



SUSE Linux Enterprise Server 15 **ストレージ管理ガイド**

ストレージ管理ガイド

SUSE Linux Enterprise Server 15

SUSE Linux Enterprise Serverサーバでストレージデバイスを管理する方法を説明します。

発行日: 2024 年 9 月 29 日

<https://documentation.suse.com> 

Copyright © 2006–2024 SUSE LLC and contributors. All rights reserved.

この文書は、GNUフリー文書ライセンスのバージョン1.2または(オプションとして)バージョン1.3の条項に従って、複製、頒布、および/または改変が許可されています。ただし、この著作権表示およびライセンスは変更せずに記載すること。ライセンスバージョン1.2のコピーは、「GNUフリー文書ライセンス」セクションに含まれています。

SUSEの商標については、<http://www.suse.com/company/legal/> を参照してください。その他の製品名および会社名は、各社の商標または登録商標です。商標記号(®、™など)は、SUSEおよび関連会社の商標を示します。アスタリスク(*)は、第三者の商標を示します。

本書のすべての情報は、細心の注意を払って編集されています。しかし、このことは絶対に正確であることを保証するものではありません。SUSE LLC、その関係者、著者、翻訳者のいずれも誤りまたはその結果に対して一切責任を負いかねます。

目次

このガイドについて xii

- 1 利用可能なマニュアル xii
- 2 フィードバック xiii
- 3 マニュアルの表記規則 xiv

I ファイルシステムとマウント 1

1 Linuxファイルシステムの概要 2

- 1.1 用語集 3
- 1.2 Btrfs 3
 - 主な機能 3 • SUSE Linux Enterprise Server上のルートファイルシステム設定 4 • ReiserFSおよびExtの各ファイルシステムからBtrfsへのマイグレーション 9 • Btrfsの管理 10 • サブボリュームに対するBtrfsクォータのサポート 11 • Btrfs send/receive 12 • データ重複排除のサポート 15
- 1.3 XFS 16
 - アロケーショングループを使用した高スケーラビリティ 16 • ディスクスペースの効率的な管理によるハイパフォーマンス 17 • 事前割り当てによるファイルシステムの断片化の回避 17
- 1.4 Ext2 18
- 1.5 Ext3 19
 - Ext2からの容易で信頼性の高いアップグレード 19 • 信頼性とパフォーマンス 19 • Ext2ファイルシステムからExt3への変換 20 • Ext3ファイルシステムのinodeサイズとinode数 20
- 1.6 Ext4 24
- 1.7 ReiserFS 25
- 1.8 サポートされている他のファイルシステム 25

- 1.9 Linux環境での大規模ファイルサポート 26
- 1.10 Linuxのカーネルにおけるストレージの制限 28
- 1.11 ファイルシステムのトラブルシューティング 28
 - Btrfsエラー: デバイスに空き領域がない 29 • 未使用のファイルシステムブロックの解放 30
- 1.12 追加情報 31

2 ファイルシステムのサイズ変更 32

- 2.1 使用例 32
- 2.2 サイズ変更のガイドライン 32
 - サイズ変更をサポートしているファイルシステム 33 • ファイルシステムのサイズの増加 33 • ファイルシステムのサイズの削減 34
- 2.3 Btrfsファイルシステムのサイズの変更 34
- 2.4 XFSファイルシステムのサイズの変更 35
- 2.5 Ext2、Ext3、またはExt4の各ファイルシステムのサイズの変更 36

3 UUIDによるデバイスのマウント 38

- 3.1 udevによる永続的なデバイス名 38
- 3.2 UUIDの理解 38
- 3.3 追加情報 39

4 ブロックデバイス操作の多層キャッシング 40

- 4.1 一般的な用語 40
- 4.2 キャッシングモード 41
- 4.3 bcache 42
 - 主な特徴 42 • bcacheデバイスのセットアップ 43 • sysfsを使用したbcacheの設定 44
- 4.4 lvmcache 44
 - lvmcacheの設定 45 • キャッシュプールの削除 46

II 論理ボリューム(LVM) 48

5 LVMの設定 49

- 5.1 論理ボリュームマネージャ(LVM)の理解 49
- 5.2 ボリュームグループの作成 51
- 5.3 論理ボリュームの作成 54
 - シンプロビジョニング論理ボリューム 57 • ミラーリングされたボリュームの作成 58
- 5.4 非ルートLVMボリュームグループの自動アクティブ化 60
- 5.5 既存のボリュームグループのサイズ変更 61
- 5.6 論理ボリュームのサイズ変更 62
- 5.7 ボリュームグループまたは論理ボリュームの削除 64
- 5.8 LVMコマンドの使用 65
 - コマンドによる論理ボリュームのサイズ変更 68 • `lvmetad`によるLVMメタデータの動的集約 71 • LVMキャッシュボリュームの使用 71
- 5.9 LVM2ストレージオブジェクトへのタグ付け 72
 - LVM2タグの使用 72 • LVM2タグの作成要件 73 • コマンドラインでのタグ構文 74 • 設定ファイル構文 74 • クラスタで簡単なアクティベーション制御にタグを使用する 76 • タグを使用して、クラスタ内の好みのホストでアクティブにする 76

6 LVMボリュームスナップショット 80

- 6.1 ボリュームスナップショットの理解 80
- 6.2 LVMによるLinuxスナップショットの作成 82
- 6.3 スナップショットの監視 82
- 6.4 Linuxスナップショットの削除 83
- 6.5 仮想ホスト上の仮想マシンに対するスナップショットの使用 83
- 6.6 スナップショットをソース論理ボリュームとマージして変更を元に戻すか、前の状態にロールバックする 85

III ソフトウェアRAID 88

7 ソフトウェアRAIDの設定 89

7.1 RAIDレベルの理解 89

RAID 0 89 • RAID 1 90 • RAID 2およびRAID 3 90 • RAID 4 90 • RAID 5 90 • RAID 6 91 • ネストしたコンプレックスRAIDレベル 91

7.2 YaSTによるソフトウェアRAID設定 92

RAIDの名前 95

7.3 ソフトウェアRAIDのトラブルシューティング 96

ディスク障害復旧後の回復 96

7.4 詳細情報 97

8 ルートパーティション用のソフトウェアRAIDの設定 98

8.1 ルートパーティション用のソフトウェアRAIDデバイスを使用するための前提条件 98

8.2 ルート(/)パーティションにソフトウェアRAIDデバイスを使用するシステムの設定 99

9 ソフトウェアRAID 10デバイスの作成 103

9.1 mdadmによるネストしたRAID 10デバイスの作成 103

mdadmによるネストしたRAID 10 (1+0)デバイスの作成 104 • mdadmによるネストしたRAID 10 (0+1)デバイスの作成 106

9.2 コンプレックスRAID 10の作成 108

コンプレックスRAID 10のデバイスおよびレプリカの数 109 • レイアウト 110 • YaSTパーティショナによるコンプレックスRAID 10の作成 112 • mdadmによるコンプレックスRAID 10の作成 115

10 ディグレードアレイの作成 118

11 mdadmによるソフトウェアRAIDアレイのサイズ変更 120

- 11.1 ソフトウェアRAIDのサイズの増加 121
コンポーネントパーティションのサイズの増加 122 • RAIDアレイのサイズの増加 123 • ファイルシステムのサイズの増加 124
- 11.2 ソフトウェアRAIDのサイズの削減 125
ファイルシステムのサイズの削減 125 • RAIDアレイサイズの削減 125 • コンポーネントパーティションのサイズの削減 127

12 MDソフトウェアRAID用のストレージエンクロージャLEDユーティリティ 129

- 12.1 ストレージエンクロージャLED監視サービス 130
- 12.2 ストレージエンクロージャLED制御アプリケーション 131
パターン名 132 • デバイスのリスト 135 • 例 136
- 12.3 追加情報 136

IV ネットワークストレージ 138

13 Linux用iSNS 139

- 13.1 iSNSのしくみ 139
- 13.2 Linux用iSNSサーバのインストール 141
- 13.3 iSNS検出ドメインの設定 143
iSNS検出ドメインの作成 143 • iSCSIノードの検出ドメインへの追加 144
- 13.4 iSNSサービスの開始 146
- 13.5 その他の情報 146

14 IPネットワークの大容量記憶域 - iSCSI 147

- 14.1 iSCSI LIOターゲットサーバとiSCSIイニシエータのインストール 148

- 14.2 iSCSI LIOターゲットサーバのセットアップ 149
iSCSI LIOターゲットサービスの起動およびファイアウォールの設定 149 • iSCSI LIOターゲットおよびイニシエータのディスカバリに対する認証の設定 151 • ストレージスペースの準備 152 • iSCSI LIOターゲットグループの設定 153 • iSCSI LIOターゲットグループの変更 157 • iSCSI LIOターゲットグループの削除 158
- 14.3 iSCSIイニシエータの設定 159
YaSTを使ったiSCSIイニシエータの設定 159 • 手動によるiSCSIイニシエータの設定 162 • iSCSIイニシエータデータベース 163
- 14.4 インストール時のiSCSIディスクの使用 165
- 14.5 iSCSIのトラブルシューティング 165
iSCSI LIOターゲットサーバにターゲットLUNをセットアップする際のポータルエラー 165 • iSCSI LIOターゲットが他のコンピュータで表示されない 166 • iSCSIトラフィックのデータパッケージがドロップされる 166 • LVMでiSCSIボリュームを使用する 166 • 設定ファイルが手動に設定されていると、iSCSIターゲットがマウントされる 167
- 14.6 iSCSI LIOターゲットの用語 167
- 14.7 追加情報 169

15 Fibre Channel Storage over Ethernet Networks: FCoE 171

- 15.1 インストール時におけるFCoEインタフェースの設定 172
- 15.2 FCoEおよびYaSTのFCoEクライアントのインストール 173
- 15.3 YaSTを使用したFCoEサービスの管理 174
- 15.4 コマンドを使用したFCoEの設定 177
- 15.5 FCoE管理ツールを使用したFCoEインスタンスの管理 179
- 15.6 追加情報 181

16 NVMe over Fabric 182

- 16.1 概要 182

- 16.2 NVMe over Fabricホストの設定 182
コマンドラインクライアントのインストール 182 • NVMe over Fabricターゲットの検出 183 • NVMe over Fabricターゲットへの接続 183 • マルチパス処理 184
- 16.3 NVMe over Fabricターゲットの設定 184
コマンドラインクライアントのインストール 184 • 設定手順 184 • ターゲット設定のバックアップと復元 186
- 16.4 特定のハードウェアの設定 187
概要 187 • Broadcom 187
- 16.5 詳細情報 187
- 17 デバイスのマルチパスI/Oの管理 189**
- 17.1 マルチパスI/Oの理解 189
- 17.2 ハードウェアサポート 189
マルチパス処理用に自動検出されるストレージアレイ 189 • マルチパス処理サポートについてテスト済みのストレージアレイ 192 • 特定のハードウェアハンドラを必要とするストレージアレイ 192
- 17.3 マルチパス処理のプランニング 193
前提条件 193 • ディスク管理タスク 194 • ソフトウェアRAID 194 • 高可用性ソリューション 195 • initrdとシステム設定との同期を常に維持する 195
- 17.4 マルチパス管理ツール 195
デバイスマッパーマルチパスモジュール 196 • マルチパスI/O管理ツール 198 • マルチパスデバイスへのMDADMの使用 199 • multipathコマンド 199 • mpathpersistユーティリティ 202
- 17.5 マルチパス処理用システムの設定 203
マルチパスI/Oサービスの有効化、無効化、起動、および停止 203 • マルチパス処理用SANデバイスの準備 204 • マルチパスデバイスのパーティショニング 205
- 17.6 /etc/multipath.conf Fileの作成または修正 206
/etc/multipath.confファイルの作成 206 • /etc/multipath.confファイルのセクション 207 • etc/multipath.confファイルでのマルチパスセットアップ

- ブの確認 208 • /etc/multipath.confファイルの変更を適用したマルチパス
マップの更新 210 • WWIDの生成 211
- 17.7 ポーリング、待ち行列、およびフェールバック用のデフォルトポリ
シーの設定 211
- 17.8 非マルチパスデバイスのブラックリスト化 213
- 17.9 ユーザフレンドリ名または別名の設定 216
HAクラスタにおけるマルチパスデバイスの名前 221
- 17.10 パスフェールオーバーのポリシーと優先度の設定 222
パスのフェールオーバーポリシーの設定 222 • フェールオーバーポリシー
の設定 223 • ターゲットパスグループの報告 231
- 17.11 ルートデバイスのマルチパスI/Oの設定 231
インストール時にマルチパスI/Oを有効にする 231 • 既存ルートデバイス
用マルチパスI/Oの有効化 234 • ルートデバイスのマルチパスI/Oの無効
化 234
- 17.12 既存ソフトウェアRAID用マルチパスI/Oの設定 235
- 17.13 マルチパスデバイスでのLVM2の使用 237
- 17.14 ベストプラクティス 238
新規デバイスのスキャン(再起動なし) 238 • パーティショニングされ
た新規デバイスのスキャン(再起動なし) 239 • マルチパスI/Oステータ
スの表示 242 • エラーになったI/Oの管理 243 • 停止したI/Oの解
決 244 • IBM Zデバイスのデフォルト設定 245 • NetAppデバイス
でのマルチパスの使用 245 • マルチパスデバイスでの--noflushの使
用 246 • ルートデバイスがマルチパスの場合のSANタイムアウト設定 246
- 17.15 MPIOのトラブルシューティング 247
マルチパスデバイスへのGRUB2のインストール 247 • マルチパスが有効
な場合、ブート時にシステムが終了して緊急シェルが起動する 247 • マ
ルチパス0.4.9以降への更新後に、個別デバイスのprio設定が失敗す
る 250 • multipath-tools-0.4.9以降への更新後に、引数を伴うprio設定が失
敗する 250 • 技術情報ドキュメント 251
- 18 NFSv4上でのアクセス制御リストの管理 252**
- A GNU利用許諾契約書 253**

このガイドについて

このガイドでは、SUSE Linux Enterprise Server 15 でストレージデバイスを管理する方法を説明します。デバイスのパーティショニングと管理については、『導入ガイド』、第10章「Expert Partitioner (エキスパートパーティショナ)」を参照してください。このガイドは、システム管理者を対象としています。

1 利用可能なマニュアル



注記: オンラインヘルプと最新のアップデート

製品に関するマニュアルは、<http://www.suse.com/documentation/> からご利用いただけます。最新のアップデートもご利用いただけるほか、マニュアルをさまざまな形式でブラウズおよびダウンロードすることができます。

また、製品マニュアルは通常、`/usr/share/doc/manual` の下にあるインストール済みシステムから入手できます。

この製品の次のマニュアルを入手できます。

項目 「インストールクイックスタート」

このクイックスタートでは、SUSE® Linux Enterprise Server 15のインストールを順を追って説明します。

『導入ガイド』

単一または複数のシステムをインストールする方法および展開インフラストラクチャに製品本来の機能を活用する方法を示します。ローカルインストールまたはネットワークインストールサーバの使用から、リモート制御の高度にカスタマイズされた自動リモートインストール技術による大規模展開まで、多様なアプローチから選択できます。

『管理ガイド』

当初のインストールシステムの保守、監視、およびカスタマイズなど、システム管理タスクについて説明します。

『Virtualization Guide』

仮想化技術全般について説明し、仮想化統合インタフェースであるlibvirt、および特定のハイパーバイザの詳細情報を紹介します。

『ストレージ管理ガイド』

SUSE Linux Enterprise Serverサーバでストレージデバイスを管理する方法を説明します。

『AutoYaST Guide』

AutoYaSTは、インストールデータと設定データが指定されたAutoYaSTプロファイルを使用してSUSE Linux Enterprise Serverの各システムを無人で大規模展開するシステムです。マニュアルに従って、自動インストールの基本的な手順(準備、インストール、および設定)を実行できます。

『Security and Hardening Guide』

システムセキュリティの基本概念を紹介し、ローカルセキュリティ/ネットワークセキュリティの両方の側面を説明します。AppArmorなど製品に付属するセキュリティソフトウェアや、セキュリティ関連イベントの情報を確実に収集する監査システムの使用方法を説明します。

『System Analysis and Tuning Guide』

問題の検出、解決、および最適化に関する管理者ガイド。ツールの監視によってシステムを検査および最適化する方法およびリソースを効率的に管理する方法を見つけることができます。よくある問題と解決、および追加のヘルプとドキュメントリソースの概要も含まれています。

『Repository Mirroring Tool Guide』

リポジトリ管理ツール(SUSEカスタマーセンターの代理システムで、リポジトリと登録ターゲットが含まれる)の管理者ガイド。ローカルのRMTサーバのインストールと設定、リポジトリのミラーリングと管理、クライアントマシンの管理、およびRMTを使用するクライアントの設定について説明します。

『GNOMEユーザガイド』

SUSE Linux Enterprise ServerのGNOMEデスクトップについて紹介します。デスクトップの使用および設定方法と、キータスクの実行方法を説明します。主として、デフォルトのデスクトップとしてGNOMEを効率的に使用したいと考えるエンドユーザ向けです。

2 フィードバック

次のフィードバックチャンネルがあります。

バグと機能拡張の要求

ご使用の製品に利用できるサービスとサポートのオプションについては、<http://www.suse.com/support/>を参照してください。

openSUSEのヘルプはコミュニティによって提供されています。詳細については、<https://en.opensuse.org/Portal:Support> を参照してください。
製品コンポーネントのバグを報告するには、<https://scc.suse.com/support/requests> にアクセスしてログインし、Create New (新規作成) をクリックします。

ユーザからのコメント

本マニュアルおよびこの製品に含まれているその他のマニュアルについて、皆様のご意見やご要望をお寄せください。オンラインドキュメントの各ページの下にあるユーザコメント機能を使用するか、または<http://www.suse.com/documentation/feedback.html> にアクセスして、コメントを入力してください。

メール

この製品のドキュメントについてのフィードバックは、doc-team@suse.com 宛のメールでも送信できます。ドキュメントのタイトル、製品のバージョン、およびドキュメントの発行日を明記してください。エラーの報告または機能拡張の提案では、問題について簡潔に説明し、対応するセクション番号とページ(またはURL)をお知らせください。

3 マニュアルの表記規則

このマニュアルでは、次の通知と表記規則が使用されています。

- `/etc/passwd`: ディレクトリ名とファイル名
- `PLACEHOLDER: PLACEHOLDER` は、実際の値で置き換えられます
- `PATH`: 環境変数PATH
- `ls, --help`: コマンド、オプション、およびパラメータ
- `user`: ユーザまたはグループ
- `package name`: パッケージの名前
- `Alt`, `Alt - F1`: 使用するキーまたはキーの組み合わせ、キーはキーボード上と同様、大文字で表示される
- ファイル、ファイル > 名前を付けて保存: メニュー項目、ボタン
- `AMD/Intel` この説明は、AMD64/Intel 64アーキテクチャにのみ当てはまります。矢印は、テキストブロックの先頭と終わりを示します。◁
- `IBM Z, POWER` この説明は、IBM ZおよびPOWERの各アーキテクチャにのみ当てはまります。矢印は、テキストブロックの先頭と終わりを示します。◁

- Dancing Penguins (「Penguins」の章、↑他のマニュアル):他のマニュアルの章への参照です。
- root 特権で実行する必要があるコマンド。多くの場合、これらのコマンドの先頭に sudo コマンドを置いて、特権のないユーザとしてコマンドを実行することもできます。

```
root # command  
tux > sudo command
```

- 特権のないユーザでも実行できるコマンド。

```
tux > command
```

- 通知



警告: 警告の通知

続行する前に知っておくべき、無視できない情報。セキュリティ上の問題、データ損失の可能性、ハードウェアの損傷、または物理的な危険について警告します。



重要: 重要な通知

続行する前に知っておくべき重要な情報。



注記: メモの通知

追加情報。たとえば、ソフトウェアバージョンの違いに関する情報です。



ヒント: ヒントの通知

ガイドラインや実際的なアドバイスなどの役に立つ情報。

I ファイルシステムとマウント

- 1 Linuxファイルシステムの概要 2
- 2 ファイルシステムのサイズ変更 32
- 3 UUIDによるデバイスのマウント 38
- 4 ブロックデバイス操作の多層キャッシング 40

1 Linuxファイルシステムの概要

SUSE Linux Enterprise Serverにはいくつかの異なるファイルシステム (Btrfs、Ext4、Ext3、Ext2、XFSなど)が付属しており、そのいずれかを選択することができます。各ファイルシステムには、それぞれ独自の利点と欠点があります。SUSE Linux Enterprise Serverにおける主要オペレーティングシステムの機能の対照比較については、<http://www.suse.com/products/server/technical-information/#FileSystem> (「ファイルシステムのサポートとサイズ」)を参照してください。この章では、それらのファイルシステムの機能および利点の概要を説明します。

SUSE Linux Enterprise 12では、オペレーティングシステム用のデフォルトファイルシステムはBtrfsであり、他はすべてXFSがデフォルトです。また、Extファイルシステムファミリ、およびOCFS2も引き続きサポートします。デフォルトでは、Btrfsファイルシステムは複数のサブボリュームと共に設定されます。ルートファイルシステムでは、Snapperインフラストラクチャを使用して、スナップショットが自動的に有効になります。Snapperの詳細については、『管理ガイド』、第7章「Snapperを使用したシステムの回復とスナップショット管理」を参照してください。

プロ級のハイパフォーマンスのセットアップには、可用性の高いストレージシステムが必要なことがあります。ハイパフォーマンスのクラスタリングシナリオの要件を満たすため、SUSE Linux Enterprise Serverでは、High Availability ExtensionアドオンにOCFS2 (Oracle Cluster File System 2)とDRBD (Distributed Replicated Block Device)を組み込んでいます。これらの高度なストレージシステムは、本書では扱いません。詳細については、『SUSE Linux Enterprise High Availability Extension Administration Guide<http://www.suse.com/doc>』を参照してください。

ただし、すべてのアプリケーションに最適なファイルシステムは存在しません。各ファイルシステムには特定の利点と欠点があり、それらを考慮する必要があります。最も高度なファイルシステムを選択する場合でも、適切なバックアップ戦略が必要です。

本項で使用するデータの完全性およびデータの一貫性という用語は、ユーザスペースデータ(ユーザが使用するアプリケーションによりファイルに書き込まれるデータ)の一貫性を指す言葉ではありません。ユーザスペースのデータが一貫しているかどうかは、アプリケーション自体が管理する必要があります。

本項で特に指定のない限り、パーティションおよびファイルシステムの設定または変更に必要なすべての手順は、YaSTパーティショナを使用して実行できます(そうすることをお勧めします)。詳細については、『導入ガイド』、第10章「Expert Partitioner (エキスパートパーティショナ)」を参照してください。

1.1 用語集

メタデータ(metadata)

ファイルシステムが内包するデータ構造です。これにより、すべてのオンディスクデータが正しく構成され、アクセス可能になります。です。ほとんどすべてのファイルシステムに独自のメタデータ構造があり、それが各ファイルシステムに異なるパフォーマンス特性が存在する理由の1つになっています。メタデータが破損しないよう維持するのは、非常に重要なことです。もし破損した場合、ファイルシステム内にあるすべてのデータがアクセス不能になる可能性があるからです。

inode

サイズ、リンク数、ファイルの内容を実際に格納しているディスクブロックへのポインタ、作成日時、変更日時、アクセス日時など、ファイルに関する各種の情報を含むファイルシステムのデータ構造。

ジャーナル(journal)

ファイルシステムのジャーナルは、ファイルシステムがそのメタデータ内で行う変更を特定のログに記録するオンディスク構造です。ジャーナル機能は、システム起動時にファイルシステム全体をチェックする長時間の検索プロセスが不要なため、ファイルシステムの回復時間を大幅に短縮します。ただし、それはジャーナルが再現できる場合に限定されます。

1.2 Btrfs

Btrfsは、Chris Masonが開発したCOW(コピーオンライト)ファイルシステムです。このシステムは、Ohad Rodehが開発したCOWフレンドリなBツリーに基づいています。Btrfsは、ログインスタイルのファイルシステムです。このシステムでは、ブロックの変更をジャーナリングする代わりに、それらの変更を新しい場所書き込んで、リンクインします。新しい変更は、最後の書き込みまで確定されません。

1.2.1 主な機能

Btrfsは、次のような耐障害性、修復、容易な管理機能を提供します。

- 書き込み可能なスナップショット。更新適用後に必要に応じてシステムを容易にロールバックしたり、ファイルをバックアップできます。
- サブボリュームのサポート: Btrfsでは、割り当てられたスペースのプールにデフォルトのサブボリュームが作成されます。Btrfsでは、同じスペースプール内で個々のファイルシステムとして機能する追加サブボリュームを作成できます。サブボリュームの数は、プールに割り当てられたスペースによってのみ制限されます。
- **scrub**を使用したオンラインでのチェックと修復の機能が、Btrfsのコマンドラインツールの一部として利用できます。ツリー構造が正しいことを前提として、データとメタデータの完全性を検証します。マウントしたファイルシステム上で、scrubを定期的に実行することができます。これは、通常の操作中にバックグラウンドプロセスとして実行されます。
- メタデータとユーザデータ用のさまざまなRAIDレベル。
- メタデータとユーザデータ用のさまざまなチェックサム。エラー検出が向上します。
- Linux LVM (Logical Volume Manager)ストレージオブジェクトとの統合。
- SUSE Linux Enterprise Server上でのYaSTパーティションおよびAutoYaSTとの統合。その際、MD (複数デバイス)およびDM (デバイスマッパー)の各ストレージ設定ではBtrfsファイルシステムの作成も行われます。
- 既存のExt2、Ext3、およびExt4ファイルシステムからの、オフラインのマイグレーション。
- **/boot** のブートローダサポート。Btrfsパーティションからの起動を可能にします。
- マルチボリュームBtrfsは、SUSE Linux Enterprise Server 15 では、RAID0、RAID1、およびRAID10プロファイルでサポートされます。それより高いレベルのRAIDは現時点サポートされませんが、将来のサービスパックでサポートされる可能性があります。
- Btrfsのコマンドを使用して、透過圧縮を設定します。

1.2.2 SUSE Linux Enterprise Server上のルートファイルシステム設定

SUSE Linux Enterprise Serverのルートパーティションは、デフォルトでBtrfsとスナップショットを使用して設定されます。スナップショットを使用すると、更新適用後に必要に応じてシステムを容易にロールバックしたり、ファイルをバックアップしたりできます。スナップショットは、『管理ガイド』、第7章「Snapperを使用したシステムの回復とスナップ

ショット管理」で説明するSUSE Snapperインフラストラクチャを使用して簡単に管理できます。SUSEのSnapperプロジェクトの一般情報については、OpenSUSE.orgにあるSnapper Portal wiki (<http://snapper.io>)を参照してください。

スナップショットを使用してシステムをロールバックする場合、ユーザのホームディレクトリ、WebサーバとFTPサーバのコンテンツ、ログファイルなどのデータがロールバック中に失われたり、上書きされたりしないようにする必要があります。それには、ルートファイルシステムでBtrfsサブボリュームを使用します。サブボリュームは、スナップショットから除外できます。インストール時にYaSTによって提示されるSUSE Linux Enterprise Serverのルートファイルシステムのデフォルト設定には、次のサブボリュームが含まれます。これらがスナップショットから除外される理由を次に示します。

/boot/grub2/i386-pc、/boot/grub2/x86_64-efi、/boot/grub2/powerpc-ieee1275、/boot/grub2/s390x-emu

ブートローダ設定のロールバックはサポートされていません。これらのディレクトリは、アーキテクチャ固有です。最初の2つのディレクトリはAMD64/Intel 64マシン上に存在し、その後の2つのディレクトリはそれぞれIBM POWERとIBM Z上に存在します。

/home

/homeが独立したパーティションに存在していない場合、ロールバック時にデータが失われるのを避けるために除外されます。

/opt、/var/opt

サードパーティ製品は通常、/optにインストールされます。ロールバック時にこれらのアプリケーションがアンインストールされるのを避けるために除外されます。

/srv

WebおよびFTPサーバ用のデータが含まれています。ロールバック時にデータが失われるのを避けるために除外されます。

/tmp、/var/tmp、/var/cache、/var/crash

スナップショットから除外される一時ファイルとキャッシュを含むすべてのディレクトリ。

/usr/local

このディレクトリは、ソフトウェアの手動インストール時に使用します。ロールバック時にこれらのインストール済みソフトウェアがアンインストールされるのを避けるために除外されます。

/var/lib/libvirt/images

libvirtで管理される仮想マシンイメージのデフォルトの場所。ロールバック時に仮想マシンイメージが古いバージョンに置き換えられないようにするために除外されます。デフォルトでは、このサブボリュームは、no copy on write オプションを使用して作成されます。

/var/lib/mailman、/var/spool

電子メールまたは電子メールキューを含むディレクトリは、ロールバック後に電子メールが失われるのを避けるために除外されます。

/var/lib/named

DNSサーバ用のゾーンデータが含まれます。ネームサーバがロールバック後に確実に動作できるように、スナップショットから除外されます。

/var/lib/mariadb、/var/lib/mysql、/var/lib/pgsql

これらのディレクトリにはデータベースのデータが格納されます。デフォルトでは、これらのサブボリュームは、no copy on write オプションを使用して作成されます。

/var/log

ログファイルの場所。壊れたシステムのロールバック後にログファイルを分析できるようにスナップショットから除外されます。



警告: ロールバックのサポート

SUSEサポートがロールバックをサポートするのは、事前設定されているサブボリュームがまったく削除されていない場合のみです。ただし、YaSTパーティションを使用して、サブボリュームを追加することはできます。

1.2.2.1 圧縮されたBtrfsファイルシステムのマウント



注記: GRUB 2およびLZO圧縮ルート

GRUB 2は、LZO圧縮ルートを読み込むことができません。圧縮を使用するには、別の /boot パーティションが必要です。

SLE12 SP1から、Btrfsファイルシステムの圧縮がサポートされるようになりました。 compress または compress-force オプションを使用し、圧縮アルゴリズム(lzo または zlib)を選択します(zlibがデフォルト値です)。zlib圧縮は、より圧縮率が高く、一方lzo圧縮はより高速でCPU負荷が低くなります。

次に例を示します。

```
root # mount -o compress /dev/sdx /mnt
```

ファイルを作成し、そのファイルに書き込む場合で、圧縮された結果のサイズが未圧縮サイズよりも大きい場合、Btrfsはこのファイルに以後も書き込みができるように圧縮をスキップします。この動作が必要ない場合、`compress-force` オプションを使用します。最初の圧縮できないデータを含むファイルには有効です。

圧縮は、新規ファイルのみに効果があることに注意してください。圧縮なしで書き込まれたファイルは、ファイルシステムが `compress` オプションまたは `compress-force` オプションを使用してマウントされたときに圧縮されません。また、`nodatacow` 属性を持つファイルのエクステンツは圧縮されません。

```
root # chattr +C FILE
root # mount -o nodatacow /dev/sdx /mnt
```

暗号化は、圧縮処理とは関係のない独立した処理です。このパーティションにデータを書き込んだら、詳細を印刷してください。

```
root # btrfs filesystem show /mnt
btrfs filesystem show /mnt
Label: 'Test-Btrfs'  uuid: 62f0c378-e93e-4aa1-9532-93c6b780749d
    Total devices 1 FS bytes used 3.22MiB
    devid    1 size 2.00GiB used 240.62MiB path /dev/sdb1
```

永続的に設定したい場合、`compress` オプションまたは `compress-force` オプションを `/etc/fstab` 設定ファイルに追加します。次に例を示します。

```
UUID=1a2b3c4d /home btrfs subvol=@/home,compress 0 0
```

1.2.2.2 サブボリュームのマウント

SUSE Linux Enterprise Server上のスナップショットからシステムをロールバックするには、まずスナップショットからブートします。これにより、ロールバックを実行する前に、スナップショットを実行しながらチェックできます。スナップショットからブートできるようにするには、サブボリュームをマウントします(通常は不要な操作です)。

1.2.2項「SUSE Linux Enterprise Server上のルートファイルシステム設定」の一覧に示されているサブボリューム以外に、`@`という名前のボリュームが存在します。これは、ルートパーティション(`/`)としてマウントされるデフォルトサブボリュームです。それ以外のサブボリュームは、このボリュームにマウントされます。

スナップショットからブートすると、@サブボリュームではなく、スナップショットが使用されます。スナップショットに含まれるファイルシステムの部分は、/として読み込み専用でマウントされます。それ以外のサブボリュームは、スナップショットに書き込み可能でマウントされます。この状態は、デフォルトでは一時的なものです。次の再起動により、前の設定が復元されます。これを永久的なものにするには、**snapper rollback** コマンドを実行します。これにより、今回のブートに使用したスナップショットが新しいデフォルトのサブボリュームになり、再起動後はこのサブボリュームが使用されます。

1.2.2.3 空き領域の確認

通常、ファイルシステムの使用量は **df** コマンドで確認します。Btrfsファイルシステムでは、**df** の出力は誤解を招く可能性があります。生データが割り当てる領域とは別に、Btrfsファイルシステムもメタデータ用の領域を割り当てて使用するからです。

その結果、まだ大量の領域を使用できるように見えても、Btrfsファイルシステムによって領域不足がレポートされることがあります。その場合、メタデータ用に割り当てられた領域はすべて使用されています。Btrfsファイルシステム上の使用済みの領域と使用可能な領域を確認するには、次のコマンドを使用します。

btrfs filesystem show

```
tux > sudo btrfs filesystem show /
Label: 'ROOT'  uuid: 52011c5e-5711-42d8-8c50-718a005ec4b3
    Total devices 1 FS bytes used 10.02GiB
    devid    1 size 20.02GiB used 13.78GiB path /dev/sda3
```

ファイルシステムの合計サイズとその使用量を表示します。最後の行のこれら2つの値が一致する場合、ファイルシステム上の領域はすべて割り当て済みです。

btrfs filesystem df

```
tux > sudo btrfs filesystem df /
Data, single: total=13.00GiB, used=9.61GiB
System, single: total=32.00MiB, used=16.00KiB
Metadata, single: total=768.00MiB, used=421.36MiB
GlobalReserve, single: total=144.00MiB, used=0.00B
```

ファイルシステムの割り当て済みの領域(**total**)および使用済みの領域の値を表示します。メタデータの **total** および **used** の値がほぼ等しい場合、メタデータ用の領域はすべて割り当て済みです。

btrfs filesystem usage

```
tux > sudo btrfs filesystem usage /
```

```
Overall:
  Device size:                20.02GiB
  Device allocated:           13.78GiB
  Device unallocated:         6.24GiB
  Device missing:             0.00B
  Used:                       10.02GiB
  Free (estimated):           9.63GiB   (min: 9.63GiB)
  Data ratio:                 1.00
  Metadata ratio:             1.00
  Global reserve:             144.00MiB   (used: 0.00B)
```

		Data single	Metadata single	System single	Unallocated
Id	Path				
1	/dev/sda3	13.00GiB	768.00MiB	32.00MiB	6.24GiB
Total		13.00GiB	768.00MiB	32.00MiB	6.24GiB
Used		9.61GiB	421.36MiB	16.00KiB	

前の2つのコマンドを組み合わせたのと同様のデータを表示します。

詳細については、[man 8 btrfs-filesystem](https://btrfs.wiki.kernel.org/index.php/FAQ) および <https://btrfs.wiki.kernel.org/index.php/FAQ> を参照してください。

1.2.3 ReiserFSおよびExtの各ファイルシステムからBtrfsへのマイグレーション

btrfs-convert ツールを使用して、既存のReiserFSまたはExt (Ext2、Ext3、またはExt4) からBtrfsファイルシステムにデータボリュームをマイグレートすることができます。これにより、アンマウントされた(オフライン)ファイルシステムのインプレース変換を実行できます。これには**btrfs-convert** ツールとともにブート可能なインストールメディアが必要な場合があります。このツールは元のファイルシステムの空き領域内にBtrfsファイルシステムを構築し、それに含まれているデータに直接リンクします。メタデータを作成するにはデバイスに十分な空き領域が必要です。さもないと変換に失敗します。元のファイルシステムはそのままとなり、Btrfsファイルシステムによって空き領域が占有されることはありません。必要なスペースの量はファイルシステムのコンテンツによって決まりますが、そこに含まれるファイルシステムオブジェクト(ファイル、ディレクトリ、拡張属性)の数によって左右される場合があります。データは直接参照されるため、ファイルシステム上のデータ量は変換に必要なスペースに影響を与えません。ただし、テールパッキングを使用するファイルや2KiBを超えるサイズのファイルは除きます。

元のファイルシステムをBtrfsファイルシステムに変換するには、次のコマンドを実行します。


```
root # btrfs-convert /path/to/device
```

❗ 重要: /etc/fstabの確認

変換後は、`/etc/fstab`に記載されている元のファイルシステムへのすべての参照で、デバイスにBtrfsファイルシステムがあることが示されるように調整されていることを確認する必要があります。

変換時には、Btrfsファイルシステムのコンテンツにソースファイルシステムのコンテンツが反映されます。ソースファイルシステムは、`fs_root/reiserfs_saved/image`で作成された関連する読み込み専用イメージを削除するまで保持されます。イメージファイルの実態は、変換前におけるReiserFSファイルシステムの「スナップショット」であり、Btrfsファイルシステムが変更されても変わりません。イメージファイルを削除するには、`reiserfs_saved`サブボリュームを削除します。

```
root # btrfs subvolume delete fs_root/reiserfs_saved>
```

ファイルシステムを元に戻すには、次のコマンドを使用します。

```
root # btrfs-convert -r /path/to/device
```

🚫 警告: 失われる変更

Btrfsファイルシステムとしてマウントされているファイルシステムへの変更はすべて失われます。マウント中には負荷分散操作を実行しないでください。さもないと、ファイルシステムが正しく復元されなくなります。

1.2.4 Btrfsの管理

Btrfsは、YaSTパーティショナおよびAutoYaST内に統合されています。これはインストール時に利用可能で、ルートファイルシステム用のソリューションを設定することができます。インストール後に、YaSTパーティショナを使用して、Btrfsのボリュームの参照と管理を行うことができます。

Btrfsの管理ツールは、`btrfsprogs`パッケージ内に用意されています。Btrfsコマンドの使用については、`man 8 btrfs`、`man 8 btrfsck`、および`man 8 mkfs.btrfs`の各コマンドを参照してください。Btrfsの機能については、Btrfs wiki (<http://btrfs.wiki.kernel.org>)を参照してください。

1.2.5 サブボリュームに対するBtrfsクォータのサポート

Btrfs rootファイルシステムのサブボリューム `/var/log`、`/var/crash` および `/var/cache` が、通常の操作時に利用可能なディスクスペースのすべてを使用でき、システムに不具合が発生します。この状況を回避するため、SUSE Linux Enterprise Serverではサブボリュームに対するBtrfsクォータのサポートを提供するようになりました。YaSTからの提案に従ってルートファイルシステムを設定する場合、ルートファイルシステムは必要に応じて準備されます。サブボリュームはすべて、クォータグループ(`qgroup`)を設定済みです。ルートファイルシステムのサブボリュームにクォータを設定するには、次の手順に従います。

1. クォータサポートを有効にします。

```
tux > sudo btrfs quota enable /
```

2. サブボリュームのリストを取得します。

```
tux > sudo btrfs subvolume list /
```

クォータは既存のサブボリュームにのみ設定できます。

3. 前の手順で表示されたサブボリュームの1つにクォータを設定します。サブボリュームは、パス(`/var/tmp` など)または `0/SUBVOLUME ID` (`0/272` など)のどちらかによって識別できます。次に、`/var/tmp` に5GBのクォータを設定する例を示します。

```
tux > sudo btrfs qgroup limit 5G /var/tmp
```

サイズは、バイト(5000000000)、キロバイト(5000000K)、メガバイト(5000M)、またはギガバイト(5G)のいずれかの単位で指定できます。結果として得られるサイズは多少異なります。これは、1024バイト=1KiB、1024KiB=1MiBなどだからです。

4. 既存のクォータを一覧にするには、次のコマンドを使用します。`max_rfer` 列に、クォータがバイト単位で表示されます。

```
tux > sudo btrfs qgroup show -r /
```



ヒント: クォータの無効化

既存のクォータを無効にする場合、クォータサイズを `none` に設定します。

```
tux > sudo btrfs qgroup limit none /var/tmp
```

特定のパーティションとそのすべてのサブボリュームのクォータサポートを無効にするには、`btrfs quota disable` を使用します。

```
tux > sudo btrfs quota disable /
```

詳細については、[man 8 btrfs-qgroup](#) および [man 8 btrfs-quota](#) を参照してください。Btrfs wiki (<https://btrfs.wiki.kernel.org/index.php/UseCases>) の UseCases ページにも詳細情報が記載されています。

1.2.6 Btrfs send/receive

Btrfs では、ファイルシステムの状態をキャプチャするためのスナップショットを作成できます。Snapper では、たとえばこの機能を使用してシステムの変更前後のスナップショットを作成することで、ロールバックを可能にしています。ただし、send/receive 機能とスナップショットを併用すると、リモートの場所にファイルシステムのコピーを作成して管理することもできます。たとえば、この機能を使用してインクリメンタルバックアップを実行できます。

btrfs send 操作は、同じサブボリュームの2つの読み込み専用スナップショットの差分を計算して、それをファイルまたはSTDOUTに送信します。**Btrfs receive** 操作は、send コマンドの結果を取得して、それをスナップショットに適用します。

1.2.6.1 前提条件

Btrfs の send/receive 機能を使用するには、次の要件を満たす必要があります。

- ソース側 (`send`) とターゲット側 (`receive`) に Btrfs ファイルシステムが必要です。
- Btrfs send/receive はスナップショットを操作するため、それぞれのデータが Btrfs サブボリュームに存在する必要があります。
- ソース側のスナップショットは読み込み専用である必要があります。
- SUSE Linux Enterprise 12 SP2 以上。それより古いバージョンの SUSE Linux Enterprise は send/receive をサポートしていません。

1.2.6.2 インクリメンタルバックアップ

次の手順では、`/data` (ソース側) のインクリメンタルバックアップを `/backup/data` (ターゲット側) に作成する場合を例にして、Btrfs send/receive の基本的な使用方法を示します。`/data` はサブボリュームである必要があります。

手順 1.1: 初期セットアップ

1. ソース側に初期スナップショット(この例では `snapshot_0` という名前)を作成し、それがディスクに書き込まれていることを確認します。

```
tux > sudo btrfs subvolume snapshot -r /data /data/bkp_data
sync
```

新しいサブボリューム `/data/bkp_data` が作成されます。これは次のインクリメンタルバックアップの基として使用されるので、参照用に保持しておく必要があります。

2. 初期スナップショットをターゲット側に送信します。これは初期の `send/receive` 操作であるため、完全なスナップショットを送信する必要があります。

```
tux > sudo bash -c 'btrfs send /data/bkp_data | btrfs receive /backup'
```

ターゲット側に新しいサブボリューム `/backup/bkp_data` が作成されます。

初期セットアップが完了したら、インクリメンタルバックアップを作成して、現在のスナップショットと以前のスナップショットの差分をターゲット側に送信できます。手順は常に同じです。

1. ソース側に新しいスナップショットを作成します。
2. 差分をターゲット側に送信します。
3. オプション: 両側のスナップショットの名前変更またはクリーンアップ、あるいはその両方を行います。

手順 1.2: インクリメンタルバックアップの実行

1. ソース側に新しいスナップショットを作成し、それがディスクに書き込まれていることを確認します。次の例では、スナップショットに `bkp_data_CURRENT_DATE` という名前が付いています。

```
tux > sudo btrfs subvolume snapshot -r /data /data/bkp_data_$(date +%F)
sync
```

新しいサブボリューム(たとえば、`/data/bkp_data_2016-07-07`)が作成されます。

2. 以前のスナップショットと新たに作成したスナップショットの差分をターゲット側に送信します。そのためには、オプション `-p SNAPSHOT` を使用して、以前のスナップショットを指定します。

```
tux > sudo bash -c 'btrfs send -p /data/bkp_data /data/bkp_data_2016-07-07 \
```

```
| btrfs receive /backup'
```

新しいサブボリューム /backup/bkp_data_2016-07-07 が作成されます。

3. その結果、それぞれの側に2つずつ、合計4つのスナップショットが存在することになります。

/data/bkp_data
/data/bkp_data_2016-07-07
/backup/bkp_data
/backup/bkp_data_2016-07-07

続行するには、次の3つのオプションがあります。

- 両方の側のすべてのスナップショットを保持する。このオプションの場合、両方の側のどのスナップショットにもロールバックすることが可能であると同時に、すべてのデータの複製を保持していることになります。これ以上のアクションは必要ありません。次のインクリメンタルバックアップを実行するときには、最後から2番目のスナップショットをsend操作の親として使用することに注意してください。
- ソース側には最新のスナップショットのみを保持し、ターゲット側にはすべてのスナップショットを保持する。この場合も、両方の側のどのスナップショットにもロールバックできます。ソース側で特定のスナップショットへのロールバックを実行するには、ターゲット側からソース側に、完全なスナップショットのsend/receive操作を実行します。ソース側で削除/移動操作を実行します。
- 両方の側に最新のスナップショットのみを保持する。この方法では、ソース側で作成された最新のスナップショットと同じ状態のバックアップがターゲット側にあります。ほかのスナップショットにロールバックすることはできません。ソース側とターゲット側で削除/移動操作を実行します。

- a. ソース側に最新のスナップショットのみを保持するには、次のコマンドを実行します。

```
tux > sudo btrfs subvolume delete /data/bkp_data  
tux > sudo mv /data/bkp_data_2016-07-07 /data/bkp_data
```

最初のコマンドで以前のスナップショットを削除し、2番目のコマンドで現在のスナップショットの名前を /data/bkp_data に変更します。これにより、バックアップされた最新のスナップショットは常に /data/bkp_data という名前になります。その結果、常にこのサブボリューム名をインクリメンタルsend操作の親として使用できます。

- b. ターゲット側に最新のスナップショットのみを保持するには、次のコマンドを実行します。

```
tux > sudo btrfs subvolume delete /backup/bkp_data  
tux > sudo mv /backup/bkp_data_2016-07-07 /backup/bkp_data
```

最初のコマンドで以前のバックアップスナップショットを削除し、2番目のコマンドで現在のスナップショットの名前を `/backup/bkp_data` に変更します。これにより、最新のバックアップスナップショットは常に `/backup/bkp_data` という名前になります。



ヒント: リモートターゲット側への送信

スナップショットをリモートマシンに送信するには、SSHを使用します。

```
tux > btrfs send /data/bkp_data | ssh root@jupiter.example.com 'btrfs receive /  
backup'
```

1.2.7 データ重複排除のサポート

Btrfsはデータ重複排除をサポートします。そのための方法として、ファイルシステム内の複数の同一ブロックを、共通ストレージロケーションにある、そのブロックの1つのコピーを指す論理リンクで置き換えます。SUSE Linux Enterprise Serverでは、ファイルシステムをスキャンして同一ブロックをチェックする **dupremove** ツールを提供しています。このツールをBtrfsファイルシステムで使用した場合、該当するブロックを重複排除することもできます。**dupremove** はデフォルトではインストールされません。これを使用できるようにするには、パッケージ `dupremove` をインストールします。



注記: 使用例

SUSE Linux Enterprise Server 15の時点では、`dupremove` はファイルシステム全体の重複排除という用途には適していません。このツールは、仮想マシンイメージなど、大量のブロックが共通する可能性がある10～50個の大容量ファイルの重複排除に使用することを想定しています。

dupremove は、ファイルのリストを処理することも、ディレクトリを再帰的にスキャンすることもできます。

```
tux > sudo duperemove OPTIONS file1 file2 file3
tux > sudo duperemove -r OPTIONS directory
```

動作モードには、読み込み専用と重複排除の2つがあります。読み込み専用モードで実行した場合(-d スイッチを指定しない)、指定されたファイルまたはディレクトリをスキャンして重複ブロックをチェックし、出力します。これは、どのファイルシステムでも機能します。

重複排除モードでの **duperemove** の実行は、Btrfsファイルシステムでのみサポートされています。指定されたファイルまたはディレクトリをスキャンした後、重複しているブロックは重複排除用に送信されます。

詳細については、[**man 8 duperemove**](#) を参照してください。

1.3 XFS

本来は、IRIX OS用のファイルシステムを意図してSGIがXFSの開発を開始したのは、1990年代初期です。XFSの開発動機は、ハイパフォーマンスの64ビットジャーナルファイルシステムの作成により、非常に厳しいコンピューティングの課題に対応することでした。XFSは大規模なファイル进行操作する点で非常に優れていて、ハイエンドのハードウェアを適切に活用します。XFSは、SUSE Linux Enterprise Serverのデータパーティション用のデフォルトファイルシステムです。

ただし、XFSの主要機能を一見すれば、XFSが、ハイエンドコンピューティングの分野で、他のジャーナリングファイルシステムの強力な競合相手となっている理由がわかります。

1.3.1 アロケーショングループを使用した高スケーラビリティ

XFSファイルシステムの作成時に、ファイルシステムの基にあるブロックデバイスは、等しいサイズをもつ8つ以上の線形の領域に分割されます。これらを「アロケーショングループ」と呼びます。各アロケーショングループは、独自のinodeと空きディスクスペースを管理します。実用的には、アロケーショングループを、1つのファイルシステムの中にある複数のファイルシステムと見なすこともできます。アロケーショングループは互いに独立しているものではないため、複数のアロケーショングループをカーネルから同時にアドレス指定できるという特徴があります。この機能は、XFSの高いスケーラビリティに大きく貢献しています。独立性の高いアロケーショングループは、性質上、マルチプロセッサシステムのニーズに適しています。

1.3.2 ディスクスペースの効率的な管理によるハイパフォーマンス

+

空きスペースとinodeは、各アロケーショングループ内のB -Treeによって処理されます。B ツリーの採用は、XFSのパフォーマンスとスケーラビリティを大きく向上させています。XFSでは、プロセスを2分割して割り当てを処理する遅延割り当てを使用します。保留されているトランザクションはRAMの中に保存され、適切な量のスペースが確保されます。XFSは、この時点では、データを正確にはどこに(ファイルシステムのどのブロックに)格納するか決定していません。決定可能な最後の瞬間まで、この決定は遅延(先送り)されます。暫定的に使用される一時データは、ディスクに書き込まれません。XFSがデータの実際の保存場所を決定するまでに、その役割を終えているからです。このように、XFSは、書き込みのパフォーマンスを向上させ、ファイルシステムのフラグメンテーションを減少させます。遅延アロケーションは、他のファイルシステムより書き込みイベントの頻度を下げる結果をもたらすので、書き込み中にクラッシュが発生した場合、データ損失が深刻になる可能性が高くなります。

1.3.3 事前割り当てによるファイルシステムの断片化の回避

データをファイルシステムに書き込む前に、XFSはファイルが必要とする空きスペースを予約(プリアロケート、事前割り当て)します。したがって、ファイルシステムの断片化は大幅に減少します。ファイルの内容がファイルシステム全体に分散することがないので、パフォーマンスが向上します。



注記: 新しいXFSオンディスクフォーマット

SUSE Linux Enterprise Serverはバージョン12以降、XFSファイルシステムの新しい「オンディスクフォーマット」(v5)をサポートしています。YaSTによって作成されるXFSファイルシステムは、この新しいフォーマットを使用します。このフォーマットの主な利点には、全XFSメタデータの自動チェックサム、ファイルタイプのサポート、および1つのファイルに対する大量のアクセス制御リストのサポートがあります。

このフォーマットは、SUSE Linux Enterpriseカーネルの3.12より古いバージョン、xfsprogsの3.2.0より古いバージョン、およびSUSE Linux Enterprise 12より前にリリースされたバージョンのGRUB 2ではサポートされて「いません」。このことは、これらの前提条件を満たさないシステムからもこのファイルシステムを使用する必要がある場合に問題になります。

XFSファイルシステムと古いSUSEシステムまたは他のLinuxディストリビューションとの相互運用性が必要な場合は、`mkfs.xfs` コマンドを使用して手動でファイルシステムをフォーマットします。これにより、古いフォーマットでXFSファイルシステムが作成されます(`-m crc=1` オプションを使用する場合を除く)。

1.4 Ext2

Ext2の起源は、Linuxの歴史の初期にさかのぼります。その前身であったExtended File Systemは、1992年4月に実装され、Linux 0.96cに統合されました。Extended File Systemにはさまざまな変更が加えられてきました。そして、Ext2はLinuxファイルシステムとして数年にわたり非常に高い人気を得ています。その後、ジャーナルファイルシステムが作成され、回復時間が非常に短くなったため、Ext2の重要性は低下しました。

Ext2の利点に関する短い要約を読むと、かつて幅広く好まれ、そして今でも一部の分野で多くのLinuxユーザから好まれるLinuxファイルシステムである理由を理解するのに役立ちます。

堅実性と速度

「古くからある標準」であるExt2は、さまざまな改良が加えられ、入念なテストが実施されてきました。だからこそ、Ext2は非常に信頼性が高いとの評価を得ることが多いでしょう。ファイルシステムが正常にアンマウントできず、システムが機能停止した場合、`e2fsck`はファイルシステムのデータの分析を開始します。メタデータは一貫した状態に戻り、保留されていたファイルとデータブロックは、指定のディレクトリ(`lost+found`)に書き込まれます。ジャーナルファイルシステムとは対照的に、`e2fsck`は、最近変更されたわずかなメタデータだけではなく、ファイルシステム全体を分析します。この結果、ジャーナルファイルシステムがログデータだけをチェックするのに比べて、かなり長い時間を要します。ファイルシステムのサイズにもよりますが、この手順は30分またはそれ以上を要することがあります。したがって、高可用性を必要とするどのようなサーバでも、Ext2を選択することは望ましくありません。ただし、Ext2はジャーナルを維持せず、わずかなメモリを使用するだけなので、他のファイルシステムより高速なことがあります。

容易なアップグレード性

Ext3は、Ext2のコードをベースとし、Ext2のオンディスクフォーマットとメタデータフォーマットも共用するので、Ext2からExt3へのアップグレードは非常に容易です。

1.5 Ext3

Ext3は、Stephen Tweedieによって設計されました。他のすべての次世代ファイルシステムとは異なり、Ext3は完全に新しい設計理念に基づいておりではありません。Ext3は、Ext2をベースとしています。これら2つのファイルシステムは、非常に似ています。Ext3ファイルシステムを、Ext2ファイルシステムの上に構築することも容易です。Ext2とExt3の最も重要な違いは、Ext3がジャーナルをサポートしていることです。要約すると、Ext3には、次の3つの主要な利点があります。

1.5.1 Ext2からの容易で信頼性の高いアップグレード

Ext2のコードは、Ext3が次世代ファイルシステムであることを明確に主張するための強力な土台になりました。Ext3では、Ext2の信頼性および堅実性がExt3で採用されたジャーナルファイルシステムの利点とうまく統合されています。XFSのような他のジャーナリングファイルシステムへの移行はかなり手間がかかります(ファイルシステム全体のバックアップを作成し、移行先ファイルシステムを新規に作成する必要があります)が、それとは異なり、Ext3への移行は数分で完了します。ファイルシステム全体を新たに作成し直しても、それが完璧に動作するとは限らないので、Ext3への移行は非常に安全でもあります。ジャーナルファイルシステムへのアップグレードを必要とする既存のExt2システムの数を見込めると、多くのシステム管理者にとってExt3が重要な選択肢となり得る理由が容易にわかります。Ext3からExt2へのダウングレードも、アップグレードと同じほど容易です。Ext3ファイルシステムのアンマウントを正常に行い、Ext2ファイルシステムとして再マウントします。

1.5.2 信頼性とパフォーマンス

他のジャーナルファイルシステムは、「メタデータのみ」のジャーナルアプローチに従っています。つまり、メタデータは常に一貫した状態に保持されますが、ファイルシステムのデータ自体については、一貫性が自動的に保証されるわけではありません。Ext3は、メタデータとデータの両方に注意するよう設計されています。「注意」の度合いはカスタマイズできます。Ext3の `data=journal` モードを有効にした場合、最大の保護(データの完全性)を実現しますが、メタデータとデータの両方がジャーナル化されるので、システムの動作が遅くなります。比較的新しいアプローチは、`data=ordered` モードを使用することです。これは、データとメタデータ両方の完全性を保証しますが、ジャーナルを適用するのはメタデータのみです。ファイルシステムドライバは、1つのメタデータの更新に対応するすべてのデータブロックを収集します。これらのブロックは、メタデータの更新前にディスクに書き込まれます。その結果、パフォーマンスを犠牲にすることなく、メタデータとデータの両方に関する一貫性を達成できます。3番目のオプションは、`data=writeback` を使用することです。これは、対応す

るメタデータをジャーナルにコミットした後で、データをメインファイルシステムに書き込むことを可能にします。多くの場合、このオプションは、パフォーマンスの点で最善と考えられています。しかし、内部のファイルシステムの完全性が維持される一方で、クラッシュと回復を実施した後では、古いデータがファイル内に再登場させてしまう可能性があります。Ext3では、デフォルトとして、`data=ordered` オプションを使用します。

1.5.3 Ext2ファイルシステムからExt3への変換

Ext2ファイルシステムをExt3に変換するには、次の手順に従います。

1. Ext3ジャーナルの作成には、`tune2fs -j` を `root` ユーザとして実行します。
この結果、デフォルトのパラメータを使用してExt3ジャーナルが作成されます。
ジャーナルのサイズおよびジャーナルを常駐させるデバイスを指定するには、`tune2fs -J` とともに適切なジャーナルオプション `size=` および `device=` を指定して、実行します。`tune2fs` プログラムの詳細については、`tune2fs` のマニュアルページを参照してください。
2. ファイル `/etc/fstab` を `root` ユーザとして編集して、該当するパーティションに指定されているファイルシステムタイプを `ext2` から `ext3` に変更し、その変更内容を保存します。
これにより、Ext3ファイルシステムが認識されるようになります。この変更結果は、次の再起動後に有効になります。
3. Ext3パーティションとしてセットアップされたルートファイルシステムをブートするには、`ext3` と `jbd` の各モジュールを `initrd` に追加します。それには、次を実行します。
 - a. 次の行を `/etc/dracut.conf.d/01-dist.conf` に追加します。

```
force_drivers+="ext3 jbd"
```
 - b. `dracut -f` コマンドを実行します。
4. システムを再起動します。

