XVM Volume Manager Administrator's Guide
（日本語版）

007-4003-006JP

制作スタッフ
著作 Steven Levine
イラスト Dany Galgani、Chris Wengelski
編集 Susan Wilkening
製作 Diane Ciardelli
協力エンジニア Mark Maule、Eric Mowat、James Leong、Colin Ngam、Loellyn Cassell、Laurie Costello、Dean Roehrich、Terry Merth、Roger Strassburg、Michael Huovinen、Dean Jansa

COPYRIGHT
© 1999-2001 Silicon Graphics, Inc. All rights reserved. このマニュアルの別の箇所に示されているとおり、提供されている部分の著作権はサード・パーティが保持している場合があります。この電子ドキュメントの内容の一部または全部について、Silicon Graphics, Inc. から事前に文書による許諾を得ずに、いかなる方法でも複製または頒布したり、派生的な文書を作成することはできません。

LIMITED RIGHTS LEGEND
The electronic (software) version of this document was developed at private expense; if acquired under an agreement with the USA government or any contractor thereto, it is acquired as "commercial computer software" subject to the provisions of its applicable license agreement, as specified in (a) 48 CFR 12.212 of the FAR; or, if acquired for Department of Defense units, (b) 48 CFR 227-7202 of the DoD FAR Supplement; or sections succeeding thereto. Contractor / manufacturer is Silicon Graphics, Inc., 1600 Amphitheatre Pkwy 2E, Mountain View, CA 94043-1351, USA.

商標および帰属
Silicon Graphics および IRIX は登録商標です。SGI および SGI ロゴは Silicon Graphics, Inc. の登録商標です。

カバー・デザイン Sarah Bolles、Sarah Bolles Design、Dany Galganim、SGI Technical Publications

このガイドでの新機能

このバージョンのマニュアルは、IRIX 6.5.12 リリースをサポートしており、XVM システム・ディスクの設定と使用について説明します。この機能については、79 ページの「XVM システム・ディスク」で説明されています。このマニュアルで以前に XVM システム・ディスクとそれらの設定手順を参照していた箇所はすべて更新されています。

改訂履歴

バージョン	情報
001	1999 年 10 月 最初の印刷版。IRIX FailSafe 6.5.6 リリースをサポート
002	1999 年 10 月 IRIX FailSafe 6.5.6 リリースをサポート
003	2000 年 1 月 IRIX FailSafe 6.5.7 リリースをサポート
004	2000 年 4 月 IRIX FailSafe 6.5.8 リリースをサポート
005	2000 年 9 月 IRIX FailSafe 6.5.10 リリースをサポート
006	2001 年 4 月 IRIX FailSafe 6.5.12 リリースをサポート 最初の日本語版（上記は英語版前版からの改訂履歴です)。

目次

このガイドでの新機能 iii

改訂履歴 v

図一覧 xiii

表一覧 xvii

このガイドについて xix

関連ドキュメント xx

このガイドで使用されている表記規則 xx

ご意見とお問い合わせ先 xxi

1. XVM ボリューム・マネージャとは 1

XVM ボリューム・マネージャの機能 2

XVM 論理ボリューム・デバイス・ディレクトリ 4

XVM 下でのパーティション・レイアウト 5

XVM 論理ボリュームの構成 10

ボリューム 13

サブボリューム 16

スライス 17

連結 17

ストライプ 19

ミラー 21

論理ボリュームへのデータの書込み 22

CXFS クラスタの XVM 論理ボリューム 23

XVM 論理ボリュームとフェイルオーバー 24

2. XVM 管理の概念 25
XVM オブジェクト 25
XVM ドメイン 26
物理ディスク管理 29
物理ボリュームの作成 29
物理ボリュームの管理 30
物理ボリュームの表示 31
物理ボリュームのドメインの変更 31
実行中のシステムへの物理ボリュームの追加 31
物理ボリュームの交換 31
物理ボリューム名の変更 32
物理ボリュームの統計 32
システム・ディスクから option ディスクへの物理ボリュームの変更 32
物理ボリュームの破棄 32
論理リソースの作成 33
トポロジーの作成 33
ボリュームおよびサブボリュームの自動作成 33
ボリューム要素の命名 34
ボリューム要素の接続 35
ボリューム要素の分離 36
空のボリューム要素 36
論理ボリュームの統計 36
スライスの作成 37
連結の作成 37
ストライプの作成 37
ミラーの作成 38
読取りポリシー 39
プライマリ・レッグ 40
-clean ミラー作成オプション 40

-norevive ミラー作成オプション 40
ボリュームの作成 40
サブボリュームの作成 41
論理ボリュームの再構成 41
論理リソースの管理 42
ボリューム要素の表示 42
ボリューム要素の無効化 43
ボリューム要素のオンライン化 43
オンラインでの変更 44
XVM 設定の保存と再生成 44
論理リソースの破損 44

3. XVM コマンド・ライン・インタフェース 47
XVM CLI の使用 47
XVM CLI コマンドのオンライン・ヘルプ 49
XVM CLI の構文 50
XVM でのオブジェクト名 51
XVM オブジェクトの指定 51
断片を使った構文 53
XVM オブジェクト名の例 55
正規表現 55
XVM デバイス・ディレクトリとパス名 56
コマンド出力とリダイレクト 57
安全なコマンドと危険なコマンド 58

4. XVM 管理コマンド 59
物理ボリューム・コマンド 59
set コマンドを使った現在のドメインの変更 60
label コマンドを使った XVM ボリューム・マネージャへのディスクの割当て 60
show コマンドを使った物理ボリュームの表示 62
change コマンドを使った物理ボリュームの変更 64

probe コマンドを使った物理ボリュームの検査 64
dump コマンドを使った XVM 物理ボリュームの再生成 65
give と steal コマンドを使った物理ボリュームのドメインの変更 65
unlabel コマンドを使った XVM ボリューム・マネージャからのディスクの削除 67
論理ボリューム・コマンド 67
ボリューム要素の作成 67
slice コマンド 68
concat コマンド 68
mirror コマンド 69
stripe コマンド 70
subvolume コマンド 71
volume コマンド 72
ボリューム要素の変更 72
change コマンド 72
attach コマンド 73
detach コマンド 74
remake コマンド 74
実行中のシステムでのボリューム要素の変更 75
insert コマンド 75
collapse コマンド 76
ボリューム要素の表示: show コマンドの使用 77
ボリューム要素の再構築: dump コマンドの使用 78
ボリューム要素の削除: delete コマンドの使用 78
XVM システム・ディスク 79
XVM システム・ディスクの作成 80
slice コマンドを使ったシステム・ディスクの設定 85
XVM システム・ディスクのミラー化 86
XVM システム・ディスクへのアップグレード 88
IRIX 6.5.11 以前からの XVM システム・ディスクへのアップグレード 89

IRIX 6.5.12f 以降からの XVM システム・ディスクへのアップグレード 90
システムがインストールされていない XVM システム・ディスクへのアップグレード 90
XVM システム・ディスクからの起動 91
XVM システム・ディスクの削除 91

5. XVM 管理の手順 93
作業に入る前に 94
3 方向ストライプを使った論理ボリュームの作成 95
ディスクの一部のストライプ 98
data サブボリュームと log サブボリュームを使った論理ボリュームの作成 102
data サブボリューム、log サブボリューム、および real-time サブボリュームを使った論理ボリュームの作成 104
上の階層からのボリュームの作成 107
ストライプ・ミラーを使った XVM 論理ボリュームの作成 109
XVM システム・ディスクの作成とミラー化 112
label -mirror コマンドを使ったシステム・ディスクのミラー化 112
ミラーの挿入によるシステム・ディスクのミラー化 118
動作中の root ディスクでのミラー化 XVM システム・ディスクの作成 121
連結を使った swap ボリュームの設定 127
ミラー化を使ったオンライン再設定 132
オンラインでの論理ボリュームの変更 139
論理ボリュームの作成 139
論理ボリュームの拡張 141
論理ボリュームのデータのミラー化 143
ミラー化を使った連結のストライプへの変換 145
ミラーの削除 147
個々のストライプ・メンバーのミラー化 148

6. 統計データ 151
物理ボリュームの統計 152
サブボリュームの統計 152

ストライプの統計 152
連結の統計 154
ミラーの統計 155
スライスの統計 156

7. XVM ボリューム・マネージャの運用 157
クラスタ・システムのスタートアップ 157
ミラーの再生 158
クラスタの回復時のミラーの再生 159
XVM の調整可能パラメータ 159
XVM サブシステム・パラメータ 160

A. XVM 論理ボリュームと XLV 論理ボリューム 161
XVM 論理ボリュームと XLV 論理ボリュームの作成の比較 161
XLV 論理ボリュームから XVM 論理ボリュームへのアップグレード 163
XLV ミラー化ストライプの XVM ストライプ・ミラーへの切替え 164

索引 167

図一覧

図 1-1 **XVM option** ディスクのパーティション・レイアウト 6

図 1-2 **root** ファイルシステムと usr ファイルシステムが組合わさった XVM システム・ディスクのパーティション・レイアウト 7

図 1-3 独立した root ファイルシステムと usr ファイルシステムを持つ XVM システム・ディスクのパーティション・レイアウト 8

図 1-4 複数の root ファイルシステムがあるシステム・ディスクのパーティション・レイアウト 9

図 1-5 基本的な XVM ストライプ論理ボリューム 10

図 1-6 ミラー化ストライプと 3 つのサブボリュームを持つ XVM 論理ボリューム 11

図 1-7 連結を挿入した後の XVM 論理ボリューム 12

図 1-8 システム定義サブボリューム・タイプを持つ XVM ボリューム 14

図 1-9 ユーザ定義サブボリューム・タイプを持つ XVM ボリューム 15

図 1-10 **XVM** サブボリュームの例 17

図 1-11 **2** つのスライスで構成される連結 18

図 1-12 **2** つのミラーで構成される連結 18

図 1-13 **3** 方向ストライプ 19

図 1-14 ストライプ・ボリューム要素上のストライプ 20

図 1-15 **2** つのスライスで構成されるミラー 21

図 1-16 **2** つのストライプで構成されるミラー 21

図 1-17 ストライプと連結で構成されるミラー 22

図 1-18 非ストライプ論理ボリュームへのデータの書込み 22

図 1-19 ストライプ論理ボリュームへのデータの書込み 23

図 2-1 ローカル・ドメイン内の XVM 物理ボリューム 27

図 2-2 クラスタ・ドメイン内の XVM 物理ボリューム 28

図2-3 ラウンドロビン読取りポリシーでのミラーからのデータの読取り 39
図2-4 シーケンシャル読取りポリシーでのミラーからのデータの読取り 39
図3-1 システム生成名を使った XVM 論理ボリューム 54
図4-1 **XVM** システム・ディスク physvol fred 84
図4-2 **root** および swap 用の XVM 論理ボリューム 84
図4-3 **XVM** ミラー化 root physvol 87
図4-4 **root** と swap ミラー化論理ボリューム 88
図5-1 **3** 方向ストライプを使った XVM 論理ボリューム 95
図5-2 ディスクの一部のストライプ 98
図5-3 **log** サブボリュームを使った XVM 論理ボリューム 102
図5-4 **data** サブボリューム、log サブボリューム、および real-time サブボリュームを使った論理ボリューム 104
図5-5 ストライプ・ミラーを使った XVM 論理ボリューム 110
図5-6 **XVM** システム・ディスク physvol root_1 113
図5-7 ミラー化する前の XVM システム・ディスクの論理ボリューム 114
図5-8 **XVM** システム・ディスクのミラー physvol root_2 116
図5-9 ミラー化完了後の XVM システム・ディスクの論理ボリューム 117
図5-10 ミラー・コンポーネントを挿入した後の XVM システム・ディスクの論理ボリューム 119
図5-11 **XVM** システム・ディスク physvol xvmdisk 122
図5-12 **XVM** 論理ボリューム xvmdisk_root0 および xvmdisk_swap1 123
図5-13 **XVM** システム・ディスク physvol のミラー 124
図5-14 **XVM** システム・ディスク physvol xvmdisk のミラー化論理ボリューム 125
図5-15 連結を使った XVM swap ボリューム 128
図5-16 **XVM** システム・ディスク physvol bootdisk 129
図5-17 **XVM** 論理ボリューム bootdisk_root0 および bootdisk_swap1 130
図5-18 **XVM** システム・ディスク physvol moreswap 131
図5-19 オンラインの元のファイルシステム 132
図5-20 ミラーを挿入した後のファイルシステム 134

図5-21 ストライプをミラーに接続した後のファイルシステム 136
図5-22 元のスライスを分離した後のファイルシステム 137
図5-23 再設定されたファイルシステム 138
図5-24 元の XVM 論理ボリューム 139
図5-25 挿入後の XVM 論理ボリューム 141
図5-26 ミラー化した後の XVM 論理ボリューム 143
図5-27 連結をミラーに変換した後の XVM 論理ボリューム 145
図5-28 ミラーを削除した後の XVM 論理ボリューム 147
図5-29 スライスをミラー化した後の XVM 論理ボリューム 148

表一覧

表 1-1 **XVM** 論理ボリューム・デバイス・ディレクトリ 4

表 3-1 オブジェクト・タイプを指定するプレフィックス 52

表 3-2 断片を使った構文による論理ボリューム要素の指定 54

表 A-1 **XVM** 論理ボリュームと XLV 論理ボリュームの作成 162

このガイドについて

『*XVM Volume Manager Administrator's Guide*』では、XVM ボリューム・マネージャを使った XVM 論理ボリュームの設定と管理について説明します。このガイドは、IRIS FailSafe 6.5.12 リリースをサポートしています。

メモ：XVM ボリューム・マネージャは、CXFS ファイルシステムで階層化されている場合にのみ使用でき、スタンドアロンのボリューム・マネージャとしてはサポートされていません。CXFS ファイルシステムについての詳細は、『*CXFS Software Installation and Administration Guide*』を参照してください。

このガイドには、以下の情報が含まれています。

- 第 1 章、「XVM ボリューム・マネージャとは」では、XVM ボリューム・マネージャの機能と、XVM 論理ボリュームのコンポーネントの概要について説明します。
- 第 2 章、「XVM 管理の概念」では、管理コマンドの基礎となる概念について説明します。
- 第 3 章、「XVM コマンド・ライン・インタフェース」では、XVM コマンド・ライン・インタフェースの操作と、このインタフェースが提供する機能について説明します。
- 第 4 章、「XVM 管理コマンド」では、XVM コマンドの概要について説明し、各コマンドの例を示します。
- 第 5 章、「XVM 管理の手順」では、多くの一般的な XVM 管理手順の例を示します。
- 第 6 章、「統計データ」では、XVM 論理ボリュームのコンポーネントに対して XVM で維持される統計データの例を示します。
- 第 7 章、「XVM ボリューム・マネージャの運用」では、XVM ボリューム・マネージャの運用方法のさまざまな側面について説明します。
- 付録 A、「XVM 論理ボリュームと XLV 論理ボリューム」では、XVM 論理ボリュームと XLV 論理ボリュームを並べて比較し、既存の XLV 論理ボリューム設定を XVM 設定に切替えるための手順について説明します。

関連ドキュメント

以下のドキュメントには、この製品を使用する上で必要な補足情報が記載されています。

- 『*CXFS Software Installation and Administration Guide*』
- 『*IRIX Admin: Disks and Filesystems*』

このガイドで使用されている表記規則

このガイドでは、以下の表記規則と記号が使用されています。

`command`	この固定スペース・フォントは、コマンド、ファイル、ルーチン、パス名、シグナル、メッセージ、プログラミング言語、電子メール・アドレスなどのリテラル項目と、画面に表示される項目を示します。
変数	イタリック体は、変数のエントリ、および定義される語や概念を示します。
`user input`	この太字体の固定スペース・フォントは、対話型セッションでユーザが入力するリテラル項目を示します。出力は、太字体ではない固定スペース・フォントで示されます。
[ ]	コマンドやディレクティブ行のオプション部分は、角括弧で囲みます。
...	省略記号は、先行する要素が繰返し可能であることを示します。
manpage(*x*)	マン・ページのセクション ID は、マン・ページ名の後に括弧で囲んで示します。

ご意見とお問い合わせ先

このマニュアルの技術的正確性、内容、または構成についてご意見等ございましたら、弊社までお問い合わせください。コメントいただくドキュメントのタイトルとドキュメント番号も必ず一緒にお聞かせいただくようお願いいたします。オンラインの場合、ドキュメント番号はマニュアルの最初の部分に記載されています。印刷マニュアルの場合は、裏表紙にドキュメント番号が記載されています。

ご連絡の際は、以下のいずれかの方法をご利用いただけます。

- 電子メールの場合は、以下のアドレスまでお送りください。

  `techpubs@sgi.com`

- 次の Technical Publications Library の World Wide Web ページからの場合は、「Feedback」オプションをクリックしてください。

  `http://techpubs.sgi.com`

- お客様相談窓口までご連絡いただき、SGI の問題追跡システムへの入力をお申付けください。
- 郵送の場合は、次の住所までお送りください。

  Technical Publications
  SGI
  1600 Amphitheatre Pkwy.
  Mountain View, California, 94043-1351, USA

- FAX の場合は、「Technical Publications」宛で以下の番号までお送りください。

  +1 650 932 0801

いただいたコメントには迅速確実に対応いたします。

XVM ボリューム・マネージャとは

XVM ボリューム・マネージャは、ディスク・ストレージに対して論理的な構成を提供します。これにより、管理者は、基礎となる物理ディスク・ストレージを、論理ボリュームと呼ぶ単一の論理単位にまとめることができます。論理ボリュームは標準的なディスク・パーティションと同様に動作し、パーティションを指定できる箇所であればどこでも引数として使用できます。

論理ボリュームを使用すると、ファイルシステムや raw デバイスを物理ディスクのサイズよりも大きくできます。また、ボリュームを複数のディスクにストライプできるため、ディスク I/O パフォーマンスも向上できます。さらに、論理ボリュームは、複数のディスクにデータをミラー化する目的でも使用できます。

メモ：XVM ボリューム・マネージャは、CXFS ファイルシステムで階層化されている場合にのみ使用でき、スタンドアロンのボリューム・マネージャとしてはサポートされていません。CXFS ファイルシステムについての詳細は、『*CXFS Software Installation and Administration Guide*』を参照してください。

この章では、XVM ボリューム・マネージャの概要について説明します。この章には以下のトピックが含まれます。

- 「XVM ボリューム・マネージャの機能」（2 ページ）
- 「XVM 論理ボリューム・デバイス・ディレクトリ」（4 ページ）
- 「XVM 下でのパーティション・レイアウト」（5 ページ）
- 「XVM 論理ボリュームの構成」（10 ページ）
- 「論理ボリュームへのデータの書込み」（22 ページ）
- 「CXFS クラスタの XVM 論理ボリューム」（23 ページ）
- 「XVM 論理ボリュームとフェイルオーバー」（24 ページ）

XVM ボリューム・マネージャの機能

XVM ボリューム・マネージャは、XLV 論理ボリューム（SGI によって開発された以前の論理ボリュームの設計）で提供されていた論理ボリュームの基本的な機能はすべて備えています。これらの機能には、以下が含まれます。

- 自己識別型のボリューム

  論理ボリュームの固定の設定と属性情報は、論理ボリュームの一部であるすべてのディスクに配布されます。この情報はディスク上のラベル・ファイルに格納されるので、ファイルシステムに依存しません。ディスクのセット全体をシステム内およびシステム間で移動できます。

- 複数のストレージ・タイプ

  論理ボリュームは、連結とストライプによって集合ストレージをサポートします。また、ミラー化により冗長ストレージもサポートします。

- 複数のアドレス空間

  論理ボリュームは、サブボリュームの形式で、相互に排他的な複数のアドレス空間をサポートできます。論理ボリューム内の各サブボリュームは、そのデータにアクセスするアプリケーションによって定義された異なる用途に使用されます。XVM ボリューム・マネージャは、ファイルシステムのメタデータをデータ自体から分離するための *log* サブボリューム、保証レート I/O パフォーマンスのための *real-time* サブボリューム、およびユーザ・ファイルをはじめとするほとんどのデータが存在する *data* サブボリュームをサポートしています。

- パス・フェイルオーバー

  XVM ボリューム・マネージャは、ホスト内とホスト間の両方で、冗長コンポーネントを使用したシステム・フェイルオーバーをサポートします。ホストに異常が発生した場合でも、システムは、ディスクとの正常な接続があるかぎり、指定した操作の続行を試みます。

- オンラインでの設定変更

  XVM ボリューム・マネージャでは、管理者は、ボリュームをオフラインにすることなく、特定のボリュームの再設定を実行できます。オンラインで実行可能なボリュームの再設定には、連結ボリュームのサイズの拡張やミラーの断片の追加や削除などがあります。

XLV 論理ボリュームが提供する機能に加え、XVM ボリューム・マネージャは以下の重要な機能を提供します。

- クラスタ環境のサポート

  XVM ボリューム・マネージャはクラスタ環境をサポートしており、XVM デバイスのイメージをクラスタ内のセル全体に提供したり、クラスタ内の任意のセルから XVM デバイスを管理できます。クラスタ内のディスクは、クラスタ全体またはクラスタ内の個々のノードにローカル・ボリュームとして動的に割当てることができます。

- ボリュームの柔軟な階層化と設定

  XVM 論理ボリュームを構成する要素は、任意の設定で階層化できます。たとえば、管理者は、XVM ボリューム・マネージャを使用して、論理ボリューム設定の任意のレベルでディスクをミラー化したり、ボリュームのスループットの向上につながる場合は単純なストライプではなくストライプ・オン・ストライプ型の階層化を使用できます。

- 論理ボリュームを使ったシステム・ディスク

  XVM ボリューム・マネージャでは、システム・ディスクとして使用できるよう XVM ディスクにラベルを付けることができます。これにより、システム・ディスクのパーティションが含まれる XVM 論理ボリュームを作成できます。root パーティションをミラー化したり、任意の論理ボリューム設定で `usr` および swap パーティションを使用できます。

- 大量のスライス

  XVM でのディスクのレイアウトは、基礎となるデバイス・ドライバから独立しています。ディスクがどのようにスライスされるかは、XVM ボリューム・マネージャによって決定されます。これにより、XVM ボリューム・マネージャでは、ディスクを任意の数のスライスに分割できます。

- 大量のボリューム

  XVM ボリューム・マネージャは、単一のディスク上の数千個のボリュームをサポートしており、必要に応じてラベル・ファイルを拡張できます。XVM では、ボリューム幅（ボリュームの最も広い層を構成するボリューム要素の数）に制限はありません。

- ミラーのパフォーマンスの向上

  XVM ボリューム・マネージャでは、XVM ミラー要素に対して読取りポリシーを指定でき、設定のニーズに応じてシーケンシャルまたはラウンドロビン方式でミラーから読取ることができます。また、ミラーの特定のレッグを読取り用に優先させるかどうかも指定できます。

  また、作成時にミラーの同期が必要ない場合を指定したり、スクラッチ・ファイルシステムのミラーなどの特定のミラーの同期が必要ないように指定することもできます。

- 内蔵統計サポート

  XVM ボリューム・マネージャは、ボリューム・ツリーのすべてのレベルの統計を追跡して、タイプに固有の統計を提供します。統計はホストごとに追跡され、全体の状態を示すための PCP (Performance Co-Pilot) とのインタフェースが用意されています。

- デバイスのホット・プラグ

  XVM 論理ボリュームが含まれるディスクを実行中のシステムに追加でき、システムは、再起動せずに XVM 設定情報を読取ることができます。この機能により、システム間でディスクを移動したり、XVM 論理ボリュームが含まれる既存のディスクから新しいシステムを設定できます。

- 挿入と削除

  XVM の管理コマンドを使用することで、既存のディスク設定に対してコンポーネントの挿入と削除を行うことができます。これにより、ボリュームが開いている実行中のシステムでディスク設定を拡張したり変更できます。

XVM 論理ボリューム・デバイス・ディレクトリ

論理ボリュームは、/dev ディレクトリのサブディレクトリにブロックおよびキャラクタ・デバイスとして指定されます。

表1-1 に、XVM 論理ボリュームが含まれるディレクトリを示します。

表 1-1　　XVM 論理ボリューム・デバイス・ディレクトリ

デバイス・ディレクトリ	内容
/dev/cxvm	CXFS クラスタで使用される XVM 論理ボリューム用のブロック特殊ファイル
/dev/rcxvm	CXFS クラスタで使用される XVM 論理ボリューム用のキャラクタ特殊ファイル
/dev/lxvm	ホストのローカル・ボリュームに使用される XVM 論理ボリューム用のブロック特殊ファイル
/dev/rlxvm	ホストのローカル・ボリュームに使用される XVM 論理ボリューム用のキャラクタ特殊ファイル

クラスタ環境が実行されていない場合、論理ボリュームはすべてローカル・ボリュームと見なされます。

これらのディレクトリでの XVM 論理ボリュームのデバイス名は *volname_subvolname* です。*volname* は XVM 論理ボリュームの名前、*subvolname* はそのボリューム下にあるアクセスされるサブボリュームの名前になります。

XVM 論理ボリューム・デバイス・ディレクトリについての詳細は、56 ページの「XVM デバイス・ディレクトリとパス名」を参照してください。XVM 論理ボリューム内のオブジェクトの名前についての詳細は、51 ページの「XVM でのオブジェクト名」を参照してください。

XVM 下でのパーティション・レイアウト

ディスク上に XVM 論理ボリュームを作成する前に、ディスクに XVM ディスクとしてラベルを付ける必要があります。XVM ディスクのパーティション設定は、XVM ボリューム・マネージャによって制御されます。パーティションは XLV 論理ボリューム用なので、XVM スライスに使用できるストレージを定義する目的では使用されません。ディスクに XVM ディスクとしてラベルを付けると、IRIX ファイルシステムのパーティションの制限（16 個）はなくなります。

メモ：ディスクに XVM ディスクとしてラベルを付けるには、IRIX ディスクとしてフォーマットされていなければなりません。出荷時のセットアップ中にディスクが初期化されていない場合は、`fx`(1M) コマンドを使用して初期化してください。

ディスクに XVM ディスクとしてラベルを付けるときは、ディスクを XVM option ディスクにするか、root ファイルシステムと `usr` ファイルシステムを組合わせた XVM システム・ディスクにするか、または独立した root ファイルシステムと `usr` ファイルシステムを持つ XVM システム・ディスクにするかを指定できます。XVM ディスクは、デフォルトでは option ディスクとしてラベルが付けられます。

ディスクに XVM ディスクとしてラベルを付ける場合についての詳細は、29 ページの「物理ボリュームの作成」および 60 ページの「label コマンドを使った XVM ボリューム・マネージャへのディスクの割当て」を参照してください。XVM ディスクにシステム・ディスクとしてラベルを付ける場合についての詳細は、79 ページの「XVM システム・ディスク」を参照してください。

図 1-1 に、XVM option ディスクのパーティション・レイアウトを示します。XVM option ディスクでは、パーティション 10 にディスク全体が含まれ、パーティション 8 にボリューム・ヘッダが含まれます。パーティション 8 の一部ではないディスクの残りは、XVM ボリューム・マネージャを使用して指定したスライスに分割されます。

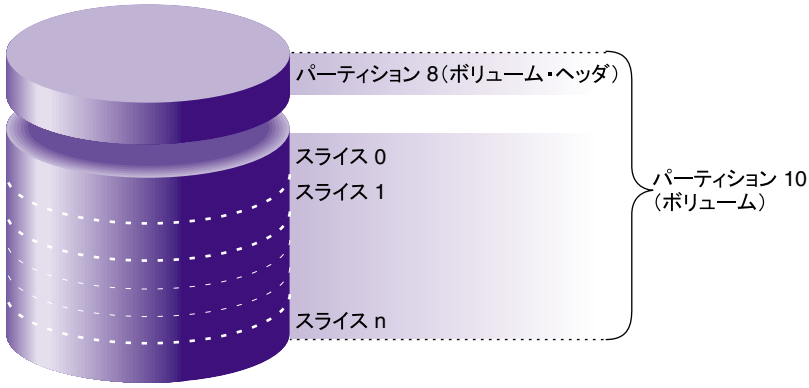

図 1-1 XVM option ディスクのパーティション・レイアウト

図1-2 に、ファイルシステムと usr ファイルシステムが組合わさった XVM システム・ディスクのパーティション・レイアウトを示します。パーティション 8 にはボリューム・ヘッダ、パーティション 9 にはディスク上の XVM ボリューム要素に関する情報が格納されている XVM ラベル領域、パーティション 0 には root パーティション、パーティション 1 には swap パーティションがそれぞれ含まれています。パーティション 10 にはディスク全体が含まれます。

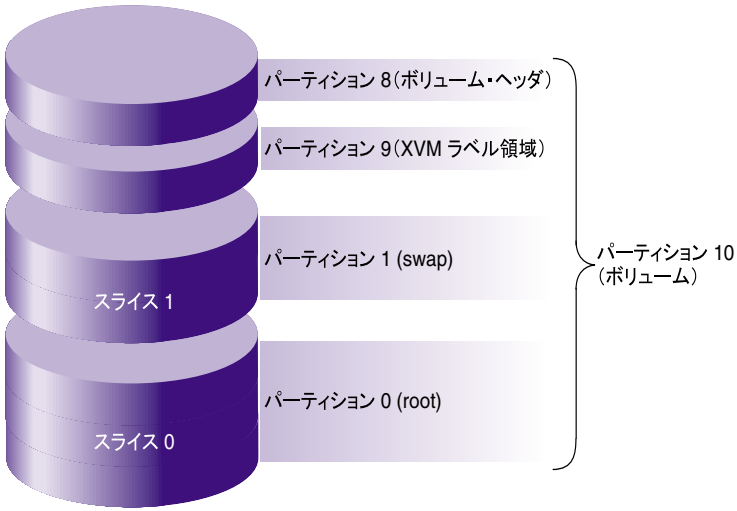

図 1-2 root ファイルシステムと usr ファイルシステムが組合わさった XVM システム・ディスクのパーティション・レイアウト

図1-3 に、独立した root ファイルシステムと usr ファイルシステムを持つ XVM システム・ディスクのパーティション・レイアウトを示します。パーティション 8 にはボリューム・ヘッダ、パーティション 9 にはディスク上の XVM ボリューム要素に関する情報が格納されている XVM ラベル領域、パーティション 0 には root パーティション、パーティション 1 には swap パーティション、パーティション 6 には usr パーティションがそれぞれ含まれています。パーティション 10 にはディスク全体が含まれます。この図の XVM システム・ディスクには、root、swap、および usr 以外のファイルシステムに使用できるディスク上の容量が含まれます。

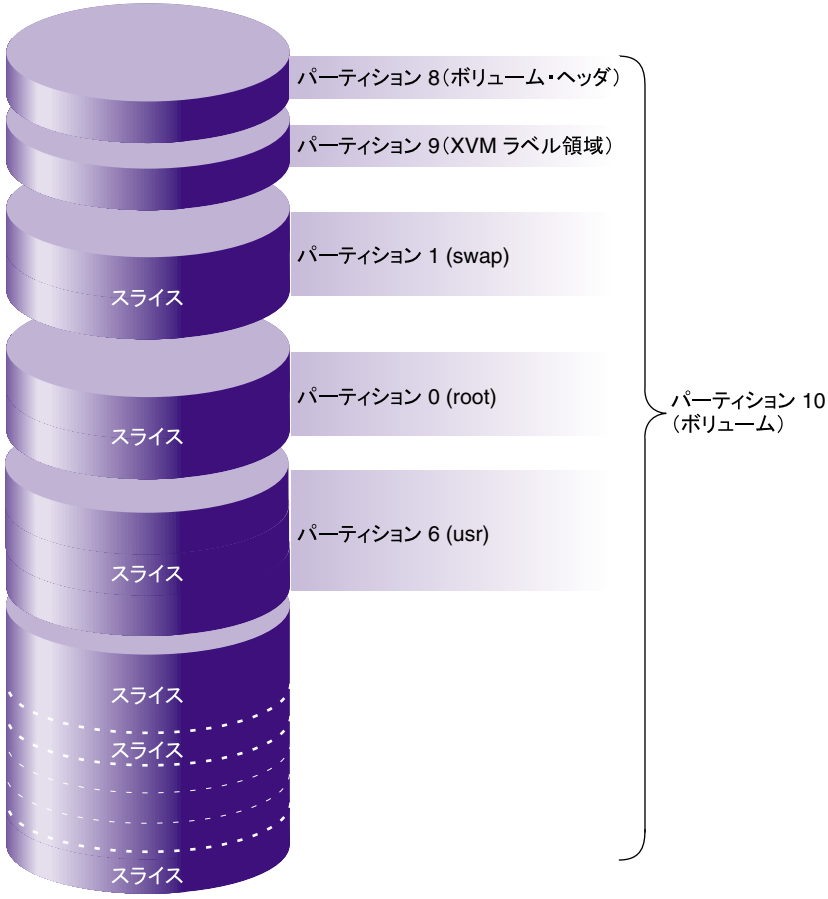

図 1-3 独立した root ファイルシステムと usr ファイルシステムを持つ XVM システム・ディスクのパーティション・レイアウト

メモ： fx(1M) コマンドを使用して XVM ディスクのパーティション・レイアウトを変更しようとすると、警告メッセージが表示されます。XVM によって管理されているディスクは、`hinv -c disk -v` コマンドを実行することで判別できます。

図 1-4 に、複数の root ファイルシステムのほかに独立した `usr` ファイルシステムもある XVM システム・ディスクのパーティション・レイアウトを示します。パーティション 8 にはボリューム・ヘッダ、パーティション 9 にはディスク上の XVM ボリューム要素に関する情報が格納されている XVM ラベル領域、パーティション 0 には最初の root パーティション、パーティション 1 には swap パーティション、パーティション 2 および 3 にはその他の root ファイルシステム、パーティション 6 には `usr` パーティションがそれぞれ含まれています。パーティション 10 にはディスク全体が含まれます。

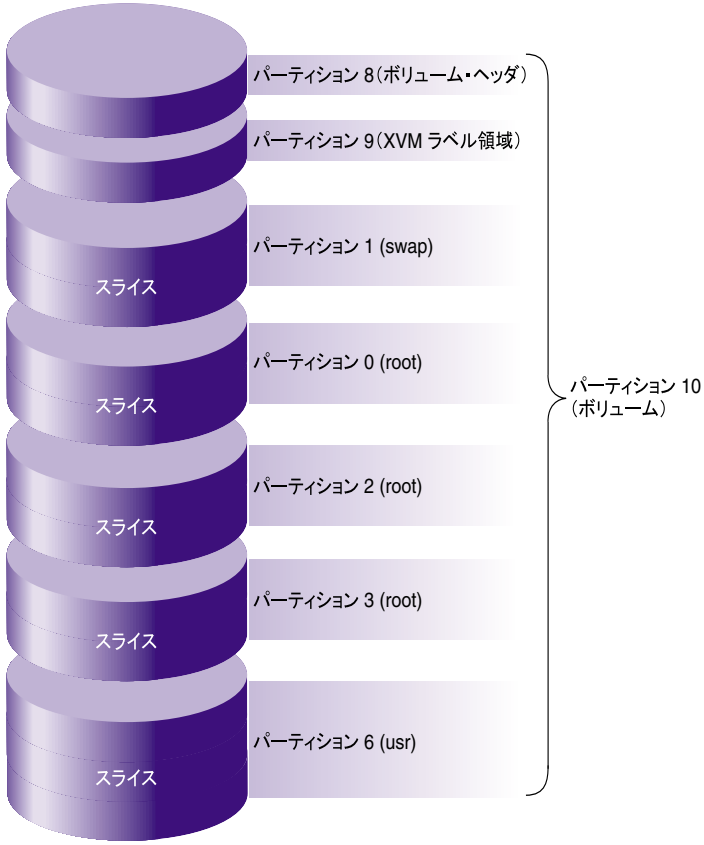

図 1-4 複数の root ファイルシステムがあるシステム・ディスクのパーティション・レイアウト

XVM 論理ボリュームの構成

XVM 論理ボリュームは、論理ストレージ・オブジェクトの階層で構成されます。つまり、ボリュームはサブボリュームで構成され、サブボリュームは、システムのニーズを満たす階層で組合わされたストライプ、ミラー、連結（連結されたボリューム要素）、およびスライスで構成されます。結果的に、階層の最下層では、各論理ストレージ・オブジェクトは物理ストレージの領域を定義するスライスで構成されることになります。これらの各論理ストレージ・オブジェクトを、ボリューム要素または *ve* と呼びます。

連結、ストライプ、およびミラー論理ボリューム要素は、自由に編成およびスタックできます。ボリュームからスライスまでは 10 レベル（ボリュームとスライスを含む）という制限があります。

階層で別のボリューム要素の下にある論理ボリューム要素は、より高いレベルのボリューム要素の子または断片と呼びます。ボリュームの子の数は 255 個、サブボリュームの子の数は 1 個、ミラーの子の数は 8 個までにそれぞれ制限されています。その他のボリューム要素の子の数は 65,536 個までに制限されています。

