NFS の管理

Sun Microsystems, Inc.
901 San Antonio Road
Palo Alto, CA 94303
U.S.A. 650–960–1300

Part No: 805-5824-10
1998 年 11 月

原典: NFS Transition Guide

Part No: 805–3737–10

Revision A

目次

はじめに **ix**

パート**I** 概要

1. Solaris NFS の環境 **3**

NFS サーバとクライアント 3

NFS ファイルシステム 4

NFS 環境 4

NFS バージョン 2 5

NFS バージョン 3 6

NFS ACL サポート 6

TCP への依存 6

ネットワークロックマネージャ 7

NFS 大型ファイルのサポート 7

NFS クライアントの障害時回避機能 7

Kerberos による NFS 環境のサポート 7

WebNFS のサポート 8

RPCSEC_GSS セキュリティ方式 8

Solaris 7 の NFS に対する拡張機能 8

autofs について 9

autofs の特長 9

パート**II　NFS** サービス全般について

2.　NFS サーバの設定　**13**

ファイルシステムの自動共有　14

▼ ファイルシステム自動共有を設定する方法　14

ファイルシステムのマウント　15

▼ ブート時のマウント　15

▼ コマンド行からマウントする方法　16

▼ オートマウンタを使ってマウントする方法　17

NFS サービスの設定　17

▼ NFS サービスを起動する方法　17

▼ NFS サービスを停止する方法　18

▼ NFS サーバ上の大型ファイルを無効にする方法　18

▼ クライアント側障害時回避機能の使用方法　19

▼ 1 つのクライアントに対するマウントアクセスを無効にする方法　19

▼ ファイアウォールを越えて NFS ファイルシステムをマウントする方法　20

▼ NFS URL を使用して NFS ファイルシステムをマウントする方法　21

Secure NFS システムの管理　21

▼ DH 認証を使って Secure NFS 環境を設定する方法　21

▼ セキュリティ保護された NFS 環境を KERB 認証を使用して設定する方法　24

WebNFS の管理タスク　25

WebNFS アクセスの計画　25

▼ WebNFS アクセスを有効にする方法　26

▼ NFS URL を使ったブラウズ　27

▼ ファイアウォール経由で WebNFS を使用する方法　28

NFS における障害追跡の概要　28

NFS の障害追跡手順　29

▼ NFS クライアントの接続性のチェック　29

▼ NFS サーバをリモートでチェックする方法　31

▼ サーバで NFS サービスを確認する方法 33
▼ NFS サービスの再起動 34
▼ rpcbind のワームスタート 35
▼ NFS ファイルサービスを提供しているホストを識別する方法 36
▼ mount コマンドに使用されたオプションを確認する方法 36
NFS のエラーメッセージ 37
3. **NFS** リファレンス **43**
NFS ファイル 43
NFS デーモン 44
lockd 45
mountd 46
nfsd 46
statd 46
NFS コマンド 47
clear_locks 48
mount 48
umount 52
mountall 53
umountall 54
share 54
unshare 60
shareall 61
unshareall 61
showmount 61
setmnt 62
その他のコマンド 63
nfsstat 63
pstack 64

rpcinfo 65
snoop 67
truss 67
コマンドを組み合わせて使う 68
バージョン 2 とバージョン 3 のネゴシエーション 68
UDP と TCP のネゴシエーション 69
ファイル転送サイズのネゴシエーション 69
ファイルシステムのマウントの詳細 70
マウント時の –public オプションと NFS URL の意味 71
クライアント側障害時回避機能 71
大型ファイル 73
WebNFS サービスの動作方法 73
Web ブラウザと比較した場合の WebNFS の制約 75
Secure NFS システム 75
Secure RPC 76
パート**III** **autofs** の詳細
4. **autofs** の管理 **83**
autofs の設定 84
▼ オートマウンタを起動する方法 84
▼ オートマウンタを停止する方法 84
一般的な作業と手順 84
マップを含む管理作業 85
マップの変更 86
▼ マスタマップを変更する 86
▼ 間接マップを変更する 87
▼ 直接マップを変更する 87
マウントポイント衝突の回避 88
NFS 以外のファイルシステムへのアクセス 88

▼ autofs を使って CD-ROM アプリケーションにアクセスする 88

▼ autofs を使って PC-DOS データのフロッピーディスクにアクセスする 89

CacheFS による NFS ファイルシステムへのアクセス 89

▼ CacheFS を使って NFS ファイルシステムにアクセスする 89

オートマウンタのカスタマイズ 90

▼ /home ディレクトリ構造を共通にする 90

▼ 複数のホームディレクトリファイルシステムを持つ /home を設定する 91

▼ プロジェクト関連ファイルの統合方法 (/ws ディレクトリ構造) 92

▼ さまざまなアーキテクチャによる共有名前空間へのアクセスを設定する方法 94

▼ 互換性のないクライアントオペレーティングシステムをサポートする 95

▼ 複数のサーバ間で共有ファイルを複写する 96

▼ セキュリティを適用する 96

▼ autofs で公共ファイルハンドルを使う方法 97

▼ autofs で NFS URL を使う方法 97

autofs の表示機能の無効化 97

▼ 単一の NFS クライアントに対して autofs の表示機能を完全に無効にする方法 98

▼ すべてのクライアントに対して autofs の表示機能を無効にする 98

▼ 1 台の NFS クライアントに対して autofs の表示機能を無効にする 99

autofs での問題発生時の対処 100

自動マウンタで生成されるエラーメッセージ 100

一般的なエラーメッセージ 102

autofs に関するその他のエラー 103

5. autofs について 105

autofs プログラム 105

automount 105

automountd 106

autofs マップ 106

マスタマップ　107

直接マップ　110

間接マップ　112

autofs のしくみ　114

autofs のネットワークナビゲート (マップ)　116

autofs のナビゲーションプロセス開始法 (マスタマップ)　116

autofs マウントプロセス　117

autofs がクライアント用の最も近い読み取り専用ファイルを選択する方法 (複数ロケーション)　119

マップエントリ内の変数　122

他のマップを参照するマップ　123

実行可能な autofs マップ　125

autofs のネットワークナビゲート法の変更 (マップの変更)　126

ネームサービスに対する autofs のデフォルトの動作　126

autofs リファレンス情報　127

メタキャラクタ　128

特殊文字　129

A.　NFS の調整機能　131

カーネルパラメータの値を設定する方法　136

索引　**139**

はじめに

本書は、NFS™ 分散ファイルシステムを正しく運用するために必要な管理作業について説明します。このファイルシステム共有製品を使用することにより、ファイルとディレクトリをネットワーク上の多数のコンピュータ間で共有することができます。

またこのマニュアルでは、autofs (以前はオートマウンタと呼んでいました) を設定することにより、NFS ファイルシステムを自動的にマウントしたり、アンマウントする方法についても説明します。

このマニュアルは、背景知識の説明と作業ごとの手順説明で構成されています。

対象読者

このマニュアルは、NFS システムの設定や管理を担当するシステム管理者を対象としています。このマニュアルの大半は経験を積んだシステム管理者を対象に書かれていますが、Solaris™ プラットフォームをはじめて扱うシステム管理者や、そのほかのユーザに役立つ情報も収録しています。

このマニュアルの構成

第 1 章 では、Solaris NFS の環境と autofs の概要を説明します。

第 2 章 では、NFS サーバの設定方法を説明します。読者はネームサービスとして NIS または NIS+ を使用しているものと仮定します。

第 3 章 では、NFS サービスのセキュリティ機能と、NFS セキュリティの設定と保守に関する基本手順について説明します。

第 4 章 では、NFS サービスを使用しているマシンで発生する問題について説明します。具体的には、NFS で発生する問題の追跡方法を説明します。また、基本的なことを説明する節と参照用の節があります。

第 5 章 では、autofs の設定と使用方法について説明します。基本的な説明、参照情報、および障害追跡の節があります。

付録 A では、NFS サービスの調整に使用する、いくつかのパラメータについて説明します。変更方法についても説明します。

関連マニュアル

このマニュアルで参照している関連マニュアルのリストを示します。

- 『*Solaris* ネーミングの管理』
- 『*Solaris* ネーミングの設定と構成』
- 『*Solaris* のシステム管理 *(*第 *1* 巻*)*』
- 『*Solaris* のシステム管理 *(*第 *2* 巻*)*』
- 『*TCP/IP* とデータ通信』

マニュアルの注文方法

SunDocs™ プログラムでは、米国 Sun Microsystems™, Inc. (以降、Sun™ とします) の 250 冊以上のマニュアルを扱っています。このプログラムを利用して、マニュアルのセットまたは個々のマニュアルをご注文いただけます。

マニュアルのリストと注文方法については、米国 SunExpress™, Inc. のインターネットホームページ `http://www.sun.com/sunexpress` にあるカタログセクションを参照してください。

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、画面上のコンピュータ出力、または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、画面上のコンピュータ出力とは区別して示します。	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 *filename* 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、または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、テキストがページ行幅を越える場合、バックスラッシュは継続を示します。	sun% **grep '^#define ** **XV_VERSION_STRING'**

ただし AnswerBook2™ では、ユーザーが入力する文字と画面上のコンピュータ出力は区別して表示されません。

コード例は次のように表示されます。

- C シェルプロンプト

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

■ Bourne シェルおよび Korn シェルのプロンプト

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

■ スーパーユーザーのプロンプト

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

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

| 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。
- 「x86」という用語は、一般に Intel 8086 ファミリに属するマイクロプロセッサを意味します。これには、Pentium、Pentium Pro の各プロセッサ、および AMD と Cyrix 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。このマニュアルでは、このプラットフォームのアーキテクチャ全体を指すときに「x86」という用語を使用し、製品名では「Intel 版」という表記で統一しています。

パート I 概要

パートI では、NFS の環境で提供されるサービスについて説明します。

- 3ページの「NFS サーバとクライアント」
- 4ページの「NFS ファイルシステム」
- 4ページの「NFS 環境」
- 9ページの「autofs について」

第 1 章

Solaris NFS の環境

この章では、NFS 環境の概要について説明します。具体的には、ネットワークの簡単な概要、NFS サービス、NFS システムの把握に必要なコンセプトを説明します。

- 3ページの「NFS サーバとクライアント」
- 4ページの「NFS ファイルシステム」
- 4ページの「NFS 環境」
- 9ページの「autofs について」

NFS サーバとクライアント

クライアントとサーバという用語は、コンピュータがファイルシステムを共有するときの役割を示すものです。ファイルシステムがあるコンピュータのディスク上に存在し、そのコンピュータがこのファイルシステムをネットワーク上のほかのコンピュータから使用できるようにしている場合、そのコンピュータをサーバと呼びます。そのファイルシステムにアクセスしているコンピュータをクライアントと呼びます。 NFS を使用することによって、どのコンピュータからもほかのコンピュータのファイルシステムにアクセスでき、それと同時に自分のファイルシステムへのアクセスも可能となります。ネットワーク上では 1 台のコンピュータがクライアントかサーバ、またはその両方の役割として動作することができます。

サーバは、ディスクレスクライアント、つまりローカルディスクがないコンピュータにファイルを提供することもあります。ディスクレスクライアントは、ファイル

の保存についてはすべてサーバに依存します。ディスクレスクライアントはクライアントにしかなれず、サーバになることは決してありません。

クライアントは、サーバの共有ファイルシステムをマウントすることによってサーバのファイルにアクセスします。クライアントがリモートファイルシステムをマウントしたとき、ファイルシステムがコピーされるのではありません。マウント処理では一連のリモートプロシージャコールによって、クライアントからサーバのディスク上にあるファイルシステムに透過的にアクセスできるようになります。マウントはローカルマウントのように行われるので、ユーザはファイルシステムがローカルにあるのと同じようにコマンドを入力します。

サーバのファイルシステムは、NFS オペレーションによって共有すると、クライアントからアクセスできるようになります。NFS ファイルシステムは、autofs を使用すると自動的にマウントできます。

NFS ファイルシステム

NFS サービスで共有できるオブジェクトは、ファイル階層の全体、またはその一部です。ファイルを 1 つだけ共有することもできます。すでに共有しているものと重複するファイル階層構造は共有できません。モデムやプリンタなどの周辺機器も共有できません。

多くの UNIX® システム環境で共有されるファイル階層構造は、1 つのファイルシステム、またはその一部です。しかし NFS サポートは複数のオペレーティングシステムにまたがって動作し、ファイルシステムという考え方は UNIX 以外の環境では通用しません。 したがって本書でファイルシステムという語を使う場合、NFS 環境でマウントし共有した、ファイルまたはファイル階層構造を指すことにします。

NFS 環境

NFS サービスとは、アーキテクチャが異なり、別のオペレーティングシステムで動作しているコンピュータが、ネットワークを通じてファイルシステムを共有できるようにするサービスのことです。NFS サポートは、MS-DOS から VMS オペレーティングシステムまで多くの、プラットフォームに実装されています。

NFS 環境は、異なるオペレーティングシステムで実現できます。アーキテクチャの仕様を定義するのではなく、ファイルシステムの抽象モデルを定義しているためです。それぞれのオペレーティングシステムでは、ファイルシステムセマンティクスに NFS 抽象モデルを適用します。これにより、書き込みや読み出しのようなファイルシステムオペレーションが、ローカルファイルにアクセスするように機能することになります。

NFS サービスの利点を以下に挙げます。

- 複数のコンピュータで同一のファイルを使用するため、ネットワーク上の誰もが同じデータにアクセスできる。
- 各ユーザアプリケーションがローカルのディスク空間を占めるのではなく、複数のコンピュータでアプリケーションを共有するため、記憶領域を有効利用できる。
- すべてのユーザが同一セットのファイルを読み出すので、データの整合性と信頼性が向上する。
- ファイルシステムをユーザに透過的な形でマウントできる。
- リモートファイルに透過的にアクセスできる。
- 様々な環境をサポートする。
- システム管理の手間を省ける。

NFS サービスを使用すると、ファイルシステムの実際の場所をユーザとは無関係に決めることができます。ユーザは場所を気にすることなく、すべての適切なファイルにアクセスできるということです。NFS サービスでは、共通して使用するファイルのコピーをすべてのシステムに置くのではなく、コピーを 1 つのコンピュータのディスクに置き、他のシステムからネットワークを通じてアクセスできるようにします。NFS オペレーションでは、リモートファイルとローカルファイルの区別がありません。

NFS バージョン 2

バージョン 2 は、一般に広く使われた初めての NFS プロトコルです。バージョン 2 は、引き続き広範囲のプラットフォームで使用できます。Solaris 2.5 以前のリリースの SunOS™ では、NFS プロトコルのバージョン 2 が使用できます。

NFS バージョン 3

NFS バージョン 3 のプロトコルは、Solaris 2.5 に新機能を追加したものです。相互運用性とパフォーマンスを向上させるために、いくつかの変更が行われました。これらをすべて有効利用するには、NFS サーバとクライアントの両方でバージョン 3 プロトコルを使用する必要があります。

バージョン 3 では、サーバで非同期の書き込みが可能になります。サーバがクライアントの書き込み要求をメモリに保存するので、効率が向上しました。クライアントは、サーバが変更内容をディスクに反映させるのを待つ必要がないため、応答時間が短縮されます。サーバは要求をバッチ処理することもできるので、サーバ上の応答時間も短縮されました。

NFS バージョン 3 では、どの操作でもローカルキャッシュに保存されているファイル属性が返されます。キャッシュの更新頻度が増えたため、ローカルキャッシュのデータを更新する操作を独立して行う必要性が少なくなります。したがってサーバに対する RPC コールの回数が減少し、パフォーマンスが向上します。

ファイルアクセス権の確認処理も改善されました。 バージョン 2 では、ユーザがアクセス権を持っていないリモートファイルをコピーしようとすると、「書き込みエラー」や「読み出しエラー」というメッセージが出力されました。バージョン 3 では、ファイルを開く前に権利がチェックされるので、「オープンエラー」がというメッセージが出力されます。

NFS バージョン 3 では、8 キロバイトの転送サイズ制限が解除されました。クライアントとサーバは、バージョン 2 で課せられていた 8 キロバイト制限を受けず、サポートできる転送サイズならばどのようなものでも処理します。Solaris 2.5 では、転送サイズが 32 キロバイトにデフォルトで設定されています。

NFS ACL サポート

Solaris 2.5 には、アクセス制御リスト (ACL) サポートが追加されました。ACL では、ファイルアクセス権を通常の UNIX よりも正確に設定します。この追加機能では効率は改善されませんんが、ファイルへのアクセスがより厳密に制限されるので、セキュリティが向上します。

TCP への依存

Solaris 2.5 では、NFS プロトコルのデフォルトのトランスポートプロトコルが TCP に変わりました。このため、低速のネットワークおよび広域ネットワークにおける

パフォーマンスが改善されます。TCP には、トラフィック抑制機能とエラー回復機能があります。TCP を利用した NFS は、バージョン 2 でもバージョン 3 でも動作します。2.5 より前のバージョンでは、NFS のデフォルトプロトコルは UDP (User Datagram Protocol) でした。

ネットワークロックマネージャ

Solaris 2.5 には、ネットワークロックマネージャの改良版も含まれています。このため NFS ファイルに対して UNIX のレコードロックと PC のファイル共有が使用できます。NFS ファイルに対するロック機構の信頼性が向上したため、ロックを使用する `ksh` や `mail` などのコマンドがハングする可能性が少なくなります。

NFS 大型ファイルのサポート

Solaris 2.6 の NFS バージョン 3 プロトコルでは、2 ギガバイトを超えるサイズのファイルも正しく処理できます。NFS バージョン 2 プロトコル、および Solaris 2.5 に実装されているバージョン 3 プロトコルでは 2 ギガバイトを超えるサイズのファイルは処理できませんでした。

NFS クライアントの障害時回避機能

Solaris 2.6 では、読み取り専用ファイルシステムの動的障害時回避機能が追加されました。これによって、マニュアルページ、AnswerBook™、共有バイナリなどのあらかじめ複製されている読み取り専用リソースを高度に利用できます。障害時回避機能は、ファイルシステムがマウントされた後ならばいつでも実行可能です。手動マウントでは、今までのリリースのオートマウンタのように複数の複製をリストできるようになりました。オートマウンタは、障害時回避の際にファイルシステムが再マウントされるまで待つ必要がなくなったこと以外は変更されていません。

Kerberos による NFS 環境のサポート

Solaris 2.0 では、Kerberos V4 クライアントがサポートされていました。Solaris 2.6 では、`mount` コマンドと `share` コマンドは、Kerberos V5 認証を使用した NFS マウントをサポートするようになりました。また、`share` コマンドはクライアントごとに異なる複数の認証方法を指定できるようになりました。

WebNFS のサポート

Solaris 2.6 には、NFS プロトコルの拡張機能を使用することによってインターネット上のファイルシステムにファイアウォール経由でアクセスできるようにする機能もあります。この WebNFS™ プロトコルを使ってインターネットにアクセスする利点の 1 つは、非常に信頼性の高い NFS バージョン 3 とバージョン 2 プロトコルの拡張機能としてサービスが構築されるということです。今後、この新しいファイルシステムのアクセスプロトコルを使ったアプリケーションがいくつも作成されるはずです。また NFS サーバでは、負荷が大きい状態のときに HTTP (HyperText Transfer Protocol) から Web サーバへのアクセスよりも高いスループットを確保できます。そのため、ファイルを取得するための時間が短縮されます。さらに、WebNFS ではそうしたファイルを共有しても匿名 `ftp` サイトを管理するオーバーヘッドが生じません。

RPCSEC_GSS セキュリティ方式

Solaris 7 では、新しいセキュリティ方式である RPCSEC_GSS がサポートされています。この方式では、標準的な GSS-API インタフェースを使用して、認証、一貫性、機密性を実現し、複数のセキュリティ機構をサポートしています。現在、これらの新しいセキュリティ方法を使用する機構は、Solaris には組み込まれていません。

Solaris 7 の NFS に対する拡張機能

Solaris 7 では、`mount` コマンドと `automountd` コマンドが拡張され、マウント要求で MOUNT プロトコルの代わりに公開ファイルハンドルも使用できるようになりました。これは、WebNFS サービスで使用されているのと同じアクセス方法です。公開ファイルハンドルを使用すると、ファイアウォールを越えたマウントが可能です。さらに、サーバとクライアント間のトランザクションが少なくて済むため、マウントにかかる時間が短縮されます。

この機能拡張で、標準のパス名の代わりに NFS URL を使用することもできるようになりました。また、`mount` コマンドとオートマウンタのマップに `–public` オプションを指定すると、必ず公開ファイルハンドルを使用するようになります。

autofs について

NFS サービスを使って共有されるファイルシステムは、「自動マウント」と呼ばれる方法によってマウントできます。クライアント側のサービスである autofs は、自動マウントを実現するファイルシステム構造です。autofs のファイルシステムは、automount で作成されます。automount は、システムを起動すると自動的に実行されます。automountd という常駐型の automount デーモンが、必要に応じてリモートディレクトリのマウントとアンマウントを行います。

automountd を実行しているクライアントコンピュータ上のユーザがリモートのファイルまたはディレクトリにアクセスしようとすると、そのファイルまたはディレクトリが所属するファイルシステムがこのデーモンによってマウントされます。このリモートファイルシステムは、必要な間はマウントされたままです。リモートファイルシステムが一定時間アクセスされないと、自動的にアンマウントされます。

ブート時にはマウントする必要はなく、ユーザはディレクトリをマウントするためにスーパーユーザのパスワードを知る必要はありません。ユーザが mount と umount コマンドを使用する必要もありません。autofs は、ユーザの介入なしに、必要に応じてファイルシステムをマウントしたり、アンマウントします。

automountd によって一部のファイル階層をマウントするということは、mount によってほかのファイル階層をマウントしないということではありません。ディスクレスコンピュータは、mount と /etc/vfstab ファイルを使って / (ルート)、/usr、/usr/kvm をマウントしなければなりません。

autofs サービスの詳細については、第 5 章 で説明します。

autofs の特長

autofs は、ローカルの名前空間に指定したファイルシステムで動作します。この情報は、NIS、NIS+、ローカルファイルに保存されます。

Solaris 2.6 には、完全にマルチスレッド化された automountd が含まれています。この拡張によって autofs はさらに信頼性が高まりました。また、複数のマウントを同時にサービスできるようになったため、サーバが使用できないときにサービスが停止することも避けられます。

この新しい automountd には、オンデマンドマウント機能もあります。今までのリリースでは、階層に含まれるすべてのファイルシステムがマウントされていました。

これからは、一番上のファイルシステムしかマウントされません。そのマウントポイントに関係するほかのファイルシステムは、必要に応じてマウントされます。

autofs サービスで、間接マップを表示できるようになりました。これによりユーザは、どのディレクトリがマウントできるかを確認するためにファイルシステムを実際に 1 つずつマウントする必要がなくなります。autofs マップに –nobrowse オプションが追加されたので、/net や /home などの大きなファイルが自動的に表示されることはありません。また、automount に対して –n を使うことによって、autofs の表示機能を各クライアントでオフにすることもできます。

パートII　NFS サービス全般について

パートII では、NFS サービスの大部分の機能とそれらの管理方法について説明します。パートIII では、autofs を取り上げます。

- 14ページの「ファイルシステムの自動共有」
- 15ページの「ファイルシステムのマウント」
- 17ページの「NFS サービスの設定」
- 21ページの「Secure NFS システムの管理」
- 25ページの「WebNFS の管理タスク」
- 29ページの「NFS の障害追跡手順」
- 37ページの「NFS のエラーメッセージ」
- 43ページの「NFS ファイル」
- 44ページの「NFS デーモン」
- 47ページの「NFS コマンド」

第 2 章

NFS サーバの設定

この章では、NFS 管理タスクの実行方法について説明します。具体的には、NFS サービスの設定、共有する新規ファイルシステムの追加、ファイルシステムのマウント、Secure NFS システムの使用、WebNFS 機能の使用などです。章の最後では障害追跡の手順を説明し、NFS の多くのエラーメッセージとその意味を示します。

- 14ページの「ファイルシステムの自動共有」
- 15ページの「ファイルシステムのマウント」
- 17ページの「NFS サービスの設定」
- 21ページの「Secure NFS システムの管理」
- 25ページの「WebNFS の管理タスク」
- 28ページの「NFS における障害追跡の概要」
- 29ページの「NFS の障害追跡手順」
- 37ページの「NFS のエラーメッセージ」

NFS 管理者の責任は、サイトの要求やネットワーク上に存在するコンピュータの役割によって変わります。管理者がローカルネットワークのコンピュータすべてに責任を持つこともありえます。そのような場合は、以下の設定事項について判断する必要があります。

- サーバ専用のコンピュータを置く場合、そのコンピュータの決定
- サーバとクライアントの両方として動作するコンピュータの決定
- クライアントとしてのみ動作するコンピュータの決定

設定が完了したサーバの保守には、以下の作業が必要です。

- ファイルシステムの共有開始と共有解除
- 管理ファイルを修正し、コンピュータが共有したり、自動的にマウントしたファイルシステムのリストを更新すること。
- ネットワークの状態のチェック
- NFS に関連した問題の診断と解決
- autofs のマップの設定

コンピュータは、サーバとクライアントのどちらにもなれることに注意してください。ローカルファイルシステムをリモートコンピュータと共有したり、リモートファイルシステムをマウントしたりできます。

ファイルシステムの自動共有

NFS 環境でファイルシステムを共有することにより、サーバのファイルシステムにアクセスできるようになります。共有するファイルシステムは、`share` コマンドや `/etc/dfs/dfstab` ファイルに指定します。

`/etc/dfs/dfstab` ファイルの項目は、NFS サーバオペレーションを起動したときに自動的に共有されます。同じファイルシステムを定期的に共有する必要がある場合は、自動共有を設定しなければなりません。たとえばサーバがディスクのないクライアントをサポートしている場合、クライアントのルートディレクトリを常に使用できるようにしておく必要があります。ファイルシステムの共有はほとんどが自動的に行われます。共有を手動で実行するのは、テストか障害追跡の場合だけです。

`dfstab` ファイルには、サーバがクライアントと共有するすべてのファイルシステムが列挙されており、どのクライアントがファイルシステムをマウントできるかを制御します。`dfstab` を修正してファイルシステムの追加や削除を行う場合、または共有方法を修正する場合には、ファイルを `vi` などのテキストエディタで編集します。コンピュータが次に実行レベル 3 に入ったときに、システムが、更新された `dfstab` を読み、ファイルシステムの共有方法が自動的に判断されます。

`dfstab` ファイルの各行は、`share` コマンドで構成されています。その `share` コマンドは、コマンド行プロンプトに入力してファイルシステムを共有するのと同じコマンドです。`share` コマンドは、`/usr/sbin` に保存されています。

▼ ファイルシステム自動共有を設定する方法

1. **`/etc/dfs/dfstab` ファイルを編集します。**

自動的に共有するファイルシステムの項目を dfstab ファイルに書き込みます。各項目は、ファイルの 1 行に納めなければなりません。構文は以下のとおりです。

```
share [-F nfs] [-o specific-options] [-d description] pathname
```

2. **NFS** サービスがサーバで動作していることを確認します。

   share コマンド、または share コマンドのセットを初めて実行する場合は、NFS デーモンが動作しない傾向が高くなります。以下のコマンドでデーモンを削除し、再起動してください。

```
# /etc/init.d/nfs.server stop
# /etc/init.d/nfs.server start
```

これで NFS サービスがサーバで実行されます。ブート時にサーバが実行レベル 3 になったときには、自動的に再起動されます。

この時点で autofs マップを設定し、サーバで共有しているファイルシステムにクライアントがアクセスできるようにします。84ページの「autofs の設定」 を参照してください。

ファイルシステムのマウント

ファイルシステムをマウントするには、いくつかの方法があります。システムをブートするときに自動的にマウントされるようにするか、コマンド行から必要に応じてマウントするか、オートマウンタを使用します。オートマウンタには、ブート時のマウントやコマンド行からのマウントに比較していくつもの利点がありますが、状況によってこれら 3 つを組み合わせることが必要です。

▼ ブート時のマウント

autofs マップを使用するのではなく、ブート時にファイルシステムをマウントするには、次の手順に従います。この手順は、すべてのローカルファイルシステムで実

行しなければなりません。すべてのクライアントで実行しなければならないので、リモートファイルシステムでの実行は推奨できません。

- /etc/vfstab ファイルを編集します。

/etc/vfstab ファイルのエントリ構文は、次のとおりです。

```
special fsckdev mountp fstype fsckpass mount-at-boot mntopts
```

vfstab ファイルの項目の例

wasp サーバの /var/mail ディレクトリをクライアントに /var/mail としてマウントするとします。そのためには、クライアント側で、ファイルシステムを /var/mail としてマウントし、読み出しと書き込みの両方ができるようにします。この場合は、以下の項目をクライアントの vfstab ファイルに追加します。

```
wasp:/var/mail - /var/mail nfs - yes rw
```

注意 - NFS サーバに NFS vfstab ファイルのエントリを作成するとデッドロックが発生する可能性があるため、作成してはなりません。NFS サービスは、/etc/vfstab のエントリがチェックされてから起動されます。そのため、互いのファイルシステムをマウントしている 2 台のサーバが同時にダウンすると、リブート中にシステムがハングする可能性があります。

▼ コマンド行からマウントする方法

通常オペレーションの間にファイルシステムを手動でマウントするには、mount コマンドをスーパーユーザとして実行します。

```
# mount -F nfs -o ro bee:/export/share/local /mnt
```

上の例では、bee サーバの /export/share/local ファイルシステムが、ローカルシステムの /mnt に読み取り専用でマウントされます。コマンド行からこのようにマウントすることにより、ファイルシステムを一時的に表示することができます。umount を実行するかローカルホストをリブートすると、このマウントは解除されます。

注意 - Solaris 2.6 およびそれ以降に出たパッチに置き換えられた mount コマンドでは、無効なオプションを指定しても警告されません。解釈できないオプションがあると無視されるだけです。予想外の結果が生じるのを避けるために、使用するオプションはすべて確認してください。

▼ オートマウンタを使ってマウントする方法

第 5 章 では、オートマウンタによるマウント方法と保守方法について説明します。通常のシステムに変更を加えずに、リモートファイルシステムが /net マウントポイントでアクセスできるようになります。前の例のように /export/share/local ファイルシステムをマウントする場合、以下を入力するだけです。

```
% cd /net/bee/export/share/local
```