1.5.4 Ext3ファイルシステムのinodeサイズとinode数

inodeには、ファイルシステム内のファイルとそのブロック位置に関する情報が格納されます。拡張した属性とACLのためのスペースをinode内に確保するため、Ext3のデフォルトのinodeサイズは、SLES 10での128バイトから、SLES 11では256バイトに拡大されまし

た。SLES 10と比較して、SLES 11上で新しいExt3ファイルシステムを作成する際、同数のinodeに対する事前割り当てされたデフォルトのスペースの量は2倍になり、ファイルシステム内のファイルに対して使用可能なスペースは、その分少なくなっています。したがって、同数のinodeとファイルを収容するのに、SLES 10上のExt3ファイルシステムの場合より大きなパーティションを使用する必要があります。

新規のExt3ファイルシステムを作成する際、inodeテーブル内のスペースは、作成可能なinodeの総数に対して事前に割り当てられています。バイト数/inode数の比率と、ファイルシステムのサイズによって、inode数の上限が決まります。ファイルシステムが作成されると、バイト数/inode数のバイト数の各スペースに対して、1つのinodeが作成されます。

```
number of inodes = total size of the file system divided by the number of bytes per inode
```

inodeの数によって、ファイルシステム内に保有できるファイルの数が決まります。つまり、各ファイルにつき1つのinodeです。inodeサイズの増大と、利用可能なスペースの縮小に対応するため、バイト数/inode数の比率のデフォルトが、SLES 10での8192バイトから、SLES 11では16384バイトに増えています。この2倍に増えた比率により、作成可能なファイルの数は、SLES 10上のExt3ファイルシステムで可能だった数の半分となります。

！ 重要: 既存のExt3ファイルシステムのInodeサイズの変更

inodeの割り当て後は、inodeサイズやバイト数/inode数の比率の設定を変えることはできません。異なる設定のファイルシステムを再度作成するか、ファイルシステムを拡張しない限り、新規のinodeは設定できません。inodeの最大数を超えると、ファイルをいくつか削除するまで、ファイルシステム上に新規のファイルを作成することはできません。

新規のExt3ファイルシステムを作成する際に、inodeのスペース使用をコントロールするためのinodeサイズとバイト数/inode数の比率、およびファイルシステム上のファイル数の上限を指定することができます。ブロックサイズ、inodeサイズ、およびバイト数/inode数の比率が指定されない場合は、`/etc/mke2fs.conf` ファイル内のデフォルト値が適用されます。詳細については、`mke2fs.conf(5)` マニュアルページを参照してください。

次のガイドラインを使用します。

- **inodeサイズ:** デフォルトのinodeサイズは256バイトです。2の累乗で、ブロックサイズ以下の128以上のバイト数の値を指定します(128、256、512など)。Ext3ファイルシステムで拡張属性またはACLを使用しない場合は、128バイトのみを使用してください。
- **バイト数/inode数の比率:** デフォルトのバイト数/inode数の比率は、16384バイトです。有効なバイト数/inode数の比率は、2の累乗で1024バイト以上(1024、2048、4096、8192、16384、32768など)です。この値は、ファイルシステム

のブロックサイズより小さくはできません。なぜなら、ブロックサイズは、データを格納するために使用するスペースの最小チャンクだからです。Ext3ファイルシステムのデフォルトのブロックサイズは、4 KBです。

また、格納する必要があるファイルの数とサイズを検討してください。たとえば、ファイルシステムに多数の小さなファイルを持つことになる場合は、バイト数/inode数の比率を小さめに指定すれば、inodeの数を増やすことができます。ファイルシステムに非常に大きなファイルを入れる場合は、バイト数/inode数の比率を大きめに指定できますが、それによって許容されるinodeの数は減ります。

一般的に、inodeの数は、足りなくなるよりは多すぎる方が得策です。inodeの数が少な過ぎてファイルも非常に小さい場合、実際には空であってもディスク上のファイルの最大数に到達してしまいます。inodeの数が多過ぎて、ファイルが非常に大きい場合は、空き領域があることが表示されたとしても、それを使うことができません。なぜなら、inode用に確保されたスペースに新規のファイルを作成することはできないからです。

Ext3ファイルシステムで拡張属性またはACLを使用しない場合は、ファイルシステムの作成時に、inodeサイズとして128バイト、バイト数/inode数の比率として8192バイトを指定して、SLES 10の動作を復元することができます。inodeサイズとバイト数/inode数の比率を設定するには、次のいずれかの方法を使用します。

- **すべての新規Ext3ファイルのデフォルト設定を変更する:** テキストエディタで、`/etc/mke2fs.conf` ファイルの `defaults` セクションを変更して、`inode_size` および `inode_ratio` を、希望するデフォルト値に設定します。その値が、すべての新規のExt3ファイルシステムに適用されます。たとえば、

```
blocksize = 4096
inode_size = 128
inode_ratio = 8192
```

- **コマンドラインで:** Ext3ファイルシステムを作成する際に、inodeサイズ(`-I 128`)およびバイト数/inode数の比率(`-i 8192`)を、`mkfs.ext3(8)` コマンドまたは `mke2fs(8)` コマンドに渡します。たとえば、次のコマンドのいずれかを使用します:

```
tux > sudo mkfs.ext3 -b 4096 -i 8092 -I 128 /dev/sda2
tux > sudo mke2fs -t ext3 -b 4096 -i 8192 -I 128 /dev/sda2
```

- **YaSTを使用したインストール時に:** インストール時に新規のExt3ファイルシステムを作成する際に、inodeサイズとバイト数/inode数の比率を渡します。YaSTフォーマット設定のオプションにあるパーティションの編集ページで、パーティションのフォーマット

トExt3を選択して、オプションをクリックします。ファイルシステムオプションダイアログで、ブロックサイズ(バイト単位)、inodeごとのバイト数、およびiノードのサイズドロップダウンボックスから、希望の値を選択します。

たとえば、ブロックサイズ(バイト単位)ドロップダウンボックスから4096を選択しinodeごとのバイト数ドロップダウンボックスから8192を選択し、iノードのサイズドロップダウンボックスから128を選択して、OKをクリックします。

ファイルシステムオプション:

ストライド長 (ブロック単位)(L)

ブロックサイズ (バイト単位)(S)

inode ごとのバイト数(B)

root 用に予約するブロックの割合(R)

☒ 通常のチェックを無効にする

iノードのサイズ(I)

☒ ディレクトリインデックス機能(D)

- **AutoYaSTを使用したインストール時に:** autoyastのプロファイルで、fs_options タグを使用して、opt_bytes_per_inode の比率の値を、-iに対して8192に、opt_inode_density の値を-Iに対して 128に設定することができます。

```
<partitioning config:type="list">
```

```

<drive>
  <device>/dev/sda</device>
  <initialize config:type="boolean">true</initialize>
  <partitions config:type="list">
    <partition>
      <filesystem config:type="symbol">ext3</filesystem>
      <format config:type="boolean">true</format>
      <fs_options>
        <opt_bytes_per_inode>
          <option_str>-i</option_str>
          <option_value>8192</option_value>
        </opt_bytes_per_inode>
        <opt_inode_density>
          <option_str>-I</option_str>
          <option_value>128</option_value>
        </opt_inode_density>
      </fs_options>
      <mount></mount>
      <partition_id config:type="integer">131</partition_id>
      <partition_type>primary</partition_type>
      <size>25G</size>
    </partition>
  </partitions>
</drive>
</partitioning>

```

詳細については、<http://www.suse.com/support/kb/doc.php?id=7009075> を参照してください(SLES11のExt3パーティションには、SLES10で格納できるファイルの50%しか格納することができません) [技術情報文書7009075]。

1.6 Ext4

2006年に、Ext4はExt3の後継として登場しました。最大1エクスビバイトのサイズのボリューム、最大16テビバイトのサイズのファイル、および無制限のサブディレクトリをサポートすることによって、Ext3のストレージに関する制限を解消しました。同時に、遅延ブロック割り当て、ファイルシステムチェックルーチンの大幅な高速化など、さまざまなパフォーマンス強化も図られています。また、Ext4は、ジャーナルチェックサムをサポートおよびナノ秒単位でのタイムスタンプの提供により、信頼性を高めています。Ext4には、Ext2およびExt3との完全な後方互換性があり、どちらのファイルシステムもExt4としてマウントできます。

1.7 ReiserFS

ReiserFSのサポートは、SUSE Linux Enterprise Server 15で廃止されました。既存のパーティションをBtrfsにマイグレートするには、[1.2.3項「ReiserFSおよびExtの各ファイルシステムからBtrfsへのマイグレーション」](#)を参照してください。

1.8 サポートされている他のファイルシステム

表1.1「Linux環境でのファイルシステムのタイプ」は、Linuxがサポートしている他のいくつかのファイルシステムを要約したものです。これらは主に、他の種類のメディアや外部オペレーティングシステムとの互換性およびデータの相互交換を保証することを目的としてサポートされています。

表 1.1: LINUX環境でのファイルシステムのタイプ

ファイルシステムのタイプ	説明
cramfs	Compressed ROM file system (圧縮ROMファイルシステム): ROM用の圧縮された読み込み専用ファイルシステムです。
hpfs	High Performance File System (ハイパフォーマンスファイルシステム) : IBM OS/2の標準ファイルシステム。読み取り専用モードでのみサポートされます。
iso9660	CD-ROMの標準ファイルシステム。
minix	このファイルシステムは、オペレーティングシステムに関する学術的なプロジェクトを起源とするもので、Linuxで最初に使用されたファイルシステムです。現在では、フロッピーディスク用のファイルシステムとして使用されています。
msdos	fat 、つまり当初はDOSで使用されていたファイルシステムであり、現在はさまざまなオペレーティングシステムで使用されています。
nfs	Network File System (ネットワークファイルシステム) : ネットワーク内の任意のコンピュータにデータを格納でき、ネットワーク経由でアクセスを付与できます。

ファイルシステムのタイプ	説明
<u>ntfs</u>	Windows NT file system (NTファイルシステム)：読み取り専用です。
<u>smbfs</u>	Server Message Block (サーバメッセージブロック): Windowsのような製品が、ネットワーク経由でのファイルアクセスを可能にする目的で採用しています。
<u>sysv</u>	SCO UNIX、Xenix、およびCoherent(PC用の商用UNIXシステム)が採用。
<u>ufs</u>	BSD、SunOS、およびNextStepで使用されています。読み取り専用モードでサポートされています。
<u>umsdos</u>	UNIX on MS-DOS(MS-DOS上のUNIX) - 標準 <u>fat</u> ファイルシステムに適用され、特別なファイルを作成することによりUNIXの機能(パーミッション、リンク、長いファイル名)を実現します。
<u>vfat</u>	Virtual FAT: <u>fat</u> ファイルシステムを拡張したものです(長いファイル名をサポートします)。

1.9 Linux環境での大規模ファイルサポート

31

当初、Linuxは、最大ファイルサイズとして 2GiB (2 バイト)をサポートしていました。また、ファイルシステムに大規模ファイルサポートが付いていない限り、32ビットシステム上での最大ファイルサイズは2GiBです。

現在、弊社のすべての標準ファイルシステムでは、LFS (大規模ファイルサポート)を提供しています。LFSは、理論的には、2 バイトの最大ファイルサイズをサポートします。表 1.2「ファイルおよびファイルシステムの最大サイズ(オンディスクフォーマット、4KiBブロックサイズ)」では、Linuxのファイルとファイルシステムの、現行のオンディスクフォーマットの制限事項を概説しています。表内の数字は、ファイルシステムで使用しているブロックサイズが、共通規格である4KiBであることを前提としています。異なるブロックサイズを使用すると結果は異なります。スパースブロックを使用している場合、表1.2「ファイルおよびファイルシステムの最大サイズ(オンディスクフォーマット、4KiBブロックサイズ)」に記載の最大ファイルサイズは、ファイルシステムの実際のサイズより大きいことがあります。



注記: バイナリの倍数

このマニュアルでの換算式: 1024バイト = 1KiB、1024KiB = 1MiB、1024MiB = 1GiB、1024GiB = 1TiB、1024TiB = 1PiB、1024PiB = 1EiB (「NIST: Prefixes for Binary Multiples (<http://physics.nist.gov/cuu/Units/binary.html>)」も参照してください)。

表 1.2: ファイルおよびファイルシステムの最大サイズ(オンディスクフォーマット、4KiBブロックサイズ)

ファイルシステム(4KiBブロックサイズ)	ファイルシステムの最大サイズ	ファイルの最大サイズ
Btrfs	16EiB	16EiB
Ext3	16TiB	2TiB
Ext4	1EiB	16TiB
OCFS2 (High Availability Extensionで使用可能な、クラスタ認識のファイルシステム)	16TiB	1EiB
XFS	8EiB	8EiB
NFSv2 (クライアント側)	8EiB	2GiB
NFSv3/NFSv4 (クライアント側)	8EiB	8EiB



重要: 制限

表1.2「ファイルおよびファイルシステムの最大サイズ(オンディスクフォーマット、4KiBブロックサイズ)」は、ディスクフォーマット時の制限について説明しています。Linuxカーネルは、操作するファイルとファイルシステムのサイズについて、独自の制限を課しています。管理の初期設定には、次のオプションがあります。

ファイルサイズ

41

32ビットシステムでは、ファイルサイズが2TiB (2 バイト)を超えることはできません。

ファイルシステムのサイズは、最大2 バイトまで可能です。しかし、この制限は、現在使用可能なハードウェアが到達可能な範囲を上回っています。

1.10 Linuxのカーネルにおけるストレージの制限

表1.3「ストレージの制限」に、SUSE Linux Enterprise Serverに関連したストレージに関するカーネルの制限をまとめています。

表 1.3: ストレージの制限

ストレージの機能	制限
サポートされるLUNの最大数	ターゲットあたり16384 LUN。
単一LUNあたりのパスの最大数	デフォルトで無制限。それぞれのパスが、通常のLUNとして扱われます。 実際の制限は、ターゲットあたりのLUNの数と、HBAあたりのターゲットの数(ファイバチャネルHBAの場合は16777215)により決まります。
HBAの最大数	無制限。実際の制限は、システムのPCIスロットの量で決まります。
オペレーティングシステムあたりの、デバイスマッパーマルチパス付きパスの最大数(合計)	約1024。実際数は、デバイス番号文字列の長さによります。これはマルチパスツール内のコンパイル時変数であり、この制限が問題となる場合は増やすこともできます。
最大サイズ(ブロックデバイスごと)	最大8EiB。

1.11 ファイルシステムのトラブルシューティング

本項では、ファイルシステムに関するいくつかの既知の問題と、考えられる解決手段について説明します。

1.11.1 Btrfsエラー: デバイスに空き領域がない

Btrfsファイルシステムを使用しているルート(/)パーティションにデータを書き込めなくなります。「`No space left on device`」というエラーが表示されます。

考えられる原因とこの問題の回避策については、この後の各項を参照してください。

1.11.1.1 Snapperスナップショットによるディスク容量の使用

BtrfsファイルシステムでSnapperが動作している場合、「`No space left on device`」が表示される問題は、通常は、システム上にスナップショットとして保存されているデータが多すぎるために発生します。

Snapperからいくつかのスナップショットを削除することはできますが、スナップショットはすぐには削除されないので、必要な容量が解放されない可能性があります。

Snapperからファイルを削除するには:

1. 端末コンソールを開きます。
2. コマンドプロンプトで、たとえば「`btrfs filesystem show`」と入力します。

```
tux > sudo btrfs filesystem show
Label: none uuid: 40123456-cb2c-4678-8b3d-d014d1c78c78
Total devices 1 FS bytes used 20.00GB
devid 1 size 20.00GB used 20.00GB path /dev/sda3
```

3. 次のように入力します。

```
tux > sudo btrfs fi balance start MOUNTPOINT -usage=5
```

このコマンドは、データを空またはほぼ空のデータチャンクに再配置して、その容量を回収し、メタデータに再割り当てしようとします。この処理にはしばらくかかります(1TBで数時間)が、処理中もシステムは使用可能です。

4. Snapperのスナップショットを一覧にします。次のように入力します。

```
tux > sudo snapper -c root list
```

5. Snapperから1つ以上のスナップショットを削除します。次のように入力します。

```
tux > sudo snapper -c root delete SNAPSHOT_NUMBER(S)
```

必ず最も古いスナップショットを最初に削除してください。古いスナップショットほど、多くの容量を使用します。

この問題が発生しないように、Snapperのクリーンアップアルゴリズムを変更できます。詳細については、『管理ガイド』、第7章「Snapperを使用したシステムの回復とスナップショット管理」、7.5.1.2項「クリーンアップアルゴリズム」を参照してください。スナップショットクリーンアップを制御する設定値は、`EMPTY_*`、`NUMBER_*`、および `TIMELINE_*` です。

ファイルシステムディスクでBtrfsとSnapperを使用する場合、標準のストレージ案の2倍のディスク容量を確保しておくことが推奨されます。YaSTパーティショナは、ルートファイルシステムでBtrfsを使用する場合のストレージ案として、自動的に標準の2倍のディスク容量を提案します。

1.11.1.2 ログ、クラッシュ、およびキャッシュのファイルによるディスク容量の使用

システムディスクがデータでいっぱいになりつつある場合、`/var/log`、`/var/crash`、`/var/lib/systemd/coredump`、および `/var/cache` からファイルを削除する方法があります。

Btrfs `root` ファイルシステムのサブボリューム `/var/log`、`/var/crash` および `/var/cache` が、通常の操作時に利用可能なディスクスペースのすべてを使用でき、システムに不具合が発生します。この状況を回避するため、SUSE Linux Enterprise Serverではサブボリュームに対するBtrfsクォータのサポートを提供するようになりました。詳細については、[1.2.5項「サブボリュームに対するBtrfsクォータのサポート」](#)を参照してください。

テストおよび開発用のマシンでは、特にアプリケーションが頻繁にクラッシュする場合、コアダンプが保存されている `/var/lib/systemd/coredump` を確認することもできます。

1.11.2 未使用のファイルシステムブロックの解放

SSD(Solid-State Drive)およびシンプロビジョニングされたボリュームでは、ファイルシステムによって使用されていないブロックに対してTrimを実行すると効果的です。SUSE Linux Enterprise Serverは、`unmap` または `trim` の手法をサポートするすべてのファイルシステムで、これらの操作を完全にサポートします。

SUSE Linux Enterprise Serverでサポートされるファイルシステム(Btrfsを除く)のTrimの方法としては、`/sbin/wiper.sh` を実行することをお勧めします。このスクリプトを実行する前に、必ず `/usr/share/doc/packages/hdparm/README.wiper` を読んでください。ほとんどのデスクトップおよびサーバシステムでは、Trimは週1回実行すれば十分です。ファイルシステムを `-o discard` でマウントすることは、パフォーマンスの低下を伴い、SSDの寿命に悪影響を与えることがあるため、推奨しません。







警告: Btrfsで**wiper.sh**を使用しないでください。


wiper.sh スクリプトは、読み書き可能でマウントされたExt4およびXFSファイルシステム、読み込み専用でマウント/マウント解除されたExt2、Ext3、Ext4、およびXFSファイルシステムにTrim操作を実行します。データを破損する可能性があるため、Btrfsファイルシステムで**wiper.sh**を使用しないでください。代わりに、`btrfsmaintenance` パッケージの一部である、`/usr/share/btrfsmaintenance/btrfs-trim.sh` を使用してください。

1.12 追加情報

ここまでで説明した各ファイルシステムのプロジェクトには、独自のWebページがあります。そこで詳しいドキュメントとFAQ、さらにメーリングリストを参照することができます。

- Kernel.orgのBtrfs Wiki: <https://btrfs.wiki.kernel.org/> 
- E2fsprogs: Ext2/3/4 File System Utilities: <http://e2fsprogs.sourceforge.net/> 
- Introducing Ext3: <http://www.ibm.com/developerworks/linux/library/l-fs7/> 
- OCFS2プロジェクト: <http://oss.oracle.com/projects/ocfs2/> 

Linuxファイルシステムの総合的なマルチパートチュートリアルは、「Advanced File System Implementor's Guide」 (<https://www.ibm.com/developerworks/linux/library/l-fs/> )にあるIBM developerWorksで見つけることができます。

ファイルシステム(Linuxファイルシステムに限らない)の詳しい比較については、Wikipediaプロジェクトの「Comparison of file systems」 (http://en.wikipedia.org/wiki/Comparison_of_file_systems#Comparison )を参照してください。

2 ファイルシステムのサイズ変更

ファイルシステムのサイズ変更(パーティションまたはボリュームのサイズ変更と混同しないでください)を使用して、物理ボリュームの使用可能な容量を増やしたり、物理ボリュームで増やした使用可能な容量を使用したりできます。

2.1 使用例

パーティションまたは論理ボリュームのサイズ変更には、YaSTパーティショナを使用することをお勧めします。その際、ファイルシステムは自動的にパーティションまたはボリュームの新しいサイズに合わせて調整されます。ただし、YaSTではファイルシステムのサイズ変更はサポートされていないので、次のようなケースでは手動でサイズを変更する必要があります。

- VM Guestの仮想ディスクのサイズを変更した後。
- NAS (Network Attached Storage)のボリュームのサイズを変更した後。
- 手動でパーティションのサイズを変更した後(たとえば、**fdisk** または **parted** を使用)、または論理ボリュームのサイズを変更した後(たとえば、**lvresize** を使用)。
- Btrfsファイルシステムを縮小する場合(SUSE Linux Enterprise Server 12の時点ではYaSTはBtrfsファイルシステムの拡大のみをサポートしています)。

2.2 サイズ変更のガイドライン

ファイルシステムのサイズ変更には、データを失う可能性をはらむリスクが伴います。



警告: データのバックアップ

データの喪失を避けるには、データを必ずバックアップしてから、サイズ変更タスクを開始します。

ファイルシステムのサイズを変更する場合は、次のガイドラインに従ってください。

2.2.1 サイズ変更をサポートしているファイルシステム

ボリュームに使用可能な容量を増やせるようにするには、ファイルシステムがサイズ変更をサポートしている必要があります。SUSE® Linux Enterprise Serverでは、ファイルシステムExt2、Ext3、およびExt4に対して、ファイルシステムのサイズ変更ユーティリティを使用できます。このユーティリティは、次のようにサイズの増減をサポートします。

表 2.1: ファイルシステムサイズ変更のサポート

ファイルシステム	ユーティリティ	サイズを増加(拡大)	サイズの削減(縮小)
Btrfs	<u>btrfs filesystem resize</u>	オンライン	オンライン
XFS	<u>xfs_growfs</u>	オンライン	サポートされていません。
Ext2	<u>resize2fs</u>	オンラインまたはオフライン	オフラインのみ
Ext3	<u>resize2fs</u>	オンラインまたはオフライン	オフラインのみ
Ext4	<u>resize2fs</u>	オンラインまたはオフライン	オフラインのみ

2.2.2 ファイルシステムのサイズの増加

デバイス上で使用可能な最大容量までファイルシステムを拡大することも、正確なサイズを指定することもできます。ファイルシステムのサイズを拡大する前に、必ずデバイス、または論理ボリュームのサイズを拡大しておいてください。

ファイルシステムに正確なサイズを指定する場合は、その新しいサイズが次の条件を満たすかどうかを必ず確認してください。

- 新しいサイズは、既存データのサイズより大きくなければなりません。さもないと、データが失われます。
- ファイルシステムのサイズは使用可能な容量より大きくできないので、新しいサイズは、現在のデバイスサイズ以下でなければなりません。

2.2.3 ファイルシステムのサイズの削減

デバイス上のファイルシステムのサイズを削減する際には、新しいサイズが次の条件を満たすかどうかを必ず確認してください。

- 新しいサイズは、既存データのサイズより大きくなければなりません。さもないと、データが失われます。
- ファイルシステムのサイズは使用可能な容量より大きくできないので、新しいサイズは、現在のデバイスサイズ以下でなければなりません。

ファイルシステムが保存されている論理ボリュームのサイズを削減する場合は、デバイス、または論理ボリュームのサイズを削減しようとする前に、必ずファイルシステムのサイズを削減しておきます。

！ 重要: XFS

XFSでフォーマットされたファイルシステムのサイズを縮小することはできません。XFSではそのような機能がサポートされていないためです。

2.3 Btrfsファイルシステムのサイズの変更

Btrfsファイルシステムのサイズは、ファイルシステムがマウントされているときに、**btrfs filesystem resize** コマンドを使用して変更できます。ファイルシステムのマウント中にサイズの増加と縮小の両方を実行できます。

1. 端末コンソールを開きます。
2. 変更するファイルシステムがマウントされていることを確認します。
3. 次のどちらかの方法で **btrfs filesystem resize** コマンドを使用して、ファイルシステムのサイズを変更します。

- ファイルシステムのサイズをデバイスの使用可能な最大サイズまで拡張するには、次のように入力します。

```
tux > sudo btrfs filesystem resize max /mnt
```

- ファイルシステムを特定のサイズに拡張するには、次のコマンドを入力します。

```
tux > sudo btrfs filesystem resize SIZE /mnt
```

SIZE を目的のサイズ(バイト単位)で置き換えます。50000K(キロバイト)、250M(メガバイト)、2G (ギガバイト)など、値の単位を指定することもできます。または、プラス(+)記号またはマイナス(-)記号を値の前に付けることにより、現在のサイズに対する増減を指定することもできます。

```
tux > sudo btrfs filesystem resize +SIZE /mnt
sudo btrfs filesystem resize -SIZE /mnt
```

4. 次のように入力して、マウントされたファイルシステムに対するサイズ変更の効果をチェックします。

```
tux > df -h
```

ディスクフリー(**df**)コマンドは、ディスクの合計サイズ、使用されたブロック数、およびファイルシステムで使用可能なブロック数を表示します。**-h**オプションは、読みやすい形式でサイズを出力します(1K、234M、2Gなど)。

2.4 XFSファイルシステムのサイズの変更

XFSファイルシステムのサイズは、ファイルシステムがマウントされているときに、**xfs_growfs** コマンドを使用して増加できます。XFSファイルシステムのサイズを縮小することはできません。

1. 端末コンソールを開きます。
2. 変更するファイルシステムがマウントされていることを確認します。
3. **xfs_growfs** コマンドを使用して、ファイルシステムのサイズを増やします。次に、ファイルシステムのサイズを、利用可能な最大値まで増やす例を示します。他のオプションについては、**man 8 xfs_growfs** を参照してください。

```
tux > sudo xfs_growfs -d /mnt
```

4. 次のように入力して、マウントされたファイルシステムに対するサイズ変更の効果をチェックします。

```
tux > df -h
```

ディスクフリー(**df**)コマンドは、ディスクの合計サイズ、使用されたブロック数、およびファイルシステムで使用可能なブロック数を表示します。**-h**オプションは、読みやすい形式でサイズを出力します(1K、234M、2Gなど)。

2.5 Ext2、Ext3、またはExt4の各ファイルシステムのサイズの変更

Ext2、Ext3、およびExt4ファイルシステムのサイズは、各パーティションがマウントされているかどうかにかかわらず、**resize2fs** コマンドを使用して増加できます。Extファイルシステムのサイズを減らすには、ファイルシステムをアンマウントする必要があります。

1. 端末コンソールを開きます。
2. ファイルシステムのサイズを減らす必要がある場合は、アンマウントします。
3. 次のどちらかの方法で、ファイルシステムのサイズを変更します。
 - ファイルシステムのサイズを `/dev/sda1` と呼ばれるデバイスの、利用可能な最大サイズまで拡大するには、次のように入力します。

```
tux > sudo resize2fs /dev/sda1
```

`size` パラメータを指定しない場合、サイズはパーティションのサイズにデフォルト設定されます。

- ファイルシステムを特定のサイズに変更するには、次のコマンドを入力します。

```
tux > sudo resize2fs /dev/sda1 SIZE
```

`SIZE` パラメータは、要求されたファイルシステムの新サイズを指定します。単位を指定しない場合の `size` パラメータの単位は、ファイルシステムのブロックサイズです。オプションとして、`size` パラメータの後ろに、次の単位指定子の1つを付けることができます。`s` は512バイトのセクタ、`K` はキロバイト(1キロバイトは1024バイト)、`M` はメガバイト、`G` はギガバイトを表します。

サイズ変更が完了するまで待って、続行します。

4. ファイルシステムがマウントされていない場合は、この時点で、ファイルシステムをマウントします。
5. 次のように入力して、マウントされたファイルシステムに対するサイズ変更の効果をチェックします。

```
tux > df -h
```

ディスクフリー(df)コマンドは、ディスクの合計サイズ、使用されたブロック数、およびファイルシステムで使用可能なブロック数を表示します。-hオプションは、読みやすい形式でサイズを出力します(1K、234M、2Gなど)。

3 UUIDによるデバイスのマウント

本項では、デバイス名(`/dev/sda1` など)の代わりにUUID (Universally Unique Identifier) を使用してファイルシステムデバイスを識別する方法について説明します。SUSE Linux Enterprise Server 12から、ブートローダファイルと `/etc/fstab` ファイルではデフォルトでUUIDが使用されています。

3.1 udevによる永続的なデバイス名

Linuxカーネル2.6以降、**udev** によって、永続的なデバイス名を使用した動的な `/dev` ディレクトリのユーザスペースソリューションが提供されます。システムに対してデバイスを追加または削除する場合は、ホットプラグシステムの一部として **udev** が実行されます。

ルールが特定デバイス属性との比較に使用されます。**udev** ルールのインフラストラクチャ(`/etc/udev/rules.d` ディレクトリで定義)は、すべてのディスクデバイスに、それらの認識順序や当該デバイスに使用される接続に関わらず、安定した名前を提供します。**udev** ツールは、カーネルが作成するすべての該当ブロックデバイスを調べ、一定のバス、ドライブタイプ、またはファイルシステムに基づいて、ネーミングルールを適用します。**udev** 用の独自ルールを定義する方法については、「Writing udev Rules (http://reactivated.net/writing_udev_rules.html)」を参照してください。

動的なカーネル提供のデバイスノード名に加えて、**udev** は、`/dev/disk` ディレクトリ内のデバイスをポイントする永続的なシンボリックリンクのクラスを保持します。このディレクトリは、さらに、`by-id`、`by-label`、`by-path`、および `by-uuid` の各サブディレクトリに分類されます。



注記: UUIDジェネレータ

udev 以外のプログラム(LVMや `md` など)も、UUIDを生成することがありますが、それらのUUIDは `/dev/disk` にリストされません。

3.2 UUIDの理解

UUID (Universally Unique Identifier)は、ファイルシステムの128ビットの番号であり、ローカルシステムと他のシステム全体に渡る固有な識別子です。UUIDは、システムハードウェア情報とタイムスタンプをそのシードの一部として、ランダムに生成されます。UUIDは、通常、デバイスに固有なタグを付けるために使用されます。

非永続的な「従来の」デバイス名(`/dev/sda1` など)を使用すると、ストレージを追加したときに、システムがブートできなくなる可能性があります。たとえば、`root (/)`が`/dev/sda1`に割り当てられている場合、SANを接続した後またはシステムにハードディスクを追加した後に、`/dev/sdg1`に再割り当てされる可能性があります。この場合、ブートローダ設定と`/etc/fstab` ファイルを調整する必要があり、そうしないとシステムは起動できなくなります。

この問題を回避する1つの方法は、ブートデバイスのブートローダファイルと`/etc/fstab` ファイルでUUIDを使用することです。SUSE Linux Enterpriseのバージョン12以降は、これがデフォルトです。UUIDは、ファイルシステムのプロパティであり、ドライブを再フォーマットすれば変更できます。デバイス名のUUIDを使用することの代替案として、IDまたはラベルでデバイスを識別する方法があります。

UUIDは、ソフトウェアRAIDデバイスのアセンブルと起動の基準としても使用できます。RAIDが作成されると、`md` ドライバは、デバイスのUUIDを生成し、その値を`md` スーパーブロックに保存します。

どのブロックデバイスのUUIDも、`/dev/disk/by-uuid` ディレクトリ内で見つけることができます。たとえば、UUIDエントリは次のようになります。

```
tux > ls -og /dev/disk/by-uuid/  
lrwxrwxrwx 1 10 Dec  5 07:48 e014e482-1c2d-4d09-84ec-61b3aefde77a -> ../../sda1
```

3.3 追加情報

`udev` によるデバイス管理の詳細については、『[管理ガイド](#)』、第22章「`udev`による動的カーネルデバイス管理」を参照してください。

`udev` コマンドの詳細については、[man 7 udev](#)を参照してください。

4 ブロックデバイス操作の多層キャッシング

多層キャッシュは、2つ以上の層で構成される複製/分散キャッシュです。1つは低速であるものの安価な回転方式のブロックデバイス(ハードディスク)に代表され、もう1つは高価であるもののデータ操作を高速に実行します(SSDフラッシュディスクなど)。

SUSE Linux Enterprise Serverは、フラッシュデバイスと回転方式のデバイスとの間のキャッシング用に、それぞれ `bcache` および `lvmcache` という2つの異なるソリューションを実装しています。

4.1 一般的な用語

本項では、キャッシュ関連機能の説明でよく使用されるいくつかの用語について説明します。

マイグレーション

論理ブロックの主コピーをデバイス間で移動すること。

昇格

低速なデバイスから高速なデバイスへのマイグレーション。

降格

高速なデバイスから低速なデバイスへのマイグレーション。

起点デバイス

大容量で低速なブロックデバイス。古いか、キャッシュデバイス上のコピーとの同期が保たれている(ポリシーによります)、論理ブロックのコピーが常に含まれます。

キャッシュデバイス

小容量で高速なブロックデバイス。

メタデータデバイス

キャッシュに入っているブロック、ダーティブロック、およびポリシーオブジェクトが使用する追加のヒントを記録する小容量のデバイス。この情報はキャッシュデバイスに配置することもできますが、別個に保持することにより、ボリュームマネージャで異なった設定にすることができます。たとえば、堅牢性を強化するためのミラーとして設定できます。メタデータデバイスを使用できるキャッシュデバイスは1つだけです。

ダーティブロック

何らかのプロセスがキャッシュに配置されたデータブロックに書き込みを行う場合、そのキャッシュされているブロックは、キャッシュ内で上書きされていて、元のデバイスにもう一度書き込む必要があるため、「ダーティ」とマークされます。

キャッシュミス

I/O操作の要求は、まず、キャッシュされたデバイスのキャッシュを参照します。要求された値が見つからなかった場合、デバイス自体を検索しますが、これは低速です。これを「キャッシュミス」と呼びます。

キャッシュヒット

要求された値がキャッシュされたデバイスのキャッシュ内で見つかった場合、その値は高速に提供されます。これを「キャッシュヒット」と呼びます。

コールドキャッシュ

値が格納されていない(空の)キャッシュのことで、「キャッシュミス」を引き起こします。キャッシュされたブロックデバイスの操作が進むにつれて、キャッシュはデータで満たされていき、「ウォーム」になります。

ウォームキャッシュ

すでに何らかの値が格納されていて、「キャッシュヒット」になる確立が高いキャッシュ。

4.2 キャッシングモード

多層キャッシュで使用する基本的なキャッシングモードは、「ライトバック」、「ライトスルー」、「ライトアラウンド」、および「パススルー」です。

ライトバック

キャッシュされているブロックに書き込まれたデータは、キャッシュにのみ書き込まれ、そのブロックはダーティとマークされます。これはデフォルトのキャッシングモードです。

ライトスルー

キャッシュされているブロックへの書き込みは、起点デバイスとキャッシュデバイスの両方にヒットするまで完了しません。「ライトスルー」キャッシュでは、クリーンブロックはクリーンな状態のままです。

ライトアラウンド

ライトスルーキャッシュと同様の手法ですが、書き込みI/Oは、キャッシュをバイパスして永続ストレージに直接書き込まれます。この手法では、直後に再読み込みされない書き込みI/Oによってキャッシュがいっぱいになるのを防ぐことができますが、最近書き込まれたデータの読み込み要求で「キャッシュミス」が発生し、低速なバルクストレージからの読み込みが必要になり、レイテンシが増加するという欠点があります。

パススルー

「パススルー」モードを有効にするには、キャッシュがクリーンである必要があります。読み込みは、キャッシュをバイパスして起点デバイスから実行されます。書き込みは起点デバイスに転送され、キャッシュブロックは「無効化」されます。「パススルー」では、データ整合性が維持されるため、データ整合性を気にすることなくキャッシュデバイスをアクティブ化できます。書き込みが実行されるにつれて、キャッシュは徐々にコールドになります。後でキャッシュの整合性を検証できる場合、または `invalidate_cblocks` メッセージを使用して整合性を保証できる場合は、キャッシュデバイスがまだウォームである間に、デバイスを「ライトスルー」または「ライトバック」モードに切り替えることができます。それ以外の場合は、目的のキャッシングモードに切り替える前に、キャッシュの内容を破棄できます。

4.3 bcache

`bcache` はLinuxカーネルブロック層のキャッシュです。1台以上の高速なディスクドライブ (SSDなど) を1台以上の低速なハードディスクのキャッシュとして動作させることができます。`bcache` は、ライトスルーとライトバックをサポートし、使用するファイルシステムから独立しています。デフォルトでは、SSDの強みである、ランダム読み込みとランダム書き込みのみのキャッシュを実行します。デスクトップやサーバのほか、ハイエンドのストレージアレイにも適しています。

4.3.1 主な特徴

- 1つのキャッシュデバイスを使用して、任意の数のバッキングデバイスをキャッシュできます。バッキングデバイスは、マウント中および使用中のランタイムに接続および切断できます。
- 不正なシャットダウンから回復します。キャッシュがバッキングデバイスと整合性があるようになるまで、書き込みは完了しません。
- 輻輳する場合、SSDへのトラフィックを制限します。
- 非常に効率的なライトバック実装。ダーティデータは常にソートされた順序で書き込まれます。
- 運用環境での使用における安定性と信頼性。

4.3.2 bcacheデバイスのセットアップ

この項では、bcache デバイスのセットアップと管理の手順を説明します。

1. bcache-tools パッケージをインストールします。

```
tux > sudo zypper in bcache-tools
```

2. バッキングデバイスを作成します(通常は機械式ドライブ)。デバイス全体、パーティション、またはその他の標準ブロックデバイスをバッキングデバイスにすることができます。

```
tux > sudo make-bcache -B /dev/sdb
```

3. キャッシュデバイスを作成します(通常はSSDディスク)。

```
tux > sudo make-bcache -C /dev/sdc
```

この例では、デフォルトのブロックサイズとバケットサイズである512Bと128KBを使用しています。ブロックサイズはバッキングデバイスのセクタサイズ(通常は512または4k)と一致している必要があります。バケットサイズは、書き込みの増大を防ぐために、キャッシングデバイスの消去ブロックサイズと一致している必要があります。たとえば、セクタが4kのハードディスクと消去ブロックサイズが2MBのSSDを使用する場合、このコマンドは次のようになります。

```
sudo make-bcache --block 4k --bucket 2M -C /dev/sdc
```



ヒント: 複数デバイスのサポート

make-bcache は、複数のバッキングデバイスとキャッシュデバイスを同時に準備および登録できます。この場合、後から手動でキャッシュデバイスをバッキングデバイスに接続する必要はありません。

```
tux > sudo make-bcache -B /dev/sda /dev/sdb -C /dev/sdc
```

4. bcache デバイスは次のように表示されます。

```
/dev/bcacheN
```

さらに、次のようにも表示されます。

```
/dev/bcache/by-uuid/UUID
```

```
/dev/bcache/by-label/LABEL
```

bcache デバイスは通常の方法で正常にフォーマットおよびマウントできます。

```
tux > sudo mkfs.ext4 /dev/bcache0  
tux > sudo mount /dev/bcache0 /mnt
```

bcache デバイスは、/sys/block/bcacheN/bcache にある sysfs によって制御できます。

5. キャッシュデバイスとバックアップデバイスの両方を登録した後、バックアップデバイスを関連キャッシュセットに接続して、キャッシュを有効にする必要があります。

```
tux > echo CACHE_SET_UUID > /sys/block/bcache0/bcache/attach
```

CACHE_SET_UUID は /sys/fs/bcache で確認できます。

6. デフォルトでは、bcache はパススルーキャッシングモードを使用します。たとえば、これをライトバックに変更するには、次のコマンドを実行します。

```
tux > echo writeback > /sys/block/bcache0/bcache/cache_mode
```

4.3.3 sysfsを使用した**bcache**の設定

bcache デバイスは、sysfs インタフェースを使用してランタイム設定値を保存します。このようにして、bcache バックアップディスクとキャッシュディスクの動作を変更したり、使用状況の統計を表示したりできます。

bcache sysfs の全パラメータのリストについては、/usr/src/linux/Documentation/bcache.txt ファイルの説明を参照してください。主に、SYSFS - BACKING DEVICE、SYSFS - BACKING DEVICE STATS、および SYSFS - CACHE DEVICE の各セクションで扱っています。

4.4 lvmcache

lvmcache は、論理ボリューム(LV)で構成されるキャッシングメカニズムです。dm-cache カーネルドライバを使用し、ライトスルー(デフォルト)およびライトバックのキャッシングモードをサポートします。lvmcache は、データの一部をより高速で小容量のLVに動的に移行することによって、大容量で低速なLVのパフォーマンスを向上させます。LVMの詳細については、[パートII「論理ボリューム\(LVM\)」](#)を参照してください。

LVMでは、この小容量で高速なLVを「キャッシュプールLV」と呼びます。一方、大容量で低速なLVを「起点LV」と呼びます。dm-cacheの要件があるため、LVMは、キャッシュプールLVをさらに「キャッシュデータLV」と「キャッシュメタデータLV」という2つのデバイスに分割します。キャッシュデータLVは、速度の向上を目的として、起点LVからのデータブロックのコピーが保持される場所です。キャッシュメタデータLVには、データブロックが保存されている場所を指定するアカウントリング情報が格納されます。

4.4.1 lvmcacheの設定

この項では、LVMベースのキャッシングの作成と設定の手順を説明します。

1. 起点LVを作成します。新しいLVを作成するか既存のLVを使用して、起点LVにします。

```
tux > sudo lvcreate -n ORIGIN_LV -L 100G vg /dev/SLOW_DEV
```

2. キャッシュデータLVを作成します。このLVには、起点LVからのデータブロックが格納されます。このLVのサイズがキャッシュのサイズになり、キャッシュプールLVのサイズとして報告されます。

```
tux > sudo lvcreate -n CACHE_DATA_LV -L 10G vg /dev/FAST
```

3. キャッシュメタデータLVを作成します。このLVには、キャッシュプールメタデータが格納されます。このLVのサイズは、キャッシュデータLVの約1000分の1にする必要があります。最小サイズは8MBです。

```
tux > sudo lvcreate -n CACHE_METADATA_LV -L 12M vg /dev/FAST
```

これまでに作成したボリュームの一覧を表示します。

```
tux > sudo lvs -a vg
LV          VG      Attr          LSize   Pool Origin
cache_data_lv  vg     -wi-a----- 10.00g
cache_metadata_lv vg     -wi-a----- 12.00m
origin_lv     vg     -wi-a----- 100.00g
```

4. キャッシュプールLVを作成します。データLVとメタデータLVをキャッシュプールLVに結合します。同時にキャッシュプールLVの動作を設定できます。

CACHE_POOL_LV は、CACHE_DATA_LV の名前を引き継ぎます。

CACHE_DATA_LV は、CACHE_DATA_LV_cdata という名前に変更されて、非表示になります。

CACHE_META_LV は、CACHE_DATA_LV_cmeta という名前に変更されて、非表示になります。

```
tux > sudo lvconvert --type cache-pool \
--poolmetadata vg/cache_metadata_lv vg/cache_data_lv
```

```
tux > sudo lvs -a vg
```

LV	VG	Attr	LSize	Pool	Origin
cache_data_lv	vg	Cwi---C---	10.00g		
[cache_data_lv_cdata]	vg	Cwi-----	10.00g		
[cache_data_lv_cmeta]	vg	ewi-----	12.00m		
origin_lv	vg	-wi-a-----	100.00g		

5. キャッシュLVを作成します。キャッシュプールLVを起点LVにリンクして、キャッシュLVを作成します。

ユーザがアクセス可能なキャッシュLVは起点LVの名前を引き継ぎ、起点LVは非表示LVになって ORIGIN_LV_corig という名前に変更されます。

キャッシュLVは、ORIGIN_LV の名前を引き継ぎます。

ORIGIN_LV は、ORIGIN_LV_corig という名前に変更されて、非表示になります。

```
tux > sudo lvconvert --type cache --cachepool vg/cache_data_lv vg/origin_lv
```

```
tux > sudo lvs -a vg
```

LV	VG	Attr	LSize	Pool	Origin
cache_data_lv	vg	Cwi---C---	10.00g		
[cache_data_lv_cdata]	vg	Cwi-ao---	10.00g		
[cache_data_lv_cmeta]	vg	ewi-ao---	12.00m		
origin_lv	vg	Cwi-a-C---	100.00g	cache_data_lv	[origin_lv_corig]
[origin_lv_corig]	vg	-wi-ao---	100.00g		

4.4.2 キャッシュプールの削除

LVキャッシュをオフにする方法はいくつかあります。

4.4.2.1 キャッシュLVからキャッシュプールLVを切断

キャッシュプールLVをキャッシュLVから接続解除して、未使用キャッシュプールLVとキャッシュされていない起点LVを残すことができます。データは、必要に応じてキャッシュプールから起点LVに書き戻されます。

```
tux > sudo lvconvert --splitcache vg/origin_lv
```

4.4.2.2 起点LVを削除せずにキャッシュプールLVを削除

この方法では、必要に応じてキャッシュプールから起点LVにデータを書き戻してから、キャッシュプールLVを削除し、キャッシュされていない起点LVを残します。

```
tux > sudo lvremove vg/cache_data_lv
```

次に示す別のコマンドでも、キャッシュLVからキャッシュプールを接続解除し、キャッシュプールを削除します。

```
tux > sudo lvconvert --uncache vg/origin_lv
```

4.4.2.3 起点LVとキャッシュプールLVの両方を削除

キャッシュLVを削除すると、起点LVとリンクされたキャッシュプールLVの両方が削除されます。

```
tux > sudo lvremove vg/origin_lv
```

4.4.2.4 その他の情報

サポートされるキャッシュモード、冗長なサブ論理ボリューム、キャッシュポリシー、既存のLVからキャッシュタイプへの変換など、[lvmcache](#)に関連するその他のトピックは、[lvmcache](#)のマニュアルページ(`man 7 lvmcache`)で参照できます。

II 論理ボリューム(LVM)

5 LVMの設定 49

6 LVMボリュームスナップショット 80

5 LVMの設定

この章では、LVM (Logical Volume Manager)の原理と多くの状況で役立つ基本機能を説明します。YaST LVMの設定は、YaST Expert Partitionerからアクセスできます。このパーティショニングツールにより、既存のパーティションを編集、および削除できます。また、LVMで使用する新規パーティションを作成することもできます。



警告: リスク

LVMを使用することでデータ損失などの危険性が増加する恐れがあります。この危険性にはアプリケーションのクラッシュ、電源障害、誤ったコマンドなども含まれます。LVMまたはボリュームの再設定を実施する前にデータを保存してください。バックアップなしでは作業を実行しないでください。

5.1 論理ボリュームマネージャ(LVM)の理解

LVMは、複数の物理ボリューム(ハードディスク、パーティション、LUN)にハードディスクスペースを柔軟に分散することができます。LVMが開発された理由は、インストール中に初期パーティショニングが終了した後でのみ、ハードディスクスペースのセグメンテーションを変更するニーズが発生する可能性があるためです。実行中のシステムでパーティションを変更することは困難なので、LVMは必要に応じて論理ボリューム(LV)を作成できるストレージスペースの仮想プール(ボリュームグループ(VG))を提供します。オペレーティングシステムは物理パーティションの代わりにこれらのLVにアクセスします。ボリュームグループは2つ以上のディスクにまたがることができます。したがって、複数のディスクまたはそれらの一部で1つのVGを構成できます。この方法で、LVMは物理ディスクスペースから一種の抽象化を行います。この抽象化により、物理パーティショニングを使用する場合よりはるかに簡単で安全な方法でセグメンテーションを変更できます。

図5.1「物理パーティショニング対LVM」では物理パーティショニング(左)とLVM区分(右)を比較しています。左側は、1つのディスクが割り当てられたマウントポイント(MP)をもつ3つの物理パーティション(PART)に分かれています。これによりオペレーティングシステムはそれぞれのパーティションにアクセスできます。右側では2つのディスクがそれぞれ3つの物理パーティションに分かれています。2つのLVMボリュームグループ(VG 1およびVG 2)が定義されています。VG 1には、DISK 1からのパーティションが2つ、DISK 2からのパーティションが1つ含まれます。VG 2には、DISK 2からの残りの2パーティションが含まれます。

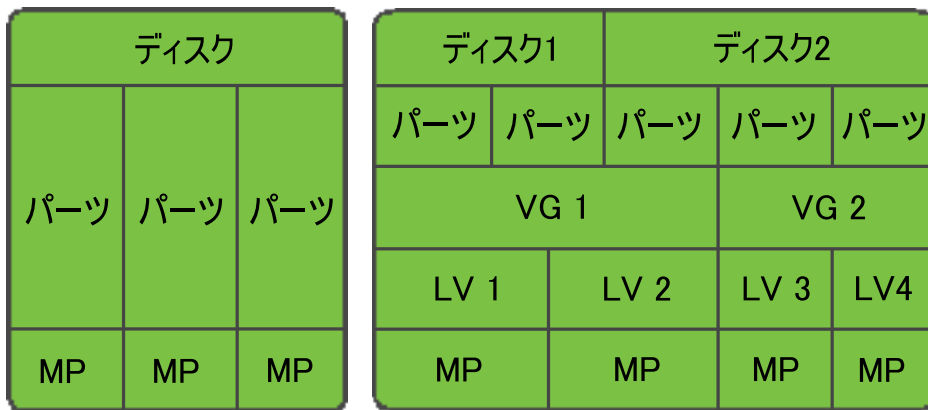


図 5.1: 物理パーティショニング対LVM

LVMでは、ボリュームグループに組み込まれた物理ディスクをPV (物理ボリューム)と呼びます。図5.1「物理パーティショニング対LVM」のボリュームグループ内には、4つの論理ボリューム(LV 1からLV 4)が定義されています。これらのボリュームは、関連付けられたマウントポイント(MP)を介してオペレーティングシステムに使用されます。別の論理ボリュームとの境界とパーティションの境界を並べることはできません。この例ではLV 1およびLV 2の間に境界があります。

LVMの機能:

- 複数のハードディスクまたはパーティションを大きな論理ボリュームにまとめることができます。
- 提供された設定が適切であれば、LV(/usr など)は空きスペースがなくなったときに拡張することができます。
- LVMを使用することで、実行中のシステムにハードディスクまたはLVを追加できます。ただし、そのためには、ディスクやLVを追加することのできるホットプラグ可能なハードウェアが必要になります。
- 複数の物理ボリューム上に論理ボリュームのデータストリームを割り当てるストライピングモードを有効にすることもできます。これらの物理ボリュームが別のディスクに存在する場合、RAID 0と同様に読み込みおよび書き込みのパフォーマンスを向上できます。
- スナップショット機能は稼働中のシステムで一貫性のある(特にサーバ)バックアップを取得できます。



注記: LVMとRAID


LVMは0/1/4/5/6のRAIDレベルもサポートしていますが、MD RAIDを使用することをお勧めします(第7章「ソフトウェアRAIDの設定」を参照してください)。ただし、LVMはRAID 0および1では適切に動作します。これは、RAID 0は一般的な論理ボリューム管理と同様である(個々の論理ブロックが物理デバイス上のブロックにマップされる)ためです。RAID 1上でLVMを使用した場合は、ミラーの同期を追跡して同期プロセスを完全に管理することができます。それより高いRAIDレベルでは、接続されたディスクの状態を監視するほか、ディスクアレイで問題が発生した場合に管理者に通知することのできる、管理デーモンが必要になります。LVMにはこのようなデーモンが組み込まれていますが、デバイス障害などの例外的な状況では、このデーモンは正しく機能しません。



警告: IBM Z: LVMルートファイルシステム

LVMまたはソフトウェアRAIDアレイでルートファイルシステムを使用してシステムを設定する場合、`/boot`を別個の非LVMまたは非RAIDパーティションに配置する必要があります。そうしないと、システムは起動しません。このパーティションの推奨サイズは500MBで、推奨ファイルシステムはExt4です。

これらの機能とともにLVMを使用することは、頻繁に使用されるホームPCや小規模サーバではそれだけでも意義があります。データベース、音楽アーカイブ、またはユーザディレクトリのように増え続けるデータストックがある場合は、LVMが特に役に立ちます。LVMを使用すると、物理ハードディスクより大きなファイルシステムの作成が可能になります。ただし、LVMでの作業は従来のパーティションでの作業とは異なることに留意してください。

YaSTパーティショナの使用によって、新規および既存のLVMストレージオブジェクトを管理できます。LVMの設定に関する指示や詳細情報については、公式のLVM HOWTO (<http://tldp.org/HOWTO/LVM-HOWTO/>) を参照してください。

5.2 ボリュームグループの作成

LVMボリュームグループ(VG)は、Linux LVMパーティションをスペースの論理プールにします。グループ内の使用可能なスペースから論理ボリュームを作成できます。グループ内のLinux LVMパーティションは、同じディスクに存在することも、さまざまなディスクに存在することも可能です。パーティションまたはディスク全体を追加することにより、グループのサイズを拡張できます。

ディスク全体を使用する場合、そのディスクにパーティションを含めることはできません。パーティションを使用した場合、それらをマウントしないでください。YaSTは、パーティションをVGに追加する際に自動的にパーティションタイプを 0x8E Linux LVM に変更します。

1. YaSTを起動してパーティショナを開きます。
2. 既存のパーティショニングセットアップを再設定する必要がある場合は、次の手順に従います。詳細については、『導入ガイド』、第10章「Expert Partitioner (エキスパートパーティショナ)」、10.1項「熟練者向けパーティション設定の使用」を参照してください。未使用のディスクまたはパーティションを使用したいだけの場合は、この手順をスキップしてください。



警告: パーティションされていないディスクの物理ボリューム

パーティションされていないディスクがオペレーティングシステムのインストール先(ブート元)ではない場合、そのディスクを物理ボリューム(PV)として使用することができます。

パーティションされていないディスクはシステムレベルで「未使用」として表示されるため、上書きされてしまったり、間違ってアクセスされたりする可能性があります。

- a. 既にパーティションが含まれているハードディスク全体を使用するには、そのディスク上にあるパーティションをすべて削除します。
 - b. 現在マウントされているパーティションを使用するには、そのパーティションをアンマウントします。
3. 左のパネルで、ボリューム管理を選択します。
既存のボリュームグループのリストが右のパネルに表示されます。
 4. [ボリューム管理] ページの左下で、追加 > ボリュームグループの順にクリックします。

ボリュームグループの追加

ボリュームグループ名

PEサイズ(P)
 4 MiB ▼

利用可能な物理ボリューム:

デバイス	サイズ	暗号	タイプ
/dev/sda1	10.00 GiB		Linux LVM
/dev/sda5	20.00 GiB		Linux native
/dev/sdb	8.00 GiB		QEMU-HARDDISK
/dev/sdc1	4.00 GiB		Linux native

合計サイズ: 42.00 GiB

選択した物理ボリューム:

デバイス	サイズ	暗号	タイプ
------	-----	----	-----

結果サイズ: 0 B

ヘルプ(H) 戻る(B) 中止(R) 完了(F)

5. ボリュームグループは次のように定義します。

a. ボリュームグループ名を指定します。

インストール時にボリュームグループを作成している場合は、SUSE Linux Enterprise Serverのシステムファイルを含むボリュームグループに対して system という名前が示唆されます。

b. PEサイズを指定します。

PEサイズは、ボリュームグループの物理ブロックのサイズを定義します。ボリュームグループにある全ディスクスペースはこの物理ブロックサイズ内で使用されます。値の範囲は、2の累乗で1KBから16GBまでです。通常、この値は4MBに設定されます。

LVM1では、LVごとに65534エクステントまでしかサポートしないので、4MBの物理エクステントで最大LVサイズとして256GBが可能でした。SUSE Linux Enterprise Serverで使用するLVM2では、物理エクステントの数に制限はありません。エクステントが多くても、論理ボリュームに対するI/Oパフォーマンスには影響しませんが、LVMツールの動作が遅くなります。

！ 重要: 物理エクステントサイズ

1つのボリュームグループに異なるサイズの物理エクステントを混在させないでください。初期設定後はエクステントを変更しないでください。

- c. 利用可能な物理ボリュームリストで、このボリュームグループに含めたいLinux LVMパーティションを選択し、追加をクリックして、それらのパーティションを選択した物理ボリュームリストに移動します。
 - d. 完了をクリックします。
ボリュームグループリストに新しいグループが表示されます。
6. [ボリューム管理] ページで、次へをクリックし、新しいグループが一覧されることを確認してから、完了をクリックします。
 7. ボリュームグループを構成している物理デバイスを確認するため、稼働中のシステムでYaSTパーティショナを開き、ボリューム管理 > 編集 > Physical Devices (物理デバイス)の順にクリックします。中止するをクリックしてこの画面を閉じます。

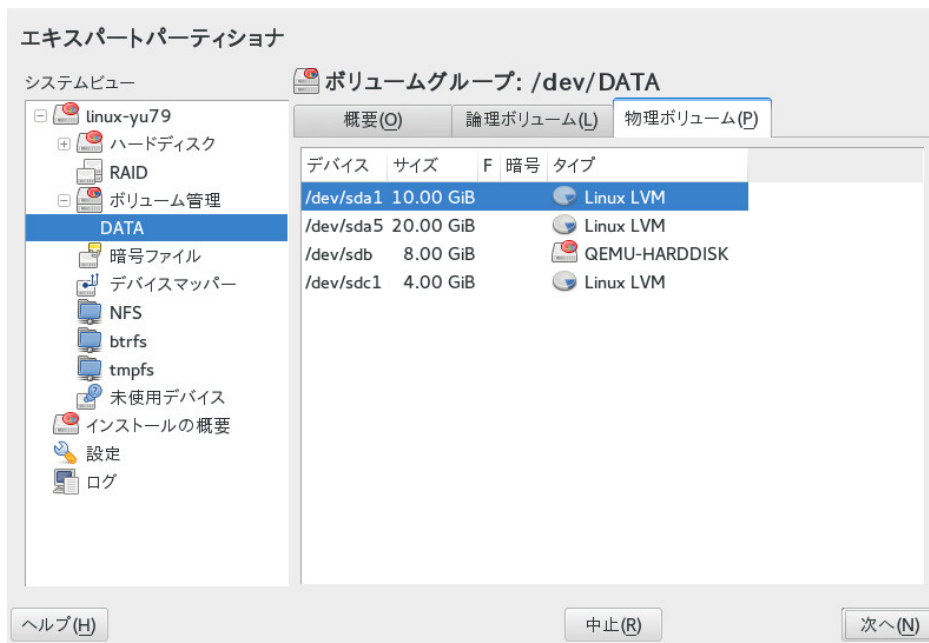


図 5.2: DATAという名前のボリュームグループ内の物理ボリューム

5.3 論理ボリュームの作成

論理ボリュームは、ハードディスクと同様に領域のプールを提供します。この領域を使用可能にするには、論理ボリュームを定義する必要があります。論理ボリュームは通常のパーティションに似ており、フォーマットやマウントが可能です。

