使用して、Encryption Key Manager サーバー上の CLI クライアントにログインします。

   ```
   login –ekmuser userID –ekmpassword password
   ```

   ここで、*userID* = EKMAdmin であり、*password* = changeME (これがデフォルトのパスワードです。既にデフォルトのパスワードを変更している場合は、ご使用の新規パスワードを使用してください) です。

   ログインが成功すると、`User successfully logged in` (ユーザーのログインが成功しました) と表示されます。
4. 次のコマンドを入力し、SSL ポートを識別します。

   ```
   status
   ```

   以下のような応答が表示されます。`server is running. TCP port: 3801, SSL port: 443` (サーバーは稼動中です。TCP ポート: 3801、SSL ポート: 443)

   構成済みの SSL ポートをメモし、これがライブラリー管理の暗号化設定を構成するのに使用するポートであることを確認してください。
5. コマンド行からログアウトします。次のコマンドを入力してください。

   ```
   exit
   ```

   コマンド・ウィンドウを閉じます。

LTO 4 および LTO 5 上での暗号化のための鍵と別名の生成

対称暗号鍵を生成する場合、最も簡単な方法は Dell Encryption Key Manager サーバー GUIを使用することです (3-6 ページの『GUI を使用した構成ファイル、鍵ストア、および証明書の作成』を参照してください)。また対称暗号鍵を生成する場合、Keytool ユーティリティーも使用できます。特に、Keytool は異なる鍵ストア間で鍵をインポートしたりエクスポートする際に役立ちます。詳細については、 3-15 ページの『Keytool -importseckey を使用したデータ鍵のインポート』および 3-15 ページの『Keytool -exportseckey を使用したデータ鍵のエクスポート』を参照してください。

Keytool は、鍵、証明書、および別名を管理するためのユーティリティーです。暗号化データ鍵を生成、インポート、およびエクスポートし、鍵ストアに保管できるようにします。

鍵ストア内の各データ鍵には、固有の別名でアクセスします。別名とは、`123456tape` など、文字のストリングです。JCEKS 鍵ストアでは、`123456Tape` は `123456tape` と同じであり、鍵ストア内の同じ項目にアクセスできるようにします。**keytool -genseckey** コマンドを使用してデータ鍵を生成する際に、対応する別名を同じコマンドに指定します。別名により、LTO 4 および LTO 5 テープ上の暗号化データの書き込みおよび読み取りに使用する正しいキー・グループおよび鍵ストア内の正しい鍵を識別できます。

注: 個々の別名および別名範囲は固有でなくてはなりません。このことは、鍵が所定の鍵ストア/Encryption Key Manager インスタンスで生成されると強制的に実行されます。ただし、複数の Encryption Key Manager/鍵ストア環境では、複数のインスタンス間で固有性を維持できる命名規則を使用してください。参照の固有性を維持しながら、インスタンス間で鍵を転送する場合、結果的にこのような命名規則を使用することが望ましくなります。

鍵と別名を生成したら、新しい別名、別名の範囲、またはキー・グループ のグループ ID 、対称鍵が保管されるファイル名、およびキー・グループが定義されるファイル名を指定するように KeyManagerConfig.properties ファイル内の symmetricKeySet プロパティーを更新してください。(詳細については、 3-17 ページの『キー・グループの作成および管理』を参照してください。) symmetricKeySet に指定された鍵のみが有効化されます (既存の別名と、適切なサイズおよびアルゴリズムの対称鍵が検査されます)。このプロパティーに無効な鍵が指定されている場合は、Key Manager は始動せず、監査レコードが作成されます。

keytool ユーティリティーは、他の鍵ストアとの間でデータ鍵のインポートおよびエクスポートも可能にします。各操作について、以下に概説します。**keytool -ekmhelp** を発行すると、以下に記載のすべての Key Manager 関連パラメーターを表示できます。

構成プロパティー・ファイルの編集

KeyManagerConfig.properties ファイルまたは ClientKeyManagerConfig.properties ファイルに変更を加えるには、以下のようにします。

1. Encryption Key Manager サーバーを停止します。

2. 任意のテキスト・エディターを使用して KeyManagerConfig.properties ファイルを開き、サーバー構成に変更を加えます。または、クライアント構成を変更するには ClientKeyManagerConfig.properties ファイルを開きます。 ^M があるため、Windows を使用してこのファイルを Linux マシン用に編集しないでください。Windows を使用する場合は、gvim/vim を使ってファイルを編集してください。
3. この資料の指示に従って、プロパティー値を変更します。
4. ファイルを保存します。
5. Encryption Key Manager サーバーを再始動します。

Keytool を使用しない場合

鍵と別名の生成に Keytool または GUI を使用しない場合、Encryption Key Manager と互換性のある範囲の鍵を生成できません。Encryption Key Manager と互換性のある個々の鍵を生成する際は、必ず、以下のフォーマットのいずれかを使用して別名を指定してください。

- 12 文字以下の印刷可能文字 (例えば、abcdefghijk)
- 3 文字の印刷可能文字、2 つのゼロ、16 の 16 進数字の順序で、正確に合計 21 文字とする (例えば、ABC000000000000000001)

Keytool -genseckey を使用したデータ鍵および別名の生成

注: どのセッションでも **keytool** コマンドを初めて使用する前に、updatePath スクリプトを実行して正しい環境を設定してください。

Windows 上の場合
cd c:\ekm にナビゲートして、**updatePath.bat** をクリックします。

Linux プラットフォーム上の場合
/var/ekm にナビゲートして、. ./updatePath.sh

と入力します。Keytool ユーティリティーは、LTO 4 および LTO 5 テープを使用して LTO 4 および LTO 5 テープ・ドライブ上での暗号化用の別名および対称鍵を生成します。1 つ以上の秘密鍵を生成し、それらを指定の鍵ストアに保管するには、**keytool -genseckey** コマンドを使用します。**keytool -genseckey** では、以下のパラメーターを受け入れます。

```
-genseckey      [-v] [-protected]
                [-alias <alias> | aliasrange <aliasRange>] [-keypass <keypass>]
                [-keyalg <keyalg>] [-keysize <keysize>]
                [-keystore <keystore>] [-storepass <storepass>]
                [-storetype <storetype>] [-providerName <name>]
                [-providerClass <provider_class_name> [-providerArg <arg>] ...
                [-providerPath <pathlist>]
```

以下のパラメーターは、LTO 4 および LTO 5 ドライブに対してテープ暗号化に役立つ Encryption Key Manager のデータ鍵を生成する場合に特に重要です。

-alias
最大 12 の印刷可能文字 (例えば、abcfrg または key123tape) を使用して単一のデータ鍵の *alias* 値を指定します。

-aliasrange
複数のデータ鍵を生成する場合、*aliasrange* を、3 文字の英字接頭部と、その後ろに一連の 16 文字 (16 進数) ストリングの下限と上限を付けたものとして指定します。これは、長さが 21 文字の別名になるように先行ゼロが自動的に埋め込まれます。例えば、`key1-a` を指定すると、`KEY000000000000000001` から `KEY00000000000000000A` までの一連の別名が生成されます。 `xyz01-FF` という *aliasrange* 値を指定すると、`XYZ000000000000000001` から `XYZ0000000000000000FF` までが生成され、これにより 255 個の対称鍵が生成されます。

-keypass
データ鍵の保護に使用されるパスワードを指定します。このパスワードは、鍵ストア・パスワードと**同じものでなければなりません**。パスワードを指定しないと、パスワードの入力を求めるプロンプトが出されます。プロンプトで「**Enter**」を押すと、キー・パスワードは、鍵ストアに使用されたパスワードと同じものに設定されます。*keypass* は、少なくとも 6 文字の長さにする必要があります。

注: 鍵ストア・パスワードを設定したら、セキュリティーが破られない限り、そのパスワードを**変更しないでください**。『鍵ストア・パスワードの変更』を参照してください。

-keyalg
データ鍵の生成に使用されるアルゴリズムを指定します。この値を `AES` と指定する必要があります。

-keysize
生成するデータ鍵のサイズを指定します。 鍵サイズは `256` と指定する必要があります。

対称鍵と関連付けられる受け入れ可能な別名の例として、次のものがあります。

```
abc000000000000000001
abc00a0120fa000000001
```

Key Manager で受け入れられない別名の例として、次のものがあります。

```
abcefghij1234567 ? wrong length
abcg0000000000000001 ? prefix is longer than 3 characters
```

鍵ストアに別名が既に存在する場合、keytool は例外をスローして停止します。

鍵ストア・パスワードの変更

注: 鍵ストア・パスワードを設定したら、セキュリティーが破られない限り、そのパスワードを**変更しないでください**。パスワードは機密漏れを排除するために暗号化されます。鍵ストア・パスワードを変更すると、次の **keytool** コマンドを使用して、その鍵ストア内のすべての鍵に対するパスワードを個別に変更する必要があります。

鍵ストア・パスワードを変更するには、次のように入力します。

```
keytool -keypasswd  -keypass old_passwd -new new_passwd -alias alias
        -keystore keystorename -storetype keystoretype
```

また、KeyManagerConfig.properties を編集し、次の方法のいずれかを使用して鍵ストア・パスワードが指定されているすべてのサーバー構成ファイル・プロパティーの鍵ストア・パスワードを変更する必要があります。

- 暗号化されたパスワード全体を削除し、次の始動時に Encryption Key Manager がプロンプトを出せるようにする。
- 暗号化されたパスワード全体を削除し、新しいパスワードを平文で入力する。これにより、次の始動時に暗号化されます。

Keytool -importseckey を使用したデータ鍵のインポート

keytool -importseckey コマンドは、秘密鍵または秘密鍵のバッチをインポート・ファイルからインポートするのに使用します。**keytool -importseckey** では、以下のパラメーターを受け入れます。

```
-importseckey        [-v]
                     [-keyalias <keyalias>] [-keypass <keypass>]
                     [-keystore <keystore>] [-storepass <storepass>]
                     [-storetype <storetype>] [-providerName <name>]
                     [-importfile <importfile>] [-providerClass <provider_class_name>]
                     [providerArg <arg>]
```

以下のパラメーターは、LTO 4 および LTO 5 ドライブに対してテープ暗号化に役立つ Encryption Key Manager のデータ鍵をインポートする場合に特に重要です。

-keyalias
importfile 内のすべてのデータ鍵を暗号化解除するために鍵ストア内の秘密鍵の別名を指定します。

-importfile
インポートされるデータ鍵が格納されているファイルを指定します。

Keytool -exportseckey を使用したデータ鍵のエクスポート

keytool -exportseckey コマンドは、秘密鍵または秘密鍵のバッチをエクスポート・ファイルにエクスポートするのに使用します。**keytool -exportseckey** では、以下のパラメーターを受け入れます。

```
-exportseckey        [-v]
                     [-alias <alias> | aliasrange <aliasRange>] [-keyalias <keyalias>]
                     [-keystore <keystore>] [-storepass <storepass>]
                     [-storetype <storetype>] [-providerName <name>]
                     [-exportfile <exportfile>] [-providerClass <provider_class_name>]
                     [providerArg <arg>]
```

以下のパラメーターは、LTO 4 および LTO 5 ドライブに対してテープ暗号化に役立つ Encryption Key Manager のデータ鍵をエクスポートする場合に特に重要です。

-alias
最大 12 の印刷可能文字 (例えば、`abcfrg` または `key123tape`) を使用して単一のデータ鍵の *alias* 値を指定します。

-aliasrange
複数のデータ鍵をエクスポートする場合、*aliasrange* を、3 文字の英字接頭部と、その後ろに一連の 16 文字 (16 進数) ストリングの下限と上限を付けたものとして指定します。これは、長さが 21 文字の別名になるように先行ゼロが自動的に埋め込まれます。例えば、key1-a を指定すると、KEY000000000000000001 から KEY00000000000000000A までの一連の別名が生成されます。 xyz01-FF という *aliasrange* 値を指定すると、XYZ000000000000000001 から XYZ0000000000000000FF までが生成されます。

-exportfile
データ鍵がエクスポートされるときに、データ鍵を保管するファイルを指定します。

-keyalias
すべてのデータ鍵を暗号化するための鍵ストア内の公開鍵の別名を指定します。対称 (データ) 鍵のインポート先となる鍵ストアに、対応する秘密鍵が格納されていることを確認してください。

JCEKS 鍵ストアを使用した LTO 4 および LTO 5 暗号化のためにセットアップされた別名と対称鍵の例

-aliasrange オプションを使用して、**KeyTool** を呼び出します。

次のように鍵アルゴリズム (-keyalg) を AES と指定し、鍵サイズ (–keysize) を 256 と指定する必要があることに注意してください。

```
/bin/keytool –genseckey –v –aliasrange AES01-FF –keyalg AES –keysize 256
–keypass password -storetype jceks –keystore path/filename.jceks
```

このように KeyTool を呼び出すと、範囲 AES000000000000000001 から AES0000000000000000FF までの 255 の順次別名と、関連する AES 256 ビット対称鍵が生成されます。どちらも、堅固な Key Manager 操作に必要な数の一定範囲およびスタンドアロンの鍵別名をセットアップするのに必要な回数だけ累積して繰り返すことができます。例えば、LTO 4 および LTO 5 用の追加の別名と対称鍵を生成するには、次のようにします。

```
/bin/keytool –genseckey –v –alias abcfrg –keyalg AES –keysize 256
–keypass password -storetype jceks –keystore path/filename.jceks
```

これを呼び出すと、スタンドアロンの別名 abcfrg が名前を指定された鍵ストアに累積的に追加されます。この鍵ストアには、上記の呼び出しによる 255 個の別名が既に含まれています。この呼び出しでは、–keystore オプションに指定された jceks ファイル内に 256 個の対称鍵が生成されます。

上で使用された別名範囲のいずれかまたはすべてに一致する後続の行と、対称鍵が保管されていたファイル名を追加するように、 KeyManagerConfig.properties ファイルの symmetricKeySet プロパティーを更新してください。無効な別名が指定された場合、Encryption Key Manager が始動しないことがあります。妥当性検査の失敗の原因として、このほかに、ビット・サイズが正しくない (AES 鍵サイズは 256 でなければなりません)、あるいはプラットフォームのアルゴリズムが正しくないことなどがあります。-keyalg は AES、-keysize は 256 でなければなりません。**config.keystore.file** に指定されたファイル名は、KeyTool の呼び出しで -keystore <filename> に指定された名前と同じものでなければなりません。

symmetricKeySet = AES01-FF,abcfrg
config.keystore.file = <filename>.jceks

symmetricKeySet に指定された鍵のみが有効化されます (既存の別名と、適切なサイズおよびアルゴリズムの対称鍵が検査されます)。このプロパティーに無効な鍵が指定されている場合は、Encryption Key Manager は始動せず、監査レコードが作成されます。

キー・グループの作成および管理

Encryption Key Manager により、LTO 4 および LTO 5 暗号化用の対称鍵をキー・グループに編成することができます。これを使用すると、暗号化するデータのタイプ、鍵へのアクセス権を持つユーザーに従って、あるいはその他の重要な特性によって鍵をグループ化できます。キー・グループを作成すると、**adddrive** コマンドで -symrec キーワードを使用して、そのキー・グループを特定のテープ・ドライブに関連付けることができます。構文については、5-9 ページの『adddrive』を参照してください。

キー・グループを作成するには、キー・グループを KeyGroups.xml ファイルに定義する必要があります。3-6 ページの『GUI を使用した構成ファイル、鍵ストア、および証明書の作成』 に手順に従った場合は、このファイルの場所は「EKM 構成 (EKM Configuration)」ページで指定されています。 構成ファイルを手動で作成する場合は、KeyGroups.xml ファイルの場所は、次のように構成プロパティー・ファイルで指定します。

```
config.keygroup.xml.file = FILE:KeyGroups.xml
```

このパラメーターが指定されていない場合は、デフォルトの動作によって、Encryption Key Manager の起動場所の作業ディレクトリーから KeyGroups.xml ファイルが使用されます。このファイルが存在しない場合、空の KeyGroups.xml ファイルが作成されます。これ以降に Encryption Key Manager サーバーを始動すると、`[Fatal Error] :-1:-1: Premature end of file.` というメッセージが **native_stderr.log** に表示される場合があります。これは空の KeyGroups.xml ファイルを構文解析する際のエラーであり、このエラーによって Encryption Key Manager サーバーの始動が妨げられることはありません。ただし、Encryption Key Manager サーバーがキー・グループを使用するように構成済みである場合を除きます。

キー・グループは、Dell Encryption Key Manager サーバー GUI を使用して、あるいは次の CLI クライアント・コマンドを使用して作成します (構文については、5-9 ページの『CLI コマンド』を参照してください)。

GUI を使用したキー・グループの定義および鍵の作成

GUI を使用すると、キー・グループの管理に必要なすべてのタスクを実行することができます。また、GUI を使用して追加の鍵を作成することもできます。

注: 次のタスクのいずれかを実行しているときに、「**Submit Changes (変更をサブミット)**」をクリックすると、バックアップ・ダイアログ・ウィンドウ (3-10 ページの図 3-6) が開いて Encryption Key Manager データ・ファイルをバックア

ップするよう促します。バックアップ・データを保管するパスを入力します。「**Submit (サブミット)**」をクリックします。バックアップ・パスを確認して、「**OK**」をクリックします。

キー・グループを作成してそれを鍵と一緒に取り込むか、あるいは既存のキー・グループに鍵を追加するには、次の手順を実行します。

1. GUI がまだ開始されていない場合は、次のように GUI を開きます。

   Windows 上の場合

   c:\ekm\gui にナビゲートして、**LaunchEKMGui.bat** をクリックします。

   Linux プラットフォーム上の場合

   /var/ekm/gui にナビゲートして、`. ./LaunchEKMGui.sh`と入力します。

2. GUI 左側のナビゲーターの「**Administration Commands (管理コマンド)**」を選択します。
3. ウィンドウ (図 3-7) の下部にある「**Create a Group of Keys (キー・グループの作成)**」をクリックします。

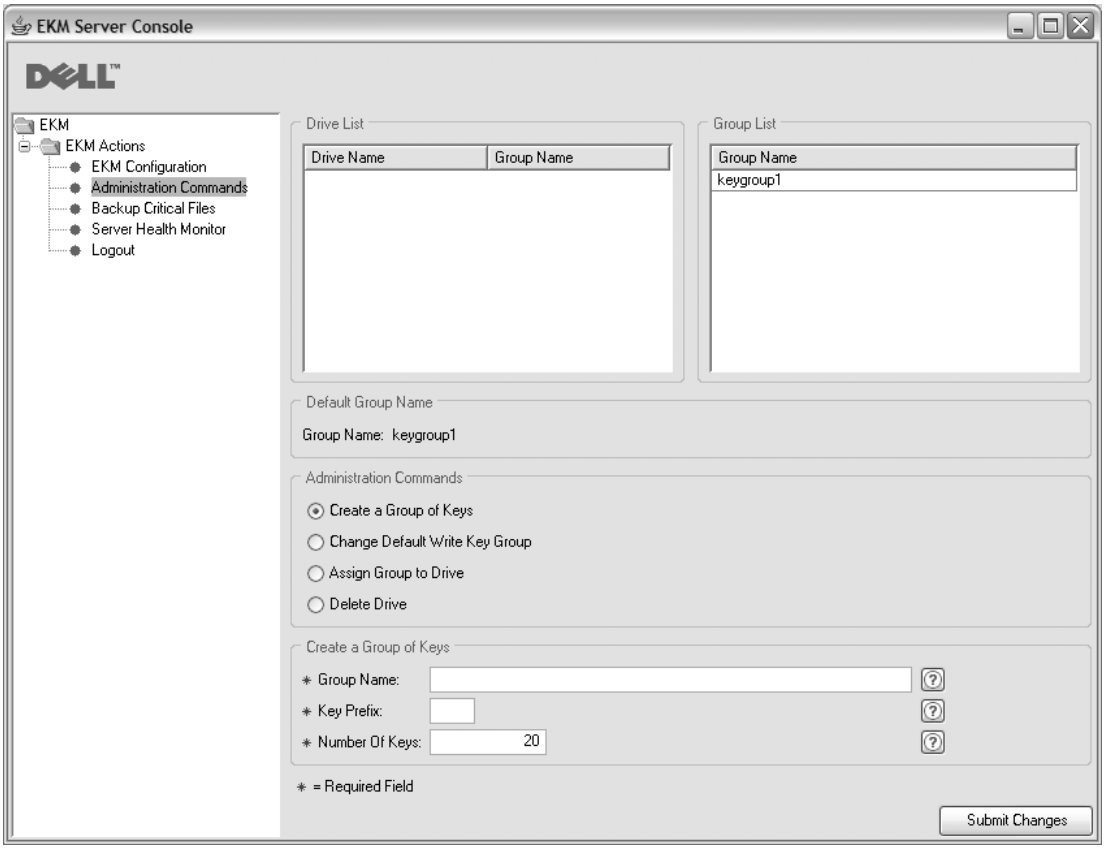

図 3-7. >Create a Group of Keys (キー・グループの作成)

4. 新しいキー・グループの名前、鍵別名に使用される接頭辞、およびグループに含まれる鍵の数を入力します。「**Submit Changes (変更をサブミット)**」をクリックします。

デフォルトのキー・グループを変更するには、次の手順を実行します。

1. GUI 左側のナビゲーターの「**Administration Commands (管理コマンド)**」を選択します。
2. ウィンドウ (図 3-8) の下部にある「**Change Default Write Key Group (デフォルト書き込みキー・グループの変更)**」をクリックします。