オートマウンタでは、すべてのユーザがファイルシステムをマウントできるので、root としてアクセスする必要はありません。またファイルシステムのマウントを自動的に解除できるので、作業の終了後、ファイルシステムのマウントを解除する必要はありません。

NFS サービスの設定

この節では、NFS サービスを起動または使用するために必要なタスクについて説明します。

▼ NFS サービスを起動する方法

- ♦ リブートせずにデーモンを起動するには、スーパーユーザとしてログインして次のコマンドを入力します。

```
# /etc/init.d/nfs.server start
```

/etc/dfs/dfstab の中にエントリがあれば、これによってデーモンが起動します。

▼ NFS サービスを停止する方法

- リブートせずにデーモンを停止するには、スーパーユーザとしてログインして次のコマンドを入力します。

```
# /etc/init.d/nfs.server stop
```

▼ NFS サーバ上の大型ファイルを無効にする方法

1. ファイルシステムに大型ファイルが存在しないことを確認します。

   大型ファイルを検索するコマンドの例を示します。

```
# cd /export/home1
# find . -xdev -size +2000000 -exec ls -l {} \;
```

   このファイルシステムに大型ファイルが存在する場合は、削除するか別のファイルシステムに移動しなければなりません。

2. ファイルシステムをアンマウントします。

```
# umount /export/home1
```

3. ファイルシステムが –largefiles を使ったマウントされていた場合は、ファイルシステムの状態をリセットします。

   fsck を使うと、大型ファイルが存在しないファイルシステムの状態をリセットできます。

```
# fsck /export/home1
```

4. –nolargefiles を使ってファイルシステムをマウントします。

```
# mount -F ufs -o nolargefiles /export/home1
```

これはコマンド行からも実行できますが、このオプションを何度も使う場合には次のようなエントリを /etc/vfstab ファイルに追加しておきます。

```
/dev/dsk/c0t3d0s1 /dev/rdsk/c0t3d0s1 /export/home1  ufs     2       yes     nolargefiles
```

注 - 今までの Solaris 環境では、大型ファイルは扱えません。クライアントが大型ファイルにアクセスする必要があるときは、NFS サーバのクライアントで実行されている Solaris のバージョンが 2.6 以上であることを確認してください。

▼ クライアント側障害時回避機能の使用方法

- **NFS** クライアントで、–ro オプションを使ってファイルシステムをマウントします。

これは、コマンド行からも、オートマウンタを使っても、または /etc/vfstab ファイルに次のようなエントリを追加することによっても実現できます。

```
bee,wasp:/export/share/local - /usr/local nfs - no -o ro
```

この構文は古いバージョンのオートマウンタでも受け入れられていましたが、ファイルシステムはマウントされても障害時回避機能は使用できなかったため、サーバが選択されるだけでした。

注 - 異なるバージョンの NFS プロトコルを実行しているサーバを、コマンド行や vfstab のエントリに混在させないでください。サポートしているプロトコルが NFS V2 のサーバと V3 のサーバとを混在できるのは、autofs を使用するときだけです。この場合、バージョン 2 かバージョン 3 のサーバのうち、多い方が使われます。

▼ 1 つのクライアントに対するマウントアクセスを無効にする方法

1. /etc/dfs/dfstab ファイルを編集します。

   1 つめの例では、eng ネットグループ内のクライアントのうち rose というホスト以外すべてに対してマウントアクセスが許可されます。2 つめの例では、eng.sun.com DNS ドメイン内のクライアントのうち rose 以外すべてに対してマウントアクセスが許可されます。

```
share -F nfs -o ro=-rose:eng /export/share/man
share -F nfs -o ro=-rose:.eng.sun.com /export/share/man
```

アクセスリストについての詳細は、58ページの「share コマンドを使ってアクセスリストを設定する」 を参照してください。

2. shareall コマンドを実行します。

   /etc/dfs/dfstab への変更は、このファイルシステムがもう 1 度共有されるかサーバがリブートされるまでは NFS サーバに反映されません。

```
# shareall
```

▼ ファイアウォールを越えて NFS ファイルシステムをマウントする方法

1. スーパーユーザになります。

2. 手動でファイルシステムをマウントします。たとえば、次のようなコマンドを入力します 。

```
# mount -F nfs -o public bee:/export/share/local /mnt
```

この例では、/export/share/local というファイルシステムは、公共ファイルハンドルを使ってローカルクライアントにマウントしています。標準のパス名の代わりに、NFS URL を使用することができます。ただし bee サーバで公共ファイルハンドルがサポートされていないと、マウント操作は失敗します。

注 - この手順は、NFS サーバ上のファイルシステムが public オプションを使用して共有されていることと、クライアントとサーバの間のすべてのファイアウォールでポート 2049 による TCP 接続が可能であることが前提です。Solaris 2.6 からは、共有されているファイルシステムはすべて公共ファイルハンドルによるアクセスが可能です。

▼ NFS URL を使用して NFS ファイルシステムをマウントする方法

1. スーパーユーザになります。

2. 手動でファイルシステムをマウントします。たとえば、次のようなコマンドを入力します 。

```
# mount -F nfs nfs://bee:3000/export/share/local /mnt
```

この例では、bee というサーバの /export/share/local というファイルシステムが、NFS ポート番号 3000 を使用してマウントされます。ポート番号は指定しなくてもかまいません。その場合、デフォルトの NFS ポート番号である 2049 が使用されます。NFS URL には public オプションを指定できます。public オプションを指定しない場合、サーバが公共ファイルハンドルをサポートしていなければ、MOUNT プロトコルが使用されます。public オプションを指定すると、必ず公共ファイルハンドルを使用するように指定され、公共ファイルハンドルがサポートされていないとマウントは失敗します。

Secure NFS システムの管理

Secure NFS システムを使用するには、自分が責任を持つすべてのコンピュータにドメイン名が必要です。「ドメイン」とは管理上の実体であり、一般には、大きなネットワークに参加する複数のコンピュータから構成されます。NIS+ を実行している場合、そのドメインに対して NIS+ ネームサービスを設定しなければなりません。『*Solaris* ネーミングの設定と構成』を参照してください。

Secure NFS 環境は、認証に Diffie-Hellman (DH) か Kerberos (KERB) バージョン 4、または両方の組み合わせを使用するように設定できます。これらの認証サービスについては、『*Solaris* のシステム管理 *(*第 *2* 巻*)*』の「システムセキュリティの管理の概要」を参照してください。

▼ DH 認証を使って Secure NFS 環境を設定する方法

1. ドメインにドメイン名を割り当て、そのドメイン名をドメイン内の各コンピュータに知らせます。

ネームサービスとして NIS+ を使っている場合は、『*Solaris* ネーミングの管理』を参照してください。

2. newkey または nisaddcred コマンドを使ってクライアントのユーザの公開鍵と秘密鍵を設定して、各ユーザに chkey コマンドを使って各自の **Secure RPC** パスワードを設定してもらいます。

注 - これらのコマンドについての詳細は、newkey(1M)、nisaddcred(1M)、chkey(1) のマニュアルページを参照してください。

公開鍵と秘密鍵が生成されると、公開鍵と暗号化された秘密鍵が publickey データベースに格納されます。

3. ネームサービスが応答していることを確認します。**NIS+** を実行している場合は、以下を入力してください。

```
# nisping -u
Last updates for directory eng.acme.com. :
Master server is eng-master.acme.com.
 Last update occurred at Mon Jun 5 11:16:10 1995

Replica server is eng1-replica-replica-58.acme.com.
 Last Update seen was Mon Jun 5 11:16:10 1995
```

NIS を実行している場合は、ypbind デーモンが動作していることを確認してください。

4. keyserv デーモン **(**キーサーバ**)** が動作していることを、以下を入力することで確認してください。

```
# ps -ef | grep keyserv
root 100    1   16  Apr 11 ?    0:00 /usr/sbin/keyserv
root  2215 2211    5 09:57:28 pts/0 0:00 grep keyserv
```

keyserv デーモンが動作していない場合は、以下を入力してキーサーバを起動します。

```
# /usr/sbin/keyserv
```

5. keylogin を実行し、秘密鍵の復号化と保存を実行してください。

   通常、ログインパスワードはネットワークパスワードと同じです。この場合、keylogin は不要です。ログインパスワードとネットワークパスワードが異なる場合、ユーザはログインしてから keylogin を実行しなければなりません。 また、keylogin -r を root として実行し、復号化した秘密鍵を /etc/.rootkey に保存しなければなりません。

注 - keylogin -r は、root の秘密鍵が変更されたか、/etc/.rootkey が損失した場合にのみ、実行する必要があります。

---

6. /etc/dfs/dfstab ファイルを編集し、該当するエントリに –sec=dh オプションを追加します **(DH** 認証用**)**。

```
share -F nfs -o sec=dh /export/home
```

7. auto_master マップを編集し、該当するエントリのマウントオプションとして –sec=dh を指定します **(DH** 認証用**)**。

```
/home auto_home -nosuid,sec=dh
```

注 - リリース 2.5 以前の Solaris では、セキュリティ保護されているものとして共有されているファイルシステムを、セキュリティ保護されたものとしてクライアントがマウントしないと、ユーザはそのユーザ本来の立場ではなく未認証としてのアクセス権しか得られません。Solaris 2.5 以降の NFS バージョン 2 では、セキュリティモードが一致しないと、share コマンド行に –sec=none が指定されていない限り、NFS サーバによってアクセスが拒否されます。NFS のバージョン 3 では、セキュリティ保護されていることを示すモードが NFS サーバから引き継がれるので、クライアントが –sec=krb4 や –sec=dh を指定する必要はありません。ユーザは、そのユーザ自身としてファイルにアクセスできます。

コンピュータを設置し直したり、移設したり、アップグレードするときに、新しい鍵を設定せず、root 用の鍵も変更しない場合は、忘れずに /etc/.rootkey

を保存してください。/etc/.rootkey を削除する場合は、以下を実行してください。

```
# keylogin -r
```

▼ セキュリティ保護された NFS 環境を KERB 認証を使用して設定する方法

1. /etc/dfs/dfstab ファイルを編集して、所定のエントリに sec=krb4 オプションを追加します。

```
# share -F nfs -o sec=krb4 /export/home
```

2. auto_master データを編集して、sec=krb4 をマウントオプションとして指定します。

```
/home auto_home -nosuid,sec=krb4
```

注 - リリース 2.5 以降の Solaris では、セキュリティ保護されているものとして共有されているファイルシステムを、セキュリティ保護されたものとしてクライアントがマウントしないと、ユーザはそのユーザ本来の立場ではなく未認証としてのアクセス権しか得られません。Solaris 2.5 以降の NFS のバージョン 2 では、セキュリティモードが一致しないと、share コマンド行に –sec=none が指定されていない限り、NFS サーバによってアクセスが拒否されます。バージョン 3 では、セキュリティ保護されていることを示すモードが NFS サーバから引き継がれるので、クライアントが –sec=krb4 や –sec=dh を指定する必要はありません。ユーザは、そのユーザ自身としてファイルにアクセスできます。

WebNFS の管理タスク

この節では、WebNFS システムを管理する方法を説明します。以下のタスクについて説明します。

- 25ページの「WebNFS アクセスの計画」
- 26ページの「WebNFS アクセスを有効にする方法」
- 27ページの「NFS URL を使ったブラウズ」
- 28ページの「ファイアウォール経由で WebNFS を使用する方法」

WebNFS アクセスの計画

WebNFS の機能を使用するには、まずアプリケーションを実行して NFS URL (`nfs://server/path` など) を読み込む必要があります。次に、WebNFS アクセスのためにエクスポートするファイルシステムを選択します。アプリケーションが Web ブラウザの場合には、Web サーバの文書のルートがよく使われます。WebNFS アクセスのためにエクスポートするファイルシステムを選択するときには、考慮すべきことがいくつかあります。

1. 各サーバには公共ファイルハンドルが 1 つずつあり、このハンドルはデフォルトではサーバのルートファイルシステムに結び付けられています。NFS URL に示されたパスは、この公共ファイルハンドルが結び付けられているディレクトリからの相対パスとして評価されます。その結果としてパスが示す先のファイルまたはディレクトリが、エクスポートされたファイルシステムの中にあると、サーバによってアクセスが実現されます。`share` コマンドの `–public` オプションを使うと、エクスポートされる特定のディレクトリにこの公開ファイルハンドルを結び付けることができます。このオプションを使うと、URL はサーバのルートファイルシステムではなく共有ファイルシステムからの相対パスになります。デフォルトでは公開ファイルハンドルはルートファイルシステムを示していますが、ルートファイルシステムを共有しないかぎりこのファイルハンドルでは Web アクセスはできません。
2. WebNFS 環境では、すでにマウント権限を持っているユーザはファイルシステムが `–public` オプションを使ってエクスポートされているかどうかに関係なく、ブラウザからファイルにアクセスできます。ユーザは NFS の設定によってファイルに対するアクセス権を持っているため、ブラウザからのアクセスを許すことによって新たにセキュリティが損なわれるおそれはありません。 ファイル

システムをマウントできないユーザは、–public オプションを使ってファイルシステムを共有するだけで WebNFS アクセスを使えるようになります。

3. FTP アーカイブの最上位ディレクトリや Web サイトの中心となる URL など、すでに公開されているファイルシステムは –public オプションを使用する対象の有力な候補です。
4. share コマンドで –index オプションを使うと、NFS URL がアクセスされたときにディレクトリがリストされるのではなく HTML ファイルがロードされます。

ファイルシステムを選択したらファイルを確認し、必要に応じてファイルやディレクトリの表示を制限するようにアクセス権を設定します。アクセス権は、共有される NFS ファイルシステムに合わせて設定します。多くのサイトでは、ディレクトリに対しては 755、ファイルに対しては 644 が適切なアクセスレベルです。

1 つの Web サイトへのアクセスに NFS URL と HTTP URL の両方を使用する場合には、ほかにも考慮すべき要素があります。75ページの「Web ブラウザと比較した場合の WebNFS の制約」 を参照してください。

▼ WebNFS アクセスを有効にする方法

バージョン 2.6 以降では、NFS マウントが可能なファイルシステムはすべてデフォルトで自動的に WebNFS アクセスでも利用できます。この手順を実行する必要があるのは、NFS マウントが許可されていないサーバで、公開ファイルハンドルをリセットすることで NFS URL を短くできるか –index オプションが必要な場合だけです。

1. /etc/dfs/dfstab ファイルを編集します。

自動的に共有したいファイルシステムに対応するエントリを 1 つ追加します。–index タグは省略可能です。

```
share -F nfs -o ro,public,index=index.html /export/ftp
```

2. NFS サービスがサーバで実行されていることを確認します。

share コマンド、または share コマンドのセットを初めて実行する場合は、NFS デーモンが動作していないと考えられます。以下のコマンドでデーモンを削除し、再起動してください。

```
# /etc/init.d/nfs.server stop
# /etc/init.d/nfs.server start
```

3. ファイルシステムを共有します。

   /etc/dfs/dfstab ファイルの中にエントリが存在していれば、システムをリブートするか shareall コマンドを実行することによってファイルシステムを共有できます。手順 2 で NFS デーモンを再起動した場合には、そのスクリプトによって shareall コマンドが実行されるため、改めて実行する必要はありません。

```
# shareall
```

4. 情報が正しいことを確認します。

   share コマンドを実行して、表示されるオプションが正しいか確認します。

```
# share
- /export/share/man ro ""
- /usr/src rw=eng ""
- /export/ftp ro,public,index=index.html ""
```

▼ NFS URL を使ったブラウズ

WebNFS アクセスをサポート可能なブラウザは、次のような形式の NFS URL を使ってアクセスを実現します。

```
nfs://server<:port>/path
```

server はファイルサーバの名前、*port* はポート番号 (デフォルトは 2049)、*path* はファイルへのパスです。パスは、そのサーバの公開ファイルハンドルまたはルートファイルシステムからの相対パスで示します。

注 - ほとんどのブラウザでは、URL サービスタイプ (nfs や http など) は別のサービスタイプの URL が読み込まれるまで次のトランザクションに引き継がれます。NFS URL を使用しているときに HTTP URL への参照が読み込まれると、それ以降のページは次に URL で NFS URL が指定されるまで、NFS プロトコルではなく HTTP プロトコルを使って読み込まれます。

▼ ファイアウォール経由で WebNFS を使用する方法

ローカルのサブネットに属していないクライアントに対して WebNFS アクセスを有効にするには、ポート 2049 での TCP 接続を許可するようにファイアウォールを設定します。httpd に対してアクセスを許可するだけでは、NFS URL が使えるようにはなりません。

NFS における障害追跡の概要

NFS のトラブルを追跡するとき、主な障害発生ポイントとしてサーバ、クライアント、またはネットワーク自体の 3 つがあることを覚えておいてください。この節で説明するのは、個々の構成要素を切り離して、正常に動作しない部分を見つ出そうというものです。リモートマウントを正常に実行するには、サーバ上で mountd と nfsd が動作している必要があります。

注 - /etc/dfs/dfstab ファイルに NFS 共有エントリがある場合、mountd と nfsd はブート時に自動的に起動します。したがって、最初に共有設定を行うときには mountd と nfsd を手作業で起動しなければなりません。

デフォルトでは、すべてのマウントに –intr オプションが設定されます。プログラムが「server not responding」(サーバが応答しません) というメッセージを出してハングした場合、これはキーボード割り込み (Ctrl-C) で終了できます。

ネットワークまたはサーバに問題がある場合、ハードマウントされたリモートファイルにアクセスするプログラムの障害と、ソフトマウントされたリモートファイルにアクセスするプログラムの障害とは異なります。ハードマウントされたリモートファイルシステムの場合、クライアントのカーネルは、サーバが再び応答するまで要求を再試行します。ソフトマウントされたリモートファイルシステムの場合、クライアントのシステムコールは、しばらく試行した後でエラーを返します。このエラーによって予想外のアプリケーションエラーやデータ破壊が起きるおそれがあるため、ソフトマウントは行わないでください。

ファイルシステムがハードマウントされていると、サーバが応答に失敗した場合には、これにアクセスしようとするプログラムはハングします。この場合、NFS は次のメッセージをコンソールに表示します。

```
NFS server hostname not responding still trying
```

サーバが少し後に応答すると、次のメッセージがコンソールに表示されます。

```
NFS server hostname ok
```

サーバが応答しないような、ソフトマウントされたファイルシステムにアクセスしているプログラムは、次のメッセージを表示します。

```
NFS operation failed for server hostname: error # (error_message)
```

注 - 読み取りと書き込みをするデータを持つファイルシステム、または実行可能ファイルを持つファイルシステムは、ソフトマウントしないでください。エラーが発生する可能性があります。アプリケーションがそのようなソフトエラーを無視すれば、書き込み可能なデータが破壊される恐れがあります。またマウントされた実行可能ファイルが正常にロードされず、動作も正常に行われない可能性があります。

NFS の障害追跡手順

NFS サービスがエラーになった場所を判断するには、いくつかの手順を踏まなければなりません。以下の項目をチェックしてください。

- クライアントとサーバがソフトウェア的に接続されているかどうか。
- クライアントが NFS サービスを受けられるかどうか。
- NFS サービスがサーバ上で動作しているかどうか。

上記の項目をチェックする過程で、ネームサービスやネットワークのハードウェアなど、ネットワークの他の部分が機能していないことが判明する場合があります。NIS+ ネームサービスのデバッグ手順については、『*Solaris* ネーミングの管理』を参照してください。問題がクライアント側のものではないことが判明することもあります (たとえば作業領域の各サブネットから、最低 1 つのトラブルコールがある場合など)。このような場合は、問題がサーバかサーバ周辺のネットワークハードウェアで発生しているとみなし、クライアントではなく、サーバでデバッグを開始するほうがよいでしょう。

▼ NFS クライアントの接続性のチェック

1. **クライアントと NFS サーバが、ソフトウェア的に接続されていることを確認します。以下のコマンドをクライアントで入力してください。**

```
% /usr/sbin/ping bee
bee is alive
```

コマンドを入力した結果、サーバが動作していることがわかったら、NFS サーバをリモートでチェックしてください (31ページの「NFS サーバをリモートでチェックする方法」 参照)。

2. クライアントとサーバがソフトウェア的に接続されていない場合は、ローカルネームサービスが動作していることを確認してください。**NIS+** クライアントでは、以下を入力します。

```
% /usr/lib/nis/nisping -u
Last updates for directory eng.acme.com. :
Master server is eng-master.acme.com.
        Last update occurred at Mon Jun  5 11:16:10 1995

Replica server is eng1-replica-58.acme.com.
        Last Update seen was Mon Jun  5 11:16:10 1995
```

3. ネームサービスが実行されている場合は、クライアントが正しいホスト情報を受け取るために次のように入力します。

```
% /usr/bin/getent hosts bee
129.144.83.117  bee.eng.acme.com
```

4. ホスト情報に誤りがなく、クライアントからサーバに接続できない場合は、別のクライアントから ping コマンドを実行します。

   ping コマンドが失敗したら、33ページの「サーバで NFS サービスを確認する方法」 を参照してください。

5. 別のクライアントとサーバがソフトウェア的に接続されている場合は、ping コマンドを使用して元のクライアントとローカルネット上の他のシステムとの接続性をチェックしてください。

   エラーになる場合は、そのクライアントのネットワークソフトウェアの構成をチェックします (/etc/netmasks、/etc/nsswitch.conf など)。

6. ソフトウェアに問題がない場合は、ネットワークハードウェアをチェックしてください。

   クライアントをネットワークの別の場所へ移動してチェックしてください。

▼ NFS サーバをリモートでチェックする方法

1. **NFS** サーバで **NFS** サービスが実行されていることを、次のコマンドを入力することによって確認します。

```
% rpcinfo -s bee|egrep 'nfs|mountd'
100003  3,2     tcp,udp                              nfs       superuser
100005  3,2,1   ticots,ticotsord,tcp,ticlts,udp     mountd    superuser
```

   デーモンが起動していなければ、34ページの「NFS サービスの再起動」 を参照してください。

2. サーバで `nfsd` プロセスが応答することをチェックしてください。クライアントで以下のコマンドを入力します。

```
% /usr/bin/rpcinfo -u bee nfs
program 100003 version 2 ready and waiting
program 100003 version 3 ready and waiting
```

   サーバが動作している場合、プログラムとバージョン番号が表示されます。`–t` オプションを使用すると、TCP 接続を検査できます。`–t` オプションでエラーが発生する場合は、33ページの「サーバで NFS サービスを確認する方法」 に進んでください。

3. サーバで `mountd` が応答することをチェックしてください。以下のコマンドを入力します。

```
% /usr/bin/rpcinfo -u bee mountd
program 100005 version 1 ready and waiting
program 100005 version 2 ready and waiting
program 100005 version 3 ready and waiting
```

–t オプションを使用すると、TCP 接続を検査できます。エラーになる場合は、33ページの「サーバで NFS サービスを確認する方法」 に進んでください。

4. ローカル **autofs** サービスを使用していた場合は、それをチェックしてください。

```
% cd /net/wasp
```

/net か /home マウントポイントのうち、適切に動作する方をチェックします。動作しない場合は、以下のコマンドをルートとしてクライアントから入力し、autofs サービスを再起動してください。

```
# /etc/init.d/autofs stop
# /etc/init.d/autofs start
```

5. サーバのファイルシステムの共有が正常に行えることを確認します。

```
% /usr/sbin/showmount -e bee
/usr/src          eng
/export/share/man      (everyone)
```

サーバの項目とローカルマウントエントリにエラーがないことをチェックします。名前空間もチェックしてください。この例で最初のクライアントが eng ネットグループの中にない場合、/usr/src ファイルシステムはマウントできません。

すべてのローカルファイルを調べて、マウント情報を含むエントリをすべて検査します。リストには、/etc/vfstab とすべての /etc/auto_* ファイルが含まれています。

▼ サーバで NFS サービスを確認する方法

1. サーバに root としてログインします。

2. サーバとクライアントがソフトウェア的に接続されていることを確認します。

```
# ping lilac
lilac is alive
```

3. サーバとクライアントがソフトウェア的に接続されていない場合は、ローカルネームサービスが動作していることを確認します。**NIS+** クライアントで以下のコマンドを入力してください。

```
% /usr/lib/nis/nisping -u
Last updates for directory eng.acme.com. :
Master server is eng-master.acme.com.
        Last update occurred at Mon Jun  5 11:16:10 1995

Replica server is eng1-replica-58.acme.com.
        Last Update seen was Mon Jun  5 11:16:10 1995
```

4. ネームサービスが動作している場合は、サーバにあるネットワークソフトウェアの構成をチェックしてください **(**/etc/netmasks、/etc/nsswitch.conf など**)**。

5. 以下のコマンドを入力し、nfsd デーモンが動作していることをチェックします。

```
# rpcinfo -u localhost nfs
program 100003 version 2 ready and waiting
program 100003 version 3 ready and waiting
# ps -ef | grep nfsd
root    232      1  0  Apr 07     ?     0:01 /usr/lib/nfs/
nfsd -a 16 root   3127
2462  1  09:32:57  pts/3  0:00 grep nfsd
```

rpcinfo の –t オプションを使用し、TCP 接続を確認してください。エラーになる場合は、NFS サービスを再起動してください (34ページの「NFS サービスの再起動」 参照)。

6. 以下のコマンドを入力し、mountd デーモンが動作していることを確認します。

```
# /usr/bin/rpcinfo -u localhost mountd
program 100005 version 1 ready and waiting
program 100005 version 2 ready and waiting
program 100005 version 3 ready and waiting
# ps -ef | grep mountd
root    145    1 0 Apr 07 ?   21:57 /usr/lib/autofs/automountd
root    234    1 0 Apr 07 ?    0:04 /usr/lib/nfs/mountd
root   3084    2462 1 09:30:20 pts/3  0:00 grep mountd
```

rpcinfo に –t オプションを指定し、TCP 接続もチェックしてください。エラーになる場合は、NFS サービスを再起動します (34ページの「NFS サービスの再起動」 参照)。

7. 以下のコマンドを入力し、rpcbind デーモンが動作していることを確認します。

```
# /usr/bin/rpcinfo -u localhost rpcbind
program 100000 version 1 ready and waiting
program 100000 version 2 ready and waiting
program 100000 version 3 ready and waiting
```

rpcbind がハングしている場合は、サーバをリブートするか、35ページの「rpcbind のワームスタート」 に示す作業を行ってください。

▼ NFS サービスの再起動

- リブートせずにデーモンを動作させるには、スーパーユーザになって以下のコマンドを入力します。

```
# /etc/init.d/nfs.server stop
# /etc/init.d/nfs.server start
```

/etc/dfs/dfstab に項目がある場合、デーモンは停止してから再起動します。

▼ rpcbind のワームスタート

何らかのプログラムが動作しているために NFS サーバをリブートできない場合は、以下のようにワームスタートすることで、RPC を使用するすべてのサービスを再起動せずに rpcbind を再起動できます。

1. **root** としてサーバにログインし、rpcbindno の **PID** を調べます。

   ps を実行すると、PID の値が第 2 カラムに表示されます。

```
# ps -ef |grep rpcbind
    root   115     1  0   May 31 ?        0:14 /usr/sbin/rpcbind
    root 13000  6944  0 11:11:15 pts/3    0:00 grep rpcbind
```

2. **SIGTERM** シグナルを rpcbind プロセスに送ります。

   以下の例では、送信するシグナルは term で、プログラムの PID は 115 です (kill(1) のマニュアルページ参照)。これにより、rpcbind は /tmp/portmap.file と /tmp/rpcbind.file に現在登録されているサービスのリストを作成します。

```
# kill -s term 115
```

注 - -s term オプションを使って rpcbind プロセスを終了させないと、rpcbind のワームスタートを完了できません。その場合は、サーバを再起動することによってサービスを再開する必要があります。

---

3. rpcbind を再起動します。

   rpcbind のワームスタートを実行すると、kill コマンドの実行で作成されたファイルが参照されるので、すべてのRPCサービスを再起動せずにプロセスを再起動できます (rpcbind(1M) のマニュアルページ参照)。

```
# /usr/sbin/rpcbind -w
```

▼ NFS ファイルサービスを提供しているホストを識別する方法

- ♦ –m オプションを指定して nfsstat コマンドを実行し、最新の **NFS** 情報を取得します。

現在のサーバの名前は、“currserver=” の後に表示されます。

```
% nfsstat -m
/usr/local from bee,wasp:/export/share/local
 Flags: vers=3,proto=tcp,sec=sys,hard,intr,llock,link,synlink,
  acl,rsize=32768,wsize=32678,retrans=5
 Failover: noresponse=0, failover=0, remap=0, currserver=bee
```

▼ mount コマンドに使用されたオプションを確認する方法

Solaris 2.6 およびそれ以降に出たパッチに置き換えられた mount コマンドでは、無効なオプションを指定しても警告されません。コマンド行に入力したオプション、または /etc/vfstab から指定したオプションが有効であるかどうかを判断するには、以下の手順を実行します。

たとえば、次のコマンドが実行されたとします。

```
# mount -F nfs -o ro,vers=2 bee:/export/share/local /mnt
```

1. **nfsstat** コマンドを実行してカーネルを確認します。

```
# nfsstat -m
/mnt from bee:/export/share/local
Flags:  vers=2,proto=tcp,sec=sys,hard,intr,dynamic,acl,rsize=8192,wsize=8192,
        retrans=5
```

ここで、bee からマウントされたファイルシステムは、プロトコルのバージョンが 2 に設定されています。nfsstat コマンドを使用しても、一部のオプションの情報は表示されませんが、オプションを確認するにはこれが最も正確な方法です。

2. /etc/mnttab でエントリを確認します。

   mount コマンドでは、無効なオプションはマウントテーブルに追加されません。したがって、実行したコマンド行のオプションと /etc/mnttab にある当該オプションを比較すれば、nfsstat コマンドによってレポートされないオプションがわかります。

```
# grep bee /etc/mnttab
bee:/export/share/local /mnt nfs        ro,vers=2,dev=2b0005e 859934818
```