図 1-5 に、単純な XVM 論理ボリュームの例を示します。この例では、単一の 2 方向ストライプで構成される 1 つの data サブボリュームがあります。

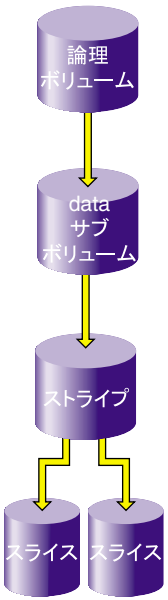

図 1-5　基本的な XVM ストライプ論理ボリューム

図 1-6 に、3 つのサブボリュームと、data サブボリュームに 1 つのミラー化ストライプを持つ XVM 論理ボリュームを示します。

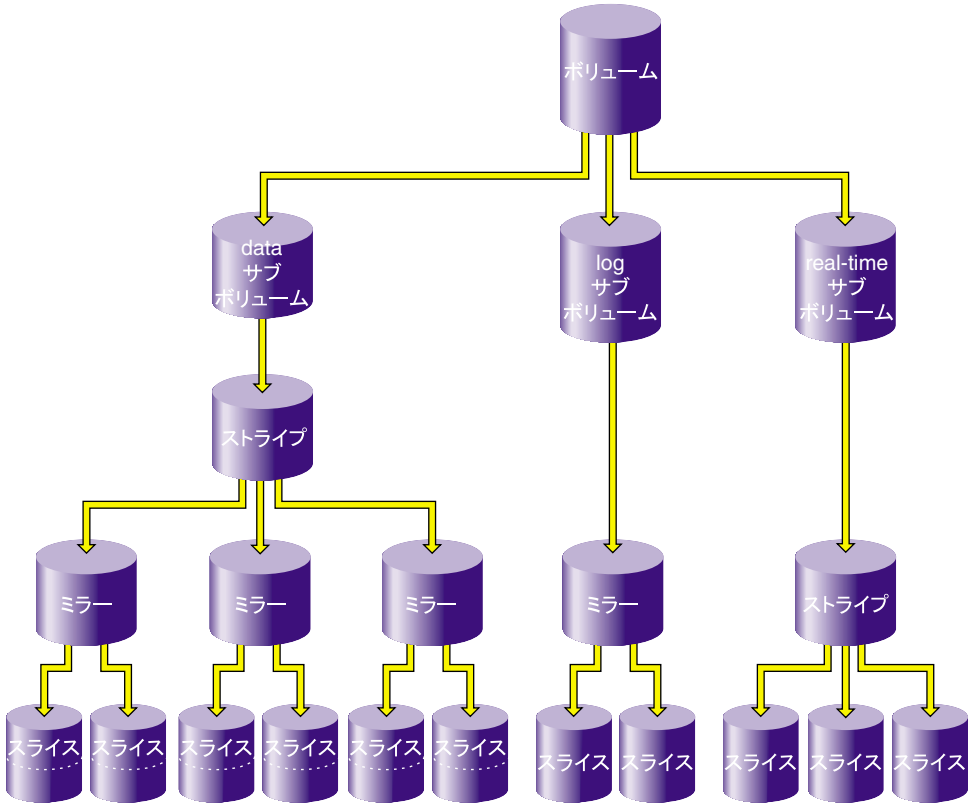

図 1-6 ミラー化ストライプと 3 つのサブボリュームを持つ XVM 論理ボリューム

図 1-7 に、図 1-6 に連結を挿入した後の例を示します。この例では、data サブボリュームを構成するディスク上の未使用ディスク容量に追加のスライスが作成されています。これらのスライスは並行ミラー化ストライプの作成に使用され、並行ミラー化ストライプを既存のミラー化ストライプと組合わせて連結が作成されています。

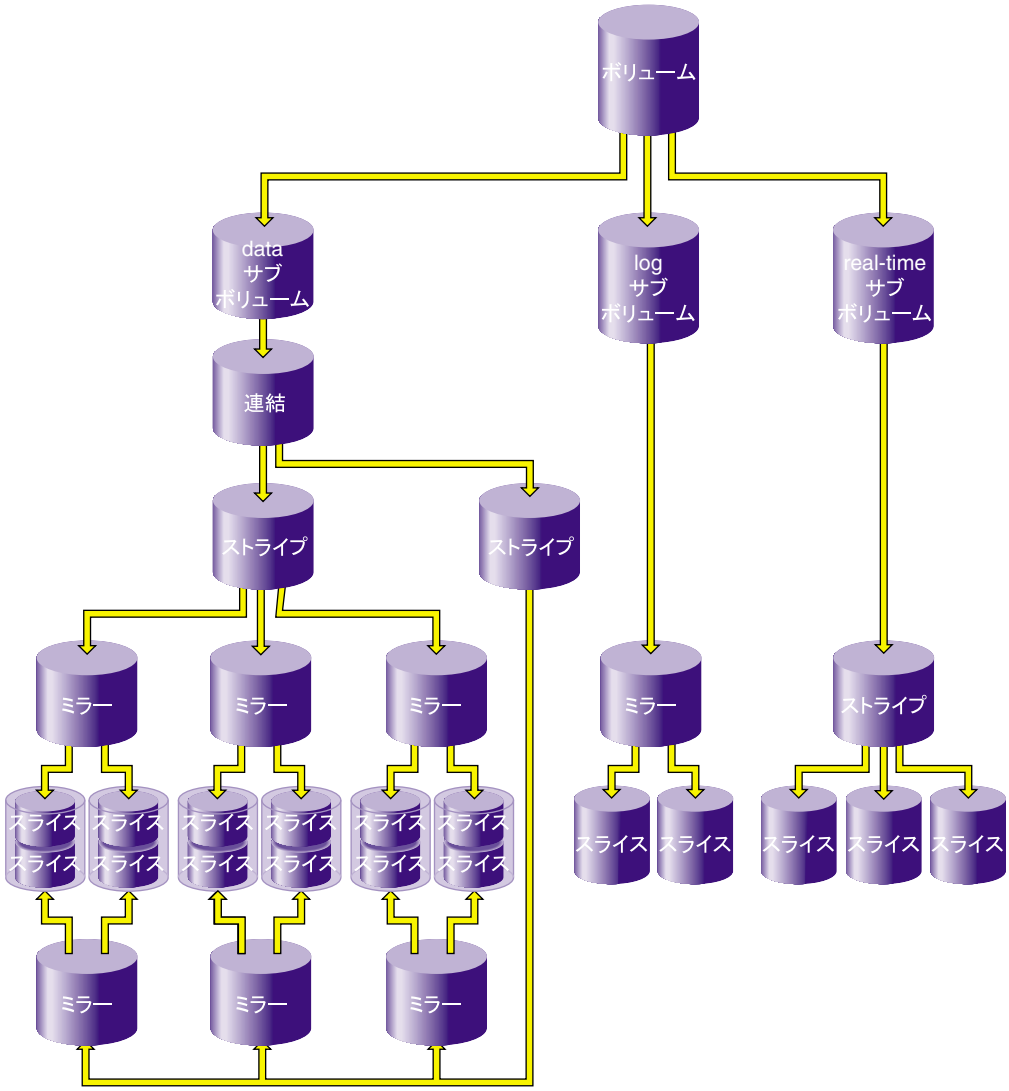

図 1-7 連結を挿入した後の XVM 論理ボリューム

以下の項では、XVM ボリューム要素について詳しく説明します。

ボリューム

ボリュームは、最上位の XVM ボリューム要素です。ボリュームはサブボリュームの集合で、単一のボリューム名でグループ化されます。

各ボリュームは単一のファイルシステムとして使用できます。システムによって使用されるボリューム情報は、そのボリュームが使用する各ディスクのボリューム・ヘッダの論理ボリューム・ラベルに格納されています。

ボリュームの作成、削除、およびシステム間での移動を行うことができます。

ボリュームを構成するサブボリュームは、data サブボリューム、log サブボリューム、および real-time サブボリュームとしてマークを付けることができます。これらはシステム定義サブボリューム・タイプで、16 ページの「サブボリューム」で説明します。サブボリュームには、ユーザ定義タイプとしてマークを付けることもできます。

同じボリューム内で特定のシステム定義タイプのサブボリュームを複数使用することはできません。つまり、ボリュームに含めることができるのは、data サブボリューム、log サブボリューム、および real-time サブボリュームをそれぞれ 1 つだけです。この制限は、ユーザ定義タイプのサブボリュームには適用されません。

図1-8 に、システム定義サブボリューム・タイプを持つ XVM ボリュームを示します。

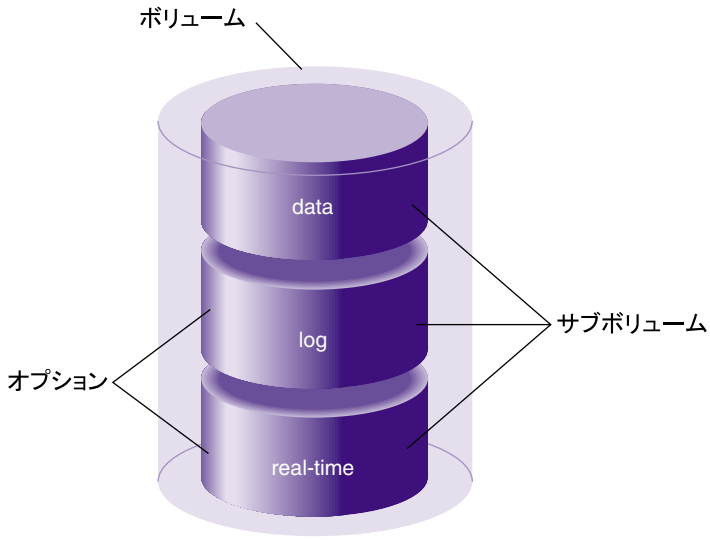

図 1-8 システム定義サブボリューム・タイプを持つ XVM ボリューム

図1-9 に、タイプ 16、17、および 18 として定義されているユーザ定義サブボリューム・タイプを持つ XVM ボリュームを示します。この例では、ボリュームの名前は `animation` で、サブボリュームの名前は `wire-data`、`shading`、および `texturemap` です。サブボリュームについての詳細は、16 ページの「サブボリューム」を参照してください。XVM オブジェクト名についての詳細は、51 ページの「XVM オブジェクトの指定」を参照してください。

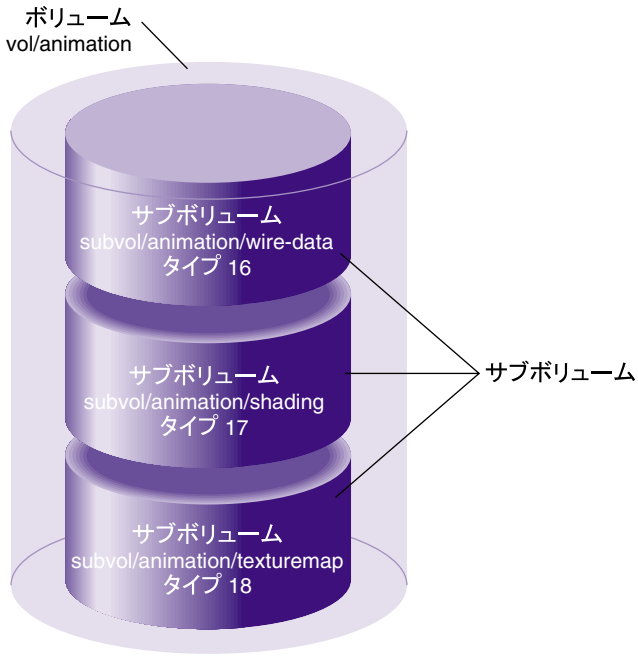

図 1-9 ユーザ定義サブボリューム・タイプを持つ XVM ボリューム

サブボリューム

サブボリュームは、XVM 論理ボリューム I/O のエントリ・ポイントです。各サブボリュームは独立したアドレス空間で、独立したタイプになります。XVM トポロジでサブボリュームの下に存在できるボリューム要素は 1 つだけです。

サブボリュームには、以下のシステム定義タイプを指定できます。

data サブボリューム
: XFS data サブボリュームは、ファイルシステム・デバイスとして動作するすべての XVM 論理ボリュームに必要です。

log サブボリューム
: log サブボリュームには XFS ファイルシステム・トランザクションのログが含まれ、クラッシュ後にシステムを高速に回復するために使用されます。XVM 論理ボリュームでは log サブボリュームはオプションです。log サブボリュームがない場合、ファイルシステム・ログは data サブボリュームに保持されます。

real-time サブボリューム
: real-time サブボリュームは、通常、データの整合性よりも応答時間が保証されていることの方が重要であるビデオなどのデータ・アプリケーションに使用されます。XVM 論理ボリュームでは real-time サブボリュームはオプションです。

  real-time サブボリュームの一部であるボリューム要素は、data または log サブボリュームに使用するボリューム要素と同じディスクには配置しないでください。優先レート保証による保証レート I/O に使用されるファイルでは、必ずこのように分離する必要があります。

システム定義サブボリューム・タイプにはユーザ定義名は使用できません。

サブボリュームには、ユーザ定義タイプとしてマークを付けることもできます。ユーザ定義タイプのサブボリュームに対しては名前を指定できます。

サブボリュームを使用すると、データのタイプが強制的に分離され、たとえば、ユーザ・データでファイルシステム・ログ・データを上書きできなくなります。また、サブボリュームを使用することで、パフォーマンスや信頼性の目標に合わせてファイルシステム・データとユーザ・データを設定することもできます。たとえば、複数のサブボリュームを異なるディスク・ドライブに配置してパフォーマンスを向上できます。

各サブボリュームは独立して構成できます。たとえば、log サブボリュームはフォールト・トレランスのためにミラー化し、real-time サブボリュームは、ビデオ再生で最大限のスループットを引出すために複数のディスクにストライプできます。

図1-10 に、XVM サブボリュームの構成の例を 4 つ示します。この図から、XVM サブボリュームには 1 つの子ボリューム要素だけを含められることがわかります。

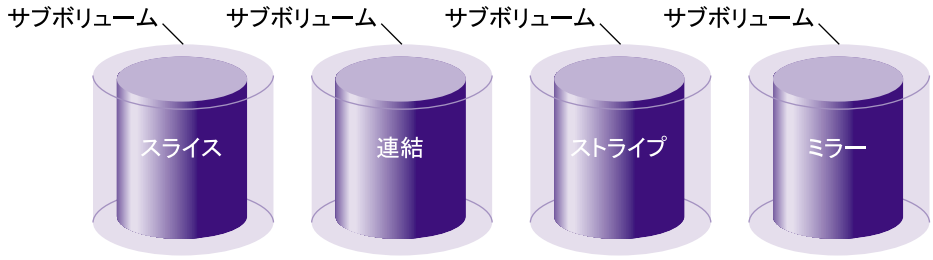

図 1-10 XVM サブボリュームの例

スライス

スライスは、XVM 論理ボリュームの階層で最下位のレベルです。スライスは物理ストレージを定義します。つまり、物理ディスクのアドレス空間をボリューム要素にマップします。

連結

連結は、その他のボリューム要素のストレージが 1 つの論理単位に組合わされるようにボリューム要素を組合わせる XVM ボリューム要素です。たとえば、2 つのスライスを 1 つの連結に組合わせることができます。

図 1-11 に、2 つのスライスで構成される連結を示します。

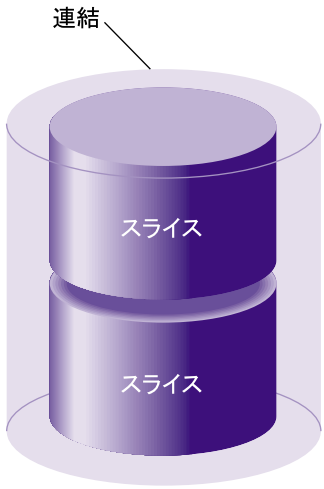

図 1-11 2 つのスライスで構成される連結

図 1-12 に、2 つのミラーで構成される連結を示します。

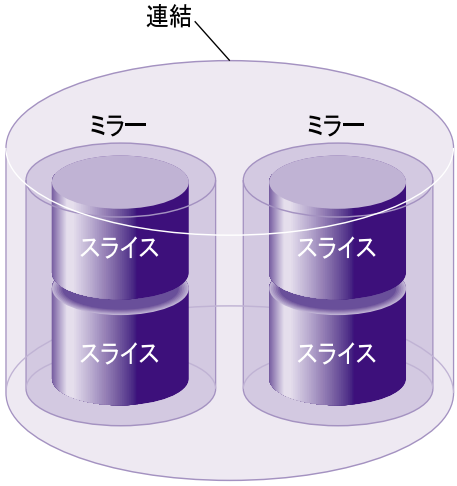

図 1-12 2 つのミラーで構成される連結

ストライプ

ストライプは、2 つ以上の基礎となるボリューム要素で構成される XVM ボリューム要素です。これらの要素は、ストライプ単位と呼ぶ一定の量のデータが、基礎となる各ボリューム要素に対してラウンドロビン形式で読書きされるように構成されます。

ストライプを使用すると、複数のディスク間でデータのセクションを順番に読書きできます。これにより、並行 I/O アクティビティが可能になり、パフォーマンス上の利点が得られます。

図 1-13 に、3 方向ストライプを示します。

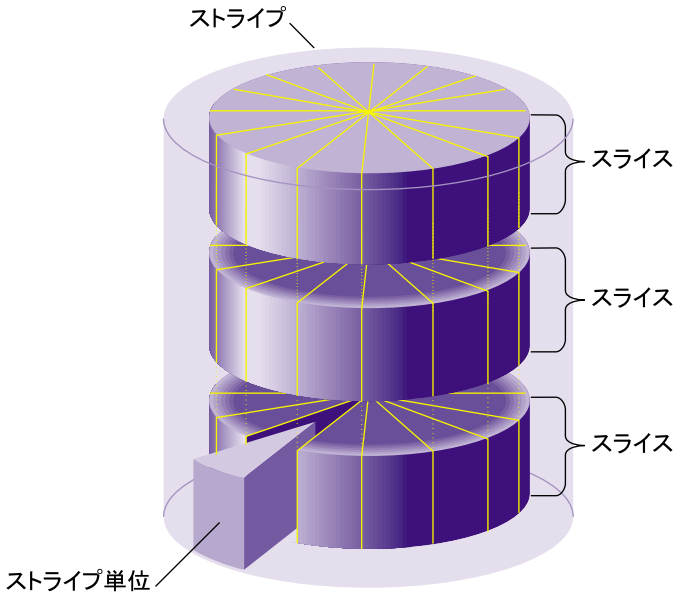

図 1-13 3 方向ストライプ

1 つのストライプにさらに別のストライプを設定すると、単一のより広いストライプを使用したときよりもパフォーマンスが向上する場合があります。図 1-14 では、2 つの 3 方向ストライプを作成してから、より大きなストライプ単位サイズを使用して再度ストライプしています。正しく設定されている場合、分離されたシーケンシャル・アクセス（さまざまなプロセスが同じアドレス空間の異なる部分にシーケンシャル I/O を行うような場合）を行うと、最上位のレベルのストライプの異なる半分が作成されます。この設定の利点は、並行大容量アクセスの場合に最上位のストライプの 2 つの半分が独立して動作できる点にあります。これに対して、単一の 6 方向ストライプでは、各ディスクに対する複数の I/O 操作が未完了となり、ディスク・シークが必要になります。

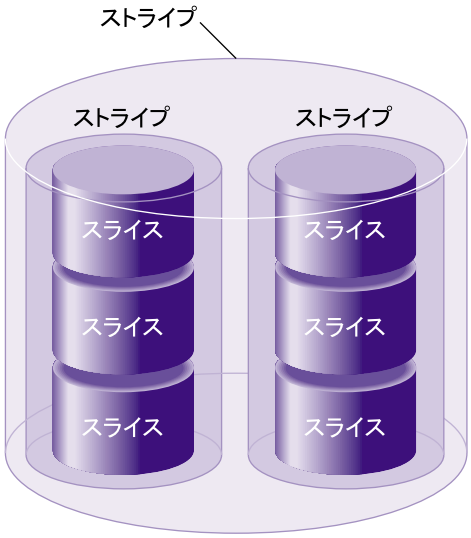

図 1-14　ストライプ・ボリューム要素上のストライプ

ミラー

ミラーは、基礎となるボリューム要素に同一のデータ・イメージを保持する XVM ボリューム要素です。このようなデータの冗長性により、システムの信頼性が向上します。ミラーのコンポーネントのサイズはすべて同じである必要はありませんが、コンポーネントのサイズが異なる場合は、大きい方のコンポーネントに未使用の容量が残ります。

図 1-15 に、2 つのスライスで構成されるミラーを示します。

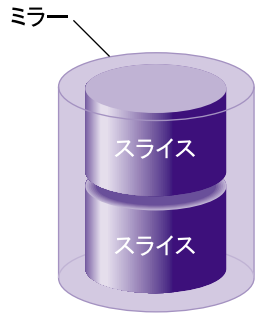

図 1-15　2 つのスライスで構成されるミラー

図 1-16 に、2 つのストライプで構成されるミラーを示します。

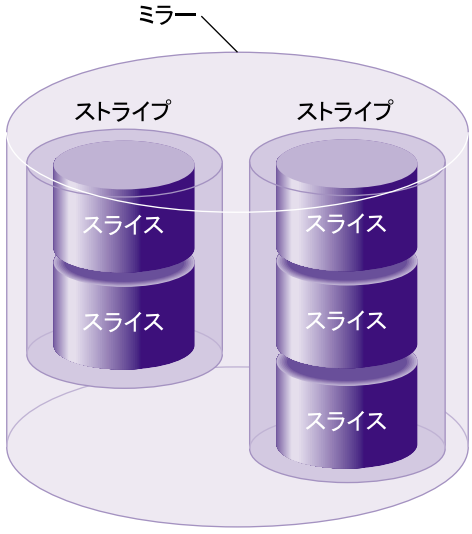

図 1-16　2 つのストライプで構成されるミラー

図 1-17 に、ストライプと連結で構成されるミラーを示します。

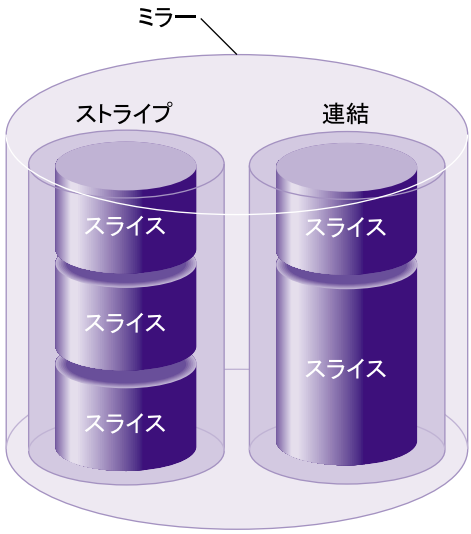

図 1-17　ストライプと連結で構成されるミラー

論理ボリュームへのデータの書込み

論理ボリュームには、別の物理ディスク・ドライブのスライスを含めることができます。論理ボリュームがストライプされていない場合、データは、ボリューム要素の最初のコンポーネントがいっぱいになるまでそのコンポーネントに書込まれてから、2 番目以降のコンポーネントに書込まれます。図 1-18 に、連結論理ボリュームにデータが書込まれる順序を示します。この図のくさび形の各部分は、ディスクに書込まれるデータの単位を表しています。データは、まず最初のコンポーネントがいっぱいになるまで最初のコンポーネントに書込まれた後、2 番目のコンポーネントがいっぱいになるまで 2 番目のコンポーネントに書込まれ、3 番目以降も同様に書込まれていきます。

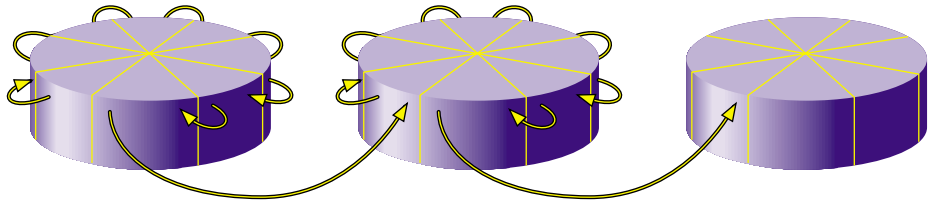

図 1-18　非ストライプ論理ボリュームへのデータの書込み

論理ボリュームがストライプされている場合は、ストライプ単位と呼ぶ一定の量のデータが、基礎となる各ボリューム要素にラウンドロビン形式で書込まれます。図1-19 に、3 方向ストライプを使用してストライプされたボリューム要素にデータが書込まれる順序を示します。くさび形の各部分は、データのストライプ単位を表しています。データの 1 つのストライプ単位がストライプの最初のコンポーネントに書込まれた後、データの 1 つのストライプ単位がストライプの 2 番目のコンポーネントに書込まれ、続いてデータの 1 つのストライプ単位がストライプの 3 番目のコンポーネントに書込まれます。その後、データの次のストライプ単位が最初のコンポーネントに書込まれます。

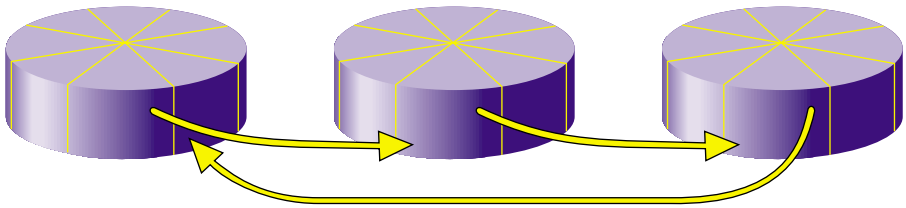

図 1-19　ストライプ論理ボリュームへのデータの書込み

CXFS クラスタの XVM 論理ボリューム

メモ：XVM ボリューム・マネージャは、CXFS ファイルシステムで階層化されている場合にのみ使用でき、スタンドアロンのボリューム・マネージャとしてはサポートされていません。CXFS ファイルシステムについての詳細は、『*CXFS Software Installation and Administration Guide*』を参照してください。

XVM ボリューム・マネージャは、CXFS クラスタ内のノードで共有される CXFS ファイルシステムによって使用されるボリューム・マネージャです。このため、XVM 物理ボリュームにはドメインがあります。ドメインは、クラスタまたはローカルにできます。クラスタ・ドメインを持つ XVM 物理ボリュームは CXFS クラスタによって所有され、ローカル・ドメインを持つ XVM 物理ボリュームは単一のノードによって所有されます。

クラスタ・ドメインを持つ XVM 物理ボリュームは、そのドメインを所有する CXFS クラスタ内の任意のノードで設定および変更できます。ローカル・ドメインを持つ XVM 物理ボリュームは、そのドメインを所有するローカル・ノードだけで設定および変更できます。ローカル・ドメインを持つ XVM 物理ボリュームに含まれる XVM 論理ボリュームは、ローカル・ボリュームと見なされます。

XVM ドメインについての詳細は、26 ページの「XVM ドメイン」を参照してください。CXFS についての詳細は、『*CXFS Software Installation and Administration Guide*』を参照してください。

XVM 論理ボリュームとフェイルオーバー

現在の XVM 設定で I/O を複数のコントローラに分散する必要がある場合は、`failover.conf` ファイルが完全に設定されていなければなりません。これは、選択したプライマリ・パスに I/O を制限するために必要です。たとえば、ストライプ・ボリュームを 2 つのホスト・バス・アダプタにまたがらせる場合は、`failover.conf` ファイルを設定してプライマリ・パスを指定する必要があります。

ストレージ・デバイス用のフェイルオーバーの設定についての詳細は、日本 SGI のサポート・プロバイダにお問い合わせください。 `failover.conf` ファイルについての詳細は、*failover*(7M) のマン・ページおよび `/etc/failover.conf` ファイル自体にも記載されています。

XVM 管理の概念

XVM 論理ボリュームの設定と管理を行う前に、管理コマンドの基礎となる概念について理解する必要があります。この章では、XVM ボリューム・マネージャで物理および論理ディスク・リソースに対して実行するタスクについて説明します。xvm コマンド・ライン・インタフェース (CLI: Command Line Interface) コマンドの詳細は、第 4 章、「XVM 管理コマンド」で各コマンドの例と併せて説明されています。

この章の主な節は、以下のとおりです。

- 「XVM オブジェクト」(25 ページ)
- 「XVM ドメイン」(26 ページ)
- 「物理ディスク管理」(29 ページ)
- 「論理リソースの作成」(33 ページ)
- 「論理リソースの管理」(42 ページ)
- 「論理リソースの破損」(44 ページ)

XVM オブジェクト

XVM オブジェクトは、以下のいずれかになります。

ラベルの付いていないディスク

ラベルの付いていないディスクとは、XVM ボリューム・マネージャによって XVM ディスクとしてラベルが付けられていないディスクです。

また、XVM ディスクとしてラベルが付けられていても、システムの最後の起動時以降に XVM ボリューム・マネージャによってラベルが読取られていないディスクも、XVM ボリューム・マネージャではラベルが付けられていないディスクと見なされます。このような状況は、以前にラベルが付けられたディスクを実行中のシステムに追加した場合などに起こることがあります。

物理ボリューム　XVM ボリューム・マネージャで使用できるようにラベルが付けられているディスクは、XVM の物理ボリューム（*physvol*）です。

外部ディスク　外部ディスクとは、XVM 物理ボリューム・ラベルの付いたディスクですが、別のノードまたは別のクラスタによって所有されているので、現在のノードで管理することはできません。

ボリューム要素　ボリューム要素（*ve*）は、XVM 論理ボリューム・トポロジーの構築ブロックです。XVM ボリューム、サブボリューム、連結、ストライプ、ミラー、およびスライスは、すべて XVM ボリューム要素です。

XVM ドメイン

XVM 物理ボリュームにはドメインがあり、クラスタまたはローカルのいずれかになります。クラスタ・ドメインを持つ XVM 物理ボリュームは、CXFS クラスタ・マネージャで定義する CXFS クラスタによって所有され、クラスタ内のどのノードでも制御できます。一方、ローカル・ドメインを持つ XVM 物理ボリュームは単一のノードによって所有され、そのノードだけで制御できます。

XVM 物理ボリュームの設定は、その物理ボリュームのオーナーだけが変更できます。XVM 物理ボリュームは、ホストから参照できる場合でも、別のホストまたは別のクラスタによって所有されている場合があります。このようなディスクは、XVM によって認識されて、外部ディスクとしてマークされます。XVM ラベルのないディスクは、ラベルの付いていないディスクとして表示されます。

図2-1 に、ローカル・オーナーによって制御される物理ボリュームを示します。この例では、ノード ricky のローカル・ドメインは XVM の physvol lucy にあります。ノード ricky は、ノード fred およびノード ethel も含む CXFS クラスタ neighbors の一部ですが、fred と ethel のどちらも lucy を制御できません。

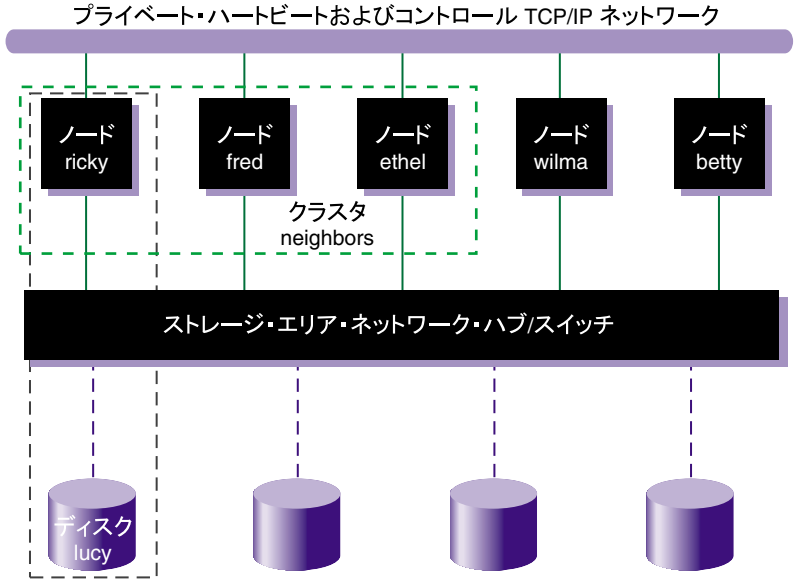

図 2-1 ローカル・ドメイン内の XVM 物理ボリューム

図 2-1 の設定では、ノード ricky は、physvol lucy の設定を参照および変更できます。lucy は、ノード fred、ethel、wilma、および betty では外部ディスクとして参照され、physvol 名自体ではなくディスク・パスのみが表示されます。必要であれば、62 ページの「show コマンドを使った物理ボリュームの表示」の説明に従って、外部ディスクに対して -show コマンドを実行し、その physvol 名を判断できます。

図 2-2 に、クラスタ・ドメインのある物理ボリュームを示します。この例では、ノード ricky、fred、および ethyl で構成されるクラスタ neighbors の所有権は、physvol lucy にあります。

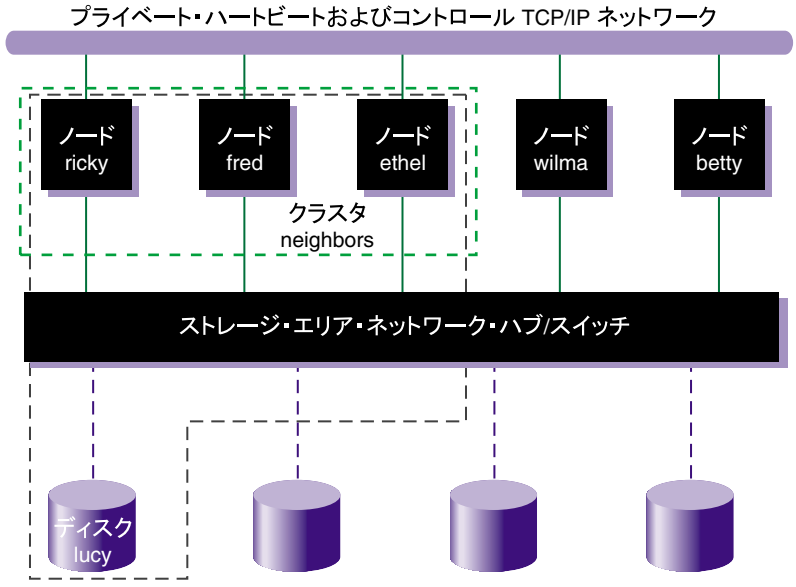

図 2-2 クラスタ・ドメイン内の XVM 物理ボリューム

図 2-2 の設定では、ノード ricky、fred、および ethyl は、physvol lucy の設定を参照および変更できます。ノード wilma および betty は、SAN ネットワークを通じて lucy に接続されていますが、lucy の設定を変更することはできません。これらのノードは、lucy を外部ディスクとして認識し、そのディスク・パスのみを表示できます。

複数の physvol にまたがる XVM 論理ボリュームは、実行中のシステムの複数のドメインにはまたがらない場合があります。ローカル・ドメインおよびクラスタ・ドメインにまたがる論理ボリュームには、オフラインのマークが付けられます。

クラスタ・サービスが有効になっているときに、xvm コマンドを使用して XVM ボリューム・マネージャを起動すると、デフォルトで xvm:cluster> プロンプトが表示されます。これは、この XVM セッションで作成した XVM 物理ボリュームがすべてクラスタ・ドメイン内にあることを示します。クラスタ・サービスが有効になっていない場合は、xvm:local> プロンプトが表示されます。これは、この XVM セッションで作成した XVM 物理ボリュームがすべてローカル・ドメイン内にあることが示します。

現在の XVM ドメインを変更するには、47 ページの「XVM CLI の使用」の説明に従って -domain オプションを使用して XVM を呼出すか、または 60 ページの「set コマンドを使った現在のドメインの変更」の説明に従って XVM ボリューム・マネージャの set コマンドを使用できます。

XVM ボリューム・マネージャをクラスタ・ドメインで実行しているときは、ローカル physvol のオーナーであるノードから実行中の場合でも、デフォルトでは、同じクラスタ・ドメイン内にある XVM の physvol しか参照および変更できません。ローカル・ディスクを参照および変更するには、set domain コマンドを使用してドメインをローカルに変更するか、physvol 名を指定するときに local: プレフィックスを使用します。同様に、ローカル・ドメインで XVM ボリューム・マネージャを実行している場合は、ドメインをクラスタに変更するか、クラスタが所有する physvol を指定するときに cluster: プレフィックスを指定しなければなりません。XVM ドメインの設定および指定についての詳細は、47 ページの「XVM CLI の使用」を参照してください。

既存の XVM の physvol のオーナーは、単一のノードまたはクラスタのどちらの場合でも、XVM の give コマンドを使用してその physvol を別のオーナーに与えることによって変更できます。物理ボリュームを現在所有しているノードまたはクラスタが give コマンドを実行できない場合は、steal コマンドを使用して、XVM 物理ボリュームのドメインを変更できます。give コマンドおよび steal コマンドについての詳細は、65 ページの「give と steal コマンドを使った物理ボリュームのドメインの変更」を参照してください。

物理ディスク管理

XVM 論理ボリュームの基礎となるのは、それらの論理ボリュームを構成する物理ボリュームです。XVM 論理ボリューム管理の一環として、以下の節の説明に従って、XVM 物理ボリュームの作成、管理、および破棄を行います。

物理ボリュームの作成

物理ディスク上に XVM 論理ボリュームを作成するには、XVM ボリューム・マネージャの label コマンドを使用して、ディスクに XVM ディスクとしてラベルを付けなければなりません。このコマンドを実行することで、XVM 物理ボリューム・ラベルがディスクに書込まれ、XVM ボリューム・マネージャでディスクのパーティション設定を制御できるようになります。CXFS クラスタでは、共有される XVM 物理ボリュームがクラスタ内のすべてのセルに物理的に接続されていなければなりません。

XVM ディスクにラベルを付けるときは、ディスクが、option ディスク、root と usr ファイルシステムが組合わさったシステム・ディスク、または root と usr ファイルシステムが独立したシ

ステム・ディスクのいずれかになるように指定できます。XVM 物理ボリュームは、デフォルトで option ディスクになります。システム・ディスクは、そこから起動するノードに対しては常にローカルなので、XVM システム・ディスクを作成するときは、XVM をローカル・ドメインで管理していなければなりません。ディスクに XVM システム・ディスクとしてラベルを付ける場合についての詳細は、79 ページの「XVM システム・ディスク」を参照してください。

実行中のシステムに、すでに XVM 物理ボリュームとしてラベルが付けられている新しいディスクを追加した場合は、そのディスクがシステムによって XVM ディスクとして認識されるよう、XVM ボリューム・マネージャの `probe` コマンドを使用して、ディスクを手動で検査しなければなりません。ただし、システム上の新しい XVM ディスクにラベルを付ける場合は、ラベル付けプロセスの一部として XVM ボリューム・マネージャによってディスクが検査されるため、この作業は必要ありません。ディスクはすべてシステムの起動時に検査され、どのディスクが XVM ディスクであるかが判断されます。

デフォルトでは、マウントされているファイルシステムとして使用中のパーティションがディスクに含まれている場合、このディスクに XVM ディスクとしてラベルを付けることはできません。この制約は、`label` コマンドの `-nopartchk` オプションでオーバーライドできます。使用中のパーティションが含まれているディスクにラベルを付けると、データが破壊されたり、システム・パニックが発生することがあるので、`-nopartchk` オプションの使用には注意が必要です。

メモ：ディスクに XVM ディスクとしてラベルを付けるには、IRIX ディスクとしてフォーマットされていなければなりません。出荷時のセットアップ中にディスクが初期化されていない場合は、`fx`(1M) コマンドを使用して初期化してください。

物理ボリュームの管理

物理ボリュームでは、以下のタスクを実行できます。

- 物理ボリュームの表示
- 物理ボリュームのドメインの変更
- 実行中のシステムへの物理ボリュームの追加
- 物理ボリュームの交換
- 物理ボリューム名の変更
- 物理ボリュームの統計の表示
- システム・ディスクから option ディスクへの物理ボリュームの変更