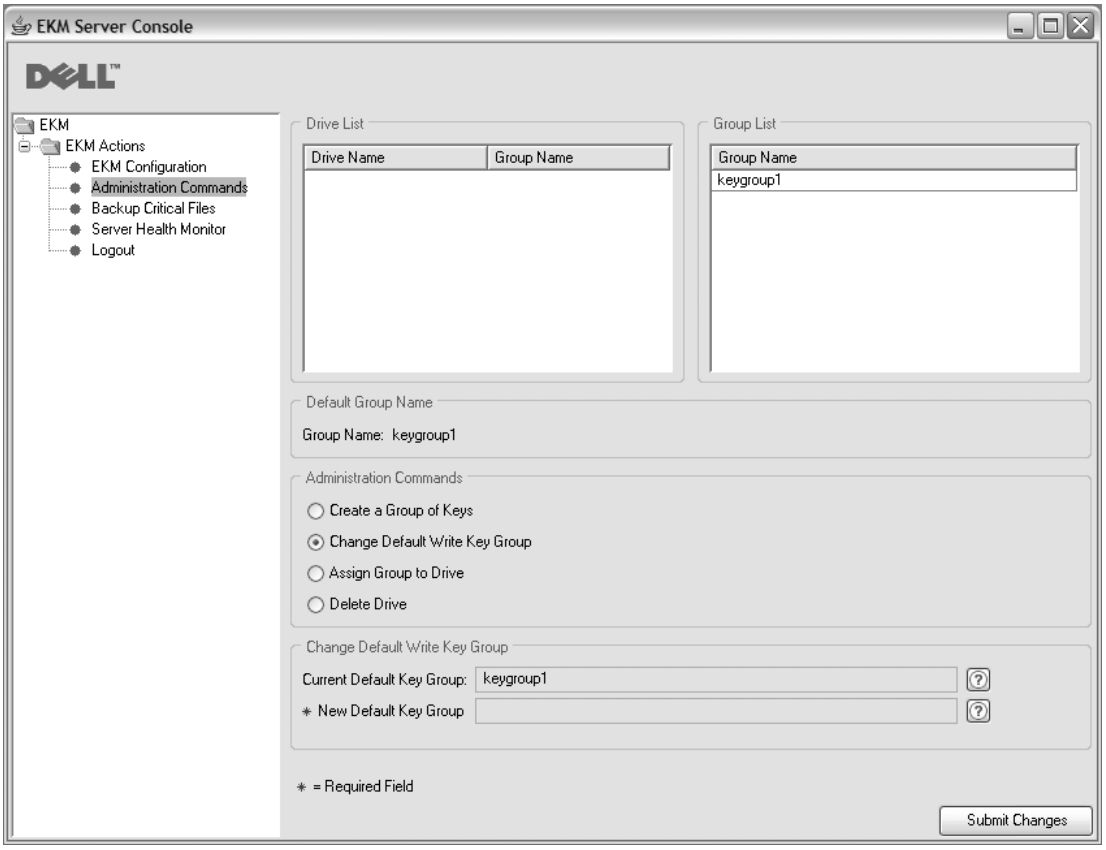

図 3-8. Change Default Write Key Group (デフォルト書き込みキー・グループの変更)

3. 右側の「Group List (グループ・リスト)」から新しいデフォルトのキー・グループを選択します。
4. ウィンドウの下部にある現行および新規のデフォルト・キー・グループを確認して、「**Submit Changes (変更をサブミット)**」をクリックします。

特定のテープ・ドライブに特定のキー・グループを割り当てるには、次の手順を実行します。

1. GUI 左側のナビゲーターの「**Administration Commands (管理コマンド)**」を選択します。
2. ウィンドウ (3-20 ページの図 3-9) の下部にある「**Assign Group to Drive (グループをドライブに割り当て)**」をクリックします。

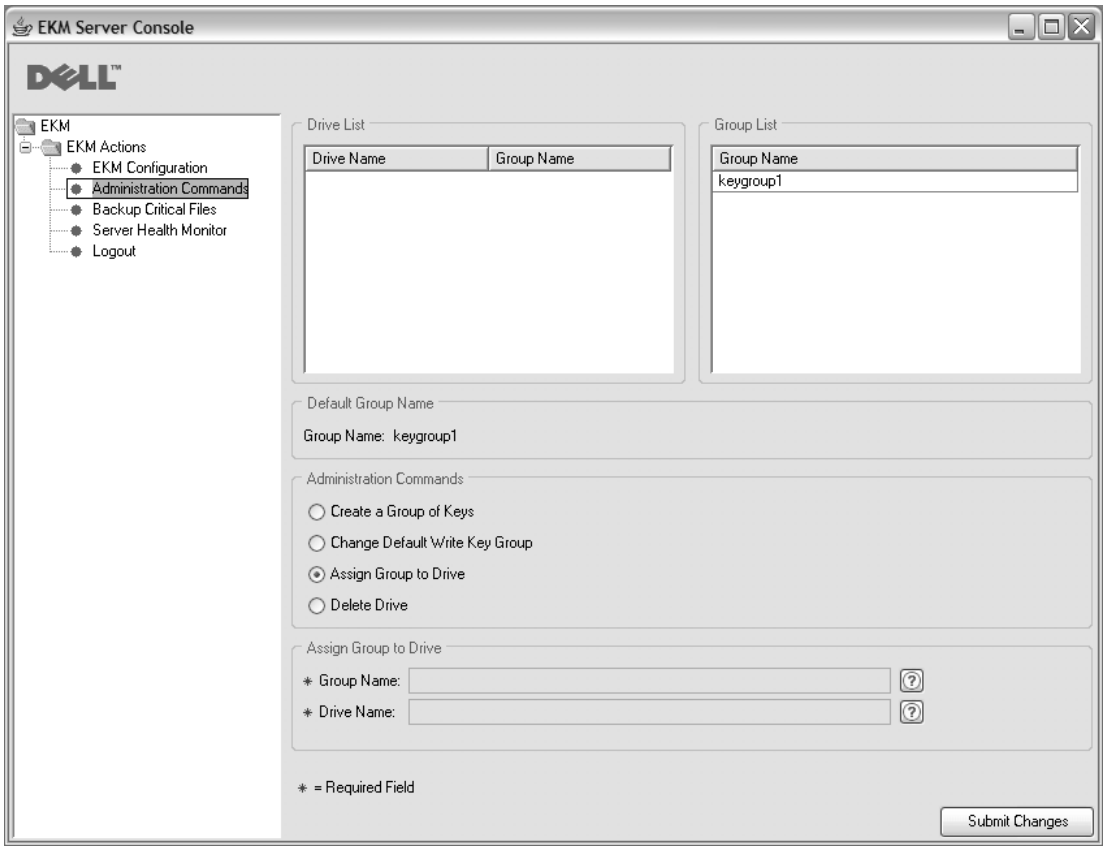

図 3-9. Assign Group to Drive (グループをドライブに割り当て)

3. 「Drive List (ドライブ・リスト)」からテープ・ドライブを選択します。
4. 「Group List (グループ・リスト)」からキー・グループを選択します。
5. ウィンドウの下部にあるドライブとキー・グループを確認して、「**Submit Changes (変更をサブミット)**」をクリックします。

ドライブ・テーブルからテープ・ドライブを削除するには、次の手順を実行します。

1. GUI 左側のナビゲーターの「**Administration Commands (管理コマンド)**」を選択します。
2. ウィンドウ (3-21 ページの図 3-10) の下部にある「**Delete Drive (ドライブの削除)**」をクリックします。

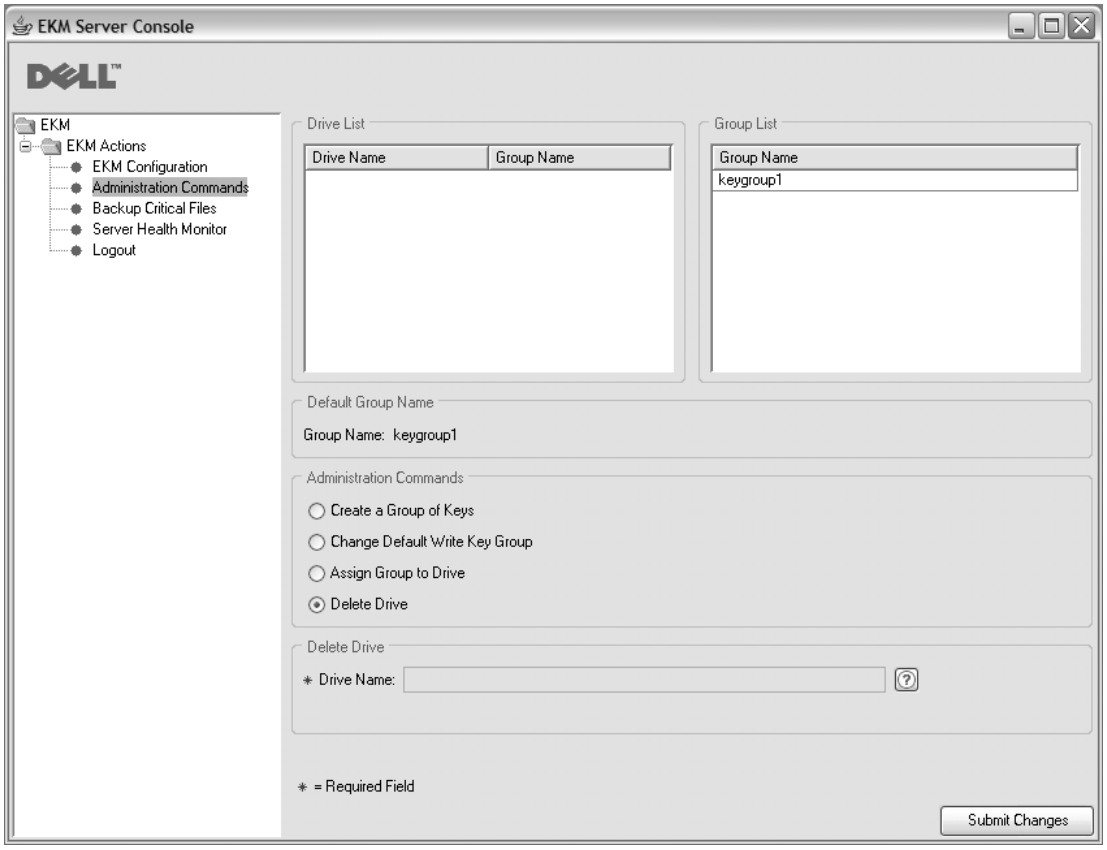

図 3-10. Delete Drive (ドライブの削除)

3. 「Drive List (ドライブ・リスト)」からテープ・ドライブを選択します。
4. ウィンドウの下部にあるドライブ名を確認して、「**Submit Changes (変更をサブミット)**」をクリックします。

CLI コマンドを使用したキー・グループの定義

Encryption Key Manager にはキー・グループ機能が備わっており、これにより鍵のセットをグループ化できます。

Encryption Key Manager アプリケーションをインストールおよび設定し (鍵ストアおよび鍵が生成される)、Encryption Key Manager サーバーを起動したら、クライアントを使用してサーバーにログインし、次のステップに従ってください。

1. **createkeygroup** コマンドを実行します。

   このコマンドを実行すると、KeyGroups.xml ファイルに最初のキー・グループ・オブジェクトが作成されます。これを 1 度だけ実行します。

   構文: **createkeygroup -password** *password*

   -password
   後の検索用に、KeyGroups.xml ファイル内の鍵ストアのパスワードを暗号化するために使用されるパスワード。鍵ストアはキー・グループの鍵を暗号化

します。これにより、各キー・グループの別名パスワードが順に暗号化されます。そのため、KeyGroups.xml ファイル内のどの鍵も平文ではありません。

例: createkeygroup -password a75xynrd

2. **addkeygroup** コマンドを実行します。

   このコマンドを実行すると、固有のグループ ID で KeyGroups.xml 内にキー・グループのインスタンスが作成されます。

   構文: **addkeygroup -groupID** *groupname*

   -groupID
   KeyGroups.xml ファイル内のグループを識別するために使用される固有のグループ名。

   例: addkeygroup -groupID keygroup1

3. **addkeygroupalias** コマンドを実行します。

   このコマンドを実行すると、鍵ストア内の既存の鍵別名に対して新しい別名を作成し、特定のキー・グループ ID に追加します。

   構文: **addkeygroupalias -alias** *aliasname* **-groupID** *groupname*

   -alias
   鍵の新しい*別名*。これは完全なキー名である必要があり、例えば Key00 の場合、key000000000000000000 と入力しなくてはなりません。

   -groupID
   KeyGroups.xml ファイル内のグループを識別するために使用される固有のグループ名。

   例: addkeygroupalias -alias key000000000000000000 -groupID keygroup1

   注: この CLI コマンドを使用する場合、1 回につき鍵を 1 つだけ追加できます。このコマンドは、キー・グループに追加する必要があるすべての鍵に対して実行しなくてはなりません。

4. キー・グループを新規のテープ・ドライブまたは既存のテープ・ドライブと関連付けます。
   a. **moddrive** コマンドを実行して、キー・グループを既存のテープ・ドライブに関連付けます。

      このコマンドを実行すると、ドライブ・テーブル内のテープ・ドライブ情報が変更されます。

      構文: **moddrive -drivename** *drivename* **-symrec** *alias*

      -drivename
      drivename は、テープ・ドライブのシリアル番号を指定します。

      -symrec
      テープ・ドライブに対して (対称鍵の) *別名*またはキー・グループ名を指定します。

例: moddrive -drivename 000123456789 -symrec keygroup1

b. **adddrive** コマンドを実行して、テープ・ドライブをドライブ・テーブルに追加し、これをキー・グループに関連付けます。

このコマンドを実行すると、ドライブを追加して特定のキー・グループと関連付けることができます。

構文: **adddrive -drivename** *drivename* **-symrec** *alias*

-drivename
drivename は、追加するドライブの 12 桁のシリアル番号を指定します。

注: 合計 12 桁にするために、10 桁のシリアル番号の前に先行ゼロを 2 つ追加する必要があります。

-symrec
テープ・ドライブに対して (対称鍵の) *別名* またはグループ ID を指定します。

例: adddrive -drivename 000123456789 -symrec keygroup1

テープ・ドライブに別名が定義されていない場合、デフォルトとして使用するキー・グループを指定するには、使用するキー・グループのグループ ID に構成プロパティー・ファイルの symmetrickeySet プロパティーを設定してください。例えば、次のようにします。

```
symmetricKeySet = keygroup1
```

GroupID は KeyGroups.xml ファイルの既存のキー・グループ ID と一致する必要があります。一致しない場合、Encryption Key Manager サーバーは始動しません。Encryption Key Manager はキー・グループ内の鍵の使用について追跡します。有効なグループ ID を指定すると、Encryption Key Manager は最後に使用された鍵を記録し、その次に指定された鍵をキー・グループの中からランダムに選択します。

キー・グループから別のキー・グループへの鍵のコピー

addaliastogroup コマンドを実行します。

このコマンドを実行すると、特定の別名が既存のキー・グループ (ソース) から新しいキー・グループ (ターゲット) にコピーされます。

構文: **addaliastogroup -aliasID** *aliasname* **-sourceGroupID** *groupname* **-targetGroupID** *groupname*

-aliasID
追加される鍵の*別名*。

-sourceGroupID
別名のコピー元となるグループを識別するために使用される固有の*グループ名*。

-targetGroupID
別名の追加先となるグループを識別するために使用される固有の*グループ名*。

例: addaliastogroup -aliasID aliasname -sourceGroupID keygroup1 -targetGroupID keygroup2

注: キーは両方のキー・グループで使用可能です。

第 4 章 Encryption Key Manager の構成

GUI を使用した Encryption Key Manager の構成

構成プロパティー・ファイルを作成するための最も簡単な方法は、 3-6 ページの『GUI を使用した構成ファイル、鍵ストア、および証明書の作成』で Dell Encryption Key Manager GUI 下記の手順を使用することです。この手順を実行した場合、既に構成ファイルが作成されているため、追加の構成は不要です。追加の Encryption Key Manager 構成オプションを利用する場合は、次の情報が役立ちます。

構成戦略

KeyManagerConfig.properties ファイルの構成設定の中には、ショートカットを可能にするものがあります。ショートカットの影響について認識することが必要になる場合があります。

テープ・ドライブ・テーブルの自動更新

Encryption Key Manager は、構成ファイル内に変数 (drive.acceptUnknownDrives) を提供します。この変数の値が `true` に設定された場合、新しいテープ・ドライブと Dell Encryption Key Manager が通信をした時点でテープ・ドライブ・テーブルを自動的に取り込みます。これにより、各テープ・ドライブまたはライブラリーにコマンド **adddrive** を使用する必要がなくなります。このモードでは、CLI クライアント・コマンドを使用して、これらの各デバイスの 10 桁のシリアル番号を入力する必要はありません。 新しいドライブは、通常の公開鍵/秘密鍵暗号方式交換を経て、テープ・デバイスの身元を検証します。この検証が完了すると、新しいデバイスは、既存のテープを、それらに保管された 鍵 ID に基づいて読み取ることが可能になります (対応する鍵情報が構成済みの鍵ストア内で見つかることを前提としています)。

注: ドライブが自動的に追加されると、ドライブがドライブ・テーブルに確実に保管されるように、GUI またはコマンド 5-15 ページの『refresh』を使用して Encryption Key Manager サーバーをリフレッシュする必要があります。

LTO 4 および LTO 5 ドライブの場合、新たに追加されたデバイス上での暗号化に対してデフォルト対称鍵プール (symmetricKeySet) を設定することができます。つまり、Encryption Key Manager に、デバイスが連絡を取ったときに関連付けられた鍵の構成要素を使用して、そのデバイスを完全に構成させることができます。デバイスがドライブ・テーブルに追加されたときにこの構成をさせないよう選択した場合は、**moddrive** コマンドを使用して、テープ・ドライブがテープ・ドライブ・テーブルに追加された後で構成させることができます。

Encryption Key Manager がサービスを提供するテープ・ドライブごとに管理者が 10 桁のシリアル番号を入力しなくて済むようにするだけでなく、大容量システム構成用のデフォルト環境にも対応できます。

このような便利さは、セキュリティーを犠牲にした上で成り立つものであることに注意してください。デバイスは自動的に追加され、認証別名と関連付けることができる (その認証別名でテープを書き込むことができる) ため、デバイスを手動で追加するときに管理者が実行する追加セキュリティー・チェックはスキップされます。テープ・ドライブ情報をドライブ・テーブルに自動的に追加して、新しいデバイスが証明書情報にアクセスすることを暗黙的に認可することが許容可能なセキュリティー・リスクであるかどうかを決定するためには、このオプションの長所および欠点を検討することが重要です。

注: プロパティー drive.acceptUnknownDrives は、デフォルトでは `false` に設定されます。そのため、Encryption Key Manager が、新しいドライブをドライブ・テーブルに自動的に追加することはありません。作動するモードを選択し、それに応じて、構成を変更してください。詳細については、付録 B を参照してください。

2 つの Key Manager サーバー間でのデータの同期化

2 つの Encryption Key Manager サーバー間で、ドライブ・テーブルと構成プロパティー・ファイルを同期させることができます。これは、CLI クライアントの **sync** コマンドを使用して手動で行うことができますが、KeyManagerConfig.properties ファイルに 4 つのプロパティーを設定することによって自動的に行うこともできます。

注

鍵ストアまたはキー・グループ XML ファイルでは、いずれの同期化方法も機能しません。これらを手動でコピーする必要があります。

自動同期化機能は、有効な IP アドレスが KeyManagerConfig.properties ファイルの sync.ipaddress プロパティーに指定されている場合にのみ使用可能です。 4-3 ページの『自動的な同期化』を参照してください。

手動による同期化

手動による方法では、CLI クライアントの **sync** コマンドを実行します。その構文は次のとおりです。

sync {**-all** | **-config** | **-drivetab**} **-ipaddr** *ip_addr* :*sslport* [**-merge** | **-rewrite**]

このコマンドは、構成ファイルのプロパティーまたはドライブ・テーブル情報、あるいはその両方をソース (つまり送信側) サーバーから、**–ipaddr** パラメーターで指定された宛先 (つまり受信側) サーバーに送信します。受信側 Encryption Key Manager サーバーは稼働中でなければなりません。

必須フィールド

-all
構成プロパティー・ファイルとドライブ・テーブル情報の両方を、**-ipaddr** で指定されたサーバーに送信します。

-config
構成プロパティー・ファイルのみを、**-ipaddr** で指定されたサーバーに送信します。

-drivetab
ドライブ・テーブル情報のみを、**-ipaddr** で指定されたサーバーに送信します。

-ipaddr
ip_addr:sslport は、受信側サーバーのアドレスおよび ssl ポートを指定します。*sslport* は、受信側サーバーの KeyManagerConfig.properties ファイルに含まれる「TransportListener.ssl.port」に指定された値と一致していなければなりません。

オプションのフィールド

-merge
受信側サーバー上で、新しいドライブ・テーブル・データを現行データにマージ (追加) します。(構成ファイルは、常に、再書き込みです。) これはデフォルトです。

-rewrite
受信側サーバー上の現行データを新しいデータと置き換えます。

自動的な同期化

ドライブ・テーブルおよびプロパティー・ファイルを 1 次 Key Manager サーバーから 2 次 サーバーに自動的に送信することができます。データの同期化が発生するようにするには、2 次サーバーが稼働している必要があります。1 次サーバーから 2 次サーバーへデータを自動的に同期させるには、1 次サーバーの KeyManagerConfig.properties ファイル内に、以下の 4 つのプロパティーを指定しておく必要があります。2 次、つまり受信側サーバーのプロパティー・ファイルを変更する必要はありません。

sync.ipaddress
受信側サーバーのアドレスおよび ssl ポートを指定します。例えば、

```
sync.ipaddress = backupekm.server.ibm.com:1443
```

このプロパティーが指定されていない、あるいは誤って指定された場合は、自動同期化が使用不可になります。

sync.action
受信側サーバーの既存データをマージまたは再書き込みします。有効な値は **merge** (デフォルト) または **rewrite** です。構成プロパティーを同期させると、結果として、常に再書き込みが行われることになります。

sync.timeinhours
データの送信頻度。この値は整数 (時間) で指定します。時間間隔は、サーバーが始動したときを開始点とします。つまり、同期化は、サーバーが指定された時間数だけ実行した後に起きます。デフォルトは 24 です。

sync.type
送信するデータを示します。有効な値は **drivetab** (デフォルト)、**config**、および **all** です。

基本構成

注: 3-6 ページの『GUI を使用した構成ファイル、鍵ストア、および証明書の作成』 の手順を実行した場合は、基本構成が既に作成されているため、以下のステップを実行する必要はありません。この情報は GUI を使用せずにこれらのタスクを実行する方法を示し、追加の構成オプションを利用する場合には有用です。