NFS のエラーメッセージ

ここでは、エラーメッセージ、そのエラーの原因となる条件、およびその問題を解決する方法を少なくとも 1 つ示します。

`Bad argument specified with index option - must be a file`

–index オプションにはファイル名を指定する必要があります。ディレクトリ名は使えません。

`Cannot establish NFS service over /dev/`*tcp*`: transport setup problem`

このメッセージは、名前空間の中のサービス情報が更新されなかったときによく発生します。UDP に関して報告されることもあります。この問題を解決するには、名前空間の中のサービスデータを更新します。NIS+ の場合、エントリは以下のとおりです。

```
nfsd nfsd tcp 2049 NFS server daemon
nfsd nfsd ucp 2049 NFS server daemon
```

NIS と /etc/services の場合、エントリは以下のとおりです。

```
nfsd    2049/tcp    nfs    # NFS server daemon
nfsd    2049/ucp    nfs    # NFS server daemon
```

`Cannot use index option without public option`

share コマンドに public オプションを指定してください。–index オプションを使うには、公開ファイルハンドルを定義する必要があります。

注 - Solaris 2.6 より前の Solaris では、share コマンドを使って公共ファイルハンドルを設定する必要があります。Solaris 2.6 以降では、公共ファイルハンドルはデフォルトで / に設定されるため、このエラーメッセージは出力されません。

`Could not use public filehandle in request to` *server*

このメッセージは、public オプションが指定されているにもかかわらず NFS サーバが公共ファイルハンドルをサポートしていない場合に表示されます。この場合、マウントは失敗します。この問題を解決するには、公共ファイルハンドルを使わずにマウント要求を行うか、NFS サーバが公共ファイルハンドルをサポートするように再設定します。

`NOTICE: NFS3: failing over from` *host1* `to` *host2*

このメッセージは、障害時回避が発生するとコンソールに表示されます。報告のためだけのメッセージです。

filename`: File too large`

NFS バージョン 2 クライアントが、2 ギガバイトを超えるサイズのファイルにアクセスしようとしています。

```
mount: ...
server not responding:RPC_PMAP_FAILURE - RPC_TIMED_OUT
```

実行レベルの誤りか、rpcbind の停止かハングのため、マウント先のファイルシステムを共有しているサーバがダウンしているか、またはソフトウェア的に接続されていないことを示すメッセージです。

`mount: ... server not responding: RPC_PROG_NOT_REGISTERED`

マウントが rpcbind によって登録されているにもかかわらず、NFS マウントデーモン (mountd) が登録されていないことを示すメッセージです。

`mount: ... No such file or directory`

リモートディレクトリかローカルディレクトリが存在しないことを示すメッセージです。ディレクトリ名の綴りをチェックするか、リモートディレクトリとローカルディレクトリの両方で `ls` コマンドを実行してください。

`mount: ...: Permission denied`

コンピュータの名前が、クライアントのリストに載っていないか、マウントするファイルシステムにアクセスできるネットグループに含まれていないことを示すメッセージです。`showmount -e` を実行し、アクセスリストを確認してください。

`nfs mount: ignoring invalid option "`*`-option`*`"`

-option フラグが無効です。正しい構文を、マニュアルページの `mount_nfs(1M)` で確認してください。

注 - このエラーメッセージは、Solaris 2.6 以降、またはそれ以前のバージョンにパッチを適用した状態で `mount` コマンドを実行したときには表示されません。

`nfs mount: NFS can't support "nolargefiles"`

NFS クライアントが、–nolargefiles オプションを使って NFS サーバからファイルシステムをマウントしようとしました。このオプションは、NFS ファイルシステムに対してはサポートされていません。

`nfs mount: NFS V2 can't support "largefiles"`

NFS バージョン 2 プロトコルでは、大型ファイルを処理できません。大型ファイルを扱う必要がある場合は、バージョン 3 を使用してください。

`NFS server `*`hostname`*` not responding still trying`

ファイル関連の作業中にプログラムがハングすると、NFSサーバは停止します。このメッセージは、NFS サーバ (*hostname*) がダウンしているか、サーバかネットワークに問題があることを示すものです。障害時回避機能を使用している場合、*hostname* はサーバ名のリストになります。29ページの「NFS の障害追跡手順」 を参照してください。

`NFS fsstat failed for server` *hostname*`: RPC: Authentication error`

様々な状況で発生するエラーです。最もデバッグが困難なのは、ユーザの属しているグループが多すぎる場合です。現在、ユーザは最大 16 個のグループに属すことができますが、NFS マウントでファイルにアクセスしている場合は、それ以下になります。NFS サーバと NFS クライアントで Solaris 2.5 以降が動作している場合に、ユーザが 16 以上のグループに属さなければならない場合は、ACL を使用することで必要なアクセス権を獲得できます。

`port` *number* `in nfs URL not the same as port` *number* `in port option`

NFS URL のポート番号は、マウントの `–port` オプションのポート番号と一致していなければなりません。一致していないと、マウントは失敗します。同じポート番号にしてコマンドを再実行するか、ポート番号の指定を省略してください。原則として、NFS URL と `–port` オプションの両方にポート番号を指定しても意味がありません。

`relicas must have the same version`

NFS 障害時回避機能が正しく機能するためには、複製の NFS サーバが同じバージョンの NFS プロトコルをサポートしていなければなりません。バージョン 2 とバージョン 3 のサーバが混在することは許されません。

`replicated mounts must be read-only`

NFS 障害時回避機能は、読み書き可能としてマウントされたファイルシステムでは動作しません。ファイルシステムを読み書き可能としてマウントすると、ファイルが変更される可能性が高くなるためです。NFS の障害時回避機能は、ファイルシステムがまったく同じであることが前提です。

`replicated mounts must not be soft`

複製されるマウントの場合、障害時回避が発生するまでタイムアウトを待つ必要があります。`soft` オプションを指定すると、タイムアウトが開始してすぐにマウントが失敗するため、複製されるマウントには `–soft` オプションは指定できません。

`share_nfs: Cannot share more than one filesystem with 'public' option`

`/etc/dfs/dfstab` ファイルを調べて、`–public` オプションによって共有するファイルシステムを複数選択していないか確認してください。公開ファイルハンドルは、サーバあたり 1 つしか設定できません。したがって、`–public` オプションで共有できるファイルシステムは 1 つだけです。

`WARNING: No network locking on` *hostname*`:`*path*`: contact admin to install server change`

NFS クライアントが、NFS サーバ上のネットワークロックマネージャと接続を確立できませんでした。この警告は、マウントできなかったことを知らせるためではなくロックが機能しないことを警告するために出力されます。

第 3 章

NFS リファレンス

この章では、NFS コマンドの概要について説明します。また、NFS 環境のすべての構成要素とそれらが互いにどのように連携するかについても説明します。

- 43ページの「NFS ファイル」
- 44ページの「NFS デーモン」
- 47ページの「NFS コマンド」
- 63ページの「その他のコマンド」
- 68ページの「コマンドを組み合わせて使う」

NFS ファイル

いくつかの ASCII ファイルでは、いずれのコンピュータ上でも NFS アクティビティをサポートする必要があります。表 3–1 に、こういったファイルとその機能をまとめます。

表 3–1 NFS の ASCII ファイル

ファイル名	機能
/etc/mnttab	自動的にマウントしたディレクトリを含む、現在マウントしているファイルシステムを一覧表示する (mnttab(4) のマニュアルページ参照)。編集しないこと。
/etc/netconfig	トランスポートプロトコルのリスト。編集しないこと。

表 3–1　NFS の ASCII ファイル　続く

ファイル名	機能
/etc/nfssec.conf	NFS のセキュリティサービスのリスト。編集しないこと。
/etc/rmtab	NFS クライアントがリモートにマウントしたファイルシステムを一覧表示する (rmtab(4) のマニュアルページ参照)。 編集しないこと。
/etc/vfstab	ローカルにマウントするファイルシステムを定義する (vfstab(4) のマニュアルページ参照)。
/etc/default/fs	ローカルファイルシステムにおけるデフォルトファイルシステムのタイプを一覧表示する。
/etc/dfs/dfstab	共有するローカルリソースを一覧表示する。
/etc/dfs/fstypes	リモートファイルシステムにおけるデフォルトファイルシステムのタイプを一覧表示する。
/etc/dfs/sharetab	共有している、ローカルとリモートのリソースを一覧表示する (sharetab(4) のマニュアルページ参照)。編集しないこと。

/etc/dfs/fstypes の最初の項目は、リモートファイルシステムにおけるデフォルトファイルシステムのタイプとして使用されることがしばしばあります。この項目は、NFS ファイルシステムのタイプをデフォルトとして定義します。

/etx/default/fs には、項目が 1 つしかありません。ローカルディスクにおけるデフォルトファイルシステムのタイプです。クライアントやサーバでサポートするファイルシステムのタイプは、/kernel/fs のファイルをチェックして決定することができます。

NFS デーモン

NFS アクティビティをサポートするには、システムが実行レベル 3 かマルチユーザモードで動作したときに、いくつかのデーモンを開始します。mountd と nfsd の 2 つのデーモンは、NFS サーバであるシステム上で動作します。サーバデーモンの自

動起動は、NFS ファイルシステムのタイプでラベル付けされた項目が /etc/dfs/sharetab に存在するかどうかで変わります。

lockd と statd の 2 つのデーモンは NFS クライアントで動作し、NFS ファイルロッキングをサポートします。NFS サーバでも動作させなければなりません。

lockd

このデーモンは NFS ファイルのレコードロックをサポートします。ロック要求をクライアントから NFS サーバに送り、NFS サーバでローカルのロックを開始します。通常は、パラメータを指定せずに起動します。使用できるオプションは 3 つあります (マニュアルページの lockd(1M) を参照してください)。

–g *graceperiod* オプションは、サーバがリブートした場合に、その何秒後にロックを再要求するかを示します。NFS サーバはこの秒数の間、それまでのロックの再要求処理しか実行しません。他のサービスに対する要求は、この時間が経過するまで待たされます。このオプションは NFS サーバの応答性に関係するため、NFS サーバでしか変更できません。デフォルト値は 45 秒です。この値を小さくすると、サーバをリブートしてからオペレーションに復帰するまでの時間は短縮されますが、クライアントがすべてのロックを復旧できなくなる可能性が増します。

–t *timeout* オプションは、ロック要求をリモートサーバに再送信するまで何秒待つかを示します。このオプションは NFS クライアントのサービスに関係します。デフォルト値は 15 秒です。この値を小さくすると、雑音の多いネットワーク上の NFS クライアントに対する応答時間を改善できますが、ロック要求が増えることによってサーバの負荷が増す可能性があります。

nthreads オプションは、サーバが 1 つの接続について同時に処理できるスレッドの数の上限を示します。この値は、NFS サーバに対して予想される負荷に基づいて決定してください。デフォルト値は 20 です。TCP を使用する NFS クライアントはそれぞれ NFS サーバと 1 つの接続を設定するため、各 TCP クライアントはサーバ上で同時に 20 までのスレッドを使うことが許されます。UDP (ユーザデーモンプロトコル) を使用する NFS クライアントは、すべてが NFS サーバと 1 つの接続を共有します。その場合、UDP 接続が使用できるスレッドの数を増やさなければならないことがあるかもしれません。簡単な目安は 1 つの UDP クライアントにつき 2 つのスレッドですが、クライアントに対する作業負荷によってはこれで不十分なこともあります。使用するスレッドを増やすことによるマイナスは、スレッドの使用によって NFS サーバで使われるメモリが増えることです。しかしスレッドが使われないならば、*nthreads* を大きくしても影響はありません。

mountd

これは、リモートシステムからのファイルシステムマウント要求を処理して、アクセス制御を行う RPC (リモートプロシージャコール) サーバで
す。/etc/dfs/sharetab を調べることによって、リモートマウントに使用可能なファイルシステムと、リモートマウントを実行可能なシステムを判断します。–v と –r の 2 つのオプションが使えます (マニュアルページの mountd(1M) を参照してください)。

–v オプションは、コマンドを冗長モードで実行します。クライアントが取得すべきアクセス権を NFS サーバが決定するたびに、コンソールにメッセージが表示されます。この情報は、クライアントがファイルシステムにアクセスできない理由を調べるときに役立ちます。

–r オプションは、その後のクライアントからのマウント要求をすべて拒絶します。すでにファイルシステムがマウントされているクライアントには影響しません。

nfsd

これは、他のクライアントからのファイルシステム要求を処理するデーモンです。このコマンドに対してはいくつかのオプションが指定できます。オプションをすべて確認するにはマニュアルページの nfsd(1M) を参照してください。

–l オプションは、接続指向トランスポートでの NFS/TCP に対する接続キューの長さを設定します。デフォルト値は 25 エントリです。

–c #*_conn* オプションは、接続指向トランスポート 1 つあたりの接続数の上限を選択します。デフォルト値はありません。

nservers オプションは、1 台のサーバが同時に処理可能な要求の数の上限です。デフォルト値は 1 ですが、起動スクリプトでは 16 が選択されます。

このデーモンの以前のバージョンとは異なり、このバージョンの nfsd では複数のコピーを作成して要求を同時に処理することはありません。処理テーブルを ps でチェックすると、動作しているデーモンのコピーが 1 つしかないことがわかります。

statd

lockd とともに動作し、ロック管理機能にクラッシュ機能と回復機能を提供します。NFS サーバでロックを保持しているクライアントの追跡を行い、サーバがクラッシュし、リブートしているあいだに、サーバ側 statd がクライアント側

statd と連絡をとります。次にクライアント側 statd は、サーバ上のすべてのロックを再要求します。クライアントがクラッシュすると、クライアント側 statd はサーバ側 statd にそのことを伝えるので、サーバ上のロックはクリアされます。このデーモンにオプションはありません。詳細については、マニュアルページの statd(1M) を参照してください。

Solaris 7 では、statd がクライアントを追跡する方法が改善されました。Solaris 7 より前のリリースの statd では、クライアントごとにそのクライアントの修飾されていないホスト名を使用して、/var/statmon/sm にファイルが作成されました。そのため、同じホスト名の 2 つのクライアントが異なるドメインに存在する場合や、クライアントが NFS サーバと異なるドメインに存在する場合に問題が発生していました。修飾されていないホスト名にはドメインや IP アドレスの情報がないため、このようなクライアントを区別する方法がありませんでした。これに対処するため、Solaris 7 の statd では、修飾されていないホスト名に対してクライアントの IP アドレスを使用して /var/statmon/sm にシンボリックリンクを作成します。このリンクは、次のような形式です。

```
# ls -l /var/statmon/sm
lrwxrwxrwx   1 root          11 Apr 29 16:32 ipv4.192.9.200.1 -> myhost
--w-------   1 root          11 Apr 29 16:32 myhost
```

この例では、クライアントのホスト名は myhost で、IP アドレスは 192.9.200.1 です。他のホストが myhost という名前を持ち、ファイルシステムをマウントしていると、myhost というホスト名に対するシンボリックリンクは 2 つ作成されます。

NFS コマンド

以下のコマンドは、root 権限で実行しなければ、十分な効果がでません。しかし情報の要求は、すべてのユーザが行えます。

- 48ページの「clear_locks」
- 48ページの「mount」
- 53ページの「mountall」
- 62ページの「setmnt」
- 54ページの「share」
- 61ページの「shareall」

- 61ページの「showmount」
- 52ページの「umount」
- 54ページの「umountall」
- 60ページの「unshare」
- 61ページの「unshareall」

clear_locks

このコマンドを使うと、ある NFS クライアントのファイル、レコード、または共有のロックをすべて削除できます。このコマンドを実行するには、スーパーユーザでなければなりません。NFS サーバから解除できるのは特定のクライアントのロックであり、NFS クライアントから解除できるのは特定のサーバ上のそのクライアントに対するロックです。次の例では、現在のシステム上の tulip という NFS クライアントに対するロックが解除されます。

```
# clear_locks tulip
```

–s オプションを指定すると、どの NFS ホストからロックを解除するかを指定できます。これは、そのロックをかけた NFS クライアントから実行しなければなりません。次の場合、クライアントによるロックが bee という名前の NFS サーバから解除されます。

```
# clear_locks -s bee
```

注意 - このコマンドは、クライアントがクラッシュしてロックを解除できないとき以外には使用しないでください。データが破壊されるのを避けるため、使用中のクライアントに関するロックは解除しないでください。

mount

このコマンドを使用すると、指定したファイルシステムをローカルかリモートで、指定したマウントポイントに添付できます。詳細については、mount(1M) のマニュアルページを参照してください。引数を指定しないと、現在ユーザのコンピュータにマウントされているファイルシステムのリストが表示されます。

Solaris の標準インストールには、さまざまな種類のファイルシステムが含まれています。すべてのファイルシステムタイプの説明は、『*Solaris* のシステム管理 *(*第 *1* 巻*)*』の「ファイルシステムのタイプ」 in *Solaris* のシステム管理 *(*第 *1* 巻*)*を参照してください。ファイルシステムの種類ごとにマニュアルページがあり、その種類に対して mount を実行するときに使用可能なオプションのリストが示されています。たとえば、NFS ファイルシステムのマニュアルページは mount_nfs(1M)、UFS ファイルシステムのマニュアルページは mount_ufs(1M) などです。

Solaris 7 では、server:/pathname という標準の構文の代わりに NFS URL を使用して NFS サーバ上のマウントするパス名を指定することが可能になりました。詳細については、21ページの「NFS URL を使用して NFS ファイルシステムをマウントする方法」を参照してください。

注意 - Solaris 2.6 以後の mount コマンドでは、無効なオプションがあっても警告されません。解釈できないオプションがあると無視されるだけです。予想外の結果が生じるのを避けるために、使用するオプションはすべて確認してください。

NFS ファイルシステムにおける mount のオプション

NFS ファイルシステムをマウントするときに –o フラグの後に指定できるオプションの一部を、以下に示します。

bg|fg

この 2 つは、マウントが失敗したときの再試行の方法を選択するオプションです。–bg オプションの場合はバックグラウンドで、–fg オプションの場合はフォアグラウンドでマウントが試みられます。デフォルトは –fg です。常に使用可能にしておく必要のあるファイルシステムに対しては –fg が適しています。この場合、マウントが完了するまで他の処理は実行できません。–bg は、マウント要求が完了しなくてもクライアントは他の処理を実行できるため、必ずしも必要でないファイルシステムに適しています。

largefiles

このオプションを使うと、Solaris 2.6 が実行されているサーバに置かれた 2 ギガバイトを超えるサイズのファイルにアクセスできるようになります。大型ファイルにアクセスできるかどうかは、サーバでしか制御できません。したがって、このオプションは NFS バージョン 3 のマウントでは無視されます。デフォルトでは、2.6 以後の UFS ファイルシステムはすべて –largefiles オプション付きでマウントさ

れます。NFS バージョン 2 プロトコルを使ったマウントでこのオプションを指定すると、エラーが発生してマウントできません。

nolargefiles

UFS マウントでこのオプションを指定すると、ファイルシステム上に大型ファイルが存在せず、この後も作成されないことが保証されます (マニュアルページの `mount_ufs(1M)` を参照してください)。大型ファイルが存在するかどうかは NFS サーバでしか制御できないため、NFS マウントを使う `–nolargefiles` にはオプションはありません。このオプションを指定してファイルシステムを NFS マウントしようとすると、エラーが発生して拒否されます。

public

このオプションを指定すると、NFS サーバにアクセスするときに必ず公共ファイルハンドルを使用するようになります。NFS サーバが公共ファイルハンドルをサポートしていれば、MOUNT プロトコルが使用されないため、マウント操作は短時間で行われます。また、MOUNT プロトコルを使用しないため、ファイアウォールを越えたマウントが可能です。

rw | ro

`–rw` オプションと `–ro` オプションは、ファイルシステムが読み書き可能と読み取り専用のどちらでマウントされるかを示します。デフォルトは読み書き可能で、これはリモートホームディレクトリやメールスプールディレクトリなどの、ユーザによる変更が必要なファイルシステムに適しています。読み取り専用オプションは、ユーザが変更してはいけないディレクトリに適しています。具体的には、マニュアルページの共有コピーなどです。

sec=*mode*

このオプションは、マウント時に使われる認証機構を指定します。*mode* の値は、表 3–2 に示したもののいずれかでなければなりません。モードは、`/etc/nfssec.conf` ファイルにも定義されます。

表 3–2 NFS セキュリティモード

モード	選択される認証サービス
krb4	Kerberos バージョン 4
none	認証なし
dh	Diffie-Hellman (DH) 認証
sys	UNIX の標準認証

soft | hard

soft オプションを指定してマウントされた NFS ファイルシステムは、サーバが応答しなくなるとエラーを返します。hard オプションが指定されていると、サーバが応答するまで再試行が続けられます。デフォルトは hard です。ほとんどのファイルシステムには hard を使用します。ソフトマウントされたファイルシステムからの値を検査しないアプリケーションが多いので、アプリケーションでエラーが発生してファイルが破壊されるおそれがあるためです。検査するアプリケーションの場合でも、ルーティングの問題などによってアプリケーションが正しい判断をできずに、ファイルが破壊されることがあります。原則として、soft は使用しないでください。hard オプションを指定した場合にファイルシステムが使えなくなると、そのファイルシステムを使うアプリケーションはファイルシステムが復旧するまでハングする可能性があります。

mount コマンドの使用

以下のコマンドのどちらも、bee サーバから NFS ファイルシステムを読み出し専用としてマウントします。

```
# mount -F nfs -r bee:/export/share/man /usr/man
```

```
# mount -F nfs -o ro bee:/export/share/man /usr/man
```

このコマンドでは –O オプションによって、/usr/man がすでにマウントされていても bee サーバのマニュアルページがローカルシステムにマウントされます。

```
# mount -F nfs -O bee:/export/share/man /usr/man
```

このコマンドでは、クライアント側障害時回避機能が使われています。

```
# mount -F nfs -r bee,wasp:/export/share/man /usr/man
```

注 - コマンド行から使用する場合、リスト内のサーバがサポートしている NFS プロトコルは同じバージョンでなければなりません。コマンド行から mount を実行するときは、バージョン 2 とバージョン 3 のサーバを混在させないでください。autofs では混在が可能なので、バージョン 2 サーバとバージョン 3 サーバの最適な組み合わせを使用できます。

---

mount コマンドで NFS URL を使用する例を示します。

```
# mount -F nfs nfs://bee//export/share/man /usr/man
```

mount コマンドに引数を指定しないと、クライアントにマウントされたファイルシステムが表示されます。

```
% mount
/ on /dev/dsk/c0t3d0s0 read/write/setuid on Tues Jan 24 13:20:47 1995
/usr on /dev/dsk/c0t3d0s6 read/write/setuid on Tues Jan 24 13:20:47 1995
/proc on /proc read/write/setuid on Tues Jan 24 13:20:47 1995
/dev/fd on fd read/write/setuid on Tues Jan 24 13:20:47 1995
/tmp on swap read/write on Tues Jan 24 13:20:51 1995
/opt on /dev/dsk/c0t3d0s5 setuid/read/write on Tues Jan 24 13:20:51 1995
/home/kathys on bee:/export/home/bee7/kathys
  intr/noquota/nosuid/remote on Tues Jan 24 13:22:13 1995
```

umount

このコマンドにより、現在マウントされているリモートファイルシステムが削除されます。umount コマンドは、テストのために –V オプションをサポートしています。また、–a オプションを使うことによって 1 度に複数のファイルシステムをアンマウントできます。–a オプションに *mount_points* を指定すると、そのファイルシステムがアンマウントされます。マウントポイントを指定しないと、/etc/mnttab のリストにあるファイルシステムのうち “required” でないものすべてのアンマウントが試みられます。“required” のファイルシステムとは、/、/usr、/var、/proc、/dev/fd、/tmp などです。

ファイルシステムがすでにマウントされていて、/etc/mnttab に項目が指定されている場合、ファイルシステムのタイプのフラグを指定する必要はありません。

ファイルシステムが使用中だと、このコマンドは実行できません。たとえば、あるユーザが cd コマンドによってファイルシステムにアクセスしていると、作業ディレクトリが他に変更されるまでそのファイルシステムは使用中となります。umount コマンドは、NFS サーバに接続できないと一時的にハングすることがあります。

umount コマンドの使用

以下の例では、/usr/man にマウントしたファイルシステムのマウントが解除されます。

```
# umount /usr/man
```

以下の例では、umount –a -V の実行結果が表示されます。

```
# umount -a -V
umount /home/kathys
umount /opt
umount /home
umount /net
```

このコマンドでは、ファイルシステムのアンマウント自体は実行されないことに注意してください。

mountall

このコマンドを使用すると、ファイルシステムテーブルに指定したすべてのファイルシステム、または特定グループのファイルシステムをマウントできます。アクセスするファイルシステムタイプを選択するための –F *FSType* オプション、ファイルシステムテーブル内のリモートファイルシステムをすべて選択する –r オプション、ローカルファイルシステムをすべて選択する –l オプションがあります。 NFS ファイルシステムタイプと指定されているファイルシステムはすべてリモートファイルシステムなので、これらのオプションは余分な指定になることがあります。詳細については、マニュアルページの mountall(1M) を参照してください。

mountall コマンドの使用

以下の 2 つの例を実行すると、同じ結果になります。

```
# mountall -F nfs
```

```
# mountall -F nfs -r
```

umountall

このコマンドを使用すると、ファイルシステムのグループをアンマウントできます。–k オプションは、*mount_point* に結び付けられているプロセスを終了させるには fuser -k *mount_point* コマンドを使う必要があることを表します。–s オプションは、アンマウントを並行処理しないことを示します。–l は、ローカルファイルシステムだけを使用することを、–r はリモートファイルシステムだけを使用することを示します。-h *host* オプションは、指定されたホストのファイルシステムをすべてアンマウントすることを指定します。-h オプションは、-l または -r とは同時に指定できません。

umountall コマンドの使用

以下のコマンドでは、リモートホストからマウントしたすべてのファイルシステムが切り離されます。

```
# umountall -r
```

以下のコマンドでは、bee サーバからマウントしたすべてのファイルシステムが切り離されます。

```
# umountall -h bee
```

share

このコマンドを使用すると、NFS サーバのローカルファイルシステムをマウントできるようになります。また、システム上のファイルシステムのうち、現在共有しているもののリストを表示します。NFS サーバが動作していないと、share コマンドは使用できません。NFS サーバソフトウェアは、/etc/dfs/dfstab に項目がある場合、ブートの途中で自動的に起動されます。NFS サーバソフトウェアが動作していないくても、このコマンドはエラーを表示しません。NFS サーバソフトウェアが動作していることを確認してからこのコマンドを使用するようにしてください。

ディレクトリツリーはすべて共有できるオブジェクトですが、各ファイルシステムの階層構造は、そのファイルシステムが位置するディスクスライスやパーティションで制限されます。たとえばルート (/) ファイルシステムを共有しても、/usr が同じディスクパーティションかスライスに存在しなければ、/usr を共有することはできません。通常、ルートはスライス 0 に、/usr はスライス 6 にインストールされま

す。また /usr を共有しても、/usr のサブディレクトリにマウントされている ローカルディスクパーティションは共有できません。

すでに共有している大きいファイルシステムの一部であるファイルシステムを共有することはできません。たとえば /usr と /usr/local が同じディスクスライスにある場合、/usr も /usr/local も共有することができますが、両方を別々の共有オプションで共有する場合、/usr/local は別のディスクスライスに移動しなければなりません。

警告 - 2 つのファイルシステムが同じディスクスライスにある場合、読み出し専用で共有しているファイルシステムに、読み出しと書き込みが可能な状態で共有しているファイルシステムのファイルハンドルでアクセスすることができます。読み出しと書き込みの両方を行うファイルシステムは、読み出し専用で共有する必要があるファイルシステムとは別のパーティションかディスクスライスに保存するほうが安全です。

share オプション

-o フラグに指定できるオプションの一部を示します。

rw | ro

pathname に指定したファイルシステムを、読み出しと書き込みの両方が可能な状態で共有するか、読み出し専用で共有するかを指定します。

rw=*accesslist*

ファイルシステムは、リスト上のクライアントに対してだけ読み書き可能で共有されます。それ以外の要求は拒否されます。*accesslist* に定義されるクライアントのリストは、Solaris 2.6 から拡張されました。詳細については、58ページの「share コマンドを使ってアクセスリストを設定する」 を参照してください。このオプションは –ro オプションよりも優先されます。

NFSファイルシステムで指定できるオプションは、以下のとおりです。

aclok

このオプションを指定すると、NFS バージョン 2 プロトコルをサポートしている NFS サーバが NFS バージョン 2 クライアントのアクセス制御を行うように設定できます。このオプションを指定しないと、すべてのクライアントは最低限のアクセ

スしかできません。指定すると、最大限のアクセスができるようになります。たとえば –aclok オプションを指定して共有したファイルシステムでは、1 人のユーザが読み取り権を持っていれば全員が読み取りを許可されます。このオプションを指定しないと、アクセス権を持つべきクライアントからのアクセスが拒否される可能性があります。アクセス権の与えすぎと制限しすぎのどちらを選ぶかは、現在のセキュリティシステムによって決定します。アクセス制御リスト (ACL) について詳細は、『*Solaris* のシステム管理 *(*第 *2* 巻*)*』の「ファイルのセキュリティの適用手順」 in *Solaris* のシステム管理 *(*第 *2* 巻*)*を参照してください。

注 - ACL を活用するためには、クライアントでもサーバでも NFS バージョン 3 と NFS_ACL プロトコルをサポートしているソフトウェアを実行します。NFS バージョン 3 プロトコルしかサポートしていないソフトウェアの場合、クライアントは正しいアクセス権を取得できますが、ACL を操作することはできません。NFS_ACL プロトコルをサポートしていれば、正しいアクセス権を取得した上で ACL の操作も可能です。この両方をサポートしているのは、Solaris 2.5 およびその互換バージョンです。

---

anon=*uid*

uid は、認証されていないユーザのユーザ ID を選択するために使用します。*uid* を -1 に設定すると、認証されていないユーザからのアクセスは拒否されます。anon=0 とするとルートアクセス権を与えることができますが、これは認証されていないユーザにルートアクセス権を与えることになるため、代わりに root オプションを使ってください。

index=*filename*

–index=*filename* オプションを使うと、ユーザが NFS URL にアクセスするとディレクトリのリストが表示されるのではなく、HTML (HyperText Markup Language) ファイルが強制的に読み込まれます。これは、HTTP URL がアクセスしているディレクトリに index.html ファイルが見つかるとブラウザのような動作をするというものです。このオプションを設定することは、httpd に対して DirectoryIndex オプションを指定するのと同じ意味があります。たとえば、dfstab ファイルのエントリが次のとおりであるとします。

```
share -F nfs -o ro,public,index=index.html /export/web
```