これらのタスクについては、以下の節で説明します。

物理ボリュームの表示

物理ボリューム（ラベルの付いているものと付いていないものの両方）に関する情報を表示するには、XVM ボリューム・マネージャの XVM show コマンドを使用します。また、show コマンドを使用して、現在のノードの外部にあるディスクの情報を表示することもできます。さらに、62 ページの「show コマンドを使った物理ボリュームの表示」に説明されているように、この機能を使用して外部ディスクの現在のオーナーを判別することもできます。

物理ボリュームのドメインの変更

既存の XVM の physvol のオーナーを変更し、physvol を別のローカルまたはクラスタのオーナーに与えるには、XVM の give コマンドを使用します。現在物理ボリュームを所有しているノードまたはクラスタが give コマンドを実行できない場合は、steal コマンドを使用して、XVM 物理ボリュームのドメインを変更できます。give コマンドおよび steal コマンドについての詳細は、65 ページの「give と steal コマンドを使った物理ボリュームのドメインの変更」を参照してください。

実行中のシステムへの物理ボリュームの追加

システムを起動すると、XVM ディスクであるかどうかを判断するために、そのシステムに接続されているすべてのディスクが検査されます。実行中のシステムに XVM ディスクを追加する場合は、カーネルによって XVM ディスクとして認識されるよう、XVM ボリューム・マネージャの probe コマンドを使用してディスクを手動で検査する必要があります。

物理ボリュームの交換

XVM ボリューム・マネージャでは、実行中のシステムのディスクを、システムを再起動せずに交換できます。この場合、交換したディスクで XVM ラベルを再生成する必要があります。物理ボリューム・ラベルを再生成するコマンドをファイルにダンプするには、XVM ボリューム・マネージャの dump コマンドを使用します。

78 ページの「ボリューム要素の再構築: dump コマンドの使用」で説明されているように、コマンドをダンプして物理ボリューム・ラベルを再生成する場合は、コマンドを個別かつ明示的にダンプして、物理ボリュームにつながるボリューム要素ツリーを再生成しなければならないことに注意してください。

物理ボリューム名の変更

物理ボリュームの名前は、change コマンドの name オプションを使用して変更できます。

物理ボリュームの統計

XVM ボリューム・マネージャは、物理ボリューム、サブボリューム、ストライプ、連結、ミラー、およびスライスの統計を維持できます。XVM ボリューム・マネージャの change コマンドの stat オプションを使用して、統計をオンまたはオフにしたり、物理ボリュームの統計をリセットできます。XVM が維持する統計についての詳細は、第 6 章、「統計データ」を参照してください。

クラスタ化された環境では、ローカル・セルの統計だけが維持されます。

システム・ディスクから option ディスクへの物理ボリュームの変更

swap ボリュームとしてマークされているボリュームは削除できないので、XVM システム・ディスクの論理ボリュームを削除する必要がある場合は、XVM システム・ディスクを XVM option ディスクに変更しなければならないことがあります。XVM ディスクは、change コマンドを使用して option ディスクに変更できます。システム・ディスクの変更についての詳細は、91 ページの「XVM システム・ディスクの削除」を参照してください。

物理ボリュームの破棄

システムから XVM 物理ボリュームを削除するには、XVM ボリューム・マネージャの unlabel コマンドを使用して XVM 物理ボリューム・ラベルを XVM ディスクから削除し、元のパーティション設定スキームを復元します。unlabel コマンドには -force オプションが用意されており、各スライスが開いているサブボリュームの一部で、それを削除するとサブボリュームの状態がオフラインになる場合でも、現在物理ボリュームに存在している各スライスを削除します。

XVM システム・ディスクの swap パーティションは削除できません。これは、swap パーティションを誤って削除した場合にシステム・パニックが発生するのを防止するためです。システム・ディスクの論理ボリュームを削除して、システムからラベルを削除する必要がある場合は、91 ページの「XVM システム・ディスクの削除」の説明に従って、まず XVM の change コマンドを使用してそのディスクをシステム・ディスクから option ディスクに変更しなければなりません。

論理リソースの作成

論理ボリュームに使用する XVM 物理ボリュームを作成したら、論理ボリュームを構成する論理ボリューム要素を作成できます。

トポロジーの作成

XVM トポロジーは、上から下または下から上へと構築できます。終端にスライスがないツリーやサブツリーを作成した場合は、ラベルがディスクに書込まれないので、再起動するとツリーやサブツリーが変更されます。

XVM トポロジーを構築するときは、XVM ボリューム・マネージャの `show` コマンドの `-topology` オプションを使用して、ボリューム要素に定義されている既存のトポロジーを表示すると、便利な場合があります。

ボリュームおよびサブボリュームの自動作成

ボリューム以外のボリューム要素を作成したときは、それらのボリューム要素をボリュームと関連付ける必要があります。ボリューム要素を作成するときにボリュームに明示的に名前を付けて作成したり、一時名を使用してボリュームが自動的に生成されるよう指定することもできます。タイプ `data` のサブボリュームは、そのボリュームに対して自動的に生成されます（作成するボリューム要素自体が別のタイプのサブボリュームである場合を除く）。ボリュームおよびサブボリュームの自動生成機能を使用すると、オブジェクトの構築と同時に、`mkfs` などのアプリケーションでそのオブジェクトを使用してファイルシステムを初期化できるようになります。

ボリュームに明示的に名前を付けた場合、ボリューム名はラベル領域に格納され、マシンを再起動しても変わりません。ボリュームおよびボリューム名をシステムに自動生成させたときは、多くの場合は別の新しい名前がシステムの再起動時に生成されます。ただし、スライスは特殊なケースで、スライスのボリューム名をシステムに生成させても、そのボリューム名は一時名にはならず、再起動しても変わりません。

一時ボリューム名を変更して再起動しても変わらないようにするには、ボリューム名を変更する `change` コマンドを使用します。

ボリューム要素の命名

XVM ボリュームを構成しているボリューム要素には、以下のように名前が付けられます。

- スライスは、作成時に自動的に名前が付けられます。

  スライス名は、マシンを再起動しても変わりません。このため、システムを再起動した後でも、各ディスクに定義したスライスを使用して論理ボリュームを効率的に再構成および再ビルドできます。

- ストライプ、連結、およびミラーには、作成時に明示的に名前を付けることができます。明示的に名前を付けない場合は、デフォルトの一時名が生成されるよう指定しなければなりません。

  ストライプ、連結、およびミラーに明示的に名前を付けると、そのボリューム要素名はラベル領域に格納され、マシンを再起動しても変わらなくなります。ラベル領域のサイズ設定についての詳細は、60 ページの「label コマンドを使った XVM ボリューム・マネージャへのディスクの割当て」を参照してください。

- サブボリュームは、ユーザ定義タイプの場合にのみ明示的に名前を付けることができます。data サブボリュームは `data`、log サブボリュームは `log` と名前が付けられます。

- 33 ページの「ボリュームおよびサブボリュームの自動作成」で説明されているように、ボリューム内に要素を作成したときに、ボリュームを作成して名前を付けることができます。また、空のボリュームを作成して、明示的に名前を付けることもできます。空のボリュームを作成するときに名前を付けない場合は、システムによって一時名が生成されるよう指定しなければなりませんが、一般的な設定の場合、この方法は推奨されていません。

一時ボリューム要素名を変更して再起動しても変わらないようにするには、ボリューム要素名を変更する `change` コマンドを使用します。

ボリューム要素を操作するときに、名前を使用する必要はありません。代わりに、論理ボリューム内での相対位置を使用できます。これらの命名オプション、およびボリューム要素名の構文に関する一般情報は、51 ページの「XVM でのオブジェクト名」で説明されています。

ボリューム要素の接続

ボリューム要素を作成するか、attach コマンドを使用することで、ボリューム要素を互いに接続して XVM 論理ボリュームを作成する場合は、XVM ボリューム・マネージャによって以下のルールおよび制約が強制的に適用されます。

- 接続のソースは、サブボリュームまたはサブボリュームの子でなければなりません。ボリュームを別のボリューム要素に接続することはできません。
- サブボリュームは、ボリュームだけに接続できます。
- サブボリュームは、1 つの子しか持つことができません。
- ボリュームは、特定のタイプのシステム定義サブボリュームを複数持つことはできません。システム定義サブボリュームは、data サブボリューム、log サブボリューム、および real-time サブボリュームです。
- ミラーは、9 つ以上のメンバーを持つことはできません。
- ボリューム要素を作成したときに位置を指定する場合や、ボリューム要素をターゲット・ボリューム要素に接続するときに位置を指定する場合、ターゲット・ボリューム要素の該当する位置にボリューム要素があってはなりません。
- 開いているサブボリュームの一部であるターゲットにボリューム要素を接続する場合、接続によってターゲットまたはターゲットの先祖のデータのレイアウトが変更されてはなりません。以下に、データ・レイアウトに影響を与える可能性のある接続の例を示します。
  - ストライプへの追加
  - 親の下位にある右端のボリューム要素ではない先祖を拡張するような連結への追加

  ボリューム要素をミラーに接続すると、ミラーの再生が開始され、その間にシステムによってデータがミラー化されます。このプロセスが完了すると、メッセージが SYSLOG に書込まれます。ミラーの再生は、いったん開始したら、ミラーの断片の 1 つを除いたすべてを分離する方法以外では中止できません。

  xvm コマンドの -safe オプションを使用するときは、開いているサブボリュームにターゲットが属していない場合でも、そのターゲットまたはターゲットの先祖のデータのレイアウトを変更するボリューム要素を接続することはできません。

複数のソース・ボリューム要素を単一のターゲット・ボリューム要素に接続すると、一度に 1 つずつ順番に接続されます。リストに含まれる特定の接続が失敗した場合、XVM はボリューム要素を以前の親に復元します。ボリューム要素を復元できない場合は、警告メッセージが表示され、手動操作が必要になります。

ボリューム要素の分離

ボリューム要素をその親から分離するには、XVM ボリューム・マネージャの detach コマンドを使用します。ボリューム要素を作成するとボリュームが作成されるのと同じように、ボリューム要素を分離すると新しいボリューム（多くの場合は data サブボリュームも）が作成されます。生成されたボリュームに明示的に名前を付けたり、ボリュームが一時名で自動的に生成されるよう指定することもできます。分離するボリューム要素に対して、タイプ data のサブボリュームが自動的に生成されます（分離するボリューム要素自体が別のタイプのサブボリュームである場合を除く）。

分離するボリューム要素が開いているサブボリュームの一部である場合、分離によってサブボリュームの状態がオフラインになることはありません。ミラー化されていないスライスを分離するなど、開いているサブボリュームのアドレス空間を減らすコマンドを実行した場合は、サブボリュームの状態がオフラインになります。開いているミラーの最後の有効な断片をそのミラーから分離することはできません。この操作を行うと、ミラーがオフラインになってしまうためです。

detach コマンドには -force オプションが用意されており、このオプションを使用することで、サブボリュームがオフラインになるようなボリューム要素の分離は行えないという制約をオーバーライドできます。また、detach コマンドには、サブボリュームが開いていない場合でもこの制約を適用する -safe オプションも用意されています。このコマンドの例については、74 ページの「detach コマンド」を参照してください。

空のボリューム要素

ストライプ、ミラー、連結、サブボリューム、およびボリュームを作成するときは、これらのボリューム要素を構成する子ボリューム要素を指定しないよう選択することもできます。子要素を指定しない場合は空のボリューム要素が作成され、後から接続できます。

論理ボリュームの統計

XVM ボリューム・マネージャは、物理ボリューム、サブボリューム、ストライプ、連結、ミラー、およびスライスの統計を維持できます。XVM ボリューム・マネージャの change コマンドの stat オプションを使用して、統計をオンまたはオフにしたり、ボリューム要素の統計をリセットできます。XVM が維持する統計についての詳細は、第 6 章、「統計データ」を参照してください。

クラスタ化された環境では、ローカル・セルの統計だけが維持されます。

スライスの作成

XVM 物理ボリュームのブロック範囲からスライスを作成するには、`slice` コマンドを使用します。スライスの開始ブロックを指定したり、スライスの長さを指定できます。さらに、以下のスライス作成方法を指定することもできます。

- 物理ボリュームのすべてのブロックからスライスを作成できます。
- 各部分が異なるスライスになるよう、指定したアドレス範囲を等しい部分に分割できます。
- 複数の物理ボリュームを一度にスライスできます。
- スライスがタイプ root、swap、または usr のシステム・スライスになるよう指定できます。システム・スライスの変更についての詳細は、85 ページの「slice コマンドを使ったシステム・ディスクの設定」を参照してください。

スライスの名前は自動的に付けられ、マシンを再起動しても変わりません。スライス名は変更できません。

連結の作成

連結は、すべての子ボリューム要素を 1 つのアドレス空間に連結するボリューム要素です。連結を作成するには、XVM ボリューム・マネージャの `concat` コマンドを使用します。

連結の作成時には、35 ページの「ボリューム要素の接続」で説明されているとおり、XVM ボリューム・マネージャによって接続のルールが強制的に適用されます。

ストライプの作成

ストライプは、ボリューム要素のセットを特定のアドレス空間全体にストライプするボリューム要素です。ストライプを作成するには、XVM ボリューム・マネージャの `stripe` コマンドを使用します。

等しくないサイズのボリューム要素で構成されるストライプを作成することもできますが、容量の大きい方のボリューム要素に未使用の容量が残ることになります。

ストライプの作成中には、35 ページの「ボリューム要素の接続」で説明されているとおり、XVM ボリューム・マネージャによって接続のルールが強制的に適用されます。

2 つのホスト・バス・アダプタにまたがるストライプの設定についての詳細は、24 ページの「XVM 論理ボリュームとフェイルオーバー」を参照してください。

ミラーの作成

ミラーは、すべての子ボリューム要素をミラー化するボリューム要素です。ミラーを作成するには、XVM ボリューム・マネージャの mirror コマンドを使用します。

複数の断片があるミラーを作成すると、ミラーが再生中であることを示すメッセージが SYSLOG に書込まれます。これは、システムによってデータのミラー化プロセスが開始されていることを示します。このプロセスが完了すると、別のメッセージが SYSLOG に書込まれます。何らかの理由で再生が失敗した場合は、SYSLOG のほかにシステム・コンソールにもメッセージが書込まれます。

大容量のミラー・コンポーネントでは、この再生プロセスに時間がかかることがあります。再生の必要のない新しいミラーを作成するときは、40 ページの「-clean ミラー作成オプション」で説明されているように、mirror コマンドの -clean オプションの使用を検討してください。再生の必要がないスクラッチ・ファイルシステムに使用する新しいミラーを作成するときは、40 ページの「-norevive ミラー作成オプション」で説明されているように、mirror コマンドの -norevive オプションの使用を検討してください。

ミラーの再生は、いったん開始したら、ミラーの断片の 1 つを除いたすべてを分離する方法以外では中止できません。ミラーの再生についての詳細は、第 7 章、「XVM ボリューム・マネージャの運用」を参照してください。

35 ページの「ボリューム要素の接続」で説明されているように、ミラーの作成中は、XVM ボリューム・マネージャによって接続のルールが強制的に適用されます。

ミラーを作成するときは、ミラーに対して以下の特性を設定できます。

- ミラーの読取りポリシー
- ミラーのプライマリ・レッグ
- 作成時にミラーを同期するかどうか（-clean オプション）
- システムの起動時にミラーを再同期するかどうか（-norevive オプション）

以下の節では、これらの各オプションについて説明します。

読取りポリシー

XVM ボリューム・マネージャでは、ミラーに対して以下のいずれかの読取りポリシーを指定できます。

ラウンドロビン　ミラーのメンバー間で I/O 負荷を分散し、ラウンドロビン方式で読取ります。

シーケンシャル　シーケンシャル I/O 操作をミラーの同じメンバーにルーティングします。

図 2-3 に、ラウンドロビン読取りポリシーでミラーのレッグからデータが読取られる方法を示します。くさび形の部分は、読取っているデータの単位を表します。最初の読取り操作で最初のレッグからデータの単位が取得され、2 番目の読取り操作で 2 番目のレッグから次のデータの単位が取得され、その次の読取り操作では 3 番目のレッグからその次のデータの単位が取得されます。その次の読取り操作では、再び最初のレッグから、要求したデータの単位が取得されます。

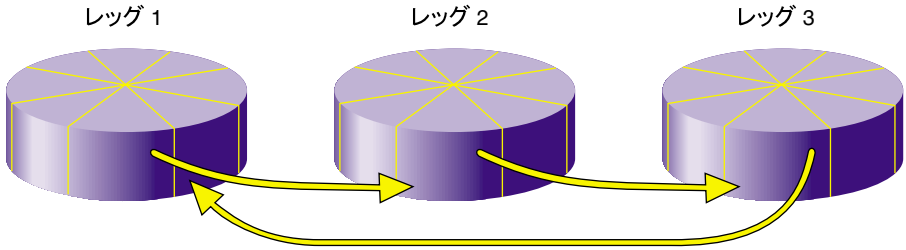

図 2-3　ラウンドロビン読取りポリシーでのミラーからのデータの読取り

図 2-4 に、シーケンシャル読取りポリシーでミラーのレッグからデータが読取られる方法を示します。この図からわかるように、単一のシーケンシャル I/O 操作では、別のミラー・メンバーはアクセスされません。

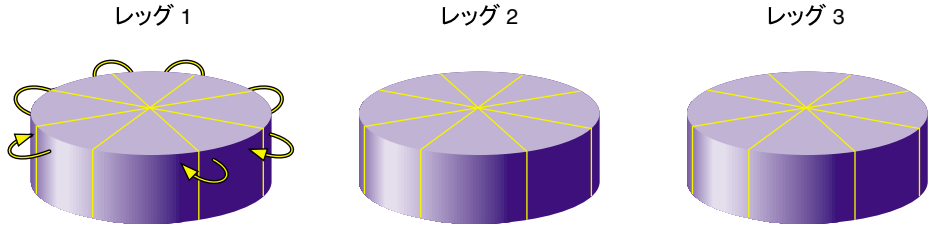

図 2-4　シーケンシャル読取りポリシーでのミラーからのデータの読取り

ミラーを定義したら、change コマンドを使用して、読取りポリシーを変更できます。

プライマリ・レッグ

ミラーの特定のレッグをプライマリ・レッグとしてマークすることによって、読取りにそのレッグを優先的に使用するかどうかを指定できます。ミラーの定義後は、change コマンドを使用して、レッグをプライマリ・レッグにするかどうかを再定義できます。

-clean ミラー作成オプション

ミラーを作成するときに、mirror コマンドの -clean オプションを使用して、作成時にミラーのレッグの再生が必要ないことを指定できます。このオプションは、ミラーのレッグがすでにミラー化されている場合や、ミラーが新しく、読取りよりも先にまずすべてのデータの書込みが行われる場合に便利です。

mirror コマンドの -clean オプションと -norevive オプションは同時に使用できません。

-norevive ミラー作成オプション

ミラーを作成するときに、mirror コマンドの -norevive オプションを使用して、システムの起動時にミラーのレッグの再生が必要ないことを指定できます。このオプションは、/tmp や swap などのスクラッチ・ファイルシステム用のミラーを作成している場合に便利です。

mirror コマンドの -clean オプションと -norevive オプションは同時に使用できません。

ボリュームの作成

XVM 論理ボリュームを明示的に作成するには、XVM ボリューム・マネージャの volume コマンドを使用します。33 ページの「ボリュームおよびサブボリュームの自動作成」で説明されているように、ボリュームは、ボリューム要素を作成したときに自動的に作成することもできます。

volume コマンドを使用してボリュームを作成する場合は、作成後にボリュームに接続するサブボリュームを指定できます。サブボリュームをボリュームに接続するときは、35 ページの「ボリューム要素の接続」で説明されているように、XVM ボリューム・マネージャによって接続のルールが強制的に適用されます。

サブボリュームの作成

サブボリュームを明示的に作成するには、XVM ボリューム・マネージャの subvolume コマンドを使用します。33 ページの「ボリュームおよびサブボリュームの自動作成」で説明されているように、タイプ data のサブボリュームは、ボリューム要素を作成したときに自動的に作成することもできます。

subvolume コマンドを使用してサブボリュームを作成する場合は、サブボリュームに接続するボリューム要素を指定できます。サブボリュームに接続するボリューム要素には、ボリュームやサブボリュームは指定できません。接続するボリューム要素を指定しなかった場合は、空のサブボリュームが作成されます。

subvolume コマンドを使用してサブボリュームを作成する場合は、サブボリューム・タイプを指定できます。システム定義サブボリューム・タイプまたはユーザ定義サブボリューム・タイプを指定できます。システム定義サブボリューム・タイプには、以下の 3 つがあります。

data	data サブボリューム
log	log サブボリューム
rt	real-time サブボリューム

特定のボリュームに同じシステム定義タイプを持つ複数のサブボリュームを含めることはできません。また、システム定義サブボリューム・タイプにはユーザ定義名を指定できません。

ユーザ定義サブボリューム・タイプの範囲は、16 ～ 255 です（0 ～ 15 はシステム定義タイプ用に予約されています）。システム定義サブボリューム・タイプは、アプリケーション依存タイプとサブボリュームを関連付けるために使用されます。1 つのボリュームで特定のユーザ定義タイプのサブボリュームを複数指定することはできません。

論理ボリュームの再構成

論理ボリュームを作成するときに、XVM ボリューム・マネージャの attach コマンドと detach コマンドを使用して、その要素を構成したり再構成できます。さらに、ボリューム要素を作成した後に、XVM ボリューム・マネージャの remake コマンドを使用してボリューム要素を再構成できます。remake コマンドは、ボリューム要素の穴を取除いたり、ボリューム要素の下位にある断片を再度並べ替えます。attach コマンドと detach コマンドを連続して実行する代わりに、remake コマンドを一度だけ実行すればよいので便利です。

論理リソースの管理

論理リソースを作成したら、以下のタスクを実行できます。

- ボリューム要素の表示
- ボリューム要素の無効化
- ボリューム要素のオンライン化
- オンラインでのボリューム要素の変更
- 論理ボリューム設定の保存

以下の節では、これらの手順の概要を説明します。

ボリューム要素の表示

ボリューム要素に関する情報を表示するには、XVM ボリューム・マネージャの `show` コマンドを使用します。

ボリューム要素は、以下の 1 つまたは複数の状態になる可能性があります。

online　ボリューム要素はオンラインで、正しく設定されており、開くことができるか、すでに開かれています。

offline　ボリューム要素はオフラインで、このボリューム要素に対して I/O を実行できません。ボリューム要素がこの状態の場合は、ボリューム要素のトポロジーをチェックして、ボリューム要素の各断片の状態に注意します。ボリューム要素がオフラインである理由を判断するために役立つほかの状態が、常に少なくとも 1 つは表示されます。

mediaerr　ボリューム要素で少なくとも 1 つのメディア・エラーが発生しています。

inconsistent　最後に異常が発生してからボリューム要素の 1 つまたは複数の断片が変更された可能性があります。要素が接続または分離されているか、見つからない断片が元の場所に戻されているか、状態が変更されている場合があります。

ボリューム要素が「inconsistent」状態の場合は、`show` コマンドの `-v` オプションを使用して、ボリューム要素の断片のタイムスタンプを表示できます。「inconsistent」状態の断片は、タイムスタンプが異なる断片です。維持する方の正しい断片を選択して他方の断片を分離するか、`remake` コマンドを使用して現在の設定をそのまま使用します。

tempname　再起動すると変わる可能性がある名前がボリューム要素に付いています。

reviving:queued
: （ミラーのみ）ミラーは再生の対象となっていますが、再生はまだ開始されていません。

reviving:*XX*%
: （ミラーのみ）システムはこのミラーの再生中で、*XX*% が完了しています。

disabled
: `change disable` コマンドを使用してボリューム要素が無効に設定されています。ボリューム要素をオンラインにするには、`change enable` を使用して明示的に有効にする必要があります。

incomplete
: ボリューム要素の 1 つまたは複数の断片が見つかりません。ミラー以外のすべてのボリューム要素をオンラインに戻すには、見つからない断片を接続するか、`remake` コマンドを使用してボリューム要素を再作成する必要があります。

pieceoffline
: ボリューム要素にオフラインの断片があります。ミラー以外のすべてのボリューム要素をオンラインに戻すには、オフラインの断片をオンラインに戻す必要があります。

open
: ボリューム要素は、開いているサブボリュームの一部です。

valid
: ボリューム要素は最新で、データは読取り可能です。

clean
: ミラー・レッグは、`mirror` コマンドの `-clean` オプションを使用して作成されており、作成時に再生の必要がないことが指定されています。したがって、以降の起動時に再生されます。

ボリューム要素の無効化

ボリューム要素を手動で無効にするには、XVM ボリューム・マネージャの `change` コマンドを使用できます。ボリューム要素を無効にすると、同じく `change` コマンドを使用して要素を明示的に有効にするまで、そのボリューム要素に対して I/O を実行できません。明示的に有効にしないかぎり、オブジェクトはマシンを再起動しても無効のままです。

ボリューム要素のオンライン化

ボリューム要素がシステム・カーネルによって無効にされて、オフラインになることがあります。ミラー・メンバーに I/O エラーが表示されているような場合、この状態になることがあります。ボリューム要素を再びオンラインにするには、XVM ボリューム・マネージャの `change` コマンドを使用できます。

オンラインでの変更

XVM ボリューム・マネージャの insert コマンドを使用して、別のボリューム要素上にミラーまたは連結を挿入できます。このコマンドは、ボリューム要素を拡張したり、実行中のシステムにミラーを追加する場合に使用できます。これは、挿入するボリューム要素が、開いているサブボリュームの一部で、アクティブな I/O が行われている可能性があるためです。

XVM ボリューム・マネージャの collapse コマンドを使用して、ツリーから層を削除できます。一般的に、前の挿入操作を元に戻すには、collapse コマンドを使用します。

XVM 設定の保存と再生成

XVM 論理ボリューム設定を保存するには、XVM ボリューム・マネージャの dump コマンドを使用して、設定を再生成するコマンドをファイルにダンプします。これにより、システムを再起動せずに、実行中のシステムのディスクを交換したり、新しいディスク上に XVM 設定を再生成できます。

ダンプを実行してデバイスを再生成するときは、交換するディスクのデータを再生成するのではなく、新しいディスク上に XVM 設定を再生成します。

78 ページの「ボリューム要素の再構築: dump コマンドの使用」で説明されているように、コマンドをダンプしてボリューム要素ツリーを再生成する場合は、コマンドを個別かつ明示的にダンプして、ツリーにつながる物理ボリュームを再生成しなければならないことに注意してください。

論理リソースの破損

ボリューム要素を削除するには、XVM ボリューム・マネージャの delete コマンドを使用します。削除されたボリューム要素の親はそのまま残り、スロットが開かれた状態となります。

一般的に、子が接続されている場合は、ボリューム要素を削除できません。ただし、-all オプションまたは -nonslice オプションを使用して、すべての子またはスライス以外のすべての子が削除されるようにそれぞれ指定することは可能です。-nonslice オプションを指定した場合は、スライスが分離され、そのスライスに対してボリュームおよび data サブボリュームが自動的に生成されます。

削除するボリューム要素が開いているサブボリュームの一部である場合、削除によってサブボリュームの状態がオフラインになることはありません。delete コマンドには、この制約をオーバーライドする -force オプションが用意されています。

XVM システム・ディスクの swap パーティションは削除できません。これは、swap パーティションを誤って削除した場合にシステム・パニックが発生するのを防止するためです。システム・ディスクの論理ボリュームを削除して、システムからラベルを削除する必要がある場合は、91 ページの「XVM システム・ディスクの削除」の説明に従って、まず XVM の `change` コマンドを使用してそのディスクをシステム・ディスクから option ディスクに変更しなければなりません。

XVM コマンド・ライン・インタフェース

XVM の管理タスクは、XVM コマンド・ライン・インタフェース (CLI: Command Line Interface) を使用して実行します。この章では、XVM CLI と、このインタフェースが提供する機能について説明します。この章の主な節は、以下のとおりです。

- 「XVM CLI の使用」
- 「XVM CLI コマンドのオンライン・ヘルプ」(49 ページ)
- 「XVM CLI の構文」(50 ページ)
- 「XVM でのオブジェクト名」(51 ページ)
- 「XVM デバイス・ディレクトリとパス名」(56 ページ)
- 「コマンド出力とリダイレクト」(57 ページ)
- 「安全なコマンドと危険なコマンド」(58 ページ)

XVM CLI の使用

XVM CLI を使用するには、以下のコマンドを入力します。

```
# xvm
```

このコマンドを入力したときにクラスタ・サービスが有効になっていると、以下の XVM CLI プロンプトが表示されます。

```
xvm:cluster>
```

このプロンプトは、現在のドメインがクラスタであることを示しています。このドメインで作成したすべてのオブジェクトは、クラスタのどのノードでも管理できます。

XVM クラスタ設定オブジェクトを表示したり、これらのオブジェクトにアクセスするには、クラスタ・サービスを開始する必要があります。クラスタ設定された IRIX システムでこのコマンドを入力したときにクラスタ・サービスが有効になっていない場合は、以下の XVM CLI プロンプトが表示されます。

```
# xvm
Notice: Cluster services have not been enabled on this cell yet.You
will only be able to manipulate local objects until cluster services
are started.
xvm:local>
```

このプロンプトは、現在のドメインがローカルであることを示しています。このドメインで作成したすべてのオブジェクトは、現在のノードだけで管理できます。

XVM コマンドの -domain オプションを使用して、XVM ボリューム・マネージャの起動時に XVM ドメインを指定できます。

```
# xvm -domain domain
```

domain 変数には、local または cluster を指定できます。ローカル・ドメインで単一のコマンドを実行するスクリプトを作成している場合は、XVM コマンドを実行するドメインを変更するためにこのオプションが役に立つことがあります。

クラスタ・ドメインで XVM ボリューム・マネージャを実行しているときは、ローカル physvol のオーナーであるノードから実行していても、そのクラスタ・ドメイン内にもある XVM の physvol しか参照および変更できません。ローカル・ディスクを参照および変更するには、set domain コマンドを使用してドメインをローカルに変更するか、physvol 名を指定するときに local: プレフィックスを使用します。同様に、ローカル・ドメインで XVM ボリューム・マネージャを実行する場合は、ドメインをクラスタに変更するか、クラスタによって所有されている physvol を指定するときに cluster: プレフィックスを指定しなければなりません。

たとえば、クラスタ・ドメインで実行しているときに、ローカル・ドメインの XVM 物理ボリュームを参照したい場合は、以下の形式を使用できます。

```
xvm:cluster> show local:phys/*
```

同様に、ローカル・ドメインで実行しているときに、自分がメンバーであるクラスタの XVM 物理ボリュームを参照したい場合は、以下の形式を使用できます。

```
xvm:local> show cluster:phys/*
```

XVM ドメインについての詳細は、26 ページの「XVM ドメイン」を参照してください。

コマンド・プロンプトが表示されたら、XVM CLI コマンドを入力して、XVM 論理ボリュームの設定や管理を行うことができます。これらのコマンドは、入力と同時にインタラクティブに実行されます。

XVM 論理ボリュームを設定するには、root としてログインしている必要があります。ただし、root 特権がなくても論理名の設定情報を表示することは可能です。XVM CLI コマンドの実行を完了したら、exit コマンドを入力してシェルに戻ります。exit コマンドのエイリアスとして、bye または quit も使用できます。

また、以下のようにコマンドの前に xvm を付けることで、個々の XVM コマンドをシェルから直接入力することもできます。

```
# xvm [command ...]
```

以下のように、入力を UNIX の標準ツールにリダイレクトする場合と同様に、XVM コマンドのファイルを XVM CLI にリダイレクトできます。

```
# xvm < myscript
```

または、以下を入力できます。

```
# cat myscript | xvm
```

シェル置換を使用して、あるコマンドの出力を別のコマンドに入力する場合についての詳細は、57 ページの「コマンド出力とリダイレクト」を参照してください。

XVM CLI コマンドのオンライン・ヘルプ

XVM CLI には help コマンドが含まれており、このコマンドを使用して XVM CLI コマンドの構文を表示できます。help コマンドのエイリアスとして、疑問符 (?) を使用できます。

引数を指定せずに help コマンドを実行すると、サポートされているコマンドのリストが表示されます。help コマンドに続けてキーワードを入力すると、指定したコマンドの概要が表示されます。キーワードの前に -verbose オプションを指定すると、すべてのコマンド・オプションと同時にコマンドの例も示される詳細なヘルプを表示できます。

以下のコマンドを実行すると、slice コマンドの概要が表示されます。

```
xvm:cluster> help slice
```

以下のコマンドを実行すると、slice コマンドの詳細なヘルプが表示されます。

```
xvm:cluster> help -verbose slice
```

help コマンドで使用できるキーワードは、xvm(1M) のマン・ページで概要が説明されている XVM CLI コマンドです。さらに、以下の help コマンドも入力できます。

`help names` XVM オブジェクト名に関する情報を表示します。

`help regexp` XVM オブジェクト名を指定するときに使用可能な正規表現に関する情報を表示します。

XVM CLI の構文

XVM CLI コマンドは、任意の一意のサブストリングに省略可能なキーワードです。コマンドでは大文字と小文字は区別されません。

コマンド・オプションはマルチキャラクタのキーワードで、コマンドでサポートされているオプションの中で一意のサブストリングに省略できます。コマンド自体と同様に、オプションでも大文字と小文字は区別されません。

たとえば、以下のコマンドを入力できます。

```
xvm:cluster> volume -volname mainvol vol1/data vol2/log vol3/rt
```

または、以下の省略形を入力できます。

```
xvm:cluster> vol -vol mainvol vol1/data vol2/log vol3/rt
```

同様に、以下の完全なコマンドも入力できます。

```
xvm:cluster> show -verbose slice/freds0
```

以前に使用したコマンドは、以下のように省略できます。

```
xvm:cluster> show -v slice/freds0
```

XVM コマンドを入力するときは、以下の構文規則と機能が適用されます。

- vol、stripe、concat、mirror、raid、slice、phys、unlabeled、および subvol キーワードは XVM CLI によって予約されており、オブジェクトの名前の指定には使用できません。
- オブジェクト名は、英数字、ピリオド (.)、アンダースコア (_)、およびハイフン (-) で構成されます。

- オブジェクト名を数字で始めることはできません。
- <> の中の XVM CLI トークンは、コメントとして解釈されます。
- 行末のバックスラッシュ (\ ) は、継続文字として機能します。
- 空白行と、シャープ記号 (#) で始まる行は無視されます。
- コマンドの先頭に感嘆符 (!) を付けると、そのコマンドはシェルに渡されます。

XVM でのオブジェクト名

スライス以外の XVM オブジェクトにはユーザ定義名を使用できます（ユーザ定義名が指定されていない場合は、デフォルトの名前が生成されます）。XVM オブジェクトの名前の長さは 32 文字までに制限されており、数字で始めることはできません。

サブボリューム以外のオブジェクトは、オブジェクト名で指定します。サブボリューム・オブジェクトは、サブボリューム名の前にボリューム名とスラッシュ (/) のプレフィックスを付けて指定しなければなりません。たとえば、`fred/data` のように指定します。この例では、`fred` がボリュームの名前で、`data` がサブボリュームの名前です。

以下の節では、XVM オブジェクトのさまざまな指定方法について説明します。以下のトピックが含まれます。

- XVM オブジェクトの指定
- 断片を使った構文
- XVM オブジェクト名の例
- 正規表現

XVM オブジェクトの指定

XVM オブジェクトは、以下のいずれかの形式のパスに似た構文で指定します。*objname* にはオブジェクトの名前を、*vepath* には特定のボリューム要素から別のボリューム要素へのパスをそれぞれ指定します。

[*domain*:][*type*/]*objname*

[*domain*:][*type*/]*vepath*

ドメイン・オプションには、ローカルまたはクラスタを指定できます。47 ページの「XVM CLI の使用」で説明したように、ドメイン・オプションは、現在のドメインに存在しない XVM オブジェクトを指定するときに含めます。

ボリューム要素に対して「..」というパス・コンポーネントを指定すると、ボリューム要素の親を指定したことになります。たとえば、`show slice/foos0/..` コマンドを実行すると、スライス `foos0` の親が表示されます。

ユーザ定義名を使用できるので、XVM オブジェクト名前空間があいまいになる可能性があります。ワイルドカードを使用せずに XVM コマンドにあいまいな名前を入力した場合、コマンドを実行すると、通常はエラー・メッセージが表示されます。ワイルドカードの使用についての詳細は、55 ページの「正規表現」を参照してください。

オブジェクト名があいまいになるのを避けるために、`concat/concat1` のように、オブジェクト名の前にオブジェクト・タイプとスラッシュのプレフィックスを付けることができます（`concat1` という名前でオブジェクト・タイプが異なる 2 つのオブジェクトが存在する場合、`concat1` を指定しただけでは十分にオブジェクトを識別できません）。オブジェクト・タイプを指定すると、オブジェクトのタイプに関する情報が提供されるので、名前解決も高速化できます。

あいまいではないサブボリュームは、`subvol`/*volname*/*subvol_name* のように 3 つの要素で構成されます。

オブジェクト・タイプの指定には、以下の表に示すプレフィックスを使用できます。また、表に続いて、`phys`、`unlabeled`、および `foreign` について説明します。

表 3-1 オブジェクト・タイプを指定するプレフィックス

プレフィックス	オブジェクト・タイプ
`vol`	ボリューム ve
`subvol`	サブボリューム ve
`concat`	連結 ve
`stripe`	ストライプ ve
`mirror`	ミラー ve
`slice`	スライス ve
`phys`	physvol

表 3-1 オブジェクト・タイプを指定するプレフィックス

プレフィックス	オブジェクト・タイプ
unlabeled	ラベルの付いていないディスク
foreign	外部ディスク

physvol は、XVM ボリューム・マネージャによって XVM 物理ボリュームとしてラベルが付けられていて、システムによって検査済みのディスクです。たとえば、phys/fred は、fred という名前の XVM physvol を指します。名前のパスの部分は、physvol にラベルを付けたときに指定された名前です。

ラベルの付いていないディスクは、XVM ラベルが付いていないディスク、または XVM ラベルが付いていても XVM サブシステムによって検査されていないディスクです。give または steal コマンドを使用して現在のオーナーに移動したディスクには、明示的に probe コマンドを実行して検査するか、またはシステムの再起動中に検査されるまで、ラベルは付けられません。

ラベルの付いていないディスクのパスの部分は、ボリューム・パーティションへのファイルシステム・パスになります。パスには、明示的なパス（unlabeled/hw/rdisk/dks0d4vol など）または相対パス（unlabeled/dks0d4vol など）を指定できます。SAN ディスクのパスでは、複数のコンポーネント（unlabeled/2000006016fe057a/lun4vol/c11p0 など）が使用されます。