YaSTパーティショナを使用して、既存のボリュームグループから論理ボリュームを作成します。各ボリュームグループに少なくとも1つの論理ボリュームを割り当ててください。ボリュームグループ内の空き領域を使い果たすまで、必要に応じて新しい論理ボリュームを作

成できます。LVM論理ボリュームをオプションでシンプロビジョニングすることによって、使用可能な空き領域を超えるサイズで論理ボリュームを作成することもできます(詳しくは5.3.1項「シンプロビジョニング論理ボリューム」を参照)。

- **通常のボリューム:** (デフォルト)ボリュームの領域は直ちに割り当てられます。
- **シンプール:** この論理ボリュームは、シンボリューム用に予約された領域のプールです。シンボリュームでは、必要な領域をそのプールからオンデマンドで割り当てることができます。
- **シンボリューム:** ボリュームは疎ボリュームとして作成されます。このボリュームでは、必要な領域はシンプールからオンデマンドで割り当てられます。
- **ミラーリングされたボリューム:** このボリュームは、定義した数のミラーで作成されます。

手順 5.1: 論理ボリュームの設定

1. YaSTを起動してパーティショナを開きます。
2. 左のパネルで、ボリューム管理を選択します。既存のボリュームグループのリストが右のパネルに表示されます。
3. ボリュームを作成するボリュームグループを選択して、追加 > 論理ボリュームの順に選択します。
4. 名前にボリューム名を入力し、通常ボリュームを選択します(シンプロビジョニングボリュームの設定については、5.3.1項「シンプロビジョニング論理ボリューム」を参照してください)。次へで続行します。

論理ボリュームを /dev/DATA に追加

名前
論理ボリューム
LOCAL

タイプ
☒ 通常ボリューム
☐ Thin プール
☐ Thin ボリューム
使用済みプール
▼

ヘルプ(H) 戻る(B) 中止(R) 次へ(N)

5. ボリュームのサイズと、複数ストライプを使用するかどうかを指定します。
ストライプボリュームを使用すると、データは複数の物理ボリュームに分散されます。
これらの物理ボリュームが別のハードディスクに存在する場合、この性質により、読み込みおよび書込みのパフォーマンスが向上します(RAID 0など)。利用可能な最大ストライプ数は、物理ボリュームの数と同じです。デフォルト(1)は、複数のストライプを使用しない設定です。

論理ボリューム LOCAL を /dev/DATA に追加

サイズ
☒ 最大サイズ (41.98 GiB)
☐ カスタムサイズ
サイズ
41.98 GiB

ストライプ
トラック番号 サイズ
1 64 KiB

ヘルプ(H) 戻る(B) 中止(R) 次へ(N)

6. 役割でボリュームの役割を選択します。ここで選択した内容は、次のダイアログのデフォルト値にのみ影響します。値は次の手順で変更可能です。わからない場合は、RAW ボリューム(未フォーマット)を選択します。



7. フォーマットオプションで、パーティションをフォーマットするを選択し、ファイルシステムを選択します。オプションメニューの内容は、ファイルシステムによって異なります。通常は、デフォルト値を変更する必要はありません。
マウントのオプションの下で、パーティションをマウントするを選択してから、マウントポイントを選択します。Fstabオプションをクリックして、このボリュームの特別なマウントオプションを追加します。
8. 完了をクリックします。
9. 次へをクリックし、変更が一覧されることを確認してから、完了をクリックします。

5.3.1 シンプロビジョニング論理ボリューム

LVM論理ボリュームはシンプロビジョニング可能です(オプション)。シンプロビジョニングを使用すると、利用可能な空き領域を超えるサイズの論理ボリュームを作成できます。任意の数のシンボリューム用に予約した未使用領域が含まれるシンプールの作成します。シンボリュームは疎ボリュームとして作成され、必要に応じてシンプールの領域が割り当てられます。ストレージ領域をコスト効果の高い方法で割り当てなければならなくなった場合、シンプールの

を動的に拡張できます。シンプロビジョニングボリュームは、Snapperで管理可能なスナップショットもサポートします。詳細については、『管理ガイド』、第7章「Snapperを使用したシステムの回復とスナップショット管理」を参照してください。

シンプロビジョニング論理ボリュームを設定するには、[手順5.1「論理ボリュームの設定」](#)の説明に従って作業を進めます。ボリュームタイプを選択する手順になったら、通常ボリュームを選択せずに、シンボリュームまたはシンプールを選択します。

シンプール

この論理ボリュームは、シンボリューム用に予約された領域のプールです。シンボリュームでは、必要な領域をそのプールからオンデマンドで割り当てることができます。

シンボリューム

ボリュームは疎ボリュームとして作成されます。このボリュームでは、必要な領域はシンプールからオンデマンドで割り当てられます。

！ 重要: クラスタにおけるシンプロビジョニングボリューム
クラスタでシンプロビジョニングボリュームを使用するには、クラスタを使用するシンプールとシンボリュームを1つのクラスタリソースで管理する必要があります。これにより、シンボリュームとシンプールを常に同じノードに排他的にマウントできます。

5.3.2 ミラーリングされたボリュームの作成

複数のミラーを使用して1つの論理ボリュームを作成できます。LVMは、下層の物理ボリュームに書き込まれたデータが別の物理ボリュームに確実にミラーリングされるようにします。そのため、1つの物理ボリュームがクラッシュしても、論理ボリューム上のデータにアクセスできます。LVMは、同期プロセスを管理するためのログファイルも保持します。このログには、現在ミラーとの同期を実行中のボリューム領域についての情報が含まれます。デフォルトでは、ログはディスク(可能であればミラーとは別のディスク)に保存されます。ただし、揮発性メモリなどの別の場所をログに指定できます。

現在のところ、使用可能なミラー実装のタイプには、「通常」(非RAID)の `mirror` 論理ボリュームと、`raid1` 論理ボリュームがあります。

ミラーリングされた論理ボリュームを作成したら、ミラーリングされた論理ボリュームで、アクティブ化、拡張、削除などの標準の操作を実行できます。

5.3.2.1 ミラーリングされた非RAID論理ボリュームの設定

ミラーリングされたボリュームを作成するには、**lvcreate** コマンドを使用します。次の例では、ボリュームグループ「vg1」を使用する、「lv1」という名前の2つのミラーを使用して、500GBの論理ボリュームを作成しています。

```
tux > sudo lvcreate -L 500G -m 2 -n lv1 vg1
```

このような論理ボリュームは、ファイルシステムのコピーを3つ提供するリニアボリューム(ストライピングなし)です。**m** オプションは、ミラーの数を指定します。**L** オプションは、論理ボリュームのサイズを指定します。

論理ボリュームは、デフォルトサイズである512KBの領域に分割されます。異なるサイズの領域が必要な場合は、**-R** オプションを使用します。このオプションの後に、目的の領域サイズをメガバイト単位で指定してください。または、**lvm.conf** ファイルの **mirror_region_size** オプションを編集して、好みの領域サイズを設定することもできます。

5.3.2.2 raid1論理ボリュームの設定

LVMはRAIDをサポートしているため、RAID1を使用してミラーリングを実装できます。このような実装には、非RAIDミラーと比較して次のような利点があります。

- LVMは、各ミラーイメージに対して完全に冗長なビットマップ領域を維持しており、これによって障害対応能力が向上する。
- ミラーイメージを一時的にアレイから分離し、マージして元に戻すことができる。
- 一時的な障害にアレイで対応できる。
- LVMのRAID 1実装はスナップショットをサポートする。

一方、このタイプのミラーリング実装では、クラスタ化されたボリュームグループ内に論理ボリュームを作成することはできません。

RAIDを使用してミラーボリュームを作成するには、次のコマンドを発行します。

```
tux > sudo lvcreate --type raid1 -m 1 -L 1G -n lv1 vg1
```

各オプション/パラメータには次のような意味があります。

- **--type - raid1** を指定する必要があります。指定しないと、暗黙のセグメントタイプ **mirror** が使用され、非RAIDミラーが作成されます。
- **-m** - ミラーの数を指定します。

- `-L` - 論理ボリュームのサイズを指定します。
- `-n` - このオプションを使用して、論理ボリュームの名前を指定します。
- `vg1` - 論理ボリュームで使用されるボリュームグループの名前です。

LVMは、アレイ内の各データボリュームに対して、1つのエクステントサイズの論理ボリュームを作成します。ミラーリングされたボリュームが2つある場合、LVMは、メタデータを保存する別のボリュームを2つ作成します。

RAID論理ボリュームを作成したら、一般的な論理ボリュームと同じ方法でそのボリュームを使用できます。アクティブ化、拡張などを行うことができます。

5.4 非ルートLVMボリュームグループの自動アクティビ化

非ルートLVMボリュームグループのアクティビ化の動作は、`/etc/lvm/lvm.conf` ファイルおよび `auto_activation_volume_list` パラメータで制御します。デフォルトでは、このパラメータは空で、すべてのボリュームがアクティビ化されます。一部のボリュームグループのみをアクティビ化するには、その名前を引用符で囲んで追加し、カンマで区切ります。次に例を示します。

```
auto_activation_volume_list = [ "vg1", "vg2/lvol1", "@tag1", "@*" ]
```

リストを `auto_activation_volume_list` パラメータで定義した場合、次のように処理されます。

1. 各論理ボリュームは、最初にこのリストに照らして確認されます。
2. 一致しない場合、論理ボリュームはアクティビ化されません。

デフォルトでは、非ルートLVMボリュームグループは、システムの再起動時にDracutによって自動的にアクティビ化されます。このパラメータにより、システムの再起動時にすべてのボリュームグループをアクティブにすることも、または指定した非ルートLVMボリュームグループのみをアクティブにすることもできます。

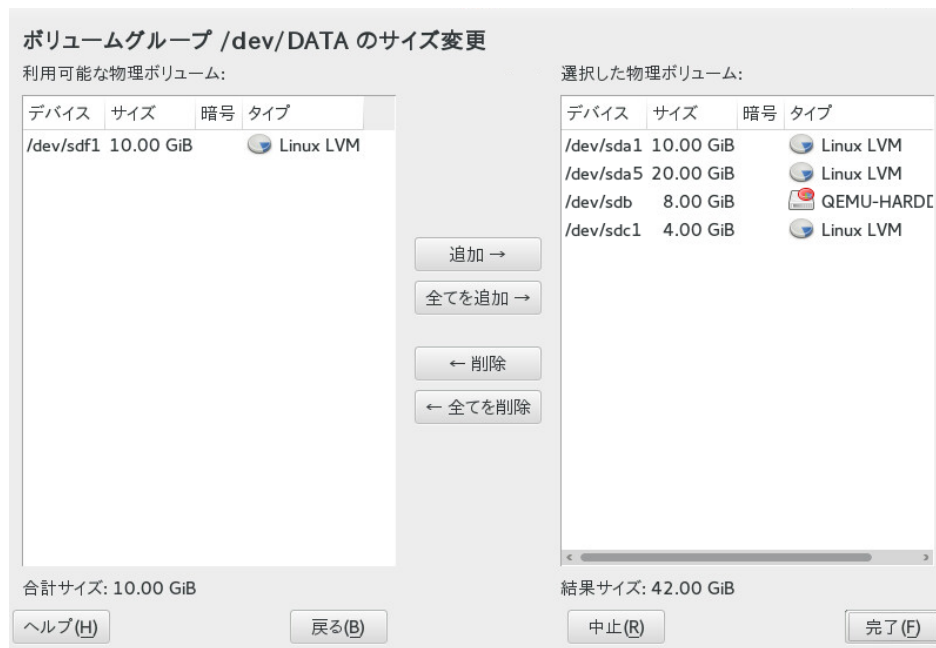
5.5 既存のボリュームグループのサイズ変更

ボリュームグループによって提供される領域は、物理ボリュームを追加することによっていつでも拡張できます。これは、システムの稼働中であっても、サービスを中断することなく実行できます。これにより、グループに論理ボリュームを追加したり、既存のボリュームのサイズを拡張したりできます。5.6項「[論理ボリュームのサイズ変更](#)」を参照してください。

また、物理ボリュームを削除してボリュームグループのサイズを縮小することもできます。YaSTで削除できる物理ボリュームは、現在未使用の物理ボリュームだけです。現在使用中の物理ボリュームを確認するには、次のコマンドを実行します。PE Ranges 列に表示されているパーティション(物理ボリューム)が使用中のものです。

```
tux > sudo pvs -o vg_name,lv_name,pv_name,seg_pe_ranges
root's password:
VG   LV    PV          PE Ranges
    /dev/sda1
DATA DEVEL /dev/sda5  /dev/sda5:0-3839
DATA    /dev/sda5
DATA LOCAL /dev/sda6  /dev/sda6:0-2559
DATA    /dev/sda7
DATA    /dev/sdb1
DATA    /dev/sdc1
```

1. YaSTを起動してパーティショナを開きます。
2. 左のパネルで、**ボリューム管理**を選択します。既存のボリュームグループのリストが右のパネルに表示されます。
3. 変更するボリュームグループを選択し、**サイズ変更**をクリックします。



4. 次のいずれかの操作を行います。

- **追加:** 1つまたは複数の物理ボリューム(LVMパーティション)を利用可能な物理ボリュームリストから選択した物理ボリュームリストに移動することにより、ボリュームグループのサイズを拡張します。
- **削除:** 1つまたは複数の物理ボリューム(LVMパーティション)を選択した物理ボリュームリストから使用可能な物理ボリュームリストに移動することにより、ボリュームグループのサイズを縮小します。

5. 完了をクリックします。

6. 次へをクリックし、変更が一覧されることを確認してから、完了をクリックします。

5.6 論理ボリュームのサイズ変更

ボリュームグループ内に利用可能な未使用の空き領域がある場合、論理ボリュームを拡張して使用可能な領域を増やすことができます。また、ボリュームのサイズを縮小してボリュームグループの領域を解放し、他の論理ボリュームで使えるようにすることもできます。



注記: 「オンライン」でのサイズ変更

ボリュームのサイズを縮小すると、そのファイルシステムのサイズもYaSTによって自動的に縮小されます。現在マウントされているボリュームのサイズを「オンライン」で(つまりマウント中に)変更できるかどうかは、ファイルシステムによって異なります。オンライン拡張をサポートするファイルシステムは、Btrfs、XFS、Ext3、およびExt4です。

オンライン縮小をサポートするファイルシステムは、Btrfsのみです。Ext2/3/4ファイルシステムを縮小するには、アンマウントする必要があります。XFSはファイルシステムの縮小をサポートしないため、XFSでフォーマットされたボリュームは縮小できません。

1. YaSTを起動してパーティショナを開きます。
2. 左のパネルで、ボリューム管理を選択します。既存のボリュームグループのリストが右のパネルに表示されます。
3. 変更する論理ボリュームを選択し、サイズ変更をクリックします。

/dev/DATA/LOCAL 論理ボリュームのサイズ変更

サイズ

☒ 最大サイズ (51.98 GiB)

☐ 最小サイズ (266.00 MiB)

☐ カスタムサイズ

サイズ

15.00 GiB

現在のサイズ: 15.00 GiB

ヘルプ(H) キャンセル(C) OK(O)

4. 次のオプションの1つを使用して目的のサイズを設定します。

- **最大サイズ**. 論理ボリュームのサイズを、ボリュームグループの残り領域をすべて使用するように拡張します。
- **最小サイズ**. 論理ボリュームのサイズを、データおよびファイルシステムメタデータによって使用されているサイズまで縮小します。
- **[Custom Size (カスタムサイズ)]** . ボリュームの新しいサイズを指定します。上に表示されている最小値から最大値までの範囲内の値を指定する必要があります。キロバイトにはK、メガバイトにはM、ギガバイトにはG、テラバイトにはTをそれぞれ使用します(たとえば 20G)。

5. OKをクリックします。

6. 次へをクリックし、変更が一覧されることを確認してから、完了をクリックします。

5.7 ボリュームグループまたは論理ボリュームの削除



警告: データ損失

ボリュームグループを削除すると、グループの各メンバーパーティションに含まれているデータがすべて破棄されます。論理ボリュームを削除すると、そのボリュームに保存されているデータがすべて破棄されます。

1. YaSTを起動してパーティショナを開きます。
2. 左のパネルで、ボリューム管理を選択します。既存のボリュームグループのリストが右のパネルに表示されます。
3. 削除するボリュームグループまたは論理ボリュームを選択して、Delete (削除)をクリックします。
4. 選択した内容に応じて警告ダイアログが表示されます。はいを選択して確認します。
5. 次へをクリックし、削除したボリュームグループが一覧されることを確認してから(削除は赤いフォントで示されます)、完了をクリックします。

5.8 LVMコマンドの使用

LVMコマンドの使用の詳細については、次の表で説明されている各コマンドのマニュアルページを参照してください。すべてのコマンドは root 特権で実行する必要があります。 sudo COMMAND を使用するか(推奨)、直接 root として実行します。

LVMコマンド

pvcreate DEVICE

LVMで物理ボリュームとして使用できるようにデバイス(/dev/sdb1 など)を初期化します。指定したデバイス上にファイルシステムが存在する場合、警告が表示されます。 blkid がインストールされている場合にのみ(デフォルトでインストールされています)、 pvcreate により既存のファイルシステムの有無が確認されることを覚えておいてください。 blkid が使用可能でない場合、 pvcreate によって何も警告が生成されず、警告なしにファイルシステムが失われる場合があります。

pvdisk DEVICE

LVM物理ボリュームに関する情報(現在、論理ボリュームで使用中かどうかなど)を表示します。

vgcreate -c y VG_NAME DEV1 [DEV2...]

指定した1つ以上のデバイスでクラスタ化ボリュームグループを作成します。

vgcreate --activationmode ACTIVATION_MODE VG_NAME

ボリュームグループのアクティブ化のモードを設定します。次のいずれかの値を指定できます。

- complete - 欠落している物理ボリュームの影響を受けない論理ボリュームのみをアクティブ化できます。特定の論理ボリュームでそのような障害が許容される場合も、同様の処理が実行されます。
- degraded - デフォルトのアクティブ化モードです。論理ボリュームをアクティブ化するための十分なレベルの冗長性がある場合、一部の物理ボリュームが欠落していても、その論理ボリュームをアクティブ化できます。
- partial - LVMは、一部の物理ボリュームが欠落していても、ボリュームグループのアクティブ化を試みます。非冗長論理ボリュームから重要な物理ボリュームが欠落している場合、通常、その論理ボリュームはアクティブ化できず、エラーターゲットとして扱われます。

vgchange -a [ey|n] VG_NAME

ボリュームグループおよびその論理ボリュームを入出力用にアクティブ(-a ey)または非アクティブ(-a n)にします。

クラスタ内のボリュームをアクティブ化する場合は、必ず ey オプションを使用してください。ロードスクリプトではこのオプションがデフォルトで使用されます。

vgremove VG_NAME

ボリュームグループを削除します。このコマンドを使用する前に、論理ボリュームを削除してボリュームグループを非アクティブにしてください。

vgdisplay VG_NAME

指定したボリュームグループに関する情報を表示します。

ボリュームグループの合計物理エクステントを確認するには、次のように入力します。

```
tux > vgdisplay VG_NAME | grep "Total PE"
```

lvcreate -L SIZE -n LV_NAME VG_NAME

指定したサイズの論理ボリュームを作成します。

lvcreate -L SIZE --thinpool POOL_NAME VG_NAME

ボリュームグループ VG_NAME から、指定したサイズのシンプールの myPool を作成します。

次の例では、ボリュームグループ LOCAL から5GBのサイズのシンプールの作成します。

```
tux > sudo lvcreate -L 5G --thinpool myPool LOCAL
```

lvcreate -T VG_NAME/POOL_NAME -V SIZE -n LV_NAME

プール POOL_NAME 内にシン論理ボリュームを作成します。次の例では、ボリュームグループ LOCAL 上のプール myPool から1GBのシンボリューム myThin1 を作成します。

```
tux > sudo lvcreate -T LOCAL/myPool -V 1G -n myThin1
```

lvcreate -T VG_NAME/POOL_NAME -V SIZE -L SIZE -n LV_NAME

シンプールの作成とシン論理ボリュームの作成を1つのコマンドに結合することもできます。

```
tux > sudo lvcreate -T LOCAL/myPool -V 1G -L 5G -n myThin1
```

lvcreate --activationmode ACTIVATION_MODE LV_NAME

論理ボリュームのアクティブ化のモードを設定します。次のいずれかの値を指定できます。

- complete - 論理ボリュームは、そのすべての物理ボリュームがアクティブな場合にのみアクティブ化できます。
- degraded - デフォルトのアクティブ化モードです。論理ボリュームをアクティブ化するための十分なレベルの冗長性がある場合、一部の物理ボリュームが欠落していても、その論理ボリュームをアクティブ化できます。
- partial - LVMは、一部の物理ボリュームが欠落していても、ボリュームのアクティブ化を試みます。この場合、論理ボリュームの一部が使用できなくなり、データが消失することがあります。このオプションは通常は使用しませんが、データを復元する場合に役立つことがあります。

activation_mode 設定オプションの上記いずれかの値を指定することによって、/etc/lvm/lvm.conf でアクティブ化モードを指定することもできます。

lvcreate -s [-L SIZE] -n SNAP_VOLUME SOURCE_VOLUME_PATH VG_NAME

指定した論理ボリュームに対してスナップショットボリュームを作成します。サイズオプション(-Lまたは--size)を指定しなかった場合、スナップショットはシンスナップショットとして作成されます。

lvremove /dev/VG_NAME/LV_NAME

論理ボリュームを削除します。

このコマンドを使用する前に、論理ボリュームを umount コマンドでアンマウントして閉じてください。

lvremove SNAP_VOLUME_PATH

スナップショットボリュームを削除します。

lvconvert --merge SNAP_VOLUME_PATH

論理ボリュームをスナップショットのバージョンに戻します。

vgextend VG_NAME DEVICE

指定したデバイス(物理ボリューム)を既存のボリュームグループに追加します。

vgreduce VG_NAME DEVICE

指定した物理ボリュームを既存のボリュームグループから削除します。

物理ボリュームが論理ボリュームによって使用中でないことを確認してください。使用中の場合は、pvmove コマンドを使用してデータを別の物理ボリュームに移動する必要があります。

lvextend -L SIZE /dev/VG_NAME/LV_NAME

指定した論理ボリュームのサイズを拡張します。その後、新たに使用可能になった領域を使用するため、ファイルシステムを拡張する必要もあります。詳細については、[第2章「ファイルシステムのサイズ変更」](#)を参照してください。

lvreduce -L SIZE /dev/VG_NAME/LV_NAME

指定した論理ボリュームのサイズを縮小します。

ボリュームを縮小する前に、まずファイルシステムのサイズを縮小してください。そうしないと、データを失うリスクがあります。詳細については、[第2章「ファイルシステムのサイズ変更」](#)を参照してください。

lvrename /dev/VG_NAME/LV_NAME /dev/VG_NAME/NEW_LV_NAME

既存のLVM論理ボリュームの名前を変更します。ボリュームグループの名前は変更されません。



ヒント: ボリューム作成時のudevのバイパス

udevルールではなくLVMを使用してLVデバイスノードとシンボリックリンクを管理する場合は、次のいずれかの方法でudevからの通知を無効にすることによって可能になります。

- `/etc/lvm/lvm.conf` で `activation/udev_rules = 0` および `activation/udev_sync = 0` を設定する。
`lvcreate` コマンドで `--nodevsysync` を指定しても、`activation/udev_sync = 0` と同じ結果になります。この場合も、`activation/udev_rules = 0` の設定が必要です。
- 環境変数 `DM_DISABLE_UDEV` を設定する。

```
export DM_DISABLE_UDEV=1
```

この方法でも、udevからの通知が無効になります。さらに、`/etc/lvm/lvm.conf` のudev関連の設定はすべて無視されます。

5.8.1 コマンドによる論理ボリュームのサイズ変更

論理ボリュームのサイズ変更には、コマンド `lvresize`、`lvextend`、および `lvreduce` が使用されます。構文とオプションについては、これらの各コマンドのマニュアルページを参照してください。LVを拡大するには、VG上に十分な未使用スペースがなければなりません。

論理ボリュームを拡大または縮小する場合、YaSTパーティショナを使用することをお勧めします。YaSTを使用すると、そのボリュームのファイルシステムのサイズも自動的に調整されます。

LVは使用中に手動で拡大または縮小できますが、LV上のファイルシステムについてはこれが不可能な場合があります。LVを拡大、縮小しても、そのボリューム内のファイルシステムのサイズは自動的に変更されません。後でファイルシステムを拡大するには、別のコマンドを使用する必要があります。ファイルシステムのサイズ変更の詳細については、[第2章「ファイルシステムのサイズ変更」](#)を参照してください。

手動でLVのサイズを変更する場合は、次に示すように正しい順序に従ってください。

- LVを拡大する場合は、ファイルシステムを拡大する前にLVを拡大する必要があります。
- LVを縮小する場合は、LVを縮小する前にファイルシステムを縮小する必要があります。

論理ボリュームのサイズを拡張するには:

1. 端末コンソールを開きます。
2. 論理ボリュームにExt2またはExt4ファイルシステム(オンライン拡張がサポートされていません)が含まれる場合、マウント解除します。仮想マシン(Xen VMなど)用に提供されているファイルシステムが含まれている場合は、最初にVMをシャットダウンします。
3. 端末コンソールのプロンプトに対して、次のコマンドを入力し、論理ボリュームのサイズを拡大します。

```
tux > sudo lvextend -L +SIZE /dev/VG_NAME/LV_NAME
```

SIZE の場合は、10GBのように、論理ボリュームに追加したい容量を指定してください。 /dev/VG_NAME/LV_NAME を、 /dev/LOCAL/DATA などの論理ボリュームへのLinuxパスに入れ替えます。次に例を示します。

```
tux > sudo lvextend -L +10GB /dev/vg1/v1
```

4. ファイルシステムのサイズを調整します。詳細については、[第2章「ファイルシステムのサイズ変更」](#)を参照してください。
5. ファイルシステムをマウント解除した場合は、再びマウントします。

たとえば、LVをLV上の(マウント済みでアクティブな) Btrfsで10GB拡張するには:

```
tux > sudo lvextend -L +10G /dev/LOCAL/DATA  
tux > sudo btrfs filesystem resize +10G /dev/LOCAL/DATA
```

論理ボリュームのサイズを縮小するには:

1. 端末コンソールを開きます。
2. 論理ボリュームにBtrfsファイルが含まれていない場合は、論理ボリュームをマウント解除します。仮想マシン(Xen VMなど)用に提供されているファイルシステムが含まれている場合は、最初にVMをシャットダウンします。XFSファイルシステムを使用しているボリュームのサイズは縮小できません。
3. ファイルシステムのサイズを調整します。詳細については、[第2章「ファイルシステムのサイズ変更」](#)を参照してください。
4. 端末コンソールのプロンプトに対して、次のコマンドを入力し、論理ボリュームのサイズをファイルシステムのサイズまで縮小します。

```
tux > sudo lvreduce /dev/VG_NAME/LV_NAME
```

5. ファイルシステムをアンマウントしてあった場合は、再びマウントします。

たとえば、LVをLV上のBtrfsで5GB縮小するには:

```
tux > sudo btrfs filesystem resize -size 5G /dev/LOCAL/DATA  
sudo lvreduce /dev/LOCAL/DATA
```



ヒント: 1つのコマンドでのボリュームとファイルシステムのサイズ変更

SUSE Linux Enterprise Server 12 SP1から、**lvextend**、**lvresize**、および**lvreduce**で**--resizefs**オプションがサポートされるようになりました。このオプションは、ボリュームのサイズを変更するだけでなく、ファイルシステムのサイズも変更します。したがって、上に示す**lvextend**および**lvreduce**の例は、次のように実行することもできます。

```
tux > sudo lvextend --resizefs -L +10G /dev/LOCAL/DATA  
tux > sudo lvreduce --resizefs -L -5G /dev/LOCAL/DATA
```

--resizefsは、ext2/3/4、Btrfs、およびXFSの各ファイルシステムでサポートされます。このオプションを使用したBtrfsのサイズ変更は、まだ上流では許可されていないため、SUSE Linux Enterprise Serverでのみ可能です。

5.8.2 `lvmetad`によるLVMメタデータの動的集約

ほとんどのLVMコマンドでは、システムのディスクデバイスに保存されているLVMメタデータを正確に把握しておく必要があります。LVMの現行の設計では、この情報がない場合、LVMはシステムのすべての物理ディスクデバイスをスキャンしなければなりません。多数のディスクで構成されるシステムでは、このために大量のI/O操作が必要になります。ディスクが応答しない場合、そのディスクを待機している間にLVMコマンドがタイムアウトすることがあります。

`lvmetad` によるLVMメタデータの動的集約は、この問題に解決策を提供します。`lvmetad` デーモンの目的は、デバイスの状態が変わるたびにメタデータ情報を動的に集約することによって、このスキャンを不要にすることです。これらのイベントはudevルールによって `lvmetad` に送信されます。このデーモンが実行されていない場合、LVMは通常どおりにスキャンを実行します。

この機能はデフォルトで有効になっています。システムでこの機能が無効になっている場合は、次の手順に従って有効にします。

1. 端末コンソールを開きます。
2. `lvmetad` デーモンを停止します。

```
tux > sudo systemctl stop lvm2-lvmetad
```

3. `/etc/lvm/lvm.conf` を編集し、`use_lvmetad` を `1` に設定します。

```
use_lvmetad = 1
```

4. `lvmetad` デーモンを再起動します。

```
tux > sudo systemctl start lvm2-lvmetad
```

5.8.3 LVMキャッシュボリュームの使用

LVMでは、大容量の低速なブロックデバイスに対して、高速なブロックデバイス(SSDデバイスなど)をライトバックキャッシュまたはライトスルーキャッシュとして使用できます。キャッシュ論理ボリュームタイプは、小容量の高速なLVを使用して、大容量の低速なLVのパフォーマンスを向上させます。

LVMキャッシングを設定するには、キャッシングデバイス上に2つの論理ボリュームを作成する必要があります。大容量の論理ボリュームはキャッシング自体に使用され、小容量のボリュームはキャッシングメタデータの保存に使用されます。これら2つのボリュームは、元のボリュームと同じボリュームグループに属している必要があります。これらのボリュームを作成したら、キャッシュプールに変換して元のボリュームに接続する必要があります。

手順 5.2: キャッシュ論理ボリュームの設定

1. 元のボリュームがまだ存在しない場合は(低速なデバイス上に)作成します。
2. 物理ボリュームを(高速なデバイスから)元のボリュームが属するボリュームグループに追加して、物理ボリューム上にキャッシュデータボリュームを作成します。
3. キャッシュメタデータボリュームを作成します。サイズは、キャッシュデータボリュームの1/1000にする必要があります。最小サイズは8MBです。
4. キャッシュデータボリュームとメタデータボリュームをキャッシュプールボリュームに結合します。

```
tux > sudo lvconvert --type cache-pool --poolmetadata VOLUME_GROUP/  
METADATA_VOLUME VOLUME_GROUP/CACHING_VOLUME
```

5. キャッシュプールを元のボリュームに接続します。

```
tux > sudo lvconvert --type cache --cachepool VOLUME_GROUP/  
CACHING_VOLUME VOLUME_GROUP/ORIGINAL_VOLUME
```

LVMキャッシングの詳細については、`lvmdcache(7)`のマニュアルページを参照してください。

5.9 LVM2ストレージオブジェクトへのタグ付け

タグは、ストレージオブジェクトのメタデータに割り当てられる順序付けのないキーワードまたは用語です。タグを使用すると、順序付けのないタグのリストをLVMストレージオブジェクトのメタデータに添付することによって、それらのオブジェクトのコレクションを有用になるように分類できます。

5.9.1 LVM2タグの使用

LVM2ストレージオブジェクトにタグを付けたら、それらのタグをコマンドで使用して、次のタスクを達成できます。

- 特定のタグの有無に応じて、処理するLVMオブジェクトを選択します。
- 設定ファイル内でタグを使用することにより、サーバ上でアクティブにするボリュームグループと論理ボリュームを制御します。
- コマンド内でタグを指定することにより、グローバル設定ファイルの設定を上書きします。

コマンドラインでLVMオブジェクトを参照する代わりに、タグを使用して、次の項目を受け入れることができます。

- オブジェクトのリスト
- 単一のオブジェクト(タグが単一オブジェクトに展開する限り)

オブジェクト名をタグで置き換えることは、一部ではサポートされていません。引数の展開後、リスト内の重複引数は、重複引数を削除し、各引数の最初のインスタンスを保留することによって解決されます。

引数のタイプが曖昧になる可能性がある場合は、タグの前にアットマーク(@)文字を付けてください(たとえば、@mytag)。それ以外の接頭辞「@」の使用はオプションです。

5.9.2 LVM2タグの作成要件

LVMでタグを使用する場合は、以下の要件を考慮してください。

サポートされている文字

LVMタグのワードには、ASCII 大文字A～Z、小文字a～z、数字0～9、下線(_)、プラス(+)、ハイフン(-)、およびピリオド(.)を含めることができます。ワードをハイフンで始めることはできません。最大128文字まで入力できます。

サポートされているストレージオブジェクト

タグ付けできるのは、LVM2の物理ボリューム、ボリュームグループ、論理ボリューム、および論理ボリュームセグメントです。PVタグは、そのボリュームグループのメタデータに保存されます。ボリュームグループを削除すると、孤立した物理ボリューム内のタグも削除されます。スナップショットにはタグを付けられませんが、元のオブジェクトはタグ付けできます。

LVM1オブジェクトは、そのディスクフォーマットがタグをサポートしていないので、タグ付けできません。

5.9.3 コマンドラインでのタグ構文

--addtag TAG_INFO

LVM2ストレージオブジェクトにタグを追加(つまり、タグ付け)します。例:

```
tux > sudo vgchange --addtag @db1 vg1
```

--deltag TAG_INFO

LVM2ストレージオブジェクトからタグを削除(つまり、タグ解除)します。例:

```
tux > sudo vgchange --deltag @db1 vg1
```

--tag TAG_INFO

アクティブまたは非アクティブにするボリュームグループまたは論理ボリュームのリストを絞り込むために使用するタグを指定します。

次の例に示すコマンドを入力すると、指定のタグに一致するタグをもつボリュームがアクティブになります。

```
tux > sudo lvchange -ay --tag @db1 vg1/vol2
```

5.9.4 設定ファイル構文

以降の各項では、特定の事例における設定例を示します。

5.9.4.1 lvmt.confファイルでのホスト名タグの有効化

次のコードを /etc/lvm/lvm.conf ファイルに追加することにより、/etc/lvm/lvm_<HOSTNAME>.conf ファイルでホストに個別に定義されているホストタグを有効にします。

```
tags {  
    # Enable hostname tags  
    hosttags = 1  
}
```

ホストの /etc/lvm/lvm_<HOSTNAME>.conf ファイルにアクティベーションコードを入力します。詳細については、[5.9.4.3項「アクティベーションを定義する」](#)を参照してください。

5.9.4.2 lvmt.confファイルでホスト名タグを定義する

```
tags {
```

```

tag1 { }
    # Tag does not require a match to be set.

tag2 {
    # If no exact match, tag is not set.
    host_list = [ "hostname1", "hostname2" ]
}
}

```

5.9.4.3 アクティベーションを定義する

/etc/lvm/lvm.conf ファイルを変更すると、タグに基づいてLVM論理ボリュームをアクティブにできます。

テキストエディタで、次のコードをファイルに追加します。

```

activation {
    volume_list = [ "vg1/lvol0", "@database" ]
}

```

@database をご使用のタグで置き換えます。ホストに設定されているすべてのタグにタグを一致させるには、"@" を使用します。

アクティベーションコマンドは、ボリュームグループと論理ボリュームのメタデータで設定されている VGNAME、VGNAME/LVNAME、または @TAG と照合を行います。ボリュームグループまたは論理グループは、メタデータタグが一致する場合のみアクティブになります。一致しない場合、デフォルトではアクティブになりません。

volume_list が存在せず、ホストにタグが定義されていると、ホストタグがメタデータタグに一致する場合のみボリュームグループまたは論理グループがアクティブになります。

volume_list が定義されていても空であり、ホストにタグが定義されていないと、アクティブになりません。

volume_list が定義されていないと、LVのアクティブ化に制限は課されません(すべて許可されます)。

5.9.4.4 複数のホスト名設定ファイルでアクティベーションを定義する

lvm.conf ファイルでホストタグが有効になっている場合、ホストの設定ファイル(/etc/lvm/lvm_<HOST_TAG>.conf)でアクティベーションコードを使用できます。たとえば、サーバの /etc/lvm/ ディレクトリに、2つの設定ファイルがあるとします。

lvm.conf

lvm_<HOST_TAG>.conf

スタートアップ時に、/etc/lvm/lvm.conf ファイルがロードされ、ファイル内のすべてのタグ設定が処理されます。ホストタグが定義されている場合、関連する /etc/lvm/lvm_<HOST_TAG>.conf ファイルがロードされます。特定の設定ファイルエントリを検索する際、最初にホストタグファイルが検索されます。続いて lvm.conf ファイルが検索され、最初に一致した箇所で停止します。lvm_<HOST_TAG>.conf ファイル内で、タグが設定された順序とは逆の順序を使用します。これによって、最後に設定されたタグのファイルが最初に検索されます。ホストタグファイルで新しいタグが設定されると、追加の設定ファイルがロードされます。

5.9.5 クラスタで簡単なアクティベーション制御にタグを使用する

簡単なホスト名のアクティベーション制御は、/etc/lvm/lvm.conf ファイルで hostname_tags オプションを有効にすることで設定できます。これがグローバル設定になるように、同じファイルをクラスタ内のすべてのコンピュータで使用します。

1. テキストエディタで、次のコードを /etc/lvm/lvm.conf ファイルに追加します。

```
tags {
    hostname_tags = 1
}
```

2. ファイルをクラスタ内のすべてのホストに複製します。
3. クラスタ内の任意のコンピュータから、vg1/lvol2 をアクティブにするコンピュータのリストに db1 を追加します。

```
tux > sudo lvchange --addtag @db1 vg1/lvol2
```

4. db1 サーバで、次のコードを入力して vg1/lvol2 をアクティブにします。

```
tux > sudo lvchange -ay vg1/vol2
```

5.9.6 タグを使用して、クラスタ内の好みのホストでアクティブにする

本項の例では、次のようなアクティベーションを行う2つの方法を示します。

- ボリュームグループ vg1 をデータベースホスト db1 および db2 でのみアクティブにします。
- ボリュームグループ vg2 をファイルサーバホスト fs1 のみでアクティブにします。
- ファイルサーバのバックアップホスト fsb1 では、最初は何もアクティブにせず、ファイルサーバのホスト fs1 に置き換わる準備をします。

5.9.6.1 オプション1:一元化された管理とホスト間で複製された設定

次のソリューションでは、単一の設定ファイルを複数のホスト間で複製します。

1. @database タグをボリュームグループ vg1 のメタデータに追加します。端末コンソールで、次のコマンドを入力します。

```
tux > sudo vgchange --addtag @database vg1
```

2. @fileserver タグをボリュームグループ vg2 のメタデータに追加します。端末コンソールで、次のコマンドを入力します。

```
tux > sudo vgchange --addtag @fileserver vg2
```

3. テキストエディタで、次のコードを使用して /etc/lvm/lvm.conf を変更することにより、@database、@fileserver、@fileserverbackup の各タグを定義します。

```
tags {
    database {
        host_list = [ "db1", "db2" ]
    }
    fileserver {
        host_list = [ "fs1" ]
    }
    fileserverbackup {
        host_list = [ "fsb1" ]
    }
}

activation {
    # Activate only if host has a tag that matches a metadata tag
    volume_list = [ "@*" ]
}
```

4. 変更した /etc/lvm/lvm.conf ファイルを4つのホスト(db1、db2、fs1、および fsb1)に複製します。

5. ファイルサーバホストが故障した場合は、次のコマンドを任意のモードで端末コンソールから入力することにより、fsb1上でvg2を起動できます。

```
tux > sudo vgchange --addtag @fileserverbackup vg2
tux > sudo vgchange -ay vg2
```

5.9.6.2 オプション2:ローカライズされた管理と設定

次のソリューションでは、各ホストがアクティブにするボリュームのクラスに関する情報をローカルに保持します。

1. @database タグをボリュームグループ vg1 のメタデータに追加します。端末コンソールで、次のコマンドを入力します。

```
tux > sudo vgchange --addtag @database vg1
```

2. @fileserver タグをボリュームグループ vg2 のメタデータに追加します。端末コンソールで、次のコマンドを入力します。

```
tux > sudo vgchange --addtag @fileserver vg2
```

3. /etc/lvm/lvm.conf ファイルでホストタグを有効にします。

- a. テキストエディタで、次のコードを使用して /etc/lvm/lvm.conf ファイルを変更することにより、ホストタグ設定ファイルを有効にします。

```
tags {
    hosttags = 1
}
```

- b. 変更した /etc/lvm/lvm.conf ファイルを4つのホスト(db1、db2、fs1、および fsb1)に複製します。

4. ホスト db1 で、データベースホスト db1 のアクティベーション設定ファイルを作成します。テキストエディタで、/etc/lvm/lvm_db1.conf ファイルを作成し、次のコードを追加します。

```
activation {
    volume_list = [ "@database" ]
}
```

5. ホスト db2 で、データベースホスト db2 のアクティベーション設定ファイルを作成します。テキストエディタで、/etc/lvm/lvm_db2.conf ファイルを作成し、次のコードを追加します。

```
activation {  
    volume_list = [ "@database" ]  
}
```

6. ホスト fs1 で、ファイルサーバホスト fs1 のアクティベーション設定ファイルを作成します。テキストエディタで、/etc/lvm/lvm_fs1.conf ファイルを作成し、次のコードを追加します。

```
activation {  
    volume_list = [ "@fileserver" ]  
}
```

7. ファイルサーバホスト fs1 が故障した場合は、スペアのファイルサーバホスト fsb1 をファイルサーバとして起動します。

- a. ホスト fsb1 で、ホスト fsb1 のアクティベーション設定ファイルを作成します。テキストエディタで、/etc/lvm/lvm_fsb1.conf ファイルを作成し、次のコードを追加します。

```
activation {  
    volume_list = [ "@fileserver" ]  
}
```

- b. 端末コンソールで、次のコマンドの1つを入力します。

```
tux > sudo vgchange -ay vg2  
tux > sudo vgchange -ay @fileserver
```

6 LVMボリュームスナップショット

LVM (Logical Volume Manager)論理ボリュームスナップショットはコピーオンライト技術の1つで、既存のボリュームのデータブロックに対する変更を監視し、いずれかのブロックに書き込みが行われると、スナップショット時のブロックの値がスナップショットボリュームにコピーされます。こうすることで、スナップショットボリュームが削除されるまで、データのその時点のコピーが保存されます。

6.1 ボリュームスナップショットの理解

ファイルシステムのスナップショットには、それ自体のメタデータと、スナップショットの作成後に変更されたソース論理ボリュームのデータブロックが含まれています。スナップショットを介してデータにアクセスすると、ソース論理ボリュームのその時点のコピーが表示されます。バックアップ媒体からデータを復元したり、変更されたデータを上書きする必要はありません。



重要: スナップショットによるボリュームのマウント

スナップショットのライフタイム中は、スナップショットを先にマウントしないと、ソース論理ボリュームをマウントできません。

LVMボリュームスナップショットでは、ファイルシステムのその時点のビューからバックアップを作成できます。スナップショットは瞬時に作成され、削除するまで保存されます。ボリューム自体はユーザが引き続き利用できるようにしながら、スナップショットからファイルシステムのバックアップを作成できます。当初のスナップショットには、スナップショットに関するメタデータが含まれていますが、ソース論理ボリュームの実際のデータは含まれていません。スナップショットはコピーオンライト技術を使用して、オリジナルデータブロックのデータ変更を検出します。スナップショットをとった際に保存されていた値をスナップショットボリューム内のブロックにコピーし、ソースブロックに新しいデータを保存することができます。ソース論理ボリュームで元の値から変更されるブロックが増えると、スナップショットのサイズが増えます。

スナップショットのサイズを決定する際には、ソース論理ボリュームに対して予想されるデータ変更量、およびスナップショットの保存期間を考慮する必要があります。スナップショットボリュームに割り当てるスペースの量は、ソース論理ボリュームのサイズ、スナップショットの保持予定期間、およびスナップショットのライフタイム中に変更が予期されるデータブロックの数によって異なります。スナップショットボリュームは、作成後のサイズ変更はできません。目安として、元の論理ボリュームの約10%のサイズで、スナップショットボ

リユームを作成してください。スナップショットの削除前に、ソース論理ボリユーム内のすべてのブロックが1回以上変更されると予期される場合は、スナップボリユームのサイズを、少なくともソース論理ボリユームサイズにそのボリユームに関するメタデータ用スペースを加えたサイズにする必要があります。データ変更が頻繁でないか、またはライフタイムが十分短いと予期される場合、必要なスペースは少なくなります。

LVM2では、スナップショットはデフォルトで読み書き可能です。データをスナップショットに直接書き込む際は、そのブロックは例外テーブルで使用中とマークされ、ソース論理ボリユームからのコピーは行われません。スナップショットボリユームをマウントし、そのスナップショットボリユームにデータを直接書き込むことによって、アプリケーションの変更をテストできます。スナップショットをマウント解除してスナップショットを削除し、ソース論理ボリユームを再マウントするだけで、変更を簡単に破棄できます。

仮想ゲスト環境では、物理サーバの場合と同様に、サーバのディスク上に作成するLVM論理ボリユームに対してスナップショット機能を使用できます。

仮想ホスト環境では、スナップショット機能を使用して、仮想マシンのストレージバックエンドをバックアップしたり、仮想マシンイメージに対する変更(パッチやアップグレードなど)を、ソース論理ボリユームを変更せずにテストしたりできます。仮想マシンは、仮想ディスクファイルの使用ではなく、ストレージバックエンドとして、LVM論理ボリユームを使用する必要があります。LVM論理ボリユームをマウントし、ファイルに格納されたディスクとして仮想マシンイメージを保存するために使用できます。また、そのLVM論理ボリユームを物理ディスクとして割り当てて、ブロックデバイスとして書き込むことができます。

SLES 11 SP3から、LVM論理ボリユームスナップショットはシンプロビジョニング可能になっています。サイズを指定しないでスナップショットを作成した場合は、シンプロビジョニングと想定されます。スナップショットは、シンプールから必要な領域を使用するシンボリユームとして作成されます。シンスナップショットボリユームは、他のシンボリユームと同じ特性を持ちます。ボリユームは個別にアクティブ化、拡張、名前変更、および削除でき、そのスナップショットを作成することもできます。

！ 重要: クラスタにおけるシンプロビジョニングボリユーム

クラスタでシンプロビジョニングスナップショットを使用するには、ソース論理ボリユームとそのスナップショットを1つのクラスタリソースで管理する必要があります。これにより、ボリユームとそのスナップショットを常に同じノードに排他的にマウントできます。

スナップショットが不要になったら、必ず、システムからスナップショットを削除してください。ソース論理ボリユームでデータブロックが変化していくのに応じて、スナップショットは最終的に満杯になります。スナップショットは満杯になると使用不可になるので、ソース論理ボリユームの再マウントができなくなります。

ソース論理ボリュームのスナップショットを複数作成している場合、スナップショットの削除は、最後に作成したものを最初に削除するという順番で行います。

6.2 LVMによるLinuxスナップショットの作成

LVM (Logical Volume Manager)は、ファイルシステムのスナップショットの作成に使用できます。

端末コンソールを開いて、次のコマンドを入力します。

```
tux > sudo lvcreate -s [-L <size>] -n SNAP_VOLUME SOURCE_VOLUME_PATH
```

サイズを指定しない場合、スナップショットはシンスナップショットとして作成されます。次に例を示します。

```
tux > sudo lvcreate -s -L 1G -n linux01-snap /dev/lvm/linux01
```

スナップショットが /dev/lvm/linux01-snap ボリュームとして作成されます。

6.3 スナップショットの監視

端末コンソールを開いて、次のコマンドを入力します。

```
tux > sudo lvdisplay SNAP_VOLUME
```

次に例を示します。

```
tux > sudo lvdisplay /dev/vg01/linux01-snap

--- Logical volume ---
LV Name                /dev/lvm/linux01
VG Name                vg01
LV UUID                QHVJYh-PR3s-A4SG-s4Aa-MyWN-Ra7a-HL47KL
LV Write Access        read/write
LV snapshot status     active destination for /dev/lvm/linux01
LV Status              available
# open                 0
LV Size                80.00 GB
Current LE             1024
COW-table size         8.00 GB
COW-table LE           512
Allocated to snapshot  30%
Snapshot chunk size    8.00 KB
Segments               1
Allocation              inherit
Read ahead sectors     0
```

6.4 Linuxスナップショットの削除

端末コンソールを開いて、次のコマンドを入力します。

```
tux > sudo lvremove SNAP_VOLUME_PATH
```

次に例を示します。

```
tux > sudo lvremove /dev/lvmvg/linux01-snap
```

6.5 仮想ホスト上の仮想マシンに対するスナップショットの使用

仮想マシンのバックエンドストレージにLVM論理ボリュームを使用すると、基礎となるデバイスを柔軟に管理でき、ストレージオブジェクトの移動、スナップショットの作成、データのバックアップなどの操作を容易に行うことができます。LVM論理ボリュームをマウントし、ファイルに格納されたディスクとして仮想マシンイメージを保存するために使用できます。また、そのLVM論理ボリュームを物理ディスクとして割り当て、ブロックデバイスとして書き込むことができます。LVM論理ボリューム上に仮想ディスクイメージを作成してから、LVMのスナップショットを作成できます。

スナップショットの読み込み/書き込み機能を利用して、1つの仮想マシンのインスタンスを複数作成できます。この場合、変更は、仮想マシンの特定のインスタンスのスナップショットに対して行われます。LVM論理ボリューム上に仮想ディスクイメージを作成してソース論理ボリュームのスナップショットを作成し、仮想マシンの特定のインスタンスのスナップショットを変更できます。ソース論理ボリュームのスナップショットをもう1つ作成して、仮想マシンの別のインスタンス用に変更できます。複数の仮想マシンインスタンスのデータの大部分は、ソース論理ボリューム上のイメージと共に存在します。

スナップショットの読み込み/書き込み機能を利用すると、仮想ディスクイメージを維持したまま、ゲスト環境でパッチやアップグレードをテストすることもできます。そのイメージが含まれるLVMボリュームのスナップショットを作成し、そのスナップショットの場所で仮想マシンを実行します。ソース論理ボリュームは変更されず、そのマシンに対する変更はすべてスナップショットに書き込まれます。仮想マシンイメージのソース論理ボリュームに戻るには、仮想マシンの電源をオフにした後、ソース論理ボリュームからスナップショットを削除します。もう一度やり直すには、スナップショットを再作成してマウントしてから、スナップショットイメージ上で仮想マシンを再起動します。

次の手順では、ファイルに格納された仮想ディスクイメージとXenハイパーバイザを使用します。本項の手順は、KVMなど、SUSE Linux Enterpriseプラットフォーム上で動作する他のハイパーバイザに適用できます。スナップショットボリュームからファイルに格納された仮想マシンイメージを実行するには:

1. ファイルに格納された仮想マシンイメージが含まれるソース論理ボリュームがマウントされていることを確認します(たとえば、マウントポイント /var/lib/xen/images/<IMAGE_NAME>)。
2. 予想される差分を保存するのに十分な領域があるLVM論理ボリュームのスナップショットを作成します。

```
tux > sudo lvcreate -s -L 20G -n myvm-snap /dev/lvmvg/myvm
```

サイズを指定しない場合、スナップショットはシンスナップショットとして作成されます。

3. スナップショットボリュームをマウントするマウントポイントを作成します。

```
tux > sudo mkdir -p /mnt/xen/vm/myvm-snap
```

4. 作成したマウントポイントにスナップショットボリュームをマウントします。

```
tux > sudo mount -t auto /dev/lvmvg/myvm-snap /mnt/xen/vm/myvm-snap
```

5. テキストエディタで、ソース仮想マシンの設定ファイルをコピーし、マウントしたスナップショットボリューム上の、ファイルに格納されたイメージファイルを指すようにパスを変更し、ファイルを /etc/xen/myvm-snap.cfg などの名前で作成します。
6. 仮想マシンのマウント済みスナップショットボリュームを使用して、仮想マシンを起動します。

```
tux > sudo xm create -c /etc/xen/myvm-snap.cfg
```

7. (オプション)スナップショットを削除して、ソース論理ボリューム上の変更されていない仮想マシンイメージを使用します。

```
tux > sudo umount /mnt/xenvms/myvm-snap  
tux > sudo lvremove -f /dev/lvmvg/mylvm-snap
```

8. (オプション)このプロセスを必要なだけ繰り返します。

6.6 スナップショットをソース論理ボリュームとマージして変更を元に戻すか、前の状態にロールバックする

スナップショットは、ボリューム上のデータを前の状態にロールバックまたは復元する必要がある場合に役立ちます。たとえば、管理者の手違いがあった場合、またはパッケージのインストールやアップグレードが失敗したり望む内容と違ったりした場合、データ変更を元に戻さなければならないことがあります。

lvconvert --merge コマンドを使用して、LVM論理ボリュームの変更を元に戻すことができます。マージは次のように開始されます。

- ソース論理ボリュームとスナップショットボリュームが両方とも開かれていない場合、マージはすぐに開始されます。
- ソース論理ボリュームまたはスナップショットボリュームが開かれていない場合、初めてソース論理ボリュームまたはスナップショットボリュームのどちらかがアクティブになって両方が閉じられた時点でマージが開始されます。
- ルートファイルシステムのように、閉じることができないソース論理ボリュームの場合、次にサーバが再起動されてソース論理ボリュームがアクティブになるときまで、マージは延期されます。
- ソース論理ボリュームに仮想マシンイメージが含まれる場合、仮想マシンをシャットダウンしてソース論理ボリュームとスナップショットボリュームを(この順序でマウント解除することによって)非アクティブにした後、mergeコマンドを発行する必要があります。マージ完了時に、ソース論理ボリュームが自動的に再マウントされてスナップショットボリュームは削除されるので、マージ完了後まで仮想マシンを再起動しないでください。マージが完了した後、生成された論理ボリュームを仮想マシンで使用します。

マージが開始されると、サーバ再起動後もマージは自動的に続行され、これはマージが完了するまで続きます。マージの進行中は、マージ中のソース論理ボリュームの新しいスナップショットを作成することはできません。

マージの進行中は、ソース論理ボリュームに対する読み込みまたは書き込みは、マージ中のスナップショットに透過的にリダイレクトされます。これにより、ユーザは直接、スナップショット作成時のデータを表示したり、そのデータにアクセスしたりできます。マージが完了するまで待つ必要はありません。

マージが完了すると、ソース論理ボリュームにはスナップショット作成時と同じデータと、マージ開始後に行われたデータ変更がすべて含まれます。生成された論理ボリュームの名前、マイナー番号、およびUUIDは、ソース論理ボリュームと同じです。ソース論理ボリュームは自動的に再マウントされ、スナップショットボリュームは削除されます。

1. 端末コンソールを開いて、次のコマンドを入力します。

```
tux > sudo lvconvert --merge [-b] [-i SECONDS] [SNAP_VOLUME_PATH[...snapN]|@VOLUME_TAG]
```

コマンドラインで1つ以上のスナップショットを指定できます。または、複数のソース論理ボリュームに同じボリュームタグを設定し、「@<VOLUME_TAG>」と指定することもできます。タグ付きボリュームのスナップショットは、それぞれのソース論理ボリュームにマージされます。タグ付き論理ボリュームについては、[5.9項「LVM2ストレージオブジェクトへのタグ付け」](#)を参照してください。

次のオプションがあります。

-b,

--background

デーモンをバックグラウンドで実行します。これにより、指定した複数のスナップショットのマージを同時に並行して実行できます。

-i,

--interval <SECONDS>

進行状況を定期的にパーセント値でレポートします。間隔を秒単位で指定します。

このコマンドの詳細については、[lvconvert\(8\)](#)のマニュアルページを参照してください。

次に例を示します。

```
tux > sudo lvconvert --merge /dev/lvmvg/linux01-snap
```

このコマンドは、[/dev/lvmvg/linux01-snap](#)をそのソース論理ボリュームにマージします。

```
tux > sudo lvconvert --merge @mytag
```

lvol1、lvol2、および lvol3 すべてにタグ mytag が付いている場合、各スナップショットは対応するソース論理ボリュームに順番にマージされます。すなわち、lvol1、lvol2、lvol3 の順にマージされます。 --background オプションが指定されている場合、各タグ付きソース論理ボリュームのスナップショットは同時に並行してマージされます。

2. (オプション)ソース論理ボリュームとスナップショットが両方とも開いていて、閉じることができる場合、手動でソース論理ボリュームを非アクティブにしてからアクティブにすることによって、すぐにマージを開始できます。

```
tux > sudo umount ORIGINAL_VOLUME
tux > sudo lvchange -an ORIGINAL_VOLUME
tux > sudo lvchange -ay ORIGINAL_VOLUME
tux > sudo mount ORIGINAL_VOLUME MOUNT_POINT
```