このとき、以下の URL によって表示される情報はすべて同じです。

```
nfs://<server>/<dir>
nfs://<server>/<dir>/index.html
```

```
nfs://<server>//export/web/<dir>
nfs://<server>//export/web/<dir>/index.html
http://<server>/<dir>
http://<server>/<dir>/index.html
```

nosuid

このオプションを使うと、setuid モードまたは setgid モードを有効にしようとしても無視されます。NFS クライアントは、setuid か setgid のビットがオンの状態ではファイルを作成できません。

public

–public オプションは、WebNFS ブラウズのために追加されました。このオプションで共有できるのは、1 台のサーバにつき 1 つのファイルシステムだけです。

root=*accesslist*

サーバが、リスト上のホストに対してルートアクセス権を与えます。デフォルトでは、サーバはどのリモートホストにもルートアクセス権は与えません。選択されているセキュリティモードが –sec=sys 以外だと、accesslist に指定できるホストはクライアントだけです。*accesslist* に定義されたクライアントのリストは、Solaris 2.6 で拡張されました。詳細については、58ページの「share コマンドを使ってアクセスリストを設定する」 を参照してください。

注意 - 他のホストにルートアクセス権を与えるには、広い範囲でセキュリティが保証されていることが前提です。–root= option は十分慎重に使用してください。

sec=*mode*[:*mode*]

mode は、ファイルシステムへのアクセス権を取得するために必要なセキュリティモードです。デフォルトのセキュリティモードは、UNIX の認証です。モードは複数指定できますが、コマンド行に指定するときは 1 行につき 1 つのセキュリティモードだけにしてください。–mode の各オプションは、次に –mode が出現するまでその後の –rw、–ro、–rw=、–ro=、–root=、–window= オプションに適用されます。–sec=none とすると、すべてのユーザがユーザ nobody にマップされます。

window=*value*

value は、NFS サーバで資格が有効な時間の上限です。デフォルトは 30000 秒 (8.3 時間) です。

share コマンドを使ってアクセスリストを設定する

リリース 2.6 より前の Solaris で、share コマンドの –ro=、–rw=、–root= オプションに指定する *accesslist* の内容は、ホスト名がネットグループ名に限定されていました。Solaris 2.6 以降では、このアクセス制御リストにドメイン名、サブネット番号、およびアクセス権を与えないエントリも指定できます。この拡張により、名前空間を変更したり多数のクライアントを定義したリストを使うことなく、サーバでのファイルアクセス制御を今までより簡単に管理できます。

以下のコマンドでは、rose と lilac では読み出しと書き込みの両方のアクセスが認められますが、その他では、読み出しのみが許可されます。

```
# share -F nfs -o ro,rw=rose:lilac /usr/src
```

以下の例では、eng ネットグループのすべてのホストで読み出しのみができるようになります。rose クライアントでは、読み出しと書き込みの両方ができます。

```
# share -F nfs -o ro=eng,rw=rose /usr/src
```

注 - rw と ro には必ず引数が必要です。読み書き可能オプションを指定しないと、デフォルトによってすべてのクライアントが読み書き可能になります。

---

1 つのファイルシステムを複数クライアントで共有するには、すべてのオプションを同じ行に指定しなければなりません。同じオブジェクトに share コマンドを複数回実行しても、最後のコマンドしか有効になりません。以下のコマンドでは、3 つのクライアントシステムで読み出しと書き込みができますが、rose と tulip では、ファイルシステムに root でアクセスできます。

```
# share -F nfs -o rw=rose:lilac:tulip,root=rose:tulip /usr/src
```

複数の認証機構を使ってファイルシステムを共有するときには、セキュリティモードの後に必ず –ro、–ro=、–rw、–rw=、–root、–window の各オプションを指定してください。この例では、eng というネットグループ内のすべてのホストに対して UNIX 認証が選択されています。これらのホストは、ファイルシステムを読み取り

専用モードでしかマウントできません。ホスト tulip と lilac は、Diffie-Hellman (DH) 認証を使えば読み書き可能でファイル・システムをマウントできます。tulip と lilac は、そのホスト名が eng ネットグループのリストに含まれていれば、DH 認証を使っていなくても読み取り専用でマウントすることは可能です。

```
# share -F nfs -o sec=dh,rw=tulip:lilac,sec=sys,ro=eng /usr/src
```

UNIX 認証はデフォルトのセキュリティモードですが、–sec を指定するとデフォルトは無効になります。他の認証機構とともに UNIX 認証も使う場合には、必ず –sec=sys オプションを指定してください。

実際のドメイン名の名前にドットを付けると、アクセスリストの中で DNS ドメイン名が使えます。ドットは、その後の文字列が完全に修飾されたホスト名ではなくドメイン名であることを表します。次のエントリは、マウントから eng.sun.com ドメイン内のすべてのホストへのアクセスを許可するためのものです。

```
# share -F nfs -o ro=.:.eng.sun.com /export/share/man
```

この例で、“.” はそれぞれ NIS または NIS+ 名前空間を通じて一致するすべてのホストに対応します。ネームサービスから返される結果にはドメイン名は含まれません。“.eng.sun.com” というエントリは、名前空間の解決に DNS を使用するすべてのホストに一致します。DNS が返すホスト名は必ず完全に修飾されるので、DNS と他の名前空間を組み合わせると長いエントリが必要です。

実際のネットワーク番号かネットワーク名の前に “@” を指定すると、アクセスリストの中でサブネット番号が使えます。これは、ネットワーク名をネットグループ名や完全に修飾されたホスト名と区別するためです。サブネットは、/etc/networks の中か NIS または NIS+ 名前空間の中で識別できなければなりません。以下のエントリは、サブネット 129.144 が eng ネットワークと識別されるならばすべて同じ意味を持ちます。

```
# share -F nfs -o ro=@eng /export/share/man
# share -F nfs -o ro=@129.144 /export/share/man
# share -F nfs -o ro=@129.144.0.0 /export/share/man
```

2 番めと 3 番めのエントリは、ネットワークアドレス全体を指定する必要がないことを表しています。

ネットワークアドレスの先頭部分がバイトによる区切りでなく、CIDR (Classless Inter-Domain Routing) のようになっている場合には、マスクの長さをコマンド行で

具体的に指定できます。この長さは、ネットワーク名かネットワーク番号の後ろにスラッシュで区切ってアドレスの接頭辞に有効ビット数として指定します。たとえば、次のようにします。

```
# share -f nfs -o ro=@eng/17 /export/share/man
# share -F nfs -o ro=@129.144.132/17 /export/share/man
```

この例で、"/17" はアドレスの先頭から 17 ビットがマスクとして使われることを表します。CIDR について詳細は、RFC 1519 を参照してください。

また、エントリの前に "-" を指定することでアクセスの拒否を示すこともできます。エントリは左から右に読み込まれるため、アクセス拒否のエントリはそれを適用するエントリの前に置く必要があることに注意してください。

```
# share -F nfs -o ro=-rose:.eng.sun.com /export/share/man
```

この例では、eng.sun.com ドメイン内のホストのうち、rose を除いたすべてに対してアクセス権が許可されます。

unshare

このコマンドを使用すると、以前に使用可能な状態になっていたファイルシステムを、クライアントがマウントできないようにします。unshare コマンドを使用すると、share コマンドで共有したファイルシステムや、/etc/dfs/dfstab で自動的に共有しているファイルシステムが共有できないようになります。unshare コマンドを使用し、dfstab ファイルで共有しているファイルシステムの共有を解除する場合は、実行レベル 3 を終了して再度実行レベル 3 に戻ると、そのファイルシステムがまた共有されることに注意してください。実行レベル 3 を終了しても変更内容を継続させるには、そのファイルシステムを dfstab ファイルから削除しなければなりません。

NFSファイルシステムの共有を解除している場合、クライアントから既存マウントへのアクセスは禁止されます。クライアントにはファイルシステムがまだマウントされている可能性がありますが、ファイルにはアクセスできません。

unshare コマンドの使用

以下のコマンドでは、指定したファイルシステムの共有が解除されます。

```
# unshare /usr/src
```

shareall

このコマンドを使用すると、複数のファイルシステムを共有することができます。オプションなしで使用すると、/etc/dfs/dfstab 内のすべてのエントリが共有されます。share コマンドを並べたファイルの名前を指定することができます。ファイル名を指定しないと、/etc/dfs/dfstab の内容が検査されます。"-" を使ってファイル名を置き換えれば、標準入力から share コマンドを入力できます。

shareall コマンドの使用

以下のコマンドでは、ローカルファイルに羅列されているすべてのファイルシステムが共有されます。

```
# shareall /etc/dfs/special_dfstab
```

unshareall

このコマンドを使用すると、現在共有されているリソースがすべて使用できなくなります。–F *FSType* オプションによって、/etc/dfs/fstypes に定義されているファイルシステムタイプのリストを選択します。このフラグによって、特定のタイプのファイルシステムだけを共有解除できます。デフォルトのファイルシステムタイプは、/etc/dfs/fstypes に定義されています。特定のファイルシステムを選択するには、unshare コマンドを使います。

unshareall コマンドの使用

次の例では、NFS タイプのすべてのファイルシステムの共有が解除されます。

```
# unshareall -F nfs
```

showmount

このコマンドを使用すると、NFS サーバから共有したファイルシステムをリモートにマウントしたすべてのクライアントや、クライアントがマウントしたファイルシステム、または共有しているファイルシステムとクライアントのアクセス情報が表示されます。構文は以下のとおりです。

showmount [-ade] [*hostname*]

-a を指定すると、すべてのリモートマウント (クライアント名とディレクトリ) のリストが表示されます。-d を指定すると、クライアントがリモートにマウントしたディレクトリのリストが表示されます。-e では、共有 (またはエクスポート) しているファイルのリストが表示されます。*hostname* には、表示する情報が保存されているNFSサーバを指定します。*hostname* を指定しないと、ローカルホストを入力するように要求されます。

showmount コマンドの使用

以下のコマンドでは、すべてのクライアント、およびマウントしたディレクトリが表示されます。

```
# showmount -a bee
lilac:/export/share/man
lilac:/usr/src
rose:/usr/src
tulip:/export/share/man
```

以下のコマンドでは、マウントしたディレクトリが表示されます。

```
# showmount -d bee
/export/share/man
/usr/src
```

以下のコマンドでは、共有しているファイルシステムが表示されます。

```
# showmount -e bee
/usr/src        (everyone)
/export/share/man     eng
```

setmnt

このコマンドを使用すると、/etc/mnttab テーブルが作成されます。このテーブルは、mount コマンドと umount コマンドで参照されます。通常、このコマンドを使用することはありません。システムがブートされるときに自動的に使用されます。

その他のコマンド

NFS の障害追跡には以下のコマンドを使用します。

- 63ページの「nfsstat」
- 64ページの「pstack」
- 65ページの「rpcinfo」
- 67ページの「snoop」
- 67ページの「truss」

nfsstat

このコマンドを使用すると、NFS と RPC 接続について統計情報を収集できます。構文は次のとおりです。

```
nfsstat [ -cmnrsz ]
```

–c を指定すると、クライアント側の情報が表示され、–m を指定すると、マウントした各 NFS ファイルシステムの統計が表示されます。–n では、クライアントとサーバの両方の NFS 情報が表示され、–r では、RPC の統計が表示されます。–s を指定すると、サーバ側の情報が表示され、–z を指定すると、統計がゼロに設定されます。コマンド行にオプションを指定しないと、–cnrs が使用されます。

新しいソフトウェアやハードウェアを処理環境に追加した場合、サーバ側の統計を収集することが、デバッグにたいへん役立ちます。このコマンドを週に最低 1 度は実行し、履歴を作成するようにしてください。統計を保存しておくと、以前の効率のよい記録になります。

nfsstat コマンドの使用

```
# nfsstat -s

Server rpc:
Connection oriented:
calls      badcalls   nullrecv   badlen      xdrcall    dupchecks  dupreqs
11420263   0          0          0           0          1428274    19
Connectionless:
calls      badcalls   nullrecv   badlen      xdrcall    dupchecks  dupreqs
14569706   0          0          0           0          953332     1601

Server nfs:
```

```
calls      badcalls
24234967   226
Version 2: (13073528 calls)
null       getattr    setattr    root       lookup     readlink   read
138612 1%  1192059 9% 45676 0%   0 0%       9300029 71% 9872 0%   1319897 10%
wrcache    write      create     remove     rename     link       symlink
0 0%       805444 6%  43417 0%   44951 0%   3831 0%    4758 0%    1490 0%
mkdir      rmdir      readdir    statfs
2235 0%    1518 0%    51897 0%   107842 0%
Version 3: (11114810 calls)
null       getattr    setattr    lookup     access     readlink   read
141059 1%  3911728 35% 181185 1% 3395029 30% 1097018 9% 4777 0%   960503 8%
write      create     mkdir      symlink    mknod      remove     rmdir
763996 6%  159257 1%  3997 0%    10532 0%   26 0%      164698 1%  2251 0%
rename     link       readdir    readdirplus fsstat    fsinfo     pathconf
53303 0%   9500 0%    62022 0%   79512 0%   3442 0%    34275 0%   3023 0%
commit
73677 0%

Server nfs_acl:
Version 2: (1579 calls)
null       getacl     setacl     getattr    access
0 0%       3 0%       0 0%       1000 63%   576 36%
Version 3: (45318 calls)
null       getacl     setacl
0 0%       45318 100% 0 0%
```

上記は、NFS サーバの統計です。最初の 5 行は RPC に関するもので、残りの部分は NFS のアクティビティのレポートです。どちらの統計でも総コール数に対する badcallsi の数や 1 週間あたりの calls 数がわかるので、障害が発生した時点を突き止めのに役立ちます。badcallsiの値は、クライアントからの不良メッセージの数を表すもので、ネットワーク上のハードウェアにおける問題を突き止められます。

いくつかの接続では、ディスクに対する書き込みアクティビティが発生します。この数値の急激な上昇は障害の可能性を示すものなので、調査が必要です。NFS バージョン 2 の場合、特に注意しなければならない接続は、setattr、write、create、remove、rename、link、symlink、mkdir、および rmdir です。 NFS バージョン 3 の場合には、commit の値に特に注意します。ある NFSサーバの commit レベルが、それと同等のサーバと比較して高い場合は、NFS クライアントに十分なメモリがあるかどうかを確認してください。サーバの commit オペレーションの数は、クライアントにリソースがない場合に上昇します。

pstack

このコマンドを使用すると、各プロセスにおけるスタックトレースが表示されます。root で実行しなければなりません。プロセスがハングした場所を判断するのに

使用します。使用できるオプションは、チェックするプロセスの PID だけです (proc(1) のマニュアルページ参照)。

以下の例では、実行中の nfsd プロセスをチェックしています。

```
# /usr/proc/bin/pstack 243
243:    /usr/lib/nfs/nfsd -a 16
 ef675c04 poll     (24d50, 2, ffffffff)
 000115dc ???????? (24000, 132c4, 276d8, 1329c, 276d8, 0)
 00011390 main     (3, effffff14, 0, 0, ffffffff, 400) + 3c8
 00010fb0 _start   (0, 0, 0, 0, 0, 0) + 5c
```

プロセスが新規の接続要求を待っていることが示されています。これは、正常な反応です。要求が行われた後でもプロセスがポーリングしていることがスタックから分かった場合、そのプロセスはハングしている可能性があります。34ページの「NFS サービスの再起動」 の指示に従って問題を解決してください。ハングしたプログラムによって問題が発生しているかどうかを確実に判断するには、29ページの「NFS の障害追跡手順」 を参照してください。

rpcinfo

このコマンドは、システムで動作している RPC サービスに関する情報を生成します。RPC サービスの変更にも使用できます。このコマンドには、たくさんのオプションがあります (rpcinfo(1M) のマニュアルページ参照)。以下は、このコマンドで使用できるオプションの概要です。

rpcinfo [–m | –s] [*hostname*]

rpcinfo [–t | –u] [*hostname*] [*progname*]

–m は rpcbind 操作の統計テーブル、–s は登録済みの RPC プログラムすべての簡易リスト、–t は TCP を使う RPC プログラム、–u は UDP を使う RPC プログラムを表示します。*hostname* は情報を取得する元のサーバ、*progname* は情報を収集する対象の RPC プログラムです。*hostname* を指定しないと、ローカルホスト名が使われます。*progname* の代わりに RPC プログラム番号が使えますが、ユーザが覚えやすいのは番号よりも名前です。NFS バージョン 3 が実行されていないシステムでは、–s オプションの代わりに –p オプションが使えます。

このコマンドで生成されるデータには、以下のものがあります。

- RPC プログラム番号
- 特定プログラムのバージョン番号

- 使用されているトランスポートプロトコル
- RPC サービスの名前
- RPC サービスの所有者

rpcinfo コマンドの使用

以下の例では、サーバで実行している RPC サービスに関する情報を収集しています。生成されたテキストには sort コマンドのフィルタをかけ、より読みやすくしています。この例では、RPC サービスの数行を省略しています。

```
% rpcinfo -s bee |sort -n
   program version(s) netid(s)                          service     owner
    100000  2,3,4     udp,tcp,ticlts,ticotsord,ticots  portmapper  superuser
    100001  4,3,2     ticlts,udp                       rstatd      superuser
    100002  3,2       ticots,ticotsord,tcp,ticlts,udp  rusersd     superuser
    100003  3,2       tcp,udp                          nfs         superuser
    100005  3,2,1     ticots,ticotsord,tcp,ticlts,udp  mountd      superuser
    100008  1         ticlts,udp                       walld       superuser
    100011  1         ticlts,udp                       rquotad     superuser
    100012  1         ticlts,udp                       sprayd      superuser
    100021  4,3,2,1   ticots,ticotsord,ticlts,tcp,udp  nlockmgr    superuser
    100024  1         ticots,ticotsord,ticlts,tcp,udp  status      superuser
    100026  1         ticots,ticotsord,ticlts,tcp,udp  bootparam   superuser
    100029  2,1       ticots,ticotsord,ticlts          keyserv     superuser
    100068  4,3,2     tcp,udp                          cmsd        superuser
    100078  4         ticots,ticotsord,ticlts          kerbd       superuser
    100083  1         tcp,udp                          -           superuser
    100087  11        udp                              adm_agent   superuser
    100088  1         udp,tcp                          -           superuser
    100089  1         tcp                              -           superuser
    100099  1         ticots,ticotsord,ticlts          pld         superuser
    100101  10        tcp,udp                          event       superuser
    100104  10        udp                              sync        superuser
    100105  10        udp                              diskinfo    superuser
    100107  10        udp                              hostperf    superuser
    100109  10        udp                              activity    superuser
  .
  .
    100227  3,2       tcp,udp                          -           superuser
    100301  1         ticlts                           niscachemgr superuser
    390100  3         udp                              -           superuser
1342177279  1,2       tcp                              -           14072
```

次の例では、サーバの特定トランスポートを使用している RPC サービスの情報を収集する方法について説明しています。

```
% rpcinfo -t bee mountd
program 100005 version 1 ready and waiting
program 100005 version 2 ready and waiting
program 100005 version 3 ready and waiting
```

(続く)

```
% rpcinfo -u bee nfs
program 100003 version 2 ready and waiting
program 100003 version 3 ready and waiting
```

最初の例では、TCP で実行している mountd サービスをチェックしています。2 番目の例では、UDP で実行している NFS サービスをチェックしています。

snoop

このコマンドは、ネットワーク上のパケットの監視によく使用されます。root として実行しなければなりません。クライアントとサーバの両方で、ネットワークハードウェアが機能しているかどうかを確認する方法としてよく使用されます。使用できるオプションは多数あります (snoop(1M) のマニュアルページ参照)。以下で、このコマンドの概要を説明します。

snoop [-d *device*] [–o *filename*] [host *hostname*]

–d *device* には、ローカルネットワークインタフェースを指定します。-o *filename* には、取り込んだすべてのパケットを保存するファイルを指定します。*hostname* には、表示するパケットが通過したホストを指定します。

–d *device* オプションは、複数のネットワークインタフェースがあるサーバで特に有効です。ホストの設定以外にも、使用できる式が多数あります。コマンド式を grep で組み合わせることでも、十分に使用できるデータを生成できます。

障害追跡をする場合は、パケットの発信元と送信先のホストが正しいことを確認してください。また、エラーメッセージも調べてください。パケットをファイルに保存すると、データの検査が容易になります。

truss

このコマンドを使用すると、プロセスがハングしたかどうかを確認できます。root で実行しなければなりません。このコマンドに指定できるオプションは多数あります (truss(1) のマニュアルページ参照)。構文の概要は以下のとおりです。

truss [-t *syscall*] –p *pid*

–t *syscall* には、追跡するシステムコールを指定します。–p *pid* には、追跡するプロセスの PID を指定します。*syscall* には、追跡するシステムコールをカンマで区切って指定することもできます。また、*syscall* の指定を ! で始めると、そのシステムコールは追跡されなくなります。

次の例は、プロセスが新しいクライアントからの接続要求を待っていることを示しています。

```
# /usr/bin/truss -p 243
poll(0x00024D50, 2, -1)         (sleeping...)
```

これは正常な反応です。新規接続の要求が行われた後でも反応が変わらない場合、そのプロセスはハングしている可能性があります。34ページの「NFS サービスの再起動」 の指示に従ってプログラムを修正してください。ハングしたプログラムによって問題が発生しているかどうかを確実に判断するには、29ページの「NFS の障害追跡手順」 を参照してください。

コマンドを組み合わせて使う

以下の節では、NFS の複雑な機能をいくつか紹介します。

バージョン 2 とバージョン 3 のネゴシエーション

NFS サーバがサポートしているクライアントが NFS バージョン 3 を使っていない場合に備えて、開始手順にはプロトコルレベルのネゴシエーションが含まれています。クライアントとサーバの両方がバージョン 3 をサポートしていると、バージョン 3 が使われます。どちらか片方でもバージョン 2 しかサポートしていないと、バージョン 2 が使われます。

ネゴシエーションによって決まった値は、mount コマンドに対して –vers オプションを使うことで変更できます (マニュアルページの mount_nfs(1M) を参照してください)。ほとんどの場合、デフォルトによって最適なバージョンが選択されるため、ユーザが指定する必要はありません。

UDP と TCP のネゴシエーション

開始時には、トランスポートプロトコルもネゴシエーションされます。デフォルトでは、クライアントとサーバの両方がサポートしているコネクション型トランスポートの中で最初に見つかったものが選択されます。それが見つからない場合には、コネクションレス型トランスポートプロトコルの中で最初に見つかったものが使われます。システムでサポートされているトランスポートプロトコルのリストは、`/etc/netconfig` にあります。TCP はコネクション型トランスポートプロトコルで、Solaris 2.6 でサポートされています。UDP はコネクションレス型トランスポートプロトコルです。

NFS プロトコルのバージョンとトランスポートプロトコルが両方ともネゴシエーションによって決まった場合は、NFS プロトコルのバージョンがトランスポートプロトコルよりも優先されます。UDP を使う NFS バージョン 3 プロトコルの方が、TCP を使う NFS バージョン 2 プロトコルよりも優先されます。`mount` コマンドでは NFS プロトコルのバージョンもトランスポートプロトコルも手動で選択できます (マニュアルページの `mount_nfs(1M)` を参照してください)。ほとんどの場合、ネゴシエーションによって選択されるオプションの方が適切です。

ファイル転送サイズのネゴシエーション

ファイル転送サイズは、クライアントとサーバの間でデータを転送するときに使われるバッファのサイズです。原則として、ファイル転送サイズが大きいほどパフォーマンスが向上します。NFS バージョン 3 には転送サイズに上限はありませんが、Solaris 2.6 以降がデフォルトで提示するバッファサイズは 32 キロバイトです。クライアントは、必要であればマウント時にこれより小さい転送サイズを提示することができますが、ほとんどの場合必要ありません。

転送サイズは、NFS バージョン 2 を使っているシステムとはネゴシエーションされません。このとき、ファイル転送サイズの上限は 8 キロバイトに設定されます。

`mount` コマンドに対して `–rsize` オプションと `–wsize` オプションを使うと、転送サイズを手動で設定できます。PC クライアントの一部では転送サイズを小さくする必要があります。また、NFS サーバが大きなファイル転送サイズに設定されている場合には、転送サイズを大きくすることができます。

ファイルシステムのマウントの詳細

クライアントがサーバからファイルシステムをマウントするとき、そのファイルシステムに対応するファイルハンドルをサーバから取得する必要があります。そのためには、クライアントとサーバの間でいくつかのトランザクションが発生します。この例では、クライアントはサーバから /home/terry をマウントします。snoop によって追跡したトランザクションは、次のとおりです。

```
client -> server PORTMAP C GETPORT prog=100005 (MOUNT) vers=3 proto=UDP
server -> client PORTMAP R GETPORT port=33492
client -> server MOUNT3 C Null
server -> client MOUNT3 R Null
client -> server MOUNT3 C Mount /export/home9/terry
server -> client MOUNT3 R Mount OK FH=9000 Auth=unix
client -> server PORTMAP C GETPORT prog=100003 (NFS) vers=3 proto=TCP
server -> client PORTMAP R GETPORT port=2049
client -> server NFS C NULL3
server -> client NFS R NULL3
client -> server NFS C FSINFO3 FH=9000
server -> client NFS R FSINFO3 OK
client -> server NFS C GETATTR3 FH=9000
server -> client NFS R GETATTR3 OK
```

この追跡結果では、クライアントがまずマウントポート番号を NFS サーバの portmap サービスに要求します。クライアントが取得したマウントポート番号 (33492) は、サーバに対する存在確認のために使用されます。このポート番号でサービスが実行中であることが確認できると、クライアントはマウントを要求します。この要求により、サーバはマウントされるファイルシステムに対するファイルハンドル (9000) を送ります。これに対してクライアントは、NFS ポート番号を要求します。クライアントはサーバからポート番号を受け取り、NFS サービス (nfsd) を ping してから、ファイルハンドルを使用してファイルシステムに関する NFS 情報を要求します。

次の追跡結果では、クライアントは –public オプションを使用してファイルシステムをマウントしています。

```
client -> server NFS C LOOKUP3 FH=0000 /export/home9/terry
server -> client NFS R LOOKUP3 OK FH=9000
client -> server NFS C FSINFO3 FH=9000
server -> client NFS R FSINFO3 OK
client -> server NFS C GETATTR3 FH=9000
server -> client NFS R GETATTR3 OK
```

デフォルトの公共ファイルハンドル (0000) を使用しているために、すべてのトランザクションにポートマップサービスから情報が与えられ、NFS ポート番号を決定するためのトランザクションはありません。

マウント時の –public オプションと NFS URL の意味

–public オプションを使用すると、マウントが失敗することがあります。NFS URL を組み合わせると、状況がさらに複雑になる可能性があります。これらのオプションを使用した場合にファイルシステムがどのようにマウントされるかは、次のとおりです。

public オプションと **NFS URL** - 公共ファイルハンドルが使用されます。公共ファイルハンドルがサポートされていないと、マウントは失敗します。

public オプションと通常のパス - 公共ファイルハンドルが使用されます。公共ファイルハンドルがサポートされていないと、マウントは失敗します。

NFS URL のみ - NFS サーバでサポートされていれば、公共ファイルハンドルを使用します。公共ファイルハンドルを使用するとマウントが失敗する場合は、MOUNT プロトコルを使用してマウントします。

通常のパスのみ - 公共ファイルハンドルは使用しないでください。MOUNT プロトコルが使用されます。

クライアント側障害時回避機能

クライアント側障害時回避機能を使うと、複製されたファイルシステムをサポートしているサーバが使用不能になったときに、NFS クライアントは別のサーバに切り替えることができます。ファイルシステムが使用不能になる原因としては、接続しているサーバのクラッシュ、サーバの過負荷、ネットワーク障害が考えられます。通常、このような場合の障害時回避機能はユーザには分かりません。設定が行われていれば、障害時回避機能はクライアント上のプロセスを中断することなく実行されます。

障害時回避機能が行われるためには、ファイルシステムが読み取り専用でマウントされている必要があります。また、ファイルシステムが完全に同じでないと障害時回避機能は成功しません。ファイルシステムが同一になる条件については、72ページの「複製されたファイルシステムとは」 を参照してください。障害時回避機能の候補としては、静的なファイルシステム、または変更の少ないファイルシステムが適しています。

CacheFS を使ってマウントされたファイルシステムは、障害時回避機能には使えません。CacheFS ファイルシステムは、それぞれについて追加情報が格納されています。この情報は障害時回避の際に更新できないため、ファイルシステムをマウントするときには障害時回避機能と CasheFS のどちらか片方の機能しか使えません。

各ファイルシステムについて用意すべき複製の数を決める要素はさまざまです。一般的に、サーバを何台か用意してそれぞれが複数のサブネットをサポートするという環境の方が、サブネット 1 つについて 1 台のサーバを用意するよりもすぐれています。この場合、リストにあるサーバを 1 台ずつチェックする必要があるため、リスト上のサーバが増えるにつれてマウントにかかる時間も増えます。

障害時回避機能に関する用語

障害時回避機能のプロセスを完全に理解するには、以下の 2 つの用語を理解しておく必要があります。

- 障害時回避機能 – 複製されたファイルシステムに対応するサーバのリストから、サーバを選択すること。通常、ソートされたリストの順番を元に、次のサーバが応答するならばそれが使われます。
- 再マッピング – 新しいサーバを使うこと。クライアントは、正常な状態のときにリモートファイルシステム上のアクティブなファイルそれぞれのパス名を格納します。再マッピング時には、そのパス名に基づいて新しいサーバ上のファイルを見つけます。

複製されたファイルシステムとは

障害時回避機能に関して、あるファイルシステムのすべてのファイルが元のファイルシステムのファイルとサイズも vnode タイプも同じ場合に、そのファイルシステムを「複製」といいます。アクセス権、作成日付などのファイル属性は関係ありません。ファイルサイズか vnode タイプが異なると再マッピングは失敗し、元のサーバが再び使用可能になるまでプロセスはハングします。

複製されたファイルシステムを保守するには、`rdist` や `cpio` などのファイル転送機構を使います。複製されたファイルシステムを更新すると不整合が発生するので、できるだけ以下を守ってください。

