



Solaris ボリュームマネージャの管理

Sun Microsystems, Inc.
4150 Network Circle
Santa Clara, CA 95054
U.S.A.

Part No: 817-0159-10
2002 年 12 月

Copyright 2002 Sun Microsystems, Inc. 4150 Network Circle, Santa Clara, CA 95054 U.S.A. All rights reserved.

本製品およびそれに関連する文書は著作権法により保護されており、その使用、複製、頒布および逆コンパイルを制限するライセンスのもとにおいて頒布されます。サン・マイクロシステムズ株式会社の書面による事前の許可なく、本製品および関連する文書のいかなる部分も、いかなる方法によっても複製することが禁じられます。

本製品の一部は、カリフォルニア大学からライセンスされている Berkeley BSD システムに基づいていることがあります。UNIX は、X/Open Company, Ltd. が独占的にライセンスしている米国ならびに他の国における登録商標です。フォント技術を含む第三者のソフトウェアは、著作権により保護されており、提供者からライセンスを受けているものです。

Federal Acquisitions: Commercial Software-Government Users Subject to Standard License Terms and Conditions.

本製品に含まれる HG 明朝 L、HG-MincyoL-Sun、HG ゴシック B、および HG-GothicB-Sun は、株式会社リコーがリコービマジクス株式会社からライセンス供与されたタイプフェイスマスタをもとに作成されたものです。HG 平成明朝体 W3@X12 は、株式会社リコーが財団法人日本規格協会からライセンス供与されたタイプフェイスマスタをもとに作成されたものです。フォントとして無断複製することは禁止されています。

Sun、Sun Microsystems、docs.sun.com、AnswerBook、AnswerBook2 は、米国およびその他の国における米国 Sun Microsystems, Inc. (以下、米国 Sun Microsystems 社とします) の商標もしくは登録商標です。

サンのロゴマークおよび Solaris は、米国 Sun Microsystems 社の登録商標です。

すべての SPARC 商標は、米国 SPARC International, Inc. のライセンスを受けて使用している同社の米国およびその他の国における商標または登録商標です。SPARC 商標が付いた製品は、米国 Sun Microsystems 社が開発したアーキテクチャに基づくものです。

OPENLOOK、OpenBoot、JLE は、サン・マイクロシステムズ株式会社の登録商標です。

Wnn は、京都大学、株式会社アステック、オムロン株式会社で共同開発されたソフトウェアです。

Wnn6 は、オムロン株式会社、オムロンソフトウェア株式会社で共同開発されたソフトウェアです。© Copyright OMRON Co., Ltd. 1995-2000. All Rights Reserved. © Copyright OMRON SOFTWARE Co., Ltd. 1995-2002 All Rights Reserved.

「ATOK」は、株式会社ジャストシステムの登録商標です。

「ATOK Server/ATOK12」は、株式会社ジャストシステムの著作物であり、「ATOK Server/ATOK12」にかかる著作権その他の権利は、株式会社ジャストシステムおよび各権利者に帰属します。

本製品に含まれる郵便番号辞書 (7 桁/5 桁) は郵政事業庁が公開したデータを元に制作された物です (一部データの加工を行なっています)。

本製品に含まれるフェイスマーク辞書は、株式会社ビレッジセンターの許諾のもと、同社が発行する『インターネット・パソコン通信フェイスマークガイド '98』に添付のものを使用しています。© 1997 ビレッジセンター

Unicode は、Unicode, Inc. の商標です。

本書で参照されている製品やサービスに関しては、該当する会社または組織に直接お問い合わせください。

OPEN LOOK および Sun Graphical User Interface は、米国 Sun Microsystems 社が自社のユーザおよびライセンス実施権者向けに開発しました。米国 Sun Microsystems 社は、コンピュータ産業用のビジュアルまたはグラフィカル・ユーザインタフェースの概念の研究開発における米国 Xerox 社の先駆者としての成果を認めるものです。米国 Sun Microsystems 社は米国 Xerox 社から Xerox Graphical User Interface の非独占的ライセンスを取得しており、このライセンスは米国 Sun Microsystems 社のライセンス実施権者にも適用されます。

DtComboBox ウィジェットと DtSpinBox ウィジェットのプログラムおよびドキュメントは、Interleaf, Inc. から提供されたものです。(© 1993 Interleaf, Inc.)

本書は、「現状のまま」をベースとして提供され、商品性、特定目的への適合性または第三者の権利の非侵害の黙示の保証を含みそれに限定されない、明示的であるか黙示的であるかを問わない、なんらの保証も行われぬものとします。

本製品が、外国為替および外国貿易管理法 (外為法) に定められる戦略物資等 (貨物または役務) に該当する場合、本製品を輸出または日本国外へ持ち出す際には、サン・マイクロシステムズ株式会社の事前の書面による承諾を得ることのほか、外為法および関連法規に基づく輸出手続き、また場合によっては、米国商務省または米国所轄官庁の許可を得ることが必要です。

原典: Solaris Volume Manager Administration Guide

Part No: 816-4518-10

Revision A



021118@4879



目次

はじめに	17
1 Solaris ボリュームマネージャの概要	23
Solaris ボリュームマネージャで行う作業の概要	24
Solaris ボリュームマネージャのロードマップ — 記憶容量	24
Solaris ボリュームマネージャのロードマップ — 可用性	25
Solaris ボリュームマネージャのロードマップ — 入出力性能	26
Solaris ボリュームマネージャのロードマップ — 管理	26
Solaris ボリュームマネージャのロードマップ — 障害追跡	27
2 記憶装置管理の概念	29
記憶装置管理の紹介	29
記憶装置ハードウェア	29
RAID レベル	30
構成計画の指針	31
記憶方式の選択	31
性能について	33
性能に関する一般的な指針	33
ランダム入出力と順次入出力の最適化	34
ランダム入出力	34
順次アクセス入出力	35
3 Solaris ボリュームマネージャの概要	37
Solaris ボリュームマネージャの機能	37
Solaris ボリュームマネージャによる記憶装置の管理方法	38

	Solaris ボリュームマネージャとの対話方法	38
	▼ Solaris ボリュームマネージャグラフィカルユーザーインターフェースを使用するには	40
	Solaris ボリュームマネージャの前提条件	41
	Solaris ボリュームマネージャコンポーネントの概要	41
	ボリューム	42
	状態データベースと状態データベースの複製	46
	ホットスペア集合	47
	ディスクセット	48
	Solaris ボリュームマネージャ構成の指針	48
	一般的な指針	48
	ファイルシステムに関する指針	49
	Solaris ボリュームマネージャ要素の作成	49
	Solaris ボリュームマネージャ要素を作成するための前提条件	49
4	Solaris ボリュームマネージャの構成と使用	51
	背景情報	51
	ハードウェア構成	51
	記憶領域の構成	52
	Solaris ボリュームマネージャ構成の詳細	52
5	状態データベース (概要)	55
	Solaris ボリュームマネージャの状態データベースと状態データベースの複製について	55
	多数決アルゴリズムとは	57
	状態データベースの複製を定義するための背景情報	58
	状態データベースの複製に関する推奨事項	58
	状態データベースの複製に関する指針	59
	状態データベースの複製のエラー処理	59
	シナリオ — 状態データベースの複製	60
6	状態データベース (作業)	63
	状態データベースの複製 (作業マップ)	63
	状態データベースの複製の作成	64
	▼ 状態データベースの複製を作成するには	64
	状態データベースの複製の保守	66
	▼ 状態データベースの複製の状態をチェックするには	66

状態データベースの複製を削除するには 67

7 RAID 0 (ストライプ方式および連結方式) ボリューム (概要)	69
RAID 0 ボリュームの概要	69
RAID 0 (ストライプ方式) ボリューム	70
RAID 0 (連結方式) ボリューム	72
RAID 0 (ストライプ方式の連結) ボリューム	74
RAID 0 ボリュームを作成するための背景情報	77
ストライプ方式と連結方式の前提条件	77
ストライプ方式および連結方式の指針	77
シナリオ — RAID 0 ボリューム	78
8 RAID 0 (ストライプ方式および連結方式) ボリューム (作業)	79
RAID 0 ボリューム (作業マップ)	79
RAID 0 (ストライプ方式) ボリュームの作成	80
▼ RAID 0 (ストライプ方式) ボリュームを作成するには	80
RAID 0 (連結方式) ボリューム	81
▼ RAID 0 (連結方式) ボリュームを作成するには	81
記憶領域の拡張	82
▼ 既存のデータの領域を拡張するには	83
▼ 既存の RAID 0 ボリュームを拡張するには	84
ボリュームの削除	86
▼ ボリュームを削除するには	86
9 RAID 1 (ミラー) ボリューム (概要)	87
RAID 1 (ミラー) ボリュームの概要	87
サブミラーの概要	88
シナリオ — RAID 1 (ミラー) ボリューム	88
RAID 1+0 と RAID 0+1 の提供	89
RAID 1 (ミラー) ボリュームに関する FAQ (よくある質問)	90
RAID 1 ボリュームの構成指針	91
RAID 1 ボリュームのオプション	92
RAID 1 ボリューム (ミラー) の再同期	93
ミラー全体の再同期	94
再同期の最適化	94
部分的な再同期	94
パス番号	94

RAID 1 ボリュームの背景情報	95
RAID 1 ボリュームを作成するための背景情報	95
RAID 1 ボリュームオプションを変更するための背景情報	96
シングルユーザーモードでの起動が RAID 1 ボリュームに与える影響	96
シナリオ — RAID 1 ボリューム (ミラー)	96

10 RAID 1 (ミラー) ボリューム (作業) 99

RAID 1 ボリューム (作業マップ)	99
RAID 1 ボリュームの作成	101
▼ 未使用のスライスから RAID 1 ボリュームを作成するには	101
▼ ファイルシステムから RAID 1 ボリュームを作成するには	103
ルート (/) のミラー化に関する特殊な考慮事項	108
起動時の警告	108
代替起動デバイスへのパスを記録するには	108
代替起動デバイスからの起動	110
サブミラーに関する作業	110
▼ サブミラーを接続するには	110
▼ サブミラーを切り離すには	111
▼ サブミラーをオフラインまたはオンラインにするには	112
▼ サブミラー内のスライスを有効にするには	113
RAID 1 ボリュームの保守	114
▼ ミラーとサブミラーの状態をチェックするには	116
▼ RAID 1 ボリュームオプションを変更するには	117
▼ RAID 1 ボリュームを拡張するには	119
RAID 1 ボリュームのコンポーネント障害に対する処置	120
▼ サブミラー内のスライスを交換するには	120
▼ サブミラーを交換するには	121
RAID 1 ボリュームの削除 (ミラー化の解除)	122
▼ ファイルシステムのミラー化を解除するには	122
▼ マウント解除できないファイルシステムのミラー化を解除するには	124
ミラーを使ったデータのバックアップ	126
▼ RAID 1 ボリュームを使ってオンラインバックアップをとるには	126

11 ソフトパーティション (概要) 129

ソフトパーティションの概要	129
ソフトパーティションの背景情報	130
ソフトパーティションの要件	130

	ソフトパーティションの指針	130
	シナリオ — ソフトパーティション	131
12	ソフトパーティション (作業)	133
	ソフトパーティション (作業マップ)	133
	ソフトパーティションの作成	134
	▼ ソフトパーティションを作成するには	134
	ソフトパーティションの保守	135
	▼ ソフトパーティションの状態をチェックするには	135
	▼ ソフトパーティションを拡張するには	136
	ソフトパーティションを削除するには	137
13	RAID 5 ボリューム (概要)	139
	RAID 5 ボリュームの概要	139
	例 — RAID 5 ボリューム	140
	例 — RAID 5 ボリュームの連結 (拡張)	141
	RAID 5 ボリュームを作成するための背景情報	143
	RAID 5 ボリュームの要件	143
	RAID 5 ボリュームの指針	144
	RAID 5 ボリューム内のスライスの置き換えと有効化 (概要)	145
	シナリオ — RAID 5 ボリューム	145
14	RAID 5 ボリューム (作業)	147
	RAID 5 ボリューム (作業マップ)	147
	RAID 5 ボリュームの作成	148
	▼ RAID 5 ボリュームを作成するには	148
	RAID 5 ボリュームの保守	149
	▼ RAID 5 ボリュームの状態をチェックするには	149
	▼ RAID 5 ボリュームを拡張するには	152
	▼ RAID 5 ボリューム内のコンポーネントを有効にするには	153
	▼ RAID 5 ボリューム内のコンポーネントを置き換えるには	154
15	ホットスペア集合 (概要)	157
	ホットスペア集合とホットスペアの概要	157
	ホットスペア	158
	ホットスペアの仕組み	158

	ホットスペア集合	159
	例 — ホットスペア集合	159
	ホットスペア集合の管理	160
	シナリオ — ホットスペア	161
16	ホットスペア集合 (作業)	163
	ホットスペア集合 (作業マップ)	163
	ホットスペア集合の作成	164
	▼ ホットスペア集合を作成するには	164
	▼ ホットスペア集合にホットスペアを追加するには	165
	ホットスペア集合とボリュームの対応付け	166
	▼ ホットスペア集合とボリュームを対応付けるには	166
	▼ ホットスペア集合の対応付けを変更するには	168
	ホットスペア集合の保守	169
	▼ ホットスペア集合とホットスペアの状態を確認するには	169
	▼ ホットスペア集合内のホットスペアを置き換えるには	171
	▼ ホットスペア集合からホットスペアを削除するには	172
	▼ ホットスペアを有効にするには	173
17	トランザクションボリューム (概要)	175
	ファイルシステムロギングについて	175
	ロギング方法の選択	176
	トランザクションボリューム	177
	例 — トランザクションボリューム	177
	例 — ログデバイスの共有	178
	トランザクションボリュームの背景情報	179
	トランザクションボリューム用の要件	179
	トランザクションボリュームの指針	180
	シナリオ — トランザクションボリューム	181
18	トランザクションボリューム (作業)	183
	トランザクションボリューム (作業マップ)	183
	トランザクションボリュームの作成	185
	▼ トランザクションボリュームを作成するには	185
	トランザクションボリュームを UFS ロギングに変換	188
	トランザクションボリュームを UFS ロギングに変換するには	189
	トランザクションボリュームの保守	192

	▼トランザクションボリュームの状態をチェックするには	192
	▼トランザクションボリュームにログデバイスを接続するには	195
	▼トランザクションボリュームからログデバイスを切断するには	196
	▼トランザクションボリュームを拡張するには	197
	▼トランザクションボリュームを削除するには	198
	▼トランザクションボリュームを削除した状態でマウントデバイスを保持するには	200
	ログデバイスの共有	202
	▼ファイルシステム間でログデバイスを共有するには	202
	エラー発生時のトランザクションボリュームの回復	204
	▼パニックした場合のトランザクションボリュームを回復するには	204
	▼ハードウェアエラー状態のトランザクションボリュームを回復するには	204
19	ディスクセット (概要)	207
	ディスクセットの機能	207
	Solaris ボリュームマネージャによるディスクセットの管理	208
	ディスクの自動パーティション分割	209
	ディスクセットの命名規則	211
	例 — 2つの共有ディスクセット	211
	ディスクセットの背景情報	212
	ディスクセットの要件	212
	ディスクセットの指針	213
	ディスクセットの管理	213
	ディスクセットの取得	214
	ディスクセットの解放	214
	シナリオ — ディスクセット	215
20	ディスクセット (作業)	217
	ディスクセット (作業マップ)	217
	ディスクセットの作成	218
	▼ディスクセットを作成するには	218
	ディスクセットの拡張	219
	▼ディスクセットにドライブを追加するには	219
	▼ディスクセットにホストを追加するには	221
	▼ディスクセットに Solaris ボリュームマネージャのコンポーネントを作成するには	222
	ディスクセットの保守	223

▼ ディスクセットの状態をチェックするには	223
▼ ディスクセットからディスクを削除するには	224
▼ ディスクセットを取得するには	225
▼ ディスクセットを解放するには	227
▼ ホストまたはディスクセットを削除するには	228
21 Solaris ボリュームマネージャの保守 (作業)	231
Solaris ボリュームマネージャの保守 (作業マップ)	231
Solaris ボリュームマネージャ構成の表示	232
▼ Solaris ボリュームマネージャのボリューム構成を表示するには	232
ボリューム名の変更	235
ボリューム名を変更するための背景情報	235
ボリューム名の交換	236
▼ ボリューム名を変更するには	237
構成ファイルの使用	238
▼ 構成ファイルを作成するには	238
▼ 構成ファイルを使って Solaris ボリュームマネージャを初期化するには	238
Solaris ボリュームマネージャのデフォルト値の変更	240
▼ デフォルトのボリューム数を増やすには	240
デフォルトのディスクセット数を増やすには	241
ファイルシステムの拡張	242
スライスやボリュームを拡張するための背景情報	243
▼ ファイルシステムを拡張するには	243
RAID 1 および RAID 5 ボリューム内のコンポーネントの交換と有効化の概要	244
コンポーネントの有効化	245
コンポーネントを他の使用可能なコンポーネントで置き換える	245
「保守 (Maintenance)」状態と「最後にエラー (Last Erred)」状態	246
ミラーおよび RAID 5 ボリューム内のスライスを交換または有効にするための背景情報	247
22 Solaris ボリュームマネージャで構築可能な最善の記憶装置構成	249
小規模なサーバーを運用する場合の構成例	249
ネットワーク接続された記憶装置に対して Solaris ボリュームマネージャを使用する場合の構成例	251
23 監視とエラーレポート (作業)	253
Solaris ボリュームマネージャの監視機能と報告機能 (作業マップ)	254

	エラーを周期的にチェックするための mdmonitord デーモンの設定	254
	▼ mdmonitord コマンドを設定してエラーを周期的にチェックするには	255
	Solaris ボリュームマネージャ SNMP エージェントの概要	255
	Solaris ボリュームマネージャ SNMP エージェントの構成	256
	▼ Solaris ボリュームマネージャ SNMP エージェントを構成するには	256
	Solaris ボリュームマネージャ SNMP エージェントの制約	258
	cron ジョブによる Solaris ボリュームマネージャの監視	259
	▼ ボリュームのエラーを自動的にチェックするには	259
24	Solaris ボリュームマネージャの障害追跡	269
	Solaris ボリュームマネージャ(作業マップ)	269
	障害追跡の概要	270
	障害追跡の前提条件	270
	Solaris ボリュームマネージャによる障害追跡の一般的な指針	270
	一般的な障害追跡方法	271
	ディスクの交換	271
	▼ 不良ディスクを交換するには	272
	ディスク移動の問題からの回復	274
	▼ ディスク移動とデバイス ID の概要	274
	名前のないデバイスに関するエラーメッセージを解決するには	274
	起動障害	275
	起動障害の背景情報	276
	▼ /etc/vfstab 内の不適切なエントリを修正するには	276
	例 — ルート (/) ミラーを回復する	277
	▼ 起動デバイスの障害から回復するには	278
	状態データベースの複製の障害からの回復	282
	▼ 状態データベースの複製数の不足から回復するには	282
	トランザクションボリュームの修復	285
	パニック	286
	トランザクションボリュームのエラー	286
	ソフトパーティション障害からの回復	286
	ソフトパーティションの構成データを復元するには	286
	別のシステムから構成を復元	289
	構成の復元	289
A	Solaris ボリュームマネージャの重要なファイル	295
	システムファイルと始動ファイル	295

手動で設定するファイル 297
md.tab ファイルの概要 297

B Solaris ボリュームマネージャのコマンド行リファレンス 299
コマンド行リファレンス 299

C Solaris ボリュームマネージャの CIM/WBEM API 301
Solaris ボリュームマネージャの管理 301

索引 303

表目次

表 1-1	Solaris ボリュームマネージャのロードマップ — 記憶容量	24
表 1-2	Solaris ボリュームマネージャのロードマップ — 可用性	25
表 1-3	Solaris ボリュームマネージャのロードマップ — 入出力性能	26
表 1-4	Solaris ボリュームマネージャのロードマップ — 管理	26
表 1-5	Solaris ボリュームマネージャのロードマップ — 障害追跡	27
表 2-1	記憶方式の選択	31
表 2-2	冗長性の最適化	32
表 3-1	Solaris ボリュームマネージャコンポーネントの要約	42
表 3-2	ボリュームクラス	43
表 3-3	ボリューム名の例	46
表 9-1	ミラーの読み取りポリシー	92
表 9-2	ミラーの書き込みポリシー	93
表 10-1	サブミラーの状態	114
表 10-2	サブミラーのスライスの状態	115
表 14-1	RAID 5 の状態	150
表 14-2	RAID 5 のスライスの状態	151
表 16-1	ホットスペア集合の状態 (コマンド行)	170
表 18-1	トランザクションボリュームの状態	194
表 19-1	ボリューム名の例	211
表 24-1	Solaris ボリュームマネージャの一般的な起動障害	275
表 B-1	Solaris ボリュームマネージャのコマンド	299

目次

図 3-1	Solaris 管理コンソール内の「拡張ディスク (Enhanced Storage)」ツール (Solaris ボリュームマネージャ) の使用例	39
図 3-2	ボリューム、物理ディスク、スライスの関係	44
図 4-1	基本的なハードウェア構成	52
図 7-1	ストライプ方式の例	72
図 7-2	連結の例	73
図 7-3	ストライプ方式の連結の例	75
図 9-1	ミラーの例	88
図 9-2	RAID 1+0 の例	89
図 13-1	RAID 5 ボリュームの例	140
図 13-2	RAID 5 ボリュームの拡張例	141
図 15-1	ホットスペア集合の例	159
図 17-1	トランザクションボリュームの例	177
図 17-2	共有ログトランザクションボリュームの例	178
図 19-1	ディスクセットの例	211
図 22-1	小規模なシステム構成	250

はじめに

本書『Solaris ボリュームマネージャの管理』は、Solaris ボリュームマネージャを使用して、RAID 0 (連結方式およびストライプ方式) ボリュームや、RAID 1 (ミラー) ボリューム、RAID 5 ボリューム、ソフトパーティション、トランザクションログデバイスの作成、変更、使用など、システムにおける記憶装置の要件を管理する方法について説明します。

対象読者

本書は、システム管理者および記憶装置管理者を対象としており、Solaris ボリュームマネージャで行うことのできる管理作業と、Solaris ボリュームマネージャを使用してデータの信頼性と可用性を向上する方法を示します。

内容の紹介

本書は、次のように構成されています。

第1章では、このマニュアルで説明する概念や作業に関する詳細な「ロードマップ」を示します。この章は、このマニュアルの全体的な内容を把握するのに役立ちます。

第2章では、記憶装置管理にまだ精通していない読者のために、一般的な記憶装置管理の概念を紹介します。

第3章では、Solaris ボリュームマネージャについて説明し、この製品に関連する重要な概念を紹介します。

第4章では、Solaris ボリュームマネージャ製品を理解するための全体的なシナリオを示します。

第5章では、状態データベースと状態データベースの複製について説明します。

第6章では、状態データベースと状態データベースの複製に関連する作業の実行方法について説明します。

第7章では、RAID 0(ストライプ方式および連結方式) ボリュームについて説明します。

第8章では、RAID 0(ストライプ方式および連結方式) ボリュームに関連する作業の実行方法について説明します。

第9章では、RAID 1(ミラー) ボリュームについて説明します。

第10章では、RAID 1(ミラー) ボリュームに関連する作業の実行方法について説明します。

第11章では、Solaris ボリュームマネージャのソフトパーティション分割化機能について説明します。

第12章では、ソフトパーティション分割に関連する作業の実行方法について説明します。

第13章では、RAID 5 ボリュームについて説明します。

第14章では、RAID 5 ボリュームに関連する作業の実行方法について説明します。

第15章では、ホットスペアとホットスペア集合について説明します。

第16章では、ホットスペアとホットスペア集合に関連する作業の実行方法について説明します。

第17章では、トランザクションボリュームについて説明します。

第18章では、トランザクションボリュームに関連する作業の実行方法について説明します。

第19章では、ディスクセットについて説明します。

第20章では、ディスクセットに関連する作業の実行方法について説明します。

第21章では、特定の Solaris ボリュームマネージャコンポーネントに関連しない、いくつかの一般的な保守作業について説明します。

第22章では、Solaris ボリュームマネージャを使用して構築できる「最善の記憶装置構成」について説明します。

第23章では、Solaris ボリュームマネージャ SNMP エージェントとその他のエラーチェック手法について説明し、その使用方法を示します。

第24章では、Solaris ボリュームマネージャ環境における障害追跡と一般的な問題の解決方法を示します。

付録 A では、Solaris ボリュームマネージャで使用される重要なファイルの一覧を示します。

付録 B では、コマンドとその他の有用な情報を要約した表を示します。

付録 C では、WBEM 準拠の管理ツールからオープンな Solaris ボリュームマネージャを利用するための CIM/WBEM API について簡単に紹介します。

関連情報

Solaris ボリュームマネージャは、Solaris オペレーティング環境で使用できるシステム管理ツールの 1 つです。システム管理の全体的な特徴や機能、および関連するツールについては、次のマニュアルを参照してください。

- 『Solaris のシステム管理 (基本編)』
- 『Solaris のシステム管理 (上級編)』

Sun のオンラインマニュアル

docs.sun.com では、Sun が提供しているオンラインマニュアルを参照することができます。マニュアルのタイトルや特定の主題などをキーワードとして、検索を行うこともできます。URL は、<http://docs.sun.com> です。

表記上の規則

このマニュアルでは、次のような字体や記号を特別な意味を持つものとして使用します。

表 P-1 表記上の規則

字体または記号	意味	例
AaBbCc123	コマンド名、ファイル名、ディレクトリ名、画面上的コンピュータ出力、コード例を示します。	.login ファイルを編集します。 ls -a を使用してすべてのファイルを表示します。 system%
AaBbCc123	ユーザーが入力する文字を、画面上的コンピュータ出力と区別して示します。	system% su password:
AaBbCc123	変数を示します。実際に使用する特定の名前または値で置き換えます。	ファイルを削除するには、rm <i>filename</i> と入力します。
『 』	参照する書名を示します。	『コードマネージャ・ユーザーズガイド』を参照してください。
「 」	参照する章、節、ボタンやメニュー名、強調する単語を示します。	第 5 章「衝突の回避」を参照してください。 この操作ができるのは、「スーパーユーザー」だけです。
\	枠で囲まれたコード例で、テキストがページ幅を超える場合に、継続を示します。	sun% grep `^#define \ XV_VERSION_STRING'

コード例は次のように表示されます。

■ C シェル

```
machine_name% command y|n [filename]
```

■ C シェルのスーパーユーザー

```
machine_name# command y|n [filename]
```

■ Bourne シェルおよび Korn シェル

```
$ command y|n [filename]
```

■ Bourne シェルおよび Korn シェルのスーパーユーザー

```
# command y|n [filename]
```

[] は省略可能な項目を示します。上記の例は、*filename* は省略してもよいことを示しています。

| は区切り文字 (セパレータ) です。この文字で分割されている引数のうち 1 つだけを指定します。

キーボードのキー名は英文で、頭文字を大文字で示します (例: Shift キーを押します)。ただし、キーボードによっては Enter キーが Return キーの動作をします。

ダッシュ (-) は2つのキーを同時に押すことを示します。たとえば、Ctrl-D は Control キーを押したまま D キーを押すことを意味します。

一般規則

- このマニュアルでは、英語環境での画面イメージを使っています。このため、実際に日本語環境で表示される画面イメージとこのマニュアルで使っている画面イメージが異なる場合があります。本文中で画面イメージを説明する場合には、日本語のメニュー、ボタン名などの項目名と英語の項目名が、適宜併記されています。
- このマニュアルでは、「IA」という用語は、Intel 32 ビットのプロセッサアーキテクチャを意味します。これには、Pentium、Pentium Pro、Pentium II、Pentium II Xeon、Celeron、Pentium III、Pentium III Xeon、Pentium 4 の各プロセッサ、および AMD が提供する互換マイクロプロセッサチップが含まれます。

第 1 章

Solaris ボリュームマネージャの概要

『Solaris ボリュームマネージャの管理』では、高度な可用性、柔軟性、および信頼性を備えた記憶装置を構成するために、Solaris ボリュームマネージャを使用してシステムを設定および管理する方法を説明します。

この章は、記憶容量の設定など、Solaris ボリュームマネージャの特定の作業に関する情報を見つけるためのガイドとして役立ちます。この章では、Solaris ボリュームマネージャを使用する必要があるすべての作業について説明しているわけではありません。次のような Solaris ボリュームマネージャの概念に関連した一般的な作業と、その実行方法を記載した参照先を示します。

- 記憶容量
- 可用性
- 入出力性能
- 管理
- 障害追跡



注意 – Solaris ボリュームマネージャの使い方が正しくないと、データを破壊してしまうおそれがあります。Solaris ボリュームマネージャは、ディスクとディスク上のデータを確実に管理するための強力な手段です。しかし、データのバックアップは、常に取るようにしてください。特に、Solaris ボリュームマネージャの実際の構成を変更するときには、バックアップが必要です。

Solaris ボリュームマネージャで行う作業の概要

Solaris ボリュームマネージャのロードマップ — 記憶容量

表 1-1 Solaris ボリュームマネージャのロードマップ — 記憶容量

作業	説明	参照先
記憶領域の設定	RAID 0 または RAID 5 ボリュームを作成することによって、複数のスライスにまたがる記憶領域を作成します。この RAID 0 または RAID 5 ボリュームは、ファイルシステムやアプリケーション (raw デバイスにアクセスするデータベースなど) のために使用できます。	80 ページの「RAID 0 (ストライプ方式) ボリュームを作成するには」 81 ページの「RAID 0 (連結方式) ボリュームを作成するには」 101 ページの「未使用のスライスから RAID 1 ボリュームを作成するには」 103 ページの「ファイルシステムから RAID 1 ボリュームを作成するには」 148 ページの「RAID 5 ボリュームを作成するには」
既存のファイルシステムの拡張	RAID 0 (連結方式) ボリュームを作成してからスライスを追加することによって、既存のファイルシステムの容量を拡張します。	83 ページの「既存のデータの領域を拡張するには」
既存の RAID 0 (連結方式またはストライプ方式) ボリュームの拡張	既存の RAID 0 ボリュームにスライスを連結することによって、RAID 0 ボリュームを拡張します。	84 ページの「既存の RAID 0 ボリュームを拡張するには」
RAID 5 ボリュームの拡張	既存の RAID 5 ボリュームにスライスを連結することによって、RAID 5 ボリュームの容量を拡張します。	152 ページの「RAID 5 ボリュームを拡張するには」
拡張されたボリューム上の UFS ファイルシステムの拡張	growfs コマンドを使って UFS のサイズを拡張することによって、ファイルシステムを拡張します。この作業は、UFS ファイルシステムをマウントしたまま実行できるので、データへのアクセスを中断する必要はありません。	243 ページの「ファイルシステムを拡張するには」

表 1-1 Solaris ボリュームマネージャのロードマップ — 記憶容量 (続き)

作業	説明	参照先
スライスまたは論理ボリュームをより小さいパーティションに分割し、8 スライスというハードパーティション制限を取り除く	ソフトパーティションを使用して論理ボリュームまたはスライスをさらに分割します。	134 ページの「ソフトパーティションを作成するには」
ファイルシステムの作成	ファイルシステムを RAID 0 (ストライプ方式または連結方式)、RAID 1 (ミラー)、RAID 5、トランザクションのいずれかのボリューム上に作成します。あるいは、ソフトパーティション上に作成します。	『Solaris のシステム管理 (基本編)』の「ファイルシステムの作成 (手順)」

Solaris ボリュームマネージャのロードマップ — 可用性

表 1-2 Solaris ボリュームマネージャのロードマップ — 可用性

作業	説明	参照先
データの可用性を最大限強化	Solaris ボリュームマネージャのミラー化機能を使ってデータの複数のコピーを保持します。データの作成に先立って未使用のスライスから RAID 1 ボリュームを作成できます。あるいは、root (/) や /usr など、既存のファイルシステムをミラー化できます。	101 ページの「未使用のスライスから RAID 1 ボリュームを作成するには」 103 ページの「ファイルシステムから RAID 1 ボリュームを作成するには」
最少限のハードウェアコストでデータの可用性を強化	Solaris ボリュームマネージャの RAID 5 ボリュームを使用することによって、最少限のハードウェアでデータの可用性を高めます。	148 ページの「RAID 5 ボリュームを作成するには」
既存の RAID 1 または RAID 5 ボリュームのデータ可用性の向上	ホットスペア集合を作成し、それをミラーのサブミラーまたは RAID 5 ボリュームと対応付けることによって、RAID 1 または RAID 5 ボリュームのデータ可用性を向上させます。	164 ページの「ホットスペア集合の作成」 166 ページの「ホットスペア集合とボリュームの対応付け」
再起動後のファイルシステムの可用性を高める	UFS ロギング (トランザクションボリューム) をシステムに追加することによって、再起動後のファイルシステムの全体的な可用性を高めます。ファイルシステムをロギングすると、システム再起動時の fsck コマンドの実行時間を短くできます。	175 ページの「ファイルシステムロギングについて」

Solaris ボリュームマネージャのロードマップ — 入出力性能

表 1-3 Solaris ボリュームマネージャのロードマップ — 入出力性能

作業	説明	参照先
RAID 1 ボリュームの読み書きポリシーの調整	RAID 1 ボリュームの読み書きポリシーを指定して、特定の構成における性能を改善します。	92 ページの「RAID 1 ボリュームの読み取りおよび書き込みポリシー」 117 ページの「RAID 1 ボリュームオプションを変更するには」
デバイス性能の最適化	RAID 0 (ストライプ方式) ボリュームを作成すると、そのストライプを構成する各デバイスの性能が最適化されます。飛び越し値を設定することで、ランダムアクセスや順次アクセスを最適化できます。	80 ページの「RAID 0 (ストライプ方式) ボリュームの作成」
RAID 0 (ストライプ方式) ボリュームにおけるデバイス性能の維持	領域が足りなくなったストライプや連結に新しいコンポーネントを連結することによって、ストライプや連結を拡張します。性能に関しては、スライスの連結よりもストライプの連結の方が優れています。	82 ページの「記憶領域の拡張」

Solaris ボリュームマネージャのロードマップ — 管理

表 1-4 Solaris ボリュームマネージャのロードマップ — 管理

作業	説明	参照先
ボリューム管理構成のグラフィカル管理	Solaris 管理コンソールを使ってボリューム管理構成を管理します。	Solaris 管理コンソールアプリケーションの Solaris ボリュームマネージャ (拡張ディスク) ノード内から使用するオンラインヘルプ
スライスやファイルシステムのグラフィカル管理	Solaris 管理コンソールのグラフィカルユーザーインターフェースを使って、ディスクやファイルシステムを管理します。ここでは、ディスクのパーティション分割や UFS ファイルシステムの構築といった作業を行います。	Solaris 管理コンソールアプリケーション内から使用するオンラインヘルプ
Solaris ボリュームマネージャの最適化	Solaris ボリュームマネージャの性能は、構成が適切に設定されているかどうかによって決まります。構成を作成したら、その構成を監視および調整する必要があります。	48 ページの「Solaris ボリュームマネージャ構成の指針」 238 ページの「構成ファイルの使用」

表 1-4 Solaris ボリュームマネージャのロードマップ — 管理 (続き)

作業	説明	参照先
将来の拡張に対する備え	ファイルシステムの領域を使い切ってしまうことがあるため、ファイルシステムを連結で構成することによって将来の拡張に備えることができます。	81 ページの「RAID 0 (連結方式) ボリューム」 82 ページの「記憶領域の拡張」

Solaris ボリュームマネージャのロードマップ — 障害追跡

表 1-5 Solaris ボリュームマネージャのロードマップ — 障害追跡

作業	説明	参照先
不良スライスの交換	ディスクに障害が発生したときは、Solaris ボリュームマネージャ構成で使用されているスライスを交換する必要があります。RAID 0 ボリュームの場合は、ボリュームを削除し、新しいスライスにボリュームを作成し直した後で、バックアップからデータを復元する必要があります。RAID 1 と RAID 5 ボリュームの場合は、スライスの交換と再同期を行ってもデータが失われることはありません。	120 ページの「RAID 1 ボリュームのコンポーネント障害に対する処置」 154 ページの「RAID 5 ボリューム内のコンポーネントを置き換えるには」
起動障害からの回復	ハードウェア障害やオペレータの操作ミスによって、システムの起動時に特殊な問題が起ることがあります。	276 ページの「/etc/vfstab 内の不適切なエントリを修正するには」 282 ページの「状態データベースの複製数の不足から回復するには」 278 ページの「起動デバイスの障害から回復するには」
トランザクションボリュームの障害に関する作業	トランザクションボリュームの障害は、マスターデバイスやロギングデバイスで発生する可能性があり、その原因としてデータやデバイスの障害が挙げられます。同じロギングデバイスを共有しているすべてのトランザクションボリュームを修復しないと、ボリュームを使用可能な状態に戻すことはできません。	204 ページの「パニックした場合のトランザクションボリュームを回復するには」 204 ページの「ハードウェアエラー状態のトランザクションボリュームを回復するには」

第 2 章

記憶装置管理の概念

この章では、一般的な記憶装置管理の概念について簡単に紹介します。記憶装置の管理についてすでに精通している場合は、第 3 章に進んでください。

この章では、以下の内容について説明します。

- 29 ページの「記憶装置管理の紹介」
- 31 ページの「構成計画の指針」
- 33 ページの「性能について」
- 34 ページの「ランダム入出力と順次入出力の最適化」

記憶装置管理の紹介

記憶装置管理は、システムのアクティブなデータが格納されているデバイスを制御するための手段です。デバイスが有効であるためには、予期せぬ事態 (ハードウェア障害やソフトウェア障害など) が発生した後も、アクティブなデータを、以前と同じ (一貫性を保った) 状態で使用できなければなりません。

記憶装置ハードウェア

データを格納するには、さまざまなデバイスを利用できます。記憶装置の要件にもっとも適したデバイスは、主に次の 3 つの要素を考慮して決定します。

- 性能
- 可用性
- コスト

Solaris ボリュームマネージャは、性能、可用性、およびコストの間で最適なバランスを保つのに役立ちます。Solaris ボリュームマネージャを使用することにより、最善のトレードオフを達成できます。

Solaris ボリュームマネージャは、Solaris™ オペレーティング環境を稼働するシステム上でサポートされているすべての記憶装置に対して使用できます。

RAID レベル

RAID は Redundant Array of Inexpensive (または Independent) Disks の略語です。この用語は基本的に、ユーザーからは1つの大きなディスクドライブに見えるディスク群 (アレイ、または、より一般的にはボリュームと呼ばれる) を意味します。構成によって異なりますが、このアレイによって信頼性や応答時間が向上し、記憶容量が増加します。

技術的には、6つの RAID レベル (0 から 5) があります。各レベルは、データの冗長性を確保しながらデータの分散を図る方法に対応しています。(RAID レベル 0 はデータの冗長性を提供しませんが、このレベルは使用するほとんどの RAID 構成の基礎になっているため、通常、RAID に分類されています。) RAID レベル 2、3、4 をサポートする記憶装置環境はほとんど存在しないため、これらのレベルの説明は省略します。

Solaris ボリュームマネージャは、次の RAID レベルをサポートします。

- RAID レベル 0 – ストライプと連結は冗長性を備えていませんが、通常、RAID 0 と呼ばれています。基本的に、データは、複数の物理ディスクに交互かつ均一に割り当てられる比較的小さな同じ大きさの領域に分散されます。ドライブの1つに障害が発生しただけで、データは失われます。RAID 0 では高いデータ転送速度や入出力スループットが得られますが、信頼性や可用性は単一ディスクよりも劣ります。
- RAID レベル 1 – ミラー化では、同じ容量の複数のディスクにデータとそのコピー (ミラー) を別々に格納します。データは、2つ以上の物理ディスクに複製 (またはミラー化) されます。データを両方のドライブから同時に読み取ることができるため (どちらのドライブも要求に応えられる)、性能が向上します。1つの物理ディスクに障害が発生しても、性能の低下やデータの損失なしにミラーを引き続き使用できます。

Solaris ボリュームマネージャは、使用する装置によっては、RAID 0+1 ミラー化と (透過的に) RAID 1+0 ミラー化の両方をサポートします。詳細は、89 ページの「RAID 1+0 と RAID 0+1 の提供」を参照してください。

- RAID レベル 5 – RAID 5 は、ストライプ化によってアレイ内の複数のディスクにデータを分散します。また RAID 5 では、パリティ情報を格納することによって、データの冗長性を確保します。RAID レベル 5 ボリュームは、1つのデバイスの障害に耐えることができます。さらに、ホットスペアと組み合わせて使用すれば、複数のデバイスの障害にも耐えることができます。ただし、障害のあるデバイスが存在する間は、性能が大幅に低下します。

RAID 5 モデルの各デバイスには、パリティストライプを含む1つの領域と、データを含むそれ以外の領域があります。パリティはアレイ内のすべてのディスクに分散されるため、個々の大量の書き込みに要する書き込み時間が短縮されます。これは、単一のパリティディスクの場合と同様に、個々の書き込み操作において、データの書き込みが可能になるまで待機する必要がないためです。

構成計画の指針

記憶装置管理の構成を計画するときは、どのようなアプリケーションにも性能、可用性、ハードウェアコストの間にトレードオフがあることを考慮する必要があります。最適な構成を見つけるためには、さまざまな要素の組み合わせを試してみる必要があります。

この節では、Solaris ボリュームマネージャの RAID 0 (連結方式およびストライプ方式) ボリューム、RAID 1 (ミラー) ボリューム、RAID 5 ボリューム、ソフトパーティション、トランザクション (ロギング) ボリューム、および、これらのボリューム上に構築されるファイルシステムを使用するための指針を示します。

記憶方式の選択

特定の記憶方式を実装する前に、どのような記憶デバイスを使用するかを決める必要があります。この一連の指針では、記憶デバイスを選択する上で必要なさまざまな記憶方式を比較します。Solaris ボリュームマネージャで実装されている個々の記憶方式には、その方式固有の指針が適用されます。詳細は、個々のボリュームタイプに関する章を参照してください。

注 - 下記の記憶方式は排他的なものではありません。つまり、これらの方式を組み合わせることで使用することによって、複数の目的を達成することができます。たとえば、RAID 1 ボリュームを作成して冗長性を確保してから、その上にソフトパーティションを作成して、独立したファイルシステムの数を増加することができます。

表 2-1 記憶方式の選択

要件	RAID 0 (連結方式)	RAID 0 (ストライプ方式)	RAID 1 (ミラー)	RAID 5	ソフトパーティション
データの冗長性	不可	不可	可能	可能	不可
読み取り性能の改善	不可	可能	使用するデバイスによって異なる	可能	不可
書き込み性能の改善	不可	可能	不可	不可	不可

表 2-1 記憶方式の選択 (続き)

要件	RAID 0 (連結方式)	RAID 0 (ストライプ方式)	RAID 1 (ミラー)	RAID 5	ソフトパーティション
デバイス当たり 8 個以上のスライスの作成	不可	不可	不可	不可	可能
使用可能な記憶装置の増加	可能	可能	不可	可能	不可

表 2-2 冗長性の最適化

	RAID 1 (ミラー)	RAID 5
書き込み操作	高速	低速
ランダム読み込み	高速	低速
ハードウェアコスト	高い	少ない

- RAID 0 デバイス (ストライプ方式および連結方式) とソフトパーティションは、データの冗長性を提供しません。
- 連結は、小規模なランダム 入出力に適しています。
- ストライプ化は、大規模な順次入出力や、不均一な入出力に適しています。
- ミラー化によって読み取り性能が向上することがあります。書き込み性能は常に低下します。
- 読み取り - 変更 - 書き込みという RAID 5 ボリューム の性質上、書き込みが 20 パーセントを超えるボリュームでは、RAID 5 を使用しないようにします。冗長性が必要な場合は、ミラー化を検討してください。
- RAID 5 の書き込み性能は、ミラー化の書き込み性能よりも常に低くなります。ミラー化の書き込みは、書き込みを保護しない方式よりも常に低速です。
- ソフトパーティションは、非常に大規模な記憶デバイスの管理に有効です。

注 - Solaris ボリュームマネージャを使って冗長性を備えたデバイスをサポートする方法については、これらの一般的な記憶方式の他に、47 ページの「ホットスペア集合」を参照してください。

性能について

性能に関する一般的な指針

記憶装置の構成を決めるときは、性能に関する次の指針を考慮する必要があります。

- 一般に、もっとも高い性能が得られるのはストライプ化ですが、ストライプ化はデータの冗長性を提供しません。書き込みが多いアプリケーションでは、一般に、RAID 1 ボリュームの方が RAID 5 ボリュームよりも高い性能を提供します。
- RAID 1 と RAID 5 ボリュームではデータの可用性は向上しますが、一般に性能は低下します (特に、書き込み操作の場合)。ミラー化は、ランダムな読み取り性能を向上します。
- RAID 5 ボリュームのハードウェアコストは RAID 1 ボリュームよりも少なくなります。RAID 0 ボリュームでは、追加のハードウェアコストは発生しません。
- もっともアクセス頻度の高いデータを特定し、そのデータに対するアクセス帯域幅をミラー化またはストライプ化によって拡張します。
- ストライプと RAID 5 ボリュームではデータが複数のディスクドライブに分散されるため、入出力負荷が均一化されます。
- 性能監視機能や一般的なツール (iostat コマンドなど) を使って、もっともアクセス頻度の高いデータを特定します。そのデータに対するアクセス帯域幅を、ストライプ化、RAID 1 ボリューム、または RAID 5 ボリュームを使って拡張します。
- ソフトパーティションのサイズを複数回変更すると、性能が低下することがあります。
- 書き込み操作の場合、RAID 5 ボリュームの性能はストライプの性能よりも低くなります。これは、RAID 5 ボリュームがパリティの計算と格納に複数の入出力操作を必要とするためです。
- ランダム raw 入出力の読み取りの場合には、ストライプと RAID 5 ボリュームの性能は同等です。ストライプでも RAID 5 ボリュームでも、データは複数のディスクに分割されます。また、スライスに障害が発生した場合を除き、RAID 5 ボリュームのパリティ計算は、読み取り性能に影響を与える要因にはなりません。
- ランダム raw 入出力の書き込みでは、ストライプの方が RAID 5 ボリュームよりも優れています。

ランダム入出力と順次入出力の最適化

この節では、特定の構成を最適化する際に用いる、Solaris ボリュームマネージャの手法について説明します。

Solaris ボリュームマネージャのボリューム上に実装するファイルシステムで順次入出力とランダム入出力のどちらが多用されるのかわからない場合は、性能のチューニングに関する以下の手法を適用しないでください。実装が適切でないと、性能が低下することがあります。

以下に示す最適化のための手法では、RAID 0 ボリュームを最適化するものとします。通常は、RAID 0 ボリュームを最適化してからそのボリュームをミラー化して、最適な性能とデータの冗長性を同時に達成します。

ランダム入出力

データベースや汎用ファイルサーバーに使用されるランダム入出力環境では、すべてのディスクスピンドルが入出力要求に対してほぼ同じ割合で使用されることが望まれます。

たとえば、データベースアプリケーション用に 40G バイトの記憶領域があるとし、4 つの 10G バイトディスクスピンドルを使ってストライプ化を行い、入出力負荷が完全にランダムであり、テーブル領域全体に均一に分布していれば、4 つのスピンドルの使用率はほぼ均一になります。これによって、通常、性能が向上します。

ランダム入出力性能を最大化するためには、ディスクの使用率を 35 パーセント以下に抑えるようにします (iostat コマンドで調べることができる)。ディスクの使用率が定常的に 65 パーセントを超える場合には問題となります。90 パーセントを超える場合には重大な問題が発生します。このような問題を解決するためには、さらに多くのディスク (スピンドル) を追加して、新しい RAID 0 ボリュームを作成する必要があります。

注 - 既存のボリュームにディスクを追加するだけでは、性能を向上することはできません。適切なパラメータを設定して新しいボリュームを作成し、性能の最適化を図る必要があります。

データをすべてのディスクに分散する場合には、飛び越し値のサイズは重要ではありません。一般的な入出力要求よりも大きい飛び越し値を指定するだけで十分です。

順次アクセス入出力

飛び越し値を通常の入出力要求のサイズよりも小さく設定することによって、構成を最適化することにより、順次入出力環境の利点を活用することができます。このような環境の例として、テーブル全体のスキャンが頻繁に行われる DBMS サーバーや、データ入出力が非常に多い NFS サーバーなどが挙げられます。

たとえば、通常の入出力要求のサイズが 256K バイトで、ストライプが 4 スピンドルに渡っているとします。この場合の適切なストライプユニットサイズは $256\text{K バイト} / 4 = 64\text{K バイト}$ またはそれ以下です。

この場合には、通常の入出力要求が複数のディスクスピンドルに分散されるため、順次入出力の帯域幅が増加します。

注 - 順次アクセスでは、シーク時間と回転待ち時間は実質的にゼロです。順次入出力の最適化では、ディスクの内部転送速度が最も重要な要素になります。

順次アプリケーションの通常の入出力サイズには大きい値を使用します (128K バイト以上、場合によっては 1M バイト以上)。ここでは、通常の入出力要求のサイズが 256K バイトで、ストライプが 4 ディスクスピンドルに渡っているとします。256K バイト / 4 = 64K バイトであるため、適切な飛び越し値は 32 から 64K バイトとなります。

第 3 章

Solaris ボリュームマネージャの概要

この章では、Solaris ボリュームマネージャの全体的な構造について説明します。この章の内容は次のとおりです。

- 37 ページの「Solaris ボリュームマネージャの機能」
- 41 ページの「Solaris ボリュームマネージャの前提条件」
- 41 ページの「Solaris ボリュームマネージャコンポーネントの概要」
- 48 ページの「Solaris ボリュームマネージャ構成の指針」
- 49 ページの「Solaris ボリュームマネージャ要素の作成」

Solaris ボリュームマネージャの機能

Solaris ボリュームマネージャは、多数のディスクとそれらのディスク上のデータを管理するためのソフトウェア製品です。Solaris ボリュームマネージャにはいろいろな用途がありますが、ほとんどの場合、次の目的で使用されます。

- 記憶容量を増加する
- データ可用性を向上する
- 大規模な記憶デバイスの管理を容易にする

状況によっては、Solaris ボリュームマネージャを使用すると、入出力性能が向上することもあります。

Solaris ボリュームマネージャによる記憶装置の管理方法

Solaris ボリュームマネージャは、仮想ディスクを使って物理ディスクとそのデータを管理します。Solaris ボリュームマネージャでは、仮想ディスクをボリュームと呼びます。ただし、従来からの慣習により、コマンド行ユーティリティではボリュームを「メタデバイス」と呼ぶこともあります。

アプリケーションやファイルシステム (UFS など) から見ると、ボリュームは物理ディスクと同じように機能します。Solaris ボリュームマネージャは、ボリュームに対する入出力要求を、そのボリュームを構成するメンバーディスクに対する入出力要求に変換します。

Solaris ボリュームマネージャのボリュームは、スライス (ディスクパーティション) または他の Solaris ボリュームマネージャボリュームから作成されます。Solaris 管理コンソールに組み込まれているグラフィカルユーザーインターフェースを使えば、ボリュームを簡単に作成できます。Solaris 管理コンソール内の「拡張ディスク」を使用することで、存在しているすべてのボリュームの情報を表示できます。管理者は、ウィザードのステップに従うだけで、Solaris ボリュームマネージャのあらゆる種類のボリュームとコンポーネントを簡単に作成できます。また、Solaris ボリュームマネージャのコマンド行ユーティリティを使用しても、ボリュームの作成や変更が行えます。

たとえば、大きな記憶領域を1つのボリュームとして作成したい場合、Solaris ボリュームマネージャを使用すると多数の小さいスライスの集まりを1つの大きなスライスまたはデバイスとして扱うように、システムを設定できます。このようなスライスから大きなボリュームを作成したら、その他のボリュームをただちに「実」スライスまたはデバイスとして使用できます。

ボリュームの詳細は、42 ページの「ボリューム」を参照してください。

Solaris ボリュームマネージャでは、RAID 1 (ミラー) ボリュームや RAID 5 ボリュームを使ってデータの信頼性と可用性を高めることができます。Solaris ボリュームマネージャのホットスペアを使用すれば、ミラーや RAID 5 ボリュームのデータ可用性をさらに向上できます。

構成の設定が完了したら、Solaris 管理コンソール内の「拡張ディスク」を使って動作状況を検査できます。

Solaris ボリュームマネージャとの対話方法

Solaris ボリュームマネージャと対話するには、次のどちらかの方法を使用します。

- Solaris 管理コンソール – このツールは、ボリューム管理機能とのグラフィカルユーザーインターフェースとして機能します。図 3-1 に、Solaris 管理コンソール内の「拡張ディスク」の使用例を示します。このインターフェースは、ボリュームや、ホットスペア、状態データベースの複製など、Solaris ボリュームマネージャのコ

ンポーネントをグラフィカルに表示します。このインタフェースではウィザードを使って Solaris ボリュームマネージャコンポーネントを操作できるため、ディスクの構成や既存の構成を簡単に変更できます。

- コマンド行インタフェース – ボリューム管理機能を実行するためのコマンドが用意されています。Solaris ボリュームマネージャのコアコマンドは、`metainit` や `metastat` のように `meta` で始まります。Solaris ボリュームマネージャコマンドの一覧については、付録 B を参照してください。

注 – コマンド行インタフェースとグラフィカルユーザーインタフェースを同時に使って、Solaris ボリュームマネージャを管理しないでください。構成に対して矛盾した変更が加えられ、予測していない動作が生じることがあります。どちらのツールを使っても、Solaris ボリュームマネージャを管理することは可能ですが、両方のツールを同時に使用することはできません。

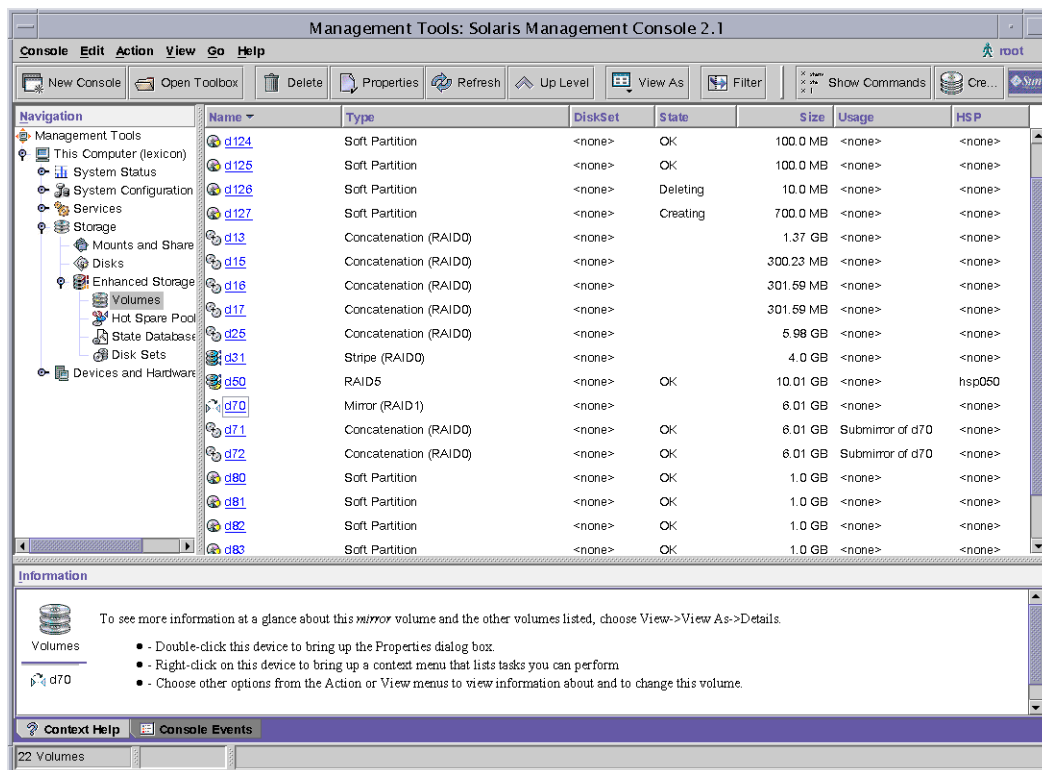


図 3-1 Solaris 管理コンソール内の「拡張ディスク (Enhanced Storage)」ツール (Solaris ポリリュームマネージャ) の使用例

▼ Solaris ポリリュームマネージャグラフィカルユーザーインターフェースを使用するには

Solaris ポリリュームマネージャのグラフィカルユーザーインターフェース (拡張ディスク) は Solaris 管理コンソールの一部です。このインターフェースにアクセスするには、次の手順に従います。

1. 次のコマンドを使って、ホストシステム上で **Solaris** 管理コンソールを起動します。

```
% /usr/sbin/smc
```
2. 「このコンピュータ (**This Computer**)」をダブルクリックします。
3. 「**Storage**」をダブルクリックします。

4. 「拡張ディスク (Enhanced Storage)」をダブルクリックして、Solaris ボリュームマネージャツールを起動します。
5. ログインを促すプロンプトが表示されたら、root または同等のアクセス権をもつユーザーとしてログインします。
6. 適切なアイコンをダブルクリックして、ボリュームや、ホットスペア集合、状態データベースの複製、ディスクセットを管理します。

ヒント – Solaris 管理コンソールの各ツールは、作業の助けになる情報をページの下部またはウィンドウパネル左側に表示します。このインタフェースでの作業について情報が必要な場合は、いつでも「ヘルプ (Help)」を選択できます。

Solaris ボリュームマネージャの前提条件

Solaris ボリュームマネージャを使用するための前提条件は、次のとおりです。

- Solaris ボリュームマネージャを管理するためにはルート権限が必要です。Solaris 管理コンソールの User Profile 機能によって同等の権限が与えられていれば、Solaris 管理コンソールを使って管理を行うことができます。ただし、Solaris ボリュームマネージャのコマンド行インタフェースを使用できるのは、スーパーユーザーだけです。
- Solaris ボリュームマネージャを使ってボリュームを作成するためには、少なくとも3つの状態データベースの複製が、Solaris ボリュームマネージャを稼働するシステム上に存在していなければなりません。最大限の信頼性を確保するためには、これらの複製を異なるコントローラとディスクに配置するようにします。状態データベースの複製の詳細については、55 ページの「Solaris ボリュームマネージャの状態データベースと状態データベースの複製について」を、状態データベースの複製の作成手順については、64 ページの「状態データベースの複製の作成」をそれぞれ参照してください。

Solaris ボリュームマネージャコンポーネントの概要

Solaris ボリュームマネージャで作成する基本的なコンポーネントタイプには、ボリューム、ディスクセット、状態データベースの複製、ホットスペア集合があります。次の表に、Solaris ボリュームマネージャのコンポーネントの概要を示します。

表 3-1 Solaris ボリュームマネージャコンポーネントの要約

Solaris ボリュームマネージャコンポーネント	定義	目的	参照先
RAID 0 ボリューム (ストライプ方式、連結方式、ストライプ方式の連結)、RAID 1 (ミラー) ボリューム、RAID 5 ボリューム	システム上で単一の論理デバイスとして扱われる物理スライスの集まり	記憶容量、性能、またはデータの可用性を高める	42 ページの「ボリューム」
ソフトパーティション	物理スライスまたは論理ボリュームを分割したもので、より小さく管理しやすい記憶ユニットを提供する	大規模な記憶ボリュームを管理し易くする	
状態データベース (状態データベースの複製)	Solaris ボリュームマネージャ構成の状態に関する情報を格納するデータベース	状態データベースの複製を作成しないと、Solaris ボリュームマネージャは動作しない	46 ページの「状態データベースと状態データベースの複製」
ホットスペア集合	サブミラーまたは RAID 5 ボリュームのコンポーネントに障害が発生したときに、自動的に交換されるように予約されているスライス (ホットスペア) の集合	RAID 1 と RAID 5 ボリュームのデータ可用性を高める	47 ページの「ホットスペア集合」
ディスクセット	個別の名前空間をもつ共有ディスクドライブの集まり。ボリュームとホットスペアを含み、複数のホストによって排他的に共有される	データの冗長性と可用性を提供し、また個別の名前空間を提供することによって管理を容易にする	48 ページの「ディスクセット」

ボリューム

ボリュームとは、システム上で単一の論理デバイスとみなされる物理スライスの集まりの名前です。実際には、ボリュームは標準 UNIX の擬似または仮想デバイスと同義です。

注 - これまで、Solstice DiskSuite™ 製品ではボリュームを「メタデバイス」と呼んでいましたが、このマニュアルでは、簡潔性と標準化のために「ボリューム」と呼びます。

ボリュームクラス

ボリュームには、RAID 0 (連結方式およびストライプ方式) ボリューム、RAID 1 (ミラー) ボリューム、RAID 5 ボリューム、ソフトパーティション、UFS ロギングボリュームがあります。

ボリュームの作成や管理には、Solaris 管理コンソール内の「拡張ディスク」かコマンド行ユーティリティを使用します。

次の表に、ボリュームクラスの要約を示します。

表 3-2 ボリュームクラス

ボリューム	説明
RAID 0 (ストライプ方式または連結方式)	そのまま使用することもでき、また、ミラーおよびトランザクションデバイスの基本的な構築ブロックとして使用することもできます。RAID 0 ボリューム自体にはデータの冗長性はありません。
RAID 1 (ミラー)	複数のコピーを保持することによってデータを複製します。RAID 1 ボリュームは、サブミラーと呼ばれる 1 つまたは複数の RAID 0 ボリュームから構成されます。
RAID 5	パリティ情報を使ってデータを複製します。ディスクに障害が発生すると、利用可能なデータとパリティ情報から失われたデータが復元されます。RAID 5 ボリュームは、通常、複数のスライスで構成されています。1 スライスに相当する領域がパリティ情報に割り当てられますが、パリティ情報は RAID 5 ボリュームのすべてのスライスに分散されません。
トランザクション	UFS ファイルシステムのログとして使用されます (ただし、この目的のためには UFS ロギングを推奨します)。トランザクションボリュームは、マスターデバイスとロギングデバイスから構成されています。これらのデバイスには、スライス、RAID 0 ボリューム、RAID 1 ボリューム、または RAID 5 ボリュームを使用できます。マスターデバイスには、UFS ファイルシステムが含まれています。
ソフトパーティション	スライスまたは論理ボリュームをより小さい 1 つまたは複数の拡張可能ボリュームに分割します。

ボリュームの使用方法

ボリュームを使用すると、記憶領域を拡張したり、性能やデータの可用性を高めたりできます。状況によっては、ボリュームを使用すると、入出力性能が向上することもあります。機能的には、ボリュームとスライスは同じように動作します。エンドユーザーや、アプリケーション、ファイルシステムからは、ボリュームとスライスは同じように見えます。物理デバイスと同じように、ボリュームはブロック または raw デバイス名を使ってアクセスされます。ボリューム名は、ブロックデバイスを使用するか raw デバイスを使用するかによって異なります。ボリューム名の詳細は、45 ページの「ボリューム名」を参照してください。

ボリューム上では、ほとんどのファイルシステムコマンド (mkfs、 mount、 umount、 ufsdump、 ufsrestore など) を使用できますが、format コマンドは使用できません。ボリューム上にマウント済みのファイルシステムが存在すれば、ボリュームとの間でファイルの読み取り、書き込み、コピーを行うことができます。

例 — 2つのスライスから構成されるボリューム

図 3-2 に、Disk A の 1つのスライスと Disk B の 1つのスライスから構成されるボリュームを示します。アプリケーションや UFS は、このボリュームを 1つの物理ディスクとして扱います。ボリュームにさらに多くのスライスを追加すれば、容量を増やすことができます。