Windows ユーザーのための注: Windows では、ブランクを含むディレクトリー・パスは受け入れられません。このようなディレクトリーについては、コマンドを入力する際に、`Program Files` の代わりに `progra~1` というようなショート・ネームを作成して指定する必要があります。ディレクトリーのショート・ネームをリストするには、**dir /x** コマンドを発行します。

この手順には、Encryption Key Manager の構成に必要な最小限のステップが含まれています。付録 A には、サーバー構成プロパティー・ファイルの例が含まれています。サーバー構成およびクライアント構成の両方に関するすべてのプロパティーの完全なリストについては、付録 B を参照してください。

1. **keytool** を使用して、JCEKS 鍵ストアを管理します。鍵ストアを作成するときに、証明書および鍵に与えられた名前のほか、パスおよびファイル名もメモしてください。この情報は、後のステップで使用します。
2. 鍵ストアが 1 つもない場合は、作成します。この新しい鍵ストアに、テープ・ドライブと一緒に使用される証明書と鍵を追加またはインポートしてください。(3-12 ページの『LTO 4 および LTO 5 上での暗号化のための鍵と別名の生成』を参照。) 証明書および鍵に与えられた名前をメモしてください。この情報は、後のステップで使用します。
3. キー・グループを作成して鍵別名で取り込みます。 3-17 ページの『キー・グループの作成および管理』を参照してください。
4. 任意のテキスト・エディターを使用して **KeyManagerConfig.properties** を開き、以下のプロパティーを指定します。 サーバーの現行設計は極めて厳密なものであることに注意してください。^M があるため、Windows を使用して、このファイルを Linux マシン用に編集しないでください。Windows を使用する場合は、gvim/vim を使ってファイルを編集してください。

   Windows ユーザーのための注: Java SDK は、Windows で実行する場合でも、順スラッシュを使用します。**KeyManagerConfig.properties** ファイルにパスを指定する場合は、必ず順スラッシュを使用してください。コマンド・ウィンドウで、完全に修飾されたパス名を指定する場合は、Windows の通常の方法に従い、逆スラッシュを使用してください。

   a. **Audit.Handler.File.Directory** – 監査ログが保管される場所を指定します。
   b. **Audit.metadata.file.name** – メタデータ XML ファイルの完全修飾パスおよびファイル名を指定します。

c. **Config.drivetable.file.url** – Encryption Key Manager が認識するドライブに関する情報の場所を指定します。サーバーまたは CLI クライアントを始動させるまでは、このファイルは必要ありません。このファイルが存在しない場合は、Encryption Key Manager サーバーのシャットダウン中に作成されます。

d. **TransportListener.ssl.keystore.name** – ステップ 1 で作成した鍵ストアのパスおよびファイル名を指定します。

e. **TransportListener.ssl.truststore.name** - ステップ 1 で作成した鍵ストアのパスおよびファイル名を指定します。

f. **Admin.ssl.keystore.name** - ステップ 1 で作成した鍵ストアのパスおよびファイル名を指定します。

g. **Admin.ssl.truststore.name** - ステップ 1 で作成した鍵ストアのパスおよびファイル名を指定します。

h. **config.keystore.file** - ステップ 1 で作成した鍵ストアのパスおよびファイル名を指定します。

i. **drive.acceptUnknownDrives** - `true` または `false` を指定します。 true という値が指定されると、Encryption Key Manager に連絡を取る新しいテープ・ドライブをドライブ・テーブルに自動的に追加できます。デフォルトは false です。

5. 以下のオプションのパスワード項目は、追加または省略される場合があります。これらの項目が **KeyManagerConfig.properties** に指定されない場合は、Encryption Key Manager がサーバーの始動中に、鍵ストア・パスワードを求めるプロンプトを出します。

   a. **Admin.ssl.keystore.password** - ステップ 1 で作成した鍵ストアのパスワードを指定します。

   b. **config.keystore.password** - ステップ 1 で作成した鍵ストアのパスワードを指定します。

   c. **TransportListener.ssl.keystore.password** - ステップ 1 で作成した鍵ストアのパスワードを指定します。

   KeyManagerConfig.properties ファイルに追加されると、Encryption Key Manager は、セキュリティーを強化するためにこれらのパスワードを暗号化します。

6. CLI クライアント認証がローカル・オペレーティング・システム・レジストリーに対して実行される場合は、オプションとして **Server.authMechanism** プロパティーを `LocalOS` の値に設定してください。 未指定 (または `EKM` に設定されている) 場合、デフォルトで CLI クライアント・ユーザーは、usr/passwd を EKMAdmin/changeME として使用して Key Manager サーバーにログインしなければなりません (このパスワードは **chgpasswd** コマンドを使用して変更できます)。

   Server.authMechanism プロパティーが LocalOS に設定されている場合、Linux プラットフォームでは追加のセットアップが必要になります。詳細については、**http://support.dell.com** にアクセスするか、製品に付属の Dell Encryption Key Manager メディアに格納されている README ファイルを参照してください。 5-6 ページの『CLI クライアント・ユーザーの認証』 には詳細情報が記載されています。

7. **KeyManagerConfig.properties** に対する変更を保管します。
8. Encryption Key Manager サーバーを始動します。GUI を使用せずにサーバーを起動するには、次の手順を実行します。

   Windows 上の場合

   cd c:\ekm\ekmserver にナビゲートして、**startServer.bat** をクリックします。

   Linux プラットフォーム上の場合

   /var/ekm/ekmserver にナビゲートして、`. ./startServer.sh`

   と入力します。詳細については、5-1 ページの『Key Manager サーバーの始動、リフレッシュ、および停止』を参照してください。
9. CLI クライアントを起動するには、次の手順を実行します。

   Windows 上の場合

   cd c:\ekm\ekmclient にナビゲートして、**startClient.bat** をクリックします。

   Linux プラットフォーム上の場合

   /var/ekm/ekmclient にナビゲートして、`. ./startClient.sh`

   と入力します。詳細については、5-6 ページの『コマンド行インターフェース・クライアント』を参照してください。
10. ステップ 4(i) で **drive.acceptUnknownDrives = false** と指定した場合は、# プロンプトで次のように入力してドライブを構成します。

    `adddrive -drivename` *`drive_name`* `-rec1` *`cert_name`* `-rec2` *`cert_name`*

    例えば、次のようにします。

    ```
    # adddrive -drivename 000001365054 -rec1 key1c1 -rec2 key1c2
    ```

    この後に、次のように続けます。

    ```
    # listdrives -drivename 000001365054
    ```

    以下が返されます。

    ```
    Entry Key:  SerialNumber = 000001365054

    Entry Key:  AliasTwo = key1c2

    Entry Key:  AliasOne = key1c1
     Deleted : false
     Updated : true
     TimeStamp : Sun Jul 03 17:34:44 MST 2007
    ```
11. # プロンプトに **listdrives** コマンドを入力して、ドライブが正常に追加されたことを確認します。

第 5 章 Encryption Key Manager の管理

Key Manager サーバーの始動、リフレッシュ、および停止

Encryption Key Manager サーバーの始動および停止は簡単です。

サーバーをリフレッシュすると、Encryption Key Manager は、鍵ストア、ドライブ・テーブル、および構成情報の現在の内容をメモリーからそれぞれのファイルにダンプしてから、これらをメモリーに再ロードします。 CLI クライアントを使用してリフレッシュを実行することは、これらのコンポーネントに何らかの変更が加えられた後で役立ちます。このような変更は Encryption Key Manager サーバーのシャットダウン時に自動的に保存されますが、サーバーのリフレッシュを実行すると、システムの異常終了または停電の場合にこれらの変更が失われることを防ぐことができます。

次の手順を使用して、Dell Encryption Key Manager GUI からEncryption Key Manager サーバーを始動します。

1. GUI がまだ開始されていない場合は、次のように GUI を開きます。

   Windows 上の場合
   c:\ekm\gui にナビゲートして、**LaunchEKMGui.bat** をクリックします。

   Linux プラットフォーム上の場合
   /var/ekm/gui にナビゲートして、`. ./LaunchEKMGui.sh` と入力します。
2. GUI 左側のナビゲーターの「**Server Health Monitor (サーバーの正常性モニター)**」をクリックします。
3. 「Server Status (サーバーの状況)」ページ (5-2 ページの図 5-1) で、「**Start Server (サーバーの始動)**」または「**Refresh Server (サーバーのリフレッシュ)**」をクリックします。

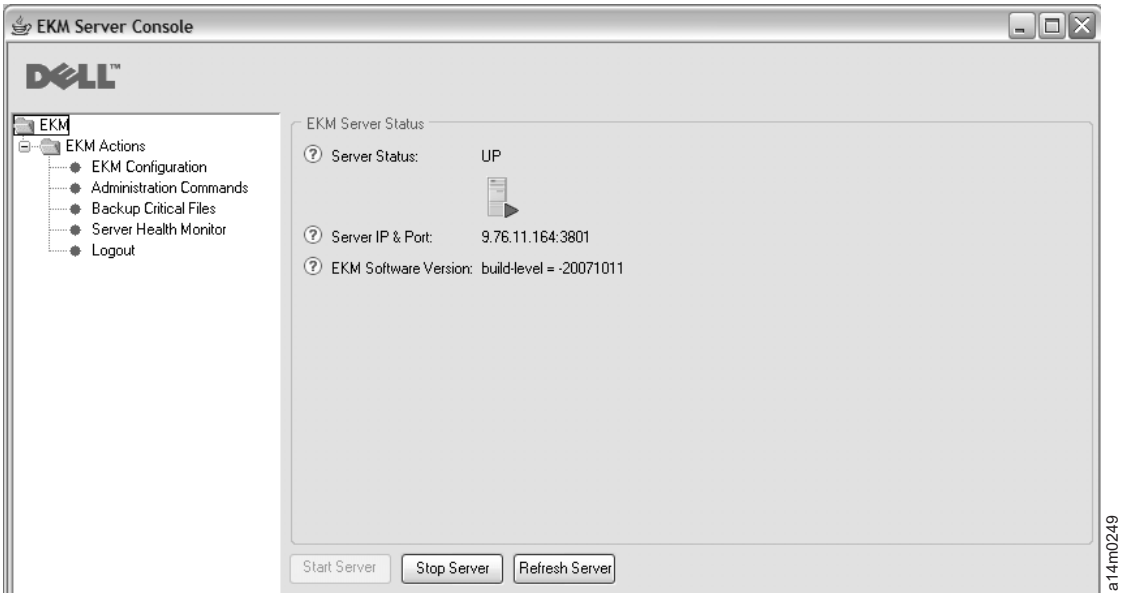

図 5-1. Server Status (サーバーの状況)

4. サーバー状況の変更が、「Server Status (サーバーの状況)」ウィンドウに反映されます。図 5-1を参照してください。
5. 「Login (ログイン)」ウィンドウが表示されます (図 5-2)。

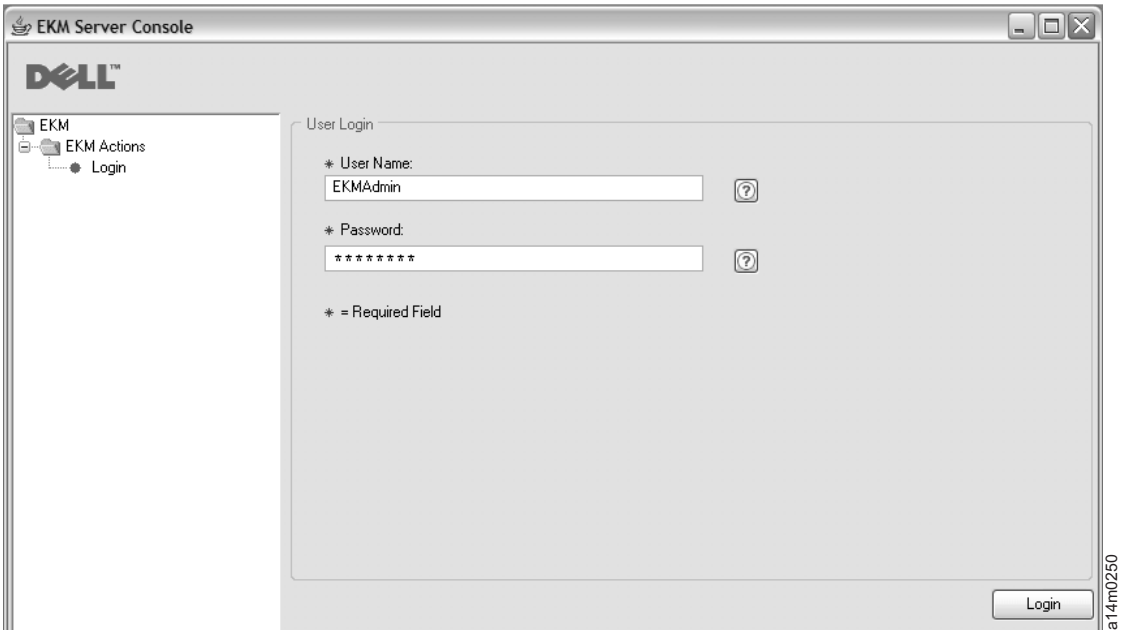

図 5-2. 「Login (ログイン)」ウィンドウ

「User Name (ユーザー名)」に EKMAdmin と入力します。初期パスワードは changeME です。ログイン後に、**chgpasswd** コマンドを使用してパスワードを変更できます。 5-11 ページの『chgpasswd』を参照してください。

注: • Dell Encryption Key Manager GUI では、ホスト IP アドレスを表示できない場合があります。

現在の GUI における以下の 2 つの制限により、Encryption Key Manager のホスト IP アドレスが「**Server Health Monitor (サーバーの正常性モニター)**」に表示されない場合があります。

- 現行のアプリケーションでは、IPV6 は認識されません。ホストが IPV6 アドレスで構成されている場合、Encryption Key Manager アプリケーションは IP アドレスを表示できません。
- Encryption Key Manager アプリケーションが Linux システムにインストールされている場合、このアプリケーションは実際にアクティブである IP ポートではなく、ローカル・ホスト・アドレスを表示します。

ホスト・システムの実際の IP アドレスを取得するには、ネットワーク構成にアクセスし、IP ポート・アドレスを見つけてください。 Windows システムでは、コマンド・ウィンドウを開き、`ipconfig` と入力します。 Linux の場合は、`isconfig` と入力します。

6. 「**Login (ログイン)**」をクリックします。

同じ「Server Status (サーバーの状況)」ページを使用してサーバーを停止します。

スクリプトを使用した Key Manager サーバーの始動

Windows 上の場合

cd c:\ekm\ekmserver にナビゲートして、**startServer.bat** をクリックします。

Linux プラットフォーム上の場合

/var/ekm/ekmserver にナビゲートして、`. ./startServer.sh` と入力します。

サーバーを停止するには、後述の 5-6 ページの『コマンド行インターフェース・クライアント』 に示す方法のいずれかを使用して **stopekm** コマンドを実行します。もう 1 つの方法として、**sigterm** を Key Manager プロセスに送信する方法があります。これにより、サーバーは正確にシャットダウンして終了します。Key Manager プロセスに **sigkill** を送信しないでください。**sigkill** の場合、プロセスは正確にシャットダウンされません。例えば、Linux システムでは、`kill -SIGTERM` *`pid`* または `kill -15` *`pid`* と入力してください。

コマンド・プロンプトからの Key Manager サーバーの始動および停止

任意のコマンド・ウィンドウまたはシェルから Encryption Key Manager サーバーを始動するには、次のように入力します。

```
java com.ibm.keymanager.EKMLaunch KeymanagerConfig.properties
```

これにより、Encryption Key Manager サーバーがバックグラウンドで起動します。適切に始動したら、Encryption Key Manager Java プロセスを、`ps -ef | grep java` コマンド (Linux プラットフォーム) または Windows タスク マネージャを使用して表示できます。Windows サービスとして実行する場合は、LaunchEKMService として表示されます。

サーバーを停止するには、後述の 5-6 ページの『コマンド行インターフェース・クライアント』 に示す方法のいずれかを使用して **stopekm** コマンドを実行します。もう 1 つの方法として、**sigterm** を Key Manager プロセスに送信する方法があります。これにより、サーバーは正確にシャットダウンして終了します。Key Manager

プロセスに **sigkill** を送信しないでください。**sigkill** の場合、プロセスは正確にシャットダウンされません。例えば、Linux システムでは、`kill -SIGTERM pid` または `kill -15 pid` と入力してください。

Windows プラットフォームでは、 Dell Encryption Key Manager は、Windows サービスとして開始された場合、「コントロール パネル」から停止することができます。

Key Manager サーバーの Windows サービスとしてのインストール

Encryption Key Manager サーバーをサービスとしてホスト・サーバーにインストールすることにより、ホスト・サーバーがリブートされた際に Encryption Key Manager サーバー・アプリケーションが確実に起動するようにします。

1. Dell サポート Web サイト (**http://support.dell.com**) のリリース・ダウンロードから、実行可能な LaunchEKMService.exe ファイルを一時ディレクトリーに解凍します。
2. サービスを適切に実行するには、いくつかの環境変数を以下のように設定する必要があります。
   a. 「スタート」メニューから「**コントロール パネル**」をクリックします。
   b. 「**システム**」をダブルクリックします。
   c. 「**詳細設定**」タブをクリックします。
   d. 「**環境変数**」をクリックします。
   e. 「システム環境変数」のリストの下にある「**新規**」をクリックします。
   f. 変数として `JAVA_HOME` を指定し、IBM JVM ディレクトリーを入力します。デフォルトのインストール・ディレクトリーは、`C:¥PROGRA~1¥IBM¥Java60` です。
   g. 「**OK**」をクリックします。
3. この手順を使用してシステム PATH 変数を編集します。

   注: PATH 変数は、コマンド行から設定しても機能しません。
   a. 「スタート」メニューから「**コントロール パネル**」をクリックします。
   b. 「**システム**」をダブルクリックします。
   c. 「**詳細設定**」タブをクリックします。
   d. 「**環境変数**」をクリックします。
   e. 「システム環境変数」のリストをスクロールして **PATH** 変数を見つけ、「**編集**」をクリックします。
   f. PATH 変数の先頭に `IBM JVM` パスを追加します。デフォルトのインストール・ディレクトリーは、`C:¥PROGRA~1¥IBM¥Java60¥jre¥bin` です。

      注: 追加するパスの終わりにセミコロンを挿入し、パス・リスト内の他のディレクトリーと区別するようにしてください。
   g. 「**OK**」をクリックします。
4. Encryption Key Manager サーバー構成プロパティー・ファイル内のパスが、完全修飾であることを確認してください。このファイルは KeyManagerConfig.properties という名前で、`C:\ekm\gui` ディレクトリーにあります。ファイル内の以下のパスをすべて確認して更新し、必ず完全修飾パスを含む

ようにする必要があります (例えば、gui\EKMKeys.jck ではなく c:\ekm\gui\EKMKeys.jck を使用します)。デフォルトのインストールを使用している場合のパスの変更方法については、以下の例を参照してください。

以下は、デフォルトのインストールと鍵ストア名を使用している場合の、プロパティーおよびそのポイント先である完全修飾パスです。
KeyManagerConfig.properties ファイル内で、以下の各項目を見つけることができます。

config.keygroup.xml.file
パスは、FILE:C:/ekm/gui/keygroups/KeyGroups.xml に変更する必要があります。

Admin.ssl.keystore.name
パスは、C:/ekm/gui/EKMKeys.jck に変更する必要があります。

TransportListener.ssl.truststore.name
パスは、C:/ekm/gui/EKMKeys.jck に変更する必要があります。

Audit.metadata.file.name
パスは、C:/ekm/gui/metadata/ekm_metadata.xml に変更する必要があります。

Audit.handler.file.directory
パスは、C:/ekm/gui/audit に変更する必要があります。

config.keystore.file
パスは、C:/ekm/gui/EKMKeys.jck に変更する必要があります。

TransportListener.ssl.keystore.name
パスは、C:/ekm/gui/EKMKeys.jck に変更する必要があります。

config.drivetable.file.url
パスは、FILE:C:/ekm/gui/drivetable/ekm_drivetable.dt に変更する必要があります。

Admin.ssl.truststore.name
パスは、C:/ekm/gui/EKMKeys.jck に変更する必要があります。

5. **LaunchEKMServices.exe** ファイルは、コマンド・プロンプトから実行する必要があります。これにアクセスするには、Windows で「**スタート**」>「**すべてのプログラム**」>「**アクセサリ**」>「**コマンド プロンプト**」の順にナビゲートします。
6. コマンド・プロンプトから、**LaunchEKMService.exe** が解凍された一時ディレクトリーへナビゲートします。以下のオプションを参照して、**LaunchEKMService.exe** ファイルを実行します。

LaunchEKMService {**-help** | **-i** *config_file* | **-u**}

-help
使用情報を表示します。

-i Encryption Key Manager を Windows サービスとしてインストールします。このオプションには、引数として引き渡される構成プロパティー・ファイルの絶対パス名が必要です。デフォルトのパスおよびファイル名は、C:\ekm\gui\KeyManagerConfig.properties です。

-u Key Manager Windows サービスをサービスとして実行する必要がなくなった場合に、アンインストールします。 EKMServer サービスは、停止してからアンインストールする必要がありますので、注意してください。このコマンドを実行すると、次のエラー・メッセージも表示される場合があります。 `Could not remove EKMServer. Error 0.` (EKMServer を除去できません。エラー 0 です。) ただし、サービスはアンインストールされている可能性があります。

Encryption Key Manager を Windows サービスとしてインストールするには、以下を実行します。

`LaunchEKMService.exe -i config file`

7. このサービスを上記のコマンドを使用してインストールすると、EKMServer がサービス・コントロール・パネルに表示され、Encryption Key Manager をサービス・コントロール・パネルから起動したり、停止することができます。

注: Windows サービスは初めて使用するときに、「コントロール パネル」を使用して手動で起動する必要があります。

コマンド行インターフェース・クライアント

Encryption Key Manager サーバーが起動したら、ローカルまたはリモートからクライアント・インターフェースを介して、CLI コマンドを発行することができます。CLI コマンドを発行するには、最初に CLI クライアントを起動する必要があります。