- 新しいバージョンのファイルをインストールするときは、あらかじめ古い方の名前を変更する
- クライアントによる使用が少ない夜間に更新を実行する

- 更新は小規模にとどめる
- コピーの数を最小限にする

障害時回避機能と NFS ロック

ソフトウェアパッケージの一部は、ファイルに読み取りロックをかける必要があります。そのようなソフトウェアが正常に動作できるようにするため、読み取り専用ファイルシステムに対しても読み取りロックがかけられるようになっています。ただし、これはクライアント側でしか認識されません。サーバ側で意識されないため、再マッピングされてもロックはそのまま残ります。ファイルはもともと変更が許されないので、サーバ側でファイルをロックする必要はありません。

大型ファイル

Solaris 2.6 およびその互換バージョンでは、2 ギガバイトを超えるファイルを扱えます。デフォルトでは、UFS ファイルシステムはこの新機能を活かすために `–largefiles` オプション付きでマウントされます。以前のリリースでは、2 ギガバイトを超えるファイルは扱えません。具体的な方法については 18ページの「NFS サーバ上の大型ファイルを無効にする方法」 を参照してください。

サーバのファイルシステムが `–largefiles` オプション付きでマウントされていれば、Solaris 2.6 の NFS クライアントでは何も変更しなくても大型ファイルにアクセスできます。しかし、Solaris 2.6 のコマンドすべてで大型ファイルが扱えるわけではありません。大型ファイルを処理可能なコマンドのリストは、`largefile(5)` を参照してください。大型ファイル用機能拡張を備えた NFS バージョン 3 プロトコルをサポートしていないクライアントは、大型ファイルには一切アクセスできません。Solaris 2.5 クライアントでは、NFS バージョン 3 プロトコルを使うことはできますが、大型ファイルを扱う機能は含まれていません。

WebNFS サービスの動作方法

WebNFS サービスとは、あるディレクトリに置かれたファイルを、公開ファイルハンドルを使ってクライアントからアクセスできるようにするものです。ファイルハンドルは、NFS クライアントがファイルを識別できるようにカーネルが生成するアドレスです。公開ファイルハンドル の値はあらかじめ決まっているため、サーバがクライアントに対してファイルハンドルを生成する必要はありません。定義済みのファイルハンドルを使用するというこの機能によって、`MOUNT` プロトコルが不要に

なってネットワークトラフィックが減り、クライアントにとってはパフォーマンスが向上します。

デフォルトでは、NFS サーバの公開ファイルハンドルはルートファイルシステムに対して設定されます。このデフォルトのため、サーバに対してマウント権限を持っているすべてのクライアントに対して WebNFS アクセス権が与えられます。公開ファイルハンドルは、`share` コマンドによって任意のファイルシステムに切り替えることができます。

あるファイルシステムに対するファイルハンドルをクライアントが持っているとき、アクセスするファイルに対応するファイルハンドルを知るには `LOOKUP` を実行します。 NFS プロトコルでは、パス名の構成要素を 1 度に 1 つしか評価できません。したがって、ディレクトリ階層のレベルが 1 つ増えるたびに 1 回ずつ `LOOKUP` を実行します。公開ファイルハンドルからの相対パスに対して `LOOKUP` を実行する場合には、WebNFS サーバは複数構成要素参照という方法によって 1 度にパス名全体を評価できます。複数構成要素参照を使うことにより、WebNFS サーバはパス名の中のディレクトリレベルを 1 つずつファイルハンドルに変換しなくても目的のファイルに対するファイルハンドルを取得できます。

また、NFS クライアントは単一の TCP 接続上で同時に複数のダウンロードを行うこともできます。これにより、複数の接続を設定することによる余分な負荷をサーバにかけずに、高速なアクセスが実現できます。Web ブラウザアプリケーションも複数ファイルを同時にダウンロードできますが、それぞれのファイルに独自の接続が確立されます。WebNFS ソフトウェアは接続を 1 つしか使わないため、サーバに対するオーバーヘッドを軽減できます。

パス名の中の最後の構成要素が他のファイルシステムに対するシンボリックリンクである場合、通常の NFS アクティビティによってあらかじめそのファイルへのアクセス権を持っていれば、クライアントはそのファイルにアクセスできます。

通常、NFS URL は公開ファイルハンドルからの相対位置として評価されます。パスの先頭にスラッシュを 1 つ追加すると、サーバのルートファイルシステムからの相対位置に変更できます。次の例では、公開ファイルハンドルが `/export/ftp` ファイルシステムに設定されていればこの 2 つの NFS URL は同等です。

```
nfs://server/junk
nfs://server//export/ftp/junk
```

Web ブラウザと比較した場合の WebNFS の制約

HTTP を使う Web サイトで実現可能な機能のいくつかは、WebNFS ではサポートされていません。この違いは、NFS サーバはファイルを送るだけであるため、特別な処理はすべてクライアントで行う必要があることが原因です。ある Web サイトを WebNFS と HTTP 両方のアクセスに対応させるには、以下を考慮してください。

- NFS によるブラウズでは CGI スクリプトは実行されません。したがって、CGI スクリプトを多用している Web サイトを含むファイルシステムは、NFS によるブラウズに適していない可能性があります。
- ブラウザからは、形式の異なるファイルを扱うために別のビューワを起動されることがあります。NFS URL からそうしたファイルにアクセスすると、ファイル名からファイルタイプが判別できるならば外部のビューワが起動されます。ブラウザは、NFS URL が使われている場合、標準の MIME タイプで決まっているファイル名拡張子をすべて認識します。WebNFS は一部の Web ブラウザとは異なり、ファイルタイプを決定するときにファイルの内部は調べません。ファイル名拡張子だけで判断します。
- NFS によるブラウズでは、サーバ側のイメージマップ (クリッカブルイメージ) は使えません。しかしクライアント側のイメージマップ (クリッカブルイメージ) は、位置とともに URL が定義されているため使えます。文書サーバからの応答は不要です。

Secure NFS システム

NFS 環境は、アーキテクチャやオペレーティングシステムの異なるコンピュータから構成されるネットワーク上でファイルシステムを共有するためには、強力で使いやすい手段です。しかし、NFS の操作によるファイルシステムの共有を便利にする機能が、一方ではセキュリティ上の問題につながっています。今まで、NFS はほとんどのバージョンで UNIX (AUTH_SYS) 認証を使ってきましたが、現在では AUTH_DH のようなより強力な認証方式も使用可能です。UNIX 認証の場合、NFS サーバはファイル要求を認証するために、その要求を行ったユーザではなくコンピュータを認証します。したがって、クライアント側のユーザがスーパーユーザでログインすると、ファイルの所有者になりすますことができます。DH 認証では、NFS サーバはユーザを認証するため、この様な操作が困難になります。

ルートへのアクセス権とネットワークプログラミングについての知識があれば、だれでも任意のデータをネットワークに入れ、ネットワークから任意のデータを取り出すことができます。ネットワークに対する最も危険な攻撃は、有効なパケットを

生成したり、または “対話” 対話を記録し後で再生することによってユーザを装うなどの手段により、データをネットワークに持ち込むことです。これらはデータの整合性に影響を与えます。許可を持つユーザを装うことなく、単にネットワークトラフィックを受信するだけの受動的な盗み聞きならば、データの整合性が損なわれることはないため、それほど危険ではありません。ユーザはネットワークに送信されるデータを暗号化することによって、機密情報のプライバシーを守ることができます。

ネットワークのセキュリティ問題における共通の対処方法は、解決策を各アプリケーションにゆだねることです。さらに優れた手法としては、すべてのアプリケーションを対象として、標準の認証システムを導入することです。

Solaris オペレーティングシステムには、NFS が実装されるメカニズムである遠隔手続き呼び出し (RPC) のレベルで、認証システムが組み込まれています。このシステムは Secure RPC と呼ばれ、ネットワーク環境のセキュリティを大幅に向上させるとともに、NFS のセキュリティを強化します。Secure RPC の機能を利用した NFS システムを、Secure NFS システムと呼びます。

Secure RPC

Secure RPC は Secure NFS システムの基本となるメカニズムです。Secure RPC の目標は、少なくともタイムシェアリングシステム (すべてのユーザが 1 台のコンピュータを共有するシステム) 程度に安全なシステムを構築することです。タイムシェアリングシステムはログインパスワードによりユーザを認証します。DES (Data Encryption Service) 認証でもこれは同じです。ユーザは、ローカル端末の場合と同じように、任意のリモートコンピュータにログインできます。ユーザのログインパスワードは、ネットワークセキュリティへのパスポートです。タイムシェアリングでは、システム管理者は信頼のおける人で、パスワードを変更してだれかを装うようなことはしないという道徳上の義務を負います。Secure RPC では、ネットワーク管理者は「公開鍵」を格納するデータベースのエントリを変更しないという前提で信頼されています。

RPC 認証システムを理解するには、「資格 (credential)」と「ベリファイア」という 2 つの用語を理解する必要があります。ID バッジを例にとれば、資格とは、名前、住所、誕生日など人間を識別するものです。ベリファイアとはバッジに添付された写真であり、バッジの写真をその所持者と照合することによって、そのバッジが盗まれたものではないことを確認できます。RPC では、クライアントプロセスは RPC 要求のたびに資格とベリファイアの両方をサーバに送信します。クライアントはサーバの資格をすでに知っているため、サーバはベリファイアだけを送り返します。

RPC の認証機能は拡張が可能で、さまざまな認証システムを組み込むことができます。現在のところ、このようなシステムには UNIX、DH、KERB (Kerberos バージョン 4) の 3 つがあります。

ネットワークサービスで UNIX 認証を使用する場合、資格にはクライアントのコンピュータ名、UID、GID、グループアクセスリストが含まれ、ベリファイアには何も含まれません。ベリファイアが存在しないため、ルートユーザは su などのコマンドを使用して、適切な資格を偽ることができます。UNIX 認証でのもう 1 つの問題は、ネットワーク上のすべてのコンピュータを UNIX コンピュータと想定していることです。UNIX 認証を異機種ネットワーク内のほかのオペレーティングシステムに適用した場合、これは正常に動作しません。

UNIX 認証の欠点を補うために、Secure RPC では DH 認証か KERB 認証を使います。

DH 認証

DH 認証は、Data Encryption Standard (DES) と Diffie-Hellman 公開鍵暗号手法を使用してネットワーク上のユーザとコンピュータの両方を認証します。DES は標準暗号化機能であり、Diffie-Hellman 公開鍵暗号手法は、公開鍵と非公開鍵という 2 つの鍵を使用する暗号方式です。公開鍵と非公開鍵は名前空間に格納されます。NIS の場合、それらの鍵を publickey マップに格納し、NIS+ は cred テーブルに格納します。これらのマップにはすべての認証の候補ユーザの公開鍵と非公開鍵が入っています。マップとテーブルの設定については、『*Solaris* ネーミングの管理』を参照してください。

DH 認証のセキュリティは、送信側が現在時刻を暗号化する機能に基づいていて、受信側はこれを復号して、自分の時刻と照合します。タイムスタンプは DES を使って暗号化されます。この方式が機能するには次の条件が必要です。

- 2 つのエージェントの現在時刻が一致している。
- 送信側と受信側が同じ暗号化鍵を使用する。

ネットワークが時間同期プログラムを実行する場合、クライアントとサーバ上の時間は自動的に同期されます。時間同期プログラムを使用できない場合、ネットワーク時間ではなく、サーバの時間を使用してタイムスタンプを計算できます。クライアントは、RPC セッションを開始する前にサーバに時間を要求し、自分のクロックとサーバのクロックとの時間差を計算します。タイムスタンプを計算するときには、この差を使用してクライアントのクロックを補正します。サーバがクライアントの要求を拒否するほど、クライアントとサーバのクロック同期がずれた場合、DH 認証システムはサーバとの間で再び同期をとります。

クライアントとサーバは、ランダムな対話鍵 (セッションキーとも呼びます) を生成することによって、同じ暗号化鍵に到達します。次に、公開鍵暗号手法 (公開鍵と秘密鍵を必要とする暗号化方式) を使用して共通鍵を推理します。この共通鍵は、クライアントとサーバだけが推理できる鍵です。対話鍵は、クライアントのタイムスタンプを暗号化および復号化するために使用されます。共通鍵は、この対話鍵を暗号化および復号化するために使用されます。

KERB 認証

Kerberos は MIT で開発された認証方式です。Kerberos での暗号化は DES に基づいています。

Kerberos はユーザのログインパスワードを認証することにより機能します。ユーザは `kinit` コマンドを入力しますが、このコマンドは、認証サーバからセッション時間 (またはデフォルトセッション時間の 8 時間) の間有効であるチケットを得ます。ユーザがログアウトするときに、このチケットは `kdestroy` コマンドを使って削除できます。

Kerberos ソフトウェアは、SunOS ソフトウェアの一部ではなく、 MIT の Athena プロジェクトから入手できます。SunOS ソフトウェアは次のソフトウェアを提供します。

- チケットの作成、取得、検証にクライアントが使用するルーチン
- Secure RPC の認証オプション
- クライアント側デーモン `kerbd`(1M)

詳細については、『*Solaris* のシステム管理 *(*第 *2* 巻*)*』の「Secure RPC の概要」を参照してください。

NFS での Secure RPC の使用

Secure RPC を使う場合は、以下の点に注意してください。

- サーバがクラッシュしたとき周囲に誰もいない場合 (停電の後など) には、システムに格納されていた秘密鍵はすべて消去されます。そのためどのプロセスからも、セキュリティ保護されたネットワークサービスにアクセスしたり NFS ファイルシステムをマウントしたりできません。リブートの際に重要なプロセスは、通常は root として実行されます。したがって、root の秘密鍵を別に保存してあればこれらのプロセスを実行できますが、周囲に誰もいない状況では秘密鍵を復号化するパスワードを入力するユーザがいません。`keylogin -r` を使うと

root の秘密鍵がそのまま /etc/.rootkey に格納され、keyserv がそれを読み取ります。

- システムによっては、シングルユーザモードでブートし、コンソールには root のログインシェルが表示されてパスワードの入力が要求されないことがあります。このような場合には、物理的なセキュリティが不可欠です。
- ディスクレスコンピュータのブートは、完全に安全とはいえません。ブートサーバになりすましてリモートコンピュータに対する秘密鍵の入力を記録するような、不正なカーネルを誰かがブートすることが考えられます。Secure NFS システムによって保護されているのはカーネルとキーサーバが起動した後だけです。それまでの間に、ブートサーバからの応答を認証する手段はありません。これは重大な問題につながる可能性がありますが、この部分を攻撃するにはカーネルのソースコードを使った高度な技術が必要です。また、不法行為の痕跡が残ります。すなわち、ネットワークを通じてブートサーバにポーリングすれば、不正なブートサーバの位置が分かります。
- ほとんどの setuid プログラムは root が所有者です。root の秘密鍵が /etc/.rootkey に格納されていれば、これらのプログラムは正常に動作します。しかし、ユーザが所有者である setuid プログラムは動作しない可能性があります。たとえば、ある setuid プログラムの所有者が dave であり、ブート以降 dave が 1 度もログインしていないと、このプログラムはセキュリティ保護されたネットワークサービスにはアクセスできません。
- リモートコンピュータに (login、rlogin、または telnet を使って) ログインし、keylogin を使ってアクセスすると、自分のアカウントへのアクセスを許したことになります。これは、秘密鍵が相手側のコンピュータのキーサーバに渡され、キーサーバがその秘密鍵を格納したためです。これが問題になるのは、相手側のリモートコンピュータを信用できない場合だけです。しかし、疑いがある場合にはパスワードを要求するリモートコンピュータにはログインしないでください。代わりに NFS 環境を使って、そのリモートコンピュータから共有されているファイルシステムをマウントします。または、keylogout を使ってキーサーバから秘密鍵を消去します。
- ホームディレクトリが共有されていて –o sec=dh オプションか –o sec=krb4 が指定されていると、リモートログインによって問題が生じる可能性があります。/etc/hosts.equiv ファイルか /.rhosts ファイルでパスワードを要求しないように設定すると、ユーザはログインできますが、ローカルで認証が行われていないために自分のホームディレクトリにアクセスできません。パスワードを要求され、入力したパスワードがネットワークパスワードと一致すれば自分のホームディレクトリにアクセスできます。

パートIII　autofs の詳細

パートIII、autofs サービスと autofs を使う手順について説明します。

- 84ページの「autofs の設定」
- 84ページの「一般的な作業と手順」
- 100ページの「autofs での問題発生時の対処」
- 105ページの「autofs プログラム」
- 106ページの「autofs マップ」
- 114ページの「autofs のしくみ」

第 4 章

autofs の管理

この章では、autofs 管理タスクの実行方法について説明します。具体的には、オートマウンタマップの変更、オートマウンタから非 NFS タイプのデバイスに対するアクセス、マップの設定などです。

- 84ページの「オートマウンタを起動する方法」
- 86ページの「マスタマップを変更する」
- 87ページの「間接マップを変更する」
- 87ページの「直接マップを変更する」
- 88ページの「autofs を使って CD-ROM アプリケーションにアクセスする」
- 89ページの「autofs を使って PC-DOS データのフロッピーディスクにアクセスする」
- 89ページの「CacheFS を使って NFS ファイルシステムにアクセスする」
- 90ページの「/home ディレクトリ構造を共通にする」
- 92ページの「プロジェクト関連ファイルの統合方法 (/ws ディレクトリ構造)」
- 94ページの「さまざまなアーキテクチャによる共有名前空間へのアクセスを設定する方法」
- 96ページの「セキュリティを適用する」
- 97ページの「autofs の表示機能の無効化」
- 100ページの「autofs での問題発生時の対処」

autofs の設定

この節では、autofs サービスの起動および停止の手順について説明します。

▼ オートマウンタを起動する方法

- リブートせずにデーモンを起動するには、スーパーユーザになって次のコマンドを入力します。

```
# /etc/init.d/autofs start
```

これでデーモンが起動します。

▼ オートマウンタを停止する方法

- リブートせずにデーモンを停止するには、スーパーユーザになって次のコマンドを入力します。

```
# /etc/init.d/autofs stop
```

一般的な作業と手順

この節では、ユーザの環境で遭遇する可能性のある最も一般的な問題について、そのいくつかを説明します。クライアントのニーズに最適な autofs を構成する際に役立つよう、各問題ごとに推奨する解決策を記載しています。

注 - この節で説明されている作業を実行するには、Solstice システム管理ツールを使用するか、または『*Solaris* ネーミングの管理』を参照してください。

マップを含む管理作業

次のリストは、autofs 環境を変更するために実行する必要が生じる可能性のある、マップに関する各種の管理作業です。

- 86ページの「マスタマップを変更する」
- 87ページの「間接マップを変更する」
- 87ページの「直接マップを変更する」
- 88ページの「マウントポイント衝突の回避」

マップの種類とその使用方法を表 4–1 に示します。

表 4–1 autofs のマップの種類とその使用方法

マップの種類	使用方法
マスタ	ディレクトリとマップを対応付ける
直接	autofs を特定のファイルシステムに割り当てる
間接	autofs を参照だけを行うファイルシステムに割り当てる

ネームサービスを基にして、ユーザの autofs 環境を変更する方法を表 4–2 に示します。

表 4–2 マップ管理

ユーザのネームサービス	変更用のツール
ローカルファイル	テキストエディタ
NIS	`make` ファイル
NIS+	`nistbladm`

表 4–3 は、変更後に `automount` コマンドを実行すべきかどうかを、変更したマップの種類別に示しています。たとえば、直接マップに追加または削除を行なった場合には、その変更を反映させるためにローカルシステムで `automount` コマンドを実行する必要があります。しかし、既存のエントリに変更を行なった場合には、`automount` コマンドを実行してその変更を反映させる必要はありません。

表 **4–3** automount コマンド実行タイミング

マップの種類	automountを再起動する **?**	
	追加または削除	既存のエントリを変更
auto_master	実行する	実行する
direct	実行する	実行しない
indirect	実行しない	実行しない

マップの変更

以下の手順では、ネームサービスとして NIS+ を使っていることが前提です。

▼ マスタマップを変更する

1. nistbladm コマンドを使って、マスタマップに必要な変更をします。
   『*Solaris* ネーミングの管理』を参照してください。
2. 各クライアントに対して、プロンプトで su を入力し、スーパーユーザのパスワードを入力してスーパーユーザになります。
3. 各クライアントに対して、automount コマンドを実行して、行った変更を確実に有効にします。
4. ユーザに変更を通知します。
   ユーザでも自分のコンピュータ上でスーパーユーザとして automount コマンドを実行できることを通知してください。

automount コマンドが実行されると、マスタマップを参照します。

▼ 間接マップを変更する

- nistbladm コマンドを使用して、間接マップへの変更を行います。
  『*Solaris* ネーミングの管理』を参照してください。

次にこのマップが使用されると、マウントが行われ、変更内容が有効となります。

▼ 直接マップを変更する

1. nistbladm コマンドを使用して、直接マップへの変更を追加または削除します。
   『*Solaris* ネーミングの管理』を参照してください。

2. 手順 **1** でマウントポイントエントリを追加または削除した場合、automount コマンドを実行します。

3. ユーザに変更を通知します。
   ユーザでも自分のコンピュータ上でスーパーユーザとして automount コマンドを実行できることを、通知してください。

注 - 既存の直接マップエントリの内容を変更しただけの場合、automount コマンドを実行する必要はありません。

たとえば、マップ auto_direct を変更し、ディレクトリ /usr/src は異なるサーバからマウントされると仮定します。このとき /usr/src がマウントされていない場合、ユーザが /usr/src をアクセスしようとすると、この新しいエントリはすぐに有効となります。/usr/src が現在マウントされている場合、自動アンマウントが行われるまで待ってから、アクセスすることができます。

注 - これらの余分な手順が必要な上、間接マップではマウントテーブル内で直接マップほどのメモリ領域を占有しないため、できるだけ間接マップを使用してください。間接マップは作成が簡単で、コンピュータファイルシステムに関する条件も直接マップほど厳しくありません。

マウントポイント衝突の回避

ローカルディスクパーティションを /src にマウントし、しかも autofs サービスを使用して他のソースディレクトリもマウントしたい場合、問題が発生することがあります。ユーザがマウントポイント /src を指定した場合、ユーザがローカルパーティションを参照しようとすると、autofs サービスがこれを隠します。

このパーティションをどこか他 (たとえば、/export/src) にマウントする必要があります。すると、たとえば、/etc/vfstab 内には次のようなエントリが必要となります。

```
/dev/dsk/d0t3d0s5 /dev/rdsk/c0t3d0s5 /export/src ufs 3 yes -
```

そして、auto_src 内のエントリは次のようになります。

```
terra  terra:/export/src
```

terra はコンピュータの名前です。

NFS 以外のファイルシステムへのアクセス

autofs は、NFS ファイル以外のファイルもマウントできます。autofs は、ファイルをフロッピーディスクや CD-ROM などの取り外し可能メディア上にマウントします。通常は、ボリュームマネージャ機能を使って取り外し可能メディアにファイルをマウントします。以下の例では、autofs を使ってこのマウント操作をどのように行うかを示しています。ボリュームマネージャ機能と autofs とは併用できないので、通常はまずボリュームマネージャを無効にしてから、これらのエントリを使用することになります。

サーバからファイルシステムをマウントするのではなく、ユーザがメディアをドライブにセットして、それをマップから参照します。autofs を使っているときに、NFS 以外のファイルシステムをアクセスしたい場合は、次の手順を参照してください。

▼ autofs を使って CD-ROM アプリケーションにアクセスする

注 - ボリュームマネージャを使わない場合は、この手順に従ってください。

♦ 次のように **CD-ROM** ファイルシステムの種類を指定してください。

```
hsfs -fstype=hsfs,ro :/dev/sr0
```

マウントする CD-ROM 装置の名前には、先頭にコロンを付けてください。

▼ autofs を使って PC-DOS データのフロッピーディスクにアクセスする

注 - ボリュームマネージャを使わない場合は、この手順に従ってください。

- フロッピーファイルシステムタイプを次のように指定します。

```
 pcfs -fstype=pcfs :/dev/diskette
```

CacheFS による NFS ファイルシステムへのアクセス

キャッシュファイルシステム (CacheFS) は、小型で高速なローカルディスクを使用することによって、特定のファイルシステムの性能を向上させるための、汎用の不揮発性キャッシュ機能です。

CacheFS を使用して、NFS ファイルシステムからのデータをローカルディスク上に保存することによって、NFS 環境の性能を改善できます。

▼ CacheFS を使って NFS ファイルシステムにアクセスする

1. `cfsadmin` コマンドを実行してローカルディスク上にキャッシュディレクトリを作成します。

```
# cfsadmin -c /var/cache
```

2. オートマウンタマップに **cachefs** エントリを追加します。

   たとえば、マスタマップに次のエントリを追加すると、すべてのホームディレクトリがキャッシュされます。

```
/home auto_home -fstype=cachefs,cachedir=/var/cache,backfstype=nfs
```

次のエントリを auto_home マップに追加すると、rich という名前のユーザのホームディレクトリだけがキャッシュされます。

```
rich  -fstype=cachefs,cachedir=/var/cache,backfstype=nfs  dragon:/export/home1/rich
```

注 - マップに指定されたオプションは、後から見つかったものがそれより前に見つかったものよりも優先されます。つまり、最後のオプションが使用されます。上の例の auto_home マップに追加する個別のエントリで、オプションの一部を変更する場合に指定する必要があるのは、マスタマップに指定されているオプションだけです。

オートマウンタのカスタマイズ

オートマウンタマップを設定する方法はいくつかあります。以下のタスクでは、使いやすいディレクトリを作成するためにオートマウンタマップをカスタマイズする方法を説明します。

▼ /home ディレクトリ構造を共通にする

ネットワークユーザにとって望ましいのは、自分や他人のホームディレクトリが /home の下に存在していることです。この /home 以下のディレクトリは、クライアントであれサーバであれすべてのコンピュータで共通になるようにしてください。

Solaris ソフトウェアをインストールすると、マスタマップ /etc/auto_master が設定されます。

```
# Master map for autofs
#
+auto_master
/net  -hosts  -nosuid,nobrowse
/home  auto_home -nobrowse
/xfn  -xfn
```

auto_home 用のマップも、/etc の下に設定されます。

```
# Home directory map for autofs
#
+auto_home
```

外部の auto_home マップへの参照を除けば、このマップは空です。/home の下にあるディレクトリをすべてのコンピュータに共通とする場合、この /etc/auto_home マップを変更しないでください。すべてのホームディレクトリエントリは、NIS または NIS+ のネームサービスマップに登録されていなければなりません。

注 - 制限がなければ、どんなユーザでも任意のコンピュータ上でスーパーユーザ特権を持つことができるため、ユーザが自分のホームディレクトリから実行可能ファイル set uid を実行することを許可しないでください。

▼ 複数のホームディレクトリファイルシステムを持つ /home を設定する

1. /export/home の下にホームディレクトリパーティションをインストールします。

   複数のパーティションがある場合、それらを別個のディレクトリ、たとえば /export/home1、/export/home2 などの下にインストールします。

2. **Solstice System Management Tools** を使用して auto_home マップの作成と管理を行います。

   新規のユーザアカウントを作成するときには、ユーザのホームディレクトリの位置を auto_home マップに入力します。マップエントリは、たとえば次のように簡単な形式になっています。

```
rusty     dragon:/export/home1/&
   gwenda    dragon:/export/home1/&
   charles   sundog:/export/home2/&
   rich      dragon:/export/home3/&
```

   アンパサンド「&」を使用してマップキーを置換することに注意してください。これは次の例での rusty を 2 回入力する場合の省略表記です。

```
rusty  dragon:/export/home1/rusty
```

auto_home マップを適切な場所に配置することによって、ユーザはパス /home/*user* で任意のホームディレクトリ (自分のものを含む) を参照できます。ここで、*user* はユーザのログイン名で、マップのキーでもあります。他のユーザのコンピュータにログインするとき、全ホームディレクトリが共通の構造になっていると便利です。autofs は、そこにユーザのホームディレクトリをマウントします。同様に、リモートのウィンドウクライアントを他のコンピュータ上で実行する場合、

そのクライアントプログラムは、ユーザがウィンドウ表示を提供するコンピュータ上で持つのと同じ /home ディレクトリの構造を持っています。

この共通の構造は、サーバにも影響します。上の例を使用し、rusty がサーバ dragon にログインした場合、autofs は、/export/home1/rusty を /home/rusty にループバックマウントすることによって、ローカルディスクへの直接アクセスを可能にします。

ユーザは、自分のホームディレクトリの実際の位置を知る必要がありません。rusty がより多くのディスク空き容量を必要とし、自分のホームディレクトリを他のサーバに再配置する必要がある場合、この新しい位置を反映するためには、auto_home マップ内の rusty のエントリを変更するだけですみます。すべてのユーザがパス /home/rusty を継続して使用できます。

▼ プロジェクト関連ファイルの統合方法 (/ws ディレクトリ構造)

あなたが大規模なソフトウェア開発プロジェクトの管理者だとします。プロジェクトに関連するすべてのファイルを /ws (Work Space の略) と呼ばれるディレクトリの下で使用できるようにしたいとします。このディレクトリは、そのサイトのすべてのワークステーション間で共通となります。

1. /ws ディレクトリ用のエントリを、**NIS** または **NIS+** のサイトの auto_master マップに追加します。

   ```
   /ws auto_ws -nosuid
   ```

   auto_ws マップによって、/ws ディレクトリの内容が決まります。

2. 用心のため、-nosuid オプションを追加します。

   このオプションを指定すると、ユーザは、ワークスペースに存在する setuid を実行できなくなります。

3. auto_ws マップにエントリを追加します。

   この auto_ws マップは、各エントリがサブプロジェクトを指定するように作成されます。最初の試みによって、次のようなマップが得られます。

```
compiler    alpha:/export/ws/&
   windows     alpha:/export/ws/&
   files       bravo:/export/ws/&
   drivers     alpha:/export/ws/&
   man         bravo:/export/ws/&
```

```
tools       delta:/export/ws/&
```

各エントリの最後にあるアンパサンド (&) は、エントリキーの単なる省略です。たとえば、最初のエントリは次のものと等しくなります。

```
compiler  alpha:/export/ws/compiler
```