次に例を示します。

```
tux > sudo umount /dev/lvmvg/lvol01
tux > sudo lvchange -an /dev/lvmvg/lvol01
tux > sudo lvchange -ay /dev/lvmvg/lvol01
tux > sudo mount /dev/lvmvg/lvol01 /mnt/lvol01
```

3. (オプション)ソース論理ボリュームとスナップショットボリュームが両方とも開いていて、ソース論理ボリュームを閉じることができない場合(root ファイルシステムなど)、サーバを再起動してソース論理ボリュームをマウントすることによって、再起動後すぐにマージを開始できます。

III ソフトウェアRAID

- 7 ソフトウェアRAIDの設定 89
- 8 ルートパーティション用のソフトウェアRAIDの設定 98
- 9 ソフトウェアRAID 10デバイスの作成 103
- 10 ディグレードアレイの作成 118
- 11 mdadmによるソフトウェアRAIDアレイのサイズ変更 120
- 12 MDソフトウェアRAID用のストレージエンクロージャLEDユーティリティ 129

7 ソフトウェアRAIDの設定

RAID (Redundant Array of Independent Disks)の目的は、複数のハードディスクパーティションを1つの大きい仮想ハードディスクに結合し、パフォーマンスとデータのセキュリティを最適化することです。ほとんどのRAIDコントローラはSCSIプロトコルを使用します。これは、IDEプロトコルも効率的な方法で多数のハードディスクのアドレスを指定でき、コマンドの平行処理に適しているからです。一方、IDEまたはSATAハードディスクをサポートしているRAIDコントローラもあります。ソフトウェアRAIDは、ハードウェアRAIDコントローラ購入による追加コストなしで、RAIDシステムの利点を提供します。ただし、これにはいくらかのCPU時間を要し、高性能なコンピュータには適さないメモリ要件があります。

！ 重要: クラスタファイルシステムのRAID

クラスタファイルシステムのソフトウェアRAIDはクラスタマルチデバイス(Cluster MD)を使用して設定する必要があります。https://www.suse.com/documentation/sle-ha-12/book_sleha/data/cha_ha_cluster-md.htmlにある高可用性拡張機能のドキュメントを参照してください。

SUSE Linux Enterpriseには、いくつかのハードディスクを1つのソフトウェアRAIDシステムに統合するオプションがあります。RAIDには、それぞれが異なる目標、利点、および属性をもついくつかのハードディスクを1つのRAIDシステムに結合するためのいくつかの戦略が含まれています。これらは通常、RAIDレベルと呼ばれます。

7.1 RAIDレベルの理解

本項では、通常のRAIDレベル(0、1、2、3、4、5)とネストしたRAIDレベルについて説明します。

7.1.1 RAID 0

このレベルでは、各ファイルのブロックが複数のディスクに分散されるので、データアクセスのパフォーマンスが向上します。このレベルはデータのバックアップを提供しないため、実際にはRAIDではありませんが、この種のシステムでは「RAID 0」という名前が一般的です。RAID 0では、2つ以上のハードディスクが互いにプールします。高いパフォーマンスが得られます。ただし、1つのハードディスクに障害が発生しただけで、RAIDシステムが破壊され、データは失われます。

7.1.2 RAID 1

このレベルでは、データが他のハードディスクに一对一でコピーされるため、データに対する適切なセキュリティが提供されます。これは、ハードディスクミラーリングとして知られています。ディスクが破壊された場合は、ディスクの内容のコピーをミラー先のもう1つのディスクで利用できます。したがって、1つのディスク以外のすべてのディスクが損なわれても、データを保全できます。ただし、損傷が検出されない場合は、正しいディスクに損傷したデータがミラーリングされる可能性があり、その場合はデータが壊れます。単一ディスクアクセスの使用時と比較すると、コピープロセスで書き込みのパフォーマンスが若干低下しますが(10～20%遅くなる)、読み取りアクセスは、通常の物理ハードディスクのどれと比べても、著しく高速です。これは、データが複製されているので、それらを並行してスキャンできるためです。RAID 1では、一般に、読み取りトランザクションの速度が単一ディスクのほぼ2倍、書き込みトランザクションの速度が単一ディスクと同じです。

7.1.3 RAID 2およびRAID 3

これらは、一般的なRAID実装ではありません。レベル2では、データは、ブロックレベルではなく、ビットレベルでストライプ化されます。レベル3は、専用パリティディスクによってバイトレベルのストライプ化を提供しますが、複数の要求を同時にサービスすることはできません。両レベルとも、まれにしか使用されません。

7.1.4 RAID 4

レベル4は、専用パリティディスクと結合されたレベル0と同様に、ブロックレベルのストライピングを提供します。データディスクがエラーになると、パリティデータで置き換え用のディスクが作成されます。ただし、パリティディスクは、書き込みアクセスにボトルネックを生成する可能性があります。にもかかわらず、レベル4は時々使用されます。

7.1.5 RAID 5

RAID 5は、レベル0とレベル1の間をパフォーマンスおよび冗長性の面で調整して、最適化したものです。ハードディスクスペースは、使用されるディスク数から1を引いたものに等しくなります。データは、RAID 0の場合のようにハードディスク間で分散されます。パーティションの1つで作成されたパリティブロックがあるのは、セキュリティ上の理由からです。各パーティションはXORによって互いにリンクされているので、システム障害の場合に、内容

が対応するパリティブロックによって再構築されます。RAID 5の場合、同時に複数のハードディスクが障害を起こすことはありません。1つのハードディスクに障害がある場合は、可能であればそのハードディスクを交換して、データ消失の危険性をなくす必要があります。

7.1.6 RAID 6

RAID 6は、RAID 5の拡張であり、2つ目の独立した分散パリティスキーム(デュアルパリティ)の使用により、耐障害性をさらに追加します。データ回復プロセスで、2つのハードディスクに障害が発生しても、システムは稼動し続け、データが失われることはありません。

RAID 6は、複数の同時ドライブエラーに耐えることで、非常に高いデータ耐障害性を提供します。RAID 6は、データを失うことなく、2つのデバイスの喪失を処理します。したがって、N個のドライブのデータを保存するには、N+2個のドライブが必要です。その結果、最低限4個のデバイスが必要となります。

通常モードおよび単一ディスク障害モードでは、RAID 5と比べ、RAID 6のパフォーマンスは若干低いですが、同程度です。デュアルディスク障害モードでは、RAID 6は非常に低速です。RAID 6設定では、書き込み操作のためかなりのCPU時間とメモリが必要です。

表 7.1: RAID 5とRAID 6の比較

機能	RAID 5	RAID 6
デバイスの数	N+1(最小限3個)	N+2(最小限4個)
パリティ	分散型、シングル	分散型、デュアル
パフォーマンス	書き込みおよび再構築に中程度の影響	シーケンシャルな書き込みでは、RAID 5より影響大
耐障害性	1つのコンポーネントデバイスの障害	2つのコンポーネントデバイスの障害

7.1.7 ネストしたコンプレックスRAIDレベル

他にもRAIDレベルが開発されています(RAID n、RAID 10、RAID 0+1、RAID 30、RAID 50など)。これらの一部は、ハードウェアベンダーによって作成された専有インプリメンテーションです。RAID 10設定の作成例については、「第9章 「ソフトウェアRAID 10デバイスの作成」」を参照してください。

7.2 YaSTによるソフトウェアRAID設定

YaSTソフトRAID設定には、YaST Expert Partitionerからアクセスできます。このパーティション設定ツールを使用すると、既存のパーティションを編集および削除したり、ソフトウェアRAIDで使用する新規パーティションを作成したりすることもできます。これらの方法は、RAIDレベル0、1、5、および6の設定に適用されます。RAID 10の設定については、[第9章「ソフトウェアRAID 10デバイスの作成」](#)で説明されています。

1. YaSTを起動してパーティショナを開きます。
2. 必要に応じて、RAID設定で使用するパーティションを作成します。パーティションをフォーマットしたり、パーティションタイプを0xFD Linux RAIDに設定したりしないでください。既存のパーティションを使用する場合、パーティションタイプを変更する必要はありません。YaSTによって自動的に変更されます。詳細については、『導入ガイド』、第10章「Expert Partitioner (エキスパートパーティショナ)」、10.1項「熟練者向けパーティション設定の使用」を参照してください。
ハードディスクのどれかに障害が発生した場合にデータを失うリスクを減らすため (RAID 1、RAID 5)、およびRAID 0のパフォーマンスを最適化するため、異なるハードディスクに保存されているパーティションを使用することを強くお勧めします。
RAID 0の場合は、少なくとも2つのパーティションが必要です。RAID 1に必要なパーティションは2つだけですが、RAID 5の場合は少なくとも3つのパーティションが必要です。RAID 6セットアップでは、少なくとも4つのパーティションが必要です。各セグメントは最小サイズのパーティションと同量のスペースしか提供できないので、同じサイズのパーティションだけを使用するようお勧めします。
3. 左のパネルで、RAIDを選択します。
既存のRAID設定のリストが右のパネルに表示されます。
4. [RAID] ページの左下で、RAIDの追加をクリックします。

RAID /dev/md0 の追加

RAID 種類 RAID 名 (N) (オプション)

☒ RAID 0(0) (ストライピング)
☐ RAID 1(1) (ミラーリング)
☐ RAID 5(5) (冗長ストライピング)
☐ RAID 6 (デュアル冗長ストライピング)
☐ RAID 10 (ミラーリングとストライピング)

使用可能なデバイス:

デバイス	サイズ	暗号	タイプ
/dev/sda1	10.00 GiB		Linux LVM
/dev/sda2	10.00 GiB		Linux native
/dev/sda5	20.00 GiB		Linux LVM
/dev/sda6	10.00 GiB		Linux LVM
/dev/sda7	4.00 GiB		Linux RAID
/dev/sdb1	4.00 GiB		Linux RAID
/dev/sdc1	4.00 GiB		Linux native

合計サイズ: 61.99 GiB

選択したデバイス:

デバイス	サイズ	暗号	タイプ	分類
------	-----	----	-----	----

結果サイズ: 0 B

- RAID種類を選択し、追加をクリックして、使用可能なデバイスダイアログから適切な数のパーティションを追加します。
オプションで、RAID名でRAIDに名前を割り当てることができます。この名前は、`/dev/md/NAME` として利用可能になります。詳細については、[7.2.1項「RAIDの名前」](#)を参照してください。



図 7.1: RAID 5設定の例

次へで続行します。

6. チャンクサイズを選択し、該当する場合はパリティアルゴリズムを選択します。最適なチャンクサイズは、データのタイプとRAIDのタイプによって変わります。詳細については、https://raid.wiki.kernel.org/index.php/RAID_setup#Chunk_sizes を参照してください。パリティアルゴリズムの詳細については、`--layout` オプションの検索時に `man 8 mdadm` を使用して参照できます。わからない場合は、デフォルト値を使用してください。
7. 役割でボリュームの役割を選択します。ここで選択した内容は、次のダイアログのデフォルト値にのみ影響します。値は次の手順で変更可能です。わからない場合は、RAW ボリューム(未フォーマット)を選択します。
8. フォーマットオプションで、パーティションをフォーマットするを選択し、ファイルシステムを選択します。オプションメニューの内容は、ファイルシステムによって異なります。通常は、デフォルト値を変更する必要はありません。
マウントのオプションの下で、パーティションをマウントするを選択してから、マウントポイントを選択します。Fstabオプションをクリックして、このボリュームの特別なマウントオプションを追加します。
9. 完了をクリックします。

10. 次へをクリックし、変更が一覧されることを確認してから、完了をクリックします。

7.2.1 RAIDの名前

デフォルトでは、ソフトウェアRAIDデバイスには、`mdN` (Nは数字)というパターンに従った数字の名前が付いています。そのため、たとえば `/dev/md127` としてデバイスにアクセスでき、`/proc/mdstat` および `/proc/partitions` には `md127` としてリストされます。このような名前では作業しづらい場合があります。SUSE Linux Enterprise Serverでは、この問題を回避する方法を2つ提供しています。

デバイスへの名前付きリンクを指定する

オプションで、YaSTでRAIDデバイスを作成する際、または `mdadm --create '/dev/md/ NAME'` を使用してコマンドラインで、RAIDデバイスの名前を指定できます。デバイス名は `mdN` のままですが、リンク `/dev/md/NAME` が作成されます。

```
tux > ls -og /dev/md
total 0
lrwxrwxrwx 1 8 Dec  9 15:11 myRAID -> ../md127
```

デバイスは `/proc` には引き続き `md127` としてリストされます。

名前付きデバイスを指定する

ご使用のセットアップでデバイスへの名前付きリンクでは不十分な場合、次のコマンドを実行して、`/etc/mdadm.conf` に「`CREATE names=yes`」という行を追加します。

```
tux > sudo echo "CREATE names=yes" >> /etc/mdadm.conf
```

これにより、`myRAID` のような名前が「実際の」デバイス名として使用されるようになります。このデバイスは `/dev/myRAID` でアクセスできるだけでなく、`/proc` にも `myRAID` としてリストされます。これは、設定ファイルの変更後に設定したRAIDにのみ適用される点に注意してください。アクティブなRAIDでは、停止して再アSEMBルするまで引き続き `mdN` 形式の名前が使用されます。



警告: 非互換のツール

一部のツールは、名前付きRAIDデバイスをサポートしていません。ツールがRAIDデバイスに `mdN` 形式の名前が付いていることを予期している場合、そのツールはデバイスを特定できません。

7.3 ソフトウェアRAIDのトラブルシューティング

`/proc/mdstat` ファイルをチェックして、RAIDパーティションが破損しているかどうかを調べます。ディスク障害が発生した場合は、Linuxシステムをシャットダウンして、問題のあるハードディスクを、同じ方法でパーティション分割されている新しいハードディスクで置き換えます。次に、システムを再起動して、`mdadm /dev/mdX --add /dev/sdX` コマンドを入力します。「X」を特定のデバイス識別子に置き換えてください。これにより、ハードディスクがRAIDシステムに自動的に統合され、そのRAIDシステムが完全に再構築されます(RAID 0を除くすべてのRAIDレベル)。

再構築中もすべてのデータにアクセスできますが、RAIDが完全に再構築されるまでは、パフォーマンスに問題が発生する場合があります。

7.3.1 ディスク障害復旧後の回復

RAIDアレイに含まれているディスクが障害を起こす理由はいくつかあります。最も一般的な理由を一覧にしました。

- ディスクメディアに問題が発生
- ディスクドライブコントローラに障害発生
- ディスクへの接続に障害発生

ディスクメディアまたはディスクコントローラの障害の場合、デバイスを交換または修理する必要があります。RAID内でホットスペアが設定されていない場合、手動による介入作業が必要です。

最後の接続障害の場合、接続の修復後(自動的に修復する場合もあります)、`mdadm` コマンドによって、障害が発生したデバイスは、自動的に再度追加されます。

`md` / `mdadm` は、ディスク障害の原因を正確に判断できないため、デバイスが正常であると明示的に指示されるまで、ディスクエラーを深刻なエラーと判断し、障害が発生しているデバイスを異常と見なします。

内部RAIDアレイを持つストレージデバイスなど、環境によっては、デバイス障害の原因の多くを接続の問題が占める場合があります。このような場合、`mdadm` に対して、デバイスが表示されたら、そのデバイスを `--re-add` によって自動的に再度追加しても問題ないと指示することができます。これには、以下の行を `/etc/mdadm.conf` に追加します。

```
POLICY action=re-add
```


再表示されたらそのデバイスを自動的に再度追加できるのは、udev ルールによって、mdadm **-I DISK_DEVICE_NAME** が、自動的に表示されたあらゆるデバイスで実行されるように設定されている場合(デフォルトの動作)、およびwrite-intentビットマップが設定されている場合(デフォルトの設定)に限られることに注意してください。

このポリシーを特定のデバイスにのみ適用し、他には適用しない場合、path= オプションを /etc/mdadm.conf 内の POLICY 行に追加して、選択したデバイスにのみデフォルトでないアクションを限定することができます。ワイルドカードを使用して、デバイスのグループを指定することができます。詳しくは、man 5 mdadm.conf を参照してください。

7.4 詳細情報

ソフトウェアRAIDの設定方法と詳細情報が、次のHOWTOにあります。

- Linux RAID wiki: https://raid.wiki.kernel.org/index.php/Linux_Raid ↗
- The Software RAID HOWTO(/usr/share/doc/packages/mdadm/Software-RAID.HOWTO.html ファイル)

「linux-raid」(<http://marc.info/?l=linux-raid> ↗)などのLinux RAIDメーリングリストもあります。

8 ルートパーティション用のソフトウェアRAIDの設定

SUSE Linux Enterprise Serverでは、Device Mapper RAIDツールがYaSTパーティショナに統合されています。インストール時にパーティショナを使用して、ルート(/)パーティションを含むシステムデバイス用にソフトウェアRAIDを作成することができます。/bootパーティションは、RAID 1以外のRAIDパーティションには保存できません。

8.1 ルートパーティション用のソフトウェアRAIDデバイスを使用するための前提条件

設定が次の要件を満たしていることを確認してください。

- RAID 1のミラーリングデバイスを作成するため、2つのハードドライブが必要です。ハードドライブは類似のサイズで構成する必要があります。RAIDは小さい方のドライブのサイズを採用します。ブロックストレージデバイスには、ローカル(マシンに内蔵、または直結されたもの)、ファイバチャネルストレージサブシステム、またはiSCSIストレージサブシステムを自由に組み合わせることができます。
- ブートローダをMBRにインストールする場合、/boot用の別のパーティションは必要ありません。ブートローダをMBRにインストールすることが不可能な場合は、/bootが別のパーティションに存在する必要があります。
- UEFIマシンの場合、専用の /boot/efi パーティションを設定する必要があります。これはVFATフォーマットである必要があります。RAID 1デバイスに配置されていれば、/boot/efiが存在する物理ディスクに障害が発生した場合にブートの問題を回避できます。
- ハードウェアRAIDデバイスを使用している場合は、その上でソフトウェアRAIDを実行しようとししないでください。
- iSCSIターゲットデバイスをご使用の場合は、RAIDデバイスを作成する前にiSCSIイニシエータサポートを有効にする必要があります。
- ご使用のストレージサブシステムが、ソフトウェアRAIDを使用する予定の直接接続されたローカルデバイス、ファイバチャネルデバイス、またはiSCSIデバイスとサーバの間で複数のI/Oパスを提供している場合は、RAIDデバイスを作成する前に、マルチパスサポートを有効にしなければなりません。

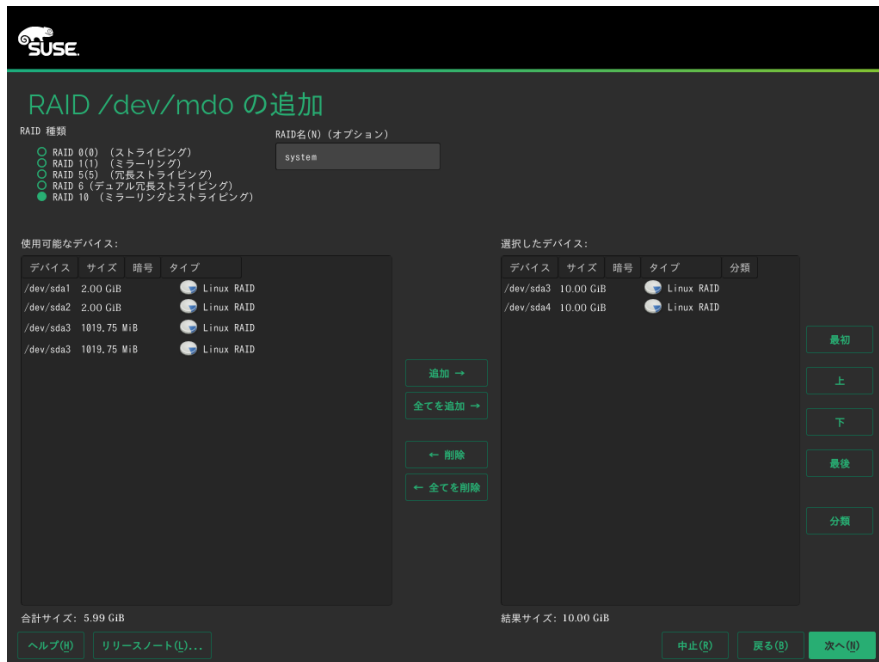
8.2 ルート(/)パーティションにソフトウェアRAIDデバイスを使用するシステムの設定

1. YaSTを使用してインストールを開始し、推奨されたパーティション分割の手順に到達するまで、『導入ガイド』、第8章「インストール手順」の説明に従って進めます。
2. エキスパートパーティショナをクリックして、カスタムパーティショニングツールを開きます。
3. (オプション)使用したいiSCSIターゲットデバイスがある場合、画面右下のセクションで設定 > Configure iSCSI (iSCSIの設定)の順に選択して、iSCSIイニシエータソフトウェアを有効にする必要があります。詳細については、[第14章「IPネットワークの大容量記憶域 - iSCSI」](#)を参照してください。
4. (オプション)使用したいデバイスへのI/Oパスが複数存在する場合、画面右下のセクションで設定 > マルチパスの設定 > はいの順に選択して、マルチパスサポートを有効にする必要があります。
5. (オプション)iSCSIもマルチパスも設定していない場合は、デフォルトの推奨設定が表示されます。デバイスの再検出をクリックすると、それらは削除されます。
6. ソフトウェアRAIDに使用する各デバイスの0xFD Linux RAIDフォーマットを設定します。/、/boot/efi、またはスワップパーティションにはRAIDを使用する必要があります。
 - a. 左パネルでハードディスクを選択し、使用するデバイスを選択してからパーティションの追加をクリックします。
 - b. 新しいパーティションの種類で、プライマリパーティションを選択し、次に次へをクリックします。
 - c. 新しいパーティションのサイズで、使用するサイズを指定し、次に次へをクリックします。
 - d. 役割でRaw Volume (Unformatted) (RAWボリューム(未フォーマット))を選択します。
 - e. フォーマットしないを選択し、ファイルシステムIDを0xFD Linux RAIDに設定します。
 - f. 終了をクリックし、2番目のパーティションに対して同じ手順を繰り返します。



7. / パーティション用のRAIDデバイスを作成します。

- a. 左パネルでRAIDを選択し、RAIDの追加を選択します。
- b. / パーティションに対して目的のRAID種類を設定し、RAID名を system に設定します。
- c. 前の手順で準備した2つのRAIDデバイスを使用可能なデバイスから選択し、追加をクリックして追加します。



次へで続行します。

- d. RAIDオプションで、ドロップダウンボックスからチャンクサイズを選択します。デフォルト値をそのまま使用するのが安全です。
 - e. 役割で、オペレーティングシステムを選択し、完了で続行します。
 - f. ファイルシステムを選択し、マウントポイントを / に設定します。完了をクリックして、このダイアログを終了します。
8. ソフトウェアRAIDデバイスはデバイスマッパーによって管理され、デバイスを /dev/md/system パスの下に作成します。
 9. UEFIマシンに対し、同様の手順を使用して、/boot/efi にマウントされるパーティションを作成することもできます。/boot/efi でサポートされるのはRAID 1のみであること、およびパーティションがFATファイルシステムでフォーマットされている必要があることを忘れないでください。

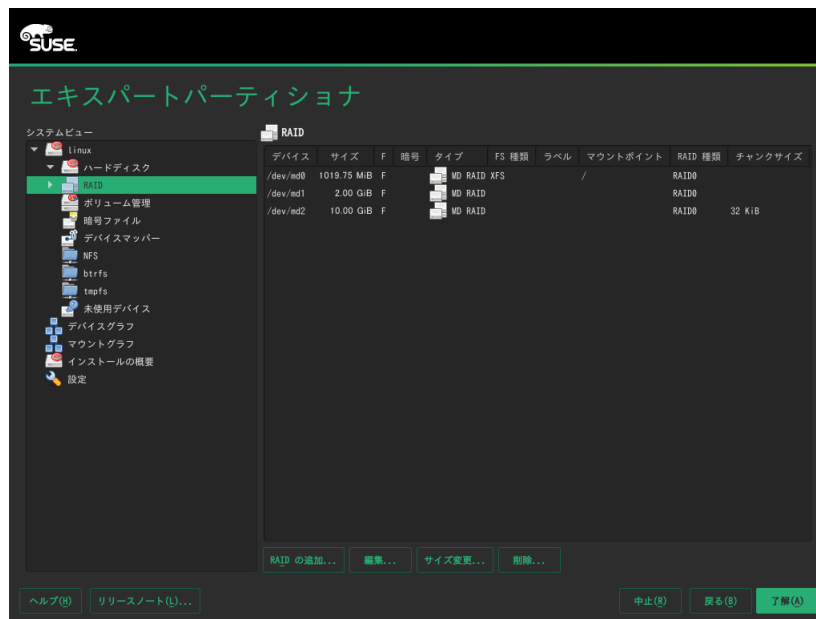


図 8.1: RAID上の//、/BOOT/EFI、およびスワップ

10. 承認をクリックして、パーティショナを終了します。
推奨されたパーティション分割ページに新しい案が表示されます。
11. インストールを続行します。独立した /boot/efi パーティションを持つUEFIマシンでは、インストールの設定画面でブートをクリックし、GRUB2 for EFI (EFI用のGRUB2)をブートローダとして設定します。Secure Bootサポートを有効にするオプションがアクティブになっていることを確認します。
サーバを再起動するたびに、デバイスマAPPERが起動時に開始し、ソフトウェアRAIDが自動的に認識され、ルート(/)パーティション上のオペレーティングシステムを開始することができます。

9 ソフトウェアRAID 10デバイスの作成

本項では、ネストしたコンプレックスRAID 10デバイスの設定方法について説明します。RAID 10デバイスは、ネストしたRAID 1 (ミラーリング)アレイとRAID 0 (ストライピング)アレイで構成されます。ネストしたRAIDは、ストライピングミラー(RAID 1+0)またはミラーリングされたストライプ(RAID 0+1)のいずれかとして設定できます。コンプレックスRAID 10のセットアップは、ミラーとストライプを組み合わせ、より高いデータ冗長性レベルをサポートすることによってデータのセキュリティを強化します。

9.1 mdadmによるネストしたRAID 10デバイスの作成

ネストしたRAIDデバイスは、物理ディスクを使用する代わりに、その基本エレメントとして別のRAIDアレイを使用するRAIDアレイで構成されます。この構成の目的は、RAIDのパフォーマンスと耐障害性を向上することです。ネストしたRAIDレベルの設定はYaSTではサポートされていませんが、**mdadm** コマンドラインツールを使用して実行できます。

ネストの順序に基づいて、2つの異なるネストしたRAIDを設定できます。このマニュアルでは、次の用語を使用します。

- **RAID 1+0:** まず、RAID 1 (ミラー) アレイが構築され、次に、それらのアレイが組み合わされてRAID 0 (ストライプ)アレイを構成します。
- **RAID 0+1:** まず、RAID 0 (ストライプ) アレイが構築され、次に、それらのアレイが組み合わされてRAID 1(ミラー)アレイを構成します。

次の表では、RAID 10ネスティングの欠点と利点を、1+0対0+1という形式で説明します。使用するストレージオブジェクトは、それぞれが専用のI/Oをもつ別々のディスクに常駐すると想定しています。

表 9.1: ネストしたRAIDレベル

RAIDレベル	説明	パフォーマンスと耐障害性
10 (1+0)	RAID 1(ミラー)アレイで構築されたRAID (ストライプ)	RAID 1+0は、高レベルのI/Oパフォーマンス、データ冗長性、およびディスク耐障害性を提供します。RAIDの各メンバーデバイスは個々にミラーリングされるので、エラーになったディスクのミラー先が異なる限り、複数ディスクの障害を許容し、データを使用することができます。

RAIDレベル	説明	パフォーマンスと耐障害性
		オプションとして、ベースをなすミラーリングされたアレイごとにスペアを設定したり、すべてのミラーに対するスペアグループに対応するスペアを設定できます。
10 (0+1)	RAID 0(ストライプ)アレイで構築されたRAID 1(ミラー)	<p>RAID 0+1は、高レベルのI/Oパフォーマンスとデータ冗長性を提供しますが、耐障害性が1+0より若干低くなります。ミラーの一方のサイドで複数のディスクがエラーになると、もう一方のミラーが使用可能になります。ただし、ミラーの両サイドで同時にディスクが失われると、すべてのデータが喪失します。</p> <p>このソリューションは1+0ソリューションより耐障害性が低いですが、別のサイトで保守を実行したり、ミラーを保持する必要がある場合、ミラーのサイド全体をオフラインにしても、完全に機能するストレージデバイスを保持することができます。また、2つのサイト間の接続が失われた場合は、どちらかのサイトがもう一方のサイトから独立して稼働します。ミラーリングされたセグメントをストライプする場合はこうなりません。ミラーが低レベルで管理されているからです。</p> <p>デバイスがエラーになると、RAID 0には耐障害性がないので、そのサイドのミラーがエラーになります。新しいRAID 0を作成して、エラーになったサイドに置き換え、次に、ミラーを再同期してください。</p>

9.1.1 mdadmによるネストしたRAID 10 (1+0)デバイスの作成

ネストしたRAID 1+0は、2つ以上のRAID 1(ミラー)デバイスを作成し、それらのRAID 1デバイスをRAID 0のコンポーネントデバイスとして使用することで構築します。

！ 重要: マルチパス処理

デバイスに対する複数の接続を管理する必要がある場合は、マルチパスI/Oを設定してから、RAIDデバイスを設定する必要があります。詳細については、[第17章「デバイスのマルチパスI/Oの管理」](#)を参照してください。

本項の手順では、次の表に示すデバイス名を使用します。それらのデバイス名は、必ず、ご使用のデバイスの名前を変更してください。

表 9.2: ネスティングでRAID 10(1+0)を作成するシナリオ

rawデバイス	RAID 1(ミラー)	RAID 1+0(ストライピングミラー)
<u>/dev/sdb1</u> <u>/dev/sdc1</u>	<u>/dev/md0</u>	<u>/dev/md2</u>
<u>/dev/sdd1</u> <u>/dev/sde1</u>	<u>/dev/md1</u>	

1. 端末コンソールを開きます。
2. 必要に応じて、partedなどのディスクパーティショナを使用して、同じサイズの0xFD Linux RAIDパーティションを4つ作成します。
3. 1デバイスごとに2つの異なるデバイスを使用して、2つのソフトウェアRAID 1デバイスを作成します。コマンドプロンプトで、次の2つのコマンドを入力します。

```
tux > sudo mdadm --create /dev/md0 --run --level=1 --raid-devices=2 /dev/sdb1 /dev/sdc1
sudo mdadm --create /dev/md1 --run --level=1 --raid-devices=2 /dev/sdd1 /dev/sde1
```

4. ネストしたRAID 1+0デバイスを作成します。コマンドプロンプトで、前の手順で作成したソフトウェアRAID 1デバイスを使用して、次のコマンドを入力します。

```
tux > sudo mdadm --create /dev/md2 --run --level=0 --chunk=64 \
--raid-devices=2 /dev/md0 /dev/md1
```

デフォルトのチャンクサイズは 64KBです。

5. RAID 1+0デバイス /dev/md2 上でファイルシステム(XFSファイルシステムなど)を作成します。

```
tux > sudo mkfs.xfs /dev/md2
```


これとは別のファイルシステムを使用するには、コマンドを変更します。

6. `/etc/mdadm.conf` ファイルを編集するか、ファイルがまだ存在しない場合は作成します(たとえば、`sudo vi /etc/mdadm.conf` を実行します)。次の行を追加します(ファイルが既に存在する場合、最初の行は記述済みの可能性があります)。

```
DEVICE containers partitions
ARRAY /dev/md0 UUID=UUID
ARRAY /dev/md1 UUID=UUID
ARRAY /dev/md2 UUID=UUID
```

各デバイスのUUIDは次のコマンドで取得できます。

```
tux > sudo mdadm -D /dev/DEVICE | grep UUID
```

7. `/etc/fstab` ファイルを編集して、RAID 1+0デバイス `/dev/md2` のエントリを追加します。次の例は、XFSファイルシステム、およびマウントポイントとして `/data` を使用するRAIDデバイスのエントリを示しています。

```
/dev/md2 /data xfs defaults 1 2
```

8. RAIDデバイスをマウントします。

```
tux > sudo mount /data
```

9.1.2 mdadmによるネストしたRAID 10 (0+1)デバイスの作成

ネストしたRAID 0+1は、2個から4個のRAID 0(ストライプ)デバイスで構築され、それらのRAID 0デバイスをミラーリングしてRAID 1のコンポーネントデバイスとします。

！ 重要: マルチパス処理

デバイスに対する複数の接続を管理する必要がある場合は、マルチパスI/Oを設定してから、RAIDデバイスを設定する必要があります。詳細については、[第17章「デバイスのマルチパスI/Oの管理」](#)を参照してください。

この構成では、RAID 0がデバイスの喪失に耐えられないので、ベースのRAID 0デバイスにスペアデバイスを指定できません。デバイスがミラーの1つのサイドでエラーになった場合は、置き換え用のRAID 0デバイスを作成して、ミラーに追加します。

本項の手順では、次の表に示すデバイス名を使用します。それらのデバイス名は、必ず、ご使用のデバイスの名前を変更してください。

表 9.3: ネストによるRAID 10 (0+1)作成のシナリオ

rawデバイス	RAID 0 (ストライプ)	RAID 0+1 (ミラー化ストライピング)
<u>/dev/sdb1</u> <u>/dev/sdc1</u>	<u>/dev/md0</u>	<u>/dev/md2</u>
<u>/dev/sdd1</u> <u>/dev/sde1</u>	<u>/dev/md1</u>	

1. 端末コンソールを開きます。
2. 必要に応じて、partedなどのディスクパーティショナを使用して、同じサイズの0xFD Linux RAIDパーティションを4つ作成します。
3. RAID 0デバイスごとに2つの異なるデバイスを使用して、2つのソフトウェアRAID 0デバイスを作成します。コマンドプロンプトで、次の2つのコマンドを入力します。

```
tux > sudo mdadm --create /dev/md0 --run --level=0 --chunk=64 \
--raid-devices=2 /dev/sdb1 /dev/sdc1
sudo mdadm --create /dev/md1 --run --level=0 --chunk=64 \
--raid-devices=2 /dev/sdd1 /dev/sde1
```

デフォルトのチャンクサイズは 64KBです。

4. ネストしたRAID 0+1デバイスの作成コマンドプロンプトで、前の手順で作成したソフトウェアRAID 0デバイスを使用して、次のコマンドを入力します。

```
tux > sudo mdadm --create /dev/md2 --run --level=1 --raid-devices=2 /dev/md0 /dev/
md1
```

5. RAID 1+0デバイス /dev/md2 上でファイルシステム(XFSファイルシステムなど)を作成します。

```
tux > sudo mkfs.xfs /dev/md2
```

これとは別のファイルシステムを使用するには、コマンドを変更します。

6. /etc/mdadm.conf ファイルを編集するか、ファイルがまだ存在しない場合は作成します(たとえば、**sudo vi /etc/mdadm.conf**を実行します)。次の行を追加します(ここでも、ファイルが既に存在する場合、最初の行は記述済みの可能性があります)。

```
DEVICE containers partitions
ARRAY /dev/md0 UUID=UUID
```

```
ARRAY /dev/md1 UUID=UUID
ARRAY /dev/md2 UUID=UUID
```

各デバイスのUUIDは次のコマンドで取得できます。

```
tux > sudo mdadm -D /dev/DEVICE | grep UUID
```

7. `/etc/fstab` ファイルを編集して、RAID 1+0デバイス `/dev/md2` のエントリを追加します。次の例は、XFSファイルシステム、およびマウントポイントとして `/data` を使用するRAIDデバイスのエントリを示しています。

```
/dev/md2 /data xfs defaults 1 2
```

8. RAIDデバイスをマウントします。

```
tux > sudo mount /data
```

9.2 コンプレックスRAID 10の作成

YaST(および `mdadm` と `--level=10` オプション)では、RAID 0(ストライピング)およびRAID 1(ミラーリング)の両方の機能を組み合わせた単一のコンプレックスソフトウェアRAID 10デバイスが作成されます。すべてのデータブロックの複数のコピーが、ストライピングの規則に従って、複数のドライブ上に配置されます。コンポーネントデバイスは、すべて同じサイズにする必要があります。

コンプレックスRAIDは、ネストしたRAID 10 (1+0)と目的は同じですが、次の点で異なります。

表 9.4: 複雑なRAID 10とネストしたRAID 10の比較

機能	コンプレックスRAID 10	ネストしたRAID 10 (1+0)
デバイスの数	偶数個または奇数個のコンポーネントデバイス	偶数個のコンポーネントデバイス
コンポーネントデバイス	単一のRAIDデバイスとして管理されます。	ネストしたRAIDデバイスとして管理されます。
ストライピング	ストライピングは、コンポーネントデバイス上にnearレイアウトまたはfarレイアウトを生じます。	ストライピングは、連続的に、すべてのコンポーネントデバイスをまたぎます。

機能	コンプレックスRAID 10	ネストしたRAID 10 (1+0)
	farレイアウトでは、RAID 1ペアの数でなく、ドライブ数で増減するシーケンシャルな読み込みスループットを提供します。	
データの複数コピー	2からアレイ内のデバイス数まで	ミラーリングされたセグメントごとにコピー
ホットスペアデバイス	単一スペアですべてのコンポーネントデバイスに対応できます。	ベースをなすミラーリングされたアレイごとにスペアを設定したり、すべてのミラーに対応するスペアグループに対するスペアを設定できます。

9.2.1 コンプレックスRAID 10のデバイスおよびレプリカの数

コンプレックスRAID 10アレイの設定時に、データブロックごとに必要なレプリカ数を指定する必要があります。デフォルトのレプリカ数は2ですが、2からアレイ内のデバイス数まで可能です。

少なくとも、指定のレプリカ数と同数のコンポーネントデバイスを使用する必要があります。ただし、RAID 10アレイのコンポーネントデバイス数は各データブロックのレプリカ数の倍数である必要はありません。有効なストレージサイズは、デバイス数をレプリカ数で割った数です。

たとえば、5個のコンポーネントデバイスで作成したアレイに2つのレプリカを指定した場合は、各ブロックのコピーが2つの異なるデバイスに保存されます。したがって、すべてのデータの1コピーの有効なストレージサイズは、 $5/2$ (つまり、コンポーネントデバイスのサイズの2.5倍)となります。

9.2.2 レイアウト

コンプレックスRAID 10のセットアップでは、ディスクにデータブロックを配置する方法を定義するレイアウトが3つサポートされています。利用可能なレイアウトは、near (デフォルト)、far、およびoffsetです。各レイアウトはパフォーマンス特性が異なるため、ワークロードに適したレイアウトを選択することが重要です。

9.2.2.1 nearレイアウト

nearレイアウトでは、異なるコンポーネントデバイス上で、データブロックのコピーが互いに接近してストライプされます。つまり、あるデータブロックの複数のコピーが異なるデバイス内で同様にオフセットされます。nearは、RAID 10のデフォルトレイアウトです。たとえば、奇数個のコンポーネントデバイスとデータの2コピーを使用する場合は、一部のコピーが、1チャンク分、デバイス内を前進します。

コンプレックスRAID 10のnearレイアウトは、半数のドライブ上のRAID 0と同様の読み書きパフォーマンスを提供します。

偶数個のディスクと2つのレプリカを使用したnearレイアウト

```
sda1 sdb1 sdc1 sde1
0    0    1    1
2    2    3    3
4    4    5    5
6    6    7    7
8    8    9    9
```

奇数個のディスクと2つのレプリカを使用したnearレイアウト

```
sda1 sdb1 sdc1 sde1 sdf1
0    0    1    1    2
2    3    3    4    4
5    5    6    6    7
7    8    8    9    9
10   10   11   11   12
```

9.2.2.2 farレイアウト

farレイアウトは、すべてのドライブの前半部分にデータをストライプし、次に、2つ目のデータコピーをすべてのドライブの後半部分にストライプして、ブロックのすべてのコピーが異なるドライブに配置されるようにします。値の2つ目のセットは、コンポーネントドライブの中ほどから開始します。

farレイアウトでは、コンプレックスRAID 10の読み込みパフォーマンスは、すべてのドライブを使用したRAID 0と同様ですが、書き込みパフォーマンスは、ドライブヘッドのシーク回数が増えるので、RAID 0よりかなり遅くなります。このレイアウトは、読み込み専用ファイルサーバなどの、読み込み集約型操作に最適です。

RAID 10の書き込み速度は、nearレイアウトを使用しているRAID 1やRAID 10などの他のミラーリングRAIDの種類と同等です。これは、ファイルシステムのエレベータが生の書き込みよりも効率のよい書き込みのスケジュールを行うためです。RAID 10をfarレイアウトで使用方法は、ミラーリングによる書き込みアプリケーションに適しています。

偶数個のディスクと2つのレプリカを使用したfarレイアウト

```
sda1 sdb1 sdc1 sde1
0    1    2    3
4    5    6    7
. . .
3    0    1    2
7    4    5    6
```

奇数個のディスクと2つのレプリカを使用したfarレイアウト

```
sda1 sdb1 sdc1 sde1 sdf1
0    1    2    3    4
5    6    7    8    9
. . .
4    0    1    2    3
9    5    6    7    8
```

9.2.2.3 offsetレイアウト

offsetレイアウトでは、あるチャンクの複数のコピーが連続したドライブ上で連続したオフセットにレイアウトされるよう、ストライプが複製されます。実際は、それぞれのストライプが複製され、コピーが1つのデバイスでオフセットされます。これにより、適度な大きさのチャンクサイズを使用している場合は、farレイアウトと同様の読み込み特性が得られますが、書き込みのシーク回数は少なくなります。

偶数個のディスクと2つのレプリカを使用したoffsetレイアウト

```
sda1 sdb1 sdc1 sde1
0    1    2    3
3    0    1    2
4    5    6    7
7    4    5    6
8    9    10   11
11   8    9    10
```

奇数個のディスクと2つのレプリカを使用したoffsetレイアウト

sda1	sdb1	sdcl	sde1	sdf1
0	1	2	3	4
4	0	1	2	3
5	6	7	8	9
9	5	6	7	8
10	11	12	13	14
14	10	11	12	13

9.2.2.4 YaSTおよびmdadmによるレプリカ数とレイアウトの指定

レプリカ数とレイアウトは、YaSTではパリティアルゴリズム、mdadmでは `--layout` パラメータで指定します。使用できる値は次のとおりです。

nN

nearレイアウトの場合、nを指定し、Nをレプリカ数で置き換えます。レイアウトおよびレプリカ数を設定しない場合、デフォルトで n2 が使用されます。

fN

farレイアウトの場合、fを指定し、Nをレプリカ数で置き換えます。

oN

offsetレイアウトの場合、oを指定し、Nをレプリカ数で置き換えます。



注記: レプリカの数

YaSTでは、パリティアルゴリズムパラメータに設定可能なすべての値が自動的に表示されます。

9.2.3 YaSTパーティショナによるコンプレックスRAID 10の作成

1. YaSTを起動してパーティショナを開きます。
2. 必要に応じて、RAID設定で使用するパーティションを作成します。パーティションをフォーマットしたり、パーティションタイプを0xFD Linux RAIDに設定したりしないでください。既存のパーティションを使用する場合、パーティションタイプを変更する必要はありません。YaSTによって自動的に変更されます。詳細については、『導入ガイド』、第10章「Expert Partitioner (エキスパートパーティショナ)」、10.1項「熟練者向けパーティション設定の使用」を参照してください。

RAID 10の場合は、少なくとも4つのパーティションが必要です。ハードディスクのどれかに障害が発生した場合にデータを失うリスクを減らすため、異なるハードディスクに保存されているパーティションを使用することを強くお勧めします。各セグメントは最小サイズのパーティションと同量のスペースしか提供できないので、同じサイズのパーティションだけを使用するようお勧めします。

3. 左のパネルで、RAIDを選択します。
既存のRAID設定のリストが右のパネルに表示されます。
4. [RAID] ページの左下で、RAIDの追加をクリックします。
5. RAID種類で、RAID 10 (ミラーリングおよびストライピング)を選択します。
オプションで、RAID Name (RAID名)でRAIDに名前を割り当てることができます。この名前は、`/dev/md/NAME` として利用可能になります。詳細については、[7.2.1項「RAIDの名前」](#)を参照してください。
6. 使用可能なデバイスリストで、希望のパーティションを選択し、次に追加をクリックして、それらを選択したデバイスリストに移動します。

RAID /dev/md0 の追加

RAID 種類

- ☐ RAID 0(0) (ストライピング)
- ☐ RAID 1(1) (ミラーリング)
- ☐ RAID 5(5) (冗長ストライピング)
- ☐ RAID 6 (デュアル冗長ストライピング)
- ☒ RAID 10 (ミラーリングとストライピング)

RAID 名(N) (オプション)

DATA

使用可能なデバイス:

デバイス	サイズ	暗号	タイプ
------	-----	----	-----

追加 →

全てを追加 →

← 削除

← 全てを削除

合計サイズ: 0 B

ヘルプ(H)

戻る(B)

選択したデバイス:

デバイス	サイズ	暗号	タイプ	最初
/dev/sda2	4.00 GiB		Li	
/dev/sdb1	4.00 GiB		Li	上
/dev/sdc1	4.00 GiB		Li	下
/dev/sdd1	4.00 GiB		Li	最後

分類

結果サイズ: 8.00 GiB

中止(R)

次へ(N)

7. (オプション) 分類をクリックして、RAIDアレイ内でのディスクの好みの順番を指定します。
RAID 10など、追加したディスクの順序が重要なRAIDタイプでは、デバイスの使用順序を指定できます。これにより、アレイの半数を特定のディスクサブシステムに配置し、もう半数を別のディスクサブシステムに配置できます。たとえば、1つのディスクサブシステムに障害が発生した場合、システムは2番目のディスクサブシステムから稼働し続けます。

- a. 各ディスクを順番に選択して、Class Xボタンのいずれかをクリックします。
ここで、Xは、そのディスクに割り当てられた文字です。用意されているクラスはA、B、C、DおよびEですが、多くの場合必要なクラスはそれより少なくなります(たとえばAとBのみ)。このようにして、すべての利用可能なRAIDディスクを割り当てます。
複数のデバイスを選択するには、**Ctrl** キーまたは **Shift** キーを押します。選択したデバイスを右クリックして、コンテキストメニューから適切なクラスを選択することもできます。

- b. 次のソートオプションのいずれかを選択して、デバイスの順序を指定します。

Sorted: クラスAのすべてのデバイスを、クラスBのすべてのデバイスより前に、というように並べます。例: AABBCC。

Interleaved: クラスAの最初のデバイス、次にクラスBの最初のデバイス、次にデバイスが割り当てられたすべての後続のクラスの順に、デバイスを並べます。次にクラスAの2番目のデバイス、クラスBの2番目のデバイス、というように続きます。クラスを持たないデバイスはすべて、デバイスリストの最後に並べられます。たとえば、ABCAABC のようになります。

Pattern File: それぞれが正規表現とクラス名である、複数の行を含む既存のファイルを選択します("sda.* A")。その正規表現に合致するすべてのデバイスが、その行に指定されたクラスに割り当てられます。正規表現は、カーネル名(/dev/sda1)、udevパス名(/dev/disk/by-path/pci-0000:00:1f.2-scsi-0:0:0:0-part1)、次にudev ID (/dev/disk/by-id/ata-ST3500418AS_9VMN8X8L-part1)に対して照合されます。デバイスの名前が、2つ以上の正規表現に合致する場合は、最初に合致したものでクラスが決定されます。

- c. ダイアログの下で、OKをクリックして、順番を確定します。

デバイス	分類
/dev/sdb1	A
/dev/sdd1	B
/dev/sda2	A
/dev/sdc1	B

分類 A

分類 B

分類 C

分類 D

分類 E

ヘルプ(H)

Sorted (AAABBBCCC)

Interleaved (ABCABCABC)

Pattern File

キャンセル(C)

OK(O)

8. 次へをクリックします。
9. RAIDオプションで、チャンクサイズとパリティアルゴリズムを指定し、次に次へをクリックします。
RAID 10の場合、パリティオプションは、n (near)、f (far)、およびo (offset)です。数字は、必要となる各データブロックのレプリカの数を示します。2がデフォルトの設定です。詳細については、[9.2.2項「レイアウト」](#)を参照してください。
10. ファイルシステムとマウントオプションをRAIDデバイスに追加して、完了をクリックします。
11. 次へをクリックします。
12. 変更する内容を確認して、完了をクリックすると、RAIDが作成されます。

9.2.4 mdadmによるコンプレックスRAID 10の作成

本項の手順では、次の表に示すデバイス名を使用します。それらのデバイス名は、必ず、ご使用のデバイスの名前に変更してください。

表 9.5: MDADMでRAID 10を作成するシナリオ

rawデバイス	RAID 10
/dev/sdf1	/dev/md3

rawデバイス	RAID 10
<u>/dev/sdg1</u>	
<u>/dev/sdh1</u>	
<u>/dev/sdi1</u>	

1. 端末コンソールを開きます。
2. 必要に応じて、partedなどのディスクパーティショナを使用して、同じサイズの0xFD Linux RAIDパーティションを少なくとも4つ作成します。
3. 次のコマンドを入力してRAID 10を作成します。

```
tux > sudo mdadm --create /dev/md3 --run --level=10 --chunk=32 --raid-devices=4 \
/dev/sdf1 /dev/sdg1 /dev/sdh1 /dev/sdi1
```

--raid-devices の値とパーティションのリストは、ご使用のセットアップに応じて調整してください。

ここに示すコマンドでは、nearレイアウトを使用し、2つのレプリカを持つアレイが作成されます。これら2つの値を変更するには、--layout を使用します。[9.2.2.4項「YaSTおよびmdadmによるレプリカ数とレイアウトの指定」](#)を参照してください。

4. RAID 10デバイス /dev/md3 上でファイルシステム(XFSファイルシステムなど)を作成します。

```
tux > sudo mkfs.xfs /dev/md3
```

これとは別のファイルシステムを使用するには、コマンドを変更します。

5. /etc/mdadm.conf ファイルを編集するか、ファイルがまだ存在しない場合は作成します(たとえば、**sudo vi /etc/mdadm.conf** を実行します)。次の行を追加します(ここでも、ファイルが既に存在する場合、最初の行は記述済みの可能性があります)。

```
DEVICE containers partitions
ARRAY /dev/md3 UUID=UUID
```

デバイスのUUIDは次のコマンドで取得できます。

```
tux > sudo mdadm -D /dev/md3 | grep UUID
```

6. /etc/fstab ファイルを編集して、RAID 10デバイス /dev/md3 のエントリを追加します。次の例は、XFSファイルシステム、およびマウントポイントとして /data を使用するRAIDデバイスのエントリを示しています。

```
/dev/md3 /data xfs defaults 1 2
```

7. RAIDデバイスをマウントします。

```
tux > sudo mount /data
```

10 ディグレードアレイの作成

ディグレードアレイは、一部のデバイスが欠けたアレイです。ディグレードアレイは、RAID 1、RAID 4、RAID 5、およびRAID 6に対してのみサポートされています。これらのRAIDタイプは、その耐障害性機能として、一部のデバイスの欠落に耐えるように設計されています。通常、デバイスに障害が発生すると、ディグレードアレイが生成されます。ディグレードアレイは、意図的に作成することもできます。

RAIDの種類	許容可能な欠落スロット数
RAID 1	1つ以外の全スロット
RAID 4	1スロット
RAID 5	1スロット
RAID 6	1個または2個のスロット

一部のデバイスが欠落したディグレードアレイを作成するには、単に、デバイス名の代わりに `missing` というワードを指定します。この指定により、`mdadm` は、アレイ内の対応するスロットを空のまま残します。

RAID 5アレイの作成時に、`mdadm` によって、余分なスペアドライブをもつディグレードアレイが自動的に作成されます。これは、一般に、ディグレードアレイ内にスペアを構築した方が、ディグレードアレイではないが正常でないアレイ上でパリティを再同期するより高速なためです。この機能は、`--force` オプションで無効にできます。

RAIDを作成したいが、使用するデバイスの1つに既にデータが入っている場合は、ディグレードアレイを作成すると便利ことがあります。その場合は、他のデバイスでディグレードアレイを作成し、その使用中のデバイスからのデータをディグレードモードで実行中のRAIDにコピーし、デバイスをRAIDに追加して、RAIDの再構築まで待機すると、データがすべてのデバイスに行き渡ります。このプロセスの例を、次のプロシージャで示します。

1. シングルドライブ `/dev/sd1` を使用してディグレードRAID 1デバイス `/dev/md0` を作成するには、コマンドプロンプトで、次のように入力します。

```
tux > sudo mdadm --create /dev/md0 -l 1 -n 2 /dev/sd1 missing
```

追加先のデバイスは、追加するデバイスと同じか、またはそれ以上のサイズをもつ必要があります。

2. ミラーに追加したいデバイスに、RAIDアレイに移動したいデータが含まれている場合は、この時点で、そのデータを、ディグレードモードで実行中のRAIDアレイにコピーします。
3. データのコピー元のデバイスをミラーに追加します。たとえば、`/dev/sdb1`をRAIDに追加するには、コマンドプロンプトで、次のように入力します。

```
tux > sudo mdadm /dev/md0 -a /dev/sdb1
```

一度に1つのデバイスのみ追加できます。カーネルがミラーを構築し、完全にオンラインにした後、別のミラーを追加できます。

4. 構築の進捗状況を監視するには、コマンドプロンプトで、次のように入力します。

```
tux > sudo cat /proc/mdstat
```

毎秒更新されている間に再構築の進捗を確認するには、次のように入力します。

```
tux > sudo watch -n 1 cat /proc/mdstat
```

11 mdadmによるソフトウェアRAIDアレイのサイズ変更

本項では、ソフトウェアRAID 1、4、5、または6のデバイスのサイズを複数デバイス管理(**mdadm(8)**)ツールで増減する方法について説明します。

既存のソフトウェアRAIDデバイスのサイズ変更には、各コンポーネントパーティションが提供するスペースの増減が必要です。デバイスの使用可能な領域の変更を利用するために、RAIDに存在するファイルシステムもサイズ変更できる必要があります。SUSE Linux Enterprise Serverでは、ファイルシステムBtrfs、Ext2、Ext3、Ext4、およびXFS (サイズの増加のみ)用のファイルシステムサイズ変更ユーティリティが提供されています。詳細については、[第2章「ファイルシステムのサイズ変更」](#)を参照してください。

mdadm ツールは、ソフトウェアRAIDレベル 1、4、5、および6に対してだけサイズ変更をサポートします。これらのRAIDレベルには耐障害性があるので、一度に1つずつ、サイズ変更するコンポーネントパーティションを削除できます。原則として、RAIDパーティションのホットリサイズが可能です。その場合は、データの保全に特に注意する必要があります。



警告: サイズ変更前のデータのバックアップ

パーティションまたはファイルシステムのサイズ変更には、データを失う可能性を伴うリスクが伴います。データの喪失を避けるには、データを必ずバックアップしてから、サイズ変更タスクを開始します。

RAIDのサイズ変更には、次のようなタスクがあります。タスクの実行順序は、サイズを増加するか、減少するかによって異なります。

表 11.1: RAIDのサイズ変更に必要なタスク

仕事	説明	サイズを増大させる場合の順序	サイズを減少させる場合の順序
各コンポーネントパーティションのサイズを変更します。	各コンポーネントパーティションのアクティブなサイズを増加または減少します。一度に1つのコンポーネントパーティションだけを削除し、そのサイズを変更してから、パーティションをRAIDに戻します。	1	2

仕事	説明	サイズを増大させる場合の順序	サイズを減少させる場合の順序
ソフトウェアRAID自体をサイズ変更します。	RAIDは、ベースのコンポーネントパーティションの増減を自動的には認識しません。したがって、RAIDに新しいサイズを知らせる必要があります。	2	3
ファイルシステムのサイズを変更します。	RAIDに常駐するファイルシステムをサイズ変更する必要があります。これは、サイズ変更のツールを提供するファイルシステムの場合のみ可能です。	3	1

以降の各項の手順では、次の表に示すデバイス名を使用します。これらの名前は変更して、必ずご使用のデバイスの名前を使用してください。

表 11.2: コンポーネントパーティションのサイズを増加するシナリオ

RAIDデバイス	コンポーネントパーティション
<u>/dev/md0</u>	<u>/dev/sda1</u> <u>/dev/sdb1</u> <u>/dev/sdc1</u>

11.1 ソフトウェアRAIDのサイズの増加

ソフトウェアRAIDのサイズを増やすには、複数のタスクを所定の順序で実行する必要があります。まずRAIDを構成するすべてのパーティションのサイズを増加させ、次にRAID自体のサイズを増加させます。そして最後に、ファイルシステムのサイズを増加させます。



警告: データ消失の可能性

RAIDに、ディスクの耐障害性がないか、単に一貫性がない場合、パーティションのどれかを削除すると、データが失われます。パーティションの削除は注意深く行い、必ず、データのバックアップをとってください。

11.1.1 コンポーネントパーティションのサイズの増加

RAID 1、4、5、または6のサイズを増加するには、本項の手順を適用します。RAID内のコンポーネントパーティションごとに、RAIDからパーティションを削除し、そのサイズを変更し、パーティションをRAIDに戻し、RAIDが安定するまで待機してから続行します。パーティションが削除されている間、RAIDはディグレードモードで動作し、ディスクの耐障害性がまったくないか、または低下しています。複数の同時ディスク障害を許容できるRAIDの場合でも、一度に2つ以上のパーティションを削除しないでください。RAID用コンポーネントパーティションのサイズを増加させるには、次の手順に従います。

1. 端末コンソールを開きます。
2. 次のように入力して、RAIDアレイが一貫性を保っており、同期されていることを確認します。

```
tux > cat /proc/mdstat
```

このコマンドの出力によって、RAIDアレイがまだ同期中とわかる場合は、同期化の完了まで待って、続行してください。

3. コンポーネントパーティションの1つをRAIDアレイから削除します。たとえば、次のように入力して、/dev/sda1を削除します。

```
tux > sudo mdadm /dev/md0 --fail /dev/sda1 --remove /dev/sda1
```