CLI クライアント・ユーザーの認証

構成ファイルの Server.authMechanism プロパティーはローカル/リモート・クライアントで使用される認証メカニズムを指定します。値が `EKM` に設定されている場合、CLI クライアント・ユーザーは user/password を EKMAdmin/changeME として使用し、サーバーにログインする必要があります (このパスワードは **chgpasswd** コマンドを使用して変更できます。 5-11 ページの『chgpasswd』を参照してください。) Server.authMechanism プロパティーのデフォルト設定は、`EKM` です。

Server.authMechanism プロパティーの値が KeyManagerConfig.properties 内で `LocalOS` と指定されている場合、クライアント認証はローカル・オペレーティング・システム・レジストリーに対して実行されます。CLI クライアント・ユーザーは OS user/password を使用してサーバーにログインする必要があります。サーバーへのログインおよびサーバーへのコマンド発行が許可されているユーザー/パスワードは、その ID の下でサーバーが実行されており、かつ superuser/root 権限を持っているユーザー ID のみであることに注意してください。

重要: Encryption Key Manager 構成ファイルにこれらの変更を加える際には、Encryption Key Manager サーバーをオフにし、GUI を閉じる必要があります。

Windows におけるローカル OS ベースの認証では、以下のようにして KeyManagerConfig.properties に Server.authMechanism=LocalOS を設定する必要があります。

1. KeyManagerConfig.properties ファイル (`c:\ekm\gui` ディレクトリー) を見つけます。

2. 任意のテキスト・エディター (ワードパッドを推奨) でこのファイルを開きます。
3. Server.authMechanism ストリングを見つけます。このストリングが存在しない場合、ファイルに`Server.authMechanism=LocalOS` を正確にこの形式で追加してください。
4. ファイルを保存します。

これで、Encryption Key Manager サーバーのユーザー ID およびパスワードが、OS ユーザー・アカウントと一致するようになりました。サーバーへのログインおよびサーバーへのコマンド発行が許可されており、管理者特権を持つユーザーのみが、Encryption Key Manager サーバーを管理できることに注意してください。

Linux プラットフォームにおけるローカル OS ベースの認証の場合、以下の追加のステップが必要です。

1. **http://support.dell.com** から Download Dell Release R175158 (EKMServicesAndSamples) をダウンロードし、そのファイルを任意のディレクトリーに解凍します。
2. このダウンロードの際に、LocalOS ディレクトリーを見つけます。
3. libjaasauth.so ファイルを、ご使用のプラットフォームに該当する JVM-JaasSetup ディレクトリーから *java_home*/jre/bin にコピーします。
   - 32 ビット Intel Linux 環境では、LocalOS-setup/linux_ia32/libjaasauth.so ファイルを *java_home*/jre/bin/ ディレクトリーにコピーします。ここで、1.6 JVM を実行する 32 ビット Intel Linux カーネルの場合、*java_home* は、通常 *java_install_path*/IBMJava-i386-60 です。
   - 64 ビット AMD64 Linux 環境では、LocalOS-setup/linux-x86_64/libjaasauth.so ファイルを *java_home*/jre/bin/ ディレクトリーにコピーします。ここで、1.6 JVM を実行する 64 ビット Linux カーネルの場合、*java_home* は、通常 *java_install_path*/IBMJava-x86_64-60 です。

Windows プラットフォームでは、このファイルは不要です。

インストールが終了したら、Encryption Key Manager サーバーを始動できます。Encryption Key Manager クライアントは、OS ベースのユーザー/パスワードでログインできるようになっています。サーバーへのログインまたはサーバーへのコマンド発行が許可されているユーザー ID は、その ID の下でサーバーが実行されており、かつ superuser/root 権限を持っているユーザー ID のみであることに注意してください。

インストールの詳細については、README ファイル (Dell 製品メディア内、および **http://support.dell.com** で入手可能) を参照してください。

コマンド行インターフェース・クライアントの起動

注: Encryption Key Manager サーバー・プロパティー・ファイルと Encryption Key Manager CLI クライアント・プロパティー・ファイルの TransportListener.ssl.port プロパティーは両方とも、同じ値に設定する必要があります。同じ値に設定されていない場合、これらは通信しません。問題が発生する場合は、6-2 ページの『CLI クライアントと EKM サーバー間の通信問題のデバッグ』を参照してください。

Encryption Key Manager CLI クライアントと Encryption Key Manager サーバーは、SSL を使用してセキュアな通信を行います。クライアント認証のないデフォルトの JSSE 構成を使用する場合、Encryption Key Manager サーバー上の TransportListener.ssl.keystore の証明書は TransportListener.ssl.truststore 内に存在しなければなりません。この方法で、クライアントはサーバーに信用性があることを認識します。Encryption Key Manager CLI クライアントが Encryption Key Manager サーバーと同じシステム上で実行されている場合、同じ構成プロパティー・ファイルを使用することができます。これにより、Encryption Key Manager CLI クライアントが、Encryption Key Manager サーバーと同じ鍵ストア/トラストストア構成を使用できるようになります。これらが同じシステム上にない場合、またはクライアントで異なる鍵ストアを使用したい場合、Encryption Key Manager サーバー構成プロパティー・ファイルで指定された TransportListener.ssl.keystore から証明書をエクスポートする必要があります。これらの証明書は、Encryption Key Manager CLI プロパティー・ファイル内の TransportListener.ssl.truststore で指定されたトラストストアにインポートする必要があります。

CLI クライアントを起動して、CLI コマンドを発行する方法は 4 つあります。いずれを選択しても、CLI 構成ファイルの名前を指定する必要があります。詳細については付録 B を参照してください。

スクリプトの使用

Windows 上の場合

cd c:\ekm\ekmclient にナビゲートして、**startClient.bat** をクリックします。

Linux プラットフォーム上の場合

/var/ekm/ekmclient にナビゲートして、`. ./startClient.sh`と入力します。

対話式に実行する場合

任意のコマンド・ウィンドウまたはシェルから対話的にコマンドを実行するには、次のように入力します。

```
java com.ibm.keymanager.KMSAdminCmd CLIconfiglfile_name -i
```

プロンプトが表示されます。いずれのコマンドも実行する前に、CLI クライアントで次のコマンド使用して Key Manager サーバーにログインする必要があります。

```
#login –ekmuser EKMAdmin –ekmpassword changeME
```

CLI クライアントが正常に Key Manager サーバーにログインしたら、どの CLI コマンドでも実行することができます。完了したら、**quit** コマンドまたは **logout** コマンドを使用して、CLI クライアントをシャットダウンします。デフォルトでは Encryption Key Manager サーバーは、クライアントが 10 分間使用されないと、通信ソケットを閉じます。この後コマンドの入力を試行しても、結果的にクライアントは終了します。 Encryption Key Manager サーバーとクライアント間のソケットのタイムアウト期間をより長く指定するには、KeyManagerConfig.properties ファイル内の TransportListener.ssl.timeout プロパティーを変更します。

コマンド・ファイルの使用

Key Manager サーバーにファイルのコマンドをまとめて実行する場合、発行しようとするコマンドを含んでいるファイル (例えば、*clifile*) を作成します。この

ファイルの最初のコマンドは、どのコマンドでも実行する前に、クライアントにログインを要求することから、**login** コマンドでなくてはなりません。例えば、clifile には次のものが含まれます。

```
login -ekmuser EKMAdmin -ekmpassword changeME
listdrives
```

次にこのコマンド・ファイルを実行するために、次のように CLI クライアントを起動します。

```
java com.ibm.keymanager.admin.KMSAdminCmd CLIconfiglfile_name –filename clifile
```

コマンドを 1 つずつ実行

コマンドごとに CLI userid_ID およびパスワードを指定して、コマンドを 1 つずつ実行することができます。任意のコマンド・ウィンドウまたはシェルから、次のように入力します。

```
java com.ibm.keymanager.KMSAdminCmd ClientConfig.properties_name -listdrives
    -ekmuser EKMAdmin -ekmpassword changeME
```

(このパスワードは **chgpasswd** コマンドを使用して変更できます。) このコマンドにより、クライアント・セッションを実行したり、終了させたりできます。

CLI コマンド

Encryption Key Manager は、次のコマンドを含むコマンド行インターフェース・クライアントからEncryption Key Manager サーバーと対話するために使用可能なコマンド・セットを提供します。

addaliastogroup

特定の別名を既存のキー・グループ (ソース) から新しいキー・グループ (ターゲット) にコピーします。これは、あるキー・グループに既に存在する別名を別のキー・グループに追加するときに役立ちます。

addaliastogroup -aliasID *aliasname* **-sourceGroupID** *groupname* **-targetGroupID** *groupname*

-aliasID
追加される鍵の別名。

-sourceGroupID
別名のコピー元となるグループを識別するために使用される固有のグループ名。

-targetGroupID
別名の追加先となるグループを識別するために使用される固有のグループ名。

例: addaliastogroup -aliasID aliasname -sourceGroupID keygroup1 -targetGroupID keygroup2

adddrive

新しいドライブを Key Manager ドライブ・テーブルに追加します。テープ・ドライブをドライブ・テーブルに自動的に追加する方法については、 4-1 ページの『テー

プ・ドライブ・テーブルの自動更新』を参照してください。別名の要件については、 2-4 ページの『暗号鍵と LTO 4 および LTO 5 テープ・ドライブ』を参照してください。

adddrive -drivename *drivename* [**-rec1** *alias*] [**-rec2** *alias*][**-symrec** *alias*]

-drivename
drivename は、追加するドライブの 12 桁のシリアル番号を指定します。

注: 合計 12 桁にするために、10 桁のシリアル番号の前に先行ゼロを 2 つ追加する必要があります。

-rec1
ドライブの証明書の*別名* (または鍵ラベル) を指定します。

-rec2
ドライブの証明書の 2 つ目の*別名* (または鍵ラベル) を指定します。

-symrec
テープ・ドライブに対して (対称鍵の) *別名*またはキー・グループ名を指定します。

例: adddrive -drivename 000123456789 -rec1 alias1 -rec2 alias2

addkeygroup

固有のグループ ID を使用して、キー・グループ XML にキー・グループのインスタンスを作成します。

addkeygroup -groupID *groupname*

-groupID
KeyGroup XML ファイル内のグループを識別するために使用される固有のグループ名。

例: addkeygroup -groupID keygroup1

addkeygroupalias

鍵ストア内の既存の鍵別名に対して新しい別名を作成し、特定のキー・グループ ID に追加します。

addkeygroupalias -alias *aliasname* **-groupID** *groupname*

-alias
鍵の新しい*別名*。

-groupID
KeyGroup XML ファイル内のグループを識別するために使用される固有のグループ名。

例: addkeygroupalias -alias aliasname -groupID keygroup1

chgpasswd

CLI クライアントのユーザー (EKMAdmin) デフォルト・パスワードを変更します。

chgpasswd -new *password*

-new
前のパスワードを置き換える新しいパスワード。

例: chgpasswd -new ebw74jxr

createkeygroup

KeyGroups.xml ファイルに最初のキー・グループ・オブジェクトを作成します。1 度だけ実行してください。

createkeygroup -password *password*

-password
後の検索用に、KeyGroups.xml ファイル内の鍵ストアのパスワードを暗号化するために使用されるパスワード。鍵ストアはキー・グループの鍵を暗号化します。これにより、各キー・グループの別名パスワードが順に暗号化されます。そのため、KeyGroups.xml ファイル内のどの鍵も平文ではありません。

例: createkeygroup -password password

deletedrive

ドライブを Key Manager ドライブ・テーブルから削除します。同等のコマンドとして、**deldrive** および **removedrive** があります。

deletedrive -drivename *drivename*

-drivename
drivename は、削除されるドライブのシリアル番号を指定します。

例: deletedrive -drivename 000123456789

delgroupalias

キー・グループから鍵別名を削除します。

delgroupalias -groupID *groupname* **-alias** *aliasname*

-groupID
KeyGroups.xml ファイル内のグループを識別するために使用される固有のグループ名。

-alias
削除される鍵の別名。

例: delgroupalias -groupID keygroup1 -alias aliasname

delkeygroup

キー・グループ全体を削除します。

delkeygroup -groupID *groupname*

-groupID
KeyGroups.xml ファイル内のグループを識別するために使用される固有のグループ名。

例: delkeygroup -groupID keygroup1

exit

CLI クライアントを終了し、Encryption Key Manager サーバーを停止します。同等コマンドは **quit** です。

例: exit

export

ドライブ・テーブルまたは Encryption Key Manager サーバー構成ファイルを、指定の URL にエクスポートします。

export {**-drivetab**|**-config**} **-url** *urlname*

-drivetab
ドライブ・テーブルをエクスポートします。

-config
Encryption Key Manager サーバー構成ファイルをエクスポートします。

-url
urlname は、ファイルを書き込む場所を指定します。

例: export -drivetab -url FILE:///keymanager/data/export.table

help

コマンド行インターフェースのコマンド名および構文を表示します。同等コマンドは **?** です。

help

import

ドライブ・テーブルまたは構成ファイルを、指定の URL からインポートします。

import {**-merge**|**-rewrite**} {**-drivetab**|**-config**} **-url** *urlname*

-merge
新しいデータを現行データとマージします。

-rewrite
現行データを新しいデータと置き換えます。

-drivetab
ドライブ・テーブルをインポートします。

-config
構成ファイルをインポートします。

-url
urlname は、新しいデータを取り出す場所を指定します。

例: import -merge -drivetab -url FILE:///keymanager/data/export.table

list

config.keystore.file プロパティーによって指定された鍵ストア内に含まれている証明書をリストします。

list [**-cert** |**-key**|**-keysym**][**-alias** *alias* **-verbose** |**-v**]

-cert
指定の鍵ストア内の証明書をリストします。

-key
指定の鍵ストア内のすべての鍵をリストします。

-keysym
指定の鍵ストア内の対称鍵をリストします。

-alias
alias は、リストする特定の証明書を指定します。

-verbose|-v
証明書 (複数可) に関する詳細を表示します。

例:
`list -v` は、鍵ストア内のものをすべてリストします。

`list -alias mycert -v` は、mycert 別名が config.keystore.file 鍵ストア内に存在する場合はそのすべての使用可能なデータをリストします。

listcerts

config.keystore.file プロパティーによって指定された鍵ストア内に含まれている証明書をリストします。

listcerts [**-alias** *alias* **-verbose** |**-v**]

-alias
alias は、リストする特定の証明書を指定します。

-verbose|-v
証明書 (複数可) に関する詳細を表示します。

例: listcerts -alias alias1 -v

listconfig

KeyManagerConfig.properties ファイルおよび **modconfig** コマンドよって行われた更新の最新内容を反映して、メモリー内の Encryption Key Manager サーバー構成プロパティーをリストします。

listconfig

listdrives

ドライブ・テーブル内のドライブをリストします。

listdrives [**-drivename** *drivename*]

-drivename
drivename は、リストするテープ・ドライブのシリアル番号を指定します。

-verbose|-v
テープ・ドライブに関する詳細を表示します。

例: listdrives -drivename 000123456789

login

Encryption Key Manager サーバー上の CLI クライアントにサインオンします。

login -ekmuser *userID* **-ekmpassword** *password*

-ekmuser
使用される認証タイプによって、 *userID* に `EKMadmin` または localOS ユーザー ID 値を指定します (5-6 ページの『CLI クライアント・ユーザーの認証』を参照)。

-ekmpassword
ユーザー ID に対して有効なパスワード。

例: login -ekmuser EKMAdmin -ekmpassword changeME

logout

現行ユーザーをログオフします。同等コマンドは **logoff** です。これらのコマンドは、クライアント・セッションが有効な場合にのみ有用です。

例: logout

modconfig

Encryption Key Manager サーバー構成プロパティー・ファイル、KeyManagerConfig.properties のプロパティーを変更します。同等コマンドは **modifyconfig** です。

modconfig {**-set** | **-unset**} **-property** *name* **-value** *value*

-set
指定のプロパティーを、指定された値に設定します。

-unset
指定のプロパティーを除去します。

-property
name は、ターゲット・プロパティーの名前を指定します。

-value
value は、**-set** が指定された場合に、ターゲット・プロパティーの新しい値を指定します。

例: modconfig -set -property sync.timeinhours -value 24

moddrive

ドライブ・テーブル内のドライブ情報を変更します。同等コマンドは **modifydrive** です。

moddrive -drivename *drivename* {**-rec1** [*alias*] | **-rec2** [*alias*]| **-symrec** [*alias*]}

-drivename
drivename は、テープ・ドライブのシリアル番号を指定します。

-rec1
ドライブの証明書の*別名* (または鍵ラベル) を指定します。

-rec2
ドライブの証明書の 2 つ目の*別名* (または鍵ラベル) を指定します。

-symrec
テープ・ドライブに対して (対称鍵の) *別名*またはキー・グループ名を指定します。

例: moddrive -drivename 000123456789 -rec1 newalias1

refresh

Encryption Key Manager に、デバッグ、監査、およびドライブ・テーブルの値を最新の構成パラメーターで更新するように指示します。

例: refresh

refreshks

鍵ストアを更新します。これは、Encryption Key Manager サーバーの実行中に鍵ストアが変更された場合に、**config.keystore.file** に指定された鍵ストアを再ロードするのに使用します。このコマンドを使用するとパフォーマンスが低下する可能性があるため、必要な場合にのみ使用してください。

例: refreshks

status

Key Manager サーバーが始動されるのか、停止されるのかを表示します。

例: status

stopekm

Encryption Key Manager サーバーを停止します。

例: stopekm

sync

別の Encryption Key Manager サーバー上の構成ファイル・プロパティーまたはドライブ・テーブル情報 (あるいはその両方) を、コマンドを発行する Key Manager サーバーのものと同期させます。

注: 鍵ストアまたは KeyGroups.xml ファイルでは、いずれの同期化方法も機能しません。これらを手動でコピーする必要があります。

sync {**-all** | **-config** | **-drivetab**} **-ipaddr** *ip_addr* :*ssl*:*port* [**-merge** | **-rewrite**]

-all
構成プロパティー・ファイルとドライブ・テーブル情報の両方を、**-ipaddr** で指定された Encryption Key Manager サーバーに送信します。

-config
構成プロパティー・ファイルのみを、**-ipaddr** で指定された Encryption Key Manager サーバーに送信します。

-drivetab
ドライブ・テーブル情報のみを、**-ipaddr** で指定された Encryption Key Manager サーバーに送信します。

-ipaddr
ip_addr:ssl:port は、受信側の Encryption Key Manager サーバーのアドレスおよび SSL ポートを指定します。*ssl:port* は、受信側サーバーの KeyManagerConfig.properties ファイルに含まれる「TransportListener.ssl.port」に指定された値と一致していなければなりません。

-merge
新しいドライブ・テーブル・データを現行データとマージします。(構成ファイルは、常に、再書き込みです。) これはデフォルトです。

-rewrite
現行データを新しいデータと置き換えます。

例: sync -drivetab -ipaddr remoteekm.ibm.com:443 -merge

version

Encryption Key Manager サーバーのバージョンを表示します。

例: version

第 6 章 問題判別

Encryption Key Manager の個々のコンポーネント、複数のコンポーネント、またはすべてのコンポーネントに対してデバッグを使用可能に設定できます。

Encryption Key Manager サーバー問題に対する重要ファイルの確認

Encryption Key Manager が始動しない場合、問題の原因を識別するのに確認が必要なファイルが、3 つあります。

- **native_stdout.log** および **native_stderr.log**
  - Encryption Key Manager サーバーはバックグラウンド・プロセスで稼働するため、通常の情報およびエラー・メッセージを表示するコンソールはありません。このようなメッセージは、この 2 つのファイルのログに記録されます。
  - Encryption Key Manager サーバー・プロパティー・ファイルにプロパティー **debug.output.file** が含まれている場合、この 2 つのファイルは、デバッグ・ログと同じディレクトリーに作成されています。
  - Encryption Key Manager サーバー・プロパティー・ファイルにプロパティー **debug.output.file** が含まれていない場合、この 2 つのファイルは作業ディレクトリーに作成されています。
  - この 2 つのファイルは、Encryption Key Manager サーバーを始動するたびに削除および再作成されます。
- **監査ログ**
  - 監査ログには、Encryption Key Manager が処理を行っている際にログに記録されたレコードが含まれています。
  - このファイルの場所は、**KeyManagerConfig.properties**、つまりEncryption Key Manager サーバー構成プロパティー・ファイル内の、以下の 2 つのプロパティーによって指定されています。
    - Audit.handler.file.directory – 監査ログが配置されるディレクトリーを指定します
    - Audit.handler.file.name – 監査ログのファイル名を指定します
  - 監査について詳しくは、7-1 ページの『第 7 章 監査レコード』を参照してください。

127 文字より大きい鍵ストア・パスワードのログ項目

Windows サービスとして Encryption Key Manager がインストール済みで、KeyManagerConfig.properties ファイルにある鍵ストア・パスワードの長さが 128 文字以上の場合、Encryption Key Manager は、受け入れ可能な長さのパスワードをプロンプトする方法がないために、始動に失敗します。ネイティブの Encryption Key Manager ログには、以下と同様の項目が含まれています。

native_stdout.log

```
Server initialized
Default keystore failed to load
```

native_stderr.log

```
at com.ibm.keymanager.KeyManagerException: Default keystore failed to load
at com.ibm.keymanager.keygroups.KeyGroupManager.loadDefaultKeyStore(KeyGroupManager.java:145)
at com.ibm.keymanager.keygroups.KeyGroupManager.init(KeyGroupManager.java:605)
at com.ibm.keymanager.EKMServer.c(EKMServer.java:243)
at com.ibm.keymanager.EKMServer.<init>(EKMServer.java:753)
at com.ibm.keymanager.EKMServer.a(EKMServer.java:716)
at com.ibm.keymanager.EKMServer.main(EKMServer.java:129)
```

CLI クライアントと EKM サーバー間の通信問題のデバッグ

EKM CLI クライアントと EKM サーバー間の通信は、サーバーとクライアントの両方の構成プロパティー・ファイルの TransportListener.ssl.port プロパティーに指定されているポート上で行われ、SSL によって保護されます。