外部ディスクは、別のノードまたは別のクラスタによって所有されているために、現在のオーナーでは管理できない XVM ディスクです。外部ディスクのパスの部分の形式は、ラベルの付いていないディスクのパスの部分と同じです。

断片を使った構文

XVM ボリューム要素は、パスに似た構文で指定することもできます。この場合、パスのコンポーネントは、ve 名または親の下位での断片番号になります。たとえば、vol/fred/data/concat0/phys0 は、親が concat0 の ve phys0 を参照しており、ボリューム fred の data サブボリュームです。また、concat0/0 は、ve concat0 の、番号がゼロの子（断片）を指します。断片を使った構文は、名前がわからないボリューム要素を指定する場合に便利です。

図3-1 に、システム生成名を使った XVM 論理ボリュームのレイアウトを示します。

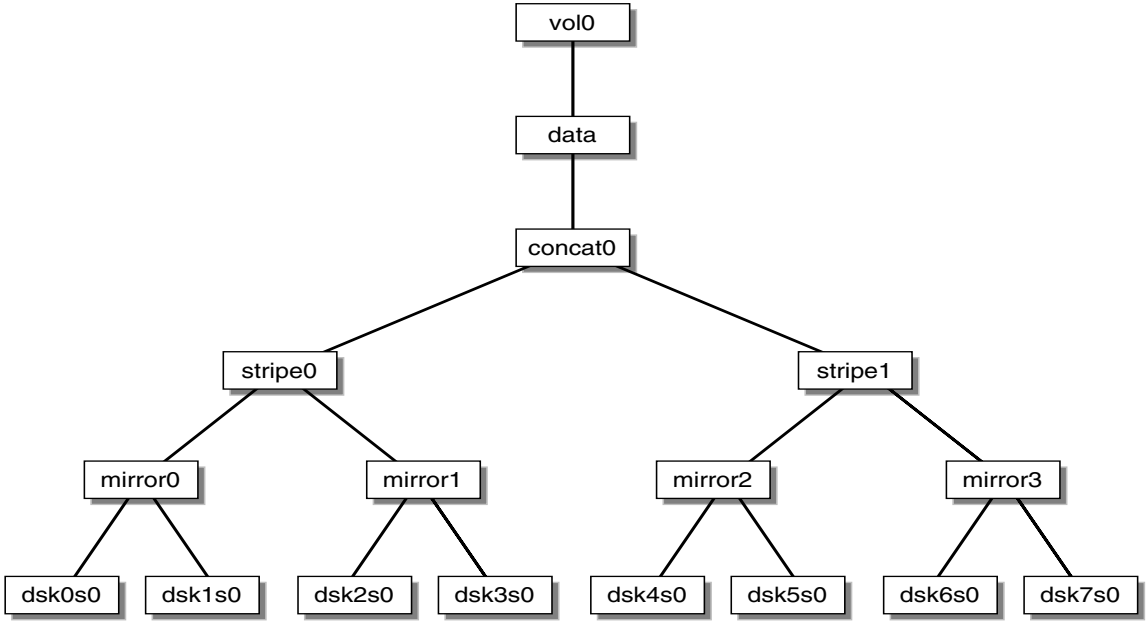

図 3-1 システム生成名を使った XVM 論理ボリューム

表 3-2 に、断片を使った構文で図 3-1 の XVM 論理ボリュームの個々のボリューム要素を指定する方法の例を示します。

表 3-2 断片を使った構文による論理ボリューム要素の指定

XVM オブジェクト	別のオブジェクト指定方法
concat/concat0	vol0/data/0
stripe/stripe0	vol0/data/0/0
stripe/stripe1	vol0/data/0/1
mirror/mirror0	vol0/data/concat0/stripe0/0 vol0/data/0/0/0
mirror/mirror2	vol0/data/concat0/stripe1/0 vol0/data/0/1/0
slice/dsk6s0	vol0/data/0/1/1/0 vol0/data/concat0/1/mirror3/0

XVM オブジェクト名の例

以下に、さまざまな XVM オブジェクトの指定方法の例を示します。

`vol0` `vol0` という名前のオブジェクト

`unlabeled/dks0d4`
コントローラ 0、ドライブ 4 上の名前の付いていないディスク

`mirror/mirror6`
`mirror6` という名前のミラー・ボリューム要素

`concat0/0` `concat0` の最も左側の断片

`vol0/data/0` ボリューム `vol0` の data サブボリュームの子

`stripe0/freds0`
ストライプ `stripe0` の下位にある `freds0` という名前のボリューム要素

`unlabeled/2000006016fe0ed0/lun3vol/clip0`
ボリューム・パーティション・パスが
`/dev/rdsk/2000006016fe0ed0/lun3vol/clip0` の SAN ディスク

`foreign/dks5d43vol`
ディスク `dks5d43vol`。このディスクは、このオブジェクト名を表示しているマシンでは管理できません。

正規表現

XVM CLI コマンドでオブジェクト名を指定するときは、正規表現を使用できます。使用できる文字はワイルドカード (*)、?、および [ ] で、各文字は、fnmatch(3G) 関数によってサポートされている標準的な正規表現の意味を持ちます。XVM CLI コマンドで正規表現を使用できるのは、XVM オブジェクト・パスの最も右側のオブジェクトだけです。

以下に、XVM CLI コマンドでの正規表現の使用方法の例を示します。

`*` すべてのオブジェクトに一致

`vol/*` すべてのボリュームに一致

`slice/*s0` `s0` で終わるすべてのスライスに一致

`concat0/*` `concat0` のすべての子に一致

`subvol/log*` すべての log サブボリュームに一致

`fred*` `fred` で始まるすべてのオブジェクトに一致

unlabeled/dks[34]d*
: コントローラ 3 および 4 上のすべての名前の付けられていないディスクに一致

/ で終わるボリューム要素パスを指定することは、/* で終わるボリューム要素パスを指定することと同じです。

オブジェクト・タイプ・キーワードで構成されるボリューム要素パスは、キーワードの後に /* を指定することと同じです。たとえば、show slice コマンドは、show slice/* コマンドと同じです。

XVM デバイス・ディレクトリとパス名

XVM 論理ボリュームは、以下のディレクトリに含まれます。

/dev/cxvm
: CXFS クラスタで使用される XVM 論理ボリューム用のブロック特殊ファイル

/dev/rcxvm
: CXFS クラスタで使用される XVM 論理ボリューム用のキャラクタ特殊ファイル

/dev/lxvm
: ホストのローカル・ボリュームに使用される XVM 論理ボリューム用のブロック特殊ファイル

/dev/rlxvm
: ホストのローカル・ボリュームに使用される XVM 論理ボリューム用のキャラクタ特殊ファイル

クラスタ環境で実行していない場合、論理ボリュームはすべてローカル・ボリュームと見なされます。

これらのディレクトリでの XVM 論理ボリュームのデバイス名は、*volname_subvolname* の形式になります。*volname* は XVM 論理ボリュームの名前、*subvolname* は、そのボリュームの下位にあってアクセスされるサブボリュームの名前になります。data サブボリュームを参照するときは、*volname_subvolname* の代わりにオプションで *volname* という短い名前も使用できます。たとえば、/dev/cxvm/fred_data と /dev/cxvm/fred のどちらも、ボリューム fred の下位にある data サブボリュームのブロック特殊ファイルを参照します。

メモ: XVM ボリューム・マネージャの以前のリリースでは、論理ボリュームのブロック特殊ファイルとキャラクタ特殊ファイルは、それぞれ /dev/xvm と /dev/rxvm に格納されていました。これらのディレクトリを指定すると、起動時にシステムがクラスタ環境で実行されていた場合はクラスタ・ディレクトリにリンクされ、その他の場合はローカル・ディレクトリ内の論理ボリュームにリンクされます。ローカル・ボリュームとクラスタ・ボリュームの混同を避けるため、このような古いディレクトリ・リンクを使用することは推奨しません。

XVM 論理ボリューム内のオブジェクトの名前についての詳細は、51 ページの「XVM でのオブジェクト名」を参照してください。

コマンド出力とリダイレクト

通常は、オブジェクトの作成や操作を行うコマンドを実行すると、コマンドが正常に完了したときに、作成されたオブジェクトやターゲット・オブジェクトの名前が出力されます。以下に例を示します。

```
xvm:cluster> concat -tempname slice/wilmas0 slice/barneys0
</dev/cxvm/vol0> concat/concat0
```

シェル置換を使用してコマンドの出力を別のコマンドに入力できます。たとえば、Korn シェルで以下のコマンドを実行すると、fred というボリューム名の連結ボリューム要素と、そのコンポーネントとして physvol phys1 および phys2 が作成されます。

```
$ xvm concat -volname fred $(xvm slice -all phys1) $(xvm slice -all \
phys2)
```

csh または sh では、このコマンドの構文は以下のようになります。

```
$ xvm concat -volname fred `xvm slice -all phys1` `xvm slice -all \
phys2`
```

失敗したコマンドや、操作対象のオブジェクトが無効になるコマンド（delete など）では、ターゲット・オブジェクト名は出力されません。

先に説明したように、ボリューム要素の作成や変更を行うコマンドでは、<> 記号の中に、ターゲット ve が属するサブボリュームのブロック特殊名も表示されます。たとえば、slice -all phys1 コマンドを実行すると、以下の出力が生成されます（コマンドが成功した場合）。slice/phys1s0 は作成されたスライスの名前で、/dev/cxvm/phys1s0 は、このスライスにアクセスするときに開くことができるサブボリュームのブロック特殊ファイルのパスです。

```
</dev/cxvm/phys1s0> slice/phys1s0
```

<> 記号の中に表示されるトークンは、CLI ではコメントとして扱われます。これにより、ボリューム要素を作成するコマンドでブロック特殊名が表示されても、その出力は <> 記号に囲まれているので、あるコマンドの出力を別のコマンドに入力するときに CLI によって無視されるようになります。

安全なコマンドと危険なコマンド

XVM コマンドは、安全なものと危険なものに分けて考えることができます。危険なコマンドとは、ve の上位にあるサブボリュームのアドレス空間に何らかの影響を及ぼすコマンドで、連結 ve の子の分離や削除を行うようなコマンドのことです。一方、安全なコマンドとは、ミラー ve の最後の子以外をすべて分離または削除するようなコマンドのことで、サブボリュームのアドレス空間に影響を及ぼしません（最後の子を分離または削除することは危険です）。

安全なコマンドは、影響を受けるサブボリュームが開いている状態かどうかに関係なくいつでも発行できますが、危険なコマンドは、サブボリュームが開いていない場合にしか発行できません。マウントされているサブボリュームは常に開いていますが、マウントされていなくてもサブボリュームが開いていることがあります。たとえば、アプリケーションが raw サブボリュームにアクセスしている場合などです。

サブボリュームを開く危険なコマンドを実行するとデフォルトでエラーが発生しますが、コマンドによっては、このような動作をオーバーライドするための `-force` オプションが用意されています。逆に、`-safe` オプションを持つコマンドもあり、このオプションを指定すると、サブボリュームが開いていない場合でも安全かどうかが強制的にチェックされます。

XVM 管理コマンド

この章では、xvm コマンド・ライン・インタフェース (CLI: Command Line Interface) コマンドの概要について説明し、各コマンドの例を示します。各コマンドの構文の詳細は、49 ページの「XVM CLI コマンドのオンライン・ヘルプ」に説明されているとおり、help コマンドを使用して参照できます。

この章には、以下のトピックに関する節が含まれています。

- 「物理ボリューム・コマンド」
- 「論理ボリューム・コマンド」（67 ページ）
- 「XVM システム・ディスク」（79 ページ）

物理ボリューム・コマンド

以下のコマンドを使用して、XVM 物理ボリュームを作成、管理、および削除できます。

- set コマンドは、デフォルトの XVM ドメインを変更するために使用できます。
- label コマンドは、XVM ボリューム・マネージャにディスクを割当てるために使用できます。
- show コマンドは、XVM 物理ボリュームを表示するために使用できます。
- change コマンドは、XVM 物理ボリュームを変更するために使用できます。
- probe コマンドは、XVM 物理ボリュームを検査するために使用できます。
- dump コマンドは、XVM 物理ボリュームを再生成するために使用できます。
- give コマンドは、XVM 物理ボリュームのドメインを変更するために使用できます。

- 物理ボリュームを現在所有しているノードまたはクラスタで give コマンドを実行できない場合は、steal コマンドを使用して、XVM 物理ボリュームのドメインを変更できます。
- unlabel コマンドは、XVM ボリューム・マネージャからディスクを削除するために使用できます。

以下の節では、これらのコマンドの概要について説明します。

set コマンドを使った現在のドメインの変更

XVM の CLI コマンドを実行しているときに現在の XVM ドメインを変更するには、set コマンドを使用します。現在のドメインは、ローカルまたはクラスタの場合があります。現在のドメインがローカルの場合、作成した XVM オブジェクトは実行している現在のノードに属します。現在のドメインがクラスタの場合、作成した XVM オブジェクトは、現在のノードが属しているクラスタに属します。現在のドメインは、xvm プロンプトの一部として xvm:cluster> または xvm:local> のように表示されます。また、local パラメータまたは cluster パラメータを指定せずに、set コマンドを実行して現在の XVM ドメインを表示することもできます。

クラスタ・サービスが開始されていない場合は、ドメインをクラスタに設定できません。

以下の例では、現在のドメインをクラスタからローカルに変更します。

```
xvm:cluster> set domain local
```

XVM ドメインについての詳細は、26 ページの「XVM ドメイン」を参照してください。

label コマンドを使った XVM ボリューム・マネージャへのディスクの割当て

XVM ボリューム・マネージャにディスクを割当てるには、label コマンドを使用します。label コマンドは、XVM 物理ボリューム・ラベルをディスクに書込んだり、変更します。クラスタ化された環境では、作業しているシステムに接続されていないディスクにラベルを付けることはできません。

XVM 物理ボリュームに名前を割当てるには、-name オプションを使用します。名前を指定しなかった場合、デフォルトの名前は、ラベルの付いていないディスクのパスの名前と同じになります（dks0d1 など）。複数のディスクを XVM に割当てているときに名前を指定すると、その名前は、各物理ボリューム名のプレフィックスとして使用され、一意な数字の接尾辞が付けられます。複数のディスクを割当てているときに名前を指定しないと、ラベルの付いていないディスクのパスが各物理ボリューム名のプレフィックスとして使用されます。

XVM ボリューム・マネージャにディスクを割当てると、そのディスクはデフォルトで XVM の option ディスクになります。このディスクには、label コマンドの -type オプションを使用して、XVM システム・ディスクとしてラベルを付けることができます。また、既存の XVM システム・ディスクのミラーであるシステム・ディスクとしてラベルを付けることもできます。XVM ディスクにシステム・ディスクまたはシステム・ディスクのミラーとしてラベルを付けることは、label コマンドの特殊なケースです。これは、このようにラベルを付けると、XVM ボリューム・マネージャにディスクが割当てられると同時に、root ファイルシステムおよび swap ファイルシステム用の論理ボリュームもそのディスクに作成されるからです。XVM システム・ディスクの作成およびミラー化についての詳細は、79 ページの「XVM システム・ディスク」を参照してください。

マウントされているファイルシステムとして現在使用中のパーティションがディスクに含まれている場合、このディスクに XVM ディスクとしてラベルを付けることはできません。複数のディスクがあるシステムでは、使用中のパーティションがディスクに含まれているかどうかのチェックに時間がかかる可能性があります。label コマンドには、この制約をオーバーライドする -nopartchk オプションが用意されています。使用中のパーティションが含まれているディスクにラベルを付けると、データが破壊されたり、システム・パニックが発生する場合があるため、-nopartchk オプションの使用には注意が必要です。

XVM ディスクにラベルを付ける場合は、-volhdrblks オプションを使用することで、ボリューム・ヘッダに割当てる容量を指定できます。デフォルト値は、ラベルを付けるディスクのボリューム・ヘッダに現在存在するブロック数になります。-xvmlabelblks オプションを使用すると、ボリューム・ヘッダの XVM ラベル領域に割当てる容量を指定できます。デフォルトは 1024 ブロックになります(通常は、XVM ラベル領域の一部ではない 3072 ブロックがボリューム・ヘッダに残ります)。

-volhdrblks オプションおよび -xvmlabelblks オプションの通常のデフォルト値は、約 5000 の XVM オブジェクトをサポートしており、ほとんどの XVM 論理ボリューム設定ではこれで十分です。ラベル領域が維持しなければならないオブジェクトが XVM の physvol にこれ以上ある場合は、XVM ラベル領域サイズを増やす必要があります。一般的には、7 つのオブジェクトに対して 1 つのブロックが必要です。ボリューム要素およびボリューム要素の名前は 2 つのオブジェクトとしてカウントされることに注意してください。

ほとんどの XVM 論理ボリューム設定では、XVM ラベル領域のデフォルトのサイズで十分ですが、ボリューム・ヘッダをデフォルト値から減らし、それに相当する分 XVM ラベル領域を増やすことによって、XVM ラベル領域サイズを増やすことができます。たとえば、デフォルトのオプションでは、XVM ラベル領域に 1024 ブロックが割当てられ、ボリューム・ヘッダには通常 3072 ブロックが割当てられます。それに続き、ユーザ・データがブロック 4096 から開始されます。ボリューム・ヘッダ・ブロックの数を 2048 に設定した場合は、XVM ラベル・ブロックの数を 2048 に設定できます。これによって、XVM ラベル領域が 2 倍になります。ボリューム・ヘッダ領域はデフォルトよりも減りますが、ユーザ・データは同じくブロック 4096 から開始されます。

以下の例では、dks0d3 に XVM 物理ボリューム fred としてラベルを付けます。

```
xvm:cluster> label -name fred dks0d3
```

以下の例では、dks0d3 に XVM 物理ボリューム fred としてラベルを付け、XVM ラベルの 4096 ブロック用に十分なスペースを予約し、デフォルトのボリューム・ヘッダを 8192 ブロックに増やします。

```
xvm:cluster> label -volhdrblks 8192 -xvmlabelblks 4096 -name fred dks0d3
```

show コマンドを使った物理ボリュームの表示

XVM オブジェクトに関する情報を表示するには、show コマンドを使用します。以下に、-verbose オプションを有効にして物理ボリュームに対して実行した show コマンドの結果の例を示します。

```
xvm:cluster> show -v phys/disk2
XVM physvol phys/disk2
=========================
size: 17779016 blocks  sectorsize: 512 bytes  state: online,cluster
uuid: a9764967-439c-1023-8c3b-0800690565c0
system physvol:  no
physical drive:  /dev/rdsk/dks5d6vol on host hugh3
Disk has the following XVM label:
  Clusterid:  0
  Host Name:  hugh_cluster
  Disk Name:  disk2
  Magic:  0x786c6162 (xlab)     Version 2
  Uuid: a9764967-439c-1023-8c3b-0800690565c0
  last update:  Sun Dec 12 16:27:16 1999
  state: 0x21<online,cluster> flags: 0x0<idle>
  secbytes:  512
  label area:  1024 blocks starting at disk block 3072 (10 used)
  user area:   17779016 blocks starting at disk block 4096

Physvol Usage:
Start        Length       Name
-----------------------------------------------------
0            17779016     slice/disk2s0

Local stats for phys/disk2 since being enabled or reset:
-----------------------------------------------------------------------
stats collection is not enabled for this physvol
```

また、show コマンドは、ラベルの付いていないディスクに関する情報を表示することもできます。以下に、ラベルの付いていないディスクに対して実行した show コマンドの結果の例を示します。

```
xvm:cluster> show -v unlabeled/dks0d1
Unlabeled disk unlabeled/dks0d1vol
================================
volume alias:      /dev/rdsk/dks0d1vol
volume full path:
/hw/module/2/slot/io1/baseio/pci/0/scsi_ctlr/0/target/1/lun/0/disk/volu
me/char
Disk does not have an XVM label
```

show コマンドは、XVM ボリューム・マネージャで外部ディスクとして認識されているディスクに関する情報を表示できます。これは、steal コマンドを使用して現在のオーナーから XVM の physvol の制御権を得る必要がある場合に便利です。この場合、外部ディスクとして表示されているので physvol として読取ることができないディスクのオーナーを判別しなければならないことがあります。オーナーは、show コマンドの出力に physvol の「ホスト名 (Host Name)」として表示されます。

以下の例では、現在のノード hugh2 の外部ディスク dks5d46 に対して show コマンドを実行します。

```
hugh2 1# xvm
xvm:cluster> show dks5d46
foreign/dks5d46vol                * private
xvm:cluster> show -v dks5d46
Foreign disk foreign/dks5d46vol
================================
volume alias:      /dev/rdsk/dks5d46vol
volume full path:
/hw/module/2/slot/io7/fibre_channel/pci/0/scsi_ctlr/0/target/46/lun/0/d
isk/volume/char
Disk has the following XVM label:
  Clusterid:  0
  Host Name:  hugh
  Disk Name:  disk5
  Magic:  0x786c6162 (xlab)     Version 2
  Uuid: 530138fd-0096-1023-8a7a-0800690592c9
  last update:  Thu Sep 16 10:25:58 1999
  state: 0x11<online,private> flags: 0x0<idle>
  secbytes:  512
  label area:  1024 blocks starting at disk block 3072 (10 used)
  user area:   17779016 blocks starting at disk block 4096
```

この例では、外部ディスクの「ホスト名 (Host Name)」は、ディスクの現在のオーナーである hugh です。

外部ディスクについての詳細は、26 ページの「XVM ドメイン」を参照してください。steal コマンドについての詳細は、65 ページの「give と steal コマンドを使った物理ボリュームのドメインの変更」を参照してください。steal コマンドは、give コマンドを使用して所有権を変更できない場合のみ使用してください。

show コマンドは、ほかのボリューム要素に関する情報を表示する場合にも使用できます。show コマンドのその他の例については、77 ページの「ボリューム要素の表示: show コマンドの使用」を参照してください。

change コマンドを使った物理ボリュームの変更

XVM 物理ボリュームの名前を変更したり、統計集合の状態（オン、オフ、またはリセット）を変更するには、change コマンドを使用できます。change コマンドの例については、72 ページの「change コマンド」を参照してください。

さらに、change コマンドを使用して、XVM システム・ディスクを XVM option ディスクに変更することもできます。swap ボリュームとしてマークされているボリュームは削除できないので、XVM システム・ディスクの論理ボリュームを削除しなければならない場合、この作業が必要になることがあります。システム・ディスクの変更についての詳細は、91 ページの「XVM システム・ディスクの削除」を参照してください。

probe コマンドを使った物理ボリュームの検査

probe コマンドは、システムがディスクを XVM ディスクとして認識できるよう、XVM ラベルが付けられているディスクを検査します。ディスクは、システムの起動時に自動的に検査されますが、実行中のシステムに XVM ディスクを追加した場合は、probe コマンドを手動で実行しなければなりません。ラベルが付けられていないディスクに対して probe コマンドを実行した場合は、エラーが返されます。

最初に、検査されるディスクがハードウェア・インベントリ内で使用可能になっている必要があります。システムにディスクを認識させるには、scsiadminswap(1M) コマンドを使用します。

以下の例では、コントローラ 0 上のドライブ 4 を検査します。

```
xvm:cluster> probe dks0d4
```

以下の例では、fred という XVM 物理ボリュームを再検査します。

```
xvm:cluster> probe fred
```

以下の例では、すべての SCSI ドライブを検査します。

```
xvm:cluster> probe dks*
```

dump コマンドを使った XVM 物理ボリュームの再生成

XVM 物理ボリュームを再生成するコマンドをファイルにダンプするには、dump コマンドを使用します。dump コマンドの例については、78 ページの「ボリューム要素の再構築: dump コマンドの使用」を参照してください。

give と steal コマンドを使った物理ボリュームのドメインの変更

XVM の physvol のドメインを変更して、その physvol の所有権を別のノードまたはクラスタに与えるには、give コマンドを使用します。

開いているサブボリュームの一部であるスライスがある physvol では、give コマンドを使用できません。このため、give コマンドは、ミラーの再生がアクティブな間は失敗します。通常は、give コマンドを physvol で実行する前に、XVM の physvol が含まれる XVM 論理ボリュームでファイルシステムをアンマウントし、ミラーの再生が完了するまで待つ必要があります。

ディスクを引渡した場合、新しくオーナーとなったノードまたはクラスタで設定を表示するには、新しいオーナーがディスクを読取る必要があります。これは、以下の 2 つのうちいずれかの方法で行われます。

- 再起動時に自動的に読取る
- 新しいオーナーが probe コマンドを使用して新しいディスクを読取る

引渡す物理ボリュームは、physvol 名またはディスク自体の名前で指定できます。次のコマンドは、ディスク dks0d4 の所有権を放棄して hosta というオーナーに渡します。

```
xvm:local> give hosta dks0d4
```

次のコマンドは、fred という physvol の所有権を放棄して mycluster というクラスタに渡します。

```
xvm:local> give -cluster mycluster fred
```

状況によっては、現在物理ボリュームを所有しているノードまたはクラスタで give コマンドを実行できない場合があります。このような場合は、steal コマンドを使用して、XVM 物理ボリュームのドメインを変更できます。steal の対象に指定できるのは、現在のノードまたはクラスタの外部ディスクだけです。

注意：steal コマンドは、XVM の physvol のオーナーを無条件に現在のノードまたはクラスタにリセットします。所有権の変更は、以前のオーナーに通知されません。このため、設定が破壊される可能性があります。steal コマンドは、give コマンドで所有権を変更できない場合のみ使用してください。

---

steal コマンドを使用して XVM 物理ボリュームのドメインを変更する必要がある場合に、物理ボリュームの現在のオーナーの名前がわからないことがあります。現在のオーナーを判別するには、62 ページの「show コマンドを使った物理ボリュームの表示」に説明されているとおり、外部ディスクに対して show コマンドを使用できます。

give コマンドの場合と同じように、開いているサブボリュームの一部であるスライスがある physvol では steal コマンドを使用できません。通常は、steal コマンドを physvol で実行する前に、XVM の physvol が含まれる XVM 論理ボリュームでファイルシステムをアンマウントし、ミラーの再生が完了するまで待つ必要があります。

以下の例では、ディスク foreign/dks0d4vol の所有権をリセットします。この例で steal コマンドが成功するには、このディスクが hosta によって所有されていなければなりません。ディスクは、現在のクラスタが所有するクラスタ・ディスクになります。

```
xvm:cluster> steal hosta foreign/dks0d4vol
```

以下の例でも、ディスク foreign/dks0d4vol の所有権をリセットします。この例では、ディスクはローカル・ドメインになります。

```
xvm:cluster> set domain local
xvm:local> steal hosta foreign/dks0d4vol
```

デフォルトでは、steal コマンドは、物理ボリュームの所有権をホストから引取ります。クラスタによって所有されている物理ボリュームの所有権を引取るには、-cluster オプションを使用しなければなりません。以下の例では、ディスク foreign/dks0d4vol の所有権をクラスタ jiminy からローカル・ホストにリセットします。steal が成功するには、ディスクがクラスタ jiminy によって所有されていなければなりません。

```
xvm:local> steal -cluster jiminy foreign/dks0d4vol
```

set コマンドについての詳細は、60 ページの「set コマンドを使った現在のドメインの変更」を参照してください。ローカル・ドメイン、クラスタ・ドメイン、および外部ディスクについての詳細は、26 ページの「XVM ドメイン」を参照してください。

unlabel コマンドを使った XVM ボリューム・マネージャからのディスクの削除

ディスクが XVM ディスクではなくなるよう、そのディスクから XVM ラベルを削除するには、unlabel コマンドを使用します。これにより、元のパーティション設定スキームがディスクに復元されます。クラスタ化された環境では、作業しているシステムに接続されていないディスクのラベルは削除できません。

以下の例では、phys1 という XVM 物理ボリュームから XVM ラベルを削除します。

```
xvm:cluster> unlabel phys1
```

以下の例では、存在する可能性のあるスライスをすべて最初に削除した後、phys1 から強制的にラベルを削除します。

```
xvm:cluster> unlabel -force phys1
```

論理ボリューム・コマンド

以下の節では、ボリューム要素の作成、変更、表示、再構築、および削除に使用する xvm コマンドについて説明します。

ボリューム要素の作成

以下の論理ボリューム要素を作成するには、それぞれ異なる xvm コマンドを使用します。

- スライス
- 連結
- ミラー
- ストライプ
- サブボリューム
- ボリューム

以下の節では、これらのコマンドの概要について説明します。

slice コマンド

slice コマンドは、XVM 物理ボリュームの指定したブロック範囲からスライスを作成します。

slice コマンドを使用して、システム・ディスクのボリューム構成に使用されるスライスを作成する場合についての詳細は、85 ページの「slice コマンドを使ったシステム・ディスクの設定」を参照してください。

以下の例では、XVM 物理ボリューム phys1 の使用可能な全容量を使用する 1 つのスライスを作成します。

```
xvm:cluster> slice -all phys1
```

以下の例では、XVM 物理ボリューム phys1 を使用する 4 つの等サイズのスライスを作成します。

```
xvm:cluster> slice -equal 4 phys1
```

以下の例では、長さが 100,000 ブロックでブロック 5,000 から開始するスライスを作成します。

```
xvm:cluster> slice -start 5000 -length 100000 phys1
```

以下の例では、ブロック 5,000 から開始する 100,000 ブロックのチャンクを 4 つの等サイズのスライスに分割します。

```
xvm:cluster> slice -start 5000 -length 100000 -equal 4 phys1
```

concat コマンド

concat コマンドは、すべての子ボリューム要素を 1 つのアドレス空間に連結するボリューム要素を作成します。連結を作成するときは、連結を接続する生成されたボリュームに名前を付けるか、またはシステムに一時ボリューム名を生成させるかを指定しなければなりません。

以下の例では、スライス freds0 および wilmas0 をより大きいアドレス空間に連結します。作成される連結ボリューム要素にはシステムによって生成された一時名が付けられ、この要素は、システムによって生成された一時名を持つボリュームに含まれます。

```
xvm:cluster> concat -tempname slice/freds0 slice/wilmas0
```

以下の例でも、スライス freds0 および wilmas0 をより大きいアドレス空間に連結します。ここでは、作成される連結には myconcat、連結が属するボリュームには concatvol という名前を明示的に付けています。

```
xvm:cluster> concat -vename myconcat -volname concatvol slice/freds0 \
slice/wilmas0
```

mirror コマンド

mirror コマンドは、すべての子ボリューム要素をミラー化するボリューム要素を作成します。ミラーを作成するときは、ミラーを接続する生成されたボリュームに名前を付けるか、またはシステムに一時ボリューム名を生成させるかを指定しなければなりません。

複数の断片があるミラーを作成すると、ミラーが再生中であることを示すメッセージが SYSLOG に書込まれます。これは、システムによってデータのミラー化プロセスが開始されていることを示します。このプロセスが完了すると、別のメッセージが SYSLOG に書込まれます。大容量のミラー・コンポーネントでは、このプロセスに時間がかかる場合があります。ミラーの再生は、いったん開始したら、ミラーの断片の 1 つを除いたすべてを分離する方法以外では中止できません。

何らかの理由で再生が失敗した場合は、SYSLOG のほかにシステム・コンソールにもメッセージが書込まれます。ミラーの再生についての詳細は、第 7 章、「XVM ボリューム・マネージャの運用」を参照してください。

ミラーを作成するときに、ミラーの読取りポリシーとプライマリ・レッグを定義できます。これらの機能は、38 ページの「ミラーの作成」で説明されています。

ミラーを作成する場合、作成時にミラーの再同期が必要がないことを指定できます。また、ミラーをまったく再同期しなくてもよいように指定することもできます。これは、スクラッチ・ファイルシステムをミラー化するときに便利です。これらの機能は、38 ページの「ミラーの作成」で説明されています。

ミラーのコンポーネントのサイズはすべて同じである必要はありませんが、コンポーネントのサイズが異なる場合は、大きい方のコンポーネントに未使用の容量が残ります。

以下の例では、スライス freds0 および wilmas0 がメンバーであるミラーを作成します。ミラーが関連付けられるボリュームには、mirvol という名前を付けています。

```
xvm:cluster> mirror -volname mirvol slice/freds0 slice/wilmas0
```

以下の例では、メンバーが slice/freds0 および slice/wilmas0 で、ボリューム名が newmirvol のミラーを作成します。この例では、ミラーの作成時に再生は開始されません。

```
xvm:cluster> mirror -volname newmirvol -clean slice/freds0 slice/wilmas0
```

以下の例では、シーケンシャル・ミラー読取りポリシーを使用する空のミラーを作成します。このミラーを使用可能にするには、attach コマンドを使用して、ボリューム要素のメンバーを明示的に接続する必要があります。このコマンドは、システムによって生成された名前を持つボリュームに含まれるミラーを作成します。ミラーの名前も、システムによって生成されます。

```
xvm:cluster> mirror -tempname -rpolicy sequential
```

以下の例では、freds0 という名前のプライマリ・メンバーを持つ 2 メンバー・ミラーを作成します。読取りはすべて freds0 に送信され、書込みは両方のメンバーに送信されます。このコマンドは、システムによって生成された名前を持つボリュームに含まれるミラーを作成します。ミラーの名前も、システムによって生成されます。

```
xvm:cluster> mirror -tempname -primary slice/freds0 slice/freds0 slice/wilmas0
```

stripe コマンド

stripe コマンドは、アドレス空間全体にボリューム要素のセットをストライプするボリューム要素を作成します。ストライプを作成するときは、ストライプを接続する生成されたボリュームに名前を付けるか、またはシステムに一時ボリューム名を生成させるかを指定しなければなりません。

サイズの等しくないボリューム要素で構成されるストライプを作成しても問題はありませんが、使用されない容量が残ります。

ストライプ・ボリューム要素の実際のサイズは、ストライプ単位サイズと、ストライプを構成するボリューム要素のサイズに基づきます。最も単純なケースでは、ボリューム要素はすべて同じサイズで、ストライプ単位サイズの偶数倍となります。たとえば、ストライプ単位が 128 個の 512 バイト・ブロック（デフォルトのストライプ単位サイズ）の場合に、それぞれが 256,000 ブロックの 2 つのスライスで構成されるストライプを作成すると、各スライスの全容量が使用されます。ストライプ・サイズは、2 つのスライスを合わせた 512,000 ブロックになります。

一方、ストライプを構成する 2 つのスライスがそれぞれ 250,000 ブロックで、ストライプ単位が 128 ブロックの場合、各スライスでは 249,984 個のブロックのみがストライプに使用され、ストライプのサイズは 499,968 ブロックになります。このような状況は、ディスクを均等に分割することによってディスク上にスライスを作成した場合や、ディスク全体を 1 つのスライスとして使用し、得られたストライプ・サイズをストライプ単位サイズに合わせて調整していない場合に発生します。

スライスを構成する各ボリューム要素で使用される容量は同じなので、2 番目の例の 2 スライス・ストライプを構成する 2 つのスライスの一方が 256,000 ブロックで、もう一方が 250,000 ブロックの場合でも、ストライプ・サイズは 499,968 ブロックになります。

以下に、ストライプ・サイズを決定する一般的な数式を示します。ここで、stripe_width は、ストライプを構成するボリューム要素の数となります。

stripe size = (smallest_stripe_member / stripe_unit) * stripe_unit * stripe_width

この数式では整数演算を使用することに注意してください。

既存のストライプのストライプ単位は、 show -v *stripe* コマンド（*stripe* は既存のストライプの名前）を使用して表示できます。

2 つのホスト・バス・アダプタにまたがるストライプの設定についての詳細は、24 ページの「XVM 論理ボリュームとフェイルオーバー」を参照してください。

以下の例では、スライス freds0 および wilmas0 をストライプします。ストライプが関連付けられているボリュームには、stripedvol という名前を付けます。

```
xvm:cluster> stripe -volname stripedvol slice/freds0 slice/wilmas0
```

以下の例では、512 ブロックのストライプ単位サイズを使用して、ミラー mirror0 および mirror1 をストライプします。

```
xvm:cluster> stripe -tempname -unit 512 mirror[01]
```

以下の例では、4 つのスライスを格納できる空のストライプを作成します。ストライプがオンラインになる前に、4 つのボリューム要素をストライプに接続する必要があります。

```
xvm:cluster> stripe -tempname -pieces 4
```

subvolume コマンド

subvolume コマンドは、サブボリュームを作成します。また、オプションで、指定したボリューム要素をそのサブボリュームに接続します。サブボリュームに接続するボリューム要素に、ボリュームや別のサブボリュームを指定することはできません。

サブボリュームを作成するときは、サブボリュームを接続する生成されたボリュームに名前を付けるか、またはシステムに一時ボリューム名を生成させるかを指定しなければなりません。

data、log、または rt（リアルタイム）のシステム定義タイプのサブボリュームを作成したり、ユーザ定義タイプのサブボリュームを作成できます。システム定義タイプのサブボリュームに対してサブボリューム名を指定することはできません。

以下の例では、log サブボリュームを作成して concat0 を接続します。このサブボリュームに関連付けられているボリュームには、myvol という名前を付けます。

```
xvm:cluster> subvolume -volname myvol -type log concat0
```

以下の例では、log サブボリュームを作成して concat0 を接続し、このサブボリュームに対応するブロック特殊ファイルとキャラクタ特殊ファイルの uid と mode を設定します。

```
xvm:cluster> subvolume -tempname -type data concat/concat0
```

以下の例では、ユーザ定義タイプが 100 のサブボリュームを作成します。

```
xvm:cluster> subvolume -tempname -type 100 concat0
```

volume コマンド

volume コマンドは、XVM ボリュームを作成します。また、オプションで、指定したサブボリュームをそのボリュームに接続します。

以下の例では、fred という空のボリュームを作成します。

```
xvm:cluster> volume -volname fred
```

以下の例では、data、log、および real-time サブボリュームを 1 つのボリュームの下層にグループ化します。作成されたボリュームには、システムによって生成された一時名が付けられます。

```
xvm:cluster> volume -tempname vol0/data vol1/log vol2/rt
```

ボリューム要素の変更

XVM ボリューム・マネージャには、ボリューム要素を作成した後に変更するために以下のコマンドが用意されています。

- change
- attach
- detach
- remake

以下の節では、これらのコマンドについて説明します。

change コマンド

change コマンドは、すでに定義されている XVM の物理ボリュームまたはボリューム要素の属性を変更します。change コマンドでは、オブジェクトに応じて XVM オブジェクトのさまざまな属性を変更できます。change コマンドを使用すると、オブジェクトの統計集合を有効にしたり、カーネルが無効になっているボリューム要素をオンラインに戻したり、ボリューム要素を手動で無効にして再度有効にできます。