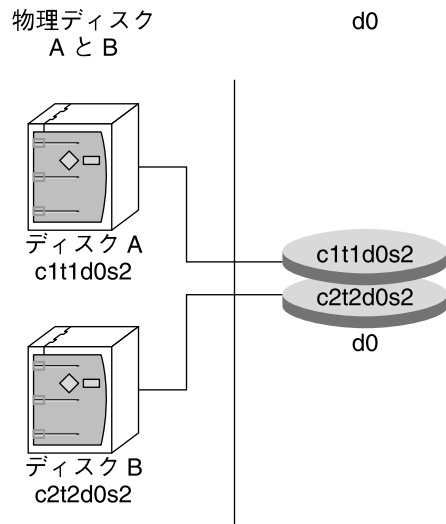


図 3-2 ボリューム、物理ディスク、スライスの関係

ボリュームとディスク領域の拡張

Solaris ボリュームマネージャでは、ボリュームにスライスを追加することによってボリュームを拡張できます。既存のボリュームにスライスを追加するには、Solaris 管理コンソール内の「拡張ディスク」かコマンド行インタフェースを使用します。

ボリュームに含まれている UFS ファイルシステムは、マウントされているかどうかに関係なく、システムを停止したりバックアップをとらなくても、拡張できます。(ただし、どのような場合でも、データのバックアップをとることを推奨します。) ボリュームを拡張したときは、growfs コマンドを使ってファイルシステムを拡張します。

注 – いったん拡張したファイルシステムを縮小することはできません。これは UFS の制約です。同じように、いったん拡張した Solaris ボリュームマネージャのパーティションを縮小することはできません。

raw ボリュームを使用するアプリケーションやデータベースは、新たに追加された領域を独自の方法で拡張し、それを認識する必要があります。Solaris ボリュームマネージャには、この機能はありません。

ボリュームのディスク領域を拡張するには、次の方法を利用できます。

- RAID 0 ボリュームに 1 つまたは複数のスライスを追加する
- RAID 1 ボリュームのすべてのサブミラーに 1 つまたは複数のスライスを追加する
- RAID 5 ボリュームに 1 つまたは複数のスライスを追加する
- パーティションを構成するコンポーネントから領域を追加することによって、ソフトパーティションを拡張する。

growfs コマンド

growfs コマンドは、サービスの中断やデータを失うことなく、UFS ファイルシステムを拡張します。ただし、growfs コマンドが実行している間は、ボリュームへの書き込みアクセスはできません。ファイルシステムは、それを格納しているスライスまたはボリュームのサイズまで拡張できます。

追加ディスク領域の一部だけを使ってファイルシステムを拡張する場合は、growfs コマンドに `-s size` オプションを指定します。

注 – ミラーを拡張すると、ミラーを構成しているすべてのサブミラーに領域が追加されます。同じように、トランザクションボリュームを拡張すると、マスターデバイスに領域が追加されます。その後で RAID 1 ボリュームまたはトランザクションボリュームに対して growfs コマンドを実行します。一般的な規則としては、ボリュームを構成するデバイスに領域を追加してから、トップレベルのデバイスに対して growfs コマンドを実行します。

ボリューム名

ボリューム名の規則

ボリュームに名前を付けるときは、次の規則に従います。

- ボリューム名は d で始まり、その後に 1 つの数字が続きます (たとえば、d0)。

- Solaris ボリュームマネージャでは、0 から127 までの 128 個のボリューム名を使用できます。次の表に、ボリューム名の例を示します。

表 3-3 ボリューム名の例

/dev/md/dsk/d0	ブロック型ボリューム d0
/dev/md/dsk/d1	ブロック型ボリューム d1
/dev/md/rdisk/d126	raw ボリューム d126
/dev/md/rdisk/d127	raw ボリューム d127

- meta* コマンドでは、/dev/md/dsk/d1 のように完全なボリューム名を指定する代わりに、d1 のように省略形のボリューム名を指定できます。
- 物理スライスと同じように、ボリュームにも、ファイルシステムで使用される論理名があります。論理ボリューム名は、/dev/md/dsk ディレクトリ (ブロックデバイスの場合) または /dev/md/rdisk ディレクトリ (raw デバイスの場合) に作成されます。
- 名前を変更しようとするボリュームが現在使用されておらず、かつ、新しい名前が他のボリュームで使用されていないければ、ボリューム名はいつでも変更できます。詳細は、236 ページの「ボリューム名の交換」を参照してください。

ボリューム名に関する指針

ボリューム名を標準化すると、管理が容易になるだけでなく、ボリュームタイプが一目でわかるようになります。次の推奨事項を考慮してください。

- 特定のボリュームタイプごとに範囲を指定します。たとえば、RAID 1 ボリュームには 0 から 20、RAID 0 ボリュームには 21 から 40 を割り当てます。
- ミラーの名前を相互に関連付けます。たとえば、ミラーの名前を 0 で終わる数字にし、サブミラーの名前を 1 や 2 で終わる数字にします。これに従えば、最初のミラーは d10、そのサブミラーは d11 と d12 になります。さらに、次のミラーは d20、そのサブミラーは d21 と d22 になります。
- スライス番号とディスク番号がボリューム番号に対応するような命名方法を使用します。

状態データベースと状態データベースの複製

状態データベースは、Solaris ボリュームマネージャ構成の状態に関する情報を格納するデータベースです。状態データベースは、構成に対して加えられた変更を記録および管理します。Solaris ボリュームマネージャは、構成や状態に変化があると、状態データベースを自動的に更新します。たとえば、新しいボリュームの作成は構成の変更であり、サブミラーの障害は状態の変化を意味します。

状態データベースは、実際には、複製された複数のデータベースコピーの集まりです。各コピーは、状態データベースの複製と呼ばれ、データベース内のデータが常に有効であることを保証します。状態データベースのコピーを複数持つことにより、単一点障害からデータを保護することができます。状態データベースは、既知の状態データベースの複製の格納場所と状態をすべて記録しています。

状態データベースとその状態データベースの複製が作成されるまで、Solaris ボリュームマネージャは動作できません。Solaris ボリュームマネージャの構成には、必ず有効な状態データベースが必要です。

構成を設定するときは、状態データベースの複製を次のどちらかに配置できます。

- 専用のスライス上
- 後でボリュームの一部となるスライス上

Solaris ボリュームマネージャは、状態データベースの複製が割り当てられているスライスを認識し、そのスライスがボリュームに使用されている場合には、複製用に予約されているスライス部分を自動的にスキップします。この部分を他の目的に使用することはできません。

複数の状態データベースのコピーを1つのスライス上に置くこともできますが、そのようにすると、システムは単一点障害に対して脆弱になります。

すべての状態データベースの複製が削除された場合でも、システムは正常に動作します。しかし、状態データベースの複製がディスク上にまったくない状態でシステムを再起動すると、すべての Solaris ボリュームマネージャの構成データが失われます。

ホットスペア集合

ホットスペア集合は、Solaris ボリュームマネージャによって予約されているスライス(ホットスペア)の集合であり、サブミラーまたは RAID 5 ボリュームのスライスに障害が発生した場合には、Solaris ボリュームマネージャがそのスライスとホットスペアを自動的に交換します。RAID 1 と RAID 5 ボリュームでは、ホットスペアを使用することによってデータの可用性が向上します。ホットスペア集合は、Solaris 管理コンソール内の「拡張ディスク」とコマンド行インタフェースのどちらでも作成できます。

エラーが発生すると、Solaris ボリュームマネージャは、障害のあるスライスと同じサイズかそれ以上のサイズのホットスペアをホットスペア集合から探します。該当するホットスペアが見つかり、Solaris ボリュームマネージャは自動的にデータの再同期をとります。適切なサイズのスライスがホットスペア集合にないと、サブミラーまたは RAID 5 ボリュームは使用不能とみなされます。詳細は、第 15 章を参照してください。

ディスクセット

共有ディスクセット、または単にディスクセットは、複数のホストによって排他的に共有される (一度に 1 台のホストだけが使用できる) ディスクドライブの集まりです。ディスクセットには、状態データベースの複製や、ボリューム、ホットスペアが含まれています。

クラスタ環境では、ディスクセットの使用によってデータの可用性が向上します。つまり、一方のホストに障害が発生しても、そのディスクセットを他方のホストが引き継ぐことができます (このタイプの構成はフェイルオーバー構成と呼ばれます)。さらに、ディスクセットを使用することにより、Solaris ボリュームマネージャの名前空間の管理や、ネットワーク接続されている記憶装置へのアクセスが容易になります。

詳細は、第 19 章を参照してください。

Solaris ボリュームマネージャ構成の指針

Solaris ボリュームマネージャの構成が適切に設定されていないと、性能が低下することがあります。この節では、Solaris ボリュームマネージャの性能を最適に保つためのヒントについて説明します。

一般的な指針

- ディスクとコントローラードライブを別々のドライブパス上のボリュームに置きます。SCSI ドライブの場合は、別のホストアダプタに置くようにします。入出力負荷をいくつかのコントローラに分散することによって、ボリュームの性能と可用性が向上します。
- システムファイル - /etc/lvm/mddb.cf や /etc/lvm/md.cf ファイルは変更したり削除したりしないでください。
これらのファイルは、定期的にバックアップしてください。
- ボリュームの整合性 - スライスボリュームとして設定し、有効にしたら、そのスライスを他の目的に使用しないでください。
- ボリュームの最大数 - 1 つのディスクセットでサポートされる最大ボリューム数は 8192 (デフォルトでは 128) です。デフォルトの最大ボリューム数を増やすには、/kernel/drv/md.conf ファイルを編集する必要があります。このファイルの詳細は、295 ページの「システムファイルと始動ファイル」を参照してください。
- ディスクとパーティションに関する情報 - 不良ディスクを再フォーマットしたり、Solaris ボリュームマネージャの構成を作成し直したりしなければならない場合に備えて、prtvtoc と metastat -p コマンドの出力コピーを保管してください。

ファイルシステムに関する指針

- ボリュームを構成しているスライス上にファイルシステムをマウントしないでください。スライスがボリュームを構成するために使用されている場合は、そのスライスをファイルシステムとしてマウントしてはなりません。可能であれば、ボリュームとして使用する予定の物理デバイスは、アクティブにする前にマウント解除してください。たとえば、UFS 用のトランザクションボリュームを `/etc/vfstab` ファイルに作成する場合は、トランザクションボリューム名を、マウントおよび `fsck` を実行するデバイスとして指定します。

Solaris ボリュームマネージャ要素の作成

Solaris ボリュームマネージャのコンポーネントを作成するときは、物理スライスを Solaris ボリュームマネージャの論理名に割り当てます。作成できる Solaris ボリュームマネージャ要素には、次のものがあります。

- 状態データベースの複製
- ボリューム (RAID 0 (ストライプ方式および連結方式)、RAID 1 (ミラー)、RAID 5、ソフトパーティション、トランザクションボリューム)
- ホットスペア集合
- ディスクセット

注 - ボリューム名の付け方のヒントについては、45 ページの「ボリューム名」を参照してください。

Solaris ボリュームマネージャ要素を作成するための前提条件

Solaris ボリュームマネージャ要素を作成するための前提条件は、次のとおりです。

- 初期状態データベースの複製を作成します。作成手順については、64 ページの「状態データベースの複製の作成」を参照してください。
- Solaris ボリュームマネージャで使用できるスライスを特定します。必要に応じて、`format` コマンド、`fmthard` コマンド、または Solaris 管理コンソールを使って、既存ディスクのパーティションを再分割します。
- ルート権限を持っていることを確認します。
- 現在のデータをバックアップします。

- グラフィカルユーザーインターフェースを使用する場合は、Solaris 管理コンソールを起動し、メニューから Solaris ボリュームマネージャ機能を選択します。詳細については、40 ページの「Solaris ボリュームマネージャグラフィカルユーザーインターフェースを使用するには」を参照してください。

第 4 章

Solaris ボリュームマネージャの構成と使用

『Solaris ボリュームマネージャの管理』では、可能なかぎり、1つの記憶装置構成に基づいた使用例を示しながら、説明を進めます。この章では、その構成について説明し、以降の章で説明する記憶装置構成のシナリオに関する情報を提供します。

この章では、以下の内容について説明します。

- 51 ページの「背景情報」
- 52 ページの「Solaris ボリュームマネージャ構成の詳細」

背景情報

このマニュアルに示されているさまざまなシナリオや多数の使用例は、1つの構成に基づいています。この構成自体は説明を簡単にするために小規模なものになっていますが、その概念はより大規模な記憶装置環境にも適用できます。

ハードウェア構成

ハードウェアシステムは、次のように構成されているとします。

- 物理的に分離された3つのコントローラが使用されている (c0 — IDE、c1—SCSI、c2 — SCSI)。
- 各 SCSI コントローラは、6つの内臓 9 G バイトディスク (c1t1 から c1t6 と c2t1 から c2t6) を持つ MultiPack に接続されている。
- 個々のコントローラ/ターミネータペア (cn tn) の使用可能な記憶容量は 8.49 G バイトです。
- ルート (/) ドライブ c0t0d0 の記憶領域は6つのパーティションに分割されている。

この構成を図に示すと次のようになります。

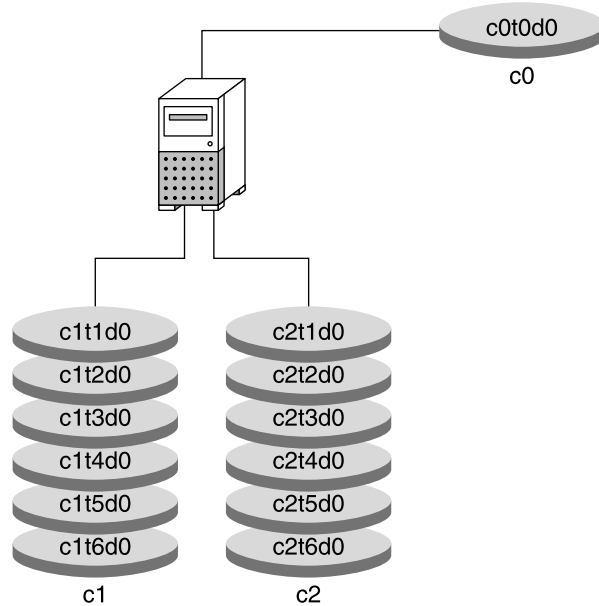


図 4-1 基本的なハードウェア構成

記憶領域の構成

Solaris ボリュームマネージャを構成する前の記憶領域の構成は次のとおりです。

- SCSI コントローラ/ターミネータペア ($c\ ntn$) の記憶容量はおよそ 20G バイトである。
- 各ディスク (たとえば、 $c1t1d0$) の記憶領域は、7つのパーティションに分割されている ($cnt\ nd0s0$ から $cnt\ nd0s6$)。

ディスクをパーティション分割するには、『Solaris のシステム管理 (基本編)』の「ディスクのフォーマット」の手順に従ってください。

Solaris ボリュームマネージャ構成の詳細

このマニュアルでは、作業ごとに特定のシナリオが用意されています。しかし、このマニュアルで示されている使用例を容易に理解できるように、最終的な構成 (-p オプションを指定した `metastat (1M)` コマンドの出力) を以下に示します。

```

[root@lexicon:/]$ metastat -p
d50 -r c1t4d0s5 c1t5d0s5 c2t4d0s5 c2t5d0s5 c1t1d0s5 c2t1d0s5 -k -i 32b
d1 1 1 c1t2d0s3
d2 1 1 c2t2d0s3
d12 1 1 c1t1d0s0
d13 1 1 c2t1d0s0
d16 1 1 c1t1d0s1
d17 1 1 c2t1d0s1
d25 2 2 c1t1d0s3 c2t1d0s3 -i 32b \
    1 c0t0d0s3
d31 1 2 c1t4d0s4 c2t4d0s4 -i 8192b
d80 -p d70 -o 1 -b 2097152
d81 -p d70 -o 2097154 -b 2097152
d82 -p d70 -o 4194307 -b 2097152
d83 -p d70 -o 6291460 -b 2097152
d84 -p d70 -o 8388613 -b 2097152
d85 -p d70 -o 10485766 -b 2097152
d70 -m d71 d72 1
d71 3 1 c1t3d0s3 \
    1 c1t3d0s4 \
    1 c1t3d0s5
d72 3 1 c2t3d0s3 \
    1 c2t3d0s4 \
    1 c2t3d0s5
d123 -p c1t3d0s6 -o 1 -b 204800
d124 -p c1t3d0s6 -o 204802 -b 204800
d125 -p c1t3d0s6 -o 409603 -b 204800
d126 -p c1t3d0s7 -o 3592 -b 20480
d127 -p c2t3d0s7 -o 3592 -b 1433600
hsp010
hsp014 c1t2d0s1 c2t2d0s1
hsp050 c1t2d0s5 c2t2d0s5
hsp070 c1t2d0s4 c2t2d0s4

```


第 5 章

状態データベース (概要)

この章では、状態データベースの複製の概念について説明します。関連する作業の実行手順については、第 6 章を参照してください。

この章では、以下の内容について説明します。

- 55 ページの「Solaris ボリュームマネージャの状態データベースと状態データベースの複製について」
- 57 ページの「多数決アルゴリズムとは」
- 58 ページの「状態データベースの複製を定義するための背景情報」
- 59 ページの「状態データベースの複製のエラー処理」

Solaris ボリュームマネージャの状態データベースと状態データベースの複製について

Solaris ボリュームマネージャの状態データベースには、すべてのボリュームや、ホットスペア、ディスクセットの構成と状態に関する情報が格納されています。Solaris ボリュームマネージャは、冗長性を確保し、システムクラッシュ時のデータの損失を防止するために、複数の状態データベースのコピー (複製) を保持しています (データクラッシュ時に損傷を受けるデータベースのコピーはせいぜい 1 つです)。

状態データベースの複製は、状態データベースのデータが常に有効であることを保証します。状態データベースが更新されると、個々の状態データベースの複製も更新されます。ただし、システムクラッシュによってすべての更新が失われるのを防ぐために、更新は一度に 1 つずつ行われます。

1 つの状態データベースの複製が失われると、Solaris ボリュームマネージャは、どの状態データベースの複製に有効なデータが格納されているかを判断する必要があります。そのために、Solaris ボリュームマネージャは多数決アルゴリズムを使用します。

このアルゴリズムでは、過半数 (半数 + 1) の複製が使用可能であり、一致していれば、それらの複製を有効であるとみなします。このアルゴリズムでは、ディスク構成を設定するときに、3 つ以上の状態データベースの複製を作成する必要があります。3 つの状態データベースの複製のうち少なくとも 2 つが有効であれば、コンセンサスが得られたことになります。

起動時には、Solaris ボリュームマネージャは、損傷した状態データベースの複製を無視します。場合によっては、損傷した複製を Solaris ボリュームマネージャが作成し直すこともあります。そうでなければ、そのような複製は管理者が修正するまで無視されます。使用しているスライスに障害が発生して複製が損傷した場合は、スライスを修理または交換してから複製を有効にする必要があります。



注意 - ファブリックに接続された記憶領域、SAN などの、システムに直接接続されていない記憶領域に、複製を格納することはできません。複製は、従来の SCSI または IDE ドライブと同じ起動プロセス点で使用できる記憶領域に格納しなければなりません。

すべての状態データベースの複製が失われると、理論的には、Solaris ボリュームマネージャのボリュームに格納されているすべてのデータが失われます。そのため、十分な数の複製を別々のドライブとコントローラに分散させて作成し、最悪の事態を回避するようにします。さらに、最初の Solaris ボリュームマネージャ構成情報とディスクパーティション情報を保存しておくのも良い方法です。

状態データベースの複製をシステムに追加する方法や失われた状態データベースの複製を回復する方法については、第 6 章を参照してください。

状態データベースの複製は、RAID 1 ボリュームの再同期領域でも使用されます。ミラーの数に比べて状態データベースの複製の数が少なすぎると、複製の入出力が RAID 1 ボリュームの性能に影響を与えることがあります。ミラーの数が多い場合は、RAID 1 ボリューム当たり少なくとも 2 つの状態データベースの複製 (ディスクセット当たりの複製数の最大数は 50) を用意してください。

個々の状態データベースの複製には、デフォルトで 4M バイト (8192 ディスクセクタ) のディスク領域が使用されます。複製は、次のデバイスに格納できます。

- 専用のローカルディスクパーティション
- ボリュームの一部となるローカルパーティション
- UFS ロギングデバイスの一部となるローカルパーティション

注 - 複製は、ルート (/)、swap、/usr スライス、およびファイルシステムやデータがすでに格納されているスライスには格納できません。ただし、複製を格納した後で、同じスライスにボリュームやファイルシステムを置くことができます。

注 - ファブリックに接続された記憶領域、SAN などの、システムに直接接続されていない記憶領域に、複製を格納することはできません。複製は、従来の SCSI または IDE ドライブと同じ起動プロセス点で使用できる記憶領域に格納しなければなりません。

多数決アルゴリズムとは

複製されたデータベースには、どのデータベースが有効で正しいデータを格納しているかを判断しなければならないという特有の問題があります。この問題を解決するために、Solaris ポリウムマネージャでは多数決アルゴリズムが使用されます。このアルゴリズムでは、過半数の複製のコンセンサスが得られないかぎり、いずれの複製も有効なものみなされません。したがって、このアルゴリズムでは、始めから 3 つ以上の複製が存在していなければなりません。3 つの複製のうち少なくとも 2 つが使用可能であれば、コンセンサスが得られたこととなります。一方、複製が 1 つしか存在していないときに、システムがクラッシュすると、すべてのポリウム構成データが失われてしまう可能性があります。

データを保護するために、Solaris ポリウムマネージャは、半数の複製が使用可能でなければ、動作しません。これにより、データの損傷が防止されます。

多数決アルゴリズムによって、システムは多数決アルゴリズムに従って次のように動作します。

- システムは、少なくとも半数の複製が使用可能であれば、動作する
- システムは、使用可能な複製が半数を下回ると、パニックを起こす
- システムは、過半数の複製が使用可能でなければ、マルチユーザーモードで再起動できない

使用可能な状態データベースの複製の数が足りない場合は、シングルユーザーモードで起動し、不良または失われた複製を削除して、規定数を満たす有効な複製を確保する必要があります。詳細は、282 ページの「状態データベースの複製数の不足から回復するには」を参照してください。

注 - 状態データベースの複製の数が奇数の場合は、その値を 2 で割り、端数を切り捨てた整数値に 1 を加えることによって過半数値が計算されます。たとえば、複製が 7 つあるシステムの過半数値は 4 です (7 を 2 で割って端数を切り捨てると 3 になり、それに 1 を足すと 4 になる)。

状態データベースの複製を定義するための背景情報

通常は、状態データベースの複製を異なるスライス、ドライブ、コントローラに分散させて、単一点障害を避けるようにするのが最善の方法です。これは、単一のコンポーネントに障害が発生した場合でも、大半の複製を利用可能な状態に保つ必要があるからです。デバイス障害などによって複製が失われると、Solaris ボリュームマネージャの動作やシステムの再起動に問題が生じることがあります。Solaris ボリュームマネージャが動作するためには、少なくとも半数の複製が有効でなければならず、システムをマルチユーザーモードで再起動するためには過半数 (半数+1) の複製が有効でなければなりません。

状態データベースの複製を使用するときは、58 ページの「状態データベースの複製に関する推奨事項」と 59 ページの「状態データベースの複製に関する指針」を考慮してください。

状態データベースの複製に関する推奨事項

- 状態データベースの複製は、4M バイト以上の容量を持つ専用スライス上に作成します。必要であれば、状態データベースの複製を、RAID 0 / RAID 1 / RAID 5 ボリューム、ソフトパーティション、またはトランザクション (マスターまたはログ) ボリュームの一部として使用されるスライス上に作成することもできます。ただし、その場合は、スライスをボリュームに追加する前に複製を作成する必要があります。Solaris ボリュームマネージャは、スライスの先頭部分を状態データベースの複製用に予約しています。
- 状態データベースの複製は、未使用のスライス上に作成できます。
- 状態データベースの複製を既存のファイルシステムや、ルート (/)、/usr、swap ファイルシステムに作成することはできません。必要であれば、swap 領域を使用して新しいスライスを作成してから (スライス名が使用可能であるとします)、そのスライスに状態データベースの複製を作成できます。
- Solaris ボリュームマネージャではディスクセット当たり最低 3 つの複製を用意します。また最大 50 の複製を作成できます。複製の格納場所については、次の指針を考慮してください。
 - ドライブが 1 つだけのシステムでは、3 つの複製すべてを 1 つのスライスに置く
 - ドライブの数が 2 から 4 のシステムでは、各ドライブに 2 つずつ複製を置く
 - ドライブの数が 5 つ以上のシステムでは、各ドライブに 1 つずつ複製を置く
- データベースのように、小容量のランダム入出力に RAID 1 ボリュームを使用する場合は、RAID 1 ボリュームごとに、その RAID 1 ボリュームに接続されていない複数のスライス (および、可能であれば複数のディスクとコントローラ) 上に 2 つ以上の複製を余分に作成します。これは、最適な性能を得るために必要な作業で

す。

状態データベースの複製に関する指針

- 状態データベースの複製は、いつでもシステムに追加できます。状態データベースの複製を追加すると、Solaris ボリュームマネージャの可用性が向上します。



注意 – Solstice DiskSuite™から Solaris ボリュームマネージャにアップグレードしたときに、スライスが、状態データベースの複製とファイルシステムまたは論理ボリュームの間で共有されている (それぞれが異なるスライス上に置かれていない) 場合は、既存の複製を削除して、同じ場所に新しいデフォルトの複製を作成しないでください。

Solaris ボリュームマネージャの状態データベースの複製のデフォルトサイズは 8192 ブロックですが、Solstice DiskSuite のデフォルトサイズは 1034 ブロックです。Solstice DiskSuite のデフォルトサイズの状態データベースの複製を削除し、Solaris ボリュームマネージャでデフォルトサイズの新しい複製を追加すると、共有スライスの残りの部分を占めているファイルシステムの先頭の 7158 ブロックが上書きされ、データが破壊されてしまいます。

- ボリュームの一部となるスライス上に状態データベースの複製が置かれている場合、ボリュームの容量は、複製によって占有される領域分だけ少なくなります。複製が占める領域はシリンダ単位で切り上げられるため、この領域はボリュームによってスキップされます。
- 状態データベースの複製のデフォルトサイズは 4M バイト (8192 ディスクブロック) です。ディスクスライスのサイズがこれより大きい場合は、状態データベースの複製を格納できるように、スライスのサイズを変更できます。スライスサイズの変更については、『Solaris のシステム管理 (基本編)』の「ディスクの管理 (手順)」を参照してください。
- 複数のコントローラが存在する場合は、複製をすべてのコントローラ上にできるだけ均一に分散するようにします。これによって、コントローラ障害に対する冗長性が確保できるだけでなく、負荷の分散も可能になります。同じコントローラ上に複数のディスクが存在する場合は、各コントローラで 2 つ以上のディスクに複製を配置します。

状態データベースの複製のエラー処理

Solaris ボリュームマネージャにおける不良複製の処理方法。

システムは、少なくとも半数の複製が使用可能であれば動作を続けますが、使用可能な複製が半数を下回ると、パニックを起します。

システムは、過半数 (半数 + 1) が使用可能であれば、マルチユーザーモードで再起動できます。使用できる複製が過半数に満たない場合は、システムをシングルユーザーモードで再起動し、metadb コマンドを使って使用不能な複製を削除する必要があります。

たとえば、4つの複製を使用しているとします。システムは、2つの複製 (半数) が使用可能であれば動作を続けます。しかし、システムをマルチユーザーモードで再起動するためには、3つの複製 (半数 + 1) が使用可能でなければなりません。

ディスクが2台の構成では、各ディスクに必ず2つ以上の複製を作成します。たとえば、ディスクが2台の構成で複製を3つしか作成しないとします (一方のディスクに2つの複製、他方のディスクに1つの複製を配置する)。この場合、2つの複製が置かれているディスクに障害が発生すると、システムは停止します。これは、残りのディスクには1つの複製しかなく、複製の数が半数に満たないからです。

注 - ディスクが2台の構成で各ディスクに2つずつ複製を作成すれば、一方のディスクに障害が発生しても、Solaris ボリュームマネージャは動作を続けます。しかし、システムの再起動には過半数の複製が必要なため、システムを再起動することはできません。

状態データベースの複製を格納するスライスに障害が発生した場合。

構成の残りの部分は正常に機能するはずですが、Solaris ボリュームマネージャは、起動時に、過半数の状態データベースの複製が使用可能であれば、有効な複製を探します。

状態データベースの複製を修復した場合。

状態データベースの複製を手動で修復して使用可能にすると、Solaris ボリュームマネージャは有効なデータを使ってその複製を更新します。

シナリオ — 状態データベースの複製

状態データベースの複製は、Solaris ボリュームマネージャ構成全体に対してデータ冗長性を提供します。第4章のサンプルシステムに基づく構成例は、状態データベースの複製をどのように分散すれば適切な冗長性が得られるかを示しています。

サンプルシステムには1つの内蔵 IDE コントローラとドライブ、さらに2台の SCSI コントローラがあり、SCSI コントローラにはそれぞれ6つのディスクが接続されています。システムには3つのコントローラがあるため、システムを適切に構成することによって単一点障害を防止できます。Solaris ボリュームマネージャでは、2つのコントローラしかないシステムで単一点障害を防止することはできません。複製を3つのコントローラすべてに (各コントローラの少なくとも1つ (可能であれば2つ) のディスクに) 均一に分散すれば、システムはどのようなハードウェアの単一点障害にも耐えることができます。

最小限の構成では、1つの状態データベースの複製をルートディスクのスライス7に置き、追加の複製を他の2つの各コントローラの1つのディスク上のスライス7に置くことができます。媒体障害が発生する可能性は極めて低いですが、十分な安全性を求めるなら、ルートディスクに2つの複製を置き、各コントローラの2つのディスクにそれぞれ複製を置くようにします(合計で6つの複製)。

さらに完全を求めるなら、6つのミラーそれぞれに対してミラーとは異なる6つのディスクに複製を2つずつ追加します。これによって、複製の数は、ルートディスクに2つ、各SCSIコントローラに8つずつで合計18になります。これらの複製は、各コントローラのディスクに分散されています。

第 6 章

状態データベース (作業)

この章では、Solaris ボリュームマネージャの状態データベースの複製に関連する作業について説明します。これらの作業に伴う概念については、第 5 章を参照してください。

状態データベースの複製 (作業マップ)

次の表に、Solaris ボリュームマネージャの状態データベースの複製を管理するのに必要な作業を示します。

作業	説明	参照先
状態データベースの複製の作成	Solaris ボリュームマネージャの GUI か <code>metadb -a</code> コマンドを使って状態データベースの複製を作成します。	64 ページの「状態データベースの複製を作成するには」
状態データベースの複製の状態チェック	Solaris ボリュームマネージャの GUI か <code>metadb -a</code> コマンドを使って、既存の複製の状態をチェックします。	66 ページの「状態データベースの複製の状態をチェックするには」
状態データベースの複製の削除	Solaris ボリュームマネージャの GUI か <code>metadb -a</code> コマンドを使って状態データベースの複製を削除します。	67 ページの「状態データベースの複製を削除するには」

状態データベースの複製の作成



注意 – Solstice DiskSuite™から Solaris ボリュームマネージャにアップグレードしたときに、スライスが、状態データベースの複製とファイルシステムまたは論理ボリュームの間で共有されている (それぞれが異なるスライス上に置かれていない) 場合は、既存の複製を削除して、同じ場所に新しいデフォルトの複製を作成しないでください。

Solaris ボリュームマネージャの状態データベースの複製のデフォルトサイズは 8192 ブロックですが、Solstice DiskSuite のデフォルトサイズは 1034 ブロックです。Solstice DiskSuite のデフォルトサイズの状態データベースの複製を削除し、Solaris ボリュームマネージャでデフォルトサイズの新しい複製を追加すると、共有スライスの残りの部分を占めているファイルシステムの先頭の 7158 ブロックが上書きされ、データが破壊されてしまいます。



注意 – ファブリックに接続された記憶領域、SAN などの、システムに直接接続されていない記憶領域に、複製を格納することはできません。複製は、従来の SCSI または IDE ドライブと同じ起動プロセス点で利用できる記憶領域に格納しなければなりません。

▼ 状態データベースの複製を作成するには

1. 49 ページの「Solaris ボリュームマネージャ要素を作成するための前提条件」をチェックします。
2. 次のどちらかの方法で状態データベースの複製を作成します。
 - Solaris 管理コンソール内の「拡張ディスク」から「状態データベースの複製」ノードを開きます。「アクション (Action)」、「複製の作成 (Create Replicas)」の順に選択し、画面の指示に従って複製を作成します。詳細は、オンラインヘルプを参照してください。
 - 次の形式の `metadb` コマンドを実行します。詳細は、`metadb (1M)` のマニュアルページを参照してください。

```
metadb -a -c n -l mmmn -f ctds-of-slice
```

- `-a` は、状態データベースの複製を追加します。
- `-f` は、複製が存在しなくても強制的に操作を実行します。
- `-c n` には、指定したスライスに追加する複製の数を指定します。
- `-l mmmn` には、新しい複製のサイズをブロック数で指定します。
- `ctds-of-slice` には、複製を格納するコンポーネントの名前を指定します。

最初の複製を強制的に追加するには、`-f` フラグを指定します。

注 - オプションを指定しないと、`metadb` コマンドはすべての複製の状態を表示します。

例 — 最初の状態データベースの複製を作成する

```
# metadb -a -f c0t0d0s7
# metadb
      flags          first blk      block count
...
      a      u          16          8192          /dev/dsk/c0t0d0s7
```

`-a` オプションは状態データベースの複製をシステムに追加し、`-f` オプションは最初の複製を強制的に作成します。2 番目以後の複製を追加するときは、`-f` オプションは省略できます。

例 — 2つの状態データベースの複製を同じスライスに追加する

```
# metadb -a -c 2 c1t3d0s1
# metadb
      flags          first blk      block count
...
      a      u          16          8192          /dev/dsk/c1t3d0s1
      a      u          8208         8192          /dev/dsk/c1t3d0s1
```

`-a` オプションは、状態データベースの複製をシステムに追加します。`-c 2` オプションは、指定したスライスに2つの複製を格納します。`metadb` コマンドは、これらの複製が有効であるかどうかをチェックします (a フラグで示される)。

また、`-l` オプションでブロック数を指定すると、状態データベースの複製のサイズを指定できます。ただし、デフォルトのサイズ (8192) は、数千の論理ボリュームを持つ構成を含め、事実上すべての構成に適しています。

例 — 指定したサイズの状態データベースの複製を追加する

既存の状態データベースの複製を置き換える場合は、複製のサイズを指定しなければならない場合があります。特に、ファイルシステムとスライスを共有している状態データベースの複製がすでに存在している場合は (たとえば、Solstice DiskSuite からアップグレードした場合など)、既存の複製を同じサイズの複製で置き換えるか、別の場所に新しい複製を作成する必要があります。



注意 – Solstice DiskSuite のデフォルトサイズ (1034 ブロック) の状態データベースの複製がファイルシステムとスライスを共有している場合は、その複製を Solaris ポリウムマネージャのデフォルトサイズの複製で置き換えないでください。置き換えると、新しい複製がファイルシステムの先頭の部分を上書きしてしまうため、データが破壊されてしまいます。

```
# metadb -a -c 3 -l 1034 c0t0d0s7
# metadb
      flags          first blk      block count
...
a    u              16              1034          /dev/dsk/c0t0d0s7
a    u             1050              1034          /dev/dsk/c0t0d0s7
a    u             2084              1034          /dev/dsk/c0t0d0s7
```

-a オプションは、状態データベースの複製をシステムに追加します。-l オプションは、追加する複製のサイズをブロック数で指定します。

状態データベースの複製の保守

▼ 状態データベースの複製の状態をチェックするには

- 次のどちらかの方法で状態データベースの複製の状態をチェックします。
 - Solaris 管理コンソール内の「拡張ディスク」から「状態データベースの複製 (State Database Replicas)」ノードを開いて、存在するすべての状態データベースの複製の状態を表示します。詳細は、オンラインヘルプを参照してください。
 - metadb コマンドを実行して、状態データベースの複製の状態を表示します。-i オプションを指定すると、複製の状態と、すべての状態フラグについての説明が表示されます。詳細は、metadb(1M) のマニュアルページを参照してください。

例 — すべての状態データベースの複製の状態をチェックする

```
# metadb -i
      flags          first blk      block count
a m p  lu0         16              8192          /dev/dsk/c0t0d0s7
a    p  lu0         8208              8192          /dev/dsk/c0t0d0s7
a    p  lu0        16400              8192          /dev/dsk/c0t0d0s7
```

```

a p luo      16          8192      /dev/dsk/c1t3d0s1
W p l        16          8192      /dev/dsk/c2t3d0s1
a p luo      16          8192      /dev/dsk/c1t1d0s3
a p luo      8208        8192      /dev/dsk/c1t1d0s3
a p luo      16400       8192      /dev/dsk/c1t1d0s3
r - replica does not have device relocation information
o - replica active prior to last mddb configuration change
u - replica is up to date
l - locator for this replica was read successfully
c - replica's location was in /etc/lvm/mddb.cf
p - replica's location was patched in kernel
m - replica is master, this is replica selected as input
W - replica has device write errors
a - replica is active, commits are occurring to this replica
M - replica had problem with master blocks
D - replica had problem with data blocks
F - replica had format problems
S - replica is too small to hold current data base
R - replica had device read errors

```

状態表示の後にすべてのフラグの説明が表示されます。デバイス名の前の文字はデバイスの状態を示します。大文字は障害状態を、小文字は「Okay」状態を示します。

状態データベースの複製を削除するには

Solaris ポリウムマネージャ構成を保守するために、状態データベースの複製を削除しなければならない場合があります。たとえば、ディスクドライブを交換する場合には、Solaris ポリウムマネージャがエラーと認識しないように、ドライブを取り外す前に状態データベースの複製を削除します。

▼ 状態データベースの複製を削除するには

- 次のどちらかの方法で状態データベースの複製を削除します。
 - Solaris 管理コンソール内の「拡張ディスク」から「状態データベースの複製 (State Database Replicas)」ノードを開いて、存在するすべての状態データベースの複製の状態を表示します。次に、削除する複製を選択してから「編集 (Edit)」、「削除 (Delete)」の順に選択して複製を削除します。詳細は、オンラインヘルプを参照してください。
 - 次の形式の `metadb` コマンドを使用します。

```
metadb -d -f ctds-of-slice
```

- `-d` は、状態データベースの複製を削除します。
- `-f` は、複製が存在しなくても強制的に操作を実行します。
- `ctds-of-slice` には、複製が格納されているコンポーネントの名前を指定します。

削除したい状態データベースの複製が格納されているすべてのスライスを指定する必要があります。詳細は、`metadb(1M)` のマニュアルページを参照してください。

例 — 状態データベースの複製を削除する

```
# metadb -d -f c0t0d0s7
```

この例では、スライスから最後の複製を削除します。

注 — システム上にある最後の複製を強制的に削除するには、`-f` オプションを指定する必要があります。

第 7 章

RAID 0 (ストライプ方式および連結方式) ボリューム (概要)

この章では、Solaris ボリュームマネージャで使用できる RAID 0 ボリューム (ストライプ方式と連結方式) について説明します。関連する作業については、第 8 章を参照してください。

この章では、次の内容について説明します。

- 69 ページの「RAID 0 ボリュームの概要」
- 77 ページの「RAID 0 ボリュームを作成するための背景情報」
- 78 ページの「シナリオ — RAID 0 ボリューム」

RAID 0 ボリュームの概要

RAID 0 ボリュームは、ストライプ方式と連結方式のどちらも、スライスまたはソフトウェアパーティションから構成され、ディスクの記憶容量を拡張するために使用されます。RAID 0 ボリュームは、そのまま使用することも、RAID 1 (ミラー) ボリュームや、トランザクションボリューム、ソフトウェアパーティションの基本構成ブロックとして使用することもできます。RAID 0 ボリュームには、次の 3 種類があります。

- ストライプ方式ボリューム (または ストライプ)
- 連結方式ボリューム (または 連結)
- ストライプ方式ボリュームの連結 (または ストライプの連結)

注 - コンポーネントとは、別の論理ボリュームの中で使用されるすべてのデバイス (スライスからソフトウェアパーティションまで) を意味します。

ストライプ方式ボリュームは、データをストライプのすべてのコンポーネントに均等に分散しますが、連結方式ボリュームはデータを、使用可能な最初のコンポーネントに書き込み、そのコンポーネントが満杯になると、次の使用可能なコンポーネントに書き込みます。ストライプ方式ボリュームの連結は、単に、コンポーネントの追加によって元の構成が拡張されたストライプ方式ボリュームです。

RAID 0 ボリュームでは、ディスクの記憶容量をすばやく簡単に拡張できます。ただし、RAID 1 や RAID 5 ボリュームとは異なり、RAID 0 ボリュームにはデータの冗長性はありません。したがって、RAID 0 ボリュームのコンポーネントに 1 つでも障害が発生すると、データは失われます。

1 つのスライスを含む RAID 0 ボリュームは、任意のファイルシステムに使用できません。

複数のコンポーネントを含む RAID 0 ボリュームは、次のファイルシステムを除く、任意のファイルシステムに使用できます。

- ルート (/)
- /usr
- swap
- /var
- /opt
- オペレーティングシステムのアップグレードやインストール時にアクセスされるファイルシステム

注 - ルート (/)、/usr、swap、/var、/opt のミラーを作成する場合は、このファイルシステムをサブミラーとして機能する、1 面の連結またはストライプ (1 つのスライスで構成される単純連結) に置きます。そして、この 1 面の連結 (サブミラー) を別のサブミラーでミラー化します。このサブミラーも連結でなければなりません。

RAID 0 (ストライプ方式) ボリューム

RAID 0 (ストライプ方式) ボリュームは、データを 1 つまたは複数のコンポーネント上に分散させたボリュームです。ストライプ方式では、同じサイズのデータセグメントが 2 つ以上のコンポーネントに順に配置され、1 つの論理記憶ユニットが構成されます。これらのセグメントはラウンドロビン (巡回的な) 方式でインターリーブされ、領域は各コンポーネントからメタデバイスに交互に割り当てられます。

ストライプ方式では、複数のコントローラがデータに同時にアクセスできます (並列アクセス)。並列アクセスではボリュームのほとんどのディスクが入出力要求の処理でビジーになるため、入出力スループットが向上します。

既存のファイルシステムをストライプに直接変換することはできません。既存のファイルシステムをストライプに置くためには、ファイルシステムのバックアップをとり、ストライプを作成してから、ファイルシステムをストライプに復元する必要があります。

ストライプ方式で順次入出力操作を行うと、Solaris ボリュームマネージャは、先頭のコンポーネントからブロックセグメント (飛び越しと呼びます) 1つ分のブロックを読み取り、次に2番目のコンポーネントのブロックセグメント1つ分のブロックを読み取るという処理を繰り返します。

連結方式の順次入出力操作では、Solaris ボリュームマネージャは、先頭のコンポーネントからすべてのブロックを最初に読み取り、次に2番目のコンポーネントのすべてのブロックを読み取るという処理を繰り返します。

連結方式でもストライプ方式でも、すべての入出力は並列に実行されます。

ストライプ方式の飛び越し値

飛び越し値は、ストライプ上の論理データセグメントのサイズに等しく、K バイト、M バイト、またはブロック数で表わされます。アプリケーションによっては、飛び越し値を変えることによって性能が向上することがあります。性能の向上は、入出力要求をいくつかのディスクアームを使って処理することによって達成されます。性能の向上が期待できるのは、入出力要求が飛び越し値よりも大きい場合です。

注 - RAID 5 ボリュームも飛び越し値を使用します。詳細は、139 ページの「RAID 5 ボリュームの概要」を参照してください。

飛び越し値は、ストライプを作成するときに設定できます。あるいは、Solaris ボリュームマネージャのデフォルト値である 16K バイトを使用することもできます。ただし、ストライプを作成した後で飛び越し値を変更することはできません。飛び越し値を変更するときは、データのバックアップをとり、ストライプを削除し、新しい飛び越し値で新しいストライプを作成してから、データを復元します。

シナリオ — RAID 0 (ストライプ方式) ボリューム

図 7-1 に、3つのコンポーネント (ディスク) からなるストライプ方式ボリュームの例を示します。

Solaris ボリュームマネージャは、このボリュームのデータを各コンポーネントに書き込む場合、チャンク 1 のデータをディスク A、チャンク 2 のデータをディスク B、チャンク 3 のデータをディスク C にそれぞれ書き込みます。次に、チャンク 4 のデータをディスク A、チャンク 5 のデータをディスク B、チャンク 6 のデータをディスク C にそれぞれ書き込み、同じ処理を繰り返します。

飛び越し値は各チャンクのサイズと同じ値に設定されています。このストライプ方式 d2 の合計容量は、最小コンポーネントのサイズにコンポーネント数を掛けた値です。(次の例に示す各スライスのサイズが 2G バイトであれば、d2 は 6G バイトです)。

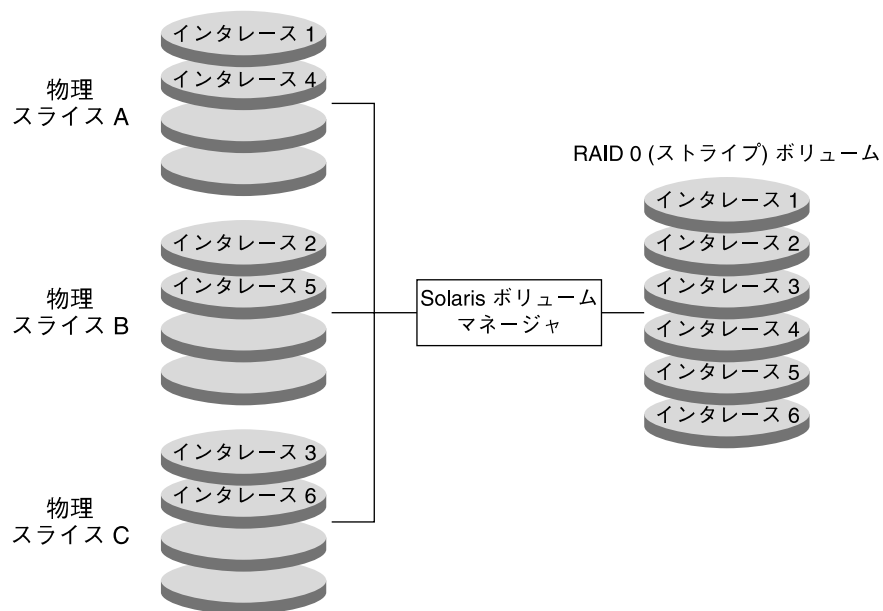


図 7-1 ストライプ方式の例

RAID 0 (連結方式) ボリューム

連結方式ボリューム (または単に連結) は、個々のコンポーネント内にデータを順番に隣接して配置し、1つの論理記憶ユニットを構成します。

連結方式では、いくつかのコンポーネントの容量を結合することによって記憶容量を拡張します。したがって、記憶容量の要件に応じてコンポーネントを追加できます。

連結方式では、記憶容量やファイルシステムのサイズをオンライン状態のまま動的に拡張できます。連結方式ボリュームにコンポーネントを追加するときは、他のコンポーネントがアクティブであってもかまいません。

注 - ストライプの容量を拡張するためには、ストライプを連結する必要があります。詳細は、74 ページの「RAID 0 (ストライプ方式の連結) ボリューム」を参照してください。

また、連結方式では、システムを停止しなくても、動作中の、マウントされている UFS ファイルシステムを拡張できます。通常、連結方式ボリュームの合計容量は、すべてのコンポーネントの合計サイズと同じです。ただし、このボリュームに状態データベースの複製を格納するスライスが含まれている場合には、連結の合計容量は、コンポーネントの合計から複製に予約されている領域を引いたものです。

連結方式ボリュームは、1つのコンポーネントから作成することもできます。後で記憶容量が必要になったら、新たにコンポーネントを追加することができます。

注 - ルート (/)、swap、/usr、/opt、または /var ファイルシステムをミラー化する場合、連結方式を使ってこれらのファイルシステムをカプセル化する必要があります。

シナリオ — RAID 0 (連結方式)

図 7-2 に、3つのコンポーネント (スライス) からなる連結の例を示します。

データブロック (チャンク) は、ディスク A から始まる個々のコンポーネントに順次、書き込まれます。したがって、ディスク A には論理チャンク 1 から 4 が、ディスク B には論理チャンク 5 から 8 が、ディスク C には論理チャンク 9 から 12 がそれぞれ書き込まれます。このボリュームの合計容量は3つのドライブの合計容量です。したがって、各ドライブの容量が 2G バイトであれば、合計容量は 6G バイトになります。

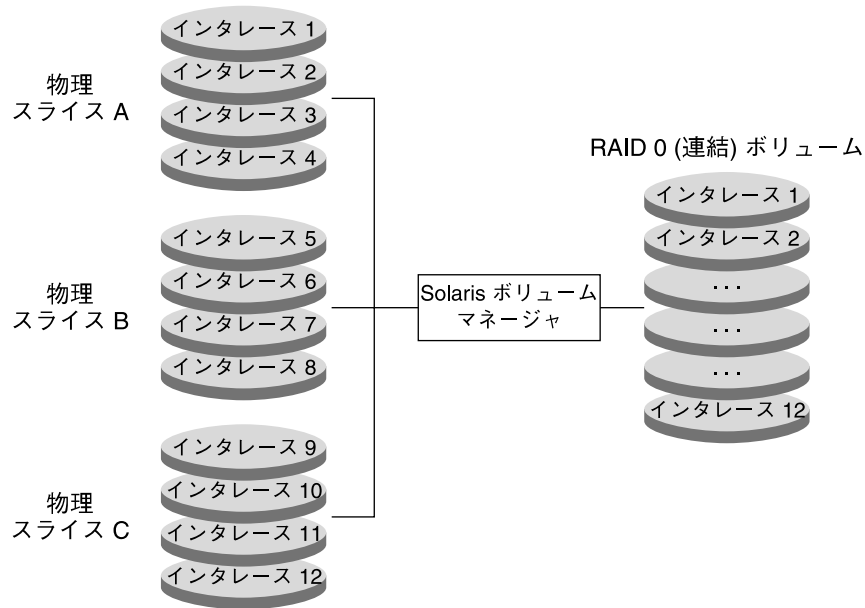


図 7-2 連結の例

RAID 0 (ストライプ方式の連結) ボリューム

ストライプ方式の連結は、コンポーネント (ストライプ) の追加によって容量が拡張されるストライプです。

ストライプ方式の連結の飛び越し値をストライプレベルで設定する場合は、Solaris 管理コンソール内の「拡張ディスク」か `metattach -i` コマンドを使用します。ストライプ方式の連結の各ストライプには、別々の飛び越し値を設定することができます。ストライプ方式の連結を新たに作成する場合は、特定のストライプに飛び越し値を指定しないと、直前のストライプの飛び越し値が使用されます。

例 —RAID 0 (ストライプ方式の連結) ボリューム

図 7-3 に、3 つのストライプを連結した `d10` の例を示します。

最初のストライプは 3 つのスライス (A から C) から構成され、飛び越し値は 16K バイトです。2 つめのストライプは 2 つのスライス (D と E) から構成され、飛び越し値は 32K バイトです。最後のストライプは 2 つのスライス (F と G) から構成されています。このストライプには飛び越し値が指定されていないため、その前のストライプの飛び越し値 (32K バイト) が継承されます。データチャンクはまず最初のストライプに順次、書き込まれます。このストライプが満杯になると、チャンクは 2 つめのストライプ

イプに書き込まれます。さらに、このストライプが満杯になると、チャンクは3つめのストライプに書き込まれます。各ストライプでは、データチャンクが、指定された飛び越し値に基づいてインタリーブされます。

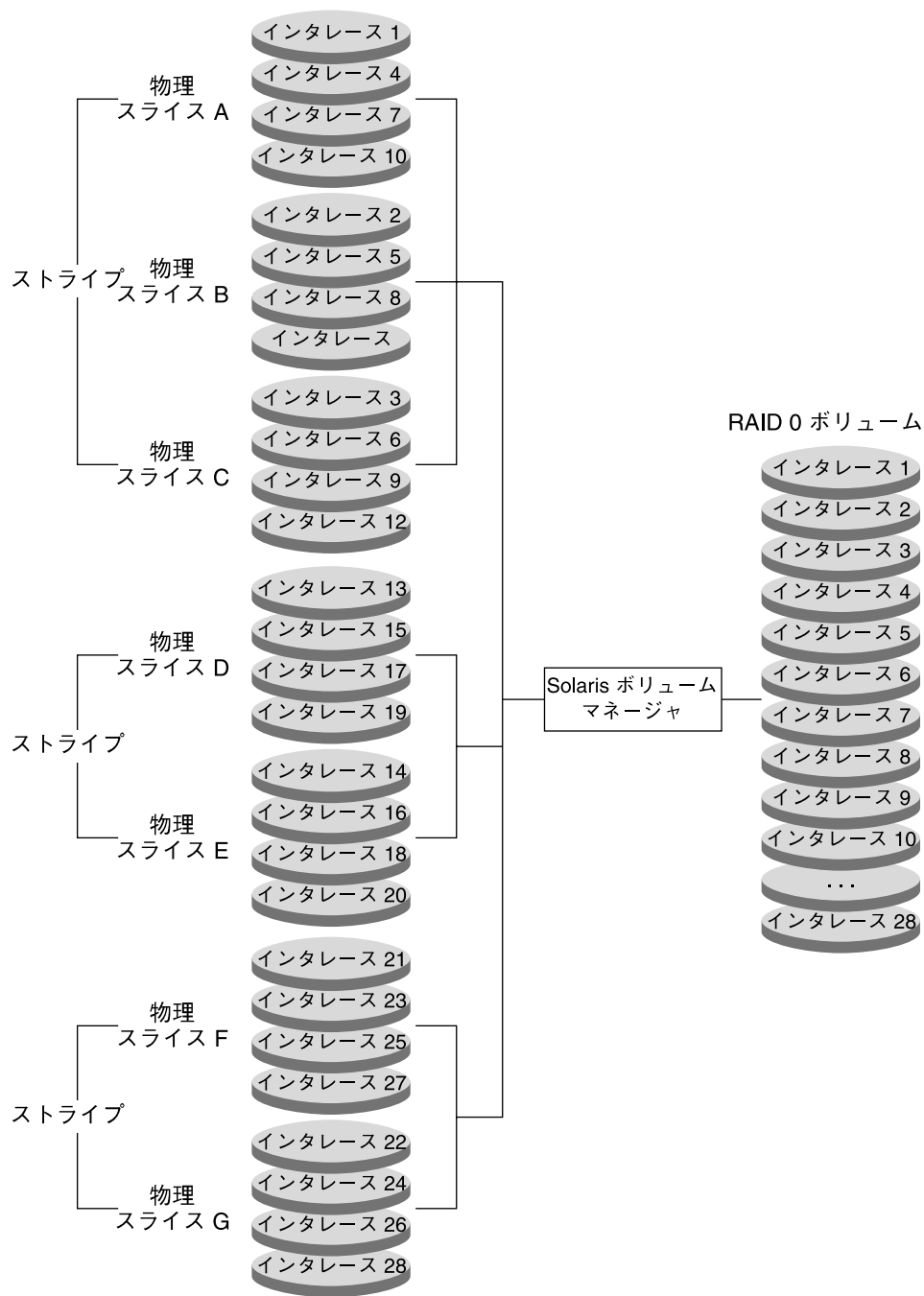


図 7-3 ストライプ方式の連結の例

RAID 0 ボリュームを作成するための背景情報

ストライプ方式と連結方式の前提条件

RAID 0 ボリュームを使用するときは、次のことを考慮してください。

- コンポーネントを異なるコントローラに置くと、同時に実行できる読み取りや書き込みの数が増します。
- 既存のファイルシステムやデータから直接ストライプを作成しないようにします。作成するとデータが破壊されます。代わりに、連結方式を使用します。(既存のデータからストライプを作成することは可能ですが、その場合には、データのバックアップをとり、それをボリュームに復元する必要があります。)
- ストライプには、同じサイズのディスクコンポーネントを使用します。サイズが異なるコンポーネントを使用すると、ディスク領域が無駄になります。
- システムやアプリケーションの入出力要求に合わせてストライプの飛び越し値を設定します。
- ストライプ方式や連結方式には複製データが含まれないため、このようなボリュームのコンポーネントに障害が発生した場合には、そのコンポーネントを交換し、ストライプまたは連結を作成し直してから、バックアップのデータを復元する必要があります。
- ストライプや連結を再作成する場合には、障害が発生したコンポーネントと同じサイズのコンポーネントを新しいコンポーネントとして使用します。

ストライプ方式および連結方式の指針

- 連結方式は、ストライプ方式ほど CPU サイクルを必要としません。連結方式は、小規模のランダム入出力や均一に分散された入出力に適しています。
- 可能であれば、ストライプ方式や連結方式のコンポーネントを異なるコントローラやバスに配置します。個々のストライプを異なるコントローラに置くと、同時に実行できる読み取りや書き込みの数が増えます。
- ストライプが置かれているコントローラに障害が発生した場合、別のコントローラが使用可能であれば、ディスクをコントローラに移動し、ストライプを再設定することによって、ストライプを新しいコントローラに「移動」できます。
- ディスクの数: ストライプの構成を性能要件に基づいて決める場合もあります。たとえば、特定のアプリケーションで 10.4M バイト/秒の性能が必要だとします。各ディスクの性能がおよそ 4M バイト/秒であれば、ストライプに必要なディスクスピンドルの数は、次の式で計算できます。

10.4 Mbyte/sec / 4 Mbyte/sec = 2.6

この例では、入出力を並列に行なうために3つのディスクが必要です。

シナリオ — RAID 0 ボリューム

RAID 0 ボリュームは、記憶領域の結合やミラーの構築に必要な基本構築ブロックとして使用されます。第4章のサンプルシステムに基づく構成例は、記憶領域の拡張や既存のファイルシステム (ルート (/) を含む) のミラー化に RAID 0 ボリュームをどのように使用できるかを示しています。

サンプルシステムには比較的小さい (9G バイト) ディスク群がありますが、アプリケーションによっては、これよりも大きい記憶領域が必要になる場合があります。領域を拡張する (そして性能を高める) ためには、複数のディスクに渡るストライプを作成する必要があります。たとえば、c1t1d0、c1t2d0、c1t3d0 と c2t1d0、c2t2d0、および c2t3d0 を、これらのディスク全体に渡るスライス 0 として構成できます。これによって、同じコントローラの3つのディスクからなるストライプはおよそ 27G バイトの記憶領域となり、アクセスも速くなります。2番目のコントローラに属する2番目のストライプは冗長性を確保するために使用できません。これについては第10章を、具体的には96ページの「シナリオ — RAID 1 ボリューム (ミラー)」をそれぞれ参照してください。

第 8 章

RAID 0 (ストライプ方式および連結方式) ボリューム (作業)

この章では、RAID 0 ボリュームに関連する作業について説明します。概念については、第 7 章を参照してください。

RAID 0 ボリューム (作業マップ)

次の表に、Solaris ボリュームマネージャの RAID 0 ボリュームを管理するのに必要な作業を示します。

作業	説明	参照先
RAID 0 (ストライプ方式) ボリュームの作成	metainit コマンドを使って新しいボリュームを作成します。	80 ページの「RAID 0 (ストライプ方式) ボリュームを作成するには」
RAID 0 (連結方式) ボリュームの作成	metainit コマンドを使って新しいボリュームを作成します。	81 ページの「RAID 0 (連結方式) ボリュームを作成するには」
記憶領域の拡張	metainit コマンドを使って既存のファイルシステムを拡張します。	83 ページの「既存のデータの領域を拡張するには」
既存のボリュームの拡張	metattach コマンドを使って既存のボリュームを拡張します。	84 ページの「既存の RAID 0 ボリュームを拡張するには」
RAID 0 ボリュームの削除	metaclear コマンドを使ってボリュームを削除します。	86 ページの「ボリュームを削除するには」

RAID 0 (ストライプ方式) ボリュームの作成



注意 – 既存のファイルシステムまたはデータからストライプを作成しないでください。作成するとデータが破壊されます。既存のデータからストライプを作成する場合は、データのバックアップをとり、それをボリュームに復元する必要があります。

▼ RAID 0 (ストライプ方式) ボリュームを作成するには

1. 49 ページの「Solaris ボリュームマネージャ要素を作成するための前提条件」と 77 ページの「RAID 0 ボリュームを作成するための背景情報」を確認します。
2. 次のどちらかの方法でストライプを作成します。
 - Solaris 管理コンソール内の「拡張ディスク」から「ボリューム (Volumes)」ノードを開きます。「アクション (Action)」、「ボリュームを作成 (Create Volume)」の順に選択し、ウィザードの指示に従ってボリュームを作成します。詳細は、オンラインヘルプを参照してください。
 - 次の形式の `metainit` コマンドを使用します。

```
metainit {volume-name} {number-of-stripes} {components-per-stripe}
{component-names...} [-i interlace-value]
```

- `volume-name` は作成するボリュームの名前です。
- `number-of-stripes` には、作成するストライプの数を指定します。
- `components-per-stripe` には、個々のストライプを構成するコンポーネントの数を指定します。
- `component-names` には、使用する個々のコンポーネントの名前を指定します。
- `-iwidth` には、ストライプに使用する飛び越し幅を指定します。

詳細は、次の例と `metainit(1M)` のマニュアルページを参照してください。

例 — 3 つのスライスからなるストライプを作成する

```
# metainit d20 1 3 c0t1d0s2 c0t2d0s2 c0t3d0s2
d20: Concat/Stripe is setup
```


ボリューム d20 は1つのストライプ(数字の1)からなり、ストライプは3つのスライス(数字の3)からなります。このストライプには飛び越し値が指定されていないため、デフォルト値の16Kバイトが使用されます。最後に、ボリュームが設定されたことを示すメッセージが出力されます。

例 — 32K バイトの飛び越し値を持つ2つのスライスからなる RAID 0 (ストライプ方式) ボリュームを作成する

```
# metainit d10 1 2 c0t1d0s2 c0t2d0s2 -i 32k
d10: Concat/Stripe is setup
```

ボリューム d10 は1つのストライプ(数字の1)からなり、ストライプは2つのスライス(数字の2)からなります。-i オプションでは、飛び越し値として32Kバイトを設定します。(飛び越し値は、8Kバイト以上、100Mバイト以下でなければなりません。)最後に、ボリュームが設定されたことを示すメッセージが表示されます。

次の作業

新たに作成したストライプにファイルシステムを作成する方法については、『Solaris のシステム管理 (基本編)』の「ファイルシステムの作成 (手順)」を参照してください。データベースなど、raw デバイスを使用するアプリケーションは、独自の方法で raw デバイスにアクセスできなければなりません。

RAID 0 (連結方式) ボリューム

▼ RAID 0 (連結方式) ボリュームを作成するには

1. 49 ページの「Solaris ボリュームマネージャ要素を作成するための前提条件」と77 ページの「RAID 0 ボリュームを作成するための背景情報」を確認します。
2. 次のどちらかの方法で連結を作成します。
 - Solaris 管理コンソール内の「拡張ディスク」から「ボリューム (Volumes)」ノードを開きます。「アクション (Action)」、「ボリュームを作成 (Create Volume)」の順に選択し、ウィザードの指示に従ってボリュームを作成します。詳細は、オンラインヘルプを参照してください。
 - 次の形式の metainit コマンドを使用します。

```
metainit {volume-name} {number-of-stripes} { [components-per-stripe]
| [component-names] ... }
```

- *volume-name* は、作成するボリュームの名前です。
- *number-of-stripes* には、作成するストライプの数を指定します。
- *components-per-stripe* には、個々のストライプを構成するコンポーネントの数を指定します。
- *component-names* には、使用する個々のコンポーネントの名前を指定します。

詳細は、次の例と `metainit(1M)` のマニュアルページを参照してください。

例 — 1つのスライスからなる連結を作成する

```
# metainit d25 1 1 c0t1d0s2
d25: Concat/Stripe is setup
```