この段階で得られるマップは単純で、まだ不十分なものです。プロジェクト責任者は、man エントリ内のドキュメントを各サブプロジェクトの下にあるサブディレクトリとして提供することを決定します。また各サブプロジェクトは、複数バージョンのソフトウェアを指定するためにサブディレクトリを必要とします。これらのサブディレクトリは、それぞれサーバ上のディスクパーティション全体に割り当てられなければなりません。

マップ内のエントリを次のように変更します。

```
compiler \
    /vers1.0    alpha:/export/ws/&/vers1.0 \
    /vers2.0    bravo:/export/ws/&/vers2.0 \
    /man        bravo:/export/ws/&/man
windows \
    /vers1.0    alpha:/export/ws/&/vers1.0 \
    /man        bravo:/export/ws/&/man
files \
    /vers1.0    alpha:/export/ws/&/vers1.0 \
    /vers2.0    bravo:/export/ws/&/vers2.0 \
    /vers3.0    bravo:/export/ws/&/vers3.0 \
    /man        bravo:/export/ws/&/man
drivers \
    /vers1.0    alpha:/export/ws/&/vers1.0 \
    /man        bravo:/export/ws/&/man
tools \
    /           delta:/export/ws/&
```

マップはとても大きくなったように見えますが、それでも 5 つのエントリが含まれているだけです。各エントリには多重マウントが含まれるため、大きくなっています。たとえば /ws/compiler への参照には、vers1.0、vers2.0、および man ディレクトリ用の 3 つのマウントが必要です。各行の最後にあるバックスラッシュは、エントリが次の行に続いていることを示します。このエントリは 1 行なのですが、行ブレークとインデントを使用して読み易くなっています。tools ディレクトリには全サブプロジェクト用のソフトウェア開発支援ツールが収められているた

め、これは同じサブディレクトリ構造になっていません。tools ディレクトリは依然として単一のマウントです。

この配置によって、管理者には高い柔軟性が与えられます。ソフトウェアプロジェクトは、大量のディスク領域を消費することが知られています。このプロジェクトの有効期間中には、さまざまなディスクパーティションの再配置と拡張が要求されます。これらの変更内容が auto_ws マップに反映される限り、/ws のもとのディレクトリ階層は変更されないため、ユーザに通知する必要はありません。

サーバ alpha と bravo は同じ autofs マップを参照するため、これらのコンピュータにログインしたユーザは、予想通りの名前空間 /ws を見ることになります。これらのユーザには、NFS マウントの代わりにループバックマウントを通じて、ローカルファイルへの直接アクセスが提供されます。

▼ さまざまなアーキテクチャによる共有名前空間へのアクセスを設定する方法

ローカルの実行可能ファイル、また、表計算ツールやワードプロセッシングのパッケージなどのアプリケーション用に、共有名前空間を 1 つにまとめる必要があります。この名前空間のクライアントは、異なる実行可能形式を必要とするさまざまなワークステーションアーキテクチャを使用します。また、一部のワークステーションはオペレーティングシステムの異なるバージョンを実行しています。

1. nistbladm コマンドを使用して auto_local マップを作成します。

   『*Solaris* ネーミングの管理』を参照してください。

2. 共有名前空間に対して **1** つのサイト名を選択し、この空間に属するファイルとディレクトリを容易に識別できるようにします。

   たとえば、この名前として /usr/local を選択した場合、/usr/local/bin というパスは、明らかにこの名前空間の一部となります。

3. ユーザを簡単に認識できるよう、**autofs** の間接マップを作成して、/usr/local にマウントします。**NIS+ (**または **NIS)** の **auto_master** マップ内に次のエントリを設定します。

   ```
   /usr/local      auto_local      -ro
   ```

   なお、マウントオプション ro は、クライアントがどのファイルやディレクトリにも書き込みできないことを意味します。

4. 適切なディレクトリをサーバ上にエクスポートします。

5. bin のエントリを auto_local マップに作成します。

   ディレクトリ構造は次のようになります。

```
bin      aa:/export/local/bin
```

   異なるアーキテクチャのクライアントに対応するというニーズを満足するため、この bin ディレクトリに対する参照は、クライアントのアーキテクチャタイプに応じて、サーバ上の異なるディレクトリに対して行う必要があります。

6. 異なるアーキテクチャのクライアントに対応するため、**autofs** の「CPU」変数を追加することによって、エントリを変更します。

```
bin      aa:/export/local/bin/$CPU
```

注 - SPARCstation™ クライアントの場合は、実行可能ファイルはサーバ上の /export/local/bin/sparc のもとで使用できるようにしてください。x86 クライアントの場合、/export/local/bin/i386 です。

▼ 互換性のないクライアントオペレーティングシステムをサポートする

1. クライアントのオペレーティングシステムタイプを判定する変数を、アーキテクチャタイプに結合します。

   autofs の「OSREL」変数を CPU 変数と結合することによって、CPU タイプと OS リリースの両方を決める名前を作成することができます。

2. 次のマップエントリを作成します。

```
bin      aa:/export/local/bin/$CPU$OSREL
```

バージョン 5.6 のオペレーティングシステムを実行している SPARC クライアントの場合、サーバから /export/local/bin/sparc5.6 をエクスポートし、同様に他のリリースもエクスポートする必要があります。オペレーティングシステムは実行可能形式での下方互換性を維持しようとするため、OS のリリースについては問題ないと想定し、このあとの例では、OS のリリースによる区別は省略します。

これまで、1 つのサーバ「aa」用のエントリを設定しました。大規模ネットワークでは、これらの共有ファイルを複数のサーバ間で複写したいという要望があります。NFS のトラフィックがローカルネットワークセグメントに限定されるよう、各サーバは、サービスを提供するクライアントに近いネットワークにある必要があります。

▼ 複数のサーバ間で共有ファイルを複写する

複製された読み取り専用のファイルシステムを共有する最良の方法は、障害時回避機能を使うことです。障害時回避機能については、71ページの「クライアント側障害時回避機能」 を参照してください

- **エントリを変更して、次のようにすべての複製サーバのリストをカンマで区切って作成します。**

```
bin     aa,bb,cc,dd:/export/local/bin/$CPU
```

autofs は最も近いサーバを選択します。サーバに複数のネットワークインタフェースがある場合、各インタフェースを指定します。autofs はクライアントに最も近いインタフェースを選択し、NFS トラフィックの不必要な経路指定を回避します。

▼ セキュリティを適用する

- **NIS または NIS+ のネームサービス** `auto_master` **ファイル内に、次のエントリを作成します。**

```
/home     auto_home     -nosuid
```

`nosuid` オプションが指定されていると、ファイルを作成するときに `setuid` ビットも `setgid` ビットも設定することができません。

このエントリは、ローカル `/etc/auto_master` ファイル内の `/home` エントリを無効にします (前の例を参照)。その理由は、外部ネームサービスマップへの `+auto_master` 参照がこのファイルの `/home` エントリよりも前に行われるためです。`auto_home` マップのエントリにマウントオプションが指定されている場合、`nosuid` オプションは上書きされます。したがって、`auto_home` マップではオプションを一切使わないか、各エントリに `nosuid` オプションを指定します。

注 - サーバ上の /home の上またはその下に、ホームディレクトリのディスクパーティションをマウントしないでください。

▼ autofs で公共ファイルハンドルを使う方法

◆ 次のようなエントリを作成します。

```
/usr/local      -ro,public     bee:/export/share/local
```

public オプションによって、公共ファイルハンドルが使用されます。NFS サーバが公共ファイルハンドルをサポートしていないと、マウントは失敗します。

▼ autofs で NFS URL を使う方法

◆ 次のようなエントリを作成します。

```
/usr/local      -ro     nfs://bee/export/share/local
```

autofs は NFS サーバ上の公共ファイルハンドルを使おうとしますが、サーバが公共ファイルハンドルをサポートしていないと、MOUNT プロトコルが使用されます。

autofs の表示機能の無効化

Solaris 2.6 およびその互換バージョンでインストールされる /etc/auto_master では、/home と /net のエントリに –nobrowse オプションが追加されています。また、/etc/auto_master の中の /home と /net のエントリが変更されていなければ、アップグレードによってこれらのエントリに –nobrowse オプションが追加されます。しかし、これらの変更を手動で行なったり、各サイトの autofs マウントポイントの表示機能を無効にしなければならないこともあります。

表示機能を無効にする方法はいくつかあります。1 つは automountd に対してコマンド行オプションを使用して無効にする方法です。この場合、クライアントに対する autofs の表示機能は完全に無効になります。もう 1 つは、NIS または NIS+ 名前空間の autofs マップを使っているクライアントすべてに対して、またはネットワーク全体で使用している名前空間がない場合にはローカルの autofs マップを使っている各クライアントごとに、マップエントリで表示機能を無効にする方法です。

▼ 単一の NFS クライアントに対して autofs の表示機能を完全に無効にする方法

1. 起動スクリプトに –n オプションを追加します。

   スーパーユーザになって /etc/init.d/autofs スクリプトを編集して、automountd デーモンを起動している行に –n オプションを追加します。

```
/usr/lib/autofs/automountd -n \
 < /dev/null > /dev/console 2>&1 # start daemon
```

2. **autofs** サービスを再起動します。

```
# /etc/init.d/autofs stop
# /usr/init.d/autofs start
```

▼ すべてのクライアントに対して autofs の表示機能を無効にする

すべてのクライアントに対して表示機能を無効にするには、NIS や NIS+ といったネームサービスを使用している必要があります。使用していない場合は、クライアントごとにオートマウンタマップを手動で編集する必要があります。この例では、/home ディレクトリの表示機能を無効にします。この手順を、無効にする間接 autofs ノードそれぞれについて実行します。

1. ネームサービスの auto_master ファイルにある /home エントリに対して、–nobrowse オプションを追加します。

```
/home      auto_home      -nobrowse
```

2. すべてのクライアントで、automount コマンドを実行します。

   変更は、このクライアントで automount コマンドを実行するかクライアントをリブートすると反映されます。

```
# /usr/sbin/automount
```

▼ 1 台の NFS クライアントに対して autofs の表示機能を無効にする

この例では、/net ディレクトリの表示機能を無効にします。同様の手順は、/home などの autofs マウントポイントにも適用できます。

1. /etc/nsswitch.conf の automount エントリを調べます。

   ローカルファイルエントリを優先させるためには、ネームサービスの切り替えファイルでネームサービスより前に files を指定します。たとえば次のようにします。

```
automount: files nisplus
```

   これは、Solaris をインストールしたときのデフォルト設定です。

2. /etc/auto_master の中で +auto_home エントリの位置を確認します。

   ローカルファイルへの追加を名前空間のエントリよりも優先させるために、+auto_master エントリを /net よりも下の位置に移動します。

```
# Master map for automounter
#
/net -hosts -nosuid
/home auto_home
/xfn -xfn
+auto_master
```

   標準の設定では、+auto_master エントリはファイルの先頭に置かれます。そのため、ローカルの変更は一切できません。

3. –nobrowse オプションを /etc/auto_master ファイルの /net エントリに追加します。

```
/net      -hosts      -nosuid,nobrowse
```

4. すべてのクライアントで、automount コマンドを実行します。

   変更は、このクライアントで automount コマンドを実行するかクライアントをリブートすると反映されます。

```
# /usr/sbin/automount
```

autofs での問題発生時の対処

autofs を使用していて問題が発生することもあります。この節の目的は、問題解決のプロセスを容易にすることです。これは 2 つの節に分かれています。

この節では、autofs が生成するエラーメッセージのリストを提供します。このリストは 2 つの部分に分かれています。

- automount の詳細 (–v) オプションによって生成されたエラーメッセージ
- 任意の時点で表示されるエラーメッセージ

それぞれのエラーメッセージには説明が続き、考えられる原因を示します。

問題を解決する際は、詳細 (–v) オプションを使って起動してください。そうしないと、問題が発生しても原因がわからない可能性があります。

以下では Autofs で問題が発生したときに表示される可能性のあるエラーメッセージを取り上げ、問題の内容について説明します。

自動マウンタで生成されるエラーメッセージ

bad key *key* in direct map *mapname*

直接マップの走査中、autofs は「/」が付いていないエントリキーを検出しました。直接マップ内のキーはフルパス名でなければなりません。

bad key *key* in indirect map *mapname*

間接マップの走査中、autofs が「/」を含むエントリキーを検出しました。間接マップのキーは、パス名ではなく、単純な名前でなければなりません。

`can't mount` *server*`:`*pathname: reason*

サーバ上のマウントデーモンが、*server:pathname* 用のファイルハンドル提供を拒否しています。サーバ上のエクスポートテーブルをチェックしてください。

`couldn't create mount point` *mountpoint*`:` *reason*

マウントに必要なマウントポイントを autofs が作成できませんでした。このメッセージが最もよく表示されるのは、サーバのエクスポートされたファイルシステムをすべて階層的にマウントしようとしたときです。必要なマウントポイントが、マウントできない (エクスポートできない) ファイルシステムにだけ存在する可能性があります。しかもエクスポートされた親ファイルシステムが読み取り専用でエクスポートされているため、これを作成できません。

`leading space in map entry` *entry* `text in` *mapname*

autofs は、先頭にスペースが含まれるエントリを自動マウントマップ内に発見しました。これは通常、たとえば次のように誤って続けて入力したマップエントリが原因です。

```
fake
/blat   frobz:/usr/frotz
```

上の例では、最初の行はバックスラッシュ (\) で終了しなければならないため、autofs が 2 番目の行を検出したとき、この警告が生成されます。

mapname`: Not found`

必要なマップを検出できません。このメッセージは、-v オプションが指定された場合にだけ出力されます。マップ名のスペルとパス名をチェックしてください。

`remount` *server*`:`*pathname* `on` *mountpoint*`: server not responding`

autofs は、以前にアンマウントしたファイルシステムの再マウントに失敗しました。このメッセージは、ファイルシステムが正常に再マウントされるまで、一定間隔ごとに表示されます。

`WARNING:` *mountpoint* `already mounted on`

autofs は既存のマウントポイントの上にマウントしようとしています。これは、autofs に内部エラー (異常) があることを意味しています。

一般的なエラーメッセージ

`dir` *mountpoint* `must start with '/'`

自動マウンタのマウントポイントはフルパス名で指定しなければなりません。マウントポイントのスペルとパス名をチェックしてください。

`hierarchical mountpoints:` *pathname1* `and` *pathname2*

autofs では、マウントポイントが階層関係を持つことはできません。つまり、オートマウンタのマウントポイントが、他のオートマウントされたファイルシステム内に含まれてはいけません。

`host` *server* `not responding`

autofs は *server* に対して通信を試みましたが、応答を受信しませんでした。

hostname`: exports:` *rpc_err*

hostname からのエクスポートリストの獲得エラー。これはサーバまたはネットワークのトラブルを示します。

`map` *mapname*`, key` *key*`: bad`

マップエントリの形式が誤っており、autofs はこれを解釈できません。エントリを再チェックしてください。おそらく、この中にエスケープを必要とする文字があります。

mapname`:` *nis_err*

NIS マップ内のエントリを検索する際のエラー。NIS のトラブルの可能性あり。

`mount of` *server*`:`*pathname* `on` *mountpoint:reason*

autofs はマウントの実行に失敗しました。これはサーバまたはネットワークのトラブルを意味します。

mountpoint`: Not a directory`

autofs は、マウントポイントがディレクトリではないため、これにそれ自身をマウントできません。マウントポイントのスペルとパス名をチェックしてください。

`nfscast: cannot send packet:` *reason*

autofs は、複写されたファイルシステム位置のリストにあるサーバに対して、問い合わせパケットを送信できません。

`nfscast: cannot receive reply: `*reason*

autofs は、複写されたファイルシステム位置のリストにあるどのサーバからも応答を受信できません。

`nfscast:select: `*reason*

これらのエラーメッセージは、複写されたファイルシステム用のサーバに対し `ping` を実行しようとしていることを示します。ネットワークに問題がある可能性があります。

`pathconf: no info for `*server*`:`*pathname*

autofs は、*pathname* 用の `pathconf` を獲得できませんでした。

`pathconf: `*server*`: server not responding`

autofs は、(POSIX) `pathconf()` 情報を提供するサーバ上のマウントデーモンと通信できません。

autofs に関するその他のエラー

`/etc/auto*` ファイルに実行ビットセットがある場合、自動マウンタはマップを実行しようとして、次のようなメッセージを作成します。

```
/etc/auto_home: +auto_home: not found
```

この場合、`auto_home` ファイルには間違ったパーミッションが入ることになります。ファイルの各エントリはほぼ上記のようなエラーメッセージを生成します。以下のコマンドを入力して、ファイルへのパーミッションをリセットする必要があります。

```
# chmod 644 /etc/auto_home
```

第 5 章

autofs について

この章では、autofs サービスの構成要素と autofs マップの詳細について説明します。

- 105ページの「autofs プログラム」
- 106ページの「autofs マップ」
- 114ページの「autofs のしくみ」
- 127ページの「autofs リファレンス情報」

autofs プログラム

autofs サービスをサポートするプログラムには、`automount` と `automountd` の 2 つがあります。どちらもシステムを起動すると実行されますが、常駐するのは `automountd` だけです。

automount

このコマンドは autofs マウントポイントをインストールし、オートマスタファイルの情報を各マウントポイントと関連づけます。コマンドの構文は次のようになります。

`automount [ –t` *duration* `] [ –v ]`

–t *duration* にはファイルシステムがマウントされたままの時間を秒単位で指定し、–v を指定すると詳細表示モードになります。詳細表示モードでこのコマンドを実行すると、問題の解決が容易になります。

特定の値を指定しないと、–t *duration* は 5 分に設定されます。ほとんどの状況ではこの値で大丈夫ですが、多くのファイルシステムを自動マウントしているシステムでは、この間隔を延ばす必要もでてきます。特にサーバに多くのアクティブユーザがいる場合、自動マウントしたファイルシステムを 5 分毎にチェックするのは効率が悪くなる恐れがあります。autofs ファイルシステムを 1800秒 (30 分) 毎にチェックさせた方がより最適です。ファイルシステムを 5 分毎にアンマウントさせないことで、`df` によってチェックされる `/etc/mnttab` が非常に大きくなることもあります。`–F` オプション (`df(1M)` のマニュアルページ参照) を使用して `df` からの出力をフィルタ処理するか、`egrep` を使用すればこの問題を解決するのに役立ちます。

もう一つ考慮すべき点としては、間隔を変更することで、マップに対する変更内容をどれだけ迅速に反映させるかも変更になることが挙げられます。変更は、ファイルシステムがアンマウントされるまでは無効です。オートマウンタマップの変更方法については、86ページの「マップの変更」 を参照してください。

automountd

このデーモンは autofs サービスからのマウントとアンマウント要求を処理します。コマンドの構文は次のようになります。

```
automountd [ –Tnv ] [ -D name=value ]
```

`–T` を指定すると、RPC コールがすべて標準出力に表示されます。`–n` を指定すると、すべての autofs ノードで表示機能が無効になります。`–v` を指定すると、コンソールへのすべてのステータスメッセージが記録されます。`–D` *name=value* を使うと、*name* で表された自動マウントマップの値の代わりに *value* が使われます。自動マウントマップのデフォルト値は `/etc/auto_master` です。`–T` オプションは障害追跡のために使います。

autofs マップ

autofs は 3 種類のマップを使用します。

- マスタマップ

- 直接マップ
- 間接マップ

マスタマップ

auto_master マップでは、ディレクトリからマップへの関連付けを行います。これは、すべてのマップを指定するマスタリストであり、autofs が参照します。auto_master ファイルの内容の例を示します。

表 5–1 /etc/auto_master ファイルの例

```
# cat /etc/auto_master
# Master map for automounter
#
+auto_master
/net        -hosts       -nosuid,nobrowse
/home       auto_home     -nobrowse
/xfn        -xfn
/-         auto_direct    -ro
```

この例では、汎用の auto_master ファイルに auto_direct マップのための追加が行われています。マスタマップ /etc/auto_master の各行は、次の構文に従っています。

mount-point *map-name* [*mount-options*]

mount-point

mount-point はディレクトリのフル (絶対) パス名です。このディレクトリが存在しない場合、可能ならば autofs はこれを作成します。このディレクトリが存在し、しかも空ではない場合、マウントすることによってその内容が隠されます。この場合、autofs は警告を出します。

マウントポイントとして /- を指定すると、マップが直接マップであり、このマップ全体に関連付けられている特定のマウントポイントがないことを表します。

map-name

map-name 名は、位置に対する指示またはマウント情報を検出するために、autofs が使用するマップです。この名前がスラッシュ (/) で始まる場合、autofs はこの名前をローカルファイルとして解釈します。そうでない場合、autofs はネームサービススイッチ構成ファイルで指定される検索によりマウント情報を検索します。/net と /xfn に対して使われる特殊なマップもあります (109ページの「マウントポイント /net」 と109ページの「マウントポイント /xfn」 を参照してください)。

mount-options

mount-options は、オプションです。*map-name* のエントリに他のオプションがある場合を除き、*map-name* で指定されたエントリのマウントに適用されるオプションをカンマで区切って並べます。これらのマウントオプションは、標準の NFS マウント用のものと同じですが、bg (バックグラウンド) と fg (フォアグラウンド) は使用できません。48ページの「mount」 を参照してください。具体的なファイルシステムごとのオプションについては、そのファイルシステムのマニュアルページで「mount」を参照してください (たとえば NFS 固有のマウントオプションについては、mount_nfs(1M))。NFS 固有のマウントポイントの場合、bg (バックグラウンド) オプションと fg (フォアグラウンド) オプションは適用されません。

で始まる行はコメント行です。この場合、その行の最後まですべて無視されます。

長い行を短い行に分割するには、行末にバックスラッシュ (\) を入力します。入力できる文字数の上限は 1024 です。

マウントポイント /home

マウントポイント /home は、/etc/auto_home (間接マップ) に記述されたエントリがマウントされるディレクトリです。

注 - autofs はすべてのコンピュータで動作し、デフォルトでは /net と /home (自動マウントされるホームディレクトリ) をサポートします。このデフォルトは、NIS ならば auto.master マップ、NIS+ ならば auto_master テーブルを使って、またはローカルの /etc/auto_master ファイルを編集することによって変更できます。

マウントポイント /net

autofs は、特別のマップ –hosts 内の全エントリをディレクトリ /net の下にマウントします。これは hosts データベースだけを使用する組み込みマップです。たとえば、コンピュータ gumbo が hosts データベース内にあり、しかもそのファイルシステムのどれかをエクスポートする場合、次のコマンドによって、カレントディレクトリがコンピュータ gumbo のルートディレクトリに変更されます。

```
%cd /net/gumbo
```

なお、autofs はホスト gumbo のエクスポートされたファイルシステムだけをマウントできます。つまり、ローカルディスク上のファイルシステムではなく、ネットワークユーザが使用できるサーバ上のファイルシステムです。したがって、gumbo にあるすべてのファイルとディレクトリは、/net/gumbo では利用できない場合があります。

/net を使ったアクセスでは、サーバ名はパスの中に指定されるため、位置に依存します。したがって、エクスポートされるファイルシステムを別のサーバに移動すると、そのパスは使えなくなります。このような場合は /net を使わずに、そのファイルシステムに対応するエントリをマップの中に設定します。

注 - autofs はマウント時だけサーバのエクスポートリストを調べます。サーバのファイルシステムが一度マウントされると、そのファイルシステムがアンマウントされ、次にマウントされるまで autofs はそのサーバをチェックしません。したがって、新たにエクスポートされたファイルシステムは、それがサーバからアンマウントされ、再度マウントされるまでは見えません。

マウントポイント /xfn

このマウントポイントは、FNS 名前空間を使って共有されるリソースの autofs ディレクトリ構造がマウントされます (FNS について詳細は、『*Solaris* ネーミングの設定と構成』を参照してください)。

直接マップ

直接マップは自動マウントポイントです。つまり、直接マップによって、クライアント上のマウントポイントとサーバ上のディレクトリが直接対応付けられます。直接マップには完全なパス名があり、明示的に関係を示します。 以下に一般的な `/etc/auto_direct` マップを示します。

```
/usr/local       -ro \
          /bin                    ivy:/export/local/sun4 \
          /share                  ivy:/export/local/share \
          /src                    ivy:/export/local/src
   /usr/man         -ro   oak:/usr/man \
                               rose:/usr/man \
                               willow:/usr/man
   /usr/games       -ro   peach:/usr/games
   /usr/spool/news  -ro   pine:/usr/spool/news \
                                       willow:/var/spool/news
```

直接マップの行は、以下の構文に従っています。

key [*mount-options*] *location*

key

key は直接マップでのマウントポイントのパス名です。

mount-options

mount-options は、このマウントに適用したいオプションです。これらのオプションは、マップのデフォルトと異なる場合だけ必要です。各ファイルシステムの種類ごとのオプションについては、そのファイルシステムのマニュアルページで「mount」を参照してください (たとえば `CacheFS` に固有のマウント操作については、マニュアルページの `mount_cachefs(1M)` を参照してください)。

location

location にはファイルシステムの位置を、NFS ファイルシステムならば *server:pathname*、High Sierra ファイルシステム (HSFS) ならば *:devicename* という形式で指定します。

注 - *pathname* には自動マウントしたマウントポイントを含めず、ファイルシステムへの実際の絶対パスである必要があります。たとえば、ホームディレクトリの位置は、*server*:/home/*username* ではなく、*server*:/export/home/*username* として表示する必要があります。

マスタマップと同様、# で始まる行はコメントです。その行のテキストの最後まですべて無視されます。長い行を短い行に分割するには、行の最後にバックスラッシュを入力します。

すべてのマップの中で、直接マップのエントリが、/etc/vfstab (vfstab にはマウントされるすべてのファイルシステムのリストが含まれる) で対応するエントリと一番単純な形式において最もよく似ています。/etc/vfstab に現れるエントリは次のようになります。

```
dancer:/usr/local - /usr/local/tmp nfs - yes ro
```

直接マップでは次のようになります。

```
/usr/local/tmp      -ro      dancer:/usr/local
```

注 - オートマウンタマップの間では、オプションの連結はされません。あるオートマウンタマップでオプションが追加されると、それまでに見つかったマップに指定されているオプションはすべて無視され、新しいオプションだけが使われます。たとえば、auto_master マップに指定されているオプションは、他のマップの中の対応するエントリによって上書きされます。

この種類のマップについては、ほかにも重要な機能があります。119ページの「autofs がクライアント用の最も近い読み取り専用ファイルを選択する方法 (複数ロケーション)」 を参照してください。

マウントポイント /–

表 5–1 にある /- というマウントポイントは、auto_direct の中のエントリを具体的なマウントポイントに関連付けないように autofs に指示します。間接マップの場合は、auto_master ファイルに定義されたマウントポイントを使います。直接マップの場合は、ここに示されたマップの中で指定されたマウントポイントを使います (直接マップのキーとマウントポイントはフルパス名であることに注意してください)。

NIS または NIS+ の auto_master ファイルには、直接マップのエントリは 1 つしか存在できません。マウントポイントは 1 つの名前空間の中で一意でなければならないためです。auto_master がローカルファイルならば、重複しないかぎり直接マップのエントリがいくつあってもかまいません。

間接マップ

間接マップは、キーの置換値を使用してクライアント上のマウントポイントとサーバ上のディレクトリとを対応させます。間接マップは、ホームディレクトリなどの特定のファイルシステムをアクセスするのに便利です。auto_home マップは間接マップの一例です。

間接マップ内の行は以下の一般的な構文になります。

key [*mount-options*] *location*

key

key は間接マップでの単純名 (スラッシュなし) です。

mount-options

mount-options は、このマウントに適用するオプションです。これらのオプションが必要なのは、マップのデフォルトと異なる場合だけです。各ファイルシステムタイプごとのオプションについては、そのファイルシステムのマニュアルページで「mount」を参照してください (たとえば NFS 固有のマウントオプションについては、マニュアルページの mount_nfs(1M) を参照してください)。

location

location はファイルシステムステムの位置を示し、*server:pathname* により (1 つまたは複数) 指定されます。

注 - *pathname* には自動マウントしたマウントポイントを含めず、ファイルシステムへの実際の絶対パスである必要があります。たとえば、ディレクトリの位置は、*server*:/net/*server*/usr/local ではなく、*server*:/usr/local として表示する必要があります。

マスタマップと同様、# で始まる行はコメントです。その行のテキストの最後まですべて無視されます。長い行を短い行に分割するには、行の最後にバックスラッ

シュ (\) を入力します。表 5–1 に、次のエントリを含む auto_master マップを示します。

```
/home       auto_home         -nobrowse
```

auto_home は、/home のもとでマウントされるエントリを含む間接マップの名前です。一般的な auto_home マップには以下の構文が含まれます。

```
david                   willow:/export/home/david
rob                     cypress:/export/home/rob
gordon                  poplar:/export/home/gordon
rajan                   pine:/export/home/rajan
tammy                   apple:/export/home/tammy
jim                     ivy:/export/home/jim
linda    -rw,nosuid     peach:/export/home/linda
```

例として、前のマップがホスト oak にあると想定します。ユーザ linda がホームディレクトリを /home/linda として指定するパスワードデータベースにエントリがある場合、コンピュータ oak にログインするたびに、autofs はコンピュータ peach に常駐する /export/home/linda ディレクトリをマウントします。彼女のホームディレクトリは、読み書き可能な nosuid にマウントされます。

次のような状況が発生したと想定してください。ユーザ linda のホームディレクトリがパスワードデータベースに、/home/linda として表示されます。Linda も含め誰でも、前の例のマップを参照するマスタマップで設定されたどのコンピュータからでも、このパスにアクセスできます。

こうした状況のもとでは、ユーザ linda はこれらのどのコンピュータでも login や rlogin を実行し、代わりに彼女用のホームディレクトリをマウントさせることができます。

さらに、これで linda は次のコマンドも入力できます。

```
% cd ~david
```

autofs は彼女のために David のホームディレクトリをマウントします (すべてのアクセス権で許可されている場合)。

注 - オートマウンタマップの間には、オプションの連結はありません。オートマウンタマップに追加されたいずれのオプションも、前に検索されたマップに表示されているすべてのオプションを上書きします。たとえば、auto_master マップに含まれているオプションは、その他いずれのマップの対応するエントリによって上書きされます。