以下に、クライアントが EKM サーバーに接続できない場合の考えられる理由を示します。これには、問題を判別し、その問題を修正する方法を示すステップが記載されています。

- EKM サーバーが稼働していないため、クライアントには通信する対象がない。
  1. コマンド・ウィンドウから **netstat –an** を発行し、EKM サーバー・プロパティー・ファイルの TransportListener.ssl.port プロパティーおよび TransportListener.tcp.port プロパティーで指定したポートが表示されているか確認してください。ポートが表示されていない場合は、サーバーは稼働していません。
- EKM CLI クライアント・プロパティー・ファイルの TransportListener.ssl.host プロパティーが、EKM サーバーが稼働する正しいホストをポイントしない。
  1. EKM CLI クライアント・プロパティー・ファイルの TransportListener.ssl.host プロパティーの値は、`localhost` をデフォルトとします。正しいホストをポイントするように、このプロパティーの値を修正してください。
- EKM サーバーと EKM CLI クライアントが、同じポートで通信しない。
  1. EKM サーバー・プロパティー・ファイルと EKM CLI クライアント・プロパティー・ファイルの両方の TransportListener.ssl.port プロパティーを確認して、これらが同じ値に設定されていることを確認してください。
- EKM サーバーと EKM CLI クライアントが、通信を保護するために使用する共通の証明書を見つけることができない。
  1. TransportListener.ssl.keystore および TransportListener.ssl.truststore CLI クライアント・プロパティーで指定された鍵ストアに、サーバー・プロパティーの Admin.ssl.keystore 鍵ストアおよび Admin.ssl.truststore 鍵ストアと同じ証明書が含まれているか確認してください。
  2. クライアント・プロパティーの TransportListener.ssl.keystore.password に正しいパスワードが指定されているか確認してください。
  3. これらの鍵ストアの証明書が、いずれも失効していないか確認してください。JSSE は失効した証明書を使用して通信を保護しません。
- EKM CLI クライアント・プロパティー・ファイルが読み取り専用である。
  1. ファイルの属性または権限を調べて、EKM CLI クライアントを実行しているユーザーがファイルへのアクセス権限およびファイルの変更権限を持っているかを確認します。

- EKM サーバー・プロパティー・ファイルには Server.authMechanism = LocalOS と指定されているが、EKMServicesAndSamples パッケージから必要なファイルがインストールされなかったか、または間違った場所にインストールされている。
  1. 認証に関する詳細については、EKMServiceAndSamples パッケージに含まれている README を参照してください。

Key Manager サーバー問題のデバッグ

Key manager に関する問題の多くは、構成または Key manager サーバーの始動に関するものです。デバッグ・プロパティーの指定については、付録 B 『デフォルトの構成ファイル』を参照してください。

Encryption Key Manager の始動が失敗した場合は、ファイアウォールの状態を調べてください。

ソフトウェア・ファイアウォールまたはハードウェア・ファイアウォールのいずれかが、Encryption Key Manager がポートにアクセスするのを妨害している可能性があります。

EKM server not started. (EKM サーバーが始動されていません。)EKM.properties config could not be loaded or found. (EKM.properties config がロードできないか、見つかりませんでした。)

1. このエラーは、プロパティー・ファイルがデフォルトのパスに置かれていないときに **KeyManagerConfig.properties** の絶対パスを指定せずに KMSAdminCmd または EKMLaunch を開始した場合に発生します。

   Windows でのデフォルトのパスは **C:/Program Files/IBM/KeyManagerServer/**

   Linux プラットフォームのデフォルト・パスは **/opt/ibm/KeyManagerServer/**
2. コマンドを再度入力して KMSAdminCmd を開始し、**KeyManagerConfig.properties** ファイルの絶対パスを含めます。詳細については、付録 B の『Encryption Key Manager 構成プロパティー・ファイル』を参照してください。

EKM server is not started. (EKM は始動されません。) File name for XML metadata file needs to be specified in the configuration file. (構成ファイルに XMLメタデータ・ファイルのファイル名を指定する必要があります。)

Audit.metadata.file.name 項目は構成ファイルから欠落しています。

この問題を訂正するには、Audit.metadata.file.name プロパティーを **KeyManagerConfig.properties** 構成ファイルに追加してください。

Failed to start EKM.Mykeys. (EKM.Mykeys を開始できませんでした。) The system cannot find the specified file. (システムは、指定されたファイルを見つけられません。)

1. このエラー・メッセージは、**KeyManagerConfig.properties**内の鍵ストア項目が既存ファイルを指していない場合に発生します。

2. この問題を訂正するには、**KeyManagerConfig.properties** ファイル内の以下の項目が既存の有効な鍵ストア・ファイルを指していることを確認してください。

   Admin.ssl.keystore.name

   TransportListener.ssl.truststore.name

   TransportListener.ssl.keystore.name

   Admin.ssl.truststore.name

   詳細については、付録 B の『Encryption Key Manager 構成プロパティー・ファイル』を参照してください。

Failed to start EKM. (EKM を始動できませんでした。) File does not exist = safkeyring://xxx/yyy (ファイルは存在していません = safkeyring://xxx/yyy)

このエラーは、Encryption Key Manager 環境シェル・スクリプト内の IJO 変数に、誤ったプロバイダーを指定したことにより発生した可能性があります。

JCECCARACFKS 鍵ストアの場合、以下を使用してください。

```
-Djava.protocol.handler.pkgs=com.ibm.crypto.hdwrCCA.provider
```

また、JCERACFKS 鍵ストアの場合、以下を使用してください。

```
-Djava.protocol.handler.pkgs=com.ibm.crypto.provider
```

Failed to start EKM. keystore was tampered with, or password was incorrect (EKM を始動できませんでした。鍵ストアが改ざんされたか、あるいはパスワードが正しくありません)

1. このエラーは、プロパティー・ファイル (付録 B『Encryption Key Manager 構成プロパティー・ファイル』を参照) 内の以下の項目の 1 つ以上に、正しくない値が入っている場合に発生します。

   config.keystore.password (config.keystore.file に対応します)

   admin.keystore.password (admin.keystore.name に対応します)

   transportListener.keystore.password (transportListener.keystore.name に対応します)
2. このエラーは、サーバーの始動時にパスワード・プロンプトに正しくないパスワードが入力された場合にも発生します。
3. パスワードが構成に入っていない場合、プロパティー・ファイル内の 3 つの鍵ストア項目すべてが固有のものであれば、最大 3 回プロンプトが出されます。プロパティーな入出力項目がすべて同じである場合は、1 回だけプロンプトが出されます。

Failed to start EKM. (EKM を始動できませんでした。)Invalid keystore format. (鍵ストア形式が無効です。)

1. このエラーは、プロパティー・ファイル内の鍵ストア項目の 1 つに正しくない鍵ストア・タイプが指定された場合に発生する可能性があります。
2. プロパティー・ファイル内の鍵ストア項目のすべてが同じファイルを指している場合、Encryption Key Manager は、config.keystore.type 値をすべての鍵ストアの鍵ストア・タイプとして使用します。

3. 特定の鍵ストアのプロパティー・ファイル内にタイプ項目がない場合、Encryption Key Manager は、そのタイプが jceks であると想定します。

Failed to start the server. (サーバーを始動できません。) Listener thread is not up and running. (リスナー・スレッドが起動して稼働していません。)

このエラーが発生する理由は、多々考えられます。

1. **KeyManagerConfig.properties** ファイル内の以下の 2 つの項目が同じポートを指している。

   TransportListener.ssl.port

   TransportListener.tcp.port

   各トランスポート・リスナーがそれぞれ固有のポートで listen するよう構成する必要があります。
2. 上記項目の 1 つまたは両方が、Key Manager サーバーと同じマシン上で稼働している既に別のサービスで使用されているポートに対して構成されている。別のサービスが使用していないポートを見つけ、それらを使用して、Key Manager サーバーを構成してください。
3. Linux オペレーティング・システムを実行中のシステムでは、このエラーは、ポートの 1 つまたは両方が 1024 より下位であり、しかも Key Manager サーバーを始動するユーザーが root でない場合に発生する可能性があります。1024 より上のポートを使用するよう、**KeyManagerConfig.properties** 内のトランスポート・リスナー項目を変更してください。

“[Fatal Error] :-1:-1: Premature end of file.”message in native_stderr.log (「[致命的エラー] :-1:-1: ファイルの途中終了」というメッセージが native_stderr.log にあります。)

このメッセージは、Encryption Key Manager が空のキー・グループ・ファイルをロードした場合に表示されます。このメッセージは XML パーサーからのものであり、Encryption Key Manager の始動を妨げるものではありません。ただし、EKM がキー・グループを使用するように構成されており、かつ **KeyManagerConfig.properties** (Encryption Key Manager サーバー・プロパティー・ファイル) 内の config.keygroup.xml.file プロパティーが指定するファイルが破損している場合を除きます。

Error: Unable to find Secretkey in the config keystore with alias:MyKey. (エラー: 別名 MyKey を使って構成鍵ストア内で秘密鍵を見つけられません。)

プロパティー・ファイル内の symmetricKeySet 項目に、config.keystore.file に存在しない鍵別名が含まれています。

この問題を訂正するには、**KeyManagerConfig.properties** の config.keystore.file 項目によって指示された鍵ストア・ファイルに存在する別名のみが含まれるように構成ファイル内の symmetricKeySet 項目を修正するか、あるいは欠落している対称鍵を鍵ストアに追加してください。詳細については、付録 B の『Encryption Key Manager 構成プロパティー・ファイル』を参照してください。

No symmetric keys in symmetricKeySet, LTO drives cannot be supported. (symmetricKeySet に対象鍵がない場合、LTO ドライブはサポートできません。)

これは情報メッセージです。Encryption Key Manager サーバーはまだ始動しますが、この Encryption Key Manager インスタンスでは LTO ドライブをサポートできません。この Encryption Key Manager と通信するよう構成されている LTO ドライブがない場合、これは問題ではありません。

Encryption Key Manager によって報告されるエラー

ここでは、Encryption Key Manager によって報告され、ドライブ・センス・データに返されるエラー・メッセージを定義します。これらは一般に障害症状コード (FSC) と呼ばれます。 表には、エラー番号、障害の簡略説明、および修正処置が示されています。デバッグ・プロパティーの指定については、付録 B『デフォルトの構成ファイル』を参照してください。

表 6-1. Encryption Key Manager によって報告されるエラー

エラー番号	説明	処置
EE02	暗号化読み取りメッセージ障害。DriverErrorNotifyParameterError。「不正な ASC & ASCQ を受け取りました。ASC & ASCQ は、鍵作成/鍵変換/鍵取得のいずれの操作にも一致しません。」	テープ・ドライブがサポートされていない処置を要求しました。最新バージョンの Encryption Key Manager を実行していることを確認してください (最新バージョンを判別するには、 3-1 ページの『最新のバージョンの Key Manager ISO イメージのダウンロード』を参照してください)。ドライブまたはプロキシー・サーバー・ファームウェアのバージョンを調べ、必要なら、それらを最新リリースに更新してください。Key Manager サーバー上でのデバッグ・トレースを有効にします。問題の再作成を試み、デバッグ・ログを収集してください。問題が解決しない場合は、本書の最初にある『はじめにお読みください』セクションの『Dell の連絡先』で、技術支援の入手に関する情報を参照してください。

表 6-1. Encryption Key Manager によって報告されるエラー (続き)

エラー番号	説明	処置
EE0F	暗号化論理エラー。内部エラー。予期しないエラー。「EKM 内の内部プログラミング・エラー。」	最新バージョンの Encryption Key Manager を実行していることを確認してください (最新バージョンを判別するには、 3-1 ページの『最新のバージョンの Key Manager ISO イメージのダウンロード』を参照してください)。ドライブまたはプロキシー・サーバー・ファームウェアのバージョンを調べ、必要なら、それらを最新リリースに更新してください。Key Manager サーバー上でのデバッグ・トレースを有効にします。問題の再作成を試み、デバッグ・ログを収集してください。問題が解決しない場合は、本書の最初にある『はじめにお読みください』セクションの『Dell の連絡先』で、 技術支援の入手に関する情報を参照してください。
	エラー: 呼び出し CSNDDSV からのハードウェア・エラー 戻りコード 12 理由コード 0。	ハードウェア暗号方式を使用する場合は、ICSF が開始されていることを確認してください。
EE23	暗号化読み取りメッセージ障害。内部エラー。「予期しないエラー。」	ドライブまたはプロキシー・サーバーから受け取ったメッセージは、一般的なエラーのために解析できませんでした。最新バージョンの Encryption Key Manager を実行していることを確認してください (最新バージョンを判別するには、 3-1 ページの『最新のバージョンの Key Manager ISO イメージのダウンロード』を参照してください)。Key Manager サーバー上でのデバッグを有効にします。問題の再作成を試み、デバッグ・ログを収集してください。問題が解決しない場合は、本書の最初にある『はじめにお読みください』セクションの『Dell の連絡先』で、 技術支援の入手に関する情報を参照してください。

表 6-1. Encryption Key Manager によって報告されるエラー (続き)

エラー番号	説明	処置
EE25	暗号化構成上の問題。ドライブ・テーブルに関連するエラーが発生しました。	config.drivetable.file.url が指定された場合、このパラメーターが KeyManagerConfig.properties ファイル内で正しいか確認してください。Encryption Key Manager サーバー上で `listdrives -drivename <drivename>` コマンドを実行して、ドライブが正しく構成されているかどうか (例えば、ドライブ・シリアル番号、別名、および証明書が正しいか) を確認してください。最新バージョンの Encryption Key Manager を実行していることを確認してください (最新バージョンを判別するには、 3-1 ページの『最新のバージョンの Key Manager ISO イメージのダウンロード』を参照してください)。ドライブまたはプロキシー・サーバー・ファームウェアのバージョンを調べ、必要なら、それらを最新リリースに更新してください。デバッグ・トレースを有効にして、操作を再試行します。問題が解決しない場合は、本書の最初にある『はじめにお読みください』セクションの『Dell の連絡先』で、技術支援の入手に関する情報を参照してください。
EE29	暗号化読み取りメッセージ障害。無効な署名。	ドライブまたはプロキシー・サーバーから受け取ったメッセージが、その署名に一致しません。最新バージョンの Encryption Key Manager を実行していることを確認してください (最新バージョンを判別するには、 3-1 ページの『最新のバージョンの Key Manager ISO イメージのダウンロード』を参照してください)。Key Manager サーバー上でのデバッグを有効にします。問題の再作成を試み、デバッグ・ログを収集してください。問題が解決しない場合は、本書の最初にある『はじめにお読みください』セクションの『Dell の連絡先』で、 技術支援の入手に関する情報を参照してください。

表 6-1. Encryption Key Manager によって報告されるエラー (続き)

エラー番号	説明	処置
EE2B	暗号化読み取りメッセージ障害。内部エラー。「DSK 内に署名がないか、または DSK 内の署名が検証できません。」	最新バージョンの Encryption Key Manager を実行していることを確認してください (最新バージョンを判別するには、 3-1 ページの『最新のバージョンの Key Manager ISO イメージのダウンロード』を参照してください)。ドライブまたはプロキシー・サーバー・ファームウェアのバージョンを調べ、必要なら、それらを最新リリースに更新してください。Key Manager サーバー上でのデバッグ・トレースを有効にします。問題の再作成を試み、デバッグ・ログを収集してください。問題が解決しない場合は、本書の最初にある『はじめにお読みください』セクションの『Dell の連絡先』で、 技術支援の入手に関する情報を参照してください。
EE2C	暗号化読み取りメッセージ障害。QueryDSKParameterError。「デバイスからの QueryDSKMessage の解析エラー。予期しない DSK カウントまたは予期しないペイロード。」	テープ・ドライブが Encryption Key Manager に、サポートされない機能の実行を求めました。最新バージョンの Encryption Key Manager を実行していることを確認してください (最新バージョンを判別するには、 3-1 ページの『最新のバージョンの Key Manager ISO イメージのダウンロード』を参照してください)。ドライブまたはプロキシー・サーバー・ファームウェアのバージョンを調べ、必要なら、それらを最新リリースに更新してください。Key Manager サーバー上でのデバッグ・トレースを有効にします。問題の再作成を試み、デバッグ・ログを収集してください。問題が解決しない場合は、本書の最初にある『はじめにお読みください』セクションの『Dell の連絡先』で、 技術支援の入手に関する情報を参照してください。
EE2D	暗号化読み取りメッセージ障害。無効なメッセージ・タイプ	Encryption Key Manager が、順不同でメッセージを受け取ったか、取り扱い方法が不明なメッセージを受け取りました。最新バージョンの Encryption Key Manager を実行していることを確認してください (最新バージョンを判別するには、 3-1 ページの『最新のバージョンの Key Manager ISO イメージのダウンロード』を参照してください)。Key Manager サーバー上でのデバッグを有効にします。問題の再作成を試み、デバッグ・ログを収集してください。問題が解決しない場合は、本書の最初にある『はじめにお読みください』セクションの『Dell の連絡先』で、技術支援の入手に関する情報を参照してください。

表 6-1. Encryption Key Manager によって報告されるエラー (続き)

エラー番号	説明	処置
EE2E	暗号化読み取りメッセージ障害。内部エラー。無効な署名タイプ。	ドライブまたはプロキシー・サーバーから受け取ったメッセージに、有効な署名タイプがありません。最新バージョンの Encryption Key Manager を実行していることを確認してください (最新バージョンを判別するには、3-1 ページの『最新のバージョンの Key Manager ISO イメージのダウンロード』を参照してください)。Key Manager サーバー上でのデバッグを有効にします。問題の再作成を試み、デバッグ・ログを収集してください。問題が解決しない場合は、本書の最初にある『はじめにお読みください』セクションの『Dell の連絡先』で、 技術支援の入手に関する情報を参照してください。
EE30	禁止された要求。	サポートされない操作が、テープ・ドライブに対して要求されました。サポートされている正しいコマンドを、ターゲット・テープ・ドライブに対して入力してください。
EE31	暗号化構成上の問題。鍵ストアに関連するエラーが発生しました。	使用しようとする鍵ラベル、またはデフォルト用に構成された鍵ラベルを調べてください。 listcerts コマンドを使用して、Encryption Key Manager で使用可能な証明書をリストすることができます。デフォルトを使用することがわかっている場合は、Encryption Key Manager サーバーで listdrives -drivename *drivename* コマンドを実行して、ドライブが正しく構成されているか (例えば、ドライブ・シリアル番号、および関連する別名/鍵ラベルが正しいか) 検査してください。問題のドライブに別名/鍵ラベルが関連付けられていない場合は、default.drive.alias1 および default.drive.alias2 の値を確認してください。これでも解決しない、あるいは別名/鍵ラベルが存在する場合は、デバッグ・ログを収集し、本書の最初にある『はじめにお読みください』セクションの『Dell の連絡先』で、技術支援の入手に関する情報を参照してください。
EE32	鍵ストア関連の問題。	最も可能性が高い原因は、異なる鍵を持つ別の Encryption Key Manager を使用してテープが暗号化されたか、あるいはこのテープを暗号化するのに使用した鍵が名前変更または鍵ストアから削除されていたか、いずれかが考えられます。 `list -keysym` を発行し、要求の別名が鍵ストア内にあることを確認してください。

表 6-1. Encryption Key Manager によって報告されるエラー (続き)

エラー番号	説明	処置
EEE1	暗号化論理エラー。内部エラー。予期しないエラー。EK/EEDK フラグがサブページと矛盾する。	最新バージョンの Encryption Key Manager を実行していることを確認してください (最新バージョンを判別するには、 3-1 ページの『最新のバージョンの Key Manager ISO イメージのダウンロード』を参照してください)。ドライブまたはプロキシー・サーバー・ファームウェアのバージョンを調べ、必要なら、それらを最新リリースに更新してください。Key Manager サーバー上でのデバッグを有効にします。問題の再作成を試み、デバッグ・ログを収集してください。問題が解決しない場合は、本書の最初にある『はじめにお読みください』セクションの『Dell の連絡先』で、 技術支援の入手に関する情報を参照してください。
EF01	暗号化構成上の問題。「ドライブが構成されていない。」	Encryption Key Manager と通信を試みているドライブがドライブ・テーブルに存在しません。config.drivetable.file.url が指定された場合、このパラメーターが KeyManagerConfig.properties ファイル内で正しいか確認してください。`listdrives` コマンドを実行して、ドライブがリストにあるかどうか調べてください。リストにない場合は、`adddrive` コマンドを使用して、正しいドライブ情報を使ってドライブを手動で構成するか、または `modconfig` コマンドを使用して「drive.acceptUnknownDrives」プロパティーを true に設定します。デバッグ・トレースを有効にして、操作を再試行します。問題が解決しない場合は、本書の最初にある『はじめにお読みください』セクションの『Dell の連絡先』で、 技術支援の入手に関する情報を参照してください。

メッセージ

以下のメッセージは Encryption Key Manager によって生成し、管理コンソール上に表示できます。

Config File not Specified (構成ファイルが指定されていません) テキスト

`Configuration file not specified: KeyManager Configuration file not specified when starting EKM.` (構成ファイルが指定されていません。EKM の始動時に KeyManager 構成ファイルが指定されていません。)

説明

KMSAdmin コマンドでは、構成ファイルをコマンド行パラメーターとして渡す必要があります。

システムの応答

プログラムは停止します。

オペレーターの応答

構成ファイルを提供して、コマンドを再試行してください。

Failed to Add Drive (ドライブを追加できません)

テキスト

`Failed to add drive (ドライブを追加できません。)Drive already exists. (ドライブは既に存在します。)`

説明

ドライブは既に Encryption Key Manager で構成され、ドライブ・テーブルに存在するため、**adddrive** コマンドは失敗しました。

オペレーターの応答

listdrives コマンドを実行して、ドライブが Encryption Key Manager で既に構成されているかどうか確認してください。ドライブが既に存在する場合、**moddrive** コマンドを使用してドライブ構成を変更できます。詳細を入手するには、**help** を実行してください。