既存のオブジェクトの名前を変更する場合にも、change コマンドを使用できます。このコマンドを使用してオブジェクトに付けた名前は、再起動しても変わりません。スライスの名前は変更できません。

change コマンドを使用して変更できる属性の完全なリストについては、このコマンドのヘルプ画面を参照してください。

以下の例では、XVM 物理ボリューム pvol0 および vol/fred の data サブボリュームの統計を有効にします。

```
xvm:cluster> change stat on phys/pvol0 vol/fred/data
```

以下の例では、統計が有効になっているすべてのオブジェクトの統計をリセットします。

```
xvm:cluster> change stat reset *
```

attach コマンド

attach コマンドは、既存のボリューム要素を別の既存のボリューム要素に接続します。接続時に XVM ボリューム・マネージャによって強制的に適用される制約についての詳細は、35 ページの「ボリューム要素の接続」を参照してください。

ボリューム要素を接続する場所を指定できます。ボリューム要素を接続する場所を明示的に指定しなかった場合、ソース・ボリューム要素はターゲット・ボリューム要素の最初（左端）の穴に接続されます。穴がない場合は、最後（右）に追加されます。

attach コマンドを使用して、複数のソース・ボリューム要素を単一のターゲット・ボリューム要素に接続できます。複数のソース・ボリューム要素を接続する場合、接続に指定した位置は最初のボリューム要素だけに対して適用されます。残りのボリューム要素は右側に配置され、穴が埋められるか追加されます。

複数のソース・ボリューム要素を単一のターゲット・ボリューム要素に接続すると、一度に 1 つずつ順番に接続されます。リストに含まれる特定の接続が失敗した場合、XVM はボリューム要素を以前の親に復元します。ボリューム要素を復元できない場合は、警告メッセージが表示され、手動操作が必要になります。

以下の例では、concat0 の最初に使用できる位置にスライス freds0 を接続します。

```
xvm:cluster> attach slice/freds0 concat0
```

以下の例では、vol0 のすべてのサブボリュームを vol1 に接続します。

```
xvm:cluster> attach vol0/* vol1
```

以下の例では、slice/freds0 を concat0 に接続し、concat0 と slice/freds0 が、開いているサブボリュームの一部でなくても、その一部である場合と同様にチェックを実行します。

```
xvm:cluster> attach -safe slice/freds0 concat0
```

detach コマンド

detach コマンドは、ボリューム要素をその親から分離します。ボリューム要素を作成するとボリュームが作成されるのと同じように、ボリューム要素を分離すると新しいボリューム（多くの場合は data サブボリュームも）が作成されます。生成されたボリュームに明示的に名前を付けたり、ボリュームが一時名で自動的に生成されるよう指定することもできます。分離するボリューム要素に対して、タイプ data のサブボリュームが自動的に生成されます（分離するボリューム要素自体が別のタイプのサブボリュームである場合を除く）。

開いているミラーの最後の有効な断片をそのミラーから分離することはできません。この操作を行うと、ミラーがオフラインになってしまうためです。

分離するボリューム要素が開いているサブボリュームの一部である場合、分離によってサブボリュームの状態がオフラインになることはありません。detach コマンドには、この制約をオーバーライドする -force オプションと、サブボリュームが開いていない場合でも強制的にこの制約を適用する -safe オプションが用意されています。

以下の例では、ボリューム要素 concat0 をその親から分離します。concat0 が関連付けられているボリュームは、分離された後、fred という名前が付けられます。

```
xvm:cluster> detach -volname fred concat0
```

以下の例では、concat0 が開いているサブボリュームの一部で、分離した結果サブボリュームがオフラインになる場合でも、分離を行います。

```
xvm:cluster> detach -force -tempname concat0
```

以下の例では、concat0 を分離しますが、対応するサブボリュームが開いていなくても、分離によってサブボリュームがオフラインにならないようにします。

```
xvm:cluster> detach -safe -tempname concat0
```

remake コマンド

remake コマンドは、ボリューム要素の穴を取除いたり、ボリューム要素の下位の断片を再度並べ替えたりして、XVM 論理ボリュームのボリューム要素を再構成します。attach コマンドと detach コマンドを連続して実行する代わりに、remake コマンドを一度だけ実行すればよいので便利です。

ボリューム要素の断片を再度並べ替える場合は、remake コマンドのオプションを使用して、再度並べ替えする以下の方法の 1 つを指定できます。

swap　特定のボリューム要素の下位にある 2 つのボリューム要素の位置を入替えます。

reorder　1 つのボリューム要素の下位にある子の順序を変更します。

以下の例では、ve concat0 の穴を取除きます。

```
xvm:cluster> remake concat0
```

以下の例では、concat0 を再度並べ替えて、断片 0 と 1 を入替えます。

```
xvm:cluster> remake concat0 swap concat0/0 concat0/1
```

以下の例では、concat0 を再度並べ替え、その 3 つの断片の順序を逆にします。

```
xvm:cluster> remake concat0 reorder concat0/2 concat0/1 concat0/0
```

実行中のシステムでのボリューム要素の変更

以下の節で説明するように、XVM ボリューム・マネージャでは、insert コマンドおよび collapse コマンドを使用して、実行中のシステムのボリューム要素を変更できます。

insert コマンド

insert コマンドは、別のボリューム要素上にミラーまたは連結ボリューム要素を挿入します。ボリュームまたはサブボリューム上にボリューム要素を挿入することはできません。

insert コマンドを使用すると、連結を挿入することによって実行中のシステムのボリューム要素を拡張したり、ミラーを挿入することによって実行中のシステムにミラーを追加することができます。拡張またはミラー化するボリューム要素は、開いているサブボリュームの一部でもかまいません。また、アクティブな I/O が行われている状態でも拡張やミラー化を実行できます。

たとえば、freds0 という単一のスライスが含まれる tinyvol という単一の論理ボリュームから開始する場合、論理ボリュームのトポロジーは以下のようになります。

```
xvm:cluster> show -top tinyvol
vol/tinyvol                          0 online
    subvol/tinyvol/data            177792 online,open
        slice/freds0                   177792 online,open
```

ボリュームのスライス上に連結を挿入できます。

```
xvm:cluster> insert concat slice/freds0
</dev/cxvm/tinyvol> concat/concat0
```

論理ボリュームのトポロジーは、以下のようになります。

```
xvm:cluster> show -top tinyvol
vol/tinyvol                          0 online
    subvol/tinyvol/data            177792 online,open
        concat/concat0                 177792 online,tempname,open
            slice/freds0                   177792 online,open
```

これで、別のスライスを連結に接続することによってボリュームを拡張できます。

以下の例では、ボリューム要素 concat0 上に 1 メンバー・ミラーを挿入します。これにより、ほかのメンバーを接続して、concat0 をオフラインにすることなくミラー化できます。

```
xvm:cluster> insert mirror concat0
```

以下の例では、スライス freds0 上に 1 メンバー連結を挿入します。これにより、ほかのメンバーを接続して、対応するサブボリュームをオフラインにすることなく拡張できます。

```
xvm:cluster> insert concat slice/freds0
```

collapse コマンド

collapse コマンドは、ボリューム要素を解除して、そのボリューム要素の子をボリューム要素の親にリンクすることによって、論理ボリューム・ツリーから層を削除します。

collapse コマンドは、開いているサブボリュームのミラーまたは連結を解除するために使用できます。一般的に、このコマンドは、前の挿入操作を元に戻すときに使用します。

たとえば、以下の連続するコマンドは、concat1 という既存の連結上にミラーを挿入した後、作成された論理ボリュームのトポロジーを表示します。

```
xvm:cluster> insert mirror concat1
</dev/cxvm/vol9>  mirror/mirror2
xvm:cluster> show -top vol9
vol/vol9                             0 online,tempname
    subvol/vol9/data               5926338 online
        mirror/mirror2                 5926338 online,tempname
            concat/concat1                 5926338 online,tempname
                slice/pebbless0                2963169 online
                slice/bettys0                  2963169 online
```

以下の連続するコマンドでは、ミラーを解除して前の例での挿入操作を元に戻し、作成された論理ボリュームのトポロジーを表示します。

```
xvm:cluster> collapse mirror/mirror2
xvm:cluster> show -top vol9
vol/vol9                            0 online,tempname
    subvol/vol9/data              5926338 online
        concat/concat1               5926338 online,tempname
            slice/pebbless0              2963169 online
            slice/bettys0                2963169 online
```

ボリューム要素の表示 : show コマンドの使用

show コマンドは、ボリューム要素に関する情報だけでなく、物理ボリュームやラベルの付いていないディスクに関する情報も表示します。このコマンドには、**-verbose** オプションが含まれています。

以下の例では、concat0 というオブジェクトの名前、サイズ、および状態に関する情報を表示します。

```
xvm:cluster> show concat0
```

以下の例では、すべての XVM スライスを表示します。

```
xvm:cluster> show slice/*
```

以下の例では、すべての XVM ボリュームの名前を表示します。

```
xvm:cluster> show -name vol/*
```

以下の例では、dks コントローラ 0 上のラベルの付いていないすべてのディスクの名前を表示します。

```
xvm:cluster> show -name unlabeled/dks0*
```

以下の例では、fred という XVM 物理ボリュームの統計を表示します。

```
xvm:cluster> show -stat phys/fred
```

ボリューム要素の再構築 : dump コマンドの使用

dump コマンドは、XVM 設定コマンドをファイルにダンプします。

dump コマンドを使用すると、個々のボリューム要素の設定情報をダンプしたり、指定したボリューム要素の下位にあるすべてのボリューム要素の設定情報をダンプすることができます。

また、dump コマンドを使用して、物理ボリュームの設定コマンドをダンプすることもできます。物理ボリュームは、ボリューム要素ツリーとは別に明示的にダンプしなければなりません。

以下の例では、fred というボリュームの 1 行の作成コマンドをダンプします。

```
xvm:cluster> dump vol/fred
```

以下の例では、各ボリュームのすべてのボリューム要素の情報をダンプします。

```
xvm:cluster> dump -topology vol/*
```

以下の例では、foo というファイルにすべての XVM 物理ボリュームとボリューム・ツリーの内容をダンプします。

```
xvm:cluster> dump -topology -file foo phys/* vol/*
```

ボリューム要素の削除 : delete コマンドの使用

delete コマンドは、ボリューム要素を削除します。削除されたボリューム要素の親はそのまま残り、スロットが開かれた状態となります。

削除するボリューム要素が開いているサブボリュームの一部である場合、削除によってサブボリュームの状態がオフラインになることはありません。delete コマンドには、この制約をオーバーライドする -force オプションが用意されています。

ボリューム要素に子が接続されている場合、そのボリューム要素は削除できません。ただし、delete コマンドには、この制約をオーバーライドする 2 つのオプションが用意されています。1 つは -all オプションで、ボリューム要素とその下にあるすべてのボリューム要素を削除します。もう 1 つは、-nonslice オプションで、ボリューム要素と、その下にあるスライス以外のすべてのボリューム要素を削除します。スライスは分離して維持します。-all オプションと -nonslice オプションは同時に使用できません。

以下の例では、vol0 と、その下にあるすべてのボリューム要素を削除します。

```
xvm:cluster> delete -all vol0
```

以下の例では、vol0 と、スライス以外の vol0 の子孫を削除します。スライスは分離され、ボリュームと data サブボリュームが自動的に生成されます。

```
xvm:cluster> delete -nonslice vol0
```

XVM システム・ディスクの swap パーティションは削除できません。これは、swap パーティションを誤って削除した場合にシステム・パニックが発生するのを防止するためです。システム・ディスクの論理ボリュームを削除して、システムからラベルを削除する必要がある場合は、91 ページの「XVM システム・ディスクの削除」の説明に従って、まず XVM の change コマンドを使用してそのディスクをシステム・ディスクから option ディスクに変更しなければなりません。

XVM システム・ディスク

ディスクにラベルを付けて XVM ボリューム・マネージャに割当てる場合、システム・ディスクとして使用できるようにディスクにラベルを付けることができます。これにより、システム・ディスクのパーティションが含まれる XVM 論理ボリュームを作成できます。

サポートされている XVM システム・ディスクの機能は、以下のとおりです。

- root パーティションをミラー化できる
- システム・ディスクに複数の root パーティションを設定できる
- usr および swap パーティションを、ミラー、連結、およびストライプを含むどのような XVM 論理ボリューム設定でも使用できる
- システム・ディスクには、root、usr、または swap パーティションの一部ではないスライスを含めることができる

XVM ボリューム・マネージャには、既存の XVM システム・ディスクのシステム・ディスク・パーティションをミラー化できるようディスクにラベルを付けるオプションが用意されています。

XVM ボリューム・マネージャでは、XVM 以外の既存のシステム・ディスクを XVM システム・ディスクに切替えた後、XVM 論理ボリュームの一部としてそれらのディスクのパーティションを使用できます。既存のシステム・ディスクを XVM システム・ディスクに切替えた後は、XVM の unlabel コマンドを使用してディスクからラベルを削除することによって、そのディスクを元の状態に戻すことができます。

以下の節では、システム・ディスクを管理するために使用するコマンドについて説明します。

- 「XVM システム・ディスクの作成」(80 ページ)
- 「slice コマンドを使ったシステム・ディスクの設定」(85 ページ)
- 「XVM システム・ディスクのミラー化」(86 ページ)
- 「XVM システム・ディスクへのアップグレード」(88 ページ)
- 「XVM システム・ディスクの削除」(91 ページ)

XVM システム・ディスクの作成およびミラー化の手順の例は、112 ページの「XVM システム・ディスクの作成とミラー化」を参照してください。

XVM システム・ディスクの作成

XVM の `label` コマンドを使用してディスクに XVM ディスクとしてラベルを付けるときは、`-type` オプションを使用して、作成する XVM ディスクのタイプを指定できます。XVM ディスクには、option、root、および `usrroot` の 3 つのタイプがあります。

`option` タイプが `option` の XVM の physvol は、デフォルトの XVM ディスク・タイプです。XVM の `label` コマンドを使用してディスクに option ディスクとしてラベルを付ける方法については、60 ページの「label コマンドを使った XVM ボリューム・マネージャへのディスクの割当て」を参照してください。

6 ページの図 1-1 に示すように、XVM の option ディスクには、ボリュームおよびボリューム・ヘッダ用のパーティションが含まれます。

`root` 7 ページの図 1-2 に示すように、タイプが `root` の XVM の physvol には、ボリューム・ヘッダおよびボリューム全体のパーティションのほかに、少なくとも 1 つの root パーティションと 1 つの swap パーティションが含まれます。

ディスクに XVM の root ディスクとしてラベルを付けると、root および swap パーティションにマップされる `root` および `swap` タイプのスライスと同様に、各 root パーティション用の論理ボリュームと swap パーティション用の論理ボリュームが作成されます。スライス・タイプについての詳細は、85 ページの「slice コマンドを使ったシステム・ディスクの設定」を参照してください。作成した root ボリュームには、*physvol*`_root`*n* という名前が付けられます。ここで、*physvol* は XVM の root 物理ボリュームの名前、*n* はボリュームを構成する root スライスのマップ先のパーティションの数になります。swap パーティションは常にパーティション 1 になるため、swap ボリュームには *physvol*`_swap1` という名前が付けられます。

XVM の root ディスクとしてラベルを付けるディスクにすでに root パーティション 0 が存在する場合は、既存のパーティション・テーブルは変更されずに残り、root およびスライス・パーティションにマップされる XVM の root および swap スライスの数とサイズを決定するために使用されます。この場合、`-rootblks` オプションおよび `-swapblks` オプションを指定しても効果はありません。この機能により、XVM 以外の既存のシステム・ディスクに XVM システム・ディスクとしてラベルを付けて、root および swap パーティションの現在のパーティション設定スキームを維持できます。既存のシステム・ディスクでは、パーティション 1、6、8、9、および 10 を除くすべてのパーティションに対して XVM の root スライスが作成されます。パーティション 1 が存在する場合、XVM の swap スライスはそのパーティションにマップされます。XVM の root ディスクを作成するときにそのパーティションが存在しない場合、`label` コマンドは中止されます。パーティション 6 が存在していると、XVM の `usr` スライスはそのパーティションにマップされます。システム・ディスクにラベルを付けるときに `label` コマンドの `-clrparts` オプションを指定すると、この機能をオーバーライドでき、XVM は既存の root パーティション 0 がない場合と同様にディスクを扱います。

ディスク上に既存の root パーティション 0 がない場合、デフォルトではタイプ root の XVM ディスクの約 10% が、swap パーティションにマップされる swap スライスに割当てられ、ディスクの残りの使用可能な部分は、root パーティション 0 にマップされる root スライスに割当てられます。これらの割合は、ディスクの容量が非常に大きい（または小さい）場合や、ディスクに複数の root スライスがある場合は変わることがあります。これらのデフォルトのサイズをオーバーライドするには、`label` コマンドの `-swapblks` オプションおよび `-rootblocks` オプションを使用できます。root および swap パーティションに使用されない root タイプの XVM の physvol 上の使用可能な容量は、ほかの XVM スライスに使用できます。

XVM の root ディスクに対して複数の root スライスを指定するには、`label` コマンドの `-rootslices` *slices* オプションを使用できます。ディスクにパーティション 0 が存在する場合、このパラメータは、`-clrparts` オプションを指定しないかぎり無視されます。追加の root スライスは、パーティション 2～5、7、および 11～15 にマップされます。

`label` コマンドの `-noparts` オプションを使用すると、スライスを作成せずに XVM システム・ディスクを作成できます。その後、85 ページの「slice コマンドを使ったシステム・ディスクの設定」の説明に従って、slice コマンドの `-type` および `-partition` オプションを使用し、root または swap スライスを作成してパーティションにマップできます。

usrroot

8 ページの図1-3 に示すように、タイプが usrroot の XVM の physvol には、少なくとも 1 つの root パーティション、1 つの swap パーティション、および 1 つの usr パーティションが含まれます。

ディスクに XVM の usrroot ディスクとしてラベルを付けることは、XVM の root ディスクとしてラベルを付けることと似ていますが、root スライスおよびボリュームに加えて usr スライスおよびボリュームも作成される点と、usr ボリュームがパーティション 6 にマップされる点が異なります。usr ボリュームには、*physvol*_usr6 という名前が付けられます。

root ディスクの場合と同様に、XVM の usrroot ディスクとしてラベルを付けるディスクにすでに root パーティション 0 が存在する場合、既存のパーティション・テーブルは変更されずに残り、label コマンドの -clrparts オプションを指定してこの機能をオーバーライドしないかぎり、このパーティション・テーブルを使用して、root、スライス、および usr パーティションにマップされる XVM の root、swap、および usr スライスの数とサイズが決定されます。usrroot ディスクとしてラベルを付ける既存のシステム・ディスクにパーティション 1 またはパーティション 6 がない場合、label コマンドは中止されます。

ディスクに既存の root パーティション 0 がない場合、デフォルトでは、タイプ usrroot の XVM ディスクの約 10% が swap パーティションに割当てられ、4% が root パーティションに割当てられます。これらの割合は、ディスクの容量が非常に大きい（または小さい）場合や、ディスクに複数の root スライスがある場合は変わることがあります。これらのデフォルトのサイズをオーバーライドするには、label コマンドの -swapblks オプションおよび -rootblocks オプションを使用できます。root、swap、および usr パーティションに使用されない usrroot タイプの XVM の physvol における使用可能な容量は、ほかの XVM スライスに使用できます。

root ディスクの場合と同様に、XVM の usrroot ディスクに対して複数の root スライスを指定するには、label コマンドの -rootslices *slices* オプションを使用できます。

label コマンドの -noparts オプションを使用すると、スライスを作成せずに XVM システム・ディスクを作成できます。その後、85 ページの「slice コマンドを使ったシステム・ディスクの設定」の説明に従って、slice コマンドの -type オプションおよび -partition オプションを使用し、root、swap、または usr スライスを作成してパーティションにマップできます。

/usr をマウントするには、fstab ファイルを変更しなければなりません。

86 ページの「XVM システム・ディスクのミラー化」で説明されているように、XVM の root ボリュームはミラーとして設定できます。swap ボリュームおよび usr ボリュームは、任意の XVM 論理ボリューム設定に設定できます。連結を使用した swap ボリューム設定の例については、127 ページの「連結を使った swap ボリュームの設定」を参照してください。

システム・ディスクは、そこから起動するノードに対してローカルとなるので、XVM システム・ディスクはローカル・ドメインから作成しなければなりません。

注意：XVM システム・ディスクの swap または usr ボリュームは、実行中のディスク上に設定しないでください。設定すると、システムを別の起動可能ディスクから再起動しなければならなくなります。swap または usr デバイスの設定を変更すると、アドレス空間が変更される可能性があります。

XVM システム・ディスクは XVM 以外の既存のシステム・ディスクから作成することをお薦めします。これにより、fx ディスク・ユーティリティを使用してパーティション・レイアウトを決定できるばかりでなく、91 ページの「XVM システム・ディスクの削除」で説明されているように、後からディスクを XVM 以外のシステム・ディスクに復元できます。

実行中の root ディスクに XVM ディスクとしてラベルを付ける手順の例については、121 ページの「動作中の root ディスクでのミラー化 XVM システム・ディスクの作成」を参照してください。

以下の例では、dks0d3 から、root ファイルシステムと usr ファイルシステムが組合わさったシステム・ディスクを作成します。この例で作成されるシステム・ディスク physvol には、fred という名前が付けられます。

```
xvm:local> label -type root -name fred dks0d3
```

このコマンドを実行すると、fred という physvol が作成され、この physvol 上に、スライスと root 用にそれぞれ 1 つずつ、2 つのスライスが定義されます。図 4-1 に、この physvol を示します。dks0d3 がもともとシステム・ディスクであった場合、root および swap スライスのサイズは、既存のパーティション・サイズによって決まります。

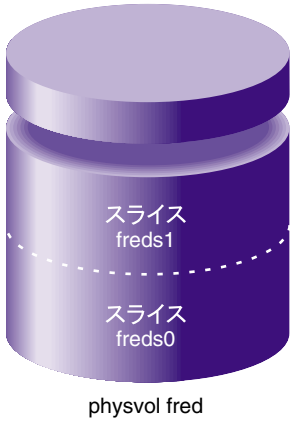

図 4-1 XVM システム・ディスク physvol fred

図 4-2 に示すように、上記のコマンドを実行すると、システム・ディスクのほかに、root ファイルシステム用の論理ボリューム `fred_root0` と swap ファイルシステム用の論理ボリューム `fred_swap1` の 2 つの論理ボリュームも作成されます。

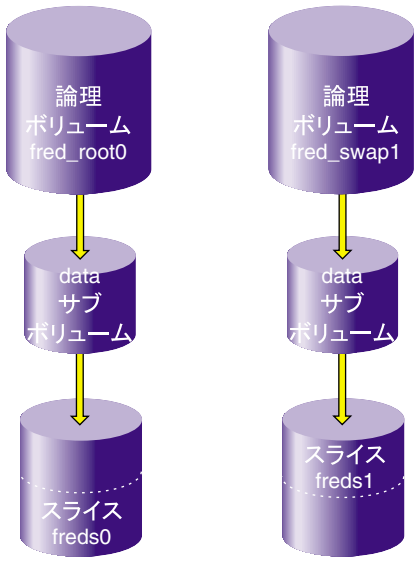

図 4-2 root および swap 用の XVM 論理ボリューム

以下の例では、root パーティションが 5 つ設定されたシステム・ディスクを作成します。この例では、-clrparts オプションを使用して、ターゲット・ディスクが、パーティション・レイアウトの異なる XVM 以外のシステム・ディスクである場合でもコマンドが機能するようにしています。

```
xvm:local> label -clrparts -rootslices 5 -type root dks1d2
```

このコマンドは、以下の show -top の出力に表示されているスライスとボリュームを作成します。

```
xvm:local> show -top vol/dks*

vol/dks1d2_root0                  0 online
    subvol/dks1d2_root0/data    3503056 online
        slice/dks1d2s0             3503056 online

vol/dks1d2_root2                  0 online
    subvol/dks1d2_root2/data    3503056 online
        slice/dks1d2s1             3503056 online

vol/dks1d2_root3                  0 online
    subvol/dks1d2_root3/data    3503056 online
        slice/dks1d2s2             3503056 online

vol/dks1d2_root4                  0 online
    subvol/dks1d2_root4/data    3503056 online
        slice/dks1d2s3             3503056 online

vol/dks1d2_root5                  0 online
    subvol/dks1d2_root5/data    3503056 online
        slice/dks1d2s4             3503056 online

vol/dks1d2_swap1                  0 online
    subvol/dks1d2_swap1/data     262144 online
        slice/dks1d2s5              262144 online
```

slice コマンドを使ったシステム・ディスクの設定

-noparts オプション（ラベルを付けたディスクにパーティション 0 が含まれていた場合は -clrparts オプションと共に）を使用して root または usrroot ディスクにラベルを付けた場合は、slice コマンドを使用して、システム・ディスク上に root、usr、および swap スライスを設定して、これらのスライスを適切なパーティションにマップできます。-clrparts オプションを使用してシステム・ディスクにラベルを付けなかった場合でも、システム・ディスクの空き容量を追加のシステム・スライスに使用できます。

XVM スライスのタイプには、root、swap、usr、または option を指定できます。デフォルトのタイプは option です。root、swap、または usr スライスを指定するには、slice コマンドの -type オプションを使用します。これにより、スライス構造にフラグが設定され、そのスライス・タイプの該当するパーティションにスライスがマップされます。デフォルトでは、swap スライスはパーティション 1 に、usr スライスはパーティション 6 に、root スライスはリスト 0、2 ～ 5、7、および 11 ～ 15 で次に使用できるパーティションにマップされます。

slice コマンドの -partition オプションを使用すると、usr、root、または swap スライスが指定のパーティションにマップされるよう指定できます。パーティション 1 は、swap スライスに対してのみ有効なパーティション番号で、パーティション 6 は、usr スライスに対してのみ有効なパーティション番号です。root スライスに対しては、パーティション番号は 0、2 ～ 5、7、または 11 ～ 15 になりますが、root スライスは、デフォルトで、そのリストで次に使用できるパーティションにマップされます。

XVM システム・ディスクのミラー化

システム・ディスクをミラー化するには、1 つまたは複数のディスクにタイプが root または usrroot の XVM ディスクとしてラベルを付けます。続いて、label コマンドの -mirror オプションを使用して、ミラー化する XVM の physvol の名前を指定します。ラベルを付けた新しいディスクは、指定したディスクの root、swap、および usr ボリュームのミラーになります。このコマンドを実行しても、新しい論理ボリュームが追加されることはなく、既存の root、swap、および usr 論理ボリュームにミラーが挿入されるだけです。システム・スライス（root、swap、または usr タイプ）ではない XVM システム・ディスク上のスライスは、このコマンドではミラー化されません。label -mirror コマンドは、すでにパーティション 0 が存在する場合でも、新しいディスク上のパーティション・テーブルを置換します。

また、ミラー化するディスクを指定せずに、追加の XVM の root または usrroot physvol を作成することもできます。続いて、既存の root、swap、または usr ボリュームにミラーを挿入し、label コマンドを実行したときに新しい XVM の physvol 上に作成された新しいシステム・スライス、または slice コマンドを使用して明示的に作成した新しいシステム・スライスを接続できます。XVM システム・ディスクをミラー化するこれらの 2 つの方法については、112 ページの「XVM システム・ディスクの作成とミラー化」で比較されています。

以下の例では、80 ページの「XVM システム・ディスクの作成」で physvol fred 上に作成した論理ボリュームをミラー化するシステム・ディスクを作成します。新しいボリュームは作成されませんが、代わりに、既存の root および swap 論理ボリュームにミラー・コンポーネントが挿入されます。

```
xvm:local> label -type root -name wilma -mirror fred dks0d4
```

図 4-3 に示すように、このコマンドを実行すると、`wilma` という physvol が作成され、その physvol 上に 2 つのスライスが定義されます。

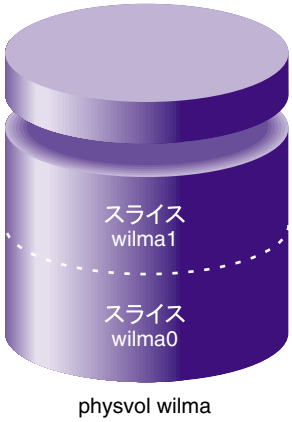

図 4-3 XVM ミラー化 root physvol

図 4-4 に示すように、physvol `wilma` 上に作成されたスライスは、既存の root および swap 論理ボリュームにミラーとして挿入されます。

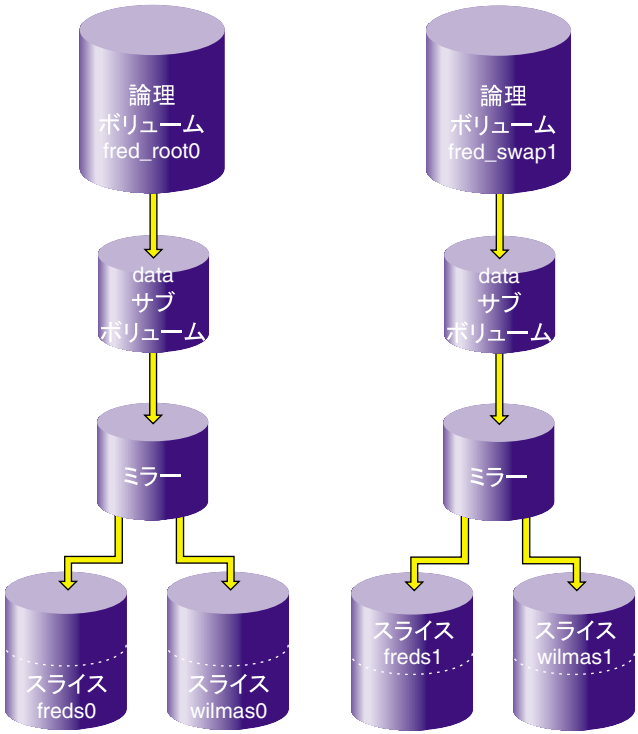

図 4-4 root と swap ミラー化論理ボリューム

XVM システム・ディスクの作成やミラー化を含むさまざまなシステム管理手順の例については、112 ページの「XVM システム・ディスクの作成とミラー化」を参照してください。

XVM システム・ディスクへのアップグレード

XVM システム・ディスクにラベルを付けたり、XVM システム・ボリュームから起動するには、IRIX リリース 6.5.12f 以降を実行している必要があります。以前のリリースを実行している場合は、これらのタスクのいずれかを実行する前に IRIX 6.5.12f をインストールしてください。また、システムがインストールされていない場合は、システムをインストールして、IRIX 6.5.12 miniroot から XVM システム・ディスクにラベルを付けることができます。

ここでは、以下の場合の手順について説明します。

- 「IRIX 6.5.11 以前からの XVM システム・ディスクへのアップグレード」（89 ページ）
- 「IRIX 6.5.12f 以降からの XVM システム・ディスクへのアップグレード」（90 ページ）
- 「システムがインストールされていない XVM システム・ディスクへのアップグレード」（90 ページ）

XVM システム・ディスクからの起動についての詳細は、91 ページの「XVM システム・ディスクからの起動」を参照してください。システム・インストールについての一般情報は、『*IRIX Admin: Software Installation and Licensing*』を参照してください。

IRIX 6.5.11 以前からの XVM システム・ディスクへのアップグレード

連結またはストライプが含まれる XVM の `usr` ボリュームを作成する場合は、それらのボリュームにシステムをインストールする前に XVM システム・ボリュームを設定する必要があります。これは、後で行うインストール中に `usr` ボリュームで同じアドレス空間が必要になるためです。ストライプまたは連結は、アドレス空間を複数のスライスに分散します。

IRIX 6.5.11 以前のシステムを実行している場合で、連結またはストライプを含む `usr` ボリュームがシステムにないときは、以下の手順を使用してディスクに XVM システム・ディスクとしてラベルを付け、その XVM システム・ディスクを再起動します。

1. 現在実行中のシステム・ディスク、または XVM システム・ディスクとして使用する予定の別のシステム・ディスクに IRIX 6.5.12f（またはそれ以降）をインストールします。
2. 新しくインストールしたディスクから再起動します。
3. XVM の `label` コマンドの `-nopartchk` オプションを使用して、ディスクに XVM システム･ディスクとしてラベルを付けます。XVM ボリュームの一部にするその他のディスクにもすべてラベルを付けます。データがボリュームに書込まれたら、システム・ボリュームをミラー化することも可能です。
4. システムを再起動します。

IRIX 6.5.11 以前のシステムを実行している場合に、連結またはストライプが含まれる `usr` ボリュームのある現在のシステム・ディスクに XVM システム・ディスクとしてラベルを付けるには、90 ページの「システムがインストールされていない XVM システム・ディスクへのアップグレード」で説明されているように、初期インストールの場合と同じ手順を使用してください。

IRIX 6.5.12f 以降からの XVM システム・ディスクへのアップグレード

現在 IRIX 6.5.12f 以降のシステムを実行している場合は、以下の手順を使用してディスクに XVM システム・ディスクとしてラベルを付けます。現在実行中のシステム・ディスクには、XVM システム・ディスクとしてラベルを付けることができます。

1. 現在実行中のシステム・ディスクにラベルを付けるときに、XVM の `label` コマンドの `-nopartchk` オプションを使用して、XVM システム・ボリュームに使用する予定のディスクにラベルを付けます。
2. 新しい XVM システム・ディスクが現在実行中のシステム・ディスクである場合は、システムを再起動します。新しい XVM システム・ディスクが新しいディスクである場合は、システムをそのディスクにインストールします。
3. IRIX の以降のリリースにアップグレードするには、システム・ディスクに新しいリリースをインストールして、ディスクから再起動します。

システムがインストールされていない XVM システム・ディスクへのアップグレード

IRIX システムが現在インストールされていない場合は、以下の手順を実行してディスクに XVM システム・ディスクとしてラベルを付け、その XVM システム・ディスクを再起動します。

1. IRIX 6.5.12f（またはそれ以降）の miniroot を起動します。
2. シェルにエスケープします。
3. XVM の `label` コマンドの `-nopartchk` オプションを使用して、XVM システム・ボリュームに使用する予定のディスクにラベルを付けます。XVM ボリュームの一部にするその他のディスクにもすべてラベルを付けます。データがボリュームに書込まれたら、システム・ボリュームをミラー化することも可能です。
4. 再起動して miniroot を起動します。
5. システムをインストールします。
6. インストールしたシステムを起動します。

XVM システム・ディスクからの起動

XVM システム・ディスクからの起動には特別な手順は必要ありませんが、新しい root または swap パーティションを指定するために、root と swap の場所を指定する以下の 4 つの環境変数を変更する必要があります。

- `root`
- `OSLoadPartition`
- `SystemPartition`
- `swap`

環境変数についての詳細は、『*IRIX Admin: System Configuration and Operation*』を参照してください。XVM システム・ディスクからの起動に関するほかの例については、112 ページの「XVM システム・ディスクの作成とミラー化」を参照してください。

XVM システム・ディスクの削除

デフォルトでは、XVM システム・ディスクの swap パーティションは削除できません。これは、swap パーティションを誤って削除した場合にシステム・パニックが発生するのを防止するためです。システム・ディスク上の論理ボリュームを削除する必要がある場合は、以下の 2 つのいずれかの方法を使用できます。

- `unlabel` コマンドの `-force` オプションを使用して、ディスクからラベルを削除します。これにより、ディスクのパーティション設定スキームが元の状態に戻り、ディスクは XVM ディスクではなくなります。
- XVM の `change` コマンドを使用して、ディスクをシステム・ディスクから option ディスクに変更します。その後、ディスク上のスライス、またはそのスライスを含むボリューム全体を削除できます。パーティションにマップされているスライスが削除された場合は、下層のパーティションも削除されます。

以下に、これらの各手順について説明します。

注意：現在 XVM システム・ディスクから実行されているシステムでは、ディスクのラベルを削除しないでください。

XVM システム・ディスクがもともとシステム・ディスクとしてパーティション設定されていた場合は、以下の手順を使用して元の状態にディスクを再設定できます。

1. 別の root ディスクからシステムを再起動します。
2. `unlabel` コマンドの `-force` オプションを使用して、ディスクからラベルを削除します。
3. これで、ラベルを削除したシステム・ディスクから実行する場合に、そのディスクから再起動できるようになります。

または、以下の手順を使用することもできます。この手順では、別の起動可能ディスクは必要ありません。

1. miniroot で実行します。
2. シェルにエスケープします。
3. root (`/root`) ファイルシステムをアンマウントします。
4. `unlabel` コマンドの `-force` オプションを使用して、ディスクからラベルを削除します。
5. システムを再起動します。

以下の一連のコマンドは、physvol `xvmroot2` を option ディスクに変更して、root および swap ボリュームを構成するスライスを削除し（ボリュームは空のボリュームとして残ります）、ディスクのラベルを削除します。

```
xvm:local> change option xvmroot2
xvm:local> delete xvmroot2/xvmroot2s0
xvm:local> delete xvmroot2/xvmroot2s1
xvm:local> unlabel xvmroot2
```

以下の一連のコマンドは、physvol `xvmroot` を option ディスクに変更して、root および swap ボリュームとそれらの子スライスを削除し、ディスクのラベルを削除します。

```
xvm:local> change option xvmroot
xvm:local> delete -all vol/xvmroot_root
xvm:local> delete -all vol/xvmroot_swap
xvm:local> unlabel xvmroot
```

XVM 管理の手順

この章は、XVM 管理の一般的な手順の例で構成されています。この章では、管理に入る前に留意しておくべきいくつかの点の概要を説明した後、以下の手順について説明します。

- 「3 方向ストライプを使った論理ボリュームの作成」(95 ページ)
- 「ディスクの一部のストライプ」(98 ページ)
- 「data サブボリュームと log サブボリュームを使った論理ボリュームの作成」(102 ページ)
- 「data サブボリューム、log サブボリューム、および real-time サブボリュームを使った論理ボリュームの作成」(104 ページ)
- 「上の階層からのボリュームの作成」(107 ページ)
- 「ストライプ・ミラーを使った XVM 論理ボリュームの作成」(109 ページ)
- 「XVM システム・ディスクの作成とミラー化」(112 ページ)
- 「連結を使った swap ボリュームの設定」(127 ページ)
- 「ミラー化を使ったオンライン再設定」(132 ページ)
- 「オンラインでの論理ボリュームの変更」(139 ページ)