この例では連結方式ボリューム `d25` を作成します。このボリュームは1つのストライプ(最初の 1) からなり、ストライプは1つのスライス(2番目の 1) からなります。最後に、ボリュームが設定されたことを示すメッセージが出力されます。

注 - この例は、既存のデータを安全にカプセル化できる連結の例です。

例 — 4つのスライスからなる連結を作成する

```
# metainit d40 4 1 c0t1d0s2 1 c0t2d0s2 1 c0t2d0s3 1 c0t2d1s3
d40: Concat/Stripe is setup
```

この例では `d40` という名前の連結を作成します。ボリュームは4つの「ストライプ」(数字の 4) からなり、各ストライプは1つのスライス(各スライスの前の数字 1) からなります。最後に、ボリュームが設定されたことを示すメッセージが出力されます。

次の作業

新たに作成した連結にファイルシステムを作成する方法については、『*Solaris* のシステム管理 (基本編)』の「ファイルシステムの作成 (手順)」を参照してください。

記憶領域の拡張

ファイルシステムに領域を追加する場合は連結を作成し、既存のストライプに領域を追加する場合はストライプを連結します。

▼ 既存のデータの領域を拡張するには

1. 49 ページの「Solaris ボリュームマネージャ要素を作成するための前提条件」と 77 ページの「RAID 0 ボリュームを作成するための背景情報」を確認します。

2. ファイルシステムをマウント解除します。

```
# umount /filesystem
```

3. 次のどちらかの方法で連結を作成します。

- Solaris 管理コンソール内の「拡張ディスク」から「ボリューム (Volumes)」ノードを開きます。「アクション (Action)」、「ボリュームを作成 (Create Volume)」の順に選択し、ウィザードの指示に従ってボリュームを作成します。詳細は、オンラインヘルプを参照してください。
- 次の形式の `metainit` コマンドを使用します。

```
metainit {volume-name} {number-of-stripes} { [components-per-stripe] | [component-names] ... }
```

- `volume-name` は、作成するボリュームの名前です。
- `number-of-stripes` には、作成するストライプの数を指定します。
- `components-per-stripe` には、個々のストライプを構成するコンポーネントの数を指定します。
- `components` には、拡張に使用する個々のコンポーネントの名前を指定します。

詳細は、`metainit (1M)` のマニュアルページを参照してください。

4. `/etc/vfstab` を編集して、このファイルシステムが連結の名前を参照するようにします。

5. ファイルシステムを再びマウントします。

```
# mount /filesystem
```

例 — 連結を作成してファイルシステムを拡張する

```
# umount /docs
# metainit d25 2 1 c0t1d0s2 1 c0t2d0s2
d25: Concat/Stripe is setup
(/etc/vfstab ファイルを編集して、このファイルシステムがスライス c0t1d0s2 の代わりにボリューム d25 を参照するようにします。)
# mount /docs
```

この例では、2つのスライス、`/dev/dsk/c0t1d0s2` (`/docs` にマウントされたファイルシステムが格納されている) と `/dev/dsk/c0t2d0s2` から `d25` という名前の連結を作成します。ファイルシステムは、最初にマウント解除する必要があります。



注意 – `metainit` コマンドに指定する最初のスライスは、ファイルシステムが格納されているスライスでなければなりません。そうでないと、データが破壊されます。

次に、`/etc/vfstab` ファイルにあるファイルシステムのエントリを、この連結を参照するように変更します (初めての場合は、入力します)。たとえば、次の行を見てください。

```
/dev/dsk/c0t1d0s2 /dev/rdsk/c0t1d0s2 /docs ufs 2 yes -
```

上記の行を次のように変更します。

```
/dev/md/dsk/d25 /dev/md/rdsk/d25 /docs ufs 2 yes -
```

最後に、ファイルシステムを再びマウントします。

次の作業

UFS ファイルシステムの場合は、連結に対して `growfs` コマンドを実行します。詳細は、243 ページの「ファイルシステムを拡張するには」を参照してください。

データベースなど、`raw` 連結を使用するアプリケーションは、独自の方法でこのボリュームを認識し、領域を拡張できなければなりません。

▼ 既存の RAID 0 ボリュームを拡張するには

ストライプを連結することによって、既存のストライプを拡張できます。たとえば、ストライプの領域が足りなくなった場合は、ストライプを連結することによって、領域を拡張できます。データのバックアップや復元は必要ありません。

この手順では、既存のストライプに別のストライプを追加するものとします。

1. 49 ページの「Solaris ボリュームマネージャ要素を作成するための前提条件」と 77 ページの「RAID 0 ボリュームを作成するための背景情報」を確認します。
2. 次のどちらかの方法でストライプ連結を作成します。
 - Solaris 管理コンソール内の「拡張ディスク」から「ボリューム (Volumes)」ノードを開きます。「アクション (Action)」、「ボリュームを作成 (Create Volume)」の順に選択し、ウィザードの指示に従ってボリュームを作成します。詳細は、オンラインヘルプを参照してください。
 - コマンド行から既存のストライプを連結する場合は、次の形式の `metattach` コマンドを使用します。

```
metattach {volume-name} {component-names ...}
```

 - `volume-name` は、拡張するボリュームの名前です。

- *components* には、拡張に使用する個々のコンポーネントの名前を指定します。

詳細は、85 ページの「例 — 1 つのスライスを追加してストライプ連結を作成する」85 ページの「例 — いくつかのスライスを追加してスライス連結を作成する」および `metattach(1M)` のマニュアルページを参照してください。

例 — 1 つのスライスを追加してストライプ連結を作成する

```
# metattach d2 c1t2d0s2
d2: components are attached
```

この例では、既存のスライス `d2` にスライスを追加します。スライスが追加されたことを示すメッセージが表示されます。

例 — いくつかのスライスを追加してスライス連結を作成する

```
# metattach d25 c1t2d0s2 c1t2d1s2 c1t2d3s2
d25: components are attached
```

この例では、既存の 3 面ストライプ `d25` に別の 3 面ストライプを連結します。これら 3 つのスライスには飛び越し値が指定されていないので、`d25` に設定された値が使用されます。最後に、ボリュームが設定されたことを示すメッセージが出力されます。

次の作業

UFS の場合は、このボリュームに対して `growfs` コマンドを実行します。詳細は、243 ページの「ファイルシステムを拡張するには」を参照してください。

データベースなど、`raw` ボリュームを使用するアプリケーションは、独自の方法でこのボリュームを認識し、領域を拡張できなければなりません。

新たに作成したストライプ連結にファイルシステムを作成する方法については、『*Solaris* のシステム管理 (基本編)』の「ファイルシステムの作成 (手順)」を参照してください。

ボリュームの削除

▼ ボリュームを削除するには

1. すべてのデータの最新のバックアップを取っているかどうか、またルート権限を持っているかどうかを確認します。
2. このボリュームを本当に削除しても問題がないか確認します。
ストライプまたは連結を削除し、そのボリュームの一部として使用されているスライスを再使用すると、ボリュームのすべてのデータが失われます。
3. 必要であれば、ファイルシステムをマウント解除します。

```
# umount /filesystem
```

4. 次のどちらかの方法でボリュームを削除します。
 - Solaris 管理コンソール内の「拡張ディスク」から「ボリューム (Volumes)」ノードを開きます。「編集」、「削除」の順に選択してから、ウィザードの指示に従ってボリュームを削除します。詳細は、オンラインヘルプを参照してください。
 - 次の形式の `metaclear` コマンドを使ってボリュームを削除します。

```
metaclear {volume-name }
```

詳細は、次の例と `metaclear(1M)` のマニュアルページを参照してください。

例 — 連結を削除する

```
# umount d8
# metaclear d8
d8: Concat/Stripe is cleared
    (/etc/vfstab ファイルを編集します)
```

この例では、マウントされたファイルシステムが格納されている連結 `d8` を削除します。ボリュームを削除する場合は、あらかじめファイルシステムをマウント解除する必要があります。連結の削除が完了すると、そのことを示すメッセージが表示されます。 `/etc/vfstab` ファイルにこのボリュームのエントリがある場合は、このエントリを削除する必要があります。これによって、存在しないボリュームにファイルシステムをマウントするのを回避できます。

第 9 章

RAID 1 (ミラー) ボリューム (概要)

この章では、Solaris ボリュームマネージャのミラーとサブミラーに関する基本的な概念について説明します。関連する作業については、第 10 章を参照してください。

この章では、以下の内容について説明します。

- 87 ページの「RAID 1 (ミラー) ボリュームの概要」
- 93 ページの「RAID 1 ボリューム (ミラー) の再同期」
- 95 ページの「RAID 1 ボリュームの背景情報」
- 96 ページの「シングルユーザーモードでの起動が RAID 1 ボリュームに与える影響」

RAID 1 (ミラー) ボリュームの概要

RAID 1 ボリューム (またはミラー) とは、同じデータのコピーを複数の RAID 0 (ストライプ方式または連結方式) ボリュームで保持しているボリュームのことです。ミラー化するためには、より多くのディスク容量が必要です。少なくとも、ミラー化するデータ量の 2 倍のディスク容量が必要になります。また、ミラー化ではデータがすべてのサブミラーに書き込まれるため、書き込み要求の処理時間が長くなります。

構成したミラーは、物理スライスと同じように使用できます。

既存のファイルシステムを含め、どのようなファイルシステムでもミラー化できます。また、ミラーは、データベースなど、どのようなアプリケーションにも使用できます。

ヒント - データの安全性と可用性を確保するためには、Solaris ボリュームマネージャのホットスペア機能とミラーを併用します。ホットスペアについては、第 15 章と第 16 章を参照してください。

サブミラーの概要

ミラー化された RAID 0 ボリュームをサブミラーと呼びます。ミラーは、1 つまたは複数の RAID 0 ボリューム (ストライプまたは連結) から構成されます。

ミラーには最大 3 つのサブミラーを使用できますが、実際には 2 面ミラーで十分です。3 つめのサブミラーを構成すると、オンラインでバックアップをとることができます。この場合、バックアップのために 1 つのサブミラーがオフラインになっていても、データの冗長性は失われません。

サブミラーを「オフライン」にすると、そのサブミラーに対する読み取りと、書き込みは停止されます。この時点で、このサブミラーへのアクセスが可能になり、バックアップを実行できます。ただし、オフライン状態のサブミラーは読み取り専用になります。サブミラーがオフライン状態の間、Solaris ボリュームマネージャはミラーに対するすべての書き込みを追跡管理します。サブミラーがオンライン状態に戻ると、サブミラーがオフラインの間に書き込まれた部分 (再同期領域) だけが再同期されます。また、サブミラーをオフラインにすると、エラーが発生した物理デバイスの問題を追跡したり修復したりすることが可能になります。

サブミラーは、いつでもミラーに接続したり、ミラーから切断することができます。ただし、少なくとも 1 つのサブミラーが常時、接続されていなければなりません。

通常は、サブミラーが 1 つだけのミラーを作成し、後で 2 つめのサブミラーを追加します。

シナリオ — RAID 1 (ミラー) ボリューム

図 9-1 に、2 つのボリューム (サブミラー) d20 と d21 から構成されるミラー d2 の例を示します。

Solaris ボリュームマネージャソフトウェアは、データの複製を複数の物理ディスク上に作成し、1 つの仮想ディスクとしてアプリケーションに提供します。ディスクへの書き込みは常に複製されますが、ディスクからの読み取りはミラーを構成するいずれかのサブミラーだけから行われます。ミラー d2 の容量は、もっとも小さいサブミラーのサイズと同じになります (サブミラーのサイズが異なる場合)。

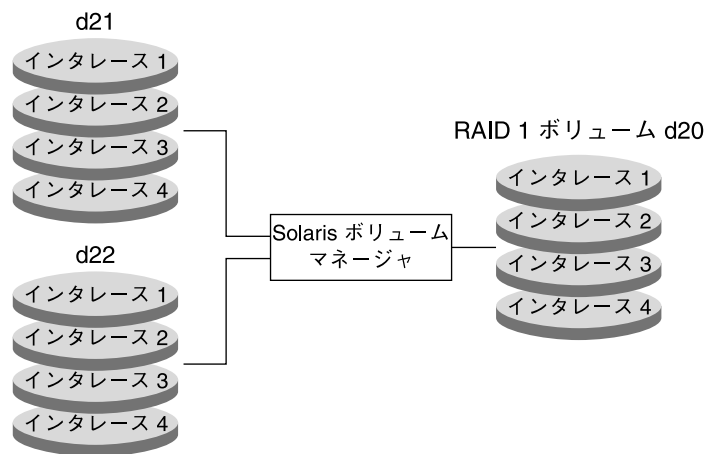


図 9-1 ミラーの例

RAID 1+0 と RAID 0+1 の提供

Solaris ボリュームマネージャは、RAID 1+0 (ミラーをストライプ化した方式) と RAID 0+1 (ストライプをミラー化した方式) の両方の冗長性をサポートします。Solaris ボリュームマネージャインタフェースは、すべての RAID 1 デバイスを RAID 0+1 として扱いますが、可能であれば、ボリュームを構成するコンポーネントやミラーを個別に認識します。

注 – Solaris ボリュームマネージャは、RAID 1+0 機能を常に提供できるわけではありません。しかし、すべてのサブミラーが同じで、ディスクスライス (ソフトウェアパーティションではない) から構成されている、最適化された環境では、RAID 1+0 も提供します。

たとえば、純粋な RAID 0+1 実装で、ストライプ化された 3 つのスライスからなる 2 面ミラーの場合、1 つのスライスに障害が発生すると、ミラーの片面が使用不能になる場合があります。また、ホットスペアが使用されていない場合、2 番目のスライスに障害が発生すると、このミラーはおそらく使用不能になります。Solaris ボリュームマネージャを使用した場合、最大 3 つのスライスに障害が発生しても、ミラーは動作を継続できます。これは、ストライプ化された 3 つのスライスがそれぞれ、ミラーのもう一方の側の対応するスライスに対してミラー化されているためです。

次の例を見てください。

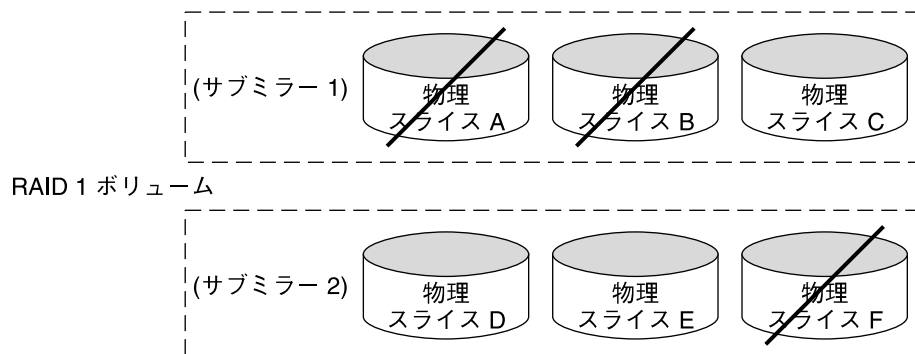


図 9-2 RAID 1+0 の例

ミラー d1 は 2 つのサブミラーから構成され、各サブミラーは、構成と飛び越し値が同じ 3 つの物理ディスクから構成されています。この場合、3 つのディスク A、B、F に障害が発生しても、ミラーの論理ブロック域全体が少なくとも 1 つのディスクによって確保されているため、ミラーは使用不能になりません。

しかし、ディスク A と D に障害が発生すると、ミラーのデータの一部がどのディスク上にも存在しないため、その部分の論理ブロックにはアクセスできなくなります。

複数のスライスの障害によってミラーの一部のデータにアクセスできなくなった場合でも、データがまだ利用可能なミラー部分にはアクセスできます。この場合、ミラーは、不良ブロックを含む単一ディスクのように機能します。損傷部分は使用不能になりますが、残りの部分は使用可能です。

RAID 1 (ミラー) ボリュームに関する FAQ (よくある質問)

ミラーを使用する理由は？

ミラーは、データの可用性を最大限に高めます。ただし、ミラーを使用すると、ミラー化するデータ量の 2 倍のスライス (ディスク) が必要になります。

ミラーを構成するサブミラーの最大数は？

Solaris ボリュームマネージャでは、3 面までのミラー (3 つのサブミラーからなるミラー) を作成できます。しかし、通常、ほとんどのアプリケーションでは、2 面ミラーによって十分なデータ冗長性が得られます。また、2 面ミラーの方が、3 面ミラーよりもディスクドライブのコストが低くなります。

1 面ミラーを最初に作成してから、後でサブミラーを追加する理由は？

ミラー化するデータがすでにある場合は、まず一次サブミラーを作成してから、新しいサブミラーを追加します。これによって、後で、次サブミラーに格納されているデータを使って新しいサブミラーを更新することができます。

- 1つのコマンドで2面または3面ミラーを作成してもかまわない場合は？
ミラー化する既存のデータがなく、すべてのサブミラーのデータが破壊されてしまってもかまわない場合は、1つのコマンドですべてのサブミラーを作成すれば、時間の節約になります。

RAID 1 ボリュームの構成指針

- サブミラーのスライスは、異なるディスクとコントローラに配置します。同じミラーの2つまたはそれ以上のサブミラーのスライスを同じディスクに置くと、データの保護機能が大幅に低下します。同じように、サブミラーは、別個のコントローラに配置します。これは、コントローラやそのケーブルでは、ディスクよりも障害が発生する確率が高いためです。これにより、ミラーの性能も向上します。
- 1つのミラーでは、同じタイプのディスクとコントローラを使用します。特に、古いタイプの SCSI 記憶装置では、ディスクやコントローラの性能がモデルやブランドによって大幅に異なることがあります。性能レベルが異なるデバイスが同じミラーに混在していると、性能が大幅に低下することがあります。
- 同じサイズのサブミラーを使用します。サイズが異なるサブミラーを使用すると、ディスク領域がむだになります。
- 直接マウントできるのはミラーデバイスだけです。オフライン状態のサブミラーを読み取り専用でマウントする場合を除き、サブミラーを直接マウントしてはなりません。また、サブミラーの一部であるスライスをマウントしてはなりません。データが壊され、システムが異常を起こすおそれがあります。
- ミラー化によって読み取り性能が向上することはありますが、書き込み性能は常に低下します。ミラー化によって読み取り性能が向上するのは、スレッド化された入出力や非同期的入出力の場合だけです。シングルスレッドによるボリュームの読み取りでは、性能の向上は得られません。
- ミラーの読み取りオプションの設定を変えてみると、性能が向上することがあります。たとえば、デフォルトの読み取りモードでは、各ディスクが巡回的に1つずつ読み取られます。このモードがデフォルトになっている理由は、UFS マルチユーザー、マルチプロセスアクティビティーでは、通常、これがもっとも効率的であるためです。

場合によっては、geometric 読み取りオプションを使用すると、ヘッドの移動とアクセス時間が最小になり、性能が向上することがあります。このオプションがもっとも効果的に機能するのは、各ディスクにスライスが1つしかない場合、同時に1つのプロセスだけがスライス/ファイルシステムを使用する場合、入出力パターンに高い順次性があるか、すべてのアクセスが読み取りの場合だけです。

ミラーオプションの変更方法については、117 ページの「RAID 1 ボリュームオプションを変更するには」を参照してください。

- `swap -l` コマンドを使ってすべての `swap` デバイスを確認します。swap として指定されたスライスは、他のスライスとは別個にミラー化しなければなりません。
- ミラーには、構成が同じサブミラーだけを使用します。たとえば、ディスクラベルのないサブミラーでミラーを作成すると、ラベルがあるサブミラーをミラーに追加できなくなります。

注 - ミラー化されたファイルシステムで、最初に接続したサブミラーがシリンダ 0 から始まらない場合、追加接続するすべてのサブミラーも、シリンダ 0 から始まらないようにする必要があります。最初のサブミラーがシリンダ 0 から始まらないミラーに、シリンダ 0 から始まるサブミラーを接続しようとする、次のエラーメッセージが表示されます。

```
can't attach labeled submirror to an unlabeled mirror
```

1つのミラー内で使用するサブミラーは、全部シリンダ 0 から始まるか、どれもシリンダ 0 から始まらないかのどちらかにする必要があります。

開始シリンダは、すべてのサブミラーで同じにする必要はありませんが、すべてのサブミラーにシリンダ 0 が含まれるか、すべてのサブミラーにシリンダ 0 が含まれないかのどちらかでなければなりません。

RAID 1 ボリュームのオプション

ミラーの性能を最適化するには、次のオプションを使用します。

- ミラーからの読み取りポリシー
- ミラーへの書き込みポリシー
- ミラーを再同期する順序 (パス番号)

ミラーオプションは、ミラーを最初に作成するときでも、ミラーを設定した後でも設定できます。これらのオプションの変更に関連する作業については、117 ページの「RAID 1 ボリュームオプションを変更するには」を参照してください。

RAID 1 ボリュームの読み取りおよび書き込みポリシー

Solaris ボリュームマネージャでは、ミラーに対してさまざまな読み取りおよび書き込みポリシーを設定できます。構成に合わせて、読み取りおよび書き込みポリシーを適切に設定すると、性能が向上することがあります。

表 9-1 ミラーの読み取りポリシー

読み取りポリシー	説明
ラウンドロビン (デフォルト)	すべてのサブミラーの負荷を均一にします。ミラーに属するすべてのサブミラーの読み取りは、ラウンドロビン方式で (1 つずつ順次に) 行われます。

表 9-1 ミラーの読み取りポリシー (続き)

読み取りポリシー	説明
ジオメトリック	読み取りを論理的なディスクブロックアドレスに基づいて個々のサブミラーに分割します。たとえば、2面サブミラーの場合は、ミラーのディスク領域が、論理アドレスに基づいて同じサイズの2つの領域に分割されます。一方のサブミラーの読み取りは論理的な領域の半分に限定され、他方のサブミラーの読み取りは同じ領域の残り半分に限定されます。ジオメトリック読み取りポリシーでは、読み取りに必要なシーク時間が減少します。このモードによって得られる性能の向上は、システムの入出力負荷やアプリケーションのアクセスパターンによって異なります。
先頭のデバイスから読み取る	すべての読み取りを最初のサブミラーに送る。このポリシーは、先頭のサブミラーを構成するデバイスが2番目のサブミラーのデバイスよりも高速な場合にのみ使用します。

表 9-2 ミラーの書き込みポリシー

書き込みポリシー	説明
並列 (デフォルト)	ミラーへの書き込みは複製され、すべてのサブミラーに対して同時に実行される。
順次	サブミラーへの書き込みは順次実行される (つまり、最初のサブミラーへの書き込みが終わってから、次のサブミラーへの書き込みが始まる)。順次オプションでは、1つのサブミラーへの書き込みが終わらないと、次のサブミラーへの書き込みは開始されない。この順次オプションは、電源の障害などによってサブミラーが読み取り不能になるのを防止するために使用する。

RAID 1 ボリューム (ミラー) の再同期

RAID 1 ボリューム (ミラー) の再同期とは、サブミラーに障害が発生した場合、システムがクラッシュした場合、サブミラーをオフラインにしてから、オンラインに戻した場合、あるいは、新しいサブミラーを追加した場合に、サブミラーのデータを他のサブミラーにコピーする処理のことです。

再同期中も、ミラーの読み書きは実行できます。

ミラーの再同期は、すべてのサブミラーに同じデータを書き込むことによって (書き込みが進行中のデータは除く)、ミラーの有効性を保証します。

注 - ミラーの再同期は必須の処理であり、省略することはできません。ただし、ミラーの再同期を手動で行う必要はありません。この処理は自動的に実行されます。

ミラー全体の再同期

ミラーに新しいサブミラーを接続 (追加) すると、別のサブミラーのすべてのデータが新しいサブミラーに自動的に書き込まれます。ミラーの再同期が完了すると、新しいサブミラーは読み取り可能になります。サブミラーは、明示的に切り離されるまでミラーに接続されたままになります。

再同期の実行中にシステムがクラッシュした場合は、システムが再起動してから、再同期が再開されます。

再同期の最適化

Solaris ボリュームマネージャは、システム障害後の再起動時や、オフラインのサブミラーがオンラインに戻ったときに、最適化されたミラーの再同期を実行します。メタディスクドライバはすべてのサブミラー領域を管理しているため、どのサブミラー領域が障害によって同期の取れない状態になっているかを判定できます。最適化された再同期は、同期が取れていない領域に対してのみ行われます。ユーザーは、再起動時にミラーを再同期する順序を指定できます。また、サブミラーのパス番号を 0 (ゼロ) に設定することによって、ミラーの再同期を省略することができます。(詳細は、94 ページの「パス番号」を参照してください。)



注意 - パス番号 0 は、読み取り専用としてマウントされているミラーに対してのみ設定します。

部分的な再同期

サブミラーを構成するスライスを新しいものに交換すると、Solaris ボリュームマネージャはデータの部分的な再同期を実行します。Solaris ボリュームマネージャは、別のサブミラーの有効なスライスから新しいスライスにデータをコピーします。

パス番号

パス番号 (0 から 9 の数字) は、システムの再起動時にミラーを再同期する順序を決定します。デフォルトのパス番号は 1 です。再同期は、パス番号の小さいミラーから行われます。0 を指定すると、ミラーの再同期はスキップされます。パス番号 0 は、読み取り専用としてマウントされているミラーに対してのみ設定します。同じパス番号をもつミラーの再同期は同時に実行されます。

RAID 1 ボリュームの背景情報

- ミラー化の解除 – Solaris 管理コンソール内の「拡張ディスク」では、ルート (/) や、/opt、/usr、swap、あるいは、システムが動作している間はマウント解除できないファイルシステムのミラー化を解除することはできません。これらのファイルシステムに対しては、コマンド行ユーティリティを使用してください。
- 接続 – サービスを中断せずにサブミラーをミラーに接続できます。サブミラーを接続することによって、2面と3面のミラーを作成できます。
- 切断とオフライン – サブミラーをオフラインにすると、そのサブミラーに対する読み取りや書き込みを禁止できますが、サブミラーとミラーの論理的な関連付けは維持されます。サブミラーがオフラインになっている間、Solaris ボリュームマネージャはミラーに対するすべての書き込みを追跡管理し、サブミラーがオンラインに戻されたときにサブミラーに書き込みを行います。Solaris ボリュームマネージャは、最適化された再同期を行うことによって、サブミラー全体ではなく、変更されたデータの再同期だけを行います。一方、サブミラーを切断すると、サブミラーとミラーの論理的な関連付けも断ち切られます。一般には、保守を行うときはサブミラーをオフラインにし、取り外すときはサブミラーを切断します。

RAID 1 ボリュームを作成するための背景情報

- ミラーを作成する前に、そのミラーに使用する RAID 0 (ストライプ方式または連結方式) ボリュームを作成する必要があります。
- ミラーは、ルート (/)、swap、/usr を始めとするファイルシステムや、データベースなどのアプリケーションに使用できます。



注意 – 既存のファイルシステムのミラーを作成する場合は、そのファイルシステムが最初のサブミラーに含まれていなければなりません。

- ミラーを作成するときは、最初に1面ミラーを作成し、その後で2番目のサブミラーを接続します。これにより、再同期が開始され、データが破壊されることはありません。
- 1面のミラーを作成しておけば、後でそれを2面または3面のミラーにすることができます。
- 3面のミラーまで作成できますが、通常、ほとんどのアプリケーションでは、2面ミラーによって十分なデータ冗長性が得られます。また、2面の方が3面ミラーよりもディスクドライブのコストが低くなります。3面ミラーでは、1つのサブミラーをオフラインにしてバックアップを実行するときも、2面ミラーでデータの冗長性を確保することができます。
- サブミラーには同じサイズのコンポーネントを使用します。コンポーネントのサイズが異なっていると、ミラーにむだな領域が生じます。

- ミラーを作成する前に状態データベースの複製を追加しておく、ミラーの性能が向上することがあります。一般的な指針として、ミラーを追加するたびに2つの状態データベースの複製をシステムに追加する必要があります。Solaris ボリュームマネージャは、追加されたこれらの複製に、最適化された再同期を実行する際に使用するダーティリージョンログ (DRL) を格納します。競合を避けるために十分な数の複製を作成することにより、あるいは、ミラーと同じディスクまたはコントローラ上にある複製を使用することによって、全体的な性能を向上できます。

RAID 1 ボリュームオプションを変更するための背景情報

- ミラーのパス番号と読み書きポリシーは、変更できます。
- ミラーオプションは、ミラーを動作させたまま、変更できます。

シングルユーザーモードでの起動が RAID 1 ボリュームに与える影響

ルート (/)、/usr、および swap (いわゆる「起動」ファイルシステム) 用のミラーを持つシステムを、`boot -s` コマンドを使ってシングルユーザーモードで起動した場合、`metastat` コマンドを実行すると、これらのミラーと、場合によってはシステム上のすべてのミラーが「Needing Maintenance」状態になっていることが示されます。また、これらのスライスに書き込みがあった場合には、ミラーのダーティリージョンが増加していることが示されます。

これは危険な状況に見えますが、心配する必要はありません。`metasync -r` コマンドは通常、起動時にミラーの再同期のために実行されますが、システムがシングルユーザーモードで起動された場合には実行を中断されます。システムを再起動すると、`metasync -r` コマンドが実行され、すべてのミラーの再同期が取られます。

これが心配な場合は、手動で `metasync -r` コマンドを実行してください。

シナリオ — RAID 1 ボリューム (ミラー)

RAID 1 ボリュームは、冗長ボリュームを構築するための手段です。これによって、RAID 1 ボリュームを構成する RAID 0 ボリュームのどれかに部分的または完全な障害が発生した場合でも、データが失われることはなく、ファイルシステムへのアクセスが中断されることもありません。第 4 章のサンプルシステムに基づく構成例は、RAID 1 ボリュームによって冗長性がいかに達成されるかを示しています。

71 ページの「ストライプ方式の飛び越し値」で説明するように、サンプルシステムには 2 つの RAID 0 ボリュームがあります。それぞれのボリュームはおよそ 27G バイトの容量を持ち、3 つのディスクにまたがっています。RAID 1 ボリュームを作成してこれら 2 つの RAID 0 ボリュームをミラー化すると、完全に冗長化された記憶領域によって、障害からの回復が可能なデータ記憶域を構築できます。

この RAID 1 ボリュームでは、どちらのディスクコントローラに障害が発生しても、ボリュームへのアクセスは中断されません。さらに、最大 3 つのディスクに障害が発生しても、アクセスが中断されない場合もあります。

アクセスの中断を引き起こす可能性がある問題に対してさらに保護が必要な場合は、ホットスペアを使用します。これについては、第 15 章 (特に 158 ページの「ホットスペアの仕組み」) を参照してください。

第 10 章

RAID 1 (ミラー) ボリューム (作業)

この章では、Solaris ボリュームマネージャの RAID 1 ボリュームに関連する作業について説明します。RAID 1 ボリュームの概念については、第 9 章を参照してください。

RAID 1 ボリューム (作業マップ)

次の表に、Solaris ボリュームマネージャの RAID 1 ボリュームを管理するのに必要な作業を示します。

作業	説明	参照先
未使用のスライスからミラーを作成する	Solaris ボリュームマネージャの GUI か <code>metainit</code> コマンドを使って、未使用のスライスからミラーを作成します。	101 ページの「未使用のスライスから RAID 1 ボリュームを作成するには」
既存のファイルシステムからミラーを作成する	Solaris ボリュームマネージャの GUI か <code>metainit</code> コマンドを使って、既存のファイルシステムからミラーを作成します。	103 ページの「ファイルシステムから RAID 1 ボリュームを作成するには」
ミラー化されたルートの代替起動デバイスへのパスを指定する	代替起動デバイスへのパスを起動手順の中で指定します。	108 ページの「代替起動デバイスへのパスを記録するには」
サブミラーを接続する	Solaris ボリュームマネージャの GUI か <code>metattach</code> コマンドを使ってサブミラーを接続します。	110 ページの「サブミラーを接続するには」

作業	説明	参照先
サブミラーを切り離す	Solaris ボリュームマネージャの GUI か <code>metattach</code> コマンドを使ってサブミラーを切り離します。	111 ページの「サブミラーを切り離すには」
サブミラーをオンラインまたはオフラインにする	Solaris ボリュームマネージャの GUI か <code>metaonline</code> コマンドを使ってサブミラーをオンラインにします。 Solaris ボリュームマネージャの GUI か <code>metaoffline</code> コマンドを使ってサブミラーをオフラインにします。	112 ページの「サブミラーをオフラインまたはオンラインにするには」
サブミラー内のコンポーネントを有効にする	Solaris ボリュームマネージャの GUI か <code>metareplace</code> コマンドを使って、サブミラー内のスライスを実効にします。	113 ページの「サブミラー内のスライスを有効にするには」
ミラーの状態をチェックする	Solaris ボリュームマネージャの GUI か <code>metastat</code> コマンドを使って、RAID 1 ボリュームの状態をチェックします。	116 ページの「ミラーとサブミラーの状態をチェックするには」
ミラーオプションを変更する	Solaris ボリュームマネージャの GUI か <code>metaparam</code> コマンドを使って、特定の RAID 1 ボリュームのオプションを変更します。	117 ページの「RAID 1 ボリュームオプションを変更するには」
ミラーを拡張する	Solaris ボリュームマネージャの GUI か <code>metattach</code> コマンドを使ってミラーの容量を拡張します。	119 ページの「RAID 1 ボリュームを拡張するには」
サブミラー内のスライスを置き換える	Solaris ボリュームマネージャの GUI か <code>metareplace</code> コマンドを使って、サブミラー内のスライスを置き換えます。	120 ページの「サブミラー内のスライスを交換するには」
サブミラーを置き換える	Solaris ボリュームマネージャの GUI か <code>metattach</code> コマンドを使ってサブミラーを置き換えます。	121 ページの「サブミラーを交換するには」
ミラーを削除する (ミラー化を解除する)	Solaris ボリュームマネージャの GUI、あるいは、 <code>metadetach</code> または <code>metaclear</code> コマンドを使って、ファイルシステムのミラー化を解除します。	122 ページの「ファイルシステムのミラー化を解除するには」
マウント解除できないファイルシステムのミラーを削除する (ミラー化を解除する)	Solaris ボリュームマネージャの GUI、あるいは、 <code>metadetach</code> または <code>metaclear</code> コマンドを使って、マウント解除できないファイルシステムのミラーを削除 (ミラー化を解除) します。	124 ページの「マウント解除できないファイルシステムのミラー化を解除するには」

作業	説明	参照先
ミラーを使ってバックアップを実行する	Solaris ボリュームマネージャの GUI、あるいは、 <code>metaonline</code> と <code>metaoffline</code> コマンドを使って、ミラーのバックアップを行います。	126 ページの「RAID 1 ボリュームを使ってオンラインバックアップをとるには」

RAID 1 ボリュームの作成

▼ 未使用のスライスから RAID 1 ボリュームを作成するには

- 49 ページの「Solaris ボリュームマネージャ要素を作成するための前提条件」と 95 ページの「RAID 1 ボリュームを作成するための背景情報」を確認します。
- サブミラーとして使用する 2 つのストライプまたは連結を作成します。
詳細は、80 ページの「RAID 0 (ストライプ方式) ボリュームを作成するには」か 81 ページの「RAID 0 (連結方式) ボリュームを作成するには」を参照してください。
- 次のどちらかの方法でミラーを作成します。
 - Solaris 管理コンソール内の「拡張ディスク」から「ボリューム (Volumes)」ノードを開き、「アクション (Action)」、「ボリュームを作成 (Create Volume)」の順に選択し、画面の指示に従ってミラーを作成します。詳細は、オンラインヘルプを参照してください。
 - 次の形式の `metainit` コマンドを使って 1 面ミラーを作成します。

```
metainit {volume-name} [-m] { submirror-name...}
```

 - `volume-name` は作成するボリュームの名前です。
 - `-m` はミラーを作成することを意味します。
 - `submirror-name` には、ミラーの最初のサブミラーとして使用するコンポーネントの名前を指定します。

詳細は、次の例と `metainit(1M)` のマニュアルページを参照してください。
- 次のどちらかの方法で 2 番目のサブミラーを接続します。
 - Solaris 管理コンソール内の「拡張ディスク」から「ボリューム (Volumes)」ノードを開き、変更するミラーを選択します。「アクション (Action)」、「プロパティ (Properties)」の順に選択し、「サブミラー (Submirrors)」タブを開き、画面の指示に従って「サブミラーを割り当て (Attach Submirror)」を開きます。詳細

は、オンラインヘルプを参照してください。

- 次の形式の `metattach` コマンドを実行します。

```
metattach {mirror-name } {new-submirror-name ...}
```

- *volume-name* は、変更する RAID 1 ボリュームの名前です。
- *submirror-name* には、ミラーの次のサブミラーとして使用するコンポーネントの名前を指定します。

詳細は、次の例と `metattach(1M)` のマニュアルページを参照してください。

例 — 2 面ミラーを作成する

```
# metainit d51 1 1 c0t0d0s2
d51: Concat/Stripe is setup
# metainit d52 1 1 c1t0d0s2
d52: Concat/Stripe is setup
# metainit d50 -m d51
d50: Mirror is setup
# metattach d50 d52
d50: Submirror d52 is attached
```

この例では、2 面ミラー `d50` を作成します。 `metainit` コマンドは、RAID 0 ボリュームである、2 つのサブミラー (`d51` と `d52`) を作成します。 `metainit -m` コマンドは、RAID 0 ボリューム `d51` から 1 面ミラーを作成します。 `metattach` コマンドは、`d52` を接続して 2 面ミラーを作成し、同期を取り直します。(接続されたサブミラー上のデータは、再同期の際に他のサブミラーによって上書きされます。) 最後にミラーが作成されたことを示すメッセージが表示されます。

例 — 2 面ミラーを作成する (再同期なし)

```
# metainit d51 1 1 c0t0d0s2
d51: Concat/Stripe is setup
# metainit d52 1 1 c1t0d0s2
d52: Concat/Stripe is setup
# metainit d50 -m d51 d52
metainit: d50: WARNING: This form of metainit is not recommended.
The submirrors may not have the same data.
Please see ERRORS in metainit(1M) for additional information.
d50: Mirror is setup
```

この例でも、2 面ミラー `d50` を作成します。 `metainit` コマンドは、RAID 0 ボリュームである、2 つのサブミラー (`d51` と `d52`) を作成します。 `metainit -m` コマンドは、2 つのサブミラーから RAID 0 ボリューム `d51` を作成します。ただし、再同期は実行されません。このミラーの情報はすべて無効であるとみなされるため、ミラーを使用する前に、`newfs` などを実行することによって情報が再生成されます。

次の作業

新たに作成したミラーにファイルシステムを作成する場合は、『Solaris のシステム管理 (基本編)』の「ファイルシステムの作成 (手順)」を参照してください。データベースなど、raw ボリュームを使用するアプリケーションは、独自の方法でこのボリュームを認識できなければなりません。

▼ ファイルシステムから RAID 1 ボリュームを作成するには

この手順では、既存のファイルシステムをミラー化します。このファイルシステムがマウント解除できる場合は、システムを再起動しなくても、ここに示すすべての手順を完了することができます。ルート (/) など、マウント解除できないファイルシステムの場合は、手順の中でシステムの再起動が必要になります。

注 - ルート (/) をミラー化するときは、一次サブミラーに障害が発生し、システムを再起動しなければならない場合に備えて、二次ルートスライスの名前を記録しておく必要があります。この情報は、システム上に記録するのではなく、書き留めておきます。システムは常に使用可能であるとは限りません。代替起動デバイスの記録と代替起動デバイスからの起動については、第 24 章を参照してください。

IA システムでルートをミラー化する場合は、RAID 0 や RAID 1 デバイスを作成する前に、起動情報を代替起動ディスクに記録する必要があります。詳細は、『Solaris のシステム管理 (基本編)』の「SPARC: システムのブート (手順)」を参照してください。

この手順では、既存のデバイスを `c1t0d0s0` とします。2 番目のデバイス `c1t1d0s0` はミラーの 2 番目として使用します。サブミラーは `d1` と `d2`、ミラーは `d0` です。

1. 49 ページの「Solaris ボリュームマネージャ要素を作成するための前提条件」と 95 ページの「RAID 1 ボリュームを作成するための背景情報」を確認します。
2. ミラー化するファイルシステムが含まれているスライスを特定します (この例では `c1t0d0s0`)。
3. 次のどちらかの方法を使って、前の手順で特定したスライスに新しい **RAID 0** ボリュームを作成します。
 - Solaris 管理コンソール内の「拡張ディスク」から「ボリューム (Volumes)」ノードを開き、「アクション (Action)」、「ボリュームを作成 (Create Volume)」の順に選択し、画面の指示に従って RAID 0 ボリュームを作成します。詳細は、オンラインヘルプを参照してください。
 - `metainit raid-0-volume-name -f 1 1 ctds-of-slice` コマンドを実行します。

```
# metainit d1 -f 1 1 c1t0d0s0
```

4. 未使用のスライス (この例では `c1t1d0s0`) に 2 番目の **RAID 0** ボリューム (連結) を作成します。これは、後で 2 番目のサブミラーとして使用します。2 番目のサブミラーのサイズは、最初のサブミラー以上でなければなりません。この手順では、次のどちらかの方法を使用します。

- Solaris 管理コンソール内の「拡張ディスク」から「ボリューム (Volumes)」ノードを開き、「アクション (Action)」、「ボリュームを作成 (Create Volume)」の順に選択し、画面の指示に従ってミラーを作成します。詳細は、オンラインヘルプを参照してください。

- `metainit second-raid-0-volume-name 1 1 ctds-of-slice` コマンドを実行します。

```
# metainit d2 1 1 c1t1d0s0
```

5. 次のどちらかの方法で 1 面ミラーを作成します。

- Solaris 管理コンソール内の「拡張ディスク」から「ボリューム (Volumes)」ノードを開き、「アクション (Action)」、「ボリュームを作成 (Create Volume)」の順に選択し、画面の指示に従ってミラーを作成します。詳細は、オンラインヘルプを参照してください。

- `metainit mirror-name - m raid-0-volume-name` コマンドを実行します。

```
# metainit d0 -m d1
```

詳細は、`metainit (1M)` のマニュアルページを参照してください。

注 – 既存のファイルシステムからミラーを作成する場合は、データが破壊されないように、次の 2 つの手順に忠実に従ってください。

ルート (`/`) 以外のファイルシステムをミラー化する場合は、そのファイルシステムのマウント手順がミラー (ブロックデバイスではなく) を参照するように `/etc/vfstab` ファイルを編集する必要があります。

`/etc/vfstab` ファイルの詳細は、『Solaris のシステム管理 (基本編)』の「ファイルシステムのマウント」を参照してください。

6. 次のいずれかの方法で、新たにミラー化したファイルシステムをマウントし直します。

- ルート (`/`) ファイルシステムをミラー化している場合は、`metaroot d0` コマンドを実行します (`d0` には、先ほど作成したミラーの名前を指定する)。次にシステムを再起動します。

詳細は、`metaroot (1M)` のマニュアルページを参照してください。

- マウント解除可能なファイルシステムをミラー化している場合は、ファイルシステムをマウント解除してから再びマウントします。

- マウント解除できない、ルート (`/`) 以外のファイルシステムをミラー化している場合は、システムを再起動します。

7. `metattach` コマンドを使って 2 番目のサブミラーを接続します。


```
# metattach d0 d2
```

詳細は、metattach(1M) のマニュアルページを参照してください。

8. ルートファイルシステムをミラー化している場合は、代替起動パスを書き留めておきます。
詳細は、108 ページの「代替起動デバイスへのパスを記録するには」を参照してください。



注意 - まず metainit コマンドで 1 面ミラーを作成してから、metattach コマンドで追加のサブミラーを接続します。metattach コマンドを使用しないと、再同期は実行されません。この場合、Solaris ボリュームマネージャはミラーの両側が同一であるとみなし、両方を区別なく使用するため、データが破壊されるおそれがあります。

例 — 2 面ミラーを作成する (マウント解除できるファイルシステム)

```
# metainit -f d1 1 1 c1t0d0s0
d1: Concat/Stripe is setup
# metainit d2 1 1 c1t1d0s0
d2: Concat/Stripe is setup
# metainit d0 -m d1
d0: Mirror is setup
# umount /master
(ファイルシステムがミラーを参照するように /etc/vfstab ファイルを編集する)
# mount /master
# metattach d0 d2
d0: Submirror d2 is attached
```

まず、-f オプションを使って最初の連結 d1 を強制的に作成します。これには、/dev/dsk/c1t0d0s0 にマウントされたファイルシステム /master が含まれています。次に 2 番目の連結 d2 を /dev/dsk/c1t1d0s0 から作成します。(このスライスのサイズは d1 以上でなければなりません。)そして、metainit コマンドに -m オプションを付けて実行し、d1 から 1 面ミラー d0 を作成します。

次に、このファイルシステムのエントリがミラーを参照するように /etc/vfstab ファイルを編集します。たとえば、次の行を見てください。

```
/dev/dsk/c1t0d0s0 /dev/rdisk/c1t0d0s0 /var ufs 2 yes -
```

上記の行を次のように変更します。

```
/dev/md/dsk/d0 /dev/md/rdisk/d0 /var ufs 2 yes -
```

最後に、ファイルシステムをマウントし直し、サブミラー d2 をミラーに接続して、ミラーの同期を取り直します。RAID 0 と RAID 1 ボリュームが設定され、サブミラー d2 が接続されたことを示すメッセージが表示されます。

例 — ルート (/) からミラーを作成する

```
# metainit -f d1 1 1 c0t0d0s0
d11: Concat/Stripe is setup
# metainit d2 1 1 c0t1d0s0
d12: Concat/Stripe is setup
# metainit d0 -m d1
d10: Mirror is setup
# metaroot d0
# lockfs -fa
# reboot
...
# metattach d0 d2
d10: Submirror d12 is attached
# ls -l /dev/rdisk/c0t1d0s0
lrwxrwxrwx  1 root      root          88 Feb  8 15:51 /dev/rdisk/c1t3d0s0 ->
../../../../devices/iommu@f,e0000000/vme@f,df010000/SUNW,pn@4d,1080000/ipi3sc@0,0/i
d@3,0:a,raw
```

注 — システムを再起動する前に、2 番目のサブミラーを接続しないでください。
metaroot コマンドを実行した後、2 番目のサブミラーを接続する前に、システムを
再起動する必要があります。

まず、-f オプションを使って最初の RAID 0 ボリューム d1 を強制的に作成します。
これには、/dev/dsk/c0t0d0s0 にマウントされているファイルシステム、ルート
(/) が含まれています。次に 2 番目の連結 d2 を /dev/dsk/c0t1d0s0 から作成しま
す。(このスライスのサイズは d1 以上でなければなりません)。そして、metainit コ
マンドに -m オプションを付けて実行し、ルート (/) を含む連結から 1 面ミラー d0 を
作成します。

次に metaroot コマンドを使って /etc/vfstab ファイルと /etc/system ファイ
ルを編集し、システムがボリューム上のルート (/) ファイルシステムから起動される
ように指定します (再起動を行う前に lockfs -fa コマンドを実行するようにしま
す)。再起動が終わると、サブミラー d2 をミラーに接続し、ミラーの再同期を実行し
ます (連結とミラーが設定され、サブミラー d2 が接続されたことを示すメッセ
ージが表示されます)。最後に、ルート raw デバイスに対して ls -l コマンドを実行して、
代替ルートデバイスへのパスを表示します。このパスは、このデバイスからシステム
を起動しなければならない状況が発生したときに必要になります。

例 — 2 面ミラーを作成する (マウント解除できないファイルシステム — /usr)

```
# metainit -f d12 1 1 c0t3d0s6
d12: Concat/Stripe is setup
# metainit d22 1 1 c1t0d0s6
d22: Concat/Stripe is setup
# metainit d2 -m d12
```

```

d2: Mirror is setup
    (/usr がミラーを参照するように /etc/vfstab ファイルを編集する)
# reboot
...
# metattach d2 d22
d2: Submirror d22 is attached

```

まず、`-f` オプションを指定して最初の連結 `d12` を強制的に作成します。これには、`/dev/dsk/c0t3d0s6` にマウントされているファイルシステム `/usr` が含まれています。次に 2 番目の連結 `d22` を `/dev/dsk/c1t0d0s6` から作成します。(このスライスのサイズは `d12` 以上でなければなりません。) `metainit` コマンドに `-m` オプションを付けて実行し、`/usr` を含む連結から 1 面ミラー `d2` を作成します。次に、`/usr` のエントリがミラーを参照するように `/etc/vfstab` ファイルを編集します。たとえば、次の行を見てください。

```
/dev/dsk/c0t3d0s6 /dev/rdisk/c0t3d0s6 /usr ufs 1 yes -
```

上記の行を次のように変更します。

```
/dev/md/dsk/d2 /dev/md/rdsk/d2 /usr ufs 1 yes -
```

再起動が終わると、2 番目のサブミラー `d22` がミラーに接続され、ミラーの再同期が実行されます。(連結とミラーが設定され、サブミラー `d22` が接続されたことを示すメッセージが表示されます。)

例 — swap からミラーを作成する

```

# metainit -f d11 1 1 c0t0d0s1
d11: Concat/Stripe is setup
# metainit d21 1 1 c1t0d0s1
d21: Concat/Stripe is setup
# metainit d1 -m d11
d1: Mirror is setup
    (swap がミラーを参照するように /etc/vfstab ファイルを編集する)
# reboot
...
# metattach d1 d21
d1: Submirror d21 is attached

```

まず、`-f` オプションを指定して最初の連結 `d11` を強制的に作成します。これには、`/dev/dsk/c0t0d0s1` にマウントされているファイルシステム `swap` が含まれています。次に 2 番目の連結 `d21` を `/dev/dsk/c1t0d0s1` から作成します。(このスライスのサイズは `d11` 以上でなければなりません)。 `metainit` コマンドに `-m` オプションを付けて実行し、`swap` を含む連結から 1 面ミラー `d1` を作成します。次に、`/etc/vfstab` ファイルに `swap` のエントリがある場合は、このエントリがミラーを参照するようにファイルを編集する必要があります。たとえば、次の行を見てください。

```
/dev/dsk/c0t0d0s1 - - swap - no -
```

上記の行を次のように変更します。

```
/dev/md/dsk/d1 - - swap - no -
```

再起動が終わると、2 番目のサブミラー d21 がミラーに接続され、ミラーの再同期が実行されます。(連結とミラーが設定され、サブミラー d21 が接続されたことを示すメッセージが表示されます。)

swap をミラー化した場合、クラッシュダンプを保存するためには、dumpadm コマンドを使ってダンプデバイスをボリュームとして構成する必要があります。たとえば、swap デバイスの名前が /dev/md/dsk/d2 であれば、dumpadm コマンドを使ってこのデバイスをダンプデバイスとして設定します。

ルート (/) のミラー化に関する特殊な考慮事項

ルート (/) をミラー化する作業は、マウント解除できない他のファイルシステムをミラー化する場合の作業と同じです。ただし、ルートのミラー化では、/etc/vfstab ファイルを手動で編集する代わりに、metaroot コマンドを実行する必要があります。詳細は、103 ページの「ファイルシステムから RAID 1 ボリュームを作成するには」を参照してください。次の各項では、ルート (/) ファイルシステムのミラー化に関する特殊な考慮事項と問題点について説明します。

起動時の警告

ルート (/) ファイルシステムをミラー化すると、エラーメッセージがコンソールに表示され、(/etc/syslog.conf での定義に従って) システムログに記録されます。これらのメッセージは問題を意味するものではありません。現在使用していないデバイスタイプすべてに対して表示されます。使用していないモジュールは強制的にロードできないからです。次のようなエラーメッセージが表示されます。

```
Jul 13 10:17:42 ifr genunix: [ID 370176 kern.warning] WARNING: forceload of
misc/md_trans failed
Jul 13 10:17:42 ifr genunix: [ID 370176 kern.warning] WARNING: forceload of
misc/md_raid failed
Jul 13 10:17:42 ifr genunix: [ID 370176 kern.warning] WARNING: forceload of
misc/md_hotspares failed
```

このような警告メッセージは無視してかまいません。

代替起動デバイスへのパスを記録するには

ルート (/) をミラー化する場合は、一次デバイスに障害が発生したときのために代替起動デバイスへのパスが必要になります。代替起動デバイスを検出および記録する方法は、システムのアーキテクチャによって異なります。詳細は、109 ページの「SPARC: 例 — 代替起動デバイスへのパスを記録する」または 109 ページの「IA: 例 — 代替起動デバイスへのパスを記録する」を参照してください。

SPARC: 例 — 代替起動デバイスへのパスを記録する

この例では、代替ルートデバイスへのパスを調べます。そのためには、ルート (/) ミラーに 2 番目のサブミラーとして接続されているスライスに対して `ls -l` コマンドを実行する必要があります。

```
# ls -l /dev/rdisk/c1t3d0s0
lrwxrwxrwx 1 root root 55 Mar 5 12:54 /dev/rdisk/c1t3d0s0 -> \
.././devices/sbus@1,f8000000/esp@1,200000/sd@3,0:a
```

ここで、`/devices` ディレクトリに続く次の文字列を記録しておきます。
`/sbus@1,f8000000/esp@1,200000/sd@3,0:a`

OpenBoot™ Prom 付きのシステムで使用するユーザーは、OpenBoot の `nvalias` コマンドを使って、二次ルート (/) ミラー用の「バックアップルート」デバイス別名を定義できます。たとえば、次のように指定します。

```
ok nvalias backup_root /sbus@1,f8000000/esp@1,200000/sd@3,0:a
```

次に、`boot-device` 別名が一次サブミラーと二次サブミラーの両方を参照するようにこの別名を再定義し、構成を保存します。サブミラーは、指定された順に使用されます。

```
ok printenv boot-device
boot-device = disk net
ok setenv boot-device disk backup-root net
boot-device = disk backup-root net
ok nvstore
```

一次ルートディスクに障害が発生すると、システムは 2 番目のサブミラーから自動的に起動されます。自動起動ではなく、手動で起動する場合は、次のように入力します。

```
ok boot backup_root
```

IA: 例 — 代替起動デバイスへのパスを記録する

この例では、代替起動デバイスへのパスを調べます。そのためには、ルート (/) ミラーに 2 番目のサブミラーとして接続されているスライスに対して `ls -l` コマンドを実行する必要があります。

```
# ls -l /dev/rdisk/c1t0d0s0
lrwxrwxrwx 1 root root 55 Mar 5 12:54 /dev/rdisk/c1t0d0s0 -> ../.
./devices/eisa/eha@1000,0/cmdk@1,0:a
```

ここで、`/devices` ディレクトリに続く次の文字列を記録しておきます。
`/eisa/eha@1000,0/cmdk@1,0:a`

代替起動デバイスからの起動

ミラー化されたルート (/) の一次サブミラーに障害が発生した場合は、他のサブミラーからシステムを起動する必要があります。ミラーの 2 番目のサブミラーから自動的に起動するように構成することもできますし、2 番目のサブミラーから手動で起動するように構成することも可能です。

詳細は、『Solaris のシステム管理 (基本編)』の「SPARC: システムのブート (手順)」を参照してください。

サブミラーに関する作業

▼ サブミラーを接続するには

1. サブミラーとして使用するコンポーネント (連結またはストライプ) を特定します。
サブミラーのサイズは、ミラー内の既存のサブミラーと同じかそれ以上でなければなりません。サブミラーとして使用するボリュームをまだ作成していない場合は、80 ページの「RAID 0 (ストライプ方式) ボリュームの作成」または 81 ページの「RAID 0 (連結方式) ボリューム」を参照してください。
2. すべてのデータの最新のバックアップを取っているか確認します。また、この操作にはルート権限が必要です。
3. 次のどちらかの方法でサブミラーを接続します。
 - Solaris 管理コンソール内の「拡張ディスク」から「ボリューム (Volumes)」ノードを開き、ミラーを選択し、「アクション (Action)」、「プロパティ (Properties)」の順に選択し、「サブミラー (Submirror)」タブをクリックします。画面の指示に従ってサブミラーを接続します。詳細は、オンラインヘルプを参照してください。
 - `metattach mirror submirror` コマンドを実行します。

```
# metattach mirror submirror
```

詳細は、`metattach(1M)` のマニュアルページを参照してください。

注 - 「can't attach labeled submirror to an unlabeled mirror」というエラーメッセージが表示された場合、ミラーに RAID 0 ボリュームを接続できなかったことを意味します。ラベル付きボリューム (サブミラー) は、最初のコンポーネントがシリンダ 0 から始まります。ラベルなしのボリュームは、最初のコンポーネントがシリンダ 1 から始まります。ラベル付きサブミラーのラベルを壊す恐れがあるので、ラベル付きサブミラーをラベルなしのミラーに接続することを Solaris ボリュームマネージャは認めません。

例 — サブミラーを接続する

```
# metastat d30
d30: mirror
      Submirror 0: d60
      State: Okay
...
# metattach d30 d70
d30: submirror d70 is attached
# metastat d30
d30: mirror
      Submirror 0: d60
      State: Okay
      Submirror 1: d70
      State: Resyncing
      Resync in progress: 41 % done
      Pass: 1
      Read option: roundrobin (default)
      Write option: parallel (default)
      Size: 2006130 blocks
...
```

この例では、サブミラー d70 を 1 面ミラー d30 に接続して 2 面ミラーを作成します。ミラー d30 は、最初、サブミラー d60 から構成されています。サブミラー d70 は RAID 0 ボリュームです。まず、サブミラーを接続する前に、metastat コマンドでミラーが「正常 (Okay)」状態であることを確認します。metattach コマンドを実行すると、新しいサブミラーと既存のミラーの同期がとられます。ミラーに新しいサブミラーが接続されると、そのことを示すメッセージが表示されます。サブミラーとミラーの同期がとられていることを確認するために、metastat コマンドを実行します。

▼ サブミラーを切り離すには

1. すべてのデータの最新のバックアップを取っているか確認します。また、この操作にはルート権限が必要です。
2. 95 ページの「RAID 1 ボリュームの背景情報」を確認します。
3. 次のどちらかの方法でサブミラーを切り離します。

- Solaris 管理コンソール内の「拡張ディスク」から「ボリューム (Volumes)」ノードを開き、ミラーを選択し、「アクション (Action)」、「プロパティ (Properties)」の順に選択し、「サブミラー (Submirror)」タブをクリックします。画面の指示に従ってサブミラーを切り離します。詳細は、オンラインヘルプを参照してください。
- `metadetach` コマンドを使ってミラーからサブミラーを切り離します。


```
# metadetach mirror submirror
```

 詳細は、`metadetach(1M)` のマニュアルページを参照してください。

例 — サブミラーを切断する

```
# metastat
d5: mirror
    Submirror 0: d50
...
# metadetach d5 d50
d5: submirror d50 is detached
```

この例では、`metadetach` コマンドを使って、ミラー `d5` からサブミラー `d50` を切り離します。`d50` のスライスは他の場所で再使用されます。サブミラーが切り離されると、そのことを示すメッセージが表示されます。

▼ サブミラーをオフラインまたはオンラインにするには

`metaonline` コマンドを実行できるのは、そのサブミラーが `metaoffline` コマンドによってオフラインにされている場合に限られます。`metaonline` コマンドを実行すると、サブミラーとミラーの再同期が自動的に開始します。

注 – `metaoffline` コマンドの機能は、`metadetach` コマンドの機能に似ています。ただし、`metaoffline` コマンドでは、サブミラーとミラーの論理的な関連付けは切り離されません。

1. すべてのデータの最新のバックアップを取っているか確認します。また、この操作にはルート権限が必要です。
2. 95 ページの「RAID 1 ボリュームの背景情報」を確認します。
3. 次のどちらかの方法でサブミラーをオンラインまたはオフラインにします。
 - Solaris 管理コンソール内の「拡張ディスク」から「ボリューム (Volumes)」ノードを開き、ミラーを選択し、「アクション (Action)」、「プロパティ (Properties)」の順に選択し、「サブミラー (Submirror)」タブをクリックします。画面の指示に従ってサブミラーをオンラインまたはオフラインにします。詳細は、オンライン

ヘルプを参照してください。

- `metaoffline` コマンドでサブミラーをオフラインにします。

```
# metaoffline mirror submirror
```

詳細は、`metaoffline(1M)` のマニュアルページを参照してください。

- `metaonline` コマンドでサブミラーをオンラインにします。

```
# metaonline mirror submirror
```

詳細は、`metaonline(1M)` のマニュアルページを参照してください。

例 — サブミラーをオフラインにする

```
# metaoffline d10 d11
d10: submirror d11 is offlined
```

この例では、サブミラー `d11` をミラー `d10` からオフラインにします。読み取りは他のサブミラーから行われます。最初の書き込みが行われると、ミラーは同期していない状態になります。この不整合の状態は、オフラインにしたサブミラーをオンラインに戻すと訂正されます。

例 — サブミラーをオンラインにする

```
# metaonline d10 d11
d10: submirror d11 is online
```

この例では、サブミラー `d11` をミラー `d10` でオンラインに戻します。

▼ サブミラー内のスライスを有効にするには

1. すべてのデータの最新のバックアップを取っているか確認します。また、この操作にはルート権限が必要です。
2. 244 ページの「**RAID 1** および **RAID 5** ボリューム内のコンポーネントの交換と有効化の概要」と 95 ページの「**RAID 1** ボリュームの背景情報」を確認します。
3. 次のどちらかの方法でサブミラー内のスライスを有効にします。
 - Solaris 管理コンソール内の「拡張ディスク」から「ボリューム (Volumes)」ノードを開き、ミラーを選択し、「アクション (Action)」、「プロパティ (Properties)」の順に選択し、「サブミラー (Submirror)」タブをクリックします。画面の指示に従ってスライスを有効にします。詳細は、オンラインヘルプを参照してください。
 - `metareplace` コマンドを使って、サブミラー内のエラーが発生したスライスを有効にします。

```
# metareplace -e mirror failed-slice
```

metareplace コマンドを実行すると、修復または交換されたスライスとミラーの他の部分との再同期が自動的に開始されます。

詳細は、metareplace (1M) のマニュアルページを参照してください。

例 — サブミラー内のスライスを有効にする

```
# metareplace -e d11 c1t4d0s7
d11: device c1t4d0s7 is enabled
```

この例の場合、ミラー d11 には、ソフトウェアが発生したスライス c1t4d0s7 を使用するサブミラーがあります。-e オプションを付けた metareplace コマンドを実行して、エラーの発生したスライスを有効にします。

注 - 物理ディスクに障害が発生した場合は、そのディスクをシステム上で利用可能な他のディスク (およびスライス) と交換できます (120 ページの「サブミラー内のスライスを交換するには」を参照)。あるいは、ディスクを修復または交換し、フォーマットした上で、この例のように、-e オプションを指定した metareplace コマンドを使用することもできます。

RAID 1 ボリュームの保守

Solaris ボリュームマネージャでは RAID 1 ボリュームやサブミラーの状態が表示されるため、システム管理者は、どのような保守が必要であるか判断できます。次の表に、ミラーの状態を示します。

表 10-1 サブミラーの状態

状態	意味
正常 (Okay)	サブミラーにはエラーがなく正常に動作しています。
再同期中 (Resyncing)	サブミラーで再同期処理が実行されている。エラーが発生したが、すでに訂正され、オンラインに戻されたか、あるいは、新しいサブミラーが接続されたところです。
保守が必要 (Needs Maintenance)	サブミラー内のスライスに入出力エラーまたはオープンエラーが発生しました。スライスに対するすべての読み取りと書き込みはすでに停止されています。

また、metastat コマンドを実行すると、サブミラー内のスライスごとに、「デバイス (Device)」(ストライプに属するスライスのデバイス名) や、「開始ブロック (Start Block)」(スライスが始まるブロック)、「Dbase」(スライスに状態データベースの複製が含まれているかどうか)、「状態」(スライスの状態)、「ホットスペア (Hot Spare)」(障害が発生したスライスのホットスペアとして使用されるスライス) が表示されます。