Failed to Archive the Log File (ログ・ファイルをアーカイブできませんでした)

テキスト

`Failed to archive the log file. (ログ・ファイルをアーカイブできませんでした。)`

説明

ログ・ファイルの名前を変更できません。

オペレーターの応答

ファイル・アクセス権と、そのドライブ上のスペースを確認してください。

Failed to Delete the Configuration (構成を削除できませんでした)

テキスト

`"modconfig" command failed. (「modconfig」コマンドが失敗しました。)`

説明

modconfig コマンドを使用して Encryption Key Manager 構成を削除できませんでした。

オペレーターの応答

help を使用してコマンド構文を調べ、提供されたパラメーターが正しいか確認してください。詳しくは、監査ログを調べてください。

Failed to Delete the Drive Entry (ドライブ項目を削除できませんでした)

テキスト

"deldrive"command failed. (「deldrive」コマンドが失敗しました。)

説明

deldrive コマンドは、ドライブ・テーブルからドライブ項目を削除できませんでした。

オペレーターの応答

help を使用してコマンド構文を調べ、提供されたパラメーターが正しいか確認してください。**listdrives** コマンドを使用して、ドライブが Encryption Key Manager で構成されているか確認してください。詳しくは、監査ログを調べてください。

Failed to Import (インポートできませんでした)

テキスト

"import" command failed. (「import」コマンドが失敗しました。)

説明

ドライブ・テーブルまたは構成ファイルをインポートできません。

システムの応答

Encryption Key Manager サーバーは始動しません。

オペレーターの応答

指定された URL が存在し、読み取り権限を持っていることを確認してください。**help** を使用してコマンド構文を確認してください。パラメーターが正しいことを確認して、再試行してください。

Failed to Modify the Configuration (構成を変更できませんでした)

テキスト

"modconfig" command failed. (「modconfig」コマンドが失敗しました。)

説明

modconfig コマンドを使用して Encryption Key Manager 構成を変更できませんでした。

オペレーターの応答

help を使用してコマンド構文を調べ、提供されたパラメーターが正しいか確認してください。詳しくは、監査ログを調べてください。

File Name Cannot be Null (ファイル名がヌルであってはなりません)

テキスト

`File name was not supplied for audit log file.` (監査ログ・ファイルにファイル名が提供されていません。)

説明

監査ファイル名が、Encryption Key Manager の構成プロパティーを介して指定されていません。このパラメーターは、必須の構成パラメーターです。

システムの応答

プログラムは停止します。

オペレーターの応答

プロパティー `Audit.handler.file.name` が、Encryption Key Manager に対して提供された構成プロパティー・ファイル内で定義されているか確認し、再始動を試みてください。

File Size Limit Cannot be a Negative Number (ファイル・サイズの限度に負の数値は使用できません)

テキスト

`Maximum file size for audit log can not be a negative number.` (監査ログの最大ファイル・サイズが負の数値であってはなりません。)

説明

Encryption Key Manager 構成ファイル内の `Audit.handler.file.size` プロパティー値は、正数でなければなりません。

システムの応答

Encryption Key Manager は始動しません。

オペレーターの応答

`Audit.handler.file.size` に有効な数値を指定して、Encryption Key Manager の再始動を試みてください。

No Data to be Synchronized (同期するデータがありません)

テキスト

No data can be found to be synchronized with "sync". (「sync」で同期されるデータが見つかりません。)

説明

sync コマンドは、同期されるデータを識別できません。

オペレーターの応答

指定された構成ファイルが存在しており、ドライブ・テーブルが *config.drivetable.file.url* を使用して構成ファイルで正しく構成されているかどうかを確認してください。**help** を使用して構文を確認し、**sync** コマンドを使用して再試行してください。

Invalid Input (無効な入力)

テキスト

Invalid input parameters for the CLI. (CLI の入力パラメーターが無効です。)

説明

特定のコマンド構文が正しくない可能性があります。

オペレーターの応答

入力されたコマンドが正しいことか確認してください。**help** を使用してコマンド構文を確認してください。提供されたパラメーターが正しいことを確認して、再試行してください。

Invalid SSL Port Number in Configuration File (構成ファイル内の SSL ポート番号が無効です)

テキスト

Invalid SSL port number in config file. (構成ファイル内の SSL ポート番号が無効です。)

説明

構成ファイルに提供された SSL ポート番号は有効な数値ではありません。

システムの応答

Encryption Key Manager は始動しません。

オペレーターの応答

Encryption Key Manager を始動する際に、構成ファイルに TransportListener.ssl.port プロパティーに有効なポート番号を指定して、再始動を試みてください。

Invalid TCP Port Number in Configuration File (構成ファイル内の TCP ポート番号が無効です)

テキスト

Invalid TCP port number specified in the EKM configuration file. (構成ファイル内の TCP ポート番号が無効です。)

説明

構成ファイルに提供された TCP ポート番号は有効な数値ではありません。

システムの応答

Encryption Key Manager は始動しません。

オペレーターの応答

Encryption Key Manager を始動する際に、構成ファイルに TransportListener.tcp.port プロパティーに有効なポート番号を指定して、再始動を試みてください。デフォルトの TCP ポート番号は 3801 です。

Must Specify SSL Port Number in Configuration File (構成ファイルに SSL ポート番号を指定する必要があります)

テキスト

SSL port number is not configured in the properties file. (プロパティー・ファイルに SSL ポート番号が構成されていません。)

説明

SSL ポート番号は、構成プロパティー・ファイルに構成する必須プロパティーです。これは、マルチサーバー環境で Encryption Key Manager サーバー間の通信に使用されます。

システムの応答

Encryption Key Manager は始動しません。

オペレーターの応答

TransportListener.ssl.port プロパティーに有効なポート番号を指定して、Encryption Key Manager の再始動を試みてください。

Must Specify TCP Port Number in Configuration File (構成ファイルに TCP ポート番号を指定する必要があります)

テキスト

TCP port number is not configured in the properties file. (プロパティー・ファイルに TCP ポート番号が構成されていません。)

説明

TCP ポート番号は、構成プロパティー・ファイルに構成する必須プロパティーです。これは、ドライブと Encryption Key Manager 間の通信に使用されます。

システムの応答

Encryption Key Manager は始動しません。

オペレーターの応答

TransportListener.tcp.port プロパティーに有効なポート番号を指定して、Encryption Key Manager の再始動を試みてください。デフォルトの TCP ポート番号は 3801 です。

Server Failed to Start (サーバーは始動できませんでした)

テキスト

EKM server failed to start. (EKM サーバーが始動できませんでした。)

説明

構成に問題があるため、Encryption Key Manager サーバーは始動できません。

オペレーターの応答

提供された構成ファイル内のパラメーターを確認してください。詳しくは、ログを調べてください。

Sync Failed (同期できませんでした)

テキスト

"sync" command failed. (「sync」コマンドが失敗しました)

説明

2 つの Encryption Key Manager サーバー間でデータを同期するための sync 操作が失敗しました。

オペレーターの応答

リモート Encryption Key Manager サーバーに対して指定された IP アドレスが正しく、そのコンピューターがアクセス可能であることを確認してください。構成ファイルが存在しており、正しいドライブ・テーブル情報が入っていることを確認して

ください。**help** を使用してコマンド構文を確認し、**sync** コマンド構文を調べてください。詳しくは、ログを調べてください。

The Specified Audit Log File is Read Only (指定の監査ログ・ファイルは読み取り専用です)

テキスト

`The audit log file can not be opened for writing.` (監査ログ・ファイルを書き込み用に開けません。)

説明

プロパティー `Audit.handler.file.name` で指定された Encryption Key Manager 構成内の監査ログ・ファイルを書き込み用に開くことができません。

システムの応答

Encryption Key Manager は始動しません。

オペレーターの応答

指定の監査ファイルおよびディレクトリーに対する権限を確認してから、Encryption Key Manager を再始動してみてください。

Unable to Load the Admin Keystore (管理鍵ストアをロードできません)

テキスト

`Keystore for Admin cannot be loaded.` (管理のための鍵ストアをロードできません)

説明

Encryption Key Manager に提供された管理鍵ストアをロードできません。管理鍵ストアは、マルチサーバー環境でのサーバー・サイド通信用に Encryption Key Manager サーバー間で使用されます。

システムの応答

Encryption Key Manager は始動しません。

オペレーターの応答

構成ファイル・セットアップを確認します。Encryption Key Manager 構成ファイル内のプロパティー `admin.keystore.file`、`admin.keystore.provider`、および `admin.keystore.type` が正しく (付録 B を参照)、鍵ストア・ファイルが既に存在し、読み取り権限を持っていることを確認してください。`admin.keystore.password` プロパティーを介して鍵ストアに提供されたか、またはコマンド行に入力されたパスワードが正しいか確認してください。Encryption Key Manager の再始動を試行します。

Unable to load the keystore (鍵ストアをロードできません)

テキスト

Keystore for EKM can not be loaded. (EKM の鍵ストアをロードできません。)

説明

Encryption Key Manager に対して指定された鍵ストアをロードできません。

システムの応答

Encryption Key Manager は始動しません。

オペレーターの応答

構成ファイル・セットアップを確認します。Encryption Key Manager 構成ファイル内のプロパティー config.keystore.file、config.keystore.provider、および config.keystore.type が正しく、鍵ストア・ファイルが既に存在し、読み取り権限を持っていることを確認してください。config.keystore.password プロパティーを介して Encryption Key Manager 鍵ストアに提供されたか、またはコマンド行に入力されたパスワードが正しいか確認してください。再始動を試みてください。

Unable to Load the Transport Keystore (移送鍵ストアをロードできません)

テキスト

Transport keystore cannot be loaded. (移送鍵ストアをロードできません。)

説明

Encryption Key Manager に提供された移送鍵ストアをロードできません。移送鍵ストアは、マルチサーバー環境でのクライアント・サイド通信用に Encryption Key Manager サーバー間で使用されます。

システムの応答

Encryption Key Manager は始動しません。

オペレーターの応答

構成ファイル・セットアップを確認します。Encryption Key Manager 構成ファイル内のプロパティー transport.keystore.file、transport.keystore.provider、および transport.keystore.type が正しく、鍵ストア・ファイルが存在し、読み取り権限を持っていることを確認してください。transport.keystore.password プロパティーを介して管理鍵ストアに提供されたか、またはコマンド行に入力されたパスワードが正しいか確認してください。Encryption Key Manager の再始動を試行します。

サポートされないアクション

テキスト

User entered action for the CLI which is not supported for EKM. (ユーザーが、EKM についてサポートされていない CLI にアクションを入力しました。)

説明

sync コマンドに提供されたアクションは、Encryption Key Manager がサポートしていないか、または理解していません。有効なアクションは、merge (マージ) または rewrite (再書き込み) です。

オペレーターの応答

help を使用してコマンド構文を確認し、再試行してください。

第 7 章 監査レコード

注: この章で説明している監査レコード・フォーマットは、プログラミング・インターフェースとみなされません。これらのレコードのフォーマットは、リリースによって異なる場合があります。フォーマットについては、監査レコードの構文解析が必要な場合に、この章で説明しています。

監査の概要

監査サブシステムは、Encryption Key Manager による要求の処理中に各種の監査可能イベントが発生したときに、テキストによる監査レコードを一連の順次ファイルに書き込みます。監査サブシステムは、ファイルに対して書き込みます (ディレクトリーおよびファイルの名前は構成可能です)。これらのファイルのファイル・サイズも構成可能です。レコードがファイルに書き込まれ、ファイルのサイズが構成可能なサイズに達すると、そのファイルはクローズされて、現在タイム・スタンプに基づいて名前変更されると、別のファイルがオープンされ、新たに作成されたファイルにレコードが書き込まれます。監査レコードのログ全体は、このようにして、構成可能なサイズのファイルに分けられ、それぞれの名前は、ファイルのサイズが構成可能なサイズを超えた時点のタイム・スタンプで順序付けられます。

監査ログ全体 (作成されたすべての順次ファイルに及びます) の情報量が大きくなりすぎて、ファイル・システムで使用可能なスペースを超えないようにしておくために、構成済みの監査ディレクトリー/フォルダー/コンテナー内のファイル・セットをモニターするスクリプトまたはプログラムを作成する必要があります。ファイルがクローズされ、タイム・スタンプに基づいて名前が付けられたら、ファイルの内容をコピーし、必要な長期連続ログの場所に付加してから、クリアする必要があります。実行中の Encryption Key Manager によってレコードが書き込まれているファイルを、除去または変更しないように注意してください (このファイルには、ファイル名にタイム・スタンプがありません)。

監査構成パラメーター

監査ログに記録されるイベント、監査ログ・ファイルの書き込み先、および監査ログ・ファイルの最大サイズを制御するのに、以下のパラメーターが Encryption Key Manager の構成ファイル内で使用されます。

Audit.event.types

構文

Audit.event.types={*type*[;*type*]}

使用法

監査ログに送信される監査タイプを指定するのに使用されます。構成パラメーターの可能な値は、次のとおりです。

all	すべてのイベント・タイプ
authentication	認証イベント
data_synchronization	Encryption Key Manager サーバー間での情報の同期中に発生するイベント
runtime	Encryption Key Manager に送信される処理動作および要求の一部として発生するイベント
configuration_management	構成変更が行われるときに発生するイベント
resource_management	Encryption Key Manager のリソース (テープ・ドライブ) 設定が変更されるときに発生するイベント

例

この構成値の指定例は、次のとおりです。

```
Audit.event.types=all
```

別の例として、次のものもあります。

```
Audit.event.types=authentication;runtime;resource_management
```

Audit.event.outcome

構文

Audit.event.outcome={*outcome*[;*outcome*]}

使用法

正しく実行された動作または正しく実行されなかった動作、あるいはその両方の結果生じたイベントを監査対象にするかどうかを指示するのに使用されます。動作が正しく実行された結果生じる、ログに記録されるイベントについて **success** を指定します。動作が正しく実行されなかった結果生じる、ログに記録されるイベントについては **failure** を指定します。

例

この構成値の指定例は、次のとおりです。

```
Audit.event.outcome=failure
```

正常なケースと失敗したケースの両方を有効にするには、次のようにします。

```
Audit.event.outcome=success;failure
```

Audit.eventQueue.max

構文

Audit.eventQueue.max=*number_events*

使用法

メモリー・キュー内に保持されるイベント・オブジェクトの最大数を設定するのに使用されます。このパラメーターはオプションですが、指定することをお勧めします。デフォルトはゼロです。

例

```
Audit.eventQueue.max=8
```

Audit.handler.file.directory

構文

Audit.handler.file.directory=*directoryName*

使用法

このパラメーターは、監査レコード・ファイルを書き込むディレクトリーを指示するのに使用されます。指定したディレクトリーが存在しない場合、Encryption Key Manager はそのディレクトリーの作成を試行することに注意してください。ただし、正常に作成できない場合、Encryption Key Manager は始動されません。Encryption Key Manager を実行する前に、目的のディレクトリーが存在しているようにしてください。Encryption Key Manager が実行されるユーザー ID が、指定されたディレクトリーに対して書き込みアクセスを持っていることも必要です。

例

ディレクトリーを **/var/ekm/ekm1/audit** に設定するには、次のように指定します。

```
Audit.handler.file.directory=/var/ekm/ekm1/audit
```

Audit.handler.file.size

構文

Audit.handler.file.size=*sizeInKiloBytes*

使用法

このパラメーターは、監査ファイルがクローズされ、新しい監査ファイルが書き込まれるサイズ限度を示すのに使用されます。結果として生じる監査ファイルの実際のサイズは、この値を数バイト上回る場合があることに注意してください。それは、サイズ限度に達した後でファイルがクローズされるためです。

例

最大ファイル・サイズをおおよそ 2 メガバイトに設定するには、次のように入力します。

```
Audit.handler.file.size=2000
```

Audit.handler.file.name

構文

Audit.handler.file.name=*fileName*

使用法

このパラメーターは、監査ログ・ファイルの作成時に基底名として使用する指定の監査ディレクトリー内で、基本ファイル名を指定するのに使用します。このパラメ

ーターは、完全修飾パス名ではなく、基本ファイル名のみを含む必要があることに注意してください。監査ログ・ファイルの絶対パス名では、そのファイルが書き込まれた時間に相当する値がこの名前に付加されます。

これを示すために、Audit.handler.file.name の値が **ekm.log** に設定されている例を考えてみます。ファイルの絶対パス名は、ekm.log.2315003554 のようなものになります。付加されたストリングは、監査ログ・ファイルが作成された順序を判別するのに役立ちます。数値が大きいほど、新しい監査ログ・ファイルであることを表します。

例

基底名を **ekm.log** に設定する例は、次のとおりです。

```
Audit.handler.file.name=ekm.log
```

Audit.handler.file.multithreads

構文

Audit.handler.file.multithreads={yes|true|no|false}

使用法

true と指定された場合、イベント・データを監査ログに書き込むために別のスレッドが使用され、監査ログの完了を待たずに実行 (動作) の現行スレッドが続行できるようになります。複数のスレッドの使用は、デフォルトの動作です。

例

基底名を **true** に設定する例は、次のとおりです。

```
Audit.handler.file.multithreads=true
```

Audit.handler.file.threadlifespan

構文

Audit.handler.file.threadlifespan=*timeInSeconds*

使用法

このパラメーターは、監査ログ項目を書き込むのにスレッドが必要とすると予想される最大時間を指定するのに使用されます。この値がクリーンアップ処理中に使用されると、スレッドは、中断される前に処理を完了することができます。バックグラウンド・スレッドが threadlifespan パラメーターによって割り当てられた時間内に処理を完了しなかった場合、終結処理時にそのスレッドは中断されます。

例

スレッドが監査ログを書き込む予想時間を 10 秒に設定するには、次のように指定します。

```
Audit.handler.file.threadlifespan=10
```

監査レコード・フォーマット

すべての監査レコードは、ここで説明されている類似の出力フォーマットを使用します。すべての監査レコードには、発生した監査イベントに固有の情報と共に、タイム・スタンプおよびレポート・タイプを含む共通情報が入っています。監査レコードの一般的なフォーマットを以下に示します。

```
AuditRecordType:[
  timestamp=timestamp
  Attribute Name=Attribute Value
  ...
  ]
```

各レコードは、ファイル内で複数の行に及びます。レコードの最初の行は、その行の最初の文字から始まる監査レコード・タイプで始まり、その後にコロン (;) と、左大括弧 ([) が続きます。同じ監査レコードに関連する後続行は、ログ・レコードを読みやすくするために、スペース 2 つ分字下げされます。1 つの監査レコードの最後の行には、右大括弧 (]) がスペース 2 つ分字下げされて入ります。各監査レコードの行数は、監査レコード・タイプと、監査レコードと一緒に提供される追加の属性情報により異なります。

監査レコードのタイム・スタンプは、Encryption Key Manager が稼働しているシステムのシステム・クロックに基づきます。これらのレコードを、タイム・スタンプに基づいて、他のシステム上で発生するイベントと相互に関連付ける場合、環境内の各種システムのクロックが許容可能なレベルの正確さに同期されるように、なんらかのタイプの時刻同期を使用する必要があります。

Encryption Key Manager 内の監査ポイント

Encryption Key Manager は、要求の処理中に発生する多くのイベントについて、構成に基づいて監査レコードを書き込むことができます。この項では、監査可能なイベントのセットについて、監査レコード構成カテゴリーと共に説明します。これらの監査レコードが監査ファイルに書き込まれるためには、このカテゴリーが有効でなければなりません (表 7-1 を参照)。

表 7-1. Encryption Key Manager が監査ファイルに書き込む監査レコード・タイプ

監査レコード・タイプ	監査タイプ	説明
認証	authentication	認証イベントをログに記録するのに使用されます
データ同期	data_synchronization	データ同期処理をログに記録するのに使用されます
ランタイム	runtime	要求の処理中に Encryption Key Manager サーバー内で発生する各種の重要な処理イベントをログに記録するのに使用されます
リソース管理	resource_management	Encryption Key Manager に対するリソースの構成方法への変更をログに記録するのに使用されます

表 7-1. *Encryption Key Manager* が監査ファイルに書き込む監査レコード・タイプ *(続き)*

監査レコード・タイプ	監査タイプ	説明
構成管理	configuration_management	Encryption Key Manager サーバーの構成への変更をログに記録するのに使用されます

監査レコード属性

以下のリストは、各監査レコード・タイプに使用できる属性を示しています。

認証イベント

これらのレコードのフォーマットは、次のとおりです。

```
Authentication event:[
  timestamp=timestamp
  event source=source
  outcome=outcome
  event type=SECURITY_AUTHN
  message=message
  authentication type=type
  users=users
  ]
```

message 値は、それのための情報が使用可能な場合にのみ表示されることに注意してください。

データ同期イベント

これらのレコードのフォーマットは、次のとおりです。

```
Data synchronization event:
  timestamp=timestamp
  event source=source
  outcome=outcome
  event type=SECURITY_DATA_SYNC
  message=message
  action=action
  resource=resource
  user=user
  ]
```

message 値および user 値は、それらのための情報が使用可能な場合にのみ表示されることに注意してください。

ランタイム・イベント

これらのレコードのフォーマットは、次のとおりです。

```
Runtime event:
  timestamp=timestamp
  event source=source
  outcome=outcome
  event type=SECURITY_RUNTIME
  message=message
  resource=resource
  action=action
  user=user
  ]
```

message 値および user 値は、それらのための情報が使用可能な場合にのみ表示されることに注意してください。

リソース管理イベント

これらのレコードのフォーマットは、次のとおりです。

```
Resource management event:
  timestamp=timestamp
  event source=source
  outcome=outcome
  event type=SECURITY_MGMT_RESOURCE
  message=message
  action=action
  user=user
  resource=resource
  ]
```

message 値は、それのための情報が使用可能な場合にのみ表示されることに注意してください。

構成管理イベント

これらのレコードのフォーマットは、次のとおりです。

```
Configuration management event:
  timestamp=timestamp
  event source=source
  outcome=outcome
  event type=SECURITY_MGMT_CONFIG
  message=message
  action=action
  command type=type
  user=user
  ]
```

message 値は、それのための情報が使用可能な場合にのみ表示されることに注意してください。

監査対象イベント

表 7-2 は、監査レコードが作成される原因となるイベントについて説明しています。この表には、以下のイベントが発生したときにログに記録される監査レコード・タイプをリストします。