ネームサービスのないネットワークでこれを行うには、ネットワーク上のすべてのシステムで、すべての関連ファイル (/etc/passwd など) を変更する必要があります。NIS では、NIS マスタサーバで変更を行い、関連するデータベースをスレーブのデータベースに伝達します。NIS+ を稼働中のネットワークでは、変更後に関連データベースがスレーブサーバに自動的に伝達されます。

autofs のしくみ

autofs は、自動的に適切なファイルシステムをマウントするためのクライアント側のサービスです。クライアントが現在マウントされていないファイルシステムをアクセスしようとすると、autofs ファイルシステムはその要求に介入し、automountd を呼び出して要求されたディレクトリをマウントします。automountd はディレクトリを検索してマウントし、応答します。応答を受け取ると、autofs は待たせてあった要求の処理を続行させます。それ以降のそのマウントへの参照は autofs によって切り替えられ、このファイルシステムが一定時間使われないために自動的にアンマウントされるまで、automountd は不要となります。

自動マウントを行うのに、次のコンポーネントが相互に動作します。

- automount コマンド
- autofs ファイルシステム
- automountd デーモン

automount コマンドは、システム起動時に呼び出され、マスタマップファイル auto_master を読み取って autofs マウントの最初のセットを作成します。これらの autofs のマウントは起動時に自動的にはマウントされません。後でファイルシステムがマウントされるポイントです。このようなポイントをトリガーノードと呼ぶこともあります。

autofs マウントが設定されると、要求があったときにファイルシステムをマウントすることができます。たとえば、autofs が、現在マウントされていないファイルシ

ステムをアクセスする要求を受け取ると、automountd を呼び出して要求されたファイルシステムを実際にマウントさせます。

Solaris 2.5 からは、automountd デーモンは完全に automount コマンドから独立しました。そのため、automountd デーモンを 1 度停止してから起動し直さなくてもマップ情報を追加、削除、変更できるようになりました。

最初に autofs マウントをマウントすると、automount コマンドを使って、auto_master 内のマウントのリストをマウントテーブルファイル /etc/mnttab (以前は /etc/mtab) 内のマウントされているファイルシステムのリストと比較し、必要な変更を行うことにより、autofs マウントを更新します。こうすることにより、システム管理者は auto_master 内のマウント情報を変更し、autofs デーモンを停止したり、再起動することなく、それらの変更結果を autofs プロセスに使用させることができます。ファイルシステムがマウントされれば、以後のアクセスに automountd は不要となります。次に automountd が必要になるのは、ファイルシステムが自動的にアンマウントされるときです。

mount とは異なり、automount はマウントすべきファイルシステムを調べるためにファイル /etc/vfstab (これは各コンピュータごとに異なる) を参照しません。automount コマンドは、ドメイン内とコンピュータ上で名前空間とローカルファイルを通して制御されます。

autofs のしくみの概要を簡単に説明します。

自動マウントのデーモンである automountd は、ブート時に /etc/init.d/autofs スクリプトから起動されます (図 5–1 を参照してください)。このスクリプトは automount コマンドも実行します。このコマンドはマスタマップを読み取り (116ページの「autofs のナビゲーションプロセス開始法 (マスタマップ)」 を参照)、autofs のマウントポイントをインストールします。

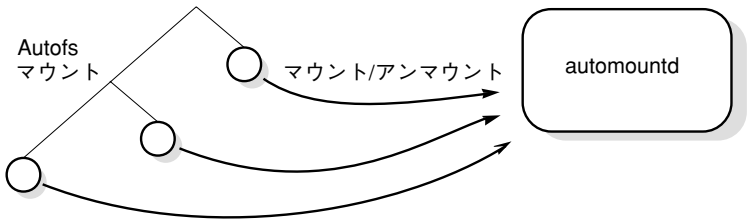

図 5–1　/etc/init.d/autofs スクリプトによる automount の起動

autofs は、自動マウント操作とアンマウント操作をサポートするカーネルファイルシステムの 1 つです。

autofs のマウントポイントにおいてファイルシステムへのアクセス要求が出された場合、autofs は次のように機能します。

1. autofs がその要求に介入します。
2. autofs は要求されたファイルシステムをマウントするよう、automountd にメッセージを送信します。
3. automountd がマップからファイルシステム情報を見つけ、マウントを実行します。
4. autofs は、介入した要求の実行を続行させます。
5. 一定時間そのファイルシステムがアクセスされないと、autofs はそれをアンマウントします。

注 - autofs サービスによって管理されるマウントは、手動でマウントしたりアンマウントは行わないでください。たとえこの操作がうまくいったとしても、autofs サービスはオブジェクトがアンマウントされたことを認識しないので、一貫性が損なわれる恐れがあります。リブートによって、autofs のマウントポイントがすべて消去されます。

---

autofs のネットワークナビゲート (マップ)

autofs は一連のマップを探索することによって、ネットワークをナビゲートします。マップとは、ネットワーク上の全ユーザのパスワードエントリ、またはネットワーク上の全ホストコンピュータの名前などの情報が収められているファイルです。つまり、UNIX 管理ファイルのネットワーク版といえます。マップはローカルに使用するか、または NIS や NIS+ のようなネットワークネームサービスを通じて使用できます。Solstice システム管理ツールを使用して、ユーザは自分の環境ニーズに適合するマップを作成します。126ページの「autofs のネットワークナビゲート法の変更 (マップの変更)」 を参照してください。

autofs のナビゲーションプロセス開始法 (マスタマップ)

automount コマンドはシステムの起動時にマスタマップを読み取ります。図 5–2 に示すように、マスタマップ内の各エントリは、直接または間接のマップ名、そのパス、およびそのマウントオプションです。エントリの順序は重要ではありません。automount は、マスタマップ内のエントリとマウントテーブル内のエントリを比較して、現在のリストを生成します。

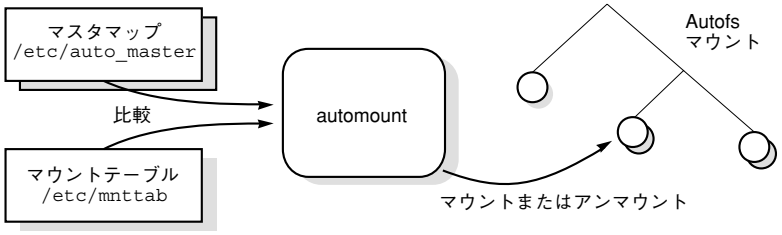

図 5–2　マスタマップによる探索

autofs マウントプロセス

マウント要求が発生したときに autofs サービスが何を実行するかは、オートマウンタマップの設定によって異なります。マウントプロセスの基本はすべてのマウントで同じですが、指定されているマウントポイントとマップの複雑さによって結果が変わります。Solaris 2.6 ではマウントプロセスも変更され、トリガーノードも作成されるようになりました。

単純な autofs マウント

autofs マウントプロセスの説明のために、以下のファイルがインストールされていると仮定します。

```
$ cat /etc/auto_master
# Master map for automounter
#
+auto_master
/net   -hosts   -nosuid,nobrowse
/home   auto_home  -nobrowse
/xfn   -xfn
/share  auto_share
$ cat /etc/auto_share
# share directory map for automounter
#
ws    gumbo:/export/share/ws
```

/share ディレクトリがアクセスされると、autofs サービスは /share/ws に対するトリガーノードを作成します。これは、/etc/mnttab の中では以下のようなエントリとなります。

```
-hosts  /share/ws     autofs  nosuid,nobrowse,ignore,nest,dev=###
```

/share/ws ディレクトリがアクセスされると、autofs サービスは以下の手順を実行します。

1. サーバのマウントサービスが有効かどうかを確認するために、サービスに対して ping を行います。
2. 要求されたファイルシステムを、/share の下にマウントします。これで、/etc/mnttab ファイルには以下のエントリが追加されます。

```
-hosts   /share/ws      autofs  nosuid,nobrowse,ignore,nest,dev=###
gumbo:/export/share/ws /share/ws    nfs    nosuid,dev=####     #####
```

階層型マウント

オートマウンタファイルに複数の層が定義されていると、マウントプロセスは多少複雑になります。上記の例の /etc/auto_shared ファイルを拡張して以下のエントリを追加したとします。

```
# share directory map for automounter
#
ws        /       gumbo:/export/share/ws
                  /usr    gumbo:/export/share/ws/usr
```

この場合、/share/ws マウントポイントがアクセスされたときのマウントプロセスは基本的に最初の例と同じです。また、/share/ws ファイルシステムの中に次のレベル (/usr) へのトリガーノードを作成することにより、そのレベルがアクセスされたときにマウントできるようにします。 この例でトリガーノードが作成されるためには、NFS に /export/share/ws/usr が存在している必要があります。

注意 - 階層を指定するときに、–soft オプションは使用しないでください。 この制限に関する説明は、118ページの「autofs アンマウント」を参照してください。

autofs アンマウント

一定時間アクセスがないためにアンマウントされるときは、マウントと反対の順序で実行されます。あるディレクトリより上位のディレクトリが使用中であれば、それより下のディレクトリだけがアンマウントされます。 アンマウントすると、トリガーノードがすべて削除され、ファイルシステムがアンマウントされます。ファイルシステムが使用中であれば、アンマウントは失敗してトリガーノードは再インストールされます。

注意 - 階層的にマウントを指定する場合は、–soft オプションは使用しないでください。–soft オプションを使用すると、トリガーノードを再インストールする要求がタイムアウトすることがあります。トリガーノードを再インストールできないと、マウントの次の階層にアクセスできません。この問題は、オートマウンタがすべての階層の構成要素をアンマウントすることでしか回避できません。具体的には、ファイルシステムが自動的にアンマウントされるのを待つか、システムをリブートすることになります。

autofs がクライアント用の最も近い読み取り専用ファイルを選択する方法 (複数ロケーション)

次のような直接マップの例では、

```
/usr/local          -ro \
    /bin                    ivy:/export/local/sun4\
    /share                  ivy:/export/local/share\
    /src                    ivy:/export/local/src
/usr/man            -ro   oak:/usr/man \
                          rose:/usr/man \
                          willow:/usr/man
/usr/games          -ro   peach:/usr/games
/usr/spool/news     -ro   pine:/usr/spool/news \
                          willow:/var/spool/news
```

マウントポイント /usr/man と /usr/spool/news には、複数のロケーション (/usr/man には 3 つ、/usr/spool/news には 2 つ) が記述されています。このような場合、複製された位置のどれからマウントしてもユーザは同じサービスを受けられます。ユーザの書き込みまたは変更が可能ならば、その変更をロケーション全体で管理しなければならなくなるので、この手順は、読み取り専用のファイルシステムをマウントするときにだけ意味があります。あるときに、あるサーバ上のファイルを変更し、またすぐに別のサーバ上で「同じ」ファイルを変更しなければならないとしたら、大変面倒な作業になります。この利点は、最も利用しやすいサーバが、そのユーザの手をまったく必要としないで自動的にマウントされるということです。

ファイルシステムを複製として設定してあると (72ページの「複製されたファイルシステムとは」 を参照してください)、クライアントは障害時回避機能を使用できます。最適なサーバが自動的に決定されるだけでなく、そのサーバが使用できなくな

るとクライアントは自動的に 2 番めに適したサーバを使います。障害時回避機能は、Solaris 2.6 の新機能です。

複製として設定するのに適しているファイルシステムの例は、マニュアルページです。大規模なネットワークでは、複数のサーバがマニュアルページをエクスポートできます。どのサーバからマニュアルページをマウントしても、そのサーバが動作しており、しかもそのファイルシステムをエクスポートしている限り、問題ありません。上の例では、複数のマウント位置は、マップエントリ内のマウント位置のリストになっています。

```
/usr/man -ro oak:/usr/man rose:/usr/man willow:/usr/man
```

これは、サーバをコンマで区切ったリストにし、その後にコロンとパス名を指定することによって入力することもできます (複製されるサーバすべてでパス名が同じ場合に限ります)。

```
/usr/man -ro oak,rose(1),willow(2):/usr/man
```

これで、サーバ `oak`、`rose`、`willow` のどれからでもマニュアルページをマウントできます。どのサーバが最適であるかは、いくつかの要素によって決まります。具体的には、ある特定の NFS プロトコルレベルをサポートしているサーバの数、サーバのとの距離、重み付けです。

順位を決定するときには、NFS バージョン 2 と NFS バージョン 3 のプロトコルをサポートしているサーバの数が数えられます。サポートしているサーバの数が多いプロトコルがデフォルトになります。そのため、クライアントにとっては利用できるサーバの数が最大になります。

プロトコルのバージョンが同じサーバの組の中で数が最も多いものが分かると、サーバのリストが距離によってソートされます。ローカルサブネット上のサーバには、リモートサブネット上のサーバよりも高い優先順位が付けられます。最も近いサーバが優先されることにより、待ち時間が短縮されネットワークトラフィックは軽減されます。94 ページの 図 5–3 に、サーバとの距離を図示します。

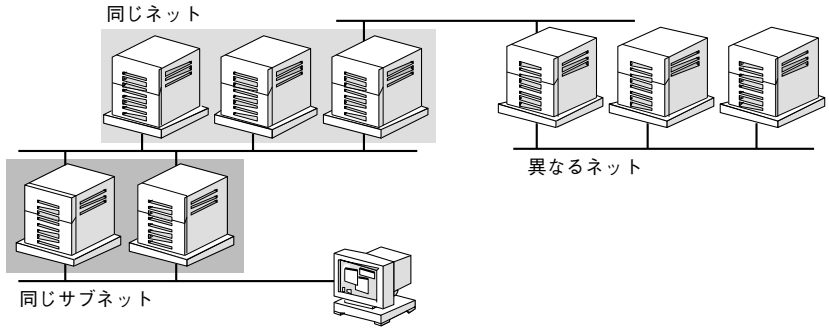

図 5–3　サーバとの距離

ローカルサブネット上に同じプロトコルをサポートしているサーバが複数あるときは、それぞれのサーバに接続する時間が計測され、速いものが使われます。優先順位には、重み付けも関係します (122ページの「autofs と重み付け」 を参照してください)。

バージョン 3 のサーバの方が多いと、優先順位の決定は複雑になります。通常、ローカルサブネット上のサーバはリモートサブネット上のサーバよりも優先されます。バージョン 2 のサーバがあり、それが最も近いバージョン 3 サーバよりも近いと状況が複雑になる可能性があります。ローカルサブネットにバージョン 2 サーバがあり、最も近いバージョン 3 サーバがリモートサブネット上にあると、バージョン 2 サーバが優先されます。このことは、バージョン 3 サーバの方がバージョン 2 サーバよりも多い場合にしかチェックされません。バージョン 2 サーバの方が多いと、バージョン 2 サーバしか選択されません。

障害時回避機能を指定していると、この優先順位はマウント時に 1 回、マウントするサーバを選択するときにチェックされ、その後は選択されたサーバが使用できなくなるたびにチェックされます。複数位置を指定しておくと、個々のサーバが一時的にファイルシステムをエクスポートできないときに便利です。

多くのサブネットを持つ大規模なネットワークでは、この機能は特に便利です。autofs は最も近いサーバを選択するため、NFS のネットワークトラフィックをローカルネットワークセグメントに制限します。複数のネットワークインタフェースを持つサーバの場合は、それぞれが別々のサーバであるとみなして、各ネットワークインタフェースに対応付けられているホスト名を指定します。autofs はそのクライアントに一番近いインタフェースを選択します。

autofs と重み付け

距離のレベルが同じサーバから 1 つを選択するために、autofs マップに重み付けの値を追加することができます。たとえば次のようにします。

```
/usr/man -ro oak,rose(1),willow(2):/usr/man
```

括弧内の数値が重み付けを表します。重み付けのないサーバの値はゼロです (選択される可能性が最高です)。重み付けの値が大きいほど、そのサーバが選択される可能性は低くなります。

注 - 重み付けは、サーバの選択に関係する要素の中で最も小さい影響力しかありません。ネットワーク上の距離が同じサーバの間で選択を行う場合に考慮されるだけです。

マップエントリ内の変数

変数名の前にドル記号 ($) を付けることによって、クライアント固有の変数を作成できます。この機能は、同じファイルシステムの位置にアクセスする異なるアーキテクチャタイプの調整に役立ちます。変数名を括弧でくくることで、そのうしろに続く文字や数字と変数とを区切ることができます。表 5–2 に定義済みのマップ変数を示します。

表 5–2 定義済みのマップ変数

変数	意味	情報提供元	例
ARCH	アーキテクチャタイプ	uname -m	sun4
CPU	プロセッサタイプ	uname -p	sparc
HOST	ホスト名	uname -n	dinky
OSNAME	オペレーティングシステム名	uname -s	SunOS

表 5–2 定義済みのマップ変数 続く

変数	意味	情報提供元	例
OSREL	オペレーティングシステムのリリース	uname -r	5.4
OSVERS	オペレーティングシステムのバージョン (リリースのバージョン)	uname -v	FCS1.0

キーとして使用する場合を除いて、変数はエントリ行内のどこにでも使用できます。たとえば、SPARC と x86 のアーキテクチャ用のバイナリを、それぞれ /usr/local/bin/sparc と /usr/local/bin/x86 からエクスポートしているファイルサーバがある場合、クライアントは次のようにマップエントリを通じてマウントできます。

```
/usr/local/bin  -ro server:/usr/local/bin/$CPU
```

これで、すべてのクライアント上の同じエントリがすべてのアーキテクチャに適用されます。

注 - どの sun4 アーキテクチャ向けに書かれたアプリケーションでも、ほとんどはすべての sun4 プラットフォームで実行できます。したがって、–ARCH 変数は sun4m でも sun4c でもなく、sun4 に固定されています。

他のマップを参照するマップ

ファイルマップで使用されたマップエントリ *+mapname* により、automount は指定されたマップを、あたかも現在のマップに組み込まれているかのように読み取ります。*mapname* の前にスラッシュがない場合、autofs はそのマップ名を文字列として扱い、ネームサービススイッチ方式を使用してこれを検出します。パス名が絶対パス名の場合、automount はその名前のローカルマップを捜します。マップ名がダッシュ「-」で始まる場合、automount は xfn や hosts といった適切な組み込みマップを参照します。

このネームサービススイッチファイルには、automount と指定されたautofs 用のエントリが収められています。そしてそのエントリには、ネームサービスが検索される順序が収められています。ネームサービススイッチファイルの例を次に示します。

```
#
# /etc/nsswitch.nis:
#
# An example file that could be copied over to /etc/nsswitch.conf;
# it uses NIS (YP) in conjunction with files.
#
# "hosts:" and "services:" in this file are used only if the /etc/netconfig
# file contains "switch.so" as a nametoaddr library for "inet" transports.
# the following two lines obviate the "+" entry in /etc/passwd and /etc/group.
passwd:         files nis
group:          files nis

# consult /etc "files" only if nis is down.
hosts:          nis [NOTFOUND=return] files
networks:       nis [NOTFOUND=return] files
protocols:      nis [NOTFOUND=return] files
rpc:            nis [NOTFOUND=return] files
ethers:         nis [NOTFOUND=return] files
netmasks:       nis [NOTFOUND=return] files
bootparams:     nis [NOTFOUND=return] files
publickey:      nis [NOTFOUND=return] files
netgroup:       nis
automount:      files nis
aliases:        files nis
# for efficient getservbyname() avoid nis
services:       files nis
```

この例では、まずローカルマップ、次に NIS マップが検索されます。したがって、最も頻繁にアクセスされるホームディレクトリに対してはローカルマップ /etc/auto_home に少数のエントリを登録しておき、その他のエントリについてはネームサービススイッチを使って NIS マップに戻るという方法が可能です。

```
bill                cs.csc.edu:/export/home/bill
bonny               cs.csc.edu:/export/home/bonny
```

組み込まれたマップを参照したあと、一致するものがなければ、automount は現在のマップの走査を続けます。これは、+ エントリのあとにさらにエントリを追加できることを意味します。たとえば、

```
bill                cs.csc.edu:/export/home/bill
bonny               cs.csc.edu:/export/home/bonny
+auto_home
```

組み込まれたマップは、ローカルファイル (ローカルファイルだけが + エントリを持つことができることに注意) または組み込みマップとすることができます。

```
+auto_home_finance      # NIS+ map
+auto_home_sales        # NIS+ map
+auto_home_engineering  # NIS+ map
+/etc/auto_mystuff      # local map
+auto_home               # NIS+ map
+-hosts                 # built-in hosts map
```

注 - NIS+ または NIS のマップでは「+」エントリを使用できません。

実行可能な **autofs** マップ

autofs マウントポイントを生成するコマンドを実行する autofs マップを作成することもできます。データベースやフラットファイルから autofs 構造を作成しなければならない場合には、実行可能な autofs マップが有効なことがあります。短所は、マップをすべてのホストにインストールしなければならないことです。実行可能なマップは、NIS と NIS+ のどちらのネームサービスにも含めることができません。

実行可能マップは、`auto_master` ファイルにエントリが必要です。

```
/execute    auto_execute
```

実行可能マップの例を示します。

```
#!/bin/ksh
#
# executable map for autofs
#

case $1 in
          src)  echo '-nosuid,hard bee:/export1' ;;
esac
```

この例が機能するためには、ファイルが `/etc/auto_execute` としてインストールされ、実行可能ビットがオン (パーミッションが 744) になっている必要があります。これらの条件のときに次のコマンドを実行すると、

```
% ls /execute/src
```

bee から `/export1` ファイルシステムがマウントされます。

autofs のネットワークナビゲート法の変更 (マップの変更)

マップへのエントリを変更、削除、または追加して、ユーザの環境ニーズに合わせることができます。ユーザが必要とするアプリケーションやその他のファイルシステムがその位置を変更すると、マップはこれらの変更を反映しなければなりません。autofs のマップは、いつでも変更できます。automountd が次にファイルシステムをマウントしたときにその変更内容が有効となるかどうかは、変更したマップと変更内容によって決まります。

ネームサービスに対する autofs のデフォルトの動作

ブート時に、autofs は /etc/init.d/autofs にあるスクリプトを使用して起動され、マスタマップ auto_master が検索されます (次に説明する規則が適用されます)。

autofs は、/etc/nsswitch.conf ファイルの自動マウントエントリで指定されたネームサービスを使用します。ローカルファイルや NIS ではなく NIS+ が指定された場合、マップ名はすべてそのまま使用されます。NIS が選択されていて autofs が必要なマップを検出できず、1 つまたは複数の下線を含むマップを検出した場合、以前の NIS ファイル名を使えるようにするため、autofs はその下線をドットに変換します。次に autofs はもう 1 度マップを調べます。この手順を 図 5–4 に示します。

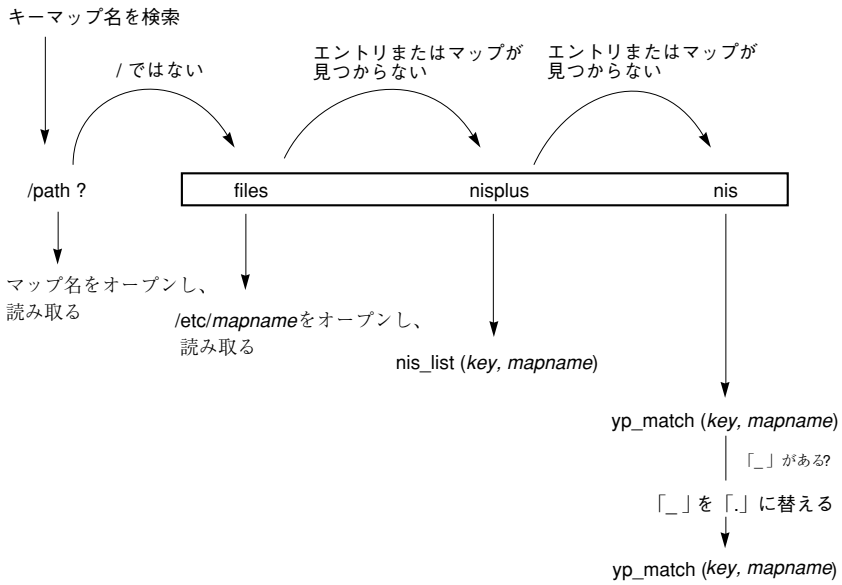

図 5–4　autofs によるネームサービスの使用

このセッションでの画面の動きは次の例のようになります。

```
$ grep /home /etc/auto_master
/home           auto_home

$ ypmatch brent auto_home
Can't match key brent in map auto_home.  Reason: no such map in
server's domain.

$ ypmatch brent auto.home
diskus:/export/home/diskus1/&
```

ネームサービスとして「ファイル」が選択された場合、すべてのマップは /etc ディレクトリ内のローカルファイルとみなされます。autofs は、使用するネームサービスとは無関係に、スラッシュで始まるマップ名をローカルとして解釈します。

autofs リファレンス情報

これ以降の節では、autofs の高度な機能を取り上げます。

メタキャラクタ

autofs は一部の文字を、特別な意味を持つものとして認識します。これらの文字には、置換に使用される文字や、他の文字を autofs のマップ構文解析機能から保護する目的のものがあります。

アンパサンド *(&)*

たとえば次のように、多数のサブディレクトリを指定したマップがある場合、

```
john        willow:/home/john
mary        willow:/home/mary
joe         willow:/home/joe
able        pine:/export/able
baker       peach:/export/baker
```

この場合、文字列の置換が使えます。アンパサンド文字 (&) を使用して、任意の位置に記述されたこのキーを置換することができます。アンパサンドを使用した場合、上のマップは次のようになります。

```
john        willow:/home/&
mary        willow:/home/&
joe         willow:/home/&
able        pine:/export/&
baker       peach:/export/&
```

キー置換はまた、次のような直接マップでも使用できます。

```
/usr/man      willow,cedar,poplar:/usr/man
```

これは次のようにも記述できます。

```
/usr/man      willow,cedar,poplar:&
```

なお、アンパサンドによる置換ではキー文字列全体を使用するため、直接マップでキーが / で始まる場合、そのスラッシュは引き継がれます。したがって、次のような指定はできません。

```
/progs    &1,&2,&3:/export/src/progs
```

その理由は、autofs がこれを次のように解釈するためです。

```
/progs     /progs1,/progs2,/progs3:/export/src/progs
```

アスタリスク *(*)*

任意のキーを一致させるのに、任意の文字を表す置換文字であるアスタリスク (*) を使用できます。このマップエントリを使って、すべてのホストから /export ファイルシステムをマウントできます。

```
*       &:/export
```

ここでは、各アンパサンドは特定のキーの値によって置換されています。autofs はこのアスタリスクをファイルの終わりとして解釈します。

特殊文字

特殊文字が含まれているマップエントリがある場合、autofs のマップ構文解析機能を混乱させるような名前のディレクトリについてはマウントする必要があります。autofs の構文解析機能は、名前に含まれるコロン、カンマ、スペースなどを認識します。これらの名前は二重引用符で囲んでください。

```
/vms    -ro    vmsserver: -  -  - "rc0:dk1 - "
/mac    -ro    gator:/ - "Mr Disk - "
```

付録 **A**

NFS の調整機能

NFS サービスの機能を高めるため、いくつかのパラメータを設定できます。これらのパラメータは /etc/system で定義することができ、システムのブート時に読み込まれます。各パラメータは、それが入っているカーネルモジュールの名前と、それを識別する記号名で識別できます。

警告 - 記号の名前、それらが常駐するモジュールおよびデフォルト値は、リリース版ごとに異なる場合があります。変更を加えたり、前のリリース版の値を適用するときは、あらかじめ稼働中の SunOS バージョンの関連ドキュメントを確認してください。

表 A–1 には、nfs モジュールの一部であるパラメータを示します。表 A–2 には、nfssrv モジュールの一部であるパラメータを示します。表 A–3 は、rpcmod モジュールの一部であるパラメータのリストです。136ページの「カーネルパラメータの値を設定する方法」 では、これらのパラメータの変更方法について説明します。/etc/system ファイルについて詳しくは、system(4) のマニュアルページを参照してください。

表 A–1 nfs モジュール用の NFS パラメータ

記号名	説明	デフォルト設定
authdes_win	この記号は AUTH_DES の使用時に、サーバとクライアント間で許容されるクロックの時間差を指定する。	デフォルトは 300 秒。
authkerb_win	この記号は AUTH_KERB の使用時に、サーバとクライアント間で許容されるクロックの時間差を指定する。	デフォルトは 300 秒。
nfs_acl_cache	この記号は NFS_ACL プロトコルを使用中のクライアントで、ACL をキャッシュするかどうかを制御する。	デフォルトはオフ (0)。これはおそらく安全に有効 (1) 設定でき、Solaris の将来のリリースではオンがデフォルトになる予定。
nfs_cots_timeo	NFS バージョン 2 クライアントによる接続指向トランスポートに対する操作の、 デフォルトのタイムアウト	600 × 1/10 秒。
nfs3_cots_timeo	NFS バージョン 3 クライアントによる接続指向トランスポートに対する操作の、 デフォルトのタイムアウト	600 × 1/10 秒。
nfs_do_symlink_cache	この記号は NFS バージョン 2 を使用してマウントしたファイルシステムで、シンボリックリンクをキャッシュするかどうかを制御する。	デフォルトはオン (1)。システムで amd などを使う場合には無効 (0) にできる。 これが無効に設定されていると、クライアントシステムのパフォーマンスが低下する場合がある。
nfs3_do_symlink_cache	この記号は NFS バージョン 3 を使ってマウントしたファイルシステムで、シンボリックリンクをキャッシュするかどうかを制御する。	デフォルトはオン (1)。これは無効 (0) に設定できるが、クライアントシステムのパフォーマンスが低下する可能性がある。
nfs_dynamic	この記号は NFS バージョン 2 を使ってマウントしたファイルシステムで、動的再転送のサポートを使用するかどうかを制御する。	デフォルトはオン (1)。 安全にオフ (0) にできるが、処理の遅いサーバや 8K バイトの読み書き転送を完全にサポートできないサーバでは、相互運用に問題が生じる可能性がある。