ミラーのエラーに対処する場合は、おそらく、スライスの状態がもっとも重要な情報となります。サブミラーの状態は、「正常 (Okay)」や「保守が必要 (Needs Maintenance)」などの一般的な状態情報を提供するだけです。サブミラーの状態が「保守が必要 (Needs Maintenance)」の場合は、スライスの状態を参照する必要があります。スライスの状態が「保守 (Maintenance)」の場合と「最後にエラー (Last Erred)」の場合では、障害から回復するための処置が異なります。スライスの状態が「保守 (Maintenance)」だけの場合は、どのような順序でスライスを修理してもかまいませんが、「保守 (Maintenance)」と「最後にエラー (Last Erred)」のスライスが混在している場合は、「保守 (Maintenance)」のスライスを修復してから「最後にエラー (Last Erred)」のスライスを修復する必要があります。詳細は、244 ページの「RAID 1 および RAID 5 ボリューム内のコンポーネントの交換と有効化の概要」を参照してください。

次の表に、サブミラーのスライスの状態と実行可能な処置を示します。

表 10-2 サブミラーのスライスの状態

状態	意味	処置
正常 (Okay)	コンポーネントにエラーはなく、正常に動作しています。	必要ない
再同期中 (Resyncing)	コンポーネントで再同期処理が実行されています。エラーが発生したが、すでに訂正され、オンラインに戻されたか、あるいは、新しいサブミラーが接続されています。	必要であれば、再同期処理が終了するまでサブミラーの状態を監視します。
保守 (Maintenance)	コンポーネントで入出力エラーかオープンエラーが発生しました。このコンポーネントに対するすべての読み取りと書き込みはすでに停止されています。	障害が発生したコンポーネントを有効にするか交換します。詳細は、113 ページの「サブミラー内のスライスを有効にするには」、または 120 ページの「サブミラー内のスライスを交換するには」を参照してください。metastat コマンドを実行すると、metareplace コマンドを使って行うべき処置を示す invoke 回復メッセージが表示されます。metareplace -e コマンドを使用することもできます。

表 10-2 サブミラーのスライスの状態 (続き)

状態	意味	処置
最後にエラー (Last Erred)	コンポーネントで入出力エラーかオープンエラーが発生しました。しかし、別のスライスに障害があるため、データは他のスライスには複製されません。したがって、入出力は引き続きこのスライスに対して行われます。この入出力がエラーになると、ミラー入出力は失敗します。	まず、「保守 (Maintenance)」状態のコンポーネントを有効にするか、交換します。詳細は、113 ページの「サブミラー内のスライスを有効にするには」、または 120 ページの「サブミラー内のスライスを交換するには」を参照してください。通常は、このエラーがあるとデータが失われるため、ミラーの修復後にミラーを検証する必要があります。ファイルシステムの場合は、fsck コマンドを実行してからデータをチェックします。アプリケーションやデータベースでは、独自の方法でデバイスを検証できなければなりません。

▼ ミラーとサブミラーの状態をチェックするには

- 次のどちらかの方法でミラーやサブミラーの状態をチェックします。
 - Solaris 管理コンソール内の「拡張ディスク」から「ボリューム (Volumes)」ノードを開き、ミラーを選択し、「アクション (Action)」、「プロパティ (Properties)」の順に選択します。画面の指示に従って状態を表示します。詳細は、オンラインヘルプを参照してください。
 - ミラーに対して `metastat` コマンドを実行し、各サブミラーの状態、パス番号、読み取りオプション、書き込みオプション、およびミラーの合計ブロック数を表示します。たとえば、1 面ミラー `d70` の状態をチェックするには、次のようにします。

```
# metastat d70
d70: Mirror
  Submirror 0: d71
    State: Okay
    Pass: 1
    Read option: roundrobin (default)
    Write option: parallel (default)
    Size: 12593637 blocks

d71: Submirror of d70
  State: Okay
  Size: 12593637 blocks
  Stripe 0:
    Device          Start Block  Dbase State    Reloc
  Hot Spare
    c1t3d0s3         0           No   Okay      Yes
  Stripe 1:
    Device          Start Block  Dbase State    Reloc
  Hot Spare
    c1t3d0s4         0           No   Okay      Yes
```

```

        Stripe 2:
          Device           Start Block  Dbase State      Reloc
Hot Spare
  c1t3d0s5                0         No    Okay          Yes

```

ミラーのバス番号や、読み取りオプション、書き込みオプションを変更する手順については、117 ページの「RAID 1 ボリュームオプションを変更するには」を参照してください。

デバイス状態のチェックについては、metastat (1M) のマニュアルページを参照してください。

例 — RAID 1 ボリュームの状態をチェックする

metastat コマンドの出力例を以下に示します。

```

# metastat
d0: Mirror
  Submirror 0: d1
    State: Okay
  Submirror 1: d2
    State: Okay
  Pass: 1
  Read option: roundrobin (default)
  Write option: parallel (default)
  Size: 5600 blocks

d1: Submirror of d0
  State: Okay
  Size: 5600 blocks
  Stripe 0:
    Device           Start Block  Dbase State      Hot Spare
    c0t2d0s7         0           No    Okay

```

...

metastat コマンドは、ミラーのサブミラーごとに、その状態、「invoke」行 (エラーがある場合)、割り当てられたホットスペア集合 (ホットスペアがある場合)、ブロック数、サブミラーの各スライスの情報を表示します。

▼ RAID 1 ボリュームオプションを変更するには

1. すべてのデータの最新のバックアップを取っているか確認します。また、この操作にはルート権限が必要です。
2. 96 ページの「RAID 1 ボリュームオプションを変更するための背景情報」を確認します。
3. 次のどちらかの方法でミラーオプションを変更します。

- Solaris 管理コンソール内の「拡張ディスク」から「ボリューム (Volumes)」ノードを開き、ミラーを選択し、「アクション (Action)」、「プロパティ (Properties)」の順に選択します。画面の指示に従ってオプションを変更します。詳細は、オンラインヘルプを参照してください。
- `metaparam` コマンドを使ってミラーのオプションを表示および変更します。たとえば、読み取りポリシーについて、ミラーを「round robin」から、「first」に変更する場合は、次のコマンドを実行します。

```
# metaparam -r first mirror
```

ミラーオプションの詳細は、92 ページの「RAID 1 ボリュームのオプション」を参照してください。また、`metaparam(1M)` のマニュアルページも参照してください。

例 — RAID 1 ボリュームの読み取りポリシーを変更する

```
# metaparam -r geometric d30
# metaparam d30
d30: mirror current parameters are:
    Pass: 1
    Read option: geometric (-g)
    Write option: parallel (default)
```

この例の `-r` オプションは、ミラーの読み取りポリシーを `geometric` に変更します。

例 — RAID 1 ボリュームの書き込みポリシーを変更する

```
# metaparam -w serial d40
# metaparam d40
d40: mirror current parameters are:
    Pass: 1
    Read option: roundrobin (default)
    Write option: serial (-S)
```

この例の `-w` オプションは、ミラーの書き込みポリシーを `serial` に変更します。

例 — RAID 1 ボリュームのパス番号を変更する

```
# metaparam -p 5 d50
# metaparam d50
d50: mirror current parameters are:
    Pass: 5
    Read option: roundrobin (default)
    Write option: parallel (default)
```

`-p` オプションは、ミラーのパス番号を 5 に変更します。

▼ RAID 1 ボリュームを拡張するには

1. すべてのデータの最新のバックアップを取っているか確認します。また、この操作にはルート権限が必要です。
2. 95 ページの「RAID 1 ボリュームの背景情報」を確認します。
3. 次のどちらかの方法でミラーを拡張します。

- Solaris 管理コンソール内の「拡張ディスク」から「ボリューム (Volumes)」ノードを開き、ミラーを選択し、「アクション (Action)」、「プロパティ (Properties)」の順に選択し、「サブミラー (Submirror)」タブをクリックします。画面の指示に従ってサブミラーを拡張します。詳細は、オンラインヘルプを参照してください。
- `metattach` コマンドを使って各サブミラーにスライスを接続します。たとえば、サブミラーにコンポーネントを接続するには、次のコマンドを実行します。

```
# metattach submirror component
```

ミラー内のすべてのサブミラーを拡張する必要があります。詳細は、`metattach (1M)` のマニュアルページを参照してください。

例 — マウントしているファイルシステムを持つ 2 面ミラーを拡張する

```
# metastat
d8: Mirror
  Submirror 0: d9
    State: Okay
  Submirror 1: d10
    State: Okay
...
# metattach d9 c0t2d0s5
d9: component is attached
# metattach d10 c0t3d0s5
d10: component is attached
```

この例では、ミラー化したマウント済みのファイルシステムを拡張します。そのために、2つのディスクドライブをミラーの2つのサブミラーに連結します。ミラー d8 は、2つのサブミラー d9 と d10 から構成されています。

次の作業

UFS の場合は、ミラーボリュームに対して `growfs (1M)` コマンドを実行します。詳細は、243 ページの「ファイルシステムを拡張するには」を参照してください。

データベースなど、`raw` ボリュームを使用するアプリケーションは、独自の方法で領域を拡張できなければなりません。

RAID 1 ボリュームのコンポーネント障害に対する処置

▼ サブミラー内のスライスを交換するには

1. すべてのデータの最新のバックアップを取っているか確認します。また、この操作にはルート権限が必要です。
2. 244 ページの「RAID 1 および RAID 5 ボリューム内のコンポーネントの交換と有効化の概要」と 95 ページの「RAID 1 ボリュームの背景情報」を確認します。
3. 次のどちらかの方法でミラー内のスライスを交換します。

- Solaris 管理コンソール内の「拡張ディスク」から「ボリューム (Volumes)」ノードを開き、ミラーを選択し、「アクション (Action)」、「プロパティ (Properties)」の順に選択し、「サブミラー (Submirror)」タブをクリックします。画面の指示に従ってスライスを交換します。詳細は、オンラインヘルプを参照してください。

- 次の形式の `metareplace` コマンドを使ってサブミラーのスライスを交換します。

```
metareplace { mirror-name } { component-name... }
```

- `mirror-name` は、作成するボリュームの名前です。
- `component-name` には、置き換えられるコンポーネントの名前を指定します。

詳細は、次の例と `metainit(1M)` のマニュアルページを参照してください。

例 — ミラー内の障害が発生したスライスを交換する

次の例では、障害が発生したスライスを交換します。ただし、システムは、ホットスペア集合を使って障害が発生したディスクを自動的に交換するようには構成されていないものとします。ホットスペア集合については、第 15 章を参照してください。

```
# metastat d6
d6: Mirror
  Submirror 0: d16
    State: Okay
  Submirror 1: d26
    State: Needs maintenance
...
d26: Submirror of d6
  State: Needs maintenance
  Invoke: metareplace d6 c0t2d0s2 <new device>
...
```



```
# metareplace d6 c0t2d0s2 c0t2d2s2
d6: device c0t2d0s2 is replaced with c0t2d2s2
```

metastat コマンドを使用して、ミラー d6 にサブミラー d26 があり、スライスの状態が「保守が必要 (Needs maintenance)」であることを確認します。metareplace コマンドにより、metastat コマンドの出力中の「起動」行に従って、このスライスをシステム内の別のスライスで置き換えます。スライスが置き換えられ、サブミラーの再同期が開始されたことを示すメッセージが表示されます。

▼ サブミラーを交換するには

1. すべてのデータの最新のバックアップを取っているか確認します。また、この操作にはルート権限が必要です。
2. 244 ページの「RAID 1 および RAID 5 ボリューム内のコンポーネントの交換と有効化の概要」と 95 ページの「RAID 1 ボリュームの背景情報」を確認します。
3. 次のどちらかの方法でサブミラーを交換します。
 - Solaris 管理コンソール内の「拡張ディスク」から「ボリューム (Volumes)」ノードを開き、ミラーを選択し、「アクション (Action)」、「プロパティ (Properties)」の順に選択し、「サブミラー (Submirror)」タブをクリックします。画面の指示に従ってサブミラーを交換します。詳細は、オンラインヘルプを参照してください。
 - metadetach、metaclear、metatinit、および metattach コマンドを使って、サブミラー全体を交換します。

例 — ミラー内のサブミラーを交換する

次の例では、アクティブなミラー内のサブミラーを交換します。

注 – 新しいボリューム d22 の構成は、置き換えるコンポーネントによって異なります。この例のように、連結で連結を置き換える場合は問題ありませんが、連結でストライプを置き換えた場合には性能が低下するおそれがあり、最適な置き換えとはいえません。

```
# metastat d20
d20: Mirror
  Submirror 0: d21
    State: Okay
  Submirror 1: d22
    State: Needs maintenance
...
# metadetach -f d20 d22
d20: submirror d22 is detached
# metaclear -f d22
```

```
d22: Concat/Stripe is cleared
# metainit d22 2 1 c1t0d0s2 1 c1t0d1s2
d22: Concat/Stripe is setup
# metattach d20 d22
d20: components are attached
```

metastat コマンドを使用して、2 面ミラー d20 にサブミラー d22 があり、その状態が「保守が必要 (Needs maintenance)」であることを確認します。この例では、サブミラー全体を削除し、作成し直します。metadetach コマンドに、-f オプションを指定してミラーから障害のあるサブミラーを強制的に切り離します。metaclear コマンドは、サブミラーを削除します。metainit コマンドは、新しいスライスからサブミラー d22 を再作成します。metattach コマンドは、再作成されたサブミラーを接続します。これによって、ミラーの再同期が自動的に開始されます。

注 - ミラーが 1 面ミラーとなっている間は、データの冗長性が一時的に失われます。

RAID 1 ボリュームの削除 (ミラー化の解除)

▼ ファイルシステムのミラー化を解除するには

この手順では、システムの動作中にマウント解除できるファイルシステムのミラー化を解除します。ルート (/)、/var、/usr、swap など、システムの動作中にマウント解除できないファイルシステムのミラー化を解除する場合は、124 ページの「マウント解除できないファイルシステムのミラー化を解除するには」を参照してください。

1. すべてのデータの最新のバックアップを取っているか確認します。また、この操作にはルート権限が必要です。
2. 95 ページの「RAID 1 ボリュームの背景情報」を確認します。
3. 少なくとも 1 つのサブミラーが「正常 (Okay)」状態であることを確認します。

```
# metastat
```

4. ファイルシステムをマウント解除します。

```
# umount /home
```

5. サブミラーを切り離します。このサブミラーは、この後もこのファイルシステムのために使用されます。

詳細は、metadetach(1M) のマニュアルページを参照してください。

```
# metadetach d1 d10
```

6. ミラーと残りのサブコンポーネントを削除します。

詳細は、metaclear(1M) のマニュアルページを参照してください。

```
# metaclear -r d1
```

7. 必要であれば、手順 5 で切り離れたコンポーネントを使用するように、`/etc/vfstab` ファイルを編集します。

8. ファイルシステムを再びマウントします。

例 — `/opt` ファイルシステムのミラー化を解除する

```
# metastat d4
d4: Mirror
   Submirror 0: d2
       State: Okay
   Submirror 1: d3
       State: Okay
...
# umount /opt
# metadetach d4 d2
d4: submirror d2 is detached
# metaclear -r d4
d4: Mirror is cleared
d3: Concat/Stripe is cleared
    (/opt のエントリが d4 の代わりにそのスライスまたはボリュームを指すように /etc/vfstab ファイル
    を編集する)
# mount /opt
```

この例の `/opt` ファイルシステムは 2 面ミラー `d4` から構成されています。そのサブミラー `d2` と `d3` はそれぞれ、スライス `/dev/dsk/c0t0d0s0` と `/dev/dsk/c1t0d0s0` から構成されています。metastat コマンドを使用して、少なくとも 1 つのサブミラーが「正常 (Okay)」状態であることを確認します。「正常 (Okay)」状態のサブミラーが存在しないミラーは、最初に修復しなければなりません。次に、このファイルシステムのマウントを解除してから、サブミラー `d2` を切り離します。metaclear -r コマンドは、このミラーと他のサブミラー `d3` を削除します。

次に、`/opt` のエントリが、該当するスライスを参照するように `/etc/vfstab` ファイルを編集します。たとえば、`d4` がミラーで、`d2` がサブミラーであれば、次のように変更します。

```
/dev/md/dsk/d4 /dev/md/rdisk/d4 /opt ufs 2 yes -
```

上記の行を次のように変更します。

```
/dev/md/dsk/d2 /dev/md/rdisk/d2 /opt ufs 2 yes -
```

サブミラー名を使用することによって、ファイルシステムをボリュームにマウントしたままにできます。最後に、/opt ファイルシステムを再びマウントします。

注 - /etc/vfstab ファイルで d4 の代わりに d2 を使用すれば、このミラーのミラー化を解除したことになります。d2 は 1 つのスライスから構成されているため、このデバイスでボリュームをサポートしたくなければ、ファイルシステムをスライス名 (/dev/dsk/c0t0d0s0) にマウントできます。

▼ マウント解除できないファイルシステムのミラー化を解除するには

ルート (/)、/usr、/opt、swap など、システムが通常に動作している間はマウント解除できないファイルシステムのミラー化を解除するには、次の手順を実行します。

1. `metastat` コマンドを実行して、少なくとも 1 つのサブミラーの状態が「正常 (Okay)」であることを確認します。
2. ルート (/)、/usr、/opt、または swap を含むミラーに対して `metadetach` コマンドを実行して、このミラーを 1 面ミラーにします。
3. /usr、/opt、swap の場合は、/etc/vfstab ファイルを編集して、このファイルシステムのエントリが **Solaris** ボリュームマネージャ以外のデバイス (スライス) を参照するようにします。
4. ルート (/) だけの場合は、`metaroot` コマンドを実行します。
5. システムを再起動します。
6. `metaclear` コマンドを実行してミラーとサブミラーを削除します。

例 — ルート (/) のミラー化を解除する

```
# metadetach d0 d20
d0: submirror d20 is detached
# metaroot /dev/dsk/c0t3d0s0
# reboot
...
# metaclear -r d0
d0: Mirror is cleared
d10: Concat/Stripe is cleared
# metaclear d20
d20: Concat/Stripe is cleared
```

この例では、ルート (/) は d0 という名前の 2 面ミラーです。そのサブミラー d10 と d20 はそれぞれ、スライス /dev/dsk/c0t3d0s0 と /dev/dsk/c1t3d0s0 から構成されています。metastat コマンドを使用して、少なくとも 1 つのサブミラーが「正常 (Okay)」状態であることを確認します。(「正常 (Okay)」状態のサブミラーが存在

しないミラーは、最初に修復しなければなりません。)次に、サブミラー d20 を切り離して、ミラー d0 を 1 面ミラーにします。metaroot コマンドには、システムの起動に使用する *rootslice* を指定します。このコマンドは、*/etc/system* と */etc/vfstab* ファイルを編集して、ルート (/) のミラー化を指定する情報を削除します。再起動後の `metaclear -r` コマンドは、ミラーともう 1 つのサブミラー d10 を削除します。最後の `metaclear` コマンドは、サブミラー d20 を削除します。

例 — swap のミラー化を解除する

```
# metastat d1
d1: Mirror
    Submirror 0: d11
        State: Okay
    Submirror 1: d21
        State: Okay
...
# metadetach d1 d21
d1: submirror d21 is detached
    (/etc/vfstab ファイルを編集して、swap のエントリをメタデバイスからスライス名に変更する)
# reboot
...
# metaclear -r d1
d1: Mirror is cleared
d11: Concat/Stripe is cleared
# metaclear d21
d21: Concat/stripes is cleared
```

この例では、`swap` は `d1` という名前の 2 面ミラーです。そのサブミラー `d11` と `d21` はそれぞれ、スライス `/dev/dsk/c0t3d0s1` と `/dev/dsk/c1t3d0s1` から構成されています。metastat コマンドを使用して、少なくとも 1 つのサブミラーが「正常 (Okay)」状態であることを確認します。(「正常 (Okay)」状態のサブミラーが存在しないミラーは、最初に修復しなければなりません。)次に、サブミラー `d21` を切り離して、ミラー `d1` を 1 面ミラーにします。次に、*/etc/vfstab* ファイルを編集して、`swap` のエントリが、サブミラー `d21` のスライスを参照するようにします。たとえば、`d1` がミラーで、`d21` がスライス `/dev/dsk/c0t3d0s1` を含むサブミラーであれば、次のように変更します。

```
/dev/md/dsk/d1 - - swap - no -
```

上記の行を次のように変更します。

```
/dev/dsk/c0t3d0s1 - - swap - no -
```

再起動後の `metaclear -r` コマンドは、ミラーともう 1 つのサブミラー `d11` を削除します。最後の `metaclear` コマンドは、サブミラー `d21` を削除します。

ミラーを使ったデータのバックアップ

Solaris ボリュームマネージャは、バックアップを意図した製品ではありませんが、ミラーをマウント解除したり、ミラー全体をオフラインにすることなく、また、システムを停止したり、データへのユーザーアクセスを中断することなく、ミラー化されたデータをバックアップする手段を提供します。データをバックアップするには、まず、サブミラーの1つをオフラインにして(ミラーの機能は一時的に失われます)、そのバックアップをとります。バックアップが完了したら、そのサブミラーをオンラインに戻し、再同期を実行します。

▼ RAID 1 ボリュームを使ってオンラインバックアップをとるには

この手順は、ルート (/) 以外のすべてのファイルシステムに使用できます。このタイプのバックアップは、活動中のファイルシステムの「スナップショット」を使用することに注意してください。ファイルシステムへの書き込みをロックしたときのファイルシステムの使用状況によっては、バックアップしたファイルやファイルの内容がディスク上の実際のファイルに対応しないことがあります。

この手順には次の制約があります。

- この手順を2面ミラーに対して使用すると、1つのサブミラーをバックアップのためにオフラインにしたときに、データの冗長性が失われます。3面ミラーにはこの問題はありません。
- バックアップの完了後にサブミラーをオンラインに戻したときに、システムにある程度のオーバーヘッドが生じます。

この手順の概要は次のとおりです。

- ファイルシステムへの書き込みをロックします (UFS のみ)。ルート (/) はロックしないようにします。
- `metaoffline` コマンドを使って、このミラーの1つのサブミラーをオフラインにします。
- ファイルシステムのロックを解除します。
- オフラインにしたサブミラーのデータをバックアップします。
- `metaonline` コマンドを使ってサブミラーをオンラインに戻します。

注 - このような手順を定常的に使用する場合は、これをスクリプトにしておくことと実行が容易になります。

1. `metastat` コマンドを実行して、ミラーが「正常 (Okay)」状態であることを確認します。
ミラーが「保守 (Maintenance)」状態の場合は、まずそれを修復する必要があります。
2. ファイルシステムがルート (/) 以外の場合は、次のコマンドを実行して書き込みをロックします。

```
# /usr/sbin/lockfs -w mount point
```

書き込みをロックする必要があるファイルシステムは UFS だけです。このボリュームがデータベース管理ソフトウェアなどのアプリケーション用に `raw` デバイスとして設定されている場合は、`lockfs` を実行する必要はありません。(ただし、ベンダー提供の適切なユーティリティを実行してバッファをフラッシュしたり、アクセスをロックする必要がある場合もあります。)



注意 – ルート (/) の書き込みをロックするとシステムがハングしますので、絶対にロックしないでください。

3. 1 つのサブミラーをオフラインにします。

```
# metaoffline mirror submirror
```

ここで、

`mirror` は、 ミラーのボリューム名です。

`submirror` は、 オフラインにするサブミラー (ボリューム) のボリューム名です。

読み取りは、他のサブミラーから引き続き行われます。最初の書き込みが行われた時点でミラーは同期していない状態になります。この不整合の状態は、手順 6 でサブミラーがオンラインに戻された時点で修復されます。

オフラインにしたファイルシステムに対して `fsck` コマンドを実行する必要はありません。

4. ファイルシステムのロックを解除し、書き込みを再開します。

```
# /usr/sbin/lockfs -u mount-point
```

上の手順 2 で使用したベンダー提供のユーティリティを使って、必要なロック解除手順を実行しなければならない場合があります。

5. オフラインにしたサブミラーのバックアップをとります。

これには、`ufsdump` コマンド、または通常使用しているバックアップユーティリティを使用します。

注 – 適切なバックアップを確実に行うためには、`raw` ボリューム (`/dev/md/rdisk/d4` など) を使用します。「`rdisk`」では、2G バイトを超えるアクセスが可能です。

6. サブミラーをオンラインに戻します。

```
# metaonline mirror submirror
```

サブミラーとミラーの再同期が自動的に開始されます。

例 — ミラーを使ってオンラインバックアップをとる

この例では、ミラー d1 はサブミラー d2 と d3 から構成されています。サブミラー d3 をオフラインにして、このサブミラーのバックアップをとります。この間、サブミラー d2 はオンラインのままです。このミラーにあるファイルシステムは /home1 です。

```
# /usr/sbin/lockfs -w /home1
# metaoffline d1 d3
d1: submirror d3 is offlined
# /usr/sbin/lockfs -u /home1
(/dev/md/rdisk/d3 を使ってバックアップをとる)
# metaonline d1 d3
d1: submirror d3 is onlined
```


第 11 章

ソフトパーティション (概要)

この章では、Solaris ボリュームマネージャのソフトパーティションについて説明します。関連する作業については、第 12 章を参照してください。

この章では、以下の内容について説明します。

- 129 ページの「ソフトパーティションの概要」
- 130 ページの「ソフトパーティションの背景情報」

ソフトパーティションの概要

ディスク容量が大きくなり、ディスクアレイによって Solaris システムが使用できる論理デバイスの容量が増大すると、ディスクや論理ボリュームを 8 つより多くのパーティションに分割し、管理しやすい大きさのファイルシステムやパーティションを使用したくなります。Solaris ボリュームマネージャのソフトパーティションは、この要求に応えるための機能です。

Solaris ボリュームマネージャは、ディスクセット (ローカルのディスクセット、無指定のディスクセットも含む) 当たり最大 8192 の論理ボリュームをサポートしますが、デフォルトのボリューム数は 128 (d0 から d127) に設定されています。論理ボリュームの数を増やしたい場合は、240 ページの「Solaris ボリュームマネージャのデフォルト値の変更」を参照してください。

注 - 論理ボリュームの数を実際に使用する予定の数よりも大幅に増やさないようにしてください。Solaris ボリュームマネージャは、最大数の論理ボリュームそれぞれに対してデバイスノード (/dev/dsk/md/*) とそれに対応するデータ構造を作成します。使用されない余分なボリュームは性能に著しい悪影響を及ぼすことがあります。

ソフトパーティションでは、ディスクスライスや論理ボリュームを任意の数のパーティションに分割できます。区画(ソフトパーティション)には、名前を付ける必要があります。これは、ストライプやミラーなど、他の記憶ボリュームの場合と同じです。名前の付いているソフトパーティションには、これがすでに別のボリュームに含まれていない限り、ファイルシステムなどのアプリケーションからアクセスできます。ボリュームにすでに含まれているソフトパーティションには、直接アクセスしないでください。

ソフトパーティションは、ディスクスライス上に直接置くことも、ミラーや、ストライプ、RAID ボリューム上に置くこともできます。ただし、ボリューム間でソフトパーティションをネストすることはできません。たとえば、ソフトパーティション上にミラー化したストライプを構築し、さらにこの上にソフトパーティションを構築することはできません。

ファイルシステムなどのアプリケーションからはソフトパーティションは連続した1つの論理ボリュームに見えますが、実際には、ソフトパーティションを構成するメディアの任意の場所にある一連のエクステンツから構成されています。システムに重大な障害が発生した場合でも、その障害から回復できるように、ソフトパーティションに加えディスク上のエクステンツヘッダー(システム回復データ域ともいう)にも、ソフトパーティションの情報が記録されます。

ソフトパーティションの背景情報

ソフトパーティションの要件

ソフトパーティションとして使用されているスライスを他の目的で使用することはできません。

ディスクをパーティション分割し、それらのスライスにファイルシステムをすでに構築している場合は、ディスクフォーマットを変更または破棄しなければ、スライスを拡張することはできません。しかし、ソフトパーティションでは、ソフトパーティションを構成するデバイスの容量が許す限りソフトパーティションを拡張できます。他のソフトパーティションにあるデータを移動したり、破棄する必要はありません。

ソフトパーティションの指針

- 技術的にはソフトパーティションのエクステンツをディスク上の任意の場所に手動で置くことは可能ですが、エクステンツは、システムが自動的に配置するようにします。ボリューム構成の表示については、232 ページの「Solaris ボリュームマネージャ構成の表示」に示される `metastat -p` の出力を参照してください。

- ソフトパーティションは、任意のスライス上に構築できますが、ディスクレベルでソフトパーティションを使用するときは、ディスク全体を占有するスライスを作成し、そこにソフトパーティションを作成するのが、もっとも効率的な方法です。
- ソフトウェアパーティションの最大サイズはスライスのサイズやソフトパーティションを構成する論理ボリュームのサイズに制約されるため、ディスクスライス上にボリュームを構築してから、そのボリューム上にソフトパーティションを構築するようにします。この方法では、ボリュームにコンポーネントを追加してから、必要に応じてソフトパーティションを拡張することができます。
- 柔軟性と可用性を最大限に高めるためには、ディスクスライスに RAID 1 (ミラー) か RAID 5 ボリュームを作成してから、そのミラーまたは RAID 5 ボリューム上にソフトパーティションを作成するようにします。

シナリオ — ソフトパーティション

ソフトパーティション用のツールを使えば、大きな記憶領域をそれより小さい管理しやすい領域に分割できます。たとえば、他のシナリオ (96 ページの「シナリオ — RAID 1 ボリューム (ミラー)」や 145 ページの「シナリオ — RAID 5 ボリューム」) では、多数の記憶域を 1 つにまとめることによって、何 G バイトという冗長記憶域が構築できます。しかし、考えられる多くのシナリオでは、少なくとも当初は、これほど大きい領域が必要になることはありません。ソフトパーティションでは、この記憶領域を管理しやすいセクションに分割し、それぞれを完全なファイルシステムとして使用することができます。たとえば、RAID 1 または RAID 5 ボリュームに 1000 のソフトパーティションを作成すれば、ユーザーは個別のファイルシステムにホームディレクトリを持つことができます。ユーザーがより多くの領域を必要とする場合は、単にそのソフトパーティションを拡張するだけですみます。

第 12 章

ソフトパーティション (作業)

この章では、Solaris ボリュームマネージャのソフトパーティションに関連する作業について説明します。これらの作業に伴う概念については、第 11 章を参照してください。

ソフトパーティション (作業マップ)

次の表に、Solaris ボリュームマネージャのソフトパーティションを管理するのに必要な作業を示します。

作業	説明	参照先
ソフトパーティションの作成	Solaris ボリュームマネージャの GUI か <code>metainit</code> コマンドを使ってソフトパーティションを作成します。	134 ページの「ソフトパーティションを作成するには」
ソフトパーティションの状態チェック	Solaris ボリュームマネージャの GUI か <code>metastat</code> コマンドを使ってソフトパーティションの状態をチェックします。	135 ページの「ソフトパーティションの状態をチェックするには」
ソフトパーティションの拡張	Solaris ボリュームマネージャの GUI か <code>metattach</code> コマンドを使ってソフトパーティションを拡張します。	136 ページの「ソフトパーティションを拡張するには」
ソフトパーティションの削除	Solaris ボリュームマネージャの GUI か <code>metaclear</code> コマンドを使ってソフトパーティションを削除します。	137 ページの「ソフトパーティションを削除するには」

ソフトパーティションの作成

▼ ソフトパーティションを作成するには

1. 130 ページの「ソフトパーティションの背景情報」を確認します。
2. 次のどちらかの方法でソフトパーティションを作成します。

- Solaris 管理コンソール内の「拡張ディスク」から「ボリューム (Volumes)」ノードを開きます。「アクション (Action)」、「ボリュームを作成 (Create Volume)」の順に選択し、ウィザードの指示に従ってソフトパーティションを作成します。詳細は、オンラインヘルプを参照してください。
- 次の形式の `metainit` コマンドを使ってソフトパーティションを作成します。

```
metainit [-s set] soft-partition -p [-e] component size
```

`-s` には、使用するディスクセットを指定します。`-s` を指定しないと、ローカル (デフォルト) のディスクセットが使用されます。

`-e` を指定すると、ディスク全体がフォーマットし直され、ディスクのほとんどの部分を占めるスライス 0 と、状態データベースの複製を格納する 4M バイト以上のスライス 7 が設定されます。

`soft-partition` はソフトパーティションの名前です。その形式は `dnnn` で、`nnn` は 0 から 8192 の数字です。

`component` は、ソフトパーティションの作成に使用するディスク、スライス、または (論理) ボリュームです。ソフトパーティションのヘッダーがコンポーネントの先頭部分に書き込まれるため、コンポーネントにあるデータはすべて破壊されます。

`size` には、ソフトパーティションのサイズを数字と次のいずれかの単位で指定します。

- M または m (メガバイト)
- G または g (ギガバイト)
- T または t (テラバイト)
- B または b (ブロック数 (セクター数))

詳細は、次の例と `metainit (1M)` のマニュアルページを参照してください。

例 — ソフトパーティションを作成する

```
# metainit d20 -p c1t3d0s2 4g
```

この例では、`d20` という名前の 4G バイトのソフトパーティションを `c1t3d0s2` に作成します。

例 — ディスク全体をソフトパーティションに使用する

この例では、ディスク `c1t2d0` のパーティションを再分割し、スライス 0 に新しいソフトパーティションを作成します。ディスクのデータはすべて破壊されます。使用するコマンドは次のとおりです。

```
metainit d7 -p -e c1t2d0 1G
```

ソフトパーティションの保守

ソフトパーティションの保守は、他の論理ボリュームの保守と同じです。次の各項でこの手順を説明します。

▼ ソフトパーティションの状態をチェックするには

1. 130 ページの「ソフトパーティションの背景情報」を確認します。
2. 次のどちらかの方法でソフトパーティションの状態をチェックします。
 - Solaris 管理コンソール内の「拡張ディスク」から「ボリューム (Volumes)」ノードを開きます。監視するソフトパーティションを選択し、「アクション (Action)」、「プロパティ (Properties)」の順に選択して、画面の指示に従って状態をチェックします。詳細は、オンラインヘルプを参照してください。
 - 次の形式の `metastat` コマンドを使って既存の構成を表示します。

```
metastat soft-partition
```

`soft-partition` はチェックしたいパーティションの名前です。

詳細は、232 ページの「Solaris ボリュームマネージャ構成の表示」を参照してください。

例 — ソフトパーティションの状態をチェックする

この例では、ソフトパーティション `d1` の状態をチェックします。このパーティションは 2 つのエクステンツから構成され、RAID 1 ボリューム `d100` 上に作成されています。

```
# metastat d1
d1: soft partition
   component: d100
   state: OKAY
   size: 42674285 blocks
```

```

                Extent                Start Block                Block Count
                0                        10234                    40674285
                1                        89377263                2000000
d100: Mirror
  Submirror 0: d10
  State: OKAY
  Read option: roundrobin (default)
  Write option: parallel (default)
  Size: 426742857 blocks

d10: Submirror of d100
  State: OKAY
  Hot spare pool: hsp002
  Size: 426742857 blocks
  Stripe 0: (interlace: 32 blocks)
    Device                Start Block  Dbase State                Hot Spare
    c3t3d0s0                0                No    Okay

```

▼ ソフトパーティションを拡張するには

ソフトパーティション上に他の論理ボリュームが構築されていない場合は、そのソフトパーティションに領域を追加できます。空き領域を見つけ、パーティションの拡張に使用します。既存のデータは移動されません。

注 - ソフトパーティションを使用して別のボリュームを作成している場合 (RAID 0 ボリュームのコンポーネントの場合など)、そのソフトパーティションは拡張できません。ソフトパーティションを収容するデバイスの領域を増やすことが目的であれば、通常、収容デバイスに他のボリュームを連結することによって目的を達成できます。詳細は、82 ページの「記憶領域の拡張」を参照してください。

1. 130 ページの「ソフトパーティションの背景情報」を確認します。
2. 次のどちらかの方法でソフトパーティションを拡張します。
 - Solaris 管理コンソール内の「拡張ディスク」から「ボリューム (Volumes)」ノードを開きます。拡張するソフトパーティションを選択し、「アクション (Action)」、「プロパティ (Properties)」の順に選択して、画面の指示に従ってパーティションを拡張します。詳細は、オンラインヘルプを参照してください。
 - 次の形式の `metattach` コマンドを使ってソフトパーティションに領域を追加します。

```
metattach [-s disk-set] soft-partition size
```

`disk-set` は、ソフトパーティションが含まれているディスクセットの名前です。

`soft-partition` は、既存のソフトパーティションの名前です。

size は追加する領域のサイズです。

例 — ソフトパーティションを拡張する

この例では、ソフトパーティションに領域を追加してから、そのソフトパーティションにあるファイルシステムを拡張します。ソフトパーティションはオンラインで、ファイルシステムはマウントされたままです。

```
# mount /dev/md/dsk/d20 /home2
# metattach d20 10g
# growfs -M /home2 /dev/md/rdisk/d20
```

ソフトパーティションを削除するには

1. 130 ページの「ソフトパーティションの背景情報」を確認します。
2. 次のどちらかの方法でソフトパーティションを削除します。
 - Solaris 管理コンソール内の「拡張ディスク」から「ボリューム (Volumes)」ノードを開きます。削除するソフトパーティションを選択し、「アクション (Action)」、「プロパティ (Properties)」の順に選択して、画面の指示に従ってパーティションを削除します。詳細は、オンラインヘルプを参照してください。
 - 次のいずれかの形式の `metaclear` コマンドを使ってソフトパーティションを削除します。

```
metaclear [-s disk-set] component
metaclear [-s disk-set] -r soft-partition
metaclear [-s disk-set] -p component
```

ここで使用されているオプション、変数は次のとおりです。

- *disk-set* は、ソフトパーティションが含まれているディスクセットの名前です。
- *soft-partition* は、削除するソフトパーティションの名前です。
- *r* は、論理ボリュームを繰り返し削除することを意味します。ただし、他のボリュームに使用されているボリュームは削除されません。
- *p* は、指定したコンポーネント上のソフトパーティションを削除することを意味します。ただし、開かれているソフトパーティションは除きます。
- *component* は、削除するソフトパーティションがあるコンポーネントです。

例 — ソフトパーティションを削除する

この例では、`c1t4d2s0` にあるすべてのソフトパーティションを削除します。

```
# metaclear -p c1t4d2s0
```


第 13 章

RAID 5 ボリューム (概要)

この章では、Solaris ボリュームマネージャの RAID 5 ボリュームの概念について説明します。関連する作業については、第 14 章を参照してください。

この章の内容は、次のとおりです。

- 139 ページの「RAID 5 ボリュームの概要」
- 143 ページの「RAID 5 ボリュームを作成するための背景情報」
- 145 ページの「RAID 5 ボリューム内のスライスの置き換えと有効化 (概要)」

RAID 5 ボリュームの概要

RAID レベル 5 は、パリティデータがすべてのコンポーネント (ディスクまたは論理ボリューム) に分散されている点を除き、ストライプ方式に似ています。コンポーネントに障害が発生した場合には、障害が発生したコンポーネント上のデータを、他のコンポーネント上に分散されているデータとパリティ情報から再構築することができます。Solaris ボリュームマネージャで、RAID 5 ボリュームは RAID レベル 5 をサポートするボリュームを意味します。

RAID 5 ボリュームは、そのボリュームの 1 つのコンポーネントに相当する記憶領域を使って、RAID 5 ボリュームの他のコンポーネントに格納されているユーザーデータの冗長情報 (パリティ) を保持します。つまり、3 つのコンポーネントがあれば、1 つのコンポーネントに相当する領域がパリティ情報に使用されます。同じように、5 つのコンポーネントがある場合にも、1 つのコンポーネントに相当する領域がパリティ情報に使用されます。パリティは、ボリューム内のすべてのコンポーネントに分散されます。ミラーと同様に、RAID 5 ボリュームではデータの可用性が向上しますが、ハードウェアのコストは最小限に抑えることができます。書き込み性能に対する影響は中程度です。ただし、RAID 5 ボリュームをルート (/)、/usr、および swap、あるいは既存のファイルシステムに対して使用することはできません。

既存のコンポーネントを置き換えると、Solaris ボリュームマネージャは、RAID 5 ボリュームの再同期を自動的に実行します。また、Solaris ボリュームマネージャは、システム障害やパニックが発生した場合、再起動時に、RAID 5 ボリュームを再同期します。

例 — RAID 5 ボリューム

図 13-1 に、RAID 5 ボリューム d40 を示します。

最初に 3 つのデータチャンクがディスク A から C に書き込まれ、次にパリティチャンクがディスク D に書き込まれます。パリティチャンクは、最初の 3 つのチャンクの排他的論理和を取ったものです。データチャンクとパリティチャンクをこのように書き込むことによって、データとパリティの両方が、RAID 5 ボリュームを構成するすべてのディスクに分散されます。各ドライブは個別に読み取ることができます。パリティ情報により、いずれか 1 つのディスクが故障しても、データの安全性が保証されます。この例の場合、各ディスクの容量が 2G バイトであれば、d40 の合計容量は 6G バイトになります (ディスク 1 つ分の領域がパリティ用に割り当てられます)。

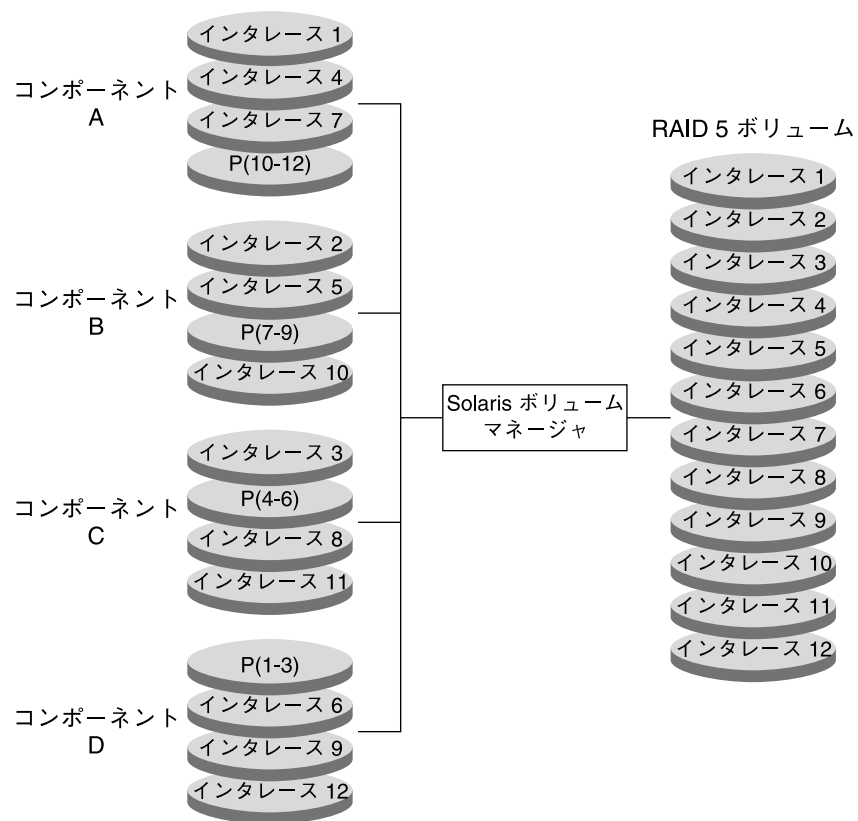


図 13-1 RAID 5 ボリュームの例

例 — RAID 5 ボリュームの連結 (拡張)

次の図に、4つのディスク (コンポーネント) から構成される RAID5 ボリュームに5つ目のディスクを動的に連結して拡張した例を示します。

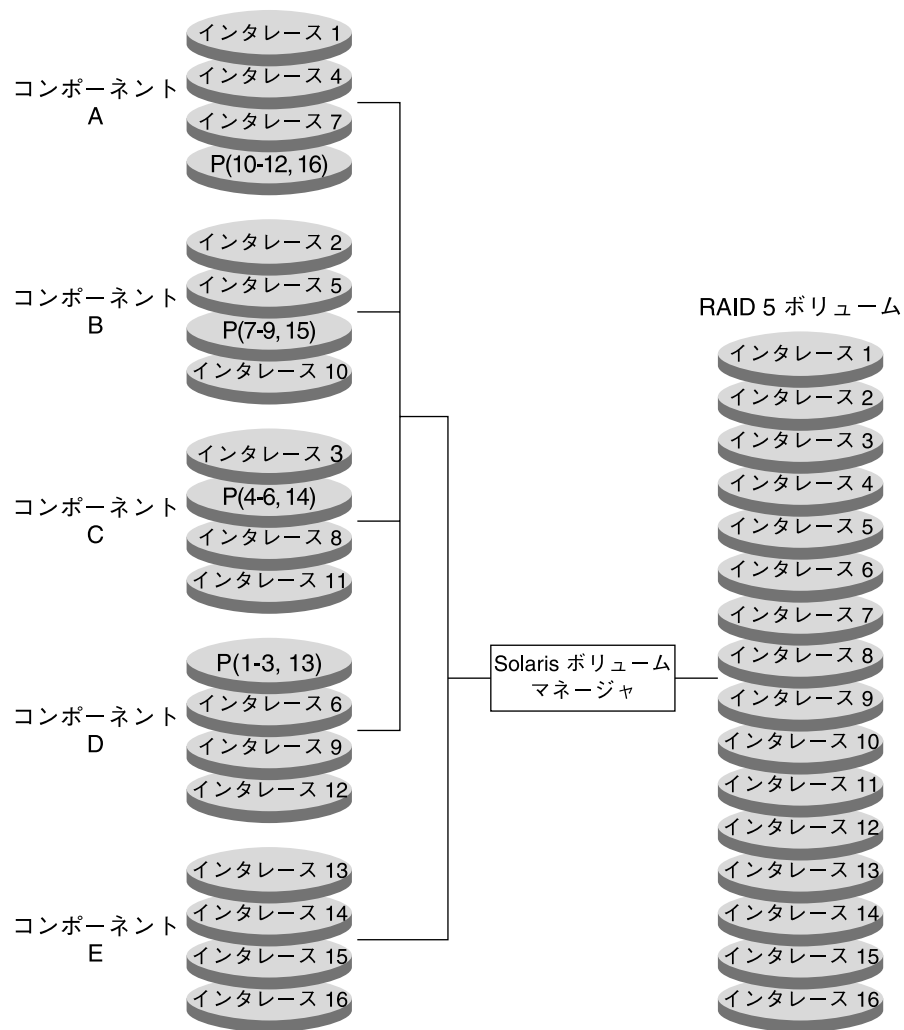


図 13-2 RAID 5 ボリュームの拡張例

パリティ領域は、RAID5 ボリュームの作成時に割り当てられます。パリティには1つのコンポーネントに相当する領域が割り当てられますが、実際のパリティブロックは、入出力を分散するためにすべてのオリジナルコンポーネントに分散されます。RAID 5 ボリュームにコンポーネントを連結すると、新しい領域はデータにのみ使用され、新しいパリティブロックは割り当てられません。ただし、連結されたコンポーネントのデータはパリティ計算の対象になるため、単一のデバイス障害からは保護されます。

連結した RAID 5 ボリュームは長期間の使用には適しません。このような RAID 5 ボリュームは、これよりも大規模な RAID 5 ボリュームを再構成し、そのボリュームにデータをコピーできるようになるまでの一時的な手段として使用します。

注 - RAID 5 ボリュームに新しいコンポーネントを追加すると、Solaris ボリュームマネージャは、そのコンポーネントのすべてのデータブロックを「ゼロ」にします。この処理は、パリティ情報によって新しいデータを保護するために実行されます。つまり、データが新しい領域に書き込まれると、Solaris ボリュームマネージャはそのデータをパリティ計算の対象とします。

RAID 5 ボリュームを作成するための背景情報

RAID 5 ボリュームを使用するときは、143 ページの「RAID 5 ボリュームの要件」と 144 ページの「RAID 5 ボリュームの指針」を考慮してください。また、RAID 5 ボリュームの構成には、ストライプ化に関する指針の多くが適用されます。これについては、77 ページの「ストライプ方式と連結方式の前提条件」を参照してください。

RAID 5 ボリュームの要件

- RAID 5 ボリュームは、少なくとも 3 つのコンポーネントから構成されていなければなりません。ただし、RAID 5 ボリュームのコンポーネントの数が多くなればなるほど、いずれかのコンポーネントに障害が発生したときの読み取りおよび書き込み時間は長くなります。
- RAID 5 ボリュームをストライプ化、連結、ミラー化することはできません。
- 既存のファイルシステムが格納されているコンポーネントから RAID 5 ボリュームを作成しないようにします。作成すると、RAID 5 の初期化時にデータが消去されます。
- RAID 5 ボリュームを作成する際に飛び越し値を設定できます。設定しないと、デフォルト値の 16K バイトが使用されます。この値は、ほとんどのアプリケーションにとって妥当な値です。
- RAID 5 ボリューム (ホットスペアなし) は、1 つのコンポーネントの障害にしか対応できません。
- RAID 5 ボリュームを作成するときは、別個のコントローラにあるコンポーネントを使用してください。これは、コントローラとそれに接続しているケーブルがディスクよりも故障する確率が高いためです。
- 同じサイズのコンポーネントを使用するようにします。サイズが異なるコンポーネントから RAID 5 ボリュームを作成すると、ディスク容量がむだになります。

RAID 5 ボリュームの指針

- RAID 5 ボリュームでは複雑なパリティ計算が必要なため、書き込みの割合が 20 パーセントを超えるボリュームには、RAID 5 ボリュームを使用しないようにします。書き込みが頻繁に発生するボリュームでデータの冗長性を確保したい場合は、ミラーの使用を検討してください。
- RAID 5 ボリューム内の各コンポーネントが異なるコントローラ上にあり、ボリュームへのアクセスが主に大容量の順次アクセスである場合は、飛び越し値を 32 K バイトに設定すると、性能が向上することがあります。
- RAID 5 ボリュームにコンポーネントを連結することによってボリュームを拡張できます。ただし、既存の RAID 5 ボリュームに新しいコンポーネントを連結すると、ボリュームの全体的な性能が低下します。これは、連結されたコンポーネント上のデータが順次処理されるためです。つまり、データは、すべてのコンポーネントにストライプ化されるわけではありません。ボリュームの元のコンポーネントのデータとパリティは、すべてのコンポーネントについてストライプ化されますが、このストライプ化は連結されたコンポーネントには適用されません。ただし、コンポーネントの入出力中はパリティが使用されますので、エラーが発生してもデータは復元されます。新しいコンポーネントが連結された RAID 5 ボリュームも 1 つのコンポーネントの障害にのみ対応できます。

連結されたコンポーネントは、それ自体のどの領域でもパリティがストライプ化されないという点で元のコンポーネントとは違います。連結されたコンポーネントでは、全内容がデータに使用されます。

コンポーネントを連結すると、大規模な書き込みや順次書き込みにおける性能上の利点は失われます。

- データブロックをゼロで初期化しなくても、RAID 5 ボリュームを作成することができます。そのためには、次のどちらかの方法を使用します。
 - `metainit` コマンドに `-k` オプションを指定します。`-k` オプションを指定すると、RAID 5 ボリュームが初期化なしで作成し直され、ディスクブロックが「正常 (Okay)」状態に設定されます。`-k` は、潜在的に危険なオプションです。ボリューム内のディスクブロックにエラーがあると、不正なデータの生成など、Solaris ボリュームマネージャが予期せぬ動作を起こすことがあります。
 - デバイスを初期化し、テープからデータを復元します。詳細は、`metainit (1M)` のマニュアルページを参照してください。

RAID 5 ボリューム内のスライスの置き換えと有効化 (概要)

Solaris ボリュームマネージャには、ミラーおよび RAID 5 ボリューム内のコンポーネントを置き換えたり、有効にしたりする機能があります。この機能についての問題点と要件は、ミラーおよび RAID 5 ボリュームに関するものと同じです。詳細は、244 ページの「RAID 1 および RAID 5 ボリューム内のコンポーネントの交換と有効化の概要」を参照してください。

シナリオ — RAID 5 ボリューム

RAID 5 ボリュームでは、RAID 1 ボリュームよりも少ないオーバーヘッドで記憶領域の冗長性を達成できます (RAID 1 ボリュームでデータの冗長性を得るには、合計記憶領域の 2 倍の領域が必要)。RAID 5 ボリュームでは、同じ数のディスクコンポーネントを使って RAID 1 ボリュームよりも大きい容量をもつ冗長記憶域を構成できます。さらに、ホットスペア (第 15 章 の特に 158 ページの「ホットスペアの仕組み」を参照) を使用すれば、RAID 1 とほぼ同じレベルの安全性が得られます。短所としては、書き込み時間の増加とコンポーネント障害時の大幅な性能低下が挙げられますが、多くの場合、このようなトレードオフが問題になることはありません。第 4 章のサンプルシステムに基づく構成例は、RAID 5 ボリュームによって追加の記憶容量がいかにか得られるかを示しています。

RAID 0 と RAID 1 ボリュームでは、2 つのコントローラに分散した 6 つのディスク上の 6 つのスライス (c1t1d0、c1t2d0、c1t3d0、c2t1d0、c2t2d0、c2t3d0) によって、27G バイトの冗長記憶域が得られます。しかし、RAID 5 構成では、同じスライスを使用することによって 45G バイトの冗長記憶域が得られる上、1 つのコンポーネントに障害が発生しても、データが失われたり、アクセスが中断されることはありません。さらに、RAID 5 ボリュームにホットスペアを追加すれば、複数のコンポーネントに障害が発生しても対応できます。このアプローチの最大の短所は、コントローラに障害が発生すると、RAID 5 ボリュームのデータが失われる点です。RAID 1 ボリュームでは、この問題は起こりません。詳細は、96 ページの「シナリオ — RAID 1 ボリューム (ミラー)」を参照してください。

第 14 章

RAID 5 ボリューム (作業)

この章では、RAID 5 ボリュームに関連する Solaris ボリュームマネージャの作業について説明します。これらの作業に伴う概念については、第 13 章を参照してください。

RAID 5 ボリューム (作業マップ)

次の表に、Solaris ボリュームマネージャの RAID 5 ボリュームを管理するのに必要な作業を示します。

作業	説明	参照先
RAID 5 ボリュームの作成	Solaris ボリュームマネージャの GUI か <code>metainit</code> コマンドを使って RAID 5 ボリュームを作成します。	148 ページの「RAID 5 ボリュームを作成するには」
RAID 5 ボリュームの状態チェック	Solaris ボリュームマネージャの GUI か <code>metastat</code> コマンドを使って、RAID 5 ボリュームの状態をチェックします。	149 ページの「RAID 5 ボリュームの状態をチェックするには」
RAID 5 ボリュームの拡張	Solaris ボリュームマネージャの GUI か <code>metattach</code> コマンドを使って RAID 5 ボリュームを拡張します。	152 ページの「RAID 5 ボリュームを拡張するには」
RAID 5 ボリュームのスライスの有効化	Solaris ボリュームマネージャの GUI か <code>metareplace</code> コマンドを使って、RAID 5 ボリュームのスライスを有効にします。	153 ページの「RAID 5 ボリューム内のコンポーネントを有効にするには」

作業	説明	参照先
RAID 5 ボリュームのスライスの交換	Solaris ボリュームマネージャの GUI か <code>metareplace</code> コマンドを使って、RAID 5 ボリュームのスライスを置き換えます。	154 ページの「RAID 5 ボリューム内のコンポーネントを置き換えるには」

RAID 5 ボリュームの作成

▼ RAID 5 ボリュームを作成するには

- 49 ページの「Solaris ボリュームマネージャ要素を作成するための前提条件」と 143 ページの「RAID 5 ボリュームを作成するための背景情報」を確認します。
- 次のどちらかの方法で RAID 5 ボリュームを作成します。
 - Solaris 管理コンソール内の「拡張ディスク」から「ボリューム (Volumes)」ノードを開き、「アクション (Action)」、「ボリュームの作成 (Create Volume)」の順に選択し、ウィザードの指示に従ってボリュームを作成します。詳細は、オンラインヘルプを参照してください。
 - 次の形式の `metainit` コマンドを使用します。

```
metainit name -r component component component
```

- `name` は、作成するボリュームの名前です。
- `r` は、RAID 5 ボリュームを作成することを意味します。
- `component` には、RAID 5 ボリュームに含めるスライスまたはソフトパーティションを指定します。

飛び越し値を指定する場合は、`-i interlace-value` オプションも指定します。詳細は、`metainit(1M)` のマニュアルページを参照してください。

例 — 3つのスライスから成る RAID 5 ボリュームを作成する

```
# metainit d45 -r c2t3d0s2 c3t0d0s2 c4t0d0s2
d45: RAID is setup
```

この例では、`-r` オプションを使って3つのスライスから成る RAID 5 ボリューム `d45` を作成します。`d45` の飛び越し値には、デフォルトの 16K バイトを使用します。RAID 5 ボリュームが設定されて、ボリュームの初期化が開始されたことを示すメッセージが表示されます。

注 - ボリュームの初期化が終わるまで RAID 5 ボリュームを使用することはできません。

次の作業

新たに作成した RAID 5 ボリュームにファイルシステムを作成する場合は、『Solaris のシステム管理 (基本編)』の「ファイルシステムの作成 (手順)」を参照してください。データベースなど、raw ボリュームを使用するアプリケーションは、独自の方法でこのボリュームを認識できなければなりません。

ホットスペアプールと RAID 5 ボリュームを関連付ける手順については、166 ページの「ホットスペア集合とボリュームを対応付けるには」を参照してください。

RAID 5 ボリュームの保守

▼ RAID 5 ボリュームの状態をチェックするには

- 次のどちらかの方法で RAID 5 ボリュームの状態をチェックします。
 - Solaris 管理コンソール内の「拡張ディスク」から「ボリューム (Volumes)」ノードを開き、ボリュームの状態を表示します。次に、ボリュームを選択し、「アクション (Action)」、「プロパティ (Properties)」の順に選択して、さらに詳しい情報を表示します。詳細は、オンラインヘルプを参照してください。
 - `metastat` コマンドを使用します。

`metastat` コマンドは、RAID 5 ボリュームのスライスごとに次の情報を表示します。

 - 「デバイス」(ストライプ内のスライスのデバイス名)
 - 「開始ブロック」(スライスの開始ブロック)
 - 「Dbase」(スライスに状態データベースの複製が含まれているかどうか)
 - 「状態」(スライスの状態)
 - 「ホットスペア」(障害のあるスライスのホットスペアとして使用されるスライス)

例 — RAID 5 ボリュームの状態を表示する

`metastat` コマンドで出力された、RAID 5 ボリュームの状態を以下に示します。

```
# metastat
d10: RAID
    State: Okay
    Interlace: 32 blocks
    Size: 10080 blocks
Original device:
    Size: 10496 blocks
    Device          Start Block  Dbase State      Hot Spare
    c0t0d0s1        330         No   Okay
    c1t2d0s1        330         No   Okay
    c2t3d0s1        330         No   Okay
```

metastat コマンドの出力には、ボリュームが RAID 5 ボリュームであることが示されています。また、RAID 5 ボリュームのスライスごとに、ストライプ内のスライスの名前、スライスの開始ブロック、スライスに状態データベースの複製が含まれていないこと、スライスが「正常」状態であること、障害のあるスライスのホットスペアとして使用されるスライスが存在しないことが示されています。

RAID 5 ボリュームの状態情報

次の表に、RAID 5 ボリュームの状態を示します。

表 14-1 RAID 5 の状態

状態	意味
初期化中 (Initializing)	スライスはすべてのディスクブロックをゼロで初期化しています。この処理は、データとパリティを飛び越し方式でストライプ化する RAID 5 ボリュームの特性上、必要になるものです。 状態が「正常」になったら、初期化が完了しており、デバイスを開くことができます。この状態になるまで、アプリケーションはエラーメッセージを受け取ります。
正常 (Okay)	今のところデバイスにはエラーはなく、使用可能な状態です。
保守 (Maintenance)	読み取りまたは書き込み操作で入出力エラーかオープンエラーが発生したため、1つのスライスがエラー状態になっています。

RAID 5 ボリュームのエラーに対処する場合は、おそらくスライスの状態がもっとも重要な情報となります。RAID 5 の状態は、「正常」や「保守が必要」などの一般的な状態情報を提供するだけです。RAID 5 の状態が「保守が必要」の場合は、スライスの状態を参照する必要があります。スライスの状態が「保守」の場合と「最後にエラー」の場合では、障害から回復するための処置が異なります。「保守」状態のスライスが1つだけ存在する場合は、データを失うことなくスライスを修理できます。「保守」状態のスライスと「最後にエラー」状態のスライスが1つずつある場合は、おそらくデータは破壊されています。この場合には、「保守」状態のスライスを修理してから「最後にエラー」状態のスライスを修理する必要があります。詳細は、244 ページの「RAID 1 および RAID 5 ボリューム内のコンポーネントの交換と有効化の概要」を参照してください。

次の表に、RAID 5 ボリュームのスライス状態と実行可能な処置を示します。

表 14-2 RAID 5 のスライスの状態

状態	意味	処置
初期化中 (Initializing)	スライスは、ディスクブ ロックをゼロで初期化して います。この処理は、デー タとパリティを飛び越し方 式でストライプ化する RAID 5 ボリュームの特性 上、必要になるものです。	通常は必要ありません。この間に入出力エ ラーが発生すると、デバイスの状態は「保 守」に変わります。初期化に失敗すると、 このボリュームの状態は「初期化失敗」 に、スライスの状態は「保守」になりま す。この場合には、ボリュームを削除して から作成し直す必要があります。
正常 (Okay)	デバイスにはエラーはな く、使用可能な状態です。	必要ありません。必要に応じて、スライス を追加および交換に使用できます。
再同期中 (Resyncing)	スライスの再同期処理が進 行しています。エラーが発 生したが、すでに訂正さ れ、スライスが有効に なっているか、あるいは、 新しいスライスが追加され た後です。	必要であれば、再同期が終了するまで RAID 5 ボリュームの状態を監視します。
保守 (Maintenance)	読み取りまたは書き込み操 作で入出力エラーかオープ ンエラーが発生したため、1 つのスライスがエラー状態 となっています。	障害が発生したスライスを有効にするか、 交換します。詳細は、153 ページ の「RAID 5 ボリューム内のコンポーネ ントを有効にするには」または 154 ページ の「RAID 5 ボリューム内のコンポーネ ントを置き換えるには」を参照してくださ い。metastat コマンドを実行すると、 metareplace コマンドを使って行うべ き処置を示す invoke 回復メッセージが 表示されます。
保守/最後にエラー (Maintenance/ Last Erred)	複数のスライスでエラーが 発生しました。エラーが発 生したスライスの状態 は「保守」か「最後にエ ラー」です。この状態の場 合、「保守」状態のスライ スには入出力は実行されな いが、「最後にエラー」状 態のスライスには実行され ます。この結果、入出力要 求の状態は、全体的に「最 後にエラー」となります。	障害が発生したスライスを有効にするか、 交換します。詳細は、153 ページ の「RAID 5 ボリューム内のコンポーネ ントを有効にするには」または 154 ページ の「RAID 5 ボリューム内のコンポーネ ントを置き換えるには」を参照してくださ い。metastat コマンドを実行すると、(- f フラグを指定した)metareplace コマ ンドを使って行うべき処置を示す invoke 回復メッセージが表示されます。この状態 は、複数のスライスに障害が発生したた め、不正なデータが生成された可能性があ ることを示しています。

注 – RAID 5 ボリュームの初期化や再同期化を中断することはできません。

▼ RAID 5 ボリュームを拡張するには

1. すべてのデータの最新のバックアップを取っているか確認します。また、この操作にはルート権限が必要です。
2. 143 ページの「RAID 5 ボリュームを作成するための背景情報」を確認します。
3. 次のどちらかの方法で RAID 5 ボリュームに他のコンポーネントを追加します。
 - Solaris 管理コンソール内の「拡張ディスク」から「ボリューム (Volumes)」ノードを開き、次に RAID 5 ボリュームを開きます。「コンポーネント (Components)」ペインを選択し、次に「コンポーネントを割り当て (Attach Component)」を選択して、指示に従ってコンポーネントを追加します。詳細は、オンラインヘルプを参照してください。
 - 次の形式の `metattach` コマンドを実行します。

```
metattach volume-name name-of-component-to-add
```