表 7-2. 監査対象イベント別の監査レコード・タイプ

監査対象イベント	監査レコード・タイプ
ユーザーが正常に認証された	authentication
ユーザー認証が失敗した	authentication
データが他の EKM に正常に送信された	data_synchronization
他の EKM へのデータの送信中にエラーが発生した	data_synchronization
sync コマンドが処理された	data_synchronization
sync コマンドの処理中にエラーが発生した	data_synchronization
コマンド行処理が開始された	runtime
exit コマンドを受信した	runtime

表 7-2. 監査対象イベント別の監査レコード・タイプ (続き)

監査対象イベント	監査レコード・タイプ
不明のコマンドが入力された	runtime
ドライブがメッセージを受信した	runtime
ドライブからのメッセージの処理中にエラーが発生した	runtime
ドライブから受け取ったメッセージからのエラー	runtime
ドライブから受け取った情報でドライブ・テーブルを更新中にエラーが発生した	runtime
ドライブ・テーブルから情報を取り出しているときにエラーが発生した	runtime
鍵ストアから情報を取り出しているときにエラーが発生した	runtime
鍵ストアからの証明の処理中にエラーが発生した	runtime
鍵ストアから秘密鍵を検索中にエラーが発生した	runtime
暗号値を計算中にエラーが発生した	runtime
メッセージ交換が正常に処理された	runtime
メッセージ処理が開始された	runtime
コマンド行処理が開始された	runtime
暗号サービスを使用中に問題が見つかった	runtime
新しいドライブが発見された	runtime
ドライブ・テーブルに対してドライブを構成中にエラーが発生した	runtime
ドライブからのメッセージの処理が正常に開始された	runtime
stopekm コマンドを受け取り、処理した	runtime
ドライブがドライブ・テーブルから除去された	resource_management
ドライブをドライブ・テーブルから除去中にエラーが発生した	resource_management
ドライブ・テーブルが正常にインポートされた	resource_management
ドライブ・テーブルのインポート中にエラーが発生した	resource_management
ドライブ・テーブルが正常にエクスポートされた	resource_management
ドライブ・テーブルのエクスポート中にエラーが発生した	resource_management
listcerts コマンドが正常に実行された	resource_management
ドライブがドライブ・テーブルに正常に追加された	resource_management

表 7-2. 監査対象イベント別の監査レコード・タイプ (続き)

監査対象イベント	監査レコード・タイプ
ドライブをドライブ・テーブルに追加中にエラーが発生した	resource_management
listdrives コマンドが正常に実行された	resource_management
listdrives コマンドの処理中にエラーが発生した	resource_management
ドライブ・テーブルが正常に変更された	resource_management
ドライブ・テーブルの変更中にエラーが発生した	resource_management
KeyStore のオープンが正常に実行された	resource_management
KeyStore のオープン中にエラーが発生した	resource_management
構成プロパティーが変更された	configuration_management
構成プロパティーの変更中にエラーが発生した	configuration_management
構成プロパティーが削除された	configuration_management
構成プロパティーの削除中にエラーが発生した	configuration_management
構成が正常にインポートされた	configuration_management
構成のインポート中にエラーが発生した	configuration_management
構成が正常にエクスポートされた	configuration_management
構成のエクスポート中にエラーが発生した	configuration_management
listconfig コマンドが正常に実行された	configuration_management

第 8 章 メタデータの使用

Encryption Key Manager は、データが暗号化されてテープに書き込まれるときに重要情報を取り込む XML ファイルを作成するよう構成する必要があります。このファイルをボリューム通し番号で照会すると、そのボリュームで使用された別名または鍵ラベルを表示できます。逆に、ファイルを別名で照会すると、その鍵ラベル/別名と関連付けられたすべてのボリュームを表示できます。

注: メタデータ・ファイルを構成しない場合、Encryption Key Manager は始動しません。

暗号化処理の実行に合わせて、Encryption Key Manager は以下のデータを収集します。

- Drive Serial Number (ドライブのシリアル番号)
- Drive WorldWideName (ドライブ World Wide Name)
- Creation Date (作成日)
- Key Alias 1 (鍵別名 1)
- Key Alias 2 (鍵別名 2)
- DKi
- VolSer

収集されたデータが一定の限度に達すると、XML ファイルに書き込まれます。デフォルトの限度は (Encryption Key Manager プロパティー・ファイル (KeyManagerConfig.properties) で設定可能) 100 個のレコードです。このファイルは一度書き込まれると、Encryption Key Manager が実行している限り、照会することができます。ファイルが大きくなりすぎないようにするために、最大ファイル・サイズに達する前に、新しいファイルに自動的にロールオーバーされます。ロールオーバーのデフォルトの最大ファイル・サイズ限度は (Encryption Key Manager プロパティー・ファイルで設定可能)、1 MB です。保管されるのは、現行ファイル・バージョンと直前のファイル・バージョンだけです。Encryption Key Manager 構成プロパティー・ファイルに設定する値は、次のものです。

Audit.metadata.file.name
メタデータが保管される XML ファイルの名前。これは必須です。

Audit.metadata.file.size
ファイルを現行バージョンから直前バージョンにローリングする前の、キロバイト単位で指定される最大ファイル・サイズ。これはオプションです。デフォルトは 1024 (1MB) です。

Audit.metadata.file.cachecount
メタデータ・ファイルを書き込む前にキャッシュに入れられるレコードの数。これはオプションです。デフォルトは 100 です。

XML ファイル・フォーマット

このファイルには、以下のフォーマットでレコードが入っています。

```
<KeyUsageEvent>
 <DriveSSN>FVTDRIVE0000</driveSSN>          -Drive Serial Number
 <VolSer>TESTER</volSer>                    -Volume Serial
 <DriveWWN>57574E414D453030</driveWWN>      -drive WWN
 <keyAlias2>cert2</keyAlias2>               -Key Alias1
 <keyAlias1>cert1</keyAlias1>               - keyAlias2
 <dateTime>Tue Feb 20 09:18:07 CST 2007</dateTime>   - creation date
</KeyUsageEvent>
```

注: LTO 4 および LTO 5 ドライブの場合、存在するのは <keyAlias1></keyAlias1> レコードのみであり、DKi は記録されません。

メタデータ XML ファイルの照会

メタデータ・ファイルを照会するには、EKMDataParser ツールを使用します。このツールは、Document Object Model (DOM) 技法を使用して XML ファイルを解析しますが、Encryption Key Manager コマンド行インターフェースからは実行できません。このツールは、次のように呼び出します。

java com.ibm.keymanager.tools.EKMDataParser -filename *full_path_to_metadata_file* {**-volser** *volser* | **-keyalias** *alias*}

metadata_path
これは、**KeyManagerConfig.properties** ファイル内の Audit.metadata.file.name のメタデータ・ファイルに対して指定されるのと同じディレクトリー・パスです。

-filename
filename は必須であり、XML メタデータ・ファイルの名前でなければなりません。これは、通常、**KeyManagerConfig.properties** ファイル内の Audit.metadata.file.name プロパティーに指定された名前と同じものです。

-volser
XML ファイル内で探しているテープ・カートリッジのボリューム通し番号。**-volser** または **-keyalias** を指定する必要があります。

-keyalias
XML ファイル内で探している鍵ラベルまたは別名。**-volser** または **-keyalias** を指定する必要があります。

例

KeyManagerConfig.properties 内のメタデータ・ファイル名プロパティー (Audit.metadata.file.name) が値 `metadata` に設定されており、ファイルは Encryption Key Manager が実行するローカル・ディレクトリーにあるとした場合、以下のコマンドは volser `72448` に関連する XML レコードのみをフィルタリング (表示) します。

```
<jvm_path>/bin/java com.ibm.keymanager.tools.EKMDataParser -filename metadata -volser 72448
```

出力のフォーマットは、次のようになります。

表 8-1. メタデータの照会出力形式

keyalias1	keyalias2	volSer	dateTime	driveSSN	dki

表 8-1. メタデータの照会出力形式 (続き)

cert1 cert2 72448 Wed Mar 14 10:31:32 CDT 2007 FVTDRIVE0004

破損したメタデータ・ファイルからのリカバリー

Encryption Key Manager が適切にシャットダウンされなかった場合、または Encryption Key Manager が実行されているシステムが異常終了した場合、Encryption Key Manager メタデータ・ファイルが破損する可能性があります。編集または変更を適切に行わなかった場合にも、メタデータ・ファイルが破損することがあります。この破損は、EKMDataParser がメタデータ・ファイルの構文解析を行うまで認識されません。 EKMDataParser が、以下のようなエラーで失敗することがあります。

```
[Fatal Error] EKMData.xml:290:16: The end-tag for element type "KeyUsageEvent" must
 end with a '>' delimiter.
org.xml.sax.SAXParseException: The end-tag for element type "KeyUsageEvent" must
 end with a '>' delimiter.
at org.apache.xerces.parsers.DOMParser.parse(Unknown Source)
at org.apache.xerces.jaxp.DocumentBuilderImpl.parse(Unknown Source)
at javax.xml.parsers.DocumentBuilder.parse(Unknown Source)
at com.ibm.keymanager.tools.EKMDataParser.a(EKMDataParser.java:136)
at com.ibm.keymanager.tools.EKMDataParser.a(EKMDataParser.java:26)
at com.ibm.keymanager.tools.EKMDataParser.main(EKMDataParser.java:93)
```

このエラーが発生した場合、原因はエレメントの XML 終了タグの欠落です。Encryption Key Manager メタデータ・ファイルをリカバリーし、EKMDataParser が再びファイルの構文解析をすることが可能です。

1. Encryption Key Manager メタデータ・ファイルのバックアップ・コピーを作成します。
2. Encryption Key Manager メタデータ・ファイルを編集します。
3. XML には、データまたはイベントの各部分に、開始タグと対応する終了タグがあるはずです。
   - 開始タグの例は、以下のようになります。
     - <KeyUsageEvent>
     - <driveSSN>
     - <keyAlias1>
   - 終了タグの例は、以下のようになります。
     - </KeyUsageEvent>
     - </driveSSN>
     - </keyAlias1>
4. ファイルをスキャンし、一致していないタグがないか確認します。EKMDataParser のエラー・メッセージにより、終了タグが欠落したタグがリストされます。これにより、検索がいくらか容易になるはずです。
5. 一致していないタグが検出されたら、一時的にイベントを削除するか、またはイベントを補完するのに必要なタグを追加します。
   - 例えば、以下の Encryption Key Manager メタデータ・ファイルの抜粋では、終了タグが欠落した最初の KeyUsageEvent を表示しています。

```
<KeyUsageEvent>
<driveSSN>001310000109</driveSSN>
<volSer>      </volSer>
<driveWWN>5005076312418B07</driveWWN>
<keyAlias1>key0000000000000000F</keyAlias1>
<dki>6B65790000000000000000F</dki>
<dateTime>Thu Aug 30 09:50:53 MDT 2007</dateTime>
<KeyUsageEvent>
<driveSSN>001310000100</driveSSN>
<volSer>      </volSer>
<driveWWN>5005076312418ABB</driveWWN>
<keyAlias1>key00000000000000000</keyAlias1>
<dki>6B6579000000000000000000</dki>
<dateTime>Thu Sep 06 16:49:39 MDT 2007</dateTime>
</KeyUsageEvent>
```

<dateTime>Thu Aug 30 09:50:53 MDT 2007</dateTime> と <KeyUsageEvent> の行の間に </KeyUsageEvent> を追加すると、最初の <KeyUsageEvent> が完全なものになります。

ファイルの破損を修復すると、EKMDataParser がデータの構文解析を正常に行うことができます。

付録 A. サンプル・ファイル

サンプル始動デーモン・スクリプト

重要: ご自分の鍵ストア・データを保持することの重要性は、どれほど強調しても強調しすぎることはありません。ご自分の鍵ストアにアクセスできなければ、暗号化されたテープを暗号化解除することはできません。鍵ストアおよびパスワード情報は、必ず保存してください。

Linux プラットフォーム

以下に、実績のある方法で EKM がバックグラウンドで開始されるようにするスクリプトのサンプルを示します。このスクリプトは、EKM を開始すると、スクリプトを通じて鍵ストア・パスワード *keystore_password* を引き渡します。この方法では、鍵ストア・パスワードが EKM 構成ファイルに入っている必要はありません (以下を参照)。スクリプト・ファイルには、次のものが入っているはずです。

```
java com.ibm.keymanager.KMSAdminCmd KeyManagerConfig.properties <<EOF
startekm
keystore_password
status
EOF
```

注: 鍵ストア・パスワードがスクリプトを通じて EKM に入っている (すなわち、EKM 構成ファイルに鍵ストア・パスワードは含まれていません) 場合、ファイル (構成ファイル、ドライブ・テーブル、および鍵ストア・バックアップ・ファイル) を秘密として取り扱う必要はありませんが、鍵ストア・パスワードが含まれているスクリプトは、安全かつ弾力的に (例えば、複数の場所に複数のコピー) 保管する**必要があります**。鍵ストア・パスワードは、機密情報であるため、そのように扱う必要があります。スクリプト・ファイルを安全にバックアップするには、鍵ストア・パスワードが入っている構成ファイルをバックアップする場合と同じオプションがあります。しかし、スクリプトは、安全に、しかも EKM バックアップ・ファイルとは別個にバックアップされ、保管/送信されます。これにより、セキュリティーに役立つレベルの間接指示が追加されます。最後に、鍵ストア・パスワードは (スクリプトまたは EKM の構成ファイルに) 保管されますが、鍵ストア・パスワードが必ずリカバリーできるように、安全かつ弾力的に保管する必要があることを強調しておきます。**鍵ストア・パスワードのすべてのコピーが失われると、鍵ストア内のすべての鍵が失われるため、これをリカバリーする手だてはありません。**

構成ファイルのサンプル

以下に、すべての鍵ストア項目が同じソフトウェア鍵ストアを指している EKM プロパティー・ファイルのサンプルを示します。

```
Admin.ssl.keystore.name = /keymanager/testkeys
Admin.ssl.keystore.type = jceks
Admin.ssl.truststore.name = /keymanager/testkeys
Admin.ssl.truststore.type = jceks
Audit.event.outcome = success,failure
Audit.event.types = all
Audit.eventQueue.max = 0
Audit.handler.file.directory = /keymanager/audit
Audit.handler.file.name = kms_audit.log
Audit.handler.file.size = 10000
Audit.metadata.file.name = /keymanager/metafile.xml
config.drivetable.file.url = FILE:///keymanager/drivetable
config.keystore.file = /keymanager/testkeys
config.keystore.provider = IBMJCE
config.keystore.type = jceks
fips = Off
TransportListener.ssl.ciphersuites = JSSE_ALL
TransportListener.ssl.clientauthentication = 0
TransportListener.ssl.keystore.name = /keymanager/testkeys
TransportListener.ssl.keystore.type = jceks
TransportListener.ssl.port = 443
TransportListener.ssl.protocols = SSL_TLS
TransportListener.ssl.truststore.name = /keymanager/testkeys
TransportListener.ssl.truststore.type = jceks
TransportListener.tcp.port = 3801
```

以下は、すべての鍵ストア項目が異なるソフトウェア鍵ストアを指している EKM プロパティー・ファイルのサンプルです。太字で示された項目が、上記の最初のサンプル・プロパティー・ファイルと異なります。

```
Admin.ssl.keystore.name = /keymanager/adminkeys.jceks
Admin.ssl.keystore.type = jceks
Admin.ssl.truststore.name = /keymanager/admintrustkeys
Admin.ssl.truststore.type = jceks
Audit.event.outcome = success,failure
Audit.event.types = all
Audit.eventQueue.max = 0
Audit.handler.file.directory = /keymanager/audit
Audit.handler.file.name = kms_audit.log
Audit.handler.file.size = 10000
Audit.metadata.file.name = /keymanager/metafile.xml
config.drivetable.file.url = FILE:///keymanager/drivetable
config.keystore.file = /keymanager/drive.keys
config.keystore.provider = IBMJCE
config.keystore.type = jceks
fips = Off
TransportListener.ssl.ciphersuites = JSSE_ALL
TransportListener.ssl.clientauthentication = 0
TransportListener.ssl.keystore.name = /keymanager/sslkeys
TransportListener.ssl.keystore.type = jceks
TransportListener.ssl.port = 443
TransportListener.ssl.protocols = SSL_TLS
TransportListener.ssl.truststore.name = /keymanager/ssltrustkeys
TransportListener.ssl.truststore.type = jceks
TransportListener.tcp.port = 3801
```

付録 B. Encryption Key Manager 構成プロパティー・ファイル

Encryption Key Manager には 2 つの構成プロパティー・ファイルが必要です。1 つは Encryption Key Manager サーバー用、もう 1 つは CLI クライアント用です。これらのファイルは、それぞれ、Java.util.Properties ロード・ファイルとして扱われて構文解析されます。この結果、プロパティーのフォーマットと仕様について、以下の特定の制約が生じます。

- 構成プロパティーは 1 行につき 1 つ記録されます。特定のプロパティーに対する値は、その行の最後まで続きます。
- パスワードのようにスペースを含むプロパティー値は、引用符で囲む必要はありません。
- 鍵ストアのパスワードは、長さが 127 文字を超えないようにしてください。
- 誤って行末に空白文字を入れると、プロパティー値の一部と解釈されることがあります。

サンプル構成プロパティー・ファイルは、**http://support.dell.com** からダウンロード用の EKMServicesandSamples ファイルを入手できます。

Encryption Key Manager サーバー構成プロパティー・ファイル

Encryption Key Manager サーバー構成ファイル (KeyManagerConfig.properties) 内のプロパティーの完全なセットは、次のもので構成されます。ファイル内のプロパティー設定の順序は問題ありません。ファイルにコメントが入っている場合があります。コメントを追加する場合は、1 つの行の最初の桁に記号「#」を使用してください。

注: KeyManagerConfig.properties ファイルに加えられた変更は、シャットダウン時に失われることがあります。したがって、構成プロパティーを編集する前に、Encryption Key Manager サーバーが実行されていないことを確認してください。 Encryption Key Manager サーバーを停止するには、CLI クライアントから **stopekm** コマンドを発行します。Encryption Key Manager サーバーを再起動すると、変更がアクティブになります。

Admin.ssl.ciphersuites = *value*

Encryption Key Manager サーバー間の通信に使用される暗号スイートを指定します。暗号スイートは、トランスポート層セキュリティー (TLS) および Secure Sockets Layer (SSL) でデータ転送用に使用される暗号アルゴリズムおよびハンドシェーク・プロトコルを記述します。

必須 オプション。

値 可能な値は、IBMJSSE2 によってサポートされるすべての暗号スイートです。

デフォルト JSSE_ALL

Admin.ssl.keystore.name = *value*

これは、Encryption Key Manager サーバー間の **sync** コマンドなどの

Secure Socket Layer クライアント操作に使用される、鍵ペアおよび証明書のデータベース名です。sync 操作では、セキュア・ソケット・クライアントがセキュア・ソケット・サーバーに提示する証明書は、この鍵ストアから取得されます。

必須 オプション。**sync** コマンドでのみ使用されます。デフォルトは **config.keystore.file** プロパティー値です。

Admin.ssl.keystore.password = password
Admin.ssl.keystore.name にアクセスするためのパスワード

必須 オプション。指定されなかった場合は、Encryption Key Manager の始動時にプロンプトが出されます。指定された場合、このプロパティーの値はセキュリティーを向上させるために暗号化され、プロパティー・ファイルのスタンザ名そのものが「Admin.ssl.keystore.password.obfuscated」という新しいスタンザに置き換えられます。

Admin.ssl.keystore.type = *value*
使用される鍵ストアのタイプ。

必須 オプション。

デフォルト jceks

Admin.ssl.protocols = *value*
セキュリティー・プロトコル。

必須 オプション。

値 SSL_TLS | SSL | TLS

デフォルト SSL_TLS

Admin.ssl.timeout = *value*
ソケットが SocketTimeoutException をスローするまでに read() を待機する時間の長さを指定します。

必須 オプション。

値 分単位で指定されます。0 はタイムアウトなしを意味します。

デフォルト 1

Admin.ssl.truststore.name = *value*
これは、セキュア・ソケット・サーバーがセキュア・ソケット・クライアントに提示するセキュア・ソケット・サーバー証明書の信頼性を確認するために使用されるデータベース・ファイル名です。

必須 オプション。**sync** コマンドでのみ使用されます。デフォルトは **config.keystore.file** プロパティー値です。

Admin.ssl.truststore.type = *value*
使用される鍵ストアのタイプ。

必須 オプション。

デフォルト jceks

Audit.event.outcome = *value*
指定された結果に終わった監査イベントのみが記録されます

必須 はい

値 success | failure。コンマまたはセミコロンで区切ると、両方を指定できます。

デフォルト success

Audit.event.Queue.max = 0
ファイルへのフラッシュが始まる、監査メモリー・キュー内でのイベント・オブジェクトの最大数。

必須 オプション。(推奨)。

値 0 - ? (0 は即時フラッシュを意味します。)

デフォルト 0

Audit.event.types = *value*
指定された結果に終わった監査イベントのみが記録されます

必須 はい

値 all | authentication | authorization | data synchronization | runtime | audit management | authorization terminate | configuration management | resource management | none。コンマまたはセミコロンで区切ると、複数の値を指定できます。

デフォルト all

Audit.handler.file.directory = ../audit
Audit.handler.file.name が配置されるディレクトリー

必須 オプション。(推奨)。

Audit.handler.file.multithreads = *value*
監査レコードを処理するために監査ハンドラーが個々のスレッドをディスパッチする必要があるかどうかを指定します。

必須 オプション。

値 true | false

デフォルト true

Audit.handler.file.name = kms_audit.log
監査項目がログに記録されるファイル名。

必須 はい

Audit.handler.file.size = 100
Audit.Handler.file.name が上書きを始めるサイズ。

必須 オプション。(推奨)。

値 0 - ? (キロバイト単位で指定。)