成功するためには、failとremoveの両方のアクションを指定する必要があります。

4. 次のオプションの1つを実行して、前の手順で削除したパーティションのサイズを増加させます。
 - YaSTパーティショナやpartedなどのディスクパーティショナを使用して、パーティションのサイズを増やします。通常は、このオプションが選択されます。
 - パーティションの常駐ディスクを、容量のより大きいデバイスに置き換えます。このオプションは、元ディスクの他のファイルシステムがシステムによりアクセスされない場合だけ選択できます。置き換え用デバイスをRAIDに追加すると、元のデバイスにあったデータをすべて再構築しなければならないので、データの同期にはるかに長い時間がかかります。
5. パーティションをRAIDアレイに再追加します。たとえば、次のように入力して、/dev/sda1を追加します。

```
tux > sudo mdadm -a /dev/md0 /dev/sda1
```

RAIDが同期され、一貫性をもつまで待機してから、次のパーティションの処理に進みます。

6. アレイ内の残りのコンポーネントデバイスごとに、これらの手順を繰り返します。正しいコンポーネントパーティションに対して、必ずコマンドを変更してください。
7. カーネルがRAIDのパーティションテーブルを再読み込みできないというメッセージが表示されたら、すべてのパーティションのサイズ変更後にコンピュータを再起動して、パーティションテーブルの更新を強制する必要があります。
8. 11.1.2項「RAIDアレイのサイズの増加」に進みます。

11.1.2 RAIDアレイのサイズの増加

RAID内の各コンポーネントパーティションのサイズ変更後(11.1.1項「コンポーネントパーティションのサイズの増加」参照)も、新しい使用可能スペースの認識を強制するまで、RAIDアレイの設定では、元のアレイサイズが使用され続けます。RAIDアレイのサイズを指定したり、使用可能な最大スペースを使用できます。

本項の手順では、RAIDデバイスのデバイス名として `/dev/md0` を使用しています。この名前は変更して、必ずご使用のデバイスの名前を使用してください。

1. 端末コンソールを開きます。
2. 次のように入力して、RAIDアレイが一貫性を保っており、同期されていることを確認します。

```
tux > cat /proc/mdstat
```

このコマンドの出力によって、RAIDアレイがまだ同期中とわかる場合は、同期化の完了まで待って、続行してください。

3. 次のように入力して、アレイのサイズとアレイに認識されるデバイスサイズをチェックします。

```
tux > sudo mdadm -D /dev/md0 | grep -e "Array Size" -e "Dev Size"
```

4. 次のいずれかの操作を行います。

- 次のように入力して、アレイサイズを使用可能な最大サイズまで増加します。

```
tux > sudo mdadm --grow /dev/md0 -z max
```

- 次のように入力して、アレイサイズを使用可能な最大サイズまで増加します。

```
tux > sudo mdadm --grow /dev/md0 -z max --assume-clean
```

アレイは、デバイスに追加された領域を使用しますが、この領域は同期されません。これがRAID 1に推奨される理由は、同期が不要だからです。メンバーデバイスに追加されたスペースが事前にゼロ化されていれば、他のRAIDレベルに有用なことがあります。

- 次のように入力して、アレイサイズを指定の値まで増加します。

```
tux > sudo mdadm --grow /dev/md0 -z SIZE
```

SIZE を、キロバイト(1キロバイトは1024バイト)単位で目的のサイズを表す整数値で置き換えます。

5. 次のように入力して、アレイのサイズとアレイに認識されるデバイスサイズを再チェックします。

```
tux > sudo mdadm -D /dev/md0 | grep -e "Array Size" -e "Dev Size"
```

6. 次のいずれかの操作を行います。

- アレイのサイズ変更が成功していたら、[11.1.3項「ファイルシステムのサイズの増加」](#)を続行します。
- アレイが予想どおりにサイズ変更されていない場合は、いったん再起動してから、このプロシージャを再試行する必要があります。

11.1.3 ファイルシステムのサイズの増加

アレイサイズの増加後は([11.1.2項「RAIDアレイのサイズの増加」](#) 参照)、ファイルシステムのサイズ変更ができます。

ファイルシステムのサイズを使用可能な最大スペースまで増加したり、正確なサイズを指定できます。ファイルシステムに正確なサイズを指定する場合は、その新しいサイズが次の条件を満たすかどうかを必ず確認してください。

- 新しいサイズは、既存データのサイズより大きくなければなりません。さもないと、データが失われます。
- ファイルシステムのサイズは使用可能なスペースより大きくできないので、新しいサイズは、現在のRAIDサイズ以下でなければなりません。

詳しい手順については、[第2章「ファイルシステムのサイズ変更」](#)を参照してください。

11.2 ソフトウェアRAIDのサイズの削減

ソフトウェアRAIDのサイズを減らすには、複数のタスクを所定の順序で実行する必要があります。まずファイルシステムのサイズを縮小し、次にRAIDを構成するすべてのパーティションのサイズを縮小します。そして最後に、RAID自体のサイズを縮小します。



警告: データ消失の可能性

RAIDに、ディスクの耐障害性がないか、単に一貫性がない場合、パーティションのどれかを削除すると、データが失われます。パーティションの削除は注意深く行い、必ず、データのバックアップをとってください。



重要: XFS

XFSでフォーマットされたファイルシステムのサイズを縮小することはできません。XFSではそのような機能がサポートされていないためです。そのため、XFSファイルシステムを使用するRAIDのサイズを縮小することはできません。

11.2.1 ファイルシステムのサイズの削減

RAIDデバイス上のファイルシステムのサイズを削減する際には、新しいサイズが次の条件を満たすかどうかを必ず確認してください。

- 新しいサイズは、既存データのサイズより大きくなければなりません。さもないと、データが失われます。
- ファイルシステムのサイズは使用可能なスペースより大きくできないので、新しいサイズは、現在のRAIDサイズ以下でなければなりません。

詳しい手順については、[第2章「ファイルシステムのサイズ変更」](#)を参照してください。

11.2.2 RAIDアレイサイズの削減

ファイルシステムのサイズ変更後([11.2.1項「ファイルシステムのサイズの削減」](#)を参照)、RAIDアレイ設定では、利用可能スペースを縮小するよう強制するまで、元のアレイサイズを使い続けます。RAIDが、削減したセグメントサイズを使用するようにするには、`mdadm --grow` モードを使用します。それを行うには、`-z` オプションを使用して、RAID内の各デバイ

スが使用するスペースの量を、キロバイトで指定する必要があります。このサイズは、チャンクサイズの倍数である必要があり、RAIDのスーパーブロックをデバイスに書き込むためのスペースとして、約128KBを残しておかなければなりません。

本項の手順では、RAIDデバイスのデバイス名として `//dev/md0` を使用しています。コマンドを変更して、必ずご使用のデバイスの名前を使用してください。

1. 端末コンソールを開きます。
2. 次のように入力して、アレイのサイズとアレイに認識されるデバイスサイズをチェックします。

```
tux > sudo mdadm -D /dev/md0 | grep -e "Array Size" -e "Dev Size"
```

3. 次のコマンドで、アレイのデバイスサイズを指定の値まで減少させます。

```
tux > sudo mdadm --grow /dev/md0 -z SIZE
```

SIZE を、キロバイト単位で目的のサイズを表す整数値で置き換えます。(1キロバイトは1024バイト)。

たとえば、次のコマンドでは、各RAIDデバイスのセグメントサイズを約40 GBに設定し、チャンクサイズは64 KBです。これには、RAIDのスーパーブロック用の128 KBが含まれます。

```
tux > sudo mdadm --grow /dev/md2 -z 41943168
```

4. 次のように入力して、アレイのサイズとアレイに認識されるデバイスサイズを再チェックします。

```
tux > sudo mdadm -D /dev/md0 | grep -e "Array Size" -e "Device Size"
```

5. 次のいずれかの操作を行います。

- アレイのサイズ変更が成功していたら、[11.2.3項「コンポーネントパーティションのサイズの削減」](#)を続行します。
- アレイが予想どおりにサイズ変更されていない場合は、いったん再起動してから、このプロシージャを再試行する必要があります。

11.2.3 コンポーネントパーティションのサイズの削減

RAID内の各デバイスで使用するセグメントサイズの縮小後(11.2.2項「RAIDアレイサイズの削減」を参照)、各コンポーネントパーティション内の残りのスペースは、そのRAIDでは使われません。パーティションを現在のサイズのまま残して将来のRAIDの拡大に備えることも、今は使用しないそのスペースを利用することもできます。

そのスペースを利用するには、コンポーネントパーティションを1つずつ削減します。コンポーネントパーティションごとに、そのパーティションをRAIDから削除し、パーティションサイズを縮小し、パーティションをRAIDに戻したら、RAIDが安定するまで待機します。メタデータに備えるには、11.2.2項「RAIDアレイサイズの削減」でRAIDに対して指定したサイズより、若干大きなサイズを指定する必要があります。

パーティションが削除されている間、RAIDはディグレードモードで動作し、ディスクの耐障害性がまったくないか、または低下しています。複数の同時ディスクエラーに耐えるRAIDの場合でも、一度に2つ以上のコンポーネントパーティションを削除しないでください。RAID用コンポーネントパーティションのサイズを縮小するには、次の手順に従います。

1. 端末コンソールを開きます。
2. 次のように入力して、RAIDアレイが一貫性を保っており、同期されていることを確認します。

```
tux > cat /proc/mdstat
```

このコマンドの出力によって、RAIDアレイがまだ同期中とわかる場合は、同期化の完了まで待って、続行してください。

3. コンポーネントパーティションの1つをRAIDアレイから削除します。たとえば、次のように入力して、/dev/sda1を削除します。

```
tux > sudo mdadm /dev/md0 --fail /dev/sda1 --remove /dev/sda1
```

成功するためには、failとremoveの両方のアクションを指定する必要があります。

4. 前の手順で削除したパーティションのサイズを、セグメントサイズに設定したサイズより若干小さいサイズに減らします。このサイズは、チャンクサイズの倍数であり、RAIDのスーパーブロック用に128 KBを確保する必要があります。YaSTパーティショナやコマンドラインツールpartedなどを使用して、パーティションのサイズを縮小します。
5. パーティションをRAIDアレイに再追加します。たとえば、次のように入力して、/dev/sda1を追加します。

```
tux > sudo mdadm -a /dev/md0 /dev/sda1
```

RAIDが同期され、一貫性をもつまで待機してから、次のパーティションの処理に進みます。

6. アレイ内の残りのコンポーネントデバイスごとに、これらの手順を繰り返します。正しいコンポーネントパーティションに対して、必ずコマンドを変更してください。
7. カーネルがRAIDのパーティションテーブルを再読み込みできないというメッセージが表示されたら、すべてのパーティションのサイズ変更後にコンピュータを再起動する必要があります。
8. (オプション)RAIDとファイルシステムのサイズを拡大して、現在は小さめのコンポーネントパーティション内のスペースの最大量を利用し、後でファイルシステムのサイズを増やします。手順については、[11.1.2項「RAIDアレイのサイズの増加」](#)を参照してください。

12 MDソフトウェアRAID用のストレージエンクロージャLEDユーティリティ

ストレージエンクロージャLEDモニタリングユーティリティ(**ledmon**)およびLEDコントロール(**ledctl**)ユーティリティは、多様なインタフェースおよびプロトコルを使用してストレージエンクロージャLEDを制御する、Linuxのユーザスペースアプリケーションです。その主たる用途は、mdadmユーティリティで作成されたLinux MDソフトウェアのRAIDデバイスの状態を視覚化することです。**ledmon** デーモンがドライブアレイの状態を監視し、ドライブLEDの状態を更新します。**ledctl** ユーティリティを使用して、指定したデバイスに対するLEDパターンを設定できます。

これらのLEDユーティリティでは、SGPIO (Serial General Purpose Input/Output)仕様(Small Form Factor (SFF) 8485)およびSCSI Enclosure Services (SES) 2プロトコルを使用して、LEDを制御します。SGPIO用のSFF-8489仕様のInternational Blinking Pattern Interpretation (IBPI)パターンを実装します。IBPIは、SGPIO規格がバックプレーン上のドライブやスロットの状態としてどのように解釈されるか、またバックプレーンがLEDでどのように状態を視覚化すべきかを定義します。

一部のストレージエンクロージャでは、SFF-8489仕様に厳格に準拠していないものがあります。エンクロージャプロセッサがIBPIパターンを受け入れていても、LEDの点滅はSFF-8489仕様に従っていない、あるいはプロセッサが限られた数のIBPIパターンしかサポートしていない場合があります。

LED管理(AHCI)およびSAF-TEプロトコルは、**ledmon** および **ledctl** ユーティリティではサポートされていません。

ledmon および **ledctl** アプリケーションは、インテルAHCIコントローラやインテルSASコントローラなどの、インテルのストレージコントローラで機能することが検証されています。MDソフトウェアのRAIDボリュームの一部であるPCIe-SSD(ソリッドステートドライブ)デバイスの、ストレージエンクロージャ状態(OK、Fail、Rebuilding)用LEDを制御するための、PCIe-SSD(ソリッドステートディスク)エンクロージャLEDもサポートされています。これらのアプリケーションは、他のベンダのIBPI準拠のストレージコントローラ(特にSAS/SCSIコントローラ)でも機能するはずですが、他のベンダのコントローラはテストされていません。

ledmon および **ledctl** は **ledmon** パッケージに付属しています。このパッケージはデフォルトではインストールされません。インストールするには、**sudo zypper in ledmon** を実行します。

12.1 ストレージエンクロージャLED監視サービス

`ledmon` アプリケーションは、MDソフトウェアRAIDデバイスの状態またはストレージエンクロージャまたはドライブベイ内のブロックデバイスの状態をコンスタントに監視する、デーモンプロセスです。一度に実行しているデーモンのインスタンスは、1つのみである必要があります。`ledmon` デーモンは、インテルのエンクロージャLEDユーティリティの一部です。

状態は、ストレージレイエンクロージャまたはドライブベイ内の、各スロットに関連付けられたLED上で視覚化されます。このアプリケーションは、すべてのソフトウェアRAIDデバイスを監視し、その状態を視覚化します。選択したソフトウェアRAIDボリュームのみを監視する方法は、備わっていません。

`ledmon` デーモンでは、2種類のLEDシステム、すなわち、2 LEDシステム(Activity LEDと Status LED)と、3 LEDシステム(Activity LED、Locate LED、およびFail LED)をサポートしています。このツールには、LEDへのアクセスの際に最高の優先度が与えられています。

`ledmon` を起動するには、次のように入力します。

```
tux > sudo ledmon [options]
```

[options]は次の1つ以上です。

`ledmon`のオプション

`-c PATH` ,

`--config=PATH`

設定は `~/.ledctl` または `/etc/ledcfg.conf` (存在する場合)から読み込まれます。このオプションは、別の設定ファイルを指定する場合に使用します。

現時点では、複数の設定ファイルのサポートはまだ実装されていないため、このオプションは有効ではありません。詳細については、[`man 5 ledctl.conf`](#)を参照してください。

`-l PATH` ,

`--log=PATH`

ローカルのログファイルへのパスを設定します。このユーザ定義ファイルを指定すると、グローバルログファイル `/var/log/ledmon.log` は使用されません。

`-t SECONDS` ,

`--interval=SECONDS`

`sysfs` のスキャン間の時間間隔を設定します。値は秒単位です。最小値は5秒です。最大値の指定はありません。

--quiet、--error、--warning、--info、--debug、--all

詳細レベルを指定します。このレベルオプションは、情報なしから、ほとんどの情報までの順番で指定されます。ロギングを行わない場合は、--quiet オプションを使用します。すべてをログする場合は、--all オプションを使用します。2つ以上の詳細オプションを指定した場合は、コマンド内の最後のオプションが適用されます。

-h ,

--help

コマンド情報をコンソールに印刷して、終了します。

-v ,

--version

ledmon のバージョンとライセンスに関する情報を表示して、終了します。



注記: 既知の問題

ledmon デーモンは、SFF-8489仕様のPFA (Predicted Failure Analysis)状態は認識しません。したがって、PFAパターンは視覚化されません。

12.2 ストレージエンクロージャLED制御アプリケーション

エンクロージャLEDアプリケーション(ledctl)は、ストレージエンクロージャまたはドライブベイの各スロットに関連付けられたLEDを制御する、ユーザスペースアプリケーションです。ledctl アプリケーションは、インテルのエンクロージャLEDユーティリティの一部です。

このコマンドを発行すると、指定したデバイスのLEDが指定したパターンに設定され、それ以外のLEDはすべてオフになります。このアプリケーションは root 特権で実行する必要があります。ledmon アプリケーションはLEDへのアクセスに際して最高の優先度を持っているため、ledmon デーモンを実行中の場合は、ledctl で設定した一部のパターンが有効にならないことがあります(Locateパターン以外)。

ledctl アプリケーションでは、2種類のLEDシステム、すなわち、2LEDシステム(Activity LEDとStatus LED)と、3LEDシステム(Activity LED、Locate LED、およびFail LED)をサポートしています。

ledctl を起動するには、次のように入力します。

```
tux > sudo [options] PATTERN_NAME=list_of_devices
```

[options]は次の1つ以上です。

-c PATH ,

--config=PATH

ローカルの環境設定ファイルへのパスを設定します。このオプションを指定すると、グローバルの環境設定ファイルとユーザの環境設定ファイルは、無効になります。

-l PATH ,

--log=PATH

ローカルのログファイルへのパスを設定します。このユーザ定義ファイルを指定すると、グローバルログファイル /var/log/ledmon.log は使用されません。

--quiet

stdout または stderr に送信されるすべてのメッセージをオフにします。メッセージは、ローカルファイルおよび syslog ファシリティには引き続きログされます。

-h ,

--help

コマンド情報をコンソールに印刷して、終了します。

-v ,

--version

ledctl のバージョンとライセンスに関する情報を表示して、終了します。

12.2.1 パターン名

ledctl アプリケーションでは、SFF-8489仕様に従い、pattern_name 引数に次の名前を使用できます。

locate

指定したデバイスまたはからのスロットに関連付けられたLocate LEDを点灯します。この状態は、スロットまたはドライブの識別に使用されます。

locate_off

指定したデバイスまたはからのスロットに関連付けられたLocate LEDを消灯します。

normal

指定したデバイスに関連付けられたStatus LED、Failure LED、およびLocate LEDを消灯します。

off

指定したデバイスに関連付けられたStatus LEDとFailure LEDのみを消灯します。

ica ,

degraded

In a Critical Array パターンを視覚化します。

再構築 ,

rebuild_p

Rebuild パターンを視覚化します。互換性とレガシーの理由から、両方のrebuild状態をサポートしています。

ifa ,

failed_array

In a Failed Array パターンを視覚化します。

hotspare

Hotspare パターンを視覚化します。

pfa

Predicted Failure Analysis パターンを視覚化します。

failure ,

disk_failed

Failure パターンを視覚化します。

ses_abort

SES-2 R/R ABORT

ses_rebuild

SES-2 REBUILD/REMAP

ses_ifa

SES-2 IN FAILED ARRAY

ses_ica

SES-2 IN CRITICAL ARRAY

ses_cons_check

SES-2 CONS CHECK

<u>ses_hotspare</u>	SES-2 HOTSPARE
<u>ses_rsvd_dev</u>	SES-2 RSVD DEVICE
<u>ses_ok</u>	SES-2 OK
<u>ses_ident</u>	SES-2 IDENT
<u>ses_rm</u>	SES-2 REMOVE
<u>ses_insert</u>	SES-2 INSERT
<u>ses_missing</u>	SES-2 MISSING
<u>ses_dnr</u>	SES-2 DO NOT REMOVE
<u>ses_active</u>	SES-2 ACTIVE
<u>ses_enable_bb</u>	SES-2 ENABLE BYP B
<u>ses_enable_ba</u>	SES-2 ENABLE BYP A
<u>ses_devoff</u>	SES-2 DEVICE OFF
<u>ses_fault</u>	SES-2 FAULT

非SES-2のパターンがエンクロージャ内のデバイスに送信されると、そのパターンは、上に示すように、SCSI Enclosure Services (SES) 2のパターンに自動的に変換されます。

表 12.1: 非SES-2パターンとSES-2パターン間での変換

非SES-2のパターン	SES-2のパターン
locate	ses_ident
locate_off	ses_ident
normal	ses_ok
off	ses_ok
ica	ses_ica
degraded	ses_ica
rebuild	ses_rebuild
rebuild_p	ses_rebuild
ifa	ses_ifa
failed_array	ses_ifa
hotspare	ses_hotspare
pfa	ses_rsvd_dev
failure	ses_fault
disk_failed	ses_fault

12.2.2 デバイスのリスト

ledctl コマンドを発行すると、指定したデバイスのLEDが指定したパターンに設定され、それ以外のLEDはすべてオフになります。デバイスのリストは、次の2つの形式のいずれかで提供できます。

- スペースなしのカンマで区切られたデバイスのリスト
- デバイスがスペースで区切られた波括弧内のリスト

同じコマンド内で複数のパターンを指定すると、各パターンに対するデバイスリストで、同一または異なるフォーマットを使用できます。2つのリスト形式を示す例は、[12.2.3項「例」](#)を参照してください。

デバイスは、`/dev` ディレクトリまたは `/sys/block` ディレクトリ内のファイルへのパスです。パスにより、ブロックデバイス、MDソフトウェアRAIDデバイス、またはコンテナデバイスを識別できます。ソフトウェアRAIDデバイスまたはコンテナデバイスの場合、報告されたLEDの状態は、関連付けられたブロックデバイスのすべてに対して設定されます。

`list_of_devices` にリストされているデバイスのLEDは、特定のパターンの `pattern_name` に設定され、それ以外のすべてのLEDは消灯されます。

12.2.3 例

単一のブロックデバイスを見つけるには

```
tux > sudo ledctl locate=/dev/sda
```

単一のブロックデバイスのLocate LEDを消灯するには

```
tux > sudo ledctl locate_off=/dev/sda
```

MDソフトウェアRAIDデバイスのディスクを見つけて、そのブロックデバイスの2つに同時に `rebuild` パターンを設定するには

```
tux > sudo ledctl locate=/dev/md127 rebuild={ /sys/block/sd[a-b] }
```

指定したデバイスに対するStatus LEDとFailure LEDを消灯するには

```
tux > sudo ledctl off={ /dev/sda /dev/sdb }
```

3つのブロックデバイスを見つけるには、次のいずれかのコマンドを実行します(どちらのコマンドでも同じです)。

```
tux > sudo ledctl locate=/dev/sda,/dev/sdb,/dev/sdc
tux > sudo ledctl locate={ /dev/sda /dev/sdb /dev/sdc }
```

12.3 追加情報

LEDのパターンおよび監視ツールに関する詳細は、次のリソースを参照してください。

- Sourceforge.net上のLEDMONオープンソースプロジェクト (<http://sourceforge.net/projects/ledmon/>) 
- SGPIO仕様SFF-8485 (<https://ftp.seagate.com/sff/SFF-8485.PDF>) 
- IBPI仕様SFF-8489 (<https://ftp.seagate.com/sff/SFF-8489.PDF>) 

IV ネットワークストレージ

- 13 Linux用iSNS **139**
- 14 IPネットワークの大容量記憶域 - iSCSI **147**
- 15 Fibre Channel Storage over Ethernet Networks: FCoE **171**
- 16 NVMe over Fabric **182**
- 17 デバイスのマルチパスI/Oの管理 **189**
- 18 NFSv4上でのアクセス制御リストの管理 **252**

13 Linux用iSNS

ストレージエリアネットワーク(SAN)には、複数のネットワークにまたがる多数のディスクドライブを使用できます。これによって、デバイス検出とデバイスの所有権の判定が難しくなります。iSCSIイニシエータはSANのストレージリソースを識別し、どれにアクセスできるか判定できる必要があります。

Internet Storage Name Service(iSNS)は、TCP/IPネットワーク上のiSCSIデバイスの自動化された検出、管理、および設定を簡素化する、標準ベースのサービスです。iSNSでは、ファイバチャネルネットワークと同等の知的なストレージの検出および管理のサービスを提供します。

！ 重要: セキュリティ上の考慮事項

iSNSは安全な社内ネットワークだけで使用してください。

13.1 iSNSのしくみ

iSCSIイニシエータがiSCSIターゲットを検出するには、ネットワークのどのデバイスがストレージリソースで、アクセスするにはどのIPアドレスが必要かを特定する必要があります。iSNSサーバへクエリすると、iSCSIターゲットとイニシエータがアクセス許可をもつIPアドレスのリストが返されます。

iSNSを使用してiSNS検出ドメインを作成し、そこにiSCSIターゲットとiSCSIイニシエータをグループ化または構成します。多くのストレージノードを複数のドメインに振り分けることで、各ホストの検出プロセスをiSNSで登録された最適なターゲットのサブセットに限定でき、これによって、不要な検出を削減し、各ホストが検出関係の確立に費やす時間を制限することで、ストレージネットワークの規模を調整できるようになります。このようにして、ディスクカバリ対象のターゲットとイニシエータの数を制御し、簡略化できます。

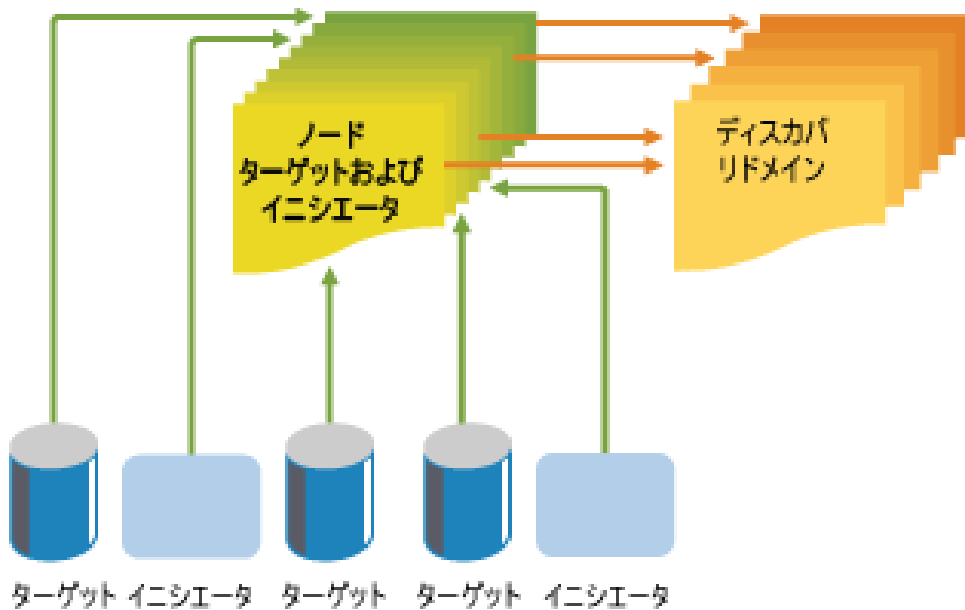


図 13.1: iSNS検出ドメイン

iSCSIターゲットとiSCSIイニシエータは両方とも、iSNSクライアントを使用して、iSNSプロトコルによるiSNSサーバとのトランザクションを開始します。iSCSIターゲットとiSCSIイニシエータは、次にデバイス属性情報を共通検出ドメインに登録し、その他の登録されたクライアント情報をダウンロードし、検出ドメインで発生したイベントの非同期通知を受け取ります。

iSNSサーバは、iSNSプロトコルクエリとiSNSクライアントがiSNSプロトコルを使用して作成した要求に応答します。iSNSサーバはiSNSプロトコル状態変更通知を開始し、登録要求から送られてきた適切に認証された情報をiSNSデータベースに保存します。

Linux用iSNSは次の利点をもたらします。

- ネットワーク接続させたストレージ資産の登録、検出、管理に役立つ情報を提供する。
- DNSインフラストラクチャと統合する。
- iSCSIストレージの登録、検出、管理を統合する。
- ストレージ管理の実装が簡素化される。
- その他のディスカバリ方法よりもスケーラビリティが向上する。

次のシナリオは、iSNSのメリットについて具体的に説明したものです。

100個のiSCSIイニシエータと100個のiSCSIターゲットが会社にあるとします。設定によっては、すべてのiSCSIイニシエータが100個のiSCSIターゲットを検出して接続しようとする可能性があります。このため、検出や接続が困難になる場合があります。イニシエータとターゲット

トをいくつかの検出ドメインにグループ化することで、ある部門のiSCSIイニシエータが別の部門のiSCSIターゲットを検出しないようにできます。その結果、特定部門のiSCSIイニシエータは、その部門の検出ドメインに属するiSCSIターゲットしか検出しません。

13.2 Linux用iSNSサーバのインストール

Linux用iSNSサーバは、SUSE Linux Enterprise Serverに付属していますが、デフォルトではインストールも設定も行われません。パッケージ `open-isns` をインストールして、iSNSサービスを設定する必要があります。



注記: 同一サーバ上のiSNSとiSCSI

iSNSは、iSCSIターゲットまたはiSCSIイニシエータのソフトウェアがインストールされる同じサーバにインストールできます。ただし、iSCSIターゲットソフトウェアとiSCSIイニシエータソフトウェアの両方を同じサーバにインストールすることはできません。

Linux向けiSNSをインストールするには、次の手順に従います。

1. YaSTを起動してネットワークサービス > iSNSサーバを選択します。
2. `open-isns` がまだインストールされていない場合、今すぐインストールするようプロンプトが表示されます。インストールをクリックして確認します。
3. iSNSサービスの設定ダイアログが表示され、自動的にサービスタブが開きます。



4. サービスの開始で、次のいずれかを選択します。

- **ブート時:** iSNSサービスは、サーバの起動時に自動的に開始します。
- **手動(デフォルト):** iSNSのインストール先サーバのサーバコンソールで、「**`sudo systemctl start isnsd`**」と入力して、iSNSサービスを手動で開始する必要があります。

5. 次のファイアウォール設定を指定します。

- **Open Port in Firewall(ファイアウォールのポートを開く):** このチェックボックスを選択して、ファイアウォールを開き、リモートコンピュータからサービスにアクセスできるようにします。ファイアウォールのポートは、デフォルトでは閉じています。
- **ファイアウォールの詳細:** ファイアウォールのポートを開いた場合、デフォルトでは、ポートがすべてのネットワークインタフェースで開きます。ポートを開くインタフェースを選択するには、Firewall Details(ファイアウォールの詳細)をクリックし、使用するネットワークインタフェースを選択し、次に、OKをクリックします。

6. OKをクリックして、設定を適用し、インストールを完了します。

7. 13.3項「iSNS検出ドメインの設定」に進みます。

13.3 iSNS検出ドメインの設定

iSCSIイニシエータおよびターゲットでiSNSサービスを使用するには、これらが検出ドメインに属している必要があります。

！ 重要: iSNSサービスがアクティブである必要がある

iSNS検出ドメインを設定するには、iSNSサービスがインストール済みで、実行されている必要があります。詳細については、[13.4項「iSNSサービスの開始」](#)を参照してください。

13.3.1 iSNS検出ドメインの作成

iSNSサービスをインストールすると、デフォルトDDというデフォルトの検出ドメインが自動的に作成されます。iSNSを使用するように設定されている既存のiSCSIターゲットとイニシエータは、デフォルト検出ドメインに自動的に追加されます。

新しい検出ドメインを作成するには、次の手順に従います。

1. YaSTを起動して、ネットワークサービスの下でiSNSサーバを選択します。

2. 検出ドメインタブをクリックします。

検出ドメイン領域に既存のすべての検出ドメインが一覧にされます。Create Discovery Domains (検出ドメインの作成)で検出ドメインを作成したり、削除で既存の検出ドメインを削除したりできます。ドメインを削除すると、ドメインのメンバーは削除されますが、iSCSIノードメンバーは削除されません。

検出ドメインメンバーの領域に、選択した検出ドメインに割り当てられているすべてのiSCSIノードがリストされます。別の検出ドメインを選択すると、その検出ドメインからのメンバーで、リストが更新されます。選択した検出ドメインからiSCSIノードを追加したり、削除できます。iSCSIノードを削除すると、そのノードは、ドメインから削除されますが、iSCSIノード自体は削除されません。

iSCSIノードメンバーの作成を使用すると、未登録のノードを検出ドメインのメンバーとして追加できます。iSCSIイニシエータまたはiSCSIターゲットがこのノードを登録すると、このノードは、このドメインの一部となります。

iSCSIイニシエータが検出要求を発行すると、iSNSサービスは同じ検出ドメイン内のメンバーであるすべてのiSCSIノードターゲットを返します。

3. 検出ドメインの作成ボタンをクリックします。
既存の検出ドメインを選択して削除ボタンをクリックして、その検出ドメインを削除できます。
4. 作成している検出ドメインの名前を指定して、OKをクリックします。
5. 13.3.2項「iSCSIノードの検出ドメインへの追加」に進みます。

13.3.2 iSCSIノードの検出ドメインへの追加

1. YaSTを起動して、ネットワークサービスの下でiSNSサーバを選択します。
2. iSCSIノードタブをクリックします。



3. ノードのリストをレビューして、iSNSサービスを使用させたいiSCSIターゲットおよびイニシエータがリストされていることを確認します。
iSCSIターゲットまたはイニシエータが一覧にない場合、ノード上のiSCSIサービスを再起動する必要があります。それには以下を実行して、

```
tux > sudo systemctl restart iscsid.socket  
tux > sudo systemctl restart iscsi
```

イニシエータまたは

```
tux > sudo systemctl restart target-isns
```

ターゲットを再起動します。

iSCSIノードを選択して削除ボタンをクリックして、そのノードをiSNSデータベースから削除できます。iSCSIノードをもう使用しない場合や名前を変更した場合に有効です。iSCSI環境設定ファイルのiSNSの部分を削除したりコメント化していない限り、iSCSIノードは、iSCSIサービスの再開始時またはサーバの再起動時に、リスト(iSNSデータベース)に自動的に追加されます。

4. 検出ドメインタブをクリックして、目的の検出ドメインを選択します。
5. Add existing iSCSI Nodeをクリックしてドメインに追加するノードを選択し、ノードの追加をクリックします。

6. 検出ドメインに追加するノードの数だけ最後の手順を繰り返し、ノードの追加が終了したら完了をクリックします。

iSCSIノードは複数の検出ドメインに属することができます。

13.4 iSNSサービスの開始

iSNSは、インストール先のサーバで起動する必要があります。まだ起動時に開始するように設定していない場合(詳細については13.2項「Linux用iSNSサーバのインストール」を参照)、端末コンソールで次のコマンドを入力します。

```
tux > sudo systemctl start isnsd
```

iSNSでは、**stop**、**status**、**restart**の各オプションも使用できます。

13.5 その他の情報

詳細については、「Linux iSNS for iSCSI project」(<http://sourceforge.net/projects/linuxisns/>)を参照してください。このプロジェクトの電子メーリングリストがhttp://sourceforge.net/mailarchive/forum.php?forum_name=linuxisns-discussionにあります。

iSNSの一般情報は、「RFC 4171: Internet Storage Name Service」(<http://www.ietf.org/rfc/rfc4171>)に記載されています。

14 IPネットワークの大容量記憶域 - iSCSI

コンピュータセンターや、サーバをサポートするサイトの主要タスクの1つは、適切なディスク容量を提供することです。この用途には、多くの場合、ファイバチャネルが使用されます。iSCSI(Internet SCSI)ソリューションは、ファイバチャネルに対する低コストの代替であり、コモディティサーバおよびEthernetネットワーク装置を活用することができます。Linux iSCSIは、iSCSIイニシエータおよびiSCSI LIOターゲットのソフトウェアの提供により、Linuxサーバを中央ストレージシステムに接続します。

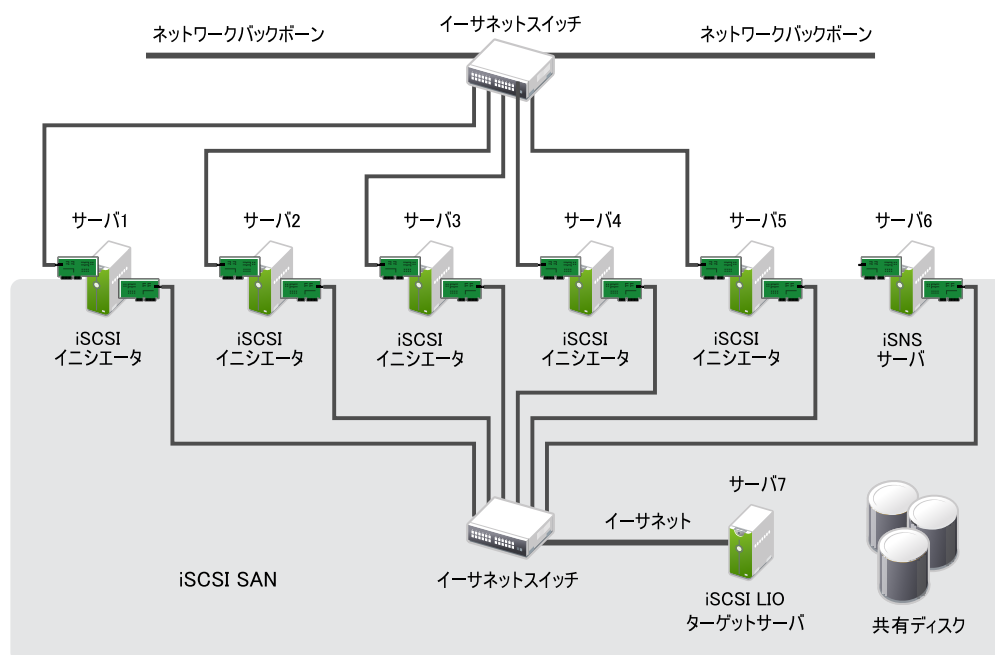


図 14.1: iSNSサーバによるiSCSI SAN



注記: LIO

LIO(<http://linux-iscsi.org>)は、Linux用の標準のオープンソースマルチプロトコル SCSIターゲットです。LIOは、Linuxカーネルのバージョン2.6.38以降において、Linuxにおける標準の統一ストレージターゲットとして、STGT (SCSI Target) フレームワークにとって代わりました。SUSE Linux Enterprise Server 12では、古いバージョンのiSCSIターゲットサーバにiSCSI LIOターゲットサーバに代わっています。

iSCSIは、ストレージネットワークングプロトコルであり、ブロックストレージデバイスとサーバ間における、TCP/IPネットワーク上でのSCSIパケットのデータ転送を簡素化にします。iSCSIターゲットソフトウェアは、ターゲットサーバ上で実行され、論理ユニットをiSCSI

ターゲットデバイスとして定義します。iSCSIイニシエータソフトウェアは異なるサーバ上で実行され、ターゲットデバイスに接続して、そのサーバ上でストレージデバイスを使用できるようにします。

iSCSI LIOターゲットサーバおよびiSCSIイニシエータサーバは、LAN内のIPレベルでSCSIパケットを送信して通信します。イニシエータサーバ上のアプリケーションがiSCSI LIOターゲットデバイスに対する照会を開始すると、オペレーティングシステムが必要なSCSIコマンドを発行します。するとSCSIコマンドが、iSCSIイニシエータと呼ばれるソフトウェアによってIPパケットに組み込まれ、必要に応じて暗号化されます。パケットは内部IPネットワーク上で、「iSCSI LIOターゲットサーバ」または単に「iSCSIターゲット」と呼ばれる、対応するiSCSIリモートステーションに転送されます。

多くのストレージソリューションが、iSCSIによるアクセス手段を提供しています。また、LinuxサーバにiSCSIターゲットの役割をさせることもできます。この場合、Linuxサーバをファイルシステムサービス用に最適化しておくことが重要です。iSCSIターゲットはLinux内のブロックデバイスにアクセスします。したがって、RAIDソリューションを使用することでディスク容量を増やし、メモリも増量してデータキャッシングを向上させることができます。RAIDの詳細については、[第7章「ソフトウェアRAIDの設定」](#)も参照してください。

14.1 iSCSI LIOターゲットサーバとiSCSIイニシエータのインストール

iSCSIイニシエータはデフォルトでインストールされますが(`open-iscsi` パッケージと `yast2-iscsi-client` パッケージ)、iSCSI LIOターゲットパッケージは手動でインストールする必要があります。



重要: イニシエータとターゲットを同一サーバで実行することはできない

iSCSIターゲットソフトウェアとiSCSIイニシエータソフトウェアを、運用環境内の同じサーバ上で実行することはできません。

iSCSI LIOターゲットサーバをインストールするには、端末コンソールで次のコマンドを実行します。

```
tux > sudo zypper in yast2-iscsi-lio-server
```

iSCSIイニシエータまたはその依存関係をインストールする必要がある場合は、コマンド `sudo zypper in yast2-iscsi-client` を実行します。

または、YaSTソフトウェア管理モジュールを使用してインストールします。

先に示したパッケージ以外の必要なパッケージは、インストーラによって自動的に組み込まれるか、またはそれぞれのYaSTモジュールの初回実行時にインストールされます。

14.2 iSCSI LIOターゲットサーバのセットアップ

本項では、YaSTを使用してiSCSI LIOターゲットサーバを構成し、iSCSI LIOのターゲットデバイスを設定する方法について説明します。任意のiSCSIイニシエータソフトウェアを使用して、ターゲットデバイスにアクセスすることができます。

14.2.1 iSCSI LIOターゲットサービスの起動およびファイアウォールの設定

iSCSI LIOターゲットサービスは、マニュアルで開始するようデフォルトで設定されています。同サービスを、システムのブート時に自動的に開始するよう設定できます。サーバでファイアウォールを使用していて、iSCSI LIOターゲットをほかのコンピュータでも利用可能としたい場合は、ターゲットへのアクセスに使用する各アダプタ用に、ファイアウォール内のポートを開放する必要があります。TCPポート3260が、iSCSIプロトコル用のポート番号です。これは、IANA (Internet Assigned Numbers Authority)により定義されています。

1. YaSTを起動し、ネットワークサービス > iSCSI LIOターゲットの順に起動します。
2. サービスタブに切り替えます。

The screenshot shows the 'iSCSI LIO ターゲットの概要' (iSCSI LIO Target Overview) window with the 'サービス' (Services) tab selected. The 'サービスの開始' (Start Service) section has two radio buttons: 'ブート時(B)' (At Boot) and '手動(M)' (Manual), with '手動(M)' selected. Below this, the 'SuSEfirewall2用のファイアウォール設定' (Firewall configuration for SuSEfirewall2) section has a checkbox 'ファイアウォールでポートを開く(F)' (Open port in firewall) which is unchecked. To the right of the checkbox is a button 'ファイアウォールの詳細(D)...' (Firewall details...). Below the checkbox is the text '設定済みネットワークインターフェースがない' (No configured network interfaces). At the bottom of the window are four buttons: 'ヘルプ(H)' (Help), '中止(R)' (Cancel), '次へ(N)' (Next), and '完了(F)' (Finish).

3. サービスを開始で、iSCSI LIOターゲットサービスの開始方法を指定します。

- **ブート時:** サービスは、サーバの再起動時に自動的に開始します。
- **手動:** (デフォルト)サーバの再起動後、`sudo systemctl start target` コマンドを実行して、手動でサービスを開始する必要があります。ターゲットデバイスは、サービスを開始するまで利用できません。

4. サーバでファイアウォールを使用していて、iSCSI LIOターゲットをほかのコンピュータでも利用可能としたい場合は、ターゲットへのアクセスに使用する各アダプタインターフェース用に、ファイアウォール内のポート3260を開放します。このポートがネットワークインターフェースのすべてに対してクローズしている場合、iSCSI LIOターゲットはほかのコンピュータでは利用できません。

サーバでファイアウォールを使用していない場合、ファイアウォール設定は無効です。この場合、次の手順をスキップして、終了を使用して設定ダイアログから移動するか、別のタブに切り替えて設定を続行します。

- a. サービスタブで、ファイアウォールのポートを開くチェックボックスをオンにして、ファイアウォール 設定を有効にします。

- b. ファイアウォールの詳細をクリックして、使用するネットワークインタフェースを確認または設定します。すべての利用可能なネットワークインタフェースが一覧表示され、デフォルトではすべてが選択されています。ポートを開く必要がないすべてのインタフェースを選択解除します。OKをクリックして設定を保存します。

5. 完了をクリックして、iSCSI LIOターゲットサービスの設定を保存して適します。

14.2.2 iSCSI LIOターゲットおよびイニシエータのディスカバリに対する認証の設定

iSCSI LIOターゲットサーバソフトウェアは、PPP-CHAP (Point-to-Point Protocol Challenge Handshake Authentication Protocol)をサポートしています。これは、Internet Engineering Task Force (IETF) RFC 1994 (<http://www.ietf.org/rfc/rfc1994.txt>)で定義されている、3方向の認証方法です。サーバはこの認証方法を、ターゲット上のファイルへのアクセスにではなく、iSCSI LIOのターゲットとイニシエータのディスカバリ用に使用します。ディスカバリへのアクセスを制限しない場合は、認証なしを選択します。デフォルトでは検出認証なしオプションが有効になっています。このサーバ上のすべてのiSCSI LIOターゲットは、認証を要求しないので、同じネットワーク上のどのiSCSIイニシエータによっても検出することができます。

よりセキュアな設定に対する認証が必要な場合は、incoming認証、outgoing認証またはその両方を使用できます。イニシエータによる認証では、iSCSIイニシエータに、iSCSI LIOターゲット上で検出を実行するパーミッションがあることを証明するよう求めます。イニシエータは、incomingのユーザ名とパスワードを入力する必要があります。ターゲットによる認証では、iSCSI LIOターゲットに、自らが目的のターゲットであることをイニシエータに対して証明するよう求めます。iSCSI LIOターゲットは、outgoingのユーザ名とパスワードを、iSCSIイニシエータに提供する必要があります。パスワードはincomingとoutgoingのディスカバリで異なる必要があります。ディスカバリに対する認証を有効にしない場合、その設定は、すべてのiSCSI LIOターゲットグループに適用されます。



重要: セキュリティ

セキュリティ上の理由により、運用環境では、ターゲットおよびイニシエータのディスカバリに認証を使用することをお勧めします。

iSCSI LIOターゲットに対して認証の初期設定を行うには

1. YaSTを起動し、ネットワークサービス > iSCSI LIOターゲットの順に起動します。
2. グローバルタブに切り替えます。

The screenshot shows the 'iSCSI LIO ターゲットの概要' (iSCSI LIO Target Overview) window with the 'ターゲット(T)' (Target) tab selected. The window has three tabs: 'サービス(S)' (Service), 'グローバル(G)' (Global), and 'ターゲット(T)' (Target). The 'ターゲット(T)' tab contains the following options and fields:

- ☒ 検出認証なし(N) (No authentication)
- ☐ ターゲットによる認証(B) (Authentication by target)
 - ユーザ ID (U): [Text field]
 - パスワード(P): [Text field]
- ☐ イニシエータによる認証(O) (Authentication by initiator)
 - ユーザ ID (I): [Text field]
 - パスワード(A): [Text field]

At the bottom of the window, there are four buttons: 'ヘルプ(H)' (Help), '中止(R)' (Cancel), '次へ(N)' (Next), and '完了(F)' (Finish).

3. デフォルトでは、認証は無効(検出認証なし)です。認証を有効にするには、イニシエータによる認証または送信認証、あるいはその両方を選択します。
4. 選択した認証方法に対して資格情報を提供します。ユーザ名とパスワードの組み合わせは、incomingとoutgoingディスカバリで異なっている必要があります。
5. 完了をクリックして、設定を保存して適用します。

14.2.3 ストレージスペースの準備

LUNをiSCSIターゲットサーバ用に設定する前に、使用するストレージを準備する必要があります。未フォーマットのブロックデバイス全体を1つのLUNとして使用することも、デバイスを複数の未フォーマットパーティションに再分割して、各パーティションを別個のLUNとして使用することもできます。iSCSIターゲット設定では、LUNをiSCSIイニシエータにエクスポートします。

YaSTのパーティショナまたはコマンドラインを使用して、パーティションを設定できます。詳細については、『導入ガイド』、第10章「Expert Partitioner (エキスパートパーティショナ)」、10.1項「熟練者向けパーティション設定の使用」を参照してください。iSCSI LIO ターゲットは、Linux、Linux LVM、またはLinux RAIDファイルシステムIDで未フォーマットのパーティションを使用できます。

！ 重要: iSCSIターゲットデバイスをマウントしない

iSCSIターゲットとして使用するデバイスやパーティションを設定したら、ローカルパス経由で直接アクセスしないでください。ターゲットサーバにパーティションをマウントしないでください。

14.2.3.1 仮想環境でのデバイスのパーティション分割

仮想マシンのゲストサーバを、iSCSI LIOターゲットサーバとして使用できます。本項では、Xen仮想マシンにパーティションを割り当てる方法を説明します。また、SUSE Linux Enterprise Serverでサポートされている他の仮想環境も使用できます。

Xen仮想環境で、iSCSI LIOターゲットデバイスに使用するストレージスペースをゲストの仮想マシンに割り当て、ゲスト環境内の仮想ディスクとしてそのスペースにアクセスします。各仮想ディスクは、ディスク全体、パーティション、ボリュームなどの物理ブロックデバイスでも、Xenホストサーバ上の大規模な物理ディスク上の単一イメージファイルが仮想ディスクになっている、ファイルバックディスクイメージのいずれでも可能です。最適なパフォーマンスを得るためには、物理ディスクまたはパーティションから各仮想ディスクを作成してください。ゲストの仮想マシンに仮想ディスクを設定したら、ゲストサーバを起動し、物理サーバの場合と同じ方法で、新しいブランクの仮想ディスクをiSCSIターゲットデバイスとして設定します。

ファイルバックディスクイメージがXenホストサーバ上に作成され、Xenゲストサーバに割り当てられます。デフォルトでは、Xenはファイルバックディスクイメージを `/var/lib/xen/images/VM_NAME` ディレクトリに保存します。ここで `VM_NAME` は仮想マシンの名前です。

14.2.4 iSCSI LIOターゲットグループの設定

YaSTを使用して、iSCSI LIOターゲットデバイスを設定することができます。YaSTでは、**lio-utils** ソフトウェアにより提供されるAPIを使用します。iSCSI LIO ターゲットは、Linux、Linux LVM、またはLinux RAIDファイルシステムIDで未フォーマットのパーティションを使用できます。

！ 重要: パーティション

始める前に、[14.2.3項「ストレージスペースの準備」](#)に記載しているように、iSCSI LIO ターゲットとして使用する未フォーマットのパーティションを作成してください。

1. YaSTを起動し、ネットワークサービス > iSCSI LIOターゲットの順に起動します。

2. ターゲットタブに切り替えます。

iSCSI LIO ターゲットの概要

サービス グローバル **ターゲット**

ターゲット ▼	ポータルグループ	TPGステータス
---------	----------	----------

追加 編集 削除

ヘルプ(H) 中止(R) 次へ(N) 完了(F)

3. 追加をクリックして、新しいiSCSI LIOのターゲットグループとデバイスを定義します。iSCSI LIOターゲットソフトウェアにより、ターゲット、識別子、ポータルグループ、IPアドレス、およびポート番号の各フィールドが自動的に記入されます。認証を使用するが、デフォルトで選択されています。
 - a. 複数のネットワークインターフェースがある場合は、[IPアドレス] ドロップダウンボックスを使用して、このターゲットグループ用に使用するネットワークインターフェースのIPアドレスを選択します。すべてのアドレスでサーバにアクセスできるようにするには、Bind All IP Addresses (すべてのIPアドレスをバインド)を選択します。
 - b. このターゲットグループに対してイニシエータ認証を不要にする場合は、認証を使用をオフにします(非推奨)。
 - c. 追加をクリックします。デバイスまたはパーティションのパスを入力するか、または参照を使用して追加します。オプションで名前を指定して、OKをクリックします。0から始まるLUN番号が自動的に作成されます。フィールドを空にしておくと、名前が自動的に生成されます。
 - d. (オプション)前の手順を繰り返し、このターゲットグループにターゲットを追加します。

- e. 目的のターゲットがすべてグループに追加されたら、次へをクリックします。
4. iSCSIターゲットイニシエータのセットアップの変更ページで、ターゲットグループ内のLUNへのアクセスを許可されるイニシエータに関する情報を設定します。

iSCSIターゲットイニシエータのセットアップの変更

ターゲット	ID	ポータル
2016-08.com.example	b1-a97e-abaebab1fa0	1

イニシエータ ▼	LUN マッピング	認証

追加 LUN の編集 認証の編集 削除 コピー

ヘルプ(H) 中止(R) 戻る(B) 次へ(N)

ターゲットグループに対して少なくとも1つ以上のイニシエータを指定すると、LUNの編集、認証の編集、削除、およびコピーの各ボタンが有効になります。追加またはコピーを使用して、ターゲットグループにイニシエータを追加できます。

[iSCSIターゲットの変更] : オプション

- **追加:** 選択したiSCSI LIOターゲットグループに、新たなイニシエータのエントリを追加します。
- **LUNを編集:** iSCSI LIOターゲットグループ内のどのLUNが、選択したイニシエータにマップするかを設定します。割り当てられたターゲットのそれぞれを、任意のイニシエータにマップすることができます。
- **認証を編集:** 選択したイニシエータに対する好みの認証方法を設定します。認証なしを指定することも、incoming認証、outgoing認証、またはその両方を設定することもできます。

- **削除:** 選択したイニシエータのエントリを、ターゲットグループに割り当てられたイニシエータのリストから削除します。
- **コピー:** 同じLUNのマッピングと認証設定を持つ新たなイニシエータのエントリを、選択したイニシエータのエントリとして追加します。これにより、容易に同じ共有LUNを、クラスタ内の各ノードに順々に割り当てることができます。

- a. 追加をクリックして、イニシエータ名を指定し、TPGからLUNをインポートチェックボックスをオンまたはオフにしてから、OKをクリックして設定を保存します。
- b. イニシエータのエントリを選択して、LUNの編集をクリックし、LUNのマッピングを変更してiSCSI LIOターゲットグループ内のどのLUNを選択したイニシエータに割り当ててかを指定して、OKをクリックして変更内容を保存します。
iSCSI LIOターゲットグループが複数のLUNで構成されている場合は、1つまたは複数のLUNを、選択したイニシエータに割り当てることができます。デフォルトでは、グループ内の使用可能なLUNのそれぞれが、イニシエータLUNに割り当てられます。

LUNの割り当てを変更するには、次の操作の1つ以上を実行します。

- **追加:** 追加をクリックして新しいイニシエータのLUNのエントリを作成し、変更ドロップダウンボックスを使用して、そのエントリにターゲットLUNをマップします。
- **削除:** イニシエータのLUNのエントリを選択し、削除をクリックしてターゲットLUNのマッピングを削除します。
- **変更:** イニシエータのLUNのエントリを選択し、変更ドロップダウンボックスを使用して、そのエントリにマップするターゲットLUNを選択します。

一般的な割り当のプランには、次のようなものがあります。

- 1台のサーバが、イニシエータとして登録されています。ターゲットグループ内のLUNがすべて、それに割り当てられています。
このグループ化戦略を使用して、特定のサーバに対して、iSCSI SANストレージを論理的にグループ化することができます。
- 複数の独立したサーバが、イニシエータとして登録されています。1つまたは複数のターゲットLUNが、それぞれのサーバに割り当てられています。それぞれのLUNは、1台のサーバのみに割り当てられています。

このグループ化戦略を使用して、データセンター内の特定の部門またはサービスのカテゴリに対して、iSCSI SANストレージを論理的にグループ化することができます。

- クラスタの各ノードが、イニシエータとして登録されています。共有のターゲットLUNがすべて、各ノードに割り当てられています。すべてのノードがデバイスに接続されていますが、ほとんどのファイルシステムに対して、クラスタソフトウェアによってデバイスによるアクセスがロックされ、一度に1つのノード上にものみデバイスがマウントされます。共有ファイルシステム(OCFS2など)では、複数のノードが同時に同じファイル構造をマウントし、読み込みおよび書き込みアクセスを持つ同じファイルを開くことが可能です。

このグループ化戦略を使用して、特定のサーバクラスタに対して、iSCSI SANストレージを論理的にグループ化することができます。

- c. イニシエータのエントリを選択して、認証の編集をクリックし、イニシエータに対する認証設定を指定してから、OKをクリックして設定を保存します。

検出認証なしとすることも、イニシエータによる認証、送信認証、またはその両方を設定することもできます。各イニシエータに対して指定できるユーザ名とパスワードの組み合わせは、1つだけです。イニシエータに対するincoming認証とoutgoing認証の資格情報は、異なっても構いません。資格情報は、イニシエータごとに異なっても構いません。

- d. このターゲットグループにアクセスできる各iSCSIイニシエータについて、前の手順を繰り返します。

- e. イニシエータの割り当てを設定し終わったら、次へをクリックします。

5. 完了をクリックして、設定を保存して適用します。

14.2.5 iSCSI LIOターゲットグループの変更

以下のようにして、iSCSI LIOターゲットグループに変更を加えることができます。

- ターゲットLUNデバイスをターゲットグループに追加または削除する
- ターゲットグループに対してイニシエータを追加または削除する

- ターゲットグループのイニシエータに対する、イニシエータLUNからターゲットLUNへのマッピングを変更する
- イニシエータ認証(incoming、outgoing、またはその両方)用のユーザ名とパスワードの資格情報を変更する

iSCSI LIOターゲットグループに対する設定を確認または変更するには:

1. YaSTを起動し、ネットワークサービス > iSCSI LIOターゲットの順に起動します。
2. ターゲットタブに切り替えます。
3. 変更するiSCSI LIOターゲットグループを選択して、編集をクリックします。
4. [iSCSIターゲットLUNのセットアップを変更] ページで、ターゲットグループにLUNを追加し、LUNの割り当てを編集するか、またはターゲットLUNをグループから削除します。すべてグループに目的の変更が行われたら、次へをクリックします。
オプション情報については、[\[iSCSIターゲットの変更\] : オプション](#)を参照してください。
5. [iSCSIターゲットイニシエータのセットアップの変更] ページで、ターゲットグループ内のLUNへのアクセスを許可されるイニシエータに関する情報を設定します。すべてグループに目的の変更が行われたら、次へをクリックします。
6. 完了をクリックして、設定を保存して適用します。

14.2.6 iSCSI LIOターゲットグループの削除

iSCSI LIOターゲットグループを削除すると、グループの定義と、イニシエータに対する関連のセットアップ(LUNのマッピングや認証資格情報を含む)が削除されます。パーティション上のデータは破棄されません。イニシエータに再度アクセス権を付与するには、ターゲットLUNを別のターゲットグループまたは新規のターゲットグループに割り当てて、それらに対するイニシエータアクセスを設定します。