作業に入る前に

XVM 論理ボリュームを設定する前に、ディスクとシステムのステータスの評価が必要な場合があります。

- 一般的に、ディスクは IRIX システム・ディスクまたは IRIX option ディスクとして SGI から出荷されています。つまり、ディスクは IRIX ボリューム・ヘッダを持ち、IRIX ディスクとしてパーティション設定されています。このようにパーティション設定されていないかぎり、ディスクに XVM ディスクとしてラベルを付けることはできません。

  IRIX ボリューム・ヘッダを持たないディスクに XVM ボリューム・マネージャを使用してラベルを付けようとすると、ディスク・ボリューム・ヘッダ・パーティションが無効であることを示すエラー・メッセージが表示されます。この場合、fx(1M) コマンドを使用してディスクを初期化する必要があります。

- xvm コマンドを実行したときに、セルでクラスタ・サービスが有効になっていないことを示すメッセージや、クラスタ・サービスを開始するまではローカル・オブジェクトしか操作できないことを示すメッセージが表示される場合があります。クラスタ・サービスの開始についての詳細は、『*CXFS Software Installation and Administration Guide*』を参照してください。

- この章の手順を開始する前に、`xvm show unlabeled/*` コマンドを実行して、XVM ボリューム・マネージャに割当てられていないシステム上のディスクの名前を確認すると役に立つ場合があります。

- マウントされているファイルシステムとして使用中のパーティションがディスクに含まれている場合、それらのディスクに XVM ディスクとしてラベルを付けることはできません。CXFS クラスタでは、共有される XVM 物理ボリュームがクラスタ内のすべてのセルに物理的に接続されていなければなりません。

- 一般的に、システムの設定とステータスの参照には show コマンドのオプションを使用するのが便利です。たとえば、`show -v stripe0` コマンドを実行すると、ストライプ単位（この場合は `stripe0` のストライプ単位）が表示されます。

- XVM 論理ボリュームを設定するには、root としてログインしている必要があります。ただし、root 特権がなくても論理名の設定情報を表示することは可能です。

メモ：XVM 論理ボリュームを設定するときは、`-v[erbose]` オプションを付けて `help` コマンドを入力すると、XVM コマンドの拡張ヘルプ情報を参照できます。たとえば、以下のコマンドを入力すると、`slice` コマンドのオプションが含まれる全画面のヘルプを表示できます。

xvm:cluster> **`help -v slice`**

---

3 方向ストライプを使った論理ボリュームの作成

以下に、3 つのディスクにデータをストライプする単純な論理ボリュームの作成手順の例を示します。この例では、各ディスクの使用可能な全ディスク容量がスライスに使用されています。

図5-1 に、この例で作成される論理ボリュームを示します。

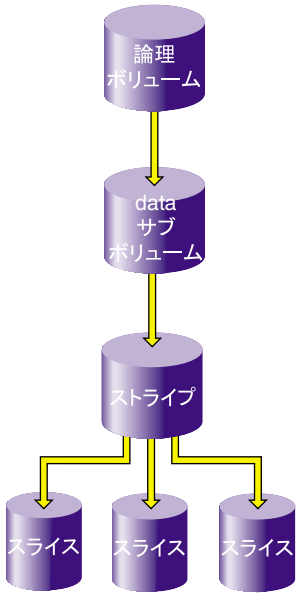

図5-1 3 方向ストライプを使った XVM 論理ボリューム

1. 管理する XVM にディスクを割当てます。この例では、XVM ボリューム・マネージャに 3 つのディスクを割当てます。この手順は、XVM 論理ボリュームの作成に使用する各ディスクに対して一度だけ実行する必要があります。

```
# xvm
xvm:cluster> label -name disk0 dks2d70
disk0
xvm:cluster> label -name disk1 dks2d71
disk1
xvm:cluster> label -name disk2 dks52d72
disk2
```

2. XVM 物理ボリュームとして XVM ボリューム・マネージャに割当てられているすべてのディスクを表示して、どのようにラベルが付けられているかを確認できます。

```
xvm:cluster> show phys/*
phys/disk0                  35542780 online
phys/disk1                  35542780 online
phys/disk2                  35542780 online
```

3. 各 XVM 物理ボリュームのすべての使用可能ブロックで構成されるスライスを作成します。

```
xvm:cluster> slice -all disk*
</dev/cxvm/disk0s0>  slice/disk0s0
</dev/cxvm/disk1s0>  slice/disk1s0
</dev/cxvm/disk2s0>  slice/disk2s0
```

4. 定義した 3 つのスライスで構成されるストライプを作成します。この例では、生成されるボリュームに明示的に stripedvol という名前を付けます。また、data サブボリュームも自動的に生成されます。

   以下のコマンドを使用して、生成されたボリュームに stripedvol という名前を付けます。

```
xvm:cluster> stripe -volname stripedvol slice/disk0s0 slice/disk1s0 slice/disk2s0
</dev/cxvm/stripedvol> stripe/stripe0
```

   この例では、以下のようになっています。

   - ボリューム名 /dev/rcxvm/stripedvol は、mkfs コマンドを実行できるボリュームの名前です。
   - ストライプ名 stripe/stripe0 は、ストライプ・オブジェクトの名前です。

   この例の場合、ストライプ・オブジェクトの名前は以降の起動時に変わりますが、ボリュームの名前は変わりません。

5. 作成した論理ボリュームのトポロジーを参照します。

```
xvm:cluster> show -top stripedvol
vol/stripedvol                      0 online
    subvol/stripedvol/data     106627968 online
        stripe/stripe0             106627968 online,tempname
            slice/disk0s0              35542780 online
            slice/disk1s0              35542780 online
            slice/disk2s0              35542780 online
```

6. exit（または quit あるいは bye）と入力して、xvm ツールを終了します。続いて、ボリュームに対して mkfs コマンドを実行できます。

```
xvm:cluster> exit
# mkfs /dev/cxvm/stripedvol
meta-data=/dev/cxvm/stripedvol   isize=256    agcount=51, agsize=261344 blks
data     =                       bsize=4096   blocks=13328496, imaxpct=25
         =                       sunit=16     swidth=48 blks, unwritten=1
naming   =version 1              bsize=4096
log      =internal log           bsize=4096   blocks=1168
realtime =none                   extsz=65536  blocks=0, rtextents=0
```

7. これで、ファイルシステムをマウントできるようになりました。CXFS クラスタの共有ファイルシステムの場合は、『*CXFS Software Installation and Administration Guide*』の説明に従って、CXFS GUI または CXFS CLI を使用してファイルシステムをマウントします。

   クラスタの一部ではないローカル・ファイルシステムの場合は、fstab ファイルで論理ボリュームを指定して mount コマンドを使用することで、作成したファイルシステムをマウントできます。

ディスクの一部のストライプ

以下に、XVM ボリューム・マネージャを使用してディスクの外側の 1/3 にストライプを作成する場合の例を示します。また、ボリューム要素の命名についてのアドバイスも行います。

図 5-2 に、この例で作成される論理ボリュームを示します。

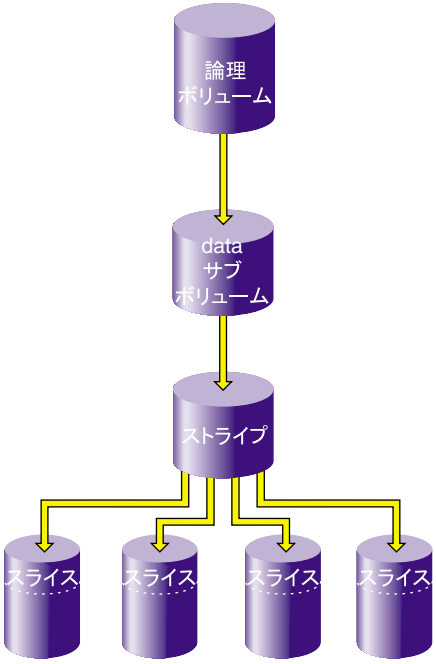

図 5-2 ディスクの一部のストライプ

1. 管理する XVM にディスクを割当てます。この例では、XVM に 4 つのディスクを割当てます。高いストライプ・パフォーマンスを得るために、独立した 4 つのコントローラが選択されている点に注意してください。

```
xvm:cluster> label -name lucy dks21d0
lucy
xvm:cluster> label -name ricky dks22d0
ricky
xvm:cluster> label -name ethyl dks23d0
ethyl
xvm:cluster> label -name fred dks24d0
fred
```

2. この例では、ディスクに 1/3 のパーティションを設定します。

   このディスクを 1/3 のパーティションに設定する方法は 2 つあります。最初の方法では、ディスク全体を割当てることができますが、最後の 1/3 (slice/lucys2) だけを使用します。

```
xvm:cluster> slice -equal 3 lucy
</dev/cxvm/lucys0> slice/lucys0
</dev/cxvm/lucys1> slice/lucys1
</dev/cxvm/lucys2> slice/lucys2
```

   2 番目の方法では、ブロック範囲をディスクの最後の 1/3 に明示的に制限できます。

```
xvm:cluster> slice -start 11852676 -length 5926340 ricky
</dev/cxvm/rickys0> slice/rickys0
xvm:cluster> slice -start 11852676 -length 5926340 ethyl
</dev/cxvm/ethyls0> slice/ethyls0
xvm:cluster> slice -start 11852676 -length 5926340 fred
</dev/cxvm/freds0> slice/freds0
```

3. 割当てを確認します。

   以下に、3 つの均等なストライプに分割されたディスク lucy の割当て例を示します。

```
xvm:cluster> show -v lucy
XVM physvol phys/lucy
=========================
...
-----------------------------------------------------
0 5926338 slice/lucys0
5926338 5926338 slice/lucys1
11852676 5926340 slice/lucys2

Local stats for phys/lucy since being enabled or reset:
-------------------------------------------------------------------
stats collection is not enabled for this physvol
```

以下の例では、明示的に割当てたディスクの 1 つである ricky での割当てを確認します。

```
xvm:cluster> show -v ricky
XVM physvol phys/ricky
========================
...
-------------------------------------------------
0 11852676 (unused)
11852676 5926340 slice/rickys0
-------------------------------------------------------------------
```

1. ストライプを作成します。この例では、生成されるボリュームに明示的に I_Love_Lucy という名前を付けます。

```
xvm:cluster> stripe -volname I_Love_Lucy -unit 128 slice/lucys2 \
slice/rickys0 slice/ethyls0 slice/freds0
</dev/cxvm/I_Love_Lucy> stripe/stripe0
```

2. 場合によっては、複雑なボリュームの各部分を名前で分類すると便利です。たとえば、高速なストライプ・オブジェクトを持つボリュームを検索できるように、ボリュームの一部に faststripe という名前を付けることができます。以下のコマンドでは、ボリュームだけでなくストライプにも名前を付けることができます。

```
xvm:cluster> stripe -volname I_Love_Lucy -vename faststripe0 \
-unit 128 slice/lucys2 slice/rickys0 slice/ethys0 slice/freds0
</dev/cxvm/I_Love_Lucy> stripe/faststripe0
```

この例のようにストライプに名前を付けた場合は、ワイルドカードを使用してすべての高速ストライプを表示できます。

```
xvm:cluster> show -top stripe/fast*
stripe/faststripe0         23705088 online
   slice/lucys2                 5926340 online
   slice/rickys0                5926340 online
   slice/ethyls0                5926340 online
   slice/freds0                 5926340 online
```

また、以下の例のように、ワイルドカードを使用して、「I」で始まるすべてのオブジェクトを表示することもできます。

```
xvm:cluster> show I*
vol/I_Love_Lucy 0 online
```

3. exit（または quit あるいは bye）と入力して、xvm ツールを終了します。これで、ボリュームに対して mkfs コマンドを実行できます。

```
xvm:cluster> exit
% mkfs /dev/cxvm/I_Love_Lucy
hugh3 2# mkfs /dev/cxvm/stripedvol
meta-data=/dev/rxvm/stripedvol   isize=256    agcount=26,
agsize=256416 blks
data     =                       bsize=4096   blocks=6666528,
imaxpct=25
         =                       sunit=16     swidth=48 blks,
unwritten=1
naming   =version 1              bsize=4096
log      =internal log           bsize=4096   blocks=1168
realtime =none                   extsz=65536  blocks=0, rtextents=0
```

4. ファイルシステムをマウントします。CXFS クラスタの共有ファイルシステムの場合は、『*CXFS Software Installation and Administration Guide*』の説明に従って、CXFS GUI または CXFS CLI を使用してファイルシステムをマウントします。

   XVM ボリュームが論理ボリュームの場合は、fstab ファイルで論理ボリュームを指定して mount コマンドを使用することで、作成したファイルシステムをマウントできます。

data サブボリュームと log サブボリュームを使った論理ボリュームの作成

以下に、data サブボリュームと log サブボリュームの両方が含まれる XVM 論理ボリュームを作成する場合の例を示します。この例では、data サブボリュームは 2 つのディスクの使用可能な全容量で構成されており、log サブボリュームは、3 つ目のディスクの使用可能な全容量で構成されています。

図5-3 に、この例で作成される論理ボリュームを示します。

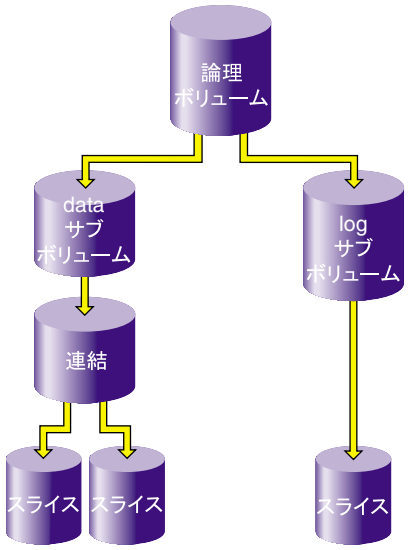

図5-3 log サブボリュームを使った XVM 論理ボリューム

1. 管理する XVM に 3 つのディスクを割当てます。

```
# xvm
xvm:cluster> label -name disk0 dks0d2
disk0
xvm:cluster> label -name disk1 dks0d3
disk1
xvm:cluster> label -name disk2 dks5d42
disk2
```

2. 作成した各 XVM 物理ボリュームのすべての使用可能ブロックで構成されるスライスを作成します。

```
xvm:cluster> slice -all disk*
</dev/xvm/disk0s0>  slice/disk0s0
</dev/xvm/disk1s0>  slice/disk1s0
</dev/xvm/disk2s0>  slice/disk2s0
```

3. 2 つのスライスを組合わせて連結にします。この例では、生成されるボリュームに concatvol という名前を付けます。

```
xvm:cluster> concat -volname concatvol  slice/disk0s0 slice/disk1s0
</dev/cxvm/concatvol>  concat/concat3
```

まだ log サブボリュームが含まれていない定義済みのボリュームの設定を参照できます。

```
xvm:cluster> show -top vol/concatvol
vol/concatvol                       0 online
    subvol/concatvol/data       35554848 online
        concat/concat3              35554848 online,tempname
            slice/disk0s0               17777424 online
            slice/disk1s0               17777424 online
```

4. 作成した 3 つ目のスライスで構成される log サブボリュームを作成します。ボリュームの一時的な名前がシステムによって生成されるよう指定するには、-tempname オプションを使用します。log サブボリュームは既存の concatvol ボリュームに接続するので、このボリュームに名前を付ける必要はありません。

```
xvm:cluster> subvol -tempname -type log slice/disk2s0
</dev/cxvm/vol7_log>  subvol/vol7/log
```

5. log サブボリュームを既存の concatvol ボリュームに接続します。

```
xvm:cluster> attach subvol/vol7/log vol/concatvol
vol/concatvol
```

6. 論理ボリュームを表示します。

```
xvm:cluster> show -top vol/concatvol
vol/concatvol                       0 online
    subvol/concatvol/data       35554848 online
        concat/concat3              35554848 online,tempname
            slice/disk0s0               17777424 online
            slice/disk1s0               17777424 online
    subvol/concatvol/log        17779016 online
        slice/disk2s0               17779016 online
```

data サブボリューム、log サブボリューム、および real-time サブボリュームを使った論理ボリュームの作成

以下に、data サブボリューム、log サブボリューム、および real-time サブボリュームが含まれる XVM 論理ボリュームを作成する場合の例を示します。この手順を実行する方法として、2 つの似た方法を示します。

図 5-4 に、この例で作成される論理ボリュームを示します。

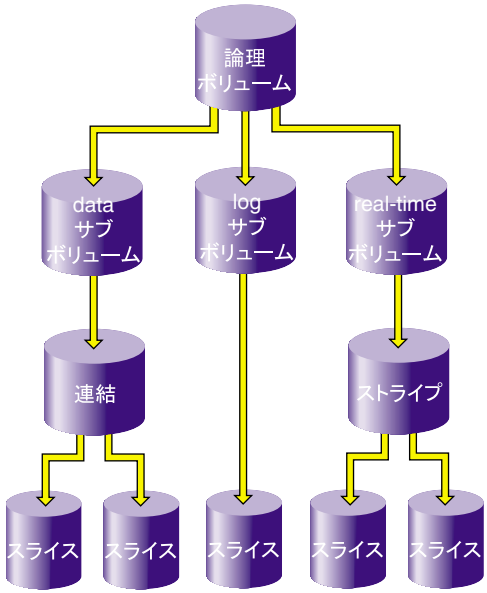

図 5-4 data サブボリューム、log サブボリューム、および real-time サブボリュームを使った論理ボリューム

この例では、管理する XVM にすでにディスクを割当て済みで、論理ボリュームのビルドに使用する 5 つのスライスである `slice/disk1s0`、`slice/disk2s0`、`slice/disk3s0`、`slice/disk4s0`、および `slice/disk5s0` が作成されていることを想定しています。

1. data サブボリュームを構成する連結を作成します。

```
xvm:cluster> concat -tempname slice/disk1s0 slice/disk2s0
</dev/cxvm/vol0>  concat/concat0
```

2. real-time サブボリュームを構成するストライプを作成します。

```
xvm:cluster> stripe -tempname slice/disk3s0 slice/disk4s0
</dev/cxvm/vol1>  stripe/stripe1
```

3. data サブボリュームを作成します。

```
xvm:cluster> subvolume -tempname -type data concat/concat0
</dev/cxvm/vol2>  subvol/vol2/data
```

4. real-time サブボリュームを作成します。

```
xvm:cluster> subvolume -tempname -type rt  stripe/stripe1
</dev/cxvm/vol3_rt>  subvol/vol3/rt
```

5. log サブボリュームを作成します。

```
xvm:cluster> subvolume -tempname -type log slice/disk5s0
</dev/cxvm/vol4_log>  subvol/vol4/log
```

6. これら 3 つのサブボリュームが含まれる論理ボリュームを作成します。

```
xvm:cluster> volume -volname myvol subvol/vol2/data \
subvol/vol4/log  subvol/vol3/rt
vol/myvol
```

7. 論理ボリュームを表示します。

```
xvm:cluster> show -top myvol
vol/myvol                          0 online
    subvol/myvol/data          35558032 online
        concat/concat0             35558032 online,tempname
            slice/disk1s0              17779016 online
            slice/disk2s0              17779016 online
    subvol/myvol/log               8192 online
        slice/disk5s0                  8192 online
    subvol/myvol/rt            35557888 online
        stripe/stripe1             35557888 online,tempname
            slice/disk3s0              17779016 online
            slice/disk4s0              17779016 online
```

以下の一連のコマンドでも同じボリュームを生成できますが、ボリューム名が concat コマンドで指定されるので、手順は 1 つ少なくなります。続いて、log サブボリュームおよび real-time サブボリュームが接続されます。

```
xvm:cluster> concat -volname myvol slice/disk1s0 slice/disk2s0
</dev/cxvm/myvol>  concat/concat1
xvm:cluster> stripe -tempname slice/disk3s0 slice/disk4s0
</dev/cxvm/vol6>  stripe/stripe2
xvm:cluster> subvolume -tempname -type rt  stripe/stripe2
</dev/cxvm/vol7_rt>  subvol/vol7/rt
xvm:cluster> subvolume -tempname -type log slice/disk5s0
</dev/cxvm/vol8_log>  subvol/vol8/log
xvm:cluster> attach subvol/vol8/log  subvol/vol7/rt myvol
vol/myvol
xvm:cluster> show -top myvol
vol/myvol                          0 online
    subvol/myvol/data          35558032 online
        concat/concat1             35558032 online,tempname
            slice/disk1s0              17779016 online
            slice/disk2s0              17779016 online
    subvol/myvol/log               8192 online
        slice/disk5s0                  8192 online
    subvol/myvol/rt            35557888 online
        stripe/stripe2             35557888 online,tempname
            slice/disk3s0              17779016 online
            slice/disk4s0              17779016 online
```

上の階層からのボリュームの作成

XVM 論理ボリュームを設定するときは、ボリュームの階層の下から上へまたは上から下へ作成できます。この節の例では、102 ページの「data サブボリュームと log サブボリュームを使った論理ボリュームの作成」の例および 図5-3 と同じ XVM 論理ボリュームを作成しますが、ボリュームの子ボリューム要素を接続する前にまず空のボリュームを作成します。

1. 管理する XVM に 3 つのディスクを割当てます。

```
# xvm
xvm:cluster> label -name disk0 dks0d2
disk0
xvm:cluster> label -name disk1 dks0d3
disk1
xvm:cluster> label -name disk2 dks5d42
disk2
```

2. 作成した各 XVM 物理ボリュームのすべての使用可能ブロックで構成されるスライスを作成します。

```
xvm:cluster> slice -all disk*
</dev/cxvm/disk0s0>  slice/disk0s0
</dev/cxvm/disk1s0>  slice/disk1s0
</dev/cxvm/disk2s0>  slice/disk2s0
```

3. topdownvol という名前の空のボリュームを作成します。

```
xvm:cluster> volume -volname topdownvol
vol/topdownvol
```

4. ボリュームを表示します。

```
xvm:cluster> show -top vol/top*
vol/topdownvol                    0 offline
    (empty)                         * *
```

5. 空の連結ボリュームを作成して、結果を表示します。

```
xvm:cluster> concat -tempname
</dev/cxvm/vol8>  concat/concat5
xvm:cluster> show -top vol/vol8
vol/vol8                          0 offline,tempname
    subvol/vol8/data                  0 offline,pieceoffline
        concat/concat5                    0 offline,tempname
            (empty)                           * *
```

6. 連結が含まれる生成済み data サブボリュームを topdownvol に接続して、結果を表示します。

```
xvm:cluster> attach subvol/vol8/data vol/topdownvol
vol/topdownvol
xvm:cluster> show -top vol/topdownvol
vol/topdownvol                      0 offline
    subvol/topdownvol/data              0 offline,pieceoffline
        concat/concat5                      0 offline,tempname
            (empty)                             * *
```

7. 2 つのスライスを接続して空の連結を埋め、結果を表示します。

```
xvm:cluster> attach slice/disk0s0 slice/disk1s0 concat/concat5
</dev/cxvm/topdownvol>  concat/concat5
xvm:cluster> show -top vol/topdownvol
vol/topdownvol                      0 online
    subvol/topdownvol/data      35554848 online
        concat/concat5              35554848 online,tempname
            slice/disk0s0               17777424 online
            slice/disk1s0               17777424 online
```

8. log サブボリュームを作成します。

```
xvm:cluster> subvol -tempname -type log
</dev/cxvm/vol9_log>  subvol/vol9/log
```

9. log サブボリュームを topdownvol に接続して、結果を表示します。

```
xvm:cluster> attach subvol/vol9/log vol/topdownvol
vol/topdownvol

xvm:cluster> show -top vol/topdownvol
vol/topdownvol                      0 offline
    subvol/topdownvol/data      35554848 online
        concat/concat5              35554848 online,tempname
            slice/disk0s0               17777424 online
            slice/disk1s0               17777424 online
    subvol/topdownvol/log               0 offline
        (empty)                             * *
```

10. 3 つ目のスライスを log サブボリュームに接続して、結果を表示します。

```
xvm:cluster> attach slice/disk2s0 subvol/topdownvol/log
</dev/cxvm/topdownvol_log>  subvol/topdownvol/log
xvm:cluster> show -top vol/topdownvol
vol/topdownvol                      0 online
    subvol/topdownvol/data     35554848 online
        concat/concat5             35554848 online,tempname
            slice/disk0s0              17777424 online
            slice/disk1s0              17777424 online
    subvol/topdownvol/log      17779016 online
        slice/disk2s0              17779016 online
```

ストライプ・ミラーを使った XVM 論理ボリュームの作成

以下に、ストライプ・ミラーを使用して論理ボリュームを作成する場合の例を示します。この例では、2 つのミラーで構成されるストライプが論理ボリュームに含まれており、各ストライプは、XVM 物理ボリュームのすべての使用可能ブロックで構成されるスライスをミラー化しています。

図5-5 に、この例で作成される論理ボリュームを示します。

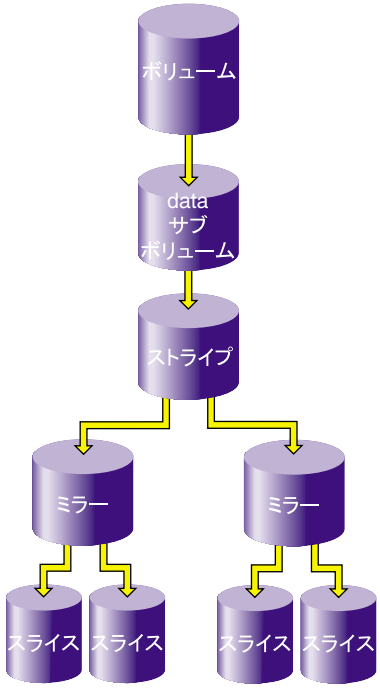

図5-5 ストライプ・ミラーを使った XVM 論理ボリューム

1. 管理する XVM に 4 つのディスクを割当てます。

```
xvm:cluster> label -name disk0 dks2d70
disk0
xvm:cluster> label -name disk1 dks2d71
disk1
xvm:cluster> label -name disk2 dks2d72
disk2
xvm:cluster> label -name disk3 dks2d73i
disk3
```

2. 各 XVM 物理ボリュームのすべての使用可能ブロックからスライスを作成します。

```
xvm:cluster> slice -all disk*
</dev/cxvm/disk0s0>  slice/disk0s0
</dev/cxvm/disk1s0>  slice/disk1s0
</dev/cxvm/disk2s0>  slice/disk2s0
</dev/cxvm/disk3s0>  slice/disk3s0
```

3. それぞれが、定義した 2 つのスライスで構成される 2 つのミラーを作成します。ここでは、読取りより先に書込みが行われる新しいミラーを作成するので、-clean オプションを指定できます。このオプションは、作成時にミラーを同期する必要がないことを示します。

   -clean オプションが指定されていない場合にこのコマンドを実行すると、ミラーの復元が開始されて、スライス上のデータが同期されます。復元が開始されたことを示すメッセージが SYSLOG に書込まれ、復元が完了すると別のメッセージが SYSLOG に書込まれます。

   生成されるボリュームに対して固定の名前を定義する必要はありません。

```
xvm:cluster> mirror -tempname -clean slice/disk0s0 slice/disk1s0
</dev/cxvm/vol2>  mirror/mirror1
xvm:cluster> mirror -tempname -clean slice/disk2s0 slice/disk3s0
</dev/cxvm/vol3>  mirror/mirror2
```

4. 定義した 2 つのミラーで構成されるストライプを作成し、このストライプを格納するために生成されたボリュームに名前を付けます。以下のコマンドを実行して、ミラーをストライプに接続します。

```
xvm:cluster> stripe -volname mirvol mirror/mirror1 mirror/mirror2
</dev/cxvm/mirvol>  stripe/stripe2
```

5. XVM 論理ボリュームを表示します。

```
xvm:cluster> show -top mirvol
vol/mirvol                    0 online
  subvol/mirvol/data     71085312 online
     stripe/stripe2         71085312 online,tempname
         mirror/mirror1         35542780 online,tempname
             slice/disk0s0          35542780 online
             slice/disk1s0          35542780 online
         mirror/mirror2         35542780 online,tempname
             slice/disk2s0           35542780 online
             slice/disk3s0           35542780 online
```

6. exit（または quit あるいは bye）と入力して、xvm ツールを終了します。これで、ボリュームに対して mkfs コマンドを実行できます。

```
xvm:cluster> exit
3# mkfs /dev/cxvm/mirvol
meta-data=/dev/cxvm/mirvol       isize=256    agcount=17, agsize=261440 blks
data     =                       bsize=4096   blocks=4444352, imaxpct=25
         =                       sunit=16     swidth=32 blks, unwritten=1
naming   =version 1              bsize=4096
log      =internal log           bsize=4096   blocks=1168
realtime =none                   extsz=65536  blocks=0, rtextents=0
```

7. ファイルシステムをマウントします。CXFS クラスタのファイルシステムの場合は、『*CXFS Software Installation and Administration Guide*』の説明に従って、CXFS GUI または CXFS CLI を使用してファイルシステムをマウントします。ローカル・ファイルシステムの場合は、fstab ファイルで論理ボリュームを指定して、mount コマンドを使用します。

XVM システム・ディスクの作成とミラー化

この節では、以下のタスクの手順について説明します。

- 「label -mirror コマンドを使ったシステム・ディスクのミラー化」（112 ページ）
- 「ミラーの挿入によるシステム・ディスクのミラー化」（118 ページ）

これら 2 つのミラー化手順は、同じミラー化 root と swap パーティションを作成する 2 つの異なる方法を示します。

さらに、この節では、以下のタスクの手順についても説明します。

- 「動作中の root ディスクでのミラー化 XVM システム・ディスクの作成」（121 ページ）

label -mirror コマンドを使ったシステム・ディスクのミラー化

以下の手順に、XVM システム・ディスクを作成し、label コマンドの -mirror オプションを使用してそのディスクをミラー化する方法を示します。118 ページの「ミラーの挿入によるシステム・ディスクのミラー化」で説明する手順では、ミラーを論理ボリュームに挿入し、スライスを空のミラー・レッグに接続することで、同じミラー化ディスクを作成する方法を説明します。

1. タイプが root の XVM ディスクとしてディスクにラベルを付けます。

   以下のコマンドを実行して dks0d3 にラベルを付け、この physvol の名前を root_1 とします。

```
xvm:local> label -type root -name root_1 dks0d3
root_1
```

このコマンドを実行すると、図5-6 に示すように 2 つのスライスを持つ physvol root_1 が作成されます。

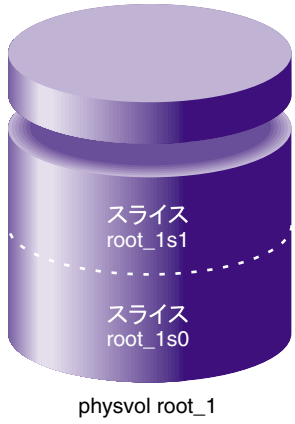

図5-6 XVM システム・ディスク physvol root_1

show -v コマンドを使用して、physvol root_1 のスライスのレイアウトを表示できます。

```
xvm:local> show -v phys/roo*
XVM physvol phys/root_1
=========================
...
Physvol Usage:
Start        Length       Name
--------------------------------------------------
0            262144       slice/root_1s1
262144       17515280     slice/root_1s0
...
```

図5-7 に示すように、このコマンドは、スライスのほかに、論理ボリューム root_1_root0 および論理ボリューム root_1_swap1 も作成します。

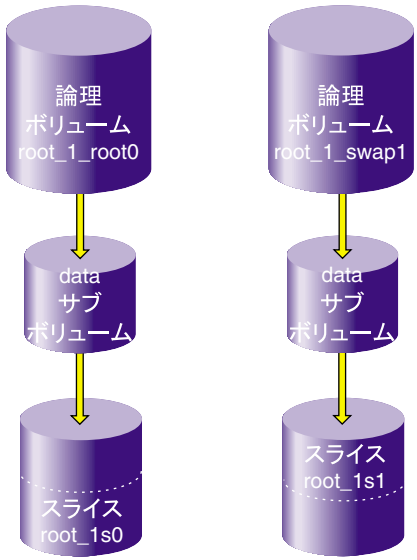

図5-7 ミラー化する前の XVM システム・ディスクの論理ボリューム

show -top コマンドを使用して、論理ボリュームのレイアウトを表示できます。

```
xvm:local> show -top vol/root*
vol/root_1_root0                    0 online
    subvol/root_1_root0/data    17515280 online
        slice/root_1s0              17515280 online

vol/root_1_swap1                    0 online
    subvol/root_1_swap1/data      262144 online
        slice/root_1s1                262144 online
```

2. 新しいシステム・ディスクにオペレーティング・システムをインストールします。

   インストールの前に、XVM ボリューム・マネージャを終了して root ファイルシステムで mkfs コマンドを実行し、root ファイルシステムをマウントしておきます。

```
xvm:local> quit
hugh2 3# mkdir /mnt
hugh2 4# mkfs /dev/lxvm/root_1_root0
meta-data=/dev/lxvm/root_1_root0 isize=256    agcount=9, agsize=243268 blks
data     =                       bsize=4096   blocks=2189410, imaxpct=25
         =                       sunit=0      swidth=0 blks, unwritten=1
naming   =version 1              bsize=4096
log      =internal log           bsize=4096   blocks=1168
realtime =none                   extsz=65536  blocks=0, rtextents=0
hugh2 5# mount /dev/lxvm/root_1_root0 /mnt
```

   これで、/mnt にオペレーティング・システムをインストールできます。

3. 新しい root または swap パーティションを指定するには、root と swap の場所を指定する 4 つの環境変数 root、 OSLoadPartition、SystemPartition、および swap を変更します。環境変数についての詳細は、『*IRIX Admin: System Configuration and Operation*』を参照してください。

   オペレーティング・システムを再起動します。

```
xvm:local> quit
hugh2 2# /etc/reboot

[...]

The system is ready

hugh2 1#
```

4. システム・ディスクのミラーとして使用できるようディスクにラベルを付けます。

   以下の例では、ディスク dks0d4 に、タイプが root で名前が root_2 の XVM physvol としてラベルを付けています。

```
xvm:local> label -type root -name root_2 -mirror root_1 dks0d4
</dev/lxvm/root_1_root0> mirror/mirror2
</dev/lxvm/root_1_swap1> mirror/mirror3
root_2
```

このコマンドを実行すると、図 5-8 に示すように、2 つのスライスが含まれる physvol root_2 が作成されます。

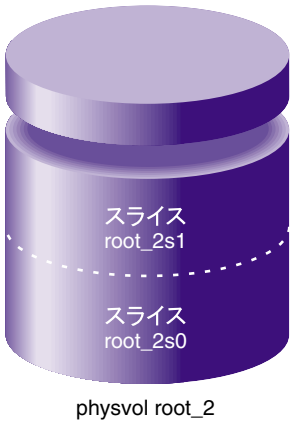

図 5-8 XVM システム・ディスクのミラー physvol root_2

図 5-9 に示すように、このコマンドはスライスを作成するだけでなく、論理ボリューム root_1_root0 および論理ボリューム root_1_swap1 もミラー化します。

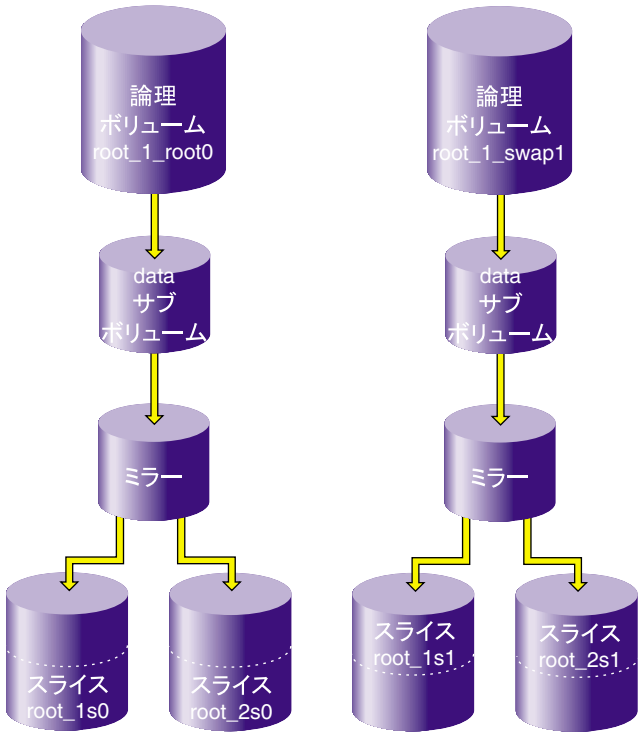

図 5-9 ミラー化完了後の XVM システム・ディスクの論理ボリューム

5. show -top コマンドを使用して、ミラー化 root と swap 論理ボリュームのレイアウトを表示できます。

```
xvm:local> show -top vol/roo*
vol/root_1_root0                  0 online
    subvol/root_1_root0/data  17516872 online
        mirror/mirror2             17516872 online,tempname
            slice/root_1s0            17516872 online
            slice/root_2s0            17516872 online

vol/root_1_swap1                  0 online
    subvol/root_1_swap1/data    262144 online
        mirror/mirror3                262144 online,tempname
            slice/root_1s1               262144 online
            slice/root_2s1               262144 online
```

ミラーの挿入によるシステム・ディスクのミラー化

以下の手順に、112 ページの「label -mirror コマンドを使ったシステム・ディスクのミラー化」で説明した手順で作成したものと同じミラー化システム・ディスクを作成する方法を示します。ただし、この手順では、ミラーを論理ボリュームに挿入してスライスを空のミラー・レッグに接続することで、ミラー化ディスクを作成します。

1. タイプが root の XVM ディスクとしてディスクにラベルを付けます。

   以下のコマンドを実行して dks0d3 にラベルを付け、この physvol の名前を root_1 とします。

   ```
   xvm:local> label -type root -name root_1 dks0d3
   root_1
   ```

   このコマンドを実行すると、2 つのスライスを持つ physvol root_1 が作成されます。これは、図5-6 に示した設定と同じです。

   また、このコマンドは、図 5-7 に示すように、論理ボリューム root_1_root0 および論理ボリューム root_1_swap1 も作成します。

2. 112 ページの「label -mirror コマンドを使ったシステム・ディスクのミラー化」の 2 で説明されている手順に従って、新しいシステム・ディスクにオペレーティング・システムをインストールします。

3. 112 ページの「label -mirror コマンドを使ったシステム・ディスクのミラー化」の 3 で説明されている手順に従って、オペレーティング・システムを再起動します。

4. これらのボリュームの root および swap パーティションを構成するスライスの上になるように、root および swap 論理ボリュームにミラーを挿入します。