表 A–1 nfs モジュール用の NFS パラメータ 続く

記号名	説明	デフォルト設定
nfs3_dynamic	この記号は NFS バージョン 3 を使ってマウントしたファイルシステムで、動的再転送のサポートを使用するかどうかを制御する。	デフォルトはオフ (0)。これは変更しないこと。
nfs_lookup_neg_cache	この記号は NFS バージョン 2 を使ってマウントしたファイルシステムで、失敗した検索要求をキャッシュさせるかどうかを制御する。	デフォルトはオフ (0)。 おそらく安全に有効 (1) に設定できるが、正常なディレクトリ名のキャッシュ処理に悪影響が生じることもある。
nfs3_lookup_neg_cache	この記号は NFS バージョン 3 を使ってマウントしたファイルシステムで、失敗した検索要求をキャッシュさせるかどうか制御をする。	デフォルトはオフ (0)。 おそらく安全に有効 (1) に設定できるが、正常なディレクトリ名のキャッシュ処理に悪影響が生じる可能性がある。
nfs_max_threads	この記号は NFS バージョン 2 を使ってマウントしたファイルシステムごとに、非同期スレッドを起動する最大数を制御する。	デフォルトは 8。この数字はファイルシステムごとのスレッド数に影響するので、ファイルシステムの多いクライアントでは、大きな変更を行うとパフォーマンスを大幅に劣化させる原因になる。
nfs3_max_threads	この記号は NFS バージョン 3 を使ってマウントしたファイルシステムごとに、非同期スレッドを起動させる最大数を制御する。	デフォルトは 8。この数字はファイルシステムごとのスレッド数に影響するので、ファイルシステムの多いクライアントでは、大きな変更を行うとパフォーマンスを大幅に劣化させる原因になる。
nfs3_max_transfer_size	この記号は NFS バージョン 3 のクライアントファイルのブロックサイズを制御する。	デフォルトは 32K バイト。変更はできるだけ行わないこと。
nfs_nra	この記号は NFS バージョン 2 を使ってマウントしたファイルシステムで、読み込まれる先読みブロックの数を制御する。	デフォルトは 4。値を大きくしてもパフォーマンスは向上せず、クライアント側でのメモリ利用が増大する。

表 A–1 nfs モジュール用の NFS パラメータ 続く

記号名	説明	デフォルト設定
nfs3_nra	この記号は NFS バージョン 3 を使ってマウントしたファイルシステムについて、読み込まれる先読みブロックの数を制御する。	デフォルトは 4。値を大きくしてもパフォーマンスは向上せず、クライアント側でのメモリ利用が増大する。
nrnode	この記号はキャッシュされる NFS rnode の数を制御する。	この記号に割り当てられる値はブート時に構成され、サーバに適合するようスケーリングされる。1 に設定してキャッシュ処理を無効にできる。
nfs_shrinkreaddir	この記号は回線上で、NFS バージョン 2 READDIR 要求を 1024 バイトに縮小するかどうかを制御する。一部古い NFS バージョン 2 のサーバの中には、1024 バイト以上の READDIR 要求を正しく処理できないものがある。	デフォルトはオフ (0) で、この場合 READDIR 要求を削減しない。これは安全に有効 (1) に設定できるが、ディレクトリの読み込み中にパフォーマンスに悪影響を与える恐れがある。
nfs_write_error_interval	この記号は NFS ENOSPC によるエラーメッセージの書き込みがログされる頻度を、秒単位で制御する。	デフォルトは 5。
nfs_write_error_to_cons_only	この記号は NFS 書き込みのエラーメッセージが、システムコンソールか、システムコンソールおよび syslog にログされるかどうかを制御する。	デフォルトはオフ (0) で、この場合 NFS 書き込みエラーメッセージをすべて、システムコンソールと syslog にログする。この機能を有効 (1) に設定すると、NFS 書き込みエラーメッセージのほとんどが、システムコンソールでしか出力されないことになる。

表 A–2 nfssrv モジュール用の NFS パラメータ

記号名	説明	デフォルト設定
nfs_portmon	この記号は IP ポート番号に基づき、NFS サーバで要求のフィルタ処理を行うかどうかを制御する。予約済みポート番号のバークレー表記法を使用する。	デフォルトはオフ (0)。有効 (1) にできるが、相互運用上の問題が発生する恐れがある。
nfsreadmap	この記号は現在ではアクティブではなく、今ではマップの読み込みは実装していない。移行を容易にするため残されている。	デフォルトはオフ (0)。
rfs_write_async	この記号は書き込みの処理能力を安全に高めるため、NFS バージョン 2 サーバが、書き込みのクラスタ化機能を使用するかどうかを制御する。	デフォルトはオン (1)。無効 (0) に設定できるが、性能は低下する場合がある。

表 A–3 rpcmod モジュール用の NFS パラメータ

記号名	説明	デフォルト設定
authdes_cachesz	authdes 応答キャッシュのサイズ。これは、セキュリティ保護された RPC 要求すべてについてクライアントの資格を確認しなくても済むようにする、パフォーマンスを向上のための機能拡張。	デフォルトは 128。この値を大きくし過ぎると、システムのパフォーマンスが低下する可能性がある。
authkerb_cachesz	authkerb 応答キャッシュのサイズ。これは、セキュリティ保護された RPC 要求すべてについてクライアントの資格を確認しなくても済むようにする、パフォーマンスを向上のための機能拡張。	デフォルトは 128。この値を大きくし過ぎると、システムのパフォーマンスが低下する可能性がある。
svc_ordrel_timeout	カーネルが接続を強制的に切断するまでの時間 (ミリ秒)。カーネル RPC に使われている TCP 接続が、万一終了の途中でハングした場合に使われる。接続がハングする原因には、NFS サーバが接続に対する正常なクローズ (FIN) を試み、クライアントがそのクローズの完了 (FIN 確認) ハンドシェークに失敗した場合が考えられる。	デフォルトは 600000 ミリ秒 (10 分)。値が小さすぎると、TCP 接続を閉じるのに十分な時間がクライアントに与えられない。大きすぎると、欠陥のあるクライアントや害を及ぼすクライアントがサーバと TCP 接続を結ぶことになる。

カーネルパラメータの値を設定する方法

1. **root** になります。

2. /etc/system ファイルを編集し、行を追加してパラメータを設定します。

   それぞれのエントリは次の形式に従ってください。

   set *module*:*symbol*=*value*

   ここで、*module* は、必要なパラメータを含むカーネルモジュールの名前であり、*symbol* はパラメータの名前、*value* はパラメータに割り当てる数値です。以下に例を上げます。

   set nfs:nfs_nra=4

これで NFS バージョン 2 を使ってマウントしたファイルシステムについて、読み込まれる先読みブロック数が変更になります。

3. システムをリブートします。

索引

記号

#
- 間接マップのコメント, 113
- 直接マップのコメント, 111
- マスタマップのコメント (auto_master), 108

/
- root ディレクトリ、ディスクレスクライアントによるマウント, 9
- が前に付いたマスタマップ名, 108
- マスタ~マップのマウントポイント /-, 107, 111

A

already mounted メッセージ, 101
ARCH マップ変数, 122
autofs, ix, 84, 129
- default behavior, 126
- /home ディレクトリの構造, 90, 91
- NFS URL and, 97
- アンマウントプロセス, 118
- オペレーティングシステム、互換性のないバージョンのサポート, 95, 96
- 概要, 9
- 機能, 9, 10
- 共有名前空間へのアクセス, 94, 95
- 公共ファイルハンドル, 97
- 作業と手順, 84
- 障害追跡, 100, 103
- タスクと手順, 94
- デフォルトの動作, 127
- 特殊文字, 129
- 名前空間データ, 9, 10
- ネームサービスの使用法, 126
- 非 NFS ファイルシステムへのアクセス, 88, 89
- 表示機能, 97
- ファイルシステムのマウント, 17
- 複数サーバ間での共有ファイルの複製, 96
- プロジェクト関連ファイルの統合, 92, 94
- ホームディレクトリサーバの設定, 91, 92
- マウントプロセス, 117, 118
- マップ, 85, 107, 109 - 112, 114, 116, 119, 121 - 123, 125 - 127
- メタキャラクタ, 128
- リファレンス, 127, 129

autofs スクリプト, 115
automountd デーモン
- autofs と, 9
- automount コマンドと, 115
- 動作のしくみ, 114, 115

automount コマンド
- -v オプション, 100, 101
- autofs と, 9
- automountd デーモンと, 115
- エラーメッセージ, 100, 103
- 実行の判断, 85
- 動作のしくみ, 114

auto_home map
- マウントポイント /home, 108

auto_home マップ
- /home ディレクトリサーバの設定, 91, 92
- /home ディレクトリの構造, 90, 91
- マウントポイント /home, 107

-a オプション
showmount コマンド, 62
umount コマンド, 52

B

bad argument specified with index option, 37
bad key メッセージ, 100

C

cache file system type
autofs access using, 89
CacheFS, 89
cannot receive reply メッセージ, 103
cannot send packet メッセージ, 102
cannot use index option without public option, 38
can't mount メッセージ, 101
CD-ROM アプリケーション、アクセス, 88
cfsadmin コマンド, 89
chkey コマンド, 22
clear_locks コマンド, 48
couldn't create mount point メッセージ, 101
CPU マップ変数, 122
cred テーブル、公開鍵, 77

D

daemons
automountd, 115
dfsmounts コマンド, 61
dfstab ファイル
kerberos オプション, 24
secure オプション, 23
構文, 15
ファイルの自動共有, 14, 15
DH 認証, ix
dfstab ファイルのオプション, 23
KERB 認証, 24, 78
概要, 77, 78
パスワードによる保護, 76
ユーザ単位の認証, 75
directory does not exist, 39
dir must start with '/' メッセージ, 102
DOS ファイル、アクセス, 89

E

/etc/.rootkey ファイル, 24
/etc/default/fs ファイル, 44
/etc/dfs/dfstab ファイル
kerberos オプション, 24
secure オプション, 23
構文, 15
ファイルの自動共有, 14, 15
/etc/dfs/fstypes ファイル, 44
/etc/dfs/sharetab ファイル
mountd デーモンと, 46
説明, 44
/etc/init.d/autofs スクリプト, 115
/etc/mnttab ファイル
auto_master マップとの比較, 115
作成, 62
説明, 43
/etc/nfssec.conf ファイル, 44
/etc/rmtab ファイル, 44
/etc/services
nfsd エントリ, 38
/etc/vfstab ファイル
automount コマンドと, 115
NFS サーバと, 16
説明, 44
ディスクレスクライアントによるマウント, 9
ブート時のファイルシステムのマウント, 16
exports メッセージ, 102

F

file too large メッセージ, 38
fstypes ファイル, 44
fs ファイル, 44
fuser -k マウントポイント, 54
-F オプション、unshareall コマンド, 61

G

GSS-API, 8
-g オプション、lockd, 45

H

hierarchical mountpoints メッセージ, 102

/home ディレクトリ
    構造, 90, 91
    サーバの設定, 91, 92
host not responding メッセージ, 102
-hosts 特殊マップ, 108
HOST マップ変数, 122
-h オプション、umountall コマンド, 54

I

index オプション
    must be a file エラーメッセージ, 37
    public オプションが指定されていないエラーメッセージ, 38
    WebNFS と, 26
    without public option エラーメッセージ, 38
intr オプション、mount コマンド, 28

K

kerbd デーモン, 78
kerberos、dfstab ファイルのオプション, 24
Kerberos (KERB) 認証, 24, 78
KERB 認証, ix
    dfstab ファイルのオプション, 24
    NFS と, 7
    概要, 78
/kernel/fs ファイル、検査, 44
keylogin プログラム
    実行, 23
    リモートログインでのセキュリティの問題, 79
keylogout プログラム, 79
keyserv デーモン、確認, 22
ksh コマンド, 7

L

largefiles オプション、mount コマンド, 50
leading space in map entry メッセージ, 101
lockd デーモン
    構文, 45
    説明, 45
login コマンド、リモートログイン, 79
-l オプション、umountall コマンド, 54

M

mail コマンド, 7
map key bad メッセージ, 102
MIT Athena プロジェクト, 78
mnttab ファイル
    auto_master マップとの比較, 115
    作成, 62
    説明, 43
mountall コマンド, 53
mountd デーモン
    rpcbind に未登録, 39
    サーバからの応答のチェック, 32
    実行の確認, 34, 39
    説明, 46
    リブートなしでの起動, 34
    リモートマウントの必要条件, 28
mount コマンド, ix, 48, 52
    autofs と, 9
    NFS URL の使用, 52
    オプション, 20, 49 - 52, 69
    障害, 52
    スーパーユーザとしての使用, 16
    説明, 48, 49
    ディスクレスクライアントでの必要条件, 9
    の使用, 51
mount コマンドの -O オプション, 51
mount コマンドの -o フラグに対する bg オプション, 49
mount コマンドの -o フラグに対する fg オプション, 49
mount コマンドの -o フラグに対する hard オプション, 51
mount コマンドの -o フラグに対する soft オプション, 51
MS-DOS ファイル、アクセス, 89

N

newkey コマンド, 22
nfscast: cannot receive reply メッセージ, 103
nfscast: cannot send packet メッセージ, 102
nfscast: select メッセージ, 103
nfsd デーモン
    構文, 46
    実行の確認, 33
    説明, 46

マウント, 70
リブートなしでの起動, 35
リモートマウントの必要条件, 28
nfsd デーモンの -c オプション, 46
nfssec.conf ファイル, 44
nfsstat コマンド, 36, 63
NFS URL
autofs and, 97
mount コマンドの例, 52
～を使用したマウント, 8, 21
NFS URL、ブラウズ, 27
NFS 環境, 4, 6
概要, 4
サーバとクライアント, 3
セキュリティ保護されている NFS システム, 75, 76
バージョン 2 プロトコル, 5
バージョン 3 プロトコル, 6
ファイルシステム, 4
利点, 5
NFS サーバ、現在の～の識別, 36
NFS サービス
起動, 17
再起動, 34
停止, 18
NFS 障害追跡, 28
NFS サービスが失敗した箇所の決定, 34
サーバの問題, 29
ハングしたプログラム, 40
方針, 28, 29
リモートマウントの問題, 39
NFS のバージョン
ネゴシエーション, 68
NFS ロック
クライアント側障害時回避機能, 73
nisaddcred コマンド, 22
nistbladm コマンド, 86, 87
nis_err メッセージ, 102
no info メッセージ, 101, 103
nolargefiles オプション、mount コマンド, 50
No such file or directory メッセージ, 39
nosuid オプション
share コマンド, 57
Not a directory メッセージ, 102
Not found メッセージ, 101

O

OSNAME マップ変数, 122
OSREL マップ変数, 123
OSVERS マップ変数, 123
-o オプション
mount コマンド, 49, 51
share コマンド, 55, 58
-o オプションP
mount コマンド, 51

P

pathconf: no info メッセージ, 103
pathconf: server not responding メッセージ, 103
PC-DOS ファイル、アクセス, 89
Permission denied メッセージ, 39
processor type マップ変数, 122
pstack コマンド, 65
publickey マップ, 22, 77
public オプション
mount コマンド, 50
share エラーメッセージ, 41
WebNFS と, 26

R

remount メッセージ, 101
replicas must have the same version, 40
replicated mounts must be read-only, 40
replicated mounts must not be soft, 40
rlogin コマンド、リモートログイン, 79
rmtab ファイル, 44
root ディレクトリ、ディスクレスクライアントによるマウント, 9
ro オプション
mount コマンドの -o フラグ, 50, 51
share コマンドの -o フラグ, 55, 58
RPC
Secure, 76 - 79
認証, 77
rpcbind デーモン
mountd デーモンが未登録, 39
停止またはハング, 39
ワームスタート, 35
rpcinfo コマンド, 65
RPCSEC_GSS, 8

rw オプション
-o フラグの share オプション, 55
mount コマンドの -o フラグ, 50
share コマンドの -o フラグ, 58
-r オプション
mount コマンド, 51
umountall コマンド, 54

S

secure
dfstab ファイルのオプション, 23
マウントオプション, 23
Secure RPC
DH authorization issues, 79
DH 認証に関する事項, 78
概要, 76, 77
server:pathname のマウントエラー, 102
server not responding メッセージ, 101, 103
キーボード割り込み, 28
ハングしたプログラム, 40
リモートマウントの問題, 39
servers
autofs によるファイルの選択, 119
setgid モード、を禁止する share コマンドのオプション, 57
setmnt コマンド, 62
setuid のモード
Secure RPC と, 79
setuid モード
を禁止する share コマンドのオプション, 57
shareall コマンド, 61
sharetab ファイル
mountd デーモンと, 46
説明, 44
share コマンド, 54, 58
/etc/dfs/dfstab ファイルのエントリ, 14
オプション, 55
使用, 58
セキュリティの問題, 55, 57
説明, 54
share コマンドの -o フラグに対する rw=client オプション, 55
share コマンドの anon オプション, 56
share コマンドの root=host オプション, 57
showmount コマンド, 61
showmount コマンドの -d オプション, 62
showmount コマンドの -e オプション, 62
slash (/)
root ディレクトリ、ディスクレスクライアントによるマウント, 9
snoop コマンド, 67
Solaris 2.5
NFS バージョン 2 でのサポート, 5
NFS バージョン 3 の改良点, 6
statd デーモン, 47

T

TCP、NFS バージョン 3 と, 7
telnet コマンド、リモートログイン, 79
truss コマンド, 68
-t オプション、lockd デーモン, 45

U

UDP、NFS バージョン 3 と, 7
umountall コマンド, 54
umountall コマンドの -h オプション, 54
umountall コマンドの -k オプション, 54
umountall コマンドの -s オプション, 54
umount コマンド, ix
autofs と, 9
説明, 52
UNIX
セキュリティの問題, 77
UNIX 認証, 75, 77
unshareall コマンド, 61
unshare コマンド, 60
/usr/kvm ディレクトリ、ディスクレスクライアントによるマウント, 9
/usr ディレクトリ、ディスクレスクライアントによるマウント, 9

V

vfstab ファイル
automount コマンドと, 115
NFS サーバと, 16
説明, 44
ディスクレスクライアントによるマウント, 9
ブート時のファイルシステムのマウント, 16
-v オプション

automount コマンド, 100, 101
-V オプション
umount コマンド, 52

W

WARNING: mountpoint already mounted on メッセージ, 101
WebNFS, 8, 73
index オプション, 26
NFS URL, 25
計画, 25
有効化, 26

あ

アクセス権, ix
NFS バージョン 3 の改良点, 6
リストにコンピュータがない, 39
アクセス制御リスト (ACL), 6
アプリケーション、ハングした, 40
暗号解除, ix
アンマウント
autofs, 118
autofs と, 9
ファイルシステムのグループ, 54
例, 53, 54

い

一覧表示
共有ファイルシステム, 58
マウントされているファイルシステム, 52
リモートマウントされたファイルシステムを持つクライアント, 61

え

エラー, ix
オープンエラー, 6
書き込みエラー, 6
エラーメッセージ, ix
No such file or directory, 39
Permission denied, 39
automount -v の場合, 100, 101
miscellaneous automount メッセージ, 103
server not responding, 28, 39, 40
サーバが応答しない, 51
その他の automount メッセージ, 102

お

大型ファイル, 73
NFS によるサポート, 7
有効化, 18
オープンエラー, 6
オペレーティングシステム
互換性のないバージョンのサポート, 95, 96
マップ変数, 122

か

カーネル、サーバの応答の検査, 29
階層型マウント (複数マウント), 118
書き込みエラー, 6
間接マップ, ix
automount コマンド実行の判断, 86
概要, 112, 114
構文, 112, 113
コメント, 113
説明, 85
変更, 87
例, 113, 114
管理
NFS ファイルとその機能, 43, 44
管理者の責任, 14

き

キーサーバ、起動, 23
キャッシュと NFS バージョン 3, 6
「キャッシュ」ファイルシステムタイプ
を使った autofs アクセス, 89
共通鍵, 78
共有されるリソース、リスト, 44

く

クライアント
NFS サービス, 3
互換性のないオペレーティングシステムのサポート, 95, 96
クライアント側障害時回避機能, 71
NFS によるサポート, 7

NFS ロック, 73
複製されたファイルシステム, 72
有効化, 19
用語, 72

こ
公開鍵方式暗号, ix
DH 認証, 77, 78
共通鍵, 78
公開鍵のデータベース, 76, 77
時刻同期, 77
対話鍵, 78
秘密鍵, 77, 79
公開ファイルハンドル
WebNFS, 25
公共ファイルハンドル
NFS マウント, 8
autofs, 97
マウント, 70
コマンド, ix
NFS コマンド, 47, 62
ハングしたプログラム, 40
コメント
間接マップの, 113
直接マップの, 111
マスタマップ (auto_master), 108
コンピュータ、再インストール、移動、アップグレード, 24
コンピュータのアップグレード, 24
コンピュータの移動, 24
コンピュータの再インストール, 24

さ
サーバ
NFS サーバと vfstab ファイル, 16
NFS サービス, 3
autofs によるファイルの選択, 121
共有ファイルの複製, 96
クラッシュと秘密鍵, 79
障害追跡, 29, 39
ホームディレクトリサーバの設定, 91, 92
保守, 14
マウント時に応答しない, 51
マップでの重み付け, 122
リモートマウントに必要なデーモン, 28

し
資格
UNIX 認証, 77
説明, 76
時刻、同期, 77
時刻の同期, 77
実行可能なマップ, 125
障害
mount コマンドの例, 52
障害時回避
NFS によるサポート, 7
エラーメッセージ, 38
障害追跡, ix
NFS, 28 - 30, 34, 39, 40
autofs, 88, 100, 101, 103
シングルユーザモードとセキュリティ, 79

す
スーパーユーザ
autofs とパスワード, 9
mount コマンドの実行, 16
スラッシュ (/)
が前に付いたマスタマップ名, 108
マスタマップのマウントポイント /-, 107, 111

せ
セキュリティ
DH 認証, 23, 75 - 78
KERB 認証, 24, 78
NFS バージョン 3 と, 6
Secure RPC, 76 - 78, 81
UNIX 認証, 75, 77
mount コマンド, 50
制約条件の適用, 96
セキュリティ保護された NFS システム, 21, 24
セキュリティ保護されている NFS システム, 75
ファイル共有の問題, 55, 57
セキュリティサービス、リスト, 44
セキュリティ方式, 8
セキュリティ保護された NFS システム
管理, 21, 24
設定, 22, 24

ドメイン名, 21
セキュリティ保護された NFS システムのドメイン名, 21
セキュリティ保護されている NFS システム
概要, 75

た

対話鍵, 78

ち

直接マップ
automount コマンド実行の判断, 86
概要, 110, 111
構文, 110, 111
コメント, 111
説明, 85
変更, 87
例, 110
直列アンマウント, 54

て

ディスクレスクライアント
NFS サービス, 4
手動マウントでの必要条件, 9
ブート時のセキュリティ, 79
デーモン
automountd, 9, 114, 115
kerbd, 78
keyserv, 22
lockd, 45
mountd, 28, 31, 34, 39, 46
nfsd, 28, 31, 33, 34, 46
rpcbind, 39
statd, 47
リモートマウントに必要な, 28
デフォルトのファイルシステムタイプ, 44

と

ドメインの定義, 21
トラブルシューティング
autofs, 102
トランスポート設定の問題
エラーメッセージ, 38
トランスポートプロトコル
ネゴシエーション, 69

な

名前空間
autofs and, 9
autofs と, 10
共有、アクセス, 94, 95

に

認証, ix
DH, 77, 78
KERB, 78
RPC, 77
UNIX, 75, 77

ね

ネームサービス
autofs による使用, 126
マップ保守の方法, 85
ネゴシエーション
NFS のバージョン, 68
トランスポートプロトコル, 69
ネットワークロックマネージャー, 7

は

バージョン 2 の NFS プロトコル
サポート, 5
バージョン 3 の NFS プロトコル, 6
バージョンのネゴシエーション, 68
パスワード
DH パスワードによる保護, 76
Secure RPC パスワードの作成, 22
autofs とスーパーユーザのパスワード, 9
バックグラウンドマウントオプション, 49
ハングしたプログラム, 40
番号記号 (#)
間接マップのコメント, 113
直接マップのコメント, 111
マスタマップのコメント (auto_master), 108

ひ

秘密鍵

サーバのクラッシュと, 79
データベース, 77
リモートサーバからの消去, 79
表記上の規則, xi
表示
共有またはエクスポートされたファイルのリスト, 62
リモートマウントされたディレクトリのリスト, 62
表示機能
無効化, 97

ふ

ファイアウォール
～を越えた NFS アクセス, 8
～を越えたマウント, 20
ファイルアクセス権, ix
NFS バージョン 3 の改良点, 6
リストにコンピュータがない, 39
ファイル共有, ix, 54, 61
NFS バージョン 3 の改良点, 6, 7
オプション, 55
概要, 54
共有解除, 60, 61
自動, 14, 15
セキュリティの問題, 55, 57, 75, 76
認証されていないユーザと, 56
複数サーバ間での共有ファイルの複製, 96
複数のファイルシステム, 61
読み書き可能アクセス, 55
読み書き可能アクセス権, 58
読み取り専用アクセス, 55, 58
リストのクライアントのみ, 55
ルートアクセス権の付与, 57
例, 58, 61
ファイル共有の解除, ix
ファイルシステムの共有解除
unshareall コマンド, 61
ファイルシステムの共有の解除
unshare コマンド, 60
ファイル属性と NFS バージョン 3, 6
ファイル転送サイズ
ネゴシエーション, 69
ファイルとファイルシステム, ix
NFS での扱い, 4
NFS の ASCII ファイルとその機能, 43, 44
autofs アクセス, 88, 89
autofs によるファイルの選択, 119, 121
自動共有, 14, 15
デフォルトのファイルシステムタイプ, 44
ファイルシステムの定義, 4
プロジェクト関連ファイルの統合, 92, 94
リモートファイルシステム, 44, 54, 61
ローカルファイルシステム, 44, 54
ファイルの自動共有, 14, 15
ブート
ディスクレスクライアントのセキュリティ, 79
ファイルシステムのマウント, 16
フォアグラウンドマウントオプション, 49
複数サーバ間での共有ファイルの複製, 96
複製されたファイルシステム, 72
複製マウント
soft オプションと, 40
プロトコルのバージョン, 40
読み取り専用でのマウント, 40
プログラム、ハングした, 40
プロジェクト関連ファイルの統合, 92, 94
プロジェクト、ファイルの統合, 92, 94

へ

ベリファイア
UNIX 認証, 77
説明, 76

ほ

ポートマッパー
マウント, 70
ホスト、全ファイルシステムのアンマウント, 54
ポンド記号 (#)
間接マップのコメント, 113
直接マップのコメント, 111
マスタマップのコメント (auto_master), 108

ま

マウント, ix
NFS URL を使用した, 21
NFS ファイルシステムのオプション, 49
autofs, 118

autofs と, 9, 10, 17
nfsd デーモン, 70
アクセスの無効化, 19
公共ファイルハンドル, 70
サーバが応答しない, 51
手動 (即時), 16
ソフトとハード, 28
ディスクレスクライアントでの必要条件, 9
テーブル内のすべてのファイルシステム, 53
中のキーボード割り込み, 28
バックグラウンドでの再試行, 49
ファイアウォールを越えた, 20
ブート時, 16
フォアグラウンドでの再試行, 49
ポートマッパー, 70
マウントされているファイルシステムのリスト, 43
マウント済みのファイルシステムに対するオーバーレイ, 51
読み書き可能の指定, 50
読み取り専用の指定, 50, 51
リモートマウント, 28, 29, 34
例, 51, 53
マウント済みのファイルシステムに対するオーバーレイ, 51
マウントに対するキーボードからの割り込み, 28
マウントポイント
fuser -k, 54
/home, 107, 108
/net, 109
重複の回避, 88
マスタマップのマウントポイント /-, 107, 111
マウントポイント /home, 107, 108
マウントポイント /net
アクセス方法, 109
説明, 109
マスタマップ (auto_master)
Secure NFS の設定, 23
automount コマンド実行の判断, 86
/etc/mnttab ファイルとの比較, 115
インストール済み, 90
概要, 107, 108
構文, 107
コメント, 108
セキュリティ制約条件, 96
説明, 85
内容, 107, 109
変更, 86
マウントポイント /-, 107, 111
優先されるオプション, 90
マップ (autofs), ix
-hosts 特殊マップ, 108
autofs のデフォルトの動作, 126, 127
automount コマンド、実行の判断, 85
間接, 112, 114
管理作業, 85
管理タスク, 127
クライアントに対する読み取り専用ファイルの選択, 121
クライアントのための読み取り専用ファイルの選択, 119
コメント, 108, 111, 113
実行可能, 125
タイプと用途, 85
他のマップの参照, 123, 125
探索プロセスの開始, 109, 116
直接, 110, 111
特殊文字, 129
長い行の分割, 108, 111, 113
ネットワーク探索, 116
複数マウント, 118
変更, 85 - 87, 126
変数, 122, 123
保守方法, 85
マウントの重複の回避, 88
マスタ, 107
マップエントリに使われる変数, 122, 123
マップでのサーバの重み付け, 122
マップによる探索
プロセスの開始, 116
マップの中のアスタリスク (*), 129
マップの中のアンパサンド (&), 128
マップの中の特殊文字, 129
マップ名の中の +, 125
マップ名の中のダッシュ (-), 123
マップ名の中のプラス記号 (+), 123, 125
マップを使った探索
概要, 116
プロセスの開始, 109

よ
読み書き可能形式
ファイルシステムの共有, 55, 58
ファイルシステムのマウント, 50
読み取り専用形式
ファイルシステムの共有, 55, 58
ファイルシステムのマウント, 50, 51
読み取り専用タイプ
autofs によるファイル選択, 121
autofs によるファイルの選択, 119

り
リスト
リモートマウントされたファイルシステム, 44
リソース、共有される, 44
リモートファイルシステム
default types, 44
グループのアンマウント, 54
リモートマウントされたファイルシステムのリスト, 44
リモートマウントされたファイルシステムを持つクライアントの一覧表示, 61
リモートマウント
障害追跡, 29, 34
必要なデーモン, 28

ろ
ローカルキャッシュと NFS バージョン 3, 6
ローカルファイルシステム
グループのアンマウント, 54
デフォルトのファイルシステムタイプ, 44
ロック
削除, 48
ロック、NFS バージョン 3 の改良点, 7

わ
ワームスタート
rpcbind サービス, 35

ネ
ネゴシエーション
ファイル転送サイズ, 69

マ
マップのなかのバックスラッシュ (\), 111
マップのなかのバックスラッシュ(\), 111
マップの中のバックスラッシュ (\), 108, 113