デフォルト 100

Audit.handler.file.threadlifespan = *value*

1 つの監査レコード処理スレッドの存続時間を制限します。audit.handler.file.multithreads= true の場合にのみ有用です。

必須 オプション。

値 ミリ秒単位で指定されます。

デフォルト 10000

Audit.metadata.file.cachecount = 100

メタデータ・ファイルを作成する前にメモリーに保管するレコードの数を指定します。

必須 いいえ

デフォルト 100

Audit.metadata.file.name = *value*

メタデータ・レコードが保管される XML ファイルの名前を指定します。

必須 はい

Audit.metadata.file.size = 1024

XML メタデータ・ファイルが閉じられ、新規ファイルが開始される前に XML メタデータ・ファイルが達する可能性がある、KB 単位で指定される最大ファイル・サイズを指定します。ファイルの現行および直前のバージョンのみが保管されます。

必須 いいえ

デフォルト 1024

config.drivetable.file.url = FILE:../filedrive.table

シリアル番号、認定など、テープ・ドライブに関する情報が含まれているファイル。

必須 はい

config.keygroup.xml.file = *value*

個々の別名がキー・グループ別に保管される場合、XML ファイルの名前を指定します

必須 オプション。

config.keystore.file = *value*

使用される鍵ストアを指定します。

必須 はい

config.keystore.password = password

config.keystore.file にアクセスするためのパスワード。指定された場合、このプロパティーの値はセキュリティーを向上させるために暗号化され、プロパティー・ファイルのスタンザ名そのものが「config.keystore.password.obfuscated」という新しいスタンザに置き換えられます。

必須 オプション。指定されなかった場合は、Encryption Key Manager の始動時にプロンプトが出されます。

config.keystore.provider = IBMJCE

必須 オプション。

config.keystore.type = jceks

必須 オプション。(推奨)。

デフォルト jceks

debug = *value*

指定された Encryption Key Manager コンポーネントについてデバッグを有効にします。

必須 オプション。

値 all | audit | server | drivetable | config | admin | transport | logic | keystore | console | none。コンマで区切ると、複数の値を指定できます。

デフォルト none

debug.output = *value*

指定された場所にデバッグ出力を送ります。

必須 オプション。

値 simple_file | console (推奨しません)。

debug.output.file = debug

デバッグ出力が書き込まれるパスおよびファイル名。

必須 オプション。debug.output = simple_file の場合は必須です。ファイルへのパスが存在している必要があります。

drive.acceptUnknownDrives = *value*

Encryption Key Manager に接続する新しいドライブを、ドライブ・テーブルに自動的に追加します。

必須 はい

値 true | false

デフォルト false

セキュリティー上の注 - 有効な drive.default.alias1 が設定されている場合にこの設定を組み合わせると、Encryption Key Manager に接続するテープ・ドライブが追加され、管理者がその追加を検証しなくても動作可能になります。詳しくは、第 3 章の「テープ・ドライブ・テーブルの自動更新」を参照してください。

fips = *value*

FIPS (連邦情報処理標準) 詳細については、第 2 章「Federal Information Processing Standard (連邦情報処理標準) 140-2 に関する考慮事項」を参照してください。

必須 オプション。

値 on | off

デフォルト off

maximum.threads = 200

Encryption Key Manager が作成できるスレッドの最大数。

必須　　オプション。

Server.authMechanism = *value*

ローカル/リモート・クライアントで使用される認証メカニズムを指定します。値が EKM に設定されている場合、CLI クライアント・ユーザーは usr/passwd を EKMAdmin/changeME として使用し、サーバーにログインする必要があります (このパスワードは chgpasswd コマンドを使用して変更できます)。値が LocalOS として指定されている場合は、クライアント認証はローカル・オペレーティング・システム・レジストリーに対して実行されます。(KeyManagerConfig.properties ファイルを変更する前に、必ず Encryption Key Manager サーバーをシャットダウンしてください。) CLI クライアント・ユーザーは OS usr/passwd を使用してサーバーにログインする必要があります。Linux のプラットフォームにおけるローカル OS ベースの認証の場合、以下の追加のステップが必要です。

1. Dell リリース R175158 (EKMServicesAndSamples) を **http://support.dell.com** からダウンロードし、任意のディレクトリーにそのファイルを解凍します。
2. EKMServiceAndSamples.jar (Dell 製品メディアに含まれている、または **http://support.dell.com** で入手可能) の内容を一時ディレクトリーに解凍します。
3. libjaasauth.so ファイルをご使用のプラットフォームに該当する LocalOS-setup から *java_home*/jre/bin にコピーします。
   - 32 ビット Intel Linux 環境では、LocalOS-setup/linux_ia32/libjaasauth.so ファイルを *java_home*/jre/bin/ ディレクトリーにコピーします。ここで、1.4.2 JVM を実行する 32 ビット Intel Linux カーネルの場合、*java_home* は、通常 *java_install_path*/IBMJava2-i386-142 です。
   - 64 ビット AMD64 Linux 環境では、LocalOS-setup/linux-x86_64/libjaasauth.so ファイルを *java_home*/jre/bin/ ディレクトリーにコピーします。ここで、1.4.2 JVM を実行する 64 ビット AMD Linux カーネルの場合、*java_home* は、通常 *java_install_path*/IBMJava2-amd64-142 です。

Windows プラットフォームでは、このファイルは不要です。

インストールが終了したら、Encryption Key Manager サーバーを始動できます。Encryption Key Manager クライアントは、OS ベースのユーザー/パスワードでログインできるようになっています。サーバーへのログインまたはサーバーへのコマンド発行が許可されているユーザー ID は、その ID の下でサーバーが実行されており、かつ superuser/root 権限を持っているユーザー ID のみであることに注意してください。

インストールの詳細については、README ファイル (Dell 製品メディア内および **http://support.dell.com** で入手可能) を参照してください。

必須　　オプション。

値　　EKM | LocalOS

デフォルト　　EKM

Server.password = *value*

内部プロパティー。編集しないでください。

symmetricKeySet = {GroupID | keyAliasList [, keyAliasList,]}

LTO 4 および LTO 5 テープ・ドライブに使用される対称鍵別名およびキー・グループを指定します。

必須 オプション。LTO 4 および LTO 5 テープ・カートリッジにのみ適用されます。

値

GroupID には 1 つの値を指定し、*keyAliasList* には 1 つ以上の値を指定します。

GroupID は、対称鍵のリストを事前に準備して、テープ・ドライブに別名が指定されないときにデフォルトとして機能するキー・グループ名を指定します。*GroupID* は KeyGroups.xml ファイルの既存のキー・グループ ID と一致する必要があります。一致しない場合は、KeyManageException が戻されます。複数の *GroupID* が指定されている場合は、KeyManagerException が戻されます。有効な *GroupID* を指定すると、キー・グループ XML で最後に使用された鍵が記録され、KeyGroups.xml から getKey が対称鍵のリストに対して呼び出されるたびに、その次の鍵がランダムに選択されて使用されます。*keyAliasList* の各指定に、*keyAlias* または *keyAliasRange* のどちらかの値が含まれます。

keyAlias は、鍵ストア内の対称鍵の名前または別名の Backus-Naur Form (BNF) を最大 12 文字の長さで、または sequentialKeyID を正確に 21 文字の長さで指定します。

keyAliasRange は、sequentialKeyID と 16 進数字をハイフン (-) で区切って、最大 18 文字で指定します。18 文字を指定する場合、先頭の 2 文字は 00 でなければなりません。1 行に指定する必要があり、改行を含んではなりません。

GroupID は別名のグループ名を指定します。

例 `symmetricKeySet = KMA0238ab34,KMB0000034acd2345678a,THZ001-FF` これは、LTO 4 および LTO 5 テープ・ドライブに鍵サービスを提供するときに、別名 KMA0238ab34 および KMB0000034acd2345678a、さらに THZ000000000000000001 から THZ0000000000000000FF までの範囲の別名を使用するよう Encryption Key Manager に指示します。これらの鍵は、プロパティー・ファイルで **config.keystore.file** によって指定される鍵ストアに存在している必要があります。

sync.action = *value*

自動同期時にデータをどのように処理するかを指定します。

必須 オプション。

値 rewrite | merge

デフォルト merge

注: 構成情報のマージは、構成情報の再書き込みと同じです。

sync.ipaddress = *ip_addr:ssl*
自動同期するリモート Encryption Key Manager の IP アドレスとポートを指定します。

必須 オプション。このプロパティーが指定されていない、あるいは誤って指定された場合は、sync 関数は使用不可になります。

値 リモート・サーバーの IP アドレス:SSL ポート番号

sync.timeinhours = *value*
リモート Encryption Key Manager との自動同期を行うまでに待機する時間数を指定します。

必須 オプション。

値 時間単位で指定されます。

デフォルト 24

sync.type = *value*
自動同期するデータを指定します。

必須 オプション。

値 config | drivetab | all

デフォルト drivetab

TransportListener.ssl.ciphersuites = JSSE_ALL
Encryption Key Manager サーバー間の通信に使用される暗号スイート。暗号スイートは、トランスポート層セキュリティー (TLS) および Secure Sockets Layer (SSL) でデータ転送用に使用される暗号アルゴリズムおよびハンドシェーク・プロトコルを記述します。

必須 オプション。

値 可能な値は、IBMJSSE2 によってサポートされるすべての暗号スイート。

TransportListener.ssl.clientauthentication = 0
Encryption Key Manager サーバー間の通信に必要な SSL 認証。

必須 オプション。

値 0 - クライアント認証なし (デフォルト)
1 - サーバーは、クライアントと一緒にクライアント認証を行うことを希望
2 - サーバーは、クライアントと一緒にクライアント認証を行う必要あり

TransportListener.ssl.keystore.name = *value*
セキュア・ソケット・サーバー用の証明書および秘密鍵を保持する Encryption Key Manager サーバーが使用するデータベースの名前。この証明書は、セキュア・ソケット・クライアントに認証および信頼性の確認用に与えられます。またこの鍵ストアは、Encryption Key Manager クライアントが Encryption Key Manager サーバーと通信し、セキュア・ソケット・クライアントとして機能するためにも使用します。

必須 はい

TransportListener.ssl.keystore.password = password

TransportListener.ssl.keystore.name にアクセスするためのパスワード。指定された場合、このプロパティーの値はセキュリティーを向上させるために暗号化され、プロパティー・ファイルのスタンザ名そのものが「TransportListener.ssl.keystore.password.obfuscated」という新しいスタンザに置き換えられます。

必須 オプション。

TransportListener.ssl.keystore.type = jceks

必須 オプション。(推奨)。

値 JCEKS

TransportListener.ssl.port = *value*

Encryption Key Manager サーバーが、他の Encryption Key Manager サーバーからの要求または Encryption Key Manager CLI クライアントからの要求がないか listen するポート。

必須 はい

値 ポート番号 (例えば、443)。これは CLI クライアント構成プロパティー・ファイルの TransportListener.ssl.port プロパティーと一致する必要があります。

TransportListener.ssl.protocols = SSL_TLS

セキュリティー・プロトコル

必須 オプション。

値 SSL_TLS (デフォルト) | SSL | TLS

TransportListener.ssl.timeout = 10

ソケットが SocketTimeoutException をスローするまでに read() を待機する時間の長さを指定します。

必須 オプション。

値 分単位で指定されます。

デフォルト 1

TransportListener.ssl.truststore.name = *value*

他のクライアントおよびサーバーの身元を確認するのに使用される公開鍵および署名済み証明書のデータベース名。

TransportListener.ssl.clientauthentication プロパティーがデフォルト値の 0 (クライアント認証は行われない) に設定されて**いない**場合、セキュア・ソケット・サーバーとして機能する Encryption Key Manager サーバーは、このファイルを使用してクライアントを認証する必要があります。またこのトラストストアは、Encryption Key Manager クライアントが Encryption Key Manager サーバーと通信し、セキュア・ソケット・クライアントとして機能するために使用します。

必須 はい

TransportListener.ssl.truststore.type = jceks

必須 オプション。(推奨)。

値 JCEKS

TransportListener.tcp.port = *value*

Encryption Key Manager サーバーが、テープ・ドライブからの要求がないか listen するポート。デフォルトの TCP ポート番号は 3801 です。

必須 はい

値 ポート番号 (例えば、10)。

TransportListener.tcp.timeout = *value*

ソケットが SocketTimeoutException をスローするまでに read() を待機する時間の長さを指定します。

必須 オプション。

値 分単位で指定されます。0 はタイムアウトなしを意味します。

デフォルト 10

CLI クライアント構成プロパティー・ファイル

このファイル (ClientKeyManagerConfig.properties) には、KeyManagerConfig.properties ファイルに含まれているプロパティーのサブセットが含まれています。このサブセットには次のプロパティーが含まれています。

TransportListener.ssl.ciphersuites = JSSE_ALL

Encryption Key Manager サーバーと CLI クライアント間の通信に使用される暗号スイート。暗号スイートは、トランスポート層セキュリティー (TLS) および Secure Sockets Layer (SSL) でデータ転送用に使用される暗号アルゴリズムおよびハンドシェーク・プロトコルを記述します。

必須 オプション。

値 この値は Encryption Key Manager サーバー・プロパティー・ファイル (KeyManagerConfig.properties) の TransportListener.ssl.ciphersuites に指定された値と一致する必要があります。

TransportListener.ssl.host = *value*

Encryption Key Manager CLI クライアントに対する Encryption Key Manager サーバーを識別します。

必須 オプション。

値 IP アドレスまたはホスト名

デフォルト ローカル・ホスト

例

```
TransportListener.ssl.host = 9.24.136.444
TransportListener.ssl.host = ekmsvr02
```

注: KeyManagerConfig.properties ファイルでは使用されません。

TransportListener.ssl.keystore.name = *value*

この鍵ストアは、Encryption Key Manager クライアントが Encryption Key Manager サーバーと通信し、セキュア・ソケット・クライアントとして機能するために使用します。

必須 はい

TransportListener.ssl.keystore.type = jceks

鍵ストアのタイプ。

必須 オプション。(推奨)。

デフォルト jceks

TransportListener.ssl.port = *value*

これは、CLI クライアントが Encryption Key Manager サーバーと通信するために使用するポートです。

必須 はい

値 この値は Encryption Key Manager サーバー・プロパティー・ファイル (KeyManagerConfig.properties) の TransportListener.ssl.port に指定された値と一致する必要があります。

TransportListener.ssl.protocols = SSL_TLS

セキュリティー・プロトコル

必須 オプション。

値 この値は Encryption Key Manager サーバー・プロパティー・ファイル (KeyManagerConfig.properties) の TransportListener.ssl.protocols に指定された値と一致する必要があります。

TransportListener.ssl.truststore.name = *value*

他のクライアントおよびサーバーの身元を確認するのに使用される公開鍵および署名済み証明書のデータベース名。

必須 はい

TransportListener.ssl.truststore.type = jceks

トラストストアのタイプ。

必須 オプション。(推奨)。

デフォルト jceks

サンプル構成プロパティー・ファイルは、**http://support.dell.com** からダウンロード用の EKMServicesAndSamples ファイルを入手できます。

付録 C. FAQ (よく尋ねられる質問)

アプリケーション・ベースの鍵管理とライブラリー管理の暗号化を組み合わせて使用できますか?

いいえ。アプリケーション管理暗号化が使用される場合、暗号化は、ライブラリー層で透過的となります。同様に、ライブラリーが管理する暗号化が使用される場合、プロセスは他の層でも透過的です。暗号化管理の各方式は、それぞれ排他的です。ライブラリー管理暗号化の場合、アプリケーションを変更する必要はありません。

テープの暗号化または暗号化解除の要求を生成する可能性のあるすべてのシステムに Encryption Key Manager をインストールして実行する必要がありますか?

ライブラリー管理の暗号化に関しては、テープ・ドライブ書き込み要求の発信元であるシステムが、Encryption Key Manager が実行中のシステムである必要はありません。しかも、Encryption Key Manager のインスタンスが、暗号化するテープ・ドライブへのアクセス元であるすべてのシステムで実行する必要もありません。

「drive.acceptUnknownDrives = True」パラメーターを組み込んだ場合でも、構成ファイルに「config.drivetable.file.url = FILE:/filename」パラメーターを含める必要がありますか?

config.drivetable.file.url は、必ず、指定する必要があります。これは、ドライブ情報が収められるところです。`drive.acceptUnknownDrives = True` を設定した場合、変数 `drive.default.alias1` および `drive.default.alias2` を正しい認証別名/鍵ラベルに指定することも必要です。

FILE:/filename は `config.drivetable.file.url` プロパティーの正しい構文ですか ?FILE:///filename がサンプル・ファイル内にあり、FILE:../ が記述にあります。

ご提示のサンプルは正しいです。これは、URL 仕様であり、通常、ディレクトリー構造仕様に予想されるものではありません。

Windows で実行する Encryption Key Manager のインスタンスの場合、KeyManagerConfig.properties ファイルに完全修飾パスを指定するためには、順スラッシュまたは逆スラッシュのいずれを使用するべきですか?

KeyManagerConfig.properties は Java のプロパティー・ファイルですので、Windows 内であっても、パス名として認められるのは順スラッシュのみです。KeyManagerConfig.properties ファイル内で逆スラッシュを使用すると、エラーが生じます。

Encryption Key Manager は証明書取り消しリスト (CRL) 検査を実行しますか?

いいえ、Encryption Key Manager は CRL 検査を実行しません。

テープを暗号化するために使用している証明書が失効すると、どのようなことが起きますか? Encryption Key Manager は以前に暗号化されたテープを読み取りますか?

証明書が失効しても、Encryption Key Manager にとっては問題ではありません。EKM は引き続き証明書を受け入れ、以前に暗号化されたテープを読み取ります。ただし、失効した証明書は、以前に暗号化されたテープを読み取ったり、追加できるようにその鍵ストアに残しておく必要があります。

Encryption Key Manager では、証明書の名前を変更したり、証明書を更新する必要がありますか?

特に指定のない限り、Encryption Key Manager は失効した証明書を使用して新規の鍵要求を受理するように構成されています。 Encryption Key Manager がこのように構成されている場合、証明書の更新は必要ありません。この機能が無効とされていても、この秘密鍵/証明書のペアを新規の鍵要求に使用する必要がある場合は、ユーザーは証明書を更新する必要があります。証明書 (有効期日) のみが更新されますが、関連付けられた鍵は更新されません。

新しいバージョンの Encryption Key Manager は、以前のバージョンのソフトウェアで作成された暗号化テープを読み取りますか?

はい。Encryption Key Manager はリリースに関係なく証明書を受理します。

特記事項

商標

Dell、Dell ロゴ、および PowerVault は、Dell Inc. の商標です。Microsoft および Windows は、Microsoft Corporation の登録商標です。他の商標および商標名は、それぞれ各社の商標、商標名、または製品です。 Dell Inc. は、自社以外の商標および商標名の所有権を放棄します。

用語集

この用語集は、本書および他の関連資料で使用する特別な用語、略語、および頭字語を定義したものです。

［ア行］

暗号化 (encryption). データを暗号に変換すること。データの暗号化および暗号化解除には鍵が必要である。暗号化により、鍵なしにデータにアクセスを試みる人またはソフトウェアから保護される。

［カ行］

鍵ストア (keystore). 秘密鍵とそれに関連する X.509 デジタル証明書チェーンのデータベースで、対応する公開鍵を認証するために使用される。環境によっては、証明書ストアまたは鍵リングともいう。

鍵の変更 (rekey). 既に暗号化されているテープに格納されているデータ・キー (DK) を保護する非対称の鍵暗号化鍵を変更するプロセス。これにより、異なるエンティティーがそのデータにアクセスすることが可能になる。

鍵ラベル (key label). 保護されている対称データ・キーをアンラップするのに必要な秘密鍵 (KEK) と EEDK を合わせるのに使用される固有の ID。使用される鍵ストアによっては、別名または証明書ラベルともいう。

鍵リング (key ring). 「鍵ストア (keystore)」を参照。

公開鍵 (public key). 非対称鍵ペアの一方の鍵。通常、暗号化に使用される。Encryption Key Manager は、公開鍵を使用して、AES データ・キーをテープ・カートリッジに保管する前にラップ (保護) する。

［サ行］

証明書 (certificate). 公開鍵を証明書所有者の身元にバインドして証明書の所有者が認証されるようにするデジタル文書。

証明書ストア (certificate store). 「鍵ストア (keystore)」を参照。

証明書ラベル (certificate label). 「鍵ラベル (key label)」を参照。

［ハ行］

秘密鍵 (private key). 非対称鍵ペアの一方の鍵。通常、暗号化解除に使用される。Encryption Key Manager は、秘密鍵を使用して、保護されている AES データ・キーを暗号化解除前にアンラップする。

別名 (alias). 「鍵ラベル (key label)」を参照。

A

AES. 拡張暗号化規格 (Advanced Encryption Standard)。米国政府が暗号化標準として採用しているブロック暗号。

D

DK. データ・キー (Data Key)。データの暗号化で使用される英数字ストリング。

E

EEDK. 外部暗号化データ・キー (Externally Encrypted Data Key)。データ・カートリッジに保管される前に鍵暗号化鍵で暗号化 (ラップ) されているデータ・キー。「KEK」を参照。

K

KEK. 鍵暗号化鍵 (Key Encrypting Key)。データ・キーを暗号化するために使用される英数字の非対称鍵。「EEDK」を参照。

P

PKDS. 公開鍵データ・セット。PKA 暗号鍵データ・セットともいう。

R

RSA. Rivest-Shamir-Adleman アルゴリズム (Rivest-Shamir-Adleman algorithm)。暗号化および認証に使用される、非対称の公開鍵暗号方式のための体系。1977 年に Ron Rivest、Adi Shamir、および Leonard Adleman によって考案された。この体系の安全性は、2 つの大きな素数の積を因数処理する難しさに依存する。

索引

日本語, 数字, 英字, 特殊文字の順に配列されています。なお, 濁音と半濁音は清音と同等に扱われています。

[ア行]

[カ行]

[サ行]

[タ行]

[ハ行]

[マ行]

［ヤ行］

［ラ行］

A

C

D

E

F

J

K

L

S

W

X