   以下のコマンドを実行して、root_1s1 および root_1s0 の上になるように、論理ボリューム root_1_root0 および root_1_swap1 にミラーを挿入します。

   ```
   xvm:local> insert mirror slice/root_1s0
   </dev/lxvm/root_1_root0>  mirror/mirror5
   xvm:local> insert mirror slice/root_1s1
   </dev/lxvm/root_1_swap1>  mirror/mirror6
   ```

ミラーを挿入すると、論理ボリューム root_1_root および root_1_swap は図 5-10 のように設定されます。

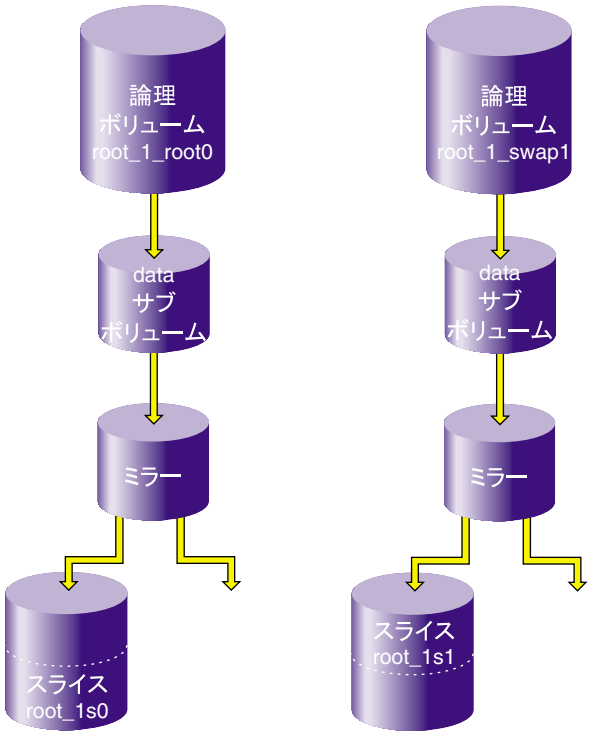

図 5-10 ミラー・コンポーネントを挿入した後の XVM システム・ディスクの論理ボリューム

show -top コマンドを使用して、ミラーを挿入した後の論理ボリューム設定を表示できます。

```
xvm:local> show -top vol/root_1*
vol/root_1_root0                0 online
    subvol/root_1_root0/data  17516872 online
        mirror/mirror5             17516872 online,tempname
            slice/root_1s0           17516872 online

vol/root_1_swap1                0 online
    subvol/root_1_swap1/data    262144 online
        mirror/mirror6              262144 online,tempname
            slice/root_1s1            262144 online
```

5. タイプが root の 2 つ目のシステム・ディスクを作成します。

```
xvm:local> label -type root -name root_2 dks5d7
root_2
```

このコマンドを実行すると、2 つのスライスが含まれる physvol root_2 が作成されます。これは、図5-8 に示す設定と同じです。この 2 つ目のディスクの root およびスライス・パーティションは、ミラー化できるように最初のディスクの root および swap パーティションの容量以上にする必要があります。

このコマンドを実行すると、論理ボリューム root_2_root0 および root_2_swap1 も作成されます。show -top コマンドを使用して、論理ボリューム設定を表示できます。

```
xvm:local> show -top vol/root_2*
vol/root_2_root0                0 online
    subvol/root_2_root0/data  17516872 online
        slice/root_2s0          17516872 online

vol/root_2_swap1                0 online
    subvol/root_2_swap1/data    262144 online
        slice/root_2s1          262144 online
```

6. root_2 上のスライスを、論理ボリューム root_1_root0 および root_1_swap1 に挿入したミラーに接続します。

```
xvm:local> attach slice/root_2s0 mirror/mirror5
</dev/lxvm/root_1_root0>  mirror/mirror5
xvm:local> attach slice/root_2s1 mirror/mirror6
</dev/lxvm/root_1_swap1>  mirror/mirror6
```

これで、root および swap 論理ボリュームは図 5-9 のように構成されました。show -top コマンドを使用して、設定を表示できます。

```
xvm:local> show -top vol/root_1*
vol/root_1_root0                0 online
    subvol/root_1_root0/data  17516872 online
        mirror/mirror5            17516872
online,tempname,reviving:53%
            slice/root_1s0          17516872 online
            slice/root_2s0          17516872 online

vol/root_1_swap1                0 online
    subvol/root_1_swap1/data    262144 online
        mirror/mirror6              262144 online,tempname
            slice/root_1s1          262144 online
            slice/root_2s1          262144 online
```

7. 以下のコマンドを実行すると表示されるように、root_2　上のスライスを論理ボリューム root_1_root0 および root_1_swap1 に接続すると、root_2_root0 と root_2_swap1 は空の論理ボリュームとして残ります。

```
xvm:local> show -top vol/root_2*
vol/root_2_root0                0 offline
    subvol/root_2_root0/data  17516872 offline,incomplete
        (empty)                         * *

vol/root_2_swap1                0 offline
    subvol/root_2_swap1/data    262144 offline,incomplete
        (empty)                         * *
```

これらの空の論理ボリュームは、次回再起動したときに削除されるか、または手動で削除できます。以下のコマンドを実行して、これらの 2 つの空のボリュームを削除し、続く show コマンドで削除を確認します。

```
xvm:local> delete -all vol/root_2_root0 vol/root_2_swap1
xvm:local> show vol/roo*
vol/root_1_root0                0 online
vol/root_1_swap1                 0 online
```

動作中の root ディスクでのミラー化 XVM システム・ディスクの作成

以下の手順では、動作中の root ディスクに XVM システム・ディスクとしてラベルを付け、システムを再起動した後にシステム・ディスクの 3 方向ミラーを作成します。XVM システム・ディスクを作成するときは、ローカル・ドメインにいなければならない点に注意してください。

メモ：既存のシステム・ディスクに XVM ディスクとしてラベルを付ける場合、パーティション 0 とパーティション 1 のレイアウトは現在のレイアウトのまま変更されません。

---

1. ローカル・ドメインから、現在の動作中の root ディスクに XVM システム・ディスクとしてラベルを付けます。

   以下のコマンドを実行して、root ディスク dks0d1 に、タイプが root で xvmdisk という名前の XVM physvol としてラベルを付けます。

```
xvm:local> label -nopartchk -type root -name xvmdisk dks0d1
xvmdisk
```

このコマンドを実行すると、図 5-11 に示すように 2 つのスライスを持つ physvol `xvmdisk` が作成されます。

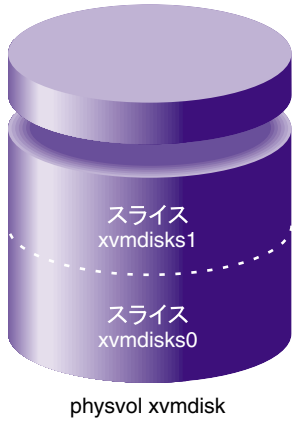

図 5-11 XVM システム・ディスク physvol xvmdisk

図5-12 に示すように、このコマンドを実行すると、root および swap の XVM スライスが作成されるだけでなく、論理ボリューム xvmdisk_root0 および論理ボリューム xvmdisk_swap1 も作成されます。

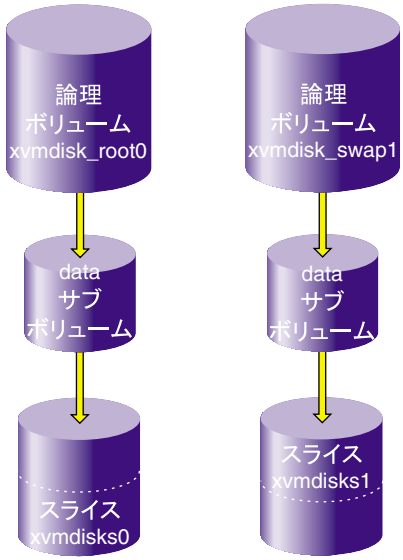

図5-12 XVM 論理ボリューム xvmdisk_root0 および xvmdisk_swap1

2. オペレーティング・システムを再起動します。ディスクがミラー化される前に、動作中の root ディスクの開いているボリュームを閉じて、XVM I/O パスを経由するように開くために再起動が必要です。

```
xvm:local> quit
hugh2 2# /etc/reboot
[...]
The system is ready
hugh2 1#
```

3. 一般的にシステム・ディスクは 1 つのノードだけを起動するために使用されるので、ローカル・ドメインで XVM ボリューム・マネージャを起動します。

```
hugh2 1# xvm -domain local
```

4. XVM システム・ディスクのミラーとして使用する 2 つのディスクにラベルを付けます。

以下のコマンドを実行して、ディスク dks0d3 および dks0d4 に、xvmdisk をミラー化する、タイプが root の XVM ディスクとしてラベルを付けます。ローカル・ドメインで論理ボリュームを作成しているので、これらのボリュームは /dev/lxvm ディレクトリに存在することに注意してください。

```
xvm:local> label -type root -mirror xvmdisk dks0d3 dks0d4
</dev/lxvm/xvmdisk_swap1>  mirror/mirror0
</dev/lxvm/xvmdisk_root0>  mirror/mirror1
dks0d3
dks0d4
```

このコマンドを実行すると、dks0d3 および dks0d4 という名前の 2 つの physvol がデフォルトで作成されます。図 5-13 に示すように、これらの各 physvol には 2 つのスライスが含まれます。

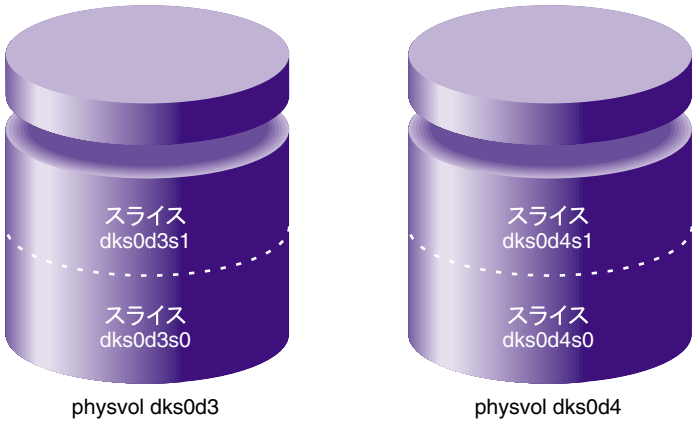

図 5-13 XVM システム・ディスク physvol のミラー

図 5-14 に示すように、このコマンドはスライスを作成するだけでなく、論理ボリューム xvmdisk_root0 および論理ボリューム xvmdisk_swap1 も 3 方向ミラーとしてミラー化します。

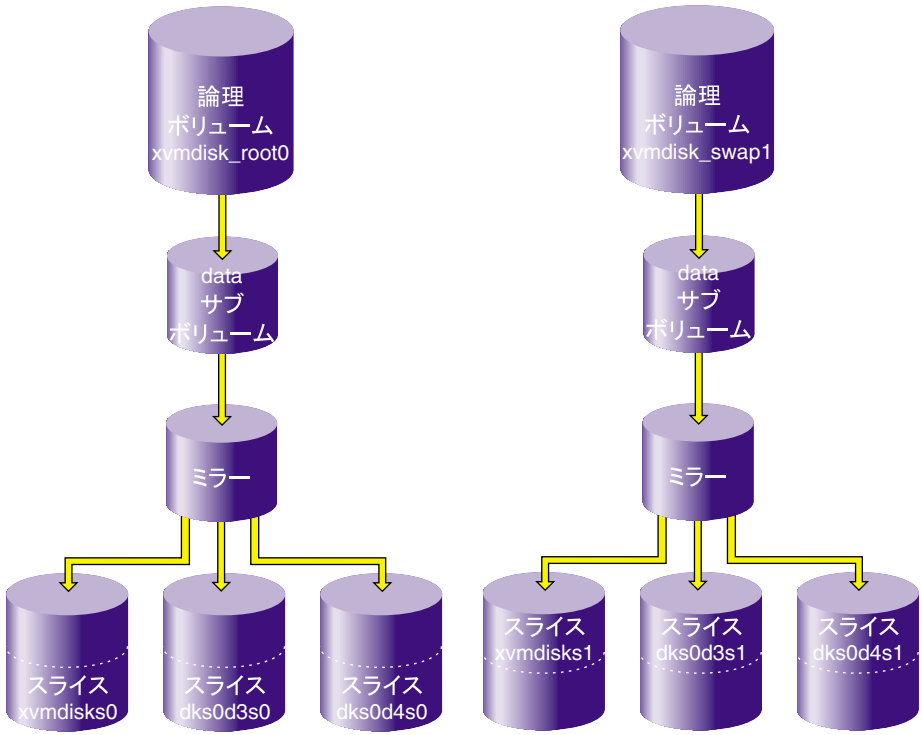

図 5-14 XVM システム・ディスク physvol xvmdisk のミラー化論理ボリューム

5. show -v コマンドを使用して、xvmdisk、dks0d3、および dks0d4 上のスライスのレイアウトを表示できます。

```
xvm:local> show -v phys/*

XVM physvol phys/dks0d3
========================

...

Physvol Usage:
Start        Length       Name
--------------------------------------------------
0            262144       slice/dks0d3s1
262144       17515280     slice/dks0d3s0

...

XVM physvol phys/dks0d4
========================

...

Physvol Usage:
Start        Length       Name
--------------------------------------------------
0            262144       slice/dks0d4s1
262144       17515280     slice/dks0d4s0
...

XVM physvol phys/xvmdisk
========================

...

Physvol Usage:
Start        Length       Name
--------------------------------------------------
0            262144       slice/xvmdisks1
262144       17515280     slice/xvmdisks0
...
```

6. show -top コマンドを使用して、ミラー化 root および swap 論理ボリュームのレイアウトを表示できます。この例では、ミラーの再生がちょうど開始されたところです。

```
xvm:local> show -top vol

vol/xvmdisk_root0                0 online
    subvol/xvmdisk_root0/data  17515280 online,open
        mirror/mirror0          17515280 online,tempname,reviving:2%,open
            slice/xvmdisks0         17515280 online,open
            slice/dks0d3s0          17515280 online,open
            slice/dks0d4s0          17515280 online,open

vol/xvmdisk_swap1                0 online
    subvol/xvmdisk_swap1/data    262144 online,open
        mirror/mirror1            262144 online,tempname,reviving:queued,open
            slice/xvmdisks1           262144 online,open
            slice/dks0d3s1            262144 online,open
            slice/dks0d4s1            262144 online,open
```

連結を使った swap ボリュームの設定

以下の手順に、連結が含まれる swap ボリュームを使用して XVM システム・ディスクを設定する方法を示します。この手順では、現在動作中の root ディスクに再度ラベルは付けません。

この手順では、図 5-15 に示すように、XVM システム・ディスク上の 1 つのスライスで構成される root 論理ボリュームと、連結を構成する 2 つのスライスで構成される swap 論理ボリュームを作成します。

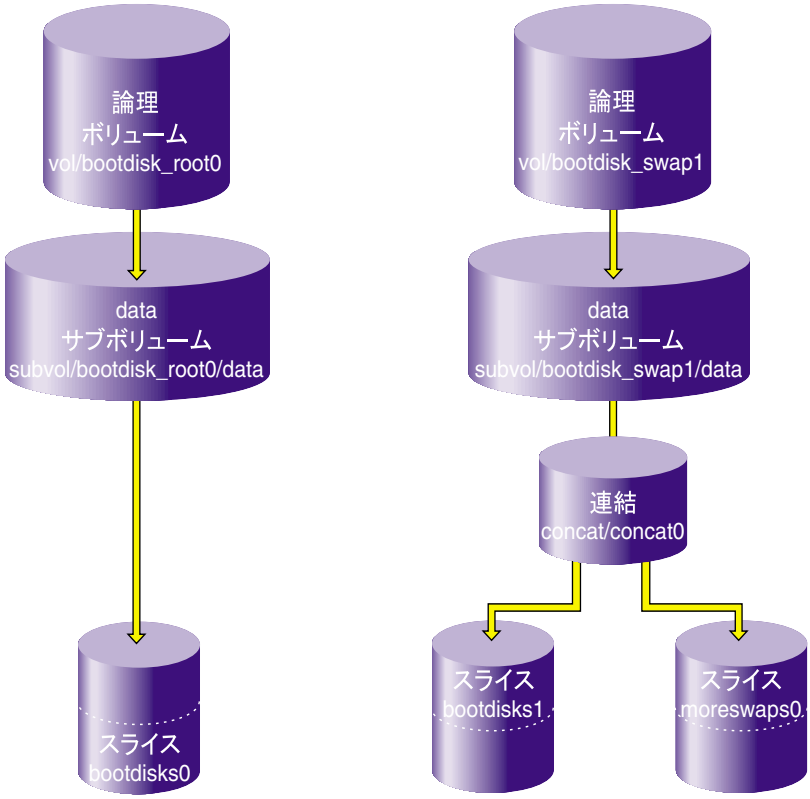

図 5-15 連結を使った XVM swap ボリューム

この例では、swap ボリュームを構成する 2 つのスライスは、2 つの異なるディスク上にあります。

1. 以下のコマンドを実行して、ディスク `dks1d2` に、`bootdisk` という名前の XVM システム・ディスクとしてラベルを付けます。XVM `label` コマンドの `-clrparts` オプションは、ディスクにすでにパーティション 0 が含まれている場合に、ディスクの既存のパーティション設定スキームをオーバーライドします。

   ```
   xvm:local> label -clrparts -type root -name bootdisk dks1d2
   bootdisk
   ```

このコマンドを実行すると、図 5-16 に示すように 2 つのスライスを持つ physvol `bootdisk` が作成されます。

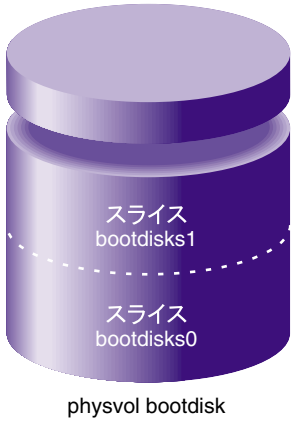

図 5-16 XVM システム・ディスク physvol bootdisk

図5-17 に示すように、このコマンドを実行すると、root および swap の XVM スライスが作成されるだけでなく、論理ボリューム bootdisk_root0 および bootdisk_swap1 も作成されます。

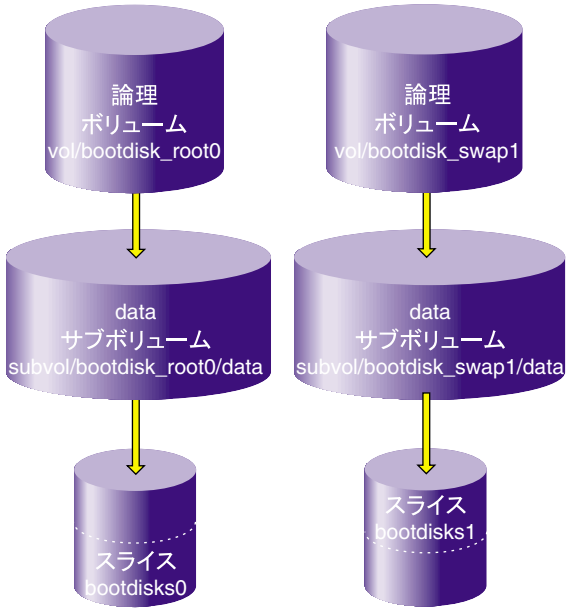

図 5-17 XVM 論理ボリューム bootdisk_root0 および bootdisk_swap1

show -top コマンドを実行すると、論理ボリュームのトポロジーと、root および swap スライスのサイズが表示されます。

```
xvm:local> show -top vol/bootdis*
vol/bootdisk_root0                  0 online
    subvol/bootdisk_root0/data    17515280 online
        slice/bootdisks0            17515280 online

vol/bootdisk_swap1                  0 online
   subvol/bootdisk_swap1/data       262144 online
        slice/bootdisks1            262144 online
```

2. 以下のコマンドを実行して、2 つ目のディスク dks1d3 に、moreswap という名前の XVM システム・ディスクとしてラベルを付けます。このディスクには root パーティションは必要なく、swap ボリュームは手動で定義するので、XVM label コマンドの -noparts オプションを使用します。この例では、この 2 つ目のディスクが、パーティション 0 が含まれるシステム・ディスクにまだ存在しないことを想定しています。その他の場合は、XVM label コマンドの -clrparts オプションを使用する必要があります。

```
xvm:local> label -noparts -type root -name moreswap dks1d3
moreswap
```

3. moreswap 上に swap スライスを作成します。

```
xvm:local> slice -type swap -start 0 -length 262144 moreswap
</dev/lxvm/moreswaps0> slice/moreswaps0
```

このコマンドを実行すると、図 5-18 に示すように、physvol moreswap 上に swap スライスが作成されます。

図 5-18 XVM システム・ディスク physvol moreswap

4. 2 つの swap スライスを接続する、2 つの断片で構成される空の連結を作成します。

```
xvm:local> concat -tempname -pieces 2
</dev/lxvm/vol15> concat/concat3
```

5. 2 つの swap スライスを連結に接続します。

```
xvm:local> attach slice/bootdisks1 slice/moreswaps0 concat3
</dev/lxvm/vol15> concat/concat3
```

6. この連結を swap サブボリュームに接続します。

```
xvm:local> attach concat3 subvol/bootdisk_swap1/data
</dev/lxvm/bootdisk_swap1> subvol/bootdisk_swap1/data
```

これにより、128 ページの図 5-15 に示す論理 swap ボリュームが作成されます。

show -top コマンドを使用して、作成した root および swap 論理ボリュームのレイアウトを表示できます。

```
xvm:local> show -t vol/bootdis*
vol/bootdisk_root0                  0 online
    subvol/bootdisk_root0/data   17515280 online
        slice/bootdisks0           17515280 online

vol/bootdisk_swap1                  0 online
    subvol/bootdisk_swap1/data      524288 online
        concat/concat3                524288 online,tempname
            slice/bootdisks1              262144 online
            slice/moreswaps0              262144 online
```

ミラー化を使ったオンライン再設定

以下の手順では、新しい設定でデータをミラー化して元の設定を分離することにより、ファイルシステムのオンライン中にファイルシステムを再設定します。この手順を実行するためにファイルシステムをアンマウントする必要はありません。

図 5-19 に、ビルドおよびマウントされている元のファイルシステムの設定を示します。

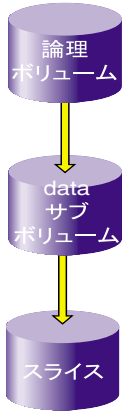

図 5-19 オンラインの元のファイルシステム

この例では、元のファイルシステムは単一のスライスで構成されるファイルシステムです。このファイルシステムには myfs という名前が付いており、以下のように設定されています。

```
xvm:cluster> show -top myfs
vol/myfs                              0 online
    subvol/myfs/data                102400 online,open
        slice/disk5s0                   102400 online,open
```

以下の手順で、このファイルシステムを、4 方向ストライプで構成される 1 つのファイルシステムに再設定します。

1. 4 方向ストライプを構成するスライスを作成します。作成するストライプは既存のファイルシステムと同じサイズにする必要があるので、この例の各スライスは、ファイルシステムのサイズの 1/4 になります。

   ```
   xvm:cluster> slice -length 25600 phys/disk[1234]
   </dev/cxvm/disk1s0>  slice/disk1s0
   </dev/cxvm/disk2s0>  slice/disk2s0
   </dev/cxvm/disk3s0>  slice/disk3s0
   </dev/cxvm/disk4s0>  slice/disk4s0
   ```

2. 4 方向ストライプを作成します。この例では、ストライプ単位は指定せず、デフォルトのストライプ単位である 128 ブロックをそのまま使用します。この場合、スライスのサイズはストライプ単位の倍数なので、デフォルトのストライプ単位を使用することで、各スライスのすべてのブロックが使用されます。

   ```
   xvm:cluster> stripe -tempname slice/disk[1234]s0
   </dev/cxvm/vol5>  stripe/stripe5
   ```

   ストライプ設定を表示します。

   ```
   xvm:cluster> show -top stripe5
   stripe/stripe5                  102400 online,tempname
       slice/disk1s0                    25600 online
       slice/disk2s0                    25600 online
       slice/disk3s0                    25600 online
       slice/disk4s0                    25600 online
   ```

3. 再設定するポイントの上に一時ミラーを挿入します。この例では、該当するポイントは slice/disk5s0 です。

```
xvm:cluster> insert mirror slice/disk5s0
</dev/cxvm/myfs>  mirror/mirror5
```

論理ボリュームを表示します。

```
xvm:cluster> show -top myfs
vol/myfs                              0 online
    subvol/myfs/data                102400 online,open
        mirror/mirror5                  102400 online,tempname,open
            slice/disk5s0                   102400 online,open
```

図 5-20 に、ミラーを挿入した後のファイルシステム myfs の設定を示します。

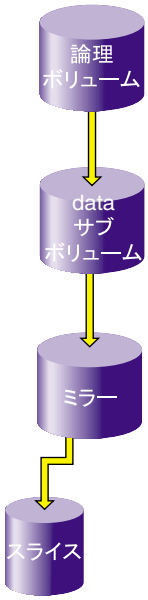

図 5-20 ミラーを挿入した後のファイルシステム

4. ストライプをミラー（この例では mirror5）に接続します。 これにより再生が開始され、stripe5 上の slice/disk5s0 のデータが複製されます。

```
xvm:cluster> attach stripe/stripe5 mirror/mirror5
</dev/cxvm/myfs>  mirror/mirror5
```

論理ボリュームを表示します。

```
xvm:cluster> show -top myfs
vol/myfs                          0 online
    subvol/myfs/data             102400 online,open
        mirror/mirror5               102400 online,tempname,open
            slice/disk5s0                102400 online,open
            stripe/stripe5               102400 online,tempname,open
                slice/disk1s0                 25600 online,open
                slice/disk2s0                 25600 online,open
                slice/disk3s0                 25600 online,open
                slice/disk4s0                 25600 online,open
```

図 5-21 に、ストライプがミラーに接続された後のファイルシステム myfs の設定を示します。

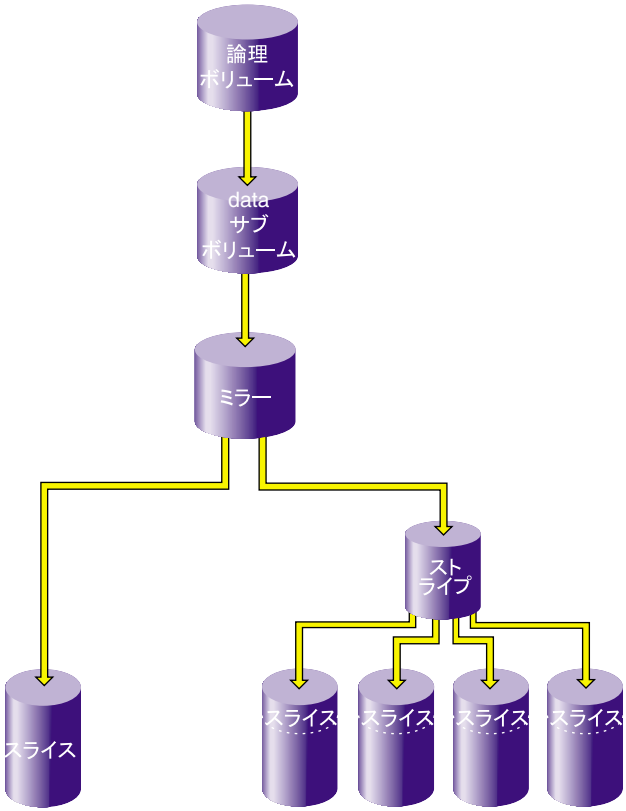

図 5-21 ストライプをミラーに接続した後のファイルシステム

5. slice/disk5s0 をミラーから分離します。開いているミラーの最後の有効な断片は分離できず、再生が完了するまではこのスライスがミラーの唯一の有効なレッグなので、分離を行うには、ミラーの再生が完了するまで待つ必要があります。

```
xvm:cluster> detach slice/disk5s0
</dev/cxvm/disk5s0>  slice/disk5s0
```

図 5-22 に、元のスライスを分離した後のファイルシステム myfs の設定を示します。

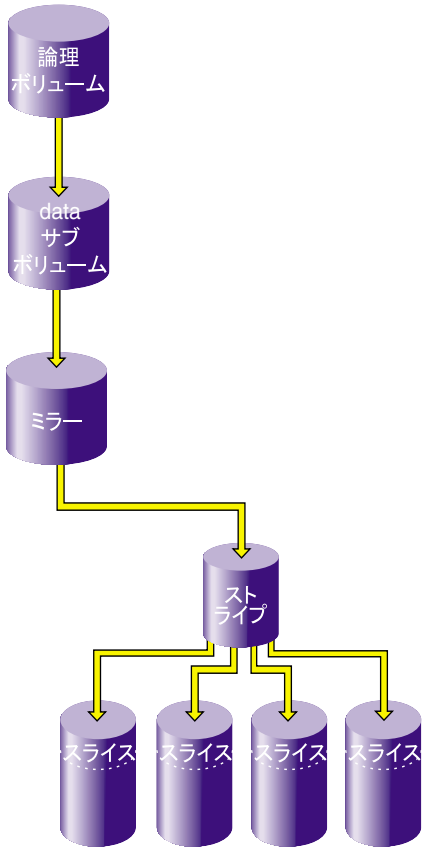

図 5-22 元のスライスを分離した後のファイルシステム

6. ミラーを解除することで、ツリーからミラー層を削除します。

```
xvm:cluster> collapse mirror/mirror5
```

これで、ファイルシステムが 4 方向ストライプとして設定されました。論理ボリュームを表示します。

```
xvm:cluster> show -top myfs
vol/myfs                              0 online
    subvol/myfs/data                102400 online,open
        stripe/stripe5                  102400 online,tempname,open
            slice/disk1s0                    25600 online,open
            slice/disk2s0                    25600 online,open
            slice/disk3s0                    25600 online,open
            slice/disk4s0                    25600 online,open
```

図 5-23 に、ファイルシステム myfs の最終的な設定を示します。

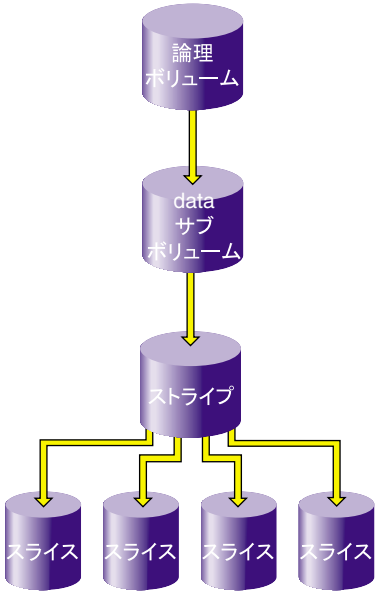

図 5-23 再設定されたファイルシステム

オンラインでの論理ボリュームの変更

以下の節では、論理ボリュームの作成および変更手順について説明します。この手順では、論理ボリューム上にファイルシステムを作成およびマウントした後に、論理ボリュームを変更します。

この手順は、以下の手順に分かれます。

- 論理ボリュームの作成
- 論理ボリュームの拡張
- 論理ボリュームのデータのミラー化
- ミラー化を使ったストライプ化データへの連結の変換
- ミラーの削除
- 個々のストライプ・メンバーのミラー化

以下の節では、これらの手順について説明していきます。

論理ボリュームの作成

以下の手順で、単一のスライスが含まれる単純な論理ボリュームを作成します。図 5-24 に、この手順で作成される元の XVM 論理ボリュームを示します。

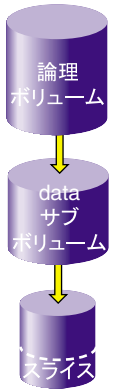

図 5-24 元の XVM 論理ボリューム

1. physvol pebble にスライスを作成し、そのスライスを含む生成されたボリュームの名前を tinyvol とします。

```
xvm:cluster> slice -volname tinyvol -start 17601210 -length 177792 \
pebble </dev/cxvm/tinyvol> slice/pebbles0
```

2. exit と入力して XVM CLI を終了し、ファイルシステムを作成します。

```
xvm:cluster> exit
# mkfs /dev/cxvm/tinyvol
meta-data=/dev/cxvm/tinyvol  isize=256   agcount=5, agsize=4445 blks
data     =                   bsize=4096  blocks=22224, imaxpct=25
         =                   sunit=0     swidth=0 blks, unwritten=1
naming   =version 1          bsize=4096
log      =internal log       bsize=4096  blocks=1168
realtime =none               extsz=65536 blocks=0, rtextents=0
```

3. ファイルシステムをマウントします。CXFS クラスタの共有ファイルシステムの場合は、『*CXFS Software Installation and Administration Guide*』の説明に従って、CXFS GUI または CXFS CLI を使用してファイルシステムをマウントします。

   ローカル・ファイルシステムの場合は、fstab ファイルで論理ボリュームを指定し、mount コマンドを使用してファイルシステムをマウントします。

論理ボリュームの拡張

以下の手順で、作成した論理ボリュームを拡張します。図 5-25 に、論理ボリュームを拡張するための連結を挿入した後の論理ボリュームを示します。

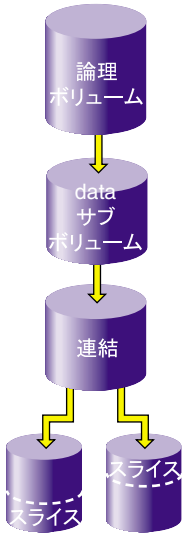

図 5-25 挿入後の XVM 論理ボリューム

1. 論理ボリューム tinyvol とこのボリュームのトポロジーを表示します。

```
xvm:cluster> show -top tinyvol
vol/tinyvol                          0 online
    subvol/tinyvol/data            177792 online,open
        slice/pebbles0                 177792 online,open
```

2. ボリューム tinyvol を変更して、連結コンテナを含めます。

```
xvm:cluster> insert concat slice/pebbles0
</dev/cxvm/tinyvol> concat/concat3
```

3. insert コマンドの結果を表示します。

```
xvm:cluster> show -top tinyvol
vol/tinyvol                          0 online
    subvol/tinyvol/data            177792 online,open
        concat/concat3                 177792 online,tempname,open
            slice/pebbles0                 177792 online,open
```

4. physvol bambam で空きスライスを検索または作成します。

```
xvm:cluster> slice -start 0 -length 177792 bambam
</dev/xvm/bambams0> slice/bambams0
```

5. このスライスを tinyvol に接続します。スライスを接続する連結ボリューム要素を指定するには、2 つの異なる方法があります。

   以下のコマンドでは、ボリューム要素の相対位置を使用してスライスを接続します。

```
xvm:cluster> attach slice/bambams0 tinyvol/data/0
</dev/cxvm/tinyvol> concat/concat3
```

   以下のコマンドでは、ボリューム要素のオブジェクト名を参照することでスライスを接続します。

```
xvm:cluster> attach slice/bambams0 concat3
```

   XVM コマンドでのオブジェクト名の参照と相対位置については、51 ページの「XVM でのオブジェクト名」を参照してください。

6. attach コマンドの結果を表示します。

```
xvm:cluster> show -top tinyvol
vol/tinyvol                          0 online
    subvol/tinyvol/data            355584 online,open
        concat/concat3                 355584 online,tempname,open
            slice/pebbles0                 177792 online,open
            slice/bambams0                 177792 online,open
```

7. exit と入力して XVM CLI を終了し、ファイルシステムを拡張します。CXFS GUI でファイルシステムをマウントしたマウント・ポイントを使用してください。この例では、マウント・ポイントは /clusterdisk になります。

```
xvm:cluster> exit
# xfs_growfs /clusterdisk
meta-data=/clusterdisk     isize=256    agcount=5, agsize=4445 blks
data     =                 bsize=4096   blocks=22224, imaxpct=25
         =                 sunit=0      swidth=0 blks, unwritten=1
naming   =version 1        bsize=4096
log      =internal         bsize=4096   blocks=1168
realtime =none             extsz=65536  blocks=0, rtextents=0
data blocks changed from 22224 to 44448
```

論理ボリュームのデータのミラー化

以下の手順で、ファイルシステム内のデータのミラーを作成します。図5-26 に、ミラーを挿入した後の XVM 論理ボリュームを示します。

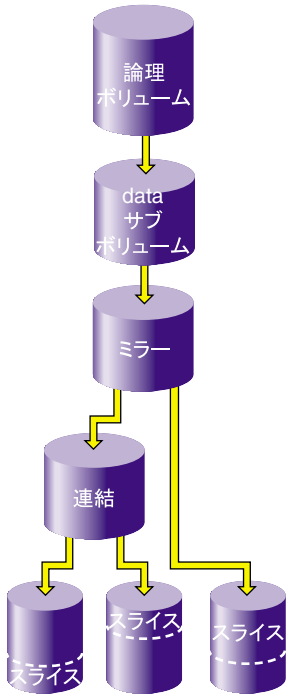

図5-26 ミラー化した後の XVM 論理ボリューム

1. ボリューム tinyvol を変更して、ミラー・コンテナを含めます。

```
# xvm
xvm:cluster> insert mirror tinyvol/data/0
</dev/cxvm/tinyvol>  mirror/mirror3
```

2. ミラーを挿入した結果を表示します。

```
xvm:cluster> show -top tinyvol
vol/tinyvol                    0 online
    subvol/tinyvol/data      355584 online,open
        mirror/mirror3           355584 online,tempname,open
            concat/concat3           355584 online,tempname,open
                slice/pebbles0           177792 online,open
                slice/bambams0           177792 online,open
```

3. 空き容量を見つけるか、または同じサイズの新しいスライスを作成します。

```
xvm:cluster> slice -start 0 -length 355584 wilma
</dev/cxvm/wilmas0>  slice/wilmas0
```

4. このスライスをミラーに接続します。

```
xvm:cluster> attach slice/wilmas0 tinyvol/data/0
</dev/cxvm/tinyvol>  mirror/mirror3
```

5. 接続の結果を表示します。この例では、スライスをミラーに接続したときに開始された再生はまだ完了していません。

```
xvm:cluster> show -top tinyvol
vol/tinyvol                  0 online
     subvol/tinyvol/data    355584 online,open
         mirror/mirror3         355584 online,tempname,open
             concat/concat3         355584 online,tempname,open
                 slice/pebbles0         177792 online,open
                 slice/bambams0         177792 online,open
             slice/wilmas0          355584 online,reviving:11%
```