1. YaSTを起動し、ネットワークサービス > iSCSI LIOターゲットの順に起動します。
2. ターゲットタブに切り替えます。
3. 削除するiSCSI LIOターゲットグループを選択して、削除をクリックします。
4. 確認のメッセージが表示されたら、続行をクリックして削除を確認するか、キャンセルをクリックしてキャンセルします。

5. 完了をクリックして、設定を保存して適用します。

14.3 iSCSIイニシエータの設定

iSCSIイニシエータを使用して、任意のiSCSIターゲットに接続できます。これは、[14.2項「iSCSI LIOターゲットサーバのセットアップ」](#)で説明されているターゲットソリューションだけに限りません。iSCSIイニシエータの設定には、利用可能なiSCSIターゲットの検出と、iSCSIセッションの設定という2つの主要ステップがあります。どちらの設定も、YaSTを使って行うことができます。

14.3.1 YaSTを使ったiSCSIイニシエータの設定

YaSTの「iSCSIイニシエータの概要」が3つのタブに分割されます。

サービス:

サービスタブでは、ブート時にiSCSIイニシエータを有効にできます。固有のイニシエータ名とディスカバリに使用するiSNSサーバも設定できます。

接続したターゲット:

Connected Targetsタブには、現在接続しているiSCSIターゲットの概要が表示されます。このタブにも、検出されたターゲットタブのように、システムに新しいターゲットを追加するオプションが用意されています。

検出されたターゲット:

検出されたターゲットタブでは、ネットワーク内のiSCSIターゲットを手動で検出することができます。

14.3.1.1 iSCSIイニシエータの設定

1. YaSTを起動し、ネットワークサービス > iSCSIイニシエータ の順に起動します。
2. サービスタブに切り替えます。

3. サービスを開始で、iSCSIイニシエータサービスの開始方法を指定します。

- **ブート時:** サービスは、サーバの再起動時に自動的に開始します。
- **手動:** (デフォルト)サーバの再起動後、`sudo systemctl start iscsi iscsid` コマンドを実行して、手動でサービスを開始する必要があります。

4. イニシエータ名を指定、または確認します。

このサーバ上のiSCSIイニシエータに、正しい形式のiSCSI修飾名(IQN)を指定します。イニシエータ名はネットワーク全体で固有のものでなければなりません。IQNは次の一般的なフォーマットを使用します。

```
iqn.yyyy-mm.com.mycompany:n1:n2
```

ここでn1とn2はアルファベットか数字です。次に例を示します。

```
iqn.1996-04.de.suse:01:a5dfcea717a
```

イニシエータ名には、サーバ上の `/etc/iscsi/initiatorname.iscsi` ファイルから対応する値が自動的に入力されます。

サーバがiBFT(iSCSI Boot Firmware Table)をサポートしている場合は、イニシエータ名にはiBFT内の対応する値が入力され、このインタフェースではイニシエータ名を変更できません。代わりにBIOSセットアップを使用して変更してください。iBFTは、サーバのiSCSIターゲットとイニシエータの説明を含む、iSCSIの起動プロセスに便利な各種パラメータを含んだ情報ブロックです。

5. 次のいずれかの方法を使用して、ネットワーク上のiSCSIターゲットを検出します。

- **iSNS:** iSNS (Internet Storage Name Service)を使用してiSCSIターゲットを検出するには、続いて14.3.1.2項「iSNSによるiSCSIターゲットの検出」を実行します。
- **検出されたターゲット:** iSCSIターゲットデバイスを手動で検出するには、続いて14.3.1.3項「iSCSIターゲットの手動検出」を実行します。

14.3.1.2 iSNSによるiSCSIターゲットの検出

このオプションを使用する前に、ご使用の環境内でiSNSサーバをインストールし、設定しておく必要があります。詳細については、第13章「Linux用iSNS」を参照してください。

1. YaSTでiSCSIイニシエータを選択し、次にサービスタブを選択します。
2. iSNSサーバのIPアドレスとポートを指定します。デフォルトのポートは3205です。
3. OKをクリックして、変更内容を保存して適用します。

14.3.1.3 iSCSIターゲットの手動検出

iSCSIイニシエータを設定しているサーバからアクセスする各iSCSIターゲットサーバについて、次の手順を繰り返し実行します。

1. YaSTでiSCSIイニシエータを選択し、次に検出されたターゲットタブを選択します。
2. 検出をクリックして「iSCSIイニシエータの検出」ダイアログを開きます。
3. IPアドレスを入力し、必要に応じてポートを変更します。デフォルトポートは3260です。
4. 認証が必要な場合は、検出認証なしをオフにして、イニシエータによる認証またはターゲットによる認証で資格情報を指定します。
5. 次へをクリックして、検出を開始し、iSCSIターゲットサーバに接続します。
6. 資格情報が必要な場合は、検出成功後、接続を使用してターゲットを有効化します。

指定したiSCSIターゲットを使用するための、認証資格情報の提供を促されます。

7. 次へをクリックして、設定を完了します
これでターゲットが接続したターゲットに表示され、仮想iSCSIデバイスが使用可能になります。
8. OKをクリックして、変更内容を保存して適用します。
9. **lsscsi** コマンドを使用すると、iSCSIターゲットデバイスのローカルデバイスパスを検出することができます。

14.3.1.4 iSCSIターゲットデバイスの起動設定

1. YaSTで、iSCSIイニシエータを選択し、次に接続したターゲットタブを選択して、現在サーバに接続されているiSCSIターゲットデバイスの一覧を表示することができます。
2. 管理するiSCSIターゲットデバイスを選択します。
3. 起動の切り替えをクリックして設定を変更します。

自動: このオプションは、iSCSIサービス自体の起動時に接続するiSCSIターゲットに使用されます。これが通常の設定です。

Onboot(起動時): このオプションは、起動時、つまりルート(/)がiSCSI上にある場合に接続するiSCSIターゲットに使用します。したがって、iSCSIターゲットデバイスはサーバの起動時にinitrdによって評価されます。このオプションはIBM Zなど、iSCSIからブートできないプラットフォームでは無視されます。したがって、これらのプラットフォームでは使用しないでください。代わりに自動を使用してください。

4. OKをクリックして、変更内容を保存して適用します。

14.3.2 手動によるiSCSIイニシエータの設定

iSCSI接続の検出や設定を行うには、iscsidが稼働していなければなりません。初めてディスクバリを実行する場合、iSCSIイニシエータの内部データベースが /etc/iscsi/ ディレクトリに作成されます。

ディスクバリがパスワードにより保護されている場合は、iscsidに認証情報を渡します。最初にディスクバリを実行するときには内部データベースが存在していないため、現時点では使用できません。かわりに、/etc/iscsid.conf 設定ファイルを編集して、情報を指定する必要があります。パスワード情報をiscsidに渡すには、/etc/iscsid.conf ファイルの最後に、次の行を追加します。

```
discovery.sendtargets.auth.authmethod = CHAP
discovery.sendtargets.auth.username = USERNAME
discovery.sendtargets.auth.password = PASSWORD
```

ディスカバリは、受け取ったすべての値を内部データベースに保存します。また、検出したターゲットをすべて表示します。次のコマンドで、このディスカバリを実行します。

```
tux > sudo iscsiadm -m discovery --type=st --portal=TARGET_IP
```

次のように出力されます。

```
10.44.171.99:3260,1 iqn.2006-02.com.example.iserv:systems
```

iSNS サーバで利用できるターゲットを検出するには、次のコマンドを使用します。

```
sudo iscsiadm --mode discovery --type isns --portal TARGET_IP
```

iSCSIターゲットに定義されている各ターゲットが、それぞれ1行に表示されます。保存されたデータの詳細については、[14.3.3項「iSCSIイニシエータデータベース」](#)を参照してください。

iscsiadm コマンドの `--login` オプションを使用すると、必要なすべてのデバイスが作成されます。

```
tux > sudo iscsiadm -m node -n iqn.2006-02.com.example.iserv:systems --login
```

新しく生成されたデバイスは **lsscsi** コマンドの出力に表示され、マウントできるようになります。

14.3.3 iSCSIイニシエータデータベース

iSCSIイニシエータにより検出されたあらゆる情報は、`/etc/iscsi` に存在する2つのデータベースファイルに保存されます。1つは、ディスカバリが検出したターゲット用のデータベースで、もう1つは検出したノード用のデータベースです。データベースにアクセスする場合、まずデータをディスカバリ用データベースから取得するのか、またはノードデータベースから取得するのかを指定する必要があります。指定するには、**iscsiadm** コマンドの `-m discovery` または `-m node` パラメータを使用します。**iscsiadm** コマンドに、どちらかのパラメータを指定して実行すると、そのデータベースに保管されているレコードの概要が表示されます。

```
tux > sudo iscsiadm -m discovery
10.44.171.99:3260,1 iqn.2006-02.com.example.iserv:systems
```

この例のターゲット名は `iqn.2006-02.com.example.iserv:systems` です。このデータセットに関連する操作を行う場合に、この名前が必要になります。ID `iqn.2006-02.com.example.iserv:systems` のデータレコードのコンテンツを調べるには、次のコマンドを使用します。

```
tux > sudo iscsiadm -m node --targetname iqn.2006-02.com.example.iserv:systems
node.name = iqn.2006-02.com.example.iserv:systems
node.transport_name = tcp
node.tpgt = 1
node.active_conn = 1
node.startup = manual
node.session.initial_cmdsn = 0
node.session.reopen_max = 32
node.session.auth.authmethod = CHAP
node.session.auth.username = joe
node.session.auth.password = *****
node.session.auth.username_in = EMPTY
node.session.auth.password_in = EMPTY
node.session.timeo.replacement_timeout = 0
node.session.err_timeo.abort_timeout = 10
node.session.err_timeo.reset_timeout = 30
node.session.iscsi.InitialR2T = No
node.session.iscsi.ImmediateData = Yes
....
```

これらの変数の値を変更する場合は、**iscsiadm** コマンドで `update` オプションを使用します。たとえば、初期化時に `iscid` を iSCSI ターゲットにログインさせる場合は、値に `automatic` と `node.startup` を設定します。

```
sudo iscsiadm -m node -n iqn.2006-02.com.example.iserv:systems \
-p ip:port --op=update --name=node.startup --value=automatic
```

`delete` で、古くなったデータセットを削除します。ターゲット `iqn.2006-02.com.example.iserv:systems` が有効なレコードでなくなった場合は、次のコマンドでこのレコードを削除してください。

```
tux > sudo iscsiadm -m node -n iqn.2006-02.com.example.iserv:systems \
-p ip:port --op=delete
```

❗ 重要: 確認は表示されない

このオプションでは、確認のメッセージを表示せずにレコードを削除するため、使用の際には細心の注意を払うようにしてください。

検出したすべてのターゲットのリストを取得するには、**sudo iscsiadm -m node** コマンドを実行します。

14.4 インストール時のiSCSIディスクの使用

iSCSI対応のファームウェアを使用している場合は、AMD64/Intel 64およびIBM POWERの各アーキテクチャ上のiSCSIディスクからのブートがサポートされています。

インストール時にiSCSIディスクを使用するには、次のパラメータをブートパラメータ行に追加する必要があります。

```
withiscsi=1
```

インストール中に、インストールプロセスで使用するiSCSIディスクをシステムに接続するオプションが記載された、追加の画面が表示されます。



注記: マウントポイントのサポート

iSCSIデバイスはブートプロセス中は非同期で表示されます。これらのデバイスがルートファイルシステム用に正しく設定されていることがinitrdによって保証されるまでの間、他のファイルシステムや `/usr` などのマウントポイントでは、これは保証されません。したがって、`/usr` や `/var` などのシステムマウントポイントはサポートされません。これらのデバイスを使用するには、必ず各サービスとデバイスを正しく同期してください。

14.5 iSCSIのトラブルシューティング

本項では、iSCSIターゲットとiSCSIイニシエータに関するいくつかの既知の問題と、考えられる解決策について説明します。

14.5.1 iSCSI LIOターゲットサーバにターゲットLUNをセットアップする際のポータルエラー

iSCSI LIOターゲットグループの追加または編集を行う際に、次のエラーが発生する:

```
Problem setting network portal IP_ADDRESS:3260
```

`/var/log/YasT2/y2log` ログファイルに、次のエラーが含まれている:

```
find: `/sys/kernel/config/target/iscsi': No such file or directory
```

この問題は、iSCSI LIOターゲットサーバソフトウェアがその時点で実行中ではない場合に発生します。この問題を解決するには、YaSTを終了して、コマンドラインで手動で **`systemctl start target`** を実行してiSCSI LIOを起動し、再試行します。

次のように入力して、**configfs**、**iscsi_target_mod**、および**target_core_mod**がロードされているかどうかチェックすることもできます。サンプルの応答を示しています。

```
tux > sudo lsmod | grep iscsi
iscsi_target_mod      295015  0
target_core_mod       346745  4
iscsi_target_mod,target_core_pscsi,target_core_iblock,target_core_file
configfs              35817  3 iscsi_target_mod,target_core_mod
scsi_mod              231620  16
iscsi_target_mod,target_core_pscsi,target_core_mod,sg,sr_mod,mptctl,sd_mod,
scsi_dh_rdac,scsi_dh_emc,scsi_dh_alua,scsi_dh_hp_sw,scsi_dh,libata,mptspi,
mptscsih,scsi_transport_spi
```

14.5.2 iSCSI LIOターゲットが他のコンピュータで表示されない

ターゲットサーバでファイアウォールを使用している場合は、他のコンピュータでiSCSI LIOターゲットを表示できるようにするために使用するiSCSIポートを開く必要があります。詳細については、[14.2.1項「iSCSI LIOターゲットサービスの起動およびファイアウォールの設定」](#)を参照してください。

14.5.3 iSCSIトラフィックのデータパッケージがドロップされる

ファイアウォールは、過剰にビジーになるとパケットをドロップすることがあります。SUSEファイアウォールのデフォルトは、3分後にパケットをドロップすることです。iSCSIトラフィックのパケットがドロップされていることが分かった場合は、ファイアウォールがビジーになったとき、パケットをドロップする代わりにキューに入れるように、SUSEファイアウォールを設定することを検討してください。

14.5.4 LVMでiSCSIボリュームを使用する

iSCSIターゲットでLVMを使用する際には、本項のトラブルシューティングのヒントを使用してください。

14.5.4.1 ブート時にiSCSIイニシエータの検出が行われるかどうかを確認する

iSCSIイニシエータをセットアップする際には、udevがブート時にiSCSIデバイスを検出し、LVMによるそれらのデバイスの使用をセットアップできるように、ブート時の検出を有効にしてください。

14.5.4.2 iSCSIターゲットの検出がブート時に起きることを確認する

udevは、デバイスのデフォルトセットアップを提供することを思い出してください。デバイスを作成するすべてのアプリケーションがブート時に起動されることを確認してください。これにより、udevがシステム起動時にそれらを認識し、デバイスを割り当てることができます。アプリケーションまたはサービスが後まで起動しない場合は、udevがブート時のように自動的にデバイスを作成することはありません。

14.5.5 設定ファイルが手動に設定されていると、iSCSIターゲットがマウントされる

Open-iSCSIは、`/etc/iscsi/iscsid.conf` ファイルで `node.startup` オプションが手動に設定されている場合でも、設定ファイルを手動で変更すれば、起動時にターゲットをマウントできます。

`/etc/iscsi/nodes/TARGET_NAME/IP_ADDRESS,PORT/default` ファイルを確認してください。このファイルには、`/etc/iscsi/iscsid.conf` ファイルを上書きする `node.startup` 設定が含まれています。YaSTインタフェースを使用してマウントオプションを手動に設定すると、`/etc/iscsi/nodes/TARGET_NAME/IP_ADDRESS,PORT/default` ファイルでも `node.startup = manual` が設定されます。

14.6 iSCSI LIOターゲットの用語

backstore

iSCSIのエンドポイントの基礎となる実際のストレージを提供する、物理的ストレージオブジェクト。

CDB (command descriptor block))

SCSIコマンドの標準フォーマット CDBは一般的に6、10、または12バイトの長さですが、16バイトまたは可変長でも構いません。

CHAP (Challenge Handshake Authentication Protocol)

ポイントツーポイントプロトコル(PPP)の認証方法で、あるコンピュータのアイデンティティを別のコンピュータに対して確認するために使用します。Link Control Protocol (LCP)によって2台のコンピュータが接続され、CHAPメソッドがネゴシエートされた後、認証者はランダムなチャレンジをピアに送信します。ピアは、チャレンジおよび秘密鍵に依存した、暗号的にハッシュされたレスポンスを発行します。認証者は、ハッシュされたレスポンスを、予想されるハッシュ値の自身の計算に対して検証し、認証を了承するか、接続を終了します。CHAPは、RFC 1994で定義されています。

CID (接続識別子)

イニシエータが生成する16ビットの番号で、2つのiSCSIデバイス間の接続を、一意に識別するもの。この番号は、ログインフェーズの間に提示されます。

エンドポイント

iSCSIターゲット名とiSCSI TPG (IQN + Tag)の組み合わせ

EUI (extended unique identifier)

世界中のあらゆるデバイスを一意に識別する、64ビットの番号。フォーマットは、会社ごとに一意である24ビットと、その会社が自社の各デバイスに割り当てる40ビットで構成されます。

イニシエータ

SCSIセッションの開始エンド。通常は、コンピュータなどの制御デバイス。

IPS (Internet Protocol storage)

IPプロトコルを使用してストレージネットワーク内のデータを移動する、プロトコルまたはデバイスのクラス。FCIP (Fibre Channel over Internet Protocol)、iFCP (Internet Fibre Channel Protocol)、およびiSCSI (Internet SCSI)は、すべてIPSプロトコルの例です。

IQN (iSCSI qualified name)

世界中のあらゆるデバイスを一意に識別する、iSCSIの名前形式(たとえば:
iqn.5886.com.acme.tapedrive.sn-a12345678)。

ISID (initiator session identifier)

イニシエータが生成する48ビットの番号で、イニシエータとターゲット間のセッションを一意に識別するもの。この値はログインプロセスの間に作成され、ログインPDUとともにターゲットに送られます。

MCS (multiple connections per session)

iSCSI仕様の一部で、イニシエータとターゲット間での複数のTCP/IP接続を可能にするもの。

MPIO (multipath I/O)

サーバとストレージ間でデータが複数の冗長パスをとることができるメソッド。

ネットワークポータル

iSCSIエンドポイントおよびIPアドレスとTCP (転送制御プロトコル)ポートの組み合わせ。TCPポート3260が、iSCSIプロトコル用のポート番号です。これは、IANA (Internet Assigned Numbers Authority)により定義されています。

SAM (SCSI architectural model)

SCSIの動作を一般的な表記で記載した文書で、異なる種類のデバイスがさまざまなメディア上で通信することを可能にするもの。

ターゲット

SCSIセッションの受信側で、通常はディスクドライブ、テープドライブ、スキャナなどのデバイス。

ターゲットグループ(TG)

ビューの作成時にすべて同じ扱いを受ける、SCSIターゲットポートのリスト。ビューを作成することで、LUN(論理ユニット番号)のマッピングが簡素化されます。それぞれのビューエントリが、ターゲットグループ、ホストグループ、およびLUNを指定します。

ターゲットポート

iSCSIエンドポイントと、1つ以上のLUNの組み合わせ。

ターゲットポートグループ(TPG)

IPアドレスとTCPポート番号のリストで、特定のiSCSIターゲットがどのインタフェースから受信するかを決定するもの。

ターゲットセッション識別子(TSID)

ターゲットが生成する 16ビットの番号で、イニシエータとターゲット間のセッションを一意に識別するもの。この値はログインプロセスの間に作成され、ログインレスポンス PDU(プロトコルデータユニット)とともにイニシエータに送られます。

14.7 追加情報

iSCSIプロトコルは、数年に渡って利用されています。iSCSIとSANソリューションの比較やパフォーマンスのベンチマークは多くのレビューで取り上げられており、ハードウェアソリューションについて説明したドキュメントもあります。詳細については、<http://www.open-iscsi.com/>にあるOpen-iSCSIプロジェクトのホームページを参照してください。

また、[iscsiadm](#)、[iscsid](#)、[ietd.conf](#)、および [ietd](#) の各マニュアルページのほか、環境設定ファイルのサンプル [/etc/iscsid.conf](#) も参照してください。

15 Fibre Channel Storage over Ethernet Networks: FCoE

多くの企業のデータセンターが、そのLANおよびデータトラフィックをEthernetに依存し、またそのストレージインフラストラクチャをファイバチャネルに依存しています。Open Fibre Channel over Ethernet (FCoE)イニシエータソフトウェアは、Ethernetアダプタが付いたサーバが、Ethernetネットワーク上でファイバチャネルストレージに接続できるようにします。このコネクティビティはこれまで、ファイバチャネルファブリック上にファイバチャネルアダプタを有するシステム用に、独占的に確保されていました。FCoEテクノロジーは、ネットワークコンバージェンスを支援することで、データセンター内の複雑性を減らします。これにより、ファイバチャネルストレージへの既存の投資を無駄にすることなく、ネットワーク管理を簡素化することができます。

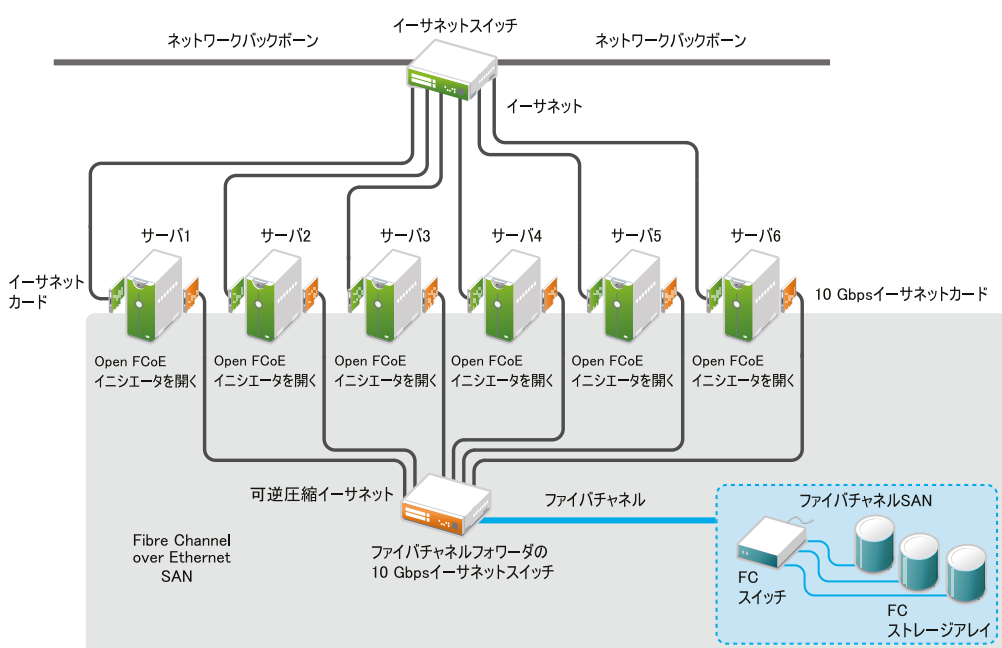


図 15.1: OPEN FIBRE CHANNEL OVER ETHERNET SAN

Open-FCoEでは、ホストバスアダプタ上の専有のハードウェアではなく、ホストでファイバチャネルのを実行することができます。対象としているのは10 Gbps (ギガバイト/秒)のEthernetアダプタですが、PAUSEフレームに対応したすべてのEthernetアダプタで使用可能です。イニシエータソフトウェアにより、ファイバチャネルプロトコルの処理モジュールと、Ethernetベースのトランスポートモジュールが提供されます。Open-FCoEモジュールは、SCSI用の低レベルドライバの役割を果たします。Open-FCoEトランスポートは、net_deviceを使用してパケットの送受信を行います。DCB(データセンターブリッジング)ドライバにより、FCoE向けのサービスの質が提供されます。

FCoEは、ファイバチャネルフレームを変えずに、ファイバチャネルのプロトコルトラフィックをEthernet接続上で動かす、カプセル化プロトコルです。これにより、ネットワークセキュリティとトラフィック管理インフラストラクチャが、ファイバチャネルにおけるのと同じようにFCoEでも機能することができます。

以下の条件が当てはまる企業では、FCoEの導入を選択してもよいでしょう。

- すでにファイバチャネルストレージシステムがあり、ファイバチャネルのスキルと知識を持つ管理者がいる。
- ネットワーク内に、10 GbpsのEthernetを展開している。

本項では、ネットワークにFCoEを設定する方法を説明します。

15.1 インストール時におけるFCoEインタフェースの設定

SUSE Linux Enterprise Server向けのYaSTのインストールでは、サーバとファイバチャネルストレージインフラストラクチャ間の接続用のスイッチでFCoEが有効になっていれば、オペレーティングシステムのインストール時にFCoEディスクの設定を行うことができます。一部のシステムBIOSタイプでは、FCoEディスクを自動的に検出することができ、そのディスクをYaSTのインストールソフトウェアに報告します。ただし、FCoEディスクの自動検出は、すべてのBIOSのタイプでサポートされているわけではありません。その場合、インストールの開始時に次の`withfcoe` オプションをカーネルのコマンドラインに追加することで、自動検出を有効にすることができます。

```
withfcoe=1
```

FCoEディスクが検出されると、YaSTのインストールでは、同時にFCoEを設定するオプションがあります。[ディスクアクティベーション] ページで、FCoEインタフェースの設定を選択して、FCoEの設定にアクセスします。FCoEインタフェースの設定については、[15.3項「YaSTを使用したFCoEサービスの管理」](#)を参照してください。



注記: マウントポイントのサポート

FCoEデバイスはブートプロセス中は非同期で表示されます。これらのデバイスがルートファイルシステム用に正しく設定されていることがinitrdによって保証されるまでの間、他のファイルシステムや `/usr` などのマウントポイントでは、これは保証されません。したがって、`/usr` や `/var` などのシステムマウントポイントはサポートされません。これらのデバイスを使用するには、各サービスとデバイスが正しく同期されていることを確認します。

15.2 FCoEおよびYaSTのFCoEクライアントのインストール

サーバへの接続用のスイッチでFCoEを有効にすることで、ストレージインフラストラクチャ内にFCoEディスクを設定することができます。SUSE Linux Enterprise Serverオペレーティングシステムのインストール時にFCoEディスクが利用可能であれば、FCoEイニシエータソフトウェアが、その時点で自動的にインストールされます。

FCoEイニシエータソフトウェアとYaST FCoEクライアントソフトウェアがインストールされていない場合は、次の手順で次のコマンドを使用して手動でインストールします。

```
tux > sudo zypper in yast2-fcoe-client fcoe-utils
```

または、YaSTソフトウェアマネージャを使用して、これらのパッケージをインストールします。

15.3 YaSTを使用したFCoEサービスの管理

YaST FCoEクライアント設定オプションを使用して、お使いのファイバチャネルストレージインフラストラクチャ内のFCoEディスク用のFCoEインタフェースの作成、設定、および削除ができます。このオプションを使用するには、FCoEイニシエータサービス(`fcoemon` デーモン)およびLink Layer Discovery Protocolエージェントデーモン(`lldpad`)がインストールされて実行中であり、FCoE接続が、FCoE対応のスイッチで有効になっている必要があります。

1. YaSTを起動し、ネットワークサービス > FCoEクライアントの設定の順に選択します。



2. サービスタブで、FCoEサービスとLldpad (Link Layer Discovery Protocolエージェントデーモン)サービスの開始時刻を確認し、必要に応じて変更します。

- **FCoEサービスの開始:** Fibre Channel over Ethernetサービスの **fcoemon** デーモンを、サーバの起動時に開始するか、マニュアルで開始するかを指定します。このデーモンは、FCoEインタフェースを制御して、**lldpad** デーモンとの接続を確立します。値は、**起動時**(デフォルト)または**マニュアル**です。
- **Lldpadサービスの開始:** Link Layer Discovery Protocolエージェント **lldpad** デーモンを、サーバの起動時に開始するか、マニュアルで開始するかを指定します。**lldpad** デーモンは、データセンタブリッジング機能およびFCoEインタフェースの設定について、**fcoemon** デーモンに情報を送ります。値は、**起動時**(デフォルト)または**マニュアル**です。

設定を変更した場合は、OKをクリックして変更内容を保存して適用します。

3. インタフェースタブで、サーバ上で検出されたすべてのネットワークアダプタに関する情報(VLANおよびFCoEの設定に関する情報を含む)を確認します。また、FCoE VLANインタフェースの作成や既存のFCoEインタフェース設定の変更、FCoEインタフェースの削除もできます。

Fibre Channel over Ethernetの環境設定

サービス(S)

インタフェース(I)

環境設定(C)

デバイス	MACアドレス	モデル	VLAN	FCoE VLANインタフェース	FCoE有効	DCBを必要とする	自動VLAN	DCB対応	ドライバ	FCoEにフ
eth0	00:0c:29:3a:7f:b9	82545EM Gigabit Ethernet Controller (Copper)		利用不可			いいえ	e1000	未設定	
eth1	00:0c:29:3a:7f:c3	82545EM Gigabit Ethernet Controller (Copper)		利用不可			いいえ	e1000	未設定	
eth2	00:0c:29:3a:7f:cd	82545EM Gigabit Ethernet Controller (Copper)		利用不可			いいえ	e1000	未設定	
eth3	00:0c:29:3a:7f:d7	82545EM Gigabit Ethernet Controller (Copper)		利用不可			いいえ	e1000	未設定	
eth4	00:0c:29:3a:7f:e1	82545EM Gigabit Ethernet Controller (Copper)		利用不可			いいえ	e1000	未設定	
eth5	00:0c:29:3a:7f:eb	82545EM Gigabit Ethernet Controller (Copper)		利用不可			いいえ	e1000	未設定	

検出の再試行(D)

設定の変更(S)

FCoEインタフェースの作成(F)

インタフェースの削除(R)

ヘルプ(H)

キャンセル(C)

OK(O)

FCoE VLANインタフェース列を使用して、FCoEが使用可能かどうかを判断します。

インタフェース名

インタフェースに名前が割り当てられている(**eth4.200** など)場合は、スイッチでFCoEが利用可能であり、FCoEインタフェースがアダプタに対してアクティブになっています。

未設定:

状態が未設定である場合は、スイッチでFCoEが有効になっていますが、FCoEインタフェースはアダプタに対してアクティブになっていません。アダプタでインタフェースを有効にするには、アダプタを選択して、FCoE VLANインタフェースを作成をクリックします。

使用不可:

状態が使用不可である場合は、FCoEがスイッチ上のその接続に対して有効になっていないため、そのアダプタではFCoEは使えません。

4. 未設定のFCoE対応アダプタを設定するには、そのアダプタを選択し、FCoE VLANインタフェースを作成をクリックします。問い合わせに対して、はいを選択して確認します。

アダプタがインタフェース名と共にFCoE VLANインタフェース列に表示されます。

5. 設定済みのアダプタの設定を変更するには、リストでそのアダプタを選択し、Change Settings (設定の変更)をクリックします。

次のオプションを設定できます。

FCoEの有効化

アダプタに対してFCoEインスタンスの作成を有効または無効にします。

DCBが必要

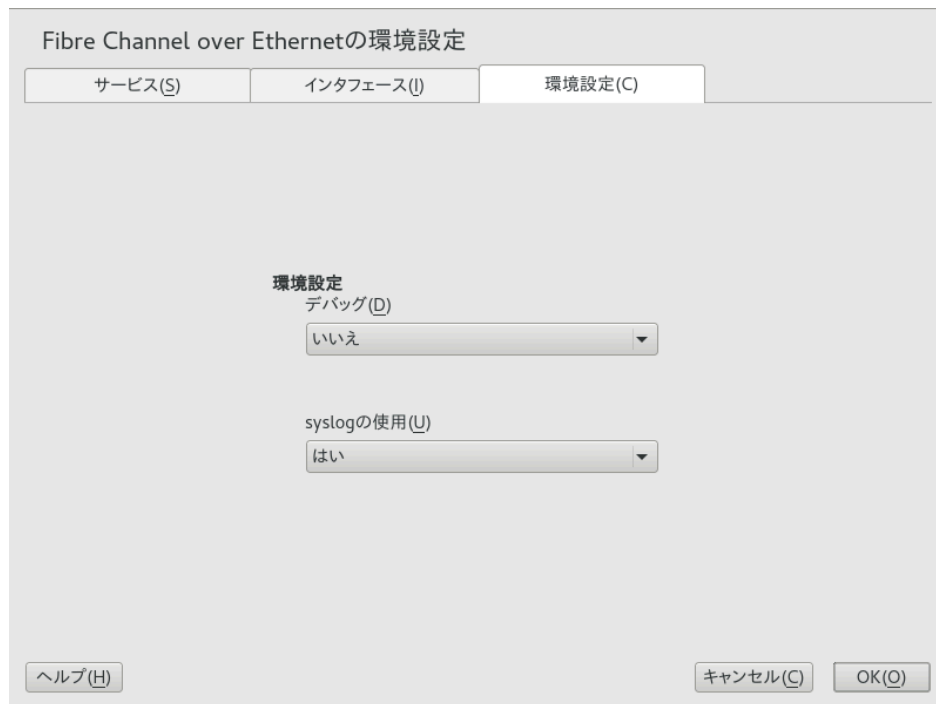
データセンタブリッジングがアダプタに必要なかどうかを指定します(通常は必要です)。

Auto VLAN

fcoemon デーモンでVLANインタフェースを作成するかどうかを指定します。

設定を変更した場合は、次へをクリックして変更内容を保存して適用します。設定は、`/etc/fcoe/cfg-ethX` ファイルに書き込まれます。fcoemon デーモンは、初期化時に各FCoEインタフェースの環境設定ファイルを読み込みます。

6. 設定済みのインタフェースを削除するには、それをリストで選択します。インタフェースの削除をクリックし、続行をクリックして確認します。FCoEインタフェースの値が、未構成に変わります。
7. 設定タブで、FCoEシステムサービスの全般設定を確認または変更します。FCoEサービススクリプトと fcoemon デーモンからのデバッグメッセージを有効/無効にしたり、メッセージをシステムログに送信するかどうかを指定したりできます。



8. OKをクリックして、変更内容を保存して適用します。

15.4 コマンドを使用したFCoEの設定

1. 端末コンソールを開きます。
2. YaSTを使用して、Ethernetネットワークインタフェースカード(eth2 など)を設定します。
3. Link Layer Discovery Protocolエージェントデーモン(llpad)を起動します。

```
tux > sudo systemctl start lldpad
```

4. お使いのEthernetアダプタ上で、データセンタースブリッジングを有効にします。

```
tux > dcbtool sc eth2 dcb on
Version:      2
Command:      Set Config
Feature:      DCB State
Port:         eth2
Status:       Successful
```

5. Priority Flow Control (PFC)設定を、データセンタースブリッジングに対して有効にします。


```
tux > sudo dcbtool sc eth<x> pfc e:1 a:1 w:1
```

引数の設定値は次のとおりです。

e:<0|1>

機能の有効化を制御します。

a:<0|1>

機能を、データセンタースブリッジ交換プロトコルを介してピアにアドバタイズするかどうかを制御します。

w:<0|1>

機能が、ピアから受け取った内容に基づいてその運用設定を積極的に変更するかどうかを制御します。

6. データセンタースブリッジを有効にして、FCoEに対するスイッチの優先度設定を受け入れます。

```
tux > sudo dcbtool sc eth2 app:fcoe e:1
Version:      2
Command:      Set Config
Feature:      Application FCoE
Port:         eth2
Status:       Successful
```

7. デフォルトのFCoE環境設定ファイルを、`/etc/fcoe/cfg-eth2`にコピーします。

```
tux > sudo cp /etc/fcoe/cfg-ethx /etc/fcoe/cfg-eth2
```

8. FCoEイニシエータサービスを開始します。

```
systemctl start fcoe.service
```

9. Link Layer Discovery Protocolエージェントデーモン(`llpad`)およびFCoEイニシエータサービスを、起動時に開始するよう設定します。

```
tux > systemctl enable llpad fcoe
```

15.5 FCoE管理ツールを使用したFCoEインスタンスの管理

fcoeadm ユーティリティは、FCoE (Fibre Channel over Ethernet) 管理ツールです。これを使用して、所定のネットワークインタフェースのFCoEインスタンスの作成、破棄、およびリセットを行うことができます。**fcoeadm** ユーティリティは、ソケットインタフェースを通じて、実行中の **fcoemon** プロセスにコマンドを送ります。**fcoemon** の詳細については、**man 8 fcoemon** を参照してください。

fcoeadm ユーティリティを使用して、以下に関してFCoEインスタンスにクエリを行うことができます。

- インタフェース
- ターゲットLUN
- ポートの統計データ

fcoeadm ユーティリティは、**fcoe-utils** パッケージの一部です。このコマンドの一般的な構文は、次のようになります。

```
fcoeadm
[-c|--create] [<ethX>]
[-d|--destroy] [<ethX>]
[-r|--reset] [<ethX>]
[-S|--Scan] [<ethX>]
[-i|--interface] [<ethX>]
[-t|--target] [<ethX>]
[-l|--lun] [<ethX>]
[-s|--stats <ethX>] [<interval>]
[-v|--version]
[-h|--help]
```

詳細については、**man 8 fcoeadm** を参照してください。

例

fcoeadm -c eth2.101

FCoEインスタンスをeth2.101上に作成します。

fcoeadm -d eth2.101

FCoEインスタンス上のeth2.101を破棄します。

fcoeadm -i eth3

インタフェース eth3 上の FCoE インスタンスすべてに関する情報を表示します。インタフェースが指定されていない場合、FCoE インスタンスが作成されているすべてのインタフェースの情報を表示します。次に、接続 eth0.201 の情報の例を示します。

```
tux > sudo fcoeadm -i eth0.201
Description:      82599EB 10-Gigabit SFI/SFP+ Network Connection
Revision:         01
Manufacturer:     Intel Corporation
Serial Number:    001B219B258C
Driver:           ixgbe 3.3.8-k2
Number of Ports:  1

Symbolic Name:    fcoe v0.1 over eth0.201
OS Device Name:   host8
Node Name:        0x1000001B219B258E
Port Name:        0x2000001B219B258E
FabricName:       0x2001000573D38141
Speed:            10 Gbit
Supported Speed:  10 Gbit
MaxFrameSize:     2112
FC-ID (Port ID):  0x790003
State:            Online
```

fcoeadm -l eth3.101

接続 eth3.101 で検出されたすべての LUN の詳細情報を表示します。接続が指定されていない場合、すべての FCoE 接続で検出されたすべての LUN の情報を表示します。

fcoeadm -r eth2.101

eth2.101 上の FCoE インスタンスをリセットします。

fcoeadm -s eth3 3

FCoE インスタンスが存在する特定の eth3 ポートに関する統計情報を 3 秒間隔で表示します。統計情報は、時間間隔ごとに 1 行ずつ表示されます。間隔を指定していない場合、デフォルトの 1 秒が間隔として使用されます。

fcoeadm -t eth3

FCoE インスタンスが存在する特定の eth3 ポートから検出されたすべてのターゲットに関する情報を表示します。検出された各ターゲットの後ろに、関連付けられた LUN が列記されます。インスタンスが指定されていない場合、FCoE インスタンスが存在するすべてのポートからのターゲットを表示します。次に、接続 eth0.201 からのターゲットの情報の例を示します。

```
tux > sudo fcoeadm -t eth0.201
Interface:        eth0.201
```

```

Roles:          FCP Target
Node Name:      0x200000D0231B5C72
Port Name:      0x210000D0231B5C72
Target ID:      0
MaxFrameSize:   2048
OS Device Name: rport-8:0-7
FC-ID (Port ID): 0x79000C
State:          Online

```

LUN ID	Device Name	Capacity	Block Size	Description
40	/dev/sdqi	792.84 GB	512	IFT DS S24F-R2840-4 (rev 386C)
72	/dev/sdpk	650.00 GB	512	IFT DS S24F-R2840-4 (rev 386C)
168	/dev/sdgy	1.30 TB	512	IFT DS S24F-R2840-4 (rev 386C)

15.6 追加情報

詳細については、以下のマニュアルを参照してください。

- Open-FCoEのサービスデーモンについては、[fcoemon\(8\)](#) マニュアルページを参照してください。
- Open-FCoEの管理ツールについては、[fcoeadm\(8\)](#) マニュアルページを参照してください。
- データセンタブリッジング設定ツールについては、[dcbtool\(8\)](#) マニュアルページを参照してください。
- Link Layer Discovery Protocolエージェントデーモンについては、[lldpad\(8\)](#) マニュアルページを参照してください。
- 一般的な情報は、Open-FCoEホームページ(<http://www.open-fcoe.org/dokuwiki/start>)を参照してください。

16 NVMe over Fabric

この章では、NVMe over Fabricホストおよびターゲットの設定方法について説明します。

16.1 概要

NVM Express (NVMe)は、不揮発性ストレージ(通常はSSDディスク)にアクセスするためのインタフェース規格です。NVMeはSATAをはるかに上回る処理速度をサポートし、レイテンシも低くなります。

NVMe over Fabricは、RDMAやNVMe over Fibre Channel (FC-NVMe)などの異なるネットワークワーキングファブリックを介してNVMeストレージにアクセスするためのアーキテクチャです。NVMe over Fabricの機能はiSCSIと同様です。耐障害性を向上させるため、NVMe over Fabricにはマルチパスのサポートが組み込まれています。NVMe over Fabricマルチパスは、従来のデバイスマッパーマルチパスに基づいていません。

NVMeホストは、NVMeターゲットに接続するマシンです。NVMeターゲットは、そのNVMeブロックデバイスを共有するマシンです。

NVMeは、SUSE Linux Enterprise Server 15でサポートされています。NVMeブロックストレージおよびNVMe over Fabricターゲットとホストには、専用のカーネルモジュールが用意されています。

ご使用のハードウェアに関する特別な考慮事項があるかどうかを確認するには、[16.4項「特定のハードウェアの設定」](#)を参照してください。

16.2 NVMe over Fabricホストの設定

NVMe over Fabricを使用するには、サポートされているネットワーク方法のいずれかでターゲットを使用可能にする必要があります。NVMe over Fibre ChannelおよびRDMAがサポートされています。以降のセクションでは、NVMe over FabricホストをNVMeターゲットに接続する方法について説明します。

16.2.1 コマンドラインクライアントのインストール

NVMe over Fabricを使用するには、[nvme](#) コマンドラインツールが必要です。インストールするには、[zypper](#) を実行します。

```
tux > sudo zypper in nvme-cli
```

すべての使用可能なサブコマンドを一覧にするには、`nvme --help`を使用します。`nvme`サブコマンド用のマニュアルページが提供されています。`man nvme-SUBCOMMAND`を実行すると、このページを参照できます。たとえば、`discover`サブコマンドのマニュアルページを参照するには、`man nvme-discover`を実行します。

16.2.2 NVMe over Fabricターゲットの検出

NVMe over Fabricターゲットで使用可能なNVMeサブシステムを一覧にするには、検出コントローラのアドレスとサービスIDが必要です。

```
tux > sudo nvme discover -t TRANSPORT -a DISCOVERY_CONTROLLER_ADDRESS -s SERVICE_ID
```

`TRANSPORT` は、基盤となる転送メディア(`loop`、`rdma`、または`fc`)で置き換ええます。`DISCOVERY_CONTROLLER_ADDRESS` は、検出コントローラのアドレスで置き換ええます。RDMAの場合、これはIPv4アドレスである必要があります。`SERVICE_ID` は、転送サービスIDで置き換ええます。RDMAのように、サービスがIPベースの場合、サービスIDはポート番号を指定します。ファイバチャネルの場合、サービスIDは必要ありません。

NVMeホストは、接続が許可されているサブシステムのみを参照します。

例:

```
tux > sudo nvme discover -t rdma -a 10.0.0.1 -s 4420
```

詳細については、`man nvme-discover`を参照してください。

16.2.3 NVMe over Fabricターゲットへの接続

NVMeサブシステムを特定した後で、`nvme connect`コマンドを使用して接続できます。

```
tux > sudo nvme connect -t transport -a DISCOVERY_CONTROLLER_ADDRESS -s SERVICE_ID -n SUBSYSTEM_NQN
```

`TRANSPORT` は、基盤となる転送メディア(`loop`、`rdma`、または`fc`)で置き換ええます。`DISCOVERY_CONTROLLER_ADDRESS` は、検出コントローラのアドレスで置き換ええます。RDMAの場合、これはIPv4アドレスである必要があります。`SERVICE_ID` は、転送サービスIDで置き換ええます。RDMAのように、サービスがIPベースの場合、これはポート番号を指定します。`SUBSYSTEM_NQN` は、検出コマンドによって検出された、目的のサブシステムのNVMe修飾名で置き換ええます。NQNは、NVMe修飾名(NVMe Qualified Name)の略語です。NQNは固有である必要があります。

例:

```
tux > sudo nvme connect -t rdma -a 10.0.0.1 -s 4420 -n nqn.2014-08.com.example:nvme:nvm-  
subsystem-sn-d78432
```

または、`nvme connect-all`を使用して、すべての検出されたネームスペースに接続します。高度な使用法については、`man nvme-connect`および`man nvme-connect-all`を参照してください。

16.2.4 マルチパス処理

NVMeネイティブマルチパス処理はデフォルトで有効になっています。マルチパスデバイスのレイアウトを印刷するには、`nvme list-subsys`コマンドを使用します。NVMeネイティブマルチパス処理を無効にするには、ブートパラメータとして`nvme-core.multipath=N`を追加します。

`multipath -ll` コマンドには互換性モードがあり、NVMeマルチパスデバイスを表示します。

16.3 NVMe over Fabricターゲットの設定

16.3.1 コマンドラインクライアントのインストール

NVMe over Fabricターゲットを設定するには、`nvmetcli` コマンドラインツールが必要です。インストールするには、`zypper`を実行します。

```
tux > sudo zypper in nvmetcli
```

`nvmetcli`の現在のドキュメントはhttp://git.infradead.org/users/hch/nvmetcli.git/blob_plain/HEAD:/Documentation/nvmetcli.txtから入手できます。

16.3.2 設定手順

次の手順に、NVMe over Fabricターゲットの設定方法の例を示します。

設定はツリー構造で格納されます。移動するには、`cd` コマンドを使用します。オブジェクトを一覧にするには、`ls`を使用します。`create`を使用して新しいオブジェクトを作成できます。

1. `nvmetcli` インタラクティブシェルを起動します。

```
tux > sudo nvmetcli
```

2. 新しいポートを作成します。

```
(nvmetcli)> cd ports  
(nvmetcli)> create 1  
(nvmetcli)> ls 1/  
o- 1  
  o- referrals  
  o- subsystems
```

3. NVMeサブシステムを作成します。

```
(nvmetcli)> cd /subsystems  
(nvmetcli)> create nqn.2014-08.org.nvmexpress:NVMf:uuid:c36f2c23-354d-416c-95de-f2b8ec353a82  
(nvmetcli)> cd nqn.2014-08.org.nvmexpress:NVMf:uuid:c36f2c23-354d-416c-95de-f2b8ec353a82/  
(nvmetcli)> ls  
o- nqn.2014-08.org.nvmexpress:NVMf:uuid:c36f2c23-354d-416c-95de-f2b8ec353a82  
  o- allowed_hosts  
  o- namespaces
```

4. 新しいネームスペースを作成し、そのネームスペースにNVMeデバイスを設定します。

```
(nvmetcli)> cd namespaces  
(nvmetcli)> create 1  
(nvmetcli)> cd 1  
(nvmetcli)> set device path=/dev/nvme0n1  
Parameter path is now '/dev/nvme0n1'.
```

5. 以前に作成したネームスペースを有効にします。

```
(nvmetcli)> cd ..  
(nvmetcli)> enable  
The Namespace has been enabled.
```

6. 作成したネームスペースを表示します。

```
(nvmetcli)> cd ..  
(nvmetcli)> ls  
o- nqn.2014-08.org.nvmexpress:NVMf:uuid:c36f2c23-354d-416c-95de-f2b8ec353a82  
  o- allowed_hosts  
  o- namespaces  
    o- 1
```

7. すべてのホストがサブシステムを使用できるようにします。この操作は、セキュリティ保護された環境でのみ実行します。


```
(nvmetcli)> set attr allow_any_host=1
Parameter allow_any_host is now '1'.
```

または、特定のホストのみが接続できるようにします。

```
(nvmetcli)> cd nqn.2014-08.org.nvmeexpress:NVMf:uuid:c36f2c23-354d-416c-95de-
f2b8ec353a82/allowed_hosts/
(nvmetcli)> create hostnqn
```

8. すべての作成されたオブジェクトを一覧にします。

```
(nvmetcli)> cd /
(nvmetcli)> ls
o- /
  o- hosts
  o- ports
    | o- 1
    |   o- referrals
    |   o- subsystems
  o- subsystems
    o- nqn.2014-08.org.nvmeexpress:NVMf:uuid:c36f2c23-354d-416c-95de-f2b8ec353a82
      o- allowed_hosts
      o- namespaces
        o- 1
```

9. RDMAを介してターゲットを使用できるようにします。

```
(nvmetcli)> cd ports/1/
(nvmetcli)> set addr adrfam=ipv4 trtype=rdma traddr=10.0.0.1 trsvcid=4420
Parameter trtype is now 'rdma'.
Parameter adrfam is now 'ipv4'.
Parameter trsvcid is now '4420'.
Parameter traddr is now '10.0.0.1'.
```

または、ファイバチャネルを介して使用可能にすることができます。

```
(nvmetcli)> cd ports/1/
(nvmetcli)> set addr adrfam=fc trtype=fc
traddr=nn-0x1000000044001123:pn-0x2000000055001123 trsvcid=none
```

16.3.3 ターゲット設定のバックアップと復元

次のコマンドを使用してJSONファイルにターゲット設定を保存できます。

```
tux > sudo nvmetcli
(nvmetcli)> saveconfig nvme-target-backup.json
```

設定を復元するには、次のコマンドを使用します。

```
(nvmetcli)> restore nvme-target-backup.json
```

現在の設定を消去することもできます。

```
(nvmetcli)> clear
```

16.4 特定のハードウェアの設定

16.4.1 概要

一部のハードウェアでは、正しく動作させるために特殊な設定が必要です。次の各セクションの見出しを参照し、記載されているデバイスまたはベンダのいずれかに該当しないか確認してください。

16.4.2 Broadcom

Broadcom Emulex LightPulse Fibre Channel SCSIドライバを使用している場合は、`lpfc` モジュールのターゲットおよびホスト上にカーネル設定パラメータを追加します。

```
tux > sudo echo "options lpfc lpfc_enable_fc4_type=3" > /etc/modprobe.d/lpfc.conf
```




Broadcomアダプタファームウェアのバージョンが11.4.204.33以降であることを確認します。現在のバージョンの `nvme-cli`、`nvmetcli`、およびカーネルがインストールされていることも確認してください。

ファイバチャネルポートをNVMeターゲットとして有効にするには、追加のモジュールパラメータを設定する必要があります。たとえば、`lpfc_enable_nvmet=COMMA_SEPARATED_WWPNS` と指定します。先行する `0x` とともにWWPNを入力します。たとえば、`lpfc_enable_nvmet=0x20000000055001122,0x20000000055003344` と指定します。一覧表で示されているWWPNのみがターゲットモードに設定されます。ファイバチャネルポートは、ターゲットまたはイニシエータとして設定できます。

16.5 詳細情報

`nvme` の機能の詳細については、`nvme nvme-help` を参照してください。

次のリンクには、NVMeおよびNVMe over Fabricの概要があります。

- <http://nvmexpress.org/> 
- http://www.nvmexpress.org/wp-content/uploads/NVMe_Over_Fabrics.pdf 
- <https://storpool.com/blog/demystifying-what-is-nvmeof> 

17 デバイスのマルチパスI/Oの管理

本項では、マルチパスI/O (MPIO)を使用して、サーバ/ブロックストレージデバイス間のマルチパスのフェールオーバーおよびパスの負荷分散を管理する方法について説明します。

17.1 マルチパスI/Oの理解

マルチパス処理とは、サーバのホストバスアダプタおよびデバイスのストレージコントローラ間で、複数の物理パスをまたいで、同じ物理または論理ブロックストレージデバイスと通信するサーバの機能です。これは、通常、FC (Fibre Channel)環境またはiSCSI SAN環境で行われます。複数のチャンネルが使用可能な際には、内蔵ストレージとのマルチ接続も可能です。

Linuxマルチ処理は、接続に耐障害性を与え、アクティブな接続全体に負荷を分散します。マルチパス処理が設定および実行されていると、自動的に、デバイス接続の障害が特定され、I/Oが代替の接続に再経路指定されます。

接続に関しては、多くのトラブルが欠陥のあるアダプタ、ケーブル、またはコントローラが原因で発生します。デバイスにマルチパスI/Oを設定すると、マルチパスドライバがデバイス間のアクティブな接続を監視します。マルチパスドライバは、アクティブなパスのI/Oエラーを検出すると、トラフィックをデバイスの指定セカンダリパスにフェールオーバーします。該当するパスが正常に戻ると、そのパスに制御を戻すことができます。

17.2 ハードウェアサポート

マルチパス処理ドライバとツールは、SUSE Linux Enterprise Serverが利用可能なすべてのアーキテクチャをサポートします。また、ほとんどのストレージアレイもサポートします。マルチパスデバイスを格納するストレージアレイは、マルチパス処理用のドライバとツールを使用するために、マルチパス処理をサポートする必要があります。一部のストレージアレイベンダは、独自のマルチパス処理管理ツールを提供しています。ベンダのハードウェアマニュアルを参照して、どのような設定が必要か判別してください。

17.2.1 マルチパス処理用に自動検出されるストレージアレイ

`multipath-tools` パッケージは、次のようなストレージアレイを自動的に検出します。

3PARdata VV
AIX NVDISK
AIX VDASD

APPLE Xserve RAID
COMPELNT Compellent Vol
COMPAQ/HP HSV101、HSV111、HSV200、HSV210、HSV300、HSV400、HSV 450
COMPAQ/HP MSA、HSV
COMPAQ/HP MSA VOLUME
DataCore SANmelody
DDN SAN DataDirector
DEC HSG80
DELL MD3000
DELL MD3000i
DELL MD32xx
DELL MD32xxi
DGC
EMC Clariion
EMC Invista
EMC SYMMETRIX
EUROLOGC FC2502
FSC CentricStor
FUJITSU ETERNUS_DX、DXL、DX400、DX8000
HITACHI DF
HITACHI/HP OPEN
HP A6189A
HP HSVX700
HP LOGICAL VOLUME
HP MSA2012fc、MSA 2212fc、MSA2012i
HP MSA2012sa、MSA2312 fc/i/sa、MCA2324 fc/i/sa、MSA2000s VOLUME
HP P2000 G3 FC|P2000G3 FC/iSCSI|P2000 G3 SAS|P2000 G3 iSCSI
IBM 1722-600
IBM 1724
IBM 1726
IBM 1742
IBM 1745、1746
IBM 1750500
IBM 1814
IBM 1815
IBM 1818
IBM 1820N00
IBM 2105800

IBM 2105F20
IBM 2107900
IBM 2145
IBM 2810XIV
IBM 3303 NVDISK
IBM 3526
IBM 3542
IBM IPR
IBM Nseries
IBM ProFibre 4000R
IBM S/390 DASD ECKD
IBM S/390 DASD FBA
Intel Multi-Flex
LSI/ENGENIO INF-01-00
NEC DISK ARRAY
NETAPP LUN
NEXENTA COMSTAR
Pillar Axiom
PIVOT3 RAIGE VOLUME
SGI IS
SGI TP9100、TP 9300
SGI TP9400、TP9500
STK FLEXLINE 380
STK OPENstorage D280
SUN CSM200_R
SUN LSCSM100_[IEFS]
SUN STK6580、STK6780
SUN StorEdge 3510、T4
SUN SUN_6180

ただし、他のほとんどのストレージアレイも有効です。ストレージアレイが自動的に検出されると、マルチパス処理のデフォルト設定が適用されます。デフォルト以外の設定を使用したい場合は、手動で `/etc/multipath.conf` ファイルを作成および設定する必要があります。自動検出されないハードウェアでも同じ作業が必要になります。詳細については、[17.6項「/etc/multipath.conf Fileの作成または修正」](#)を参照してください。

次の点に留意してください。

- 自動検出されたすべてのストレージレイがSUSE Linux Enterprise Serverでテスト済みというわけではありません。17.2.2項「マルチパス処理サポートについてテスト済みのストレージレイ」も参照してください。
- 一部のストレージレイでは、特定のハードウェアハンドラが必要なことがあります。ハードウェアハンドラは、パスグループの切り替え時とI/Oエラーの処理時に、ハードウェア固有のアクションを実行するカーネルモジュールです。詳細については、17.2.3項「特定のハードウェアハンドラを必要とするストレージレイ」を参照してください。
- `/etc/multipath.conf` ファイルを変更したら、そのたびに **dracut** `-f` を実行してシステム上に `initrd` を再作成する必要があります。続いて、実行した変更を有効にするため再起動します。

17.2.2 マルチパス処理サポートについてテスト済みのストレージレイ

次のベンダのストレージレイは、SUSE Linux Enterprise Serverでテスト済みです。

EMC
Hitachi
Hewlett-Packard/Compaq
IBM
NetApp
SGI

他のベンダのストレージレイもほとんど機能するはずです。該当するベンダのマニュアルを参照してください。 `multipath-tools` パッケージによって認識されるデフォルトストレージレイのリストについては、17.2.1項「マルチパス処理用に自動検出されるストレージレイ」を参照してください。

17.2.3 特定のハードウェアハンドラを必要とするストレージレイ

あるパスから他のパスにフェールオーバーするには特別なコマンドが必要なストレージレイ、または非標準の特別なエラー処理が必要なストレージレイには、より拡張されたサポートが必要なことがあります。したがって、デバイスマAPPERマルチパスサービスには、ハードウェアハンドラ用フックがあります。たとえば、そのようなEMC CLARiiON CXファミリアレイ用ハンドラが1つ、既に提供されています。

！ 重要: その他の情報

ハードウェアベンダのマニュアルを参照して、そのハードウェアハンドラをデバイスマッパーマルチパス用にインストールする必要があるかどうか判別してください。

`multipath -t` コマンドは、特定のハードウェアハンドラで特別な処理を必要とするストレージレイの内部テーブルを表示します。ただし、表示されるリストは、サポートされているレイの包括的なリストではありません。特別な処理を必要とし、`multipath-tools` の開発者がツールの開発中にアクセスしたレイだけがリストされます。