- `volume-name` は、拡張するボリュームの名前です。
- `name-of-component-to-add` は、RAID 5 ボリュームに追加するコンポーネントの名前です。

詳細は、`metattach(1M)` のマニュアルページを参照してください。

注 – 一般に、コンポーネントの追加は、領域が不足している RAID 5 ボリュームに対する一時的な解決策です。性能上の理由から、「純粋な」RAID 5 ボリュームの使用をお勧めします。

例 — RAID 5 ボリュームにコンポーネントを追加する

```
# metattach d2 c2t1d0s2
d2: column is attached
```

この例では、既存の RAID 5 ボリューム `d2` にスライス `c2t1d0s2` を追加します。

次の作業

UFS の場合は、RAID 5 ボリュームに対して `growfs` コマンドを実行します。詳細は、44 ページの「ボリュームとディスク領域の拡張」を参照してください。

データベースなど、raw ボリュームを使用するアプリケーションは、独自の方法で領域を拡張できなければなりません。

▼ RAID 5 ボリューム内のコンポーネントを有効にするには

1. すべてのデータの最新のバックアップを取っているか確認します。また、この操作にはルート権限が必要です。
2. 次のどちらかの方法で、**RAID 5** ボリューム内の、障害が発生したコンポーネントを有効にします。
 - Solaris 管理コンソール内の「拡張ディスク」から「ボリューム (Volumes)」ノードを開き、次に RAID 5 ボリュームを開きます。「コンポーネント (Components)」ペインを選択し、障害のあるコンポーネントを選択します。「コンポーネントを有効にする (Enable Component)」をクリックし、指示に従ってコンポーネントを有効にします。詳細は、オンラインヘルプを参照してください。
 - 次の形式の `metareplace` コマンドを実行します。

```
metareplace -e volume-name component-name
```

- `-e` は、同じ場所で障害が発生したコンポーネントを新しいコンポーネントで置き換える (おそらく、ディスクを物理的に交換した後で) ことを意味します。
- `volume-name` は、障害が発生したコンポーネントを含むボリュームの名前です。
- `component-name` は、置き換えられるコンポーネントの名前です。

`metareplace` は、新しいコンポーネントと RAID 5 ボリュームの残りのコンポーネントとの再同期を自動的に開始します。

例 — RAID 5 ボリューム内のスライスを実効にする

```
# metareplace -e d20 c2t0d0s2
```

この例では、RAID 5 ボリューム `d20` を構成するスライス `c2t0d0s2` にソフトエラーが発生したものとします。`metareplace` コマンドに `-e` オプションを指定して、このスライスを実効にします。

注 — ディスクドライブに障害が発生した場合は、そのスライスの代わりにシステムの他のディスク (およびスライス) を使用することができます (154 ページの「RAID 5 ボリューム内のコンポーネントを置き換えるには」を参照)。あるいは、そのディスクを修理または交換し、ディスクにラベルを付け、`metareplace` コマンドに `-e` オプションを指定して実行します。

▼ RAID 5 ボリューム内のコンポーネントを置き換えるには

この作業では、1つのスライスに障害が発生した RAID 5 ボリュームでそのスライスを交換します。



注意 – 複数のスライスでエラーが発生している状態で、障害のあるスライスを1つだけ交換すると、不正なデータが生成されることがあります。その場合のデータの完全性は保証されません。

1. すべてのデータの最新のバックアップを取っているか確認します。また、この操作にはルート権限が必要です。
2. 次のどちらかの方法で **RAID 5** ボリュームのどのスライスを交換するか判断します。
 - Solaris 管理コンソール内の「拡張ディスク」から「ボリューム (Volumes)」ノードを開き、次に RAID 5 ボリュームを開きます。「コンポーネント (Components)」ペインを選択して、個々のコンポーネントの状態を調べます。詳細は、オンラインヘルプを参照してください。
 - `metastat` コマンドを使用します。
「保守」というキーワードを探して、障害のあるスライスを特定します。
3. 次のどちらかの方法で、障害のあるスライスを別のスライスで置き換えます。
 - Solaris 管理コンソール内の「拡張ディスク」から「ボリューム (Volumes)」ノードを開き、次に RAID 5 ボリュームを開きます。「コンポーネント (Components)」ペインを選択し、障害のあるコンポーネントを選択します。「コンポーネントを置換 (Replace Component)」をクリックして、画面の指示に従ってコンポーネントを交換します。詳細は、オンラインヘルプを参照してください。
 - 次の形式の `metareplace` コマンドを実行します。

```
metareplace volume-name failed-component new-component
```

 - `volume-name` は、障害のあるコンポーネントが含まれるボリュームの名前です。
 - `failed-component` は、置き換えられるコンポーネントの名前です。
 - `new-component` は、障害のあるコンポーネントの代わりにボリュームに追加するコンポーネントの名前です。詳細は、`metareplace(1M)` のマニュアルページを参照してください。
4. 手順 2 のどちらかの方法で、新しいスライスの状態を確認します。
スライスは、「再同期中 (Resyncing)」状態か「正常 (Okay)」状態になるはずですが。

例 — RAID 5 コンポーネントを置き換える

```
# metastat d1
d1: RAID
State: Needs Maintenance
  Invoke: metareplace d1 c0t14d0s6 <new device>
  Interlace: 32 blocks
  Size: 8087040 blocks
Original device:
  Size: 8087520 blocks
Device          Start Block  Dbase State      Hot Spare
c0t9d0s6        330         No   Okay
c0t13d0s6       330         No   Okay
c0t10d0s6       330         No   Okay
c0t11d0s6       330         No   Okay
c0t12d0s6       330         No   Okay
c0t14d0s6       330         No   Maintenance

# metareplace d1 c0t14d0s6 c0t4d0s6
d1: device c0t14d0s6 is replaced with c0t4d0s6
# metastat d1
d1: RAID
  State: Resyncing
  Resync in progress: 98% done
  Interlace: 32 blocks
  Size: 8087040 blocks
Original device:
  Size: 8087520 blocks
Device          Start Block  Dbase State      Hot Spare
c0t9d0s6        330         No   Okay
c0t13d0s6       330         No   Okay
c0t10d0s6       330         No   Okay
c0t11d0s6       330         No   Okay
c0t12d0s6       330         No   Okay
c0t4d0s6        330         No   Resyncing
```

metastat コマンドを実行すると、RAID 5 ボリューム d1 内の不良スライスを修復するための処置が表示されます。交換用として使用可能なスライスを特定してから metareplace コマンドを実行します。このコマンドには、まず障害が発生したスライスを指定し、次に交換用のスライスを指定します。(使用可能なスライスがない場合は、metareplace コマンドに -e オプションを付けて実行し、障害のあるデバイスを再同期することによって、予想されるソフトエラーからの回復を試みます。) 複数のエラーがある場合は、まず「保守 (Maintenance)」状態のスライスを交換するか有効にする必要があります。その後で、「最後にエラー (Last Erred)」状態のスライスを修理します。metareplace コマンドの次に metastat コマンドを実行し、再同期の進捗状況を監視します。交換中は、ボリュームの状態と新しいスライスは「再同期中 (Resyncing)」状態となります。この状態の間は、ボリュームを使い続けることができます。

注 - metareplace コマンドを障害が発生していないデバイス上で実行すれば、ディスクスライスなどのコンポーネントを交換できます。この方法は、RAID 5 ポリユームの性能を調整するときなどに便利です。

第 15 章

ホットスペア集合 (概要)

この章では、Solaris ボリュームマネージャでホットスペア集合をどのように使用するかについて説明します。関連する作業については、第 16 章を参照してください。

この章では、以下の内容について説明します。

- 157 ページの「ホットスペア集合とホットスペアの概要」
- 158 ページの「ホットスペアの仕組み」
- 160 ページの「ホットスペア集合の管理」

ホットスペア集合とホットスペアの概要

ホットスペア集合とは、RAID 1 (ミラー) や RAID 5 ボリュームのデータ可用性を高めるために Solaris ボリュームマネージャが使用するスライスの集合 (ホットスペア) です。サブミラーまたは RAID 5 ボリュームのスライスに障害が発生すると、Solaris ボリュームマネージャはそのスライスをホットスペアで自動的に置き換えます。

注 - ホットスペアは、RAID 0 ボリュームや 1 面ミラーには適用されません。交換を自動的に行うためには、冗長データが必要です。

アイドル状態のホットスペアにデータや状態データベースの複製を格納することはできません。ホットスペアは、対応付けられているボリューム内のスライスに障害が発生した場合には、ただちに使用できる状態でなければなりません。したがって、ホットスペアを使用するためには、システムが通常の動作を行うのに必要なディスクに加え、予備のディスクを用意しておく必要があります。

ホットスペア

ホットスペアとは、動作しているが、使用されていないスライス (ボリュームでない) のことです。ホットスペアは、サブミラーや RAID5 ボリュームのスライスに障害が発生したときに、ただちに交換できる予備のスライスとして用意されています。

ホットスペアはボリュームをハードウェア障害から保護します。RAID 1 や RAID 5 ボリュームのスライスに障害が発生すると、スライスは自動的にホットスペアで置き換えられ、同期が取られます。ホットスペアは、サブミラーや RAID 5 ボリュームのスライスが修復されるか交換されるまで一時的に使用されるスライスです。

ホットスペアはホットスペア集合内に作成します。同じホットスペアを複数のホットスペア集合に登録することができます。たとえば、サブミラーとホットスペアが2つずつあるとします。この場合、これらのホットスペアを2つのホットスペア集合に登録し、それぞれの集合の中でホットスペアの優先順位を変えておくこともできます。これによって、最初に使用するホットスペアを指定でき、またホットスペアの数を増やしておけばデータの可用性を向上させることもできます。

サブミラーや RAID5 ボリュームの不良スライスの代わりに使用できるホットスペアのサイズは、そのスライスと同じかそれ以上でなければなりません。たとえば、サブミラーの容量が 1G バイトであれば、サブミラー用のホットスペアの容量は 1G バイト以上でなければなりません。

ホットスペアの仕組み

サブミラーや RAID 5 ボリュームのスライスに障害が発生すると、そのスライスの代わりに、対応付けられているホットスペア集合内のスライスが使用されます。Solaris ボリュームマネージャは、ホットスペア集合内でホットスペアを、登録させている順に探します。適切なサイズをもつ最初に見つかったホットスペアが不良スライスの置き換えに使用されます。ホットスペア集合内でのホットスペアの順序は、置き換えの後後も変わりません。

ヒント - ホットスペア集合にホットスペアを追加するときは、小さいものから順に追加してください。これによって、小さいスライスの置き換えのために「大きい」スライスがむだに使用されることを防げます。

スライスで入出力エラーが発生すると、そのスライスは「障害 (Broken)」状態になります。この状態から回復するためには、まず、不良スライスを修理または交換し、次に Solaris 管理コンソール内の「拡張ディスク」または `metahs -e` コマンドを使って、スライスを「使用可能 (Available)」状態に戻す必要があります。

サブミラーや RAID 5 ボリュームの不良スライスの代わりにホットスペアが使用されているときに、不良スライスを有効にするか交換すると、そのホットスペアはホットスペア集合内で「使用可能 (Available)」状態に戻り、再びホットスペアとして使用可能になります。

ホットスペア集合

ホットスペア集合は、ホットスペアの順序付きリスト (集合) です。

ホットスペアを複数のホットスペア集合に登録することによって、最小数のスライスで最大限の柔軟性と安全性を達成できます。つまり、ホットスペアとして使用する 1 つのスライスを複数のホットスペア集合に割り当てれば、個々のホットスペア集合でさまざまなスライスや特性をもつことができます。個々のホットスペア集合は、任意の数のサブミラーボリュームや RAID 5 ボリュームに割り当てることができます。

注 – 同じホットスペア集合を複数のサブミラーボリュームや RAID5 ボリュームに割り当ててはできませんが、個々のサブミラーボリュームや RAID5 ボリュームは 1 つのホットスペア集合しか割り当ててはできません。

入出力エラーが発生すると、Solaris ボリュームマネージャは、障害のあるスライスとサイズが同じかそれ以上のサイズの最初のホットスペアをホットスペア集合から探します。適切なホットスペアが見つかり、Solaris ボリュームマネージャはそのホットスペアを「使用中 (In-Use)」の状態にして、データの同期化を自動的に開始します。ホットスペアの同期化には、サブミラーの場合は有効なサブミラーのデータが使用され、RAID 5 ボリュームの場合は、同じボリュームの他のスライスが使用されます。十分な大きさのホットスペアがホットスペア集合にないと、障害が発生したサブミラーや RAID 5 ボリュームはエラー状態となり、ホットスペアは使用されません。この場合、サブミラーではデータを完全に複製することができなくなり、RAID 5 ボリュームではデータの冗長性が失われます。

例 — ホットスペア集合

図 15-1 に、ミラー d1 のサブミラー d11 と d12 に対応付けられているホットスペア集合 hsp000 を示します。どちらかのサブミラーのスライスに障害が発生すると、そのスライスは自動的にホットスペアで置き換えられます。ホットスペア集合自体はミラーではなく、個々のサブミラーボリュームに対応付けられています。必要であれば、このホットスペア集合を他のサブミラーや RAID5 ボリュームに対応付けることもできます。

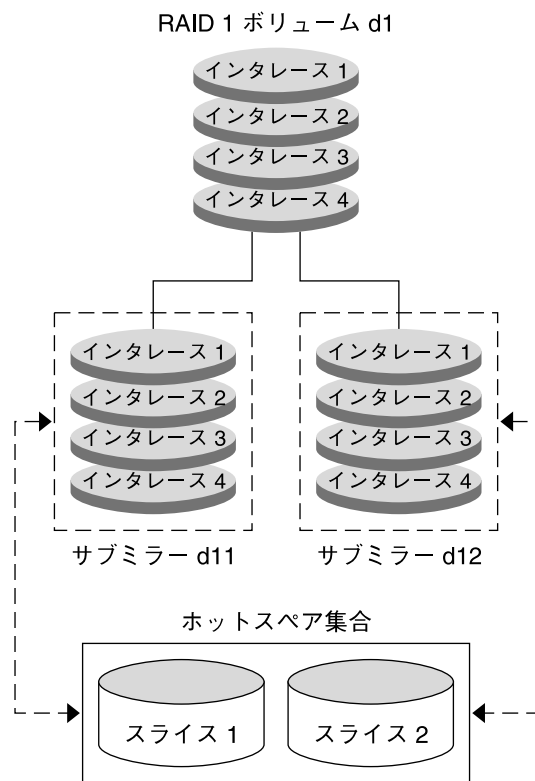


図 15-1 ホットスペア集合の例

ホットスペア集合の管理

Solaris ボリュームマネージャでは、ホットスペア集合内のホットスペアを動的に追加、削除、交換、および有効化することができます。ホットスペアやホットスペア集合の管理には、Solaris 管理コンソールまたはコマンド行ユーティリティを使用します。関連する作業の詳細については、第 16 章を参照してください。

シナリオ — ホットスペア

ホットスペアは、冗長ボリューム (RAID 1 および RAID 5) の保護機能を強化し、データの安全性をさらに向上します。ホットスペアと、RAID 0 サブミラーや RAID 5 ボリュームを構成するスライスを対応付けることによって、障害が発生したスライスは、ホットスペア集合内のホットスペアで自動的に置き換えられます。不良スライスの代わりに使用されるホットスペアは、正しい情報で更新され、元のスライスと同じように動作します。それらのホットスペアは、修理されたスライスといつでも交換できます。

第 16 章

ホットスペア集合 (作業)

この章では、Solaris ボリュームマネージャのホットスペアとホットスペア集合の使用方法について説明します。ホットスペア集合の概念については、第 15 章を参照してください。

ホットスペア集合 (作業マップ)

次の表に、Solaris ボリュームマネージャ のホットスペア集合を管理するのに必要な作業を示します。

作業	説明	参照先
ホットスペア集合を作成する	Solaris ボリュームマネージャ の GUI か <code>metainit</code> コマンドを使ってホットスペア集合を作成します。	164 ページの「ホットスペア集合を作成するには」
ホットスペア集合にスライスを追加する	Solaris ボリュームマネージャ の GUI か <code>metahs</code> コマンドを使ってホットスペア集合にスライスを追加します。	165 ページの「ホットスペア集合にホットスペアを追加するには」
ホットスペア集合とボリュームを対応付ける	Solaris ボリュームマネージャ の GUI か <code>metaparam</code> コマンドを使ってホットスペア集合とボリュームを対応付けます。	166 ページの「ホットスペア集合とボリュームを対応付けるには」
ホットスペア集合とボリュームの対応付けを変更する	Solaris ボリュームマネージャ の GUI か <code>metaparam</code> コマンドを使ってホットスペア集合とボリュームの対応付けを変更します。	168 ページの「ホットスペア集合の対応付けを変更するには」

作業	説明	参照先
ホットスペアとホットスペア集合の状態をチェックする	Solaris ボリュームマネージャの GUI か、 <code>metastat</code> または <code>metahs -i</code> コマンドを使って、ホットスペアやホットスペア集合の状態をチェックします。	169 ページの「ホットスペア集合とホットスペアの状態を確認するには」
ホットスペア集合内のホットスペアを交換する	Solaris ボリュームマネージャの GUI か <code>metahs</code> コマンドを使ってホットスペア集合内のホットスペアを交換します。	171 ページの「ホットスペア集合内のホットスペアを置き換えるには」
ホットスペア集合からホットスペアを削除する	Solaris ボリュームマネージャの GUI か <code>metahs</code> コマンドを使ってホットスペア集合からホットスペアを削除します。	172 ページの「ホットスペア集合からホットスペアを削除するには」
ホットスペアを有効にする	Solaris ボリュームマネージャの GUI か <code>metahs</code> コマンドを使ってホットスペア集合内のホットスペアを有効にします。	173 ページの「ホットスペアを有効にするには」

ホットスペア集合の作成

▼ ホットスペア集合を作成するには

- 49 ページの「Solaris ボリュームマネージャ要素を作成するための前提条件」を確認します。
- 次のどちらかの方法でホットスペア集合を作成します。
 - Solaris 管理コンソール内の「拡張ディスク」から「ホットスペアプール (Hot Spare Pools)」ノードを開き、「アクション (Action)」、「ホットスペアプールの作成 (Create Hot Spare Pool)」の順に選択します。詳細は、オンラインヘルプを参照してください。
 - 次の形式の `metainit` コマンドを使用します。

```
metainit hot-spare-pool-name ctds-for-slice
```

ここで、`ctds-for-slice` には、ホットスペア集合に割り当てるホットスペアを順に指定します。詳細は、`metainit (1M)` のマニュアルページを参照してください。

注 – ホットスペア集合の作成には、`metahs` コマンドを使用することもできます。

例 — ホットスペア集合を作成する

```
# metainit hsp001 c2t2d0s2 c3t2d0s2
hsp001: Hotspare pool is setup
```

この例では、ホットスペア集合 `hsp001` にホットスペアとして 2 つのディスクを割り当てます。ホットスペア集合が設定されたことを示すメッセージが出力されます。



注意 – 作成するホットスペアの大きさが十分であるかどうかを示すメッセージは出力されません。ホットスペアのサイズが、対応付けられているボリュームのサイズ以上でないと、ホットスペアは使用されません。

次の作業

ホットスペア集合にホットスペアを追加する手順については、165 ページの「ホットスペア集合にホットスペアを追加するには」を参照してください。ホットスペア集合を作成したら、それをサブミラーまたは RAID 5 ボリュームと対応付ける必要があります。詳細は、166 ページの「ホットスペア集合とボリュームを対応付けるには」を参照してください。

▼ ホットスペア集合にホットスペアを追加するには

1. 49 ページの「Solaris ボリュームマネージャ要素を作成するための前提条件」を確認します。
2. 次のどちらかの方法で既存のホットスペア集合にホットスペアを追加します。
 - Solaris 管理コンソール内の「拡張ディスク」から「ホットスペアプール (Hot Spare Pools)」ノードを開き、変更したいホットスペア集合を選択します。次に「アクション (Action)」、「プロパティ (Properties)」を選択し、「ホットスペア (Components)」パネルを選択します。詳細は、オンラインヘルプを参照してください。
 - 次の形式の `metahs` コマンドを使用します。

```
metahs -a hot-spare-pool-name slice-to-add
```

指定するホットスペア集合にスライスを追加する場合は、`hot-spare-pool-name` に `-a` を指定します。

すべてのホットスペア集合にスライスを追加する場合は、`hot-spare-pool-name` に `-all` を指定します。詳細は、`metahs (1M)` のマニュアルページを参照してください。

注 – 同じホットスペアを複数のホットスペア集合に追加することができます。ホットスペアをホットスペア集合に追加すると、ホットスペアは、ホットスペア集合のスライスの最後に追加されます。

例 — ホットスペアスライスを1つのホットスペア集合に追加する

```
# metahs -a hsp001 /dev/dsk/c3t0d0s2
hsp001: Hotspare is added
```

この例では、`-a` オプションを使ってスライス `/dev/dsk/c3t0d0s2` をホットスペア集合 `hsp001` に追加します。スライスがホットスペア集合に追加されたことを示すメッセージが表示されます。

例 — ホットスペアスライスをすべてのホットスペア集合に追加する

```
# metahs -a -all /dev/dsk/c3t0d0s2
hsp001: Hotspare is added
hsp002: Hotspare is added
hsp003: Hotspare is added
```

この例では、`-a` と `-all` オプションを使って、スライス `/dev/dsk/c3t0d0s2` をすべてのホットスペア集合に追加します。スライスがすべてのホットスペア集合に追加されたことを示すメッセージが表示されます。

ホットスペア集合とボリュームの対応付け

▼ ホットスペア集合とボリュームを対応付けるには

1. 49 ページの「**Solaris** ボリュームマネージャ要素を作成するための前提条件」を確認します。
2. 次のどちらかの方法でホットスペア集合と **RAID 5** ボリュームまたはサブミラーを対応付けます。

- Solaris 管理コンソール内の「拡張ディスク」から「ボリューム (Volumes)」を開き、ボリュームを選択します。「アクション (Action)」、「プロパティ (Properties)」の順に選択し、「ホットスペアプール (Hot Spare Pool)」パネルを選択して、「ボリュームに割り当て (Attach HSP)」を選択します。詳細は、オンラインヘルプを参照してください。
- 次の形式の `metaparam` コマンドを実行します。

```
metaparam -h hot-spare-pool component
```

```
-h          ホットスペア集合とボリュームを対応付けることを意味します。
hot-spare-pool  ホットスペア集合の名前です。
component     ホットスペア集合に対応付けるサブミラーまたは RAID 5 ボリュームの名前です。
```

詳細は、`metaparam(1M)` のマニュアルページを参照してください。

例 — ホットスペア集合とサブミラーを対応付ける

```
# metaparam -h hsp100 d10
# metaparam -h hsp100 d11
# metastat d0
d0: Mirror
   Submirror 0: d10
   State: Okay
   Submirror 1: d11
   State: Okay
...

d10: Submirror of d0
   State: Okay
   Hot spare pool: hsp100
...

d11: Submirror of d0
   State: Okay
   Hot spare pool: hsp100
...
```

この例では、`-h` オプションを指定して、ホットスペア集合 `hsp100` をミラー `d0` のサブミラー `d10` と `d11` に対応付けます。`metastat` コマンドを実行して、ホットスペア集合と2つのサブミラーが対応付けられていることを確認します。

例 — ホットスペア集合と RAID 5 ボリュームを対応付ける

```
# metaparam -h hsp001 d10
# metastat d10
d10: RAID
     State: Okay
     Hot spare pool: hsp001
...
```

-h オプションを指定して、ホットスペア集合 hsp001 と RAID 5 ボリューム d10 を対応付けます。metastat コマンドを実行して、ホットスペア集合と RAID 5 ボリュームが対応付けられていることを確認します。

▼ ホットスペア集合の対応付けを変更するには

1. 49 ページの「Solaris ボリュームマネージャ要素を作成するための前提条件」を確認します。
2. 次のどちらかの方法でホットスペア集合とボリュームの対応付けを変更します。
 - Solaris 管理コンソール内の「拡張ディスク」から「ボリューム (Volumes)」ノードを開き、ボリュームを選択します。「アクション (Action)」、「プロパティ (Properties)」の順に選択し、「ホットスペアプール (Hot Spare Pool)」パネルを選択します。画面の指示に従って不要なホットスペア集合を切り離してから、新しいホットスペア集合を対応付けます。詳細は、オンラインヘルプを参照してください。
 - 次の形式の metaparam コマンドを実行します。

```
metaparam -h hot-spare-pool-name RAID5-volume-or-submirror-name
```

-h ホットスペア集合の対応付けを変更することを意味します。

hot-spare-pool 新しいホットスペア集合の名前です。ホットスペア集合の対応付けを削除する場合は、特別なキーワード none を指定します。

component ホットスペア集合に対応付けるサブミラーまたは RAID 5 ボリュームの名前です。

詳細は、metaparam(1M) のマニュアルページを参照してください。

例 — ホットスペア集合の対応付けを変更する

```
# metastat d4
d4: RAID
     State: Okay
     Hot spare pool: hsp001
...
```



```
# metaparam -h hsp002 d4
# metastat d4
d4: RAID
    State: Okay
    Hot spare pool: hsp002
...
```

この例では、ホットスペア集合 hsp001 と RAID 5 ボリューム d4 がすでに対応付けられているものとします。このホットスペア集合の代わりにホットスペア集合 hsp002 をボリュームに対応付けます。metastat コマンドを実行して、ホットスペア集合の対応付けが変更されていることを確認します。

例 — ホットスペア集合の対応付けを削除する

```
# metastat d4
d4: RAID
    State: Okay
    Hot spare pool: hsp001
...
# metaparam -h none d4
# metastat d4
d4: RAID
    State: Okay
    Hot spare pool:
...
```

この例では、ホットスペア集合 hsp001 と RAID 5 ボリューム d4 がすでに対応付けられているものとします。ホットスペア集合の対応付けを none に変更します。これによって、このボリュームは、ホットスペア集合がまったく対応付けられていない状態になります。metastat コマンドを実行して、ホットスペア集合の対応付けが削除されていることを確認します。

ホットスペア集合の保守

▼ ホットスペア集合とホットスペアの状態を確認するには

- 次のどちらかの方法でホットスペア集合とそのホットスペアの状態を表示します。
 - Solaris 管理コンソール内の「拡張ディスク」から「ホットスペアプール (Spare Pools)」ノードを開き、ホットスペア集合を選択します。「アクション (Action)」、「プロパティ (Properties)」の順に選択して詳細なステータス情報を表示しま

す。詳細は、オンラインヘルプを参照してください。

- 次の形式の `metastat` コマンドを実行します。

```
metastat hot-spare-pool-name
```

注 - ホットスペア集合の状態を確認するには、`metahs` コマンドを使用することもできます。

例 — ホットスペア集合の状態を表示する

`metastat` コマンドを使ってホットスペア集合の状態を表示する例を以下に示します。

```
# metastat hsp001
hsp001: 1 hot spare
          c1t3d0s2                Available        16800 blocks
```

ホットスペア集合の状態

次の表に、ホットスペア集合の状態とそれに対する処置を示します。

表 16-1 ホットスペア集合の状態 (コマンド行)

状態	意味	処置
使用可能 (Available)	ホットスペアは正常であり、データを受け付けられる状態だが、今のところ書き込みも読み取りも行われていません。	必要ない
使用中 (In-use)	このホットスペア集合には、冗長ボリュームの不良コンポーネントの代わりに使用されているスライスが含まれています。	ホットスペアの使用状況を調べます。次に、このホットスペア集合によって置き換えられたボリュームのスライスを修理します。
障害 (Broken)	ホットスペアかホットスペア集合に問題があります。しかし、ただちにデータを失うおそれがあるわけではありません。すべてのホットスペアが使用中であったり、ホットスペアのどれかが故障している場合も、この状態になります。	ホットスペアの使用状況または故障の原因を調べます。必要であれば、別のホットスペアをホットスペア集合に追加することができます。


```
hsp003: Hotspare c0t2d0s2 is replaced with c3t1d0s2
```

最初に、`metastat` コマンドを使って、ホットスペアが使用中でないことを確認します。`metahs -r` コマンドは、ホットスペア集合 `hsp003` のホットスペア `/dev/dsk/c0t2d0s2` を `/dev/dsk/c3t1d0s2` で置き換えます。

例 — 対応付けられているすべてのホットスペア集合内のホットスペアを置き換える

```
# metahs -r all c1t0d0s2 c3t1d0s2
hsp001: Hotspare c1t0d0s2 is replaced with c3t1d0s2
hsp002: Hotspare c1t0d0s2 is replaced with c3t1d0s2
hsp003: Hotspare c1t0d0s2 is replaced with c3t1d0s2
```

この例では、キーワード `all` を使用することによって、対応付けられているすべてのホットスペア集合内のホットスペア `/dev/dsk/c1t0d0s2` を `/dev/dsk/c3t1d0s2` で置き換えます。

▼ ホットスペア集合からホットスペアを削除するには

1. 次のどちらかの方法で、ホットスペアがすでに使用されているかどうかを確認します。
 - Solaris 管理コンソール内の「拡張ディスク」から「ホットスペアプール (Spare Pools)」ノードを開き、ホットスペア集合を選択します。「アクション (Action)」、「プロパティ (Properties)」の順に選択し、「ホットスペア (Hot Spares)」パネルを選択して、画面の指示に従ってホットスペアの使用状況を調べます。詳細は、オンラインヘルプを参照してください。
 - 次の形式の `metastat` コマンドを実行します。

```
metastat hot-spare-pool-name
```

詳細は、`metastat (1M)` のマニュアルページを参照してください。
2. 次のどちらかの方法でホットスペアを削除します。
 - Solaris 管理コンソール内の「拡張ディスク」から「ホットスペアプール (Spare Pools)」ノードを開き、ホットスペア集合を選択します。「アクション (Action)」、「プロパティ (Properties)」の順に選択し、「ホットスペア (Hot Spares)」パネルを選択して、画面の指示に従ってホットスペアを削除します。詳細は、オンラインヘルプを参照してください。
 - 次の形式の `metahs` コマンドを使用します。

```
metahs -d hot-spare-pool-name current-hot-spare
```

<code>-d</code>	ホットスペア集合からホットスペアを削除することを意味します。
<code>hot-spare-pool</code>	ホットスペア集合の名前です。対応付けられているすべてのホットスペア集合を選択する場合は、特別なキーワード <code>all</code> を指定します。
<code>current-hot-spare</code>	削除するホットスペアの名前です。

詳細は、`metahs (1M)` のマニュアルページを参照してください。

例 — 1つのホットスペア集合からホットスペアを削除する

```
# metastat hsp003
hsp003: 1 hot spare
        c0t2d0s2                Broken        5600 blocks
# metahs -d hsp003 c0t2d0s2
```

最初に、`metastat` コマンドを使って、ホットスペアが使用中でないことを確認します。次に、`metahs -d` コマンドを使って、ホットスペア集合 `hsp003` からホットスペア `/dev/dsk/c0t2d0s2` を削除します。

▼ ホットスペアを有効にするには

- 次のどちらかの方法でホットスペアを「使用可能 (**available**)」状態に戻します。
 - Solaris 管理コンソール内の「拡張ディスク」から「ホットスペアプール (Spare Pools)」ノードを開き、ホットスペア集合を選択します。「アクション (Action)」、「プロパティ (Properties)」の順に選択し、「ホットスペア (Hot Spares)」パネルを選択して、画面の指示に従ってホットスペアを有効にします。詳細は、オンラインヘルプを参照してください。
 - 次の形式の `metahs` コマンドを使用します。

```
metahs -e hot-spare-slice
```

`-e` ホットスペアを有効にすることを意味します。

`hot-spare-slice` 有効にするスライスの名前です。

詳細は、`metahs (1M)` のマニュアルページを参照してください。

例 — ホットスペアを有効にする

```
# metahs -e c0t0d0s2
```

この例では、ホットスペア /dev/dsk/c0t0d0s2 を修理した後で、「使用可能 (Available)」状態に戻します。ホットスペア集合を指定する必要はありません。

第 17 章

トランザクションボリューム (概要)

この章では、2 種類のファイルシステムロギング (トランザクションボリュームと UFS ロギング) の概念について説明します。トランザクションボリュームに関連する作業については、第 18 章を参照してください。UFS ロギングについては、『Solaris のシステム管理 (基本編)』の「ファイルシステムのマウントとマウント解除 (手順)」を参照してください。

この章では、次の内容について説明します。

- 175 ページの「ファイルシステムロギングについて」
- 179 ページの「トランザクションボリュームの背景情報」
- 181 ページの「シナリオ — トランザクションボリューム」

注 - 将来の Solaris リリースでは、トランザクションボリュームは Solaris オペレーティング環境から削除される予定です。Solaris 8 リリースからサポートされている UFS ロギングは、トランザクションボリュームと同じ機能を備えており、より高い性能を提供します。UFS ロギングでは、システム管理の要件やオーバーヘッドが軽減されません。最適な性能と機能を得る上でこれらの利点は非常に重要です。

ファイルシステムロギングについて

ファイルシステムロギングには、トランザクションボリュームを使用する方法と UFS ロギングを使用する方法の 2 種類があります。

ファイルシステムロギングとは、UFS ファイルシステムを更新する前にその変更をログに書き込む処理のことです。ログに記録されたトランザクション情報は、後でファイルシステムに適用することができます。たとえば、新しいディレクトリを作成する場合、`mkdir` コマンドは、ログに記録されてからファイルシステムに適用されません。

システムの再起動時に、不完全なトランザクションは廃棄され、操作が完結しているトランザクションは適用されます。つまり、完結したトランザクションだけが適用されるために、ファイルシステムの整合性が常に保たれます。したがって、`fsck` コマンドを使ってファイルシステムの整合性をチェックする必要はありません。

システムクラッシュが起こると、実行中のシステムコールが中断され、UFS の整合性が損なわれます。`fsck` コマンドを実行せずに UFS をマウントすると、このような不整合によってシステム異常やデータの損傷が発生することがあります。

大規模なファイルシステムの整合性をチェックするには、ファイルシステムのデータを読み取り、検証する必要があるため、長い時間がかかります。UFS ロギングを使用すれば、完結しなかったシステムコールの呼び出しによる変更は破棄されるため、起動時に UFS ファイルシステムをチェックする必要はありません。

ロギング方法の選択

UFS ロギングとトランザクションボリュームの機能は、ファイルシステム情報のログを維持するという点では同じです。両者の主な違いは次のとおりです。

- トランザクションボリュームではログ情報を物理的に異なるデバイスに書き込むことができるが、UFS ロギングでは、ログとファイルシステムが同じボリュームに作成されます。
- UFS ロギングの性能は、どのような場合でもトランザクションボリュームよりも優れています。
- UFS ロギングではルート (/) を含むあらゆる UFS ファイルシステムのログをとることができるが、トランザクションボリュームではルート (/) ファイルシステムのログをとることはできません。

注 - 将来の Solaris リリースでは、トランザクションボリュームは Solaris オペレーティング環境から削除される予定です。Solaris 8 リリースからサポートされている UFS ロギングは、トランザクションボリュームと同じ機能を備えており、より高い性能を提供します。UFS ロギングでは、システム管理の要件やオーバーヘッドが軽減されます。最適な性能と機能を得る上でこれらの利点は非常に重要です。

UFS ロギングを有効にするには、ファイルシステムに対して `mount_ufs -logging` オプションを使用するか、`/etc/vfstab` ファイルを編集してファイルシステムのマウントオプションに `logging` を追加する必要があります。UFS ロギングが有効な状態でファイルシステムをマウントする手順については、『Solaris のシステム管理 (基本編)』の「ファイルシステムのマウントとマウント解除 (手順)」と `mount_ufs(1M)` のマニュアルページを参照してください。

トランザクションボリュームの使用方法については、以降の節をお読みください。

注 – UFS ファイルシステムのロギングを新たに使用する場合は、トランザクションボリュームではなく、UFS ロギングを使用してください。

トランザクションボリューム

トランザクションボリュームは、ファイルシステムのロギングを管理するためのボリュームで、本質的には UFS ロギングと同じです。どちらの方法でも、UFS の更新をファイルシステムに適用する前にログに記録します。

トランザクションボリュームは、2つのデバイスからなります。

- マスターデバイスは、ログをとるファイルシステムが格納されているスライスまたはボリュームです。
- ログデバイスは、ログが格納されているスライスまたはボリュームです。ログデバイスは、複数のファイルシステムで共有することができます。ログは、ファイルシステムへの変更を表すレコードの集合です。



注意 – ログデバイスやマスターデバイスは、物理的なスライスでも、ボリュームでもかまいません。ただし、信頼性と可用性を高めるために、ログデバイスには RAID 1 ボリューム (ミラー) を使用するようにします。これは、物理ログデバイスにデバイスエラーが発生すると、データが失われることがあるからです。マスターデバイスにも、RAID 1 や RAID 5 ボリュームを使用することができます。

トランザクションボリュームにログデバイスがあれば、ロギングは、トランザクションボリュームがマウントされたときに自動的に開始されます。マスターデバイスには、UFS ファイルシステムがあらかじめ格納されていてもかまいません (トランザクションボリュームを作成しても、マスターデバイスの内容が変更されることはありません)。あるいは、後でトランザクションボリュームにファイルシステムを作成することもできます。同じように、トランザクションボリュームを削除しても、マスターデバイスの UFS ファイルシステムが変更されることはありません。

トランザクションボリュームを構成したら、そのトランザクションボリュームを物理スライスや別の論理ボリュームと同じように使用できます。トランザクションボリュームの作成手順については、185 ページの「トランザクションボリュームの作成」を参照してください。

例 — トランザクションボリューム

次の図に、マスターデバイス d3 とミラー化されたログデバイス d30 から構成されているトランザクションボリューム d1 を示します。

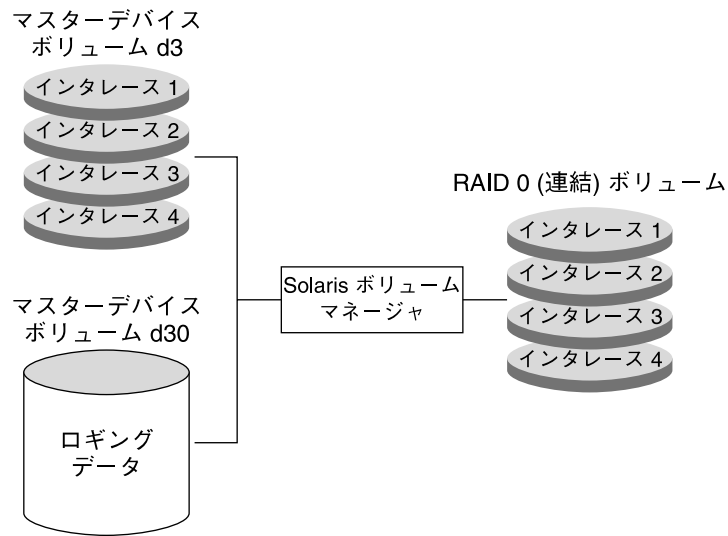


図 17-1 トランザクションボリュームの例

例 — ログデバイスの共有

次の図に、ミラー化されたログデバイス d30 を共有している 2 つのトランザクションボリューム d1 と d2 を示します。両方のマスターデバイスと共有ログデバイスはどれも RAID 1 ボリュームです。

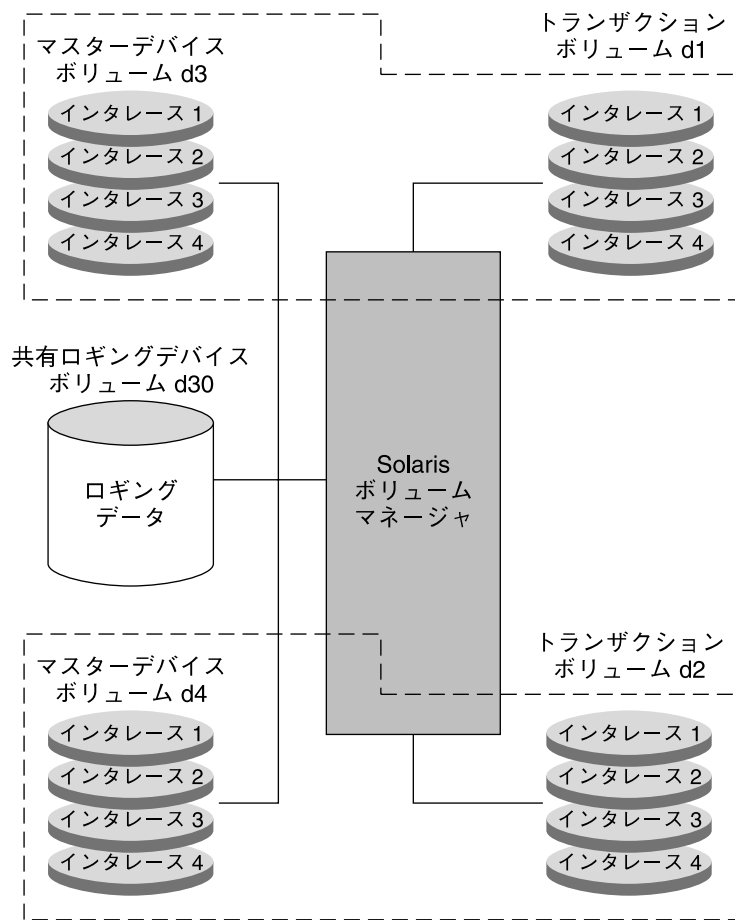


図 17-2 共有ログトランザクションボリュームの例

トランザクションボリュームの背景情報

トランザクションボリュームを使用する場合は、179 ページの「トランザクションボリューム用の要件」と 180 ページの「トランザクションボリュームの指針」を考慮してください。

トランザクションボリューム用の要件

トランザクションボリュームを使用する場合は、次の要件を考慮してください。

- トランザクションボリュームを作成する前に、マスターデバイスおよびログデバイスとして使用するスライスまたはボリュームを特定します。
- ルート (/) 以外の任意のファイルシステムをロギングできます。
- データの冗長性を確保するためにログデバイスをミラー化します。
- 負荷の高いディスクにログを置かないようにします。
- ログには少なくとも 1M バイトの記憶領域を割り当てます。(ログが大きければ、それだけ多くのファイルシステムのトランザクションを格納できます。) 100M バイトのファイルシステムデータごとに、1M バイトのログ領域を割り当てます(ログ領域は最大でも 64M バイトまでにします)。ログの最大領域は 1G バイトですが、ほとんどの場合、64M バイトを超えるログ領域が必要になることはなく、領域がむだになります。
- 同じトランザクションボリュームのログデバイスとマスターデバイスは、入出力負荷を均一にするために異なるドライブ(できれば異なるコントローラ)上に配置するようにします。
- トランザクションボリュームは、ログデバイスを共有できます。ただし、負荷の高いファイルシステムには独自のログを用意します。ログデバイスを共有した場合、ある種のエラーが発生すると、ログデバイスを共有するすべてのファイルシステムを fsck コマンドでチェックしなければならないことがあります。
- トランザクションボリュームを設定したら、ログデバイスをファイルシステム間で共有できます。
- ログ(ログデバイス)は通常、頻繁にアクセスされます。最適な性能を得るために、ログを使用頻度の高いディスクに置かないようにします。また、ログをディスクの中央に配置すると、ログにアクセスするときの平均シーク時間を最小にできます。
- ログサイズが大きいほど、性能が向上します。ログサイズが大きいほど、並列性(1秒間に実行されるファイルシステム操作の数)が高くなります。

注 - ログデバイスのミラー化を強くお勧めします。デバイスエラーによってログデバイスのデータが失われると、ファイルシステムの整合性が損なわれ、ユーザの介入がなければ、fsck では修復できなくなってしまうことがあります。マスターデバイスとして RAID 1 ボリュームを使用する方法は、データの冗長性を確保する上で有効です。

トランザクションボリュームの指針

- 一般には、もっとも大きな UFS ファイルシステムと、データの更新頻度が高くなるものも高い UFS ファイルシステムをロギングします。通常は、ほとんど読み取り操作だけが行われる小規模なファイルシステムをロギングする必要はありません。
- ログデバイス用のスライスがない場合でも、トランザクションボリュームを構成できます。これは、ログデバイス用のスライスが余分のないときに、エクスポートされたファイルシステムをロギングしたい場合に有用です。ログデバイス用にスライスが使用可能になったら、トランザクションボリュームにログデバイスを追加する

だけですみます。

- システムに十分な数の使用可能なスライスがない場合や、ログデバイスを共有するファイルシステム上の主な操作が書き込みではなく読み取りである場合は、ファイルシステムの間でログデバイスを共有することを検討してください。



注意 – ログデバイスを共有するマスターデバイスの1つがエラー状態になると、ログデバイスは変更を転送できなくなります。この問題が起こると、ログデバイスを共有するすべてのマスターデバイスがハードエラー状態になります。

シナリオ – トランザクションボリューム

トランザクションボリュームは、UFS ファイルシステムに対して UFS ログングと同様なログング機能を提供します。第4章のサンプルシステムに基づく構成例は、トランザクションボリュームのファイルシステムログングによって再起動の時間をいかに短縮できるかを示しています。

注 – トランザクションボリュームの特別な機能 (特に、ログデバイス以外のデバイスにログを記録する機能) を使用する必要がある場合を除き、UFS ログングの使用を検討してください。UFS ログングの性能は、トランザクションボリュームよりも優れています。

サンプルシステムには、ルート (/) や /var のミラーを始め、ログングによって最大限のアップタイムと可用性を確保する必要がある論理ボリュームがいくつかあります。ログの記録先として第3の RAID 1 ボリュームを使用するようにトランザクションボリュームを構成すると、冗長性を確保し、再起動の時間を短縮できます。

第 18 章

トランザクションボリューム (作業)

この章では、トランザクションボリュームに関連する作業について説明します。これらの作業に伴う概念については、第 17 章を参照してください。

注 - 将来の Solaris リリースでは、トランザクションボリュームは Solaris オペレーティング環境から削除される予定です。Solaris 8 リリースからサポートされている UFS ロギングは、トランザクションボリュームと同じ機能を備えており、より高い性能を提供します。UFS ロギングでは、システム管理の要件やオーバーヘッドが軽減されます。最適な性能と機能を得る上でこれらの利点は非常に重要です。

トランザクションボリューム (作業マップ)

次の表に、Solaris ボリュームマネージャのトランザクションボリュームを管理するのに必要な作業を示します。

作業	説明	参照先
トランザクションボリュームを作成する	Solaris ボリュームマネージャの GUI か <code>metainit</code> コマンドを使ってトランザクションボリュームを作成します。	185 ページの「トランザクションボリュームを作成するには」

作業	説明	参照先
トランザクションボリュームを UFS ログインに変換する	metaclear を使ってトランザクションボリュームを削除してから、mount コマンドを使って、UFS ログインを使用するファイルシステムをマウントします。	189 ページの「トランザクションボリュームを UFS ログインに変換するには」
トランザクションボリュームの状態をチェックする	Solaris ボリュームマネージャの GUI か metastat コマンドを使って、トランザクションボリュームの状態をチェックします。	192 ページの「トランザクションボリュームの状態をチェックするには」
トランザクションボリュームにログデバイスを接続する	Solaris ボリュームマネージャの GUI か metattach コマンドを使ってログデバイスを接続します。	195 ページの「トランザクションボリュームにログデバイスを接続するには」
トランザクションボリュームからログデバイスを切断する	Solaris ボリュームマネージャの GUI か metadetach コマンドを使ってログデバイスを切り離します。	196 ページの「トランザクションボリュームからログデバイスを切断するには」
トランザクションボリュームを拡張する	Solaris ボリュームマネージャの GUI か metattach コマンドを使ってトランザクションボリュームを拡張します。	197 ページの「トランザクションボリュームを拡張するには」
トランザクションボリュームを削除する	Solaris ボリュームマネージャの GUI か、metadetach または metarename コマンドを使ってトランザクションボリュームを削除します。	198 ページの「トランザクションボリュームを削除するには」
トランザクションボリュームの削除とマウントポイントの保持	Solaris ボリュームマネージャの GUI か metadetach コマンドを使ってトランザクションボリュームを削除します。	200 ページの「トランザクションボリュームを削除した状態でマウントデバイスを保持するには」
ログデバイスを共有する	Solaris ボリュームマネージャの GUI か metainit コマンドを使ってトランザクションボリュームのログデバイスを共有します。	202 ページの「ファイルシステム間でログデバイスを共有するには」
ファイルシステムに障害が発生した場合のトランザクションボリュームの回復	fsck コマンドを使って、障害のあるトランザクションボリュームを回復します。	204 ページの「パニックした場合のトランザクションボリュームを回復するには」
ハードウェアエラー時のトランザクションボリュームの回復	fsck コマンドを使って、ハードウェアのあるトランザクションボリュームを回復します。	204 ページの「ハードウェアエラー状態のトランザクションボリュームを回復するには」

トランザクションボリュームの作成

注 - 将来の Solaris リリースでは、トランザクションボリュームは Solaris 8 オペレーティング環境から削除される予定です。Solaris 8 リリースからサポートされている UFS ロギングは、トランザクションボリュームと同じ機能を備えており、より高い性能を提供します。UFS ロギングでは、システム管理の要件やオーバーヘッドが軽減されます。最適な性能と機能を得る上でこれらの利点は非常に重要です。

▼ トランザクションボリュームを作成するには

1. 49 ページの「Solaris ボリュームマネージャ要素を作成するための前提条件」と 179 ページの「トランザクションボリュームの背景情報」を確認します。
2. 可能であれば、ロギングを有効にする UFS ファイルシステムのマウントを解除します。

```
# umount /export
```

注 - ファイルシステムのマウントを解除できない場合は、後でトランザクションボリュームをアクティブにする前にシステムを再起動する必要があります。

3. 次のどちらかの方法でトランザクションボリュームを作成します。
 - Solaris 管理コンソール内の「拡張ディスク」から「ボリューム (Volumes)」ノードを開き、「アクション (Action)」、「ボリュームを作成 (Create Volume)」の順に選択し、ウィザードの指示に従ってトランザクションボリュームを作成します。詳細は、オンラインヘルプを参照してください。
 - 次の形式の `metainit` コマンドを使用します。

```
metainit trans-volume -t master-device log-device
```

- `trans-volume` は、作成するトランザクションボリュームの名前です。
- `master-device` は、ロギングするファイルシステムが格納されているデバイスの名前です。
- `log-device` は、ログを格納するデバイスの名前です。

ログデバイスやマスターデバイスは、スライスでも論理ボリュームでもかまいません。詳細は、`metainit (1M)` のマニュアルページを参照してください。

たとえば、スライス c0t0d0s6 にあるファイルシステムのログ格納先として c0t0d0s7 を指定し、トランザクションボリューム (d10) を作成するには、次のように指定します。

```
# metainit d10 -t c0t0d0s6 c0t0d0s7
```

注 – 同じログデバイス (この例では c0t0d0s7) をいくつかのマスターデバイスのために使用できます。つまり、ログデバイスの共有が完全にサポートされていません。

4. /etc/vfstab ファイルを編集して、**UFS** ファイルシステムの現在の情報を、作成したトランザクションボリュームの情報で置き換えます。

たとえば、/export が c0t0d0s6 にあり、新しいトランザクションボリュームが d10 であれば、/etc/vfstab を次のように編集します。これによって、マウントは、raw ディスクスライスではなく、トランザクションボリュームをポイントします。

```
#/dev/dsk/c0t0d0s6 /dev/rdisk/c0t0d0s6 /export ufs 2 yes -
/dev/md/dsk/d10 /dev/md/rdisk/d10 /export ufs 2 yes -
```

5. 可能であれば、ファイルシステムをマウントし直します。

注 – /usr など、マウント解除できないファイルシステムのトランザクションボリュームを作成する場合は、ここでシステムを再起動して、トランザクションボリュームをマウントし直し、ロギングを開始します。

例 — スライス用のトランザクションボリュームを作成する

```
# umount /home1
# metainit d63 -t c0t2d0s2 c2t2d0s1
d63: Trans is setup
(ファイルシステムがトランザクションボリュームを参照するように /etc/vfstab ファイルを編集する)
# mount /home1
```

スライス /dev/dsk/c0t2d0s2 には、/home1 にマウントされているファイルシステムが格納されています。ログデバイスとして使用するスライスは /dev/dsk/c2t2d0s1 です。最初にファイルシステムのマウントを解除します。metainit コマンドに -t オプションを指定して、トランザクションボリューム d63 を作成します。

次に、このファイルシステムのエントリがトランザクションボリュームを参照するように /etc/vfstab ファイルを編集します。

```
/dev/dsk/c0t2d0s2 /dev/rdisk/c0t2d0s2 /home1 ufs 2 yes -
```

上記の行を次のように変更します。

```
/dev/md/dsk/d63 /dev/md/rdisk/d63 /home1 ufs 2 yes -
```

ファイルシステムのロギングは、ファイルシステムをマウントし直したときに有効になります。

以降の再起動では、`fsck` コマンドは、このファイルシステムをチェックする代わりに、トランザクションボリュームに対するログメッセージを表示します。

```
# reboot
...
/dev/md/rdisk/d63: is logging
```

例 — /usr 用のトランザクションボリュームを作成する

```
# metainit -f d20 -t c0t3d0s6 c1t2d0s1
d20: Trans is setup
(ファイルシステムがトランザクションボリュームを参照するように /etc/vfstab ファイルを編集する)
```

```
# reboot
```

スライス `/dev/dsk/c0t3d0s6` にはファイルシステム `/usr` が格納されています。ログデバイスとして使用されるスライスは `/dev/dsk/c1t2d0s1` です。`/usr` はマウント解除できないため、`metainit` コマンドに `-f` オプションを指定して、トランザクションボリューム `d20` を強制的に作成します。次に、`/etc/vfstab` ファイルを編集して、このファイルシステムをマウントする行がトランザクションボリュームを参照するようにします。たとえば、次の行を見てください。

```
/dev/dsk/c0t3d0s6 /dev/rdisk/c0t3d0s6 /usr ufs 1 no -
```

これを次のように変更します。

```
/dev/md/dsk/d20 /dev/md/rdisk/d20 /usr ufs 1 no -
```

ファイルシステムのロギングは、システムの再起動時に有効になります。

例 — 論理ボリューム用のトランザクションボリュームを作成する

```
# umount /home1
# metainit d64 -t d30 d12
d64: Trans is setup
(ファイルシステムがトランザクションボリュームを参照するように /etc/vfstab ファイルを編集する)
```

```
# mount /home1
```

RAID 1 ボリューム `d30` には、`/home1` にマウントされているファイルシステムが格納されています。ログデバイスが格納されるミラーは `d12` です。最初にファイルシステムのマウントを解除します。`metainit` コマンドに `-t` オプションを指定して、トランザクションボリューム `d64` を作成します。

次に、`/etc/vfstab` ファイルを編集して、このファイルシステムをマウントする行がトランザクションボリュームを参照するようにします。たとえば、次の行を見てください。

```
/dev/md/dsk/d30 /dev/md/rdisk/d30 /home1 ufs 2 yes -
```

これを次のように変更します。

```
/dev/md/dsk/d64 /dev/md/rdisk/d64 /home1 ufs 2 yes -
```

ファイルシステムのロギングは、ファイルシステムをマウントし直したときに有効になります。

これ以降のファイルシステムの再マウントやシステムの再起動では、`fsck` はこのファイルシステムをチェックする代わりに、トランザクションボリュームに対するログメッセージを表示します。

```
# reboot
...
/dev/md/rdisk/d64: is logging
```

注 - `/etc/vfstab` ファイルの編集を避けたい場合は、`metarename (1M)` コマンドを使って、元の論理ボリュームの名前と新しいトランザクションボリュームを入れ換えることができます。詳細は、235 ページの「ボリューム名の変更」を参照してください。

トランザクションボリュームを UFS ロギングに変換

既存のトランザクションボリュームを UFS ロギングに変換すれば、性能と保守性を向上できます。また、トランザクションボリュームは将来サポートされなくなるため、いつかは UFS ロギングに移行する必要があります。次の各項ではこの変換手順について説明します。

トランザクションボリュームを UFS ロギングに変換するには

注 - UFS ロギングに移行するためには、ログ用に 1M バイト以上の空き領域が必要です (デフォルトのシステム設定を使用する場合)。これはログが、ログボリュームに置かれるためです。十分な空き領域がない場合は、ファイルを削除するか、ファイルシステムを拡張してから変換手順を実行する必要があります。

▼ トランザクションボリュームを UFS ロギングに変換するには

1. `metastat` コマンドを使って、トランザクションボリュームと、対応するログデバイスを特定し、出力中で `Trans` と `Logging device` を探します。

```
# metastat
d2: Trans
  State: Okay
  Size: 2869209 blocks
  Master Device: d0
  Logging Device: d20

d20: Logging device for d2
  State: Okay
  Size: 28470 blocks

d20: Concat/Stripe
  Size: 28728 blocks
  Stripe 0: (interlace: 32 blocks)
    Device          Start Block  Dbase State      Reloc  Hot Spare
    d10              0           No   Okay       No
    d11              0           No   Okay       No
    d12
```

後で必要になるので、これらの名前を書き留めておきます。

2. `df` コマンドを使って、`Trans device` がマウントされているかチェックします。出力中でトランザクションボリュームの名前を探します。トランザクションボリュームがマウントされていない場合は、手順 7 に進みます。

```
# df | grep d2
/mnt/transvolume (/dev/md/dsk/d2 ): 2782756 blocks 339196 files
```

3. `df -k` コマンドを使って、トランザクションボリュームに十分な空き容量があることを確認します。

```
# df -k /mnt/transvolume
file system          kbytes  used  avail capacity  Mounted on
/dev/md/dsk/d2      1391387 91965 1243767    7%    /mnt/transvolume
```

4. アプリケーションを停止するか、システムをシングルユーザーモードに移行して、ファイルシステム上のすべてのアクティビティを停止します。

```
# init s
[root@lexicon:lexicon-setup]$ init s
INIT: New run level: S
The system is coming down for administration. Please wait.
Dec 11 08:14:43 lexicon syslogd: going down on signal 15
Killing user processes: done.

INIT: SINGLE USER MODE

Type control-d to proceed with normal startup,
(or give root password for system maintenance):
single-user privilege assigned to /dev/console.
Entering System Maintenance Mode

Dec 11 08:15:52 su: 'su root' succeeded for root on /dev/console
Sun Microsystems Inc. SunOS 5.9 s81_51 May 2002
#
```

5. `lockfs -f` を使って、ファイルシステムのログを消去します。

```
# /usr/sbin/lockfs -f /mnt/transvolume
```

6. ファイルシステムをマウント解除します。

```
# umount /mnt/transvolume
```

7. ファイルシステムを含むトランザクションボリュームを削除します。
この操作を実行しても、ファイルシステム上のデータに影響はありません。

```
# metaclear d2
d2: Trans is cleared
```

この手順の始めで特定した Logging device はもはや使用されないで、他の目的に使用できます。しかし、この手順の始めに特定したマスターデバイスにはファイルシステムが格納されているため、マスターデバイスはその後マウントされ使用されません。

8. `/etc/vfstab` ファイルを編集して、ファイルシステムのマウント情報を変更します。

`raw` およびブロックマウントポイントを変更し、ファイルシステムのオプションに `logging` を追加する必要があります。トランザクションボリュームが使用されている場合には、`/etc/vfstab` ファイルのエントリは次のようになっています。

```
/dev/md/dsk/d2 /dev/md/rdisk/d2 /mnt/transvolume ufs 1 no -
```

マウントポイントをトランザクションボリューム `d2` からそのデバイス `d0` に変更し、ロギングオプションを追加した後の `/etc/vfstab` のエントリは次のようになります。

```
#device      device      mount      FS      fsck      mount      mount
#to mount    to fsck      point      type    pass      at boot    options
#
/dev/md/dsk/d0 /dev/md/rdisk/d0 /mnt/transvolume ufs 1 no logging
```

9. ファイルシステムをマウントし直します。

```
# mount /mnt/transvolume
```

注 - mount コマンドから次のような「 /dev/md/dsk/d0 が正常ではありません。読み取り／書き込みとマウントしようとして、fsck を実行し、再度行なってください。」が返されることがあります。この場合には、raw デバイス (fsck /dev/md/rdisk/d0) に対して fsck コマンドを実行し、y を入力して、スーパーブロック内のファイルシステム状態を修正する必要があります。その後で mount コマンドを再び実行します。

10. マウントしたファイルシステムのロギングが有効になっているか確認します。そのためには、/etc/mnttab ファイルを調べ、ファイルシステムのオプションとしてロギングが指定されていることを確認します。

```
# grep mnt /etc/mnttab
mnttab /etc/mnttab      mntfs      dev=43c0000      1007575477
/dev/md/dsk/d0 /mnt/transvolume      ufs rw,intr,largefiles,
logging,xattr,onerror=panic,suid,dev=1540000 1008085006
```

11. この手順の中でシステムをシングルユーザーモードに移行している場合は、ここでマルチユーザーモードに戻ることができます。

例 — トランザクションボリュームを UFS ロギングに変換する

トランザクションボリュームを UFS ロギングに変換する例を以下に示します。

```
# metastat
d50: Trans
State: Okay
Size: 204687 blocks
Master Device: c1t14d0s0
Logging Device: c1t12d0s0

      Master Device      Start Block  Dbase Reloc
      c1t14d0s0          0            No      Yes

c1t12d0s0: Logging device for d50
State: Okay
Size: 30269 blocks

      Logging Device      Start Block  Dbase Reloc
```

```
clt12d0s0          5641      No    Yes
```

この後のステップで必要になるので、マスターデバイスとログデバイスの名前を書き留めておく。

トランザクションボリュームに、マウントされたファイルシステムが格納されているか判別する。

```
# df | grep d50
/home1              (/dev/md/dsk/d50  ): 161710 blocks   53701 files
```

十分な空き領域 (1M バイト以上) があることを確認する。

```
# df -k /home1
filesystem          kbytes   used   avail capacity  Mounted on
/dev/md/dsk/d50     95510   14655   71304    18%    /home1
```

シングルユーザーモードにする。

```
# /usr/sbin/lockfs -f /home1
# /usr/sbin/umount /home1
# /usr/sbin/metaclear d50
```

d50: Trans is cleared

/etc/vfstab ファイルを編集して、関連するボリュームをマウントし、ロギングオプションを追加する。

```
# cat /etc/vfstab
#device            device                mount   FS   fsck   mount
mount
#to mount          to fsck                point   type pass   at boot
options
/dev/dsk/clt14d0s0 /dev/rdsk/clt14d0s0  /home1  ufs  2     yes
logging
```

```
# mount /home1
# /usr/bin/grep /home1 /etc/mnttab
/dev/dsk/clt14d0s0 /home1 ufs
rw,intr,largefiles,logging,xattr,onerror=panic,suid,dev=740380
1008019906
```

マルチユーザーモードに戻る。

トランザクションボリュームの保守

▼ トランザクションボリュームの状態をチェックするには

- 次のどちらかの方法でトランザクションボリュームの状態をチェックします。

- Solaris 管理コンソール内の「拡張ディスク」から「ボリューム (Volumes)」ノードを開き、ボリュームの状態を表示します。トランザクションボリュームを右クリックし、「プロパティ (Properties)」を選択して、さらに詳しい状態情報を表示します。詳細は、オンラインヘルプを参照してください。
- `metastat` コマンドを使用します。
詳細は、`metastat (1M)` のマニュアルページを参照してください。