ミラー化を使った連結のストライプへの変換

以下の手順で、以前に作成した連結を、ミラー内の連結に代わるストライプに変換します。図 5-27 に、変換後の XVM 論理ボリュームを示します。

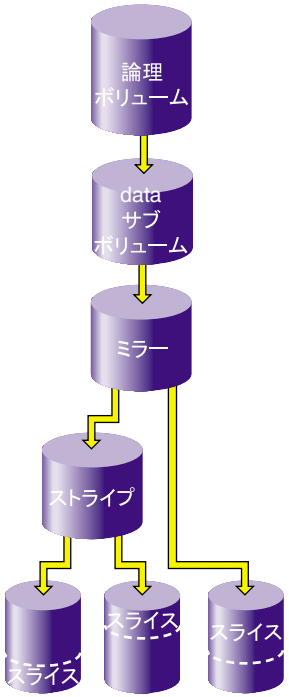

図 5-27 連結をミラーに変換した後の XVM 論理ボリューム

1. ミラーを分離します。

```
xvm:cluster> detach -tempname mirror3/0
</dev/cxvm/vol6>  concat/concat3
```

2. 連結オブジェクトを削除します。その連結オブジェクトを構成するスライスは、分離して保持します。

```
xvm:cluster> delete -nonslice concat3
</dev/cxvm/pebbles0>  slice/pebbles0
</dev/cxvm/bambams0>  slice/bambams0
```

3. これらのスライスを使用してストライプを作成します。

```
xvm:cluster> stripe -tempname -unit 128 slice/pebbles0 slice/bambams0
</dev/cxvm/vol7>  stripe/stripe0
```

4. ストライプをミラーに接続します。

```
xvm:cluster> attach stripe0 mirror3
```

5. 接続の結果を表示します。この例では、ストライプをミラーに接続したときに開始された再生はまだ完了していません。

```
xvm:cluster> show -top tinyvol
vol/tinyvol                  0 online
   subvol/tinyvol/data    355584 online,open
       mirror/mirror3         355584 online,tempname,open
           stripe/stripe0         355584 online,tempname,reviving:5%
               slice/pebbles0         177792 online,open
               slice/bambams0         177792 online,open
           slice/wilmas0          355584 online,open
```

ミラーの削除

以下の手順で、論理ボリュームからミラー層を削除します。図 5-28 に、ミラーを削除した後の XVM 論理ボリュームを示します。

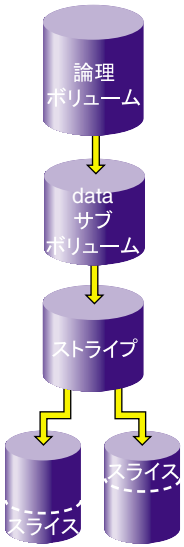

図 5-28 ミラーを削除した後の XVM 論理ボリューム

1. データがミラー化されていたスライスを分離します。

```
xvm:cluster> detach -tempname slice/wilmas0
</dev/cxvm/wilmas0>  slice/wilmas0
```

2. ミラー層を削除します。

```
xvm:cluster> collapse mirror3
```

3. collapse コマンドの結果を表示します。

```
xvm:cluster> show -top tinyvol
vol/tinyvol                          0 online
    subvol/tinyvol/data            355584 online,open
        stripe/stripe0                 355584 online,tempname,open
            slice/pebbles0                 177792 online,open
            slice/bambams0                 177792 online,open
```

個々のストライプ・メンバーのミラー化

以下の手順で、ストライプを構成する個々のスライスをミラー化します。図 5-29 に、この例で作成される XVM 論理ボリュームを示します。

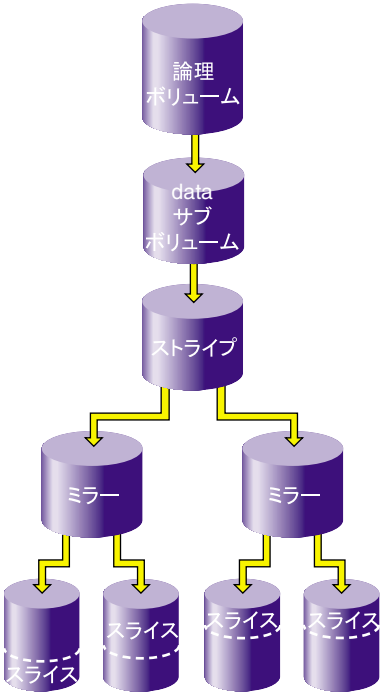

図 5-29 スライスをミラー化した後の XVM 論理ボリューム

1. ミラー・コンテナ内にスライスを配置します。以下の例に、スライスを指定する 2 つの方法を示します。

```
xvm:cluster> insert mirror tinyvol/data/0/0
</dev/cxvm/tinyvol>  mirror/mirror4

xvm:cluster> insert mirror slice/bambams0
</dev/cxvm/tinyvol>  mirror/mirror5
```

2. 2 つの insert コマンドの結果を表示します。

```
xvm:cluster> show -top tinyvol
vol/tinyvol                   0 online
     subvol/tinyvol/data      355584 online,open
         stripe/stripe0           355584 online,tempname,open
             mirror/mirror4           177792 online,tempname,open
                 slice/pebbles0           177792 online,open
             mirror/mirror5           177792 online,tempname,open
                 slice/bambams0           177792 online,open
```

3. 空き容量を見つけるか、複数の未使用スライスを流用します。

```
xvm:cluster> slice -start 0 -length 177792 betty
</dev/cxvm/bettys0>  slice/bettys0

xvm:cluster> show slice/wilmas0
slice/wilmas0                 355584 online,autoname
```

4. スライスをミラーに接続します。wilmas0 は pebbles0 よりも大きい点に注意してください。このミラーは、引続き最小サイズを使用します。

```
xvm:cluster> attach slice/wilmas0 tinyvol/data/0/0
</dev/cxvm/tinyvol>  mirror/mirror4

xvm:cluster> attach slice/bettys0 stripe0/1
</dev/cxvm/tinyvol>  mirror/mirror4
```

5. 接続の結果を表示します。

```
xvm:cluster> show -top tinyvol
vol/tinyvol                   0 online
     subvol/tinyvol/data      355584 online,open
         stripe/stripe1           355584 online,tempname,open
             mirror/mirror4           177792 online,tempname,open
                 slice/pebbles0           177792 online,open
                 slice/wilmas0            355584 online,open
             mirror/mirror5           177792 online,tempname,open
                 slice/bambams0           177792 online,open
                 slice/bettys0            177792 online,open
```

統計データ

XVM ボリューム・マネージャでは、physvol、サブボリューム、ストライプ、連結、ミラー、およびスライスの統計を維持できます。

change コマンドの stat オプションを使用して、統計をオンまたはオフにしたり、ボリューム要素の統計をリセットできます。統計はデフォルトではオフです。change コマンドを使用して統計をオンにすると、指定した層の統計だけが収集されます。XVM 論理ボリュームの複数の層の統計を収集したい場合は、各層を明示的に指定してください。

クラスタ化された環境では、ローカル・セルの統計だけが維持されます。

すべてのボリューム要素と physvol の統計では、読取り操作と書込み操作の数のほかに、512 バイト・ブロック読取りと書込みの数も表示されます。以下の節では、ボリューム要素の各タイプに固有の統計について説明します。

- 「物理ボリュームの統計」(152 ページ)
- 「サブボリュームの統計」(152 ページ)
- 「ストライプの統計」(152 ページ)
- 「連結の統計」(154 ページ)
- 「ミラーの統計」(155 ページ)
- 「スライスの統計」(156 ページ)

物理ボリュームの統計

以下に、show コマンドを実行して、統計がオンになっている物理ボリュームの統計を表示した結果の例を示します。

```
xvm:cluster> show -stat betty

Local stats for phys/betty since being enabled or reset:
------------------------------------------------------------------------
client read requests:          3
client write requests:         42
client 512 byte blks read:     257
client 512 byte blks written:  4681
```

サブボリュームの統計

以下に、show コマンドを実行して、統計がオンになっているサブボリュームの統計を表示した結果の例を示します。

```
xvm:cluster> show -stat tinyvol/data
Local stats for subvol/tinyvol/data since being enabled or reset:
------------------------------------------------------------------------
read requests:          12
write requests:         109
512 byte blks read:     1034
512 byte blks written:  9533
```

ストライプの統計

ストライプの統計では、操作のサイズとストライプ幅のサイズ、および操作が 512 バイトの境界に配置されているかどうかが表示されます。最も回数の多い要求が始点と終点の両方に配置されていると、最適なパフォーマンスが得られます。

以下に、統計がオンになっているストライプに対して show コマンドを実行した結果の例を示します。

```
xvm:cluster> show -v stripe/stripe0
XVM ve stripe/stripe0
=======================
volname: vol/cxfsvol  subvolume: subvol/cxfsvol/data
size: 35557888  iou: 1  pieces: 2  open: no
state: 0xa (valid,online) user-flags: online,autoname
uuid: 31350c60-7aa4-1022-85f1-0800690592c9
tid: 922806085 (03/30/99_09:01:25)

Type-specific information:
---------------------------
stripe unit size: 128

Local stats for stripe/stripe0 since being enabled or reset:
------------------------------------------------------------------------
read requests:          1826
write requests:         0
512 byte blks read:     15236
512 byte blks written:  0
Requests aligned at both start and end
        equal to stripe width:      10
        greater than stripe width:  10
        less than stripe width:     0
Requests aligned at start
        greater than stripe width:  0
        less than stripe width:     20
Requests aligned at end
        greater than stripe width:  0
        less than stripe width:     19

Requests unaligned
        equal to stripe width:      0
        greater than stripe width:  0
        less than stripe width:     1767

Pieces:
#    Size     Timestamp         Type/Name            State
-------------------------------------------------------------
0    17779016 03/30/99_09:00:49 slice/cxfsdsk1s0     valid,online
1    17779016 03/30/99_09:01:01 slice/cxfsdsk2s0     valid,online
```

連結の統計

連結の統計では、ある断片とその次の断片の境界を超える操作の数が表示されます。

以下に、統計がオンになっている連結に対して show コマンドを実行した結果の例を示します。

```
xvm:cluster> show -v concat/concat3
XVM ve concat/concat3
========================
volname: vol/vol3  subvolume: subvol/vol3/data
size: 11110890  iou: 1  pieces: 5  open: no
state: 0xa (valid,online) user-flags: online,autoname
uuid: 39798cff-9c4f-1022-8544-0800690565c0
tid: 926520978 (05/12/99_09:56:18)

Type-specific information:
--------------------------
(n/a)

Local stats for concat/concat3 since being enabled or reset:
------------------------------------------------------------------------
read requests:          10
write requests:         40
512 byte blks read:     640
512 byte blks written:  14080

reads straddling slices:       0
writes straddling slices:       0

Pieces:
#    Size     Timestamp         Type/Name            State
-------------------------------------------------------------
0     2222178 05/12/99_06:48:56 slice/maules0        valid,online
1     2222178 05/12/99_06:48:56 slice/maules1        valid,online
2     2222178 05/12/99_06:51:01 concat/concat0       valid,online
3     2222178 05/12/99_06:48:56 slice/maules3        valid,online
4     2222178 05/12/99_06:48:56 slice/maules4        valid,online
```

ミラーの統計

ミラーの統計では、ミラーに対する読取りおよび書込み要求のほかに、ミラー同期の読取りおよび書込みも表示されます。

以下に、統計がオンになっているミラーに対して show コマンドを実行した結果の例を示します。

```
xvm:cluster> show -v mirror/mirror0
XVM ve mirror/mirror0
=========================
volname: vol/vol3  subvolume: subvol/vol3/data
size: 11110890  iou: 1  pieces: 1  open: no
state: 0xa (valid,online) user-flags: online,autoname
uuid: 39798d05-9c4f-1022-8544-0800690565c0
tid: 926509854 (05/12/99_06:50:54)

Type-specific information:
---------------------------
rpolicy: 0 (round-robin)  config: 0x0 (none)
drl flush frequency (sec): (n/a) primary piece: 0 reviving: no

Local stats for mirror/mirror0 since being enabled or reset:
-------------------------------------------------------------------------
read requests:          10
write requests:         40
512 byte blks read:     640
512 byte blks written:  14080
Mirror synchronization reads:         0
Mirror synchronization writes:        0

Leg             Reads           Writes
0               10              40

Pieces:
#    Size     Timestamp         Type/Name            State
-----------------------------------------------------------
0    11110890 05/12/99_09:56:18 concat/concat3       valid,online
```

スライスの統計

スライスの統計では、読取り操作と書込み操作の数に加え、512K ブロック読取りと書込みの数も表示されます。

以下に、統計がオンになっているスライスに対して show コマンドを実行した結果の例を示します。

```
xvm:cluster> show -v slice/cxfsdsk1s0
XVM ve slice/cxfsdsk1s0
========================
volname: vol/cxfsvol  subvolume: subvol/cxfsvol/data
size: 17779016  iou: 1  pieces: 0  open: no
state: 0xa (valid,online) user-flags: online,autoname
uuid: 31350c53-7aa4-1022-85f1-0800690592c9
tid: 922806049 (03/30/99_09:00:49)

Type-specific information:
---------------------------
physvol: cxfsdsk1
start: 0  length: 17779016

Local stats for slice/cxfsdsk1s0 since being enabled or reset:
------------------------------------------------------------------------
read requests:          956
write requests:         0
512 byte blks read:     7684
512 byte blks written:  0

Pieces:
#    Size     Timestamp         Type/Name            State
-------------------------------------------------------------
(Ve has no pieces)
```

XVM ボリューム・マネージャの運用

この章では、XVM ボリューム・マネージャの運用方法のさまざまな側面について説明します。この章は、以下の節で構成されています。

- 「クラスタ・システムのスタートアップ」
- 「ミラーの再生」（158 ページ）
- 「クラスタの回復時のミラーの再生」（159 ページ）
- 「XVM の調整可能パラメータ」（159 ページ）
- 「XVM サブシステム・パラメータ」（160 ページ）

クラスタ・システムのスタートアップ

XVM 論理ボリュームが含まれるクラスタ・システムを起動すると、以下の操作が実行されます。

1. システムが起動して、すべてのディスク（SGI SAN ディスク、FC ハブ・ディスク、内部 SCSI など）を検査します。
2. XVM システム・ディスクから起動した場合は、XVM ボリューム・マネージャがすべての XVM ラベルを読取って、すべてのボリュームのローカル表示が作成されます。この時点ではクラスタ・ボリュームは表示されません。
3. `rc` スクリプトによって、PRISA などのサード・パーティ製 SAN デバイスが初期化されます。
4. XVM システム・ディスクから起動していない場合は、`rc` によってすべてのラベルの読取りが開始され、すべてのボリュームのローカル表示が作成されます。この時点ではクラスタ・ボリュームは表示されません。
5. クラスタが初期化されます。
6. クラスタ内の各ノードで、XVM ボリューム・マネージャがディスクのすべてのラベルを読取り、サード・パーティ製 SAN ボリュームを含むすべてのボリュームのクラスタ全体の表示を作成します。

この手順は、XVM ボリュームはクラスタが初期化されるまで表示されないことを意味している点に注意してください（つまり、シングルユーザ・モードではボリュームを使用できません）。XVM の初期化順序のため、root デバイスをサード・パーティ製 SAN ディスクに配置することはできません。

ミラーの再生

ミラーの再生とは、ミラーのメンバー上のデータを同期するプロセスのことです。ミラーの再生は、以下の場合に開始されます。

- 複数の断片を持つミラーを初めて構築したとき
- 断片をミラーに接続したとき
- 同期されていないミラーを使ってシステムが起動されたとき
- ミラーが開いている場合にクラスタ内のノードがクラッシュしたとき。この状況についての詳細は、159 ページの「XVM の調整可能パラメータ」を参照してください。

ミラーの再生が開始されると、SYSLOG にメッセージが書込まれます。このプロセスが完了すると、別のメッセージが SYSLOG に書込まれます。何らかの理由で再生が失敗した場合は、SYSLOG のほかにシステム・コンソールにもメッセージが書込まれます。

大容量のミラー・コンポーネントの場合は、生成プロセスに時間がかかることがあります。ミラーの再生は、いったん開始したら、ミラーの断片の 1 つを除いたすべてを分離する方法以外では中止できません。

作成時やシステムの再起動時に再生の必要がないミラーもあります。このようなミラーの作成についての詳細は、40 ページの「-clean ミラー作成オプション」および 40 ページの「-norevive ミラー作成オプション」を参照してください。

ミラーの再生中も、そのミラーが含まれる XVM 論理ボリュームを設定したり、ミラーに対して I/O を実行できます。ミラー・ボリューム要素を表示すると、同期が完了したミラー・ブロックの割合が表示されます。

以前に開始されたミラーの再生がまだ実行中のときにミラーの再生が必要になった場合は、ミラーの再生をキューに入れることができます。ミラーのトポロジーを表示すると、これがミラーの状態として表示されます。

XVM の調整可能パラメータを使用して、ミラーの再生のシステム・パフォーマンスを変更できます。ミラーの再生を制御する XVM の調整可能パラメータについての詳細は、159 ページの「XVM の調整可能パラメータ」を参照してください。

クラスタの回復時のミラーの再生

クラスタ内のノードがクラッシュした場合、ノード内のミラーの再生が開始されることがあります。これが実行されるのは、クラッシュしたノードがミラーを使用しており、ミラーの断片が不均等な不正な状態のままになっている可能性があるときです。この場合、XVM ボリューム・マネージャは、すべての断片を強制的に再同期する必要があります。

ミラーの使用中にノードがクラッシュした場合は、完全なミラー再同期が実行されます。これには、多少時間がかかる場合があります。

XVM の調整可能パラメータ

ミラーの再生のシステム・パフォーマンスが低いと思われる場合は、ミラー再生リソースの再設定が必要になることがあります。ミラー再生リソースは、`var/sysgen/mtune/xvm` ファイルの XVM 調整可能パラメータで設定します。

ミラー再生リソース・パラメータは動的変数です。これらのパラメータを変更すると、システムを再起動しなくても、`systune`(1M) を実行すると変更が反映されます。

XLV ミラーと XVM ミラーを同一のシステムで共有している場合は、作業に使用できるスレッドの数を減らすことが推奨されています（この数は、`xvm_max_revive_threads` パラメータで制御します)。これにより、XLV ボリューム・マネージャの多くのリソースが XVM ボリューム・マネージャによって使用されてしまうのを防止します。より多くの再生を並行して実行したい場合は、スレッドの数を増やすことができます。

スレッドの数を増やしたり、再生プロセスに使用される並行 I/O プロセスの数を減らすことができます（この数は、`xvm_max_revive_rsc` パラメータで制御します)。リソースを減らすと、開いているファイルシステムへの影響は抑えられますが、データの再生にかかる時間の合計が増えます。

以下に、一般的なガイドラインを示します。

- できるだけ迅速な再生が必要で、通常の I/O プロセスのパフォーマンスが影響を受けてもかまわない場合は、スレッドとリソースを増やします。
- XLV ボリューム・マネージャと XVM ボリューム・マネージャが同じシステムを共有している場合は、スレッドを減らします。
- 特定のファイルシステムへの影響を抑えたい場合は、リソースを減らします。

XVM サブシステム・パラメータ

XVM サブシステムは、現在実行中の XVM カーネルのさまざまな側面を反映したサブシステム・パラメータのセットを保持しています。以下に、これらのパラメータを示します。

apivers　カーネルと互換性があるライブラリのバージョン

config gen　サブシステム情報の最後のチェック以降に XVM 設定が変更されているかどうかを示すマーカー

privileged　現在呼出されている CLI に特権が付与されていて、設定を変更できるかどうかを示します（特権がない場合は、表示だけが許可されます）。

clustered　カーネルがクラスタを認識できるかどうかを示します。

cluster initialized
　クラスタ・サービスが初期化されているかどうかを示します。

以下の例のように、これらのパラメータのステータスは、show コマンドの -subsystem オプションを使用して表示できます。

```
xvm:local> show -subsystem
XVM Subsystem Information:
--------------------------
apivers:              19
config gen:           15
privileged:           1
clustered:            0
cluster initialized:  0
```

XVM 論理ボリュームと XLV 論理ボリューム

この章には、以下のトピックに関する節が含まれています。

- 「XVM 論理ボリュームと XLV 論理ボリュームの作成の比較」
- 「XLV 論理ボリュームから XVM 論理ボリュームへのアップグレード」（163 ページ）
- 「XLV ミラー化ストライプの XVM ストライプ・ミラーへの切替え」（164 ページ）

XVM 論理ボリュームと XLV 論理ボリュームの作成の比較

表 A-1 に、単純な XVM 論理ボリュームと XLV 論理ボリュームの作成の概要を示します。この例は、95 ページの「3 方向ストライプを使った論理ボリュームの作成」で挙げた例と同じです。XVM 論理ボリュームの作成手順の順を追った説明については、該当する節を参照してください。

この例で説明するコマンドでは、データが 3 つのディスクにストライプされる `stripedvol` という論理ボリュームを作成します。これらのどちらの例も、ディスクがすでに IRIX option ディスクとしてパーティション設定されていることを想定しています。なお、この表ではコマンドが左右の欄に並べて記載されていますが、必ずしも同じアクションに対応しているわけではありません。

表 A-1 **XVM** 論理ボリュームと XLV 論理ボリュームの作成

XVM 論理ボリューム	XLV 論理ボリューム
ディスクに XVM 物理ボリュームとしてラベルを付けます。 xvm:cluster> **label -name disk0 \ dks2d70vol** xvm:cluster> **label -name disk1 \ dks2d71vol** xvm:cluster> **label -name disk2 \ dks2d72vol**	論理ボリュームを作成します。 xlv_make> **vol stripedvol**
	data サブボリュームを作成します。 xlv_make> **data**
各ディスクから XVM スライスを作成します。 xvm:cluster> **slice -all disk***	プレックスを作成します。 xlv_make> **plex**
ストライプ・ボリューム要素を作成して、ボリューム stripedvol に接続します（この間に data サブボリュームが作成されます）。 xvm: cluster> **stripe -volname \ stripedvol slice/disk0s0 \ slice/disk1s0 slice/disk2s0**	ストライプ・ボリューム要素を作成して、3 つのディスクにストライプします。 xlv_make> **ve -stripe dks0d2s7 \ dks0d3s7 dks5d4s7**
XVM ボリューム・マネージャを終了します。 xvm:cluster> **quit**	XLV ボリューム・マネージャを終了します。xlv_make -A オプションが使用されていないので、xlv_assemble が自動的に呼出され、実行中のカーネルにボリュームが設定されます。 xlv_make> **end** xlv_make> **exit**
ファイルシステムに対して mkfs コマンドを実行します。 # **mkfs /dev/cxvm/stripedvol**	ファイルシステムに対して mkfs コマンドを実行します。 # **mkfs /dev/xlv/stripedvol**
ファイルシステムをマウントします。CXFS クラスタの共有ファイルシステムの場合は、CXFS GUI または CXFS CLI を使用してマウントします。ローカル・ファイルシステムの場合は、mount コマンドを使用できます。	ファイルシステムをマウントします。 # **mount /dev/xlv/stripedvol /mnt**

XLV 論理ボリュームから XVM 論理ボリュームへのアップグレード

XLV 論理ボリューム設定を XVM 論理ボリューム設定に切替えるには、以下の手順に従います。

1. 古い XLV 設定とディスク・パーティションを保存します。

   ```
   # xlv_mgr -c "script -write xlvconfig all"
   # prtvtoc /dev/rdsk/*vh > diskparts
   ```

2. 既存のファイルシステムをバックアップします。

3. 切替える XLV ボリュームを削除します。

   すべてのボリュームを削除するには、すべての XLV ファイルシステムをアンマウントしてから、以下のコマンドを実行します。

   ```
   # xlv_shutdown
   # xlv_mgr -x -c "delete all_labels"
   ```

   XVM ボリュームは root ボリュームとして使用できないので、XLV root ボリュームを使用している場合は、すべての XLV ボリュームを削除しないでください。

4. XLV ディスクに XVM ディスクとしてラベルを付けて、管理する XVM に移動します。

   ```
   # xvm label -name xvmdisk1 dks0d1vh
   ```

5. XLV で使用されていた古いパーティションに合わせてスライスを作成します。

   XVM の `-start` 引数については、`prtvtoc` パーティションの開始ブロックからボリューム・ヘッダのサイズを引いて求めます。この場合、スライス 7 がブロック 4096 から始まっていて、ボリューム・ヘッダに隣接しているときは、`-start 0` を使用します。

   `-length` 引数には、`prvtoc` パーティションのサイズを指定します。

   ```
   # xvm slice -start 0 -length 17779016 xvmdisk1
   ```

6. スライスの上位に連結ボリュームまたはストライプ・ボリュームを作成します。できるだけ `/etc/fstab` の内容を変更せずに済むように、以前の XLV ボリューム名に合わせてボリュームに名前を付けます。

7. ファイルシステムをマウントし、ファイルシステムに対して `xfs_check` を実行します。ファイルシステムをマウントできない場合、または `xfs_check` で問題が発生する場合は、以下のチェックを行ってください。

   - 各ディスクに作成されたスライスを、保存されている `prtvtoc` の情報と比較します。
   - ストライプおよび連結ボリュームの構成を XLV の情報と比較します。

XLV 設定の復元が必要になった場合は、以下の手順を使用できます。ただし、XLV で複製できない (たとえば XLV の連結ボリューム要素はストライプできません) 新しい設定をまだ XVM ボリューム・マネージャで作成していない場合にかぎります。

1. ファイルシステムをアンマウントします。

   ```
   # umount -a
   ```

2. すべての物理ボリュームから XVM ボリューム・ヘッダを削除して、元のパーティションを復元します。

   ```
   # xvm unlabel -force phys/*
   ```

3. XLV 設定を再ビルドします。

   ```
   # xlv_make xlvconfig
   ```

XLV ミラー化ストライプの XVM ストライプ・ミラーへの切替え

XLV 論理ボリュームのミラー設定方法では、2 つのディスクの異常がミラー全体の異常につながる可能性が高くなります。これは、XLV 論理ボリュームにはミラー化ストライプを含めることはできますが、ストライプ・ミラーを含めることはできないためです。これに対して、XVM ボリューム・マネージャではストライプ・ミラーを使用できます。

以下の手順に従って、XLV ミラー化ストライプを XVM ストライプ・ミラーに安全に切替えることができます。この手順では、論理ボリュームの名前は `alpha` で、ディスク名は `dks18*` および `dks5*` です。

1. 切替えるファイルシステムをアンマウントします。
2. 後で `alpha` の XLV 設定を再作成する場合に備えて、この設定を再生成するためのスクリプトを作成しておきます。

   ```
   # xlv_mgr -c "script -write alpha object alpha"
   ```

3. プレックス 0 を XLV ボリュームから分離します。

   ```
   # xlv_mgr -c "detach plex alpha.data.0 alphaplex"
   ```

4. このオブジェクトを削除します。ディスクにその他の XLV ボリュームが存在する場合は、それらのボリュームの切替えも必要なので、ボリュームを再生成するためのスクリプトを作成します。

   ```
   # xlv_mgr -c "delete object alphaplex"
   ```

5. XVM ボリューム・マネージャを呼出して、plex0 ディスクにラベルを付けます。

   ```
   xvm:local> label dks18*
   ```

6. この例では、ファイルシステムは各ディスクに 1 つのスライスを持っています。これらの各ディスクに、ディスク全体で構成される XVM スライスを作成します。

```
xvm:local> slice -all dks18*
```

7. XLV 設定に合わせてストライプを作成します。

```
xvm:local> stripe -volname alpha slice/dks18*
```

8. XVM ボリューム・マネージャを終了して、ファイルシステムが正しくマウントされることを確認します。

```
# mount /dev/lxvm/alpha /mountpoint
```

9. これで、XVM ボリューム・マネージャを通じてデータを利用できるようになりましたが、データはミラー化されていません。ボリューム alpha のトポロジーに注意してください。

```
xvm:local> show -t alpha
vol/alpha                                0 online
    subvol/alpha/data                106664448 online,open
        stripe/stripe0               106664448 online,tempname,open
            slice/dks18d1s0           17777424 online,open
            slice/dks18d2s0           17777424 online,open
            slice/dks18d3s0           17777424 online,open
            slice/dks18d4s0           17777424 online,open
            slice/dks18d5s0           17777424 online,open
            slice/dks18d6s0           17777424 online,open
```

ボリューム alpha の各スライスの上位にミラーを挿入します。

```
xvm:local> insert mirror slice/dks18*
```

この結果、以下のような設定になります。

```
xvm:local> show -t alpha
vol/alpha                                0 online
    subvol/alpha/data                106664448 online,open
        stripe/stripe0               106664448 online,tempname,open
            mirror/mirror0            17777424 online,tempname
                slice/dks18d1s0           17777424 online,open
            mirror/mirror1           17777424 online,tempname
                slice/dks18d2s0           17777424 online,open
            mirror/mirror2            17777424 online,tempname
                slice/dks18d3s0           17777424 online,open
            mirror/mirror3              17777424 online,tempname
                slice/dks18d4s0           17777424 online,open
            mirror/mirror4            17777424 online,tempname
                slice/dks18d5s0           17777424 online,open
            mirror/mirror5            17777424 online,tempname
                slice/dks18d6s0           17777424 online,open
```

10. XLV プレックスのもう一方の半分を XVM ボリュームに切替えて、ミラーに接続します。

```
# xlv_mgr -c "delete object alpha"
```

```
xvm:local> label dks5*
xvm:local> slice -all dks5*
xvm:local> attach slice/dks5d1s0 mirror0
xvm:local> attach slice/dks5d1s1 mirror1
xvm:local> attach slice/dks5d1s2 mirror2
xvm:local> attach slice/dks5d1s3 mirror3
xvm:local> attach slice/dks5d1s4 mirror4
xvm:local> attach slice/dks5d1s5 mirror5
```

また、attach コマンドには以下の構文も使用できます。

```
xvm:local> attach slice/dks5d1s0 stripe0/0
xvm:local> attach slice/dks5d1s1 stripe0/1
xvm:local> attach slice/dks5d1s2 stripe0/2
xvm:local> attach slice/dks5d1s3 stripe0/3
xvm:local> attach slice/dks5d1s4 stripe0/4
xvm:local> attach slice/dks5d1s5 stripe0/5
```

ボリューム alpha のトポロジーを表示します。

```
xvm:local> show -t alpha
vol/alpha                0 online
  subvol/alpha/data  106664448 online,open
    stripe/stripe0  106664448 online,tempname,open
       mirror/mirror0  17777424 online,tempname,reviving:28%,open
           slice/dks18d1s0  17777424 online,open
           slice/dks5d1s0   17777424 online,open
       mirror/mirror1  17777424 online,tempname,reviving:queued,open
           slice/dks18d2s0  17777424 online,open
           slice/dks5d2s0   17777424 online,open
       mirror/mirror2  17777424 online,tempname,reviving:queued,open
           slice/dks18d3s0  17777424 online,open
           slice/dks5d3s0   17777424 online,open
       mirror/mirror3  17777424 online,tempname,reviving:queued,open
           slice/dks18d4s0  17777424 online,open
           slice/dks5d4s0   17777424 online,open
       mirror/mirror4  17777424 online,tempname,reviving:queued,open
           slice/dks18d5s0  17777424 online,open
           slice/dks5d5s0   17777424 online,open
       mirror/mirror5  17777424 online,tempname,reviving:queued,open
           slice/dks18d6s0  17777424 online,open
           slice/dks5d6s0   17777424 online,open
```

11. /etc/fstab ファイルのエントリを /dev/xlv/alpha から /dev/lxvm/alpha に更新します。

索引

A

apivers サブシステム・パラメータ　160
attach コマンド　35, 41, 73

C

change コマンド　32, 36, 64, 72
  stat オプション　151
clean
  mirror コマンドのオプション　40
  状態　43
cluster initialized サブシステム・パラメータ　160
clustered パラメータ　160
collapse コマンド　44, 76
concat コマンド　37, 68
config gen サブシステム・パラメータ　160
CXFS ファイルシステム　23

D

data サブボリューム
  定義　16
delete コマンド　44, 78
detach コマンド　36, 41, 74
/dev/cxvm　4, 56
/dev/lxvm　4, 56
/dev/rcxvm　4, 56
/dev/rlxvm　4, 56
/dev/rxvm　56
/dev/xvm　56
「disabled」状態　43
dump コマンド　31, 44, 65, 78

F

failover.conf ファイル　24
-force オプション
  delete コマンド　44
  unlabel コマンド　32
  一般　58
  分離　36
fx コマンド　9

G

give コマンド　65

H

help コマンド　49
hinv コマンド　9

I

「incomplete」状態　43
「inconsistent」状態　42
insert コマンド　44, 75

L

label コマンド　29, 60, 99
log サブボリューム
　定義　16

M

「mediaerr」状態　42
miniroot のインストール　90
mirror コマンド　38, 69
　-clean オプション　40
　-norevive オプション　40
mtune パラメータ　159

N

-name オプション、label コマンド　60
-nopartchk オプション、label コマンド　61
norevive
　mirror コマンドのオプション　40

O

「offline」状態　42, 43
　論理ボリューム　44
「online」状態　42, 43
「open」状態　43
option ディスク　6

P

physvol オブジェクト・タイプ
　定義　53
　パス名　53
physvol。物理ボリュームを参照
「pieceoffline」状態　43
privileged サブシステム・パラメータ　160
probe コマンド　64

R

real-time サブボリューム
　定義　16
remake コマンド　41, 74
「reviving」状態　43
root スライス　85
root ディスク。システム・ディスクを参照
root 特権　49
root パーティション　80

S

-safe オプション
　一般　35, 58
　分離　36
SAN ディスクのパス　53
set コマンド　60
show コマンド　42, 62, 77, 99, 160
slice
　システム　85
slice コマンド　37, 68, 99
slice コマンド　85
steal コマンド　63, 66
stripe コマンド　37, 70, 100
-subsystem オプション、show コマンド　160
subvolume コマンド　41, 71, 72
swap スライス　85
swap パーティション　80

T

「tempname」状態　42

U

unlabel コマンド 32, 67
usr スライス 85
usr パーティション 82
usr ファイルシステム 5

V

-volhdrdblks オプション、label コマンド 61
volume コマンド 40

X

XLV
  XVM へのアップグレード 163
  論理ボリューム 2
XLV から XVM へのアップグレード 163
XVM オブジェクト
  正規表現 55
  タイプ 52
  名前 51
xvm コマンド
  キーワード 50
  構文 50
  出力 55
  省略形 50
  ヘルプ 49
  リダイレクト 55
XVM システム・ディスク 7, 79-92
  アップグレード 88
  インストール 88
  起動 91
  削除 91
  作成 80
  複雑な論理ボリューム 127
  ミラー化 86, 121
  ラベルを付ける 121
XVM システム・ディスクからの起動 91
XVM システム・ディスクのインストール 88
XVM システム・ディスクへのアップグレード 88
XVM トポロジー
  作成 33
  表示 33
XVM ボリューム
  制限 35
XVM 論理ボリューム
  再生成 44
  制限 10, 13
  定義 13
  保存 44
  論理レベル 10
xvm_max_revive_rsc パラメータ 159
xvm_max_revive_threads パラメータ 159
-xvmlabelblks オプション、label コマンド 61

あ

安全なコマンド 58

お

オブジェクト名
  正規表現 55
オブジェクト・タイプ 53

か

外部ディスク 26, 53, 63

き

キーワード
  xvm コマンド 50
キャラクタ特殊ファイル 4, 56

く

クラスタ・ドメイン　26, 47, 60

こ

子ボリューム要素
　定義　10

さ

サブボリューム
　data　16, 41
　log　16, 41
　real-time　16, 41
　作成　41
　制限　13
　タイプ　41
　定義　16
　統計　152
　名前　51
　ユーザ定義　16, 41

し

シーケンシャル・ミラー読取りポリシー　39
システム・ディスク　7, 79-92
　アップグレード　88
　インストール　88
　起動　91
　削除　91
　作成　80
　復元　92
　複雑な論理ボリューム　127
　ミラー化　86, 112, 121
　ラベルを付ける　121
状態
　clean　43
　disabled　43
　incomplete　43
　inconsistent　42
　mediaerr　42
　offline　42
　online　42
　open　43
　pieceoffline　43
　reviving　43
　tempname　42
　ボリューム要素　42

す

ストライプ
　サイズ　70
　作成　37
　単位　70
　定義　19
　統計　152
スライス
　作成　37
　定義　17
　統計　156

せ

正規表現、オブジェクト名　55

た

断片
　定義　10

ち

調整可能パラメータ　159

て

ディスクの検査。probe コマンドを参照
デバイスのホット・プラグ　4

と

統計　36, 151
　サブボリューム　152
　ストライプ　152
　スライス　156
　物理ボリューム　152
　ミラー　155
　連結　154
トポロジー。XVM トポロジーを参照
ドメイン
　クラスタ　23, 26, 60
　ローカル　23, 26, 60

は

パーティション
　root　80
　swap　80
　usr　82
パーティション・レイアウト　5-9
パスの指定　53

ふ

フェイルオーバー　24
物理ボリューム
　管理　30
　交換　31
　作成　29
　実行中のシステムへの追加　31
　定義　26
　統計　152
　破棄　32
　表示　31
プライマリ・レッグ、ミラー　40
ブロック特殊ファイル　4, 56

ほ

ホット・プラグ　4
ボリューム
　作成　40
　定義　13
ボリューム名、一時　33
ボリューム要素
　オンライン化　44
　空のボリューム要素　36
　作成　33
　自動作成　33
　制限　10
　接続　35
　断片を使った構文　53
　定義　10, 26
　表示　42
　分離　36
　命名　34
ボリューム・ヘッダ　13

み

ミラー
　再生　35, 38, 69, 74, 158, 159
　再生リソース　159
　削除　147
　作成　38
　作成時に同期しない　40
　ストライプのミラー化　148
　接続　35
　定義　21
　統計　155
　プライマリ・レッグ　40
　分離　36, 74
　読取りポリシー　39

め

命名
  ボリューム　33
  ボリューム要素　34

ゆ

ユーザ定義サブボリューム　16

ら

ラウンドロビン・ミラー読取りポリシー　39
ラベルの付いていないディスク　53
  定義　25

れ

連結
  作成　37
  定義　17
  統計　154

ろ

ローカル・ドメイン　26, 48, 60
論理ボリューム
  「offline」状態　43, 44
  「online」状態　43
  オンラインでの変更　139
  拡張　141
  管理　42
  再構成　41
  作成　139
  作成例　98
  データの書込み　22
  破棄　44
  表示　42
  ミラー化　143
  論理ボリュームの拡張　141

わ

ワイルドカード、正規表現　55