！ 重要: [Exceptions (例外)]

真のアクティブ/アクティブマルチパスサポートをもつレイは、特別な処理を必要としないので、`multipath -t` コマンドでは表示されません。

また、`multipath -t` テーブルでリストされている場合でも、必ずしも、その特定ハードウェアでSUSE Linux Enterprise Serverがテスト済みということではありません。テスト済みのストレージレイのリストについては、[17.2.2項「マルチパス処理サポートについてテスト済みのストレージレイ」](#)を参照してください。

17.3 マルチパス処理のプランニング

マルチパスI/Oソリューションのプランニング時には、本項のガイドラインに従ってください。

17.3.1 前提条件

- マルチパス処理は、デバイスレベルで管理されます。
- マルチパス処理対象のデバイスに使用するストレージレイで、マルチパス処理がサポートされている必要があります。詳細については、[17.2項「ハードウェアサポート」](#)を参照してください。
- サーバのホストバスアダプタおよびブロックストレージデバイスのバスコントローラ間に複数の物理パスが存在している場合のみ、マルチパス処理を設定する必要があります。論理デバイスのマルチパスは、サーバの見地から設定します。

- 一部のストレージアレイについては、アレイの物理および論理デバイスのマルチパス処理を管理するための独自のマルチパス処理ソフトウェアがベンダから提供されます。この場合は、ベンダの指示に従って、それらのデバイスのマルチパス処理を設定してください。
- 仮想化環境でマルチパス処理を使用する場合、マルチパス処理は、ホストサーバ環境で制御されます。デバイスのマルチパス処理を設定してから、デバイスを仮想ゲストマシンに割り当ててください。

17.3.2 ディスク管理タスク

マルチパスをもつ物理デバイスまたは論理デバイスのマルチパス処理を設定する前に、まず、次のようにディスク管理タスクを実行してください。

- サードパーティーツールで、物理ディスクを小さな論理ディスクに切り分けます。
- サードパーティーツールで、物理ディスクまたは論理ディスクをパーティションに分割します。稼働中のシステムでパーティションを変更した場合は、DM-MP(Device Mapper Multipath: デバイスマッパーマルチパス)モジュールによるそれら変更の自動的な検出や反映は行われません。DM-MPIOは再初期化する必要があり、それには、通常、再起動が必要です。
- サードパーティーのSANアレイ管理ツールを使用して、ハードウェアRAIDデバイスを作成および設定します。
- サードパーティーのSANアレイ管理ツールを使用して、LUNなどの論理デバイスを作成します。所定のアレイにサポートされている論理デバイスタイプは、アレイベンダによって異なります。

17.3.3 ソフトウェアRAID

LinuxのソフトウェアRAIDの管理ソフトウェアは、マルチパス処理の上で実行されます。複数のI/Oパスを持ち、ソフトウェアRAIDで使用予定の各デバイスは、まず、マルチパス処理用に設定してから、ソフトウェアRAIDデバイスとして作成する必要があります。マルチパスデバイスは自動検出できません。ソフトウェアRAIDは、その下で実行されているマルチパス処理管理を認識しません。

既存のソフトウェアRAID用のマルチパス処理の設定については、[17.12項「既存ソフトウェアRAID用マルチパスI/Oの設定」](#)を参照してください。

17.3.4 高可用性ソリューション

ストレージリソースのクラスタリング用の高可用性ソリューションは、各ノード上でマルチパス処理サービスをベースとして実行されます。各ノード上の `/etc/multipath.conf` ファイル内の構成設定が、クラスタ全体で同一であるようにしてください。

マルチパスデバイスがすべてのデバイス間で同じ名前であるようにしてください。詳細については、[17.9.1項「HAクラスタにおけるマルチパスデバイスの名前」](#)を参照してください。

LAN上のデバイスをミラーリングするDRBD (Distributed Replicated Block Device)高可用性ソリューションは、マルチパス処理をベースとして実行されます。複数のI/Oパスを持ち、DRBDソリューションで使用予定のデバイスごとに、マルチパス処理用デバイスを設定してから、DRBDを設定する必要があります。

17.3.5 `initrd`とシステム設定との同期を常に維持する

マルチパスを使用する際に最も重要な要件の1つは、ルートファイルシステムと、システムをブートするために必要な他のファイルシステムすべてについて、`initrd`とインストール済みシステムの動作が同じになるようにすることです。システムでマルチパスが有効になっている場合は`initrd`でも有効にする必要があり、その逆も同様です。詳細については、[17.5.1項「マルチパスI/Oサービスの有効化、無効化、起動、および停止」](#)を参照してください。

`initrd`とシステムが同期されていない場合、システムは正しくブートせず、起動手順を実行すると緊急シェルが起動します。このようなシナリオを回避または修復する方法については、[17.15.2項「マルチパスが有効な場合、ブート時にシステムが終了して緊急シェルが起動する」](#)を参照してください。

17.4 マルチパス管理ツール

SUSE Linux Enterprise Serverのマルチパス処理のサポートは、Linuxカーネルのデバイスマッパーマルチパスモジュールと `multipath-tools` ユーザスペースパッケージに基づいています。MDADM (Multiple Devices Administration)ユーティリティ(`multipath`)を使用すると、マルチパスデバイスの状態を表示できます。

17.4.1 デバイスマッパーマルチパスモジュール

デバイスマッパーマルチパス(DM-MP)モジュールは、Linuxにマルチパス処理機能を提供します。DM-MPIOは、SUSE Linux Enterprise Serverでのマルチパス処理の推奨ソリューションです。DM-MPIOは、SUSEによって完全にサポートされている製品に付属する唯一のマルチパス処理オプションです。

DM-MPIOは、多様なセットアップでマルチパス処理サブシステムを自動設定します。デバイスごとに最大8個のパスの設定がサポートされています。アクティブ/パッシブ(1つのパスがアクティブで、他のパスがパッシブ)またはアクティブ/アクティブ(ラウンドロビン方式の負荷分散で全パスがアクティブ)の構成がサポートされています。

DM-MPIOフレームワークは、2つの方法で拡張できます。

- 特定のハードウェアハンドラの使用詳細については、[17.2.3項「特定のハードウェアハンドラを必要とするストレージアレイ」](#)を参照してください。
- ラウンドロビンアルゴリズムより高度な負荷分散アルゴリズムの使用。

DM-MPIOのユーザスペースコンポーネントにより、自動的なパスの検出とグループ化のほか、自動的なパスの再テストが実行されるので、障害が発生したパスは、正常に戻ると自動的に復帰します。これにより、管理者の手間を最低限に抑えることができます。

DM-MPIOは、デバイス自体の障害ではなく、デバイスへのパスの障害からシステムを保護します。アクティブなパスの1つが失われると(たとえば、ネットワークアダプタが破損する、光ファイバケーブルが外れるなど)、残りのパスにI/Oをリダイレクトします。アクティブ/パッシブ構成の場合は、パスがパッシブパスの1つにフェールオーバーします。ラウンドロビン式負荷分散構成を使用している場合は、トラフィックの負荷が残りの正常なパス全体に分散されます。すべてのアクティブパスに障害が起きた場合は、アクティブでないセカンダリパスが有効になり、約30秒の遅延でフェールオーバーが開始されます。

ディスクアレイに複数のストレージプロセッサがある場合は、アクセスしたいLUNを所有するストレージプロセッサにSANスイッチが接続していることを必ず確認してください。ほとんどのディスクアレイでは、すべてのLUNが両方のストレージプロセッサに属しているので、両方の接続がアクティブです。



注記: ストレージプロセッサ

一部のディスクアレイでは、ストレージアレイがストレージプロセッサを介してトラフィックを管理するので、一度に1つのストレージプロセッサだけが提示されます。1つのプロセッサがアクティブとなり、もう1つのプロセッサは障害が発生するまでパッシブ

ブとなります。間違ったストレージプロセッサ(パッシブなパスをもつプロセッサ)に接続している場合は、予期されたLUNが表示されなかったり、それらのLUNが表示されてもアクセスしようとするとエラーが発生することがあります。

表 17.1: ストレージアレイのマルチパスI/O機能

ストレージアレイの機能	説明
アクティブ/パッシブコントローラ	<p>1つのコントローラはアクティブで、すべてのLUNに対応します。2つ目のコントローラは、スタンバイとして機能します。2つ目のコントローラは、オペレーティングシステムが冗長なパスを認識するように、マルチパスコンポーネントに対するLUNの提示も行います。プライマリコントローラに障害が発生した場合は、セカンダリコントローラが引き継ぎ、すべてのLUNに対応します。</p> <p>一部のアレイでは、LUNをさまざまなコントローラに割り当てることができます。所定のLUNは、そのアクティブコントローラとなる1つのコントローラに割り当てられます。一度に1つのコントローラがあらゆるLUNのディスクI/Oを処理し、2つ目のコントローラがそのLUNのスタンバイコントローラとなります。2つ目のコントローラは、パスの提示もしますが、ディスクI/Oは行えません。そのLUNを使用するサーバは、LUNの割り当て先のコントローラに接続します。LUNのセットに対するプライマリコントローラに障害が発生すると、セカンダリコントローラが引き継ぎ、すべてのLUNに対応します。</p>
アクティブ/パッシブコントローラ	<p>両方のコントローラが、すべてのLUNの負荷を共有し、あらゆるLUNのディスクI/Oを処理できます。1つのコントローラに障害が発生すると、2つ目のコントローラが自動的にすべてのトラフィックを処理します。</p>
負荷分散	<p>デバイスマッパーマルチパスドライバは、自動的に、すべてのアクティブパス全体にトラフィックの負荷を分散します。</p>
コントローラのフェールオーバー	<p>アクティブなコントローラがパッシブなコントローラにフェールオーバーすると、デバイスマッパーマルチパスドライバがホスト/スタンバイ間のパスを自動的に有効にし、それらをプライマリパスにします。</p>

ストレージレイの機能	説明
ブート/ルートデバイスのサポート	<p>マルチパス処理は、SUSE Linux Enterprise Server 10以降のルート(/)デバイスに対してサポートされます。ホストサーバは、ブートデバイス用の、現在アクティブなコントローラおよびストレージプロセッサに接続する必要があります。</p> <p>マルチパス処理は、SUSE Linux Enterprise Server 11以降の <u>/boot</u> デバイスに対してサポートされています。</p>

デバイスマッパーマルチパスは、マルチパスデバイスの各パスを個別のSCSIデバイスとして検出します。SCSIデバイス名は、/dev/sdNの形式をとります。ここで、Nは、デバイスに対して自動生成される文字であり、aで始まり、デバイスの生成に応じてシーケンシャルに発行されます(/dev/sda、/dev/sdbなど)。デバイス数が26を超えると、文字が2つ使用され、/dev/sdzの次のデバイスは/dev/sdaa、その次は、その次は/dev/sdabと続きます。

複数のパスが自動的に検出されない場合は、それらを /etc/multipath.conf ファイルで手動設定できます。multipath.conf ファイルは、システム管理者によって作成および設定されるまで存在しません。詳細については、[17.6項「/etc/multipath.conf Fileの作成または修正」](#)を参照してください。

17.4.2 マルチパスI/O管理ツール

パッケージ multipath-tools および kpartx では、自動パス検出とグループ化を扱うツールが提供されています。これらのパッケージは、自動的にパスの定期テストを行うので、障害が発生したパスは、正常に戻ると、自動的に復帰します。これにより、管理者の手間を最低限に抑えることができます。

表 17.2: **MULTIPATH-TOOLS**パッケージに含まれるツール

ツール	説明
<u>multipath</u>	システムをスキャンしてマルチパスデバイスを検出し、アセンブルします。
<u>multipathd</u>	mapsイベントを待機し、 <u>multipath</u> を実行します。
<u>kpartx</u>	マルチパスデバイス上のパーティションにリニアdevmapをマップします。これにより、デバイス上のパーティションのマルチパスモニタリングを作成することが可能になります。

ツール	説明
<u>mpathpersist</u>	デバイスマッパーマルチパスのデバイス.ppcでSCSIの永続的な予約を管理します。

17.4.3 マルチパスデバイスへのMDADMの使用

デフォルトのデバイスハンドラはUdevであり、デバイスは、デバイスノード名ではなく、Worldwide IDによって、システムに自動的に認識されます。これによって、環境設定ファイル(mdadm.confとlvm.conf)がマルチパスデバイスを正しく認識しないという、MDADMおよびLVMの旧リリースにあった問題が解決します。

LVM2の場合のようにmdadmでは、デバイスノードパスではなく、IDによってデバイスをアクセスする必要があります。したがって、/etc/mdadm.conf内のDEVICEエントリを次のように設定してください。

```
DEVICE /dev/disk/by-id/*
```

ユーザフレンドリな名前を使用している場合は、次のようにパスを指定し、マルチパス処理の設定後に、デバイスマッパー名だけがスキャンされるようにします。

```
DEVICE /dev/disk/by-id/dm-uuid-.*-mpath-.*
```

17.4.4 multipath コマンド

マルチパスデバイスを設定し、管理するには、**multipath(8)** コマンドを使用します。このコマンドの一般的な構文は、次のようになります。

```
multipath [-v verbosity_level] [-b bindings_file] [-d] [-h|-l|-ll|-f|-F|-B|-c|-q|-r|-w|-W|-t] [-p failover|multibus|group_by_serial|group_by_prio|group_by_node_name] [DEVICENAME]
```

詳細については、**man 8 multipath**を参照してください。

一般的な例

multipath

すべてのマルチパスデバイスを設定します。

multipath DEVICENAME

特定のマルチパスデバイスを設定します。

DEVICENAME を、/dev/sdb (udevにより\$DEVNAME変数で表示)または major:minor 形式などのデバイスノード名で置き換えます。デバイス名は、マルチパスマップ名でも構いません。

multipath -f

マルチパスマップとそのデバイスにマップされたパーティションを選択的に抑制します。

multipath -d

ドライ実行。可能性のあるマルチパスデバイスを表示しますが、デバイスの作成やデバイスマップの更新は行いません。

multipath -v2 -d

ドライ実行で可能性のあるマルチパスデバイスのマルチパスマップ情報を表示します。-v2オプションを使用すると、ローカルディスクのみが表示されます。この冗長レベルでは、kpartxなどの他のツールへのフィード用としてのみ、作成または更新したマルチパスの名前をプリントします。

デバイスがすでに存在し、変更がない場合には、出力はありません。設定されているマルチパスデバイスのステータスを見るには、multipath -llを使用します。

multipath -v2 DEVICENAME

特定の可能性のあるマルチパスデバイスを設定し、そのマルチパスマップ情報を表示します。この冗長レベルでは、kpartxなどの他のツールへのフィード用として、作成または更新したマルチパスの名前だけをプリントします。

デバイスがすでに存在し、変更がない場合には、出力はありません。設定されているマルチパスデバイスのステータスを見るには、multipath -llを使用します。

DEVICENAME を、/dev/sdb (udevにより\$DEVNAME変数で表示)または major:minor 形式などのデバイスノード名で置き換えます。デバイス名は、マルチパスマップ名でも構いません。

multipath -v3

可能性のあるマルチパスデバイスを設定し、それらのマルチパスマップ情報を表示します。この冗長レベルでは、すべての検出されたパス、マルチパス、およびデバイスマップがプリントされます。WWIDおよび devnode の両方でブラックリスト化されたデバイスが表示されます。

multipath -v3 DEVICENAME

特定の可能性のあるマルチパスデバイスを設定し、それらの情報を表示します。-v3オプションを使用すると、フルパスリストが表示されます。この冗長レベルでは、すべての検出されたパス、マルチパス、およびデバイスマップがプリントされます。WWIDおよび devnode の両方でブラックリスト化されたデバイスが表示されます。

DEVICENAME を、/dev/sdb (udev により \$DEVNAME 変数で表示) または major:minor 形式などのデバイスノード名で置き換えます。デバイス名は、マルチパスマップ名でも構いません。

multipath -ll

すべてのマルチパスデバイスの状態を表示します。

multipath -ll DEVICENAME

指定されたマルチパスデバイスのステータスを表示します。

DEVICENAME を、/dev/sdb (udev により \$DEVNAME 変数で表示) または major:minor 形式などのデバイスノード名で置き換えます。デバイス名は、マルチパスマップ名でも構いません。

multipath -F

すべての未使用のマルチパスデバイスマップをフラッシュします。これによって、マルチパスが解消したり、デバイスが削除されることはありません。

multipath -F DEVICENAME

指定されたマルチパスデバイスの未使用のマルチパスデバイスマップをフラッシュします。これによって、マルチパスが解消したり、デバイスが削除されることはありません。

DEVICENAME を、/dev/sdb (udev により \$DEVNAME 変数で表示) または major:minor 形式などのデバイスノード名で置き換えます。デバイス名は、マルチパスマップ名でも構いません。

multipath -p [failover | multibus | group_by_serial | group_by_prio | group_by_node_name]

次の表に説明されているグループポリシーオプションの1つを指定することにより、グループポリシーを設定します。

表 17.3: MULTIPATH -P コマンドのグループポリシーオプション

ポリシーオプション	説明
failover (フェールオーバー)	(デフォルト)優先度グループごとに1つのパス一度に使用できるスパスは1つだけです。
multibus	1つの優先度グループ内にすべてのパス
group_by_serial	検出されたSCSIシリアル番号(コントローラノードの全世界規模の番号)ごとに1つの優先度グループ

ポリシーオプション	説明
group_by_prio	パス優先度値ごとに1つの優先度グループ同じ優先度のパスは同じ優先度グループに属します。優先度は、コールアウトプログラムで決定されます。それらのプログラムは、グローバル、コントローラごと、またはマルチパスごとのオプションとして <code>/etc/multipath.conf</code> 環境設定ファイルで指定されます。
group_by_node_name	ターゲットノード名ごとに1つの優先度グループターゲットノード名は、 <code>/sys/class/fc_transport/target*/node_name</code> にフェッチされます。

multipath -t

マルチパスの内部ハードウェアテーブルとアクティブな設定を表示します。設定パラメータの詳細については、[man multipath](#) を参照してください。

17.4.5 mpathpersistユーティリティ

mpathpersist ユーティリティを使用して、デバイスマップーマルチパスのデバイスでSCSIの永続的な予約を管理できます。このコマンドの一般的な構文は、次のようになります。

```
mpathpersist [options] [device]
```

詳細については、[man 8 mpathpersist](#) を参照してください。

このユーティリティを `/etc/multipath.conf` ファイルのサービスアクション予約キー (`reservation_key` 属性) と共に使用して、SCSIデバイスの永続的な予約を設定します。この属性はデフォルトでは使用されません。この属性が設定されていない場合、**multipathd** デーモンは、新しく検出されたパスまたは復元されたパスの永続的な予約があるかどうかを確認しません。

```
reservation_key <RESERVATION_KEY>
```

この属性は `defaults` セクションまたは `multipaths` セクションに追加できます。次に例を示します。

```
multipaths {
    multipath {
        wwid      XXXXXXXXXXXXXXXXX
        alias      yellow
        reservation_key  0x123abc
    }
}
```

```
}  
}
```

永続的な管理の対象にするすべてのマルチパスデバイスに対して `reservation_key` パラメータを設定し、次のコマンドを実行して `multipathd` デーモンを再起動します。

```
tux > sudo systemctl restart multipathd
```

設定後、`mpathpersist` コマンドで予約キーを指定できます。

例

mpathpersist --out --register --param-sark=123abc --prout-type=5 -d /dev/mapper/mpath9

`/dev/mapper/mpath9` デバイスのサービスアクション予約キーを登録します。

mpathpersist -i -k -d /dev/mapper/mpath9

`/dev/mapper/mpath9` デバイスのサービスアクション予約キーを読み込みます。

mpathpersist --out --reserve --param-sark=123abc --prout-type=8 -d /dev/mapper/mpath9

`/dev/mapper/mpath9` デバイスのサービスアクション予約キーを予約します。

mpathpersist -i -s -d /dev/mapper/mpath9

`/dev/mapper/mpath9` デバイスの予約状態を読み込みます。

17.5 マルチパス処理用システムの設定

17.5.1 マルチパスI/Oサービスの有効化、無効化、起動、および停止

マルチパスサービスを有効にしてブート時に起動するには、次のコマンドを実行します。

```
tux > sudo systemctl enable multipathd
```

稼働中のシステムでサービスを手動で起動したり、サービスの状態を確認したりするには、次のいずれかのコマンドを入力します。

```
tux > sudo systemctl start multipathd
```

```
tux > sudo systemctl status multipathd
```

現在のセッションでマルチパスサービスを停止して無効にし、システムの次回ブート時にサービスが起動しないようにするには、次のコマンドを実行します。

```
tux > sudo systemctl stop multipathd  
tux > sudo systemctl disable multipathd
```

！ 重要: initrdの再構築

マルチパスサービスを有効または無効にした場合は、initrdの再構築も必要です。そうしないと、システムがブートしなくなるおそれがあります。マルチパスサービスを有効にする場合は、次のコマンドを実行してinitrdの再構築も行います。

```
tux > dracut --force --add multipath
```

マルチパスサービスを無効にする場合は、次のコマンドを実行してinitrdを再構築します。

```
tux > dracut --force -o multipath
```

(オプション)さらに、マルチパスを手動で起動するときにもマルチパスデバイスが設定されないようにする場合は、initrdを再構築する前に、/etc/multipath.confの最後に次の行を追加します。

```
blacklist {  
    wwid ".*"  
}
```

17.5.2 マルチパス処理用SANデバイスの準備

SANデバイスのマルチパスI/Oを設定する前に、必要に応じて、次のようにSANデバイスを準備してください。

- ベンダのツールで、SANデバイスを設定し、ゾーン化します。
- ベンダのツールで、ストレージアレイ上のホストLUNのパーミッションを設定します。
- Linux HBAドライバモジュールをインストールします。モジュールがインストールされると、ドライバがHBAを自動的にスキャンして、ホスト用のパーミッションをもつSANデバイスを検出します。それらのSANデバイスは、以降の設定のため、ホストに提示されます。



注記: ネイティブのマルチパス処理を有効化しない

ご使用のHBAドライバのネイティブマルチパス処理が有効化していないことを確認してください。

詳細については、ベンダの特定マニュアルを参照してください。

- ドライバモジュールがロードされたら、特定アレイのLUNまたはパーティションに割り当てられたデバイスノードを検出します。
- SANデバイスがサーバ上でルートデバイスとして使用される場合は、[17.14.9項「ルートデバイスがマルチパスの場合のSANタイムアウト設定」](#)に示されているように、デバイスのタイムアウト設定を変更します。

HBAドライバがLUNを認識しない場合は、`lsscsi`を使用して、SCSIデバイスがオペレーティングシステムによって正しく認識されているかどうかチェックできます。LUNがHBAドライバによって認識されない場合は、SANのゾーン化セットアップをチェックします。特に、LUNのマスキングがアクティブであるかどうか、LUNがサーバに正しく割り当てられているかどうかをチェックしてください。

LUNがHBAドライバによって認識されても、対応するブロックデバイスが存在しない場合は、カーネルパラメータを追加して、SCSIデバイスのスキャン動作を変更する必要があります(LUNが連続的に番号付けされていないことを示すなど)。詳細については、SUSEナレッジベース(<https://www.suse.com/support/kb/doc.php?id=3955167>)にあるTID 3955167: Troubleshooting SCSI (LUN) Scanning Issuesを参照してください。

17.5.3 マルチパスデバイスのパーティショニング

複数のパスをもつパーティショニングデバイスは、推奨できませんが、サポートされています。`kpartx` ツールを使用すると、再起動なしでマルチパスデバイスにパーティションを作成できます。マルチパス処理の設定前に、YaSTのパーティショナ機能またはサードパーティーのパーティショニングツールの使用により、デバイスをパーティショニングすることもできます。

マルチパスデバイスはデバイスマッパーデバイスです。コマンドラインツール(`parted`、`kpartx`、`fdisk`など)を使用してデバイスマッパーデバイスを変更することはできませんが、他の層を更新するために必要なudevイベントが生成されるとは限りません。デバイスマッパーデバイスをパーティション化した後、マルチパスマップをチェックして、デバイス

マッパーデバイスがマップされていることを確認する必要があります。デバイスが見つからない場合は、マルチパスデバイスを再マップするかサーバを再起動すると、マルチパスマップにある新しいパーティションをすべて検出できます。

マルチパスデバイス上にあるパーティションのデバイスマッパーデバイスは、独立したデバイスと同じではありません。デバイス全体を使用するLVM論理ボリュームを作成する場合、パーティションが含まれないデバイスを指定する必要があります。マルチパスパーティションをLVM論理ボリュームのターゲットデバイスとして設定すると、LVMは、ベースを成す物理デバイスがパーティション化されていると認識し、作成に失敗します。SANデバイスを再分割する必要がある場合、SANデバイス上のLUNを分割し、各LUNを別個のマルチパスデバイスとしてサーバに認識させることができます。

17.6 /etc/multipath.conf Fileの作成または修正

`/etc/multipath.conf` ファイルは、作成しない限り、存在しません。マルチパスの設定ファイルを作成して設定をパーソナライズしない限り、`multipathd` デモンの実行時にデフォルトのマルチパスデバイス設定が自動的に適用されます。

！ 重要: /etc/multipath.confからの変更のテストおよび恒久的な適用

`/etc/multipath.conf` ファイルの作成または修正を行った場合、ファイルを保存する際に変更が自動的に適用されません。これにより、変更をコミットする前に、それを検証するためにドライ実行を行うことができます。改訂した設定に満足な場合、[17.6.4 項「/etc/multipath.confファイルの変更を適用したマルチパスマップの更新」](#)の説明にあるようにマルチパスマップを更新できます。

17.6.1 /etc/multipath.confファイルの作成

1. 好みのエディタで空の `/etc/multipath.conf` ファイルを作成します。
2. SANの適切な `device` セクションを追加してください。大半のベンダは、`device` セクションの正しいセットアップに関するマニュアルを提供しています。異なるSANには別々の `device` セクションが必要です。
自動検出されたストレージサブシステムを使用している場合([17.2.1項「マルチパス処理用に自動検出されるストレージアレイ」](#)を参照)、`device` の追加は必要ありません。この場合、このデバイスのデフォルト設定が適用されます。

3. 使用中のマシンのマルチパス以外のデバイスをすべて含む `blacklist` セクションを作成します。詳細については、17.8項「非マルチパスデバイスのブラックリスト化」を参照してください。
4. 必要に応じて、設定ファイルにセクションを追加します。簡単な説明は、17.6.2項「`/etc/multipath.conf`ファイルのセクション」を参照してください。詳細は、`man 5 multipath.conf` を実行すると参照できます。
5. 終了したら、`/etc/multipath.conf` を保存し、17.6.3項「`/etc/multipath.conf`ファイルでのマルチパスセットアップの確認」の説明にあるように設定をテストします。
6. 設定の確認を完了したら、17.6.4項「`/etc/multipath.conf`ファイルの変更を適用したマルチパスマップの更新」の説明にあるようにこれを適用します。

17.6.2 `/etc/multipath.conf`ファイルのセクション

`/etc/multipath.conf` ファイルは、以下のセクションで構成されています。詳細は、`man 5 multipath.conf` を参照してください。

defaults

マルチパスI/Oの全般的デフォルト設定。これらの値は、適切なデバイスセクションまたはマルチパスセクションで値が指定されていない場合に使用されます。詳細については、17.7項「ポーリング、待ち行列、およびフェールバック用のデフォルトポリシーの設定」を参照してください。

blacklist

マルチパスの候補ではないとして破棄するデバイス名の一覧。デバイスは、そのデバイスノード名(`devnode`)、そのWWID(`wwid`)、またはそのベンダまたは製品文字列(`device`)によって識別できます。通常、非マルチパスデバイス(`hpsa`、`fd`、`hd`、`md`、`dm`、`sr`、`scd`、`st`、`ram`、`raw`、`loop` など)は無視できます。詳細と使用例については、17.8項「非マルチパスデバイスのブラックリスト化」を参照してください。

blacklist_exceptions

ブラックリストに記載されている場合でもマルチパスの候補として扱うデバイスのデバイス名の一覧。デバイスは、そのデバイスノード名(`devnode`)、そのWWID(`wwid`)、またはそのベンダまたは製品文字列(`device`)によって識別できます。対象のデバイスを指定するには、ブラックリストで使ったのと同じキーワードを使用する必要があります。たとえば、ブラックリスト内のデバイスに `devnode` キーワードを使用した場合は、`devnode` キーワードを使用して、ブラックリスト例外にあるデバイスの一部を除外

します。devnode キーワードを使用し、それらの一部のデバイスを wwid キーワードを使用して除外することで、デバイスをブラックリストに入れることはできません。詳細と使用例については、[17.8項「非マルチパスデバイスのブラックリスト化」](#)を参照してください。

multipaths

個々のマルチパスデバイスの設定を指定します。個別設定をサポートしていない設定を除き、これらの値により、設定ファイルの defaults および devices セクションで指定された値が上書きされます。

devices

個々のストレージコントローラの設定を指定します。これらの値により、設定ファイル内の defaults セクションで指定された値が上書きされます。デフォルトではサポートされていないストレージアレイを使用している場合は、devices サブセクションを作成して、そのデフォルト設定を指定することができます。これらの値は、個々のマルチパスデバイスの設定により上書きが可能です(キーワードでそれが許可されていれば)。詳細については、次のリンクを参照してください。

- [17.9項「ユーザフレンドリ名または別名の設定」](#)
- [17.14.6項「IBM Zデバイスのデフォルト設定」](#)

17.6.3 etc/multipath.confファイルでのマルチパスセットアップの確認

/etc/multipath.conf ファイルの作成または修正を行った場合、ファイルを保存する際に変更が自動的に適用されません。セットアップの「ドライ実行」を行って、マルチパスのマップを更新する前に、マルチパスセットアップを確認することができます。

サーバのコマンドプロンプトで、次のように入力します。

```
tux > sudo multipath -v2 -d
```

このコマンドによりデバイスがスキャンされ、変更をコミットしたときにセットアップがどのようになるかが表示されます。/etc/multipath.conf ファイルを修正してドライ実行を行う際に、変更前の(またはデフォルトの)マルチパス設定で、multipathd デーモンがすでに実行されていることを前提とします。変更内容に問題がなければ、次の手順に進みます。

出力の例を以下に示します。

```
26353900f02796769  
[size=127 GB]  
[features="0"]
```



```
[hwhandler="1    emc"]

\_ round-robin 0 [first]
  \_ 1:0:1:2 sdav 66:240 [ready ]
  \_ 0:0:1:2 sdr  65:16  [ready ]

\_ round-robin 0
  \_ 1:0:0:2 sdag 66:0   [ready ]
  \_ 0:0:0:2 sdc  8:32   [ready ]
```

パスは、優先度グループでグループ化されます。一度に1つの優先度グループだけがアクティブに使用されます。アクティブ/アクティブ構成をモデル化するには、すべてのパスを同じグループにします。アクティブ/パッシブ構成をモデル化する場合は、並行してアクティブにしないパスを複数の別の優先度グループに振り分けます。これは、通常、デバイス検出時に自動的行われます。

出力として、順序、グループ内でのI/O負荷の分散に使用されるスケジュールポリシー、および各優先度グループのパスが表示されます。また、各パスに対して、その物理アドレス(ホスト:バス:ターゲット:LUN)、デバイスノード名、メジャー:マイナー番号、および状態が表示されます。

ドライ実行で冗長レベルの-v3を使用することによって、すべての検出されたパス、マルチパス、およびデバイスマップを表示できます。WWIDおよびデバイスノードの両方でブラックリスト化されたデバイスが表示されます。

2つのQlogic HBAをXitech Magnitude 3000 SANに接続した64ビットSLES 11 SP2サーバでの-v3出力の例を、次に示します。例を短くするため、複数エントリの一部は省略されています。

```
tux > sudo multipath -v3 d
dm-22: device node name blacklisted
< content omitted >
loop7: device node name blacklisted
< content omitted >
md0: device node name blacklisted
< content omitted >
dm-0: device node name blacklisted
sdf: not found in pathvec
sdf: mask = 0x1f
sdf: dev_t = 8:80
sdf: size = 105005056
sdf: subsystem = scsi
sdf: vendor = XI0tech
sdf: product = Magnitude 3D
sdf: rev = 3.00
sdf: h:b:t:l = 1:0:0:2
sdf: tgt_node_name = 0x202100d0b2028da
sdf: serial = 000028DA0014
sdf: getuid= "/lib/udev/scsi_id --whitelisted --device=/dev/%n" (config file default)
```



```

sdf: uid = 200d0b2da28001400 (callout)
sdf: prio = const (config file default)
sdf: const prio = 1
[...]
raml5: device node name blacklisted
[...]
===== paths list =====
uuid          hcil      dev dev_t pri dm_st  chk_st  vend/prod/rev
200d0b2da28001400 1:0:0:2 sdf 8:80 1  [undef][undef] XI0tech,Magnitude 3D
200d0b2da28005400 1:0:0:1 sde 8:64 1  [undef][undef] XI0tech,Magnitude 3D
200d0b2da28004d00 1:0:0:0 sdd 8:48 1  [undef][undef] XI0tech,Magnitude 3D
200d0b2da28001400 0:0:0:2 sdc 8:32 1  [undef][undef] XI0tech,Magnitude 3D
200d0b2da28005400 0:0:0:1 sdb 8:16 1  [undef][undef] XI0tech,Magnitude 3D
200d0b2da28004d00 0:0:0:0 sda 8:0  1  [undef][undef] XI0tech,Magnitude 3D
params = 0 0 2 1 round-robin 0 1 1 8:80 1000 round-robin 0 1 1 8:32 1000
status = 2 0 0 0 2 1 A 0 1 0 8:80 A 0 E 0 1 0 8:32 A 0
sdf: mask = 0x4
sdf: path checker = directio (config file default)
directio: starting new request
directio: async io getevents returns 1 (errno=Success)
directio: io finished 4096/0
sdf: state = 2
[...]

```

17.6.4 /etc/multipath.confファイルの変更を適用したマルチパスマップの更新

/etc/multipath.conf ファイルに対する変更は、**multipathd**の実行中は有効になりません。変更を行ったら、ファイルを保存して閉じ、次のように、変更内容を適用してマルチパスマップを更新してください。

1. **multipathd** サービスを停止します。

```
tux > sudo systemctl stop multipathd
```

2. 次のコマンドの入力で、古いmultipathバインディングをクリアします。

```
tux > sudo /sbin/multipath -F
```

3. 次のコマンドの入力で、新しいmultipathバインディングを作成します。

```
tux > sudo /sbin/multipath -v2 -l
```

4. **multipathd** サービスを再起動します。

```
tux > sudo systemctl start multipathd
```

5. **dracut -f**を実行し、システム上に initrd イメージを再作成してから、再起動して変更内容を有効にします。

17.6.5 WWIDの生成

異なるパス上のデバイスを識別するため、マルチパスは、各デバイスに対してWorld Wide Identification (WWID)を使用します。2つのデバイスパスのWWIDが同じである場合、それらは同じデバイスを表すものと想定されます。やむを得ない理由がある場合を除き、WWIDの生成方法を変更しないことをお勧めします。詳細については、man multipath.confを参照してください。

17.7 ポーリング、待ち行列、およびフェールバック用のデフォルトポリシーの設定

マルチパスI/Oの最終目標は、ストレージシステムとサーバ間のコネクティビティ耐障害性を提供することです。望ましいデフォルトの動作は、サーバがスタンダロンのサーバか、高可用性クラスタ内のノードかによって異なります。

スタンドアロンサーバに対してマルチパスI/Oを構成する際は、no_path_retryの設定により、サーバのオペレーティングシステムを、I/Oエラーの受信から可能な限り保護することができます。この設定により、メッセージはマルチパスのフェールオーバーが発生するまで待ち行列に入れられ、正常な接続が保たれます。

高可用性クラスタ内のノードに対してマルチパスI/Oを構成するときには、マルチパスでリソースのフェールオーバーをトリガするためにI/O障害が報告されるようにして、マルチパスのフェールオーバーが解決されるのを待たなくて済むようにするとよいでしょう。クラスタ環境では、no_path_retry設定を、ストレージシステムへの接続が失われた場合に、クラスタノードがクラスタ検証プロセスに関連するI/Oエラー(ハートビート許容値の50%を推奨)を受信するように変更する必要があります。また、パスの障害によるリソースのピンポンを避けるため、マルチパスI/Oのフェールバックをマニュアルに設定するとよいでしょう。

/etc/multipath.conf ファイルには、ポーリング、待ち行列、およびフェールバックのデフォルト動作を指定できる **defaults** セクションが含まれています。 **device** セクションで、フィールドが別途指定されていない場合は、そのSAN構成にデフォルト設定が適用されます。デフォルト設定では、以下のようにコンパイルされています。パーソナライズした /etc/multipath.conf ファイルを作成して構成することでこれらの値を上書きしない限り、この設定が使用されます。

```
defaults {
```

```

verbosity 2
# udev_dir is deprecated in SLES 11 SP3
# udev_dir          /dev
polling_interval    5
# path_selector default value is service-time in SLES 11 SP3
# path_selector     "round-robin 0"
path_selector       "service-time 0"
path_grouping_policy failover
# getuid_callout is deprecated in SLES 11 SP3 and replaced with uid_attribute
# getuid_callout     "/lib/udev/scsi_id --whitelisted --device=/dev/%n"
# uid_attribute is new in SLES 11 SP3
uid_attribute       "ID_SERIAL"
prio                "const"
prio_args           ""
features            "0"
path_checker        "tur"
alias_prefix        "mpath"
rr_min_io_rq        1
max_fds             "max"
rr_weight           "uniform"
queue_without_daemon "yes"
flush_on_last_del   "no"
user_friendly_names "no"
fast_io_fail_tmo     5
bindings_file        "/etc/multipath/bindings"
wwids_file           "/etc/multipath/wwids"
log_checker_err      "always"

retain_attached_hw_handler "no"
detect_prio          "no"
failback             "manual"
no_path_retry        "fail"
}

```

ポーリング、待ち行列、およびフェールバックの詳細については、[17.10項「パスフェールオーバーのポリシーと優先度の設定」](#)に記載のパラメータを参照してください。

- [polling_interval](#)
- [no_path_retry](#)
- [failback](#) (フェールバック)

/etc/multipath.conf ファイルの変更後、**dracut -f** を実行してシステム上に initrd を再作成してから、サーバを再起動して変更内容を有効にする必要があります。詳細については、[17.6.4項「/etc/multipath.confファイルの変更を適用したマルチパスマップの更新」](#)を参照してください。

17.8 非マルチパスデバイスのブラックリスト化

`/etc/multipath.conf` ファイルに **blacklist** セクションを含め、すべての非マルチパスデバイスを一覧にできます。WWID (`wwid` キーワード)、デバイス名(`devnode` キーワード)、またはデバイスタイプ(`device` セクション)を使用してデバイスをブラックリスト化できます。`blacklist_exceptions` セクションを使って、`blacklist` セクションで使用している正規表現によってブラックリスト化された特定のデバイスに対してマルチパスを有効にすることもできます。



注記: 推奨するブラックリスト化方法

デバイスをブラックリスト化する場合に推奨する方法は、「WWID」または「ベンダーと製品」です。「devnode」によるブラックリスト化は推奨しません。デバイスノードは変わる可能性があり、デバイスを常時識別する目的では役に立たないからです。



警告: multipath.confの正規表現

`/etc/multipath.conf` では、正規表現は一般に「無効」です。正規表現は、一般的な文字列を検索する場合にのみ有効です。ただし、マルチパスの標準設定には、すでにさまざまなデバイスとベンダーを表す正規表現が含まれています。正規表現で別の正規表現を検索することはできません。`multipath -t` で表示される文字列のみを検索するようにしてください。

通常、非マルチパスデバイス

(`hpsa`、`fd`、`hd`、`md`、`dm`、`sr`、`scd`、`st`、`ram`、`raw`、`loop` など)は無視できます。たとえば、ローカルのSATAハードディスクやフラッシュディスクにはマルチパスはありません。`multipath` で単一パスデバイスを無視する場合は、それらのデバイスを **blacklist** セクションに記述します。



注記: 互換性

キーワード `devnode_blacklist` は廃止され、キーワード `blacklist` に代わりました。SUSE Linux Enterprise Server 12では、`glibc` で提供されている正規表現が使用されます。任意の文字列に一致させるには、`"**"` ではなく `".*"` を使用する必要があります。

たとえば、`hpsa` ドライバからローカルデバイスとすべてのアレイを、`multipath` による管理から外してブラックリストに載せるには、**blacklist** セクションを次のように指定します。

```
blacklist {
    wwid "26353900f02796769"
    devnode "^(ram|raw|loop|fd|md|dm-|sr|scd|st)[0-9]*"
    devnode "^sd[a-z][0-9]*"
}
```

アレイ全体でなく、ドライバからのパーティションだけをブラックリスト化することもできます。たとえば、次の正規表現を使用すると、アレイ全体ではなく、ccissドライバからのパーティションだけをブラックリスト化できます。

```
blacklist {
    devnode "^cciss!c[0-9]d[0-9]*[p[0-9]*]"
}
```

特定のデバイスタイプをブラックリスト化するには、ブラックリストに device セクションを追加して、キーワード vendor および product を使用します。

```
blacklist {
    device {
        vendor "DELL"
        product ".*"
    }
}
```

blacklist_exceptions セクションを使って、blacklist セクションで使用している正規表現によってブラックリスト化された特定のデバイスに対してマルチパスを有効にできます。WWID (wwid キーワード)、デバイス名 (devnode キーワード)、またはデバイスタイプ (device セクション) を使用して例外を追加します。例外は、対応するデバイスをブラックリスト化したときと同じ方法で指定する必要があります。つまり、wwid 例外は wwid ブラックリストに適用され、devnode 例外は devnode ブラックリストに適用され、デバイスタイプ例外はデバイスタイプブラックリストに適用されます。

たとえば、同じベンダのデバイスタイプが複数ある場合、目的のデバイスタイプに対してマルチパスを有効にできます。そのベンダのデバイスタイプすべてを blacklist セクションに記述してブラックリスト化してから、blacklist_exceptions セクションに device セクションを追加し、目的のデバイスタイプに対してマルチパスを有効にします。

```
blacklist {
    devnode "^(ram|raw|loop|fd|md|dm-|sr|scd|st|sda)[0-9]*"
    device {
        vendor "DELL"
        product ".*"
    }
}

blacklist_exceptions {
    device {
        vendor "DELL"
    }
}
```

```

        product "MD3220i"
    }
}

```

blacklist_exceptionsを使用して、特定のデバイスに対してのみマルチパスを有効にすることもできます。次に例を示します。

```

blacklist {
    wwid ".*"
}

blacklist_exceptions {
    wwid "3600d02300000000000e13955cc3751234"
    wwid "3600d02300000000000e13955cc3751235"
}

```

/etc/multipath.conf ファイルの変更後、**dracut -f** を実行してシステム上に **initrd** を再作成してから、サーバを再起動して変更内容を有効にする必要があります。詳細については、[17.6.4項「/etc/multipath.confファイルの変更を適用したマルチパスマップの更新」](#)を参照してください。

再起動後は、**multipath -ll** コマンドを発行しても、ローカルデバイスはマルチパスマップにリストされません。



注記: find_multipaths オプションの使用

SUSE Linux Enterprise Server 12 SP2より、マルチパスツールは、**/etc/multipath.conf** の **defaults** セクションでオプション **find_multipaths** をサポートするようになりました。簡単に言うと、このオプションは、マルチパスと **multipathd** が、パスが1つだけのデバイスにマルチパスマップを設定しないようにします(詳細については、**man 5 multipath.conf** を参照してください)。特定の設定では、これによって、管理者がローカルSATAディスクなどのブラックリストエントリを作成する手間を省くことができます。

find_multipaths オプションを使用すると一見便利そうですが、欠点もあります。まず、システムのブートが複雑化して低速になります。見つかったすべてのデバイスについて、そのデバイスに2つ目のパスが存在するかどうかを確認するため、すべてのデバイスが検出されるまでブートロジックが待機しなければならないからです。さらに、ブート時に一部のパスがダウンしていたり、他の理由で不可視になっていたりすると、問題が発生する可能性もあります。つまり、デバイスがシングルパスデバイスとして誤検出されてアクティブ化され、後で他のパスを追加できなくなる可能性があります。

find_multipaths は、WWIDが一致すれば、**/etc/multipath/wwids** に一覧にされたデバイスをすべてマルチパスデバイスとみなします。これは、**find_multipaths** を初めて有効にする場合に重要です。wwidsファイルには既存のすべてのマルチパスマップ

(シングルパスマップを含む)が一覧にされているため、`/etc/multipath/wwids` を削除または編集しない限り、このオプションを有効にしても効果はありません。マルチパスルートファイルシステムを持つSANブートシステムでは、初期RAMディスクとファイルシステムとの間で `/etc/multipath/wwids` の同期が維持されるようにしてください。

まとめると、`find_multipaths` を使用すると便利ですが、SUSEは、これまでと同様に適切に設定されたブラックリストとブラックリスト例外を使うデフォルト設定をお勧めします。

17.9 ユーザフレンドリ名または別名の設定

マルチパスデバイスは、そのWWID、ユーザフレンドリな名前、またはそれに割り当てた別名で識別されます。`/dev/sdn` および `/dev/dm-n` の形式のデバイスノード名は、再起動の際に変わる可能性があり、毎回異なるデバイスに割り当てられることになります。デバイスのWWID、ユーザフレンドリ名、および別名は、再起動の際にも変わることなく、デバイスの識別には望ましい方法です。

！ 重要: 永続的な名前の使用の推奨

`/dev/sdn` および `/dev/dm-n` 形式のデバイスノード名は、再起動時に変更される可能性があるので、マルチパスデバイスは、そのWWIDで参照することを推奨します。また、再起動時にデバイスを一意に識別するために、WWIDにマップされたユーザフレンドリ名または別名を使用することもできます。

次の表では、`/etc/multipath.conf` ファイル内のデバイスに使用できるデバイス名のタイプについて説明しています。`multipath.conf` 設定の例については、`/usr/share/doc/packages/multipath-tools/multipath.conf.synthetic` ファイルを参照してください。

表 17.4: マルチパスデバイス名のタイプの比較

名前のタイプ	説明
WWID (デフォルト)	シリアルWWID (Worldwide Identifier)は、グローバルに固有または非変更であることを保証されたマルチパスデバイスの識別子です。マルチパス処理で使用するデフォルト名は、 <code>/dev/disk/by-id</code> ディレクトリにある論理ユニットのIDです。たとえば、WWID が <code>3600508e0000000009e6baa6f609e7908</code> のデバイスは、 <code>/dev/disk/by-id/scsi-3600508e0000000009e6baa6f609e7908</code> と記載されています。

名前のタイプ	説明
ユーザフレンドリ	<u>/dev/mapper</u> ディレクトリ内のデバイスマッパーマルチパスのデバイス名は、論理ユニットのIDも参照します。これらのマルチパスデバイス名は、 <u>/dev/mapper/mpathN</u> 形式を使用するユーザフレンドリな名前です(たとえば、 <u>/dev/mapper/mpath0</u>)。これらの名前は、 <u>/var/lib/multipath/bindings</u> ファイルを使用してUUIDとユーザフレンドリな名前の関連付けを追跡するので、固有かつ永続的です。
別名	別名は、グローバルに固有な名前であり、管理者がマルチパスデバイスに提供します。別名は、WWIDとユーザフレンドリな <u>/dev/mapper/mpathN</u> 名に優先します。 user_friendly_nameを使用している場合は、別名をmpath_N形式に設定しないでください。mpathN形式にすると、自動的に割り当てられたユーザフレンドリ名と競合し、デバイスノードが正しくなくなる可能性があります。

/etc/multipath.conf ファイルのグローバルマルチパスオプション

user_friendly_names は、マルチパスデバイスのユーザフレンドリ名の使用を有効または無効にするために使用されます。このオプションが no (デフォルト)に設定されている場合、マルチパスはデバイス名としてWWIDを使用します。このオプションが yes に設定されている場合は、/var/lib/multipath/bindings ファイルが使用されて、mpath<N>形式の永続的で固有の名前が、/dev/mapper ディレクトリ内でデバイスに割り当てられます。/etc/multipath.conf ファイルの bindings file オプションを使用すると、bindings ファイルに代替の場所を指定できます。

/etc/multipath.conf ファイルのグローバルマルチパスオプション alias は、デバイスに名前を明示的に割り当てるために使用されます。別名がマルチパスデバイスに設定されている場合は、WWIDまたはユーザフレンドリ名の代わりにその別名が使用されます。

user_friendly_names オプションの使用は、以下の状況では問題を引き起こす可能性があります。

ルートデバイスでマルチパスを使用している場合:

システムルートデバイスでマルチパスを使用中に、user_friendly_names オプションを使用する場合は、option, the user-friendly settings in the /var/lib/multipath/bindings ファイルのユーザフレンドリ設定が initrd に組み込まれます。デバイスの追

加や削除などで、後でストレージのセットアップを変更した場合は、initrd内のバインディング設定と /var/lib/multipath/bindings 内のバインディング設定に不一致が生じます。



警告: バインディングの不一致

initrd と /var/lib/multipath/bindings のバインディングが不一致だと、デバイスに間違ったマウントポイントが割り当てられることがあり、その場合は、ファイルシステムが破損し、データが失われます。

この問題を回避するには、システムルートデバイスにデフォルトのWWID設定を使用することを推奨します。システムのルートデバイスには、別名を使用してはなりません。デバイス名が異なることがあるため、別名を使用すると、カーネルのコマンドラインを通じてマルチパス処理をシームレスにスイッチオフすることができなくなります。

別のパーティションから/varをマウントする場合:

user_friendly_names 設定ファイルのデフォルトの格納場所は、/var/lib/multipath/bindings です。/var データがシステムルートデバイス上になく、このデータを別のパーティションからマウントする場合は、マルチパス処理のセットアップ時に bindings ファイルを利用できません。

/var/lib/multipath/bindings ファイルをシステムルートデバイスで使用し、マルチパスで検出できるようにしてください。これは、たとえば、次の手順で実行できます。

1. /var/lib/multipath/bindings ファイルを /etc/multipath/bindings に移動します。
2. この新しい場所に、/etc/multipath.conf の defaults セクションにある bindings_file オプションを設定します。次に例を示します。

```
defaults {  
    user_friendly_names yes  
    bindings_file "/etc/multipath/bindings"  
}
```

マルチパスがinitrdに含まれている場合:

システムルートデバイスがマルチパス上でない場合でも、マルチパスが initrd に含まれることがあります。これは、たとえば、システムルートデバイスがLVM上にある場合に起こります。user_friendly_names オプションを使用し、マルチパスが initrd 内にある場合は、パラメータ multipath=off でブートして問題を回避してください。

これにより、システム起動中は、`initrd`内でのみマルチパスが無効になります。システム起動後は、ブートスクリプト `boot.multipath` および `multipathd` によって、マルチパス処理を有効にすることができます。

HAクラスタでマルチパス処理を行う場合:

詳細については、[17.9.1項「HAクラスタにおけるマルチパスデバイスの名前」](#)を参照してください。

ユーザフレンドリな名前を有効にするか、別名を指定する場合:

1. `root` 特権を使用して `/etc/multipath.conf` ファイルをテキストエディタで開きます。
2. (オプション) `/var/lib/multipath/bindings` ファイルの場所を変更します。
代替パスは、マルチパスが代替パスを見つけることができるシステムルートデバイス上に存在する必要があります。

- a. `/var/lib/multipath/bindings` ファイルを `/etc/multipath/bindings` に移動します。
- b. この新しい場所に、`/etc/multipath.conf` の `defaults` セクションにある `bindings_file` オプションを設定します。次に例を示します。

```
defaults {
    user_friendly_names yes
    bindings_file "/etc/multipath/bindings"
}
```

3. (オプション、非推奨)ユーザフレンドリ名の有効にする:
 - a. `defaults` セクションとその閉じ括弧を非コメント化します。
 - b. `user_friendly_names` オプションを非コメント化し、次に、その値をNoからYesに変更します。
次に例を示します。

```
## Use user-friendly names, instead of using WWIDs as names.
defaults {
    user_friendly_names yes
}
```

4. (オプション) `alias` オプション(`multipath` セクションにある)を使用して、独自のデバイス名を指定します。
次に例を示します。

```
## Use alias names, instead of using WWIDs as names.
```

```

multipaths {
    multipath {
        wwid          36006048000028350131253594d303030
        alias          blue1
    }
    multipath {
        wwid          36006048000028350131253594d303041
        alias          blue2
    }
    multipath {
        wwid          36006048000028350131253594d303145
        alias          yellow1
    }
    multipath {
        wwid          36006048000028350131253594d303334
        alias          yellow2
    }
}

```

! 重要: WWWIDとWWNの比較

`/etc/multipath.conf` ファイル内でデバイスの別名を定義する場合は、必ず各デバイスのWWID (3600508e00000000009e6baa6f609e7908 など)を使用し、そのWWNは使用しないようにしてください。WWNは、デバイスIDの最初の文字を `0x` で置き換えます(0x600508e00000000009e6baa6f609e7908 など)。

5. 変更内容を保存し、ファイルを閉じます。
6. `/etc/multipath.conf` ファイルの変更後、**dracut** `-f` を実行してシステム上に `initrd` を再作成してから、サーバを再起動して変更内容を有効にする必要があります。詳細については、[17.6.4項「`/etc/multipath.conf`ファイルの変更を適用したマルチパスマップの更新](#)」を参照してください。

LUNディレクトリ全体を使用する場合は(たとえばSAN機能を使用してストレージのパーティションを行っている場合など)、**mkfs**、`/etc/fstab`、ご使用のアプリケーションなどに、`/dev/disk/by-id/xxx` という名前を使用することができます。パーティションで分割されたデバイスは、デバイス名の後ろに `_part<n>` が付加されます(`/dev/disk/by-id/xxx_part1` など)。

`/dev/disk/by-id` ディレクトリでは、マルチパスのマップ処理がなされたデバイスは、`dm-uuid*` 名または別名(`/etc/multipath.conf` ファイル内で別名を割り当てている場合)で表されます。 `scsi-` および `wwn-` のデバイス名は、そのデバイスへの物理的パスを表します。

17.9.1 HAクラスタにおけるマルチパスデバイスの名前

以下を行って、マルチパスデバイスがすべてのデバイス間で同じ名前であるようにしてください。

- UUIDと別名を使用して、マルチパスデバイスの名前が、クラスタ内のすべてのノードで同一となるようにします。別名は、すべてのノードにわたって一意である必要があります。 `/etc/multipath.conf` ファイルを、ノードからクラスタ内の他のすべてのノードの `/etc/` ディレクトリにコピーします。
- マルチパスがマップされたデバイスを使用する場合は、 `dm-uuid*` 名または別名を `/dev/disk/by-id` ディレクトリ内で指定し、デバイスの固定パスインスタンスは指定しないようにします。詳細については、17.9項「ユーザフレンドリ名または別名の設定」を参照してください。
- `user_friendly_names` 構成オプションを、無効にしないよう設定します。ユーザフレンドリ名はノードに固有ですが、クラスタ内のすべてのノードにおいてデバイスに同じユーザフレンドリ名が割り当てられてはいない可能性があります。



注記: ユーザフレンドリ名

実際にユーザフレンドリ名を使用する必要がある場合は、以下の操作により、システム定義のユーザフレンドリ名を、クラスタ内のすべてのノードについて同一にすることができます。

- 1つのノード上の `/etc/multipath.conf` ファイル内で、
 1. `user_friendly_names` 構成オプションをyesに設定して有効にします。マルチパスは、 `/var/lib/multipath/bindings` ファイルを使用して、 `/dev/mapper` ディレクトリ内で `mpath<N>` の形式で、デバイスに永続的かつ固有の名前を割り当てます。
 2. (オプション) `bindings` ファイルに対して別の場所を指定するには、 `/etc/multipath.conf` ファイルの `defaults` セクションにある、 `bindings_file` オプションを設定します。デフォルトの場所は、 `/var/lib/multipath/bindings` です。
2. ノード上のマルチパスデバイスをすべて設定します。
3. `/etc/multipath.conf` ファイルを、ノードからクラスタ内の他のすべてのノードの `/etc/` ディレクトリにコピーします。

4. `bindings` ファイルを、ノードから、クラスタ内の他のすべてのノード上の `bindings_file` パスにコピーします。
5. `/etc/multipath.conf` ファイルの変更後、`dracut -f` を実行してシステム上に `initrd` を再作成してから、ノードを再起動して変更内容を有効にする必要があります。詳細については、[17.6.4項「/etc/multipath.confファイルの変更を適用したマルチパスマップの更新」](#)を参照してください。これは、影響を受けるすべてのノードに適用されます。

17.10 パスフェールオーバーのポリシーと優先度の設定

Linuxホスト内で、ストレージコントローラへのパスが複数ある場合は、各パスが別個のブロックデバイスとして表示され、その結果、1つのLUNに複数のブロックデバイスが存在することになります。デバイスマッパーマルチパスサービスは、同じLUN IDをもつ複数のパスワードを検出し、そのIDで新しいマルチパスデバイスを作成します。たとえば、1つの非ゾーン化されたファイバチャネルのスイッチを介して2つのポートでストレージコントローラに接続した2つのHBAをもつホストは、4つのブロックデバイスを認識します(`/dev/sda`、`/dev/sdb`、`/dev/sdc`、`/dev/sdd`)。デバイスマッパーマルチパスサービスは、1つのブロックデバイス `/dev/mppath/mpath1` を作成します。このデバイスは、既に示した4つのブロックデバイスを介してI/Oを再経路指定します。

本項では、フェールオーバーのポリシーを指定し、パスの優先順位を設定する方法について説明します。`/etc/multipath.conf` ファイルの変更後、`dracut -f` を実行してシステム上に `initrd` を再作成してから、サーバを再起動して変更内容を有効にする必要があることに注意してください。詳細については、[17.6.4項「/etc/multipath.confファイルの変更を適用したマルチパスマップの更新」](#)を参照してください。

17.10.1 パスのフェールオーバーポリシーの設定

`multipath` コマンドを `-p` オプション付きで使用して、パスフェールオーバーポリシーを設定します。

```
tux > sudo multipath DEVICENAME -p POLICY
```