例 — トランザクションボリュームの状態をチェックする

`metastat` コマンドを使って表示したトランザクションボリュームの出力例を以下に示します。

```
# metastat
d20: Trans
    State: Okay
    Size: 102816 blocks
    Master Device: c0t3d0s4
    Logging Device: c0t2d0s3

    Master Device      Start Block  Dbase
    c0t3d0s4           0            No

c0t2d0s3: Logging device for d0
    State: Okay
    Size: 5350 blocks

    Logging Device      Start Block  Dbase
    c0t2d0s3            250         No
```

`metastat` コマンドを使用して、マスターデバイスとログデバイスの情報を表示します。デバイスごとに次の情報が表示されます。

- 「デバイス (Device)」はスライスまたはボリュームのデバイス名です。
- 「開始ブロック (Start Block)」はデバイスの開始ブロックです。
- 「Dbase」は、デバイスに状態データベースの複製が含まれているかどうかを示します。
- 「状態 (State)」はログデバイスの状態です。

次の表に、トランザクションボリュームの状態と実行可能な処置を示します。

表 18-1 トランザクションボリュームの状態

状態	説明	処置
正常 (Okay)	デバイスは正常に機能している。マウントされているファイルシステムに対してはロギングが実行されているため、起動時にファイルシステムのチェックは行われません。	必要ない
接続中 (Attaching)	ログデバイスは、トランザクションボリュームが閉じられるかマウント解除されたときに、トランザクションボリュームに接続されます。接続されると、デバイスは「正常 (Okay)」状態に移行します。	必要ない
切断済み (Detached)	トランザクションボリュームにはログデバイスがない。URS ロギングの利点は無効になっています。	起動時に <code>fsck</code> コマンドがデバイスを自動的にチェックする。詳細は、 <code>fsck (1M)</code> のマニュアルページを参照。
切断中 (Detaching)	ログデバイスは、トランザクションボリュームが閉じられるかマウント解除されたときに、トランザクションボリュームから切断されます。切断されると、デバイスは「切断済み (Detached)」状態になります。	必要ない
ハードウェアエラー (Hard Error)	デバイスの使用中にデバイスエラーまたはパニックが発生した。デバイスが閉じられるかマウント解除されるまで、すべての読み取りと書き込みに対して入出力エラーが返されます。デバイスは、最初に開かれたときに「エラー (Error)」状態に移行します。	トランザクションボリュームを修復します。詳細は、204 ページの「パニックした場合のトランザクションボリュームを回復するには」か 204 ページの「ハードウェアエラー状態のトランザクションボリュームを回復するには」を参照。

表 18-1 トランザクションボリュームの状態 (続き)

状態	説明	処置
エラー (Error)	デバイスは読み書き可能。ファイルシステムを読み取り専用でマウントできるが、実際には読み取りや書き込みを行うたびに入出力エラーが返されます。読み取りや書き込みはデバイスエラーを受け取ります。この後でデバイスエラーが起っても、デバイスは「ハードウェアエラー (Hard Error)」状態には戻りません。	トランザクションボリュームを修復します。詳細は、204 ページの「パニックした場合のトランザクションボリュームを回復するには」か 204 ページの「ハードウェアエラー状態のトランザクションボリュームを回復するには」を参照。fsck または newfs コマンドが正常に終了すると、デバイスは「正常 (Okay)」状態に戻る。デバイスが「ハードウェアエラー (Hard Error)」か「エラー (Error)」状態にある場合は、システムの起動時に fsck コマンドがファイルシステムを自動的にチェックし、修復する。newfs コマンドを実行すると、デバイスのデータは破棄される。

▼ トランザクションボリュームにログデバイスを接続するには

- 49 ページの「Solaris ボリュームマネージャ要素を作成するための前提条件」と 179 ページの「トランザクションボリュームの背景情報」を確認します。
- ロギングを有効にする UFS ファイルシステムのマウントを解除します。
- 次のどちらかの方法でトランザクションボリュームにログデバイスを接続します。
 - Solaris 管理コンソール内の「拡張ディスク」から「ボリューム (Volumes)」ノードを開き、リストからトランザクションボリュームを選択します。ボリュームを右クリックし、「プロパティ (Properties)」を選択します。詳細は、オンラインヘルプを参照してください。
 - 次の形式の `metattach` コマンドを実行します。

```
metattach master-volume logging-volume
```

master-volume は、ロギングするファイルシステムが格納されているトランザクションボリュームの名前です。

logging-volume は、ログが格納されているボリュームまたはスライスの名前です。

詳細は、`metattach(1M)` のマニュアルページを参照してください。
- ファイルシステムを再びマウントします。

例 — トランザクションボリュームにログデバイスを接続する

この例では、ログデバイス、つまりスライス (c1t1d0s1) を、/fs2 にマウントされているトランザクションボリューム d1 に接続します。

```
# umount /fs2
# metattach d1 c1t1d0s1
d1: log device d0c1t1d0s1 is attached
# mount /fs2
```

▼ トランザクションボリュームからログデバイスを切断するには

1. 49 ページの「Solaris ボリュームマネージャ要素を作成するための前提条件」と 179 ページの「トランザクションボリュームの背景情報」を確認します。
2. ロギングを無効にしたり、ログデバイスを変更したい **UFS** ファイルシステムのマウントを解除します。
3. 次のどちらかの方法でトランザクションボリュームからログデバイスを切り離します。
 - Solaris 管理コンソール内の「拡張ディスク」から「ボリューム (Volumes)」ノードを開き、リストからトランザクションボリュームを選択します。ボリュームを右クリックし、「プロパティ (Properties)」を選択します。詳細は、オンラインヘルプを参照してください。
 - 次の形式の `metadetach` コマンドを使用します。

```
metadetach master-volume
```

`master-volume` は、ロギングするファイルシステムが格納されているトランザクションボリュームの名前です。

詳細は、`metadetach(1M)` のマニュアルページを参照してください。
4. ファイルシステムを再びマウントします。

例 — トランザクションボリュームからログデバイスを切り離す

この例では、ログデバイス、つまりスライス (c1t1d0s1) を、/fs2 にマウントされているトランザクションボリューム d1 から切り離します。

```
# umount /fs2
# metadetach d1
d1: log device c1t1d0s1 is detached
```

mount /fs2

▼ トランザクションボリュームを拡張するには

注 - トランザクションボリューム内のマスターデバイスを拡張するためには、マスターデバイスがボリューム (RAID 0、RAID 1、または RAID 5) でなければなりません。

1. 49 ページの「Solaris ボリュームマネージャ要素を作成するための前提条件」と 179 ページの「トランザクションボリュームの背景情報」を確認します。
2. マスターデバイスがボリューム (基本スライスではなく) の場合は、次のどちらかの方法でマスターデバイスに他のスライスを追加します。

- Solaris 管理コンソール内の「拡張ディスク」から「ボリューム (Volumes)」ノードを開き、リストからトランザクションボリュームを選択します。ボリュームを右クリックし、「プロパティ (Properties)」を選択してから、「コンポーネント (Components)」パネルを選択します。詳細は、オンラインヘルプを参照してください。
- 次の形式の `metattach` コマンドを実行します。

```
metattach master-volume component
```

`master-volume` は、ロギングするファイルシステムが格納されているトランザクションボリュームの名前です。

`component` は、接続するボリュームまたはスライスの名前です。

詳細は、`metattach(1M)` のマニュアルページを参照してください。

注 - マスターデバイスがミラーの場合は、個々のサブミラーにスライスを追加する必要があります。

3. マスターデバイスがスライスの場合は、そのスライスを直接拡張することはできません。その場合は、次のようにします。
 - 現在のトランザクションボリュームを削除します。
 - マスターデバイスのスライスをボリュームに含めます。
 - トランザクションボリュームを再び作成します。この処理が終わったら、上記の手順に従ってマスターデバイスを拡張します。

例 — トランザクションボリューム内の RAID 1 マスターデバイスを拡張する

```
# metastat d10
d10: Trans
    State: Okay
    Size: 102816 blocks
    Master Device: d0
    Logging Device: d1
d0: Mirror
    Submirror 0: d11
        State: Okay
    ...
    Submirror 1: d12
        State: Okay
    ...
# metattach d11 c0t2d0s5
d11: component is attached
# metattach d12 c0t3d0s5
d12: component is attached
```

この例では、トランザクションボリューム d10 を拡張します。このマスターデバイスは 2 面の RAID 1 ボリューム d0 で、サブミラー d11 と d12 を含んでいます。そのため、サブミラーごとに metattach コマンドを実行します。スライスが追加されたことを示すメッセージが表示されます。

次の作業

UFS の場合は、マスターデバイスではなくトランザクションボリュームに対して growfs コマンドを実行します。詳細は、243 ページの「ファイルシステムを拡張するには」を参照してください。

データベースなど、raw ボリュームを使用するアプリケーションは、独自の方法で領域を拡張できなければなりません。

▼ トランザクションボリュームを削除するには

1. 49 ページの「Solaris ボリュームマネージャ要素を作成するための前提条件」と 179 ページの「トランザクションボリュームの背景情報」を確認します。
2. トランザクションボリュームを削除し、ロギングを無効にしたい UFS ファイルシステムのマウントを解除します。

```
# umount /filesystem
```

3. 次のどちらかの方法でトランザクションボリュームからログデバイスを切り離します。

- Solaris 管理コンソール内の「拡張ディスク」から「ボリューム (Volumes)」ノードを開き、リストからトランザクションボリュームを選択します。ボリュームを右クリックし、「プロパティ (Properties)」を選択します。詳細は、オンラインヘルプを参照してください。
- 次の形式の `metadetach` コマンドを使用します。

```
metadetach master-volume
```

`master-volume` は、ロギングするファイルシステムが格納されているトランザクションボリュームの名前です。

詳細は、`metadetach(1M)` のマニュアルページを参照してください。

4. 次のどちらかの方法でトランザクションボリュームを削除 (クリア) します。

- Solaris 管理コンソール内の「拡張ディスク」から「ボリューム (Volumes)」ノードを開き、リストからトランザクションボリュームを選択します。ボリュームを右クリックし、「削除 (Delete)」を選択します。詳細は、オンラインヘルプを参照してください。
- 次の形式の `metaclear` コマンドを実行します。

```
metaclear master-volume
```

詳細は、`metaclear(1M)` のマニュアルページを参照してください。

5. 必要であれば、`/etc/vfstab` ファイルを編集して、上記の手順で削除したトランザクションボリュームではなく、トランザクションボリュームを構成するボリュームをマウントします。
6. ファイルシステムを再びマウントします。

例 — トランザクションボリュームを削除する

この例では、`/fs2` にマウントされているトランザクションボリューム `d1` を削除します。この処理が終わると、トランザクションボリュームを構成するスライス `c1t1d0s1` が直接マウントされます。

```
# umount /fs2
# metadetach d1
d1: log device d2 is detached
# metaclear d1
d1: Trans is cleared
```

(`/etc/vfstab` を編集して、`/fs2` に `c1t1d0s1` (`d1` ではなく) をマウントする。)

```
# mount /fs2
```

▼ トランザクションボリュームを削除した状態でマウントデバイスを保持するには

この手順は、トランザクションボリュームとそのボリュームを構成するデバイスが共に Solaris ボリュームマネージャの論理ボリュームである場合にのみ有効です。

1. 49 ページの「Solaris ボリュームマネージャ要素を作成するための前提条件」と 179 ページの「トランザクションボリュームの背景情報」を確認します。
2. トランザクションボリュームを削除し、ロギングを無効にしたい UFS ファイルシステムのマウントを解除します。
3. 次のどちらかの方法でトランザクションボリュームからログデバイスを切断します。

- Solaris 管理コンソール内の「拡張ディスク」から「ボリューム (Volumes)」ノードを開き、リストからトランザクションボリュームを選択します。ボリュームを右クリックし、「プロパティ (Properties)」を選択します。詳細は、オンラインヘルプを参照してください。
- 次の形式の `metadetach` コマンドを使用します。

```
metadetach master-volume
```

`master-volume` は、ロギングするファイルシステムが格納されているトランザクションボリュームの名前です。

詳細は、`metadetach(1M)` のマニュアルページを参照してください。

4. トランザクションボリュームの名前とマスターデバイスの名前を交換します。
5. 次のどちらかの方法でトランザクションボリュームを削除 (クリア) します。
 - Solaris 管理コンソール内の「拡張ディスク」から「ボリューム (Volumes)」ノードを開き、リストからトランザクションボリュームを選択します。ボリュームを右クリックし、「削除 (Delete)」を選択します。詳細は、オンラインヘルプを参照してください。
 - 次の形式の `metaclear` コマンドを実行します。

```
metaclear master-volume
```

詳細は、`metaclear(1M)` のマニュアルページを参照してください。

6. マスターデバイスに対して `fsck` コマンドを実行します。
スーパーブロック内のファイルシステムの状態を修正するかどうかの問い合わせに対して `y` を入力します。
7. ファイルシステムを再びマウントします。

例 — トランザクションボリュームを削除した状態でマウントデバイスを保持する

この例では、まず、マウントされたファイルシステムが格納されているトランザクションボリューム d1 の状態を調べ、最後に、トランザクションボリューム内のマスターデバイス d1 にファイルシステムをマウントします。

```
# metastat d1
d1: Trans
    State: Okay
    Size: 5600 blocks
    Master Device: d21
    Logging Device: d0

d21: Mirror
    Submirror 0: d20
        State: Okay
    Submirror 1: d2
        State: Okay
...

d0: Logging device for d1
    State: Okay
    Size: 5350 blocks
# umount /fs2
# metadetach d1
d1: log device d0 is detached
# metarename -f -x d1 d21
d1 and d21 have exchanged identities
# metastat d21
d21: Trans
    State: Detached
    Size: 5600 blocks
    Master Device: d1

d1: Mirror
    Submirror 0: d20
        State: Okay
    Submirror 1: d2
        State: Okay
# metaclear 21
# fsck /dev/md/dsk/d1
** /dev/md/dsk/d1
** Last Mounted on /fs2
** Phase 1 - Check Blocks and Sizes
** Phase 2 - Check Pathnames
** Phase 3 - Check Connectivity
** Phase 4 - Check Reference Counts
** Phase 5 - Check Cyl groups

FILE SYSTEM STATE IN SUPERBLOCK IS WRONG; FIX? y

3 files, 10 used, 2493 free (13 frags, 310 blocks, 0.5%
fragmentation)
```

```
# mount /fs2
```

最初に `metastat` コマンドを実行して、トランザクションボリューム `d1` が「正常 (Okay)」状態であることを確認します。トランザクションボリュームのログデバイスを切断する前に、ファイルシステムのマウントを解除します。-f (強制) フラグを指定して、トランザクションボリュームとそのミラー化されたマスターデバイスを交換します。metastat コマンドを再び実行して、交換が実際に行われたことを確認します。次に、トランザクションボリューム `d21` とログデバイス `d0` (必要な場合) を削除します。fsck コマンドをミラー `d1` に対して実行し、プロンプトの問い合わせに対して `y` を入力します。fsck コマンドの実行が終わったら、ファイルシステムを再びマウントします。/fs2 のマウントデバイスは変更されていないため、/etc/vfstab ファイルを編集する必要はありません。

ログデバイスの共有

▼ ファイルシステム間でログデバイスを共有するには

この手順では、別のファイルシステム用のログを含むトランザクションボリュームがすでに設定されているものとします。

1. 49 ページの「Solaris ボリュームマネージャ要素を作成するための前提条件」と 179 ページの「トランザクションボリュームの背景情報」を確認します。
2. 可能であれば、ロギングを有効にしたいファイルシステムのマウントを解除します。
3. ログデバイスがすでに存在している場合は、次のどちらかの方法でトランザクションボリュームから切断します。
 - Solaris 管理コンソール内の「拡張ディスク」から「ボリューム (Volumes)」ノードを開き、リストからトランザクションボリュームを選択します。ボリュームを右クリックし、「プロパティ (Properties)」を選択します。詳細は、オンラインヘルプを参照してください。
 - 次の形式の `metadetach` コマンドを使用します。

```
metadetach master-volume
```

詳細は、`metadetach(1M)` のマニュアルページを参照してください。
4. 次のどちらかの方法でトランザクションボリュームにログデバイスを接続します。
 - Solaris 管理コンソール内の「拡張ディスク」から「ボリューム (Volumes)」ノードを開き、リストからトランザクションボリュームを選択します。ボリュームを右クリックし、「プロパティ (Properties)」を選択します。詳細は、オンラインヘルプを参照してください。

ブを参照してください。

- 次の形式の `metattach` コマンドを実行します。

```
metattach master-volume logging-volume
```

詳細は、`metattach(1M)` のマニュアルページを参照してください。

5. `/etc/vfstab` ファイル内の、ファイルシステム用のエントリを変更（または追加）して、そのエントリがトランザクションボリュームを参照するようにします。
6. ファイルシステムをマウントし直します。ファイルシステムのマウントを解除できなかった場合は、システムを再起動して変更を有効にします。

例 — ログデバイスを共有する

```
# umount /xyzfs
# metainit d64 -t c0t2d0s4 d10
d64: Trans is setup
    (/etc/vfstab ファイルを編集して、/xyzfs のエントリがトランザクションボリューム d64 を参照する
    ようにする)
# mount /xyzfs
# metastat
...
d10: Logging device for d63 d64
...
```

この例では、他のトランザクションボリュームのログとして設定されているログデバイス (d10) を新しいトランザクションボリューム (d64) と共有します。マスターデバイスとして設定するファイルシステムは `/xyzfs` で、これにはスライス `/dev/dsk/c0t2d0s4` を使用します。 `metainit -t` コマンドを使用して、この構成がトランザクションボリュームであることを指定します。次に `/etc/vfstab` ファイル内の、ファイルシステム用のエントリを変更（エントリがなければ作成）して、そのエントリがトランザクションボリュームを参照するようにします。

```
/dev/dsk/c0t2d0s4 /dev/rdisk/c0t2d0s4 /xyzfs ufs 2 yes -
```

上記の行を次のように変更します。

```
/dev/md/dsk/d64 /dev/md/rdisk/d64 /xyzfs ufs 2 yes -
```

`metastat` コマンドを実行して、ログが共有されていることを確認します。ファイルシステムのロギングは、システムの再起動時に有効になります。

これ以降の再起動では、`fsck` コマンドは、このファイルシステムをチェックする代わりに、2つのファイルシステムについて次のようなメッセージを表示します。

```
/dev/md/rdisk/d63: is logging.
/dev/md/rdisk/d64: is logging.
```

エラー発生時のトランザクションボリュームの回復

▼ パニックした場合のトランザクションボリュームを回復するには

- `fsck` コマンドでファイルシステムを修復できない場合は、そのログデバイスを共有するファイルシステムが格納されているすべてのトランザクションボリュームに対して `fsck` コマンドを実行する必要があります。

例 — トランザクションボリュームを回復する

```
# fsck /dev/md/rdsk/trans
```

`fsck` コマンドは、関連するすべてのトランザクションボリュームをチェックし、修復してから、障害のあるトランザクションボリュームを「正常 (Okay)」状態にします。

▼ ハードウェアエラー状態のトランザクションボリュームを回復するには

この手順は、トランザクションボリュームを「正常 (Okay)」状態に戻すときに使用します。

トランザクションボリュームの状態をチェックする手順については、192 ページの「トランザクションボリュームの状態をチェックするには」を参照してください。

ロギングデータの処理中にマスターデバイスまたはログデバイスにエラーがあると、デバイスは「正常 (Okay)」状態から「ハードウェアエラー (Hard Error)」状態に移行します。デバイスが「ハードウェアエラー (Hard Error)」状態か「エラー (Error)」状態になると、デバイスエラーまたは異常が発生します。どちらの場合でも、回復方法は同じです。

注 - ログデバイスが共有されている場合に、トランザクションボリューム内のいずれかのスライスに異常が発生すると、そのトランザクションボリュームに関連しているすべてのスライスまたはボリュームが異常状態に切り換わります。

1. 49 ページの「Solaris ボリュームマネージャ要素を作成するための前提条件」と 179 ページの「トランザクションボリュームの背景情報」を確認します。

2. 179 ページの「トランザクションボリュームの背景情報」を読みます。
3. `lockfs` コマンドを実行して、ロックされているファイルシステムを特定します。

```
# lockfs
```

該当するファイルシステムはロックタイプが `hard` と表示されます。同じログデバイスを共有するファイルシステムはすべてハードロックされます。

4. 該当するファイルシステムのマウントを解除します。

ロックされているファイルシステムがエラー発生時に使用されていても、そのファイルシステムのマウントを解除できます。関連するプロセスが、ハードロックまたはマウント解除されているファイルシステム上の開かれたファイルまたはディレクトリにアクセスすると、エラーが返されます。
5. (省略可能) アクセス可能なデータのバックアップをとります。

デバイスエラーを修復する前に、通常は、できるだけ多くのデータを回復するようにします。`tar` や `cpio` コマンドを使用する場合のように、バックアップ手順を実行するためにはファイルシステムがマウントされていなければならない場合は、ファイルシステムを読み取り専用でマウントできます。`dump` や `volcopy` コマンドを使用する場合のように、ファイルシステムがマウントされている必要がない場合は、トランザクションボリュームに直接アクセスできます。
6. デバイスエラーを修復します。

この時点で、読み書き操作のためにトランザクションボリュームを開くかマウントすると、ログデバイス上にあるアクセス可能なすべてのデータが適切なマスターデバイスに転送されます。読み取りまたは書き込みできないデータは破棄されます。しかし、トランザクションボリュームを読み取り専用で開くかマウントすると、ログはスキャンされるだけでマスターデバイスに転送されません。したがって、エラーは修復されません。つまり、マスターデバイスとログデバイスのデータは、読み取りおよび書き込み操作のために最初に開かれるかマウントされるまで、変更されません。
7. `fsck` コマンドを実行してファイルシステムを修復します。あるいは、データを復元する場合は、`newfs` コマンドを実行します。

同じログデバイスを共有するすべてのトランザクションボリュームに対して `fsck` コマンドを実行します。`fsck` コマンドですべてのトランザクションボリュームが修復されると、各ボリュームは「正常 (Okay)」状態に戻ります。

`newfs` コマンドを使用しても、ファイルシステムは「正常 (Okay)」状態に戻ります。しかし、この場合には、ファイルシステムのすべてのデータが破棄されます。`newfs` コマンドは、通常、バックアップからファイルシステムを復元したいときに使用します。

同じログデバイスを共有するすべてのトランザクションボリュームに対して `fsck` または `newfs` コマンドを実行しないと、デバイスは「正常 (Okay)」状態に戻されません。
8. `metastat` コマンドを実行して、関連するデバイスが「正常 (Okay)」状態になっていることを確認します。

例 — ロギングデバイスのエラーを修復する

```
# metastat d5
d5: Trans
    State: Hard Error
    Size: 10080 blocks
    Master Device: d4
    Logging Device: c0t0d0s6

d4: Mirror
    State: Okay
...
c0t0d0s6: Logging device for d5
    State: Hard Error
    Size: 5350 blocks
...
# fsck /dev/md/rdisk/d5
** /dev/md/rdisk/d5
** Last Mounted on /fs1
** Phase 1 - Check Blocks and Sizes
** Phase 2 - Check Pathnames
** Phase 3 - Check Connectivity
** Phase 4 - Check Reference Counts
** Phase 5 - Check Cyl groups
WARNING: md: log device: /dev/dsk/c0t0d0s6 changed state to
Okay
4 files, 11 used, 4452 free (20 frags, 554 blocks, 0.4%
fragmentation)
# metastat d5
d5: Trans
    State: Okay
    Size: 10080 blocks
    Master Device: d4
    Logging Device: c0t0d0s6

d4: Mirror
    State: Okay
...

c0t0d0s6: Logging device for d5
    State: Okay
...
```

この例では、「ハードウェアエラー (Hard Error)」状態のログデバイスを含むトランザクションボリューム d5 を修復します。このトランザクションボリュームに対して fsck コマンドを実行すると、トランザクションボリュームは「正常 (Okay)」状態に戻されます。metastat コマンドを実行して、「正常 (Okay)」状態になっていることを確認します。

第 19 章

ディスクセット (概要)

この章では、ディスクセットの概念について説明します。関連する作業の実行手順については、第 20 章を参照してください。

この章では、次の内容について説明します。

- 207 ページの「ディスクセットの機能」
- 208 ページの「Solaris ボリュームマネージャによるディスクセットの管理」
- 212 ページの「ディスクセットの背景情報」
- 213 ページの「ディスクセットの管理」
- 215 ページの「シナリオ — ディスクセット」

ディスクセットの機能

共有ディスクセット (または単に ディスクセット) とは、排他的に共有される (複数のホストが同時に使用することはできない) ボリュームやホットスペアからなるディスクドライブの集まりです。ディスクセットには個別の名前空間が与えられ、Solaris ボリュームマネージャのボリュームはその名前空間内で管理できます。

ディスクセットを使用すると、データの冗長性と可用性が向上します。1 つのホストに障害が発生しても、他のホストがそのホストのディスクセットを引き継ぐことができます (このような構成をフェイルオーバー構成と呼びます)。各ホストはディスクセットを制御できますが、1 度に 1 台のホストだけが排他的にディスクセットを制御します。

注 - ディスクセットは、SPARC ベースのプラットフォームでも IA ベースのプラットフォームでもサポートされています。

注 - ディスクセットは、Sun Cluster、Solstice HA (High Availability)、またはサポートされる他社製の HA フレームワークと組み合わせて使用することを想定しています。Solaris ボリュームマネージャ単体では、フェイルオーバー構成を実装するのに必要なすべての機能が提供されるとは限りません。

Solaris ボリュームマネージャによる ディスクセットの管理

各ホストは、共有ディスクセットの他にローカルディスクセットも備えています。ローカルディスクセットは、ホスト上のすべてのディスクのうちで共有ディスクセットに含まれないディスクで構成されます。ローカルディスクセットは特定のホストだけに属します。ローカルディスクセットには、そのホストの構成を記録した状態データベースが格納されています。

共有ディスクセット内のボリュームやホットスペア集合は、そのディスクセット内のドライブだけで構成されます。ディスクセットに作成したボリュームは、物理スライスと同じように使用できます。ただし、ディスクセットでは、`/etc/vfstab` ファイルを介したファイルシステムのマウントはサポートされません。

ディスクセット内のボリューム上にあるファイルシステムを、起動時に `/etc/vfstab` ファイルを介して自動的にマウントすることはできません。これは、ディスクセットのマウント操作に必要な RPC デーモン (`rpc.metad` と `rpc.metamhd`) が、起動時にはまだ実行されていないためです。また、ディスクセットの所有権は再起動時に失われます。

共有ディスクセットの場合と同様に、ローカルディスクセット内のボリュームやホットスペア集合は、ローカルディスクセットのドライブだけで構成されます。

ディスクセットにディスクを追加すると、Solaris ボリュームマネージャはディスクセット上に状態データベースの複製を自動的に作成します。Solaris ボリュームマネージャは、ディスクセットの状態データベースの複製をそのディスクに配置できるように、ディスクのパーティションを再分割することがあります (209 ページの「ディスクの自動パーティション分割」を参照)。

ローカルディスクセットを管理する場合とは異なり、ディスクセットの状態データベースを手動で作成したり削除したりする必要はありません。Solaris ボリュームマネージャは、ディスクセット内のすべてのドライブに、1つの状態データベースの複製 (スライス7に常駐する) を分散させて配置します (ディスクセット当たり最大 50 の複製を作成できる)。

注 - ディスクセットはシングルホスト構成でもサポートされますが、通常、「ローカル」な (二重に接続されていない) 使用形態には適していません。例外的な使用形態として、ディスクセットを使って論理ボリュームの名前空間を管理しやすくする場合と、Storage Area Network (SAN) 構成においての記憶領域の管理を容易にする場合が挙げられます (215 ページの「シナリオ - ディスクセット」を参照)。

ディスクの自動パーティション分割

ディスクセットに新しいディスクを追加すると、Solaris ボリュームマネージャは、ディスクフォーマットを調べ、必要に応じてディスクのパーティションを再分割して、状態データベースの複製を格納できるように適切に設定されたスライス 7 を作成します。スライス 7 の厳密なサイズはディスクの幾何学的な構造によって異なりますが、通常は、4M バイトを下回ることなく、おおよそ 6M バイト程度です (シリンダ境界がどこにあるかによって異なります)。

注 - スライス 7 の最小限のサイズは、状態データベースの複製のサイズ、状態データベースの複製に保管する情報など、さまざまな要因によって、将来変わってくる可能性があります。

ディスクセットで使用するディスクには、次の条件を満たしたスライス 7 を与える必要があります。

- セクター 0 から始まる
- ディスクラベルと状態データベースの複製とを格納できる領域がある
- マウントできない
- スライス 2 などの他のスライスと重なり合わない

既存のパーティションテーブルがこれらの条件を満たしていない場合、Solaris ボリュームマネージャはディスクを再分割します。各ドライブで、Solaris ボリュームマネージャによって使用される小さい領域が、スライス 7 として確保されます。各ドライブ上の残りの領域はスライス 0 に割り当てられます。パーティションが再分割されると、ディスク上のすべてのデータが失われます。

ヒント - ディスクセットにドライブを追加した、ドライブのパーティションは必要に応じて再分割ができますが、スライス 7 を変更することはできません。

スライス 7 の最小限のサイズは、ディスクの幾何学的な構造によって異なりますが、つねに 4M バイト以上です。

以下に、prtvtoc コマンドで表示した、ディスクセットに追加する前のディスク情報の出力例を示します。

```

[root@lexicon:apps]$ prtvtoc /dev/rdisk/c1t6d0s0
* /dev/rdisk/c1t6d0s0 partition map
*
* Dimensions:
*   512 bytes/sector
*   133 sectors/track
*   27 tracks/cylinder
* 3591 sectors/cylinder
* 4926 cylinders
* 4924 accessible cylinders
*
* Flags:
* 1: unmountable
* 10: read-only
*
*
* Partition Tag  Flags      First      Sector      Last
* Partition Tag  Flags      Sector     Count       Sector  Mount Directory
*   0      2    00          0    4111695    4111694
*   1      3    01    4111695    1235304    5346998
*   2      5    01          0    17682084   17682083
*   3      0    00    5346999    4197879    9544877
*   4      0    00    9544878    4197879    13742756
*   5      0    00   13742757    3939327    17682083

```

注 – Solstice DiskSuite ソフトウェアで使用していたディスクセットが存在する場合、それらのディスクセット上の状態データベースの複製のデフォルトサイズは 1034 ブロックです。一方、Solaris ボリュームマネージャで使用されるデフォルトサイズは 8192 ブロックです。そのため、Solstice DiskSuite で追加されたディスクのスライス 7 は、Solaris ボリュームマネージャで追加されたディスクのスライス 7 よりも小さくなっています。

このディスクをディスクセットに追加すると、prtvtoc の出力は次のようになります。

```

[root@lexicon:apps]$ prtvtoc /dev/rdisk/c1t6d0s0
* /dev/rdisk/c1t6d0s0 partition map
*
* Dimensions:
*   512 bytes/sector
*   133 sectors/track
*   27 tracks/cylinder
* 3591 sectors/cylinder
* 4926 cylinders
* 4924 accessible cylinders
*
* Flags:
* 1: unmountable
* 10: read-only
*
*
* Partition Tag  Flags      First      Sector      Last
* Partition Tag  Flags      Sector     Count       Sector  Mount Directory

```

```

          0      0      00      10773  17671311  17682083
          7      0      01          0      10773      10772
[root@lexicon:apps]$

```

ディスクセットに追加するディスク上に適切なスライス7(シリンダ0から開始し、状態データベースの複製を格納できる十分な領域がある)がある場合は、ディスクのパーティション分割は行われません。

ディスクセットの命名規則

ディスクセットのコンポーネント名は Solaris ボリュームマネージャの他のコンポーネント名と似ていますが、ディスクセットのコンポーネント名には、その一部としてディスクセット名が含まれます。

- ボリュームパス名では、/dev/md/ とパス内の実際のボリューム名の間にはディスクセット名が入ります。
- 次の表に、ディスクセットボリューム名の例を示します。

表 19-1 ボリューム名の例

/dev/md/blue/dsk/d0	ディスクセット blue 内のブロック型ボリューム d0
/dev/md/blue/dsk/d1	ディスクセット blue 内のブロック型ボリューム d1
/dev/md/blue/rdsk/d126	ディスクセット blue 内の raw ボリューム d126
/dev/md/blue/rdsk/d127	ディスクセット blue 内の raw ボリューム d127

同様に、ホットスペア集合の場合も、その名前の一部にディスクセット名が含まれます。

例 — 2つの共有ディスクセット

図 19-1 に、2つの共有ディスクセットを使用する構成例を示します。

この構成では、ホスト A とホスト B がディスクセット A と B を共有しています。また、それぞれのホストには、共有されていないローカルディスクセットがあります。ホスト A に障害が発生すると、ホスト B がホスト A の共有ディスクセット (ディスクセット A) の制御を引き継ぎます。同じように、ホスト B に障害が発生すると、ホスト A がホスト B の共有ディスクセット (ディスクセット B) の制御を引き継ぎます。

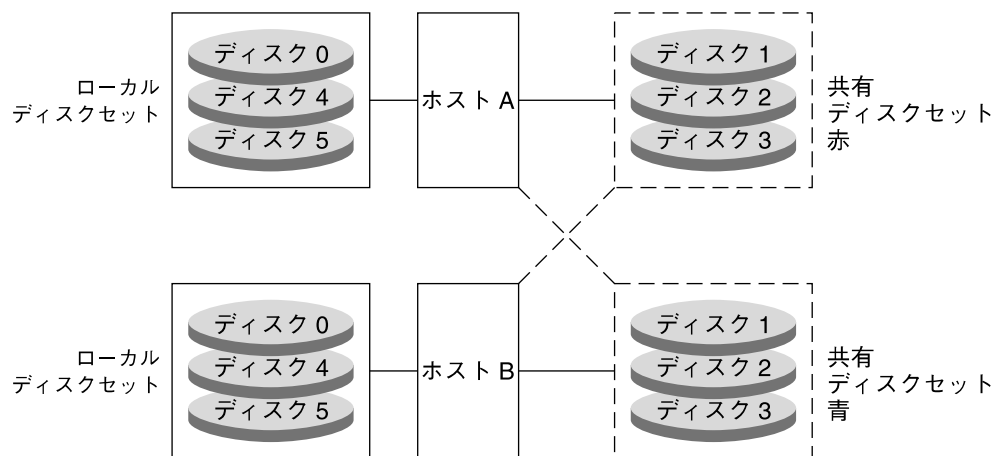


図 19-1 ディスクセットの例

ディスクセットの背景情報

ディスクセットを使用するときには、212 ページの「ディスクセットの背景情報」と 213 ページの「ディスクセットの管理」を参照してください。

ディスクセットの要件

- ディスクセットに接続されるホストごとに Solaris ボリュームマネージャを構成する必要があります。
- ディスクセットを作成するためには、各ホスト上にローカルの状態データベースが設定されていなければなりません。
- クラスタ環境でディスクセットを作成および使用する場合は、root がグループ (Group) 14 のメンバーでなければなりません。あるいは、各ホストの `/.rhosts` ファイルに、他のホスト名のエントリが含まれていなければなりません。
- ディスクセットの保守を行う場合は、ホストがディスクセットの所有者であるか、ディスクセットを予約していなければなりません。(ホストは、ディスクセットに最初のドライブを置くことによって、暗黙的にディスクセットの所有者となります)。
- 使用中のドライブをディスクセットに追加することはできません。ドライブを追加する前に、それがファイルシステムやデータベースなどのアプリケーションによって使用されていないことを確認します。
- 保存したいデータが格納されているドライブをディスクセットに追加しないようにします。ドライブをディスクセットに追加すると、パーティションが再分割され、データが破壊されます。

- ホスト間で共有しようとする、ディスクセット内のすべてのディスクは各ホストに接続されており、各ホスト上でまったく同じパス、ドライバ、名前を持っていないければなりません。特に、共有ディスクドライブは、両方のホスト上で同じデバイス番号 (c#t#d#) を持っていません。両方のホスト上で番号が異なると、ドライブをディスクセットに追加するときに、「drive c#t#d# is not common with host xxx」というメッセージが返されます。共有ディスクでは、同じドライバ名 (ssd) を使用する必要があります。ディスクセット内に共有ディスクドライブを設定する手順については、219 ページの「ディスクセットにドライブを追加するには」を参照してください。

ディスクセットの指針

- システムに設定できるデフォルトのディスクセット数は 4 です。
/kernel/drv/md.conf ファイルを編集すれば、この数を 32 に増やすことができます。これについては、241 ページの「デフォルトのディスクセット数を増やすには」を参照してください。共有ディスクセットの数は常に md_nsets の値から 1 を引いた数です。これは、md_nsets に、ローカルディスクセットが含まれているためです。
- ローカルボリュームの管理と異なり、ディスクセット上の状態データベースの複製を手動で作成したり削除したりする必要はありません。Solaris ボリュームマネージャがディスクセットの各ディスクに適切な数の状態データベースの複製を配置します。
- ディスクセットにドライブを追加すると、Solaris ボリュームマネージャは、既存のドライブにある状態データベースの複製の配置を再調整します。必要であれば、後で metadb コマンドを使って複製の配置を変更できます。

ディスクセットの管理

ディスクセットの作成と構成は、Solaris ボリュームマネージャのコマンド行インタフェース (metaset コマンド) または Solaris 管理コンソール内の「拡張ディスク」を使って行います。

ディスクセットにドライブを追加したら、そのディスクセットを共有するホストは、ディスクセットを予約 (確保) したり解放したりすることができます。他のホストは、予約されたディスクセットのデータにアクセスすることはできません。ディスクセットの保守を行うためには、そのホストがディスクセットを所有しているか予約していなければなりません。ホストは、ディスクセットに最初のドライブを設定することによって、暗黙的にディスクセットの所有者になります。

ディスクセットの取得

ホストは、ディスクセットのドライブを使用する前にディスクセットを取得する必要があります。ディスクセットを取得するには、次の2とおりの方法があります。

- 安全取得 - この方法でディスクセットを取得すると、Solaris ボリュームマネージャはこのディスクセットを確保しようとし、他のホストはこのディスクセットを解放しようとし、この解放 (つまり、取得) は失敗することもあります。
- 強制取得 - この方法でディスクセットを取得すると、Solaris ボリュームマネージャは、別のホストがこのディスクセットを取得しているかどうかに関係なく、ディスクセットを取得します。この方法は、通常、ディスクセットを共有するホストの1つが停止したり、他のホストと通信していないときに使用されます。これによって、そのディスクセットのすべてのディスクが新しいホストに引き継がれます。そして、この取得を行なったホストが状態データベースを読み込み、このディスクセットの共有ボリュームにアクセスできるようになります。この時点で他のホストがこのディスクセットをすでに取得していると、そのホストは取得が失われたためにパニックを起こします。

ディスクセットを共有する2つのホストは相互に協調しているため、一般には、ディスクセットのドライブが同時に2つのホストによって取得されることはありません。正常な状態とは、両方のホストが動作し、相互に通信している状態のことです。

注 - 取得されているはずのドライブが取得されていない状態になると (おそらく、このディスクセットを共有する別のホストがこのドライブを強制的に確保したことが原因)、そのホストはパニックを起こします。これによって、2つのホストが同時に同じドライブにアクセスした場合に起こり得るデータの消失が最小限に抑えられます。

ディスクセットの取得については、225 ページの「ディスクセットを取得するには」を参照してください。

ディスクセットの解放

ディスクセットの物理ドライブを保守する場合には、ディスクセットをあらかじめ解放しておくのが便利です。ディスクセットを解放すると、そのディスクセットはホストからアクセスできなくなります。ディスクセットを共有しているホストが両方ともディスクセットを解放すると、どちらのホストもそのディスクセットのドライブにアクセスできなくなります。

ディスクセットの解放については、227 ページの「ディスクセットを解放するには」を参照してください。

シナリオ — ディスクセット

第 4 章 のサンプルシステムに基づく構成例は、ディスクセットを使って SAN (Storage Area Network) 構成上に存在する記憶領域をどのように管理するかを示しています。

サンプルシステムには、ファイバスイッチと SAN 記憶領域に接続されたもう 1 つのコントローラがあるものとします。SAN 構成の記憶領域は、SCSI や IDE ディスクなどの他のデバイスと同様に起動処理の初期段階では使用できないため、Solaris ボリュームマネージャは起動時に SAN 構成上の論理ボリュームを使用不能とみなします。ただし、この記憶領域をディスクセットに追加し、ディスクセットツールを使って管理すれば、起動時の使用不能の問題は回避できます。また、SAN 構成上に接続されている記憶領域は、ディスクセットで制御された別個の名前空間内でローカル記憶領域から容易に管理できます。

第 20 章

ディスクセット (作業)

この章では、ディスクセットに関連する作業について説明します。これらの作業に伴う概念については、第 19 章を参照してください。

ディスクセット (作業マップ)

次の表に、Solaris ポリウムマネージャのディスクセットを管理するのに必要な作業を示します。

作業	説明	参照先
ディスクセットを作成する	Solaris ポリウムマネージャの GUI か <code>metaset</code> コマンドを使ってディスクセットを作成します。	218 ページの「ディスクセットを作成するには」
ディスクセットにドライブを追加する	Solaris ポリウムマネージャの GUI か <code>metaset</code> コマンドを使ってディスクセットにドライブを追加します。	219 ページの「ディスクセットにドライブを追加するには」
ディスクセットにホストを追加する	Solaris ポリウムマネージャの GUI か <code>metaset</code> コマンドを使ってディスクセットにホストを追加します。	221 ページの「ディスクセットにホストを追加するには」
ディスクセットに Solaris ポリウムマネージャのポリウムを作成する	Solaris ポリウムマネージャの GUI か <code>metainit</code> コマンドを使ってディスクセットにポリウムを作成します。	222 ページの「ディスクセットに Solaris ポリウムマネージャのコンポーネントを作成するには」

作業	説明	参照先
ディスクセットの状態をチェックする	Solaris ボリュームマネージャの GUI か <code>metastat</code> コマンドを使ってディスクセットの状態をチェックします。	223 ページの「ディスクセットの状態をチェックするには」
ディスクセットからボリュームを削除する	Solaris ボリュームマネージャの GUI か <code>metaset</code> コマンドを使ってディスクセットからボリュームを削除します。	224 ページの「ディスクセットからディスクを削除するには」
ディスクセットを取得する	Solaris ボリュームマネージャの GUI か <code>metaset</code> コマンドを使ってディスクセットを取得します。	225 ページの「ディスクセットを取得するには」
ディスクセットを解放する	Solaris ボリュームマネージャの GUI か <code>metaset</code> コマンドを使ってディスクセットを解放します。	227 ページの「ディスクセットを解放するには」
ディスクセットからホストを削除する	Solaris ボリュームマネージャの GUI か <code>metaset</code> コマンドを使ってディスクセットからホストを削除します。	228 ページの「ホストまたはディスクセットを削除するには」
ディスクセットを削除する	Solaris ボリュームマネージャの GUI か <code>metaset</code> コマンドを使ってディスクセットから最後のホストを削除し、ディスクセットを削除します。	228 ページの「ホストまたはディスクセットを削除するには」

ディスクセットの作成

▼ ディスクセットを作成するには

- 212 ページの「ディスクセットの背景情報」を確認します。
- 次のどちらかの方法でディスクセットを作成します。
 - Solaris 管理コンソール内の「拡張ディスク」から「ディスクセット (Disk Sets)」ノードを開きます。「アクション (Action)」、「ディスクセットを作成 (Create Disk Set)」の順に選択し、ウィザードの指示に従ってディスクセットを作成します。詳細は、オンラインヘルプを参照してください。
 - コマンド行から次の形式の `metaset` コマンドを実行して、ディスクセットを最初から作成します。

```
metaset [-s diskset-name ] [-a] [-h hostname]
```

- s *diskset-name* metaset コマンドが使用するディスクセットの名前です。
- a 指定したディスクセットにホストを追加します。Solaris ボリュームマネージャは1つのディスクセットで4つのホストをサポートします。
- h *hostname* ディスクセットに追加する1つまたは複数のホストを指定します。最初のホストを追加すると、ディスクセットが作成されます。2つめのホストは後で追加することができます。ただし、ディスクセット内のすべてのドライブが指定した *hostname* 内に存在しなければなりません。 *hostname* は、`/etc/nodename` ファイルに指定されている名前と同じでなければなりません。

詳細は、`metaset (1M)` のマニュアルページを参照してください。

3. `metaset` コマンドを実行して新しいディスクセットの状態をチェックします。

```
# metaset
```

例 — ディスクセットを作成する

```
# metaset -s blue -a -h lexicon
# metaset
Set name = blue, Set number = 1
```

```
Host                Owner
  lexicon
```

この例では、ホスト `lexicon` から共有ディスクセット `blue` を作成します。`metaset` コマンドは状態を表示します。この時点ではディスクセットの所有者はいません。ディスクセットにディスクを追加するホストがデフォルトで所有者になります。

ディスクセットの拡張

▼ ディスクセットにドライブを追加するには

ディスクセットに追加されるドライブは、次の条件を満たしていなければなりません。

- ドライブがボリュームやホットスเปア集合内で使用されていたり、ドライブに状態データベースの複製が含まれてはなりません。
- ドライブがマウントされていたり、スワップされていたり、アプリケーションによって開かれてはなりません。

1. 212 ページの「ディスクセットの背景情報」を確認します。

2. 次のどちらかの方法でディスクセットにドライブを追加します。

- Solaris 管理コンソール内の「拡張ディスク」から「ディスクセット (Disk Sets)」ノードを開きます。変更するディスクセットを選択し、それを右クリックして、「プロパティ (Properties)」を選択します。「ディスク (Disks)」タブを選択し、「ディスクを追加 (Add Disk)」をクリックして、ウィザードの指示に従ってドライブを追加します。詳細は、オンラインヘルプを参照してください。
- コマンド行から次の形式の `metaset` コマンドを実行して、ディスクセットにドライブを追加します。

```
metaset [-s diskset-name ] [a] [disk-name]
```

`-s diskset-name` ドライブを追加するディスクセットの名前です。

`-a` ディスクセットにドライブを追加することを意味します。

`disk-name` ディスクセットに追加するドライブを指定します。ドライブ名の形式は、名前の最後にスライス識別子「`sx`」を含まない形式、`cxtxd x`です。ドライブ名は、ディスクセットを共有するすべてのホストで共通でなければなりません。

詳細は、`metaset` のマニュアルページ (`metaset (1M)`) を参照してください。

ドライブを最初にディスクセットに追加するホストがディスクセットの所有者になります。



注意 – データが格納されているディスクをディスクセットに追加しないでください。この処理ではディスクのパーティションが再分割され、データが破壊されることがあります。詳細は、211 ページの「例 — 2つの共有ディスクセット」を参照してください。

3. `metaset` コマンドを使ってディスクセットとドライブの状態を確認します。

```
# metaset
```

例 — ディスクセットにドライブを追加する

```
# metaset -s blue -a clt6d0
# metaset
Set name = blue, Set number = 1
```

```
Host                Owner
  lexicon           Yes

Drive              Dbase
  c1t6d0           Yes
```

この例では、ホスト名は `lexicon` で、共有ディスクセットの名前は `blue` です。この時点でディスクセット `blue` に追加されているドライブは1つだけです。

コマンド行から複数のドライブを指定すれば、同時に複数のドライブを追加することもできます。たとえば、次のように指定します。

```
# metaset -s blue -a c1t6d0 c2t6d0
```

▼ ディスクセットにホストを追加するには

Solaris ボリュームマネージャは、ディスクセット当たり最大で4つのホストをサポートします。この手順では、1つのホストがすでに接続されているディスクセットに別のホストを追加する手順について説明します。

1. 212 ページの「ディスクセットの背景情報」を確認します。
2. 次のどちらかの方法でディスクセットにホストを追加します。
 - Solaris 管理コンソール内の「拡張ディスク」から「ディスクセット (Disk Sets)」ノードを開き、変更するディスクセットを選択します。変更するディスクセットを右クリックして、「プロパティ (Properties)」を選択します。「ホスト (Hosts)」タブを選択し、「ホストを追加 (Add Host)」をクリックして、ウィザードの指示に従ってホストを追加します。詳細は、オンラインヘルプを参照してください。
 - コマンド行から次の形式の `metaset` コマンドを実行して、ディスクセットにホストを追加します。

```
metaset [-s diskset-name ] [-a] [-h hostname]
```

```
-s diskset-name   ホストを追加するディスクセットの名前です。
```

```
-a                 ディスクセットにホストを追加することを意味します。
```

```
-h hostname      ディスクセットに追加する1つまたは複数のホスト名を指定します。最初のホストを追加すると、ディスクセットが作成されます。ホスト名は、/etc/nodename ファイルに指定されている名前と同じでなければなりません。
```

詳細は、`metaset` のマニュアルページ (`metaset (1M)`) を参照してください。

3. オプションを指定せずに `metaset` コマンドを実行して、ホストがディスクセットに追加されていることを確認します。

```
# metaset
```

例 — ディスクセットに別のホストを追加する

```
# metaset -s blue -a -h idiom
# metaset -s blue
Set name = blue, Set number = 1
```

Host	Owner
lexicon	Yes
idiom	

Drive	Dbase
c1t6d0	Yes
c2t6d0	Yes

この例では、ディスクセット blue にホスト idiom を追加します。

▼ ディスクセットに Solaris ボリュームマネージャのコンポーネントを作成するには

ディスクセットを作成したら、ディスクセットに追加したドライブを使ってボリュームやホットスペア集合を作成できます。この操作には、Solaris 管理コンソール内の「拡張ディスク」かコマンド行ユーティリティを使用します。

- 次のどちらかの方法で、ボリュームなどの Solaris ボリュームマネージャのデバイスをディスクセット内に作成します。
 - Solaris 管理コンソール内の「拡張ディスク」から「ボリューム (Volumes)」、「状態データベース複製 (State Database Replicas)」、または「ホットスペアプール (Hot Spare Pools)」ノードを選択します。「アクション (Action)」、「作成 (Create)」の順に選択し、ウィザードの指示に従ってコンポーネントを作成します。詳細は、オンラインヘルプを参照してください。
 - コマンド行ユーティリティを使ってディスクセット内にコンポーネントを作成します。この構文は、コマンドのすぐ後に `-s diskset-name` を指定することを除き、ディスクセットを使用しない場合の基本構文と同じです。

例 — ディスクセットに Solaris ボリュームマネージャのボリュームを作成する

```
# metainit -s blue d11 1 1 c1t6d0s0
blue/d11: Concat/Stripe is setup
# metainit -s blue d12 1 1 c2t6d0s0
blue/d12: Concat/Stripe is setup
# metainit -s blue d10 -m d11
blue/d10: Mirror is setup
# metattach -s blue d10 d12
blue/d10: submirror blue/d12 is attached
```

```

# metastat -s blue
blue/d10: Mirror
  Submirror 0: blue/d11
    State: Okay
  Submirror 1: blue/d12
    State: Resyncing
  Resync in progress: 0 % done
  Pass: 1
  Read option: roundrobin (default)
  Write option: parallel (default)
  Size: 17674902 blocks

blue/d11: Submirror of blue/d10
  State: Okay
  Size: 17674902 blocks
  Stripe 0:
    Device                Start Block  Dbase State          Reloc  Hot Spare
    c1t6d0s0                0           No   Okay

blue/d12: Submirror of blue/d10
  State: Resyncing
  Size: 17674902 blocks
  Stripe 0:
    Device                Start Block  Dbase State          Reloc  Hot Spare
    c2t6d0s0                0           No   Okay

```

この例では、ディスクセット blue にミラー d10 を作成します。ミラーは、サブミラー (RAID 0 デバイス) d11 と d12 からなります。

ディスクセットの保守

▼ ディスクセットの状態をチェックするには

- 次のどちらかの方法でディスクセットの状態をチェックします。
 - Solaris 管理コンソール内の「拡張ディスク」から「ディスクセット (Disk Sets)」ノードを開きます。監視するディスクセットを右クリックし、メニューから「プロパティ (Properties)」を選択します。詳細は、オンラインヘルプを参照してください。
 - `metaset` コマンドを使ってディスクセットの状態を表示します。詳細は、`metaset (1M)` のマニュアルページを参照してください。

注—ディスクセットの所有権は、ディスクセットを所有するホストだけに表示されます。

例 — ディスクセットの状態をチェックする

```
red# metaset -s blue
```

```
Set name = blue, Set number = 1
```

Host	Owner
idiom	Yes

Drive	Dbase
c1t6d0	Yes
c2t6d0	Yes

metaset コマンドに `-s` オプションとディスクセット名 `blue` を指定して、ディスクセットの状態情報を表示します。metaset コマンドを所有者側ホスト `idiom` から実行すると、`idiom` が実際にこのディスクセットの所有者であることがわかります。metaset コマンドでは、ディスクセットに属するドライブも表示されます。

オプション指定せずに metaset コマンドを実行すると、すべてのディスクセットの状態が表示されます。

▼ ディスクセットからディスクを削除するには

ディスクセットを削除するためには、まず、ディスクセットからすべてのドライブを削除する必要があります。

- Solaris 管理コンソール内の「拡張ディスク」から「ディスクセット (Disk Sets)」ノードを開きます。解放するディスクセットを右クリックし、メニューから「プロパティ (Properties)」を選択します。「ディスク (Disks)」タブをクリックし、指示に従ってドライブを削除します。
- 次の形式の metaset コマンドを実行します。

```
metaset -s diskset-name -d drivename
```

`-s diskset-name` 削除するホストを含むディスクセットの名前です。

`drive-name` ディスクセットから削除するドライブの名前です。ドライブ名の形式は、名前の最後にスライス識別子「`sx`」を含まない形式、`cxtxd x` です。

詳細は、metaset (1M) のマニュアルページを参照してください。

- `metaset -s diskset-name` コマンドを使って、ディスクがディスクセットから削除されていることを確認します。

```
# metaset -s blue
```

例 — ディスクセットからディスクを削除する

```
lexicon# metaset -s blue -d c1t6d0  
lexicon# metaset -s blue
```

```
Set name = blue, Set number = 1
```

Host	Owner
lexicon	
idiom	

Drive	Dbase
c2t6d0	Yes

この例では、ディスクセット blue からディスクを削除します。

▼ ディスクセットを取得するには

- 次のどちらかの方法でディスクセットを取得します。
 - Solaris 管理コンソール内の「拡張ディスク」から「ディスクセット (Disk Sets)」ノードを開きます。取得するディスクセットを右クリックし、メニューから「所有権を取得 (Take Ownership)」を選択します。詳細は、オンラインヘルプを参照してください。
 - 次の形式の `metaset` コマンドを実行します。

```
metaset -s diskset-name -t
```

`-s diskset-name` 取得するディスクセットの名前です。

`-t` ディスクセットを取得することを意味します。

`-f` ディスクセットを強制的に取得することを意味します。

詳細は、`metaset (1M)` のマニュアルページを参照してください。

ディスクセットの1つのホストがそのディスクセットを取得したら、他のホストからこのディスクセットのドライブのデータにアクセスすることはできません。

`metaset` コマンドのデフォルト動作では、他方のホストがディスクセットを解放できなければ、このホストからディスクセットを取得することはできません。

ディスクセットを強制的に取得する場合は、`-f` オプションを指定します。このオプションを指定すると、ディスクセットは、別のホストが所有しているか否かにかかわらず、このホストによって取得されます。この方法は、通常、ディスクセットを共有

するホストの1つが停止したり、通信していないときに使用されます。ディスクセットを強制的に取得させられた他のホストは、このディスクセットの入出力操作を試みたときにパニック状態になります。

注 - ディスクセットの所有権は、ディスクセットを所有するホストだけに表示されません。

例 — ディスクセットを取得する

```
lexicon# metaset
...
Set name = blue, Set number = 1

Host          Owner
  lexicon
  idiom
...
lexicon# metaset -s blue -t
lexicon# metaset
...
Set name = blue, Set number = 1

Host          Owner
  lexicon          Yes
  idiom
```

この例では、ホスト lexicon がホスト idiom と通信し、idiom がディスクセットを解放したことを確認してから、ディスクセットを取得します。

注 - この例の場合、ホスト idiom がディスクセット blue を所有していたとしても、出力の「所有者 (Owner)」欄はブランクになります。metaset コマンドは、このコマンドを実行したホスト (他のホストとは関係なく) がディスクセットを所有しているかどうかだけを表示します。

例 — ディスクセットを強制的に取得する

```
# metaset -s blue -t -f
```

この例では、ディスクセットを取得しようとするホストが他のホストと通信しません。したがって、このホストは、ディスクセットのドライブを警告なしに取得します。他のホストがディスクセットを所有していた場合には、そのホストが、ディスクセットの入出力操作を試みたときにパニックを引き起こします。

▼ ディスクセットを解放するには

ディスクセットの物理ドライブを保守する場合は、ディスクセットをあらかじめ解放しておくとう便利です。ディスクセットを解放すると、ホストからそのディスクセットにはアクセスできなくなります。ディスクセットを共有する両方のホストがディスクセットを解放すると、そのディスクセットに設定されているボリュームやホットスペア集合にはどちらのホストからもアクセスできなくなります。ただし、`c*t*d*` という名前を使えば、どちらのホストからでもディスクに直接アクセスできます。

1. 212 ページの「ディスクセットの背景情報」を確認します。
2. 次のどちらかの方法でディスクセットを解放します。
 - Solaris 管理コンソール内の「拡張ディスク」から「ディスクセット (Disk Sets)」ノードを開きます。解放するディスクセットを右クリックし、メニューから「所有権を開放 (Release Ownership)」を選択します。詳細は、オンラインヘルプを参照してください。
 - 次の形式の `metaset` コマンドを実行して、ディスクセットの所有権を解放します。

```
metaset -s diskset-name -r
```

`-s diskset-name` 解放するディスクセットの名前です。

`-r` ディスクセットを解放することを意味します。ディスクセット内のすべてのディスクの予約が取り消されます。ディスクセット内のボリュームにはアクセスできなくなります。

詳細は、`metaset (1M)` のマニュアルページを参照してください。

注 – ディスクセットの所有権は、ディスクセットを所有するホストだけに表示されます。

3. オプションを指定せずに `metaset` コマンドを実行して、ディスクセットがこのホストから解放されていることを確認します。

```
# metaset
```

例 — ディスクセットを解放する

```
lexicon# metaset -s blue -r
lexicon# metaset -s blue

Set name = blue, Set number = 1

Host          Owner
lexicon
```

```

        idiom

Drive          Dbase
c1t6d0         Yes
c2t6d0         Yes

```

この例では、ディスクセット blue を解放します。この時点では、ディスクセットの所有者がいないことに注意してください。ホスト lexicon から状態を調べると、誤解を招くおそれがあります。これは、ホストが調査できるのは、そのホストがディスクセットを所有しているかどうかだけだからです。たとえば、ホスト idiom がディスクセットを取得したとしても、ホスト lexicon からそのことを知ることはできません。この場合、取得していることを知ることができるのは、ホスト idiom だけです。

▼ ホストまたはディスクセットを削除するには

ディスクセットを削除する場合は、ディスクセットにドライブが含まれていたり、ディスクセットに他のホストが接続してはなりません。最後のホストを削除すると、ディスクセットは削除されます。

1. 次のどちらかの方法でディスクセットからホストを削除し、そのホストが最後のホストである場合は、ディスクセットも削除します。
 - Solaris 管理コンソール内の「拡張ディスク」から「ディスクセット (Disk Sets)」ノードを開きます。解放するディスクセットを右クリックし、メニューから「削除 (Delete)」を選択します。詳細は、オンラインヘルプを参照してください。
 - 次の形式の `metaset` コマンドを実行してディスクセットからホストを削除し、そのホストが最後のホストである場合は、ディスクセットも削除します。

```

metaset -s diskset-name-d hostname

-s diskset-name   削除するホストを含むディスクセットの名前です。
-d               ディスクセットからホストを削除することを意味します。
hostname         削除するホストの名前を指定です。

# metaset -s blue -d idiom
詳細は、metaset (1M) のマニュアルページを参照してください。

```

2. `metaset` コマンドを実行して、このホストがディスクセットから削除されていることを確認します。出力には、現在の (所有者側) ホストだけが表示され、他のホストがすでに削除されていることがわかります。

```

# metaset -s blue
Set name = blue, Set number = 1

Host          Owner
lexicon       Yes

```

Drive	Dbase
c1t2d0	Yes
c1t3d0	Yes
c1t4d0	Yes
c1t5d0	Yes
c1t6d0	Yes
c2t1d0	Yes

例 — ディスクセットから最後のホストを削除する

```
lexicon# metaset -s blue -d lexicon
lexicon# metaset -s blue
```

```
metaset: lexicon: setname "blue": no such set
```

この例では、ディスクセット blue から最後のホストを削除します。

第 21 章

Solaris ボリュームマネージャの保守 (作業)

この章では、Solaris ボリュームマネージャの一般的な記憶域管理に関連する保守作業について説明します。

この章の内容は次のとおりです。

- 231 ページの「Solaris ボリュームマネージャの保守 (作業マップ)」
- 232 ページの「Solaris ボリュームマネージャ構成の表示」
- 235 ページの「ボリューム名の変更」
- 238 ページの「構成ファイルの使用」
- 240 ページの「Solaris ボリュームマネージャのデフォルト値の変更」
- 242 ページの「ファイルシステムの拡張」
- 244 ページの「RAID 1 および RAID 5 ボリューム内のコンポーネントの交換と有効化の概要」

Solaris ボリュームマネージャの保守 (作業マップ)

次の表に、Solaris ボリュームマネージャの保守に必要な作業を示します。

作業	説明	参照先
Solaris ボリュームマネージャ構成の表示	Solaris ボリュームマネージャの GUI か <code>metastat</code> コマンドを使ってシステム構成を表示します。	232 ページの「Solaris ボリュームマネージャのボリューム構成を表示するには」

作業	説明	参照先
ボリューム名の変更	Solaris ボリュームマネージャの GUI か <code>metarename</code> コマンドを使ってボリューム名を変更します。	237 ページの「ボリューム名を変更するには」
構成ファイルの作成	<code>metastat -p</code> コマンドと <code>metadb</code> コマンドを使って構成ファイルを作成します。	238 ページの「構成ファイルを作成するには」
構成ファイルを使用した Solaris ボリュームマネージャの初期化	<code>metainit</code> コマンドを使って構成ファイルから Solaris ボリュームマネージャを初期化します。	238 ページの「構成ファイルを使って Solaris ボリュームマネージャを初期化するには」
使用できるボリューム数の変更	<code>/kernel/drv/md.conf</code> ファイルを編集して、使用できるボリューム数を増やします。	240 ページの「デフォルトのボリューム数を増やすには」
使用できるディスクセット数の変更	<code>/kernel/drv/md.conf</code> ファイルを編集して、使用できるディスクセット数を増やします。	241 ページの「デフォルトのディスクセット数を増やすには」
ファイルシステムの拡張	<code>growfs</code> コマンドを使ってファイルシステムを拡張します。	243 ページの「ファイルシステムを拡張するには」
コンポーネントの有効化	Solaris ボリュームマネージャの GUI か <code>metareplace</code> コマンドを使ってコンポーネントを有効にします。	245 ページの「コンポーネントの有効化」
コンポーネントの交換（置き換え）	Solaris ボリュームマネージャの GUI か <code>metareplace</code> コマンドを使ってコンポーネントを交換します。（置き換える）	245 ページの「コンポーネントを他の使用可能なコンポーネントで置き換える」

Solaris ボリュームマネージャ構成の表示

▼ Solaris ボリュームマネージャのボリューム構成を表示するには

- 次のどちらかの方法でボリューム構成を表示します。
 - Solaris 管理コンソール内の「拡張ディスク」から「ボリューム (Volumes)」ノードを開きます。詳細は、オンラインヘルプを参照してください。
 - 次の形式の `metastat` コマンドを実行します。


```
metastat -p -i component-name
```