次のポリシーオプションの1つで、`POLICY` を置き換えます。

表 17.5: **MULTIPATH -P**コマンドのグループポリシーオプション

ポリシーオプション	説明
failover (フェールオーバー)	(デフォルト)優先度グループごとに1つのパス
multibus	1つの優先度グループ内にすべてのパス
group_by_serial	検出されたシリアル番号ごとに1つの優先度グループ
group_by_prio	パス優先度値ごとに1つの優先度グループ優先度は、コールアウトプログラムで決定されます。それらのプログラムは、グローバル、コントローラごと、またはマルチパスごとのオプションとして <code>/etc/multipath.conf</code> 環境設定ファイルで指定されます。
group_by_node_name	ターゲットノード名ごとに1つの優先度グループターゲットノード名は、 <code>/sys/class/fc_transport/target*/node_name</code> にフェッチされます。

17.10.2 フェールオーバーポリシーの設定

デバイスのフェールオーバーポリシーは、手動で、`/etc/multipath.conf` ファイルに入力する必要があります。すべての設定とオプションの例は、`/usr/share/doc/packages/multipath-tools/multipath.conf.annotated` ファイルにあります。

17.10.2.1 優先度グループと属性の理解

優先度グループは、同じ物理LUNに属するパスのコレクションです。デフォルトでは、I/Oは、グループ内のすべてのパス全体にラウンドロビン方式で配分されます。`multipath` コマンドは、SANの`path_grouping_policy`設定に基づいてそのSANの各LUNごとに、自動的に優先度グループを作成します。`multipath` コマンドは、グループ内のパス数にグループの優先度を掛け合わせて、どのグループがプライマリか決定します。計算された値が最も高いグループがプライマリグループです。プライマリグループ内のすべてのパスが失敗すると、次に値の高い優先度グループがアクティブになります。

パス優先度は、パスに割り当てられた整数値です。値が高いほど、優先度が高くなります。パスごとに優先度を割り当てるには、外部プログラムが使用されます。所定のデバイスに関して、同じ優先度のパスが同じ優先度グループに属します。

prio 設定は、/etc/multipath.conf ファイルの defaults{} または devices{} セクションで使用します。multipath{} セクションの個別の multipaths 定義に指定されている場合は、暗黙のうちに無視されます。prio 行で、Prioritizerが指定されます。Prioritizerが引数を必要とする場合、その引数は2行目の prio_args キーワードで指定します。

デフォルトセクションまたはデバイスセクションのprio設定

prio

パス優先度の値を取得するために呼び出すPrioritizerプログラムを指定します。加重は、障害の発生時に使用する次のパスグループを決定するため、それぞれのパスグループに対して合計されます。

指定したPrioritizerで引数が必要な場合は、prio_args キーワードを使用して、引数を指定します。

prio_ キーワードを指定しない場合は、すべてのパスが同等になりますデフォルトの設定は const で、prio_args の設定には値がありません。

```
prio "const"
prio_args ""
```

Prioritizerのプログラム例には、以下のものがあります。

Prioritizerプログラム	説明
<u>alua</u>	SCSI-3 ALUA設定に基づいてパス優先度を生成します。
<u>const</u>	すべてのパスに同じ優先度を生成します。
<u>emc</u>	EMCアレイのパス優先度を生成します。
<u>hdc</u>	Hitachi HDS Modularストレージアレイのパス優先度を生成します。
<u>hp_sw</u>	アクティブ/スタンバイモードのCompaq/HPコントローラのパス優先度を生成します。
<u>ontap</u>	NetAppアレイのパス優先度を生成します。
<u>random</u>	パスごとにランダムな優先度を生成します。
<u>rdac</u>	LSI/Engenio RDACコントローラのパスの優先度を生成します。

Prioritizerプログラム	説明
<u>weightedpath</u>	<p><u>prio_args</u> に対する引数内で指定した加重値に基づいて、パス優先度を生成します。たとえば</p> <pre>[hctl devname] REGEX1 PRI01 REGEX2 PRI02...</pre> <p><u>hctl regex</u> の引数形式では、SCSI H:B:T:L 表記法 (1:0:... および *:0:0:. など) を、加重値とともに使用します。ここで、H、B、T、L はそれぞれ、デバイスのホスト、バス、ターゲット、および LUN ID を示します。たとえば、</p> <pre>prio "weightedpath" prio_args "hctl 1:0:0:0 2 4:0:0:0 4"</pre> <p><u>devname</u> の正規表現の引数形式では、デバイスノード名を、各ノードの加重値とともに使用します。次に例を示します。</p> <pre>prio "weightedpath" prio_args "devname sda 50 sde 10 sdc 50 sdf 10"</pre>

prio_args

引数を必要とする指定した Prioritizer プログラムに対する引数を指定します。ほとんどの prio プログラムでは、引数は不要です。

デフォルト値はありません。値は、prio の設定と、Prioritizer が引数を必要とするかどうかによります。

マルチパス属性

デバイスに対するマルチパス I/O の動作を制御するには、マルチパス属性を使用します。すべてのマルチパスデバイスに対して、デフォルトとして属性を指定できます。また、あるマルチパスデバイスにのみ適用する属性を、そのデバイス用のエントリを、マルチパス設定ファイルの multipaths セクションで作成することで、指定することもできます。

user_friendly_names

WWID(world-wide ID)を使用するか、または /var/lib/multipath/bindings ファイルを使用して永続的で固有な別名を /dev/mapper/mpathN 形式のマルチパスデバイスに割り当てるか指定します。

このオプションは、devices セクションおよび multipaths セクションで使用できます。

値	説明
<u>x</u>	(デフォルト) <code>/dev/disk/by-id/</code> に示されたWWIDを使用します。
<u>yes</u>	実際のIDの代わりに、マルチパスデバイスのエイリアスとして、ユーザフレンドリな名前を自動生成します。

failback (フェールバック)

エラーになったパスの回復を監視するかどうか指定し、パスサービス回復後のグループのフェールバックのタイミングを示します。

エラーになったパスは、回復すると、この設定に基づいてマルチパス対応パスのリストに戻されます。`multipath`は、優先度グループを評価し、プライマリパスの優先度がセカンドリパスのそれを超えると、アクティブな優先度グループを変更します。

値	説明
<u>manual</u>	デフォルト。エラーになったパスの回復は監視されません。管理者が <code>multipath</code> コマンドを実行して、有効なパスと優先度グループを更新します。
<u>followover</u>	パスグループの最初のパスがアクティブになるときにのみ自動フェールバックを実行します。これにより、別のノードがフェールオーバーを要求したときに、ノードが自動的にフェールバックされないようにします。
<u>immediate</u>	パスが回復したら、ただちにパスを有効にします。
<u>N</u>	パスが回復したら、 <u>N</u> 秒後にパスを有効にします。0より大きい整数値を指定してください。

クラスタ環境内のマルチパスに対するフェールバックの設定は、マルチパスのフェールオーバーのピンポンを避けるため、`manual` にすることを推奨します。

```
failback "manual"
```

! 重要: 検証

フェールバックの設定については、ストレージシステムのベンダに確認するようにしてください。ストレージシステムが異なれば、必要な設定も異なります。

no_path_retry

パスの障害時に使用する動作を指定します。

値	説明
<u>N</u>	multipath コマンドで待ち行列が停止し、パスがエラーになるまでの再試行数を指定します。0より大きい整数値を指定してください。 クラスタでは、「0」を指定して、待ち行列を回避し、リソースのフェールオーバーを許可することができます。
<u>fail</u>	即時失敗(待ち行列なし)を指定します。
<u>queue</u>	待ち行列を停止しません(パスが復帰するまで永久に待機します)。

クラスタでの作業では、`/etc/multipath.conf` ファイルの再試行設定を、`fail` または `0` にすることを推奨します。これにより、ストレージへの接続が失われた場合に、リソースのフェールオーバーが起こります。そうしないと、メッセージの待ち行列とリソースのフェールオーバーが行えません。

```
no_path_retry "fail"  
no_path_retry "0"
```



重要: 検証

再試行設定については、ストレージシステムのベンダに確認するようにしてください。ストレージシステムが異なれば、必要な設定も異なります。

path_checker

パスの状態を判別します。

値	説明
<u>directio</u>	直接I/Oを持つ最初のセクタを読み込みます。DASDデバイスの場合、有効です。障害メッセージを <code>systemd</code> ジャーナル

値	説明
	に記録します(『管理ガイド』、第15章「 <code>journalctl:systemd</code> ジャーナルのクエリ」を参照してください)。
<code>tur</code>	デバイスに対してSCSIテストユニットレディコマンドを発行します。これはLUNによってサポートされている場合の推奨設定です。このコマンドは、障害時に、 <code>systemd</code> ログジャーナルにメッセージを出力しません。
<code>CUSTOM_VENDOR_VALUE</code>	<p>一部のSANベンダは、カスタムオプションとして<code>path_checker</code>を提供しています。</p> <ul style="list-style-type: none"> • <code>cciss_tur</code>: HP Smart Storage Arrayへのパスの状態をチェックします。 • <code>emc_clariion</code>: EMC ClariionのEVPDページ0xC0をクエリしてパスの状態を判別します。 • <code>hp_sw</code>: Active/StandbyファームウェアをもつHPストレージアレイのパスの状態(アップ、ダウン、またはゴースト)をチェックします。 • <code>rdac</code>: LSI/Engenio RDACストレージコントローラのパスmp状態をチェックします。

`path_grouping_policy`

所定のコントローラがホストとなるマルチパスデバイスのパスグループ化ポリシーを指定します。

値	説明
<code>フェールオーバー</code>	(デフォルト)一度に1つのパスだけ使用されるように、優先度グループごとに1つのパスが割り当てられます。

値	説明
<u>multibus</u>	すべての有効なパスが1つの優先度グループに含まれます。トラフィックが、グループ内のアクティブなパスすべてに渡って負荷分散されます。
<u>group_by_prio</u>	パス優先度値ごとに、1つの優先度グループが存在します。同じ優先度のパスは同じ優先度グループに属します。優先度は外部プログラムによって割り当てられます。
<u>group_by_serial</u>	パスがSCSIターゲットシリアル番号(コントローラノードのWWN)でグループ化されます。
<u>group_by_node_name</u>	ターゲットノード名ごとに1つの優先度グループが割り当てられます。ターゲットノード名は <u>/sys/class/fc_transport/target*/node_name</u> にフェッチされます。

path_selector

負荷分散に使用するパスセレクトアルゴリズムを指定します。

値	説明
<u>round-robin 0</u>	優先度グループ内のすべてのアクティブパスに渡るトラフィックの分散に使用される負荷分散アルゴリズム。
<u>queue-length 0</u>	least-pendingオプションと同様に、パス上で実行中のI/Oの数に基づく、動的負荷分散装置。
<u>service-time 0</u>	(デフォルト)遅延に従って、パス上のI/Oを調整するサービス時間に基づく負荷分散装置。

pg_timeout

パスグループのタイムアウト処理を指定します。値を指定することはできません。内部のデフォルトが設定されています。

polling_interval

1つのパスチェックサイクルの終了から次のパスチェックサイクルの開始までの時間を、秒単位で指定します。

0より大きい整数値を指定してください。デフォルト値は5です。polling_intervalの設定については、ストレージシステムのベンダに確認するようにしてください。ストレージシステムが異なれば、必要な設定も異なります。

rr_min_io_rq

現在のパスグループ内の次のパスに切り替える前に、リクエストベースのデバイス-マップ-マルチパスを使用して、あるパスヘルートするI/Oリクエストの回数を指定します。

0より大きい整数値を指定してください。デフォルト値は「1」です。

```
rr_min_io_rq "1"
```

rr_weight

パスの重み付けの方法を指定します。

値	説明
<u>uniform</u>	(デフォルト)すべてのパスが同じラウンドロビン方式の重み付けを持ちます。
<u>priorities</u>	各パスの重み付けは、パスの優先度にrr_min_io_rq設定値を掛け合わせて決定します。

uid_attribute

固有のパス識別子を提供するudev属性。デフォルト値は ID_SERIAL です。

17.10.2.2 ラウンドロビン式負荷分散の設定

すべてのパスがアクティブです。一定の秒数または一定数のI/Oトランザクションの後で、シーケンスの次のオープンパスに移動するように、I/Oを設定します。

17.10.2.3 単一パスフェールオーバーの設定

優先度が最も高い（最も低い値の）単一パスがトランザクションに対してアクティブになります。他のパスは、フェールオーバーに使用できますが、フェールオーバーの発生までは使用されません。

17.10.2.4 ラウンドロビン式負荷分散用I/Oパスのグループ化

同じ優先度をもつ複数のパスがアクティブグループを形成します。そのグループのすべてのパスがエラーになると、デバイスが優先度の次に高いグループにフェールオーバーします。グループのすべてのパスが、ラウンドロビン方式の負荷分散で、トラフィックロードを共有します。

17.10.3 ターゲットパスグループの報告

SCSIターゲットポートグループの報告(`sg_rtpg(8)`)コマンドを使用します。詳細については、`sg_rtpg(8)`のマニュアルページを参照してください。

17.11 ルートデバイスのマルチパスI/Oの設定

SUSE Linux Enterprise Serverでは、DM-MPIO (デバイスマッパマルチパスI/O)が使用可能であり、`/boot`と`/root`に対してサポートされています。また、YaSTインストーラ内のYaSTパーティションは、インストール中のマルチパスの有効化をサポートします。

17.11.1 インストール時にマルチパスI/Oを有効にする

オペレーティングシステムをマルチパスデバイスにインストールするには、マルチパスソフトウェアがインストール時に実行されている必要があります。`multipathd`デーモンは、システムのインストール時に自動的にアクティブになりません。このデーモンは、YaSTパーティションのマルチパスの設定オプションを使用することによって起動できます。

17.11.1.1 アクティブ/アクティブマルチパスストレージLUNでインストール時にマルチパスI/Oを有効にする

1. インストール時に推奨されたパーティション分割画面でエキスパートパーティションを選択します。
2. ハードディスクメインアイコンを選択し、設定ボタンをクリックし、最後に、マルチパスの設定を選択します。
3. `multipath`を起動します。
YaSTがディスクの再スキャンを開始し、利用可能なマルチパスデバイスを表示します(`/dev/disk/by-id/dm-uuid-mpath-3600a0b80000f4593000012ae4ab0ae65` など)。これが、以降の処理すべての対象デバイスになります。
4. 次へをクリックして、インストールを続行します。

17.11.1.2 アクティブ/パッシブマルチパスストレージLUNでインストール時にマルチパスI/Oを有効にする

`multipathd` デーモンは、システムのインストール時に自動的にアクティブになりません。このデーモンは、YaSTパーティショナのマルチパスの設定オプションを使用することによって起動できます。

アクティブ/パッシブマルチパスストレージLUNに対するインストール時にマルチパスI/Oを有効にするには:

1. インストール時に推奨されたパーティション分割画面でエキスパートパーティショナを選択します。
2. ハードディスクメインアイコンを選択し、設定ボタンをクリックし、最後に、マルチパスの設定を選択します。
3. `multipath` を起動します。
YaSTがディスクの再スキャンを開始し、利用可能なマルチパスデバイスを表示します(`/dev/disk/by-id/dm-uuid-mpath-3600a0b80000f4593000012ae4ab0ae65` など)。これが、以降の処理すべての対象デバイスになります。デバイスのパスとUUIDを書き留めてください。後で必要になります。
4. 次へをクリックして、インストールを続行します。
5. すべての設定が完了し、インストールが終了すると、YaSTは、ブートローダ情報の書き込みを開始し、システム再起動のカウントダウンを表示します。中止をクリックしてカウンタを中止し、**Ctrl - Alt - F5>** を押してコンソールにアクセスします。
6. コンソールを使用して、`/boot/grub/device.map` ファイルの `hd0` エントリにパッシブパスが入力されているかどうか判別します。
これは、インストールではアクティブパスとパッシブパスが区別されない所以需要です。

- a. 次のように入力して、ルートデバイスを `/mnt` にマウントします。

```
tux > sudo mount /dev/disk/by-id/UUID;_part2 /mnt
```

例えば、次のように入力して、すべてのフォントについてアンチエイリアスを無効にします。

```
tux > sudo mount /dev/disk/by-id/dm-uuid-mpath-3600a0b80000f4593000012ae4ab0ae65_part2 /mnt
```

- b. 次のように入力して、ブートデバイスを `/mnt/boot` にマウントします。

```
tux > sudo mount /dev/disk/by-id/UUID_part1 /mnt/boot
```

例えば、次のように入力して、すべてのフォントについてアンチエイリアスを無効にします。

```
tux > sudo mount /dev/disk/by-id/dm-uuid-  
mpath-3600a0b80000f4593000012ae4ab0ae65_part2 /mnt/boot
```

- c. `/mnt/boot/grub/device.map` ファイルで `hd0` エントリがパッシブパスをポイントしているかどうか判別し、次のいずれかを実行します。

- **アクティブパス:** 操作は必要ありません。残りの手順をすべてスキップし、`Ctrl - Alt - F7>` を押してYaSTグラフィック環境に戻り、インストールを続行します。
- **パッシブパス:** 設定を変更し、ブートローダを再インストールする必要があります。

7. `hd0` エントリがパッシブパスをポイントする場合は、設定を変更し、ブートローダを再インストールします。

- a. コンソールプロンプトで、次のコマンドを入力します。

```
mount -o bind /dev /mnt/dev  
mount -o bind /sys /mnt/sys  
mount -o bind /proc /mnt/proc  
chroot /mnt
```

- b. コンソールで、`multipath -ll` を実行し、その出力をチェックして、アクティブパスを見つけます。

パッシブパスには `ghost` フラグが付いています。

- c. `/boot/grub/device.map` ファイルで `hd0` エントリをアクティブパスに変更し、変更内容を保存し、ファイルを閉じます。

- d. 次のコマンドを入力して、ブートローダを再インストールします。

```
grub-install /dev/disk/by-id/UUID_part1 /mnt/boot
```

例えば、次のように入力して、すべてのフォントについてアンチエイリアスを無効にします。

```
grub-install /dev/disk/by-id/dm-uuid-  
mpath-3600a0b80000f4593000012ae4ab0ae65_part2 /mnt/boot
```


e. 次のコマンドを入力します。

```
exit
umount /mnt/*
umount /mnt
```

8. **Ctrl** - **Alt** - **F7** を押して、YaSTグラフィック環境に戻ります。

9. OKをクリックして、インストールを再起動します。

17.11.2 既存ルートデバイス用マルチパスI/Oの有効化

1. Linuxをインストールし、1つだけパスをアクティブにします。このパスは、パーティションで by-id シンボリックリンクがリストされるパスがお勧めです。
2. インストール時に使用した /disk//disk/by-id パスを使用してデバイスをマウントします。
3. 次の行を追加して、dm-multipathを /etc/dracut.conf.d/01-dist.conf に追加します。

```
force_drivers+="dm-multipath"
```

4. IBM Zの場合、**dracut** の実行前に、/etc/zipl.conf ファイルを編集して zipl.conf 内の by-path 情報を、/etc/fstab で使用された by-id 情報に変更します。
5. **dracut** **-f** を実行して、initrd イメージを更新します。
6. IBM Zの場合は、**dracut** の実行後、**zipl** を実行します。
7. サーバを再起動します。

17.11.3 ルートデバイスのマルチパスI/Oの無効化

multipath=off をカーネルコマンドラインに追加します。この変更はYaSTのブートローダモジュールで行うことができます。ブートローダのインストール > Kernel Parameters (カーネルパラメータ) の順に開き、両方のコマンドラインにパラメータを追加します。

これは、ルートデバイスだけに影響します。他のすべてのデバイスは影響されません。

17.12 既存ソフトウェアRAID用マルチパスI/Oの設定

理想的には、デバイスのマルチパス処理を設定してから、それらのデバイスをソフトウェアRAIDデバイスのコンポーネントとして使用してください。ソフトウェアRAIDデバイスの作成後にマルチパス処理を追加した場合は、再起動時に **multipath** サービスの後でDM-MPIOサービスが開始することがあります。その場合は、マルチパス処理がRAIDに使用できないように見えます。本項の手順を使用すると、すでに存在しているソフトウェアRAIDに対してマルチパス処理を実行できます。

たとえば、次のような場合は、ソフトウェアRAID内のデバイスにマルチパス処理を設定する必要があります。

- 新規インストールまたはアップグレード時にパーティショニング設定の一部として、新しいソフトウェアRAIDを作成する場合
- マルチパス処理用に設定しなかったデバイスをメンバーデバイスまたはスペアとしてソフトウェアRAIDで使用する場合
- 新しいHBAアダプタをサーバに追加するか、またはSAN内でストレージサブシステムを拡張することで、システムを大きくする場合



注記: 前提

以降の説明では、ソフトウェアRAIDデバイスを `/dev/mapper/mpath0` (カーネルによって認識されるデバイス名)と想定しています。 `/etc/multipath.conf` ファイルで、ユーザフレンドリ名を有効にしている(17.9項「[ユーザフレンドリ名または別名の設定](#)」に記載)ことを想定しています。

ソフトウェアRAIDのデバイス名の指定は、必ず変更してください。

1. 端末コンソールを開きます。
特に指示のない限り、この端末を使用して、以降のステップでコマンドを入力します。
2. ソフトウェアRAIDデバイスが現在マウントされているか、または実行中の場合、デバイスごとに次のコマンドを入力して、デバイスをアンマウントし、停止します。

```
tux > sudo umount /dev/mapper/mpath0
tux > sudo mdadm --misc --stop /dev/mapper/mpath0
```

3. 次のように入力して、**md** サービスを停止します。

```
tux > sudo systemctl stop mdmonitor
```

4. 次のコマンドを入力することにより、multipathd デーモンを起動します。

```
tux > systemctl start multipathd
```

5. マルチパス処理サービスの開始後、ソフトウェアRAIDのコンポーネントデバイスが /dev/disk/by-id ディレクトリにリストされているかどうか確認します。次のいずれかの操作を行います。

- **デバイスがリストされている:** デバイス名に、デバイスマッパーマルチパスのデバイス名 (/dev/dm-1 など)へのシンボリックリンクがあるはずです。
- **デバイスがリストされていない:** 次のように入力して、デバイスをフラッシュし、再検出することで、マルチパスサービスにデバイスを認識させます。

```
tux > sudo multipath -F  
tux > sudo multipath -v0
```

これで、デバイスが /dev/disk/by-id 内にリストされ、デバイスマッパーマルチパスのデバイス名へのシンボリックリンクを持ちます。次に例を示します。

```
lrwxrwxrwx 1 root root 10 2011-01-06 11:42 dm-uuid-  
mpath-36006016088d014007e0d0d2213ecdf11 -> ../../dm-1
```

6. 次のように入力して、mdmonitor サービスとRAIDデバイスを再起動します。

```
tux > sudo systemctl start mdmonitor
```

7. 次のように入力して、ソフトウェアRAIDの状態をチェックします。

```
tux > sudo mdadm --detail /dev/mapper/mpath0
```

RAIDのコンポーネントデバイスは、そのデバイスマッパーマルチパスのデバイス名 (/dev/disk/by-id ディレクトリにデバイスのシンボリックリンクとしてリストされている) と一致する必要があります。

8. ルート(/)デバイス、またはそのいずれかの要素 (/var、/etc、/log など)がSAN上にあり、ブートするためにマルチパスが必要な場合、initrd を再構築します。

```
tux > dracut -f --add-multipath
```

9. サーバを再起動して、変更内容を適用します。

10. RAIDステータスをチェックして、ソフトウェアRAIDアレイが、マルチパスデバイスの上に正しく示されることを確認します。以下を入力してください。

```
tux > sudo mdadm --detail /dev/mapper/mpath0
```

次に例を示します。

Number	Major	Minor	RaidDevice	State
0	253	0 0	active sync	/dev/dm-0
1	253	1 1	active sync	/dev/dm-1
2	253	2 2	active sync	/dev/dm-2



注記: マルチパスデバイスでのmdadmの使用

mdadm ツールでは、デバイスのノードパスではなく、IDでデバイスにアクセスする必要があります。詳細については、[17.4.3項「マルチパスデバイスへのMDADMの使用」](#)を参照してください。

17.13 マルチパスデバイスでのLVM2の使用

マルチパス使用時に、リソースへのすべてのパスがデバイスツリーのデバイスとして存在します。デフォルトでは、LVMは、デバイスツリーの任意のデバイス上にマルチパスデバイスがあるかどうかを確認します。LVMがマルチパスデバイスを検出すると、そのデバイスはマルチパスコンポーネントであるとみなされ、(基盤となっている)デバイスは無視されます。ほとんどの場合はこの動作で問題ありませんが、[/etc/lvm/lvm.conf](#)で設定を変更できます。multipath_component_detectionを0に設定すると、LVMはマルチパスコンポーネントデバイスをスキャンしません。lvm.confのデフォルトのエントリは次のとおりです。

```
# By default, LVM2 will ignore devices used as component paths
# of device-mapper multipath devices.
# 1 enables; 0 disables.
multipath_component_detection = 1
```

17.14 ベストプラクティス

17.14.1 新規デバイスのスキャン(再起動なし)

ご使用のシステムがマルチパス処理用に設定されており、後からSANにストレージを追加する必要がある場合は、**rescan-scsi-bus.sh** スクリプトを使用して新しいデバイスをスキャンすることができます。デフォルトでは、このスクリプトは典型的なLUN範囲ですべてのHBAをスキャンします。このコマンドの一般的な構文は、次のようになります。

```
tux > sudo rescan-scsi-bus.sh [options] [host [host ...]]
```

ほとんどのストレージサブシステムでは、このスクリプトはオプションを指定しなくても正常に実行されます。ただし、特殊な場合は、次のオプションを1つ以上使用する必要があります。詳細については、**rescan-scsi-bus.sh --help** を実行してください。



警告: EMC PowerPath環境

EMC PowerPath環境では、SCSIバスをスキャンする場合に、オペレーティングシステムに付属する **rescan-scsi-bus.sh** ユーティリティまたはHBAベンダスクリプトを使用しないでください。ファイルシステムが破損する可能性を避けるため、EMCでは、Linux用EMC PowerPathのベンダマニュアルに記載されている手順に従うよう求めています。

次のプロシージャを使用して、システムを再起動せずに、デバイスをスキャンして、マルチパス処理に使用できるようにします。

1. ストレージサブシステムで、ベンダのツールを使用してデバイスを割り当て、そのアクセス制御設定を更新して、Linuxシステムが新しいストレージにアクセスできるようにします。詳細については、ベンダのマニュアルを参照してください。
2. すべてのターゲットをスキャンしてホストの有無を調べ、LinuxカーネルのSCSIサブシステムのみドルレイヤに新しいデバイスを認識させます。端末コンソールのプロンプトで、次のように入力します。

```
tux > sudo rescan-scsi-bus.sh
```

セットアップによっては、オプションのパラメータを指定して **rescan-scsi-bus.sh** を実行しなければならない場合があります。詳細については、**rescan-scsi-bus.sh --help** を参照してください。

3. `systemd` ジャーナルでスキンの進行状況を確認します(詳細については、『管理ガイド』、第15章「`journalctl:systemd`ジャーナルのクエリ」を参照してください)。端末コンソールのプロンプトで、次のように入力します。

```
tux > sudo journalctl -r
```

このコマンドは、ログの最後の行を表示します。次に例を示します。

```
tux > sudo journalctl -r
Feb 14 01:03 kernel: SCSI device sde: 81920000
Feb 14 01:03 kernel: SCSI device sdf: 81920000
Feb 14 01:03 multipathd: sde: path checker registered
Feb 14 01:03 multipathd: sdf: path checker registered
Feb 14 01:03 multipathd: mpath4: event checker started
Feb 14 01:03 multipathd: mpath5: event checker started
Feb 14 01:03:multipathd: mpath4: remaining active paths: 1
Feb 14 01:03 multipathd: mpath5: remaining active paths: 1
[...]
```

4. 前の各手順を繰り返し、新しいデバイスに接続しているLinuxシステム上の他のHBAアダプタを介して、パスを追加します。
5. `multipath` コマンドを実行して、DM-MPIO設定用のデバイスを認識します。端末コンソールのプロンプトで、次のように入力します。

```
tux > sudo multipath
```

これで、新しいデバイスをマルチパス処理用に設定できます。

17.14.2 パーティショニングされた新規デバイスのスキャン(再起動なし)

本項の例を使用して、新たに追加したマルチパスLUNを再起動なしで検出します。



警告: EMC PowerPath環境

EMC PowerPath環境では、SCSIバスをスキャンする場合に、オペレーティングシステムに付属する `rescan-scsi-bus.sh` ユーティリティまたはHBAベンダスクリプトを使用しないでください。ファイルシステムが破損する可能性を避けるため、EMCでは、Linux用EMC PowerPathのベンダマニュアルに記載されている手順に従うよう求めています。

1. 端末コンソールを開きます。
2. すべてのターゲットをスキャンしてホストの有無を調べ、LinuxカーネルのSCSIサブシステムのミドルレイヤに新しいデバイスを認識させます。端末コンソールのプロンプトで、次のように入力します。

```
tux > rescan-scsi-bus.sh
```

セットアップによっては、オプションのパラメータを指定して **rescan-scsi-bus.sh** を実行しなければならない場合があります。詳細については、**rescan-scsi-bus.sh --help** を参照してください。

3. 次のように入力して、デバイスが認識されていること(リンクに新しいタイムスタンプが付いているかどうかなど)を確認します。

```
tux > ls -lrt /dev/dm-*
```

次のように入力して、/dev/disk/by-id 内のデバイスを確認することもできます。

```
tux > ls -l /dev/disk/by-id/
```

4. 次のように入力して、新しいデバイスがログに表示されることを確認します。

```
tux > sudo journalctl -r
```

5. テキストエディタで、デバイスの新しいエイリアス定義を /etc/multipath.conf ファイルに追加します(data_vol3 など)。
たとえば、UUIDが 36006016088d014006e98a7a94a85db11 であれば、次の変更を行います。

```
defaults {
    user_friendly_names    yes
}
multipaths {
    multipath {
        wwid      36006016088d014006e98a7a94a85db11
        alias     data_vol3
    }
}
```

6. 次の入力で、デバイスのパーティションテーブルを作成します。

```
tux > fdisk /dev/disk/by-id/dm-uuid-mpath-<UUID>
```

UUIDをデバイスのWWID(36006016088d014006e98a7a94a85db11 など)で置き換えます。

7. 次のように入力して、udevをトリガします。

```
tux > sudo echo 'add' > /sys/block/DM_DEVICE/uevent
```

たとえば、dm-8上のパーティションに対して、デバイスマッパードデバイスを生成するには、次のように入力します。

```
tux > sudo echo 'add' > /sys/block/dm-8/uevent
```

8. デバイス /dev/disk/by-id/dm-uuid-mpath-UUID_partN 上にファイルシステムを作成します。選択するファイルシステムに応じて、このために **mkfs.btrfs**、**mkfs.ext3**、**mkfs.ext4**、または **mkfs.xfs** のいずれかのコマンドを使用できます。詳細については、それぞれのマニュアルページを参照してください。UUID_partNを、実際のUUIDおよびパーティション番号(36006016088d014006e98a7a94a85db11_part1など)で置き換えます。

9. 次のコマンドを入力して、新しいパーティションのラベルを作成します。

```
tux > sudo tune2fs -L LABELNAME /dev/disk/by-id/dm-uuid-UUID_partN
```

UUID_partNを、実際のUUIDおよびパーティション番号(36006016088d014006e98a7a94a85db11_part1など)で置き換えます。LABELNAMEは好みのラベルに代えてください。

10. 次の入力で、DM-MPIOを再設定して、エイリアスを読み込ませます。

```
tux > sudo multipathd -k'reconfigure'
```

11. 次の入力で、デバイスが multipathd によって認識されていることを確認します。

```
tux > sudo multipath -ll
```

12. テキストエディタで、/etc/fstab ファイルにマウントエントリを追加します。この時点では、前の手順で作成したエイリアスは、まだ /dev/disk/by-label ディレクトリにあります。マウントエントリを /dev/dm-9 パスに追加した後、次の再起動の前に、マウントエントリを次のように変更します。

```
LABEL=LABELNAME
```

13. マウントポイントとして使用するディレクトリを作成し、デバイスをマウントします。

17.14.3 マルチパスI/Oステータスの表示

マルチパスI/Oのステータスをクエリすると、マルチパスマップの現在のステータスが出力されます。

multipath -l オプションを使用すると、パスチェッカが最後に実行された時点での現行パスステータスが表示されます。ただし、パスチェッカは実行されません。

multipath -ll オプションを使用すると、パスチェッカが実行され、パス情報が更新され、最後に、現在のステータス情報が表示されます。このコマンドは、常にパスステータスの最新情報を表示します。

```
tux > sudo multipath -ll
3600601607cf30e00184589a37a31d911
[size=127 GB][features="0"][hwhandler="1 emc"]

\_ round-robin 0 [active][first]
  \_ 1:0:1:2 sdav 66:240 [ready ][active]
  \_ 0:0:1:2 sdr  65:16  [ready ][active]

\_ round-robin 0 [enabled]
  \_ 1:0:0:2 sdag 66:0   [ready ][active]
  \_ 0:0:0:2 sdc  8:32  [ready ][active]
```

デバイスごとに、デバイスのID、サイズ、機能、およびハードウェアハンドラが表示されます。

デバイスへのパスは、自動的に、デバイス検出時に優先度グループとしてグループ化されます。一度に1つの優先度グループだけがアクティブになります。アクティブ/アクティブ構成の場合、すべてのパスが同じグループに属します。アクティブ/パッシブ構成の場合、パッシブパスは別個の優先度グループに属します。

グループごとに、次の情報が表示されます。

- ラウンドロビン方式など、グループ内でのI/O負荷の分散に使用されるスケジューリングポリシー
- グループがアクティブか、無効か、または有効か
- 最初の(優先度の最も高い)グループかどうか
- グループ内に含まれるパス

パスごとに、次の情報が表示されます。

- HOST:BUS:TARGET:LUN としての物理アドレス(1:0:1:2など)
- デバイスノード名(sda など)

- メジャー/マイナー番号
- デバイスのステータス

17.14.4 エラーになったI/Oの管理

queue_if_no_pathを有効にすることで、すべてのパスで同時に障害が発生した場合は、I/Oをキューに登録するように、マルチパス処理を設定する必要があるかもしれません。設定しておかないと、すべてのパスに障害が発生するとI/Oもすぐに失敗してしまいます。ドライバ、HBA、またはファブリックにスプリアスエラーが発生したというシナリオでは、それらのエラーですべてのパスが失われるI/Oをすべて待ち行列に入れ、エラーを上方にプロパゲートしないように、DM-MPIOを設定してください。

マルチパスデバイスをクラスタで使用する場合は、queue_if_no_pathを無効にすることができます。これにより、I/Oがキューに入る代わりに、パスがエラーになり、そのI/Oエラーがエスカレートしてクラスタリソースのフェールオーバーを引き起こします。

ただし、queue_if_no_pathを有効にすると、パスが回復しない限り、I/Oがいつまでもキューに留まることになるので、**multipathd**が実行中であり、シナリオに有効なことを必ず確認してください。確認しておかないと、再起動するまで、またはキューの代わりに手動でフェールオーバーに戻すまで、影響を受けたマルチパスデバイスでI/Oが無限に停止する可能性があります。

シナリオをテストするには:

1. 端末コンソールを開きます。
2. 次のように入力して、デバイスI/Oに関して、フェールオーバーの代わりに待ち行列処理をアクティブにします。

```
tux > sudo dmsetup message DEVICE_ID 0 queue_if_no_path
```

DEVICE_ID を実際のデバイスのIDに置き換えます。値0はセクタを表し、セクタ情報が必要でないときに使用されます。

たとえば、次のように入力します。

```
tux > sudo dmsetup message 3600601607cf30e00184589a37a31d911 0 queue_if_no_path
```

3. 次のように入力して、デバイスI/Oのフェールオーバーに戻ります。

```
tux > sudo dmsetup message DEVICE_ID 0 fail_if_no_path
```

このコマンドにより、ただちに、待ち行列に入ったすべてのI/Oがエラーになります。

`DEVICE_ID` を実際のデバイスのIDに置き換えます。例えば、次のように入力して、すべてのフォントについてアンチエイリアスを無効にします。

```
tux > sudo dmsetup message 3600601607cf30e00184589a37a31d911 0 fail_if_no_path
```

待ち行列内のI/Oをすべてのパスがエラーになるシナリオ用に設定するには:

1. 端末コンソールを開きます。
2. `/etc/multipath.conf` ファイルをテキストエディタで開きます。
3. `defaults` セクションとその閉じ括弧を非コメント化した後、次のように `default_features` 設定を追加します。

```
defaults {  
    default_features "1 queue_if_no_path"  
}
```

4. `/etc/multipath.conf` ファイルの変更後、`dracut -f` を実行してシステム上に `initrd` を再作成してから、再起動して変更内容を有効にします。
5. デバイスI/Oのフェールオーバーに戻る準備ができれば、次のように入力します。

```
tux > sudo dmsetup message MAPNAME 0 fail_if_no_path
```

`MAPNAME` を該当デバイスのマップされたエイリアス名またはデバイスIDに置き換えます。値0はセクタを表し、セクタ情報が必要でないときに使用されます。このコマンドにより、待ち行列で待機中のすべてのI/Oがエラーとなり、エラーが呼び出し側アプリケーションにプロパゲートします。

17.14.5 停止したI/Oの解決

すべてパスが同時にエラーとなり、I/Oが待ち行列に入って停止している場合は、次のプロセスを実行します。

1. 端末コンソールのプロンプトで、次のコマンドを入力します。

```
tux > sudo dmsetup message MAPNAME 0 fail_if_no_path
```

`MAPNAME` をデバイスの正しいデバイスIDまたはマップされたエイリアス名で置き換えます。値0はセクタを表し、セクタ情報が必要でないときに使用されます。このコマンドにより、待ち行列で待機中のすべてのI/Oがエラーとなり、エラーが呼び出し側アプリケーションにプロパゲートします。

2. 次のコマンドを入力して、待ち行列を再びアクティブにします。

```
tux > sudo dmsetup message MAPNAME 0 queue_if_no_path
```

17.14.6 IBM Zデバイスのデフォルト設定

IBM Zデバイスのマルチパス処理に関するテストを実施した結果、`dev_loss_tmo` パラメータを90秒に、`fast_io_fail_tmo` パラメータを5秒に設定する必要があることわかりました。IBM Zデバイスを使用している場合は、`/etc/multipath.conf` ファイルを変更して、値を次のように指定します。

```
defaults {
    dev_loss_tmo 90
    fast_io_fail_tmo 5
}
```

`dev_loss_tmo` パラメータは、マルチパスリンクに不良のマーキングがされるまでの秒数を設定します。パスに障害が発生したら、そのパスの現在のI/Oが失敗します。デフォルト値は使用するデバイスドライバによって異なります。0～600秒の値の範囲が有効です。ドライバの内部タイムアウトを使用するには、ゼロ(0)または600より大きい値に設定します。

`fast_io_fail_tmo` パラメータは、リンク障害を検出した場合に、I/Oが失敗するまでの待機時間を設定します。ドライバに到達したI/Oは失敗します。ブロックしたキューにI/Oがある場合は、I/Oは`dev_loss_tmo` で指定された時間が経過するまでは失敗せず、キューのブロックが解除されます。

`/etc/multipath.conf` ファイルを変更した場合、その変更内容は、マルチパスマップを更新するまで、または`multipathd` デモンを再起動(`systemctl restart multipathd`)するまで適用されません。

17.14.7 NetAppデバイスでのマルチパスの使用

NetAppデバイスでマルチパスを使用する場合は、`/etc/multipath.conf` ファイルで次の設定を行うことを推奨します。

- NetAppデバイスに対してグローバルに、次のパラメータにデフォルト値を設定する。

```
max_fds max
queue_without_daemon no
```

- ハードウェアテーブル内で、NetAppデバイスに対する次のパラメータにデフォルト値を設定する。

```
dev_loss_tmo infinity
fast_io_fail_tmo 5
features "3 queue_if_no_path pg_init_retries 50"
```

17.14.8 マルチパスデバイスでの--noflushの使用

マルチパスデバイス上で実行する場合は、オプション `--noflush` を必ず使用する必要があります。

たとえば、テーブルのリロードを行うスクリプトでは、マルチパストポロジ情報が必要なので、再開時に `--noflush` オプションを使用して、残っているI/Oがフラッシュされないようにします。

```
load
resume --noflush
```

17.14.9 ルートデバイスがマルチパスの場合のSANタイムアウト設定

マルチパスデバイスにルート(/)があるシステムは、すべてのパスに障害が発生し、それらのパスがシステムから削除されると、停止することがあります。これは、ストレージサブシステム(ファイバチャネルストレージアレイなど)から `dev_loss_tmo` タイムアウトを受信するからです。

システムデバイスがマルチパスを使用して設定され、マルチパスの `no_path_retry` 設定がアクティブな場合は、ストレージサブシステムの `dev_loss_tmo` 設定を適宜変更して、すべてのパスがダウンするシナリオでデバイスが削除されないようにする必要があります。 `dev_loss_tmo` 値をマルチパスの `no_path_retry` 設定以上にすることを強くお勧めします。

ストレージサブシステムの `dev_loss_tmo` の推奨設定は、次のとおりです。

```
<dev_loss_tmo> = <no_path_retry> * <polling_interval>
```

マルチパス値については、次の定義が適用されます。

- `no_path_retry` は、パスが失われたとみなされて入出力のキューイングが停止されるまでのマルチパス入出力の再試行数です。
- `polling_interval` は、パッチチェックの間隔(秒単位)です。

これらの各マルチパス値は、`/etc/multipath.conf` 環境設定ファイルから設定する必要があります。詳細については、17.6項「`/etc/multipath.conf` Fileの作成または修正」を参照してください。

17.15 MPIOのトラブルシューティング

本項では、MPIOに関するいくつかの既知の問題と、考えられる解決手段について説明します。

17.15.1 マルチパスデバイスへのGRUB2のインストール

Btrfsを使用したレガシBIOSシステムでは、許可がないため **grub2-install** が失敗する可能性があります。これを修正するには、`/boot/grub2/SUBDIR/` サブボリュームが読み書き(rw)モードでマウントされるようにしてください。`SUBDIR` は `x86_64-efi` または `i386-pc` にできます。

17.15.2 マルチパスが有効な場合、ブート時にシステムが終了して緊急シェルが起動する

ブート中にシステムが終了して緊急シェルが起動し、次のようなメッセージが表示されます。

```
[ OK ] Listening on multipathd control socket.
        Starting Device-Mapper Multipath Device Controller...
[ OK ] Listening on Device-mapper event daemon FIFOs.
        Starting Device-mapper event daemon...
        Expecting device dev-disk-by\x2duuid-34be48b2\x2dc21...32dd9.device...
        Expecting device dev-sda2.device...
[ OK ] Listening on udev Kernel Socket.
[ OK ] Listening on udev Control Socket.
        Starting udev Coldplug all Devices...
        Expecting device dev-disk-by\x2duuid-1172afe0\x2d63c...5d0a7.device...
        Expecting device dev-disk-by\x2duuid-c4a3d1de\x2d4dc...ef77d.device...
[ OK ] Started Create list of required static device nodes ...current kernel.
        Starting Create static device nodes in /dev...
[ OK ] Started Collect Read-Ahead Data.
[ OK ] Started Device-mapper event daemon.
[ OK ] Started udev Coldplug all Devices.
        Starting udev Wait for Complete Device Initialization...
[ OK ] Started Replay Read-Ahead Data.
        Starting Load Kernel Modules...
```

```

Starting Remount Root and Kernel File Systems...
[ OK ] Started Create static devices
[*    ] (1 of 4) A start job is running for dev-disk-by\x2du...(7s / 1min 30s)
[*    ] (1 of 4) A start job is running for dev-disk-by\x2du...(7s / 1min 30s)

...

Timed out waiting for device dev-disk-by\x2duuid-c4a...cfef77d.device.
[DEPEND] Dependency failed for /opt.
[DEPEND] Dependency failed for Local File Systems.
[DEPEND] Dependency failed for Postfix Mail Transport Agent.
Welcome to emergency shell
Give root password for maintenance
(or press Control-D to continue):

```

この問題は次の状況で発生します。

- マルチパスが有効な場合に、非マルチパスルートファイルシステムがブラックリスト化されていない場合。詳細については、[手順17.1「緊急シェル: ファイルシステムのブラックリスト化」](#)を参照してください。
- マルチパスルートファイルシステムを持つシステムで、`initrd`を再構築せずにマルチパスを有効/無効にした場合。詳細については、[手順17.2「緊急シェル: `initrd`の再構築」](#)を参照してください。
- Network Attached Storageとマルチパスが有効なシステムで、`initrd`にネットワークストレージドライバがない場合。

手順 17.1: 緊急シェル: ファイルシステムのブラックリスト化

この修正は、ルートファイルシステムがマルチパス上にないにもかかわらずマルチパスが有効になっている場合に必要です。このようなセットアップの場合、マルチパスはブラックリスト化されていないすべてのデバイスに対してパスを設定しようとし、ルートファイルシステムがあるデバイスは既にマウントされているためマルチパスではアクセスできず、これが失敗の原因になります。この問題を修復するには、`/etc/multipath.conf`でルートデバイスをブラックリスト化して、マルチパスを正しく設定します。

1. 緊急シェルで`multipath -v2`を実行し、ルートファイルシステムのデバイスを特定します。この結果、次のような出力が表示されます。

```

root # multipath -v2
Dec 18 10:10:03 | 3600508b1001030343841423043300400: ignoring map

```

| ~ : の間の文字列が、ブラックリスト化に必要なWWIDです。

マルチパスが有効な場合、ブート時にシステムが終了して緊急シェルが起動する

2. `/etc/multipath.conf`を開いて以下を追加します。

```
blacklist {
    wwid "WWWID"
}
```

`WWWID`は、前の手順で取得したIDに置き換えます。詳細については、[17.8項「非マルチパスデバイスのブラックリスト化」](#)を参照してください。

3. `Ctrl + D>` を押し、緊急シェルを終了してサーバを再起動します。

手順 17.2: 緊急シェル: `initrd`の再構築

この修正は、[マルチパスの状態] (有効または無効)が `initrd` とシステムの間で異なる場合に必要です。修正するには、`initrd` を再構築します。

1. システムでマルチパスが「有効」になっている場合、次のコマンドを使用し、マルチパスサポートを指定して `initrd` を再構築します。

```
tux > dracut --force --add multipath
```

システムでマルチパスが「無効」になっている場合、次のコマンドを使用し、マルチパスサポートを指定して `initrd` を再構築します。

```
tux > dracut --force -o multipath
```

2. `Ctrl + D>` を押し、緊急シェルを終了してサーバを再起動します。

手順 17.3: 緊急シェル: `initrd`の再構築

この修正は、`initrd`にNetwork Attached Storageアクセス用のドライバが含まれていない場合に必要です。たとえば、マルチパスを設定せずにシステムをインストールした場合や、各ハードウェアを追加または交換する場合などが該当します。

1. 必要なドライバをファイル `/etc/dracut.conf.d/01-dist.conf` 内の変数 `force_drivers` に追加します。たとえば、システムに `hpsa` ドライバでアクセスされるRAIDコントローラがあり、`qla23xx` ドライバでアクセスされるQlogicコントローラにマルチパスデバイスが接続されている場合は、次のようなエントリになります。

```
force_drivers+="hpsa qla23xx"
```

2. 次のコマンドを使用して `initrd` を再構築します。

```
tux > dracut -f --add-multipath
```


3. ネットワークストレージの接続に失敗した場合にシステムが緊急モードでブートしないようにするため、`/etc/fstab` の各エントリにマウントオプション `_netdev` を追加することをお勧めします。
4. `Ctrl + D` を押し、緊急シェルを終了してサーバを再起動します。

17.15.3 マルチパス0.4.9以降への更新後に、個別デバイスのprio設定が失敗する

バージョン 0.4.9以降のマルチパスツールでは、`/etc/multipath.conf` ファイルの `defaults{}` セクションまたは `devices{}` セクションの `prio` 設定を使用します。キーワード `prio` が、`multipath{}` セクションの個別の `multipaths` 定義に指定された場合は、暗黙のうちに無視されます。

マルチパスツール0.4.8では、`multipaths{}` セクションの個別の `multipath` 定義内の `prio` 設定で、`defaults{}` または `devices{}` セクションの `prio` 設定を上書きすることができました。

17.15.4 multipath-tools-0.4.9以降への更新後に、引数を伴うprio設定が失敗する

`multipath-tools-0.4.8` から `multipath-tools-0.4.9` に更新すると、引数を必要とする Prioritizer の場合、`/etc/multipath.conf` ファイル内の `prio` 設定が壊れます。`multipath-tools-0.4.9` では、Prioritizer の指定には `prio` キーワードが使われ、引数を必要とする Prioritizer の指定には、`prio_args` キーワードが使われます。これまでは、Prioritizer とその引数はいずれも、同じ `prio` 行で指定していました。

たとえば、`multipath-tools-0.4.8` では、次の行を使用して Prioritizer とその引数を同じ行で指定していました。

```
prio "weightedpath hctl [1,3]:...+:.+ 260 [0,2]:...+:.+ 20"
```

`multipath-tools-0.4.9` 以降への更新後は、このコマンドを使用するとエラーになります。メッセージの例を以下に示します。

```
<Month day hh:mm:ss> | Prioritizer 'weightedpath hctl [1,3]:...+:.+ 260  
[0,2]:...+:.+ 20' not found in /lib64/multipath
```

この問題を解決するには、テキストエディタで、`/etc/multipath.conf` ファイル内の `prio` 行を変更します。2つの行を作成して、`prio` 行に Prioritizer を指定し、その下の `prio_args` 行に Prioritizer の引数を指定します。

```
prio "weightedpath"  
prio_args "hctl [1,3]:...+...+ 260 [0,2]:...+...+ 20"
```

`sudo systemctl restart multipathd`を実行して **multipathd** デーモンを再起動し、変更を有効にします。

17.15.5 技術情報ドキュメント

SUSE Linux Enterprise ServerのマルチパスI/Oの問題のトラブルシューティングについては、SUSEナレッジベースにある、次のTID (技術情報ドキュメント)を参照してください。

- Using LVM on local and SAN attached devices (<http://www.suse.com/support/kb/doc.php?id=3617600>) ↗
- Using LVM on Multipath (DM MPIO) Devices (<http://www.suse.com/support/kb/doc.php?id=7007498>) ↗
- HOWTO: Add, Resize and Remove LUN without restarting SLES (<https://www.suse.com/support/kb/doc.php?id=7009660>) ↗

18 NFSv4上でのアクセス制御リストの管理

Linuxには、ユーザ、グループ、およびその他(ugo)に対する簡単な読み込み、書き込み、および実行(rwx)の各フラグ以上の、ACL (アクセス制御リスト)の単一標準はありません。よりきめ細かな制御のオプションの1つにDraft POSIX ACLがあります。ただし、これらのACLは、POSIXによって正式に標準化されたことはありません。もう1つは、NFSv4ネットワークファイルシステムの一部として設計されたNFSv4 ACLです。NFSv4 ACLは、Linux上のPOSIXシステムとMicrosoft Windows上のWIN32システム間に適切な互換性を提供することを目的としています。

NFSv4 ACLは、Draft POSIX ACLを正しく実装できるほど十分ではないので、NFSv4クライアントへのACLアクセスのマッピングは試みられていません(**setfacl**の使用など)。


NFSv4の使用時は、Draft POSIX ACLはエミュレーションでさえ使用できず、NFSv4 ACLを直接使用する必要があります。つまり、**setfacl**をNFSv3で動作させながら、NFSv4で動作させることはできません。NFSv4 ACLをNFSv4ファイルシステムで使用できるようにするため、SUSE Linux Enterprise Serverでは、次のファイルを含む nfs4-acl-tools パッケージを提供しています。

- nfs4-getfacl
- nfs4-setfacl
- nfs4-editacl

これらの動作は、NFSv4 ACLを検証および変更する **getfacl** および **setfacl** とほぼ同様です。これらのコマンドは、NFSサーバ上のファイルシステムがNFSv4 ACLを完全にサポートしている場合にのみ有効です。サーバによって課される制限は、クライアントで実行されているこれらのプログラムに影響を与え、ACE (Access Control Entries)の一部の特定の組み合わせが不可能なことがあります。

エクスポート元のNFSサーバにNFSボリュームをローカルにマウントすることはサポートされていません。

追加情報

詳細については、Introduction to NFSv4 ACLs (http://wiki.linux-nfs.org/wiki/index.php/ACLs#Introduction_to_NFSv4_ACLs )を参照してください。

A GNU利用許諾契約書

この付録には、GNU Free Documentation Licenseバージョン1.2が含まれています。

GNU Free Documentation License

Copyright (C) 2000, 2001, 2002 Free Software Foundation, Inc. 51 Franklin St, Fifth Floor, Boston, MA 02110-1301 USA. Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

0. PREAMBLE

The purpose of this License is to make a manual, textbook, or other functional and useful document "free" in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or non-commercially. Secondly, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of "copyleft", which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

1. APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work, in any medium, that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. Such a notice grants a world-wide, royalty-free license, unlimited in duration, to use that work under the conditions stated herein. The "Document", below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as "you". You accept the license if you copy, modify or distribute the work in a way requiring permission under copyright law.

A "Modified Version" of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A "Secondary Section" is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document's overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (Thus, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The "Invariant Sections" are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License. If a section does not fit the above definition of Secondary then it is not allowed to be designated as Invariant. The Document may contain zero Invariant Sections. If the Document does not identify any Invariant Sections then there are none.

The "Cover Texts" are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License. A Front-Cover Text may be at most 5 words, and a Back-Cover Text may be at most 25 words.

A "Transparent" copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, that is suitable for revising the document straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format

whose markup, or absence of markup, has been arranged to thwart or discourage subsequent modification by readers is not Transparent. An image format is not Transparent if used for any substantial amount of text. A copy that is not "Transparent" is called "Opaque".

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML, PostScript or PDF designed for human modification. Examples of transparent image formats include PNG, XCF and JPG. Opaque formats include proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML, PostScript or PDF produced by some word processors for output purposes only.

The "Title Page" means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, "Title Page" means the text near the most prominent appearance of the work's title, preceding the beginning of the body of the text.

A section "Entitled XYZ" means a named subunit of the Document whose title either is precisely XYZ or contains XYZ in parentheses following text that translates XYZ in another language. (Here XYZ stands for a specific section name mentioned below, such as "Acknowledgements", "Dedications", "Endorsements", or "History".) To "Preserve the Title" of such a section when you modify the Document means that it remains a section "Entitled XYZ" according to this definition.

The Document may include Warranty Disclaimers next to the notice which states that this License applies to the Document. These Warranty Disclaimers are considered to be included by reference in this License, but only as regards disclaiming warranties: any other implication that these Warranty Disclaimers may have is void and has no effect on the meaning of this License.

2. VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or non-commercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

3. COPYING IN QUANTITY

If you publish printed copies (or copies in media that commonly have printed covers) of the Document, numbering more than 100, and the Document's license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a computer-network location from which the general network-using public has access to download using public-standard network protocols a complete Transparent copy of the Document, free of added material. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in

quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

4. MODIFICATIONS

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

- A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.
- B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has fewer than five), unless they release you from this requirement.
- C. State on the Title page the name of the publisher of the Modified Version, as the publisher.
- D. Preserve all the copyright notices of the Document.
- E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.
- F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.
- G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.
- H. Include an unaltered copy of this License.
- I. Preserve the section Entitled "History"; Preserve its Title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section Entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.
- J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.
- K. For any section Entitled "Acknowledgements" or "Dedications", Preserve the Title of the section, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.
- L. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.
- M. Delete any section Entitled "Endorsements". Such a section may not be included in the Modified Version.
- N. Do not retitle any existing section to be Entitled "Endorsements" or to conflict in title with any Invariant Section.
- O. Preserve any Warranty Disclaimers.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections

as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles.

You may add a section Entitled "Endorsements", provided it contains nothing but endorsements of your Modified Version by various parties—for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

5. COMBINING DOCUMENTS

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice, and that you preserve all their Warranty Disclaimers.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections Entitled "History" in the various original documents, forming one section Entitled "History"; likewise combine any sections Entitled "Acknowledgements", and any sections Entitled "Dedications". You must delete all sections Entitled "Endorsements".

6. COLLECTIONS OF DOCUMENTS

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

7. AGGREGATION WITH INDEPENDENT WORKS

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, is called an "aggregate" if the copyright resulting from the compilation is not used to limit the legal rights of the compilation's users beyond what the individual works permit. When the Document is included in an aggregate, this License does not apply to the other works in the aggregate which are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one half of the entire aggregate, the Document's Cover Texts may be placed on covers that bracket the Document within the aggregate, or the electronic equivalent of covers if the Document is in electronic form. Otherwise they must appear on printed covers that bracket the whole aggregate.

8. TRANSLATION

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License, and all the license notices in the Document, and any Warranty Disclaimers, provided that you also include the original English version of this License and the original versions of those notices and disclaimers. In case of a disagreement between the translation and the original version of this License or a notice or disclaimer, the original version will prevail.

If a section in the Document is Entitled "Acknowledgements", "Dedications", or "History", the requirement (section 4) to Preserve its Title (section 1) will typically require changing the actual title.

9. TERMINATION

You may not copy, modify, sublicense, or distribute the Document except as expressly provided for under this License. Any other attempt to copy, modify, sublicense or distribute the Document is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

10. FUTURE REVISIONS OF THIS LICENSE

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See <http://www.gnu.org/copyleft/>.

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License "or any later version" applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation.

ADDENDUM: How to use this License for your documents

```
Copyright (c) YEAR YOUR NAME.
Permission is granted to copy, distribute and/or modify this document
under the terms of the GNU Free Documentation License, Version 1.2
or any later version published by the Free Software Foundation;
with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts.
A copy of the license is included in the section entitled "GNU
Free Documentation License".
```

If you have Invariant Sections, Front-Cover Texts and Back-Cover Texts, replace the "with...Texts." line with this:

```
with the Invariant Sections being LIST THEIR TITLES, with the
Front-Cover Texts being LIST, and with the Back-Cover Texts being LIST.
```

If you have Invariant Sections without Cover Texts, or some other combination of the three, merge those two alternatives to suit the situation.

If your document contains nontrivial examples of program code, we recommend releasing these examples in parallel under your choice of free software license, such as the GNU General Public License, to permit their use in free software.