- `-p` は、`md.tab` ファイルの作成に適した要約を出力することを意味します。
- `-i` は、すべてのデバイスがアクセス可能かどうかを検証することを意味します。
- `component-name` は表示するボリュームの名前です。ボリューム名を指定しないと、すべてのコンポーネントが表示されます。

ヒント – `metastat` の出力はソートされていません。構成リストを編集したい場合は、`metastat -p` コマンドの出力を入力にして `sort` または `grep` コマンドを実行してください。

詳細は、`metastat (1M)` のマニュアルページを参照してください。

例 — Solaris ボリュームマネージャのボリューム構成を表示する

`metastat` コマンドの出力例を以下に示します。

```
# metastat
d50: RAID
  State: Okay
  Interlace: 32 blocks
  Size: 20985804 blocks
Original device:
  Size: 20987680 blocks
  Device          Start Block  Dbase State      Reloc  Hot Spare
  c1t4d0s5        330         No  Okay       Yes
  c1t5d0s5        330         No  Okay       Yes
  c2t4d0s5        330         No  Okay       Yes
  c2t5d0s5        330         No  Okay       Yes
  c1t1d0s5        330         No  Okay       Yes
  c2t1d0s5        330         No  Okay       Yes

d1: Concat/Stripe
  Size: 4197879 blocks
  Stripe 0:
    Device          Start Block  Dbase Reloc
    c1t2d0s3        0           No    Yes

d2: Concat/Stripe
  Size: 4197879 blocks
  Stripe 0:
    Device          Start Block  Dbase Reloc
    c2t2d0s3        0           No    Yes

d80: Soft Partition
```

```

Device: d70
State: Okay
Size: 2097152 blocks
  Extent      Start Block      Block count
    0          1          2097152

```

```

d81: Soft Partition
Device: d70
State: Okay
Size: 2097152 blocks
  Extent      Start Block      Block count
    0          2097154      2097152

```

```

d70: Mirror
Submirror 0: d71
  State: Okay
Submirror 1: d72
  State: Okay
Pass: 1
Read option: roundrobin (default)
Write option: parallel (default)
Size: 12593637 blocks

```

```

d71: Submirror of d70
State: Okay
Size: 12593637 blocks
Stripe 0:
  Device      Start Block  Dbase State      Reloc Hot Spare
  c1t3d0s3    0           No  Okay      Yes
Stripe 1:
  Device      Start Block  Dbase State      Reloc Hot Spare
  c1t3d0s4    0           No  Okay      Yes
Stripe 2:
  Device      Start Block  Dbase State      Reloc Hot Spare
  c1t3d0s5    0           No  Okay      Yes

```

```

d72: Submirror of d70
State: Okay
Size: 12593637 blocks
Stripe 0:
  Device      Start Block  Dbase State      Reloc Hot Spare
  c2t3d0s3    0           No  Okay      Yes
Stripe 1:
  Device      Start Block  Dbase State      Reloc Hot Spare
  c2t3d0s4    0           No  Okay      Yes
Stripe 2:
  Device      Start Block  Dbase State      Reloc Hot Spare
  c2t3d0s5    0           No  Okay      Yes

```

```

hsp010: is empty

```

```

hsp014: 2 hot spares
Device      Status      Length      Reloc
c1t2d0s1    Available   617652 blocks Yes

```

```

c2t2d0s1          Available      617652 blocks  Yes

hsp050: 2 hot spares
Device            Status      Length      Reloc
c1t2d0s5          Available   4197879 blocks Yes
c2t2d0s5          Available   4197879 blocks Yes

hsp070: 2 hot spares
Device            Status      Length      Reloc
c1t2d0s4          Available   4197879 blocks Yes
c2t2d0s4          Available   4197879 blocks Yes

```

Device Relocation Information:

Device	Reloc	Device ID
c1t2d0	Yes	id1,sd@SSEAGATE_ST39204LCSUN9.0G3BV0N1S200002103AF29
c2t2d0	Yes	id1,sd@SSEAGATE_ST39204LCSUN9.0G3BV0P64Z00002105Q6J7
c1t1d0	Yes	id1,sd@SSEAGATE_ST39204LCSUN9.0G3BV0N1EM00002104NP2J
c2t1d0	Yes	id1,sd@SSEAGATE_ST39204LCSUN9.0G3BV0N93J000071040L3S
c0t0d0	Yes	id1,dad@s53554e575f4154415f5f53543339313430412525415933

ボリューム名の変更

ボリューム名を変更するための背景情報

metarename コマンドに `-x` オプションを指定することによって、親子関係のあるボリュームの名前を交換できます。詳細は、237 ページの「ボリューム名を変更するには」と、`metarename(1M)` のマニュアルページを参照してください。

Solaris ボリュームマネージャでは、多少の制約はありますが、ほとんどのタイプのボリュームの名前をいつでも変更できます。

ボリューム名の変更や交換は、ボリューム名を管理する上で便利な機能です。たとえば、ファイルシステムのすべてのマウントポイントがある範囲の数値内に収めることができます。あるいは、論理ボリュームの命名規則に従ってボリューム名を変更したり、トランザクションボリュームの名前にそのボリュームを構成しているボリュームが使用している名前と同じものを使用することができます。

ボリューム名を変更するときは、ボリュームが使用中でないことを確認してください。ファイルシステムとして使用されている場合は、マウントされていたり、`swap` として使用されていないことを確認してください。データベースなど、`raw` デバイスを使用するその他のアプリケーションは、独自の方法でデータへのアクセスを停止できなければなりません。

ボリューム名の変更に伴う特別な考慮事項は、次のとおりです。

- 次のものを除く任意のボリュームの名前を変更できます。
 - ソフトパーティション
 - ソフトパーティションが直接作成されているボリューム
 - ログデバイスとして使用されているボリューム
 - ホットスペア集合
- ディスクセット内のボリュームの名前を変更することはできますが、それによって、そのボリュームを別のディスクセットに移動することはできません。

ボリューム名の変更は、Solaris 管理コンソール内の「拡張ディスク」かコマンド行 (`metarename (1M)` コマンド) から行うことができます。

ボリューム名の交換

`metarename` コマンドの `-x` オプションを使用すれば、ボリュームの名前とそのボリュームを構成するデバイスの名前を交換できます。たとえば、この交換は、ミラーとそのサブミラーの間や、トランザクションボリュームとそのマスターデバイスの間で行うことができます。

注 - ボリューム名の交換には、コマンド行を使用する必要があります。この機能は、現在のところ Solaris ボリュームマネージャの GUI にはありません。ただし、ボリューム名の変更はコマンド行からでも GUI からでも行えます。

`metarename -x` コマンドを使用すれば、既存ボリュームを簡単にミラー化したりミラー解除したりできます。あるいは、既存ボリュームからトランザクションボリュームを簡単に作成したり削除したりできます。

- 使用されているボリュームの名前を変更することはできません。このタイプのボリュームの例としては、マウントされているファイルシステム、`swap` として使用されているボリューム、アプリケーションやデータベースのアクティブ記憶領域として使用されているボリュームなどが挙げられます。したがって、`metarename` コマンドを使用する前に、名前を変更するボリュームへのすべてのアクセスを停止する必要があります。たとえば、マウントされているファイルシステムのマウントを解除します。
- 障害のあるボリュームや、ホットスペアが使用されているボリュームを交換することはできません。
- 交換するボリュームは、親子関係のあるものでなければなりません。たとえば、マスターデバイスであるミラー内のストライプと、トランザクションボリュームを直接交換することはできません。
- トランザクションデバイスのメンバー間で交換を行う場合は、`-f` (強制) フラグを使用する必要があります。
- ログインデバイスの交換 (または名前の変更) を行うことはできません。交換が必要な場合は、ログインデバイスを切断し、名前を変更してから、トランザクションデータベースに再び接続します。あるいは、ログインデバイスを切断してから、適

切な名前をもつ別のロギングデバイスを接続します。

- 交換できるのはボリュームだけです。スライスやホットスペアは交換できません。

▼ ボリューム名を変更するには

1. ボリューム名の要件 (45 ページの「ボリューム名」と 235 ページの「ボリューム名を変更するための背景情報」)を確認します。
2. このボリュームを使用するファイルシステムのマウントを解除します。
3. 次のどちらかの方法でボリューム名を変更します。

- Solaris 管理コンソール内の「拡張ディスク」から「ボリューム (Volumes)」ノードを開き、名前を変更するボリュームを選択します。そのアイコンを右クリックし、「プロパティ (Properties)」オプションを選択して、画面の指示に従ってボリューム名を変更します。詳細は、オンラインヘルプを参照してください。
- 次の形式の `metarename` コマンドを実行します。

```
metarename old-volume-name new-volume-name
```

- `old-volume-name` は既存のボリュームの名前です。
- `new-volume-name` は既存のボリュームに指定する新しい名前です。

詳細は、`metarename (1M)` のマニュアルページを参照してください。

4. 必要であれば、エントリが新しいボリューム名を参照するように `/etc/vfstab` ファイルを編集します。
5. ファイルシステムを再びマウントします。

例 — ファイルシステムとして使用されているボリュームの名前を変更する

```
# umount /home
# metarename d10 d100
d10: has been renamed to d100
(ファイルシステムが新しいボリュームを参照するように /etc/vfstab ファイルを編集する)
# mount /home
```

この例では、ボリュームの名前 `d10` を `d100` に変更します。`d10` にはマウントされているファイルシステムが格納されているため、ボリューム名を変更するためには、このファイルシステムのマウントを解除する必要があります。このファイルシステムのエントリが `/etc/vfstab` ファイル内に存在している場合は、そのエントリが新しいボリューム名を参照するように変更します。

```
/dev/md/dsk/d10 /dev/md/rdisk/d10 /docs ufs 2 yes -
```

上記の行を次のように変更します。

```
/dev/md/dsk/d100 /dev/md/rdisk/d100 /docs ufs 2 yes -
```

次に、ファイルシステムを再びマウントします。

注 - 既存のミラーやトランザクションボリュームが存在する場合、`metarename -x` コマンドでミラーやトランザクションボリュームを削除しても、そのボリュームを構成するボリュームのデータは保持できます。たとえば、トランザクションボリュームの場合には、マスターデバイスがボリューム (RAID 0、RAID 1、または RAID 5 ボリューム) である限り、そのボリュームのデータは保持されます。

構成ファイルの使用

Solaris ボリュームマネージャの構成ファイルには、Solaris ボリュームマネージャの基本情報の他に、構成を再設定するのに必要なほとんどのデータが含まれています。次の各項では、構成ファイルに関連する作業について説明します。

▼ 構成ファイルを作成するには

- **Solaris** ボリュームマネージャ環境用のすべてのパラメータを適切に設定したら、`metastat -p` コマンドで `/etc/lvm/md.tab` ファイルを作成します。

```
# metastat -p> /etc/lvm/md.tab
```

このファイルには、`metainit` や `metabs` コマンドで使用するすべてのパラメータが記述されています。このファイルは、類似した複数の環境を設定したり、障害発生時に構成を再作成したりするときに使用されます。

`md.tab` ファイルについては、297 ページの「`md.tab` ファイルの概要」を参照してください。

▼ 構成ファイルを使って Solaris ボリュームマネージャを初期化するには



注意 - この手順は、Solaris ボリュームマネージャ構成が完全に失われたときや、保存されている構成ファイルから構成を新たに作成する場合にだけ使用してください。

状態データベースに保持されていた情報が失われた場合 (たとえば、すべての状態データベースの複製が削除された後にシステムを再起動した) でも、`md.cf` または `md.tab` ファイルを使って Solaris ボリュームマネージャ構成を復元できます。ただし、状態データベースが失われた後にボリュームがまったく作成されていない場合に限ります。

注 - `md.cf` ファイルには、アクティブなホットスペアの情報は格納されません。そのため、Solaris ボリュームマネージャ構成が失われたときにホットスペアが使用されていると、アクティブなホットスペアを使用していたボリュームの内容は破壊されることがあります。

これらのファイルについては、`md.cf` (4) のマニュアルページと `md.tab` (4) のマニュアルページを参照してください。

1. 状態データベースの複製を作成します。
詳細は、64 ページの「状態データベースの複製の作成」を参照してください。
2. `/etc/lvm/md.tab` ファイルを作成または更新します。
 - Solaris ボリュームマネージャ構成最後の状態を回復する場合は、`md.cf` ファイルを `md.tab` ファイルにコピーします。
 - 保存されている `md.tab` ファイルのコピーを使って新しい Solaris ボリュームマネージャ構成を作成する場合は、このコピーを `/etc/lvm/md.tab` に置きます。
3. 「新しい」`md.tab` ファイルを編集してから次の作業を行います。
 - 新しい構成を作成している場合や、クラッシュの後で構成を回復している場合には、ミラーを 1 面ミラーとして構成します。ミラーの各サブミラーのサイズが同じでない場合は、もっとも小さいサブミラーをこの 1 面ミラーのために使用する必要があります。そうしないと、データを失うおそれがあります。
 - 既存の構成を回復している場合、Solaris ボリュームマネージャが正常に終了しているのであれば、ミラー構成を多面ミラーのままにして回復します。
 - RAID 5 ボリュームの場合は、デバイスの再初期化を防止するために `-k` オプションを指定します。詳細は、`metainit(1M)` のマニュアルページを参照してください。
4. 次の形式の `metainit` コマンドを使って `md.tab` ファイルのエントリの構文だけをチェックします (実際の処理は行われません)。

```
# metainit -n -a component-name
```

 - `-n` は、デバイスを実際には作成しないことを意味します。このオプションでは、期待どおりの結果が得られるかどうかだけを確認します。
 - `-a` は、デバイスをアクティブにすることを意味します。
 - `component-name` は、初期化するコンポーネントの名前です。コンポーネントを指定しないと、すべてのコンポーネントが作成されます。

5. 前の手順で特に問題がなければ、`md.tab` ファイルを使ってボリュームとホットスペア集合を作成し直します。

```
# metainit -a component-name
```

- `-a` は、デバイスをアクティブにすることを意味します。
- `component-name` は、初期化するコンポーネントの名前です。コンポーネントを指定しないと、すべてのコンポーネントが作成されます。

6. 必要であれば、`metattach` コマンドを使って 1 面ミラーを多面ミラーにします。
7. ボリューム上のデータが正しいこと確認します。

Solaris ボリュームマネージャのデフォルト値の変更

Solaris ボリュームマネージャ構成では、次のデフォルト値が使用されています。

- ディスクセットあたり 128 個の ボリューム
- 4 つのディスクセット
- 状態データベースの複製

ボリュームの合計数とディスクセットの数は、必要に応じて変更できます。この節では、そのための手順について説明します。

▼ デフォルトのボリューム数を増やすには

この手順では、デフォルトのボリューム数 128 を増やします。指定できる最大数は 8192 です。



注意 – この数を減らした場合、前の数と新しい数の間にあるボリュームは使用不能になり、データを失うおそれがあります。「`md: d200: not configurable, check /kernel/drv/md.conf`」というメッセージが表示された場合は、`md.conf` ファイルを編集し、この手順に従ってボリュームの数を増やす必要があります。

1. 前提条件 (270 ページの「障害追跡の前提条件」) を確認します。
2. `/kernel/drv/md.conf` ファイルを編集します。
3. `nmd` フィールドの値を変更します。設定できる最大数は **8192** です。
4. 変更を保存します。

5. 再構成のために再起動を行なってボリューム名を作成します。

```
# boot -r
```

例 —md.conf ファイル

ボリューム数として 256 が設定されている md.conf ファイルの例を以下に示します。

```
#
#ident "@(#)md.conf 1.7 94/04/04 SMI"
#
# Copyright (c) 1992, 1993, 1994 by Sun Microsystems, Inc.
#
#
#pragma ident "@(#)md.conf 2.1 00/07/07 SMI"
#
# Copyright (c) 1992-1999 by Sun Microsystems, Inc.
# All rights reserved.
#
name="md" parent="pseudo" nmd=256 md_nsets=4;
```

デフォルトのディスクセット数を増やすには

この手順では、デフォルトのディスクセット数 4 を増やします。



注意 – ディスクセットがすでに設定されている場合は、デフォルトのディスクセット数を減らさないでください。この数を減らすと、既存のディスクセットが使用不能になるおそれがあります。

1. 前提条件 (270 ページの「障害追跡の前提条件」) を確認します。
2. /kernel/drv/md.conf ファイルを編集します。
3. md_nsets フィールドの値を変更します。設定できる最大数は 32 です。
4. 変更を保存します。
5. 再構成のために再起動を行なってボリューム名を作成します。

```
# boot -r
```

例 —md.conf ファイル

5 つの共有ディスクセットを使用できるように設定されている md.conf ファイルの例を以下に示します。md_nsets の値が 6 であるため、5 つの共有ディスクセットと 1 つのローカルディスクセットを使用できます。

```
#
#
#pragma ident    "@(#)md.conf      2.1      00/07/07 SMI"
#
# Copyright (c) 1992-1999 by Sun Microsystems, Inc.
# All rights reserved.
#
name="md" parent="pseudo" nmd=128 md_nsets=6;
# Begin MDD database info (do not edit)
...
# End MDD database info (do not edit)
```

ファイルシステムの拡張

ファイルシステムが格納されているボリュームを拡張(領域を追加)したときに、そのボリュームに UFS が格納されている場合は、ファイルシステムを拡張して新しい領域が認識されるようにする必要があります。ファイルシステムの拡張は、`growfs` コマンドを使って手動で行う必要があります。`growfs` コマンドは、マウントされているファイルシステムも拡張できます。ただし、`growfs` コマンドの実行中は、ファイルシステムに書き込みアクセスを行うことはできません。

データベースなど、`raw` デバイスを使用するアプリケーションは、独自の方法で領域を拡張できなければなりません。`Solaris` ボリュームマネージャには、この機能はありません。

`growfs` コマンドは、ファイルシステムを拡張する間、マウントされているファイルシステムを「書き込みロック」します。ファイルシステムが書き込みロックされている時間を短縮する必要がある場合は、ファイルシステムを段階的に拡張できます。たとえば、1G バイトのファイルシステムを 2G バイトに拡張する場合は、`-s` オプションを使って新しいファイルシステムの合計サイズを拡張処理の各段階で指定することによって、ファイルシステムを 16M バイト単位で拡張できます。

拡張処理の間はファイルシステムが書き込みロックされるため、このファイルシステムへの書き込みは実行できません。書き込みアクセスは `growfs` コマンドがファイルシステムのロックを解除するまで自動的に中断され、ロックが解除されると再開されます。読み取りアクセスには影響はありませんが、ロックが行われている間の読み取りアクセス時間は保証されません。

スライスやボリュームを拡張するための背景情報

注 - Solaris ボリュームマネージャのボリュームは拡張することはできますが、縮小することはできません。

- ボリュームは、ファイルシステム、アプリケーション、データベースのいずれで使用されていても拡張できます。RAID 0 (ストライプ方式と連結方式) ボリューム RAID 1 (ミラー) ボリューム、RAID 5 ボリュームだけでなく、ソフトパーティションも拡張できます。
- ファイルシステムが格納されているボリュームを連結できます。ファイルシステムが UFS であれば、`growfs` コマンドを使って、このファイルシステムを、拡張された領域まで拡張できます。このとき、データへの読み取りアクセスは中断されません。
- いったん拡張されたファイルシステムは、UFS の制約により、縮小することはできません。
- `raw` デバイスを使用するアプリケーションやデータベースは、独自の方法で領域を増やし、それを認識できなければなりません。Solaris ボリュームマネージャには、この機能はありません。
- RAID 5 ボリュームにコンポーネントを追加すると、コンポーネントはデバイスへの連結になります。新しいボリュームにはパリティ情報は格納されませんが、このコンポーネントのデータは、ボリュームに対して行われる全体的なパリティ計算によって保護されます。
- コンポーネントを追加することによってログデバイスを拡張できます。再起動時に Solaris ボリュームマネージャは新しい領域を自動的に認識するため、`growfs` コマンドを実行する必要はありません。
- ソフトパーティションを拡張するには、そのパーティションを構成するボリュームまたはスライスから領域を追加します。その他のボリュームはすべて、スライスを追加することによって拡張できます。

▼ ファイルシステムを拡張するには

1. 49 ページの「Solaris ボリュームマネージャ要素を作成するための前提条件」を確認します。
2. `growfs` コマンドを使ってローカルボリュームの UFS を拡張します。

```
# growfs -M /mount-point /dev/md/rdisk/volumename
```

詳細は、次の例と `growfs(1M)` のマニュアルページを参照してください。

例 — ファイルシステムを拡張する

```
# df -k
Filesystem                kbytes    used    avail capacity  Mounted on
```

```

...
/dev/md/dsk/d10          69047   65426       0   100%   /home2
...
# growfs -M /home2 /dev/md/rdisk/d10
/dev/md/rdisk/d10:      295200 sectors in 240 cylinders of 15 tracks, 82 sectors
      144.1MB in 15 cyl groups (16 c/g, 9.61MB/g, 4608 i/g)
super-block backups (for fsck -F ufs -o b=#) at:
   32, 19808, 39584, 59360, 79136, 98912, 118688, 138464, 158240, 178016, 197792,
  217568, 237344, 257120, 276896,
# df -k
Filesystem              kbytes    used   avail capacity  Mounted on
...
/dev/md/dsk/d10         138703   65426   59407    53%    /home2
...

```

この例では、新しいスライスがボリューム d10 にすでに追加されています。このボリュームには、マウントされているファイルシステム /home2 があります。growfs コマンドでは、-M オプションを使ってマウントポイントに /home2 を指定し、これを raw ボリューム /dev/md/rdisk/d10 上に拡張します。growfs コマンドが終了すると、このファイルシステムはボリューム全体を占めます。拡張の前後で df -hk コマンドを使用すれば、全体のディスク容量を確認できます。

注 - ミラーやトランザクションボリュームの場合は、サブミラーやマスターデバイスに領域を追加する場合でも、必ずトップレベルのボリューム (サブミラーやマスターデバイスではなく) に対して growfs コマンドを実行する必要があります。

RAID 1 および RAID 5 ボリューム内のコンポーネントの交換と有効化の概要

Solaris ボリュームマネージャには、RAID 1 (ミラー) および RAID 5 ボリューム内のコンポーネントを交換したり、有効にしたりする機能があります。

Solaris ボリュームマネージャでは、コンポーネントの交換は、サブミラーまたは RAID 5 ボリューム内のコンポーネントをシステム上で使用可能な他のコンポーネントと交換することを意味します。このプロセスは、コンポーネントの物理的な交換ではなく論理的な交換です。詳細は、245 ページの「コンポーネントを他の使用可能なコンポーネントで置き換える」を参照してください。

コンポーネントの有効化とは、コンポーネントをアクティブにする (または、コンポーネントをそれ自身で置き換える) ことを意味します。したがって、コンポーネント名は変わりません。詳細は、245 ページの「コンポーネントの有効化」を参照してください。

注 - ディスクエラーから回復するときは、`/var/adm/messages` を調べ、発生したエラーを特定してください。エラーが一時的なものであり、ディスク自体に問題がない場合は、コンポーネントを有効にしてみます。また、`format` コマンドを使ってディスクをテストすることもできます。

コンポーネントの有効化

コンポーネントを有効にする必要があるのは次のような場合です。

- Solaris ボリュームマネージャが物理ディスクにアクセスできない場合。この問題は、たとえば、電源が切断されていたり、ドライブケーブルが外れていることが原因で発生します。Solaris ボリュームマネージャは、コンポーネントを「保守 (Maintenance)」状態に変更します。この場合には、電源を復旧したり、ケーブルを接続し直したりしてドライブを使用可能にしてから、ボリュームのコンポーネントを有効にする必要があります。
- 物理ディスクに、ディスクに関連しない一時的な障害があると考えられる場合。コンポーネントが「保守 (Maintenance)」状態であれば、そのコンポーネントを有効にするだけでコンポーネントを修復できる場合があります。これで問題が解決しない場合は、ディスクドライブを物理的に交換してからそのコンポーネントを有効にするか、システム上の他の使用可能なコンポーネントでそのコンポーネントを置き換える必要があります。

ドライブを物理的に置き換える場合は、新しいドライブを前のものと同じようにパーティション分割し、各コンポーネント上に十分な領域を確保する必要があります。

注 - 交換対象のドライブ上に状態データベースの複製やホットスペアがないか必ずチェックしてください。状態データベースの複製がエラー状態になっている場合は、ディスクを交換する前にその複製を削除し、交換後にコンポーネントを有効にしてから同じサイズで作成し直す必要があります。ホットスペアについても同じように処理してください。

コンポーネントを他の使用可能なコンポーネントで置き換える

既存のコンポーネントを他の未使用のコンポーネントで置き換える (または交換する) には、`metareplace` コマンドを使用します。

このコマンドを使用できるのは、次のような場合です。

- ディスクドライブに障害があり、代わりになるドライブはないが、システム上に使用可能なコンポーネントが他にある場合。

どうしても交換する必要があるが、システムを停止したくない場合には、この方法を使用します。

- ソフトエラーが発生した場合。

Solaris ボリュームマネージャではミラー/サブミラーや RAID 5 ボリュームが「正常 (Okay)」であっても、物理ディスクがソフトエラーを報告することがあります。問題のコンポーネントを他の使用可能なコンポーネントで置き換えると、ハードウェアエラーの発生を防止するための予防的保守を実行できます。

- 性能の調整を行いたい場合。

たとえば、Solaris 管理コンソール内の「拡張ディスク」の性能監視機能を使えば、RAID 5 ボリュームの特定のコンポーネントが「正常 (Okay)」状態であっても、平均して高い負荷を処理していることがわかります。その場合には、ボリュームの負荷を均一化するために、このコンポーネントを、これより使用率の低いディスクのコンポーネントで置き換えることができます。このような処理は、ボリュームへのサービスを中断せずにオンラインで行うことができます。

「保守 (Maintenance)」状態と「最後にエラー (Last Erred)」状態

ミラーや RAID 5 ボリュームのコンポーネントにエラーが発生すると、Solaris ボリュームマネージャはそのコンポーネントを「保守 (Maintenance)」状態にします。これ以降、「保守 (Maintenance)」状態のコンポーネントには読み書きは実行されません。同じボリューム内の他のコンポーネントで次のエラーが発生した場合、そのエラーの処理方法は、ボリュームのタイプによって異なります。RAID 1 ボリュームでは、多数のコンポーネントが「保守 (Maintenance)」状態になっても、読み取りや書き込みを継続できることがあります。RAID 5 ボリュームは、その定義からして、「保守 (Maintenance)」状態のコンポーネントが 1 つだけであれば、引き続き動作します。

RAID 0 や RAID 5 ボリュームのコンポーネントにエラーが発生し、読み取りに使用できる冗長コンポーネントが存在しない場合、たとえば、RAID 5 ボリューム内の 1 つのコンポーネントが「保守 (Maintenance)」状態になった場合、冗長コンポーネントが存在しないため、次に障害が発生したコンポーネントは「最後にエラー (Last Erred)」状態になります。ミラーや RAID 5 ボリュームの 1 つのコンポーネントが「最後にエラー (Last Erred)」状態になっても、そのコンポーネントに対してはまだ入出力が試みられます。これは、Solaris ボリュームマネージャからは「最後にエラー (Last Erred)」状態のコンポーネントに最新の正しいデータコピーが含まれているように見えるからです。「最後にエラー (Last Erred)」状態のコンポーネントを含むボリュームは正常なデバイス (ディスク) のように動作し、アプリケーションに入出力エラーを返します。通常、この時点では、データの一部がすでに失われています。

必ず「保守 (Maintenance)」状態のコンポーネントを先に交換してから、「最後にエラー (Last Erred)」状態のコンポーネントを交換します。コンポーネントを置き換え、再同期を取ったら、`metastat` コマンドを使ってコンポーネントの状態を確認し、データの妥当性を検証します。

ミラー-コンポーネントが「保守 (Maintenance)」状態である限り、データは失われていません。コンポーネントをどのような順序で置き換え、有効にしてもかまいません。コンポーネントが「最後にエラー (Last Erred)」状態の場合は、「保守 (Maintenance)」状態の他のすべてのコンポーネントを置き換えてから、このコンポーネントを置き換えます。「最後にエラー (Last Erred)」状態のコンポーネントを交換したり有効にすることは、通常、一部のデータがすでに失われていることを意味します。ミラーの修復後に、ミラー上のデータの妥当性を必ず確認します。

RAID 5 ボリューム-RAID 5 ボリュームは、1つのコンポーネントの障害に耐えることができます。「保守 (Maintenance)」状態のコンポーネントが1つである限り、そのコンポーネントを置き換えてもデータが失われることはありません。他のコンポーネントに障害が発生すると、そのコンポーネントは「最後にエラー (Last Erred)」状態になります。この時点で RAID 5 ボリュームは読み取り専用になります。必要な回復処置を行なって、RAID 5 ボリュームを安定状態にし、データ損失の可能性を減らします。RAID 5 ボリュームが「最後にエラー (Last Erred)」状態の場合には、データがすでに失われている可能性高くなります。RAID 5 ボリュームを修復したら、必ずデータを検証してください。

ミラーおよび RAID 5 ボリューム内のスライスを交換または有効にするための背景情報

ミラーまたは RAID 5 ボリューム内のコンポーネントを交換する場合には、次の指針に従ってください。

- 必ず「保守 (Maintenance)」状態のコンポーネントを先に交換してから、「最後にエラー (Last Erred)」状態のコンポーネントを交換します。
- コンポーネントを交換して再同期を取ったら、`metastat` コマンドを使ってボリュームの状態を確認し、データが正しいことを検証します。「最後にエラー (Last Erred)」状態のコンポーネントを交換したり有効にすることは、通常、一部のデータがすでに失われていることを意味します。ボリュームの修復後に、そのデータが正しいことを必ず確認してください。UFS の場合は、`fsck` コマンドを使って「メタデータ」(ファイルシステムの構造)を検証してから、実際のユーザーデータをチェックする必要があります。(実際には、ユーザーが各自のファイルを確認する必要があります。) データベースなどのアプリケーションは、独自の方法で内部のデータ構造を検証できなければなりません。
- コンポーネントを交換する場合は、状態データベースの複製やホットスペアが存在していないか必ずチェックします。エラー状態の状態データベースの複製がある場合は、物理ディスクを交換する前に削除する必要があります。そして、コンポーネントを有効にする前に、この状態データベースの複製を追加してください。ホットスペアの場合も同じ処理が必要です。
- RAID 5 ボリューム-コンポーネントの交換中、データは、現在使用中のホットスペアか RAID レベル 5 パリティ (ホットスペアが使用されていない場合) を使用して復元されます。

- RAID 1 ボリューム – コンポーネントを交換すると、新しいコンポーネントとミラーの残りのコンポーネントとの間で、再同期が自動的に開始されます。再同期が終了すると、新しいコンポーネントは読み書きが可能になります。一方、コンポーネントのデータがホットスペアから復元されると、ホットスペアは「使用可能 (Available)」の状態にされ、他のスペア交換に使用できる状態になります。
- 新しいコンポーネントのサイズは、古いコンポーネントを置き換えられるだけの大きさでなければなりません。
- 「最後にエラー (Last Erred)」状態のデバイスを交換する場合は、用心のためにすべてのデータのバックアップをとってください。

注 – サブミラーや RAID 5 ボリュームは、障害が発生したコンポーネントの代わりにホットスペアを使用していることがあります。障害が発生したコンポーネントをこの手順で有効にしたり、交換すると、そのホットスペアはホットスペア集合で「使用可能 (Available)」状態に戻り、使用可能になります。

第 22 章

Solaris ボリュームマネージャで構築可能な最善の記憶装置構成

この章では、Solaris ボリュームマネージャを使用して構築できる最善の記憶装置構成について、現実的な構成例を示して説明します。まず一般的な構成について説明し、次にその構成の分析を行い、最後に要件にもっとも合った推奨 (最善の) 構成を示します。

この章では、次の内容について説明します。

- 249 ページの「小規模なサーバーを運用する場合の構成例」
- 251 ページの「ネットワーク接続された記憶装置に対して Solaris ボリュームマネージャを使用する場合の構成例」

小規模なサーバーを運用する場合の構成例

分散コンピューティング環境においては、ISP や、地理的に分散した営業所、テレコムサービスプロバイダは、類似したまたは同じサーバーを複数の場所で運用しなければならないことがよくあります。これらのサーバーは、通常、ルーター、ファイアウォールサービス、eメールサービス、DNS キャッシュ、Usenet (ネットワークニュース) サーバー、DHCP サービスなど、さまざまな場所の要件にもっとも合った形でサービスを提供します。これらの小規模なサーバーは、次のような共通する要件を満たす必要があります。

- 高い信頼性
- 高い可用性
- 汎用性と性能に優れたハードウェア

まず最初の構成例として、1つの SCSI バスと2つの内蔵ディスクを備えた Netra システムを考えてみます。このシステムはすぐに使用できる構成となっており、分散サーバーとして基本的な機能を備えています。Solaris ボリュームマネージャを使えば、一部またはすべてのスライスをミラー化し、冗長記憶領域を構成することにより、ディスク障害に対する保護機能を簡単に強化できます。次の図に構成例を示します。

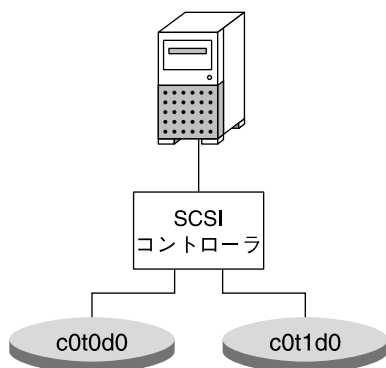


図 22-1 小規模なシステム構成

この例のような構成では、ルート (/)、/usr、swap、/var および、/export ファイルシステムに加え、状態データベースの複製 (ディスクごとに1つ) をミラー化できます。そのため、各ミラーの片側に障害が発生しても、システム障害に至るとは限りません。通常、このようなシステムは、最大5つの個別の障害に耐えることができます。しかし、このシステムは、ディスクやスライスの障害に対して十分に保護されているとはいえません。さまざまな潜在的な障害によって致命的なシステム障害が引き起こされ、オペレータの介入が必要になる場合があります。

この構成は致命的なディスク障害に対してある程度の保護機能を備えていますが、次のような重大な単一点障害が存在します。

- 1つの SCSI コントローラが単一点障害の原因となり得ます。コントローラに障害が発生すると、そのコントローラが交換されるまでシステムは停止します。
- 2つのディスクでは、状態データベースの複製の分散という観点からは十分とはいえません。多数決アルゴリズムでは、状態データベースの複製の半数が使用可能でなければ、システムは動作を続けることはできません。また、再起動のためには半数プラス1の複製が必要です。したがって、各ディスクに1つの状態データベースの複製がある場合には、1つのディスクまたは複製が置かれているスライスに障害が発生すると、システムは再起動できなくなります (したがって、ミラー化されたルートファイルシステムは無効になります)。各ディスクに2つ以上の状態データベースの複製がある場合には、1つのスライスに障害があっても問題はありませんが、ディスクに障害があると、再起動できなくなります。各ディスクに異なる数の複製がある場合には、一方のディスクに半数以上の複製が、他方のディスクには半数未満の複製が存在します。複製の数が少ない方のディスクに障害が発生しても、システムは再起動し、動作を続けられますが、複製の数が多き方のディスクに障害が発生すると、システムはただちにパニック状態になります。

結論として、このシステムについては、1つのコントローラと1つのハードドライブを追加した構成が「最善の構成」となります。このように構成を修正すると、耐障害性が大幅に向上します。

ネットワーク接続された記憶装置に対して Solaris ボリュームマネージャを使用する場合の構成例

Solaris ボリュームマネージャは、ネットワーク接続された記憶装置、特に、構成可能な RAID レベルと柔軟なオプションを備えた記憶装置に対しても使用できます。通常、Solaris ボリュームマネージャと他のデバイスを組み合わせると、一方だけの場合よりも優れた性能と柔軟性が得られます。

一般には、冗長性を備えたハードウェア記憶デバイス (RAID 1 および RAID 5 ボリューム) 上に Solaris ボリュームマネージャの RAID 5 ボリュームを設定しないようにします。極めて特殊な状況を除けば、性能に悪影響があるだけでなく、冗長性や高可用性の観点からも得るものはほとんどありません。

一方、RAID 5 ハードウェア記憶デバイスに RAID 5 以外のボリュームを作成するのは非常に効果的な方法です。これは、RAID 5 ボリュームが Solaris ボリュームマネージャにとって、適切な基本コンポーネントであるためです。ハードウェア RAID 5 を使用すると、Solaris ボリュームマネージャの RAID 1 ボリュームや、ソフトウェアパーティション、その他のボリュームでは、冗長性がより向上します。

注 - 類似したソフトウェアデバイスとハードウェアデバイスをいっしょに構成しないようにします。たとえば、ハードウェア RAID 1 デバイス上にソフトウェア RAID 1 ボリュームを作成しないでください。類似したデバイスを使用すると、性能が低下し、信頼性が向上することはありません。

RAID 1 ハードウェア記憶デバイス上に作成された Solaris ボリュームマネージャの RAID 1 ボリュームは RAID 1+0 ではありません。Solaris ボリュームマネージャは、RAID 1 ボリュームを構成する記憶デバイスを正しく認識できないため、RAID 1+0 機能を提供できません。

ハードウェア RAID 5 デバイスに Solaris ボリュームマネージャの RAID 1 ボリュームを作成し、さらにその上にソフトウェアパーティションを作成すると、柔軟で耐障害性に優れた構成になります。

監視とエラーレポート (作業)

Solaris ボリュームマネージャは、スライスレベルの物理エラーのためにボリュームに書き込みできないなどの問題を検出すると、システム管理者に通知するためにボリュームの状態を変更します。ただし、Solaris 管理コンソールを通じて Solaris ボリュームマネージャのグラフィカルユーザーインターフェースを使用したり、`metastat` コマンドを実行してボリューム状態の変化を定期的にチェックしないと、状態の変化をタイムリーに把握することはできません。

この章では、Solaris ボリュームマネージャ SNMP エージェント (Solstice Enterprise Agents™ 監視ソフトウェアのサブエージェント) など、Solaris ボリュームマネージャのさまざまな監視ツールについて説明します。Solaris ボリュームマネージャ SNMP エージェントを設定して SNMP トラップを報告する方法の他にも、シェルスクリプトを作成して Solaris ボリュームマネージャのさまざまな機能を能動的に監視することができます。このようなシェルスクリプトは cron ジョブとして動作し、潜在的な問題が顕在化する前にそれらを検出する上で役立ちます。

この章の内容は次のとおりです。

- 254 ページの「Solaris ボリュームマネージャの監視機能と報告機能 (作業マップ)」
- 254 ページの「エラーを周期的にチェックするための `mdmonitor` デーモンの設定」
- 255 ページの「Solaris ボリュームマネージャ SNMP エージェントの概要」
- 256 ページの「Solaris ボリュームマネージャ SNMP エージェントの構成」
- 258 ページの「Solaris ボリュームマネージャ SNMP エージェントの制約」
- 259 ページの「cron ジョブによる Solaris ボリュームマネージャの監視」

Solaris ボリュームマネージャの監視機能と報告機能 (作業マップ)

次の表に、Solaris ボリュームマネージャのエラーレポート機能を管理するのに必要な作業を示します。

作業	説明	参照先
mdmonitord デーモンを設定してエラーを定期的にチェックする	/etc/rc2.d/S95svm.sync ファイルを編集して、mdmonitord デーモンが使用するエラーチェックの間隔を設定します。	254 ページの「エラーを定期的にチェックするための mdmonitord デーモンの設定」
Solaris ボリュームマネージャ SNMP エージェントを構成する	/etc/snmp/conf ディレクトリの構成ファイルを編集して、Solaris ボリュームマネージャが適切なシステムにトラップを送信できるようにします。	256 ページの「Solaris ボリュームマネージャ SNMP エージェントの構成」
cron コマンドでスクリプトを実行して Solaris ボリュームマネージャを監視する	エラーをチェックするスクリプトを作成または変更し、cron コマンドでスクリプトを実行します。	259 ページの「cron ジョブによる Solaris ボリュームマネージャの監視」

エラーを定期的にチェックするための mdmonitord デーモンの設定

Solaris ボリュームマネージャには、Solaris ボリュームマネージャボリュームのエラーをチェックする /usr/sbin/mdmonitord デーモンが含まれています。デフォルトでは、このプログラムは、書き込みエラーなどにより、いずれかのボリュームでエラーが検出された場合にのみ、すべてのボリュームのエラーをチェックします。ただし、指定の間隔でエラーを能動的にチェックするようにこのプログラムを設定することもできます。

▼ mdmonitord コマンドを設定してエラーを周期的にチェックするには

/etc/rc2.d/S95svm.sync スクリプトは、起動時に mdmonitord コマンドを実行します。/etc/rc2.d/S95svm.sync スクリプトを編集して、チェックする間隔を追加します。

1. スーパーユーザーになります。
2. /etc/rc2.d/S95svm.sync スクリプトを編集し、mdmonitord コマンドで始まる行に `-t` フラグとチェックする間隔 (秒数) を指定します。

```
if [ -x $MDMONITORD ]; then
    $MDMONITORD -t 3600
    error=$?
    case $error in
    0)      ;;
    *)      echo "Could not start $MDMONITORD. Error $error."
            ;;
    esac
fi
```

3. mdmonitord コマンドを停止し、再起動して変更を有効にします。

```
# /etc/rc2.d/S95svm.sync stop
# /etc/rc2.d/S95svm.sync start
```

詳細は、mdmonitord(1M) のマニュアルページを参照してください。

Solaris ボリュームマネージャ SNMP エージェントの概要

Solaris ボリュームマネージャ SNMP トラップエージェントには、コアパッケージ SUNWlvmr と SUNWlvma、それに Solstice Enterprise Agent パッケージが必要です。必要なパッケージは、次のとおりです。

- SUNWmibii
- SUNWsacom
- SUNWsadmi
- SUNWsasnm
- SUNWsasnx

これらのパッケージは Solaris オペレーティング環境の一部なので、パッケージの選択をインストール時に変更したり、最小限のパッケージ群だけをインストールしたりしなければ、通常はデフォルトでインストールされます。pkginfo *pkgname* コマンド (たとえば、pkginfo SUNWsasnx) を使って5つのパッケージが存在することを確認したら、次の各項の説明に従って Solaris ボリュームマネージャ SNMP エージェントを構成します。

Solaris ボリュームマネージャ SNMP エージェントの構成

Solaris ボリュームマネージャ SNMP エージェントはデフォルトでは有効にされていません。SNMP トラップを有効にするには、次の手順を実行します。

▼ Solaris ボリュームマネージャ SNMP エージェントを構成するには

1. スーパーユーザーになります。
2. /etc/snmp/conf/mdlogd.rsrc- 構成ファイルを /etc/snmp/conf/mdlogd.rsrc に移動します。

```
# mv /etc/snmp/conf/mdlogd.rsrc- /etc/snmp/conf/mdlogd.rsrc
```
3. /etc/snmp/conf/mdlogd.acl ファイルを編集して、**SNMP** トラップをどのホストに送信するかを指定します。次の部分を探してください。

```
trap = {
  {
    trap-community = SNMP-trap
    hosts = corsair
    {
      enterprise = "Solaris Volume Manager"
      trap-num = 1, 2, 3
    }
  }
}
```

hosts = corsair の行に、Solaris ボリュームマネージャ SNMP トラップの送信先ホストを指定します。たとえば、SNMP トラップを lexicon に送信する場合は、この行を hosts = lexicon に変更します。複数のホストを指定する場合は、hosts = lexicon, idiom のように、ホスト名をコンマで区切って指定します。

4. 次に、/etc/snmp/conf/snmpdx.acl ファイルを編集して、**SNMP** トラップをどのホストに送信するかを指定します。
trap = で始まるブロックを探し、ここに前の手順で指定したのと同じホスト名のリストを指定します。このセクションは # でコメント文にされていることがあります。その場合は、必要な行の始めにある # を取り除いてください。このセクションでは他

の行もコメント文にされている場合がありますが、それらの行はそのまま残しておいても、見やすくするために削除してもかまいません。必要な行のコメントを解除し、ホスト名の行を変更した後のセクションは次のようになります。

```
#####
# trap parameters #
#####

trap = {
  {
    trap-community = SNMP-trap
    hosts =lexicon
    {
      enterprise = "sun"
      trap-num = 0, 1, 2-5, 6-16
    }
#    {
#      enterprise = "3Com"
#      trap-num = 4
#    }
#    {
#      enterprise = "snmp"
#      trap-num = 0, 2, 5
#    }
#  }
# {
#   trap-community = jerry-trap
#   hosts = jerry, nanak, hubble
#   {
#     enterprise = "sun"
#     trap-num = 1, 3
#   }
#   {
#     enterprise = "snmp"
#     trap-num = 1-3
#   }
# }
}
```

注 - /etc/snmp/conf/snmpdx.ac1 ファイルに同じ数の左括弧と右括弧があることを確認してください。

5. 前の手順でコメントを解除した、/etc/snmp/conf/snmpdx.ac1 ファイルのセクションの中に新しい **Solaris** ボリュームマネージャセクションを追加します。

```
trap-community = SNMP-trap
hosts = lexicon
{
  enterprise = "sun"
  trap-num = 0, 1, 2-5, 6-16
}
{
```

```
        enterprise = "Solaris Volume Manager"
        trap-num = 1, 2, 3
    }
```

追加する 4 行は、enterprise = "sun" ブロックのすぐ後に挿入する必要があります。

6. /etc/snmp/conf/enterprises.oid ファイルの最後に次の行を追加します。

```
"Solaris Volume Manager" "1.3.6.1.4.1.42.104"
```

7. **Solstice Enterprise Agents** サーバーを停止し、再起動します。

```
# /etc/init.d/init.snmpdx stop
# /etc/init.d/init.snmpdx start
```

注 - おそらく、Solaris オペレーティング環境をアップグレードするたびに、/etc/snmp/conf/enterprises.oid ファイルを編集し、その終わりに手順 6 の行を追加してから、Solaris Enterprise Agents サーバの停止と再起動を行う必要があります。

この手順が終わると、SNMP トラップが、指定されたホストに送信されるようになります。送信されたトラップを表示するためには、Solstice Enterprise Agents ソフトウェアなど、適切な SNMP モニターを使用する必要があります。

注 - 問題が発生したときにトラップを受信するためには、mdmonitord コマンドを設定してシステムを定期的にチェックする必要があります。詳細は、254 ページの「エラーを周期的にチェックするための mdmonitord デーモンの設定」を参照してください。また、その他のエラーチェックオプションについては、259 ページの「cron ジョブによる Solaris ボリュームマネージャの監視」を参照してください。

Solaris ボリュームマネージャ SNMP エージェントの制約

Solaris ボリュームマネージャ SNMP エージェントには一定の制約があります。そのため、SNMP エージェントは、システム管理者が把握する必要がある、Solaris ボリュームマネージャのすべての問題をトラップできるわけではありません。具体的には、このエージェントは、次の場合にのみトラップを送信します。

- RAID 1 または RAID 5 のサブコンポーネントが「保守が必要」状態に移行した場合
- ホットスペアが、障害のあるディスクに代わって使用されるようになった場合

- ホットスペアが再同期処理を開始した場合
- ホットスペアが再同期処理を完了した場合
- ミラーがオフライン状態になった場合
- ディスクセットが別のホストに予約されたため、現在のホストがパニック状態になった場合

RAID 0 ボリュームやソフトパーティションが定義されているディスクが使用不能な場合など、問題が発生しても、SNMP トラップが送信されない状況は少なくありません。これは、このデバイスに対する読み取りや書き込みが行われた場合でも同様です。そのような場合、一般に SCSI や IDE のエラーは検出されますが、このようなエラーが監視コンソールに報告されるためには、他の SNMP エージェントがトラップを送信する必要があります。

cron ジョブによる Solaris ボリュームマネージャの監視

▼ ボリュームのエラーを自動的にチェックするには

- **Solaris** ボリュームマネージャ構成のエラーを自動的にチェックするために、cron ユーティリティで定期的に行えるスクリプトを作成します。
次のスクリプト例は、必要に応じて変更することができます。

注 - このスクリプトは、Solaris ボリュームマネージャのエラーチェック機能を自動化するための基本的なスクリプトです。独自の構成に合わせて変更する必要があります。

```
#
#ident "@(#)metacheck.sh 1.3 96/06/21 SMI"
#!/bin/ksh
#!/bin/ksh -x
#!/bin/ksh -v
# ident='%Z%M% %I% %E% SMI'
#
# Copyright (c) 1999 by Sun Microsystems, Inc.
#
# metacheck
#
# メタデバイス構成の状態をチェックする。問題がある場合は、
# ゼロ以外の終了コードを返す。オプションに応じて e メール通知を送信する。
#
```

```

# -h
# ヘルプ
# -s setname
# チェックするセットを指定する。デフォルトでは「local」セットがチェックされる。
# -m recipient [recipient...]
# 指定された受信者に e メール通知を送信する。これは
# 最後の引数でなければならない。この通知は簡単な e メールメッセージとして送信される。
# その件名は次のとおり。
# "Solaris Volume Manager Problem: metacheck.who.nodename.setname"
# これには、問題の要約と詳しい情報の入手方法が示されている。
# 「setname」は -s オプションから、「who」は
# -w オプションからそれぞれとられ、「nodename」は uname(1) によって報告される。
# e メール通知は次のオプションによっても影響される。
# -f 問題が検出された後の追加メッセージを抑制する
# -d 抑制を制御する
# -w だれが e メールを生成したのかを示す
# -t 問題がないときでも e メールを強制的に送る
# -w who
# だれがコマンドを実行しているのかを示す。デフォルトでは、
# これは id(1M) によって報告される user-name である。これは、
# e メール通知を送信する (-m) ときに使用される。
# -f
# フィルタ機能を有効にする。 フィルタ機能は e メール通知 (-m) に適用される。
# フィルタ機能にはルート権限が必要。e メール通知を送信するときに、
# /etc/lvm/metacheck.setname.pending ファイルを使ってフィルタを制御する。
# 次のマトリクスはフィルタの動作を指定する。
#
# 問題の発見      ファイルあり
# yes             no             ファイルを作成し、通知を送信する。
# yes             yes            現在の日付 (-d datefmt で指定された) が
#                  ファイルの日付と異なる場合は、
#                  通知を再び送信する。
# no              yes            ファイルを削除し、問題が解決したことを示す
#                  通知を送信する。
# no              no             -t が指定されている場合は、通知を送信する。
#
# -d datefmt
# フィルタ機能 (-f) の日付形式を指定する。このオプションは、
# e メールで通知を再送する頻度を制御する。
# 指定された形式 (strftime(3C)) に基づく現在の日付が
# /etc/lvm/metacheck.setname.pending ファイルに含まれている日付
# と同じなら、メッセージは抑制される。
# デフォルトの日付形式は「%D」で、通知を 1 日に 1 回再送する
# ことを意味する。
# -t
# テストモード。問題がない場合でも、eメールの生成を有効にする。
# この機構や e メールアドレスを全体的に確認するときに使用する。
#
# 次のオプションは、メタチェックを crontab に組み込む。
# たとえば、次の root crontab エントリがあるとする。
#
# 0,15,30,45 * * * * /usr/sbin/metacheck -f -w SVMcron \
# -d '\%D \%h' -m notice@example.com 2148357243.8333033@pager.example.com

```

```

#
# 15 分ごとに問題をチェックし、問題がある場合には、1 時間ごとに
# eメールを notice@example.com 宛てに生成する (そして、eメール
# ページャサービスに送信する)。crontab エントリの「%」文字の前にある \ に注意。
# 戻ってくる e メールは root@nodename に返される。
# 上の行によって生成される e メールメッセージの件名は次のようになる。
# Solaris Volume Manager Problem: metacheck.SVMcron.nodename.local
#
# 制御端末にデバッグ行を表示する (一連のパイプとして動作)。
decho()
{
    if [ "$debug" = "yes" ] ; then
        echo "DEBUG: $" < /dev/null > /dev/tty 2>&1
    fi
}

# $1 文字列が $2-* に含まれる場合は、$1 を返す。それ以外の場合は "" を返す。
strstr()
{
    typeset    look="$1"
    typeset    ret=""

    shift

# decho "strstr LOOK .$look. FIRST .$1."
while [ $# -ne 0 ] ; do
    if [ "$look" = "$1" ] ; then
        ret="$look"
    fi
    shift
done
echo "$ret"
}

# $1 文字列が $2-* に含まれる場合は、削除して結果を返す。
strdstr()
{
    typeset    look="$1"
    typeset    ret=""

    shift

# decho "strdstr LOOK .$look. FIRST .$1."
while [ $# -ne 0 ] ; do
    if [ "$look" != "$1" ] ; then
        ret="$ret $1"
    fi
    shift
done
echo "$ret"
}

merge_continued_lines()
{
    awk -e '\
BEGIN { line = ""; } \
$NF == "\\\" { \

```

```

        $NF = ""; \
        line = line $0; \
        next; \
    } \
$NF != "\\\" { \
    if ( line != "" ) { \
        print line $0; \
        line = ""; \
    } else { \
        print $0; \
    } \
} \
}'
}

# メタデバイスと関係ない部分を破棄する
find_meta_devices()
{
    typeset    devices=""

#   decho "find_meta_devices .$.*"
    while [ $# -ne 0 ] ; do
        case $1 in
            d+([0-9]) )    # メタデバイス名
                devices="$devices $1"
                ;;
        esac
    shift
done
    echo "$devices"
}

# トップレベルのメタデバイスのリストを返す。
toplevel()
{
    typeset    comp_meta_devices=""
    typeset    top_meta_devices=""
    typeset    devices=""
    typeset    device=""
    typeset    comp=""

    metastat$setarg -p | merge_continued_lines | while read line ; do
        echo "$line"
        devices=`find_meta_devices $line`
        set -- $devices
        if [ $# -ne 0 ] ; then
            device=$1
            shift
            # デバイスがコンポーネントとしてすでに参照されているかチェックする。
            comp=`strstr $device $comp_meta_devices`
            if [ -z $comp ] ; then
                top_meta_devices="$top_meta_devices $device"
            fi
            # コンポーネントリストにコンポーネントを追加し、トップリストから削除する。
            while [ $# -ne 0 ] ; do
                comp=$1

```

```

        comp_meta_devices="$comp_meta_devices $comp"
        top_meta_devices=`strdstr $comp $top_meta_devices`
        shift
    done
fi
done > /dev/null 2>&1
echo $top_meta_devices
}

#
# - MAIN
#
METAPATH=/usr/sbin
PATH=/usr/bin:$METAPATH
USAGE="usage: metacheck [-s setname] [-h] [[-t] [-f [-d datefmt]] \
    [-w who] -m recipient [recipient...]]"

datefmt="%D"
debug="no"
filter="no"
mflag="no"
set="local"
setarg=""
testarg="no"
who='id | sed -e 's/^uid=[0-9][0-9]*(// -e 's/).*//''

while getopts d:Dfms:tw: flag
do
    case $flag in
    d)    datefmt=$OPTARG;
        ;;
    D)    debug="yes"
        ;;
    f)    filter="yes"
        ;;
    m)    mflag="yes"
        ;;
    s)    set=$OPTARG;
        if [ "$set" != "local" ] ; then
            setarg=" -s $set";
        fi
        ;;
    t)    testarg="yes";
        ;;
    w)    who=$OPTARG;
        ;;
    \?)  echo $USAGE
        exit 1
        ;;
    esac
done

# mflag が指定されている場合は、他のすべての部分が受信者に属す。
shift `expr $OPTIND - 1`
if [ $mflag = "no" ] ; then

```

```

        if [ $# -ne 0 ] ; then
            echo $USAGE
            exit 1
        fi
    else
        if [ $# -eq 0 ] ; then
            echo $USAGE
            exit 1
        fi
    fi
recipients="$*"

curdate_filter=`date +%datefmt`
curdate=`date`
node=`uname -n`

# ファイルを確立する。
msg_f=/tmp/metacheck.msg.$$
msgs_f=/tmp/metacheck.msgs.$$
metastat_f=/tmp/metacheck.metastat.$$
metadb_f=/tmp/metacheck.metadb.$$
metahs_f=/tmp/metacheck.metahs.$$
pending_f=/etc/lvm/metacheck.$set.pending
files="$metastat_f $metadb_f $metahs_f $msg_f $msgs_f"

rm -f $files > /dev/null 2>&1
trap "rm -f $files > /dev/null 2>&1; exit 1" 1 2 3 15

# metadb が動作可能かチェックする。
have_metadb="yes"
metadb$setarg > $metadb_f 2>&1
if [ $? -ne 0 ] ; then
    have_metadb="no"
fi
grep "there are no existing databases" < $metadb_f > /dev/null 2>&1
if [ $? -eq 0 ] ; then
    have_metadb="no"
fi
grep "/dev/md/admin" < $metadb_f > /dev/null 2>&1
if [ $? -eq 0 ] ; then
    have_metadb="no"
fi

# metadbs のアクセスに問題がないかチェックする。
retval=0
if [ "$have_metadb" = "no" ] ; then
    retval=1
    echo "metacheck: metadb problem, can't run '$METAPATH/metadb$setarg' " \
        >> $msgs_f
else
    # 状態のスナップショット
    metadb$setarg 2>&1 | sed -e 'ld' | merge_continued_lines > $metadb_f
    metastat$setarg 2>&1 | merge_continued_lines > $metastat_f
    metahs$setarg -i 2>&1 | merge_continued_lines > $metahs_f

```



```

#
# 複製の問題をチェックする。フラグの中の大文字は
# エラーを意味する。フィールドはタブで区切られる。
#
  problem='awk < $metadb_f -F\t '{if ($1 ~ /[A-Z]/) print $1;}'`
if [ -n "$problem" ] ; then
  retval='expr $retval + 64`
echo "\
metacheck: metadb problem, for more detail run:\n\t$METAPATH/metadb$setarg -i" \
      >> $msgs_f
fi

#
# メタデバイスの状態をチェックする。
#
problem='awk < $metastat_f -e \
  '/State:/ {if ($2 != "Okay" && $2 != "Resyncing") print $0;}'`
if [ -n "$problem" ] ; then
  retval='expr $retval + 128`
echo "\
metacheck: metadvice problem, for more detail run:" \
      >> $msgs_f

# 問題があるトップレベルメタデバイスへのメッセージの精度を高める。
top='toplevel`
set -- $top
while [ $# -ne 0 ] ; do
  device=$1
  problem='metastat $device | awk -e \
    '/State:/ {if ($2 != "Okay" && $2 != "Resyncing") print $0;}'`
  if [ -n "$problem" ] ; then
    echo "\t$METAPATH/metastat$setarg $device" >> $msgs_f
    # デバイスに何がマウントされているかを調べる。
    mp='mount|awk -e '/\dev/md/dsk/'$device' [ \t]/{print $1;}'`
    if [ -n "$mp" ] ; then
      echo "\t\t$mp mounted on $device" >> $msgs_f
    fi
  fi
  shift
done
fi

#
# ホットスペアが使用されているかチェックする。
#
problem=""
grep "no hotspare pools found" < $metahs_f > /dev/null 2>&1
if [ $? -ne 0 ] ; then
  problem='awk < $metahs_f -e \
    '/blocks/ { if ( $2 != "Available" ) print $0;}'`
fi
if [ -n "$problem" ] ; then
  retval='expr $retval + 256`
echo "\
metacheck: hot spare in use, for more detail run:\n\t$METAPATH/metahs$setarg -i" \

```

```

        >> $msgs_f
    fi
fi

# エラーが発生している場合は、レポートを送信する。
if [ $retval -ne 0 ] ; then
    if [ -n "$recipients" ] ; then
        re=""
        if [ -f $pending_f ] && [ "$filter" = "yes" ] ; then
            re="Re: "
            # 保留された通知がある。日付をチェックして再送するか決める。
            penddate_filter=`cat $pending_f | head -1`
            if [ "$curdate_filter" != "$penddate_filter" ] ; then
                rm -f $pending_f > /dev/null 2>&1
            else
                if [ "$debug" = "yes" ] ; then
                    echo "metacheck: email problem notification still pending"
                    cat $pending_f
                fi
            fi
        fi
        if [ ! -f $pending_f ] ; then
            if [ "$filter" = "yes" ] ; then
                echo "$curdate_filter\n\tDate:$curdate\n\tTo:$recipients" \
                    > $pending_f
            fi
            echo "\
Solaris Volume Manager: $node: metacheck$setarg: Report: $curdate" >> $msg_f
            echo "\
-----" >> $msg_f
            cat $msg_f $msgs_f | mailx -s \
                "${re}Solaris Volume Manager Problem: metacheck.$who.$set.$node" $recipients
        fi
    else
        cat $msgs_f
    fi
else
    # 問題は検出されていない。
    if [ -n "$recipients" ] ; then
        # デフォルトでは、メールを送信しない。また、何も印刷しない。
        echo "\
Solaris Volume Manager: $node: metacheck$setarg: Report: $curdate" >> $msg_f
        echo "\
-----" >> $msg_f
        if [ -f $pending_f ] && [ "$filter" = "yes" ] ; then
            # 保留されているフィルタが存在する。削除し、OK を送信する。
            rm -f $pending_f > /dev/null 2>&1
            echo "Problem resolved" >> $msg_f
            cat $msg_f | mailx -s \
                "Re: Solaris Volume Manager Problem: metacheck.$who.$node.$set" $recipients
        elif [ "$testarg" = "yes" ] ; then
            # テストのため、問題がなくても、毎回、メールを送信する。
            echo "Messaging test, no problems detected" >> $msg_f
            cat $msg_f | mailx -s \
                "Solaris Volume Manager Problem: metacheck.$who.$node.$set" $recipients
        fi
    fi
fi

```

```
    fi
    else
    echo "metacheck: Okay"
    fi
fi

rm -f $files > /dev/null 2>&1
exit $retval
```

cron ユーティリティを使ってスクリプトを起動する手順については、cron(1M) のマニュアルページを参照してください。

第 24 章

Solaris ボリュームマネージャの障害追跡

この章では、Solaris ボリュームマネージャに関連する問題の解決方法について説明します。また、この章では、障害追跡の一般的な指針と、いくつかの特定の問題を解決するための具体的な手順も示します。

この章では、次の内容について説明します。

- 269 ページの「Solaris ボリュームマネージャ(作業マップ)」
- 270 ページの「障害追跡の概要」
- 271 ページの「一般的な障害追跡方法」
- 274 ページの「ディスク移動の問題からの回復」
- 275 ページの「起動障害」

この章は、Solaris ボリュームマネージャの問題とその解決方法について説明し、一般的なシナリオと回復手順を示すことを目的としています。ここでは、包括的な情報は提供されません。

Solaris ボリュームマネージャ(作業マップ)

次の表に、Solaris ボリュームマネージャの障害追跡に必要な作業を示します。

作業	説明	参照先
不良ディスクを交換する	ディスクを交換してから、新しいディスク上の状態データベースの複製と論理ボリュームを更新します。	272 ページの「不良ディスクを交換するには」

作業	説明	参照先
ディスク移動の問題から回復する	ディスクを元の場所に戻すか、製品サポートに連絡します。	274 ページの「ディスク移動の問題からの回復」
/etc/vfstab 内の不適切なエントリを修正する	ミラーに対して fsck コマンドを実行してから、システムが正しく起動するように /etc/vfstab ファイルを編集します。	276 ページの「/etc/vfstab 内の不適切なエントリを修正するには」
起動デバイスの障害から回復する	他のサブミラーから起動します。	278 ページの「起動デバイスの障害から回復するには」
状態データベースの複製数の不足から回復する	metadb コマンドを使って、使用不能な複製を削除します。	282 ページの「状態データベースの複製数の不足から回復するには」
ソフトパーティションの失われた構成データを復元する	metarecover コマンドを使って、ソフトパーティションの構成データを復元します。	286 ページの「ソフトパーティションの構成データを復元するには」
異なるディスクから Solaris ボリュームマネージャ構成を復元する	新しいシステムにディスクを追加し、既存の状態データベースの複製から Solaris ボリュームマネージャ構成を再構築します。	289 ページの「構成を復元するには」

障害追跡の概要

障害追跡の前提条件

Solaris ボリュームマネージャの記憶領域管理に関連する問題を解決するには、次の条件を満たしている必要があります。

- ルート権限を持っている
- すべてのデータの最新バックアップを取っている

Solaris ボリュームマネージャによる障害追跡の一般的な指針

Solaris ボリュームマネージャで障害追跡を行うときは、次の情報を用意してください。

- metadb コマンドの出力

- `metastat` コマンドの出力
- `metastat -p` コマンドの出力
- `/etc/vfstab` ファイルのバックアップコピー
- `/etc/lvm/mddb.cf` ファイルのバックアップコピー
- `prtvtoc` コマンド (SPARC™ システム) または `fdisk` コマンド (IA ベースのシステム) の出力 (ディスクパーティションの情報)
- Solaris のバージョン
- インストールされている Solaris のパッチ
- インストールされている Solaris ボリュームマネージャのパッチ

ヒント – Solaris ボリュームマネージャ構成を更新したり、記憶領域やオペレーティング環境に関連するその他の変更をシステムに適用した場合は、その構成情報の最新コピーを生成してください。cron ジョブを使えば、この情報を自動的に生成できます。

一般的な障害追跡方法

Solaris ボリュームマネージャに関連するすべての問題の検証を可能にする手順はありませんが、一般には次の手順に従って障害を追跡します。

1. 現在の構成に関する情報を収集します。
2. `metastat` や `metadb` コマンドの出力など、最新の状態情報を調べます。これらの出力から、どのコンポーネントで障害が発生したかを示す情報が得られます。
3. 障害が起こりそうなハードウェア部分をチェックします (すべてのハードウェアが適切に接続されているか、最近、停電がなかったか、最近、機器を追加または変更しなかったか、など)。

ディスクの交換

この節では、Solaris ボリュームマネージャ環境でディスクを交換する方法について説明します。



注意 – 不良ディスク上または不良ディスク上に構築したボリューム上でソフトウェアパーティションを使用していた場合は、新しいディスクを交換対象ディスクと同じ物理位置に配置し、同じ `c*t*d` * 番号を使用する必要があります。

▼ 不良ディスクを交換するには

1. `/var/adm/messages` ファイルと `metastat` コマンドの出力を調べて、交換する不良ディスクを特定します。
2. 不良ディスクに状態データベースの複製がないかチェックします。

`metadb` コマンドを使って複製を表示します。

`metadb` コマンドの出力を見ると、不良ディスクの状態データベースの複製にエラーが表示されていることがわかります。この例では、`c0t1d0` のデバイスに問題があります。

```
# metadb
  flags      first blk      block count
a m   u       16             1034         /dev/dsk/c0t0d0s4
a     u       1050          1034         /dev/dsk/c0t0d0s4
a     u       2084          1034         /dev/dsk/c0t0d0s4
W pc luo    16             1034         /dev/dsk/c0t1d0s4
W pc luo    1050          1034         /dev/dsk/c0t1d0s4
W pc luo    2084          1034         /dev/dsk/c0t1d0s4
```

この出力には、ローカルディスク `c0t0d0` と `c0t1d0` のスライス 4 にそれぞれ 3 つの状態データベースの複製があることがわかります。`c0t1d0s4` スライスのフラグフィールドの `W` は、このデバイスに書き込みエラーがあることを示しています。`c0t0d0s4` スライスの 3 つの複製は正常です。

3. 状態データベースの複製があるスライス名とその複製の数を書き留めておき、状態データベースの複製を削除します。

状態データベースの複製の数は、`metadb` コマンド出力に同じスライス名が表示される行数と同じです。この例では、`c0t1d0s4` にある 3 つの状態データベースの複製を削除します。

```
# metadb -d c0t1d0s4
```



注意 – 不良の状態データベースの複製を削除すると、複製の数が 3 以下になることがあります。その場合は、状態データベースの複製を追加してから先に進みます。これによって、前と同じ構成情報が保たれます。

4. 不良ディスクにホットスペアがないか探して、あれば削除します。ホットスペアの検索には、`metastat` コマンドを使用します。この例では、`c0t1d0s6` がホットスペア集合 `hsp000` に含まれていたため、集合から削除します。


```
# metahs -d hsp000 c0t1d0s6
hsp000: Hotspare is deleted
```

5. 不良ディスクを物理的に交換します。
6. `devfsadm` コマンド、`luxadm` コマンド、または使用ハードウェアと環境に適した他のコマンドを使って、不良ディスクを論理的に交換します。

7. 新しいディスクのパーティションを再分割します。
`format` か `fmthard` コマンドを使い、不良ディスクと同じスライス情報に基づいてディスクをパーティション分割します。不良ディスクに対する `prtvtoc` の出力がある場合は、`fmthard -s /tmp/failed-disk-prtvtoc-output` で新しいディスクをフォーマットできます。

8. 状態データベースの複製を削除した場合は、同じ数の複製を適切なスライスに追加します。

この例では、`/dev/dsk/c0t1d0s4` を使用します。

```
# metadb -a -c 3 c0t1d0s4
```

9. ディスク上のスライスが、**RAID 5** ボリュームのコンポーネントである場合、あるいは **RAID 0** ボリュームのコンポーネントで、それが **RAID 1** ボリュームのサブミラーになっている場合は、各スライスに `metareplace -e` コマンドを実行します。
この例では、`/dev/dsk/c0t1d0s4` およびミラー `d10` を使用します。

```
# metareplace -e d10 c0t1d0s4
```

10. ソフトパーティションが交換ディスクのスライス上に直接構築されている場合は、ソフトパーティションのあるスライスごとに `metarecover -d -p` コマンドを実行して、ディスクのエクステンツヘッダーを再生します。

この例では、`/dev/dsk/c0t1d0s4` の再作成されたディスクにソフトパーティションのマーキングが必要なので、このディスクをスキャンし、状態データベースの複製の情報に基づいてマーキングを再度適用します。

```
# metarecover c0t1d0s4 -d -p
```

11. ディスク上のソフトパーティションが、**RAID 5** ボリュームのコンポーネントである場合、あるいは **RAID 0** ボリュームのコンポーネントで、それが **RAID 1** ボリュームのサブミラーになっている場合は、各スライスに `metareplace -e` コマンドを実行します。

この例では、`/dev/dsk/c0t1d0s4` およびミラー `d10` を使用します。

```
# metareplace -e d10 c0t1d0s4
```

12. **RAID 0** ボリューム上にソフトパーティションが構築されている場合は、各 **RAID 0** ボリュームに `metarecover` コマンドを実行します。

この例では、**RAID 0** ボリューム `d17` にソフトパーティションが構築されています。

```
# metarecover d17 -m -p
```

- 削除されたホットスペアを置き換え、それを1つまたは複数の適切なホットスペア集合に追加します。

```
# metahs -a hsp000 c0t0d0s6
hsp000: Hotspare is added
```

- ソフトパーティションまたは冗長ボリュームが障害の影響を受けた場合は、バックアップからデータを復元します。冗長ボリュームだけが影響を受けた場合は、データを検証します。

すべてのボリュームのユーザーデータやアプリケーションデータをチェックします。必要であれば、アプリケーションレベルの整合性チェックプログラムなどのツールを使ってデータをチェックします。

ディスク移動の問題からの回復

ここでは、Solaris ボリュームマネージャ環境でディスクを移動させた後に発生する予想外の問題から回復する方法について説明します。

▼ ディスク移動とデバイス ID の概要

Solaris ボリュームマネージャでは、特定のディスクと対応づけられたデバイス ID を使用して、Solaris ボリュームマネージャ構成で使用されているあらゆるディスクを追跡します。ディスクを別のコントローラに移した場合、または SCSI ターゲット番号が変更された場合、通常は Solaris ボリュームマネージャが移動を正しく認識して、関連するすべての Solaris ボリュームマネージャレコードを相応に更新するので、システム管理者の介入は不要です。とはいえ、まれに、Solaris ボリュームマネージャがレコードを正しく更新できず、起動時にエラーが通知されることがあります。

名前のないデバイスに関するエラーメッセージを解決するには

新しいハードウェアを追加したり、ハードウェアを移動させると (あるコントローラから別のコントローラに一連のディスクを移動させた場合など)、Solaris ボリュームマネージャが、移動されたディスクに対応するデバイス ID を調べ、内部 Solaris ボリュームマネージャレコードの `c*t*d*` 名を適切に更新します。レコードを更新できなかった場合、`/etc/rc2.d/S95svm.sync (/etc/init.d/svm.sync へのリンク)` によって生成された起動プロセスが、起動時に次のようなエラーをコンソールに通知します。

```
Unable to resolve unnamed devices for volume management.
Please refer to the Solaris Volume Manager documentation,
Troubleshooting section, at http://docs.sun.com or from
your local copy."
```

この問題によってデータが消失したわけでも、特に何かが起きるわけでもありません。Solaris ボリュームマネージャの名前レコードが一部しか更新されなかったため、`metastat` コマンドの出力に、以前使用されていた `c*t*d*` 名と移動後の状態が反映された `c*t*d*` 名が混在する可能性があることを、このエラーメッセージは伝えています。

この条件下で Solaris ボリュームマネージャ構成の更新が必要になった場合は、`meta*` コマンドを実行する際に、必ず `metastat` コマンドの出力どおりの `c*t*d*` 名を使ってください。

このエラー条件が発生した場合、次の方法のどちらかで解決できます。

- すべてのディスクを元の場所に戻す。次に、再構成時の再起動を実行するか、(単一コマンドとして) 次のコマンドを実行する

```
/usr/sbin/devfsadm && /usr/sbin/metadevadm -r
```

コマンドの完了後、エラー条件が解決されるので、処理を続けることができます。

- サポートの担当者に連絡し、指示を受ける

注 - このエラー条件はめったに発生しません。万一発生した場合は、高い確率で、ファイバチャネルに接続された記憶装置が影響を受けます。

起動障害

Solaris ボリュームマネージャではルート (/)、`swap`、および `/usr` ディレクトリをミラー化できるため、システムの起動時に、ハードウェア障害やオペレータの操作ミスによって特殊な障害が発生することがあります。この節の作業は、そのような障害に対処するためのものです。

次の表に、そのような障害の原因と適切な解決方法を示します。

表 24-1 Solaris ボリュームマネージャの一般的な起動障害

障害の原因	参照先
<code>/etc/vfstab</code> ファイルの情報が正しくない	276 ページの「 <code>/etc/vfstab</code> 内の不適切なエントリを修正するには」
状態データベースの複製の数が足りない	282 ページの「状態データベースの複製数の不足から回復するには」
起動デバイス (ディスク) に障害が発生した	278 ページの「起動デバイスの障害から回復するには」

表 24-1 Solaris ボリュームマネージャの一般的な起動障害 (続き)

障害の原因	参照先
起動ミラーに障害が発生した	

起動障害の背景情報

- エラーのためにボリュームが Solaris ボリュームマネージャによってオフラインにされた場合は、障害が発生したディスクにあるすべてのファイルシステムのマウントを解除する必要があります。個々のディスクスライスは独立しているため、同じディスクに複数のファイルシステムがマウントされていることがあります。ドライバソフトウェアに障害が発生した場合には、同じディスクの他のスライスでもまもなく障害が発生するはずですが、ディスクスライスに直接マウントされているファイルシステムは Solaris ボリュームマネージャのエラー処理対象ではないため、そのようなファイルシステムをマウントしたままにしておくと、システムのクラッシュによってデータを失う恐れがあります。
- サブミラーを無効な状態やオフラインのままにしておく時間を最小限に抑えます。再同期やオンラインバックアップの処理中は、ミラー化による保護は不完全になります。

▼ /etc/vfstab 内の不適切なエントリを修正するには

たとえば、ルート (/) のミラー化などの際に、/etc/vfstab ファイルに正しくないエントリを設定すると、システムは最初は正しく起動しているように見えますが、やがて起動に失敗します。この問題を解決するためには、シングルユーザーモードで /etc/vfstab ファイルを編集する必要があります。

/etc/vfstab ファイル内の不適切なエントリを修正するおおよその手順は次のとおりです。

1. シングルユーザーモードでシステムを起動します。
2. ミラーボリュームに対して fsck コマンドを実行します。
3. ファイルシステムを読み取り/書き込みモードでマウントし直します。
4. (省略可能) ルート (/) ミラーに対して metaroot コマンドを実行します。
5. /etc/vfstab ファイル内のファイルシステムエントリがこのボリュームを参照していることを確認します。
6. 再起動します。

例 — ルート (/) ミラーを回復する

次の例では、ルート (/) が 2 面ミラー d0 でミラー化されています。/etc/vfstab ファイルのルート (/) エントリが何らかの理由でこのファイルシステムの元のスライスに戻されていますが、/etc/system ファイルの情報は、起動がミラー d0 から行われるようになっています。最も一般的な原因としては、metaroot コマンドを使わずに /etc/system や /etc/vfstab ファイルを変更したか、/etc/vfstab ファイルの古いコピーを復元したことが考えられます。

間違った /etc/vfstab ファイルの例を以下に示します。

```
#device      device      mount      FS      fsck      mount      mount
#to mount    to fsck     point      type    pass     at boot    options
#
/dev/dsk/c0t3d0s0 /dev/rdisk/c0t3d0s0 /      ufs      1      no      -
/dev/dsk/c0t3d0s1 -           -          swap     -      no      -
/dev/dsk/c0t3d0s6 /dev/rdisk/c0t3d0s6 /usr    ufs      2      no      -
#
/proc        -           /proc     proc     -      no      -
swap        -           /tmp      tmpfs    -      yes     -
```

エラーがあるために、システムは、起動時に自動的にシングルユーザーモードになります。

```
ok boot
...
configuring network interfaces: hme0.
Hostname: lexicon
mount: /dev/dsk/c0t3d0s0 is not this fstype.
setmnt: Cannot open /etc/mnttab for writing

INIT: Cannot create /var/adm/utmp or /var/adm/utmpx

INIT: failed write of utmpx entry: " "

INIT: failed write of utmpx entry: " "

INIT: SINGLE USER MODE

Type Ctrl-d to proceed with normal startup,
(or give root password for system maintenance): <root-password>
```

この時点では、ルート (/) と /usr は読み取り専用でマウントされています。次の手順を実行します。

1. ルート (/) ミラーに対して fsck コマンドを実行します。

注 — ルートのボリューム名を間違えないでください。

```
# fsck /dev/md/rdsk/d0
** /dev/md/rdsk/d0
** Currently Mounted on /
```

```

** Phase 1 - Check Blocks and Sizes
** Phase 2 - Check Pathnames
** Phase 3 - Check Connectivity
** Phase 4 - Check Reference Counts
** Phase 5 - Check Cyl groups
2274 files, 11815 used, 10302 free (158 frags, 1268 blocks,
0.7% fragmentation)

```

2. /etc/vfstab ファイルを編集できるように、ルート (/) を読み取り/書き込みモードでマウントし直します。

```

# mount -o rw,remount /dev/md/dsk/d0 /
mount: warning: cannot lock temp file </etc/.mnt.lock>

```

3. metaroot コマンドを実行します。

```

# metaroot d0

```

このコマンドを実行すると、/etc/system と /etc/vfstab ファイルが編集され、ルート (/) ファイルシステムのエントリがボリューム d0 を参照します。

4. /etc/vfstab ファイルに正しいボリュームのエントリが含まれていることを確認します。

/etc/vfstab ファイルのルート (/) エントリは次のようになっているはずです。これによって、ファイルシステムのエントリは RAID 1 ボリュームを正しく参照します。

#device	device	mount	FS	fsck	mount	mount
#to mount	to	point	type	pass	at	boot options
#	/dev/md/dsk/d0	/dev/md/rdisk/d0	/	ufs	1	no -
	/dev/dsk/c0t3d0s1	-	-	swap	-	no -
	/dev/dsk/c0t3d0s6	/dev/rdisk/c0t3d0s6	/usr	ufs	2	no -
#	/proc	-	/proc	proc	-	no -
	swap	-	/tmp	tmpfs	-	yes -

5. システムを再起動します。
システムは正常な動作に戻ります。

▼ 起動デバイスの障害から回復するには

ルート (/) がミラー化されているシステムの起動デバイスに障害がある場合は、代替起動デバイスを設定する必要があります。

この手順の概要は次のとおりです。

1. 代替ルート (/) サブミラーからシステムを起動します。
2. エラーのある状態データベースの複製とボリュームを特定します。
3. 不良ディスクを修復します。

4. 状態データベースの複製とボリュームを元の状態に復元します。

次の例では、6つの状態データベースの複製のうち2つが起動デバイスにあり、ルート(/)、swap、および/usrの各サブミラーが使用不能になっています。

起動デバイスに障害が発生すると、最初に次のようなメッセージが表示されます。このメッセージは、アーキテクチャによってこれとは異なる場合があります。

```
Rebooting with command:
Boot device: /iommu/sbus/dma@f,81000/esp@f,80000/sd@3,0
The selected SCSI device is not responding
Can't open boot device
...
```

このメッセージが表示された場合は、このデバイス名を書き留めておき、次の手順を実行します。

1. ルート(/)の別のサブミラーからシステムを起動します。

この例では2つの状態データベースの複製だけが異常であるため、システムを起動することができます。起動できない場合は、アクセスできない状態データベースの複製をシングルユーザーモードで削除する必要があります。この手順については、282ページの「状態データベースの複製数の不足から回復するには」を参照してください。

ルート(/)ファイルシステムのミラーを作成したときに、その手順の中で代替起動デバイスを書き留めてあるはずですが、この例では、disk2が代替起動デバイスになります。

```
ok boot disk2
SunOS Release 5.9 Version s81_51 64-bit
Copyright 1983-2001 Sun Microsystems, Inc. All rights reserved.
Hostname: demo
...
demo console login: root
Password: <root-password>
Dec 16 12:22:09 lexicon login: ROOT LOGIN /dev/console
Last login: Wed Dec 12 10:55:16 on console
Sun Microsystems Inc. SunOS 5.9 s81_51 May 2002
...
```

2. metadb コマンドを使って、2つの状態データベースの複製が使用不能であることを確認します。

```
# metadb
      flags          first blk    block count
M     p             unknown     unknown    /dev/dsk/c0t3d0s3
M     p             unknown     unknown    /dev/dsk/c0t3d0s3
a m  p luo         16          1034      /dev/dsk/c0t2d0s3
a     p luo         1050        1034      /dev/dsk/c0t2d0s3
a     p luo         16          1034      /dev/dsk/c0t1d0s3
a     p luo         1050        1034      /dev/dsk/c0t1d0s3
```

不良ディスクの一部であるスライス /dev/dsk/c0t3d0s3 上の状態データベースの複製は、システムから認識されません。

3. `metastat` コマンドを使って、ルート (/)、`swap`、および `/usr` の各ミラーの半分が使用不能であることを確認します。

```
# metastat
d0: Mirror
  Submirror 0: d10
    State: Needs maintenance
  Submirror 1: d20
    State: Okay
...

d10: Submirror of d0
  State: Needs maintenance
  Invoke: "metareplace d0 /dev/dsk/c0t3d0s0 <new device>"
  Size: 47628 blocks
  Stripe 0:
  Device                Start Block  Dbase State          Hot Spare
  /dev/dsk/c0t3d0s0      0           No   Maintenance

d20: Submirror of d0
  State: Okay
  Size: 47628 blocks
  Stripe 0:
  Device                Start Block  Dbase State          Hot Spare
  /dev/dsk/c0t2d0s0      0           No   Okay

d1: Mirror
  Submirror 0: d11
    State: Needs maintenance
  Submirror 1: d21
    State: Okay
...

d11: Submirror of d1
  State: Needs maintenance
  Invoke: "metareplace d1 /dev/dsk/c0t3d0s1 <new device>"
  Size: 69660 blocks
  Stripe 0:
  Device                Start Block  Dbase State          Hot Spare
  /dev/dsk/c0t3d0s1      0           No   Maintenance

d21: Submirror of d1
  State: Okay
  Size: 69660 blocks
  Stripe 0:
  Device                Start Block  Dbase State          Hot Spare
  /dev/dsk/c0t2d0s1      0           No   Okay

d2: Mirror
  Submirror 0: d12
    State: Needs maintenance
  Submirror 1: d22
    State: Okay
...
```



```

d2: Mirror
  Submirror 0: d12
    State: Needs maintenance
  Submirror 1: d22
    State: Okay
...

d12: Submirror of d2
  State: Needs maintenance
  Invoke: "metareplace d2 /dev/dsk/c0t3d0s6 <new device>"
  Size: 286740 blocks
  Stripe 0:
    Device                Start Block  Dbase State          Hot Spare
    /dev/dsk/c0t3d0s6      0           No   Maintenance
...

d22: Submirror of d2
  State: Okay
  Size: 286740 blocks
  Stripe 0:
    Device                Start Block  Dbase State          Hot Spare
    /dev/dsk/c0t2d0s6      0           No   Okay

```

この例では、次のサブミラーの保守が必要であることがわかります。

- サブミラー d10、デバイス c0t3d0s0
- サブミラー d11、デバイス c0t3d0s1
- サブミラー d12、デバイス c0t3d0s6

4. システムを停止して、ディスクを交換し、`format` か `fmthard` コマンドを使って、ディスクのパーティションを障害前と同じ状態に分割します。

ヒント – 新しいディスクが他方のディスク (つまり、ミラーの別の半分があるディスク) と同じ構成であれば、`prtvtoc /dev/rdisk/c0t2d0s2 | fmthard - -s /dev/rdisk/c0t3d0s2` を使って、新しいディスク (c0t3d0) を短時間でフォーマットできます。

```

# halt
...
Halted
...
ok boot
...
# format /dev/rdisk/c0t3d0s0

```

5. システムを再起動します。

ここでは、ルート (/) ミラーの他方の半分から起動する必要があります。この代替起動デバイスは、ミラーを作成したときに書き留めておいたものです。

```

# halt
...
ok boot disk2

```

6. `metadb` コマンドを使って、使用不能な状態データベースの複製を削除してから、新しい複製を追加します。

```
# metadb
      flags          first blk    block count
M     p             unknown      unknown    /dev/dsk/c0t3d0s3
M     p             unknown      unknown    /dev/dsk/c0t3d0s3
a m  p  luo         16           1034      /dev/dsk/c0t2d0s3
a   p  luo         1050          1034      /dev/dsk/c0t2d0s3
a   p  luo          16           1034      /dev/dsk/c0t1d0s3
a   p  luo         1050          1034      /dev/dsk/c0t1d0s3
# metadb -d c0t3d0s3
# metadb -c 2 -a c0t3d0s3
# metadb
      flags          first blk    block count
a m  p  luo         16           1034      /dev/dsk/c0t2d0s3
a   p  luo         1050          1034      /dev/dsk/c0t2d0s3
a   p  luo          16           1034      /dev/dsk/c0t1d0s3
a   p  luo         1050          1034      /dev/dsk/c0t1d0s3
a     u             16           1034      /dev/dsk/c0t3d0s3
a     u             1050          1034      /dev/dsk/c0t3d0s3
```

7. `metareplace` コマンドを使ってサブミラーを有効にします。

```
# metareplace -e d0 c0t3d0s0
Device /dev/dsk/c0t3d0s0 is enabled

# metareplace -e d1 c0t3d0s1
Device /dev/dsk/c0t3d0s1 is enabled

# metareplace -e d2 c0t3d0s6
Device /dev/dsk/c0t3d0s6 is enabled
```

しばらくすると、再同期処理が終了します。これで、また元のデバイスから起動できるようになります。

状態データベースの複製の障害からの回復

▼ 状態データベースの複製数の不足から回復するには

ドライブ障害などのために、状態データベースの複製の数が規定数に達していないと、システムをマルチユーザーモードで再起動できなくなります。この状態になるのは、使用可能な状態データベースの複製の数が半数に達しないことを Solaris ボリューム

ムマネージャが検出した後 (パニックの後) や、使用可能な状態データベースの複製の数が半数以下の状態でシステムが再起動されたときなどです。Solaris ボリュームマネージャでは、これを、状態データベースの複製が「無効」状態になったといいます。この問題から回復するための手順を次に示します。

1. システムを起動して、どの状態データベースの複製に障害があるのか調べます。
2. 使用不能な状態データベースの複製を特定します。
次の形式の `metadb` コマンドを実行します。

```
metadb -i
```

3. 1つまたは複数のディスクが使用不能であることがわかっている場合は、それらのディスク上の状態データベースの複製をすべて削除します。そうでない場合は、障害のある状態データベースの複製 (`metadb` の出力のステータスフラグ **W**、**M**、**D**、**F**、**R** で示される) を削除して、状態データベースの複製の過半数が正常なものであるようにします。

`metadb -d` コマンドを使って、不良ディスク上の状態データベースの複製を削除します。

ヒント – 大文字の状態フラグが付いている状態データベースの複製は障害のある複製であり、小文字のフラグが付いているものは正常な複製です。

4. `metadb` コマンドを使って、複製が削除されていることを確認します。
5. システムを再起動します。
6. 必要であれば、ディスクを交換し、適切にフォーマットしてから、必要な状態データベースの複製をディスクに追加します。詳細は、64 ページの「状態データベースの複製の作成」の手順を参照してください。
交換用のディスクが用意できたら、システムを停止し、不良ディスクを交換してから、システムを再起動します。 `format` か `fmthard` コマンドを使って、障害の発生前と同じようにディスクをパーティション分割します。

例 — 状態データベースの複製数の不足から回復する

この例では、7つの状態データベースの複製を含むディスクに障害が発生したものとします。システムには正常な複製が3つしか残らないため、システムはパニック状態になり、マルチユーザーモードで再起動できなくなります。

```
panic[cpu0]/thread=70a41e00: md: state database problem

403238a8 md:mddb_commitrec_wrapper+6c (2, 1, 70a66ca0, 40323964, 70a66ca0, 3c)
  %l0-7: 0000000a 00000000 00000001 70bbcc0 70bbcd04 70995400 00000002 00000000
40323908 md:alloc_entry+c4 (70b00844, 1, 9, 0, 403239e4, ff00)
  %l0-7: 70b796a4 00000001 00000000 705064cc 70a66ca0 00000002 00000024 00000000
```

```

40323968 md:md_setdevname+2d4 (7003b988, 6, 0, 63, 70a71618, 10)
  %l0-7: 70a71620 00000000 705064cc 70b00844 00000010 00000000 00000000 00000000
403239f8 md:setnm_ioctl+134 (7003b968, 100003, 64, 0, 0, ffbffc00)
  %l0-7: 7003b988 00000000 70a71618 00000000 00000000 000225f0 00000000 00000000
40323a58 md:md_base_ioctl+9b4 (157ffff, 5605, ffbffa3c, 100003, 40323ba8, ff1b5470)
  %l0-7: ff3f2208 ff3f2138 ff3f26a0 00000000 00000000 00000064 ff1396e9 00000000
40323ad0 md:md_admin_ioctl+24 (157ffff, 5605, ffbffa3c, 100003, 40323ba8, 0)
  %l0-7: 00005605 ffbffa3c 00100003 0157ffff 0aa64245 00000000 7efefeff 81010100
40323b48 md:mdioctl+e4 (157ffff, 5605, ffbffa3c, 100003, 7016db60, 40323c7c)
  %l0-7: 0157ffff 00005605 ffbffa3c 00100003 0003ffff 70995598 70995570 0147c800
40323bb0 genunix:ioctl+1dc (3, 5605, ffbffa3c, ffffffff8, fffffffe0, ffbffa65)
  %l0-7: 0114c57c 70937428 ff3f26a0 00000000 00000001 ff3b10d4 0aa64245 00000000

```

```

panic:
stopped at      edd000d8:      ta      %icc,%g0 + 125
Type 'go' to resume

```

```

ok boot -s
Resetting ...

```

```

Sun Ultra 5/10 UPA/PCI (UltraSPARC-III 270MHz), No Keyboard
OpenBoot 3.11, 128 MB memory installed, Serial #9841776.
Ethernet address 8:0:20:96:2c:70, Host ID: 80962c70.

```

```

Rebooting with command: boot -s
Boot device: /pci@1f,0/pci@1,1/ide@3/disk@0,0:a File and args: -s
SunOS Release 5.9 Version s81_39 64-bit

```

```

Copyright 1983-2001 Sun Microsystems, Inc. All rights reserved.
configuring IPv4 interfaces: hme0.
Hostname: dodo

```

```
metainit: dodo: stale databases
```

```
Insufficient metadb database replicas located.
```

```

Use metadb to delete databases which are broken.
Ignore any "Read-only file system" error messages.
Reboot the system when finished to reload the metadb database.
After reboot, repair any broken database replicas which were deleted.

```

```

Type control-d to proceed with normal startup,
(or give root password for system maintenance): root password
single-user privilege assigned to /dev/console.
Entering System Maintenance Mode

```

```

Jun  7 08:57:25 su: 'su root' succeeded for root on /dev/console
Sun Microsystems Inc. SunOS 5.9 s81_39 May 2002

```

```

# metadb -i
      flags          first blk      block count
a m p lu           16             8192      /dev/dsk/c0t0d0s7
a p l              8208             8192      /dev/dsk/c0t0d0s7
a p l             16400             8192      /dev/dsk/c0t0d0s7

```

```

M    p    16    unknown    /dev/dsk/c1t1d0s0
M    p    8208    unknown    /dev/dsk/c1t1d0s0
M    p    16400    unknown    /dev/dsk/c1t1d0s0
M    p    24592    unknown    /dev/dsk/c1t1d0s0
M    p    32784    unknown    /dev/dsk/c1t1d0s0
M    p    40976    unknown    /dev/dsk/c1t1d0s0
M    p    49168    unknown    /dev/dsk/c1t1d0s0
# metadb -d c1t1d0s0
# metadb
      flags      first blk      block count
a m p l u      16            8192          /dev/dsk/c0t0d0s7
a    p    l      8208            8192          /dev/dsk/c0t0d0s7
a    p    l     16400            8192          /dev/dsk/c0t0d0s7
#

```

システムがパニック状態になったのは、状態データベースの複製をスライス /dev/dsk/c1t1d0s0 上で検出できなかったためです (このスライスは、障害が発生したディスクの一部であるか、障害が発生したコントローラに接続されています)。最初の metadb - i コマンドによって、このスライス上の複製のマスターブロックに問題があることがわかります。

無効状態の状態データベースの複製を削除する場合、ルート (/) ファイルシステムは読み取り専用になっています。mddb.cf エラーメッセージが表示されますが、無視してください。

この時点でシステムは再び動作できるようになりますが、おそらく、システムには本来の数よりも少ない状態データベースの複製しかありません。また、障害が発生した記憶領域を使用していたボリュームは異常状態またはエラー状態になっているか、あるいはホットスペアで置き換えられています。このような問題は、速やかに解決しなければなりません。

トランザクションボリュームの修復

トランザクションボリュームは、マスターデバイスとロギングデバイスからなる「階層構造」ボリュームです。また、ロギングボリュームは複数のファイルシステムによって共有されることがあるため、障害のあるトランザクションボリュームを修復する作業には、特別な回復手順が必要です。

デバイスのエラーやパニックの修復には、コマンド行ユーティリティを使用する必要があります。

パニック

ファイルシステムは、使用されている間に内部的な不整合を検出すると、システムをパニック状態にします。また、ファイルシステムがロギングとして構成されている場合には、再起動時にファイルシステムのチェックが必要であることをトランザクションボリュームに通知します。トランザクションボリュームは、それ自身を「ハードウェアエラー (Hard Error)」状態にします。また、同じログデバイスを共有するすべてのトランザクションボリュームも「ハードウェアエラー (Hard Error)」状態になります。

再起動時に `fsck` は、このファイルシステムをチェック、修復し、「正常 (Okay)」状態に戻します。 `fsck` は、 `/etc/vfstab` ファイルにリストされているトランザクションボリュームのうち、このログデバイスを共有するすべてのトランザクションボリュームに対してこの処理を実行します。

トランザクションボリュームのエラー

トランザクションボリュームがロギングデータを処理しているときにマスターデバイスかログデバイスでデバイスエラーが起こると、そのデバイスは「正常 (Okay)」状態から「ハードウェアエラー (Hard Error)」状態に移行します。デバイスが「ハードウェアエラー (Hard Error)」状態か「エラー (Error)」状態の場合は、デバイスエラーまたはパニックが起きたことを意味します。

また、障害が発生したログデバイスを共有するデバイスも「エラー (Error)」状態になります。

ソフトパーティション障害からの回復

次の各項では、ソフトパーティションの構成情報を復元する方法について説明します。この方法は、状態データベースの複製がすべて失われており、また `metastat -p` の出力、 `md.cf` ファイル、または最新の `md.tab` ファイルの最新コピーあるいは正確なコピーがまったくないときにだけ使用します。

ソフトパーティションの構成データを復元するには

各ソフトパーティションの先頭部分には、ソフトパーティションエクステンットの開始位置を示すセクターがあります。これらの隠しセクターはエクステンットヘッダーと呼ばれ、ソフトパーティションのユーザーには見えません。すべての Solaris ボリュームマネージャ構成が失われた場合には、ディスクをスキャンして構成データを生成することができます。

この手順は、失われたソフトパーティションの構成情報を復元する最後の手段です。したがって、`metarecover` コマンドは、`metadb` ファイルと `md.cf` ファイルの両方が失われており、また `md.tab` も失われているか、`md.tab` が古い場合にのみ使用します。

注 – この手順は、ソフトパーティションの情報を復元するためのものです。したがって、他の構成や他の Solaris ボリュームマネージャボリュームの構成情報を復元するためには使用できません。

注 – ソフトパーティション上に他の Solaris ボリュームマネージャボリュームが作成されている場合には、ソフトパーティションを復元してから他のボリュームを復元する必要があります。

ソフトパーティションの構成情報は、デバイスと状態データベースに格納されています。どちらかのソースが破壊されている可能性があるため、どちらが信頼できるソースなのかを `metarecover` コマンドに知らせる必要があります。

最初に、`metarecover` コマンドを使って2つのソースが一致するかどうかを判定します。両者が一致する場合は、`metarecover` コマンドを使って変更を行うことはできません。両者が一致しない場合は、出力を慎重に検討して、ディスクと状態データベースのどちらが破壊されているか判定する必要があります。次に `metarecover` コマンドを使い、適切なソースに基づいて構成を再構築します。

1. 130 ページの「ソフトパーティションの背景情報」を確認します。
2. `metarecover` コマンドを実行し、出力されたソフトパーティション回復情報を検討します。

metarecover component -p {-d }

ここで、コンポーネントには、`raw` コンポーネントの `c*t*d*s*` 名を使用します。`-d` オプションは、物理スライスをスキャンして、ソフトパーティションのエクステンツヘッダーを検出することを意味します。

詳細は、`metarecover(1M)` のマニュアルページを参照してください。

例 — ディスク上のエクステンツヘッダーからソフトパーティションを復元する

```
# metarecover c1t1d0s1 -p -d
The following soft partitions were found and will be added to
your metadvice configuration.
  Name           Size      No. of Extents
  ----           -
  d10            10240         1
  d11            10240         1
```

```

d12          10240          1
# metarecover c1t1d0s1 -p -d
The following soft partitions were found and will be added to
your metadevice configuration.
Name          Size      No. of Extents
d10           10240          1
d11           10240          1
d12           10240          1
WARNING: You are about to add one or more soft partition
metadevices to your metadevice configuration.  If there
appears to be an error in the soft partition(s) displayed
above, do NOT proceed with this recovery operation.
Are you sure you want to do this (yes/no)?yes
c1t1d0s1: Soft Partitions recovered from device.
bash-2.05# metastat
d10: Soft Partition
Device: c1t1d0s1
State: Okay
Size: 10240 blocks
Device          Start Block  Dbase Reloc
c1t1d0s1         0           No      Yes

Extent          Start Block          Block count
0                1                    10240

d11: Soft Partition
Device: c1t1d0s1
State: Okay
Size: 10240 blocks
Device          Start Block  Dbase Reloc
c1t1d0s1         0           No      Yes

Extent          Start Block          Block count
0                10242           10240

d12: Soft Partition
Device: c1t1d0s1
State: Okay
Size: 10240 blocks
Device          Start Block  Dbase Reloc
c1t1d0s1         0           No      Yes

Extent          Start Block          Block count
0                20483           10240

```

この例では、すべての状態データベースの複製が誤って削除されているときに、ディスクから3つのソフトパーティションを復元します。

別のシステムから構成を復元

Solaris ボリュームマネージャ構成を元のシステムから別のシステムに復元することができます。たとえば、6つのディスクからなる外部 Multipack と Solaris ボリュームマネージャ構成のシステムがあるとします。これらのディスクには、少なくとも1つの状態データベースの複製があります。システムの障害時には、Multipack を異なるシステムに接続し、ローカルディスクから完全な構成を復元することができます。

注 - Solaris ボリュームマネージャ構成の復元先となるシステムには、Solaris ボリュームマネージャ構成がすでに存在してはなりません。存在していると、既存の論理ボリュームを、新しい論理ボリュームで置き換え、システムを壊してしまうおそれがあります。

注 - この処理は、ローカルディスクセットからボリュームを復元する場合にのみ有効です。

構成の復元

▼ 構成を復元するには

1. **Solaris** ボリュームマネージャ構成が含まれている 1 つまたは複数のディスクを、**Solaris** ボリュームマネージャ構成が存在していないシステムに追加します。
2. 再起動してシステムを再構成し、新たに追加したディスクをシステムが認識できるようにします。

```
# reboot -- -r
```

3. 状態データベースの複製が含まれている (新たに追加したディスク上の) スライスのメジャー/マイナー番号を特定します。
ls -lL 出力のグループ名と日付の間にある 2 つの数字がこのスライスのメジャー/マイナー番号です。

```
# ls -lL /dev/dsk/c1t9d0s7  
brw-r----- 1 root sys 32, 71 Dec 5 10:05 /dev/dsk/c1t9d0s7
```

4. 必要であれば、/etc/name_to_major 内のメジャー番号を調べ、そのメジャー番号に対応するメジャー名を特定します。

```
# grep " 32" /etc/name_to_major  
sd 32
```

5. 2つのコマンドを使って /kernel/drv/md.conf ファイルを更新します。最初のコマンドでは、有効な状態データベースの複製が新しいディスクのどこにあるかを Solaris ボリュームマネージャに知らせます。次のコマンドでは、新しい複製を優先して使用し、矛盾するデバイス ID がシステム上にあってもそれを無視するようにします。

mddb bootlist1 で始まる下記の行の sd を、前の手順で特定したメジャー名で置き換えます。また、同じ行の 71 を、手順3で特定したマイナー番号で置き換えます。

```
#pragma ident    "@(#)md.conf    2.1    00/07/07 SMI"

#
# Copyright (c) 1992-1999 by Sun Microsystems, Inc.
# All rights reserved.
#
name="md" parent="pseudo" nmd=128 md_nsets=4;
#
#pragma ident    "@(#)md.conf    2.1    00/07/07 SMI"
#
# Copyright (c) 1992-1999 by Sun Microsystems, Inc.
# All rights reserved.
#
name="md" parent="pseudo" nmd=128 md_nsets=4;
# Begin MDD database info (do not edit)
mddb bootlist1="sd:71:16:id0";
md_devid_destroy=1;# End MDD database info (do not edit)
```

6. システムを再起動して、Solaris ボリュームマネージャに構成を再ロードさせます。次のようなメッセージがコンソールに表示されます。

```
volume management starting.
Dec  5 10:11:53 lexicon metadevadm: Disk movement detected
Dec  5 10:11:53 lexicon metadevadm: Updating device names in
Solaris Volume Manager
The system is ready.
```

7. metadb と metastat コマンドを使って構成を確認します。

```
# metadb
      flags          first blk      block count
a m p  luo          16              8192      /dev/dsk/c1t9d0s7
a      luo          16              8192      /dev/dsk/c1t10d0s7
a      luo          16              8192      /dev/dsk/c1t11d0s7
a      luo          16              8192      /dev/dsk/c1t12d0s7
a      luo          16              8192      /dev/dsk/c1t13d0s7

# metastat
d12: RAID
State: Okay
Interlace: 32 blocks
Size: 125685 blocks
Original device:
Size: 128576 blocks
Device          Start Block  Dbase State          Reloc  Hot Spare
c1t11d0s3       330         No  Okay             Yes
c1t12d0s3       330         No  Okay             Yes
```

```

        c1t13d0s3                330      No      Okay      Yes

d20: Soft Partition
  Device: d10
  State: Okay
  Size: 8192 blocks
    Extent          Start Block          Block count
      0                3592                8192

d21: Soft Partition
  Device: d10
  State: Okay
  Size: 8192 blocks
    Extent          Start Block          Block count
      0                11785                8192

d22: Soft Partition
  Device: d10
  State: Okay
  Size: 8192 blocks
    Extent          Start Block          Block count
      0                19978                8192

d10: Mirror
  Submirror 0: d0
    State: Okay
  Submirror 1: d1
    State: Okay
  Pass: 1
  Read option: roundrobin (default)
  Write option: parallel (default)
  Size: 82593 blocks

d0: Submirror of d10
  State: Okay
  Size: 118503 blocks
  Stripe 0: (interlace: 32 blocks)
    Device          Start Block  Dbase State          Reloc  Hot Spare
    c1t9d0s0         0           No   Okay             Yes
    c1t10d0s0       3591        No   Okay             Yes

d1: Submirror of d10
  State: Okay
  Size: 82593 blocks
  Stripe 0: (interlace: 32 blocks)
    Device          Start Block  Dbase State          Reloc  Hot Spare
    c1t9d0s1         0           No   Okay             Yes
    c1t10d0s1         0           No   Okay             Yes

Device Relocation Information:
Device      Reloc      Device ID
c1t9d0      Yes       id1,sd@SSEAGATE_ST39103LCSUN9.0GLS3487980000U00907AZ
c1t10d0     Yes       id1,sd@SSEAGATE_ST39103LCSUN9.0GLS3397070000W0090A8Q

```

```

c1t11d0      Yes      id1,sd@SSEAGATE_ST39103LCSUN9.0GLS3449660000U00904NZ
c1t12d0      Yes      id1,sd@SSEAGATE_ST39103LCSUN9.0GLS32655400007010H04J
c1t13d0      Yes      id1,sd@SSEAGATE_ST39103LCSUN9.0GLS3461190000701001T0
#
# metadb
      flags          first blk      block count
a m p luo      16          8192          /dev/dsk/c1t9d0s7
a          luo      16          8192          /dev/dsk/c1t10d0s7
a          luo      16          8192          /dev/dsk/c1t11d0s7
a          luo      16          8192          /dev/dsk/c1t12d0s7
a          luo      16          8192          /dev/dsk/c1t13d0s7
# metastat
d12: RAID
      State: Okay
      Interlace: 32 blocks
      Size: 125685 blocks
Original device:
      Size: 128576 blocks
      Device          Start Block  Dbase State          Reloc  Hot Spare
c1t11d0s3          330         No  Okay          Yes
c1t12d0s3          330         No  Okay          Yes
c1t13d0s3          330         No  Okay          Yes

d20: Soft Partition
      Device: d10
      State: Okay
      Size: 8192 blocks
      Extent          Start Block          Block count
      0              3592              8192

d21: Soft Partition
      Device: d10
      State: Okay
      Size: 8192 blocks
      Extent          Start Block          Block count
      0              11785             8192

d22: Soft Partition
      Device: d10
      State: Okay
      Size: 8192 blocks
      Extent          Start Block          Block count
      0              19978             8192

d10: Mirror
      Submirror 0: d0
      State: Okay
      Submirror 1: d1
      State: Okay
      Pass: 1
      Read option: roundrobin (default)
      Write option: parallel (default)
      Size: 82593 blocks

d0: Submirror of d10

```

```

State: Okay
Size: 118503 blocks
Stripe 0: (interlace: 32 blocks)
  Device          Start Block  Dbase State      Reloc  Hot Spare
  c1t9d0s0        0           No   Okay         Yes
  c1t10d0s0       3591        No   Okay         Yes

```

```

d1: Submirror of d10
State: Okay
Size: 82593 blocks
Stripe 0: (interlace: 32 blocks)
  Device          Start Block  Dbase State      Reloc  Hot Spare
  c1t9d0s1        0           No   Okay         Yes
  c1t10d0s1       0           No   Okay         Yes

```

Device Relocation Information:

```

Device          Reloc  Device ID
c1t9d0          Yes   id1,sd@SSEAGATE_ST39103LCSUN9.0GLS3487980000U00907AZ1
c1t10d0         Yes   id1,sd@SSEAGATE_ST39103LCSUN9.0GLS3397070000W0090A8Q
c1t11d0         Yes   id1,sd@SSEAGATE_ST39103LCSUN9.0GLS3449660000U00904NZ
c1t12d0         Yes   id1,sd@SSEAGATE_ST39103LCSUN9.0GLS32655400007010H04J
c1t13d0         Yes   id1,sd@SSEAGATE_ST39103LCSUN9.0GLS3461190000701001T0

```

metastat -p

```

d12 -r c1t11d0s3 c1t12d0s3 c1t13d0s3 -k -i 32b
d20 -p d10 -o 3592 -b 8192
d21 -p d10 -o 11785 -b 8192
d22 -p d10 -o 19978 -b 8192
d10 -m d0 d1 1
d0 1 2 c1t9d0s0 c1t10d0s0 -i 32b
d1 1 2 c1t9d0s1 c1t10d0s1 -i 32b
#

```


Solaris ボリュームマネージャの重要なファイル

この付録では、Solaris ボリュームマネージャのファイルについて説明します。これらのファイルは、参照目的で使用してください。この章の内容は次のとおりです。

- 295 ページの「システムファイルと始動ファイル」
- 297 ページの「手動で設定するファイル」

システムファイルと始動ファイル

この節では、Solaris ボリュームマネージャが正常に動作するのに必要なファイルについて説明します。いくつかの特別な構成変更を行う場合を除き、これらのファイルを使用したり、変更したりする必要はありません。

- `/etc/lvm/mddb.cf`



注意 – このファイルを編集しないでください。このファイルを変更すると、Solaris ボリュームマネージャ構成を破壊することがあります。

`/etc/lvm/mddb.cf` ファイルには、状態データベースの複製の格納場所が記録されています。状態データベースの複製の格納場所が変わると、Solaris ボリュームマネージャは、すべての状態データベースの格納場所が記録されている `mddb.cf` ファイルにエントリを作成します。詳細は、`mddb.cf` (4) のマニュアルページを参照してください。

- `/etc/lvm/md.cf`

`/etc/lvm/md.cf` ファイルには、自動的に生成された、デフォルト (名前のない、またはローカルの) ディスクセットの構成情報が格納されています。Solaris ボリュームマネージャ構成が変更されると、Solaris ボリュームマネージャは `md.cf` ファイルを自動的に更新します (使用中のホットスペアの情報を除く)。詳細は、

md.cf (4) のマニュアルページを参照してください。



注意 – このファイルを編集しないでください。このファイルを変更すると、Solaris ボリュームマネージャ構成が破壊されたり、復元できなくなることがあります。

状態データベースに保持されている情報が失われた場合でも、その後でボリュームが変更されたり、作成されたりしていなければ、md.cf ファイルを使ってこの構成を復元できます。詳細は、238 ページの「構成ファイルを使って Solaris ボリュームマネージャを初期化するには」を参照してください。

- /kernel/drv/md.conf

md.conf 構成ファイルは、Solaris ボリュームマネージャによって起動時に読み取られます。ユーザーは、このファイルの 2 つのフィールド `nmd` と `md_nsets` を編集できます。前者のフィールドには、この構成でサポートされているボリューム (メタデバイス) の数が、後者のフィールドにはディスクセットの数がそれぞれ指定されています。`nmd` のデフォルト値は 128 で、最大数は 8192 です。`md_nsets` のデフォルト値は 4 で、最大数は 32 です。`md_nsets` にはデフォルト (名前のない、またはローカルの) ディスクセットが 1 つ含まれているため、名前の付いたディスクセットの合計数は常に `md_nsets` 値から 1 を引いた数です。

注 – `nmd` と `md_nsets` には、できるだけ小さい値を指定してください。これは、デバイスを実際に作成するかどうかには関係なく、`nmd` と `md_nsets` の値によって決まるデバイス数のメモリ構造が自動的に作成されるためです。最適な性能を維持するためには、実際に使用するボリュームの数よりもわずかに大きい値を `nmd` と `md_nsets` に指定するようにします。

- /etc/rcS.d/S35svm.init

このファイルは、システムの起動時に Solaris ボリュームマネージャを構成、および起動するために使用されます。また、管理者は、このファイルを使ってデーモンの起動や停止を制御できます。

- /etc/rc2.d/S95svm.sync

このファイルは、システムの起動時に Solaris ボリュームマネージャ構成をチェックし、必要であればミラーの再同期処理を開始し、アクティブな監視デーモンを起動します。(詳細は、`mdmonitord(1M)` のマニュアルページを参照してください。)

手動で設定するファイル

md.tab ファイルの概要

/etc/lvm/md.tab ファイルには、Solaris ボリュームマネージャの構成情報が格納されています。この情報を使えば、Solaris ボリュームマネージャ構成を再構築することができます。Solaris ボリュームマネージャは、このファイルをコマンド行ユーティリティ `metainit`、`metadb`、`metahs` への入力として使用することによって構成を再構築できます。このファイルには、通常、ボリュームやディスクセット、ホットスペア集合のエントリが含まれています。このファイルを作成する手順 (`metastat -p > /etc/lvm/md.tab` を使用) については、238 ページの「構成ファイルを作成するには」を参照してください。

注 - /etc/lvm/md.tab ファイルの構成情報と、実際に使用中のボリュームやホットスペア、状態データベースの複製が異なる場合があります。このファイルは、意図する構成を手動で保存するためのものです。Solaris ボリュームマネージャ構成を変更したら、このファイルを作成し直し、バックアップコピーを保管しておいてください。

このファイルを作成または更新したら、`metainit` や `metahs`、`metadb` コマンドを使って、このファイルに指定したボリュームや、ホットスペア集合、状態データベースの複製をアクティブにすることができます。

/etc/lvm/md.tab ファイルの各行には、1 つの ボリュームの完全な構成エントリの情報を、`metainit`、`metadb`、`metahs` コマンドの構文に基づいて指定します。

その上で `metainit` コマンドに `-a` オプションを指定すれば、/etc/lvm/md.tab ファイルに指定されているすべてのボリュームをアクティブにできます。あるいは、このファイルの特定のエントリに対応するボリュームを指定すれば、そのボリュームをアクティブにできます。

注 - Solaris ボリュームマネージャが /etc/lvm/md.tab ファイルに構成情報を書き込んだり、格納したりすることはありません。Solaris ボリュームマネージャコンポーネントを再作成するためには、ユーザーがこのファイルを編集し、`metainit` や、`metahs`、`metadb` コマンドを実行する必要があります。

詳細は、`md.tab(4)` のマニュアルページを参照してください。

付録 B

Solaris ボリュームマネージャのコマンド行リファレンス

この付録では、Solaris ボリュームマネージャのコマンド行インタフェースで使用できるコマンドの要約を示します。

コマンド行リファレンス

次の表に、Solaris ボリュームマネージャの管理に使用するコマンドを示します。詳細は、各コマンドのマニュアルページを参照してください。

表 B-1 Solaris ボリュームマネージャのコマンド

Solaris ボリュームマネージャのコマンド	説明	マニュアルページ
growfs	UFS ファイルシステムを安全に拡張します。	growfs (1M)
metaclear	アクティブなボリュームやホットスペア集合を削除します。	metaclear (1M)
metadb	状態データベースの複製を作成および削除します。	metadb (1M)
metadetach	RAID 1 (ミラー) ボリュームからボリュームを、またはトランザクションボリュームからロギングデバイスを切断します。	metadetach (1M)
metadevadm	デバイス ID 構成をチェックします。	metadevadm (1M)
metahs	ホットスペアとホットスペア集合を管理します。	metahs (1M)
metainit	ボリュームを作成します。	metainit (1M)

表 B-1 Solaris ボリュームマネージャのコマンド (続き)

Solaris ボリュームマネージャのコマンド	説明	マニュアルページ
metaoffline	サブミラーをオフラインにします。	metaoffline (1M)
metaonline	サブミラーをオンラインにします。	metaonline (1M)
metaparam	ボリュームパラメータを変更します。	metaparam (1M)
metarecover	ソフトパーティションの構成情報を復元します。	metarecover (1M)
metarename	ボリューム名を変更または交換します。	metarename (1M)
metareplace	サブミラーおよび RAID 5 ボリュームのコンポーネントを置き換えます。	metareplace (1M)
metaroot	ルート (/) をミラー化するためのシステムファイルを設定します。	metaroot (1M)
metaset	ディスクセットを管理します。	metaset (1M)
metastat	ボリュームまたはホットスベア集合の状態を表示します。	metastat (1M)
metasync	再起動時にボリューム間の再同期をとります。	metasync (1M)
metattach	RAID 0 または RAID 1 ボリュームにコンポーネントを、あるいはトランザクションボリュームにログデバイスを接続します。	metattach (1M)

Solaris ボリュームマネージャの CIM/WBEM API

Solaris ボリュームマネージャの管理

Solaris ボリュームマネージャの CIM/WBEM アプリケーションプログラミングインタフェース (API) は、標準に準拠したオープンなプログラムインタフェースを提供し、Solaris ボリュームマネージャの管理や構成を可能にします。この API は、Distributed Management Task Force (DMTF) の Common Information Model (CIM) に基づいています。DMTF については、<http://www.dmtf.org> をご覧ください。

CIM は、「スキーマ」と呼ばれるデータモデルを定義します。このスキーマには、次のものが規定されています。

- SVM デバイスの属性と SVM デバイスに対する操作
- さまざまな SVM デバイス間の関係
- SVM デバイスと、オペレーティングシステムの機能 (ファイルシステムなど) との関係

このモデルは、Solaris Web Based Enterprise Management (WBEM) SDK を通じて使用されます。WBEM SDK は、CIM で規定されているシステム管理機能へのアクセスを可能にする Java™ テクノロジーに基づく API セットです。

CIM/WBEM SDK の詳細は、『Solaris WBEM SDK 開発ガイド』を参照してください。

索引

数字・記号

3 面ミラー, 95

C

cron コマンド, 267

D

DiskSuite ツール

グラフィカルインタフェースを参照

E

/etc/lvm/md.cf ファイル, 295

/etc/lvm/mddb.cf ファイル, 295

/etc/rc2.d/S95lvm.sync ファイル, 296

/etc/rcS.d/S35lvm.init ファイル, 296

/etc/vfstab ファイル, 124, 187, 203

不適切なエントリの修正, 276

F

fmthard コマンド, 281, 283

format コマンド, 281, 283

fsck コマンド, 204, 205

G

growfs コマンド, 44, 45, 243, 244, 299

GUI, サンプル, 40

K

/kernel/drv/md.conf ファイル, 240, 296

L

lockfs コマンド, 127, 205

M

md.cf file, Solaris ボリュームマネージャ構成の回復, 239

md.cf ファイル, 296

md.tab ファイル, 239

概要, 297

metaclear コマンド, 86, 122, 123, 299

metadb コマンド, 65, 67, 272, 282, 299

metadetach コマンド, 112, 122, 299

metahs コマンド, 174, 299

metainit コマンド, 185, 195, 196, 199, 200, 202, 203, 239, 299

metaoffline コマンド, 113, 300

metaonline コマンド, 300

metaparam コマンド, 118, 167, 168, 169, 300

metarename コマンド, 237, 300

metareplace コマンド, 114, 121, 153, 155, 282, 300
metaroot コマンド, 300
metaset コマンド, 219, 224, 300
metastat コマンド, 117, 149, 193, 300
metasync コマンド, 300
metattach コマンド, 85, 104, 111, 119, 152, 198, 240, 288, 300
mirror, オプションの変更, 118

N

newfs コマンド, 205

R

RAID, Solaris ボリュームマネージャでサポートされるレベル, 30
RAID 5 のパリティ計算, 144
RAID 5 ボリューム
4つのスライスの場合, 140
エラー状態の説明, 246
置き換えと有効化の概要, 244
拡張, 152
コンポーネントを置き換え、有効にするための情報, 247
作成, 148
障害があったスライスを有効化, 153
障害があるスライスの交換, 155
スライスの初期化, 140
スライスの同期化, 140
RAID 0 ボリューム
定義, 69, 70
RAID 5 ボリューム
定義, 30, 43
デバイスを拡張する場合, 141
と飛び越し値, 143
パリティ情報, 139, 142
「保守 (Maintenance)」状態と「最後にエラー (Last Erred)」状態, 246
RAID 0 ボリューム
用途, 70
raw ボリューム, 81, 82, 103, 149

S

SCSI ディスク
交換, 271, 274
Solaris Volume Manager
Solaris ボリュームマネージャを参照
Solaris ボリュームマネージャ
構成の回復, 239
構成の指針, 48
Solaris ボリュームマネージャインタフェース
Solaris 管理コンソール, 38
コマンド行, 39
サンプル GUI, 40
Solaris ボリュームマネージャの要素, 概要, 41
swap
ミラー化, 107
ミラー化解除, 125

U

UFS ログイン, 定義, 175
/usr
ミラー化, 106
ミラー化解除, 124
ログイン, 187

V

/var/adm/messages ファイル, 245, 272

X

x86

IAを参照

い

インタフェース

Solaris ボリュームマネージャインタフェースを参照

え

エラー、スクリプトを使ってチェック, 259

お

オンラインバックアップ, 126

か

解放、ディスクセットの, 224, 227, 228

書き込みポリシーの概要, 93

拡張ディスク

グラフィカルインタフェースを参照

き

起動障害, 275

起動、シングルユーザーモードとして, 96

起動デバイス、障害からの回復, 278

共有ディスクセット, 48

く

グラフィカルインタフェース、概要, 38

こ

構成計画

概要, 31

構成計画 (続き)

指針, 31

バランス, 33

さ

再同期

最適化された, 94

全面的な, 94

部分的な, 94

サブミラー, 88

オフラインでの操作, 88

オフラインまたはオンラインにする, 112

切り離し, 88

障害があったスライスを有効にする, 114

障害のあるスライスの交換, 121

全体を交換, 122

追加, 88

し

システムファイル, 295

取得、ディスクセットの, 226

順次入出力, 35

障害追跡、一般的な指針, 270

状態, 224

状態データベース

概念, 46, 56

損傷, 56

定義, 42, 46

複製数の不足を解決, 282

状態データベースの複製, 47

2 ディスク構成, 60

エラー, 60

同じスライス上に複数を作成, 58

基本的な機能, 56

最小数, 58

追加, 64

定義, 47

場所, 47, 58, 59

複製数の不足を解決, 282

用途, 55

より大きい複製の追加, 67

シンプルなボリューム、定義, 43

シンプルボリューム
RAID 0 ボリュームを参照

す

ストライプ
3つのスライスの場合, 71
拡張, 85
再作成のための情報, 77
削除, 86
作成, 81
作成するための情報, 77
定義, 70
ストライプ方式の連結
3つのストライプ方式を連結する場合, 74
定義, 74
ストライプ方式ボリューム
ストライプを参照
ストライプ連結, 削除, 86
スライス
RAID 5 ボリュームに追加, 152
拡張, 84

せ

性能に関する一般的な指針, 33

そ

ソフトパーティション
拡大, 136
拡張, 136
構成データを復元, 286
削除, 137
作成, 130, 134
指針, 130
状態のチェック, 135
場所, 130
ソフトパーティションf, 除去, 137

た

代替起動デバイス, IA, 110
代替起動パス, 105

多数決アルゴリズム, 56

つ

追加, ホットスペアの, 166

て

ディスクセット, 207
2つの共有ディスクセットの場合, 211
/etc/vfstab ファイルでの使用はできない, 208
Solstice HA, 208
主な用途, 208
解放, 214, 224, 227, 228
管理, 213, 214
作成, 219
取得, 214, 226
取得動作, 214
取得の種類, 214
状態のチェック, 223, 224, 225, 228
状態を確認, 227
所有者を表示, 224
定義, 42, 48
ディスクを追加, 208
デフォルトの数を増やす, 241
ドライブを追加, 219, 221
複製の配置, 208
別のホストを追加, 221, 222
ボリュームやホットスペア集合との関係, 208
用途, 207

と

飛び越し値, 指定, 81
トランザクションボリューム
metarename を使って削除, 201
metarename を使って作成, 196, 199
エラーからの回復, 206, 285
拡張, 198
指針, 179
状態, 194
定義, 43
と /etc/vfstab ファイル, 186

トランザクションボリューム (続き)
マウント解除できないファイルシステム用に
作成, 187
ミラーを使用する場合, 177
ミラーを使って作成, 188
用途, 177
ロギングするファイルシステムの選択, 180
ログデバイスを共有する場合, 178

に
入出力, 34

は
パス番号
定義, 94
と読み取り専用ミラー, 94

ふ
ファイルシステム
拡張, 242
拡張方法の概要, 44, 45
指針, 49
パニック, 286
ミラー化解除, 125
連結を作成して拡張, 83
フェイルオーバー構成, 48, 207
複製, 47

ほ
ホットスペア, 158
概念, 158
ホットスペア集合における置き換え, 172
ホットスペア集合に追加, 166
有効化, 174
ホットスペア集合, 47
概念, 157, 159
管理, 160
関連付け, 168
関連付けの変更, 169
基本的な機能, 47

ホットスペア集合 (続き)
作成, 165
状態, 170
定義, 42, 47
ミラーの場合, 159
ボリューム
概念, 42
仮想ディスク, 38
使用, 43
状態のチェック, 193
タイプ, 43
定義, 42
ディスク領域の拡張, 44
デフォルトの数, 240
デフォルトの数を増やす, 240
名前の交換, 236, 237
名前の変更, 238
ファイルシステムコマンドの使用, 44
命名規則, 45, 211
ボリューム名の交換, 237
ボリューム名のヒント, 46
ボリューム名の変更, 235

ま
マスターデバイス, 定義, 177

み
ミラー, 87
2つのサブミラーの場合, 88
2面ミラー, 102
3面ミラー, 95
エラー状態の説明, 246
オプション, 92
およびディスクのジオメトリ, 96
拡張, 119
切り離しとオフラインの違い, 95
コンポーネントの置き換えや有効化の概
要, 145
コンポーネントを置き換え、有効にするため
の情報, 247
再同期, 93, 94
作成するための情報, 95
サブミラー数の最大, 90
サブミラーの接続, 111

ミラー (続き)

- 指針, 91
- 状態の出力例, 117
- スライスの置き換えと有効化の概要, 244
- 定義, 43
- とオンラインバックアップ, 126
- 「保守 (Maintenance)」状態と「最後にエラー (Last Erred)」状態, 246

ミラー化

- root (/)、/usr、swap, 107
- 可用性に伴う考慮事項, 90
- マウント解除可能なファイルシステム, 105
- 未使用のスライス, 101
- 読み取り/書き込み性能, 32

め

- メタデバイス
- ボリュームを参照

ゆ

- 有効化、RAID 5 ボリュームのスライスの, 153
- 有効化、ホットスベアの, 173
- 有効にする、サブミラーのスライスを, 114

よ

- 読み取りポリシーの概要, 92

ら

- ランダム入出力, 34

る

- ルート (/)
- ミラー化, 106
- ミラー化解除, 124

れ

連結

- 3つのスライスの場合, 73
 - UFS ファイルシステムの拡張, 72
 - 拡張, 85
 - 再作成のための情報, 77
 - 削除, 86
 - 作成, 82
 - 作成するための情報, 77
 - 定義, 72
 - 用途, 72
- 連結方式ボリューム
- 連結を参照

ろ

- ローカルディスクセット, 208
- ロギングデバイス, 「ハードウェアエラー (Hard Error)」状態, 286
- ログデバイス
- エラーからの回復, 206
- 共有, 177, 180, 203
- 共有時の問題, 204
- 定義, 177
- 必要な領域, 180